

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
VIÑA DEL MAR – CHILE



**“DESARROLLO DE UN SISTEMA DE
RECOMENDACIÓN MUSICAL MEDIANTE
INTEGRACIÓN DE MODELOS DE LENGUAJE Y
API’S DE STREAMING”**

MIGUEL IGNACIO ALFARO FLORES

TRABAJO DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO EN
INFORMÁTICA

PROFESOR GUÍA:
PROFESOR CORREFERENTE:

ÓSCAR CARRASCO VERA
DIEGO CÁCERES SOLÍS

Septiembre - 2025

A



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: DESARROLLO DE UN SISTEMA DE RECOMENDACIÓN MUSICAL MEDIANTE INTEGRACIÓN DE
MODELOS DE LENGUAJE Y API'S DE STREAMING

Nombre del candidato(a): Miguel Ignacio Alfaro Flores

Carrera / Grado: Ingeniería Informática.

Campus: Viña del Mar

Departamento: Departamento de Electrotecnia e Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Óscar Carrasco Vera, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente

DEJO CONSTANCIA que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por (marcar una opción):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 11/03/2026

Firma:

Estudiante o Candidato(a):

Fecha: 11/03/2026

Firma:

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

AGRADECIMIENTOS

Agradezco a mi mamá y a mis abuelos por su apoyo desde San Felipe. También a mis amigos que me acompañaron estos años, principalmente a Natán Apablaza, Fernando Bastías, Sofia Jara, Felipe Bravo, Matías Pino y Matía Zambelli.

RESUMEN

El presente trabajo aborda el desarrollo de un sistema de recomendación musical centrado en la interacción conversacional. Este sistema se logra mediante la integración de la interfaz de programación de aplicaciones (API) de *Spotify* y un modelo de lenguaje grande (LLM), específicamente *Google Gemini*. La justificación de este proyecto radica en el potencial de los modelos de lenguaje actuales para la comprensión y el procesamiento del lenguaje humano. Si bien esta tecnología ha sido recientemente empleada en la industria del *streaming* musical, no se ha explotado plenamente su capacidad conversacional como mecanismo para fomentar la exploración y el descubrimiento autónomo de música por parte del usuario, un factor crucial para el desarrollo de algoritmos y esencial para la continuidad del uso del servicio y la sostenibilidad del negocio.

La arquitectura implementada divide las tareas entre un entorno *backend* (utilizando *FastAPI*) y un entorno *frontend* (utilizando *Vite+React*). Ambos entornos fueron desplegados en la nube para el uso restringido de usuarios de prueba, empleando *Render* para el *backend* y *Vercel* para el *frontend*. El resultado es un sistema capaz de interpretar solicitudes en lenguaje natural y combinarlas con el historial de reproducciones de *Spotify* del usuario para generar una respuesta que incluye una sugerencia musical, su enlace a la plataforma y la descripción y justificación de la selección. De este modo, se proporciona un valor en un contexto adicional cuyo objetivo es incentivar al usuario a una mejor comprensión musical, a generar un vínculo más estrecho y a explorar alternativas similares. Los resultados obtenidos en las pruebas piloto indicaron que el 90% de los usuarios experimentó una mayor motivación para explorar nueva música gracias al contexto proporcionado. Se concluye que la humanización del sistema automático de recomendación confiere un valor que lo distingue de soluciones previas y lo posiciona como una solución novedosa y válida para el negocio del *streaming* musical.

Palabras clave: Modelo de lenguaje grande, sistemas de recomendación, *streaming* musical, procesamiento de lenguaje natural.

GLOSARIO

AI: Artificial Intelligence. Inteligencia artificial AMD: Advanced Micro Devices.

API: *Application Programming Interface*. Interfaz de programación de aplicaciones.

IA: Inteligencia Artificial.

DJ: *Disc Jockey*. Operador de discos.

FIPP: *Fédération Internationale de la Presse Périodique*. Federación internacional de la prensa periódica

JSON: JavaScript Object Notation.

LLM: *Large Language Model*. Modelo de lenguaje grande.

RAG: *Retrieval Augmented Generation*. Generación aumentada por refuerzo

RAM: *Random Access Memory*. Memoria de acceso aleatorio

RIAA: Recording Industry Association of América. Asociación de Industria de grabaciones de América.

SDK: *Software Development Kit*. Kit de desarrollo de software. SSD: *Solid State Drive*. Unidad de estado sólido

UML: Unified Modeling Language. Lenguaje de modelado unificado. URL: Uniform Resource Locator. Localizador de recurso uniforme.

ÍNDICE DE CONTENIDOS.

INTRODUCCIÓN.....	1
1. DEFINICIÓN DEL PROBLEMA.....	3
1.1. La solución propuesta.....	5
1.2. Objetivo general	6
1.3. Objetivos específicos.....	6
1.4. Metodología y plan de trabajo	7
2. MARCO CONCEPTUAL.....	8
2.1. Modelos de procesamiento de lenguaje natural LLM	8
2.1.1. Comparación en capacidades.....	9
2.1.2. Conclusión de comparación.	10
2.2. API de streaming de música.....	10
2.2.1. Web API de Spotify	12
2.3. Generación aumentada por recuperación (RAG).....	12
2.4. La importancia del anclaje emocional.	13
2.5. Tecnologías para la arquitectura del software	14
2.6. Diseño de interfaz de recomendador musical (UX/UI)	18
2.7. Sistemas similares existentes.....	19
2.8. Modelado UML y conceptos asociados.....	20
3. PROPUESTA DE SOLUCIÓN	23
3.1. Gestión de tiempo.....	23
3.1.1. Carta Gantt	24
3.1.2. Evaluación de usuarios.	24
3.2. Gestión de recursos	25
3.2.1. Estimación de costos de los recursos:	26
3.3. Análisis de riesgo y mitigaciones.....	26
3.4. Diagrama de actividad recomendador musical	30
3.5. Modelo de negocio Canvas	32
3.6. Requisitos funcionales.	33
3.7. Requisitos no funcionales.....	34

3.8. Casos de uso.	35
3.8.1. Caso de Uso: CU1 -: Autorizar acceso a cuenta.	36
3.8.2. Caso de uso: CU2 - Escribir petición y recibir recomendación.	37
3.8.3. Caso de uso: CU3 - Continuar el chat.	38
3.8.4. Caso de uso: CU4 - Comenzar nuevo chat.	39
3.8.5. Caso de uso: CU5 - Enviar retroalimentación.	39
3.8.6. Caso de uso: CU6 - Borrar historial de chat.	40
3.8.7. Caso de uso: CU7 - Cerrar sesión.	40
3.9. Diagrama de modelo de datos NoSQL	41
3.10. Diagrama de clases.	41
3.11.1 Diagrama de secuencia CU1.	44
3.11.2 Diagrama de secuencia CU2.	45
3.11.3 Diagrama de secuencia CU3.	46
3.11.4 Diagrama de secuencia CU4.	47
3.11.5 Diagrama de secuencia CU5.	48
3.11.6 Diagrama de secuencia CU6.	49
3.11.7 Diagrama de secuencia CU7.	50
3.12 Diagrama de componentes	51
3.13 Diagrama de despliegue.	53
3.14 Concepto inicial de la aplicación web.	54
3.14.1 Vista de autorización.	54
3.14.2 Vista principal.	56
3.14.3 Vista de Feedback	57
3.14.4 Vista Ajustes de usuario.	58
4. VALIDACIÓN DE LA SOLUCIÓN	59
4.1. Metodología de validación	59
4.2. Pruebas funcionales	60
4.2.1. Autorizar acceso a la cuenta	60
4.2.2. Escribir petición y recibir recomendación.	62
4.2.3. Continuar el chat.	66
4.2.4. Comenzar nuevo chat.	67
4.2.5. Enviar retroalimentación.	68
4.2.6. Borrar conversación.	69

4.2.7.	Cerrar sesión	74
4.2.8.	Rendimiento y tiempos de respuesta.	75
4.2.9.	Usabilidad básica	75
4.3.	Evaluación de usuarios en prueba piloto.	76
4.3.1.	Evaluación de estrellas.....	76
4.3.2.	Evaluación de interacción	76
4.3.3.	Motivación a explorar	76
4.3.4.	Comentarios adicionales.....	76
5. CONCLUSIONES.....		78
5.1.	Estado de objetivo principal.	79
5.2.	Estado de objetivos específicos.....	80
6. REFERENCIAS BIBLIOGRÁFICAS.....		84

ÍNDICE DE FIGURAS

Figura 1. Ganancias globales de streaming de música desde 2012 hasta 2024.	1
Figura 3. Porcentaje de ganancias en industria musical según medio en 2020.	2
Figura 4. Índice de inteligencia de análisis artificial.	9
Figura 5. Índice de análisis de código artificial.	9
Figura 6. Precios de entrada y salida.	10
Figura 7. Los servicios de streaming de música más populares en el mundo en 2019.....	11
Figura 8. Suscriptores de Streaming de música por aplicación desde 2016 hasta 2024.	12
Figura 9. Ejemplos de íconos a utilizar dentro de la interfaz web.	18
Figura 10. Imagen prototipo de interfaz de usuario.	19
Figura 11. Diagrama de Gantt de Trabajo de Título.	24
Figura 12. Diagrama de actividad para comenzar nuevo chat.	31
Figura 13. Modelo de negocio Canvas.	32
Figura 14. Diagrama general de casos de uso.....	35
Figura 15. Diagrama de modelo de datos NoSql.....	41
Figura 16. Modelo de Clases.....	43
Figura 17. Diagrama de Secuencia: CU1 - Autorizar acceso a la cuenta. ...	44
Figura 18. Diagrama de Secuencia: CU2 - Escribir petición y recibir recomendación.....	45
Figura 19. Diagrama de Secuencia: CU3 - Continuar el chat.....	46
Figura 20. Diagrama de Secuencia: CU4 - Comenzar nuevo chat	47
Figura 21. Diagrama de Secuencia: CU5 - Enviar retroalimentación.....	48
Figura 22. Diagrama de Secuencia: CU6 - Borrar historial de chat.....	49
Figura 23. Diagrama de Secuencia: CU7 - Cerrar sesión	50
Figura 24. Diagrama de Componentes.....	52
Figura 25. Diagrama de despliegue	54
Figura 26. Vista de autorización.....	55
Figura 27. Vista Principal.	56
Figura 28. Vista de Feedback.....	57
Figura 29. Vista ajustes de usuario.	58
Figura 30. Autorización de acceso a cuenta.....	60
Figura 31. Inicio de sesión en plataforma Spotify.....	61
Figura 32. Interfaz de usuario de la aplicación desplegada.	61
Figura 33. Captura de loading skeleton generado tras escribir petición y esperar recomendación.	62

Figura 34. Captura de respuesta generada tras recibir recomendación de parte del sistema.....	63
Figura 35. Captura de sección del artista "HEALTH".....	63
Figura 36. Captura de petición reiterada en un nuevo chat y su respuesta.	64
Figura 37. Captura de petición refiriendo a sentimientos y su respuesta. ..	64
Figura 38. Captura de petición no relacionada a la función principal y su respuesta.	65
Figura 39. Captura de petición con información inventada y su respuesta.	66
Figura 40. Captura de continuación de conversación.	66
Figura 41. Captura de continuación de conversación con tercer mensaje...	67
Figura 42. Captura de nuevo chat.	67
Figura 43. Captura de petición respecto a conversación en otro chat y su respuesta.	68
Figura 44. Modal de retroalimentación.....	68
Figura 45. Notificación de agradecimiento.....	69
Figura 46. Captura de registro de retroalimentación en base de datos.	69
Figura 47. Captura de historial de chats de usuario.	69
Figura 48. Notificación de confirmación de acción.	70
Figura 49. Captura de historial de chat sin el chat borrado.....	70
Figura 50. Conversación 49c en firestore	70
Figura 51. Conversación 79c en firestore	71
Figura 52. Conversación ab3 en firestore	71
Figura 53. Conversación cb1 en firestore	71
Figura 54. Modal de configuración.....	72
Figura 55. Notificación de confirmación de acción de borrado.	72
Figura 56. Captura de interfaz de usuario con chats eliminados.	73
Figura 57. Captura de conversaciones de usuario en Firestore tras eliminar el historial.....	73
Figura 58. Modal de configuración.....	74
Figura 59. Autorización de acceso a cuenta tras cierre de sesión.....	74

ÍNDICE DE TABLAS

Tabla 1. Compatibilidad de herramientas a utilizar.	16
Tabla 2. Tabla de riesgos.	30
Tabla 3. Caso de uso narrativo: CU1 – Autorizar acceso a cuenta.....	36
Tabla 4. Caso de uso narrativo: CU2 – Escribir petición y recibir recomendación	37
Tabla 5. Caso de uso narrativo: CU3 – Continuar Chat	38
Tabla 6. Caso de uso narrativo: CU4 – Comenzar nuevo Chat.....	39
Tabla 7. Caso de uso narrativo: CU5 – Enviar retroalimentación	39
Tabla 8. Caso de uso narrativo: CU6 – Borrar historial de chat	40
Tabla 9. Caso de uso narrativo: CU7 – Cerrar sesión	40
Tabla 10. Tabla de registro de medición de tiempo de respuesta.....	75

INTRODUCCIÓN.

La industria musical ha experimentado notorios cambios a lo largo del tiempo, conforme la tecnología avanza y ofrece innovadoras soluciones que benefician tanto a oyentes como a artistas y distribuidoras. Los límites que existían en la experiencia de explorar música se han ido eliminando: las pocas emisoras de radio disponibles en una zona geográfica, los conciertos locales y el privilegiado pasatiempo de comprar nuevos discos de vinilo ya no son los únicos métodos para descubrir artistas o canciones. En esta era del streaming, servicios como Spotify, Apple Music o Deezer ofrecen catálogos con decenas de millones de canciones de todos los géneros y de distintas partes del mundo. Además, estos catálogos son entregados al usuario mediante algoritmos que recomiendan canciones que posiblemente sean de su agrado. Sin embargo, las soluciones actuales siguen teniendo límites y oportunidades de mejora.

Global music streaming revenue 2012 to 2024 (\$bn)

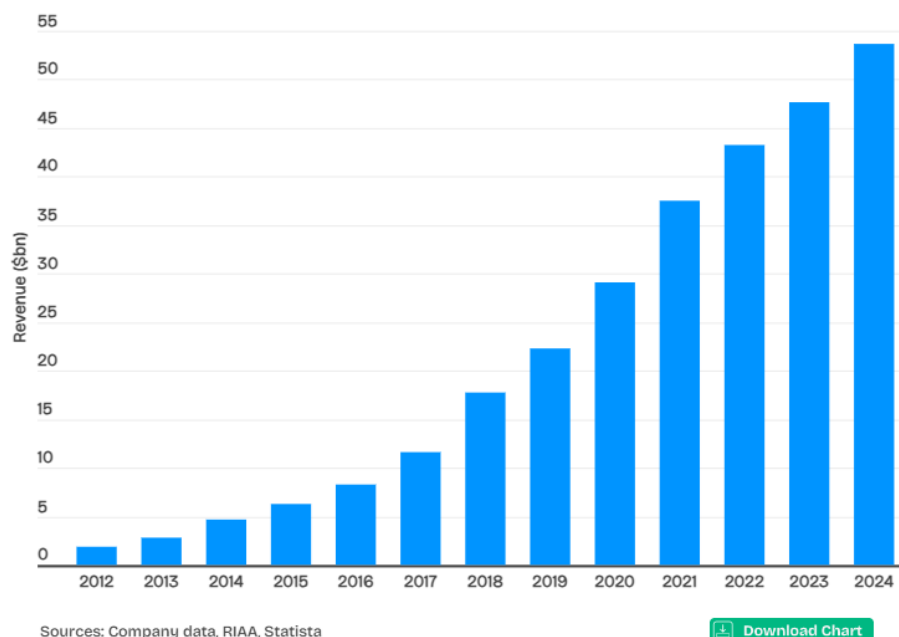
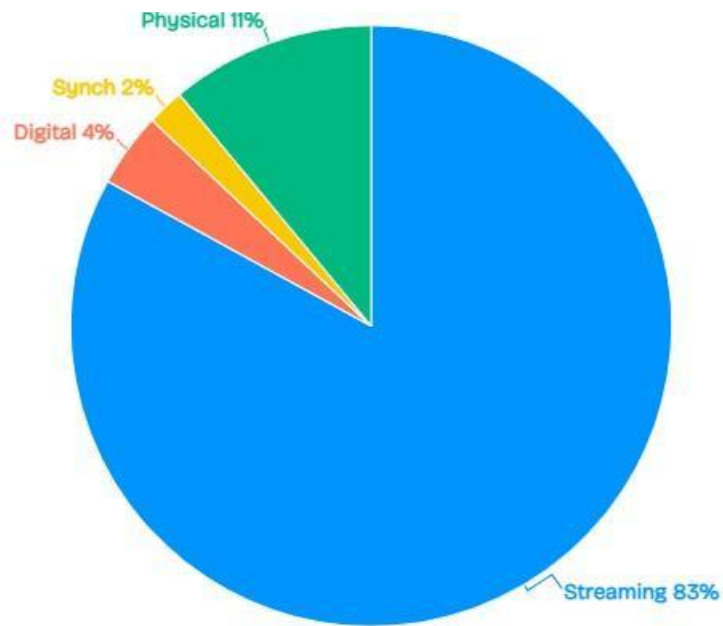


Figura 1. Ganancias globales de streaming de música desde 2012 hasta 2024.

Fuente: RIAA, Statista.



Note: This covers recorded music sales in the United States. Source: RIAA

 [Download Chart](#)

Figura 2. Porcentaje de ganancias en industria musical según medio en 2020.

Fuente: RIAA

1. DEFINICIÓN DEL PROBLEMA.

El catálogo de los servicios de *streaming* y sus algoritmos de recomendación han logrado avanzar efectivamente en la democratización de la música, pero, conforme se crean nuevas comodidades, también surgen nuevos problemas. En *Spotify*, por ejemplo, los usuarios pueden buscar el nombre de artistas, álbumes, *playlists* o canciones y encontrar resultados de todo el mundo. Además, según lo que escuchen, el propio servicio ofrecerá alternativas basadas en el historial de reproducción. Sin embargo, esta dinámica se limita a entregar opciones similares a lo ya experimentado, generando un bucle de retroalimentación que repite lo mismo una y otra vez, sin abrir la puerta a nuevas experiencias o estados de ánimo. A esto se suma que las recomendaciones resultan ineficaces cuando el historial de escucha no está lo suficientemente desarrollado.

Actualmente, se ha integrado una nueva funcionalidad en el servicio de *streaming Spotify*, la cual se alinea con la innovación tecnológica propiciada por el desarrollo de los Modelos de Lenguaje de Gran Escala (LLM), conocidos coloquialmente como "IAs". Esta funcionalidad consiste en la incorporación de un "DJ" inteligente, capaz de interpretar solicitudes ambiguas de los usuarios formuladas en lenguaje natural. Gracias a su comprensión del lenguaje, este DJ virtual puede generar listas de reproducción acordes a lo solicitado, independientemente de la forma en que el usuario lo exprese. Esto permite al usuario solicitar música en función de su estado de ánimo, su contexto actual o la emoción que busca evocar. Por ejemplo, el usuario podría escribir: "Me siento abrumado por la universidad y necesito recuperar la motivación para aprobar el semestre". El DJ ofrecerá una breve introducción que denota el reconocimiento de la necesidad de motivación para luego reproducir una canción apropiada a la solicitud. En una prueba, se reprodujo "Last Son" de David Flemming, una pista de la banda sonora de *Superman* (2025).

La conclusión inicial, basada en la percepción subjetiva individual, es que el DJ comprendió la solicitud y proporcionó efectivamente una canción que evoca la emoción deseada en el usuario.

No obstante, la utilización de los Modelos de Lenguaje de Gran Escala como instrumento para nuevas funcionalidades en el ámbito de la música y el streaming dista de haberse agotado. El DJ con inteligencia artificial de *Spotify* fue concebido con el propósito exclusivo de generar una lista de reproducción

tras interpretar una solicitud de usuario formulada en lenguaje natural. Este sistema presenta limitaciones que le impiden generar respuestas distintas a esta función; si un usuario formula una pregunta o petición ajena a la solicitud musical, el *DJ* emitirá una respuesta negativa, explicando su incapacidad para asistir en lo solicitado.

Uno de los objetivos primordiales en el campo de la inteligencia artificial es desarrollar soluciones que emulen cada vez más el comportamiento humano. Si bien este *DJ* representa un avance en dicha dirección, carece de aspectos fundamentales de la interacción humana, como la conversación. Al analizar la situación humana que el *DJ* busca replicar, se observa que cuando una persona solicita una recomendación musical a otra, esta última no siempre se limita a proporcionar el nombre de la canción, álbum o artista. En su lugar, entabla una conversación en torno a su elección, explicando las razones que motivaron su recomendación.

Entre sus argumentos pueden figurar la melodía, el significado de la letra, la historia del grupo o artista compositor, entre otras formas en que la conversación se transforma en un medio narrativo que no solo busca ofrecer un nombre, sino también incentivar al solicitante a escuchar, explorar y establecer un vínculo personal con la selección.

El componente de conversación narrativa está ausente en la solución actual de *Spotify*. El *DJ* ofrece una selección cuya justificación permanece en una "caja negra"; el usuario puede percibir que su solicitud ha sido satisfecha, pero desconoce el motivo subyacente, lo cual podría desalentar su exploración futura. A gran escala, esta situación podría generar inconvenientes para el negocio. Un usuario que dependa exclusivamente de las recomendaciones del *DJ* podría caer en un bucle de retroalimentación, lo que podría estancar el algoritmo, dado que la exploración es el factor que enriquece los datos utilizados para generar nuevas recomendaciones. Esto, a su vez, repercutiría en la desmotivación y en el escaso apoyo a artistas emergentes. En síntesis, el *DJ* de *Spotify* no fomenta un aspecto crucial: la pasión por la música.

1.1. La solución propuesta.

Para abordar las limitaciones previamente descritas, esta investigación propone el desarrollo de un sistema de recomendación musical basado en la interacción conversacional entre el usuario y un Modelo de Lenguaje Grande. Este enfoque explota la capacidad de procesamiento de lenguaje natural de estos modelos contemporáneos de inteligencia artificial para realizar consultas personalizadas a una *API* de *streaming* musical. Las respuestas se proporcionan en formato de texto, no solo con la información mínima requerida, sino también con un enriquecimiento verbal destinado a fomentar la exploración musical y a incrementar el interés del usuario en la escucha.

Este sistema aspira a constituir una alternativa en la aplicación de LLM en la industria del *streaming* musical, complementando las deficiencias identificadas en la implementación actual del "*DJ AI*". En esencia, busca ofrecer una experiencia centrada en el diálogo, con una mayor libertad de expresión en las solicitudes, que permita al usuario profundizar su conocimiento sobre la música que consume y la que podría descubrir. Este objetivo se logra a costa de restringir la interacción exclusivamente al texto, prescindiendo de la voz sintética del *DJ* y su funcionalidad de lista de reproducción automática.

La justificación de estos parámetros radica, en primer lugar, en no interrumpir la experiencia de audio del usuario, dado que una conversación prolongada mediante audio resulta contraproducente e interfiere con el propósito principal de escuchar música. Por ello, la interacción se limita al texto, permitiendo al usuario utilizar el sistema en paralelo a la reproducción musical. En segundo lugar, si bien una lista de reproducción generada de manera inmediata satisface el valor del "*DJ AI*" al ofrecer una respuesta rápida a una emoción o intención del usuario, esta acción tiende a suplantar el proceso de exploración. El sistema conversacional de texto, por el contrario, otorga al usuario la última palabra para decidir qué buscar y escuchar, un proceso que, aunque más extenso, puede propiciar una conexión más significativa entre el usuario y su selección.

Por consiguiente, el valor diferencial del recomendador conversacional basado en texto reside en el incentivo a la exploración musical exhaustiva, posicionándose como la contraparte del "*DJ AI*", cuyo enfoque principal es el servicio inmediato y preciso.

1.2. Objetivo general

Este trabajo de título tiene como objetivo general “desarrollar un sistema de recomendación musical a través del procesamiento de lenguaje natural, interpretando las solicitudes textuales de los usuarios mediante una interfaz de chat y utilizando datos de historial de uso e información global de música disponibles en una API de streaming de música, para entregar sugerencias personalizadas acorde a lo solicitado con un contexto narrativo que impulse el consumo, exploración y descubrimiento del usuario”.

1.3. Objetivos específicos

Para lograr este objetivo, se han definido los siguientes objetivos específicos a alcanzar.

1. Evaluar y comparar las alternativas disponibles en el mercado respecto a Modelos de Lenguaje y *APIs* de *streaming* musical, para seleccionar la combinación óptima que equilibre capacidades técnicas, documentación robusta y viabilidad económica para el desarrollo del prototipo.
2. Diseñar una interfaz de usuario que permita ingresar y enviar texto de 20 a 300 caracteres.
3. Integrar un modelo de lenguaje natural al sistema de entrada de texto, que permita interpretar las solicitudes del usuario.
4. Solicitar y utilizar el acceso al perfil de usuario autenticado en el servicio de *streaming* de música para analizar patrones de consumo musical, con el objetivo de generar recomendaciones personalizadas que incluyan contenido no explorado previamente por el usuario.
5. Desarrollar la lógica de comportamiento del modelo LLM con el fin de realizar búsquedas en fuentes de información específicas, de acuerdo con la solicitud del usuario, y generar una respuesta que incorpore información adicional, incluyendo análisis lírico, contexto histórico del compositor(es), e información sobre el género o movimiento musical. Presentándose de manera no estructurada, con el objetivo de simular una conversación humana.

6. Optimizar el uso de recursos externos, en particular la *API* del modelo *LLM*, con el fin de garantizar la funcionalidad mínima del sistema manteniendo el consumo de tokens y solicitudes dentro de un presupuesto reducido.
7. Realizar pruebas de funcionalidad y rendimiento para garantizar la calidad del sistema.
8. Evaluar la funcionalidad principal del sistema mediante pruebas controladas con usuarios de distintos perfiles, con el fin de recopilar retroalimentación sobre la satisfacción, utilidad y potencial aumento de interés tras uso de la herramienta.

1.4. Metodología y plan de trabajo

La metodología para el desarrollo de este proyecto utilizará los principios ágiles de SCRUM adaptados a un contexto de desarrollo individual, esto permite una estructura necesaria para la investigación académica que a su vez es flexible y otorga la entrega iterativa de valor. Se dividirá en tres fases principales: Investigación, desarrollo y evaluación. La primera iteración tendrá como objetivo completar el objetivo específico 1, es decir, una fase de investigación donde se comparan las herramientas de desarrollo idóneas para completar este trabajo. Las iteraciones posteriores cumplirán los objetivos específicos de desarrollo técnico, la agilidad de la metodología permite cambios y mejoras durante el desarrollo por lo que es ideal para un trabajo donde se utilizan herramientas nuevas con la finalidad de desarrollar una funcionalidad no aplicada al 100% en el contexto dado. Las últimas iteraciones corresponden a la evaluación, el proyecto está estrechamente relacionado a la experiencia del usuario, por lo que tras pasar las pruebas de funcionalidad y rendimiento se deberá poner a prueba con usuarios que utilizan diversas formas de expresarse y tienen diferentes gustos musicales, el objetivo será satisfecho si la evaluación ponderada de los usuarios es mayormente positiva.

2. MARCO CONCEPTUAL

2.1. Modelos de procesamiento de lenguaje natural LLM

El procesamiento del lenguaje natural es un subcampo de la informática y la Inteligencia Artificial, utiliza el aprendizaje de máquina para la comprensión del lenguaje cotidiano por parte del software. Los avances de este campo han dado como fruto desde los bots de asistencia hasta los actuales modelos de lenguaje de gran tamaño, estos últimos, conocidos popularmente solo como "IA" son modelos de redes neuronales caracterizados por el uso de cantidades inmensas de datos para su entrenamiento, lo que les permite realizar tareas más complejas en el ámbito de la comprensión de lenguaje natural. Los datos son utilizados para detectar patrones en las conversaciones humanas y generan un lenguaje siguiendo estos, poseen una superior flexibilidad en comparación a los motores de búsqueda tradicionales donde se usan algoritmos que hacen coincidir palabras clave, esto debido a que pueden comprender el contexto y razonamientos que van más allá de detectar las palabras más importantes de una oración. La enorme cantidad de datos requeridos para construir un *LLM* hacen que la labor de crear un modelo para propósitos de trabajo de título sea una meta casi inalcanzable, es un esfuerzo que lleva más tiempo y recursos de lo que se dispone, afortunadamente, las empresas dueñas de *LLM* suelen permitir el acceso controlado con sus APIs y herramientas de desarrollo, generalmente a cambio de un precio monetario, por lo que para este proyecto se optará por utilizar la opción más accesible y útil que las empresas ofrecen, para ello es necesario una fase de análisis e investigación.

2.1.1. Comparación en capacidades.

La página artificialanalysis.ai ofrece una serie de gráficos de comparación entre múltiples modelos LLM. Entre ellos la "inteligencia" de cada modelo a momento de análisis y programación.

