Actualización de la estructura y código de los portales de grupo en el proyecto COMA para mejorar la personalización y adaptabilidad responsiva.

Karen Tatiana Pimiento Martínez y Marco Andrés Pinzón Gómez

Trabajo de Grado para Optar al Título de Ingeniero de Sistemas

Director

Luis Ignacio González Ramírez

Magíster en Informática

Universidad Industrial de Santander

Facultad de Ingenierías Físico-Mecánicas

Escuela de Sistemas e Informática

Ingeniería de Sistemas

Bucaramanga

2026

Dedicatoria

Primero, doy gracias a Dios, a mi familia y, de manera muy especial, a mi madre, Evelia Martínez Patiño, quien hizo posible este sueño de estudiar en la UIS. Aunque en muchos momentos lo vi difícil, este camino se logró con esfuerzo, dedicación y mucho amor. Ella ha sido mi motor, al igual que mi hermanita, Mariana Almenárez Martínez, a quien quiero demostrarle que todo es posible cuando uno se lo propone, y ser para ella un ejemplo de perseverancia.

Hoy culmina una etapa muy especial en mi vida, y gran parte de este logro se lo debo a la confianza, el amor y el apoyo incondicional que he recibido durante este largo camino de aprendizaje. También quiero agradecer a la Universidad por brindar oportunidades de ingreso a través de sus sedes, porque gracias a ello, y con esfuerzo y dedicación, pude entrar sin necesidad de haber obtenido un puntaje perfecto en el ICFES.

Agradezco también al grupo de Innovación y Desarrollo de Calumet por su respaldo y por todos los conocimientos compartidos. En especial, a Juan Lipez, por su valiosa orientación y sus consejos para fortalecer este proyecto. Asimismo, agradezco a todos mis compañeros, con quienes compartí risas, lágrimas y grandes aprendizajes. Aunque fui una de las pocas mujeres en el grupo, siempre me sentí como en casa, y eso hizo de esta experiencia algo aún más significativo.

Este logro también se lo debo a muchas personas maravillosas que han pasado por mi vida, enseñándome algo nuevo cada día y acompañándome en este paso tan importante hacia mi vida profesional. Estoy profundamente agradecida, porque la universidad no solo abre puertas, sino que también transforma vidas y deja enseñanzas que permanecen para siempre.

Karen Tatiana Pimiento Martínez

Dedicatoria

Agradezco profundamente a **Zully Stella Gómez** y **Marcos Ramces Pinzón**, mis padres, quienes con su apoyo incondicional y su esfuerzo diario me han acompañado en estos años de estudio y me inculcaron los valores que tanto me caracterizan; a ellos, quienes han hecho posible esta nueva meta en mi vida. También agradezco a ese grupo de amigos con el que he estado desde el colegio, esos cinco personajes que me han acompañado y/o aconsejado en cada decisión laboral, académica o emocional de mi vida. A ellos les agradezco el tiempo que dedicaron para hacer mi vida más feliz. Agradezco a toda mi familia la cual siempre ha estado pendiente de cualquier circunstancia en la que pudiera necesitar ayuda y con quienes sé que siempre contaré para apoyarme y animarme, y también a aquellas personas que, gracias a la vida y al universo, he conocido y que poco a poco han hecho de mi estancia en la universidad una de las mejores etapas de mi vida: a las personas con las que comparto deporte, a mis amigos de canchas, a mis amigos de CALUMET, a mis amigos del gimnasio y a mis amigos de la carrera. Todos ellos llenaron de alegría mi camino y me recordaron siempre que en el mundo todavía hay gente muy buena, divertida e increíble. Una mención especial a **Juan Lipez**, que con sus años de experiencia nos orientó en la realización de este proyecto; a CALUMET, por permitirme ingresar al grupo y acogerme con tanta amabilidad; allí conocí personas que me enseñaron muchas cosas que recordaré a lo largo de mi vida; a mi compañera **Karen** y al director **Luis Ignacio**, por el aporte que también tuvieron en la realización de este proyecto.

Me siento muy agradecido con todos. Gracias, Dios; gracias, Universo; y gracias, UIS.

Marco Andrés Pinzón Gómez

Tabla de contenido

Lista de Tablas

Lista de Figuras

Lista de Apéndices

Los apéndices están adjuntos y puede visualizarlos en la base de datos de la biblioteca UIS.

Apéndice A. Vistas de escritorio y móvil del portal de grupos actual.

Apéndice B. Vistas de escritorio y móvil en el nuevo portal de grupo.

Apéndice C. Manual de usuario del portal de grupos.

Apéndice D. Reporte de pruebas de integración.

Apéndice E. Resumen de los escenarios evaluados con los requerimientos.

Apéndice F. Resumen de los escenarios evaluados con los casos de uso.

Resumen

Título: Actualización de la estructura y código de los portales de grupo en el proyecto COMA para mejorar la personalización y adaptabilidad responsiva*

Autor: Karen Tatiana Pimiento Martínez, Marco Andrés Pinzón Gómez**

Palabras clave: Modernización, Sistema COMA, Interfaz de usuario, Experiencia del usuario, Responsive.

Descripción:

Este trabajo de grado presenta la actualización de la estructura y el código de los portales de grupo del sistema Comunidad Académica (COMA) de la Universidad Industrial de Santander, con el propósito de mejorar su personalización, adaptabilidad responsiva, rendimiento y mantenibilidad. El proyecto surgió debido a diversas limitaciones identificadas en la plataforma original, especialmente relacionadas con el diseño visual, la experiencia de usuario, la organización del código y la adaptación a dispositivos móviles. Estas problemáticas afectan tanto la interacción de los usuarios con los portales como la facilidad de mantenimiento y evolución de la plataforma por parte de los desarrolladores.

La propuesta se desarrolló mediante una metodología compuesta por fases de análisis, diseño, implementación y validación. Durante la etapa de análisis se identificaron los principales problemas presentes en la arquitectura y en la interfaz de usuario. Posteriormente, se diseñó una nueva estructura frontend enfocada en mejorar la experiencia visual y la capacidad de adaptación a diferentes resoluciones y dispositivos. En la fase de implementación se integraron nuevas tecnologías y componentes que permitieron modernizar la apariencia de los portales y optimizar la comunicación entre frontend y backend.

Como resultado, se implementó una nueva versión con una interfaz más dinámica, moderna y adaptable, acompañada de mejoras en el backend que fortalecen la mantenibilidad, escalabilidad y organización del sistema COMA. Además, las pruebas realizadas permitieron evidenciar mejoras en navegación y usabilidad en distintos entornos de uso.

---

*Trabajo de Grado

**Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería de Sistemas e Informática. Ingeniería de Sistemas. Director: Luis Ignacio González Ramírez

Abstract

Title: Updated the structure and code of the group portals in the COMA project to improve customization and responsiveness*

Author: Karen Tatiana Pimiento Martinez, Marco Andres Pinzon Gomez.**

Keywords: Modernization, Usability, COMA System, User Interface, User Experience, Responsive

Description:

This undergraduate thesis presents an update to the structure and code of the group portals within the Academic Community (COMA) system at the Industrial University of Santander, with the aim of improving their customization, responsive design, performance, and maintainability. The project arose due to various limitations identified in the original platform, particularly related to visual design, user experience, code organization, and adaptation to mobile devices. These issues affected both user interaction with the portals and the ease with which developers could maintain and evolve the platform.

The proposal was developed using a methodology comprising analysis, design, implementation, and validation phases. During the analysis phase, the main problems in the architecture and user interface were identified. Subsequently, a new frontend structure was designed focused on improving the visual experience and the ability to adapt to different resolutions and devices. During the implementation phase, new technologies and components were integrated to modernize the portals' appearance and optimize communication between the frontend and backend.

As a result, a new version of the portal was launched with a more dynamic, modern, and adaptable interface, accompanied by backend improvements that enhance maintainability, scalability and organization of the COMA system. In addition, the tests conducted demonstrated improvements in navigation, usability, and compatibility across different usage environments.

---

*Undergraduate Thesis

**Faculty of Physical-Mechanical Engineering. School of Systems and Computer Engineering. Systems Engineering. Advisor: Luis Ignacio González Ramírez

Introducción

La Universidad Industrial de Santander, a través de la plataforma Comunidad Académica (COMA), dispone de un sistema de portales de grupo orientado a la publicación y administración de información académica e institucional de diferentes escuelas, programas y grupos universitarios. Por medio de estos portales, la comunidad académica puede consultar noticias, eventos, calendarios, inscripciones y otros contenidos de interés, mientras que los administradores cuentan con herramientas para gestionar y actualizar la información del grupo.

Sin embargo, con el paso del tiempo, los portales de grupo empezaron a presentar problemas claros en dos frentes: el técnico y el de usabilidad. Desde el punto de vista técnico, parte importante de su estructura visual y de su código se mantenía sobre una base antigua, apoyada en tecnologías y enfoques que ya no responden adecuadamente a las necesidades actuales del sistema. En etapas anteriores, el uso de herramientas como Backbone facilitó la depuración y solución de ciertos problemas, pero con la evolución del proyecto, esta base terminó dificultando el mantenimiento, la escalabilidad y la incorporación de nuevas funcionalidades. Además, el código de esta sección permaneció sin una actualización profunda desde aproximadamente 2015, lo que incrementó su obsolescencia y complejidad de mantenimiento.

En cuanto a la usabilidad, la principal limitación identificada fue la falta de diseño responsivo. Los portales no se adaptan correctamente a dispositivos móviles ni a diferentes tamaños de pantalla, lo que afectaba la visualización del contenido, la navegación y la

experiencia general del usuario. Esta situación representaba una desventaja importante, considerando que gran parte de la comunidad universitaria accede actualmente a la información institucional desde teléfonos móviles y tabletas.

A partir de este diagnóstico, surgió la necesidad de actualizar la estructura y el código de los portales de grupo del proyecto COMA, con el propósito de mejorar su personalización, modernizar su base tecnológica y garantizar una experiencia de usuario más clara, estable y responsiva. Esta actualización implicó no solo un rediseño visual, sino también una reorganización del frontend y la adopción de una arquitectura más modular y mantenible.

Para ello, el proyecto se desarrolló con tecnologías modernas orientadas al desarrollo frontend, entre ellas **React 19, TypeScript, Vite, SSR, Hono, Tailwind CSS** y la librería institucional **Elise UI**, además de herramientas complementarias para la administración de contenido, calendarios e inscripciones. Esta nueva base permitió construir una solución más robusta, organizada y escalable, preparada para integrarse con los servicios backend ya existentes y para facilitar la evolución futura de los portales de grupo desde una base tecnológica común.

En este sentido, el presente trabajo de grado constituye un aporte orientado a la modernización de los portales de grupo de COMA, buscando resolver problemas concretos de mantenimiento y responsividad, y fortaleciendo una herramienta institucional clave para la comunicación y gestión de información dentro de la Universidad Industrial de Santander.

1. Planteamiento y justificación del problema

La plataforma Comunidad Académica (COMA) es un sistema de información web desarrollado y administrado por el grupo Calumet de la Universidad Industrial de Santander (UIS). Su propósito es complementar los espacios de interacción digital de las diferentes escuelas de la institución y apoyar el cumplimiento de la misión universitaria en sus tres funciones sustantivas: docencia, investigación y extensión. A través de COMA, la comunidad universitaria puede actualizar información personal, publicar noticias, organizar eventos, gestionar agendas, intercambiar documentos y acceder a un amplio catálogo de servicios académicos y administrativos. Dada su criticidad operativa como eje central de la vida digital institucional, su arquitectura debe garantizar eficiencia, mantenibilidad y un modelo de seguridad robusto capaz de escalar con el crecimiento continuo de la plataforma.

La arquitectura de COMA corresponde al modelo cliente/servidor en tecnología Java EE, en el que la lógica de negocio reside en un servidor de aplicaciones Tomcat, la persistencia se gestiona sobre una base de datos relacional MySQL y la capa de presentación se implementa mediante Java Server Pages (JSP). Este modelo, ampliamente adoptado en sistemas empresariales durante las últimas décadas, ofrece una separación clara de responsabilidades entre el cliente y el servidor; sin embargo, la evolución orgánica del sistema a lo largo del tiempo, con la incorporación sostenida de nuevos servicios y módulos, ha generado una acumulación de deuda técnica que compromete sus atributos de calidad fundamentales.

Trabajos previos, como el de Escalante Pinilla y Pérez Bolívar (2025), ya habían abordado aspectos de mantenimiento e infraestructura de COMA, consolidando la plataforma

en un único servidor institucional. No obstante, los problemas estructurales relacionados con la gestión de menús, la arquitectura de autorización y el modelo de datos de permisos permanecían sin resolver y constituyen el punto de partida del presente trabajo.

1.1. Impacto esperado

La reestructuración y modernización del código de los portales de grupo del sistema COMA representa un avance significativo para mejorar la experiencia digital de la comunidad universitaria. Este proyecto permitirá una navegación más ágil, intuitiva y compatible con dispositivos móviles, facilitando el acceso constante a la información institucional. Al optimizar el rendimiento y actualizar el diseño visual. Además, la actualización del sistema reducirá las dificultades de mantenimiento técnico y asegurará la sostenibilidad tecnológica a largo plazo.

1.2. Viabilidad

El proyecto es viable, ya que se basa en el uso de tecnologías de código abierto que eliminan la necesidad de licencias costosas. El grupo CALUMET, responsable del desarrollo y mantenimiento, cuenta con experiencia en proyectos previos de la plataforma COMA, lo cual garantiza un conocimiento profundo del entorno técnico y funcional. Asimismo, las escuelas y facultades disponen de infraestructura tecnológica suficiente, incluyendo servidores institucionales y personal técnico calificado. La existencia de una comunidad universitaria comprometida con la mejora continua del sistema digital asegura el respaldo humano y organizativo necesario para su ejecución exitosa.

1.3. Viabilidad técnica

La viabilidad técnica del proyecto se sustenta en que la actualización de los portales de grupo se desarrollará sobre tecnologías modernas y consolidadas para frontend, entre ellas React, TypeScript, Vite, SSR, Hono y Tailwind CSS, además del uso de componentes reutilizables.

Estas herramientas permiten construir una solución modular, escalable y mantenible, adecuada para la evolución del sistema. Asimismo, la nueva estructura del frontend conserva la integración con los servicios backend ya existentes en COMA, lo que permite modernizar la parte visual y funcional de los portales sin afectar la continuidad general de la plataforma. A esto se suma el uso de prácticas de organización del código y control de versiones que favorecen la estabilidad, la trazabilidad de cambios y el mantenimiento futuro.

1.4. Viabilidad económica

Desde el punto de vista económico, el proyecto es viable porque no requiere una inversión alta en adquisición de licencias o infraestructura adicional. La propuesta se apoya principalmente en tecnologías de uso libre y en recursos tecnológicos institucionales ya existentes, lo que reduce significativamente los costos de implementación. Además, al tratarse de una actualización sobre una plataforma ya operativa, se aprovechan componentes, servicios y conocimientos previos del sistema COMA. De igual forma, la modernización del código contribuirá a disminuir costos futuros asociados al mantenimiento, ya que una estructura más limpia y organizada facilita la corrección de errores y la incorporación de mejoras posteriores.

1.5. Viabilidad social

La iniciativa tendrá un impacto positivo en la comunidad universitaria al facilitar la interacción entre estudiantes, docentes y personal administrativo mediante un entorno digital más accesible y actualizado. Los portales renovados fomentarán una comunicación más directa y transparente, fortaleciendo los lazos académicos y administrativos entre los distintos miembros de la universidad. Además, al garantizar el acceso equitativo desde cualquier dispositivo, se promueve la inclusión digital y se mejora la participación de los usuarios en los procesos institucionales.

2. Objetivos

2.1. Objetivo general

Reingeniería de los portales de grupo en la plataforma COMA, diseñando y desarrollando un diseño adaptable a diferentes tamaños de dispositivo, además agregando mayores opciones de personalización y facilitando la gestión de los administradores de grupos.

2.2. Objetivos específicos

Revisar la estructura y el diseño de la página para los portales de grupos de COMA, buscando oportunidades de mejora.

Definir los requerimientos funcionales relacionados con la sesión de portales de grupos.

Diseñar y desarrollar una nueva interfaz adaptable que garantice una experiencia de usuario fluida y consistente en dispositivos móviles, tablets y pantallas de escritorio.

Implementar y mejorar opciones de personalización visual de los grupos para que los administradores los modifiquen según sus preferencias.

Actualizar la base de código de los portales de grupos con nuevas herramientas que faciliten el mantenimiento para futuras actualizaciones.

Validar el funcionamiento de los cambios realizados en la plataforma COMA en los diferentes grupos y en distintos dispositivos.

3. Marco teórico

3.1. Antecedentes del sistema COMA

El sistema Comunidad Académica (COMA) fue creado en 2015 como una plataforma orientada a centralizar la gestión y publicación de información académica, administrativa e institucional de las diferentes escuelas, facultades y grupos de la Universidad Industrial de Santander. Su propósito fue ofrecer un entorno web unificado que facilitara la consulta y administración de contenidos relevantes para la comunidad universitaria.

En sus primeras etapas, el sistema COMA se desarrolló sobre una arquitectura tecnológica que incorporaba HTML, CSS, PHP y bases de datos MySQL, lo que permitió construir una plataforma funcional acorde con los estándares de desarrollo web vigentes en ese momento. Esta base tecnológica hizo posible consolidar un espacio institucional para la difusión de información y la interacción entre estudiantes, docentes, administrativos y grupos académicos, facilitando procesos de consulta y gestión dentro de la universidad.

Con el paso del tiempo, la plataforma fue creciendo mediante la incorporación de nuevos módulos y funcionalidades. Sin embargo, esta evolución no se produjo de manera homogénea en todos sus componentes. Mientras algunas partes del sistema recibieron mejoras y actualizaciones, otras conservan estructuras visuales y bases de código antiguas, especialmente en lo relacionado con los portales de grupo. Como resultado, comenzaron a presentarse dificultades de mantenimiento, inconsistencias en la experiencia de usuario y limitaciones para incorporar cambios de forma ágil.

Una de las principales debilidades identificadas en esta evolución fue la permanencia de enfoques tecnológicos anteriores en el frontend de los portales de grupo. En etapas previas del sistema se utilizaron herramientas como Backbone, que en su momento facilitaron la organización del código y la depuración de ciertos problemas puntuales. No obstante, con el crecimiento del proyecto, esta base dejó de ser suficiente para responder adecuadamente a necesidades actuales de mantenibilidad, escalabilidad y modernización visual. Como consecuencia, realizar ajustes, corregir errores e incorporar mejoras en los portales de grupo se volvió un proceso más complejo para el equipo de desarrollo.

3.2. Diseño Web Responsivo

Es un enfoque de desarrollo que permite adaptar un sitio web a diferentes tamaños de pantalla y dispositivos, garantizando una visualización correcta y una interacción consistente para el usuario. Para ello, utiliza una misma base de código junto con técnicas de diseño flexible que reorganizan los elementos de la interfaz según el espacio disponible.

Actualmente, este enfoque es fundamental en el desarrollo web, debido al incremento del acceso desde dispositivos móviles. En el proyecto COMA, su implementación es especialmente relevante, ya que responde a una de las principales limitaciones identificadas en los portales de grupo: la falta de adaptabilidad a distintos dispositivos. Por ello, el diseño responsivo se convierte en un componente clave de la actualización propuesta, al mejorar la navegación, la visualización del contenido y el acceso a la información institucional.

3.3. Lenguajes y Tecnologías

3.3.1. HTML

Es el lenguaje de marcado utilizado para estructurar el contenido de las páginas web mediante etiquetas. Define elementos como encabezados, párrafos, enlaces, imágenes y otros componentes que conforman la base de una página web.

3.3.2. CSS y Elise

CSS: Cascading Style Sheets (CSS) es un lenguaje central de la plataforma web abierta y se utiliza para agregar estilos (ejemplo, fuentes, colores , espaciado) a los documentos web.

Elise: Biblioteca de estilos visuales diseñada por el grupo Calumet; su función es brindar a los programadores una serie de diseños y componentes para la implementación en la página web.

3.3.3. JavaScript

Es un lenguaje de programación que permite agregar funcionalidades avanzadas a las páginas web, utilizado principalmente del lado del cliente, permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

3.3.4. Java, JSP y Servlets

Java: Es un lenguaje de programación ampliamente utilizado para codificar aplicaciones web; su programación es rápida, segura y confiable para modificarlo todo, desde aplicaciones móviles y software empresarial hasta aplicaciones de macrodatos y tecnologías del servidor.

JSP: La tecnología JavaServer Pages permite generar contenido web dinámico como, por ejemplo, archivos HTML, DHTML, XHTML y XML, para incluirlos en una aplicación web. Los archivos JSP son una forma de implementar contenido de páginas dinámico del lado del servidor.

