INSTITUT
PUIG CASTELLAR

Kaizen

Implementación de un sistema de reservas de cine y gestión de entradas.

Proyecto de integración y desarrollo

Mohamed Eghribel

2DAW 2024 - 2025

LICENSE

MIT License

Copyright (c) 2025 Mohamed Eghribel

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS ", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Nota: La siguiente es una traducción al español de la **licencia MIT**. Se proporciona únicamente para facilitar la comprensión. En caso de discrepancias, prevalecerá la versión original en inglés.

Copyright (c) 2025 Mohamed Eghribel

Por la presente se concede permiso, libre de cargos, a cualquier persona que obtenga una copia de este software y de los archivos de documentación asociados (el "Software"), a utilizar el Software sin restricción, incluyendo sin limitación los derechos a usar, copiar, modificar, fusionar, publicar, distribuir, sublicenciar, y/o vender copias del Software, y a permitir a las personas a las que se les proporcione el Software a hacer lo mismo, sujeto a las siguientes condiciones:

El aviso de copyright anterior y este aviso de permiso se incluirán en todas las copias o partes sustanciales del Software.

EL SOFTWARE SE PROPORCIONA "COMO ESTÁ", SIN GARANTÍA DE NINGÚN TIPO, EXPRESA O IMPLÍCITA, INCLUYENDO PERO NO LIMITADO A GARANTÍAS DE COMERCIALIZACIÓN, IDONEIDAD PARA UN PROPÓSITO PARTICULAR E INCUMPLIMIENTO. EN NINGÚN CASO LOS AUTORES O PROPIETARIOS DE LOS DERECHOS DE AUTOR SERÁN RESPONSABLES DE NINGUNA RECLAMACIÓN, DAÑOS U OTRAS RESPONSABILIDADES, YA SEA EN UNA ACCIÓN DE CONTRATO, AGRAVIO O CUALQUIER OTRO MOTIVO, DERIVADAS DE, FUERA DE O EN CONEXIÓN CON EL SOFTWARE O SU USO U OTRO TIPO DE ACCIONES EN EL SOFTWARE.

RESUMEN

El proyecto se centra en el desarrollo de un sistema de reservas y gestión de entradas de cine, basado en tecnologías de código abierto como Laravel, PostgreSQL y SvelteKit. Su objetivo es ofrecer una solución gratuita y adaptable para pequeñas y medianas empresas del sector cinematográfico.

El sistema permite consultar películas, registrarse, seleccionar butacas y generar entradas digitales. También incluye un panel administrativo básico y funcionalidades integradas como formularios de contacto y chat en línea.

Se ha seguido una metodología de tres fases: análisis de requisitos, diseño de arquitectura, desarrollo del backend y frontend, y pruebas funcionales. Utilizando herramientas de control de versiones, automatización y documentación técnica.

Como conclusión, se ha creado una base funcional y ampliable. Aunque aún existen mejoras pendientes, el sistema cumple con los objetivos principales y puede evolucionar como solución profesional o formativa.

ABSTRACT

The project focuses on the development of a cinema ticket reservation and management system, built with open-source technologies such as Laravel, PostgreSQL, and SvelteKit. Its objective is to offer a free and adaptable solution for small and medium-sized businesses in the cinema industry.

The system allows users to browse movies, register, select seats, and generate digital tickets. It also includes a basic administrative panel and integrated features such as contact forms and live chat.

A three-phase methodology was followed: requirements analysis, architectural design, backend and frontend development, and functional testing. Version control, automation, and technical documentation tools were used throughout the process.

In conclusion, a functional and expandable foundation has been created. Although some improvements are still pending, the system meets its main objectives and can evolve into a professional or educational solution.

PALABRAS CLAVE - KEYWORDS

Sistema de reservas de cine	Cinema reservation system
Gestión de entradas	Ticket management
Automatización cine	Cinema automation
Proyecto open source cine	Open source cinema project
Gestión películas y usuarios	Movies and user management

ÍNDICE

Introducción

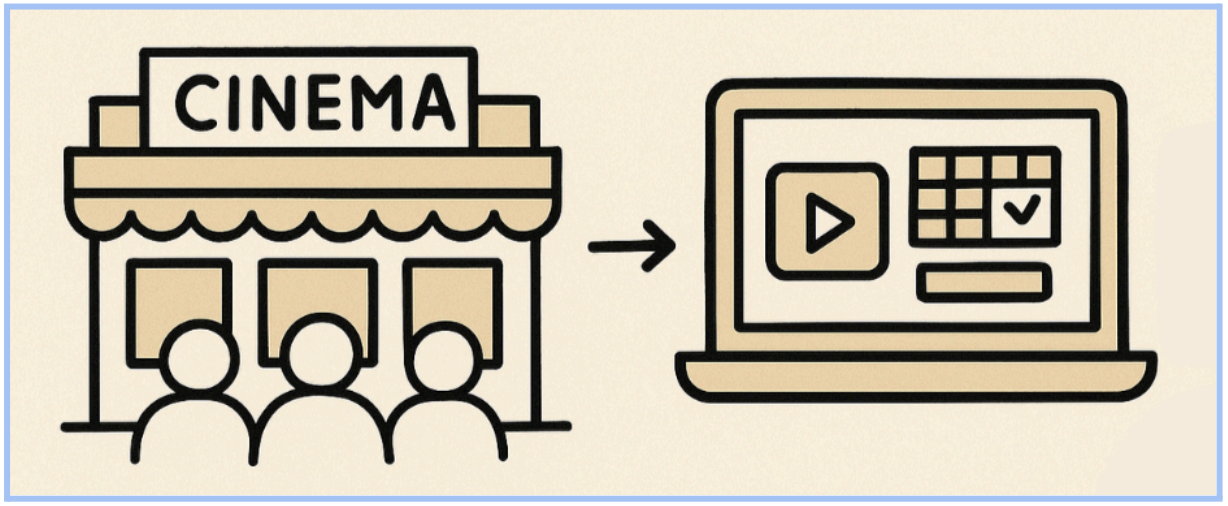
En la era digital actual, los cines continúan siendo un punto de referencia en el entretenimiento de muchas personas. La gestión eficiente y segura de estos espacios resulta crucial para garantizar una experiencia satisfactoria, tanto para clientes como para administradores.

En respuesta a la escasa oferta de este tipo de sistemas de código abierto, se presenta una plataforma destinada a automatizar las reservas de entradas, optimizar la administración de funciones y butacas, y facilitar una experiencia fluida para los usuarios finales.

Además de constituir una solución gratuita, el enfoque empresarial de este proyecto radica en el propio producto programado, complementado con un servicio de personalización del software.

- **Versión gratuita open source:** accesible para quienes deseen implementar el sistema de reservas por cuenta propia.

- **Versión premium:** incluye personalización del diseño, integración con herramientas específicas, soporte técnico y consultoría para optimizar su uso en entornos empresariales.



PARTE I. Administración del proyecto

1. Gestión del proyecto

1.1. Objetivos

1.1.1. Objetivo general

Desarrollar una solución que permita automatizar la gestión de reservas de cine y la administración de entradas, incluyendo el registro de usuarios y la visualización de películas disponibles.

1.1.2. Objetivo específico

- Crear una base de datos sencilla (que permita almacenar y consultar información sobre películas).
- Desarrollar una API básica para gestionar el acceso a los datos.
- Implementar una interfaz web donde se pueda registrar un usuario, iniciar sesión y hacer reservas.
- Incluir un panel sencillo para añadir y editar películas, así como ver datos básicos de uso.
- Probar las funciones principales del sistema para comprobar que todo responde correctamente.
- Automatizar tareas simples como el despliegue o la documentación usando GitHub Actions.
- Establecer las bases de una arquitectura desacoplada (frontend y backend independientes), permitiendo futuras ampliaciones o integraciones externas.
- Transmitir claramente el funcionamiento del sistema, mostrando que, aunque su interfaz parezca simple, el proyecto integra múltiples capas técnicas complejas.

1.2. Entorno del proyecto

1.2.1. Contexto

La existencia de este proyecto es una demanda de sistemas eficientes para gestionar entornos relacionados con el entretenimiento (en este caso, las salas de cine). Estos entornos enfrentan diferentes retos importantes (gestión correcta de asientos, visualización de horarios de películas, necesidad de supervisión por parte del personal administrativo, etc.).

Las soluciones existentes en el mercado suelen ser costosas, cosa que hace que sean poco accesibles para empresas emergentes o pequeños negocios.

Se carece de buenas opciones en el ámbito open source. Esta falta de herramientas ha sido uno de los factores clave que han motivado el desarrollo del presente proyecto.

El sistema desarrollado desea ofrecer una experiencia sencilla para los usuarios al reservar sus entradas y mantener una solución lo suficientemente cómoda para gestionar datos. A medida que avance el desarrollo, se irán mejorando diversos aspectos, todo ello basado en un enfoque de código abierto.

1.2.2. Justificación

El mundo web ha transformado la vida cotidiana del mundo pero los cines se mantienen como un punto clave en la industria del entretenimiento, siendo un pilar fundamental para muchas personas.
Lamentablemente, muchos de los sistemas de gestión de reservas de entradas están lejos de cubrir las necesidades de las pequeñas y medianas empresas por el costo de estas.
Con este proyecto se pretende ofrecer una plataforma de código abierto con un diseño sencillo, con el objetivo de conseguir una experiencia del cliente sencilla y la eficiencia de la administración, reduciendo así la barrera económica en el sector.

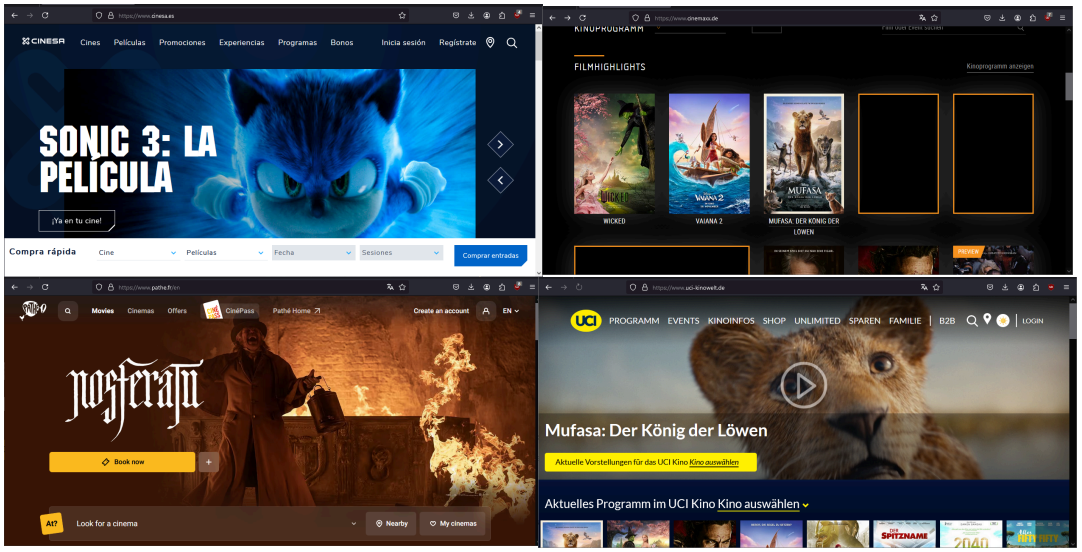
1.3. Soluciones existentes

Es sencillo encontrar plataformas comerciales que facilitan la administración de la venta de boletos para cines. Este punto representa una pequeña comparación entre algunas de las soluciones más destacadas del mercado, junto con algunas opciones de código abierto. La finalidad de esto es reconocer funcionalidades compartidas y evaluar la posición de este proyecto en relación a estas.

Productos en el mercado

Durante la fase de análisis, se tomaron de referencia varias plataformas activas en distintos países (España, Francia y Alemania):

Cinesa (España)	Plataforma con un sistema intuitivo y completo. Ha sido la principal referencia en este proyecto.
Pathé (Francia)	Sistema similar en estructura a Cinesa, pero con un enfoque más visual y elementos promocionales destacados.
Cinemaxx (Alemania)	Clara y funcional, fidelización y promociones en portada.
UCI Kinowelt (Alemania)	Diseño bastante diferente, pero eficaz y cómodo en cuanto a la selección de películas y visualización de horarios.



Funcionalidades comunes detectadas

A pesar de las diferencias visuales y culturales entre estos sistemas, es posible identificar un conjunto de elementos recurrentes que conforman el estándar de la industria en cuanto a plataformas de reserva de entradas de cine. No obstante, que una funcionalidad sea habitual no implica necesariamente que esté presente en todos los casos.

La siguiente tabla resume las características más comunes observadas:

Categoría	Funcionalidad común detectada
Información de película	Imagen/miniatura, título, sinopsis, duración, clasificación por edad
Contenido multimedia	Tráiler integrado (opcional)
Clasificación	Género de la película (acción, drama, comedia…)
Actores	Lista de actores y, en algunos casos, enlaces a sus perfiles
Valoración del público	Likes, puntuaciones o sistemas de valoración visual
Selección de cine	El usuario elige su ciudad y sala de preferencia
Selección de butaca	Sistema visual para seleccionar asientos específicos
Sistema de usuario	Registro, login y acceso a entradas compradas
Clasificación por edad	Cada película muestra la edad recomendada para su visionado.

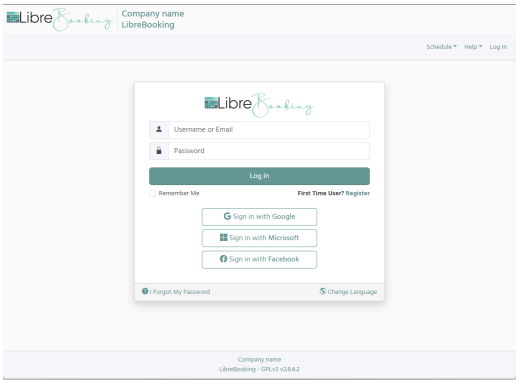
Estos elementos definen la experiencia base esperada por los usuarios, y han sido tomados como referencia para el desarrollo de este sistema.

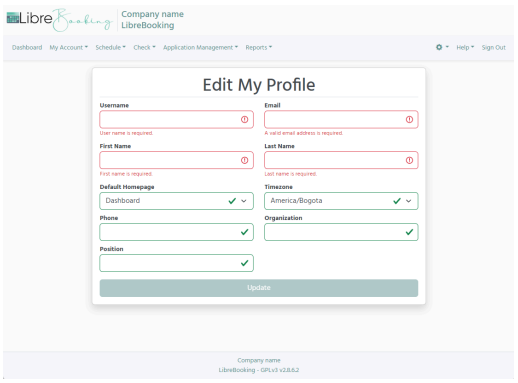
Alternativas de código abierto

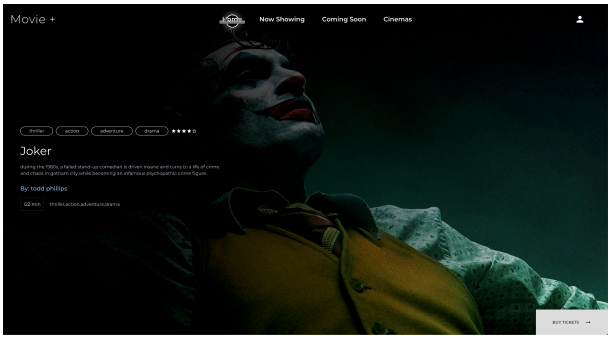
Durante la investigación se encontraron dos proyectos de código abierto que, aunque no cumplen completamente con los objetivos de este sistema, ofrecen enfoques interesantes.

LibreBooking: Plataforma pensada para la reserva de espacios, como salas o recursos compartidos. Está desarrollada en PHP y utiliza MySQL y Bootstrap.
No está orientada a cines, pero su estructura modular y personalizable puede servir de referencia para la gestión de usuarios y reservas. Sin embargo, no incluye funciones clave como la selección de butacas o la gestión de películas.

CinemaPlus: Más específico para cines, desarrollado con el stack MERN (MongoDB, Express, React y Node.js). Ofrece funcionalidades como administración de películas y reservas, generación de códigos QR y exportación en PDF. A pesar de su enfoque más cercano, el proyecto está desactualizado y no cuenta con una comunidad activa de desarrollo.







Los repositorios del proyecto y las plataformas consultadas están recogidos en la sección "Enlaces de referencia" de los Anexos.

