# ONDAS 4x4

**Autor: Adolfo Rendón Fernández**

**Tutor: Pedro González Gordillo**

**Fecha de entrega: 30/05/2024**

**Convocatoria:** 2023/2024

INDICE

# INTRODUCCIÓN

En la actualidad, la música hip hop ha ganado una enorme popularidad entre el público joven. Esta subcultura musical ha trascendido sus raíces urbanas para convertirse en un fenómeno global que resuena profundamente con las nuevas generaciones. Sin embargo, a pesar de su popularidad, una gran cantidad de música hip hop, especialmente la creada por artistas emergentes y underground, no está disponible en las plataformas de streaming convencionales como Spotify, Apple Music o Amazon Music. Esta música, a menudo libre de derechos de autor, está dispersa a lo largo de múltiples plataformas y medios, como SoundCloud, Bandcamp, YouTube y blogs de música independientes. Esta dispersión dificulta el acceso centralizado y continuo por parte de los oyentes, que deben navegar entre diversas fuentes para encontrar y disfrutar de sus canciones favoritas.

Este problema crea una brecha significativa en el mercado, dejando a los jóvenes sin un medio eficiente para acceder a su música preferida. Los usuarios, en su afán por descubrir nueva música, deben invertir mucho tiempo y esfuerzo en buscar canciones en diferentes sitios web y aplicaciones, lo que puede resultar frustrante y desalentador. Además, la falta de una plataforma unificada limita la visibilidad de los artistas emergentes, que dependen de la difusión y el acceso fácil a su música para construir una base de seguidores y crecer en su carrera.

La aplicación de radio y música hip hop sin copyright que se desarrolla en este proyecto está orientada precisamente a llenar este vacío. Al aglutinar todos estos contenidos en una sola plataforma accesible y fácil de usar, se facilita significativamente el acceso a esta música para los oyentes jóvenes. La aplicación no solo centraliza la música de artistas emergentes y underground, sino que también ofrece una plataforma donde estos artistas pueden ser descubiertos y apreciados por una audiencia más amplia. Esto no solo mejora la experiencia del usuario, sino que también apoya el crecimiento y la sostenibilidad de la comunidad musical independiente.

## Justificación del Tema Seleccionado

El motivo principal para seleccionar este tema radica en la creciente demanda de los jóvenes por un acceso más conveniente y centralizado a la música hip hop libre de derechos de autor. La juventud actual, más conectada y tecnológicamente avanzada, busca constantemente nuevas experiencias musicales y valora la autenticidad y originalidad que a menudo se encuentra en la música independiente. Estos jóvenes no

necesariamente recurren a métodos ilegales para acceder a esta música, ya que muchas de estas obras están disponibles gratuitamente en diversas plataformas legales. Sin embargo, la dispersión de estos contenidos a través de múltiples medios y la falta de una plataforma unificada representan una barrera significativa para su disfrute continuo y sin interrupciones.

Una aplicación dedicada que pueda centralizar este tipo de contenido no solo facilitaría el acceso a esta música, sino que también apoyaría a los artistas independientes al darles una plataforma para llegar a más oyentes. Al centralizar la música libre de derechos de autor en una sola aplicación, se crea un ecosistema donde los artistas pueden compartir su trabajo de manera más efectiva y donde los oyentes pueden descubrir nueva música sin las frustraciones asociadas con la búsqueda en múltiples plataformas. Además, este tipo de aplicación es innovadora y única en el mercado, ofreciendo una solución que actualmente no existe y, por tanto, es altamente relevante y necesaria.

Esta justificación se basa en la observación de la tendencia creciente entre el público joven de buscar alternativas musicales fuera del dominio de las grandes plataformas comerciales, y su disposición a explorar y apoyar música libre de derechos de autor. La aplicación no solo proporciona una solución técnica a un problema práctico, sino que también tiene el potencial de transformar la manera en que la música independiente es descubierta y consumida. Al ofrecer una plataforma accesible y centralizada, se facilita el acceso a música de calidad y se promueve la diversidad musical.

## Resumen del Contenido del Trabajo

Este trabajo detalla el desarrollo de una aplicación Android que proporciona un servicio de radio y una plataforma de música hip hop sin copyright, destinada principalmente a un público joven. La aplicación está diseñada para ser intuitiva y fácil de usar, con una interfaz moderna que resuena con la estética y las preferencias del público objetivo. Permite a los usuarios registrarse, iniciar sesión, y gestionar sus listas de reproducción favoritas, proporcionando una experiencia personalizada y coherente.

La aplicación también incluye funcionalidades avanzadas para los administradores, que tienen la capacidad de añadir, modificar y eliminar listas de música y clientes. Esto asegura que la base de datos esté siempre actualizada y relevante, permitiendo una gestión eficiente del contenido y de los usuarios. Los administradores pueden monitorear el uso de la aplicación, gestionar la base de datos de usuarios y listas de reproducción, y asegurar que la plataforma funcione de manera óptima.

En términos técnicos, el proyecto utiliza una combinación de tecnologías modernas y prácticas de desarrollo de software para garantizar un rendimiento óptimo y una experiencia de usuario superior. Se hace uso de SQLite para la gestión de la base de

datos, asegurando que los datos sean almacenados y recuperados de manera eficiente. La aplicación también integra servicios como ExoPlayer para la reproducción de radio en vivo, proporcionando una experiencia auditiva continua y de alta calidad.

Además, se implementa un plan de pruebas exhaustivo para asegurar que todas las funcionalidades de la aplicación funcionen correctamente y que cualquier problema se identifique y resuelva de manera oportuna. Este plan de pruebas incluye tanto casos de prueba de camino feliz (happy path) como casos de prueba de camino infeliz (unhappy path), asegurando que la aplicación maneje adecuadamente tanto el uso esperado como las condiciones de error.

En resumen, este proyecto representa una solución innovadora y necesaria para la centralización y accesibilidad de la música hip hop libre de derechos de autor. Al proporcionar una plataforma unificada y fácil de usar, la aplicación facilita el acceso a música de calidad, apoya a los artistas independientes y responde a una necesidad real y creciente en el mercado. Este trabajo documenta de manera exhaustiva todas las etapas del desarrollo del proyecto, desde el análisis de requisitos hasta la implementación y las pruebas, asegurando una base sólida para su futura expansión y mejora.

# MOTIVACIÓN

Desde mi adolescencia, la música hip hop ha sido una parte integral de mi vida. Me fascinaba descubrir nuevos artistas, especialmente aquellos que no estaban firmados por grandes discográficas y que compartían su música libremente, fuera de los circuitos comerciales. Durante años, he recopilado una extensa colección de música hip hop sin copyright, explorando rincones ocultos de la web para encontrar esos tesoros auditivos que no se encontraban en plataformas convencionales como Spotify. Esta pasión por la música independiente me permitió conectar con una comunidad global de artistas y oyentes que valoran la autenticidad y la creatividad por encima de las ganancias comerciales.

Paralelamente, mi interés por la tecnología y la programación me llevó a profundizar en el desarrollo de aplicaciones móviles. Android Studio se convirtió en mi herramienta favorita, ofreciéndome la libertad y el desafío técnico que buscaba. Con el tiempo, desarrollé numerosas aplicaciones como parte de mi aprendizaje y proyectos personales, disfrutando cada momento de escribir código y ver cómo las ideas cobraban vida en la pantalla de un smartphone.

La idea de crear una aplicación de radio y música hip hop sin copyright surgió como una confluencia natural de mis dos grandes pasiones: la música y la programación. Observando la dificultad que enfrentan muchos jóvenes para acceder de manera centralizada a música hip hop independiente, vi una oportunidad clara. En el mercado actual, no existe una plataforma dedicada que aglutine este tipo de contenido, y percibí un vacío que podría ser llenado con una solución innovadora. Esta aplicación no solo facilitaría el acceso a música libre de derechos, sino que también ofrecería una plataforma para que los artistas independientes lleguen a una audiencia más amplia sin necesidad de recurrir a métodos ilegales para compartir su trabajo.

Además de la satisfacción personal de combinar mis hobbies en un proyecto tangible, también vislumbro un potencial comercial significativo. La creciente demanda de alternativas a las grandes plataformas de streaming y el interés por el contenido independiente y auténtico sugieren que esta aplicación podría resonar con un público joven y entusiasta. Al desarrollar esta aplicación, no solo estoy persiguiendo una pasión personal, sino que también estoy explorando una oportunidad de negocio que podría tener un impacto real en la forma en que se consume y se valora la música hip hop. Este proyecto es el resultado de años de recopilación, aprendizaje y una visión clara de cómo la tecnología puede mejorar la accesibilidad y el disfrute de la música.

