**Proyecto de desarrollo de aplicaciones multiplataforma**

**GESTION ENTRADAS DE CINE**

CICLO FORMATIVO DE GRADO SUPERIOR

**Desarrollo de Aplicaciones Multiplataforma (IFCS02)**

**Curso 2021-22**

Autor/a/es:

**JEREMY ALEXANDER RAMOS SEGURA**

Tutor/a:

**JOSE LUIS GONZALEZ SANCHEZ**

Departamento de Informática y Comunicaciones

**I.E.S. Luis Vives**

Contenido

[2 INTRODUCCIÓN 3](#_Toc132359510)

[2.1 OBJETIVO 3](#_Toc132359511)

[2.2 ALCANCE 4](#_Toc132359512)

[2.3 Descripción del ptoyecto 4](#_Toc132359513)

[3 IMPLEMENTACIÓN 5](#_Toc132359514)

[3.1 REQUISITOS FUNCIONALES Y NO FUNCIONALES Y DE INFORMACION 5](#_Toc132359515)

[3.2 ANALISIS DEL MERCADO 7](#_Toc132359516)

[3.3 ANALISIS TECNOLÓGICO 8](#_Toc132359517)

[Ktor + Mongo 10](#_Toc132359518)

[Spring + Security 10](#_Toc132359519)

[3.4 IMPLANTACIÓN 13](#_Toc132359520)

[3.5 DOCUMENTACIÓN 13](#_Toc132359521)

[3.6 ANÁLISIS DE LA APLICACIÓN 13](#_Toc132359522)

[4 RESULTADOS Y DISCUSIÓN 14](#_Toc132359523)

[5 TRABAJO FUTURO (Opcional) 14](#_Toc132359524)

[6 CONCLUSIONES 14](#_Toc132359525)

[7 BIBLIOGRAFÍA 14](#_Toc132359526)

[ANEXOS 14](#_Toc132359527)

[I. ANEXO 1 14](#_Toc132359528)

[II. ANEXO 2… 14](#_Toc132359529)

# INTRODUCCIÓN

Como introducción al proyecto quiero recalcar que ya he tenido en cuenta la cantidad y gama de aplicaciones similares las cuales desempeñan el mismo papel. Pero como proyecto me pareció interesante ya que en estas aplicaciones hay algunas funcionalidades que para desempeñarlas como proyecto me resultan interesantes tales como la gestión de las reservas, la pasarela de pago etc. Por eso como opción al proyecto me resulto la más factible y teniendo en cuenta que el tema a tratar va de la mano con las películas será más ameno.



## OBJETIVO

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteCon este proyecto pretendo tanto aprender nuevas tecnologías orientadas al desarrollo multiplataforma. A través de encontrarme con Frameworks y otras características las cuales no he visto. También desarrollarme en el mundo multiplataforma para tener mas cerca el objetivo de un desempeño full-stack.

Reformar y adquirir nuevos conocimientos para que de forma autóctona me refuercen la base para entrar en el mundo laboral.

## ALCANCE

-Establecer una comunicación simultanea en entre clientes y un servidor de forma íntegra.

-Realizar de forma correcta la gestión de validación mediante QR´s.

-Envío en Tiempo Real de Actualizaciones de estados de las Butacas reservadas. (Mediante el mecanismo de webSockets)

-Exportar la aplicación al apartado web y al móvil.

Servidor

Se encargará de gestionar las peticiones de uno o varios clientes a la vez y almacenará dichos datos dentro de una base de datos no relacional.

Cliente

Móvil y Web: Los cuales tendrán como funcionalidad el login de si mismos, la apreciación de un catálogo activo con las películas donde puedan realizar reservas a estas.

