

Common Room

Universidad Tecnológica Nacional
Facultad Regional Mar del Plata
Tecnicatura Universitaria en Programación

Gestión de Proyectos de Software

- Comisión 3
- *Adrian Aroca*

Programación IV

- Comisión 2
- *Lucrecia Bazan*

Integrantes

Fenoy Rivas, Delfina
Francano, Ian
Pérez, Lola

Introducción

Nuestro proyecto, de nombre **Common Room**, es una aplicación web full stack desarrollada como parte de las materias **Programación IV**, **Gestión de Desarrollo del Software** de la Tecnicatura Universitaria en Programación de la UTN – Facultad Regional Mar del Plata.

En este documento se abordan aspectos vinculados a la planificación del proyecto, el análisis funcional, la estimación del tamaño mediante el método de **Puntos de Función (PFA)**, y la estimación de esfuerzo y costo utilizando el modelo **COCOMO**.

Contexto del Proyecto

El desarrollo de **Common Room** surgió originalmente en el marco de las materias **Programación III** y **Programación IV**, donde el desafío era construir una aplicación cliente-servidor que combinara tecnologías modernas y fomentará la integración de distintas capas de desarrollo.

Durante su implementación, el equipo trabajó en el diseño del backend con **Spring Boot (Java)**, junto con un frontend basado en **HTML, CSS, JavaScript y Thymeleaf**.

A partir de esta base, en el trabajo actual se busca analizar el proyecto desde la perspectiva de **Gestión de Proyectos**, aplicando técnicas de estimación, planificación y análisis de alcance utilizadas en entornos profesionales.

Opción elegida: Full Stack

Para el desarrollo de este trabajo se optó por la **Opción A – Enfoque Full Stack**, ya que el sistema contempla tanto el desarrollo del **backend (Spring Boot – Java)** como del **frontend (Angular / HTML / CSS / JS)**.

Misión y Alcance del Proyecto

Misión

La misión de **Common Room** es ofrecer una plataforma web en la que los usuarios puedan compartir sus opiniones, reseñas y valoraciones sobre películas, fomentando el intercambio y la participación dentro de una comunidad cinéfila activa.

El sistema busca brindar una experiencia intuitiva, dinámica y confiable, que facilite la interacción entre usuarios mediante reseñas, calificaciones y perfiles públicos.

Alcance

El alcance del proyecto incluye el desarrollo de las siguientes funcionalidades principales:

- Registro y autenticación de usuarios (*con distintos roles: visitante, usuario y moderador*).
 - Creación, edición y eliminación de reseñas de películas.
 - Visualización detallada de información sobre películas obtenida desde la API externa TheMovieDB.
 - Búsqueda y filtrado de películas por título o categoría.
 - Visualización de perfiles de usuario y sus reseñas publicadas.
 - Gestión básica de usuarios y mantenimiento de datos desde la interfaz administrativa.
-

Descripción General del Sistema

Common Room es una aplicación web diseñada bajo una arquitectura **cliente-servidor**, que combina un backend robusto y un frontend dinámico.

El **backend**, desarrollado en **Spring Boot (Java 21)**, se encarga de la lógica de negocio, la autenticación mediante **tokens JWT**, la persistencia de datos en **MySQL** y la comunicación con la **API externa TheMovieDB** para obtener información actualizada de películas.

El **frontend**, implementado en **Angular**, brinda una interfaz visual moderna e intuitiva, permitiendo a los usuarios interactuar de manera sencilla con el sistema, realizar búsquedas, escribir reseñas y explorar el contenido disponible.

El sistema fue pensado para ser escalable, modular y adaptable a futuras mejoras, con el objetivo de ofrecer una experiencia fluida y segura tanto para usuarios registrados como visitantes.

Requisitos Funcionales

Listado de Requisitos

- **RF-01** → El Sistema debe permitir al Visitante registrarse.
- **RF-02** → El Sistema debe permitir al Miembro y Moderador iniciar sesión.
- **RF-03** → El Sistema debe permitir al Miembro y Moderador cerrar sesión.
- **RF-04** → El Sistema debe permitir al Miembro y Moderador realizar una alta de reseñas.
- **RF-05** → El Sistema debe permitir al Miembro y Moderador realizar una baja de reseñas propias.
- **RF-06** → El Sistema debe permitir al Miembro y Moderador modificar reseñas propias.
- **RF-07** → El Sistema debe permitir al Visitante, Miembro y Moderador visualizar reseñas de una película.
- **RF-08** → El Sistema debe permitir al Visitante, Miembro y Moderador visualizar reseñas de un usuario.
- **RF-09** → El Sistema debe permitir al Miembro y Moderador visualizar reseñas propias.
- **RF-10** → El Sistema debe permitir al Visitante, Miembro y Moderador visualizar películas.
- **RF-11** → El Sistema debe permitir al Visitante, Miembro y Moderador visualizar una película.
- **RF-12** → El Sistema debe permitir al Visitante, Miembro y Moderador filtrar películas.
- **RF-13** → El Sistema debe permitir al Moderador realizar una baja de reseñas.
- **RF-14** → El Sistema debe permitir al Visitante, Miembro y Moderador visualizar usuarios.
- **RF-15** → El Sistema debe permitir al Visitante, Miembro y Moderador visualizar un perfil de usuario.
- **RF-16** → El Sistema debe permitir al Miembro y Moderador visualizar perfil propio.
- **RF-17** → El Sistema debe permitir al Miembro y Moderador modificar perfil propio.
- **RF-18** → El Sistema debe permitir al Miembro y Moderador eliminar perfil propio.