# ABSTRACT

This project involves the development of a comprehensive Android application designed for both users and administrators. The application includes key functionalities such as user registration, login, and role management, allowing differentiation between regular users and administrators. Administrators have additional privileges to manage lists and clients. Users can add, modify, delete, and view lists, as well as mark and view favorite lists. The app features robust navigation through a main navigation bar and a sidebar menu, along with a service for radio playback. Key non-functional requirements include usability with an intuitive interface, performance with optimized resource usage, security through encrypted passwords and data protection, compatibility with various devices and Android versions, maintainability through modular code and documentation, scalability to handle data and user growth, high availability of services, and accessibility support for users with disabilities. The project uses tools such as Android Studio, Java, and XML, and follows a modular code structure with detailed UML diagrams for system architecture, ensuring a well-organized and efficient application.

# OBJETIVOS

## Objetivos Generales

1. Desarrollar una estación de radio online multiplataforma que ofrezca una experiencia de usuario óptima y funcional en diversas plataformas y dispositivos.

2. Garantizar la seguridad y privacidad de la información mediante la implementación de políticas y estrategias de protección de datos.

3. Ofrecer una interfaz de usuario intuitiva y atractiva que mejore la experiencia de los usuarios y facilite la navegación y el uso de la aplicación.

4. Fomentar la integración con plataformas externas y redes sociales para aumentar la visibilidad y distribución del contenido.

5. Implementar una metodología de desarrollo ágil y bien documentada para asegurar una gestión eficiente del proyecto y la adaptabilidad a cambios.

## Objetivos Específicos

1. Desarrollar una aplicación web y Android utilizando Java para Android y HTML, CSS, y JavaScript para la web.

2. Asegurar compatibilidad y optimización en diversos dispositivos y navegadores.

3. Implementar un sistema de manejo de usuarios con autenticación, gestión de perfiles y personalización.

4. Diseñar una base de datos escalable y segura para gestionar el contenido de la radio y las interacciones del usuario.

5. Implementar encriptación para proteger información sensible y garantizar privacidad.

6. Desarrollar un plan de marketing con identidad visual y promoción en diversos medios.

# METODOLOGÍA

La metodología Scrum es ideal incluso para proyectos en solitario de desarrollo de aplicaciones Android debido a su flexibilidad, capacidad de adaptación a los cambios, enfoque en la entrega continua de valor y mejora continua. Estas características aseguran que puedas manejar los desafíos del proyecto de manera eficiente, manteniendo una alta calidad del producto y satisfaciendo las necesidades del usuario final. Al adoptar Scrum, puedes maximizar la productividad, minimizar riesgos y garantizar que el proyecto se mantenga en el buen camino hacia el éxito.

**1. Flexibilidad y Adaptabilidad**
Incluso en un proyecto en solitario, la capacidad de adaptarse rápidamente a los cambios es crucial. Las prioridades y requisitos pueden cambiar frecuentemente debido a la retroalimentación del usuario, nuevas ideas innovadoras o descubrimientos durante el desarrollo. Scrum permite realizar ajustes en cada sprint, garantizando que puedas adaptarte a las nuevas circunstancias sin grandes interrupciones en tu flujo de trabajo.

**2. Iteraciones Cortas y Metas Frecuentes**
La estructura de Scrum, basada en sprints de dos a cuatro semanas, es ideal para mantener un ritmo constante de trabajo y alcanzar metas de forma regular. Esto te permite completar incrementos funcionales de la aplicación Android con regularidad. Las metas frecuentes facilitan la detección temprana de problemas y la recepción de retroalimentación continua, permitiendo ajustes oportunos. Para un proyecto en solitario, esto significa que puedes lanzar versiones beta de la aplicación para pruebas tempranas y recibir comentarios valiosos para mejorar el producto continuamente.

**3. Transparencia y Visibilidad**
Scrum promueve la transparencia y la visibilidad del progreso del proyecto. Aunque trabajes solo, las prácticas de mantener un backlog y revisar el progreso al final de cada sprint te ayudan a mantener una visión clara de tus objetivos y avances. Esto es vital para mantener la motivación y la eficiencia, ya que tienes una comprensión constante de lo que se ha logrado y lo que queda por hacer.

**4. Mejora Continua**
La metodología Scrum incluye la práctica de realizar retrospectivas al final de cada sprint. Esto te permite reflexionar sobre lo que salió bien, lo que se puede mejorar y cómo aplicar esos aprendizajes en los siguientes sprints. Este enfoque de mejora

continua es crucial en el desarrollo de software, donde siempre debes buscar formas de mejorar tus procesos y productos.

**5. Gestión Eficiente del Riesgo**
La naturaleza iterativa de Scrum permite la identificación y mitigación temprana de riesgos. Al dividir el proyecto en sprints manejables y revisar el progreso al final de cada sprint, puedes abordar los problemas antes de que se conviertan en grandes obstáculos. Esto es especialmente importante en el desarrollo de aplicaciones móviles, donde los problemas técnicos o cambios en los requisitos pueden tener un impacto significativo si no se gestionan a tiempo.

**6. Estructura y Disciplina**
Trabajar en solitario puede ser un desafío en términos de mantener la disciplina y la estructura. Scrum proporciona un marco claro para planificar, ejecutar y revisar el trabajo, ayudándote a mantenerte organizado y enfocado. Esto es especialmente útil para asegurar que no se pierda de vista ningún detalle importante y que se mantenga un progreso constante hacia la finalización del proyecto.

# Ciclo de Vida del Proyecto

## Semana 1: Planificación y Configuración Inicial

*Objetivos*:
Establecer el entorno de desarrollo.
Planificar la arquitectura de la aplicación.
Crear la base del proyecto y configurar las dependencias necesarias.

*Tareas*:
Establecer el Entorno de Desarrollo:
Instalar Android Studio y las herramientas necesarias.
Configurar un repositorio de control de versiones (GitHub, GitLab, etc.).

Planificación de la Arquitectura:
Diseñar la arquitectura de la aplicación (MVC, MVVM, etc.).
Definir las clases principales y sus responsabilidades.

Creación del Proyecto:
Crear un nuevo proyecto en Android Studio.
Configurar las dependencias en build.gradle (app y proyecto).

Configuración de Recursos:
Definir los colores, estilos y temas de la aplicación en archivos XML.
Configurar el AndroidManifest.xml con los permisos necesarios.

Implementación de Clases Base:
Crear clases base como DbHelper, SessionManager, y otros utilitarios.

Revisión:
Asegurarse de que el entorno está correctamente configurado.
Revisar y ajustar la arquitectura según sea necesario.

## Semana 2: Desarrollo de Funcionalidades Básicas

*Objetivos*:
Implementar la funcionalidad de registro y login.
Crear la estructura básica de la interfaz de usuario.

*Tareas*:
Registro y Login:
Implementar la MainActivity con el formulario de login.
Crear la RegistroActivity para la funcionalidad de registro.
Añadir validación de campos y manejo de errores.
Encriptación de contraseñas y almacenamiento seguro en SQLite.

Interfaz de Usuario Básica:
Diseñar los layouts principales (activity_main.xml, activity_registro.xml).
Implementar la navegación básica entre actividades.

Configuración de SharedPreferences:
Implementar la lógica para guardar y recuperar datos de usuario.
Crear métodos en SessionManager para gestionar la sesión del usuario.

Pruebas Iniciales:
Realizar pruebas básicas de las funcionalidades de registro y login.
Asegurarse de que los datos se almacenan y recuperan correctamente.

Revisión:
Validar que el registro y login funcionan sin problemas.
Revisar la interfaz de usuario para asegurarse de que es intuitiva.
Semana 3: Desarrollo de Funcionalidades Avanzadas y UI Mejorada

## Semana 3: Desarrollo de Funcionalidades Avanzadas

*Objetivos*:
Implementar funcionalidades avanzadas (listas, clientes).
Mejorar la interfaz de usuario y la navegación.

*Tareas*:
Funcionalidad de Listas:
Crear ListasFragment con opciones para añadir, modificar, eliminar y mostrar listas.
Implementar MostrarListaFragment para visualizar las listas.

Funcionalidad de Clientes:
Crear ClienteFragment con opciones para añadir, modificar, eliminar y mostrar clientes.
Implementar MostrarClienteFragment para visualizar los clientes.