2. Presupuesto

2.1. Elementos de la estimación inicial

2.1.1. Desarrollo del sistema

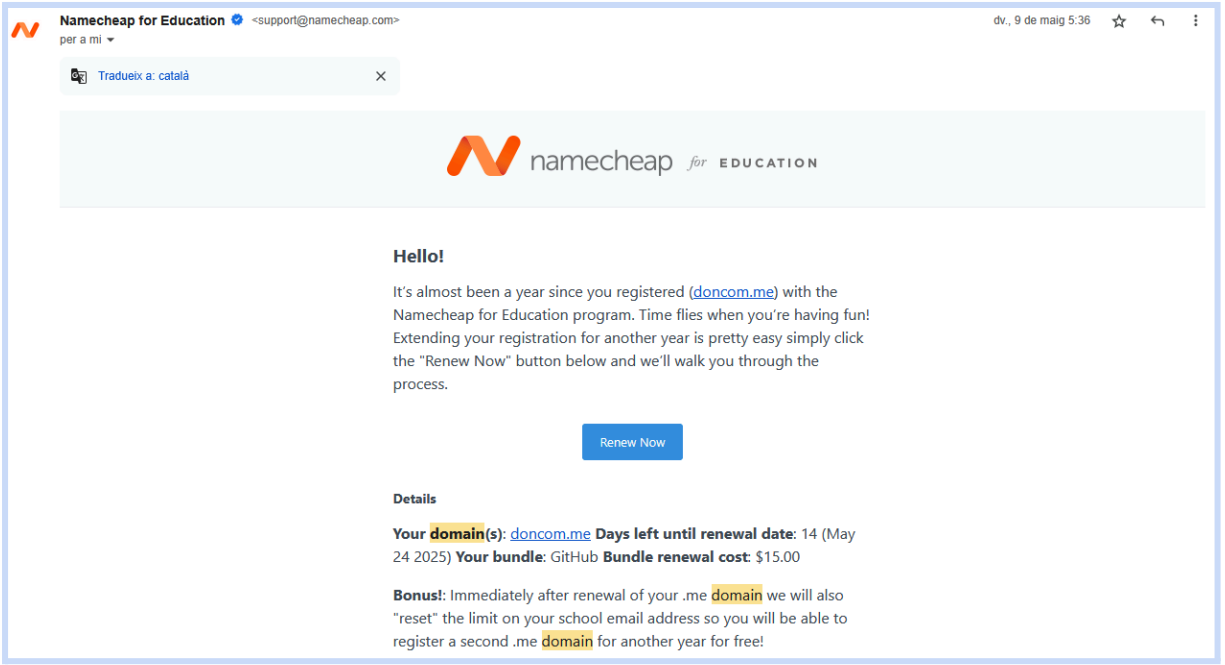
El tiempo de desarrollo estimado ha sido de aproximadamente 7 meses, considerando que el proyecto ha sido realizado por un desarrollador principiante con conocimientos previos básicos en las tecnologías empleadas. Abarcando todas las fases del proyecto: análisis, diseño, desarrollo del backend y frontend, integración de funcionalidades clave, documentación técnica y despliegue.

Al estar basado en tecnologías open source, el desarrollo no ha supuesto ningún coste por licencias, lo cual ha permitido mantener un enfoque accesible y económicamente viable.

2.1.2. Infraestructura

Para presentar el sistema de forma relativamente accesible desde casi cualquier dispositivo, se ha utilizado un dominio personalizado previamente adquirido, que forma parte de un almacén personal de proyectos. Aunque inicialmente se compró para usos generales, con el tiempo ha ido acumulando distintas soluciones personales y académicas. Este proyecto ha sido la pieza final que ha justificado su renovación y consolidación como soporte principal.

El dominio ha sido renovado recientemente hasta 2026, con un coste de 15 € anuales. Aunque el acceso proporcionado actualmente es solo al frontend del sistema, esta configuración ha sido suficiente para mostrar el diseño visual y la interfaz del proyecto en un entorno relativamente funcional y profesional.



2.1.3. Mantenimiento

Aunque el sistema no requiere un mantenimiento técnico intensivo, se contempla una pequeña partida para cubrir tareas menores, como la actualización de dependencias, ajustes visuales, correcciones puntuales o pequeñas mejoras funcionales tras su despliegue. Este margen permite adaptar el sistema a cambios futuros en los navegadores o librerías utilizadas, garantizando que siga siendo accesible y funcional con el paso del tiempo. Se estima un coste simbólico de 10 € para este tipo de intervenciones.

2.1.4. Contingencias

A lo largo de cualquier desarrollo pueden surgir necesidades inesperadas: desde la compra de una licencia puntual, hasta la contratación de un servicio externo o la reestructuración de parte del diseño. Por ello, se ha reservado una pequeña cantidad como colchón económico para este tipo de eventualidades.
Este fondo de contingencia, estimado en 10% del total, asegura cierta flexibilidad y capacidad de respuesta ante imprevistos sin comprometer la continuidad ni la estabilidad del proyecto.

2.1.5. Calculos

Si este mismo proyecto se realizará para un cliente y no como trabajo personal o académico, el coste real sería mucho más alto, principalmente por la mano de obra.

CONCEPTO	DESCRIPCIÓN	COSTE ESTIMADO
Dominio web	Renovación anual del dominio utilizado como almacén de proyectos personales	15€
Mantenimiento	Pequeñas actualizaciones, correcciones y mejoras tras el despliegue	50€
Contingencias	Fondo de reserva para imprevistos (licencias, rediseños, servicios puntuales)	TOTAL * 0.1€
Tecnologías	Herramientas y frameworks open source sin coste asociado	0€
Desarrollo	Trabajo personal (sin coste económico directo, estimado en 7 meses de dedicación)	1100 x 7 = 7700€
TOTAL		7765€ + 776,5€ = 8541,5 €

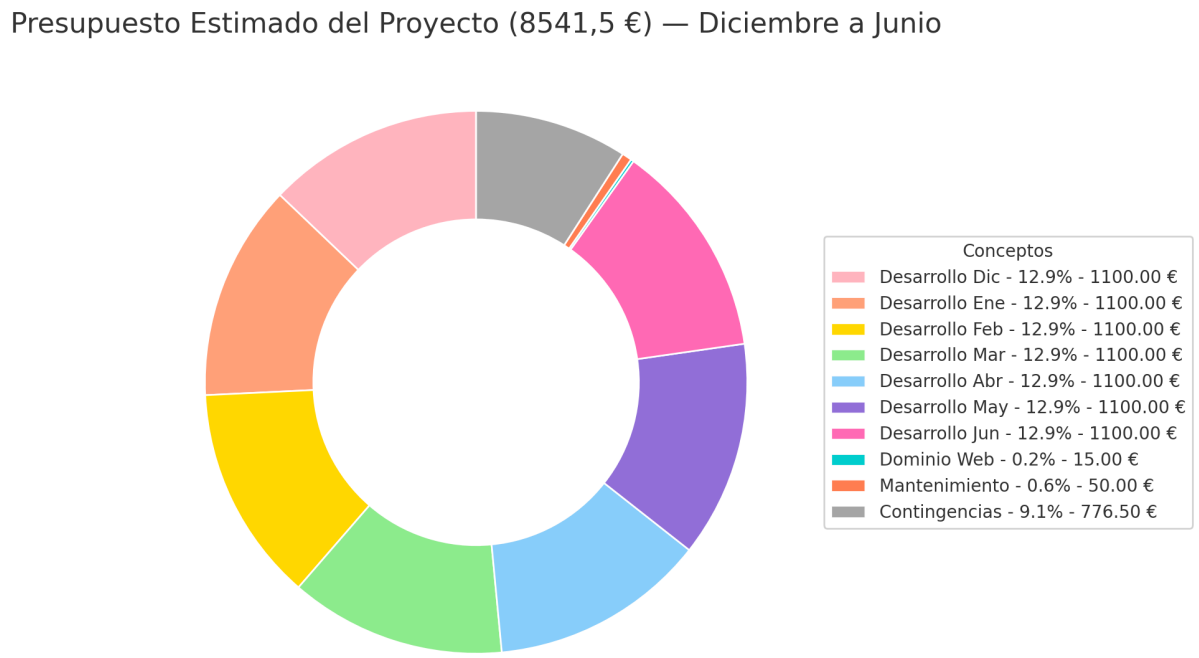
2.2. Análisis económico y justificación del presupuesto

El presupuesto del proyecto se ha planteado con un enfoque realista, orientado a optimizar recursos sin comprometer la calidad del desarrollo. Como se ha explicado anteriormente, el uso de tecnologías open source ha eliminado los costes por licencias, lo que ha permitido centrar el presupuesto en aspectos clave como el dominio web, el mantenimiento básico y un fondo de contingencia para imprevistos.

La valoración simbólica del trabajo de desarrollo, estimado en 7 meses a razón de 1100 € mensuales, sirve como referencia para dimensionar el esfuerzo involucrado, aunque no haya supuesto un gasto directo en este caso. De tratarse de un encargo profesional, este sería el componente más significativo del coste total, como mínimo.

El control del presupuesto se prevé mediante una revisión periódica del estado del sistema, el seguimiento de posibles necesidades de mantenimiento y la gestión responsable del fondo de contingencia. Esta previsión asegura que el proyecto pueda mantenerse operativo y actualizado en el tiempo, con un coste reducido y predecible.

En conjunto, el proyecto resulta económicamente viable, tanto en un contexto académico como profesional, demostrando una correcta planificación y gestión de recursos.



PARTE II. Ejecución del proyecto

1. Análisis

1.1. Especificación de requisitos

1.1.1. Funcionales

Gestión de usuarios

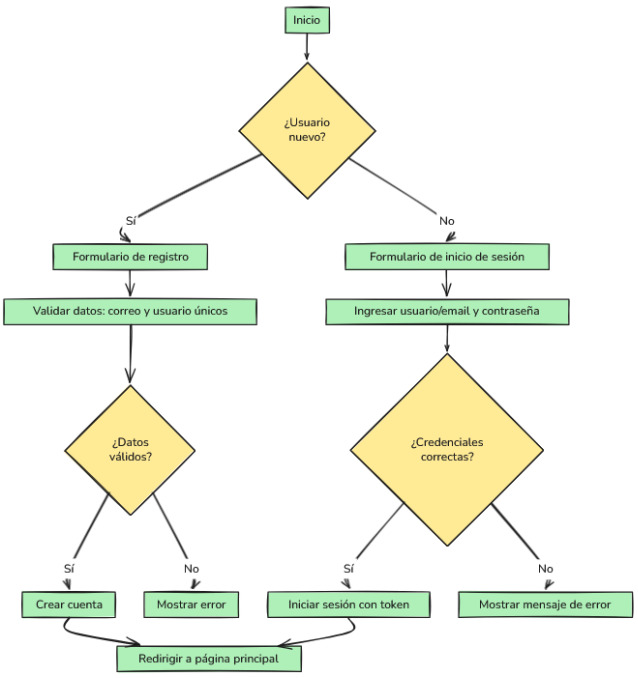
El sistema debe permitir el registro de usuarios proporcionando ciertos datos: nombre completo, nombre de usuario, correo electrónico y contraseña. El nombre de usuario será elegido por el propio usuario, y la contraseña deberá cumplir con requisitos mínimos de seguridad. Estos criterios pueden variar, pero inicialmente se plantea un mínimo de 8 caracteres. En el futuro, podrían exigirse condiciones más estrictas, como incluir al menos una letra mayúscula, una minúscula, un número, un carácter especial y alcanzar un mínimo de 10 caracteres.

Durante el proceso de registro, se validarán los datos introducidos para evitar duplicados. No se permitirá la creación de cuentas con un correo electrónico o nombre de usuario ya existentes en el sistema.

Para iniciar sesión, se podrá utilizar el nombre de usuario o el correo electrónico, junto con la contraseña correspondiente. Si las credenciales son correctas, el sistema redirigirá a la página principal con la sesión iniciada.

En caso de error, se mostrará un mensaje simple indicando que las credenciales son incorrectas.

La gestión de sesiones se realizará mediante el uso de tokens para mantener la seguridad.



Gestión de películas

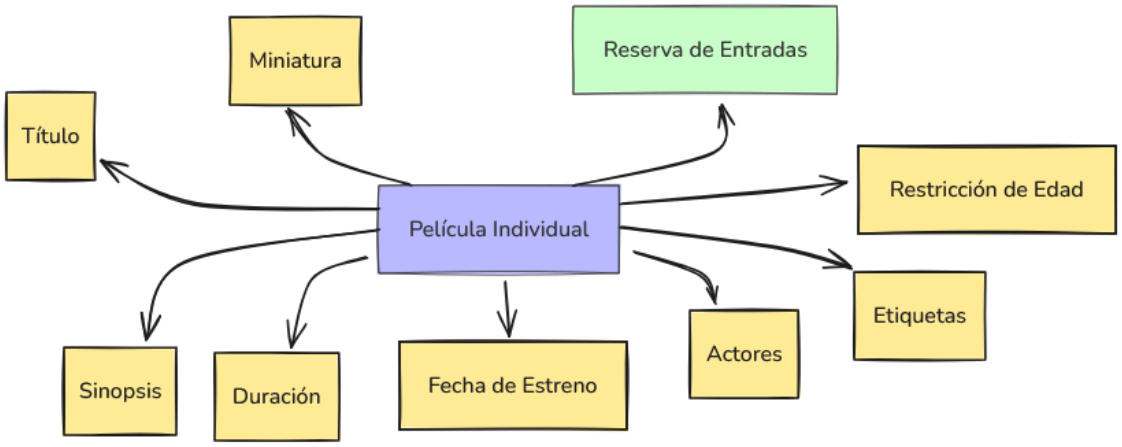
En el sitio se mostrará un listado general de películas, accesible desde una sección específica. Este espacio permitirá a los usuarios visualizar el conjunto de títulos disponibles de forma clara y ordenada.

Se prevé también la inclusión de un apartado destacado que agrupe las películas más relevantes o populares, facilitando el acceso a los contenidos más demandados.

Cada película contará con una página informativa individual, en la que se incluirán detalles como la miniatura, el título, la sinopsis, la duración, la fecha de estreno, la lista de actores y las etiquetas asociadas. Además, se indicará si existen restricciones de edad asociadas a la visualización del contenido.

Desde esta misma página, será posible iniciar el proceso de reserva de entradas, lo que permitirá una transición directa entre la consulta de la información y la acción por parte del usuario.

En el mejor de los casos, habría toda esta información:



Gestión de entradas

El sistema contempla la posibilidad de que los usuarios seleccionen butacas concretas al realizar una reserva. Esta selección podrá hacerse de dos formas: mediante un sistema interactivo (clic sobre la butaca) o introduciendo manualmente la fila y el número deseado.

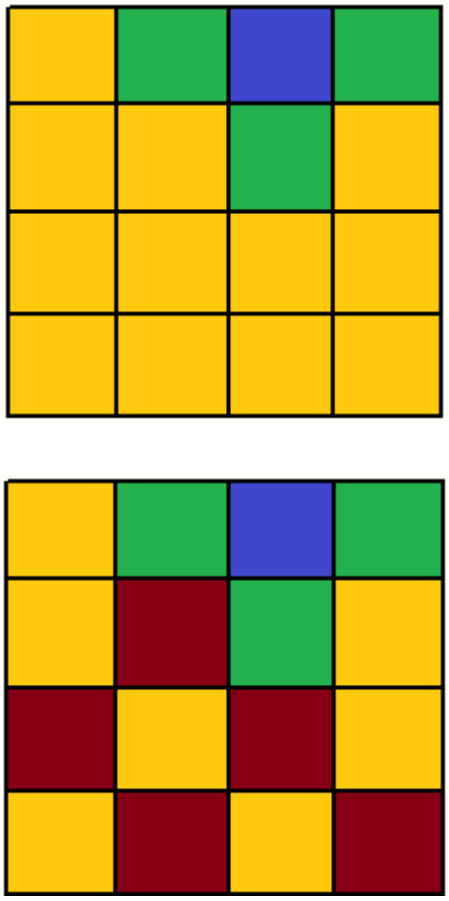
Una vez completado el proceso de compra, se generará una entrada en formato digital que podrá ser descargada por el usuario. Esta entrada estará disponible en una sección específica del sitio web mientras la proyección no haya finalizado, permitiendo la recuperación en caso de pérdida.

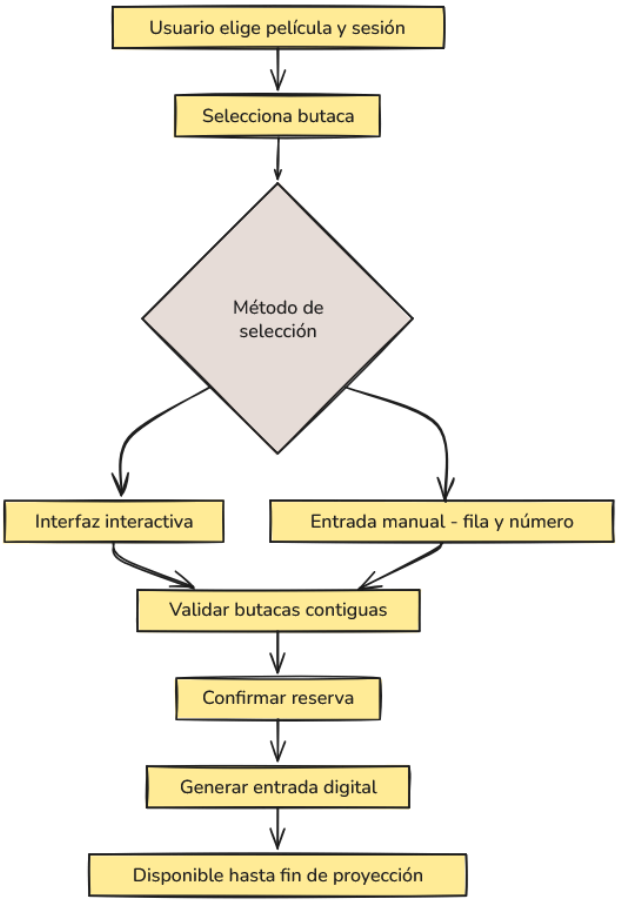
Para el diseño y comportamiento general de esta funcionalidad se ha tomado como referencia el panel de reservas de Cinesa.

Cinesa contiene diferentes niveles de butacas. Se puede ver en el anexo “Cinesa Butacas”.

Selección inteligente de butacas

El sistema podrá incorporar una lógica adicional en la selección de butacas con el fin de evitar elecciones que dejen asientos aislados o desconectados del resto. La idea es que solo puedan seleccionarse butacas que estén contiguas o conectadas a la zona de asientos ya elegida por el usuario, garantizando así una ocupación coherente de la sala. En la imagen, si la butaca seleccionada inicialmente es la azul, únicamente deberían poder seleccionarse las verdes, ya que están directamente conectadas con ella. Por el contrario, las butacas rojas quedarían deshabilitadas para la selección al no estar conectadas en forma inmediata. Esta estrategia evita configuraciones incoherentes o problemáticas desde el punto de vista de la distribución de asientos.