- **RNF-01** → El Sistema debe incluir un Administrador ya creado.
- **RNF-02** → El Sistema debe tener una conexión a la API de “*TheMovieDB*”.
- **RNF-03** → El Sistema debe incluir una interfaz gráfica fácil y simple de entender

Requisitos funcionales detallados

Campo	Descripción
<i>Código</i>	RF-01
<i>Nombre</i>	Registro de Usuarios
<i>Descripción Detallada</i>	El sistema debe permitir que los usuarios se registren mediante un nombre de usuario, email, contraseña, ícono y descripción opcional.
<i>Contexto</i>	Requisito base para <u>RF-02 (Inicio de Sesión)</u> y <u>RF-03 (Gestión de Reseñas)</u> .
<i>Objetivo</i>	Permitir que nuevos usuarios creen su cuenta de forma segura para acceder a funcionalidades personalizadas como reseñas y listas.
<i>Entradas</i>	<ul style="list-style-type: none">- Username (<i>String</i>, entre 5 y 20 caracteres)- Email (<i>String</i>, formato válido, máx. 50 caracteres)- Password (<i>String</i>, mínimo 8 caracteres)- Profile Picture URL (<i>String</i>, opcional)- Description (<i>String</i>, máx. 255 caracteres, opcional)
<i>Salidas</i>	<ul style="list-style-type: none">- Token de sesión (<i>JWT</i>)- Datos del usuario autenticado- Mensaje de error si el registro falla
<i>Proceso</i>	<ol style="list-style-type: none">1. Usuario completa el formulario de registro.2. Sistema valida los datos ingresados.3. Verifica que el <i>username</i> y el email no existan.4. Si todo es válido, guarda el nuevo usuario y genera un token.5. Si ocurre un error, muestra un mensaje de validación o error general.
<i>Reglas de negocio</i>	<ul style="list-style-type: none">- Contraseñas encriptadas (<i>bcrypt</i>).- Token de sesión con vigencia de 24 h.- Username y email deben ser únicos.
<i>Prioridad</i>	Alta
<i>Notas adicionales</i>	Este requisito es esencial para el acceso de todos los perfiles.

Campo	Descripción
<i>Código</i>	RF-02
<i>Nombre</i>	Inicio de Sesión
<i>Descripción Detallada</i>	El sistema debe permitir que los usuarios ingresen con su nombre de usuario y contraseña para acceder a su cuenta.
<i>Contexto</i>	Requisito base para RF-03 (Gestión de Reseñas) y RF-05 (Visualizar Perfil) .
<i>Objetivo</i>	Autenticar usuarios registrados y otorgar acceso a las funcionalidades personalizadas del sistema.
<i>Entradas</i>	<ul style="list-style-type: none"> - Username (<i>String</i>, entre 5 y 20 caracteres) - Password (<i>String</i>, mínimo 8 caracteres)
<i>Salidas</i>	<ul style="list-style-type: none"> - Token de sesión (<i>JWT</i>) - Datos del usuario autenticado - Mensaje de error si las credenciales son inválidas
<i>Proceso</i>	<ol style="list-style-type: none"> 1. Usuario ingresa credenciales en formulario de login. 2. Sistema valida el formato de los datos. 3. Consulta la BD para verificar la existencia del usuario. 4. Si las credenciales son válidas, genera token y redirige al <i>Home</i>. 5. Si son inválidas, muestra un mensaje de error.
<i>Reglas de negocio</i>	<ul style="list-style-type: none"> - Contraseñas encriptadas (<i>bcrypt</i>). - Token con vigencia de 24 horas.
<i>Prioridad</i>	Alta
<i>Notas adicionales</i>	X

