

REDFEST: App web de gestión de la programación de festivales de cine

Autora: Marina Rojo Urquiola

Consultor: Jordi Ustrell Garrigós

Profesor: Ferrán Adell Español

Tutor: Vicent Moncho Mas

Grado Multimedia

Ingeniería Web

Enero 2022

Créditos



Esta obra está sujeta a una licencia de Reconocimiento- NoComercial-SinObraDerivada [3.0 España de Creative Commons.](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

FICHA DEL TRABAJO FINAL

Título del trabajo:	<i>REDFEST: app web de gestión de la programación de un festival de cine</i>
Nombre del autor:	<i>Marina Rojo Urquiola</i>
Nombre del colaborador/a docente :	<i>Jordi Ustrell Garrigós</i>
Nombre del PRA:	<i>Ferrán Adell Español</i>
Fecha de entrega (mm/aaaa):	<i>MM/AAAA</i>
Titulación o programa:	<i>Grado Multimedia</i>
Área del Trabajo Final:	<i>Ingeniería Web</i>
Idioma del trabajo:	<i>Español</i>
Palabras clave	<i>Aplicación Web, Javascript, MEVN</i>
Resumen del Trabajo (máximo 250 palabras): <i>Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo</i>	
<p>Este proyecto se trata de una aplicación web que sirve para gestionar la base de datos y la calendarización de un festival de cine. Tras 15 años de experiencia laboral en diferentes festivales de cine como el de San Sebastián, la Seminci de Valladolid o el Europeo de Sevilla, he detectado una necesidad de optimización de ciertos procesos vinculados a la programación de cada edición. En todos los festivales se trata de un proceso manual y, por ende, muy lento y poco eficiente. A partir de este problema, el objetivo de esta aplicación web es que todos los departamentos del festival accedan a una única base de datos donde cada uno aporte y trabaje únicamente con la información que le corresponde. El proceso de mayor optimización es el de calendarización, en el que, teniendo en cuenta la información aportada por cada departamento, se puede generar automáticamente una primera versión de programa que será modificable por el curador de contenido correspondiente.</p> <p>El desarrollo de la aplicación web se va a realizar con las siguientes tecnologías: HTML para la estructura de la web, CSS con el framework Vuetify de Vue para los estilos, Node JS y el framework Express para el backend, MongoDB con Postman y Mongo Compass para las bases de datos, VueJS para el frontend. Estas tecnologías conforman el <i>full stack</i> denominado MEVN (Mongo, Express, Vue, Node), de manera que será una aplicación reactiva, generando una mejor experiencia de usuario.</p>	
Abstract (in English, 250 words or less):	
<p>This project is a web app to manage a film festival's database and schedule. After 15 years working at film festivals such as the one on San Sebastian, Seminci at Valladolid or the one at</p>	

Seville, I've realized that there's a need to optimize certain processes regarding the programme of each edition. This process is manual at all the film festivals, so it's a very slow and inefficient process. Taking this into account, the aim of this web app is that all the departments of a festival can have access to a unique database where they can work only with the information they should manage. The most powerful feature is the one regarding the programme, in which, considering all the information given by each department, it can automatically create a first version of the programme that can be modified by the corresponding content curator.

These are the technologies used for the development of the web app: HTML for the structure of the web, CSS with Vuetify framework for the styles, NodeJS and Express framework for the backend, MongoDB with Postman and Mongo Compass for the databases, VueJS for the frontend. These technologies are part of the MEVN full stack (Mongo, Express, Vue, Node), so the app will be a reactive web app, which is going to give a better user experience.

Cita

“Si no te gusta cómo son las cosas, cámbialas.” Jim Rohn

Agradecimientos

A toda mi familia por apoyarme desde el principio de esta aventura. A mi mujer Cris Codina, especialmente, por animarme, cuidarme y quererme tras horas y horas de cansancio delante de ordenador.

Y como cada cosa que hago en esta vida, a mi amoña, María Cuní, de la que aprendí tantas cosas y en quien me sigo inspirando.

Notaciones y Convenciones

Tipo de texto	Fuente	Tamaño	Formato
Título 1	Arial	20	Negrita
Título 2	Arial	13	Negrita
Título 3	Arial	12	Negrita
Contenido	Arial	10	Normal
<i>Citas</i>	Arial	10	Cursiva
<i>Xenismos</i>	Arial	10	Cursiva
<i>Figuras</i>	Arial	8	Cursiva
Código	Consolas	10	Normal

Índice

1. Introducción.....	14
1.1. Introducción/Prefacio.....	14
1.2. Descripción/Definición	15
1.3. Objetivos generales	17
1.3.1. Objetivos principales.....	17
1.3.2. Objetivos secundarios	17
1.4. Metodología y proceso de trabajo.....	18
1.5. Planificación.....	19
1.6. Presupuesto	23
1.7. Estructura del resto del documento	24
2. Análisis de mercado	25
2.1. Público objetivo y perfiles de usuario	25
2.2. Competencia y antecedentes.....	28
2.3. Análisis DAFO.....	34
3. Propuesta.....	35
3.1. Definición de objetivos/especificaciones del producto	35
3.2. Estrategia de marketing.....	35
4. Diseño.....	37
4.1. Arquitectura general de la aplicación/sistema/servicio	37
4.2. Arquitectura de la información y diagramas de navegación	41
4.2.1. Arquitectura de la información.....	41
4.2.2. Diagramas de navegación	45
4.3. Diseño gráfico e interfaces	47
4.3.1. Estilos	47
4.3.2. Usabilidad /UX	52
4.4. Lenguajes de programación y APIs utilizados	54
4.4.1. Lenguajes de programación	54
4.4.2. Software.....	56

5.	Implementación	57
5.1.	Requisitos de instalación	57
5.1.1.	Requisitos durante el desarrollo	57
5.1.2.	Requisitos para el cliente	57
5.2.	Instrucciones de instalación	57
5.2.1.	NODE JS	57
5.2.2.	MONGO DB	58
5.2.3.	MONGO DEV TOOLS	59
5.2.4.	BABEL	60
5.2.5.	ARRANCAR SERVIDOR	60
5.2.6.	ARRANCAR VUE	61
5.2.7.	ACCESO Y LOGIN	61
6.	Demostración	62
6.1.	Instrucciones de uso	62
6.2.	Prototipos	62
6.2.1.	Prototipos Lo-Fi	62
6.2.2.	Prototipos Hi-Fi	70
6.3.	Tests	76
6.4.	Ejemplos de uso del producto (o guía de usuario)	78
6.4.1.	LOGIN y LOGOUT	78
6.4.2.	VER PROGRAMACIÓN	78
6.4.3.	CREAR, MODIFICAR y ELIMINAR SESIONES	81
6.4.4.	VER PELÍCULAS	82
6.4.5.	CREAR, MODIFICAR y ELIMINAR PELÍCULAS	84
6.4.6.	VER CARACTERÍSTICAS PELÍCULAS, SESIONES Y ARTISTAS	86
6.4.7.	FILTRAR PELÍCULAS POR SECCIÓN	88
6.4.8.	VER SALAS	89
7.	Conclusiones y líneas de futuro	91
7.1.	Conclusiones	91
7.2.	Líneas de futuro	92
	Bibliografía	93
	Anexos	96

Figuras y tablas

Índice de figuras

<i>Figura 1: Diagrama fases de la metodología en cascada</i>	18
<i>Figura 2: Diagrama de Gantt global</i>	19
<i>Figura 3: Diagrama de Gantt PEC 1 y PEC 2</i>	20
<i>Figura 4: Diagrama de Gantt PEC 3</i>	20
<i>Figura 5: Diagrama de Gantt PEC 4 Backend</i>	21
<i>Figura 6: Diagrama de Gantt PEC 4 y PEC 5 Frontend</i>	21
<i>Figura 7: Diagrama de Gantt PEC 5 Testeo, Iteraciones y Deploy</i>	22
<i>Figura 8: Porcentaje de adultos que en los últimos 12 meses han realizado tareas relacionadas con conocimientos informáticos por grupos de edad y por tarea</i>	26
<i>Figura 9: Tipo de clientes a los que se dirige LENND</i>	28
<i>Figura 10: Funcionalidades que ofrece Digievents</i>	29
<i>Figura 11: Calendario de gestión de agenda de un evento gestionado en Digievents</i>	29
<i>Figura 12: Listado de artistas de un festival de música gestionado en Festival Pro</i>	30
<i>Figura 13: Listado de tareas de un Project Manager de un festival de música gestionado en Festival Pro</i>	31
<i>Figura 14: Factura de un artista de un festival de música gestionado en Festival Pro</i>	31
<i>Figura 15: Calendario de un evento deportivo gestionado en Lennd</i>	32
<i>Figura 16: Lista de invitados de un evento musical gestionado en Lennd</i>	32
<i>Figura 17: Análisis DAFO del proyecto REDFEST App</i>	34
<i>Figura 18: Arquitectura general de la aplicación</i>	37
<i>Figura 19: Diagrama que parte del modelo Festival</i>	42
<i>Figura 20: Diagrama entre los modelos Sede, Sección, Película, Sala, Sesión y Calendario</i>	43
<i>Figura 21: Relación de herencia de Staff</i>	44
<i>Figura 22: Árbol de navegación</i>	46
<i>Figura 23: Logo de la aplicación REDFEST</i>	47
<i>Figura 24: Creación del logo fase 1</i>	47
<i>Figura 25: Creación del logo fase 2</i>	47
<i>Figura 26: Creación del logo fase 3</i>	48
<i>Figura 27: Creación del logo fase 4</i>	48
<i>Figura 28: Creación del logo fase 5</i>	48
<i>Figura 29: Creación del logo fase 6</i>	48
<i>Figura 30: Creación del logo última fase</i>	49
<i>Figura 31: Tipografía Montserrat</i>	49
<i>Figura 32: Tipografía Montserrat en Google Fonts</i>	50
<i>Figura 33: Paleta de colores principales creada a través de Colors</i>	50
<i>Figura 34: Paleta de colores secundarios</i>	51
<i>Figura 35: Paleta de colores terciarios para el texto</i>	51
<i>Figura 36: Ejemplo de botones básicos con diferentes fondos y objetivos</i>	51

<i>Figura 37: Ejemplo de enlace sin entrar y enlace con el ratón encima</i>	51
<i>Figura 38: Proceso de Diseño Centrado en el Usuario</i>	52
<i>Figura 39: Landing de instalación de NodeJS</i>	58
<i>Figura 40: Instalación de Mongo DB Local</i>	58
<i>Figura 41: Instalación de MongoDB Dev Tools</i>	59
<i>Figura 42: Carpeta “dump” copiada en el directorio de las Mogo DevTools</i>	59
<i>Figura 43: Mongorestore para restaurar la base de datos</i>	59
<i>Figura 44: Instalación global de Babel para que transpile el código</i>	60
<i>Figura 45: Comando para arrancar la API REST</i>	60
<i>Figura 46: Comando para arrancar la aplicación</i>	61
<i>Figura 47: Pantalla de login</i>	62
<i>Figura 48: Pantalla de programación sin calendario creado</i>	63
<i>Figura 49: Pantalla de creación de la estructura de la programación</i>	63
<i>Figura 50: Vista estándar de la Programación</i>	64
<i>Figura 51: Vista estándar del listado de películas por secciones</i>	64
<i>Figura 52: Vista para el departamento de Películas del listado de películas por secciones</i>	65
<i>Figura 53: Vista estándar del listado de películas</i>	65
<i>Figura 54: Vista para el departamento de Películas del listado de películas</i>	66
<i>Figura 55: Vista de las sedes del festival</i>	66
<i>Figura 56: Vista para el departamento técnico de las salas de una sede</i>	67
<i>Figura 57: Vista de las características de las salas de una sede</i>	67
<i>Figura 58: Ventana modal con la ficha de una película</i>	68
<i>Figura 59: Ventana modal para el departamento películas con la ficha de una película</i>	68
<i>Figura 60: Ventana modal para el departamento de invitados con la ficha de una película</i>	69
<i>Figura 61: Ventana modal para el departamento técnico con la ficha de una película</i>	69
<i>Figura 62: Ventana modal para el departamento de invitados con la ficha de un artista</i>	70
<i>Figura 63: Pantalla de login</i>	70
<i>Figura 64: Vista estándar de la Programación</i>	71
<i>Figura 65: Vista del listado de películas por secciones</i>	71
<i>Figura 66: Vista para el departamento de películas del listado de películas</i>	72
<i>Figura 67: Vista estandar del listado de películas</i>	72
<i>Figura 68: Vista de las sedes del festival</i>	73
<i>Figura 69: Vista para el departamento técnico de las salas de una sede</i>	73
<i>Figura 70: Ventana modal con la ficha de una película</i>	74
<i>Figura 71: Ventana modal de edición para el departamento películas</i>	74
<i>Figura 72: Ventana modal de edición para el departamento de invitados</i>	75
<i>Figura 73: Ventana modal de edición para el departamento técnico</i>	75
<i>Figura 74: Ventana modal para el departamento de invitados con la ficha de un artista</i>	76
<i>Figura 75: Venta de login a la aplicación</i>	78
<i>Figura 76: Botón de logout</i>	78
<i>Figura 77: Menú de navegación lateral</i>	79
<i>Figura 78: Vista de programación como Programador</i>	80
<i>Figura 79: Iconos para ver, editar y eliminar sesiones</i>	81

<i>Figura 80: Botón para crear una nueva sesión</i>	81
<i>Figura 81: Formulario para crear una nueva sesión</i>	81
<i>Figura 82: Formulario de edición de sesiones</i>	82
<i>Figura 83: Pregunta de confirmación de eliminación de sesiones</i>	82
<i>Figura 84: Vista de películas como admin o departamento película</i>	83
<i>Figura 85: Iconos para ver, editar y eliminar películas</i>	84
<i>Figura 86: Botón para crear una nueva película</i>	84
<i>Figura 87: Formulario de creación de nuevas películas</i>	84
<i>Figura 88: Formulario de edición de películas</i>	85
<i>Figura 89: Pregunta de confirmación de eliminación de películas</i>	85
<i>Figura 90: Ficha datos película</i>	86
<i>Figura 91: Ficha datos sesión</i>	87
<i>Figura 92: Ficha datos artista</i>	87
<i>Figura 93: Formulario de modificación de datos de un artista</i>	88
<i>Figura 94: Pregunta de confirmación de eliminación de un artista</i>	88
<i>Figura 95: Menú desplegable</i>	88
<i>Figura 96: Listado de películas de una sección</i>	89
<i>Figura 97: Menú de sedes</i>	89
<i>Figura 98: Listado de salas con sus características</i>	89
<i>Figura 99: Formulario de edición de las características de una sala</i>	90
<i>Figura 100: Pregunta de confirmación de eliminación de una sala</i>	90
<i>Figura 101: Anexo C: Estructura base de la app</i>	99
<i>Figura 102: Anexo C:Estructura base del backend</i>	99
<i>Figura 103: Anexo C:Estructura base del frontend</i>	100
<i>Figura 104: Anexo C: Index del backend</i>	101
<i>Figura 105: Anexo C:Ejemplo de creación de un modelo en el backend. En este caso, el modelo Película</i>	102
<i>Figura 106: Anexo C: Extracto de un controlador en el backend. En este caso, el controlador de Película con uno de sus métodos</i>	103
<i>Figura 107: Anexo C:Ejemplo de router del backend. En este caso, el router de Película donde se vincula el controlador con el router principal de la API, y se controlan los accesos</i>	103
<i>Figura 108: Anexo C:Archivo router del backend donde se derivan a las rutas correspondientes al recibir una petición en la API</i>	104
<i>Figura 109: Anexo C:Extracto del archivo en el que se codifica, decodifica y chequea el token de autenticación</i>	105
<i>Figura 110: Anexo C: Extracto del archivo en el que se codifica, decodifica y chequea el token de autenticación. En este caso, las funciones de creación del token y de decodificación del token</i>	105
<i>Figura 111: Anexo C: Extracto del archivo en el que se chequean las autorizaciones. En este caso, se verifica si hay token y si el usuario pertenece al departamento de IT</i>	106
<i>Figura 112: Anexo C:Extracto del archivo package.json donde se indican las dependencias y las dependencias de desarrollo</i>	107
<i>Figura 113: Anexo C: Carpeta con los archivos generados al construir el frontend. Estos archivos se copian en la carpeta public de la API que se despliega en la red, en este caso en Heroku</i>	107

Figura 114: Anexo C: Extracto del archivo package.json donde se indican las dependencias y las dependencias de desarrollo. 108

Figura 115: Anexo C: Archivo principal del frontend que indica la conexión con la API e importa todo lo necesario. 109

Figura 116: Anexo C: Extracto de archivo principal de la aplicación donde se escriben los elementos que siempre tienen que estar cargados. 109

Figura 117: Anexo C: Extracto del archivo que maneja las rutas del frontend..... 110

Figura 118: Anexo C: Extracto de la función que controla si se está logueado. 111

Figura 119: Anexo C: Extracto de código que controla el estado de logueo y ejecuta las acciones de guardado del token, de autologin y de desloguearse..... 112

Figura 120: Anexo C: Extracto de código que importa la librería Vuetify para manejar la parte estética de la aplicación..... 113

Figura 121: Anexo C: Extracto de código de creación de uno de los componentes. 114

Figura 122: Anexo C: Diagrama UML completo..... 115

Índice de tablas

Tabla 1: Tabla de planificación de tareas..... 19

Tabla 2: Presupuesto..... 23

Tabla 3: Perfiles 27

Tabla 4: Ventajas y desventajas de otras aplicaciones..... 33

Tabla 5: Tabla de endpoints de la API REST desarrollada 40

Tabla 6: Tabla de modelos..... 41

Tabla 7: Tabla de modelos..... 61

1.Introducción

1.1. Introducción/Prefacio

La idea de este TFG nace hace varios años durante la preproducción del Festival Internacional de Cine de San Sebastián¹.El trabajo que gira en torno al festival comienza un par de meses antes de este, momento en el que se confirma qué películas participarán, qué artistas serán los invitados, qué personal trabajará en esa edición, etc. A partir de ese momento, todos los departamentos trabajan cada uno en su área requiriendo de mucha información de los demás. Por ejemplo, el departamento de películas recibe la información relativa a las películas por parte de las distribuidoras y productoras, como qué personas del equipo artístico asistirán o cuál es el formato de la película. Esta información es importante, por un lado, para el departamento de invitados que gestionará el *hospitality* de los invitados, y por otro para el departamento técnico que gestionará la proyección de la película. Este es solo un ejemplo de los cientos de casos que hay que gestionar rápidamente en menos de dos meses de preproducción.

El problema reside en que estas gestiones, a día de hoy, se realizan a través de *emails* o de tablas de Excel compartidas. Esto, edición tras edición, genera muchos errores humanos e información que se traspapela.

El momento de la preproducción donde más se acusa este problema es un par de semanas antes cuando el programador o curador de contenido, habitualmente una única persona, recoge toda esa información de cada departamento y crea el calendario de proyecciones. Este programador tiene que tener en cuenta variables como a qué sección pertenece la película, en qué soporte va a llegar la copia, la duración, las limitaciones de fechas de los invitados o el contenido de la película, entre otros. Este último factor es el único que tiene que ser supervisado realmente por un programador, ya que es cuestión de criterio y conocimiento saber qué necesita el espectador según el día del festival. El resto de variables que he enumerado, son totalmente controlables por un *software*, no es necesario que, por ejemplo, una persona vaya revisando una por una si la película seleccionada que llegará en 35mm² puede ser proyectada en la Sala 3 que no tiene proyector de ese soporte.

Teniendo en cuenta esta problemática, la aplicación web REDFEST pretende optimizar los procesos de trabajo de la preproducción de un festival de cine.

¹ Festival de San Sebastián. <https://www.sansebastianfestival.com/es/>

² «Película de 35 mm». El 35mm es el formato de película fotográfico en el que se rodaba y se proyectaba antes de aparecer el cine digital
https://es.wikipedia.org/wiki/Pel%C3%ADcula_de_35_mm

1.2. Descripción/Definición

El objetivo de este Trabajo de Final de Grado es el de crear una aplicación web que proporcione herramientas nuevas de trabajo a los festivales de cine con el fin de optimizar sus recursos. Más concretamente, la aplicación web será la red de trabajo interna del festival, donde los departamentos tendrán toda su información y donde podrán acceder a la información que necesitan de los demás departamentos, con el fin de obtener una programación sin fisuras y en un plazo de tiempo muy corto.

Esta aplicación girará en torno a la base de datos compuesta por todas las películas que participarán, y cada usuario podrá ver o editar la información que le corresponda de esta base de datos. De esta manera, la base de datos será única y tendrá toda la información, evitando así problemas habituales como la información duplicada o la inconsistencia entre listados. Esta base de datos estará creada en MongoDB³, que permite trabajar con bases de datos no relacionales que, en lugar de guardar los datos en tablas, los guarda en documentos, y permite trabajar con los registros como objetos JSON.

Para llevar a cabo esta aplicación web se va a utilizar el *full stack* MEVN⁴ (Mongo, Express, Vue, Node), de manera que toda la aplicación será desarrollada en Javascript tanto en el lado del servidor como en el lado del cliente.

En el *backend* se va a utilizar Node⁵ y su *framework* Express⁶, creando una API REST (*Application Programming Interface – Representational State Transfer*)⁷, es decir, una interfaz de programación de aplicaciones de tipo REST. Una API de tipo REST es la que utiliza el protocolo HTTP⁸ para enviar los datos solicitados en formato XML o JSON⁹. A través de la API REST el objetivo es separar el cliente del servidor de manera que cuando se quiera crear una aplicación móvil que trabaje con los mismos datos que esta aplicación web, se pueda acceder fácilmente a esta API REST y obtener todos los datos sin tener que trabajar un *backend* específico.

Por el lado del cliente, se va a trabajar con el *framework* Vue¹⁰ que permite trabajar con componentes, es decir, que habrá código encapsulado en diferentes componentes pudiendo manejar cada componente como requiera la situación. Esta característica permite que la aplicación sea fácilmente

³ «La base de datos líder del mercado para aplicaciones modernas». <https://www.mongodb.com/es>

⁴ «What is MEVN Stack ? - GeeksforGeeks». <https://www.geeksforgeeks.org/what-is-mevn-stack/>

⁵ Node.js, «Acerca». <https://nodejs.org/es/about/>

⁶ «Express - Infraestructura de aplicaciones web Node.js». <https://expressjs.com/es/>

⁷ «¿Qué es una API REST?» <https://tech.tribalyte.eu/blog-que-es-una-api-rest>

⁸ «Generalidades Del Protocolo HTTP - HTTP | MDN».

<https://developer.mozilla.org/es/docs/Web/HTTP/Overview>

⁹ «IBM Docs», 24 de junio de 2021. <https://www.ibm.com/docs/es/baw/20.x?topic=formats-javascript-object-notation-json-format>

¹⁰ «Vue.js»; Pulido, «VueJS Crear Una Aplicación Con Múltiples Layouts.»; Au-Yeung, «Vue. Js 3 by Example»; Camden, «Front-End Development Projects with Vue. Js»; *Vue.Js 3 By Example*.

escalable y además se genere una SPA¹¹ (*Single Page Application*), una aplicación web que tiene una única página que carga unos u otros componentes dependiendo de lo que se solicite. Otra de las características interesantes de Vue es que la aplicación será reactiva, lo que quiere decir que cuando se realicen cambios, esta se actualizará automáticamente sin necesidad de refrescar la página. La reactividad y que se trate de una SPA, agilizará mucho los procesos de los trabajadores haciendo que no pierdan el tiempo buscando y abriendo documentos.

¹¹ «Single-page application». https://es.wikipedia.org/wiki/Single-page_application

1.3. Objetivos generales

1.3.1. Objetivos principales

Objetivos de la aplicación:

- Definir la arquitectura adecuada para la aplicación
- Definir la línea gráfica y la usabilidad acorde al producto y a los usuarios
- Crear un sistema de *login* dando permisos diferentes dependiendo del tipo de usuario.
- Crear una SPA donde tengamos diferentes secciones desde las que se podrá visualizar y modificar la base de datos:
 - Listados que se puedan ordenar por diferentes campos
 - Vistas por película donde se puedan ver los detalles de cada registro
 - Vistas por departamento para que cada usuario pueda modificar los datos que le corresponden exclusivamente
 - Calendario donde se muestre la programación de las películas

Objetivos para el cliente/usuario:

- Optimizar recursos
- Mantener toda la información en un único espacio reduciendo así los errores
- Ser más ágiles en la resolución de problemas ya que un error se solucionará con una única acción, no habrá que solucionar el error en varios sitios
- Menos tiempo de búsqueda de información
- Mejor comunicación entre departamentos

Objetivos personales del autor del TF:

- Entregar una aplicación completamente funcional
- Consolidar los conocimientos aprendidos en todas las áreas del grado (diseño, gestión de proyectos, desarrollo)
- Aprender sobre el *full stack* MEVN y otras herramientas como la gestión de repositorios Git

1.3.2. Objetivos secundarios

Objetivos adicionales que enriquecen el TF.

- Crear un histórico de cada edición del festival
- Automatizar el proceso de programación creando reglas sencillas que generarán una primera versión de calendario, permitiendo su modificación manual a posteriori.
- Crear un tipo de usuario para las productoras y distribuidoras, de manera que puedan introducir toda su información a través de un formulario, y que esta información se vuelque en la base de datos del festival.

1.4. Metodología y proceso de trabajo

Esta aplicación web no se basa en ninguna aplicación ya existente ni utiliza APIs de terceros que aceleren el desarrollo de ciertas funcionalidades. Debido a esto y a que los futuros usuarios tendrán que aprender a trabajar con la aplicación muy rápidamente, se va a desarrollar desde cero, tratando de mantener la usabilidad y los procesos simples.

Teniendo en cuenta el plan de entregas de PECs ya establecido con el que se afronta el TFG, la metodología que se ha seleccionado es la metodología en cascada. El orden en el que están establecidas las PECs permite hacer un plan de tareas con los hitos correspondientes y seguir las fases de la metodología.

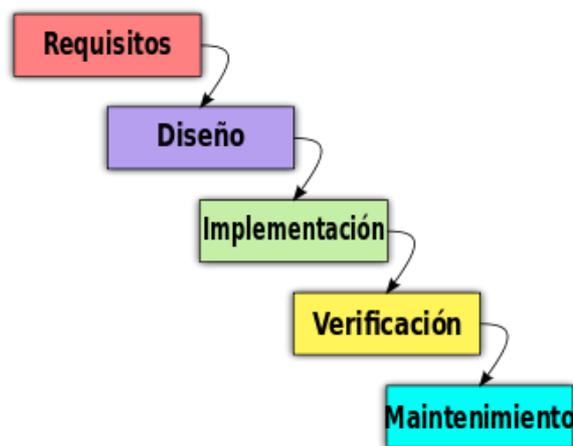


Figura 1: Diagrama fases de la metodología en cascada¹²

A través de esta metodología, las fases sobre las que se va a iterar serán las siguientes:

- Análisis de los requisitos: fase en la que se analizan las necesidades del proyecto.
- Diseño: fase en la que se diseña la aplicación tanto a nivel de arquitectura como desde un punto de vista formal y técnico.
- Implementación: fase de desarrollo del código.
- Testeo: fase de pruebas del código, usabilidad y accesibilidad.

Tras estas cuatro fases se vuelve a iterar desde la fase 1 hasta conseguir un entregable ya funcional que cumpla con los objetivos. Una vez completados los *tests* satisfactoriamente, se despliega la aplicación y solo queda la fase de mantenimiento, en la que se trata de mantener la aplicación estable, sin bugs y actualizada para que siga funcionando correctamente a pesar de los cambios en los navegadores, lenguajes, etc.

