

Desarrollo de una red social colaborativa con ReactJS, Spring Boot y MySQL desplegada en AWS
EC2

THE SPHERE

C.P.R. Liceo “La Paz”

Proyecto Fin de Ciclo

Desarrollo de Aplicaciones Web



Autor: Alejandro Brea Gascón
Tutor: Jesús Ángel Pérez-Roca Fernández

Resumen

Desarrollo de una red social con ReactJS, Spring Boot y MySQL

TheSphere es una red social que busca unir perfiles técnicos, como desarrolladores, con perfiles más artísticos, como diseñadores. Para ello construiremos una interfaz con ReactJS diseñada en Figma. Para el backend de la aplicación usaremos Spring para llamar a la base de datos MySQL y mandar objetos JSON con la información de la BDD al frontend mediante API Rest.

Implementamos microservicios como la API de Cloudinary para poder enviar, recibir y almacenar las imágenes de nuestros usuarios y de sus proyectos.

TheSphere fomenta la colaboración y el intercambio de ideas entre sus usuarios, permitiendo que tanto técnicos como artistas puedan trabajar juntos en proyectos innovadores. Al proporcionar un espacio donde se puedan compartir y desarrollar ideas, TheSphere facilita la creación de redes profesionales y la búsqueda de oportunidades laborales en ambos campos.

Abstract

Development of a Social Network with ReactJS, Spring Boot, and MySQL

TheSphere is a social network that aims to unite technical profiles, such as developers, with more artistic profiles, such as designers. To achieve this, we will build an interface with ReactJS designed in Figma. For the backend of the application, we will use Spring Boot to interact with the MySQL database and send JSON objects with the database information to the frontend via REST API.

We have implemented microservices such as the Cloudinary API to send, receive, and store images from our users and their projects.

TheSphere promotes collaboration and the exchange of ideas among its users, allowing both technical and artistic profiles to work together on innovative projects. By providing a space where ideas can be shared and developed, TheSphere facilitates the creation of professional networks and the search for job opportunities in both fields.

Palabras Clave

ReactJS

Framework de JavaScript utilizada para construir interfaces de usuario. React permite crear componentes reutilizables que facilitan el desarrollo de aplicaciones web interactivas y eficientes.

Spring Boot

Framework para Java que simplifica la creación de aplicaciones web robustas y escalables. Spring Boot proporciona configuraciones automáticas y una estructura de proyecto clara, lo que acelera el desarrollo del backend.

MySQL

(Structured Query Language) Sistema de gestión de bases de datos relacional de código abierto. MySQL se utiliza para almacenar y gestionar los datos de la aplicación, garantizando un acceso rápido y seguro a la información.

Figma

Herramienta de diseño colaborativo basada en la web. Figma se utilizó para crear prototipos de la interfaz de usuario de TheSphere, permitiendo a los diseñadores y desarrolladores trabajar juntos en tiempo real.

API Rest

(Application Programming Interface) Estilo de arquitectura para la creación de servicios web que permite la comunicación entre el frontend y el backend mediante el uso de HTTP. Las API Rest son utilizadas para enviar y recibir datos en formato JSON.

JSON

(JavaScript Object Notation) Formato de intercambio de datos ligero y fácil de leer para los humanos y de escribir para las máquinas. JSON se utiliza en TheSphere para la comunicación entre el frontend y el backend.

AWS

Amazon Web Services, una plataforma de servicios en la nube que proporciona una infraestructura escalable y fiable para desplegar aplicaciones. En TheSphere, AWS EC2 se utiliza para alojar y gestionar la aplicación en un entorno de producción.

Cloudinary

Servicio en la nube que permite la gestión y entrega de imágenes y videos. En TheSphere, Cloudinary se utiliza para almacenar y servir las imágenes subidas por los usuarios, optimizando el rendimiento y la calidad.

Docker

Plataforma de contenedorización que permite a los desarrolladores empaquetar aplicaciones y sus dependencias en contenedores. Docker facilita la portabilidad y el despliegue de aplicaciones en diferentes entornos.

Markdown

Lenguaje de marcado ligero que permite a los usuarios escribir texto con formato utilizando una sintaxis sencilla. En TheSphere, se utiliza para permitir a los usuarios formatear sus descripciones de proyectos, incluyendo ****negritas****, **cursivas**, [enlaces](https://www.example.com) y bloques de código.

Accesibilidad

Práctica de hacer aplicaciones web utilizables por personas con discapacidades. En TheSphere, se implementan técnicas de accesibilidad para asegurar que la plataforma sea inclusiva y fácil de usar para todos los usuarios, incluyendo aquellos con discapacidades visuales, auditivas y motoras.

*A mis padres por el esfuerzo para que yo estudiara, a mis amigos por las experiencias,
relaciones y conexiones que me dieron la idea y a mis compañeros de Deloitte por
echarme una mano cuando lo necesitaba.*

SUMARIO

Resumen	3
Abstract.....	4
Palabras Clave	5
Introducción/motivación	11
Objetivos.....	12
Estado del arte	13
Caso de estudio	16
Arquitectura de la Aplicación	18
Diagramas	23
Wireframes, Mockups y Guía de Estilos	28
Desarrollo del proyecto	34
Manual Administrador	42
Manual Usuario	47
Viabilidad tecno-económica	55
Trabajo futuro	65
Conclusiones.....	67
Biblioteca de recursos web y referencias.....	68
Anexos.....	69

Introducción/motivación

En mis años en la universidad mi grupo de amigos estaba formado mayoritariamente por estudiantes de la carrera de diseño industrial. Eso y algunos aspectos que estaba viendo en mi carrera hizo que me interesara por el diseño partiendo de las bases inculcadas por mi profesora de dibujo técnico de Bachillerato.

El 30 de septiembre del 2023 tuve una conversación con esos amigos en el cumpleaños de mi amiga Lola. Necesitábamos un lugar, físico o virtual, donde poder compartir nuestras ideas, diseños, proyectos, en resumen, compartir nuestras “movidas”.

De esa conversación y de esa necesidad nació una idea de llevar la amistad un paso más allá. Una idea de permitir a otros unir sus conocimientos técnicos y artísticos para construir un espacio donde hubiese cabida para ambos y que sirviese de nexo de unión para compartir nuestros proyectos y seguir fomentando nuestra amistad.

Me parece muy potente esta unión entre técnicos y artistas, por ello quise llevar esta amistad más allá construyendo un espacio donde haya cabida para ambos y fomentar estos lazos.

Mi motivación es que todo aquel que use mi aplicación pueda experimentar esa unión que tengo con mis amigos y esa necesidad de compartir “movidas”. Ver su código, tomar la forma que imaginamos, ver su diseño y cumplir el propósito por el que se creó, esa es la verdadera motivación.

No existe aplicación sin arte ni arte sin propósito.

Objetivos

El principal objetivo de esta aplicación es dar visibilidad a esos pequeños proyectos que surgen en nuestra habitación, proporcionando una plataforma donde puedan ser vistos, compartidos y apoyados por otros. Queremos crear un espacio donde técnicos y artistas puedan mostrar sus trabajos, recibir retroalimentación constructiva y colaborar en nuevas ideas.

Además de la visibilidad, buscamos fomentar la colaboración interdisciplinaria. Al unir perfiles técnicos y artísticos, esperamos inspirar a nuestros usuarios a cruzar las fronteras de sus propias disciplinas y encontrar nuevas formas de resolver problemas y crear juntos. Este intercambio no solo enriquecerá los proyectos individuales, sino que también fortalecerá la comunidad al permitir que los usuarios aprendan unos de otros y crezcan juntos.

Finalmente, deseamos que TheSphere sea un lugar de apoyo y crecimiento personal y profesional. Al compartir nuestras “movidas”, esperamos que los usuarios encuentren no solo compañeros de proyecto, sino también amigos y mentores que los ayuden a alcanzar sus metas y a superar los desafíos que enfrentan en sus respectivas áreas.

Estado del arte

Existen varias aplicaciones y plataformas en el mercado que buscan conectar a profesionales de diversas disciplinas, cada una con sus propios puntos fuertes y débiles en comparación con TheSphere. A continuación, se describen algunas de las más relevantes:

Behance

Behance es una plataforma de Adobe que permite a los diseñadores y artistas compartir su trabajo, descubrir el trabajo de otros y conectar con oportunidades profesionales.

Puntos Fuertes:

- Red de Talento: Amplia red de creativos y profesionales del diseño.
- Integración con Adobe: Integración perfecta con las herramientas de Adobe.
- Visibilidad: Alta visibilidad para proyectos destacados.

Puntos Débiles:

- Enfoque Limitado: Principalmente enfocado en diseño gráfico y fotografía, con poca atención a proyectos técnicos.
- Competencia: Alta competencia entre usuarios puede dificultar la visibilidad para nuevos creadores.

GitHub

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos de software utilizando Git. Es ampliamente utilizada por desarrolladores para compartir código, colaborar en proyectos y gestionar versiones de software.

Puntos Fuertes:

- Colaboración en Código: Herramientas robustas para la colaboración y control de versiones.
- Integraciones: Integraciones con numerosas herramientas de desarrollo y CI/CD.
- Comunidad Técnica: Gran comunidad de desarrolladores.

Puntos Débiles:

- Falta de Enfoque en Diseño: Poco enfoque en aspectos artísticos o de diseño.
- Curva de Aprendizaje: Puede ser intimidante para diseñadores y artistas sin experiencia técnica.

Dribbble

Dribbble es una plataforma donde los diseñadores pueden compartir capturas de pantalla de sus proyectos, recibir feedback y encontrar oportunidades laborales.

Puntos Fuertes:

- Enfoque en Diseño: Fuerte enfoque en el diseño y la creatividad.
- Comunicación Visual: Centrado en la presentación visual de proyectos.
- Red Profesional: Conexiones con empresas y oportunidades laborales.

Puntos Débiles:

- Falta de Proyectos Técnicos: Escaso soporte para proyectos de desarrollo o técnicos.
- Visibilidad: La visibilidad puede ser un reto debido al gran volumen de contenido.

Kaggle

Kaggle es una plataforma de competencias de ciencia de datos y aprendizaje automático. Permite a los usuarios participar en competencias, publicar conjuntos de datos y compartir notebooks de código.

Puntos Fuertes:

- Competencias de Ciencia de Datos: Kaggle es conocido por sus competencias que permiten a los usuarios resolver problemas de ciencia de datos y aprendizaje automático del mundo real, con premios significativos para los ganadores.
- Recursos Educativos: Ofrece una amplia gama de recursos educativos, incluidos notebooks interactivos y tutoriales que ayudan a los usuarios a aprender y mejorar sus habilidades.
- Comunidad Activa: Una comunidad activa y colaborativa donde los usuarios pueden compartir conocimientos y colaborar en proyectos de ciencia de datos.

Puntos Débiles:

- Enfoque Limitado a Ciencia de Datos: Kaggle está altamente especializado en ciencia de datos y aprendizaje automático, lo que limita su atractivo para otros perfiles técnicos y creativos.
- Curva de Aprendizaje para Principiantes: Aunque ofrece recursos educativos, la complejidad de los problemas y la competencia puede ser intimidante para los principiantes.

Tabla Comparativa de Funcionalidades:

Funcionalidad	Behance	Github	Dribbble	Kaggle	TheSphere
Compartir Proyectos	Sí	Sí	Sí	No	Sí
Colaboración	No	Sí	No	Sí	Sí
Integración de Código	No	Sí	No	Sí	Sí
Enfoque en Diseño	Sí	No	Sí	No	Sí
Enfoque en Desarrollo	No	Sí	No	Sí	Sí
Comentarios/Feedback	Sí	Sí	Sí	Sí	Sí
Eventos/Desafíos	No	No	No	Sí	Sí
Perfil Personalizado	Sí	Sí	Sí	Sí	Sí

Figura 1: Tabla comparativa de funcionalidades. Elaboración propia.

La tabla comparativa de funcionalidades presenta un análisis de las características clave de cinco plataformas: Behance, GitHub, Dribbble, Kaggle y TheSphere. Se evaluaron ocho funcionalidades esenciales para determinar la cobertura que cada plataforma proporciona.

Esta comparación demuestra que TheSphere se posiciona como una plataforma versátil y completa, capaz de satisfacer las necesidades tanto de desarrolladores como de diseñadores, promoviendo la colaboración interdisciplinaria y ofreciendo una experiencia enriquecedora para sus usuarios.

Caso de estudio

En este caso de estudio, se explican los puntos clave que se desarrollarán en TheSphere para abordar la problemática existente de desconexión entre perfiles técnicos y artísticos y para mejorar las soluciones que ya existen en el mercado. A continuación, se describen estos puntos detalladamente:

1. Interfaz de Usuario Integrada y Amigable

- Problemática: Las plataformas actuales suelen estar especializadas en un solo tipo de usuarios, ya sean técnicos o artísticos, lo que dificulta la colaboración entre estos perfiles.
- Solución: TheSphere ofrece una interfaz de usuario integrada y amigable, que facilita la navegación y la interacción tanto para desarrolladores como para diseñadores. La interfaz será intuitiva y estará optimizada para que ambos tipos de usuarios puedan compartir sus proyectos, explorar el trabajo de otros y colaborar fácilmente.

2. Espacios de Proyecto Colaborativos

- Problemática: La colaboración en proyectos entre diseñadores y desarrolladores es a menudo desorganizada y fragmentada debido a la falta de plataformas que integren ambas disciplinas.
- Solución: TheSphere proporcionará espacios de proyecto colaborativos donde los usuarios pueden trabajar juntos en tiempo real. Estos espacios permitirán la integración de herramientas como Figma para diseño y GitHub para control de versiones y gestión de código. Además, se implementarán funciones de chat y discusión para facilitar la comunicación y el intercambio de ideas.

3. Perfiles Personalizados y Portafolios

- Problemática: Las plataformas existentes no siempre permiten a los usuarios mostrar de manera efectiva una combinación de habilidades técnicas y artísticas.
- Solución: Los usuarios de TheSphere podrán crear perfiles personalizados que incluyan su portafolio de proyectos, destacando tanto sus habilidades técnicas como artísticas. Estos perfiles estarán optimizados para ser fácilmente navegables y permitirán a los usuarios conectar con otros profesionales de manera más efectiva.

4. Sistema de Publicación y Feedback

- Problemática: La obtención de feedback constructivo y la visibilidad de los proyectos pueden ser limitadas en las plataformas actuales.
- Solución: TheSphere implementará un sistema de publicación donde los usuarios pueden compartir sus proyectos y recibir feedback de la comunidad. Habrá herramientas para comentar proyectos, lo que incentivará la participación activa y el intercambio de conocimientos. Además, se promoverán los proyectos de forma aleatoria en la página principal para aumentar su visibilidad.

5. Eventos y Retos Comunitarios

- Problemática: La falta de interacción y eventos comunitarios limita las oportunidades de aprendizaje y colaboración.
- Solución: TheSphere organizará eventos y retos comunitarios que animen a los usuarios a participar y colaborar en proyectos conjuntos. Estos eventos incluirán hackathons, desafíos de diseño y talleres en línea que fomentarán la creatividad y la innovación dentro de la comunidad.

6. Integración con Herramientas Externas

- Problemática: La desconexión entre las herramientas utilizadas por desarrolladores y diseñadores puede dificultar la colaboración efectiva.
- Solución: TheSphere integrará herramientas externas populares entre desarrolladores y diseñadores, como Figma, Adobe Creative Cloud, GitHub y otros. Esta integración permitirá una transición fluida entre diferentes etapas del proyecto y mejorará la colaboración interdisciplinaria.