Mejora de UI:
Diseñar y aplicar layouts más avanzados y responsivos.
Implementar RecyclerView para mostrar listas y clientes de forma eficiente.

Integración de Glide:
Utilizar Glide para cargar imágenes en RecyclerView y otros componentes visuales.

Pruebas y Ajustes:
Realizar pruebas de las nuevas funcionalidades y ajustar según sea necesario.
Asegurarse de que todas las funcionalidades son accesibles y funcionan correctamente.

Revisión:
Validar que las funcionalidades de listas y clientes están completamente implementadas.
Revisar la consistencia y usabilidad de la interfaz de usuario.

## Semana 4: Servicios y Notificaciones

*Objetivos*:
Implementar servicios en segundo plano para la reproducción de radio.
Configurar notificaciones persistentes.

*Tareas*:
Implementación de RadioService:
Crear el servicio RadioService utilizando ExoPlayer.
Implementar métodos para iniciar y detener la reproducción de radio.

Notificaciones Persistentes:
Configurar NotificationManager para mostrar notificaciones mientras el servicio está activo.
Asegurarse de que las notificaciones permiten controlar la reproducción (play, pause).

Pruebas de Servicios:
Realizar pruebas exhaustivas para asegurarse de que el servicio de radio funciona correctamente en segundo plano.
Verificar que las notificaciones se muestran y actualizan adecuadamente.

Optimización de Recursos:
Asegurarse de que los recursos (CPU, memoria) se utilizan de manera eficiente durante la reproducción de radio.
Implementar métodos para liberar recursos cuando el servicio no esté activo.

Revisión:
Validar que el servicio de radio y las notificaciones funcionan sin problemas.
Revisar el uso de recursos y optimizar si es necesario.

## Semana 5: Integración Final y Pruebas Complejas

*Objetivos*:
Integrar todas las funcionalidades y realizar pruebas complejas.
Preparar la aplicación para su lanzamiento.

*Tareas*:
Integración de Funcionalidades:
Asegurarse de que todas las funcionalidades están integradas y funcionan de manera cohesiva.
Resolver cualquier conflicto que surja durante la integración.

Pruebas de Usuario:
Realizar pruebas con usuarios finales para obtener feedback sobre la usabilidad y funcionalidad.
Ajustar la aplicación según el feedback recibido.

Documentación:
Documentar el código y las funcionalidades implementadas.
Crear guías de usuario y desarrollador para facilitar el uso y mantenimiento de la aplicación.

Preparación para el Lanzamiento:
Asegurarse de que la aplicación cumple con las directrices de Google Play.
Crear los archivos APK y AAB necesarios para la distribución.
Subir la aplicación a Google Play (si aplica) y configurar los detalles de la publicación.

Revisión:
Validar que todas las funcionalidades están completamente implementadas y funcionan correctamente.
Asegurarse de que la aplicación está lista para su lanzamiento y cumple con todos los requisitos.

# TECNOLOGÍA Y HERRAMIENTAS

Este proyecto Android hace uso de una variedad de tecnologías y herramientas para proporcionar una experiencia de usuario robusta y funcional tanto para usuarios normales como para administradores. A continuación, se describen y enumeran las principales tecnologías y herramientas utilizadas, incluyendo lenguajes de programación, editores de código, frameworks, bibliotecas, y otras utilidades esenciales.

## Lenguajes de Programación

Java: Utilizado como el principal lenguaje de programación para desarrollar las actividades, fragmentos, servicios, y otros componentes de la aplicación Android.
XML: Usado para definir los layouts y recursos de la interfaz de usuario. Todos los diseños de actividades y fragmentos están escritos en XML.

## Entorno de Desarrollo

Android Studio: El entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones Android. Proporciona un editor de código avanzado, herramientas de depuración, emuladores de dispositivos, y soporte para el desarrollo y la construcción de aplicaciones Android.

## **Frameworks y Bibliotecas**

Android SDK: El conjunto de herramientas y bibliotecas proporcionadas por Google para el desarrollo de aplicaciones Android. Incluye las API necesarias para interactuar con el sistema operativo Android y sus componentes. He utilizado la API mínima 24
AndroidX: Un conjunto de bibliotecas que proporcionan componentes modernos y soporte hacia atrás para versiones anteriores de Android. AndroidX es la evolución de las bibliotecas de soporte originales de Android.
ExoPlayer: Una biblioteca de reproducción de medios de código abierto para Android. Utilizada en el proyecto para la reproducción de radio en segundo plano con RadioService.
Glide: Una biblioteca de carga de imágenes eficiente y flexible para Android. Utilizada en el proyecto para cargar y mostrar imágenes en los RecyclerView y otros componentes de la interfaz de usuario.
Google Material Components: Una biblioteca que proporciona componentes de interfaz de usuario diseñados según las directrices de Material Design de Google. Utilizada para implementar BottomNavigationView, NavigationView, y otros componentes de la interfaz de usuario.

## Herramientas de Construcción

Gradle: La herramienta de construcción utilizada por Android Studio para compilar, construir y gestionar las dependencias del proyecto. Gradle permite la configuración de tareas automatizadas y la integración continua.

## Gestión de Base de Datos

SQLite: La base de datos embebida utilizada para el almacenamiento local de datos en la aplicación. SQLite es utilizado en el proyecto para gestionar la información de usuarios, listas, y otras entidades.
SQLiteOpenHelper: Una clase de utilidad proporcionada por el SDK de Android para gestionar la creación y actualización de bases de datos SQLite. DbHelper en el proyecto extiende SQLiteOpenHelper para manejar las operaciones CRUD.

Almacenamiento de Preferencias
SharedPreferences: Una interfaz de almacenamiento de datos clave-valor proporcionada por Android para almacenar datos primitivos de manera persistente. Utilizada en el proyecto para almacenar la sesión del usuario y otra configuración de la aplicación.

## Servicios y Notificaciones

NotificationManager: Utilizado para mostrar notificaciones en el sistema Android. En el proyecto, RadioService utiliza PlayerNotificationManager para manejar las notificaciones persistentes mientras se reproduce la radio.

## Otros Recursos y Utilidades

Resources: El sistema de gestión de recursos de Android para manejar cadenas, imágenes, estilos, y otros recursos. Utilizado extensamente en el proyecto para gestionar la interfaz de usuario y sus elementos visuales.
Layouts Responsivos:Uso de diferentes layouts y estilos para asegurar que la interfaz de usuario se vea bien en diferentes tamaños de pantalla y dispositivos. Los archivos XML para layouts definen la estructura visual de las actividades y fragmentos.

## Editor de Código

Android Studio Code Editor: Un editor de código avanzado con características como autocompletado, análisis de código estático, refactorización, y navegación rápida. Proporciona herramientas específicas para el desarrollo de Android como el editor de layouts visuales y el inspector de vistas.

## Gestión de Recursos

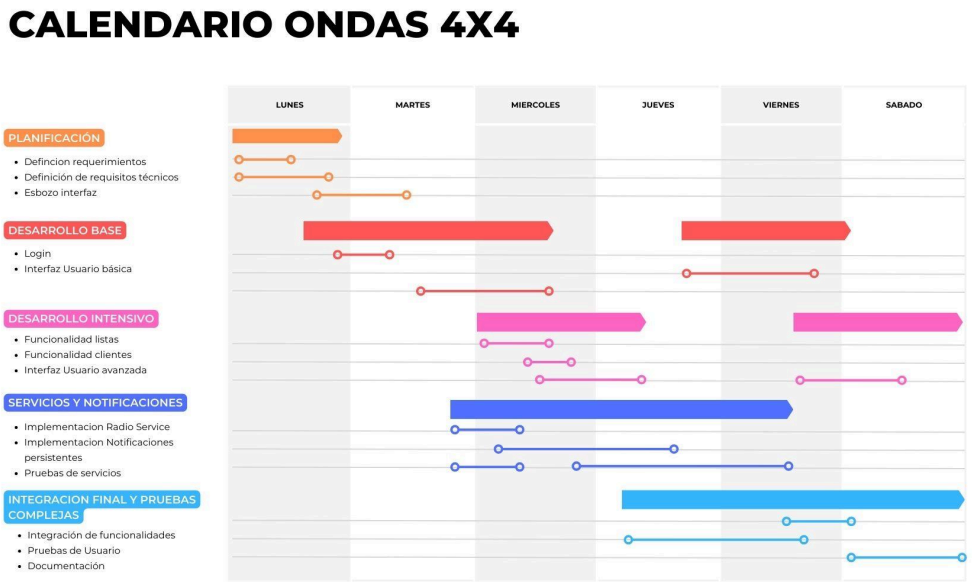
Drawable: Recursos gráficos utilizados en la aplicación para definir las imágenes y gráficos de la interfaz de usuario. Incluye imágenes estáticas y estados gráficos como fondos de botones y otros componentes interactivos.

