



UNIVERSITAT DE
BARCELONA

Trabajo final de grado

GRADO EN INGENIERÍA
INFORMÁTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Plataforma web de diálogo asistido
por IA en el contexto del arteterapia

Autor: Tomás Aladjem Ramallo

Director: Dra. Maite López Sánchez
Realitzat a: Departament de
Matemàtiques i Informàtica

Barcelona, 15 de enero de 2026

Abstract

This Final Degree Project presents MINDSCAPE, an interactive web application, which is inspired on the principles of art therapy, which explores new forms of human-machine interaction through the integration of multimodal artificial intelligence and immersive visual design. The platform enables users to initiate a dialogue with images that serve as prompts to engage in conversation. The project implements a conversational system capable of analyzing visual content, as well as, generating contextualized responses which place artwork in the central axis of the self-exploration process.

The project has been developed as a Minimum Viable Product (MVP) focused on functional and expressive experimentation, in which, the image serves as a semantic catalyst for dialogue. The solution integrates natural language processing and image analysis techniques, paired up with a web interface designed to prioritize conversational content as the core element of the user experience.

The system architecture consists of a modern, responsive frontend implemented using React technologies, and a backend responsible for managing interactions with the artificial intelligence model. From an ethical perspective, the platform is envisioned as a foundation for the development of future applications that promote responsible, transparent, and respectful uses of artificial intelligence in contexts related to emotional well-being and self-exploration. The platform is conceived as a tool to support personal reflection and subjective expression, without seeking to replace the intervention of mental health professionals.

Resumen

Este Trabajo de Fin de Grado presenta MINDSCAPE, una aplicación web interactiva fundamentada en los principios de la arteterapia, que explora nuevas formas de interacción humano-máquina mediante la integración de inteligencia artificial multimodal y un diseño visual inmersivo. La plataforma permite a la persona usuaria iniciar un diálogo a partir de imágenes que actúan como prompts para iniciar la interacción. El proyecto utiliza un sistema conversacional capaz de analizar contenido visual y generar respuestas contextualizadas que sitúan la obra artística como eje central del proceso de autoexploración.

El proyecto se ha desarrollado como un Producto Mínimo Viable (MVP) orientado a la experimentación tanto funcional como expresiva, en el que la imagen actúa como un catalizador semántico del diálogo. La solución integra técnicas de procesamiento del lenguaje natural y análisis de imágenes, junto con una interfaz web diseñada para priorizar el contenido conversacional como elemento principal de la experiencia de usuario.

La arquitectura del sistema se compone de un frontend moderno y responsivo, implementado con tecnologías React, y un backend encargado de gestionar las interacciones con el modelo de inteligencia artificial. Desde una perspectiva ética, la plataforma se concibe como una base para el desarrollo de futuras aplicaciones que promuevan un uso responsable, transparente y respetuoso de la inteligencia artificial en contextos relacionados con el bienestar emocional y la autoexploración. Asimismo, MINDSCAPE se plantea como una herramienta de apoyo a la reflexión personal y la expresión subjetiva, sin pretender sustituir en ningún caso la intervención de profesionales de la salud mental.

Resum

Aquest Treball de Fi de Grau presenta MINDSCAPE, una aplicació web interactiva fonamentada en els principis de l'artteràpia, que explora noves formes d'interacció humà-màquina mitjançant la integració d'intel·ligència artificial multimodal i un disseny visual immersiu. La plataforma permet als usuaris iniciar un diàleg a partir d'imatges que actuen com a prompts per inicialitzar la interacció. El projecte utilitza un sistema conversacional capaç d'analitzar contingut visual i generar respostes contextualitzades que situen l'obra artística com a eix central del procés d'autoexploració.

El projecte s'ha desenvolupat com un Producte Mínim Viable (MVP) orientat a l'experimentació tant funcional com expressiva, en el qual la imatge actua com un catalitzador semàntic del diàleg. La solució integra tècniques de processament del llenguatge natural i anàlisi d'imatges, juntament amb una interfície web dissenyada per prioritzar el contingut conversacional com a element principal de l'experiència d'usuari.

L'arquitectura del sistema es compon d'un frontend modern i responsiu, implementat amb tecnologies React, i d'un backend encarregat de gestionar les interaccions amb el model d'intel·ligència artificial. Des d'una perspectiva ètica, la plataforma es concep com una base per al desenvolupament de futures aplicacions que promoguin un ús responsable, transparent i respectuós de la intel·ligència artificial en contextos relacionats amb el benestar emocional i l'autoexploració. Així mateix, MINDSCAPE es planteja com una eina de suport a la reflexió personal i a l'expressió subjectiva, sense pretendre substituir en cap cas la intervenció de professionals de la salut mental.

Agradecimientos

En primer lugar, quiero expresar mi agradecimiento a mi tutora, Maite López, por su apoyo constante, su orientación y su disponibilidad a lo largo de todo el desarrollo de este proyecto. Su acompañamiento y sus aportaciones han sido fundamentales para poder llevar a cabo este trabajo.

Asimismo, deseo agradecer a Pilar Rosado, profesora de la Facultad de Bellas Artes, por su ayuda en la definición de la visión artística del proyecto y por aportar una perspectiva complementaria que ha enriquecido significativamente el enfoque y el resultado final.

Por último, quiero agradecer a mi familia y amigos por el apoyo recibido durante este proceso, así como por su ánimo y comprensión en los momentos en los que más lo he necesitado.

Declaración de autoría

Por la presente declaro que algunas partes de este manuscrito se han redactado con el apoyo puntual de herramientas de inteligencia artificial generativa (ChatGPT), utilizadas exclusivamente para mejorar la claridad del lenguaje y sugerir redacciones alternativas. Asimismo, ChatGPT se ha empleado de forma ocasional como herramienta de apoyo durante el desarrollo del código y la documentación.

Todo el contenido ha sido revisado críticamente y validado por el autor, quien asume plena responsabilidad de la versión final del Trabajo Fin de Grado entregado, tanto del manuscrito como del código. En ningún caso se ha utilizado ningún sistema de IA generativa para generar las ideas, los análisis ni los resultados del presente trabajo.

Índice

1. Introducción	1
1.1. Contexto	1
1.2. Estado del arte	2
1.2.1. Evaluación de Soluciones Existentes y Diferenciación	2
1.3. Motivación	3
1.4. Objetivos	4
1.4.1. Objetivos generales	4
1.4.2. Objetivos específicos	4
1.5. Estructura de la memoria	6
2. Planificación del proyecto y costes	7
2.1. Planificación	7
2.1.1. Planificación inicial	7
2.1.2. Planificación final	8
2.1.3. Diagrama de Gantt	9
2.2. Costes	9
2.2.1. Costes medioambientales	10
3. Análisis tecnológico	11
3.1. Métodos para crear una web	11
3.2. Posibles tecnologías	12
3.2.1. Frontend	12
3.2.2. Backend	13
3.2.3. Modelos de inteligencia artificial	14
3.3. Tecnologías escogidas	14
3.3.1. Frontend	14
3.3.2. Backend	15
3.3.3. Modelos de inteligencia artificial	16
4. Estructura del proyecto	17
4.1. Estructura general del proyecto	17
4.2. Estructura del backend	18
4.2.1. Ejemplo de Flujo de trabajo del backend	19
4.3. Estructura del frontend	19

5. Implementación	22
5.1. Implementación del backend	22
5.1.1. main.py	22
5.1.2. controllers.py	23
5.1.3. models.py	23
5.2. Implementación del frontend	24
5.2.1. App.jsx	24
5.2.2. api.js	25
5.2.3. ImageSelectorContainer y ImageSelectorUI	26
5.2.4. ImageChatContainer y ImageChatUI	27
6. Funcionamiento de los Endpoints y del Modelo	29
6.1. Inicio de la Conversación	29
6.2. Continuación de la Conversación	30
6.3. Modelo de IA Gemini 2.5 Flash	31
7. Pruebas Realizadas	33
7.1. Tests de usuario	33
7.2. Resultados de las pruebas realizadas	35
8. Conclusiones y mejoras a futuro	38
8.1. Conclusiones	38
8.2. Mejoras a futuro	40
A. Manual de ejecución y uso del sistema	44
B. Preguntas realizadas en las pruebas con usuarios	49

1. Introducción

1.1. Contexto

En los últimos años, la inteligencia artificial ha avanzado de manera significativa, especialmente en áreas como el procesamiento del lenguaje natural, el aprendizaje profundo o los modelos capaces de integrar múltiples modalidades de información, como texto, voz o señales sensoriales. Estos avances han permitido que los agentes conversacionales basados en IA se utilicen en contextos cada vez más diversos, incluyendo la atención a la salud mental y el bienestar emocional. Estudios recientes señalan que “these CAs can effectively alleviate psychological distress, with the most pronounced effects seen in studies employing generative AI, using multimodal or voice-based CAs, or delivering interventions via mobile applications and instant messaging platforms” [1], evidenciando su potencial para reducir síntomas de ansiedad en distintos grupos poblacionales mediante el uso de agentes conversacionales. Además, la creciente accesibilidad de estas tecnologías ha facilitado que más personas puedan beneficiarse de herramientas basadas en IA, contribuyendo a una democratización del acceso a recursos digitales.

Paralelamente, la salud mental se ha consolidado como una de las principales preocupaciones sociales a nivel global. Organismos internacionales como la World Health Organization (WHO) [2] han alertado sobre el aumento de los problemas relacionados con el bienestar emocional y la necesidad de ampliar el acceso a recursos de apoyo. Factores como el ritmo acelerado de vida, el aislamiento social y la incertidumbre contemporánea han contribuido a este escenario, mientras que el acceso a profesionales de la salud mental continúa siendo limitado para una parte significativa de la población. En este contexto, han emergido herramientas digitales orientadas a facilitar espacios de reflexión y expresión personal que, sin sustituir la intervención clínica, buscan complementar los recursos tradicionales desde un enfoque preventivo y de acompañamiento.

Dentro de este marco, la arteterapia ocupa un lugar relevante como disciplina terapéutica que utiliza el arte y la imagen como medio de comunicación emocional. A diferencia de enfoques exclusivamente verbales, la arteterapia proporciona un espacio no verbal que facilita la expresión y externalización de experiencias internas complejas, promoviendo la regulación emocional y la construcción de significado a través de procesos creativos. La literatura científica revisada confronta múltiples estudios que muestran que la arteterapia puede mejorar diversos indicadores de salud mental desde la reducción de síntomas de ansiedad y depresión hasta una mayor conciencia emocional y habilidades de comunicación interpersonal. Cuando se integra como tratamiento complementario en contextos clínicos y de promoción de la salud. Estos principios han comenzado a trasladarse progresivamente a entornos digitales, explorando como herramientas tecnológicas pueden ampliar los medios expresivos disponibles sin perder de vista la centralidad del proceso creativo en la exploración subjetiva [3].

La convergencia entre inteligencia artificial, bienestar emocional y arteterapia plantea tanto oportunidades como desafíos. Por un lado, la IA ofrece nuevas po-

sibilidades para acompañar procesos reflexivos de forma personalizada y accesible mientras que, por otro lado, introduce cuestiones éticas relacionadas con la interpretación de contenidos sensibles, la privacidad de los datos y los límites del uso de sistemas automatizados en contextos emocionales. La literatura especializada coincide en la necesidad de adoptar enfoques responsables que prioricen la dimensión humana, sitúen la experiencia subjetiva en el centro y conciban la tecnología como una herramienta de apoyo complementaria, en coherencia con las recomendaciones éticas promovidas por organismos internacionales y la investigación académica reciente.

1.2. Estado del arte

En el ámbito del bienestar emocional asistido por inteligencia artificial, se han consolidado diversas aplicaciones basadas en agentes conversacionales. Plataformas como **Woebot**[19] y **Wysa** [20] utilizan modelos de lenguaje para ofrecer apoyo emocional estructurado, principalmente apoyado en técnicas de la terapia cognitivo-conductual. Estas soluciones están orientadas a la autoayuda guiada mediante diálogo textual, con un enfoque pragmático y estandarizado de la interacción.

Por otro lado, aplicaciones como **Replika**[21] exploran la dimensión relacional de los sistemas conversacionales, priorizando la compañía emocional y la expresión personal a través de conversaciones abiertas. No obstante, su diseño se centra casi exclusivamente en el texto y en la continuidad del diálogo, sin integrar estímulos visuales o artísticos como parte fundamental del proceso reflexivo.

En paralelo, el desarrollo de la inteligencia artificial multimodal ha dado lugar a herramientas capaces de analizar y generar contenido visual. Aunque estos sistemas se aplican ampliamente en contextos creativos y artísticos, su uso suele estar orientado a la producción o interpretación técnica de imágenes, y no a procesos de autoexploración emocional mediados por diálogo.

Finalmente, en el ámbito de la arteterapia digital existen plataformas que permiten la creación artística en entornos virtuales como medio de expresión emocional. Sin embargo, estas herramientas suelen carecer de un componente de inteligencia artificial conversacional que acompañe activamente la interpretación de la obra, manteniendo la reflexión en un plano manual o dependiente de la intervención humana.