Servlets: Los servlets son programas Java™ que utilizan la interfaz de programación de aplicaciones (API) de servlet Java. Debe empaquetar los servlets en un archivo WAR (archivador de aplicación web) o un módulo web para su despliegue en el servidor de aplicaciones.

3.3.5. Tomcat

Apache Tomcat funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Oracle Corporation.

3.3.6. Git

Es un sistema de control de versiones distribuido, gratuito y de código abierto, diseñado para gestionar todo, desde proyectos pequeños hasta muy grandes, con velocidad y eficiencia.

3.3.7. TypeScript

Es una extensión de JavaScript que incorpora tipado estático y herramientas que facilitan la detección temprana de errores. Su uso permite desarrollar aplicaciones más organizadas, mantenibles y escalables, especialmente en proyectos grandes o con múltiples módulos. En este proyecto aporta mayor control y claridad al código.

3.3.8. React

Es una biblioteca de JavaScript orientada a la construcción de interfaces de usuario mediante componentes reutilizables. Este enfoque facilita dividir la aplicación en partes más pequeñas, organizadas y fáciles de mantener. En el proyecto se utiliza para desarrollar una interfaz moderna, dinámica y estructurada.

3.3.9. Vite

Es una herramienta de desarrollo y compilación para aplicaciones frontend modernas. Se caracteriza por ofrecer tiempos de carga rápidos durante el desarrollo y una construcción eficiente del proyecto para producción. Su uso mejora la velocidad del entorno de trabajo y optimiza el proceso de desarrollo.

3.3.10. SSR (Server-Side Rendering)

Es una técnica de renderizado en la que parte del contenido de la aplicación se genera en el servidor antes de enviarse al navegador. Esto permite mejorar el tiempo de carga inicial y la entrega del contenido al usuario.

3.3.11. Hono

Es un framework ligero para servidores web que permite gestionar solicitudes y respuestas de forma eficiente. En este proyecto se utiliza como parte de la capa de servidor que soporta la ejecución del frontend.

3.3.12. Tailwind CSS

Es un framework de estilos basado en clases utilitarias que facilita la construcción rápida de interfaces visuales. Permite aplicar estilos directamente en los componentes sin necesidad de escribir grandes hojas de estilo separadas. Su uso contribuye a mantener consistencia visual y agilidad en el desarrollo.

3.3.13. Elise UI

Es una librería desarrollada por CALUMET; su función es ofrecer elementos de interfaz ya definidos para mantener uniformidad gráfica y facilitar la construcción de páginas, el proyecto ayuda a estandarizar la presentación visual de los portales.

3.3.14. FullCalendar

Es una biblioteca que permite integrar calendarios interactivos dentro de aplicaciones web. Facilita la visualización y organización de eventos, fechas importantes y actividades programadas.

3.3.15. TipTap

Es un editor de texto enriquecido que permite crear y editar contenido con formato dentro de una aplicación web. Ofrece flexibilidad para administrar textos estructurados sin depender de ediciones manuales en el código.

3.3.16. Puck

Es una herramienta orientada a la construcción visual de páginas mediante bloques editables. Permite organizar contenido de forma más intuitiva y facilita que ciertos elementos puedan administrarse sin modificar directamente la base del sistema.

3.3.17. Suamox

Es un meta-framework para el desarrollo de aplicaciones web, construido sobre Vite, React y Hono. Proporciona funcionalidades como renderizado del lado del servidor (SSR), generación estática de sitios (SSG), enrutamiento basado en sistema de archivos y manejo de layouts, permitiendo estructurar aplicaciones web modernas de forma organizada y estandarizada.

3.3.18. Arquitectura Feature-Based

Es un enfoque de organización del software en el que el código se agrupa según funcionalidades o características del sistema. En lugar de separar los archivos solo por tipo técnico, cada módulo reúne sus componentes, lógica, servicios y recursos relacionados con una función específica de la aplicación.

3.3.19. Arquitectura por capas

Es un modelo de diseño de software que organiza un sistema en diferentes niveles, donde cada capa tiene una responsabilidad específica. Generalmente, estas capas separan la presentación, la lógica del negocio y el acceso a datos, facilitando la organización, el mantenimiento y la evolución del software.

3.3.20. Arquitectura Multitenant

Es un enfoque arquitectónico en el que una misma aplicación atiende a múltiples clientes, organizaciones o unidades funcionales, denominados tenants, compartiendo una base común de software. Cada tenant conserva su identidad, configuración y datos de operación, aunque utilice la misma plataforma.

3.3.21. Endpoints y APIs web

Una API web es un mecanismo que permite la comunicación entre sistemas mediante reglas y puntos de acceso definidos. Dentro de una API, un endpoint corresponde a una ruta específica expuesta por el servidor para consultar, crear, modificar o eliminar información.

3.3.22. Desacoplamiento entre frontend y backend

El desacoplamiento entre frontend y backend consiste en separar la capa de presentación de la capa encargada de la lógica de negocio y del acceso a datos, de manera que ambas puedan evolucionar con mayor independencia. En este esquema, el frontend consume servicios expuestos por el backend en lugar de depender de su implementación interna.

3.4 Plantillas y componentes web

Las plantillas web son estructuras prediseñadas que sirven como base para organizar el contenido y la presentación de una página o aplicación web. Su uso permite mantener una distribución estandarizada de los elementos, facilitando el trabajo de desarrollo y diseño, además de aportar uniformidad visual entre las diferentes secciones del sistema.

Los componentes web, por su parte, son elementos reutilizables de interfaz como botones, menús, formularios o tarjetas. Su utilización favorece el desarrollo modular, mejora la consistencia visual y facilita el mantenimiento del sistema. En este proyecto también se emplea Elise UI 2.0, una librería de componentes desarrollada por el Grupo de Innovación y Desarrollo CALUMET como evolución de la primera versión de Elise, utilizada internamente para estandarizar y modernizar la construcción de interfaces.

3.5 Experiencia de usuario y accesibilidad

La experiencia de usuario, o UX, se refiere a la percepción que tiene una persona al interactuar con un sistema digital. Su diseño busca que la navegación, la consulta de información y el uso general de la plataforma sean claros, intuitivos y satisfactorios. En aplicaciones institucionales, una buena experiencia de usuario facilita el acceso a los contenidos y mejora la interacción con el sistema.

La accesibilidad web consiste en diseñar y desarrollar sitios y herramientas digitales para que puedan ser utilizados por la mayor cantidad posible de personas, incluyendo aquellas con limitaciones físicas, sensoriales o tecnológicas. Estos aspectos son relevantes, ya que la actualización de los portales busca mejorar tanto la experiencia de navegación como el acceso a la información desde distintos dispositivos.

3.6 Desarrollo modular y escalable

El desarrollo modular es un enfoque en el que una aplicación se construye a partir de partes independientes, cada una con una función específica. Esto facilita la organización del

código, el mantenimiento del sistema y la reutilización de componentes, permitiendo realizar cambios de manera más controlada y ordenada.

La escalabilidad se refiere a la capacidad de un sistema para crecer e incorporar nuevas funcionalidades sin afectar su estabilidad; gracias a esto, la estructura modular y escalable permite modernizar los portales de grupo, facilitar futuras actualizaciones y responder de mejor manera a nuevas necesidades funcionales.

4. Metodología

La metodología define el conjunto de actividades y procedimientos que orientan el desarrollo de la actualización de los portales de grupo del sistema COMA. En este proyecto se adoptó una metodología de desarrollo ágil basada en prototipado evolutivo, organizada en fases secuenciales, con el propósito de analizar el estado actual del sistema, identificar necesidades funcionales, diseñar una propuesta de mejora e implementar una solución más mantenible, modular y responsiva. Este enfoque permitió desarrollar la solución de manera progresiva, realizando ajustes continuos a partir de la revisión técnica y de los requerimientos identificados durante el proceso.

Las fases definidas para el desarrollo son las siguientes; posteriormente, en cada una de estas etapas se ejecutaron actividades específicas y se obtuvieron resultados parciales que permitieron dar trazabilidad al proceso y respaldar el cumplimiento de los objetivos planteados.

4.1. Consultar la documentación.

En esta fase se revisará la documentación disponible del sistema COMA, entendida como el conjunto de sus recursos formales y de su propio código fuente, para comprender su funcionamiento y establecer el estado actual del módulo de portales de grupo.

4.1.1. Actividades

- Investigación sobre tecnologías actuales del sistema COMA
- Revisión del funcionamiento y conexiones con el backend y base de datos
- Revisar el diseño antiguo que fue implementado en los portales de grupo.

4.1.2. Entregables

- Modelo Relacional de las Bases de Datos
- Registro visual del estado actual del portal.

4.2. Dialogar con el administrador de COMA.

En esta fase se establecerá un diálogo continuo con el administrador de COMA para alinear la actualización del portal con las necesidades reales de la plataforma y de los usuarios.

4.2.1. Actividades

- Conversación sobre expectativas y limitaciones de las secciones de grupo
- Identificar las nuevas capacidades solicitadas para el portal.
- Precisar los tres tipos de usuarios del portal:

Público: acceso a información general del grupo.

Miembros: profesores, estudiantes o egresados vinculados al grupo.

Administradores: responsables de la gestión del portal.

- Definir las funcionalidades esperadas para cada tipo de usuario y sus respectivos permisos.
- Elaborar requerimientos funcionales detallados a partir de la información suministrada por el administrador.
- Construir casos de uso que describan las principales interacciones de público, miembros y administradores con el portal.
- Definir la estrategia de integración entre frontend y backend.

4.2.2. Entregables

- Requerimientos funcionales proporcionados por el administrador de COMA.
- Casos de uso describiendo actores, objetivos y flujos principales.
- Diagramas de casos de uso que representen gráficamente las interacciones con el sistema.

4.3. Crear un diseño preliminar para las páginas.

En esta fase se propondrá un diseño inicial del portal, coherente con los requerimientos levantados y orientado a mejorar la experiencia de usuario y la adaptabilidad a distintos dispositivos.

4.3.1. Actividades

- Elaborar un diseño que satisfaga los requerimientos del cliente.
- Adaptar el diseño a los diferentes dispositivos móviles (teléfono, tableta y computador).

4.3.2. Entregables

- Histórico de diseños

4.4. Implementación del diseño planteado

En esta fase se llevará a código la propuesta diseñada, con énfasis en una arquitectura clara y en la facilitación de la mantenibilidad a través de componentes reutilizables.

4.4.1. Actividades

- Aplicar el diseño aprobado.
- Aplicar el marco de desarrollo basado en prototipos evolutivos.
- Implementar el sistema de control de versiones.

4.4.2. Entregables

- Capturas del diseño final del portal de grupos.
- Manual de usuario del portal de grupos.
- Código fuente.

4.5. Pruebas de funcionamiento y rendimiento

Finalmente, se validará que el portal actualizado funcione de manera adecuada en distintos escenarios de uso y dispositivos, y que cumple con los objetivos planteados de experiencia y rendimiento.

4.5.1. Actividades

- Ejecutar dichas pruebas de integración con estudiantes reales en dispositivos móviles y de escritorio.
- Evaluar tiempos de respuesta, tiempos de carga y uso de recursos del sistema mediante herramientas de análisis de rendimiento.

4.5.2. Entregables

- Reporte de pruebas de integración y compatibilidad.
- Informe de rendimiento, con métricas de tiempos de respuesta, consumo de recursos y tiempos de carga.

5. Consultar la documentación.

En esta fase se realizó una revisión detallada del portal de grupos existente dentro de COMA, con el fin de comprender su arquitectura, su comportamiento actual en distintos dispositivos y las decisiones técnicas que han moldeado su desarrollo. La revisión se apoyó en dos registros de documentación complementarios. En primer lugar, se consultaron los recursos

formales disponibles sobre la plataforma, entre ellos el portal Calumet Estándar, que consolida los lineamientos arquitectónicos y las convenciones de código del ecosistema de CALUMET.

En segundo lugar, dado que estos recursos se concentran en la descripción a nivel de plataforma y no profundizan en el detalle de cada módulo, la revisión se extendió al código fuente, al esquema de bases de datos y a los artefactos de configuración del sistema, que en la práctica funcionan como la fuente autoritativa del estado actual del módulo. Sobre ambos registros se tomó el sistema como punto de partida funcional para identificar posteriormente oportunidades de mejora en su organización interna, su mantenibilidad y su adaptación a entornos actuales, estableciendo así la línea base sobre la cual se apoya el rediseño desarrollado en las fases siguientes.

5.1. Investigación sobre tecnologías actuales del sistema COMA

Para el desarrollo de esta actividad se partió de los recursos formales de documentación de la plataforma, que se complementaron con una revisión de la versión actual del sistema en producción, abarcando desde la infraestructura del servidor hasta la implementación de las vistas en el navegador. A partir de estas fuentes se estableció que COMA se ejecuta sobre Tomcat, un contenedor de aplicaciones Java, y que dentro de este entorno las vistas han sido construidas principalmente con JSP (JavaServer Pages), una tecnología que permite combinar marcado HTML con lógica programada en Java directamente dentro de la página.

Este recorrido permitió identificar el stack tecnológico actual de COMA: Tomcat como servidor de aplicaciones, Java como lenguaje de servidor, JSP como motor de vistas, HTML y CSS como lenguajes de marcado y presentación, y MySQL como gestor de base de datos. Sin

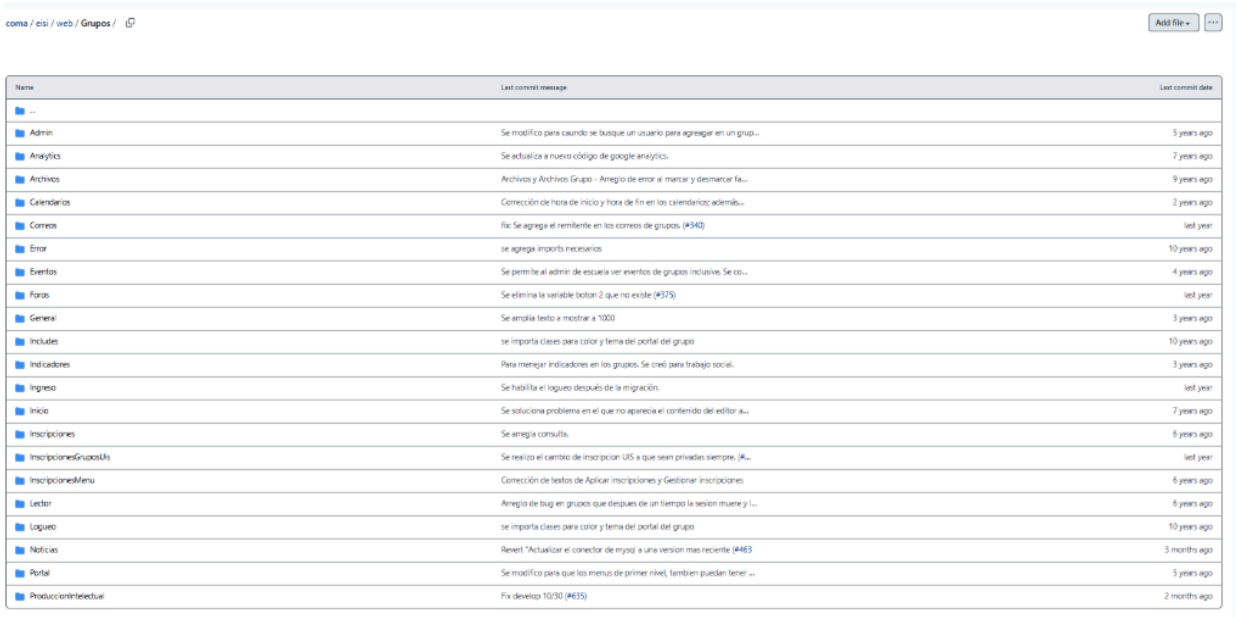
embargo, también dejó en evidencia una limitación arquitectónica importante para el desarrollo propuesto en este trabajo. El esquema basado en JSP que importa clases Java y ejecuta directamente métodos de negocio respondía adecuadamente a un entorno monolítico alojado en un mismo servidor; no obstante, presenta restricciones relevantes frente a una arquitectura de frontend desacoplada y desplegada de manera independiente. En ese escenario, la ausencia de una capa de controladores claramente definida impedía exponer de forma ordenada los servicios necesarios para el consumo remoto, lo que dificultaba tanto la integración entre sistemas como la evolución mantenible del módulo.

Dado que los recursos formales no profundizan en el detalle interno de cada módulo, la revisión se extendió al código fuente del proyecto para analizar la estructura de directorios, con especial énfasis en la forma en que se organizan los módulos, los servicios y las vistas del portal de grupos. Este análisis se realizó tanto a nivel de carpetas como a nivel de código, con el propósito de comprender no solo la ubicación de los archivos, sino también la manera en que interactúan entre sí.

Se constató que la carpeta Grupos/ (ubicada dentro de eisi/web/) agrupa los archivos JSP correspondientes a la funcionalidad del portal, organizados a su vez en subcarpetas por sección, como Admin, Noticias, Eventos, Calendarios, Inscripciones, Foros, Portal e Ingreso, entre otras. Estos archivos constituyen la capa de vista, se ejecutan en el servidor y posteriormente se envían al navegador del usuario. Por su parte, la lógica Java asociada a la gestión de grupos no se encuentra centralizada en una única carpeta, sino distribuida en dos paquetes, grupo, que residen dentro de módulos distintos de la capa de negocio: muisca/grupo/ y comunidad/uis/grupo/, cada uno vinculado a una base de datos diferente. En estos paquetes se

encuentran clases como Grupo, NoticiaGrupo, EventoGrupo e InscripcionGrupo, con métodos orientados a crear, modificar, listar y eliminar la información asociada con la funcionalidad.

Figura 1. Estructura de directorios: carpeta de vistas (archivos JSP), grupos.

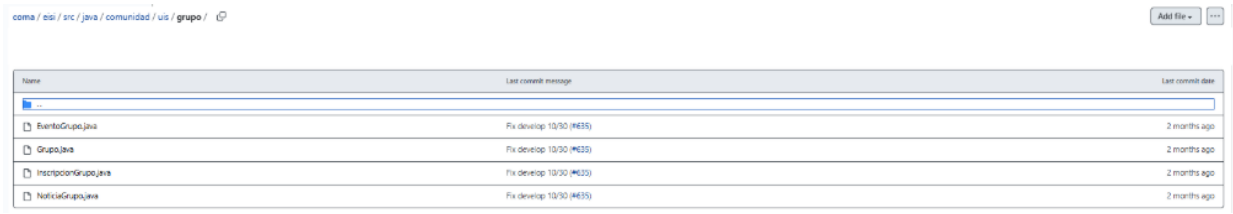


Nota. Tomado del Repositorio oficial en GitHub del proyecto COMA

Adicionalmente, se identificaron clases Java que operan sobre datos de grupos, escuelas u otras entidades relacionadas ubicadas en otros módulos del proyecto, como muisca/noticias, muisca/eventos, muisca/inscripciones o muisca/archivos. Esta dispersión implica que la responsabilidad asociada a una misma funcionalidad del portal de grupos atraviesa varios módulos, lo cual reduce la cohesión interna, complica la trazabilidad del código y dificulta la comprensión global de la arquitectura. En particular, se reconocieron dos módulos clave dentro de la capa de lógica de negocio: comunidad/uis/, cuyos paquetes agrupan las clases

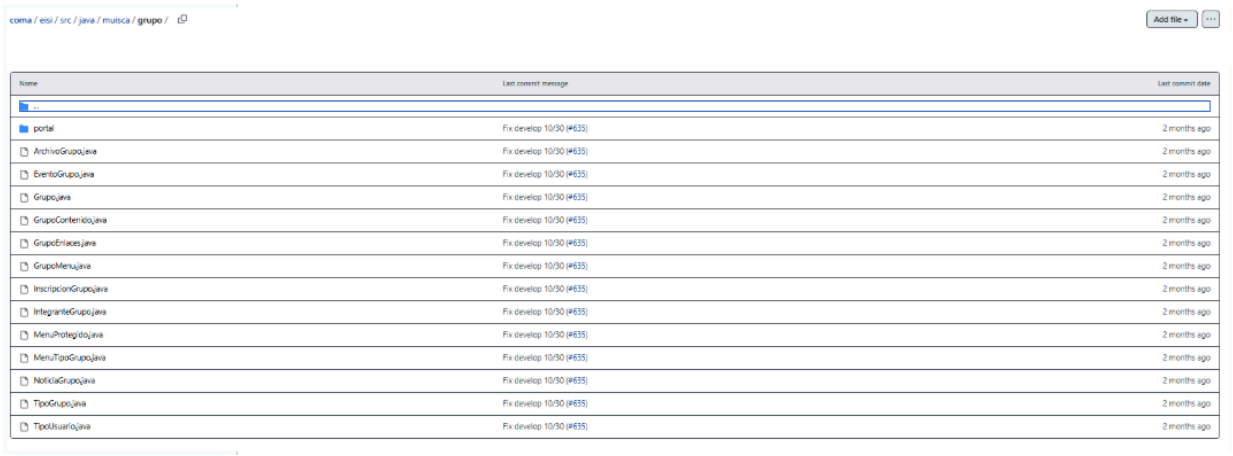
sincronizadas con la base de datos Poseidón, correspondiente a la base de datos global; y muisca/, cuyos paquetes contienen las clases que interactúan con la base de datos Diamante, entendida como la base local por escuela.

