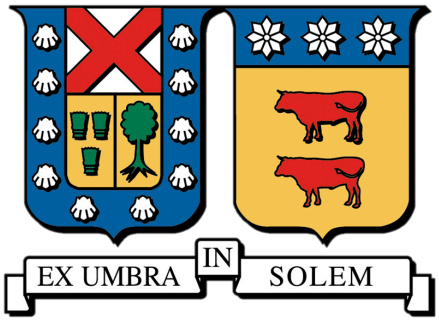
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAÍSO – CHILE



Diseño y Desarrollo de Plataforma de Apreciación Artistica para Personas con Discapacidad Visual

Alexey Nikolay Mitjaew Hupat

Memoria de Titulación para optar al título de Ingeniero Civil Telemático

Profesor Guía:	Patricio Olivares Roncagliolo
Profesor Correferente:	Nicolás Torres Rudloff

Diciembre 2025

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

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

Título del trabajo:
Diseño y Desarrollo de Plataforma de Apreciación Artistica para Personas con Discapacidad Visual

Nombre del candidato(a):
Alexey Nikolay Mitjaew Hupat

Carrera / Grado:
Ingeniero Civil Telemático

Campus: San Joaquin **Departamento:** Electrónica

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Patricio Olivares Roncagliolo, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

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

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

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

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

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

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

4.- FIRMAS

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

Fecha: 22/06/2026 **Firma:** _____

Estudiante o Candidato(a):

Fecha: 22/06/2026 **Firma:** _____

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

Agradecimientos

A mi familia,
mi perro Elvis
y el buen café.

Esta tesis recibió apoyo del Concurso FONDEF IDeA I+D 2025 número ID25I10018 «Teloprogramo: plataforma innovadora de aprendizaje personalizado de programación, basada en inteligencia artificial» financiado por la Agencia Nacional de Investigación y Desarrollo (ANID).

Resumen

Este documento expone el desarrollo de una plataforma accesible de apreciación artística para público con discapacidad visual. La propuesta consiste en transformar obras en representaciones sonoras multifacéticas, integrando descripciones narrativas detalladas, contexto histórico-artístico y atmósferas auditivas por medio del uso de modelos multimodales de lenguaje, sistemas de síntesis de voz y modelos generativos de audio basados en texto.

La solución expuesta propone facilitar el acceso al arte mediante una experiencia adaptada, con el objetivo de ofrecer alternativas de apreciación artística para personas con discapacidad visual.

Palabras clave: *Accesibilidad / Machine Learning / Modelos de Lenguaje / Síntesis de Audio*

Lista de Figuras

Figura 1 Codificación Multimodal Llama V4 16
Figura 2 Arquitectura de Modelo Tacotron 2 18
Figura 3 Arquitectura de Modelo AudioLDM 19
Figura 4 Arquitectura General de Sistema 22
Figura 5 Segmentación por Cuadrantes de Obra «El Aquelarre» 24
Figura 6 Modelo de Datos de la Plataforma 27
Figura 7 Flujo de Navegación Frontend 28
Figura 8 Documentación Swagger de la API 29
Figura 9 Frontend de la Plataforma 30
Figura 10 Funcionalidad de Búsqueda en Frontend 30
Figura 11 Vista General de Obra en Frontend 31
Figura 12 Vista de Ambientes Sonido en Frontend 31
Figura 13 Obra «El Aquelarre» de Francisco de Goya 32

Lista de Tablas

Tabla 1 Comparación del Estado del Arte en Accesibilidad Artística 14
Tabla 2 Muestra de Elementos Detectados en Obra «El Aquelarre» 35

Contenidos

I) Introducción 7

I.1) Estructura 8

I.2) Objetivo General 8

I.3) Objetivos Particulares 8

II) Marco Teórico 9

II.1) Modelos de Lenguaje Multimodales 9

II.2) Modelos Generativos de Audio 10

II.3) Modelos de Text to Speech 10

II.4) Estándares de Accesibilidad 11

III) Estado del Arte 12

III.1) Tecnologías y Prácticas en Accesibilidad Cultural y Artística 12

III.1.a Accesibilidad cultural tradicional y práctica museográfica 12

III.1.b Proyectos y colectivos de accesibilidad en arte 12

III.1.c Investigación académica y tecnologías emergentes 13

III.1.d Plataformas digitales de difusión cultural 13

III.2) Comparación Cruzada de Alternativas 14

III.3) Modelos de Lenguaje Multimodales 15

III.4) Modelos de Texto a Voz 18

III.5) Modelos Generativos de Ambiente Sonoro 19

IV) Desarrollo de la Plataforma 20

IV.1) Toma de Requerimientos 20

IV.1.a Requisitos Funcionales 20

IV.1.b Requisitos No Funcionales 21

IV.2) API Gateway 21

IV.2.a Generador de Audio Descriptivo 23

IV.2.b Generador de Narraciones de Contexto 23

IV.2.c Generador de Sonidos Ambientales 24

IV.2.d Catálogo de Obras/Imágenes 25
IV.2.e Modelo de Datos 26
IV.3) Frontend 28
IV.4) Limitaciones 28
V) Resultados 29
V.1) API Gateway 29
V.2) Frontend 30
V.3) Módulos Generativos 32
V.3.a Generación de Texto de Audio Descriptivo 33
V.3.b Generación de Texto de Narraciones de Contexto 34
V.3.c Generador de Sonidos Ambientales 35
V.3.d Audio Generado con TTS 36
VI) Conclusiones 37
VII) Bibliografía 38
VIII) Anexo 41
VIII.1) Utilidades Generales 41
VIII.2) Pipeline de Extracción de Fuentes 42
VIII.3) Generación de Contenido General 44
VIII.4) Utilidades Generales de Procesamiento de Imágenes 48
VIII.5) Generación de Audio Descriptivo 49
VIII.6) Narrador de Contexto 50
VIII.7) Generación de Sonidos Ambientales 51
VIII.8) Text to Speech 54

I) Introducción

El acceso equitativo al arte constituye un componente esencial del desarrollo cultural y social [1]. Sin embargo, las personas con discapacidad visual continúan enfrentando barreras significativas para disfrutar de obras visuales en condiciones comparables al resto del público. Aunque las tecnologías de apoyo han avanzado [2], aún existe una brecha entre la experiencia estética que ofrecen los museos y plataformas digitales tradicionales frente a las necesidades sensoriales de quienes no pueden percibir elementos visuales directamente. Esta disparidad se origina porque el arte visual comunica a través de composición, color, textura y simbolismo; dimensiones que las soluciones de accesibilidad convencionales, como el texto alternativo o las audioguías elaboradas manualmente, no logran traducir en una experiencia estética equivalente.

En este contexto, la inteligencia artificial emerge como una oportunidad para reimaginar la forma en que se transmite el contenido artístico, permitiendo crear experiencias que integren narrativa, sonido y contextualización cultural. A diferencia de las tecnologías asistivas previas, los modelos multimodales de lenguaje son capaces de interpretar el contenido de una obra con profundidad semántica (identificando elementos, estilos, emociones y simbolismos), mientras que los modelos generativos de audio permiten transformar esas interpretaciones en experiencias sonoras inmersivas. Esta convergencia de capacidades habilita una mediación artística automatizada y escalable que no era posible con herramientas convencionales.

Este proyecto desarrolla una alternativa de accesibilidad para consumo de contenido orientado a personas con discapacidad visual. En particular se plantea utilizar modelos de lenguaje multimodales, sistemas de síntesis de voz y modelos generativos de audio para una pipeline integrada de producción de piezas de audio que comprendan desde narraciones habladas hasta paisajes sonoros que representen los contenidos de las obras.

Dado que esta investigación se fundamenta en los principios de accesibilidad universal y diseño inclusivo, el acceso al contenido se implementará a través de un cliente web que considere estándares establecidos en la materia [2].

I.1) Estructura

El presente informe incluye:

- *Introducción del **Marco Teórico** de los tópicos mas relevantes del trabajo.*
- *Revisión del **Estado del Arte** sobre modelos generativos y estandares de accesibilidad.*
- *Descripción del proceso de **Desarrollo de Plataforma** a nivel de componentes.*
- *Revisión de **Resultados** obtenidos.*
- ***Conclusiones** sobre el potencial de aplicación del trabajo realizado.*

I.2) Objetivo General

Mejorar la experiencia estética y promover la accesibilidad universal a las obras de arte para personas con discapacidad visual mediante el diseño e implementación de una plataforma accesible que utilice inteligencia artificial para transformar automáticamente imágenes de obras artísticas en paisajes sonoros y descripciones auditivas significativas.

I.3) Objetivos Particulares

1. **Proveer un catálogo curado de obras de dominio público**, basado en fuentes autoritativas y verificadas.
2. **Generar audios descriptivos** que contengan una representación objetiva y comprensible del contenido visual de las obras.
3. **Crear audios narrativos con contenido histórico/biográfico** fundamentadas en fuentes reconocidas, que contextualizen la obra, su autor(a) y su relevancia artística.
4. **Producir ambientes sonoros inmersivos** que recreen la atmósfera de las obras con piezas de audio evocativo y coherente con el contenido representado.
5. **Diseñar una interfaz centrada en accesibilidad**, que habilite una experiencia de navegación cómoda para personas con discapacidad visual.

Los aportes realizados en este proyecto corresponden principalmente a los objetivos **1** y **4**. No obstante, también se hicieron observaciones y contribuciones significativas en los módulos **2** y **3**. Por ello, el presente informe se centrará en el potencial de uso de herramientas de estado del arte para la producción de contenido accesible.

II) Marco Teórico

II.1) Modelos de Lenguaje Multimodales

Los modelos de lenguaje (LLMs, por sus siglas en inglés) representan sistemas computacionales diseñados para procesar, comprender y generar texto en lenguaje natural mediante arquitecturas basadas en aprendizaje profundo. Estos modelos, entrenados en vastos corpus textuales, capturan patrones lingüísticos, semánticos y contextuales, permitiendo tareas como la traducción automática, la generación de respuestas coherentes o el análisis de sentimientos. Su evolución ha dado paso a los modelos de lenguaje multimodales, los cuales extienden estas capacidades al integrar contenido multimedial como imágenes y audio.

Estos nos permiten generar descripciones detalladas de escenas, objetos o composiciones artísticas, traduciendo elementos visuales en narrativas estructuradas. Además, su capacidad multimodal se puede utilizar para procesar entradas de voz y participar en flujos conversacionales voz a voz, facilitando interacciones más naturales y accesibles. Esta convergencia de modalidades (texto, imagen y sonido) facilita la creación de sistemas más ricos y adaptables, capaces de ofrecer interpretaciones contextualizadas y personalizadas según las necesidades de percepción del usuario. Modelos representativos del estado del arte incluyen *Llama V4* [3] (Meta), con codificación multimodal y de código abierto; *GPT-4o* [4] (OpenAI) y *Gemini 1.5* [5] (Google), con integración nativa de texto, imagen y audio.

II.2) Modelos Generativos de Audio

Los modelos generativos de audio representan sistemas capaces de crear sonidos y música a partir de prompts de texto. Estos modelos, basados en arquitecturas de aprendizaje profundo, pueden sintetizar desde sonidos de animales y objetos (e.g. canto de pajaros, truenos, lluvia) hasta composiciones musicales, ajustándose a estilos, géneros o emociones específicas. Su funcionamiento se sustenta en la interpretación de instrucciones textuales o parámetros acústicos, permitiendo generar audio coherente y contextualizado sin necesidad de muestras preexistentes. Un ejemplo relevante es *AudioLDM* [6], que genera audio ambiental mediante modelos de difusión latente condicionados por descripciones textuales, utilizando embeddings CLAP [7] para mejorar la alineación semántica entre texto y sonido.

