UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

LICENCIATURA EN CIENCIAS COMPUTACIONALES

TESIS

Gasly LMS: Plataforma de creación de cursos en línea

Para obtener el título de

Licenciado en Ciencias Computacionales

PRESENTA

OMAR MORALES GONZALEZ

Director (a)

Mtra. Norma Laura Salazar Viveros

Comité tutorial

Dra. Anilu Franco Arcega
Dra. Verónica Martínez Lazcano
Dra. Theira Irasema Samperio Monrroy

Pachuca de Soto, Hidalgo, 01 Junio de 2026.

Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniería
School of Engineering and Basic Sciences
Dirección
Office of the Director

Mineral de la Reforma, Hgo., a 22 de mayo de 2026

Número de control: ICBI-D/984/2026
Asunto: Autorización de impresión.

MTRA. OJUKY DEL ROCÍO ISLAS MALDONADO
DIRECTORA DE ADMINISTRACIÓN ESCOLAR DE LA UAEH

Con Título Quinto, Capítulo II, Capítulo V, Artículo 51 Fracción IX del Estatuto General de nuestra Institución, por este medio, le comunico que el Jurado asignado al egresado de la Licenciatura en Ciencias Computacionales **Omar Morales González,** quien presenta el trabajo de titulación **"GASLY LMS: Plataforma de creación de cursos en línea",** ha decidido, después de revisar fundamento en lo dispuesto en el Título Tercero, Capítulo I, Artículo 18 Fracción IV; dicho trabajo en la reunión de sinodales, **autorizar la impresión del mismo**, una vez realizadas las correcciones acordadas.

A continuación, firman de conformidad los integrantes del Jurado:

Presidente: Dra. Verónica Martínez Lazcano

Secretario: Dra. Anilú Franco Árcega

Vocal: M.I.D. Norma Laura Salazar Viveros

Suplente: Dra. Theira Irasema Samperio Monroy

Sin otro particular por el momento, reciba un cordial saludo.

Atentamente
"Amor, Orden y Progreso"

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO
ICBI
DIRECCIÓN

Dr. Gabriel Vergara Rodríguez
Director del ICBI

GVR/MMM

Ciudad del Conocimiento, Carretera Pachuca-Tulancingo
Km. 4.5 Colonia Carboneras, Mineral de la Reforma, Hidalgo,
México. C.P 42184
Teléfono: 52 (771) 71 720 00 Ext. 40001
direccion_icbi@uaeh.edu.mx, vergarar@uaeh.edu.mx

Resumen

El presente trabajo desarrolla Gasly LMS, una plataforma web orientada a la creación, organización y administración de cursos en línea. La propuesta surge a partir de la necesidad de contar con una herramienta que permita gestionar contenidos educativos, recursos complementarios, evaluaciones, tareas y seguimiento del progreso académico dentro de un entorno digital accesible. Para su desarrollo se analizaron conceptos relacionados con educación en línea, e-learning, sistemas de gestión del aprendizaje, usabilidad y tecnologías web, así como distintas metodologías de diseño para aplicaciones hipermedia y sitios web.

El sistema fue desarrollado mediante tecnologías como Next.js, React, Tailwind CSS, Prisma y Auth.js, integrando perfiles de usuario para estudiantes, instructores y administradores. La metodología WSDM fue utilizada como guía para el análisis de usuarios, el diseño conceptual, el diseño de implementación y la construcción técnica de la plataforma. Entre las principales funciones implementadas se encuentran la creación y administración de cursos, organización por capítulos, carga de materiales, evaluaciones de opción múltiple, gestión de tareas, autenticación por roles, seguimiento del avance y comunicación básica mediante chat.

Para validar la usabilidad del sistema se aplicó la escala *System Usability Scale* (SUS) a 34 participantes. El resultado promedio obtenido fue de 75.59 sobre 100, lo que indica una percepción favorable respecto a la facilidad de uso, consistencia, claridad e interacción con la plataforma. Los resultados obtenidos permiten considerar a Gasly LMS como una propuesta funcional y viable para apoyar procesos de formación en línea, con posibilidades de crecimiento mediante futuras mejoras orientadas a comunicación, analítica, evaluación avanzada y experiencia móvil.

Palabras clave: sistema de gestión del aprendizaje, educación en línea, WSDM, usabilidad, plataforma web.

Abstract

This work presents the development of Gasly LMS, a web-based platform aimed at creating, organizing, and managing online courses. The proposal arises from the need to develop a tool that enables the administration of educational content, complementary resources, assessments, assignments, and academic progress tracking within an accessible digital environment. For its development, concepts related to online education, e-learning, learning management systems, usability, and web technologies were analyzed, along with different design methodologies for hypermedia applications and websites.

The system was developed using technologies such as Next.js, React, Tailwind CSS, Prisma, Auth.js, and MySQL, integrating user profiles for students, instructors, and administrators. The WSDM methodology was used as a guide for user analysis, conceptual design, implementation design, and technical construction of the platform. The main implemented features include course creation and administration, chapter-based organization, material uploading, multiple-choice assessments, assignment management, role-based authentication, academic progress tracking, and basic communication through chat.

To validate the usability of the system, the System Usability Scale (SUS) was applied to 34 participants. The average score obtained was 75.59 out of 100, which indicates a favorable perception regarding ease of use, consistency, clarity, and interaction with the platform. The results obtained allow Gasly LMS to be considered a functional and viable proposal to support online training processes, with opportunities for future improvement in communication, analytics, advanced assessment, and mobile experience.

Keywords: learning management system, online education, WSDM, usability, web platform.

Índice

Introducción

El sistema educativo ha experimentado transformaciones significativas en las últimas décadas, aunque aún persisten modelos tradicionales apoyados en aulas físicas y materiales impresos. La incorporación de tecnologías digitales ha modificado la forma en que las personas acceden, consultan y utilizan la información, generando nuevas posibilidades para la enseñanza, la capacitación y la distribución de contenidos educativos mediante herramientas tecnológicas.

La adopción de dispositivos electrónicos y plataformas digitales ha aumentado de manera considerable, ofreciendo alternativas flexibles para almacenar, organizar y consultar información de forma centralizada. Este cambio representa una oportunidad para optimizar el acceso a los recursos formativos, reducir la dependencia de materiales físicos y ampliar las posibilidades de aprendizaje en distintos entornos.

La educación a distancia ha cobrado una importancia significativa en los últimos años, especialmente a partir de la pandemia de la COVID-19, periodo en el que instituciones públicas y privadas tuvieron que adaptarse con mayor rapidez al uso de entornos virtuales. Esta situación modificó la percepción de la educación en línea, evidenciando la necesidad de contar con herramientas capaces de facilitar la continuidad de los procesos formativos sin depender exclusivamente de la asistencia presencial.

La transformación de los procesos educativos y de capacitación ha generado la necesidad de contar con plataformas digitales que no solo permitan publicar contenidos, sino que también se ajusten a las demandas actuales de accesibilidad, flexibilidad, organización y seguimiento. En este sentido, los sistemas de gestión del aprendizaje, conocidos como LMS por sus siglas en inglés (*Learning Management System*), permiten administrar cursos, usuarios, materiales, actividades y evaluaciones dentro de un mismo entorno digital, facilitando tanto la distribución del contenido como el acompañamiento del proceso de aprendizaje (Vidal Ledo et al., 2014).

A partir de lo anterior, el presente trabajo propone el desarrollo de Gasly LMS, un sistema web orientado a la creación, organización y administración de cursos digitales. La plataforma integra contenidos educativos, recursos complementarios, actividades, evaluaciones y

mecanismos de seguimiento en un entorno accesible desde la web, con el propósito de apoyar procesos de capacitación a distancia de manera estructurada.

La creación de este sistema de gestión del aprendizaje surge como respuesta a la necesidad identificada en un organismo público de radiodifusión perteneciente a una entidad federativa de México, el cual requería una herramienta para gestionar y distribuir material de capacitación previamente elaborado, así como recursos de apoyo dirigidos a grupos de interés específicos. Con ello, el proyecto contribuye al uso de tecnologías web en entornos formativos institucionales, mediante una solución flexible, accesible y acorde con las necesidades actuales de formación en línea.

Justificación

La capacitación constante y la actualización de conocimientos constituyen elementos relevantes para organizaciones, instituciones educativas, áreas de formación, instructores independientes, docentes y grupos que requieren transmitir información de manera ordenada y accesible. En distintos contextos, la formación continua permite fortalecer habilidades, actualizar procedimientos, compartir información especializada y mejorar el desempeño de quienes participan en procesos educativos o de capacitación. Esta situación adquiere mayor importancia cuando se incorporan nuevos participantes de manera periódica, por lo que resulta necesario contar con mecanismos que faciliten su integración, orientación y acceso a los conocimientos necesarios para cumplir sus actividades o avanzar en su aprendizaje.

Los esquemas tradicionales de formación pueden presentar limitaciones cuando los participantes se encuentran en distintas áreas, municipios, ciudades o entidades federativas, o cuando no es posible reunirlos en un mismo espacio físico y horario. Estas condiciones dificultan la continuidad de las actividades, especialmente ante restricciones de tiempo, distancia o disponibilidad. Por ello, el uso de herramientas digitales permite reducir la dependencia de la presencialidad y favorecer el acceso a materiales desde diferentes ubicaciones.

El uso de medios digitales también puede contribuir a disminuir recursos asociados con la capacitación presencial, como traslados, impresión de materiales y uso de espacios físicos. Si bien el aprendizaje digital genera una huella ambiental, diversos estudios señalan que esta

puede ser menor que la asociada a modelos presenciales, principalmente por la reducción de desplazamientos y del uso de infraestructura física (Davies et al., 2024).

Una plataforma de cursos en línea no debe limitarse a alojar materiales, sino que también puede favorecer la interacción y el aprendizaje colaborativo. La incorporación de actividades, evaluaciones y recursos de comunicación permite fortalecer la participación de los usuarios, promover el intercambio de ideas y generar espacios de apoyo entre quienes forman parte del proceso. En este sentido, el aprendizaje colaborativo en línea representa una alternativa para construir conocimiento mediante la interacción entre participantes, especialmente cuando se integra en entornos digitales estructurados (Roberts, 2004).

Aunque existen sistemas de gestión del aprendizaje ampliamente utilizados, muchos incorporan una gran cantidad de funciones, secciones y configuraciones. Esto puede ser útil en entornos con personal especializado, pero también puede dificultar la administración para usuarios con menor familiaridad tecnológica. En estos casos, crear cursos, organizar materiales, configurar actividades o revisar el avance de los participantes puede convertirse en un proceso confuso o poco práctico.

Ante este panorama, Gasly LMS plantea una solución enfocada en la administración de cursos en línea mediante una interfaz clara y funcional. Aunque la situación inicial fue identificada en un organismo público de radiodifusión perteneciente a una entidad federativa de México, su desarrollo no se limita a dicho contexto, ya que puede adaptarse a distintas organizaciones, instituciones educativas, áreas de capacitación, instructores o grupos formativos que requieran estructurar cursos digitales.

La implementación de este sistema permite que los responsables de la formación conserven mayor control sobre sus materiales, al poder administrarlos, actualizarlos y distribuirlos desde un mismo entorno digital. Esto favorece la continuidad de las actividades, reduce la dependencia de recursos físicos y facilita que los usuarios consulten la información de acuerdo con sus necesidades, disponibilidad y ritmo de avance.

Además del desarrollo técnico, el proyecto adquiere relevancia metodológica al considerar la evaluación de la usabilidad mediante la escala *System Usability Scale* (SUS). Esta validación permite conocer la percepción de los usuarios respecto a la facilidad de uso,

claridad e interacción con la plataforma, aportando una base objetiva para valorar el sistema y orientar futuras mejoras.

De esta forma, Gasly LMS contribuye a la creación de cursos en línea mediante una propuesta flexible, accesible y adaptable. Su importancia no radica únicamente en la construcción de una herramienta tecnológica, sino en la posibilidad de reducir barreras de acceso y evaluar la experiencia de uso desde la perspectiva de los usuarios finales.

Antecedentes

Previo al desarrollo de Gasly LMS, el contexto que dio origen al proyecto se caracterizaba por la ausencia de una plataforma propia para gestionar cursos, materiales didácticos, recursos de apoyo y seguimiento del proceso de aprendizaje. La formación del personal se realizaba principalmente de manera presencial, lo que implicaba que los participantes tuvieran que trasladarse a una sede determinada para recibir capacitación sobre procesos actualizados o de reciente incorporación. Aunque este modelo permitía la transmisión directa de conocimientos, también presentaba limitaciones relacionadas con la disponibilidad de horarios, los tiempos de traslado y la coordinación necesaria para reunir a los participantes en un mismo espacio, sin afectar el desarrollo de sus actividades habituales. Esta situación representaba un reto particular cuando los equipos de trabajo se encontraban distribuidos en distintos municipios, áreas operativas o sedes, ya que la asistencia a capacitaciones presenciales podía generar inversión adicional de tiempo, gastos de transportación y ajustes en la organización interna de las actividades. Además, cuando el personal debía suspender temporalmente sus funciones para acudir a una sesión, existía el riesgo de afectar la continuidad de tareas ordinarias o de procesos que requerían atención constante.

Otro aspecto relevante se relacionaba con los tiempos administrativos propios de algunas instituciones públicas. La adquisición, contratación o implementación de soluciones externas podía requerir licitaciones, convenios, autorizaciones o trámites internos que no siempre permitían responder con premura a los requerimientos de formación. En consecuencia, cuando un procedimiento, lineamiento o material debía difundirse en un periodo específico, los retrasos podían ocasionar que la información perdiera vigencia o que su distribución ocurriera cuando el contenido ya necesitaba ser actualizado.

También se identificó que algunas plataformas de gestión del aprendizaje disponibles en el mercado, aunque amplias en funcionalidades, podían resultar poco prácticas para usuarios sin experiencia en la administración de entornos virtuales. En ciertos casos, la cantidad de secciones, configuraciones y herramientas podía dificultar tareas básicas como crear un curso, organizar materiales, agregar recursos o revisar el avance de los participantes. Esta condición evidenció la conveniencia de contar con una solución más clara, enfocada en funciones esenciales y adecuada para personas con distintos niveles de familiaridad tecnológica.

A partir de estos antecedentes, surgió la propuesta de desarrollar Gasly LMS como una plataforma de creación de cursos en línea orientada a facilitar la administración de contenidos educativos, recursos complementarios, evaluaciones y seguimiento del aprendizaje. Si bien el punto de partida fue una problemática identificada en un organismo público, el sistema fue planteado con un alcance más amplio, de manera que pudiera adaptarse a instituciones educativas, áreas de capacitación, instructores independientes, docentes o grupos que requieran organizar cursos digitales de forma estructurada.

Objetivo general

Desarrollar un sistema de gestión del aprendizaje accesible mediante tecnologías web para la creación de cursos con temas específicos a través de contenido audiovisual que incluya aspectos teóricos y prácticos.

Objetivos específicos

- Analizar los requerimientos generales de un sistema de gestión del aprendizaje a través de la comparación de sistemas existentes para identificar las funcionalidades necesarias en la creación, administración y consulta de cursos en línea.
- Diseñar una interfaz de usuario clara, intuitiva y funcional con base en criterios de usabilidad y experiencia de usuario que faciliten la navegación de los usuarios finales dentro de la plataforma.
- Modelar la estructura de datos del sistema mediante el diseño de diagramas de entidad-relación para definir las entidades, relaciones y atributos necesarios en la gestión de usuarios, cursos, capítulos, recursos, evaluaciones y progreso académico.

- Implementar la base de datos haciendo uso de Prisma para almacenar, consultar y administrar la información utilizada por la plataforma.
- Elaborar el sistema de administración mediante el desarrollo de módulos funcionales para gestionar la creación de cursos, capítulos, materiales, tareas, evaluaciones y usuarios desde una interfaz web.
- Aplicar mecanismos de seguridad basados en autenticación, roles de usuario, cifrado de contraseñas y manejo de tokens para proteger la información de los usuarios dentro del sistema.
- Integrar herramientas de evaluación dentro de los cursos a través de la creación de módulos interactivos para reforzar el aprendizaje de los usuarios mediante actividades asociadas al contenido.
- Incorporar mecanismos de seguimiento del progreso académico por medio del registro de capítulos completados, evaluaciones realizadas y avance por curso para conocer el desempeño de los usuarios dentro de la plataforma.
- Realizar pruebas piloto con un grupo seleccionado de usuarios a partir de la interacción directa con la plataforma para identificar errores que corregir y mejoras que aplicar antes de la implementación final.

Alcances

El desarrollo de Gasly LMS permite la construcción de una versión funcional de una plataforma web que integra las funciones necesarias para crear y estructurar cursos, así como organizar contenidos por capítulos. De igual forma, el sistema facilita la incorporación de materiales de apoyo y texto, la gestión de tareas, la aplicación de evaluaciones y el seguimiento del avance de los usuarios dentro de la propia plataforma.

El alcance del proyecto comprende también la implementación de una interfaz diseñada para distintos perfiles de usuario, considerando los roles de estudiante, instructor y administrador. Asimismo, se incluyen mecanismos de autenticación, control de acceso y protección de la información para garantizar la seguridad del entorno.

De esta manera, los alcances establecidos reflejan las funcionalidades desarrolladas en esta etapa y se describen a continuación:

- Creación y administración de cursos en línea desde un entorno web, considerando distintos perfiles de usuario como estudiantes, instructores y administradores.
- Organización del contenido mediante capítulos y secciones, permitiendo estructurar los cursos de forma similar a un temario, con capítulos numerados, títulos específicos y apartados internos para distribuir el material educativo.
- Incorporación de contenido audiovisual dentro de los cursos o capítulos, facilitando la presentación de materiales de apoyo para el aprendizaje.
- Carga de archivos adjuntos en distintos niveles del curso, ya sea como recursos generales o como materiales asociados a capítulos específicos, incluyendo documentos PDF y otros archivos complementarios.
- Gestión de tareas asociadas a los cursos, considerando fechas de entrega, visualización del tiempo restante para su cumplimiento y registro de entregas tardías.
- Interfaz clara, intuitiva y funcional para facilitar la navegación de estudiantes, instructores y administradores dentro del entorno web.
- Módulo de administración del instructor para gestionar cursos, capítulos, materiales, tareas y evaluaciones desde una interfaz web.
- Módulo de administración general para gestionar usuarios, crear y actualizar cuentas, consultar los cursos generados por instructores y supervisar su actividad dentro de la plataforma.
- Acceso administrativo a funciones de instructor, permitiendo que el administrador intervenga en la gestión de cursos, capítulos, materiales, tareas y evaluaciones cuando sea necesario.
- Evaluaciones con preguntas de opción múltiple asociadas al contenido de los cursos o capítulos, orientadas a reforzar el aprendizaje y determinar el cumplimiento de actividades necesarias para el avance dentro del curso.

- Seguimiento del progreso académico mediante el registro de capítulos completados, evaluaciones realizadas y avance por curso.
- Mecanismos de seguridad mediante autenticación, roles de usuario, cifrado de contraseñas y manejo de tokens para proteger la información de los usuarios.
- Módulo básico de chat para permitir la comunicación entre estudiantes e instructores vinculados a un curso.

Limitaciones

Las limitaciones del presente proyecto permiten identificar los aspectos técnicos, funcionales y operativos que no forman parte de la versión actual de Gasly LMS, así como las condiciones externas que pueden influir en su funcionamiento. Estas delimitaciones no impiden el uso de la plataforma, pero establecen el marco bajo el cual debe entenderse el alcance real del sistema desarrollado.

- La plataforma depende de la disponibilidad de conexión a Internet, por lo que las fallas de conectividad pueden afectar el acceso de los usuarios y la experiencia general de aprendizaje.
- La experiencia de uso está orientada principalmente a equipos de cómputo, como laptops y computadoras de escritorio. Si bien la plataforma puede consultarse desde dispositivos móviles, algunas funciones de administración, edición y seguimiento académico pueden resultar más cómodas en pantallas amplias, por lo que una optimización específica para teléfonos inteligentes se contempla como mejora posterior.
- Al tratarse de una nueva implementación, los estudiantes, instructores y administradores pueden enfrentar una curva de aprendizaje inicial durante el uso de la plataforma.
- El contenido de los cursos se gestiona principalmente en un idioma, por lo que la disponibilidad multilingüe no forma parte de esta versión.

- El sistema se encuentra sujeto a las tecnologías web utilizadas durante su desarrollo, por lo que futuras actualizaciones de dependencias, librerías o servicios externos podrían requerir ajustes técnicos para mantener su funcionamiento.
- La descarga o resguardo externo de videos puede estar limitada por criterios de control del contenido, derechos de autor o propiedad intelectual definidos por quienes administren los cursos.
- La gestión de grandes cantidades de videos o archivos multimedia puede requerir estrategias adicionales de almacenamiento, optimización y distribución para mantener un acceso eficiente al material educativo.
- El módulo de chat permite una comunicación básica entre estudiantes e instructores vinculados a un curso; sin embargo, en esta versión no se implementa mediante WebSockets, por lo que no cuenta con actualización en tiempo real persistente.
- Las evaluaciones implementadas se limitan a preguntas de opción múltiple, por lo que otros tipos de reactivos, como preguntas abiertas, verdadero/falso, relación de columnas o ejercicios interactivos, no forman parte de esta etapa.
- Los foros de discusión segmentados por grupo, clase o cohorte no forman parte de la versión actual del sistema.
- Aunque la plataforma permite gestionar tareas con fechas de entrega, no cuenta con una agenda o calendario académico global que concentre actividades, evaluaciones, recordatorios y eventos relevantes.
- La versión desarrollada no incluye herramientas de videoconferencia integradas dentro de la plataforma.
- Las rúbricas de evaluación avanzadas, los certificados de finalización, la analítica académica detallada y las recomendaciones personalizadas de aprendizaje quedan fuera del alcance de esta etapa.

Capítulo 1. Marco teórico

La construcción de una plataforma digital requiere una base de referencia que permita comprender el contexto en el que se desarrolla la propuesta, así como los elementos que intervienen en su diseño, funcionamiento y validación. Por ello, resulta necesario reunir fundamentos que expliquen tanto la dimensión formativa del proyecto como los aspectos técnicos y metodológicos que permiten su implementación.

La revisión de estos elementos permite relacionar el problema identificado con los recursos empleados para su atención, considerando tanto las necesidades de los usuarios como las decisiones tomadas durante el desarrollo del sistema. Con ello, se establece una base para comprender la propuesta desde los fundamentos que la sustentan y su relación con el desarrollo de la plataforma.

1.1 Marco conceptual

La formulación de propuestas dentro del campo educativo requiere una base conceptual que permita comprender los componentes principales que intervienen en los procesos de enseñanza y aprendizaje en entornos digitales. En este contexto, es necesario analizar los conceptos necesarios para la interpretación de la educación a distancia, *e-Learning* y sistemas que ayuden a organizar experiencias formativas basadas en plataformas digitales.

En esta dirección, a continuación, se recopilan los conceptos que permiten interpretar la función de los entornos virtuales de aprendizaje, sistemas de gestión del aprendizaje y tipos diferentes de cursos en línea. Además, se analizan los componentes necesarios relacionados con los roles de usuario, gestión de curso y contenido, evaluación del aprendizaje, seguimiento del progreso académico y usabilidad en plataformas educativas, ya que estos componentes son necesarios para el entorno en el cual está formulada la propuesta.

1.1.1 Educación en línea

La educación en línea es una modalidad de enseñanza-aprendizaje que se desarrolla mediante el uso de tecnologías digitales y redes de comunicación, permitiendo que los procesos formativos puedan llevarse a cabo sin requerir la coincidencia física entre docentes y

estudiantes. Esta modalidad ha adquirido relevancia por su capacidad para ampliar el acceso a la educación, flexibilizar los tiempos de estudio y facilitar la distribución de contenidos en distintos contextos, mediante entornos apoyados en recursos digitales e Internet. (UNESCO, 2024)

A través de la educación en línea, los estudiantes pueden acceder a materiales didácticos, actividades, evaluaciones y espacios de interacción desde dispositivos con conexión a Internet. De esta forma, se favorecen procesos de aprendizaje más accesibles y adaptables a las necesidades de los usuarios, al tiempo que se impulsa el uso de plataformas digitales que permiten organizar y dar seguimiento a las actividades académicas de manera estructurada. (Sánchez Rodríguez, 2009)

1.1.2 e-Learning

El e-Learning, también denominado aprendizaje electrónico, es una modalidad de enseñanza y formación apoyada en medios digitales y tecnologías de la información y la comunicación. Su finalidad es facilitar el acceso a contenidos, recursos y actividades formativas mediante dispositivos electrónicos, principalmente a través de Internet. El término proviene del inglés *electronic learning* y hace referencia al aprendizaje desarrollado por medio de recursos digitales conectados a la red (Santander Open Academy, 2022).

Entre sus principales características se encuentran la flexibilidad en el acceso a los materiales, el uso de plataformas digitales para organizar cursos y recursos, la existencia de canales de comunicación entre docentes y estudiantes, la interactividad mediante actividades digitales y la posibilidad de dar seguimiento al progreso académico. Estos elementos permiten que el proceso formativo pueda adaptarse a distintos ritmos, necesidades y contextos de aprendizaje, fortaleciendo la experiencia del usuario dentro de entornos virtuales (López Rodrigo et al., 2011).

El e-Learning puede presentarse en distintas modalidades, como el aprendizaje síncrono, cuando la interacción ocurre en tiempo real; el aprendizaje asíncrono, cuando el estudiante accede a los contenidos según su propio ritmo; el b-learning o aprendizaje combinado, que integra actividades presenciales con recursos digitales; el m-learning, desarrollado mediante dispositivos móviles; y el microlearning, basado en contenidos breves y focalizados. Estas

modalidades permiten ampliar las posibilidades de formación, reducir barreras geográficas y favorecer una experiencia más flexible y personalizada (González et al., 2017).

1.1.3 Entornos virtuales de aprendizaje

Un entorno virtual de aprendizaje es un espacio digital diseñado para apoyar los procesos de enseñanza y aprendizaje mediante herramientas tecnológicas. En estos entornos, los usuarios pueden acceder a contenidos, participar en actividades, comunicarse con otros participantes y desarrollar experiencias formativas en línea. Su finalidad es proporcionar una estructura organizada que facilite la interacción educativa en contextos no presenciales o mixtos, permitiendo que los recursos y actividades se integren dentro de una misma plataforma (Barrera et al., 2016).

Los entornos virtuales de aprendizaje integran recursos como materiales multimedia, tareas, foros, evaluaciones y mecanismos de seguimiento del progreso. Gracias a estas características, permiten organizar la experiencia educativa de manera flexible y dinámica, además de constituir un apoyo importante para docentes e instituciones en la administración de cursos y supervisión del desempeño estudiantil. En el caso de propuestas como Gasly LMS, este concepto permite comprender la importancia de concentrar contenidos, actividades y seguimiento dentro de un entorno digital accesible para distintos perfiles de usuario (Sánchez Rodríguez, 2009).

1.1.4 Sistema de gestión del aprendizaje (LMS)

Un sistema de gestión del aprendizaje, conocido como LMS por sus siglas en inglés (*Learning Management System*), es una plataforma digital diseñada para administrar, distribuir y supervisar procesos de enseñanza y aprendizaje. Su función principal consiste en centralizar cursos, contenidos, usuarios, actividades, evaluaciones y reportes académicos dentro de un mismo entorno, permitiendo que la experiencia formativa se organice de manera estructurada y accesible (Vidal Ledo et al., 2014).

Los LMS permiten que docentes, instituciones o responsables de formación organicen recursos educativos, definan actividades, integren evaluaciones y den seguimiento al avance de los participantes. Para los estudiantes, estas plataformas facilitan el acceso a materiales, la realización de actividades y la consulta de su progreso académico. Por estas características,

los sistemas de gestión del aprendizaje se han consolidado como herramientas relevantes dentro de la educación en línea y la capacitación digital (Sánchez Rodríguez, 2009).

1.1.5 Tipos de cursos en línea

En el ámbito de la educación digital existen distintos tipos de cursos en línea, los cuales responden a necesidades particulares de acceso, duración, alcance, número de participantes y nivel de acompañamiento. Esta clasificación permite distinguir entre propuestas abiertas y masivas, cursos breves de actualización y experiencias formativas dirigidas a grupos más específicos. Entre las modalidades más reconocidas se encuentran los MOOC, NOOC y SPOC, cada una con características propias dentro de los entornos virtuales de aprendizaje (Sánchez Gordón, 2017).

1.1.5.1 MOOC

El término MOOC proviene de *Massive Open Online Course* y se refiere a cursos en línea de carácter masivo, abierto y accesible mediante Internet. Este tipo de cursos permite que una gran cantidad de participantes pueda acceder a contenidos formativos sin las limitaciones propias de un aula presencial, como el espacio físico o el número reducido de estudiantes. Su finalidad principal es ampliar las oportunidades de aprendizaje y favorecer el acceso al conocimiento en distintos contextos (Cabero-Almenara et al., 2014).

Los MOOC ofrecen flexibilidad en cuanto al tiempo, lugar y ritmo de estudio, ya que los participantes pueden revisar materiales, realizar actividades y acceder a evaluaciones dentro de plataformas digitales. Además, suelen integrar recursos multimedia, espacios de interacción y mecanismos de seguimiento que permiten organizar la experiencia formativa. Por estas características, se han consolidado como una modalidad relevante dentro de la educación abierta y en línea (Sánchez Gordón, 2017).

1.1.5.2 NOOC

Los NOOC, o *Nano Open Online Courses*, son cursos en línea abiertos de corta duración y enfoque temático específico. A diferencia de los MOOC, que suelen

abordar contenidos más amplios, los NOOC se orientan a objetivos de aprendizaje concretos, por lo que resultan útiles para procesos de actualización, capacitación breve o fortalecimiento de competencias puntuales (INTEF, s. f.).

Este tipo de curso permite que los participantes accedan a contenidos focalizados en un periodo reducido, favoreciendo una experiencia de aprendizaje ágil y flexible. Debido a su estructura breve, los NOOC pueden emplearse como complemento de programas formativos más amplios o como recurso independiente para atender necesidades específicas de aprendizaje (García-Peñalvo et al., 2017).

