UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Máster en Ingeniería en Informática

TFM del Máster en Ingeniería Informática

GoAvatar

Presentado por Manuel C. León Rivera
en Universidad de Burgos — 23 de julio de 2025
Tutor: Dª. María Belén Vaquerizo García

AGRADECIMIENTOS

A mi familia, en especial a Verónica Rodríguez Gavilanes y a mi madre, por sus valiosos consejos y recomendaciones. Siempre acertados. Sin ellas, este TFM no habría sido el mismo.

A mi tutora Belén Vaquerizo, siempre atenta y dispuesta, su guía ha sido invaluable en este proyecto.

A Vinila y a Bowie. Los dos han sido una fuente inagotable de esperanza. En especial Vini, que nos ha dejado en la recta final, luchando como una campeona, sin perder jamás su espíritu ante la adversidad. Siempre estarás en nuestros corazones.

Y a todos aquellos que confiaron en este proyecto durante su desarrollo.

Gracias a todos.

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Máster en Ingeniería en Informática

Dª. María Belén Vaquerizo García, profesora del Área de Lenguajes y Sistemas Informáticos del Departamento de Ingeniería Civil.

Expone:

Que el alumno D. Manuel Carlos León Rivera, con DNI 76257535T, ha realizado el Trabajo final de Máster en Ingeniería Informática titulado GoAvatar.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, X de julio de 2025

Vº. Bº. del Tutor:

María Belén Vaquerizo García

Resumen

El presente Trabajo de Fin de Máster se enmarca en el ámbito de las tecnologías de inteligencia artificial aplicadas a la generación automatizada de contenido audiovisual mediante avatares digitales realistas.

Este proyecto aborda el desarrollo de una aplicación web capaz de integrar y automatizar la generación de vídeos con avatares a partir de texto, voz y parámetros visuales, todo ello mediante el uso de APIs avanzadas como D-ID, HeyGen, Synthesia o Heigen. Además, se contempla la posibilidad de personalizar el fondo, los subtítulos y la voz del avatar, así como la automatización del flujo de generación y publicación de vídeos y notificaciones a redes sociales.
El objetivo principal es analizar y comparar las capacidades de diferentes APIs de generación de avatares, seleccionar la más adecuada y construir un sistema propio que permita generar estos vídeos sin necesidad de acceder manualmente a los portales de los proveedores.

Como resultado, se obtiene una herramienta útil que puede aplicarse en ámbitos como la atención al cliente, el marketing, la educación o la comunicación institucional.

Descriptores

Avatares virtuales, generación automática de vídeo, inteligencia artificial, API, webhook, publicación automática, aplicación web, proveedores.

Abstract

This Master's Thesis is framed within the field of artificial intelligence technologies applied to the automated generation of audiovisual content through realistic digital avatars.
The project focuses on the development of a web application capable of integrating and automating the generation of avatar-based videos from text, voice, and visual parameters, using advanced APIs such as D-ID, HeyGen, Synthesia, or Anam AI. It also includes features for customizing backgrounds, subtitles, and voice, as well as automating the workflow for video generation, publication, and notifications to social media platforms.
The main goal is to analyze and compare the capabilities of different avatar generation APIs, select the most suitable one, and build a proprietary system that allows companies to generate videos independently, without relying on the providers' online platforms.
As a result, the application becomes a valuable tool that can be applied in sectors such as customer service, marketing, education, or institutional communication.

Keywords

Virtual avatars, automatic video generation, artificial intelligence, API, webhook, automatic publishing, web application, providers.

Índice General

Introducción

La inteligencia artificial lleva años ganando terreno, y ya no es solo una promesa del futuro. Hoy forma parte del día a día en muchas empresas, y poco a poco se está convirtiendo en una herramienta para mejorar procesos, ahorrar tiempo y ofrecer nuevos servicios. Entre sus múltiples aplicaciones, hay una que está empezando a destacar con fuerza: la generación automática de contenido en vídeo con avatares digitales.

Esta tecnología permite crear vídeos donde una imagen, que puede ser incluso una foto, cobra vida y habla de forma creíble, con gestos y expresiones que simulan bastante bien a una persona real. Esto ya se está viendo en sectores como la atención al cliente, la formación online o la publicidad. No se trata de sustituir a nadie, sino de ofrecer una forma más cercana de comunicar, sin necesidad de grabaciones ni producción audiovisual tradicional. Plataformas como D-ID, Synthesia o HeyGen han desarrollado APIs que permiten generar vídeos a partir de una imagen, un texto y una voz, consiguiendo que parezca que una persona real nos está hablando. Sin embargo, estas herramientas, aunque potentes, todavía están muy ligadas a sus propias plataformas web, lo que limita su integración en aplicaciones más amplias o personalizadas. Muchas empresas necesitan algo más flexible, que puedan controlar desde dentro y automatizar contenido sin depender de interfaces externas.

Este Trabajo de Fin de Máster nace con la idea de investigar a fondo qué posibilidades ofrecen actualmente estas APIs y de qué manera se pueden integrar en una aplicación web propia. Para ello, el primer paso ha sido analizar las principales opciones disponibles y, a partir de ese estudio, construir un sistema experto basado en reglas que permita elegir la API más adecuada según unos criterios definidos por el usuario: calidad del vídeo, rapidez, coste, nivel de personalización, etc.