II.3) Modelos de Text to Speech

Los modelos de *Text to Speech* (TTS) constituyen sistemas computacionales que transforman texto escrito en habla sintética mediante técnicas de procesamiento de lenguaje natural y síntesis de voz. Los modelos de estado del arte están fundamentados en arquitecturas de aprendizaje profundo (incuyendo por lo general el uso de arquitecturas *Difussion* y *Transformers*), analizan la estructura lingüística, semántica y prosódica del input textual para generar audio con entonación, ritmo y naturalidad cercanos a la expresión humana. Su aplicación abarca desde asistentes virtuales y herramientas de accesibilidad hasta sistemas de comunicación aumentativa, permitiendo la generación de voces personalizadas, multilingües y adaptadas a contextos específicos, lo que facilita la interacción con usuarios en entornos digitales y físicos. Arquitecturas de referencia en esta categoría incluyen *Tacotron 2* [8], basado en un encoder-decoder con mecanismo de atención que genera espectrogramas mel de forma autoregresiva, y *VITS* [9], que realiza síntesis end-to-end combinando flujos normalizadores y aprendizaje adversarial. Como implementación de uso práctico, *Kokoro-82M* [10] demuestra que es posible alcanzar voz de alta calidad con solo 82 millones de parámetros, incluyendo soporte para el idioma español.

II.4) Estándares de Accesibilidad

Los estándares de accesibilidad proporcionan el marco normativo que guía el diseño de interfaces inclusivas, asegurando que los contenidos digitales sean utilizables por personas con diversas capacidades sensoriales, motoras o cognitivas. Entre los más relevantes se encuentran las Pautas de Accesibilidad para el Contenido Web (WCAG) [2], que establecen criterios relacionados con la perceptibilidad, operabilidad, comprensibilidad y robustez de los sistemas web. En el contexto chileno, estas directrices se integran mediante la Ley N.° 20.422 [11], que regula la igualdad de oportunidades e incorpora exigencias específicas para tecnologías asistivas. La plataforma desarrollada en este proyecto adopta estos lineamientos para garantizar compatibilidad con lectores de pantalla, navegación por teclado, descripciones alternativas, controles auditivos accesibles y una estructura de interacción que facilite la exploración autónoma.

El cumplimiento de estos estándares es esencial para asegurar que la solución propuesta sea inclusiva y centrada en las necesidades de los usuarios objetivo.

III) Estado del Arte

III.1) Tecnologías y Prácticas en Accesibilidad Cultural y Artística

La accesibilidad al arte ha sido abordada desde múltiples perspectivas, incluyendo prácticas museográficas tradicionales, tecnologías asistivas, proyectos artísticos inclusivos y desarrollos experimentales basados en inteligencia artificial. Su evolución reciente ha estado marcada por la incorporación de herramientas digitales y enfoques multimodales que buscan ampliar el acceso a la experiencia artística para personas con discapacidades sensoriales.

III.1.a Accesibilidad cultural tradicional y práctica museográfica

Las instituciones culturales han implementado históricamente recursos de accesibilidad como audiodescripciones, maquetas táctiles, textos en braille y recorridos guiados adaptados. Estas prácticas están documentadas en guías oficiales de accesibilidad para museos [12], donde se destacan las audioguías adaptadas y las signoguías como mecanismos fundamentales para la inclusión cultural.

A nivel institucional, también se han desarrollado programas específicos de accesibilidad multimedia. Un ejemplo relevante es el programa **Experiencias multimediales MAC para la accesibilidad** [13], impulsado por el Museo de Arte Contemporáneo de la Universidad de Chile, el cual incorpora audioguías, cápsulas audiovisuales subtituladas y contenidos en lengua de señas chilena para ampliar el acceso a sus exposiciones.

III.1.b Proyectos y colectivos de accesibilidad en arte

Existen iniciativas orientadas específicamente a la mediación artística inclusiva. El proyecto **Audioguiarte** [14] desarrolla audioguías personalizadas y recorridos accesibles, combinando narrativas sonoras y contenidos adaptados para públicos diversos.

De forma complementaria, colectivos como **MiméticasLab** [15] exploran la recreación de obras artísticas mediante formatos táctiles y multisensoriales, empleando tecnologías de fabricación digital como impresión 3D para facilitar el acceso de personas con discapacidad visual.

Un caso documentado de implementación museográfica accesible se observa en el Museo Provincial de Bellas Artes Emilio Caraffa (Argentina) [16], donde se desarrolló una sala accesible para personas ciegas mediante códigos QR, audiodescripciones y paisajes sonoros, permitiendo una experiencia estética alternativa.

III.1.c Investigación académica y tecnologías emergentes

En el ámbito académico, se han propuesto sistemas que traducen representaciones visuales a formatos multimodales accesibles. El proyecto **MAIDR** [17] plantea la representación multimodal de visualizaciones estadísticas mediante texto, audio y otros canales, con el objetivo de facilitar su interpretación por parte de personas con discapacidad visual.

Asimismo, investigaciones recientes [18] exploran la generación automática de música a partir de obras pictóricas, buscando traducir atributos visuales y emocionales en experiencias sonoras accesibles.

Estas propuestas, si bien prometedoras, se presentan mayoritariamente como prototipos experimentales sin una integración directa en plataformas de uso cultural generalizado.

III.1.d Plataformas digitales de difusión cultural

Plataformas digitales como **Smartify** y **Google Arts & Culture** han ampliado el acceso al patrimonio artístico mediante la digitalización de colecciones y el uso de dispositivos móviles para contextualizar obras de arte. **Smartify** [19] permite el reconocimiento de obras mediante imágenes y la entrega de información textual y auditiva asociada.

Por su parte, **Google Arts & Culture** [20] ofrece recorridos virtuales y colecciones digitales de museos de todo el mundo, facilitando el acceso remoto al arte, aunque sin un foco específico en accesibilidad profunda o personalización multimodal.

III.2) Comparación Cruzada de Alternativas

En Tabla 1 se sintetiza las principales categorías de soluciones revisadas, comparando sus iniciativas representativas, las modalidades que emplean y sus limitaciones.

Categoría	**Ejemplos / Iniciativas**	**Modalidades**	**Limitaciones**
Accesibilidad museográfica tradicional	• Audioguías • Signoguías • Braille	• Audio • Táctil • Texto	Escalabilidad limitada y alta dependencia de intervención humana
Programas institucionales	Experiencias multimediales (MAC)	• Audio • Video • Lengua de señas	Alcance restringido a exposiciones específicas
Proyectos inclusivos independientes	Audioguiarte	Audio narrativo	No universalizado ni automatizado
Fabricación digital accesible	MiméticasLab	• Táctil • Multisensorial	Requiere recursos físicos y producción manual
Investigación académica	MAIDR, arte a música	• Texto • Audio	Prototipos experimentales sin despliegue productivo
Plataformas digitales generalistas	• Smartify • Google Arts & Culture	• Imagen • Texto • Audio	Baja personalización orientada a accesibilidad

Tabla 1: Comparación del Estado del Arte en Accesibilidad Artística

A pesar de los avances en tecnologías de accesibilidad artística, persiste una brecha significativa en la integración de flujos automatizados multimodales para la producción de contenido. Las soluciones actuales suelen requerir procesos manuales o semiautomáticos, donde la generación de descripciones auditivas, representaciones táctiles o traducciones sonoras se realiza de forma aislada, sin una coordinación eficiente entre modalidades. Esta fragmentación limita la escalabilidad y la coherencia en la producción de experiencias artísticas accesibles, dificultando su adopción masiva en entornos culturales.

III.3) Modelos de Lenguaje Multimodales

Durante los últimos años los modelos múltimodales de lenguaje han experimentado avances significativos, integrando capacidades de procesamiento de texto, imagen, audio y video. Este progreso ha sido impulsado por arquitecturas unificadas basadas en Transformers que permiten compartir un espacio de representación común entre modalidades. Avances como los *Vision-Language Encoders (VLMs)* [21] y la *Tokenización Unificada* [22] han permitido integrar imágenes y texto como secuencias compatibles.
Algunos modelos destacados a la fecha son:

- **Llama v4 (Meta)** [3]: Modelo multimodal con pesos liberados capaz de elaborar descripciones semánticas de imágenes por medio de codificación con *MetaCLIP*.
- **GPT-4o (OpenAI)** [4] **y Gemini 1.5 (Google)** [5]: Modelos con integración nativa de texto, imagen y audio. Sin pesos liberados.

Llama v4 [3] fue seleccionado para las pruebas de concepto de este trabajo, debido a la combinación de tres factores: disponibilidad de pesos abiertos, soporte nativo para codificación de imágenes mediante MetaCLIP, y acceso a través de proveedores de inferencia en la nube a coste reducido —ventajas que los modelos privativos como GPT-4o o Gemini 1.5 no ofrecen simultáneamente.

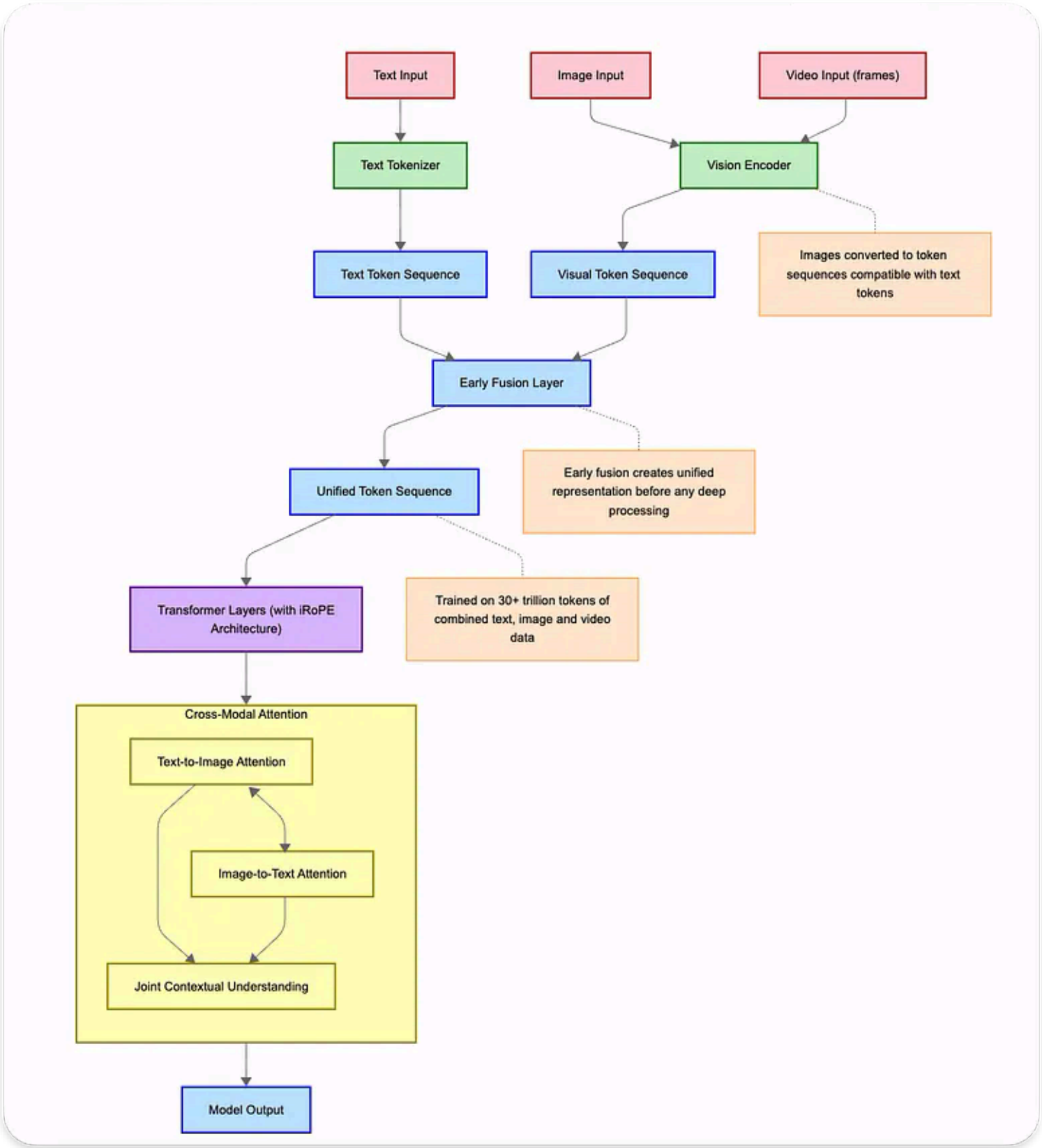


Figura 1: Codificación Multimodal Llama V4

En Figura 1 se demuestra la funcionalidad de la **codificación multimodal con fusión temprana** de *Llama v4* [3], esta se puede apreciar en un esquema vertical que comprende:

- **Parte superior (etapa de entrada).** Se evaluan tres tipos de entradas: **Text Input**, **Image Input** y **Video Input (frames)**. El texto desciende hacia un **Text Tokenizer**, que lo transforma en una **Text Token Sequence**. En paralelo, las imágenes y los fotogramas de video pasan por un **Vision Encoder** basado en un modelo de visión (alineado con Llama), que genera una **Visual Token Sequence**. Estas representaciones visuales se convierten en secuencias de tokens que pueden integrarse con los tokens de texto, permitiendo un espacio de representación común antes del procesamiento conjunto.
- **Zona media (fusión temprana).** Más abajo, ambas secuencias (la textual a la izquierda y la visual a la derecha) confluyen en el bloque central denominado **Early Fusion Layer**. Esta capa produce una **Unified Token Sequence**. En paralelo, la unificación ocurre antes de cualquier procesamiento profundo: las distintas modalidades se concatenan y se procesan conjuntamente desde los primeros bloques del modelo.
- **Sección inferior (procesamiento profundo y output).** La secuencia unificada alimenta las **Transformer Layers (with iRoPE Architecture)**. Estas capas operan de forma conjunta sobre todos los tokens y aplican el proceso de **Cross-Modal Attention**, donde se explicitan dos direcciones complementarias: **Text-to-Image Attention** e **Image-to-Text Attention**. Ambas interacciones confluyen en un nivel de **Joint Contextual Understanding**, representando la construcción de una comprensión contextual compartida entre modalidades. Finalmente, de los mecanismos de atención del transformer se obtiene la salida final del modelo en modalidad de predicción de tokens.

III.4) Modelos de Texto a Voz

En los últimos años, arquitecturas basadas en RNNs han ganado relevancia. Estos modelos, generalmente ligeros en cantidad de parámetros, se pueden autohostear con GPUs de grado consumidor e incluso (en algunos casos) directo en CPU a coste de mayor tiempo de inferencia. Algunos aportes importantes son

- **Tacotron 2** [8]: Modelo de síntesis de voz basado en una arquitectura encoder–decoder con mecanismo de atención, en el cual una red recurrente genera espectrogramas mel de manera autoregresiva a partir de secuencias de texto o fonemas, utilizando posteriormente un vocoder neural para la generación de la señal de audio (Figura 2).
- **VITS** [9]: Modelo de síntesis de voz end-to-end que combina un encoder de texto, un modelo variacional con normalizing flows y un generador adversarial para producir directamente la forma de onda de audio, incorporando un mecanismo de alineamiento monotónico latente para modelar la correspondencia entre texto y audio.
- **StyleTTS2** [23]: Introduce una arquitectura de síntesis de voz multi-componente que emplea encoders separados para contenido lingüístico y estilo, junto con un generador adversarial, permitiendo modelar variaciones de prosodia y timbre a través de representaciones latentes explícitas durante el proceso de síntesis.

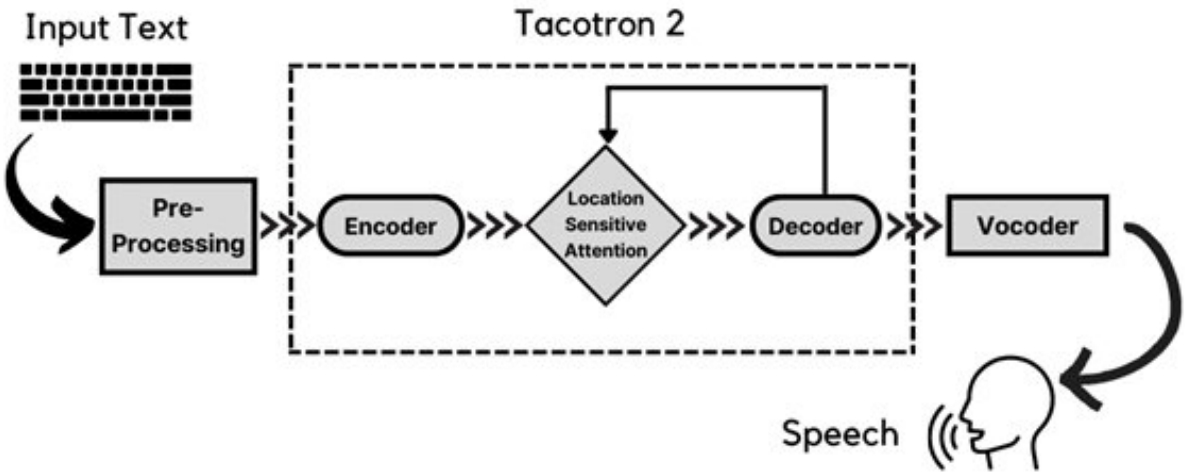


Figura 2: Arquitectura de Modelo Tacotron 2

Kokoro TTS [10], basado en *StyleTTS2* [23], destaca por generar voces de alta calidad con 82 millones de parametros, actualmente soporta 3 voces preentrenadas para el español.

III.5) Modelos Generativos de Ambiente Sonoro

La generación de ambientes sonoros de alta calidad sigue en fase experimental, sin soluciones comerciales consolidadas. Modelos como CLAP permiten mapear texto y audio en un espacio semántico compartido, aunque no generan audio directamente.

Entre los modelos inicialmente considerados para la plataforma se encuentran:

- **I Hear Your True Colors** [24]: La arquitectura propuesta en este paper combina el uso de un VQVAE, transformers y CLIP. El VQVAE extrae representaciones jerárquicas como secuencias discretas, los transformers las modelan de forma autorregresiva y CLIP alinea el audio con lo visual. Aunque el código es abierto, no existen fuentes públicas de pesos preentrenados ni versiones comerciales por parte de proveedores externos.
- **AudioLDM** [6]: Utiliza un modelo de difusión latente para generar espectrogramas de audio condicionados por texto. El sistema emplea un autoencoder variacional (VAE) para proyectar los espectrogramas al espacio latente, donde un modelo de difusión tipo U-Net aprende el proceso generativo. La condición textual se incorpora mediante embeddings CLAP [7], lo que mejora significativamente la alineación semántica entre descripciones textuales y el contenido sonoro generado. Tras el proceso de difusión, las representaciones latentes se decodifican nuevamente a audio (Figura 3). Presenta resultados prometedores en coherencia semántica y diversidad acústica. El modelo preentrenado se encuentra disponible con pesos abiertos.

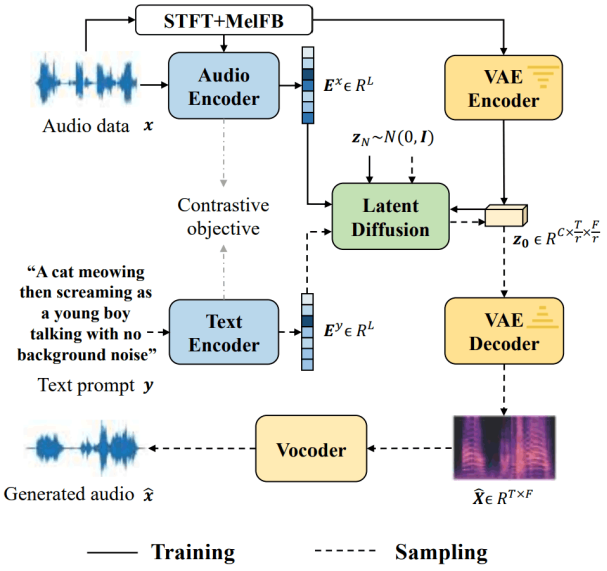


Figura 3: Arquitectura de Modelo AudioLDM

IV) Desarrollo de la Plataforma

IV.1) Toma de Requerimientos

IV.1.a Requisitos Funcionales

- **Catálogo de obras**: El sistema debe construir, procesar y exponer un catálogo digital curado de obras artísticas de dominio público, a partir de fuentes confiables previamente identificadas, incorporando imágenes y metadatos descriptivos (título, autor, técnica, periodo y contexto histórico).
- **Audio descriptivo**: Se deben generar descripciones textuales objetivas de las obras mediante modelos multimodales, y posteriormente convertir dichas descripciones en audio utilizando tecnología de síntesis de voz, validando su calidad con usuarios.
- **Narración contextual**: El sistema debe integrar un módulo automatizado de extracción, síntesis y generación narrativa que, a partir de fuentes históricas autoritativas, produzca narraciones coherentes y estructuradas sobre el contexto histórico y cultural de cada obra, las cuales deben poder reproducirse mediante síntesis de voz.
- **Ambiente sonoro**: La plataforma debe seleccionar, evaluar e integrar modelos generativos de imagen a audio para producir dinámicamente paisajes sonoros asociados a las obras.
- **Interfaz accesible**: El cliente web debe diseñarse e implementarse conforme a estándares de accesibilidad [2], considerando revisión normativa, análisis de buenas prácticas, prototipado accesible y validación con usuarios, garantizando compatibilidad con lectores de pantalla, navegación por teclado y cumplimiento de criterios básicos de accesibilidad.

IV.1.b Requisitos No Funcionales

- **Accesibilidad**: La interfaz debe ser intuitiva y de fácil integración con herramientas de accesibilidad.
- **Modularidad**: Los módulos de generación de contenido deben ser independientes y reemplazables entre sí.
- **Escalabilidad**: La arquitectura debe permitir incorporar nuevas obras sin modificaciones estructurales al sistema.
- **Interoperabilidad**: Los servicios deben exponer sus funcionalidades mediante una API REST con endpoints documentados.

La arquitectura propuesta como solución se fundamenta en un modelo cliente-servidor, decisión que responde a la necesidad de desacoplar las interfaces de usuario de los procesos computacionalmente intensivos asociados a la inferencia de modelos de aprendizaje automático y al procesamiento multimedia. Este enfoque permite, por un lado, mantener una experiencia de usuario fluida en dispositivos con capacidades limitadas y, por otro, facilitar la escalabilidad y mantenibilidad del sistema.

