



**universidad  
de león**



# **Escuela de Ingenierías Industrial, Informática y Aeroespacial**

## **GRADO EN INGENIERÍA INFORMÁTICA**

**Proyecto Sistemas de Información de Gestión y Business  
Intelligence.**

**SPOTIFAI: UN AGENTE DE RECOMENDACIÓN MUSICAL  
CON GRAMOS Y LLMs.**

**Autora: Marta García Fernández**



# 1. Descripción del problema

Las plataformas de música actuales ofrecen sistemas de recomendación muy potentes, pero suelen ser opacos, poco explicables y poco flexibles para el usuario final. Normalmente funcionan a partir de historiales de escucha y modelos cerrados, sin permitir al usuario expresar en lenguaje natural qué quiere escuchar en un momento concreto.

El problema que aborda este proyecto es cómo construir un sistema de recomendación musical que combine datos estructurados (grafos), similitud semántica y lenguaje natural, ofreciendo recomendaciones explicables y controladas por el usuario.

## 2. Estado del arte

### 2.1. Sistemas de recomendación musical

Los sistemas de recomendación musical han evolucionado notablemente en los últimos años y constituyen un área de investigación muy activa dentro de la inteligencia artificial y la minería de datos. En la práctica, la mayoría de plataformas comerciales utilizan una combinación de los siguientes enfoques:

#### Filtrado colaborativo

El filtrado colaborativo se basa en el comportamiento de los usuarios. La idea principal es que usuarios con gustos similares tienden a preferir canciones similares. Para ello, se analizan patrones como reproducciones, “likes” o valoraciones, y se recomiendan canciones que han gustado a usuarios con perfiles parecidos. Este enfoque es ampliamente utilizado en plataformas como Spotify o Last.fm.

Sus principales ventajas son que no necesita conocer las características internas de las canciones y que es eficaz cuando existe una gran cantidad de usuarios y datos históricos.

Sin embargo, presenta limitaciones importantes, como el problema del cold start (usuarios nuevos o canciones nuevas), la escasa explicabilidad (el usuario no sabe por qué se le recomienda algo), o la dependencia total del historial previo.

#### Content-based filtering

El filtrado basado en contenido recomienda canciones similares a aquellas que el usuario ya ha escuchado o valorado, utilizando características propias de las canciones. Estas características pueden incluir géneros musicales, artistas, características de audio (tempo, energía, valencia, etc.) y metadatos textuales.



Este enfoque permite una mayor coherencia en las recomendaciones y funciona incluso con pocos usuarios. No obstante, también tiene desventajas:

- Tiende a generar recomendaciones repetitivas.
- Limita el descubrimiento de música nueva fuera del perfil habitual del usuario.
- Requiere una buena definición y calidad de las características.

## Modelos híbridos

Los modelos híbridos combinan filtrado colaborativo y filtrado basado en contenido, incorporando además técnicas de machine learning y reglas heurísticas. Actualmente, este es el enfoque más habitual en sistemas comerciales, ya que permite compensar las debilidades de cada método por separado.

Aun así, estos sistemas siguen siendo en gran medida opacos, con poca capacidad de control por parte del usuario final y una adaptación limitada al contexto puntual (estado de ánimo, momento del día, intención concreta).

## 2.2. Limitaciones de los sistemas actuales

A pesar de su sofisticación, los sistemas de recomendación musical actuales presentan varias limitaciones comunes:

- Funcionan como una “caja negra”, con escasa explicabilidad.
- Ofrecen poco control directo al usuario sobre el tipo de recomendación.
- Repiten con frecuencia los mismos artistas o estilos.
- Tienen dificultades para adaptarse a peticiones contextuales expresadas en lenguaje natural.
- Estas limitaciones abren la puerta a enfoques más flexibles y explicables.

## 2.3 Tecnologías relacionadas

En los últimos años han surgido nuevas tecnologías que permiten abordar estas limitaciones desde otro ángulo:

- Grafos (Neo4j): permiten modelar relaciones complejas entre canciones, artistas, géneros y usuarios de forma natural, facilitando consultas semánticas y personalización.
- Embeddings vectoriales: representan canciones y consultas en espacios vectoriales, permitiendo medir similitud semántica más allá de reglas estrictas.
- Modelos de lenguaje (LLMs): posibilitan la interacción en lenguaje natural y la generación de explicaciones comprensibles para el usuario.

## 2.4 Posicionamiento del proyecto

Este proyecto se sitúa en la intersección de estos tres enfoques, proponiendo un sistema de recomendación que combina un grafo musical estructurado, búsqueda semántica mediante



embeddings y un agente conversacional basado en modelos de lenguaje con el objetivo de ofrecer recomendaciones musicales más controlables, explicables y adaptadas al contexto del usuario.