¹² «Desarrollo en cascada». https://es.wikipedia.org/wiki/Desarrollo_en_cascada

1.5. Planificación

Hitos / Tareas	Duración	Fecha Inicio	Fecha Final
PEC 1	5 días	15/09/2021	19/09/2021
Briefing	5 días	15/09/2021	19/09/2021
PEC 2	14 días	20/09/2021	03/10/2021
Definir producto	2 días	20/09/2021	21/09/2021
Definir objetivos	2 días	22/09/2021	23/09/2021
Diseño del plan de trabajo	8 días	24/09/2021	01/10/2021
Presentación del plan de trabajo	2 días	2/10/2021	03/10/2021
PEC 3	28 días	04/10/2021	31/10/2021
Análisis de mercado	2 días	04/10/2021	05/10/2021
Propuesta de negocio	2 días	06/10/2021	07/10/2021
Análisis de contenido	6 días	07/10/2021	12/10/2021
Análisis formal	17 días	13/10/2021	29/10/2021
Análisis técnico	5 días	13/10/2021	17/10/2021
PEC 4 y PEC 5	64 días	1/11/2021	03/01/2021
Backend	16 días	1/11/2021	16/11/2021
Frontend	29 días	17/11/2021	15/12/2021
Testeo	7 días	16/12/2021	22/12/2021
Iteraciones	7 días	23/12/2021	29/12/2021
Deploy	5 días	30/12/2021	03/01/2021

Tabla 1: Tabla de planificación de tareas

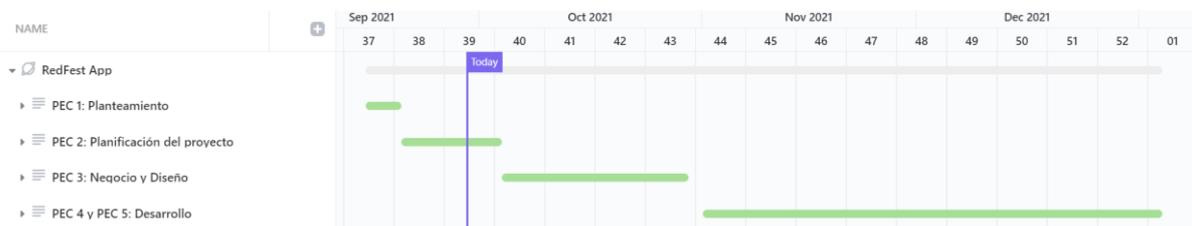


Figura 2: Diagrama de Gantt global

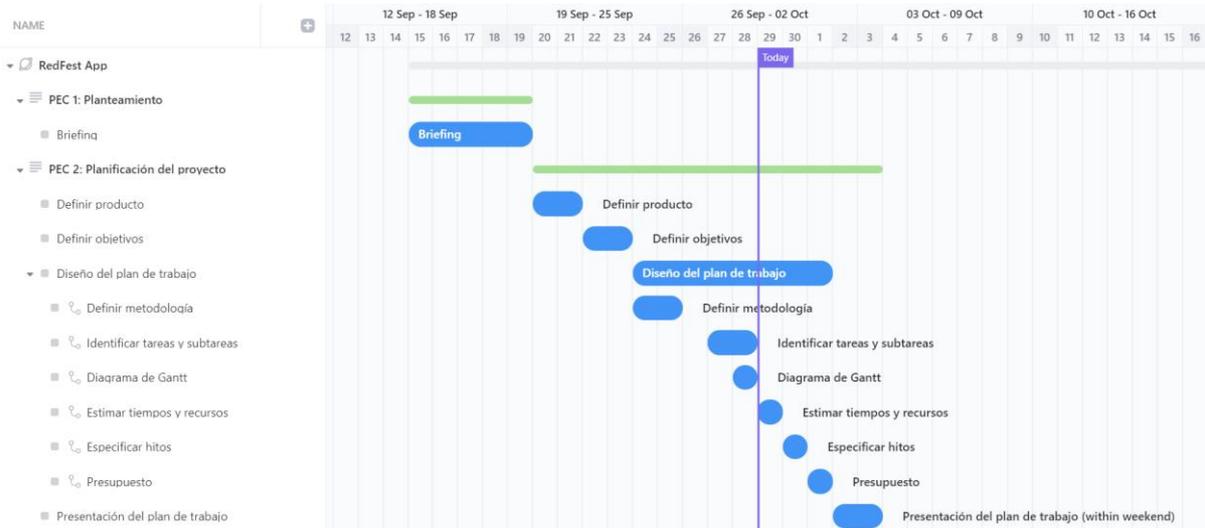


Figura 3: Diagrama de Gantt PEC 1 y PEC 2

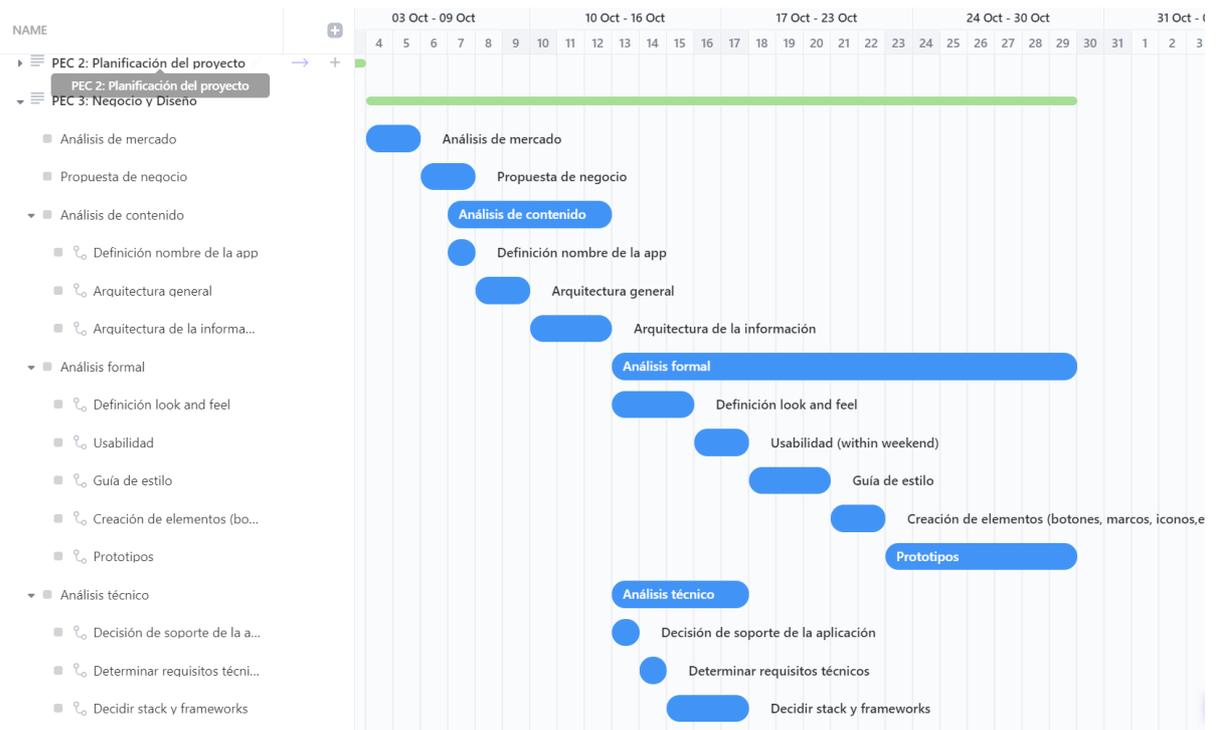


Figura 4: Diagrama de Gantt PEC 3

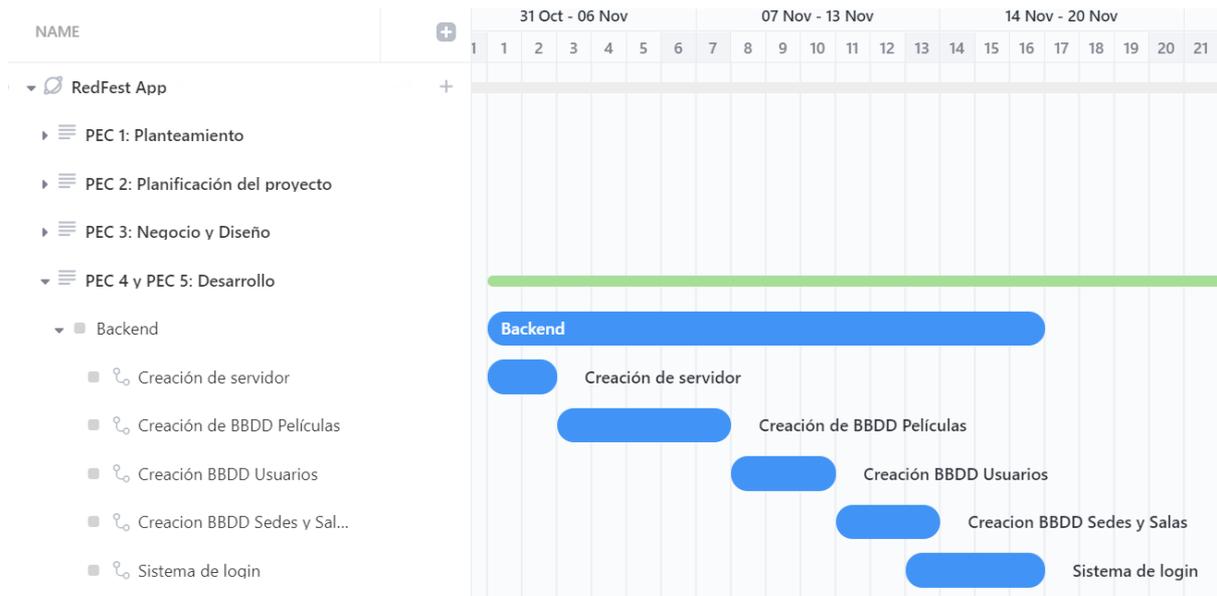


Figura 5: Diagrama de Gantt PEC 4 Backend

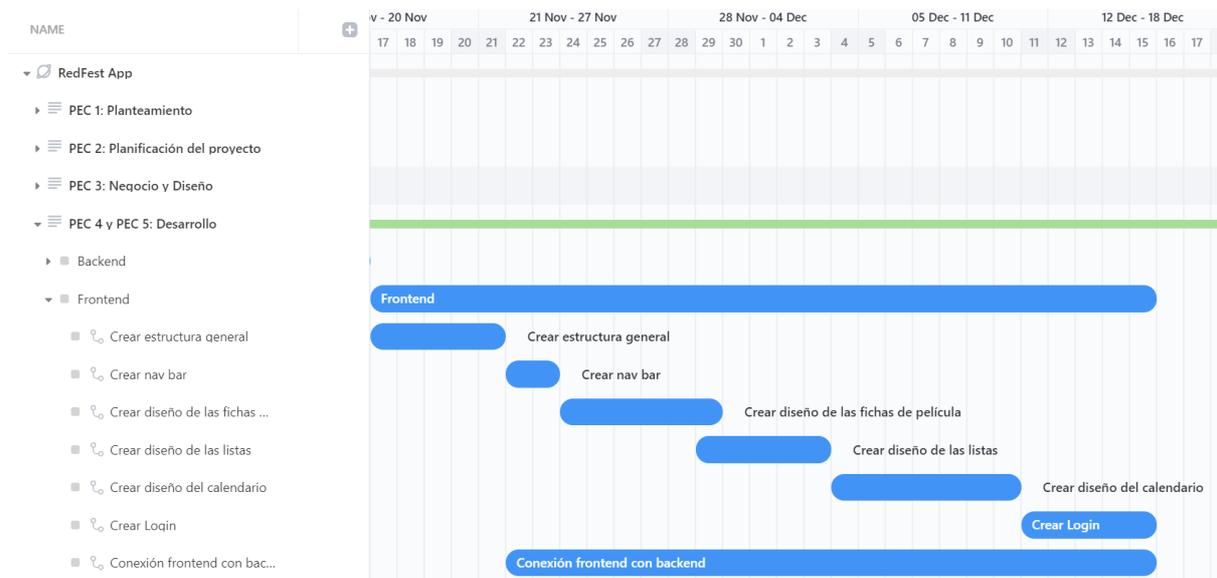


Figura 6: Diagrama de Gantt PEC 4 y PEC 5 Frontend

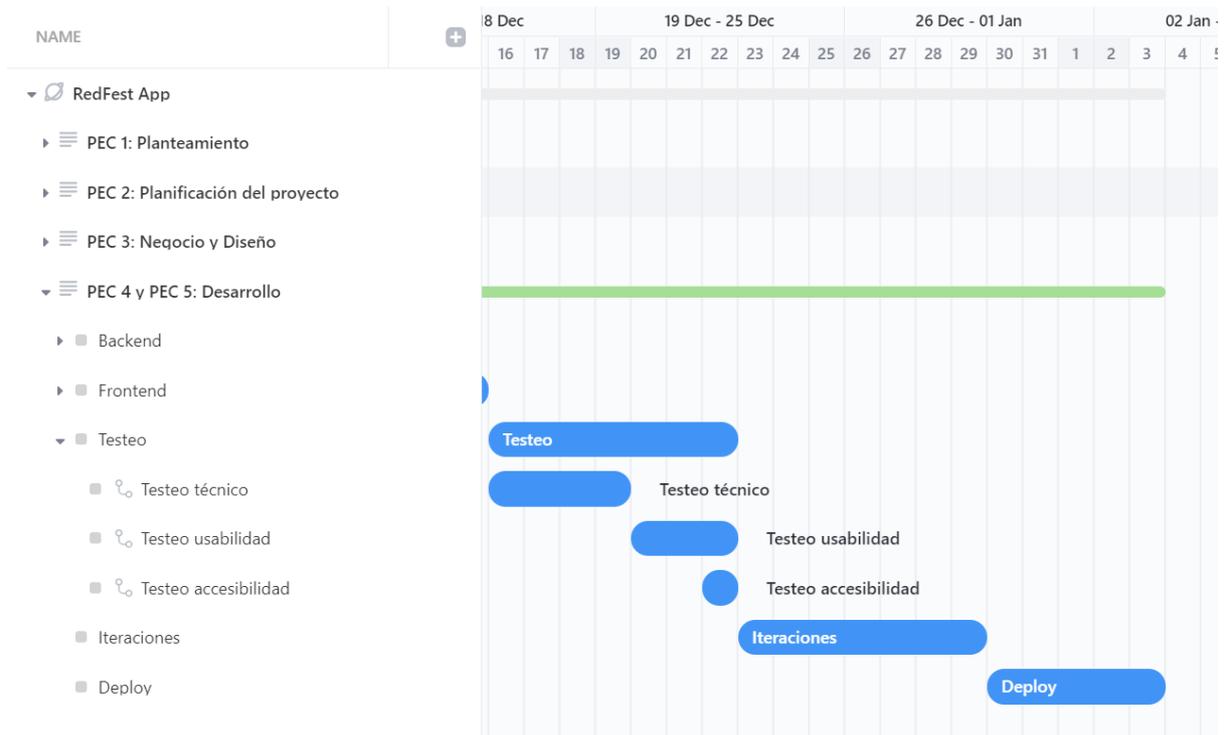


Figura 7: Diagrama de Gantt PEC 5 Testeo, Iteraciones y Deploy

1.6. Presupuesto

Este presupuesto es un presupuesto orientativo de lo que podría costar el desarrollo de esta aplicación teniendo en cuenta que está desarrollada por una única persona y que el tiempo de entrega viene determinado por este mismo hecho, 3-4 meses por todo el proyecto. Como proveedor de este tipo de aplicación, no se presupuesta el análisis del teórico briefing recibido, solo se presupuesta el gasto directo generado y las horas dedicadas a diseñar y desarrollar el producto.

Concepto	Horas	Coste/Hora	Coste
Diseño del producto			3200€
Diseño arquitectura de la aplicación	40	20€/hora	800€
Línea gráfica	40	20€/hora	800€
Diseño de contenidos	80	20€/hora	1600€
Desarrollo del producto			5640€
Backend (API REST)	80	20€/hora	1600€
Frontend	160	20€/hora	3200€
Testing	40	20€/hora	800€
Deploy	24	20€/hora	480€
Licencias de software			350€
Photoshop	3 meses	30€/mes	90€
Illustrator	3 meses	30€/mes	90€
Adobe XD	4 meses	20€/mes	80€
Adobe Stock o similar	3 meses	30€/mes	90€
Mantenimiento (a determinar)			750€
Servicio 24h durante la preproducción y producción del festival	Min. 1 mes	200€/mes	200€
Resto del año	Máx. 11 meses	50€/mes	550€
TOTAL			9.940€

Tabla 2: Presupuesto

1.7. Estructura del resto del documento

Estos son los siguientes puntos de la memoria y una breve descripción del contenido que se puede encontrar dentro:

- **Análisis de mercado:** en este apartado se hace un análisis del público al que está destinada la aplicación, de aplicaciones similares o que buscan soluciones del mismo tipo, y de las debilidades, amenazas, fortalezas y oportunidades del proyecto.
- **Propuesta:** en este apartado se trabajan los objetivos de la aplicación en el contexto de negocio y se presenta un plan de marketing y negocio.
- **Diseño:** en este apartado se crea la estructura de la aplicación tanto a nivel de contenido, como a nivel formal y técnico. Se trata de la fase de toma de decisiones tras haber analizado los dos apartados anteriores.
- **Implementación:** en este apartado se describen los requisitos técnicos y pasos a seguir para poder instalar la aplicación
- **Demostración:** en este apartado además de indicar cómo se utiliza la aplicación con una guía de usuario, se muestran los prototipos y los *tests* utilizados para revisar la usabilidad, el desarrollo y la accesibilidad.
- **Conclusiones y líneas de futuro:** en este último apartado se hace una revisión de en qué punto está la aplicación, qué nuevas funcionalidades se podrían agregar a partir de ahí y cómo se puede mejorar la aplicación y el plan de negocio.

2. Análisis de mercado

2.1. Público objetivo y perfiles de usuario

Los usuarios de esta aplicación web serán trabajadores de festivales de cine ya que será una aplicación, exclusivamente, de uso interno. Este perfil de usuario puede tener entre 18 y 65 años, y habitualmente predomina el rango de edad entre los 25 y los 55 años. Al tratarse de un rango tan amplio, es importante conocer qué conocimientos informáticos tiene cada perfil.

Porcentaje de adultos (16 a 74 años) que en los últimos 12 meses han realizado tareas relacionadas con conocimientos informáticos por sexo, grupos de edad y tipo de tarea
Encuesta sobre Equipamiento y Uso de Tecnologías de Información y Comunicación en los Hogares, Tipo de tareas informáticas, Total, Grupos de edad, 2020

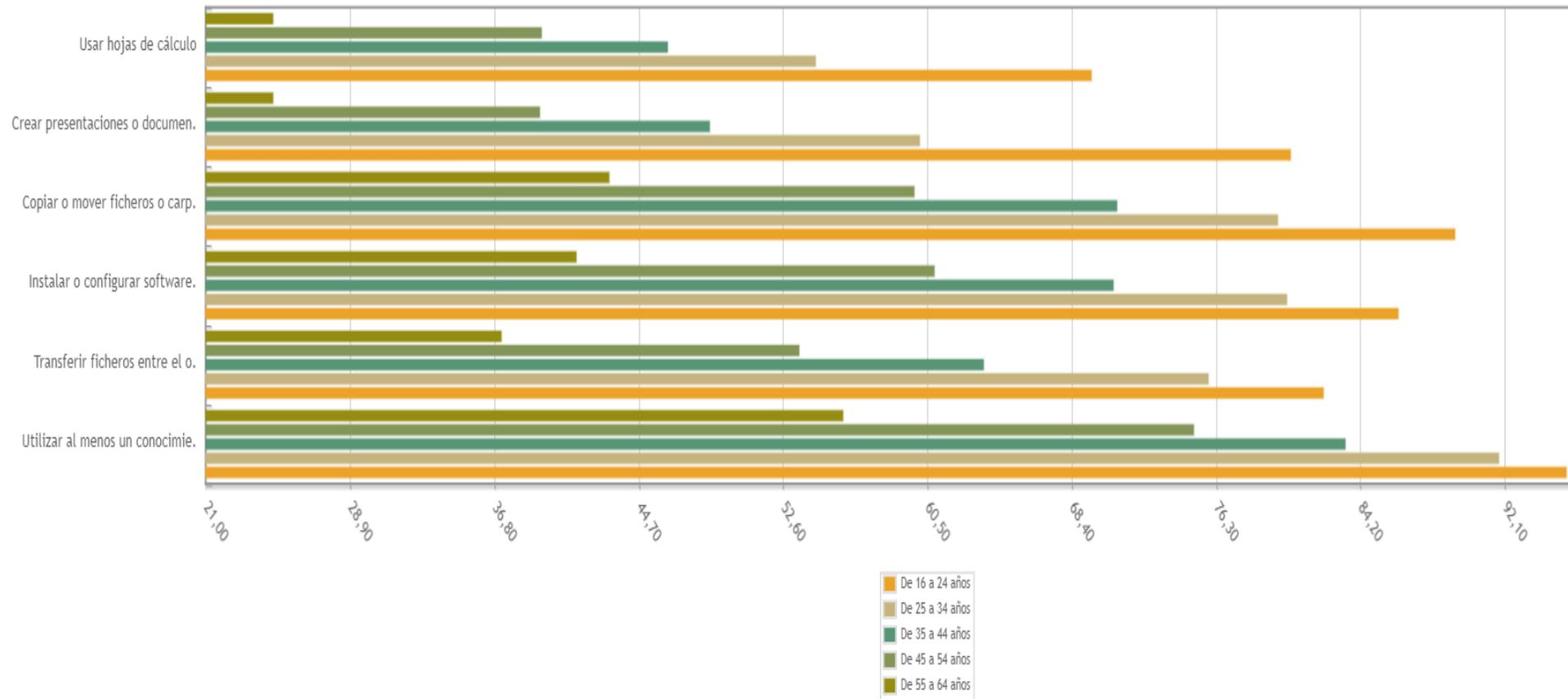


Figura 8: Porcentaje de adultos que en los últimos 12 meses han realizado tareas relacionadas con conocimientos informáticos por grupos de edad y por tarea.¹³

¹³ «Porcentaje de adultos (16 a 74 años) que en los últimos 12 meses han realizado tareas relacionadas con conocimientos informáticos por sexo, grupos de edad y tipo de tarea(46292)». <https://www.ine.es/jaxiT3/Datos.htm?t=46292#!tabs-grafico>

La conclusión a la que se llega tras analizar este estudio es que, al dirigirse a un amplio rango de edades, la aplicación debe mantener lo más simplificada posible su usabilidad. A pesar de esto hay que tener en cuenta que, al tratarse de un tipo de negocio temporal, la mayoría de sus trabajadores son gente joven muy habituada a utilizar aplicaciones y diferentes tipos de *software*. Sin embargo, el sistema debe contemplar a los trabajadores de mayor edad, ya que tienen mayor conocimiento sobre el trabajo que la aplicación debe realizar. Posibles perfiles concurrentes de la aplicación:

Nombre	Edad	Departamento	Tareas	Foto
Jon	25	Técnico	<ul style="list-style-type: none"> ➤ Sacar listados de las películas en diferentes órdenes ➤ Modificar los datos de las películas según se van revisando ➤ Revisar el calendario de proyecciones ➤ Crear y modificar los datos de las salas que se utilizarán en la edición actual 	
Carla	35	Películas	<ul style="list-style-type: none"> ➤ Crear y modificar los registros de las películas ➤ Revisar el calendario de proyecciones ➤ Consultar las características de las salas 	
Eugenia	45	Invitados	<ul style="list-style-type: none"> ➤ Sacar listados de las películas en diferentes órdenes ➤ Modificar los datos de los invitados de cada película ➤ Revisar el calendario de proyecciones ➤ Crear y modificar los registros de los artistas invitados 	
Jose Antonio	60	Programación	<ul style="list-style-type: none"> ➤ Sacar listados de las películas en diferentes órdenes ➤ Revisar ciertos datos sobre los artistas invitados por cada película ➤ Crear y modificar el calendario de proyecciones 	
Mónica	55	Prensa	<ul style="list-style-type: none"> ➤ Sacar listados de las películas en diferentes órdenes ➤ Revisar ciertos datos sobre los artistas invitados de cada película ➤ Revisar el calendario de proyecciones 	
Pablo	50	Producción	<ul style="list-style-type: none"> ➤ Sacar listados de las películas en diferentes órdenes ➤ Revisar ciertos datos sobre los artistas invitados de cada película ➤ Revisar el calendario de proyecciones 	

Tabla 3: Perfiles¹⁴

¹⁴ «Licencia de fotos y vídeos de stock gratuitos - Pexels». <https://www.pexels.com/es-es/license/>

2.2. Competencia y antecedentes

El principal problema para que un equipo de un festival deje los procesos de trabajo obsoletos con los que se organiza es la falta de tiempo de reproducción. Habitualmente, este tiempo ronda entre uno y dos meses donde la principal preocupación es sacar adelante la edición, no hay tiempo para centrarse en los procesos de trabajo.

Esta problemática no es única de los festivales de cine, hay muchos otros tipos de festivales y eventos que trabajan con los mismos tiempos. Es por esta razón que ya se han desarrollado otras soluciones, aunque pertenezcan a otro sector. De hecho, las aplicaciones que se analizan a continuación tienen muchas más funcionalidades que REDFEST, pero son mucho más generalistas, como se indica en esta imagen de abajo perteneciente a la aplicación LENND¹⁵. REDFEST pretende ser una solución directa a un problema concreto, mientras todas estas aplicaciones pretenden servir como soluciones a muchas casuísticas diferentes.

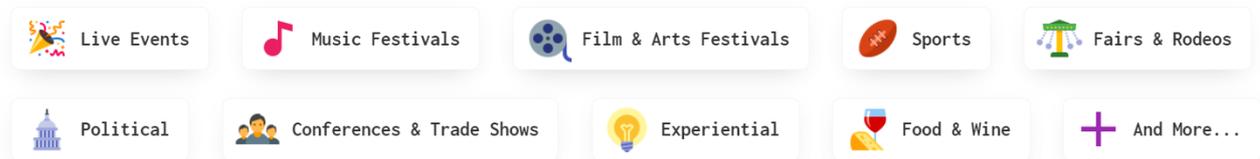


Figura 9: Tipo de clientes a los que se dirige LENND

Las tres principales aplicaciones que ofrecen este tipo de soluciones son:

- Digivents¹⁶: plataforma de gestión de eventos, centrada especialmente en la conexión entre ponentes y asistentes

¹⁵ «Lennd - Live & Hybrid Event Operations Platform». <https://www.lennd.com/>

¹⁶ «Registro, check-in y app para eventos y ferias». <https://www.digivents.com/es/>



Figura 10: Funcionalidades que ofrece Digivents



Figura 11: Calendario de gestión de agenda de un evento gestionado en Digivents

- Festival Pro¹⁷: plataforma de gestión de festivales de todo tipo, especialmente de música.

¹⁷ FestivalPro, «Festival Pro». <https://www.festivalpro.com/index.html>

The screenshot displays the 'ARTISTS' section of the Festival Pro application. A search dropdown menu is active, showing options: 'View All Artists', 'View All Confirmed', 'Add New', 'Advancing Returns', 'Search', and 'Show Festival Calendar'. The main table lists 12 artists with the following columns: Artist, Artist Confirmed, Type of Act, Stage, Performance Date and On Stage Time, Actions, and Shortcuts. At the bottom, there are buttons for 'Add Correspondence', 'Update', 'Export', 'SMS', and 'Bulk Mail', along with pagination for 'Results 1 - 20 of 64'.

Artist	Artist Confirmed	Type of Act	Stage	Performance Date and On Stage Time	Actions	Shortcuts
201 DANCE COMPANY Andrea Walker +7475863896 (M) andrea@201dance.dance more...	Yes	Theater	Theatre Stage;Comedy Stage	15/07/2018 01:00	[Icons]	Schedule
A NEW CREW Oliver Maw olily@subgiant.co.uk more...		Dj	Third Stage	14/07/2018 01:00	[Icons]	Schedule
BARNS COURTNEY Barn Courtney 578-5687-895 (M) barns@softmail.com more...	Yes	Acoustic / Duo	Acoustic Stage;Theatre Stage	13/07/2018 21:00	[Icons]	Schedule
BLACK HONEY Marvin Holmes 548-4257-962 (M) marvin@blackhoney.co.uk more...	Yes	Full Live Act	Second Stage;Dance Stage	13/07/2018 17:00	[Icons]	Schedule
BOMBAY BICYCLE CLUB Neil Grange 07584648374 (M) 07584648374 (H) neilgrange2@festivalpro.co.uk more...	Yes	Full Live Act	TBC		[Icons]	Schedule
BRASSTRACKS Matt Medved 987-965-965 (M) matt@brasstracks.co more...	Yes	Dj	Dance Stage;Third Stage	14/07/2018 02:00	[Icons]	Schedule
CLEAN CUT KID Evelyn Halls +4479658848 (M) evelyn@cck.co.uk more...	Yes	Full Live Act	Third Stage;Dance Stage	13/07/2018 20:00	[Icons]	Schedule
DECLAN MCKENNA Declan McKenna 073365968 (M) dec_mac@gmail.com more...	Yes	Acoustic / Duo	Acoustic Stage;Main Stage	23/08/2019 23:00	[Icons]	Schedule
DIGITALISM Nick Reilly 07956473536 (M) 07956473536 (H) nickreilly@festivalpro.co.uk more...	Yes	Dj	Second Stage;Third Stage	25/08/2019 19:00	[Icons]	Schedule
DJ SEINFELD DJ Seinfeld +568974168245 (M) djseinfeld@google.va more...	Yes	Dj	Dance Stage	15/07/2018 21:00	[Icons]	Schedule

Figura 12: Listado de artistas de un festival de música gestionado en Festival Pro

PROJECT MANAGER - PRODUCTION TASKS -

Search Filter

[Add Task](#)

Tools	#	Active	New	Assigned	Unassigned	Open	WIP/Waiting	Priority	Assigned	Status	Created	Delivery Date	Severity
	1975	<input checked="" type="checkbox"/>	<input type="checkbox"/>	High	Example Fest	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00 Σ Time 0h 0m Δ Estimate 11h 0m	Serious				
	1957	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Medium	Demo System	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00 Σ Time 0h 0m Δ Estimate 10h 30m	Major				
	1965	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Medium	Demo System	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00 Σ Time 0h 0m Δ Estimate 10h 30m	Major				
	1967	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Medium	Demo System	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00	Normal				
	1969	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Medium	None	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00	Normal				
	1971	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Medium	Demo System	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00 Σ Time 0h 0m Δ Estimate 10h 30m	Major				
	1979	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Medium	Example Fest	New	Sun 27/Jan 16:00	Tue 19/Feb 16:00 Σ Time 0h 0m Δ Estimate 10h 30m	Normal				

Totals per user for assigned tasks only

Show All	Active	New	Open	In Progress	Waiting	In QA	Complete	Complete confirmed	On Hold	Addition	Invalid	Duplicate	Total	Total Time	Estimated Set %	Delivery Set %
Unassigned	0	5	0	0	0	0	0	0	0	0	0	0	0		100%	100%
Totals for all tasks assigned and unassigned																
	0	5	0	0	0	0	0	0	0	0	0	0	0	0h 0m	100%	100%

[CSV](#) [Email](#)

General Terms and Conditions

Figura 13: Listado de tareas de un Project Manager de un festival de música gestionado en Festival Pro

FESTIVALPRO Logged In

examplefest

ARTIST ADVANCE

Agreed Billing for:
SUBGIANT at ExampleFest 2019

Artist Name: **SUBGIANT**
to perform a 45,45 minute set (note: this is subject to change)
This constitutes an agreement detailing the following:

Performance Fee: **£1000**

Date of Performance: **23/08/2019 19:00,24/08/2019 20:00**
Please Note: If onstage time is past midnight, date shown will be the following day.

Stage: **Second Stage,Dance Stage**

Stage Time & Set Length: **(45,45 Minutes)**
PLEASE NOTE: This is Subject to change

Exclusivity: **Non-Exclusive**

Artist Ticket Allocation: **4**

Guest Ticket Allocation: **3**

Please fill in all fields below

Figura 14: Factura de un artista de un festival de música gestionado en Festival Pro

- Lennd: plataforma de gestión de festivales de todo tipo, similar a Festival Pro, creada para que todo lo relacionado con un festival pase por la aplicación.