IV.2) API Gateway

Se incorpora una API Gateway como componente central de la arquitectura. La elección de una API Gateway se justifica por tres motivos principales:

- Centralizar el acceso a funcionalidades heterogéneas bajo una interfaz uniforme, reduciendo la complejidad del frontend
- Aislar los servicios internos del sistema, permitiendo su evolución independiente
- Concentrar las tareas de mayor costo computacional en un entorno controlado, optimizado para la ejecución de modelos de IA.

Además, se desarrolla un *frontend* que aplica principios de accesibilidad, con especial atención a las necesidades de usuarios con discapacidad visual.

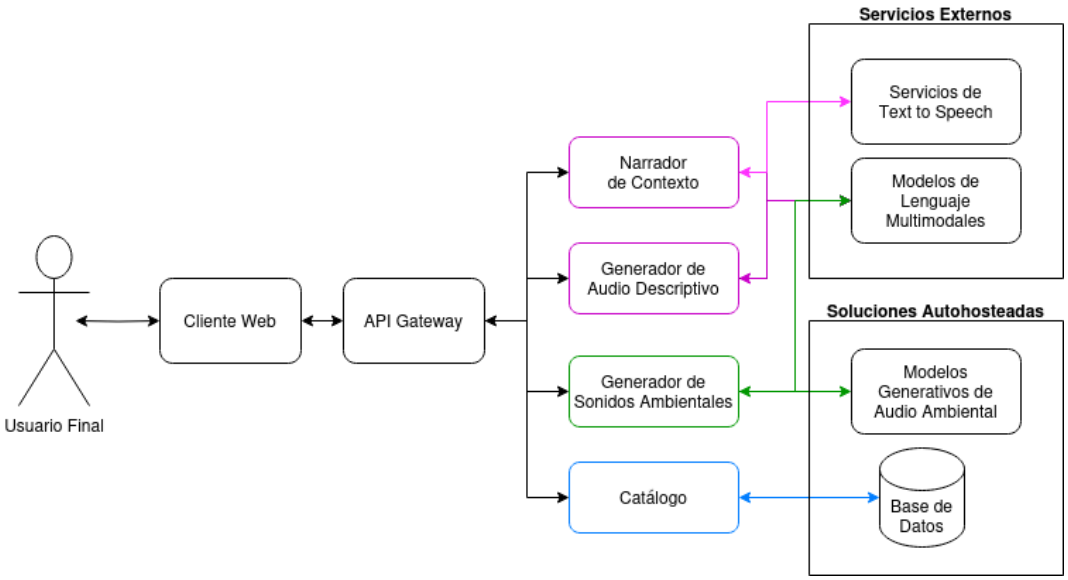


Figura 4: Arquitectura General de Sistema

Se decide utilizar Django como framework por su capacidad para agilizar el desarrollo de APIs robustas mediante su sistema de ORM, su integración directa de utilidades de desarrollo de APIs REST y su ecosistema de paquetes especializados. Además, su soporte integrado para autenticación, manejo de rutas y middleware facilita la implementación de políticas de seguridad y gestión de solicitudes, aspectos críticos para en el futuro robustecer nuestro sistema con flujos de administración de contenido.

Como se observa en Figura 4, la API Gateway orquesta 4 módulos funcionales:

- **Narrador de Contexto**: Encargado de crear narraciones sobre el contexto histórico de la producción de las obras. Utiliza modelos de inferencia externos para producción de su contenido.
- **Generador de Audio Descriptivo**: Encargado de transformar representaciones visuales en audios descriptivos de sus contenidos literales. Al igual que el módulo anterior utiliza servicios de inferencia externos para producir su contenido.
- **Generador de Sonidos Ambientales**: Sintetiza audio contextual apoyado en modelos autoalojados para representar el contenido las obras en un formato interactivo. Se apoya en servicios de inferencia externos para extraer caracteristicas de imagenes.
- **Catálogo**: Gestiona los recursos del sistema y su persistencia en la base de datos.

Esta separación permite integrar tanto servicios externos especializados como soluciones autoalojadas dentro de una misma arquitectura, manteniendo un flujo desacoplado en el que cada componente cumple una función específica pero coordinada. De este modo, se favorece la escalabilidad, la mantenibilidad y la futura incorporación de nuevos modelos o proveedores tecnológicos.

A continuación una explicación detallada de cáda módulo:

IV.2.a Generador de Audio Descriptivo

Este módulo se encarga de producir descripciones textuales detalladas de los contenidos visuales presentes en las obras. Para ello, emplea modelos de lenguaje multimodales, en particular *Llama V4 Maverick* [3] para evaluar entrada de imagenes y texto.
La imagen se evalua junto a un *prompt* diseñado para extraer información de los contenidos de la obra, además se añaden instrucciones para mantener una cadencia de redacción fluida (Anexo / Código 13).
Posteriormente, las descripciones generadas son convertidas a formato de audio mediante sistemas de síntesis de voz (TTS) (Anexo / Código 19) utilizando *Kokoro 82M* [10].

IV.2.b Generador de Narraciones de Contexto

Este módulo genera narraciones auditivas que contextualizan históricamente cada obra. Para ello, se recopila información de fuentes autoritativas (en este caso, se procesó manualmente un conjunto de datos con artículos de Wikipedia vinculados a cada pieza) (Anexo / Código 8). A partir de estos datos, se elabora un relato contextualizado y adaptado al caso de uso (Anexo / Código 14), que posteriormente se convierte en audio mediante modelos de síntesis de voz (Text to Speech) (Anexo / Código 19).

IV.2.c Generador de Sonidos Ambientales

Este módulo se encarga de la generación de ambientes sonoros que representan los contenidos literales (no abstractos) de sus imagenes de entrada. La heurística utilizada consiste en:

- Dividir el espacio en cuadrantes, en nuestro caso 9 (Anexo / Código 15), como se puede apreciar en Figura 5.
- Procesar cada cuadrante mediante modelos de lenguaje multimodales (Anexo / Código 16). Por razones de operatividad con modelos de lenguaje y facilidad de interpretación se decide procesar la salida de esta parte en formato JSON.
- Por cada cuadrante generar una mezcla de sonido integrando todos los elementos detectados en cada respectiva sección (Anexo / Código 17).



Figura 5: Segmentación por Cuadrantes de Obra «El Aquelarre»

IV.2.d Catálogo de Obras/Imágenes

Se elaboró un conjunto de datos que incluye 30 obras artísticas [25], conteniendo todo el material requerido para generar el contenido de la plataforma. Este dataset incorpora, además, enlaces a los artículos de Wikipedia asociados a cada obra artística para tener referencias autoritativas en la generación de narraciones historicas sobre la producción de las mismas. Se escoge formato JSON por facilidad de integración nativa en python. El Listado 1 muestra la estructura de una entrada del dataset: cada registro contiene los campos de identificación de la obra (`author`, `title`, `date`, `technique`, `period`), una descripción de sus elementos visuales (`elements`), el contexto histórico (`context`), y los enlaces a su artículo de Wikipedia y a la imagen de dominio público (`link`, `im_link`).

```
  [
    {
      "author": "Pieter Bruegel the Elder",
      "title": "The Hunters in the Snow",
      "technique": "óleo sobre panel",
      "period": "Renacimiento",
      "elements": "Tres cazadores cansados con perros...",
      "context": "Parte de la serie 'Meses/Estaciones'...",
      "link": "https://en.wikipedia.org/wiki/The_Hunters_in_the_Snow",
      "im_link": "https://upload.wikimedia.org/wikipedia/commons/d/d8/
Pieter_Bruegel_the_Elder_-_Hunters_in_the_Snow_%28Winter%29_-_Google_Art_
Project.jpg",
      "date": 1565
    },
    ...
]
```

Listado 1: Muestra de Dataset

Una vez procesado el conjunto de datos, sus campos son expuestos a través de la **API** mediante endpoints específicos. Los datos textuales se almacenan en una base de datos SQLite, mientras que las imágenes y archivos de audio generados se guardan en el sistema de archivos local del backend.

IV.2.e Modelo de Datos

Se organiza la información artística mediante un conjunto de entidades interrelacionadas que permiten representar tanto los metadatos estructurales de una obra como sus componentes descriptivos y narrativos. Diagrama UML del modelo se puede ver en Figura 6.

1. **Artwork**

   Entidad central del modelo, representa una obra artística individual. Almacena información general y contextual de la obra, incluyendo su título, fecha de creación y contexto histórico.

   Se relaciona con todas las entidades del modelo de datos.

   Campos:

   - `id`
   - `title`
   - `date`
   - `historic_context`
   - `author_id`
   - `technique_id`
   - `style_id`

1. **Author** Representa al autor o creador de una o más obras artísticas. Un autor puede estar asociado a múltiples obras, estableciendo una relación de tipo 1–N con **Artwork**.

   Campos:

   - `id`
   - `name`
   - `description`

2. **Technique** Describe la técnica artística utilizada en la creación de una obra. Una técnica puede ser utilizada por múltiples obras, manteniendo una relación 1–N con la entidad **Artwork**.

   Campos:

   - `id`
   - `name`
   - `description`

3. **Style** Define el estilo artístico al que pertenece una obra. Un estilo puede asociarse a múltiples obras, estableciendo una relación 1–N con **Artwork**.

   Campos:

   - `id`
   - `name`
   - `description`

4. **Artwork Ambience** Representa elementos de ambientación asociados a una obra, permitiendo modelar componentes espaciales o interactivos mediante coordenadas bidimensionales. Se relaciona con **Artwork** a través de una relación 1–N.

   Campos:

   - `id`
   - `x_pos`
   - `y_pos`
   - `artwork_id`

5. **Artwork Narration** Contiene fragmentos narrativos o explicativos asociados a una obra. Una obra puede poseer múltiples narraciones, cada una vinculada mediante una relación de tipo 1–N.

   Campos:

   - `id`
   - `content`
   - `artwork_id`

6. **Artwork Description** Almacena descripciones textuales de los contenidos de una obra. Se relaciona con **Artwork** mediante una relación de tipo 1–N.

   Campos:

   - `id`
   - `content`
   - `artwork_id`

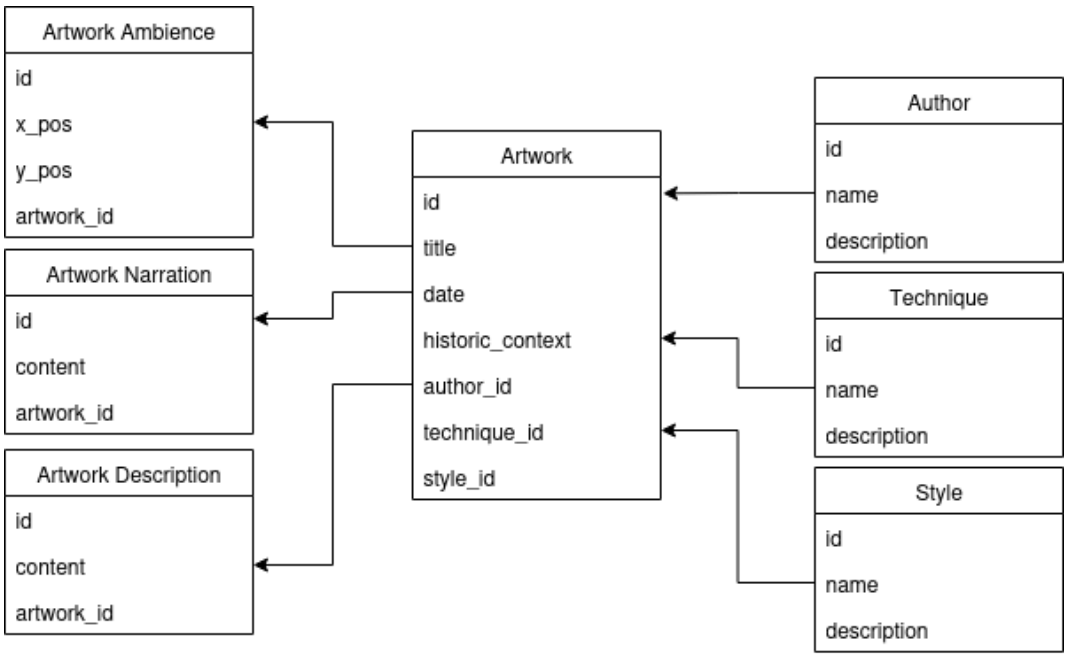


Figura 6: Modelo de Datos de la Plataforma

IV.3) Frontend

El frontend se implementó utilizando NextJS para proporcionar acceso al contenido gestionado por la API Gateway. En su diseño, se incorporaron textos alternativos y etiquetas HTML descriptivas con el objetivo de garantizar una integración completa con herramientas de asistencia para usuarios con discapacidad visual. Asimismo, se desarrolló una interfaz de usuario que facilita la navegación mediante atajos de teclado, optimizando la accesibilidad y usabilidad del sistema.