Artificial Analysis Intelligence Index

Artificial Analysis Intelligence Index v3.0 incorporates 10 evaluations: MMLU-Pro, GPQA Diamond, Humanity's Last Exam, LiveCodeBench, SciCode, AIME 2025, IFBench, AA-LCR, Terminal-Bench Hard, τ^2 -Bench Telecom

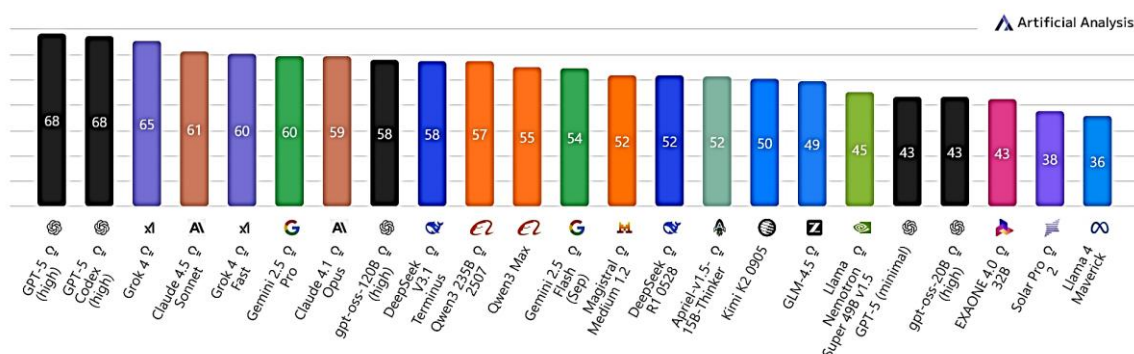


Figura 3. Índice de inteligencia de análisis artificial.

Fuente: Artificial Analysis.

Los modelos más destacados en inteligencia analítica son GPT-5, Grok 4, Claude 4.5 sonnet, Gemini 2.5 Pro y Claude 4.1 Opus.

A nivel de código, análisis necesario para conocer la capacidad de interpretación de lenguaje natural a peticiones de datos los modelos destacados son: *Grok 4*, *GPT-5*, *Codex*, *Deepseek* y *Gemini 2.5*

Artificial Analysis Coding Index

Represents the average of coding benchmarks in the Artificial Analysis Intelligence Index (LiveCodeBench, SciCode, Terminal-Bench Hard)

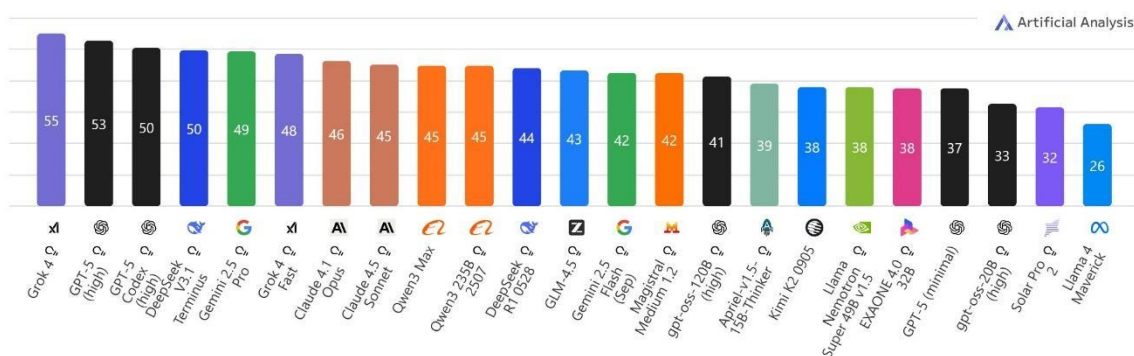


Figura 4. Índice de análisis de código artificial.

Fuente: Artificial Analysis

También ofrece una comparativa de precios.

Pricing: Input and Output Prices

Price: USD per 1M Tokens

■ Input price ■ Output price

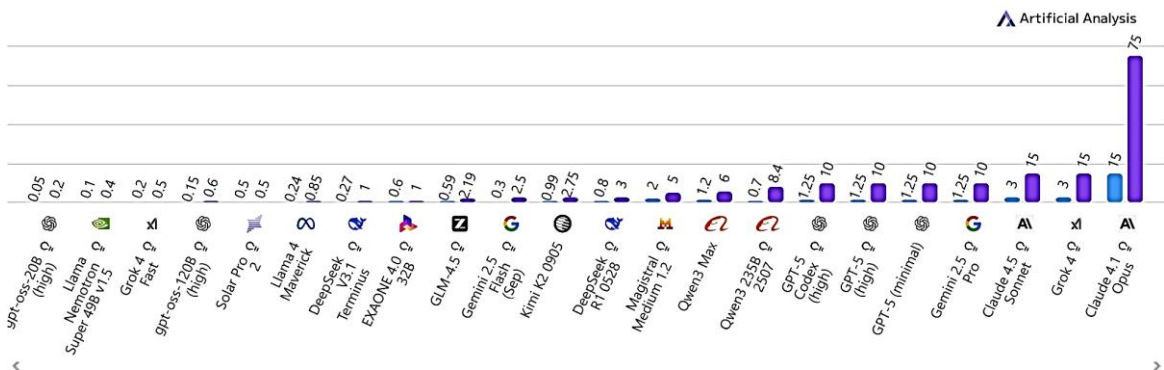


Figura 5. Precios de entrada y salida.

Fuente: Artificial Analysis

Los modelos más baratos resultan ser los Open models de Gpt, lo siguen Llama Nemotron, Grok 4 fast, Solar Pro 2, Llama 4 Maverick y Deepseek V3.1.¹

2.1.2. Conclusión de comparación.

A partir de este análisis, el modelo de procesamiento de lenguaje natural que más ha destacado entre sus competidores, en precio e inteligencia, ha sido Grok 4 fast. Sin embargo, el nivel gratuito que ofrece google para el uso de Gemini es una mejor alternativa para el desarrollo y testeo temprano², dejando a Grok 4 como una versión para etapas posteriores de evaluación.

2.2. API de streaming de música.

Las interfaces de programación de aplicaciones permiten la comunicación adecuada entre aplicaciones para el intercambio de datos y funcionalidades de una forma que la seguridad no se encuentre comprometida. Dentro del proyecto, la solución de un recomendador de música requiere de la

¹ Para revisar las estadísticas de los modelos actuales, visitar la página web: <https://artificialanalysis.ai/models>

² Para más información respecto a los precios de Gemini, visitar la documentación de su API: <https://ai.google.dev/>

información de artistas, álbumes y canciones.

Información que se encuentra principalmente usada en servicios de streaming de música.

Los datos manejados por estos servicios son además utilizados para la creación de algoritmos, los cuales funcionan principalmente comparando las coincidencias entre usuarios respecto a lo que escuchan, es por eso que existe una ventaja demarcada por la cantidad de usuarios que maneja la aplicación. A nivel global, esta ventaja está tomada por *Spotify*, teniendo 100 millones de suscriptores en 2019 y en la actualidad 150.

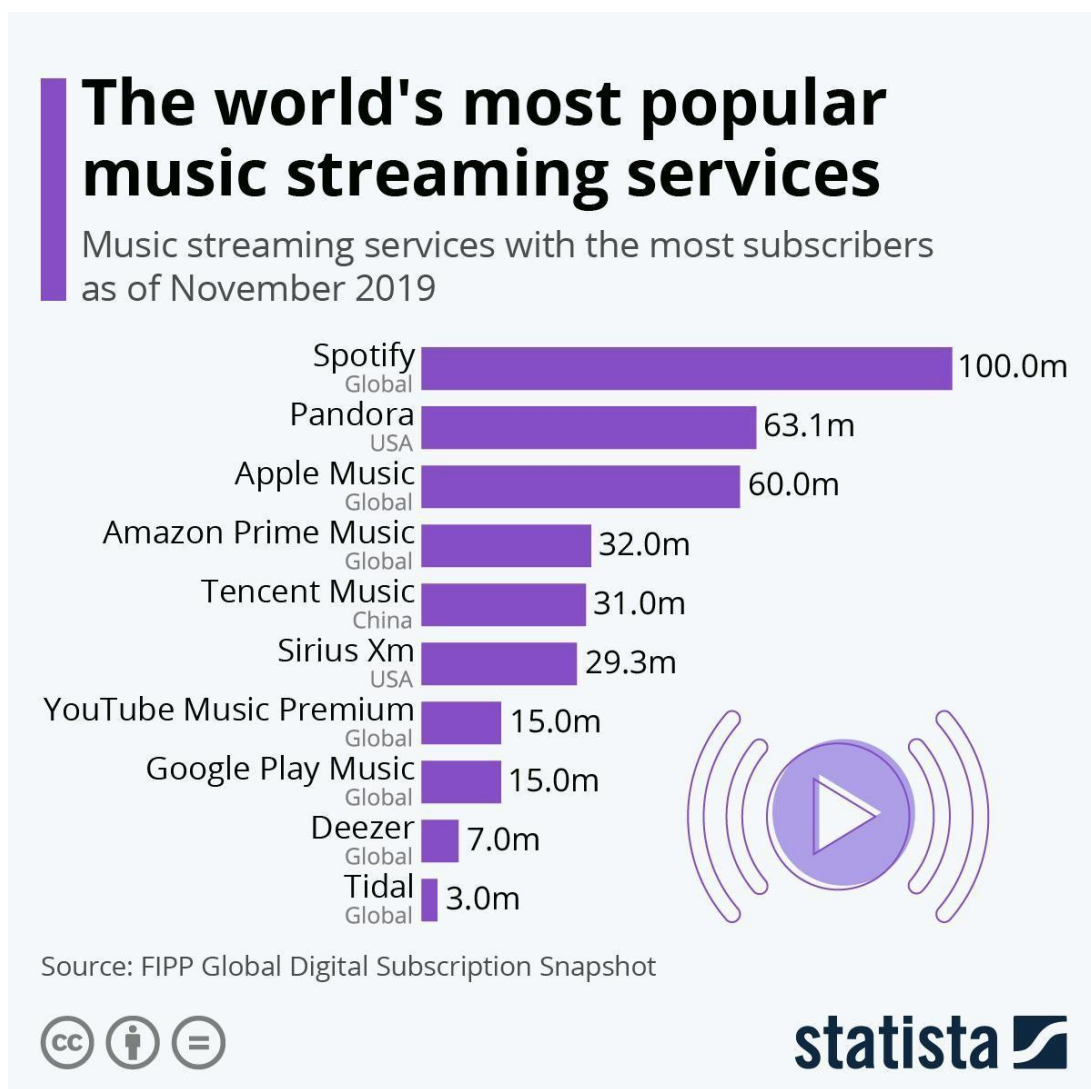
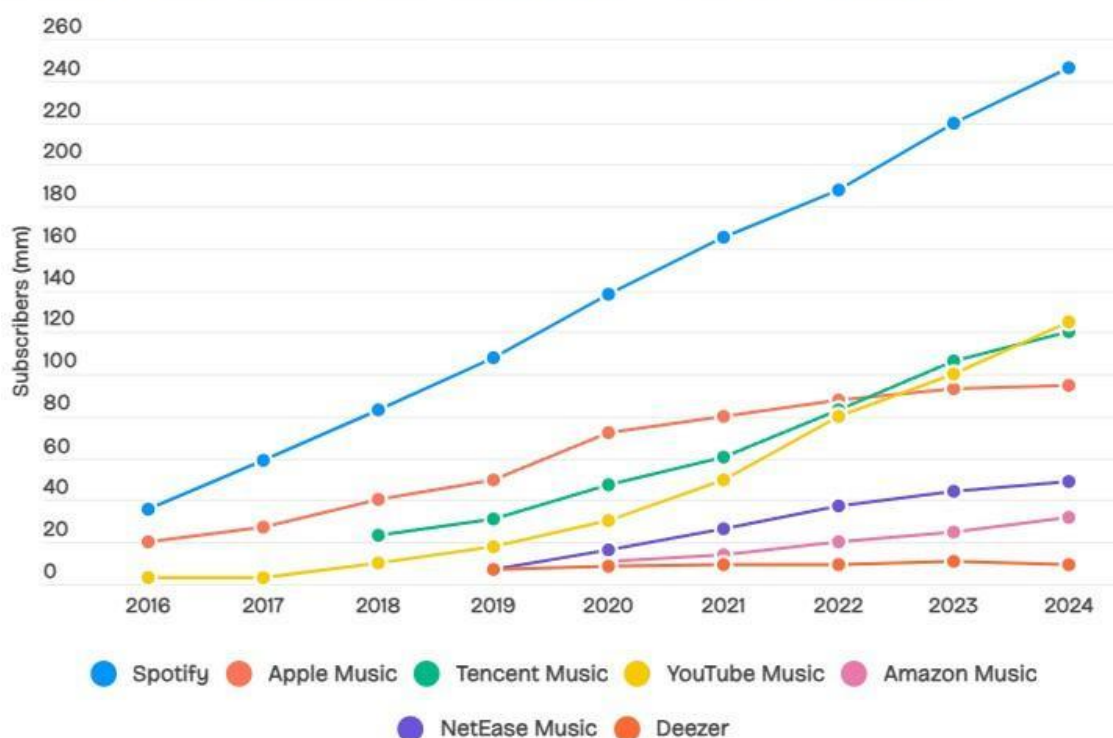


Figura 6. Los servicios de streaming de música más populares en el mundo en 2019.

Fuente: FIPP Global Digital Subscription Snapshot.

Music streaming subscribers by app 2016 to 2024 (mm)



Sources: Company data, Edison Trends

[Download Chart](#)

Figura 7. Suscriptores de Streaming de música por aplicación desde 2016 hasta 2024.

Fuente: Edison Trends.³

2.2.1. Web API de Spotify

La *API* web de *Spotify* permite interactuar con múltiples características disponibles en la plataforma, entre estas se encuentra la información de artistas, álbumes y pistas, el análisis de listas de reproducción de usuarios y el acceso a algoritmos de recomendación personalizados de Spotify. El acceso a estas herramientas es necesario para cumplir los objetivos específicos de este proyecto.

2.3. Generación aumentada por recuperación (RAG)

También conocida como Retrieval-augmented generation, es un patrón de arquitectura de software utilizado en el campo de la Inteligencia Artificial Generativa, donde al modelo *LLM* se le otorga información adicional externa a su base de conocimientos adquiridos por entrenamiento para la generación de una respuesta. Este método permite generar respuestas con datos actualizados y específicos con una precisión más confiable, disminuyendo las

chances de otorgar respuestas erróneas o inventadas por parte del modelo.

Actualmente, esta arquitectura es utilizada ampliamente para el uso interno de organizaciones que actualizan los datos de entrenamiento con información de la empresa, también es utilizado por modelos *LLM* avanzados, los cuales tienen acceso a la recuperación de información desde internet, siendo así cómo logran mantener sus conocimientos actualizados hasta la fecha.

La integración del marco arquitectónico RAG en el proyecto se justifica como imperativa para la inclusión de conocimientos que no pueden ser adquiridos durante la fase de entrenamiento, tales como la información específica del usuario, concretamente, los artistas con mayor número de reproducciones y las canciones reproducidas recientemente. Dado que este conocimiento es inherentemente variable por usuario y por sesión, se requiere que dichos datos sean recuperados *a posteriori* del entrenamiento.

2.4. La importancia del anclaje emocional.

Biológicamente, la música ejerce un impacto significativo en la liberación de neurotransmisores. Su influencia incide positivamente en la experiencia emocional, facilitando la evocación de recuerdos, sentimientos y lazos sociales. Los circuitos neuronales de placer y recompensa estimulados por la música propician la liberación de dopamina, lo cual se traduce en un placer inmediato en el individuo, incrementando su motivación e incluso su autoestima. Esto convierte a la música no solo en un medio de expresión, sino también en una herramienta biológica para la modulación del estado de ánimo. Diversos estudios han señalado que los circuitos de recompensa se activan también durante emociones consideradas negativas, como la tristeza. Los factores externos que refuerzan un procesamiento seguro de la emoción generan placer, aun siendo esto aparentemente contradictorio, lo que denota la relevancia de la expresión y audición emotiva musical de todo el espectro, y no únicamente de la felicidad, para un procesamiento saludable de la psique humana.

La complejidad inherente al ser humano y sus múltiples dimensiones han constituido la base para el análisis de conceptos que trascienden el mero sonido de una melodía y su repercusión en la producción de dopamina. Se denomina apreciación musical a la audición activa de esta con el fin de comprender los elementos subyacentes al sonido, tales como el contexto

histórico y cultural, así como el significado e interpretación de la lírica que frecuentemente acompaña a las composiciones. La experiencia de pistas musicales cuyo contexto es previamente conocido genera un impacto diferente al de una pista desconocida, dado que todo aquello que no es el sonido en sí mismo posee una influencia cognitiva en el oyente.

El proyecto tiene como objetivo fundamental optimizar la experiencia de la audición musical mediante la facilitación del acceso a información contextual en un formato conversacional. Se anticipa que esta aproximación estimulará al usuario a incrementar su consumo del contenido musical de su preferencia. La justificación para emplear una inteligencia artificial como sistema de recomendación conversacional, en contraste con soluciones basadas exclusivamente en algoritmos matemáticos, radica en el vínculo emocional que el individuo establece tanto con el medio auditivo como con el sistema proveedor de la recomendación. El ser humano manifiesta una propensión a la humanización y a la empatía hacia conceptos inanimados que poseen atributos similares, como es el caso de personajes animados o chats de IA. La interacción conversacional activa con máquinas que emulan el lenguaje natural provoca una relación empática, fenómeno conocido como efecto ELIZA, donde los usuarios, aun siendo conscientes de la naturaleza inerte del sistema informático, desarrollan un vínculo similar al que establecerán al dialogar con otro ser humano.

Esta empatía generada en el usuario se complementa con la relación derivada de la comprensión del contexto inherente a la música. El conocimiento del significado de las letras, la identidad de los compositores, o el movimiento artístico al que pertenece una obra, constituyen elementos informativos que otorgan una conexión más profunda y significativa con el medio. El impacto de este fenómeno se ha evidenciado en el incremento de la reproducción de pistas en las plataformas de *streaming*. Un caso notorio fue el de la película biográfica *Bohemian Rhapsody*, que narra la historia del grupo musical "Queen"; su éxito no solo se reflejó en la taquilla, sino también en un aumento del 156% en las reproducciones dentro de la plataforma Spotify, un hito que se ha replicado con otras producciones biográficas y que demuestra que el interés por la música se intensifica al conocer su contexto.³

2.5. Tecnologías para la arquitectura del software

³ véase la noticia en: <https://www.biobiochile.cl/noticias/artes-y-cultura/musica/2018/11/07/reproducciones-de-queen-a-nivel-mundial-se-disparan-tras-estreno-de-bohemian-rhapsody-en-los-cines.shtml>

Las herramientas seleccionadas para la arquitectura del proyecto fueron elegidas priorizando la interoperabilidad, la baja latencia y el procesamiento asíncrono. Esto se debe a que el valor fundamental reside en la calidad de la respuesta generada y en la experiencia conversacional, lo cual exige la capacidad de soportar tiempos de espera variables tras la consulta a las APIs sin que se vea obstaculizada la interfaz de usuario.

a) Backend: Python FastAPI

FastAPI es un *framework* web basado en *Python* diseñado para la construcción de APIs. Sus características principales incluyen su velocidad, alto rendimiento, diseño intuitivo, robustez y soporte para programación asíncrona. Estas particularidades resultan idóneas para la funcionalidad primordial del *backend*, que es la conexión entre diversas APIs sin comprometer la fluidez de la interfaz de usuario⁴.

b) Frontend: Vite + React

Vite constituye una herramienta de compilación que facilita un desarrollo rápido y ágil. En conjunto con la biblioteca Javascript React, permite el desarrollo de una interfaz de usuario moderna, veloz y altamente responsiva gracias a la utilización de componentes.⁵

c) Base de datos: Firestore

Google Firebase proporciona una variedad de herramientas para su utilización en la nube. Entre ellas, destaca *Firestore*, una base de datos NoSQL óptima para la gestión de datos no estructurados, como el historial de chat, además de ofrecer acceso a otras utilidades como las herramientas de autenticación.⁶

d) Orquestación de LLM: Langchain

La librería *Langchain* ofrece herramientas para la gestión de modelos grandes de lenguaje, lo cual es esencial para el manejo de prompts y el historial del usuario.⁷

⁴ Para más información visitar: <https://fastapi.tiangolo.com/>

⁵ Para más información visitar: <https://es.vite.dev/guide/> y <https://es.react.dev/learn>

⁶ Para más información visitar: <https://firebase.google.com/docs>

⁷ Para más información visitar: <https://www.langchain.com/>

e) Herramientas de desarrollo: Google generative SDK

Kit de desarrollo de *software* oficial de google dedicado a la inteligencia artificial generativa, ofrece interfaces y bibliotecas de *python* necesarias para la interacción con los modelos de *Gemini*.⁸

f) Despliegue web frontend: Vercel

Vercel es una plataforma en la nube que permite crear, desplegar y escalar sitios web y aplicaciones de alto rendimiento, su infraestructura optimizada es ideal para el despliegue de prueba del proyecto⁹

g) Despliegue web backend: Render

Render es una plataforma en la nube que ofrece servicios de alojamiento para desplegar y escalar aplicaciones *web*, bases de datos, etc. Simplifica el despliegue desde repositorios Git y la gestión de infraestructura.

Permite el despliegue de contenedores *Docker* y servicios *web* persistentes, garantizando un servicio sin interrupciones.¹⁰

h) Tabla de compatibilidad de herramientas a utilizar

En orden de cerciorarse de que la arquitectura es viable, se debe revisar la compatibilidad de los elementos previamente mencionados, la siguiente tabla demuestra aquello:

Tabla 1. Compatibilidad de herramientas a utilizar.

Herramienta	Versión mínima recomendada	Funcionalidad	Restricciones
Python	3.10.0	Entorno de backend	<i>Langchain</i> no otorga soporte a versiones inferiores a 3.8.
FastAPI	0.100.0	Framework de backend	La versión mínima soporta el uso de <i>Pydantic</i> v2 para la validación de datos.

⁸ Para más información visitar: <https://docs.cloud.google.com/vertex-ai/generative-ai/docs/sdks/overview?hl=es>

⁹ Para más información visitar: <https://vercel.com/docs>

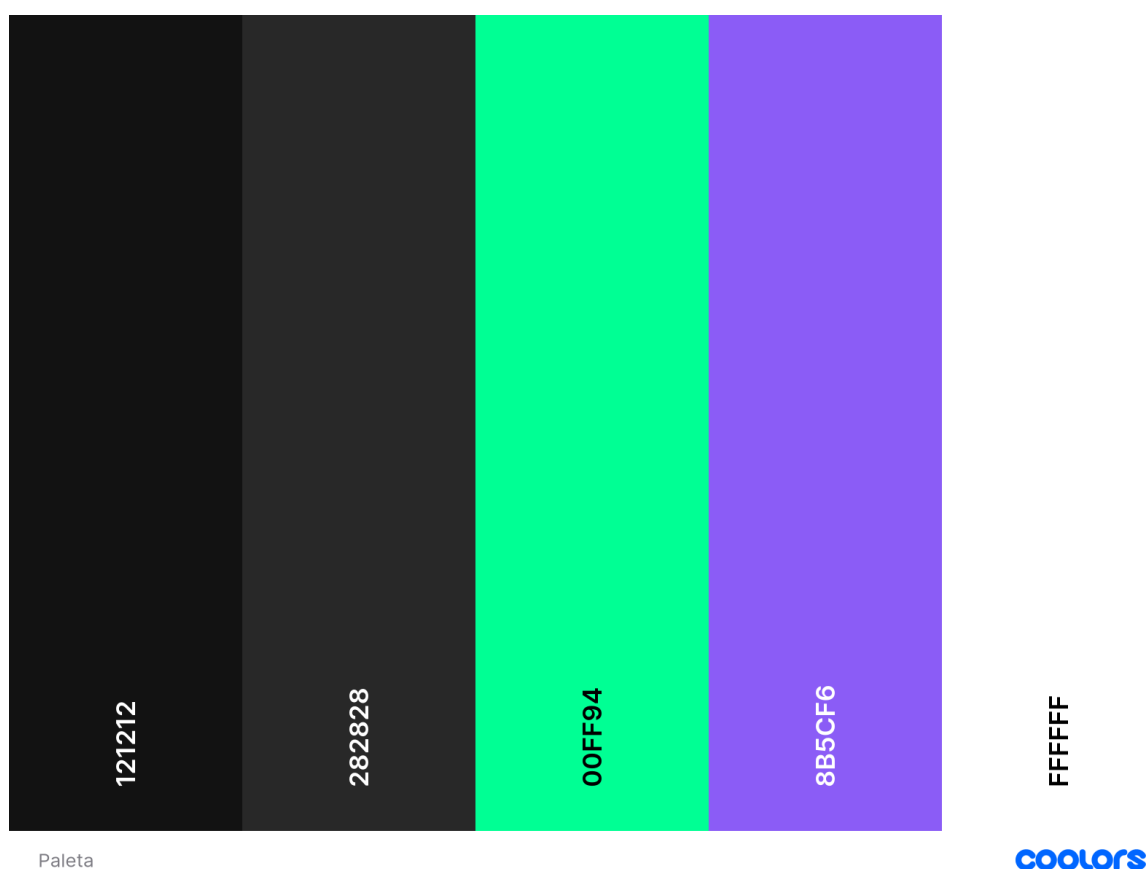
¹⁰ Para más información visitar: <https://render.com/docs>

React	18.2.0	Librería para Frontend	Versiones anteriores no cuentan con " <i>Concurrent mode</i> ", una funcionalidad que permite renderizado sin congelamiento de interfaz.
Vite	5.0	Herramienta de compilación para <i>Frontend</i>	Versión de <i>vite</i> optimizada para tiempos de carga rápidos.
Langchain	0.1.0	Orquestador de IA	Versión estable.
Google Generative SDK	0.3.0	Herramientas de desarrollo para <i>gemini</i>	Soporta modelos con historial de chat nativo y llamadas a funciones.
Node.js	18.0	Entorno frontend	Versión mínima para utilizar <i>vite</i> 5 y funciones modernas de react.
Firestore client SDK	2.14.0	Persistencia de datos	Se requiere una versión del kit de desarrollo reciente para asegurar un soporte estable de la nube de google.

Fuente: Elaboración propia.

2.6. Diseño de interfaz de recomendador musical (UX/UI)

El diseño de la interfaz se basará en las interfaces de *chat* de otras LLM reconocidas, las cuales, para reducir la fatiga visual, suelen adoptar una paleta de colores oscuros, con el gris oscuro como tono predominante. El blanco se utilizará para generar contraste en el texto, y se emplearán los colores "*Spring green*" y "*Medium State Blue*" para establecer un diseño análogo al utilizado por Spotify, indicando así una relación con los datos del usuario extraídos de dicha plataforma.



Para la iconografía, se selecciona el uso del paquete de íconos Lucide React por su diseño liviano y limpio acorde a un diseño de interfaz moderno.



Figura 8. Ejemplos de íconos a utilizar dentro de la interfaz web.

Fuente: Lucide

Para la selección tipográfica, se optó por Roboto debido a su asociación con el texto de inteligencia artificial.

A continuación, se presenta una imagen prototipo que sirve de referencia para

la interfaz de usuario. Esta es exclusivamente referencial y está sujeta a modificaciones.

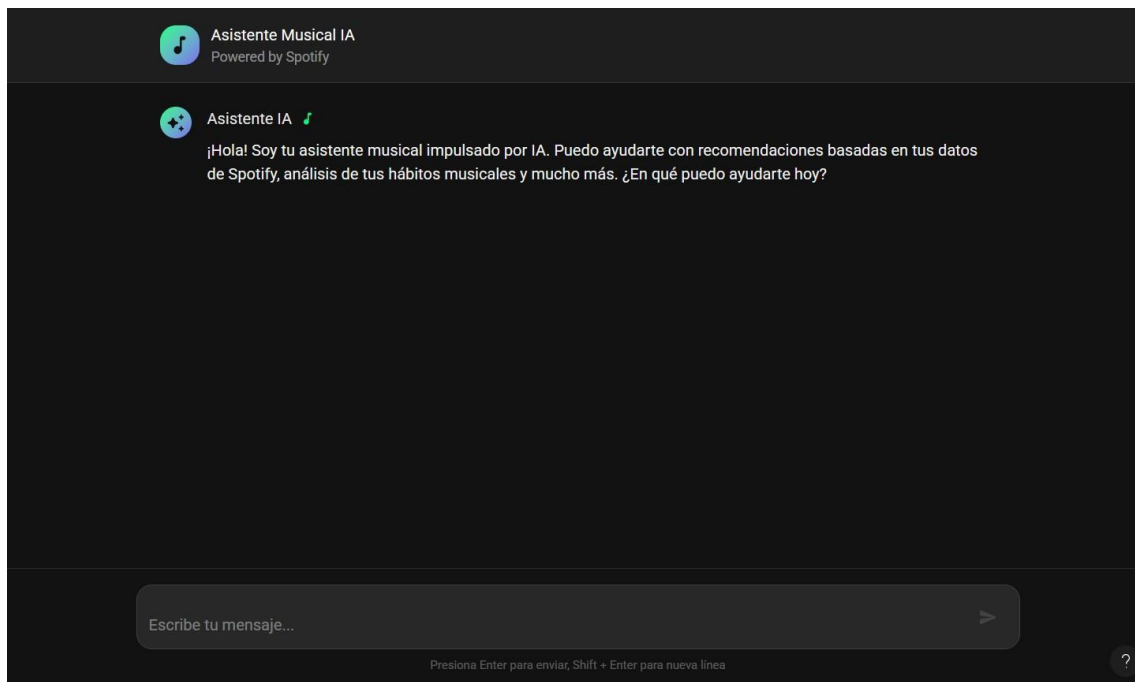


Figura 9. Imagen prototipo de interfaz de usuario.

Fuente: Figma

2.7. Sistemas similares existentes.

El mercado de software impulsado por Modelos de Lenguaje Grande experimenta un notable auge, y hasta la fecha, han emergido numerosos proyectos que ofrecen diversas soluciones para el avance de la industria musical. Con el propósito de sustentar que este proyecto representa una solución novedosa que aporta un valor diferenciador, se procede a la mención de sus similares.

