UNIVERSITAT DE BARCELONA

Trabajo de fin de grado

GRADO DE INGENIERÍA INFORMÁTICA

Facultad de Matemáticas e Informática
Universidad de Barcelona

CHATBOT DE RECOMENDACIÓN MUSICAL BASADO EN DIALOGFLOW

Autor: Chang Ye

Director:	**Dra. Maria Salamó Llorente**
Realizado a:	**Departamento de Matemáticas e Informática**

Barcelona, 13 de junio de 2023

Abstract

A chatbot is a virtual assistant that can interact with users through text or voice messages. Currently, there are few virtual assistants linked to a recommendation system, which can recommend items or content of interest to the user.

In this project we propose to develop a chatbot specialized in the field of music that can interact with users through text messages. It will be able to provide relevant information about artists, song recommendations and music genres. In addition, a usability evaluation will be performed with real users and the results obtained from this evaluation.

Resumen

Un chatbot es un asistente virtual que puede interactuar con los usuarios mediante mensajes de texto o voz. Actualmente, existen pocos asistentes virtuales vinculados con sistema de recomendación, los cuales permiten recomendar items o contenido que sea del interés del usuario.

En este proyecto se propone desarrollar un chatbot especializado en el ámbito de la música que pueda interactuar con los usuarios mediante mensajes de texto. Estará capacitado para proporcionar información relevante sobre artistas, recomendaciones de canciones y géneros musicales. Además, también se hará una evaluación de usabilidad con usuarios reales y los resultados obtenido de esta evaluación.

Resum

Un chatbot és un assistent virtual que pot interactuar amb els usuaris mitjançant missatges de text o de veu. Actualment, hi ha pocs assistents virtuals vinculats a un sistema de recomanació, que pugui recomanar items o continguts d'interès per a l'usuari.

En aquest projecte proposem desenvolupar un chatbot especialitzat en el domini de la música que pugui interactuar amb els usuaris mitjançant missatges de text. Podrà proporcionar informació rellevant sobre artistes, recomanacions de cançons i gèneres musicals. A més, es realitzarà una avaluació d'usabilitat amb usuaris reals i els resultats obtinguts d'aquesta avaluació.

Agradecimientos

En primer lugar, quiero dar agradecimientos a mi tutora, Dra. Maria Salamó, por su valioso conocimiento, orientación y apoyo durante todo el desarrollo de este proyecto y la oportunidad de realizar este proyecto.

En segundo lugar, a David Contreras y Víctor por su generosidad al compartir su experiencia y por su disposición a resolver las dudas que surgieron durante la implementación del chatbot.

Por último, pero no menos importante, quiero agradecer a mi familia y a mis amigos por su constante apoyo a lo largo de esta etapa de mi vida.

Índice

1. Introducción **1**

1.1. Motivación 1

1.2. Objetivos 1

1.3. Planificación 2

2. Estado del arte **3**

2.1. Sistemas de Recomendación 3

2.1.1. Datos en un Sistema de Recomendación 3

2.1.2. Estructura de datos 4

2.1.3. Métricas de similitud 4

2.1.4. Clasificación de Sistemas de Recomendación 6

2.2. Chatbots 8

2.2.1. Historia y antecedentes 9

2.2.2. Clasificación de chatbot 10

2.2.3. Técnicas de diseño 12

2.2.4. Reconocimiento de Intenciones y PLN 12

2.2.5. Procesamiento de Lenguaje Natural 14

2.2.6. Caso de usos reales 15

3. Herramientas utilizadas **18**

3.1. Introducción 18

3.2. Flask 18

3.3. Dialogflow 19

3.3.1. Intentos 20

3.3.2. Entidades 21

3.3.3. Integraciones 22

3.4. Cornac 22

4. Análisis previo **23**

4.1. Dataset 23

4.1.1. Exploración de los datos 24

5. Implementación **26**

5.1. Desarrollo del chatbot 26

5.1.1. Flujo conversacional 26
5.1.2. Creación de intentos 28
5.1.3. Entrenamiento de frases 30
5.1.4. Entidades 32
5.1.5. Front-end 33
5.1.6. Back-end 35
5.1.7. Recomendador 36
5.2. Cuestionarios 36
5.2.1. Pre Test 37
5.2.2. Post Test 40
5.3. Instrucción para ejecutar la aplicación 43

6. Conclusiones **44**
6.1. Conclusiones del proyecto 44
6.2. Trabajo futuro 44

7. Referencias **45**

Índice de figuras

1. Diagrama de Gantt 2
2. Clasificación de Recomendación [39] 6
3. Test de Turing 9
4. Evolución del interés de los chatbots en las búsquedas de Google . . 10
5. Clasificación general de los chatbots 10
6. Estructura del proyecto 18
7. Arquitectura general de Dialogflow 19
8. Ejemplo de intent 20
9. Flujo básico de intents 21
10. Integraciones disponibles en Dialogflow 22
11. Top 10 categorías musicales 25
12. Diagrama del diseño de la conversación 27
13. diagrama del diseño de la conversación 29
14. Parámetros necesarios de un webhook 29
15. Vista general de un Intento 30
16. Frases de entrenamiento del intento Artist 31
17. Confianza de detección de intentos 31
18. Front-end de la aplicación 33
19. Porcentaje de usuarios de si conocían el concepto de chatbots . . . 37
20. Porcentaje de usuarios de si han interactuado con un chatbot 37
21. Calificación según el interés los chatbots 38
22. Calificación de la música 38
23. Porcentaje del lugar donde van a descubrir nuevas canciones 38
24. Calificación de si les interesa un chatbot que recomienda música . . 39
25. Calificación de si son cómodos interactuando con un chatbot 39
26. Calificación de si ha sido fácil conversar con el chatbot 40
27. Calificación de si les ha gustado los las canciones recomendadas . . 40
28. Calificación de si los elementos recomendados van encaja con sus gustos 41
29. Porcentaje de usuarios que han encontrado bibliografía de los artistas 41
30. Calificación de satisfacción 42
31. Calificación de si recomendarían a otras personas el chatbot 42
32. Posibles mejoras 42

Índice de cuadros

1. Comparación de las métricas de similitud 6
2. Estructura de la tabla Conversaciones 35

1. Introducción

En esta sección se va a introducir el marco del problema, la motivación que me ha llevado a realizar este trabajo de fin de grado, los objetivos y la planificación.

1.1. Motivación

En las películas siempre hemos visto como podría ser una conversación entre una persona y robot, cosa que hace años atrás era de ciencia ficción. Sin embargo, gracias a la evolución de la inteligencia artificial (IA), esta interacción se ha convertido en algo de uso cotidiano.

Este trabajo de fin de grado nace de un interés personal por las ramas de IA y Sistemas de Recomendación (SR) que se han presentado a lo largo del grado de Ingeniería Informática. Del conocimiento adquirido, surge la idea de desarrollar un chatbot [1] con la capacidad de dialogar con el usuario y, además, permita recomendar contenidos. En concreto se ha centrado en el dominio de música por ser uno de los más interés suscita en general a las personas.

Otra de las motivaciones del proyecto es que permite profundizar y aplicar muchos de los conocimientos adquiridos a lo largo del grado. Además de comprender a fondo los sistemas de recomendación y qué són los chatbots, también se hará una análisis de usabilidad, se hará un diseño del sistema y se aprenderán nuevas plataformas, como DialogFlow [2].

1.2. Objetivos

El objetivo principal de este proyecto es el diseño e implementación del chatbot con recomendación en el ámbito de la música, una área que no se ha visto durante el grado de Ingeniería Informática.

Durante el desarrollo de este, se pretende completar los siguientes subobjetivos:

- Proporcionar una visión general sobre el estado actual de los Sistemas de Recomendación.
- Definir la estructura de datos más utilizados para el desarrollo de Sistemas de Recomendación.
- Proporcionar una visión general sobre el estado actual de los chatbots.
- Implementación de un Sistema de Recomendación para la música.
- Implementación del chatbot en una aplicación web.
- Análisis de la usabilidad con usuarios reales mediante encuestas.

1.3. Planificación

El proyecto se ha estructurado en distintos bloques de trabajo: investigación, diseño e implementación, y redacción de la memoria. A su vez, los objetivos se desglosan en las siguientes tareas:

1. Lectura e investigación.
2. Redacción de la parte teórica a partir de las lecturas y los conocimientos adquiridos durante la investigación.
3. Diseño del sistema.
4. Desarrollo del proyecto.
5. Encuestas y análisis de resultados.
6. Redacción de la parte práctica.
7. Revisión y perfeccionamiento del código y la memoria.

En la Figura 1 podemos ver el tiempo dedicado a cada tarea.

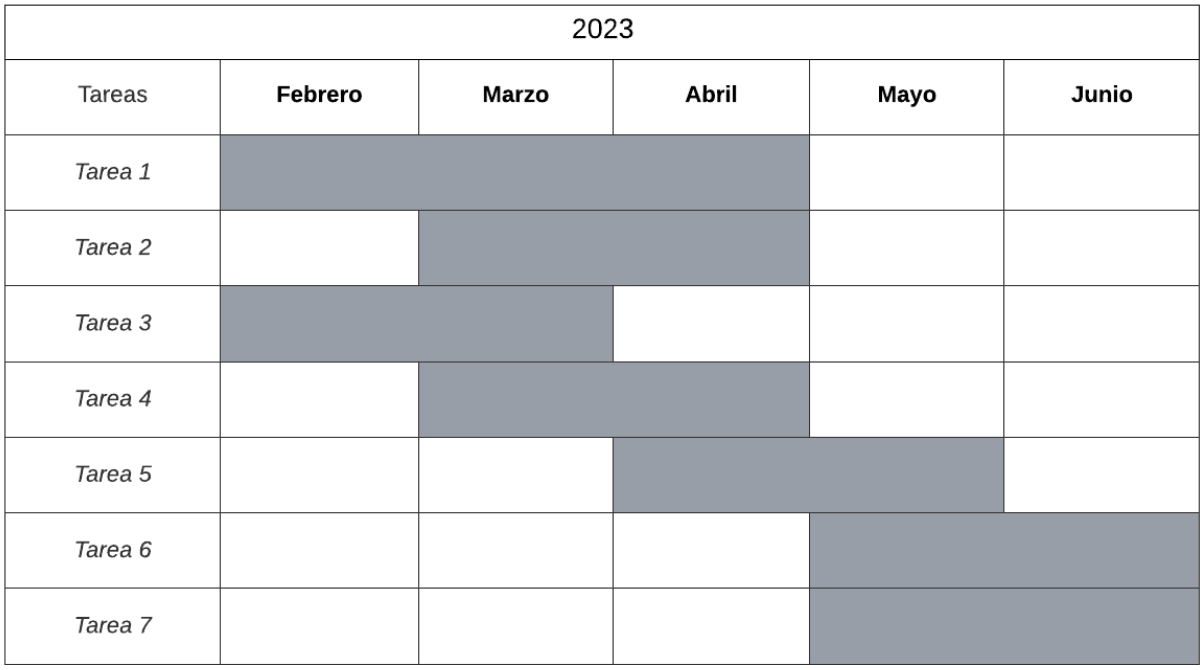

Figura 1: Diagrama de Gantt

2. Estado del arte

En este capítulo se va a presentar el estado del arte relacionado con los temas tratados en el proyecto. En primer lugar, se presentan los Sistemas de Recomendación (véase sección Subsección 2.1) y en segundo lugar los chatbots (véase Sección Subsección 2.2).

2.1. Sistemas de Recomendación

En la era digital, generamos una cantidad considerable de datos mientras navegamos por Internet, y a veces puede resultar difícil para encontrar la información que queremos. Un Sistema de Recomendación (SR) es un motor de filtrado de información que establece un conjunto de criterios sobre los datos del usuario y ayuda a los usuarios a encontrar información personalizada según sus preferencias. En el momento de recomendar un producto, las opiniones y valoraciones de los usuarios desempeñan un papel importante (Shah et al., 2017).

Una buena implementación de un SR no solo ayuda al usuario, sino que también es de suma importancia para las empresas de e-commerce como Amazon. Los SR pueden recomendar productos o servicios nuevos que pueden ser de interés para el usuario, haciendo que el usuario termine comprando un producto no considerado (Mehta et al., 2020b).

El motor de recomendación puede realizar una recomendación basándose en los productos más populares entre los usuarios y en función de sus valoraciones. Estos métodos tienen sus inconveniencias. El primer método al recomendar los más populares entre todos los usuarios recomendará siempre los mismos productos para cualquier usuario, sin tener en cuenta sus preferencias. En el caso del segundo método, a medida que aumentamos la cantidad de usuarios, también lleva consigo un aumento en la complejidad del algoritmo de recomendación que puede ser una tarea difícil de realizar eficientemente.

2.1.1. Datos en un Sistema de Recomendación

Los Sistemas de Recomendación recolectan datos activamente de diferentes fuentes [25], por ejemplo, páginas webs, aplicaciones de smartphone, entre otros. Estos datos están relacionados con los usuarios y también los items o productos. Dependiendo del enfoque del SR, se pueden aplicar diferentes técnicas de recomendación.

Los datos que recolecta un SR se pueden categorizar en las siguientes categorías:

- **Explícita:** es información que se proporciona intencionalmente por parte del usuario, como por ejemplo la valoración que da después de comprar un producto.
- **Implícitamente:** es información que no se proporciona intencionalmente, que pueden ser historial de compras, historial de búsquedas, entre otros.

2.1.2. Estructura de datos

Como se ha comentado anteriormente, los datos que recopila un SR son principalmente sobre usuarios e items. A continuación se pueden ver las tres principales fuentes de datos (i.e., en inglés se denominan datasets) que necesita un SR para su funcionamiento.

- **Elementos**: los elementos son los ítems o productos que se recomiendan, que también puede ser personas dependiendo de su enfoque de recomendación (por ejemplo, amigos que quizás conoces en las redes sociales). Se pueden definir por muchas características. En el caso de una canción de música, las características serían el título, el autor/cantante/grupo, la duración, entre otros.
- **Usuarios**: para poder dar recomendaciones personalizadas es necesario tener información relevante sobre el usuario en concreto. De este modo, el SR puede usar las preferencias del usuario para recomendarle un producto que encaje con sus gustos.
- **Calificaciones**: son las interacciones entre un usuario y un elemento, más en concreto, la retroalimentación con la que califica el elemento. Las calificaciones pueden tomar diferentes valores, pueden ser valoraciones numéricas, binario.