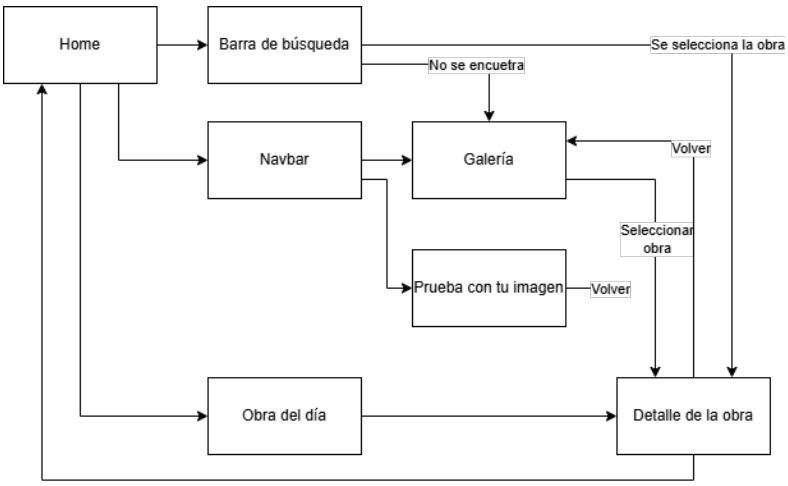


Figura 7: Flujo de Navegación Frontend

IV.4) Limitaciones

Por limitaciones computacionales durante el desarrollo, los módulos no se integraron directamente en la API Gateway. En su lugar, se implementó un prototipo funcional en un cuadernillo Jupyter [26] para validar el diseño y flujo de procesamiento. Esto permitió generar y preprocesar los conjuntos de datos necesarios, que fueron incorporados a la base de datos para su uso en la versión final.

V) Resultados

V.1) API Gateway

El sistema opera correctamente, incluye autodocumentación en *Swagger* y *OpenAPI* y un conjunto completo de operaciones CRUD para todos los modulos de la plataforma.
Debido a las limitaciones computacionales mencionadas previamente, los endpoints solamente habilitan operaciones de lectura y escritura en la base de datos y sistema de archivos para facilitar la subida de elementos preprocesados.
El sistema permite tanto la consulta general de los elementos registrados (obras, autores, períodos históricos, entre otros) como la gestión individual de estos en cada uno de los módulos correspondientes.

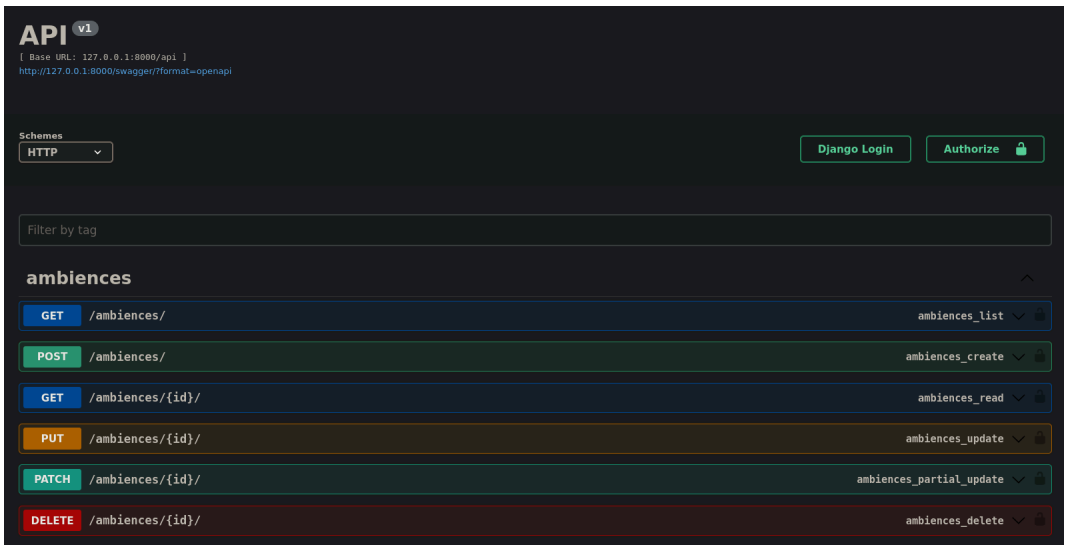


Figura 8: Documentación Swagger de la API

Para garantizar compatibilidad con APIs externas de servicios de cómputo, los datos no estructurados (imágenes y audio) se transmiten codificados en base64. Este método, aunque aumenta el tamaño de los datos enviados (alrededor del 33%), facilita la integración con SDKs de proveedores de inferencia compatibles con la API de OpenAI (como Groq, Novita o TogetherAI). Como mejora futura, se propone adoptar multipart-form-data en la subida y carga de archivos para optimizar el rendimiento de la plataforma.

V.2) Frontend

Aunque no se realizaron pruebas con los usuarios objetivo, se evaluó el funcionamiento de la navegación dentro del sitio mediante herramientas de accesibilidad diseñadas para usuarios con discapacidad visual.

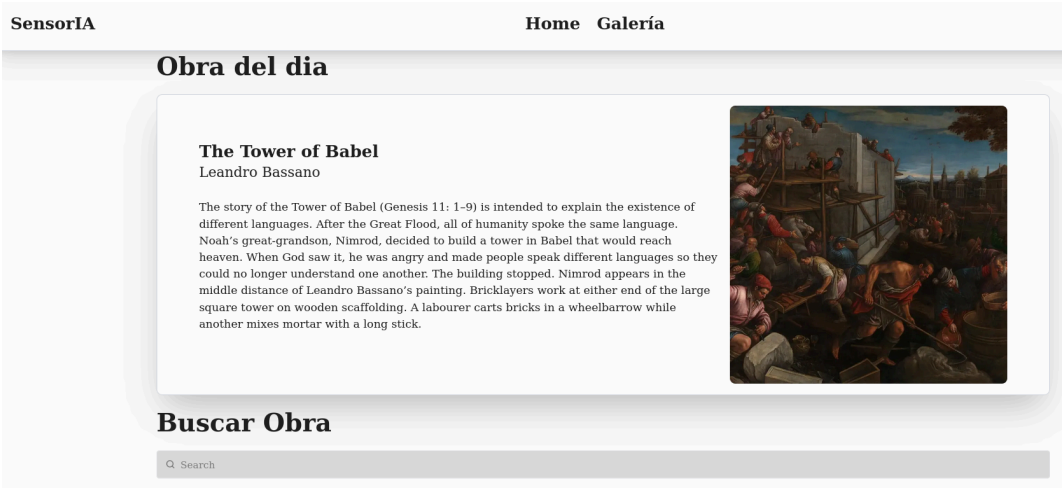


Figura 9: Frontend de la Plataforma

La funcionalidad de navegación implementa correctamente búsquedas por obra, autor, período y técnica. Sin embargo, el sistema emplea un mecanismo de coincidencia de strings que espera un match de acentuación, lo que hace rígida la utilización de la barra de busqueda.

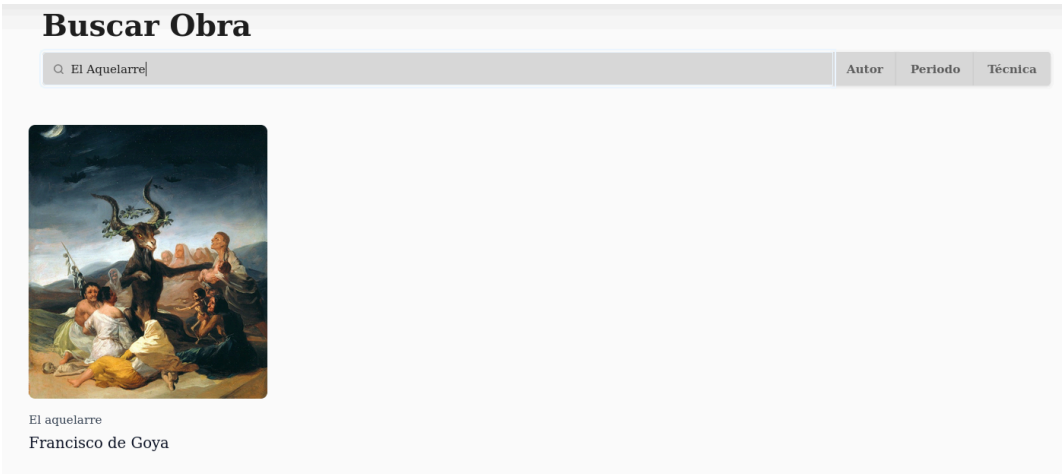


Figura 10: Funcionalidad de Búsqueda en Frontend

Cada input cuenta con placeholders, lo que facilita la integración con utilidades de accesibilidad para navegador web.

En la visualización directa de la obra se muestran los metadatos generales de la obra (título, autor, fecha, período y técnica) junto al contenido de los módulos de *Narración* y *Descripción* con sus audios procesados con TTS.

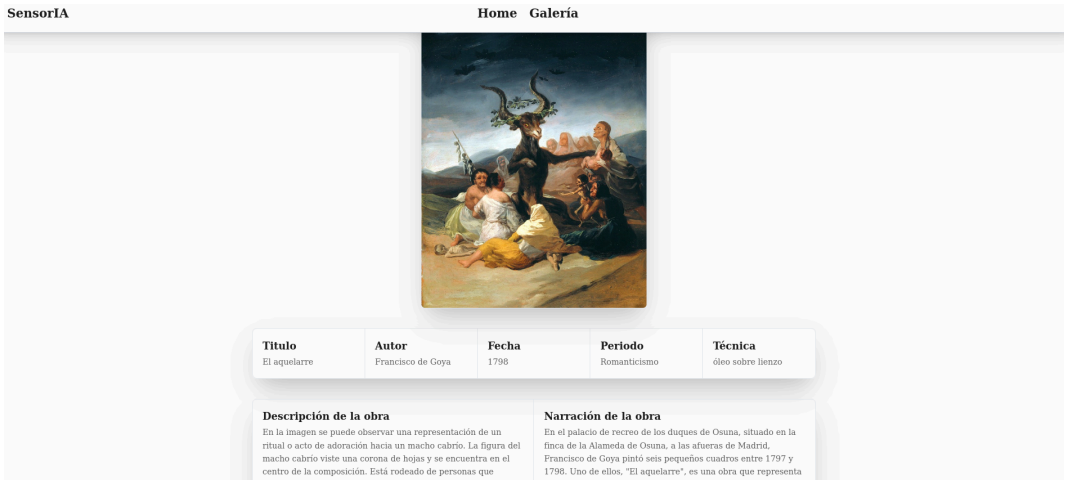


Figura 11: Vista General de Obra en Frontend

El acceso a los ambientes sonoros de la obra se realiza mediante un click o la selección de la misma. Sin embargo, este elemento carece de un etiquetado adecuado que indique claramente su funcionalidad, lo que dificulta su uso para personas con discapacidad visual. Como mejora pendiente, se propone implementar un etiquetado que permita identificar de manera explícita los contenidos y la interacción asociada a esta componente.

