

# **BEAT BINDER**

## **Player Semántico Interactivo**

**Reproductor de multimedia adaptativo: Interacción  
en tiempo real e IA.**

**3a Documentación**



# **Beat Binder**

**Gestión y Distribución de la Información Empresarial**

Luis Miguel Vargas Durán & Mario Ventura Burgos

Universitat de les Illes Balears

Curso 2023/24

# Índice

|  |           |
|--|-----------|
| <b>Introducción.....</b>   | <b>2</b>  |
| <b>Objetivos.....</b>  | <b>2</b>  |
| <b>Funcionalidades implementadas y relaciones con la práctica 2.....</b>     | <b>3</b>  |
| Chat en Tiempo Real Usando Web Sockets.....                                  | 3         |
| IA para Reconocimiento y Clasificación de Sonidos.....                       | 3         |
| <b>Implementación de comunicación en tiempo real usando WebSockets:.....</b> | <b>4</b>  |
| Implementación y funcionamiento de los sockets.....                          | 5         |
| Frontend / Estilo / Atmósfera.....   | 6         |
| <b>Técnicas de IA offline usadas.....</b>                                    | <b>7</b>  |
| <b>Reflexión.....</b>  | <b>9</b>  |
| <b>Autoevaluación.....</b>   | <b>10</b> |
| Aspectos necesarios (enunciado de la práctica).....                          | 10        |
| Aspectos opcionales (valoración adicional).....                              | 11        |
| Puntos fuertes.....  | 11        |
| Puntos mejorables.....   | 11        |

# Introducción

En el presente documento, se detallarán los pasos que se han seguido para hacer que el reproductor semántico *Beat Binder*, cuyas funcionalidades y explicación técnica se desarrollan en la 2a documentación entregada, ahora también tenga la capacidad de establecer una comunicación en tiempo real. Es decir, *Beat Binder* se expande como reproductor audiovisual semántico y de streaming adaptativo, para integrar ahora **funcionalidades de comunicación en tiempo real** que permiten una **interacción con los usuarios conectados** en cada momento.

Esta práctica integra tecnologías como *WebSockets* para la interacción en tiempo real y *Hugging Face* para la creación del nuevo contenido. A continuación, la URL a *Beat Binder*: <https://gdie2408.ltim.uib.es>

## Objetivos

El objetivo principal de esta práctica es ampliar las capacidades del reproductor audiovisual semántico desarrollado previamente, incorporando funcionalidades de interacción en tiempo real y técnicas de Inteligencia Artificial

Los objetivos principales de la práctica son:

- Implementar y gestionar algún tipo de interacción en tiempo real que permita que diferentes usuarios de la página estén conectados simultáneamente desde dispositivos distintos.
- Utilizar alguna técnica de inteligencia artificial, ya sea off-line o on-line

Adicionalmente, esta práctica buscará valorar aspectos complementarios que enriquezcan la experiencia del usuario y optimicen el rendimiento del reproductor, tales como la adecuación funcional al caso de uso específico, la calidad de los elementos multimedia, el diseño visual de la aplicación, y la incorporación de funcionalidades avanzadas como la selección de múltiples pistas de audio o subtítulos, la interacción avanzada mediante metadatos semánticos y el uso de APIs externas para enriquecer el contenido ofrecido.

# Funcionalidades implementadas y relaciones con la práctica 2

En la continuación y mejora del reproductor audiovisual semántico, se han implementado varias funcionalidades adicionales que enriquecen la experiencia del usuario y optimizan la funcionalidad del sistema, en línea con los objetivos de la práctica 2. Estas nuevas características incluyen:

## Chat en Tiempo Real Usando Web Sockets

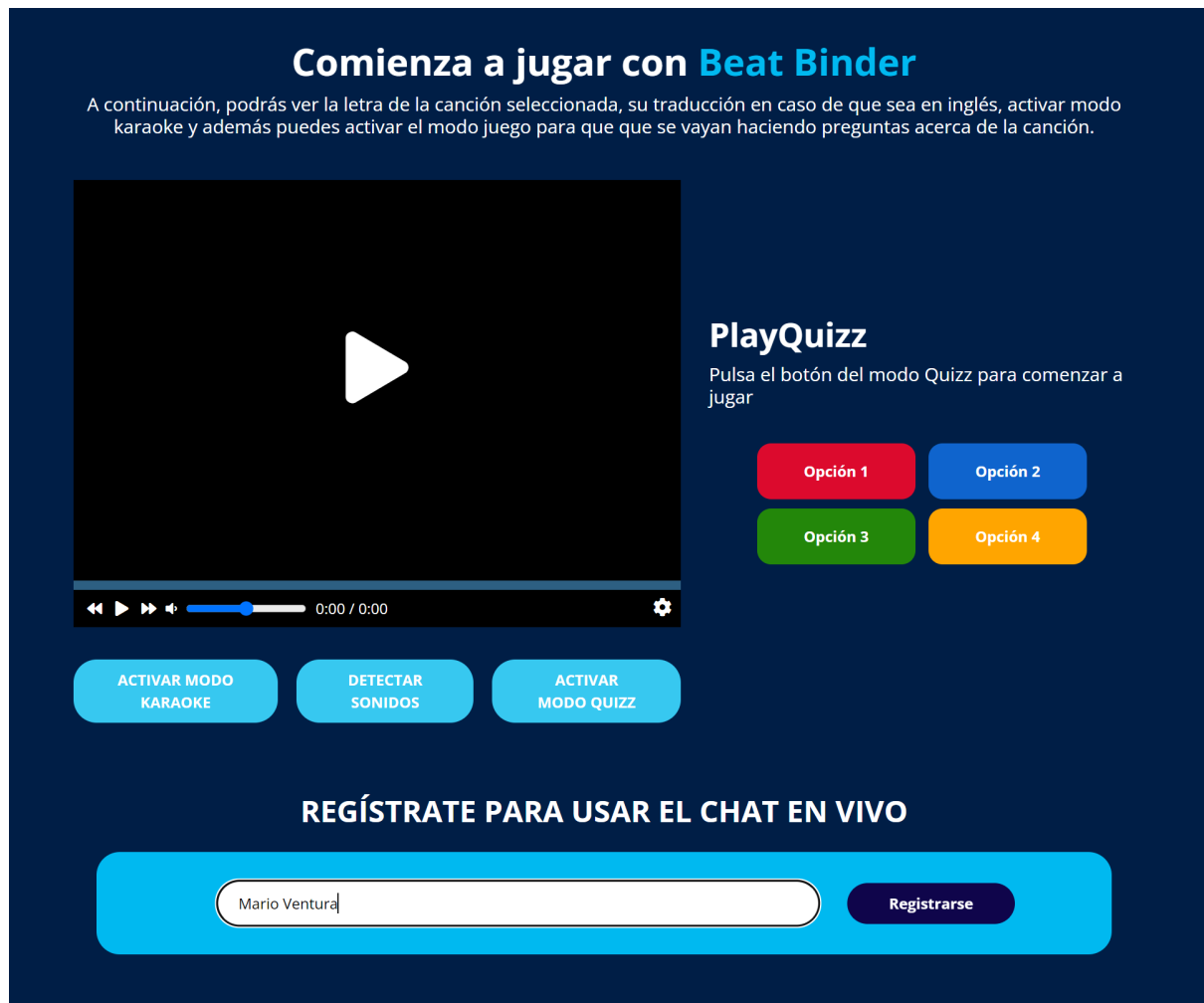
- **Descripción:** Se ha integrado un chat en tiempo real que permite a los usuarios comunicarse mientras visualizan el contenido multimedia. Esta funcionalidad utiliza Web Sockets para mantener una conexión constante y eficiente entre los usuarios, asegurando una comunicación instantánea.
- **Relación con la Práctica 2:** La adición del chat en tiempo real mejora la interacción social y la experiencia colaborativa de los usuarios, complementando y dando más credibilidad a las capacidades de streaming adaptativo y la distribución de contenido.

## IA para Reconocimiento y Clasificación de Sonidos

- **Descripción:** Se ha implementado una inteligencia artificial que analiza los primeros 20 segundos de cada canción, identificando y clasificando los sonidos presentes. Esta IA es capaz de reconocer patrones y categorizar los elementos sonoros, proporcionando metadatos adicionales que enriquecen la experiencia de usuario.
- **Relación con la Práctica 2:** Esta funcionalidad se alinea con la práctica 2 al añadir un nivel de interacción avanzada con el contenido multimedia. La clasificación de sonidos mediante IA proporciona datos semánticos valiosos que pueden ser utilizados para mejorar la adaptabilidad y la personalización del contenido.