Campo	Descripción
Código	RF-03
Nombre	Gestión de Reseñas (ABM)
Descripción Detallada	El sistema debe permitir a los usuarios autenticados crear, leer, modificar y eliminar sus propias reseñas sobre películas.
Contexto	Este requisito es base para RF-04 (Visualizar ficha de una película) y RF-05 (Visualizar perfil de usuarios) .
Objetivo	Fomenta la interacción y la generación de contenido por parte de la comunidad, proveyendo valoraciones que ayuden a otros usuarios.
Entradas	<p>Alta:</p> <ul style="list-style-type: none"> • rating (doublé, entre 0.5 y 5) • description (string, máximo 700 caracteres) • movieId (long, puesto por el sistema) <p>Baja:</p> <ul style="list-style-type: none"> • reviewId (long, puesto por el sistema) <p>Modificación:</p> <ul style="list-style-type: none"> • reviewId (long, puesto por el sistema) • user (User, para saber si tiene el permiso) <p>Consulta:</p> <ul style="list-style-type: none"> • movieId (long, por la ficha de película) • userId (long, por el perfil del usuario)
Salidas	<p>Alta → Confirmación de alta.</p> <p>Baja → Confirmación de eliminación.</p> <p>Modificación → Confirmación de cambios, datos actualizados.</p> <p>Consulta → Lista de las review según el ID utilizado</p>
Proceso	<p>Alta:</p> <ol style="list-style-type: none"> 1. Usuario ingresa descripción y puntaje en un “model” de nueva reseña. 2. Sistema valida que se haya ingresado un rating y la descripción. 3. Sistema agrega a la BD la reseña y le incluye el ID de la película. <p>Baja:</p> <ol style="list-style-type: none"> 1. Usuario selecciona la review propia a eliminar 2. Sistema elimina de la base de datos la review. <p>Modificación:</p> <ol style="list-style-type: none"> 1. Usuario selecciona la review a editar. 2. Usuario ingresa descripción o puntaje a modificar. 3. Sistema valida los cambios. 4. Sistema actualiza en la BD la reseña <p>Consulta:</p> <ol style="list-style-type: none"> 1. Usuario visualiza la ficha de una película 2. Sistema muestra las reseñas relacionadas a esa película 3. Usuarios visualiza el perfil de otro usuario 4. Sistema muestra las reseñas relacionadas a ese Perfil
Reglas de negocio	<ul style="list-style-type: none"> - Un usuario solo puede publicar una reseña por película. - El rating debe estar entre 0.5 y 5, en incrementos de 0.5.
Prioridad	Alta

Campo	Descripción
<i>Código</i>	RF-04
<i>Nombre</i>	Visualizar ficha de una película
<i>Descripción Detallada</i>	Permite a usuarios y visitantes ver la ficha completa de una película, incluyendo su información general y reseñas asociadas.
<i>Contexto</i>	Base para RF-06 (Búsqueda y filtrado de películas) .
<i>Objetivo</i>	Brindar información completa sobre una película, incluyendo reseñas y valoraciones.
<i>Entradas</i>	- movied (Long, obtenido desde la URL)
<i>Salidas</i>	- Datos de la película (id, título, sinopsis, duración, fecha, promedio, poster, etc.) - Listado de reseñas asociadas
<i>Proceso</i>	1. Usuario selecciona película desde el listado o buscador. 2. Sistema consulta la BD/API para obtener datos. 3. Recupera reseñas relacionadas. 4. Muestra la información completa. 5. Si el usuario está autenticado, se habilita el botón "Review +". 6. Si hay error, se redirige a "Error404".
<i>Reglas de negocio</i>	- Solo películas activas o disponibles serán visibles. - Se mostrarán reseñas de usuarios activos. - Visitantes pueden ver fichas pero no agregar reseñas.
<i>Prioridad</i>	Alta
<i>Notas adicionales</i>	- Toda la información de la película es extraída desde la API "TMDB", únicamente las reseñas salen de la Base de Datos MySQL

Campo	Descripción
<i>Código</i>	RF-05
<i>Nombre</i>	Visualizar perfil de usuarios
<i>Descripción Detallada</i>	Permite visualizar el perfil público de un usuario, mostrando su información básica, reseñas y listas creadas.
<i>Contexto</i>	Depende de RF-02 (Inicio de Sesión) para determinar permisos y autenticación.
<i>Objetivo</i>	Permitir consultar perfiles públicos y fomentar interacción entre usuarios.
<i>Entradas</i>	<ul style="list-style-type: none"> - username (<i>String</i>, desde la URL) - token JWT (<i>String</i>, opcional)
<i>Salidas</i>	<ul style="list-style-type: none"> - username (<i>String</i>) - description (<i>String</i>) - profilePictureUrl (<i>String</i>) - createdAt (<i>LocalDateTime</i>) - Listado de reseñas del usuario - Botón “Edit Profile” visible para el propietario
<i>Proceso</i>	<ol style="list-style-type: none"> 1. Usuario accede al perfil desde menú o reseña. 2. Sistema obtiene datos públicos del usuario. 3. Carga reseñas del perfil. 4. Si el usuario autenticado es el dueño, muestra el botón “Edit Profile”. 5. Si no existe o falla, muestra modal de error.
<i>Reglas de negocio</i>	<ul style="list-style-type: none"> - Los perfiles son públicos pero limitados a datos no sensibles. - Solo el propietario ve el botón “Edit Profile”. - Las reseñas mostradas deben pertenecer al usuario visualizado. - Si no hay reseñas, se muestra mensaje informativo.
<i>Prioridad</i>	Alta
<i>Notas adicionales</i>	X