Figura 2. Estructura de directorios: carpeta de lógica en comunidad UIS (archivos Java)



Nota. Tomado del Repositorio oficial en GitHub del proyecto COMA

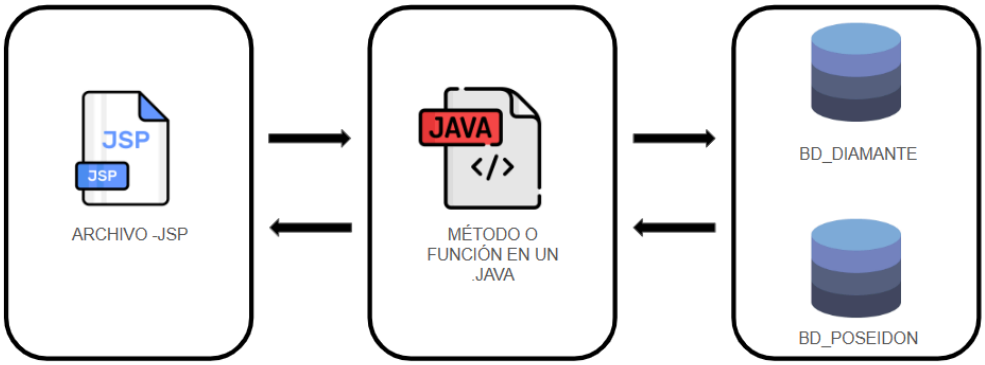
Figura 3. Estructura de directorios: carpeta de lógica en Muisca (archivos Java) grupo



Nota. Tomado del Repositorio oficial en GitHub del proyecto COMA

Comprender esta arquitectura resulta fundamental para identificar con precisión dónde deben realizarse los cambios y de qué manera esos cambios pueden repercutir en el resto del sistema. La falta de una separación clara entre capas, aunque permite que el sistema funcione en su estado actual, dificulta considerablemente su mantenimiento, su prueba y su evolución futura.

Figura 4. Diagrama del flujo actual JSP-Java-Base de datos.



Para ilustrar este flujo con un ejemplo concreto del sistema actual, puede considerarse el caso de la carga dinámica de un listado de tipos de eventos. En el archivo JSP, cuando es necesario poblar un elemento <select> con los tipos de evento disponibles, el código ejecuta el siguiente bloque:

Figura 5. Código de un JSP para explicar flujo .

```
<select  id="idtipoevent">
    <option value=""></option>  <!-- // El option vacio se coloca para que el placeholder
    <%  //
        TipoEvento tipo = new TipoEvento();
        List<TipoEvento> listaTipo = tipo.ListarTipoEvento();
        for (TipoEvento t : listaTipo) {
    %>
    <option value="<%=t.getIdTipoEvento()%>"><%=t.getTipoEvento()%></option>
    <%}%>
</select>
```

Nota. Tomado del Código de CrearEvento.JSP del proyecto COMA

Esta llamada desencadena la ejecución del método ListarTipoEvento() en la clase TipoEvento, ubicada en el paquete muisca. El método inicia creando una lista vacía y posteriormente instancia una conexión a la base de datos Diamante:

Figura 6. Código del método Java de ejemplo.

```
public List<TipoEvento> ListarTipoEvento() {

    List<TipoEvento> listaevent = new ArrayList<>();
    ConexionesDiamante bd = new ConexionesDiamante();
    String sentencia = "SELECT * FROM TB_TipoEvento WHERE Estado='ACTIVO'";
    ResultSet tipoevent = bd.consultarBD(sentencia);
    try {
        while (tipoevent.next()) {
            TipoEvento e = new TipoEvento();
            e.setIdTipoEvento(tipoevent.getString(1));
            e.setTipoEvento(tipoevent.getString(2));
            e.setEstado(tipoevent.getString(3));
            listaevent.add(e);
        }
    } catch (SQLException ex) {
        Logger.getLogger(TipoEvento.class.getName()).log(Level.SEVERE, null, ex);
    }
    bd.cerrarConexion();
    return listaevent;
}
```

Nota. Tomado del Código de TipoEvento.java del proyecto COMA

Una vez que el método retorna la lista al JSP, este recorre los datos y genera dinámicamente las opciones del elemento <select>. Como resultado, el HTML que finalmente llega al navegador ya contiene las opciones pobladas, sin que el usuario observe el proceso interno de consulta y construcción de la información:

Figura 7. Ejemplo de HTML renderizado en el navegador.

```
> <select name="tipoEvento">
      <option value="1">Academico</option>
      <option value="2">Cultural</option>
      <option value="3">Deportivo</option>
  </select>
```

5.2. Revisar el funcionamiento y conexiones con el backend y base de datos.

En el marco de esta actividad, se realizó una revisión integral de la documentación técnica y funcional del sistema COMA, con especial énfasis en la estructura y funcionamiento de los portales de grupo. Este proceso permitió comprender cómo se articulan los diferentes componentes del sistema, tanto a nivel de arquitectura como en la forma en que se gestionan y presentan los datos dentro de los portales.

Desde una perspectiva funcional, se identificó que el portal de grupos se encarga de gestionar y visualizar información relacionada con actividades académicas, noticias, eventos, inscripciones y comunicaciones internas. Estas funcionalidades están diseñadas para operar en

distintos niveles de alcance: algunas se limitan al contexto específico de un grupo, mientras que otras tienen impacto a nivel de escuela o incluso institucional.

En el plano técnico, se evidenció que el funcionamiento del portal se apoya en una arquitectura que combina múltiples fuentes de datos y servicios distribuidos. En particular, se identificó la interacción con dos bases de datos principales: Diamante y Poseidón, las cuales, aunque operan como instancias independientes, cumplen roles complementarios dentro del sistema.

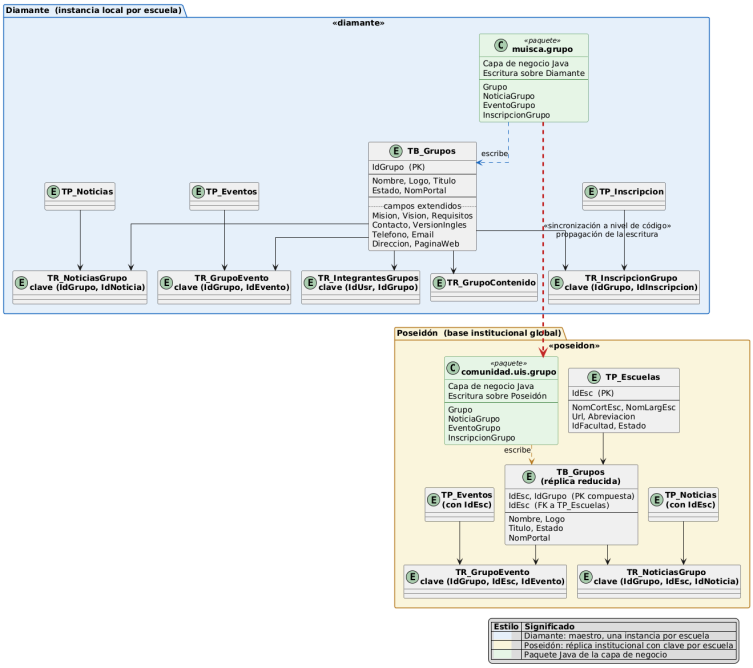
Diamante opera como una instancia local que cada escuela despliega y administra de forma autónoma, y cumple el rol de repositorio maestro y detallado de la información del portal de grupos. En su tabla principal TB_Grupos se registran no solo los datos básicos de cada grupo (nombre, logo, portal, estado), sino también los campos extendidos que sostienen la presentación pública del portal, como la misión, la visión, los requisitos, la información de contacto, los acuerdos institucionales y configuraciones particulares relacionadas con la analítica del sitio o la activación del idioma inglés. A este núcleo se suman las tablas transaccionales asociadas al contenido del portal, entre ellas TP_Noticias, TP_Eventos, TP_Inscripcion y TR_GrupoContenido, así como las tablas de relación que vinculan cada pieza de contenido con su grupo correspondiente, como TR_NoticiasGrupo, TR_GrupoEvento, TR_InscripcionGrupo y TR_IntegrantesGrupos, todas ellas con claves compuestas que incluyen el identificador de grupo.

Esta organización concentra en un único lugar la totalidad del detalle operativo del portal y permite que cada escuela mantenga su información aislada de las demás instancias.

Poseidón, en cambio, cumple un rol centralizado e institucional. Allí reside el registro maestro de escuelas en la tabla TP_Escuelas, desde la cual se resuelve dinámicamente la lista de instancias activas del sistema, y se conserva una réplica reducida y denormalizada de la información de grupos. Su tabla TB_Grupos utiliza una clave compuesta que combina el identificador de escuela con el identificador de grupo, y almacena únicamente los campos necesarios para resolver identidad y presentación a nivel institucional, como el nombre, el logo, el título, el estado y el nombre del portal, dejando los atributos extendidos del lado de Diamante. De manera análoga, las tablas de relación de Poseidón, como TR_NoticiasGrupo o TR_GrupoEvento, incorporan el identificador de escuela como parte de su clave, lo que permite que un mismo identificador de grupo coexista entre escuelas sin ambigüedad y habilita consultas transversales al ecosistema UIS.

La coherencia entre ambas bases no se apoya en un mecanismo de replicación del motor de base de datos ni en uniones en tiempo de consulta, sino en una sincronización explícita a nivel de código: las clases del paquete muisca.grupo ejecutan la escritura sobre Diamante y, al confirmar la transacción, instancian su contraparte en comunidad.uis.grupo para propagar los datos esenciales a Poseidón en una segunda operación de escritura. Este patrón de doble escritura, con Diamante como fuente autoritativa y Poseidón como réplica indexada por escuela, explica la duplicación intencional de clases entre ambos paquetes y determina la forma en que una noticia creada a nivel de escuela queda disponible como referencia institucional para consultas de alcance global.

Figura 8. Esquema dual Diamante y Poseidón del portal de grupos y patrón de doble escritura entre los paquetes Java de la capa de negocio.



Durante la revisión también se observó que la mayor parte de los vínculos entre entidades en ambas bases no se expresa mediante claves foráneas del motor, sino que se gestiona de manera lógica desde el código Java, apoyándose en campos como IdGrupo e IdEsc para establecer las relaciones. Esta decisión responde a las características de la versión de MySQL sobre la que opera el sistema, en la que la verificación de integridad referencial impone una sobrecarga apreciable sobre operaciones concurrentes y consultas que involucran tablas de alto volumen, lo que en la práctica se traducía en cuellos de botella para el portal. Al trasladar esa responsabilidad a la capa de aplicación, el sistema cede la garantía automática que ofrecería el motor a cambio de un control más predecible sobre los tiempos de respuesta.

Adicionalmente, se observó que el acceso a la información varía según el tipo de funcionalidad requerida por el portal. Cuando se necesita mostrar información específica de un

grupo, el sistema consulta la base de datos Diamante correspondiente; en cambio, para información de carácter general o institucional, se recurre a Poseidón. De manera similar, las operaciones de escritura siguen esta misma lógica: los datos locales se almacenan en Diamante, mientras que aquellos con alcance global se registran en Poseidón. Funcionalidades como noticias, eventos, inscripciones, correos y calendarios evidencian este comportamiento dual en la gestión de la información.

En conjunto, esta revisión permitió comprender no solo la estructura técnica del sistema, sino también la lógica funcional que rige la organización de los portales de grupo. Asimismo, facilitó identificar cómo se distribuyen las responsabilidades entre los distintos componentes, lo cual resulta fundamental para futuras tareas de mantenimiento, integración o evolución del sistema.

5.3. Revisar el diseño antiguo que fue implementado en los portales de grupo.

Para esta actividad se realizó una evaluación práctica del portal de grupos, accediendo a él desde diferentes tipos de dispositivos: computadores de escritorio , tablets y teléfonos móviles.

En computadores de escritorio, se observó que el portal muestra la información principal de manera completa y funcional. Sin embargo, carece de una estructura visual consistente y coherente. Los elementos (encabezados, menús de navegación, contenido principal, barras laterales) no siguen un patrón de proporción claro, lo que genera una experiencia visual desigual. No existe un esquema evidente de diseño responsivo; más bien, el portal fue diseñado asumiendo un ancho de pantalla específico, sin consideración por variaciones.

Figura 9. Visualización en dispositivos: Versión de escritorio

Nota. Tomado de la Página oficial del portal del grupo CALUMET

https://ingsistemas.uis.edu.co/eisi/grupo/calumet/#views/gm1/inicio

En dispositivos tablet, el portal comienza a mostrar signos de desadaptación. Aunque la mayoría del contenido es aún visible, se observan pequeños desajustes en la alineación de elementos y algunos componentes pueden aparecer con proporciones extrañas. Sin embargo, la situación no es crítica en este punto.

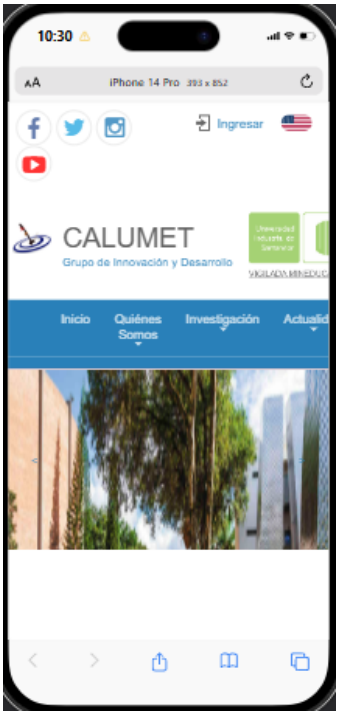
Figura 10. Visualización en dispositivos: Versión de tablet

Nota. Tomado de la Página oficial del portal del grupo CALUMET

https://ingsistemas.uis.edu.co/eisi/grupo/calumet/#views/gm1/inicio

En teléfonos móviles, la situación es notoriamente más problemática. El portal fue claramente concebido para pantallas grandes, y su comportamiento en móviles es inapropiado: parte del contenido no se visualiza adecuadamente, quedando fuera del área visible de la pantalla; aparecen barras de desplazamiento horizontal, obligando al usuario a moverse lateralmente para ver la información, algo que va en contra de las convenciones de navegación en móviles; algunos elementos (como tablas, formularios o menús) están diseñados con un ancho fijo que excede el ancho de la pantalla del dispositivo, causando desbordamiento; y la legibilidad se ve comprometida, ya que el texto puede quedar cortado o parcialmente visible.

Figura 11. Visualización en dispositivos: Versión móvil



Nota. Tomado de la Página oficial del portal del grupo CALUMET

https://ingsistemas.uis.edu.co/eisi/grupo/calumet/#views/gm1/inicio

Esta evidencia visual refuerza de manera contundente la necesidad de replantear la capa de presentación con un enfoque de diseño web responsivo, capaz de adaptarse automáticamente a diferentes tamaños de pantalla sin pérdida de información o funcionalidad. Un rediseño responsivo no solo mejoraría la accesibilidad del portal, sino que también alinearía el sistema con los estándares de usabilidad web contemporáneos.

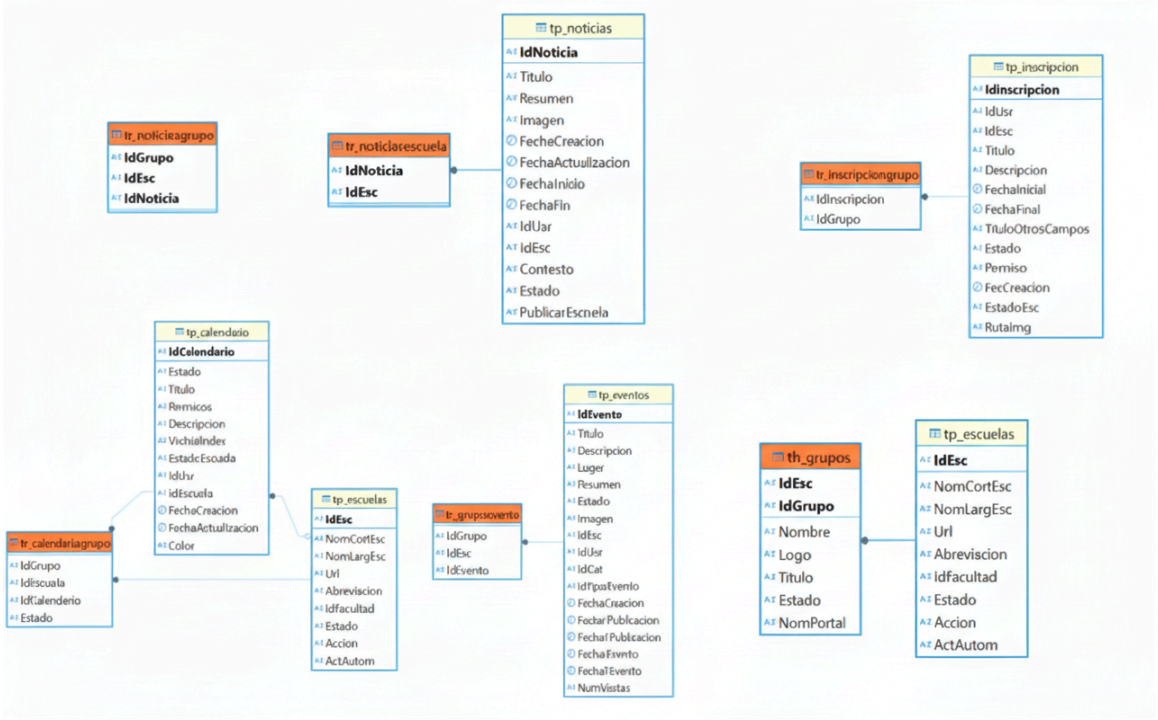
5.4. Entregables

Los resultados de esta fase de consulta y análisis de documentación se concretan en los siguientes artefactos:

5.4.1. Modelo relacional de las bases de datos

Este artefacto presenta una representación gráfica de las tablas más relevantes asociadas al portal de grupos en las bases de datos Poseidón y Diamante, así como de sus relaciones lógicas a través de columnas clave como IdGrupo e IdEsc. A partir de estos diagramas es posible identificar qué tablas intervienen en servicios como noticias, eventos, calendarios, inscripciones y menús del portal, y cómo se encadenan mediante relaciones uno a muchos y tablas de asociación.

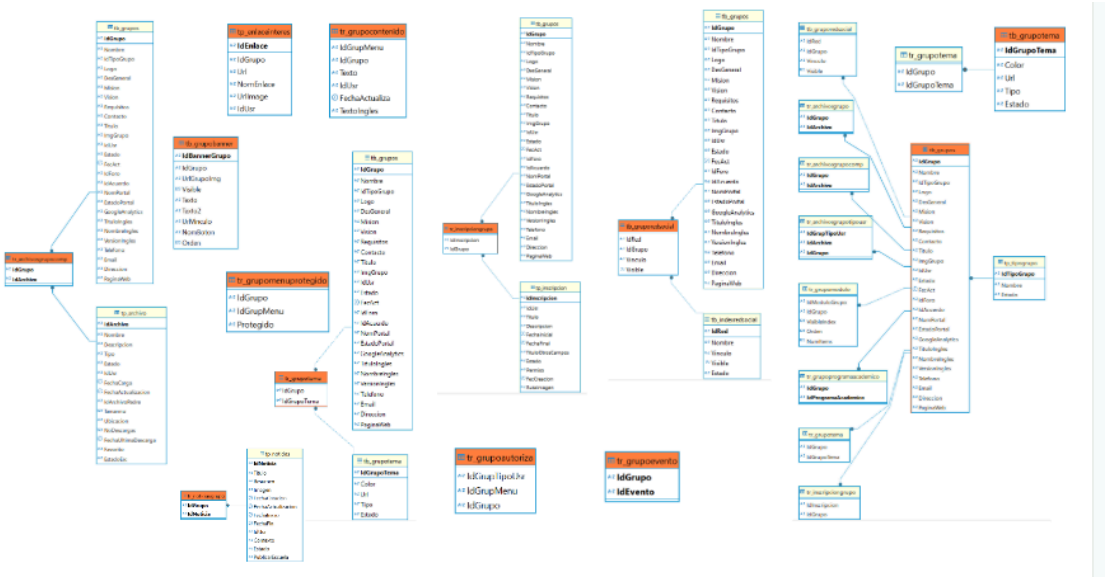
Figura 12. Modelo relacional de la base de datos Poseidón para el portal de grupos



Nota. Base de datos Poseidón, vista de diagrama en DBeaver.

