

# GrooveLogs



Proyecto PWA  
Gestión y Valoración Musical

DAW3-PWA-DISCOGS-2025

Sara Monzón Quesada

3DAW Noche

Curso 2025-2026

## Índice

1. Resumen ejecutivo .....	2
Alcance del proyecto.....	2
2. Requisitos del proyecto.....	3
Requisitos funcionales.....	3
Requisitos no funcionales .....	3
Requisitos técnicos.....	4
3. Estructura de trabajo y cronograma .....	4
División en fases o tareas principales. ....	4
Fechas estimadas y responsables .....	6
Formato de planificación.....	6
Recursos y roles del equipo.....	6
Recursos técnicos y materiales necesarios .....	7
4. Gestión de riesgos .....	7
Posibles problemas o retrasos. ....	7
Plan de contingencia o prevención .....	7
5. Conclusión .....	7

<b>Nombre del fichero:</b>	ETS_XXX_BBB_1. Documento de alcance.odt
<b>Fecha de esta versión:</b>	14/12/2025 20:03:00
<b>Fecha de esta versión:</b>	19/01/2026 19:51

## 1. Resumen ejecutivo

En este proyecto se desarrollará una Aplicación Web Progresiva (PWA) cuyo objetivo es permitir a los usuarios buscar y explorar canciones, álbumes y artistas, así como gestionar su colección musical personal.

Una Aplicación Web Progresiva (PWA) es una aplicación web que combina características propias de las aplicaciones web tradicionales y de las aplicaciones nativas. Se ejecuta en el navegador, pero puede instalarse en el dispositivo del usuario, funcionar parcialmente sin conexión a Internet y ofrecer una experiencia similar a la de una aplicación móvil.

Las PWA hacen uso de tecnologías como Service Workers, Web App Manifest y almacenamiento en caché, lo que permite mejorar el rendimiento, la disponibilidad y la experiencia de usuario en distintos dispositivos y plataformas.

La aplicación permitirá marcar contenidos musicales como favoritos y asignarles una puntuación del 0 al 5, facilitando la organización y valoración de la experiencia musical del usuario.

Para garantizar una gestión personalizada y segura de los datos, las funcionalidades de añadir a favoritos y puntuar contenidos estarán disponibles únicamente para aquellos usuarios que se encuentren registrados y autenticados en la base de datos del sistema.

### Alcance del proyecto

Los objetivos generales del proyecto son desarrollar una Aplicación Web Progresiva que permita a los usuarios buscar, valorar y gestionar su colección musical personal, incorporando funcionalidades de autenticación y almacenamiento de datos de forma segura.

Se incluye la búsqueda y visualización de información musical, la gestión de usuarios registrados además de un sistema de favoritos y puntuaciones. Para ello haremos uso de una API externa para obtener datos musicales y contará con las funcionalidades básicas propias de una PWA.

No incluirá sin embargo la reproducción de audio y otras funcionalidades propias de redes sociales como comentarios, seguidores o un chat. Tampoco habrá pasarela de pagos o

servicios premium, con el registro ya los usuarios tendrán acceso a todas las funcionalidades de la aplicación.

## 2. Requisitos del proyecto

### Requisitos funcionales

- RF-01: El sistema deberá permitir a cualquier usuario buscar canciones, álbumes y artistas mediante una API externa de información musical.
- RF-02: El sistema deberá mostrar la información de las canciones, álbumes y artistas obtenida de la API.
- RF-03: El sistema deberá permitir a los usuarios registrarse en la aplicación.
- RF-04: El sistema deberá permitir a los usuarios iniciar y cerrar sesión.
- RF-05: El sistema deberá permitir a los usuarios registrados añadir canciones, álbumes y artistas a su lista de favoritos.
- RF-06: El sistema deberá permitir a los usuarios registrados eliminar contenidos de su lista de favoritos.
- RF-07: El sistema deberá permitir a los usuarios registrados puntuar contenidos musicales con un valor del 0 al 5.
- RF-08: El sistema deberá almacenar de forma persistente los favoritos y puntuaciones asociados a cada usuario.
- RF-09: El sistema deberá restringir las funcionalidades de favoritos y puntuaciones únicamente a usuarios autenticados.
- RF-10: El sistema deberá permitir al usuario consultar su perfil, visualizando sus favoritos y puntuaciones.

### Requisitos no funcionales

- RNF-01: La aplicación deberá contar con una interfaz responsive, adaptándose a distintos tamaños de pantalla.
- RNF-02: La aplicación deberá ofrecer una experiencia de usuario intuitiva y accesible.
- RNF-03: El sistema deberá garantizar la seguridad de los datos de los usuarios, evitando accesos no autorizados.

- RNF-04: La aplicación deberá ofrecer tiempos de respuesta adecuados en las búsquedas y navegación.
- RNF-05: La PWA deberá ser instalable en dispositivos compatibles.
- RNF-06: El sistema deberá permitir un uso parcial en modo offline (visualización de datos previamente cargados).
- RNF-07: El código fuente deberá ser modular, mantenible y documentado.
- RNF-08: La aplicación deberá ser compatible con los principales navegadores web actuales.

