APLICACIÓN WEB PARA LA COMPRA/VENTA DE RECURSOS EDUCATIVOS ONLINE DE FORMA SEGURA

WEB APPLICATION FOR THE SECURE PURCHASE/SALE OF EDUCATIONAL RESOURCES ONLINE

TRABAJO FIN DE GRADO
CURSO 2023-2024

AUTORES
LUCERO ABREGÚ REÁTEGUI
PABLO DELGADO GÓMEZ

DIRECTOR
RAMÓN GONZÁLEZ DEL CAMPO RODRÍGUEZ BARBERO

DOBLE GRADO EN INGENIERÍA INFORMÁTICA Y ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

APLICACIÓN WEB PARA LA COMPRA/VENTA DE RECURSOS EDUCATIVOS ONLINE DE FORMA SEGURA

WEB APPLICATION FOR THE SECURE PURCHASE/SALE OF EDUCATIONAL RESOURCES ONLINE

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTORES
LUCERO ABREGÚ REÁTEGUI
PABLO DELGADO GÓMEZ

DIRECTOR
RAMÓN GONZÁLEZ DEL CAMPO RODRÍGUEZ BARBERO

CONVOCATORIA: SEPTIEMBRE 2024

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

13 DE SEPTIEMBRE DE 2024

AGRADECIMIENTOS

Lo dejamos antes de la entrega definitiva por si acaso ponemos algo antes de entregarlo

RESUMEN

Aplicación Web Para La Compra/Venta De Recursos Educativos Online De Forma Segura

Este Trabajo de Fin de Grado (TFG) consiste en el desarrollo de una aplicación web destinada a la compraventa de recursos educativos mediante distintas funcionalidades importantes para la aplicación.

La aplicación ha sido desarrollada utilizando tecnologías modernas, con React y Tailwind para la construcción del front-end, y Node.js para la implementación del back-end, lo que permite una experiencia de usuario dinámica y eficiente, así como un manejo robusto del servidor.

Entre las funcionalidades principales se incluyen el registro de usuarios, la compra y oferta de cursos, la valoración de cursos, y la administración de diversas tablas relacionadas con los recursos educativos y la gestión de usuarios. Además, los usuarios pueden consultar la información de su cuenta personal, recibir notificaciones cuando se actualizan los cursos en los que están inscritos, asegurando que siempre estén al tanto de las novedades en sus cursos de interés.

El proyecto aborda tanto la creación de una interfaz intuitiva y amigable para el usuario, como la construcción de una infraestructura back-end segura y escalable, proporcionando una solución completa para la compra y venta de recursos educativos en línea.

Palabras clave

Aplicación web, compra/venta, recursos educativos, React, Tailwind, Node.js, interfaz intuitiva.

ABSTRACT

Web Application For The Secure Purchase/Sale Of Educational Resources Online

The Final Degree Project (TFG) involves the development of a web application designed for the buying and selling of educational resources, incorporating various key functionalities for the application.

The application has been developed using modern technologies, with React for the frontend construction and Node.js for the backend implementation. This combination ensures a dynamic and efficient user experience, as well as robust server management.

The main features include user registration, the purchase and offering of courses, course rating, and the administration of various tables related to educational resources and user management. Additionally, users can view their personal account information, receive notifications when courses they are enrolled in are updated, ensuring they are always aware of new developments in their courses of interest.

The project addresses both the creation of an intuitive and user-friendly interface and the construction of a secure and scalable backend infrastructure, providing a comprehensive solution for the online buying and selling of educational resources.

Keywords

Web aplication, purchase/sale, educational resources, React, Node.js, intuitive interface.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

Capítulo 1 - Introducción

La educación es un factor clave en el desarrollo personal y profesional, y en el panorama educativo actual, las herramientas digitales han pasado de ser un complemento opcional para convertirse en un elemento esencial que potencia el proceso de enseñanza y aprendizaje, además de ofrecer oportunidades laborales a los educadores. La rápida evolución de la tecnología ha dado lugar a una diversidad de recursos digitales que transforman la educación, brindando a educadores y estudiantes nuevas oportunidades para explorar y adquirir conocimientos de manera flexible y accesible (Gamarra, 2023).

Este trabajo se centra en el desarrollo de una plataforma web orientada a la compraventa de material docente, diseñada para atender las necesidades de alumnos y profesores. En este contexto, las aplicaciones web para la compraventa de material docente tienen una relevancia creciente debido a múltiples factores, entre los que destacan la accesibilidad, la flexibilidad y la capacidad de adaptarse a diferentes niveles y modalidades educativas.

Las plataformas educativas virtuales, como la desarrollada en este proyecto, se caracterizan por ofrecer una estructura que facilita la creación de entornos de aprendizaje en línea, permitiendo la gestión de contenidos y el seguimiento del progreso de los estudiantes. Estas plataformas no solo optimizan el proceso de enseñanza, sino que también presentan varias ventajas significativas, como el ahorro de tiempo y recursos, y la mejora de la participación de los estudiantes.

1.1 Motivación

La motivación para la realización de este Trabajo de Fin de Grado (TFG) surge de la creciente demanda por plataformas educativas accesibles y efectivas, especialmente en un contexto donde la digitalización del aprendizaje ha tomado un papel fundamental. La educación en línea ha demostrado ser una herramienta poderosa para democratizar el acceso al conocimiento, permitiendo que personas de distintas geografías y condiciones puedan aprender y mejorar sus habilidades a su

propio ritmo. Sin embargo, aún existen desafíos en cuanto a la disponibilidad y calidad de los recursos educativos, así como en la facilidad de uso de las plataformas existentes.

En este sentido, el desarrollo de una aplicación web para la compraventa de recursos educativos ofrece una solución a la necesidad de contar con un espacio centralizado y confiable donde educadores y estudiantes puedan intercambiar conocimientos de manera segura y eficiente. La elección de React y Node.js como tecnologías clave para este proyecto responde a la necesidad de crear una plataforma moderna, escalable y de alto rendimiento que pueda ofrecer una experiencia de usuario intuitiva y atractiva, a la vez que mantiene una infraestructura robusta y segura en el back-end.

Además, el proyecto refleja una preocupación por la calidad educativa, al incluir funcionalidades como la valoración de cursos y las notificaciones personalizadas, que no solo permiten a los usuarios encontrar recursos de alta calidad, sino que también fomentan una comunidad de aprendizaje activa y comprometida. La capacidad de los usuarios para gestionar su cuenta, realizar compras, ofrecer sus propios cursos y recibir actualizaciones relevantes, convierte a esta aplicación en una herramienta integral para el aprendizaje continuo y la educación colaborativa.

Finalmente, este TFG no solo representa un desafío técnico, sino también un compromiso con el avance de la educación digital, contribuyendo a la construcción de un entorno donde la educación de calidad sea accesible para todos, potenciando el intercambio de conocimientos y el desarrollo personal y profesional de los usuarios.

1.2 Objetivos

El objetivo principal es desarrollar una aplicación web para la compraventa de recursos educativos que permita a los distintos tipos de usuarios (alumno, profesor) comprar o vender cursos, apuntes, etc.

Los objetivos específicos por seguir para poder desarrollar el proyecto son:

- Inmersión en tecnologías modernas: profundizar en el uso de tecnologías clave como React para el front-end y Node.js para el back-end, optimizando el rendimiento y escabilidad de la plataforma.

- Análisis de mercado: se llevará a cabo un análisis del mercado de plataformas educativas para identificar las necesidades de alumnos y profesores, evaluando la competencia existente. Además, se determinará la viabilidad del proyecto en cuanto a demanda, rentabilidad y sostenibilidad a largo plazo.
- Desarrollo de una plataforma web intuitiva y segura: diseñar y desarrollar una plataforma web que facilite la compraventa de material docente, asegurando que tanto alumnos como profesores puedan navegar y utilizarla de manera intuitiva y segura.
- Análisis y evaluación del rendimiento: realizar pruebas para evaluar su rendimiento y seguridad, garantizando que la plataforma pueda manejar una demanda moderada de usuarios sin comprometer su funcionalidad.
- Optimización y escalabilidad futuras: planificar los cambios necesarios en la arquitectura que permita futuras expansiones y mejoras, asegurando que la aplicación pueda adaptarse a nuevas necesidades educativas y a un creciente número de usuarios.
- Documentación detallada del proceso: elaborar una documentación completa que describa cada fase del desarrollo incluyendo la justificación de decisiones técnicas y los retos superados durante el proceso.

1.3 Plan de trabajo

Para lograr los objetivos de desarrollar una aplicación web para la compraventa de recursos educativos, se ha estructurado el proyecto en varias fases clave, viéndose estas en la *Figura 1-1* a través de un diagrama de Gantt.

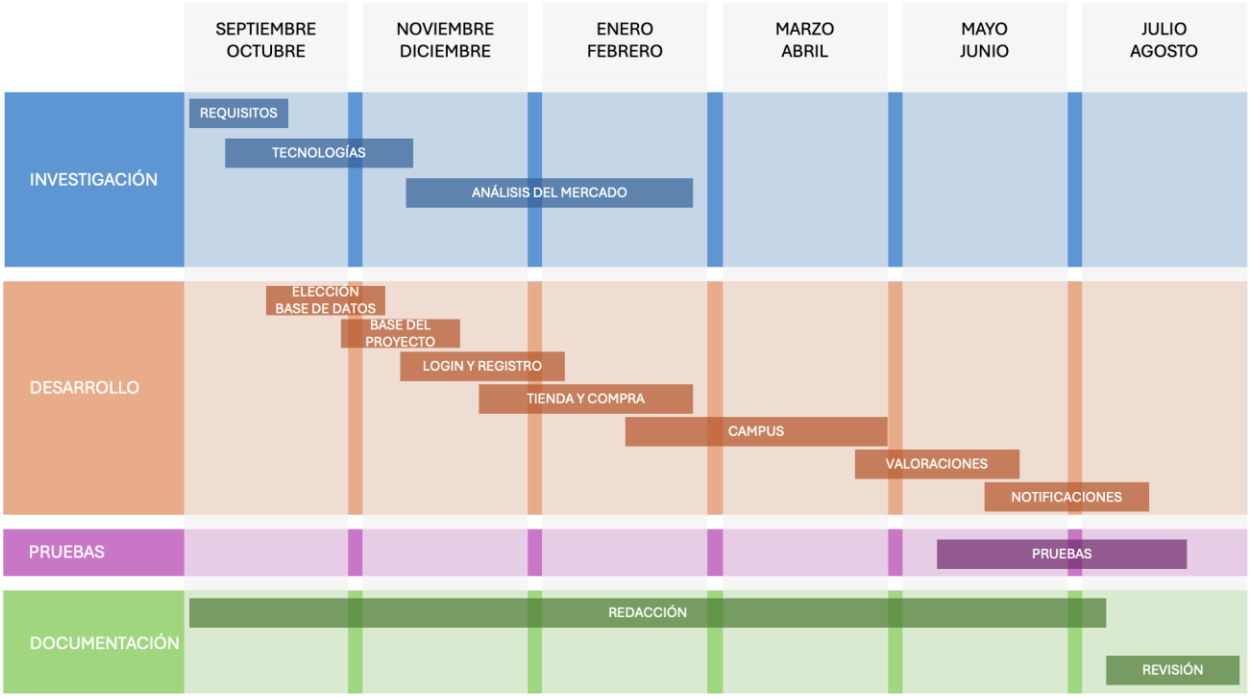


Ilustración 1 Diagrama de Gantt

A continuación, se presenta un plan de trabajo detallado para guiar el desarrollo de la aplicación:

- Investigación y análisis preliminar:
    - Exploración tecnológica: profundizar en el uso de tecnologías modernas como React para el front-end y Node.js para el back-end, evaluando sus capacidades y requerimientos para asegurar una implementación óptima.
    - Análisis del mercado: realizar un estudio exhaustivo del mercado de plataformas educativas para identificar las necesidades específicas de alumnos y profesores. Evaluar la competencia existente para detectar oportunidades de valor diferencial y determinar la viabilidad del proyecto en términos de demanda, rentabilidad y sostenibilidad a largo plazo.
- Desarrollo de la plataforma web:
    - Diseño de la aplicación: diseñar una plataforma web intuitiva y segura, enfocada en la compraventa de material docente. Esto incluye definir

las funcionalidades clave, los roles de usuario (alumno, profesor), y garantizar una experiencia de usuario fluida.

  - Implementación: desarrollar el front-end utilizando React y el back-end con Node.js. Se ha de asegurar que la plataforma cumpla con los estándares de seguridad y rendimiento requeridos, facilitando una navegación intuitiva para todos los usuarios.

- Evaluación del rendimiento y seguridad:
  - Pruebas exhaustivas: realizar pruebas completas para evaluar el rendimiento y la seguridad de la aplicación. Comprobar que la plataforma pueda manejar una carga moderada de usuarios sin comprometer su funcionalidad y seguridad.
  - Optimización: identificar y aplicar mejoras basadas en los resultados de las pruebas para asegurar una operación eficiente y segura de la aplicación.
- Planificación para la escalabilidad futura:
  - Arquitectura escalable: planificar y diseñar la arquitectura para permitir futuras expansiones y mejoras. Se ha de asegurar que la aplicación pueda adaptarse a nuevas necesidades educativas y a un creciente número de usuarios sin pérdida de rendimiento.
- Documentación y entrega:
  - Documentación completa: elaborar una documentación detallada que describa cada fase del desarrollo, incluyendo la justificación de decisiones técnicas, desafíos superados y recomendaciones para futuros desarrollos.
  - Entrega final: preparar y entregar todos los documentos y materiales desarrollados durante el proyecto, asegurando que la aplicación esté lista para su uso y revisión.

Introduction

Education is a key factor in personal and professional development, and in the current educational landscape, digital tools have evolved from being an optional complement to becoming an essential element that enhances the teaching and learning process, as well as providing job opportunities for educators. The rapid advancement of technology has led to a diversity of digital resources that are transforming education, offering both educators and students new opportunities to explore and acquire knowledge in a flexible and accessible manner (Gamarra, 2023).

This work focuses on the development of a web platform aimed at buying and selling teaching materials, designed to meet the needs of students and teachers. In this context, web applications for the trading of teaching materials are gaining increasing relevance due to several factors, including accessibility, flexibility, and the ability to adapt to different educational levels and modalities.

Educational virtual platforms, such as the one developed in this project, are characterized by providing a structure that facilitates the creation of online learning environments, allowing for content management and student progress tracking. These platforms not only optimize the teaching process but also offer several significant advantages, such as saving time and resources and improving student engagement.

Motivation

The motivation for undertaking this Bachelor's Thesis (TFG) arises from the growing demand for accessible and effective educational platforms, especially in a context where the digitalization of learning has become fundamental. Online education has proven to be a powerful tool for democratizing access to knowledge, allowing individuals from diverse geographies and backgrounds to learn and enhance their skills at their own pace. However, challenges remain regarding the availability and quality of educational resources, as well as the usability of existing platforms.

In this context, the development of a web application for buying and selling educational resources offers a solution to the need for a centralized and reliable space where educators and students can exchange knowledge securely and efficiently. The

choice of React and Node.js as key technologies for this project addresses the need to create a modern, scalable, and high-performance platform that can provide an intuitive and engaging user experience while maintaining a robust and secure back-end infrastructure.

Additionally, the project reflects a commitment to educational quality by including features such as course ratings and personalized notifications, which not only help users find high-quality resources but also foster an active and engaged learning community. The ability for users to manage their accounts, make purchases, offer their own courses, and receive relevant updates makes this application a comprehensive tool for continuous learning and collaborative education.

Ultimately, this thesis represents not only a technical challenge but also a commitment to advancing digital education, contributing to the creation of an environment where quality education is accessible to all, enhancing knowledge exchange and the personal and professional development of its users.

Goals

The main objective is to develop a web application for buying and selling educational resources that allows different types of users (students, teachers) to buy or sell courses, notes, and other materials.

The specific objectives to achieve this project are:

- Immersion in Modern Technologies: Deepen the use of key technologies such as React for the front-end and Node.js for the back-end, optimizing the platform's performance and scalability.
- Market Analysis: Conduct a market analysis of educational platforms to identify the needs of students and teachers, assessing existing competition. Additionally, determine the project's viability in terms of demand, profitability, and long-term sustainability.
- Development of an Intuitive and Secure Web Platform: Design and develop a web platform that facilitates the buying and selling of teaching materials, ensuring that both students and teachers can navigate and use it intuitively and securely.

- Performance Analysis and Evaluation: Conduct tests to evaluate the platform's performance and security, ensuring that it can handle a moderate user demand without compromising functionality.
- Future Optimization and Scalability: Plan necessary changes in the architecture to allow for future expansions and improvements, ensuring that the application can adapt to new educational needs and an increasing number of users.
- Detailed Documentation of the Process: Prepare comprehensive documentation that describes each phase of development, including the justification for technical decisions and the challenges overcome during the process.

Work Plan

To achieve the objectives of developing a web application for the buying and selling of educational resources, the project has been structured into several key phases, as seen in *Figure 2 Gantt Diagram*.

