PROYECTO – APLICACIÓN WEB “MusicConnect”

RA1 – DEFINICIÓN DEL PROYECTO

## RA1.a – Clasificación de empresa

Introducción

El proyecto **MusicConnect** consiste en el desarrollo de una aplicación web de streaming musical con un fuerte componente social. Inspirada en plataformas como Spotify, busca fomentar la interacción entre los usuarios mediante comentarios, valoraciones y la posibilidad de compartir listas de reproducción.

Su objetivo es **transformar la experiencia musical en línea** en algo más colaborativo y participativo, creando una comunidad donde los usuarios puedan descubrir nueva música a través de otros miembros, más allá de los algoritmos automáticos.

Tipo de organización

El proyecto se plantea en el contexto de una **empresa privada tecnológica** dedicada al desarrollo de soluciones web y plataformas multimedia. La empresa busca posicionarse dentro del sector de las **aplicaciones web de entretenimiento y redes sociales musicales**.

Servicio ofrecido

MusicConnect ofrecerá servicios de:

* Reproducción de música en línea.
* Creación, gestión y compartición de listas de reproducción.
* Interacción social mediante comentarios, valoraciones y reportes.
* Distinción entre usuarios **normales**, **premium** y **administradores** con diferentes permisos.

Justificación de la necesidad

Actualmente, la mayoría de plataformas de streaming se centran en el consumo individual. MusicConnect cubrirá la necesidad de una **plataforma musical social**, donde la música sirva como punto de encuentro entre personas con gustos similares.  
Esta propuesta combina funcionalidades de **reproducción musical** y **red social**, generando así una experiencia más cercana, colaborativa y moderna.

## RA1.b – Caracterización de usuarios

Actores principales

1. **Administrador**
2. **Usuario normal**
3. **Usuario premium**

Roles y funciones

| **Rol** | **Descripción** | **Permisos principales** |
| --- | --- | --- |
| **Administrador** | Supervisa toda la plataforma. Gestiona usuarios, canciones, reportes y estadísticas. | CRUD completo de canciones y usuarios, generación de informes en PDF, gestión de reportes y bloqueos. |
| **Usuario normal** | Interactúa con la plataforma, escucha música, crea y comparte listas. | CRUD de listas personales, valorar canciones, comentar, reportar comentarios inapropiados. |
| **Usuario premium** | Disfruta de la experiencia completa sin límites. | Todas las funciones del usuario normal + acceso a catálogo ampliado y sin anuncios. |

## RA1.c – Catálogo de requisitos funcionales y de usuario

Requisitos funcionales (RF)

| **ID** | **Requisito funcional** | **Necesidad asociada** |
| --- | --- | --- |
| RF1 | El sistema permitirá el registro y autenticación de usuarios. | Control de acceso y seguridad. |
| RF2 | El usuario podrá escuchar canciones disponibles. | Acceso básico al contenido. |
| RF3 | Los usuarios podrán crear, editar y eliminar sus listas de reproducción. | Organización de contenido personalizado. |
| RF4 | El usuario podrá hacer públicas o privadas sus listas. | Privacidad y compartición. |
| RF5 | Los usuarios podrán valorar canciones con estrellas o puntuaciones. | Interacción social y retroalimentación. |
| RF6 | Los usuarios podrán dejar comentarios en canciones o listas. | Participación activa y comunidad. |
| RF7 | El usuario podrá reportar comentarios inapropiados. | Moderación de la comunidad. |
| RF8 | El administrador podrá eliminar usuarios o canciones. | Mantenimiento y control del sistema. |
| RF9 | El sistema generará informes en PDF sobre actividad o recaudación. | Análisis y gestión administrativa. |
| RF10 | Se enviarán emails automáticos ante registro, sanción o resolución de reporte. | Comunicación automatizada y trazabilidad. |

Requisitos de usuario (RU)

| **ID** | **Requisito de usuario** | **Necesidad asociada** |
| --- | --- | --- |
| RU1 | El usuario quiere una interfaz clara y atractiva. | Usabilidad y diseño responsive. |
| RU2 | El usuario desea escuchar música sin interrupciones. | Rendimiento y experiencia fluida. |
| RU3 | El usuario quiere compartir sus listas con amigos. | Conectividad social. |

## RA1.d – Estudio de viabilidad técnica Tecnologías y tendencias

Tecnologías seleccionadas

* **Lenguaje backend:** PHP 8
* **Base de datos:** MySQL
* **Frontend:** HTML5, CSS3, JavaScript, Bootstrap 5
* **Framework CSS:** Bootstrap para diseño responsive
* **AJAX:** Para actualizaciones en tiempo real sin recargar la página (por ejemplo, al valorar canciones o enviar comentarios).
* **Librería adicional:** SweetAlert 2 para notificaciones visuales.

Justificación tecnológica

Estas tecnologías son ampliamente usadas en el desarrollo web moderno, compatibles con la mayoría de servidores de hosting y alineadas con la formación del ciclo DAW.  
Además, se integran perfectamente con arquitecturas **MVC**, permitiendo escalabilidad y mantenimiento.

Tendencias relacionadas

* **Aplicaciones web interactivas y sociales.**
* **Integración multimedia en la nube.**
* **Smart Music Apps** con interacción comunitaria.
* **Responsive Design** para garantizar acceso desde cualquier dispositivo.

## RA1.e – Tipo de proyecto

Entidades (mínimo 4 CRUDs completos)

1. **Usuario** – Datos de autenticación, tipo de cuenta, estado.
2. **Canción** – Información musical (título, artista, duración, género).
3. **Lista de Reproducción** – Creada por un usuario, con canciones asociadas.
4. **Comentario** – Asociado a canciones o listas.
5. **Reporte** – Asociado a comentarios, con gestión de estado por parte del administrador.

Roles diferenciados

* **Administrador:** Acceso total y gestión global.
* **Usuario Normal:** Funcionalidades básicas.
* **Usuario Premium:** Funcionalidades extendidas (sin anuncios, catálogo completo).