Arquitectura de la Aplicación

La aplicación TheSphere sigue una arquitectura de microservicios, dividiendo claramente el frontend y el backend para permitir escalabilidad, mantenimiento y despliegue independiente. A continuación, se detalla la arquitectura de la aplicación:

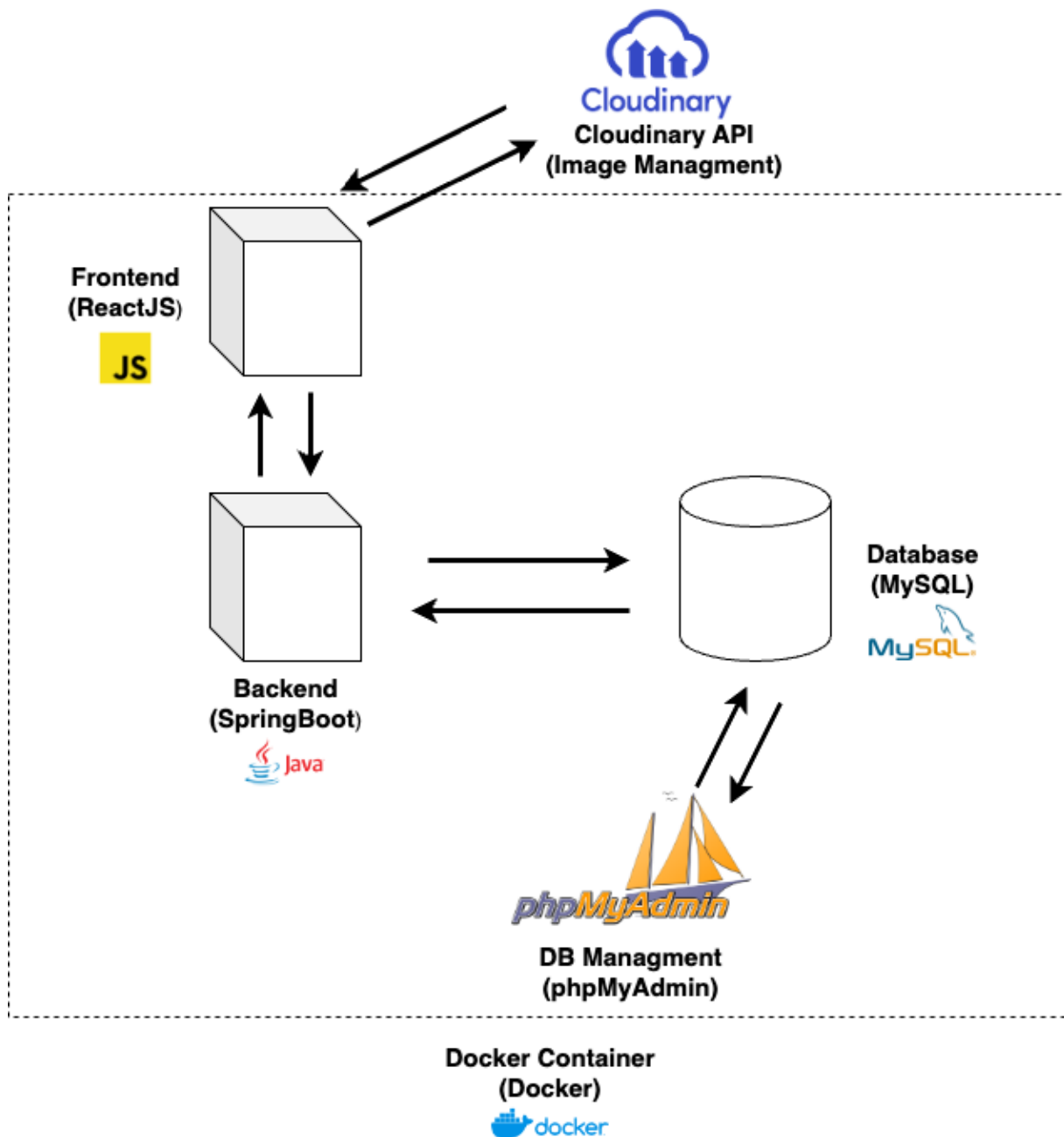


Figura 2: Arquitectura de la aplicación. Elaboración propia.

- Frontend (ReactJS): Representa la interfaz de usuario de la aplicación. Es responsable de renderizar las páginas y componentes con los que interactúan los usuarios.
- Backend (Spring Boot): Maneja la lógica de negocio y sirve como intermediario entre el frontend y la base de datos. Expone APIs REST que el frontend consume.
- Database (MySQL): Almacena todos los datos necesarios para la aplicación, como usuarios, publicaciones, comentarios, etc. El backend interactúa con la base de datos a través de JPA.
- Cloudinary API: Se encarga de la gestión de imágenes. El frontend interactúa con este servicio para subir y obtener imágenes.
- phpMyAdmin: Herramienta de gestión de base de datos MySQL, utilizada para administrar y gestionar la base de datos.
- Docker Containers: La aplicación está contenedorizada usando Docker, lo que facilita el despliegue y la escalabilidad. Incluye contenedores separados para el frontend, backend, base de datos, Cloudinary API y phpMyAdmin.

Las flechas indican las interacciones principales:

- API Calls: Las llamadas API desde el frontend al backend para realizar operaciones como autenticación de usuarios, creación de publicaciones, etc.
- JPA: La interacción del backend con la base de datos para realizar operaciones de persistencia de datos.
- Image Management: Gestión de imágenes entre el frontend y Cloudinary API.
- DB Management: Gestión y administración de la base de datos a través de phpMyAdmin.

1. Frontend

El frontend está construido con ReactJS, lo que permite crear una interfaz de usuario dinámica y receptiva. La estructura del frontend incluye componentes reutilizables y se organiza en módulos claros para mantener la mantenibilidad del código.

Componentes Clave del Frontend:

- Componentes de UI: Navbar, Footer, PostDetails, etc.
- Páginas: Home, Login, Register, Profile, CreatePost, etc.

- Gestión de Estado: Utiliza Context API de React para compartir estado entre componentes.
- Rutas: React Router para manejar la navegación.
- Estilos: CSS modularizado para cada componente.
- Interacción con Backend: Axios para realizar llamadas a la API REST.

Diagrama Simplificado del Frontend:

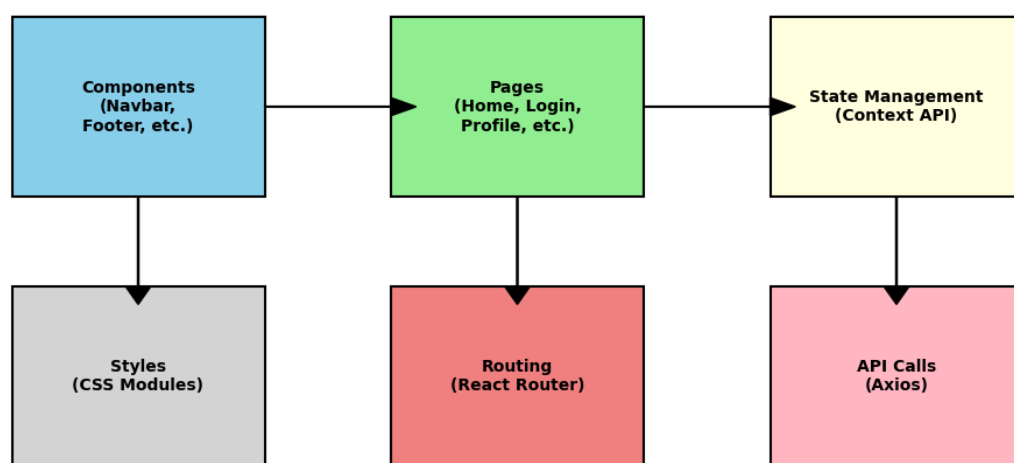


Figura 3: Arquitectura Frontend. Elaboración propia.

2. Backend

El backend está construido con Spring Boot, proporcionando un framework robusto y escalable para manejar las operaciones del servidor. La arquitectura del backend sigue un patrón de capas típico:

Capas del Backend:

- Controladores (Controllers): Manejan las solicitudes HTTP y responden con datos en formato JSON.
- Servicios (Services): Contienen la lógica de negocio de la aplicación.
- Repositorios (Repositories): Interactúan con la base de datos utilizando JPA (Java Persistence API).
- Modelos (Models): Representan las entidades de la base de datos.
- Configuración de Seguridad: Maneja la autenticación y autorización.

Diagrama del Backend

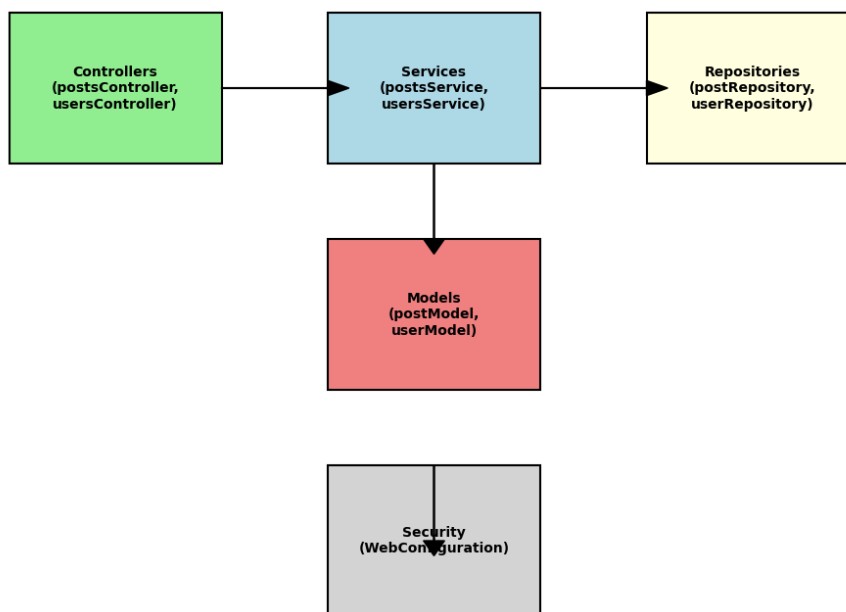


Figura 4: Arquitectura Backend. Elaboración propia.

3. Base de Datos

La base de datos de TheSphere está basada en un modelo relacional, utilizando MySQL como sistema de gestión de bases de datos (DBMS). Este modelo permite una organización estructurada de los datos en tablas relacionadas, garantizando la integridad y consistencia de la información almacenada.

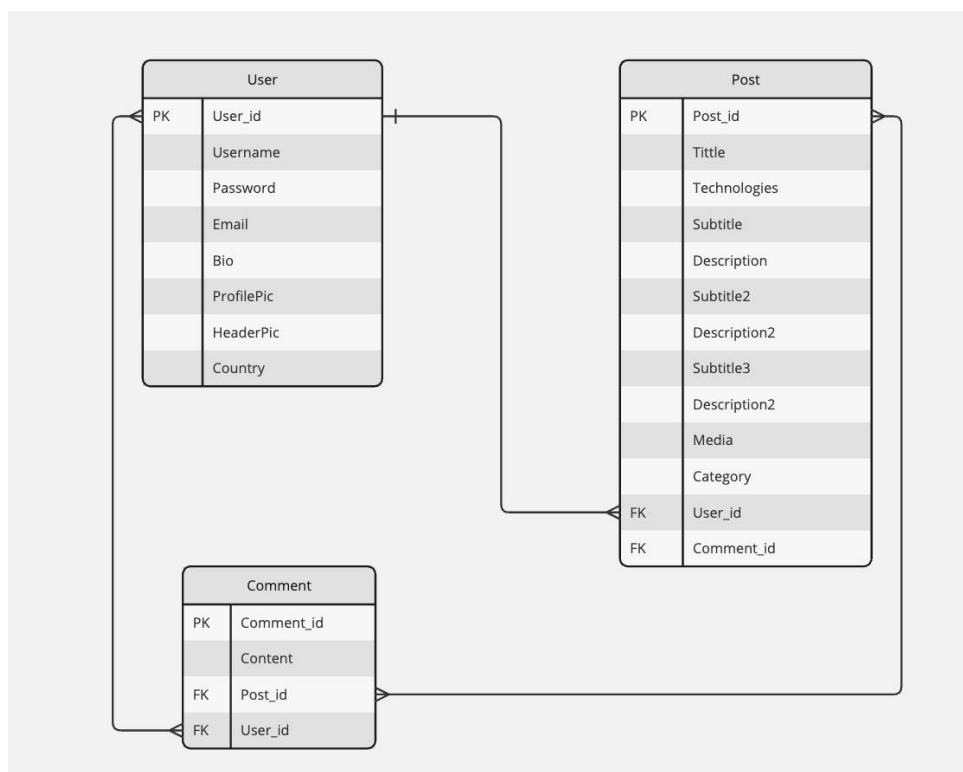


Figura 5: Diagrama de tablas. Elaboración propia.

Entidades Principales

- Usuarios (Users)
 - Almacena la información personal de los usuarios, como nombre, correo electrónico, contraseña cifrada, país, biografía, y enlaces a las imágenes de perfil y banner.
 - Relaciones:
 - Un usuario puede crear múltiples publicaciones.
 - Un usuario puede hacer múltiples comentarios.
- Proyectos (Posts)
 - Contiene los datos de las publicaciones realizadas por los usuarios, incluyendo título, tecnologías utilizadas, subtítulos, descripciones, imagen asociada y categoría (DESIGN o DEVELOPMENT).
 - Relaciones:
 - Cada publicación pertenece a un usuario.
 - Una publicación puede tener múltiples comentarios.
- Comentarios (Comments)
 - Guarda los comentarios realizados por los usuarios en las publicaciones, incluyendo el contenido del comentario y referencias a los usuarios y publicaciones correspondientes.
 - Relaciones:
 - Cada comentario pertenece a un usuario.
 - Cada comentario pertenece a una publicación.

Diagramas

Diagrama E-R (Entidad-Relación)

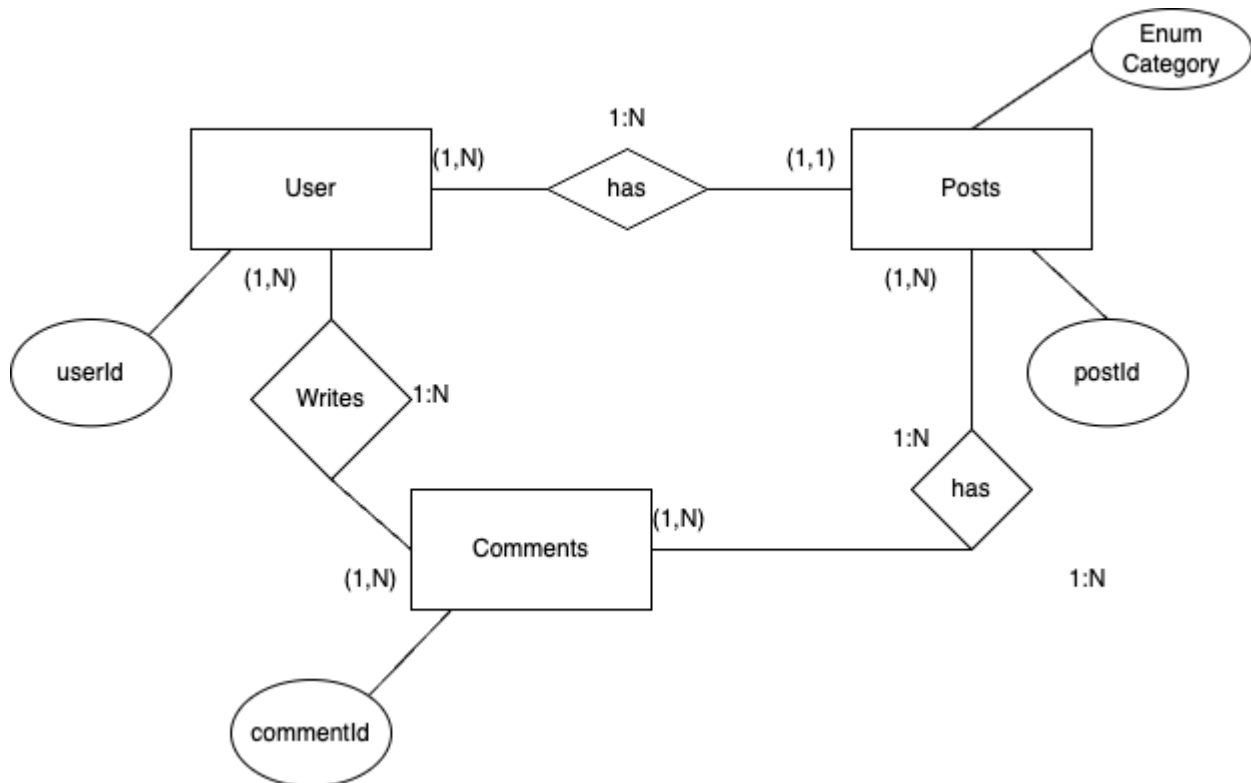


Figura 6: Diagrama Entidad-Relación. Elaboración propia.

Explicación del Diagrama E-R:

- User: Representa a los usuarios de la aplicación. Cada usuario tiene atributos como id, username, email, password, profilePic, headerPic y bio.
- Post: Representa las publicaciones que los usuarios pueden crear. Cada post tiene atributos como id, title, description, media, category y una relación con el usuario que lo creó (user_id).
- Comment: Representa los comentarios que los usuarios pueden dejar en los posts. Cada comentario tiene atributos como id, content y relaciones con el usuario que lo hizo (user_id) y el post en el que se comentó (post_id).
- Category: Enum que define las categorías de los posts, como DESIGN y DEVELOPMENT.

Diagrama de Casos de Uso

El diagrama de casos de uso muestra las interacciones entre los actores (usuarios) y el sistema. Los casos de uso principales incluyen Register, Login, Create Post, Edit Profile, Comment on Post, Search Posts y View Profile.



Figura 7: Diagrama de casos de uso. Elaboración propia.

Explicación del Diagrama de Casos de Uso:

- Register: El usuario se registra proporcionando un nombre de usuario, correo electrónico y contraseña.
- Login: El usuario inicia sesión con su correo electrónico y contraseña.
- Create Post: El usuario crea una nueva publicación, subiendo una imagen y proporcionando detalles como título, descripción y categoría.
- Edit Profile: El usuario edita su perfil, cambiando su foto de perfil, banner, bio y otros detalles.
- Comment on Post: El usuario comenta en una publicación existente.
- Search Posts: El usuario busca publicaciones utilizando la barra de búsqueda.
- View Profile: El usuario ve el perfil de otro usuario.

Diagrama de Clases (UML)

El diagrama de clases muestra la estructura del código, incluyendo las clases principales y las relaciones entre ellas.