Values: Directorio utilizado para definir recursos de valores como cadenas, colores, dimensiones, y estilos en archivos XML. Facilita la gestión centralizada de recursos reutilizables en la aplicación.

## Resumen

El proyecto Android Ondas 4x4 utiliza una amplia gama de tecnologías y herramientas para desarrollar una aplicación robusta y funcional. Java y XML son los lenguajes base para la programación y definición de interfaces, respectivamente. Android Studio es el entorno de desarrollo principal, apoyado por Gradle para la construcción y gestión de dependencias. El uso de bibliotecas y frameworks como AndroidX, ExoPlayer, y Glide proporciona funcionalidades avanzadas y optimización de recursos. La gestión de bases de datos se realiza con SQLite y SQLiteOpenHelper, mientras que SharedPreferences se utiliza para el almacenamiento persistente de configuraciones de usuario. En conjunto, estas tecnologías y herramientas permiten el desarrollo eficiente de una aplicación completa, manejando tanto las necesidades de los usuarios como las operaciones administrativas.

# ESTIMACIÓN DE RECURSOS Y PLANIFICACIÓN




Durante el desarrollo de la aplicación de radio, encontré varios desafíos técnicos que extendieron el tiempo de desarrollo en una semana más allá del plan original de cinco semanas. Esta extensión fue necesaria principalmente debido a dos factores: errores de compilación e investigación de mejoras.

Varios errores de compilación surgieron durante el desarrollo, incluyendo incompatibilidades de bibliotecas y problemas de configuración del entorno. Por ejemplo, surgieron conflictos entre versiones de bibliotecas utilizadas en el proyecto, especialmente con ExoPlayer para la reproducción de radio. Estos problemas requirieron ajustes en las dependencias y, en algunos casos, la búsqueda de alternativas compatibles. Además, configurar el entorno de desarrollo adecuadamente fue más complejo de lo esperado, requiriendo ajustes en el archivo gradle y la configuración de la IDE para asegurar una compilación correcta. Resolver estos errores tomó tiempo considerablemente, ya que cada problema requería una investigación exhaustiva y pruebas para garantizar que la solución no causara otros problemas en el sistema.

Además de los errores de compilación, dediqué tiempo adicional a investigar y aplicar mejoras, incluyendo optimización del rendimiento, mejora de la seguridad y

refinamiento de la interfaz de usuario. Implementé técnicas avanzadas para optimizar el rendimiento de la aplicación, especialmente en el manejo de bases de datos y la eficiencia del RecyclerView. También reforcé la seguridad en la gestión de contraseñas y datos sensibles de los usuarios, implementando cifrado más robusto y mejores prácticas de almacenamiento de datos. Realicé iteraciones en el diseño de la interfaz para mejorar la usabilidad y la experiencia del usuario, basándome en pruebas de usabilidad y feedback.

Aunque el proyecto se extendió una semana más de lo previsto, este tiempo adicional fue crucial para asegurar que la aplicación cumpliera con los requisitos funcionales y ofreciera una experiencia de usuario de alta calidad, rendimiento optimizado y seguridad robusta. Esta extensión ha contribuido significativamente a la calidad del producto final.

# ANÁLISIS

## Requisitos Funcionales del Proyecto

Gestión de Usuarios:

Registro de Usuarios: El proyecto debe permitir a los usuarios registrarse proporcionando información como nombre, apellidos, dirección, teléfono, correo electrónico, contraseña y NIF.
Inicio de Sesión: El proyecto debe permitir a los usuarios iniciar sesión utilizando su correo electrónico y contraseña.
Roles de Usuario: El proyecto debe diferenciar entre usuarios normales y administradores. Los administradores tendrán permisos adicionales para gestionar listas y clientes.

Gestión de Listas:

Añadir Lista: El proyecto debe permitir a los usuarios, especialmente a los administradores, añadir nuevas listas proporcionando nombre, categoría, enlace e imagen.
Modificar Lista: El proyecto debe permitir a los usuarios modificar listas existentes.
Eliminar Lista: El proyecto debe permitir a los usuarios eliminar listas existentes.
Visualizar Listas: El proyecto debe permitir a los usuarios visualizar todas las listas disponibles.

Gestión de Clientes:

Añadir Cliente: El proyecto debe permitir a los administradores añadir nuevos clientes con los detalles necesarios.
Modificar Cliente: El proyecto debe permitir a los administradores modificar los detalles de clientes existentes.
Eliminar Cliente: El proyecto debe permitir a los administradores eliminar clientes existentes.
Visualizar Clientes: El proyecto debe permitir a los administradores visualizar la lista completa de clientes registrados.
Interacción con Listas:

Marcar Lista como Favorita: El proyecto debe permitir a los usuarios marcar listas como favoritas.
Visualizar Listas Favoritas: El proyecto debe permitir a los usuarios ver una lista de sus listas favoritas.

Navegación:

Navegación Principal: El proyecto debe permitir a los usuarios navegar entre diferentes secciones de la aplicación, como inicio, listas, cuenta y favoritos.
Menú Lateral: El proyecto debe proporcionar un menú lateral para opciones adicionales como cerrar sesión y modificar datos personales.

Servicios:

Reproducción de Radio: El proyecto debe permitir a los usuarios iniciar y detener la reproducción de radio en vivo.
Requisitos No Funcionales del Proyecto

## Requisitos No Funcionales del Proyecto

Usabilidad:

Interfaz de Usuario Intuitiva: El proyecto debe tener una interfaz de usuario intuitiva y fácil de usar.
Consistencia: Debe haber consistencia en el diseño y la navegación a lo largo de toda la aplicación del proyecto.

Rendimiento:

Tiempo de Respuesta: Las operaciones del proyecto deben ser rápidas y tener tiempos de respuesta mínimos.
Optimización de Recursos: El proyecto debe estar optimizado para un uso eficiente de los recursos del dispositivo.

Seguridad:

Encriptación de Contraseñas: Las contraseñas de los usuarios deben estar encriptadas antes de ser almacenadas en la base de datos del proyecto.
Protección de Datos: Los datos sensibles de los usuarios deben ser protegidos contra accesos no autorizados en el proyecto.

Compatibilidad:

Compatibilidad con Dispositivos: El proyecto debe ser compatible con una amplia gama de dispositivos Android.
Versiones de Android: El proyecto debe ser compatible con versiones recientes de Android, preferiblemente desde Android 5.0 (Lollipop) en adelante.

Mantenimiento:

Código Modular: El código del proyecto debe estar estructurado de manera modular para facilitar el mantenimiento y las actualizaciones futuras.

Documentación: Debe haber documentación adecuada del código y de la estructura de la base de datos del proyecto.

Escalabilidad:

Crecimiento de Datos: El proyecto debe ser capaz de manejar un crecimiento en el volumen de datos sin degradación significativa del rendimiento.
Número de Usuarios: El proyecto debe ser capaz de soportar un número creciente de usuarios activos.

Disponibilidad:

Disponibilidad del Servicio: Los servicios críticos del proyecto, como la reproducción de radio, deben tener alta disponibilidad y ser resistentes a fallos.

Accesibilidad:

Compatibilidad con Accesibilidad: El proyecto debe ser accesible para usuarios con discapacidades, incluyendo soporte para lectores de pantalla y otras tecnologías de asistencia.
Estos requisitos funcionales y no funcionales aseguran que el proyecto cumpla con las expectativas de los usuarios finales y los administradores, proporcionando una experiencia de usuario fluida, segura y eficiente.