Después de eso, el siguiente objetivo ha sido desarrollar una aplicación que permita generar vídeos automáticamente a partir de esa API seleccionada. El usuario puede configurar aspectos como el texto que dirá el avatar, la voz, el fondo, o incluso la imagen utilizada. Y no solo eso: también se ha planteado la posibilidad de

automatizar la publicación de estos vídeos en redes sociales u otros canales digitales.

Además, la aplicación incluye un módulo de conversación en tiempo real con un avatar, pensado para que un usuario pueda recibir asesoramiento o información de forma interactiva. Este apartado se basa en modelos conversacionales con IA, lo que da pie a muchas posibilidades futuras, como usarlo para soporte al cliente, asesoramiento comercial o incluso como tutor educativo.

Esta propuesta busca dar respuesta a una necesidad real: contar con una herramienta que permita generar contenido sin estar atado a la web del proveedor. Muchas plataformas ofrecen funcionalidades avanzadas, pero sólo desde su propia interfaz, lo que limita mucho a empresas que quieren integrar estos servicios dentro de sus propios sistemas. En lugar de depender del acceso manual a un panel externo, este proyecto plantea tener el control total desde una aplicación propia, que pueda adaptarse a las necesidades específicas de cada organización.

Por ejemplo, una empresa que lanza campañas periódicas en redes sociales tendría que acceder manualmente cada vez al panel del proveedor para generar el vídeo, descargarlo y subirlo a sus canales. Con esta solución, todo ese proceso se puede automatizar desde su propia plataforma, sin pasos intermedios ni limitaciones impuestas por el proveedor. Esto no solo ahorra tiempo, sino que permite escalar y adaptar el sistema a cualquier flujo de trabajo empresarial.

Otro ejemplo: un centro de formación podría utilizar este sistema para enviar vídeos explicativos personalizados a cada alumno, o una empresa para responder dudas frecuentes de sus clientes sin necesidad de que un agente lo atienda en directo. Y, lo más importante: 24 horas al día, 7 días a la semana, sin descanso.

Este proyecto no busca reinventar la rueda, sino aprovechar al máximo las herramientas que ya están disponibles y ver hasta dónde pueden llevarnos si las usamos con criterio. La tecnología que lo hace posible ya existe, pero todavía está dando sus primeros pasos en cuanto a integración y automatización real. No es difícil imaginar que en pocos años las empresas empiecen a incorporar de forma habitual un avatar digital como parte de su estrategia de comunicación, tanto para atención al cliente como para generar contenido personalizado a gran escala. El objetivo aquí es adelantarse un poco a esa tendencia y demostrar que ya es posible construir una solución con potencial real de crecimiento.

1.1. Estructura de la memoria

- **Introducción:** Presenta el contexto del proyecto, la problemática abordada y la motivación que da origen al trabajo. Además, se describe la estructura general del documento y el contenido de los anexos que lo complementan.

- **Objetivos del proyecto:** Se detallan los objetivos perseguidos con este trabajo, distinguiendo entre los de carácter general y aquellos de naturaleza técnica, necesarios para alcanzar las metas propuestas.

- **Conceptos teóricos:** Se introducen y explican los conceptos clave relacionados con la temática del proyecto, con el fin de facilitar su comprensión a lo largo de la memoria.

- **Técnicas y herramientas:** Se exponen las tecnologías, lenguajes, bibliotecas y metodologías empleadas durante el desarrollo del sistema, justificando su elección y uso.

- **Aspectos relevantes en el desarrollo:** Se describen las fases más destacadas del proceso de desarrollo, así como los retos técnicos enfrentados, las decisiones tomadas y las soluciones implementadas.

- **Trabajos relacionados:** Se analizan otros trabajos previos o tecnologías afines que guardan relación con el objetivo del proyecto, situando así este trabajo dentro de un marco comparativo y evolutivo.
- **Conclusiones y líneas de trabajo futuras:** Se presentan las conclusiones extraídas tras la realización del proyecto, junto con una reflexión sobre posibles mejoras y extensiones que podrían desarrollarse en el futuro.

1.2. Estructura de los anexos

- **Plan de proyecto software:** Planificación temporal y estudio económico. Contiene, además, el estudio del arte de los proveedores y el sistema experto basado en reglas.

- **Especificación de requisitos:** Detalle de los requisitos necesarios y establecidos para la elaboración del proyecto.

- **Especificación de diseño:** Diseño de los datos, arquitectura y flujos de los procesos.

- **Manual del programador:** El proyecto desde el punto de vista de código, instalación y continuación.

- **Manual de usuario:** Manual basado en las distintas pantallas de la aplicación y la explicación de los elementos de las mismas.

1.3. Enlaces adicionales

- **Repositorio:** https://github.com/Maeroth/TFM_AVATAR

- **YouTube:** Presentación:
  Demostración:

- **URL de la aplicación en producción**:
  https://avatar-frontend-afyu.onrender.com/

Objetivos del Proyecto

Este apartado describe los objetivos generales y objetivos técnicos del proyecto.

2.1 Objetivos generales

- Investigar y comparar las principales APIs del mercado para la generación de contenido audiovisual mediante avatares virtuales realistas.
- Realizar un sistema experto basado en reglas para elegir el mejor proveedor mediante unas entradas.
- Desarrollar una aplicación web que permita la generación de vídeos con avatares virtuales y asesoramiento en tiempo real.
- Publicar el contenido generado de la aplicación en una red social.