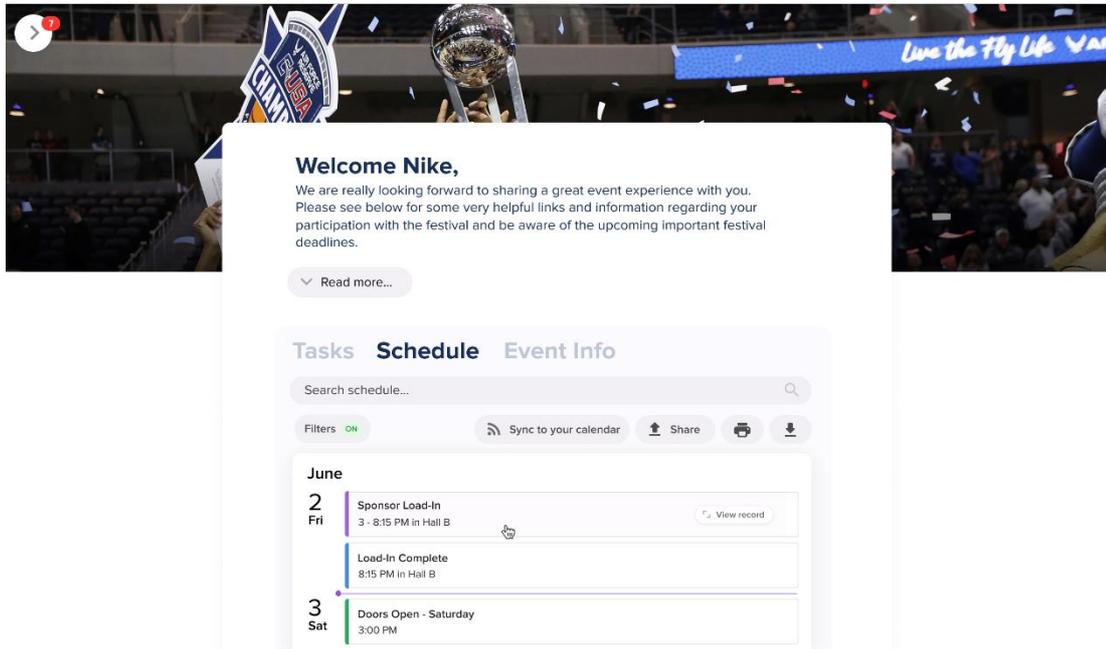


Figura 15: Calendario de un evento deportivo gestionado en Lennd

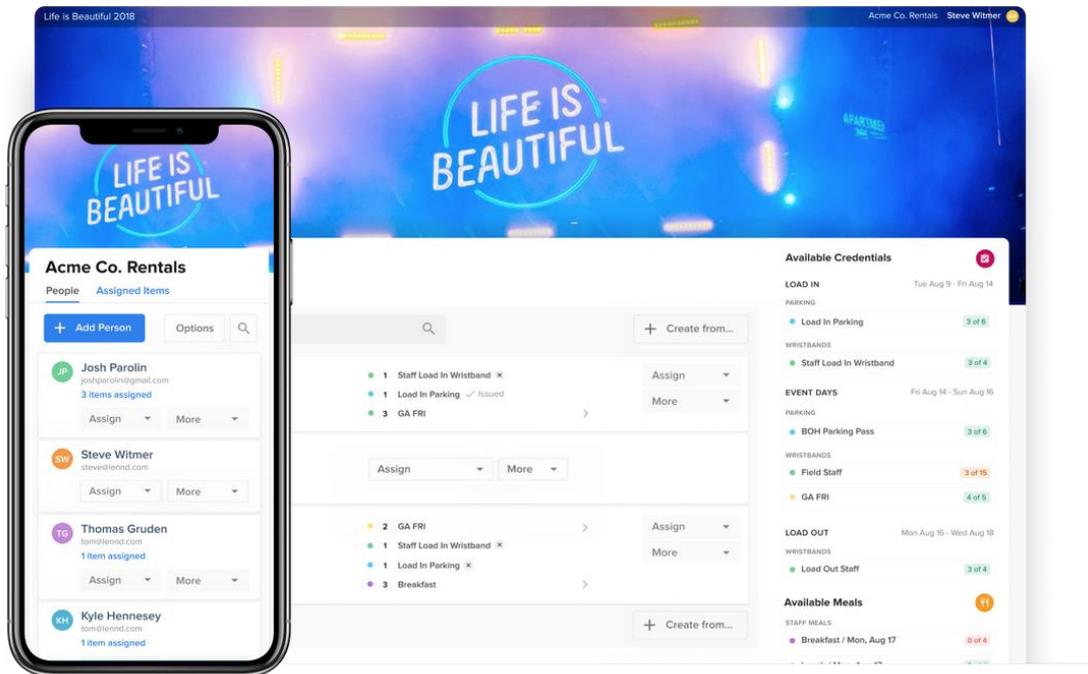


Figura 16: Lista de invitados de un evento musical gestionado en Lennd

VENTAJAS	DESVENTAJAS
<ul style="list-style-type: none"> ○ Desarrollo ya terminado ○ Aplicación web y móvil que permite acceder a ella desde cualquier sitio ○ Abarcan todos los detalles del festival ○ Aúnan muchas tecnologías que permiten trabajar con códigos QR, gestión de entradas, facturación, compras, mailing, acreditaciones, etc. 	<ul style="list-style-type: none"> ○ No solucionan el problema concreto de un festival de cine ○ El festival tendría que adaptarse a lo que la aplicación ofrece y no al revés ○ El cliente paga por muchas funcionalidades que no utilizará ○ Exceso de opciones que dificultarán la configuración rápida que requiere un festival de cine. ○ Faltan detalles concretos para poder desarrollar la automatización de calendario.

Tabla 4: Ventajas y desventajas de otras aplicaciones

En conclusión, hay bastantes herramientas que pueden ser una gran solución para productores de eventos y festivales de cualquier tipo. El principal inconveniente es que, al intentar abarcar tantos sectores y así crear un negocio más amplio, no se presta el servicio preciso necesario, lo que genera que los festivales de cine no consuman este tipo de aplicaciones.

Esta falta de especificidad es lo que da valor y sentido a una aplicación sencilla como REDFEST. Es una buena solución para un tipo de cliente como un festival de cine, que busca soluciones muy concretas en un periodo muy corto de tiempo.

2.3. Análisis DAFO

Este es el análisis de las principales debilidades, amenazas, fortalezas y oportunidades de este proyecto.

<p style="text-align: center;">Debilidades</p> <ul style="list-style-type: none"> ★ Pocos clientes a los que llegar ★ Soluciones limitadas ★ Obligación de ofrecer atención 24h durante un tiempo 	<p style="text-align: center;">Amenazas</p> <ul style="list-style-type: none"> ★ Aplicaciones más completas ★ Rechazo por parte de los clientes potenciales por nuevas aplicaciones
<p style="text-align: center;">Fortalezas</p> <ul style="list-style-type: none"> ★ Cliente específico ★ Usabilidad sencilla ★ Solución directa a un problema concreto 	<p style="text-align: center;">Oportunidades</p> <ul style="list-style-type: none"> ★ Posibilidad de ampliar la aplicación trabajando junto al cliente ★ Ser la primera aplicación que resuelve este problema sin abrumar con más funcionalidades ★ Personalizar funcionalidades según el cliente para que no se tengan que adaptar a la app si no al revés.

Figura 17: Análisis DAFO del proyecto REDFEST App

3.Propuesta

3.1. Definición de objetivos/especificaciones del producto

El objetivo principal de esta aplicación es lograr optimizar el tiempo y recursos de cualquier equipo de trabajo de un festival de cine. Debido a la idiosincrasia de un festival de cine no hay ninguno en todo el territorio nacional que trabaje con un *software* como este. Es por ello por lo que REDFEST pretende solucionar los problemas propios de un festival en el tiempo requerido, adaptando el producto a estas necesidades específicas.

Las características y especificaciones que tiene la aplicación para cumplir con estos objetivos son:

- Sistema de *login* dando permisos diferentes dependiendo del tipo de usuario.
- Listados que se puedan ordenar por diferentes campos
- Vistas por película donde se puedan ver los detalles de cada registro
- Vistas por departamento para que cada usuario pueda modificar los datos que le corresponden exclusivamente
- Calendario donde se muestre la programación de las películas

Todas estas especificaciones, la aplicación tiene que cubrirlas de la manera más simplificada posible debido a que los equipos de trabajo tienen muy poco tiempo de adaptación y aprendizaje. Es por esto por lo que la fortaleza de la aplicación no reside en la cantidad de funcionalidades sino en la usabilidad y la optimización.

La aplicación tendrá como plataforma el navegador web, ya que el trabajo que se realiza en ella no suele ejecutarse en otros dispositivos. Este es otro de los puntos en el que el desarrollo del proyecto se centra en solucionar los problemas reales, no en crear más funcionalidades o soluciones de las necesarias.

3.2. Estrategia de marketing

En cuanto al *branding*, al tratarse de un producto con un cliente tan específico, el logo de la aplicación está directamente relacionado con la industria a la que pertenece. Este es el logo final de la aplicación:



Este logo pretende ser un reclamo para el cliente potencial. El uso del dibujo de un fotograma es una declaración de intenciones en la que se muestra claramente que este producto se trata de un producto exclusivo para la industria del cine. Una de las inquietudes o reticencias de los equipos de dirección de los festivales de cine es que no existe ninguna empresa de desarrollo que realmente sepa cuál es su problemática. Es por esto por lo que la imagen de la aplicación muestra claramente que el objetivo de esta solución tecnológica es servir de herramienta exclusivamente a festivales de cine.

Teniendo en cuenta esta situación, la estrategia a la que se opta es a la presentación personalizada a cada uno de los principales festivales de cine del territorio español. Para ellos se preparará una demo con ciertas características intrínsecas de cada festival como las salas o secciones particulares, para que el cliente vea que es una aplicación ad hoc a su proyecto.

Esta es la lista de los principales objetivos en este primer proceso de venta de la aplicación:

- Festival Internacional de Cine de San Sebastián
- Festival Europeo de Cine de Sevilla
- Festival Internacional de Las Palmas
- SEMINCI (Festival Internacional de Cine de Valladolid)
- Festival de Cine Español de Málaga
- Festival de Cine Internacional de Sitges
- Festival de Cine de Gijón

4. Diseño

4.1. Arquitectura general de la aplicación/sistema/servicio

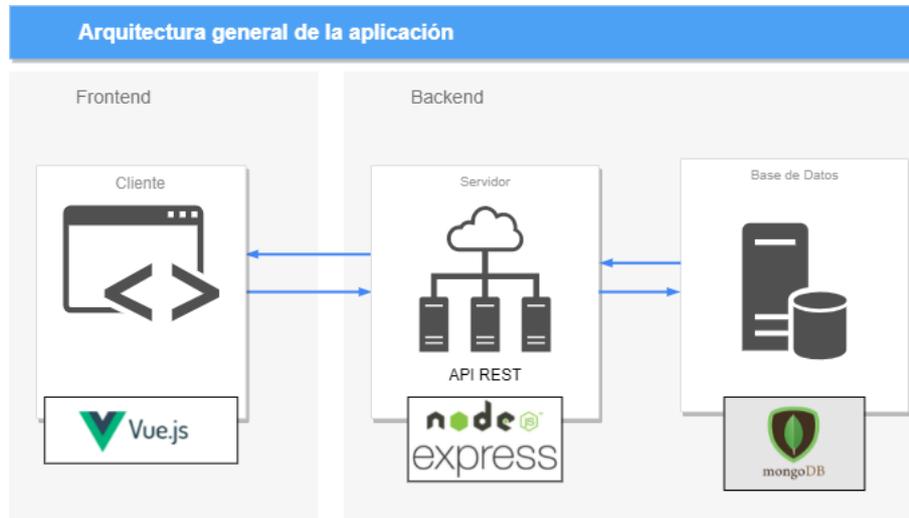


Figura 18: Arquitectura general de la aplicación

- *Frontend*: el *frontend* de la aplicación, es decir, el lado del cliente se va a realizar con el *framework* Vue. Con este *framework* la aplicación podrá mostrar todas sus características a través de una SPA, de manera que el contenido se cargará automática y asíncronamente dependiendo de la acción del usuario, sin necesidad de recargar la página constantemente.
- *Backend*: el *backend* de la aplicación donde se reciben las solicitudes del cliente será una API REST.

Una API REST es el intermediario entre el usuario y la base de datos, es decir la interfaz de programación que se encarga de las solicitudes y devuelve las respuestas en formato JSON o XML. Una API es REST porque se basa en la arquitectura RESTful, que se define por las siguientes características:

- El servidor se encarga de manejar los datos mientras que el cliente se encarga de manejar las interacciones del usuario.
- Las peticiones recibidas por el servidor son cacheables
- Cada interacción está asociada a una dirección única "URI"
- Cada petición recibida en el servidor es independiente y tiene toda la información necesaria para procesarla
- El servidor puede tener varias capas.

Los métodos que se utilizan en una API REST para comunicarse entre el cliente y el servidor son:

- GET: devuelve un objeto
- POST: genera un nuevo dato u objeto
- PUT: crea o modifica un nuevo dato u objeto
- PATCH: actualiza parcialmente datos u objetos
- DELETE: eliminar un objeto

Para gestionar la base de datos se ha seleccionado MongoDB que es un sistema de bases de datos no SQL, es decir, no relacional. Este sistema trabaja con objetos y colecciones de dichos objetos en lugar de con registros interrelacionados. Esto nos permite que los modelos con los que trabajemos sean más flexibles.

Esta API REST ofrece los siguientes *endpoints*:

End point	¿Qué hace?	¿Quién?	¿Qué devuelve?
POST api/staff/login	Registro para entrar al sistema	Usuarios registrados	Éxito (200): Staff y Jason Web Token Error (404): error de login Error (500): error de sistema
POST api/staff	Añade un usuario	Admin	Éxito (200):Registro de Staff Error (500): error de sistema
GET api/staff/:id	Consulta un usuario concreto	Admin	Éxito (200): Datos de un usuario Error (404): error de consulta Error (500): error de sistema
GET api/staff	Consulta todos los usuarios	Admin	Éxito (200):Datos de todos los usuarios Error (500): error de sistema
PUT api/staff/:id	Modifica datos de un usuario	Admin	Éxito (200):Datos modificados de un usuario Error (500): error de sistema
DELETE api/staff/:id	Elimina un usuario	Admin	Éxito (200):Usuario eliminado Error (500): error de sistema
POST api/artista	Añade un artista	Admin e Invitados	Éxito (200):Registro de Artista Error (500): error de sistema
GET api/artista/:id	Consulta un artista concreto	Admin e Invitados	Éxito (200):Datos de un artista Error (404): error de consulta Error (500): error de sistema
GET api/artista	Consulta todos los artistas	Admin e Invitados	Éxito (200):Datos de todos los artistas Error (500): error de sistema
PUT api/artista/:id	Modifica datos de un artista	Admin e Invitados	Éxito (200):Datos modificados de un artista Error (500): error de sistema
DELETE api/artista/:id	Elimina un artista	Admin e Invitados	Éxito (200):Artista eliminado Error (500): error de sistema
POST api/calendario	Genera un calendario	Admin y Programador	Éxito (200):Calendario Error (500): error de sistema

GET api/calendario/:id	Consulta calendario	Admin y Programador	Éxito (200):Datos del calendario Error (404): error de consulta Error (500): error de sistema
PUT api/calendario/:id	Modifica datos del calendario	Admin y Programador	Éxito (200):Datos modificados del calendario Error (500): error de sistema
DELETE api/calendario/:id	Elimina calendario	Admin y Programador	Éxito (200):Calendario eliminado Error (500): error de sistema
POST api/edicion/	Añade una edición	Admin	Éxito (200):Registro de edición Error (500): error de sistema
GET api/edicion/:id	Consulta datos de una edición	Admin	Éxito (200):Datos de una edición Error (404): error de consulta Error (500): error de sistema
GET api/edicion	Consulta todas las ediciones	Todos	Éxito (200):Datos de todas las ediciones Error (500): error de sistema
PUT api/edicion/:id	Modifica datos de una edición	Admin	Éxito (200):Datos modificados de una edición Error (500): error de sistema
DELETE api/edicion/:id	Elimina una edición	Admin	Éxito (200):Edición eliminada Error (500): error de sistema
POST api/festival	Añade un festival	Admin	Éxito (200):Registro de festival Error (500): error de sistema
GET api/festival/:id	Consulta datos de un festival	Admin	Éxito (200):Datos de un festival Error (404): error de consulta Error (500): error de sistema
GET api/festival	Consulta todos los festivales	Admin	Éxito (200):Datos de todos los festivales Error (500): error de sistema
PUT api/festival/:id	Modificar datos del festival	Admin	Éxito (200):Datos modificados del festival Error (500): error de sistema
DELETE api/festival/:id	Eliminar un festival	Admin	Éxito (200):Festival eliminado Error (500): error de sistema
POST api/pelicula	Añade una película	Admin y Películas	Éxito (200):Registro de película Error (500): error de sistema
GET api/pelicula/:id	Consulta datos de una película	Todos	Éxito (200):Datos de una película Error (404): error de consulta Error (500): error de sistema
GET api/película	Consulta todas las películas	Todos	Éxito (200):Datos de todas las películas Error (500): error de sistema
PUT api/pelicula/:id	Modifica datos de una película	Todos	Éxito (200):Datos modificados de una película Error (500): error de sistema
DELETE api/pelicula/:id	Eliminar una película	Admin y Películas	Éxito (200):Película eliminada Error (500): error de sistema
POST api/sala	Añade una sala	Admin y Técnica	Éxito (200):Registro de sala Error (500): error de sistema
GET api/sala/:id	Consulta datos de una sala	Todos	Éxito (200):Datos de una sala Error (404): error de consulta Error (500): error de sistema
GET api/sala	Consulta todas las salas	Todos	Éxito (200):Datos de todas las salas Error (500): error de sistema
PUT api/sala/:id	Modifica datos de una sala	Técnica y Admin	Éxito (200):Datos modificados de una sala

			Error (500): error de sistema
DELETE api/sala/:id	Eliminar una sala	Admin y Técnica	Éxito (200):Sala eliminada Error (500): error de sistema
POST api/seccion	Añade una sección	Admin	Éxito (200):Registro de sección Error (500): error de sistema
GET api/seccion/:id	Consulta datos de una sección	Todos	Éxito (200):Datos de una sección Error (404): error de consulta Error (500): error de sistema
GET api/seccion	Consulta todas las secciones	Todos	Éxito (200):Datos de todas las secciones Error (500): error de sistema
PUT api/seccion/:id	Modifica datos de una sección	Admin y Películas	Éxito (200):Datos modificados de una sección Error (500): error de sistema
DELETE api/seccion/:id	Eliminar una sección	Admin	Éxito (200):Sección eliminada Error (500): error de sistema
POST api/sede	Añade una sede	Admin	Éxito (200):Registro de sedes Error (500): error de sistema
GET api/sede/:id	Consulta datos de una sede	Todos	Éxito (200):Datos de una sede Error (404): error de consulta Error (500): error de sistema
GET api/sede	Consulta todas las sedes	Todos	Éxito (200):Datos de todas las sedes Error (500): error de sistema
PUT api/sede/:id	Modifica datos de una sede	Admin	Éxito (200):Datos modificados de una sede Error (500): error de sistema
DELETE api/sede/:id	Eliminar una sede	Admin	Éxito (200):Sede eliminada Error (500): error de sistema
POST api/sesion	Añade una sesión	Admin y Programador	Éxito (200):Registro de sesión Error (500): error de sistema
GET api/sesion/:id	Consulta datos de una sesión	Todos	Éxito (200):Datos de una sesión Error (404): error de consulta Error (500): error de sistema
GET api/sesion	Consulta todas las sesiones	Todos	Éxito (200):Datos de todas las sesiones Error (500): error de sistema
PUT api/sesion/:id	Modifica datos de una sesión	Admin y Programador	Éxito (200):Datos modificados de una sesión Error (500): error de sistema
DELETE api/sesion/:id	Eliminar una sesión	Admin y Programador	Éxito (200):Sesión eliminada Error (500): error de sistema

Tabla 5: Tabla de endpoints de la API REST desarrollada

4.2. Arquitectura de la información y diagramas de navegación

4.2.1. Arquitectura de la información

Los modelos detectados que forman parte de la aplicación son los siguientes:

Modelo	Descripción
Festival	{ <u>NOMBRE</u> , LOGO , EDICIONES} donde {EDICIONES} referencia Edicion
Edicion	{ <u>ID</u> , NUMEDICION,SECCIONES,SEDES,STAFF} donde {SECCIONES} referencia Seccion, {SEDES} referencia Sede Y {STAFF} referencia Staff
Sede	{ <u>ID</u> , NOMBRE,SALAS, DIRECCION} donde {SALAS} referencia Sala
Sala	{ <u>ID</u> ,NOMBRE,SEDE,SOPORTES,SONIDOS,SUBTELECTR,AFORO} donde {SEDE} referencia Sede
Sección	{ <u>ID</u> ,NOMBRE,IMAGEN,PELICULAS} donde {PELICULAS} referencia Pelicula
Película	{ <u>ID</u> ,TITULOORIGINAL,TITULOSSECUNDARIO,DURACION,IDIOMA,SUBTITULOS, SOPORTE, RATIO,SONIDO,DIRECTOR,ACTORES,SECCION,NUMPASES,NUMCOPIAS,OBS ERVACIONES} donde {DIRECTOR} Y {ACTORES} referencia Artista y {SECCION} referencia Seccion
Artista	{ <u>ID</u> ,NOMBRE,APELLIDOS,PELICULA,ASISTE,CHECKIN,CHECKOUT,IDIOMA} Donde {PELICULA} referencia Pelicula
Staff	{ <u>ID</u> ,NOMBRE,APELLIDOS,EMAIL,PASSWORD}

Tabla 6: Tabla de modelos

La relación entre estos modelos se indica en el siguiente diagrama UML, dividido por fragmentos para una mejor visualización de este.

Como se puede observar en la figura 19, cada festival se relaciona únicamente con las ediciones de dicho festival. Y son las ediciones, cada una de ellas, las que se relacionan con las sedes, las secciones, el calendario y el staff. Las sedes es la única clase con la que no tiene una relación de composición porque estas son independientes a la edición del festival, simplemente unas sedes u otras participan en una edición o no, pero fuera del festival estas sedes siguen existiendo.

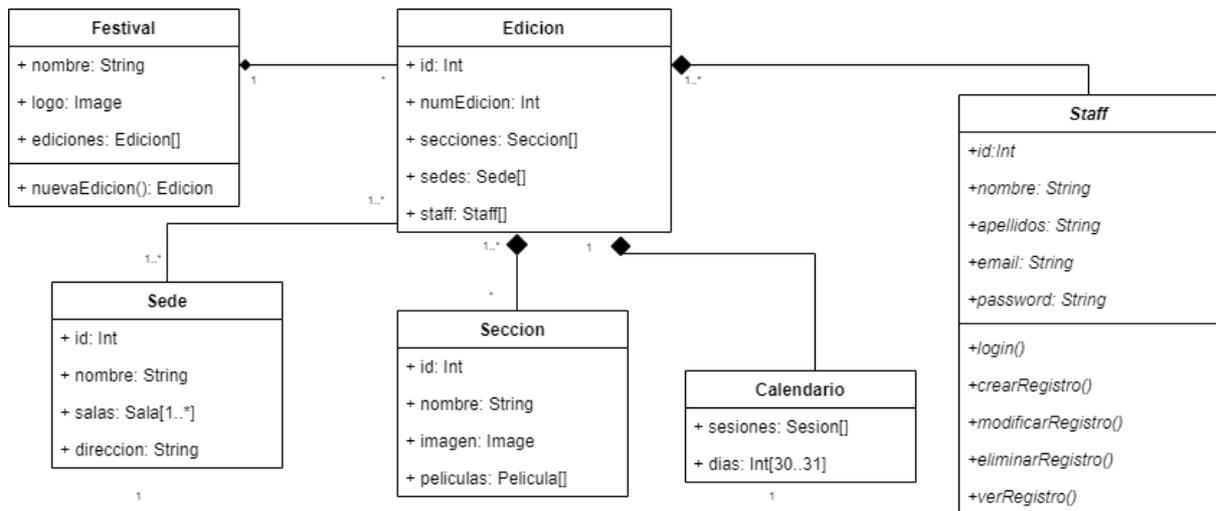


Figura 19: Diagrama que parte del modelo Festival

Las sedes contienen una o varias salas de cine. Las secciones están formadas por películas y se asocian a través de una relación de agregación ya que las películas existen por sí mismas, pertenezcan a una sección o no. En cuanto al calendario, está formado por todas las sesiones que se crean a través de la relación entre las salas y las películas.

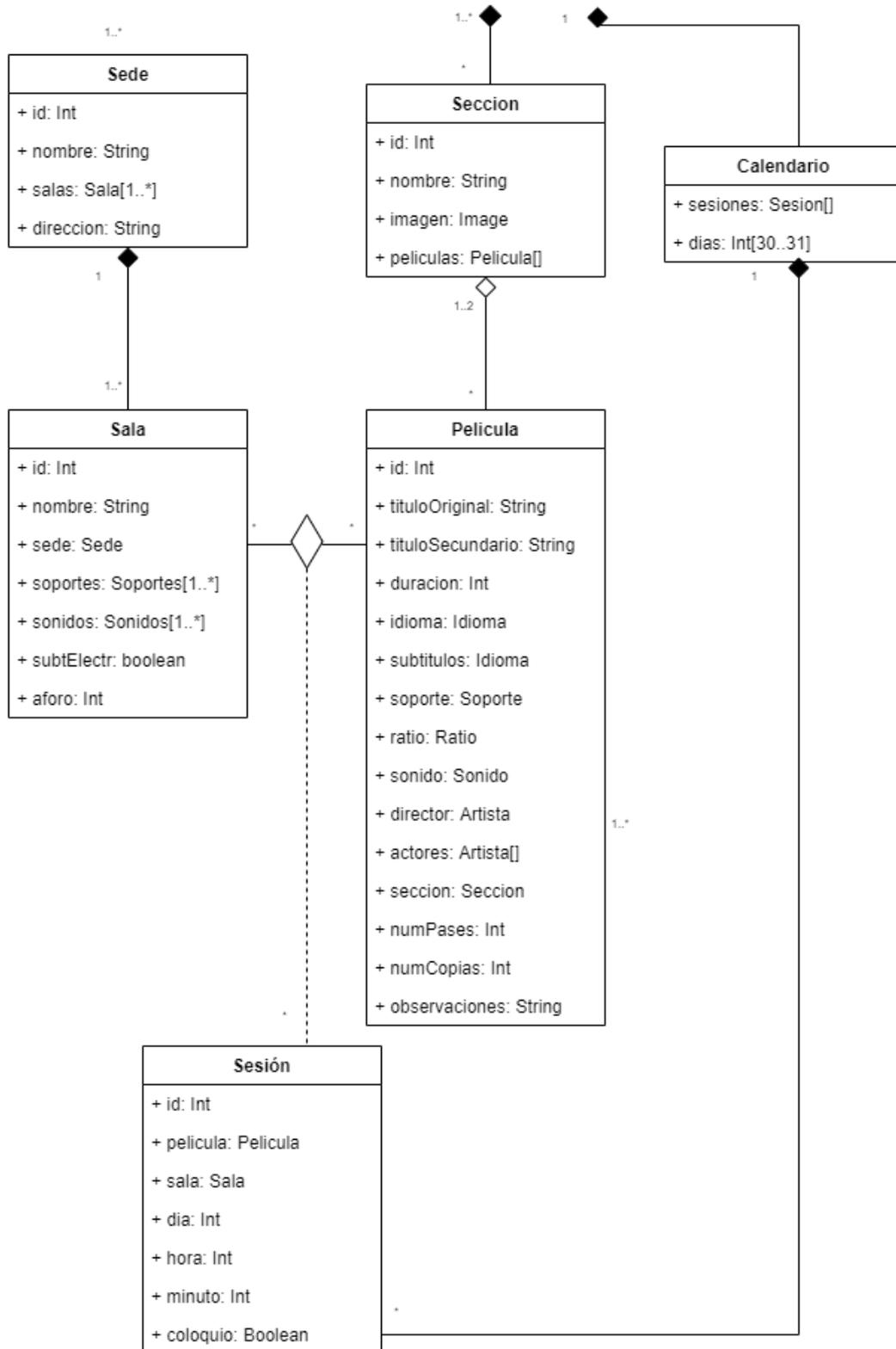


Figura 20: Diagrama entre los modelos Sede, Sección, Película, Sala, Sesión y Calendario

En cuanto a los usuarios de la aplicación, se trabaja desde la premisa de que todos son trabajadores del festival, por lo que comparten muchos atributos y métodos. De esta manera, el staff es una clase abstracta, que no tiene identidad por sí misma, sino que es la clase madre de todos los departamentos que trabajarán con la aplicación. Cada clase hija que hereda los atributos y métodos de staff, implementa los métodos según lo que necesita ejecutar.

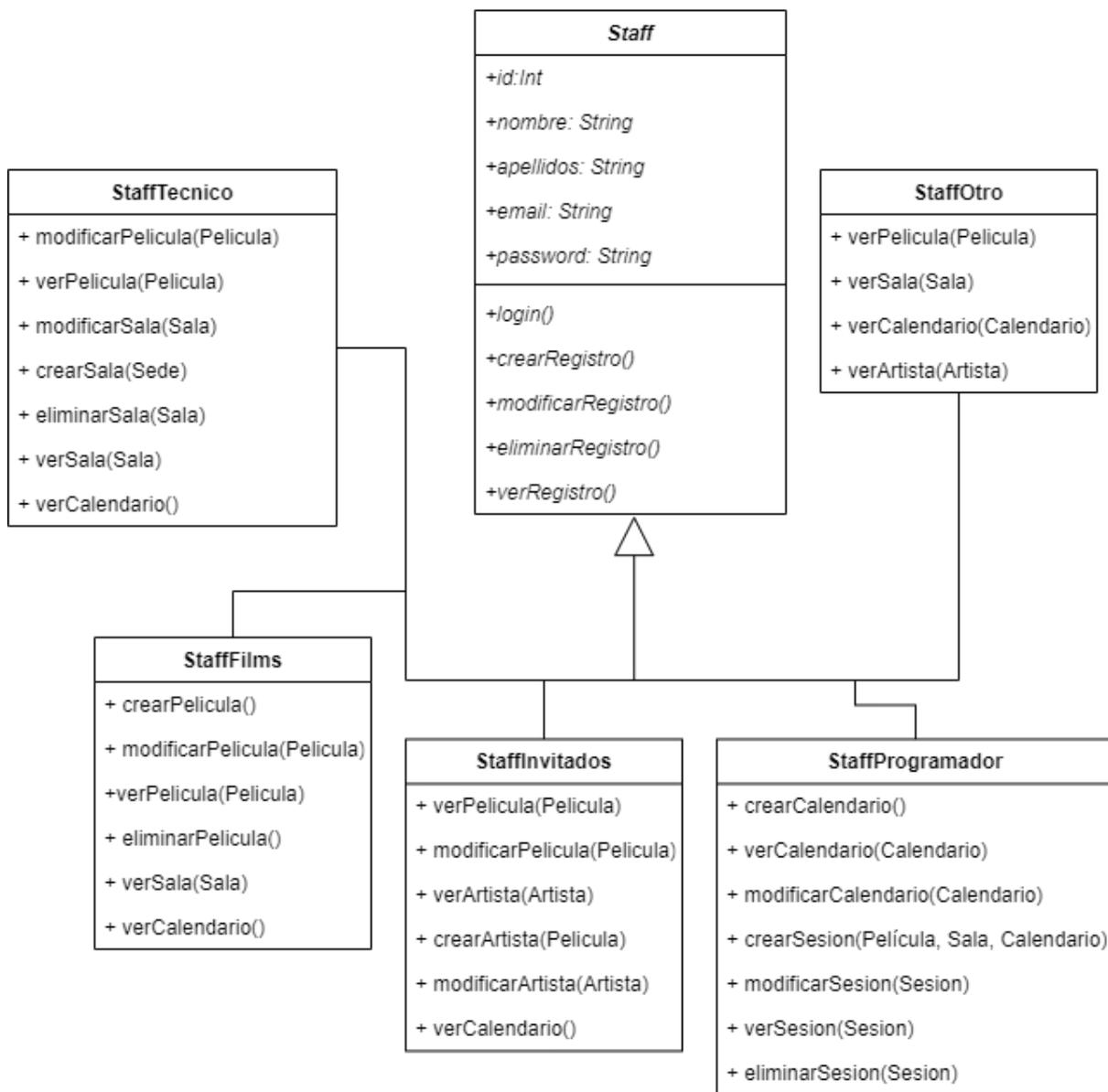


Figura 21: Relación de herencia de Staff

4.2.2. Diagramas de navegación

En la siguiente imagen, figura 22, se detalla el árbol de navegación de la aplicación web. Este diagrama divide el contenido entre la navegación principal, contenido principal y navegación secundaria.