- **Spotify - DJ AI:** Constituye el principal referente del proyecto. Utiliza la capacidad de un LLM para operar un asistente de voz con la habilidad de comprender peticiones formuladas en lenguaje natural y generar listas de reproducción basadas en dichas solicitudes.

Diferencia: Tal como se ha expuesto previamente, la presente iniciativa se centra en la conversación y la narración con el objetivo de motivar al usuario a emprender su propia exploración, un enfoque que resulta opuesto, pero a la vez complementario, al sistema mencionado.

- **Songtell:** Integra inteligencia artificial generativa con el fin de interpretar y explicar la lírica de las composiciones musicales.¹¹

Diferencia: El enfoque conversacional del proyecto busca englobar, de una manera menos especializada, diversas funcionalidades, entre las cuales se incluye la explicación de letras. Adicionalmente, utiliza la información del perfil de Spotify del usuario para ofrecer recomendaciones personalizadas.

- **Modelos LLM** (ChatGPT, Gemini, Grok, etc): Estos modelos proporcionan una interfaz de texto cuya función es responder a cualquier tipo de requerimiento del usuario, fundamentándose en el conocimiento accesible en la web y ofreciendo una explicación redactada en lenguaje natural.

Diferencia: El proyecto establece conexión con la cuenta de *Spotify* del usuario con el fin de utilizar su información para generar recomendaciones personalizadas, empleando una serie de prompts ajustados para optimizar la experiencia de una conversación con temática musical.

- **Suno AI:** Emplea un modelo de inteligencia artificial para la generación de composiciones musicales a partir de las instrucciones proporcionadas por el usuario.¹²

Diferencia: El presente proyecto no centra su objetivo en la generación de contenido musical, sino en el desarrollo de un chatbot conversacional enfocado en la recomendación de nuevos elementos musicales. Esto incluye la provisión de información adicional con el propósito de motivar la exploración de nueva música.

2.8. Modelado UML y conceptos asociados

Para una correcta planificación y diseño de la aplicación propuesta, se emplea el Lenguaje Unificado de Modelado (UML), el cual permite representar de

¹¹ Página oficial: <https://songtell.com/>

¹² Página oficial: <https://suno.com/>

manera estructurada los distintos aspectos de un sistema, como sus funcionalidades, flujos, componentes y relaciones con la base de datos. A continuación, se definen los principales diagramas y conceptos utilizados:

a) Diagrama de actividad

Un diagrama de actividad es una representación gráfica del flujo de trabajo o de las actividades que realiza un sistema o proceso. Este tipo de diagrama modela tanto procesos secuenciales como decisiones y concurrencias. Es útil para visualizar cómo se comporta el sistema ante diferentes caminos de ejecución, permitiendo identificar cuellos de botella y optimizar procesos.

b) Diagrama de componentes y despliegue

El diagrama de componentes muestra la estructura del sistema desde el punto de vista del *software*. Representa los distintos módulos, bibliotecas o servicios, y sus relaciones. Es fundamental para visualizar cómo se divide la aplicación (*frontend*, *backend*, base de datos, etc.).

Por su parte, el diagrama de despliegue representa la arquitectura física del sistema, incluyendo los nodos (servidores o dispositivos físicos/virtuales) y los artefactos que se ejecutan en ellos. Este tipo de diagrama es esencial para comprender cómo se implementará el sistema en la infraestructura tecnológica disponible.

c) Requisitos Funcionales

Los requisitos funcionales son especificaciones que definen lo que el sistema debe hacer. Representan funcionalidades explícitas. Estos requisitos se derivan directamente de las necesidades del usuario y se documentan para guiar el desarrollo del software, asegurando que se cumplan las expectativas de los interesados.

d) Requisitos no funcionales

Los requisitos no funcionales son especificaciones que delimitan las restricciones y la capacidad operativa del sistema. Estos definen el correcto funcionamiento del software, garantizando el cumplimiento de los mínimos necesarios para la prestación del servicio.

e) Casos de uso y Diagrama de casos de uso

Un caso de uso describe una funcionalidad del sistema desde el punto de vista del usuario. Incluye actores (usuarios o sistemas externos) y las

interacciones que realizan. Los diagramas de casos de uso permiten representar de forma gráfica las funcionalidades del sistema y quiénes las utilizan.

f) Diagrama de clases

Un diagrama de clases representa la estructura estática de un sistema mediante el modelado de clases, atributos, operaciones y relaciones entre objetos.

g) Diagrama de secuencia

Un diagrama de secuencia proporciona una visualización de la interacción entre los objetos o componentes de un sistema en un orden cronológico, enfocándose principalmente en el flujo de mensajes y la secuencia de eventos asociados a una tarea específica.

3. PROPUESTA DE SOLUCIÓN

En este capítulo se presentan los componentes principales relacionados al desarrollo e implementación de la solución, tales como: diseño, desarrollo y requerimientos de la aplicación web; así como también las pruebas unitarias necesarias para asegurar su correcto funcionamiento.

3.1. Gestión de tiempo

El proyecto emplea la metodología SCRUM adaptada para un desarrollo individual. La gestión del tiempo, estructurada en sprints de una semana de duración aproximadamente, se articula en torno a tres fases principales: Investigación, desarrollo y pruebas.

- **Sprint 1:** Dedicado a la investigación, se destina esta semana a explorar las modalidades de implementación de la solución propuesta y a identificar las herramientas más adecuadas para dicho objetivo.
Duración: 1 semana
- **Sprint 2:** Configuración de la arquitectura base, implementación de la autenticación de usuarios y desarrollo de la interfaz de usuario.
Duración: 1.5 semanas
- **Sprint 3:** Centrado en la configuración de las solicitudes API en el backend.
Duración: 1.5 semanas
- **Sprint 4:** Configuración del modelo LLM, incluyendo la gestión de prompts con una finalidad conversacional.
Duración: 0.5 semana
- **Sprint 5:** Integración final de componentes y despliegue del proyecto en la nube.
Duración: 1 semana
- **Sprint 6:** Destinado a las pruebas, esta última semana se dedica a la fase de pruebas funcionales, corrección de errores y difusión para la recopilación de retroalimentación.
Duración: 1 semana

Duración total estimada: 6.5 semanas

La duración estimada posibilita una mejor distribución del tiempo y una mayor capacidad de adaptación ante imprevistos, considerando que el plazo total

disponible para el desarrollo del trabajo de título consta de 8 semanas.

3.1.1. Carta Gantt

Para visualizar gráficamente lo descrito en este punto, se hace uso de una carta Gantt, un diagrama que expresa las tareas a realizar a través de una línea de tiempo.

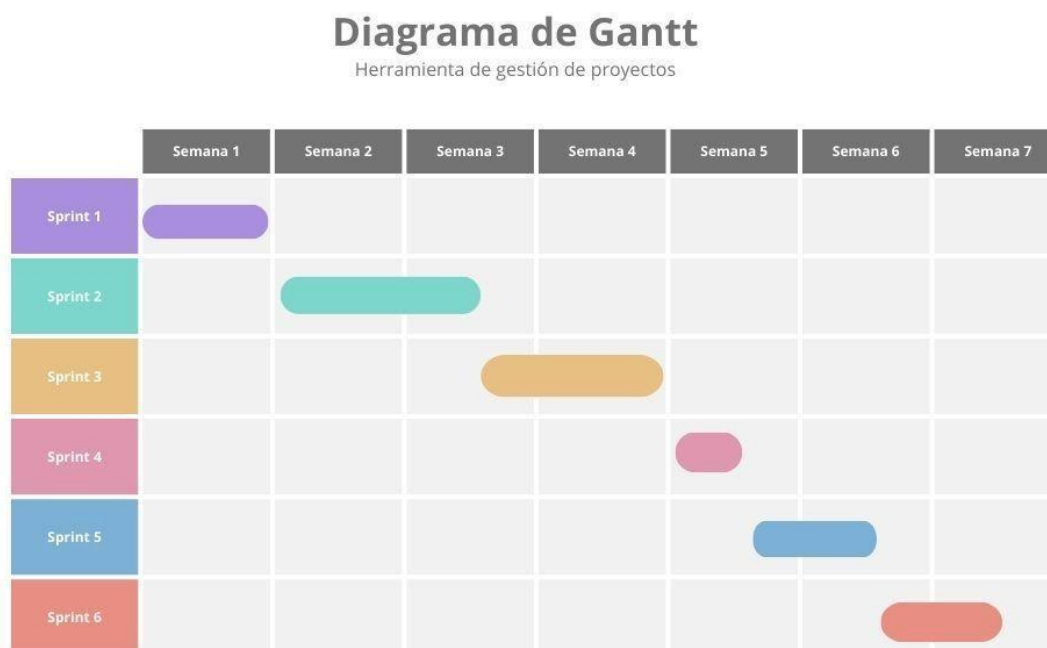


Figura 10. Diagrama de Gantt de Trabajo de Título.
Fuente: Elaboración propia.

3.1.2. Evaluación de usuarios.

Para evaluar la efectividad de la experiencia del proyecto, tanto en términos de precisión como de narratividad percibida en las respuestas, se implementará una validación empírica mediante el despliegue del prototipo funcional en la plataforma Vercel.

Respecto a la selección de la muestra, se invitará a utilizar el prototipo a usuarios de redes sociales. Tras la fase de prueba, los usuarios proporcionarán su retroalimentación a través de una escala de puntuación de 1 a 5 estrellas (donde 1 indica una experiencia deficiente y 5 una experiencia satisfactoria), una selección binaria (Bien o mal) sobre las siguientes interrogantes: “¿La interacción con el bot resultó agradable e interesante?”, “¿La conversación le motivó a una mayor exploración o audición musical?” y

por último una caja de comentarios donde el usuario podrá agregar retroalimentación con sus propias palabras. Las opiniones recopiladas se complementarán con el registro de la actividad en la plataforma web, específicamente el tiempo de uso. Se garantiza la confidencialidad de la información recibida, asegurando que las opiniones serán almacenadas de forma anónima, limitándose a una respuesta por individuo.

La selección de la muestra de voluntarios estará delimitada con el propósito de optimizar el control presupuestario. La demostración estará disponible para los primeros 20 usuarios que se registren como usuarios de prueba, quienes podrán utilizar las funcionalidades de la aplicación *web*, con un límite de 4 peticiones diferentes por usuario.

3.2. Gestión de recursos

Se establece en esta sección los recursos a utilizar para el avance del proyecto, constan de recursos relacionados directamente al desarrollo de software, incluyendo hardware, software y recursos en la nube.

Adicionalmente se estimará un costo aproximado del desarrollo, el costo de los elementos ya adquiridos no será considerados.

Hardware:

Computador personal: 16 GB de memoria RAM, procesador *AMD ryzen 7 5700X 8-Core Processor*, 1 TB de almacenamiento en memoria SSD.

Software:

- Backend: Python FastAPI
- Frontend, Vite+React.
- Base de datos: *Firestore*.
- Orquestador LLM: *Langchain*.
- Herramientas de desarrollo: *SDK google generative*.

Servicios en la nube:

- Modelo de lenguaje LLM: *Google Gemini Pro 1.5*.
- Despliegue frontend: Vercel.

- -Despliegue Backend: Render

3.2.1. Estimación de costos de los recursos:

Para la fase de desarrollo, no se prevén costos adicionales, dado que se utilizará el nivel gratuito del modelo Google Gemini. Con las limitaciones establecidas en la fase de pruebas, se espera que el proyecto se mantenga dentro del uso de los niveles de servicio gratuitos. En caso de que se contemple escalar el proyecto para un uso a nivel global, se requerirá la elaboración de un nuevo presupuesto que considere variables adicionales.

Considerando los parámetros determinados en el punto 3.1.1, el cálculo de costo del uso del modelo LLM Gemini 2.5 Flash se determina de la siguiente forma: **Límites LLM Free tier:** Peticiones por minuto: 10 Tokens por minuto: 250.000 Peticiones por día: 25

Considerando las limitaciones inherentes, la fase de pruebas se llevará a cabo con una muestra de 5 usuarios diarios distribuidos en diversos horarios. Se establecerá un límite de 4 peticiones por usuario, lo cual garantiza que la cuota de la capa gratuita del modelo LLM no sea excedida, asegurando así la correcta funcionalidad del sistema durante las pruebas. Es importante señalar que estas pruebas se centran exclusivamente en la funcionalidad y no incluyen una evaluación exhaustiva de la capacidad de red del sistema. Dicha evaluación se pospone para una fase futura, orientada a un objetivo comercial, y no forma parte del alcance del presente trabajo de título.

3.3. Análisis de riesgo y mitigaciones

Algunos de los riesgos que se consideran importantes de evaluar, incluyendo posibles riesgos durante y posteriores al desarrollo. En base a la matriz de riesgo, se utilizará como métrica la probabilidad de ocurrencia, medida en puntuación de 1 a 5, y la gravedad del riesgo, medido con la misma puntuación. Estas métricas se multiplican entre sí para dar como total el impacto de riesgo, el resultado determina la prioridad y estrategia con la que se aborda el riesgo.

1. **Servicios en la nube caídos:** Los servicios alojados en la nube, debido a cualquier contingencia, no se encuentran operativos, impidiendo el funcionamiento adecuado del proyecto.
 - Gravedad: Severa. (5)
 - Probabilidad: Baja. (1)
 - Impacto de riesgo: 5
 - Prioridad: Media.
 - Estrategia: Aceptar la situación y ofrecer disculpas por la indisponibilidad del servicio.

2. **Solicitud no musical:** El usuario realiza consultas al bot que se desvían del objetivo original de la recomendación musical, lo que genera una utilización inadecuada en términos de retroalimentación.
 - Gravedad: Significativa. (3)
 - Probabilidad: Moderada. (3)
 - Impacto de riesgo: 9
 - Prioridad: Media.
 - Estrategia: Mitigar el riesgo instruyendo al bot para que dirija la conversación exclusivamente a la temática de recomendación musical. En caso de reiteración con voluntarios enumerados, se deberá repetir la fase de pruebas con una muestra reducida.

3. **Latencia excesiva:** El sistema exhibe un tiempo de respuesta prolongado, lo que resulta en la pérdida de interés por parte del usuario.
 - Gravedad: Significativa (3)
 - Probabilidad: Probable (4)
 - Impacto de riesgo: 12
 - Prioridad: Alta
 - Estrategia: Mitigar el riesgo de desinterés del usuario manteniendo una percepción de progreso en la interfaz de usuario mediante el uso de loading skeletons y respuestas por streaming.

4. **Información limitada del usuario:** El usuario posee un perfil de Spotify relativamente nuevo o con un uso poco frecuente, lo que resulta en una representación incompleta de sus intereses musicales y dificulta la precisión de las recomendaciones.

- Gravedad: Significativa (3).
 - Probabilidad: Probable (4).
 - Impacto del riesgo: 12.
 - Prioridad: Alta.
 - Estrategia: Mitigar la dificultad de las recomendaciones instruyendo al bot para que no dependa exclusivamente del perfil de usuario.
5. **Respuesta poco satisfactoria:** El sistema automatizado proporciona una respuesta que no cumple con las expectativas o requerimientos del usuario.
- Gravedad: Significativa (3)
 - Probabilidad: Probable (4)
 - Impacto de riesgo: 12
 - Prioridad: Alta
 - Estrategia: Se debe evitar en la mayor medida posible este resultado, realizando el desarrollo del proyecto con decisiones óptimas y bien analizadas. Es imprescindible aceptar la retroalimentación negativa y realizar las correcciones necesarias para maximizar la experiencia del usuario.
6. **Respuesta inventada:** El sistema genera información que carece de veracidad o sustento.
- Gravedad: Mayor (4)
 - Probabilidad: Probable (4)
 - Impacto del riesgo: 16
 - Prioridad: Alta
 - Estrategia: Mitigar la probabilidad de respuestas inventadas, proporcionando al sistema instrucciones precisas para que nunca emita información sin el debido respaldo. En caso de que el sistema no disponga de suficiente conocimiento, deberá notificar su incapacidad para ofrecer asistencia en lugar de proporcionar información inventada.
7. **El access token de Spotify ha caducado:** la credencial de acceso que autoriza la conexión al perfil de usuario de Spotify expira tras una hora de inactividad, interrumpiendo el flujo del proyecto.
- Gravedad: Significativo (3) Probabilidad: Casi seguro (5)

- Impacto de riesgo: 15
 - Prioridad: Alta
 - Estrategia: Evitar la interrupción del servicio mediante la automatización de la renovación del token en caso de expiración.
8. **Sobreconsumo de tokens del Modelo de Lenguaje de Gran Escala:** La interacción de los usuarios durante la fase de prueba proyectada podría superar el límite del nivel gratuito de Gemini, generando costos operacionales adicionales.
- Gravedad: Alta (4)
 - Probabilidad: Casi seguro (5)
 - Impacto del riesgo: 20
 - Prioridad: Crítica.
 - Estrategia: Se propone mitigar el sobreconsumo de tokens mediante la delimitación estricta de la cantidad de mensajes, el número de caracteres y la cantidad de voluntarios para la fase de pruebas, tal como se detalla en la sección 3.1.1.
9. **Caída de servicio debido a exceso de solicitudes:** Un número considerable de usuarios accede al servicio de manera concurrente, lo que ocasiona la interrupción de este.
- Gravedad: Mayor (4)
 - Probabilidad: Alta (4)
 - Impacto de riesgo: 16
 - Prioridad: Alta
 - Estrategia: Evitar esta situación mediante la implementación de medidas de control de tráfico y, si es necesario, la utilización de mecanismos de puesta en cola.
10. **Inyección de prompt por parte del usuario:** Vulneración de seguridad que consiste en introducir una instrucción diseñada para engañar al modelo LLM, ejecutando acciones no deseadas.
- Gravedad: Severo (5)
 - Probabilidad: Probable (4)
 - Impacto de riesgo: 20
 - Prioridad: Muy alta

- Estrategia: Mitigar las posibilidades de vulneración de seguridad mediante el uso de delimitadores en la instrucción inicial del modelo que enfatizan la imperativa necesidad de acatar rigurosamente las directrices. Además, se debe asegurar que el modelo solo gestione información a la cual el usuario ya posee acceso por otros canales.

Tabla 2. Tabla de riesgos.

Gravedad/ Probabilidad	Mínimo (1)	Menor (2)	Significativo (3)	Mayor (4)	Severo (5)
Casi seguro(5)			7	8	
Probable (4)			3-4-5	6-9	10
Moderado (3)			2		
Poco probable (2)					
Raro (1)					1

Fuente: Elaboración propia

La identificación y el análisis de riesgos son cruciales, ya que no solo permiten anticipar posibles fallos en el sistema, sino que también guían las decisiones de diseño y desarrollo de los procesos clave del proyecto. De igual importancia es la comprensión del flujo de trabajo actual, puesto que facilita la detección de debilidades. Por ello, a continuación, se presenta un diagrama de actividad del proceso. Este diagrama servirá como base fundamental para proponer mejoras que aseguren un flujo más robusto y seguro para el recomendador musical.

3.4. Diagrama de actividad recomendador musical

Las siguientes imágenes corresponden al mismo diagrama, segmentado para facilitar su resolución y lectura. Este diagrama representa el flujo de actividad desde el acceso del usuario a la página web que alberga la interfaz del proyecto hasta el abandono del sitio.

Diagrama de actividad

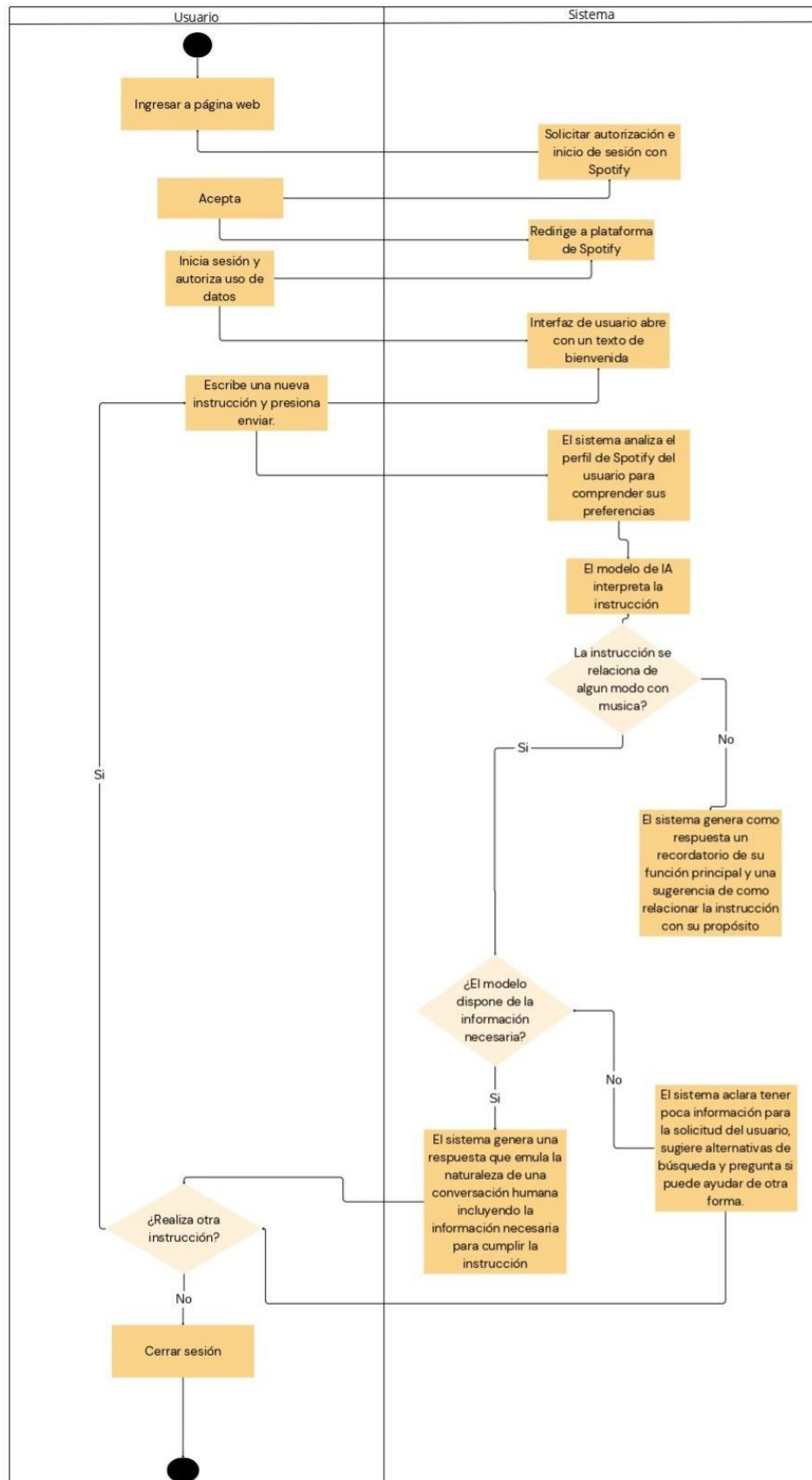


Figura 11. Diagrama de actividad para comenzar nuevo chat.
Fuente: elaboración propia

3.5. Modelo de negocio Canvas

Respecto a los elementos y la estrategia de negocio del proyecto, en el caso de ser implementado como un servicio con disponibilidad global, se exponen visualmente a través del formato del modelo de negocio Canvas. Dicha herramienta sintetiza los aspectos fundamentales para la sostenibilidad económica de la iniciativa, demostrando su viabilidad en un contexto comercial.

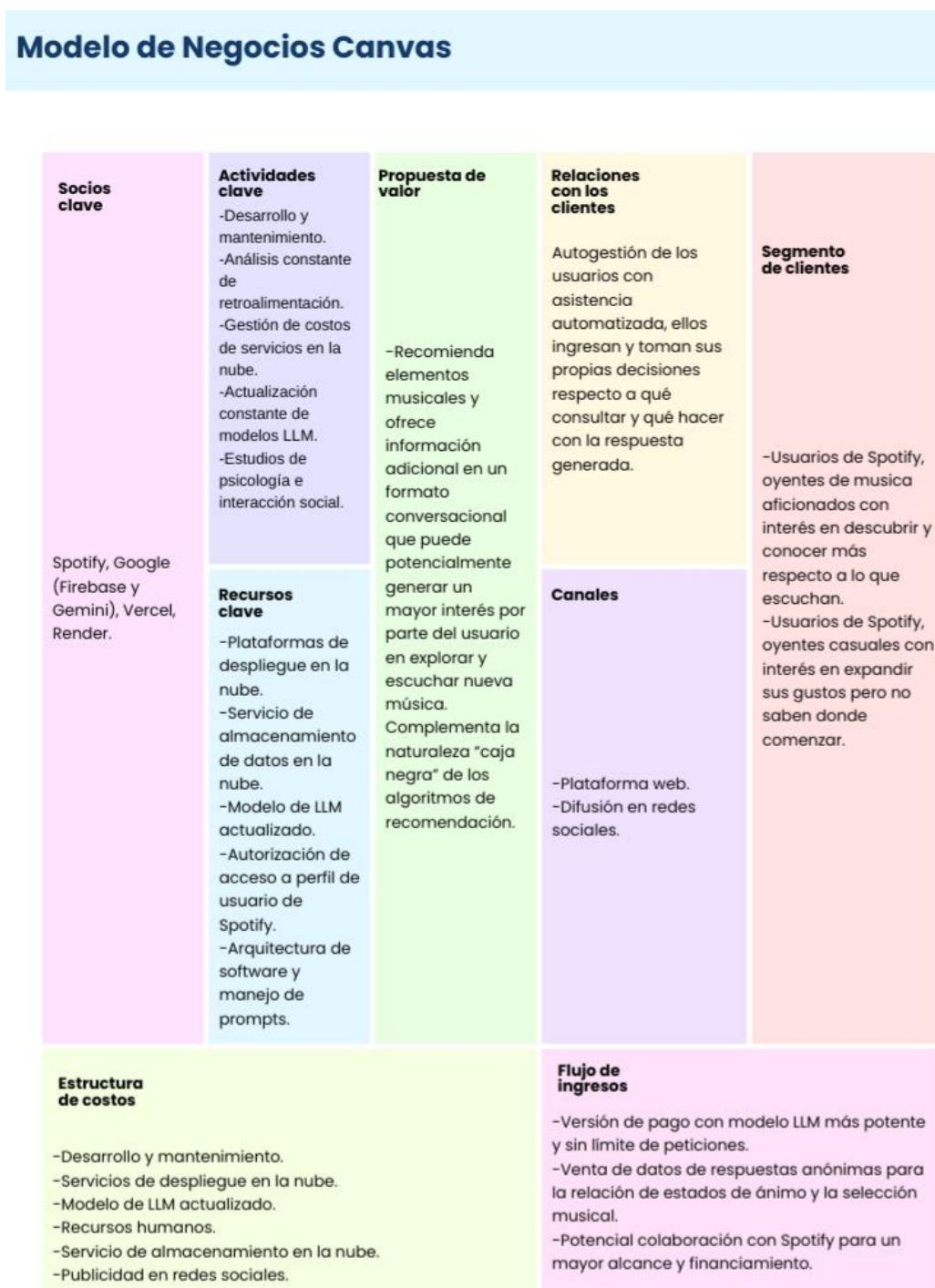


Figura 12. Modelo de negocio Canvas.
Fuente: Elaboración propia.

3.6. Requisitos funcionales.

Para asegurar el éxito en el desarrollo de una solución funcional, efectiva y en línea con los objetivos del proyecto, se han definido los siguientes requisitos esenciales. Estos requisitos especifican las acciones principales para que el sistema logre cumplir su objetivo de brindar información musical solicitada contemplando una experiencia conversacional que motive a la exploración y escucha.

- RF 1. El sistema deberá requerir la autorización del usuario para acceder a sus datos de Spotify.
- RF 2. Se debe implementar una redirección del usuario a la plataforma de Spotify para la autenticación e inicio de sesión, y la consecuente autorización de acceso.
- RF 3. El sistema ha de proveer una interfaz de chat que permita al usuario ingresar y enviar texto.
- RF 4. El sistema debe mostrar en pantalla un mensaje de bienvenida y una pequeña descripción de cómo usar el servicio.
- RF 5. Se requiere que el sistema utilice el modelo LLM integrado para la correcta interpretación de la información suministrada por el usuario.
- RF 6. El sistema debe extraer del perfil de Spotify del usuario los artistas más escuchados y las canciones reproducidas recientemente.
- RF 7. El sistema deberá realizar consultas a la API de Spotify para obtener información pertinente a la solicitud del usuario.
- RF 8. La respuesta del sistema se construirá a partir de la información recopilada, manteniendo un formato que simule una interacción conversacional humana.
- RF 9. El sistema puede entregar un enlace que redirecciona a *Spotify* para poder reproducir la recomendación otorgada.
- RF 10. El sistema debe garantizar la continuidad de la conversación, proporcionando al usuario la capacidad de responder a la sugerencia del recomendador.
- RF 11. El sistema debe almacenar las interacciones del usuario de un modo que garantice la no violación de su privacidad.
- RF 12. El sistema debe ofrecer un módulo de retroalimentación compuesto por preguntas diseñadas para impulsar su mejora continua.

RF 13. El sistema debe permitir el inicio de nuevas conversaciones donde no exista un contexto preestablecido.

RF 14. El sistema debe permitir borrar conversaciones del historial de chat.

3.7. Requisitos no funcionales

A fin de definir la calidad del sistema, independientemente de las funcionalidades establecidas, se presentarán los requisitos no funcionales que el sistema debe satisfacer para maximizar la satisfacción percibida por el usuario y la estabilidad de las funcionalidades.