#### Requisitos técnicos.

- RT-01: El frontend de la aplicación se desarrollará utilizando React, haciendo uso de componentes reutilizables para la construcción de la interfaz de usuario.
- RT-02: El backend se implementará mediante Spring Boot.
- RT-03: La comunicación entre frontend y backend se realizará a través de una API REST, utilizando el protocolo HTTPS.
- RT-04: La aplicación se implementará como una Aplicación Web Progresiva (PWA), incorporando Service Workers y Web App Manifest.
- RT-05: Se utilizará una API externa de información musical (Discogs<sup>1</sup>) para la obtención de datos relacionados con canciones, álbumes y artistas.
- RT-06: El sistema contará con una base de datos relacional para el almacenamiento de usuarios, favoritos y puntuaciones.
- RT-07: Se implementará un sistema de autenticación y autorización en el backend para la gestión de usuarios.
- RT-08: El proyecto utilizará Git como sistema de control de versiones.
- RT-09: La aplicación seguirá una arquitectura cliente-servidor, separando frontend y backend.
- RT-10: El despliegue de la aplicación se realizará en un entorno web accesible desde navegadores modernos (aquellos con soporte para Service Workers y Web App Manifest).

### 3. Estructura de trabajo y cronograma

División en fases o tareas principales.

---

<sup>1</sup> Acceso a la [página](#) web de la API

## Fase 1: Análisis y planificación

- Definición de los requisitos funcionales, no funcionales y técnicos.
- Análisis del alcance y limitaciones del proyecto.
- Estudio de la API externa de información musical (Discogs).
- Elaboración de la planificación y cronograma del proyecto.

## Fase 2: Diseño del sistema

- Bocetos y diseño inicial de la interfaz de usuario.
- Diseño de la arquitectura general (frontend y backend).
- Diseño de la base de datos.
- Definición de la estructura de la API REST.

## Fase 3: Desarrollo del backend

- Creación del proyecto Spring Boot.
- Implementación del modelo de datos.
- Desarrollo del sistema de autenticación y gestión de usuarios.
- Implementación de los endpoints REST para favoritos y puntuaciones.
- Pruebas básicas de la API.

## Fase 4: Desarrollo del frontend

- Creación del proyecto React.
- Implementación de la interfaz de usuario.
- Integración con la API REST del backend.
- Implementación de búsqueda, favoritos y puntuaciones.
- Gestión del estado de la aplicación.

## Fase 5: Implementación PWA

- Configuración del Web App Manifest.
- Implementación del Service Worker.
- Gestión básica del modo offline.
- Pruebas de instalación como PWA.

## Fase 6: Pruebas y validación

- Pruebas funcionales del sistema.
- Verificación del cumplimiento de los requisitos.
- Corrección de errores detectados.
- Pruebas de usabilidad.

## Fase 7: Documentación y entrega

- Redacción de la memoria final del proyecto.
- Preparación de la presentación.
- Revisión final del proyecto.

## Fechas estimadas y responsables

Fase	Período Estimado	Responsable
<b>Análisis y planificación</b>	Semana 1	Alumno
<b>Diseño del sistema</b>	Semana 2	Alumno
<b>Desarrollo del backend</b>	Semana 3-4	Alumno
<b>Desarrollo del frontend</b>	Semana 5-6	Alumno
<b>Implementación PWA</b>	Semana 7	Alumno
<b>Pruebas y validación</b>	Semana 8	Alumno
<b>Documentación y entrega</b>	Semana 9	Alumno

## Formato de planificación

Para la planificación y seguimiento del proyecto, se utilizará GitHub Projects, donde el control de las tareas estará asociado a tableros Kanban. Habrá distintas columnas para ir controlando el estado en el que se encuentran las tareas (Backlog, Pendientes, En progreso y Realizadas), además de diferenciar entre tareas de frontend, backend y documentación.

## Recursos y roles del equipo

El presente proyecto será desarrollado íntegramente de forma individual, asumiendo la autora la responsabilidad del diseño e implementación del frontend y backend de la aplicación,

así como del diseño UX/UI y la elaboración de la documentación técnica y funcional necesaria para el correcto desarrollo y presentación del proyecto.

### Recursos técnicos y materiales necesarios

Necesitamos un ordenador con conexión a Internet, un editor de código (en este caso, usaremos Visual Studio Code y Eclipse), una base de datos, que en este caso una base de datos relacional, implementada mediante un gestor de base de datos (SQL), para almacenar la información. Además, necesitamos acceso y conexión a la API de donde vamos a obtener los datos y tener nuestro repositorio subido a GitHub para realizar el control de versiones en la nube, no solo localmente con Git.

## 4. Gestión de riesgos

### Posibles problemas o retrasos.

Uno de los problemas principales que encuentro, es la posible dificultad a la hora de integrar la API externa en la aplicación (límite de uso, tiempo de respuesta elevado). También puede dar complicaciones la sincronización entre el frontend y el backend de la aplicación, además de las tareas un poco más complicadas que hemos presentado, como el agregar favoritos y la gestión del modo offline.

### Plan de contingencia o prevención

Priorizar el desarrollo de las funcionalidades principales, dejando las características secundarias para fases posteriores. Hay que ir revisando periódicamente el estado de las tareas de GitHub Projects e ir haciendo pruebas continuas sobre la aplicación para detectar posibles fallos de funcionalidad.

## 5. Conclusión

El proyecto GrooveLog presenta un proyecto que se apoya en tecnologías ampliamente utilizadas en el desarrollo web actual, como React para el frontend y Spring Boot para el backend, así como en el uso de una API externa de información musical.

La implementación de una PWA está pensada para mejorar la accesibilidad y la experiencia de usuario en distintos dispositivos, abarcando un público más amplio porque actualmente hay un gran auge de usuarios de aplicaciones móviles.



A nivel académico, este proyecto intenta reflejar lo aprendido durante el ciclo, abarcando diversas áreas de desarrollo como pueden son el desarrollo frontend, backend, gestión de base de datos, autenticación y planificación de proyectos.