1.1.5.3 SPOC

El término SPOC proviene de *Small Private Online Course* y hace referencia a cursos en línea dirigidos a grupos reducidos y privados de participantes. A diferencia de los MOOC, cuyo propósito es alcanzar una audiencia masiva, los SPOC se enfocan en comunidades delimitadas, lo que permite mayor control sobre el proceso formativo y una atención más cercana a los estudiantes (Fox, 2013).

Los SPOC suelen utilizarse en contextos universitarios, empresariales o institucionales, ya que permiten combinar la flexibilidad de la educación en línea con un mayor nivel de personalización y acompañamiento. Esta modalidad facilita la adaptación de contenidos, actividades y evaluaciones a las necesidades de un grupo específico, lo que favorece procesos de aprendizaje más controlados y contextualizados (Kaplan & Haenlein, 2016).

1.1.6 Roles de usuario en un LMS

En un sistema de gestión del aprendizaje participan distintos roles de usuario, cada uno con funciones específicas dentro de la plataforma. Entre los más comunes se encuentran los administradores, instructores o docentes, y estudiantes. Los administradores se encargan de la configuración general del entorno y de la gestión de usuarios; los instructores crean cursos, publican materiales y administran actividades; mientras que los estudiantes acceden a los contenidos, realizan evaluaciones y consultan su avance académico (Instructure, s. f.).

La definición de roles permite organizar responsabilidades, delimitar permisos y proteger la información dentro del sistema. Esta separación resulta importante porque cada usuario debe

interactuar únicamente con las funciones relacionadas con sus actividades dentro del proceso formativo. De esta manera, los roles contribuyen al orden, la seguridad y la administración adecuada de un entorno de aprendizaje en línea (Moodle, s. f.).

1.1.7 Gestión de cursos y contenidos

La gestión de cursos y contenidos constituye una función central dentro de los sistemas de gestión del aprendizaje, ya que permite estructurar de manera organizada los elementos que forman parte de una experiencia educativa. Esta función comprende la creación, administración y publicación de cursos, así como la integración de materiales, actividades, evaluaciones y participantes dentro de un mismo entorno digital. Su propósito es facilitar que los contenidos se presenten de forma lógica, accesible y acorde con los objetivos de aprendizaje (Díaz Becerro, 2009).

Mediante esta funcionalidad, los docentes o instructores pueden dividir un curso en módulos, unidades, capítulos o temas, incorporar recursos de apoyo y definir actividades relacionadas con el contenido. La gestión no se limita a la publicación inicial de materiales, sino que también permite actualizarlos, reorganizarlos y mantenerlos vigentes conforme avanza el proceso formativo. Esto resulta relevante en entornos digitales, donde la flexibilidad y la actualización de la información forman parte de la dinámica educativa (Moodle, s. f.).

Una adecuada organización de cursos y contenidos favorece la experiencia de aprendizaje, ya que evita la dispersión de materiales y facilita el acceso a la información relevante. Cuando los recursos se presentan de manera estructurada, los usuarios pueden identificar con mayor claridad el orden de estudio, las actividades pendientes y los elementos necesarios para avanzar dentro del curso (Instructure, s. f.).

1.1.8 Evaluación del aprendizaje

La evaluación del aprendizaje es un proceso mediante el cual se recopila, interpreta y utiliza información para determinar el nivel de logro alcanzado por los estudiantes en relación con los objetivos establecidos. En entornos virtuales, este proceso puede realizarse mediante cuestionarios, tareas, exámenes, actividades prácticas y otros instrumentos que permiten valorar conocimientos, habilidades y competencias (Zapata-Ros, 2003).

Dentro de un LMS, la evaluación adquiere importancia porque permite documentar el desempeño del estudiante de manera sistemática. A través de estos mecanismos, los docentes pueden identificar avances, dificultades y áreas que requieren mayor atención. De igual forma, la plataforma puede facilitar la entrega de retroalimentación, elemento fundamental para fortalecer el aprendizaje y orientar al estudiante durante su proceso formativo (Black & Wiliam, 1998).

La evaluación no debe entenderse únicamente como un medio para medir resultados finales, sino también como una herramienta de apoyo para mejorar el proceso de enseñanza y aprendizaje. La información obtenida mediante actividades evaluativas permite ajustar estrategias, reforzar contenidos y tomar decisiones académicas con mayor fundamento (Brookhart, 2017).

1.1.9 Seguimiento del progreso académico

El seguimiento del progreso académico consiste en la observación y registro continuo del avance del estudiante dentro de un proceso de aprendizaje. Esta funcionalidad permite conocer el cumplimiento de actividades, el acceso a contenidos, los resultados de evaluaciones y el nivel de participación dentro de un curso. En plataformas digitales, este seguimiento resulta útil para identificar el desempeño de los usuarios y reconocer posibles áreas de atención (Zapata-Ros, 2003).

Los sistemas de gestión del aprendizaje facilitan este proceso al registrar información relacionada con capítulos completados, actividades realizadas, evaluaciones presentadas y avance general dentro de los cursos. Esta información puede ser consultada tanto por docentes como por estudiantes, lo que favorece la supervisión del aprendizaje y permite tomar decisiones oportunas para reforzar contenidos o acompañar a quienes presentan rezagos (Instructure, s. f.).

1.1.10 Usabilidad en plataformas educativas

La usabilidad se refiere al grado en que un sistema puede ser utilizado por usuarios específicos para alcanzar objetivos determinados de manera efectiva, eficiente y satisfactoria dentro de un contexto de uso. En plataformas educativas, este concepto resulta especialmente importante, ya que docentes, estudiantes y administradores requieren interactuar con

herramientas digitales de forma clara y comprensible para realizar actividades académicas o de gestión (ISO, 2018).

En este sentido, las plataformas de aprendizaje en línea deben proporcionar navegación clara, acceso sencillo a la información y procesos comprensibles para los usuarios. Cuando las tareas dentro del sistema pueden realizarse sin dificultad, la experiencia mejora y se incrementa la posibilidad de que los participantes utilicen la plataforma de manera constante y efectiva (Lazar, 2006).

Por el contrario, una interfaz compleja, inconsistente o difícil de interpretar puede provocar frustración, errores de uso o abandono de la plataforma. Por ello, la usabilidad no solo se relaciona con la apariencia visual, sino también con la organización de la información, la claridad de las funciones y la facilidad para completar tareas dentro del sistema (Nielsen, 2012).

1.1.11 Software

El software comprende el conjunto de instrucciones, datos, estructuras y procedimientos que permiten a un sistema informático realizar funciones determinadas. A diferencia del hardware, que corresponde a los componentes físicos de un equipo, el software constituye la parte lógica que permite ejecutar tareas, procesar información y ofrecer funcionalidades al usuario. En este sentido, puede entenderse como el elemento intangible que da operación y propósito a los dispositivos computacionales (Pressman, 2010).

El término software tiene antecedentes vinculados con el desarrollo histórico de la informática. Su uso en el contexto computacional se atribuye comúnmente a John W. Tukey, quien lo empleó en una publicación de 1958 para diferenciar los programas y datos de los componentes físicos de una computadora. Asimismo, la idea de ejecutar instrucciones para realizar operaciones puede relacionarse con los primeros planteamientos de máquinas programables, como la máquina analítica propuesta por Charles Babbage, y con los fundamentos teóricos de la computación desarrollados por Alan Turing en su trabajo sobre los números computables (Computer History Museum, s. f.; Tukey, 1958; Turing, 1936).

Dentro de un sistema informático, el software puede adoptar distintas formas, como sistemas operativos, aplicaciones, programas de servicio, herramientas de desarrollo o sistemas web.

Su importancia radica en que permite transformar requerimientos y procesos en soluciones funcionales, capaces de atender necesidades específicas dentro de distintos contextos de uso (Sommerville, 2011).

1.1.12 Ingeniería de Software

La ingeniería de software es una disciplina orientada al desarrollo sistemático, controlado y mantenible de productos de software. Su propósito es aplicar principios, métodos, procesos y herramientas que permitan construir sistemas confiables, eficientes y acordes con las necesidades de los usuarios. Desde esta perspectiva, no se limita a la programación, sino que abarca actividades como análisis, diseño, implementación, pruebas, documentación y mantenimiento (Pressman, 2010).

En términos generales, la ingeniería de software busca producir soluciones que sean correctas, utilizables, mantenibles y económicamente viables. Para ello, se apoya en modelos de proceso y metodologías que permiten organizar el trabajo de desarrollo, reducir riesgos y mejorar la calidad del producto final. La selección de un enfoque adecuado depende del tipo de sistema, sus requerimientos, el contexto de aplicación y los recursos disponibles (Sommerville, 2011).

1.1.13 Lenguajes de programación

Un lenguaje de programación es un conjunto de reglas, símbolos y estructuras que permite expresar instrucciones para que una computadora pueda interpretarlas o ejecutarlas. Estos lenguajes proporcionan los medios necesarios para representar algoritmos, manipular datos y construir programas de manera organizada. Cada lenguaje cuenta con una sintaxis y una semántica propias, lo que determina la forma en que deben escribirse y entenderse sus instrucciones (Sebesta, 2012).

Los lenguajes de programación pueden clasificarse de distintas maneras, por ejemplo, según su nivel de abstracción, paradigma o propósito. Los lenguajes de bajo nivel se encuentran más cercanos a la arquitectura de la máquina, mientras que los de alto nivel facilitan la escritura de programas mediante estructuras más comprensibles para los desarrolladores. Esta clasificación permite seleccionar la tecnología más adecuada según las características del proyecto y el tipo de solución que se desea construir (Scott, 2016).

1.1.14 Sistema basado en la web

Un sistema basado en la web es una aplicación informática que opera a través de un navegador y permite el acceso a sus funciones mediante Internet o una red. Este tipo de sistema utiliza tecnologías propias del entorno web para presentar información, procesar solicitudes y permitir la interacción entre usuarios y servicios alojados en un servidor. Su funcionamiento se apoya en el intercambio de datos entre cliente y servidor, generalmente mediante protocolos como HTTP (MDN Web Docs, 2026).

Los sistemas web resultan relevantes porque permiten acceder a una aplicación desde distintos dispositivos sin requerir una instalación local compleja. Además, favorecen la actualización centralizada, la disponibilidad remota y la interacción con bases de datos, servicios externos y componentes dinámicos. Estas características los convierten en una opción adecuada para proyectos que requieren acceso distribuido y administración desde un entorno en línea (MDN Web Docs, 2025a).

1.1.15 Arquitectura cliente-servidor

La arquitectura cliente-servidor es un modelo de organización utilizado en sistemas informáticos y aplicaciones web, en el cual las responsabilidades se distribuyen entre dos componentes principales. El cliente solicita recursos o servicios, mientras que el servidor procesa dichas solicitudes, ejecuta la lógica correspondiente y devuelve una respuesta. En el contexto web, el cliente suele representarse por el navegador, mientras que el servidor aloja la aplicación y administra el acceso a la información (MDN Web Docs, 2025a).

Este modelo permite separar la presentación de la información, la lógica de procesamiento y la gestión de datos. Gracias a esta división, las aplicaciones pueden mejorar su mantenibilidad, seguridad y escalabilidad, ya que cada componente cumple una función específica dentro del sistema. En aplicaciones modernas, esta arquitectura facilita la comunicación entre interfaces, servidores, bases de datos y servicios externos (Kurose & Ross, 2021).

1.1.16 Frontend

El *frontend* corresponde a la parte visible de una aplicación web, es decir, a los elementos con los que el usuario interactúa directamente. Comprende la interfaz gráfica, los menús, botones, formularios, textos, imágenes y componentes visuales que conforman la experiencia de navegación. Su función principal es presentar la información de manera clara y permitir que las acciones realizadas por el usuario puedan ejecutarse de forma comprensible dentro del sistema (MDN Web Docs, s. f.).

Para su desarrollo se utilizan tecnologías como HTML, CSS y JavaScript. HTML permite estructurar el contenido de las páginas, CSS define la presentación visual y JavaScript agrega dinamismo e interacción. La combinación de estas tecnologías hace posible construir interfaces funcionales, adaptables y orientadas a mejorar la experiencia de uso en aplicaciones web (MDN Web Docs, s. f.).

1.1.17 Backend

El *backend* es la parte interna de una aplicación web encargada de procesar la lógica del sistema y gestionar las operaciones que no son visibles directamente para el usuario. Entre sus funciones se encuentran la validación de información, el manejo de solicitudes, la administración de sesiones, la comunicación con bases de datos y la ejecución de reglas necesarias para que la aplicación responda correctamente (IBM, s. f.).

A diferencia del frontend, que se enfoca en la interacción visual, el backend trabaja del lado del servidor. Su propósito es recibir las acciones realizadas desde la interfaz, procesarlas y devolver la información necesaria para su visualización o almacenamiento. Esta capa resulta fundamental en sistemas que requieren autenticación, control de permisos, consultas dinámicas y protección de datos (MDN Web Docs, s. f.).

En aplicaciones con múltiples usuarios, el backend también contribuye a mantener la estabilidad y escalabilidad del sistema. Gracias a esta capa, es posible atender varias solicitudes, administrar recursos compartidos y asegurar que las funciones principales se ejecuten de manera ordenada. Por ello, constituye un componente esencial para el funcionamiento de aplicaciones web dinámicas y seguras (IBM, s. f.).

1.1.18 Bases de datos

Las bases de datos son sistemas diseñados para almacenar, organizar, consultar y administrar información de manera estructurada. En las aplicaciones web resultan fundamentales porque permiten conservar datos relacionados con usuarios, cursos, contenidos, evaluaciones, inscripciones y configuraciones del sistema, manteniéndolos disponibles para su consulta y actualización posterior (Silberschatz et al., 2020).

Una característica importante de las bases de datos es la persistencia de la información, ya que los datos se mantienen aun cuando la aplicación se reinicie o el servidor deje de ejecutarse temporalmente. Esto permite que un sistema web conserve registros, avances, resultados y demás información necesaria para garantizar continuidad en su funcionamiento.

Existen distintos tipos de bases de datos, entre las cuales destacan las relacionales y las no relacionales. Las bases relacionales organizan la información en tablas compuestas por filas y columnas, estableciendo relaciones entre entidades definidas. Este modelo resulta adecuado para sistemas como Gasly LMS, donde es necesario relacionar usuarios, cursos, capítulos, recursos, evaluaciones y progreso académico de manera ordenada y consistente (Elmasri & Navathe, 2016).

1.1.19 Navegador web

Un navegador web es una aplicación que permite acceder, interpretar y visualizar recursos disponibles en la web. Su función consiste en solicitar documentos o servicios a través de una red, procesar el contenido recibido y presentarlo al usuario mediante una interfaz gráfica. De esta manera, actúa como intermediario entre el usuario y los sistemas web a los que se desea acceder (Mozilla, s. f.).

Además de mostrar páginas web, los navegadores permiten ejecutar código del lado del cliente, administrar pestañas, almacenar historial, gestionar permisos y aplicar medidas básicas de seguridad durante la navegación. Estas funciones los convierten en herramientas indispensables para interactuar con aplicaciones desarrolladas mediante tecnologías como HTML, CSS y JavaScript (MDN Web Docs, s. f.).

1.1.20 Protocolos de comunicación

Un protocolo de comunicación es un conjunto de reglas y estándares que permite el intercambio de datos entre distintos sistemas, dispositivos o aplicaciones. Estos protocolos establecen la forma en que la información debe transmitirse, interpretarse y responderse, permitiendo que los elementos conectados dentro de una red puedan comunicarse de manera ordenada, incluso cuando han sido desarrollados con tecnologías o plataformas diferentes (Kurose & Ross, 2021).

Los protocolos definen aspectos como el formato de los datos, la sincronización, la secuenciación, los mecanismos de control y los métodos para detectar o corregir errores durante la transmisión. Sin estas reglas, los dispositivos no podrían interpretar correctamente la información recibida, lo que dificultaría la comunicación entre sistemas. Su importancia radica en que permiten la interoperabilidad, la estandarización, la seguridad y la eficiencia en la transferencia de información dentro de redes informáticas (Tanenbaum & Wetherall, 2011).

Existen diversos protocolos utilizados en Internet, cada uno diseñado para cumplir una función específica. Entre los más comunes se encuentran FTP, empleado para la transferencia de archivos; SMTP, utilizado para el envío de correos electrónicos; DNS, encargado de traducir nombres de dominio en direcciones IP; y TCP/IP, que constituye la base de la comunicación en Internet al permitir el direccionamiento, enrutamiento y entrega confiable de datos entre dispositivos conectados (Kurose & Ross, 2021).

Entre estos protocolos, uno de los más importantes para el funcionamiento de la web es HTTP (*Hypertext Transfer Protocol*), ya que permite la transferencia de información entre clientes y servidores. Este protocolo opera bajo un modelo de solicitud y respuesta, en el cual el cliente, generalmente representado por un navegador web, envía una petición al servidor, y este procesa la solicitud para devolver el recurso correspondiente o un mensaje que indique el resultado de la operación (Fielding et al., 2022).

Los métodos HTTP permiten especificar la acción que se desea realizar sobre un recurso. Entre los más utilizados se encuentran GET, para solicitar información; POST, para enviar datos al servidor; PUT y PATCH, para actualizar recursos; DELETE, para eliminarlos;

HEAD, para solicitar únicamente los encabezados de una respuesta; y OPTIONS, para conocer las opciones de comunicación disponibles. Estos métodos son fundamentales para la interacción entre aplicaciones web, API y servicios distribuidos (MDN Web Docs, s. f.).

Los códigos de estado HTTP indican el resultado de una solicitud. Los códigos 1xx corresponden a respuestas informativas; los 2xx indican éxito; los 3xx señalan redirecciones; los 4xx representan errores del cliente; y los 5xx corresponden a errores del servidor. Además, los encabezados HTTP proporcionan información adicional sobre la solicitud o la respuesta, como el tipo de contenido, autenticación, caché, idioma, origen de la petición y otros metadatos relevantes para la comunicación (Fielding et al., 2022).

El funcionamiento de HTTP puede entenderse como un proceso en el que el cliente envía una solicitud, el servidor la procesa y posteriormente devuelve una respuesta. Esta respuesta puede contener una página web, un archivo, datos en formato JSON, una imagen o cualquier otro recurso solicitado. Gracias a este modelo, los sistemas web pueden mostrar contenido dinámico, consultar servicios externos y permitir la interacción entre el usuario y la aplicación (MDN Web Docs, s. f.).

El uso de HTTP es común en la navegación web, en la carga de recursos multimedia y en la comunicación mediante API REST. Sin embargo, para proteger la información transmitida, se emplea HTTPS, una versión segura que utiliza TLS para cifrar la comunicación entre cliente y servidor. De esta forma, se protege la privacidad, integridad y autenticidad de los datos intercambiados dentro de aplicaciones web (Rescorla, 2018).

1.1.21 Protocolo de seguridad SSL y TLS

SSL (*Secure Sockets Layer*) y TLS (*Transport Layer Security*) son protocolos criptográficos diseñados para proteger las comunicaciones realizadas a través de una red. Su función principal consiste en garantizar la privacidad, integridad y autenticación de los datos transmitidos entre dos sistemas, como un navegador y un servidor web. Aunque SSL fue ampliamente utilizado en los inicios de las conexiones seguras, actualmente TLS es el estándar recomendado para este tipo de comunicación (Rescorla, 2018).

SSL fue desarrollado por Netscape en la década de 1990 con el propósito de establecer conexiones cifradas entre clientes y servidores. A pesar de su relevancia histórica, sus

versiones antiguas presentaron vulnerabilidades importantes, por lo que dejaron de considerarse seguras. Debido a ello, TLS surgió como su evolución, incorporando mejoras en los algoritmos criptográficos, los procesos de autenticación y la eficiencia de la comunicación segura (Cloudflare, s. f.).

TLS permite establecer una conexión protegida mediante un proceso de negociación entre cliente y servidor. Durante esta etapa se intercambian parámetros criptográficos, se valida el certificado digital del servidor y se generan claves de sesión que serán utilizadas para cifrar y descifrar los datos transmitidos. Este procedimiento ayuda a evitar que terceros puedan leer, modificar o falsificar la información enviada durante la comunicación (Rescorla, 2018).

Una de las funciones más importantes de TLS es permitir el uso de HTTPS en sitios y aplicaciones web. Al utilizar HTTPS, la información transmitida entre el navegador y el servidor viaja cifrada, lo cual resulta especialmente relevante en sistemas que manejan credenciales, sesiones, datos personales o información sensible. Por ello, este tipo de protocolo constituye un componente esencial para fortalecer la seguridad en aplicaciones web modernas (Cloudflare, s. f.).

1.1.22 Servidor web

Un servidor web es un software o infraestructura encargada de recibir solicitudes provenientes de clientes, procesarlas y devolver recursos como páginas HTML, archivos estáticos, imágenes, hojas de estilo, scripts o respuestas generadas dinámicamente. En el funcionamiento de la web, el servidor espera peticiones realizadas por un navegador u otro cliente y responde mediante protocolos como HTTP o HTTPS, permitiendo que el usuario acceda al contenido solicitado (MDN Web Docs, s. f.).

El servidor web interpreta la URL solicitada, localiza el recurso correspondiente y envía una respuesta al cliente. Cuando se trata de contenido estático, puede entregar archivos almacenados directamente en el sistema. En cambio, cuando la solicitud requiere contenido dinámico, el servidor puede ejecutar lógica interna, consultar una base de datos, validar permisos o comunicarse con otros servicios antes de generar la respuesta final (IBM, s. f.).

Existen distintos tipos de servidores y entornos especializados para aplicaciones web. Algunos se orientan principalmente a servir archivos estáticos, mientras que otros permiten

ejecutar aplicaciones completas del lado del servidor. En este sentido, tecnologías como Node.js permiten construir aplicaciones basadas en JavaScript capaces de procesar solicitudes, administrar rutas, generar respuestas dinámicas y comunicarse con bases de datos o servicios externos (Node.js, s. f.).

Los servidores web son fundamentales para el despliegue de aplicaciones modernas, ya que permiten alojar sistemas, administrar recursos y ofrecer acceso remoto a los usuarios. En un sistema de gestión del aprendizaje, este componente permite que los distintos módulos puedan consultarse desde un navegador, procesar acciones del usuario y mantener comunicación con la base de datos y los servicios necesarios para el funcionamiento del sistema.

1.2 Marco tecnológico

El desarrollo de una plataforma web requiere la selección de herramientas tecnológicas que permitan transformar los fundamentos conceptuales en un sistema funcional. En este sentido, el marco tecnológico describe los lenguajes, *frameworks*, bibliotecas, herramientas de desarrollo y servicios utilizados para la construcción de Gasly LMS. Estos elementos intervienen en distintas capas del sistema, desde la estructura y presentación de la interfaz hasta la lógica de procesamiento, la administración de datos, la autenticación de usuarios y la comunicación entre los componentes de la aplicación.

1.2.1 HTML

HTML (*HyperText Markup Language*) es el lenguaje estándar utilizado para estructurar el contenido de las páginas web. Su función consiste en organizar la información mediante etiquetas que permiten definir encabezados, párrafos, imágenes, enlaces, listas, formularios, tablas y otros elementos que conforman la estructura básica de un documento web. HTML no se encarga de la programación lógica ni del diseño visual avanzado, sino de establecer la organización semántica del contenido (WHATWG, s. f.).

El lenguaje HTML se basa en un modelo de documentos compuesto por elementos y atributos. Los elementos representan partes específicas de una página, mientras que los atributos proporcionan información adicional sobre ellos, como identificadores, clases, rutas, textos alternativos o comportamientos asociados. Esta estructura permite que los

navegadores interpreten el contenido y lo presenten de forma comprensible para el usuario (MDN Web Docs, s. f.).

Una página HTML normalmente parte de una estructura general que incluye un elemento raíz, una sección de encabezado y una sección de cuerpo. En el encabezado se colocan metadatos, enlaces a hojas de estilo, scripts y el título del documento, mientras que en el cuerpo se integra el contenido visible para el usuario. Esta organización facilita la separación entre información, presentación y comportamiento dentro de una aplicación web (WHATWG, s. f.).

El contenido principal de una página puede organizarse mediante etiquetas como <h1> para encabezados principales, <p> para párrafos, <img> para imágenes, <a> para enlaces, <ul> y <ol> para listas, entre muchas otras. Estos elementos pueden anidarse para crear estructuras más complejas, lo que permite construir páginas organizadas, accesibles y compatibles con distintos navegadores y dispositivos (MDN Web Docs, s. f.).

1.2.2 CSS

CSS (*Cascading Style Sheets*) es un lenguaje de hojas de estilo utilizado para definir la presentación visual de documentos escritos en HTML u otros lenguajes de marcado. A través de CSS es posible establecer colores, tipografías, márgenes, tamaños, alineaciones, espaciados y distribución de elementos dentro de una página web. Su propósito principal es separar la estructura del contenido de su apariencia visual (MDN Web Docs, s. f.).

Una de las principales ventajas de CSS es que permite controlar la apariencia de múltiples páginas desde una misma hoja de estilo. Esto favorece la coherencia visual de un sitio web y facilita la actualización de diseños, ya que los cambios pueden aplicarse de forma general sin modificar individualmente cada página. Gracias a esta característica, CSS contribuye a mantener interfaces más consistentes y fáciles de administrar (W3C, s. f.).

CSS utiliza reglas compuestas por selectores y declaraciones. Los selectores permiten indicar a qué elementos del documento se aplicarán ciertos estilos, mientras que las declaraciones definen las propiedades visuales correspondientes. Estos selectores pueden apuntar a etiquetas HTML, clases, identificadores, atributos o relaciones entre elementos, lo que

proporciona flexibilidad para diseñar interfaces adaptadas a distintos contextos de uso (MDN Web Docs, s. f.).

1.2.3 JavaScript

JavaScript es un lenguaje de programación interpretado, ampliamente utilizado en el desarrollo web para agregar interactividad y dinamismo a las páginas. Fue creado en 1995 y actualmente constituye una de las tecnologías principales de la web, junto con HTML y CSS. Mientras HTML estructura el contenido y CSS define su presentación, JavaScript permite modificar elementos, responder a eventos y ejecutar lógica dentro del navegador (MDN Web Docs, s. f.).

Una de sus características más relevantes es su capacidad para interactuar con el DOM (*Document Object Model*), que representa la estructura de una página web como un conjunto de objetos. Mediante esta interacción, JavaScript puede seleccionar elementos, modificar su contenido, cambiar estilos, validar formularios o responder a acciones del usuario como clics, desplazamientos o escritura en campos de entrada (Flanagan, 2020).

Además de su uso en el navegador, JavaScript también puede emplearse del lado del servidor mediante entornos como Node.js. Esta posibilidad ha permitido que el lenguaje se utilice en el desarrollo de aplicaciones completas, facilitando la comunicación con API, la manipulación de datos y la construcción de experiencias interactivas. Por esta razón, JavaScript se considera una tecnología central en el desarrollo web moderno (MDN Web Docs, s. f.).

1.2.4 TypeScript

TypeScript es un lenguaje de programación desarrollado por Microsoft que extiende JavaScript mediante la incorporación de tipado estático y características adicionales para mejorar la organización del código. Su propósito es facilitar la detección temprana de errores durante el desarrollo, antes de que la aplicación sea ejecutada en el navegador o en el servidor. Al compilarse a JavaScript, puede ejecutarse en cualquier entorno compatible con dicho lenguaje (TypeScript, s. f.).

Entre sus características se encuentran el uso de tipos, interfaces, clases, genéricos y herramientas de análisis que permiten construir aplicaciones más mantenibles. Estas funciones ayudan a que los desarrolladores puedan definir con mayor claridad la forma de los datos, las propiedades esperadas en los objetos y los contratos entre distintas partes del sistema. Esto resulta útil en proyectos medianos o grandes, donde la organización del código se vuelve fundamental (Bierman et al., 2014).

TypeScript mantiene compatibilidad con JavaScript, lo que permite adoptarlo de manera gradual dentro de un proyecto existente. Además, mejora la experiencia de desarrollo mediante autocompletado, detección de errores, documentación integrada y mayor claridad en la estructura del código. Por estas razones, se ha convertido en una herramienta ampliamente utilizada en aplicaciones web modernas, especialmente en proyectos desarrollados con frameworks como React o Next.js (TypeScript, s. f.).

1.2.5 ORM

Un ORM, por sus siglas en inglés *Object-Relational Mapping*, es una técnica que permite relacionar los modelos utilizados en una aplicación con las tablas de una base de datos relacional. Su propósito es facilitar la interacción con la información almacenada, permitiendo que los desarrolladores trabajen con objetos, clases o modelos dentro del código, en lugar de escribir directamente consultas SQL para cada operación (Fowler, 2002).

El uso de un ORM permite realizar operaciones comunes sobre la base de datos, como crear, consultar, actualizar o eliminar registros, mediante estructuras propias del lenguaje de programación utilizado. Esto contribuye a reducir código repetitivo, mejorar la organización del proyecto y disminuir ciertos errores asociados con la escritura manual de consultas. En aplicaciones web, este tipo de herramienta resulta útil porque actúa como una capa intermedia entre la lógica del sistema y la base de datos (Prisma, s. f.).

1.2.6 Prisma

Prisma es una herramienta ORM que facilita la interacción entre una aplicación y una base de datos. Permite definir modelos de datos dentro del proyecto y trabajar con ellos mediante una API de JavaScript o TypeScript, evitando escribir consultas SQL de manera directa para

cada operación. Esta característica contribuye a mejorar la organización del código y la seguridad en el manejo de datos (Prisma, s. f.).

Además, Prisma permite generar migraciones, consultar registros, crear relaciones entre entidades y mantener sincronizada la estructura de la base de datos con los modelos definidos en la aplicación. También incluye herramientas como Prisma Client, utilizado para realizar operaciones sobre la base de datos, y Prisma Studio, que permite visualizar y administrar datos mediante una interfaz gráfica.

1.2.7 API

Una API, por sus siglas en inglés *Application Programming Interface*, es un conjunto de reglas, mecanismos y definiciones que permite la comunicación entre distintos componentes de software. En el desarrollo web, las API se utilizan para intercambiar información entre el frontend y el backend, así como entre una aplicación y servicios externos. Gracias a ellas, es posible solicitar, enviar, actualizar o eliminar datos de manera estructurada, generalmente mediante protocolos como HTTP (Fielding et al., 2022).