## Casos de uso

| Caso de Uso | Descripción | Actores | Precondiciones | Postcondiciones | Flujo Principal |
|---|---|---|---|---|---|
| Registro de Usuarios | Permite a los usuarios registrarse proporcionando información personal | Usuario | El usuario no debe estar registrado previamente | Se crea una nueva cuenta de usuario en el sistema | 1. El usuario selecciona la opción de registro. 2. El sistema solicita los datos personales (nombre, apellidos, dirección, teléfono, correo electrónico, contraseña, NIF). 3. El usuario proporciona los datos solicitados. 4. El sistema valida los datos y crea la cuenta de usuario. 5. El usuario recibe una confirmación de registro exitoso. |
| Inicio de Sesión | Permite a los usuarios iniciar sesión utilizando su correo electrónico | Usuario | El usuario debe estar registrado | El usuario es autenticado y redirigido a su correspondiente pantalla de inicio | 1. El usuario ingresa su correo electrónico y contraseña. 2. El sistema valida las credenciales. 3. El sistema determina el rol del usuario (normal o administrador). 4. El usuario es redirigido a la pantalla de inicio correspondiente. 5. El usuario recibe un mensaje de bienvenida. |

| | | | | | |
|---|---|---|---|---|---|
| | y contraseña | | | | |
| Cerrar Sesión | Permite a los usuarios cerrar sesión en la aplicación | Usuario | El usuario debe estar autenticado | La sesión del usuario es cerrada y se redirige a la pantalla de inicio de sesión | 1. El usuario selecciona la opción de cerrar sesión. 2. El sistema cierra la sesión del usuario. 3. El usuario es redirigido a la pantalla de inicio de sesión. |

| Caso de Uso | Descripción | Actores | Precondiciones | Postcondiciones | Flujo Principal |
|---|---|---|---|---|---|
| Añadir Cliente | Permite a los administradores añadir nuevos clientes | Administrador | El administrador debe estar autenticado | Se crea un nuevo cliente en el sistema | 1. El administrador selecciona la opción para añadir un cliente. 2. El sistema solicita los datos del cliente (nombre, apellidos, dirección, teléfono, correo electrónico, NIF). 3. El administrador proporciona los datos solicitados. 4. El sistema valida los datos y crea el cliente. 5. El administrador recibe una confirmación de que el cliente ha sido añadido. |
| Modificar Cliente | Permite a los administradores modificar los detalles de clientes existentes | Administrador | El administrador debe estar autenticado | Los datos del cliente son actualizados en el sistema | 1. El administrador selecciona la opción para modificar un cliente. 2. El sistema muestra los clientes disponibles. 3. El administrador selecciona el cliente a modificar. 4. El sistema muestra los datos actuales del cliente. 5. El administrador edita los datos necesarios. 6. El sistema valida los cambios y actualiza los datos del cliente. 7. El administrador recibe una confirmación de que los datos del cliente han sido modificados. |
| Eliminar Cliente | Permite a los administradores eliminar clientes existentes | Administrador | El administrador debe estar autenticado | El cliente es eliminado del sistema | 1. El administrador selecciona la opción para eliminar un cliente. 2. El sistema muestra los clientes disponibles. 3. El administrador selecciona el cliente a eliminar. 4. El sistema solicita confirmación para eliminar el cliente. 5. El administrador confirma la eliminación. 6. El sistema elimina el |

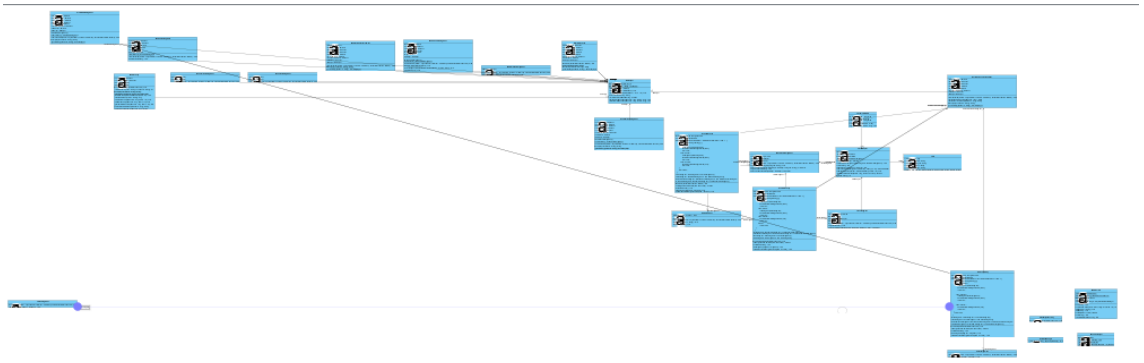
| | | | | | |
|---|---|---|---|---|---|
| | | | | | cliente. 7. El administrador recibe una confirmación de que el cliente ha sido eliminado. |
| Visualizar Clientes | Permite a los administradores visualizar la lista completa de clientes registrados | Administrador | El administrador debe estar autenticado | Los clientes se muestran al administrador | 1. El administrador selecciona la opción para visualizar clientes. 2. El sistema muestra todos los clientes registrados. 3. El administrador puede seleccionar un cliente para ver más detalles. |

| Caso de Uso | Descripción | Actores | Precondiciones | Postcondiciones | Flujo Principal |
|---|---|---|---|---|---|
| Añadir Lista | Permite a los administradores añadir nuevas listas | Administrador | El administrador debe estar autenticado | Se crea una nueva lista en el sistema | 1. El administrador selecciona la opción para añadir una lista. 2. El sistema solicita los datos de la lista (nombre, categoría, enlace, imagen). 3. El administrador proporciona los datos solicitados. 4. El sistema valida los datos y crea la lista. 5. El administrador recibe una confirmación de que la lista ha sido añadida. |
| Modificar Lista | Permite a los administradores modificar listas existentes | Administrador | El administrador debe estar autenticado | La lista es actualizada en el sistema | 1. El administrador selecciona la opción para modificar una lista. 2. El sistema muestra las listas disponibles. 3. El administrador selecciona la lista a modificar. 4. El sistema muestra los datos actuales de la lista. 5. El administrador edita los datos necesarios. 6. El sistema valida los cambios y actualiza la lista. 7. El administrador recibe una confirmación de que la lista ha sido modificada. |
| Eliminar Lista | Permite a los administradores eliminar listas | Administrador | El administrador debe estar autenti | La lista es eliminada del sistema | 1. El administrador selecciona la opción para eliminar una lista. 2. El sistema muestra las listas disponibles. 3. El administrador selecciona la lista a eliminar. 4. El sistema solicita confirmación para |

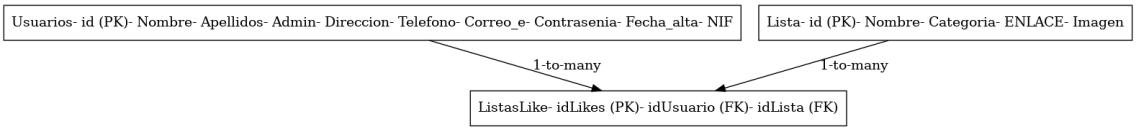
| | | | | | |
|---|---|---|---|---|---|
| | existentes | | cado | | eliminar la lista. 5. El administrador confirma la eliminación. 6. El sistema elimina la lista. 7. El administrador recibe una confirmación de que la lista ha sido eliminada. |
| Visualizar Listas | Permite a los usuarios visualizar todas las listas disponibles | Usuario | El usuario debe estar autenticado | Las listas se muestran al usuario | 1. El usuario selecciona la opción para visualizar listas. 2. El sistema muestra todas las listas disponibles. 3. El usuario puede seleccionar una lista para ver más detalles. |

| Caso de Uso | Descripción | Actores | Precondiciones | Postcondiciones | Flujo Principal |
|---|---|---|---|---|---|
| Marcar Lista como Favorita | Permite a los usuarios marcar listas como favoritas | Usuario | El usuario debe estar autenticado | La lista es marcada como favorita en el sistema | 1. El usuario selecciona una lista. 2. El usuario selecciona la opción para marcar como favorita. 3. El sistema marca la lista como favorita. 4. El usuario recibe una confirmación de que la lista ha sido marcada como favorita. |
| Visualizar Listas Favoritas | Permite a los usuarios ver una lista de sus listas favoritas | Usuario | El usuario debe estar autenticado | Las listas favoritas se muestran al usuario | 1. El usuario selecciona la opción para visualizar listas favoritas. 2. El sistema muestra todas las listas marcadas como favoritas por el usuario. 3. El usuario puede seleccionar una lista favorita para ver más detalles. |