Este enfoque está inspirado en las directrices y prácticas empleadas por plataformas reales como Cinesa. Para más información, se puede consultar la documentación oficial en el Anexo.

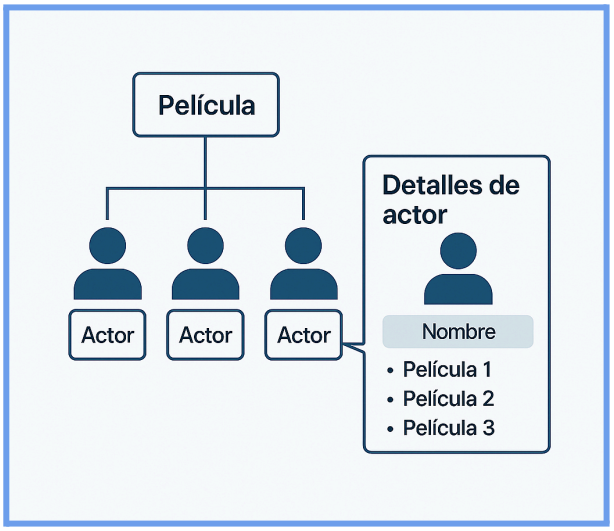
Gestión de actores

Cada película registrada en el sistema está asociada a un conjunto de actores que participan en ella. Estos actores representan un elemento clave dentro de la ficha informativa de cada título, ya que permiten al usuario identificar fácilmente el reparto que forma parte de la producción.

En el diseño del sistema, se contempla la posibilidad de integrar un módulo específico donde se muestre información detallada de los actores, incluyendo su nombre, y en caso de estar implementado, una lista de películas en las que han participado dentro del catálogo disponible.

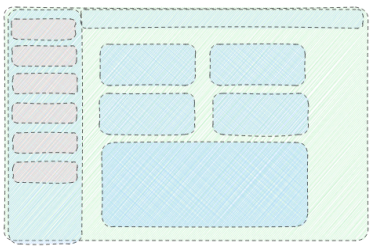
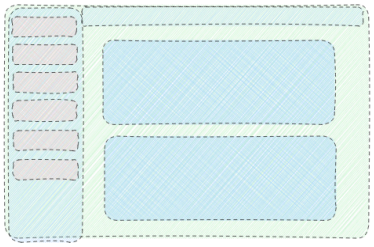
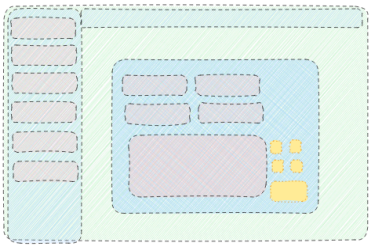
Esta funcionalidad permitiría al usuario explorar las trayectorias de los actores, navegando de forma cruzada entre películas y perfiles individuales.

Si el desarrollo del sistema lo permite dentro del tiempo estimado, esta funcionalidad podrá completarse e integrarse plenamente. En caso contrario, se deja contemplada como una mejora futura que puede ser incorporada sin afectar la estructura ya existente del sistema.



Gestión administrativa

En el apartado administrativo, los responsables del sistema podrán añadir, modificar o eliminar películas, así como subir imágenes asociadas a estas.
También tendrán acceso a ventas y gestión de usuarios.

1.1.2. No funcionales

Rendimiento	El sistema debe ofrecer tiempos de respuesta adecuados para garantizar una experiencia de usuario fluida, especialmente durante operaciones habituales como el inicio de sesión, la visualización de películas o la reserva de entradas.
Seguridad	Las contraseñas deberán ser almacenadas utilizando técnicas de cifrado seguro, con el objetivo de proteger los datos sensibles de los usuarios. Asimismo, se plantea gestionar las sesiones mediante tokens, a fin de garantizar un acceso seguro y controlado durante la interacción con el sistema.
Mantenibilidad	Se prioriza que el código sea claro y esté razonablemente documentado con el fin de facilitar futuras ampliaciones o la incorporación de colaboradores. Para ello, se contempla el uso de herramientas como Doxygen, cuya aplicación se detalla en apartados posteriores. Además, se plantea incorporar un sistema básico de registros que permita identificar errores o eventos importantes durante la ejecución del sistema.
Recuperación ante desastres	Se valora la posibilidad de implementar un sistema básico de copias de seguridad periódicas para la base de datos, como medida preventiva ante posibles fallos o pérdidas de información. Esta funcionalidad se aplicará si el tiempo y los recursos lo permiten, o se considerará una mejora futura.
Auditoria	Se plantea registrar parte de las acciones clave realizadas por los usuarios (inicios de sesión, modificación de datos). En caso de implementarse estos registros se podrían preservar durante un periodo definido y exportarse en formatos simples como CSV o JSON. Tanto la creación de estos registros como el alcance y la duración dependerá del tiempo disponible y como avance el proyecto.
Integración	Aunque a este punto no se ha definido una estrategia de integración compleja con otras herramientas externas, se deja abierta la posibilidad de conectar servicios adicionales, como pasarelas de pago, sistemas de notificación o herramientas de análisis.

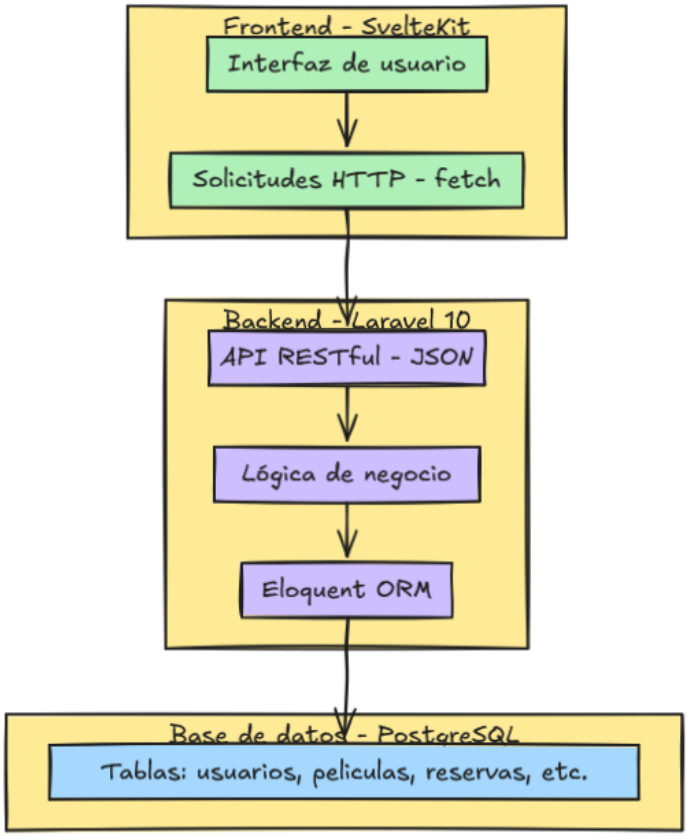
2. Diseño

2.1. Arquitectura

2.1.1. Descripción general

El sistema sigue una arquitectura Cliente-Servidor desacoplada, donde el cliente está desarrollado con SvelteKit y el servidor con Laravel. Esta separación permite una evolución independiente de cada parte del sistema, facilitando el mantenimiento y el añadir futuras integraciones con otros servicios.

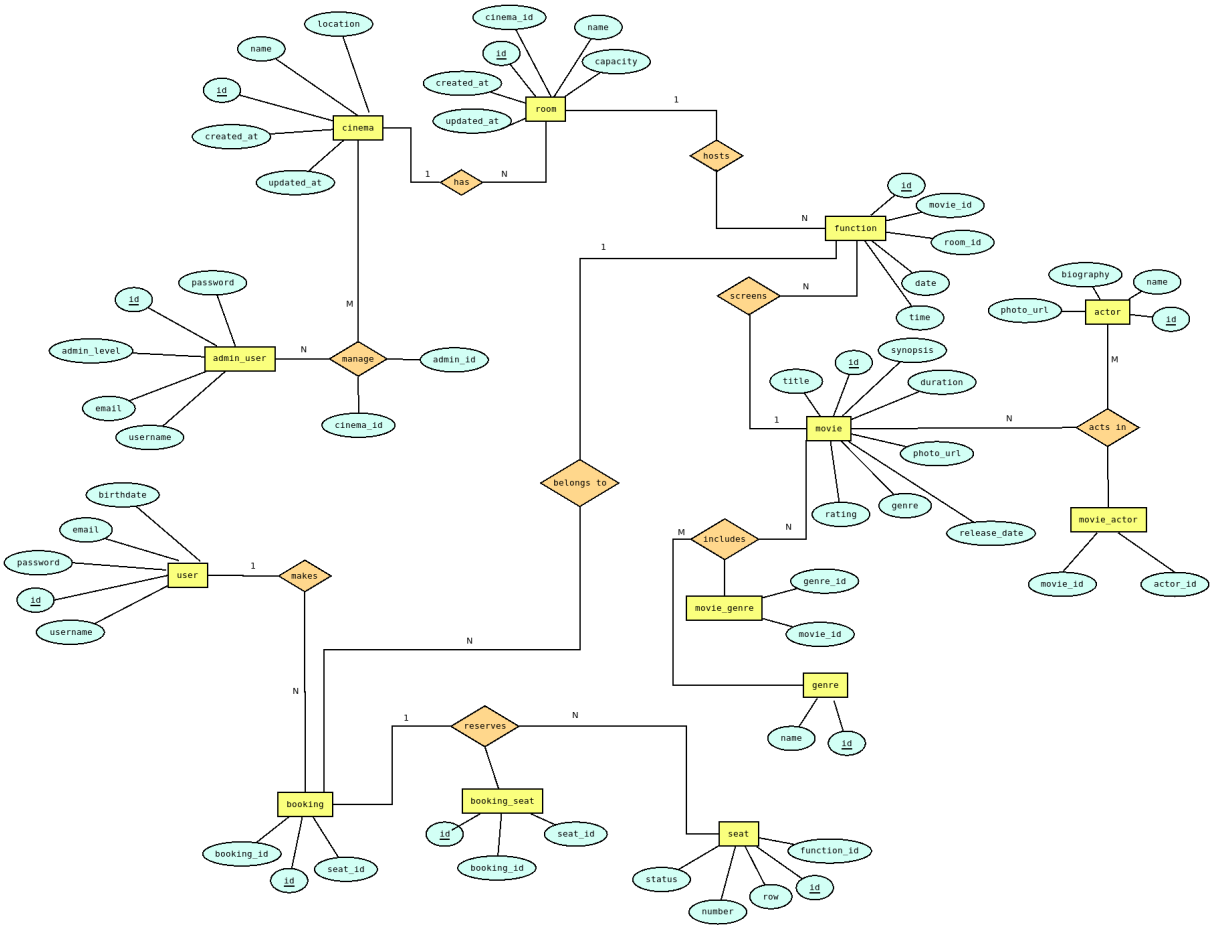
El cliente se encarga de gestionar la interfaz de usuario y realiza solicitudes HTTP al backend mediante una API RESTful, intercambiando datos en formato JSON. El backend implementa la lógica de negocio, valida las operaciones y se comunica con una base de datos relacional (PostgreSQL) a través del ORM Eloquent.



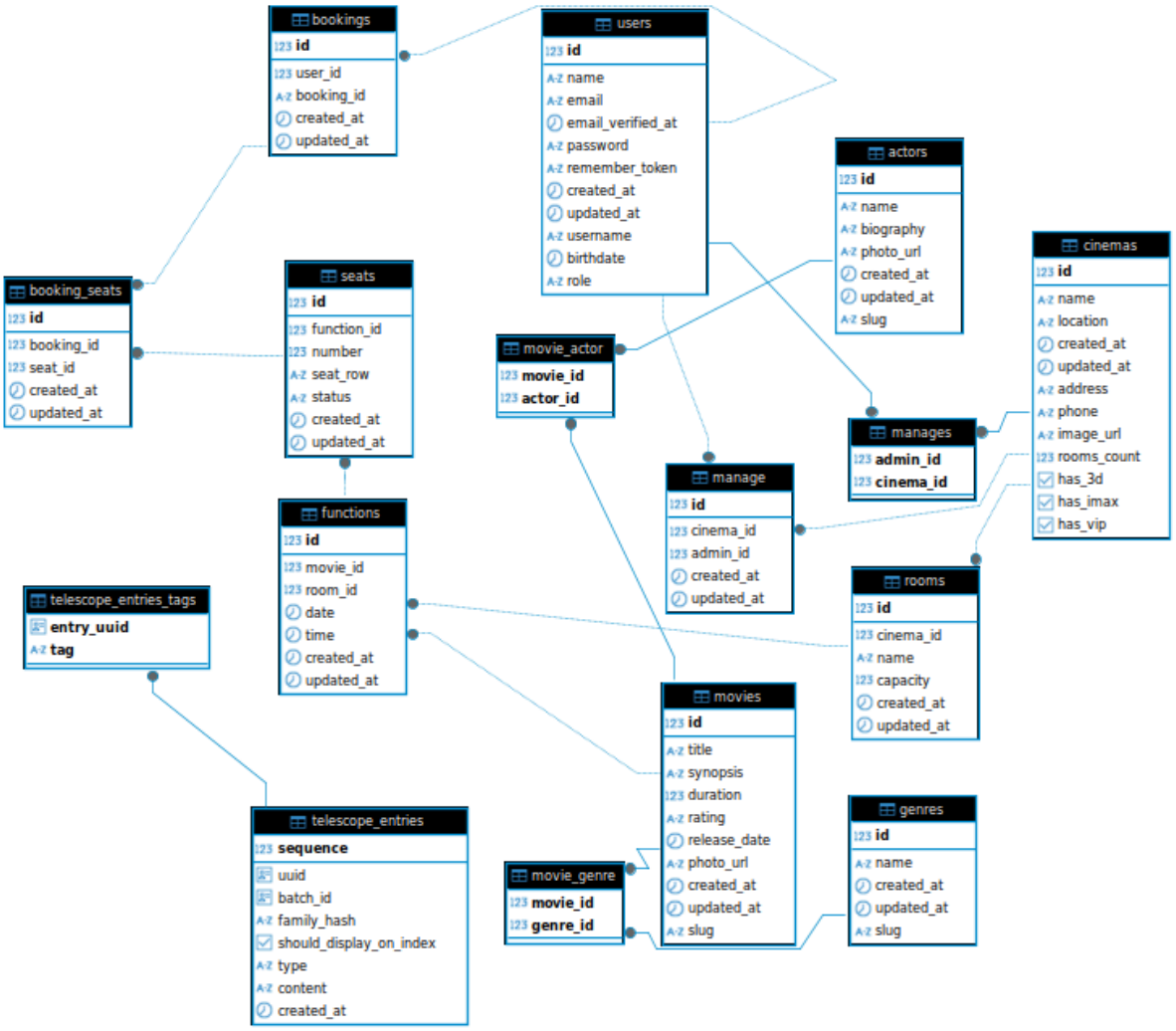
Cada funcionalidad (como la autenticación, el listado de películas o la gestión de reservas) está expuesta como un conjunto de endpoints que cumplen con las convenciones REST (GET, POST, PUT, DELETE), permitiendo así un acceso claro y estructurado a los recursos del sistema.

2.1.2. Diagramas

Diagrama ER (Esquema Entidad-Relación)



Esquema de base de datos



2.1.2.1. Casos de uso

Para una simplificación, se ha decidido separar los usuarios corrientes de los administradores, eso quiere decir que el administrador no hereda las propiedades de usuario. Por la forma en la que se ha planeado la base de datos por ahora se tomará de forma separada cada rol.

- Registrarse en el sistema
- Iniciar sesión
- Buscar películas
- Filtrar películas
- Visualizar detalles de una película
- Reservar butacas
- Descargar entradas
- Ver lista de reservas

2.1.2.2. Diagrama de estados

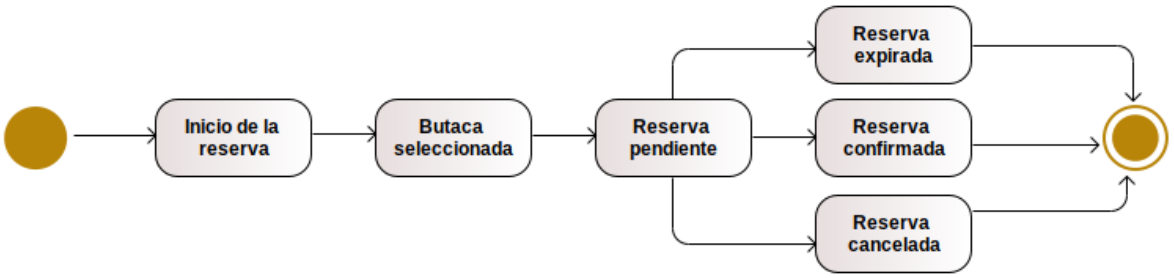
PROCESO DE COMPRA DE BUTACA:

ACCIÓN	DESCRIPCIÓN
Inicial (Inicio de la reserva)	El usuario selecciona una película, horario y butacas disponibles.
Butaca seleccionada	El usuario elige las butacas específicas. Se valida la disponibilidad de las butacas seleccionadas.
Reserva pendiente	El sistema solicita al usuario que confirme el pago. Opciones disponibles: Confirmar o Cancelar.
Reserva confirmada	La reserva se registra como válida. Se genera la entrada en formato PDF para descargar.
Reserva cancelada	Si el usuario cancela antes de confirmar, se finaliza el proceso.
Reserva expirada	Si el usuario no realiza el pago en un tiempo límite, el sistema cancela automáticamente la reserva y libera las butacas.