Las API son importantes en las aplicaciones web modernas porque favorecen la interoperabilidad, la modularidad y la separación de responsabilidades dentro del sistema. En muchos casos, el backend expone rutas o endpoints que pueden ser consumidos por el frontend para mostrar información dinámica al usuario. De esta forma, las API funcionan como un puente de comunicación que permite conectar la interfaz con la lógica del sistema y con la información almacenada en la base de datos.

1.2.8 Node.js

Node.js es un entorno de ejecución de JavaScript del lado del servidor, construido sobre el motor V8 de Google Chrome. Su propósito es permitir que JavaScript, tradicionalmente utilizado en el navegador, también pueda ejecutarse en el backend para construir servidores, API y aplicaciones web escalables. Gracias a su arquitectura basada en eventos, resulta útil en sistemas que requieren manejar múltiples solicitudes de manera eficiente (Node.js, s. f.).

Además de su rendimiento, Node.js cuenta con un amplio ecosistema de paquetes administrados mediante npm, lo que facilita la incorporación de librerías y herramientas para

el desarrollo. Esta característica permite acelerar la construcción de aplicaciones, reutilizar componentes existentes y mantener una mayor coherencia entre el frontend y el backend cuando ambos utilizan JavaScript o TypeScript.

En el desarrollo de aplicaciones web modernas, Node.js se utiliza con frecuencia para construir servicios de backend, procesar solicitudes, conectarse con bases de datos y exponer rutas mediante API. En este sentido, representa una base tecnológica importante para frameworks y herramientas actuales orientadas al desarrollo web.

1.2.9 React

React es una biblioteca de JavaScript utilizada para construir interfaces de usuario mediante componentes reutilizables. Fue desarrollada originalmente por Facebook, actualmente Meta, y se ha consolidado como una herramienta ampliamente utilizada en el desarrollo de aplicaciones web interactivas. Su enfoque basado en componentes permite dividir la interfaz en partes independientes, lo que facilita su construcción, mantenimiento y reutilización (Meta, s. f.).

Los componentes de React permiten representar elementos visuales y funcionales de una aplicación, como botones, formularios, tarjetas, menús o secciones completas de una página. Cada componente puede manejar propiedades, estado e interacción con el usuario, lo que permite construir interfaces dinámicas de manera organizada. Esta forma de trabajo favorece la claridad del código y la separación de responsabilidades dentro del frontend.

Otra característica importante de React es su capacidad para actualizar la interfaz de manera eficiente cuando cambian los datos de la aplicación. Esto permite crear experiencias más fluidas para el usuario, especialmente en sistemas donde la información se modifica constantemente, como paneles administrativos, plataformas educativas o aplicaciones con contenido dinámico.

1.2.10 Next.js

Next.js es un framework de desarrollo web basado en React que permite construir aplicaciones modernas con funcionalidades tanto del lado del cliente como del lado del servidor. Entre sus características principales se encuentran el enrutamiento basado en

archivos, la optimización de rendimiento, el soporte para TypeScript y la posibilidad de trabajar con distintos modelos de renderizado, como renderizado del lado del servidor y generación estática (Vercel, s. f.).

Este framework facilita la creación de aplicaciones web estructuradas, ya que integra herramientas para la organización de rutas, manejo de componentes, carga de datos, optimización de imágenes y construcción de API dentro del mismo proyecto. De esta manera, permite desarrollar aplicaciones más completas sin depender de múltiples configuraciones externas.

1.2.11 Tailwind CSS

Tailwind CSS es un framework de estilos utilitarios que permite construir interfaces web mediante clases predefinidas aplicadas directamente en el marcado HTML o JSX. A diferencia de otros frameworks visuales que ofrecen componentes ya diseñados, Tailwind proporciona utilidades para definir colores, espaciados, tamaños, tipografías, bordes, distribución y comportamiento responsivo de los elementos (Tailwind Labs, s. f.).

Una de las principales ventajas de Tailwind CSS es que favorece la productividad y la coherencia visual dentro de un proyecto, ya que facilita la aplicación de estilos de forma uniforme en distintos componentes de la interfaz. Además, cuenta con un sistema de configuración que permite personalizar aspectos como colores, tipografías, espaciados, tamaños y puntos de quiebre para diseño responsivo.

1.2.12 JSON Web Token (JWT)

JSON Web Token, conocido por sus siglas JWT, es un estándar utilizado para representar información de manera compacta y segura entre dos partes. En aplicaciones web, este tipo de token se emplea comúnmente en procesos de autenticación y autorización, ya que permite verificar la identidad de un usuario y mantener información relacionada con su sesión sin depender exclusivamente de almacenamiento en el servidor (Jones et al., 2015).

Un JWT se compone de tres partes: encabezado, carga útil y firma. El encabezado indica el tipo de token y el algoritmo utilizado; la carga útil contiene los datos o declaraciones relacionadas con el usuario; y la firma permite verificar que el contenido no haya sido

alterado. Gracias a esta estructura, los tokens pueden utilizarse para proteger rutas, validar permisos y controlar el acceso a funciones específicas dentro de una aplicación web.

En sistemas con distintos perfiles de usuario, como estudiantes, instructores y administradores, el uso de tokens facilita la administración de sesiones y el control de acceso. Este mecanismo resulta relevante porque permite restringir determinadas funciones según el rol del usuario, protegiendo la información y asegurando que cada perfil interactúe únicamente con los módulos correspondientes.

1.2.13 Auth.js

Auth.js es una biblioteca orientada a la autenticación en aplicaciones web modernas, especialmente dentro del ecosistema de Next.js. Su función principal es simplificar la implementación de inicio de sesión, manejo de sesiones, proveedores de autenticación y control de acceso. También ofrece soporte para distintos métodos de autenticación, como proveedores OAuth, OpenID Connect y credenciales personalizadas (Auth.js, s. f.).

Una de sus ventajas es que facilita la gestión de sesiones y tokens dentro de la aplicación, reduciendo la necesidad de implementar desde cero mecanismos complejos de autenticación. Esto permite que los desarrolladores se enfoquen en las funcionalidades principales del sistema, manteniendo al mismo tiempo una estructura segura para validar usuarios y proteger rutas.

1.2.14 Visual Studio Code

Visual Studio Code, también conocido como VS Code, es un editor de código fuente desarrollado por Microsoft. Este editor es compatible con distintos lenguajes de programación y tecnologías utilizadas en el desarrollo web, como JavaScript, TypeScript, HTML, CSS y frameworks modernos. Entre sus características principales se encuentran el resaltado de sintaxis, autocompletado inteligente, depuración, integración con Git y soporte para extensiones (Microsoft, s. f.).

Además, VS Code cuenta con un ecosistema robusto de extensiones que permiten personalizar y ampliar su funcionalidad. Los desarrolladores pueden instalar extensiones para soporte de depuración, control de versiones con Git, integración con servicios en la nube y mucho más. Su interfaz amigable y su capacidad de integración con una variedad de

herramientas de desarrollo lo convierten en una opción popular tanto para proyectos pequeños como grandes, y es compatible con sistemas operativos como Windows, macOS y Linux.

1.2.15 Vercel

Vercel es una plataforma de alojamiento y despliegue en la nube orientada a la publicación de aplicaciones web modernas. Su función principal es permitir que los proyectos desarrollados con tecnologías como Next.js, React y otros frameworks web puedan construirse, desplegarse y escalarse en un entorno accesible desde Internet. De acuerdo con su documentación oficial, Vercel proporciona herramientas e infraestructura en la nube para construir, desplegar y escalar aplicaciones web, además de permitir la conexión con repositorios de código para generar despliegues automáticos y entornos de vista previa antes de publicar cambios en producción (Vercel, s. f.).

Una de las características relevantes de Vercel es que facilita el flujo de trabajo entre desarrollo, prueba y producción. Cuando un proyecto se conecta a un repositorio Git, cada actualización puede generar un despliegue automático, lo que permite revisar cambios mediante URL de vista previa antes de liberarlos en el entorno final. Asimismo, la plataforma ofrece funciones relacionadas con rendimiento, seguridad y distribución global, como red de entrega de contenido, compatibilidad con HTTPS, manejo de variables de entorno y optimización para aplicaciones desarrolladas con frameworks modernos (Vercel, s. f.).

1.3 Marco metodológico

Para llevar a cabo cualquier proyecto de desarrollo de software de manera profesional, es fundamental emplear una metodología. Las metodologías desempeñan un papel crucial en la planificación, ejecución y entrega de proyectos.

En el contexto del desarrollo de software, una metodología se refiere a un conjunto sistemático de prácticas, técnicas, procedimientos y directrices que guían dicho desarrollo. Por lo tanto, tener un buen conocimiento de estas metodologías es esencial para comprender desde el análisis hasta la entrega final del producto, abarcando también el diseño y la implementación de los sistemas.

Un aspecto relevante de las metodologías es la oportunidad que ofrecen a los equipos de trabajo para gestionar de manera más efectiva todos los recursos disponibles y, de igual forma, mitigar cualquier riesgo que pueda afectar la entrega del proyecto. Esto garantiza la calidad del producto final y, en consecuencia, la satisfacción de las necesidades del cliente.

Para desarrollar este sistema de gestión del aprendizaje, se han tenido en cuenta las siguientes metodologías, que se describen y comparan a continuación: RMM, WSDM, EORM y OOWS.

1.3.1 RMM

La metodología RMM (*Relationship Management Methodology*) es un enfoque sistemático y estructurado para el diseño y modelado de aplicaciones hipermedia. Esta metodología permite representar el dominio de información, sus relaciones y las estructuras de navegación necesarias para que el usuario pueda recorrer los contenidos de manera organizada (Isakowitz et al., 1995).

Esta metodología está orientada al desarrollo de aplicaciones hipermedia que cuentan con clases de objetos bien definidas y relaciones claras entre ellas. Su utilidad se encuentra en la posibilidad de organizar la información, establecer relaciones entre entidades y definir rutas de navegación coherentes para los usuarios finales.

El modelo propone un enfoque estructurado que facilita la descripción de los objetos del dominio, sus relaciones y las operaciones de navegación hipermedia dentro de la aplicación. Al emplear entidades, atributos y relaciones asociativas, se logra una representación completa y coherente del sistema.

Una característica importante en lo que respecta a este modelo es el concepto de *slice* o fragmento, el cual permite abordar la presentación de las entidades de manera más efectiva. Estos *slices* representan subconjuntos de atributos de una entidad específica que se agrupan para su presentación conjunta, simplificando así la visualización de la información (Isakowitz et al., 1995).

Además, la navegación dentro de la aplicación se modela mediante primitivas de acceso y enlaces estructurales, que definen las conexiones entre *slices*, así como visitas guiadas

condicionales, índices condicionales y agrupaciones, que determinan la forma en que los usuarios pueden moverse entre las distintas entidades.

El esquema completo del dominio y de la navegación se conoce como esquema RMDM (*Relationship Management Data Model*), obtenido como resultado de las primeras etapas del método. Este enfoque proporciona una base sólida para el diseño y desarrollo de aplicaciones, permitiendo una comprensión profunda de la estructura de los datos y las interacciones dentro del sistema (Isakowitz et al., 1995).

La metodología RMM (*Relationship Management Methodology*) utiliza el modelo entidad-relación como una herramienta de apoyo en las etapas iniciales de diseño, ya que permite representar el dominio de información de la aplicación y las relaciones existentes entre sus componentes. A partir de este esquema, es posible identificar las unidades principales de contenido, sus atributos y los vínculos que posteriormente servirán como base para definir la estructura de navegación dentro de una aplicación hipermedia (Isakowitz et al., 1995; Pérez Rodríguez & Sánchez Torres, 2007).

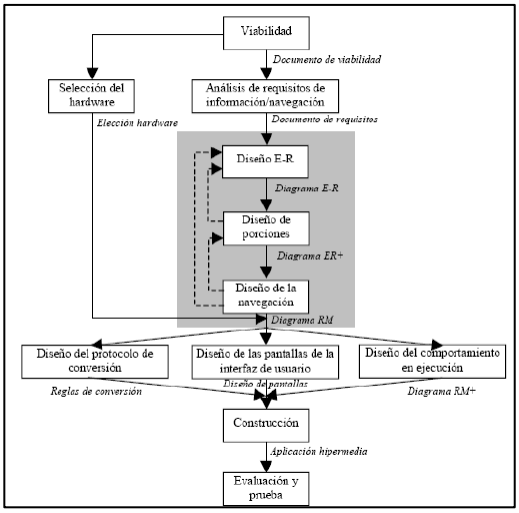
El modelo entidad-relación se compone de tres elementos fundamentales: entidades, atributos y relaciones. Las entidades representan objetos o conceptos relevantes dentro del dominio de aplicación; los atributos describen las características asociadas a cada entidad, y las relaciones establecen los vínculos existentes entre dos o más entidades. Estas relaciones pueden presentar distintas cardinalidades, como uno a uno, uno a muchos o muchos a muchos. En este último caso, las relaciones muchos a muchos pueden descomponerse en dos relaciones uno a muchos para facilitar su manejo dentro del diseño de la base de datos (Elmasri & Navathe, 2016).

En términos generales, RMM plantea un proceso que parte de la identificación del dominio de información y continúa con la organización de sus unidades de contenido, la definición de estructuras de navegación, el diseño de la interfaz y la prueba de la aplicación. Esta secuencia permite relacionar la estructura de los datos con la experiencia de navegación del usuario, aspecto relevante en el desarrollo de aplicaciones hipermedia y sistemas educativos basados en recursos digitales (Balasubramanian et al., 2001; Pérez Rodríguez & Sánchez Torres, 2007).

En la Figura 1.1 se muestra el ciclo de vida de RMM, representación que permite comprender las fases que se describen a continuación.

Figura 1.1

Ciclo de vida de RMM



Nota. Tomado de *La metodología RMM (Relationship Management Methodology) aplicable al desarrollo del software educativo multimedia Topografía*, por Y. Pérez Rodríguez y J. Sánchez Torres, 2007, Universidad de las Ciencias Informáticas.

1.3.1.1 Fases de la metodología

Es importante destacar que, para iniciar cualquier proyecto, es necesario seguir ciertos procedimientos, tal como se hace en otras metodologías. Esto implica realizar un estudio de viabilidad y un análisis de requerimientos, así como seleccionar el hardware y el software que mejor se adapten a los objetivos del proyecto.

Fase 1: modelo entidad-relación

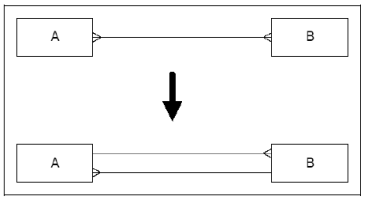
La primera etapa de la metodología RMM corresponde al diseño entidad-relación, en la cual se representa el dominio de información de la aplicación mediante un diagrama E-R. En esta fase se identifican las entidades y relaciones relevantes del sistema, ya que estas constituyen la base sobre la que posteriormente se organizarán los nodos, enlaces y recorridos de navegación. Su objetivo principal es hacer explícitas las relaciones entre los objetos del dominio, debido a que dichas relaciones pueden convertirse en rutas de acceso a la

información dentro de la aplicación hipermedia. Asimismo, se consideran las cardinalidades entre entidades, como las relaciones uno a uno, uno a muchos y muchos a muchos, estas últimas descompuestas en dos relaciones uno a muchos para facilitar su representación. De esta manera, el modelo entidad-relación no solo permite estructurar la información del sistema, sino que también sirve como punto de partida para definir los posibles caminos de navegación que tendrá el usuario dentro de la aplicación (Pérez Rodríguez & Sánchez Torres, 2007, p. 42).

Una vez identificadas las entidades, se procede a analizar las relaciones entre ellas. Estas pueden tomar diversas formas, como uno a uno (1:1), uno a muchos (1:N) o muchos a muchos (N:M). En este último caso, las relaciones muchos a muchos pueden desglosarse en dos relaciones uno a muchos para facilitar su implementación y comprensión dentro del diseño de la base de datos. La Figura 1.2 muestra la representación de este tipo de relación dentro del diagrama entidad-relación.

Figura 1.2

Relaciones N:M en el diagrama E-R



Nota. Tomado de *La metodología RMM (Relationship Management Methodology) aplicable al desarrollo del software educativo multimedia Topografía*, por Y. Pérez Rodríguez y J. Sánchez Torres, 2007, Universidad de las Ciencias Informáticas, p. 41.

Fase 2: diseño de slices

Dentro de la segunda fase se busca organizar y presentar la información de forma eficaz dentro de entornos interactivos. Para ello, el sistema web o recurso hipermedia se descompone en *slices* o componentes significativos del contenido. Cada *slice* representa una unidad de información que puede presentarse de manera individual o interrelacionada con

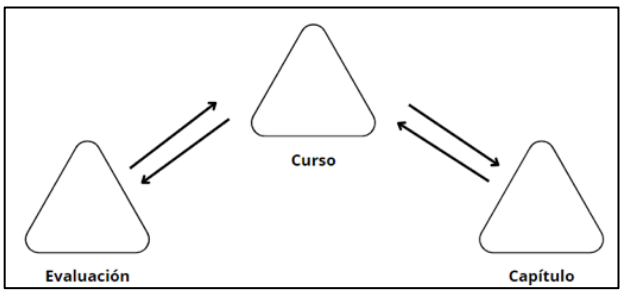
otros fragmentos. Estos componentes se organizan en una red de hipertexto que permite al usuario navegar entre ellos mediante enlaces unidireccionales o bidireccionales, determinando qué partes de la información serán mostradas y cómo se accederá a ellas.

En cada entidad existe un slice cabeza, el cual corresponde a la vista inicial mediante la que se accede a dicha entidad. El diagrama de slices representa la estructura y relación entre estos componentes, utilizando enlaces estructurales para conectar partes dentro de una misma instancia de entidad y enlaces de asociación para conectar diferentes instancias. Como resultado de esta fase, se obtiene un diagrama enriquecido de entidad-relación, denominado E-R+, en el que cada entidad ha sido reemplazada por su esquema de entidad, constituido por los slices y sus relaciones estructurales. Este proceso permite organizar la presentación de la información y facilitar la navegación dentro de aplicaciones hipermedia (Isakowitz et al., 1995).

Para ejemplificar este concepto dentro del contexto del proyecto, la Figura 1.3 representa los enlaces estructurales entre cursos, capítulos y evaluaciones. Estos elementos permiten comprender cómo los distintos componentes del sistema pueden relacionarse entre sí para favorecer una navegación ordenada. Por ejemplo, un estudiante puede acceder desde un curso a sus capítulos y evaluaciones asociadas, mientras que instructores y administradores pueden gestionar dichos elementos desde sus respectivos módulos.

Figura 1.3

Representación de enlaces estructurales entre slices *en RMM aplicada al LMS*



Nota. Adaptado de *La metodología RMM (Relationship Management Methodology) aplicable al desarrollo del software educativo multimedia Topografía*, por Y. Pérez Rodríguez y J. Sánchez Torres, 2007, Universidad de las Ciencias Informáticas, p. 45.

Fase 3: navegación

En esta fase se diseñan las rutas que permitirán la navegación dentro de la aplicación. Para ello, se analiza cada relación de asociación identificada en el diagrama E-R+ y se determina si debe convertirse en una estructura accesible para el usuario. Cuando una relación se considera necesaria para la navegación, se transforma en una primitiva de acceso RMDM. Debido a que RMM puede emplearse en dominios donde la información cambia con frecuencia, los accesos navegacionales se especifican de manera genérica, utilizando propiedades de entidades y relaciones.

Estas estructuras de acceso pueden representarse mediante índices condicionales, visitas guiadas condicionales y visitas guiadas indexadas condicionales. Las visitas guiadas suelen utilizarse cuando el número de instancias es reducido, mientras que los índices resultan más adecuados cuando existe una mayor cantidad de elementos. También se define el acceso al *slice* principal de cada entidad y se establece una jerarquía de menús, procurando evitar niveles excesivos que puedan desorientar al usuario. Como resultado de esta fase, el diagrama E-R+ se transforma en un diagrama RMDM, el cual describe tanto la información relacionada con las clases de objetos como las posibilidades de navegación entre ellas (Pérez Rodríguez & Sánchez Torres, 2007).

Fase 4: definir el protocolo de conversión

La fase 4 se enfoca en la traducción de las primitivas de acceso RMDM a primitivas de acceso del lenguaje o sistema que será utilizado para implementar el diseño. Durante este paso, se establece un conjunto de reglas de conversión que permite transformar los elementos del diagrama RMDM en objetos compatibles con la plataforma de destino. Este protocolo busca asegurar que las estructuras definidas en el modelo puedan adaptarse de manera coherente al entorno de implementación, facilitando la transición entre el diseño conceptual de la navegación y su representación práctica dentro de la aplicación. En este sentido, herramientas como RMCase muestran que RMM también fue planteada como una metodología susceptible de apoyo computacional para el diseño y construcción de aplicaciones hipermedia (Díaz & Isakowitz, 1995).

Fase 5: diseño de interfaz de usuario

En esta fase se realiza el diseño detallado de la interfaz de usuario, tomando como base los *slices* identificados en las etapas anteriores. Cada pantalla debe representar de manera clara la información asociada a dichos fragmentos, de modo que los usuarios puedan comprender la estructura del contenido y acceder a las secciones correspondientes sin dificultad. A diferencia de las fases previas, donde se define principalmente la estructura de la información y la navegación, en esta etapa se busca trasladar esos elementos a una representación visual comprensible para el usuario. Por ello, la interfaz no se diseña de forma aislada, sino en relación con el modelo RMDM y con las estructuras de acceso previamente establecidas.

El diseño también contempla la forma en que se presentarán los enlaces, anclas, índices y mecanismos de regreso dentro de la aplicación. En este punto resulta importante diferenciar visualmente los enlaces estructurales, que conectan información dentro de una misma entidad, y los enlaces de asociación, que permiten desplazarse hacia otras entidades relacionadas. Esta distinción ayuda a que el usuario mantenga una mejor orientación dentro del sistema y comprenda con mayor facilidad las relaciones entre los elementos del modelo. En este sentido, la interfaz funciona como una representación visual de la estructura informativa y navegacional definida por la metodología RMM (Isakowitz et al., 1995).

Fase 6: diseño del comportamiento en ejecución

La fase de diseño del comportamiento en ejecución define la manera en que la aplicación responderá durante su uso. Esto incluye aspectos como el historial de navegación, la capacidad de retroceso, la selección de enlaces y otros mecanismos que influyen directamente en la interacción del usuario con el sistema. También se determina si los nodos y enlaces serán generados de forma estática o dinámica; la generación estática implica preparar previamente el contenido y mantenerlo almacenado, mientras que la generación dinámica permite construir o presentar información en función de las acciones del usuario o de los datos disponibles en el sistema. Esta decisión resulta importante porque afecta la flexibilidad, actualización y respuesta de la aplicación ante distintos escenarios de navegación. De esta manera, RMM no solo considera la estructura del contenido, sino también el comportamiento que tendrá la aplicación al momento de ser utilizada (Pérez Rodríguez & Sánchez Torres, 2007).

Fase 7: prueba de la aplicación

La última fase corresponde a la prueba de la aplicación, cuyo objetivo es verificar que el diseño conceptual, la navegación y la interfaz funcionen de acuerdo con lo establecido en las etapas anteriores. En este proceso se revisa que las entidades, relaciones, *slices*, enlaces y estructuras de acceso se comporten de manera adecuada dentro del sistema implementado.

Esta etapa permite identificar errores, inconsistencias o dificultades de uso antes de considerar concluido el desarrollo. Las pruebas ayudan a confirmar que las rutas de navegación sean funcionales, que los enlaces conduzcan a los elementos esperados y que la presentación de la información sea coherente con el modelo RMDM. Desde esta perspectiva, la prueba de la aplicación constituye el cierre del proceso metodológico, ya que permite validar la correspondencia entre el diseño propuesto y el funcionamiento real del sistema (Balasubramanian et al., 2001).

1.3.2 WSDM

La metodología WSDM (*Web Site Design Method*) es un método orientado al diseño y desarrollo de sitios web a partir del análisis de los usuarios que interactuarán con el sistema. Su principal característica consiste en considerar, desde las primeras etapas, los distintos grupos de usuarios, sus necesidades, objetivos y formas de interacción, de manera que la estructura del sitio no se defina únicamente desde la información disponible, sino también desde las personas que harán uso de ella (De Troyer & Leune, 1998).

WSDM surgió a partir de las propuestas de De Troyer y Leune, quienes plantearon la importancia de diseñar sitios web con base en los grupos de usuarios identificados. De acuerdo con Escalona Cuaresma (2001), esta metodología se compone de cuatro fases principales: modelo de usuario, diseño conceptual, diseño de implementación e implementación. A su vez, el modelo de usuario se divide en clasificación y descripción de usuarios, mientras que el diseño conceptual considera el modelado de objetos y el diseño navegacional.

A diferencia de metodologías centradas principalmente en la estructura de datos o en la navegación hipermedia, WSDM coloca el análisis del usuario como punto de partida del diseño. Esta característica resulta relevante porque un mismo sitio puede ser utilizado por

distintos perfiles, cada uno con necesidades, expectativas y tareas particulares. Por ello, la metodología busca que las decisiones de organización, navegación e implementación se deriven de los grupos de usuarios identificados (Molina Ríos & Zea Ordoñez, 2017).

La metodología también se relaciona con principios de diseño centrado en el usuario, ya que busca comprender las necesidades, comportamientos y expectativas de quienes interactúan con el sitio. A partir de este análisis se definen perfiles, requisitos de información y estructuras de navegación que guían el diseño conceptual y la posterior implementación. De esta manera, WSDM no solo atiende la organización de la información, sino también la forma en que cada grupo de usuarios accederá a los contenidos y funciones disponibles (De Troyer & Leune, 1998).

Además de su orientación hacia el usuario, WSDM contempla aspectos de diseño conceptual, navegación, diseño de implementación y desarrollo técnico. Esto permite que los requisitos identificados en las primeras fases se transformen progresivamente en modelos, interfaces, rutas de navegación y funcionalidades concretas. En este sentido, la metodología resulta útil para proyectos web donde la organización de contenidos y la diferenciación de usuarios son elementos importantes del sistema.

A continuación, se describen las fases principales del ciclo de vida de la metodología WSDM.

Fase 1: modelo de usuario

En esta etapa inicial se identifican y clasifican los usuarios que interactuarán con la aplicación web. WSDM parte de la idea de que un sitio no debe diseñarse únicamente a partir de la información que contendrá, sino también considerando a las personas que lo utilizarán, los objetivos que persiguen y las tareas que necesitan realizar. Por ello, esta fase permite reconocer los distintos grupos de usuarios desde el inicio del proceso metodológico, evitando que el diseño se construya de forma general o indiferenciada (De Troyer & Leune, 1998).

El modelo de usuario contempla dos actividades principales: la clasificación de usuarios y la descripción de los grupos identificados. En la primera se reconocen los tipos de usuarios que tendrá el sistema, considerando el entorno donde será utilizado y los procesos que se generarán. En la segunda se describen con mayor detalle las características y necesidades de cada grupo, incluyendo los requisitos de información que deberán ser considerados durante

el diseño. Escalona Cuaresma (2001) señala que esta fase permite generar una representación inicial de los usuarios y de sus necesidades dentro del sistema.

Además, esta fase sirve como base para las decisiones posteriores de diseño conceptual, navegación e implementación. Al definir los perfiles desde el inicio, es posible anticipar qué información debe mostrarse, qué rutas serán necesarias y qué tipo de interacción tendrá cada grupo con el sitio. De esta manera, el modelo de usuario no solo describe quién utilizará la aplicación, sino que también orienta la organización general del sistema y contribuye a que el diseño responda a necesidades concretas.

Fase 2: diseño conceptual

La segunda fase corresponde al diseño conceptual del sitio web. En esta etapa se define la estructura general de la información, los objetos principales del sistema y la manera en que estos se relacionan entre sí. Su propósito es transformar los requisitos identificados en el modelo de usuario en una organización lógica que permita construir una experiencia de navegación clara y coherente (Escalona Cuaresma, 2001).

Esta fase se divide en dos subfases: modelado de objetos y diseño navegacional. En el modelado de objetos se representan formalmente los requisitos de información de cada grupo de usuarios mediante modelos específicos. En el diseño navegacional se definen las posibilidades de navegación del sistema, considerando cómo un usuario concreto podrá acceder a determinada información. Escalona Cuaresma (2001) explica que WSDM propone construir el modelo navegacional a partir del modelo de clases correspondiente a cada usuario, tomando cada uno de estos modelos como un contexto.

Dentro de esta fase, la navegación adquiere un papel central. No basta con establecer qué información estará disponible, sino que también debe definirse cómo se accederá a ella, desde qué puntos del sitio y bajo qué estructura. Por esta razón, el diseño conceptual puede apoyarse en componentes de información, componentes de navegación, enlaces y referencias externas, con el fin de organizar el recorrido del usuario dentro del sitio.

El modelo conceptual permite diferenciar entre la información que forma parte del sistema y las rutas mediante las cuales será consultada. Esta separación ayuda a evitar que la navegación dependa de enlaces aislados o de decisiones improvisadas de interfaz. En

consecuencia, el diseño conceptual funciona como una representación previa del sitio antes de su construcción técnica, permitiendo validar la coherencia entre usuarios, contenidos y navegación.

Fase 3: diseño de implementación

La fase de diseño de implementación permite trasladar el diseño conceptual hacia una propuesta más cercana a la construcción real del sitio web. En esta etapa se definen las interfaces, la arquitectura de navegación, la distribución de los elementos visuales y la forma en que las funciones serán presentadas dentro de la aplicación. Su importancia radica en que convierte los modelos abstractos en elementos más concretos, como pantallas, menús, componentes, rutas y criterios de interacción (Escalona Cuaresma, 2001).