2.2 Objetivos técnicos

- Proyecto desarrollado en iteraciones en SPRINTS (SCRUM).
- Para el control de versiones se utiliza el repositorio GitHub.
- Utilización del Plugin Zenhub para la gestión de tareas en GitHub.
- GitHub Desktop y Git para sincronizar con el repositorio.
- Docker para la encapsulación de entornos controlados que facilite la migración a otros entornos.
- Base de datos MongoDB para guardar la información en formato JSON.
- Desarrollo de una aplicación Backend en Node para la gestión de peticiones al servidor.
- Desarrollo de una aplicación Frontend en React para las distintas interfaces del proyecto.
- Para dar visibilidad a un proveedor estando en localhost, se habilita un servidor utilizando ngrock que permite peticiones en https (sólo para pruebas).
- Para hacer responsive la aplicación se utiliza Bootstrap.
- Las aplicaciones frontend y backend se guardan en contenedores en Render.
- La base de datos está ubicada en Cloud.MongoDB.

- Para el sistema basado en reglas utiliza como base Excel para las pruebas. Una vez testeado, se migra a javascript en backend.
- Para la documentación se utiliza Microsoft Word.
- Programas externos: Visual Paradigm Online y draw.io para los diagramas.

Conceptos teóricos

A continuación, se explican los principales conceptos y tecnologías involucrados en este Trabajo de Fin de Máster, con el fin de facilitar la comprensión de los aspectos técnicos desarrollados posteriormente.

3.1 Avatar Virtual

Un avatar virtual es una representación visual, generalmente de apariencia humana, que puede simular lenguaje corporal, expresión facial y comunicación verbal. En el contexto de este proyecto, se utilizan avatares realistas generados por inteligencia artificial para simular a una persona hablando en un vídeo.

3.2 API (Application Programming Interface)

Hace referencia al uso de modelos de inteligencia artificial para crear vídeos a partir de texto, voz o imágenes. Las plataformas como D-ID o HeyGen utilizan técnicas avanzadas de deep learning para animar una imagen estática y sincronizarla con una pista de voz.

3.3 Webhook

Un webhook es un mecanismo que permite a un servidor enviar datos automáticamente a otro cuando ocurre un evento. En este proyecto, se utiliza para recibir notificaciones cuando la generación del vídeo ha finalizado.

3.4 Sistema experto basado en reglas

Un sistema experto es un tipo de software que emula el comportamiento de un experto humano para tomar decisiones o resolver problemas en un dominio específico. En este caso, el sistema experto se emplea para ayudar en la selección de la API de generación de avatares más adecuada, según los parámetros y necesidades definidos por el usuario.

Este sistema se basa en reglas de tipo **si-condición-entonces-acción**, conocidas como reglas de producción. Estas reglas permiten evaluar diferentes criterios (por ejemplo: calidad del vídeo, tiempo de generación, personalización del

avatar, coste por vídeo, etc.) y determinar cuál es la API que mejor se ajusta al escenario de uso planteado. No requiere aprendizaje automático, sino que actúa sobre una base de conocimientos predefinida.

Por ejemplo:

- **SI** el usuario necesita vídeos de respuesta rápida **Y** no requiere personalización de voz, **ENTONCES** se recomienda usar la API X.
- **SI** el vídeo debe tener alta calidad de sincronización labial **Y** el presupuesto lo permite, **ENTONCES** se recomienda la API Y.

Este tipo de sistema es útil cuando los criterios de decisión pueden expresarse claramente y no requieren procesamiento estadístico o entrenamiento sobre datos.

3.5 Frontend y Backend

Hace referencia a la capa de presentación (Frontend) y a la lógica de negocio (backend).

3.6 Feature

Una característica concreta de la aplicación, como la generación de vídeo.

3.7 Streaming

Se refiere, básicamente, para la interactividad entre un usuario y un avatar en tiempo real.

Técnicas y herramientas

En este apartado se detallan las tecnologías, entornos de desarrollo y metodologías utilizadas para la implementación del proyecto, justificando su elección según las necesidades del sistema.

4.1 Lenguajes de programación

He optado por utilizar JavaScript [1] porque se puede aplicar tanto en backend como en frontend, esto me ha servido para simplificar el trabajo.

También porque permite trabajar con buenas librerías para servicios REST y Webhooks.

Al ser casi todo servicios REST en formato JSON, javascript es excelente para este cometido, con Node.js [2] y React [3] he podido trabajar de forma rápida y eficiente.

También hay mucha documentación y es muy fácil desplegar en otros entornos con este lenguaje.

Para el Backend he optado por Javascript / Node.js. Esto ha gestionado la lógica de negocio, la comunicación con las APIs de generación de avatares (como D-ID), la gestión de webhooks y la interacción con la base de datos.

Para el frontend he utilizado Javascript con el framework React. He desarrollado las interfaces de forma modular, para facilitar la visualización y la interacción con el usuario.

4.2 Frameworks y Librerías

Para crear un servidor en backend he usado Express.js [4], que permite gestionar las rutas de forma fácil.

Para el sistema de base de datos NoSQL he utilizado MongoDB [5] y Mongoose [6] (librería de node). He preferido trabajar con NoSQL ya que nos interesa más guardar entidades que relaciones, y, de este modo, trabajar más rápido con los accesos

a BD y a los servicios REST. Con Mongoose se configura de forma fácil los parámetros de la conexión, los metadatos y el control del estado.

Para el responsive design he optado por Bootstrap [7], que es una librería CSS para interfaz web.