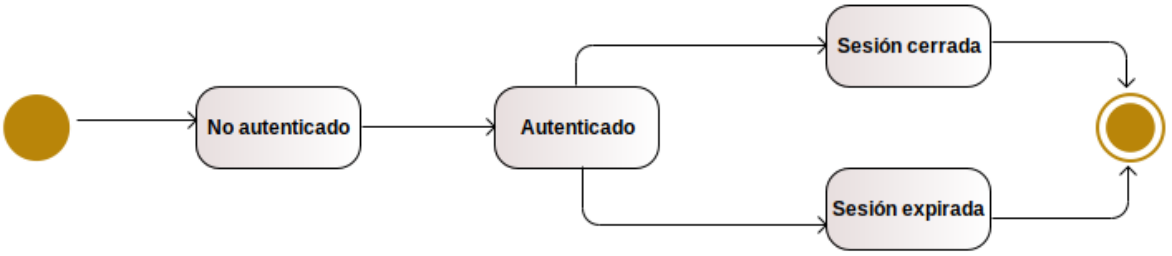
AUTENTICACIÓN DE USUARIO:

ACCIÓN	DESCRIPCIÓN
No autenticado	El usuario no ha iniciado sesión.
Autenticado	El usuario ingresa sus credenciales y accede al sistema.
Sesión cerrada	El usuario cierra su sesión manualmente.
Sesión expirada	Si el usuario permanece inactivo durante un tiempo, el sistema cierra la sesión automáticamente
De No autenticado a Autenticado: Cuando el usuario proporciona credenciales válidas. De Autenticado a Sesión cerrada: Cuando el usuario cierra la sesión manualmente. De Autenticado a Sesión expirada: Cuando el tiempo de inactividad excede el límite permitido.	

PROCESO DE COMPRA DE BUTACA:



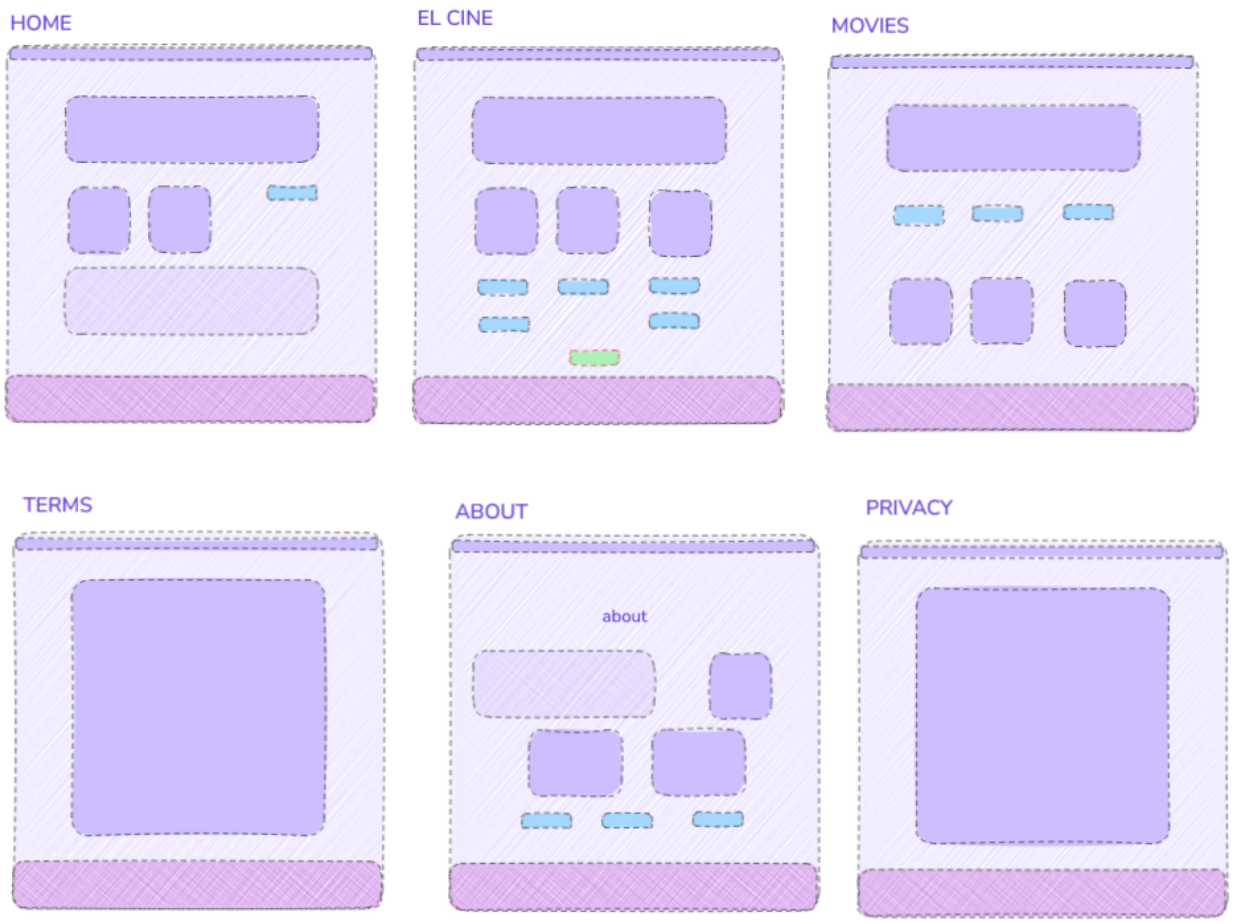
AUTENTICACIÓN DE USUARIO:



2.2. Interfaz

Para el desarrollo de la interfaz de usuario, se trabajó inicialmente con estructuras base y esquemas visuales que definieron la organización general de cada vista del sitio. Estas maquetas preliminares, simuladas con componentes simplificados, funcionaron como referencia para establecer la disposición de elementos clave como menús, botones, bloques de contenido y formularios. Se priorizó la claridad y detección de componentes comunes.

El proceso se abordó desde una lógica de diseño estructural, donde se representaron los distintos módulos de forma esquemática antes de avanzar con los estilos finales. Esto permitió tener una visión global del sitio y garantizar la coherencia entre páginas como Home, Cines, Movies, About, Privacy, entre otras.



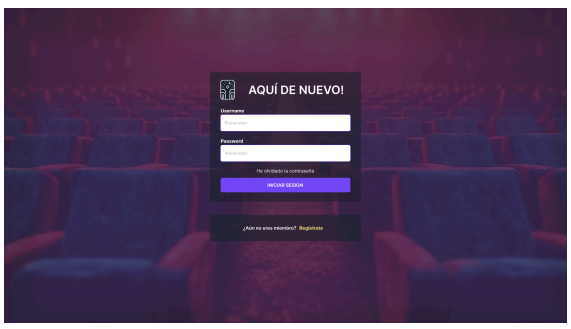
Inicio de sesión, Registro y Restablecer Contraseña

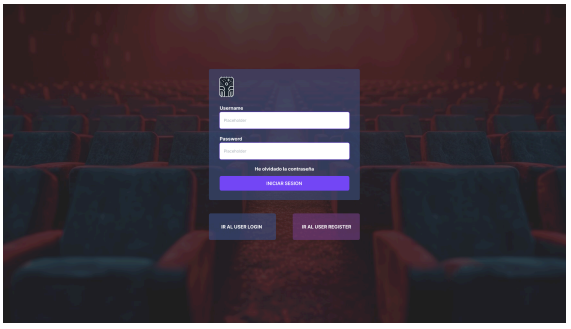
En las fases iniciales del desarrollo se contempló el diseñar las pantallas de autenticación (inicio de sesión, registro y restablecimiento de contraseña) utilizando herramientas de prototipado como Figma, con el objetivo de establecer una guía visual clara y coherente que luego pudiera trasladarse a código.

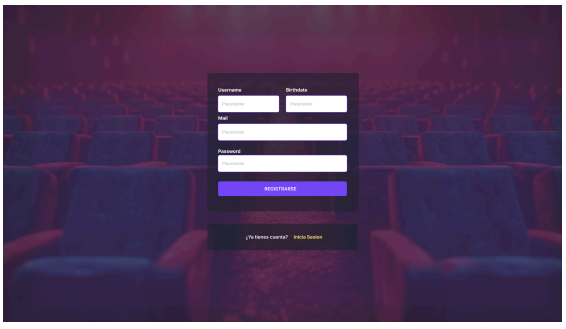
Sin embargo, dado que se trataba de un proyecto individual y que en ese momento el alcance funcional aún no estaba completamente definido, se optó por priorizar la implementación técnica de estos flujos por encima del diseño detallado. El desarrollo visual exhaustivo implicaba una inversión considerable de tiempo y esfuerzo, especialmente en tareas como prototipado, validación iterativa y fidelidad visual entre pantallas, lo cual no era viable en paralelo con la definición estructural y lógica del sistema.

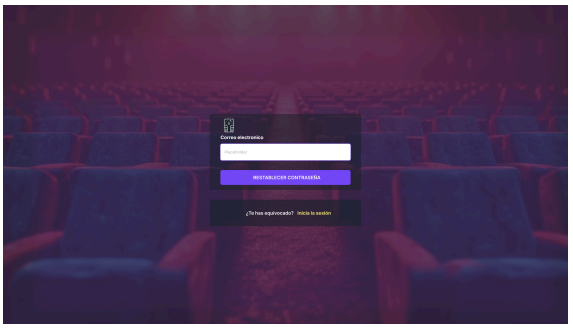
Por ello, las pantallas de autenticación se implementaron en una versión funcional mínima (beta), enfocada en garantizar la operatividad del sistema de login, el registro de nuevos usuarios y el flujo de recuperación de contraseña.

Aunque no cuentan con un diseño final estilizado, estas pantallas cumplen su propósito técnico y ofrecen una base sólida para futuras iteraciones centradas en usabilidad y estética.









2.3. Tecnología

2.3.1. Backend

2.3.1.1. Lenguaje de programación

Para el desarrollo del backend se ha seleccionado PHP, un lenguaje consolidado en el entorno del desarrollo web, que permite la creación de aplicaciones dinámicas del lado del servidor.

Una de las razones para su elección es la familiaridad previa con el lenguaje, lo que ha permitido agilizar el desarrollo del proyecto. PHP ofrece soporte tanto para la programación estructurada como orientada a objetos, facilitando una organización del código clara y mantenible.

2.3.1.2. Framework

Posiblemente la parte más importante, ya que facilitará la lógica de negocio y es la base de lo que se nombró en el apartado Descripción general. Mantiene la arquitectura **Cliente-Servidor**, lo que permite que el cliente se comunique con el servidor para obtener o enviar datos sin que estos se mezclen con la lógica interna del servidor.

2.3.2. Frontend

El frontend se desarrolló principalmente con SvelteKit, elegido por su rendimiento y simplicidad frente a opciones como Vue o React. Se utilizó Bootstrap para el diseño responsive, y Blade solo en partes puntuales del backend, aprovechando su integración con Laravel.

2.3.2.1. UI Framework

Para facilitar el diseño responsive y garantizar una experiencia de usuario fluida en distintos dispositivos, se ha optado por utilizar Bootstrap 5.x como framework de interfaz.

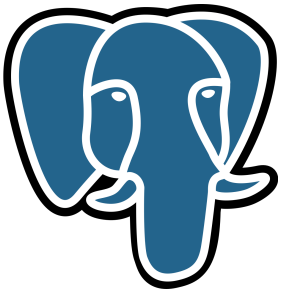
Esta elección se basa en la experiencia previa con otros proyectos, lo que permite retomarlo fácilmente. Este ofrece una amplia gama de componentes predefinidos (botones, barras de navegación, forms, etc.), reduciendo el tiempo y asegurando coherencia visual sin necesidad de diseñar desde cero.

2.3.2.2. Iconografía

Para mejorar la experiencia visual del usuario, se ha utilizado Bootstrap Icons, una librería oficial mantenida por el equipo de Bootstrap. Ofrece una amplia variedad de iconos vectoriales integrables de forma sencilla en componentes como botones, enlaces o encabezados.

2.3.3. Base de datos

Gestor de Base de Datos



Para la gestión de datos se ha elegido PostgreSQL, un sistema de gestión de bases de datos relacional avanzado, reconocido por su robustez, rendimiento y adherencia a los estándares SQL.

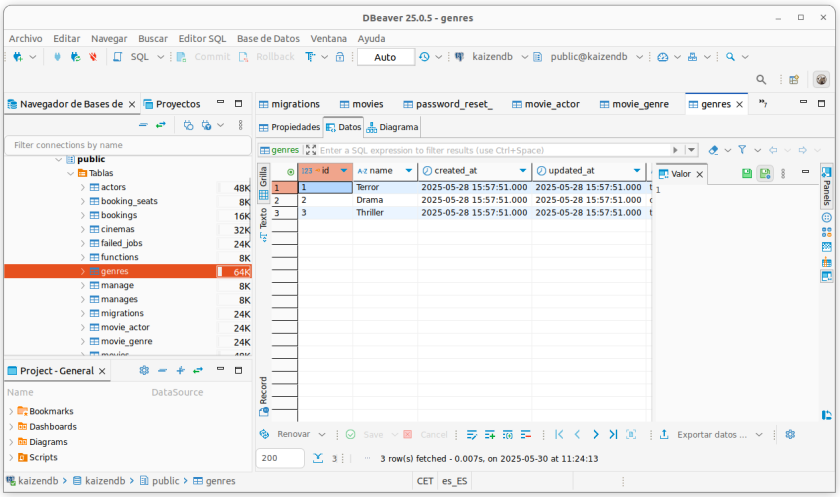
PostgreSQL destaca por su capacidad para manejar operaciones complejas de forma eficiente, su escalabilidad en proyectos grandes, y su amplia compatibilidad con herramientas de desarrollo modernas.

2.3.3.1. Sistema visual

Para la gestión visual de la base de datos se ha optado por DBeaver, una herramienta ampliamente reconocida por su eficacia, versatilidad y facilidad de uso.

DBeaver permite trabajar con múltiples sistemas de gestión de bases de datos (como PostgreSQL, MySQL, SQLite, entre otros), lo cual aporta una gran flexibilidad al entorno de desarrollo.

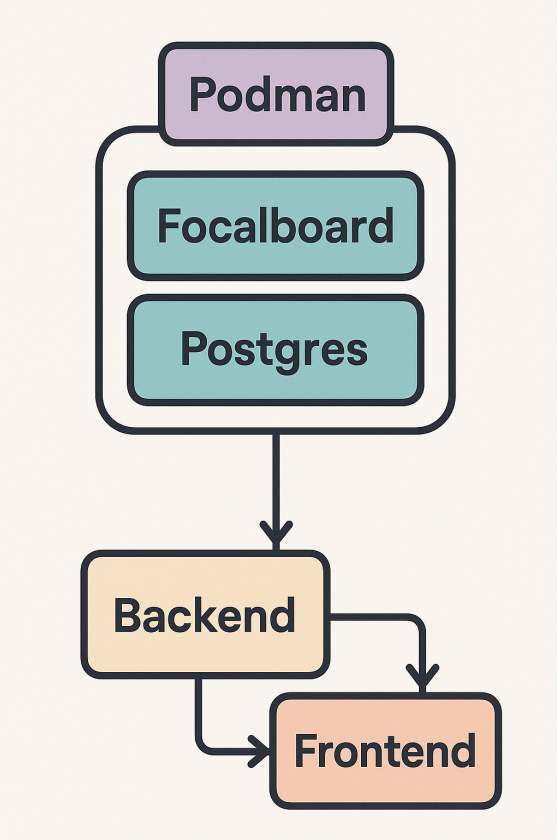
Ofrece funcionalidades avanzadas como visualización gráfica de esquemas, ejecución y edición de consultas SQL, navegación por tablas y datos, así como exportaciones en distintos formatos.



2.3.4. Contenerización

Podman es una herramienta de código abierto que permite crear, ejecutar y gestionar contenedores y pods. Su funcionamiento es similar al de Docker, con la diferencia clave de que no requiere un demonio en ejecución con privilegios de root, lo que mejora significativamente la seguridad del sistema.

En el proyecto, he utilizado Podman para contenerizar tanto Focalboard (una herramienta de gestión de tareas, sobre la cual profundizaré más adelante) como la base de datos que utiliza. De este modo, todo el entorno relacionado con Focalboard se encuentra encapsulado en contenedores, lo que facilita su despliegue y mantenimiento.



Para la gestión de estos contenedores, he desarrollado scripts específicos. Uno de ellos permite ejecutar comandos relacionados con los contenedores de Podman (como iniciar, detener o consultar su estado), mientras que otro está orientado al arranque y control del servicio Focalboard en particular.

Contenerización de PostgreSQL

La base de datos principal del sistema se ejecuta en un contenedor independiente con PostgreSQL 15, configurado mediante variables de entorno y archivos de inicialización personalizados. Este contenedor gestiona la persistencia de datos a través de un volumen nombrado (kaizen_pgdata), y expone el puerto 5433 para evitar conflictos con instalaciones locales.

El archivo podman-compose.yml define esta configuración de forma estructurada y declarativa, mientras que el script pg-podman.sh permite su control mediante comandos.

Contenerización de Focalboard

En este sistema, Focalboard se ejecuta en un contenedor propio, completamente aislado y gestionado mediante Podman.
El contenedor se inicia mediante el script fb-podman.sh, que proporciona una interfaz sencilla para operaciones habituales.

Volúmenes y persistencia de datos

Ambos servicios hacen uso de volúmenes nombrados en Podman para asegurar que los datos persisten incluso cuando los contenedores se detienen o eliminan. Esto permite restaurar el entorno rápidamente sin perder configuraciones ni registros importantes.