Campo	Descripción
<i>Código</i>	RF-06
<i>Nombre</i>	Búsqueda y Filtrado de películas
<i>Descripción Detallada</i>	Permite a usuarios y visitantes buscar películas por título y filtrar los resultados provenientes de la API externa.
<i>Contexto</i>	Depende de RF-04 (Visualizar Ficha de Película) .
<i>Objetivo</i>	Facilitar la exploración del catálogo mediante búsqueda y paginación de resultados.
<i>Entradas</i>	- query (<i>String</i> , desde la barra de búsqueda) - page (<i>Integer</i> , opcional para paginación)
<i>Salidas</i>	- Lista de películas (<i>id</i> , <i>título</i> , <i>sinopsis</i> , <i>fecha</i> , <i>póster</i>) - Datos de paginación (<i>totalPages</i> , <i>currentPage</i>)
<i>Proceso</i>	1. Usuario ingresa texto en la barra de búsqueda. 2. Sistema consulta la API externa. 3. Procesa resultados y los muestra en lista. 4. Si no hay resultados, muestra el mensaje "No se encontraron películas...". 5. Si hay varias páginas, genera botones de navegación. 6. Al seleccionar una película, redirige a su ficha detallada.
<i>Reglas de negocio</i>	- La búsqueda sólo se ejecuta si el campo no está vacío. - Solo se muestran películas activas o disponibles. - El sistema debe paginar resultados. - Cualquier usuario (visitante o autenticado) puede buscar.
<i>Prioridad</i>	Alta
<i>Notas adicionales</i>	X

Medición Funcional (PFA)

Tabla de conteo

<i>Tipo de Función</i>	<i>Descripción</i>	<i>Complejidad (con DETs y FTRs)</i>	<i>Puntos Asignados</i>	<i>Justificación</i>
El	Alta de Usuario (Registro) Permite crear un nuevo usuario.	Media (≈ 5 DETs, 1 FTR)	4	Alta de datos validados (<i>nombre, email, password, ícono</i>).
El	Inicio de Sesión (Login) Autenticación de usuario mediante credenciales.	Media (≈ 3 DETs, 1 FTR)	4	Valida credenciales y genera token JWT.
El	Actualizar Perfil Modificación de datos personales del usuario.	Media (≈ 4 DETs, 1 FTR)	4	Actualiza descripción, imagen, username, etc.
El	Cambiar contraseña Permite modificar la clave de acceso.	Media (≈ 3 DETs, 1 FTR)	4	Verifica la actual, válida y guarda la nueva contraseña.
El	Crear Reseña Carga de una nueva reseña por parte del usuario.	Media (≈ 4 DETs, 2 FTRs)	4	Relaciona usuario y película con texto y puntuación.
El	Modificar Reseña Edición de una reseña existente.	Media (≈ 3 DETs, 2 FTRs)	4	Permite actualizar texto o rating.
El	Eliminar Reseña Eliminación de una reseña propia.	Baja (≈ 2 DETs, 2 FTRs)	3	Quita reseña sin afectar otras entidades.
EO	Listado de Películas Populares Muestra películas desde la API externa.	Alta (≈ 7 DETs, 2 FTRs)	7	Llama a la API y muestra datos procesados.
EO	Detalle de Película Visualiza toda la información de una película.	Alta (≈ 8 DETs, 2 FTRs)	7	Incluye datos de la película y reseñas asociadas.
EO	Listado de Reseñas por Película Muestra todas las reseñas vinculadas a una película.	Media (≈ 5 DETs, 2 FTRs)	5	Consulta la base de datos y presenta resultados combinados.