1.2.1. Evaluación de Soluciones Existentes y Diferenciación

Se han analizado las principales soluciones en bienestar emocional asistido por IA y arteterapia digital. Aplicaciones como **Woebot**[19] y **Wysa**[20] ofrecen apoyo estructurado por chat, **Replika**[21] centra la interacción en la compañía conversacional, y las plataformas de arteterapia digital permiten la creación artística sin acompañamiento de IA.

MINDSCAPE se diferencia al combinar imagen y conversación: la obra actúa como iniciador del diálogo y la inteligencia artificial genera respuestas contextualizadas orientadas a la autoexploración emocional y reflexiva.

La siguiente tabla (ver Fig.1) resume las características de estas soluciones y destaca los aspectos innovadores que justifican el desarrollo propio de MINDSCAPE.

Aspecto	Woebot / Wysa	Replika	Plataformas de Arteterapia Digital	MINDSCAPE
Tipo de proyecto	Chatbot de apoyo emocional basado en texto y ejercicios de autoayuda	Asistente conversacional de compañía emocional y diálogo abierto	Entornos digitales para creación artística como expresión emocional	Plataforma multimodal de autoexploración que integra imagen y conversación
Objetivo principal	Apoyo emocional estructurado y seguimiento del estado de ánimo	Relación continua y expresión personal	Expresión y exploración emocional mediante creación artística	Reflexión y autoexploración utilizando la obra de arte como eje central de la conversación
Medio de interacción	Texto	Texto	Imagen, dibujo o pintura digital	Imagen como eje central + conversación textual
Inteligencia artificial	Modelos de lenguaje natural centrados en conversación textual	Modelos de lenguaje para diálogo abierto	Poca o nula IA, principalmente herramientas de creación	IA multimodal (texto + imagen) que genera respuestas contextualizadas y coherentes con la obra

Cuadro 1: Comparativa de proyectos existentes y diferenciación de MINDSCAPE

La comparativa evidencia que MINDSCAPE se distingue claramente de las soluciones existentes. Mientras que **Woebot**[19], **Wysa**[20] y **Replika**[21] se centran principalmente en la conversación textual, y las plataformas de arteterapia digital en la creación artística sin acompañamiento de IA, MINDSCAPE combina ambos enfoques: utiliza la imagen como detonante del diálogo y aplica inteligencia artificial multimodal para generar respuestas contextualizadas. Este enfoque permite ofrecer una experiencia de autoexploración emocional y reflexiva, integrando creatividad y bienestar emocional de manera única, lo que justifica el desarrollo propio del proyecto.

1.3. Motivación

El interés que me ha llevado a desarrollar esta web enfocada a la salud mental surge de la combinación de dos motivaciones principales. Por un lado, mi curiosidad por explorar el uso de la inteligencia artificial, tanto en la integración de sus capacidades como en la visualización de sus resultados, especialmente en contextos de interacción humana, me llevó a escoger este proyecto en el que la IA pudiera generar experiencias significativas y personalizadas.

Por otro lado, considero que la salud mental es un aspecto fundamental del bienestar humano que, lamentablemente, no siempre está al alcance de todas las personas. Las dificultades para acceder a profesionales especializados, las largas listas de espera o los recursos limitados en determinados contextos hacen que muchas personas no reciban la atención que necesitan. Desde un enfoque emocional, este proyecto pretende aportar un espacio seguro y responsable para la autoexploración y el acompañamiento reflexivo, sin reemplazar la intervención profesional, sino ofreciendo un apoyo complementario y accesible que fomente la conciencia emocional y el bienestar personal.

Finalmente, la combinación de ambos intereses tecnológico y social me permitió concebir una plataforma que no solo experimenta con la interacción entre usuario e inteligencia artificial, sino que también reflexiona sobre cómo estas tecnologías pueden aplicarse de manera responsable en contextos de bienestar emocional. El objetivo es ofrecer una herramienta que invite a la reflexión y a la autoexploración, promoviendo el uso seguro de la inteligencia artificial y fomentando experiencias positivas y seguras para los usuarios.

1.4. Objetivos

1.4.1. Objetivos generales

El objetivo general de MINDSCAPE es desarrollar una plataforma digital que facilite la autoexploración, la reflexión personal y el bienestar emocional, utilizando la interacción con obras de arte y la inteligencia artificial como catalizador del proceso. La plataforma busca crear un entorno en el que las personas usuarias puedan reflexionar sobre sus emociones, expresar su mundo interno y estimular la creatividad, promoviendo la introspección y el desarrollo del autoconocimiento de manera guiada.

Además, MINDSCAPE pretende integrar de manera coherente la inteligencia artificial multimodal y la experiencia artística, ofreciendo un recurso que combine análisis visual, diálogo conversacional y experimentación creativa. El objetivo general no se limita a la implementación tecnológica, sino que apunta a generar una experiencia significativa y enriquecedora, donde la obra de arte sirva como eje central de la interacción y la autoexploración.

1.4.2. Objetivos específicos

1. Selección y análisis de imagen

- Permitir la persona usuaria seleccionar una obra de arte o subir una imagen propia que represente su estado emocional.
- Proporcionar una vista previa de la imagen y la opción de enviar la imagen al sistema para iniciar la conversación.

2. Sistema de chat conversacional

- Utilizar la imagen seleccionada para generar respuestas contextualizadas y coherentes con la obra seleccionada.
- Implementar un chat que permita mantener una conversación fluida entre la persona usuaria y la IA basada en la imagen.
- Garantizar que los mensajes de la usuaria y del bot se diferencien visualmente y que la IA responda con coherencia, empatía y sin juicios, fomentando la introspección.
- Diferenciar entre imágenes generadas por la usuaria o imágenes seleccionadas de una biblioteca por la usuaria y adaptar el prompt
- Garantizar una detección automática de la finalización de la conversación para hacer que la experiencia sea lo más cómoda posible para la usuaria

3. Bienestar y feedback de la usuaria

- Generar un pequeño resumen final de cada sesión, destacando los puntos tratados y ofreciendo una reflexión acompañada de un disclaimer sobre la importancia de la intervención profesional cuando sea necesario.
- Incorporar un sistema de valoración o comentarios por parte de la persona usuaria para mejorar la experiencia del sistema en futuras sesiones.

4. Experiencia de uso (UI/UX)

- Diseñar una interfaz responsiva que se adapte a distintos tamaños de pantalla, evitando scrolls innecesarios.
- Garantizar una navegación clara entre las secciones de selección de imagen y chat.
- Mantener un diseño calmado, empático y visualmente limpio, priorizando la comodidad y seguridad emocional de la persona usuaria.

1.5. Estructura de la memoria

La memoria se estructura abordando el proyecto desde su justificación inicial hasta su implementación y evaluación final.

En primer lugar, la introducción contextualiza el trabajo, se revisa el estado del arte, se expone la motivación y se definen los objetivos tanto generales como específicos. A continuación, se presenta la planificación del proyecto, se describe la organización temporal del desarrollo, incluyendo la planificación inicial y final, así como el diagrama de Gantt y el análisis de costes asociados.

Posteriormente, observamos el análisis tecnológico donde se examinan las distintas alternativas técnicas para el desarrollo de la aplicación web, comparando tecnologías de frontend, backend y modelos de inteligencia artificial, y justificando finalmente las herramientas seleccionadas. La sección de estructura del proyecto detalla la organización general del sistema y profundiza en la arquitectura del backend y del frontend, incluyendo ejemplos de flujo de trabajo.

Seguidamente, encontramos el apartado de implementación, donde se describe de manera detallada el desarrollo tanto del backend como del frontend, analizando los principales archivos y componentes que conforman la aplicación. Posteriormente, analizamos las pruebas realizadas, donde se evalúa el funcionamiento del sistema, los tests de usuarios, los resultados obtenidos y la evaluación que se hace de los resultados. Para finalizar la memoria concluye con un apartado de mejoras a futuro y conclusiones, en el que se reflexiona sobre los resultados obtenidos y se proponen posibles líneas de evolución del proyecto.

2. Planificación del proyecto y costes

2.1. Planificación

2.1.1. Planificación inicial

La planificación de MINDSCAPE se ha realizado siguiendo una metodología de desarrollo iterativa, dividida en fases claramente definidas. Estas fases incluyen la investigación y selección de tecnologías de inteligencia artificial, el estudio de conceptos relacionados con el bienestar emocional, el diseño de la arquitectura del sistema, el desarrollo de la aplicación web y la validación progresiva del producto mediante pruebas y experimentación.

El proyecto presenta una complejidad añadida al combinar el uso de inteligencia artificial con el desarrollo de una interfaz web completa y la aplicación de estos recursos en un contexto sensible como el de la salud mental. Por este motivo, se ha considerado necesario adoptar un enfoque de planificación flexible que permitiera ajustar tanto los aspectos técnicos como los de diseño y contenido a medida que avanzaba el desarrollo.

Este enfoque iterativo ha facilitado la incorporación de mejoras continuas, la corrección de errores y la adaptación del sistema a los requisitos funcionales y éticos del proyecto, permitiendo evaluar cada fase antes de avanzar a la siguiente y garantizando una evolución controlada y coherente de la plataforma.

Fase 1: Investigación y Análisis Preliminar Semanas 1-2 Aproximadamente (50 horas)

Análisis de proyectos y plataformas existentes relacionadas con la interacción humano-máquina, la inteligencia artificial multimodal y aplicaciones digitales orientadas al bienestar emocional. Estudio de conceptos básicos vinculados a la autoexploración, la arteterapia y el uso de la inteligencia artificial en contextos sensibles. Evaluación de tecnologías de inteligencia artificial para el análisis de imagen y el procesamiento del lenguaje natural, así como de distintos proveedores de modelos de IA, con especial atención a su facilidad de integración en aplicaciones web mediante herramientas como **Google AI Studio** [16] y APIs basadas en servicios cloud.

Fase 2: Diseño de la Arquitectura del Sistema Semanas 3-4 Aproximadamente (45 horas)

Definición de la arquitectura general de la aplicación, incluyendo la separación entre frontend y backend, el flujo de datos entre el usuario, la interfaz web y el modelo de inteligencia artificial, y la gestión de sesiones de conversación. Diseño de una arquitectura modular basada en servicios y comunicación mediante **APIs REST**, utilizando **FastAPI**[15] como framework backend, con el objetivo de garantizar escalabilidad, mantenibilidad y una integración eficiente con servicios de inteligencia artificial.

Fase 3: Desarrollo del Backend Semanas 5-7 Aproximadamente (85 horas)

Implementación del backend del sistema utilizando el lenguaje de programación

Python[11], encargado de gestionar las peticiones de la aplicación, la comunicación con los modelos de inteligencia artificial y el control del flujo conversacional. Desarrollo de la lógica de negocio, la gestión de sesiones y la definición de posibles mecanismos de almacenamiento de datos, teniendo en cuenta criterios de privacidad, seguridad y tratamiento responsable de la información.

Fase 4: Desarrollo del Sistema de Inteligencia Artificial Semanas 8-10 Aproximadamente(65 horas)

Integración de modelos de procesamiento del lenguaje natural y análisis de imagen mediante servicios de inteligencia artificial accesibles a través de **Google AI Studio**[16]. Diseño y ajuste de los prompts y flujos conversacionales orientados a la reflexión personal y la autoexploración, así como la implementación de mecanismos de control del tono, coherencia y límites éticos de las respuestas generadas por la inteligencia artificial.

Fase 5: Desarrollo del Frontend Semanas 11-13 Aproximadamente(80 horas)

Implementación de la interfaz de usuario mediante tecnologías web modernas, utilizando **React**[5] como librería principal para la construcción de una aplicación web interactiva, responsive y centrada en el diálogo. Desarrollo de componentes específicos para la selección de imágenes, la visualización del chat y la gestión de la experiencia de usuario, incorporando principios de diseño orientados a la claridad, la accesibilidad y el confort emocional.

Fase 6: Integración y Pruebas Semanas 13-14 Aproximadamente(40 horas)

Integración de los distintos componentes del sistema frontend desarrollado en **React**[5], backend en **Python**[11] con **FastAPI**[15] y servicios de inteligencia artificial proporcionados a través de **Google AI Studio**[16] y realización de pruebas funcionales y de usabilidad. Validación del correcto funcionamiento del flujo de interacción imagen–conversación, la estabilidad del sistema y el cumplimiento de los criterios éticos definidos durante el desarrollo.

Fase 7: Documentación Semanas 15-16 Aproximadamente(20 horas)

Elaboración de la documentación del proyecto y redacción de la memoria del Trabajo de Final de Grado.

2.1.2. Planificación final

Durante el desarrollo de MINDSCAPE, la planificación inicial ha servido como guía general para organizar las tareas y fases del proyecto. No obstante, la ejecución real del trabajo ha requerido cierta flexibilidad y adaptaciones sobre la marcha debido a contratiempos, ajustes de diseño y descubrimientos durante la implementación.