Servicio	Volumen	Uso principal
PostgreSQL	kaizen_pgdata	Archivos de base de datos
Focalboard	kaizen_focalboard_data	Tareas, tableros y adjuntos

2.3.5. Control de versiones

Git

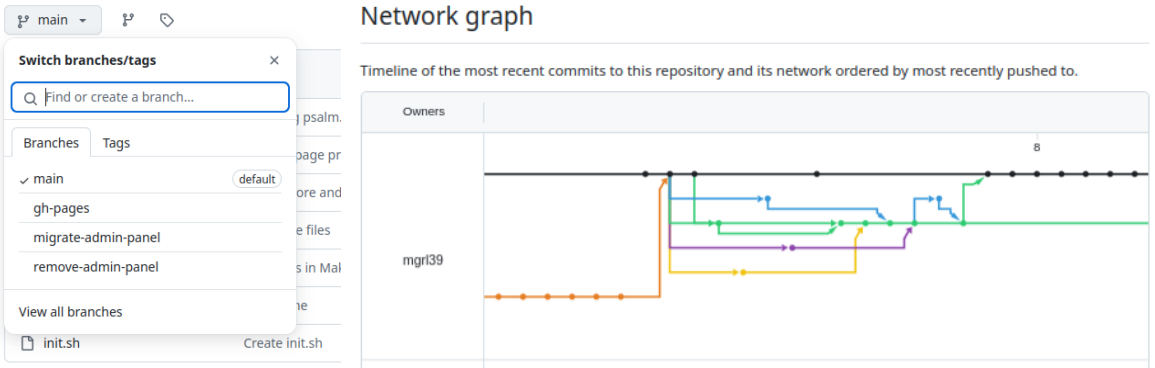
Git es un software de control de versiones pensando en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Esto quiere decir que pueden existir distintas vidas del proyecto en el mismo, y poder transportar el proyecto a distintas máquinas de manera relativamente rápida.

Github

En este proyecto, GitHub desempeña un papel fundamental, ya que no solo actúa como plataforma de almacenamiento y control de versiones del código fuente, sino que también aprovecho varias de sus funcionalidades avanzadas.

Entre estas herramientas destacan GitHub Actions y las integraciones externas, que permiten automatizar flujos de trabajo directamente desde el propio repositorio. Esto ha permitido establecer un entorno de desarrollo automatizado que optimiza tareas como el despliegue, las pruebas o la integración continua.

Dado el peso e impacto que tiene GitHub en la organización del proyecto, más adelante se dedica un apartado específico de esta memoria a detallar su uso y configuración como una solución eficiente para la gestión de proyectos y tareas.



2.3.6. Dominio y acceso al sistema

Para facilitar el acceso al sistema de forma clara y profesional, se ha configurado un subdominio personalizado: kaizen.doncom.me. Este subdominio apunta a la instancia del frontend desplegada automáticamente en Vercel, lo que permite acceder a la interfaz desde cualquier dispositivo sin depender de rutas locales o direcciones técnicas.

Es importante tener en cuenta que actualmente solo está desplegada la parte del frontend. El proyecto en sí incluye tanto backend como frontend, pero no se ha desplegado el backend en producción para evitar los costes asociados a servidores privados (VPS). Por tanto, en la versión pública no es posible completar acciones que requieran conexión con la base de datos, como el registro de usuarios, autenticación o gestión de reservas.

Sin embargo, está plenamente funcional en local y se encuentra en desarrollo activo. En el siguiente apartado se detalla cómo se gestiona el despliegue automático del frontend y cómo se podría ampliar a futuro para incluir el backend.

2.3.7. Herramientas de desarrollo

Editor de código multiplataforma utilizado para escribir, organizar y depurar el código del proyecto con soporte para extensiones y Git.

Gestor de paquetes de Node.js usado para instalar y administrar dependencias del frontend, como herramientas de compilación y librerías JS.

Gestor de dependencias PHP empleado para instalar paquetes necesarios en Laravel y mantener el backend organizado

Archivo de automatización utilizado para ejecutar tareas comunes del proyecto (como iniciar el servidor o compilar assets) con comandos simples

3. Desarrollo

3.1. Estrategia de desarrollo

La estrategia de desarrollo de este proyecto ha sido la de construir una aplicación nueva desde cero, sin reutilizar plantillas ni sistemas externos complejos.
Aunque no se ha realizado un análisis técnico profundo ni estudios comparativos formales, sí se ha observado el funcionamiento de otras páginas web de reserva de entradas (como Cinesa o Pathé) para tener una idea general de qué funcionalidades se suelen ofrecer y cómo está estructurada la experiencia de usuario.

A nivel técnico, se optó por utilizar dependencias y librerías modernas (como SvelteKit, Bootstrap o Laravel) no solo para facilitar el desarrollo, sino también como una forma de aprendizaje, buscando entender su funcionamiento interno y buenas prácticas en proyectos reales.

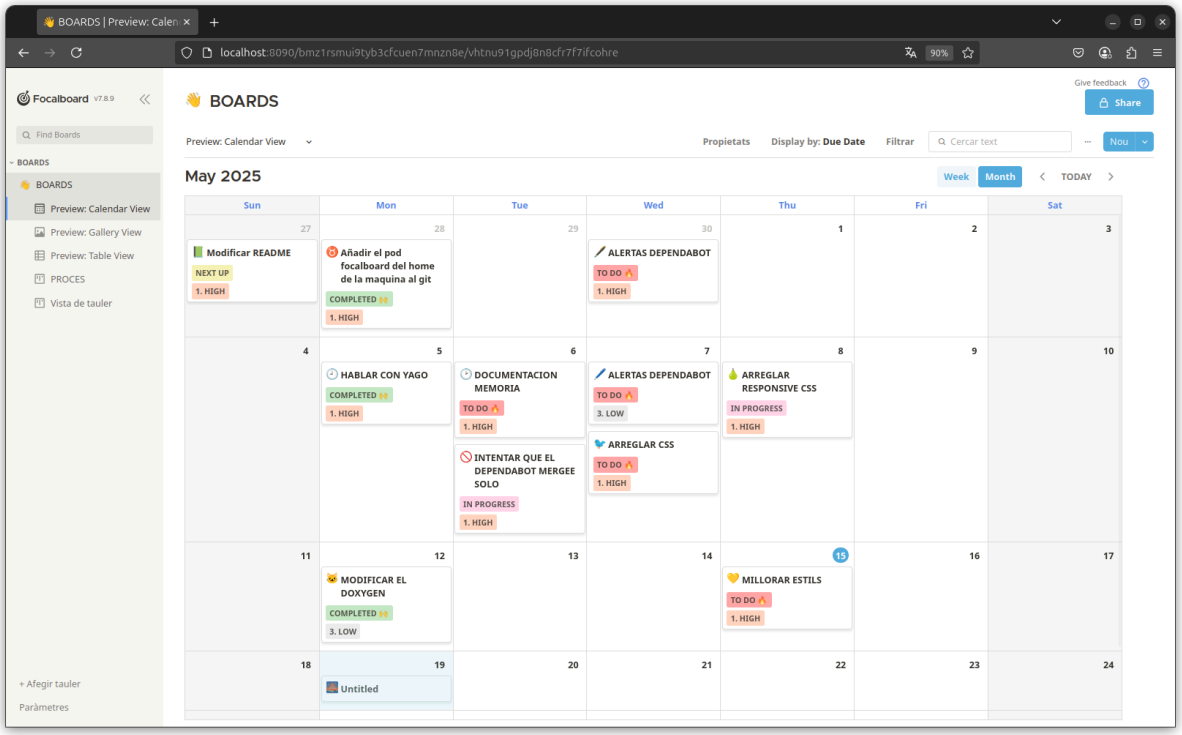
El desarrollo no siguió una metodología estricta, sino que fue adaptándose en función del avance personal, problemas detectados y mejoras posibles, manteniendo siempre como prioridad que el sistema fuera funcional, entendible y útil como proyecto integrador.

3.2. Metodología de trabajo

Al ser un proyecto individual, se descartó un enfoque tradicional como Waterfall y se optó por una gestión flexible inspirada en metodologías ágiles. No se aplican todos los componentes formales de Scrum, ya que no hay equipo ni reuniones diarias.

La planificación y el seguimiento de tareas se realiza con Focalboard, una herramienta visual open source que permite organizar el trabajo de forma flexible y adaptarse fácilmente a cambios o nuevas necesidades durante el desarrollo. Esto encaja con la filosofía open source del propio proyecto, promoviendo la transparencia y la colaboración futura de la comunidad.

Focalboard



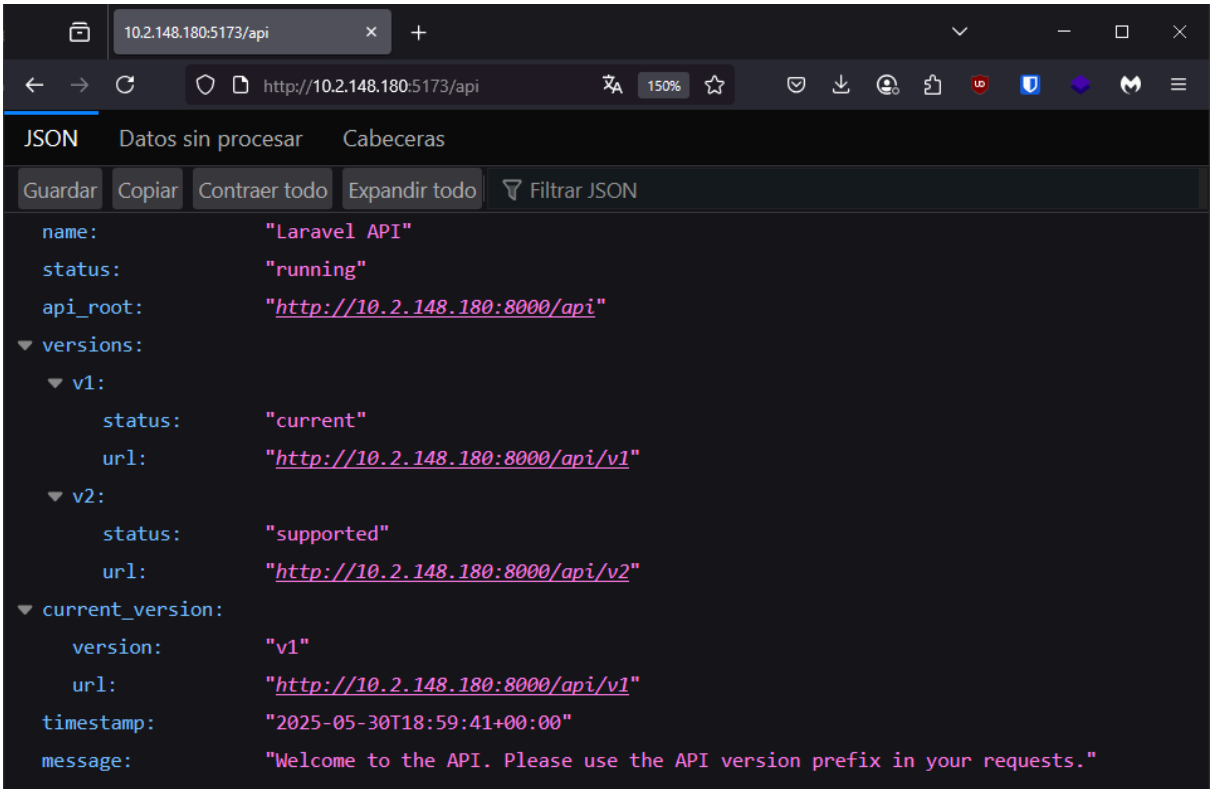
3.3. Documentación técnica del sistema

3.3.1. Backend descrito

El backend del sistema, actualmente implementado en Laravel 10, sirve como interfaz de acceso a los datos y a la lógica de negocio del sistema de gestión cinematográfica. Sin embargo, la arquitectura está diseñada para que sea lo más agnóstica respecto al framework posible, permitiendo su adaptación a otros entornos tecnológicos en el futuro.

Arquitectura

El sistema está diseñado siguiendo una arquitectura RESTful, en la que cada recurso cuenta con un conjunto de controladores especializados que encapsulan su lógica de negocio correspondiente. Para facilitar la evolución y mantener la compatibilidad hacia atrás, se ha implementado un sistema de rutas versionadas (por ejemplo, /api/v1). Este enfoque permite desarrollar futuras versiones (como una posible /api/v2) sin afectar al funcionamiento de la versión original, garantizando así estabilidad y escalabilidad en el tiempo.



SEPARACIÓN DE RESPONSABILIDADES	
Controladores	Gestionan las solicitudes HTTP y coordinan la lógica
Servicio	Encapsulan la lógica de negocio
Middlewares	Gestionan acceso y seguridad (condiciones de entrada)
Modelos Eloquent	Representan y manipulan los datos en la base de datos PostgreSQL

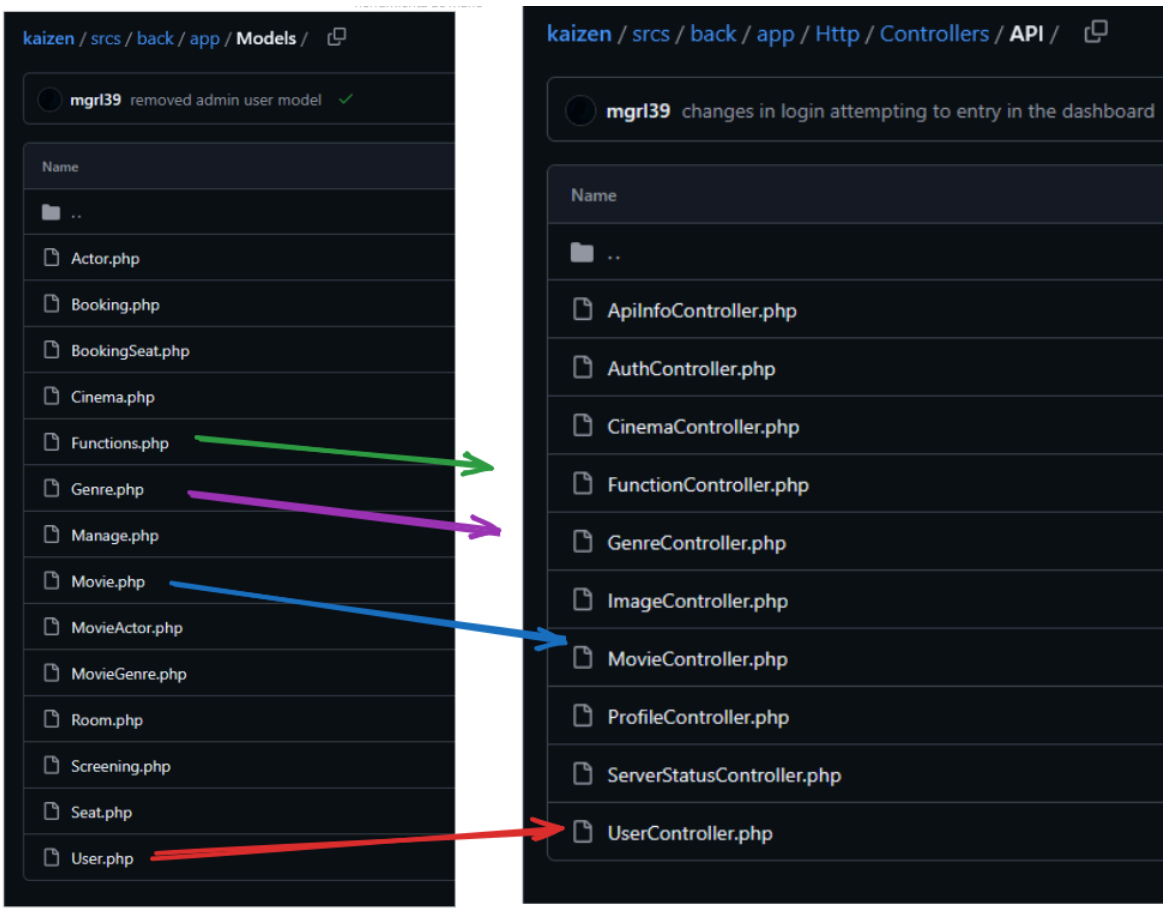
Seguridad y Autenticación

La autenticación se basa en JSON Web Tokens (JWT). La seguridad incluye validación CSRF (panel admin) y rate limiting. Asegurando protección contra ataques comunes.