Para la autenticación en la web, he preferido utilizar Tokens JWT [8] (Json Web Token) que se usa para proteger rutas y controlar el acceso a la aplicación en lugar de una seguridad por sesión.

4.3 APIs externas

Para la comunicación con un proveedor he optado por la API de D-ID [9], que es el proveedor que ha salido con mejor resultado al aplicar el sistema experto basado en reglas con unas entradas descritas en los anexos. Esta API permite generar vídeos con avatares realistas a partir de texto y audio. Además, permite la generación de contenido en tiempo real, que era una de las features que nos interesaban.

Para la comunicación entre sistemas externos, pensé que sería buena idea utilizar Zapier [10] , que es una API que conecta diferentes servicios y lanza un evento cada vez que hay una orden. Por ejemplo: al crear un vídeo en D-ID, que pudiera publicar directamente en una red social. Descarté esta opción para poder utilizar la API de X [11] (antiguo Twitter) directamente en el código, así tener un control más directo de los eventos.

4.4 Herramientas de desarrollo

Se ha utilizado Visual Studio Code [12] como entorno de desarrollo principal debido a su ligereza, extensibilidad y compatibilidad con múltiples tecnologías, lo que facilita trabajar tanto en el frontend como en el backend desde un solo entorno. Para realizar pruebas y depurar las peticiones a las APIs externas, se ha empleado Postman [13], una herramienta para probar servicios REST y validar respuestas de forma fácil.

Durante el desarrollo y las pruebas con webhooks de servicios como D-ID, he optado por Ngrok [14], que permite exponer el servidor local a través de una URL pública y facilita la recepción de notificaciones externas sin necesidad de desplegar la aplicación. Esto me ha servido para las notificaciones del proveedor de D-ID cuando tenía un vídeo listo y lo notificaba a la aplicación.

La gestión de los datos se ha llevado a cabo mediante MongoDB Compass [15], una herramienta gráfica que facilita la visualización y edición de colecciones de forma fácil. Inicialmente me instalé el servidor de MongoDB en local, pero finalmente, preferí utilizar la plataforma https://account.mongodb.com/ para montar el servidor de forma remota y es la que se utiliza actualmente en el entorno de producción, de este modo me ahorré tener que cambiar cada dos por tres la configuración en el fichero .env.

Para la gestión del control de versiones y la colaboración, se ha usado Git, junto con la plataforma GitHub [16], que ha permitido mantener el código organizado y versionado. Se ha utilizado tanto la versión de escritorio como la web, además del plugin git en Visual Studio Code que lo hacía más directo.

Para realizar la lógica del avatar en tiempo real se utilizó el SDK de D-ID. En su página ofrecen una aplicación de ejemplo [17] que adapté para la configuración del avatar y la edición de la pantalla para las necesidades de este proyecto.

Finalmente, se ha incorporado Docker [18] como plataforma de contenedores para la portabilidad y la reproducibilidad del entorno de desarrollo y despliegue. Con Docker, se puede encapsular la aplicación con todas sus dependencias, asegurando que se comporte de la misma manera independientemente del sistema en el que se ejecute. Con Docker, me he ahorrado tener que configurar el servidor de producción desde cero, simplemente, desplegando el contenedor, he tenido funcionalidad desde el principio.

4.5 D-ID Studio

D-ID Studio (https://studio.d-id.com/) es la plataforma directa del proveedor D-ID donde están todas las funcionalidades e incluso algunas que no vienen en la API, desde aquí la creación de contenido es directa y la configuración avatares más intuitiva.

Durante el desarrollo del proyecto he tenido que acceder para obtener ciertas claves (TOKENS) que me han valido para que el avatar de streaming funcionara correctamente en la aplicación.

Por desgracia, actualmente, no hay un desenganche directo del proveedor y la API, hay configuraciones que dependen de su plataforma y hay que acceder a ella obligatoriamente. Quizás esto no suceda con otros proveedores, pero con D-ID es así.

4.6 Metodología de desarrollo

Al ser un desarrollo de forma iterativo, he decidido usar SCRUM [19] desde GitHub a través de un plugin llamado Zenhub [20] que permite organizar las tareas en los sprints (milestones) y configurarlas según su estado. Esto me ha permitido llevar un seguimiento en el tiempo para valorar las siguientes tareas a abordar.

Aspectos relevantes del desarrollo del proyecto

Este apartado describe los hitos o las partes que más peso han tenido en el proyecto, como el uso de una tecnología específica o los cambios sobre la marcha.

5.1 El origen del proyecto

La idea de este proyecto surgió a partir de la solicitud de una empresa que deseaba disponer de una aplicación desde la cual pudieran generar contenido audiovisual con personas hablando de forma automática, sin necesidad de contratar actores ni participar ellos mismos en las grabaciones. Esta solución, además de agilizar el proceso de creación de contenidos, representaría un ahorro significativo en costes de producción.

Al estudiar su propuesta, se identificó que varios proveedores actuales de generación de contenido audiovisual mediante inteligencia artificial ofrecen APIs que permiten integrar estas funcionalidades en aplicaciones externas, otorgando a las empresas mayor control sobre el contenido generado. Esta capacidad abre un abanico de posibilidades para automatizar la creación y distribución de vídeos personalizados de manera rápida y eficiente.

A partir de esta premisa surgió el interés por profundizar en este campo en expansión. Cabe destacar que la generación de contenido con IA aún se encuentra en una fase de evolución, y muchas de las APIs disponibles actualmente pueden considerarse tecnologías emergentes o en fase de prototipo.