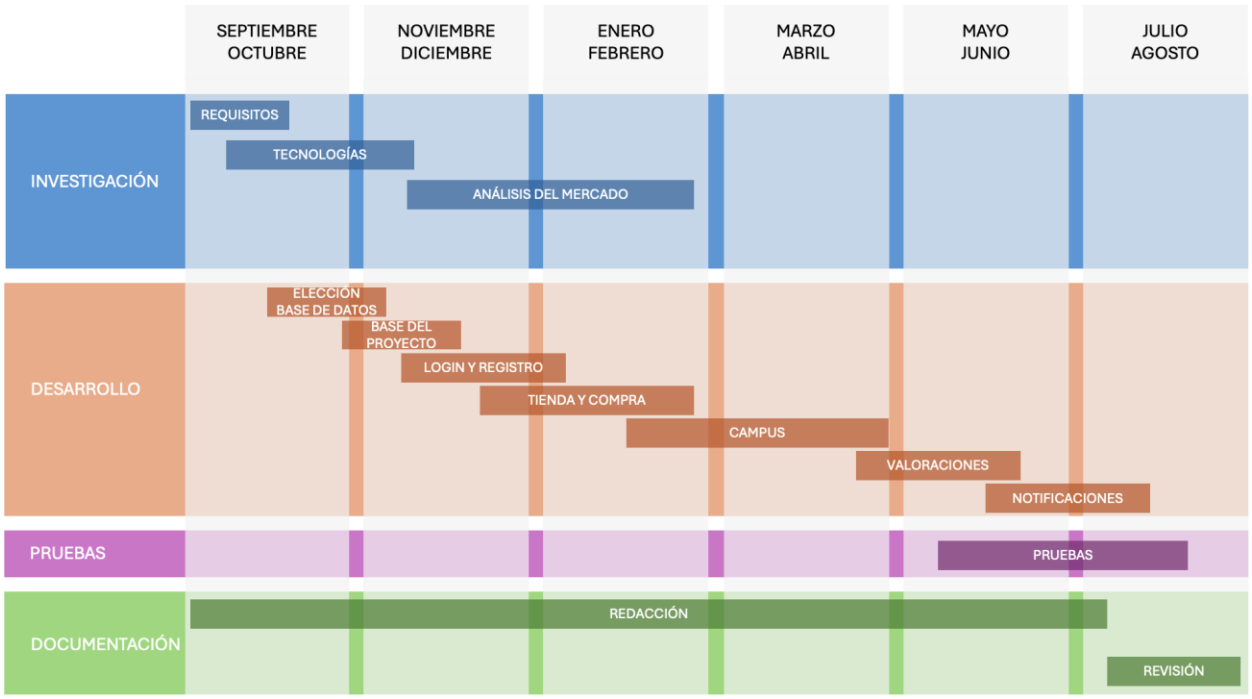


Figure 1 Gantt Diagram

Here is a detailed work plan to guide the development of the application:

• Preliminary Research and Analysis:

    - Technological Exploration: Delve deeper into the use of modern technologies like React for the front-end and Node.js for the back-end, assessing their capabilities and requirements to ensure optimal implementation.
    - Market Analysis: Conduct a comprehensive study of the educational platform market to identify the specific needs of students and teachers. Evaluate the existing competition to detect opportunities for differentiation and assess the project's viability in terms of demand, profitability, and long-term sustainability.
- Web Platform Development:
    - Application Design: Design an intuitive and secure web platform focused on the buying and selling of educational materials. This includes defining key functionalities, user roles (student, teacher), and ensuring a seamless user experience.
    - Implementation: Develop the front-end using React and the back-end with Node.js. The platform must comply with required security and performance standards, facilitating intuitive navigation for all users.
- Performance and Security Evaluation:
    - Comprehensive Testing: Carry out thorough testing to evaluate the application's performance and security. Ensure that the platform can handle a moderate user load without compromising functionality or security.
    - Optimization: Identify and apply improvements based on the test results to ensure efficient and secure operation of the application. • Future Scalability Planning:
    - Scalable Architecture: Plan and design the architecture to allow for future expansions and enhancements. Ensure the application can adapt to new educational needs and a growing user base without losing performance.
- Documentation and Delivery:
    - Complete Documentation: Create detailed documentation describing each phase of development, including the rationale for technical

decisions, challenges overcome, and recommendations for future developments.

- Final Delivery: Prepare and deliver all developed documents and materials, ensuring the application is ready for use and review.

Capítulo 2 - Fase de investigación

2.1 Análisis de mercado

Se realizará el análisis comparativo entre diversas plataformas en línea para evaluar de manera detallada las funcionalidades, la usabilidad y el valor diferencial de la plataforma web en desarrollo frente a otras soluciones existentes en el mercado. En particular, se realizará un análisis de Wuolah, el Campus Virtual de la Universidad Complutense de Madrid (UCM) y Scribd. Estas plataformas han sido seleccionadas por su relevancia y popularidad en el ámbito de la educación en línea y el intercambio de recursos educativos. Comparar la página web en desarrollo con Wuolah permitirá identificar características esenciales para la difusión de material educativo, mientras que el análisis de PACV, el Campus Virtual de la UCM, el cual aportará una visión sobre cómo integrar funcionalidades específicas para el entorno académico de manera electrónica. Por último, el estudio de Scribd será útil para entender cómo incentivar a los usuarios a comprar recursos educativos, destacando estrategias de monetización efectivas y cómo atraer a los usuarios a través de un modelo que garantice la calidad y el valor de los materiales ofrecidos. Este análisis comparativo será importante para identificar fortalezas y áreas de mejora.

2.1.1 Wuolah

Wuolah es una plataforma que permite a los estudiantes compartir y descargar apuntes de manera gratuita (Wuolah, s.f.). Los usuarios suben sus propios apuntes, que luego son descargados por otros usuarios. Wuolah monetiza esta actividad mediante la inclusión de publicidad tanto al momento de descargar los archivos como dentro de los propios apuntes. Los usuarios tienen la opción de pagar mediante una moneda virtual exclusiva de la página, la cual se adquiere por separado, para evitar la publicidad al descargar y/u obtener los documentos sin publicidad.

- Puntos fuertes:
    - Comunidad activa: Wuolah cuenta con una comunidad activa de estudiantes que suben y descargan apuntes regularmente,

incentivados por la oportunidad de ganar dinero al compartir sus materiales. Esta dinámica fomenta un flujo constante de contenido en la plataforma, asegurando una oferta de recursos educativos actualizados.

- Monetización accesible: la plataforma permite a los usuarios descargar apuntes de forma gratuita, monetizando a través de la publicidad. Esto lo hace atractivo para estudiantes que buscan recursos sin costo. Además, los usuarios tienen la opción de evitar la publicidad mediante un pago, lo que ofrece flexibilidad en la experiencia de usuario.
- Fácil acceso: la interfaz intuitiva y el sistema de búsqueda eficiente de la plataforma permite a los estudiantes encontrar rápidamente los materiales necesarios.

- Puntos débiles:
    - Dependencia de la publicidad: la experiencia de usuario puede verse afectada negativamente por la presencia en los apuntes descargados y durante el proceso de descarga. En ciertas ocasiones puede llegar a ser intrusiva, lo que desincentiva el uso de la plataforma de manera regular.
    - Calidad variable de los contenidos: dado que los apuntes son generados por los propios estuantes, la calidad de los materiales puede variar significativamente. Esto puede afectar la confianza de los usuarios en los recursos disponibles. Además, la falta de control riguroso de calidad podría resultar en la difusión de contenidos menos útiles o incorrectos.

Wuolah ha logrado crear una plataforma en el que los estudiantes no solo consumen recursos, sino que también contribuyen con sus propios apuntes. Este modelo participativo no solo aumenta la cantidad de contenido, sino que también fideliza a los usuarios al hacerlos sentir parte de una comunidad colaborativa. Sin embargo, es crucial mejorar el control de calidad del contenido disponible en la web y explorar y modelos de monetización alternativos que no sean invasivos para los usuarios.

2.1.2 PACV (Campus Virtual de la UCM)

El Campus Virtual de la Universidad Complutense de Madrid es una plataforma educativa que facilita a profesores y estudiantes acceder a materiales de curso, la participación en foros, la realización de evaluaciones y la gestión de diversas actividades académicas en línea (Madrid, s.f.) (Scribd, s.f.). Este servicio está incluido en la matrícula del grado universitario o máster, sin costos adicionales ni publicidad, lo que lo convierte, en esencia, en una herramienta gratuita para los usuarios de la universidad.

- Puntos fuertes:
    - Integración completa en el entorno académico: PACV está perfectamente integrado con los programas académicos de la universidad, lo que facilita el acceso a los materiales y recursos directamente relacionados con los cursos en los que los estudiantes están matriculados.
    - Acceso gratuito para estudiantes y profesores: el acceso a la plataforma está incluido en la matrícula de cada alumno en su universidad, lo que significa que los usuarios no necesitan realizar pagos adicionales para utilizar la web. Esto elimina barreras económicas y garantiza que todos los estudiantes tengan acceso igualitario a los recursos.
    - Diversidad de herramientas: la plataforma ofrece una amplia gama de funcionalidades, como foros, evaluaciones en línea, gestión de actividades académicas… permitiendo una experiencia educativa completa y unificada dentro de un solo entorno digital.
    - Privacidad y seguridad: al ser una plataforma oficial de la universidad, el Campus Virtual cuenta con estrictas medidas de seguridad y privacidad, protegiendo la información personal y académica de los usuarios.
- Puntos débiles:

- Interfaz poco intuitiva: la interfaz de PACV puede ser percibida como menos intuitiva en comparación con otras plataformas, lo que puede dificultar la navegación para algunos usuarios y disminuir la experiencia general.
- Limitaciones en el acceso a recursos externos: aunque es una plataforma robusta para recursos internos, el Campus Virtual de la UCM tiene limitaciones en la integración de recursos externos o en la incorporación de tecnologías de vanguardia que pueden enriquecer la experiencia de aprendizaje.

Se puede destacar en el Campus Virtual de la UCM la importancia de integrar herramientas educativas de manera que se adapten a las necesidades académicas específicas. Sin embargo, es fundamental mantener una interfaz de usuario moderna y sencilla, y mejorar la interactividad para atraer a más usuarios.

2.1.3 Scribd

Scribd es una plataforma global de suscripción que ofrece a los usuarios un acceso a variedad de documentos, libros, audiolibros y revistas (Scribd, s.f.). La plataforma permite la visualización previa de los contenidos, aunque de manera limitada. Para acceder a un documento completo, los usuarios deben optar por una suscripción de pago o, alternativamente, subir tres archivos propios que deben ser previamente aprobados por la plataforma, lo que les permite descargar un solo archivo de su elección.

- Puntos fuertes:
  - Amplio catálogo de contenidos: Scribd destaca por su extensa biblioteca de documentos, libros, audiolibros, y revistas, que abarca una gran variedad de temas y géneros. Esto convierte a la plataforma en un recurso integral para usuarios interesados en acceder a una amplia gama de contenidos, desde material educativo hasta entretenimiento. Esto lo convierte en un recurso valioso para la investigación y el aprendizaje autodirigido.

    - Funcionalidad de previsualización: la plataforma permite a los usuarios previsualizar una parte de los documentos antes de decidir si quieren pagar por acceder al contenido completo. Esta función es útil para atraer a los usuarios, ya que les permite evaluar la relevancia y calidad del material antes de realizar una compra.
    - Modelo de suscripción flexible: a diferencia de Wuolah, Scribd se basa en un modelo de suscripción que genera ingresos recurrentes, lo que permite la sostenibilidad a largo plazo.
- Puntos débiles:
    - Restricciones de acceso: aunque permite previsualizar el contendio, para acceder completamente a los documentos, es necesario pagar o subir documentos propios, lo que puede desalentar a algunos usuarios.
    - Suscripciones necesarias: la obligatoriedad de una suscripción, incluso después de la prueba gratuita, puede ser una barrera para usuarios que buscan acceso ocasional o gratuito.
    - Revisión y aprobación de contenido: el proceso de aprobación de los documentos subidos puede ser lento o riguroso, lo que podría desanimar a los usuarios que desean compartir rápidamente su contenido.

Lo más destacable de Scribd es la opción de previsualización, que permite a los usuarios evaluar el contenido ante de la compra. Sin embargo, es importante tener en cuenta la necesidad de suscripción para acceder al contenido completo, ya que esto puede limitar la accesibilidad para algunos usuarios.

2.1.4 Puntos clave

El análisis comparativo de plataformas como Wuolah, el Campus Virtual de la UCM y Scribd nos ha permitido identificar una serie de puntos fuertes y áreas de mejora que serán cruciales para el desarrollo de nuestra propia plataforma de compraventa de recursos educativos. Mientras Wuolah destaca por su modelo de monetización basado en la publicidad y las opciones de pago para eliminar anuncios, el Campus

Virtual de la UCM resalta por su integración sin costo adicional para los estudiantes dentro de su estructura académica, y Scribd ofrece un catálogo extenso y opciones de previsualización. Estos análisis nos muestran la necesidad de ofrecer una plataforma que combine lo mejor de cada uno: una interfaz intuitiva, un acceso flexible a materiales y la capacidad de manejar transacciones directamente entre usuarios, sin un coste adicional para la plataforma. Además, es crucial abordar los problemas identificados, como la calidad del contenido de Wuolah, la accesibilidad en el Campus Virtual, y el modelo de suscripción de Scribd, para crear una solución que no solo se diferencie en el mercado, sino que también optimice la experiencia del usuario y aproveche el hueco existente en el mercado.

Nuestra plataforma se destacará por facilitar un intercambio directo de dinero entre compradores y vendedores de cursos, eliminando así los costes adicionales asociados con intermediarios y ofreciendo una experiencia más directa y accesible para todos los usuarios.

Capítulo 3 - Tecnologías utilizadas

3.1 Desarrollo Front-end

3.1.1 React

Para el desarrollo del Front-end se ha utilizado React, una biblioteca de Javascript de código abierto que permite crear interfaces de usuario interactivas y dinámicas ideales para el desarrollo de aplicaciones web modernas. La sintaxis que utiliza combina JavaScript con HTML mediante la extensión JSX. Es mantenido por Facebook y la comunidad (React, s.f.).

3.1.1.1 Componentes

Una de las principales ventajas de React es su arquitectura basada en componentes. Esto permite dividir la interfaz de usuario en piezas más pequeñas y manejables, que pueden ser desarrolladas, testeadas y reutilizadas de manera independiente.

3.1.1.2 Hooks

La adopción de React Hooks ha permitido gestionar el estado y los efectos secundarios de manera más simple y eficiente. Por ejemplo, el hook useState se ha utilizado para manejar los estados de los componentes, mientras que useEffect ha sido clave para gestionar operaciones asíncronas, como la obtención de datos desde el backend. También existe el hook useContext que permite compartir datos entre componentes sin necesidad de pasar props a través de cada nivel del árbol de componentes.

3.1.1.3 Virtual DOM

React utiliza un DOM virtual (VOM) para minimizar las manipulaciones directas del DOM real, lo que mejora significativamente el rendimiento de la aplicación. Esto es particularmente beneficioso en aplicaciones que requieren un alto nivel de interactividad y actualizaciones frecuentes en la interfaz.

3.1.1.4 Router

Para la navegación dentro de la aplicación, se ha implementado React Router, que permite manejar de manera sencilla rutas y vistas diferentes dentro de la misma aplicación de una sola página. Esto mejora la experiencia de usuario al proporcionar transiciones fluidas entre las distintas secciones de la plataforma (React Router, s.f.).

3.1.1.5 Integración con APIs

La integración con el backend es un componente esencial en el desarrollo de esta aplicación. Se ha utilizado Axios, una popular librería de JavaScript, para manejar las solicitudes HTTP (Axios, s.f.).

3.2 Desarrollo Back-end

3.2.1 Node.js

Node.js es un entorno de ejecución de JavaScript en el servidor que permite construir aplicaciones web rápidas y escalables. Para este proyecto, Node.js se ha utilizado como base para el desarrollo del backend, encargado de manejar la lógica de negocio y las comunicaciones con la base de datos (Nodejs, s.f.).

3.2.1.1 Servidor asíncrono

Node.js destaca por su modelo de I/O asíncrono y no bloqueante, lo que permite manejar múltiples solicitudes simultáneamente sin degradar el rendimiento del servidor. Esto es esencial en una aplicación donde los usuarios interactúan frecuentemente con la plataforma.

3.2.1.2 Express.js

Sobre Node.js, se ha utilizado Express.js, un framework minimalista y flexible para la construcción de aplicaciones web y APIs. Express facilita la creación de rutas, la gestión de peticiones HTTP y la integración con middleware (Expressjs, s.f.).

3.2.1.3 API RESTful

El backend de la aplicación se ha estructurado como una API RESTful, lo que permite una comunicación clara y estructurada entre el frontend (React) y el backend (Node.js). Esta API maneja todas las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) necesarias para la gestión de usuarios, cursos, valoraciones, y demás funcionalidades de la plataforma.

3.2.1.4 Gestión de la seguridad

Se han implementado medidas de seguridad utilizando middleware en Express, como jsonwebtoken para la autenticación de usuarios mediante tokens JWT (JSON Web Tokens), garantizando que solo los usuarios autenticados puedan acceder a ciertas partes de la aplicación. Además, se ha utilizado bcrypt para la encriptación de contraseñas y CORS para asegurar que solo los dominios especificados pueden interactuar con la API del servidor .

3.3 Base de datos

La base de datos es un componente fundamental en cualquier aplicación web, ya que se encarga de almacenar, gestionar y recuperar la información de manera eficiente y segura. En este proyecto, se ha utilizado una base de datos relacional con MySQL, gestionada a través de phpMyAdmin, para manejar la persistencia de datos.

3.3.1 MySQL

MySQL es un sistema de gestión de bases de datos relacional (RDBMS) ampliamente utilizado que se basa en el modelo relacional para almacenar y gestionar datos. Es conocido por su robustez, rendimiento y escalabilidad. MySQL sigue el modelo de datos relacional, utilizando tablas para almacenar datos en filas y columnas, y relaciones entre estas tablas para mantener la integridad y la coherencia de los datos.

Características Principales:

- **Relacional**: Utiliza tablas para estructurar los datos y permite definir relaciones entre diferentes tablas mediante claves primarias y foráneas.
- **Transacciones**: Soporta transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) para asegurar la integridad de los datos.

- **Escalabilidad**: Ofrece opciones para escalar tanto vertical como horizontalmente, manejando grandes volúmenes de datos y altas tasas de transacciones.

Uso en el Proyecto: MySQL se ha elegido por su fiabilidad y capacidad para manejar datos estructurados de manera eficiente, siendo adecuado para la gestión de usuarios, cursos, valoraciones y otras entidades en la aplicación.

3.3.2 phpMyAdmin

phpMyAdmin es una herramienta de administración de bases de datos MySQL basada en web. Proporciona una interfaz gráfica que facilita la gestión de bases de datos y permite realizar operaciones complejas sin necesidad de interactuar directamente con la línea de comandos (phpMyAdmin, s.f.).