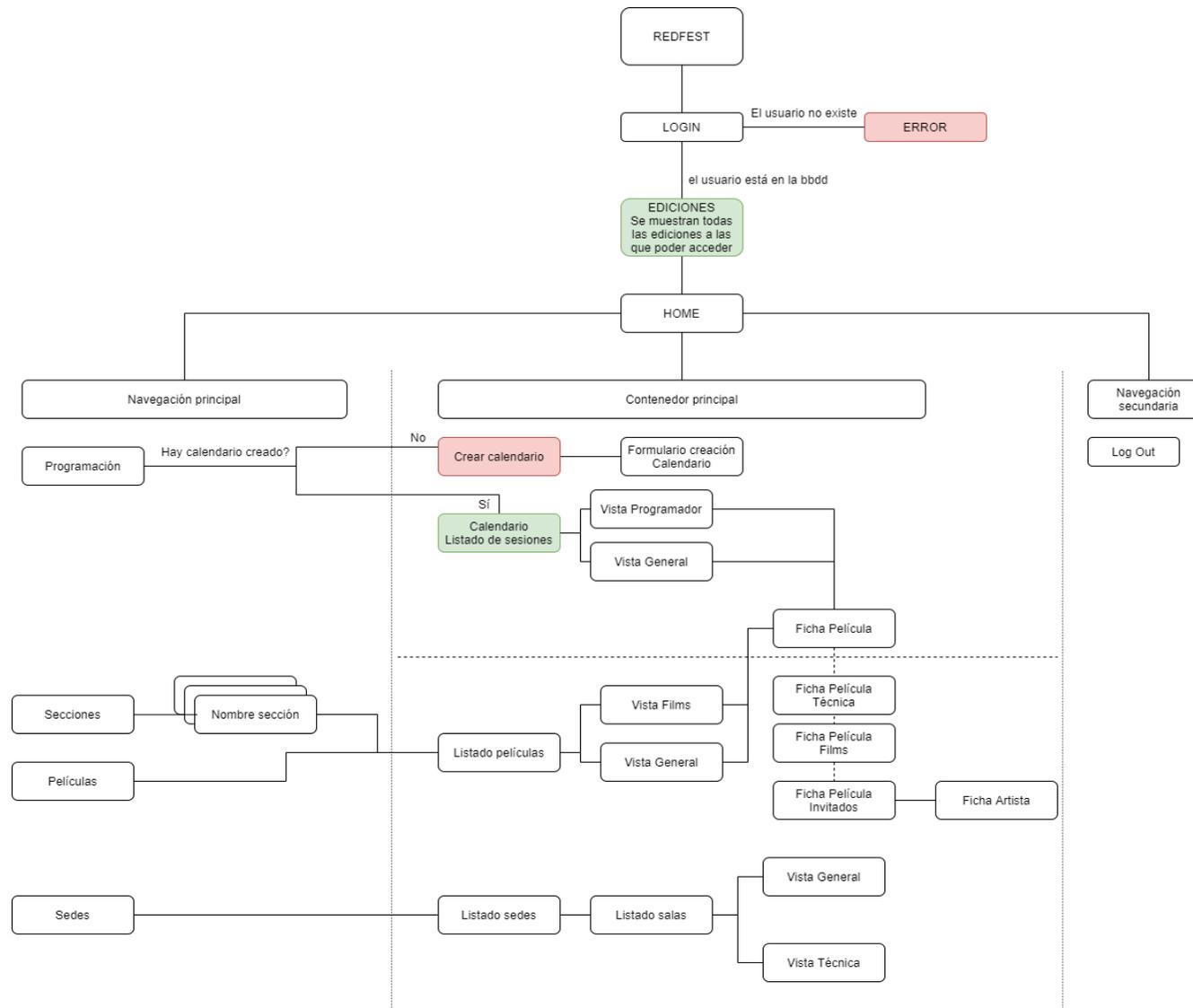


Figura 22: Árbol de navegación

4.3. Diseño gráfico e interfaces

4.3.1. Estilos

LOGOTIPO

Este es el logotipo creado para la aplicación y sus diferentes versiones:



Figura 23: Logo de la aplicación REDFEST

El proceso de creación del logotipo ha sido el siguiente:

- 1- Primero se ha trabajado la unión de las dos palabras que conforman el anagrama del nombre de la aplicación.



Figura 24: Creación del logo fase 1

- 2- Tras esta primera unión, la forma ha evocado a un paraguas. Ese paraguas bien podría ser la metáfora de lo que pretende ser la aplicación para el festival, ese paraguas bajo el que están todos los puntos, departamentos o elementos que conforman el festival.



Figura 25: Creación del logo fase 2

- 3- Para indicar esos elementos que están bajo ese paraguas se hacen pruebas de diferentes formas como se ve en estos ejemplos. Por una cuestión de proporciones se prescinde de parte del nombre.



Figura 26: Creación del logo fase 3

- 4- Con el objetivo de lograr un logotipo con elementos que nos remitan al mundo del cine, se transforman los puntos convirtiéndolos en la forma que tiene una perforación de un fotograma en 35mm.



Figura 27: Creación del logo fase 4

- 5- En un trabajo de forzar este concepto y convertirlo en un mensaje menos sutil y más directo hacia el cliente, se invierten los colores, mostrando claramente las bandas perforadas de un fotograma.



Figura 28: Creación del logo fase 5

- 6- Tras este cambio, se apuesta por convertir el medio arco superior en la banda superior del fotograma para así crear el fotograma completo.



Figura 29: Creación del logo fase 6

- 7- Finalmente, se ha optado por retomar la idea inicial de unir las dos palabras que forman el nombre de la aplicación, ya que el anagrama “fest” es clave para que el cliente se sienta más identificado.



Figura 30: Creación del logo última fase

TIPOGRAFÍA

La tipografía seleccionada para la aplicación es la Montserrat. Esta tipografía de Google está basada en la tipografía de la señalización urbana del barrio de Montserrat en Buenos Aires. Es una tipografía que facilita la lectura, incluso en tamaños pequeños. Su sencillez y facilidad de lectura son las razones principales por la que esta es la tipografía seleccionada. Esta tipografía está bajo la licencia *Open Font License*¹⁸

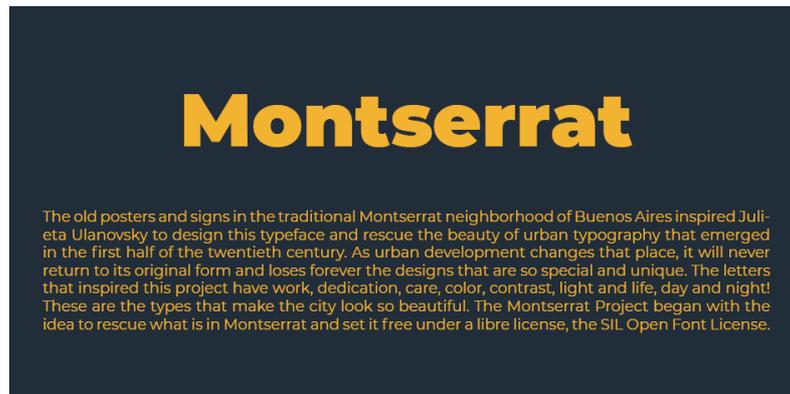


Figura 31: Tipografía Montserrat¹⁹

¹⁸ «SIL Open Font License (OFL)». https://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=OFL

¹⁹ «Montserrat». <https://localfonts.eu/freefonts/bulgarian-cyrillic/montserrat/>



Figura 32: Tipografía Montserrat en Google Fonts²⁰

PALETA DE COLORES

Para crear el logo se han seleccionado dos colores, uno más neutro y otro mucho más vivo que hace que la palabra “RED” destaque. Se ha jugado con el doble sentido de la palabra “Red”, rojo en inglés, para resaltar su presencia e importancia. El color neutro utilizado es el que predominará en los elementos oscuros de la aplicación. Es un color que transmite seriedad, pero alejándose de la agresividad de un color negro puro.



Figura 33: Paleta de colores principales creada a través de Colors²¹

Para el resto de la paleta se han seleccionado los siguientes colores:

²⁰ «Google Fonts». [Montserrat - Google Fonts](#)

²¹ «Create a Palette - Colors». <https://colors.co/>

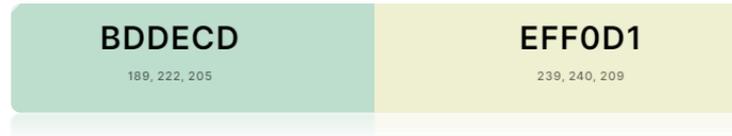


Figura 34: Paleta de colores secundarios

Estos dos primeros colores complementan al color principal, creando una pequeña paleta de rojo, verde y amarillo, colores básicos reconocibles por cualquier perfil de usuario. La idea es utilizar códigos de color básicos como este no solo para decorar si no para transmitir cierta información dentro de la aplicación.

Por último, se utilizará el color oscuro principal y el color de la siguiente imagen para tratar todos los textos según su contexto e intención.



Figura 35: Paleta de colores terciarios para el texto

Se ha tratado de reducir la saturación de toda la paleta para generar cierta sensación de calma a los usuarios. Esta serie de colores con poca saturación, generan poco contraste entre unos elementos y otros, pero sí el contraste suficiente para que aporten valor a la usabilidad.

BOTONES Y OTROS ELEMENTOS

Estos son varios ejemplos de botones, enlaces y formularios que demuestran lo indicado en el apartado anterior. La intención es que los colores de los botones indiquen cuál es su propósito y así facilitar el uso de la aplicación a personas menos acostumbradas al uso de nuevas tecnologías.



Figura 36: Ejemplo de botones básicos con diferentes fondos y objetivos

Link Link

Figura 37: Ejemplo de enlace sin entrar y enlace con el ratón encima

4.3.2. Usabilidad /UX

Como se comenta en otros apartados del TFG, la principal característica o ventaja de esta aplicación frente a otras que pretenden cubrir el mismo objetivo es que REDFEST está completamente enfocada al cliente. Cuando se nombra al cliente no solo se está hablando de un festival de cine como entidad si no de los trabajadores de ese festival, es decir los usuarios de la aplicación.

Una de las carencias analizadas en la fase de análisis de mercado es que tanto la diversidad de tipos de usuario como el poco tiempo de adaptación dificultan o imposibilitan que las aplicaciones ya existentes puedan implementarse en un flujo de trabajo como es el de un festival de cine. Es por esto por lo que la usabilidad adquiere una gran relevancia en la conceptualización de la aplicación, y se ha optado por un diseño centrado en el usuario (DCU²²).

Para llevar a cabo este proceso de creación del diseño, las fases son las siguientes:

- Especificar el contexto de uso
- Especificar los requisitos y objetivos del usuario
- Diseñar soluciones a esos requisitos y objetivos
- Evaluar a través de tests con usuarios



Figura 38: Proceso de Diseño Centrado en el Usuario²³

Cabe destacar que, para trabajar la usabilidad, además de crear un diseño centrado en el usuario, se ha trabajado bajo el decálogo de Jakob Nielsen²⁴:

²² «Diseño centrado en el usuario». <http://multimedia.uoc.edu/blogs/fem/es/disenio-centrado-en-el-usuario-conceptos-basicos/>

²³ Hassan Montero y Ortega Santamaría, «Informe APEI sobre usabilidad». <http://www.nosolousabilidad.com/manual/3.htm>

²⁴ «Jakob Nielsen» https://es.wikipedia.org/wiki/Jakob_Nielsen; «Las 10 reglas heurísticas de usabilidad de Jakob Nielsen | Diseño grafico, UI, UX y arquitectura de la información Barcelona».

- 1- Visibilidad del estado del sistema: el sistema debe informar al usuario de lo que está ocurriendo, en qué fase de cada proceso se encuentra. Por ejemplo, si no hay un calendario creado se debe indicar con un aviso para que el usuario no tenga dudas de si hay un error de carga o no.
- 2- Relación entre el sistema y el mundo real: el sistema debe utilizar una comunicación con el usuario que a este le resulte familiar.
- 3- Control y libertad del usuario: el usuario debe poder salir de cualquier proceso fácilmente. Es natural que el usuario se equivoque y quiera cancelar una acción.
- 4- Consistencia y estándares: se deben seguir las convenciones establecidas, no crear unas nuevas, para facilitar la comprensión por parte del usuario.
- 5- Prevención de errores: indicar claramente de antemano qué criterios pueden llevar a generar un error. Por ejemplo, será importante que el usuario tenga un botón de guardar cada vez que edite cualquier registro de la base de datos, así se asegurará de que el cambio se ha realizado.
- 6- Reconocimiento antes que recuerdo: mantener siempre visibles todas las opciones que están al alcance del usuario, no asumir que ya debería saber ejecutar ningún proceso.
- 7- Flexibilidad y eficiencia de uso: Crear atajos para agilizar los procesos a los usuarios más hábiles.
- 8- Estética y diseño minimalista: como se ha comentado varias veces en el TFG, es importante que la aplicación solo muestre la información verdaderamente relevante para facilitar su uso.
- 9- Ayudar a los usuarios a reconocer errores: indicar los errores con un mensaje claro y conciso con una posible vía de solución.
- 10- Ayuda y documentación: crear la documentación necesaria para resolver las dudas que el usuario pueda tener.

4.4. Lenguajes de programación y APIs utilizados

4.4.1. Lenguajes de programación

La aplicación REDFEST, al tratarse de una herramienta de trabajo que sustituye métodos y procesos que solo se ejecutan en PC, se ha creado para que se utilice exclusivamente en navegadores web y así mantener el *hardware* al que el usuario está acostumbrado. Es por esto por lo que el lenguaje de programación seleccionado es Javascript.

BACKEND

Javascript es un lenguaje de programación que originalmente se utilizaba solo en el lado del cliente pero que, en los últimos años, gracias a su popularidad y demanda, se ha extendido hasta el lado del servidor. Este cambio se ha podido llevar a cabo gracias a NodeJS que nos permite utilizar Javascript en el *backend*.



NodeJS es un entorno de ejecución basado en el lenguaje Javascript que permite crear un servidor web con este lenguaje, facilitando la comunicación entre el *frontend* y el *backend*. Es un entorno que permite crear programas altamente escalables.



Para facilitar el uso de NodeJS y agilizar ciertos procesos, se ha utilizado el *framework* Express. Este *framework* de NodeJS también utiliza Javascript. Gracias al uso de NodeJS a través de ExpressJS, el *backend* de la aplicación se basa en la creación de una API REST.

Una API REST permite la conexión entre el usuario y la base de datos, es decir, la API es la interfaz de programación que se encarga de las solicitudes y devuelve las respuestas en formato JSON²⁵. Una API es REST porque se basa en la arquitectura RESTful, que se define por las siguientes características:

- El servidor se encarga de manejar los datos mientras que el cliente se encarga de manejar las interacciones del usuario.
- Las peticiones recibidas por el servidor son cacheables

²⁵ «IBM Docs», 24 de junio de 2021. <https://www.ibm.com/docs/es/baw/20.x?topic=formats-javascript-object-notation-json-format>

- Cada interacción está asociada a una dirección única “URI”
- Cada petición recibida en el servidor es independiente y tiene toda la información necesaria para procesarla
- El servidor puede tener varias capas.



La información que se maneja en la API está almacenada en una base de datos no relacional creada en MongoDB. Este sistema de base de datos no relacional trabaja con objetos y colecciones de dichos objetos en lugar de con registros interrelacionados.

FRONTEND



En cuanto al lado del cliente, el *framework* seleccionado es VueJS. Este es un *framework* que permite trabajar por componentes generando una SPA que trata o muestra cada componente según sea necesario. VueJS se comunica con la API REST solicitándole información a través de diferentes *endpoints*, y la API devuelve la información para que VueJS la muestre.

En cuanto a los estilos de la aplicación, el *framework* elegido es Vuetify²⁶ que permite trabajar con Vue a través del denominado *Material Design*²⁷.



Esta combinación de herramientas que trabajan con Javascript conforman el denominado *full-stack* MEVN.

²⁶ «Vuetify — A Material Design Framework for Vue.js» <https://vuetifyjs.com/en/>; «Divider Component»; [vuetifyjs/vuetify](https://vuetifyjs.com/en/).

²⁷ «Material Icons Guide | Google Fonts | Google Developers»; «Material Design» <https://material.io/components>

4.4.2. Software

El software que se ha utilizado para la realización de este TFG es el siguiente:

- **Adobe Creative Cloud**²⁸ para la creación de todos los diseños de imágenes, vídeo de presentación y creación de línea gráfica.
- **Just in Mind**²⁹ para la creación de prototipos
- **Draw.io**³⁰ para la creación de diagramas
- **Visual Studio Code**³¹ para la escritura de código
- **MongoDB Compass**³² para la gestión de las bases de datos a través de una interfaz.
- **Postman**³³ para gestionar los ciclos de vida de la API
- **Git**³⁴ para la gestión de versiones
- **Github**³⁵ para la gestión del repositorio

²⁸ «Adobe Creative Cloud | Detalles y productos | Adobe».

<https://www.adobe.com/es/creativecloud.html?>

²⁹ «Free prototyping tool for web & mobile apps - Justinmind». <https://www.justinmind.com/>

³⁰ «diagrams.net». <https://app.diagrams.net/>

³¹ «Visual Studio Code - Code Editing. Redefined». <https://code.visualstudio.com/>

³² «La base de datos líder del mercado para aplicaciones modernas».

<https://www.mongodb.com/es/products/compass>

³³ «Postman API Platform | Sign Up for Free». <https://www.postman.com/>

³⁴ «Git». <https://git-scm.com/>

³⁵ «Build Software Better, Together». <https://github.com/>

5. Implementación

5.1. Requisitos de instalación

5.1.1. Requisitos durante el desarrollo

Para instalar la aplicación web durante el desarrollo de esta, los requisitos son los siguientes:

- Software
 - Sistema operativo Windows, Linux o Mac
 - Tener Node.js instalado
 - Tener MongoDB instalado
 - Tener Mongo Dev Tools instalado
 - Tener Babel instalado para transpilar el código Javascript

- Hardware
 - PC con navegador instalado (Chrome, Firefox a ser posible)

5.1.2. Requisitos para el cliente

Una vez la aplicación está desplegada, se accede a la aplicación a través del navegador web sin necesidad de realizar ninguna instalación extra.

El *link* de acceso a la aplicación es <https://redfest.herokuapp.com/>

5.2. Instrucciones de instalación

Durante el desarrollo de la aplicación hay que realizar todos los siguientes pasos para poder acceder a la aplicación. En el caso de acceder a la aplicación una vez esté desplegada, solo hay que realizar las instrucciones del paso 5.2.7 "Acceso y *Login*"

5.2.1. NODE JS

Para instalar NodeJS³⁶ hay que acceder a <https://nodejs.org/es/> y descargar la versión recomendada como se muestra en la imagen.

³⁶ Node.js <https://nodejs.org/es/>



Figura 39: Landing de instalación de NodeJS

5.2.2. MONGO DB

Para instalar MongoDB³⁷ hay que acceder a www.mongodb.com/try/download/community y descargar el archivo .msi del MongoDB Community Server.

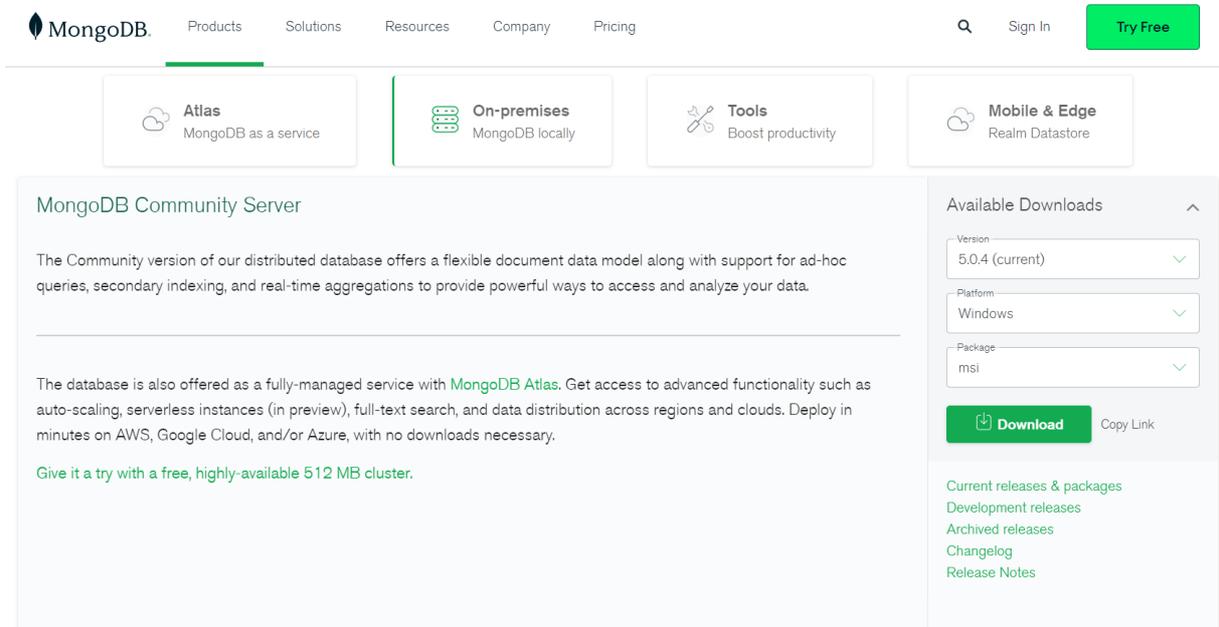


Figura 40: Instalación de Mongo DB Local

³⁷ «MongoDB Community Download». <https://www.mongodb.com/try/download/community>

5.2.3. MONGO DEV TOOLS

Hay que instalar estas herramientas de Mongo para poder restaurar la base de datos. Hay que acceder a <https://www.mongodb.com/try/download/database-tools> y descargarse el msi con la *MongoDB Database Tools*.³⁸

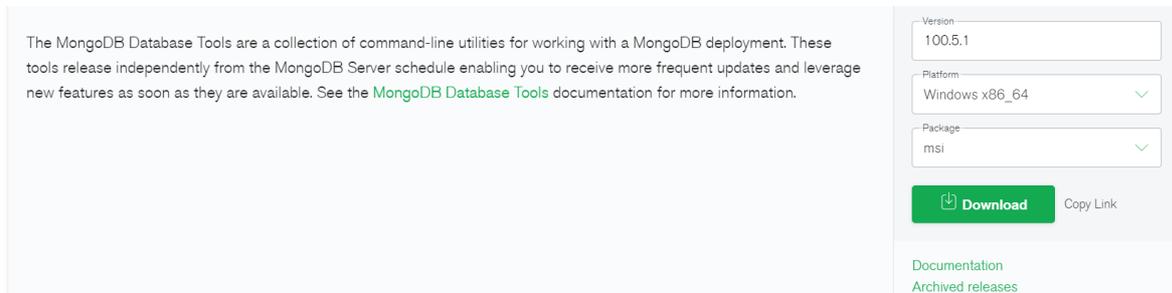


Figura 41: Instalación de MongoDB Dev Tools

Una vez instaladas, estos son los pasos a seguir para restaurar la base de datos adjuntada en el proyecto:

- Copiar la carpeta llamada “dump” en la carpeta **c:/Archivos de programa/Mongodb>Tools>100>bin**

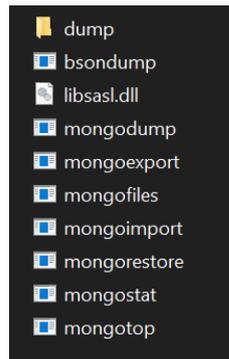


Figura 42: Carpeta “dump” copiada en el directorio de las Mogo DevTools

- En la terminal a través del comando “cd” ir al directorio **Mongodb>Tools>100>bin** donde se encuentra la carpeta “dump”
- Ejecutar en la terminal “mongorestore --db dbredfest dump/dbredfest”

```
c:\Archivos de programa\MongoDB\Tools\100\bin>mongorestore --db dbredfest dump/dbredfest
```

Figura 43: Mongorestore para restaurar la base de datos

- Abrir Mongo Compass y clicar en Connect para comprobar que la base de datos “dbredfest” ha sido correctamente restaurada con todas sus colecciones dentro.

³⁸ «Download MongoDB Command Line Database Tools». <https://www.mongodb.com/try/download/database-tools>

5.2.4. BABEL

Babel es un transpilador de código Javascript que nos permite convertir el código Javascript de la aplicación en código compatible con todos los navegadores y versiones antiguas de Node.

Se instala a través del siguiente comando “npm install -g babel -cli”:

```
C:\Users\marina\Desktop\Nueva carpeta\RedFestApp\api>npm install -g babel-cli
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated chokidar@1.7.0: Chokidar 2 will break on node v14+. Upgrade to chokidar 3 with 15x less dependencies
npm WARN deprecated core-js@2.6.12: core-js@<3.3 is no longer maintained and not recommended for usage due to the number
of issues. Because of the V8 engine whims, feature detection in old core-js versions could cause a slowdown up to 100x
even if nothing is polyfilled. Please, upgrade your dependencies to the actual version of core-js.
added 237 packages, and audited 238 packages in 35s
```

Figura 44: Instalación global de Babel para que transpile el código

5.2.5. ARRANCAR SERVIDOR

Una vez está todo lo anterior instalado, para arrancar el servidor hay que, en la terminal, cambiar al directorio **RedFestApp > api**, donde se haya guardado la carpeta RedFestApp, y usar el siguiente comando “npm start”:

```
C:\Users\PcCom\OneDrive\Documentos\UOC\TFG\RedFestApp\api>npm start
> redfest@1.0.0 start
> babel-node index.js --exec

server on port 3000
Conectando con la base de datos en el puerto 27017
```

Figura 45: Comando para arrancar la API REST

5.2.6. ARRANCAR VUE

Con el servidor ya corriendo, solo falta arrancar vue. Como el código entregado ya está en producción hay que descomentar la línea 8 del archivo main.js dentro de **RedFestApp > vue** y comentar la línea 9. Esta última línea es solo para producción, no para desarrollo.

```
axios.defaults.baseURL='http://localhost:3000/api/';
//axios.defaults.baseURL='https://redfest.herokuapp.com/api/';
```

Después hay que abrir otra terminal, ir al directorio **RedFestApp > vue** y ejecutar el siguiente comando “**npm run serve**”:

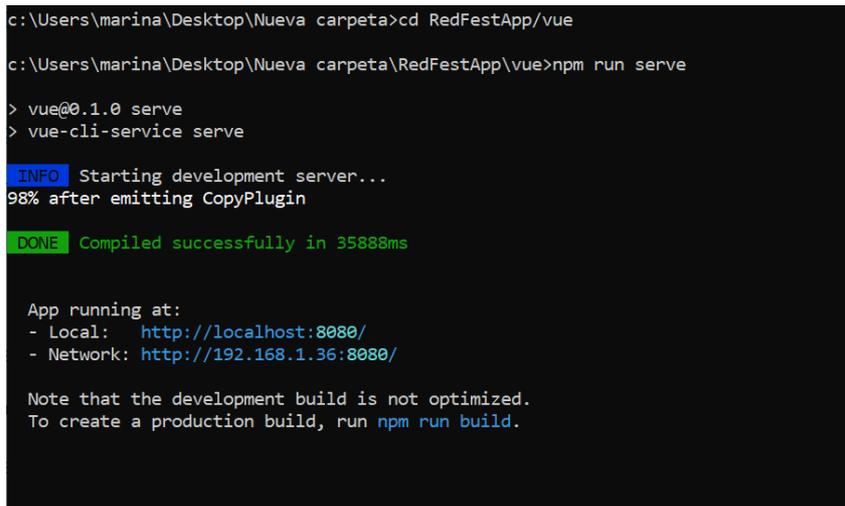


Figura 46: Comando para arrancar la aplicación

Esto nos devolverá un mensaje que indica que podemos acceder a nuestra aplicación a través de <http://localhost:8080/>

5.2.7. ACCESO Y LOGIN

Con la aplicación ya abierta, lo primero que tendremos es la pantalla de *login* donde hay que introducir el email y usuario. Estos son los diferentes usuarios y contraseñas para probar las diferencias de opciones entre roles:

Rol	Email	Password
Admin	admin@redfest.com	1234
Dept. Películas	peliculas@redfest.com	Peliculas
Dept. Técnica	tecnica@redfest.com	Tecnica
Dept. Invitados	invitados@redfest.com	Invitados
Dept. Programación	programador@redfest.com	Programador

Tabla 7: Tabla de modelos

6. Demostración

6.1. Instrucciones de uso

Para empezar a utilizar la aplicación solo hay que acceder a <https://redfest.herokuapp.com/>

Una vez la página ha cargado hay que *loguearse* para acceder a todas las opciones habilitadas para ese tipo de perfil. Los diferentes ejemplos de uso se detallan en el apartado [6.4.](#)