Controladores y Endpoints

Cada entidad principal tiene un controlador específico (ej: MovieController, UserController). La organización sigue el patrón REST convencional:

ENDPOINTS	INSTRUCCIÓN
GET /movies	listar
GET /movies/{id}	obtener detalle
POST /movies	crear
PUT /movies/{id}	actualizar
DELETE /movies/{id}	eliminar

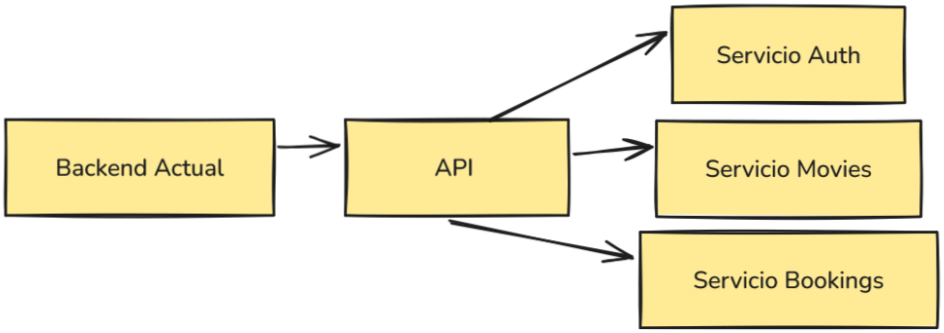


Persistencia y Modelado de Datos

La base de datos relacional (PostgreSQL) es gestionada mediante migraciones y seeders de Laravel. Haciendo referencia al diagrama de la base de datos, las entidades principales incluyen:

users	movies	cinemas	rooms	functions	bookings	genres

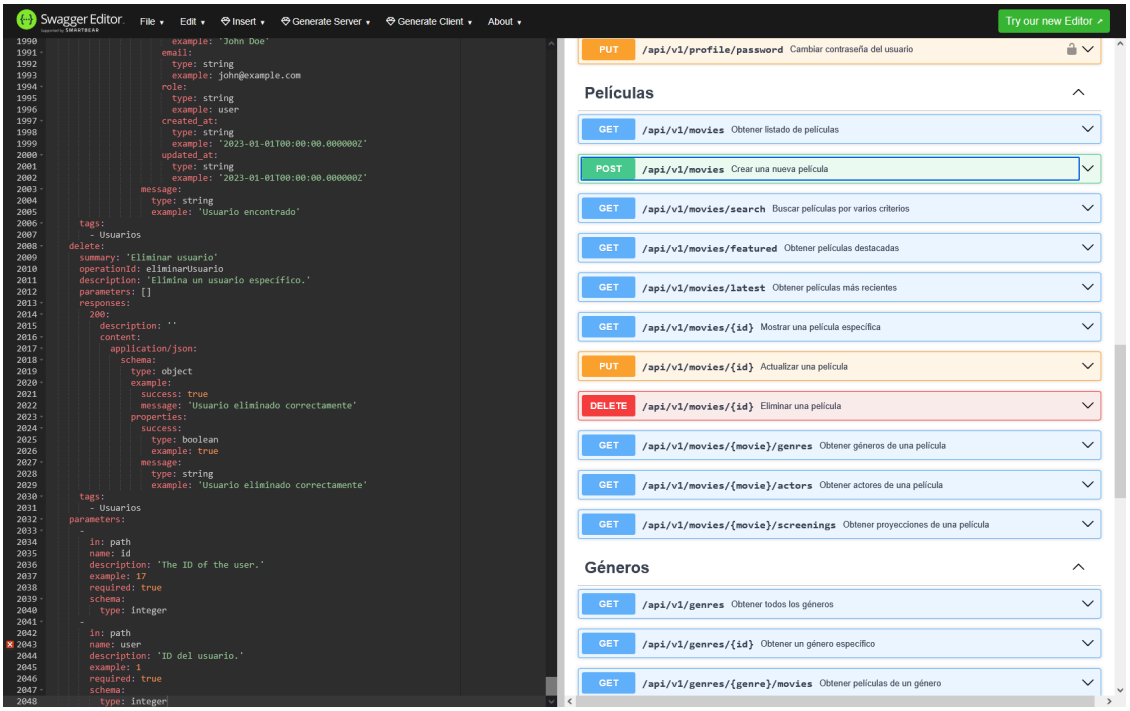
El modelo Eloquent proporciona métodos para acceder y navegar estas relaciones de forma intuitiva.



Herramientas de documentación.

El backend está integrado con Scribe, una herramienta que genera documentación interactiva de la API basada en las rutas y anotaciones del código.

Genera una vista HTML navegable, una colección Postman descargable y una especificación OpenAPI (Swagger).

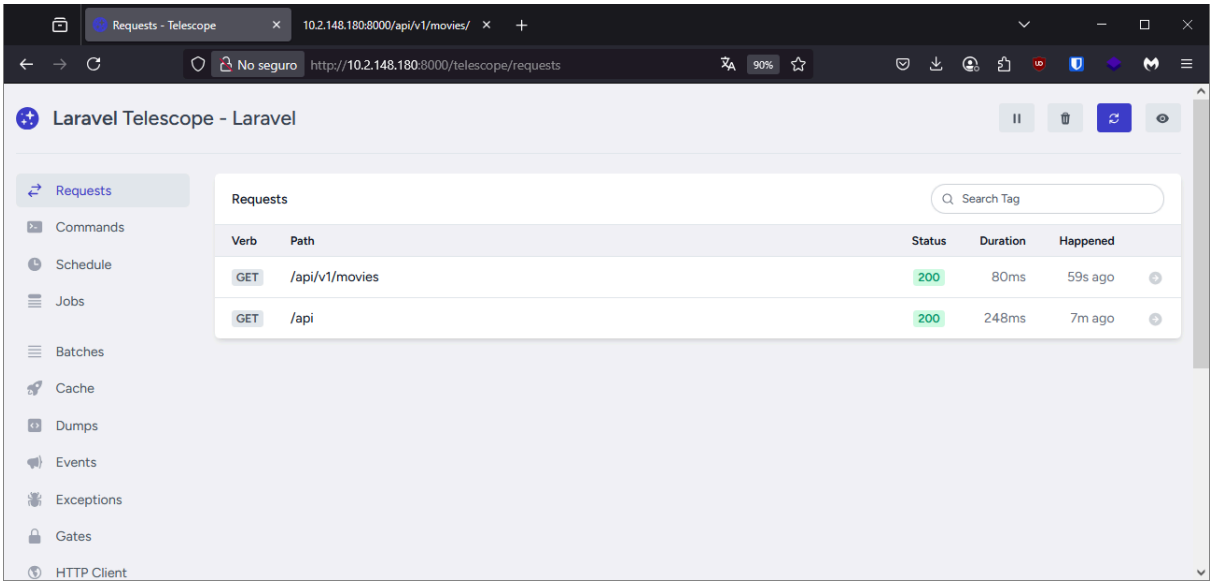


Supervisión y trazabilidad

Para facilitar el monitoreo durante el desarrollo y en producción, se ha habilitado Laravel Telescope, con las siguientes configuraciones destacadas:

Acceso vía URI /telescope

Esto permite revisar en tiempo real peticiones, errores, logs y eventos relevantes del sistema.



Evolutividad y Sostenibilidad Tecnológica

El sistema ha sido construido reconociendo que ninguna tecnología es permanente. Aunque Laravel 10 es una opción moderna y madura, se ha evitado un acoplamiento excesivo:

- La arquitectura está basado en principios universales (MVC, REST, SOLID)
- El frontend consume la API permitiendo sustituir el backend sin impacto directo.

De este modo, el sistema puede ser replicado o portado a diferentes entornos (Node.js, Django, .NET…).

3.3.2. Frontend descrito

La aplicación frontend de Kaizen Cinema está desarrollada utilizando SvelteKit, un framework moderno para crear aplicaciones web reactivas y modulares. Se implementa como SPA (Single Page Application), con ruteo basado en archivos, integración con API REST (Laravel backend) y soporte para internacionalización, gestión de temas, y despliegue en Vercel.

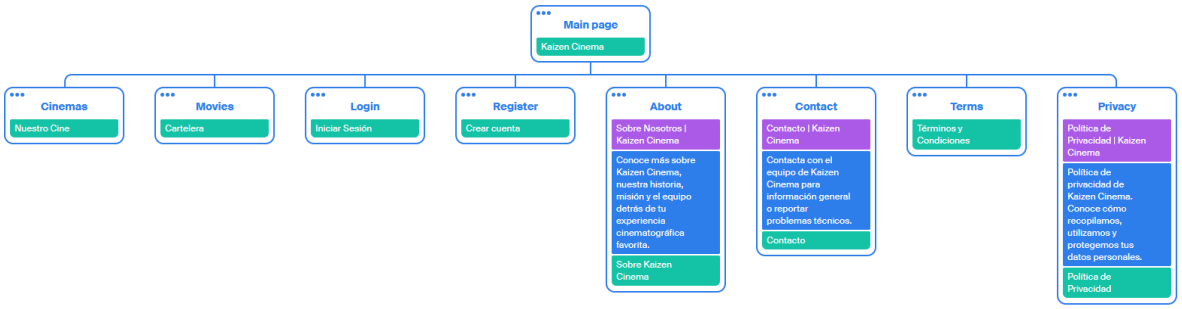
Tecnología base

SvelteKit	Para estructura de rutas y componentes
Bootstrap 5 + CSS	Estilos
Vite	Como sistema de build

Estructura general del sistema

/movies	Para estructura de rutas y componentes
/movies/[slug]	Detalle de película
/profile	Perfil de usuario
/register	Registro de usuario
/login	Inicio de sesión
/cinemas	Información sobre el

Cada página implementa su propia validación de formularios e integración con la API.



Páginas principales

Catálogo de películas	Permite al usuario explorar el conjunto completo de películas disponibles en el sistema. Cada película se representa mediante una tarjeta visual que incluye: Imagen, título, género principal y acceso al detalle completo. La página está conectada a la API para obtener los datos de películas y actualizar el listado.
Detalle de película	Accediendo desde el catálogo, se carga una página con: Sinopsis y datos generales Reparto, Proyecciones disponibles (con fecha y hora), Botones de reserva, Pestañas de navegación: Info / Proyecciones / Reseñas Se gestiona con rutas dinámicas basadas en slug (/movies/[slug]).
Perfil de usuario	Nombre, email y otros datos personales Cambio de contraseña Historial de reservas
Registro de usuario	Formulario para nuevos usuarios con validación cliente-servidor. Se implementan controles para: Email válido,Contraseña mínima y coincidente, Campos obligatorios, aceptación de términos Si el registro es exitoso, se guarda el token y se redirige al inicio.
Página del cine	Contiene información estática sobre las salas, servicios del cine, contacto y enlaces rápidos a reservas o cartelera. No depende de la API y está construida con datos hardcoded.

Componentes clave

Navbar	Menú de películas, cines, perfil Acciones de login/logout Cambiador de idioma Cambiador de tema (claro/oscuro) Comportamiento responsivo para móvil
Footer	Enlaces legales (privacidad, términos) Información de contacto Branding
Internacionalización	Soporte para múltiples idiomas (ES / EN) mediante archivos de traducción. Persistencia de idioma en localStorage Selección de idioma desde la barra de navegación

Flujo de autenticación

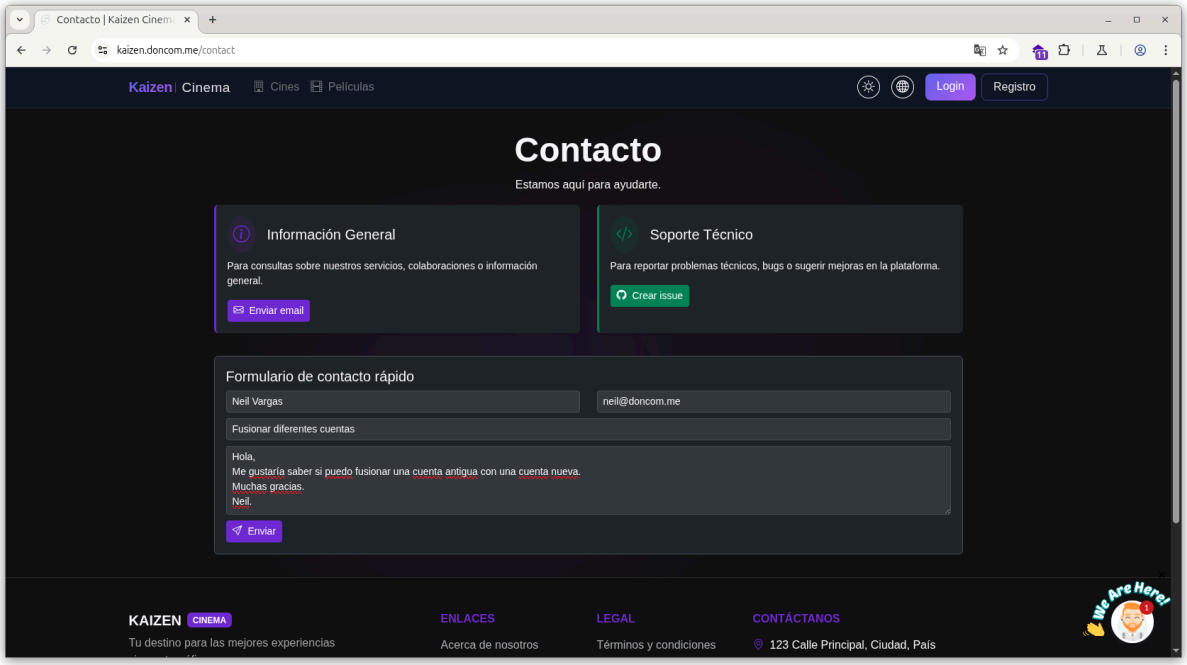
- El login genera un token JWT
- Se guarda en localStorage
- Las peticiones autenticadas incluyen Authorization: Bearer <token>
- El logout borra token y estado

- **Integraciones externas**
  - **Formspree**

Se ha añadido un formulario de contacto alojado directamente en el frontend, que permite a los usuarios comunicarse con el equipo del proyecto.

Para lograr esta funcionalidad, se ha integrado el servicio externo Formspree, que permite la recolección y el reenvío automático de datos introducidos en formularios HTML a un destino especificado (como una dirección de correo electrónico).

Se ha vinculado Formspree tanto a una cuenta de correo electrónico como a un canal personal de Telegram, mediante su servicio de automatización intermedio. De esta manera cada vez que un usuario completa y envía el formulario de contacto, se generan dos notificaciones: una a través del correo electrónico con los datos del mensaje, y otra mediante Telegram, lo que facilita una respuesta más ágil por parte del responsable del sistema.



Funcionamiento general

En el apartado de “Contacto” de la web, el formulario permite a cualquier usuario enviar mensajes breves relacionados con dudas generales o problemas técnicos. El formulario solicita los siguientes datos: nombre, correo electrónico y mensaje.

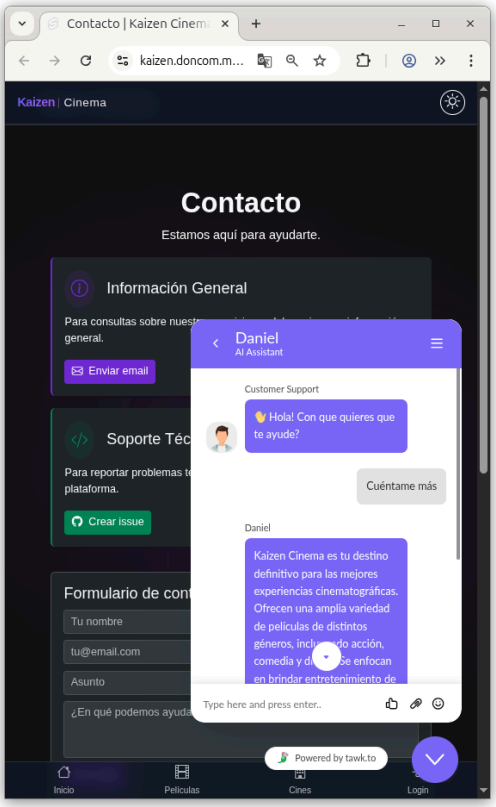
Como se ha podido observar, Una vez enviado el formulario, el sistema realiza dos acciones automáticas:

- Envía un correo electrónico a la cuenta configurada (como puede verse en la parte izquierda de la captura).

- Envía una notificación directa a Telegram, a través de un bot conectado a un canal privado

3.4. tawk.to

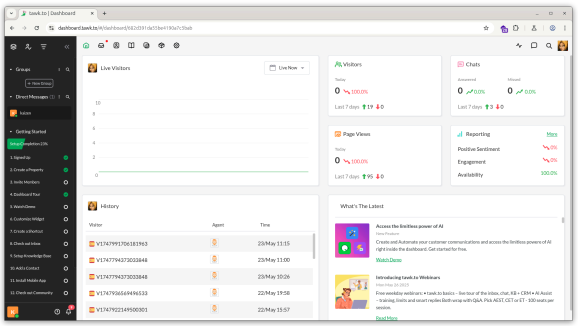
La segunda integración externa que se ha implementado en el proyecto es **tawk.to**, una herramienta gratuita para incorporar un sistema de **chat en tiempo real** dentro del sitio web.

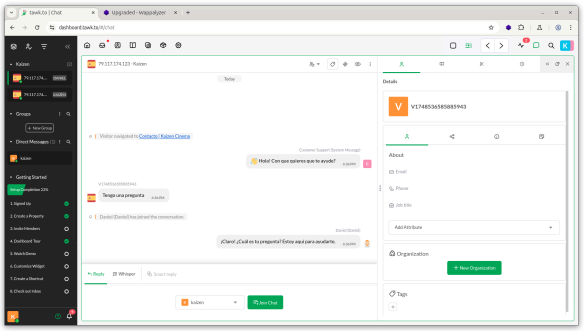


tawk.to permite que los usuarios puedan iniciar conversaciones directamente desde el frontend, facilitando la atención al cliente, la resolución de dudas y el soporte en general.

Su principal objetivo es dar una **experiencia de soporte en vivo**, y mostrar que el sistema está preparado para integrar herramientas de atención al usuario en versiones futuras. Se puede configurar que un bot te responda pero un administrador puede entrar a la conversación y hablar con el usuario directamente por chat, consiguiendo así un trato más humano

En las siguientes dos imágenes se muestra el panel de administración, donde se reciben las alertas de mensajes enviados por los usuarios. Desde este panel, es posible que un agente responda a las consultas o dudas planteadas.