Características Principales:

- **Interfaz Gráfica**: Permite la administración de bases de datos a través de una interfaz de usuario intuitiva, simplificando tareas como la creación y modificación de tablas, así como la ejecución de consultas SQL.
- **Gestión de Esquemas**: Facilita la visualización y edición de esquemas de bases de datos, así como la importación y exportación de datos.
- **Consultas SQL**: Permite ejecutar consultas SQL directamente desde la interfaz, lo que facilita la prueba y el desarrollo de consultas.

Uso en el Proyecto: phpMyAdmin se utiliza para gestionar la base de datos de manera eficiente durante el desarrollo y mantenimiento de la aplicación, proporcionando herramientas útiles para la administración de esquemas y datos.

3.3.3 Conexión con Node.js

Node.js se conecta a MySQL mediante la biblioteca específica de mysql, que permite la interacción entre el entorno de ejecución de JavaScript y la base de datos.

3.4 Diseño

El diseño de la aplicación web ha sido implementado utilizando Tailwind CSS, un framework de CSS de utilidad que permite construir interfaces de usuario personalizadas y responsive de manera eficiente.

3.4.1 Tailwind CSS

Tailwind CSS es un framework de CSS que proporciona una amplia gama de clases utilitarias predefinidas, permitiendo a los desarrolladores aplicar estilos directamente en el HTML sin necesidad de escribir CSS personalizado. A diferencia de otros frameworks que ofrecen componentes preestablecidos, Tailwind CSS se basa en una metodología de diseño basada en utilidades, facilitando una mayor flexibilidad y control sobre el estilo de la aplicación (Tailwind CSS, s.f.).

Características Principales:

- **Clases Utilitarias**: Ofrece clases para propiedades CSS comunes como márgenes, padding, colores, tipografía y flexbox, que se pueden combinar para crear diseños personalizados.
- **Responsividad**: Incluye utilidades para crear diseños responsivos con facilidad, permitiendo ajustar el diseño para diferentes tamaños de pantalla mediante clases específicas.
- **Configuración Personalizable**: Permite personalizar y extender el diseño utilizando un archivo de configuración (tailwind.config.js), lo que facilita la adaptación del framework a los requerimientos específicos del proyecto.

Ventajas:

- **Flexibilidad y Control**: Tailwind CSS ofrece un alto nivel de flexibilidad al permitir la combinación de clases utilitarias para diseñar componentes específicos. Esto evita la necesidad de escribir CSS adicional y proporciona un control detallado sobre el estilo de cada elemento.

- **Desarrollo Rápido**: La metodología de clases utilitarias permite aplicar estilos rápidamente, reduciendo el tiempo de desarrollo al evitar la necesidad de crear hojas de estilo complejas y extensas.
- **Consistencia en el Diseño**: Al utilizar clases utilitarias predefinidas, Tailwind CSS ayuda a mantener la coherencia en el diseño a lo largo de la aplicación, ya que las mismas clases se aplican de manera uniforme en diferentes componentes.
- **Escalabilidad**: La capacidad de personalizar el archivo de configuración y extender el framework facilita la adaptación del diseño a medida que la aplicación crece, asegurando que el estilo pueda evolucionar sin necesidad de reescribir el CSS existente.

3.4.2 Font Awesome

Font Awesome es una popular biblioteca de iconos vectoriales que proporciona una amplia variedad de íconos listos para usar en aplicaciones web. En este proyecto, Font Awesome se ha utilizado para mejorar la interfaz de usuario añadiendo iconos visualmente atractivos y funcionales (Font Awesome, s.f.).

Características Principales:

- **Amplia Colección de Íconos:** Font Awesome ofrece miles de íconos en diferentes estilos (sólido, regular, ligero, etc.), lo que facilita encontrar el ícono adecuado para cada necesidad en la aplicación.
- **Personalización con CSS:** Los íconos de Font Awesome son altamente personalizables. Se pueden ajustar en tamaño, color, sombra y otros estilos directamente a través de CSS, permitiendo una integración fluida con el diseño global de la aplicación.
- **Compatibilidad con Tailwind CSS:** Al usar clases utilitarias de Tailwind CSS, los íconos de Font Awesome se pueden estilizar de manera coherente con el resto de la interfaz de usuario.

3.5 Entorno de desarrollo

El entorno de desarrollo es crucial para la eficiencia y efectividad en la programación y la gestión del proyecto. En este proyecto, se ha utilizado Visual Studio Code (VS Code) como principal entorno de desarrollo integrado (IDE). A continuación, se detallan los aspectos clave relacionados con el uso de VS Code.

3.5.1 Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft. Es conocido por su versatilidad, velocidad y extensibilidad. VS Code es un editor ligero pero potente, que ofrece una serie de características que mejoran la productividad del desarrollador (Visual Studio Code, s.f.).

Características Principales:

- **Editor de Código**: Proporciona un editor de texto eficiente y enriquecido con características avanzadas como resaltado de sintaxis, completado automático y navegación por el código.
- **Extensibilidad**: Permite la instalación de extensiones para añadir funcionalidades adicionales, como soporte para diferentes lenguajes de programación, herramientas de depuración, y más.
- **Integración con Git**: Incluye soporte integrado para control de versiones con Git, facilitando la gestión del código fuente y la colaboración en equipo.
- **Terminal Integrado**: Ofrece un terminal integrado que permite ejecutar comandos directamente dentro del editor, mejorando la fluidez del flujo de trabajo.

3.5.2 Github

Github es una plataforma de alojamiento de repositorios Git que facilita la gestión del código fuente, el control de versiones y la colaboración entre desarrolladores. A lo largo del desarrollo del proyecto, GitHub ha sido utilizado para almacenar y versionar el código, así como para facilitar la colaboración y el seguimiento de las tareas del proyecto.

Características Principales:

- **Control de Versiones con Git**: GitHub se basa en Git, un sistema de control de versiones distribuido que permite llevar un historial detallado de todos los cambios realizados en el código. Esto facilita la revisión y la reversión de cambios cuando sea necesario.
- **Colaboración y Pull Requests**: GitHub facilita la colaboración entre varios desarrolladores a través de *pull requests*, que permiten proponer, revisar y fusionar cambios en el código. Esto es especialmente útil para gestionar contribuciones y asegurar que todo el código se revisa antes de integrarse en la rama principal.
- **Issues y Gestión de Proyectos**: La función de *issues* en GitHub permite rastrear errores, solicitar nuevas características y gestionar tareas. Además, las herramientas de gestión de proyectos de GitHub, como los *projects boards*, permiten organizar y priorizar el trabajo de manera visual y eficiente.
- **Integraciones y CI/CD**: GitHub se integra con diversas herramientas de integración continua y entrega continua (CI/CD), lo que permite automatizar pruebas, despliegues y otras tareas repetitivas.

3.5.3 XAMPP

XAMPP es un paquete de software de código abierto que proporciona una solución completa para configurar un servidor local, integrando Apache (servidor web), MySQL (sistema de gestión de bases de datos), PHP (lenguaje de programación) y Perl. Este entorno de desarrollo se utiliza para probar y desarrollar aplicaciones web en un entorno controlado antes de su despliegue en un servidor de producción (Github, s.f.).

Características Principales:

- **Facilidad de Configuración:** XAMPP simplifica la instalación y configuración de un entorno de servidor local, lo que permite a los desarrolladores iniciar rápidamente un servidor Apache con MySQL y PHP.

- **Gestión de Bases de Datos:** Incluye phpMyAdmin, una herramienta basada en web para la gestión de bases de datos MySQL, que facilita la administración de bases de datos durante el desarrollo.
- **Entorno de Pruebas Local:** Permite probar aplicaciones web localmente en un entorno que simula un servidor real, lo que es crucial para depurar y ajustar aplicaciones antes de su lanzamiento.
- **Compatibilidad Multiplataforma:** Disponible para Windows, macOS y Linux, XAMPP es una herramienta versátil que se adapta a diferentes entornos de desarrollo.

Uso en el Proyecto:

- **Servidor Local para Desarrollo:** XAMPP se ha utilizado como entorno de desarrollo local para ejecutar y probar la aplicación web durante su desarrollo. Esto permitió realizar ajustes y pruebas de manera rápida y eficiente sin la necesidad de un servidor en la nube.
- **Gestión de MySQL:** Gracias a la integración de phpMyAdmin en XAMPP, se facilitó la gestión y administración de la base de datos MySQL utilizada en el proyecto, permitiendo crear y modificar tablas, así como realizar consultas SQL de forma sencilla.

Capítulo 4 - Desarrollo de la Aplicación

4.1 Estructura de la aplicación

La estructura de la aplicación es fundamental para garantizar un desarrollo organizado y escalable, permitiendo una clara separación de responsabilidades entre los distintos componentes del sistema. Esta arquitectura facilita la implementación de nuevas funcionalidades, el mantenimiento del código y la colaboración entre los miembros del equipo. A continuación, se describe cómo se ha estructurado la aplicación, detallando los diferentes módulos y componentes que la conforman, y cómo se integran para proporcionar una experiencia de usuario cohesiva y eficiente.

En primer lugar, se va a mostrar la estructura general en la siguiente figura.

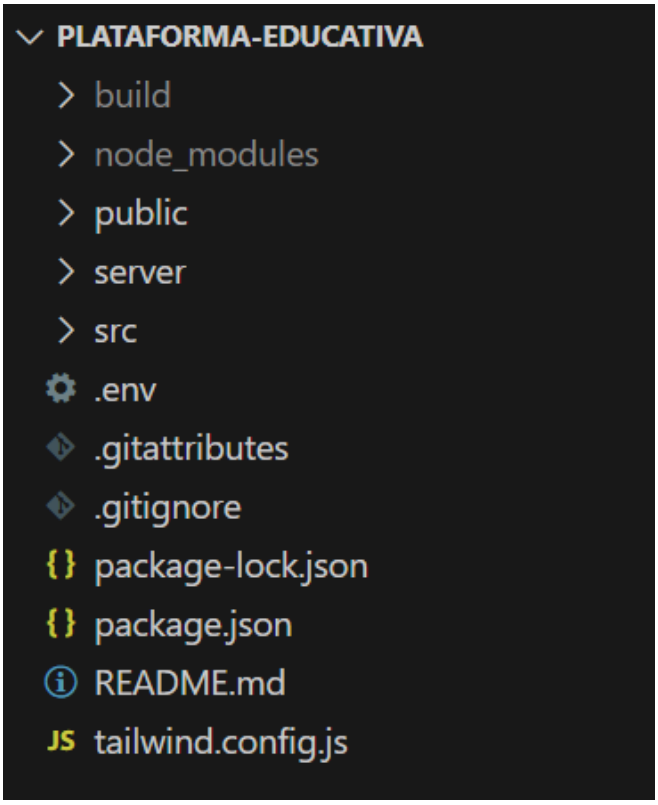


Ilustración 2 Estructura general de la aplicación

La aplicación se llama plataforma-educativa y está dividida en diferentes directorios:

- **build**: no se ha tocado, viene predeterminado.

- **node_modules**: es una carpeta clave en los proyectos de Node.js, ya que almacena todas las dependencias y módulos de terceros que tu proyecto necesita para funcionar correctamente. Cuando se actualizan dependencias o instalan nuevas, se actualiza para reflejar los cambios realizados.
- **public**: contiene los archivos estáticos que serán servidos directamente al navegador del usuario. Lo más importante que contiene es el index.html, que es el archivo HTML que sirve de raiz para montar la aplicación usando React.
- **server**: contiene todo el código relacionado con el backend de la aplicación. Este directorio se encarga de manejar la lógica del servidor, interactuar con la base de datos, gestionar las API, la seguridad, y cualquier otro proceso que no esté directamente relacionado el frontend.
- **src:** organiza el código fuente del frontend de la aplicación en una manera que facilita el desarrollo, mantenimiento y escalabilidad.
- **.env**: contiene la clave secreta para la autenticación.
- **.gitattributes**: asegura que Git gestione automáticamente la conversión de finales de línea según el sistema operativo del usuario.
- **.gitignore:** contiene una lista de patrones que especifican los archivos y directorios que Git debe ignorar.
- **package-lock.json:** esencial para la gestión precisa y consistente de dependencias en proyectos Node.js. Garantiza que el proyecto funcione de manera predecible en todos los entornos al proporcionar un registro detallado de las versiones exactas de las dependencias y sus subdependencias.
- **package.json:** fundamental para la gestión de proyectos Node.js, proporcionando una forma estructurada de definir dependencias, scripts, y metadatos del proyecto. Permite a los desarrolladores gestionar de manera eficiente las configuraciones del proyecto y asegurar que las dependencias y herramientas utilizadas sean consistentes en todos los entornos.
- **tailwind.config.js:** esencial para adaptar Tailwind CSS a las necesidades específicas de tu proyecto.

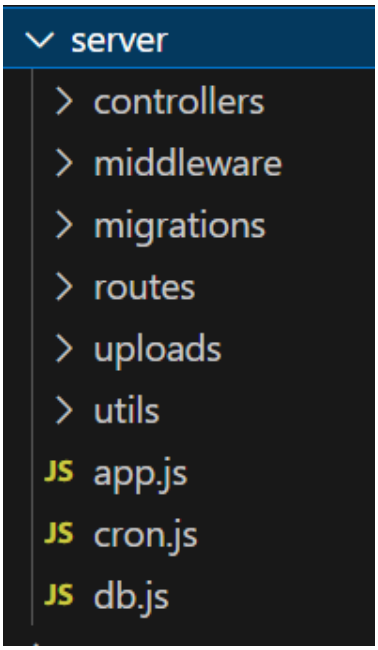


Ilustración 3 Estructura carpeta Server

- **controllers:** se encargan de manejar la lógica de negocio de la aplicación. Su función principal es recibir las solicitudes (requests) en una ruta específica de la API, procesarlas y devolver una respuesta (response) adecuada. Realizan las consultas de SQL para realizar las operaciones CRUD.
- **middleware:** contiene un middleware que verifica la validez de los tokens JWT para proteger rutas y garantizar que solo los usuarios autenticados puedan acceder a ciertos recursos.
- **migrations:** contiene los archivos .sql para la creación de la base de datos y la inserción de los datos.
- **routes:** definen los puntos de entrada de la API y vinculan las solicitudes HTTP a los controladores específicos que deben manejar esas solicitudes. En Express, las rutas son responsables de definir qué controlador se debe invocar para una ruta y un método HTTP dados (GET, POST, PUT, DELETE).
- **uploads:** contiene todos los archivos que se suban a los cursos del campus.
- **utils:** contiene funciones y módulos que ofrecen funcionalidades auxiliares y que pueden ser reutilizados en varias partes del proyecto. En este caso las relacionadas con las cookies.

- **app.js:** fundamental para la configuración y ejecución de la aplicación. En él se configuran el middleware, las rutas y el servidor. Este archivo actúa como el punto de entrada principal para la aplicación, centralizando la configuración y la inicialización de los componentes clave.
- **cron.js:** gestiona y ejecuta tareas programadas en intervalos de tiempo específicos.
- **db.js:** crea la conexión con la base de datos.

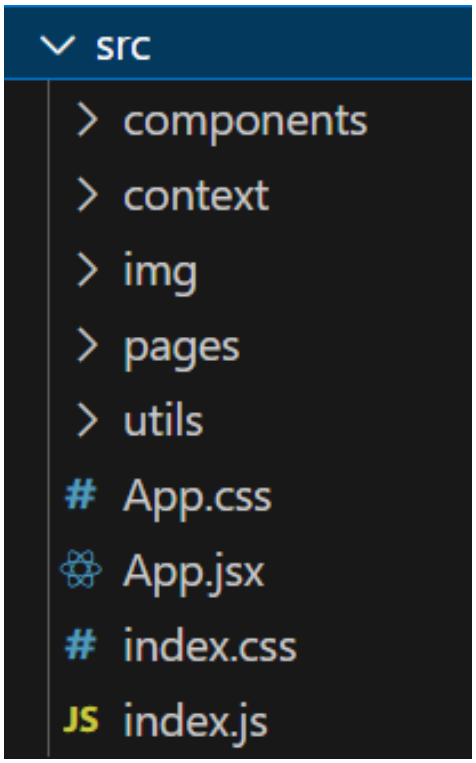


Ilustración 4 Estructura carpeta Src

- **components:** contiene los componentes de la interfaz de usuario (UI) que son reutilizables en toda la aplicación. Estos componentes son piezas básicas que conforman las vistas y páginas de la aplicación.
- **context:** contiene los contextos de React, que permiten manejar el estado global y compartir datos entre componentes sin necesidad de pasar props manualmente por todos los niveles del árbol de componentes.
- **img:** contiene las imágenes necesarias para el contenido visual de la aplicación.

- **pages:** contiene las páginas de la aplicación, que son los componentes que representan vistas completas o pantallas de la aplicación. Cada página corresponde a una ruta o URL específica en la aplicación.
- **utils:** contiene funciones y módulos que ofrecen funcionalidades auxiliares y que pueden ser reutilizados en varias partes del proyecto. En este caso las relacionadas con las cookies.
- **App.css:** archivo de hoja de estilos en cascada (CSS) que se utiliza para definir los estilos globales aplicados a la aplicación React.
- **App.jsx:** es el archivo de componente principal en React que organiza la estructura de la aplicación, integrando componentes, rutas y lógica básica. Es esencialmente el punto de partida para el renderizado de la UI.
- **index.css:** archivo de hoja de estilos en cascada (CSS) que contiene los estilos globales para la aplicación.
- **index.js:** es el punto de entrada de la aplicación React, responsable de montar el componente principal (App) en el DOM y de configurar cualquier característica global como enrutamiento o estado global.

4.2 Base de datos

Para la implementación de la base de datos, como se ha mencionado anteriormente, se ha optado por utilizar phpMyAdmin, sirviendo como núcleo de almacenamiento y gestión de la información de la aplicación web, proporcionando una estructura organizada y eficiente para el manejo de datos.