Esta fase actúa como un puente entre la planeación conceptual y la codificación. Mientras el diseño conceptual establece qué información y rutas deben existir, el diseño de implementación determina cómo se verán y cómo serán utilizadas dentro del sitio. Por ello, en esta etapa se toman decisiones relacionadas con la presentación visual, la organización de pantallas, el uso de elementos multimedia, la consistencia de la interfaz y la facilidad con la que el usuario podrá desplazarse entre las diferentes secciones.

Un punto importante es que esta fase no debe confundirse con la programación final del sistema. Su función principal es preparar la representación visual y funcional del sitio antes de la implementación, procurando que la interfaz mantenga relación con los modelos definidos previamente. De esta manera, se reduce la distancia entre los modelos conceptuales y el producto que finalmente será desarrollado.

Fase 4: implementación

La fase de implementación corresponde a la construcción técnica del sitio web a partir de los modelos definidos en las etapas anteriores. En esta fase se realiza la codificación de las interfaces, rutas, componentes y funcionalidades necesarias para que la aplicación pueda operar de manera funcional. También se integran las tecnologías seleccionadas para el desarrollo, así como los mecanismos que permiten conectar la interfaz con la lógica del sistema, la base de datos y los servicios requeridos para su funcionamiento. En este sentido,

la implementación no consiste únicamente en programar, sino en materializar las decisiones tomadas durante el análisis de usuarios, el diseño conceptual y el diseño de implementación.

Además de la codificación, esta etapa implica revisar que el sitio construido mantenga coherencia con los requisitos establecidos. Esto supone verificar que las funciones respondan a los objetivos del proyecto, que la navegación conserve relación con el modelo conceptual y que la interfaz permita una interacción clara para los usuarios. De esta manera, WSDM plantea una transición progresiva desde la identificación de usuarios hasta la construcción del sitio, procurando que el resultado final conserve relación con las necesidades detectadas en las fases iniciales (De Troyer & Leune, 1998).

En conjunto, las fases de WSDM permiten desarrollar un sitio web mediante una secuencia que inicia con el análisis de los usuarios y avanza hacia la organización conceptual, la definición de la interfaz y la implementación técnica. Esta estructura favorece que las decisiones de diseño no dependan únicamente de criterios técnicos, sino también de los perfiles, tareas y formas de interacción de quienes utilizarán el sistema.

Por esta razón, WSDM resulta una metodología pertinente para proyectos en los que la navegación, la organización de contenidos y la diferenciación de usuarios son elementos centrales. Su enfoque permite mantener una relación clara entre las necesidades identificadas al inicio del proceso y las funcionalidades que finalmente se incorporan en la aplicación.

1.3.3 EORM

La metodología EORM (*Enhanced Object Relationship Methodology*) es una propuesta orientada al diseño de aplicaciones hipermedia mediante el uso del paradigma de orientación a objetos. Su propósito consiste en representar los elementos principales del sistema como objetos y enriquecer sus relaciones mediante enlaces, de manera que el modelo no solo describa la estructura de la información, sino también las posibilidades de navegación entre sus componentes. De acuerdo con Escalona Cuaresma (2001), EORM surge a partir de propuestas como RMM y HDM, pero se orienta hacia el paradigma de objetos, por lo que resulta relevante dentro de las metodologías de diseño hipermedia.

Una de sus características principales es que propone un proceso iterativo basado en el enriquecimiento de un modelo de objetos. En lugar de tratar las relaciones únicamente como

asociaciones estructurales, EORM las representa como enlaces que pueden formar parte de la navegación dentro del sistema. Esta forma de trabajo permite separar el modelo conceptual de la estructura navegacional, lo cual favorece la reutilización y facilita el mantenimiento cuando se requiere modificar la navegación sin alterar los objetos principales del dominio.

Desde una perspectiva comparativa, EORM puede ubicarse entre las metodologías de desarrollo web que utilizan técnicas de modelado orientado a objetos. Molina Ríos y Zea Ordoñez (2017) la describen como una metodología que permite desarrollar aplicaciones web mediante una estructura orientada a objetos y que se compone de tres fases principales: análisis, diseño y construcción.

1.3.3.1 Semántica de vínculos básicos de clases en EORM

Dentro de EORM, los vínculos permiten representar las relaciones hipermedia entre los objetos del sistema. Estos vínculos no solo conectan elementos, sino que también definen la manera en que el usuario podrá recorrer la información. Entre los principales tipos se encuentran los siguientes:

SimpleLink: vínculo base que permite establecer conexiones entre objetos, así como operaciones generales de creación, eliminación y recorrido.

NavigationalLink: vínculo orientado a la navegación hipermedia, incluyendo mecanismos relacionados con el recorrido y el almacenamiento de información histórica.

NodeToNode: vínculo que conecta un objeto con otro objeto, permitiendo la navegación directa entre nodos.

SpanToNode: vínculo que relaciona una parte específica del contenido de un objeto con otro objeto.

StructureLink: vínculo estructural utilizado para representar relaciones dentro de un contexto organizativo.

SetLink y ListLink: vínculos estructurales que permiten acceder a colecciones de objetos, ya sea de forma desordenada o mediante una organización secuencial.

1.3.3.2 Fases de EORM

Esta clasificación permite comprender cómo EORM amplía el modelado orientado a objetos al incorporar relaciones de navegación propias de las aplicaciones hipermedia. En el análisis comparativo realizado por Molina Ríos y Zea Ordoñez (2017), EORM se caracteriza por trabajar con clases, relaciones orientadas a objetos y enlaces simples, navegacionales, nodo a nodo y tramo a nodo, lo que refuerza su orientación hacia la representación de vínculos dentro de sistemas web.

Fase 1: análisis

En la fase inicial de análisis, EORM estudia el sistema desde una perspectiva orientada a objetos. En esta etapa se identifican los objetos principales del dominio, sus atributos, comportamientos y relaciones, sin incorporar todavía los aspectos hipermedia. El propósito es construir un modelo de objetos que represente la organización conceptual del sistema y sirva como base para las fases posteriores. Escalona Cuaresma (2001) señala que en esta primera etapa no se consideran todavía la navegación ni la interfaz, ya que estos elementos se desarrollan en fases posteriores.

Este modelo permite organizar los componentes del sistema de manera modular y establecer una representación inicial del dominio antes de definir enlaces o recorridos de navegación. Por esta razón, la fase de análisis funciona como una base conceptual sobre la cual se agregan posteriormente los elementos hipermedia. En este sentido, EORM primero identifica la estructura del sistema y después incorpora las relaciones navegacionales que permitirán conectar los objetos dentro de la aplicación.

Fase 2: diseño

En la fase de diseño, el modelo de objetos se enriquece mediante la incorporación de semántica hipermedia a las relaciones previamente identificadas. Esto significa que las relaciones entre objetos dejan de representarse únicamente como asociaciones estructurales y pasan a convertirse en vínculos que permiten definir recorridos de navegación dentro del sistema. De esta manera, EORM integra el modelado orientado a objetos con elementos propios del diseño hipermedia.

Durante esta etapa, el modelo EORM refleja tanto la estructura de la información como las posibilidades de navegación ofrecidas por el sistema. Para ello, se utiliza una librería o repositorio de clases de enlaces, donde se especifican operaciones asociadas a cada enlace, como creación, eliminación o recorrido, además de atributos vinculados con su presentación. Escalona Cuaresma (2001) explica que en esta fase el modelo de objetos se modifica mediante la adición de semántica suficiente para representar enlaces, incorporando clases como *simpleLink*, *navigationalLink*, *nodeToNode*, *spanToNode*, *structureLink*, *setLink* y *listLink*.

La importancia de esta fase radica en que permite diferenciar entre el modelo conceptual del sistema y su estructura navegacional. Al agregar enlaces hipermedia, el diseño puede representar rutas de acceso, conexiones entre objetos y formas de exploración de la información. Esto resulta útil en aplicaciones donde la relación entre objetos no solo tiene valor estructural, sino también un efecto directo en la forma en que los usuarios recorren el sistema.

Fase 3: construcción

La fase de construcción consiste en transformar los esquemas diseñados en elementos implementables dentro del entorno de desarrollo seleccionado. En esta etapa, los modelos definidos en las fases anteriores se convierten en código, estructuras de almacenamiento e interfaces que permiten materializar el diseño conceptual y navegacional en una aplicación funcional.

Además de la generación de código, esta fase contempla la preparación de la interfaz gráfica de usuario. Escalona Cuaresma (2001) indica que en esta etapa se prepara el código fuente correspondiente a las clases y a la interfaz gráfica; sin embargo, también señala que la metodología no ofrece recomendaciones especiales para esta fase, lo cual representa una de sus limitaciones.

En este sentido, la construcción no debe entenderse únicamente como programación, sino como la traducción de los modelos de objetos, enlaces y navegación hacia componentes funcionales. La coherencia entre lo diseñado y lo implementado resulta fundamental, ya que

las relaciones definidas en el modelo deben conservar su significado dentro de la aplicación final.

En conjunto, EORM se distingue por aplicar el paradigma orientado a objetos al diseño de aplicaciones hipermedia y por representar las relaciones entre objetos mediante clases de enlaces. Su principal aportación se encuentra en la separación entre la estructura conceptual y la estructura navegacional, lo que favorece la reutilización y el mantenimiento. No obstante, Escalona Cuaresma (2001) advierte que también presenta limitaciones, ya que aborda de forma específica el almacenamiento y la navegación, pero deja menos desarrollados aspectos como la funcionalidad, la interfaz y la captura de requisitos.

Dentro del análisis metodológico de esta investigación, EORM se considera una alternativa relevante para comprender el modelado orientado a objetos aplicado a sistemas hipermedia. Sin embargo, su enfoque se orienta con mayor fuerza al tratamiento de objetos y vínculos navegacionales que al análisis inicial de perfiles de usuario, por lo que se revisa principalmente como punto de comparación frente a metodologías centradas en el usuario.

1.3.4 OOWS

La metodología OOWS (*Object-Oriented Web Solutions*) es un enfoque de ingeniería web orientado al desarrollo de aplicaciones web a partir de modelos conceptuales. Esta metodología surge como una extensión del método orientado a objetos OO-Method, incorporando primitivas y modelos específicos para representar requisitos propios de las aplicaciones web, como la navegación, la presentación y la interacción del usuario con la información (Pastor et al., 2008).

OOWS se fundamenta en el modelado conceptual, por lo que busca describir el sistema antes de su implementación técnica. A través de este enfoque, el desarrollo no inicia directamente con la codificación, sino con la construcción de modelos que representan la estructura estática, el comportamiento, las funciones, la navegación y la presentación del sistema. Esto permite mantener una relación más clara entre los requisitos definidos y los elementos que posteriormente serán implementados en la aplicación (Castillo-Montes et al., 2018).

Una característica importante de OOWS es que separa el modelo funcional del modelo navegacional y del modelo de presentación. Esta separación permite analizar, por una parte,

la lógica del sistema y, por otra, la forma en que los usuarios recorrerán la aplicación y visualizarán la información. De acuerdo con Pastor et al. (2008), OOWS propone extensiones conceptuales para aplicaciones web mediante modelos que permiten representar tanto la navegación como los requisitos de interfaz.

El método también ha sido vinculado con enfoques de desarrollo dirigido por modelos, ya que sus especificaciones pueden transformarse progresivamente en componentes de implementación. En este sentido, OOWS no se limita a describir la estructura de una aplicación web, sino que busca facilitar el paso desde los modelos conceptuales hacia aplicaciones funcionales, manteniendo consistencia entre el análisis, el diseño y la construcción del sistema (Valderas et al., 2005).

Desde una perspectiva metodológica, OOWS resulta útil para proyectos que requieren una descripción detallada del sistema antes de la implementación, especialmente cuando se busca representar información, navegación y presentación mediante modelos formales. Sin embargo, al estar fuertemente orientada al modelado conceptual y al desarrollo dirigido por modelos, su aplicación puede resultar más compleja cuando se requiere una metodología más centrada en perfiles de usuario y necesidades de interacción desde las primeras etapas.

1.3.4.1 Fases de OOWS

Aunque OOWS se reconoce principalmente por su énfasis en el modelado conceptual, navegacional y de presentación, para efectos de esta revisión metodológica su proceso puede organizarse en cinco fases generales. Esta organización permite explicar la transición desde la identificación de requisitos hasta la validación de la aplicación, manteniendo relación con los modelos que caracterizan a esta metodología.

Fase 1: análisis de requisitos

En la fase de análisis de requisitos se identifican las necesidades del sistema y se establecen las funciones que la aplicación web deberá cumplir. Esta etapa permite reconocer los actores involucrados, los objetivos del sistema, los casos de uso y las restricciones que condicionan el desarrollo. En OOWS, este análisis sirve como punto de partida para construir los modelos que representarán tanto la lógica interna como los aspectos propios de la navegación web.

A partir de los requisitos obtenidos, se define una representación inicial del sistema que posteriormente será refinada mediante modelos conceptuales. Esta fase resulta importante porque permite evitar que el desarrollo avance directamente hacia la codificación sin una comprensión previa del dominio, los usuarios, las funciones esperadas y las restricciones técnicas. En metodologías basadas en modelos, esta claridad inicial es necesaria para que las etapas posteriores mantengan coherencia con los objetivos del sistema (Castillo-Montes et al., 2018).

Fase 2: diseño conceptual

El diseño conceptual constituye una de las etapas centrales de OOWS. En esta fase se construyen modelos que representan la estructura y el comportamiento del sistema, retomando elementos del OO-Method. Entre estos modelos se encuentran el modelo estructural, que define clases y relaciones; el modelo dinámico, que describe cambios de estado; el modelo funcional, que especifica transformaciones y operaciones; y el modelo de presentación, que organiza la interacción con el usuario (Castillo-Montes et al., 2018).

Esta etapa permite describir el sistema de manera independiente a la tecnología que posteriormente será utilizada para implementarlo. De esta forma, el diseño conceptual funciona como una representación formal del dominio de aplicación, permitiendo identificar las entidades principales, sus relaciones, reglas de comportamiento y funciones. Para aplicaciones web, OOWS amplía este modelado con elementos relacionados con navegación y presentación, los cuales son necesarios para representar cómo los usuarios accederán a la información.

Fase 3: diseño navegacional

El diseño navegacional permite definir la forma en que los usuarios recorrerán la aplicación web. En esta etapa se construyen estructuras de acceso, contextos de navegación y enlaces que conectan los diferentes elementos del sistema. La finalidad es representar los caminos mediante los cuales los usuarios podrán consultar información, ejecutar acciones y desplazarse entre las distintas secciones de la aplicación.

En OOWS, el modelo navegacional se construye a partir del modelo conceptual, por lo que la navegación no se define de manera aislada, sino como una extensión de la estructura del

sistema. Esta característica permite mantener coherencia entre los objetos representados en el modelo y las rutas disponibles para el usuario. Pastor et al. (2008) señalan que OOWS incorpora abstracciones específicas para capturar requisitos de navegación y presentación en aplicaciones web.

Fase 4: diseño de presentación

El diseño de presentación se enfoca en definir cómo será mostrada la información al usuario dentro de la aplicación web. En esta fase se establecen criterios relacionados con la organización visual de los contenidos, la disposición de elementos, la interacción con formularios, menús, enlaces y componentes de interfaz. Su propósito es transformar los modelos conceptuales y navegacionales en una representación más cercana a la experiencia final del usuario.

Esta fase permite diferenciar entre la estructura lógica de la aplicación y la manera en que dicha estructura será presentada. En este sentido, el modelo de presentación contribuye a definir la apariencia, distribución y comportamiento visual de los elementos, sin romper la relación con los modelos anteriores. OOWS reconoce esta separación como parte del desarrollo de aplicaciones web, donde la navegación y la presentación requieren una atención específica además del modelado funcional del sistema (Pastor et al., 2008).

Fase 5: implementación

La fase de implementación consiste en transformar los modelos definidos en componentes funcionales de la aplicación web. En esta etapa se genera o desarrolla el código necesario para materializar la estructura, navegación, presentación y comportamiento previamente especificados. En enfoques basados en modelos, como OOWS, la implementación busca conservar la relación entre las decisiones tomadas durante el diseño conceptual y el sistema finalmente construido.

Además, OOWS ha sido utilizado dentro de propuestas de desarrollo dirigido por modelos, en las que los modelos conceptuales pueden servir como base para generar aplicaciones o componentes tecnológicos específicos. Valderas et al. (2005) explican que OOWS ha sido vinculado con enfoques MDA, donde el modelado conceptual facilita la transición hacia plataformas de implementación mediante transformaciones sucesivas.

En conjunto, OOWS se distingue por su énfasis en el modelado conceptual de aplicaciones web y por la separación entre estructura, navegación y presentación. Esta metodología ofrece una base sólida para representar formalmente los elementos del sistema antes de su implementación, lo que puede favorecer la consistencia del desarrollo. No obstante, dentro del análisis metodológico de esta investigación, OOWS se considera principalmente como una alternativa de comparación, debido a que su enfoque se orienta con mayor fuerza al modelado conceptual y a la construcción de soluciones web a partir de modelos que al análisis inicial de grupos de usuarios.

1.3.5 Comparación entre metodologías

Realizar la elección de una metodología para el desarrollo de un LMS requiere una planificación y ejecución cuidadosa, ya que debe seleccionarse una metodología que garantice un diseño robusto, proporcione una excelente experiencia de usuario y asegure una gestión efectiva de la información.

Comparar las metodologías WSDM, OOWS, EORM y RMM es esencial para identificar cuál de ellas se alinea mejor con los objetivos y requisitos específicos de un LMS. La elección de la metodología adecuada puede influir significativamente en la eficiencia del desarrollo, la facilidad de uso del sistema final, y su capacidad para adaptarse a futuros cambios y necesidades.

A continuación, en la Tabla 1.1, se muestran las características, ventajas y limitaciones de cada una de estas metodologías, proporcionando una visión clara para elegir la más adecuada según los objetivos específicos del LMS en desarrollo.

Tabla 1.1

Comparación entre las metodologías descritas

Característica	OOWS	EORM	WSDM	RMM
Enfoque principal	Modelado conceptual de soluciones web orientadas a objetos	Modelado orientado a objetos con enlaces hipermedia	Diseño de sitios web centrado en el usuario	Diseño estructurado de aplicaciones hipermedia
Fases del proceso	Análisis de requisitos, diseño conceptual, diseño navegacional, diseño de presentación e implementación	Análisis, diseño y construcción	Modelo de usuario, diseño conceptual, diseño de implementación e implementación	Modelo E-R, diseño de slices, navegación, protocolo de conversión, interfaz de usuario, comportamiento en ejecución y pruebas
Principios clave	Modelado conceptual, modularidad, navegación y presentación	Objetos, relaciones y enlaces hipermedia	Identificación de usuarios, diseño conceptual y navegación orientada a perfiles	Entidad-relación, slices, RMDM y navegación hipermedia
Metodología de desarrollo	Basada en modelos conceptuales	Iterativa y centrada en objetos	Progresiva y centrada en usuarios	Estructurada y orientada a navegación
Herramientas y tecnologías	UML, modelos conceptuales, patrones de diseño	Modelado orientado a objetos, clases de enlaces y relaciones hipermedia	Herramientas de diseño web, modelos de usuario y navegación	Modelo entidad-relación, RMDM, *slices* y estructuras de navegación
Casos de uso	Aplicaciones web complejas basadas en modelos	Aplicaciones hipermedia orientadas a objetos	Sitios web con distintos perfiles de usuario	Aplicaciones hipermedia estructuradas
Ventajas	Favorece la trazabilidad entre modelos e implementación	Separa estructura conceptual y navegación	Prioriza necesidades de usuario desde etapas iniciales	Organiza información y rutas de navegación de forma estructurada
Desafíos	Puede resultar complejo por su orientación al modelado	Puede dejar menos desarrollada la captura de requisitos e interfaz	Requiere identificar adecuadamente los perfiles de usuario	Se enfoca más en estructura y navegación que en necesidades del usuario

Nota. *Elaboración propia a partir del análisis de las metodologías RMM, WSDM, EORM y OOWS descritas en el presente capítulo.*

La metodología WSDM resulta pertinente para el desarrollo de Gasly LMS debido a su enfoque centrado en la identificación de usuarios, sus necesidades y sus formas de interacción con el sistema. Esta orientación permite que el diseño de la plataforma no dependa únicamente de criterios técnicos, sino también de los perfiles que harán uso de ella, como estudiantes, instructores y administradores.

A partir de esta metodología, el desarrollo puede organizarse de manera progresiva, desde el análisis de usuarios hasta el diseño conceptual, el diseño de implementación y la construcción técnica. Esto favorece una relación más clara entre las necesidades identificadas y las funcionalidades incorporadas en la plataforma.

De esta manera, WSDM se considera adecuada para el proyecto, ya que permite estructurar el desarrollo de un sistema web donde la navegación, la organización de contenidos y la experiencia de uso son elementos relevantes para el funcionamiento del LMS.

1.4 Estado del arte

Los sistemas de gestión del aprendizaje han adquirido relevancia dentro del ámbito educativo y de capacitación debido a su capacidad para organizar contenidos, administrar usuarios, distribuir materiales y dar seguimiento a los procesos formativos en entornos digitales. Su incorporación en instituciones educativas, organizaciones y áreas de formación responde a la necesidad de contar con espacios accesibles, estructurados y capaces de apoyar modalidades presenciales, híbridas o completamente en línea.

A partir de la expansión de la educación digital, las plataformas LMS han dejado de ser únicamente repositorios de materiales para convertirse en entornos integrales de gestión académica. En ellos es posible administrar cursos, registrar participantes, publicar recursos, asignar actividades, aplicar evaluaciones, facilitar la comunicación y monitorear el avance de los usuarios. Estas funciones permiten que docentes, instructores o responsables de formación gestionen el proceso educativo de manera más ordenada, mientras que los estudiantes pueden acceder a contenidos y actividades desde distintos dispositivos con conexión a Internet.

Desde finales de los años noventa, las plataformas tecnológicas aplicadas a la educación han permitido al profesorado crear, administrar y distribuir cursos mediante Internet. Sánchez

Rodríguez (2005) señala que estos espacios institucionales reúnen aplicaciones informáticas instaladas en servidores, cuya finalidad es facilitar la gestión educativa en línea. En este sentido, los LMS se han consolidado como herramientas que apoyan la organización de experiencias formativas, especialmente cuando se requiere centralizar información, recursos y actividades dentro de un mismo entorno.

Entre las funciones principales de un sistema de gestión del aprendizaje se encuentran la administración de usuarios, la gestión de cursos, la publicación de contenidos, la incorporación de herramientas de comunicación y el seguimiento del desempeño. Lozano (2008) identifica como funciones relevantes la gestión de participantes, la administración de cursos y la comunicación entre usuarios. Estas características resultan esenciales para estructurar procesos de enseñanza y aprendizaje en ambientes virtuales, ya que permiten articular los distintos componentes que intervienen en una experiencia formativa.

Además de la administración de contenidos, los LMS favorecen la interacción entre los participantes mediante herramientas como foros de discusión, mensajería, chats o espacios colaborativos. Estos recursos permiten ampliar la comunicación más allá de la entrega de materiales, promoviendo la participación, el intercambio de ideas y la construcción colectiva del conocimiento. En consecuencia, una plataforma de aprendizaje no solo debe facilitar el acceso a recursos, sino también propiciar condiciones para la interacción académica y el acompañamiento durante el proceso formativo.

Otra característica relevante de estas plataformas es la posibilidad de dar seguimiento al avance de los usuarios. Mediante registros de acceso, actividades completadas, evaluaciones realizadas y calificaciones obtenidas, los LMS permiten observar el desempeño de los participantes y detectar necesidades de apoyo. Esta información puede ser utilizada por docentes o instructores para ajustar estrategias, proporcionar retroalimentación y fortalecer el proceso de aprendizaje.

No obstante, la implementación de un LMS también implica desafíos. Entre ellos se encuentran la capacitación de docentes y usuarios, la disponibilidad de infraestructura tecnológica, la seguridad de la información, la privacidad de los datos y el acceso equitativo a los recursos digitales. Estos factores deben considerarse desde la planeación del sistema,

ya que pueden influir directamente en su adopción, uso continuo y efectividad dentro del entorno educativo o institucional.

En la actualidad, existen diversas plataformas de gestión del aprendizaje que han sido utilizadas en contextos educativos y de capacitación. Algunas se caracterizan por ser de código abierto, otras por su enfoque comercial, su capacidad de integración, su accesibilidad o sus herramientas de evaluación y seguimiento. A continuación, se describen algunas plataformas representativas, con el propósito de identificar sus principales características y establecer una base comparativa para el desarrollo de Gasly LMS.

1.4.1 Moodle

Moodle es una de las plataformas de gestión del aprendizaje más utilizadas en contextos educativos y de capacitación. Su relevancia se relaciona con su capacidad para crear, administrar y distribuir cursos en línea mediante un entorno flexible, configurable y orientado al trabajo colaborativo. Al tratarse de un sistema de código abierto, puede ser descargado, utilizado, modificado y adaptado de acuerdo con las necesidades de instituciones educativas, áreas de formación o docentes que requieren organizar experiencias de aprendizaje en línea (Llorente Cejudo, 2007).

La plataforma fue creada por Martin Dougiamas y se ha desarrollado con una base pedagógica vinculada al constructivismo social. Desde esta perspectiva, el aprendizaje no se concibe únicamente como transmisión de información, sino como un proceso en el que el estudiante participa activamente en la construcción de conocimiento. Por ello, Moodle no solo permite publicar materiales, sino también diseñar actividades, promover la comunicación entre participantes y generar espacios de interacción académica (Santamaría, 2012).

Una de sus principales características es la organización modular. Moodle permite estructurar cursos mediante secciones, temas, recursos y actividades, lo que facilita la administración del contenido educativo. Entre sus herramientas se encuentran tareas, cuestionarios, foros, chats, glosarios, wikis, talleres, encuestas y recursos como archivos, páginas web o directorios. Esta variedad de módulos permite que el docente configure diferentes estrategias

de enseñanza, evaluación y seguimiento dentro de un mismo entorno digital (Cosano Rivas, s. f.).

En cuanto a la comunicación y colaboración, Moodle ofrece herramientas como foros de discusión, mensajería, chat y actividades grupales. Estos recursos favorecen la interacción entre docentes y estudiantes, así como el intercambio de ideas entre los propios participantes. De acuerdo con Ros Martínez de Lahidalga (2008), la plataforma resulta útil para gestionar asignaturas, publicar contenidos multimedia, evaluar tareas, realizar exámenes en línea y fomentar el autoaprendizaje y el aprendizaje cooperativo.

La gestión de cursos y evaluaciones también representa una de sus fortalezas. Los docentes pueden crear cuestionarios con distintos tipos de preguntas, recibir tareas en línea, calificar actividades, revisar el avance de los estudiantes y administrar recursos asociados al curso. Además, la plataforma permite registrar calificaciones, controlar fechas de entrega y consultar información sobre la participación de los usuarios. Estas funciones facilitan la planificación y el seguimiento de actividades académicas dentro de entornos virtuales (Cosano Rivas, s. f.).

Otra característica relevante es su capacidad de personalización. Moodle puede adaptarse mediante temas, complementos y configuraciones administrativas que modifican tanto la apariencia como la funcionalidad del entorno. Esta flexibilidad permite que diferentes instituciones ajusten la plataforma a sus necesidades específicas, aunque también puede implicar una curva de aprendizaje para quienes administran el sistema o configuran cursos por primera vez.

No obstante, Moodle también presenta algunos desafíos. Su amplitud funcional puede resultar compleja para usuarios sin experiencia previa en plataformas educativas, especialmente cuando se requiere configurar cursos, permisos, complementos o actividades avanzadas. Además, su uso efectivo puede depender de factores como la disponibilidad de conexión a Internet, la capacitación de docentes y estudiantes, y la familiaridad de los usuarios con entornos digitales.

En términos generales, Moodle constituye una plataforma robusta para la gestión del aprendizaje, ya que integra herramientas de administración de cursos, comunicación,

evaluación, seguimiento y organización de recursos. Estas características explican su uso extendido en distintos contextos educativos y la convierten en una referencia importante para analizar sistemas de gestión del aprendizaje orientados a la enseñanza en línea y semipresencial (Bailón & Vélez, 2021).

En la Figura 1.4 se muestra el logotipo de Moodle, mientras que en la Figura 1.5 se presenta un ejemplo de un curso dentro de la plataforma.

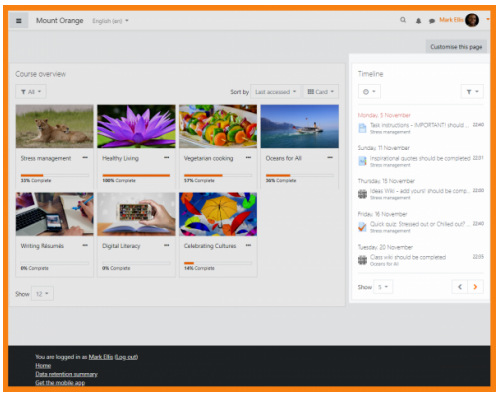
Figura 1.4.

Logo de Moodle

Nota. *Tomado de Moodle: online learning with the world's most popular LMS, por Moodle, s. f., Moodle. Recuperado el 10 de octubre de 2024, de https://moodle.com/*

Figura 1.5.

Ejemplo de la vista general de un curso en Moodle



Nota. *Tomado de Course overview and timeline, por Moodle Docs, s. f., Moodle Docs. Recuperado el 3 de mayo de 2026, de https://docs.moodle.org/all/es/images_es/thumb/5/59/docstimeline.png/600px-docstimeline.png*

1.4.2 Canvas

Canvas es una plataforma de gestión del aprendizaje desarrollada por Instructure y lanzada en 2011. Dentro del estado del arte, su revisión resulta útil porque permite observar el funcionamiento de un LMS ampliamente utilizado en contextos educativos, especialmente en aspectos relacionados con la administración de cursos, la comunicación entre usuarios, el seguimiento académico y la integración de herramientas digitales. Rubio Fernández et al. (2017) describen Canvas como un LMS basado en la nube, creado por Instructure, que ha sido empleado como entorno para la obtención de información académica a partir de datos generados por los estudiantes dentro de la plataforma.