Se destacan tablas como TB_Grupos, TP_Escuelas, TP_Eventos, TP_Calendario, TP_Inscripcion y TP_Noticias, junto con sus tablas de relación (TR_GrupoEvento, TR_NoticiasGrupo, TR_CalendarioGrupo, TR_InscripcionGrupo), articuladas mediante las columnas IdGrupo e IdEsc.

Figura 13. Modelo relacional de la base de datos Diamante para el portal de grupos



Nota. Base de datos Diamante, vista de Diagrama en DBeaver

Se representan las tablas operativas utilizadas por cada grupo (por ejemplo, menús, productos, enlaces, configuraciones de portada y parámetros de grupo), mostrando cómo la estructura local por grupo se organiza en torno al identificador IdGrupo.

5.4.2. Registro visual del estado actual del portal

Este artefacto constituye un compendio de pantallazos que documentan la interfaz actual del portal de grupos en sus versiones para escritorio y dispositivos móviles. Este registro visual actúa como una línea base documentada del estado actual, lo que permitirá que se pueda comparar el antes y el después, validar que las mejoras han sido efectivamente implementadas y evidenciar el impacto en la experiencia de usuario.

Las capturas de pantalla se encuentran en el apartado de los Apéndices, específicamente en el Apéndice A.

6. Dialogar con el Administrador de COMA.

En esta fase se estableció un diálogo con el administrador de COMA para alinear la actualización del portal de grupos con las necesidades reales de la plataforma y de sus usuarios, considerando el funcionamiento actual del sistema, sus restricciones técnicas y la integración con las bases de datos Diamante y Poseidón. A partir de este intercambio no solo se precisaron actores, funcionalidades y requerimientos, sino que también se definió la orientación técnica general que debía seguir la solución propuesta.

6.1. Conversación sobre expectativas y limitaciones de las secciones de grupos

Se realizaron varias reuniones de trabajo con el administrador de COMA para identificar las necesidades del portal de grupos, definir prioridades de actualización y reconocer las restricciones técnicas que debían respetarse durante el desarrollo. En estos encuentros se revisó el funcionamiento actual del portal, las secciones que debían mantenerse, como noticias, eventos, inscripciones, calendarios, correos y páginas de contenido, así como los principales problemas detectados, especialmente en la visualización desde dispositivos móviles y en la dificultad de mantenimiento de la estructura existente. Estas conversaciones permitieron

confirmar que la actualización no debía centrarse únicamente en un cambio visual, sino también en una parte de ella, mejorando la base de código que facilitara futuras modificaciones y permitiera mantener el sistema de forma más ordenada.

A partir de este objetivo, se discutió la necesidad de implementar una arquitectura de código más clara, modular y mantenible, que permitiera organizar mejor las funcionalidades del portal y simplificar su evolución en el tiempo. En este sentido, se planteó trabajar con tecnologías actuales como React, Vite y Hono, incorporando además herramientas internas de CALUMET como Suamox y Elise, con el fin de construir una solución más consistente con el entorno tecnológico del proyecto. Esta renovación buscaba dejar atrás la estructura anterior, que dificulta las actualizaciones, y avanzar hacia una organización del código orientada por funcionalidades y capas, favoreciendo la reutilización de componentes, la separación de responsabilidades y la escalabilidad del sistema.

Asimismo, en las reuniones se dejó claro que esta modernización debía realizarse sin afectar la lógica institucional ni las integraciones existentes con las fuentes de datos del sistema. A medida que se precisaban las necesidades del proyecto, también se hizo evidente que la renovación no podía limitarse exclusivamente al frontend. Por esta razón, se definió que la nueva solución debía construirse como un frontend desacoplado, desplegado en un servidor distinto al backend tradicional y capaz de atender múltiples portales de grupo desde una misma base tecnológica. Esta decisión introdujo explícitamente un enfoque multi-tenant para la capa de presentación y, como consecuencia, hizo necesario adecuar la arquitectura del módulo para permitir la comunicación remota entre ambas partes. En consecuencia, el alcance quedó

definido no solo en torno a la renovación del frontend y de su estructura interna, sino también en torno a una intervención intermedia en el backend que permitiera exponer los servicios requeridos por la nueva interfaz. De esta manera, estas reuniones no solo sirvieron para aclarar expectativas y restricciones, sino también para establecer la dirección técnica de la actualización, en concordancia con el objetivo específico de actualizar la base de código de los portales de grupo con herramientas que facilitaran su mantenimiento en futuras versiones.

Más allá de conservar la paridad funcional con el sistema anterior, durante estas conversaciones el administrador también planteó un conjunto de capacidades nuevas, orientadas principalmente a ampliar las opciones de personalización del portal y a simplificar la labor administrativa sobre el contenido, las cuales se detallan en la siguiente sección.

6.2. Identificar las nuevas capacidades solicitadas para el portal.

En paralelo con la definición de paridad funcional respecto al sistema anterior, el administrador de COMA manifestó un conjunto de necesidades no cubiertas por el portal heredado, orientadas a mejorar la autonomía del administrador del grupo, la adaptabilidad visual del portal y la calidad de presentación de la información. Estas solicitudes dieron forma a cuatro capacidades nuevas que se incorporaron al alcance del proyecto.

La primera capacidad corresponde a la autoría visual de contenido por bloques. El portal anterior no ofrecía a los administradores un mecanismo accesible para redactar o modificar páginas de contenido, ya que su construcción dependía de la manipulación directa de archivos

HTML o JSP por parte del equipo técnico; como consecuencia, cualquier cambio editorial, por pequeño que fuera, requería intervención de desarrollo. El administrador solicitó un editor visual orientado a personas no técnicas que permitiera componer páginas mediante bloques reutilizables, tales como encabezados, párrafos, imágenes, tarjetas, tablas, pestañas y contenido embebido, con versiones independientes en español e inglés. Esta capacidad se alinea con el objetivo de facilitar el mantenimiento y la actualización del portal sin dependencia permanente del equipo de desarrollo.

La segunda capacidad corresponde a la personalización extendida del tema visual. En el portal heredado la configuración de apariencia se limitaba a dos colores del encabezado, lo cual resultaba insuficiente para que cada grupo transmitiera una identidad gráfica diferenciada. El administrador pidió una herramienta de personalización más amplia, que contemplara la definición por grupo de un conjunto de tokens de diseño, entre ellos los colores de fondo, texto, acentos y bordes, la configuración de una escala de sombras derivada de un color base, la aplicación de presets predefinidos y la edición de la imagen de fondo de la pantalla de inicio de sesión. Se solicitó además que el administrador pudiera observar el efecto de los cambios en tiempo real sobre una vista previa navegable del portal antes de guardarlos. Esta capacidad materializa de manera directa el objetivo específico 2.2.4 del proyecto, orientado a implementar y mejorar opciones de personalización visual de los grupos.

La tercera capacidad corresponde a la consolidación de un panel administrativo unificado. En el sistema anterior las pantallas de administración se distribuían en distintas vistas JSP sin una organización coherente que permitiera al administrador ubicar con facilidad las

tareas relacionadas con una misma área de responsabilidad. El administrador de COMA solicitó concentrar la gestión del portal en un único panel, organizado por secciones de responsabilidad, como perfil del grupo, contenido, integrantes, diseño y menús con roles, que sirviera como punto de entrada común para todas las operaciones administrativas.

Finalmente, la cuarta capacidad corresponde a la exposición de datos de contacto institucional en la presentación pública del portal. El administrador solicitó incorporar un espacio visible para el público en el que se mostraran el teléfono, el correo, la dirección y la página web del grupo, información que no tenía un lugar definido en el portal anterior, y que debía poder ser gestionada por el propio administrador desde la configuración del grupo.

6.3. Precisar los tres tipos de usuarios del portal.

A partir de las reuniones, se precisaron los tres tipos de usuarios que interactúan con el portal de grupos. En primer lugar, se identificó al usuario público, que accede sin autenticarse y debe poder consultar información general del grupo, como noticias y eventos visibles para cualquiera. En segundo lugar, se definió al usuario autenticado, que corresponde a cualquier persona vinculada a un grupo específico, como profesores, estudiantes o egresados, y que por tanto requiere acceso a secciones e información interna adicional. Finalmente, se caracterizó al administrador del grupo, responsable de gestionar el contenido del portal, configurar secciones y decidir qué información se publica únicamente a nivel de grupo o también a nivel de escuela o universidad.

Conviene aclarar que, aunque los usuarios autenticados pueden asumir distintos roles dentro de un grupo, como estudiante, profesor o egresado, dichos roles no constituyen actores diferentes dentro del modelo de casos de uso. Los roles no corresponden a categorías fijas definidas por el sistema, sino a entradas configurables que el administrador del grupo define y gestiona en cada portal, junto con los permisos asociados a cada uno. En consecuencia, un mismo requerimiento, como la creación de noticias, puede habilitarse para los usuarios con rol de estudiante y restringirse para los de egresado, o viceversa, según la configuración establecida por el administrador. Las capacidades del usuario autenticado no vienen determinadas, por tanto, por su tipo de vinculación con la universidad, sino por la política de acceso definida por el administrador sobre los menús y funcionalidades del portal. Por esta razón, los roles se tratan como un mecanismo de configuración interno dentro del actor usuario autenticado, y no como actores independientes.

Esta clasificación fue esencial para organizar las funcionalidades y permisos.

6.4. Definir las funcionalidades para cada tipo de usuario y sus respectivos permisos.

Con los tres tipos de usuario ya definidos, se elaboró una lista de funcionalidades esperadas para cada uno. Este paso permitió pasar de la caracterización general de actores a la determinación concreta de las operaciones que el sistema debía soportar. Para el público se consideraron acciones como consultar noticias y eventos públicos del grupo, así como acceder a la información básica relacionada con el mismo. Para el usuario autenticado se incluyeron funcionalidades orientadas a visualizar, crear, editar y eliminar noticias y eventos propios, así como a participar en foros restringidos y acceder a otros contenidos vinculados específicamente al grupo al que pertenece. Dado que la disponibilidad efectiva de estas operaciones depende del

rol asignado y de los permisos configurados por el administrador, en esta etapa se trabajó con el conjunto de funcionalidades que el actor puede llegar a ejercer, dejando la restricción particular por rol como parte de la configuración del portal.

En el caso de los administradores, se definieron capacidades de creación, edición y eliminación de noticias y eventos; gestión de inscripciones; administración de calendarios; y configuración de la visibilidad de los contenidos. Además, se discutió con detalle en qué situaciones un contenido debía almacenarse en la base de datos Diamante de la escuela y en qué casos debía registrarse en Poseidón. De esta manera, cada funcionalidad quedó ligada no solo a un conjunto de permisos, sino también a la base de datos sobre la que opera.

6.5. Elaborar requerimientos a partir de la información suministrada por el administrador.

A partir de la lista de funcionalidades, se procedió a redactar requerimientos funcionales más detallados. Para cada requerimiento se especificó el actor al que estaba dirigido (público, usuario autenticado o administrador), de modo que la información levantada con el administrador pudiera traducirse en una base más precisa para el diseño y la implementación del sistema.

6.6. Construir casos de uso que describen las principales interacciones de público, miembros y administradores con el portal, tanto en forma textual como gráfica.

Finalmente, se elaboraron casos de uso que describen las interacciones más importantes de cada tipo de usuario con el portal de grupos. En primer lugar, se redactaron descripciones textuales organizadas en tablas, donde para cada actor se enumeran los casos de uso asociados y

se incluye una breve explicación del rol de ese usuario en el sistema. Por ejemplo, para el administrador se agruparon acciones como gestionar noticias, eventos y calendarios del grupo, con la aclaración de que dispone de permisos para administrar el contenido y decidir su ámbito de publicación. De forma análoga, para el público y para los miembros se listaron las acciones que pueden realizar y el tipo de acceso que se les concede.

En segundo lugar, se presentaron estos mismos casos de uso mediante diagramas, en los que se muestran los tres actores (público, usuario autenticado y administrador) y los casos de uso vinculados a cada uno. Estos diagramas permiten visualizar de manera sintética las relaciones entre usuarios y funcionalidades, y sirven como puente entre la especificación funcional y las decisiones de diseño que se abordan en las fases posteriores. De manera particular, también ayudaron a precisar qué operaciones debían quedar disponibles como servicios consumibles por el nuevo frontend, aspecto que condujo a la definición de una estrategia específica de integración con el backend.

6.7. Definir la estrategia de integración entre frontend y backend.

Como resultado del proceso anterior, y una vez consolidada la decisión de construir un frontend desacoplado con enfoque multi-tenant, fue necesario traducir dichas necesidades en una estrategia técnica de integración entre frontend y backend. El análisis confirmó que el esquema previo, basado en vistas JSP con invocaciones directas a clases Java, no ofrecía un punto de integración adecuado para un frontend moderno desplegado en otro servidor y responsable de atender múltiples portales según el grupo e idioma solicitados.

Frente a esta necesidad, se concluyó que el módulo requería una capa de controladores más clara, capaz de intermediar entre la capa de presentación y la lógica de negocio. En lugar de mantener la comunicación acoplada entre JSP y clases Java, se optó por exponer endpoints consumibles por el frontend, de manera que la nueva interfaz pudiera solicitar información y ejecutar operaciones mediante servicios definidos explícitamente.

Para implementar esta transición, se adoptó el estándar vigente de CALUMET para la plataforma COMA, basado en el uso de servlets. Esta decisión permitió mantener coherencia con las prácticas institucionales del ecosistema existente y, al mismo tiempo, introducir una organización más clara del módulo. Como parte de esta intervención, se separaron con mayor precisión las responsabilidades entre modelo, lógica de negocio y controladores, reduciendo la dependencia directa de las vistas sobre el acceso a datos y sobre las reglas del sistema.

La definición de esta estrategia constituyó una fase intermedia clave dentro del proyecto, ya que habilitó técnicamente la comunicación entre el nuevo frontend y el backend heredado sin exigir una reescritura completa del sistema. De este modo, el desarrollo posterior pudo orientarse de manera coherente no solo a la implementación de nuevas interfaces, sino también a su integración efectiva con los servicios necesarios para soportar autenticación, contenido, menús, noticias, eventos, calendarios, inscripciones y demás funcionalidades del portal.

6.8. Entregables

Como resultado de esta fase se generaron los siguientes documentos:

6.8.1. Requerimientos funcionales proporcionados por el administrador de COMA.

A partir del análisis del sistema actual COMA, de la revisión de la estructura de los portales de grupo y de la identificación de las necesidades funcionales del proyecto, se definieron los requerimientos funcionales del sistema. Estos requerimientos describen los servicios y comportamientos que la plataforma debe ofrecer tanto para usuarios públicos como para usuarios autenticados y administradores.

Tabla 1. Requerimientos funcionales del sistema por módulos

Módulo	**Descripción general**
Autenticación	Permite el acceso, cierre de sesión y gestión del usuario autenticado.
Navegación y contenido	Gestiona menús dinámicos, cambio de idioma, contenido por páginas y datos visuales del grupo.
Banners	Permite visualizar y administrar banners en la página principal.
Noticia	Gestiona el registro, edición, visualización y valoración de noticias.
Eventos	Permite crear, editar, listar e interactuar con eventos del grupo.
Calendarios	Gestiona calendarios y eventos asociados.

Foros	Soporta creación, visualización y participación en foros.
Inscripciones	Permite administrar convocatorias, postulaciones y seguimiento de interesados.
Correos	Gestiona el envío de correos individuales o masivos.
Configuración	Permite personalizar datos, apariencia y permisos del portal.
Producción intelectual	Permite consultar productos académicos y sus autores.

Tabla 2. Requerimientos funcionales del módulo de autenticación

Código	**Nombre**	**Descripción**
RF-AUT-001	Inicio de sesión	El sistema debe permitir iniciar sesión contra el backend del grupo mediante credenciales institucionales.
RF-AUT-002	Cierre de sesión	El sistema debe permitir cerrar sesión e invalidar el estado local del usuario.
RF-AUT-003	Obtención de usuario actual	El sistema debe permitir obtener el identificador del usuario autenticado para operar en módulos privados.

Tabla 3. Requerimientos funcionales del módulo de navegación y contenido

Código	Nombre	Descripción
RF-NAV-001	Menús dinámicos por grupo	El sistema debe renderizar los menús del grupo desde el backend y soportar submenús.
RF-NAV-002	Cambio de idioma	El sistema debe permitir alternar entre español e inglés cuando el grupo tenga inglés activo.
RF-CNT-001	Contenido por página	El sistema debe cargar contenido dinámico por página desde el backend.
RF-CNT-002	Autoría visual por bloques	El sistema debe permitir al administrador componer y editar las páginas de contenido del portal mediante un editor visual por bloques, con versiones independientes para español e inglés.
RF-GRP-001	Datos del grupo y tema	El sistema debe obtener datos del grupo para logo, título, subtítulo y tema visual.

Tabla 4. Requerimientos funcionales del módulo de banners

Código	Nombre	Descripción
RF-BAN-001	Banners visibles en inicio	El sistema debe listar y renderizar banners activos del grupo en la página principal.

RF-BAN-002	Administración de banners	El sistema debe permitir crear, editar y eliminar banners del portal.

Tabla 5. Requerimientos funcionales del módulo de noticias

Código	**Nombre**	**Descripción**
RF-NEW-001	Listado de noticias	El sistema debe listar noticias activas e inactivas del grupo y de escuelas asociadas.
RF-NEW-002	Crear noticia	El sistema debe permitir crear noticias del grupo.
RF-NEW-003	Editar noticia	El sistema debe permitir editar noticias existentes.

Tabla 6. Requerimientos funcionales del módulo de eventos

Código	**Nombre**	**Descripción**
RF-EVE-001	Listado de eventos	El sistema debe listar eventos en línea del grupo y de escuelas.
RF-EVE-002	Crear evento	El sistema debe permitir crear eventos del grupo.

RF-EVE-003	Editar evento	El sistema debe permitir editar eventos existentes.
RF-EVE-004	Interacción con eventos	El sistema debe permitir calificar, comentar y registrar asistencia a eventos.

Tabla 7. Requerimientos funcionales del módulo de calendarios

Código	**Nombre**	**Descripción**
RF-CAL-001	Crear calendario	El sistema debe permitir crear calendarios de grupo.
RF-CAL-002	Editar calendario	El sistema debe permitir editar calendarios existentes y su visibilidad.
RF-CAL-003	Listar calendarios y eventos	El sistema debe listar calendarios visibles y los eventos asociados.
RF-CAL-004	Gestión de eventos de calendario	El sistema debe permitir crear, editar y eliminar eventos de un calendario.

Tabla 8. Requerimientos funcionales del módulo de foros

Código	**Nombre**	**Descripción**

RF-FOR-001	Listar foros	El sistema debe listar los foros del grupo y los creados por el usuario.
RF-FOR-002	Crear foro	El sistema debe permitir crear foros por grupo o por tipo de usuario.
RF-FOR-003	Participación en foros	El sistema debe permitir responder, comentar y calificar mensajes de foro.

Tabla 9. Requerimientos funcionales del módulo de inscripciones

Código	**Nombre**	**Descripción**
RF-INS-001	Listado de inscripciones	El sistema debe listar inscripciones públicas, privadas y del usuario.
RF-INS-002	Crear inscripción	El sistema debe permitir crear inscripciones en variante general o UIS.
RF-INS-003	Editar inscripción	El sistema debe permitir editar inscripciones existentes.
RF-INS-004	Campos personalizados y requerimientos	El sistema debe permitir definir campos personalizados y requerimientos de inscripción.

RF-INS-005	Aplicar a inscripción	El sistema debe permitir postularse a una inscripción con validación de identidad.
RF-INS-006	Seguimiento y requisitos de interesados	El sistema debe permitir gestionar seguimiento y cumplimiento de requisitos.
RF-INS-007	Enviar correo al creador.	El sistema debe permitir enviar un correo al creador de una inscripción con adjuntos.

Tabla 10. Requerimientos funcionales del módulo de correos

Código	**Nombre**	**Descripción**
RF-COR-001	Correos masivos	El sistema debe permitir enviar correos a usuarios del grupo, escuela o UIS con filtros.

Tabla 11. Requerimientos funcionales del módulo de configuración

Código	**Nombre**	**Descripción**
RF-CFG-001	Configuración de datos del grupo	El sistema debe permitir actualizar título, subtítulo, idioma, logo y fondo de login.

RF-CFG-002	Configuración del tema	El sistema debe permitir personalizar la apariencia del portal mediante tokens de diseño (colores, escala de sombras y dimensiones), presets predefinidos e imagen de fondo del login, con vista previa en tiempo real del resultado.
RF-CFG-003	Datos de contacto institucional	El sistema debe permitir registrar y exponer en la presentación pública del portal los datos de contacto institucional del grupo, incluyendo teléfono, correo, dirección y página web.
RF-MNU-001	Configuración de menús y roles	El sistema debe permitir configurar visibilidad pública o privada y roles por menú y submenú.