RNF 1. El sistema debe minimizar la percepción de latencia desde el momento en que el usuario realiza una petición hasta que el sistema otorga una respuesta, integrando indicadores de carga y minimizando el tamaño del prompt entregado al LLM .

RNF 2. El sistema debe poder manejar múltiples solicitudes asíncronas al servicio de *Spotify* y *Google* con el reforzamiento necesario para evitar la caída del servicio en caso de surgir inconvenientes.

RNF 3. El sistema debe gestionar los *tokens* de manera óptima en la solicitud enviada al LLM, con el objetivo de minimizar el consumo de *tokens* por respuesta y mantener las funcionalidades principales.

RNF 4. El sistema deberá garantizar la confidencialidad de las credenciales gestionadas, previniendo su exposición a terceros ajenos al flujo principal.

RNF 5. El sistema debe manejar la comunicación con el usuario y los servicios externos a través del protocolo HTTPS para asegurar la integridad de los datos de tránsito.

RNF 6. El diseño de la interfaz de usuario debe utilizar principalmente colores de tonos oscuros y contraste alto para minimizar la fatiga visual del usuario y facilitar la lectura.

RNF 7. La interfaz debe ser responsiva y adaptarse correctamente a las resoluciones estándar de dispositivos electrónicos con acceso a búsquedas de internet, tales como computadores personales, *laptops* y teléfonos móviles.

RNF 8. Las respuestas otorgadas por el sistema deben entregar información verídica, minimizando la posibilidad de alucinación por parte del LLM.

3.8. Casos de uso.

A continuación, se presenta el Diagrama General de Casos de Uso y se describen los principales casos de uso del sistema de recomendación musical, los que se enfocan en las acciones realizadas por el usuario que ingresa a la página web.

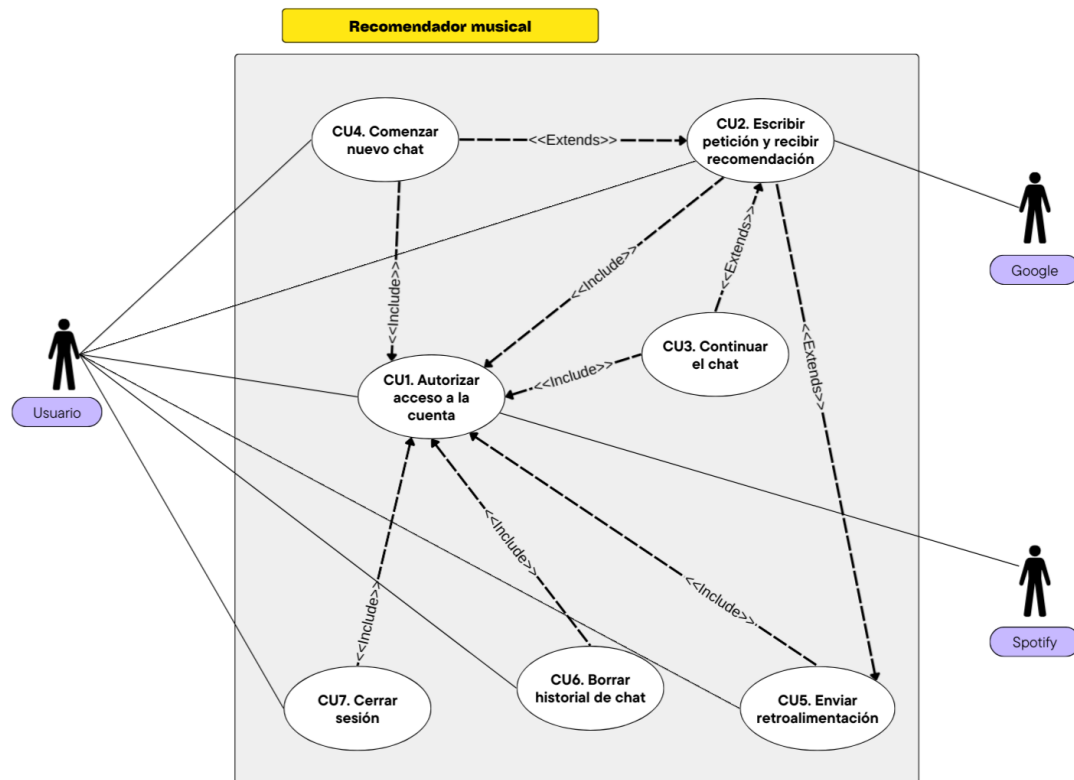


Figura 13. Diagrama general de casos de uso
Fuente: Elaboración propia

3.8.1. Caso de Uso: CU1 -: Autorizar acceso a cuenta.

Tabla 3. Caso de uso narrativo: CU1 – Autorizar acceso a cuenta

Nombre:	CU1: Autorizar acceso a cuenta.
Descripción:	El usuario presiona el botón de autorizar y se redirige a la plataforma de <i>Spotify</i> para completar el inicio de sesión y autorización. referencias cruzadas: "El sistema deberá requerir la autorización del usuario para acceder a sus datos de <i>Spotify</i> ".
Ref. Cruzadas:	
Actores:	Usuario - primario - activo. <i>Spotify</i> - secundario - pasivo.
Precondiciones:	- El usuario debe tener cuenta de <i>Spotify</i>. - El usuario debe ingresar a la página <i>web</i>.
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario accede al sitio web	2. El sistema solicita la autorización para acceder al perfil de usuario de <i>Spotify</i> mediante una interfaz emergente
3. El usuario concede la autorización al pulsar el botón de confirmación.	4. El sistema redirige al usuario a la plataforma de <i>Spotify</i> .
5. El usuario introduce sus credenciales y otorga el consentimiento de acceso	6. El sistema espera la respuesta de <i>Spotify</i>
	7. El sistema redirige al usuario de vuelta al sitio web.
	8. El sistema posee acceso a la información del usuario de <i>Spotify</i> .
2.1 FLUJO ALTERNATIVO: El usuario ya tiene sesión activa	
1. El usuario accede al sitio web	2.1 El sistema conserva los tokens del usuario y éste posee una sesión abierta en la web de <i>Spotify</i>
	8. El sistema posee acceso a la información del usuario de <i>Spotify</i>

Fuente: Elaboración propia

3.8.2. Caso de uso: CU2 - Escribir petición y recibir recomendación.

Tabla 4. Caso de uso narrativo: CU2 – Escribir petición y recibir recomendación

Nombre:	CU2: Escribir petición y recibir recomendación
Descripción:	El usuario interactúa con el recomendador musical pidiéndole mediante chat una recomendación en lenguaje natural, el sistema comprende la solicitud y construye una respuesta que busque satisfacer la petición original.
Ref. Cruzadas:	Include: CU1. Autorizar acceso a la cuenta. Extends: CU5. Enviar retroalimentación.
Actores:	Usuario - primario - activo. LLM - Secundario - Pasivo. Spotify - Secundario - Pasivo
Precondiciones:	- El usuario debió haber autorizado el acceso a su perfil de Spotify por parte del sistema. - El sistema debe tener sus dependencias en la nube en funcionamiento.
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario se dirige a la barra de texto	
2. El usuario escribe su petición relacionada a la música	
3. El usuario presiona el botón de enviar.	4. El sistema realiza la generación de prompt utilizando la petición del usuario, sus datos de perfil de Spotify y las instrucciones base para el modelo y lo envía al LLM.
	5. El sistema carga la interfaz con <i>loading skeletons</i> .
	6. El sistema recibe la respuesta en formato JSON otorgada por el LLM
	7. El sistema utiliza los valores de los campos del JSON para realizar una búsqueda del elemento musical en la API de <i>Spotify</i>
	8. El sistema recibe la URL del elemento musical en la web de <i>Spotify</i> y la portada relacionada a éste.
	9. El sistema estructura la información recopilada en forma de una respuesta que emula una conversación humana, respondiendo la petición original con información de interés adicional, el url e

	imagen del elemento recomendado.
11. El usuario presiona el enlace entregado.	
2.1 FLUJO ALTERNATIVO: Petición no musical	
2.1. El usuario escribe una petición no relacionada a la música	
3.1. El usuario presiona el botón de enviar.	4. El sistema realiza la generación de prompt utilizando la petición del usuario, sus datos de perfil de Spotify y las instrucciones base para el modelo y lo envía al LLM.
	5. El sistema carga la interfaz con loading skeletons.
	6.1 El sistema recibe una respuesta en formato JSON otorgada por el LLM sin valores para búsqueda de elemento musical

Fuente: Elaboración propia

3.8.3. Caso de uso: CU3 - Continuar el chat.

Tabla 5. Caso de uso narrativo: CU3 – Continuar Chat

Nombre:	CU3 – Continuar Chat
Descripción:	El usuario desea solicitar recomendaciones siguiendo el hilo de una conversación previamente realizada
Ref. Cruzadas:	Include: CU1. Autorizar acceso a la cuenta. Extends: CU2. Escribir petición y recibir recomendación.
Actores:	Usuario - primario - activo. LLM - secundario - pasivo. Spotify - secundario - pasivo
Precondiciones:	- El usuario debe haber iniciado una conversación con el recomendador. - El sistema debe tener sus dependencias en la nube en funcionamiento.
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario se dirige al menú lateral y selecciona una conversación previamente comenzada por él.	2. La interfaz carga la conversación realizada entre él y el recomendador.
3. El usuario realiza una nueva petición.	4. El sistema conserva el hilo de la conversación y responde teniendo en cuenta los mensajes anteriores.

Fuente: Elaboración propia

3.8.4. Caso de uso: CU4 - Comenzar nuevo chat.

Tabla 6. Caso de uso narrativo: CU4 – Comenzar nuevo Chat

Nombre:	CU4 - Comenzar nuevo chat
Descripción:	El usuario opta por iniciar una nueva conversación con el fin de realizar peticiones sin un contexto preestablecido.
Ref. Cruzadas:	CU1. Autorizar acceso a la cuenta. CU2. Escribir petición y recibir recomendación.
Actores:	Usuario - primario - activo
Precondiciones:	- El usuario debió haber autorizado el acceso a su perfil de Spotify por parte del sistema. - El sistema debe tener sus dependencias en la nube en funcionamiento
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario se dirige al menú lateral y selecciona la opción para iniciar una nueva conversación.	2. El sistema otorga una interfaz de chat nueva sin contexto preestablecido.

Fuente: Elaboración propia

3.8.5. Caso de uso: CU5 - Enviar retroalimentación.

Tabla 7. Caso de uso narrativo: CU5 – Enviar retroalimentación

Nombre:	CU5 – Enviar retroalimentación
Descripción:	El usuario decide registrar su opinión respecto a la experiencia ofrecida por el servicio.
Ref. Cruzadas:	CU1. Autorizar acceso a la cuenta
Actores:	Usuario - primario - activo
Precondiciones:	- El usuario debió haber enviado al menos una petición y haber recibido respuesta por parte del sistema - El sistema debe tener sus dependencias en la nube en funcionamiento
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario presiona la opción de retroalimentación.	2. El sistema abre el módulo de retroalimentación
3. El usuario evalúa la experiencia proporcionada por el servicio asignando una calificación de 1 a 5 estrellas.	
4. El usuario completa los campos de preguntas y expresa su opinión en formato de texto libre.	
5. El usuario pulsa el botón de envío.	6. El sistema registra la retroalimentación.

Fuente: Elaboración propia

3.8.6. Caso de uso: CU6 - Borrar historial de chat.

Tabla 8. Caso de uso narrativo: CU6 – Borrar historial de chat

Nombre:	CU6 – Borrar historial de chat
Descripción:	El usuario decide borrar una conversación de su historial.
Ref. Cruzadas:	Include: CU1. Autorizar acceso a la cuenta.
Actores:	Usuario - primario - activo
Precondiciones:	- El usuario debió haber autorizado el acceso a su perfil de Spotify por parte del sistema. - El sistema debe tener sus dependencias en la nube en funcionamiento
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario se dirige a la barra lateral.	
2. El usuario visualiza la conversación que desea borrar.	
3. El usuario presiona el botón de borrar	4. El sistema elimina de la base de datos la conversación.
1.1 FLUJO ALTERNATIVO: Borrar historial desde ajustes	
1.1. El usuario se dirige a los ajustes de perfil.	
1.2 El usuario presiona la opción de ajustes	1.3. El sistema abre el modal de ajustes
1.4 El usuario selecciona opción de borrar historial.	1.5 El sistema borra de la base de datos todas las conversaciones del usuario.

Fuente: Elaboración propia

3.8.7. Caso de uso: CU7 - Cerrar sesión.

Tabla 9. Caso de uso narrativo: CU7 – Cerrar sesión

Nombre:	CU7 – Cerrar sesión.
Descripción:	El usuario decide desvincular su cuenta de Spotify del servicio.
Ref. Cruzadas:	Include: CU1. Autorizar acceso a la cuenta
Actores:	Usuario - Primario - activo
Precondiciones:	- El usuario debió haber autorizado el acceso a su perfil de Spotify por parte del sistema - El sistema debe tener sus dependencias en la nube en funcionamiento
FLUJO NORMAL	
Acción del Usuario	Respuesta del sistema
1. El usuario se dirige a los ajustes de perfil.	
2. El usuario elige la opción de ajustes.	3. El sistema abre el modal de ajustes.
4. El usuario selecciona opción de cerrar sesión.	5. El sistema desvincula la cuenta de Spotify del usuario del sistema.

Fuente: Elaboración propia

3.9. Diagrama de modelo de datos NoSQL

En esta sección se presenta el diagrama de colecciones que representa la arquitectura lógica de la base de datos NoSQL diseñada para almacenar la información de los usuarios registrados, incluyendo su historial de chat y la retroalimentación proporcionada. Este modelo permite comprender la estructura de la información que será almacenada en *Firestore*.

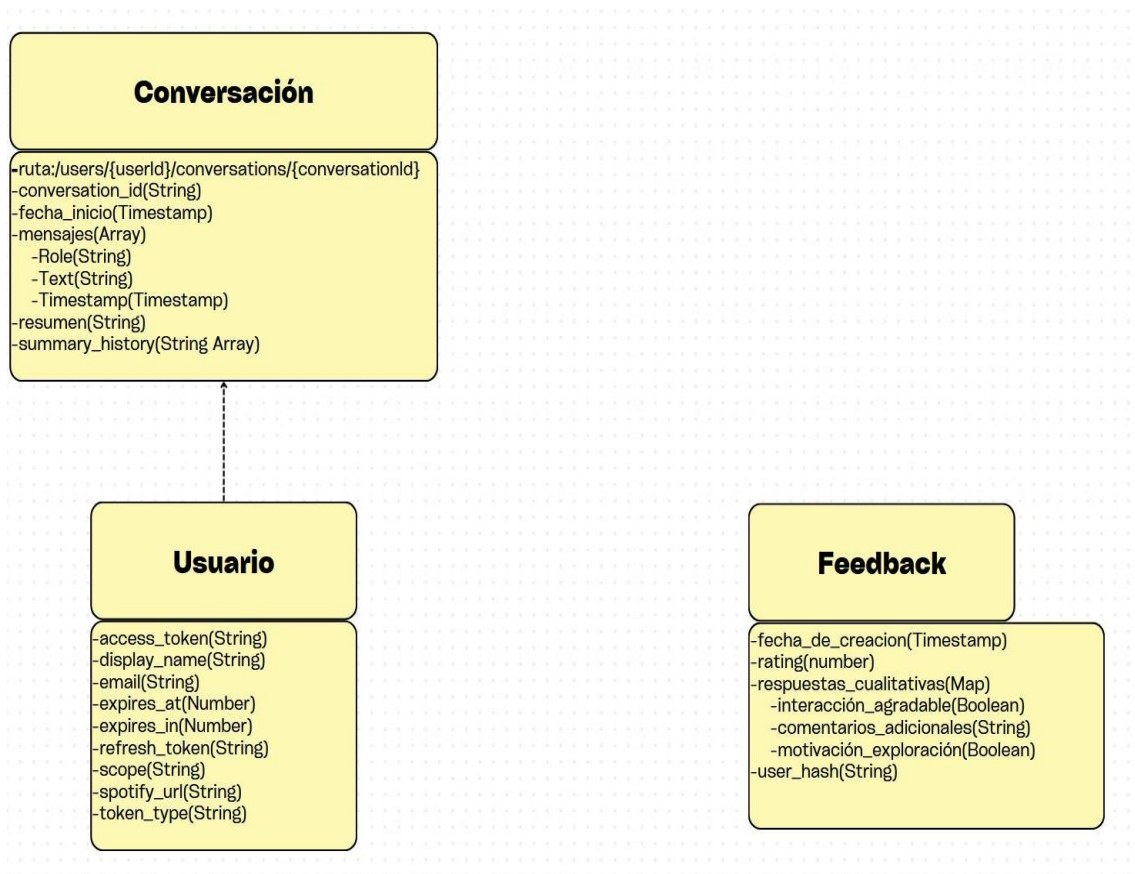


Figura 14. Diagrama de modelo de datos NoSql
Fuente: Elaboración propia.

3.10. Diagrama de clases

El siguiente diagrama representa la estructura y relaciones de las clases del sistema que componen la lógica del *backend*. Permite gestionar el flujo de información existente en la aplicación. Se presentan las entidades que gestionan el sistema, como el usuario, la conversación y el *feedback*, las clases que manejan los servicios externos y los controladores encargados de recibir las peticiones y delegar las funcionalidades.

Clases del Modelo de Datos:

- **Usuario:** Representa al usuario que accede al sistema con su cuenta de *Spotify*, posee un ID, *token* de acceso y *refresh* a *Spotify*, su correo, nombre de usuario y está relacionado a 0 o varias conversaciones desarrolladas entre él y el servicio.
- **Conversation:** Representa el diálogo entre usuario y sistema, se conforma por una lista de mensajes redactados por ambas partes, un resumen usado como título, y un *summary history* usado para mantener el hilo del LLM
- **Feedback:** Almacena la retroalimentación proporcionada por el usuario, incluyendo las puntuaciones y los comentarios.
- **Spotify Service:** encapsula las operaciones de obtención de datos del perfil de usuario y la búsqueda de elementos musicales.
- **LLM Service:** Maneja la lógica de generación de *prompt* y respuesta de parte de la LLM.
- **Database Service:** Maneja la extracción e inyección de datos dentro de la base de datos en *Firebase*.
- **Chat Controller:** Se encarga de las peticiones HTTPS recibidas desde *front* para el manejo de los chats y texto enviado del usuario.
- **Auth Controller:** Maneja las peticiones de login y logout de la cuenta *Spotify*.
- **Feedback Controller:** Recibe el *feedback* desde *frontend* por parte del usuario.

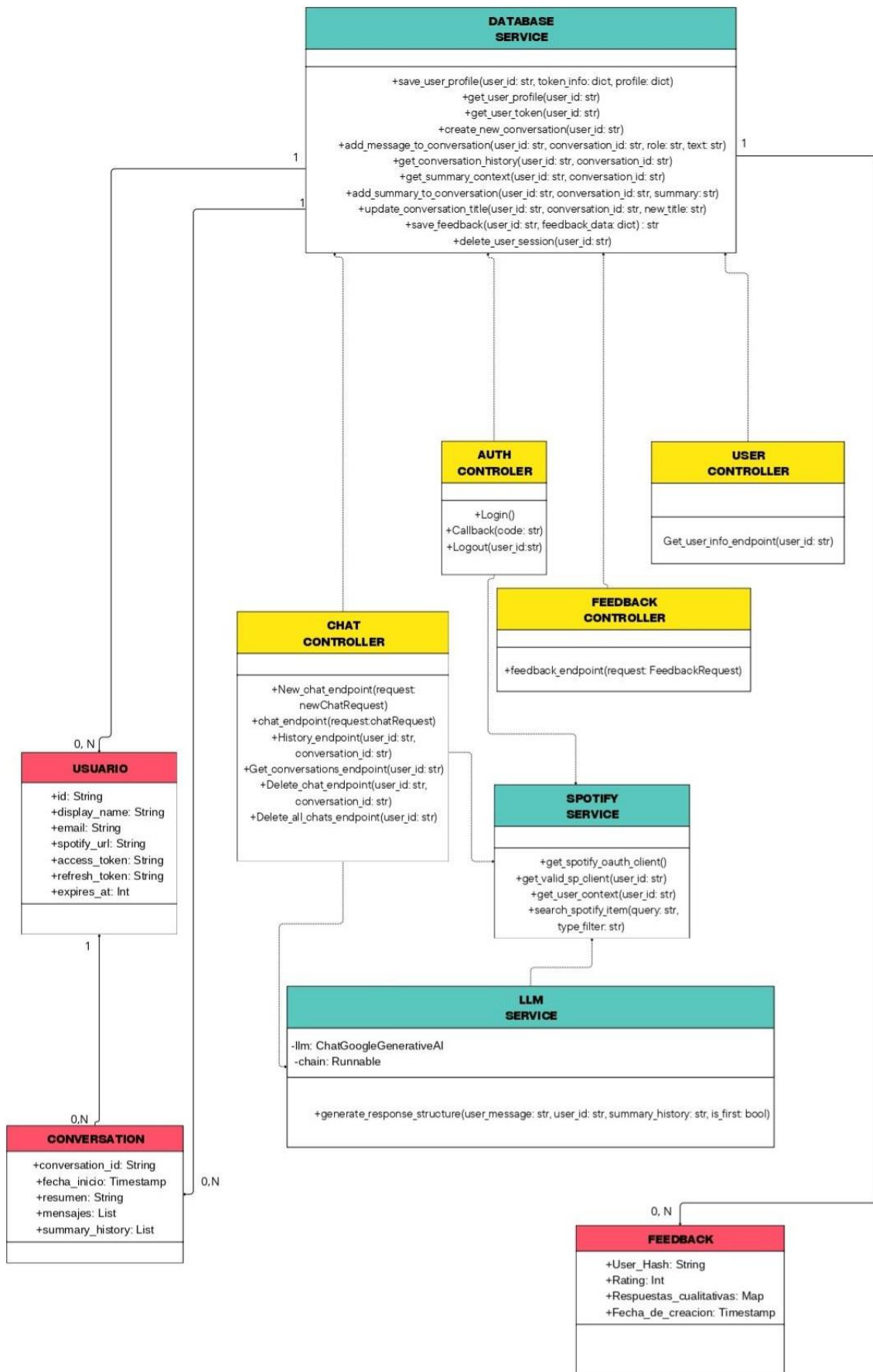


Figura 15. Modelo de Clases
Fuente: Elaboración propia

3.11.1 Diagrama de secuencia CU1.

Con el objetivo de describir la interacción entre los componentes del sistema, se exponen los diagramas de secuencia asociados a cada caso de uso. Estos esquemas permiten comprender la sucesión cronológica de las acciones y la carga operativa de cada etapa. Se inicia esta sección con el Caso de Uso 1, denominado Autorizar acceso a la cuenta.

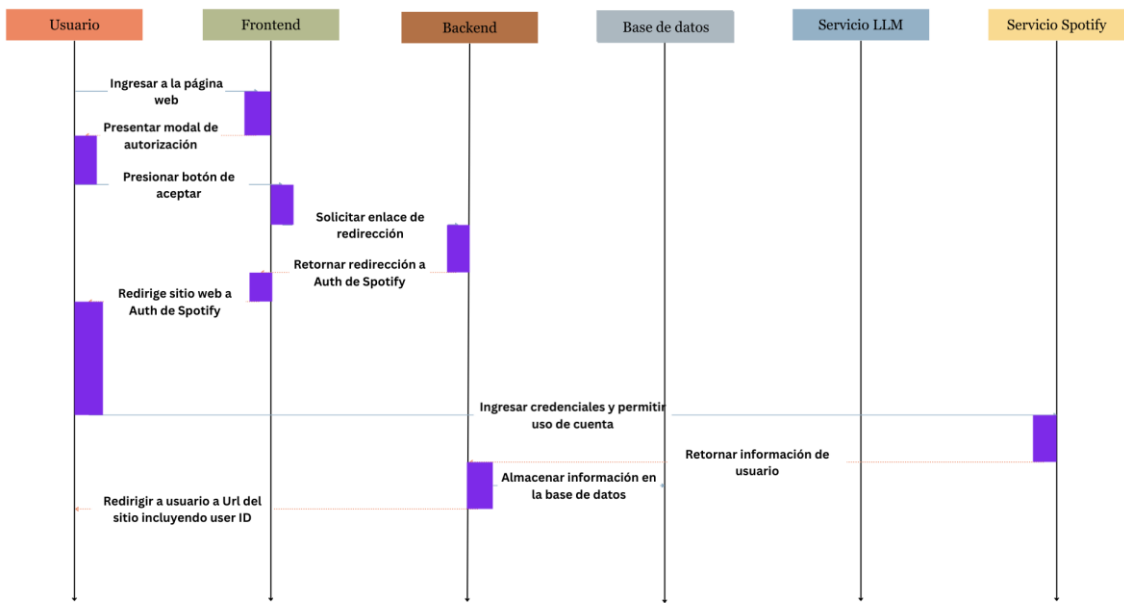


Figura 16. Diagrama de Secuencia: CU1 - Autorizar acceso a la cuenta.
Fuente: Elaboración propia

3.11.2 Diagrama de secuencia CU2.

El caso de uso que confiere el valor principal a la aplicación se denomina "Escribir petición y recibir recomendación". Con el objetivo de comprender de mejor manera el flujo generado por la interacción entre el usuario y el sistema, se presenta a continuación el diagrama de secuencia dedicado a este caso.

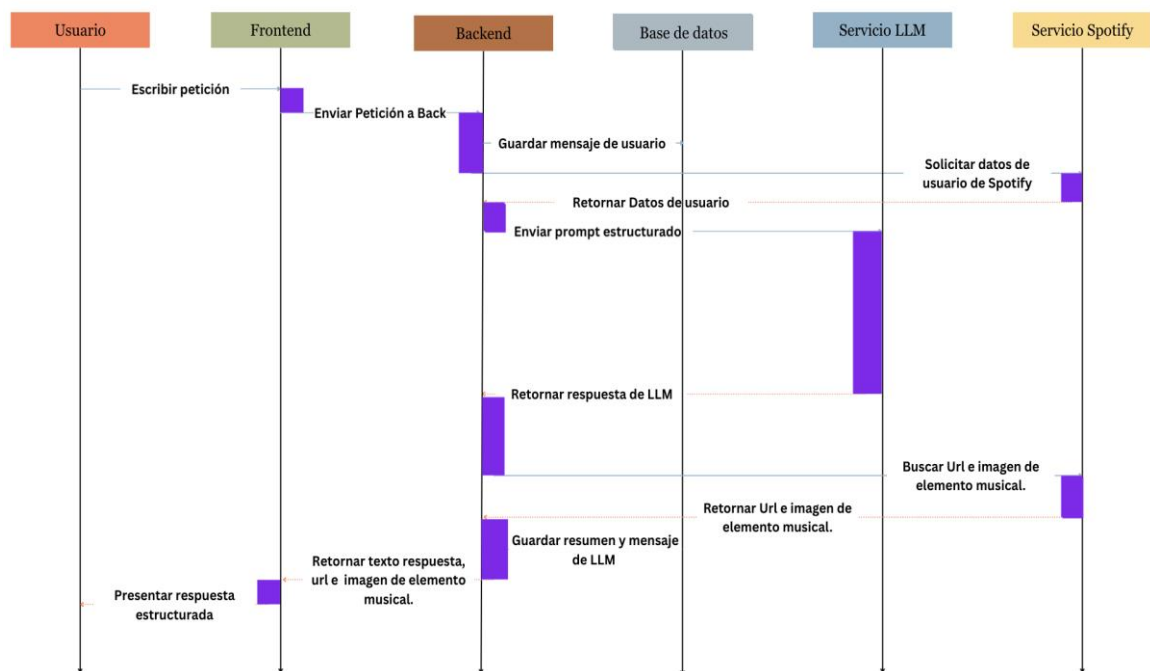


Figura 17. Diagrama de Secuencia: CU2 - Escribir petición y recibir recomendación
Fuente: Elaboración propia

3.11.3 Diagrama de secuencia CU3.

El siguiente diagrama de secuencia corresponde al caso de uso titulado “Continuar el chat”, donde se describe al usuario continuando una conversación previa, donde el bot mantiene el hilo de la conversación.