## RA1.f – Características técnicas específicas ()

| **Característica** | **Descripción** |
| --- | --- |
| CRUDs completos | Al menos 4 entidades con creación, lectura, actualización y eliminación. |
| Roles diferenciados | 3 roles (admin, usuario normal, usuario premium). |
| Emails automáticos | En registro, sanción y resolución de reportes. |
| AJAX | En comentarios y valoraciones. |
| Listados con filtros + paginación | Para canciones, usuarios y comentarios. |
| Validaciones FE/BE | Validaciones de formularios tanto en frontend (JS) como backend (PHP). |
| Diseño responsive | Uso de Bootstrap para adaptarse a móvil, tablet y PC. |

## RA1.g – Obligaciones legales

### Ley de Protección de Datos (LOPD GDD / RGPD)

**Descripción:** MusicConnect maneja datos personales de usuarios (nombre, correo, tipo de cuenta, historial de reproducción, comentarios). La Ley de Protección de Datos exige garantizar la privacidad y seguridad de estos datos, permitir el consentimiento explícito del usuario y ofrecer opciones de acceso, rectificación y eliminación.

**Implementación práctica:**

* **Consentimiento informado:** Al registrarse, el usuario debe aceptar la política de privacidad mediante un checkbox obligatorio.
* **Política de privacidad:** Página web con explicación clara de cómo se recopilan, usan y protegen los datos.
* **Gestión de usuarios:** Posibilidad para que los usuarios eliminen su cuenta y sus datos.
* **Seguridad:** Encriptación de contraseñas (bcrypt o similar), uso de HTTPS y restricciones de acceso según rol.
* **Registro de actividad:** Guardar logs de acciones importantes (reportes, cambios de contraseña, comentarios eliminados) para trazabilidad.

### Accesibilidad web (Ley de Accesibilidad y WCAG 2.1)

**Descripción:** Todas las páginas deben ser accesibles a personas con discapacidades visuales, auditivas o motoras. Se recomienda seguir las pautas WCAG 2.1.

**Implementación práctica:**

* **Etiquetas semánticas:** Uso correcto de <header>, <nav>, <main>, <footer>.
* **Textos alternativos:** Todas las imágenes y botones con alt descriptivo.
* **Contraste de colores:** Asegurar contraste adecuado entre texto y fondo (mínimo 4.5:1).
* **Navegación con teclado:** Todos los elementos interactivos deben ser accesibles mediante teclado (tabindex).
* **Lectores de pantalla:** Etiquetas ARIA para elementos dinámicos (comentarios, valoraciones

### Prevención de riesgos laborales (aplicable en el desarrollo)

**Descripción:** Aunque el proyecto es digital, el desarrollo y mantenimiento implica riesgos ergonómicos y psicológicos. La normativa obliga a prevenir daños derivados del trabajo en pantallas.

**Implementación práctica:**

* Pausas recomendadas y ergonomía en el puesto de trabajo.
* Formación en buenas prácticas de seguridad digital.
* Copias de seguridad automáticas para proteger información y evitar sobrecarga de trabajo por pérdida de datos.

## RA1.h – Ayudas y subvenciones

**Descripción:** Existen distintas ayudas y subvenciones disponibles para proyectos tecnológicos y startups en España, especialmente para el desarrollo de aplicaciones web y plataformas innovadoras.

**Posibles fuentes de financiación:**

1. **ICEX España Exportación e Inversiones:** Apoyo a startups tecnológicas con potencial internacional.
2. **ENISA (Empresa Nacional de Innovación):** Préstamos participativos para proyectos innovadores.
3. **Red.es:** Programas de subvenciones para digitalización, desarrollo web y apps interactivas.
4. **Ayudas autonómicas (Junta de Andalucía):** Convocatorias para empresas TIC, incluyendo subvenciones a la innovación digital.
5. **Fondos Europeos (FEDER, Horizonte Europa):** Financiación de proyectos tecnológicos con impacto social y cultural.

**Implementación práctica en MusicConnect:**

* Justificación de uso de fondos: desarrollo de plataforma, integración multimedia, implementación de seguridad y accesibilidad.
* Inclusión de informes y métricas: actividad de usuarios, ingresos por suscripciones, impacto social de la interacción musical.
* Presentación de proyecto como innovador y social: combinación de streaming con red social musical.