Estas nuevas funcionalidades se suman a las ya existentes, tales como la reproducción adaptativa basada en MPEG-DASH y HLS, la gestión de múltiples calidades de vídeo. Juntas, estas características no solo mejoran la calidad del servicio y la experiencia del usuario, sino que también demuestran la capacidad del sistema para integrar tecnologías avanzadas y proporcionar un entorno multimedia interactivo y adaptable.

Tras la implementación de estas dos nuevas funcionalidades, el reproductor de Beat Binder, eje central de la plataforma, tiene ahora el siguiente aspecto:



Puede verse un chat en vivo en la parte inferior del reproductor, y un nuevo botón con el texto "Detectar sonidos". La implementación de estas funcionalidades se detalla en el siguiente apartado.

## Implementación de comunicación en tiempo real usando WebSockets:

Se ha implementado un chat en tiempo real que permite la emisión y recepción de mensajes entre los distintos usuarios activos en la web. El objetivo es permitir la comunicación en tiempo real entre los usuarios mientras visualiza contenido multimedia. Para la implementación se han usado Web Sockets, y se ha modificado el html y css de index.html y style.css respectivamente. La implementación detallada es la descrita a continuación.

Lo primero que se debe hacer es incluir la librería de Socket.IO en el archivo index.html. Esto se hace añadiendo una simple línea de código.

```
<script src="/socket.io/socket.io.js"></script>
```

Hecho esto, la implementación de Web Sockets para crear interacción en tiempo real puede dividirse en dos partes diferenciadas:

## Implementación y funcionamiento de los sockets

Respecto a los detalles de la implementación referentes al funcionamiento interno, se ha seguido la lógica descrita a continuación.

En la parte del frontend, se define un script para gestionar la comunicación con el servidor mediante WebSockets. Este script maneja eventos como el envío de mensajes, la notificación de nuevos usuarios que se unen, la actualización de la lista de usuarios y la gestión de desconexiones. Todo esto aparecerá en contenedores cuyos detalles de implementación se detallarán posteriormente.

El script tiene eventos asociados a estos sockets, de forma que cada evento hace una acción distinta. Por ejemplo, el evento que maneja el envío de mensajes es el siguiente:

```
// Enviar mensaje
form.addEventListener('submit', function (e) {
  e.preventDefault();
  if (input.value) {
    socket.emit('chat message', { username, message:
                                     input.value });
    input.value = '';
  }
});
```

Con este código se está añadiendo un event listener al formulario (form) que se activa cuando el formulario se envía (evento 'submit'). Este evento es de tipo 'submit' en el formulario, y al detectarse se ejecuta la función `function (e) { ... }`.

`e.preventDefault()`; se utiliza para prevenir el comportamiento predeterminado del formulario, que normalmente recargará la página al enviarse. Después se verifica el valor del input, se envía el mensaje a través del socket usando **`socket.emit(...)`** y se limpia el campo input por si se quiere enviar otro mensaje después.


En index.js la aplicación gestiona las conexiones de Web Sockets mediante Socket.IO. Se mantiene una lista de usuarios conectados y se manejan eventos como la unión de nuevos usuarios, el envío de mensajes y la desconexión de usuarios.

Las acciones gestionadas por eventos de los sockets son:

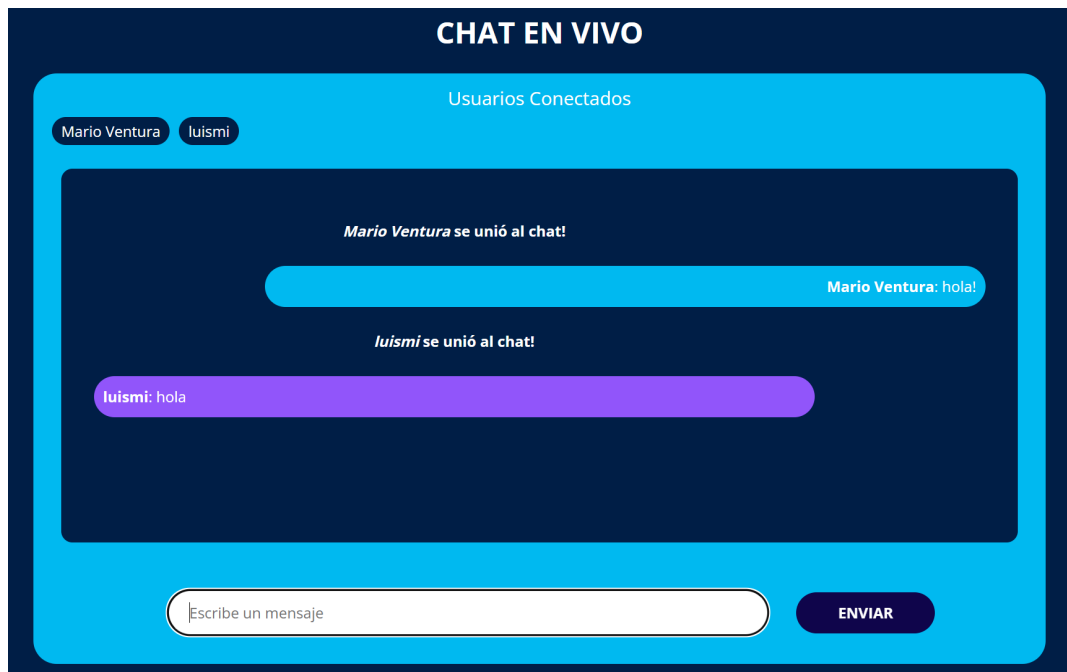
- Añadir un nombre de usuario.
- Enviar mensaje.
- Escuchar mensajes entrantes y mostrarlo en la interfaz.
- Manejar el evento cuando el usuario se une exitosamente.
- Actualizar la lista de usuarios activos en la interfaz de usuario.
- Mandar mensajes de sistema cuando un usuario abandona el chat.
- Manejar el evento cuando el nombre de usuario ya está tomado.

## Frontend / Estilo / Atmósfera

Para mantener la atmósfera/estilo de la página, se crean los elementos html para el chat, que son formularios para el registro de usuario y envío de mensajes, así como un contenedor para mostrar los mensajes. Estos contenedores están creados pero el del chat se mantiene oculto hasta que el usuario se registra con un nombre de usuario con el que se identificará al enviar mensajes. Estos nombres de usuario son únicos, de forma que se asegura que no se va a poder tener nunca dos usuarios con el mismo nombre en una misma sesión.

A registration form for a live chat. It features a dark blue header with the text "REGÍSTRATE PARA USAR EL CHAT EN VIVO" in white. Below the header is a light blue rounded rectangle containing a white input field with the text "Mario Ventura" and a dark blue button with the text "Registrarse" in white.

Una vez el usuario se ha registrado, este contenedor se oculta (propiedad hidden en css) y aparece el contenedor con los mensajes en vivo.



Este panel cuenta con una lista de todos los usuarios conectados en la sesión activa (“luismi” y “Mario Ventura” en la imagen), y tiene un tamaño máximo que si se completa, se convertirá en un panel donde se podrá ir haciendo scroll para ver los mensajes.

Se distinguen 3 tipos de mensajes:

1. **Mensajes enviados por el usuario:** Mensajes que envías a otros usuarios.
2. **Mensajes enviados por otros usuarios:** Mensajes que otros usuarios envían.
3. **Mensajes de sistema:** Avisos del sistema, como “Luismi se unió al chat” o similares.

En función de si el nombre de usuario del emisor del mensaje es igual al nombre de usuario introducido al registrarse en el chat, se sabe si el mensaje lo ha mandado el propio usuario (caso 1), otro usuario conectado (caso 2), o si es un mensaje de sistema (caso 3).

Esto permite crear un chat en vivo que sigue la atmósfera e identidad de la página.

