# Anteproyecto de Fin de Grado

## 1. Título del proyecto

A ver que sacamos para esto.

## 2. Introducción

En la actualidad, el consumo de contenido audiovisual ha crecido exponencialmente gracias a las plataformas de streaming. Sin embargo, muchas veces los usuarios no disponen de un espacio unificado donde puedan puntuar, reseñar y descubrir películas basándose en las opiniones de otros usuarios reales.  
  
El objetivo de este proyecto es desarrollar una aplicación multiplataforma que permita a los usuarios buscar películas mediante una API externa, añadir valoraciones personales y consultar las puntuaciones de otros usuarios. La aplicación estará desarrollada con tecnologías modernas y escalables, tanto en el frontend como en el backend, y contará con integración completa en la nube.

## 3. Objetivos del proyecto

Objetivo general:  
Desarrollar una aplicación multiplataforma que permita gestionar reseñas y valoraciones de películas, con autenticación de usuarios y sincronización en la nube.  
  
Objetivos específicos:  
- Diseñar una interfaz moderna e intuitiva con Flutter.  
- Implementar un backend en Spring Boot que gestione la lógica de negocio y las peticiones del cliente.  
- Integrar una API externa de películas (por ejemplo, The Movie Database - TMDb) para obtener información actualizada.  
- Utilizar Firebase como base de datos en la nube.  
- Gestionar el flujo de trabajo con Jira.  
- Controlar las versiones del código mediante Git.  
- Desplegar la aplicación en Google Cloud.

## 4. Justificación

El desarrollo de esta aplicación permitirá poner en práctica todos los conocimientos adquiridos durante el ciclo formativo de DAM, abarcando desde la programación móvil hasta la gestión de bases de datos, APIs y despliegue en la nube.  
  
Además, la temática elegida —las reseñas de películas— es atractiva, cercana y con un claro potencial de ampliación. Este proyecto busca no solo demostrar habilidades técnicas, sino también un enfoque completo de desarrollo de software.

## 5. Metodología de trabajo

El desarrollo se llevará a cabo siguiendo una metodología ágil tipo Scrum, cada paso incluirá planificación, desarrollo, pruebas y revisión.  
  
Fases previstas:  
1. Análisis y diseño.  
2. Desarrollo del backend (Spring Boot).  
3. Desarrollo del frontend (Flutter).  
4. Integración con Firebase y API externa.  
5. Pruebas y correcciones.  
6. Despliegue en Google Cloud y documentación final.

## 6. Tecnologías utilizadas

### **Jira – Gestión del flujo de trabajo**

Jira será la herramienta principal de planificación y control del proyecto. A través de su sistema de tableros Kanban y la posibilidad de trabajar con metodologías ágiles como Scrum, se gestionarán las tareas, incidencias y fases de desarrollo.  
 Cada tarea (issue) representará una parte concreta del trabajo: diseño, desarrollo, pruebas o documentación. Además, permitirá hacer un seguimiento del progreso general y mantener una visión clara del estado del proyecto en todo momento.

**Funciones que se aprovecharán:**

* Creación y asignación de tareas con prioridades.
* Seguimiento del flujo de trabajo mediante estados (“Por hacer”, “En progreso”, “Hecho”).
* Medición de la carga de trabajo y del avance del equipo.
* Control de versiones del producto mediante integración con GitHub/GitLab.

**Ventaja principal:**  
 Permite organizar el proyecto como se haría en un entorno profesional, asegurando que cada fase esté controlada y documentada.

### **Flutter – Desarrollo del frontend**

Flutter será el framework principal para el desarrollo de la interfaz de usuario. Gracias a su capacidad multiplataforma, permitirá generar versiones de la aplicación tanto para Android como para iOS (e incluso web), sin necesidad de desarrollar código separado.  
 El lenguaje utilizado será **Dart**, diseñado por Google, que ofrece gran rendimiento, una sintaxis clara y una integración fluida con los componentes nativos del dispositivo.

La interfaz se construirá mediante **widgets personalizables**, lo que permite un diseño moderno, atractivo y coherente con las pautas de Material Design.  
 Además, Flutter incorpora la funcionalidad de **Hot Reload**, que facilita la prueba y modificación instantánea del código sin necesidad de recompilar toda la aplicación.

**Funciones clave en este proyecto:**

* Creación de pantallas interactivas para búsqueda, detalle y puntuación de películas.
* Conexión con el backend mediante peticiones HTTP (API REST).
* Integración con Firebase para la autenticación de usuarios.
* Diseño adaptable a distintos tamaños de pantalla.