Al visualizar los ambientes de sonido basta con hacer click o seleccionar cualquiera de los 9 cuadrantes para reproducir el paisaje sonoro asociado a dicha sección.

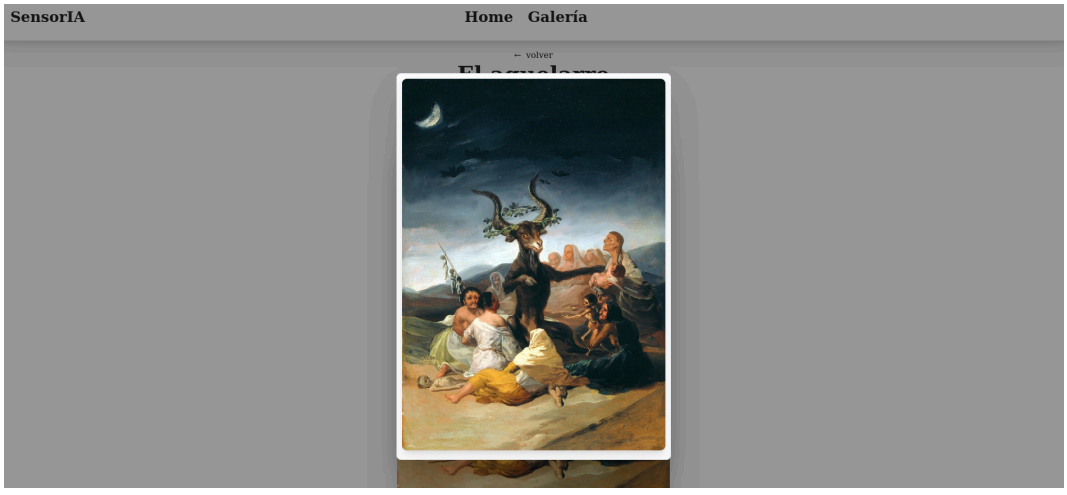


Figura 12: Vista de Ambientes Sonido en Frontend

V.3) Módulos Generativos

A continuación se exponen los resultados obtenidos en los módulos generativos de la plataforma. Tal como se indicó en la sección anterior, estas componentes fueron implementadas en un entorno independiente [26] como parte de un desarrollo modular complementario al sistema principal.

Como caso de estudio representativo, se ha seleccionado la obra **El Aquelarre** de Francisco de Goya para ilustrar el funcionamiento de los módulos.



Figura 13: Obra «El Aquelarre» de Francisco de Goya

A continuación, se evalúan los siguientes aspectos:

- La calidad del texto en los módulos de *Audio Descriptivo* y *Narraciones de Contexto*.
- La fidelidad y realismo del audio producido por el *Generador de Sonidos Ambientales*.
- La naturalidad y claridad de la voz sintetizada empleada en las pruebas de concepto.

V.3.a Generación de Texto de Audio Descriptivo

Los resultados obtenidos demuestran una notable correspondencia entre las imágenes de entrada y las descripciones generadas, las cuales mantienen una estructura sintáctica y una cadencia adecuadas para su posterior procesamiento mediante modelos de *Text to Speech*. Este hallazgo sugiere que una estrategia basada en *prompting directo* resulta suficiente para que un modelo de lenguaje multimodal genere descripciones precisas del contenido visual de una obra, sin requerir fuentes autoritativas externas (como artículos de Wikipedia) de referencia para complementar el análisis.

El Aquellare:

```
En la imagen se puede observar una representación de un ritual o acto de
adoración hacia un macho cabrío. La figura del macho cabrío viste una
corona de hojas y se encuentra en el centro de la composición. Está
rodeado de personas que parecen estar participando en algún tipo de
ceremonia.

Al fondo de la escena se puede ver una luna creciente en el cielo
nocturno, lo que sugiere que la escena se desarrolla durante la noche.
Detrás del macho cabrío hay un grupo de personas vestidas con túnicas,
algunas de las cuales tienen la cara cubierta.

La escena general parece representar una temática relacionada con
prácticas antiguas o rituales paganos.
```

Texto 2: Descripción de Obra El Aquelarre de Francisco de Goya

La suficiencia de los modelos de lenguaje sin necesidad de incluir información extraída de fuentes autoritativas contrasta con lo observado en el siguiente módulo.

V.3.b Generación de Texto de Narraciones de Contexto

Como es señalado en Anexo / Código 14, para evitar alucinaciones y producir información rica en contexto se toma por entrada un articulo de Wikipedia, para el caso ilustrado:

https://es.wikipedia.org/wiki/El_aquelarre_(1798)

Una comparación cruzada entre el texto generado y la información contenida en el artículo de Wikipedia demuestra una adecuada correspondencia con la fuente original, logrando transformar el contenido enciclopédico en una narración fluida y de carácter natural.
El pipeline genera narraciones contextuales y evita alucinaciones vistas en prototipos anteriores de este módulo al utilizar consultas directas a modelos de lenguaje.
Un trabajo pendiente es la corrección de casos borde, en particular anglicismos y siglas.

El Aquellare:

```
En el palacio de recreo de los duques de Osuna, situado en la finca de la
Alameda de Osuna, a las afueras de Madrid, Francisco de Goya pintó seis
pequeños cuadros entre 1797 y 1798. Uno de ellos, "El aquelarre", es una
obra que representa un ritual de brujería presidido por un gran macho
cabrío, forma del demonio, rodeado de brujas ancianas y jóvenes que
entregan niños, según la superstición de la época, para alimentar al
demonio. La escena se desarrolla bajo un cielo nocturno iluminado por la
luna, con animales nocturnos volando en el fondo, creando una atmósfera
siniestra y oscura.
La serie a la que pertenece "El aquelarre" forma parte de la estética del
"sublime terrible", una corriente del prerromanticismo que buscaba generar
desasosiego y una sensación de pesadilla en el espectador. Goya estaba
trabajando en "Los Caprichos" en ese momento, una obra estrechamente
relacionada temáticamente con la serie de cuadros para los duques de
Osuna. La brujería era un tema de actualidad entre los ilustrados
españoles amigos de Goya, y su interés en este tema se refleja en la
ambientación nocturna y los tonos oscuros que predominan en "El
aquelarre". La obra forma parte de la colección de la Fundación Lázaro
Galdiano, después de ser adquirida por José Lázaro Galdiano en 1928.
```

Texto 3: Narración de Obra El Aquelarre de Francisco de Goya

V.3.c Generador de Sonidos Ambientales

El proceso de detección de elementos acústicos en la obra presenta un desempeño adecuado, aunque se observan alucinaciones ocacionales en la identificación de componentes específicos, como puede apreciarse en el objeto detectado en segunda imagen de Tabla 2. No obstante, por lo general los elementos detectados mantienen coherencia temática con el contenido representado.

Segmento	Sonidos Generados
	**Objeto** goat bleating **Fondo** tree branch creaking, people whispering **Audio** https://youtu.be/dqOECGiEiBI
	**Fondo** sand **Objeto** puppy sleeping **Audio** https://youtu.be/mT7UBvVd9m8

Tabla 2: Muestra de Elementos Detectados en Obra «El Aquelarre»

Respecto a la calidad del audio generado se puede decir que, aunque la mayoría de los cuadrantes producen resultados verosímiles y coherentes con las escenas representadas, se observa un grado variable de ruido y artefactos en las mezclas de sonido y se identifican *outliers* donde los ambientes degeneran en ruido. Este comportamiento sugiere limitaciones en la generalización del modelo ante elementos visuales ambiguos y/o composiciones atípicas dentro de las obras.

V.3.d Audio Generado con TTS

La implementación del módulo de Text-to-Speech (TTS) (Anexo / Código 19) utilizando *Kokoro82M* [10] demostró un nivel de naturalidad adecuado para la generación de audio en español, con una entonación fluida y una pronunciación generalmente clara de los textos descriptivos y narrativos. El tono neutral adoptado por el modelo resulta apropiado para una aplicación de accesibilidad cultural, ya que evita sesgos emocionales que podrían interferir con la percepción objetiva del contenido.

No obstante, se identificaron dificultades específicas en la pronunciación de anglicismos y siglas. Términos y nombres propios de otros idiomas son interpretados con reglas fonéticas del español, generando resultados poco intuitivos para el oyente. Esta limitación sugiere la necesidad de implementar un preprocesamiento que genere descripciones alternativas para dicho contenido (lo que fué implementado parcialmente en las estrategias de prompting de la generación de narraciones de contexto, ver Anexo / Código 14) o, alternativamente, incorporar un diccionario de excepciones para preservar la inteligibilidad en contextos técnicos.

Los audios generados para la descripción y la narración de «El Aquelarre» se pueden encontrar en los siguientes vínculos:

- **Descripción**: https://youtu.be/7NwKlbHQKeo
- **Narración**: https://youtu.be/P9EOz7Au4ZA

VI) Conclusiones

Los resultados obtenidos evidencian la utilidad de los modelos de lenguaje multimodales para casos de uso de accesibilidad, al demostrar que estos sistemas son capaces de interpretar de manera integrada información visual y textual y transformarla en representaciones comprensibles para usuarios con discapacidad visual. En particular, la capacidad de los modelos para generar descripciones detalladas y coherentes a partir de imágenes artísticas complejas permite sustituir, en gran medida, la percepción visual directa por una mediación auditiva rica en contenido semántico. Asimismo, la adaptación del estilo narrativo y la cadencia del texto generado facilita su posterior conversión a audio mediante sistemas de síntesis de voz, asegurando una experiencia de escucha fluida y cognitivamente accesible.

Adicionalmente, el uso de modelos multimodales reduce la dependencia de anotaciones manuales extensivas y de metadatos previamente estructurados, lo que representa una ventaja significativa en términos de escalabilidad y mantenimiento de plataformas culturales accesibles. La posibilidad de combinar análisis visual, generación de lenguaje natural y producción de audio en un mismo flujo de procesamiento permite construir soluciones integrales que responden a diversas necesidades de accesibilidad, desde la descripción literal de una obra hasta la contextualización histórica y la identificación de elementos acústicos. En conjunto, estos resultados respaldan el potencial de los modelos de lenguaje multimodales como herramientas clave para democratizar el acceso al patrimonio cultural, ampliando la inclusión y la participación de públicos tradicionalmente excluidos de la experiencia artística visual.

Por otra parte, la utilización de modelos generativos de audio, si bien presenta limitaciones en su capacidad de generalización y estabilidad, demuestra un alto potencial para la creación de contenido apreciativo e inmersivo; no obstante, los resultados obtenidos ponen de manifiesto la necesidad de incorporar mecanismos de revisión y control de calidad con participación humana (human-in-the-loop) para garantizar la coherencia, fidelidad y utilidad final del contenido generado.