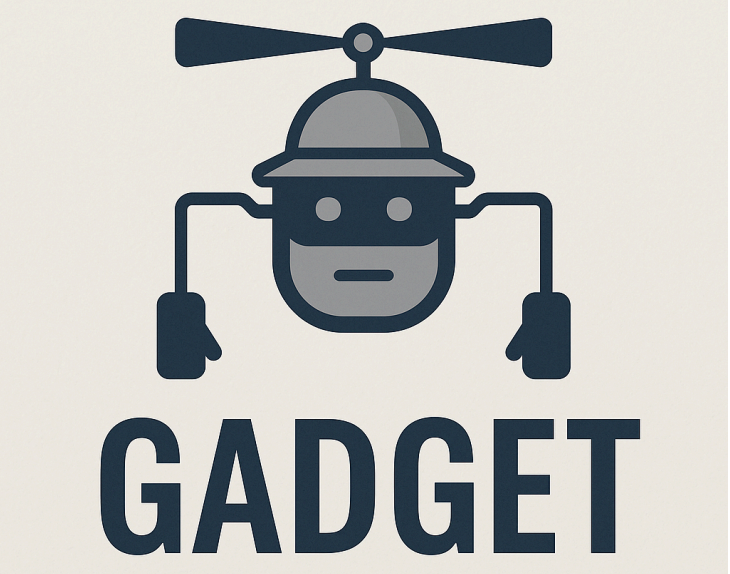
3.5. Gadget

Gadget es un script creado para extraer información de películas desde la web de Cinesa y guardarla en una base de datos PostgreSQL. Se utiliza para automatizar la recolección de datos como títulos, horarios, detalles y carteles de películas.

El sistema se ejecuta desde la línea de comandos y también cuenta con una interfaz web básica. Usa tecnologías como Selenium y BeautifulSoup para navegar por la web y extraer el contenido, y guarda los datos en formato JSON.

Se desarrolló para alimentar el sistema de reservas del proyecto con información real y actualizada sin necesidad de introducirla manualmente.

En el anexo se encuentra el repositorio del proyecto.



4. Control de versiones y gestión del proyecto con GitHub

El control de versiones y la organización del desarrollo se han gestionado a través de GitHub, siguiendo prácticas comunes de proyectos open source y manteniendo una estructura clara y coherente del repositorio.

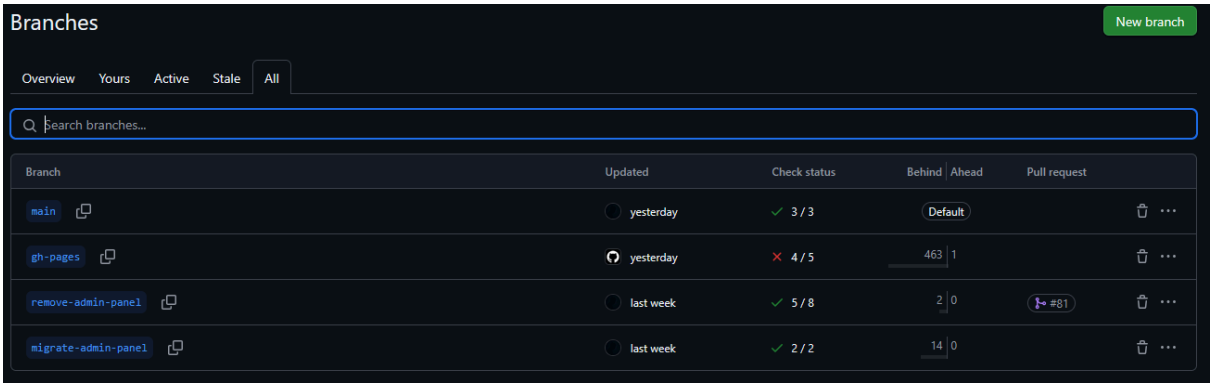
4.1. Organización del repositorio

El repositorio se estructura en carpetas separadas para frontend, backend y configuración, facilitando el crecimiento del sistema y la comprensión del código.

CARPETA	CONTENIDO
srcs/front/	Contiene la aplicación frontend desarrollada con Svelte.
srcs/back/	Incluye el backend en Laravel y la lógica de negocio.
srcs/conf/	Agrupa scripts, definiciones de contenedores y configuraciones de infraestructura.
.github/	Define workflows de integración continua, seguridad y documentación.

Flujo de trabajo con Git

RAMA	DEFINICIÓN
main	Rama estable para despliegues o versiones finales
gh-pages	Rama dedicada a la publicación automática de documentación técnica.
Ramas auxiliares	Creadas para nuevas funcionalidades, pruebas o correcciones puntuales.



4.2. Automatización y CI/CD con GitHub Actions

GitHub Actions convierte el propio repositorio en un entorno de desarrollo inteligente. Mediante una serie de workflows definidos en archivos YAML, se automatizan tareas clave del ciclo de vida del software, sin necesidad de intervención manual.

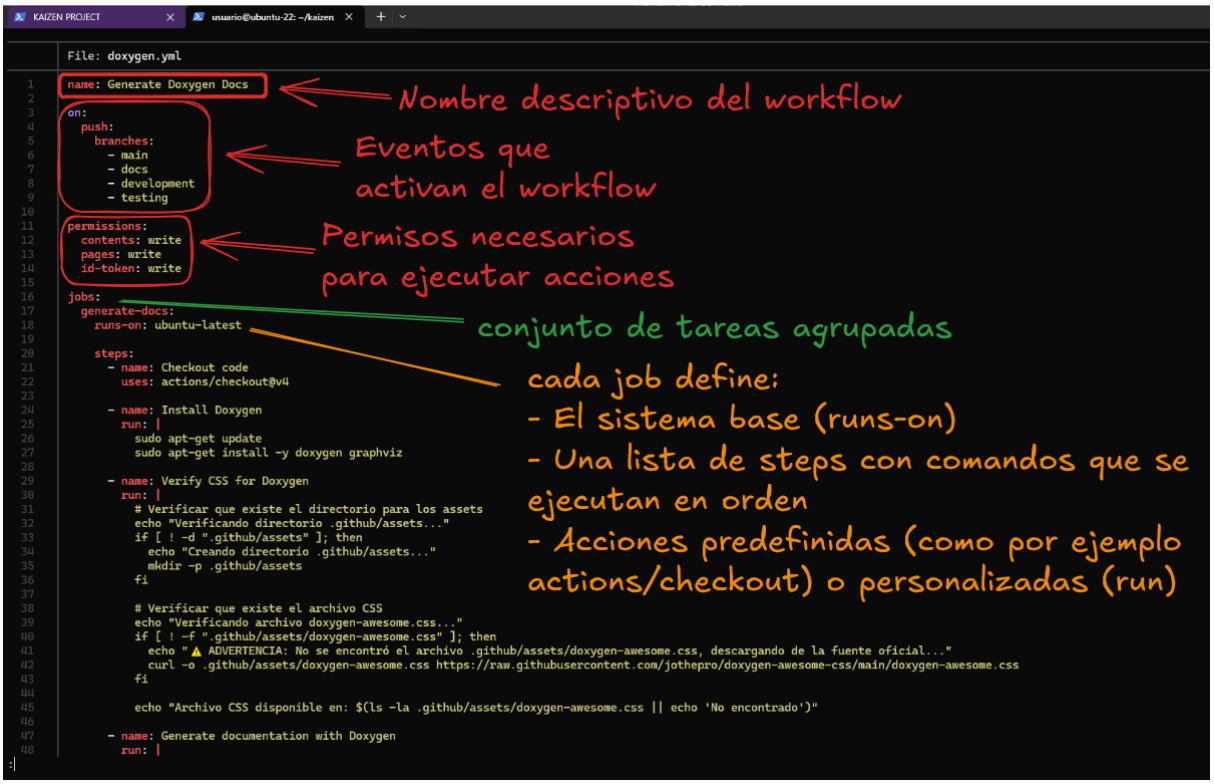
Estos workflows se activan automáticamente ante eventos como push, pull_request o incluso en intervalos programados, asegurando que el sistema se revise, documente y actualice de forma continua. Gracias a esta infraestructura, se minimizan errores humanos, se aceleran los procesos de validación y se mejora la calidad del código desde el primer commit.

Tareas automatizadas con GitHub Actions
Ejecutar análisis estático de código con herramientas como Psalm, PHPStan y PHPMD.
Validar la consistencia y seguridad de dependencias (Composer para backend, npm para frontend).
Detectar vulnerabilidades en librerías y configuraciones mediante escaneo automático.
Generar y publicar documentación técnica del backend utilizando Doxygen.
Automatizar actualizaciones de dependencias mediante Dependabot
Gestionar flujos de despliegue continuo del frontend a través de plataformas como Vercel.

4.2.1. Formato de los archivos workflow

Para definir los workflows, GitHub Actions utiliza archivos en formato YAML ubicados en la ruta estándar (**.github/workflows/**).

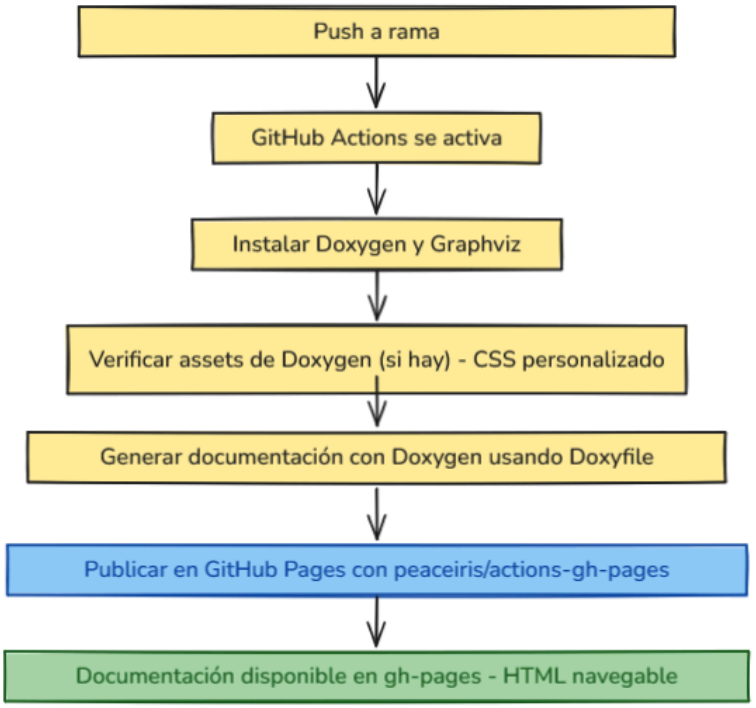
La siguiente imagen explica la estructura general de estos archivos, donde se especifican los eventos que activan el workflow, los permisos requeridos y las tareas automatizadas agrupadas en distintos jobs.

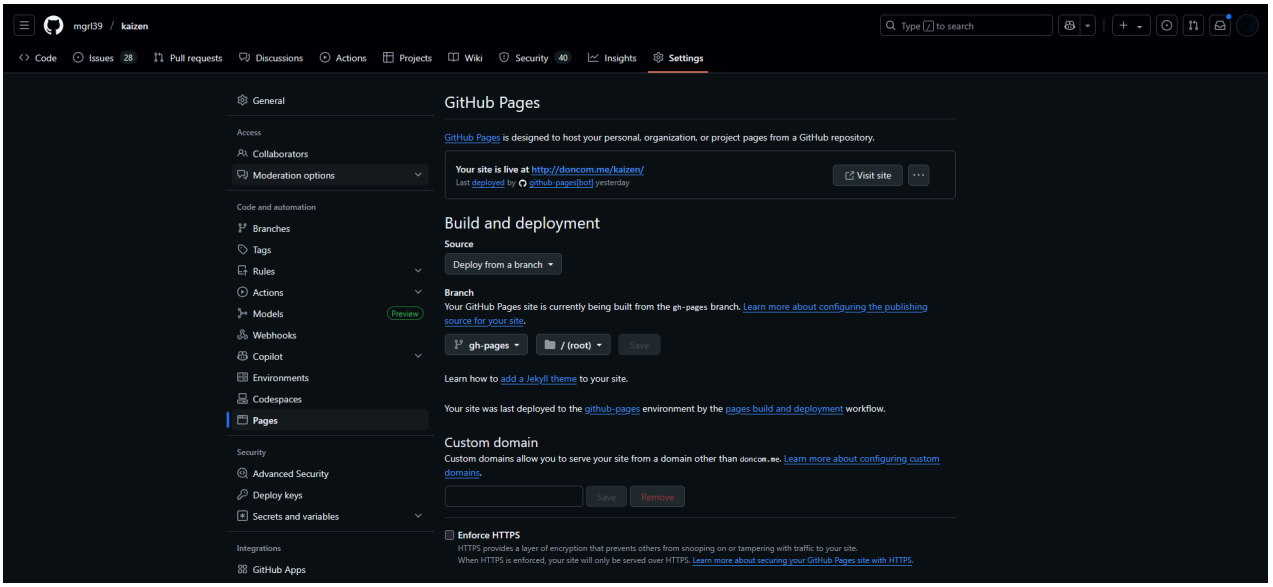


Formato / Herramienta	**Similitud**
Script .sh (Bash)	Define comandos que se ejecutan en orden, de forma similar a los steps dentro de un job en un workflow.
Makefile	Cada regla del Makefile puede compararse con un job, ya que automatiza tareas específicas bajo ciertas condiciones.
Archivo Ansible (.yml)	Utiliza formato YAML y organiza tareas en bloques (tasks) que se ejecutan de forma secuencial sobre uno o varios hosts, como un workflow estructurado.

4.2.2. Generación automática de documentación con Doxygen

El sistema de documentación utiliza Doxygen para procesar los comentarios estructurados del código fuente y generar documentación navegable. Esta se publica de forma automática en GitHub Pages, facilitando la exploración del sistema sin necesidad de revisar directamente el código.



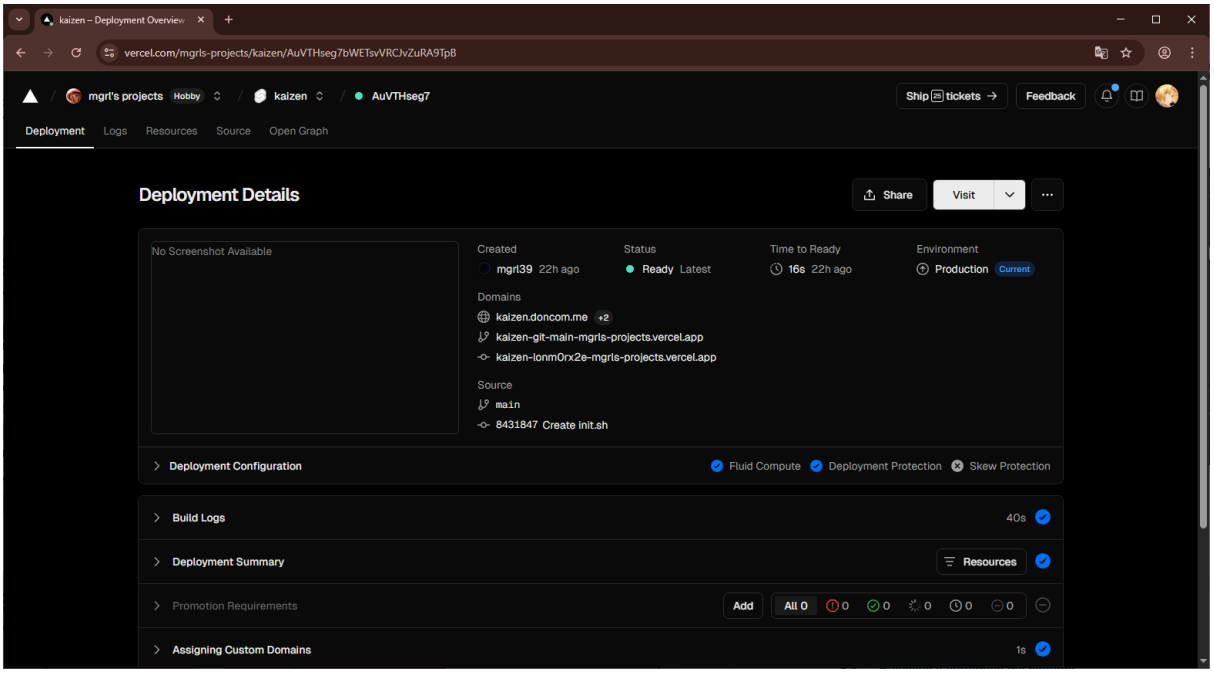


4.3. Despliegue automático del frontend

Uno de los aspectos destacables del proyecto es el sistema de auto-deploy configurado en el repositorio. Se ha integrado Vercel de modo que el frontend se despliega automáticamente cada vez que se realiza un push.

El proyecto está estructurado en tres partes. Dentro del directorio Source se encuentran tanto el frontend como el backend, aunque actualmente solo el frontend está configurado para desplegarse de forma automática. El desarrollo del frontend se realiza utilizando SvelteKit y, gracias a esta integración, la aplicación puede ser accedida desde cualquier lugar mediante el dominio kaizen.doncom.me.