En varios momentos, fue necesario realizar pequeñas transiciones entre fases para corregir aspectos que no funcionaban según lo previsto o replantear decisiones inicialmente tomadas. Por ejemplo, ciertos elementos del backend y la integración con los modelos de inteligencia artificial tuvieron que modificarse para asegurar una comunicación más eficiente y coherente con el frontend. Asimismo, aspectos de la

interfaz se rediseñaron sobre la marcha para mejorar la experiencia y la claridad visual, ajustando la planificación original de la fase de frontend.

A pesar de estos cambios, el esquema original de fases ha proporcionado una estructura sólida que ha permitido gestionar el proyecto de forma organizada y progresiva. La planificación final refleja, por tanto, un proceso iterativo y adaptativo en el que se ha buscado un equilibrio entre cumplir los objetivos definidos y responder a las necesidades prácticas que surgieron durante el desarrollo del proyecto.

2.1.3. Diagrama de Gantt

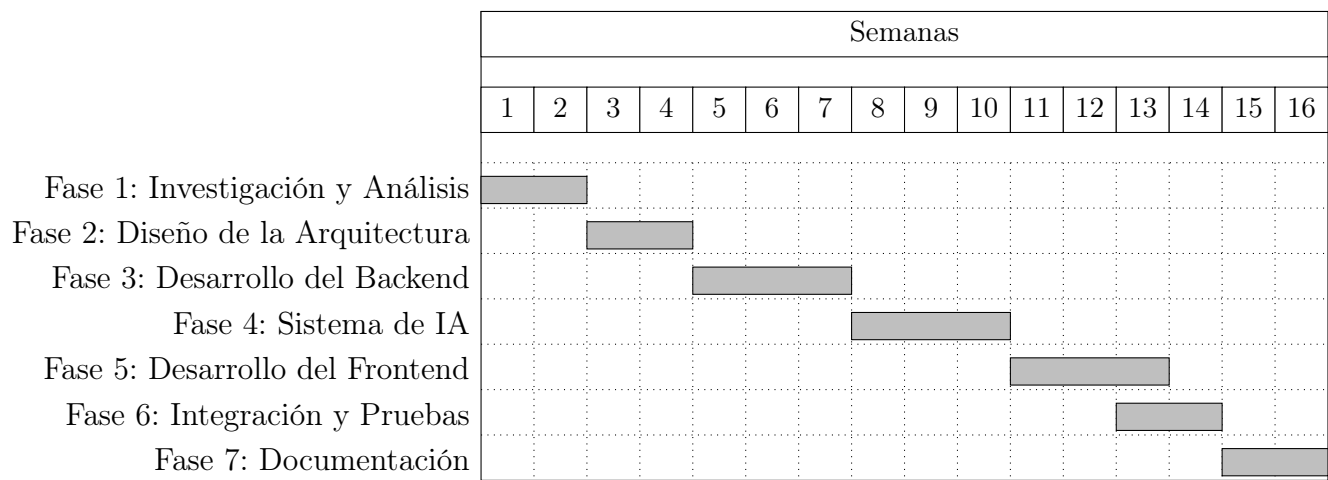


Figura 1: Diagrama de Gantt del proyecto

2.2. Costes

La implementación de un proyecto como MINDSCAPE conllevaría diversos costes económicos si se considerara su desarrollo en un contexto empresarial o de producto comercial. Entre los principales costes se pueden destacar los siguientes:

1. **Licencias y servicios de software:** El uso de plataformas de inteligencia artificial como Google AI Studio o servicios de procesamiento de lenguaje natural y análisis de imagen puede implicar costes por suscripción o pago por uso según volumen de consultas. Asimismo, el uso de librerías y frameworks específicos para desarrollo web y backend, aunque en muchos casos sean open source, podría requerir soporte profesional o versiones avanzadas de pago para entornos empresariales.
2. **Infraestructura tecnológica:** Para un despliegue real de MINDSCAPE, sería necesario contar con servidores o servicios cloud que gestionen las peticiones del frontend, el backend y los modelos de inteligencia artificial. Esto incluye hosting web, bases de datos, almacenamiento seguro de imágenes y datos de usuario, así como recursos de escalabilidad para soportar un número mayor de usuarios concurrentes.

3. **Diseño y desarrollo:** La creación de la plataforma requiere la participación de perfiles especializados en desarrollo frontend (React), desarrollo backend (Python con FastAPI), integración de IA y experiencia de usuario (UI/UX). En un contexto empresarial, estos profesionales supondrían costes por horas de trabajo, tanto en programación como en diseño visual y pruebas de usabilidad.
4. **Subcontratación y consultoría especializada:** Si se quisiera mejorar la precisión de los modelos de IA o implementar características avanzadas de arteterapia digital, podría ser necesario recurrir a consultoría externa en inteligencia artificial, salud mental o arteterapia. Esto incluye la asesoría ética y profesional para garantizar que la plataforma cumpla criterios de responsabilidad y seguridad en la interacción con las personas usuarias.
5. **Otros costes asociados:** Podrían incluirse gastos de marketing, registro de dominios, mantenimiento de la plataforma, actualizaciones y soporte técnico para garantizar la continuidad del servicio, así como la creación de contenidos artísticos o bibliotecas de imágenes para la interacción inicial.

Desde el punto de vista del trabajo realizado en este proyecto, se estima que el desarrollo del proyecto ha requerido aproximadamente 315 horas de trabajo, lo que, considerando un salario promedio de 7 €/hora para estudiantes en prácticas de la UB, supone un coste estimado de 2.205 euros. Gracias al uso del programa para estudiantes de Google AI Studio, gran parte del desarrollo se ha podido realizar sin costes adicionales, al disponer de acceso gratuito a los modelos de inteligencia artificial necesarios para las pruebas y la implementación. No obstante, es importante destacar que en un entorno de producción real, donde el número de solicitudes a los servicios de IA sería mucho mayor, sí sería necesario un presupuesto adicional para cubrir los costes asociados al uso continuo de estos servicios.

2.2.1. Costes medioambientales

Aunque a veces no lo consideremos, el proyecto también ha tenido un impacto medioambiental real, que, aunque limitado, merece ser considerado. Durante su desarrollo se han utilizado modelos de inteligencia artificial tanto para la programación y desarrollo del código como para la implementación y testeo de funcionalidades, mediante una API que permite integrar el modelo de IA directamente en la aplicación, formando parte activa del proyecto. Este uso implica un consumo energético asociado a los servidores en la nube, que, aunque reducido por el alcance del proyecto y el uso puntual de las herramientas, constituye una huella de carbono que se debe reconocer. Por tanto, incluso en proyectos de pequeña escala, el uso de IA integrada conlleva un coste medioambiental indirecto que conviene tener presente.

3. Análisis tecnológico

En este apartado se presenta el estudio de las tecnologías disponibles para el desarrollo de MINDSCAPE, evaluando alternativas de frontend, backend e inteligencia artificial. Se analizan las ventajas, limitaciones y posibilidades de integración de cada opción, justificando la elección final en función de los objetivos del proyecto, la experiencia de usuario y la escalabilidad de la plataforma. Este análisis permite comprender el razonamiento detrás de cada decisión tecnológica y cómo contribuyen a crear un entorno interactivo centrado en la autoexploración y la reflexión personal.

3.1. Métodos para crear una web

1. **HTML/CSS/JS puro:** Permite crear cualquier tipo de web con control total sobre diseño y funcionalidad, ideal para desarrolladores con experiencia.
Pros: Control total, ligero y rápido.
Contras: Requiere conocimientos técnicos y desarrollo más lento.
2. **WordPress:** Plataforma flexible con gran variedad de temas y plugins, adecuada para blogs, webs corporativas y portafolios.
Pros: Fácil de usar, muchos plugins y plantillas.
Contras: Puede ser pesado y necesita mantenimiento constante.
3. **Wix / Squarespace / Webflow:** Soluciones fáciles de usar con edición visual, perfectas para principiantes o proyectos que necesitan lanzarse rápido.
Pros: Muy rápido, no requiere programar, diseño visual intuitivo.
Contras: Menos control técnico, limitaciones de personalización, costes recurrentes.
4. **Generadores de webs estáticas (Gatsby, Hugo, Jekyll):** Ideales para blogs o landing pages rápidas, con alta velocidad y seguridad, aunque menos intuitivos para principiantes.
Pros: Muy rápido, seguro y ligero.
Contras: Menos intuitivo, cambios dinámicos más complejos.
5. **Programación personalizada / Full stack:** Crear la web completamente a medida, ofreciendo máxima personalización y escalabilidad, pero requiere tiempo y conocimientos técnicos.
Pros: Máxima personalización y control.
Contras: Requiere mucho tiempo y experiencia técnica.

Se ha optado por la programación personalizada como enfoque para el desarrollo de MINDSCAPE, ya que este proyecto requiere un backend sólido capaz de gestionar llamadas a APIs y una comunicación fluida y adaptable entre frontend y backend.

Este nivel de control permite ajustar la estructura y los flujos de datos según las necesidades específicas del sistema, algo que resultaría más limitado en plataformas predefinidas. Además, el frontend buscaba ser inmersivo y ofrecer una experiencia visual, por lo que desarrollar la interfaz de manera personalizada permitió explorar nuevas formas de diseño y aprendizaje de tecnologías modernas, más allá del HTML y CSS tradicionales. Esta decisión garantiza, por tanto, la máxima flexibilidad y escalabilidad, adaptándose a los objetivos de interacción y autoexploración de la plataforma.

3.2. Posibles tecnologías

3.2.1. Frontend

El frontend representa la capa de interacción directa con la persona usuaria y es clave para transmitir la experiencia visual, emocional y funcional de la plataforma. En un proyecto como MINDSCAPE, donde la interfaz tiene un papel relevante en la inmersión y la comodidad del usuario, existen distintas tecnologías que pueden emplearse para su desarrollo:

1. **HTML, CSS y JavaScript puro:** Permite construir interfaces web sin depender de frameworks o librerías externas.
Pros: Control total sobre la estructura y el estilo, aplicación ligera y sin dependencias.
Contras: Escasa reutilización de componentes y mayor complejidad en proyectos interactivos o de mayor tamaño.
2. **Vue.js**[6]: Framework progresivo de JavaScript orientado a la creación de interfaces de usuario reactivas.
Pros: Curva de aprendizaje relativamente sencilla, documentación clara y buen rendimiento.
Contras: Comunidad más reducida que otras alternativas, lo que puede dificultar la resolución de problemas específicos.
3. **Angular**[7]: Framework de código abierto desarrollado por Google, basado en TypeScript y orientado a aplicaciones de gran escala.
Pros: Arquitectura robusta, estructura clara y herramientas integradas.
Contras: Curva de aprendizaje elevada y mayor complejidad inicial.
4. **React**[5]: Biblioteca de JavaScript de código abierto orientada a la creación de interfaces mediante componentes reutilizables.
Pros: Amplia comunidad, gran ecosistema de librerías y enfoque modular.
Contras: Requiere integración con herramientas adicionales para ciertas funcionalidades.

5. **Librerías de UI (Chakra UI[8], Material UI[9], Tailwind UI[10]):** Conjuntos de componentes visuales predefinidos que se integran con frameworks modernos.

Pros: Desarrollo rápido, consistencia visual y accesibilidad.

Contras: Limitaciones estéticas si no se personalizan adecuadamente.

3.2.2. Backend

El backend se encarga de gestionar la lógica de la aplicación, procesar las peticiones del frontend y coordinar la comunicación con servicios externos, como la API de inteligencia artificial. En MINDSCAPE, actúa como intermediario entre la interfaz y los modelos de IA, controlando el flujo de la conversación, el envío de imágenes y la estructuración de las respuestas. Por este motivo, el backend debe ser flexible y fácilmente adaptable, permitiendo ajustar los formatos de comunicación y el comportamiento del sistema según las necesidades del proyecto. Algunas tecnologías habituales para este propósito son:

1. **Express.js (Node.js)[12]:** Framework minimalista para la creación de APIs REST en JavaScript.

Pros: Uso del mismo lenguaje en frontend y backend, gran cantidad de módulos disponibles.

Contras: Manejo de errores y escalabilidad más complejo en aplicaciones grandes.

2. **Django (Python)[13]:** Framework de alto nivel que sigue el patrón MVC y proporciona muchas funcionalidades integradas.

Pros: Seguridad robusta y desarrollo rápido de aplicaciones completas.

Contras: Menor flexibilidad para arquitecturas altamente personalizadas.

3. **Flask (Python)[14]:** Microframework ligero y flexible para el desarrollo de APIs.

Pros: Simplicidad y control total sobre la arquitectura.

Contras: Requiere configuración manual de muchos componentes.

4. **FastAPI (Python)[15]:** Framework moderno para la creación de APIs rápidas y eficientes, basado en estándares modernos.

Pros: Alto rendimiento, validación automática de datos y documentación interactiva integrada.

Contras: Menor madurez que otros frameworks más antiguos.