Una de sus características relevantes es la posibilidad de concentrar información académica y generar datos derivados de la interacción de los usuarios. En este sentido, Canvas no solo funciona como espacio para publicar contenidos, sino también como un entorno desde el cual pueden analizarse mensajes en foros, respuestas a cuestionarios y otros indicadores relacionados con el proceso de aprendizaje. Rubio Fernández et al. (2017) señalan que este tipo de datos puede utilizarse para obtener información académica que apoye a los docentes en la mejora del aprendizaje.

En cuanto a su uso por parte del profesorado, Canvas puede presentar resultados distintos según la experiencia previa de los docentes con plataformas LMS y según la disciplina académica en la que se utilice. Fathema y Akanda (2020) identificaron diferencias significativas en el uso de Canvas por parte de instructores de distintas áreas, así como la necesidad de ofrecer capacitación general y específica de acuerdo con el nivel de experiencia de los usuarios. Esto muestra que la adopción de un LMS no depende únicamente de sus funciones, sino también de la preparación y necesidades de quienes lo utilizan.

Canvas incorpora herramientas para publicar materiales, entregar tareas, aplicar cuestionarios, comunicarse mediante foros o mensajes y dar seguimiento al avance de los estudiantes. Estas funciones lo colocan como una referencia importante dentro de los LMS actuales; sin embargo, su aprovechamiento puede variar de acuerdo con la forma en que cada institución configure los cursos y con la consistencia en el uso de la plataforma. Williams (2022), en un estudio centrado en la experiencia de estudiantes con Canvas, señala que los

problemas de satisfacción pueden no depender únicamente del LMS, sino de la manera inconsistente en que se organiza y utiliza dentro de una institución.

También se ha estudiado Canvas desde la perspectiva de aceptación y percepción estudiantil. Lobo et al. (2025) analizan factores que influyen en la aceptación de Canvas LMS y Zoom como herramientas educativas, lo que permite observar que la percepción de los estudiantes resulta un elemento importante para valorar la utilidad de este tipo de plataformas en contextos formativos.

Desde una perspectiva comparativa, Canvas ha sido revisado junto con otras plataformas como Moodle. Segovia-García (2024) realizaron un estudio comparativo entre ambos LMS mediante una matriz con más de noventa criterios, lo que permite entender que estas herramientas pueden evaluarse desde múltiples dimensiones y que ninguna plataforma debe considerarse como una solución única para todos los contextos.

No obstante, el uso de Canvas también presenta desafíos. Oudat y Othman (2024) señalan que, aunque Canvas puede favorecer el aprendizaje asíncrono, la colaboración y el análisis del desempeño estudiantil, también pueden presentarse dificultades relacionadas con problemas técnicos, accesibilidad, privacidad y comprensión de contenidos. Por ello, su análisis resulta útil para reconocer tanto funciones valiosas como aspectos que deben considerarse al desarrollar nuevas propuestas de LMS.

La Figura 1.6 muestra el logotipo de Canvas, mientras que la Figura 1.7 presenta un ejemplo de su tablero principal.

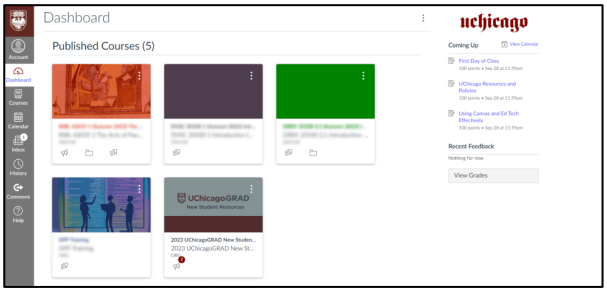
Figura 1.6.

Logo de Canvas.

Nota. *Tomado de Canvas LMS logo, por Instructure, s. f., Instructure. Recuperado el 3 de mayo de 2026, de https://www.instructure.com/about/brand-guide/canvas*

Figura 1.7.

Tablero principal de Canvas.



Nota. *Tomado de Canvas, por Academic Technology Solutions, 2023, University of Chicago. Recuperado el 3 de mayo de 2026, de https://academictech.uchicago.edu/files/2023/10/DashboardBlur-980x460.png*

1.4.3 Edu 2.0

Edu 2.0 es una plataforma educativa orientada a la gestión de cursos, recursos y actividades en línea. Su revisión dentro del estado del arte resulta pertinente porque representa una alternativa de LMS vinculada con el uso de herramientas Web 2.0, en las que la comunicación, la interacción y la participación de los usuarios forman parte del entorno virtual de aprendizaje. En el contexto de educación superior, Chávez Martínez (2016) señala que el uso de plataformas educativas se ha incorporado como parte de los procesos de enseñanza y aprendizaje mediados por tecnologías digitales.

Entre sus funciones se encuentran la organización de contenidos, la publicación de materiales, la asignación de actividades y la habilitación de espacios de comunicación entre docentes y estudiantes. Estas características permiten que el profesorado administre cursos desde un entorno digital y que los estudiantes accedan a recursos asociados a sus actividades académicas. Asimismo, las herramientas apoyadas en Edu 2.0 pueden incluir recursos interactivos como blogs, actividades colaborativas y espacios de participación, aunque su aprovechamiento depende de la forma en que sean integrados en la planeación didáctica (Páez Cantillo, 2023).

La plataforma puede utilizarse en contextos presenciales, semipresenciales o a distancia, dependiendo de las condiciones institucionales y de la estrategia educativa adoptada. En este sentido, se relaciona con los planteamientos del *e-learning* 2.0, donde las herramientas

digitales no se limitan a distribuir información, sino que también pueden favorecer la colaboración, la interacción y la construcción colectiva del conocimiento. No obstante, la incorporación de herramientas Web 2.0 en educación superior requiere considerar la preparación de docentes y estudiantes, así como la forma en que dichas herramientas se integran en las actividades académicas, ya que su aprovechamiento depende del contexto institucional y de la estrategia pedagógica utilizada (Bennett et al., 2012).

No obstante, la adopción de Edu 2.0 no implica automáticamente una mejora en los procesos educativos. Su efectividad depende de factores como la capacitación de los usuarios, la claridad en la organización de los cursos, la disponibilidad de conectividad y la manera en que docentes y estudiantes se apropian de sus herramientas. En el estudio presentado por Chávez Martínez (2016), se reconoce que la transición hacia entornos colaborativos mediante Edu 2.0 no siempre resulta sencilla, ya que los resultados pueden variar según el contexto, los grupos, los profesores y los alumnos.

De esta manera, Edu 2.0 se considera una plataforma de referencia dentro del análisis de LMS, especialmente por su relación con herramientas colaborativas y recursos interactivos. Su revisión permite identificar funciones útiles para la administración de cursos y la comunicación académica, pero también evidencia la importancia de diseñar plataformas claras, organizadas y adecuadas a las necesidades reales de los usuarios.

Para complementar la descripción de esta plataforma, en la Figura 1.8 se presenta su logotipo, mientras que en la Figura 1.9 se muestra un ejemplo de su interfaz.

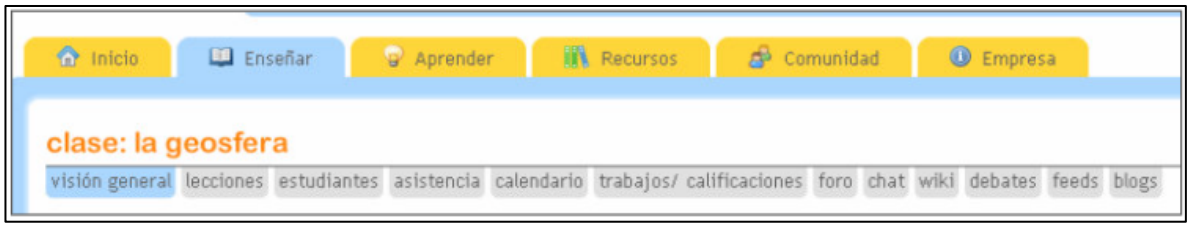
Figura 1.8.

Logo de Edu 2.0.

Nota. *Tomado de Images 41 [Imagen], por D. Zarta Rojas, 2013, WordPress. https://danielazartarojas.files.wordpress.com/2013/02/images-41.jpg*

Figura 1.9.

Muestra del contenido de una clase en Edu 2.0



Nota. *Tomado de Edu 2.0 Plataforma en línea aplicada a la educación superior, por R. E. Cuevas Valencia y E. Zenón Rodríguez, 2013, Universidad Autónoma de Guerrero, Figura 3.6.*

1.4.4 Blackboard

Blackboard es una plataforma de gestión del aprendizaje utilizada en instituciones educativas para organizar cursos, distribuir materiales, administrar actividades y facilitar la comunicación entre docentes y estudiantes. Su revisión dentro del estado del arte permite analizar un LMS orientado a la gestión académica en entornos virtuales, especialmente en contextos de educación superior, formación a distancia y modalidades semipresenciales. Ferreiro Martínez et al. (2013) analizan el uso de Blackboard en una modalidad semipresencial dentro de un caso práctico universitario, lo que permite observar su aplicación como recurso de apoyo en procesos educativos mediados por tecnología.

Entre sus funciones se encuentran la publicación de contenidos, la creación de actividades, el uso de herramientas de comunicación y la gestión de evaluaciones. La guía de uso de Blackboard para profesores de la Universidad de las Américas Puebla describe recursos asociados a la administración de cursos, manejo de contenidos, actividades, evaluaciones y comunicación dentro de la plataforma, lo que permite observar su orientación hacia la organización del trabajo docente en línea (UDLAP, s. f.).

La plataforma también incorpora herramientas que permiten estructurar cursos mediante recursos, actividades y espacios de interacción. Aliaga y Dávila (2021) señalan que Blackboard integra elementos como archivos, carpetas, páginas, enlaces, paquetes SCORM,

foros, wikis, blogs, encuestas, cuestionarios y tareas. Estas funciones permiten organizar parte del proceso formativo dentro de un entorno digital, aunque su aprovechamiento depende de la planeación pedagógica y del grado de familiaridad de los usuarios con la plataforma.

En relación con la comunicación, Blackboard puede complementarse con herramientas como Blackboard Collaborate, orientadas a la interacción síncrona mediante videoconferencias y espacios de trabajo en línea. Villalón et al. (2019) analizan el uso de Blackboard Collaborate en una universidad a distancia, destacando su aplicación en contextos donde la comunicación virtual forma parte del proceso educativo. No obstante, este tipo de herramientas requiere una adecuada organización de las sesiones y una participación activa de los usuarios para que su uso sea significativo.

En cuanto al seguimiento académico, Blackboard ofrece opciones relacionadas con calificaciones, retroalimentación, revisión de entregas y consulta de resultados. Aliaga y Dávila (2021) describen que el panel de control y el calificador permiten revisar entregas, registrar calificaciones, consultar informes y administrar actividades dentro del curso. Sin embargo, estas funciones no sustituyen la claridad de las instrucciones ni el acompañamiento docente, ya que el aprendizaje depende también de la forma en que se diseñen los contenidos y actividades.

Desde una perspectiva más general, el uso de plataformas digitales en la enseñanza universitaria requiere considerar no solo las herramientas disponibles, sino también su integración en la práctica educativa. De Pablos et al. (2019) señalan que las plataformas digitales deben analizarse desde sus usos educativos y no únicamente desde sus funcionalidades técnicas, ya que su impacto depende de la manera en que se incorporan en los procesos de enseñanza y aprendizaje.

A pesar de sus funciones, Blackboard también presenta desafíos. La amplitud de herramientas puede requerir capacitación previa para docentes y estudiantes, además de tiempo para la adaptación y configuración de los cursos. Aliaga y Dávila (2021) advierten que el desconocimiento del propósito de una plataforma virtual puede derivar en desinterés, incumplimiento de actividades o bajo aprovechamiento de sus recursos, especialmente cuando el docente no está familiarizado con su manejo.

En este sentido, Blackboard se considera una plataforma de referencia dentro del análisis de LMS, no como un modelo definitivo. Su revisión permite identificar funciones relacionadas con administración de cursos, comunicación, evaluación y seguimiento; al mismo tiempo, evidencia la importancia de diseñar entornos claros, accesibles y adecuados a necesidades específicas. Por ello, su análisis contribuye a reconocer tanto características aprovechables como limitaciones que deben considerarse en nuevas propuestas de sistemas de gestión del aprendizaje.

En la Figura 1.10 se presenta el logotipo de Blackboard, mientras que en la Figura 1.11 se muestra un ejemplo de su interfaz principal.

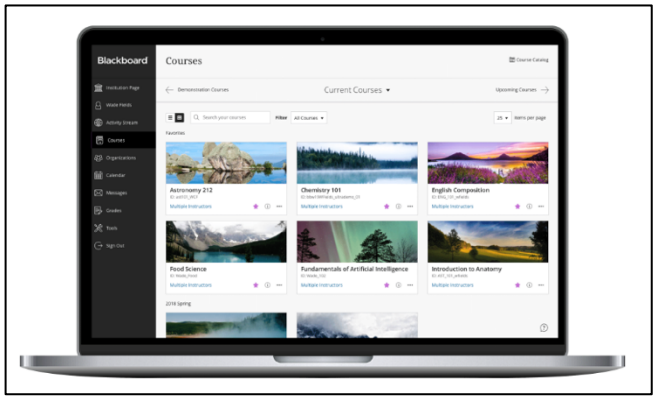
Figura 1.10.

Logo de Blackboard.

Nota. *Tomado de Blackboard, por MSMK, 2024, MSMK University. Recuperado el 25 de mayo de 2025, de https://msmk.university/wp-content/uploads/2024/09/Blackboard-learn.png*

Figura 1.11.

Página de cursos de Blackboard.



Nota. *Tomado de Blackboard, por MSMK, 2024, MSMK University. Recuperado el 25 de mayo de 2025, de https://msmk.university/blackboard/*

1.4.5 Claroline

Claroline es una plataforma de gestión del aprendizaje de código abierto orientada a la creación y administración de cursos en línea. Su desarrollo se originó en la Universidad Católica de Lovaina, en Bélgica, y se ha caracterizado por ofrecer un entorno enfocado en la organización de recursos educativos, actividades y espacios de colaboración dentro de cursos virtuales o semipresenciales (Bendezú Paytán, 2018).

A diferencia de otros LMS con estructuras más amplias o complejas, Claroline se presenta como una alternativa con herramientas básicas para la gestión educativa en línea. Entre sus funciones se encuentran la publicación de documentos en distintos formatos, la administración de enlaces, la creación de ejercicios, la asignación de tareas, el uso de foros, la programación de actividades mediante agenda y la aplicación de evaluaciones. Estas características permiten que docentes y estudiantes cuenten con un espacio digital para distribuir materiales, entregar trabajos y dar seguimiento a determinadas actividades académicas (Santos Regalado et al., 2022).

Asimismo, Claroline puede utilizarse como apoyo en procesos de enseñanza presencial, híbrida o completamente virtual, ya que facilita la concentración de materiales y actividades en un mismo entorno. No obstante, su alcance debe entenderse en función de las necesidades de cada institución, debido a que su simplicidad puede representar una ventaja para ciertos contextos, pero también una limitación cuando se requieren funciones más avanzadas de gestión, analítica o personalización.

En este sentido, Claroline resulta relevante dentro del análisis de plataformas LMS por representar una opción de software libre orientada a la administración básica de cursos y recursos educativos. Su revisión permite identificar cómo algunas plataformas priorizan la facilidad de uso, la organización de contenidos y la colaboración entre usuarios, elementos que forman parte de la evolución de los entornos virtuales de aprendizaje.

La Figura 1.12 presenta el logotipo de Claroline. Por su parte, la Figura 1.13 muestra un ejemplo del área de trabajo dentro de la plataforma, en la cual se observan recursos organizados como evaluaciones, actividades, rutas de aprendizaje y archivos de texto.

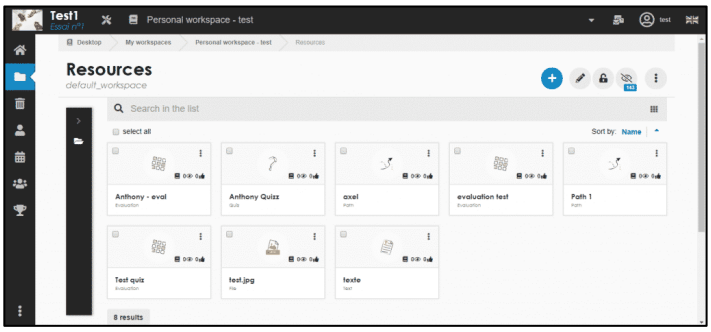
Figura 1.12.

Logotipo de Claroline

Nota. *Adaptado de Claroline [Imagen], por Sistema LMS Wiki, s. f., Fandom.*

Figura 1.13.

Ejemplo de área de trabajo en Claroline



Nota. *Adaptado de User experience [Imagen], por Bit4learn, 2019, Bit4learn*.

1.4.6 ATutor

ATutor es una plataforma de gestión del aprendizaje de código abierto orientada a la creación, administración y distribución de cursos en línea. Su funcionamiento se relaciona con la organización de contenidos educativos, la gestión de usuarios, la publicación de recursos y el uso de herramientas de comunicación y evaluación dentro de un entorno virtual de aprendizaje. En este sentido, Cervera Llop (2007) analiza su implantación como parte de una propuesta de e-learning, destacando su utilidad para estructurar cursos y administrar materiales educativos en línea.

Una de las características asociadas con ATutor es su atención a la accesibilidad, aspecto relevante dentro de los entornos educativos digitales, debido a que permite considerar las

condiciones de uso de distintos perfiles de usuarios. Sin embargo, esta característica no debe entenderse como suficiente por sí misma, ya que la accesibilidad de un curso también depende de la forma en que se diseñen los contenidos, actividades y recursos incorporados por los docentes.

En cuanto a sus funciones, ATutor permite organizar cursos mediante herramientas para publicar documentos, estructurar contenidos, configurar actividades, aplicar evaluaciones y utilizar espacios de comunicación. Estas funciones lo ubican dentro de los LMS orientados a apoyar procesos de enseñanza en línea o semipresenciales, especialmente cuando se requiere concentrar materiales y actividades en una misma plataforma.

Desde una perspectiva comparativa, Ardila Muñoz et al. (2015) evaluaron ATutor junto con Moodle, Claroline, Chamilo y una plataforma institucional, considerando dimensiones relacionadas con el modelo pedagógico, el usuario y los aspectos técnicos. En dicho estudio se señala que la selección de un LMS debe realizarse de acuerdo con las necesidades institucionales, los estándares de formación en línea y los criterios de calidad del software. Asimismo, los autores identificaron que ATutor obtuvo resultados favorables en algunos aspectos técnicos, aunque también presentó limitaciones en factores como la evaluación formativa, la usabilidad y las herramientas disponibles para apoyar la relación enseñanza-aprendizaje.

De esta manera, ATutor resulta pertinente dentro del análisis de plataformas LMS, no como una solución superior frente a otras alternativas, sino como un ejemplo de sistema de código abierto que permite revisar la importancia de la accesibilidad, la organización de contenidos y la evaluación de plataformas a partir de criterios pedagógicos, técnicos e institucionales.

La Figura 1.14 presenta el logotipo de ATutor. Por su parte, la Figura 1.15 muestra el proceso de creación de un curso dentro de la plataforma, donde se observan campos de configuración como el nombre del curso, la descripción, los atajos y la estructura de navegación.

Figura 1.14.

Logo de ATutor.

Nota. *Adaptado de ATutor [Imagen], s. f., Blogger. https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEikRcbb6fiQCuHKrOmundSwEK7ay47fEFaQQAJTEgYAhGswlLNXue5Yh07kLdt3hzQSPwuSltTFbR3VV_KpdImD9nwkNFFq2O1KuLzGmhCp0NfwW8DDCmESNSEqiApchJwkszu1hyte02/s1600/ATT22.jpg*

Figura 1.15.

Creación de un curso en ATutor.

Nota. *Adaptado de ATutor [Captura de pantalla], s. f., ComparaSoftware. https://www.comparasoftware.com/image-assets/412/NDEyfHdwLWNvbnRlbnQvdXBsb2Fkcy8yMDE4LzA4L2NhcHR1cmVhdHV0b3IyLnBuZw.webp*

1.4.7 Comparación de plataformas de gestión del aprendizaje

Con la finalidad de contrastar las características generales de distintos sistemas de gestión del aprendizaje, en la Tabla 1.2 se presenta una comparación entre plataformas LMS ampliamente utilizadas y Gasly LMS, plataforma desarrollada como parte del presente proyecto. Esta comparación no tiene como propósito establecer una superioridad absoluta entre sistemas, sino identificar diferencias generales en cuanto a origen, tipo de

licenciamiento, gestión de cursos, herramientas de evaluación, seguimiento y facilidad de uso.

Tabla 1.2

Comparación entre sistemas de gestión del aprendizaje

Característica	Moodle	Canvas	Edu 2.0	Blackboard	ATutor	Gasly LMS
Lanzamiento	2002	2011	2006	1997	2002	2024
Desarrollador	Martin Dougiamas	Instructure	Graham Glass	Blackboard Inc.	Centro ATutor	Omar Morales González
Tipo de acceso o licencia	Código abierto	Código abierto / comercial	Comercial / versión gratuita y de pago	Comercial	Código abierto	En proceso de desarrollo con futura incorporación comercial.
Accesibilidad	Buena	Muy buena	Buena	Alta	Muy alta	Alta
Personalización	Alta	Alta	Moderada	Alta	Alta	Alta
Comunicación	Foros	Mensajería, videoconferencias	Chats, foros	Mensajería, videoconferencias	Foros, grupos de trabajo	Foros, avisos, mensajería
Gestión de cursos	Variada	Personalizados	Modular	Completa	Modular	Modular
Evaluaciones	Cuestionarios, tareas	Evaluaciones, rúbricas	Tareas, evaluaciones	Exámenes, rúbricas	Pruebas, calificación automática	Evaluaciones por capítulo
Seguimiento	Sí	Sí	Sí	Sí	Sí	Sí
Compatibilidad	Alta	Muy alta	Alta	Alta	Alta	Alta
Facilidad de uso	Moderada	Alta	Alta	Moderada	Alta	Alta
Comunidad	Extensa	Extensa	Moderada	Extensa	Extensa	No aplica

Característica	Moodle	Canvas	Edu 2.0	Blackboard	ATutor	Gasly LMS
Acceso actual	Sin costo de licencia	Versión gratuita y de pago	Versión gratuita y de pago	Costoso	Sin costo de licencia	Posible incorporación comercial futura

Nota. Elaboración propia con base en la revisión documental de las plataformas LMS analizadas y en las características implementadas en Gasly LMS. La información correspondiente a Gasly LMS se presenta de acuerdo con el estado actual del desarrollo del proyecto.

Capítulo 2. Modelo de usuario y diseño conceptual de WSDM

En este capítulo se abordará cada fase de la metodología WSDM en detalle, comenzando con la identificación de los usuarios y sus necesidades, seguido del diseño conceptual, y culminando con la implementación.

Las herramientas de creación de cursos estarán disponibles y podrán ser gestionadas por un administrador de alguna organización, ya sea con docentes o personal capacitado para generar contenido, o por un instructor independiente que maneje ambas partes.

La plataforma también ofrecerá funcionalidades de búsqueda, inscripción y estadísticas, junto con una sección para visualización de las distintas partes del curso, dirigida al usuario estudiante.

Finalmente, se explica cómo la metodología WSDM permite orientar el desarrollo del LMS a partir de la identificación de usuarios, sus necesidades y sus formas de interacción. Este enfoque contribuye a que el diseño conceptual y la implementación mantengan relación con los perfiles de estudiantes, instructores y administradores, favoreciendo una estructura clara para la organización de contenidos, navegación y funcionalidades principales.

2.1 Modelo de usuario

El primer paso en la implementación de WSDM es la identificación y clasificación de los usuarios que interactuarán con el sistema. Esta fase es crucial porque establece la base para todas las decisiones de diseño posteriores. Los usuarios de un LMS típicamente incluyen estudiantes, instructores y administradores, cada uno con necesidades y expectativas diferentes. Comprender estas necesidades es esencial para crear una plataforma que sea intuitiva y eficiente.

Identificar a los usuarios implica más que simplemente enumerar quiénes usarán el sistema. Requiere un análisis detallado de sus comportamientos, preferencias y objetivos. Los estudiantes pueden preferir diferentes métodos de aprendizaje (visual, auditivo, kinestésico), tener horarios variados y usar distintos dispositivos para acceder al contenido. Los instructores, por otro lado, pueden tener preferencias sobre cómo crear y gestionar el

contenido, y los administradores pueden necesitar herramientas específicas para la gestión y el mantenimiento del sistema.

La clasificación de los usuarios en grupos distintos permite una personalización más efectiva del LMS. Cada grupo puede tener interfaces y funcionalidades específicas adaptadas a sus necesidades. Por ejemplo, los estudiantes pueden necesitar acceso rápido a los cursos y evaluaciones, mientras que los instructores pueden necesitar herramientas para la creación y evaluación del contenido. Los administradores, a su vez, pueden requerir acceso a configuraciones avanzadas y reportes detallados para la gestión del sistema.

La clasificación general de los usuarios identificados se presenta en la Tabla 2.1.

Tabla 2.1.

Clasificación de los usuarios del sistema de gestión del aprendizaje

Tipo de usuario	Descripción	Ejemplos de roles
Estudiantes	Usuarios que consumen el contenido educativo	Estudiantes bachillerato o universidad, personal de una empresa.
Instructores	Usuarios que crean y gestionan el contenido	Profesores, tutores, expertos
Administradores	Usuarios que administran y mantienen el sistema	Técnicos de TI, coordinadores

Durante esta fase se llevan a cabo los siguientes pasos:

1. **Identificación de usuarios:**
    - Estudiantes: Usuarios que acceden a los cursos y recursos educativos. Estos usuarios buscan una plataforma intuitiva que facilite el acceso al contenido y las actividades del curso.

- Instructores: Encargados de crear y gestionar el contenido educativo. Los instructores necesitan herramientas para crear contenido interactivo, evaluar el progreso de los estudiantes y facilitar la comunicación.
- Administradores: Responsables de la administración y mantenimiento del sistema. Los administradores requieren acceso a configuraciones avanzadas, herramientas de gestión de usuarios y reportes detallados.

2. **Clasificación de usuarios:**
   - Estudiantes: Se pueden clasificar según su nivel de experiencia (principiante, intermedio, avanzado), área de estudio (ciencias, humanidades, tecnología, etc.), y otras características demográficas.
   - Instructores: Clasificación según su área de especialización, nivel educativo (escuela secundaria, educación superior, educación profesional), y experiencia en el uso de tecnología educativa.
   - Administradores: Clasificación según sus roles específicos dentro de la administración del LMS, como técnicos de TI, coordinadores de programas, y otros roles administrativos.
3. **Análisis de preferencias y comportamientos:**
   - Estudiantes: Preferencias de aprendizaje, horarios de estudio, y dispositivos utilizados para acceder a la plataforma. Esto incluye la identificación de patrones de uso y preferencias de interacción con el contenido.
   - Instructores: Métodos de enseñanza preferidos, herramientas de creación de contenido utilizadas, y necesidades específicas para la gestión del curso.
   - Administradores: Necesidades de gestión y mantenimiento, herramientas de administración utilizadas, y expectativas en cuanto a la seguridad y la privacidad de los datos.
4. **Definición de escenarios de uso:**
   - Estudiantes: Acceso a cursos, realización de evaluaciones, participación en foros de discusión, y acceso a recursos adicionales.

- Instructores: Creación de contenido educativo, evaluación del progreso de los estudiantes, gestión de foros de discusión, y comunicación con los estudiantes.
- Administradores: Configuración del sistema, gestión de usuarios, mantenimiento de la plataforma, y generación de reportes para el análisis del desempeño del LMS.

Una vez identificados los principales usuarios del sistema —estudiantes, instructores y administradores—, es importante señalar que estos roles pueden coincidir en una misma persona, aunque no necesariamente en todos los casos. Por ejemplo, un instructor puede administrar su propio contenido y, al mismo tiempo, participar como estudiante en otro curso. De igual manera, un administrador puede limitarse a gestionar la plataforma para apoyar el trabajo de docentes o instructores. Independientemente del escenario, la estructura del sistema se organiza en torno a estos tres perfiles principales.

2.2 Diseño conceptual

El diseño conceptual es una fase crítica en el desarrollo de un LMS (Sistema de Gestión del Aprendizaje), ya que define cómo se presentará la información a través de la plataforma y cómo interactuarán los usuarios con ella. Esta fase asegura que el sistema sea intuitivo, accesible y eficiente, facilitando una experiencia de usuario positiva.

Un diseño conceptual bien pensado no solo mejora la usabilidad, sino que también contribuye a la satisfacción general de los usuarios.

El primer paso en esta fase es la creación de una estructura jerárquica de la navegación. Esta estructura debe ser clara y lógica, permitiendo a los usuarios acceder rápidamente a las secciones más importantes del LMS. Por ejemplo, la página de inicio debe ofrecer accesos directos a los cursos, perfiles de usuario y herramientas de administración. Cada una de estas secciones debe estar organizada de manera que sea fácil de entender y utilizar.

Las páginas de índice y secciones juegan un papel crucial en el diseño conceptual. Estas páginas deben estar organizadas de manera que los usuarios puedan encontrar rápidamente la información que buscan. La página de inicio puede incluir un resumen de los cursos disponibles, noticias relevantes y accesos rápidos a las principales funcionalidades del LMS. Las páginas de curso deben proporcionar detalles completos sobre el contenido, incluyendo

el título del curso, la cantidad de capítulos disponibles, la categoría a la que pertenecen y el proceso que exista en caso de que aplique.

Durante esta fase se llevan a cabo los siguientes pasos:

1. Estructura jerárquica de la navegación:

a) Inicio

La página principal del LMS debe proporcionar un resumen claro y accesible de las principales secciones de la plataforma. Esto incluye enlaces a los cursos disponibles, perfiles de usuario y herramientas de administración. La página de inicio debe ser intuitiva y fácil de navegar, con accesos directos a las secciones más utilizadas.

b) Cursos

Esta sección debe listar todos los cursos disponibles, permitiendo a los usuarios buscar y filtrar cursos según sus intereses y necesidades. Cada curso debe tener una página dedicada con información detallada sobre el contenido, los objetivos y los recursos disponibles.

c) Perfil del Usuario

Cada usuario debe tener una página de perfil personalizada donde puedan gestionar su información personal, ver su progreso en los cursos y ajustar sus preferencias de usuario. Esta sección debe ser fácil de acceder y actualizar.

d) Administración

Esta sección está destinada a los administradores del LMS y debe incluir herramientas para la gestión de usuarios, configuración del sistema y generación de reportes. La interfaz de administración debe ser intuitiva y proporcionar acceso rápido a las funciones más importantes.