Definición formal del problema

Según (Esteban et al., 2018), podemos definir el problema de un SR como lo siguiente:

Dado un conjunto de M usuarios $U = u_1, u_2, ..., u_M$, un conjunto de N elementos $I = i_1, i_2, ..., i_N$, y un conjunto de elementos I_{ui} que han sido valorados por el u_i, la tarea de un SR consiste en encontrar para un usuario u, un elemento $i \in I \setminus I_{ui}$ en el cual existe la posibilidad de que u esté interesado.

Las calificaciones asignadas por los usuarios, denotadas como u_i, pueden tomar diferentes valores dentro de un conjunto definido S. Este conjunto puede representar una escala numérica, como por ejemplo $S = [1, 5]$, o puede ser una clasificación binaria, como $S = \{A, B\}$.

Es importante destacar que se asume que cada par $\{usuario, elemento\}$ puede recibir solo una calificación. Además, se introduce la notación I_{uv} para referirse al conjunto de elementos que han sido calificados por dos usuarios, u y v. En otras palabras, I_{uv} representa la intersección de los conjuntos de elementos valorados por los usuarios u y v. De manera similar, se utiliza la notación U_{ij} para hacer referencia al conjunto de usuarios que han calificado los elementos i y j.

2.1.3. Métricas de similitud

Las métricas de similitud en un RS se refiere a una medida utilizada para determinar la similitud entre usuarios o elementos. Esta métrica desempeña un papel

fundamental a la hora de implementar un SR eficiente.

Existen diversas métricas de similaridad en un SR [26], como la distancia Euclidiana, similitud del coseno, distancia de Manhattan, distancia de Minkowski.

Distancia euclidiana

Es la distancia más popular cuando los datos son escalares y es la distancia entre dos puntos.

$$d_E(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.1}$$

Donde x_i e y_i son dos puntos (usuarios o items en SR) y n la cantidad de preferencias de los usuarios o la cantidad de características de los items.

Similitud del coseno

Esta métrica considera los elementos como vectores en un espacio de n dimensiones y calcula su similitud mediante el coseno del ángulo que forman.

$$d_C(x, y) = \frac{x \cdot y}{\|x\| \cdot \|y\|} \tag{2.2}$$

Donde el producto entre dos vectores x e y se representa como $x \cdot y$, y la norma del vector x se presenta como $\|x\|$.

Distancia de Manhattan

La distancia Manhattan entre dos puntos x e y (usuarios o items en SR) en un espacio de dimensiones n se calcula sumando el valor absoluto de las diferencias entre las coordenadas de los puntos.

$$d_M(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{2.3}$$

Donde x_i e y_i representan las coordenadas de los puntos x e y en la dimensión i.

Distancia de Minkowski

La distancia de Minkowski es una distancia generalizada que engloba tanto la distancia Euclidiana como la distancia Manhattan.

$$d_M(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^r\right)^{\frac{1}{r}} \tag{2.4}$$

Donde x e y son los puntos (usuarios o items en SR) en el espacio de n dimensiones, x_i e y_i las coordenadas de los puntos en la dimensión i, y r determina el grado de la distancia.

Para terminar con esta las métricas, en Tabla 1 se muestra una breve explicación de ventajas y desventajas de las métricas que hemos visto.

Cuadro 1: Comparación de las métricas de similitud

Métricas	**Ventajas**	**Desventajas**
Euclidiana	Es simple y muy popular. Funciona bien cuando el dataset es consolidado.	Sensitivo a valores extremos.
Coseno	Independiente de la longitud del vector y rotación.	No está sujeto a conversiones lineales.
Manhattan	Funciona bien cuando los datos del dataset están separados o compactados.	Sensible a valores extremos.
Minkowski	Funciona bien cuando los datos del dataset están separados o compactados.	Dificultad para manejar características que están en diferentes escalas.

2.1.4. Clasificación de Sistemas de Recomendación

En función de la técnica utilizada para generar las recomendaciones, los SR se pueden clasificar en diferentes categorías; filtrado colaborativo [27], filtrado basado en contenido [28], sistemas basados en conocimiento [29] y sistemas híbridos [3].

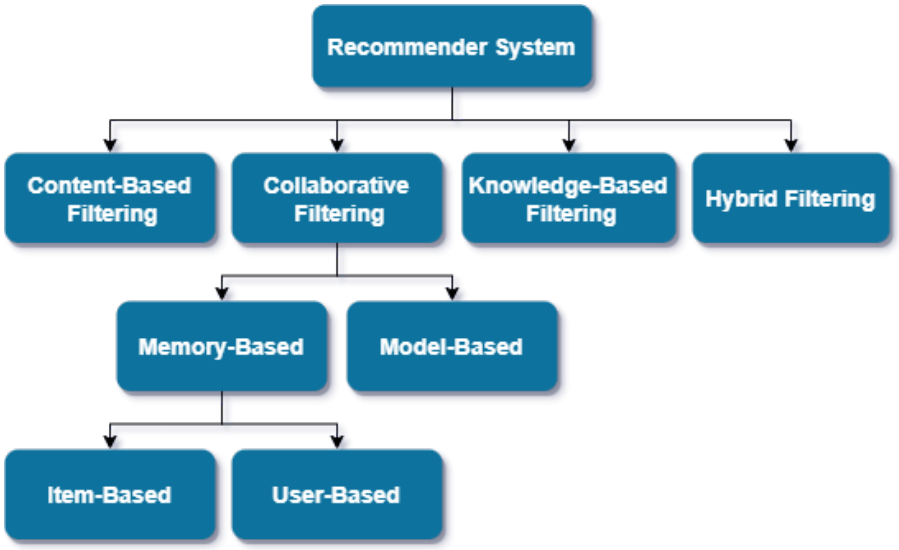

Figura 2: Clasificación de Recomendación [39]

En la Figura 2 se puede ver una versión simplificada depediendo de la técnica que emplea los SR.

Filtrado colaborativo

Los SR basado en filtrado colaborativo (en inglés Collaborative Filtering) tienen el objetivo de recomendar, predecir la utilidad de un determinado elemento al usuario basándose en el historial pasado de la interacción de los usuarios con los elementos (Schafer et al., 2007).

(Breese et al., 1998) describe los algoritmos del filtrado colaborativo en dos clases:

- **Memory-based**: utiliza el historial pasado de las interacciones del usuario para predecir la clasificación que el usuario le daría a este elemento.

  Este tipo de algoritmo son relativamente fáciles de implementar y utilizan únicamente la información de las interacciones de los usuarios sin necesidad de conocimiento adicional sobre los usuarios o los elementos. También son estables y se adaptan a las preferencias de los usuarios, ya que como he mencionado, se basa en el historial pasado de las interacciones del usuario.

  Sin embargo, tiene problemas de escalabilidad con las aplicaciones reales. Este tipo de algoritmo tienen como incoveniente que requieren un numero de recursos computacionales que crece a medida que aumenta el numero de de usuarios e items.

- **Model-based**: a diferencia de los algoritmos memory-based, los filtrados colaborativos basado en modelo utilizan algoritmos complejos de aprendizaje automático para generar las recomendaciones.

  Una de las ventajas de este algoritmo es su capacidad para manejar conjunto de datos grandes y dispersos, ya que son capaces de capturar patrones complejos y realizar recomendaciones de elementos diferentes o que no han sido vistos en el pasado.

  Igual que en el caso anterior, este tipo de algoritmo requiere un procesamiento computacional más intensivo, especialmente para el entrenamiento del modelo. Además, cada vez que hay usuarios o elementos nuevos, es posible que se necesite volver a entrenar el modelo.

Filtrado basado en contenido

A diferencia del filtrado colaborativo que está basado en las interacciones de los usuarios, el filtrado basado en contenido se centra en las características y los atributos de los elementos. Las características pueden incluir palabras clave, temas, géneros, entre otros.

Las ventajas que puede aportar este tipo de filtrado respecto a los de colaborativo es que no tiene problemas con los elementos nuevos, ya que recomienda elementos basándose en una comparación del contenido y del usuario.

Al igual que cualquier tipo de SR, también presenta algunas desventajas. Tiene dificultad para capturar la complejidad y utiliza de las preferencias de los usuarios que van más allá de las características de los elementos. También sufre del problema donde las recomendaciones se limitan a elementos similares que el usuario ya ha visto en el pasado, limitando los nuevos elementos a recomendar.

Filtrado basado en conocimiento

El filtrado basado en conocimiento se centra en los patrones de comportamiento del usuario o en las características de los elementos basados en un dominio en específico. Según (Tarus et al., 2018), existen varias formas de implementar este tipo de filtrado, dos de los más conocidos son: case-based y constrain-based.

El primero tiene la ventaja de poder adaptarse a situaciones específicas y proporcionar recomendaciones a los usuarios basadas en experiencias. Además, es flexible y puede manejar la actualización de preferencias del usuario.

El segundo utiliza un conjunto de reglas predefinidas para realizar las recomendaciones. Este conjunto de reglas pueden ser, por ejemplo, las preferencias del usuario.

Sistemas híbridas

Combinan múltiples técnicas que hemos visto hasta ahora. Este tipo de SR intenta superar las desventajas presentadas previamente aprovechando las ventajas que le puede aportar otro algoritmo, para así mejorar la precisión de las recomendaciones.

A lo largo del tiempo, se han propuesto varios modelos híbridos donde se proponen ofrecer mejoras en algún sector en concreto [31, 32, 33].

2.2. Chatbots

Con el paso del tiempo, la Inteligencia Artificial se ha incorporado en el día a día de nuestras vidas, aportando utilidades y descubrimiento que facilitan la vida de las personas. Esto ha llevado a que las empresas inviertan en nuevos productos derivados de esta tecnología.

En la actualidad, aparte de las conversaciones entre humanos, también existen comunicación entre humanos y máquinas. Esta gran revolución, combinada con la inversión en IA, ha dado lugar a la aparición de programas informáticos con los que se puede mantener una conversación, solicitar información, o dar órdenes (Contac Center Hub, 2018).

Los chatbots son conjuntos de algoritmos o programas informáticos que están diseñados para poder comunicarse con los humanos utilizando el lenguaje natural, y para esto, utilizan Procesamiento del Lenguaje Natural (PLN).

2.2.1. Historia y antecedentes

En 1950, Alan Turing publicó el artículo *Computing machinery and intelligence* [20] donde desarrolló el método el Test de Turing ("*¿Pueden las máquinas pensar?*"). En dicho test consiste en una conversación entre una persona y una máquina diseñada especialmente para interactuar con humanos y tenía que identificar correctamente si está hablando con una máquina o con una persona después de haber mantenido la conversación durante 5 minutos.

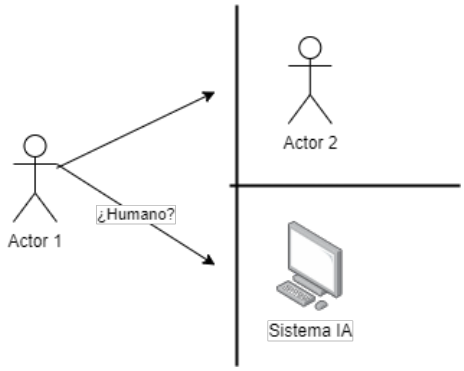

Figura 3: Test de Turing

Uno de los primeros bots conocidos fue desarrollado en 1966, conocido como ELIZA [6]. Fue creado como un psicólogo cuyo enfoque principal era la interacción con los humanos a través del lenguaje natural. ELIZA era capaz de mantener conversaciones con usuarios, haciéndoles creer que el programa era capaz de entender como si fuera un humano, y el primero en pasar el Test de Turing. Utilizaba una base de datos con frases hechas con las que podía utilizar para reconocer palabras clave, no obstante, tenía la limitación de no poder memorizar ni aprender de sus conversaciones con los usuarios.

En 1972 se creó una personalidad llamada PARRY [7] que simulaba un paciente para ELIZA, los dos chatbots eran capaces de mantener una conversación.

Desde entonces, la tecnología de chatbots ha avanzado a pasos gigantes gracias a los avances tecnológicos en los ámbitos de IA y PLN hasta hoy en día que conocemos a los famosos asistentes personales como Apple Siri [8], Microsoft Cortana [9], Amazon Alexa [10], IBM Watson [11] y Google Assistant [12].

En noviembre de 2022, OpenAI creó ChatGPT [13], un chatbot conversacional que ha generado la atención en todo el mundo por su impresionante capacidad de generar respuestas que parecen ser escritas por un humano. En la Figura 4 podemos ver que el interés en el término de chatbot siempre ha ido en auge, hasta que nació ChatGPT y el interés aumentó exponencialmente.

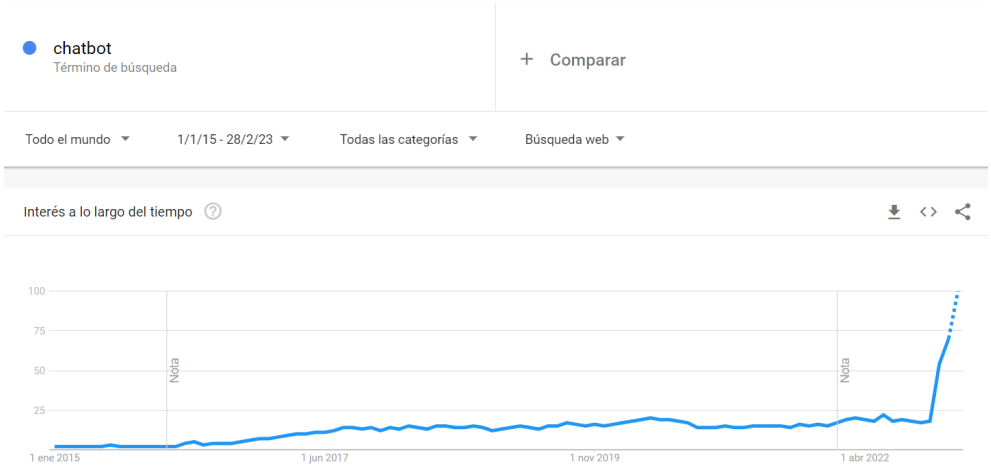

Figura 4: Evolución del interés de los chatbots en las búsquedas de Google

2.2.2. Clasificación de chatbot

Según (Hussain et al. 2019), la clasificación general de los chatbots pueden ser basada en cuatro criterios (véase Figura 5);

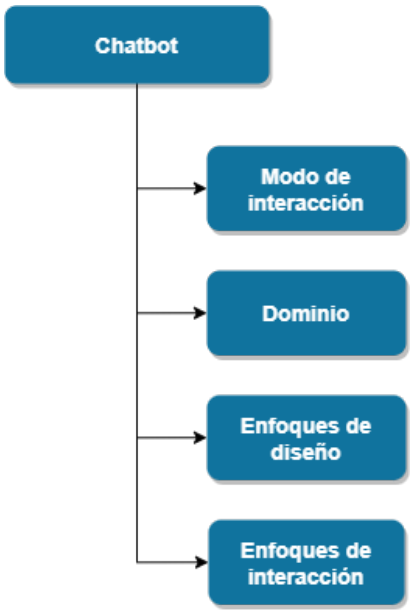

Figura 5: Clasificación general de los chatbots

1. **Modos de interacción**: los usuarios pueden conversar con los chatbots tanto vía texto como a través de voz.
    - **Interacción basada en voz**: este tipo de chatbot, por ejemplo, Apple Siri, Amazon Alexa y Google Assistant han ganado una gran popularí-

dad y se han convertido en herramientas de uso cotidiano para muchas personas. Utilizan tecnologías de reconocimiento y síntesis de voz para poder reconocer las preguntas de los usuarios.