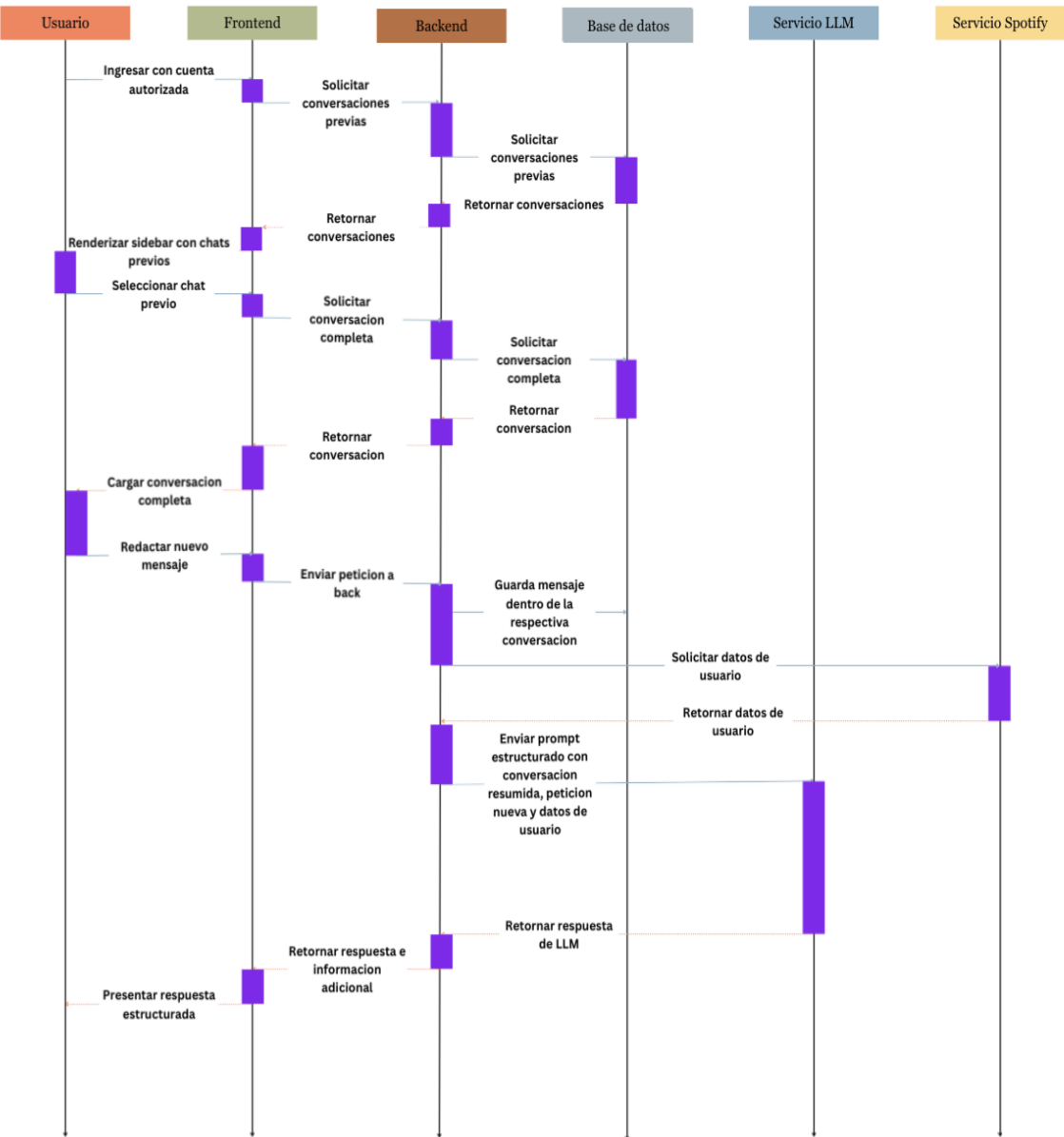


Figura 18. Diagrama de Secuencia: CU3 - Continuar el chat
Fuente: Elaboración propia

3.11.4 Diagrama de secuencia CU4.

El siguiente diagrama de secuencia corresponde al caso de uso titulado "Comenzar nuevo chat", donde se describe al usuario iniciando una nueva conversación sin un sesgo por contextos previos.

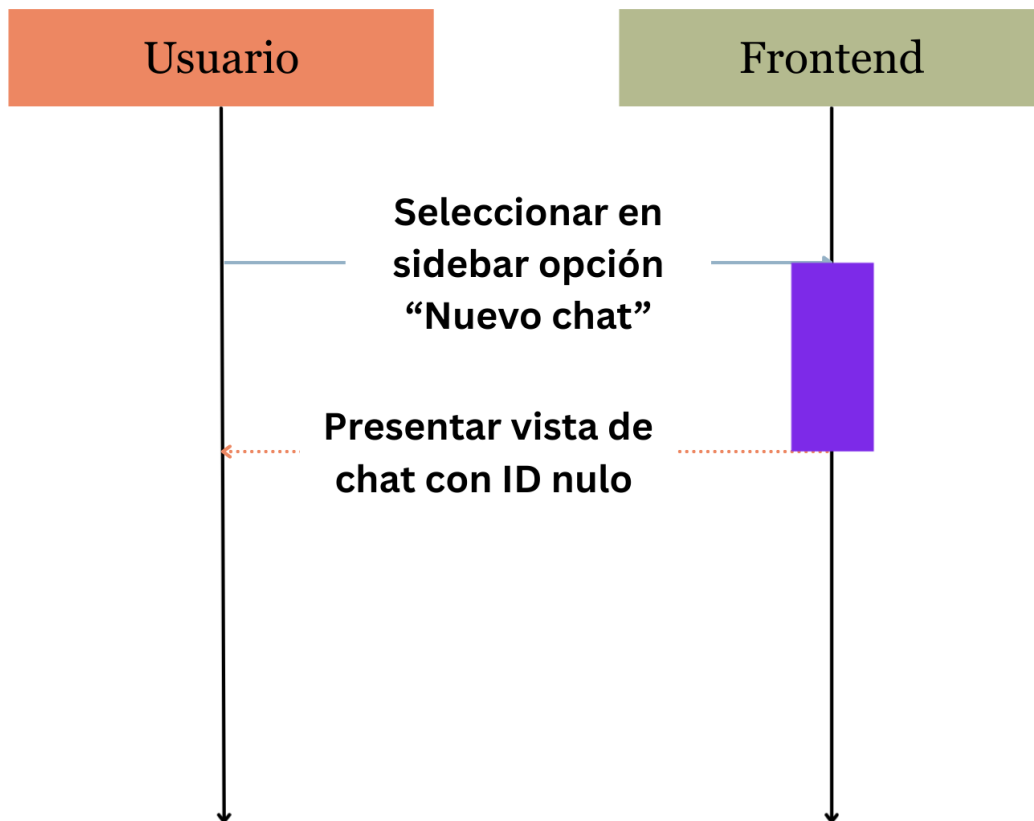


Figura 19. Diagrama de Secuencia: CU4 - Comenzar nuevo chat
Fuente: Elaboración propia

3.11.5 Diagrama de secuencia CU5.

El siguiente diagrama de secuencia corresponde al caso de uso titulado “Enviar retroalimentación”, donde se describe al usuario registrando su opinión respecto al servicio recibido.

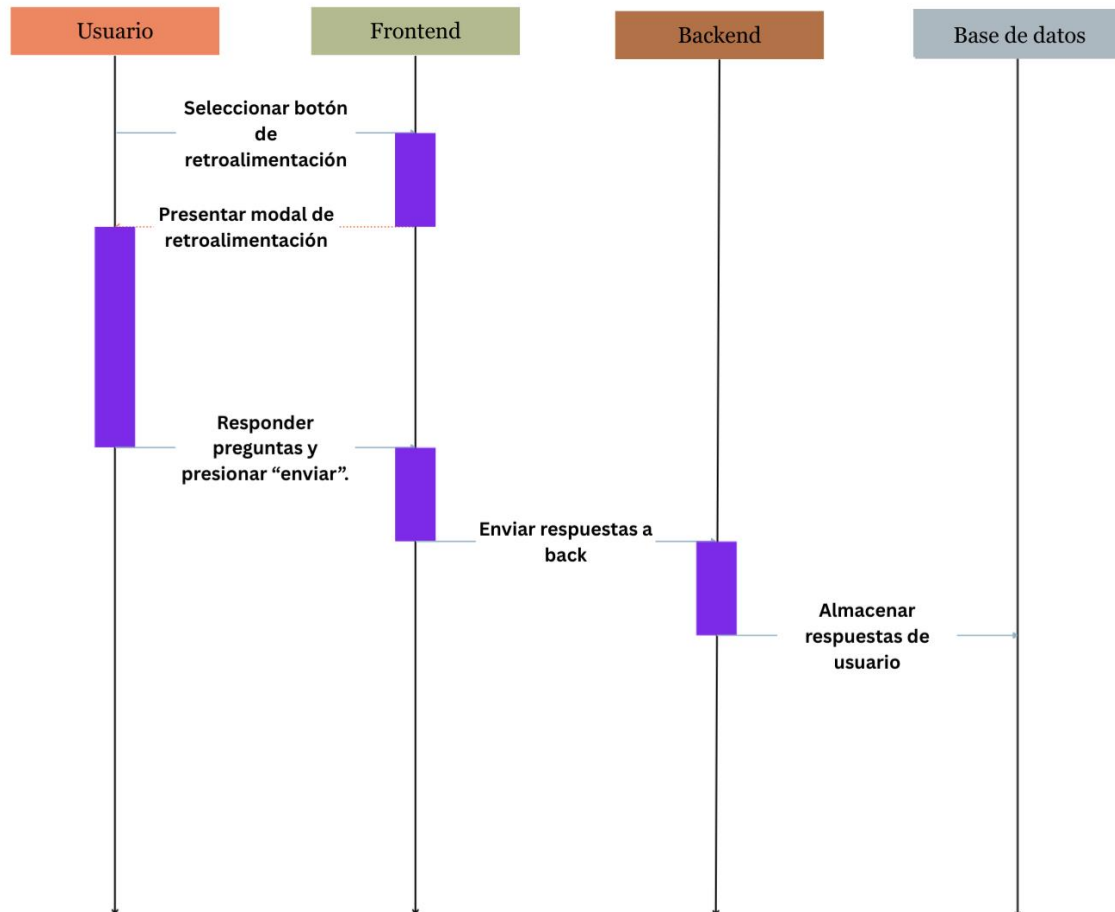


Figura 20. Diagrama de Secuencia: CU5 - Enviar retroalimentación
Fuente: Elaboración propia

3.11.6 Diagrama de secuencia CU6.

El siguiente diagrama de secuencia corresponde al caso de uso titulado "Borrar historial de chat", donde se describe al usuario eliminando sus conversaciones previas en el sitio web.

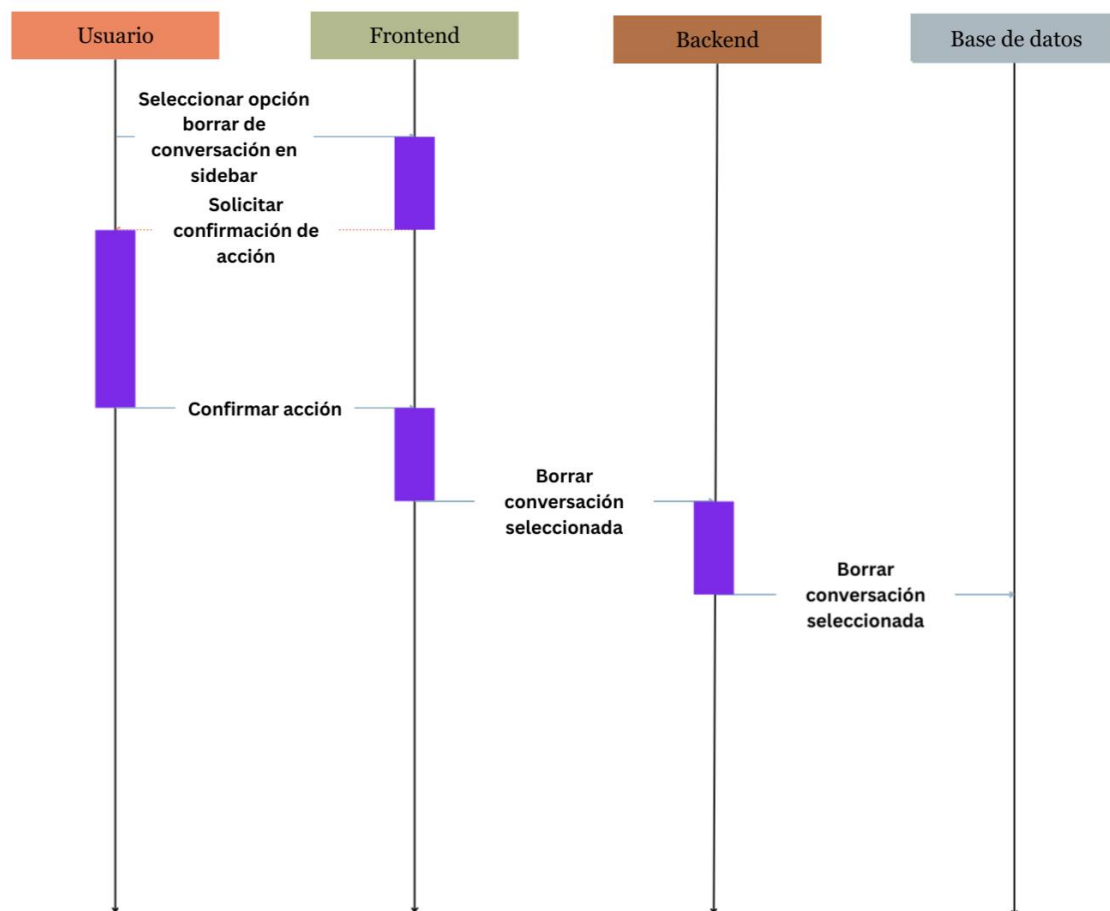


Figura 21. Diagrama de Secuencia: CU6 - Borrar historial de chat
Fuente: Elaboración propia

3.11.7 Diagrama de secuencia CU7.

El siguiente diagrama de secuencia corresponde al caso de uso titulado "Cerrar sesión", donde se describe al usuario cerrando sesión, eliminando la conexión de su cuenta de Spotify con el sitio web.

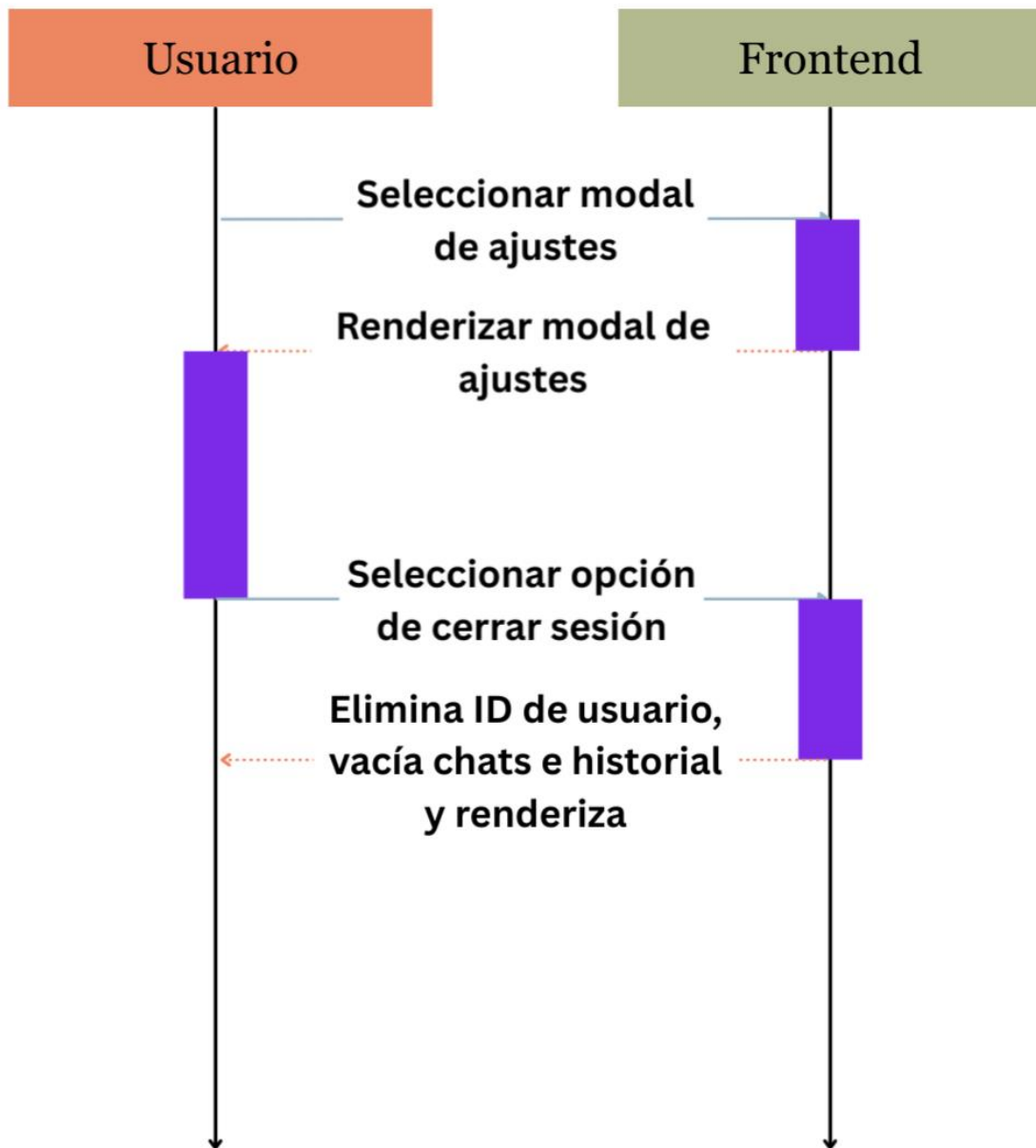


Figura 22. Diagrama de Secuencia: CU7 - Cerrar sesión
Fuente: Elaboración propia

3.12 Diagrama de componentes

Este diagrama ilustra la organización de la arquitectura del software, la cual se estructura en cuatro capas principales:

- **Capa de Interfaces:** Representa las vistas con las que interactúa el usuario final. Incluye la interfaz de autorización, la interfaz de chat, la barra lateral y el módulo de retroalimentación.
- **Capa de *Backend*:** Define la lógica de negocio que sustenta la operatividad de las interfaces. Cada clase se responsabiliza del procesamiento de la información y de la generación de las acciones visibles en la interfaz.
- **Capa de Datos de *Firebase*:** Corresponde a la administración de las colecciones almacenadas en la base de datos NoSQL de Firebase.
- **Capa de servicios externos:** Se refiere a los servicios ajenos al proyecto a los que se realizan consultas de información y se utilizan sus funcionalidades.

Las líneas punteadas indican las relaciones entre componentes, de esta forma se visualiza cómo las interfaces utilizan las clases y cómo estas acceden a la web y almacenan la información en la base de datos.

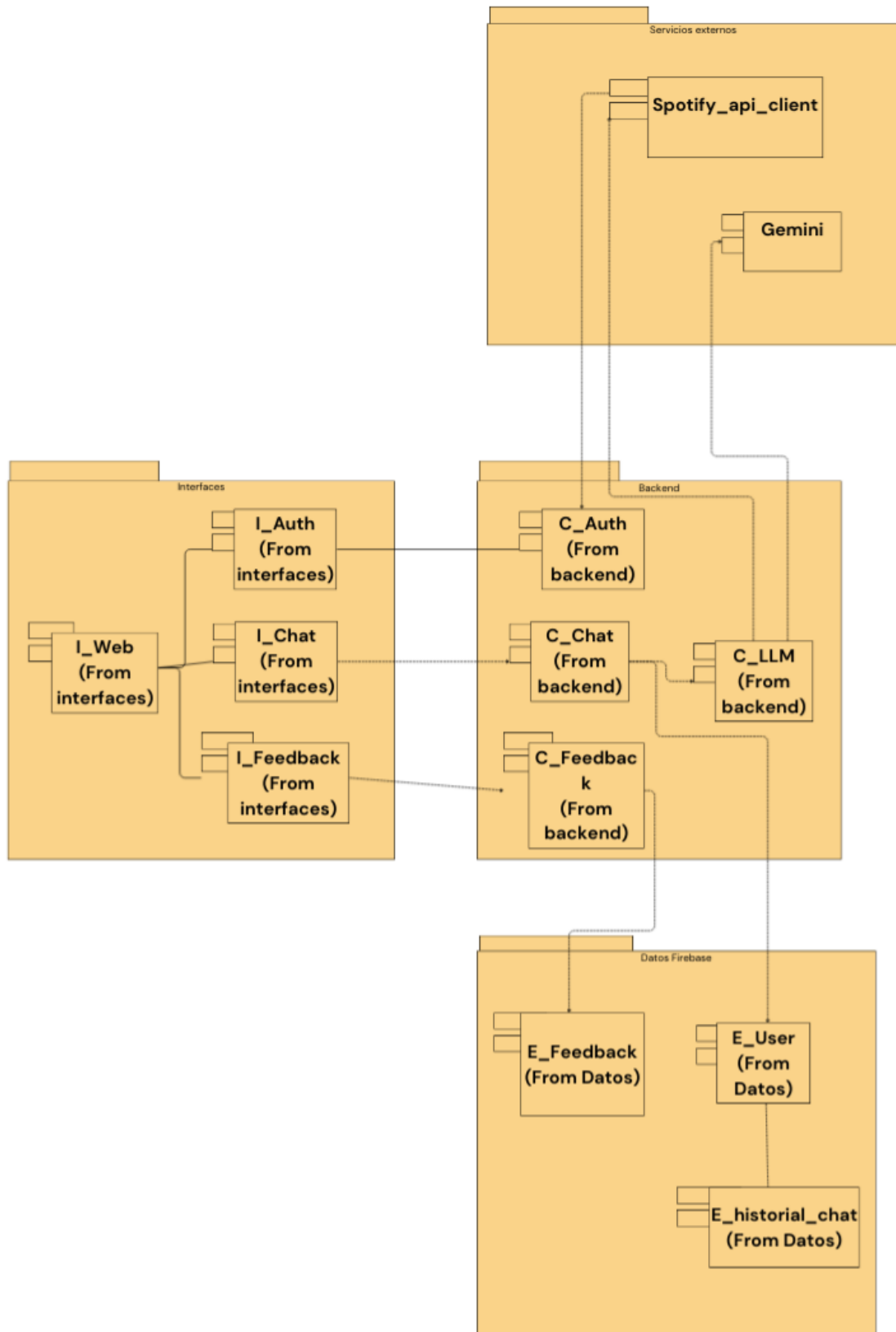


Figura 23. Diagrama de Componentes.
Fuente: Elaboración propia

3.13 Diagrama de despliegue.

A continuación, se presenta el diagrama de despliegue, el cual ilustra los diversos nodos involucrados y sus respectivos componentes en el proyecto. Los elementos principales son:

- **Dispositivo del usuario:** Constituye el dispositivo electrónico con capacidad de navegación *web* del usuario, utilizado para acceder a la página que aloja la interfaz del sistema recomendador.
- **Servidor *frontend*:** Es el responsable de mantener disponible la vista de usuario a nivel web, siendo Vercel la plataforma propietaria.
- **Servidor *backend*:** Este servidor alberga la lógica de negocio en la nube para su acceso desde la interfaz de usuario. Su gestión está a cargo de la plataforma *Render*.
- **Base de datos:** Alojada en la nube de *Firebase*, la base de datos contiene la información recopilada a partir de la interacción del usuario con el proyecto.
- **Nube de *Google*:** Se accede a este recurso para la utilización de los modelos LLM que ofrece.
- **Servidor *API Spotify*:** Servidor al cual el proyecto se conecta para la adquisición de datos de usuario y de contenido musical.

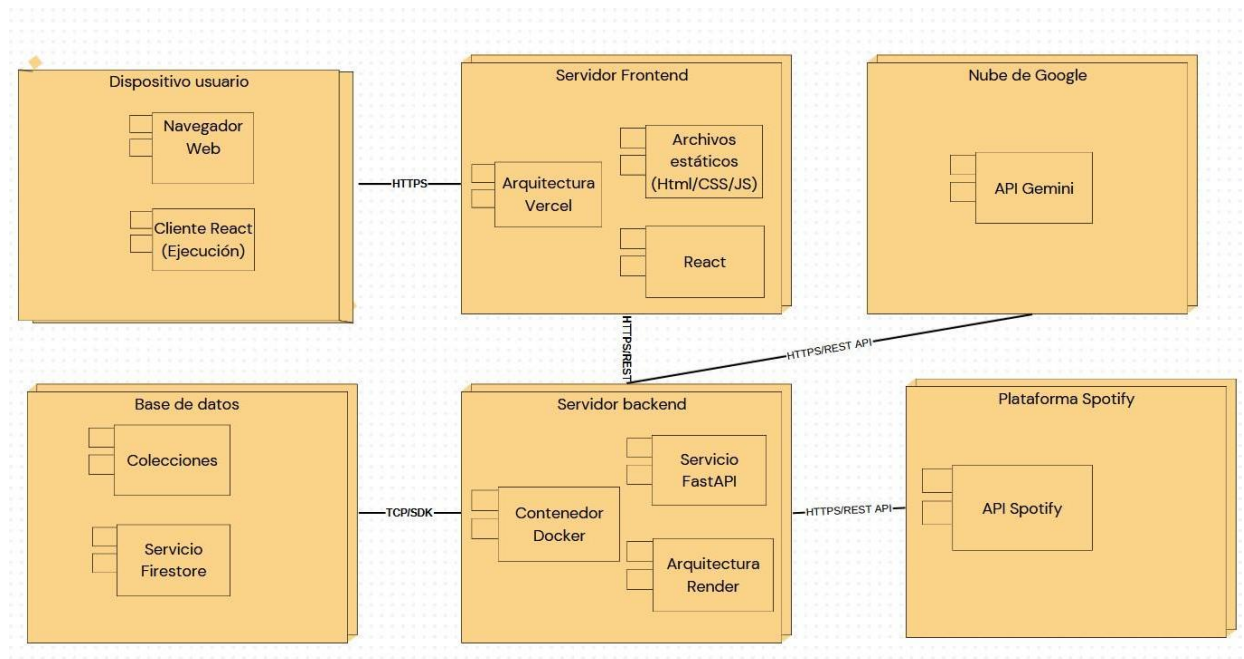


Figura 24. Diagrama de despliegue
Fuente: Elaboración propia

3.14 Concepto inicial de la aplicación web

El propósito de esta sección es ilustrar la estructura general, las funciones principales y el flujo de interacción del recomendador musical. Las ilustraciones fueron elaboradas en la plataforma *web Figma* y no constituyen una demostración funcional; su único objetivo es la representación visual del sistema.

3.14.1 Vista de autorización.

La visualización inicial del sistema se presenta mediante una ventana modal que requiere la conexión a la cuenta de usuario de *Spotify*. El botón correspondiente redirige a la plataforma de *Spotify*, cuya descripción se omite en esta sección por ser ajena al alcance del proyecto.

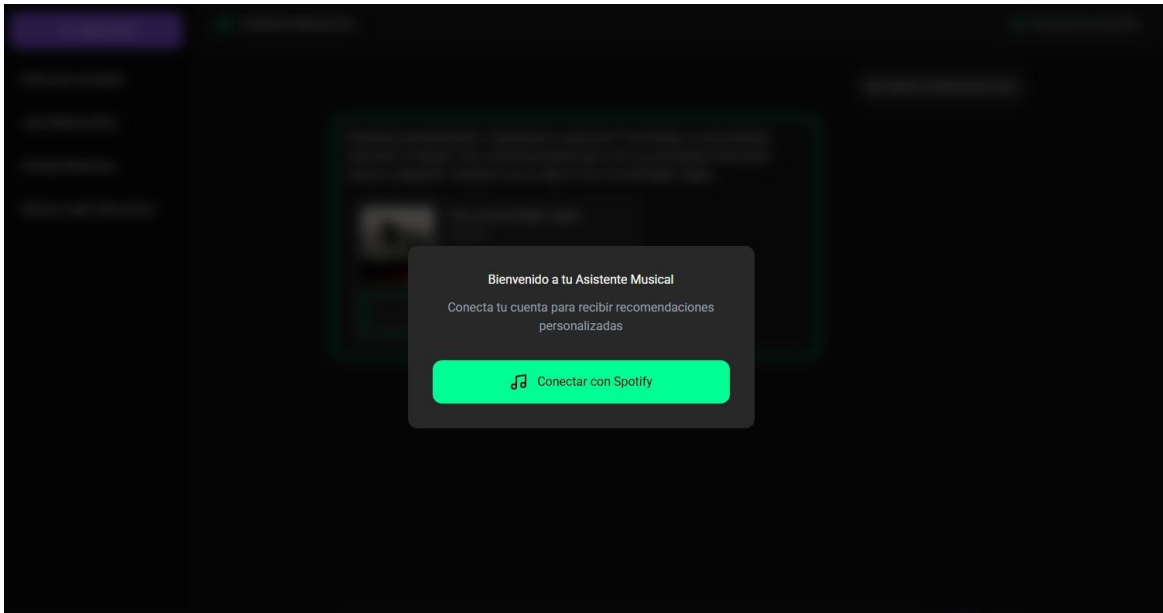


Figura 25. Vista de autorización.
Fuente: Elaboración en Figma.

3.14.2 Vista principal.

Esta vista presenta la sección de mayor valor de la aplicación. Se observa, de izquierda a derecha, la barra lateral que permite la creación de un nuevo *chat* y la gestión de la cuenta de usuario. A continuación, se encuentra la interfaz de chat, donde se ejemplifica la interacción entre un usuario y el asistente musical. En esta demostración, el asistente intenta comprender las preferencias del usuario y recomienda un elemento musical, incluyendo en el mensaje el contexto de su selección y ofreciendo la opción de reproducción en la plataforma *Spotify*. En la esquina inferior derecha se sitúa un botón con forma de estrella, cuya pulsación despliega la opción de *feedback*, la cual será detallada en la siguiente vista.