**Ventaja principal:**  
 Desarrollo rápido, eficiente y multiplataforma con una experiencia de usuario nativa y fluida.

### **Spring Boot – Desarrollo del backend**

Spring Boot será la base del servidor de la aplicación, encargado de procesar las peticiones provenientes del frontend y gestionar toda la lógica de negocio.  
 Se trata de un framework basado en Java que permite crear aplicaciones backend robustas, seguras y escalables de forma sencilla.

El backend se encargará de conectar con la base de datos alojada en Firebase, así como de procesar y almacenar las reseñas, valoraciones y datos de usuarios. Además, gestionará la comunicación con la **API externa de TMDb** para obtener la información de las películas.

**Funciones clave en este proyecto:**

* Implementación de endpoints REST para comunicación con Flutter.
* Validación de usuarios y autenticación mediante tokens.
* Gestión de datos de películas, valoraciones y usuarios.
* Integración con servicios externos (Firebase, TMDb).

**Ventaja principal:**  
 Permite separar claramente la lógica de negocio de la interfaz de usuario, lo que facilita el mantenimiento, la escalabilidad y la reutilización del código en el futuro.

### **Firebase – Base de datos y servicios en la nube**

Firebase será el servicio de backend-as-a-service (BaaS) que gestionará la base de datos de la aplicación.  
 Su modelo **NoSQL (Cloud Firestore)** permite almacenar datos en tiempo real, con sincronización inmediata entre dispositivos.  
 También se utilizarán sus herramientas integradas de **autenticación de usuarios**, permitiendo registro y acceso mediante correo electrónico o proveedores externos (Google, por ejemplo).

**Funciones clave en este proyecto:**

* Almacenamiento de información sobre usuarios, películas puntuadas y valoraciones.
* Autenticación y control de acceso seguro.
* Sincronización en tiempo real de los datos.
* Posibilidad de futuras ampliaciones (notificaciones push, estadísticas, etc.).

**Ventaja principal:**  
 Reduce la complejidad del backend y facilita el trabajo con datos en tiempo real sin necesidad de montar un servidor de base de datos tradicional.

### **Git – Control de versiones**

Git será la herramienta utilizada para gestionar el código fuente del proyecto.  
 A través de un repositorio remoto (GitHub o GitLab), se mantendrá un registro completo de todos los cambios, asegurando la trazabilidad del desarrollo y evitando la pérdida de código.

**Funciones clave en este proyecto:**

* Creación de ramas (branches) para cada funcionalidad o corrección.
* Control de versiones con commits descriptivos.
* Fusión de ramas (merge) y resolución de conflictos.
* Integración con Jira para relacionar tareas con commits específicos.

**Ventaja principal:**  
 Permite un desarrollo ordenado, colaborativo y seguro, siguiendo las mismas prácticas empleadas en entornos profesionales.

### **Google Cloud – Despliegue y alojamiento**

Google Cloud Platform (GCP) será el entorno de alojamiento elegido para desplegar tanto el backend como los servicios relacionados con Firebase.  
 Al estar directamente integrado con las herramientas de Google, simplifica el proceso de publicación y mantenimiento del proyecto.

**Funciones clave en este proyecto:**

* Despliegue del servidor Spring Boot en un entorno cloud (por ejemplo, App Engine o Cloud Run).
* Integración con Firebase para almacenamiento y autenticación.
* Supervisión del rendimiento y análisis de uso.
* Escalabilidad automática según la demanda.

**Ventaja principal:**  
 Garantiza alta disponibilidad, seguridad y escalabilidad, permitiendo que la aplicación esté disponible 24/7 sin depender de servidores locales.

### **API externa (TMDb) – Datos de películas**

La **API de The Movie Database (TMDb)** será utilizada para obtener información actualizada sobre películas, series, actores y otros datos relevantes.  
 Esta API pública y gratuita proporciona un gran volumen de información, incluyendo sinopsis, géneros, carteles, puntuaciones globales y más.

**Funciones clave en este proyecto:**

* Búsqueda de películas mediante título o palabras clave.
* Obtención de detalles completos de una película (año, duración, reparto, etc.).
* Visualización de imágenes y carteles oficiales.
* Enriquecimiento de la experiencia del usuario con datos actualizados.

**Ventaja principal:**  
 Evita tener que mantener una base de datos propia de películas, reduciendo la carga de trabajo y ofreciendo siempre información actual y verificada.