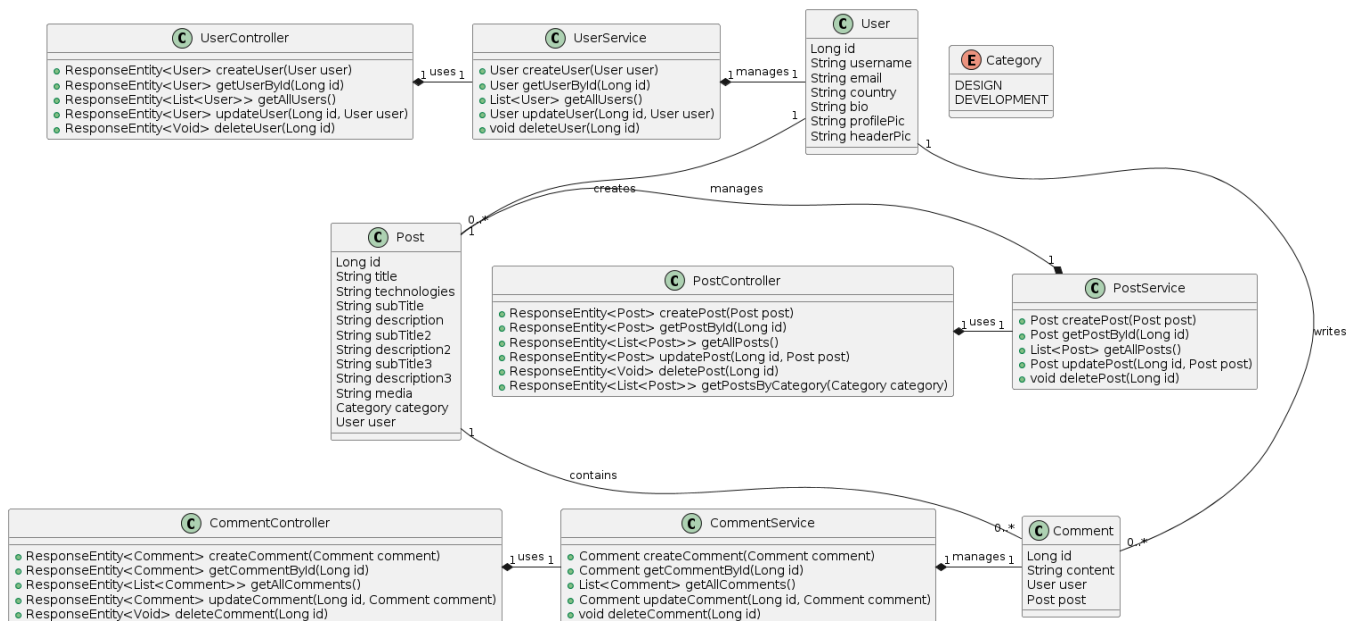


Figura 8: Diagrama de clases UML. Elaborado a través de [PlantUML](#)

Explicación del Diagrama de Clases:

- User: Clase que representa a los usuarios de la aplicación.
- Post: Clase que representa las publicaciones creadas por los usuarios.
- Comment: Clase que representa los comentarios hechos por los usuarios en los posts.
- Category: Enum que define las categorías posibles para los posts.
- Service: Clases de servicio que contienen la lógica de negocio de la aplicación.
- Controller: Clases de controlador que manejan las peticiones HTTP y envían respuestas en formato JSON.

Mapa de Navegación

El mapa de navegación es una representación visual de cómo se estructuran y conectan las diferentes páginas y secciones de la aplicación. Este diagrama es esencial para entender la experiencia del usuario, ya que muestra la arquitectura de la información y el flujo de navegación.

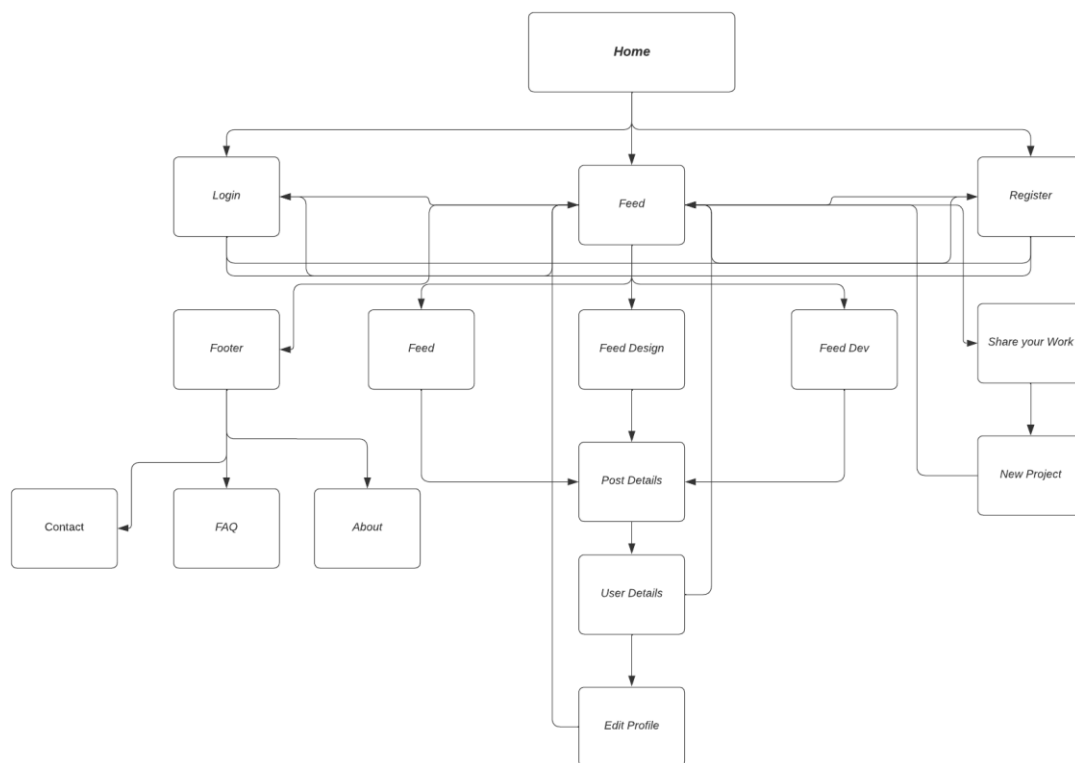


Figura 9: Mapa de navegación. Elaboración propia.

Estructura General

La aplicación TheSphere se compone de varias páginas principales, cada una de las cuales tiene subpáginas y funcionalidades específicas. A continuación, se describen las principales secciones del mapa de navegación:

- **Página de Inicio:** Actúa como punto de entrada y salida, proporcionando accesos rápidos a las secciones clave como el registro, login, y feed.
- **Registro/Login:** Facilitan la creación de nuevas cuentas y el acceso a cuentas existentes, con enlaces adicionales a la recuperación de contraseña y términos y condiciones para asegurar el cumplimiento y soporte al usuario.
- **Feed:** Es el corazón de la interacción social, mostrando publicaciones categorizadas y permitiendo un fácil acceso a detalles de publicaciones individuales y perfiles de

usuario. Incluye filtros y opciones de navegación para mejorar la experiencia del usuario.

- Perfil de Usuario: Muestra información detallada del usuario y sus publicaciones, permitiendo ediciones y accesos rápidos a otras secciones relevantes como publicaciones detalladas y el feed.
- Crear Publicación: Página dedicada para la creación de nuevo contenido, integrada con herramientas para añadir imágenes, código y categorías, asegurando que las publicaciones sean completas y atractivas.
- Publicación Detallada: Proporciona una vista ampliada de las publicaciones con todos sus detalles, comentarios y opciones de interacción, conectando al usuario con otros perfiles y el feed principal.
- Contacto: Facilita la comunicación directa con el equipo de soporte, asegurando que los usuarios puedan enviar consultas y recibir asistencia de manera eficiente.
- Admin Dashboard: Solo accesible por administradores. Proporciona herramientas avanzadas para la gestión de usuarios y contenido, crucial para mantener la plataforma segura y moderada.

Wireframes, Mockups y Guía de Estilos

Wireframes:

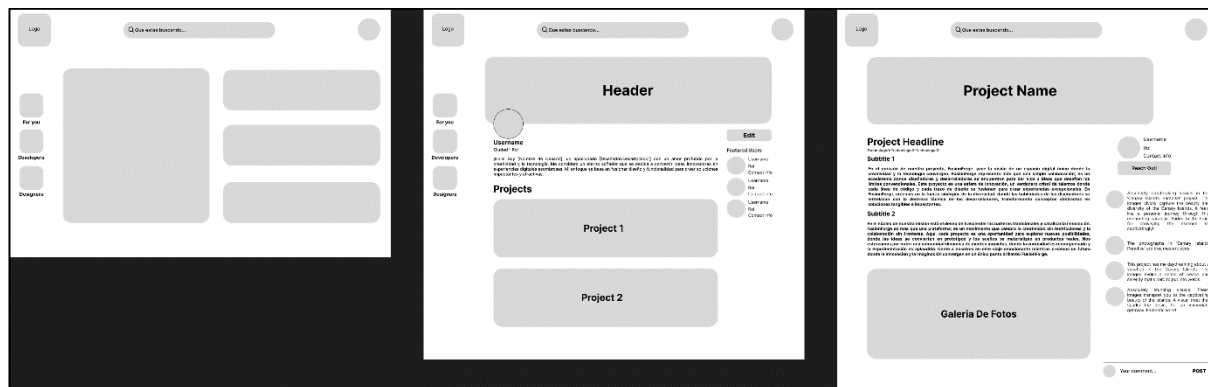


Figura 10: Wireframes.

Desde la primera versión, el diseño de TheSphere ha sufrido muchos cambios, pero siempre manteniendo el concepto inicial inspirado en el Material Design de Google y el diseño de Apple. El objetivo ha sido proporcionar una interfaz limpia, intuitiva y visualmente atractiva que facilite la navegación y la interacción de los usuarios.

Al principio, como se puede ver en los wireframes, la barra de navegación se ubicaba en el lateral izquierdo con botones cuadrados. Este diseño fue modificado para mejorar la usabilidad y la experiencia del usuario. Ahora, la barra de navegación se encuentra en la parte superior de la web, ubicada entre el logo y la barra de búsqueda, lo que permite un acceso más directo y rápido a las diferentes secciones de la plataforma.

Otro cambio significativo fue la eliminación del botón de seguir en el perfil de usuario. La razón detrás de esta decisión es evitar la ansiedad que puede generar el conteo de seguidores y seguidos, promoviendo así un ambiente más saludable y centrado en el contenido en lugar de en la popularidad. No queremos generar esa comparación innecesaria entre usuarios basada en números, sino fomentar la colaboración y el intercambio de ideas sin presión.

Los wireframes representan las diferentes vistas de la aplicación, incluyendo la página de inicio, el feed de publicaciones, el perfil de usuario y las páginas de creación y edición de posts. Estas representaciones visuales nos han permitido planificar y ajustar el diseño de la interfaz de usuario, asegurando que todos los elementos necesarios estén presentes y correctamente alineados con la visión del proyecto.

Mockups:

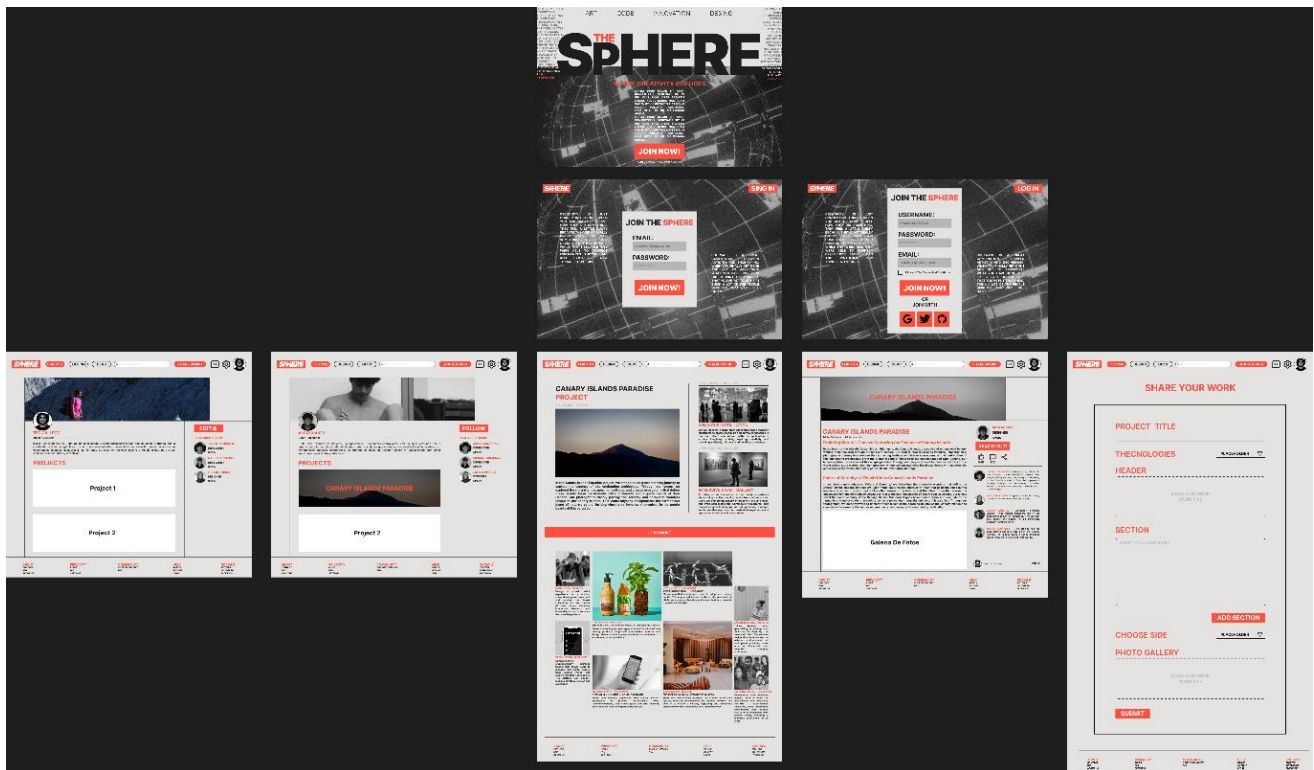


Figura 11: Mockups.

En los mockups que se muestran a continuación, podemos ver los cambios de diseño y filosofía explicados anteriormente, incluyendo la barra de navegación superior y las páginas de login y register.

El layout y los grids elegidos para la aplicación permiten dar más enfoque y protagonismo al contenido multimedia, como imágenes, sin que pierda relevancia el texto que describe el proyecto. Esta organización visual asegura que los usuarios puedan apreciar tanto el aspecto visual como el descriptivo de cada proyecto, ofreciendo una experiencia equilibrada y atractiva.

Los wireframes y mockups se diseñaron en Figma y representan las diferentes vistas de la aplicación, incluyendo la página de inicio, el feed de publicaciones, el perfil de usuario y las páginas de creación y edición de posts. Estas vistas proporcionan una guía visual clara para el desarrollo de la interfaz de usuario, asegurando que todos los elementos necesarios estén presentes y correctamente alineados con la visión del proyecto.

A continuación, se detallan las principales características de cada sección:

- Página de Inicio: La página de inicio presenta una introducción visualmente impactante de la plataforma, con un banner que resalta la propuesta de valor y botones de llamada a la acción para unirse a la comunidad.
- Barra de Navegación Superior: La barra de navegación proporciona acceso rápido a las principales secciones de la plataforma, incluyendo el feed de publicaciones, los perfiles de usuario, y las opciones de búsqueda y compartir.
- Páginas de Login y Registro: Estas páginas están diseñadas para facilitar el acceso de nuevos usuarios y la autenticación de usuarios existentes, con un diseño limpio y accesible.
- Feed de Proyectos: El feed de publicaciones muestra los proyectos más recientes y populares, permitiendo a los usuarios explorar y descubrir nuevos contenidos de forma sencilla.
- Perfil de Usuario: La página de perfil permite a los usuarios mostrar sus proyectos y biografía, proporcionando un espacio personal dentro de la plataforma.
- Creación y Edición de Proyectos: Estas páginas permiten a los usuarios subir y editar sus proyectos, incluyendo la adición de imágenes y descripciones detalladas.

Estos elementos visuales y funcionales fueron cuidadosamente diseñados para proporcionar una experiencia de usuario intuitiva y atractiva, facilitando la interacción y el compromiso dentro de la comunidad de TheSphere.

En resumen, los cambios de diseño han mejorado la usabilidad y la estética de la plataforma, manteniendo siempre en mente la filosofía de TheSphere: unir a perfiles técnicos y artísticos en un espacio colaborativo donde el contenido de calidad es el verdadero protagonista.

Guía de Estilo:

La guía de estilo es fundamental para mantener la coherencia visual y la identidad de marca en todas las interfaces y componentes de la aplicación. A continuación, se detallan los elementos clave de la guía de estilo de TheSphere.