VII) Bibliografía

[1] UNESCO, «Implementation of the UN Convention on the Rights of Persons with Disabilities in the area of art and culture (Article 30)». [En línea]. Disponible en: https://www.unesco.org/creativity/en/policy-monitoring-platform/implementation-un-convention-rights-persons-disabilities-area-art-and-culture-article-30

[2] W3C, «Web Content Accessibility Guidelines (WCAG) 2.1». [En línea]. Disponible en: https://www.w3.org/TR/WCAG21/

[3] Meta, «Llama 4: Multimodal Intelligence». [En línea]. Disponible en: https://ai.meta.com/blog/llama-4-multimodal-intelligence/

[4] OpenAI *et al.*, «GPT-4o System Card». [En línea]. Disponible en: https://arxiv.org/abs/2410.21276

[5] G. Team *et al.*, «Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context». [En línea]. Disponible en: https://arxiv.org/abs/2403.05530

[6] H. Liu *et al.*, «AudioLDM: Text-to-Audio Generation with Latent Diffusion Models». [En línea]. Disponible en: https://arxiv.org/abs/2301.12503

[7] B. Elizalde, S. Deshmukh, M. A. Ismail, y H. Wang, «CLAP: Learning Audio Concepts From Natural Language Supervision». [En línea]. Disponible en: https://arxiv.org/abs/2206.04769

[8] J. Shen *et al.*, «Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions». [En línea]. Disponible en: https://arxiv.org/abs/1712.05884

[9] J. Kim, J. Kong, y J. Son, «Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech». [En línea]. Disponible en: https://arxiv.org/abs/2106.06103

[10] hexgrad, «Kokoro-82M». [En línea]. Disponible en: https://huggingface.co/hexgrad/Kokoro-82M

[11] Biblioteca del Congreso Nacional de Chile, «Ley 20.422: Establece normas sobre igualdad de oportunidades e inclusión social de personas con discapacidad». [En línea]. Disponible en: https://www.bcn.cl/leychile/navegar?idNorma=1010903

[12] Ayuntamiento de Málaga, «Guía de accesibilidad en museos». [En línea]. Disponible en: https://www.malaga.eu/recursos/sociales/accesibilidad/guias/ocio_deporte/museos.html

[13] Museo de Arte Contemporáneo de la Universidad de Chile, «MAC pone en marcha programa de experiencias multimediales para la accesibilidad». [En línea]. Disponible en: https://mac.uchile.cl/mac-pone-en-marcha-programa-experiencias-multimediales/

[14] Audioguiarte, «Proyectos de accesibilidad cultural». [En línea]. Disponible en: https://audioguiarte.com/proyectos/

[15] Ver Sin Límites Accesibilidad, «MiméticasLab: arte accesible para personas con discapacidad visual». [En línea]. Disponible en: https://versinlimitesaccesibilidad.com/mimeticaslab-arte-accesible-personas-discapacidad-visual/

[16] El País, «El arte colectivo de traducir las obras de un museo para ciegos». [En línea]. Disponible en: https://elpais.com/america-futura/2025-04-19/el-arte-colectivo-de-traducir-las-obras-de-un-museo-para-ciegos.html

[17] Y. Wang y others, «MAIDR: Making Statistical Visualizations Accessible with Multimodal Data Representation». [En línea]. Disponible en: https://arxiv.org/abs/2403.00717

[18] H. Zhang y others, «Bridging Paintings and Music: Emotion-Based Music Generation through Paintings». [En línea]. Disponible en: https://arxiv.org/abs/2409.07827

[19] Wikipedia contributors, «Smartify». [En línea]. Disponible en: https://en.wikipedia.org/wiki/Smartify

[20] Google, «Google Arts & Culture». [En línea]. Disponible en: https://artsandculture.google.com/

[21] T. Zhang *et al.*, «Vision-Language-Vision Auto-Encoder: Scalable Knowledge Distillation from Diffusion Models». [En línea]. Disponible en: https://arxiv.org/abs/2507.07104

[22] C. Ma *et al.*, «UniTok: A Unified Tokenizer for Visual Generation and Understanding». [En línea]. Disponible en: https://arxiv.org/abs/2502.20321

[23] Y. A. Li, C. Han, V. S. Raghavan, G. Mischler, y N. Mesgarani, «StyleTTS 2: Towards Human-Level Text-to-Speech through Style Diffusion and Adversarial Training with Large Speech Language Models». [En línea]. Disponible en: https://arxiv.org/abs/2306.07691

[24] R. Sheffer y Y. Adi, «I Hear Your True Colors: Image Guided Audio Generation». [En línea]. Disponible en: https://arxiv.org/abs/2211.03089

[25] Alexey Mitjaew, «Artwork Sensotia Dataset». [En línea]. Disponible en: https://www.kaggle.com/datasets/alexeymitjaew/artwork-sensotia

[26] Alexey Mitjaew, «SensorIA Generation Pipeline». [En línea]. Disponible en: https://www.kaggle.com/code/alexeymitjaew/sound-generation-pipeline

VIII) Anexo

VIII.1) Utilidades Generales

En nuestra prueba de concepto se emplearon modelos de lenguaje proporcionados por la plataforma *Groq*, accedidos mediante el cliente de Python de OpenAI. Este estándar es compatible con la mayoría de los proveedores de servicios de inferencia para modelos de lenguaje.

```python
client = openai.OpenAI(
    base_url = "https://api.groq.com/openai/v1",
    api_key = user_secrets.get_secret("GROQ_API_KEY")
)
```

Anexo / Código 4: Inicialización de Cliente

Adicionalmente a lo anterior, se implementó una abstracción genérica para el procesamiento de **prompts**. Esta función permite definir parámetros mediante notación de llaves (por ejemplo, `{clave}`) para su posterior sustitución mediante argumentos nombrados al invocar la función (ejemplo: `PARSE_PROMPT(prompt, clave=valor)`).

```python
def PARSE_PROMPT(BASE_PROMPT, **kwargs):
    for key in kwargs:
        BASE_PROMPT = BASE_PROMPT.replace(
            '{' + str(key) + '}',
            str(kwargs[key])
        )
    return BASE_PROMPT
```

Anexo / Código 5: Utilidad de Procesado de Prompts

VIII.2) Pipeline de Extracción de Fuentes

Para extraer información procesada de fuentes autoritativas se procedió con el desarrollo de utilidades para extraer datos desde una URL (Anexo / Código 6) y procesarlos en forma de *bullet points* para presentar los resultados de manera estructurada y clara (Anexo / Código 8).

```python
def extract_text_from_url(url: str) -> str:
    headers = {
        "User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0",
    }
    response = requests.get(url, headers=headers).raise_for_status()
    soup = BeautifulSoup(response.text, 'html.parser')

    for tag in soup(['script', 'style']):
        tag.decompose()

    paragraphs = [p.get_text(strip=True) for p in soup.find_all('p')]
    return '\n\n'.join(paragraphs)
```

Anexo / Código 6: Scraper Extractor de Información

```python
SOURCE_EXTRACTOR_PROMPT = """Eres un extractor de información precisa y concisa. Tu tarea es leer el texto de un artículo y devolver una lista de viñetas (•) con todos los hechos y datos relevantes, escritos en español natural y claro.

Instrucciones:
1. Identifica los hechos principales, cifras, declaraciones, fechas, nombres y conclusiones clave del texto.
2. No incluyas opiniones, lenguaje publicitario o frases irrelevantes.
3. Si el texto está en inglés u otro idioma, traduce los puntos al español correctamente.
4. Usa frases breves pero completas, cada una iniciando con un punto (•).
5. Mantén el tono informativo y objetivo.

Texto del artículo: {TEXT}
Tu respuesta:"""
```

Anexo / Código 7: Prompt de Extracción de Información

```python
def extract_url_context(url):
    text = extract_text_from_url(url)[:8_000]

    completion = client.chat.completions.create(
        model="openai/gpt-oss-120b",
        messages=[{
            "role": "user",
            "content": [{
                "type": "text",
                "text": PARSE_PROMPT(SOURCE_EXTRACTOR_PROMPT,
                TEXT=text)
            }]
        }],
        temperature=0,
        max_completion_tokens=2048,
        top_p=1,
        stream=False,
        stop=None,
    )
    content = completion.choices[0].message.content
    return content
```

Anexo / Código 8: Script de Extracción de Información

VIII.3) Generación de Contenido General

Para los metadatos generales de la plataforma, aunque no visibles en el **frontend** de la prueba de concepto, se pobló la base de datos con períodos, técnicas pictóricas y breves biografías de los autores. Estos proporcionan información contextual esencial para la interpretación y categorización de las obras de arte que en una etapa posterior del proyecto podrían enriquecer la experiencia del usuario con una base mas robusta de contenido.
Pasa la generación de estos se usaron dos enfoques distintos. Para técnicas pictóricas y períodos artísticos, se empleó generación directa con modelos de lenguaje. En cambio, para biografías de autores se extrajo información de Wikipedia para evitar imprecisiones en datos biográficos complejos, priorizando la calidad de la fuente.

```python
TECHNIQUE_PROMPT = \
"""Explícame en detalle una técnica de pintura o dibujo.
Considera lo siguiente:
- Debe estar en un formato adecuado para narración.
- Debe ser conciso y corto en duración.
- NO digas en resumen.
- NO seas redundante.
- NO utilices títulos ni subtitulos, solo escribe en parrafos narrados.

Técnica: {TECHNIQUE}
Explicación:"""

def get_technique(technique):
    completion = client.chat.completions.create(
        #model="llama-3.1-8b-instant",
        model="llama-3.3-70b-versatile",
        messages=[{
            "role": "user",
            "content": [{
                "type": "text",
                "text": PARSE_PROMPT(TECHNIQUE_PROMPT, TECHNIQUE=technique)
            }]
        }],
        temperature=0,
        max_completion_tokens=1024,
        top_p=1,
        stream=False,
        stop=None,
    )
    content = completion.choices[0].message.content
    return content
```

Anexo / Código 9: Generador de Texto de Técnica

Como se ha detallado previamente, la generación de este contenido requiere únicamente el nombre de la técnica o estilo pictórico a describir. Además, se han implementado consideraciones específicas en cuanto a la cadencia y el estilo del texto generado, con el objetivo de optimizar su procesamiento mediante modelos de síntesis de voz (TTS).

```python
PERIOD_PROMPT = \
"""Explícame en detalle un estilo pictórico en particular.
Considera lo siguiente:
- Debe estar en un formato adecuado para narración.
- Debe ser conciso y corto en duración.
- NO digas en resumen.
- NO seas redundante.
- NO utilices títulos ni subtitulos, solo escribe en parrafos narrados.

Estilo pictórico: {PERIOD}
Explicación:"""

def get_period(period):
    completion = client.chat.completions.create(
        #model="llama-3.1-8b-instant",
        model="llama-3.3-70b-versatile",
        messages=[{
            "role": "user",
            "content": [{
                "type": "text",
                "text": PARSE_PROMPT(PERIOD_PROMPT, PERIOD=period)
            }]
        }],
        temperature=0,
        max_completion_tokens=1024,
        top_p=1,
        stream=False,
        stop=None,
    )
    return completion.choices[0].message.content
```

Anexo / Código 10: Generador de Texto de Período

En la generación de la biografía se utilizó un script de extracción de información (Anexo / Código 8) para procesar un breve texto biográfico del autor.

```python
AUTHOR_BIO_PROMPT = \
"""Eres un experto en redacción biográfica y narrativa.
Tu tarea es leer el siguiente texto y redactar una biografía breve y fluida del autor o artista mencionado, adecuada para acompañar una ficha de obra de arte.
Instrucciones:
1. Resume los aspectos esenciales de la vida y obra del autor: formación, estilo, periodo histórico, aportes y relevancia artística.
2. Mantén un tono natural, informativo y narrativo, sin usar listas ni formato enciclopédico.
3. Evita redundancias, tecnicismos innecesarios y frases genéricas.
4. No incluyas fechas o datos no presentes en el texto fuente.
5. No menciones el proceso de redacción ni expresiones como “esta biografía trata sobre”.
6. Si el texto está en otro idioma, tradúcelo al español de manera natural.
7. Extensión máxima: dos párrafos.
8. Evita completamente el uso de caracteres especiales

Texto: {TEXT}
Tu respuesta:"""

def get_bio(url):
    text = extract_url_context(url)
    completion = client.chat.completions.create(
        model="llama-3.3-70b-versatile",
        messages=[{
            "role": "user",
            "content": [{
                "type": "text",
                "text": PARSE_PROMPT(AUTHOR_BIO_PROMPT, TEXT=text)
            }]
        }],
        temperature=0,
    )
    return completion.choices[0].message.content
```

Anexo / Código 11: Author Bio Generation

VIII.4) Utilidades Generales de Procesamiento de Imágenes

A continuación, se definen utilidades de preprocesamiento general de imágenes que se aplican antes de realizar llamadas a proveedores de servicios de inferencia o de emplearlas en modelos autohospedados.