6.2. Prototipos

6.2.1. Prototipos Lo-Fi

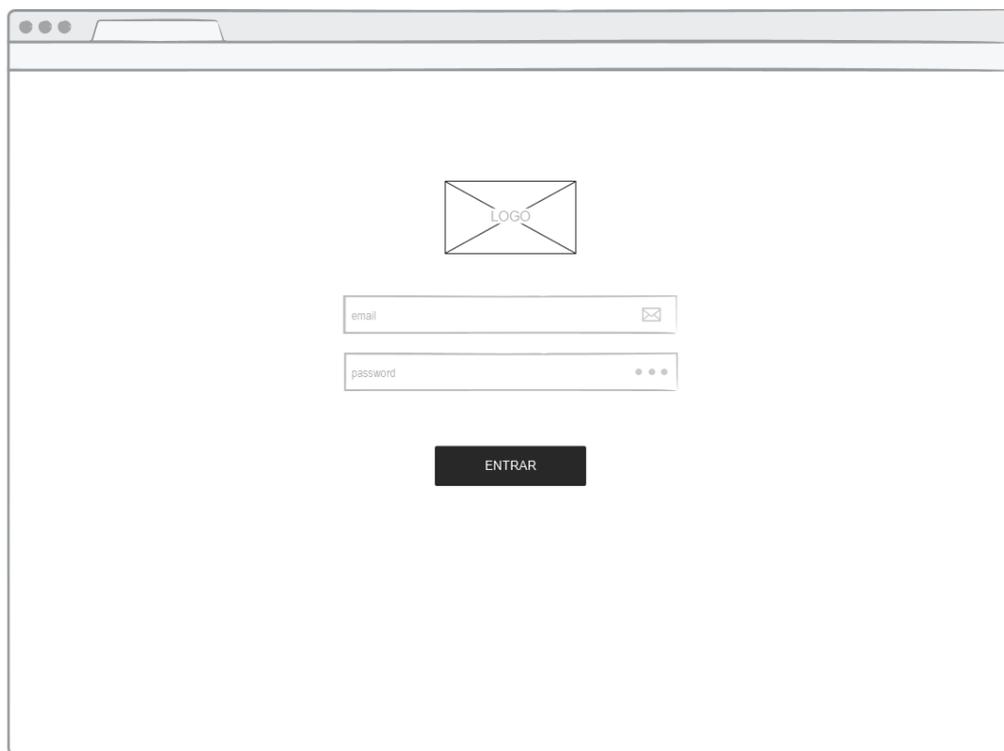


Figura 47: Pantalla de login

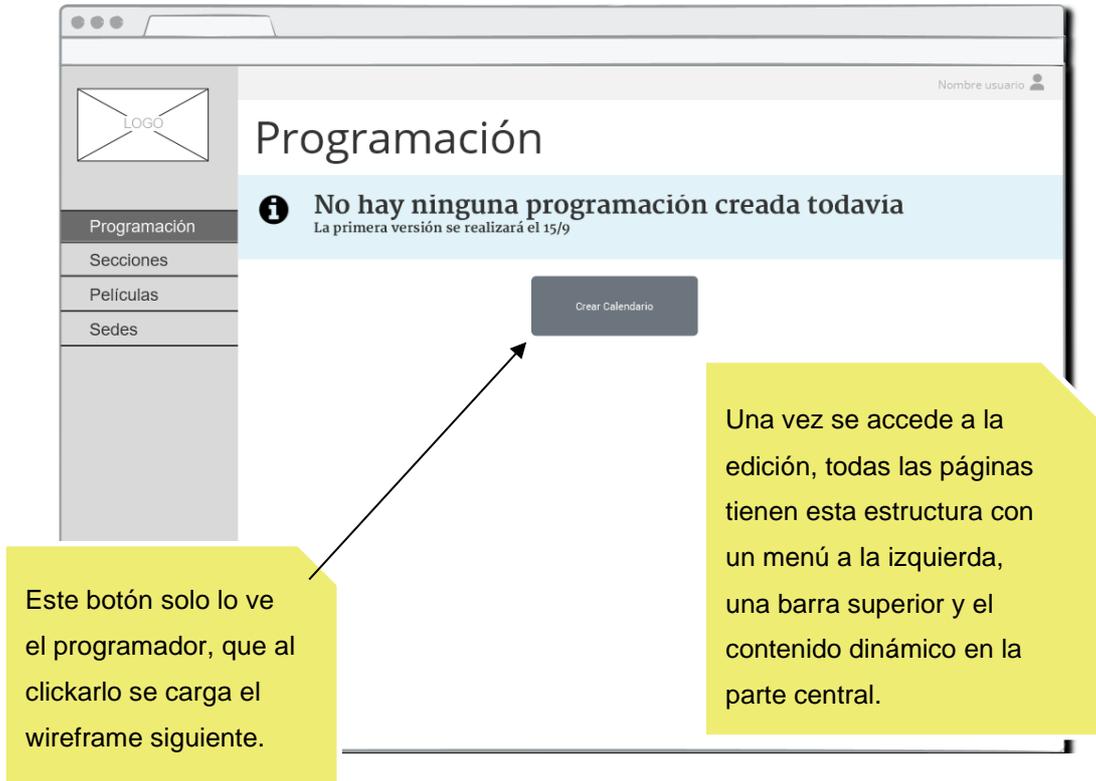


Figura 48: Pantalla de programación sin calendario creado

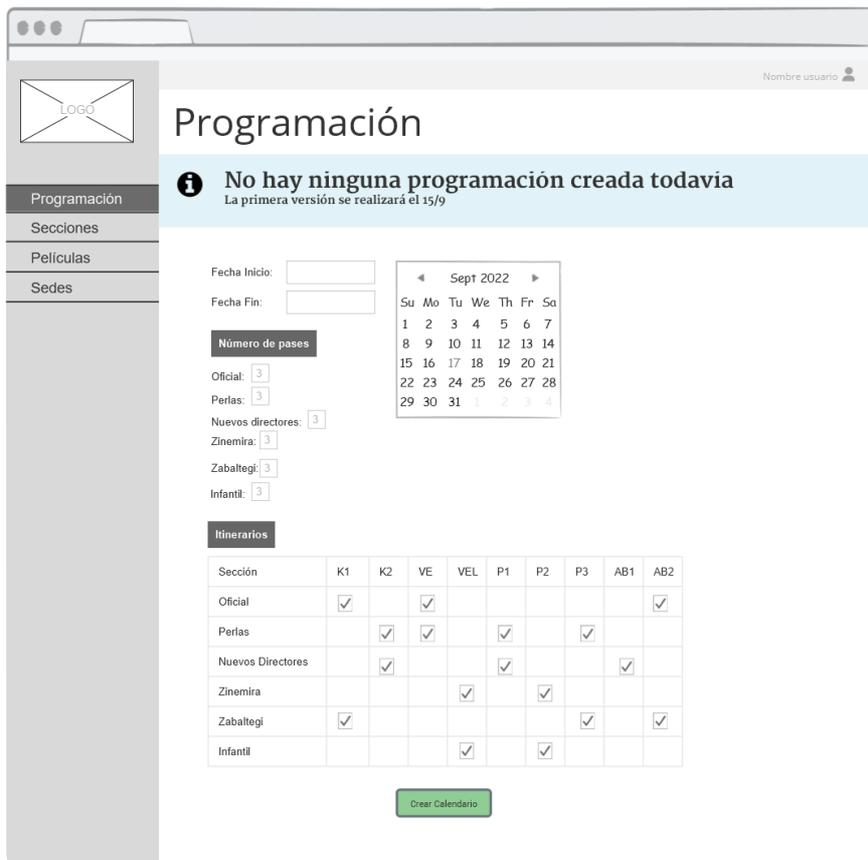


Figura 49: Pantalla de creación de la estructura de la programación

Título	Min	Idioma	Subt.	Electr.	Día	Hora	Sala	Sección
Película 1	74	Español	Inglés	No	22	12:00	K2	Oficial
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi
Película 2	85	Alemán	Inglés	Español	23	16:00	VE	Zabaltegi

Figura 50: Vista estándar de la Programación

Título	Min	Idioma	Subt.	Electr.	Soporte	Sonido	Director	Actores
Película 1	74	Español	Inglés	No	35MM	5.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	16MM	5.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	7.2	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	HDCAM	ATMOS	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	35MM	5.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	5.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	7.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	HDCAM	5.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	35MM	5.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	16MM	7.2	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	ATMOS	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	7.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	7.1	Nombre Apellidos	Nombre Apellidos
Película 2	85	Alemán	Inglés	Español	DCP	7.1	Nombre Apellidos	Nombre Apellidos

El menú Secciones es un desplegable

Figura 51: Vista estándar del listado de películas por secciones

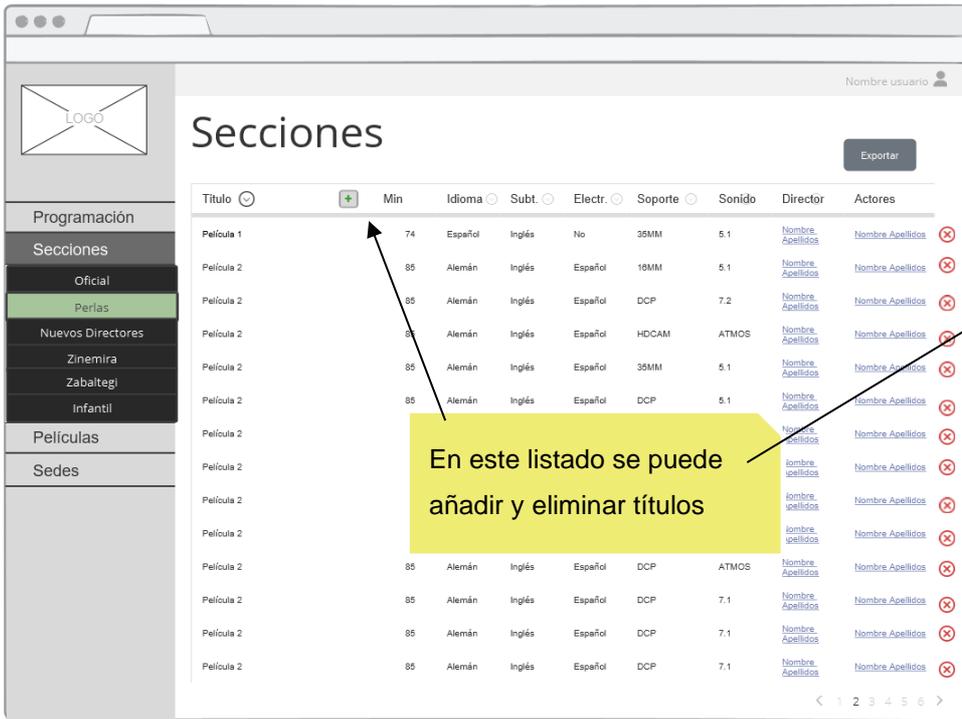


Figura 52: Vista para el departamento de Películas del listado de películas por secciones

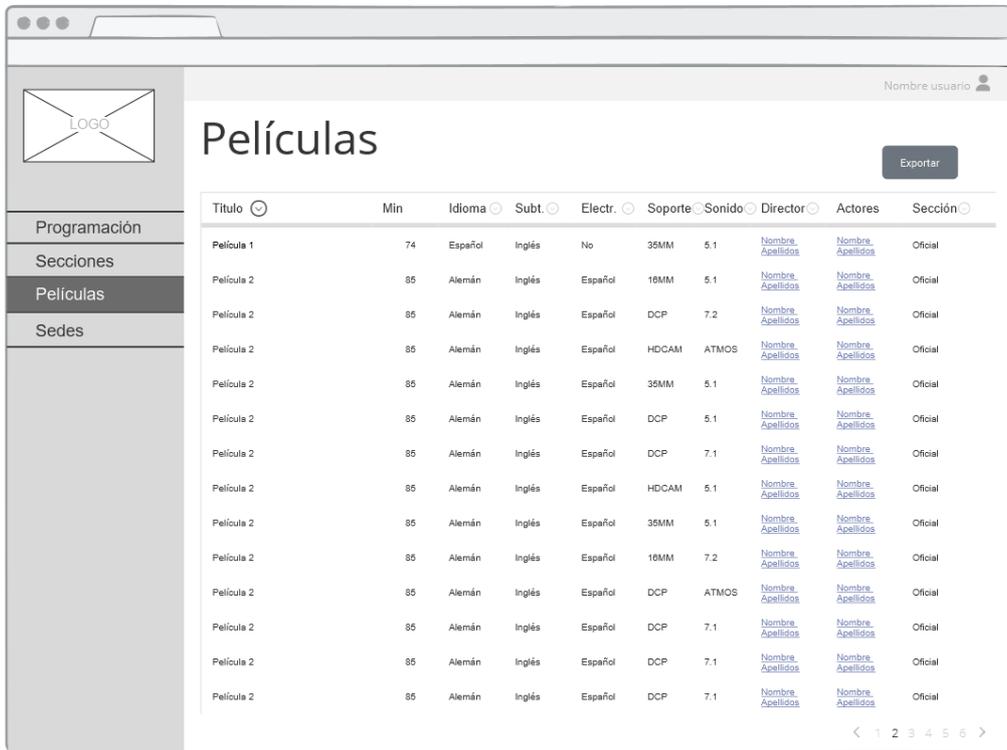


Figura 53: Vista estándar del listado de películas

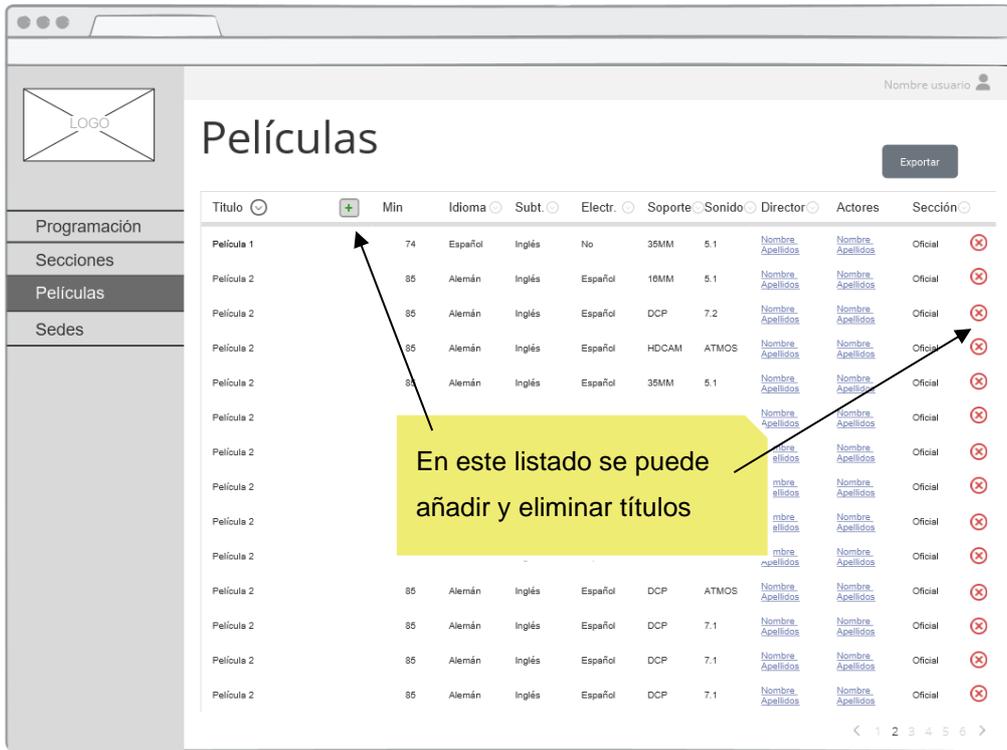


Figura 54: Vista para el departamento de Películas del listado de películas

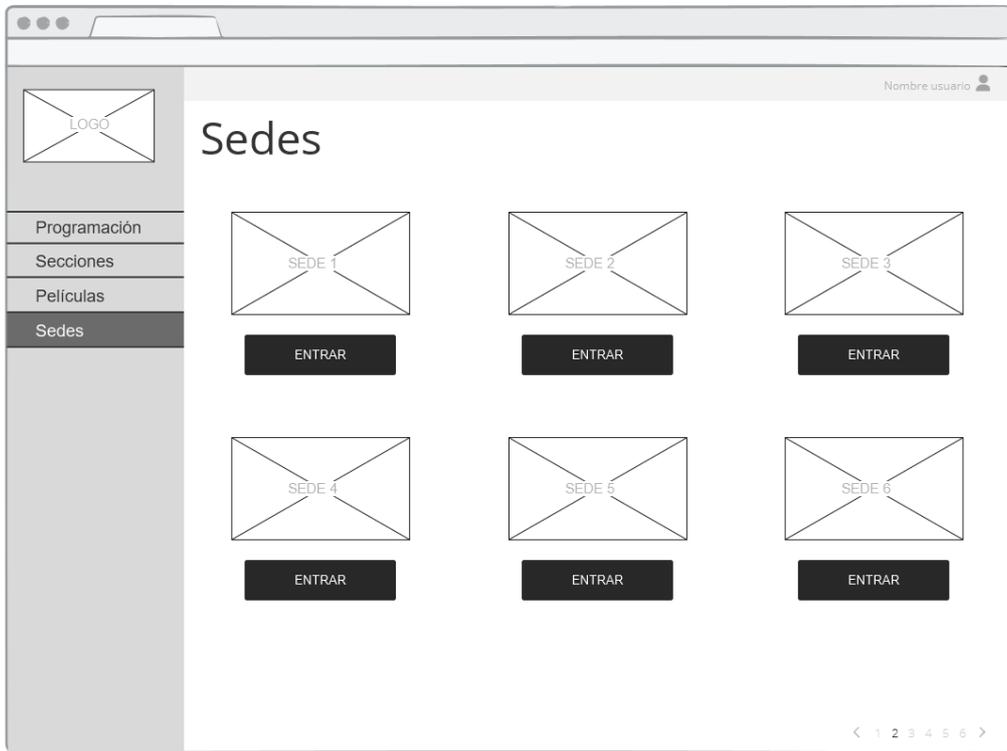


Figura 55: Vista de las sedes del festival

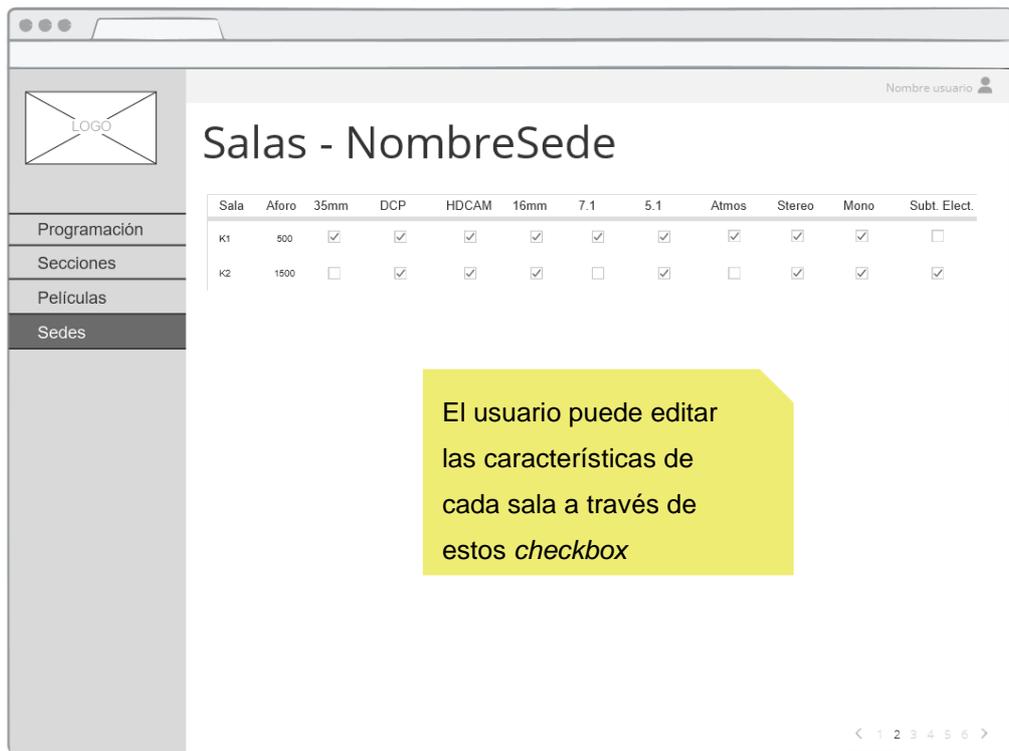


Figura 56: Vista para el departamento técnico de las salas de una sede

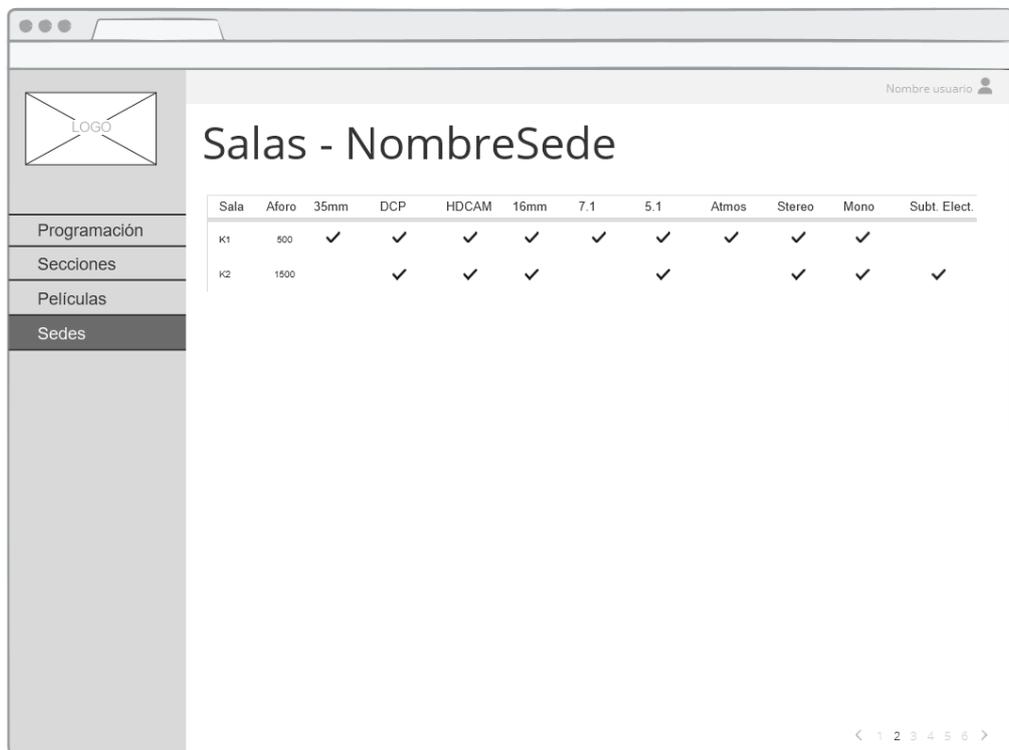


Figura 57: Vista de las características de las salas de una sede

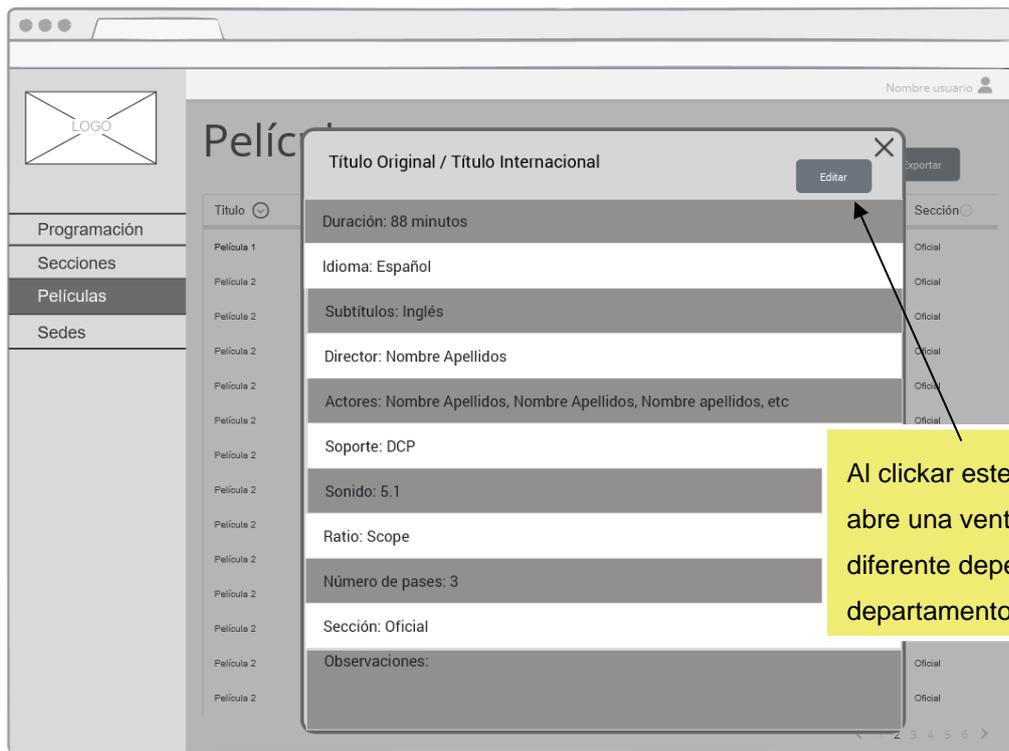


Figura 58: Ventana modal con la ficha de una película

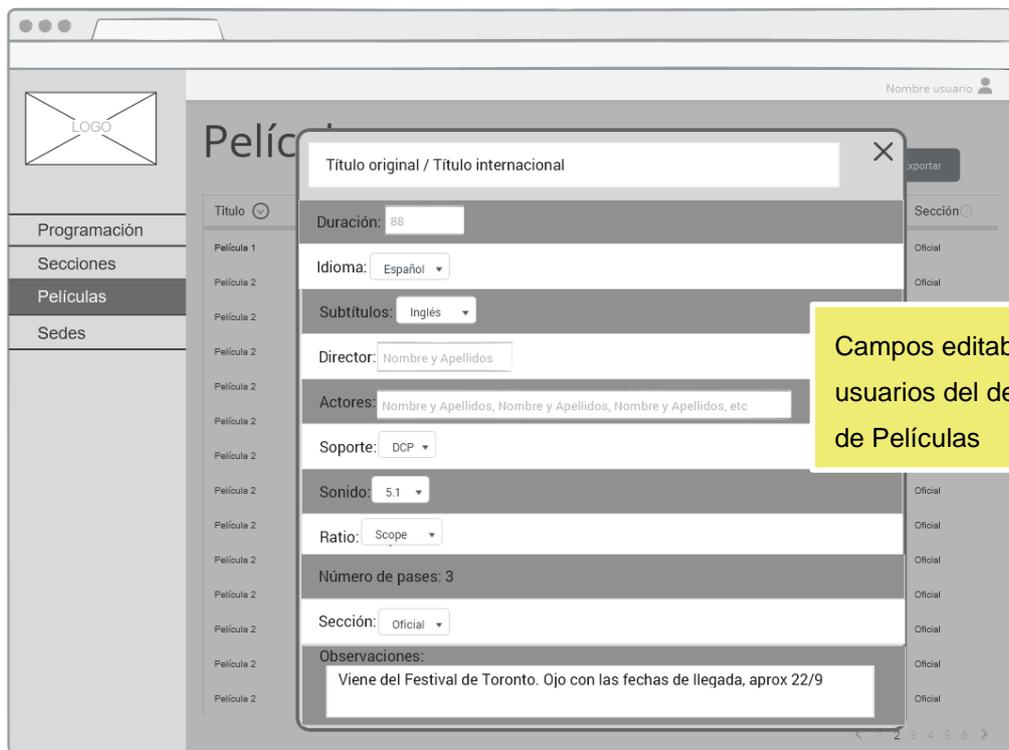


Figura 59: Ventana modal para el departamento películas con la ficha de una película

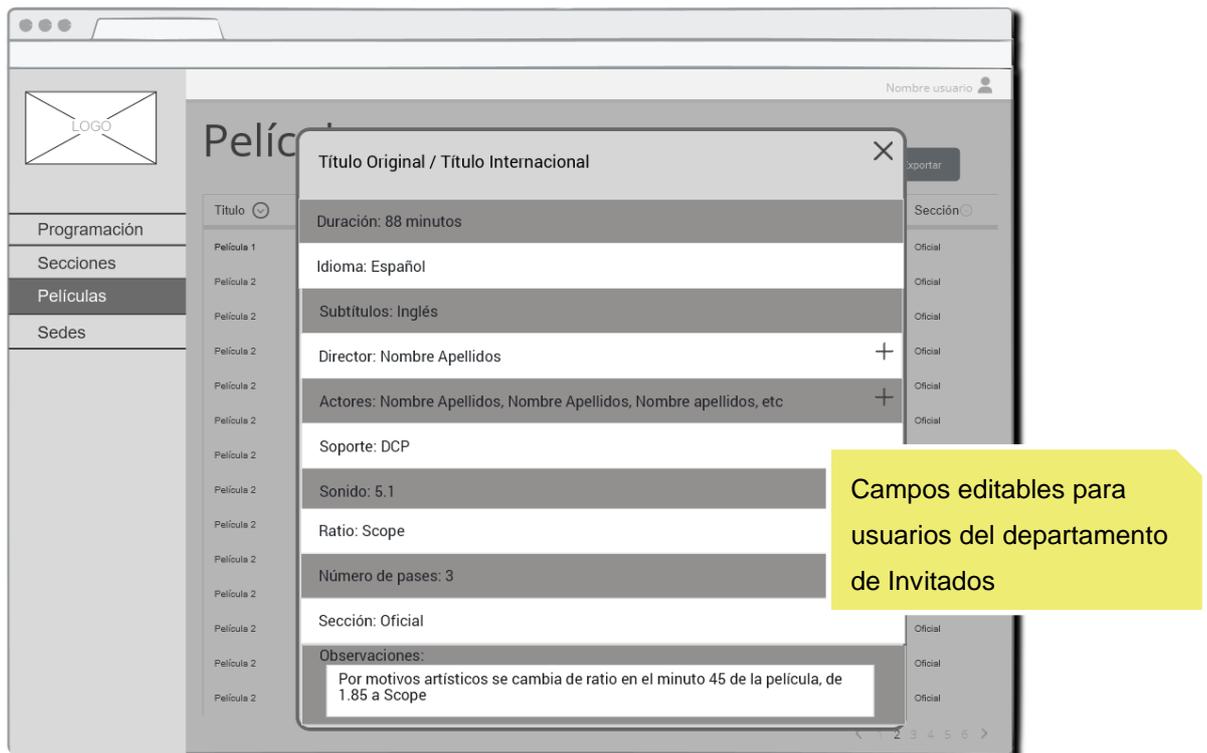


Figura 60: Ventana modal para el departamento de invitados con la ficha de una película



Figura 61: Ventana modal para el departamento técnico con la ficha de una película

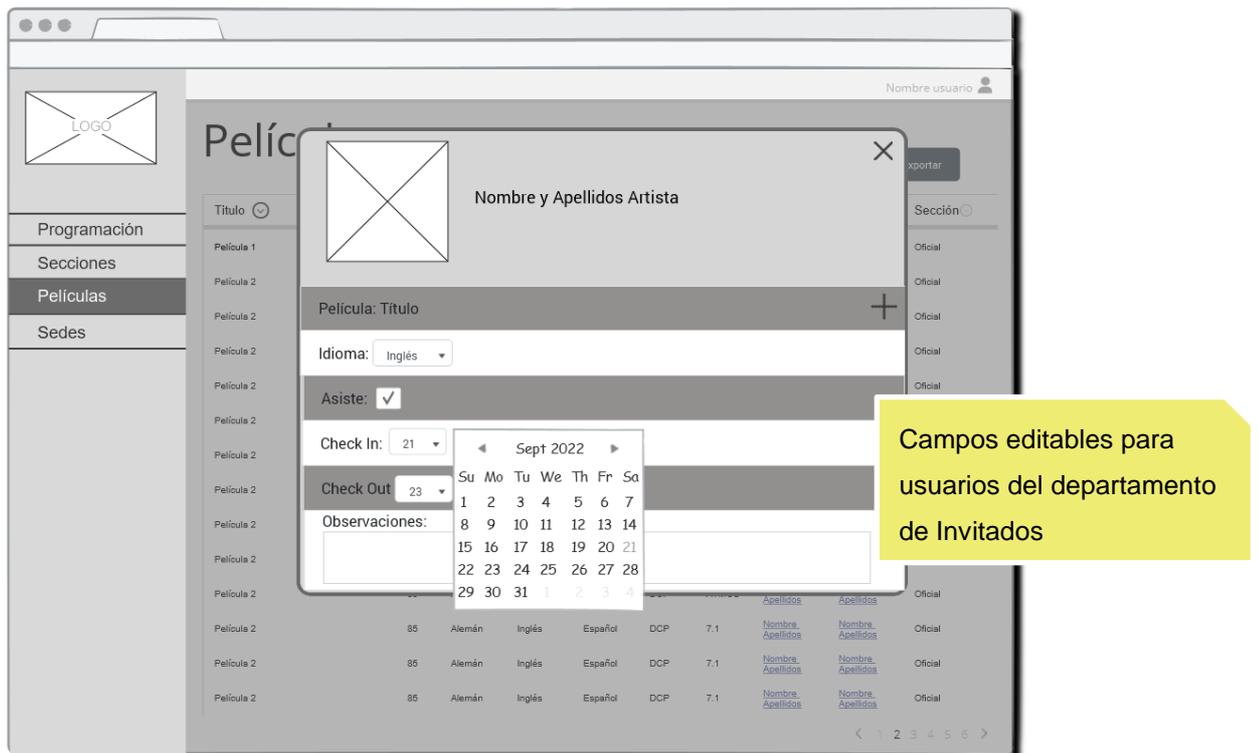


Figura 62: Ventana modal para el departamento de invitados con la ficha de un artista