## Descripción del ptoyecto

Como he indicado en mi anteproyecto tengo como objetivo proporcionar un sistema de gestión de entradas de cine. El programa contará con un **cliente** que se le mostrará al usuario el cual dispondrá de ciertas opciones **que puede ver estando logueado o no dentro del sistema. “*Tal vez implementar algún tipo de descuento por estar logueado*”** Tales como: Ver en cartelera, Próximamente, junto con su respectiva portada, titulo, referencia de edades y tandas. **Sera una vista en cuadricula para maximizar el atractivo de la aplicación**. Una vez seleccionada la película deseada aparecerá la opción de elegir hora de visita y a su dicha selección aparecerá el apartado de butacas donde el usuario seleccionará el numero de butacas que desea siempre y cuando no estén seleccionadas, una vez hecho aparecerá una pantalla de resumen para que el usuario vea sus opciones y se pase al método de pago. Una vez hecho el pago al usuario se le entregará las entradas con un **código QR** adjunto el cual se usará para validar la entrada. Una vez finalizada la selección de entradas, película y butacas la aplicación te indicará si quieres seguir en la compra por si le interesa comprar más entradas y en caso de estar con la sesión iniciada desloguear también nos llevaría al apartado de películas habiendo estado logueados o no dentro de la aplicación.

En el aparado del servidor todo los datos internos de la aplicación: (*Datos de películas, datos de usuarios entrantes en la app, registros de entradas y redistribución de las mismas*) se guardarán dentro del mismo en su respectiva base de datos. Por consiguiente, al cliente contar con un sistema de logueo el servidor gestionará esa entrada con un framework que expondremos más adelante. También la gestión de las entradas se tiene que gestionar en tiempo real para evitar disturbios y sobrecarga de selección de entradas. Cuyo comportamiento y/o funcionalidades serán consumidos por nuestro cliente.

# IMPLEMENTACIÓN

## REQUISITOS FUNCIONALES Y NO FUNCIONALES Y DE INFORMACION

Requisitos Funcionales

Un requisito funcional es una declaración de cómo debe comportarse un sistema. Define lo que el sistema debe hacer para satisfacer las necesidades o expectativas del usuario. Los requisitos funcionales se pueden considerar como características que el usuario detecta. Son diferentes de los requisitos no funcionales, que definen cómo debe funcionar internamente el sistema (p. ej., rendimiento, seguridad, etc.).

Los requisitos funcionales se componen de dos partes: función y comportamiento. La función es lo que hace el sistema. El comportamiento es cómo lo hace el sistema.

R-F1) Crear, modificar, eliminar y mostrar películas a elección.

R-F2) Crear, modificar, eliminar y listar salas.

R-F3) Crear, modificar, eliminar y listar usuarios.

R-F4) Gestionar el sistema de generación de reservas.

R-F5) Sistema de identificación de usuario.

R-F6) Sistema de Registro de un Usuario.

R-F7) Método de pago asociado al formato especificado: Tarjeta, Paypal etc.

R-F8) Identificación del tipo de usuario para la gestión de ofertas.

R-F9) Permitir búsquedas mediante filtros.

R-F10) Validar la información introducida en el apartado de inicio de sesión.

R-F11) El usuario tiene la capacidad de restablecer su contraseña en cualquier momento.

R-F12) Permitir al usuario ver el resumen total de su pedido y/o modificar el mismo.

R-F13) No permitir que el varios usuarios de forma simultanea tengan la misma butaca.

R-F14) Generar una entrada impresa la cual mediante QR´s se pueda validar.

Requisitos No Funcionales

Un requisito no funcional representa las restricciones o condiciones que impone el

cliente al programa que necesita, por ejemplo, el tiempo de entrega del programa, el

lenguaje o la cantidad de usuarios.

Los requisitos no funcionales de la aplicación son:

RNF1) La aplicación cliente web y móvil ha de estar expuesta en Angular.

RNF2) Cliente y servidor deben ser dos subsistemas distintos interconectados.

RNF3) La parte del servidor tiene que albergar la posibilidad de gestionar varios clientes simultaneamente.

RNF4) El lenguaje de programación en él servidor debe ser Kotlin.

RNF6) El intercambio de información entre cliente y servidor será mediante WebSockets.

RNF7) El servidor contará con una base de datos No relacional para la gestión de reservas.

RNF8) El servidor contará con una base de datos No relacional para la gestión de Usuarios.

RNF9) La aplicación contará con un inicio de sesión de usuarios.

RNF10) La aplicación contará con un sistema de roles los cuales desempeñarán una acción.

RNF11) Las entradas se validarán mediante un sistema QR.

RNF12) La aplicación contará con un inicio de sesión de usuarios Mediante tokens.

RNF13) No es necesario que el usuario esté logueado para ver las butacas.