- **`resize_if_oversized`**: ajusta imágenes manteniendo proporciones, evitando que excedan un tamaño máximo para optimizar almacenamiento y rendimiento.
- **`download_img`**: descarga imágenes desde URLs usando un **user-agent** personalizado (para evitar bloqueos de conexión) y las redimensiona si superan el límite, garantizando su adaptación al sistema.
- **`image_to_base64`**: convierte imágenes PIL a base64, facilitando su transmisión y almacenamiento como texto plano para integración en APIs o bases de datos.

```python
def resize_if_oversized(image, max_dim=1024):
    width, height = image.size
    if max(width, height) <= max_dim:
        return image  # nothing to do

    # scale preserving proportions
    scale = max_dim / max(width, height)
    new_size = (int(width * scale), int(height * scale))
    return image.resize(new_size, Image.Resampling.LANCZOS)

def download_img(url, max_dim: int = 1024):
    headers = { "User-Agent": "Chrome/51.0.2704.103 Safari/537.36" }
    r = requests.get(url, headers=headers)
    im = Image.open(BytesIO(r.content))
    im = resize_if_oversized(im, max_dim)
    return im

def image_to_base64(image: Image.Image, format="PNG") -> str:
    """Convert PIL Image to base64 string."""
    buffered = BytesIO()
    image.save(buffered, format=format)
    return base64.b64encode(buffered.getvalue()).decode("utf-8")
```

Anexo / Código 12: Utilidades de Procesamiento de Imágenes

VIII.5) Generación de Audio Descriptivo

Este modulo toma como entrada una *URL* a una imagen, para luego procesarla en una descripción objetiva de los elementos representados.

Para el procesamiento de las imagenes se utilizan las utilidades definidas en Anexo / Código 12. Para la generación del texto se emplea un **prompt** estructurado que guía al modelo en la creación de descripciones del contenido representado utilizando una cadencia adecuada para su narración por voz.

```python
DESCRIPTION_PROMPT = \
"""Describe los elementos retratados en la imagen.
Considera lo siguiente:
- Debe estar en un formato adecuado para narración.
- Debe ser conciso y corto en duración.
- NO digas en resumen.
- NO seas redundante.
- NO utilices títulos ni subtitulos, solo escribe en parrafos narrados.

Elementos retratados: """

def get_description(url):
    image = download_img(url)
    b64_image = image_to_base64(image)
    completion = client.chat.completions.create(
        model="meta-llama/llama-4-scout-17b-16e-instruct",
        messages=[{
            "role": "user",
            "content": [
                { "type": "text",
                  "text": DESCRIPTION_PROMPT },
                { "type": "image_url",
                  "image_url": {
                    "url": f"{b64_image}"
                  } }
            ]
        }],
        temperature=0,
    )
    return completion.choices[0].message.content
```

Anexo / Código 13: Generación de texto para audios dscriptivos

VIII.6) Narrador de Contexto

Este modulo toma como entrada una *URL* de un articulo de Wikipedia para generar una narración didáctica sobre el contexto en que una obra se produjo.

Utiliza el módulo descrito en Anexo / Código 8 para la extracción de la información.

```python
SOURCE_SUMMARY_PROMPT = \
"""Eres un experto en redacción narrativa. Tu tarea es leer el siguiente texto y escribir un resumen breve (máximo dos párrafos) en un estilo natural, fluido y adecuado para narración oral o escrita.

Instrucciones:
1. Mantén un tono narrativo, como si estuvieras contando los hechos de manera clara y atractiva.
2. Sé conciso: evita repeticiones, rodeos o frases innecesarias.
3. No incluyas títulos, subtítulos ni listas.
4. No uses expresiones como “en resumen”, “este artículo trata sobre”, ni menciones al proceso de resumen.
5. Si el texto está en otro idioma, tradúcelo al español con naturalidad.
6. Enfócate en los hechos principales y el contexto historico de la producción de la obra.

Texto: {TEXT}
Tu respuesta:"""

def get_narration(url):
    text = extract_url_context(url)
    completion = client.chat.completions.create(
        model="llama-3.3-70b-versatile",
        messages=[{
            "role": "user",
            "content": [{
                "type": "text",
                "text": PARSE_PROMPT(SOURCE_SUMMARY_PROMPT, TEXT=text)
            }]
        }],
        temperature=0,
    )
    return completion.choices[0].message.content
```

Anexo / Código 14: Generación de texto para narraciones

VIII.7) Generación de Sonidos Ambientales

La heurística utilizada para la generación de sonidos ambientales consiste en 3 pasos:

1. **Segmentación de la imagen** (Anexo / Código 15): La obra se divide en cuadrantes mediante un algoritmo de particionamiento espacial. Cada cuadrante se utiliza posteriormente como entrada para los pasos posteriores.
2. **Extracción semántica de elementos** (Anexo / Código 16): Mediante Llama 4 Scout 17B se analizan cada cuadrante para detectar objetos, seres vivos y elementos con potencial sonoro. La salida en JSON incluye descripción y clasificación (fondo/primer plano) de los elementos detectados.
3. **Síntesis de audio ambiental** (Anexo / Código 17): Utilizando modelos generativos de audio basados en difusión (AudioLDM), se produce una composición sonora multicanal que integra los elementos detectados. Cada componente sonoro se ajusta en términos de volumen de acuerdo a si es categorizado como un objeto de fondo.

```python
def get_quadrants(image: Image, N:int=3):
    total_width, total_height = image.size
    w, h = total_width//N, total_height//N
    w_pos = [(w*i, w*(i+1)) for i in range(N)]
    h_pos = [(h*i, h*(i+1)) for i in range(N)]
    boxes = [
        (w_pos[i][0], h_pos[j][0], w_pos[i][1], h_pos[j][1])
        for j in range(N) for i in range(N)
    ]
    cropped = [ image.crop(box)
                for box in boxes ]
    coords= [
      ( (i/N + (i+1)/N)/2,
        (j/N + (j+1)/N)/2 )
      for j in range(N) for i in range(N)
    ]

    return cropped, coords
```

Anexo / Código 15: Segmentación de imagen en cuadrantes

```python
EXTRACTOR_PROMPT = \
"""Extract audio ambience generation keys from the following image.

Consider the following:
- Your output should be a list of json objects containing the keys:
    {"is_background": bool, "object": string}
- Only indicate objects with sound ambience relevance.
- Ignore abstract elements
- Indicate 3 elements at most
- Group elements when there are more than 1 depicted
- You should only output ONE list
- Alive elements should be added with their corresponding sound description (bear growling, dog barking)
- Non alive elements should be specified as the element itself
- Output just the required JSON results

JSON RESULT:"""

def get_semantic_elements(image: Image, max_attempts=3):
    b64_image = image_to_base64(image)
    for _ in range(max_attempts):
        try:
            completion = client.chat.completions.create(
                model="meta-llama/llama-4-scout-17b-16e-instruct",
                messages=[{
                "role": "user",
                "content": [
                { "type": "text",
                    "text": EXTRACTOR_PROMPT },
                { "type": "image_url",
                    "image_url": {
                    "url": f"{b64_image}" } }
                ]
                }],
                temperature=0,
            )
            content = completion.choices[0].message.content
            return json.loads(content)
        except:
            continue
    return []
```

Anexo / Código 16: Extracción semántica de elementos en la obra

```python
from diffusers import AudioLDMPipeline
from pydub import AudioSegment
import numpy as np
import tempfile
import os
import torch
VOLUME_DB_BACKGROUND = -20
VOLUME_DB_FOREGROUND = -5

pipe = AudioLDMPipeline.from_pretrained(
    "cvssp/audioldm"
    torch_dtype=torch.float16
).to("cuda")

def tensor_to_audiosegment(audio_tensor, sample_rate=16000):
    samples = (audio_tensor * 32767).astype(np.int16)
    return AudioSegment(
        samples.tobytes(),
        frame_rate=sample_rate,
        sample_width=2,  # 16-bit = 2 bytes
        channels=1
    )

def generate_ambient_sound(scene, filename):
    final_mix = AudioSegment.silent(duration=20000)
    for i, obj in enumerate(scene):
        prompt = obj['object']
        negative_prompt = ["low quality, average quality"]
        result = pipe(
            [prompt],
            negative_prompt=negative_prompt,
            num_inference_steps=30,
            audio_length_in_s=20,
            return_dict=True
        )
        audio_tensor = result.audios[0].squeeze()
        audio_segment = tensor_to_audiosegment(audio_tensor)
        volume_db = VOLUME_DB_BACKGROUND if obj["is_background"] else VOLUME_DB_FOREGROUND
        audio_segment = audio_segment + volume_db
        final_mix = final_mix.overlay(audio_segment)

    final_mix.export(f"{filename}.wav", format="wav")
```

Anexo / Código 17: Generación de sonido ambiental con AudioLDM

VIII.8) Text to Speech

Para el módulo de **Text-to-Speech** (TTS) se utilizó fué **Kokoro 82M** [10], para el cual existe una biblioteca en Python con módulos de inferencia preimplementados.

```python
# !pip install -q kokoro>=0.9.4 soundfile
# !apt-get -qq -y install espeak-ng > /dev/null 2>&1

from kokoro import KPipeline
from IPython.display import display, Audio
import soundfile as sf
import torch

tts_pipeline = KPipeline(lang_code='e')
```

Anexo / Código 18: Setup Kokoro TTS

Nuestro función de generación de TTS añade breves silencios entre parrafos para mantener una cadencia fluida en la narración.

```python
def tts(text, filename):
    generator = tts_pipeline(
        text,
        voice='em_santa',
        speed=1, split_pattern=r'\n+'
    )

    pause_duration = 0.5
    silence = np.zeros(int(24000 * pause_duration), dtype=np.float32)
    all_audio = []

    for i, (gs, ps, audio) in enumerate(generator):
        all_audio.append(audio)
        all_audio.append(silence)

    final_audio = np.concatenate(all_audio)
    sf.write(f"{filename}.wav", final_audio, 24000)
```

Anexo / Código 19: Generación de audio con Kokoro TTS