En las Figuras 2.1 y 2.2 se muestra el diseño navegacional de la aplicación, considerando el acceso a la ventana de autenticación, la página de cursos, el apartado de administración y las demás secciones principales. En ambos casos, la navegación parte de una página principal, desde la cual el usuario puede dirigirse a las rutas correspondientes mediante los botones o enlaces disponibles.

Figura 2.1.

Diseño navegacional de acceso a la plataforma

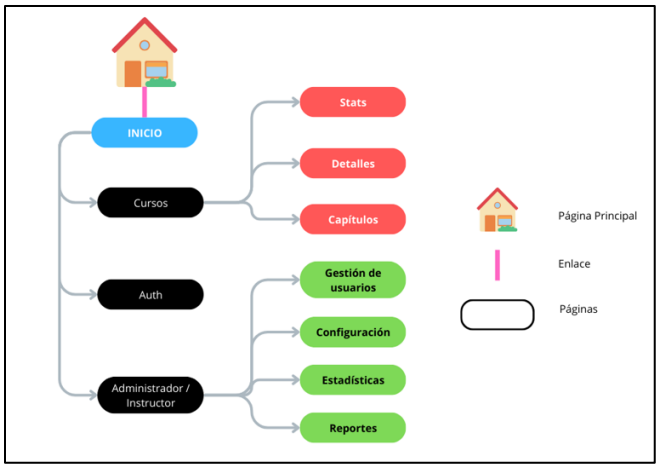
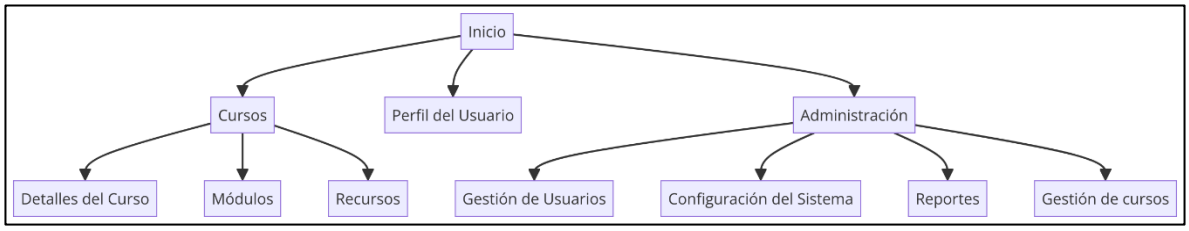


Figura 2.2.

Diseño navegacional interno de la plataforma



Es importante señalar que, para acceder a los cursos, a las herramientas del instructor o al seguimiento del progreso, primero se debe iniciar sesión. De lo contrario, el sistema redirige al usuario a la ventana de autenticación cuando intenta ingresar a una sección protegida sin las credenciales correspondientes.

El flujo de acceso inicia en la ventana principal, desde donde el usuario debe autenticarse. Tras un inicio de sesión exitoso, el sistema lo redirigirá al panel de inicio para permitir la navegación por las herramientas correspondientes a su rol.

La Figura 2.3 representa el concepto de la página de inicio de la plataforma, en la cual se contempla la visualización de cursos disponibles, avisos relevantes y accesos directos a las principales funciones del LMS. Esta vista también permite presentar información general de

cada curso, como el título, la cantidad de capítulos, la categoría correspondiente y el progreso del usuario, cuando este exista.

De acuerdo con el diseño conceptual, la disposición visual de estos elementos busca organizar la información de manera clara y jerárquica, facilitando que el usuario identifique rápidamente las secciones principales de la plataforma. De esta forma, la página de inicio funciona como un punto de acceso inicial hacia los cursos, el perfil del usuario y las herramientas disponibles según el rol correspondiente.

Figura 2.3.

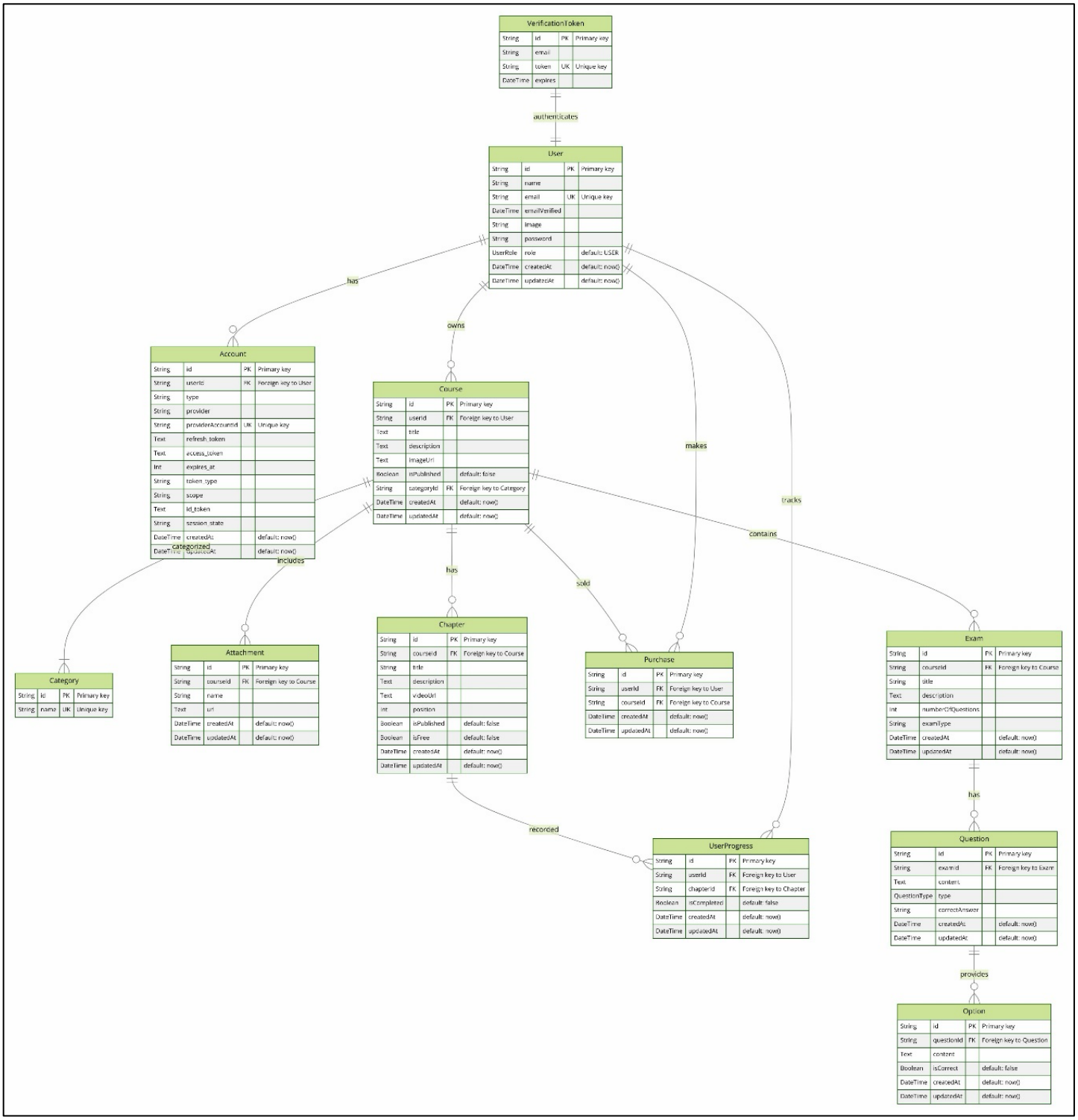
Concepto de la página de inicio.

Para que esta información pueda representarse de manera dinámica, resulta necesario contar con una base de datos que almacene el progreso de los estudiantes, así como la información de usuarios, cursos, capítulos, recursos y evaluaciones.

En la Figura 2.4 se presenta el modelo de la base de datos propuesto para la plataforma. Este modelo permite representar las entidades principales del sistema y las relaciones necesarias para gestionar usuarios, cursos, capítulos, recursos, evaluaciones e inscripciones. Aunque la estructura puede ajustarse en futuras versiones, su diseño inicial establece una base para organizar la información y permitir el funcionamiento de los principales módulos de Gasly LMS.

Figura 2.4.

Modelo de la base de datos



Capítulo 3. Diseño de implementación de WSDM

En este capítulo se describe la fase de diseño de implementación de WSDM, en la cual los elementos definidos durante el diseño conceptual se transforman en componentes funcionales de la plataforma. Para ello, se abordan aspectos relacionados con el modelado de la interfaz, la arquitectura de navegación, la implementación técnica, la codificación, las pruebas y la optimización del rendimiento. Esta fase permite vincular las decisiones de diseño con la construcción real del sistema, procurando que la plataforma mantenga coherencia con las necesidades de los usuarios y con los objetivos establecidos para Gasly LMS.

3.1 Modelado de la interfaz de usuario

Uno de los principales objetivos de esta fase es el modelado de la interfaz de usuario. Esto implica diseñar y desarrollar interfaces gráficas que sean intuitivas y fáciles de usar para estudiantes, instructores y administradores. Cada grupo de usuarios tendrá diferentes necesidades y expectativas, por lo que es esencial que las interfaces sean personalizadas para satisfacer estos requisitos específicos.

Dentro de las primeras tareas se encuentra el modelado de la interfaz de usuario para cada uno de los roles dentro de este sistema. Esto implica diseñar y desarrollar las interfaces gráficas adaptadas a las necesidades y funciones de los alumnos, docentes y administradores, garantizando así una experiencia personalizada y coherente dentro de la plataforma de aprendizaje.

3.2 Construcción de la arquitectura de navegación

Otro objetivo crucial es la construcción de la arquitectura de navegación del sitio. Esto incluye definir la estructura y el flujo de navegación que guiarán a los usuarios a través del contenido y las funcionalidades del LMS. Un buen diseño de navegación es esencial para asegurar que los usuarios puedan encontrar rápidamente la información que necesitan y moverse por la plataforma sin dificultades.

Además de eso, se realiza la construcción de la arquitectura de navegación del sitio, mediante la cual se define la estructura y el flujo de navegación que guiarán a los usuarios a través del contenido y las funcionalidades del LMS o SGA de manera efectiva, intuitiva y eficiente. Se

establecen mapas de navegación claros que han de facilitar la interacción de manera fluida entre las distintas secciones y herramientas del sistema.

3.3 Implementación técnica

La implementación técnica es otra parte fundamental de esta fase. Esta implica el desarrollo de funcionalidades clave como la gestión de cursos, administración de usuarios, seguimiento del progreso y herramientas de evaluación. Estas funcionalidades deben ser robustas y eficientes, proporcionando una experiencia de usuario de alta calidad.

En cuanto a la implementación técnica, se desarrollan e integran elementos de alta funcionalidad, como animaciones, gráficos y características interactivas, las cuales mejoran la experiencia del usuario y enriquecen la presentación visual del LMS.

3.4 Codificación y pruebas

Esta fase de desarrollo también implica la parte que respecta a la codificación de los programas técnicos necesarios para la funcionalidad del sistema gestor del aprendizaje. Es crucial utilizar tecnologías adecuadas para garantizar un rendimiento óptimo y una respuesta rápida del sistema.

A su vez, se implementan funcionalidades clave, como la gestión de los cursos, la administración de usuarios, seguimiento del proceso, instrumentos de evaluación, entre otras, asegurando así que la plataforma cumpla con los requisitos educativos y tecnológicos que se desea alcanzar.

3.5 Optimización del rendimiento

Finalmente, es esencial realizar pruebas exhaustivas para asegurar que el LMS funcione correctamente y cumpla con todos los requisitos técnicos y educativos. Esto incluye pruebas unitarias, de integración y de aceptación para identificar y corregir cualquier error o problema que se detecte durante las pruebas.

Optimizar el código y los recursos garantiza que el LMS funcione de manera eficiente, con tiempos de carga rápidos y una navegación fluida, asegurando así una experiencia de usuario de alta calidad.

Durante esta fase se llevan a cabo los siguientes pasos:

1. **Modelado de la interfaz de usuario:**

   - **Estudiantes**: La interfaz de los estudiantes debe ser clara y fácil de usar, proporcionando acceso rápido a los cursos, evaluaciones y recursos adicionales. Debe incluir elementos visuales atractivos y una estructura intuitiva que facilite la navegación y el acceso a la información.

   - **Instructores**: La interfaz de los instructores debe incluir herramientas para la creación y gestión de contenido, así como para la evaluación del progreso de los estudiantes. Debe ser flexible y permitir una fácil personalización de los cursos y las actividades.

   - **Administradores**: La interfaz de los administradores debe proporcionar acceso a herramientas avanzadas de gestión y configuración del sistema. Debe ser clara y eficiente, permitiendo a los administradores realizar tareas de mantenimiento y gestión sin complicaciones.

2. **Construcción de la arquitectura de navegación:**

   - **Definir la estructura de navegación**: Crear una estructura de navegación clara y lógica que permita a los usuarios acceder fácilmente a las diferentes secciones del LMS. Esto incluye la definición de menús, submenús y enlaces de navegación.

   - **Crear mapas de navegación**: Desarrollar mapas de navegación detallados que muestren cómo se conectan las diferentes secciones del LMS. Estos mapas deben ser claros y fáciles de entender, proporcionando una guía visual para la navegación del usuario.

   - **Implementar rutas de navegación**: Asegurarse de que las rutas de navegación sean intuitivas y eficientes, permitiendo a los usuarios moverse por la plataforma de manera rápida y sin problemas.

3. **Implementación técnica:**

   - **Desarrollo de funcionalidades clave**: Desarrollar e integrar funcionalidades clave como la gestión de cursos, administración de usuarios, seguimiento del progreso y herramientas de evaluación. Estas funcionalidades deben ser robustas y eficientes, proporcionando una experiencia de usuario de alta calidad.

   - **Integración de elementos interactivos**: Añadir elementos interactivos como animaciones, gráficos y características visuales que mejoren la experiencia del usuario.

Estos elementos deben ser atractivos y funcionales, ayudando a los usuarios a interactuar de manera más efectiva con el LMS.

- **Codificación y pruebas:** Utilizar tecnologías adecuadas para garantizar un rendimiento óptimo del sistema. Realizar pruebas exhaustivas para asegurar que el LMS funcione correctamente y cumpla con todos los requisitos técnicos y educativos.

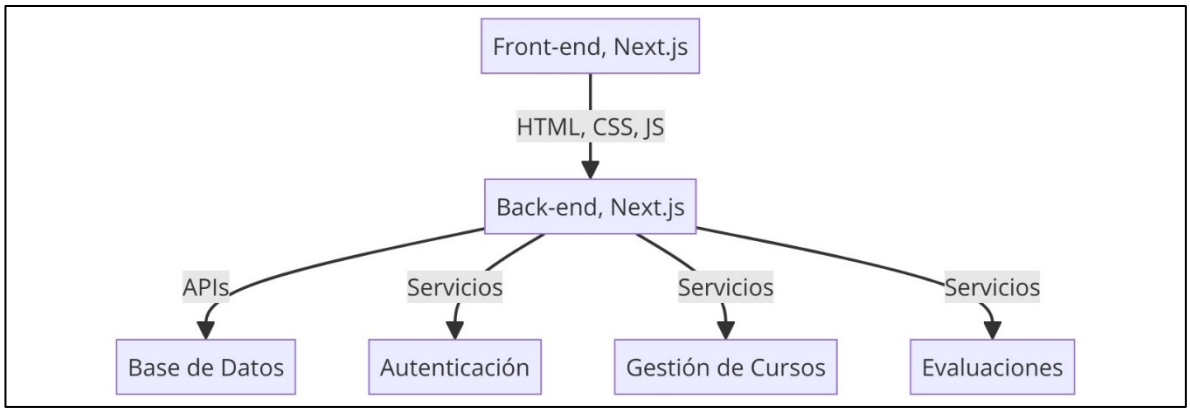
4. **Codificación y pruebas:**

- **Tecnologías adecuadas:** Utilizar tecnologías modernas y eficientes para el desarrollo del LMS. Esto incluye lenguajes de programación, frameworks y herramientas que aseguren un rendimiento óptimo y una experiencia de usuario de alta calidad.
- **Pruebas exhaustivas**: Realizar pruebas unitarias, de integración y de aceptación para asegurar que todas las funcionalidades del LMS funcionan correctamente. Identificar y corregir cualquier error o problema que se detecte durante las pruebas.
- **Optimización del rendimiento**: Asegurarse de que el LMS funcione de manera eficiente, con tiempos de carga rápidos y una navegación fluida. Optimizar el código y los recursos para garantizar una experiencia de usuario de alta calidad.

A continuación, la Figura 3.1 muestra el diagrama arquitectónico de la aplicación Gasly LMS, ejemplificando aspectos fundamentales para el desarrollo de la plataforma.

Figura 3.1.

Diagrama arquitectónico.



3.6 Implementación

La fase de implementación es la culminación del proceso de desarrollo, donde el diseño conceptual que se tenía en un inicio, se transforma en un sistema funcional. Esta fase incluye la codificación de todas las funcionalidades, la integración de componentes y la realización de pruebas exhaustivas para garantizar que el sistema web funcione según lo previsto.

Considerando el objetivo de desarrollar un LMS robusto, eficiente y preparado para su uso por parte de estudiantes, instructores y administradores es que se utilizó Next.js para el desarrollo de la aplicación, con Tailwind CSS para la estilización, y se trabajó en Visual Studio Code (VS Code) para la implementación.

El frontend fue creado utilizando Next.js y React, enfocándose en la creación de componentes reutilizables. Cada elemento de la interfaz, como las tarjetas de cursos (course cards) y las listas de tarjetas (card lists), se implementó como un componente individual en React.

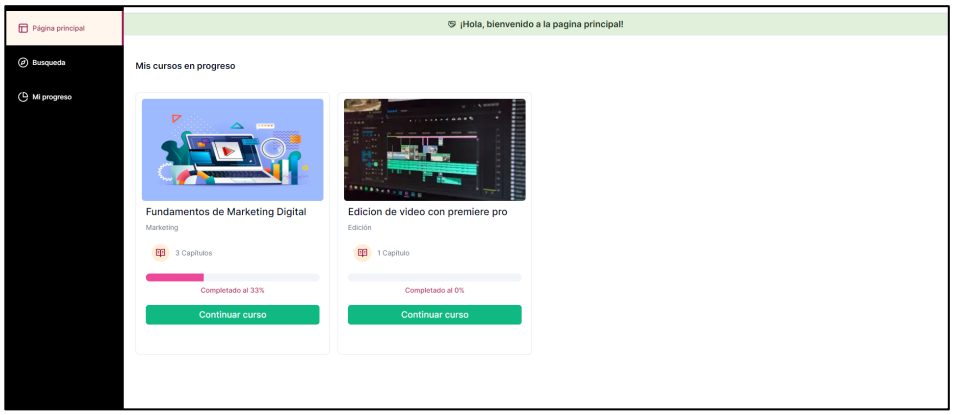
Tailwind CSS se utilizó para la estilización, permitiendo aplicar estilos de manera rápida y eficiente sin necesidad de escribir CSS personalizado. Las clases de Tailwind se utilizaron para asegurar una apariencia uniforme y profesional en toda la aplicación. Además, se aprovechó la capacidad de Tailwind para crear diseños responsivos, garantizando que la aplicación funcionara correctamente en una variedad de dispositivos y tamaños de pantalla.

Cada componente React incorporó HTML para estructurar el contenido, utilizando Tailwind CSS para la estilización. Este enfoque ayudó a mantener una clara separación entre la estructura y la presentación, mejorando la legibilidad y el mantenimiento del código. Se puso especial énfasis en la usabilidad y la responsividad para asegurar una experiencia de usuario óptima en diferentes dispositivos.

En la Figura 3.2 se muestra la página de inicio implementada en Gasly LMS, resultado de la transformación del diseño conceptual en una interfaz funcional. Esta vista presenta información relevante para el usuario cuando existen cursos en progreso; en caso contrario, se muestra un mensaje indicando que no hay cursos activos.

Figura 3.2.

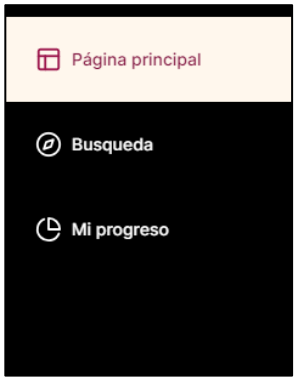
Página de inicio de Gasly LMS.



La tarjeta del curso integra la imagen, el título, la cantidad de capítulos disponibles, la categoría correspondiente y el progreso individual del usuario. Este componente se organiza dentro de una lista de tarjetas, lo que permite visualizar los cursos en las secciones de cursos completados y cursos en proceso. Esta interfaz cuenta con una barra lateral, como se muestra en la Figura 3.3. Esta barra cuenta con enlaces dinámicos según el *layout* utilizado a lo largo de las diferentes secciones entre ellas, cursos, capítulos y administración.

Figura 3.3.

Barra lateral en layout de estudiante

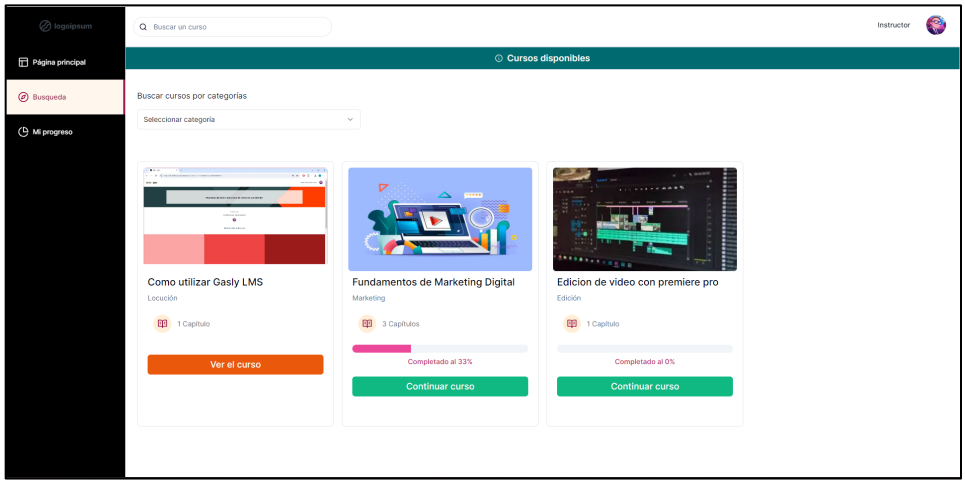


En la Figura 3.4 se muestra el apartado de búsqueda, donde se presentan los cursos disponibles, independientemente de si el usuario se encuentra inscrito o no. En este apartado,

los cursos pueden localizarse mediante una barra de búsqueda o a través de un selector de categorías, en el cual se puede realizar la búsqueda de acuerdo a las categorías existentes.

Figura 3.4.

Búsqueda de cursos



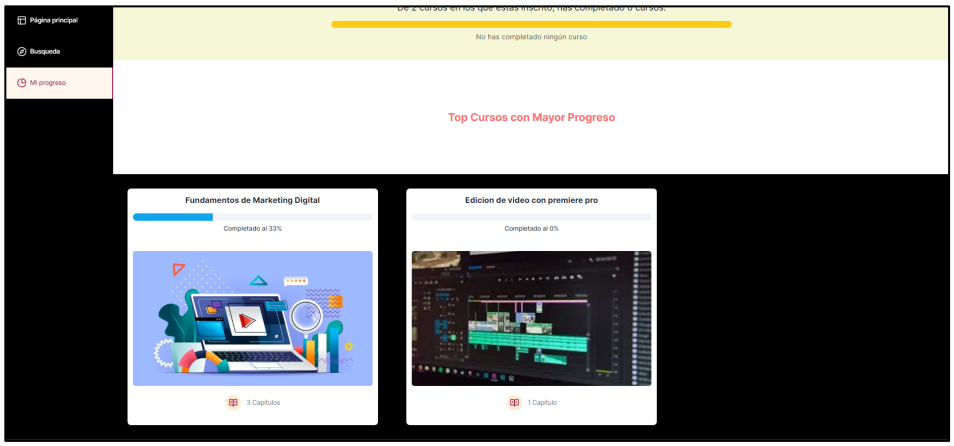
En las Figuras 3.5 y 3.6 se muestran los elementos implementados para representar el avance del usuario dentro de la plataforma. La primera presenta una gráfica circular y una barra de progreso general, mientras que la segunda permite identificar los cursos con mayor avance registrado.

Figura 3.5.

Gráfica circular y barra de progreso

Figura 3.6.

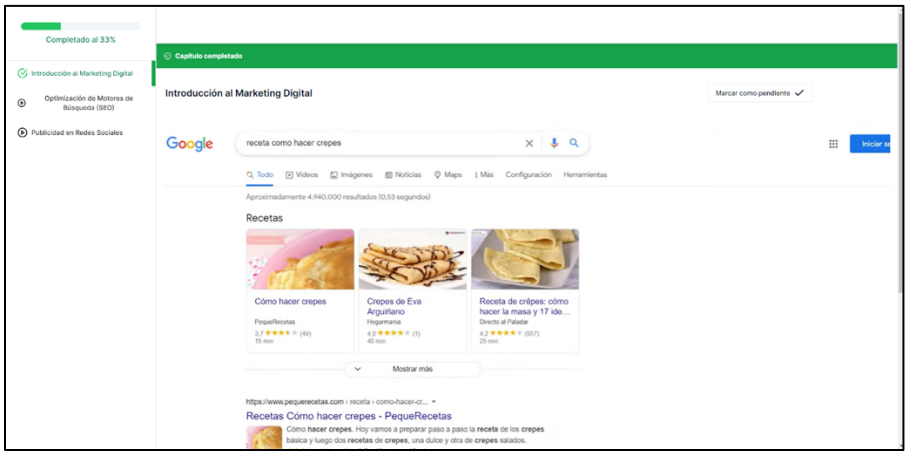
Cursos con mayor progreso



Al acceder a la ruta “chapters” desde las tarjetas disponibles en el inicio, la búsqueda o la sección de estadísticas, se presenta la vista de capítulos del curso. En esta sección se visualizan los videos, archivos adjuntos y descripciones correspondientes, además del estado de avance de cada capítulo, como se muestra en la Figura 3.7.

Figura 3.7.

Sección de capítulos del curso



Las Figuras 3.8 y 3.9 muestran la interfaz de la ruta “detalles”, donde se presentan los datos generales del curso y los capítulos disponibles. Cuando el usuario no se encuentra inscrito, algunos capítulos se muestran bloqueados y se habilita la opción para inscribirse al curso, como se observa en la Figura 3.10.

Figura 3.8.

Detalles del curso

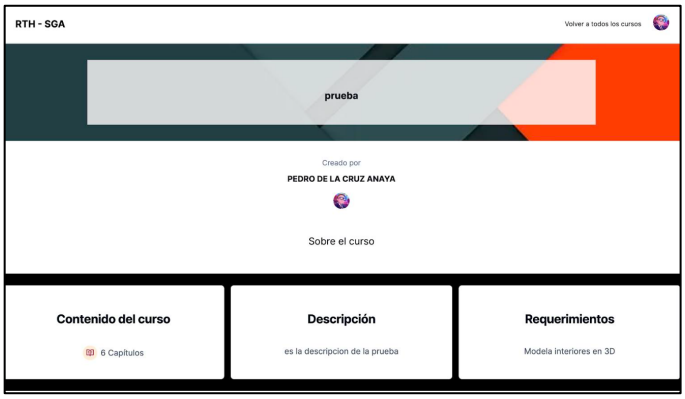
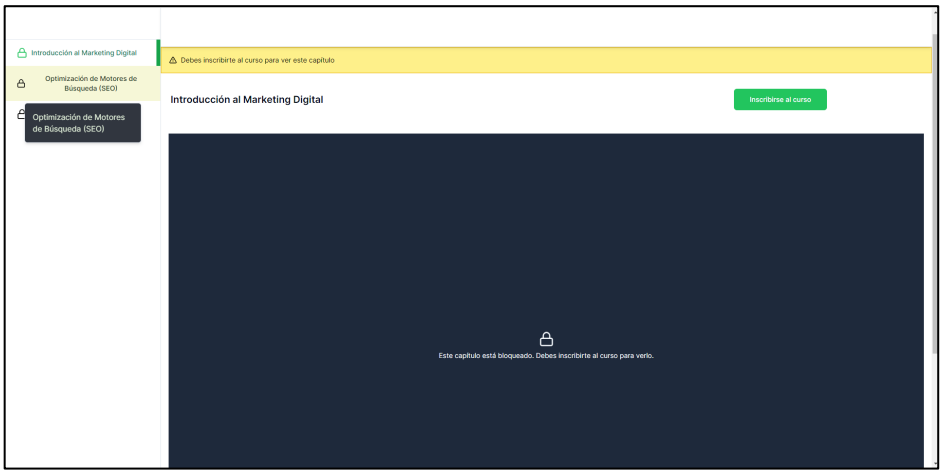


Figura 3.9.

Capítulos disponibles en detalles del curso

Figura 3.10.

Capítulos bloqueados en un curso sin inscripción



En el apartado de instructor, cuando la cuenta tiene los permisos necesarios, se habilita el acceso mediante el botón “Instructor”, visible en la página de inicio del estudiante. Este apartado incluye accesos en la barra lateral hacia evaluaciones, estadísticas, creación y edición de cursos, así como al módulo de chat, como se muestra en la Figura 3.11.