Tabla 12. Requerimientos funcionales del módulo de producción intelectual

Código	**Nombre**	**Descripción**
RF-PROD-001	Producción intelectual	El sistema debe listar productos de producción intelectual por tipo y sus autores.

La organización de los requerimientos funcionales por módulos permite evidenciar que los portales de grupo del proyecto COMA no se limitan a la publicación de información, sino

que integran múltiples procesos de administración, interacción y personalización. Esta clasificación facilita la comprensión del alcance del sistema y sirve como base para el diseño, desarrollo y validación de la propuesta planteada en este trabajo de grado.

6.8.2. Casos de uso describiendo actores, objetivos y flujos principales.

Estas tablas agrupan, para cada actor, los casos de uso correspondientes y una breve descripción de su objetivo y del tipo de interacción.

Tabla 13. Casos de uso del actor: Usuario público

Código	Caso de uso	Objetivo	Flujo principal	Tipo de interacción
CU-PUB-001	Consultar información pública del grupo.	Permitir al usuario visualizar la información general y pública del portal del grupo.	El usuario ingresa al portal, navega por el menú principal y consulta banners, noticias, eventos, calendarios, contenido y producción intelectual visibles.	Consulta

CU-PUB-002	Cambiar idioma del portal	Permitir al usuario alternar entre los idiomas disponibles del portal.	El usuario selecciona el idioma, el sistema conserva la ruta actual y cambia el prefijo correspondiente del idioma.	Navegación
CU-PUB-003	Consultar contenido de páginas.	Permitir el acceso a páginas de contenido administradas desde el CMS del portal.	El usuario entra a una página del menú; el sistema consulta el contenido asociado y lo presenta en pantalla.	Consulta
CU-PUB-004	Ver detalle de noticia	Permitir la consulta completa de una noticia publicada.	El usuario selecciona una noticia del listado, el sistema carga su detalle y muestra la información asociada.	Consulta
CU-PUB-005	Consultar eventos públicos	Permitir la visualización de eventos publicados por el grupo o por escuelas asociadas.	El usuario entra al módulo de eventos, el sistema lista los eventos disponibles y permite abrir su detalle.	Consulta

CU-PUB-006	Consultar calendarios públicos	Permitir al usuario revisar calendarios y eventos visibles del grupo.	El usuario accede al módulo de calendarios; el sistema muestra los calendarios públicos y los eventos vinculados.	Consulta
CU-PUB-007	Consultar inscripciones públicas	Permitir al usuario revisar convocatorias o inscripciones abiertas.	El usuario accede al módulo de inscripciones; el sistema lista las convocatorias públicas y muestra su información.	Consulta
CU-PUB-008	Consultar producción intelectual.	Permitir al usuario visualizar los productos de producción intelectual del grupo y sus autores.	El usuario ingresa al módulo correspondiente; el sistema lista los productos por tipo y permite consultar sus autores.	Consulta

Tabla 14. Casos de uso del actor: Usuario autenticado

Código	**Caso de uso**	**Objetivo**	**Flujo principal**	**Tipo de interacción**

CU-AUT-001	Iniciar sesión	Permitir al usuario acceder a funcionalidades privadas del portal.	El usuario entra a la página de login, ingresa credenciales, el sistema solicita la llave pública, cifra la información, valida con el backend y habilita la sesión.	Autenticación
CU-AUT-002	Cerrar sesión	Permitir al usuario finalizar su sesión en el sistema.	El usuario selecciona la opción de cerrar sesión; el sistema invalida la sesión en backend y elimina el estado local.	Autenticación
CU-AUT-003	Crear noticia	Permitir al usuario autenticado registrar una noticia para el grupo.	El usuario abre el formulario, ingresa título, resumen, fechas, contenido y configuración de publicación, luego guarda la noticia y el sistema valida y registra la información.	Creación

CU-AUT-004	Editar noticia propia	Permitir al autor modificar una noticia previamente creada.	El usuario accede a sus noticias, selecciona una noticia propia, modifica sus datos y guarda los cambios.	Edición
CU-AUT-005	Crear evento	Permitir al usuario autenticado registrar un evento del grupo.	El usuario abre el formulario de evento, diligencia la información requerida, define publicación si aplica y guarda el registro.	Creación
CU-AUT-006	Editar evento propio	Permitir al autor modificar un evento previamente creado.	El usuario consulta sus eventos, selecciona uno propio, actualiza la información y guarda los cambios.	Edición
CU-AUT-007	Interactuar con eventos	Permitir al usuario comentar, calificar o registrar asistencia a un evento.	El usuario abre el detalle del evento y realiza la acción disponible; el sistema valida la sesión y registra la interacción.	Interacción

CU-AUT-008	Crear calendario	Permitir al usuario autenticado crear un calendario para organizar actividades.	El usuario accede al módulo de calendarios, diligencia título, descripción, permisos y color, y luego guarda el calendario.	Creación
CU-AUT-009	Editar calendario propio	Permitir al usuario modificar un calendario creado por él mismo.	El usuario consulta sus calendarios, selecciona uno propio, ajusta su configuración y guarda los cambios.	Edición
CU-AUT-010	Gestionar eventos de calendario	Permitir al usuario crear, editar o eliminar eventos dentro de un calendario.	El usuario selecciona un calendario, registra o modifica un evento y el sistema valida la información antes de guardarla.	Gestión
CU-AUT-011	Crear foro	Permitir al usuario abrir un espacio de discusión dentro del grupo.	El usuario entra al módulo de foros, define el título y mensaje inicial, y publica la conversación.	Creación

CU-AUT-012	Participar en foro	Permitir al usuario responder, comentar o calificar mensajes en un foro.	El usuario accede a un foro existente, escribe una respuesta o califica un mensaje y el sistema registra la participación.	Interacción
CU-AUT-013	Postularse a inscripción	Permitir al interesado aplicar a una convocatoria o inscripción.	El usuario selecciona una inscripción abierta, diligencia el formulario requerido, valida sus datos y envía la postulación.	Registro
CU-AUT-014	Enviar correo al creador de inscripción.	Permitir la comunicación con el responsable de una inscripción.	El usuario abre el módulo de comunicación, redacta asunto y mensaje, adjunta archivos permitidos y envía el correo.	Comunicación

Tabla 15. Casos de uso del actor: Administrador

Código	**Caso de uso**	**Objetivo**	**Flujo principal**	**Tipo de interacción**

CU-ADM-001	Crear banner	Permitir al administrador registrar banners visibles en la portada del portal.	El administrador abre el formulario de banner, adjunta imagen, completa los textos y guarda la información.	Configuración
CU-ADM-002	Editar o eliminar banner	Permitir al administrador actualizar o remover banners existentes.	El administrador consulta los banners activos, selecciona uno, modifica su información o ejecuta su eliminación.	Administración
CU-ADM-003	Configurar datos del grupo	Permitir la actualización de información general del portal, como títulos, subtítulos, logo o fondo.	El administrador ingresa al módulo de configuración, modifica los datos del grupo y guarda los cambios.	Configuración
CU-ADM-004	Configurar tema e idioma del portal.	Permitir la personalización visual extendida del portal (tokens de color, sombras,	El administrador accede al editor de apariencia, ajusta los tokens de diseño, observa el efecto en la vista previa en	Configuración

		dimensiones y fondo de login) y la activación del idioma inglés.	tiempo real, guarda la configuración y, de forma independiente, gestiona la activación del idioma inglés del portal.	
CU-ADM-005	Configurar menús, roles y permisos	Permitir definir visibilidad pública o privada de menús y asignar roles autorizados.	El administrador accede al módulo de menús, modifica la protección o los roles por menú y guarda la configuración.	Administración
CU-ADM-006	Gestionar contenido CMS	Permitir crear o modificar páginas de contenido del portal mediante el editor visual por bloques, con versiones independientes en español e inglés.	El administrador selecciona una página o crea una nueva, compone su contenido en el editor visual utilizando los bloques disponibles, alterna entre las versiones en español e inglés y guarda los cambios.	Gestión

CU-ADM-007	Crear inscripción administrativa	Permitir al administrador registrar inscripciones generales o tipo UIS.	El administrador ingresa los datos de la inscripción, selecciona escuelas si aplica, adjunta imagen y guarda la convocatoria.	Creación
CU-ADM-008	Editar inscripción administrativa	Permitir modificar convocatorias previamente registradas.	El administrador consulta las inscripciones, selecciona una, actualiza los campos requeridos y guarda los cambios.	Edición
CU-ADM-009	Definir campos personalizados y requerimientos.	Permitir parametrizar formularios y requisitos asociados a una inscripción.	El administrador entra al módulo de requerimientos, crea o ajusta campos personalizados y define sus condiciones.	Parametrización
CU-ADM-010	Realizar seguimiento a interesados.	Permitir gestionar el estado, cumplimiento de requisitos y	El administrador selecciona una inscripción, consulta los interesados y actualiza	Gestión

		seguimiento de postulantes.	seguimiento, requisitos o estado.	
CU-ADM-011	Enviar correos masivos	Permitir el envío de comunicaciones a usuarios del grupo, escuela o UIS.	El administrador selecciona destinatarios, redacta asunto y mensaje, adjunta archivos si aplica y ejecuta el envío.	Comunicación
CU-ADM-012	Gestionar contenidos creados por otros usuarios.	Permitir al administrador editar o eliminar contenidos del sistema por privilegio de rol.	El administrador consulta noticias, eventos, calendarios o inscripciones del grupo y ejecuta acciones de edición o eliminación cuando corresponda.	Administración

6.8.3. Diagramas de casos de uso que representan las interacciones con el sistema.

Los diagramas muestran, de forma visual, los tres tipos de usuario y los casos de uso que se derivan de los requerimientos, ofreciendo una visión general de cómo se relacionan los distintos actores con las funcionalidades del portal de grupos.

Figura 14. Diagrama de casos de uso — Usuario público

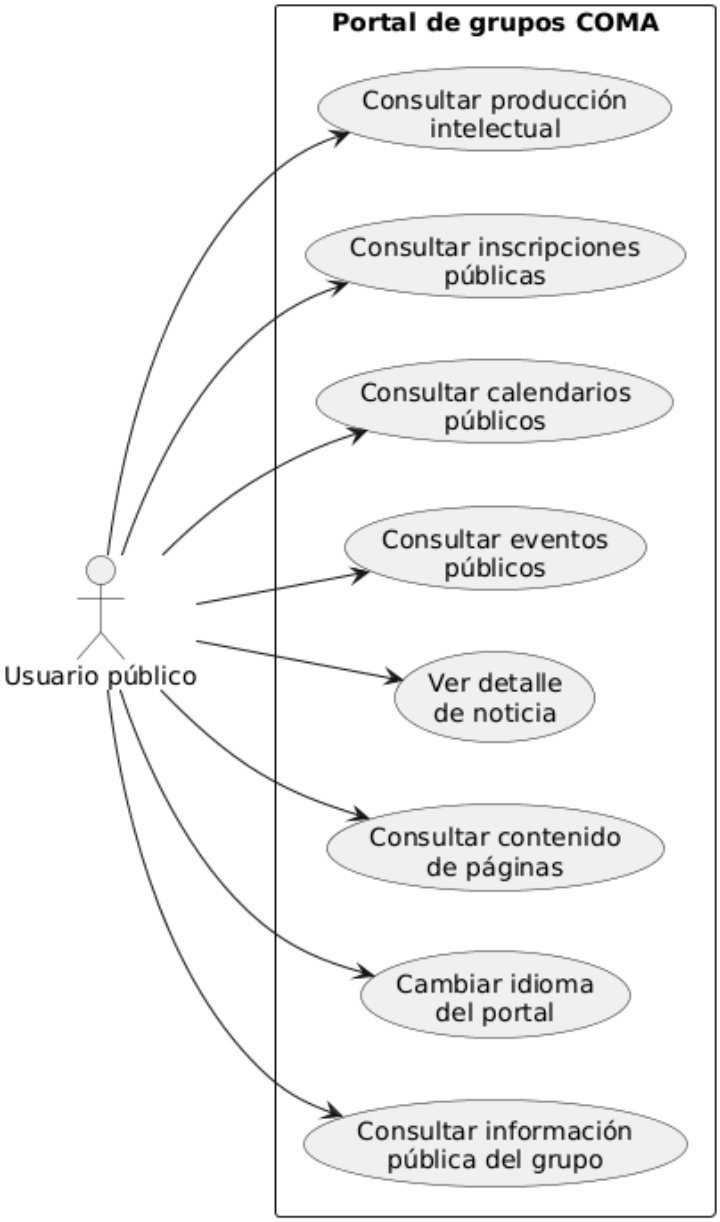


Figura 15. Diagrama de casos de uso — Usuario autenticado

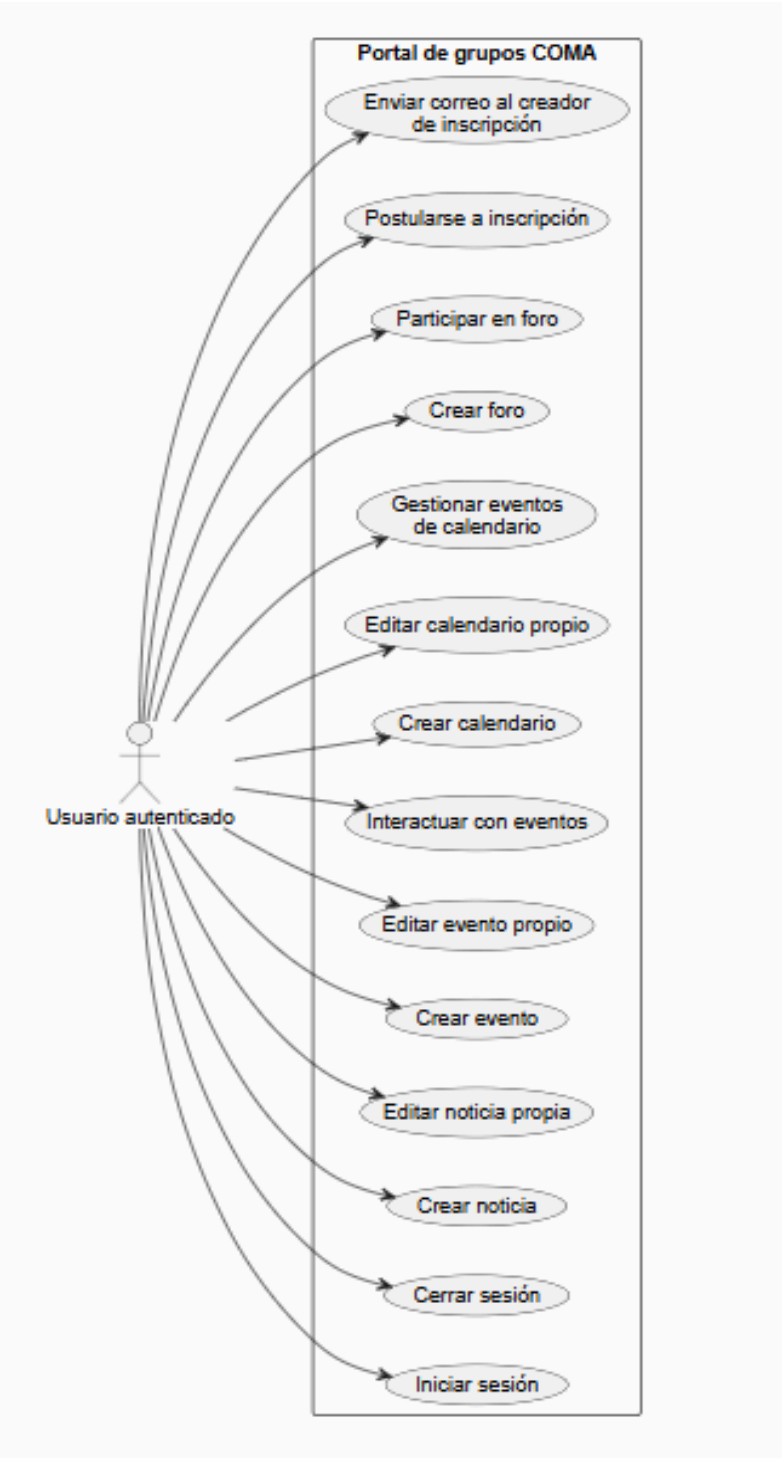
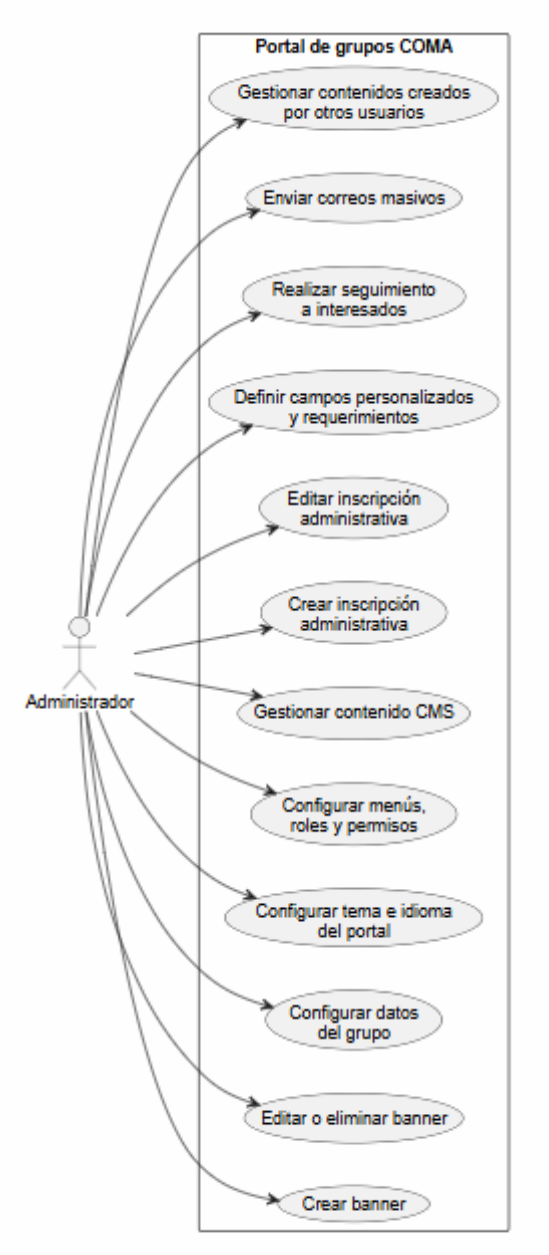


Figura 16. Diagrama de casos de uso — Administrador



7. Crear un diseño preliminar para las páginas

En esta fase se definió la apariencia y organización inicial del nuevo portal de grupos, tomando como referencia los requerimientos funcionales establecidos junto con el administrador de COMA, así como las observaciones obtenidas del análisis de la versión anterior en diferentes dispositivos (escritorio, tablet y móvil). Asimismo, se consideró la estrategia técnica definida en la fase previa, con el propósito de garantizar la coherencia entre el diseño del frontend y la arquitectura general del sistema. De esta manera, el diseño preliminar se desarrolló en concordancia con el plan de trabajo del proyecto, contemplando tanto las necesidades de los distintos tipos de usuario como la integración con el backend existente.

7.1. Elaborar un diseño que satisfaga los requerimientos.

En la etapa inicial de diseño, se prioriza la construcción de una propuesta que respondiera de manera directa a los requerimientos funcionales definidos con el cliente. Para ello, se tomó como base la estructura de navegación del sistema original, con el objetivo de conservar la familiaridad en la experiencia de uso para los usuarios habituales del portal.

A partir de estos lineamientos, se diseñó un prototipo que incluyó los elementos esenciales de la interfaz, tales como el encabezado (header), el formulario de inicio de sesión (login) y el pie de página (footer), garantizando la correcta identificación del grupo y el acceso a las distintas secciones del portal. Este diseño inicial no solo atendía aspectos visuales, sino

que también respondía a la necesidad de construir una interfaz alineada con una arquitectura desacoplada, en la cual la capa de presentación depende de servicios proporcionados por el backend y no de lógica embebida directamente en las vistas.

El prototipo se enfocó inicialmente en las vistas públicas y de miembros, organizando secciones como inicio, noticias, eventos, miembros y contacto, de acuerdo con los distintos perfiles de usuario identificados. No obstante, durante esta etapa se evidenció la necesidad de diferenciar con mayor claridad las funciones de gestión. Como resultado, se propuso la incorporación de un panel de administración (dashboard) dirigido al rol administrador, en el que se concentran las acciones relacionadas con la configuración, gestión y control de la información del portal. Esta decisión permitió alinear aún más el diseño con los requerimientos del cliente, al ofrecer una solución más organizada y funcional para la administración del sistema.