- **Interacción basada en texto**: por otro lado, este tipo de chatbot facilitan la comunicación con los usuarios a través de texto, con PLN son capaces de comprender y generar respuestas en lenguaje natural. Suelen integrarse en aplicaciones de mensajería para brindar una mejor experiencia.

2. **Dominio**: los chatbots tienen la capacidad de adaptar su conversación en función del dominio al que pertenecen. El dominio define el ámbito o área de conocimiento en el cual se especializa.

   - **Dominio abierto**: tienen la capacidad de establecer conversaciones sobre cualquier ámbito en lenguaje natural. Esta flexibilidad conlleva un mayor nivel de complejidad en su implementación, ya que require de algoritmos más complejos para poder responder adecuadamente a las diversas consultas.
   - **Dominio cerrado**: este tipo de chatbot se dedica exclusivamente a un tema en específico. Está diseñado para ofrecer respuestas y soluciones, por lo que se centra en brindar un servicio especializado, por ejemplo, la gestión de reservas de entradas para eventos. Estos chatbots están desarrollados con el propósito de optimizar y agilizar la interacción con los usuarios.

3. **Enfoques de diseño**: dependiendo del enfoque de diseño, los chatbots tienen un funcionamiento y características distintas.

   - **Enfoque basado en la lógica**: se basan en reglas de lógica predefinidas para tomar decisiones y generar respuestas. Utiliza un conjunto de reglas y patrones para dar la respuesta más adecuada en función de la pregunta del usuario. Este enfoque es particularmente útil en situaciones donde la conversación tiene un flujo estructurado, por ejemplo, el servicio para pedir una cita sanitaria o atención al cliente. Tiene la desventaja de que no son capaces de responder a ninguna pregunta que no esté en las preguntas predefinidas [21].
   - **Enfoque basado en la recuperación**: permite a los chatbots proporcionar respuestas claras y basadas en un repositorio de respuestas predefinidas. La desventaja de este enfoque es que tiene problemas para dar una respuesta en función del contexto.
   - **Enfoque generativo**: la idea principal de este enfoque es desarrollar una nueva conversación a partir de entrenar un gran dataset que contiene conversaciones anteriores. Este supera la desventaja del enfoque anterior aplicando una combinación de técticas de aprendizaje como el aprendizaje supervisado, no supervisado y aprendizaje por refuerzo [24]

4. **Enfoques de Interacción**:

- **Chatbot Orientado a Tareas**: este enfoque se centra en proporcionar asistencia y realizar tareas específicas para los usuarios. Brindan ayuda en escenarios específicos, como el sector bancario [16], atención al cliente [17], entre otros.
- **Chatbot no Orientado a Tareas**: por otro lado, este tipo de chatbot no tienen una tarea o propósito específico. Están diseñados para simular interacciones no estructuradas, imitando conversaciones humanas.

2.2.3. Técnicas de diseño

Los tipos de chatbots vistos en la sección anterior están basados en diferentes técnicas, en esta sección veremos algunas de las más importantes [14].

- **Análisis sintáctico**: esta técnica extrae la información relevante de los textos y lo convierte en un conjunto de palabras simples para poder guardarlos y manipularlos más fácilmente. ELIZA fue el primer chatbot que se benefició de esta técnica.
- **Concordancia de patrones**: es una técnica fundamental en chatbots cuyo función es la de responder a preguntas. Aunque los primeros chatbots como ELIZA utilizaba patrones simples, los chatbots modernos emplean las ventajas de este modelo de forma más compleja.
- **AIML**: AIML, o lenguaje de Marcado de Inteligencia artifical, es un lenguaje derivado de XML (Lenguaje de Marcado Extensible) para crear el flujo conversacional de los chatbots. Se compone de elementos AIML, que son objetos conocidos como categorías. Estos están formados por unidades llamadas temas y contienen patrones y "plantillas" que permiten al chatbot interpretar la pregunta del usuario y generar una respuesta [19].
- **Chatscript**: es una herramienta de autoría de código abierto para construir chatbots. Tiene una estructura más eficiente que AIML para concordancia de patrones. Chatscript ganó el premio Loebner y tiene una estructura eficiente basada en el lenguaje de programación C++.
- **Las Redes Neuronales Artificiales**: Las Redes Neuronales Artificiales o por sus siglas en inglés (ANN) son una secuencia de algoritmos que intentan imitar la estructura del cerebro humano. Estas redes han permitido el desarrollo de chatbots inteligentes, aunque los chatbots basados en ANN utilizan técnicas estructurales que difieren de los chatbots basados en reglas. Pueden utilizar enfoques tanto de recuperación como generativos (vistos en la subsección anterior) para generar respuestas.

2.2.4. Reconocimiento de Intenciones y PLN

La asistencia en diálogos han sido capaces de aportar recomendaciones personalizadas cada vez mas precisos con el paso del tiempo. Más notablemente en empresas

de e-commerce y de plataformas de streaming online, por ejemplo, Netflix. Estos sistemas son creados para realizar unos objetivos específicos basados en la filtración de la información.

Uno de los principales retos de los chatbots es la de extracción de intención para poder realizar la acción correcta dada la solicitud del usuario. Esto es sumamente importante, ya que si el chatbot no es capaz de entender la entrada del usuario, no sería posible mantener una conversación fluida. Por tanto, identificar la acción adecuada según la solicitud del usuario es una tarea esencial para todo sistema de diálogo.

Clasificación de Intenciones independiente del Dominio

El conjunto de intenciones del usuario varia según las necesidades del usuario y condiciones del sistema. Normalmente este conjunto de intenciones se encuentran predefinidos, pero no hay suficiente investigación relevante para la clasificación de intención del usuario en los chatbots. Aunque la mayoría de los chatbots disponen de un conjunto de intenciones de usuario comunes, podemos agrupar las intenciones en las siguientes intenciones principales (Moradizeyveh, 2022):

- **Iniciar, abandonar, reiniciar la conversación**: la primera y la más básica, una conversación se comienza con un saludo, que puede ser realizado por el agente o el usuario. En ciertos escenarios, el usuario podría querer iniciar una conversación y después decidir reiniciar reiniciarlo. En última instancia tenemos la finalización de la conversación. La conversación puede terminar y darse por terminada o redirigir al usuario independientemente de su satisfacción [22]. Esto es normal en chabots de atención al cliente, donde el usuario inicia la conversación para intentar solventar algún problema, pero ese problema no se pueda solucionar de manera automatizada y sea necesario redirigirlo a un experto.

- **Charla**: la charla normal es un elemento sumamente importante para la experiencia del usuario en el uso de los chatbots. Esto se ha demostrado en una investigación [23], dando a entender que la capacidad de mantener una conversación natural podría ser un objetivo en común de todos los chatbots.

- **Petición de explicaciones y detalles**: los usuarios pueden preguntar por recomendaciones al principio de la conversación, incluso después de haber recibido los detalles o explicación. En estas situaciones, el usuario quiere saber más acerca del item, por lo tanto, aprender de los detalles es una parte importante del sistema. Por otro lado, también puede ocurrir el caso en donde el agente pide al usuario proporcionar más detalles para saber con exactitud la intención del usuario.

- **Preferencias y recomendaciones**: las preferencias de los usuarios pueden adoptar de muchas formas, por lo que comprender las preferencias es un factor crítico para los Sistemas de Recomendación. Sin conocer las preferencias del

usuario, es muy probable que se recomiende algo que no sea de su agrado, por ejemplo en el caso de música, una canción de género Pop es muy popular entre todos los usuarios, pero al usuario en concreto no le gusta Pop.

- **Retroalimentación**: la retroalimentación del usuario es una fuente de información muy importante para que los chatbot vinculado a SR puedan aprender más acerca del usuario para poder ofrecer una recomendación mejor basado en la experiencia del usuario y su satisfacción. La satisfacción del usuario pueden ser negativas o positivas, y son ganados durante la conversación o mediante algún cuestionario sobre la experiencia del usuario al final de la conversación.

Clasificación de intención abierta

Comprender las intenciones del usuario a partir

Comprender las intenciones del usuario a partir del lenguaje escrito es vital para ofrecer respuestas automatizadas. Los estudios en este campo se dividen en dos categorías principales: la investigación de intenciones predefinidas específicas (clasificación de mundo cerrado) y el descubrimiento de intenciones abiertas (mundo abierto). La mayoría de las investigaciones actuales se basan en una intención predefinida e intentan clasificar los enunciados en una lista restringida de propósitos. Gestionar la clasificación de intenciones abiertas es un reto importante, y extraer la intención correcta es significativo en los sistemas conversacionales.

Propusieron un modelo híbrido que estima la probabilidad de las diferentes clases de objetos para determinar la clasificación de la intención abierta. Introducen una nueva línea de base basada en PLN y visión por ordenador para estimar la probabilidad de la distribución y detectar los ejemplos fuera de distribución. Presentan una técnica para descubrir el límite de decisión adaptable para identificar la clasificación de intención abierta.

2.2.5. Procesamiento de Lenguaje Natural

Según (Moradizeyveh, 2022), el Procesamiento del Lenguaje Natural (PLN) es un campo basado en datos para comprender, manipular y producir lenguaje humano mediante la extracción de información de datos textuales a través de estimaciones estadísticas y probabilísticas. PLN es una de las razones por la que ha habido tanto avance en los chatbots.

PLN juega un papel importante en interpretar el lenguaje humano mediante avanzados algoritmos específicos y proporcionar respuestas adecuadas mediante técnicas de aprendizaje automático (AA). Esto es aplicado en aplicaciones reales como los asistentes de Google, que tal y como he comentado, utiliza NLP para el procesar y analizar los textos. A la hora de analizar los textos, este extrae la información valiosa que hay en los textos para intentar comprender la intención del usuario.

Tradicionalmente, la PLN se clasifica en áreas básicas y áreas de aplicación [22]. Basándose en esta clasificación, muchos campos diversos (por ejemplo, sistemas con-

versacionales, traducción automática y sistemas de respuesta a preguntas) utilizan la PLN en función de sus necesidades.

2.2.6. Caso de usos reales

En esta sección se presentará una comparativa de los distintos enfoques y metodologías utilizados en algunos chatbots con personalización, con el objetivo de identificar cuáles son las más efectivas y cuáles son los desafíos que se presentan en este campo. A partir de esta comparativa, se pretende obtener una visión más completa de las oportunidades y desafíos que presenta la personalización de chatbots, y explorar posibles áreas de aplicación o líneas de investigación futuras en este campo.

MusicBot

MusicBot [41] es un chatbot diseñado para dar recomendaciones musicales basadas en técnicas de crítica. A diferencia de los sistemas tradicionales, este chatbot permite la interacción de manera conversacional, imitando una conversación en lenguaje natural.

- **Arquitectura**: Es un multi-modal que incorpora la conversación mediante texto y voz. Utiliza dos técnicas de crítica distintas; el usuario puede dar una crítica o recibir sugerencia del chatbot, de modo que da una experiencia de recomendación más personalizada. Para entender el lenguaje natural utiliza Dialogflow para procesar los textos y el reconocimiento de intenciones, y luego está integrado con Spotify API para brindar recomendaciones de música.
- **Metodología**: Emplea un método de investigación mixto entre cualitativo y cuantitativo. Incluye un estudio de usuarios para evaluar la experiencia del usuario con las recomendaciones generadas por diferentes técnicas de crítica. El estudio utiliza una técnica de crítica y mide varias métricas como la coincidencia de preferencia entre usuarios para obtener una personalización más interesante.
- **Modos de interacción**: Como he comentado en arquitctura, admite la interacción mediante texto y voz.
- **Interacción con los datos**: Interactúa con los datos de Spotify API. Utiliza la API para generar recomendaciones basadas en las características de la canción (artista, géneros) y las preferencias del usuario (artista, géneros, canciones).

CRS Culinario

Este chatbot [42] de recomendaciones culinarios es un tipo de SR basado en conocimiento. Utilizan este enfoque porque es capaz de solucionar el problema de

cold start en los SR basado en el Filtrado Colaborativo. Este chatbot aprovecha la ubicación del usuario para generar recomendaciones, ya que dependiendo del lugar, la comida típica que se suele comer es diferente.

La implementación del chatbot se hizo en la plataforma de teléfonos móviles, más en concreto, en la aplicación de Telegram. Gracias al crecimiento en la popularidad en los teléfonos inteligentes en la última década, con esta elección pueden alcanzar a muchos más usuarios.

- **Arquitectura**: El sistema se basa en una arquitectura de chatbot basada en reglas que utiliza un enfoque de aprendizaje profundo para mejorar la comprensión del lenguaje natural y la generación de respuestas coherentes y útiles para el usuario. El chatbot se integra con el sistema de recomendación de recetas basado en filtrado colaborativo, que utiliza un algoritmo de factorización matricial para predecir la valoración que un usuario daría a una receta determinada. El sistema también se diseñó para adaptarse al contexto del usuario utilizando un enfoque de aprendizaje por refuerzo.
- **Metodología**: Utiliza una combinación de técnicas de procesamiento de lenguaje natural y filtrado colaborativo para proporcionar recomendaciones personalizadas a los usuarios. Recopila las preferencias de los usuarios a través de preguntas explícitas o retroalimentación del usuario. El sistema interactúa con los usuarios a través de diálogos de texto y botones. Incorpora un modelo de estilo formulario, donde el usuario debe responder a una serie de preguntas para alcanzar los objetivos deseados.
- **Modos de interacción**: Se comunica con los usuarios a través de mensajes de texto.
- **Interacción con los datos**: El sistema utiliza un conjunto de datos relacionados al usuario y las características del lugar (preferencia de comida, ubicación, entre otros). Estos datos los han recolectado mediante web scrapping en diferentes plataformas como Traveloka y PergiKuliner.