## Diagrama de clases

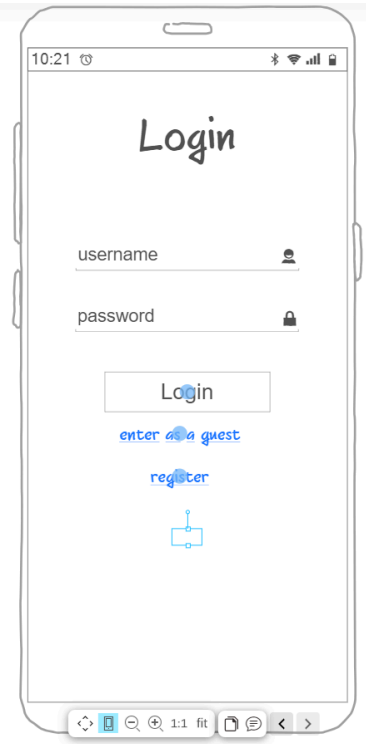


## Diagrama de entidad relación

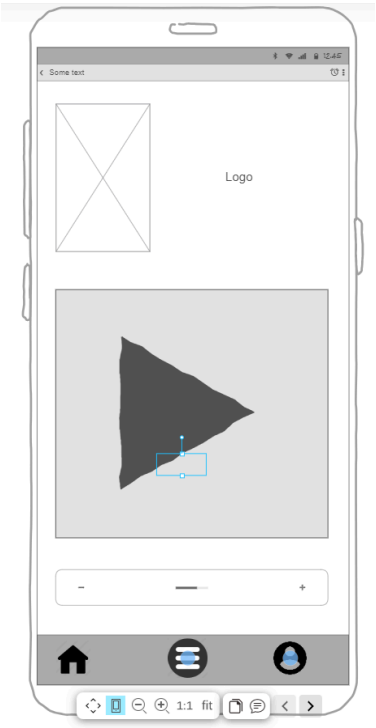


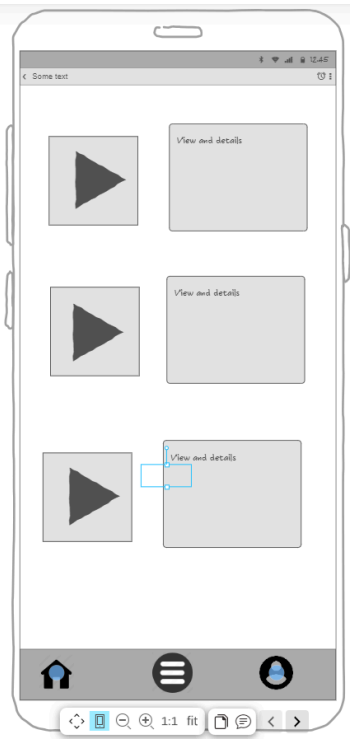
# DISEÑO

## MockUps







## Análisis del diseño del Proyecto Android

Este proyecto Android parece ser una aplicación completa que incluye funcionalidades tanto para usuarios normales como para administradores. La estructura del proyecto abarca una amplia variedad de componentes, desde actividades y fragmentos hasta servicios y proveedores de contenido. A continuación, se presenta un análisis detallado de las partes más importantes del proyecto.

### 1. Archivo AndroidManifest.xml

El archivo AndroidManifest.xml es crucial para la aplicación, ya que define la estructura básica y los componentes principales. En este archivo se declaran todas las actividades, servicios, receptores de difusión y permisos necesarios. Es la guía para el sistema operativo sobre cómo debe interactuar con la aplicación.

### 2. Actividades Principales

#### 2.1. MainActivity

La MainActivity es la actividad principal que maneja el inicio de sesión de los usuarios. Contiene lógica para verificar las credenciales del usuario y determinar si el usuario es un administrador o un usuario normal. Utiliza SharedPreferences para mantener la sesión del usuario. Además, esta actividad redirige a las actividades correspondientes (UsuarioActivity o AdminActivity) según el tipo de usuario. Esta clase es fundamental para la autenticación y la navegación inicial dentro de la aplicación.

#### 2.2. UsuarioActivity

UsuarioActivity es la actividad que se carga después de que un usuario inicia sesión correctamente. Esta actividad gestiona la navegación entre diferentes fragmentos utilizando un BottomNavigationView y un NavigationView lateral para las opciones de cuenta. Los fragmentos manejados incluyen vistas de inicio, listas, y cuenta del usuario. Esta actividad es esencial para la experiencia de usuario, proporcionando una interfaz cohesiva y unificada para navegar por la aplicación.

#### 2.3. AdminActivity

AdminActivity maneja las funcionalidades administrativas, permitiendo a los administradores gestionar usuarios y listas. Utiliza fragmentos para cada una de las operaciones administrativas, tales como añadir, modificar y eliminar clientes y listas. Esta actividad centraliza todas las operaciones administrativas, haciendo que la gestión de la aplicación sea eficiente y organizada.

### 3. Fragmentos Principales

#### 3.1. InicioFragment

InicioFragment muestra el contenido principal que ve el usuario al iniciar sesión. Incluye un ViewFlipper para mostrar diferentes imágenes de forma cíclica. Este fragmento actúa como la pantalla de inicio, proporcionando acceso rápido a las funcionalidades más importantes y un resumen visual de lo que ofrece la aplicación.

#### 3.2. LikesFragment

LikesFragment muestra las listas que el usuario ha marcado como favoritas. Utiliza un RecyclerView para mostrar los elementos de forma eficiente. Este fragmento permite a los usuarios gestionar sus listas favoritas, mejorando la personalización y la experiencia de usuario.

### 3.3. MostrarListaFragment

MostrarListaFragment muestra todas las listas disponibles o listas filtradas por categoría. También utiliza un RecyclerView para la visualización. Este fragmento es crucial para la navegación y gestión de contenido dentro de la aplicación, permitiendo a los usuarios explorar y gestionar listas de forma intuitiva.

### 3.4. ModificarClienteFragment

ModificarClienteFragment permite a los administradores modificar la información de los clientes. Incluye campos de entrada para todos los datos relevantes y actualiza la base de datos cuando se envían los cambios. Este fragmento es esencial para mantener la precisión y actualización de los datos de los clientes en la aplicación.

## 4. Servicios

### 4.1. RadioService

RadioService es un servicio que permite reproducir radio en segundo plano utilizando ExoPlayer y mostrar una notificación persistente mientras se reproduce. Este servicio es fundamental para la funcionalidad de transmisión de audio en la aplicación, proporcionando una experiencia de usuario continua y sin interrupciones.

## 5. Adaptadores y Clases de Utilidad

### 5.1. ListaAdapter

ListaAdapter es un adaptador para el RecyclerView que muestra las listas en diferentes fragmentos. Maneja la lógica de visualización y actualización de las listas. Este adaptador es crucial para la eficiencia y funcionalidad de las vistas de listas, asegurando que los datos se presenten de manera coherente y optimizada.

### 5.2. DbHelper

DbHelper es una clase que extiende SQLiteOpenHelper y maneja la creación y actualización de la base de datos SQLite utilizada por la aplicación. Incluye métodos para operaciones CRUD en diferentes tablas. Esta clase es esencial para la gestión de datos, proporcionando una interfaz estructurada y eficiente para interactuar con la base de datos de la aplicación.

### 5.3. SessionManager

SessionManager es una clase de utilidad que maneja la sesión del usuario. Permite iniciar y cerrar sesiones de manera sencilla utilizando SharedPreferences. Esta clase es fundamental para la seguridad y gestión de usuarios, asegurando que las sesiones de usuario se manejen de manera eficiente y segura.

### 5.4. ResourceHelper

ResourceHelper es una clase de utilidad para acceder a recursos de la aplicación, como imágenes y cadenas de texto, de manera centralizada y eficiente. Esta clase mejora la organización y accesibilidad de los recursos dentro de la aplicación.

## 6. Layouts XML

### 6.1. activity_main.xml

Este archivo define el diseño principal de la MainActivity, incluyendo los campos de entrada para el correo y la contraseña, y los botones para iniciar sesión, registrarse o entrar como invitado. Este layout es crucial para la primera impresión de la aplicación, proporcionando una interfaz limpia y accesible para el inicio de sesión.

### 6.2. fragment_listas.xml

Define el diseño para los fragmentos que muestran las listas, utilizando un RecyclerView para la visualización de los elementos. Este layout asegura que las listas se presenten de manera ordenada y visualmente atractiva.

### 6.3. item_lista.xml

Este archivo define el diseño de cada elemento individual en el RecyclerView, incluyendo una imagen, un nombre y un ícono de favoritos. Este layout es fundamental para la presentación detallada de cada lista, mejorando la experiencia de usuario mediante una visualización clara y atractiva.