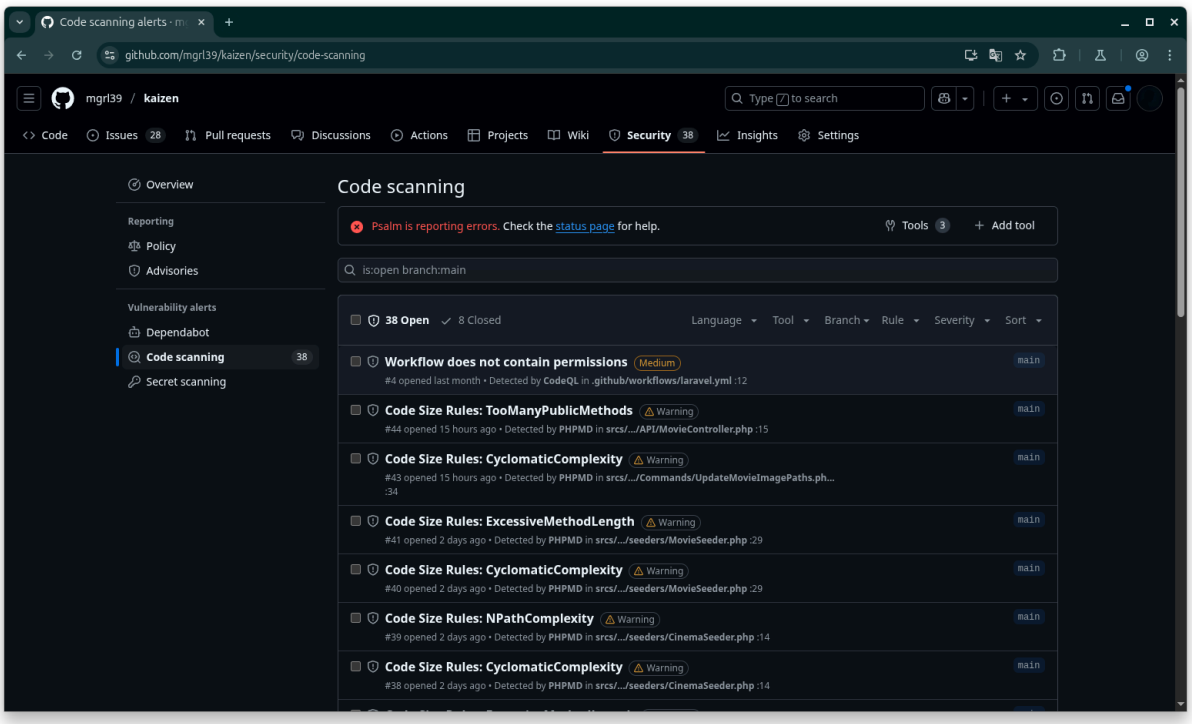
Cabe aclarar que, por el momento, no se ha implementado lógica de backend: no existe base de datos, persistencia ni datos reales. El despliegue actual funciona únicamente como una demostración visual y funcional de algunas integraciones externas que ya están conectadas al proyecto, como el formulario de contacto (que se encuentra operativo) y un sistema de chat que funciona mediante un servicio externo.



4.4. Seguridad y configuración del repositorio

Con el objetivo de asegurar buenas prácticas en el desarrollo y mantener la integridad del código fuente, se ha utilizado el apartado de “Security” de GitHub para habilitar herramientas de análisis estático y detección de vulnerabilidades.

CODE SCANNING ALERTS	Permite escanear el código automáticamente en busca de problemas de calidad o seguridad. Se integró el análisis con Psalm y PHP Mess Detector (PHPMD) para evaluar la complejidad ciclomática, longitud de métodos, permisos de workflows, entre otros aspectos.
DEPENDABOT	Monitoriza dependencias declaradas en el proyecto y genera alertas cuando alguna presenta una vulnerabilidad conocida. Esto permite mantener el software actualizado y seguro frente a exploits externos.
SECRET SCANNING	Útil para detectar si accidentalmente se han subido claves API, tokens o secretos al repositorio, ayudando a evitar filtraciones de información sensible



Las advertencias no representan fallos críticos, pero sirven como guía para mejorar la calidad y mantenibilidad del código. El uso de estas herramientas refuerza las buenas prácticas de desarrollo seguro y control de calidad dentro del ciclo de vida del software.

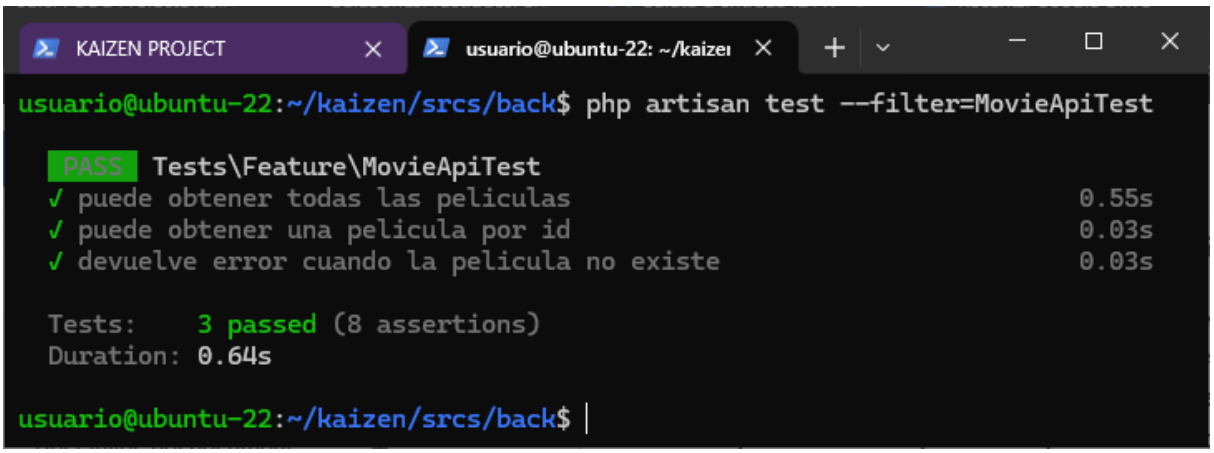
5. Pruebas

5.1. Unitarias

Como parte del proceso de validación del backend, se desarrolló una primera batería de pruebas automáticas centradas en garantizar el comportamiento esperado de la API en operaciones clave.

En esta fase, se comenzó por los endpoints vinculados a la gestión y consulta de películas, verificando su respuesta ante diferentes escenarios de uso. Estas pruebas se ejecutan utilizando el sistema de testing integrado de Laravel (php artisan test), lo que permite detectar posibles regresiones o inconsistencias en la lógica de los controladores de forma eficiente.

Aunque se trata de una base inicial, este enfoque establece una estructura sólida que puede ampliarse progresivamente y que ya contribuye a mejorar la estabilidad y mantenibilidad del sistema.

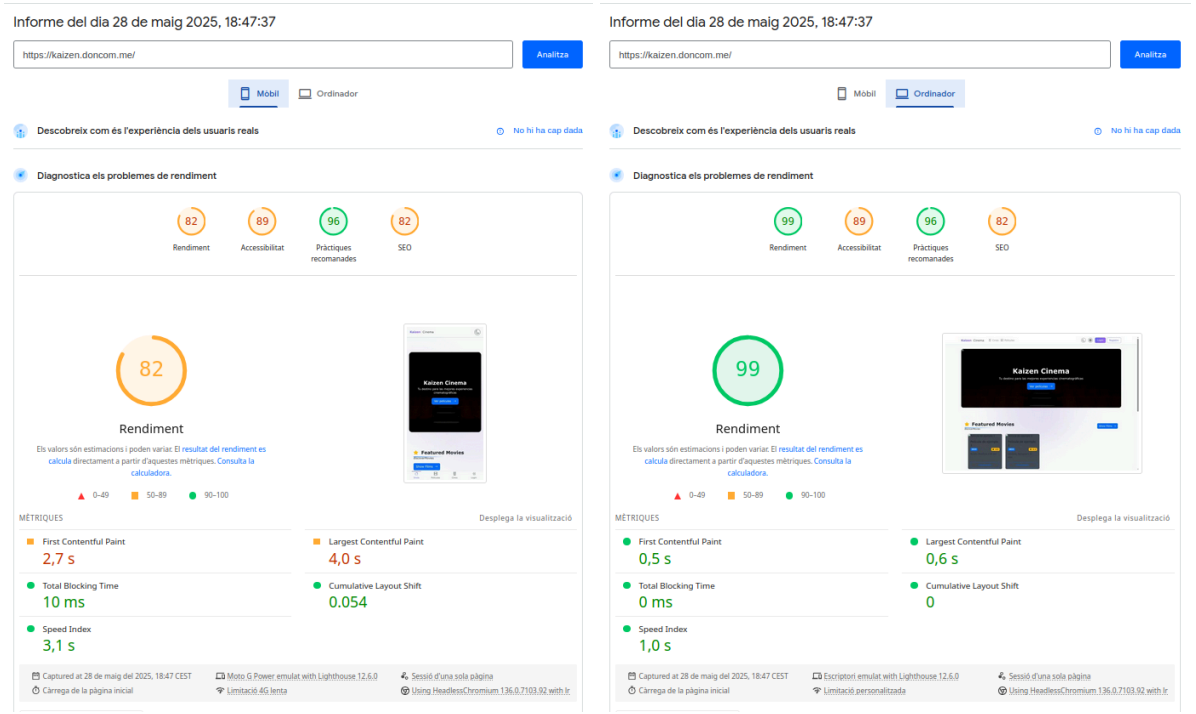


5.2. Rendimiento

Para revisar el rendimiento, se utilizó PageSpeed Insights, una herramienta de Google para analizar el rendimiento de un sitio web. Esta, además, ofrece sugerencias para mejorar los errores encontrados.

Mediante esta tecnología, fueron detectados resultados positivos, aunque se identificaron posibles mejoras como implementaciones futuras.

- Dispositivos móviles: 82/100
- Ordenadores de escritorio: 99/100



6. Conclusiones

El desarrollo del sistema ha sido realizado íntegramente por una sola persona, lo que ha supuesto un reto considerable tanto a nivel técnico como organizativo. A pesar de ello, se ha conseguido avanzar de forma continua y estructurada, cumpliendo la mayoría de los objetivos inicialmente planteados. Se ha logrado implementar una base de datos funcional, una API RESTful estable, y una interfaz de usuario operativa que permite el registro, la visualización de películas y la reserva de entradas.

La elección de tecnologías y herramientas se basó principalmente en la familiaridad previa y en criterios de accesibilidad, lo que facilitó la ejecución del proyecto y el aprendizaje activo durante todo el proceso. No obstante, se ha identificado un área crítica para el futuro: la dependencia de librerías externas. Aunque estas han permitido acelerar el desarrollo, también podrían comprometer la sostenibilidad del sistema a largo plazo. Por ello, se plantea como línea de mejora la progresiva sustitución de componentes externos por soluciones propias más controladas, con el fin de aumentar la estabilidad, mantener el control total del código y garantizar la continuidad del software en el tiempo.

De cara a futuras versiones, el sistema podría ampliarse con nuevas funcionalidades como la integración de pasarelas de pago, motores de recomendación, o tecnologías inmersivas, así como una mejora visual y de accesibilidad. También se abre la puerta a la participación de la comunidad open source como vía para consolidar el proyecto y hacerlo crecer con garantías.

7. Bibliografía

1. Contributors, M. O. J. T. a. B. (n.d.). *Get started with Bootstrap*. https://getbootstrap.com/docs/5.3/getting-started/introduction/
2. *Cinema*. (s. f.). https://cinema.noeosorio.com/
3. Nadermx. (n.d.). *GitHub - nadermx/backgroundremover: Background Remover lets you Remove Background from images and video using AI with a simple command line interface that is free and open source.* GitHub. https://github.com/nadermx/backgroundremover
4. *The MIT license*. (n.d.). Open Source Initiative. https://opensource.org/license/MIT
5. Scribbr. (2024, October 23). *Free Citation Generator | APA, MLA, Chicago | Scribbr*. https://www.scribbr.com/citation/generator/
6. Para poder investigar sobre cómo se han hecho ciertas páginas utilicé la extensión de https://www.wappalyzer.com/
7. Macke, S. (n.d.). Download Dia for free! Install the popular Open Source diagramming solution on your Windows PC. Easy setup using the installation wizard. http://dia-installer.de/download/index.html.en
8. Modelio Open Source - UML and BPMN free modeling tool. (s. f.). https://www.modelio.org/
9. Mitraissiteadmin. (2022, November 23). *How to Deploy Next.js with Vercel*. Mitrais. https://www.mitrais.com/news-updates/how-to-deploy-next-js-with-vercel/
10. Podman. (n.d.). https://podman.io/
11. *Installation - LaRavel 12.X - the PHP framework for web Artisans*. (n.d.). https://laravel.com/docs/12.x
12. Doxygen homepage. (n.d.). https://www.doxygen.nl/
13. PageSpeed Insights. (n.d.). PageSpeed Insights. https://pagespeed.web.dev/

8. Anexos

8.1. Repositorio de Github: www.github.com/mgrl39/kaizen

8.2. Repositorio de Gadget www.github.com/mgrl39/gadget

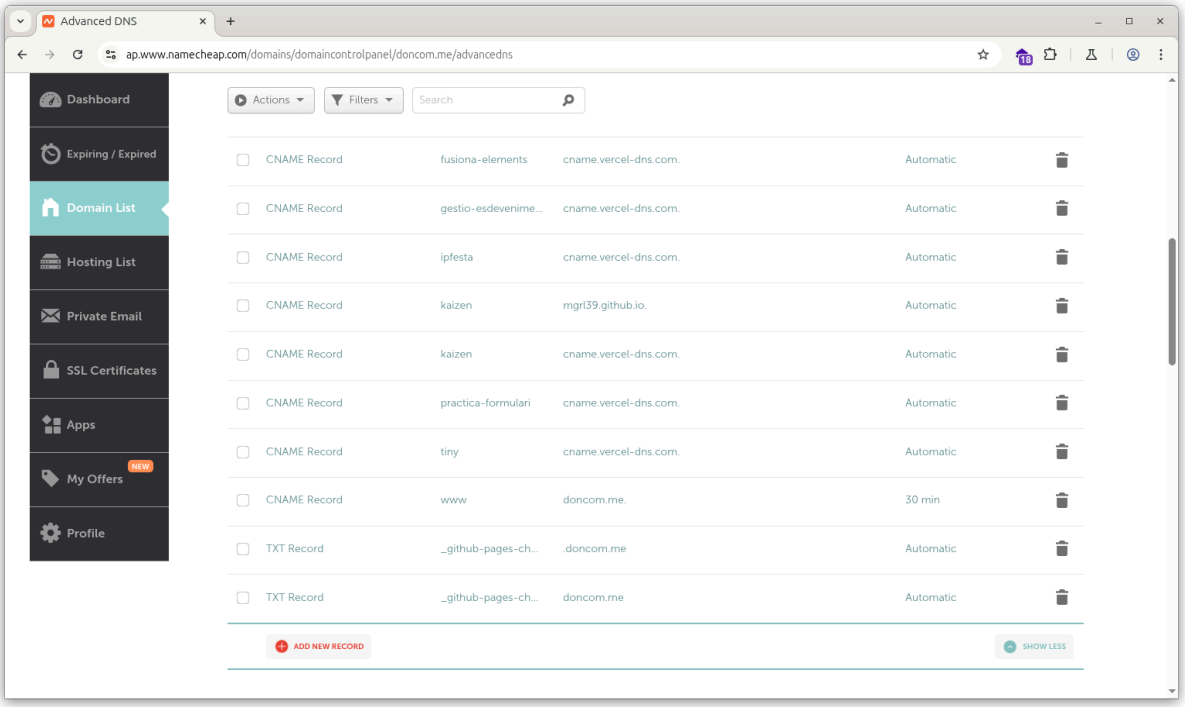
8.3. Cinesa butacas https://ayuda.cinesa.es/hc/es/articles/7535729761437--Qu%C3%A9-tipos-de-cine-tenemos-Qu%C3%A9-tipos-de-sala-y-butacas-hay

8.4. 2025 Pla de prevenció de RISCOS LABORALS - Neil Mohamed Dani - https://drive.google.com/file/d/1zcMonkfdeGAAqWZjE2dobFj_pHKhHxBY/view?usp=sharing

8.5. Enlaces a páginas de referencia

Páginas de referencia	Alternativas Open Source
Cinesa https://www.cinesa.es/ Pathé https://www.pathe.fr/ Cinemaxx https://www.cinemaxx.de/ UCI Kinowelt https://www.uci-kinowelt.de/	LibreBooking https://github.com/LibreBooking/app CinemaPlus https://github.com/georgesimos/cinema-plus

8.6. Namecheap sub dominios.



9. Glosario

1. **Apache2:** Servidor web HTTP de código abierto.
2. **CI/CD: (Integración continua / Despliegue continuo).** Estrategia que automatiza el proceso de pruebas, integración y despliegue del software.
3. **Contenerización:** Técnica que permite encapsular aplicaciones y sus dependencias en contenedores aislados para facilitar su despliegue y portabilidad.
4. **Doxygen:** Herramienta que genera documentación del código fuente automáticamente a partir de comentarios estructurados.
5. **Eloquent ORM:** Es una herramienta que viene con Laravel y sirve para trabajar con la base de datos usando código fácil de entender, en lugar de escribir directamente comandos complicados en lenguaje SQL. Permite guardar, buscar o modificar datos como si se estuviera trabajando con objetos o listas en el propio lenguaje de programación.
6. **Forja:** Plataforma de desarrollo colaborativo de software. Se enfoca hacia la cooperación entre desarrolladores para la difusión de software y el soporte al usuario. En este tipo de plataformas se albergan múltiples proyectos de software, en los que los desarrolladores han de registrarse para poder contribuir.
7. **Open Source:** Software cuyo código fuente es accesible y modificable por cualquier persona.
8. **Stakeholders:** Personas y grupos que de alguna manera se ven influenciadas o afectadas por las acciones de la empresa (tienen poder de influir en su funcionamiento independientemente de la relación contractual). Actualmente, las empresas se sitúan en estado de interdependencia mutua y debe tener a este grupo de personas en las decisiones empresariales. Este grupo incluye no solo a los accionistas y trabajadores de la empresa, sino también a proveedores, clientes, entidades reguladoras e incluso la competencia, dependiendo del contexto.
9. **Token de autenticación:** Elemento que se usa para identificar y validar a un usuario durante una sesión segura en una aplicación web.

10. Agradecimientos

Jordi Manzanera ElAidi como beta tester.

Rubén Arroyo, Marta Martinez, Yago Morales, David Tomas por el acompañamiento, aprendizaje, ayuda y correcciones durante el curso.

Jaume Oktorok por la motivación y Mohamed El Attar por ser una gran inspiración.