MOOCBuddy

MOOCBuddy [43] es un SR de Cursos en Línea Masivos y Abiertos (Massive Open Online Course) que funciona en la plataforma de Facebook Messenger. Utiliza el perfil de redes sociales e intereses del usuario para ofrecer recomendaciones personalizadas de cursos en línea. Su objetivo es facilitar la búsqueda de cursos en línea abre nuevas oportunidades para el aprendizaje personalizado.

- **Arquitectura**: Es un chatbot implementado en Messenger de Facebook. Contiene mensajes estructurados con múltiples burbujas que representan como una lista de la interfaz de usuario. Estos mensajes contienen información, imágenes y botones.

- **Metodología**: Utiliza características como temas, idioma, fecha de publicación, universidades, entre otros. que ofrecen los MOOCs para realizar las búsquedas y proporcionar recomendaciones personalizadas a los usuarios.
- **Modos de interacción**: MOOCBuddy interactúa con los usuarios a través de la plataforma Messenger de facebook, de modo que es mediante texto.
- **Interacción con los datos**: Tiene una base de datos actualizada de MOOCs rumanos para proporcionar recomendaciones y enlaces a búsquedas específicas en directorio de MOOCs. Los usuarios también pueden compartir los resultados que han obtenido de otras plataformas, por ejemplo las redes sociales. Pueden comentar sobre las búsquedas y al compartir, está recomendando el MOOC a sus amigos.

CRS para pacientes diabéticos

Este chatbot propuesto para paciente con diabetes [44] tiene objetivo de dar asistencia médica virtual para controlar la diabetes. Está basado en investigaciones anteriores, como el exitoso VPbot.

- **Arquitectura**: Este chatbot no se ha llegado a implementar, sino que se da un diseño arquitectónico para que el chatbot funcione como un médico virtual para diagnóstico básico de pacientes diabéticos. Básicamente consiste en que el paciente entra a la sesión y realiza una serie de preguntas y respuestas.
- **Metodología**: El proceso de la conversación se basa en una sesión de chat como he comentado en arquitectura. Estos interactúan a través de preguntas y respuestas en lenguaje natrual. Por último, el chatbot realiza una serie de preguntas secuenciales dependiendo de las respuestas dadas por el paciente, detectando palabras claves para llegar a dar un diagnóstico adecuado y dar los consejos necesarios para mejorar el diabetes.
- **Modos de interacción**: Interactua principalmente mediante texto.
- **Interacción con los datos**: Este bot detecta las palabras claves y las compara en una base de datos donde tiene unos patrones para ver si esta palabra esta relacionada con diabetes. También es capaz de reemplazar sinónimos en las respuestas del paciente utilizando una conjunto de datos donde tiene almacenado los sinónimos.

3. Herramientas utilizadas

En este capítulo explicaremos detalladamente el funcionamiento de las diversas herramientas que se han ido utilizando para la implementación del chatbot. En concreto veremos Dialogflow, como está estructurado este proyecto y las librerías necesarias.

3.1. Introducción

El proyecto se ha desarrollado en lenguaje Python. Consiste en una parte frontend que contiene una página web con las instrucciones brevemente de lo que es capaz de hacer el chatbot y una parte back-end que realiza un registro del historial de la conversación y, por último la recomendación.

Respecto la estructura del proyecto (véase Figura 6), tiene una estructa relativamente simple. La aplicación está

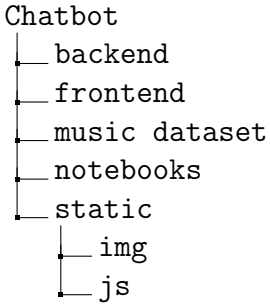

Figura 6: Estructura del proyecto

3.2. Flask

Flask [37] es un framework web de Python ligero y flexible que se utiliza para construir aplicaciones web. Dada la naturaleza de la aplicación, pienso que no es realmente necesario profundizarme tanto en el desarrollo del front-end y el back-end.

Por otro lado, para hacer que la aplicación sea visible a través de internet he utilizado la librería ngrok [46] y, además, un servidor remoto de Linux para que la página web pueda estar en funcionamiento todos los días.

Debido a su tamaño reducido y su simplicidad y que solo necesito tener las funcionalidades esenciales de la aplicación web (algunos textos y ventana del chatbot). Además, permite enrutamiento, por lo que es posible definir rutas y asociar funciones a estas rutas, permitiendo gestión de solicitudes y respuestas de HTTP, lo cual se necesitan para conectar la aplicación con Dialogflow.

3.3. Dialogflow

Dialogflow [2] es una plataforma de desarrollo de chatbots y agentes virtuales basada en IA. Se utiliza ampliamente para crear aplicaciones de conversación interactiva en diversos plataformas de mensajería.

La razón por la que escogí Dialogflow como herramienta principal es porque ya viene integrado con la capacidad de comprender y responder a las preguntas de los usuarios en lenguaje natural, utilizando PLN mencionado anteriormente y aprendizaje automático para analizar entradas de texto o voz. Además, es posible integrar el chatbot en aplicaciones externas mediante API sin necesidad de realizar código.

Por último, la plataforma está muy bien documentada con ejemplos de codificación, por lo cual es fácil poder guiarse entre sus páginas.

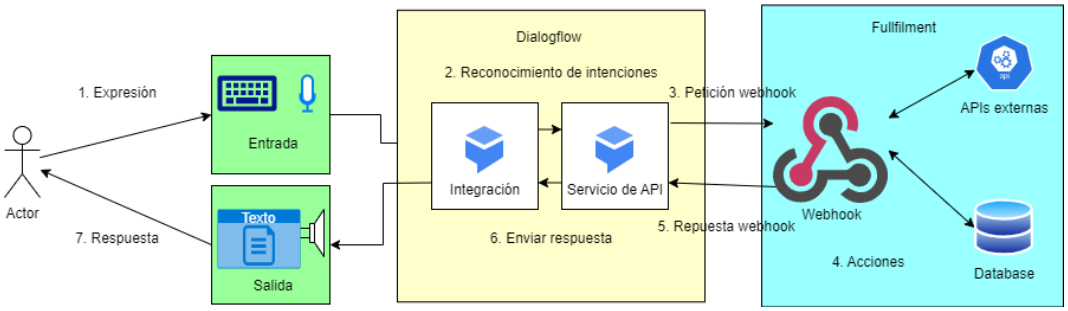

Figura 7: Arquitectura general de Dialogflow

En la Figura 7 podemos ver el flujo general de como sería el flujo en la arquitectura de Dialogflow cuando un usuario envía una pregunta:

1. **Expresión:** primero el usuario empieza introduciendo un texto de entrada.
2. **Reconocimiento de intenciones:** desde Dialogflow el chatbot intenta reconocer la intención del usuario para saber qué es lo que quiere.
3. **Petición webhook:** una vez que el chatbot ha reconocido la intención del usuario, se procede a realizar una petición al webhook que puede estar conectado a un base de datos y a unas APIs externas y le envía la solicitud del usuario al webhook.
4. **Acciones:** cuando webhook ha conseguido enviar correctamente la intención y procede a realizar la acción para esa intención. Puede ser por ejemplo en el caso de que un usuario salude al chatbot, este entiende que le ha saludado, y desde procede a devolver la acción que en este caso sería saludarle de vuelta.
5. **Respuesta webhook:** una vez que se ha decido la acción a realizar, se envía el contenido de la acción (p.e. texto de saludo "Hola!") a Dialogflow.
6. **Enviar respuesta:** cuando la API de dialogflow recibe la acción enviada por webhook, este envía sea en formato texto o voz al usuario.

7. **Respuesta:** por último, el usuario recibe la acción de salida que recibe y puede leer su contenido.

3.3.1. Intentos

Los intentos establecen una conexión entre la entrada del usuario y la acción que el bot debe realizar. Cada acción requiere la creación de un intento que se activará cuando se reciban entradas del usuario. En casos de intentos más complejos, se da una gran importancia al uso de eventos y, sobre todo, al uso de contextos.

Los contextos representan el estado actual de la petición del usuario. Por ejemplo, si un usuario pregunta por música de un género en concreto y durante la conversación menciona "Me recomiendas una canción del género Pop?", el chatbot puede entender que el usuario desea una recomendación de canción debido al contexto establecido previamente en el intento anterior. Cada intento puede tener tantos contextos de entrada como contextos de salida. Los contextos de entrada son necesarios para activar el intento deseado, mientras que los contextos de salida son nuevos contextos que aparecen después de activar un intento. Se requiere que la frase de entrada y el contexto coincida para poder activar un intento en concreto.

Cuando un usuario inicia la conversación, se puede solicitar un contexto y si se asigna a un intento, este iniciará la acción asignada. En la Figura 8 se muestra un ejemplo (extraído de la documentación de dialogflow).

Figura 8: Ejemplo de intent

Un intentobásico está compuesto principalmente de los siguientes elementos:

- **Frases de entrenamiento**: son frases de ejemplo que un usuario podría decir y que sirven como entrada de texto. La plataforma incorpora aprendizaje automático para entrenar los intents con frases similares a los del entrenamiento para así expandir la lista de frases similares.
- **Acción**: cuando un intentocoincide, puedes indicar al chatbot que realice las acciones definidas.
- **Parámetros**: cuando un intentocoincide, se genera el parámetro llamado entidad. Esto dicta como se extraen los datos.
- **Respuestas**: se puede definir una respuesta estática dentro del intentopara mostrar al usuario una vez que se haya activado el intent.

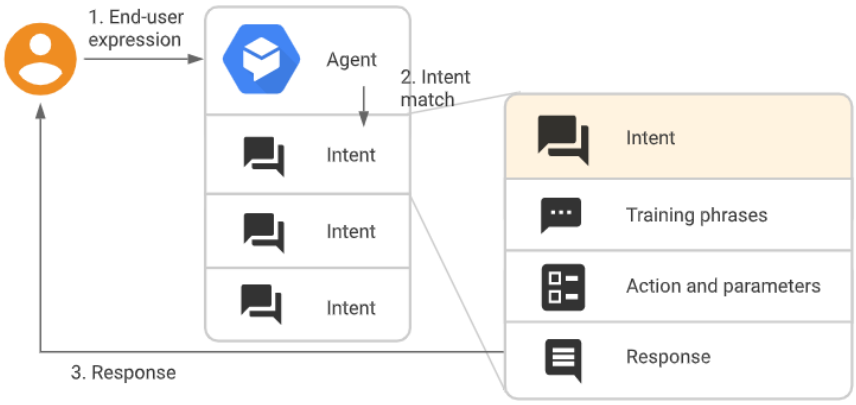

Figura 9: Flujo básico de intents

Aparte de las respuestas estáticas mencionadas, un intentopuede tener un "fulfillment" para proporcionar una respuesta más dinámica. Esto significa que cuando se activa el intent, la información es enviada mediante Webhook [35], y desde allí se trata la información recibida y devuelve una respuesta personalizada según los datos recibidos. La información que envía a través de Webhook incluyen información de la sesión de conversación, y los parámetros y contextos que recopila el chatbot.

3.3.2. Entidades

Las entidades son un mecanismo para extraer parámetros de las entradas de texto proporcionado por los usuarios. Cada parámetro tiene un tipo, como tipo de entidad, que determina de forma precisa en que se extraen los datos de una expresión. Véase Figura 9. En este podemos ver que las frases de entrada están marcados de diferentes colores, y en el paso final se extraen los parámetros relacionados con el tiempo y localización.

En Dialogflow se puede tener múltiples entidades, pero tiene una limitación de 30.000 entidades únicas, por lo que podría suponer un problema para algunos chatbots. Al tener esta limitación significa que no podremos añadir tantos nombre de artistas como deseemos o los títulos de canciones.

La ventaja de tener múltiples entidades es porque cada una de ellas puede tener sinónimos que ayudan a agrupar por el contexto que el usuario elija, y esto nos puede ayudar a entrenar las frases de entrenamiento.

La plataforma ya viene con algunas entidades preentrenadas como puede ser tiempo y localización vistos en Figura 9, aparte de estos también incorpora entidades de nombre de personas, geolocalización, lenguaje, entre otros. Debido a la naturaleza del proyecto, la de recomendar elementos en el ámbito de la música, finalmente no se han utilizado entidades preentrenadas, ya que lo que necesitamos son nombre de artistas y título de las canciones.

3.3.3. Integraciones

La plataforma permite la integración del chatbot en múltiples aplicaciones como se puede apreciar en la Figura 10. En concreto, he utilizado la integración de Dialogflow Messenger.

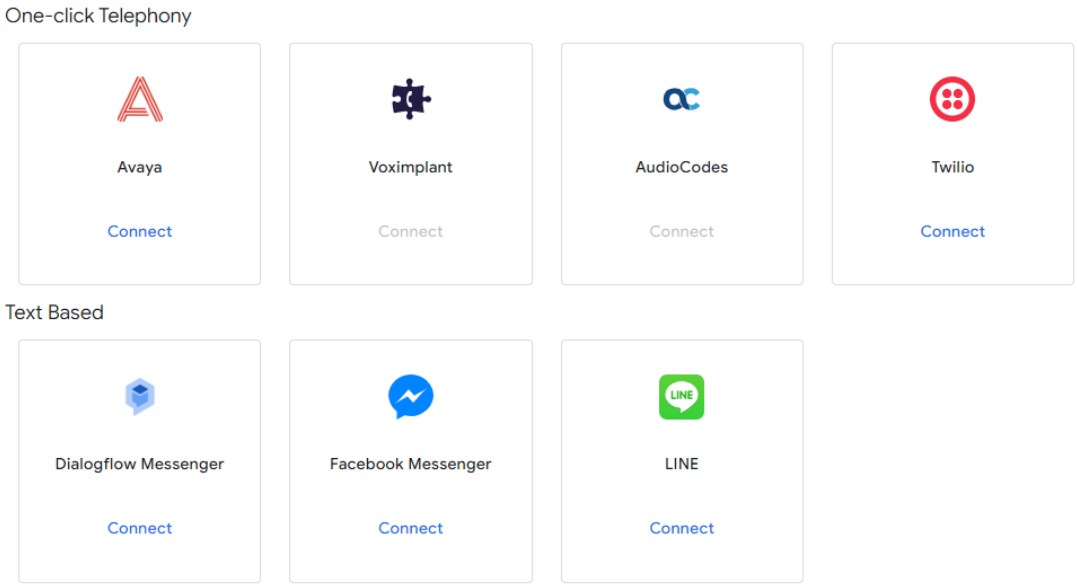

Figura 10: Integraciones disponibles en Dialogflow

3.4. Cornac

Cornac es un framework para SR multi-modal. Cornac ofrece implementaciones de varios algoritmos de recomendación populares, como algoritmos basados en filtrado colaborativos, filtrado basados en contenido, entre otros. También tiene la funcionalidad de crear sistemas híbridos combinando diferentes algoritmos del mismo framework.