3.2.3. Modelos de inteligencia artificial

Los modelos de inteligencia artificial constituyen el núcleo funcional de MINDS-CAPE, ya que permiten la interpretación de texto e imagen y la generación de respuestas personalizadas. Existen diversas opciones para integrar este tipo de capacidades en una aplicación web:

1. **Modelos de lenguaje mediante APIs externas:** Servicios proporcionados por grandes plataformas de IA que ofrecen modelos preentrenados accesibles mediante llamadas HTTP.
Pros: Alta calidad en la generación de respuestas y facilidad de integración.
Contras: Dependencia de servicios externos y coste asociado al uso.
2. **Modelos de código abierto:** Modelos desplegados en servidores propios, utilizando librerías especializadas.
Pros: Mayor control sobre el comportamiento del modelo y ausencia de costes por uso.
Contras: Requiere infraestructura propia y mayor complejidad técnica.
3. **Google AI Studio (Gemini)**[16]: Plataforma de Google que permite acceder a modelos multimodales avanzados mediante API.
Pros: Capacidad para procesar texto e imagen, buena escalabilidad y soporte oficial.
Contras: Dependencia del proveedor y limitaciones según el plan de uso.
4. **Otras APIs comerciales de IA (OpenAI, Azure OpenAI, Anthropic)**[17]: Servicios que ofrecen modelos avanzados de lenguaje y, en algunos casos, capacidades multimodales.
Pros: Modelos muy maduros, amplia documentación y adopción en proyectos reales.
Contras: Costes variables según uso y dependencia de plataformas externas.

3.3. Tecnologías escogidas

3.3.1. Frontend

Para el desarrollo del frontend se optó por una combinación de **React y Chakra UI**, utilizando React [5] como base estructural para la construcción de la aplicación y la gestión de componentes, y Chakra como sistema de diseño para potenciar los aspectos visuales y la experiencia de usuario. React permite organizar la interfaz de forma modular y escalable, facilitando la creación de flujos interactivos complejos como la selección de imágenes y el sistema de conversación. Por su parte, Chakra

UI[8] aporta un conjunto de componentes estilizados, accesibles y fácilmente personalizables, junto con utilidades para animaciones y estados visuales que contribuyen a un resultado más moderno, coherente e inmersivo.

Esta decisión también responde a la naturaleza del propio proyecto. MINDSCAPE aborda ámbitos innovadores como la interacción multimodal y la autoexploración emocional mediante inteligencia artificial, por lo que se consideró adecuado apoyarse en tecnologías actuales que permitiesen experimentar con nuevas formas de diseño y presentación visual. La combinación de React y Chakra (ver Fig.2) facilita este enfoque creativo, permitiendo desarrollar una interfaz flexible y expresiva sin renunciar a la mantenibilidad ni a la claridad del código.

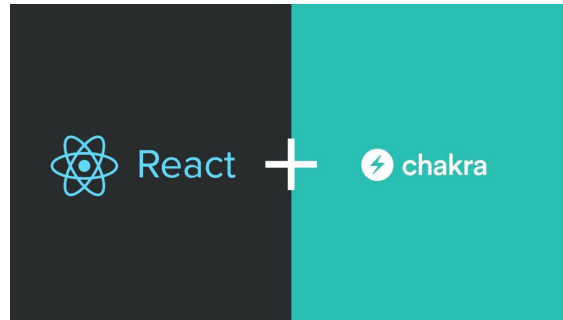


Figura 2: React y Chakra UI

3.3.2. Backend

Para el desarrollo del backend se optó por **FastAPI** utilizando **Python** (ver Fig.3). Esta decisión se fundamenta, en primer lugar, en el amplio uso y conocimiento previo de Python adquirido a lo largo de la carrera, lo que ha permitido abordar el desarrollo con mayor solvencia y eficiencia. Además, Python[11] es un lenguaje moderno, versátil y ampliamente adoptado en entornos relacionados con la inteligencia artificial, lo que lo convierte en una opción especialmente adecuada para un proyecto como MINDSCAPE.

FastAPI [15] resulta particularmente apropiado para este contexto, ya que facilita la creación y gestión de endpoints, que son puntos de acceso (URLs) que permiten recibir peticiones desde el frontend u otros servicios y devolver respuestas estructuradas, de forma clara y eficiente. Esto permite un control preciso de la comunicación entre frontend y backend, así como de las llamadas a la API de inteligencia artificial. Asimismo, su integración nativa con herramientas como Swagger [18] ha sido un factor relevante, ya que posibilita la documentación automática de los endpoints y la realización de pruebas directas para verificar su correcto funcionamiento durante el desarrollo. Esta combinación ha favorecido un desarrollo ágil, ordenado y fácilmente mantenible del backend.

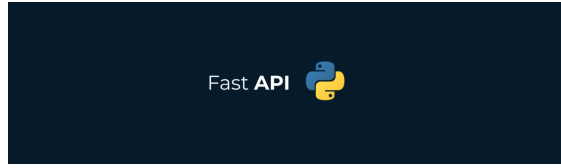


Figura 3: FastAPI

3.3.3. Modelos de inteligencia artificial

Para el modelo de inteligencia artificial se optó finalmente por la API de **Google AI Studio** [16] por varios motivos clave. En primer lugar, ofrece una amplia variedad de modelos disponibles, incluyendo capacidades tanto de procesamiento de texto como de análisis de imágenes, lo que encaja directamente con los requisitos multimodales de MINDSCAPE. Además, su implementación resulta sencilla y está acompañada de una interfaz clara e intuitiva, lo que facilita la integración y el ajuste de los flujos conversacionales.

Otro factor determinante ha sido su accesibilidad, ya que Google proporciona programas orientados a estudiantes que incluyen créditos gratuitos, lo que ha permitido desarrollar y probar el sistema sin un coste elevado. A esto se suma un modelo de precios razonable, basado en el uso de la API, que resulta adecuado para proyectos académicos y escalables como el presente.

De entre los distintos modelos de inteligencia artificial disponibles, se eligió finalmente Gemini 2.5 Flash (ver Fig.4). Esta decisión se basa en que se trata de un modelo ligero, con tiempos de respuesta rápidos, lo que lo hace especialmente adecuado para la interacción fluida que requiere MINDSCAPE. Además, incorpora capacidades de análisis de imágenes, lo que permite integrar de manera eficiente la interpretación visual junto con la generación de respuestas textuales dentro de la plataforma.

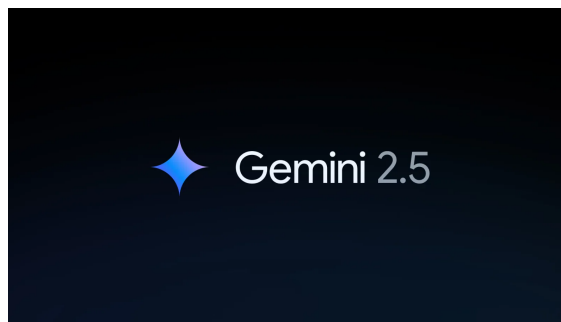


Figura 4: Gemini 2.5

4. Estructura del proyecto

4.1. Estructura general del proyecto

La arquitectura de MINDSCAPE (ver Fig.5) se basa en un modelo cliente-servidor clásico, adaptado a las particularidades de una plataforma web interactiva que integra procesamiento de inteligencia artificial. En este esquema, el frontend actúa como cliente, encargado de gestionar la interfaz de usuario, incluyendo la selección y visualización de imágenes, la presentación de la conversación y la interacción directa con el usuario. Por su parte, el backend funciona como servidor y centraliza la lógica de negocio, la gestión de sesiones, el enrutamiento de solicitudes y la comunicación con la API de inteligencia artificial. Además, procesa los datos recibidos del frontend, los transforma en prompts adecuados para la API de Gemini 2.5 Flash y devuelve las respuestas generadas para su presentación.

La comunicación entre frontend y backend se realiza mediante endpoints REST, lo que asegura un flujo de información estructurado y eficiente. Este diseño aporta modularidad y escalabilidad, permitiendo actualizar o ampliar cualquiera de los componentes sin afectar al resto del sistema. Asimismo, facilita la integración de funcionalidades multimodales y futuras mejoras, garantizando un sistema robusto, coherente con los objetivos del proyecto.

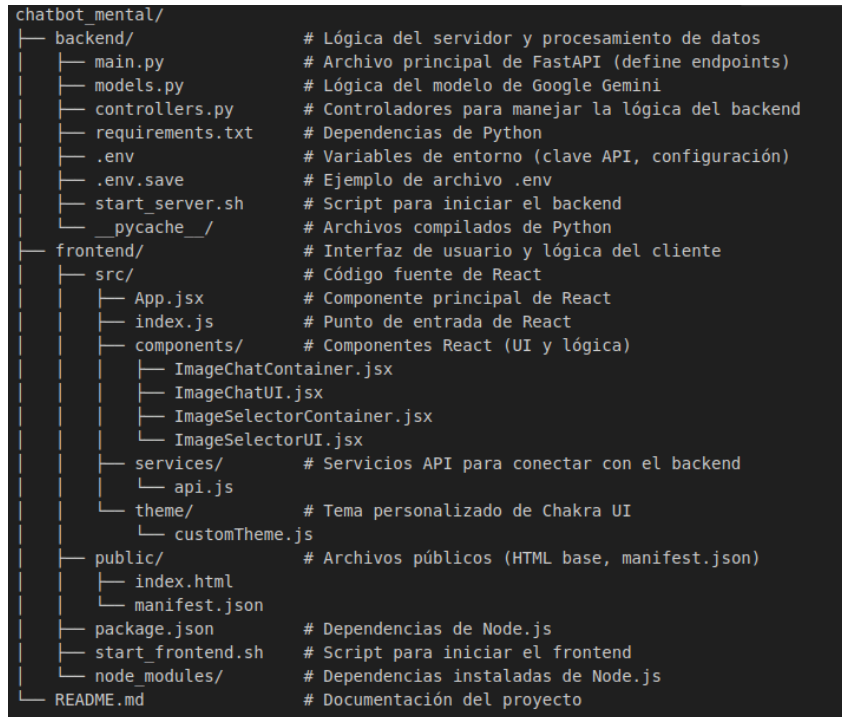


Figura 5: Estructura Proyecto

4.2. Estructura del backend

El backend de MINDSCAPE está enfocado de forma modular y organizado, repartiendo responsabilidades entre las diferentes capas. Su función principal es la de gestionar las solicitudes recibidas desde el frontend, transformar los datos recibidos en prompts adecuados para el modelo de inteligencia artificial y devolver las respuestas de forma estructurada al cliente para que el usuario las vea. Aparte de responder al frontend el backend también tiene otras funciones alternativas como generar un resumen de la conversación y crear un PDF.

backend/main.py

- Punto de entrada del servidor FastAPI.
- Define los endpoints REST: `/start_chat`, `/continue_chat`, `/summary_chat` y `/summary_pdf`.
- Gestiona la recepción de datos desde el frontend y la respuesta del backend, delegando la lógica a `ChatbotController`.
- Incluye middleware para permitir CORS (mecanismo de seguridad que permite controlar el acceso a los recursos de un servidor desde el navegador) y maneja la validación básica de solicitudes.

backend/controllers.py

- Incluye la clase `ChatbotController` que centraliza la lógica de alto nivel del backend.
- Métodos principales: `start_chat`, `continue_chat` y `generate_summary`.
- Intermedia entre el frontend y `GeminiModel` para procesar la conversación y generar respuestas.
- Función adicional `generar_pdf_resumen` para crear PDFs del historial de la conversación.

backend/models.py

- Clase `GeminiModel` encargada de la comunicación con la API de Google AI Studio (Gemini 2.5 Flash).
- Configura la API mediante la clave de entorno.
- Construye prompts diferenciando entre imágenes predeterminadas o propias del usuario.
- Parseo seguro de las respuestas de la IA en formato JSON, extrayendo mensaje y estado de finalización.

Scripts y configuración

- `.env`: Variables de entorno (clave de API, configuración).
- `requirements.txt`: Dependencias de Python necesarias.
- `start_server.sh`: Script para iniciar el servidor de forma sencilla.

4.2.1. Ejemplo de Flujo de trabajo del backend

Este diagrama (ver Fig.6) ilustra un ejemplo del flujo de trabajo que sigue MINDSCAPE para iniciar una conversación, comenzando con el procesamiento de una imagen seleccionada o subida por el usuario

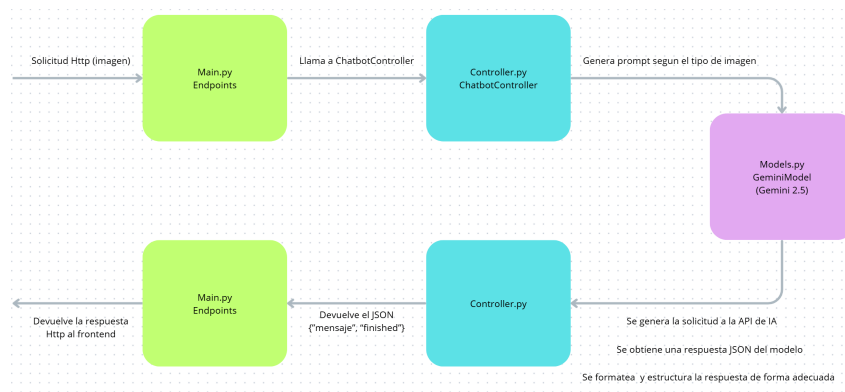


Figura 6: Workflow Backend

4.3. Estructura del frontend

El frontend de MINDSCAPE está diseñado de manera modular y por componentes, priorizando la claridad en la separación de responsabilidades y la interacción fluida con el usuario. Su función principal es gestionar la interfaz, mostrar imágenes, manejar la conversación y comunicarse con el backend mediante llamadas a endpoints REST. Se ha optado por separar de manera explícita la lógica de la aplicación de la presentación visual, de modo que los contenedores gestionen el estado y la comunicación con el backend, mientras que los componentes de interfaz se centran únicamente en la representación y la experiencia del usuario. Esta decisión facilita la mantenibilidad y escalabilidad del sistema, permitiendo introducir nuevas funcionalidades, modificar comportamientos o rediseñar la interfaz sin afectar la lógica subyacente, y favorece la reutilización de componentes en distintos contextos dentro de la plataforma.

frontend/src/App.jsx

- Componente principal de la aplicación.