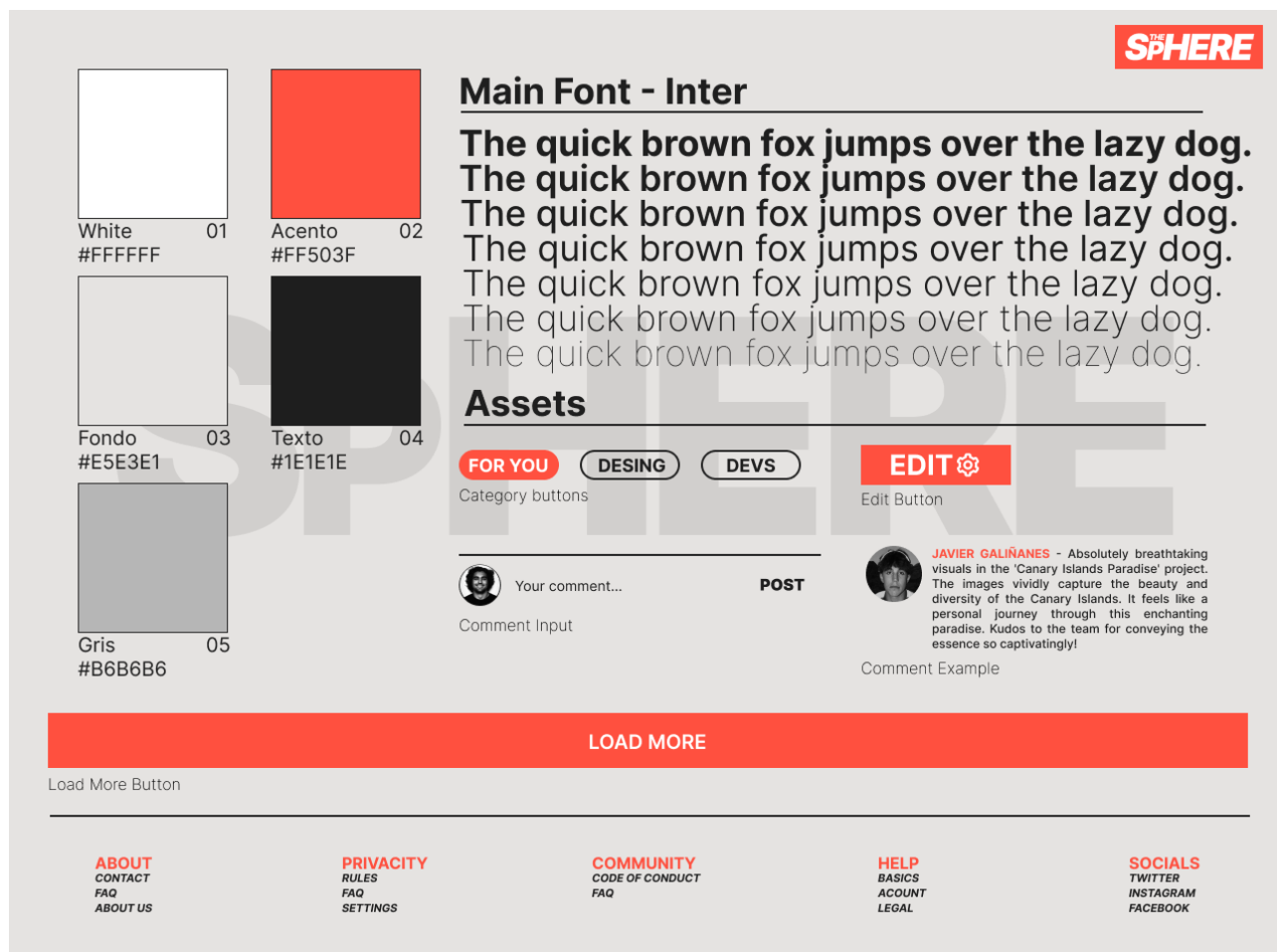


Figura 12: Guía de estilos.

Colores

La paleta de colores de TheSphere está diseñada para ser moderna y atractiva, asegurando una experiencia visual agradable para los usuarios.

- **Naranja:** #FF503F - Utilizado para botones principales, acentos y elementos interactivos clave.
- **Negro:** #1E1E1E - Utilizado para el texto principal y elementos de navegación.
- **Blanco:** #FFFFFF - Utilizado como color de fondo y para textos en botones de acento.
- **Crema:** #E5E3E1 - Utilizado como color de fondo alternativo para secciones.
- **Gris:** #B6B6B6 - Utilizado para bordes y elementos de texto secundarios.

Tipografías

La tipografía principal utilizada en TheSphere es "Inter", una fuente sans-serif que proporciona una lectura clara y moderna. Se usa en varios tamaños y pesos para crear jerarquía y distinción entre diferentes tipos de contenido.

- Principal: Inter, sans-serif

Botones

Los botones en TheSphere están diseñados para ser claros y accesibles, utilizando una combinación de colores y cambios de estado para proporcionar feedback visual a los usuarios.

- Botones Principales: Tienen un fondo naranja con texto blanco y cambian a negro con texto blanco al pasar el cursor. Este diseño asegura que los botones importantes se destaquen y sean fácilmente identificables.
- Botones de Navegación: No tienen fondo inicialmente, tienen un borde naranja con texto naranja y cambian a fondo naranja con texto blanco al pasar el cursor. Esto ayuda a mantener la navegación limpia y enfocada en la interacción principal.

Elementos de la Guía de Estilo

La guía de estilo también incluye ejemplos de cómo se ven los elementos de interfaz en uso, asegurando que cada componente siga los principios de diseño establecidos. A continuación, se presenta un resumen visual de los elementos principales:

- Colores: Los colores que conforman la paleta de TheSphere
- Main Font - Inter: Ejemplo de la tipografía principal en diferentes tamaños y pesos.
- Assets: Ejemplos de botones de categoría, campos de entrada de comentarios y otros elementos de interfaz clave.
- Category Buttons: Botones para las diferentes categorías ("FOR YOU", "DESIGN", "DEVS") con sus estilos correspondientes.
- Edit Button: Ejemplo del botón de edición, mostrando su estado normal y al pasar el cursor.
- Comment Input: Ejemplo de cómo se verá el campo de entrada para comentarios, incluyendo el texto del usuario y el botón de "POST".
- Load More Button: Ejemplo del botón para cargar más contenido.

Uso de los Elementos de la Guía de Estilo

Cada elemento de la guía de estilo se ha diseñado para garantizar la coherencia y la accesibilidad en toda la aplicación. La paleta de colores, las tipografías y los estilos de botones se aplican de manera uniforme en todas las vistas de la aplicación, desde la página de inicio hasta los perfiles de usuario y los detalles de los proyectos. Esto no solo mejora la estética de la aplicación, sino que también mejora la experiencia del usuario al proporcionar una interfaz intuitiva y agradable.

Para más desarrollo e información sobre el diseño de la aplicación consultar la Guía de Identidad Visual en los [Anexos](#).

Desarrollo del proyecto

Tecnologías Utilizadas

- Frontend: ReactJS
- Backend: Spring Boot
- Base de Datos: MySQL
- Hosting de Imágenes: Cloudinary
- Contenedores: Docker
- Despliegue: AWS EC2

Estructura del Proyecto

1. Frontend:

- a. React Components: Los componentes están organizados en carpetas según su funcionalidad (Navbar, Footer, PostDetails, etc.).
- b. Pages: Las páginas están organizadas en carpetas según su rol.
- c. State Management: Se utiliza el Context API de React para la gestión del estado del usuario.
- d. Routing: React Router se usa para manejar la navegación entre diferentes vistas.
- e. Estilos: Los estilos CSS están organizados en archivos correspondientes a cada componente.

2. Backend:

- a. Controladores REST: Controladores Spring Boot que manejan las peticiones HTTP y envían respuestas en formato JSON.
- b. Servicios: Servicios que contienen la lógica de negocio de la aplicación.
- c. Modelos: Clases Java que representan las entidades de la base de datos.
- d. Repositorios: Interfaces que extienden JpaRepository para interactuar con la base de datos MySQL.

Desarrollo del proyecto

El desarrollo del proyecto se dividió en varias fases, comenzando con la creación del backend, seguido por la implementación del frontend, y culminando con la integración de ambos mediante llamadas API. A continuación, se detallan los pasos y tecnologías utilizadas en cada fase del desarrollo:

Creación del Backend

La primera etapa del desarrollo se centró en la construcción del backend utilizando Spring Boot. Este framework se eligió por su robustez y facilidad de configuración. Se inició

con la configuración del proyecto y la creación de la estructura básica, incluyendo los paquetes para los controladores, servicios y modelos.

1. Configuración del Proyecto: Se creó un nuevo proyecto Spring Boot con dependencias para JPA, MySQL y Spring Web. Se configuró la conexión a la base de datos MySQL mediante el archivo `application.properties`.
2. Modelado de la Base de Datos: Se diseñó el esquema de la base de datos utilizando entidades JPA. Las principales entidades incluyeron User, Post y Comment, con relaciones adecuadas entre ellas.
3. Servicios y Repositorios: Se implementaron servicios para manejar la lógica de negocio y repositorios JPA para la interacción con la base de datos. Esto permitió operaciones CRUD para usuarios y publicaciones.
4. Controladores REST: Se desarrollaron controladores REST para exponer endpoints que el frontend pudiera consumir. Estos endpoints manejaban operaciones como registro de usuarios, autenticación, creación de publicaciones y comentarios.

Implementación del Frontend

El siguiente paso fue la implementación del frontend utilizando ReactJS. Este framework se eligió por su modularidad y su capacidad para crear interfaces de usuario dinámicas y responsivas.

1. Diseño de la Interfaz: Se diseñaron las vistas principales de la aplicación utilizando Figma, lo que proporcionó una guía visual para el desarrollo de los componentes React.
2. Configuración del Proyecto: Se inició con la configuración de un nuevo proyecto React utilizando Vite. Se estableció la estructura del proyecto, organizando los componentes, contextos y estilos.
3. Componentes y Estado: Se crearon componentes reutilizables para diferentes partes de la aplicación, como formularios de registro, perfiles de usuario, y vistas de publicaciones. El estado de la aplicación se gestionó mediante el Context API de React, permitiendo compartir datos entre componentes.
4. Interacción con el Backend: Se integraron los componentes con el backend mediante llamadas a los endpoints REST utilizando la biblioteca Axios. Esto incluyó la gestión de autenticación de usuarios, la visualización de publicaciones y la creación de nuevos comentarios.

Integración y Pruebas

Una vez implementados el backend y el frontend, se procedió a la integración y pruebas del sistema completo.

1. Pruebas de Endpoints: Se realizaron pruebas de los endpoints REST utilizando Postman, asegurando que las respuestas fueran correctas y gestionando adecuadamente los errores.
2. Integración Frontend-Backend: Se probaron las interacciones entre el frontend y el backend, verificando que las operaciones como el registro de usuarios, la autenticación y la gestión de publicaciones funcionaran sin problemas.
3. Optimización y Refinamiento: Se realizaron ajustes y optimizaciones en el código, tanto en el frontend como en el backend, para mejorar el rendimiento y la experiencia del usuario.

Contenerización con Docker

Para asegurar que el despliegue de la aplicación sea consistente y reproducible en diferentes entornos, se utilizó Docker Compose. Este enfoque permite definir y gestionar múltiples contenedores Docker que ejecutan diferentes partes de la aplicación. A continuación, se detalla el proceso de desarrollo de la configuración de Docker Compose para TheSphere.

1. Creación de Dockerfiles para Frontend y Backend.

Primero, se crearon los archivos Dockerfile para el frontend y el backend, asegurando que cada uno tenga las dependencias y configuraciones necesarias para construir y ejecutar las respectivas aplicaciones.

- Dockerfile para Frontend

```
FROM node:18 AS build
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build
RUN ls -la /app/dist
EXPOSE 5173
CMD ["npm", "run", "dev"]
```

- Dockerfile para Backend:

```
FROM maven:3.8.7-eclipse-temurin-17 AS build
WORKDIR /app
COPY pom.xml ./
RUN mvn dependency:go-offline
COPY src ./src
RUN mvn clean install -DskipTests -X
FROM eclipse-temurin:17-jre
WORKDIR /app
COPY --from=build /app/target/*.jar app.jar
```

```
COPY wait-for-it.sh /app/wait-for-it.sh
RUN chmod +x /app/wait-for-it.sh
EXPOSE 8080
CMD ["/wait-for-it.sh", "mysql:5012", "--", "java", "-jar", "app.jar"]
```

2. Diseño del archivo docker-compose.yml.

Se diseñó el archivo docker-compose.yml para orquestar los contenedores del frontend, backend, base de datos MySQL y una imagen de phpmyadmin para gestión de la DB. Este archivo define cómo se construyen, configuran y comunican los servicios entre sí.

- Estructura del docker-compose.yml

```
version: '3.8'

services:
  mysql:
    image: mysql:8.0
    container_name: TheSphereDB
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: subesdb
    ports:
      - "3306:3306"
    expose:
      - "3306"
    healthcheck:
      test: ["CMD-SHELL", "mysqladmin ping -h localhost"]
      interval: 10s
      timeout: 5s
      retries: 3
    networks:
      - springapimysql-net
    volumes:
      - mysql-data:/var/lib/mysql

  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    container_name: PhpMyAdmin
    environment:
      PMA_HOST: mysql
      MYSQL_ROOT_PASSWORD: root
    ports:
      - "8081:80"
    depends_on:
      - mysql
    networks:
      - springapimysql-net

  backend:
    container_name: TheSphereBackend
    build:
      context: ./Backend
      dockerfile: Dockerfile
    depends_on:
      mysql:
        condition: service_healthy
    ports:
      - "8080:8080"
```

```
networks:
  - springapimysql-net

frontend:
  container_name: TheSphereFrontend
  build:
    context: ./Frontend
    dockerfile: Dockerfile
  networks:
    - springapimysql-net
  ports:
    - "5173:5173"

networks:
  springapimysql-net:
    driver: bridge

volumes:
  mysql-data:
```

3. Variables de entorno.

Se configuraron variables de entorno en el archivo `application.properties` para permitir la conexión del backend a la base de datos MySQL. Esto incluye la URL de la base de datos, el nombre de usuario y la contraseña.

4. Definición de volúmenes.

Para persistir los datos de la base de datos MySQL, se definió un volumen en el archivo `docker-compose.yml`. Esto asegura que los datos no se pierdan cuando los contenedores se detienen o se reinician.

5. Configuración de dependencias entre Services.

Se utilizaron las directivas `depends_on` para gestionar el orden de inicio de los contenedores. El contenedor del frontend depende del backend, y el backend depende de la base de datos.

6. Pruebas y ajustes.

Finalmente, se realizaron pruebas exhaustivas para asegurar que todos los servicios se iniciaran correctamente y se comunicaran entre sí. Se hicieron ajustes en las configuraciones según fuera necesario para optimizar el rendimiento y asegurar la estabilidad del sistema.

Despliegue con AWS EC2

Para desplegar TheSphere en un entorno de pruebas, utilizamos Amazon Web Services (AWS) EC2, que proporciona una infraestructura escalable y fiable para alojar nuestra aplicación. A continuación, se detallan los pasos para configurar y desplegar la aplicación en AWS EC2.

Creamos una instancia t2.large con 8gb de ram con Amazon Linux en eu-west-2 para correr nuestra aplicación en un entorno de pruebas controlado, para ello debemos generar y crear la clave de par para poder conectarnos e instalar los componentes necesarios. A continuación, se listan los pasos que seguimos para configurar la instancia:

1. Generar y descargar la clave de de par

Durante la configuración de la instancia, genera una nueva clave de par (archivo .pem) y descárgala. Esta clave se utilizará para acceder a la instancia EC2 mediante SSH.

2. Configurar la seguridad

Debemos añadir nuevas reglas de entrada para exponer los puertos de nuestros servicios. Para ello navegamos hasta el apartado de seguridad y añadimos una regla personalizada con protocolo TCP y origen 0.0.0.0/0.

- Abrir el puerto 5173 para el Frontend
- Abrir el puerto 8081 para phpMyAdmin

3. Instalamos git para poder clonar el repositorio

```
sudo yum install -y git  
git clone https://github.com/BreaGG/TheSphere
```

4. Instalamos Docker

```
sudo yum install -y docker
```

5. Instalamos Docker Compose

```
sudo curl -L https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-  
$(uname -m) -o /usr/local/bin/docker-compose  
  
sudo chmod +x /usr/local/bin/docker-compose  
  
docker-compose version
```

Una vez tenemos todo listo e instalado navegamos a la raíz de nuestro repositorio para levantar los contenedores y tener acceso a nuestra aplicación a través de la ip pública que nos genera AWS con el puerto de nuestro frontend (5173).