6.2.2. Prototipos Hi-Fi

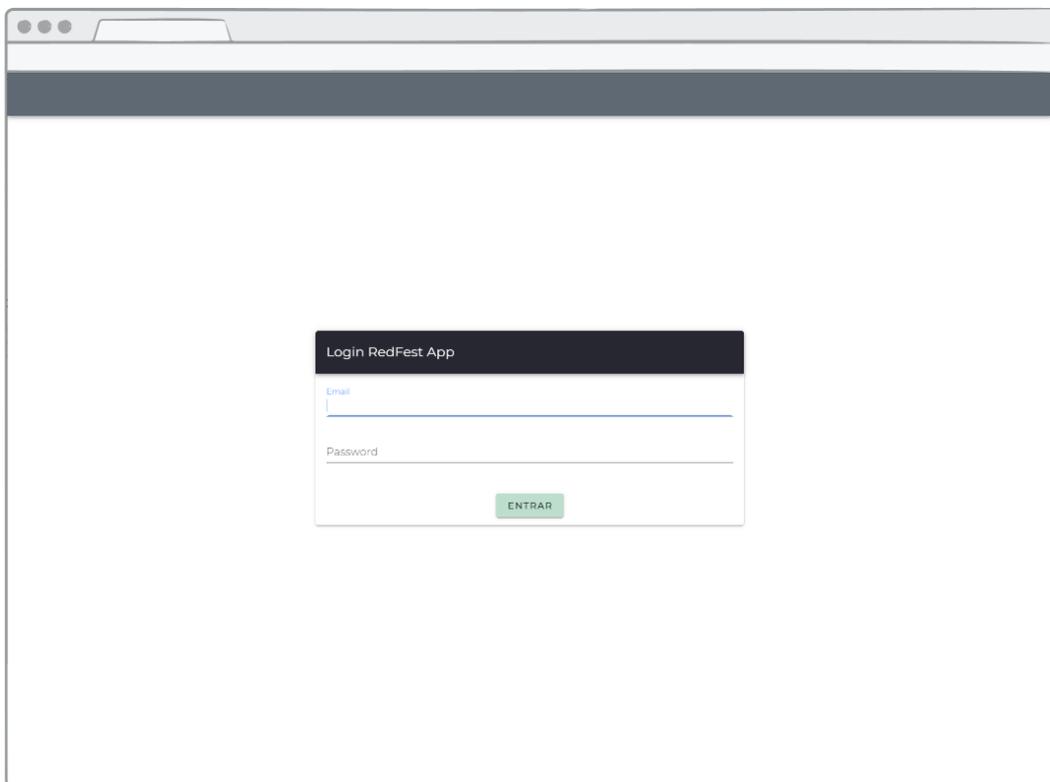


Figura 63: Pantalla de login

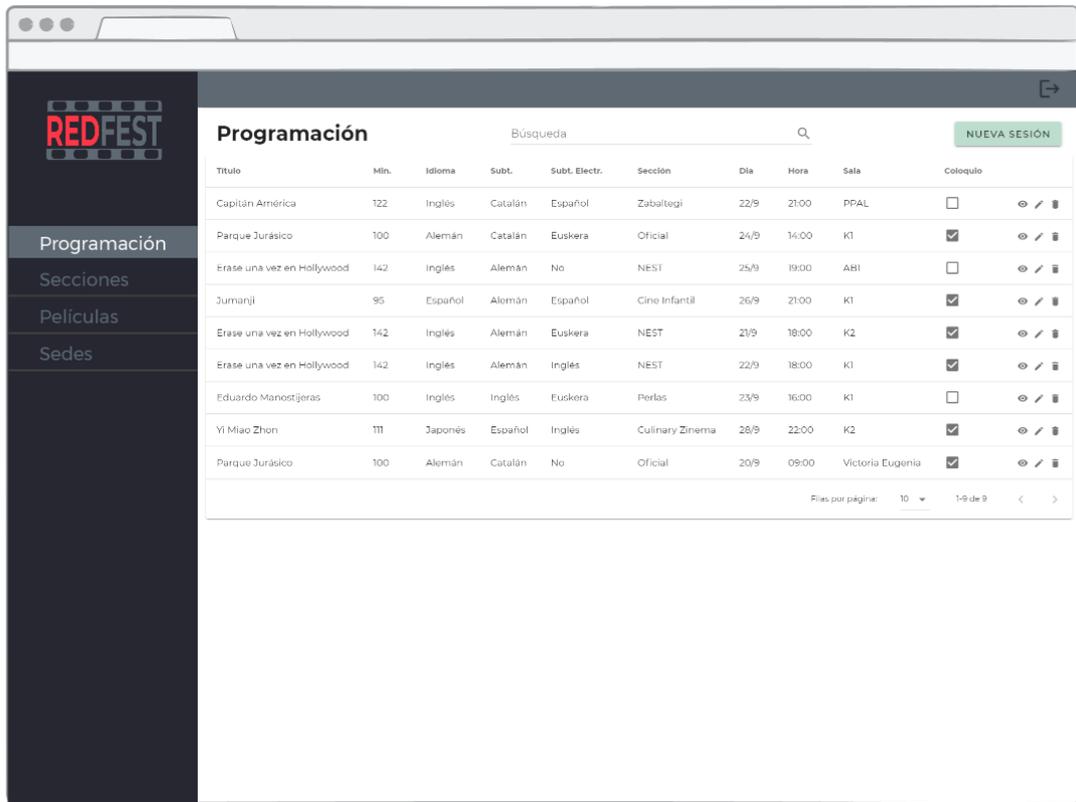


Figura 64: Vista estándar de la Programación

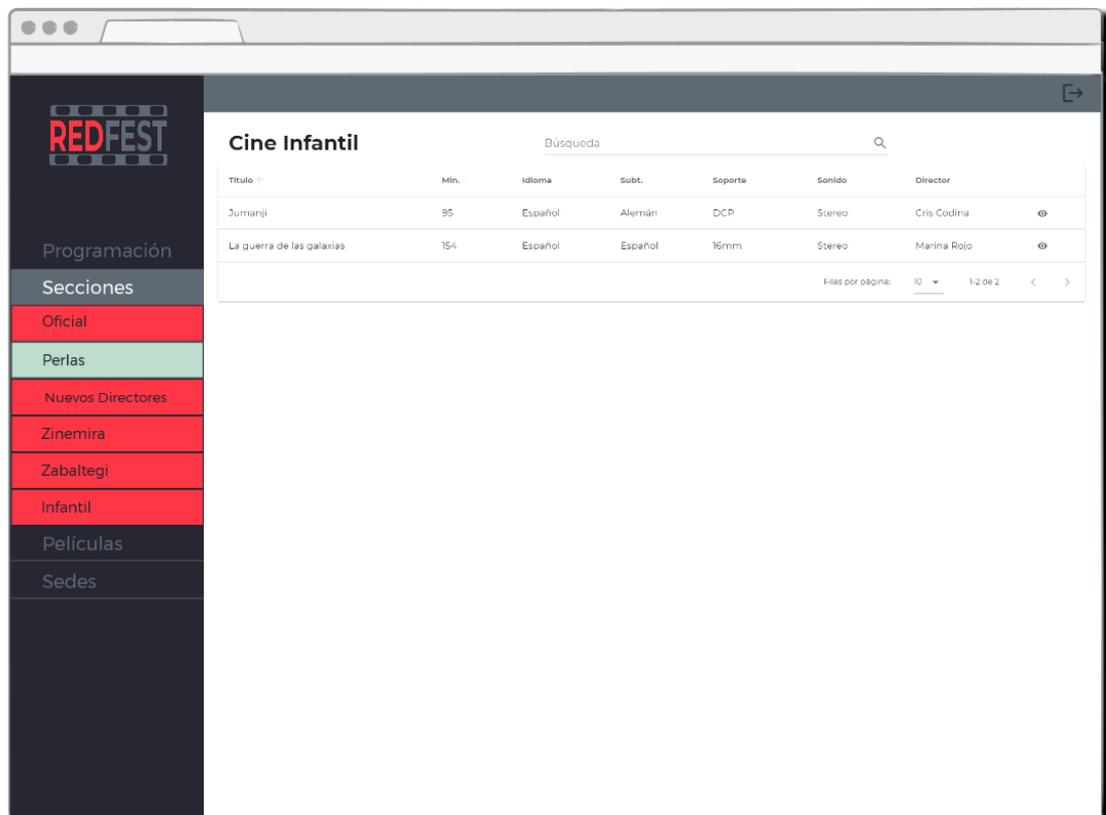


Figura 65: Vista del listado de películas por secciones

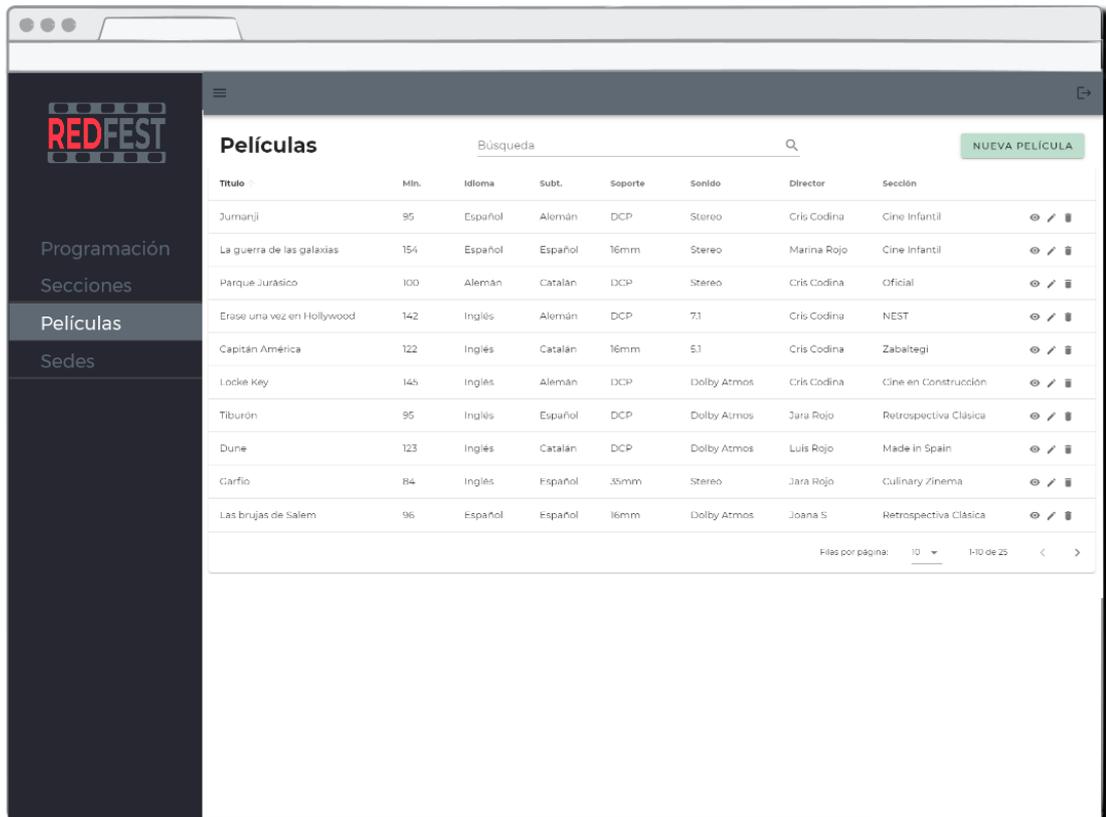


Figura 66: Vista para el departamento de películas del listado de películas

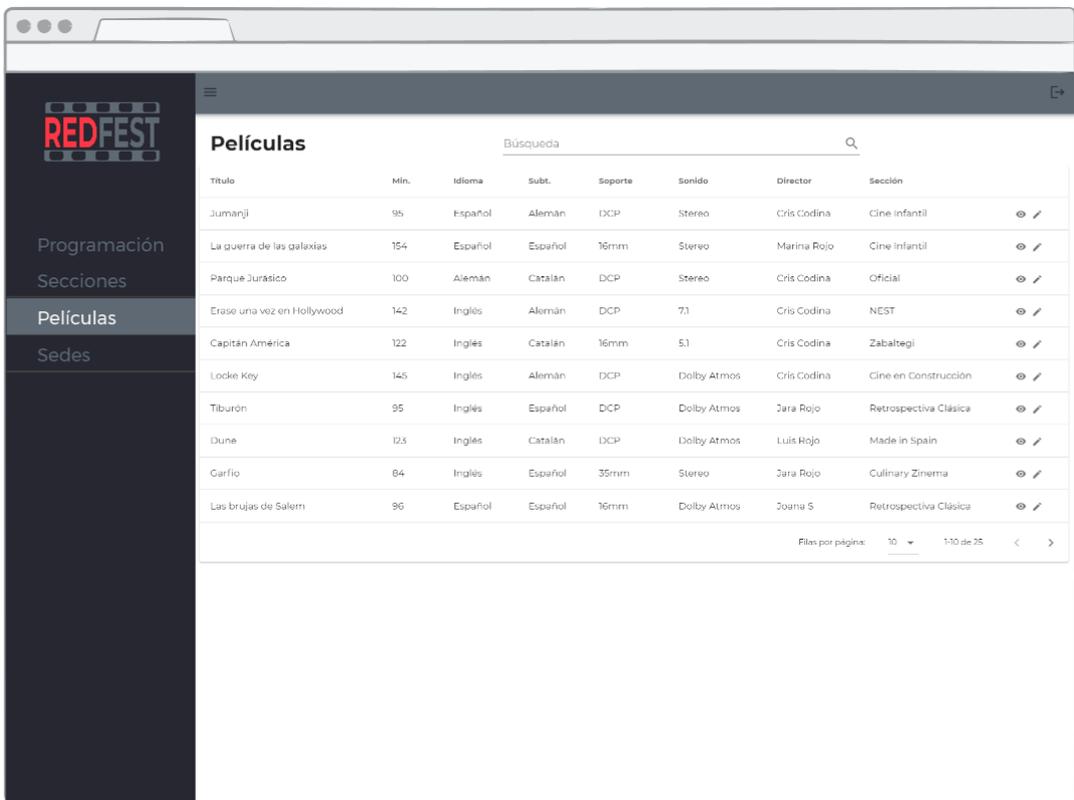


Figura 67: Vista estandar del listado de películas

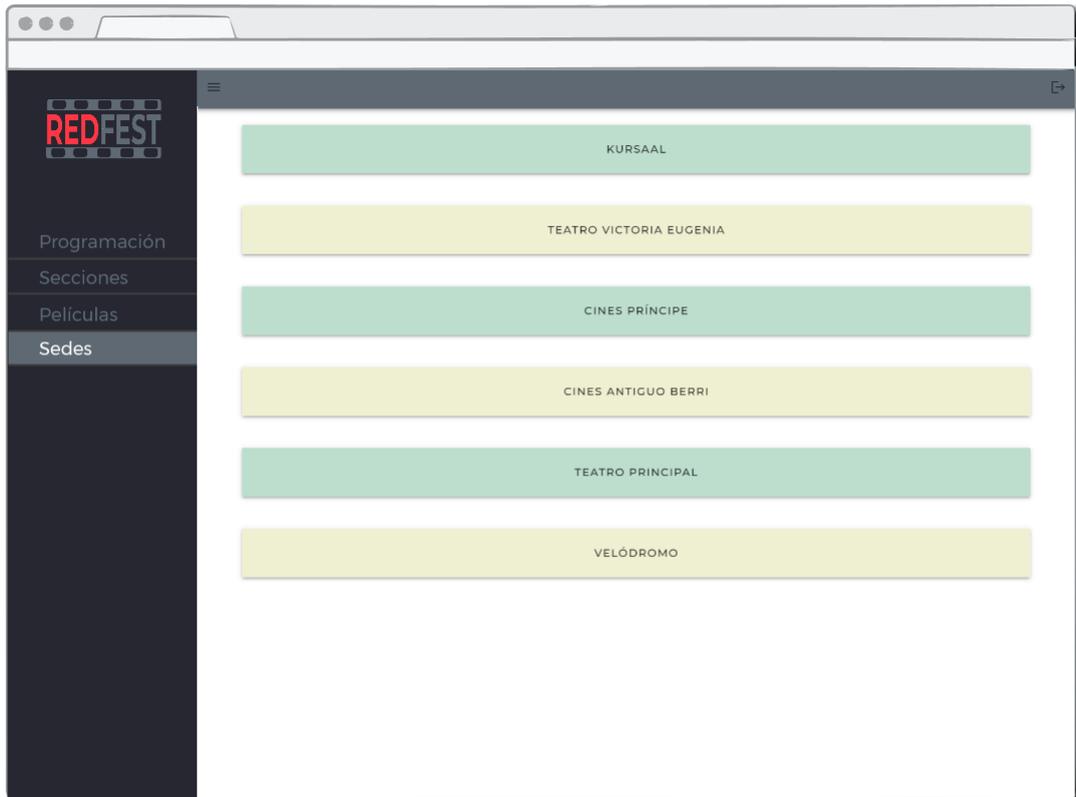


Figura 68: Vista de las sedes del festival

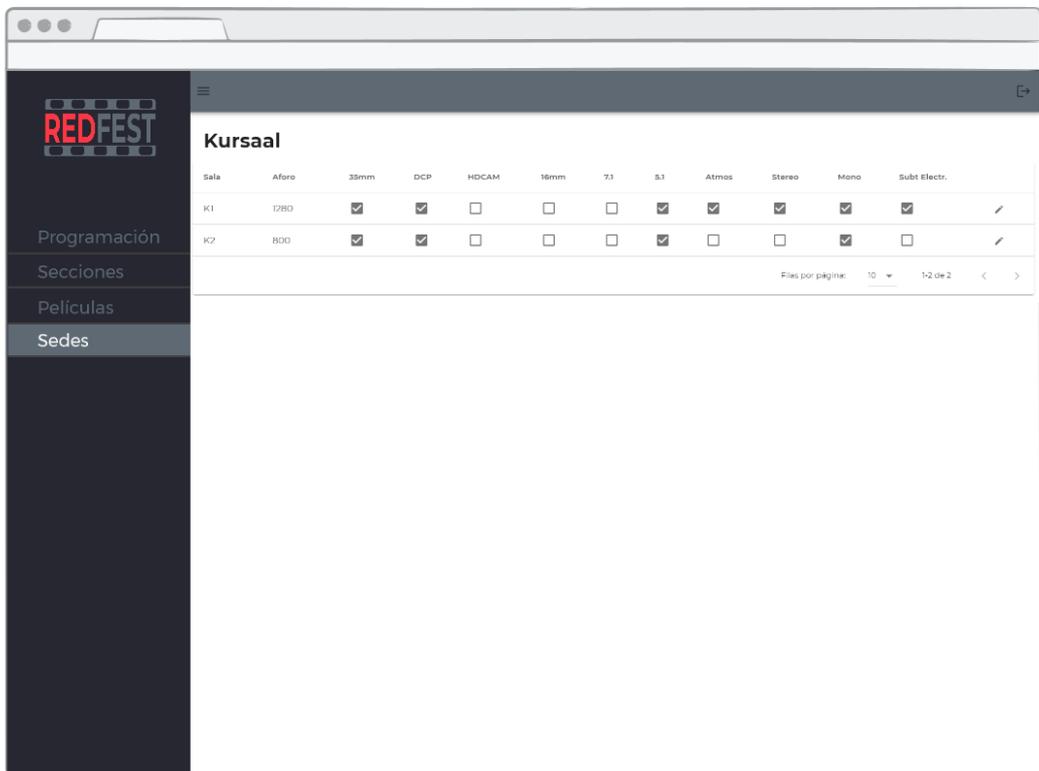


Figura 69: Vista para el departamento técnico de las salas de una sede

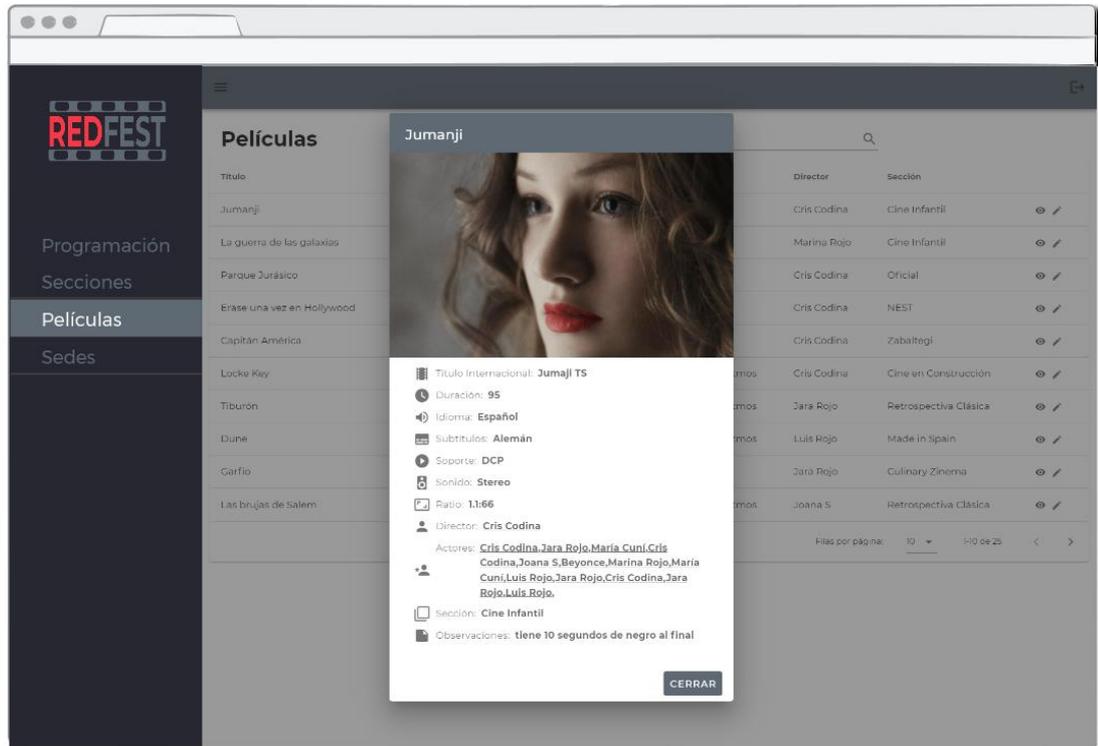


Figura 70: Ventana modal con la ficha de una película

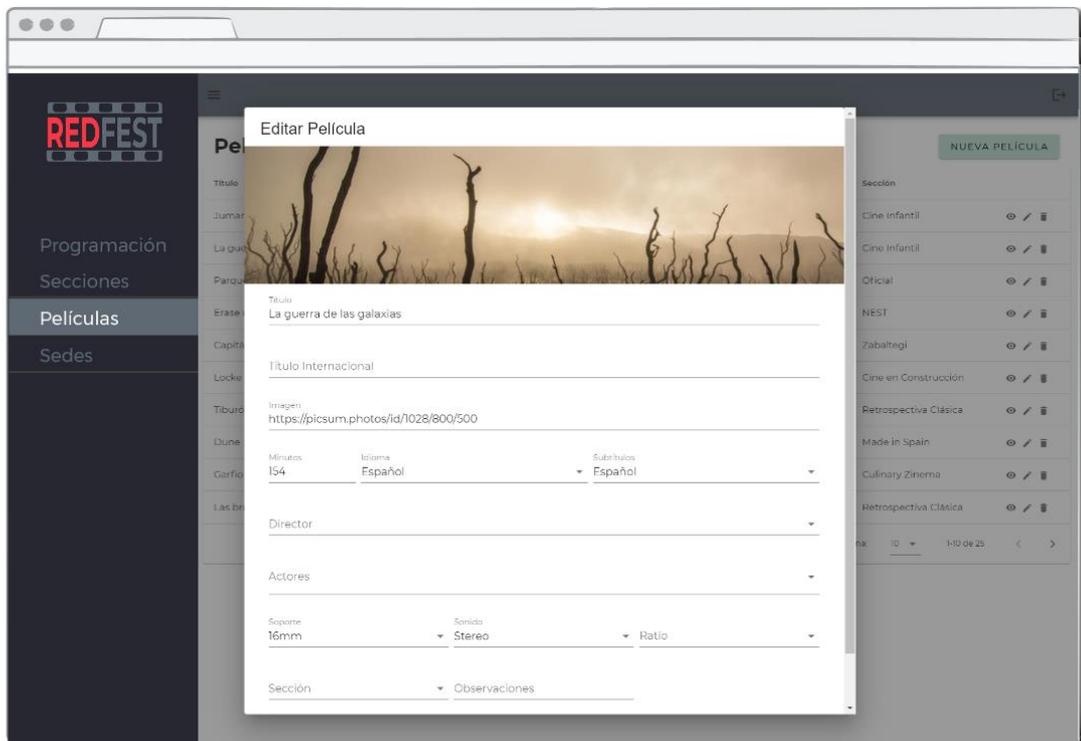


Figura 71: Ventana modal de edición para el departamento películas

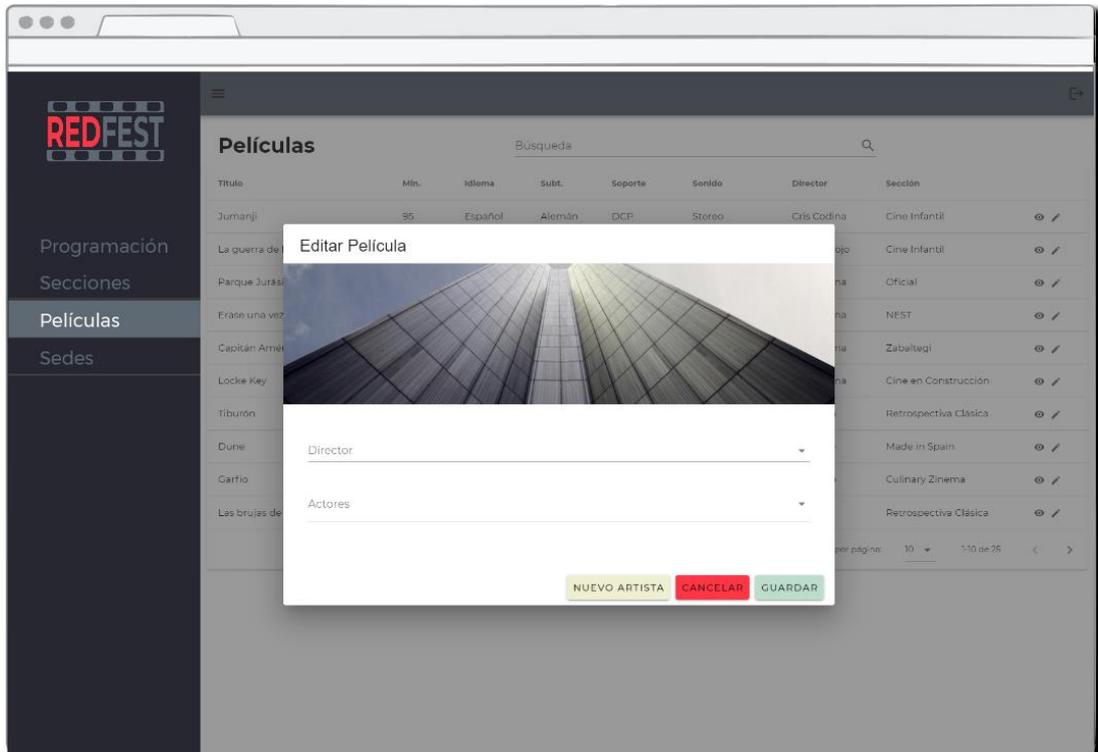


Figura 72: Ventana modal de edición para el departamento de invitados

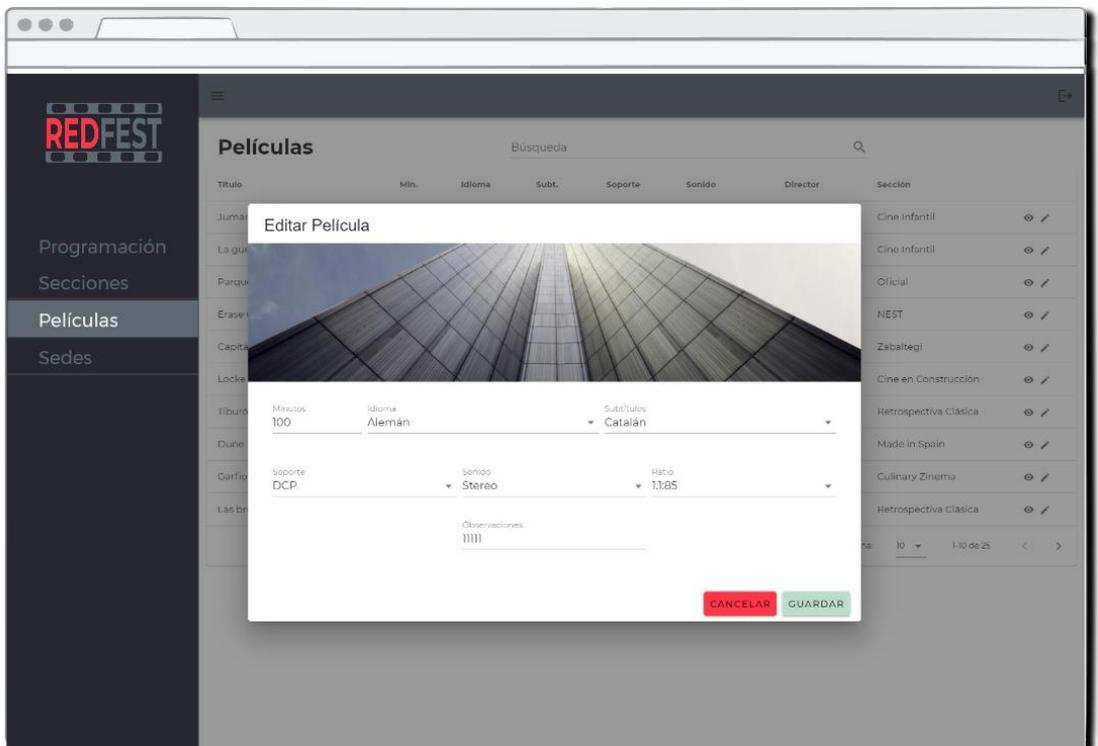


Figura 73: Ventana modal de edición para el departamento técnico

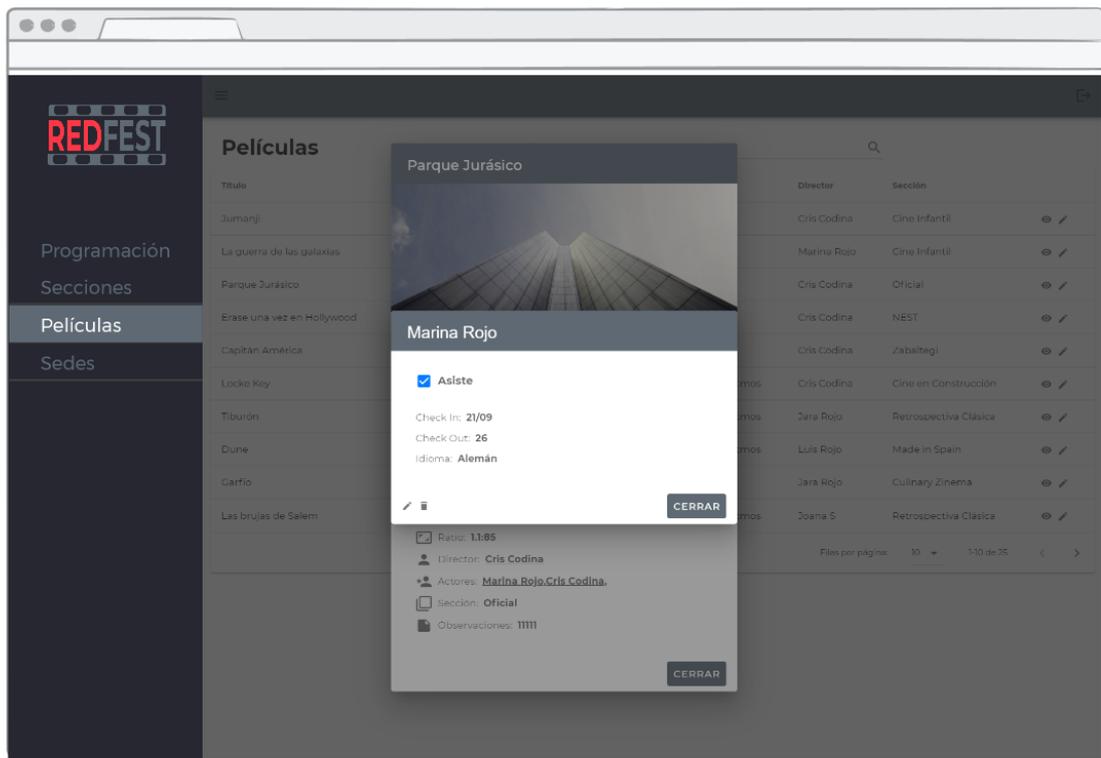


Figura 74: Ventana modal para el departamento de invitados con la ficha de un artista

6.3. Tests

Se han realizado dos tipos de tests, de usabilidad y de funcionalidad:

- Usabilidad: se ha trabajado con un *Focus Group*³⁹ al que se le han asignado diferentes tareas basadas en los posibles casos de uso de la aplicación.