4.2.1 Alcance de la base de datos

El alcance de la base de datos en la plataforma se define por los diferentes módulos funcionales que cubren las necesidades clave del sistema. Las funcionalidades que facilitan la gestión integral de la plataforma abarcan la administración de usuarios, la organización y manejo de cursos, el proceso de compras y la gestión del carrito, la comunicación entre los usuarios y el sistema, y la valoración de los cursos y servicios ofrecidos.

El sistema se estructura en torno a los siguientes módulos:

- Gestión de usuarios: este módulo se encarga de la administración completa de los perfiles de usuario dentro de la plataforma. Incluye procesos clave como el registro de nuevos usuarios, la autenticación y el inicio de sesión, así como la gestión y actualización de las suscripciones de cada usuario existente. Este módulo se encarga de que cada usuario pueda acceder a la plataforma de acuerdo con sus roles y permisos específicos.
- Gestión de cursos: en este módulo se gestiona toda la información relacionada con los cursos ofrecidos en la plataforma. Esto incluye la creación, modificación y eliminación de cursos, así como la asociación a sus respectivos profesores y de los archivos relevantes a cada curso. Estos se organizan de manera que los usuarios puedan acceder fácilmente a los contenidos ofrecidos y realizar un seguimiento de estos.
- Gestión de compras y carrito: este módulo maneja el proceso de compra de cursos y la gestión del carrito de compras. Los usuarios pueden añadir cursos a su carrito, proceder a la compra, y se registran las transacciones en la base de datos, añadiéndolos a su campus virtual para poder disfrutar del contenido del curso. También se realiza un seguimiento de las compras y la validez de los contratos asociados.
- Gestión de comunicación: a través de este módulo, se facilita la comunicación entre los usuarios y el sistema mediante mensajes y notificaciones. Los usuarios pueden enviar y recibir mensajes, según su tipo de rol, relacionados con diversos aspectos de la plataforma, y las notificaciones se utilizan para informar a los usuarios sobre actualizaciones relevantes, como compras realizadas y fin de suscripciones, entre otros.
- Gestión de valoraciones: este módulo permite a los usuarios valorar los cursos y proporcionar retroalimentación. Las valoraciones incluyen comentarios y calificaciones que se almacenan en la base de datos para su posterior visualización, ayudando a otros usuarios a tomar decisiones informadas sobre qué cursos seguir.

4.2.2 Normalización

Desde los inicios del desarrollo de la aplicación, se ha aplicado el principio de normalización para asegurar la integridad y eficiencia de la base de datos. La normalización es un proceso que busca eliminar redundancias y asegurar la coherencia de los datos mediante la organización estructurada de tablas. Este enfoque no solo ha facilitado la gestión de datos, sino que también optimiza las consultas y garantiza la integridad referencial entre las distintas entidades del sistema, lo cual podemos ver en el siguiente apartado del documento *Tablas y relaciones*.

Se ha alcanzado hasta la Tercera Forma Normal (3FN), asegurando que cada tabla esté libre de redundancias innecesarias y dependencias funcionales incorrectas, minimizando las anomalías durante las operaciones de inserción, actualización y eliminación.

Un ejemplo de esto se encuentra en las tablas "cursos" donde en vez de añadir varios campos pertenecientes a los datos del profesor asociado en una misma tupla, para evitar redundancia de datos, se creó una tabla "curso_profesor" donde se relaciona la tabla usuarios con la de "cursos", donde la información del profesor se encuentra en la tabla usuarios, la información de los cursos se encuentra en la tabla "cursos" y la relación del curso con su profesor se encuentra en la tabla "curso_profesor". Así, se evita la duplicación de información y se mejora la escalabilidad, ya que, si un curso tiene más de un profesor, solo se debe añadir más filas a la tabla "curso_profesor", en lugar de tener que agregar nuevas columnas. Separando los datos de cursos y profesores en tablas distintas, cualquier cambio en la información de un profesor se puede realizar en un solo lugar, y esto se reflejará automáticamente en todos los cursos asociados.

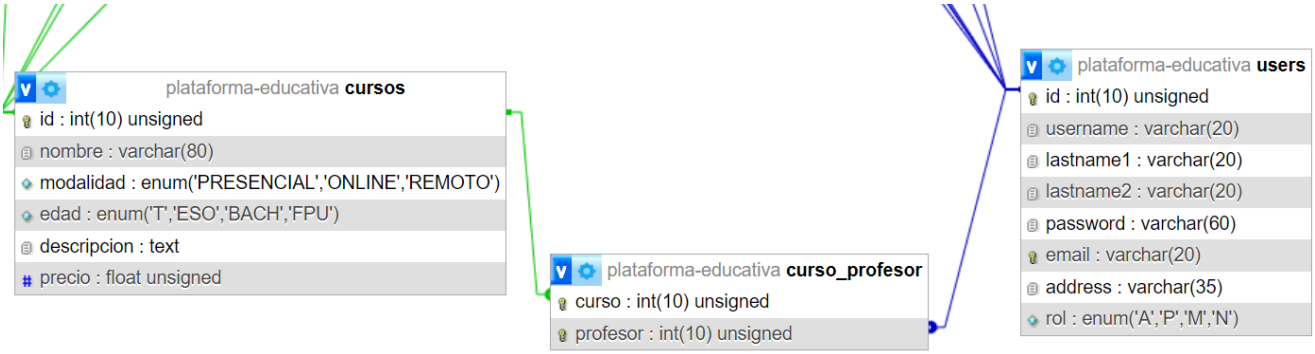


Ilustración 5 Ejemplo de normalización

Al añadir las nuevas tablas, también fue necesario crear relaciones entre estas, mediante el uso de claves primarias y foráneas, garantizando la consistencia de los datos

4.2.3 Tablas y relaciones

En la *Ilustración 6* se muestra el diagrama entidad-relación que se ha diseñado para el correcto funcionamiento de la aplicación:

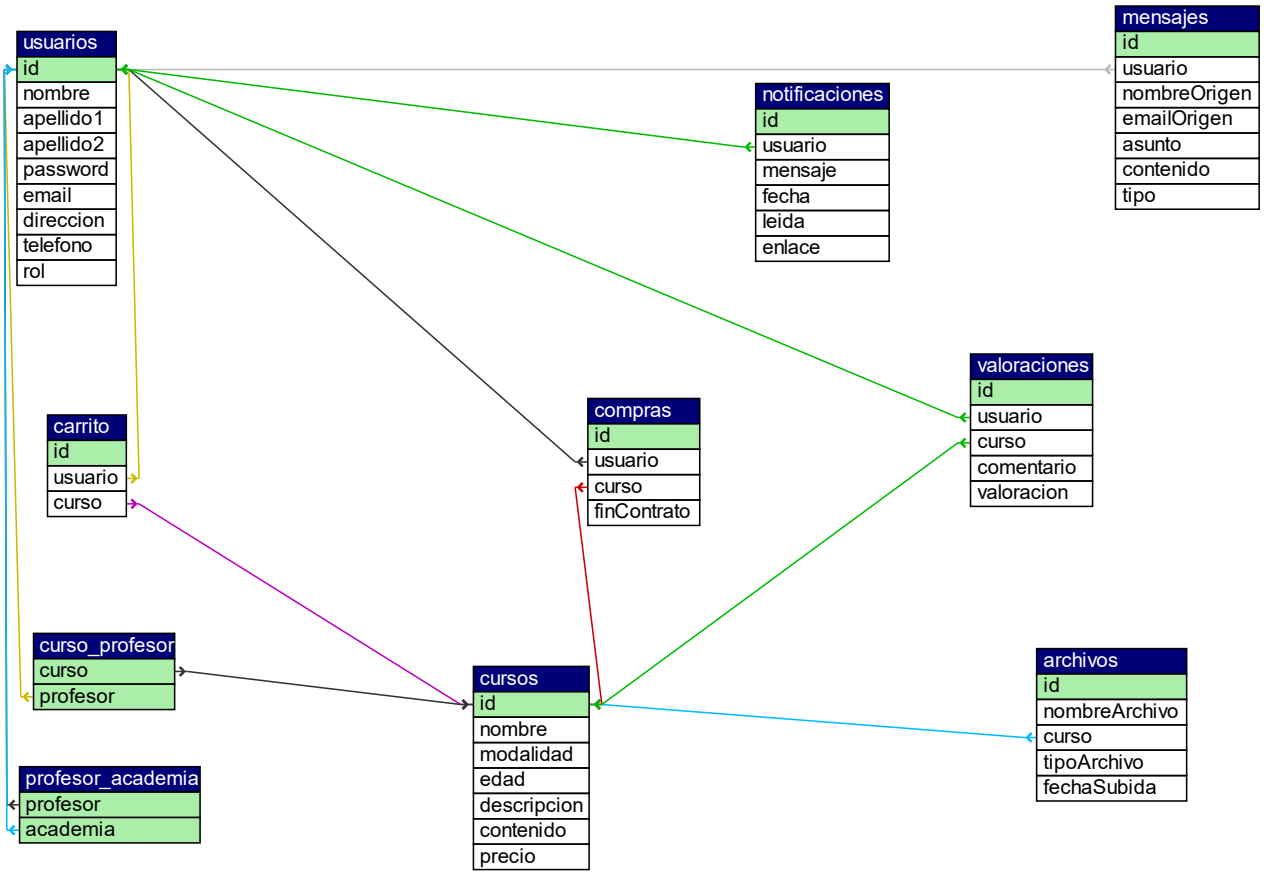


Ilustración 6 Estructura de la base de datos

Dado que la base de datos es una parte fundamental del sistema, su estructura y diseño implican una cantidad significativa de información técnica. Para mantener la claridad y fluidez en la presentación del sistema, se ha optado por incluir todos los detalles específicos de las tablas en un apéndice al final del documento *Apéndice A*.

4.2.4 Integridad de datos

En el sistema desarrollado, se han implementado varias medidas para garantizar que la información almacenada cumpla con los estándares de integridad, evitando así errores y anomalías que puedan comprometer la calidad de los datos.

4.2.4.1 Integridad de entidad

La integridad de entidad se asegura mediante el uso de claves primarias en todas las tablas. Estas claves garantizan que cada fila de una tabla sea única y fácilmente

identificable. A continuación, se mencionan algunos ejemplos, pudiéndose ver con mayor detalle en el *Apéndice A*:

- Tabla usuarios: la clave primaria es "id".
- Tabla cursos: la clave primaria es "id".

4.2.4.2 Integridad referencial

La integridad referencial se garantiza a través de las claves foráneas que aseguran que las relaciones entre tablas se mantengan consistentes. Esto evita la inconsistencia en las relaciones entre tablas. A continuación, se mencionan algunos ejemplos, pudiéndose ver con mayor detalle en el *Apéndice A*:

- Tabla compras: como claves foráneas tiene "usuario" y "curso", que están relacionadas con las tablas usuarios y cursos, respectivamente. Esto asegura que cada compra registrada esté vinculada a un usuario existente y a un curso válido.
- Tabla curso_profesor: como claves foráneas tiene "curso" y "profesor", que están relacionadas con las tablas cursos y usuarios, respectivamente. De este modo, cada relación entre un curso y un profesor se basa en registros válidos y existentes.

4.2.4.3 Integridad de dominio

La integridad de dominio se garantiza mediante la definición de tipos de datos y restricciones para cada columna, asegurando que los valores almacenados sean válidos y estén dentro del rango permitido:

- Enumeraciones en las tablas usuarios con "rol", cursos con "modalidad" y "edad", y en la tabla "mensajes" con "asunto" y "tipo", limitan los valores posibles a un conjunto predefinido, asegurando que solo se introduzcan valores válidos.
- Restricciones de longitud en columnas como "email" y "nombre" de varias tablas, aseguran que los datos no excedan el tamaño permitido.

- Columnas NOT NULL en varias tablas, como "email" en usuarios y "nombre" en cursos, aseguran que ciertos campos críticos no puedan quedar vacíos.

4.2.4.4 Integridad de claves

La base de datos también garantiza la integridad de claves al definir claves únicas, evitando la duplicación de valores en campos que deben ser únicos, como se puede ver en la tabla usuarios para los campos "email", los cuales aseguran que no existan usuarios diferentes con el mismo correo electrónico o nombre de usuario.

4.2.4.5 Mecanismos de cascada

Para mantener la integridad referencial cuando se realizan operaciones de eliminación o actualización, se han definido restricciones de cascada en las claves foráneas. Esto se puede ver en las tablas como compras, carrito y curso_profesor, con la opción ON DELETE CASCADE y ON UPDATE CASCADE se garantiza que cuando se elimina o actualiza un registro padre, los registros relacionados en tablas dependientes se actualicen o eliminen automáticamente, evitando inconsistencias.

4.3 Actores de la aplicación

Se han identificado varios actores clave que interactúan con el sistema, cada uno con sus roles y funcionalidades específicas. A continuación, se detallan los principales actores involucrados, junto con una descripción de sus roles y cómo interactúan con las diferentes funcionalidades de la aplicación.

4.3.1 Usuarios no registrados

Los usuarios no registrados pueden acceder a la aplicación de forma restringida. Tiene acceso a la página principal, los cursos disponibles, la página de contacto y sobre nosotros, los planes de suscripción y por supuesto, al registro y login para poder iniciar sesión.

Sin embargo, no tienen acceso al campus, carrito, mi cuenta y notificaciones. Por motivos razonables, tampoco pueden valorar un curso, ya que no lo han comprado.

4.3.2 Usuarios registrados

Los usuarios registrados pueden acceder a toda la aplicación con diferentes particularidades en función del tipo de usuario. Estas particularidades se mencionan a continuación

4.3.2.1 Usuario normal (normal)

Los usuarios normales pueden comprar añadir cursos al carrito, dentro del carrito comprarlos y tras esto, se añaden al campus. Al acceder al campus pueden ver únicamente los cursos comprados. Dentro de estos cursos pueden ver los archivos subidos por el profesor. En la página de mis datos pueden ver y editar la información del usuario, ver las valoraciones y compras realizadas y

4.3.2.2 Usuario profesor (medium)

Los usuarios profesor pueden también realizar las funciones de un usuario normal. Las diferencias radican en que además pueden crear cursos, modificar o eliminarlos. Dentro de estos pueden subir archivos y cambiar la vista para ver cómo verían el curso los alumnos. Desde la página principal del campus, podrán elegir si quiere ver los cursos en los que es profesor, o si, por el contrario, quiere ver los cursos que ha comprado en los que sería alumno.

4.3.2.3 Usuario profesor limitado (limited)

Los usuarios profesor limitado pueden realizar las funciones de un usuario profesor, con la diferencia de que no pueden gestionar cursos, es decir, crear, modificar o eliminar cursos, sino que solo pueden gestionar los archivos de los cursos en los que es profesor, es decir, crear, modificar y eliminar archivos.

4.3.2.4 Usuario academia (premium)

Los usuarios academia pueden realizar las funciones de un usuario profesor. A estas se añaden que pueden crear usuarios tipo profesor limitado y asignarles un curso de la academia.

4.3.2.5 Usuario administrador (admin)

Los usuarios administradores pueden visualizar la página en su completo (páginas de inicio, mensajes, notificaciones, tienda de cursos…) sin poder ver las páginas del carrito ni campus, ya que no van a poder comprar cursos ni estar asignados a estos. Este tipo de usuario cuenta con un espacio especial donde pueden crear, modificar o eliminar usuarios, cursos y compras.

4.4 Funcionalidades de la aplicación

En este apartado se detallarán las principales funcionalidades que ofrece la aplicación, describiendo cómo los usuarios pueden interactuar con el sistema de acuerdo con su tipo de rol y los permisos que se les otorgan. A continuación, se presentan con detalle las funcionalidades más importantes, como el proceso de registro y login, la gestión de compras, la navegación en el campus y las interacciones entre usuarios mediante notificaciones y valoraciones. Además, se ilustrarán las funcionalidades con ejemplos de vistas de la propia aplicación. Es importante destacar que, durante el desarrollo de las pantallas, se ha prestado especial atención al diseño responsive, garantizando una correcta visualización en dispositivos de diferentes tamaños de ordenador.

4.4.1 Login y Registro

El registro permite a los usuarios crear una cuenta. Para ello deben rellenar un formulario con los datos correspondientes siendo el email y la contraseña los más importantes. El email debe ser único en la base de datos, es decir, no se puede repetir. Para garantizar la seguridad de la contraseña se ha usado bcrypt, que sirve para almacenarla en la base de datos de forma cifrada, y así garantizar la privacidad de datos sensibles. Además, los usuarios deben aceptar la casilla de Términos y Condiciones para poder terminar la acción de registro.