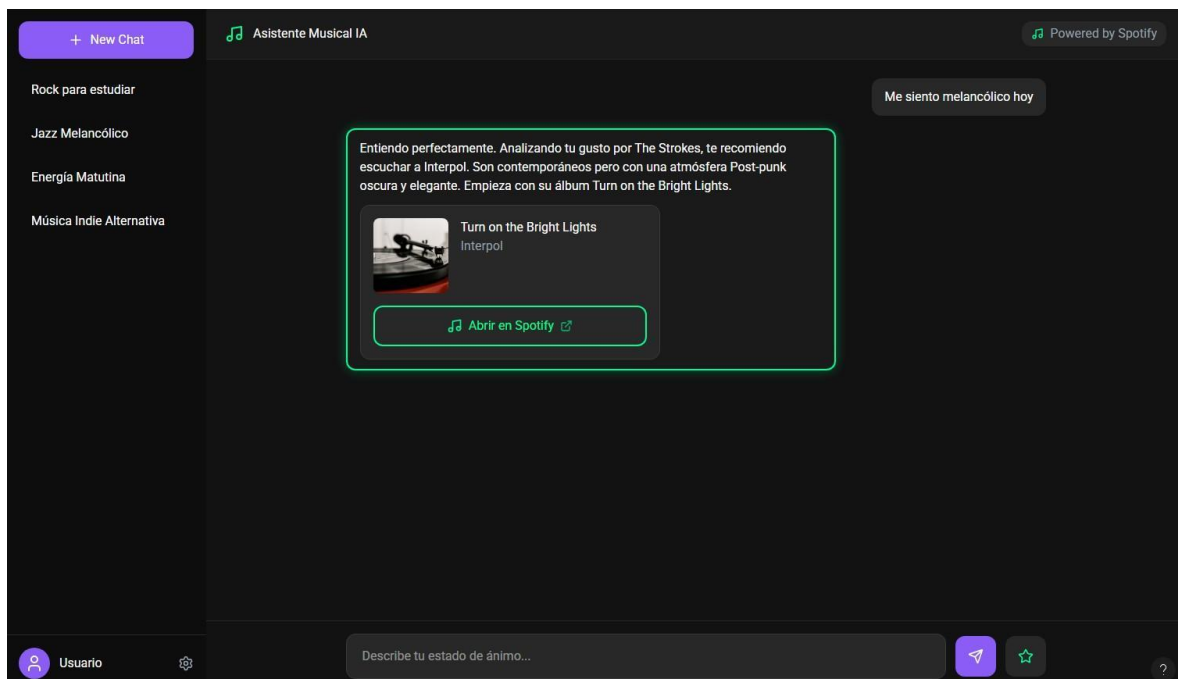


Figura 26. Vista Principal.
Fuente: Elaborada en Figma

3.14.3 Vista de Feedback

Se presenta una ventana modal que constituye la sección de retroalimentación, la cual incluye un sistema de calificación de 1 a 5 estrellas. Adicionalmente, contiene dos preguntas de naturaleza binaria y un campo de comentarios para que el usuario exprese su opinión de forma textual. Ofrece las opciones de "Enviar *Feedback*" y "Cancelar".

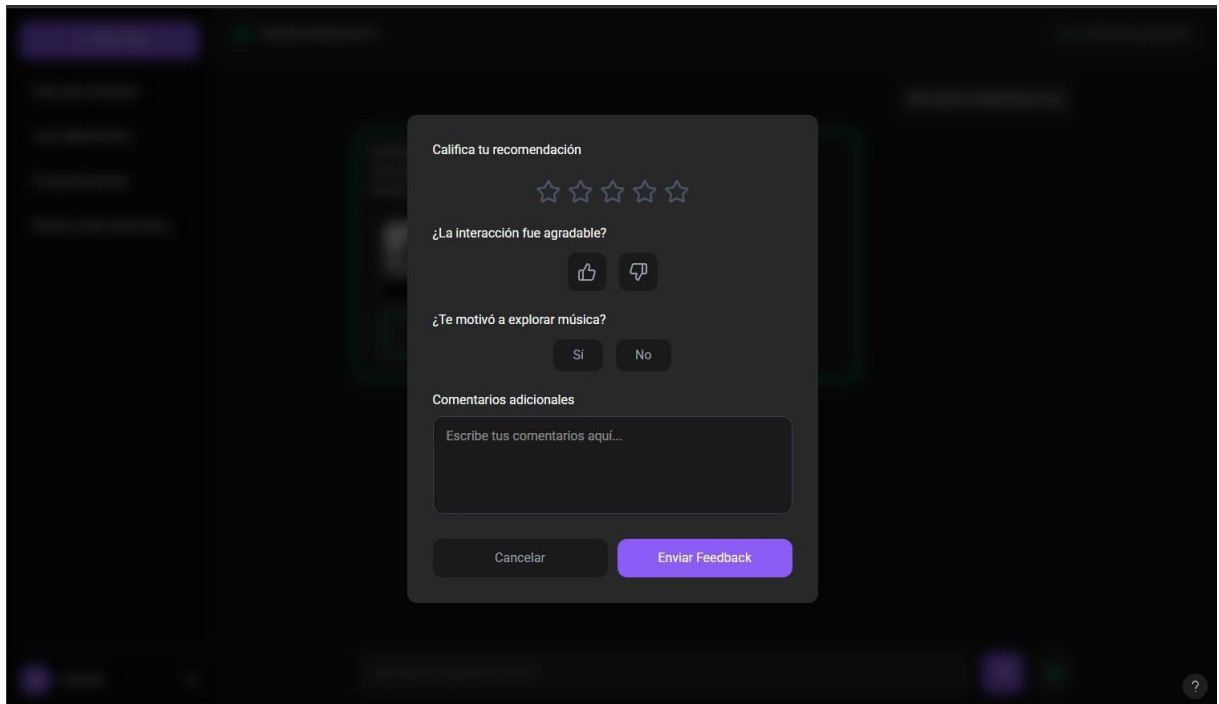


Figura 27. Vista de Feedback.
Fuente: Elaboración en Figma.

3.14.4 Vista Ajustes de usuario

Esta vista presenta una ventana modal de gestión de usuario que incluye dos funcionalidades principales: la eliminación del historial de conversaciones y el cierre de sesión. Adicionalmente, se visualiza una notificación de sincronización exitosa, el nombre de la cuenta de *Spotify* y un botón de cierre para retornar a la vista principal.

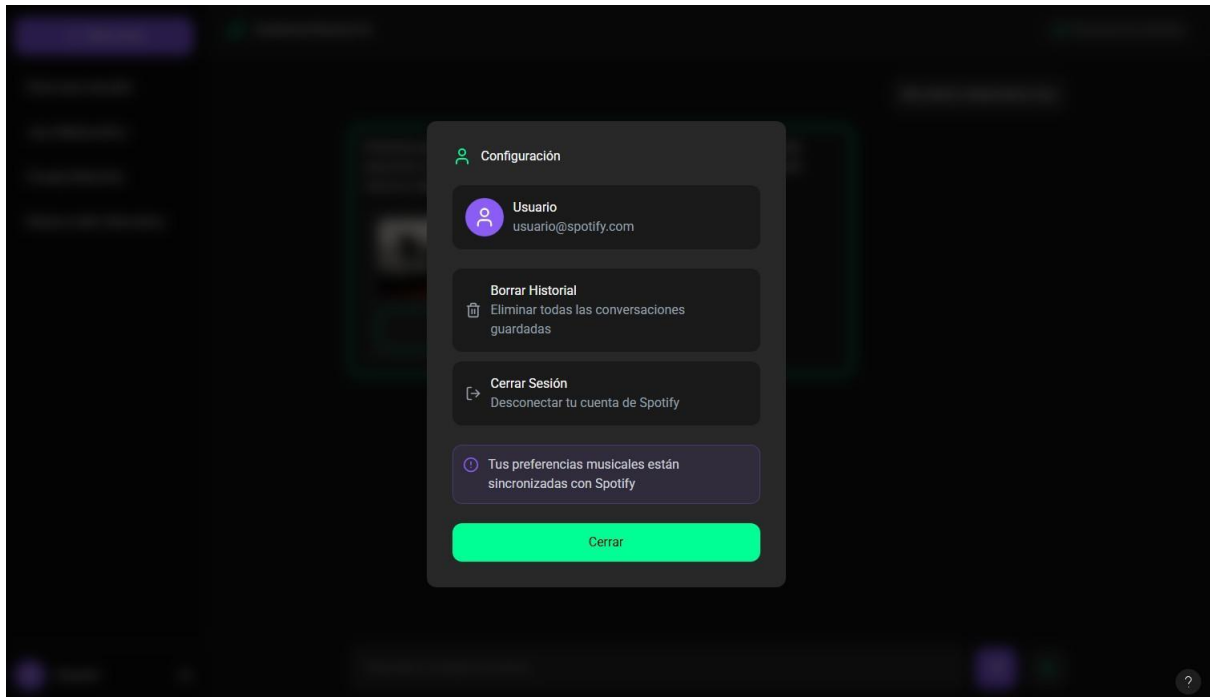


Figura 28. Vista ajustes de usuario.
Fuente: Elaboración en Figma

4. VALIDACIÓN DE LA SOLUCIÓN

4.1. Metodología de validación

Para llevar a cabo la validación de la solución propuesta, se procedió a realizar las pruebas funcionales como parte del sprint 6, de acuerdo con lo estipulado en la gestión del tiempo. Una vez concluidas las pruebas, se inició la convocatoria a usuarios para la evaluación externa de la aplicación. Para este propósito, se efectuó el despliegue web del proyecto, con el *backend* en Render y el *frontend* en Vercel; los usuarios pueden acceder a la plataforma mediante el enlace "songyapper.vercel.app". El título constituye un nombre temporal resultante de la combinación de la palabra en inglés *song*, que se traduce como canción, y *yapper*, un término coloquial inglés utilizado para referirse a alguien que habla en exceso. Con el objetivo de recopilar retroalimentación sobre el estado base de la aplicación, se solicitó a los usuarios su evaluación a través del módulo de *feedback* integrado en la interfaz de usuario.

La implementación de la prueba piloto se realizó mediante la selección de 5 usuarios de prueba cada 24 horas, solicitándoles utilizar la funcionalidad principal de la aplicación, preferentemente con monitoreo en tiempo real, para obtener una retroalimentación más exhaustiva. Hasta la fecha, se ha recolectado la retroalimentación de 20 usuarios. Cada retroalimentación comprende los siguientes elementos:

Evaluación de 1 a 5: Se solicita con la frase "Califica tu recomendación".

Este puntaje tiene como propósito evaluar la calidad de la funcionalidad principal de la aplicación, la recomendación musical, donde el valor mínimo representa la calidad percibida más deficiente y el valor máximo, la calidad más sobresaliente.

Pregunta de selección binaria 1: Se formula la pregunta "¿La interacción fue agradable?" y se ofrecen como posibles opciones de respuesta los iconos de un pulgar arriba y un pulgar abajo, que representan afirmación y negación respectivamente. Esta pregunta busca evaluar la experiencia del usuario como positiva o negativa.

Pregunta de selección binaria 2: Se pregunta "¿Te motivó a explorar música?" y se proporcionan como opciones de respuesta "Sí" y "No". Esta consulta tiene como objetivo confirmar la meta a largo plazo de la aplicación, que es el incentivo a la exploración musical por parte del usuario, lo cual beneficiará el uso de servicios de *streaming* y la democratización de la música a través del descubrimiento de bandas y artistas con menor popularidad.

Caja de texto: Con la descripción "comentarios adicionales", brinda al usuario la oportunidad de expresar, con sus propias palabras, retroalimentación complementaria a las preguntas anteriores, constituyendo el recurso principal para la mejora del sistema.

4.2. Pruebas funcionales

Las siguientes pruebas funcionales permitieron validar el correcto funcionamiento de las funcionalidades básicas que se mencionaron en los casos de uso. Las pruebas fueron las siguientes:

4.2.1. Autorizar acceso a la cuenta

Al ingresar a la página web, automáticamente se presenta el modal de conexión con Spotify.

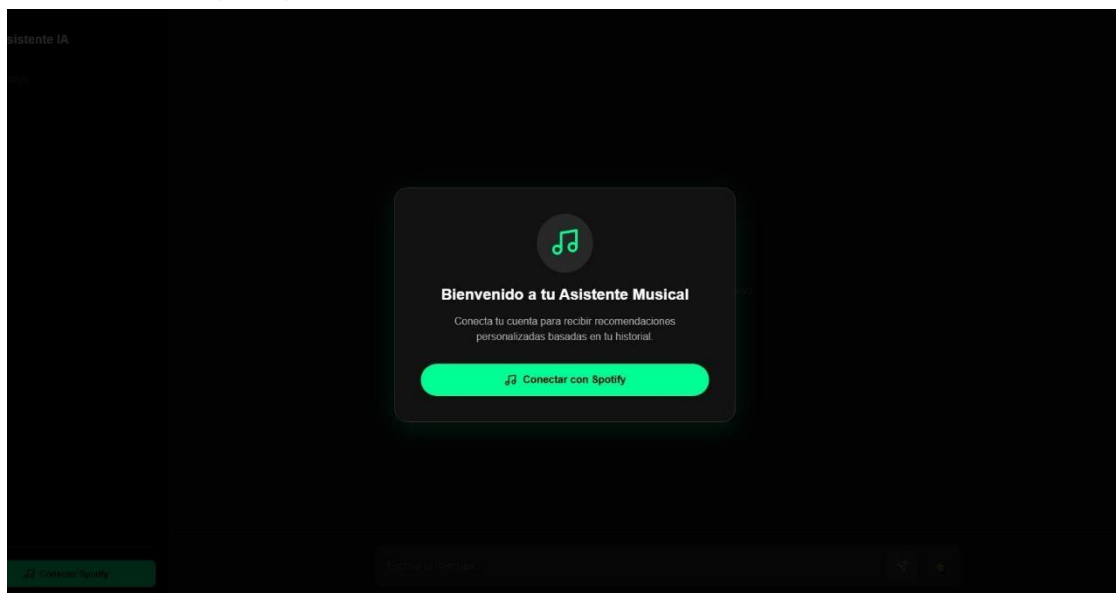


Figura 29. Autorización de acceso a cuenta.
Fuente: songyapper.vercel.app

En ausencia de una autorización previa, o si la conexión con la plataforma

web de Spotify no está activa, la aplicación redirige al sistema de autorización del servicio, cuyo funcionamiento es gestionado por este último.

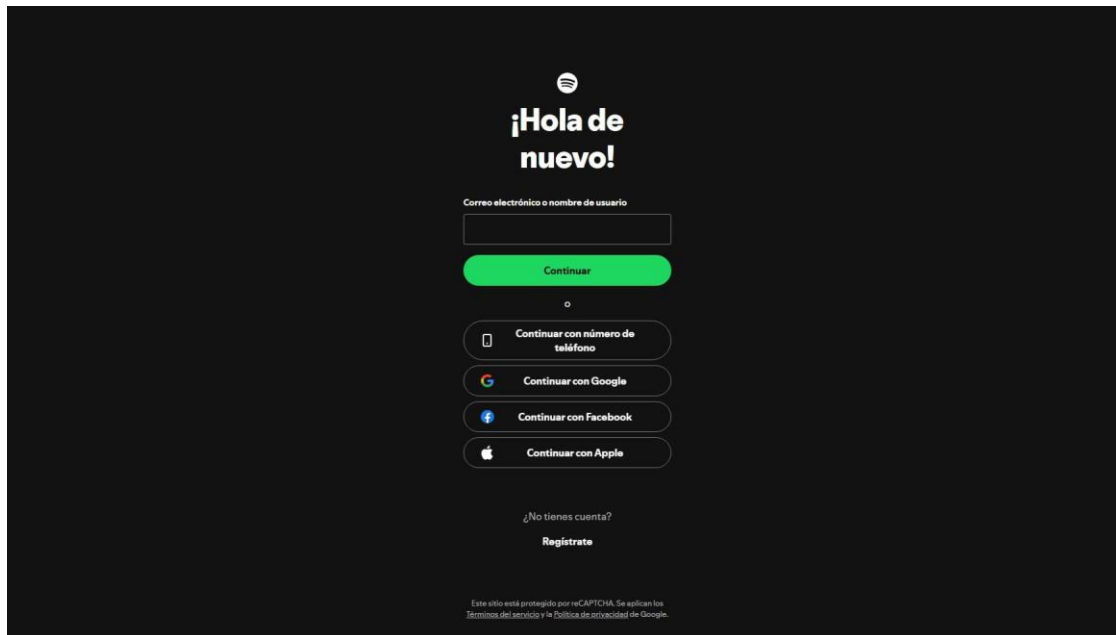


Figura 30. Inicio de sesión en plataforma Spotify.
Fuente: Spotify

De haber otorgado autorización previamente, y teniendo una sesión iniciada en la plataforma web de Spotify, el sistema permite su uso sin solicitar autorización.

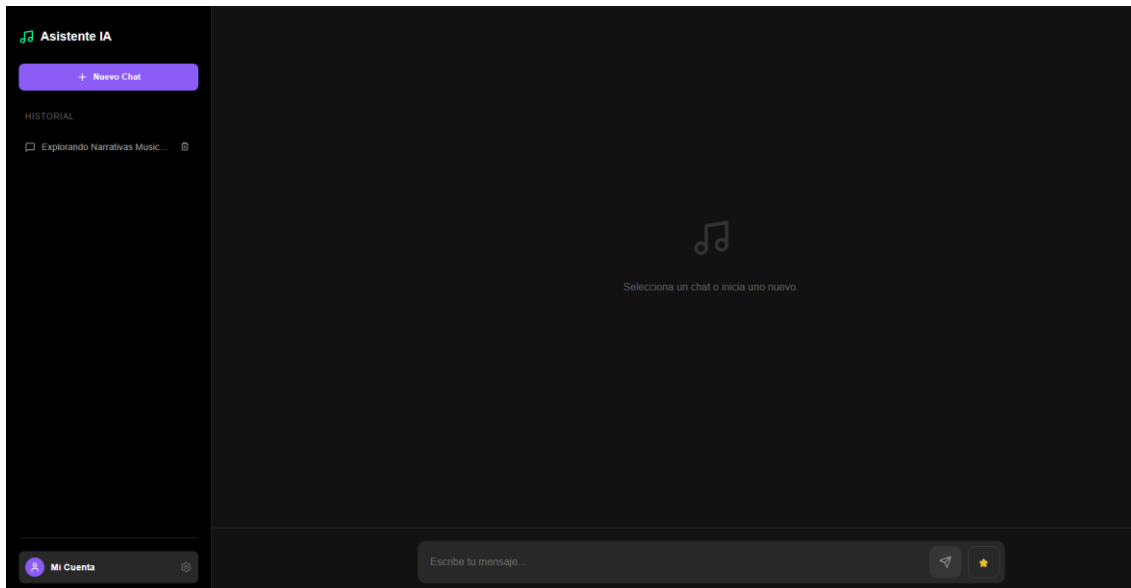


Figura 31. Interfaz de usuario de la aplicación desplegada.
Fuente: songyapper.vercel.app

4.2.2. Escribir petición y recibir recomendación.

Se evaluó la funcionalidad con el mensaje básico proporcionado por el usuario, "Dame una recomendación". La respuesta inicial presentó un *loading skeleton*, y el tiempo de respuesta fue de aproximadamente 12 segundos.

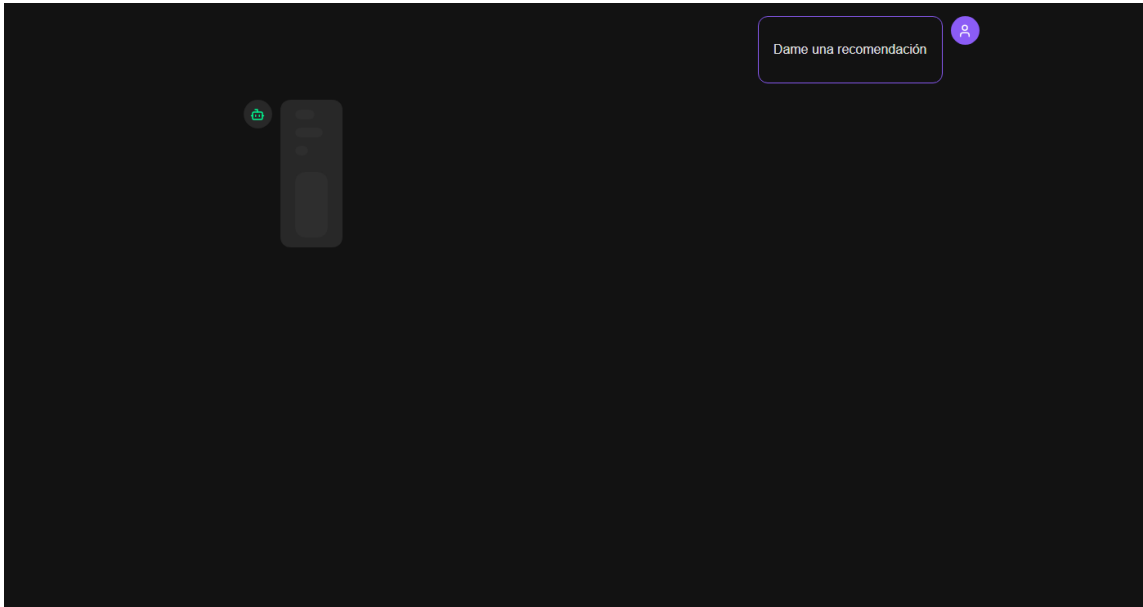


Figura 32. Captura de loading skeleton generado tras escribir petición y esperar recomendación.

Fuente: songyapper.vercel.app

Posteriormente, el sistema presenta la respuesta utilizando el efecto typewriter. Esta se estructura mediante una introducción que confirma la lectura del perfil de usuario de Spotify y anuncia su recomendación, en este caso, el artista conocido como "HEALTH".

A continuación, se incluye un enlace personalizado con la imagen de portada del artista, el tipo de elemento (en este caso, Artista) y su nombre.

A continuación, se presenta un párrafo de mayor extensión que incluye el lugar de origen del artista, los géneros musicales predominantes, una descripción cualitativa de su sonido, una justificación basada en las preferencias detectadas del usuario, y la recomendación de un álbum específico.

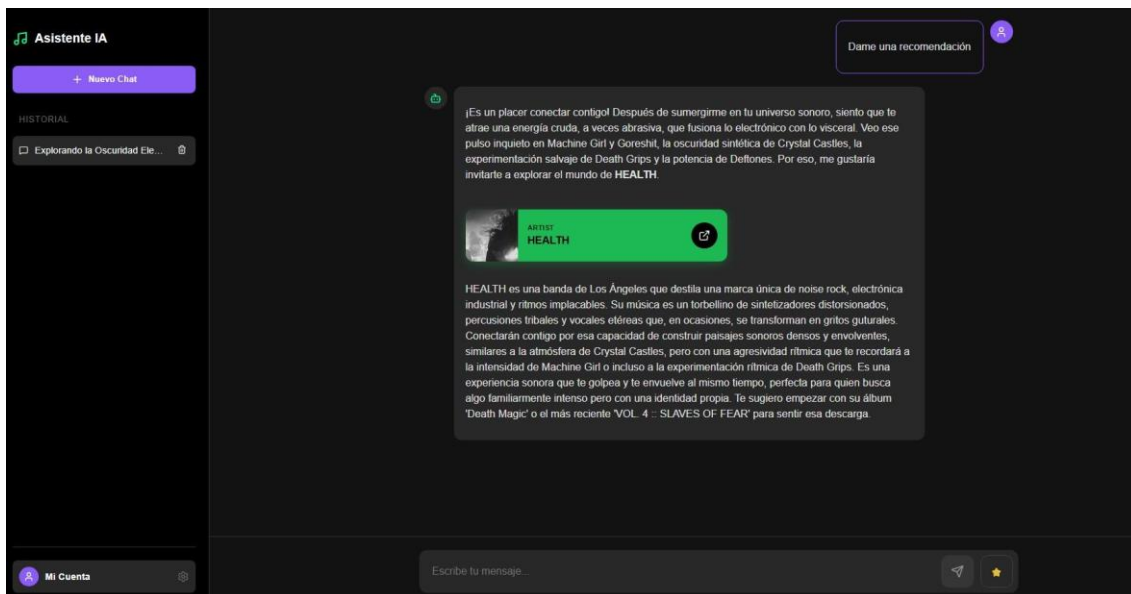


Figura 33. Captura de respuesta generada tras recibir recomendación de parte del sistema.

Fuente: songyapper.vercel.app

El enlace redirige exitosamente a la sección del artista dentro de la plataforma web de Spotify.

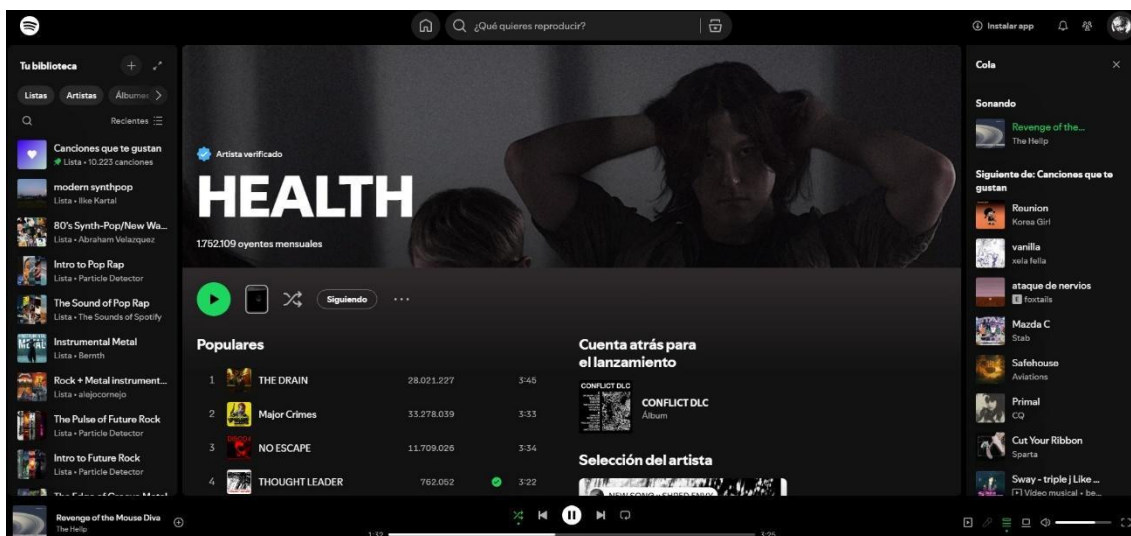


Figura 34. Captura de sección del artista "HEALTH"

Fuente: Spotify.

Se evaluó la diversidad de las respuestas solicitando la misma instrucción en una instancia diferente. Si bien el elemento recomendado coincidió, la descripción generada se redactó de forma distinta; no obstante, mantuvo el mismo formato y una justificación similar.

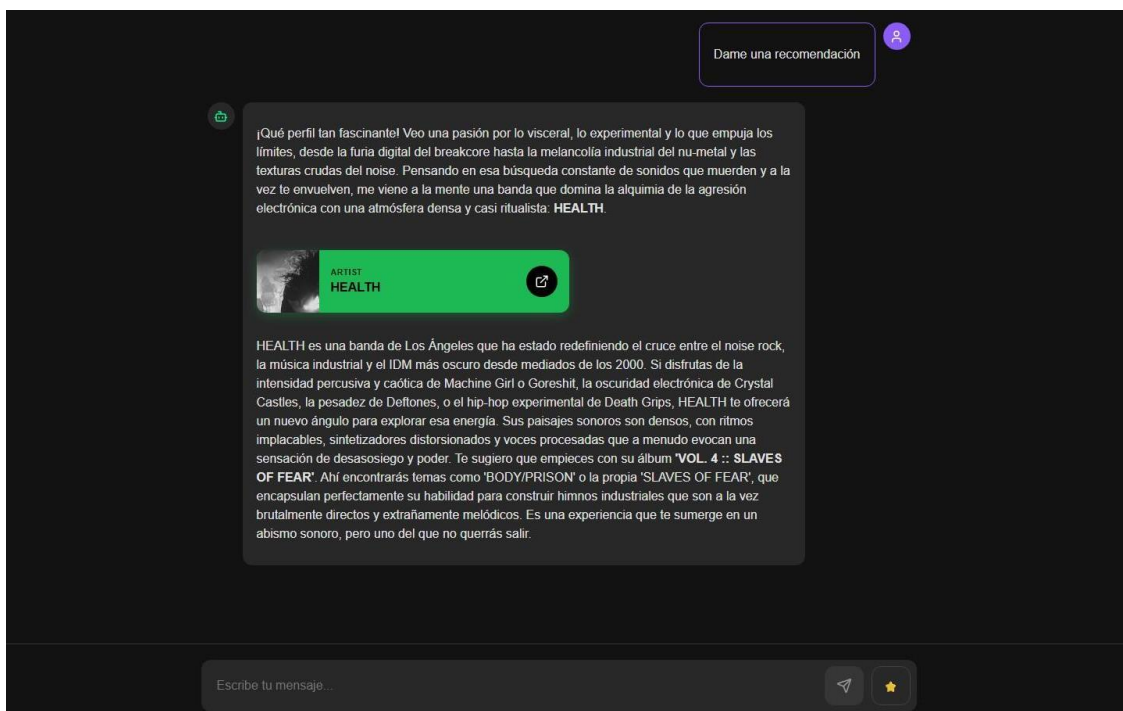


Figura 35. Captura de petición reiterada en un nuevo chat y su respuesta.
Fuente: songyapper.vercel.app

Se evaluó la comprensión de sentimientos. El LLM demostró la capacidad de generar recomendaciones basadas en la expresión de emociones, sin requerir una solicitud explícita relacionada con la música.