Los participantes cubrían los perfiles de programador, departamento de invitados y departamento técnico indicados en la [tabla 3](#).

Las tareas que se les han indicado son las siguientes:

- Acceder a la web y hacer *login*
- Ver listado de películas y acceder a los datos de una película
- Ver listado de películas de una única sección
- Programador: crear una sesión y editarla
- Invitados: Crear un nuevo artista y editarlo
- Técnico: Ver lista de salas de una sede y editar una característica

³⁹ «Test de usabilidad de una web» <https://www.diligent.es/test-de-usabilidad-a-una-pagina-web/>; «Grupo focal». https://es.wikipedia.org/wiki/Grupo_focal

La conclusión principal positiva de esta prueba es que la aplicación se ha mantenido lo suficientemente sencilla como para que perfiles tan diversos encuentren la información y los activadores de acciones rápidamente.

Por otro lado, como aspectos a mejorar se han detectado los siguientes puntos:

- Habrá que habilitar que el acceso a la información de la película no se haga exclusivamente desde el icono correspondiente, sino que se pueda acceder *clickando* la línea de la tabla en cualquier punto.
 - Al acceder tras *loguearse*, hasta que no haya elección de ediciones, debería mostrarse la barra navegación lateral directamente.
- Funcionalidad: se han seguido todos los casos de uso habituales en un festival de cine para detectar posibles fallas en la aplicación. Los casos son los siguientes:
- Una persona del departamento de películas crea una película, edita posteriormente algunos detalles de esa película y finalmente la elimina porque no participará en el festival.
 - Una persona del departamento de invitados accede a la información de una película, crea un nuevo artista y lo asigna a esa película.
 - Una persona del departamento de invitados accede a la información de un artista y edita su información
 - Un técnico accede a una película y edita sus datos una vez revisada.
 - Un técnico accede a los datos de una sala dentro de una sede y cambia una de sus características.
 - Un programador crea una sesión con una película previamente creada por el departamento de películas. Edita la información de esa sesión y finalmente elimina la sesión.

Los bugs detectados que se corregirán próximamente son:

- Cuando se elimina una película, las sesiones se quedan sin película asignada y no se pueden eliminar.
- Cuando se crea un nuevo artista no se puede editar inmediatamente la película, hay que cerrar la ventana modal y volver a acceder.
- Al crear una sesión o una película no se pueden crear sin todos los elementos vinculados a otros modelos de la base de datos seleccionados. O bien hay que indicar la obligatoriedad de rellenar todos los datos o hay que permitir que se creen con un único dato requerido.

6.4. Ejemplos de uso del producto (o guía de usuario)

6.4.1. LOGIN y LOGOUT

Para acceder a la aplicación hay que estar registrado y para ello, nada más entrar nos aparece esta única opción:

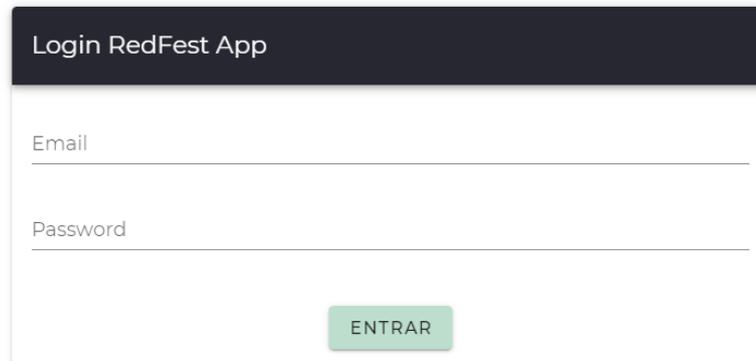
A screenshot of a login form titled "Login RedFest App". The form has a dark header with the title in white. Below the header, there are two input fields: "Email" and "Password", each with a horizontal line for text entry. At the bottom center of the form is a green button with the text "ENTRAR" in white capital letters.

Figura 75: Vista de login a la aplicación

Los datos de acceso de los diferentes perfiles creados están en el punto [5.2.7](#).

Para acceder como un usuario diferente o, simplemente, para cerrar sesión, solo hace falta *clickar* en el botón de *logout* ubicado arriba a la derecha:



Figura 76: Botón de logout

6.4.2. VER PROGRAMACIÓN

En el menú lateral se puede acceder a la primera opción "Programación" donde se pueden ver todas las sesiones creadas por el programador del festival. En este listado se pueden realizar búsquedas y ordenar por diferentes campos.



Figura 77: Menú de navegación lateral

☰
➔

Programación

Búsqueda
🔍
NUEVA SESIÓN

Título	Min.	Idioma	Subt.	Subt. Electr.	Sección	Día	Hora	Sala	Coloquio	
Capitán América	122	Inglés	Catalán	Español	Zabaltegi	22/9	21:00	PPAL	<input type="checkbox"/>	👁️ ✎️ 🗑️
Parque Jurásico	100	Alemán	Catalán	Euskera	Oficial	24/9	14:00	K1	<input checked="" type="checkbox"/>	👁️ ✎️ 🗑️
Erase una vez en Hollywood	142	Inglés	Alemán	No	NEST	25/9	19:00	AB1	<input type="checkbox"/>	👁️ ✎️ 🗑️
Jumanji	95	Español	Alemán	Español	Cine Infantil	26/9	21:00	K1	<input checked="" type="checkbox"/>	👁️ ✎️ 🗑️
Erase una vez en Hollywood	142	Inglés	Alemán	Inglés	NEST	21/9	12:00	AB2	<input type="checkbox"/>	👁️ ✎️ 🗑️
Erase una vez en Hollywood	142	Inglés	Alemán	Inglés	NEST	22/9	16:00	AB2	<input checked="" type="checkbox"/>	👁️ ✎️ 🗑️
Eduardo Manostijeras	100	Inglés	Inglés	Euskera	Perlas	23/9	16:00	K1	<input type="checkbox"/>	👁️ ✎️ 🗑️
Yí Miao Zhon	111	Japonés	Español	Inglés	Culinary Zinema	28/9	22:00	K2	<input checked="" type="checkbox"/>	👁️ ✎️ 🗑️
Parque Jurásico	100	Alemán	Catalán	No	Oficial	20/9	09:00	Victoria Eugenia	<input checked="" type="checkbox"/>	👁️ ✎️ 🗑️

Filas por página: 10 1-9 de 9 < >

Figura 78: Vista de programación como Programador

6.4.3. CREAR, MODIFICAR y ELIMINAR SESIONES

A la derecha de cada sesión hay tres iconos para poder visualizar, editar o eliminar cada sesión. La opción “ver” está habilitada para todos, las demás solo las ve el programador.



Figura 79: Iconos para ver, editar y eliminar sesiones

Para crear una nueva sesión, el programador selecciona el botón de la figura 52, donde se abre una ventana modal (figura 53) con un formulario donde introducir los datos de la sesión que se quiere crear.



Figura 80: Botón para crear una nueva sesión

Nueva Sesión

Película ▼

Subt. electr. ▼

Día ▼ Hora ▼

Sala ▼ coloquio

CANCELAR
GUARDAR

Figura 81: Formulario para crear una nueva sesión

Si lo que se quiere es modificar una sesión previamente creada, se selecciona el icono del lápiz y se abre una ventana modal con un formulario para editar los campos.

Editar Sesión

Subt. electr.
Euskera

Día
24/9

Hora
14:00

Sala

coloquio

CANCELAR **GUARDAR**

Figura 82: Formulario de edición de sesiones

Para eliminar una sesión, se selecciona el icono de la papelera y se abrirá una ventana modal con la pregunta de confirmación de la acción.

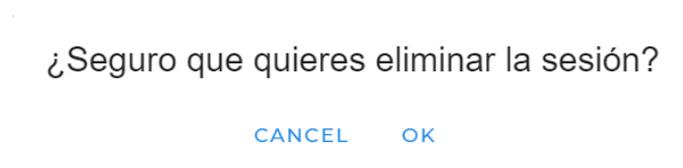


Figura 83: Pregunta de confirmación de eliminación de sesiones

6.4.4. VER PELÍCULAS

En el menú lateral se puede acceder a la sección “Películas” desde donde se puede visualizar el listado de todas las películas de ese festival. En este listado se pueden realizar búsquedas y ordenar por diferentes campos.

☰
➔

Películas

Búsqueda 🔍
NUEVA PELÍCULA

Título	Min.	Idioma	Subt.	Soporte	Sonido	Director	Sección	
Jumanji	92	Inglés	Español	DCP	7.1	Jara	Culinary Zinema	👁️ ✎️ 🗑️
La guerra de las galaxias	154	Inglés	Español	16mm	5.1	Cris	Retrospectiva Temática	👁️ ✎️ 🗑️
Parque Jurásico	100	Alemán	Español	DCP	Stereo	Joana	NEST	👁️ ✎️ 🗑️
Erase una vez en Hollywood	142	Inglés	Alemán	DCP	7.1	Artista 2	Oficial	👁️ ✎️ 🗑️
Capitán América	122	Inglés	Catalán	35mm	5.1	Marina	Perlas	👁️ ✎️ 🗑️
Locke Key	145	Inglés	Alemán	35mm	Dolby Atmos	Marina	Cine en Construcción	👁️ ✎️ 🗑️
Tiburón	95	Inglés	Español	DCP	Dolby Atmos	Jara	Retrospectiva Clásica	👁️ ✎️ 🗑️
Dune	123	Inglés	Catalán	35mm	Dolby Atmos	Cris	Perlas	👁️ ✎️ 🗑️
Garfio	84	Alemán	Español	HDCAM	Stereo	Marina	Made in Spain	👁️ ✎️ 🗑️
Las brujas de Salem	96	Inglés	Español	HDCAM	Mono	Artista 2	Made in Spain	👁️ ✎️ 🗑️

Filas por página: 10 ▼ 1-10 de 16 ⏪ ⏩

Figura 84: Vista de películas como admin o departamento película

6.4.5. CREAR, MODIFICAR y ELIMINAR PELÍCULAS

A la derecha de cada película hay tres iconos para poder visualizar, editar o eliminar cada película. Las opciones habilitadas dependen de cada tipo de usuario.



Figura 85: Iconos para ver, editar y eliminar películas

Si lo que se quiere es crear una nueva película, hay que clicar el botón “Nueva película” donde se abre una ventana modal con un formulario.



Figura 86: Botón para crear una nueva película

Figura 87: Formulario de creación de nuevas películas

Para editar los datos de una película se debe seleccionar el icono del lápiz y se abrirá una ventana modal con un formulario para editar los campos que se quieran cambiar.



Título
 La guerra de las galaxias

Título Internacional

Imagen
 https://picsum.photos/id/1028/800/500

Minutos
 154

Idioma
 Español

Subtítulos
 Español

Director

Actores

Soporte
 16mm

Sonido
 Stereo

Ratio

Sección
 Observaciones

CANCELAR GUARDAR

Figura 88: Formulario de edición de películas

Para eliminar una película, se selecciona el icono de la papelera y se abrirá una ventana modal con la pregunta de confirmación de la acción.

¿Seguro que quieres eliminar la película?

CANCEL OK

Figura 89: Pregunta de confirmación de eliminación de películas

6.4.6. VER CARACTERÍSTICAS PELÍCULAS, SESIONES Y ARTISTAS

Para ver los datos de las películas y de los artistas, hay que seleccionar el icono del ojo de las figuras 51 y 57. Se abrirá una ventana modal donde se ven los datos de la película o sesión, y se puede seleccionar un artista abriendo una nueva ventana modal con los datos del artista.

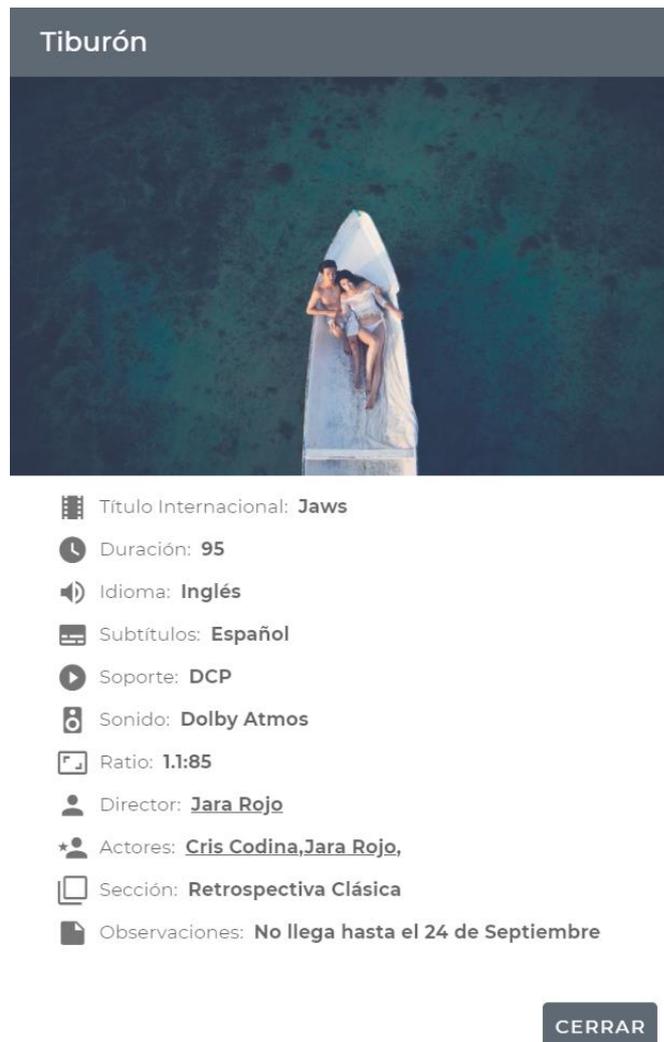


Figura 90: Ficha datos película

Capitán América



- 📺 Título Internacional:
- 🕒 Duración: 122
- 🔊 Idioma: Inglés
- 📄 Subtítulos: Catalán
- ▶ Soporte: 16mm
- 🔊 Sonido: 5.1
- 📐 Ratio: Scope
- 📁 Sección: Zabaltegi
- 📄 Observaciones: Faltan los datos técnicos
- 📅 Día: 22/9
- 🕒 Hora: 21:00
- 👤 Sala: PPAL

CERRAR

Figura 91: Ficha datos sesión

Jara Rojo

- Asiste
- Check In: 20/9
- Check Out: 22/09
- Idioma: Inglés

✎ 🗑

CERRAR

Figura 92: Ficha datos artista

En la ficha de artista, una persona del departamento de Invitados podrá editar o eliminar los datos de ese artista seleccionando los iconos del lápiz o la papelera.

Editar Jara Rojo

Nombre y Apellidos
Jara Rojo

Asiste

Check In
20/9

Check Out
22/09

Idioma
Inglés

CANCELAR GUARDAR

Figura 93: Formulario de modificación de datos de un artista

¿Seguro que quieres eliminar el artista?

CANCEL OK

Figura 94: Pregunta de confirmación de eliminación de un artista

6.4.7. FILTRAR PELÍCULAS POR SECCIÓN

Para poder filtrar las películas por sección hay que seleccionar qué sección se quiere ver en el menú desplegable del menú lateral.



Figura 95: Menú desplegable

Esta vista es igual que la de “Películas” menos las opciones de editar, crear y eliminar películas.

Nuevos Directores							Búsqueda
Titulo	Mín.	Idioma	Subt.	Soporte	Sonido	Director	
prueba	1	Español	Español	35mm	Mono	Luis Rojo	👁
Rosa Rosae	98	Español	Inglés	35mm	Dolby Atmos	María Cuní	👁
Prueba multiple	20	Inglés	Inglés	35mm	Mono	Jara Rojo	👁
Prueba 2 Multiple	213	Inglés	Español	35mm	Mono	Marina Rojo	👁

Filas por página: 10 1-4 de 4 < >

Figura 96: Listado de películas de una sección

6.4.8. VER SALAS

Para ver las características de cada sede, hay que seleccionar “Salas” en el menú lateral. En esta vista se selecciona la sede que se quiere ver.

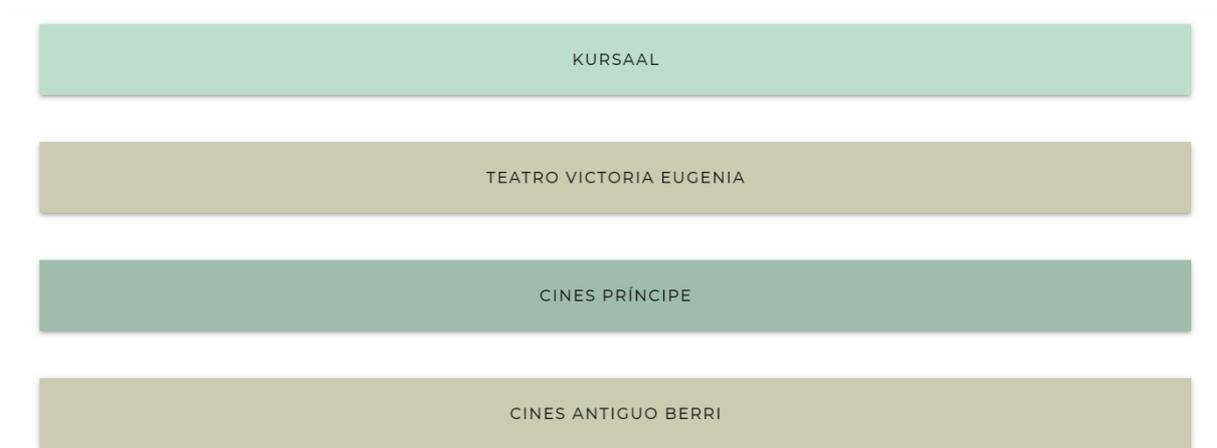


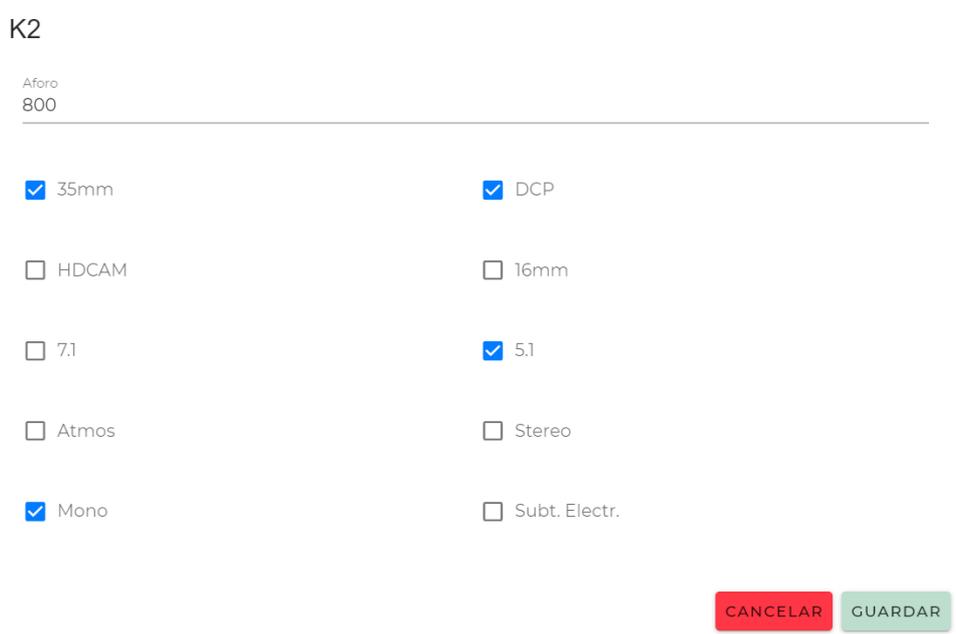
Figura 97: Menú de sedes

Una vez seleccionada una sede, se muestra una tabla con todas las salas y sus características.

Kursaal												
Sala	Aforo	35mm	DCP	HDCAM	16mm	7.1	5.1	Atmos	Stereo	Mono	Subt Electr.	
K1	1280	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	✎ 🗑				
K2	800	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	✎ 🗑

Figura 98: Listado de salas con sus características

El departamento técnico puede editar los datos de las salas seleccionando el icono del lápiz. Se abre una ventana modal con un formulario con campos de tipo *checkbox* para editar las características de dicha sala.



K2

Aforo
800

35mm DCP

HDCAM 16mm

7.1 5.1

Atmos Stereo

Mono Subt. Electr.

CANCELAR GUARDAR

Figura 99: Formulario de edición de las características de una sala

El administrador puede eliminar salas de una sede seleccionando el icono de la papelera y confirmando la pregunta.

¿Seguro que quieres eliminar la sala?

CANCEL OK

Figura 100: Pregunta de confirmación de eliminación de una sala

7. Conclusiones y líneas de futuro

7.1. Conclusiones

Tras finalizar el desarrollo de una primera versión funcional de la aplicación REDFEST puedo pasar a analizar qué objetivos se han logrado, cómo ha sido el proceso de trabajo, qué he aprendido en este proceso y cómo he sorteado los problemas encontrados.

Para comenzar, me gustaría destacar lo importante que me ha parecido el proceso de trabajo del TFG para darle sentido a todo el conjunto de asignaturas aprendidas a lo largo de varios años. A lo largo de los meses he ido retrocediendo a los apuntes y trabajos entregados en asignaturas pasadas para crear un TFG lo más completo posible.

En lo que a la organización de un proyecto de estas características respecta, me he dado cuenta de la importancia que tiene hacer una buena preparación y análisis antes de empezar a desarrollar. Esto es algo que me daba miedo al inicio del TFG, por creer que debía aprovechar cada día escribiendo código. Sin embargo, la aplicación se sustenta en otros pilares igual de importantes como son el análisis de las necesidades de los usuarios, el análisis del mercado, el listado de objetivos concretos a lograr, etc.

En este TFG he planificado cada objetivo, guiada por las PEC, calendarizándolos al detalle. Este calendario y división de tareas me ha permitido enfocarme en cada tema en su momento. De no haberlo hecho de esta manera, podría haber querido encontrar resultados visibles mucho antes, generando caos en el desarrollo y desvirtuando la intención de la aplicación.

A pesar de la correcta organización y ejecución de las tareas, el principal escollo que he encontrado han sido las tecnologías. Para comenzar, al idear el TFG, hice un pequeño trabajo de investigación sobre qué tecnologías podían ser más adecuadas para un proyecto de estas características. Con lo aprendido a lo largo del Grado, decidí trabajar con el lenguaje de programación que más conozco, Javascript, y seleccioné tecnologías, como Vue, con una curva de aprendizaje relativamente baja. Puedo afirmar que he aprendido mucho, no solo de estos *frameworks* y tecnologías, sino que he conseguido comprender el proceso de desarrollo desde el principio hasta el final, uniendo *backend* y *frontend* como no había podido probar hasta ahora. Otro aspecto que destacar de lo aprendido respecto a las tecnologías es el concepto de trabajar por componentes. Al no haber trabajado nunca una aplicación al completo no había tenido que afrontar una situación en la que la optimización es importante. Aprender el sistema de desarrollo por componentes que permite Vue, ha agilizado mucho el desarrollo y, lo que es mejor, permite que los cambios futuros sean mucho más rápidos y fáciles de realizar.

El hecho de ser la primera vez que afronto un proyecto de estas características me ha hecho encontrarme problemas de programación que me frenaban. Todos estos problemas los he ido solventando, consiguiendo los propósitos iniciales, pero me han obligado a desechar funcionalidades que me hubiera gustado implementar.

Al comienzo del TFG propuse unos objetivos principales y unos secundarios. Dejé como secundarios los que me imaginaba que podía no alcanzar debido a estos posibles problemas. Y así ha sido, he cumplido todos los objetivos principales, tanto los personales como los de la aplicación, pero no he desarrollado las funcionalidades indicadas como objetivos secundarios. A pesar de esto, he trabajado la aplicación pensando en una futura implementación de estas funcionalidades, de manera que, aunque en el resultado final no se muestre, parte del *backend* ya está avanzado.

Para finalizar con las conclusiones, destaco el gran paso que siento haber dado con todo lo aprendido estos últimos meses trabajando en el TFG. Poder haber llevado a cabo el desarrollo de esta aplicación ha sido el cierre perfecto para unificar y darle sentido a todo el trabajo realizado estos años de Grado.

7.2. Líneas de futuro

Como he comentado en el apartado anterior, los objetivos secundarios no se han podido cumplir, así que serán algunos de los próximos pasos a seguir en el desarrollo de la aplicación.

- Se trabajará la opción de que el programador cree unos parámetros para cada sección.
- Cuando estén estos parámetros creados, la aplicación creará una primera versión de sesiones de cada película. El programador seguirá pudiendo editar las sesiones para adaptarlas a lo que crea conveniente, pero esto reducirá mucho el tiempo de creación de las sesiones.
- Guardar un histórico de todas las ediciones pudiendo acceder a ellas independientemente.
- Crear un tipo de acceso diferente para distribuidoras y productoras donde puedan registrar sus películas e introducir la información requerida en un formulario. Esta información creará directamente una película en la base de datos.
- Crear la posibilidad de exportación de las vistas en PDF.

Además de cumplir estos objetivos, otro siguiente paso importante es arreglar los *bugs* detectados, y me gustaría dedicar tiempo a mejorar detalles de las funcionalidades que ya están implementadas.

Una vez la versión de la aplicación sea más sólida y estable con estas funcionalidades y mejoras, me gustaría trabajar con un equipo con el que poder presentar la aplicación a un festival, adaptarnos a sus características y abrir esa puerta en el mercado.

Bibliografía

Adobe Creative Cloud «Adobe Creative Cloud | Detalles y productos | Adobe».

<https://www.adobe.com/es/creativecloud.html>. Accedido 22 de octubre de 2021.

Au-Yeung, John. «*Vue. Js 3 by Example: Blueprints to Learn Vue Web Development, Full-Stack Development, and Cross-Platform Development Quickly*». Packt Publishing, Limited, 2021.

Camden, Raymond. «*Front-End Development Projects with Vue. Js: Learn to Build Scalable Web Applications and Dynamic User Interfaces with Vue*». Birmingham, England: Packt, 2020.

Colaboradores de Wikipedia. *Desarrollo en cascada* [en línea]. Wikipedia, La enciclopedia libre, 2021 [fecha de consulta: 12 de octubre del 2021]. Disponible en

https://es.wikipedia.org/w/index.php?title=Desarrollo_en_cascada&oldid=137652816.

Colaboradores de Wikipedia. *Diseño centrado en el usuario* [en línea]. Wikipedia, La enciclopedia libre, 2020 [fecha de consulta: 21 de octubre del 2021]. Disponible en

https://es.wikipedia.org/w/index.php?title=Dise%C3%B1o_centrado_en_el_usuario&oldid=126520734

Colaboradores de Wikipedia. *Grupo focal* [en línea]. Wikipedia, La enciclopedia libre, 2021 [fecha de consulta: 23 de diciembre del 2021]. Disponible en

https://es.wikipedia.org/w/index.php?title=Grupo_focal&oldid=140610935.

Colaboradores de Wikipedia. *Jakob Nielsen* [en línea]. Wikipedia, La enciclopedia libre, 2020 [fecha de consulta: 21 de octubre del 2021]. Disponible en

https://es.wikipedia.org/w/index.php?title=Jakob_Nielsen&oldid=125536495

Colaboradores de Wikipedia. *Película de 35 mm* [en línea]. Wikipedia, La enciclopedia libre, 2021 [fecha de consulta: 3 de octubre del 2021]. Disponible en

https://es.wikipedia.org/w/index.php?title=Pel%C3%ADcula_de_35_mm&oldid=135993561.

Colaboradores de Wikipedia. *Single-page application* [en línea]. Wikipedia, La enciclopedia libre, 2021 [fecha de consulta: 3 de octubre del 2021]. Disponible en

https://es.wikipedia.org/w/index.php?title=Single-page_application&oldid=134844163.

Coolors.co. «Create a Palette - Coolors». <https://coolors.co/> Accedido 14 de octubre de 2021.

Diagrams.net. <https://app.diagrams.net/>. Accedido 22 de octubre de 2021.

Digivents «Registro, check-in y app para eventos y ferias». <https://www.digivents.com/es/>. Accedido 30 de diciembre de 2021.

Diligent. «Test de usabilidad de una web: para qué sirve y cómo hacerlo paso a paso», 20 de agosto de 2018. <https://www.diligent.es/test-de-usabilidad-a-una-pagina-web/>.

Express «Infraestructura de aplicaciones web Node.js». <https://expressjs.com/es/>. Accedido 3 de octubre de 2021.

Festival de San Sebastián. <https://www.sansebastianfestival.com:443/es/>. Accedido 3 de octubre de 2021.