EO	Mostrar Perfil de Usuario Visualiza información pública de un usuario.	Media (≈ 6 DETs, 2 FTRs)	5	Presenta datos personales y reseñas del usuario.
EO	Mostrar Reseñas del Usuario Listado de reseñas propias.	Media (≈ 5 DETs, 2 FTRs)	5	Consulta filtrada por userId.
EQ	Búsqueda de Películas Permite buscar películas por nombre o palabra clave.	Alta (≈ 6 DETs, 2 FTRs)	6	Realiza consulta a API externa con parámetros dinámicos.
EQ	Filtrado por Género Filtrar resultados según género o año.	Media (≈ 4 DETs, 2 FTRs)	4	Filtro mediante parámetros enviados a la API.
EQ	Consulta de Usuarios Obtienen información pública de usuarios.	Media (≈ 4 DETs, 2 FTRs)	4	Recupera listado de usuarios registrados visibles públicamente.
ILF	USUARIOS Archivo lógico con datos de los usuarios registrados.	Media (≈ 8 DETs, 1 FTR)	10	Contiene username, email, password, rol, fecha de alta, estado, etc.
ILF	PELÍCULAS Archivo con información de películas gestionadas en el sistema.	Alta (≈ 12 DETs, 2 FTRs)	15	Registra datos extensos: título, año, género, director, duración, rating, sinopsis, poster, etc.
ILF	RESEÑAS Archivo de reseñas de usuarios sobre películas.	Media (≈ 7 DETs, 1 FTR)	10	Registra id, user_id, movie_id, rating, comentario y fechas.
EIF	API TMDB Interfaz externa para obtener información de películas.	Alta (≈ 10 DETs, 1 FTR)	10	Interacción directa con la API de TheMovieDB (servicio externo).

Ayuda memoria

Sigla	Significa	Qué representa	Ejemplo
DET	Data Element Type	Cantidad de campos o datos visibles al usuario	<i>username, email, rating, posterUrl, etc.</i>
FTR	File Type Referenced	Archivos/tablas accedidos o consultados por una función	<i>Usuarios, Reseñas, Películas, API TMDB</i>

Factor de ajuste

N°	Característica	Valor
1	<i>Comunicación de datos</i>	4
2	<i>Procesamiento distribuido</i>	3
3	<i>Requisitos de rendimiento</i>	3
4	<i>Configuración utilizada</i>	2
5	<i>Frecuencia de transacciones</i>	3
6	<i>Entrada de datos en línea</i>	4
7	<i>Eficiencia del usuario final</i>	4
8	<i>Actualización en línea</i>	4
9	<i>Procesamiento complejo</i>	2
10	<i>Reusabilidad del código</i>	3
11	<i>Facilidad de instalación</i>	2
12	<i>Facilidad de operación</i>	3
13	<i>Múltiples localizaciones</i>	1
14	<i>Facilidad de cambios</i>	3

Datos finales:

- Total PFSA = 115
 - Total FA = 41
 - Total PFA = $115 * [0.65 + (0.01 * 41)] = 121.9 \approx \underline{\underline{122}}$
-

Modelo COCOMO

Definición de arquitectura y equipo:

Componente	Tecnología	PF	SLOC por PF	KSLOC
Backend	Java (SpringBoot)	80	50	4,00
Frontend	Angular (TypeScript/HTML/CSS)	35	40	1,40
<u>Total</u>		115		5,40 KSLOC

Cálculos

Esfuerzo (Persona-Mes)

$$PM = 2,4 \times (5,40)^{1,05} = 14,10 \approx 15 \text{ meses-persona}$$

Tiempo de desarrollo

$$TDEV = 2,5 \times (15)^{0,35} = 6,45 \approx 7 \text{ meses}$$

Personas promedio necesarias

$$N = 15/7 = 2,14 \approx 3 \text{ personas}$$

Técnica Tres Puntos

Escenarios

RF-01 (Registro de Usuarios)

Escenario	Duración	Justificación
Optimista (O)	<u>2 días</u>	<ul style="list-style-type: none">- El equipo ya tiene experiencia con Spring Security.- JWT es un proceso conocido.- No surgen problemas con la validación de datos.
Más Probable (M)	<u>3 días</u>	<ul style="list-style-type: none">- Experiencia promedio del equipo.- Tiempo estándar para configurar el endpoint, el - servicio de usuario y las validaciones.- Pequeños ajustes en la configuración de seguridad.
Pesimista (P)	<u>5 días</u>	<ul style="list-style-type: none">- Problemas con la configuración del filtro de JWT.- Dificultades con las dependencias de seguridad.- Errores inesperados en la persistencia de datos con la contraseña encriptada.

RF-02 (Inicio de Sesión)

Escenario	Duración	Justificación
Optimista (O)	<u>1 días</u>	<ul style="list-style-type: none">- Se reutiliza el 90% de la configuración de seguridad de RF-01.- La lógica de autenticación es directa.- No hay imprevistos.
Más Probable (M)	<u>2 días</u>	<ul style="list-style-type: none">- Implementación estándar del endpoint de login.- Se realizan pruebas de autenticación para casos de éxito y error.
Pesimista (P)	<u>4 días</u>	<ul style="list-style-type: none">- Problemas al verificar las contraseñas.- Conflictos en la cadena de filtros de Spring Security.- Dificultad para gestionar correctamente las excepciones de autenticación.