7.2. Adaptar el diseño a los diferentes dispositivos móviles (teléfono, tableta y computador).

Una vez definido el diseño funcional, se procedió a su adaptación para garantizar una correcta visualización e interacción en diferentes dispositivos. En esta etapa, se consolidó un diseño gráfico más detallado, en el que se seleccionaron colores para fondos y elementos interactivos, priorizando el contraste y la legibilidad con el fin de facilitar la identificación de acciones principales dentro de la interfaz.

Se realizaron ajustes específicos para dispositivos móviles, asegurando una distribución adecuada de los elementos que evitara problemas como el desbordamiento horizontal y

mejorará la navegación en pantallas reducidas. Asimismo, se procuró que los componentes más relevantes permanecieran visibles y accesibles, optimizando la experiencia del usuario independientemente del dispositivo utilizado.

Como parte de esta adaptación, se definió una organización basada en componentes reutilizables, tales como formularios para la creación y edición de información, tablas para la visualización de listados, tarjetas para la presentación de noticias y eventos, y elementos específicos para el dashboard de administración. Además, se incorporaron mecanismos de retroalimentación visual, que permiten informar al usuario sobre el resultado de sus acciones de manera clara e inmediata.

Este enfoque no solo responde a la necesidad de adaptación a múltiples dispositivos, sino que también se encuentra alineado con la implementación del frontend, favoreciendo la reutilización de componentes, la separación entre lógica y presentación, y la mantenibilidad del sistema.

7.3. Entregables

7.3.1. Histórico de diseños

Se muestra la evolución de los prototipos de la interfaz gráfica elaborados durante el proyecto, incluyendo capturas.

Figura 17. Primer diseño presentado en Figma

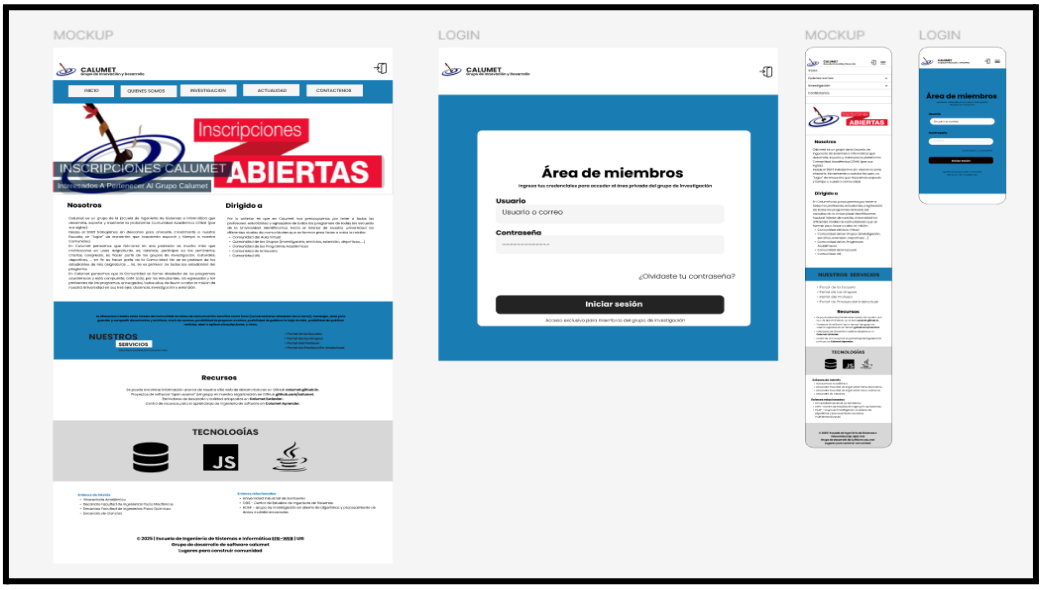


Figura 18. Página de inicio del segundo diseño planteado

Figura 19. Página de gestión de contenido dentro del panel de administración del segundo diseño planteado.

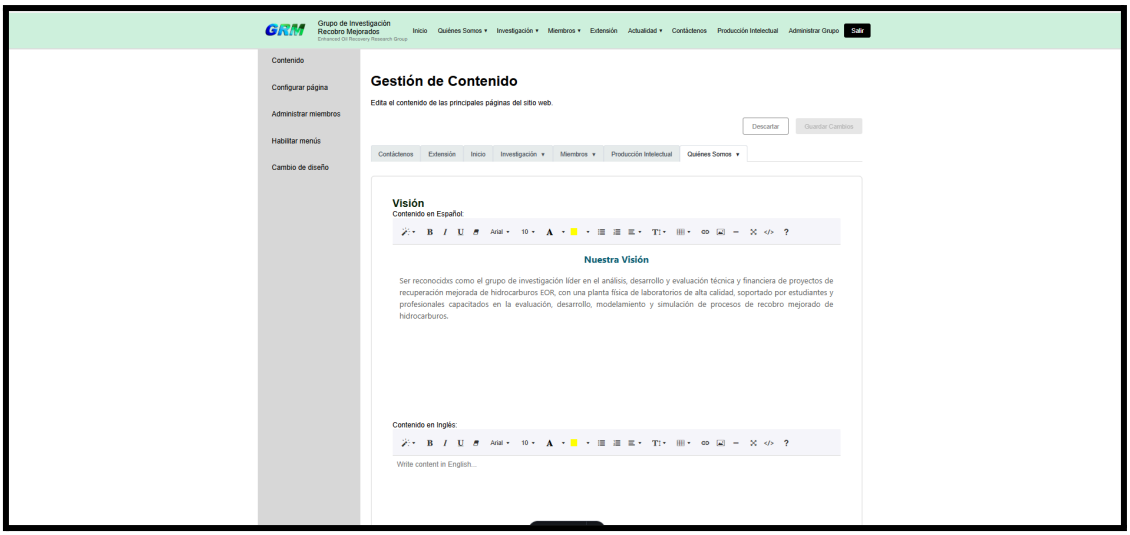


Figura 20. Página de login del segundo diseño planteado.

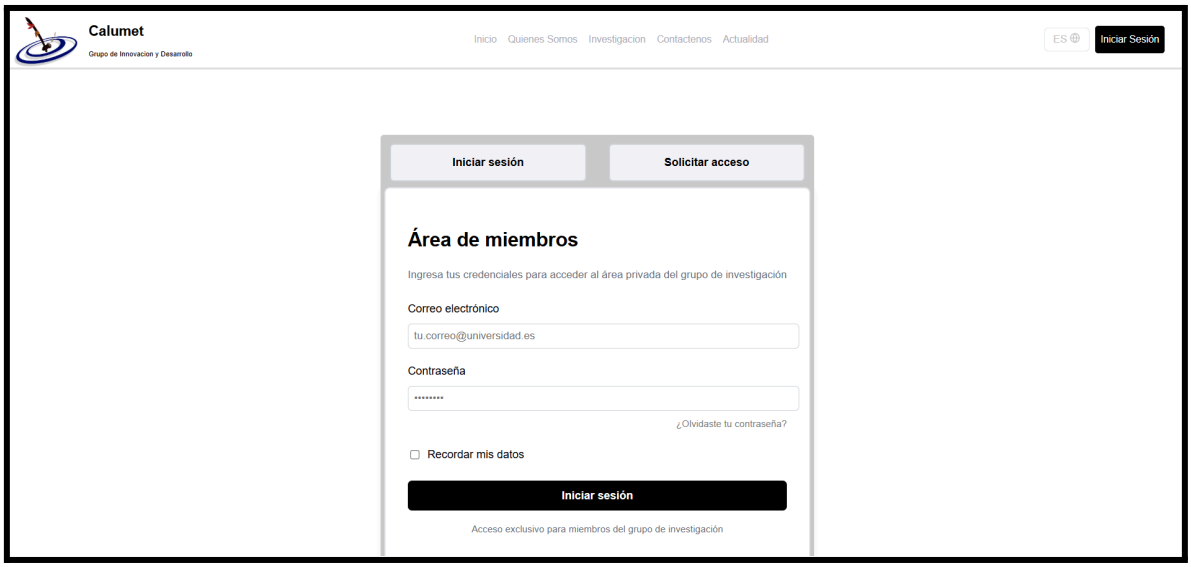


Figura 21. Página de inicio del tercer diseño planteado.

Figura 22. Página de login del tercer diseño planteado.

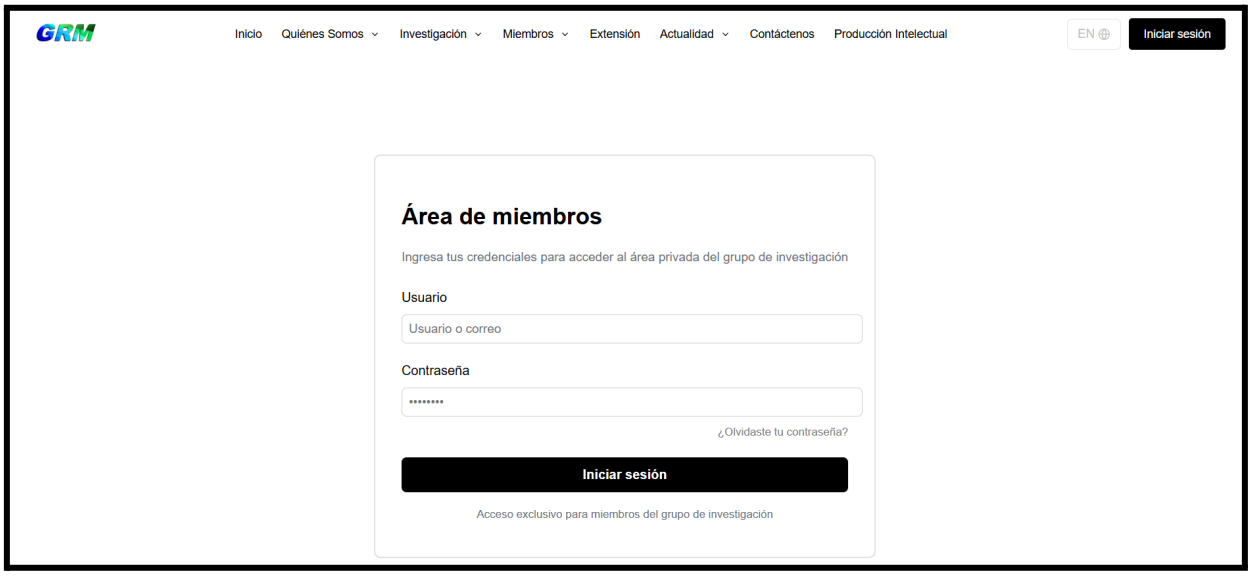


Figura 23. Página de gestión de contenido del tercer diseño planteado.

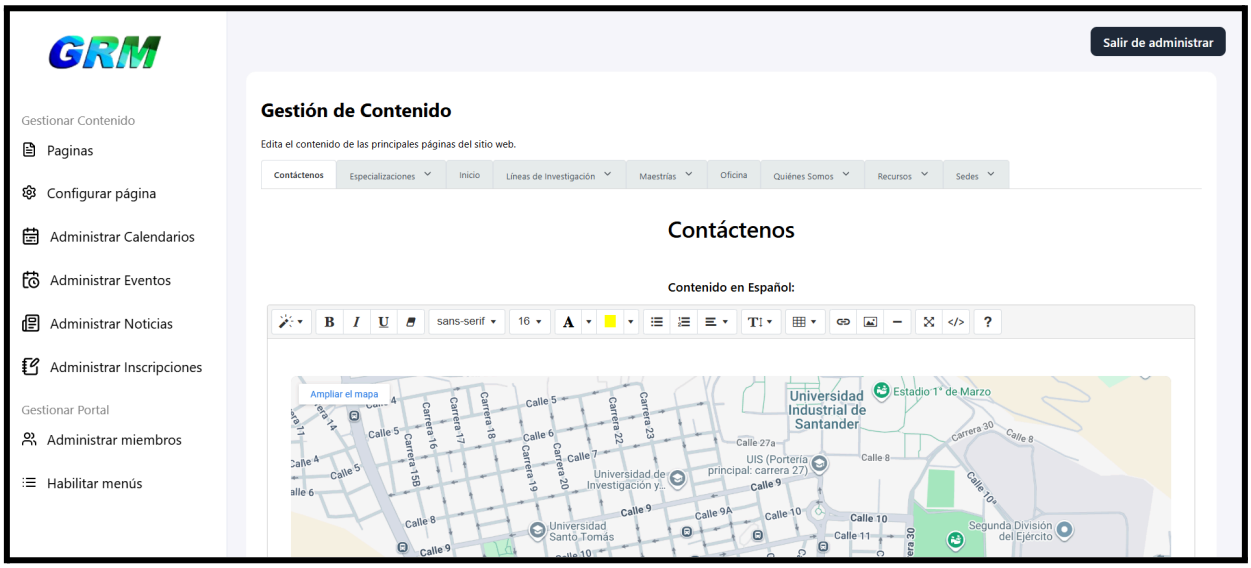


Figura 24. Página de inicio del diseño final.

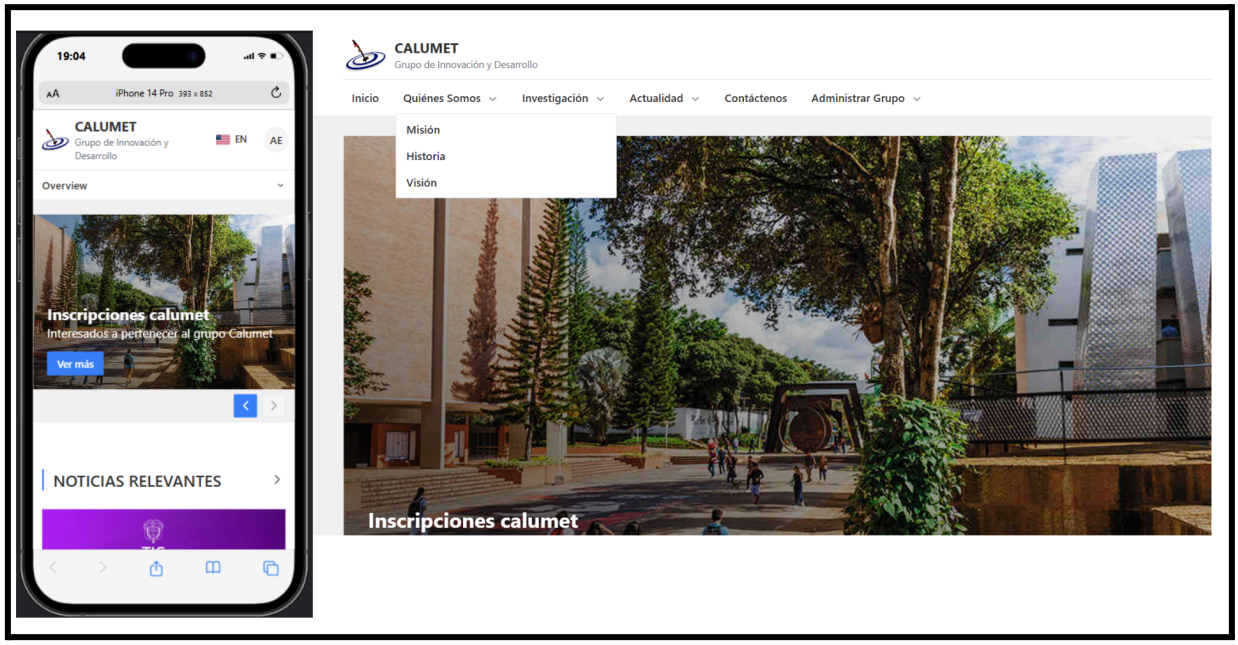
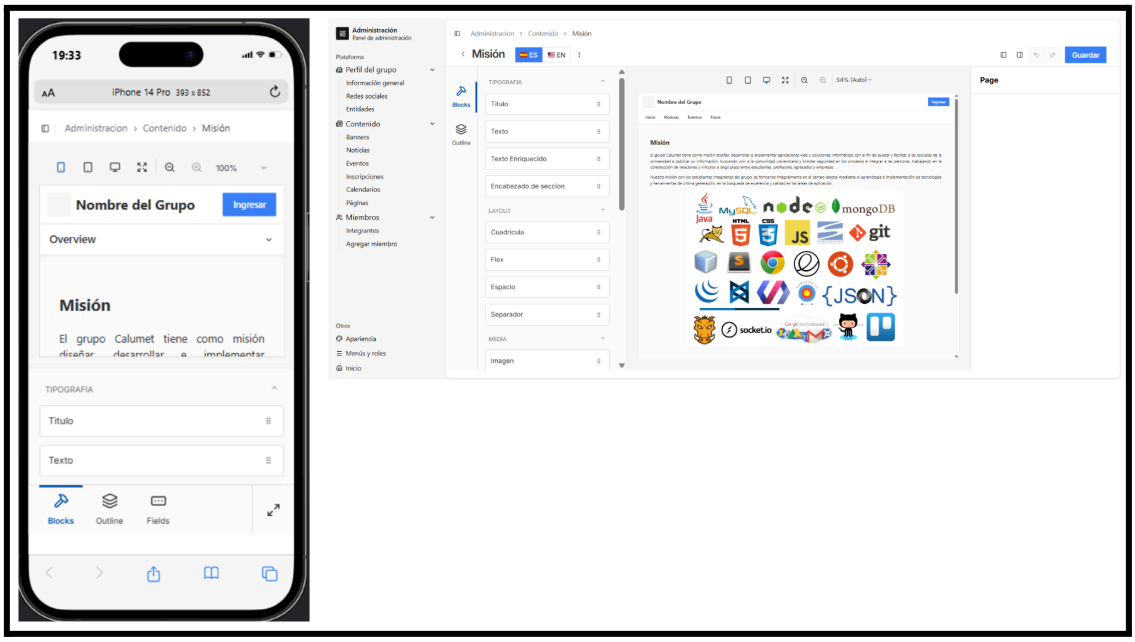


Figura 25. Página del diseño final para que el administrador pueda editar el contenido de las páginas.



8. Implementación del diseño planteado

En esta fase se materializó el diseño previamente aprobado, pasando de prototipos conceptuales a un portal de grupos completamente funcional. En coherencia con el plan de trabajo y la estrategia definida en la fase anterior, la implementación no se limitó al desarrollo visual del frontend, sino que incluyó la validación progresiva mediante retroalimentación continua. Este proceso se desarrolló de manera iterativa, lo que permitió refinar tanto los aspectos técnicos como los funcionales a medida que se evaluaban los resultados con el administrador de COMA.

8.1. Aplicar el diseño aprobado.

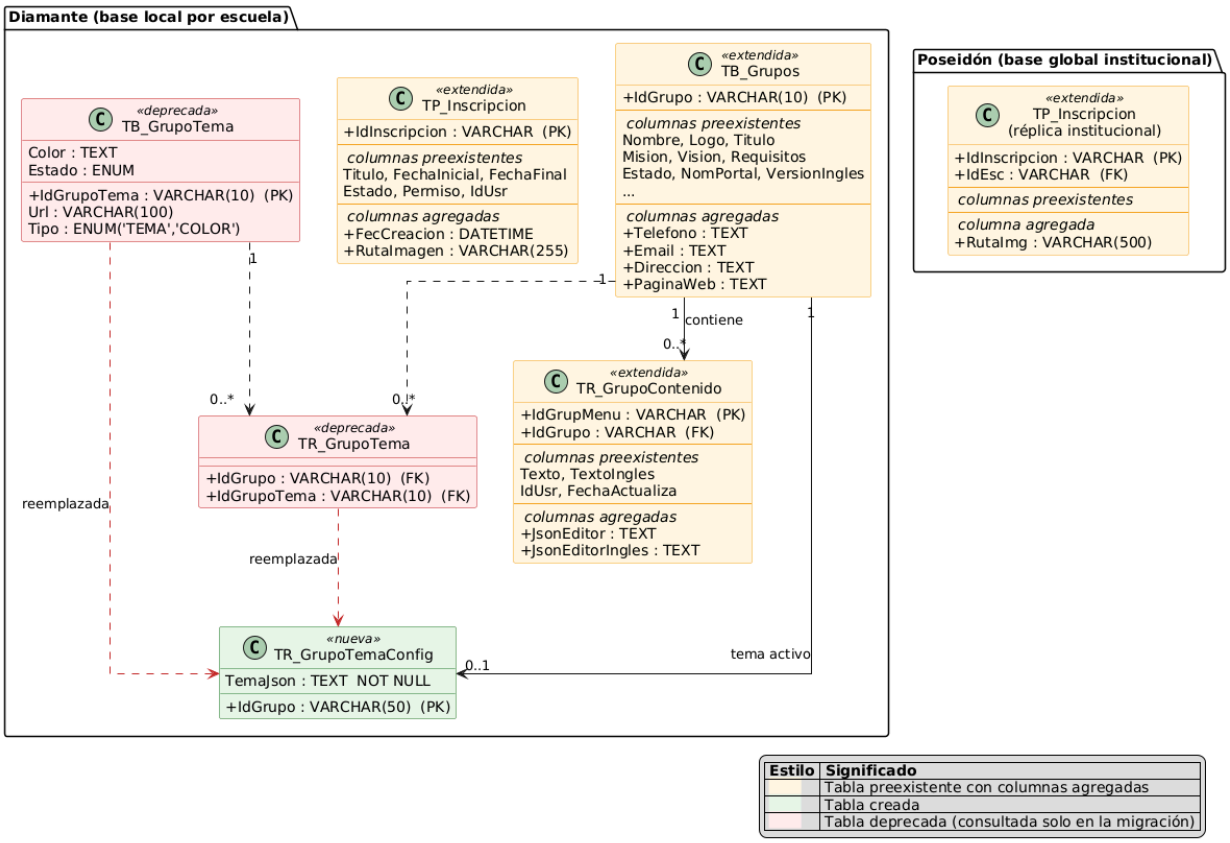
La primera tarea consistió en implementar con React, TypeScript, Vite, Hono y las herramientas del ecosistema de CALUMET las páginas definidas en el diseño preliminar: la página de inicio del grupo, las secciones de noticias y eventos, el apartado de miembros, la página de contacto, el dashboard de administración, entre otras. Para cada una de estas vistas se construyeron las estructuras de layout acordadas en la fase de diseño, incorporando encabezados, menús, bloques de contenido y áreas específicas para la interacción de los diferentes tipos de usuario.