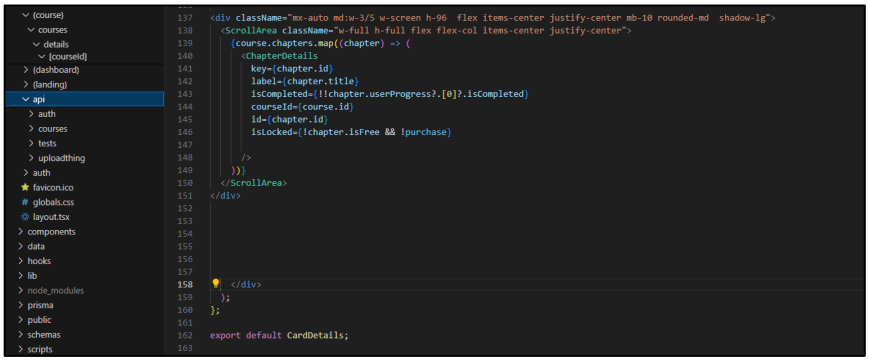
Figura 3.11.

Apartado del instructor

El backend del LMS se desarrolló también utilizando Next.js, organizando las funciones de servidor en la carpeta “api”. Estas funciones manejaron la lógica de la aplicación, la autenticación de usuarios, la gestión de sesiones y el acceso a la base de datos.

La Figura 3.12 muestra la carpeta “api” en la parte izquierda, mientras que a la derecha se observa el componente “CardDetails”, utilizado en la sección de detalles del curso.

Figura 3.12.
Carpeta API en el IDE y ejemplo de componente



Se implementaron mecanismos de autenticación y autorización para permitir que solo los usuarios autorizados accedieran a determinadas secciones del LMS. Asimismo, se gestionaron sesiones y solicitudes concurrentes, considerando que la plataforma puede ser utilizada por varios usuarios al mismo tiempo.

Las funciones del servidor definidas en la carpeta “api” se comunicaron con los componentes del frontend, proporcionando datos dinámicos y actualizaciones en tiempo real. Por ejemplo, cuando un usuario accede a la lista de cursos, se envía una solicitud desde el frontend al backend, donde una función de la API consulta la base de datos y devuelve la información necesaria. Esta información se muestra luego en las tarjetas de cursos, ofreciendo una experiencia de usuario fluida.

La Figura 3.13 muestra una parte del código utilizado para crear un curso, donde se emplea el método POST y se accede a valores como el “userId” para identificar al usuario que publica el curso.

Figura 3.13.
Ejemplo de creación de un curso

El funcionamiento de la aplicación permite observar la interacción entre los distintos componentes que integran un sistema web, como la comunicación entre rutas, el renderizado de interfaces, la conexión entre componentes y el intercambio de información entre *frontend* y *backend*. Además de esta interacción, la base de datos cumple un papel fundamental, ya que permite almacenar, organizar y consultar la información necesaria para el funcionamiento de la plataforma.

El esquema físico de la base de datos representado en la Figura 2.4 es fundamental para comprender la estructura y las relaciones entre los diferentes elementos que componen el sistema. Este esquema proporciona una descripción general de las tablas y sus relaciones, facilitando la gestión y el acceso a la información de manera eficiente.

Mediante Prisma como ORM, se definió un conjunto de modelos que representan entidades clave como usuarios, cursos, cuentas, capítulos, categorías, archivos adjuntos, inscripciones y exámenes. La configuración inicial del esquema incluye el generador de cliente, así como la fuente de datos para conectar con la base de datos MySQL.

A través de las relaciones definidas entre las tablas, se asegura que la integridad de los datos se mantenga y que las operaciones de CRUD (Crear, Leer, Actualizar, Eliminar) se realicen de manera coherente y segura.

Cada modelo en el esquema, desde User hasta Exam, fue diseñado para reflejar las necesidades del sistema y garantizar una organización clara y lógica de la información. Este enfoque permite una administración eficaz y un acceso optimizado a los datos, lo que es esencial para el rendimiento general y la escalabilidad del sistema.

La tabla User almacena información de los usuarios, incluyendo roles, credenciales y datos personales, y se relaciona con las tablas Course y Account. En esta relación, Account maneja cuentas de usuario relacionadas con distintos proveedores de autenticación, almacenando tokens y datos de sesión.

La tabla Course contiene información sobre los cursos, como título, descripción y estado de publicación, y se relaciona con User, Category, Chapter, Attachment, Purchase y Exam.

Category agrupa los cursos en diferentes categorías, mientras que Attachment almacena archivos adjuntos a los cursos, eliminándolos si se borra el curso correspondiente.

La tabla Chapter representa capítulos de los cursos, incluyendo contenido y estado de publicación, y se relaciona con UserProgress. UserProgress rastrea el progreso del usuario en los capítulos. Mientras que Purchase registra las inscripciones a los cursos de parte de los usuarios.

El modelo de exámenes se integró directamente con los cursos en la base de datos. Cada curso puede tener múltiples exámenes asociados, lo que permite evaluar el progreso de los estudiantes de manera eficiente. Las tablas Exam, Question y Option se diseñaron para almacenar la información de los exámenes, preguntas y opciones de respuesta, respectivamente.

El backend maneja la lógica para la creación, actualización y eliminación de exámenes, así como la asociación de estos exámenes con los cursos correspondientes. Las funciones de la API permiten al frontend interactuar con estos datos de manera fluida, asegurando que los estudiantes puedan acceder a los exámenes relacionados con sus cursos.

El desarrollo de Gasly LMS requirió un proceso estructurado para cumplir con los objetivos planteados al inicio del proyecto. Una vez integrados el frontend y el backend, el último paso consistió en el despliegue de la aplicación, realizado mediante Vercel, una plataforma orientada al alojamiento de aplicaciones web modernas.

Vercel se encargó de la construcción del proyecto, proporcionando URLs de preview y promoviendo los cambios a producción. Para optimizar el rendimiento, se utilizaron funcionalidades como la CDN integrada de Vercel, optimización automática de imágenes, y compresión de archivos. Además, Vercel habilitó HTTPS por defecto y ofreció opciones seguras para la gestión de secretos.

Como resultado de esta fase, Gasly LMS se consolidó como una plataforma funcional orientada a la gestión del aprendizaje en entornos digitales. La implementación permitió integrar una arquitectura web capaz de responder a necesidades relacionadas con la administración de cursos, la autenticación de usuarios, la gestión de contenidos, las evaluaciones y el seguimiento del progreso académico.

El uso de tecnologías como Next.js, Prisma, Tailwind CSS y Auth.js favoreció la construcción de una solución estructurada, segura y escalable, con una interfaz clara y una organización coherente entre sus componentes. Asimismo, la aplicación de la metodología WSDM permitió mantener un enfoque centrado en el usuario durante las distintas fases del desarrollo, contribuyendo a que la plataforma ofreciera una experiencia de uso comprensible y funcional.

En conjunto, la implementación técnica realizada permitió materializar los elementos conceptuales y metodológicos planteados en los capítulos anteriores, sentando las bases para la evaluación de la usabilidad del sistema, la cual se presenta en el capítulo siguiente.

Capítulo 4. Validación del sistema

La validación del sistema constituye una fase esencial en el desarrollo de Gasly LMS, ya que permite verificar si la plataforma satisface de manera adecuada los criterios de usabilidad desde la perspectiva de quienes la utilizan. Una vez definido el modelo de usuario, elaborado el diseño conceptual, implementada la arquitectura del sistema y ejecutadas las pruebas funcionales correspondientes, se vuelve indispensable evaluar la experiencia de interacción que ofrece la plataforma en un entorno real de uso.

Desde esta perspectiva, la validación no se limita a comprobar el correcto funcionamiento técnico del sistema, sino que también permite conocer cómo perciben los usuarios aspectos como la facilidad de uso, la claridad de la interfaz, la consistencia del sistema y el nivel de aprendizaje necesario para emplearlo. Estos elementos adquieren especial importancia en un sistema de gestión del aprendizaje, ya que su efectividad no depende únicamente de las funcionalidades que integra, sino también de la forma en que estas pueden ser entendidas y aprovechadas por estudiantes, docentes y administradores.

Para realizar esta evaluación se recurrió a la escala System Usability Scale (SUS), un instrumento ampliamente reconocido para medir la usabilidad percibida en sistemas interactivos. La aplicación de esta escala hizo posible recopilar información cuantitativa sobre la experiencia de uso de Gasly LMS, obteniendo tanto una valoración global del sistema como una apreciación específica de distintos aspectos relacionados con la interacción dentro de la plataforma.

En este capítulo se presenta, en primer lugar, la descripción del instrumento empleado; posteriormente, se desarrolla el análisis de las respuestas obtenidas en cada una de las preguntas del cuestionario; y, por último, se exponen los resultados generales derivados de la aplicación de la escala SUS, con el fin de interpretar el nivel de usabilidad alcanzado por la plataforma.

4.1 System Usability Scale (SUS)

Para la evaluación de la usabilidad de Gasly LMS se empleó la escala System Usability Scale (SUS), desarrollada por John Brooke en 1986. Este instrumento se ha consolidado como uno de los más utilizados en la valoración de sistemas interactivos, debido a su sencillez de

aplicación, facilidad de interpretación y efectividad para obtener una apreciación global de la experiencia de uso.

La escala SUS está compuesta por diez afirmaciones orientadas a conocer la percepción del usuario frente a un sistema. Estas afirmaciones permiten valorar aspectos como la frecuencia con la que se utilizaría la plataforma, el nivel de complejidad percibido, la facilidad de uso, la necesidad de asistencia técnica, la integración de sus funciones, la consistencia del sistema, la rapidez de aprendizaje y la confianza que genera el uso de la herramienta.

Cada una de las afirmaciones se responde mediante una escala de cinco niveles, en la que:

1 = Totalmente en desacuerdo
2 = En desacuerdo
3 = Ni de acuerdo ni en desacuerdo
4 = De acuerdo
5 = Totalmente de acuerdo

Una de las particularidades más relevantes de la escala SUS es que combina enunciados formulados tanto en sentido positivo como negativo. Esta característica contribuye a disminuir posibles sesgos en las respuestas y favorece una valoración más equilibrada de la experiencia del usuario.

En el marco de esta investigación, la encuesta fue aplicada a 34 participantes, quienes hicieron uso previo de la plataforma y posteriormente respondieron el cuestionario de usabilidad. Para calcular la puntuación SUS, las respuestas fueron procesadas de acuerdo con el método tradicional: en los ítems positivos se resta 1 al valor seleccionado por el participante, mientras que en los ítems negativos se resta la respuesta dada a 5. Luego, la suma de los valores ajustados se multiplica por 2.5, obteniéndose un puntaje global en una escala de 0 a 100.

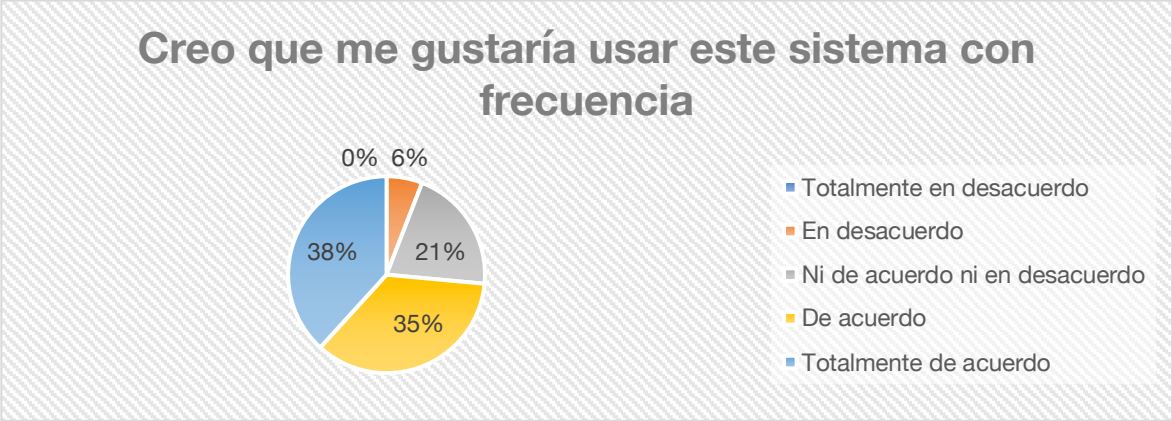
La utilización de este instrumento permitió obtener una medida cuantitativa de la usabilidad percibida de Gasly LMS, así como reconocer sus principales fortalezas y detectar posibles oportunidades de mejora dentro de la plataforma.

4.2 Análisis de resultados

Con el propósito de realizar un análisis más detallado de la percepción de los usuarios, se revisaron individualmente los resultados correspondientes a cada una de las diez afirmaciones de la escala SUS. Este análisis permite observar qué aspectos del sistema fueron mejor valorados y cuáles presentan oportunidades de mejora.

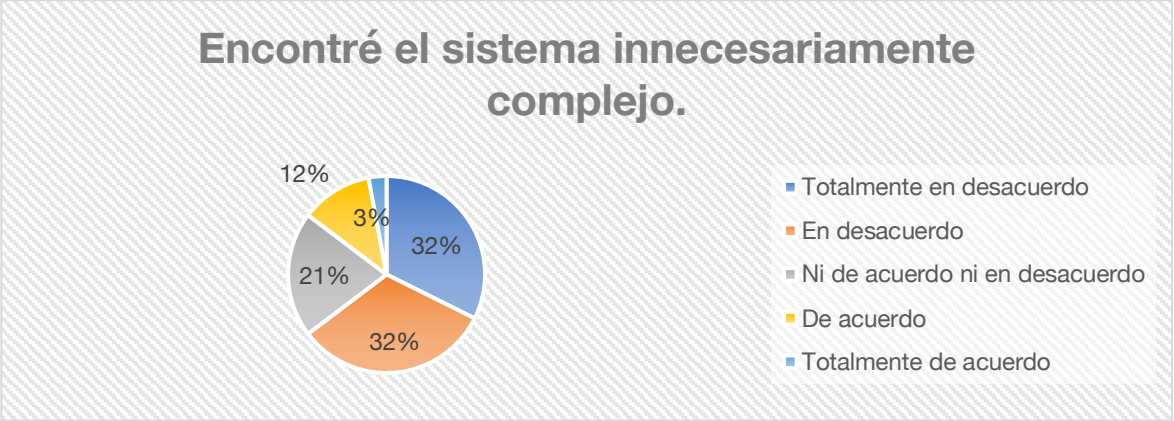
En la pregunta 1, relacionada con la posibilidad de utilizar el sistema con frecuencia, las respuestas mostraron una tendencia positiva. La mayoría de los participantes manifestó estar de acuerdo o totalmente de acuerdo, lo cual indica una buena aceptación general de la plataforma y una disposición favorable hacia su uso continuo**,** como puede observarse en la Gráfica 4.1.

Gráfica 4.1.
Primera pregunta de la evaluación SUS



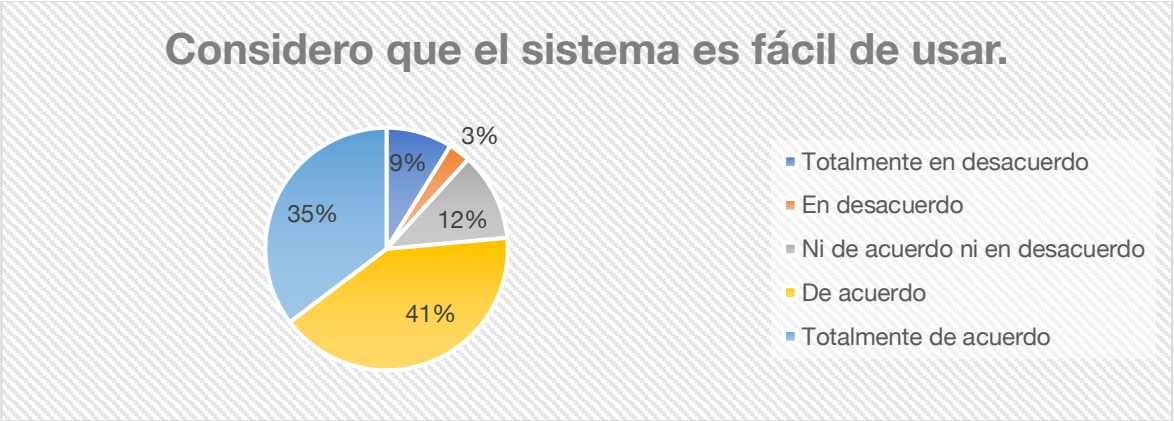
En la pregunta 2, orientada a identificar si el sistema fue percibido como innecesariamente complejo, predominó el desacuerdo. Este comportamiento resulta positivo, ya que sugiere que los usuarios consideran que la interacción con la plataforma es relativamente clara y no presenta niveles altos de complejidad, tal como se muestra en la Gráfica 4.2**.**

Gráfica 4.2.
Segunda pregunta de la evaluación SUS



En la pregunta 3, referente a la facilidad de uso del sistema, los resultados también fueron favorables. La mayoría de los usuarios coincidió en que Gasly LMS es fácil de usar, lo que refleja que la estructura de navegación y la organización general de la interfaz permiten una interacción comprensible, como se aprecia en la Gráfica 4.3**.**

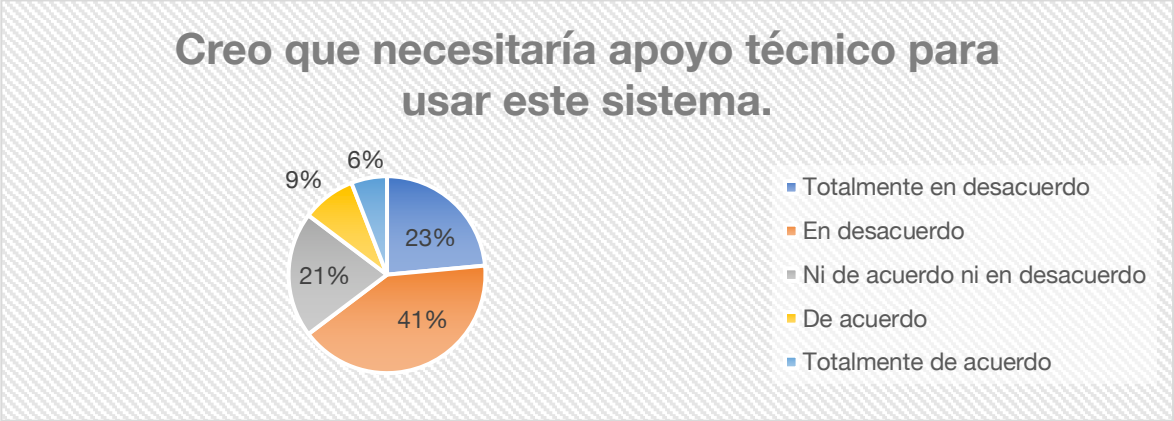
Gráfica 4.3.
Tercera pregunta de la evaluación SUS



En la pregunta 4, que explora si el usuario considera necesario contar con apoyo técnico para utilizar la plataforma, la mayor parte de las respuestas se inclinó hacia el desacuerdo. Esto indica que el sistema puede ser utilizado de manera autónoma por la mayoría de los

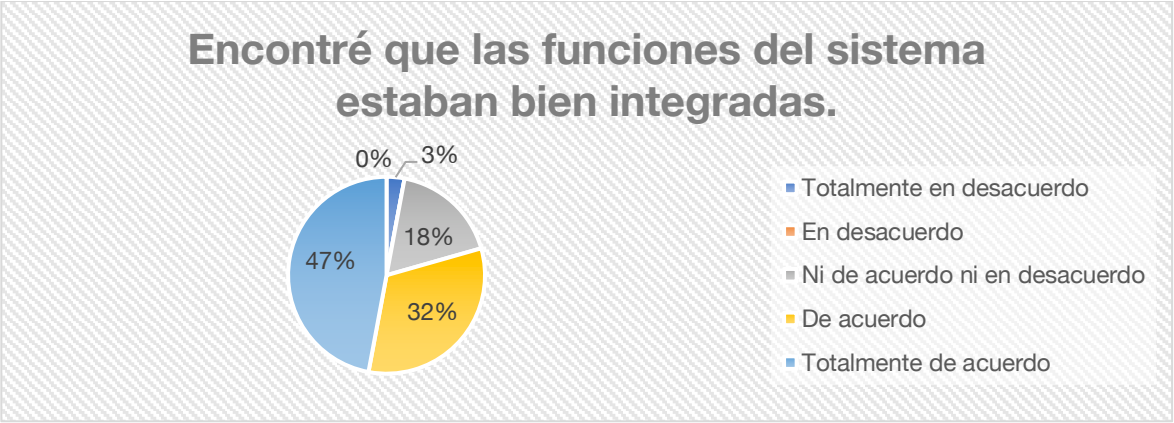
participantes, sin una dependencia excesiva de ayuda externa, lo cual puede observarse en la Gráfica 4.4.

Gráfica 4.4.
Cuarta pregunta de la evaluación SUS



En la pregunta 5, relacionada con la integración de las funciones del sistema, las respuestas fueron predominantemente favorables. Los participantes consideraron que los componentes y herramientas de la plataforma están articulados de forma adecuada, lo que contribuye a una experiencia de uso más fluida, como puede verse en la Gráfica 4.5.

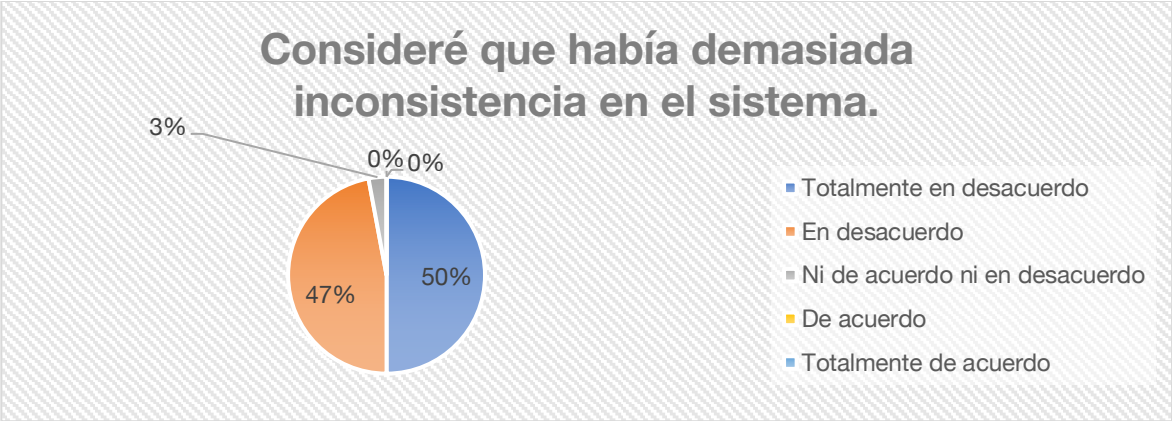
Gráfica 4.5.
Quinta pregunta de la evaluación SUS



En la pregunta 6, referente a la posible inconsistencia del sistema, se observó uno de los resultados más positivos de toda la escala. La mayoría de los usuarios manifestó estar en

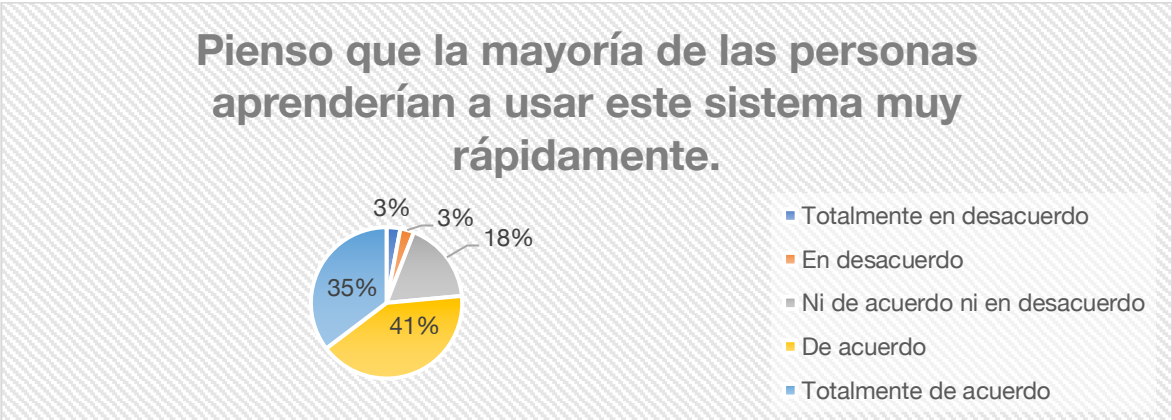
desacuerdo con la afirmación, lo que sugiere que Gasly LMS mantiene una estructura coherente y uniforme en su funcionamiento, como se describe en la Gráfica 4.6.

Gráfica 4.6.
Sexta pregunta de la evaluación SUS



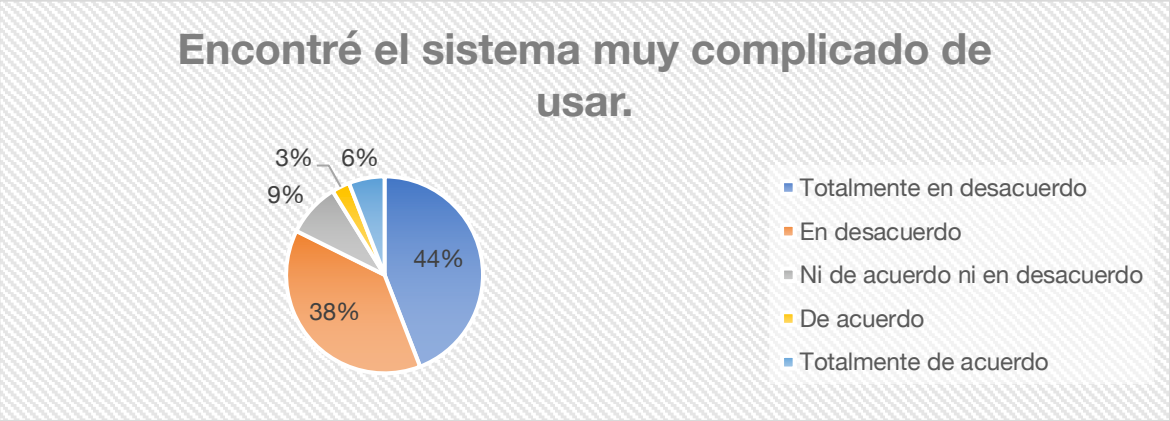
En la pregunta 7, orientada a determinar si otras personas podrían aprender a usar el sistema rápidamente, las respuestas mostraron nuevamente una tendencia favorable. Esto permite inferir que la plataforma presenta una curva de aprendizaje accesible y que sus funciones pueden ser comprendidas con relativa facilidad, tal como se representa en la Gráfica 4.7.

Gráfica 4.7.
Séptima pregunta de la evaluación SUS



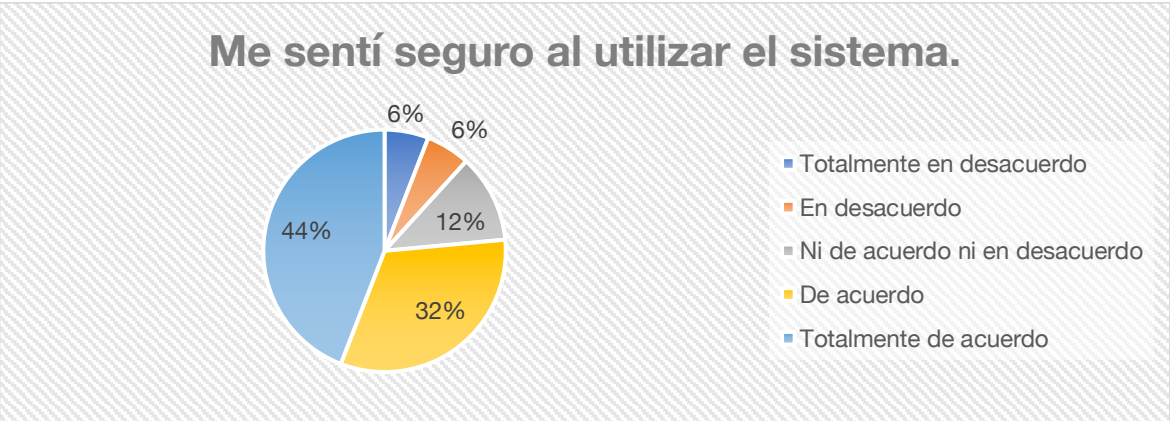
En la pregunta 8, relacionada con la dificultad general de uso, predominó el desacuerdo, reforzando la percepción de que el sistema no resulta complicado para los usuarios y que la experiencia de interacción es clara, como se muestra en la Gráfica 4.8.

Gráfica 4.8.
Octava pregunta de la evaluación SUS



En la pregunta 9, que evalúa la seguridad o confianza percibida al utilizar el sistema, la mayoría de los participantes respondió de forma positiva. Este resultado es importante, ya que la confianza en el uso de una plataforma educativa influye directamente en la disposición del usuario para interactuar con ella, como puede observarse en la Gráfica 4.9.

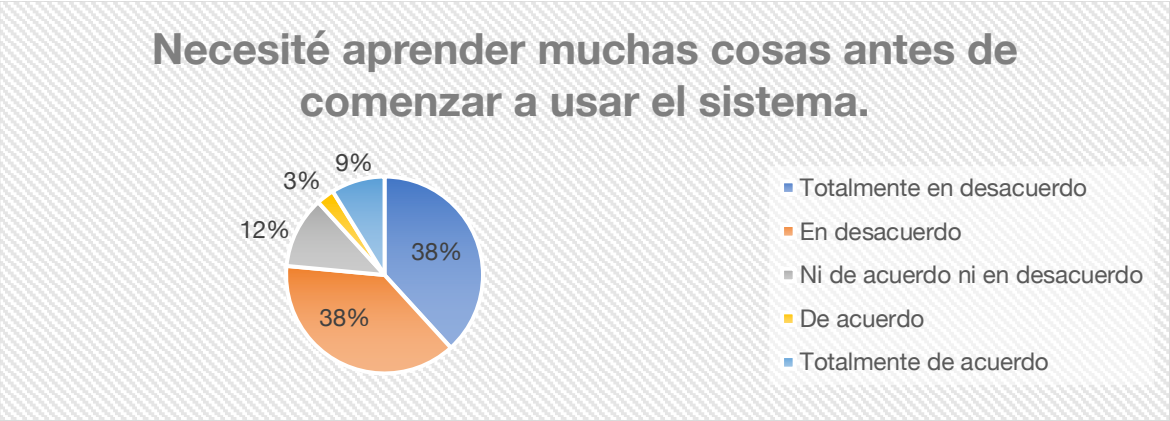
Gráfica 4.9.
Novena pregunta de la evaluación SUS



Por último, en la pregunta 10, referente a la necesidad de aprender muchas cosas antes de comenzar a usar el sistema, se observó nuevamente una tendencia al desacuerdo. Esto

confirma que Gasly LMS puede utilizarse sin requerir un proceso de aprendizaje extenso antes del primer uso, tal como se muestra en la Gráfica 4.10.