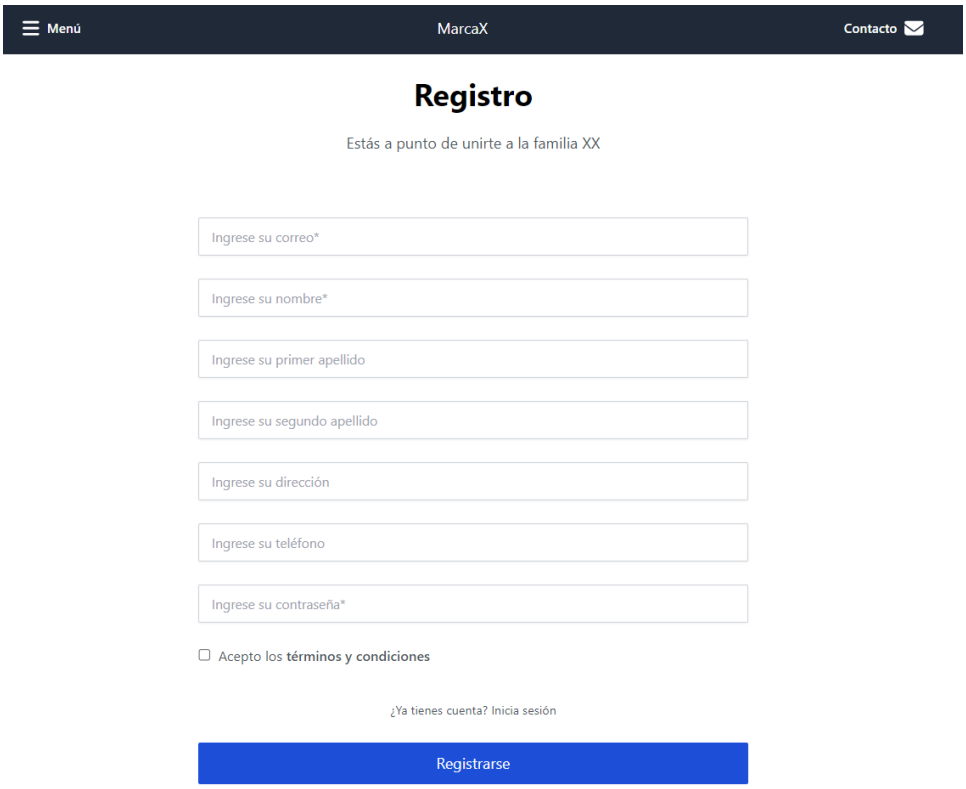


Ilustración 7. Pantalla de registro

Para realizar el login, el usuario introduce el email y la contraseña. Se verifica la existencia del usuario y que la contraseña desencriptada de la base de datos es correcta. Si los datos son correctos se completa el inicio de sesión. Al iniciar sesión, se crea una cookie que contiene la información del usuario. Esta cookie se firma con jsonwebtoken (JWT), dura un día y está disponible para toda la aplicación siempre que la sesión se mantenga activa. Cuando hace logout la cookie se elimina.

Ilustración 8. Pantalla de login

Para mantener la sesión iniciada en toda la aplicación se ha utilizado un contexto creado con createContext. Este contexto recoge la cookie para verificar si el usuario está autenticado, y decodifica el token JWT (la cookie) utilizando jwt-decode. El motivo detrás de este proceso es ajustar la visibilidad y funcionalidad de la aplicación en función del tipo de usuario que ha iniciado sesión.

4.4.2 Comprar cursos

Todo tipo de usuario puede ver los cursos disponibles en la pantalla de cursos, Tienda. Dentro de la página, se han añadido filtros en la parte superior de la pantalla para poder seleccionar los cursos por edades (Todas las edades, ESO, Bachillerato y, FP y Universidad). Además, se ha añadido un buscador para poder buscar palabras claves teniendo en cuenta el filtro de edad que se tenga elegido.

Con respecto a los cursos, si se trata de un usuario no registrado, únicamente puede observarlos y entrar a ver más detalle el curso ofertado. Cuando se trate de un usuario registrado, excepto los usuarios administradores, academias y profesores limitados, podrán pulsar el botón de añadir al carrito. Este botón se encontrará desde la página de Tienda y desde la propia página del curso, y cambia dependiendo de varias opciones:

- Si el usuario ya tiene comprado el curso o es profesor de este, el botón cambiará y al pulsarlo, podrá redirigir a la página del curso dentro del Campus, viéndolo como usuario alumno o como usuario profesor, respectivamente.
- Si es que la suscripción del curso está pronto de caducar (entre 1 a 3 días antes de su caducidad), este cambiará y al pulsarlo, podrá redirigir a la página de del carrito, añadiendo el curso permitiendo renovar la suscripción.
- Si el usuario es academia y es dueño del curso en cuestión, podrá acceder, igual que el usuario profesor, al curso dentro del Campus como rol academia.

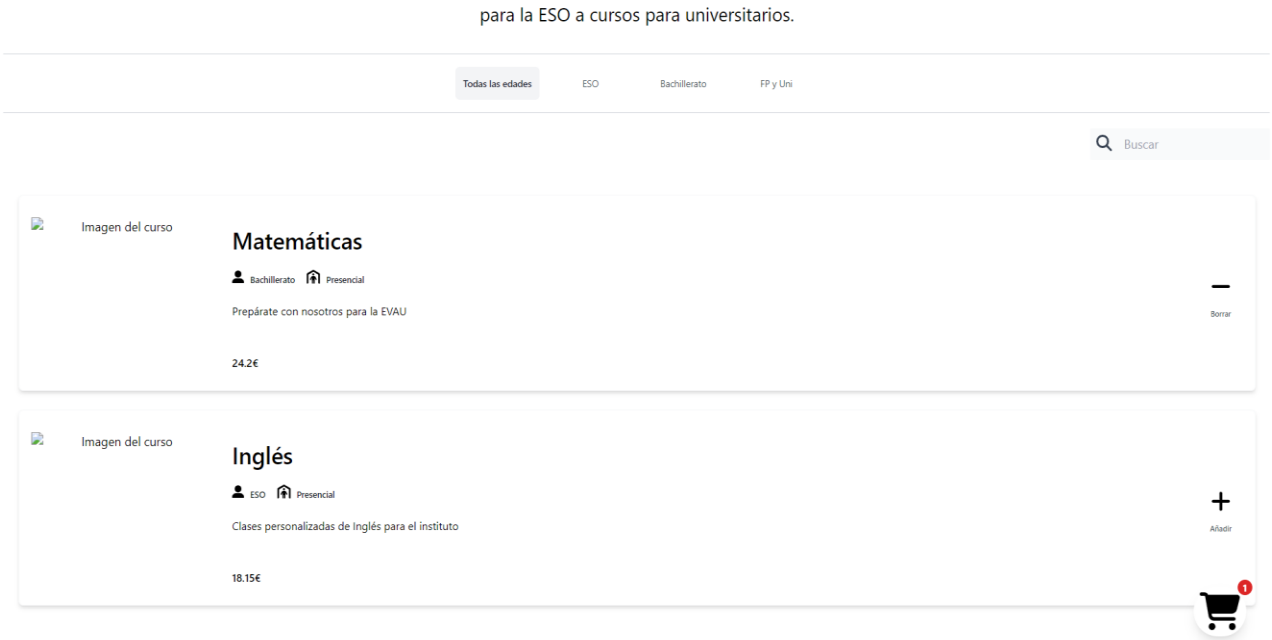


Ilustración 9 Pantalla de cursos

Tras pulsar el botón de añadir al carrito, se puede observar cómo el icono de carrito ubicado abajo a la derecha de la pantalla incrementa en 1 unidad. Además, desde esta pantalla también se puede quitar el curso del carrito. Al pulsar el icono y acceder al carrito puedes ver los cursos añadidos, quitarlos o comprarlos. Se podrán eliminar los cursos uno por uno, pulsando en el icono de flecha a la derecha del curso, o pulsando el botón Vaciar carrito, eliminando todos los cursos del carrito. En el caso de que se quite algún curso del carrito se disminuye el contador de artículos de carrito. En el caso de que se compren, se borra del carrito y de forma automática se tendrán disponibles en el campus los cursos comprados.

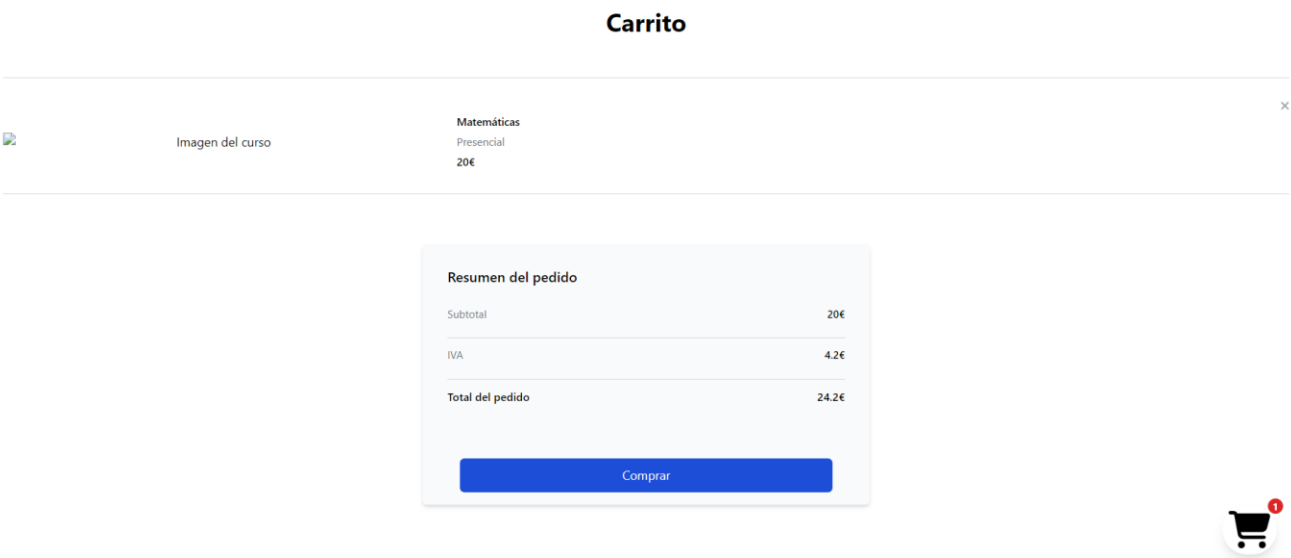


Ilustración 10. Pantalla de carrito

4.4.3 Campus

El campus funciona de forma distinta para los diferentes tipos de usuario. Para acceder a la página se debe pulsar el icono de la parte superior derecha de la pantalla o haciendo clic en "Campus" en el footer de la página.

4.4.3.1 Página principal de campus

- **Usuarios sin sesión y administradores**: no existe.
- **Usuarios normales**: en la página principal les van a aparecer los cursos comprados.
- **Usuarios profesores limitados**: podrán elegir entre ver el listado de cursos asignados por ser usuarios profesores.
- **Usuarios profesores**: Además de tener la opción de ver los cursos comprados, excepto las academias, y los que gestionan mediante filtros en la parte superior de la pantalla, pueden crear cursos tras completar un formulario en el que se introducen los datos correspondientes.

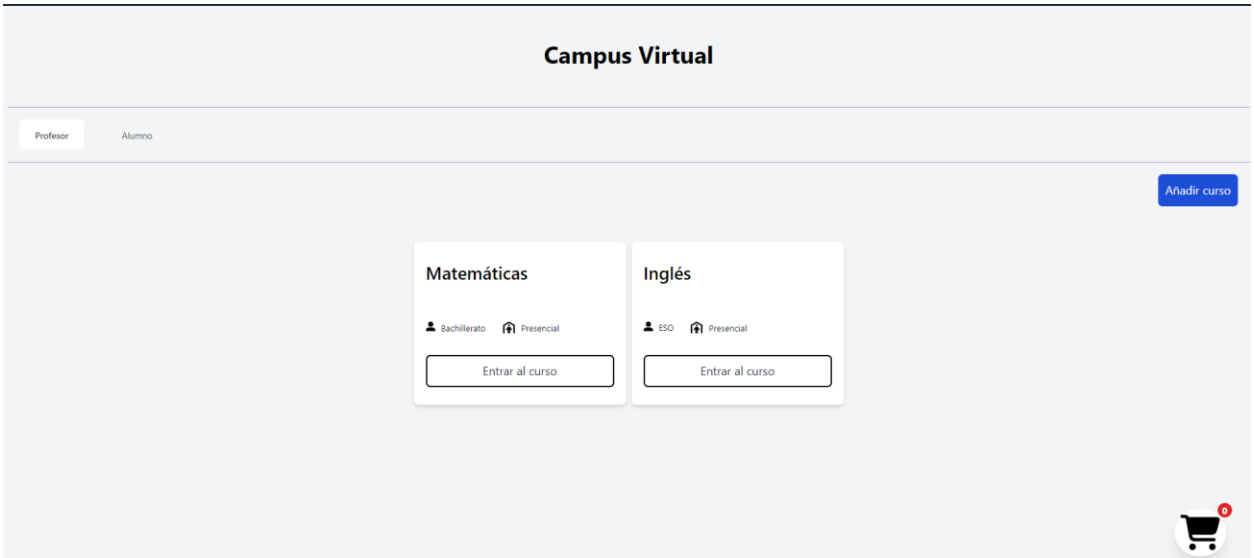
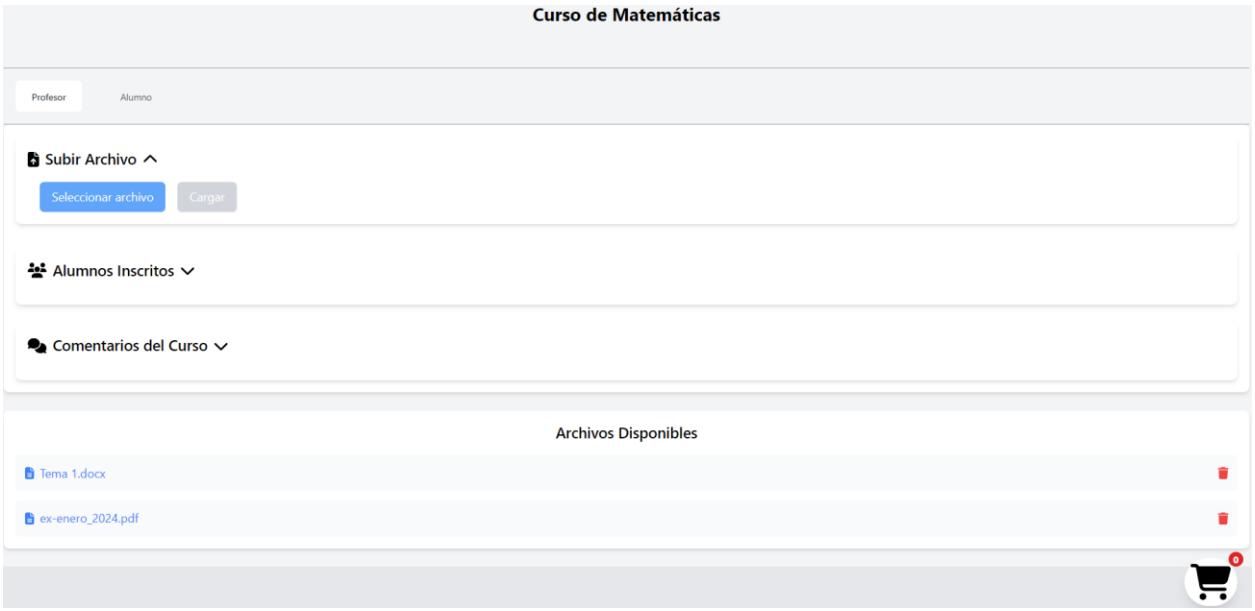


Ilustración 11. Pantalla campus

4.4.3.2 Página dentro de curso

Cabe destacar que se comprueba que por http no se puede acceder a un curso no comprado, o acceder con otro tipo de rol de usuario que no sea el propio según la sesión activa. En otras palabras, la funcionalidad permitida se restringe a la del rol que corresponde según la base de datos.

- **Usuarios normales con curso comprado:** pueden ver la información general del curso, los archivos subidos, la valoración y los comentarios de los alumnos del curso. Además, si es que no han valorado el curso comprado, podrán hacerlo pulsando en el botón Añadir valoración y rellenando los datos correspondientes.
- **Resto de usuarios (profesores y academia)**: Además de poder ver la descripción general del curso, los archivos subidos y los comentarios, tienen la posibilidad de ver los alumnos suscritos y, subir y eliminar archivos. Los tipos de archivos permitidos que se pueden subir actualmente son videos, word, pdf y excel. En esta página, también se puede cambiar la apariencia y ver el curso para comprobar la perspectiva de un alumno.



4.4.4 Notificaciones

Dentro de la aplicación se cuenta con un apartado para las notificaciones de los usuarios registrados dentro del sistema. Se podrá acceder a estos pulsando un botón desde el header o desde el footer.

Dentro de la página Notificaciones, se cuenta con filtros de selección en la parte superior de la pantalla, pudiendo elegir entre poder ver todas las notificaciones, las notificaciones leídas o las pendientes por leer.

Existen distintos tipos de notificaciones, provenientes del sistema, por acción del propio usuario o por un segundo usuario. Todo tipo de notificación cuenta con un enlace que redirige a la página a la que se refiera la notificación.

4.4.4.1 Notificaciones del sistema

Con respecto al primer tipo de notificaciones, para que se creen automáticamente, se ha utilizado un archivo .cron, con el cual, el sistema revisará todos los días a las 00:00 horas, las suscripciones a cursos que estén prontas de caducar, enviando una notificación por curso cuando le queden 3 días por caducar, 2 días y 1 día. Además, si es que los usuarios no renuevan sus compras, estas se eliminarán de manera automática.

4.4.4.2 Notificaciones por acción del propio usuario

Con respecto a las notificaciones por acción del propio usuario, podemos encontrar las siguientes causas:

- Compras: se enviará una notificación a los usuarios que compren un nuevo curso, informándoles que su nuevo curso se encuentra en su Campus. Al pulsar en la notificación se le redirigirá a la página del Campus del curso en cuestión.
- Renovación de compra: se enviará una notificación a los usuarios que, habiéndoles quedado entre 1 y 3 días para que se caduque su suscripción a un curso, renueven su compra, informándoles de la acción y pudiendo pulsar el en la notificación e ir a la página del curso correspondiente en el Campus.
- Enviar mensajes: se enviará una notificación a los usuarios registrados que envíen mensajes a los administradores o a los profesores, informándoles que su mensaje ha sido enviado y guardado en su bandeja de mensajes enviados. Al pulsar en la notificación se le redirigirá a la página Mensajes.

4.4.4.3 Notificaciones por acción de otro usuario

Con respecto a las notificaciones por acción de otro usuario, podemos encontrar las siguientes:

- Recibir mensajes: se enviará una notificación a los usuarios de tipo profesor o academia si es que se ponen en contacto con ellos, informándoles que han recibido un nuevo mensaje. Al pulsar en la notificación se le redirigirá a la página Mensajes.

4.4.5 Mensajes

La aplicación cuenta con un apartado para poder enviar mensajes desde el formulario de contacto, el cual se encuentra en diferentes apartados de la web.