6. Levantamos nuestro contenedor y ya tenemos todo listo

```
cd TheSphere/  
  
Docker-compose up --build
```

Funcionalidades Principales

1. Registro y Autenticación de Usuarios: Los usuarios pueden registrarse proporcionando un nombre de usuario, correo electrónico y contraseña.
2. Perfil de Usuario: Los usuarios pueden ver y editar su perfil, incluyendo su foto de perfil y banner.
3. Publicación de Posts: Los usuarios pueden crear, editar y eliminar publicaciones. Cada publicación puede incluir un título, subtítulos, descripciones, categoría y una imagen.
4. Soporte para Markdown: Los usuarios pueden escribir bloques de código en las publicaciones gracias al soporte en los posts para mostrar código Markdown. Además, este código puede ser copiado fácilmente al hacer clic en el botón de copiar, aparece un tooltip indicando que el código ha sido copiado al portapapeles.

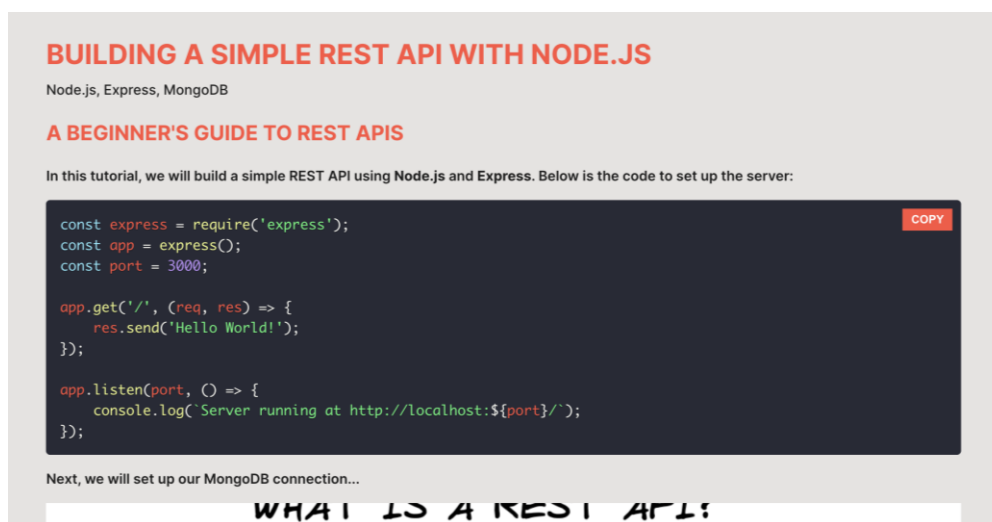


Figura 13: Ejemplo proyecto que usa Markdown.

5. Comentarios en los Posts: Los usuarios pueden comentar en las publicaciones de otros usuarios.
6. Búsqueda y Filtrado de Posts: Los usuarios pueden buscar publicaciones utilizando una barra de búsqueda y filtrar publicaciones por categoría (DESIGN o DEVELOPMENT).

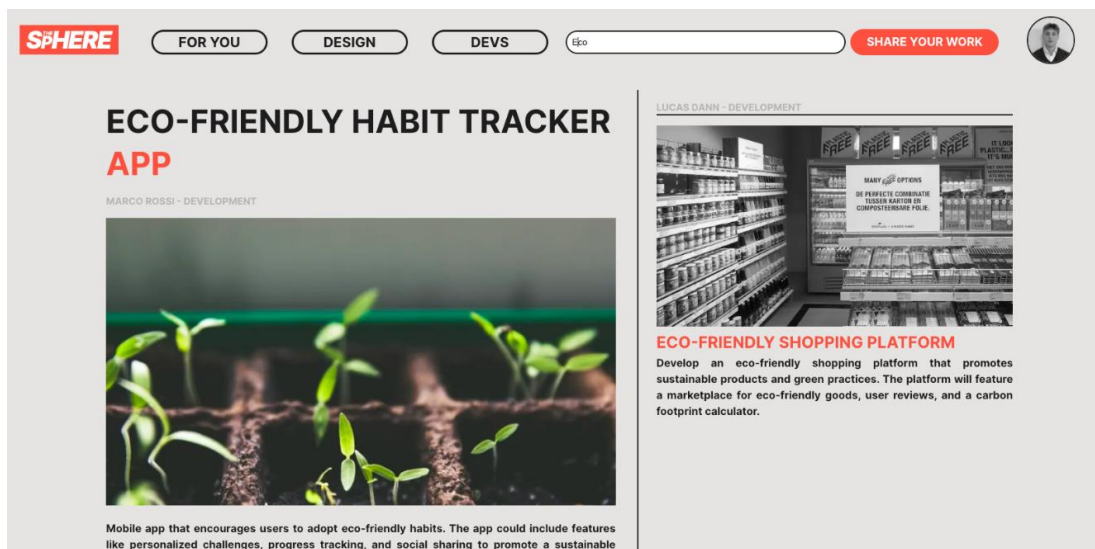


Figura 14: Ejemplo de feed buscando por "Eco"

Ejemplo de Flujo de Trabajo

1. Inicio de Sesión: El usuario navega a la página de inicio de sesión, ingresa sus credenciales y accede a la aplicación.

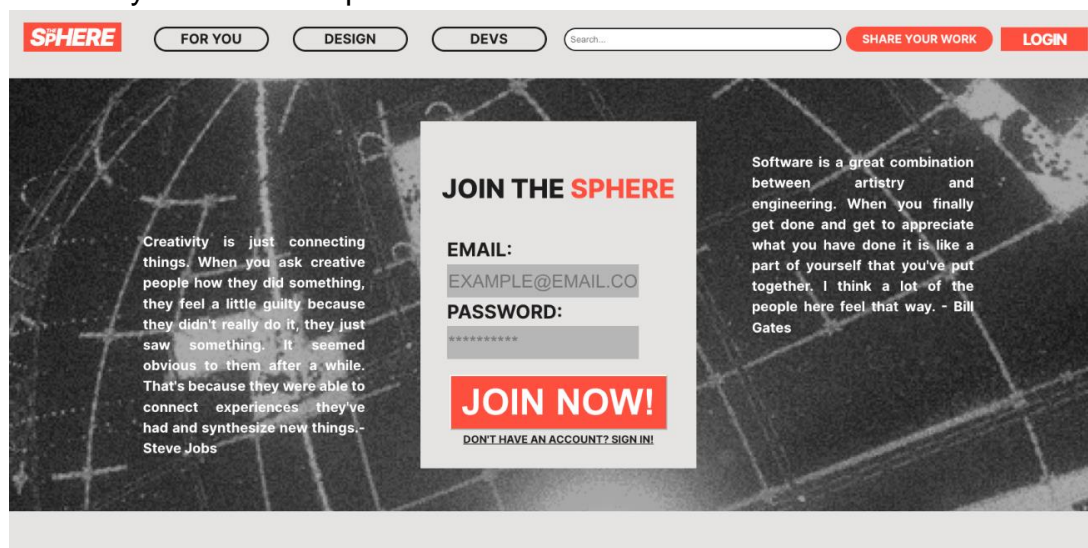


Figura 15: Pantalla de Login.

2. Creación de un Post:
 - a. El usuario navega a la página de creación de posts, llena el formulario y sube una imagen.
 - b. La imagen se sube a Cloudinary y se obtiene una URL segura.
 - c. El post se guarda en la base de datos y se muestra en el feed de la aplicación.
3. Comentario en un Post:
 - a. El usuario navega a los detalles de un post y agrega un comentario.
 - b. El comentario se guarda en la base de datos y se muestra inmediatamente en la sección de comentarios del post.

Manual Administrador

Requisitos Previos

Antes de comenzar, asegúrese de tener instalados los siguientes componentes en su sistema (Si despliega con Docker solo necesita Docker como requisito previo):

- Node.js (v16 o superior)
- npm (v6 o superior)
- JDK (Java Development Kit) 17
- Maven (v3.6 o superior)
- MySQL (o cualquier otra base de datos compatible con JDBC)
- Docker y Docker Compose

Opciones de instalación

Existen dos formas de instalar y correr la aplicación, desplegando el contenedor Docker o instalando en local las dependencias. Recomendamos encarecidamente desplegar con Docker para evitar incompatibilidades.

Instalación con Docker Compose (Recomendado)

1. Clonar el repositorio:

```
git clone https://github.com/BreaGG/TheSphere
```

2. Construir y levantar los contenedores:

```
docker-compose up --build
```

Este comando construirá las imágenes Docker para el frontend y el backend, y luego levantará todos los servicios definidos en docker-compose.yml.

3. Acceder al servicio:

- a. El frontend estará disponible en <http://localhost:5173>
- b. El backend estará disponible en <http://localhost:8080>

Instalación del Frontend (React + Vite)

4. Clonar el repositorio:

```
git clone https://github.com/BreaGG/TheSphere  
cd Fronted
```

5. Instalar las dependencias:

```
npm install
```

6. Iniciar el servidor de desarrollo:

```
npm run dev
```

Esto iniciará el servidor de desarrollo y la aplicación estará disponible en <http://localhost:5137>.

7. Compilar la aplicación para producción:

```
npm run build
```

8. Previsualizar la aplicación compilada:

```
npm run preview
```

Instalación del Backend (Spring Boot)

1. Clonar el repositorio

```
git clone https://github.com/BreaGG/TheSphere  
cd Backend
```

2. Configurar la base de datos:

Cree una base de datos en MySQL y actualice el archivo `application.properties` en el proyecto Spring Boot con las credenciales de la base de datos.

```
spring.datasource.url=jdbc:mysql://mysql:3306/tu_base_de_datos  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.username= mi_usuario  
spring.datasource.password= mi_contraseña  
server.port=8080  
  
spring.jpa.generate-ddl=true  
spring.jpa.hibernate.ddl-auto=create-drop  
spring.jpa.defer-datasource-initialization=true  
spring.sql.init.mode=always  
spring.jpa.show-sql=true  
  
spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
```

3. Compilar y ejecutar la aplicación:

```
./mvn clean install
```

```
./mvn spring-boot:run
```

Esto iniciará el servidor backend y estará disponible en <http://localhost:8080>.

4. Scripts SQL para la Base de Datos

Asegúrese de tener un script SQL para cargar los datos de prueba. Si no tienes uno puedes usar el Data.sql de la aplicación, puedes obtenerlo [aquí](#).

Notas Adicionales

Asegúrese de que el puerto 3306 de MySQL y los puertos 3000 y 8080 para las aplicaciones frontend y backend respectivamente, estén abiertos y disponibles.

Para personalizar la configuración del servidor de desarrollo o de la compilación, modifique los scripts en el archivo package.json y las propiedades en el archivo pom.xml y application.properties.

Acceso al Modo Admin

Para acceder al modo administrador en TheSphere, sigue estos pasos:

1. Iniciar Sesión como Admin:

- Dirígete a la página de inicio de sesión de TheSphere.
- Ingresa las credenciales de administrador:
- Correo Electrónico: `admin@thesphere`
- Contraseña: `admin`
- Haz clic en "Iniciar Sesión".

2. Activación del Modo Admin:

- Una vez que hayas iniciado sesión con las credenciales de administrador, se habilitará automáticamente un botón en la parte superior de la pantalla.
- Este botón te llevará al "Admin Dashboard".

3. Uso del Admin Dashboard:

- En el Admin Dashboard, tendrás la capacidad de eliminar usuarios y publicaciones.

- b. La interfaz mostrará dos columnas principales: "Users" (Usuarios) y "Posts" (Publicaciones).
- c. Cada fila en estas columnas tendrá un botón "Delete" que te permitirá eliminar el usuario o la publicación correspondiente.

Capturas de Pantalla del Dashboard

A continuación, se muestra una captura de pantalla del Admin Dashboard:

ADMIN DASHBOARD			
USERS		POSTS	
Marco Rossi - usuario2@example.com	Delete	Accesible Ticket Website - Marco Rossi	Delete
Javier Galifanes - usuario3@example.com	Delete	Eco-friendly Habit Tracker App - Marco Rossi	Delete
Teo Gallego - usuario4@example.com	Delete	AI-Powered Personal Health Assistant - Javier Galifanes	Delete
Lola Bermudez - usuario5@example.com	Delete	Branding and Packaging for Sustainable Products - Javier Galifanes	Delete
Adriano Fric - chele@mail.com	Delete	Immersive Vr Art Gallery - Javier Galifanes	Delete
Manuel Casal - manu@mail.com	Delete	Minimalist Home Automation Interface - Teo Gallego	Delete
Gustavo Nieto - gus@mail.com	Delete	Comprehensive cryptocurrency portfolio - Teo Gallego	Delete
Carmen Faginas - mela@mail.com	Delete	Chatas Ibiza - Lola Bermudez	Delete
Lucia Parra - ina@mail.com	Delete	Luminous Installation - Lola Bermudez	Delete
Malena Fariña - malena@mail.com	Delete	Virtual FashionWeek - Lola Bermudez	Delete
Sergio Sanchez - sergios@mail.com	Delete	Smart Garden Monitoring System - Sergio Sanchez	Delete
Alejandro Brea - brea@mail.com	Delete	Interactive Portfolio Website - Marco Rossi	Delete
Anail Perez - anail@mail.com	Delete	AI-Powered Recipe App - Javier Galifanes	Delete
Marta Vilas - marta@mail.com	Delete	Virtual Reality Travel Experience - Anail Perez	Delete
Lucas Dann - lucas@mail.com	Delete	Eco-Friendly Shopping Platform - Lucas Dann	Admin Dashboard

Figura 16: Ejemplo de Dashboard de administrador.

Descripción de la Interfaz:

- Users: Lista de usuarios registrados en la plataforma. Cada entrada incluye el nombre del usuario y su correo electrónico.
- Posts: Lista de publicaciones realizadas por los usuarios. Cada entrada incluye el título del post y el nombre del autor.
- Botón Delete: Botón naranja al lado de cada usuario y publicación para eliminarlos del sistema.

Acceso al Admin Dashboard

1. Botón de Acceso: El botón "Admin Dashboard" aparece en la parte inferior derecha después de iniciar sesión con las credenciales de administrador.
2. Funcionalidad del Botón: Al hacer clic en este botón, serás redirigido al Admin Dashboard, donde puedes gestionar usuarios y publicaciones.

Ejemplo de Flujo de Trabajo

1. Iniciar Sesión:

- Ingresa *admin@thesphere* como correo electrónico y *admin* como contraseña.

2. Acceso al Dashboard:

- Haz clic en el botón "Admin Dashboard" que aparece en la parte inferior derecha.

3. Eliminar Usuario o Publicación:

- En la sección "Users" o "Posts", haz clic en el botón "Delete" junto al usuario o publicación que deseas eliminar.

4. Confirmar eliminación:

- Una vez clicado el botón "Delete" saltará un modal para confirmar la acción.

Esta guía te ayudará a gestionar eficientemente los usuarios y publicaciones en TheSphere, asegurando que tengas el control total sobre el contenido de la plataforma.

Manual Usuario

Página de Inicio

La página de inicio da la bienvenida a los usuarios y ofrece una breve descripción de la plataforma.

- **Elementos:**
 - Logo: Enlace a la página de inicio.
 - Menú de Navegación: Enlaces a las secciones principales de la aplicación, como "Contactar" y "Login/Register".
 - Descripción: Breve resumen de la plataforma y sus objetivos.

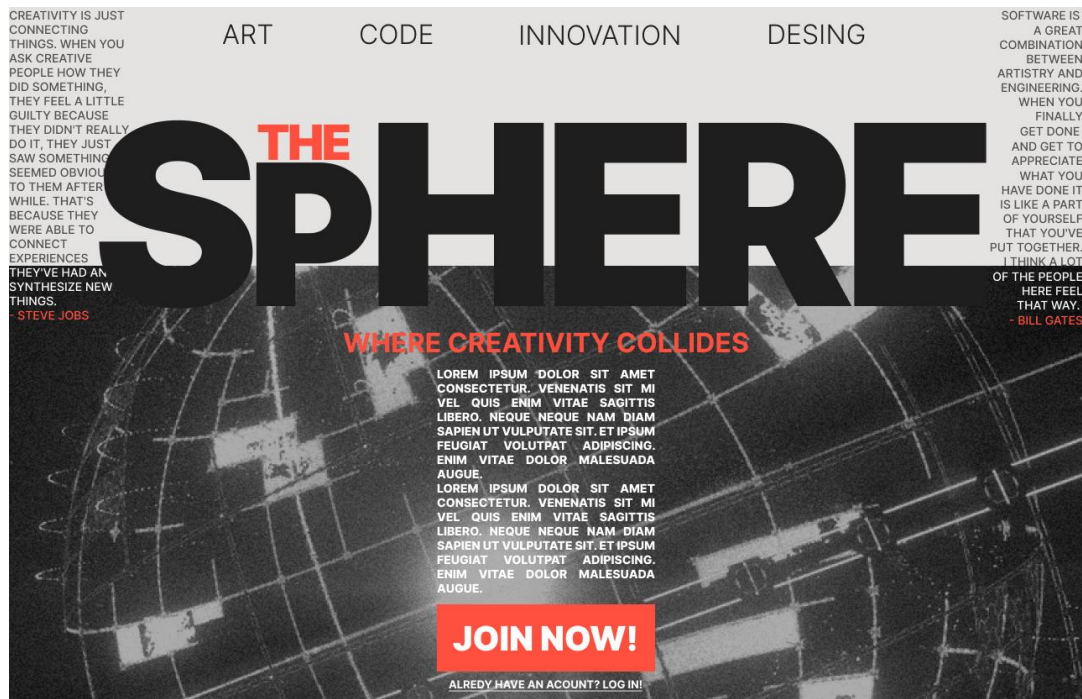


Figura 17: Mockup pantalla de home.