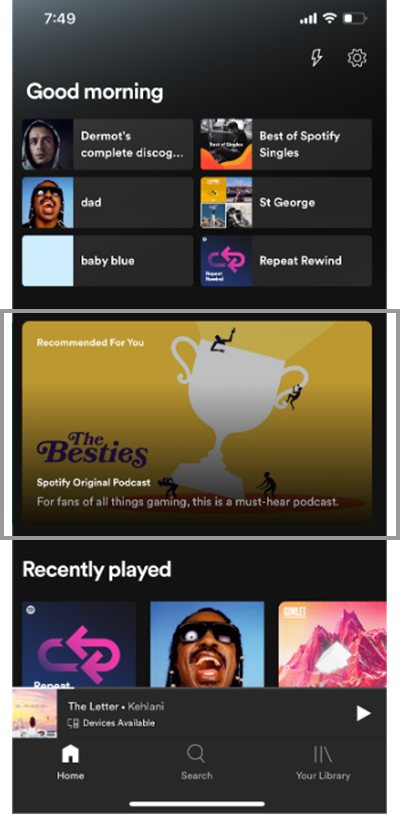
RA2 – ANÁLISIS Y PLANIFICACIÓN

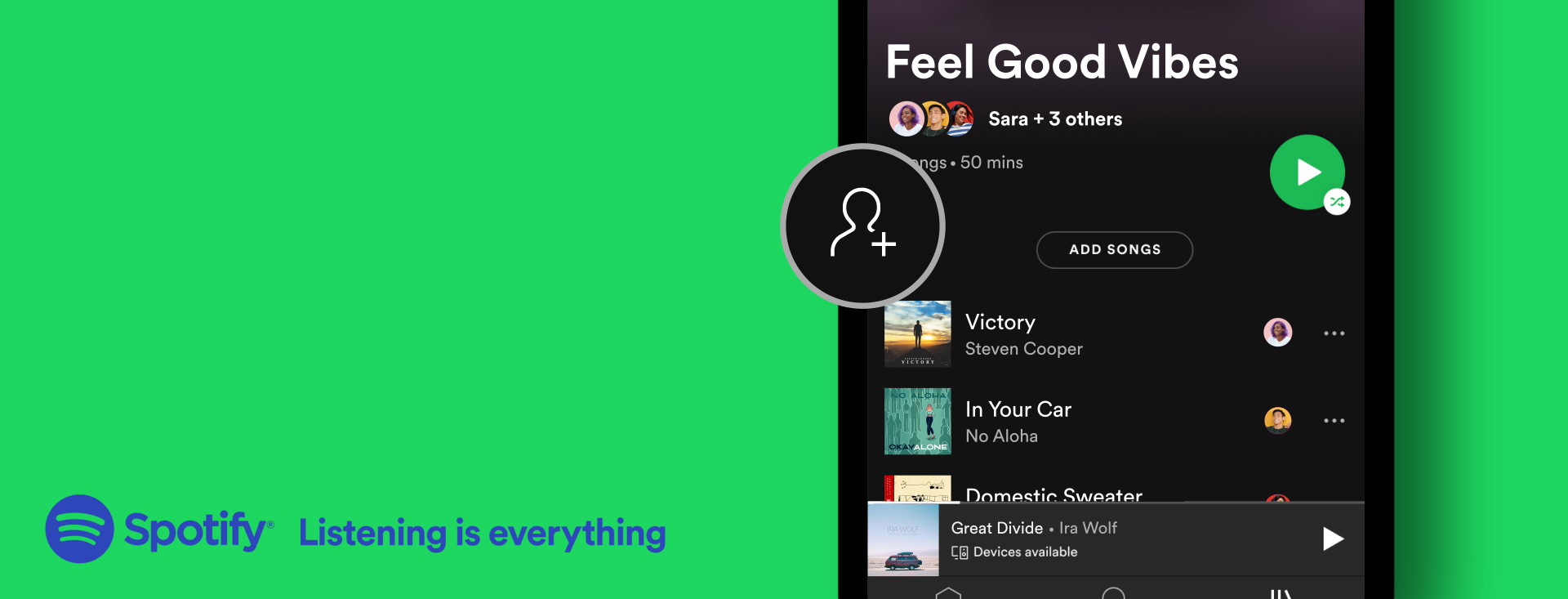
## RA2.a – Recopilación de información

Catálogo de requisitos (expandido y estructurado):  
1) Requisitos funcionales (RF) - mínimo 10 (detallados y relacionados con necesidad):  
 **RF1** Registro y autenticación de usuarios. Necesidad: control de acceso.  
 **RF2** Recuperación de contraseña por email. Necesidad: recuperación de acceso.  
 **RF3** Reproducción de canciones (play/pause). Necesidad: acceso al contenido.  
 **RF4** CRUD de listas de reproducción (crear/editar/eliminar/consultar). Necesidad: gestión personal de contenido.  
 **RF5** Añadir/Eliminar canciones en listas. Necesidad: personalización de listas.  
 **RF6** Sistema de valoraciones (estrellas/like). Necesidad: retroalimentación social.  
 **RF7** Comentarios en canciones y listas (con moderación). Necesidad: interacción social.  
 **RF8** Reporte de comentarios y gestión por administradores. Necesidad: moderación y control de contenido.  
 **RF9** Gestión de usuarios por administradores (sanciones, bloqueos). Necesidad: mantenimiento de la comunidad.  
 **RF10** Generación de informes y exportación a PDF (estadísticas básicas). Necesidad: análisis y gestión administrativa.  
 **RF11** Pago y gestión de suscripciones (alta, baja, historial). Necesidad: monetización (añadido como extra).  
  
2) Requisitos de usuario :  
 RU1 Interfaz clara y usable (diseño responsive). Necesidad: usabilidad en distintos dispositivos.  
 RU2 Escuchar música sin interrupciones y con buena calidad de reproducción. Necesidad: experiencia de usuario.  
 RU3 Compartir listas y seguir a otros usuarios. Necesidad: componente social y descubrimiento.  
  
3) Requisitos tecnológicos:  
 - **Backend**: PHP 8 con patrón MVC.  
 - **Base de datos**: MySQL 8.  
 - **Frontend**: HTML5, CSS3, JavaScript (ES6+), Bootstrap 5.  
 - **AJAX** para operaciones asíncronas en valoraciones y comentarios.  
 - Servicio **SMTP** para emails automáticos.  
  
4) Requisitos de interfaz:  
 - Accesibilidad WCAG 2.1 nivel AA.  
 - Compatibilidad con navegadores modernos.  
 - Controles de reproducción accesibles por teclado y ARIA labels.

Comparativa de spotify y deezer (mis aplicaciones de referencia)