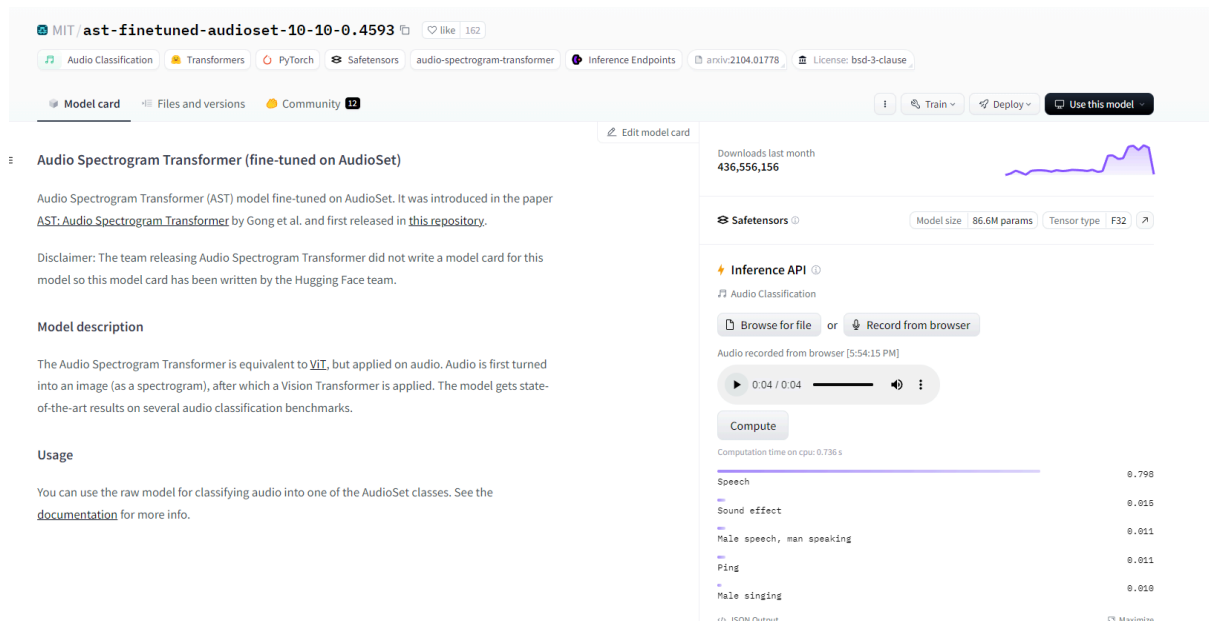
## Técnicas de IA offline usadas

En el desarrollo de nuestra aplicación interactiva y semántica de vídeo y audio, hemos integrado técnicas avanzadas de inteligencia artificial (IA) utilizando las capacidades proporcionadas por HuggingFace.

Hemos optado por utilizar la API de HuggingFace, especialmente el modelo MIT/ast-finetuned-audioset-10-10-0.4593, que es una implementación altamente optimizada para el análisis de audio. Este modelo está entrenado para identificar diversos sonidos y



características del audio, lo que nos permite ofrecer una experiencia rica y detallada al analizar las canciones reproducidas en nuestra aplicación. A continuación, se puede ver una imagen del modelo de HuggingFace usado:



The screenshot shows the HuggingFace model card for MIT/ast-finetuned-audioset-10-10-0.4593. The card includes a title, a description of the model, and a table of inference results. The table shows the model's performance on various audio classification tasks.

| Task                      | Score |
|---------------------------|-------|
| Speech                    | 0.798 |
| Sound effect              | 0.015 |
| Male speech, man speaking | 0.011 |
| Ping                      | 0.011 |
| Male singing              | 0.010 |

Para integrar esta capacidad, el código incorpora llamadas a la API de Hugging Face, enviando fragmentos de audio y recibiendo análisis detallados sobre los tipos de sonidos detectados. Aunque esta integración requiere conectividad, la forma en que preparamos y manejamos los datos de audio se realiza de manera local, preparando el sistema para una posible transición a soluciones completamente offline en el futuro. Este es el código en el que se puede ver esta llamada a la API para conectar con el servidor de HuggingFace:

```
async function query(filename) {
  const formData = new FormData();
  formData.append('file', filename, 'audio.aac');
  const response = await
  fetch("https://api-inference.huggingface.co/models/MIT/ast-finetuned-audioset-10-10-0.4593",
    {
      headers: {'Authorization': 'Bearer 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',
        'Content-Type': 'application/json'},
      method: "POST",
      body: formData
    }
  );
  const result = await response.json();
  return result;
}
```

La preparación de datos de audio se maneja localmente utilizando técnicas de procesamiento digital de señales. Esto incluye la extracción de las pistas de audio, enviándolo a la API para el análisis.

La integración de IA, a través de la plataforma HuggingFace, ha sido un componente crucial para enriquecer la interacción del usuario y la experiencia de navegación en nuestra aplicación y en la siguiente imagen se puede ver una captura de cómo se muestra la IA en el contenido de la web:



Vemos que el JSON que devuelve la IA se transforma en este gráfico que es mucho más atractivo visualmente para los usuarios.