Por una parte, para los usuarios no registrados, el formulario de contacto, accesible desde el header y el footer, enviará mensajes de tipo RECIBIDO a los administradores del sistema.

Por otra parte, para los usuarios registrados existen 2 tipos de mensajes, de tipo RECIBIDO y ENVIADO, dependiendo del rol que tenga el usuario, tendrá de uno o de ambos tipos:

- Usuarios normales: podrán acceder a dos tipos de formularios de contacto.
    - El primero, desde el footer y desde la página miCuenta en el apartado Contacto, podrán enviar mensajes a los administradores del sistema. Además, se les guardará una copia del mensaje de tipo ENVIADO y asunto EMPRESA en la base de datos y podrán acceder a ellos desde el apartado Mensajes de la página miCuenta.
    - El segundo, desde la página del curso del Campus de los que no cuentan con una suscripción activa. Desde esa página podrán enviar a los profesores sobre dudas del curso antes de suscribirse a él. Se guardará una copia del mensaje de tipo ENVIADO y asunto CURSO.
    - El tercero, desde la propia página del curso del Campus o desde la página del curso de la Tienda, podrán enviar mensajes a los profesores de los cursos con suscripción activa con dudas, guardándose una copia en su página de mensajes como ENVIADO y asunto ALUMNO.
- Usuarios profesor: podrán acceder a los mismos tipos de mensajes que un usuario normal, con la diferencia de contar con un nuevo tipo de mensaje, RECIBIDO, proveniente de los alumnos de sus cursos, con asunto ALUMNO, o de usuarios con el curso no comprado, con asunto CURSO.
- Usuarios administradores: este tipo de usuario solo contará con un tipo de mensaje, RECIBIDO, proveniente de los usuarios con dudas sobre la empresa, con asunto EMPRESA.

Para los usuarios de tipo profesor, se contará con filtros para poder ver sus mensajes, pudiendo elegir entre ver los ENVIADOS o los RECIBIDOS.

4.4.6 Valoraciones

La plataforma incorpora la posibilidad de valorar un curso. La valoración únicamente se permite a los usuarios que han comprado ese curso, es decir, los usuarios

que pueden proporcionar retroalimentación a otros usuarios potenciales sobre la calidad y el contenido de los cursos

Para valorar un curso, se podrá hacer desde la página del curso específico a través de la Tienda, apareciendo un botón "Añadir valoración" en caso de que no se haya valorado previamente, si no es el caso, no aparecerá el botón, ya que solo se permite valorar una única vez y sin poder cambiarlo a posteriori.

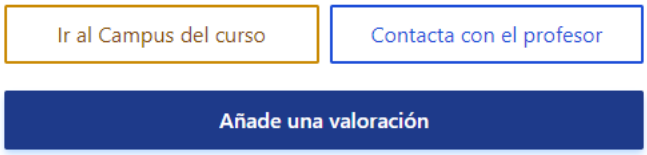


Ilustración 12 Botón valoración

También se cuenta con un botón en la página del curso dentro del Campus, el cual tiene el mismo comportamiento antes descrito, aparece si es que no se ha valorado el curso en cuestión.

Una vez que se pulsa en el botón "Añadir valoración", se redirige a una nueva página "ValoracionPage" desde donde sale el formulario de la valoración, teniendo que rellenar las estrellas (de 1 a 5) de forma obligatoria, ya que el comentario es opcional. Para terminar la acción, se puede pulsar el botón "Enviar valoración", el cual registra la valoración, o pulsar el botón "Cancelar", el cual descarta los datos rellenados. En cualquiera de las dos alternativas, se vuelve a la página de la que se redirigió, es decir, a la página del curso de la Tienda o a la del curso del Campus.

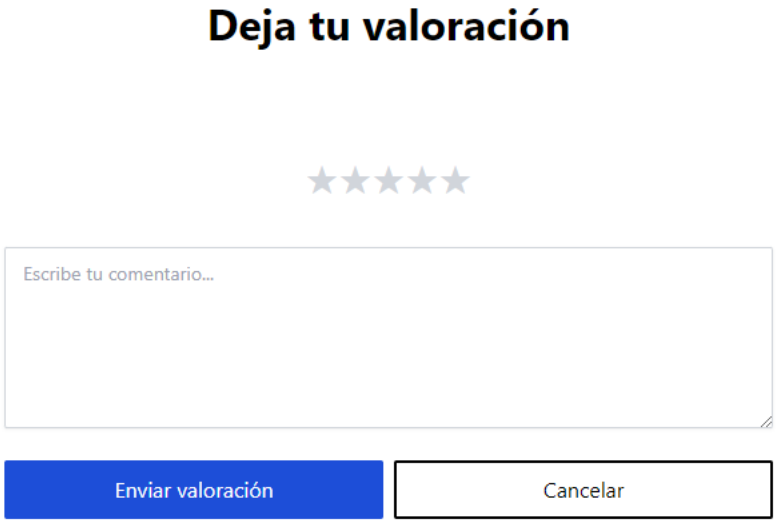


Ilustración 13 Formulario valoración

Si es que se ha registrado la valoración, se actualiza, tanto en la página del curso de la Tienda como del Campus, el promedio de valoraciones (estrellas). Con respecto al apartado de comentarios de ambas páginas, Tienda y Campus, si es que el usuario al rellenar el formulario añadió un comentario, este se mostrará, si es que no fue añadido el comentario, saldrá un texto informando que aún no hay comentarios sobre el curso.

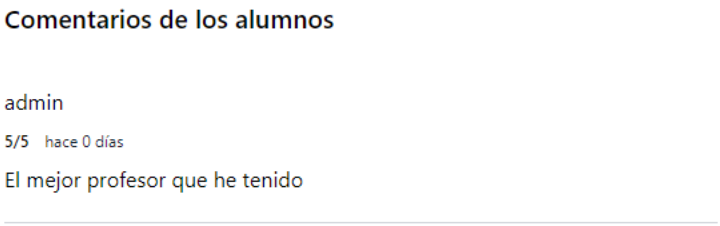


Ilustración 14 Valoración con comentario

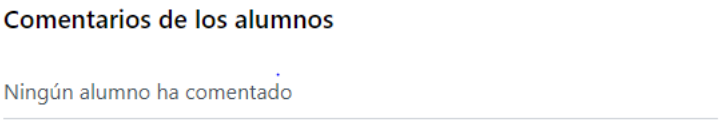


Ilustración 15 Valoración sin comentario

Con respecto a la media de las valoraciones, aparecerá como sin valoraciones si es que el curso no ha obtenido valoraciones, mientras que, si se ha valorado por otros usuarios, aparecerá la media de estos.

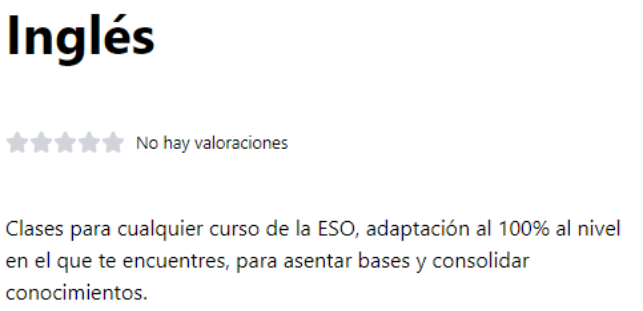


Ilustración 16 Curso sin valorar

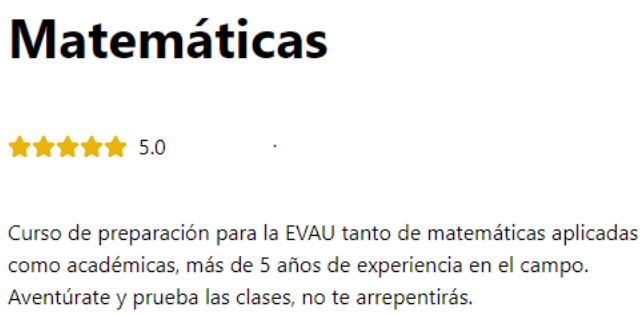


Ilustración 17 Curso valorado

4.4.7 Suscripciones

La aplicación cuenta con 3 tipos de suscripciones, las cuales se pueden ver en la página Planes de suscripción, pudiendo acceder a él desde el footer. En especial, los usuarios registrados podrán acceder a su plan de suscripción específica desde su cuenta. Además de las características estándar de cada plan, el cual se ha explicado en el apartado 4.3 Actores de la aplicación, existen condiciones específicas en cuanto a la cancelación de suscripciones y el manejo de los usuarios asociados, especialmente en el caso del plan Profesor Limited, que tiene particularidades adicionales en cuanto a la gestión y el acceso a los cursos.

Para el contrato de las suscripciones, se ha implementado el usuario normal y profesor. Sin embargo, las funcionalidades de todos los actores dentro de la aplicación están operativos.

4.4.7.1 Plan gratuito (usuario normal)

Este plan no tiene suscripción asociada, por lo que los usuarios pueden continuar usando la plataforma sin restricciones de tiempo. Los usuarios gratuitos pueden seguir

comprando cursos de forma individual, y no hay condiciones de cancelación asociadas, ya que no se involucra ninguna tarifa o suscripción recurrente.

4.4.7.2 Plan limited (usuario profesor limited)

El plan Limited está dirigido a aquellos profesores que solo gestionan cursos bajo la supervisión de una academia. Este tipo de usuario no puede comprar cursos ni crear nuevos; su acceso está limitado a gestionar los cursos asignados por la academia.

- Cancelación del plan de academia: si una academia decide cancelar su suscripción, se notificará a los profesores Limited asociados. Estos profesores tendrán que cumplir con sus obligaciones hasta que finalicen las suscripciones activas de los estudiantes en los cursos que gestionan.
    - Durante este período de notificación, los profesores podrán seguir gestionando sus cursos normalmente.
    - Desactivación de cuentas: una vez finalizadas todas las suscripciones activas de los estudiantes, las cuentas de los profesores Limited serán eliminadas, ya que no tienen la capacidad de adquirir cursos de forma independiente ni de continuar con sus actividades sin la academia. Se desactivarán sus claves de acceso y ya no podrán ingresar a la plataforma.

4.4.7.3 Plan medium (usuario profesor)

En el caso del plan Medium (pago único de 6 meses), la cancelación de la suscripción afectará la capacidad del usuario para crear o gestionar nuevos cursos. Sin embargo, los cursos ya creados permanecerán en la plataforma y seguirán siendo accesibles para los estudiantes que los hayan adquirido, pero sin posibilidad de renovar la suscripción al curso, después del cual, se eliminará el curso permanentemente de la Tienda y la base de datos. El profesor perderá la capacidad de gestionar sus cursos y subir nuevo contenido, pero no se eliminarán los cursos ya comprados, funcionabilidad de usuario normal. La suscripción puede ser reactivada en cualquier momento para recuperar estas capacidades.

4.4.7.4 Plan premium (usuario academia)

El plan Premium permite la administración de múltiples profesores y cursos. En caso de que una academia decida cancelar su suscripción, deberán seguirse ciertas condiciones particulares:

- Notificación y transición: la academia y los profesores Limited serán notificados de la cancelación, y se les informará que deben cumplir con sus compromisos hasta que finalicen las suscripciones activas de los estudiantes. Durante este período, la academia podrá seguir gestionando sus cursos y profesores.
- Eliminación de cuentas de profesores Limited: tras el vencimiento de las suscripciones de los estudiantes, los profesores Limited asociados a la academia perderán su acceso y sus cuentas serán eliminadas, ya que su actividad dependía de la relación directa con la academia.
- Responsabilidad sobre el contenido: los cursos creados por la academia seguirán disponibles para los estudiantes que los compraron antes de la cancelación. Sin embargo, la academia perderá la capacidad de gestionar estos cursos o realizar cambios, a menos que reactive su suscripción.

4.4.7.5 Políticas de cancelación generales

En todos los planes de suscripción, la relación implica la interrupción de los beneficios asociados, pero no afecta al acceso a los cursos previamente adquiridos. La cancelación del plan puede resultar en la eliminación de cuentas de usuarios, con las notificaciones correspondientes y el cumplimiento de obligaciones hasta finalizar los compromisos adquiridos.

Capítulo 5 - Conclusiones y trabajo futuro

5.1 Conclusiones

El desarrollo de esta aplicación ha sido un proceso enriquecedor que ha permitido combinar diversas tecnologías y metodologías para crear una plataforma funcional orientada a la compra y gestión de cursos online. A lo largo de este proyecto, se han abordado desafíos tanto técnicos como de diseño, con el objetivo de ofrecer una solución que responda a las necesidades del usuario.

En términos generales, este proyecto ha cumplido con los objetivos iniciales planteados, proporcionando una plataforma eficaz para la gestión de cursos y usuarios, con una experiencia de usuario satisfactoria. No obstante, se reconoce que el sector tecnológico está en constante evolución, y la aplicación deberá adaptarse a las nuevas necesidades y exigencias del mercado para seguir siendo competitiva y relevante.

5.2 Trabajo Futuro

A pesar de que la aplicación es plenamente funcional, hay varias áreas que pueden mejorarse y expandirse en el futuro. Estas mejoras están orientadas a optimizar la experiencia del usuario, mejorar la escalabilidad, la seguridad y añadir funcionalidades que puedan ampliar el alcance de la plataforma.

- Implementación de los planes de suscripción: introducir las funcionalidades correspondientes a los cambios y contratación de diversos planes de suscripción y sus funciones .cron correspondientes.
- Añadir nuevas funcionalidades a las academias para crear un espacio más amplio y completo, como contratar a los usuarios profesor limited y controlar la eliminación de los cursos, con sus respectivas notificaciones y acciones.
- Integración de cuentas de terceros: implementar opciones para que los usuarios puedan iniciar sesión mediante servicios de terceros como Google, Facebook o LinkedIn. Esto no solo mejoraría la experiencia de usuario, sino que también simplificaría el proceso de registro y login.

- Foros de discusión: añadir un foro para la interacción entre usuarios, donde los estudiantes puedan realizar preguntas, resolver dudas, o discutir sobre temas relacionados con los cursos. Esto fomentaría una comunidad activa dentro de la plataforma y facilitaría el aprendizaje colaborativo.
- Pasarela de pagos segura: integrar una pasarela de pago como PayPal para facilitar la compra de cursos de manera segura.
- Notificaciones automáticas: incluir un sistema de notificaciones automáticas vía email o SMS para informar a los usuarios sobre actualizaciones en los cursos, recordatorios de pagos, y otros eventos importantes.
- Optimización de consultas a la base de datos: para mejorar el rendimiento de la aplicación, se debería implementar una optimización avanzada en las consultas a la base de datos. Esto incluye el uso de índices, cachés y consultas preparadas para reducir la carga en el servidor.
- Despliegue en la web: actualmente, la aplicación se encuentra alojada localmente. Subirla a un servidor web garantizaría que más usuarios puedan acceder desde cualquier parte.
- Autoevaluaciones y exámenes: implementar una nueva funcionalidad que permita a los profesores crear autoevaluaciones y exámenes dentro de los cursos. Esto ofrecería a los alumnos la oportunidad de medir su progreso de manera periódica, y recibir retroalimentación inmediata sobre sus fortalezas y áreas de mejora. Podrían agregarse diferentes tipos de preguntas, como selección múltiple, verdadero/falso, o respuestas abiertas.
- Plan de Marketing Estratégico: desarrollar un plan de marketing integral para promover la aplicación y aumentar su visibilidad. Este plan debería incluir campañas de publicidad digital en plataformas sociales como Instagram o TikTok, así como en motores de búsqueda como Google. Además, se podrían explorar estrategias de marketing de contenido, como colaboraciones con influencers en el ámbito educativo, para atraer a un público más amplio de manera coste-efectividad. El objetivo es maximizar el alcance de la plataforma sin requerir grandes inversiones, utilizando tácticas de bajo coste y alto impacto.

Estas futuras implementaciones y mejoras permitirán que la aplicación crezca en funcionalidad, mejores la experiencia del usuario y aumente su alcance, haciéndola más competitiva en el mercado de plataformas educativas online.

5.3 Contribuciones personales

5.3.1 Lucero Abregú Reátegui

5.3.1.1 Tablas base de datos

Fui responsable del diseño y la implementación de las Tablas de la Base de Datos para la aplicación. Mis contribuciones incluyeron:

- **Diseño del Esquema de la Base de Datos**: Definí y estructuré el esquema de la base de datos, creando tablas que reflejan los requisitos de la aplicación y soportan las funcionalidades necesarias, como usuarios, cursos, valoraciones y compras.
- **Creación de Tablas**: Implementé las tablas necesarias para gestionar los diferentes aspectos de la aplicación, asegurando que las relaciones entre ellas (por ejemplo, entre cursos y usuarios) se modelaran de manera eficiente.
- **Optimización y Mantenimiento**: Realicé ajustes y optimizaciones para garantizar el rendimiento y la integridad de las tablas, además de asegurarme de que las consultas y operaciones en la base de datos se ejecutaran de manera eficaz.
- **Documentación**: Documenté el diseño de la base de datos y las relaciones entre las tablas para facilitar el mantenimiento y futuras ampliaciones del sistema.

5.3.1.2 Notificaciones y mensajes

Fui responsable del desarrollo e integración del sistema de Notificaciones y Mensajes en la aplicación. Mis contribuciones incluyeron:

- **Diseño del Sistema de Notificaciones**: Definí los requisitos y el flujo para el sistema de notificaciones, permitiendo que los usuarios reciban alertas sobre eventos importantes como nuevas valoraciones, mensajes o actualizaciones de cursos.

- **Implementación de Funcionalidades**: Desarrollé las funcionalidades para enviar y recibir notificaciones, asegurando que se presenten de manera efectiva en la interfaz de usuario y que los usuarios puedan gestionar sus preferencias de notificación.
- **Desarrollo del Sistema de Mensajes**: Implementé la capacidad para enviar y recibir mensajes entre usuarios, permitiendo la comunicación directa a través de la plataforma. Esto incluyó la creación de interfaces para leer y enviar mensajes, así como la gestión de conversaciones.
- **Integración y Pruebas**: Integré el sistema de notificaciones y mensajes con el resto de la aplicación, realizando pruebas para asegurarme de que funcionara correctamente en diferentes escenarios y dispositivos.
- **Documentación y Soporte**: Documenté el sistema de notificaciones y mensajes, proporcionando guías sobre su uso y mantenimiento para los futuros desarrolladores y usuarios.