Gráfica 4.10.
Décima pregunta de la evaluación SUS



En conjunto, el análisis individual de las preguntas muestra una valoración favorable de la plataforma, particularmente en aspectos como la consistencia, la facilidad de uso, la baja complejidad percibida y la facilidad de aprendizaje.

4.3 Resultados generales

Para obtener el resultado general de usabilidad de Gasly LMS se aplicó el procedimiento de cálculo establecido por la escala System Usability Scale (SUS). Este instrumento se compone de diez afirmaciones con respuestas en escala Likert de cinco niveles, donde 1 corresponde a “Totalmente en desacuerdo”, 2 a “En desacuerdo”, 3 a “Ni de acuerdo ni en desacuerdo”, 4 a “De acuerdo” y 5 a “Totalmente de acuerdo”. Sin embargo, estos valores no se suman directamente, debido a que la escala combina preguntas redactadas en sentido positivo y negativo.

En primer lugar, se identificaron los ítems positivos y negativos del cuestionario. Las preguntas positivas corresponden a los ítems 1, 3, 5, 7 y 9, ya que una puntuación alta en ellas representa una percepción favorable del sistema. En estos casos, el ajuste se realizó restando 1 al valor seleccionado por cada participante. Por otra parte, las preguntas negativas corresponden a los ítems 2, 4, 6, 8 y 10, debido a que están formuladas en sentido contrario;

por ello, el ajuste se realizó restando el valor seleccionado a 5. De esta manera, todas las respuestas quedaron orientadas en el mismo sentido, donde un valor ajustado más alto representa una mejor percepción de usabilidad.

Una vez ajustadas las respuestas, se sumaron los valores obtenidos en los diez ítems para cada participante. Posteriormente, la suma individual se multiplicó por 2.5, con el propósito de convertir el resultado a una escala de 0 a 100. A partir de las 34 respuestas consideradas en la evaluación, la suma total de los valores ajustados fue de 1028 puntos. Al dividir esta cantidad entre los 34 participantes, se obtuvo un promedio ajustado de 30.23 puntos sobre un máximo posible de 40. Finalmente, al multiplicar dicho promedio por 2.5, se obtuvo un puntaje SUS promedio de 75.59 sobre 100.

El procedimiento anterior puede expresarse de la siguiente manera:

$$\text{Promedio ajustado} = 1028 \div 34 = 30.23$$

$$\text{Puntaje } \textit{SUS} \text{ promedio} = 30.23 \times 2.5 = 75.59$$

Este resultado permite considerar que Gasly LMS presenta una usabilidad favorable, ya que refleja una percepción positiva por parte de los participantes respecto a la interacción con la plataforma. De manera general, los resultados muestran que los usuarios consideran que el sistema es comprensible, relativamente fácil de usar y con una estructura coherente. Los reactivos mejor valorados fueron aquellos relacionados con la consistencia del sistema, la baja complejidad percibida y la facilidad para aprender a utilizarlo.

Asimismo, las respuestas obtenidas indican una percepción positiva respecto a la integración de funciones y a la seguridad o confianza al interactuar con el sistema. Esto sugiere que Gasly LMS ofrece una experiencia estructurada, donde sus diferentes componentes mantienen relación entre sí y contribuyen al cumplimiento de los objetivos de aprendizaje y administración del contenido.

Aunque la evaluación general fue favorable, algunos reactivos presentaron una mayor variación en las respuestas, particularmente aquellos vinculados con la frecuencia de uso y con la necesidad de apoyo técnico. Si bien en ambos casos la tendencia general fue positiva, estos resultados también permiten identificar áreas susceptibles de mejora, por ejemplo,

mediante una mayor optimización de ciertos flujos de interacción o mediante recursos de apoyo inicial para nuevos usuarios.

En términos generales, los resultados derivados de la escala SUS permiten afirmar que Gasly LMS cumple adecuadamente con criterios de usabilidad, ofreciendo una experiencia de interacción positiva, coherente y funcional para los usuarios que participaron en la evaluación. Esto respalda la validez del sistema como una plataforma utilizable dentro del contexto educativo y fortalece la propuesta desarrollada a lo largo de esta investigación.

Conclusiones

El desarrollo de Gasly LMS permite afirmar que se cumplió con el objetivo general de este proyecto, desarrollando un sistema de gestión de aprendizaje accesible a través de tecnologías web, con una orientación hacia la administración de cursos y la distribución de contenidos educativos en formato digital. La propuesta responde a la necesidad de tener una plataforma que proporcione la posibilidad de aprender y capacitarse de manera flexible en un entorno virtual.

La metodología WSDM aplicada en la implementación del sistema resultó adecuada para el desarrollo de este último, ya que permitió abordar las fases de diseño y desarrollo de manera centrada en el usuario. El modelo de usuario, la definición conceptual y la posterior implementación permitieron construir una plataforma coherente y relevante para el perfil de estudiantes, profesores y administradores del mismo.

Desde el punto de vista técnico, la utilización de herramientas como Next.js, Prisma, Tailwind CSS y Auth.js permitió desarrollar una plataforma funcional, con interfaz clara, métodos seguros de autenticación y una estructura que facilita la escalabilidad del sistema. La incorporación de funciones como administración de cursos, evaluaciones y seguimiento del progreso del estudiante en los mismos aportó características relevantes a la plataforma, consolidándola como sistema de gestión de aprendizaje.

Asimismo, la aplicación de la prueba de usabilidad mediante la escala SUS resultó útil para obtener información cuantitativa acerca de la usabilidad del sistema y así evaluar su funcionalidad desde la perspectiva del usuario. Esta evaluación, además de servir para obtener una calificación global de usabilidad de la plataforma, permite identificar áreas de mejora para futuras versiones del sistema.

Finalmente, se concluye que Gasly LMS constituye una propuesta viable como sistema de gestión del aprendizaje, al integrar características pedagógicas, técnicas y funcionales en una única plataforma. De manera complementaria, el desarrollo de este proyecto permitió reconocer que la construcción de un sistema de gestión del aprendizaje no depende únicamente de la implementación técnica de funcionalidades, sino también de la integración equilibrada entre criterios pedagógicos, de usabilidad, organización de la información y toma

de decisiones metodológicas. A lo largo del proceso se puso de manifiesto la importancia de analizar las necesidades de los usuarios, estructurar adecuadamente la navegación, mantener coherencia entre diseño e implementación y evaluar el sistema desde la experiencia de uso. En este sentido, el proyecto no solo dio como resultado una plataforma funcional, sino que también evidenció el valor que tiene un enfoque integral en el desarrollo de software orientado al ámbito educativo, donde la calidad del sistema está estrechamente vinculada con su utilidad, accesibilidad y pertinencia para los usuarios finales.

Trabajos futuros

Como parte del crecimiento de Gasly LMS, se contemplan distintas líneas de mejora orientadas a fortalecer la interacción, la organización académica y el seguimiento del aprendizaje. Entre ellas destaca la incorporación de foros de discusión, con el fin de favorecer el intercambio de ideas y la colaboración entre estudiantes e instructores dentro de cada curso. Asimismo, se plantea la implementación de una agenda o calendario académico que permita visualizar fechas de entrega, evaluaciones, recordatorios y eventos relevantes.

De igual manera, podrían ampliarse las herramientas de comunicación mediante un sistema de mensajería más completo, así como incorporarse notificaciones inteligentes, analítica académica, recomendaciones personalizadas de aprendizaje y mecanismos de motivación y reconocimiento del progreso. Finalmente, también se contempla la posibilidad de integrar videoconferencias en tiempo real, rúbricas de evaluación más completas y una experiencia optimizada para dispositivos móviles. En conjunto, estas mejoras permitirían consolidar a Gasly LMS como una plataforma más completa, flexible e innovadora dentro del ámbito de los sistemas de gestión del aprendizaje.

Referencias

Academic Technology Solutions. (2023). Canvas [Imagen]. University of Chicago. https://academictech.uchicago.edu/files/2023/10/DashboardBlur-980x460.png

Aliaga Meléndez, C. L., & Dávila Rojas, O. M. (2021). La plataforma Blackboard: una herramienta para el proceso de enseñanza-aprendizaje. Hamut'ay, 8(1), 42–58. https://doi.org/10.21503/hamu.v8i1.2237

Ardila Muñoz, J. Y., Ruiz Cañadulce, E. M., & Castro Molano, I. L. (2015). Estudio comparativo de sistemas de gestión del aprendizaje: Moodle, ATutor, Claroline, Chamilo y Universidad de Boyacá. Academia y Virtualidad, 8(1), 54–65.

ATutor. (s. f.). ATutor home. Recuperado el 18 de octubre de 2024, de https://ATutor.github.io/

Auth.js. (s. f.). Getting started. Recuperado el 20 de noviembre de 2025, de https://authjs.dev/getting-started

Bailón, J. E. D., & Vélez, J. M. (2021). La plataforma Moodle: caracterización, aplicaciones y beneficios para las competencias docentes. Revista Cognosis, 6(4). https://doi.org/10.33936/cognosis.v6i4.3046

Balasubramanian, V., Bieber, M., & Isakowitz, T. (2001). A case study in systematic hypermedia design. Information Systems, 26(4), 295–320.

Barrera, A., Peña Sklyar, I., & Peña Matos, M. (2016). Diseño e implementación de un entorno virtual de aprendizaje (EVA) utilizando la plataforma educativa Moodle. Estudio de caso: asignatura Ergonomía. Revista Universidad y Sociedad, 8(2), 33–40.

Bendezú Paytán, M. (2018). LMS concepto de sistemas de gestión de aprendizaje (LMS), tipos y clasificación, importancia, beneficios que brindan los LMS, plataformas virtuales: Moodle, Chamilo, Claroline, Blackboard, Dokeos, Docebo, Edu 2.0, aplicaciones [Monografía de segunda especialidad, Universidad Nacional de Educación Enrique Guzmán y Valle]. Repositorio Institucional UNE.

Bierman, G., Abadi, M., & Torgersen, M. (2014). Understanding TypeScript. En R. Jones (Ed.), ECOOP 2014 – Object-Oriented Programming (pp. 257–281). Springer. https://doi.org/10.1007/978-3-662-44202-9_11

Black, P., & Wiliam, D. (1998). Assessment and classroom learning. Assessment in Education: Principles, Policy & Practice, 5(1), 7–74. https://doi.org/10.1080/0969595980050102

Blackboard. (s. f.). What is a learning management system (LMS)? Blackboard. Recuperado el 22 de octubre de 2024, de https://www.blackboard.com/learning-management-system/what-is-an-lms

Blogger. (s. f.). ATutor [Imagen]. Recuperado el 3 de mayo de 2026, de https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEikRcbb6fiQCuHKrOmundSwEK7ay47fEFaQQAJTEgYAhGswlLNXue5Yh07kLdt3hzQSPwuSltTFbR3VV_KpdImD9nwkNFFq2O1KuLzGmhCp0NfwW8DDCmESNSEqiApchJwkszu1hyte02/s1600/ATT22.jpg

Brooke, J. (1996). SUS: A quick and dirty usability scale. En P. W. Jordan, B. Thomas, I. L. McClelland, & B. Weerdmeester (Eds.), Usability evaluation in industry (pp. 189–194). Taylor & Francis.

Brookhart, S. M. (2017). How to give effective feedback to your students (2.ª ed.). ASCD.

Cabero-Almenara, J., Llorente-Cejudo, M. C., & Vázquez-Martínez, A. I. (2014). Las tipologías de MOOC: su diseño e implicaciones educativas. Profesorado. Revista de Currículum y Formación del Profesorado, 18(1), 13–26.

Castillo-Montes, M., Vargas-Enríquez, J., & García-Mundo, L. (2018). Modelado conceptual de una aplicación web usando la metodología OOWS: un caso práctico. TecnoINTELECTO, 15(2), 19–28.

Chávez Martínez, J. de J. (2016). Plataforma Edu 2.0 en educación superior. Revista Electrónica sobre Cuerpos Académicos y Grupos de Investigación, 3(6).

Cloudflare. (s. f.). What is Transport Layer Security (TLS)? Recuperado el 5 de abril de 2026, de https://www.cloudflare.com/learning/ssl/transport-layer-security-tls/

ComparaSoftware. (s. f.). ATutor [Captura de pantalla]. Recuperado el 3 de mayo de 2026, de https://www.comparasoftware.com/image-assets/412/NDEyfHdwLWNvbnRlbnQvdXBsb2Fkcy8yMDE4LzA4L2NhcHR1cmVhdHV0b3IyLnBuZw.webp

Computer History Museum. (s. f.). The Babbage engine. Recuperado el 5 de abril de 2026, de https://www.computerhistory.org/babbage/engines/

Cosano Rivas, F. (s. f.). La plataforma de aprendizaje Moodle como instrumento para el Trabajo Social en el contexto del Espacio Europeo de Educación Superior. Universidad de Málaga.

Cuevas Valencia, R. E., & Zenón Rodríguez, E. (2013). Edu 2.0 plataforma en línea aplicada a la educación superior [Tesis de pregrado, Universidad Autónoma de Guerrero].

Davies, D. L., Lawal, A., Orji, A. E., Tytherleigh, C., & Walsh, K. (2024). Digital learning, face-to-face learning and climate change. Future Healthcare Journal, 11(3), 100156. https://doi.org/10.1016/j.fhj.2024.100156

De Pablos, J., Bravo, M. P., López-Gracia, A., & Lázaro, I. (2019). Los usos de las plataformas digitales en la enseñanza universitaria: perspectivas desde la investigación educativa. REDU: Revista de Docencia Universitaria, 17(1), 15.

De Troyer, O. M. F., & Leune, C. J. (1998). WSDM: A user-centered design method for web sites. Computer Networks and ISDN Systems, 30(1–7), 85–94.

Díaz, A., & Isakowitz, T. (1995). RMCase: A tool to design WWW applications. Computer Networks and ISDN Systems, 28(1–2), 249–259.

Díaz Becerro, S. (2009). Plataformas educativas, un entorno para profesores y alumnos. Temas para la Educación: Revista Digital para Profesionales de la Enseñanza, (2).

Elmasri, R., & Navathe, S. B. (2016). Fundamentals of database systems (7.ª ed.). Pearson.

Escalona Cuaresma, M. J. (2001). Metodologías para el desarrollo de sistemas de información global: análisis comparativo y propuesta. Universidad de Sevilla.

Fathema, N., & Akanda, M. H. (2020). Effects of instructors' academic disciplines and prior experience with learning management systems: A study about the use of Canvas. Australasian Journal of Educational Technology, 36(4), 113–125. https://doi.org/10.14742/ajet.5660

Ferreiro Martínez, V. V., Garambullo, A. I., & Brito Laredo, J. (2013). Prácticas innovadoras: uso de la plataforma Blackboard en modalidades semipresenciales. Caso práctico UABC FIN Tecate. RIDE. Revista Iberoamericana para la Investigación y el Desarrollo Educativo, 4(7), 129–150. https://www.redalyc.org/pdf/4981/498150315007.pdf

Fielding, R., Nottingham, M., & Reschke, J. (2022). HTTP semantics (RFC 9110). Internet Engineering Task Force. https://doi.org/10.17487/RFC9110

Flanagan, D. (2020). JavaScript: The definitive guide (7.ª ed.). O'Reilly Media.

Fowler, M. (2002). Patterns of enterprise application architecture. Addison-Wesley.

Fox, A. (2013). From MOOCs to SPOCs. Communications of the ACM, 56(12), 38–40. https://doi.org/10.1145/2535918

García-Peñalvo, F. J., Fidalgo-Blanco, Á., & Sein-Echaluce, M. L. (2017). Los MOOC: un análisis desde una perspectiva de la innovación institucional universitaria. La Cuestión Universitaria, (9), 117–135.

González, M. A., Perdomo, K. V., & Smith, O. Y. (2017). Aplicación de las TIC en modelos educativos blended learning: una revisión sistemática de literatura. Sophia, 13(1), 144–154.

IBM. (s. f.). What is backend development? Recuperado el 5 de abril de 2026, de https://www.ibm.com/topics/backend-development

IBM. (s. f.). Web server vs. application server: What's the difference? Recuperado el 5 de abril de 2026, de https://www.ibm.com/think/topics/web-server-application-server

Instructure. (s. f.). Canvas by Instructure. Recuperado el 15 de octubre de 2024, de https://www.instructure.com/canvas

Instructure. (s. f.). Canvas LMS logo [Imagen]. Recuperado el 3 de mayo de 2026, de https://www.instructure.com/about/brand-guide/canvas

Instructure. (s. f.). Learning management system (LMS). Recuperado el 15 de octubre de 2024, de https://www.instructure.com/lms-learning-management-system

Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado. (s. f.). ¿Qué es un NOOC? Recuperado el 5 de abril de 2026, de https://educalab.es/intef/formacion/formacion-en-red/nooc

International Organization for Standardization. (2018). ISO 9241-11:2018 Ergonomics of human-system interaction—Part 11: Usability: Definitions and concepts. ISO.

Isakowitz, T., Stohr, E. A., & Balasubramanian, P. (1995). RMM: A methodology for structured hypermedia design. Communications of the ACM, 38(8), 34–44.

Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT) (RFC 7519). Internet Engineering Task Force. https://doi.org/10.17487/RFC7519

Kaplan, A. M., & Haenlein, M. (2016). Higher education and the digital revolution: About MOOCs, SPOCs, social media, and the Cookie Monster. Business Horizons, 59(4), 441–450. https://doi.org/10.1016/j.bushor.2016.03.008

Karat, J. (1997). Evolving the scope of user-centered design. Communications of the ACM, 40(7), 33–38.

Kurose, J. F., & Ross, K. W. (2021). Computer networking: A top-down approach (8.ª ed.). Pearson.

Lazar, J. (2006). Web usability: A user-centered design approach. Pearson Addison Wesley.

Lebrun, M., Docq, F., & Smidts, D. (2003). Claroline, an Internet teaching and learning platform to foster teachers' professional development and improve teaching quality: First approaches. International Journal of Continuing Engineering Education and Life-Long Learning, 13(3–4), 347–363.

Llorente Cejudo, M. C. (2007). Moodle como entorno virtual de formación al alcance de todos. Comunicar, 15(28), 197–202.

Lobo, J., Teodosio, K., Estilo, K., Achas, M., Aliser, J., & Codod, C. L. (2025). Canvas Learning Management System and Zoom: Serviceability directed toward physical activities and health and fitness courses for synchronous and asynchronous classes. Journal of Educators Online, 22(2). https://doi.org/10.9743/JEO.2025.22.2.2

López Rodrigo, J., Hernández Silva, M., & Mortega Mohedano, J. (2011). Aproximación pedagógica de las plataformas open source en las universidades españolas.

Lozano, J. (2008). El e-learning y su terminología.

MDN Web Docs. (s. f.). CSS. Mozilla. Recuperado el 5 de abril de 2026, de https://developer.mozilla.org/es/docs/Web/CSS

MDN Web Docs. (s. f.). HTML: Lenguaje de etiquetas de hipertexto. Mozilla. Recuperado el 5 de abril de 2026, de https://developer.mozilla.org/es/docs/Web/HTML

MDN Web Docs. (s. f.). HTTP. Mozilla. Recuperado el 5 de abril de 2026, de https://developer.mozilla.org/es/docs/Web/HTTP

MDN Web Docs. (s. f.). JavaScript. Mozilla. Recuperado el 5 de abril de 2026, de https://developer.mozilla.org/es/docs/Web/JavaScript

MDN Web Docs. (s. f.). Learn web development. Mozilla. Recuperado el 5 de abril de 2026, de https://developer.mozilla.org/en-US/docs/Learn

MDN Web Docs. (2025). Client-server overview. Mozilla. https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/First_steps/Client-Server_overview

Meta. (s. f.). React. Recuperado el 12 de noviembre de 2025, de https://react.dev/

Microsoft. (s. f.). Visual Studio Code. Recuperado el 5 de abril de 2026, de https://code.visualstudio.com/

Molina Ríos, J. R., & Zea Ordóñez, M. P. (2017). Metodologías de desarrollo de aplicaciones web. 3C Tecnología, 6(3), 54–71.

Moodle. (s. f.). Managing roles. Moodle Docs. Recuperado el 5 de abril de 2026, de https://docs.moodle.org/

Moodle. (s. f.). Moodle: Online learning with the world's most popular LMS. Recuperado el 10 de octubre de 2024, de https://moodle.com/

Moodle Docs. (s. f.). Course overview and timeline [Imagen]. Recuperado el 3 de mayo de 2026, de https://docs.moodle.org/all/es/images_es/thumb/5/59/docstimeline.png/600px-docstimeline.png

Mozilla. (s. f.). ¿Qué es un navegador web? Recuperado el 5 de abril de 2026, de https://www.mozilla.org/es-ES/firefox/browsers/what-is-a-browser/

MSMK. (2024, 4 de septiembre). Blackboard. MSMK University. https://msmk.university/blackboard/

Nielsen, J. (2012). Usability 101: Introduction to usability. Nielsen Norman Group. https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Node.js. (s. f.). About Node.js. Recuperado el 14 de noviembre de 2025, de https://nodejs.org/en/about

Oudat, Q., & Othman, M. (2024). Embracing digital learning: Benefits and challenges of using Canvas in education. Journal of Nursing Education and Practice, 14(10), 39. https://doi.org/10.5430/jnep.v14n10p39

Páez Cantillo, Y. (2023). Plataforma Edu 2.0: uso de recursos y actividades interactivas por estudiantes de formación básica. Telos: Revista de Estudios Interdisciplinarios en Ciencias Sociales, 25(1), 24–35. https://doi.org/10.36390/telos251.03

Pastor, O., Fons, J., & Pelechano, V. (2001). An object-oriented approach to automate web applications development. En International Conference on Conceptual Modeling (pp. 404–418). Springer.

Pastor, O., Fons, J., & Pelechano, V. (2008). OOWS: A method to develop web applications from web-oriented conceptual models. En G. Rossi, O. Pastor, D. Schwabe, & L. Olsina (Eds.), Web engineering: Modelling and implementing web applications (pp. 65–99). Springer.

Pelechano, V., Pastor, O., & Fons, J. (2003). Developing web applications from conceptual models. En CAiSE Workshops 2003 (pp. 13–24). CEUR Workshop Proceedings.

Pérez Rodríguez, Y., & Sánchez Torres, J. (2007). La metodología RMM (Relationship Management Methodology) aplicable al desarrollo del software educativo multimedia Topografía [Trabajo de diploma, Universidad de las Ciencias Informáticas].

Pressman, R. S. (2010). Ingeniería del software: un enfoque práctico (7.ª ed.). McGraw-Hill.

Prisma. (s. f.). Prisma ORM documentation. Recuperado el 16 de noviembre de 2025, de https://www.prisma.io/docs

Rescorla, E. (2018). The Transport Layer Security (TLS) protocol version 1.3 (RFC 8446). Internet Engineering Task Force. https://doi.org/10.17487/RFC8446

Roberts, T. S. (Ed.). (2004). Online collaborative learning: Theory and practice. Information Science Publishing.

Ros Martínez de Lahidalga, I. (2008). Moodle, la plataforma para la enseñanza y organización escolar. Ikastorratza. e-Revista de Didáctica, 2.

Rubio Fernández, A., Santamaría González, F., Muñoz Merino, P. J., & Delgado Kloos, C. (2017). A data collection experience with Canvas LMS as a learning platform. CEUR Workshop Proceedings, 1925.

Ruíz Reyes, N., Vera Candeas, P., García Galán, S., Viciana, R., Cañada, F., & Reche, P. J. (2009). Comparing open-source e-learning platforms from adaptivity point of view.

Sánchez Gordón, S. (2017). Desarrollo de un proceso de implementación de cursos en línea masivos y abiertos-accesibles.

Sánchez Rodríguez, J. (2005). Plataformas tecnológicas para el entorno educativo. Acción Pedagógica, 14(1), 18–24.

Sánchez Rodríguez, J. (2009). Plataformas de enseñanza virtual para entornos educativos. Pixel-Bit. Revista de Medios y Educación, (34), 217–233.

Santamaría, J. S. (2012). Usos pedagógicos de Moodle en la docencia universitaria desde la perspectiva de los estudiantes. Revista Iberoamericana de Educación, 60, 15–38.

Santander Open Academy. (2022, 30 de marzo). E-learning: qué es. Santander Open Academy. https://www.santanderopenacademy.com/es/blog/e-learning.html

Santos Regalado, O., López Ruiz, I., López Ruiz, A., Toledo Antonio, M., Peláez Estrada, U. V., & Girón Enriquez, Y. (2022). Campus Claroline como herramienta virtual educativa en el proceso enseñanza en el CCI. Innovación y Desarrollo Tecnológico Revista Digital, 14(1), 1214–1222.

Scott, M. L. (2016). Programming language pragmatics (4.ª ed.). Morgan Kaufmann.

Sebesta, R. W. (2012). Concepts of programming languages (10.ª ed.). Pearson.

Segovia-García, N. (2024). Análisis multidimensional de plataformas educativas: Canvas vs. Moodle en la educación superior. Revista Virtual Universidad Católica del Norte, (72), 4–39. https://doi.org/10.35575/rvucn.n72a2

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database system concepts (7.ª ed.). McGraw-Hill.

Sommerville, I. (2011). Ingeniería de software (9.ª ed.). Pearson Educación.

Tailwind Labs. (s. f.). Tailwind CSS documentation. Recuperado el 18 de noviembre de 2025, de https://tailwindcss.com/docs

Tanenbaum, A. S., & Wetherall, D. J. (2011). Computer networks (5.ª ed.). Pearson.

Tukey, J. W. (1958). The teaching of concrete mathematics. The American Mathematical Monthly, 65(1), 1–9.

Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, 42(2), 230–265.

TypeScript. (s. f.). The TypeScript handbook. Recuperado el 19 de noviembre de 2025, de https://www.typescriptlang.org/docs/handbook/intro.html

UNESCO. (2024, 17 de septiembre). What you need to know about digital learning and transformation of education. UNESCO. https://www.unesco.org/en/digital-education/need-know?hub=84636

Universidad de las Américas Puebla. (s. f.). Guía de uso de plataforma Blackboard para profesores. UDLAP.

Valderas, P., Pelechano, V., Pastor, O., & Torres, V. (2005). A requirements engineering approach for web applications. En International Conference on Web Engineering (pp. 628–638). Springer.

Vercel. (s. f.). Next.js documentation. Recuperado el 17 de noviembre de 2025, de https://nextjs.org/docs

Vidal Ledo, M. J., Rodríguez Dopico, R. M., & Martínez Hernández, G. (2014). Sistemas de gestión del aprendizaje. Educación Médica Superior, 28(3), 603–615.

Villalón, R., Luna, M., & García, A. (2019). Valoración y uso de la plataforma Blackboard Collaborate en una universidad a distancia: estudio de caso sobre las prácticas declaradas de docentes del Grado de Psicología. Digital Education, 35.

W3C. (s. f.). CSS. Recuperado el 5 de abril de 2026, de https://www.w3.org/Style/CSS/

WHATWG. (s. f.). HTML living standard. Recuperado el 5 de abril de 2026, de https://html.spec.whatwg.org/

Williams, D. (2022). Improving user experience in learning management systems: A case study about Canvas [Tesis de maestría, KTH Royal Institute of Technology].

Zapata-Ros, M. (2003). Evaluación de un sistema de gestión del aprendizaje. RED. Revista de Educación a Distancia, (9).

Zarta Rojas, D. (2013). Images 41 [Imagen]. WordPress. https://danielazartarojas.files.wordpress.com/2013/02/images-41.jpg

Bit4learn. (2019, 1 de diciembre). User experience [Imagen]. https://bit4learn.com/es/lms/user-experience-lms/

Caballero, P. A. (2017). Implementación de la plataforma educativa Schoology como un medio para el aprendizaje [Tesis de licenciatura, Universidad de Guayaquil]. Repositorio Institucional de la Universidad de Guayaquil. http://repositorio.ug.edu.ec/handle/redug/23118

Cervera Llop, J. (2007). Implantación de la plataforma de e-learning ATutor [Proyecto de final de carrera, Universitat Politècnica de Catalunya]. UPCommons. http://hdl.handle.net/2099.1/4351

Fakhreldeen Abbas Saeed. (2013). Comparing and evaluating open source e-learning platforms. International Journal of Soft Computing and Engineering, 3(3), 244–249. https://www.ijsce.org/wp-content/uploads/papers/v3i3/C171207331.pdf

Santivañez Bernardo, S. V. (2019). Aplicación de la plataforma virtual LMS para mejorar el programa de capacitación laboral en el Colegio Particular Andino – Huancayo 2019 [Tesis de licenciatura, Universidad Nacional del Centro del Perú]. Repositorio Institucional UNCP. https://repositorio.uncp.edu.pe/bitstreams/9f15b928-2520-4b76-af92-784b07fda8a4/download

Sistema LMS Wiki. (s. f.). Claroline [Imagen]. Fandom. Recuperado el 5 de mayo de 2026, de https://sistema-lms.fandom.com/es/wiki/Claroline?file=Claroline.png

Bennett, S., Bishop, A., Dalgarno, B., Waycott, J., & Kennedy, G. (2012). Implementing Web 2.0 technologies in higher education: A collective case study. Computers & Education, 59(2), 524–534. https://doi.org/10.1016/j.compedu.2011.12.022

Vercel. (s. f.). *Vercel documentation*. Recuperado el 4 de abril de 2026, de https://vercel.com/docs