Uno de los hitos más recientes en este ámbito ha sido la incorporación de streaming en tiempo real con avatares generados por IA, una funcionalidad que no estaba disponible en años anteriores (2023–2024). Aprovechando el TFM, he visto posible indagar más en este campo para ver si tecnológicamente es viable actualmente o aún queda camino por recorrer.

5.2 El proceso iterativo

Para organizar el desarrollo de este proyecto se ha seguido un enfoque basado en

la metodología ágil SCRUM, que permite trabajar por fases e ir añadiendo funcionalidades poco a poco, en lugar de hacerlo todo de una vez.

Aunque este proyecto se ha desarrollado de forma individual, se ha intentado seguir el espíritu de SCRUM dividiendo el trabajo en bloques o etapas. Cada etapa ha tenido unos objetivos concretos, como por ejemplo crear el sistema de login, integrar la API de D-ID o diseñar la interfaz de usuario. Al finalizar cada una de ellas, se revisaban los resultados y se decidía cómo continuar.

Para llevar un control visual de las tareas y organizar el trabajo, se ha utilizado ZenHub, una herramienta que se integra con GitHub y permite ver en qué estado está cada parte del proyecto (pendiente, en progreso o terminada). Esto ha sido muy útil para mantener el orden y no perder de vista los pasos que quedaban por hacer.

5.3 Durante el estudio del arte

Inicialmente, la idea era utilizar un enfoque basado en razonamiento por casos (*Case-Based Reasoning*, CBR) [21], que consistía en ir probando cada API de los distintos proveedores e ir construyendo una base de conocimiento con los resultados obtenidos. Esta base permitiría consultar decisiones anteriores en contextos similares, generando una especie de historial útil para recomendar soluciones.

Sin embargo, durante la fase de exploración surgieron varias limitaciones. Muchos de los proveedores no permiten el acceso a sus APIs sin una suscripción previa, lo que dificultaba seriamente la experimentación libre y la recopilación de ejemplos prácticos.

Como consecuencia, el enfoque tuvo que modificarse a mitad del estudio (15 de mayo), adoptando un sistema experto basado en reglas [22]. En este nuevo modelo, se definen reglas lógicas y se asignan pesos a diferentes características técnicas (por ejemplo, calidad del vídeo, coste, tiempo de respuesta, personalización, etc.). El sistema utiliza esas reglas y ponderaciones para determinar qué proveedor es el más adecuado según las necesidades del usuario. De este modo, se obtiene una valoración más objetiva y adaptable del rendimiento de cada API.

Con esto, se pudo valorar el mejor proveedor para nuestro proyecto: D-ID (Plan Launch), con un coste mensual de 30.71€ y que permite comunicación en streaming en tiempo real.

5.4 Durante el desarrollo

Algunas herramientas previstas al inicio fueron descartadas. Por ejemplo, Zotero [23], que se planteó como apoyo para gestionar referencias, acabó no siendo necesario. También se valoró el uso de Zapier para publicar automáticamente los vídeos en redes sociales, pero se vio que no ofrecía una integración completa con la API de D-ID, y que era más eficiente encargarse directamente de esa funcionalidad desde nuestra aplicación.

Para el desarrollo del frontend se eligió React porque facilita la creación de pantallas modulares y reactivas. A diferencia de métodos más antiguos, no hace falta manipular directamente los elementos de la página cada vez que algo cambia; el propio sistema se encarga de mostrar lo que toca en cada momento. En cuanto al backend, se optó por Node.js, que permite montar servidores rápidos con poco esfuerzo. Sin embargo, a medida que el proyecto fue creciendo, especialmente al incluir la lógica de selección de proveedor, hubo que dedicar más tiempo al backend del que inicialmente se pensaba para probar diferentes casos de la API.

En un momento del proyecto se valoró crear una pantalla para gestionar configuraciones como el token de la API o la ruta de descarga de los vídeos. Finalmente se descartó por motivos de seguridad, ya que exponer esta información desde el navegador podía ser peligroso. Se optó por mantener estos datos en un archivo .env, accesible solo desde el backend.

Durante las pruebas se utilizó Ngrok para poder exponer el servidor local a internet y recibir las notificaciones de la API de D-ID, lo que me permitió hacer pruebas sin necesidad de desplegar la aplicación en un servidor real.

En cuanto a la generación de voz, se decidió usar las voces de Microsoft, ya que ofrecían un estándar más estable que las de otros proveedores, además de simplificar el proceso de elección de voces.

También surgieron distintos problemas técnicos:

- **Conversión de audio:** El navegador (frontend) graba el audio en formato .webm, pero la API de D-ID sólo acepta .mp3. Se resolvió añadiendo una conversión automática en el backend.

- **Subtítulos:** Al probar la generación de subtítulos con D-ID, se detectó un

problema de “no tienes permisos”. Buscando información, supe que esto le pasa a más gente y parece un problema con el plan contratado y que aún está pendiente de resolver el proveedor [24].

- **Respuestas del agente (LLM):** De forma similar, cuando se probó la conversación en tiempo real con el avatar, las respuestas también venían en inglés si no se especificaba el idioma en cada mensaje. Actualmente no hay un campo directo para definir el idioma por defecto, por lo que esta configuración también debe gestionarse manualmente. En mi caso, tengo que poner que detecte el idioma desde la etiqueta de voces elegidas. Por ejemplo: “Alejandro, Spanish (Spain)”, tengo que obtener Spanish (Spain) para ponerlo en el mensaje enviado al LLM y que lo detecte de esta manera.