FestivalPro <https://www.festivalpro.com/>. Accedido 30 de diciembre de 2021.

- GeeksforGeeks-bhavikbhatia17** «What is MEVN Stack ? - GeeksforGeeks». <https://www.geeksforgeeks.org/what-is-mevn-stack/>. Accedido 30 de octubre de 2021.
- Git**. <https://git-scm.com/>. Accedido 30 de diciembre de 2021.
- GitHub**. «Build Software Better, Together». <https://github.com>. Accedido 30 de diciembre de 2021.
- Google Fonts** <https://fonts.google.com/>. Accedido 15 de octubre de 2021.
- Google Fonts** «Material Icons Guide | Google Fonts | Google Developers». https://developers.google.com/fonts/docs/material_icons?hl=es. Accedido 17 de noviembre de 2021.
- Hassan Montero, Yusef, y Sergio Ortega Santamaría**. «Informe APEI sobre usabilidad». <http://www.nosolousabilidad.com/manual/3.htm>. Accedido 21 de octubre de 2021.
- IBM Docs**, <https://prod.ibmdocs-production-dal-6099123ce774e592a519d7c33db8265e-0000.us-south.containers.appdomain.cloud/docs/es/baw/20.x?topic=formats-javascript-object-notation-json-format>. 24 de junio de 2021.
- INE**. «Porcentaje de adultos (16 a 74 años) que en los últimos 12 meses han realizado tareas relacionadas con conocimientos informáticos por sexo, grupos de edad y tipo de tarea(46292)». <https://www.ine.es/jaxiT3/Datos.htm?t=46292#!tabs-grafico>. Accedido 7 de octubre de 2021.
- Just in mind** «Free prototyping tool for web & mobile apps ». <https://www.justinmind.com/> Accedido 22 de octubre de 2021.
- Leider, John** *vuetifyjs/vuetify*. TypeScript. 2016. Reprint, vuetify, 2021. <https://github.com/vuetifyjs/vuetify/blob/c5dacb6bcfef07f54c159e467c3f6bb3173591e4/packages/docs/src/examples/wireframes/base.vue>
- Lennd** - «Live & Hybrid Event Operations Platform». <https://www.lennd.com/>. Accedido 18 de octubre de 2021.
- Local Fonts**. «Montserrat». <https://localfonts.eu/freetype/bulgarian-cyrillic/montserrat/>. Accedido 20 de octubre de 2021.
- Mateo, Jorge** «Las 10 reglas heurísticas de usabilidad de Jakob Nielsen | Diseño grafico, UI, UX y arquitectura de la información Barcelona». <https://designbarcelona.jorgemateo.es/las-10-reglas-heuristicas-de-usabilidad-de-jakob-nielsen/>. Accedido 21 de octubre de 2021.
- Material Design** <https://material.io/design/color/the-color-system.html#color-theme-creation>. Accedido 17 de noviembre de 2021.
- Material Design** Accedido 17 de noviembre de 2021. <https://material.io/design/typography/the-type-system.html#type-scale>.
- MDN Web Docs** «Generalidades Del Protocolo HTTP - HTTP | MDN». 2021. <https://developer.mozilla.org/es/docs/Web/HTTP/Overview>. Accedido 3 de octubre de 2021.
- MongoDB**. «La base de datos líder del mercado para aplicaciones modernas». <https://www.mongodb.com/es>. Accedido 30 de diciembre de 2021.
- MongoDB**. «MongoDB Community Download». <https://www.mongodb.comtry>. Accedido 30 de diciembre de 2021.

- MongoDB.** «Download MongoDB Command Line Database Tools». <https://www.mongodb.com/try>.
Accedido 30 de diciembre de 2021.
- Node.js.** «Acerca». Node.js. <https://nodejs.org/es/about/>. Accedido 3 de octubre de 2021.
- Node.js** <https://nodejs.org/es/>. Accedido 30 de diciembre de 2021.
- Oreilly** *Vue.Js 3 By Example*. https://learning.oreilly.com/library/view/vue-js-3-by/9781838826345/B14405_FM_ePub_RK.xhtml. Accedido 22 de septiembre de 2021.
- Pexels** «Licencia de fotos y vídeos de stock gratuitos». <https://www.pexels.com/es-es/license/>.
Accedido 30 de diciembre de 2021.
- Postman.** «Postman API Platform ». <https://www.postman.com/>. Accedido 30 de diciembre de 2021.
- Pulido, Nicolás Mateo Arias.** «VueJS Crear Una Aplicación Con Múltiples Layouts.» *Medium* (blog), 10 de julio de 2020. <https://nmariasdev.medium.com/vue-js-crear-una-aplicaci%C3%B3n-con-m%C3%BAltiples-layouts-bcfdede95ba6>.
- SIL** «SIL Open Font License (OFL)». https://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=OFL.
Accedido 30 de diciembre de 2021.
- Tribalyte Technologies.** «¿Qué es una API REST? Características y usos de las APIs REST», 24 de mayo de 2021. <https://tech.tribalyte.eu/blog-que-es-una-api-rest>.
- Visual Studio Code** « Code Editing. Redefined». <https://code.visualstudio.com/>. Accedido 30 de diciembre de 2021.
- Vue.Js.** <https://vuejs.org/>. Accedido 3 de octubre de 2021.
- Vuetify.** «Divider Component». <https://vuetifyjs.com/en/components/dividers/>. Accedido 19 de noviembre de 2021.
- Vuetify.** « A Material Design Framework for Vue.Js». <https://vuetifyjs.com/>. Accedido 2 de diciembre de 2021.

Anexos

Anexo A: Glosario

API (Application Programming Interface): Interfaz de programación de aplicaciones que establece la comunicación entre capas de un software

API REST: API diseñada con los fundamentos REST (Representational State Transfer- Transferencia de Estado Representacional)

BACKEND: capa que se ejecuta del lado del servidor que procesa la información para alimentar al frontend

BUG: error en el código de un software

CHECKBOX: campo de un formulario en forma de caja, que puede estar activado o desactivado

DAFO: tipo de análisis que estudia las debilidades, amenazas, fortalezas y oportunidades de un proyecto

DCU (Diseño centrado en el usuario): es una metodología que sitúa al usuario como elemento principal a la hora de diseñar.

DEPLOY: término que se utiliza para referirse al despliegue de una aplicación.

ENDPOINTS: las URLs de una API que responden a una petición.

EXPRESS: framework que facilita el trabajo con Node

FOCUS GROUP: es una metodología que se utiliza para analizar el feedback de un grupo de usuarios.

FRONTEND: capa del lado del cliente desde donde se hacen peticiones al backend y se encarga de mostrar los resultados en el navegador.

FULL STACK: es un conjunto de las tecnologías que conforman todo el desarrollo de una aplicación

FRAMEWORK: es un marco de trabajo que provee de funcionalidades más sencillas para utilizar tecnologías complejas

HTTP(Hypertext Transfer Protocol): es un protocolo de transmisión de documentos que nos permite hacer solicitudes desde el frontend al backend.

JAVASCRIPT: lenguaje de programación que se utiliza para el desarrollo web tanto en el lado del cliente como en el lado del servidor, aunque inicialmente nació como un lenguaje solo para el lado del cliente.

JSON (Javascript Object Notation): Es un formato de datos en el que se transfieren los objetos de Javascript.

MATERIAL DESIGN: son una serie de normas de diseño visual creadas por Google.

MEVN: representa el stack formado por Mongo, Express, Vue y Node

MONGO: es un sistema de gestión de bases de datos NoSQL.

NODE: es un entorno en tiempo de ejecución para el lado del servidor que se trabaja en Javascript.

NOSQL: tipo de base de datos basado en esquemas flexibles.

PEC: Prueba de evaluación continua del grado de la UOC

SPA (Single Page Application): aplicación web que solo carga el contenido necesario en cada momento en lugar de recargar la página entera.

TFG: Trabajo de fin de grado

UML: Lenguaje unificado de modelado de software que sirve para esquematizar la creación de un software.

UR (Uniform Resource Locator): es una dirección única que se vincula a un recurso en la web

VUE: framework de Javascript para el lado del cliente.

XML (eXtensive Markup Language): lenguaje que almacena datos de forma legible.

Anexo B: Entregables del proyecto

El proyecto se ha entregado en un único archivo zip con la siguiente estructura de archivos y carpetas:

- Carpeta Proyecto > RedFest App
 - o Carpeta API: Contiene los archivos que conforman la API REST de la aplicación, es decir, toda la lógica del backend
 - o Carpeta VUE: Contiene los archivos que conforman el frontend de la aplicación
- Carpeta Documentos:
 - o Documento en PDF con la memoria del TFG:
PEC_FINAL_mem_RojoUrquiola_Marina
 - o Informe de autoevaluación: PEC_FINAL_autoevaluacion_RojoUrquiola_Marina
- Carpeta Presentaciones:
 - o Presentación en PDF: PEC_FINAL_prs_RojoUrquiola_Marina
 - o Vídeo de presentación: PEC_FINAL_vid_RojoUrquiola_Marina
- Archivos de trabajo:
 - o Proyectos de Illustrator
 - o Carpeta dump
 - o Prototipos

Anexo C: Capturas de pantalla

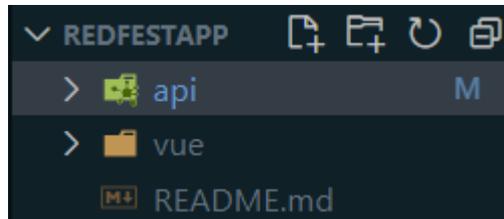


Figura 101: Anexo C: Estructura base de la app

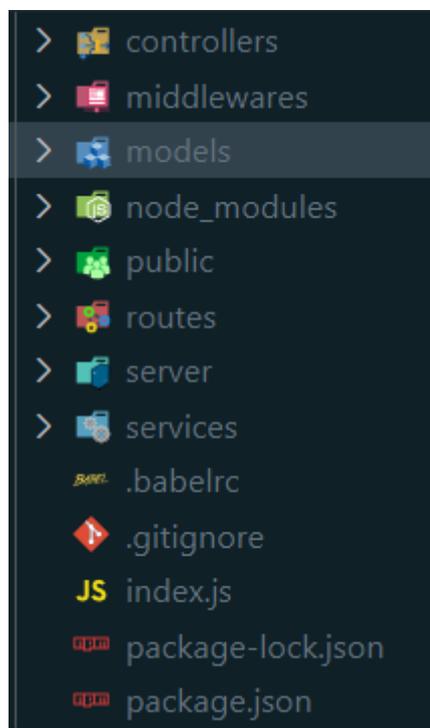


Figura 102: Anexo C:Estructura base del backend

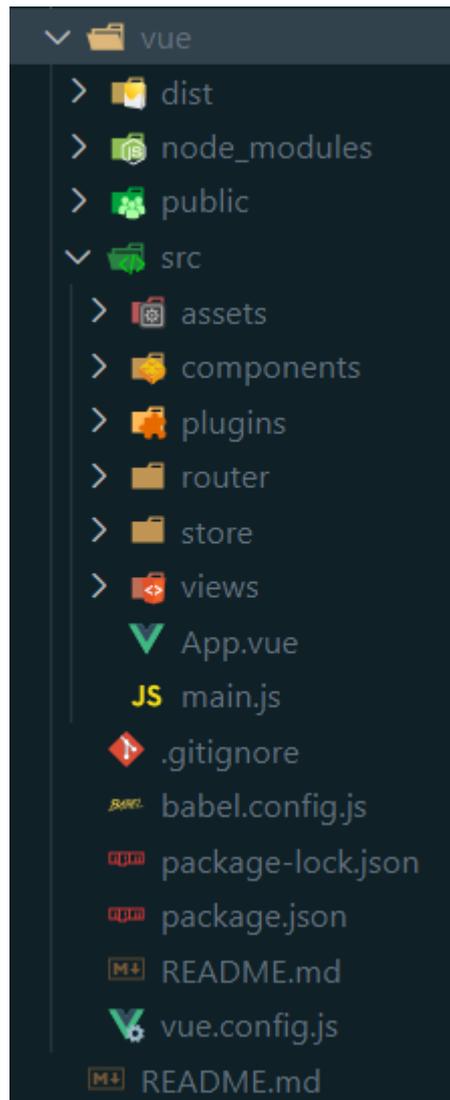


Figura 103: Anexo C:Estructura base del frontend

```
import express from 'express';
import morgan from 'morgan';
import cors from 'cors';
import path from 'path';
import mongoose from 'mongoose';
import router from './routes';

//Conexión a la base de datos MongoDB
mongoose.Promise=global.Promise;
//const dbUrl= 'mongodb://localhost:27017/dbredfest';
const dbUrl='mongodb+srv://marinarojo:redfestapp@redfest.zthx9.mongodb.net/dbredfest?retryWrites=true&w=majority'
mongoose.set('useCreateIndex', true);
mongoose.connect(dbUrl,{ useNewUrlParser:true, useFindAndModify:true, useUnifiedTopology: true})
.then(mongoose=>console.log("Conectando con la base de datos en el puerto 27017"))
.catch(err => console.log(err));
const app=express();
app.use(morgan('dev'));
app.use(cors());

app.use(express.json());
app.use(express.urlencoded({extended:true}));
app.use(express.static(path.join(__dirname, 'public')));

app.use('/api',router);
//asignar puerto por el sistema
app.set('port',process.env.PORT || 3000);

app.listen(app.get('port'),()=>{
  console.log('server on port '+ app.get('port'));
});
```

Figura 104: Anexo C: Index del backend

```
import mongoose,{Schema} from 'mongoose';
const idiomas=require('./idioma');
const ratio=require('./ratio');
const soporte=require('./soporte');
const sonido=require('./sonido');

const peliculaSchema= new Schema({
  tituloOriginal:{type:String},
  tituloSecundario:{type:String},
  duracion:{type:Number},
  idioma:{type:String, enum: idiomas},
  subtitulos:{type:String, enum: idiomas},
  soporte:{type:String, enum: soporte},
  ratio:{type:String, enum: ratio},
  sonido:{type:String, enum: sonido},
  director:{type:mongoose.Schema.Types.ObjectId, ref: 'artista'},
  actores:[{type:mongoose.Schema.Types.ObjectId, ref: 'artista'}],
  seccion:{type:mongoose.Schema.Types.ObjectId, ref: 'seccion'},
  numPases:{type:Number},
  numCopias:{type:Number},
  observaciones:{type:String},
  imagen:{type:String}
});

const Pelicula = mongoose.model('pelicula',peliculaSchema);
export default Pelicula;
```

Figura 105: Anexo C:Ejemplo de creación de un modelo en el backend. En este caso, el modelo Película

```

import models from '../models';

export default {
  add:async (req,res,next)=>{
    try{
      const reg=await models.Pelicula.create(req.body);
      res.status(200).json(reg);
    }catch(e){
      res.status(500).send({
        message: "Ha habido un error"
      });
      next(e);
    }
  },
},

```

Figura 106: Anexo C: Extracto de un controlador en el backend. En este caso, el controlador de Película con uno de sus métodos.

```

import routerx from 'express-promise-router';
import peliculaController from '../controllers/PeliculaController';
import auth from '../middlewares/auth';

const router=routerx();

router.post('/',auth.verifyFilms, peliculaController.add);
router.get('/:id',auth.verifyOtros, peliculaController.query);
router.get('/',auth.verifyOtros, peliculaController.list);
router.put('/:id',auth.verifyOtros, peliculaController.update);
router.delete('/:id', peliculaController.remove);

export default router;

```

Figura 107: Anexo C:Ejemplo de router del backend. En este caso, el router de Película donde se vincula el controlador con el router principal de la API, y se controlan los accesos.

```
import routerx from 'express-promise-router';
import festivalRouter from './festival';
import artistaRouter from './artista';
import peliculaRouter from './pelicula';
import calendarioRouter from './calendario';
import edicionRouter from './edicion';
import salaRouter from './sala';
import seccionRouter from './seccion';
import sedeRouter from './sede';
import sesionRouter from './sesion';
import staffRouter from './staff';

const router=routerx();

router.use('/festival',festivalRouter);
router.use('/artista',artistaRouter);
router.use('/pelicula',peliculaRouter);
router.use('/calendario',calendarioRouter);
router.use('/edicion',edicionRouter);
router.use('/sala',salaRouter);
router.use('/seccion', seccionRouter);
router.use('/sede',sedeRouter);
router.use('/sesion',sesionRouter);
router.use('/staff',staffRouter);
export default router;
```

Figura 108: Anexo C: Archivo router del backend donde se derivan a las rutas correspondientes al recibir una petición en la API.

```

import jwt from 'jsonwebtoken';
import models from '../models';

async function checkToken(token){
  let __id=null;
  try{
    const{_id}=await jwt.decode(token);
    __id=__id;
  }catch(e){
    return false;
  }
  const staff= await models.Staff.findOne({_id:__id});
  if(staff){
    const token = jwt.sign({_id:__id},'GloriaSwansonSSIFF1edicion',{expiresIn:'30d'});
    return {token,departamento:staff.departamento};
  }else{
    return false;
  }
}
}

```

Figura 109: Anexo C:Extracto del archivo en el que se codifica, decodifica y chequea el token de autenticación.

```

export default {
  encode: async(_id,departamento,email)=> {
    const token= jwt.sign({_id:_id,departamento:departamento,email:email},'GloriaSwansonSSIFF1edicion',{expiresIn:'30d'});
    return token;
  },

  decode: async(token)=>{
    try{
      const {_id}= await jwt.verify(token,'GloriaSwansonSSIFF1edicion');
      const staff=await models.Staff.findOne({_id});
      if(staff){
        return staff;
      }else{
        return false;
      }
    }catch(e){
      const newToken=await checkToken(token);
      return newToken;
    }
  }
}

```

Figura 110: Anexo C: Extracto del archivo en el que se codifica, decodifica y chequea el token de autenticación. En este caso, las funciones de creación del token y de decodificación del token

```
import tokenService from '../services/token';

export default {
  verifyIT: async (req, res, next) => {
    if (!req.headers.token){
      return res.status(404).send({
        message: 'No token'
      });
    }
    const response=await tokenService.decode(req.headers.token);
    if (response.departamento =='IT'){
      next();
    } else{
      return res.status(403).send({
        message: 'No autorizado'
      });
    }
  },
}
```

Figura 111: Anexo C: Extracto del archivo en el que se chequean las autorizaciones. En este caso, se verifica si hay token y si el usuario pertenece al departamento de IT.

```

"scripts": {
  "start": "babel-node index.js --exec",
  "dev": "nodemon index.js --exec babel-node"
},
"keywords": [],
"author": "",
"license": "ISC",
"devDependencies": {
  "nodemon": "^2.0.14"
},
"dependencies": {
  "babel-cli": "^6.26.0",
  "babel-preset-env": "^1.7.0",
  "babel-preset-stage-3": "^6.24.1",
  "bcryptjs": "^2.4.3",
  "cors": "^2.8.5",
  "express": "^4.17.1",
  "express-promise-router": "^4.1.0",
  "jsonwebtoken": "^8.5.1",
  "mongoose": "^5.13.13",
  "morgan": "^1.10.0"
}

```

Figura 112: Anexo C: Extracto del archivo package.json donde se indican las dependencias y las dependencias de desarrollo.

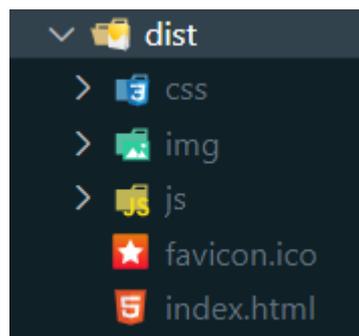


Figura 113: Anexo C: Carpeta con los archivos generados al construir el frontend. Estos archivos se copian en la carpeta public de la API que se despliega en la red, en este caso en Heroku.

```
"scripts": {
  "serve": "vue-cli-service serve",
  "build": "vue-cli-service build"
},
"dependencies": {
  "axios": "^0.24.0",
  "core-js": "^3.6.5",
  "jwt-decode": "^3.1.2",
  "vue": "^2.6.11",
  "vue-router": "^3.2.0",
  "vuetify": "^2.4.0",
  "vuex": "^3.4.0"
},
"devDependencies": {
  "@mdi/font": "^6.5.95",
  "@vue/cli-plugin-babel": "~4.5.0",
  "@vue/cli-plugin-router": "~4.5.0",
  "@vue/cli-plugin-vuex": "~4.5.0",
  "@vue/cli-service": "~4.5.0",
  "material-design-icons-iconfont": "^6.1.1",
  "sass": "~1.32.0",
  "sass-loader": "^10.0.0",
  "vue-cli-plugin-vuetify": "~2.4.3",
  "vue-template-compiler": "^2.6.11",
  "vuetify-loader": "^1.7.0"
},
```

Figura 114: Anexo C: Extracto del archivo package.json donde se indican las dependencias y las dependencias de desarrollo.

```

import Vue from 'vue'
import App from './App.vue'
import router from './router'
import store from './store'
import vuetify from './plugins/vuetify'
import axios from 'axios'
Vue.config.productionTip = false
//axios.defaults.baseURL='http://localhost:3000/api/';
axios.defaults.baseURL='https://redfest.herokuapp.com/api/';
new Vue({
  router,
  store,
  vuetify,
  render: h => h(App)
}).$mount('#app')

```

Figura 115: Anexo C: Archivo principal del frontend que indica la conexión con la API e importa todo lo necesario.

```

<template>
  <v-app id="inspire" class="change-font">
    <v-navigation-drawer
      v-model="drawer"
      v-if="logueado"
      app
      color='principal2'
    >
      <router-link to="/ediciones"><v-img src="@/assets/logo_neg.png" style="width: 180px;height:auto; margin:auto;margin-top:5%;"/></router-link>
    <v-list dense flat>
      <v-list-item :to="{name:'programacion'}" style="margin:auto;color:white;" class="mt-10">
        <v-list-item-content>
          Programación
        </v-list-item-content>
      </v-list-item>
    </v-list>
  </v-app>
</template>

```

Figura 116: Anexo C: Extracto de archivo principal de la aplicación donde se escriben los elementos que siempre tienen que estar cargados.

```
import Vue from 'vue'
import VueRouter from 'vue-router'
import store from '../store/index'
import Ediciones from '../views/Home.vue'
import Programacion from '../components/Programacion.vue'
import Peliculas from '../components/Peliculas.vue'
import Sedes from '../components/Salas.vue'
import Login from '../components/Login.vue'
import Seccion from '../components/Seccion.vue'
import Salas from '../components/DetalleSalas.vue'

Vue.use(VueRouter)

const routes = [
  {
    path: '/ediciones',
    name: 'Home',
    component: Ediciones,
    meta: {
      IT: true,
      Programador: true,
      Tecnica: true,
      Films: true,
      Invitados: true,
      Produccion: true,
      Marketing: true,
      Direccion: true,
      Prensa: true,
      Traslados: true
    }
  },
],
```

Figura 117: Anexo C: Extracto del archivo que maneja las rutas del frontend

```
router.beforeEach((to, from, next) => {
  if (to.matched.some(record => record.meta.libre)) {
    next()
  } else if (store.state.staff && store.state.staff.departamento == 'IT') {
    if (to.matched.some(record => record.meta.IT)) {
      next();
    }
  } else if (store.state.staff && store.state.staff.departamento == 'Programador') {
    if (to.matched.some(record => record.meta.Programador)) {
      next();
    }
  } else if (store.state.staff && store.state.staff.departamento == 'Films') {
    if (to.matched.some(record => record.meta.Films)) {
      next();
    }
  }
}
```

Figura 118: Anexo C: Extracto de la función que controla si se está logueado.

```
export default new Vuex.Store({
  state: {
    token:null,
    staff:null
  },
  mutations: {
    setToken(state,token){
      state.token=token;
    },
    setStaff(state,staff){
      state.staff=staff;
    }
  },
  actions: {
    guardarToken({commit},token){
      commit("setToken",token)
      commit("setStaff",decode(token))
      localStorage.setItem("token",token)
    },
    autoLogin({commit}){
      let token=localStorage.getItem("token");
      if(token){
        commit("setToken",token)
        commit("setStaff",decode(token));
      }
      router.push({name:'home'});
    },
  },
});
```

Figura 119: Anexo C:Extracto de código que controla el estado de logueo y ejecuta las acciones de guardado del token, de autologin y de desloguearse.

```
import Vue from 'vue';
import Vuetify from 'vuetify/lib/framework';
import es from 'vuetify/lib/locale/es';

Vue.use(Vuetify);

export default new Vuetify({
  theme: {
    options: {
      customProperties: true,
    },
    themes: {
      light: {
        primary: '#007BFF',
        secondary: '#424242',
        accent: '#82B1FF',
        error: '#FF5252',
        info: '#2196F3',
        success: '#4CAF50',
        warning: '#FFC107',
        principal: '#FF3645',
        principal2: '#262730',
        verde: '#BDDECD',
        amarillo: '#FFF0D1',
        terciario: '#5E6973'
      }
    }
  },
  lang: {
    locales: { es },
    current: 'es',
  }
});
```

Figura 120: Anexo C: Extracto de código que importa la librería Vuetify para manejar la parte estética de la aplicación

```
<template>
  <v-layout align-start>
    <v-flex>
      <v-toolbar flat color="white">
        <v-toolbar-title><h2>Películas</h2></v-toolbar-title>

        <v-spacer></v-spacer>
        <v-text-field class="text-xs-center" v-model="search" append-icon="search"
          label="Búsqueda" single-line hide-details></v-text-field>
        <v-spacer></v-spacer>
        <v-dialog v-model="dialog" persistent max-width="900px" transition="dialog-bottom-transition" >
          <template v-slot:activator="{ on }">
            <v-btn color="verde" v-on="on" v-if="esPeliculas || esAdmin">Nueva Película</v-btn>
          </template>
          <v-card>
            <v-card-title>
              <span class="headline">{{ formTitle }}</span>
            </v-card-title>
            <v-img
              v-if="nuevoRegistro"
              height="205px"
              :src="imagen"
            ></v-img>
          </v-card>
        </v-dialog>
      </v-flex>
    </v-layout>
  </template>
```

Figura 121: Anexo C: Extracto de código de creación de uno de los componentes.

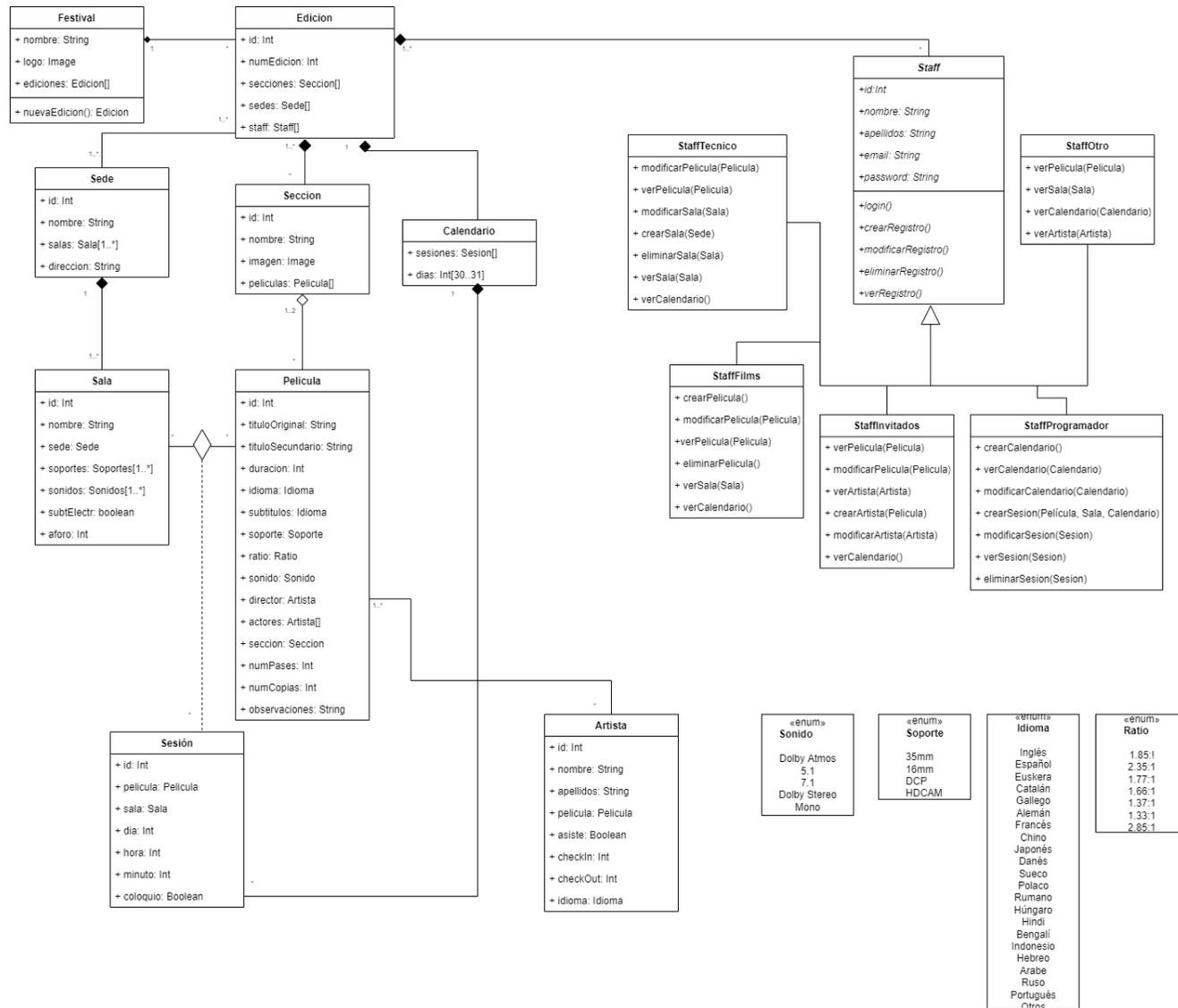


Figura 122: Anexo C: Diagrama UML completo

Anexo D: Currículum Vitae

Marina Rojo Urquiola, nacida en San Sebastián en 1986, está graduada en Dirección de Cine por el Centre d'Estudis Cinematogràfics de Catalunya, y también tiene el Grado Superior en Realización y Producción de proyectos audiovisuales y espectáculos.

Ha dedicado 15 años de su carrera como coordinadora técnica a festivales de cine tales como el Festival Internacional de Cine de San Sebastián, el Festival de Cine Europeo de Sevilla, la SEMINCI de Valladolid o el Festival Internacional de Cine de Las Palmas. Ha compaginado estas labores con las de ayudante de dirección en teatro con sus propias obras y la compañía catalana LaminimaL, en obras como "El suicidio del elefante hipotecado", "La supervivencia de las luciérnagas", "La grandeza de ser uno entre tantos" y "Dimecres".

Los últimos 5 años los ha dedicado a adentrarse laboralmente en el mundo de los esports, trabajando hasta 2021 como responsable de producción en la LVP (Liga de Videojuegos Profesional), pasando a formar parte del comité de dirección en 2021 como Directora de Operaciones.

En paralelo a esta entrada en una industria tecnológica, emprendió los estudios de Grado Multimedia en la UOC que finaliza con la entrega de este TFG.