Feed de Publicaciones

El feed de publicaciones muestra las publicaciones recientes de los usuarios, con opciones para filtrarlas por categoría (Diseño o Desarrollo).

- Elementos:
 - Barra de Búsqueda: Permite a los usuarios buscar publicaciones por título o palabra clave.
 - Filtros: Opción para filtrar publicaciones por categoría.
 - Lista de Publicaciones: Muestra las publicaciones con título, autor, categoría, imagen y descripción corta. Cada publicación es un enlace a su detalle.

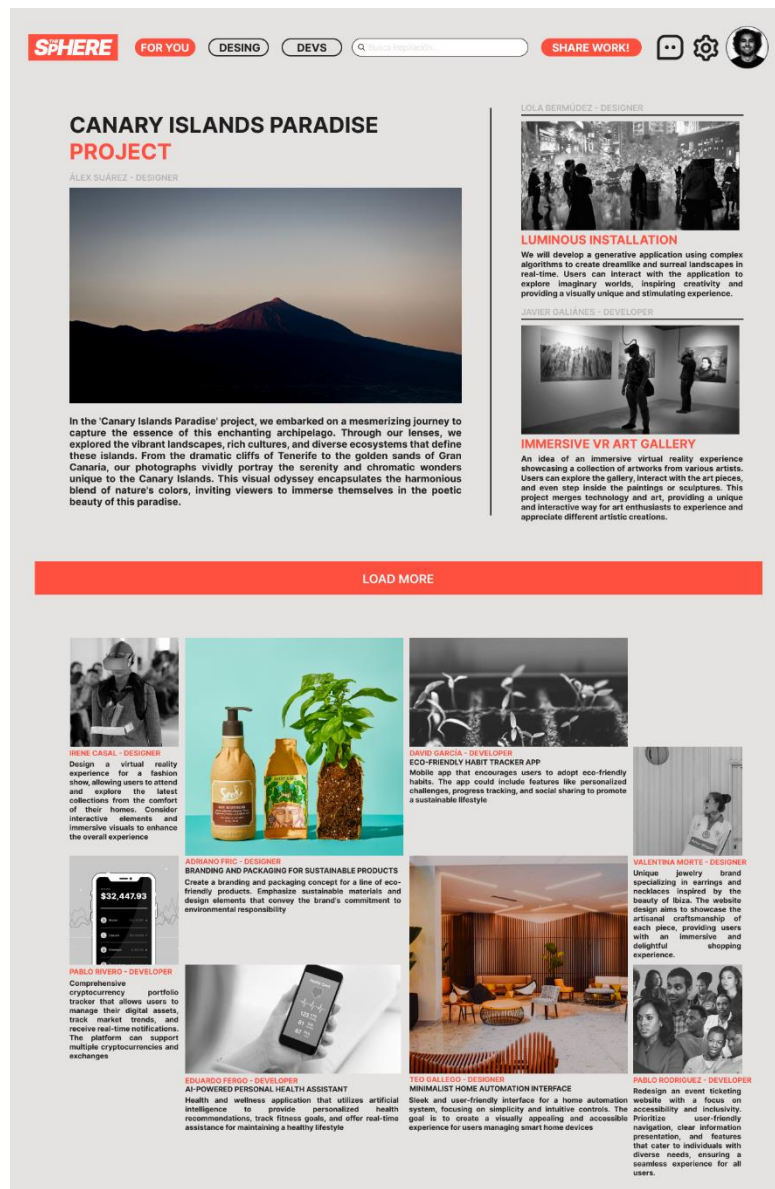


Figura 18: Ejemplo feed completo

Detalle de la Publicación

La página de detalle de la publicación muestra toda la información relacionada con una publicación específica, incluyendo los comentarios de otros usuarios.

- Elementos:
 - Título y Descripción: Título de la publicación y su descripción completa. La descripción admite formato Markdown, por si el usuario quiere introducir bloques de código o usar este formato.
 - Imagen: Imagen destacada de la publicación.
 - Autor: Información sobre el autor de la publicación con enlace a su perfil.
 - Copiado de Código: Los usuarios pueden copiar bloques de código incluidos en la descripción. Al hacer clic en el botón "Copy", aparece un tooltip que indica "Code copied to clipboard".
 - Comentarios: Lista de comentarios hechos por otros usuarios. Cada comentario incluye el perfil del usuario que lo hizo, su nombre y el contenido del comentario.
 - Formulario de Comentario: Permite a los usuarios añadir comentarios a la publicación.

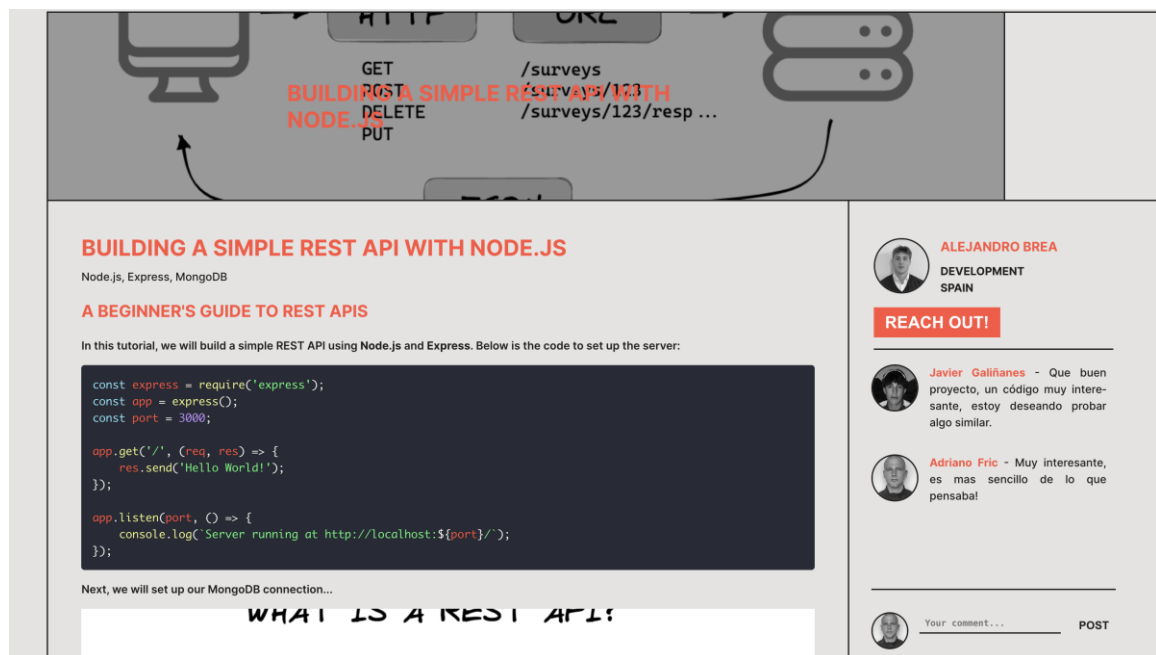


Figura 19: Ejemplo detalles proyecto completo.

Perfil de Usuario

La página de perfil de usuario muestra la información del perfil del usuario, incluyendo su biografía, publicaciones y opciones para editar su perfil.

- Elementos:
 - Imagen de Perfil y Banner: Imagen de perfil y banner del usuario.
 - Información del Usuario: Nombre de usuario, país, biografía.
 - Lista de Publicaciones: Publicaciones realizadas por el usuario.
 - Botón de Edición: Permite al usuario editar su perfil si está viendo su propio perfil.

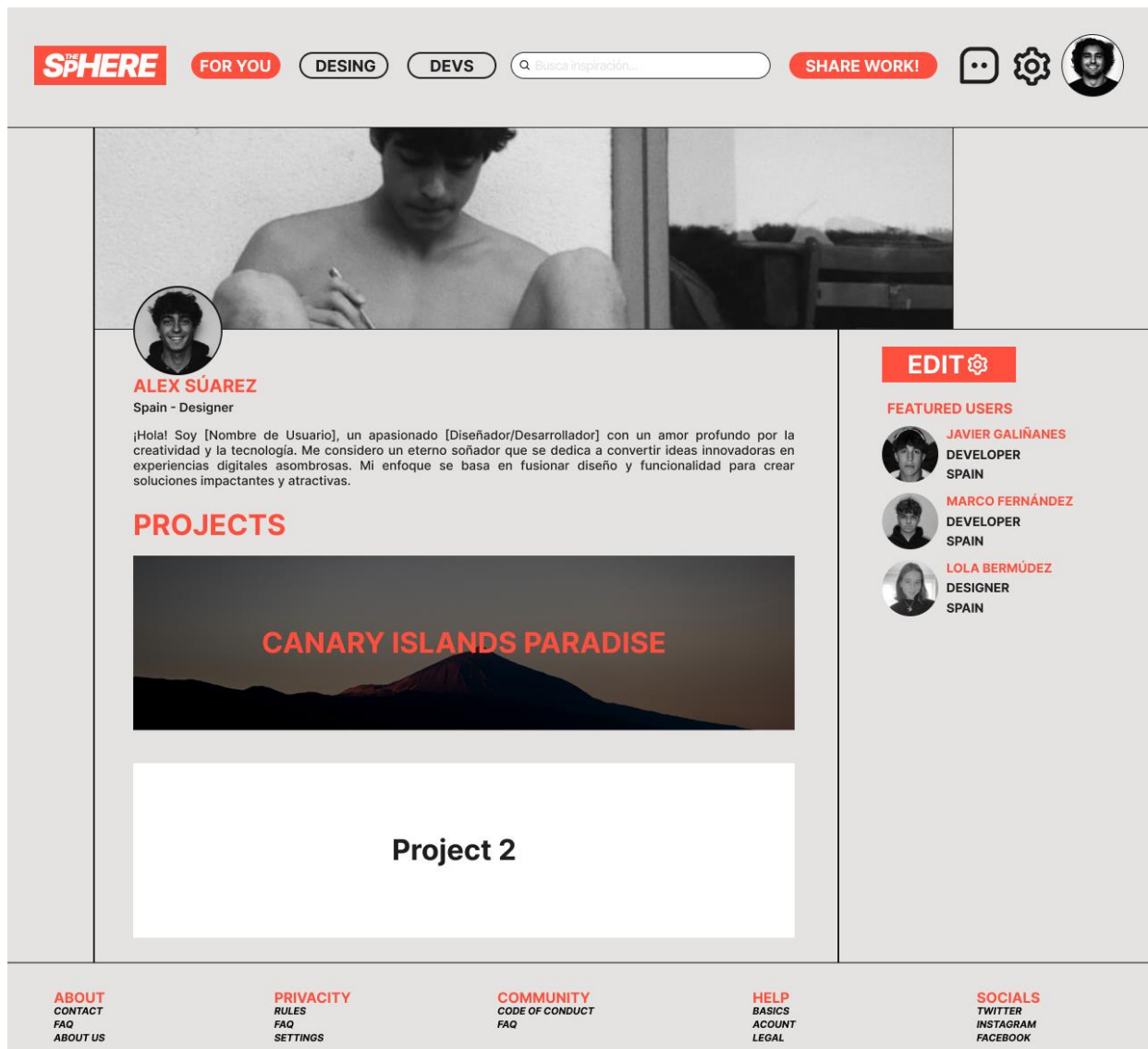
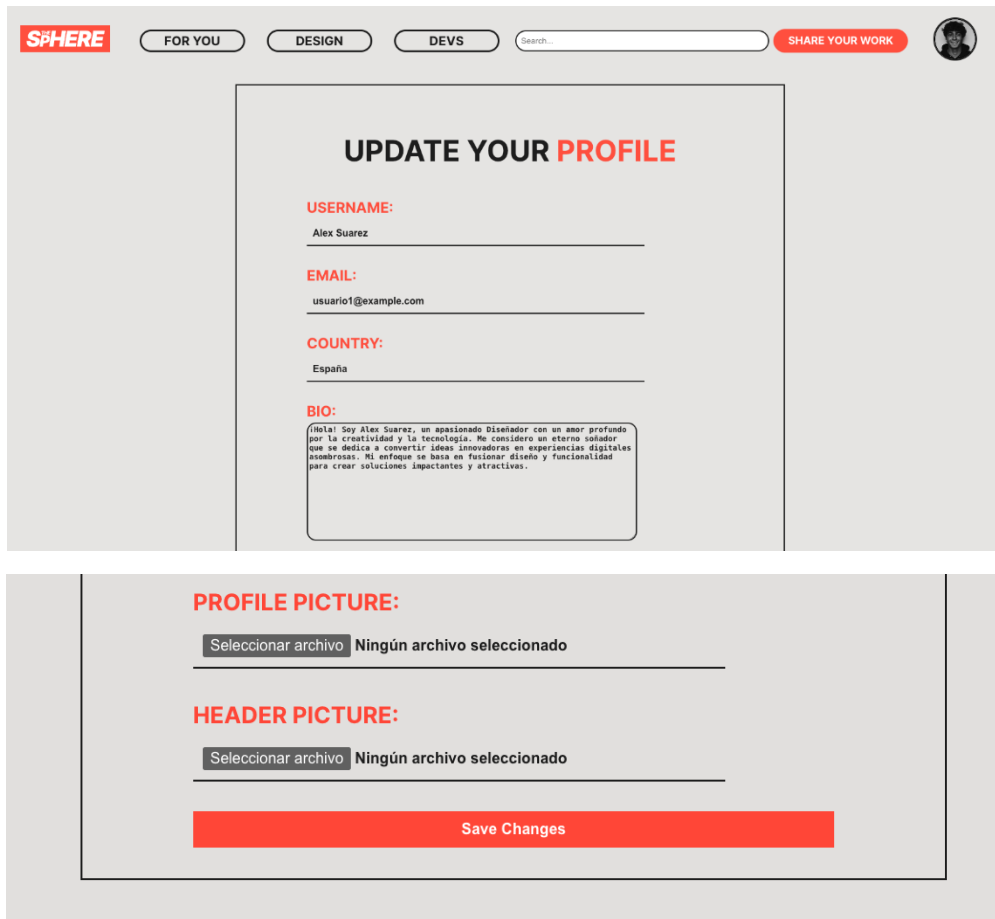


Figura 20: Mockup Ejemplo perfil de usuario logeado.

Edición de Perfil

La página de edición de perfil permite a los usuarios actualizar su información personal, como nombre de usuario, email, país, biografía y subir nuevas imágenes de perfil y banner.

- Elementos:
 - Formulario de Edición: Campos para editar el nombre de usuario, email, país y biografía.
 - Subida de Imágenes: Permite subir nuevas imágenes para el perfil y el banner. Las imágenes se cargan en Cloudinary y se muestran en el perfil del usuario.
 - Botón Guardar Cambios: Guarda los cambios realizados y actualiza la información del perfil.



The screenshot displays the 'UPDATE YOUR PROFILE' form within the SPHERE application. The top navigation bar includes the 'SPHERE' logo, tabs for 'FOR YOU', 'DESIGN', and 'DEVS', a search bar, and a 'SHARE YOUR WORK' button next to a user profile picture. The main form area is titled 'UPDATE YOUR PROFILE' and contains the following fields:

- USERNAME:** A text input field with the value 'Alex Suarez'.
- EMAIL:** A text input field with the value 'usuario1@example.com'.
- COUNTRY:** A text input field with the value 'España'.
- BIO:** A text area containing a bio: '¡Hola! Soy Alex Suarez, un apasionado Diseñador con un amor profundo por la creatividad y la tecnología. Me considero un eterno aprendiz que se dedica a convertir ideas innovadoras en experiencias digitales asombrosas. Mi enfoque se basa en fusionar diseño y funcionalidad para crear soluciones impactantes y atractivas.'

Below the text fields, there are two sections for image uploads:

- PROFILE PICTURE:** A button labeled 'Seleccionar archivo' and the text 'Ningún archivo seleccionado'.
- HEADER PICTURE:** A button labeled 'Seleccionar archivo' and the text 'Ningún archivo seleccionado'.