- **Avatares personales Premium no disponibles:** Aunque en la documentación de D-ID se habla de avatares personales Premium, en el plan usado para este proyecto esa opción no estaba disponible, sólo Express. Además, la clasificación entre avatares Premium y Express no está clara en la API. Intenté crear un avatar personal Express pero la API no lo permite, así que tuve que prescindir de esta feature.

- **Avatares con fondos predefinidos:** Durante el mes de Mayo de 2025, añadieron a la lista de Avatares seleccionables un conjunto de Avatares con fondo predefinido. Éstos dan problemas a la hora de generar el vídeo. Actualmente, la aplicación funciona correctamente al elegir avatares con sin fondo predefinido.

Trabajos relacionados

En este apartado, se informa sobre aquellas aplicaciones que están basadas en APIs de generación de vídeo como en nuestro caso.

6.1. Convo.ai [25]

Esta aplicación, orientada principalmente al entorno móvil, permite mantener conversaciones en tiempo real con un avatar digital que responde mediante voz generada por inteligencia artificial, acompañada de animación facial sincronizada.

Convo.ai se apoya en la API de D-ID [26] en modo "live streaming" para la generación del vídeo en tiempo real, combinando esta funcionalidad con voces sintéticas proporcionadas por ElevenLabs [27]. La aplicación permite al usuario diseñar su propio avatar, seleccionar su voz, e incluso configurar elementos como su personalidad y apariencia. Además, incluye una función de memoria que le permite recordar información de la conversación anterior.

Desde el punto de vista técnico, se emplean tecnologías como WebSockets para mantener la conexión continua entre el cliente y el servidor, necesaria para la baja latencia en las respuestas. Aunque no se detalla públicamente toda su arquitectura, es razonable suponer el uso de entornos backend como Node.js.

Convo.ai tienen un enfoque claramente dirigido a la interacción directa y personal con el avatar, sin cubrir aspectos como la generación de vídeos bajo demanda, la publicación en redes sociales o la evaluación de múltiples proveedores para adaptarse a diferentes contextos de uso.

6.2. Defined.ai [28]

Uno de los casos más relevantes en el uso institucional de avatares con inteligencia artificial ha sido el impulsado por la empresa Defined.ai en el marco del consorcio Accelerat.ai, una iniciativa respaldada por el Gobierno de Portugal [29]. Como parte de este proyecto, y en colaboración con la Agencia para la Modernización Administrativa (AMA), se implementó un asistente virtual con avatar para mejorar la atención ciudadana a través del portal público ePortugal.

Este asistente virtual está pensado para resolver preguntas frecuentes sobre trámites digitales, como la activación de credenciales electrónicas o el acceso a servicios digitales. El sistema responde mediante voz generada por IA y con un avatar que simula gestos faciales en tiempo real, haciendo más accesible la interacción para ciudadanos menos familiarizados con entornos digitales.

Desde el punto de vista técnico, la solución integra la API de D-ID [30] para la animación del avatar y utiliza tecnología de IA conversacional a través de Azure OpenAI, proporcionada por Microsoft. Todo esto se coordina desde una arquitectura backend que gestiona las sesiones y respuestas. La interfaz ha sido diseñada con especial atención a la accesibilidad.

En términos de resultados, el asistente ya había superado las 24.000 interacciones a mediados de 2023, y se ha consolidado como una herramienta útil para descongestionar canales tradicionales de atención, especialmente en contextos de alta demanda.

Aunque se trata de un ejemplo avanzado de uso institucional, el proyecto está centrado exclusivamente en un único proveedor y orientado a la atención conversacional en tiempo real. No contempla generación de vídeos bajo demanda, selección entre distintos proveedores ni publicación en redes sociales.

6.3. SingIt [31]

En el ámbito educativo, me pareció interesante el caso de SingIt, una startup que ha creado una aplicación pensada para ayudar a niños a aprender inglés a través de canciones y la interacción con un avatar animado. La propuesta mezcla entretenimiento con aprendizaje, algo que siempre funciona bien con los niños, especialmente si se presenta de una forma visual.

La aplicación utiliza la API de D-ID para animar un personaje virtual que hace de “profesor”. Este avatar canta, explica e invita al alumno a participar. La interacción es en tiempo real, con voz sintética y expresiones faciales sincronizadas. Además, incluye algún tipo de detección de participación para dar respuestas en función de la reacción del niño.

Según comentan los responsables del proyecto, durante las pruebas, más del 85 % de los estudiantes mostraron preferencia por aprender con el avatar frente a métodos

más tradicionales.

Desde el punto de vista técnico, no hay demasiados detalles disponibles sobre su infraestructura, pero parece que se apoyan completamente en D-ID para la animación y en un backend que organiza los contenidos y gestiona las respuestas. Es una solución simple, centrada en un único proveedor, con un propósito muy concreto.

No incluye aspectos como la comparación entre proveedores, generación de vídeos a demanda o automatización de envíos, pero sí es un buen ejemplo de cómo aplicar esta tecnología en un contexto lúdico y educativo, consiguiendo buenos resultados con un enfoque sencillo.