4. Análisis previo

En el desarrollo de un chatbot es fundamental realizar un análisis previos del dataset que se utilizará como base de conocimiento. Este análisis tiene como objetivo explorar y comprender en profundidad los datos disponibles, con el fin de obtener información relevante.

El dataset utilizado en este proyecto fue proporcionado por mi tutora Dra. Maria Salamó. Los datos provienen de una plataforma web que proporciona servicios de música en línea y los datos recopilados va desde octubre de 2002 hasta diciembre de 2022.

4.1. Dataset

Al explorar el dataset, es importante examinar las diferentes columnas o atributos disponibles, así como comprender el significado y la naturaleza de cada uno de ellos. Esto ayuda a establecer una base sólida de conocimiento sobre el dominio del chatbot y definir las áreas en las que el chatbot puede brindar información o asistencia. A continuación se muestra las cuatro diferentes tablas existentes, y solo mostraré algunos atributos que pueden ser de interés para este proyecto:

- **users**: tiene 31.013 usuarios y 6 columnas. Los usuarios están anominizados, pero con la información demográfica.
  - user_id: *Integer* con un identificador único para cada usuario.
  - country: *String* el país donde reside el usuario.
  - gender: *String*.
  - playcount: *Integer* representa el número de veces que ha escuchado las canciones en la plataforma.
  - registered: *Integer* la fecha en la que se registró. Está en formato POSIX time que es la cantidad de segundos transcurrido desde 1 de enero de 1970.
  - continent: *String* el continente donde reside el usuario.
- **artists**: tiene 36.541 artistas y 11 columnas.
  - artist_id: *Integer* con un identificador único para cada artista.
  - name: *String* nombre artístico.
  - gender: *String*.
  - origin: *String* el lugar de nacimiento del artista.
  - listeners: *Integer* cantidad de personas que han escuchado la música de este artista.
  - playcount: *Integer* cantidad de reproducciones de sus canciones.

- url: *String* enlace a la su perfil en la plataforma.
- bio_summary: *String*.
- similar_artists: *String* listado de artistas similares.
- country: *String* país de origen.
- category: *String* géner de música que produce.

- **tracks**: tiene 646.155 canciones y 5 columnas.
  - track_id: *Integer* con un identificador único para cada canción.
  - name: *String* título de la canción.
  - artist_id: *Integer* identificador del artista.
  - duration: *Integer* duración en segundos.
  - url: *String* enlace para escuchar la canción.
  - category: *String* género de la canción.

- **ratings**: tiene 3.331.462 calificaciones y 4 columnas.
  - user_id: *Integer*
  - track_id: *Integer*
  - rating: *Integer* calificación, que puede oscilar entre [1, 5]
  - playcount: *String* número de veces que el usuario ha escuchado esta canción.

4.1.1. Exploración de los datos

En la exploración de datos, llevaré a cabo una serie de pasos y técnicas para analizar en detalle los datos que tenemos a disponibilidad.

- **Visualización de datos:** visualizar con gráficos para identificar tendencias.
- **Identificación de valores faltantes o inconsistentes:** revisaré si hay datos que faltan o valores inconsistentes en el dataset y tomaré decisiones sobre cómo manejarlos. Esto es importante para garantizar la integridad de los datos y evitar posibles sesgos o errores.

En el primer caso, uno de los atributos que podría ser relevante es la de categoría música. La categoría musical es una forma común de categorizar y clasificar la música según sus características distintivas, como el estilo, la instrumentación y la temática. En la tabla de canciones existen un total de 19 categorías y, a continuación miraremos las distribuciones de las 10 más populares (véase Figura 11).

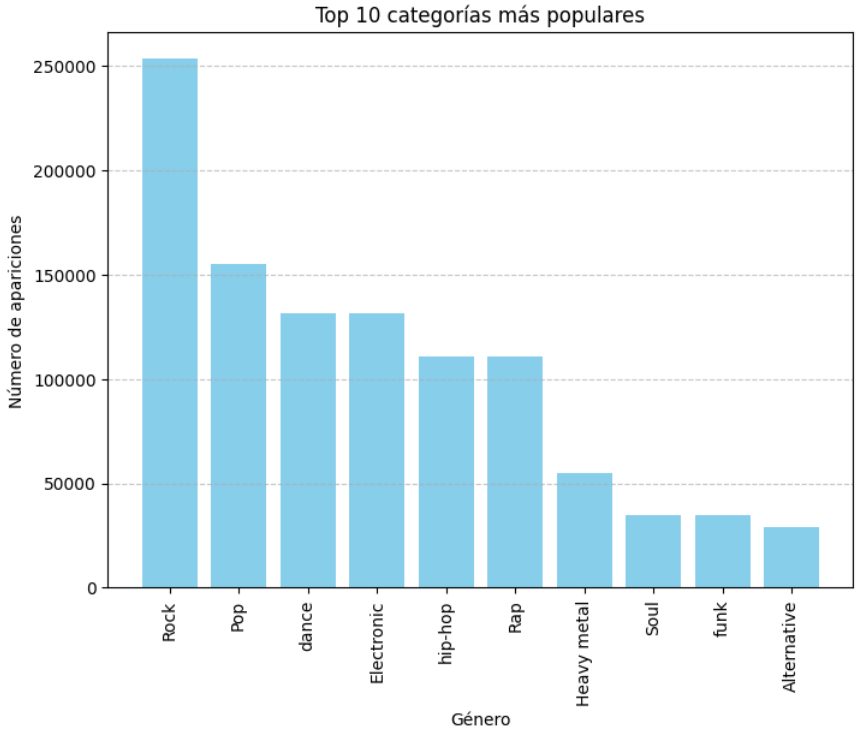

Figura 11: Top 10 categorías musicales

Para el caso de valores faltantes o inconsistentes, se ha avistado los siguientes problemas:

- **Tabla de Artistas:** hay artistas con nombres duplicados, para solucionar esto he buscado todos los artistas con el nombre duplicado y eliminando el valor duplicado.
- **Tabla de Canciones:** hay valores nulos en el atributo del título. Sin el título no podremos recomendar una canción al usuario. Otro de los problemas que hay está relacionado con el problema de artistas con nombres duplicados, y es que al haber nombres de artistas duplicados, también había otro identificador único.
- **Tabla de Usuarios:** en esta tabla únicamente hay usuarios duplicados.
- **Tabla de Valoraciones:** En este se ha observado que hay muchas valoraciones duplicadas, donde el par $(user\_id, track\_id)$ con valores iguales aparecen múltiples veces.

Una vez identificado todos los problemas (valores nulos y duplicados) podemos eliminar estas filas, dando de ejemplo la tabla de usuarios:

```
# Encontramos los índices de usuarios repetidos y los eliminamos
duplicated_users = users[users.index.duplicated()].index
users.drop(duplicated_users, inplace=True)
```

5. Implementación

En este capítulo se detalla el proceso de creación de mi chatbot después de haber terminado todos los capítulos anteriores. Veremos las especificaciones que el chatbot ha de completar, como se relaciona el chatbot con la base de datos que tenemos.

5.1. Desarrollo del chatbot

En esta sección se detalla el proceso que he seguido para tener la parte de chatbot en funcionamiento. Para ser considerado funcional, ha de poder atender a las peticiones de los usuarios. A continuación se lista una serie de objetivos:

- proporcionar la bibliografía de un artista.
- proporcionar información sobre artistas relacionados a una canción.
- recomendar una canción popular a partir de una categoría.
- recomendar una canción popular de cualquier categoría.
- recomendar una canción personalizada para el usuario, es decir, que el usuario le pedirá una recomendación y el item (canción) ha de ser relevante y de su interés.

5.1.1. Flujo conversacional

La primera etapa de definición de la conversación es de alta importancia, ya que es como la plantilla que se seguirá para la implementación de nuestro chatbot. En la definición conceptual se deben de definir las intenciones que podría tener el usuario. Además, en cada intención, dada la entrada del usuario, también deberemos de establecer el comportamiento.

A continuación se muestra el diseño de la conversación en Figura 12. Podemos ver que tiene unas casillas de colores, aquí se detallan su significado:

- **Casilla verde**: son las casillas de acción del chabot. Estas acciones son lanzadas como respuesta a una interacción de usuario.
- **Casilla amarilla**: son las entradas del usuario para el chatbot. Puede ser para especificar una tarea de recomendación a realizar, u otra funcionalidad.
- **Casilla negra**: chatbot no entiende la entrada de texto del usuario. En caso de llegar aquí desde la casilla azul, representa que el chatbot ha entendido la intención, pero no es capaz de recomendarle algo. Como en el caso de un artista, es posible que entienda que la intención sea la de "háblame sobre Harry Styles", pero si en la base de datos no existe este artista, entonces no podrá proporcionarle la información deseada.

- **Casilla azul**: chatbot entiende la entrada del usuario y entra en la intención. Aquí el chatbot realizará la acción vinculada a la intención del usuario, que puede ser recomendar una canción, o pedir información de un artista.
- **Casilla roja**: aquí el chatbot no realiza acciones hasta que el usuario vuelva a interactuar.

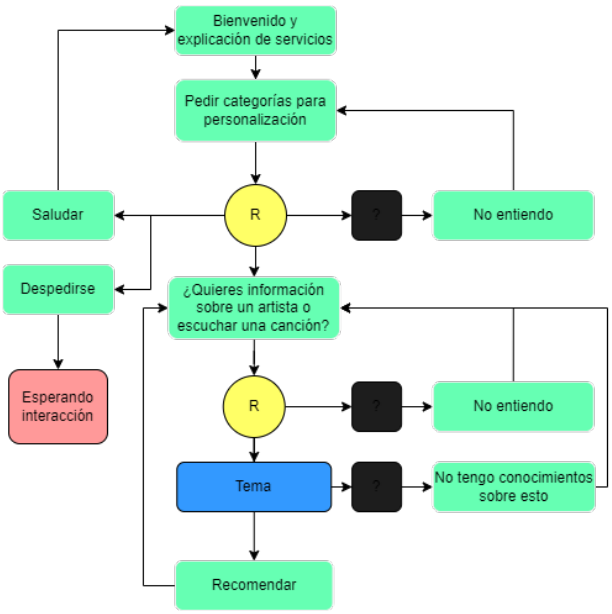

Figura 12: Diagrama del diseño de la conversación

Cogiendo el diagrama de conversación anterior, un flujo normal de conversación podría ser lo siguiente:

1. Usuario: Hola!
2. Chatbot: ¡Hola! Soy tu asistente musical. Puedo proporcionarte información sobre artistas, recomendar canciones y mucho más. Si tienes alguna pregunta específica o quieres conocer más sobre algún artista en particular, ¡no dudes en preguntar!
3. Chatbot: ¿Tienes alguna preferencia en categoría de música?
4. Usuario: Me encantan los géneros rock, pop y classic.
5. Chatbot: ¿Cómo puedo ser de ayuda?
6. Usuario: Conoces a Mariah Carey?
7. Chatbot: Voy a encontrar la información de ese artista para ti.
8. Chatbot: "devuelve biografía de artista".

9. Chatbot: ¿Deseas saber algo mas?

En la conversación anterior podemos observar el flujo de inicio a fin para preguntar la biografía de un artista. Los intentos detectados cada vez que el usuario escribe al chatbot serían los siguientes:

1. Intención saludo detectado
2. Intención get_genres detectado (guardar preferencias del usuario)
3. Itención artist_bio detectad (pide biografía de un artista)

Según este diseño, cubre todas las necesidades que tenemos para el chatbot, y es capaz de superar los objetivos de desarrollo planteados al inicio de este capítulo.

5.1.2. Creación de intentos

Antes de empezar a replicar el diseño planteado, primero necesitamos definir las casillas de intenciones. Como he comentario en la sección anterior es necesario configurar parámetros esenciales para que un intentopueda funcionar como es debido.

Intento de Información

Cogeremos de ejemplo el intento que pregunta al usuario si quiere recomendación de una canción, ya sea por categoría, o similitud entre usuarios. Lo primero que hemos de configurar es el fulfillment. Dentro de un intento podemos definir respuestas estáticas que se activarán en caso de que el chatbot llegue a entender la entrada del usuario. Véase Figura 13.

Aquí podemos ver que el agente está configurado para dar 5 tipos de respuestas diferentes. Todos ellos tienen el mismo propósito, preguntar al usuario lo que desea hacer. La decisión de hacer múltiples respuestas estáticas en los intentos es para dar más variedad al usuario, ya que puede ser sofocante si la respuesta siempre es la misma.

Como se ha mencionado al inicio de la capítulo, dentro de los intentos podemos utilizar servicios de webhook para poder generar una respuesta más dinámica. Para dotar a un intento de webhook, simplemente hay que ir a la sección de webhooks dentro de la consola de Dialogflow.

En la Figura 14 se puede observar que hemos creado un webhook para los artistas y hay definido un webhook URL. Este URL es donde se enviarán un log del contenido del chat, como pueden ser los parámetros requeridos del intento, el contexto y otras informaciones que pueden ser relevantes para llevar a cabo la acción de dar biografía de un artista.