5.3.1.3 Tienda

Desempeñé un papel importante en el desarrollo y gestión de la funcionalidad de la tienda. Mis contribuciones incluyeron:

- **Diseño y Desarrollo de la Tienda**: Implementé la sección de la tienda en la aplicación, lo que incluía la visualización de productos, gestión de carritos de compra y procesos de compra. Utilicé React para desarrollar la interfaz de usuario, asegurando que los usuarios pudieran navegar, seleccionar y comprar productos de manera eficiente.
- **Integración con el Backend**: Trabajé en la integración de la tienda con el backend utilizando Node.js y Axios. Implementé las funcionalidades necesarias para realizar operaciones CRUD en los productos, gestionar el estado del carrito y procesar pedidos.
- **Gestión del Estado del Carrito**: Implementé la lógica para la gestión del carrito de compras, que incluyó la adición y eliminación de productos, el cálculo de precios totales y la persistencia del carrito entre sesiones de usuario.

- **Procesamiento de Pagos**: Diseñé e implementé la lógica para el procesamiento de pagos, integrando métodos de pago y asegurando que los pedidos se registraran correctamente en la base de datos.
- **Interfaz de Usuario y Experiencia**: Desarrollé una interfaz de usuario intuitiva y atractiva para la tienda, utilizando Tailwind CSS para un diseño responsivo y atractivo. Me aseguré de que la experiencia de compra fuera fluida y libre de errores.
- **Pruebas y Optimización**: Realicé pruebas exhaustivas de la funcionalidad de la tienda para identificar y corregir errores, y optimicé el rendimiento para garantizar tiempos de carga rápidos y una experiencia de usuario satisfactoria.

5.3.2 Pablo Delgado Gómez

5.3.2.1 Decisión y Conocimiento de Tecnologías Usadas

Tuve un papel clave en la selección y aplicación de las tecnologías utilizadas, y en el desarrollo general de la aplicación. Mis contribuciones incluyeron:

- **Evaluación y Selección de Tecnologías**: Investigué y evalué las tecnologías más adecuadas para el proyecto, incluyendo Node.js para el backend y React para el frontend. Tomé decisiones informadas sobre la utilización de estas tecnologías basándome en sus características, ventajas y compatibilidad con los requisitos del proyecto.
- **Desarrollo con Node.js**: Utilicé Node.js para construir el backend de la aplicación, aprovechando su capacidad para manejar múltiples conexiones simultáneas y su ecosistema de módulos. Implementé APIs RESTful para gestionar las operaciones CRUD y la comunicación entre el frontend y la base de datos.
- **Desarrollo con React**: Desarrollé la interfaz de usuario utilizando React, aprovechando su arquitectura basada en componentes para construir una interfaz interactiva y dinámica. Implementé el enrutamiento, el manejo del estado y la integración con el backend para una experiencia de usuario fluida.

- **Uso de Herramientas y Librerías**: Seleccioné e integré herramientas y librerías adicionales, como Axios para las solicitudes HTTP y Tailwind CSS para el diseño. Estas herramientas ayudaron a mejorar la eficiencia del desarrollo y la calidad de la interfaz de usuario.
- **Documentación y Justificación de Elecciones**: Documenté las decisiones tecnológicas y las razones detrás de la selección de herramientas específicas, proporcionando un marco claro para su uso y mantenimiento en el futuro.

5.3.2.2 Registro y login

Este módulo fue clave para gestionar el acceso a la plataforma de manera segura y efectiva. Mi aportación incluyó:

- **Registro de usuarios**: Creé el flujo de registro, que incluye la validación de los datos de los usuarios (nombre, correo electrónico y contraseña), asegurando que los datos ingresados sean correctos y estén protegidos.
- **Encriptación de contraseñas**: Implementé la encriptación de contraseñas usando tecnologías adecuadas para garantizar la seguridad de los datos sensibles de los usuarios.
- **Login y gestión de sesiones**: Diseñé el sistema de inicio de sesión que permite a los usuarios autenticarse con sus credenciales y mantiene las sesiones activas de forma segura a través de JSON Web Tokens (JWT).

5.3.2.3 Campus

Se trata de una sección clave de la plataforma donde los usuarios pueden acceder y gestionar sus cursos. Mis contribuciones principales incluyeron:

- **Interfaz del Campus**: Desarrollé la estructura y diseño para mostrar la lista de cursos disponibles, separando las diferentes vistas en función del tipo de usuario.
- **Gestión de Cursos**: Implementé la funcionalidad que permite a los usuarios visualizar el contenido de sus cursos inscritos, además de acceder a información detallada sobre cada curso, como descripciones, precios y modalidades.

- **Interacción con la Base de Datos**: Colaboré en la creación de consultas eficientes para recuperar los cursos asociados a cada usuario, garantizando la correcta sincronización entre la interfaz y la base de datos.

5.3.2.4 Valoraciones

Permite a los usuarios evaluar y comentar sobre los cursos que han completado. Mis responsabilidades incluyeron:

- **Desarrollo del Sistema de Valoraciones**: Implementé el módulo para que los usuarios pudieran dejar calificaciones y comentarios sobre los cursos. Esto incluyó la creación de formularios para enviar valoraciones y un sistema para mostrar reseñas en la interfaz de usuario.
- **Integración con la Base de Datos**: Diseñé y optimicé las consultas necesarias para almacenar y recuperar valoraciones y comentarios desde la base de datos, asegurando la integridad de los datos y una respuesta rápida del sistema.
- **Interfaz de Usuario**: Trabajé en el diseño de la interfaz para la visualización de valoraciones y comentarios, proporcionando una experiencia intuitiva y atractiva para los usuarios.

Además hemos sido ambos responsables de solucionar errores y resto de funcionalidades y código.

Conclusions and future work

The development of this application has been an enriching process that allowed us to combine various technologies and methodologies to create a functional platform aimed at the purchase and management of online courses. Throughout this project, both technical and design challenges have been addressed with the goal of providing a solution that meets user needs.

Overall, this project has achieved its initial objectives, delivering an effective platform for managing courses and users with a satisfactory user experience. However, it is recognized that the technology sector is constantly evolving, and the application will need to adapt to new market needs and demands to remain competitive and relevant.

Future Work

Although the application is fully functional, there are several areas that can be improved and expanded in the future. These improvements are aimed at optimizing the user experience, enhancing scalability, increasing security, and adding features that can broaden the platform's reach.

- Third-Party Account Integration: Implement options for users to log in using third-party services such as Google, Facebook, or LinkedIn. This would not only enhance the user experience but also simplify the registration and login process.
- Discussion Forums: Add a forum for user interaction where students can ask questions, resolve doubts, or discuss topics related to the courses. This would foster an active community within the platform and facilitate collaborative learning.
- Secure Payment Gateway: Integrate a payment gateway like PayPal to facilitate secure course purchases.
- Automated Notifications: Include an automated notification system via email or SMS to inform users about course updates, payment reminders, and other important events.

- Database Query Optimization: To improve application performance, advanced optimization of database queries should be implemented. This includes using indexes, caching, and prepared queries to reduce server load.
- Web Deployment: Currently, the application is hosted locally. Deploying it on a web server would ensure that more users can access it from anywhere.
- Self-Assessments and Exams: Implement a new feature that allows instructors to create self-assessments and exams within the courses. This would give students the opportunity to periodically measure their progress and receive immediate feedback on their strengths and areas for improvement. Different question types could be added, such as multiple choice, true/false, or open-ended questions.
- Strategic Marketing Plan: Develop a comprehensive marketing plan to promote the application and increase its visibility. This plan should include digital advertising campaigns on social media platforms like Instagram or TikTok, as well as on search engines like Google. Additionally, content marketing strategies could be explored, such as collaborations with educational influencers, to attract a broader audience in a cost-effective manner. The goal is to maximize the platform's reach without requiring significant investments, using low-cost and high-impact tactics.

These future implementations and improvements will allow the application to grow in functionality, enhance the user experience, and expand its reach, making it more competitive in the online educational platform market.

BIBLIOGRAFÍA

Axios. (s.f.). Obtenido de https://axios-http.com/es/

Expressjs. (s.f.). Obtenido de https://expressjs.com/

Font Awesome. (s.f.). Obtenido de https://fontawesome.com/

Gamarra, M. (7 de Noviembre de 2023). *INESEM*. Recuperado el 2 de Abril de 2024, de https://www.inesem.es/revistadigital/educacion-sociedad/herramientas-digitales-educativas/

Github. (s.f.). Obtenido de https://github.com/

Legacy. (s.f.). Recuperado el 2 de Mayo de 2024, de https://legacy.reactjs.org/

Madrid, U. C. (s.f.). Recuperado el 23 de Mayo de 2024, de https://www.ucm.es/campusvirtual/pacv#:~:text=PACV%2DPortal%20de%20Acceso%20al%20Campus%20Virtual&text=Campus%20Virtual%20%2D%20UCM

Nodejs. (s.f.). Obtenido de https://nodejs.org/en

phpMyAdmin. (s.f.). Obtenido de https://www.phpmyadmin.net/

React. (s.f.). Recuperado el 2 de Junio de 2024, de https://es.react.dev

React Router. (s.f.). Obtenido de https://reactrouter.com/en/main

Scribd. (s.f.). Recuperado el 28 de Mayo de 2024, de https://es.scribd.com/about

Tailwind CSS. (s.f.). Obtenido de https://tailwindcss.com/

Visual Studio Code. (s.f.). Obtenido de https://code.visualstudio.com/

Wuolah. (s.f.). Recuperado el 16 de Abril de 2024, de https://help.wuolah.com/docs/es/360002097179

APÉNDICES

Apéndice A - Estructura de las tablas de la base de datos

En esta sección se incluye una descripción más detallada de la base de datos, contemplando sus tablas y el detalle de las relaciones existentes entre ellas, especificando en cada caso las claves primarias y foráneas utilizadas.

En los atributos como claves foráneas de las tablas se han añadido acciones de actualización (ON UPDATE CASCADE) y acciones de eliminación (ON DELETE CASCADE).

Tabla usuarios

La tabla usuarios almacena la información principal de todos los usuarios del sistema. El atributo "id" actúa como la clave primaria (PRIMARY KEY) de la tabla y es un entero que se auto incrementa, lo que garantiza que cada usuario tenga un identificador único y no nulo.

El atributo "email" cuenta con un índice único (UNIQUE), asegurando que no puedan existir dos usuarios con el mismo correo electrónico en la base de datos.

El campo "rol" almacena el tipo de usuario (administrador, premium, médium, limited o normal), utiliza un tipo de dato ENUM, lo que garantiza que solo se pueden asignar valores específicos predefinidos, manteniendo la consistencia en los roles de los usuarios.

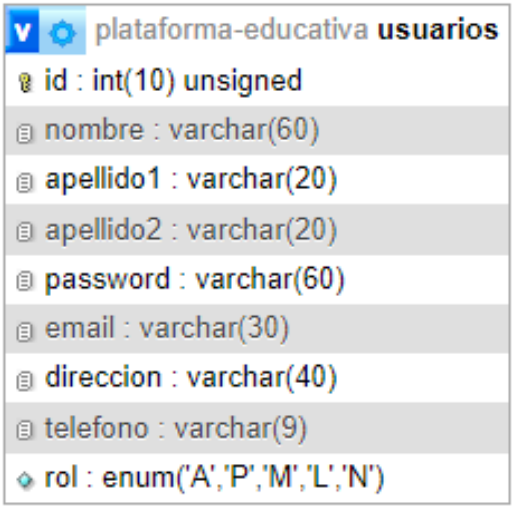


Ilustración 18 Tabla usuarios

Tabla notificaciones

La tabla notificaciones está diseñada para almacenar los mensajes del sistema a los propios usuarios. El atributo "id" es la clave primaria de la tabla, definido como un entero que se auto incrementa.

El atributo "usuario" es una clave foránea (FOREIGN KEY) que referencia el campo "id" de la tabla usuarios, lo que asegura que cada notificación esté asociada a un usuario existente en la base de datos.

El atributo "leida", de tipo TINYINT, funciona como un indicador booleano para marcar si la notificación ha sido leída (1) o no (0) por el usuario. El campo "enlace" se utiliza para almacenar un enlace opcional relacionado con la notificación, proporcionando al usuario acceso directa a una página o recurso relevante en la plataforma.

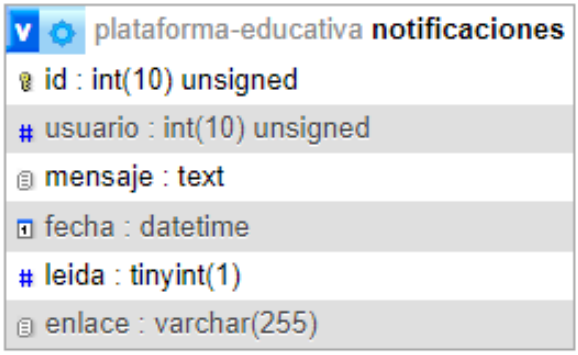


Ilustración 19 Tabla notificaciones

Tabla mensajes

La tabla mensajes está diseñada para gestionar la comunicación entre los usuarios del sistema, almacenando tanto mensajes enviados como recibidos. El atributo "id" actúa como la clave primaria de la tabla, definido como un entero auto incrementable y no nulo.

El campo "usuario" es una clave foránea que hace referencia al campo "id" de la tabla usuarios, asegurando que cada mensaje esté asociado a un usuario existente en el sistema.

El motivo por el que los atributos "nombreOrigen" e "emailOrigen" no son claves foráneas es el que los usuarios no registrados pueden mandar mensajes a los administradores.

El campo "asunto", enumerado que representa el asunto del mensaje, pudiendo tomar valores como EMPRESA, representando el mensaje de un usuario registrado o no registrado, con dudas sobre el funcionamiento de la página, dirigiéndose a los usuarios administradores; CURSO, representando el mensaje de un usuario registrado pidiendo más información sobre un curso, dirigiéndose a los profesores del curso; o ALUMNO, representando al mensaje de un usuario que tenga comprado el curso con dudas sobre este, dirigiéndose al profesor del curso.

Finalmente, el campo "tipo", también de tipo enumerado (ENUM), diferencia entre mensajes enviados (ENVIADO) y recibidos (RECIBIDO). Cuando un usuario registrado envía un mensaje, este se guarda como una copia en su bandeja de

mensajes con el tipo ENVIADO, ya sea que el destinatario sea un administrador o un profesor. Simultáneamente, el mensaje se almacena en la bandeja del destinatario con el tipo RECIBIDO, asegurando que ambos usuarios puedan acceder al historial completo de comunicación. Las respuestas a los mensajes se realizarán por correo electrónico exterior, no por dentro de la propia aplicación.

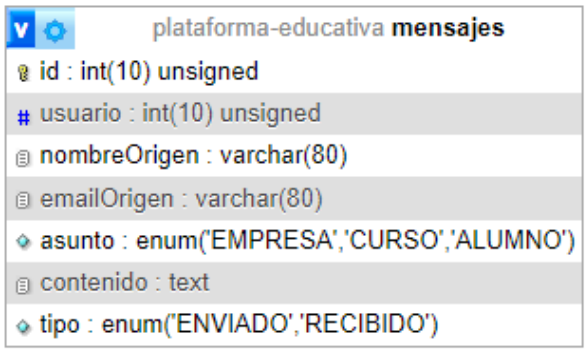


Ilustración 20 Tabla mensajes

Tabla cursos

La tabla cursos almacena la información esencial de cada curso disponible en la plataforma. El atributo "id" es la clave primaria de la tabla y se define como un entero que se auto incrementa y no es nulo.

El campo "modalidad" de tipo enumerado, el cual indica si el curso se imparte de manera presencial (PRESENCIAL), online (ONLINE) o remota (REMOTA), es decir, sin clases, solo material.

El campo "edad", también de tipo enumerado, indica el ámbito estudiantil para el que se imparte el curso, a usuarios de Secundaria (ESO), de Bachillerato (BACH), FP y Universidades (FPU) o para todas las edades (T).

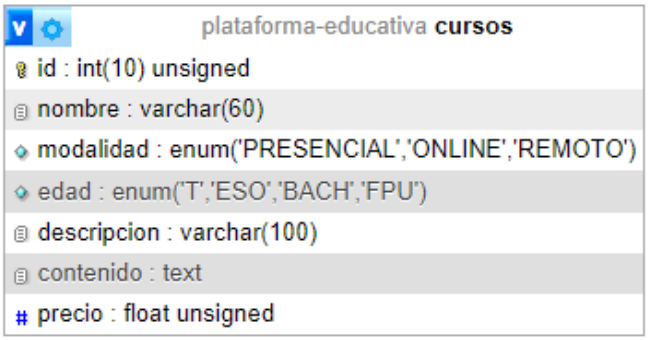


Ilustración 21 Tabla cursos

Tabla archivos

La tabla archivos almacena información sobre los archivos asociados a los cursos en la plataforma. El atributo "id" es la clave primaria de la tabla, definido como un entero que se auto incrementa, garantizando un identificador único y no nulo para cada archivo.

El atributo "cursoId" es una clave foránea que referencia el "id" de la tabla "cursos", lo que establece una relación entre cada archivo y su curso correspondiente.

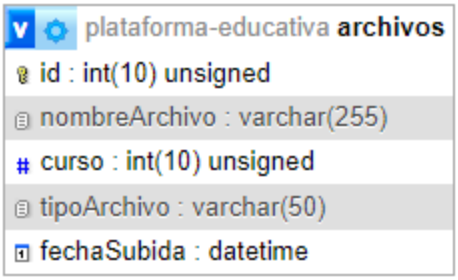


Ilustración 22 Tabla archivos

Tabla carrito

La tabla carrito es utilizada para almacenar la información relacionada con los cursos que un usuario ha agregado a su carrito de compras. Esta tabla permite a los usuarios seleccionar y guardar temporalmente cursos que desean comprar antes de finalizar su compra.