## 1. Spotifyhttps://cms-fym.s3.nl-ams.scw.cloud/spotify_features_screenshot_2_94eac6cdb3.png

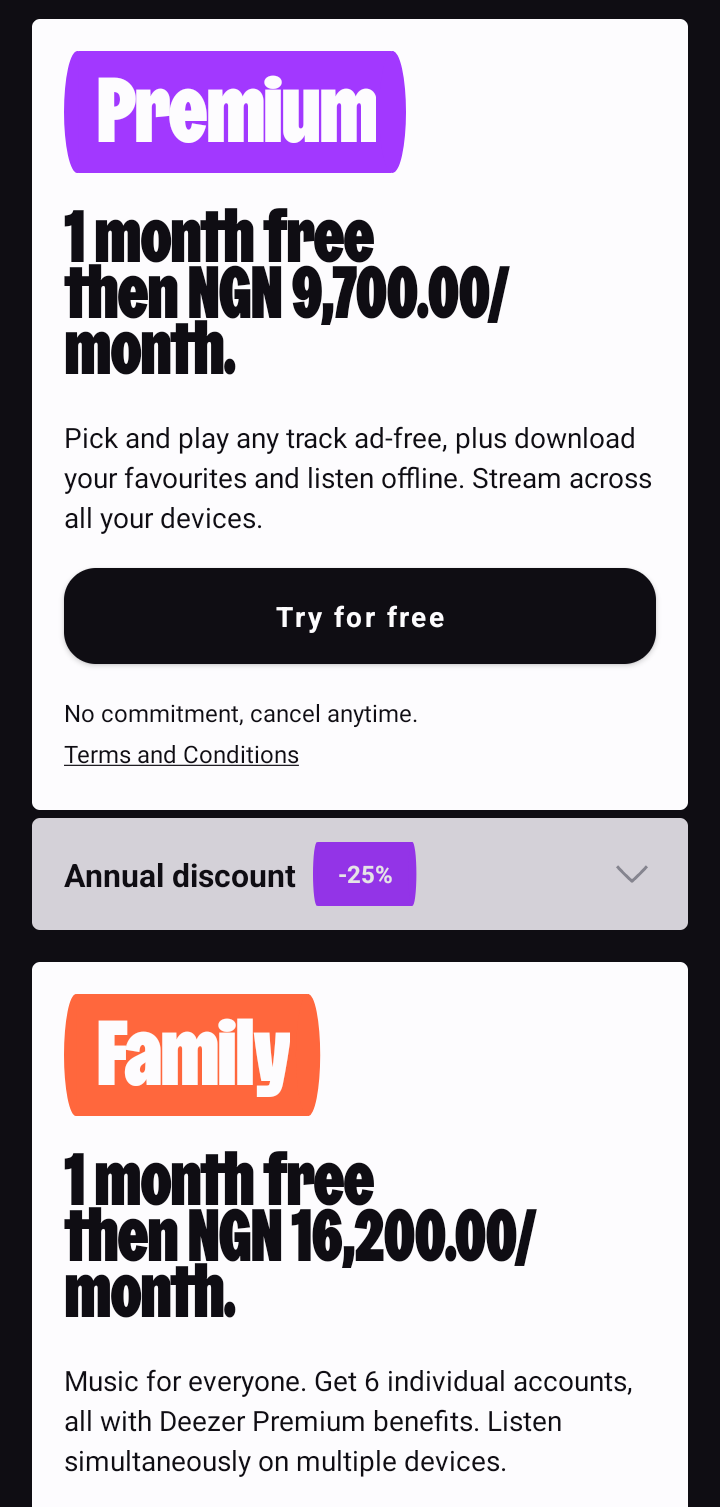




**Descripción:** Spotify es una de las plataformas de streaming de música más grandes del mundo. Ofrece acceso a millones de canciones y podcasts mediante suscripción o en versión gratuita con anuncios. [Spotify](https://es.wikipedia.org/wiki/Spotify)  
 **Puntos relevantes para mi proyecto:**

* Permite creación de listas de reproducción, compartir música y cierto grado de interacción entre usuarios (por ejemplo compartir listas o ver actividad).
* En su versión Premium ofrece experiencia sin anuncios, acceso completo al catálogo, reproducciones sin límite.
* Enfoque principalmente en consumo individual, más que en comunidad o comentarios sociales intensos

2. Deezer





**Descripción:** Deezer es otra plataforma de streaming musical con un catálogo global y presencia en muchos países. [Deezer](https://en.wikipedia.org/wiki/Deezer?utm_source=chatgpt.com) **Puntos relevantes para tu proyecto:**

* Deezer destaca por funciones de descubrimiento musical («Flow»), playlists personalizadas, buen soporte multiplataforma.
* Tiene versión gratuita con anuncios y versiones de pago sin límite.
* Aunque tiene funcionalidades sociales (listas, compartir), no se centra tanto en comunidad, comentarios o red social musical como pones en tu proyecto.

## Comparación – ¿Cómo se compara MusicConnect con Spotify y Deezer?

|  |  |  |  |
| --- | --- | --- | --- |
| **Característica** | **Spotify** | **Deezer** | **MusicConnect (tu proyecto)** |
| Creación y gestión de listas | ✅ Fuertemente soportado | ✅ Soportado | ✅ Incluido como requisito |
| Compartición de listas y actividad social | ✅ Compartir y seguir listas, ver actividad | ✅ Compartir listas | ✅ Enfoque aún mayor en interacción social: comentarios, valoraciones, reportes |
| Comentarios y comunidad de usuarios | 🔹 Limitado: más bien compartir que comentar | 🔹 Similar | ✅ Muy presente: los usuarios pueden comentar, valorar y reportar comentarios |
| Roles diferenciados (gratis vs premium) | ✅ Gratis + Premium | ✅ Gratis + Premium | ✅ Usuario normal + Usuario premium + Administrador |
| Funcionalidad de moderación/comunidad | 🔹 Moderación limitada visible | 🔹 Moderación limitada | ✅ Administrador modera comentarios, usuarios reportan y hay sistema de sanciones |
| Experiencia social como núcleo | ⚠️ Principalmente consumo individual | ⚠️ Mezcla pero aún consumo individual | ✅ Social como elemento central del producto |
| Diseño responsive y acceso multiplataforma | ✅ Alto nivel | ✅ Alto nivel | ✅ También previsto: responsive con Bootstrap y acceso desde móvil, tableta, PC |
| Innovación en comunidad musical | 🔹 En fase | 🔹 Mitad camino | ✅ Diferenciador clave de MusicConnect |
| Tecnología / arquitectura | Amplia infraestructura, gran escala | Similar | Proyecto más pequeño, enfoque educativo/prototipo, pero bien definido |

## RA2.b – Viabilidad técnica

Despliegue y accesibilidad:  
- La aplicación se desplegará en un servicio de hosting que soporte PHP 8 y MySQL (o VPS si se requiere mayor control).  
- Se garantiza accesibilidad desde móvil, tablet y ordenador mediante diseño responsive con Bootstrap 5.  
- Pruebas de compatibilidad: Chrome, Firefox, Edge, Safari (móvil y escritorio).  
- Requisitos mínimos del servidor: 2 vCPU, 2GB RAM, 50GB almacenamiento para empezar; escalable según demanda.

