



INSTITUTO POLITÉCNICO NACIONAL

CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

DIPLOMADO EN INTELIGENCIA ARTIFICIAL

PROYECTO

USO DE API DE IA GENERATIVA

LUIS ARIEL VÁZQUEZ PIÑA



SEPTIEMBRE 2025

Resumen del Proyecto

El proyecto **Melodia** consiste en el desarrollo de una aplicación web que funciona como un asistente creativo para músicos y compositores. Utilizando el poder de un Modelo de Lenguaje Grande (LLM), la herramienta ayuda a los usuarios a superar el bloqueo creativo, generar ideas líricas, refinar textos y estructurar canciones de manera interactiva. Este informe detalla la selección del modelo de IA, la definición de las tareas generativas, el modelo de negocio propuesto y la implementación técnica del prototipo funcional.


Paso 0: Preparación del entorno

I. Elección de un LLM y obtención de una API Key

Gemini (de Google). Suele ser más sencillo de configurar y tiene una capa gratuita generosa.

Para obtener la API KEY se siguieron los siguientes pasos:

- Se inicia sesión con una cuenta de Google en [Google AI Studio](#).
- Hacer clic en **"Get API key"** -> **"Create API key in new project"**.
- **Copiar y guardar esa clave en un lugar seguro.**

Número del proyecto	Nombre del proyecto	Clave de API	Fecha de creación	Plan
...7471	Gemini API 	...o4YE	17 sept 2025	Free tier Configurar facturación Ver datos de uso

II. Preparación del Software

- Instalar Python y/o actualizar la última versión.
- Preparar el entorno en un editor de código en este caso Visual Studio Code.

```
7
8  genai.configure(api_key=API_KEY)
9
10 # Crea el modelo
11 model = genai.GenerativeModel('gemini-1.5-flash')
12
13 # --- LA PREGUNTA A LA IA ---
14 # Este es el "prompt" que le enviaremos.
15 prompt = "Dame 5 palabras que rimen con 'canción' en español."
16
17 print("Enviando petición a la IA...")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CODE REFERENCE LOG

powershell + - [] [] ... [] [] X

ee the list of available models and their supported methods.

PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI> python test_api.py

Respuesta de la IA:

No existen muchas palabras en español que rimen perfectamente con "canción". La "c" con sonido de "k" y la "n" final limitan mucho las posibilidades. Podríamos considerar algunas aproximaciones, dependiendo del grado de rima que se busque:

1. **acción:** Rima asonante (rima en la vocal)
2. **sazón:** Rima asonante
3. **reacción:** Rima asonante
4. **ilusión:** Rima asonante
5. **fracción:** Rima asonante (menos cercana)

Es importante notar que todas estas son rimas imperfectas o asonantes. Una rima perfecta con "canción" es extremadamente difícil de encontrar en español.

```
7
8  genai.configure(api_key=API_KEY)
9
10 # Crea el modelo
11 model = genai.GenerativeModel('gemini-1.5-flash')
12
13 # --- LA PREGUNTA A LA IA ---
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CODE REFERENCE LOG

powershell + - [] [] ... [] [] X

ee the list of available models and their supported methods.

PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI> python test_api.py

Respuesta de la IA:

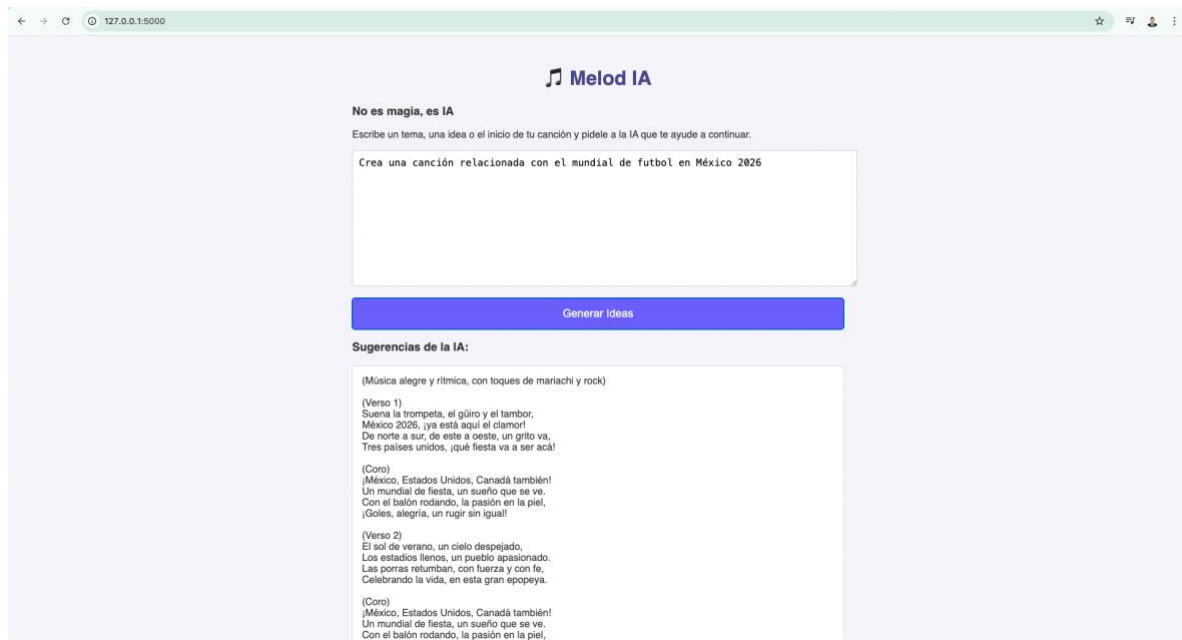
No existen muchas palabras en español que rimen perfectamente con "canción". La "c" con sonido de "k" y la "n" final limitan mucho las posibilidades. Podríamos considerar algunas aproximaciones, dependiendo del grado de rima que se busque:

1. **acción:** Rima asonante (rima en la vocal)
2. **sazón:** Rima asonante
3. **reacción:** Rima asonante
4. **ilusión:** Rima asonante
5. **fracción:** Rima asonante (menos cercana)

Es importante notar que todas estas son rimas imperfectas o asonantes. Una rima perfecta con "canción" es extremadamente difícil de encontrar en español.

PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI> █

Desarrollo de la aplicación web index.html



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  CODE REFERENCE LOG

el sistema. Ejecute únicamente los scripts de los editores de confianza.
PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI> pip install google-generativeai
Collecting google-generativeai
  Downloading google_generativeai-0.8.5-py3-none-any.whl.metadata (3.9 kB)
Collecting google-ai-generativelanguage==0.6.15 (from google-generativeai)
  Downloading google_ai_generativelanguage-0.6.15-py3-none-any.whl.metadata (5.7 kB)
Collecting google-api-core (from google-generativeai)
  Downloading google_api_core-2.25.1-py3-none-any.whl.metadata (3.0 kB)
Collecting google-api-python-client (from google-generativeai)
  Downloading google_api_python_client-2.182.0-py3-none-any.whl.metadata (7.0 kB)
Collecting google-auth>=2.15.0 (from google-generativeai)
  Downloading google_auth-2.40.3-py2.py3-none-any.whl.metadata (6.2 kB)
Collecting protobuf (from google-generativeai)
Collecting pydantic (from google-generativeai)
  Downloading pydantic-2.11.9-py3-none-any.whl.metadata (68 kB)
Collecting tqdm (from google-generativeai)
```

Se desarrolla en Script en Python y se ejecuta un prompt de prueba.

```
7
8  genai.configure(api_key=API_KEY)
9
10 # Crea el modelo
11 model = genai.GenerativeModel('gemini-1.5-flash')
12
13 # --- LA PREGUNTA A LA IA ---
14 # Este es el "prompt" que le enviaremos.
15 prompt = "Dame 5 palabras que rimen con 'canción' en español."
16
17 print("Enviando petición a la IA...")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CODE REFERENCE LOG

ee the list of available models and their supported methods.
PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI> python test_api.py

Respuesta de la IA:

No existen muchas palabras en español que rimen perfectamente con "canción". La "c" con sonido de "k" y la "n" final limitan mucho las posibilidades. Podríamos considerar algunas aproximaciones, dependiendo del grado de rima que se busque:

1. **acción:** Rima asonante (rima en la vocal)
2. **sazón:** Rima asonante
3. **reacción:** Rima asonante
4. **ilusión:** Rima asonante
5. **fracción:** Rima asonante (menos cercana)

Es importante notar que todas estas son rimas imperfectas o asonantes. Una rima perfecta con "canción" es extremadamente difícil de encontrar en español.

```
7
8  genai.configure(api_key=API_KEY)
9
10 # Crea el modelo
11 model = genai.GenerativeModel('gemini-1.5-flash')
12
13 # --- LA PREGUNTA A LA IA ---
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CODE REFERENCE LOG

ee the list of available models and their supported methods.
PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI> python test_api.py

Respuesta de la IA:

No existen muchas palabras en español que rimen perfectamente con "canción". La "c" con sonido de "k" y la "n" final limitan mucho las posibilidades. Podríamos considerar algunas aproximaciones, dependiendo del grado de rima que se busque:

1. **acción:** Rima asonante (rima en la vocal)
2. **sazón:** Rima asonante
3. **reacción:** Rima asonante
4. **ilusión:** Rima asonante
5. **fracción:** Rima asonante (menos cercana)

Es importante notar que todas estas son rimas imperfectas o asonantes. Una rima perfecta con "canción" es extremadamente difícil de encontrar en español.

○ PS C:\Users\Programador\Documents\PROYECTOS CON IA\GEMINI>

PASO 1. Selección del Modelo de IA

Para este proyecto, se ha seleccionado el modelo de lenguaje: Google Gemini.



La elección se justifica por las siguientes razones:

- **Flexibilidad y Potencia:** Gemini ofrece una familia de modelos, como gemini-1.5-flash y gemini-1.5-pro, que proporcionan un excelente balance entre velocidad de respuesta y capacidad de razonamiento creativo, ideal para las tareas de composición musical.
- **Capa Gratuita Robusta:** La API de Gemini cuenta con una generosa cuota de uso gratuito que es perfectamente adecuada para la fase de desarrollo, prototipado y pruebas del proyecto sin incurrir en costos.
- **Capacidad Creativa Demostrada:** Durante las pruebas iniciales, el modelo demostró una alta capacidad para generar texto poético, mantener la coherencia temática y responder a instrucciones estructuradas, como la creación de listas de rimas.

Conoce a los modelos

[Prueba Gemini en Google AI Studio](#)

2.5 Pro ✦

Nuestro modelo de razonamiento más potente, con funciones para el razonamiento complejo y mucho más

2.5 Flash ✦

Nuestro modelo multimodal más reciente, con funciones de nueva generación y capacidades mejoradas

2.5 Flash-Lite ✦

Nuestro modelo multimodal más rápido y rentable con un gran rendimiento para tareas de alta frecuencia

Veo 3

Nuestro modelo de generación de videos de vanguardia

Gemini 2.5 Flash Image

Nuestro modelo de generación de imágenes altamente eficaz y preciso

Gemini Embeddings []

Nuestro primer modelo de embedding de Gemini, diseñado para flujos de trabajo de RAG de producción

PASO 2. Identificación de Tareas de IA

MelodIA integra múltiples tareas de IA generativa para ofrecer una solución completa y versátil.

Tarea Principal: Generación de Texto (letras de las canciones)

El núcleo de la aplicación es la capacidad de generar contenido lírico. Esto abarca desde la creación de ideas iniciales y títulos a partir de un tema, hasta la redacción de estrofas, estribillos y puentes completos, adaptándose al contexto proporcionado por el usuario.

Tareas Secundarias y Avanzadas:

Generación de Datos Estructurados: Una de las funciones más innovadoras del proyecto es la capacidad de la IA para devolver no solo texto plano, sino datos en

formato: JSON. Esto permite estructurar la información de la canción para usos avanzados, como:

- Separar la letra por secciones (verso, coro, etc.).
 - Sugerir un ritmo o género musical.
 - Proponer una progresión de acordes básicos.
 - Incluir marcas de tiempo para una futura sincronización en un formato de karaoke.
-