RNF14) El usuario tiene que estar logueado para comprarlas.

RNF15) La Base de datos de la aplicación estará regulada mediante NoSQL.

Requisitos De Información

RI1) Usuario. Es aquel que ejecuta la acción de interactuar con nuestra aplicación. Cada usuario tendrá un rol en este caso admin o usuario normal. El admin realizará acciones de actualización del listado de películas y gestión de los usuarios dentro de la aplicación, así como la gestión de entadas de ofertas, menús, salas, etc. Y un usuario puede hacer reservas y consultar catálogo. Tiene como parámetros: Nombre usuario, email, contraseña, rol, estado.

RI2) Películas. Son las distintas variantes que se mostrarán en un catálogo. Tienen como parámetros: Nombre, Portada, Directores, Actores, Duración, Categorías, Descripción y estado.

RI3) Salas. Son los diferentes espacios los cuales albergarán los usuarios para el disfrute de la película. Tiene como parámetros: Nombre, Tipo, Numero de Butacas, Tipo de butacas. Fila, Butaca,

RI4) Menús. Modelo que contempla la opción de al realizar una reserva tener como complemento un menú. Tiene como parámetros: Nombre, Ingredientes, Portada, estado.

RI5) Reserva. El resumen total con los parámetros de gestión. Tiene como parámetros: Nombre, Nombre Usuario, Identificador, Fecha, estado, Película, Sala, Numero Butacas, Total.

RI6) Entradas. Modelo el cual presentará la entrada una vez realizada la reserva. Tiene como parámetros: Nombre Película, Sala, Fecha, Hora, Fila, Butaca, Tipo Sala, Precio, Código QR.

RI7) Butacas. Son los componentes de las salas que tendremos en el cine. Tienen como parámetros: Fila, Numero, Tipo.

## ANALISIS DEL MERCADO

Como ya sabemos tenemos como base que partimos de que ya existen aplicaciones similares una de estas es la aplicación del **cinesa**.



Características de la app

Dicha aplicación cuenta con un sistema similar a lo expuesto en este proyecto ya que cuenta con un apartado del login el cual se complementa de usuario y contraseña con la posibilidad de registrarse y/o cambiar contraseña. También cuentan con la implementación de la gestión de las butacas por lo visto en tiempo real lo que conlleva un punto muy fuerte de la aplicación.

También cuenta con un buscador de cines lo cual lo hace independiente ya que podremos localizar cualquier cine de la franquicia y hacer lo pertinente sobre la aplicación.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Imagen que contiene teclado

Descripción generada automáticamente

## ANALISIS TECNOLÓGICO

Aquí se expondrán los distintos puntos de vista desde el nivel tecnológico de los cuales he elegido las tecnologías pertinentes que voy a utilizar en mi aplicación.

Base de Datos

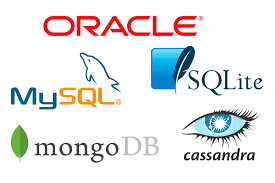
**Bases de Datos relacionales**

Las bases de datos relacionales se caracterizan por ser una colección ordenada de registros que se organizan en un conjunto de tablas. Estas tablas se relacionan entre sí.

Para acceder a estos datos, usaremos lo que se conoce como Lenguaje de Consultas Estructuradas, (SQL, Structured Query Language). Con SQL podemos obtener y alterar datos de una forma organizada siempre y cuando tengamos en cuenta cuál es la estructura de la base de datos con la que estamos trabajando.

Las bases de datos relacionales se organizan a través de identificadores. De este modo, cada tabla tiene un identificador único que es el que va a establecer su relación con el resto de tablas. A su vez, estos identificadores hacen que sea más fácil organizar cada una de las tablas por separado.

Los principales sistemas gestores de bases de datos relacionales son: MySQL, MariaDB, SQLite, PostgreSQL, SQL Server y Oracle.



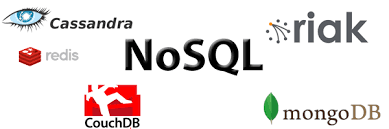
**Bases de datos no relacionales**

Las bases de datos no relacionales están diseñadas para modelos de datos específicos y que no necesitan ser relacionados con otros modelos. Cada tabla funciona de forma independiente y son mucho más sencillas que los modelos relacionales.