### -Hosting gratuito

* Ejemplo: **Infinity Free**
* Permite PHP + MySQL.
* El certificado SSL puede requerir configuración manual.
* Adecuado para demostración y pruebas.

## -Hosting de pago básico

* Proveedor recomendado: **Hostinger** o **IONOS**
* Ventajas:  
  + Mejor estabilidad
  + Certificado SSL integrado
  + Rendimiento superior

## RA2.c – Identificación de fases (Gantt resumido)

Fases principales con duración estimada (ejemplo total 6 meses):  
1. Análisis de requisitos – 3 semanas  
2. Diseño (ER, UML, UI) – 2-3 semanas  
3. Implementación Sprint 1 (Autenticación, usuarios, CRUD canciones) – 3 semanas  
4. Implementación Sprint 2 (Listas, comentarios, reportes) – 2 semanas  
5. Integración y pruebas – 3 semanas  
6. Despliegue y puesta en producción – 2 semanas  
7. Ajustes y mantenimiento inicial   
  
Dependencias:  
- Diseño antes de implementación. Autenticación antes de funciones de usuario completas.

## RA2.d - Objetivos y alcance

**Objetivo general:**  
-Desarrollar una plataforma web de *streaming musical* con funcionalidades sociales que permita el descubrimiento, interacción, gestión y reproducción de contenido musical, ofreciendo una experiencia accesible, personalizada y segura para distintos tipos de usuarios.  
  
**Objetivos específicos:**

* Implementar **búsqueda avanzada** de canciones mediante filtros y etiquetas.
* Incorporar **funcionalidades sociales**: comentar canciones/listas, compartir contenido.
* Implementar un **sistema de notificaciones** (ej mediante emails).
* Asegurar **seguridad y privacidad** de datos mediante cifrado, políticas de contraseñas y consentimiento explícito.
* Diseñar una **interfaz responsive y accesible**, cumpliendo estándares WCAG 2.1 AA.
* Integrar **sistema de anuncios**
* Implementar un **módulo de estadísticas** para administradores (actividad, canciones más reproducidas, usuarios activos).
* Facilitar la **gestión de suscripciones Premium**, incluyendo renovación y cancelación.

**Alcance (lo que incluye):**  
Plataforma web responsive (compatible con escritorio y móviles).

* Autenticación mediante JWT o similar.
* Sistema de roles y permisos con interfaz de administración.
* Subida y gestión de canciones(formato mp3)
* Creación y edición de listas de reproducción dinámicas (añadir, eliminar y reordenar canciones).
* Reproductor embebido con controles básicos (play, pause, skip, volumen, barra de progreso).
* Panel de administración con gestión de usuarios, contenido
* Envío de correos automáticos (confirmación, recuperación de contraseña, notificaciones).
* Cumplimiento legal y técnico con RGPD (gestión de datos personales) y WCAG 2.1 (accesibilidad).
* Base de datos relacional (ej. MySQL o PostgreSQL) con relaciones entre entidades.
* Documentación técnica y manual de usuario básico

**Exclusiones (lo que no incluye):**  
No se implementará una aplicación móvil nativa (solo versión web adaptable).

* No se desarrollarán algoritmos avanzados de recomendación basados en IA o machine learning.
* No se incluirá monetización real (suscripciones con pago o integración con pasarelas de pago).
* No se contemplan funciones de streaming en tiempo real (solo reproducción bajo demanda).
* No se garantiza compatibilidad con todos los formatos de audio ni almacenamiento distribuido.
* Incluir **recomendaciones personalizadas** basadas en comportamiento del usuario (reproducciones, likes, etc.).
* Sección de perfil del usuario con historial de reproducciones y estadísticas personales.

## RA2.e – Recursos necesarios

Repositorio y organización:  
- GitHub con ramas: main (producción), develop (integración), feature/xxx para cada funcionalidad.  
  
Hosting e infra:  
- Proveedor compartido o VPS; servicio SMTP (Mailgun, SendGrid o servicio del hosting).  
- Dominio registrado (ejemplo: musicconnect.example).  
  
Recursos humanos:  
- 1 desarrollador backend, 1 desarrollador frontend, 1 diseñador UI/UX, 1 tester/QA.

## RA2.f – Presupuesto económico

Tabla resumida de costes (anual estimado):  
- Dominio: 10-15 €/año  
- Hosting compartido: 50-120 €/año; VPS básico: 100-300 €/año  
- Servicio de emails (SMTP): 0-50 €/año según plan  
- Certificado SSL: 0-30 €/año (Let's Encrypt gratuito posible)  
- Mantenimiento y backups: 100-300 €/año  
  
Comparativa mínima entre dos proveedores (ejemplos):  
Proveedor A (Hostinger): dominio+hosting ~72 €/año.  
Proveedor B (IONOS/OVH): oferta inicial similar, precios y prestaciones a valorar según tráfico.

## RA2.g – Financiación

Fuentes de financiación a explorar:  
- ENISA (préstamos para startups tecnológicas)  
- Red.es (programas para digitalización)  
- Subvenciones autonómicas y fondos europeos (FEDER)  
  
Uso de la financiación:  
- Desarrollo plataforma, contratación de personal clave, servicios externos (hosting, email, monitorización).