## 3. Bases de datos y tecnologías utilizadas

### Base de datos

- Neo4j como base de datos principal.
- Grafo con nodos:
  - Track
  - Artist
  - Genre
  - User
- Relaciones:
  - (:Track)-[:BY\_ARTIST]->(:Artist)
  - (:Track)-[:HAS\_GENRE]->(:Genre)
  - (:User)-[:LIKES {rating}]->(:Track)
- Índice vectorial para búsqueda semántica (track\_embedding\_index).

### Tecnologías

- Python
- Streamlit (interfaz)
- LlamaIndex
- Ollama (modelos locales como Qwen)
- Langdetect (detección de idioma)
- Neo4j Aura / local

## 4. Descripción explícita del problema

El objetivo no es solo recomendar canciones, sino permitir consultas abiertas (“quiero algo tranquilo para estudiar”), evitar resultados incoherentes (idiomas raros, géneros extremos), limitar repetición de artistas y dar explicaciones comprensibles, no inventadas

El reto principal es controlar al LLM para que no alucine, no invente datos, no escriba poesía y no rompa la experiencia de usuario.



## 5. Aportación del proyecto

Este proyecto aporta una aproximación híbrida e innovadora al problema de la recomendación musical mediante la integración de bases de datos estructuradas y modelos de lenguaje de gran tamaño.

La principal aportación es el desarrollo de un agente recomendador híbrido, que combina un grafo musical almacenado en Neo4j con un modelo de lenguaje (LLM). A diferencia de los sistemas tradicionales, el modelo no actúa de forma aislada, sino que está estrictamente controlado por los datos y reglas del sistema.

Además, el sistema introduce un control explícito de las recomendaciones, permitiendo regular aspectos clave como:

- El idioma de las canciones recomendadas.
- El nivel de popularidad.
- El número máximo de canciones por artista.
- El filtrado de estilos o géneros poco deseados.

Otro elemento diferencial es la generación de explicaciones seguras y filtradas, diseñadas para evitar alucinaciones, invenciones de datos o respuestas incoherentes. Las explicaciones están limitadas en extensión, contenido y estilo, priorizando claridad y naturalidad.

Desde el punto de vista académico, el proyecto resulta especialmente interesante por:

- La integración real y funcional de LLMs con bases de datos estructuradas.
- El tratamiento explícito del problema de las alucinaciones.
- El uso práctico de grafos como herramienta central en sistemas de recomendación.

## 6. Solución propuesta

La solución propuesta consiste en una aplicación web de recomendación musical basada en interacción conversacional.

El funcionamiento general del sistema es el siguiente:

El usuario introduce una petición en lenguaje natural, que puede expresar un estado de ánimo, un contexto o un artista de referencia. A partir de esta petición, el sistema realiza una búsqueda semántica mediante embeddings almacenados en Neo4j, obteniendo un conjunto amplio de canciones candidatas.



Posteriormente, estas canciones se filtran y refinan aplicando criterios explícitos de:

- Idioma.
- Género musical.
- Artista (limitando repeticiones).
- Popularidad.

En función del contexto detectado (relajación, estudio, energía, etc.), las canciones se reordenan para priorizar aquellas que mejor se ajustan a la intención del usuario. Finalmente, el sistema genera una explicación breve y controlada y presenta el resultado en un formato conversacional claro y comprensible.

## 7. Aplicación – Guía de usuario

### Pantalla principal (Chat)

La pantalla principal de la aplicación es un chat conversacional.

El usuario escribe libremente qué tipo de música desea escuchar y el sistema responde con:

- Una lista de canciones recomendadas.
- Una breve explicación de por qué esa selección es adecuada.

Ejemplos de uso habituales incluyen:

- “Quiero música tranquila para relajarme”.
- “Me gusta Coldplay y Keane”.
- “Sorpréndeme con algo nuevo”.

### Buscar

La sección “Buscar” ofrece un modo más estructurado de interacción.

Permite al usuario definir explícitamente el número de canciones que desea recibir, manteniendo el resto del comportamiento del sistema.

### Configurar perfil musical

En esta sección, el usuario puede puntuar canciones propuestas por el sistema. Estas valoraciones se almacenan en Neo4j y se utilizan para personalizar progresivamente las recomendaciones futuras, mejorando la adecuación al gusto real del usuario.





## 8. Casos de uso

### Caso pesimista


El usuario introduce una petición ambigua, o relacionada con algo que no se encuentra en la base de datos.