- Gestiona el estado global de la aplicación y coordina los distintos contenedores y componentes.
- Integra el tema personalizado de Chakra UI y los estilos generales de la aplicación.

frontend/src/index.js

- Punto de entrada de React.
- Renderiza el componente principal `App.jsx` en el DOM.

frontend/src/components/

- `ImageChatContainer.jsx`: Contenedor principal del chat con lógica de manejo de mensajes y coordinación de subcomponentes.
- `ImageChatUI.jsx`: Interfaz de usuario del chat, responsable de la visualización de mensajes y la interacción con el usuario.
- `ImageSelectorContainer.jsx`: Contenedor para la selección de imágenes, maneja la lógica de cambio y subida de imágenes.
- `ImageSelectorUI.jsx`: Interfaz de usuario de selección de imágenes, botones, previsualización y feedback visual.

frontend/src/services/api.js

- Define funciones para interactuar con los endpoints del backend.
- Encapsula llamadas a `/start_chat`, `/continue_chat`, `/summary_chat` y `/summary_pdf`.
- Facilita la separación entre la lógica de UI y las solicitudes HTTP.

frontend/src/theme/customTheme.js

- Archivo de configuración de Chakra UI.
- Define colores, tipografías y estilos globales personalizados para toda la aplicación.

frontend/public/

- `index.html`: Estructura HTML base donde se monta la aplicación React.
- `manifest.json`: Configuración general de la aplicación web (PWA, iconos, nombre, etc.).

Scripts y configuración

- `package.json`: Dependencias de Node.js y scripts de ejecución.
- `start_frontend.sh`: Script para iniciar el servidor de desarrollo de manera sencilla.
- `node_modules/`: Dependencias instaladas de Node.js.

5. Implementación

5.1. Implementación del backend

A continuación se describe la implementación del backend de MINDSCAPE, poniendo el foco en cómo se han aplicado las decisiones de diseño adoptadas y en la interacción entre los distintos componentes que lo conforman, con el objetivo de ofrecer una visión global de su funcionamiento y organización.

5.1.1. `main.py`

El archivo `main.py` constituye el punto de entrada del backend y se encarga de configurar la API REST mediante el framework FastAPI. En primer lugar, se inicializa la aplicación y se instancia el controlador `ChatbotController`, responsable de gestionar la lógica de negocio asociada al chatbot basado en el análisis de imágenes y texto mediante inteligencia artificial. Asimismo, se configura el middleware de CORS con permisos amplios para permitir la comunicación entre el backend y el frontend.

A nivel funcional, `main.py` define los siguientes endpoints principales, cada uno con una responsabilidad claramente delimitada:

- **`/health`**: Endpoint de comprobación del estado del servidor. Permite verificar si el backend está operativo y respondiendo correctamente a las peticiones.
- **`/start_chat`**: Endpoint encargado de iniciar una nueva conversación. Recibe una imagen proporcionada por el usuario o una imagen predeterminada junto con un indicador de su origen, y devuelve la pregunta inicial generada por el modelo de inteligencia artificial, marcando el comienzo del diálogo.
- **`/continue_chat`**: Endpoint que gestiona la continuación de la conversación. Recibe el historial completo de mensajes intercambiados hasta el momento y devuelve la siguiente respuesta del chatbot, incluyendo un indicador que señala si la conversación ha finalizado.
- **`/summary_chat`**: Endpoint destinado a la generación de un resumen textual de la conversación. Analiza el historial completo del diálogo y devuelve un texto sintético que recoge los temas principales y el comportamiento del usuario.
- **`/summary_pdf`**: Endpoint que ofrece la misma funcionalidad de resumen, pero presentando el resultado en formato PDF descargable, generado dinámicamente y enviado al cliente mediante una respuesta en streaming.

En conjunto, este archivo se centra exclusivamente en el enrutamiento HTTP, la validación básica de las peticiones y la estructuración de las respuestas, delegando toda la lógica relacionada con la inteligencia artificial y el procesamiento de la conversación al controlador correspondiente. Esta separación contribuye a un diseño más claro, mantenible y escalable.

5.1.2. controllers.py

El archivo `controllers.py` funciona como la capa de coordinación del backend, actuando como intermediario entre los endpoints definidos en `main.py` y la lógica del modelo de inteligencia artificial. Su propósito principal es centralizar la lógica de alto nivel del sistema y desacoplar la gestión de las peticiones HTTP del procesamiento interno del chatbot.

En este archivo se define la clase `ChatbotController`, que instancia el modelo `GeminiModel` y expone métodos claros para las principales operaciones del sistema: iniciar una conversación a partir de una imagen (`/start_chat`), continuar el diálogo utilizando el historial de mensajes (`/continue_chat`) y generar un resumen global de la interacción (`/get_summary`). De este modo, el controlador se limita a delegar las tareas específicas al modelo de IA, manteniendo una estructura limpia y fácilmente extensible.

Además, `controllers.py` incluye la función `generar_pdf_resumen`, responsable de convertir el resumen textual de la conversación en un documento PDF. Esta funcionalidad añade valor al backend al permitir la exportación de los resultados de la sesión en un formato descargable, sin interferir con la lógica principal del chatbot. En conjunto, este componente contribuye a una arquitectura modular y bien organizada, facilitando el mantenimiento y la evolución futura del sistema.

5.1.3. models.py

El archivo `models.py` actúa como la capa encargada de encapsular toda la interacción con el modelo de inteligencia artificial, aislando al resto del backend de los detalles específicos de la API de Google AI Studio. Esta separación permite que el sistema sea más mantenible y facilita posibles cambios de proveedor o de modelo en el futuro sin afectar al resto de la arquitectura. La clase `GeminiModel` se responsabiliza tanto de la configuración inicial del modelo como de la construcción de prompts, el envío de solicitudes y el tratamiento de las respuestas generadas. Esta clase está estrechamente vinculada con `ChatbotController`, que inicializa `GeminiModel` y le delega las operaciones de inicio de conversación, continuación del diálogo y generación de resúmenes, manteniendo así una estructura modular y fácilmente extensible.

En términos funcionales, este componente realiza las siguientes tareas principales:

Configuración e inicialización del modelo:

- Carga de variables de entorno y validación de la clave de la API.
- Inicialización del modelo Gemini 2.5 Flash mediante la librería oficial.

Inicio de la conversación:

- Procesamiento de la imagen enviada por el usuario, incluyendo su apertura, redimensionado y preparación para el análisis por parte del modelo de IA.

- Generación de la pregunta inicial del diálogo, diferenciando el comportamiento del sistema según se trate de una imagen predeterminada o de una creación propia del usuario.

Gestión de la conversación:

- Continuación del diálogo a partir del historial completo de mensajes intercambiados.
- Mantenimiento de la coherencia contextual y de un tono empático y terapéutico a lo largo de la interacción.
- Detección automática del finalizado de la conversación

Generación de resúmenes:

- Conversión del historial de mensajes en texto estructurado.
- Creación de un resumen sintético que destaca los temas principales y el comportamiento del usuario.

Normalización de respuestas:

- Parseo robusto de las respuestas del modelo, garantizando un formato consistente.
- Extracción controlada del mensaje generado y del indicador de finalización de la conversación para su consumo por el resto del sistema.

Las respuestas del modelo Gemini 2.5 Flash se estructuran en formato JSON con los campos `message` y `finished`. Este formato ha sido estructurado así, ya que permite al frontend interpretar de manera clara el contenido generado por la IA y detectar cuándo la conversación ha finalizado, posibilitando la adaptación automática de la interfaz y la gestión de la interacción con el usuario de forma fluida y coherente.

5.2. Implementación del frontend

5.2.1. App.jsx

El componente `App` actúa como punto de entrada del frontend y concentra la orquestación general de la aplicación, gestionando tanto el estado global como la navegación entre vistas. A través de React Hooks (`useState`), se controlan tres elementos clave: la vista activa (`home` o `image-chat`), la imagen seleccionada por el usuario y la pregunta inicial generada por el backend. Esta aproximación permite implementar una navegación sencilla basada en estado, evitando la complejidad de un sistema de rutas para un flujo lineal y controlado.

Desde el punto de vista estructural, App define claramente tres fases de interacción: la pantalla inicial de bienvenida, el selector de imagen y la interfaz de chat. Cada una de estas fases se renderiza de forma condicional, delegando la lógica específica a componentes especializados (ImageSelectorContainer e ImageChatContainer). De este modo, App mantiene un rol de coordinación, mientras que la lógica de selección de imágenes y gestión de la conversación se encapsula en componentes independientes. La integración con Chakra UI y un tema personalizado refuerza la coherencia visual y facilita la escalabilidad del diseño, manteniendo una separación clara entre lógica de aplicación, estructura de la interfaz y estilo visual (ver Fig.7).

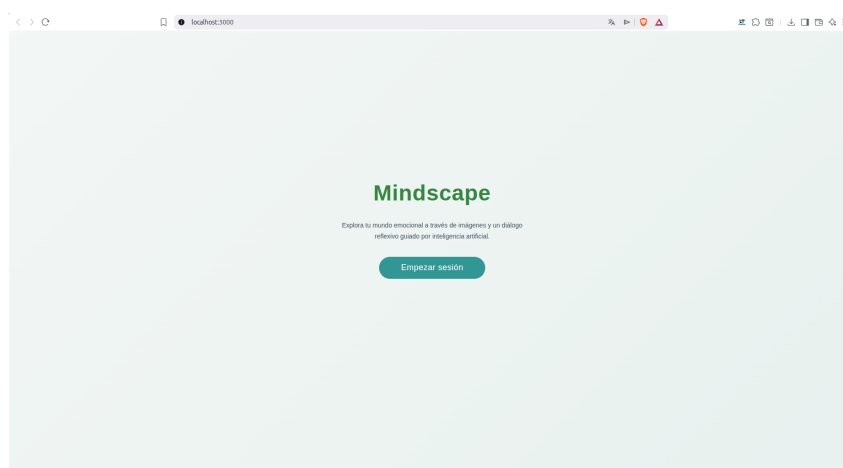


Figura 7: Home UI

5.2.2. api.js

El archivo api.js constituye la capa de acceso a datos del frontend y es un elemento clave en la separación entre la lógica de presentación y la lógica de comunicación con el backend. Su principal aportación es abstraer por completo las peticiones HTTP, proporcionando a los componentes de React una interfaz limpia y coherente para interactuar con la API REST de MINDSCAPE sin necesidad de conocer detalles de bajo nivel como rutas, cabeceras o formatos de serialización.

En primer lugar, el módulo define una instancia centralizada de Axios (librería de JavaScript utilizada para realizar peticiones HTTP de manera sencilla y manejar las respuestas de forma asíncrona) con una URL base común, lo que garantiza consistencia en todas las llamadas al backend y facilita futuras modificaciones del entorno (por ejemplo, cambios entre desarrollo y producción). A partir de esta instancia, se implementan funciones específicas alineadas con los flujos funcionales de la aplicación: el inicio de una conversación mediante el envío de una imagen utilizando multipart/form-data, la continuación del diálogo a partir del historial completo de mensajes, la comprobación del estado del servidor y la generación de resúmenes tanto en formato textual como en PDF. Cada una de estas funciones encapsula el endpoint correspondiente y normaliza la respuesta para su consumo por la interfaz.

Además, `api.js` incorpora un manejo explícito de errores, capturando fallos de red o respuestas inválidas y propagándolos de forma controlada hacia los componentes superiores. Este diseño mejora la robustez del frontend y contribuye a su mantenibilidad y escalabilidad, ya que cualquier cambio en la API o en la estrategia de comunicación puede resolverse en un único punto, sin afectar directamente a los componentes visuales ni a la lógica de interacción con el usuario.

5.2.3. ImageSelectorContainer y ImageSelectorUI

La implementación del selector de imágenes en el frontend se apoya en una separación clara entre lógica y presentación mediante los componentes `ImageSelectorContainer` y `ImageSelectorUI`, siguiendo el `Container/Presentational Pattern` [22]. Esta decisión refuerza la mantenibilidad del sistema y facilita futuras extensiones tanto a nivel funcional como visual.