En paralelo con esta construcción de la capa de presentación, se implementó la adecuación del backend necesaria para cumplir la estrategia definida en el capítulo anterior. Para ello, se crearon y ajustaron controladores basados en servlets, se definieron endpoints para las operaciones requeridas por el portal y se reorganizó el módulo con el propósito de separar de manera más clara el modelo, la lógica de negocio y los controladores. Estos endpoints pasaron a constituir el punto de acceso del frontend hacia la información almacenada en las bases de datos Diamante y Poseidón. Como parte de esta mejora en la mantenibilidad y en la claridad arquitectónica, también se dejó registro técnico de los servicios implementados para facilitar su comprensión y futura evolución.

De forma complementaria, fue necesario realizar ajustes puntuales sobre el esquema de base de datos para dar soporte a las capacidades nuevas acordadas con el administrador. En la

tabla TB_Grupos se agregaron las columnas Teléfono, Email, Dirección y PáginaWeb, que alimentan el espacio de contacto institucional expuesto en la presentación pública del portal. En TP_Inscripcion se incorporaron las columnas FecCreacion y RutaImagen, con el propósito de permitir el ordenamiento cronológico de las inscripciones y su presentación en tarjetas con imagen asociada, en lugar del listado exclusivamente textual del portal anterior. En TR_GrupoContenido se añadieron las columnas JsonEditor y JsonEditorIngles, destinadas a persistir el estado del editor visual por bloques para las versiones en español e inglés de cada página de contenido. Adicionalmente, se creó la tabla TR_GrupoTemaConfig, con el identificador de grupo como clave primaria y un campo destinado a almacenar la configuración extendida del tema en formato estructurado. Esta tabla reemplaza a las dos estructuras del sistema anterior que almacenaban únicamente un par de colores para el encabezado, las cuales quedaron deprecadas y se consultaron una sola vez durante la migración inicial con el fin de sembrar, para cada portal preexistente, un tema de base coherente con su configuración previa y evitar así la aparición de portales sin estilo tras el despliegue. En conjunto, estos cambios no constituyeron una modificación aislada del esquema de datos, sino la adecuación necesaria para que la nueva capa de presentación pudiera operar de manera coherente con la refactorización del backend y con las capacidades incorporadas al portal.

Figura 26. Modificaciones en el esquema de base de datos requeridas para la integración del nuevo frontend del portal de grupos.



Una vez establecida esta base de integración, los componentes de interfaz definidos previamente se llevaron a código como elementos reutilizables. Se desarrollaron tablas para organizar listados de información, formularios para la creación y edición de registros, y elementos de retroalimentación como toasts y modales de confirmación, que informan al usuario si una operación se realizó correctamente o si ocurrió algún problema. De esta forma, la implementación del frontend y la adecuación del backend avanzaron de manera articulada, conforme a lo previsto en el plan del proyecto.

Con esta estructura se superó el esquema anterior basado en JSP con invocaciones directas a clases Java. Las vistas del nuevo frontend se limitan ahora a presentar la información y a reaccionar a las acciones del usuario, mientras que la lógica de negocio y el acceso a datos se concentran en controladores y servicios. Esta separación clara facilita el mantenimiento, reduce el acoplamiento entre presentación y backend, y hace que el código sea más comprensible para futuros desarrollos y nuevas integraciones.

8.2. Aplicar el marco de desarrollo basado en prototipos evolutivos

El desarrollo del portal se llevó a cabo siguiendo un enfoque de prototipos evolutivos, lo que permitió construir y validar progresivamente la solución. En una primera iteración se desarrolló una versión mínima funcional del sistema, que incluía los principales flujos de navegación y las operaciones básicas para la administración de contenidos, como la gestión de noticias y eventos.

Esta versión inicial permitió verificar tanto la interacción de la interfaz como el correcto funcionamiento de la comunicación entre el frontend, los servicios del backend y las bases de datos. A partir de esta base, se realizaron iteraciones sucesivas en las que se incorporaron mejoras tanto a nivel visual como funcional.

Durante estas iteraciones, se optimizó el diseño responsivo para distintos tamaños de pantalla, se organizaron secciones del dashboard con el fin de mejorar la usabilidad y se refinaron los componentes de formularios y tablas mediante la incorporación de validaciones y mensajes de error más claros. Cada ciclo de desarrollo concluía con pruebas internas y sesiones

de validación con el administrador de COMA, en las cuales se evaluaban aspectos como la claridad de la interfaz, la facilidad de uso y la organización de la información.

La retroalimentación obtenida en estas revisiones se integraba en la siguiente iteración, manteniendo así un proceso continuo de mejora. Este enfoque permitió asegurar que el producto final no solo cumpliera con los requerimientos definidos, sino que también ofreciera una experiencia de usuario más intuitiva y coherente.

8.3. Implementar el sistema de control de versiones.

Con el fin de garantizar la calidad del software y mantener un registro organizado de la evolución del proyecto, se implementó un sistema de control de versiones basado en Git. Este permite gestionar de manera estructurada los cambios realizados durante el desarrollo, facilitando el seguimiento de modificaciones y la colaboración en el proyecto.

Se trabajó mediante el uso de ramas específicas para nuevas funcionalidades o cambios relevantes, lo que permitió aislar desarrollos, realizar pruebas controladas y comparar versiones antes de su integración. Esta práctica contribuyó a reducir errores y a mantener la estabilidad del sistema.

Adicionalmente, se llevaron a cabo revisiones periódicas del código, orientadas a verificar la coherencia en la estructura del proyecto y el cumplimiento de los principios arquitectónicos definidos. En estas revisiones se comprobó, por ejemplo, la correcta separación entre páginas, componentes, controladores y servicios; el uso adecuado de componentes reutilizables en las vistas; y la correcta implementación de los servicios de acceso a datos desde el backend, evitando su uso directo desde la capa de presentación.

Este proceso también permitió identificar oportunidades de mejora, como la eliminación de lógica duplicada, la simplificación de funciones y la corrección de inconsistencias menores que podrían afectar la mantenibilidad del sistema a largo plazo. En consecuencia, el código final no solo refleja fielmente el diseño aprobado, sino que se encuentra preparado para futuras modificaciones, integraciones y escalabilidad dentro del ecosistema del sistema COMA.

8.4. Entregables

En correspondencia con lo planteado en la fase de implementación del plan de trabajo, los entregables de esta etapa se concretan en los siguientes artefactos:

8.4.1. Capturas del diseño final del portal de grupos

Conjunto de imágenes que muestra la implementación alcanzada en las principales vistas del portal, incluyendo secciones públicas y administrativas en escritorio y dispositivos móviles. Estas capturas permiten evidenciar la aplicación del diseño aprobado, la adaptación responsiva de la interfaz y la materialización visual del desarrollo realizado.

Las capturas de las vistas implementadas se encuentran en el Apéndice B.

8.4.2. Manual de usuario del portal de grupos

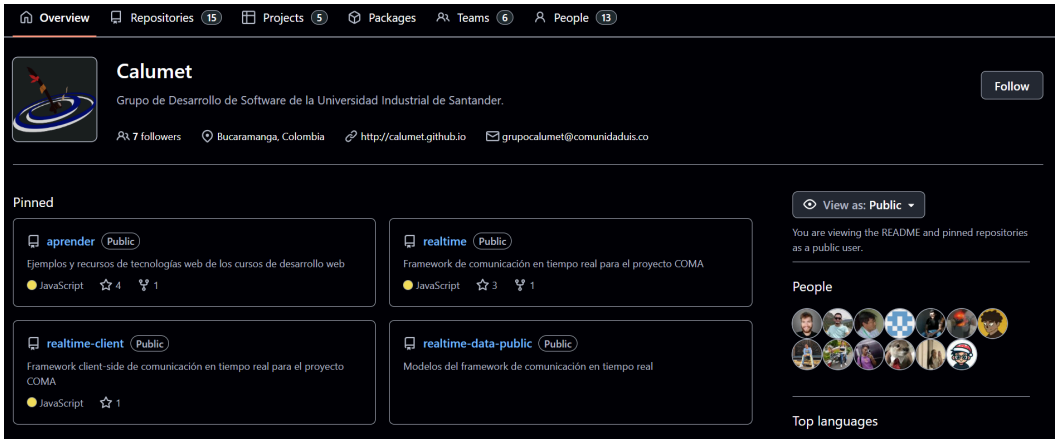
Documento dirigido a los tres tipos de usuario (público, miembros y administradores) donde se explica, con capturas del diseño final, cómo navegar por el portal, cómo consultar noticias y eventos, y cómo utilizar el dashboard para crear, editar y eliminar contenidos.

El manual de usuario se encuentra en el Apéndice C

8.4.3. Código Fuente

El código fuente de la implementación se encuentra gestionado mediante control de versiones, lo que permitió registrar iteraciones, revisar cambios y mantener trazabilidad durante el desarrollo. El repositorio es privado y pertenece al Grupo de Investigación CALUMET, organización de la Universidad Industrial de Santander. Su acceso se encuentra restringido al entorno institucional correspondiente.

Figura 27. Página principal de la Organización Calumet



9. Pruebas de funcionamiento y rendimiento

El presente capítulo describe las actividades de validación realizadas sobre la nueva versión del portal de grupos, organizadas en dos ejes complementarios: la verificación integral de los requerimientos funcionales y los casos de uso mediante pruebas de integración ejecutadas con usuarios de prueba, y la evaluación técnica del sistema mediante mediciones de tiempos de respuesta, tiempos de carga y consumo de recursos. Las secciones que siguen describen cómo se llevaron a cabo ambas actividades; el análisis global de los resultados y su interpretación frente a los objetivos del proyecto se desarrolla en el capítulo siguiente.

9.1 Pruebas de integración y compatibilidad

Las pruebas de integración tuvieron como propósito verificar que cada requerimiento funcional y caso de uso definidos en la fase de levantamiento pudiera ejecutarse correctamente dentro del portal desplegado, y que esa ejecución fuera viable tanto desde computadores de escritorio como desde dispositivos móviles. La actividad combina, de este modo, una validación funcional del sistema con una validación de compatibilidad de la capa de presentación frente a los dos formatos de pantalla más comunes en el uso real de la plataforma.

Para su ejecución se construyó una matriz de prueba a partir de los requerimientos funcionales del apartado 6.8.1 y los casos de uso del apartado 6.8.2, distribuidos entre los tres actores del sistema, con ocho casos del usuario público, catorce del usuario autenticado y once

del administrador. La matriz, montada sobre una hoja de cálculo compartida, registra para cada artefacto su identificador, el flujo esperado, el actor responsable y los campos destinados a consignar el resultado de la verificación. El portal utilizado fue una instancia desplegada sobre un entorno equivalente al de producción, conectada a los servicios reales del monolito y a las bases de datos institucionales, y poblada previamente con datos representativos, entre ellos noticias, eventos, calendarios, inscripciones y páginas de contenido, de modo que los auditores pudieran ejercer los flujos sobre información realista.

La ejecución se asignó a siete auditores externos al equipo de desarrollo, sin contacto previo con la plataforma, con el fin de obtener una lectura objetiva de la facilidad de uso desde la perspectiva de un usuario nuevo. Cada auditor, tras recibir una breve explicación del propósito de la prueba, de la escala de calificación y de las credenciales necesarias para asumir el rol que le correspondiera, ejecutó los artefactos asignados sobre el portal primero desde un dispositivo móvil y luego desde un computador de escritorio. Para cada ejecución se registraron dos valores en la matriz: el resultado de la operación en forma dicotómica y una calificación de dificultad percibida en una escala de uno a cinco, donde uno corresponde a una ejecución directa y sin fricción y cinco a una ejecución confusa o que exigió un esfuerzo considerable. La calificación de dificultad no evalúa el resultado funcional del sistema, sino la facilidad de uso subjetiva de la operación, y se incorporó al protocolo con el propósito de detectar secciones de la interfaz que, aun funcionando correctamente, pudieran presentar oportunidades de mejora en la experiencia de usuario.

La versión consolidada de la matriz, que contiene para los 68 artefactos probados el auditor responsable, el actor asumido, el resultado en cada medio y la calificación de dificultad correspondiente, constituye el insumo sobre el que se desarrolla la lectura de resultados de funcionamiento y compatibilidad en el capítulo 10, y se preserva de manera íntegra en el anexo dedicado al reporte de pruebas de integración.

9.2 Pruebas de rendimiento y uso de recursos

Las pruebas de rendimiento se orientaron a evaluar el comportamiento del portal bajo distintas condiciones de carga, cuantificar los recursos que demanda su operación y disponer de una línea de comparación directa con la versión anterior del sistema. La evaluación se estructuró sobre tres ejes: la experiencia percibida por el usuario final en la carga inicial de las páginas, la capacidad de respuesta del servidor ante concurrencia y el consumo de memoria y procesamiento del proceso responsable del renderizado. La Tabla 3 resume los escenarios ejecutados y sus parámetros de configuración.

Tabla 16. Resume los escenarios evaluados durante las pruebas con los requerimientos.

Eje	Escenario	Herramienta	Parámetros principales
Experiencia del cliente	Auditoría sobre 9 rutas del portal nuevo y 7	Lighthouse 13.1.0	Cuatro condiciones de uso (escritorio y móvil, con y sin caché del navegador); tres

	equivalentes del portal anterior		corridas por combinación y mediana reportada
Servidor	Estrés escalonado	autocannon 8.0.0	25, 50, 75 y 100 conexiones concurrentes, 2 minutos por escalón
Servidor	Carga sostenida	autocannon 8.0.0	20 conexiones concurrentes durante 20 minutos
Servidor	Arranque en frío	autocannon 8.0.0	Un request inmediatamente después del reinicio, seguido de un segundo request para comparación
Servidor	Comportamiento multi-tenant	autocannon 8.0.0	10 conexiones durante 3 minutos, alternando peticiones entre dos portales
Backend	Latencia del monolito en aislado	autocannon 8.0.0	10 conexiones contra el API del monolito durante 2 minutos
Comparativo	Línea base del portal anterior	autocannon 8.0.0	5, 10 y 25 conexiones, 1 minuto por escalón, sobre producción

Recursos	Muestreo de memoria residente y uso de CPU del proceso	pidusage 4.0.1	Captura a 1 Hz en paralelo con los escenarios de estrés y carga sostenida

9.2.1. Pruebas de experiencia del cliente

Las auditorías de cliente se realizaron con Lighthouse, reproduciendo cuatro condiciones de uso habituales. Las dos condiciones de escritorio corresponden a una pantalla estándar, con y sin caché del navegador, y sin estrangulamiento adicional. Las dos condiciones móviles corresponden a una pantalla de teléfono, con y sin caché del navegador, y con un estrangulamiento de CPU de cuatro veces y una red Slow 4G simulada; esta última configuración es la que se utiliza como referencia para el cálculo de los Core Web Vitals. Sobre cada combinación de ruta y condición se ejecutaron tres corridas y se reporta la mediana, con el fin de atenuar variaciones puntuales de medición. Las rutas evaluadas cubren la página de inicio, los listados y el detalle de noticias y eventos, una página de contenido administrable, el formulario de inicio de sesión y dos rutas equivalentes en un segundo portal, lo que permite verificar la consistencia del rendimiento entre portales distintos.

Los resultados obtenidos en la condición móvil con caché vacía, que representa el escenario más exigente para el sistema, se resumen en la Tabla 4 sobre la página de inicio, que concentra la mayor parte del contenido público y actúa como indicador representativo del portal.

Tabla 17. Métricas de experiencia del cliente sobre la página de inicio en condición móvil exigente

Métrica	***Portal anterior***	***Portal nuevo***	***Variación***
Largest Contentful Paint (LCP)	*5,66 s*	*3,34 s*	*-40,9 %*
First Contentful Paint (FCP)	*4,37 s*	*3,27 s*	*-25,3 %*
Puntaje Lighthouse de rendimiento	*57*	*85*	*+49,1 %*
Puntaje Lighthouse de accesibilidad	*41*	*96*	*+134,1 %*
Peso total de la página	*4 824 KB*	*1 879 KB*	*-61,0 %*

9.2.2. Pruebas de carga y uso de recursos del servidor

Las pruebas de carga del servidor se ejecutaron con autocannon desde el mismo equipo sobre el que corre el frontend, con el propósito de eliminar la latencia de red como variable y obtener mediciones limpias del comportamiento del renderizado y de las capas de caché. Sobre esa premisa se ejecutaron cinco escenarios: un estrés escalonado para identificar el punto de saturación del proceso, una carga sostenida de veinte minutos para observar la estabilidad de memoria y throughput bajo carga prolongada, un escenario de arranque en frío para medir el

costo del primer request tras un reinicio, un escenario multi-tenant para verificar que no exista interferencia entre las entradas de caché de portales distintos, y una línea base del API del monolito en aislado. En paralelo, un muestreador basado en pidusage registró cada segundo la memoria residente y el uso de procesamiento del proceso encargado del renderizado. Adicionalmente, se ejecutó una línea base del portal anterior sobre el entorno de producción, con un escalón máximo de 25 conexiones concurrentes para no afectar a usuarios reales.

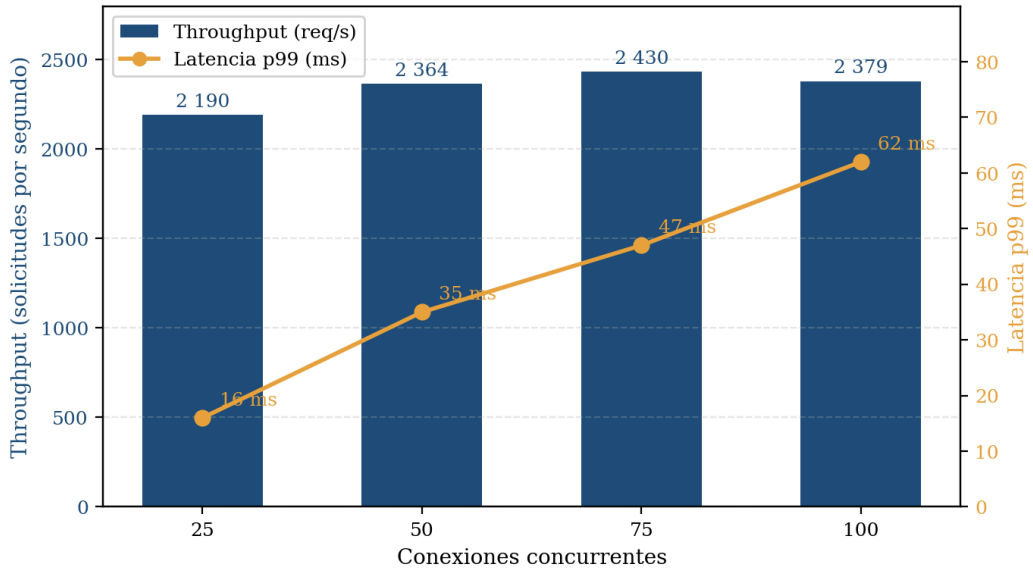
El escenario de estrés escalonado se ejecutó con cuatro niveles de concurrencia (25, 50, 75 y 100 conexiones). El throughput se estabilizó entre 2 190 y 2 430 solicitudes por segundo a partir de 25 conexiones, lo que indica que el techo de capacidad del proceso se alcanza en ese rango y no crece al aumentar la concurrencia, como corresponde a un sistema limitado por CPU en un único núcleo. La latencia crece de forma aproximadamente lineal con las conexiones, pero se mantiene por debajo del percentil 99 de 62 ms incluso a 100 conexiones, sin errores en el total de 1 123 325 solicitudes procesadas durante el escalonado. La Tabla 5 recoge los valores obtenidos en cada escalón.