Conclusiones y líneas de trabajo futuras

7.1 Conclusiones

Elegí este trabajo como un reto, sobre todo por el cambio de mentalidad que suponía pasar de trabajar con Java. He trabajado mucho con JavaScript, pero no tanto con Node.js. Quería comprobar hasta qué punto era práctico y rápido trabajar con este lenguaje, sobre todo en el desarrollo de servicios REST. Ahora tengo claro que es una tecnología muy eficiente, fácil de poner en marcha y con muchas posibilidades.

Por otro lado, este proyecto me ha servido para meterme de lleno en un mundo que ya está marcando el presente: la inteligencia artificial. Estamos en un momento en el que todo gira en torno a ella. Muchas empresas están lanzando sus propias APIs y soluciones, y eso abre la puerta a nuevas formas de hacer las cosas. En mi caso, cuando una empresa me propuso una idea tan particular como crear avatares que hablasen por ellos en sus campañas publicitarias, me pareció algo tan original que decidí investigar más a fondo. La idea me sorprendió y al mismo tiempo me hizo ver que esto no era ciencia ficción: era algo real, con un potencial enorme.

Recuerdo perfectamente la primera vez que generé un vídeo con un avatar hablando lo que yo había escrito. Ver cómo el personaje lo decía de forma fluida, con mi voz, fue bastante curioso. No era perfecto, pero el resultado ya era más que convincente. A partir de ahí, fue muy entretenido probar con diferentes voces, fondos y estilos, e ir viendo cómo cada pequeño cambio daba lugar a un resultado distinto.

Eso sí, también me encontré con algunas limitaciones importantes. Muchas de estas APIs aún están a medio camino: hay funciones que aparecen en las plataformas web pero que no se pueden hacer desde la API. Por ejemplo, para crear un avatar para streaming, tuve que hacerlo manualmente desde el panel de D-ID, porque la API no lo permite. Esto te obliga a hacer parte del trabajo “a mano”, lo que a veces retrasa el desarrollo y te hace dar algún paso atrás.

Aun así, ha sido un proceso muy interesante. Ir probando cada funcionalidad, ajustar parámetros y comprobar cómo respondía el sistema ha sido, sinceramente, la

parte más divertida del proyecto. Esta tecnología puede tener muchos usos: desde la educación y la atención al cliente, hasta la hostelería o el contenido para redes sociales. Ya no hablamos sólo de una voz generada por ordenador: ahora estamos hablando con un personaje que, aunque no es real, se siente cercano.

Si tuviera que volver a empezar este proyecto, seguiría apostando por Node.js y React sin pensármelo. Node me ha permitido trabajar de forma rápida, sin las complicaciones que suelen aparecer con otros lenguajes. La integración con MongoDB, por ejemplo, ha sido muy sencilla. No he tenido que preocuparme por cursores, transacciones o configuraciones complejas. Y en cuanto a React, me ha gustado mucho cómo se estructura todo, con componentes reutilizables y librerías fáciles de instalar. En general, he podido trabajar de forma fluida, sin tener que pelearme con errores extraños ni con dependencias problemáticas.

Este trabajo también me ha permitido mejorar profesionalmente. No sólo por el conocimiento técnico adquirido, sino porque ahora me siento más preparado para afrontar proyectos reales que incluyan inteligencia artificial y aplicaciones distribuidas. Haber trabajado con tecnologías actuales como Node.js, React o MongoDB, junto con la integración de servicios externos mediante APIs, ha sido un desafío superado, conocimiento adquirido al fin y al cabo.

Además, la utilidad práctica de la aplicación desarrollada es evidente. No se trata sólo de una prueba de concepto, sino de una herramienta con múltiples posibilidades en ámbitos como el marketing digital, la formación online o la atención automatizada. Por ejemplo, una pequeña academia de idiomas podría utilizar esta tecnología para crear vídeos personalizados con un avatar que explique conceptos gramaticales o corrija errores comunes, adaptando el contenido según el nivel de cada estudiante. La importancia es mayor en cursos a distancia. Poder generar vídeos personalizados o contenido en tiempo real a partir de parámetros definidos por el usuario y con posibilidad de publicarlos automáticamente es algo que, a día de hoy, puede marcar la diferencia en muchos sectores.

En definitiva, ha sido un proyecto con mucho aprendizaje, en el que he descubierto herramientas nuevas y he podido trabajar con tecnologías que claramente van a ser muy importantes en los próximos años.

7.2 Líneas de trabajo futuras

Como he dicho, aún queda mucho por hacer, las APIs deben seguir evolucionando. Como he probado la API de D-ID hablaré sobre esta. Aquí dejo algunas de las características que he echado en falta para poder mejorar la aplicación.

- La creación de un avatar Express personal desde la API: Actualmente, la documentación dice que se pueden crear avatares Premium, pero con el plan que he contratado no es cierto, sólo permite Avatares Express y estos no están contemplado en la API.

- En la aplicación, poner un indicador en el menú de creación de vídeos donde nos informara de los créditos que tenemos para consumir en vídeos. Esto la API no lo ofrece actualmente y tenemos que ver los créditos que nos quedan en la web de D-ID.

- Poder realizar videoconferencias con una o más personas, donde el interlocutor pueda ir escribiendo o hablando por el micro y el avatar vaya reproduciendo la información. Esto sería interesante para poder realizar Webinars. Esto, actualmente es viable con la API actual.

- Poder generar vídeos subtitulados. Actualmente, con el plan actual (Launch) no tengo permisos, mientras que la API indica que puedo generar vídeos con subtítulos. Existe un hilo abierto con este problema en D-ID [32].