## RA2.h - Documentación de diseño (Modelo E/R y normalización)

Propuesta de Modelo E/R con ≥8 tablas (sugeridas):  
1. usuarios (id, nombre, apellidos, email, password\_hash, tipo\_cuenta, fecha\_alta)  
2. canciones (id, titulo, artista, duracion, genero\_id, archivo\_url, fecha\_subida)  
3. generos (id, nombre)  
4. listas (id, usuario\_id, nombre, descripcion, publica)  
5. lista\_cancion (lista\_id, cancion\_id, orden) -- tabla N:M  
6. comentarios (id, usuario\_id, cancion\_id/null, lista\_id/null, texto, fecha, estado) -- entidad débil  
7. reportes (id, comentario \_id, usuario\_reporta\_id, motivo, estado, fecha)  
8. suscripciones (id, usuario\_id, tipo\_plan, fecha\_inicio, fecha\_fin) -- relación 1:1 con usuario en caso de plan activo  
9. pagos (id, suscripcion\_id, cantidad, fecha, metodo)  
  
Relaciones y cardinalidades:  
- usuarios 1:N listas (un usuario puede tener muchas listas).  
- listas N:M canciones (tabla lista\_cancion).  
- canciones 1:comentarios.  
- usuarios 1:N comentarios.  
- comentarios 1:1 reportes (un reporte apunta a un comentario concreto).  
  
Normalización: diseño pensado en 3FN: atributos atómicos, separación de entidades repetitivas (géneros, lista\_cancion), claves primarias y foráneas definidas.

## RA2.y – Control de calidad

Validaciones y pruebas:  
- Validaciones DE: campos obligatorios, formatos (email), límites de longitud.  
- Validaciones BE: comprobación de permisos, saneamiento de entradas, reglas de negocio.  
- Pruebas funcionales: login, registro, CRUD completo, valoraciones, comentarios, reportes.  
- Pruebas de usabilidad y accesibilidad (WCAG). Test con lectores de pantalla y navegación solo con teclado.  
- Listados con filtros y paginación: pruebas con datasets grandes para verificar rendimiento.

RA3 – IMPLEMENTACIÓN Y GESTIÓN

## RA3.a – Secuenciación de actividades

Para organizar el desarrollo de MusicConnect se sigue una **metodología incremental con Sprints**, estructurada en fases lógicas y dependencias entre módulos.

### ****Fases del ciclo completo (Gantt detallado)****

1. **Preparación del entorno (1 semana)**
   * Instalación de XAMPP / Hosting temporal.
   * Configuración de PHP 8, MySQL 8.
   * Estructuración de carpetas MVC.
   * Inicialización del repositorio GitHub.
   * Definición de ramas: main, develop, feature/\*.
2. **Análisis y diseño técnico (2-3 semanas)**
   * Elaboración del Diagrama E/R con 8+ tablas.
   * Modelado UML (casos de uso, clases, secuencias).
   * Diseño de la interfaz (wireframes y prototipos).
   * Elección de patrones (MVC, DAO, validación).
3. **Sprint 1 – Implementación de módulos base (3 semanas)**
   * Módulo de Usuarios:
     + Registro, login, roles, autenticación.
     + Validación FE/BE, encriptación con password\_hash().
     + Emails automáticos de bienvenida.
   * Módulo de Canciones:
     + CRUD de canciones (admin).
     + Subida de archivos MP3.
     + Reproductor básico HTML5.
   * Sistema de comentarios y valoraciones.
4. **Sprint 2 – Funcionalidades sociales (2 semanas)**
   * Listas de reproducción (CRUD completo).
   * Comentarios con AJAX.
   * Reportes y moderación por administrador.
   * Paginación y filtros en listados.
5. **Integración entre módulos (1-2 semanas)**
   * Pruebas de compatibilidad entre usuarios, listas, canciones, comentarios.
   * Comprobación de permisos según rol.
   * Optimización de consultas SQL.
6. **Pruebas globales (2 semanas)**
   * Pruebas funcionales, usabilidad, accesibilidad WCAG 2.1.
   * Pruebas en móviles y distintos navegadores.
   * Validación de emails SMTP.
7. **Despliegue en hosting (1 semana)**
   * Subida al hosting (Hostinger, InfinityFree…).
   * Configuración de dominio y HTTPS.
   * Migración de base de datos.
   * Pruebas finales en producción.

### ****Dependencias esenciales****

* Autenticación debe existir antes de probar interacciones usuario-canción.
* CRUD de canciones debe existir antes de listas.
* Listas deben existir antes de valoraciones y comentarios.
* Reportes dependen de comentarios funcionando.

## RA3.b – Recursos y logística

## Recursos Hardware

* PC con mínimo **8GB RAM** (ideal 16GB).
* Monitor adicional recomendado para desarrollo Frontend.
* Hosting para despliegue:
  + Opción gratuita: InfinityFree.
  + Opción profesional: Hostinger / IONOS.

## Recursos Software

* **Backend**
  + PHP 8
  + Composer para posibles librerías
  + Servidor Apache
  + PHPMailer para emails
* **Base de datos**
  + MySQL 8
  + phpMyAdmin para gestión local
* **Frontend**
  + HTML5, CSS3, JS (ES6)
  + Bootstrap 5
  + SweetAlert2 (notificaciones)
  + AJAX para interacción fluida
* **Control de versiones y gestión**
  + Git / GitHub (commits y ramas)
  + Trello para tareas
  + Visual Studio Code como IDE

### ****Logística del equipo****

* Uso de GitHub Projects + Trello como seguimiento dual.

## RA3.c – Permisos y autorizaciones

### ****Base de Datos****

Se definen distintos niveles de permisos para proteger la información:

* **Usuario de aplicación (app\_user):**
  + SELECT, INSERT, UPDATE, DELETE solo en tablas del proyecto.
  + No puede modificar estructuras.
* **Administrador de BD (admin\_user):**
  + Permisos para ALTER, CREATE, DROP.
  + Solo para despliegue y mantenimiento.

### ****Hosting****

Configuración obligatoria:

* Creación de archivo .env con:
  + DB\_HOST, DB\_USER, DB\_PASS, DB\_NAME
  + SMTP\_HOST, SMTP\_PORT, SMTP\_USER, SMTP\_PASS
* Activación de **SSL / HTTPS obligatoria**.

### ****SMTP****

* TLS o SSL para enviar emails.
* Protecciones:
  + Límite de correos por hora.
  + Logs de envío.
  + Claves SMTP en variables de entorno (no en código).