En este caso, el sistema devuelve una lista variada (mezcla de estilos) porque la búsqueda semántica intenta encontrar “algo parecido” sin un contexto claro:

 musica

 **Recomendaciones**

1. Hoopty – The ART MuZeum (hip-hop, popularidad 13)
2. The Rhythm of the Night – Corona (pop, popularidad 70)
3. cycles – .Eehou (jazz, popularidad 48)
4. Radiant Memories – TANUKI (funk, popularidad 37)
5. Paradise – Coldplay (pop, popularidad 81)
6. Tenkodo – lorleaf (jazz, popularidad 47)
7. Song For O – Ward Dhoore (electronic, popularidad 11)

 **Por qué te pueden gustar**

La selección mantiene un rollo parecido a lo que buscas, con predominio de pop, jazz, hip-hop. Además, la mayoría son bastante accesibles (popularidad media ~43), ideales para entrar rápido.

### Caso optimista

El usuario introduce una petición clara, como:  
“Quiero música tranquila para estudiar, tipo indie pop”.

El sistema devuelve una selección coherente, con canciones adecuadas al contexto, idioma apropiado y una explicación clara y comprensible.



😊 Quiero música tranquila para estudiar, tipo indie pop



### 🎵 Recomendaciones

1. Wanted It To Be You – Sigrid (pop, popularidad 30)
2. Som du vill – Theo (pop, popularidad 38)
3. Call Me Up – daydreamers (pop, popularidad 56)
4. Part Of Me – Katy Perry (pop, popularidad 70)
5. Best of My Love – The Emotions (pop, popularidad 73)
6. I'm Yours – Jason Mraz (pop, popularidad 74)
7. Glad You Came – The Wanted (pop, popularidad 76)



### Por qué te pueden gustar

Esta selección se ajusta a los estilos de pop y indie, ofreciendo una música tranquila y relajante para estudiar. Es popularidad media con cerca de 60 puntos.

## Caso inverosímil

El usuario realiza una petición poco realista, por ejemplo:

“quiero música medieval trap japonesa”.

El sistema informa al usuario de que no puede satisfacer exactamente la petición.



quiero música medieval trap japonesa



No he encontrado canciones que encajen con lo que pides 😞.

## 9. Análisis de resultados

En general, el sistema funciona bien cuando el usuario da una señal razonable (género, mood, artista). La arquitectura híbrida ayuda a que las recomendaciones se basen en datos reales del grafo, y no solo en texto generado.





### Aspectos que han funcionado bien:

- interacción en lenguaje natural
- control de repetición por artista
- filtros de idioma y “rareza”
- explicaciones breves (cuando el modelo coopera)

### Limitaciones observadas:

- el rendimiento depende mucho del dataset (si faltan artistas/estilos, la recomendación se degrada)
- con modelos pequeños en local, las explicaciones pueden volverse genéricas o incoherentes
- no hay análisis real de audio (solo metadatos y embeddings textuales)

## 10. Análisis DAFO

### Fortalezas

- Arquitectura clara y modular.
- Alto nivel de explicabilidad.
- Control explícito del comportamiento del LLM.

### Debilidades

- Dataset incompleto.
- Dependencia de reglas heurísticas.
- Capacidad limitada del modelo en local.

### Oportunidades

- Mejora de la personalización.
- Integración con APIs reales de streaming.
- Recomendación contextual avanzada.

### Amenazas

- Competencia de modelos comerciales cerrados.
- Problemas de escalabilidad.
- Sesgos presentes en los datos de entrenamiento.



## 11. Aprendizajes

A nivel personal, el proyecto me ha servido para entender que “usar un LLM” no es suficiente: hay que controlarlo, porque si no el sistema se vuelve impredecible. También he visto el valor práctico de los grafos para relacionar entidades musicales (artista–canción–género) y para enriquecer la recomendación.

Para alguien que revise el repositorio, este proyecto enseña especialmente:

- cómo integrar un LLM con una base de datos real
- cómo reducir alucinaciones con reglas, prompts y filtros
- cómo diseñar una app conversacional usable con Streamlit

## 12. Líneas de futuro

Como posibles extensiones del proyecto se plantean:

- Incorporar análisis de audio real (tempo, energía, etc.).
- Implementar un perfilado dinámico del usuario.
- Generar explicaciones personalizadas según el historial.
- Desplegar la aplicación en la nube.
- Integrar modelos multimodales.

## 13. Ciberseguridad

Desde el punto de vista de la seguridad, la aplicación no almacena datos personales sensibles, utiliza identificadores de usuario simples, no expone credenciales y emplea modelos locales, evitando el envío de datos a servicios externos.

Los principales riesgos identificados son el *prompt injection* y los sesgos del modelo.

Para mitigarlos, se aplican prompts restrictivos, sanitización de las respuestas del modelo y validaciones previas a la generación final.