Los parámetros y rutas, otras de las múltiples funcionalidades que se le puede añadir a un intento. Los parámetros son “variables“ que el intento puede recibir dada

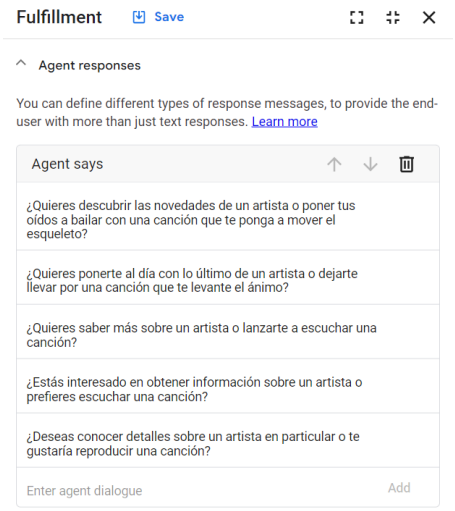

Figura 13: diagrama del diseño de la conversación

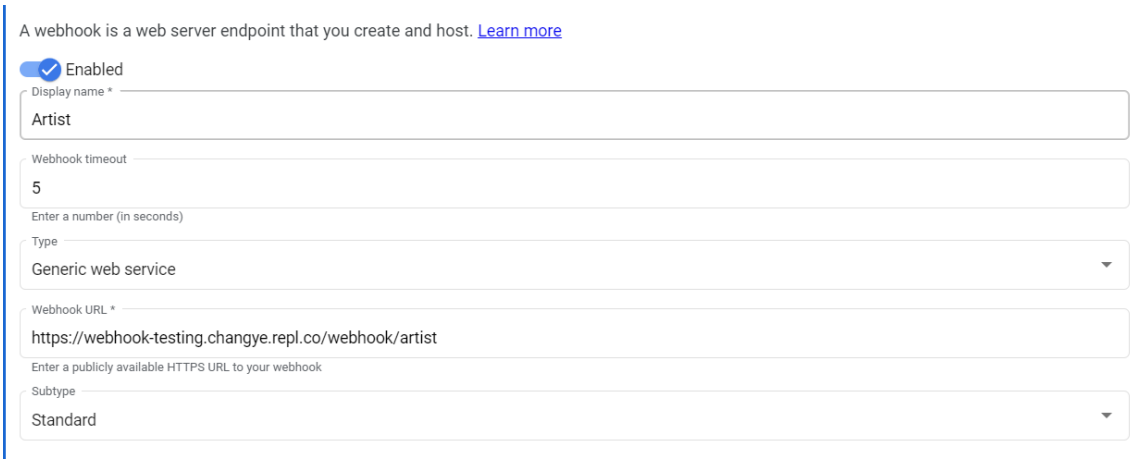

Figura 14: Parámetros necesarios de un webhook

una entrada del usuario. Estos parámetros pueden ser opcionales u obligatorios, en caso de que sean obligatorios, el bot no realizará la acción del intento hasta obtener los parámetros. En el caso de Rutas, es la acción que realizará. Por lo tanto, para que la ejecución del intento sea exitoso, el usuario ha de realizar los siguientes pasos:

1. Enviar un saludo al chatbot
2. Introducir texto de entrada para definir las categorías de música preferidas.
3. Ahora en este paso puede elegir entre los siguientes tipo intentos; pedir biografía de un artista, pedir recomendación de una canción a partir de una categoría de música, preguntar por una canción (para ver artistas relacionados con esta canción), una canción personalizada a sus gustos.

En el caso donde el usuario no cumple alguno de los pasos, se activa el evento de “No tengo conocimiento de esto“ y el usuario tendrá que volver a introducir la frase en entrada. Por ejemplo en el primero paso el chatbot espera el saludo, si le da una entrada diferente de saludo en el paso uno se activará el intento de que no lo entiende.

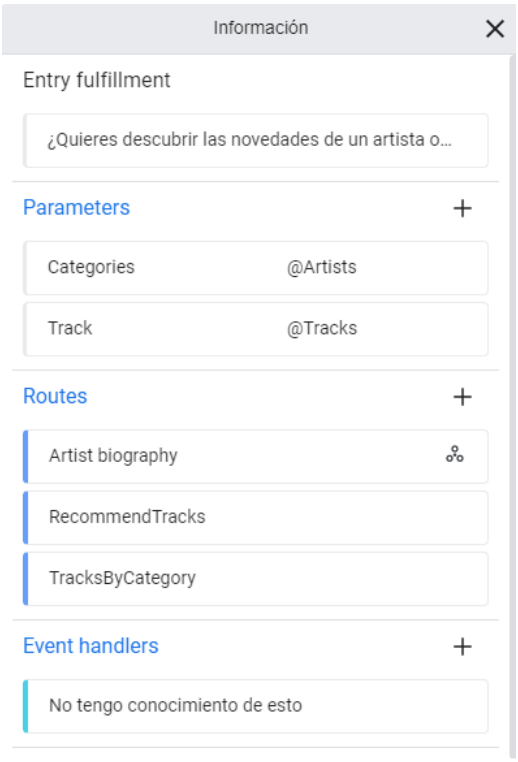

Figura 15: Vista general de un Intento

5.1.3. Entrenamiento de frases

Hasta ahora hemos creado un intento y las posibles respuestas que puede devolver, sea estática o dinámica, pero para que el chatbot pueda reconocer las intenciones del usuario, tenemos que añadir frases de entrenamiento.

Las frases de entrenamiento reflejan las posibles formas en las que los usuarios pueden expresar esa intención en el lenguaje natural. Estas frases tienen que ser expresadas de manera diferente, pero todos con el mismo significado.

En Figura 16 tenemos algunas de las posibles frases que el usuario podría solicitar a la hora de pedir información del artista. Podemos ver que las frases contienen un texto marcado en azul, este campo es la entidad. En este caso, tenemos la entidad Artista, dentro de esta entidad se guardan nombres de los artistas.

Como se ha visto a lo largo del proyecto, Dialogflow tiene la capacidad aprender de estas frases mediante PLN, esto quiere decir que si le damos unas pocas frases donde se indique claramente la intención, y luego la entidad, a partir de esto,

Dialogflow es capaz de generar frases de entrenamiento similares.

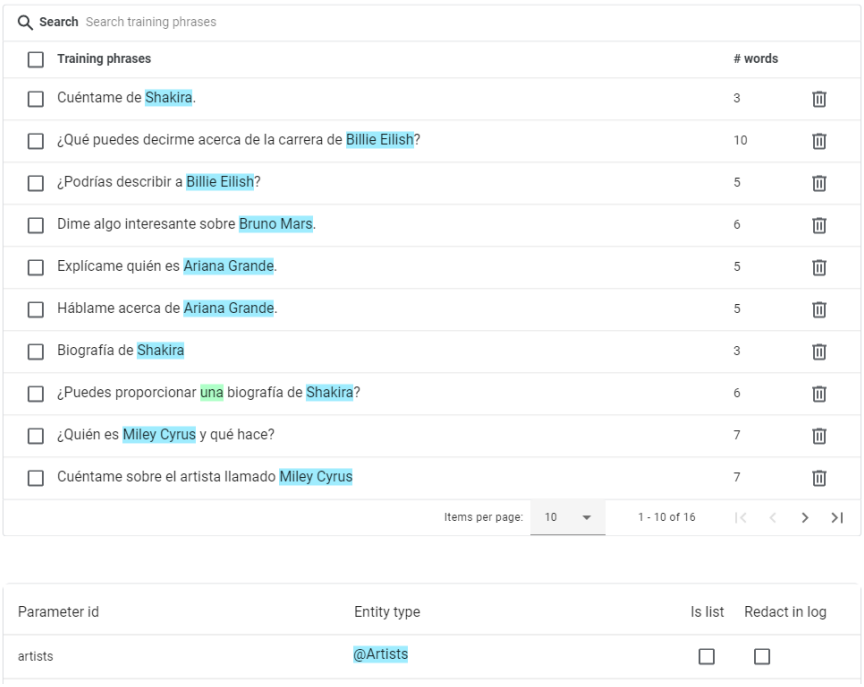

Figura 16: Frases de entrenamiento del intento Artist

A la hora de entrenar las frases de entrenamiento, existe un parámetro llamado "Classfication Threshold". El chatbot recoge la entrada del usuario y calculará mediante algoritmos de coincidencia gramaticales y dará una puntuación.

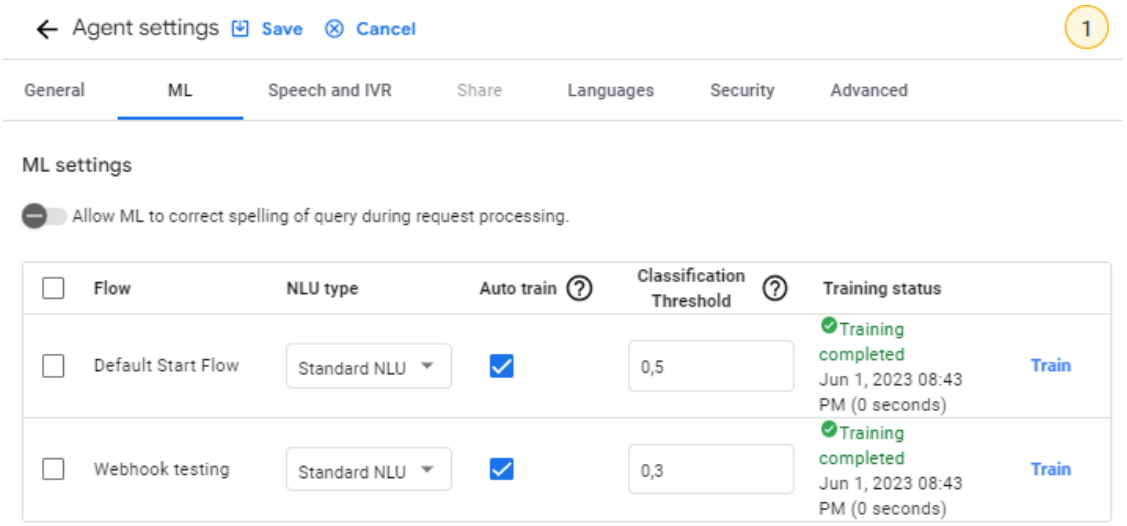

Figura 17: Confianza de detección de intentos

En la Figura 17 se puede observar que la confianza de detección para el agente predeterminado es de un 0,5. Estos valores van desde 0 (incierto) hasta 1 (cierto) y

una vez que recibe la puntuación, procede a comparar toda la puntuación de todos los intentos y pueden ocurrir los siguientes resultados:

- El intento con la puntuación más alta y a la vez mayor que 0,5 entonces muestra una coincidencia.
- Si ningún intento alcanza el umbral de 0,5 nos llevará a los intents de "no entiendo" vistos en la Figura 12.

5.1.4. Entidades

Dado que tenemos 36.541 artistas en la tabla de usuarios, para poder importar esto a Dialogflow he tenido que crear un JSON (JavaScript Object Notation), un formato ligero de intercambio de datos, a partir del código de Python. Los pasos para la extracción de nombre artistas hasta importarlos en Dialofglow son los siguientes:

1. Leer el contenido de la tabla utilizando la librería pandas.DataFrame de Python.
2. Exportar la columna de nombres con el formato correcto.

```
{"entities": [
    {"value": "Harry Styles",
    "synonyms": ["Harry Styles"],
    "languageCode": "es"
    },
    {"value": "Lil Nas X",
    "synonyms": ["Lil Nas X"],
    "languageCode": "es"
    }
]}
```

   Es posible introducir sinónimos para las entidades, por tanto, si se añade más variaciones de sinónimos, hay mejor posibilidad de detectar la entidad en caso de que el usuario introduzca mal el nombre. Sin embargo, dada la cantidad de artistas que hay en la base de datos, no es factible buscar y añadir sinónimos de todos los artistas.

3. Importarlo desde la consola de Dialogflow el JSON y añadirlo como una entidad nueva.

Una vez que tenemos las entidades definidas nuestro con los intents y frase de entrenamiento, nuestro chatbot ya es funcional.

5.1.5. Front-end

Como se ha mencionado en el capítulo anterior de análisos, la libería que uso en esta aplicación es Flask.

La página principal consta de un mensaje donde se da instrucciones al usuario de como proceder para comenzar una conversación con el chatbot y la burbuja del chat.

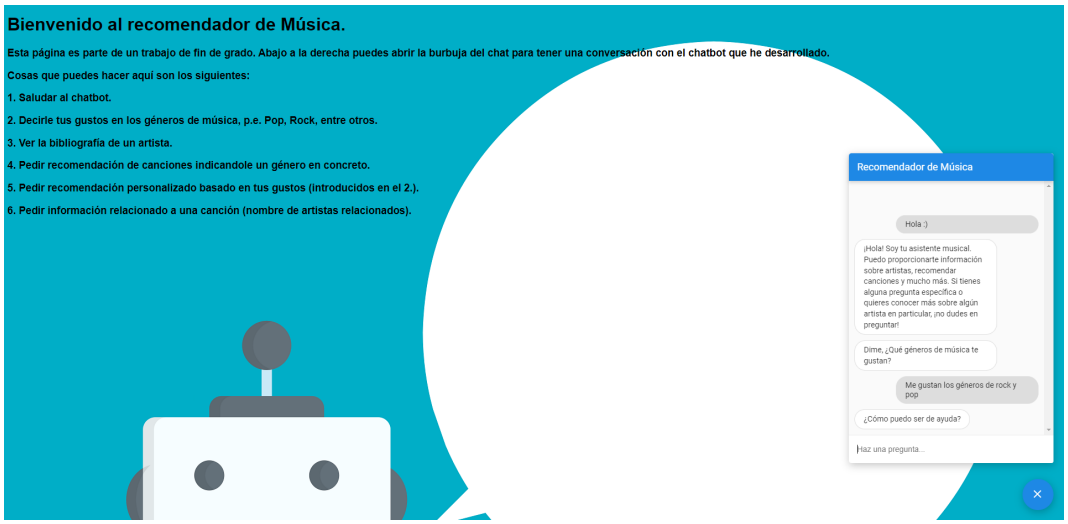

Figura 18: Front-end de la aplicación

Aparte de esto, desde front-end también he creado un JavaScript (JS) y consiste en hacer web scrapping de la página para extraer información relevante de la conversación, como puede ser la id de sesión, que es un identificador único por cada conversación. Con este JS lo que pretendo es crear un historial de conversaciones y guardarlos en la base de datos.