## Reflexión

La integración de WebSockets ha sido un paso crítico para lograr una interacción fluida y en tiempo real entre los usuarios. Esta tecnología nos ha permitido establecer una comunicación bidireccional y persistente, lo que es fundamental para funciones como el chat en tiempo real. Este chat no solo mejora la experiencia social y colaborativa del usuario sino que también amplía las funcionalidades de streaming adaptativo, proporcionando un canal directo de comunicación que enriquece la experiencia de cada usuario al permitirle

compartir impresiones y comentarios en tiempo real sobre el contenido que están consumiendo.

El uso de la IA de HuggingFace para el análisis y clasificación de sonidos es otro aspecto transformador para Beat Binder. Al analizar los primeros 15 segundos de cada canción, el sistema no solo identifica diversos sonidos y características sino que también enriquece el contenido multimedia con metadatos útiles. Esta capacidad de reconocimiento y clasificación agrega una capa de interacción semántica que mejora la personalización y la relevancia del contenido presentado a los usuarios, haciéndolo más interactivo y adaptativo a sus preferencias.

La fusión de estas tecnologías no solo ha mejorado la calidad del servicio ofrecido sino que también ha demostrado la capacidad de BeatBinder para adaptarse e integrar nuevas tecnologías que mejoran la interacción del usuario. Por ejemplo, el chat en tiempo real y la detección de sonidos mediante IA transforman la manera en que los usuarios interactúan con la plataforma, no solo como espectadores sino como participantes activos en una comunidad dinámica.

## Autoevaluación

Para poder evaluar el trabajo realizado con cierta objetividad se valorarán diferentes aspectos, teniendo en cuenta los establecidos en el enunciado, los puntos mejorados respecto al día de la presentación en clase, los puntos fuertes (cosas bien hechas y requisitos cumplidos) y los puntos débiles (aspectos mejorables o requisitos no cumplidos). De esta forma se podrá justificar la nota puesta.

### Aspectos necesarios (enunciado de la práctica)

| ASPECTO   | NOTA (1-10) + JUSTIFICACIÓN |
|---|-----------------------------|
| Interacció de l'usuari en temps real, ja sigui amb el servidor o amb altres clients, amb (com a mínim) una de les tècniques esmentades. | 10 (hecho)                  |
| Ús d'alguna tècnica d'Intel·ligència Artificial, ja sigui on-line o off-line  | 10 (hecho)                  |

## Aspectos opcionales (valoración adicional)

| ASPECTO   | NOTA (1-10) + JUSTIFICACIÓN                 |
|---|---|
| Adequació entre la funcionalitat i el cas d'ús triat. | 7 (correcto)                                |
| Interacció més avançada (p. ex.: WebSockets, WebRTC)  | 9 (interacción muy correcta entre usuarios) |
| Ús on-line de tècniques d'Intel·ligència Artificial.  | 0 (no realizado)                            |

## Puntos fuertes

1. **Entorno / atmósfera creada.** Beat Binder tiene una interfaz visual y una identidad de marca (paleta de colores, concepto, etc.) que añaden valor a la plataforma más allá de sus funcionalidades. Se considera que el entorno creado es muy bueno.
2. **Interacción entre usuarios:** La presencia del chat añadido al vídeo permite que los usuarios puedan hablar entre ellos e intercambiar opiniones acerca de lo que están viendo en el reproductor, lo que crea comunidad.
3. **Clasificación de sonidos:** Esta nueva funcionalidad, le añade valor a la web, ya que el usuario podrá encontrar y descubrir los sonidos que aparecen en el videoclip, ayudándole a entrar en todos los detalles de la canción

## Puntos mejorables

1. **Clasificación de sonidos:** Como se ha comentado anteriormente, esta IA escogida para la clasificación de sonidos solo coge 15 segundos del audio de la canción, como punto mejorable estaría bien que la IA cogiera los últimos 15 segundos escuchados por el usuario y detectara los sonidos de estos últimos 15 segundos.

Teniendo todos estos aspectos en cuenta, y evaluando la relevancia que se considera que tiene cada uno de ellos, **se estima una nota final de entre 7,5 y 8,5**. Se cree que esta nota es acorde al trabajo presentado ya que se cumplen con los requisitos obligatorios, los cuales han sido implementados correctamente según lo propuesto, y se implementan también algunos aspectos adicionales. Aun así, el trabajo no es excelente ya que, como siempre, todavía existe cierto margen de mejora.