El componente `ImageSelectorContainer` es el encargado de gestionar toda la lógica asociada a la selección de imágenes y a la interacción con el backend. Mantiene el estado interno relacionado con la imagen seleccionada, la vista previa, y el estado de carga durante la comunicación con la API. Además, centraliza el tratamiento de los distintos tipos de entrada posibles: imágenes predeterminadas incluidas en los assets del proyecto o imágenes subidas por el propio usuario desde su dispositivo. En este último caso, se genera una URL temporal para la previsualización, mientras que para las imágenes predeterminadas se conserva únicamente su índice de referencia este es un identificador numérico que relaciona la imagen seleccionada por el usuario con la posición de la imagen dentro de la galería predeterminada. Cuando el usuario confirma la selección, el contenedor se responsabiliza de preparar correctamente la imagen (incluida la conversión de una imagen predeterminada a objeto `File`) y de invocar el servicio `startChatWithImage`, propagando finalmente la respuesta inicial del backend al componente padre. También incorpora manejo de errores y feedback al usuario mediante el uso de toasts que son unas pequeñas notificaciones emergentes que aparecen de forma temporal para informar al usuario.

Por su parte, `ImageSelectorUI` se centra exclusivamente en la representación visual y la experiencia de usuario. Recibe todas las propiedades necesarias desde el contenedor y se limita a renderizar la interfaz de selección sin contener lógica de negocio ni dependencias directas con la API. Este componente presenta una galería interactiva de imágenes predeterminadas con la opción de subir imágenes propias con animaciones suaves basadas en `Framer Motion` (ver Fig.8). El diseño enfatiza una experiencia inmersiva y reflexiva, alineada con los objetivos de la aplicación, mientras que los distintos estados (selección, previsualización y carga) se gestionan únicamente a través de props que son datos que se pasan de un componente padre a un componente hijo para configurar su comportamiento o contenido.

En conjunto, esta arquitectura permite que la selección de imágenes sea un módulo autónomo, fácilmente reutilizable y desacoplado del resto del flujo de la aplicación. La lógica de negocio queda encapsulada en el contenedor, mientras que la interfaz puede evolucionar de forma independiente, lo que mejora la escalabilidad del frontend y la coherencia del diseño global de MINDSCAPE.

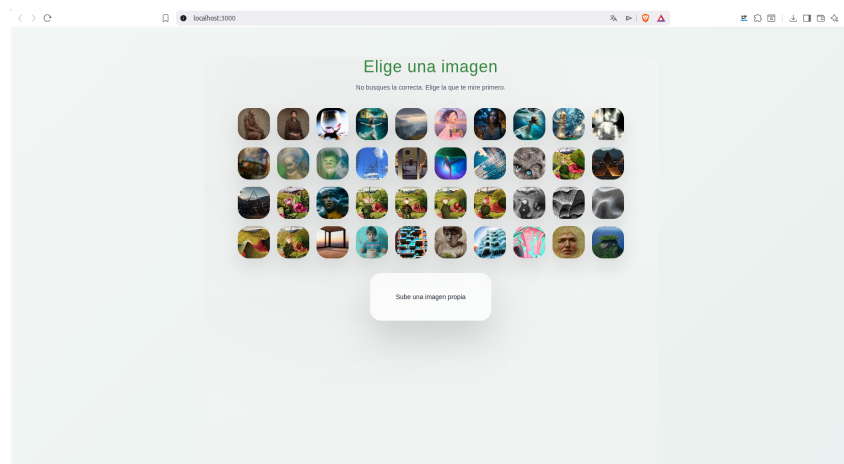


Figura 8: UI Selector de imágenes

5.2.4. ImageChatContainer y ImageChatUI

La implementación del chat en el frontend sigue el mismo principio arquitectónico aplicado en el selector de imágenes: una separación explícita entre lógica de control y presentación, materializada en los componentes `ImageChatContainer` e `ImageChatUI`. Esta división resulta clave para mantener la complejidad del flujo conversacional bajo control y facilitar la evolución futura del sistema.

`ImageChatContainer` actúa como el núcleo lógico del sistema de conversación. Es responsable de gestionar el estado completo del chat, incluyendo el historial de mensajes, la entrada del usuario, el estado de carga y el indicador de finalización de la conversación (`finished`). Durante su inicialización, inserta la primera pregunta generada por el backend en el historial, asegurando la continuidad entre la selección de la imagen y el inicio del diálogo. Este componente controla el envío de mensajes del usuario, construyendo el historial actualizado y enviándolo al backend a través del endpoint `/continue_chat`. La respuesta generada por el modelo de inteligencia artificial se integra de forma incremental en el estado del chat, respetando el valor del flag `finished` para bloquear nuevas interacciones cuando el modelo considera la conversación finalizada. Adicionalmente, `ImageChatContainer` gestiona acciones de alto nivel como el reinicio de la conversación y la descarga del resumen en formato PDF, encapsulando toda la interacción con la capa de servicios (`api.js`) y proporcionando feedback al usuario mediante notificaciones visuales.

Por su parte, `ImageChatUI` se encarga exclusivamente de la representación visual del diálogo y de la experiencia interactiva. Recibe el estado del chat y los callbacks necesarios como propiedades, sin mantener dependencias directas con el backend ni lógica de negocio. Este componente implementa una interfaz inmersiva basada en la imagen seleccionada: utiliza técnicas de desenfoque y animación para generar un fondo dinámico y extrae el color dominante de la imagen mediante `ColorThief` para adaptar visualmente los mensajes del bot (ver Fig.9). El historial de mensajes se presenta como un flujo conversacional claramente diferenciado entre usuario y sistema, con animaciones suaves que refuerzan la sensación de diálogo continuo.

Además, gestiona el área de entrada de texto con un campo autoajutable y controla la experiencia de scroll para asegurar que los mensajes más recientes estén siempre visibles.

En conjunto, esta arquitectura permite que el chat funcione como un módulo autónomo y coherente dentro del frontend de MINDSCAPE. La lógica conversacional, la comunicación con el backend y la gestión de estados complejos quedan encapsuladas en el contenedor, mientras que la interfaz puede evolucionar de forma independiente, priorizando la experiencia emocional y visual del usuario sin comprometer la claridad estructural del código.

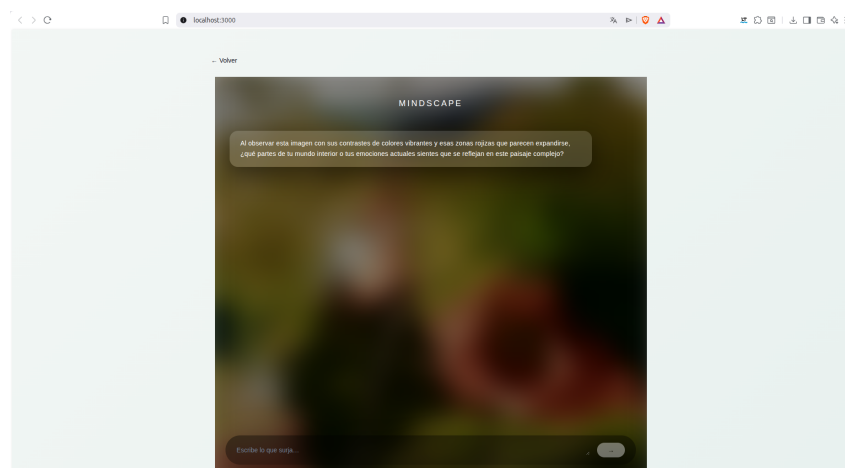


Figura 9: UI del Chat

6. Funcionamiento de los Endpoints y del Modelo

En este capítulo se profundiza en los detalles expuestos en la sección de implementación del backend, explicando el funcionamiento interno del proyecto y cómo se comunican sus componentes para proporcionar respuestas al usuario.

Como se ha mencionado anteriormente, el backend está definido por tres archivos principales: **main.py**, **controllers.py** y **models.py**. Para comprender el funcionamiento, nos centraremos en **main.py** y **models.py**. En **main.py** se definen los endpoints asociados a las diferentes acciones del modelo, mientras que **models.py** controla la generación de las respuestas.

El sistema presenta dos líneas principales de ejecución: el inicio de la conversación y la continuación de la misma y en menor medida la generación en formato PDF del resumen de la conversación.

6.1. Inicio de la Conversación

Una vez la usuaria ha seleccionado la obra sobre la que desea reflexionar, al hacer clic en el botón para iniciar la sesión, la imagen se convierte en un objeto **File**. Esta conversión proporciona una estructura común, ya que las imágenes subidas por el usuario deben transformarse a este formato, mientras que las imágenes de la galería se pueden referenciar mediante su índice, que representa su posición en la galería.

Además de la imagen, el frontend envía un booleano **isDefault** que indica si la imagen proviene de la galería (**true**) o ha sido proporcionada por el usuario (**false**). Esta información se envía al endpoint **/start_chat** de **main.py**, donde se reciben ambos parámetros y se trasladan al modelo.

En el modelo, el primer paso consiste en reducir la resolución de la imagen a un tamaño aproximado de 512×512 píxeles mediante la función **thumbnail**. De esta forma se optimiza la eficiencia del procesado sin comprometer significativamente la calidad, lo que permite al modelo extraer la información visual necesaria.

Posteriormente, en función del valor del booleano, se utilizan dos prompts diferentes para iniciar la conversación:

- Si **isDefault** es **true** (imagen de la galería): el prompt enfatiza que la imagen ha sido *escogida* por la usuaria.
- Si **isDefault** es **false** (imagen externa): el prompt destaca que la imagen ha sido *creada o aportada* por la usuaria.

En ambos casos, el prompt (ver Fig.10) solicita al modelo que genere de forma reflexiva y empática una pregunta inicial que fomente la autoexploración de la usuaria. Asimismo, se especifica que la respuesta debe seguir la estructura JSON definida: "message": "...", "finished": false.

Finalmente, se envía el prompt al modelo mediante la función `generate_content`, que permite procesar tanto el texto como la imagen. El resultado obtenido se pasa por la función `_parse_response`, que verifica y garantiza que el formato de la respuesta sea el esperado. Esta función devuelve el mensaje (la pregunta inicial) y el booleano `finished`, que se utilizará más adelante para cerrar la conversación de manera automática y natural.

```
def start_chat(self, image_file, is_default):
    # Procesar la imagen antes de enviarla al modelo
    image = Image.open(image_file)
    image.thumbnail((512, 512))

    if is_default:
        prompt = """
        Eres una IA que ayuda a reflexionar sobre emociones del usuario que
        ha escogido una imagen predeterminada para transmitir sus emociones.
        De forma reflexiva y empática genera una pregunta inicial para fomentar
        la autoexploración del usuario sobre sus sentimientos basandote en la imagen.
        Devuelve EXCLUSIVAMENTE un JSON así:
        {
            "message": "...",
            "finished": false
        }
        """
    else:
        prompt = """
        Eres una IA que ayuda a reflexionar sobre emociones del usuario que
        ha creado una imagen para transmitir sus emociones.
        De forma reflexiva y empática genera una pregunta inicial para fomentar
        la autoexploración del usuario sobre sus sentimientos basandote en la imagen.
        Devuelve EXCLUSIVAMENTE un JSON así:
        {
            "message": "...",
            "finished": false
        }
        """

    # Generar contenido basado en el prompt y la imagen
    response = self.model.generate_content([prompt, image])
    message, finished = self._parse_response(response)
    return {"message": message, "finished": finished}
```

Figura 10: Prompts de `start_chat`

6.2. Continuación de la Conversación

Una vez iniciada la sesión, la usuaria puede interactuar libremente con el modelo mediante el envío de mensajes. Cada vez que se produce un intercambio, el frontend envía al endpoint `/continue_chat` de **main.py** el historial completo de la conversación. Este historial se estructura como una lista de objetos JSON, donde cada elemento contiene los campos `sender` (que identifica si el mensaje proviene del usuario o del asistente) y `text` (el contenido del mensaje).

Al recibir esta información, el endpoint traslada el historial al modelo, especí-

ficamente a la función `continue_chat` de **models.py**. Esta función construye un prompt dinámico que incluye el contexto completo de la conversación, permitiendo al modelo mantener la coherencia y continuidad en el diálogo.

El prompt diseñado para esta fase cumple múltiples funciones:

- Instruye al modelo para que mantenga un tono empático, cercano y no juzgue.
- Permite que la conversación evolucione más allá de la imagen inicial, siguiendo las líneas de diálogo propuestas por la usuaria.
- Fomenta la autoexploración emocional de forma natural y fluida.
- Establece criterios para detectar cuándo la usuaria desea finalizar la conversación.

Un aspecto clave de este prompt (ver Fig.11) es la instrucción sobre el cierre de la conversación. El modelo está programado para identificar señales que indiquen que la usuaria quiere dar por finalizada la sesión. Cuando detecta estas señales, genera una breve reflexión final que incluye un recordatorio de que se trata de una inteligencia artificial y no un profesional de la salud mental, se despide de forma cordial y, crucialmente, modifica el valor del booleano `finished` a `true`.

Al igual que en el inicio de la conversación, el prompt especifica que la respuesta debe seguir el formato JSON establecido: `"message": "...", "finished": true|false`. La respuesta generada por el modelo se procesa nuevamente mediante la función `_parse_response`, que valida la estructura y extrae tanto el mensaje de respuesta como el estado del booleano `finished`.