Ahora explicaré cómo poder hacer el web scrapping de la aplicación y guardar toda esa información en la base de datos. Si miramos a la Figura 18, podemos ver que en la página de front-end tenemos la burbuja de chat. Cuando Dialogflow nos envía el mensaje a la página del front, recibimos un conjunto de datos, incluyendo los parámetros. Este conjunto está estructurado de la siguiente forma:

```
{"queryResult": {text: "hola", languageCode: "es", ...},
...,
diagnosticInfo: {
    ...
    "Alternative Matched Intents": [
   {
     "Score": 1,
     "DisplayName": "Default Welcome Intent",
     "Type": "NLU",
     "Id": "00000000-0000-0000-0000-000000000000",
```

```
      "Active": true
    },
    Session Id: "9b1dda-cfd-17a-cb8-873316776",
    ...
  ]
  },
  ...
  "responseMessages": [
  {
    "responseType": "HANDLER_PROMPT",
    "source": "VIRTUAL_AGENT",
    "text": {
      "redactedText": [
        "¡Hola!"
      ],
      "text": [
        "¡Hola!"
      ]
    }
  }
```

Ese conjunto de datos tiene el texto del último mensaje enviado por el usuario o el chatbot, además, también contiene información del intento actual y, por último, la id de la sesión. Este id es un número que se genera aleatoriamente cada vez que un usuario se conecta a nuestra página web. Entonces, para poder extraer estos datos, necesitamos interactuar con los elementos Document Object Model (DOM) mediante JS. El DOM es una representación en memoria de un documento HTML o XML. Este organiza los elementos del documento en una estructura de dato y nos permite acceder y manipular fácilmente los elementos individuales con JS.

```
window.addEventListener('dfMessengerLoaded', function (event) {
    const dfMessenger = document.querySelector('df-messenger');
    $sesionId = dfMessenger.getAttribute("session-id");
}
```

El código anterior agrega un event listener y espera a que la aplicación de Dialogflow Messenger (df-messenger) sea cargado, y una vez cargado ya podemos extraer los datos que queramos del chat, solo tenemos que acceder mediante la función $dfMessenger.getAttribute("atributodeseado")$. De la siguiente forma podemos capturar todo el flujo de la conversación.

La información que captamos mediante webscrapping se tiene que enviar al backend. Para poder enviar los datos he utilizado AJAX..

Podemos ver que lo que hace es básicamente enviar un JSON al url *historial*, que es donde se procesa los datos que se consiguen del historial de chat.

```
$.ajax({
      url: "/historial",
      method: "POST",
      data: JSON.stringify(user_data),
      dataType: "json", // Specify the response data type as JSON
      contentType: "application/json",
      success: function (response) {
        console.log(response)
      },
      error: function (xhr, status, error) {
        console.log(error);
      }
    });
```

En data contiene la información relevante sobre el usuario, pero también las del chatbot.

5.1.6. Back-end

Para back-end he utilizado un base de datos relacional, utilizando Flask-SQLAlchemy. La información que guardamos es relativamente simple, a continuación se muestra su estructura.

En la Tabla 2 podemos ver que solo contiene 4 columnas. Estos datos son los únicos que necesitamos para poder mantener un historial del chat, de la cual podemos extraer las preferencias del usuario. Un identificador se genera a partir de la " sesion_id" que es un valor único que se genera en cada conversación. El nombre del usuario es para determinar quién ha escrito el mensaje, de esta forma podemos recuperar todo lo necesario. Luego tenemos la fecha que indica la hora exacta de cuando se ha generado ese mensaje.

Es importante destacar que toda esta información se guarda con el objetivo que como trabajo futuro se use para incluir, además de un recomendador basado en filtrado colaborativo, un recomendador basado en contenido, el cual use el historial del chat para conocer las preferencias de los usuarios.

Columna	**Tipo de datos**	**Descripción**
sesion_id	VARCHAR	Identificador único de la sesión
usuario	BOOLEAN	0 = usuario, 1 = bot
texto	TEXT	Texto del mensaje
fecha	TIMESTAMP	Fecha y hora del mensaje

Cuadro 2: Estructura de la tabla Conversaciones

5.1.7. Recomendador

Para la implementación del recomendador he creado dos modelos; uno que emplea la técnica de la factorización matricial y el otro es de usuarios KNN. La decisión para esto es porque los usuarios que se conectan a la aplicación no tienen valoraciones previas, y eso puede causar el problema que he comentado de cold-start. Una solución simple que he intentado hacer es, a partir de las preferencias del usuario (las categorías de música), buscar a un usuario en la base de datos que sea similar al usuario que está utilizando el chatbot. Una vez que encuentra un usuario con preferencias similares, se procede a utilizar a buscar una canción que pueda ser de interés al otro usuario, ya al tener preferencias similares hay una alta probabilidad de que le guste.

Por tanto, el flujo a la hora de recomendar una canción a un usuario sin valoraciones previas sería el siguiente:

1. El usuario indica su preferencia en la música.
2. Se busca otro usuario que también le gusten ese tipo de música.
3. Encontrar canciones que les pueda gustar al perfil encontrado, y se recomienda la canción.

5.2. Cuestionarios

Para saber si los chatbots es un concepto que le interesan a las personas, sobre todo, la de recomendación musical, he decidido realizar dos tests.

- **Pre Test**: aunque la popularidad de los chatbots ha aumentado mucho, todavía hay gente que desconocen este término. Este cuestionario está destinado tener una visión general de cuantas personas conocen esta tecnología y si han tenido experiencia con ella.
- **Post Test**: en este cuestionario es para saber la satisfacción del usuario después de haber probado el chatbot desarrollado en este proyecto de fin de grado. También se harán algunas preguntas relacionadas con la eficiencia del chatbot.

Como podemos ver en las descripciones de los dos tests, uno está orientado a conocer cuál es el conocimiento de los encuestados sobre chatbos, mientras que el segundo es para ver la efectividad y el grado de satisfacción de los usuarios después de haber probado el chatbot.

En total han acudido 24 personas a la sesión de prueba. El cuestionario consiste en respuestas de dar un valor escalar de $[1, 5]$, consideraremos que $calificacion >= 4$ significa satisfecho, y otras de responder “Sí“ o “No“.

5.2.1. Pre Test

En este apartado se presentan los resultados obtenidos para cada pregunta.

- *¿Conoces sobre el concepto de chatbot?*
  - Respuestas: un 83,3 % de los usuarios conocen el concepto de chatbot.

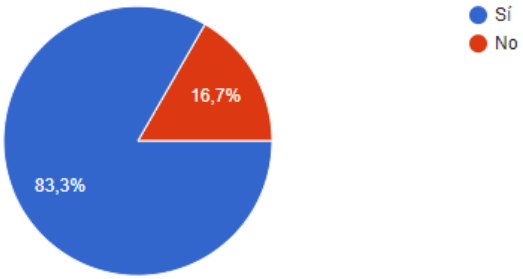

Figura 19: Porcentaje de usuarios de si conocían el concepto de chatbots

- *¿Has interactuado alguna vez con un chatbot?*
  - Respuestas: un 70,8 % de usuarios han interactuado con chatbots previamente.

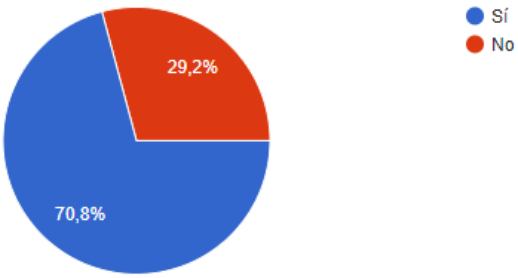

Figura 20: Porcentaje de usuarios de si han interactuado con un chatbot

- *¿Qué opinas de los chatbots?¿Crees que son interesantes?*
  - Respuestas: las personas que han calificado con 4 y 5 sí consideran que son interesantes, un 70,8 %, mientras que hay un 20,8 % que no están a favor ni en contra y sólo un 8,3 % no lo encuentran demasiado interesante.

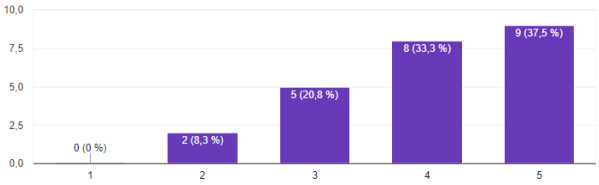

Figura 21: Calificación según el interés los chatbots

- *¿Te gusta la música?*
  - Respuestas: las personas que han calificado con 4 y 5 sí que les gustan la música, un 87,5 %, mientras que hay un 8,3 % que no están a favor ni en contra y sólo un 4,2 % no les gustan la música.

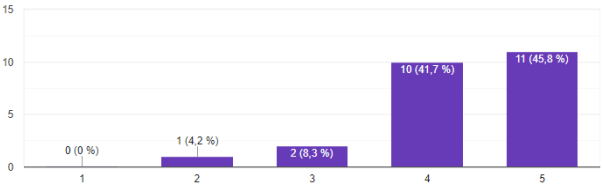

Figura 22: Calificación de la música

- *¿Dónde te diriges para descubrir música nueva?*
  - Respuestas: Un 50 % descubren en Youtube, 20,8 % en Spotify, 12,5 % Otros y, 8,3 % en amigos y Redes sociales

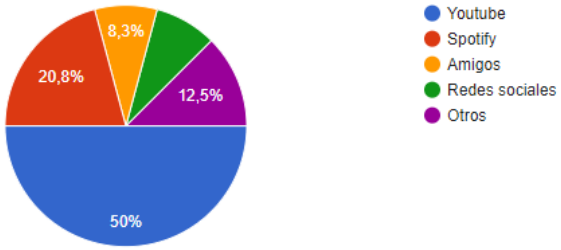

Figura 23: Porcentaje del lugar donde van a descubrir nuevas canciones

- *¿Qué te parece un chatbot que recomienda música?*
  - Respuestas: las personas que han calificado con 4 y 5 sí les parecen interesante la idea, un 75 %, mientras que un 20,8 % no están a favor ni en contra y sólo 4,2 % no les parecen muy interesante.

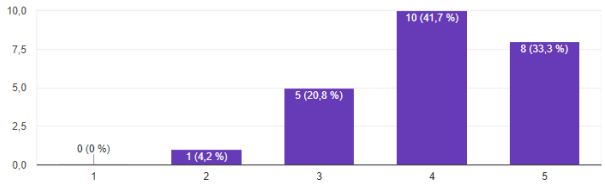

Figura 24: Calificación de si les interesa un chatbot que recomienda música

- *¿Te sientes cómodo interactuando con un chatbot en lugar de un ser humano?*
  - Respuestas: un 79,1 % de las personas que han calificado entre 4 y 5 no sienten indiferencia interactuar con un chatbot que un ser humano, mientras que un 12,5 % no están a favor ni en contra y sólo un 8,3 % se encuentran incómodos interactuar con un chatbot en lugar de un ser humano.

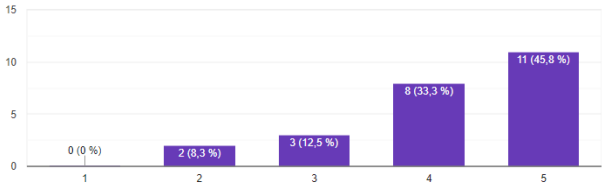

Figura 25: Calificación de si son cómodos interactuando con un chatbot

5.2.2. Post Test

En esta apartado se presentan los resultados obtenidos después de que los usuarios han probado el chatbot desarrollado en este proyecto fin de grado.

- *Me ha sido fácil hablar con el chatbot.*
  - Respuestas: un 83,3 % de usuarios han puntuado entre 4 y 5 indicando que no han tenido problemas para hablar con el chatbot, mientras que un 16,7 % han tenido algunos problemas.

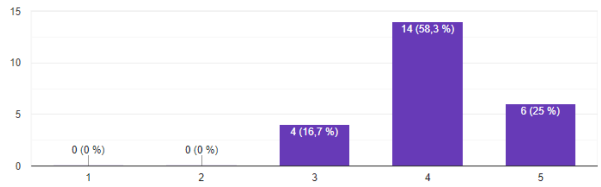

Figura 26: Calificación de si ha sido fácil conversar con el chatbot

- *Me ha gustado las canciones recomendadas.*
  - Respuestas: La mitad de los usuarios han puntuado entre 4 y 5 indicando que les ha gustado las canciones recomendadas. mientras que un 37,5 % les ha parecido regular y un 12,5 % no les ha gustado demasiado lo que se ha recomendado.

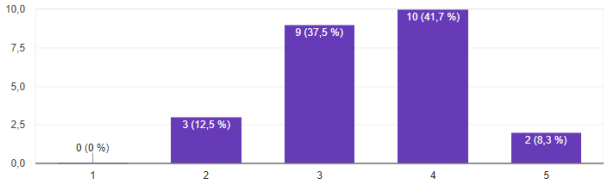

Figura 27: Calificación de si les ha gustado los las canciones recomendadas

- *Las canciones recomendadas son relevantes y adecuados.*
    - Respuestas: un 50 % de los usuarios han puntuado entre 4 y 5 creen que las canciones recomendadas coincide con sus preferencias, mientras que un 41,7 % creen que ha sido regular y sólo un 8,3 % creen que no encaja con sus preferencias.

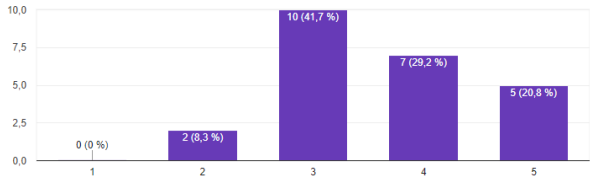

Figura 28: Calificación de si los elementos recomendados van encaja con sus gustos

- *He podido encontrar bibliografía de los artistas.*
    - Respuestas: un 66,7 % de usuarios han encontrado la bibliografía del artista buscado, mientras que un 33,3 % de usuarios no han sido capaces de encontrar lo que buscaban. Como bien se ha mencionado, Dialogflow tiene unas limitaciones y es que no se le puede añadir demasiadas entidades. Por lo que los usuarios que no han encontrado el artista se debe principalmente a esta limitación.

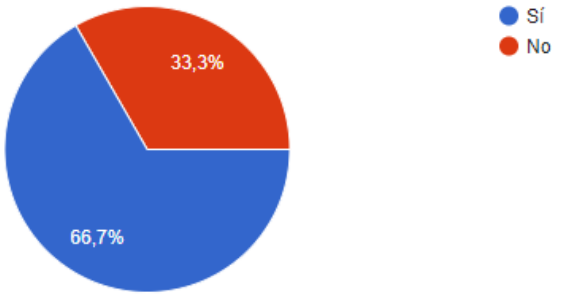

Figura 29: Porcentaje de usuarios que han encontrado bibliografía de los artistas

- *Estoy satisfecho con el chatbot*
    - Respuestas: un 45,8 % de los usuarios han calificado entre 4 y 5 indicando una experiencia satisfactoria con el chatbot, mientras que un 50 % creen que ha sido bastante regular y sólo un 4,2 % no están muy satisfechos.