Las bases de datos no relacionales pueden tener identificador único, es decir, para identificar cada uno de los registros de la base de datos, pero este identificador no se usará (generalmente) para relacionar unos registros con otros. Como veremos, la información se organiza normalmente mediante documentos y es muy útil cuando no tenemos un esquema exacto de lo que se va a almacenar.

Con respecto a los formatos que se utilizan en las bases de datos no relacionales, podríamos decir que el formato más popular es el del documento. En muchos casos, lo que se utiliza es un objeto con una clave y un valor para que el acceso a la información sea pueda realizar de una forma sencilla.

Los principales sistemas gestores de bases de datos no relacionales son: MongoDB, Redis y Cassandra.



Elección

En este caso me decanté por el sistema de base de datos no relacional ya que en función de lo que tengo planteado para mi aplicación tanto gestión de reservas, como de los distintos modelos representativos de la información, será mas provechosa que una base de datos Relacional.

Back-End

En este caso tengo Dos opciones: **Ktor y SpringBoot**

Mi elección en este caso ha sido Spring + Spring Security, ya que por consiguiente a lo expuesto anteriormente en el apartado de introducción las tecnologías expuestas en este programa me gustaría sentar una base para que en el entorno profesional prospere como programador y pienso que spring será una gran oportunidad.

## Ktor + MongoLogotipo Descripción generada automáticamente con confianza media

Imagen que contiene Logotipo

Descripción generada automáticamente

Podríamos utilizar ktor gracias a que nos proporciona un sistema de programación asíncrono gracias a las corrutinas lo que nos permite una gestión eficiente de los recursos. Nos ofrece la posibilidad de una arquitectura basada en **plugins** y de módulos y/o librerías de forma eficiente y rápida.  
Como opción a la base de datos tenemos MongoDB en este apartado ya que nos ofrece ventajas frente a una base de datos Relacional que nos permitirán manejar de forma íntegra gran cantidad de datos junto con su diseño NoSQ.

## Spring + Security

Logotipo, nombre de la empresa

Descripción generada automáticamentePodríamos utilizar Spring mas especifico SpringBoot para el manejo Api´s. Lo decidimos de esta manera gracias a que podremos crear Api’s con facilidad ya que cuenta con un conjunto de características RESTful las cuales nos ayudarán con la gestión de solicitudes http como ktor y como apunte importante es su gran escalabilidad y su buena integración frente a otros Frameworks como en este caso que hemos integrado la comunicación entre Ktor y Spring. La seguridad está basada en Spring 3 ya que es una versión reciente con lo cual estará actualizada y sobre todo la seguridad con spring es muy completa contando con clases específicas para la gestión de usuarios tales como: UserDetails y UserDetailService. Gracias a esta la gestión de usuarios se ve monitorizada por spring y en todo caso podremos configurar nosotros la gestión de estos en caso beneficiario.

Front-End

Estudiando las alternativas multiplataforma, y que tenga algún interés en aprender de este apartado he dado con la conclusión que tengo dos opciones Angular + Capacitor o Flutter.

**Angular + Capacitor**

Angular es un framework Javascript potente, muy adecuado para el desarrollo de aplicaciones frontend modernas, de complejidad media o elevada. El tipo de aplicación Javascript que se desarrolla con Angular es del estilo SPA (Single Page Application) o también las denominadas PWA (Progressive Web App).

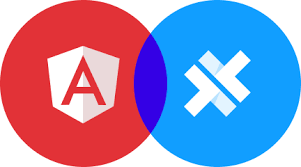
El framework Angular ofrece una base para el desarrollo de aplicaciones robustas, escalables y optimizadas, que promueve además las mejores prácticas y un estilo de codificación homogéneo y de gran modularidad.

El desarrollo en Angular se hace por medio de TypeScript (aunque también se podría desarrollar con Javascript, todas las guías y recomendaciones se basan en usar TypeScript), un superset del lenguaje Javascript que ofrece muchas herramientas adicionales al lenguaje, como el tipado estático o los decoradores.