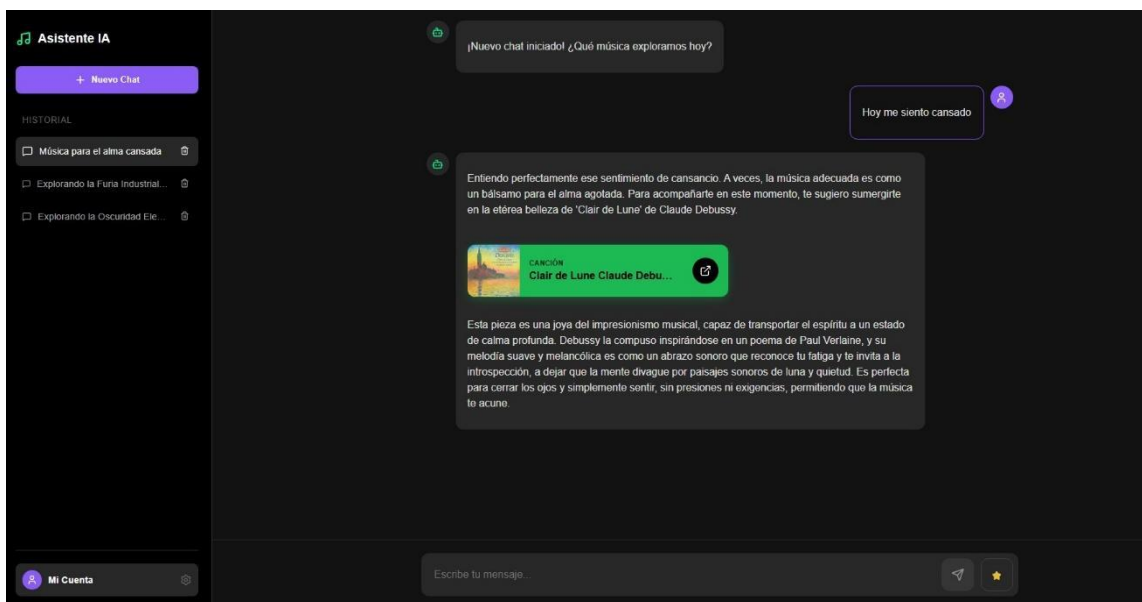


Figura 36. Captura de petición refiriendo a sentimientos y su respuesta.
Fuente: songyapper.vercel.app

Se evaluó la situación de una solicitud no relacionada con música. El sistema de recomendación enfatiza su funcionalidad principal y sugiere una nueva solicitud pertinente al contexto de la conversación.

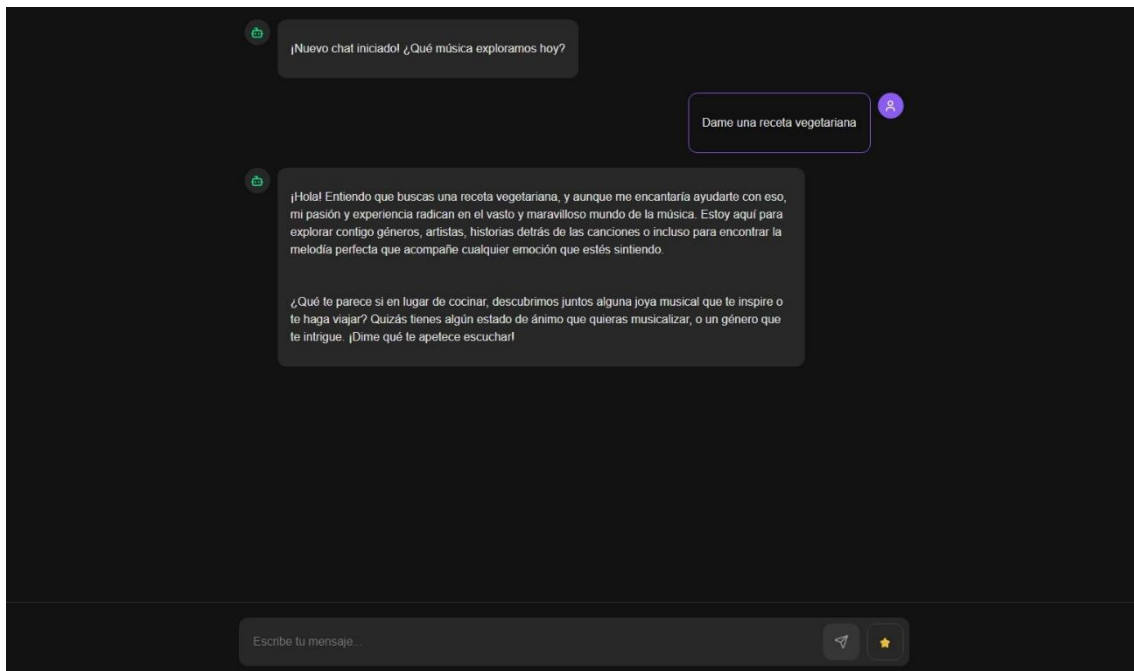


Figura 37. Captura de petición no relacionada a la función principal y su respuesta.
Fuente: songyapper.vercel.app

Se evaluó la decisión del modelo entre proporcionar una respuesta con alucinación o declarar desconocimiento. El usuario hizo referencia a una banda musical ficticia, ante lo cual el sistema recomendador manifestó no tener conocimiento y procuró reconducir la conversación hacia un ámbito conocido.

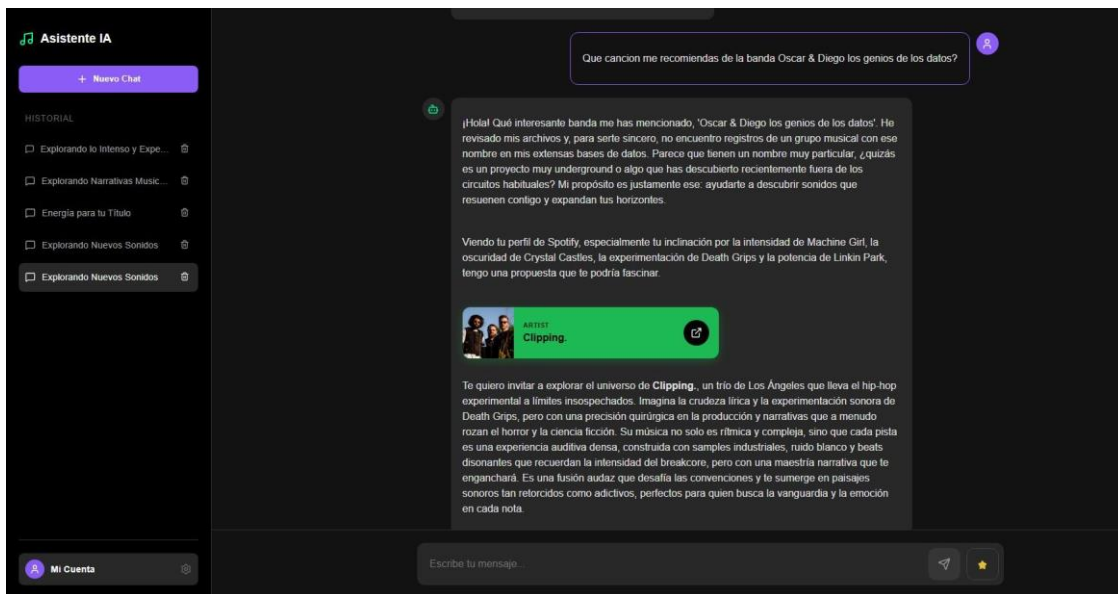


Figura 38. Captura de petición con información inventada y su respuesta.
Fuente: songyapper.vercel.app

4.2.3. Continuar el chat.

Se evaluó la continuidad de la conversación del chat, observándose un tiempo de espera aproximado de 9 segundos. La respuesta obtenida demostró mantener la coherencia del hilo conversacional.

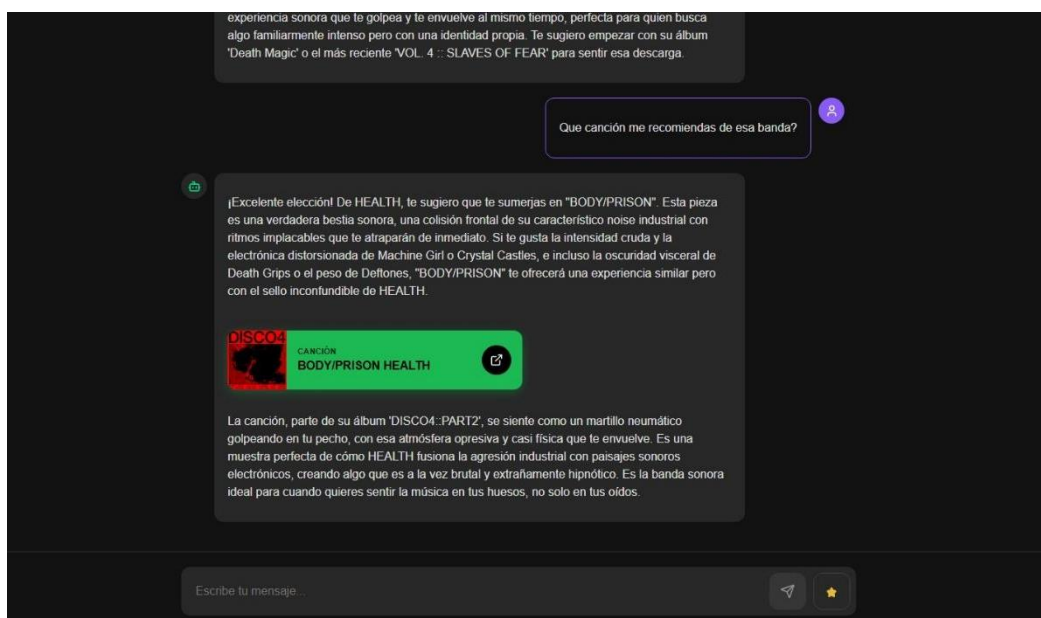


Figura 39. Captura de continuación de conversación.
Fuente: songyapper.vercel.app

Se puso a prueba la conservación del hilo de la conversación una vez más, con una respuesta de aproximadamente 13 segundos, el sistema demuestra mantener una conversación con memoria.

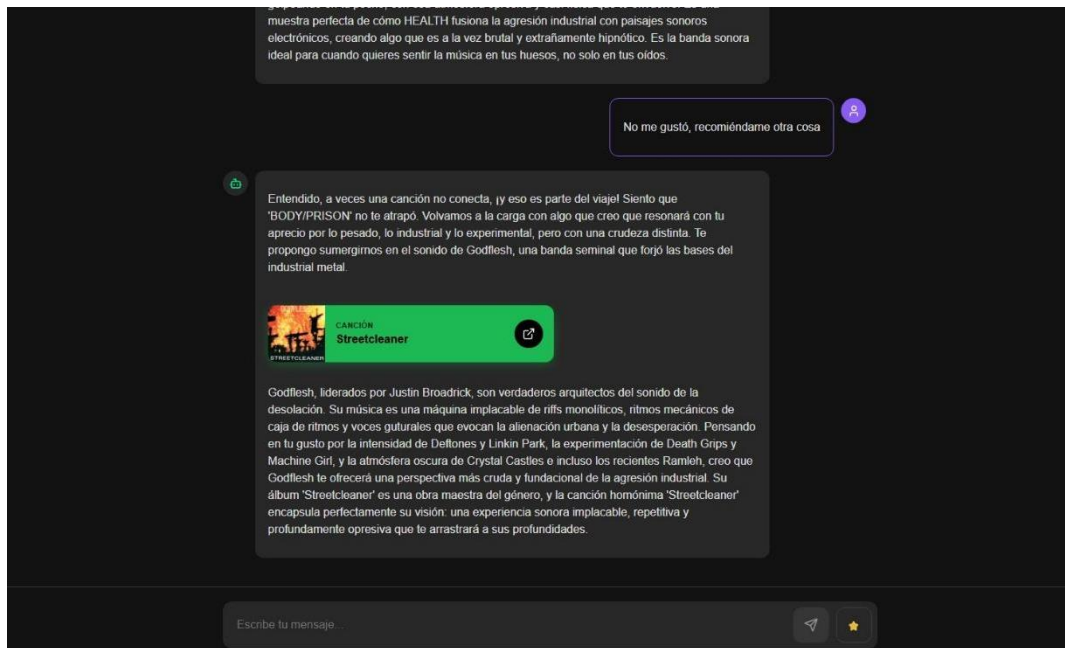


Figura 40. Captura de continuación de conversación con tercer mensaje.
Fuente: songyapper.vercel.app

4.2.4. Comenzar nuevo chat

Se evaluó el uso de la funcionalidad de iniciar nuevo chat, la interfaz no muestra rastros de conversaciones anteriores.

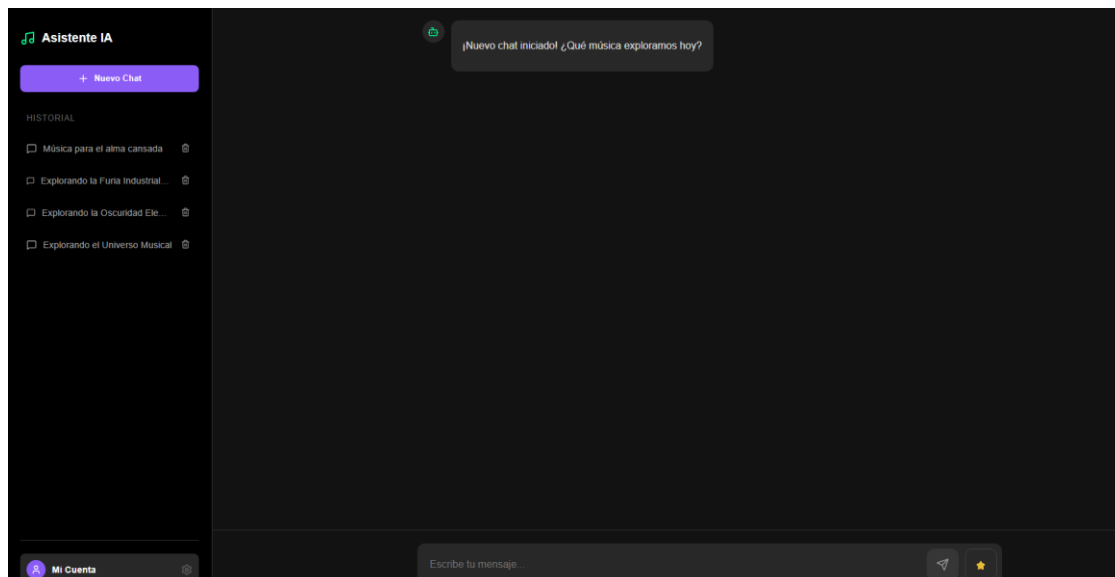


Figura 41. Captura de nuevo chat.
Fuente: songyapper.vercel.app

Se evidencia el registro de diversas conversaciones en la barra lateral; no obstante, en la nueva interacción, el sistema de recomendación declara ser

su primer contacto conversacional con el usuario, lo que establece que no existen sesgos derivados de interacciones previas.

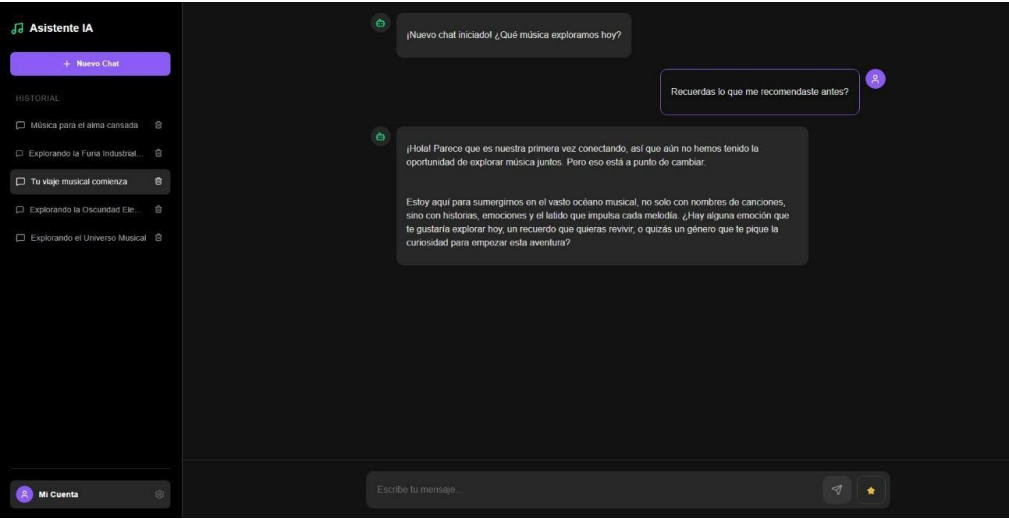


Figura 42. Captura de petición respecto a conversación en otro chat y su respuesta.
Fuente: songyapper.vercel.app

4.2.5. Enviar retroalimentación.

Se evaluó la aparición del modal de retroalimentación al momento de seleccionar la opción de feedback. Todos los elementos responden adecuadamente a la interacción del usuario.

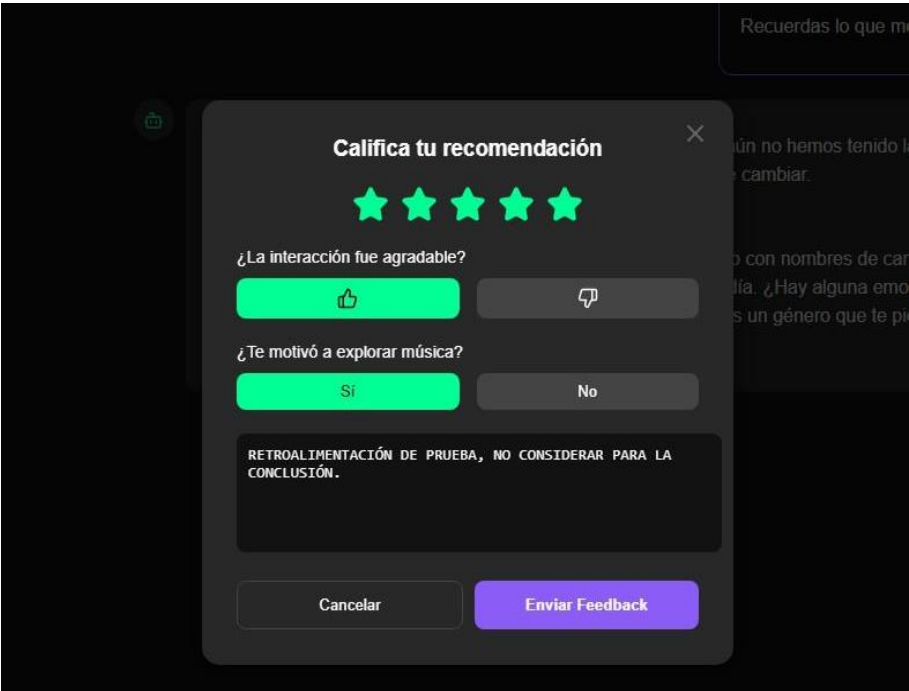


Figura 43. Modal de retroalimentación.
Fuente: songyapper.vercel.app

Tras enviar la retroalimentación aparece una notificación de agradecimiento.

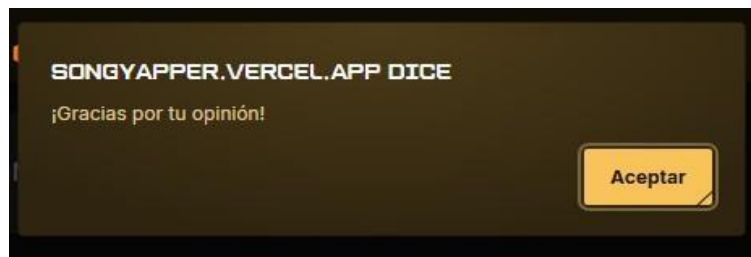


Figura 44. Notificación de agradecimiento.
Fuente: songyapper.vercel.app

El registro de retroalimentación se guardó exitosamente dentro de la base de datos de Firebase.

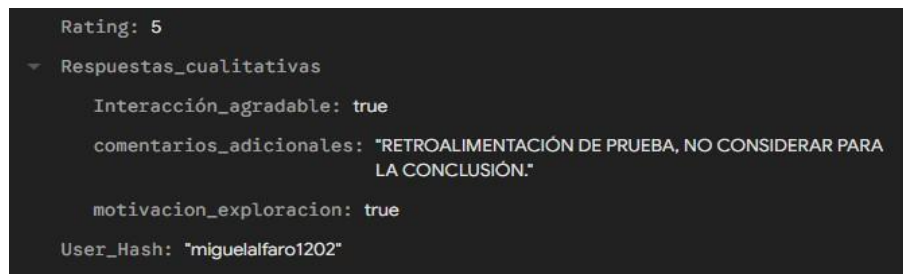


Figura 45. Captura de registro de retroalimentación en base de datos.
Fuente: Base de datos cloud firestore privada.

4.2.6. Borrar conversación

Se puso a prueba la funcionalidad de borrar conversación de la base de datos. En este caso se demuestra con el chat titulado "Tu viaje musical comienza"

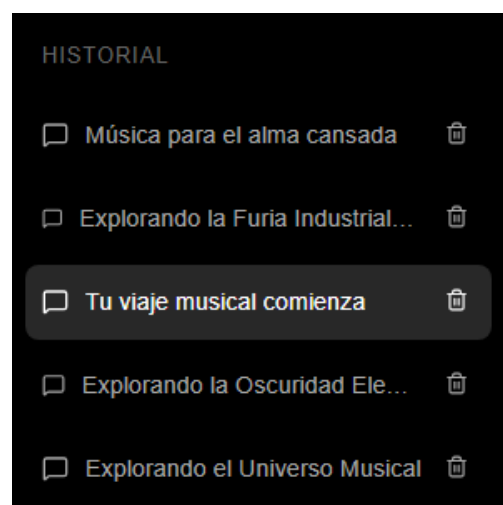


Figura 46. Captura de historial de chats de usuario.
Fuente: songyapper.vercel.app

Una vez presionado el ícono de papelera situado al costado, se despliega la notificación de confirmación de la acción.

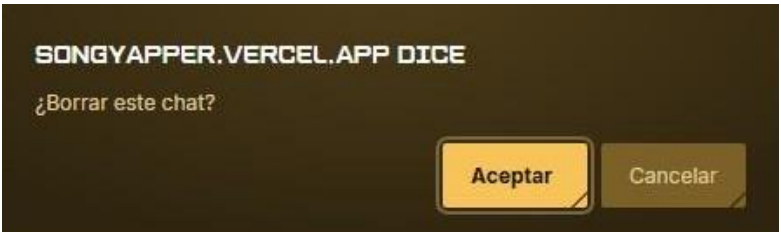


Figura 47. Notificación de confirmación de acción.
Fuente: songyapper.vercel.app

La conservación antes mencionada desaparece de la lista de *chats* del usuario.

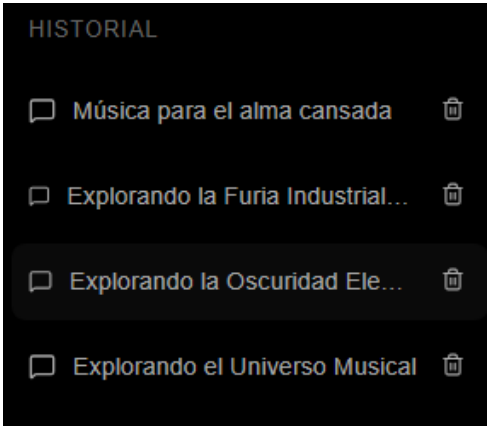


Figura 48. Captura de historial de chat sin el chat borrado.
Fuente: songyapper.vercel.app

Dentro de firebase aparecen sólo cuatro conversaciones, ninguna tiene como resumen “Tu viaje musical comienza”

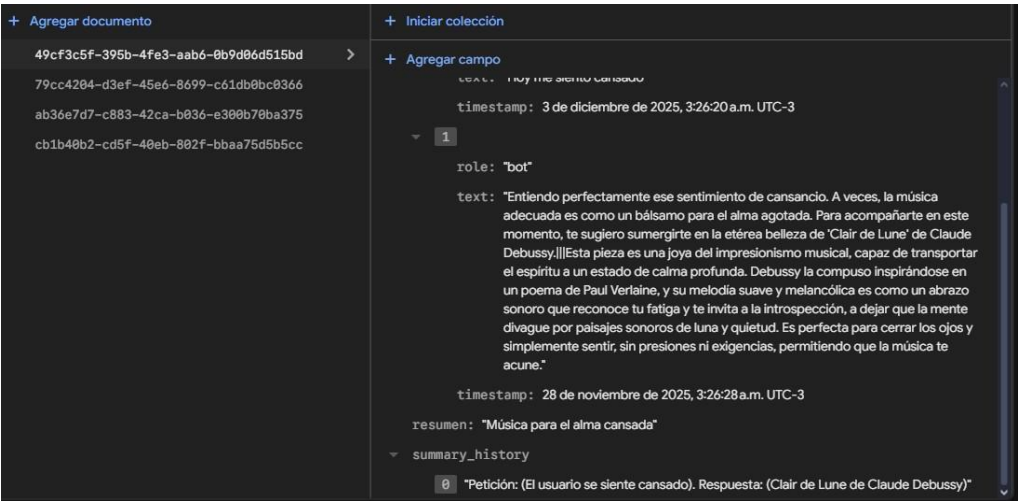


Figura 49. Conversación 49c en firestore
Fuente: Base de datos cloud firestore privada.

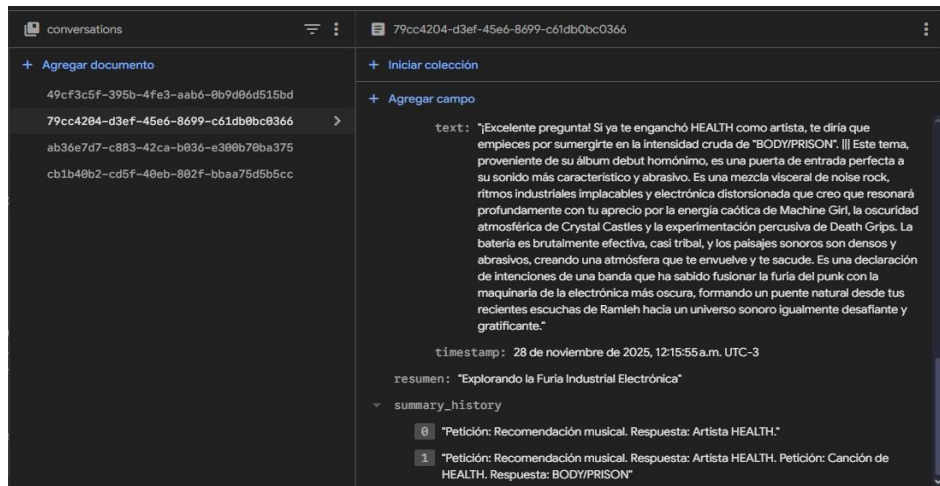


Figura 50. Conversación 79c en firestore
Fuente: Base de datos cloud firestore privada.

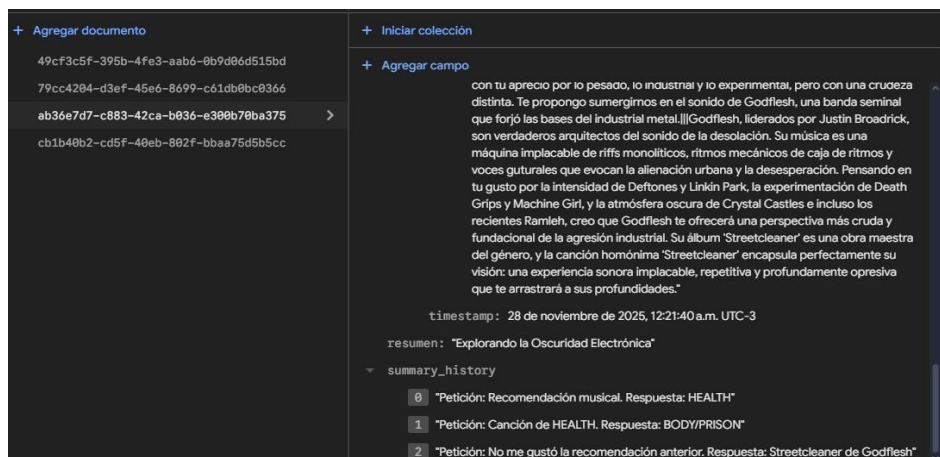


Figura 51. Conversación ab3 en firestore
Fuente: Base de datos cloud firestore privada.

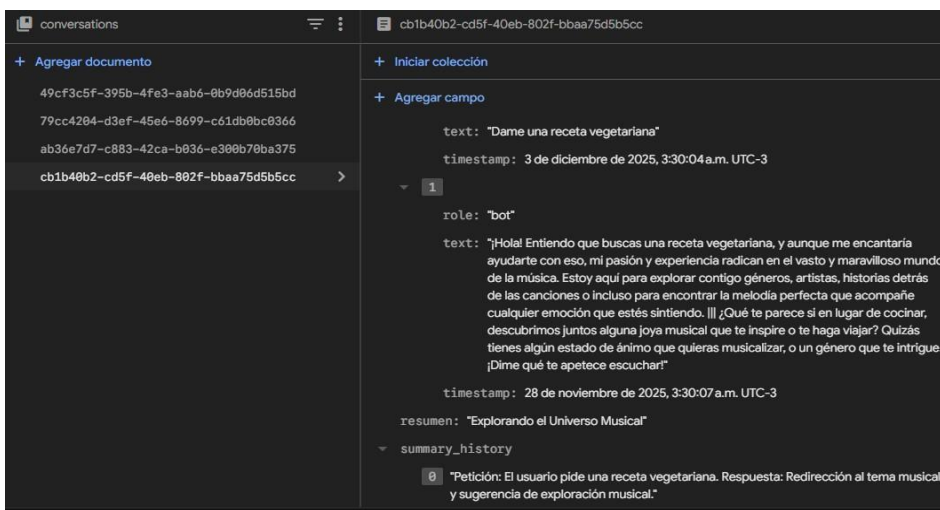


Figura 52. Conversación cb1 en firestore
Fuente: Base de datos cloud firestore privada.

Como alternativa, se evaluó la utilización de la opción de borrado de historial en el módulo de configuración, la cual posibilita eliminar la totalidad de los chats con una única acción.