### 6.4. fragment_modificar_cliente.xml

Define el diseño para el fragmento que permite modificar los datos de los clientes. Incluye campos de entrada para todos los datos relevantes del cliente. Este layout es esencial para la funcionalidad administrativa, proporcionando una interfaz clara y eficiente para la modificación de datos de los clientes.

### 6.5. fragment_mostrar_cliente.xml

Define el diseño para el fragmento que muestra todos los clientes. Utiliza un ScrollView para permitir la navegación a través de una lista de clientes. Este layout es importante para la funcionalidad de visualización de datos, permitiendo a los administradores ver y gestionar los datos de los clientes de manera eficiente.

Este proyecto Android es una aplicación completa que abarca una variedad de funcionalidades esenciales para un sistema de administración y usuario. Las actividades y fragmentos están bien organizados y cumplen con sus roles específicos, ya sea para gestionar usuarios, listas o manejar la reproducción de radio. Los servicios, como RadioService, y las utilidades, como SessionManager y DbHelper, proporcionan una base robusta para la funcionalidad de la aplicación. En general, la arquitectura y el diseño del proyecto son sólidos y extensibles, lo que permite futuras expansiones y mejoras con facilidad. La aplicación está bien estructurada para manejar tanto las necesidades de los usuarios como las de los administradores, ofreciendo una experiencia de usuario completa y coherente.

# DESPLIEGUE Y PRUEBAS

**Pruebas basadas en los casos de uso**

| Categoría | Caso de Uso | Escenario | Resultado Esperado | Resultado Real |
|---|---|---|---|---|
| Gestión de Usuarios | Registro de Usuarios | Usuario introduce todos los datos requeridos y hace clic en 'Registrar' | El usuario es registrado y redirigido a la página principal | El usuario es registrado y redirigido a la página principal |
| Gestión de Usuarios | Inicio de Sesión | Usuario introduce correo y contraseña correctos y hace clic en 'Iniciar Sesión' | El usuario inicia sesión y es redirigido a la página principal | El usuario inicia sesión y es redirigido a la página principal |
| Gestión de Usuarios | Inicio de Sesión | Usuario introduce correo incorrecto y contraseña y hace clic en 'Iniciar Sesión' | La aplicación permanece pasiva | La aplicación permanece pasiva |
| Gestión de Usuarios | Roles de Usuario | Administrador inicia sesión | El administrador es redirigido a la página de administración | El administrador es redirigido a la página de administración |
| Gestión de Usuarios | Roles de Usuario | Usuario normal inicia sesión | El usuario es redirigido a la página de usuario | El usuario es redirigido a la página de usuario |
| Gestión de Listas | Añadir Lista | Administrador introduce datos de la lista y hace clic en 'Añadir' | La lista es añadida y visible en la vista de listas | La lista es añadida y visible en la vista de listas |

| | | | | |
|---|---|---|---|---|
| Gestión de Listas | Modificar Lista | Administrador modifica los datos de una lista y hace clic en 'Guardar' | La lista es modificada y los cambios son visibles en la vista de listas | La lista es modificada y los cambios son visibles en la vista de listas |
| Gestión de Listas | Eliminar Lista | Administrador selecciona una lista y hace clic en 'Eliminar' | La lista es eliminada de la vista de listas | La lista es eliminada de la vista de listas |
| Gestión de Listas | Visualizar Listas | Usuario navega a la vista de listas | El usuario puede ver todas las listas disponibles | El usuario puede ver todas las listas disponibles |
| Gestión de Clientes | Añadir Cliente | Administrador introduce datos del cliente y hace clic en 'Añadir' | El cliente es añadido y visible en la vista de clientes | El cliente es añadido y visible en la vista de clientes |
| Gestión de Clientes | Modificar Cliente | Administrador modifica los datos de un cliente y hace clic en 'Guardar' | El cliente es modificado y los cambios son visibles en la vista de clientes | El cliente es modificado y los cambios son visibles en la vista de clientes |
| Gestión de Clientes | Eliminar Cliente | Administrador selecciona un cliente y hace clic en 'Eliminar' | El cliente es eliminado de la vista de clientes | El cliente es eliminado de la vista de clientes |
| Gestión de Clientes | Visualizar Clientes | Administrador navega a la vista de clientes | El administrador puede ver todos los clientes registrados | El administrador puede ver todos los clientes registrados |
| Interacción con Listas | Marcar Lista como Favorita | Usuario marca una lista como favorita | La lista es añadida a la vista de listas favoritas del usuario | La lista es añadida a la vista de listas favoritas del usuario |

| | | | | |
|---|---|---|---|---|
| Interacción con Listas | Visualizar Listas Favoritas | Usuario navega a la vista de listas favoritas | El usuario puede ver todas las listas marcadas como favoritas | El usuario puede ver todas las listas marcadas como favoritas |

**Pruebas basadas en UnHappy Paths**

| ID Caso de Prueba | Caso de Uso | Descripción | Precondición | Entrada | Resultado Esperado | Resultado Real |
|---|---|---|---|---|---|---|
| 1 | Registro de Usuario | Intentar registrar un usuario con un correo electrónico ya existente | El usuario ya está registrado | Correo electrónico existente, contraseña, nombre, apellidos, dirección, teléfono, NIF | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 2 | Inicio de Sesión | Intentar iniciar sesión con contraseña incorrecta | El usuario existe en el sistema | Correo electrónico, contraseña incorrecta | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 3 | Añadir Lista | Intentar añadir una lista sin nombre | El usuario está autenticado como administrador | Nombre vacío, categoría, enlace, imagen | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 4 | Modificar Lista | Intentar modificar una lista inexistente | El usuario está autenticado como administrador | ID de lista inexistente, nuevos datos | La aplicación permanece pasiva | La aplicación permanece pasiva |

| | | | | | | |
|---|---|---|---|---|---|---|
| 5 | Eliminar Lista | Intentar eliminar una lista inexistente | El usuario está autenticado como administrador | ID de lista inexistente | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 6 | Añadir Cliente | Intentar añadir un cliente con NIF inválido | El usuario está autenticado como administrador | Nombre, apellidos, dirección, teléfono, correo electrónico, contraseña, NIF inválido | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 7 | Modificar Cliente | Intentar modificar un cliente inexistente | El usuario está autenticado como administrador | ID de cliente inexistente, nuevos datos | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 8 | Eliminar Cliente | Intentar eliminar un cliente inexistente | El usuario está autenticado como administrador | ID de cliente inexistente | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 9 | Marcar Lista como Favorita | Intentar marcar como favorita una lista inexistente | El usuario está autenticado | ID de lista inexistente | La aplicación permanece pasiva | La aplicación permanece pasiva |
| 10 | Reproducción de Radio | Intentar reproducir radio con URL inválida | El usuario está autenticado | URL inválida | La aplicación permanece pasiva | La aplicación permanece pasiva |

# CONCLUSIONES

## Objetivos Alcanzados

Al finalizar este proyecto, se han logrado implementar varios de los objetivos planteados inicialmente, aunque no todos. A continuación, se detallan los objetivos alcanzados y los que no se lograron, junto con una explicación de las causas.

En primer lugar, se ha cumplido completamente con el desarrollo de una aplicación Android utilizando Java. La aplicación Android se ha desarrollado utilizando Java, y todas las funcionalidades principales están implementadas y funcionando como se esperaba. Además, se ha logrado que la aplicación sea compatible y funcione de manera óptima en una amplia gama de dispositivos Android, desde la versión 5.0 (Lollipop) en adelante. Se han realizado pruebas en diferentes dispositivos para asegurar una experiencia de usuario consistente.

También se ha implementado un sistema de manejo de usuarios con autenticación, gestión de perfiles y personalización. Los usuarios pueden registrarse, iniciar sesión, y personalizar sus perfiles. La autenticación se maneja de manera segura utilizando encriptación de contraseñas. Además, se ha diseñado e implementado una base de datos SQLite que es tanto escalable como segura. La base de datos maneja eficientemente las listas de reproducción y la información de los usuarios.

Por último, se ha implementado encriptación para proteger información sensible y garantizar privacidad. La encriptación de contraseñas y la protección de datos sensibles han sido implementadas exitosamente, asegurando que la información de los usuarios esté segura.