## RA3.d – Procedimientos de actuación

### ****Metodología GitHub + Trello****

1. Cada tarea se crea como **Tarjeta en Trello**.
2. Por cada tarea se crea una **rama Git**:
   * feature/login
   * feature/comentarios
   * feature/reportes
3. Cuando la funcionalidad está lista:
   * Push → Pull Request hacia develop.
4. Revisión de código:
   * Se validan estándares, seguridad, estilo.
5. Merge solo tras aprobación.
6. Integración final en main antes de despliegue.

### ****Buenas prácticas de commits****

Formato recomendado:

feat: añadido sistema de reportes

fix: corregido fallo en paginación

style: limpieza de indentación

refactor: optimizado módulo de usuarios

## RA3.e – Riesgos y prevención

**1. Pérdida de datos por fallo del servidor**

**Riesgo:** corrupción de base de datos, error de hosting.  
**Prevención:**

* Backups automáticos diarios.
* Copias en almacenamiento externo (Google Drive o S3).
* Restauración probada antes del despliegue.

**2. Vulnerabilidades de seguridad**

**Riesgo:** SQL Injection, XSS, fuga de datos.  
**Prevención:**

* Uso de consultas preparadas PDO.
* Sanitización de comentarios y nombres.
* Contraseñas cifradas con password\_hash().
* Roles estrictos con ACL.

**3. Retrasos por dependencias externas**

**Riesgo:** fallos de hosting, esperas por pruebas o APIs.  
**Prevención:**

* Planificación con margen (buffer del 10%).
* Alternativas de hosting en caso de fallo.
* Priorizar módulos críticos primero.

**4. Carga excesiva del servidor por MP3**

**Prevención:**

* Límite de tamaño por canción.
* Cacheado de consultas.
* Optimización de imágenes y recursos.

## RA3.f – Asignación de recursos y tiempos

### ****Roles del equipo****

* **Coordinador / Product Owner:** supervisión de requisitos y entregables.
* **Desarrollador Backend:**
  + Controladores
  + Modelos
  + Seguridad
  + Emails
* **Desarrollador Frontend:**
  + Diseño
  + Interfaz
  + Integración Bootstrap
  + Validaciones JS
* **QA Tester:**
  + Pruebas funcionales
  + Informes de errores
  + Validación accesibilidad

### ****Tiempos****

Ya reflejados en el Gantt del RA2.c, estimados entre **10 y 14 semanas** de desarrollo activo.

## RA3.g – Valoración económica implementación

### ****Costes iniciales****

| **Concepto** | **Coste estimado** |
| --- | --- |
| Desarrollo (6 meses) | 4.000 – 8.000 € (según tarifas) |
| Hosting + Dominio | 120 €/año |
| Emails (SMTP) | 0 – 50 €/año |
| Herramientas de diseño (Figma/Canva) | 0 – 12 €/mes |
| Backups y monitorización | 50 – 200 €/año |

### ****Resumen****

Implementar MusicConnect cuesta entre **4.200 y 8.500 €**, dependiendo de si es un proyecto profesional o académico.

## RA3.h – Documentación de implementación

# **Requisitos previos**

Para que MusicConnect funcione, se necesita:

### ****Servidor****

* PHP 8+
* MySQL 8
* Apache (o nginx)

### ****Librerías****

* PHPMailer para emails
* Bootstrap 5
* OwlCarousel (solo para sliders)
* jQuery
* FontAwesome

# **2️ Instalación local (XAMPP)**

### ****Pasos****

1. Copiar la carpeta del proyecto a:  
   C:\xampp\htdocs\MusicConnect
2. Crear base de datos bdreproductor.
3. Importar el archivo bdreproductor.sql.
4. Configurar el archivo /include/config.php:

define("BASE\_URL", "http://localhost/MusicConnect/");

define("DB\_HOST", "localhost");

define("DB\_NAME", "bdreproductor");

define("DB\_USER", "root");

define("DB\_PASS", "");

1. Iniciar Apache + MySQL desde XAMPP.
2. Entrar a la web:  
   http://localhost/MusicConnect/

# **3️ Despliegue en hosting**

1. Subir los archivos al dominio (FTP o panel web).
2. Crear base de datos en el hosting.
3. Editar config.php con los nuevos datos:

define ("DB\_USER", "usuario\_hosting");

define ("DB\_PASS", "contraseña\_hosting");

define ("BASE\_URL", "https://midominio.com/");

1. Importar bdreproductor.sql en phpMyAdmin del hosting.
2. Configurar SMTP desde el panel:

$mail->Host = "smtp.hosting.com";

$mail->Username = "correo@midominio.com";

$mail->Password = "contraseña";

1. Probar:

* Login
* Emails
* Subida de canciones
* Reproducción

# **4️ Uso del repositorio GitHub**

### ****Estructura recomendada****

* main: versión estable
* dev: desarrollo general
* feature/comentarios
* feature/listas
* feature/subidas

Cada nueva función → nueva rama.

# **5️ Evidencias de pruebas**

Incluye capturas:

* Registro correcto
* Login correcto
* Crear lista
* Subir canción
* Comentar una canción
* Reportar comentario
* Vista del panel admin
* Responsive en móvil

RA4 – EVALUACIÓN Y CONTROL

**RA4.a – Procedimiento de evaluación (DETALLADO)**

El procedimiento de evaluación garantiza que cada parte del proyecto cumple con los requisitos funcionales, técnicos y de calidad.