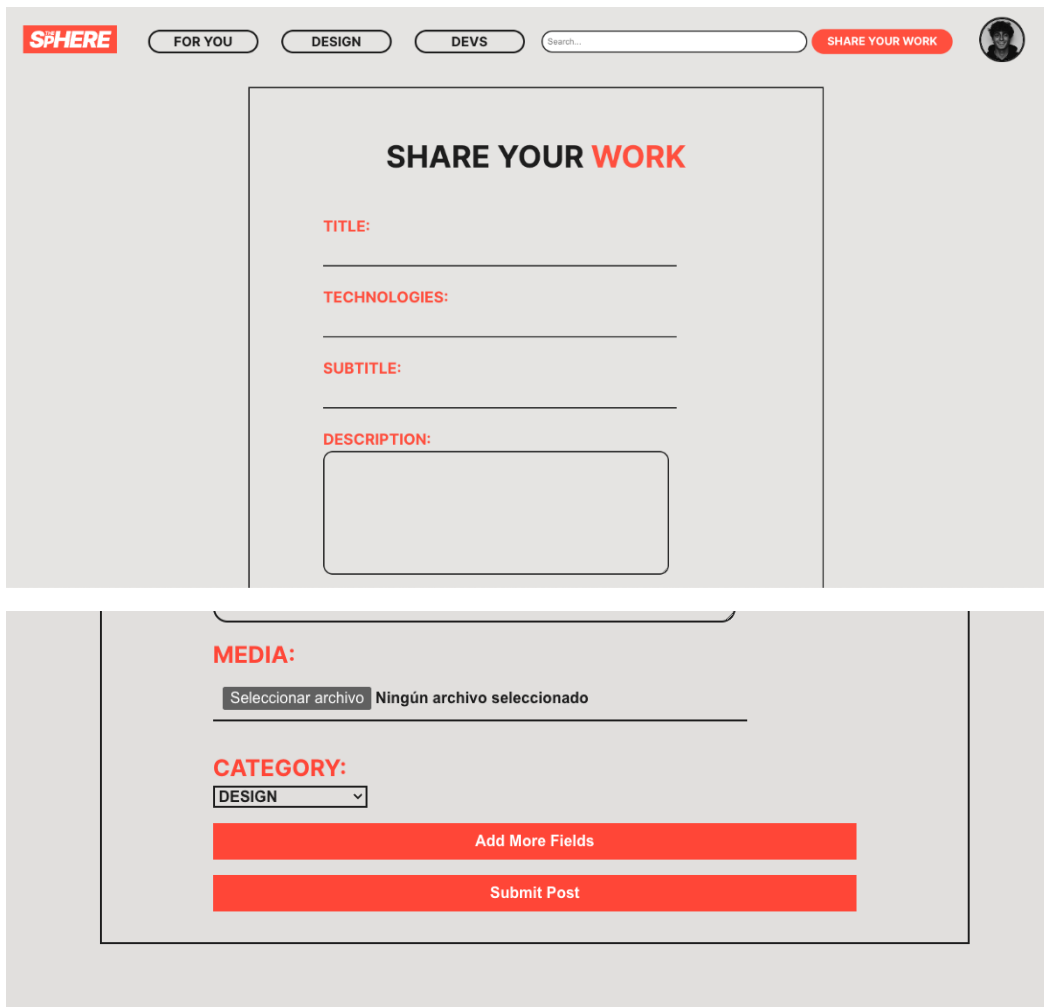
At the bottom of the form is a large red button labeled 'Save Changes'.

Figura 21: Ejemplo formulario de edición de perfil.

Publicar Nuevo Post

La página de creación de publicaciones permite a los usuarios crear una nueva publicación, proporcionando título, tecnologías utilizadas, descripciones, categoría e imagen.

- Elementos:
 - Formulario de Publicación: Campos para ingresar el título, tecnologías, descripciones, categoría e imagen.
 - Subida de Imagen: Permite subir una imagen para la publicación, que se almacena en Cloudinary.
 - Botón Publicar: Envía la publicación al servidor para que sea visible en el feed de publicaciones.



The image shows a web application interface for creating a post. At the top, there is a navigation bar with the 'SPHERE' logo, tabs for 'FOR YOU', 'DESIGN', and 'DEVS', a search bar, and a 'SHARE YOUR WORK' button next to a user profile icon. The main content area is titled 'SHARE YOUR WORK' and contains a form with the following fields: 'TITLE:', 'TECHNOLOGIES:', 'SUBTITLE:', and 'DESCRIPTION:'. Below these fields is a 'MEDIA:' section with a file selection button labeled 'Seleccionar archivo' and the text 'Ningún archivo seleccionado'. A 'CATEGORY:' dropdown menu is set to 'DESIGN'. At the bottom of the form are two red buttons: 'Add More Fields' and 'Submit Post'.

Figura 22: Ejemplo formulario de creación de proyecto.

Contactar

La página de contacto proporciona información sobre cómo ponerse en contacto con los administradores de TheSphere.

- Elementos:
 - Dirección: Dirección física de la oficina.
 - Correo Electrónico: Dirección de correo electrónico para contactar.
 - Teléfono: Número de teléfono para consultas.

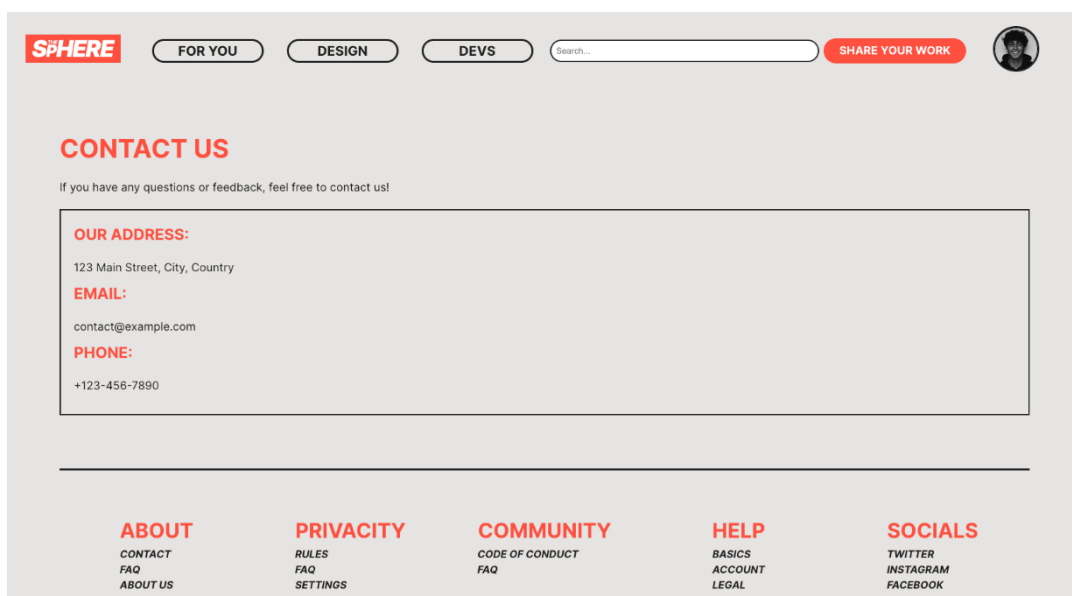


Figura 23: Página de contacto.

Modo alta accesibilidad

El botón de la parte inferior a la derecha permite al usuario alternar los colores de TheSphere a unos de mayor contraste y accesibilidad, cambiando nuestro característico Naranja por un negro sobre fondo blanco además las imágenes cobran color.

- Elementos:
 - Botón: En la parte inferior a la derecha.

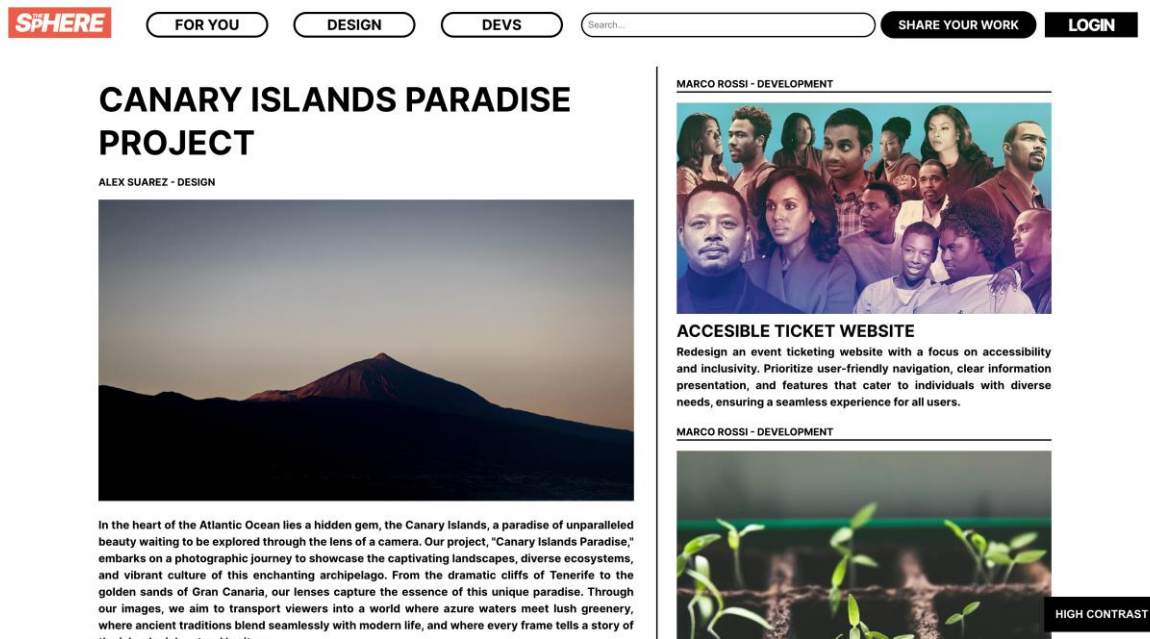


Figura 24: Feed de alto contraste.

Explicación de Funcionamiento

- Navegación: Los usuarios pueden navegar entre las diferentes secciones de la aplicación utilizando el menú de navegación en la parte superior.
- Autenticación: Los usuarios deben registrarse e iniciar sesión para crear publicaciones, comentar y editar su perfil.
- Interacciones: Los usuarios pueden interactuar con las publicaciones mediante comentarios y pueden seguir enlaces a perfiles de otros usuarios.
- Gestión de Contenidos: Las imágenes subidas por los usuarios se almacenan en la Base de Datos.

Viabilidad tecno-económica

Análisis de Mercado

Para entender la viabilidad tecno-económica de TheSphere, es crucial realizar un análisis del mercado. A continuación, se presenta un análisis de mercado que considera las tendencias actuales, el tamaño del mercado, los competidores.

Tamaño del Mercado y Tendencias

El mercado de redes sociales es vasto y en constante crecimiento. Según estadísticas recientes, el número de usuarios de redes sociales en todo el mundo supera los 3.6 mil millones y se espera que alcance los 4.4 mil millones en 2025. Dentro de este mercado, hay un segmento emergente y creciente de plataformas que se especializan en nichos específicos, como las redes profesionales y las comunidades creativas.

TheSphere se posiciona en la intersección de dos nichos clave:

- Redes Profesionales: Plataformas como LinkedIn y Kaggle han demostrado que hay una gran demanda de espacios donde profesionales de diversos campos pueden conectarse, colaborar y avanzar en sus carreras.
- Comunidades Creativas: Sitios como Behance y Dribbble han mostrado el valor de proporcionar un espacio para que los creativos compartan su trabajo y reciban retroalimentación.

Competidores

- LinkedIn
 - Puntos Fuertes: Gran base de usuarios, funcionalidades de búsqueda de empleo, amplia red profesional.
 - Puntos Débiles: Menos enfoque en la creatividad y el diseño, interfaz más orientada a los negocios.
- Behance
 - Puntos Fuertes: Fuerte comunidad de diseñadores, excelente plataforma para portafolios visuales.
 - Puntos Débiles: Enfocado principalmente en diseñadores, falta de integración con desarrolladores técnicos.

- Dribbble
 - Puntos Fuertes: Comunidad vibrante de diseñadores gráficos, plataforma para mostrar trabajos creativos.
 - Puntos Débiles: Similar a Behance, se centra más en el diseño que en el desarrollo técnico.
- Kaggle
 - Puntos Fuertes: Fuerte comunidad de científicos de datos, plataforma para competencias y colaboraciones.
 - Puntos Débiles: Enfocado en datos y ciencia, menos atractivo para diseñadores y creativos.

Análisis DAFO

El análisis DAFO (Debilidades, Amenazas, Fortalezas y Oportunidades) es una herramienta estratégica utilizada para identificar y evaluar los factores internos y externos que pueden afectar el éxito del proyecto TheSphere. A continuación, se presenta un análisis DAFO del proyecto:

Fortalezas

1. Tecnología Actual: TheSphere utiliza tecnologías modernas como ReactJS para el frontend y Spring Boot para el backend, lo que garantiza una arquitectura robusta y escalable.
2. Integración de Microservicios: La implementación de microservicios como la API de Cloudinary para la gestión de imágenes asegura una funcionalidad eficiente y especializada.
3. Enfoque en la Comunidad: La plataforma está diseñada para unir perfiles técnicos y artísticos, lo que fomenta la colaboración y el intercambio de ideas entre diferentes disciplinas.
4. Desarrollo Completo por el Autor: El control completo del desarrollo por una sola persona asegura coherencia y una visión unificada del proyecto.

Debilidades

1. Recursos Limitados: El desarrollo por una sola persona puede ser un cuello de botella, limitando la velocidad de desarrollo y la capacidad de manejo de tareas simultáneas.
2. Dependencia en Terceros: La utilización de servicios externos como Cloudinary y AWS implica una dependencia en la estabilidad y los costos de estos servicios.

3. Curva de Aprendizaje: La adopción de nuevas tecnologías puede requerir tiempo adicional para la formación y la implementación correcta.

Oportunidades

1. Creciente Demanda de Plataformas Colaborativas: Con el aumento del trabajo remoto y la colaboración en línea, existe una gran oportunidad para que TheSphere se posicione como una plataforma líder para profesionales técnicos y artísticos.
2. Expansión del Mercado: La posibilidad de expandir la plataforma a otros mercados y segmentos de usuarios, como educadores y estudiantes, puede aumentar significativamente la base de usuarios.
3. Innovación Continua: Las oportunidades de incorporar nuevas tecnologías y funcionalidades, como inteligencia artificial para recomendaciones y análisis de contenido, pueden mejorar la propuesta de valor de la plataforma.

Amenazas

1. Competencia Intensa: Existen muchas plataformas establecidas como Behance, Dribbble y Kaggle que compiten por la atención de profesionales técnicos y creativos.
2. Seguridad y Privacidad: Las preocupaciones sobre la seguridad y la privacidad de los datos pueden afectar la confianza de los usuarios y la adopción de la plataforma.
3. Cambios en las Políticas de Servicios Externos: Cambios en las políticas de precios o disponibilidad de los servicios externos utilizados (como Cloudinary y AWS) pueden afectar los costos operativos y la viabilidad a largo plazo.

DEBILIDADES <ul style="list-style-type: none"> • Recursos limitados • Dependencia en terceros • Curva de aprendizaje 	AMENAZAS <ul style="list-style-type: none"> • Competencia intensa • Seguridad y privacidad • Cambios en las políticas de servicios externos
FORTALEZAS <ul style="list-style-type: none"> • Tecnología actual • Integración de microservicios • Enfoque en la comunidad • Desarrollo completo por el autor 	OPORTUNIDADES <ul style="list-style-type: none"> • Creciente demanda de plataformas colaborativas • Expansión del mercado • Innovación continua

Figura 25: Diagrama análisis DAFO. Elaboración propia.

Coste de Implementación

La implementación de TheSphere requiere una inversión inicial y costos operativos continuos. A continuación, se desglosan los principales costos asociados al desarrollo, implementación y mantenimiento de la plataforma, considerando que yo seré el único desarrollador y diseñador del proyecto.

1. Hardware y Servidores

- Servidores de Desarrollo y Producción: Utilizando AWS para el despliegue de la aplicación, se estima un coste mensual de 100€ a 200€, dependiendo de la capacidad requerida.
- Equipos de Desarrollo: Ordenador personal y periféricos para el desarrollo. Se estima un costo inicial de 2.000€ para equipar un entorno de trabajo adecuado.

2. Software y Herramientas

- Licencias de Software: Herramientas de diseño como Adobe Creative Cloud (59,99€/mes).
- Servicios en la Nube: Utilización de Snowflake para la base de datos, con un costo estimado de 200€ mensuales.
- API de Terceros: Costes asociados al uso de APIs externas, como Cloudinary para almacenamiento de imágenes (89€/mes para el plan avanzado).

3. Desarrollo y Personal

- Desarrollador y Diseñador: No hay costos de salario adicionales ya que seré el único desarrollador y diseñador.
- Contratistas/Freelancers: Podrían necesitarse servicios adicionales como marketing y legales. Se estima un costo de 1.000€ a 2.000€ mensuales.

4. Marketing y Publicidad

- Lanzamiento y Promoción: Campañas iniciales de marketing y publicidad en redes sociales, Google Ads, etc., se estiman en 5.000€ para los primeros tres meses.
- Marketing Continuo: Costos de marketing mensual de 500€ a 1.000€ para mantener la visibilidad de la plataforma.

5. Otros Gastos Operativos

- Mantenimiento y Soporte: Costos de mantenimiento continuo del sistema, soporte al usuario, y mejoras de la plataforma, estimados en 500€ mensuales.
- Gastos Administrativos: Alquiler de oficina, servicios públicos, y otros gastos administrativos se estiman en 500€ mensuales.

Resumen de Costes Iniciales y Operativos

Concepto	Coste Inicial	Coste Mensual
Servicios y Hardware	2.000€	200€
Software y Licencias	-	150€
Servicios Cloud	-	200€
Contratista/Freelancer	-	1.500€
Marketing y Publicidad	5.000€	750€
Mantenimiento y Soporte	-	500€
Gastos Administrativos	500€	500€
Total	7.500€	3.800€

Figura 26: Tabla comparativa de costes. Elaboración propia.