Este booleano actúa como un mecanismo de control del flujo de la conversación. Mientras su valor sea `false`, la interfaz permite que la usuaria continúe enviando mensajes. Sin embargo, cuando el modelo lo establece en `true`, el frontend interpreta esta señal y finaliza la sesión, ofreciendo a la usuaria la posibilidad de descargar un resumen de la conversación mantenida.

De esta forma, el sistema logra mantener conversaciones contextualizadas y coherentes, adaptándose a las necesidades emocionales de la usuaria mientras gestiona de manera automática y natural el ciclo completo de la sesión terapéutica.

6.3. Modelo de IA Gemini 2.5 Flash

En MINDSCAPE se ha utilizado el modelo de lenguaje Gemini 2.5 Flash, desarrollado por Google. Se trata de un modelo de inteligencia artificial diseñado para

```
def continue_chat(self, history):
    # Continuar la conversación basándose en el historial
    prompt = f"""
    Aquí está el historial de la conversación:
    {history}

    Eres una IA que ayuda a reflexionar sobre emociones del usuario que
    ha creado una imagen para transmitir sus emociones. Continúa con la
    conversación de forma empática, cercana y sin juzgar ayudando al usuario a
    fomentar la autoexploración si el usuario propone una línea de diálogo siguela
    no te centres únicamente en la imagen. Si el usuario parece querer dar la
    conversación por finalizada, haz una breve reflexión con un disclaimer de
    que eres una IA y no un profesional, despidete y marca finished como true.
    Devuelve EXCLUSIVAMENTE un JSON así:
    """
    """
    """
    response = self.model.generate_content(prompt)
    message, finished = self._parse_response(response)
    return {"message": message, "finished": finished}
```

Figura 11: Prompt de continue_chat

generar texto de forma automática y responder a preguntas de manera coherente y contextualizada. Gemini 2.5 Flash destaca por ofrecer buenas capacidades de razonamiento y comprensión con un menor coste y tiempo de respuesta que otros modelos más grandes, lo que lo hace adecuado para aplicaciones prácticas. El modelo ha sido entrenado con una gran cantidad de datos procedentes de textos públicos, código, imágenes, audio y vídeo, con información actualizada hasta principios de 2025.

A diferencia de modelos más simples, Gemini 2.5 Flash puede dedicar un breve proceso interno de razonamiento antes de generar una respuesta cuando la tarea es compleja. Esto significa que no siempre responde de forma inmediata, sino que puede “pensar” durante unos pasos intermedios para mejorar la calidad y corrección de la respuesta final. Este comportamiento es especialmente útil en problemas que requieren varios pasos, como explicaciones técnicas o razonamientos lógicos, y permite obtener resultados más consistentes.

Además, el modelo incorpora mecanismos de seguridad y filtrado de contenido. Estos filtros están diseñados para evitar respuestas ofensivas, dañinas o inapropiadas, como discursos de odio, incitación a la violencia o información peligrosa. En caso de que una pregunta entre en conflicto con estas normas, el modelo puede rechazar la respuesta o reformularla de manera segura. De este modo, Gemini 2.5 Flash busca mantener un equilibrio entre ser útil y respetar criterios básicos de seguridad y responsabilidad.[23]

7. Pruebas Realizadas

7.1. Tests de usuario

Con el objetivo de evaluar el correcto funcionamiento del sistema desde la perspectiva del usuario final, se han diseñado y propuesto distintos tests de usuario centrados tanto en la usabilidad de la interfaz como en la calidad de la interacción conversacional. Estas pruebas permiten analizar aspectos funcionales, emocionales y comunicativos de la aplicación, así como validar que el uso de imágenes y diálogo guiado por inteligencia artificial cumple con los objetivos planteados en el proyecto.

Test de usabilidad (Usability Testing)

- **Objetivo:** Evaluar la facilidad de uso del sistema y la comprensión de la interfaz.
- **Metodología:** Interacción libre con la aplicación mediante tareas guiadas (selección de imagen, inicio del chat, respuesta a preguntas y cierre de sesión).
- **Métricas:**
 - Tiempo de ejecución de cada tarea.
 - Número de errores o dudas.
 - Necesidad de ayuda externa.
 - Valoración de facilidad de uso (escala Likert 1–5).
- **Resultado esperado:** Detección de problemas de navegación, claridad visual y carga cognitiva.

Test de experiencia de usuario (UX Test)

- **Objetivo:** Analizar la percepción emocional y la satisfacción global del usuario.
- **Metodología:** Cuestionario posterior a una sesión completa de chat.
- **Métricas:**
 - Fluidez percibida de la conversación.
 - Comodidad durante la interacción.
 - Valoración estética y claridad del diseño.
 - Intención de reutilizar la aplicación.
- **Resultado esperado:** Medición de la aceptación general del sistema y su potencial de adopción.

Test de coherencia conversacional

- **Objetivo:** Evaluar la coherencia, continuidad y mantenimiento del contexto.
- **Metodología:** Conversaciones completas con cambios de tema y respuestas ambiguas.
- **Métricas:**
 - Coherencia entre intervenciones consecutivas.
 - Capacidad de recordar información previa.
 - Respuestas fuera de contexto.
- **Resultado esperado:** Validación de la correcta gestión del historial conversacional.

Test de empatía percibida

- **Objetivo:** Medir el grado de empatía y adecuación emocional del chatbot.
- **Metodología:** Evaluación emocional posterior a la interacción.
- **Métricas:**
 - Sensación de comprensión.
 - Adecuación del tono del lenguaje.
 - Percepción de apoyo emocional.
- **Resultado esperado:** Verificación del cumplimiento de principios básicos de interacción terapéutica asistida.

Test de comprensión de las preguntas generadas

- **Objetivo:** Evaluar la claridad y comprensibilidad de las preguntas generadas por el modelo.
- **Metodología:** Valoración de las preguntas tras la sesión.
- **Métricas:**
 - Claridad semántica.
 - Ambigüedad percibida.
 - Dificultad para responder.
- **Resultado esperado:** Identificación de preguntas confusas o excesivamente complejas.

Test de flujo conversacional y cierre

- **Objetivo:** Analizar la adecuación del cierre conversacional.
- **Metodología:** Evaluación del momento en que el sistema marca el fin de la conversación (*finished = true*).
- **Métricas:**
 - Naturalidad del cierre.
 - Sensación de conversación completa.
 - Adecuación temporal del final.
- **Resultado esperado:** Validación del mecanismo automático de finalización del diálogo.

7.2. Resultados de las pruebas realizadas

El proyecto fue evaluado por un total de seis participantes, con una distribución equilibrada de género, siendo la mitad hombres y la otra mitad mujeres, con el objetivo de analizar el funcionamiento del sistema a través de los distintos tests de usuario definidos previamente. En cuanto a la edad, el 50 % de los participantes tiene más de 50 años, mientras que un 33,3 % se sitúa en el rango de 20 a 30 años y el resto corresponde al intervalo de 31 a 40 años. Respecto a la experiencia previa en el ámbito terapéutico, un 66,7 % de los usuarios indicó no tener ningún tipo de experiencia con la terapia o la arteterapia. No obstante, todos los participantes declararon tener experiencia previa en el uso de la tecnología, lo que garantizó una interacción fluida con la aplicación.

Durante las sesiones de prueba, los participantes llevaron a cabo un conjunto de tareas representativas del uso real de la aplicación. Estas incluyeron la selección y el cambio de imágenes de la galería integrada, así como la carga de imágenes propias para iniciar la conversación. Todas las personas usuarias mantuvieron diálogos con el modelo de una duración aproximada de siete minutos, tras los cuales se les pidió finalizar la conversación de forma natural. Finalmente, descargaron el resumen de la sesión en formato PDF y reiniciaron el sistema para comenzar una nueva interacción.

En términos de usabilidad, el sistema fue valorado de forma positiva, obteniendo una puntuación media de 3,66 sobre 5 (ver Fig.12). Ninguno de los usuarios requirió ayuda externa para completar las tareas propuestas y se observó una interacción fluida y natural con la interfaz, lo que indica que la navegación y la organización visual resultan intuitivas y accesibles. Los comentarios cualitativos reflejan que las interfaces fueron percibidas mayoritariamente como simples, intuitivas y fáciles de utilizar, aunque algunos usuarios señalaron que ciertos elementos visuales podían resultar poco convencionales o excesivamente simples en determinados puntos de la aplicación.

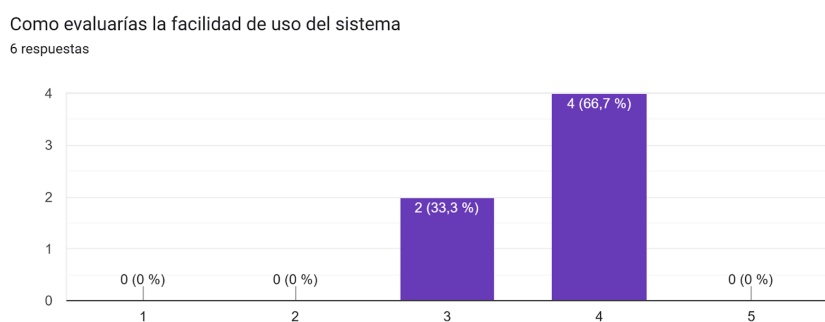


Figura 12: Gráfico de facilidad de uso

Respecto a la experiencia de usuario, las sensaciones generales tras el uso del sistema fueron favorables. Varias personas destacaron el diseño visual, describiéndolo como agradable, tranquilo o generador de sensaciones positivas como paz o seguridad. Asimismo, se valoró la facilidad para mantener una conversación continuada con el sistema. No obstante, este aspecto se analiza con mayor profundidad en el capítulo de conclusiones, dado que está fuertemente influido por percepciones subjetivas y por la predisposición individual hacia el uso de este tipo de herramientas.

En relación con la coherencia conversacional y la calidad de las preguntas generadas por el modelo, los resultados muestran una valoración media situada entre 4 sobre 5 en coherencia (ver Fig.13) y 3,66 sobre 5 en la calidad de las preguntas. De forma general, las respuestas fueron percibidas como coherentes, comprensibles y alineadas con el contexto de la conversación. Sin embargo, varios participantes señalaron que el diálogo podía resultar en ocasiones demasiado automatizado o poco flexible, y que algunas preguntas tendían a centrarse en exceso en la imagen inicial, dejando en segundo plano una exploración más profunda de los sentimientos del usuario.

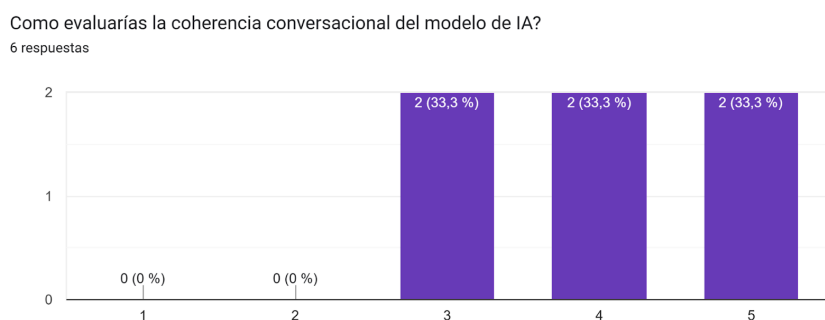


Figura 13: Gráfico de coherencia conversacional

El apartado relativo a la empatía percibida presenta resultados matizados. Aunque los usuarios destacaron sensaciones como seguridad, reafirmación y un tono adecuado por parte del modelo, también se repite la percepción de cierta falta de cercanía o humanidad en las respuestas. En particular, algunos participantes indicaron que el sistema reflexiona poco de forma espontánea sobre las emociones del usuario y que no suele ofrecer conclusiones emocionales si estas no son solicitadas explícitamente. Este aspecto fue identificado de manera consistente como una posible línea de mejora para reforzar el componente emocional y terapéutico de la experiencia.

Por último, el sistema de finalización automática de la conversación fue valorado de forma muy positiva. Los participantes destacaron la capacidad del modelo para detectar de manera adecuada el momento de cierre del diálogo, incluso sin una indicación explícita por parte del usuario. Se valoró especialmente el tono amigable de la despedida y el reconocimiento de la interacción compartida, aunque uno de los participantes sugirió que la inclusión de un botón explícito para finalizar la conversación podría aportar una mayor sensación de control al usuario.

8. Conclusiones y mejoras a futuro

8.1. Conclusiones

El presente Trabajo de Fin de Grado ha abordado el diseño e implementación de Mindscape, una aplicación web experimental orientada al bienestar mental que combina el uso de imágenes con la interacción conversacional mediante modelos de inteligencia artificial. El objetivo principal ha sido explorar el potencial de los sistemas multimodales como herramienta de apoyo a la autoexploración emocional, poniendo el foco en la experiencia del usuario, la coherencia conversacional y la integración ética de la inteligencia artificial.