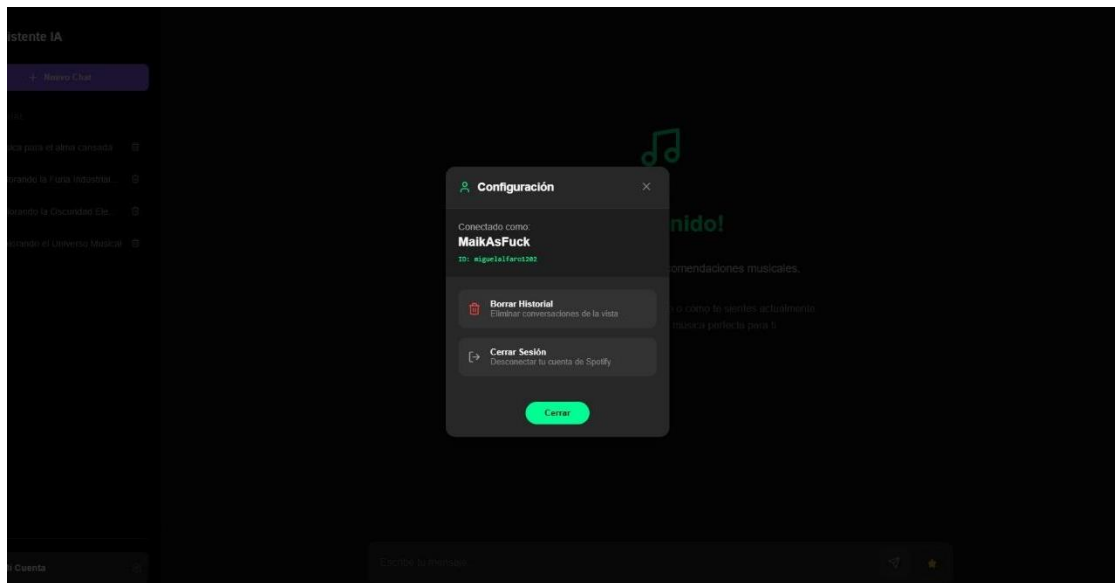


Figura 53. Modal de configuración.
Fuente: songyapper.vercel.app

Al presionar la opción, se despliega la notificación de confirmación de la acción.

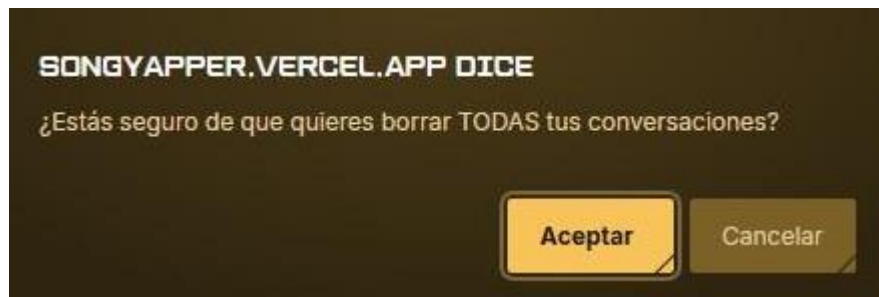


Figura 54. Notificación de confirmación de acción de borrado.
Fuente: songyapper.vercel.app

Tras el lapso de 2 segundos, la acción se lleva a cabo y todo el historial es borrado.

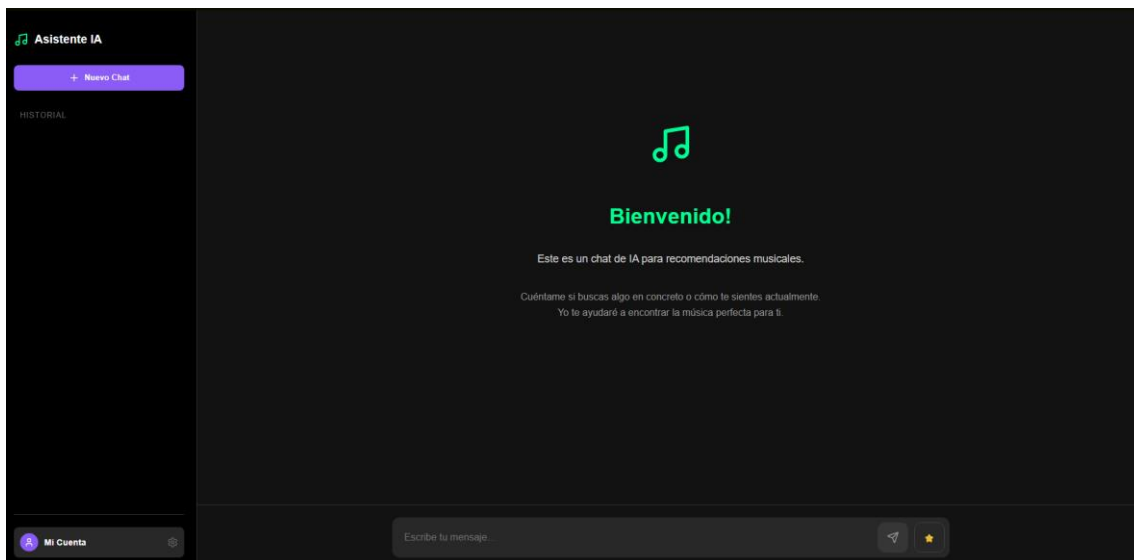


Figura 55. Captura de interfaz de usuario con chats eliminados.
Fuente: songyapper.vercel.app

Se confirma dentro de la base de datos la inexistencia de los chats previamente vistos.

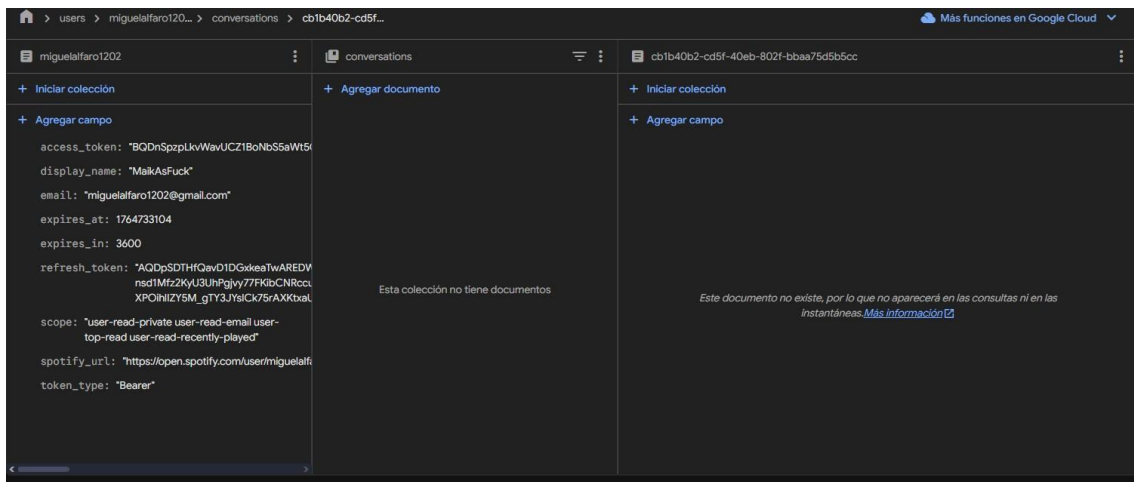


Figura 56. Captura de conversaciones de usuario en Firestore tras eliminar el historial.
Fuente: Bases de datos cloud Firestore privada.

4.2.7. Cerrar sesión

Se puso a prueba la funcionalidad de cerrar sesión de usuario en el modal de ajustes.

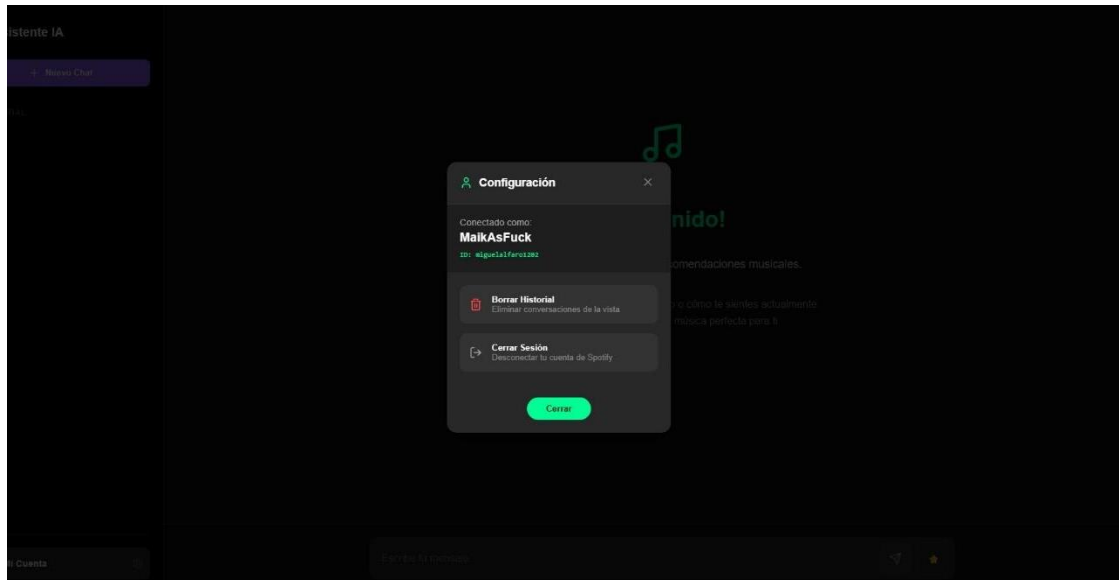


Figura 57. Modal de configuración.
Fuente: songyapper.vercel.app

Una vez presionado, la desconexión con la cuenta es llevada a cabo.

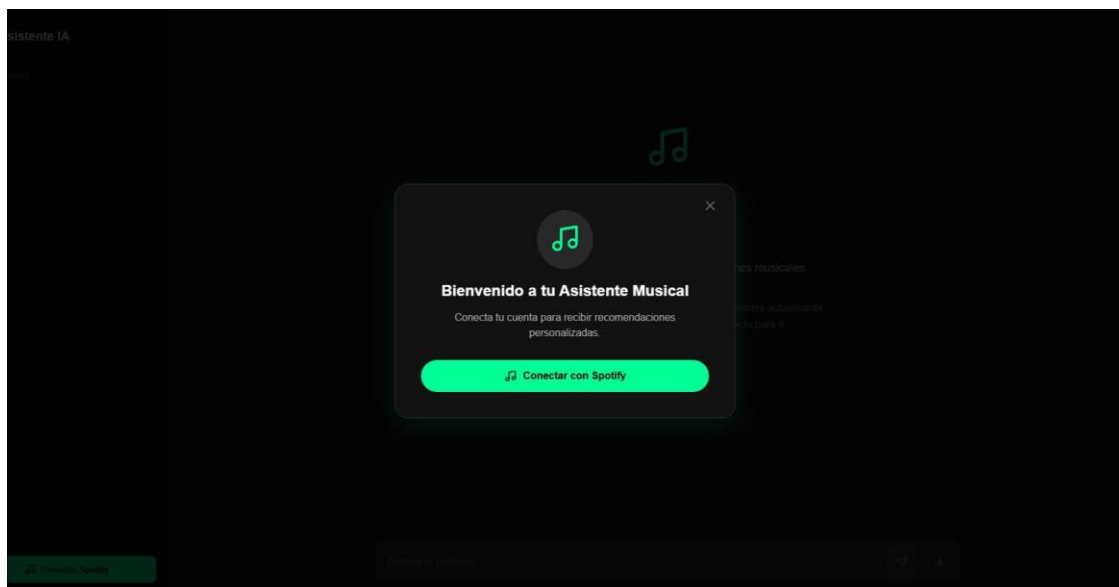


Figura 58. Autorización de acceso a cuenta tras cierre de sesión.
Fuente: songyapper.vercel.app

4.2.8. Rendimiento y tiempos de respuesta.

A continuación, se detalla el registro de la medición del tiempo promedio de respuesta de las operaciones durante las pruebas funcionales. El tiempo de respuesta se midió de forma manual mediante un cronómetro, abarcando desde el inicio de la funcionalidad hasta su conclusión. Se realizaron 5 intentos por cada funcionalidad, obteniéndose el tiempo promedio de respuesta. La letra O representa la palabra "operación".

Tabla 10. Tabla de registro de medición de tiempo de respuesta.

	O.1	O.2	O.3	O.4	O.5	Promedio
Autorización de acceso	13.57s	9.04s	10.35s	9.19s	7.90s	10.01s
recibir recomendación	27.56s	33.98s	35.64s	31.41s	36.55s	33.028s
Chat de 3 peticiones	1 m 36.90s	1 m 34.45s	1m 38.66s	1m 37.55s	1m 40.01s	1m 37.514s
Comenzar nuevo chat	0s	0s	0s	0s	0s	0s
Enviar feedback	3.18s	1.52s	1.80s	1.12s	1.05s	8.67s
Borrar chat individual	1.58s	2.98s	1.88s	1.99s	1.74s	2.034s
Borrar historial completo	3.09s	2.54s	1.43s	2.28s	1.88s	2.244s
Cerrar sesión	0s	0s	0s	0s	0s	0s

Fuente: Elaboración propia.

4.2.9. Usabilidad básica

Las pruebas realizadas demostraron que la usabilidad básica, como la funcionalidad de los botones, cumple con los requisitos establecidos.

Todas las pruebas unitarias de las funciones principales fueron ejecutadas exitosamente, con un desempeño correcto. Estas pruebas se llevaron a cabo internamente en la versión desplegada en la nube y no se reportaron errores de ejecución.

4.3. Evaluación de usuarios en prueba piloto.

Se ha conseguido la participación de 20 usuarios de prueba que utilizaron la aplicación web, de los cuales 8 accedieron a ser monitoreados. Todos ellos hicieron uso de la funcionalidad principal y proporcionaron retroalimentación tanto de forma verbal como escrita a través del módulo de feedback. Los resultados se declaran a continuación y los comentarios realizados se han empleado para la mejora continua del sistema.

4.3.1. Evaluación de estrellas.

5 estrellas: 12

4 estrellas: 7

3 estrellas: 0

2 estrellas: 1

1 estrella: 0

Promedio de estrellas: 4.5

4.3.2. Evaluación de interacción

Pulgar arriba: 20

Pulgar abajo: 0

4.3.3. Motivación a explorar

Si: 19

No: 1

4.3.4. Comentarios adicionales.

La sección de comentarios arrojó diversas respuestas, tal como se había anticipado. Se mencionaron tanto la precisión de la recomendación emitida como ciertas deficiencias. Estas últimas se atribuyeron principalmente a errores del sistema de recomendación al interpretar la instrucción de generar una sugerencia independiente de las preferencias del usuario.

Adicionalmente, se identificaron fallos en la retención de la conversación previa, lo que resultó en imprecisión al ser consultado sobre una recomendación realizada en mensajes anteriores.

Uno de los aspectos más valorados del sistema fue su estructura. Varios usuarios encontraron satisfactoria la descripción que justifica la recomendación mediante diversos parámetros. Otros comentarios señalaron que la extensión del texto podría ser excesiva, una percepción que se mitigó posteriormente al incorporar el efecto visual de "typewriter".

Durante las pruebas, se presentó un error con el enlace a Spotify: la recomendación correspondía a un álbum del artista Chystemc, pero el enlace redirigía a un álbum de Luis Fonsi. La causa de este error se determinó al verificar que el álbum de Chystemc no estaba disponible en Spotify, por lo que la búsqueda en la API del servicio arrojó como resultado más cercano al segundo artista. Otros comentarios señalaron a problemas de responsividad que se ajustaron durante la fase de pruebas.

Finalmente, se registró la sugerencia recurrente de incluir la reproducción de una vista previa de la música recomendada. Esto mejoraría la experiencia al evitar la necesidad de salir de la página web para reproducir el elemento en la plataforma Spotify.

El error fue corregido mediante la implementación de una política más estricta en los resultados de búsqueda, estableciendo que, si no se encuentra la coincidencia exacta de la petición, simplemente no se proporcionará un enlace. La sugerencia de la vista previa será considerada para una futura iteración. Los problemas de memoria e imprecisión ante instrucciones contrarias al prompt base serán evaluados para encontrar una solución que no comprometa la funcionalidad esencial del sistema.

5. CONCLUSIONES

El desarrollo del presente trabajo de titulación ha introducido una alternativa novedosa para la integración de Modelos de Lenguaje Grande con servicios de streaming musical. Esta integración aporta valor al enriquecer la experiencia de descubrimiento musical a través de una conversación narrativa que detalla diversos aspectos de interés del elemento recomendado. La solución logró su objetivo al desarrollar un sistema capaz de interpretar solicitudes en lenguaje natural y generar recomendaciones basadas tanto en estas peticiones como en las preferencias musicales del usuario. El formato de entrega promueve la exploración musical gracias a una redacción elaborada, un aspecto no explorado en soluciones previas.

El principal referente de esta solución, el *DJ AI* de *Spotify*, también denominado "DJ Livi" en Latinoamérica, demostró el potencial de los LLM en el servicio de streaming de música, si bien presentó limitaciones a superar. La rapidez en la generación de listas de reproducción se veía contrarrestada por la escasa explotación de la capacidad conversacional de los modelos de inteligencia artificial. Este proyecto identificó dicha limitación como una oportunidad de complemento, estableciendo un vínculo más significativo con el usuario al proporcionar un único elemento reforzado con una justificación narrativa que ofrece conocimiento suficiente para que el usuario inicie su propia exploración, sin depender de la opacidad algorítmica. Esto se corroboró en los resultados de la prueba piloto, donde, contando con 20 usuarios, el 95% de ellos concluyó que este proyecto les motivó a explorar nueva música.

La arquitectura empleada demostró ser suficientemente eficiente y viable económicamente para una fase demostrativa. Los costos se mantuvieron en cero gracias a dicha arquitectura y a una gestión y control de recursos adecuados. El panorama sugiere que el proyecto es viable, en términos de arquitectura, para un despliegue de uso público. Sin embargo, esto requerirá una inversión significativa para establecer un presupuesto que permita financiar una plataforma en la nube con capacidad para gestionar un volumen mayor de solicitudes y el uso de modelos LLM con mejor capacidad de cómputo y velocidad.

No obstante, las pruebas piloto han evidenciado debilidades que deben abordarse en relación con el uso de los Modelos LLM. La formulación de un *prompt* base que equilibre la optimización de *tokens*, las restricciones del negocio y las libertades del usuario constituye un área de mejora continua,

donde el desequilibrio de alguno de estos puntos es un riesgo frecuente.

Para una futura iteración, se contempla la incorporación de una vista previa de audio que acompañe la recomendación proporcionada por el sistema, con el fin de optimizar la experiencia del usuario. De igual modo, se profundizará en el refinamiento del *prompt* utilizado por el sistema para obtener resultados óptimos.

Este trabajo ha sentado una base sólida para un sistema cuyo valor reside en la recomendación musical, acompañada de una justificación pertinente que impulse el descubrimiento, todo ello facilitado por la comprensión y generación del lenguaje natural. Se ofrece una nueva vía para que el usuario perciba la pasión inherente al descubrimiento musical, oculta en la conversación y la narración.

5.1. Estado de objetivo principal.

Citando el objetivo establecido en el Capítulo 1, el cual indica: "Este trabajo de título tiene como objetivo desarrollar un sistema de recomendación musical a través del procesamiento de lenguaje natural, interpretando las solicitudes textuales de los usuarios mediante una interfaz de chat y utilizando datos de historial de uso e información global de música disponibles en una API de *streaming* de música, para entregar sugerencias personalizadas acorde a lo solicitado con un contexto narrativo que impulse el consumo, exploración y descubrimiento del usuario." Se puede concluir que el desarrollo del sistema se ha logrado implementar exitosamente. Para ello, se utilizó el modelo LLM *Gemini 2.5 Flash* como herramienta de procesamiento de lenguaje natural para la interpretación de solicitudes textuales de usuarios que previamente otorgaron su consentimiento para el uso de su información de perfil de *Spotify*, específicamente para acceder a sus 15 artistas más reproducidos y las últimas 10 canciones escuchadas en la plataforma.

El sistema fue capaz de generar sugerencias de elementos musicales, incluyendo canciones, álbumes o artistas, que se encuentran relacionados tanto con la información recopilada del usuario como con la solicitud específica planteada por este, abarcando peticiones basadas en emoción, contexto o cualquier otro aspecto particular. Adicionalmente, el elemento sugerido se presentó acompañado de un párrafo introductorio y otro de

información complementaria. Esta información adicional pudo incluir aspectos del contexto histórico y sociocultural, una interpretación de la lírica y las emociones que la obra busca transmitir, además de detalles respecto al sonido y los géneros relacionados. Dicha información se proporcionó en un formato conversacional no formal. De acuerdo con los usuarios de prueba, este formato resultó agradable y sirvió como un motivador para la futura exploración musical, confirmando así que el objetivo principal del proyecto fue alcanzado.

5.2. Estado de objetivos específicos.

En este punto se analizará punto por punto los objetivos específicos en orden de asegurar su estado final, ya sea logrado o no logrado.

- "1. Evaluar y comparar las alternativas disponibles en el mercado respecto a Modelos de Lenguaje y *APIs* de *streaming* musical, para seleccionar la combinación óptima que equilibre capacidades técnicas, documentación robusta y viabilidad económica para el desarrollo del prototipo."

La elección del modelo de lenguaje se fundamentó en un análisis comparativo de costos, capacidad de razonamiento y precisión. No obstante, la industria de los modelos de lenguaje de gran escala (LLM) se encuentra en constante evolución, y las estadísticas y datos relativos a los modelos líderes del mercado han experimentado variaciones significativas hasta la fecha. La selección se centró primordialmente en optar por el modelo más "permisivo" en términos de capas gratuitas, de entre los mejores disponibles, con el fin de mantener un presupuesto de \$0.

Google, con su modelo Gemini, resultó ser el elegido. Sin embargo, sus términos y condiciones para el uso de la capa gratuita han sido modificados con el tiempo, lo que subraya la necesidad de que el modelo a utilizar no sea estático y pueda variar tanto en versión como en proveedor.

Respecto a la selección de la API de *Streaming* de música, la elección de Spotify, dada su predominancia en el mercado, se estableció como la opción idónea, principalmente por la facilidad para encontrar usuarios de prueba con cuentas activas del servicio. Si bien el diseño de la interfaz y

la arquitectura dificultan un eventual cambio de API, el pronóstico indica que esta acción no será necesaria en el futuro cercano.

- "2. Diseñar una interfaz de usuario que permita ingresar y enviar texto de 20 a 300 caracteres."

La interfaz de usuario diseñada cumplió con el objetivo de facilitar el ingreso y la remisión de solicitudes del usuario con la extensión de caracteres establecida, constituyendo este logro el aspecto fundamental para la operatividad del sistema.

- "3. Integrar un modelo de lenguaje natural al sistema de entrada de texto, que permita interpretar las solicitudes del usuario."

Tal como se mencionó previamente, el modelo seleccionado fue *Gemini 2.5 flash*, un modelo perteneciente a la capa gratuita que proporciona un procesamiento rápido, lo cual beneficia la experiencia del usuario. Las pruebas funcionales y las pruebas de usuario concluyeron la adecuada interpretación de la mayoría de las solicitudes. No obstante, existen complicaciones inherentes a aquellas peticiones que contradicen las instrucciones base del modelo, lo que sugiere un margen de error que podría mitigarse mediante la búsqueda del *prompt* base óptimo que garantice libertad al usuario sin comprometer la integridad del sistema.

- "4. Solicitar y utilizar el acceso al perfil de usuario autenticado en el servicio de *streaming* de música para analizar patrones de consumo musical, con el objetivo de generar recomendaciones personalizadas que incluyan contenido no explorado previamente por el usuario."

Este objetivo se logró parcialmente. Si bien se consiguió acceder al perfil de *Spotify* del usuario para proporcionar recomendaciones personalizadas basadas en sus preferencias, no fue posible garantizar que el contenido no hubiera sido explorado previamente por el usuario. La arquitectura RAG implementada, que suministra al modelo una cadena de texto con los principales artistas y las últimas canciones, presenta dificultades para ofrecer el historial completo del usuario debido al elevado costo potencial en términos de *tokens*. Adicionalmente, la API de *Spotify* impone restricciones al acceso total del perfil, lo que incrementa la probabilidad de que el modelo sugiera elementos ya escuchados por el usuario. La

solución a esta limitación conlleva diversos obstáculos y requiere nuevos enfoques arquitectónicos. No obstante, el sistema está concebido para la conversación: el usuario puede indicar que ya conoce la recomendación y el modelo sugerirá un nuevo elemento. Además, el valor del sistema reside no solo en la recomendación en sí, sino también en el contexto adjunto, el cual puede servir como punto de partida para una exploración personal del usuario al revelar aspectos como el género musical, las emociones transmitidas o el contexto sociocultural de la obra.

- "5. Desarrollar la lógica de comportamiento del modelo LLM con el fin de realizar búsquedas en fuentes de información específicas, de acuerdo con la solicitud del usuario, y generar una respuesta que incorpore información adicional, incluyendo análisis lírico, contexto histórico del compositor(es), e información sobre el género o movimiento musical.

Presentándose de manera no estructurada, con el objetivo de simular una conversación humana."

La lógica de comportamiento cumplió con los objetivos establecidos en relación con la adición de información complementaria, el análisis lírico, el contexto histórico y los datos relativos al género o movimiento, presentándose todos estos aspectos en diversas respuestas para simular de manera más efectiva la naturaleza de una conversación. No obstante, se identificaron limitaciones relacionadas con el modelo empleado, el cual, con el fin de mantener un sistema sin costo, no corresponde al modelo LLM considerado actualmente como el más avanzado en términos de inteligencia, precisión y creatividad. Adicionalmente, y en lo concerniente a la recuperación de fuentes de información, el modelo carece de acceso a internet para la generación de respuestas actualizadas, restringiéndose a su información de entrenamiento cuyo límite temporal es el año 2024.

Este aspecto es susceptible de mejora en futuras iteraciones mediante la asignación de un mayor presupuesto y una reevaluación de los modelos más idóneos en el momento.

- "6. Optimizar el uso de recursos externos, en particular la *API* del modelo *LLM*, con el fin de garantizar la funcionalidad mínima del sistema manteniendo el consumo de tokens y solicitudes dentro de un presupuesto reducido."

Si bien se experimentó con múltiples *prompts* base para el sistema, la optimización de los recursos del modelo constituye un aspecto de mejora continua y no puede considerarse un objetivo cumplido. Garantizar la funcionalidad mínima representa una cuestión compleja que requiere más tiempo para una respuesta definitiva, siendo un objetivo que probablemente se mantenga como referente para la optimización constante e indefinida del sistema.

- "7. Realizar pruebas de funcionalidad y rendimiento para garantizar la calidad del sistema."

Este objetivo fue cumplido mediante la ejecución de las pruebas declaradas, asegurando la calidad del sistema, al menos en una fase con usuarios de prueba. No obstante, se requiere la realización de nuevas pruebas de rendimiento, latencia y capacidad si se contempla desplegar el sistema para su uso público en el futuro.

- "8. Evaluar la funcionalidad principal del sistema mediante pruebas controladas^o con usuarios de distintos perfiles, con el fin de recopilar retroalimentación sobre la satisfacción, utilidad y potencial aumento de interés tras uso de la herramienta."

El objetivo se cumplió. La participación de 20 usuarios posibilitó el análisis de comportamientos diversos frente al sistema, lo cual proporcionó nuevas perspectivas, oportunidades de mejora, la identificación de errores no detectados y la confirmación del cumplimiento del propósito principal: ofrecer una recomendación musical que permita al usuario no solo acceder a un nuevo elemento, sino también comprender su contexto intrínseco, un aspecto relevante para una nueva apreciación, sirviendo como motivación para el descubrimiento, la exploración y la escucha activa de música, lo que se traduce en un incremento en la pasión por la música.

6. REFERENCIAS BIBLIOGRÁFICAS

C.Stryker y J.Holdsword, "What is NLP (natural language processing)?[online]?",

What is NLP?, Agosto, 2024. Disponible en:

<https://www.ibm.com/es-es/think/topics/natural-language-processing>

C.Stryker, "What are large language models (LLMs)?[online]", *What are LLMs?*, Disponible en:

<https://www.ibm.com/think/topics/large-language-models>

M.Goodwin "What is an API (application programming interface)?[online]",

What is an API?, Abril, 2024. Disponible en: <https://www.ibm.com/mx-es/think/topics/api>

NCCIH "Música y salud: Lo que tiene que saber[online]" *¿Puede la música ser buena para usted?*, Septiembre, 2022. Disponible en: <https://www.nccih.nih.gov/health/espanol/musica-y-salud-lo-que-tiene-que-saber#:~:text=Estos%20efectos%20en%20el%20cerebro,y%20promover%20los%20v%C3%ADnculos%20sociales>.

SINC, "La música utiliza circuitos neuronales específicos para activar el placer[online]", Marzo, 2014. Disponible en:

<https://www.agenciasinc.es/Noticias/La-musica-utiliza-circuitos-neuronales-especificos-para-activar-el-placer>

J.C.Correa "El análisis del contenido emocional de la música y cómo usarlo para una ejecución expresiva", Primera edición 2020. universidad autónoma de aguascalientes, 2020.

E.Luque "Qué es el efecto Eliza, o los riesgos de la humanización de la inteligencia artificial[online]", Abril, 2024. Disponible en: <https://computerhoy.20minutos.es/ciencia/efecto-eliza-riesgos-humanizacion-ai-1376293>

G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley, 2005.

C. Coronel and S. Morris, *Database Systems: Design, Implementation, &*

Management, 13th ed., Cengage Learning, 2016.

I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard,
Object-Oriented Software Engineering: A Use Case Driven Approach,
Addison-Wesley, 1992