**Fases de evaluación:**

1. **Definición de criterios de aceptación**
   * Cada funcionalidad (registro, login, subida de canciones, reportes, reproductor…) se documenta con:
     + Qué debe hacer exactamente.
     + Qué datos necesita.
     + Qué resultado debe mostrar.
2. **Creación de Checklist por RA en Trello o GitHub Projects**  
   Cada RA (RA1, RA2, RA3, RA4) tiene sus tareas divididas en:
   * Pendiente
   * En proceso
   * En revisión
   * Completado
3. **Revisión por QA (o persona responsable de pruebas)**  
   Comprueba:
   * Funcionalidad
   * Seguridad (login seguro, sesiones)
   * Interfaz (fallos visuales, responsive)
   * Base de datos (insertar, modificar, eliminar correctamente)
4. **Aprobación final por el coordinador del proyecto**  
   Un responsable revisa que la funcionalidad:
   * Está correctamente implementada
   * No rompe otra parte del sistema
   * Cumple los requisitos mínimos del pliego

**Resultado final**

Cuando todo está validado → puede pasarse a producción.

**RA4.b – Indicadores de calidad (MUY DETALLADOS)**

Para medir objetivamente la calidad del proyecto se definen KPIs.

**Indicadores técnicos:**

1. **Validaciones FE/BE**
   * Objetivo: **>95% de formularios validados correctamente**
   * FE: evitar campos vacíos, emails inválidos, etc.
   * BE: sanitizar datos, evitar duplicados, prevenir SQL Injection.
2. **Rendimiento del sistema**
   * **Tiempo de carga**: < 2 segundos página principal.
   * **Consultas BD**: < 300 ms para:
     + Mostrar listas
     + Filtrar canciones
     + Mostrar canciones por usuario
3. **Emails automáticos funcionando al 100%**
   * Mensaje de registro
   * Recuperación o cambio de contraseña
   * Resolución de reporte por parte del admin
4. **Responsive completo**  
   Se prueban **10 combinaciones de dispositivos**, por ejemplo:
   * iPhone
   * Android
   * iPad
   * Portátil
   * Pantalla grande  
     Resultados esperados:
   * Menús accesibles
   * No se desconfigura el diseño
   * Reproductor accesible en móvil
5. **Estabilidad del sistema**
   * No debe tener caídas durante las pruebas.
   * Sin errores graves en consola o PHP.

**RA4.c – Gestión de incidencias (SUPER DETALLADO)**

**Procedimiento de reporte de incidencias:**

1. **El usuario detecta un problema**  
   Puede ser:
   * Error al registrarse
   * No se carga una canción
   * El reproductor falla
   * Problemas de sesión
2. **Crea un reporte desde la aplicación**  
   Tu app tiene un sistema de “reportes” → se envía a admin.
3. **Admin revisa el reporte**  
   En el panel puede:
   * Ver el mensaje del usuario
   * Ver fecha y estado
   * Marcar como:
     + Pendiente
     + En análisis
     + Resuelto
4. **Emails automáticos** (mínimo 3, ya adaptados):
   * ✔ **Email de confirmación de registro**
   * ✔ **Email de acción clave (bloqueo, cambio de contraseña, etc.)**
   * ✔ **Email al resolver un reporte**
5. **Documentación obligatoria por reporte**
   * Pasos para reproducirlo
   * Resultado esperado
   * Resultado obtenido
   * Capturas
   * Responsable asignado
   * Tiempo estimado

**RA4.d – Gestión de cambios (explicado al detalle)**

**✔ Política de cambios:**

Cualquier propuesta de mejora o modificación sigue estos pasos:

1. **Se registra como issue en GitHub/Trello**  
   Debe incluir:
   * Descripción del cambio
   * Justificación
   * Impacto esperado
2. **Planificación en Sprint**  
   Los cambios se organizan en ciclos de trabajo (1-2 semanas).
3. **Uso de ramas (branching)**
   * Rama principal: **main** (producción)
   * Rama de desarrollo: **develop**
   * Rama para cada mejora: **feature/nombre-cambio**
4. **Pull Request + Revisión obligatoria**  
   Nadie sube algo a main sin revisión previa.
5. **Versionado Semántico**  
   Ejemplo:
   * 1.0.0 → Versión inicial
   * 1.1.0 → Se añade nueva funcionalidad (listas)
   * 1.1.1 → Se corrige bug menor

**RA4.e – Documentación de evaluación (MUY EXTENDIDO)**

Debes documentar al menos **5 pruebas completas**, incluyendo:

**Prueba 1: Login**

* Resultado esperado: accede correctamente
* Resultado obtenido: OK
* Evidencias: captura pantalla

**Prueba 2: Registro**

* Validación de email, contraseñas iguales
* Envío de email automático
* Inserción en BD
* Resultado: OK

**Prueba 3: Crear lista + añadir canción**

* BD actualiza tablas playlist/playlist\_song
* Reproductor reconoce la lista
* Resultado: OK

**Prueba 4: Crear reporte + admin lo modera**

* Usuario envía reporte
* Admin cambia estado
* Email automático al usuario
* Resultado: OK

**Prueba 5: Responsive**

* Se prueba en móvil
* Menú visible
* Reproductor no se deforma
* Resultado: OK

**Además:**

* Clasificación de fallos:
  + Críticos
  + Medios
  + Leves
* Plan de corrección detallado para los fallos.

**RA4.f – Participación de usuarios (DETALLADÍSIMO)**

**✔ Encuestas de satisfacción**

Preguntas típicas:

* ¿Es fácil subir canciones?
* ¿La interfaz es intuitiva?
* ¿El reproductor funciona bien?
* ¿Qué mejorarías?

**✔ Accesibilidad**

Se validará con:

* Herramientas:
  + WAVE
  + Axe
* Pruebas reales con personas:
  + móvil
  + tablet
  + PC
* Comprobaciones:
  + Colores accesibles
  + Botones con texto alternativo
  + Navegación clara

**RA4.g – Cumplimiento del Pliego**

El proyecto se revisa para confirmar que:

**Obligaciones cumplidas:**

* CRUD de canciones
* CRUD de listas
* Login / Registro / Roles
* Panel Admin
* Sistema de reportes
* Reproductor funcional
* Emails automáticos
* Navegación dinámica según sesión
* Estándares mínimos de calidad

**Comprobación final:**

Se realiza un “checklist” final con todas las evidencias:

* Capturas
* Logs
* Pruebas
* Documentos de implementación
* Documentos de configuración