RF-03 (Gestión de Reseñas)

Escenario	Duración	Justificación
Optimista (O)	<u>3 días</u>	<ul style="list-style-type: none">- El desarrollador tiene claro el manejo de relaciones en JPA.- Los endpoints CRUD se implementan rápidamente.- La lógica de negocio (<i>una reseña por usuario/película</i>) es simple de implementar.
Más Probable (M)	<u>5 días</u>	<ul style="list-style-type: none">- Tiempo estándar para desarrollar los 4 endpoints (<i>Crear, Leer, Actualizar, Borrar</i>).- Implementación de la capa de servicio con validaciones y lógica de autorización para que un usuario solo modifique lo suyo.
Pesimista (P)	<u>8 días</u>	<ul style="list-style-type: none">- Complejidad en el manejo de las relaciones entre Usuario, Película y Reseña.- Problemas con las transacciones de la base de datos.- La lógica de autorización para las operaciones de modificar y eliminar resulta más compleja de lo esperado.

RF-04 (Visualizar ficha de una película)

Escenario	Duración	Justificación
Optimista (O)	<u>2 días</u>	<ul style="list-style-type: none">- La consulta para obtener la película y sus reseñas asociadas es sencilla.- No hay problemas de rendimiento.
Más Probable (M)	<u>3 días</u>	<ul style="list-style-type: none">- Se requiere optimizar la consulta para evitar problemas (cargar reseñas de forma eficiente).- Desarrollo estándar del endpoint y los DTOs (Data Transfer Objects) para la respuesta.
Pesimista (P)	<u>6 días</u>	<ul style="list-style-type: none">- La consulta a la base de datos es lenta y requiere una optimización avanzada.- Problemas con la serialización de datos.- Se necesita refactorizar la estructura de los DTOs.

RF-05 (Visualizar perfil de un usuario)

Escenario	Duración	Justificación
Optimista (O)	<u>2 días</u>	<ul style="list-style-type: none">- La lógica es muy similar a RF-04 y se reutiliza parte del código.- La consulta para obtener el usuario y sus reseñas es directa.
Más Probable (M)	<u>3 días</u>	<ul style="list-style-type: none">- Implementación de la lógica para obtener datos públicos y la lista de reseñas.- Se añade la verificación para mostrar el botón "Edit Profile" si el usuario autenticado es el dueño.
Pesimista (P)	<u>5 días</u>	<ul style="list-style-type: none">- Surgen problemas de rendimiento al cargar perfiles con muchas reseñas.- La lógica para diferenciar entre el dueño del perfil y un visitante genera bugs.

RF-06 (Búsqueda y filtrado de Películas)

Escenario	Duración	Justificación
Optimista (O)	<u>2 días</u>	<ul style="list-style-type: none">- La API externa está bien documentada y es estable.- El mapeo de la respuesta de la API a los objetos Java es directo.
Más Probable (M)	<u>4 días</u>	<ul style="list-style-type: none">- Tiempo estándar para configurar el cliente HTTP.- Implementación del manejo de la paginación y la lógica de búsqueda.- Gestión de errores básicos (ej. 404 si no hay resultados).
Pesimista (P)	<u>7 días</u>	<ul style="list-style-type: none">- La API externa es inconsistente o tiene una documentación pobre.- Dificultades para mapear una respuesta JSON compleja.- Problemas con el manejo de errores de red, timeouts o límites de peticiones.

Cálculo Final

Duración esperada = $(O + 4M + P) \div 6$

- ***RF-01: Registró*** = $(2 + 4 \times 3 + 5) \div 6 = 19 \div 6 \approx \underline{\underline{3.2 \text{ días}}}$
- ***RF-02: Login*** = $(1 + 4 \times 2 + 4) \div 6 = 13 \div 6 \approx \underline{\underline{2.2 \text{ días}}}$
- ***RF-03: Gestión de Reseñas*** = $(3 + 4 \times 5 + 8) \div 6 = 31 \div 6 \approx \underline{\underline{5.2 \text{ días}}}$
- ***RF-04: Ficha de Pelicula*** = $(2 + 4 \times 3 + 6) \div 6 = 20 \div 6 \approx \underline{\underline{3.3 \text{ días}}}$
- ***RF-05: Perfil de Usuario*** = $(2 + 4 \times 3 + 5) \div 6 = 19 \div 6 \approx \underline{\underline{3.2 \text{ días}}}$
- ***RF-06: Búsqueda*** = $(2 + 4 \times 4 + 7) \div 6 = 25 \div 6 \approx \underline{\underline{4.2 \text{ días}}}$