## Objetivos No Alcanzados

Sin embargo, no todos los objetivos se han cumplido. Durante el desarrollo, surgieron varios errores de compilación y desafíos técnicos que consumieron más tiempo del anticipado. Además, se dedicó tiempo a investigar mejoras en la aplicación Android, lo que dejó menos tiempo disponible para desarrollar la aplicación web utilizando HTML, CSS, y JavaScript. Como resultado, este objetivo no se ha logrado.

Otro objetivo que no se ha alcanzado completamente es el desarrollo de un plan de marketing con identidad visual y promoción en diversos medios. Aunque se han realizado algunos esfuerzos en esta dirección, el plan de marketing completo aún no se ha desarrollado. La atención se centró principalmente en la funcionalidad técnica y la estabilidad de la aplicación Android.

## Opinión sobre el Trabajo Realizado

El trabajo realizado ha sido satisfactorio en muchos aspectos. La aplicación Android cumple con la mayoría de los objetivos iniciales y ofrece una plataforma útil y funcional para los usuarios interesados en la música hip hop sin copyright. Durante el desarrollo, surgieron varias dificultades.

Uno de los mayores desafíos fueron los errores de compilación. Estos errores fueron particularmente desafiantes y requirieron una considerable cantidad de tiempo y esfuerzo para resolver. Además, se invirtió tiempo en investigar y probar mejoras para la aplicación, lo que si bien fue beneficioso para la calidad del producto final, también contribuyó a que algunos objetivos no se cumplieran a tiempo. Asegurar que la aplicación funcione perfectamente en una amplia gama de dispositivos Android fue otro desafío significativo, requiriendo pruebas exhaustivas y ajustes continuos.

A pesar de estas dificultades, el proyecto ha sido una experiencia enriquecedora. He aprendido mucho sobre el desarrollo de aplicaciones Android y la gestión de bases de datos, así como sobre la importancia de la planificación y la gestión del tiempo en proyectos de software. El producto final es robusto y funcional, y las mejoras futuras y objetivos pendientes se abordarán en la siguiente fase de desarrollo.

# VÍAS FUTURAS

Para completar los objetivos pendientes del proyecto de aplicación de radio y música hip hop, se implementará un enfoque en fases. Primero, se desarrollará una versión web utilizando HTML, CSS, y JavaScript con frameworks modernos como React.js para el frontend y Node.js para el backend, asegurando la integración con la base de datos existente. Segundo, se realizarán pruebas de compatibilidad y optimización en diversos dispositivos y navegadores mediante herramientas como BrowserStack, y se aplicarán prácticas de optimización como el uso de imágenes comprimidas y un CDN para mejorar el rendimiento.

Tercero, se mejorará el sistema de manejo de usuarios añadiendo autenticación de dos factores (2FA), permitiendo la personalización de perfiles y utilizando algoritmos de recomendación para personalizar contenidos. Cuarto, se rediseñará la base de datos para mejorar la escalabilidad y seguridad, implementando particionamiento, replicación, y encriptación robusta de datos tanto en reposo como en tránsito. Quinto, se fortalecerá la encriptación utilizando AES-256 para datos en reposo y HTTPS para datos en tránsito, garantizando la privacidad y protección de la información sensible.

Finalmente, se desarrollará un plan de marketing integral que incluya la creación de una identidad visual coherente, estrategias de marketing digital en redes sociales y colaboraciones con influencers, así como campañas publicitarias en Google Ads y Facebook Ads para aumentar la visibilidad de la aplicación. Este enfoque asegurará una aplicación robusta, segura y accesible, satisfaciendo las necesidades del público objetivo y potenciando su impacto en el mercado de la música hip hop.

# GLOSARIO

Actividad (Activity):

Una pantalla con la cual los usuarios pueden interactuar en una aplicación Android.

Administrador:

Usuario con permisos especiales para gestionar contenidos y usuarios dentro de la aplicación.

AndroidManifest.xml:

Archivo de configuración que define los componentes esenciales de la aplicación y sus permisos.

API (Application Programming Interface):

Conjunto de protocolos y herramientas para construir software y aplicaciones.

AppCompatActivity:

Subclase de Activity que proporciona compatibilidad con versiones anteriores de Android.

Base de Datos SQLite:

Sistema de gestión de bases de datos ligero, utilizado para almacenar datos localmente en dispositivos Android.

BottomNavigationView:

Componente de diseño de Android que permite la navegación entre diferentes secciones de la aplicación mediante una barra de navegación en la parte inferior de la pantalla.

CardView:

Widget de interfaz de usuario que se utiliza para mostrar contenido dentro de una tarjeta con sombras y esquinas redondeadas.

Cipher:

Algoritmo utilizado para cifrar y descifrar datos.

Cliente:

Usuario registrado en la aplicación que puede acceder y gestionar sus listas de música favoritas.

ContentValues:

Clase de Android utilizada para almacenar un conjunto de valores que el método insert() puede usar.

Context:

Información global sobre el entorno de una aplicación Android, utilizado para acceder a recursos y clases específicas de la aplicación.

Fragmento (Fragment):

Componente de la interfaz de usuario reutilizable que puede ser insertado en una actividad.

Intent:

Mensaje que se utiliza para solicitar una acción de otro componente de la aplicación.

LinearLayout:

Layout que organiza sus elementos hijos en una única columna o fila.

MainActivity:

Actividad principal de la aplicación que maneja el inicio de sesión de los usuarios.

MediaItem:

Clase en ExoPlayer que representa un medio (audio, video, etc.) que se puede reproducir.

Notification:

Mensaje que aparece fuera de la interfaz de usuario de la aplicación para proporcionar recordatorios, comunicaciones de otros usuarios u otra información oportuna de la aplicación.

PendingIntent:

Token que se utiliza para permitir a otra aplicación usar los permisos de la aplicación original para ejecutar una operación predefinida.

RecyclerView:

Widget de Android que permite mostrar grandes conjuntos de datos en listas y cuadrículas de forma eficiente.

RelativeLayout:

Layout que posiciona sus elementos hijos en relación unos con otros o con el contenedor principal.

ScrollView:

Layout que permite a los usuarios desplazarse por el contenido que es más grande que la pantalla.

Servicio (Service):

Componente de la aplicación que realiza operaciones en segundo plano sin proporcionar una interfaz de usuario.

SessionManager:

Clase de utilidad que maneja la sesión del usuario utilizando SharedPreferences.

SharedPreferences:

API de Android que permite guardar datos en un formato clave-valor de manera persistente.

TextView:

Widget de interfaz de usuario que se utiliza para mostrar texto.

Usuario:

Persona que usa la aplicación para escuchar música y gestionar listas de reproducción.

ViewFlipper:

Widget de Android que permite la transición entre múltiples vistas o imágenes de manera cíclica.

WebView:

Componente de Android que permite mostrar contenido web dentro de una aplicación.

Widget:

Componente de interfaz de usuario que permite a los usuarios interactuar con la aplicación.

# WEBGRAFÍA

1. Developer Android. Sitio oficial de Android para documentación, tutoriales y guías de desarrollo. Recuperado de https://developer.android.com/

2. Stack Overflow. Comunidad de programadores para resolver dudas y encontrar soluciones a problemas específicos en el desarrollo Android. Recuperado de https://stackoverflow.com/

Cómo resolver errores de compilación en Android Studio. Recuperado de https://stackoverflow.com/questions/33183816/why-am-i-getting-a-gradle-error-in-android-studio

Implementar RecyclerView en Android. Recuperado de https://stackoverflow.com/questions/40584424/simple-android-recyclerview-example

Cómo manejar la navegación entre fragmentos en Android. Recuperado de https://stackoverflow.com/questions/27428303/navigation-drawer-with-fragments

Solución de problemas con ExoPlayer. Recuperado de https://stackoverflow.com/questions/28092379/how-to-play-mp4-video-with-exoplayer

Mejorar el rendimiento de la base de datos SQLite en Android. Recuperado de https://stackoverflow.com/questions/38042521/how-to-improve-performance-of-sqlite-database-in-android

Uso de SharedPreferences en Android. Recuperado de https://stackoverflow.com/questions/3624280/how-to-use-sharedpreferences-in-android-to-store-fetch-and-edit-values

3. ChatGPT by OpenAI. Asistencia con inteligencia artificial para obtener respuestas y guía en tiempo real sobre diversos temas relacionados con el proyecto. Recuperado de https://chat.openai.com/

# ANEXO

1. Mockup del proyecto: https://ninjamock.com/s/RSSBKLx
2. Diagrama de Gantt: Diagrama de Gantt tareas moderno líneas varios col…
3. Diagrama de clases: Class Diagram1.jpg