**Capacitor**

Ionic es un framework para el desarrollo de aplicaciones híbridas. Con Ionic podemos construir aplicaciones para móviles Android, iOS, así como PWA y aplicaciones de escritorio, con una única base de código, aprovechando tus conocimientos de HTML, CSS y Javascript.

Gracias a Capacitor es muy sencillo acceder al SDK nativo de cada plataforma, con una interfaz de desarrollo unificada y optimizada para su uso con el framework Ionic. Capacitor es potente, sencillo y extensible vía plugins creados por el propio equipo de Ionic y la enorme comunidad de este framework.



**Flutter**

Flutter es el kit de herramientas de UI de Google para realizar hermosas aplicaciones, compiladas nativamente, para móvil, web y escritorio desde una única base de código.

Algunas características son:

-**Desarrollo ágil**, con features como Hot reload, que te permite hacer cambios y ver las actualizaciones de manera instantánea.

-**Apps nativas con una misma base de código.** A diferencia de otros frameworks donde la gran parte del código se comparte, en Flutter el 100% del código funciona para ambas plataformas, y no tienes que escribir código personalizado para ninguna, esto mientras compila a aplicaciones nativas.



Elección

A mi criterio esta elección es muy difícil ya que si comparamos los pros y contras de cada framework flutter cuenta con unas ventajas sobre angular. Pero me puse a pensar y viendo que en el entorno profesional el cual me desenvolveré por ahora la preferencia de las aplicaciones es en angular y simplemente por el motivo de la opción a aprender tecnologías referentes a lo esencial actualmente me decanto por angular. También la idea del capacitor me llamo mucho la atención con lo cual sí, me decanto por angular. También de forma simultanea en las prácticas estaba desempeñando la creación de interfaces mediante angular con lo cual compaginar estas tecnologías será ameno.

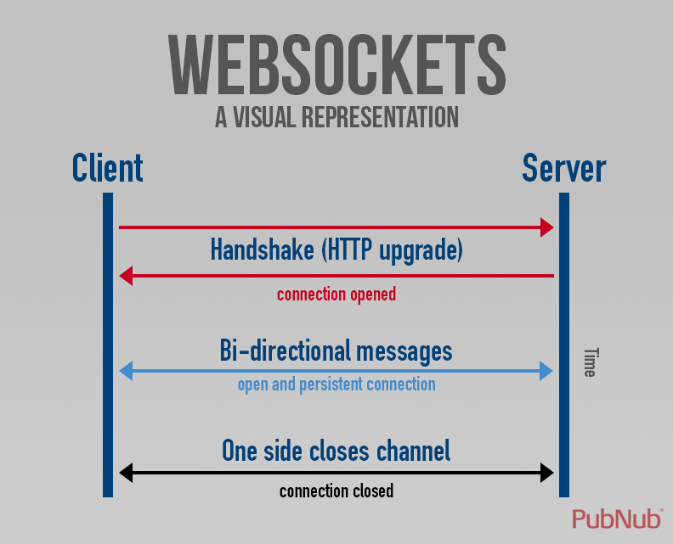
Icono

Descripción generada automáticamente

**WebSockets**

**Como funciona WebSocket:**

La comunicación a través de plataformas web está basada en HTTP, la cual permite requerir información desde un cliente, usualmente un navegador, a un servidor, por ejemplo en Node.js. El problema surge cuando queremos hacer el proceso inverso. WebSocket resuelve el problema permitiendo que la información se transmita a través de mensajes en los dos sentidos en una misma conexión.

Esto nos trae muchas ventajas, como la reducción de carga a través de la red y principalmente una arquitectura que nos permite diseñar nuestros proyectos de una manera más sencilla, minimizando los problemas que nos podemos encontrar. Ahora bien, suele ser muy conveniente utilizar abstracciones que trabajan encima de WebSocket al desarrollar nuestras aplicaciones, ya que ellas nos proveen de utilidades ya pensadas para resolver los problemas más comunes a ocurrir.

## IMPLANTACIÓN

…

## DOCUMENTACIÓN

…

## ANÁLISIS DE LA APLICACIÓN

# RESULTADOS Y DISCUSIÓN

…

# TRABAJO FUTURO (Opcional)

…

# CONCLUSIONES

…

# BIBLIOGRAFÍA

…

# ANEXOS

…

## ANEXO 1

## ANEXO 2…