Total estimado de días = 21.3 días

Arquitectura del Sistema

Diagrama de contexto

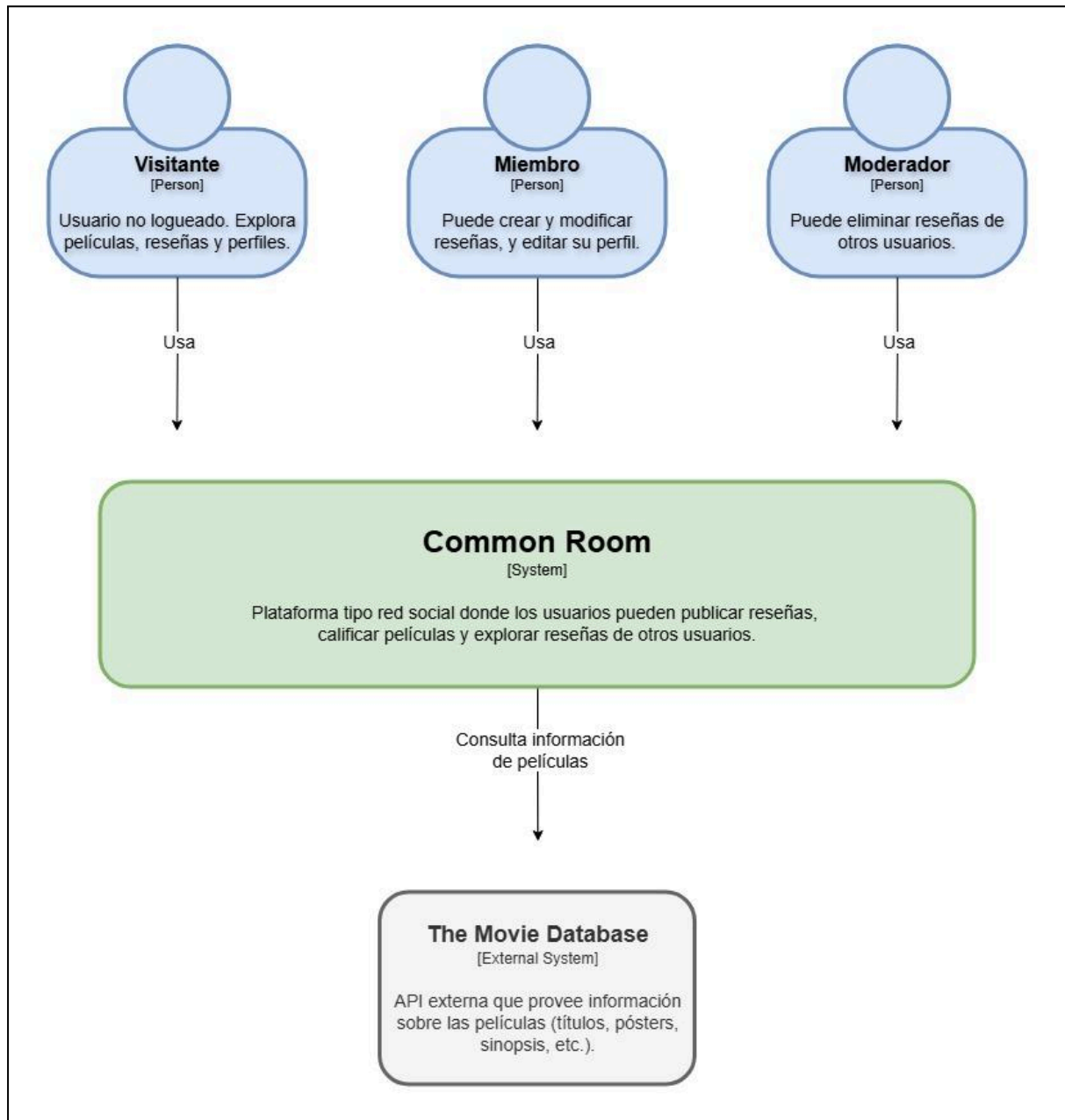
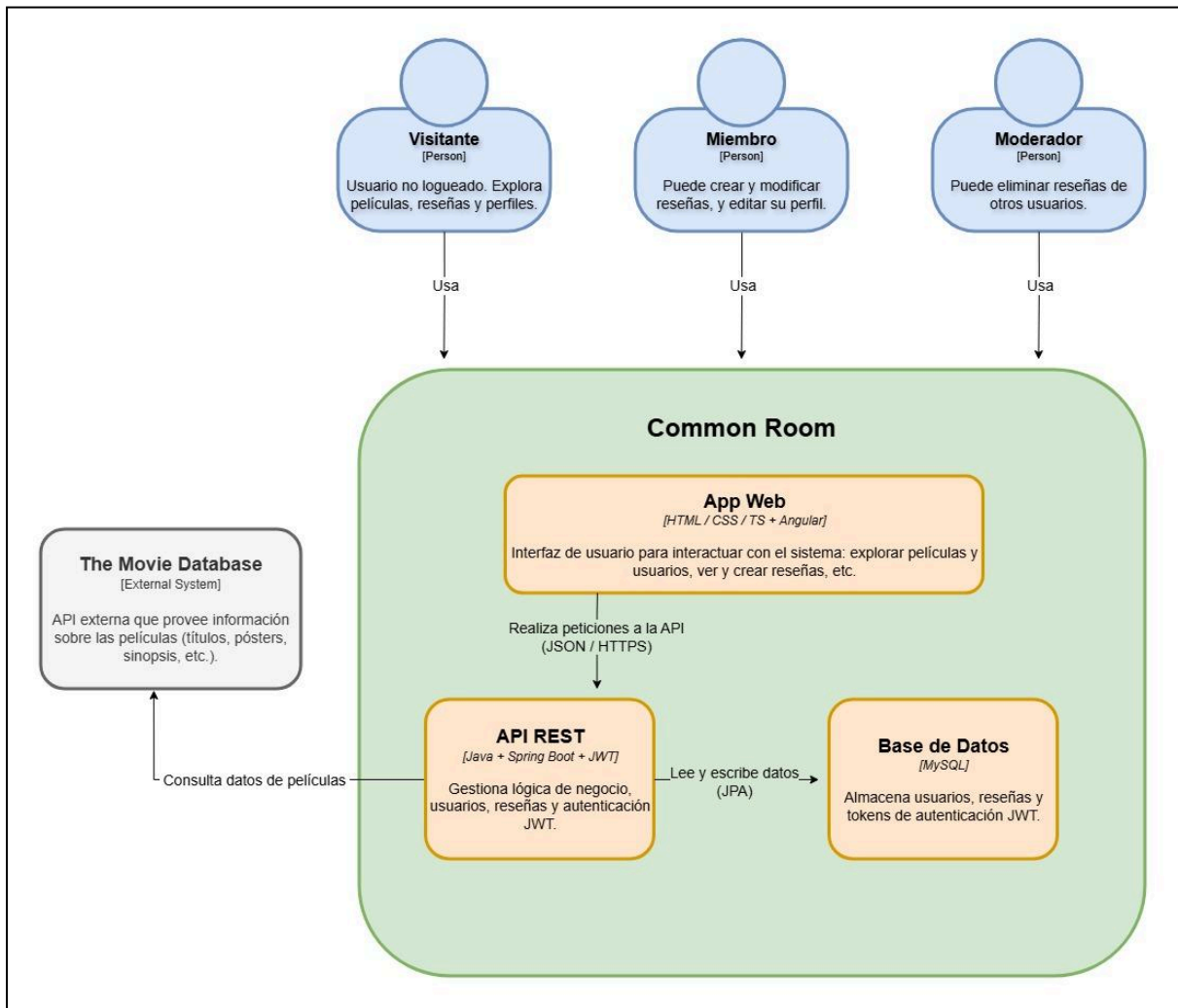