- Generación de vídeos personalizados. Es decir, que cada vídeo sea el mismo pero, añadiendo parámetros, se pueda configurar a quién va dirigido. Por ejemplo, esto sería un texto para la generación de vídeos personalizados: “Hola, {nombre_cliente}, soy su asesor personal. Quería hablarle del paquete {producto_01} que puede ser de su interés…”.

- Envío por correo. Aprovechando el punto anterior, se podría enviar el vídeo personalizado al correo del interesado.

- Publicación en más redes sociales. En este proyecto se ha abordado únicamente la plataforma X (Twitter).

- Mayor definición de parámetros para el LLM del avatar. Actualmente es muy básico y no busca información en internet. Tampoco razona demasiado, es

posible que estén usando un modelo de OPENAI antiguo para la generación de contenido.

- Mayor personalización de avatares. Actualmente sólo se puede elegir la voz y la apariencia en la aplicación. La documentación de D-ID es bastante escueta en este sentido y no explica en detalle qué emociones puede expresar el avatar. Del mismo modo, sería interesante poder elegir la ropa o la indumentaria del avatar.

- Clonación de voz: hacer un menú nuevo para generar una voz sintética a partir de la voz propia y que se guarde en la base de datos del proveedor para poder usarla en la generación de vídeos.

- Automatización de vídeos en distintos idiomas. Habría que estudiarlo ya que al generar un vídeo, si se selecciona una voz española y el texto está en inglés, tendrá un acento español. Lo interesante sería generar el vídeo en distintos idiomas y que el sistema detectara qué voz utilizar en cada caso.

Las posibilidades son muchas, y la evolución de las APIs es constante. Así que lo que se pueda hacer ahora quedará obsoleto en poco tiempo o se hará de forma más sencilla en un futuro. No obstante, las posibilidades están ahí y la oportunidad también.

Referencias Bibliográficas

1. *JavaScript. (s. f.). JavaScript | MDN Web Docs.* https://developer.mozilla.org/en-US/docs/Web/JavaScript

2. Node.js. (s. f.). *Node.js*. https://nodejs.org

3. React. (s. f.). *React – A JavaScript library for building user interfaces*. https://reactjs.org

4. Express.js. (s. f.). *Express - Node.js web application framework*. https://expressjs.com

5. MongoDB. (s. f.). *MongoDB Documentation*. https://www.mongodb.com/docs

6. Mongoose. (s. f.). *Mongoose ODM*. https://mongoosejs.com

7. Bootstrap. (s. f.). Bootstrap. https://getbootstrap.com

8. JWT.io. (s. f.). *JSON Web Token*. https://jwt.io

9. D-ID. (s. f.). *D-ID Developer Docs*. https://docs.d-id.com

10. Zapier. (s. f.). *Zapier – Automation Made Easy*. https://zapier.com

11. Twitter Developer Platform. (s. f.). *Twitter API Documentation*. https://developer.twitter.com/en/docs

12. Visual Studio Code. (s. f.). *Visual Studio Code*. https://code.visualstudio.com

13. Postman. (s. f.). *Postman API Platform*. https://www.postman.com

14. Ngrok. (s. f.). *Ngrok*. https://ngrok.com

15. MongoDB Compass. (s. f.). *MongoDB Compass*. https://www.mongodb.com/products/compass

16. GitHub. (s. f.). *GitHub*. https://github.com

17. Docker. (s. f.). *Docker*. https://www.docker.com

18. Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game*. https://scrumguides.org

19. ZenHub. (s. f.). *ZenHub – Project Management for GitHub*. https://www.zenhub.com

20. GitHub (2024). *D-ID: Explore our demo repository on GitHub to see the Agents SDK in action!* https://github.com/de-id/Agents-SDK-Demo

21. Aamodt, A., & Plaza, E. (1994). *Case-based reasoning: Foundational issues, methodological variations, and system approaches*. AI Communications, 7(1), 39–59. https://www.iiia.csic.es/~enric/papers/AICom.pdf

22. Giarratano, J., & Riley, G. (2004). *Expert Systems: Principles and Programming* (4th ed.). Course Technology.

23. Zotero. (s. f.). *Your personal research assistant*. https://www.zotero.org

24. D-ID. (2025, mayo 25). *Issue: Subtitles language mismatch in API responses* [Discusión]. D-ID Community. https://docs.d-id.com/discuss/671f3725c3d7c70050ea4de3

25. Convo.ai. (2025). Convo: Talk to digital people. https://www.convo.ai

26. Convo.ai – D-ID (s.f.) https://www.d-id.com/resources/case-study/convoai/

27. ElevenLabs. (s. f.). Text to Speech. https://www.elevenlabs.io

28. Defined.ai. (2023). AI-powered avatars for government services. https://www.defined.ai

29. Defined.ai – Gobierno de Portugal (2023) https://defined.ai/press-room/defined-ai-to-play-significant-role-in-accelerat-ai-the-government-of-portugal-approved-ai-initiative

30. Defined.ai – D-ID (2023) https://www.d-id.com/resources/case-study/definedai/

31. SingIt – D-ID (s. f.) https://www.d-id.com/resources/case-study/singit/

32. D-ID (2025) Problema para generar subtítulos. https://docs.d-id.com/discuss/671f3725c3d7c70050ea4de3

REFERENCIAS BIBLIOGRÁFICAS