El campo "id" actúa como clave primaria de la tabla, de tipo entero sin signo. Este campo se auto incrementa, asegurando que cada registro en la tabla tenga un identificador único y no nulo.

El campo "usuario" actúa como clave foránea, referencia al campo "id" de la tabla usuarios, utilizado para identificar qué usuario ha agregado el curso al carrito. Mientras que el campo "curso", clave foránea, referencia al campo "id" de la tabla cursos, para identificar el curso agregado al carrito.

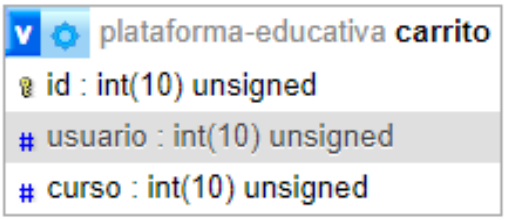


Ilustración 23 Tabla carrito

Tabla compras

La tabla compras se encarga de almacenar la información relacionada con las compras realizadas por los usuarios dentro del sistema. El campo "id" funciona como clave primaria, de tipo entero sin signo y auto incremental.

Contiene como claves foráneas el campo "usuario", haciendo referencia al campo "id" de la tabla usuarios, y el campo "curso", haciendo referencia al campo "id" de la tabla cursos.

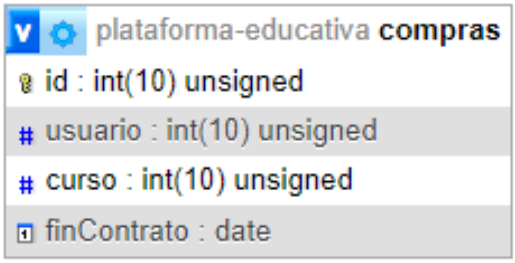


Ilustración 24 Tabla compras

Tabla valoraciones

La tabla valoraciones almacena información sobre las valoraciones que los usuarios han hecho a los cursos de los que estas matriculados en la plataforma. El campo "id" actúa como clave primaria de la tabla y es auto incrementable.

Como claves foráneas se encuentran los atributos "usuario" referenciando el "id" de la tabla usuarios, "curso" referenciando el campo "id" de la tabla cursos.

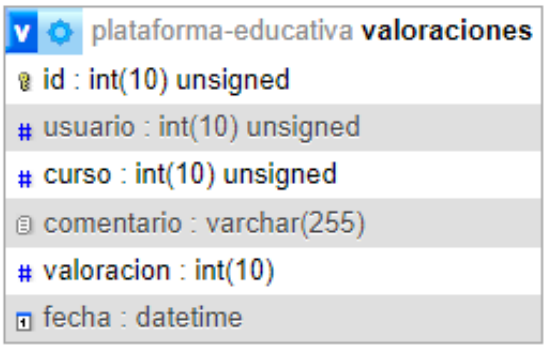


Ilustración 25 Tabla valoraciones

Tabla profesor_academia

La tabla profesor_academia es utilizada para gestionar la relación entre los profesores (usuario limited) y las academias a las que están asociados.

La combinación de los campos "profesor" y "academia" constituye la clave primaria, esto asegura que no se puedan duplicar las combinaciones de profesores y academias, garantizando la unicidad de cada asociación. Esta clave compuesta garantiza que un profesor no pueda estar asociado más de una vez a la misma academia.

Como claves foráneas se tienen al campo "profesor", el cual referencia al campo "id" de la tabla usuarios de tipo limited, y el campo "academia", el cual referencia al campo "id" de la tabla usuarios de tipo premium.

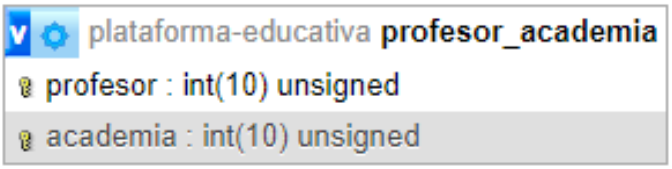


Ilustración 26 Tabla profesor_academia

Tabla curso_profesor

La tabla curso_profesor está diseñada para gestionar la relación entre los cursos y los profesores que los imparten.

La combinación de los campos “curso” y “profesor” constituye la clave primaria. Esto asegura que no se puedan duplicar las combinaciones de cursos y profesores.

Como claves foráneas tenemos el campo “curso” que referencia al campo “id” de la tabla cursos y el campo “profesor” que referencia al campo “id” de la tabla usuarios.

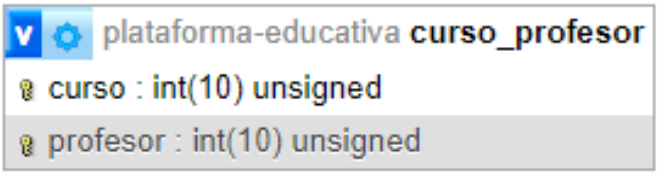


Ilustración 27 Tabla curso_profesor

Apéndice B - Manual de instalación

En esta sección se proporcionarán las instrucciones detalladas para la instalación y configuración del sistema, incluyendo la puesta en marcha de la base de datos. Este manual está diseñado para guiar a los administradores del sistema a través de los pasos necesarios para desplegar el entorno de la aplicación de manera efectiva y eficiente.

Base de datos

La instalación comienza ejecutando el servidor XAMPP. En caso de utilizar un ordenador personal, hay que descargar XAMPP en el siguiente enlace: https://www.apachefriends.org y una vez descargado e instalado se deberá abrir XAMPP CONTROL PANEL. Se deberá ejecutar tanto APACHE como MYSQL para poder conectarse correctamente a la base de datos y que la página web funcione correctamente.

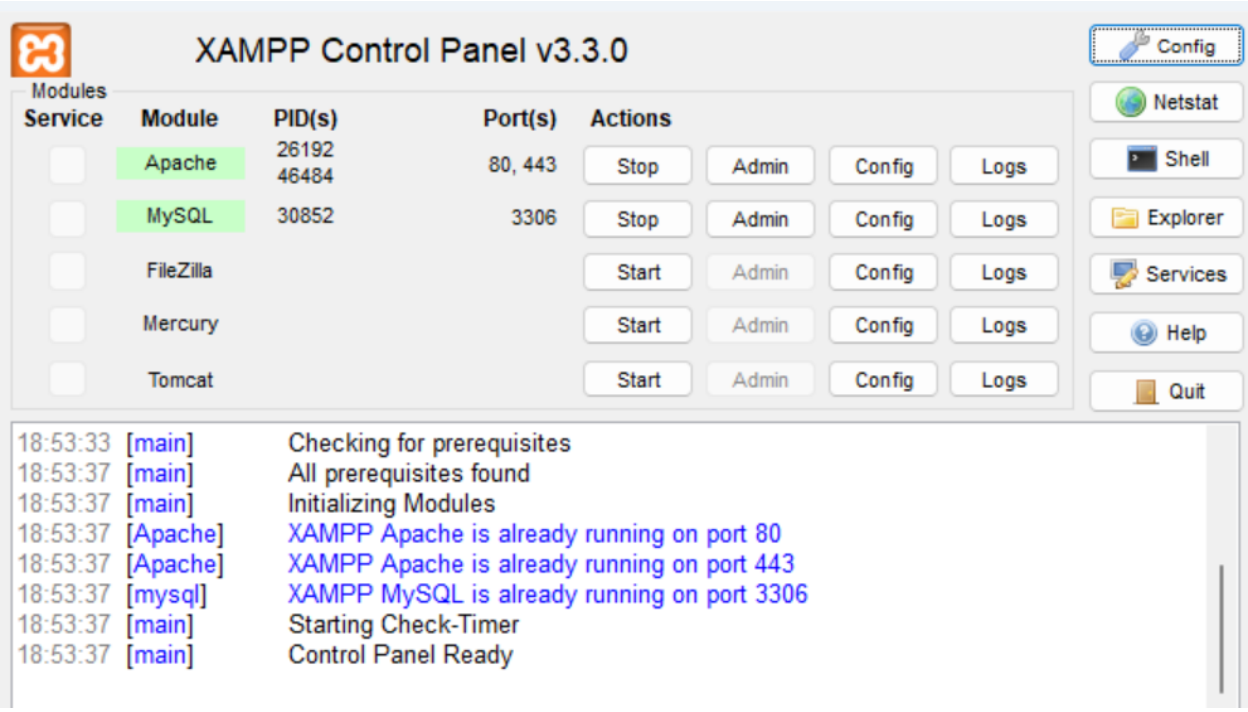


Ilustración 28 Control Panel de XAMPP

Una vez iniciada la conexión, para inicializar la base de datos se deberá ingresar en http://localhost/phpmyadmin , con un navegador, por ejemplo.

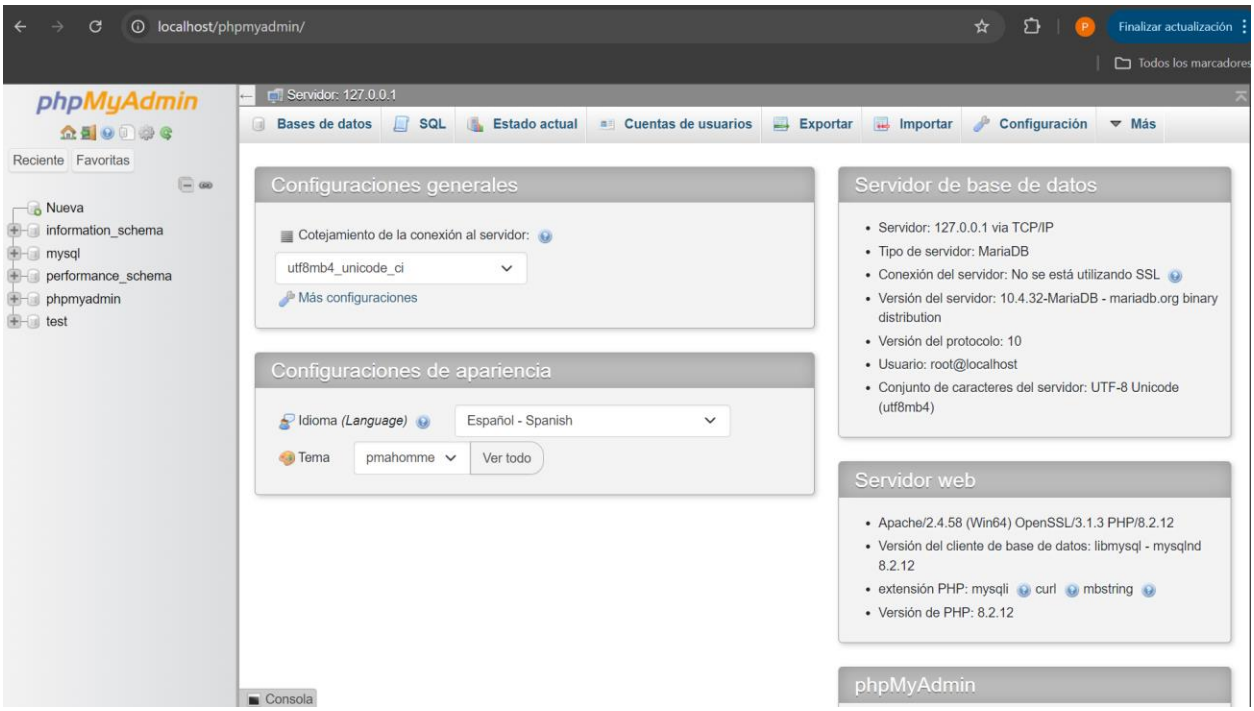


Ilustración 29 Página principal phpMyAdmin

Para inicializar la base de datos es necesario importar una serie de archivos .sql. Estos archivos están incluidos en la carpeta "migrations".

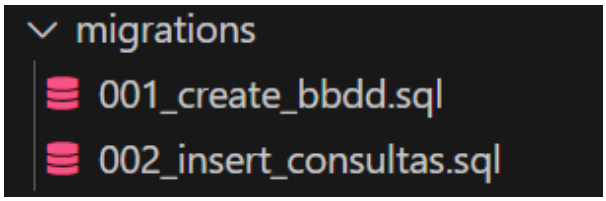


Ilustración 30 Carpeta migrations

Para importar dichos archivos, en la base de datos, en la pestaña "Importar" se debe seleccionar cada archivo uno por uno, haciendo clic en "Seleccionar archivo".

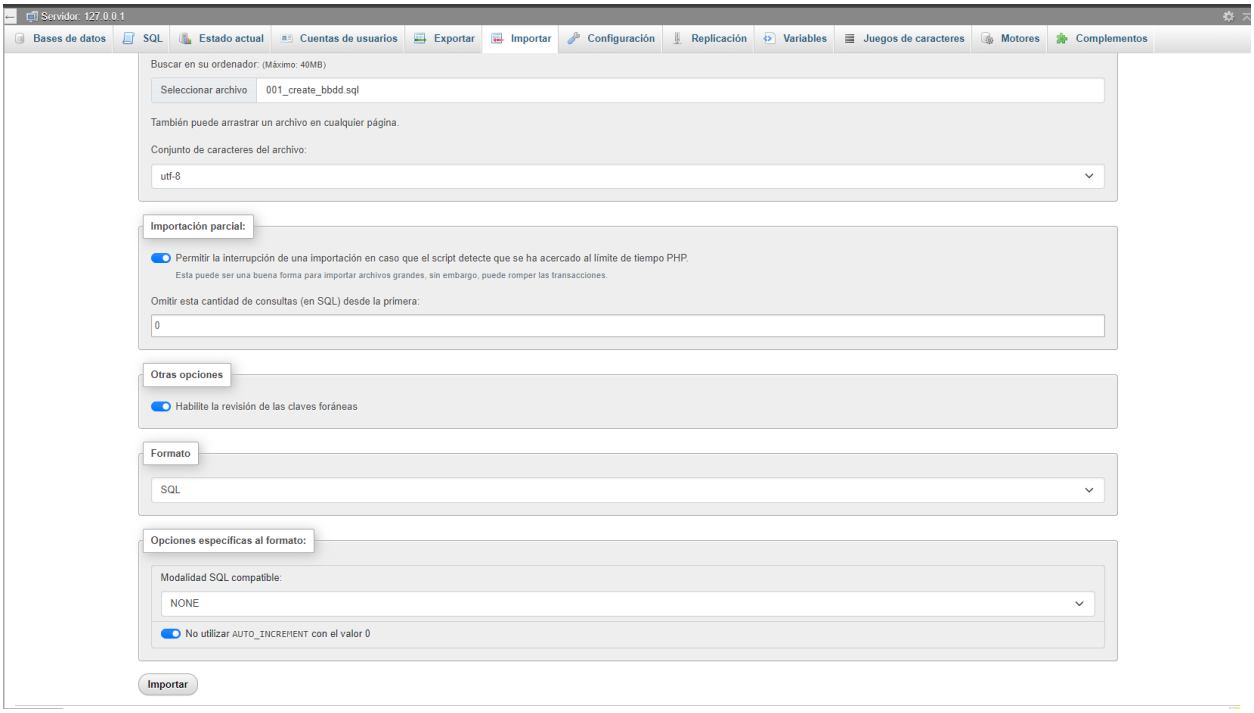


Ilustración 31 Pestaña Importar de phpMyAdmin

Primero se importará el archivo "001-create_bbdd.sql", este primer archivo creará la base de datos llamada "plataforma-educativa". Una vez creada la base de datos, entre en ella y seleccione la pestaña "Importar", entonces, seleccione el segundo archivo "002-insert_consultas", el cual inicializará los datos de usuarios y cursos.

Una vez finalizada la importación, se deben de haber creado las siguientes tablas:

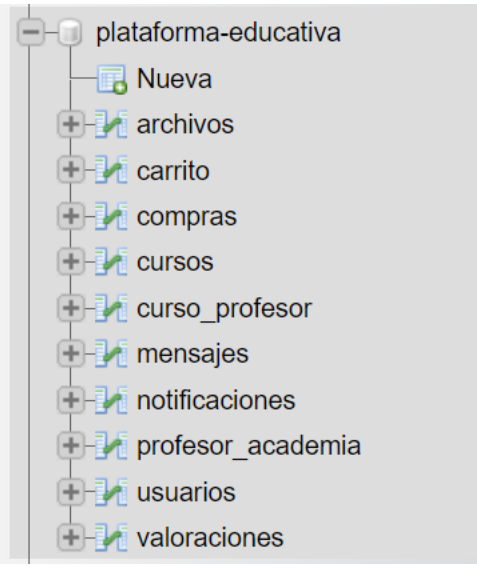


Ilustración 32 Tablas creadas en phpMyAdmin

La tabla de usuarios será inicializada con cinco usuarios:

- El usuario admin, de correo admin@gmail.com y contraseña "admin" tendrá rol administrador en la web.
- El usuario Alvaro, de correo alvaro@gmail.com y contraseña "alvaro" tendrá el rol de usuario normal.
- El usuario Clara, de correo clara@gmail.com y contraseña "clara" tendrá el rol de profesor (usuario médium) en la plataforma.
- El usuario Amanda, de correo amanda@gmail.com y contraseña "amanda" tendrá el rol de profesor limitado (usuario limited) en la web.
- El usuario academia, de correo academia@gmail.com y contraseña "academia", tendrá el rol de academia (usuario premium).

Los cursos también serán incializados con una serie de cursos básicos y sus profesores correspondientes:

De la misma manera, se tendrá inicializada la tabla de cursos_profesor y profesor_academia:

Finalmente, se podrán crear una serie de valoraciones de prueba para los cursos. Para ello se crearán comentarios de reseña para los cursos comprados por Alvaro.

Una vez inicializada la base de datos, ya se podrá acceder a la página web con normalidad. Están a disposición del usuario los cinco usuarios ya creados en la base de datos, para probar las distintas funcionalidades de la aplicación.