- Traducción de Texto: Como funcionalidad futura, se contempla la traducción de las letras generadas a otros idiomas, con un prompt diseñado para que la IA intente preservar la rima y el sentido poético original.

PASO 3. Definición del Modelo de Negocio

Se propone un modelo de negocio basado en la metodología de Producto Mínimo Viable (MVP) para validar la idea en el mercado.

3.1 MVP (Producto Mínimo Viable)

Producto de Mayor Valor: El valor fundamental para el usuario es una herramienta interactiva que actúa como un "co-piloto" creativo, eliminando la fricción del bloqueo del escritor y acelerando el proceso de composición.

Mecanismo de Ganancia / Dependencia: La ganancia inicial se mide en la validación del producto. El objetivo es que los compositores encuentren la herramienta tan útil

que se convierta en una parte indispensable de su flujo de trabajo creativo, creando una dependencia positiva.

Funcionalidades del MVP:

- Interfaz web con un área de texto principal.
- Un único botón para "Sugerir Idea" o "Continuar Letra".
- Visualización clara de la sugerencia generada por la IA.

3.2 Mecanismo

Componentes Visuales: La interfaz de usuario (Frontend) se construye con tecnologías web estándar: HTML para la estructura, CSS para el diseño visual y JavaScript para la interactividad del usuario (capturar texto, enviar peticiones y mostrar resultados).

 **Melodia** 

No es Magia, es IA

Escribe un tema, una idea o el inicio de tu canción y pidele a la IA que te ayude a continuar.

Ej: Escribe el coro para una canción sobre un viaje por carretera en verano...

Generar Ideas

Sugerencias de la IA:

Aquí aparecerán las ideas generadas por la IA.

Datos Involucrados: El flujo de datos es el siguiente:

- El usuario introduce un prompt (texto) en el Frontend.
- JavaScript envía este prompt al Backend.
- El Backend, construido con Python y Flask, recibe la petición.
- El Backend se comunica con la API de Gemini, enviando el prompt.
- La API devuelve el texto generado al Backend.
- El Backend envía la respuesta final al Frontend, que la muestra al usuario.



MelodIA - Asistente de Composición

No es magia, es IA

Escribe un tema, una idea o el inicio de tu canción y pídele a la IA que te ayude a continuar.

crea una canción para el mundial de futbol de México 2026

Generar Ideas

Sugerencias de la IA:

(Música alegre y rítmica, con instrumentos de mariachi y toques electrónicos)

(Verso 1)

¡México, México! El grito retumba ya,
Veintiséis, veintiséis, la fiesta está acá.
De norte a sur, de este a oeste, una sola emoción,
La copa del mundo llega, a nuestra nación.

(Coro)

¡Dos mil veintiséis, el sueño se hará realidad!
En tierras aztecas, la gloria se verá.
Con el corazón en la mano, y la garra bien puesta,
México, campeón, ¡nuestra copa será la testa!

(Verso 2)

Desde el Azteca hasta el norte vibrante,
Se siente la magia, un ambiente fascinante.
Bandera en alto, con orgullo y pasión,
Jugaremos con fuerza, con mucha emoción.

3.3 IA: Prompts e Integración

Construcción de Prompts: La calidad de los resultados depende directamente de la ingeniería de prompts. Se han diseñado prompts específicos para cada tarea, por ejemplo:

Para rimas: "Genera una lista de 10 palabras en español que tengan una rima consonante perfecta con la palabra 'canción'."

Para ideas: "Escribe un coro para una canción pop-rock sobre un reencuentro en una vieja estación de tren."

Para datos JSON: "Convierte la siguiente letra a un formato JSON que incluya: la letra dividida en 'verso1' y 'coro', y una sugerencia de 'ritmo'."

Integración de Resultados: La integración es dinámica. El Frontend no se recarga. JavaScript gestiona la petición de forma asíncrona, mostrando un estado de "Pensando..." y actualizando el contenedor de resultados únicamente cuando la respuesta de la IA está lista. El Backend utiliza la función jsonify de Flask para asegurar que la comunicación con el Frontend sea eficiente y estandarizada.