Tabla 18. Throughput y latencia del escenario de estrés escalonado

Conexiones concurrentes	***Throughput***	***Latencia p50***	***Latencia p90***	***Latencia p99***	***Errores***
25	*2 190 req/s*	*10 ms*	*12 ms*	*16 ms*	*0*
50	*2 364 req/s*	*20 ms*	*24 ms*	*35 ms*	*0*

75	*2 430 req/s*	*29 ms*	*34 ms*	*47 ms*	*0*
100	*2 379 req/s*	*41 ms*	*46 ms*	*62 ms*	*0*

Figura 28. Barras de throughput con línea de latencia p99 creciendo linealmente.



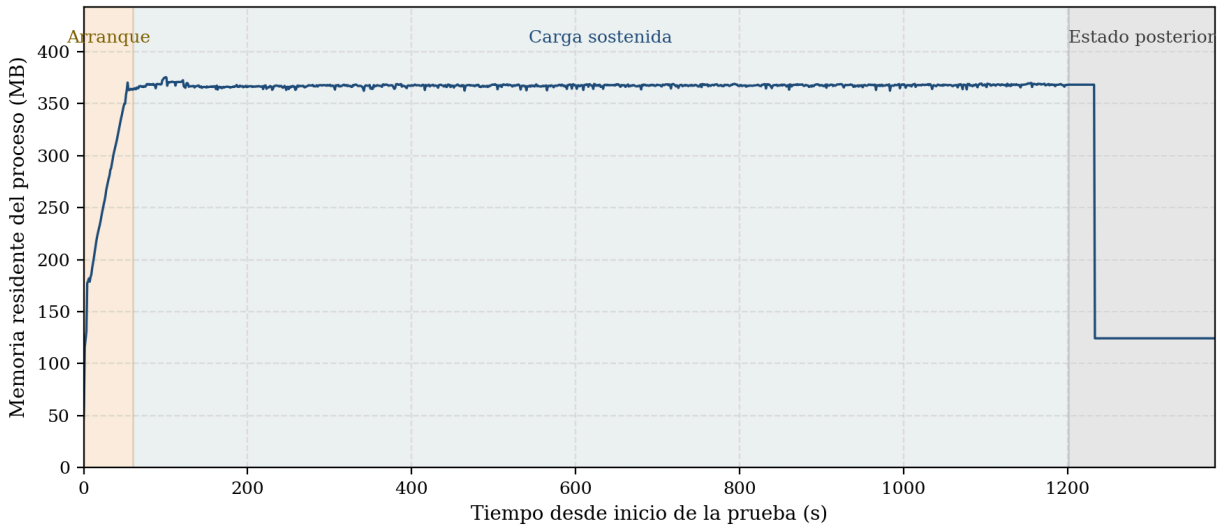
El muestreador de recursos, ejecutado en paralelo con los escenarios anteriores, permitió observar la carga que el proceso impone sobre el servidor. La Tabla 6 resume los valores agregados de CPU y memoria residente durante el estrés escalonado y durante la prueba de carga sostenida. Los valores de CPU superiores al 100 % se deben a que, aunque el hilo principal de JavaScript está limitado a un único núcleo, el proceso completo utiliza núcleos adicionales para operaciones internas como la recolección de memoria y la compresión HTTP.

Tabla 19. Uso de CPU y memoria del proceso durante los escenarios de carga

Escenario	***CPU promedio***	***CPU p95***	***CPU máximo***	***RSS promedio***	***RSS máximo***
Estrés escalonado	*97,5 %*	*125,0 %*	*157,9 %*	*294,1 MB*	*363,9 MB*
Carga sostenida (20 min)	*66,5 %*	*121,8 %*	*148,4 %*	*274,4 MB*	*390,7 MB*

La prueba de carga sostenida durante veinte minutos, con veinte conexiones concurrentes, procesó más de 2,4 millones de solicitudes sin errores. El análisis de la serie temporal de memoria residente permite distinguir tres fases: un arranque en el que la memoria del proceso asciende de 45 MB a alrededor de 365 MB durante los primeros sesenta segundos, correspondiente al llenado del heap del motor V8 y de los buffers HTTP; una fase de carga sostenida en la que la memoria se mantiene estable entre 365 y 368 MB, con una variación máxima de 3 MB; y una fase posterior a la prueba en la que, al detenerse la carga, la memoria residente desciende hasta 124 MB, lo que evidencia que el recolector de memoria libera los buffers temporales y confirma la ausencia de retención indebida de objetos.

Figura 29. Serie temporal de RSS aislada correctamente a la ventana de soak.



Los escenarios de arranque en frío y de comportamiento multi-tenant permitieron caracterizar dos aspectos operacionales del servidor. En el arranque en frío, el primer request tras el reinicio del proceso tarda casi cuatro segundos, ya que desencadena la consulta al monolito para resolver el catálogo de portales de las 32 escuelas; el segundo request, ejecutado inmediatamente después del primero, se resuelve en siete milisegundos, lo que confirma que el catálogo queda en memoria y la penalidad sólo se incurre una vez por reinicio o despliegue. En el escenario multi-tenant, el servidor alternó solicitudes entre dos portales distintos durante tres minutos y entregó una latencia del percentil 99 uniforme, de entre ocho y nueve milisegundos, para todas las rutas evaluadas, sin diferencias significativas entre tenants, lo que confirma que las entradas de caché de un portal no interfieren con las del otro. La Tabla 7 reúne los valores principales de ambos escenarios.

Tabla 20. Resultados del arranque en frío y del escenario multi-tenant

Escenario	Indicador	Valor
Arranque en frío	Latencia del primer request	3 982 ms
Arranque en frío	Latencia del segundo request	7 ms
Multi-tenant	Throughput agregado	2 157 req/s
Multi-tenant	Throughput por ruta (seis rutas distribuidas entre dos portales)	359 req/s
Multi-tenant	Latencia p99 por ruta	8 a 9 ms
Multi-tenant	Errores	0

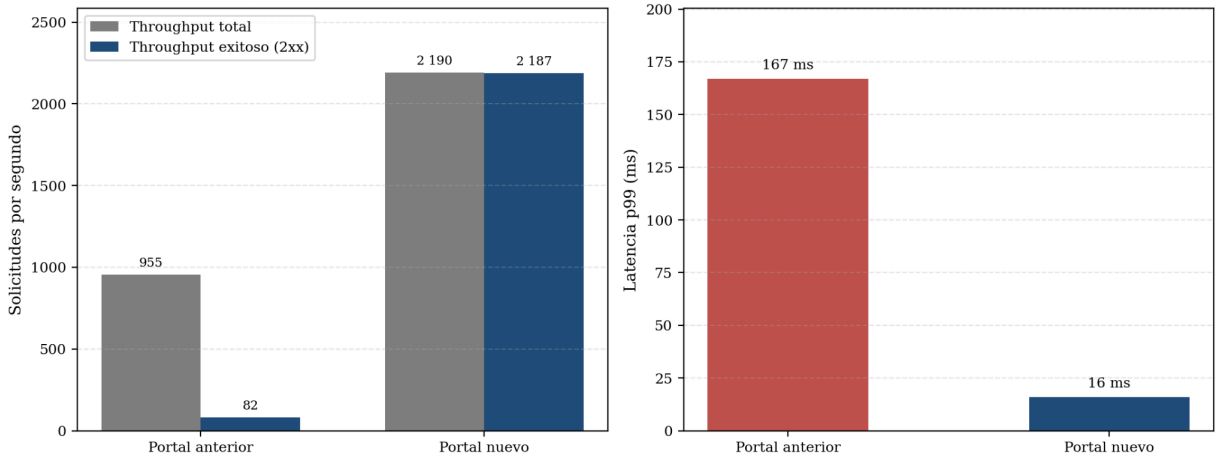
Adicionalmente, la línea base del backend, medida contra el API del monolito en aislado, obtuvo una latencia del percentil 50 de 3 ms y del percentil 99 de entre 7 y 9 ms sobre los endpoints consumidos por el frontend, con un throughput combinado de 3 242 solicitudes por segundo. Comparando esta latencia con la del servidor nuevo bajo caché caliente, con un percentil 50 de 10 ms en el estrés a 25 conexiones, se estima que la capa de renderizado introduce aproximadamente 7 ms de sobrecosto por solicitud, correspondiente al procesamiento HTTP, la búsqueda en la estructura de caché y la construcción de la respuesta.

Finalmente, la línea base del portal anterior, ejecutada sobre el entorno de producción con escalones de 5, 10 y 25 conexiones concurrentes, permitió comparar ambas versiones del sistema bajo la misma herramienta de carga. La Tabla 8 presenta esta comparación para el escalón de 25 conexiones, que fue el máximo que pudo aplicarse sobre el portal anterior sin afectar a los usuarios reales.

Tabla 21. Comparación del servidor nuevo con el portal anterior a 25 conexiones concurrentes

Métrica	Portal anterior	Portal nuevo	Factor de mejora
Throughput de respuestas	955 req/s con 91,4 % de respuestas no satisfactorias	2 190 req/s sin errores	Aproximadamente 2,3 veces en throughput total, y hasta 23 veces sobre el throughput exitoso del portal anterior
Latencia p50	38 ms	10 ms	3,8 veces menor
Latencia p99	167 ms	16 ms	10,4 veces menor
Errores no satisfactorios	91,4 %	0 %	No comparable

Figura 30. Dos paneles: throughput y latencia p99 .



La interpretación de estos resultados frente a los objetivos del proyecto, así como la discusión de las limitaciones observadas, entre ellas el costo del arranque en frío, la dependencia del rendimiento del monolito para la entrega de recursos estáticos y los defectos estructurales del backend heredado que se evidencian en la elevada proporción de respuestas no satisfactorias del portal anterior, se desarrolla en el capítulo 10.

9.3 Entregables

En correspondencia con lo definido para esta fase en la metodología del proyecto, los entregables se concretan en los siguientes artefactos:

9.3.1. Reporte de pruebas de integración y compatibilidad.

Documento que consolida la matriz de verificación elaborada durante las pruebas descritas en la sección 9.1. Contiene, para cada uno de los requerimientos funcionales y casos de uso evaluados, el identificador del artefacto, su descripción, el flujo esperado, el nombre del auditor responsable, el actor asumido y el resultado registrado en cada uno de los dos medios de prueba (teléfono y computador), junto con la calificación de dificultad asociada. Este reporte constituye la evidencia objetiva del cumplimiento funcional de la solución y el insumo sobre el cual se realiza la lectura de resultados de funcionamiento y compatibilidad en el capítulo siguiente.

Las pruebas realizadas están en el Apéndice E y F.

9.3.2. Informe de rendimiento

Documento técnico que reúne los resultados de las pruebas descritas en la sección 9.2. Incluye la descripción detallada de los escenarios ejecutados, los valores observados de throughput, latencia y consumo de recursos para cada uno de ellos, la línea base del monolito y la del portal anterior, así como las tablas y mediciones que respaldan la lectura comparativa presentada en este capítulo. El informe incorpora, adicionalmente, un glosario de métricas y una caracterización del entorno sobre el que se ejecutaron las pruebas, de modo que los valores reportados puedan ser reproducidos e interpretados con claridad. Como complemento, el informe se acompaña del conjunto de figuras generadas a partir de los datos crudos capturados durante las pruebas, entre ellas la curva de throughput y latencia frente a la concurrencia, la

serie temporal de memoria residente durante la carga sostenida y la comparación gráfica con el portal anterior.

El reporte completo se encuentra en el Apéndice D

10. Conclusiones

El trabajo desarrollado a lo largo de este proyecto permitió completar la reingeniería de los portales de grupo de la plataforma COMA, avanzando desde un análisis exhaustivo del sistema heredado hasta una solución desplegada y validada en un entorno equivalente al de producción. La intervención no se limitó a un rediseño visual; abarcó la reestructuración del frontend sobre una arquitectura moderna, la adecuación intermedia del backend necesaria para soportar su operación, la incorporación de las capacidades nuevas solicitadas por el administrador de COMA y la evaluación empírica de la solución desde las perspectivas funcional, de experiencia de usuario y de desempeño técnico.

Desde el punto de vista técnico, el portal pasó de una capa de presentación basada en JSP acoplada a Backbone.js a un frontend desacoplado construido sobre React 19, Vite, Hono y el meta-framework Suamox, desplegado como un único proceso multi-tenant capaz de servir los portales de todas las escuelas desde una misma base tecnológica. La implementación incorporó un modelo de carga de datos en el servidor, una estrategia de caché de tres capas con invalidación granular por etiquetas, y una refactorización del módulo en el monolito que

introdujo controladores basados en servlets para exponer los servicios consumidos por el nuevo frontend. Esta combinación dejó una base mantenible y escalable, alineada con el estándar de CALUMET y preparada para soportar la evolución futura de los portales sin exigir nuevamente una reescritura completa del sistema.

Desde la perspectiva del administrador del grupo, el portal amplió de manera significativa las opciones de gestión y personalización disponibles. La configuración del tema, antes limitada al ajuste de dos colores del encabezado, se reemplazó por un editor de apariencia basado en tokens de diseño, con vista previa en tiempo real y presets aplicables. La creación y edición de contenido, antes dependiente del equipo de desarrollo, pasó a un editor visual por bloques con soporte bilingüe que cualquier administrador puede operar sin escribir una sola línea de código. La administración del portal, antes dispersa en distintas vistas del sistema anterior, se consolidó en un panel único organizado por áreas de responsabilidad. A estas capacidades se sumaron la exposición de datos de contacto institucional en la presentación pública y un sistema de roles y permisos configurable por el administrador para gobernar la visibilidad del contenido y de los menús.

La validación del sistema proporcionó evidencia concreta sobre el cumplimiento de los objetivos planteados. En las pruebas de integración, los requerimientos funcionales y los casos de uso evaluados por un grupo de auditores externos al equipo se ejecutaron de manera satisfactoria tanto en dispositivos móviles como en computadores de escritorio, lo que confirma la paridad funcional con el sistema anterior y la adaptabilidad responsiva de la nueva interfaz. Las pruebas de rendimiento, por su parte, mostraron mejoras sustanciales en la experiencia del

usuario final, con reducciones del Largest Contentful Paint superiores al cuarenta por ciento en la condición móvil exigente y un incremento notable del puntaje de accesibilidad sobre la página de inicio, así como una capacidad de respuesta del servidor que sostiene más de dos mil solicitudes por segundo con latencias del percentil noventa y nueve por debajo de los sesenta y dos milisegundos y sin fugas de memoria durante pruebas prolongadas.

El proceso de validación también dejó en evidencia algunas limitaciones atribuibles al entorno heredado, entre ellas el costo del arranque en frío del servidor, la dependencia del rendimiento del monolito para la entrega de recursos estáticos compartidos con otros módulos del sistema y la elevada proporción de respuestas no satisfactorias del portal anterior bajo carga, asociada a defectos estructurales del código existente. Estas limitaciones, documentadas en detalle en el informe técnico adjunto, no comprometen el cumplimiento de los objetivos del proyecto, pero sí señalan caminos concretos de trabajo posterior que exceden su alcance.

En conjunto, la evidencia reunida a lo largo del trabajo permite afirmar que el proyecto alcanzó el objetivo general planteado: la reingeniería de los portales de grupo de COMA hacia un diseño adaptable a distintos tamaños de dispositivo, con mayores opciones de personalización para los administradores y una base tecnológica que simplifica su gestión y su mantenimiento futuro. La solución entregada resuelve las limitaciones identificadas al inicio del trabajo, materializa las capacidades nuevas acordadas durante la fase de levantamiento y deja al sistema COMA con un módulo de portales de grupo más moderno, más accesible y mejor posicionado para su evolución continuada dentro de la Universidad Industrial de Santander.

11. Recomendaciones

A partir de la implementación realizada y de la evidencia recogida durante la fase de pruebas, se identifican oportunidades de mejora que, sin comprometer el cumplimiento de los objetivos del proyecto, abren caminos concretos para su evolución futura. Las siguientes subsecciones describen cada una de estas recomendaciones.

11.1. Extensión del catálogo de bloques del editor visual

El editor visual por bloques cubre los casos generales de redacción de contenido, pero admite un crecimiento sostenido hacia componentes más específicos del dominio universitario. Se recomienda ampliar el catálogo con piezas diseñadas para los escenarios más recurrentes de los grupos de investigación, entre ellos listados de integrantes con integración directa a la base de datos del grupo, galerías de publicaciones provenientes del módulo de producción intelectual, líneas de tiempo de hitos, fichas de convenios y entidades afines, secciones de preguntas frecuentes con plegado, banners de convocatoria con fechas y formularios de contacto embebidos.

11.2. Exportación e importación del tema visual

La configuración extendida de tema persistente en TR-GrupoTemaConfig es un documento estructurado naturalmente serializable. Se sugiere exponer en la sección de apariencia del administrador una función de exportación a archivo y una de importación controlada, de modo que sea posible compartir temas entre grupos, partir de plantillas

preparadas por CALUMET para escuelas que inician su portal y conservar respaldos del tema vigente antes de aplicar cambios significativos sobre la identidad gráfica.

11.3. Automatización de la disciplina de pruebas

Las pruebas de integración descritas en este trabajo se ejecutaron de manera manual por un grupo de auditores externos, lo cual fue apropiado para el cierre del proyecto pero resulta costoso de repetir sobre cada despliegue posterior. Se recomienda convertir los 68 flujos críticos de la matriz de verificación en pruebas de extremo a extremo automatizadas sobre un navegador real, utilizando una herramienta como Playwright, complementadas con pruebas unitarias sobre las funciones críticas del servidor, entre ellas la resolución de tenants, la invalidación de caché por etiquetas y los loaders de datos. El marco de pruebas de rendimiento construido durante este proyecto ya se encuentra listo para integrarse al mismo ciclo, de modo que cada despliegue significativo deje evidencia cuantitativa de su impacto sobre el throughput, la latencia y el uso de recursos.

Referencias Bibliográficas

Acumbamail. (s. f.). *Template*. Glosario de Acumbamail. https://acumbamail.com/glosario/template/

Calumet. (s. f.). *CALUMET - Grupo de Innovación y Desarrollo*. Escuela de Ingeniería de Sistemas e Informática, UIS. https://ingsistemas.uis.edu.co/eisi/grupo/calumet/

Calumet, G. (s. f.). *Calumet Estándar*. Escuela de Ingeniería de Sistemas e Informática, UIS. https://ingsistemas.uis.edu.co/eisi/Calumet/Estandar/

Carlos, R. D. L., & Angel, L. B. R. (2010). *Administración, soporte a usuarios, mantenimiento del portal web eisiweb, análisis, diseño, desarrollo e implementación de nuevos servicios para el portal web de la escuela de ingeniería de sistemas e informática*. Repositorio Institucional UIS. https://noesis.uis.edu.co/items/cb4e0fd5-8f24-423e-80ba-c151ad627b57

Grupo de Desarrollo de Software Calumet. (s. f.). *Página oficial*. Escuela de Ingeniería de Sistemas e Informática, UIS. https://ingsistemas.uis.edu.co/eisi/eisi.jsp

HTML: Lenguaje de etiquetas de hipertexto. (s. f.). *MDN Web Docs*. https://developer.mozilla.org/es/docs/Web/HTML

Interaction Design Foundation. (s. f.). *What is User Experience (UX) Design?* Recuperado de https://www.interaction-design.org/literature/topics/ux-design

MDN contributors. (2025, July 18). *Responsive Web Design (RWD)*. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Glossary/Responsive_web_design

Orlando, M. P. O. (2016). *Mantenimiento del portal web, análisis, desarrollo e implementación de nuevos servicios para el sistema comunidad académica que soporta los portales web de las escuelas y facultades*. Repositorio Institucional UIS. https://noesis.uis.edu.co/items/caa94822-bc42-48e6-97af-1336a0d991a5

Schade, A. (2014, May 4). *Responsive Web Design (RWD) and user experience*. Nielsen Norman Group. https://www.nngroup.com/articles/responsive-web-design-definition/

Socialmood. (2015, 1 de abril). *¿Qué es el Diseño Responsive?* 40deFiebre. https://www.40defiebre.com/que-es/diseno-responsive

Tomcat. (s. f.). *En Wikipedia, la enciclopedia libre*. Recuperado el 31 de mayo de 2024 de https://es.wikipedia.org/wiki/Tomcat

Yessenia, R. C. A. (2019). *Análisis, mantenimiento, diseño, desarrollo e implementación de nuevas funcionalidades y mejoras relacionadas con los módulos de portal web de grupos y de trabajos de grado en la plataforma COMA comunidad académica*. Repositorio Institucional UIS. https://noesis.uis.edu.co/items/b38282aa-4bed-4182-b60e-a4c67f1d6e4a