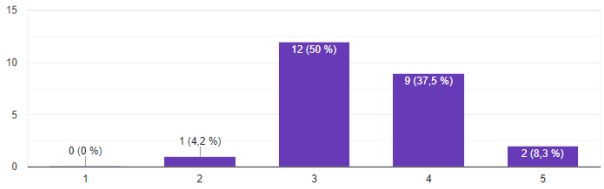

Figura 30: Calificación de satisfacción

- *¿Recomendarías el chatbot musical a otras personas interesadas en descubrir música?*
  - Respuestas: un 54,2 % de usuarios han calificado entre 4 y 5 indicando que sí recomendarían a otras personas el chatbot desarrollado en este proyecto fin de grado, mientras que un 37,5 % no están muy decididos y sólo un 8,3 % creen que no recomendarían.

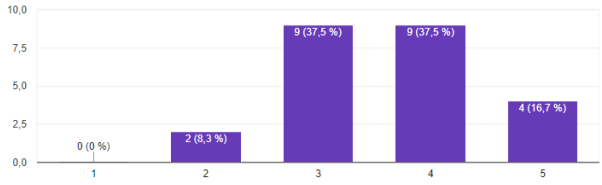

Figura 31: Calificación de si recomendarían a otras personas el chatbot

- *¿Qué mejoras te gustaría ver?*
  - Respuestas: un 37,5 % de usuarios quieren más funcionalidades, 33,3 % quieren mejor algoritmo de recomendación, 12,5 % quieren mejoras estéticas, 8,3 % otras mejoras, 4,2 % quieren más diálogos y 4,2 % creen que ya está bien implementada.

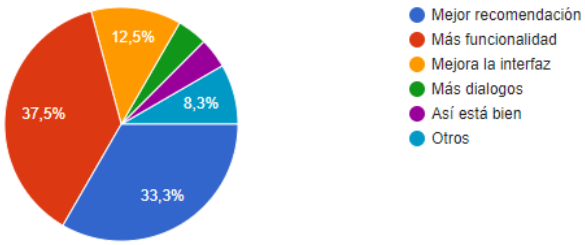

Figura 32: Posibles mejoras

5.3. Instrucción para ejecutar la aplicación

En esta última sección de la implementación se dará una pequeña guía de como hacer funcionar el chatbot en caso de querer probarlo.

Como muy bien se ha visto en el capítulo de Herramientras utilizadas, para otros usuarios se puedan conectar a la página del chatbot he utilizado ngrok. Esta librería genera una dirección URL aleatoria cada vez que se ejecuta, por tanto, es necesario modificar en Dialogflow los URL correspondientes de los webhook.

Por otro lado, todas las librerías que se ha utilizado en este proyecto fin de grado están en el fichero *requirements.txt* del código. Entonces será necesario instalar todas las librerías con el comando "pip install -r /path/to/requirements.txt". Una vez instalado las librerías correspondientes, es necesario ejecutar los siguientes comandos:

- flask db init
- flask db migrate -m Ïnitial migration"
- flask db upgrade

Estos comandos son utilizados para levantar la base de datos y por último en el código del proyecto también hay una carpeta que se llama "dialogflow agent". Este es el agente de dialogflow. Para poder utilizarlo será necesario registrarse una cuenta en Dialogflow CX, una vez registrado se puede crear un agente nuevo en su proyecto de prueba y hay una opción que es para importar agente a partir de archivos JSON.

6. Conclusiones

Este último capítulo está dividido en dos partes. En primer lugar, se explicarán las conclusiones del proyecto y evaluaré si se han completado los objetivos propuestos. En segundo lugar, hablaré de las posibles mejoras que se podrían llevar a cabo si se continúa mejorando el chatbot musical.

6.1. Conclusiones del proyecto

De los objetivos planteados al inicio de este trabajo de fin de grado, se han logrado terminar los siguientes objetivos:

- Haber desarrollado el chatbot con todas las funcionalidades planteadas de un inicio, tenemos un chatbot con una base de datos capaz de recomendar productos dependiendo de las entradas del usuario.
- Sesión de pruebas con usuarios reales para determinar la efectividad del chatbot.

6.2. Trabajo futuro

Hay varias mejoras que se pueden expandir en un futuro:

- Recuperación del historial de chat: actualmente sólo se guarda la conversación. Esta conversación podría usarse para tener un sistema híbrido y añadir al chatbot musical también un sistema de recomendación basado en contenido con el historial como fuente para conocer las preferencias de los usuarios.
- Como trabajo futuro también hay algunas pequeñas mejoras que han sugerido los usuarios, como añadir más funcionalidades. Muchos usuarios han comentado la posibilidad de añadir una mejor página de front-end, más funciones en el ámbito de la música, como buscar la letra de una canción y una mejora en la recomendación.
- Integración del chatbot en aplicaciones de mensajería: existe la posibilidad de integrar el chatbot en aplicaciones de mensajería como Telegram y Facebook. Esto podría haber facilitado mucho la evaluación con los usuarios reales, ya que hoy en día todos tiene un teléfono inteligente y es mucho más accesible.
- Evaluar otras alternativas de recomendación. Actualmente solo se utiliza un filtrado colaborativo, pero como se ha podido observar en la evaluación con usuarios, no se ha podido resolver muy bien el problema del cold-start, ya que solo buscando la similitud en las preferencias de categoría de música no es suficiente.

7. Referencias

[1] Adamopoulou, E., and Moussiades, L. (2020). An Overview of Chatbot Technology. En IFIP Advances in Information and Communication Technology (pp. 373-383). https://doi.org/10.1007/978-3-030-49186-4_31

[2] Dialogflow | Google Cloud. (s. f.). Google Cloud. https://cloud.google.com/dialogflow?hl=es

[3] Mehta, Y., Singhania, A. V., Tyagi, A., Shrivastava, P., & Mali, M. N. (2020). A Comparative Study of Recommender Systems. En Springer eBooks (pp. 1021-1029). https://doi.org/10.1007/978-981-15-1420-3_112

[4] Esteban, V. L. (2018). Allyn, A Recommender Assistant for Online Bookstores. https://upcommons.upc.edu/handle/2117/126021

[5] McAuley, J. (s. f.). Recommender Systems Datasets. https://cseweb.ucsd.edu/~jmcauley/datasets.html#amazon_reviews

[6] Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. Communications of The ACM, 9(1), 36-45. https://doi.org/10.1145/365153.365168

[7] Colby, K. M., Weber, S., & Hilf, F. D. (1971). Artificial Paranoia. Artificial Intelligence, 2(1), 1-25. https://doi.org/10.1016/0004-3702(71)90002-6

[8] Apple. (s. f.). Siri. https://www.apple.com/siri/

[9] Cortana - Your personal productivity assistant. (s. f.). Cortana - Your personal productivity assistant. https://www.microsoft.com/en-us/cortana

[10] Bizzaco, M., Rawes, E., & Wetzel, K. (2022). What is Amazon Alexa, and what can it do? Digital Trends.

[11] IBM Watson | IBM. (s. f.). https://www.ibm.com/watson

[12] Google Assistant, your own personal Google default. (s. f.). Assistant. https://assistant.google.com/

[13] Lo, C. K. (2023). What Is the Impact of ChatGPT on Education? A Rapid Review of the Literature. Education Sciences, 13(4), 410. https://doi.org/10.3390/educsci13040410

[14] Hussain, S., Sianaki, O. A., & Ababneh, N. (2019). A Survey on Conversational Agents/Chatbots Classification and Design Techniques. En Advances in intelligent systems and computing (pp. 946-956). Springer Nature. https://doi.org/10.1007/978-3-030-15035-8_93

[15] Serban, I., V. (2017). A Deep Reinforcement Learning Chatbot. arXiv.org. https://arxiv.org/abs/1709.02349

[16] Li, C., Yeh, S., Chang, T., Tsai, M., Chen, K., & Chang, Y. (2020). A Conversation Analysis of Non-Progress and Coping Strategies with a Banking Task-Oriented Chatbot. https://doi.org/10.1145/3313831.3376209

[17] Agarwal, A. (2021). Evaluating Empathetic Chatbots in Customer Service Set-

tings. arXiv.org. https://arxiv.org/abs/2101.01334

[18]Hub, R. C. C. (2022). Los auténticos chatbots son conversacionales - Contact Center Hub. Contact Center Hub.

[19] Shawar, B. A., & Atwell, E. (2007). Chatbots: are they really useful?. Journal for Language Technology and Computational Linguistics, 22(1), 29-49. https://jlcl.org/article/download/88/86

[20] Turing, A. M. (1950). I.—COMPUTING MACHINERY AND INTELLIGENCE. Mind, LIX(236), 433-460. https://doi.org/10.1093/mind/lix.236.433

[21] Thorat, S. A., & Jadhav, V. C. (2020). A Review on Implementation Issues of Rule-based Chatbot Systems. Social Science Research Network. https://doi.org/10.2139/ssrn.3567047

[22] Kang, J., Condiff, K., Chang, S., Konstan, J. A., Terveen, L., & Harper, F. M. (2017). Understanding How People Use Natural Language to Ask for Recommendations. https://doi.org/10.1145/3109859.3109873

[23] Yan, Z., Duan, N., Chen, P., Zhou, M., Zhou, J., & Li, Z. (2017). Building Task-Oriented Dialogue Systems for Online Shopping. Proceedings of the . . . AAAI Conference on Artificial Intelligence, 31(1). https://doi.org/10.1609/aaai.v31i1.11182

[24] Kapočiūtė-Dzikienė, J. (2020). A Domain-Specific Generative Chatbot Trained from Little Data. Applied sciences, 10(7), 2221. https://doi.org/10.3390/app10072221

[25] Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to Recommender Systems Handbook. En Springer eBooks (pp. 1-35). https://doi.org/10.1007/978-0-387-85820-3_1

[26] Ali, S. K., Aydam, Z. M., & Rashed, B. M. (2020). Similarity metrics for classification: A Review. IOP Conference Series: Materials Science and Engineering, 928(3), 032052. https://doi.org/10.1088/1757-899x/928/3/032052

[27] Schafer, J., Frankowski, D., Herlocker, J. A., & Sen, S. (2007). Collaborative Filtering Recommender Systems. En Springer eBooks (pp. 291-324). https://doi.org/10.1007/978-3-540-72079-9_9

[28] Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based Recommender Systems: State of the Art and Trends. En Springer eBooks (pp. 73-105). https://doi.org/10.1007/978-0-387-85820-3_3

[29] Tarus, J. K., Niu, Z., & Mustafa, G. (2018). Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. Artificial Intelligence Review, 50(1), 21-48. https://doi.org/10.1007/s10462-017-9539-5

[30] Breese, J. S. (1988). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. arXiv.org. https://arxiv.org/abs/1301.7363

[31] Huang, K. L., & Jane, C. (2009). A hybrid model for stock market forecasting and portfolio selection based on ARX, grey system and RS theories. Expert Systems With Applications, 36(3), 5387-5392. https://doi.org/10.1016/j.eswa.2008.

06.103

[32] Karn, A. L., Karna, R. K., Kondamudi, B. R., Bagale, G., Pustokhin, D. A., Pustokhina, I. V., & Sengan, S. (2022). Customer centric hybrid recommendation system for E-Commerce applications by integrating hybrid sentiment analysis. Electronic Commerce Research, 23(1), 279-314. https://doi.org/10.1007/s10660-022-09630-z

[33] Jiang, Z., Jiang, Y. D., Wang, Y., Zhang, H., Cao, H., & Tian, G. (2019). A hybrid approach of rough set and case-based reasoning to remanufacturing process planning. Journal of Intelligent Manufacturing, 30(1), 19-32. https://doi.org/10.1007/s10845-016-1231-0

[34] Devlin, J. (2018, 11 octubre). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv.org. https://arxiv.org/abs/1810.04805

[35] colaboradores de Wikipedia. (2022a. Webhook. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Webhook

[36] Cornac: A Comparative Framework for Multimodal Recommender Systems — Cornac 1.15.4 documentation. (s. f.). https://cornac.readthedocs.io/en/latest/

[37] Welcome to Flask — Flask Documentation (2.3.x). (s. f.). https://flask.palletsprojects.com/en/2.3.x/

[38] PostgreSQL. (2023, 3 junio). PostgreSQL.https://www.postgresql.org/

[39] Raghuwanshi, S. K., & Pateriya, R. K. (2018). Recommendation Systems: Techniques, Challenges, Application, and Evaluation. En Advances in intelligent systems and computing (pp. 151-164). Springer Nature. https://doi.org/10.1007/978-981-13-1595-4_12

[40] Moradizeyveh, S. (2022). intentoRecognition in Conversational Recommender Systems. arXiv.org. https://arxiv.org/abs/2212.03721

[41] Jin, Y., Cai, W., Chen, L., Htun, N. N., & Verbert, K. (2019). MusicBot: Evaluating Critiquing-Based Music Recommenders with Conversational Interaction. https://doi.org/10.1145/3357384.3357923

[42] Fadhlullah, G. A., Baizal, Z. K. A., & Ikhsan, N. I. (2021). Conversational Recommender Systems Based on Mobile Chatbot for Culinary. Jurnal media informatika Budidarma, 5(4), 1233. https://doi.org/10.30865/mib.v5i4.3242

[43] Holotescu, C., & Holotescu, V. (2016). MOOCBuddy: a chatbot for personalized learning with MOOCs. ResearchGate. https://www.researchgate.net/publication/304037510_MOOCBuddy_a_chatbot_for_personalized_learning_with_MOOCs

[44] Thongyoo, P., Anantapanya, P., Jamsri, P., & Chotipant, S. (2020). A Personalized Food Recommendation Chatbot System for Diabetes Patients. En Springer eBooks (pp. 19-28). https://doi.org/10.1007/978-3-030-60816-3_3

[45] pandas.DataFrame — pandas 2.0.2 documentation. (s. f.). https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

[46] Flask | ngrok documentation. (2023). https://ngrok.com/docs/using-ngrok-with/flask/