Diagrama de contenedores



Gestion de Riesgos

ID	Riesgo	Categoría	Probabilidad	Impacto	Exposición	Estrategia	Responsable
R - 01	Problemas de integración entre el frontend y backend (CORS)	Técnico	<u>Media</u>	<u>Alto</u>	Alta	Mitigar: Realizar pruebas de integración tempranas y continuas.	Ian Francano
R - 02	Fallas o cambios inesperados en la API externa para la búsqueda de películas (RF - 06)	Técnico	<u>Media</u>	<u>Alto</u>	Alta	Mitigar: Desarrollar un manejo de errores robusto para informar al usuario de manera amigable si la API no responde.	Delfina Fenoy
R - 03	Bajo rendimiento en consultas a la base de datos al cargar perfiles y fichas de películas con sus reseñas	Técnico	<u>Alta</u>	<u>Medio</u>	Alta	Mitigar: Utilizar DTOs para devolver solo la información necesaria.	Lola Pérez
R - 04	Vulnerabilidades de seguridad en la implementación de la autenticación con JWT y la gestión de contraseñas	Técnico	<u>Media</u>	<u>Alto</u>	Alta	Mitigar: Almacenar el "secret key" de JWT de forma segura (variables de entorno)	Delfina Fenoy
R - 05	La lógica de negocio para la gestión de reseñas (RF - 03) resulta ser más compleja de lo previsto	Alcance	<u>Media</u>	<u>Alto</u>	Alta	Mitigar: Priorizar funcionalidades	Delfina Fenoy
R - 06	Las estimaciones de tiempo resultan ser demasiado optimistas, generando retrasos en la entrega	Tiempo	<u>Media</u>	<u>Alto</u>	Alta	Mitigar: Realizar seguimiento diario del progreso contra las estimaciones.	Lola Pérez
R - 07	Disponibilidad limitada de un desarrollador clave con conocimiento crítico en Spring Security	Equipo	<u>Baja</u>	<u>Alto</u>	Medio	Mitigar: Fomentar la programación en pareja en las tareas de seguridad.	Ian Francano