Escenarios de Proyección de Ingresos

Escenario Optimista

- Supuestos:
 - Usuarios iniciales: 1.000 en los primeros 6 meses.
 - Tasa de conversión a Premium: 20%.
 - Ingresos por publicidad y comisiones: 500 €/mes.
- Ingresos Mensuales:
 - Usuarios Premium: $200 * 10 \text{ €} = 2.000 \text{ €}$
 - Publicidad y comisiones: 500 €
 - Total: 2.500 €/mes
- Punto Muerto:
 - Usuarios necesarios para cubrir costes: $3.800 \text{ €} / 2.5 \text{ €} \approx 1.520$ usuarios.

Escenario Realista

- Supuestos:
 - Usuarios iniciales: 500 en los primeros 6 meses.
 - Tasa de conversión a Premium: 10%.
 - Ingresos por publicidad y comisiones: 200 €/mes.
- Ingresos Mensuales:
 - Usuarios Premium: $50 * 10 \text{ €} = 500 \text{ €}$
 - Publicidad y comisiones: 200 €
 - Total: 700 €/mes
- Punto Muerto:
 - Usuarios necesarios para cubrir costes: $3.800 \text{ €} / 1.4 \text{ €} \approx 2.714$ usuarios.

Escenario Pesimista

- Supuestos:
 - Usuarios iniciales: 300 en los primeros 6 meses.
 - Tasa de conversión a Premium: 5%.
 - Ingresos por publicidad y comisiones: 100 €/mes.
- Ingresos Mensuales:
 - Usuarios Premium: $15 * 10 \text{ €} = 150 \text{ €}$
 - Publicidad y comisiones: 100 €
 - Total: 250 €/mes
- Punto Muerto:
 - Usuarios necesarios para cubrir costes: $3.800 \text{ €} / 1 \text{ €} \approx 3.800$ usuarios

Análisis de Rentabilidad

Ingresos Potenciales

- Modelos de Ingresos:
 - Suscripción Premium: Usuarios pueden pagar una suscripción mensual de 10€ para acceder a funciones avanzadas.
 - Publicidad: Ingresos generados por publicidad integrada en la plataforma.
 - Comisiones por Freelance: Comisión del 5% al 10% por proyectos freelance gestionados a través de la plataforma.

Proyección de Usuarios e Ingresos

- Usuarios Iniciales: Se espera atraer a 500 usuarios en los primeros 6 meses.
- Tasa de Conversión a Premium: Estimamos que el 10% de los usuarios se convertirán en suscriptores premium.
- Ingresos Estimados:
 - 50 usuarios premium pagando 10€/mes: 500€/mes.
 - Publicidad y comisiones: 200€/mes.
 - Total Ingresos Mensuales: 700€/mes.

Cálculo del Punto Muerto

Para encontrar el número de usuarios necesarios para alcanzar el punto muerto:

$$\text{Usuarios Necesarios} = \frac{\text{Costos Fijos}}{\text{Ingreso Promedio por Usuario}}$$
$$\text{Usuarios Necesarios} = \frac{3.800}{1.4/\text{usuario}} \approx 2.714 \text{ usuarios}$$

Esto significa que, para alcanzar el punto muerto, TheSphere necesita aproximadamente 2.714 usuarios activos por mes.

- Mes 1 a 6:
 - Usuarios estimados: 500
 - Ingresos mensuales: 700€
 - Costos mensuales: 3.800€
 - Déficit mensual: 3.100€
- Acumulado al mes 6:

$$\text{Déficit Total} = 6 \times 3.100 = 18.600$$

Estrategia para Alcanzar el Umbral de Rentabilidad

1. Incrementar la Tasa de Conversión a Premium: Mejorar las funciones premium para atraer más usuarios a la suscripción.
2. Expandir la Publicidad: Aumentar las oportunidades de ingresos por publicidad mediante estrategias de marketing digital.

3. Promover Proyectos Freelance: Facilitar más proyectos freelance a través de la plataforma para aumentar las comisiones.
4. Atraer Nuevos Usuarios: Incrementar el marketing y las promociones para aumentar la base de usuarios activos.

Coste de Adquisición de Cliente (CAC)

El Coste de Adquisición de Cliente (CAC) es una métrica fundamental para evaluar la viabilidad tecno-económica del proyecto TheSphere. Esta métrica nos ayuda a entender cuánto cuesta adquirir un nuevo usuario en la plataforma, considerando todos los gastos relacionados con marketing y ventas.

1. Definición de CAC

El CAC se calcula sumando todos los costos de ventas y marketing durante un período específico y dividiéndolos por el número de nuevos clientes adquiridos en ese período.

$$\text{CAC} = \text{Gastos Totales de Marketing y Ventas} / \text{Nuevos Clientes Adquiridos}$$

2. Gastos de Marketing y Ventas

Componentes del gasto:

- Marketing Digital: Incluye campañas en redes sociales, Google Ads, y otros canales de publicidad en línea.
- Marketing de Contenidos: Creación de blogs, videos, y otros contenidos para atraer usuarios.
- Eventos y Promociones: Participación en eventos de la industria, promociones y descuentos.
- Costos Operativos de Ventas: Incluye salarios de personal de ventas (si aplica), herramientas de CRM y otros gastos relacionados.

3. Ejemplo de Cálculo de CAC

Supuestos:

- Gastos mensuales de marketing digital: 500€
- Gastos mensuales de marketing de contenidos: 250€
- Gastos en eventos y promociones: 1.500€
- Costes operativos de ventas: 500€
- Nuevos clientes adquiridos en el mes: 200

$$CAC = (500€ + 250€ + 1.500€ + 500€) / 200 = 3.500€ / 200 = 13,75€$$

Esto significa que el coste de adquirir un nuevo cliente para TheSphere es de 13,75€.

4. Análisis de CAC

Comparación con el Valor del Ciclo de Vida del Cliente (CLV):
Para determinar si el CAC es sostenible, debe compararse con el Valor del Ciclo de Vida del Cliente (CLV). El CLV representa el ingreso neto que se espera obtener de un cliente a lo largo de su relación con la empresa.

Supuestos de CLV:

- Suscripción mensual promedio: 10€
- Duración promedio de suscripción: 12 meses

$$CLV = 10€ \times 12 = 120€$$

Comparando el CLV con el CAC: Relación CLV/CAC = $120€ / 13,75€ \approx 6,72$

Una relación CLV/CAC mayor que 1 indica que el costo de adquisición está cubierto por los ingresos generados por cada cliente. En este caso, una relación de aproximadamente 6,86 es muy favorable, lo que sugiere que TheSphere tiene un modelo de adquisición de clientes rentable.

5. Estrategias para Optimizar el CAC

- **Mejorar la Eficiencia del Marketing:**
 - Segmentación y Personalización: Enfocar las campañas de marketing en segmentos específicos que tienen más probabilidades de convertirse en clientes de pago.
 - Optimización de Publicidad: Ajustar y optimizar continuamente las campañas de publicidad para maximizar el retorno de la inversión.
- **Aumentar la Tasa de Conversión:**
 - Mejora de la Experiencia del Usuario: Asegurarse de que la plataforma es intuitiva y ofrece valor desde el primer momento para convertir visitantes en usuarios registrados.
 - Pruebas A/B: Realizar pruebas A/B en diferentes elementos de la página de registro y de inicio para identificar y aplicar las mejores prácticas.
- **Fomentar el Marketing de Referencia:**

- Colaboraciones y Alianzas: Establecer colaboraciones con otras plataformas y organizaciones que puedan referir usuarios interesados.

Evaluación de Rentabilidad

Con una inversión inicial de 7.500€ y costos operativos mensuales de 3.800€, es crucial alcanzar una masa crítica de usuarios rápidamente para lograr la sostenibilidad financiera. Los ingresos iniciales de 700€ mensuales necesitarán crecer significativamente para cubrir los costes operativos.

Para evaluar la rentabilidad y determinar el punto muerto (umbral de rentabilidad), utilizaremos la fórmula del umbral de rentabilidad:

$$\text{Umbral de Rentabilidad} = \frac{\text{Costos Fijos}}{\text{Precio de Venta por Unidad} - \text{Costo Variable por Unidad}}$$

En este caso:

- Costes Fijos Mensuales: 3.800€
- Precio de Venta por Unidad (ingreso promedio por usuario):
 - Suscriptores Premium: 10€/mes.
 - Publicidad y comisiones (suponiendo que afecta a todos los usuarios):
200€/500 usuarios = 0.4€/mes por usuario.

Ingreso Promedio por Usuario (PAU):

$$\text{PAU} = 10\% \times 10\text{€} + 0.4\text{€} = 1\text{€} + 0.4\text{€} = 1.4\text{€/usuario/mes}$$

Conclusión

La viabilidad económica de TheSphere depende de su capacidad para atraer y retener usuarios, así como de la efectividad de sus modelos de ingresos. Con una estrategia de marketing efectiva y un enfoque en la mejora continua de la plataforma, es posible alcanzar la rentabilidad a mediano plazo. Las alianzas estratégicas y la oferta de servicios adicionales también pueden contribuir a mejorar la rentabilidad y asegurar el éxito a largo plazo del proyecto.

Trabajo futuro

Hay mucho trabajo por delante para llegar a ser una aplicación líder en el sector y que cubra todas las necesidades que nuestros usuarios necesitan. A continuación, listaré las principales áreas de mejora y expansión.

1. Implementación de un Sistema de Seguridad Avanzado

- Objetivo: Mejorar la seguridad y protección de datos de los usuarios.
- Acciones Específicas:
 - Cifrado de Contraseñas: Implementar algoritmos de cifrado robustos (como bcrypt) para todas las contraseñas de usuario en la base de datos.
 - Tokens de Autenticación: Introducir autenticación basada en JSON Web Tokens (JWT) para proteger las sesiones de usuario.
 - MFA (Autenticación Multifactor): Desarrollar un sistema de autenticación multifactor para mayor seguridad.

2. Mejora de la Experiencia de Usuario

- Objetivo: Hacer la interfaz más intuitiva y mejorar la experiencia del usuario.
- Acciones Específicas:
 - Interfaz de Usuario: Basada en el feedback de los usuarios, iterar sobre el diseño de la interfaz para hacerla más intuitiva y amigable.
 - Optimización de Rendimiento: Mejorar los tiempos de carga y la eficiencia de la aplicación, utilizando técnicas como el lazy loading y la optimización de imágenes.
 - Soporte Multilenguaje: Implementar soporte para múltiples idiomas para atraer a una audiencia global.

3. Expansión de Funcionalidades

- Objetivo: Añadir nuevas funcionalidades que aumenten la usabilidad y el atractivo de la plataforma.
- Acciones Específicas:
 - Mensajería y Notificaciones: Desarrollar un sistema de mensajería interna y notificaciones push para mejorar la comunicación entre usuarios.
 - Sistema de Colaboración: Implementar funcionalidades que permitan la colaboración en tiempo real, como la edición conjunta de documentos y la gestión de tareas.
 - Mercado de Servicios: Crear un marketplace donde los usuarios puedan ofrecer y contratar servicios freelance.

4. Integración con Herramientas de Terceros

- Objetivo: Facilitar la interoperabilidad con otras herramientas y plataformas populares.
- Acciones Específicas:

- API de Integración: Desarrollar APIs que permitan la integración con herramientas populares como Slack, Trello, y Asana.
- Plugins y Extensiones: Permitir a los desarrolladores externos crear plugins y extensiones para añadir funcionalidades personalizadas a la plataforma.

5. Separación de Contenedores en Instancias Diferentes y Mejora de la Arquitectura Cloud

- Objetivo: Mejorar la arquitectura en la nube para aumentar la escalabilidad, seguridad y rendimiento.
- Acciones Específicas:
 - Desplegar cada contenedor (frontend, backend, base de datos) en instancias EC2 separadas para mejorar el aislamiento y la gestión de recursos.
 - Implementar balanceadores de carga para distribuir el tráfico entre las instancias, asegurando alta disponibilidad y rendimiento óptimo.
 - Utilizar servicios de AWS como RDS para la gestión de bases de datos, ECS o EKS para la orquestación de contenedores, y S3 para el almacenamiento de archivos estáticos y backups.
 - Optimizar el uso de recursos mediante escalado automático (auto-scaling) para ajustar dinámicamente la capacidad en función de la carga de trabajo.

6. Estrategias de Marketing y Crecimiento

- Objetivo: Aumentar la base de usuarios y la visibilidad de TheSphere.
- Acciones Específicas:
 - Campañas de Marketing Digital: Iniciar campañas en redes sociales, Google Ads, y colaboraciones con influencers del sector.
 - Programa de Referencias: Crear un programa de referidos que incentive a los usuarios actuales a invitar a nuevos usuarios.
 - Eventos y Webinars: Organizar eventos en línea y webinars para mostrar las capacidades de la plataforma y atraer a más usuarios.

El plan de trabajo futuro para TheSphere está diseñado para fortalecer la seguridad, mejorar la experiencia del usuario, expandir funcionalidades, facilitar la integración con otras herramientas, y aumentar la base de usuarios a través de estrategias de marketing y crecimiento. Con un enfoque en la evaluación continua y la retroalimentación, TheSphere está bien posicionado para evolucionar y adaptarse a las necesidades cambiantes de sus usuarios.

Conclusiones

En el desarrollo de TheSphere, hemos alcanzado varios de los objetivos propuestos al inicio del proyecto. Hemos logrado crear una plataforma que facilita la colaboración entre perfiles técnicos y artísticos, proporcionando un espacio donde los usuarios pueden compartir y desarrollar sus proyectos.

La integración de la API de Cloudinary ha sido un reto conseguido, permitiendo a los usuarios subir, almacenar y gestionar imágenes de manera efectiva. Esta funcionalidad es crucial para una plataforma enfocada en compartir proyectos visuales.

Sin embargo, no todos los objetivos se han cumplido en su totalidad. A pesar de los esfuerzos, aún quedan áreas por mejorar, como la implementación de un sistema de roles más robusto y la optimización del rendimiento para manejar la carga de imágenes pesadas. Estas áreas no se completaron debido a limitaciones de tiempo y recursos durante el desarrollo del proyecto.

Este proyecto no solo es el resumen de los 2 años de duración del ciclo si no de los últimos 4 años de mi vida, he podido aplicar conceptos que aprendí en la carrera y transmitir las experiencias que viví con mis amigos durante los años en la facultad y hasta el día de hoy, estoy muy orgulloso de ello.

En conclusión, TheSphere ha sido un gran reto para mí. Con el poco tiempo que tengo, tener que trabajar y desarrollar un proyecto de esta magnitud en poco menos de 3 meses ha sido una experiencia agri dulce y dura. Aunque se hayan alcanzado muchos de los objetivos iniciales, hay espacio para seguir mejorando y expandiendo la funcionalidad y el alcance de la aplicación en el futuro.

Biblioteca de recursos web y referencias

1. Apple. (2024): Apple Design Guidelines, Apple Inc., disponible en: <https://developer.apple.com/design/>
2. Google. (2024): Material Design 3 Guidelines, Material Design, disponible en: <https://m3.material.io/>
3. Are.na. (2024): Are.na Platform, Are.na, disponible en: <https://www.are.na/>
4. The Sun. (2024): News and Updates, The Sun, disponible en: <https://www.thesun.co.uk/>
5. West, Kanye. (2016): Kanye West: The Life of Pablo, GOOD Music, disponible en: https://open.spotify.com/album/7gsWAHLeT0w7es6FofOXk1?si=k6_S3xN4T06-x3nPXt3RzA
6. Statista (2023): Number of worldwide social network users, Statista, disponible en: <https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/#:~:text=How%20many%20people%20use%20social,over%20six%20billion%20in%202028>
7. React Syntax Highlighter. (2024): React Syntax Highlighter, GitHub, disponible en: <https://github.com/react-syntax-highlighter/react-syntax-highlighter>
8. Remark. (2024): React Markdown, GitHub, disponible en: <https://github.com/remarkjs/react-markdown>
9. Hootsuite (2023): Social Media Trends 2023, Hootsuite, disponible en: <https://www.hootsuite.com/research/social-trends>
10. Bukowski, Charles. (~1972) Style, YouTube, disponible en: <https://www.youtube.com/watch?v=6HpgtW8tGfA>
11. Baeldung (2024): Dockerizing a Spring Boot Application, disponible en: <https://www.baeldung.com/dockerizing-spring-boot-application>

Anexos

1. Guía de Identidad Visual de TheSphere

La Guía de Identidad Visual de TheSphere es una extensión fundamental del apartado de diseño de la memoria. Esta guía proporciona una visión detallada de la filosofía de diseño de la aplicación y cómo esta se refleja en cada uno de los elementos visuales que conforman la experiencia de usuario.

Este documento es esencial para asegurar que todos los elementos visuales de TheSphere sean consistentes y reflejen los valores y la misión de la aplicación. La Guía de Identidad Visual no solo establece las bases para el diseño actual, sino que también proporciona directrices para futuras actualizaciones y expansiones de la aplicación.

El documento está disponible [aquí](#).