

|  |  |
| --- | --- |
| CineStream  Proyecto fin de bootcamp | CineStream es una plataforma web para el alquiler de películas y series. Los usuarios pueden registrarse, acceder a un catálogo de contenido, realizar reservas temporales y gestionar sus cuentas. Los administradores tienen herramientas para gestionar contenido y usuarios. Utiliza tecnologías como Node.js, Express, MongoDB Atlas y Angular para ofrecer una experiencia segura y eficiente.  Beatriz Vázquez Amat, Edwar David Calvopiña Savedra, Iván Borra López y Cristina Sánchez-Cabezudo Goytre |

## 1.Introduccion

Se ha visto un crecimiento exponencial en la demanda de contenido audiovisual en plataformas digitales. CineStream se presenta como una solución innovadora para el alquiler de películas y series, permitiendo a los usuarios disfrutar de sus contenidos favoritos con facilidad y flexibilidad.

## 2. Objetivo

El objetivo principal del proyecto es realizar una aplicación de alquiler de contenido audiovisual, concretamente para alquilar películas y series durante el tiempo que el cliente desee.

A continuación, se exponen los objetivos secundarios de la aplicación:

* Registro y logueo de usuarios.
* Mostrar los catálogos de contenido disponible y la opción de alquilarlo, por los usuarios logueados, durante el tiempo deseado. El precio del alquiler varía en función del número de días que el usuario desee el contenido disponible.
* El usuario podrá visualizar su reserva y cancelarla, siempre que la reserva no haya comenzado aún. Por otro lado, también podrá editar su contraseña y borrar la cuenta. Todas estas funcionalidades se ponen a disposición del usuario en el apartado “Mi perfil”
* Los administradores cuentan con un espacio de gestión de recursos a través del cual pueden actualizar y eliminar el contenido de los catálogos, las reservas y los usuarios

## 3. Metodología

### 3.1. Tecnologías

Las tecnologías utilizadas para llevar a cabo este proyecto han sido:

* Mongo Atlas como base de datos en la que alojar toda la información
* Para el backend se ha usado Node + Express. Se han probado las rutas con Postman
* Para realizar el frontend se ha usado el framework Angular.
* Se ha usado Git y GitHub para facilitar el trabajo colectivo de los desarrolladores, asegurar que el código esté siempre disponible y actualizado y para dividir, organizar y asignar todas las tareas.
* El croquis del diseño de la interfaz de los usuarios y administradores se realizó con Figma

### 3.1.1. Base de datos

La base de datos en Mongo Atlas se organiza en cuatro colecciones: usuarios, películas, series y reservas. Las colecciones de usuarios, películas y series están relacionadas con las reservas

* Colección 1: Usuarios.
  + Nombre
  + Email
  + Contraseña
  + Role: administrador o usuario
* Colección 2: Películas
  + Titulo
  + Director
  + Actores
  + Año
  + Temática
  + Sinopsis
  + Valoración
  + Restricción edad
  + Precio
  + Imagen
  + Tráiler
* Colección 3: Series
  + Titulo
  + Director
  + Actores
  + Año
  + Temática
  + Sinopsis
  + Valoración
  + Restricción edad
  + Precio
  + Imagen
  + Tráiler
* Colección 4: reservas
  + Usuario que realiza la reserva (idUser)
  + Tipo del contenido reservado (type): película o serie
  + Fecha de inicio (startDate)
  + Fecha de fin (endDate)
  + Película o serie reservada (idSerie o idMovie)

### 3.1.2. Backend

El backend se ha realizado en JavaScript con Node y Express. Además, se han usado las librerías:

* Jsonwebtoken
* Bcrypt
* Cors
* Dotenv
* Mongoose

Se ha realizado un CRUD (crear, eliminar, modificar y consultar) completo para estas colecciones:

* + Películas
  + Series
  + Usuarios
  + Reservas

Se han implementado middlewares para proteger rutas y dar acceso a usuarios logueados y a usuarios administradores logueados

### 3.1.3. Frontend

El frontend se ha hecho con Typscript y con el framework de Angular. Además, se han usado las siguientes librerías:

* bootstrap: ^5.3.3
* cookie-service: ^0.0.0
* frame: ^1.0.0-alpha.2
* jquery: ^3.7.1
* ngx-cookie-service: ^17.1.0
* rxjs: ~7.8.0
* sass: ^1.77.8
* sweetalert2: ^11.12.3
* tslib: ^2.3.0
* zone.js: ~0.14.3

las páginas diseñadas han sido:

* Home
* Registro
* Login
* Catálogo de películas
* Catálogo de series
* Detalle del producto con la opción de reservarlo
* Área “Mi perfil” con dos vistas, una para consultar y eliminar las reservas y otra para consultar, editar y borrar la cuenta del usuario.
* Área de gestión de administradores, con tres vistas: una para la gestión del catálogo de productos donde pueden editar, añadir y eliminar contenidos, otra para las reservas, en donde se pueden editar y eliminar las reservas de los usuarios y por último la vista de gestión de usuarios en la cual pueden borrar a los usuarios.

Se han implementado rutas protegidas para realizar ciertas acciones a acceder a ciertas áreas de la aplicación:

* + Para reservar se requiere estar logueado
  + Para acceder al área “Mi perfil” tienes que estar logueado, y solo puedes ver tu información.
  + Para acceder al área de gestión, has de estar logueado y ser administrador.
  + Los usuarios no logueados pueden acceder a la página principal, a los catálogos, a la zona de registro y de inicio de sesión

## 4. Reparto del trabajo

Se ha repartido el trabajo de la siguiente forma

1. Backend
   1. Movies (controller, rutas y modelos): Beatriz
   2. Series (controller, rutas y modelos): Ivan
   3. Boockings (controller, rutas y modelos): Edwar
   4. Usuarios (controller, rutas y modelos): Cristina
   5. Configuración de la base de datos y el backend: Cristina
   6. Midelwares: Cristina
2. Frontend
   1. Boceto de la aplicación en figma: Edwar
   2. Catalogo de películas y series: Edwar
   3. Detalle del contenido: Edwar
   4. Realizar una reserva: Ivan
   5. Home: Cristina
   6. Área personal: Cristina
   7. Administración de usuarios: Beatriz
   8. Administración de reservas: Beatriz
   9. Administración del catálogo: Iván
   10. Registro: Iván
   11. Log in: Iván