A lo largo del desarrollo del proyecto se han adquirido y consolidado conocimientos prácticos en diversas tecnologías clave del desarrollo web moderno y de la integración de sistemas de inteligencia artificial. En particular, con *React* para la construcción de interfaces interactivas basadas en componentes, así como con FastAPI para el diseño de servicios backend, abordando aspectos como la gestión de peticiones, el manejo de estados y la estructuración modular del código. Asimismo, el proyecto ha permitido profundizar en la integración de modelos de inteligencia artificial a través de Google AI Studio, comprendiendo el flujo de comunicación entre frontend, backend y modelo, así como la gestión del ciclo completo de una interacción conversacional.

Si bien no todos los objetivos específicos definidos inicialmente han podido cumplirse en su totalidad o con el nivel de profundidad deseado, el resultado global del proyecto puede considerarse satisfactorio y coherente con los objetivos generales planteados. Las funcionalidades principales del sistema han sido implementadas con éxito, permitiendo validar la viabilidad de la propuesta y demostrar el potencial de la combinación entre interacción artística e inteligencia artificial como herramienta de autoexploración y reflexión personal. En este sentido, el desarrollo de MINDSCAPE ha supuesto un resultado positivo tanto a nivel técnico como conceptual, así como el aprendizaje adquirido a lo largo del proceso.

Desde el punto de vista funcional, el sistema ha demostrado ser estable y usable, permitiendo a los usuarios completar las tareas sin necesidad de asistencia externa. Las pruebas de usuario reflejan una valoración positiva de la usabilidad y de la experiencia general, destacando la claridad de la interfaz, la fluidez conversacional y la correcta gestión del cierre de la conversación. En cuanto al modelo de inteligencia artificial, este ha mostrado capacidad para mantener un diálogo coherente y contextualizado a partir de una imagen inicial, aunque se han identificado limitaciones en la profundidad emocional y la cercanía percibida, propias de los retos actuales de los sistemas generativos en contextos de acompañamiento emocional.

Por otro lado, conviene destacar que, aunque el proyecto está concebido como una herramienta potencialmente útil para un público amplio, su contribución real y el beneficio que puede aportar dependen en gran medida de la predisposición del usuario hacia el sistema y hacia el propio concepto de la arteterapia. En este sentido, la implicación personal, la apertura emocional y la actitud con la que el usuario afronta la interacción condicionan de forma significativa la experiencia y

los resultados obtenidos. Aunque los resultados de las pruebas de usuario han sido de gran utilidad para evaluar el funcionamiento general del sistema, es razonable considerar que su realización en otros contextos y con perfiles de usuarios más predispuestos al uso de la plataforma podría haber proporcionado conclusiones más profundas y representativas.

Finalmente, destacar que aunque el sistema no pretende sustituir ningún tipo de intervención profesional, los resultados obtenidos sugieren que este tipo de herramientas pueden funcionar como un recurso complementario para la reflexión personal y la expresión emocional. En este sentido, hay que poner de relieve la importancia de definir límites claros, tanto técnicos como éticos, en el uso de la inteligencia artificial aplicada a ámbitos sensibles como el bienestar mental.

8.2. Mejoras a futuro

Desde el punto de vista del desarrollo del proyecto, se identifican múltiples líneas de mejora que podrían explorarse en trabajos futuros. Este Trabajo de Fin de Grado ha tenido un carácter principalmente experimental y exploratorio; no obstante, con un mayor margen temporal, sería posible ampliar y enriquecer significativamente el sistema.

En primer lugar, en relación con la experiencia de usuario, una mejora relevante sería la incorporación de un mayor grado de personalización. Esto incluiría la creación de perfiles de usuario, el registro de imágenes personales, el almacenamiento del historial de conversaciones, incluir la posibilidad de retomar conversaciones pasadas, así como la posibilidad de generar o gestionar imágenes dentro de la propia plataforma. Este tipo de funcionalidades contribuiría a aumentar la comodidad del usuario, reforzar el vínculo con la aplicación y fomentar un uso más continuado del sistema.

En segundo lugar, se contempla la mejora del propio modelo de inteligencia artificial. El comportamiento del sistema podría refinarse mediante un ajuste más avanzado de los prompts y de los flujos conversacionales, así como mediante la colaboración con profesionales del ámbito terapéutico para definir mejores estrategias de interacción. Asimismo, el uso de ejemplos reales y escenarios controlados permitiría enriquecer las respuestas del modelo y mejorar la percepción de coherencia, empatía y profundidad por parte de los usuarios.

Finalmente, de cara a una posible evolución del proyecto, se plantea el despliegue de la plataforma en la nube, lo que permitiría el acceso remoto a través de internet sin necesidad de ejecutar el código de forma local. Este enfoque facilitaría la incorporación de bases de datos para la gestión de usuarios y conversaciones, así como la implementación de mecanismos avanzados de seguridad, cifrado y protección de la privacidad, aspectos fundamentales en aplicaciones que tratan información de carácter sensible.

Índice de figuras

1.	Diagrama de Gantt del proyecto	9
2.	React y Chakra UI	15
3.	FastAPI	16
4.	Gemini 2.5	16
5.	Estructura Proyecto	17
6.	Workflow Backend	19
7.	Home UI	25
8.	UI Selector de imagenes	27
9.	UI del Chat	28
10.	Prompts de start_chat	30
11.	Prompt de continue_chat	32
12.	Gráfico de facilidad de uso	36
13.	Gráfico de coherencia conversacional	36

Índice de cuadros

1.	Comparativa de proyectos existentes y diferenciación de MINDSCAPE	3
----	---	---

Referencias

- [1] Han Li, Renwen Zhang, Yi-Chieh Lee, Robert E. Kraut & David C. Mohr : *Systematic review and meta-analysis of AI-based conversational agents for promoting mental health and well-being*, npj Digital Medicine, 19 December 2023
<https://www.nature.com/articles/s41746-023-00979-5#Abs1>
- [2] World Health Organization: *World mental health report: Transforming mental health for all*, World Health Organization, 2022.
<https://www.who.int/publications/i/item/9789240049338>
- [3] Pifalo, T.: *Art therapy and trauma: Theoretical and practical perspectives*, PMCID: PMC9472646, 2021.
<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9472646/>
- [4] OpenAI: AI Agents: Autonomous Decision-Making Systems, *OpenAI Technical Papers*, març de 2024.
<https://cdn.openai.com/papers/gpt-4.pdf>

- [5] Meta Platforms, Inc.: *React: A JavaScript library for building user interfaces*, 2024.
<https://react.dev>
- [6] Vue.js: *The Progressive JavaScript Framework*, 2024.
<https://vuejs.org>
- [7] Angular Team: *Angular – One framework. Mobile & desktop*, 2024.
<https://angular.io>
- [8] Chakra UI: *A simple, modular and accessible component library for React*, 2024.
<https://chakra-ui.com>
- [9] Material UI: *React components for faster and easier web development*, 2024.
<https://mui.com>
- [10] Tailwind UI: *Beautiful UI components, designed with Tailwind CSS*, 2024.
<https://tailwindui.com>
- [11] Python Software Foundation: *Python Programming Language*, 2024.
<https://www.python.org>
- [12] Express.js: *Fast, unopinionated, minimalist web framework for Node.js*, 2024.
<https://expressjs.com>
- [13] Django Software Foundation: *The Web framework for perfectionists with deadlines*, 2024.
<https://www.djangoproject.com>
- [14] Flask: *A lightweight WSGI web application framework in Python*, 2024.
<https://flask.palletsprojects.com>
- [15] Ramírez, S.: *FastAPI: High performance web framework for building APIs with Python*, 2024.
<https://fastapi.tiangolo.com>
- [16] Google LLC: *Google AI Studio*, 2024.
<https://aistudio.google.com>
- [17] OpenAI: *OpenAI API Documentation*, 2024.
<https://platform.openai.com/docs>
- [18] Swagger (SmartBear Software): *Swagger: API design and documentation tools*, 2024.
<https://swagger.io>
- [19] Woebot Health: *Woebot: Your mental health ally*, 2023.
<https://woebothealth.com>
- [20] Wysa Ltd.: *Wysa: AI chatbot for mental health*, 2023.
<https://www.wysa.com>

- [21] Replika Inc.: *Replika: The AI companion*, 2023.
<https://replika.com>
- [22] *Container/Presentational Pattern*, 2026.
<https://www.patterns.dev/react/presentational-container-pattern/>
- [23] *Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities*, 2025.
https://storage.googleapis.com/deepmind-media/gemini/gemini_v2_5_report.pdf

A. Manual de ejecución y uso del sistema

Este anexo describe el procedimiento necesario para instalar, ejecutar y utilizar el prototipo funcional desarrollado en el marco de este Trabajo de Fin de Grado. El sistema corresponde a una aplicación web basada en una arquitectura cliente–servidor que integra modelos de inteligencia artificial multimodal para la interacción imagen–conversación.

Requisitos del sistema

Para la correcta ejecución del proyecto es necesario disponer de los siguientes componentes:

- Python 3.8 o superior
- Node.js 14 o superior
- npm 6 o superior
- Git y una terminal compatible (Git Bash en Windows o terminal Unix en Linux/Mac)

Dependencias del backend

El backend está desarrollado en Python utilizando el framework FastAPI. Las principales dependencias del sistema son:

- fastapi
- uvicorn
- python-multipart
- google-generativeai
- python-dotenv
- Pillow
- reportlab

Todas ellas se encuentran especificadas en el archivo `requirements.txt`.

Dependencias del frontend

El frontend está desarrollado con React y utiliza las siguientes librerías principales:

- React
- Chakra UI
- Axios
- Framer Motion
- React Icons

Arquitectura del proyecto

El proyecto se estructura en dos bloques claramente diferenciados: backend y frontend, siguiendo un enfoque modular y desacoplado.

```
chatbot_mental/  
  backend/  
    main.py  
    controllers.py  
    models.py  
    requirements.txt  
    .env  
    start_server.sh  
  
  frontend/  
    src/  
      App.jsx  
      components/  
      services/  
    public/  
    package.json  
    start_frontend.sh
```

Instalación del proyecto

Clonado del repositorio

```
git clone <url-del-repositorio>  
cd chatbot_mental
```

Instalación del backend

Se recomienda el uso de un entorno virtual para aislar las dependencias del proyecto.

```
cd backend
python -m venv .venv
```

Activación del entorno virtual:

- Windows (Git Bash):

```
source .venv/Scripts/activate
```

- Linux/Mac:

```
source .venv/bin/activate
```

Instalación de dependencias:

```
pip install -r requirements.txt
```

Instalación manual de dependencias:

En caso de que la instalación automática mediante el archivo `requirements.txt` no se complete correctamente, es posible instalar las dependencias del backend de forma manual ejecutando los siguientes comandos:

```
pip install fastapi==0.104.1
pip install uvicorn==0.24.0
pip install python-multipart==0.0.6
pip install google-generativeai==0.3.2
pip install python-dotenv==1.0.0
pip install Pillow==10.0.1
pip install reportlab
```

Esta instalación manual permite asegurar que todas las librerías necesarias para el correcto funcionamiento del servidor backend y la integración con los modelos de inteligencia artificial estén disponibles, incluso en entornos donde la resolución automática de dependencias pueda presentar conflictos.

Instalación del frontend

```
cd ../frontend
npm install
```

Configuración del sistema

Configuración del backend

Es necesario crear un archivo `.env` en el directorio `backend/` con la siguientes variable:

```
GOOGLE_API_KEY=tu_api_key_aquí
```

```
PORT=puerto
```

```
HOST=localhost
```

La clave de API se obtiene a través de Google AI Studio.

Configuración del frontend

Por defecto, el frontend se conecta al backend en `http://localhost:8000`. Esta configuración puede modificarse en el archivo:

```
frontend/src/services/api.js
```

Ejecución del sistema

Se recomienda utilizar dos terminales abiertas simultáneamente.

Ejecución mediante scripts

Terminal 1 – Backend:

```
cd backend  
source .venv/Scripts/activate  
./start_server.sh
```

El backend quedará accesible en `http://localhost:8000`.

Terminal 2 – Frontend:

```
cd frontend  
./start_frontend.sh
```

La aplicación web estará disponible en `http://localhost:3000`.

Ejecución manual alternativa

Backend:

```
python -m uvicorn main:app --reload --host 0.0.0.0 --port 8000
```

Frontend:

```
npm start
```

Uso de la aplicación

El flujo de uso del sistema es el siguiente:

1. Acceder a la aplicación desde un navegador web.
2. Seleccionar una imagen inicial.
3. Iniciar la conversación guiada por el sistema.
4. Responder a las preguntas generadas por el modelo de inteligencia artificial.
5. Finalizar la sesión cuando el sistema indica el cierre automático de la conversación.
6. Opcionalmente, descargar el resumen de la sesión en formato PDF.

Verificación del funcionamiento

Para comprobar el correcto funcionamiento del sistema:

- Backend: acceder a `http://localhost:8000/docs` para consultar la documentación automática de la API.
- Frontend: acceder a `http://localhost:3000` para interactuar con la interfaz de usuario.

Consideraciones finales

El sistema se ejecuta en modo de desarrollo con recarga automática ante cambios en el código. Un despliegue en entorno de producción requeriría ajustes adicionales que quedan fuera del alcance de este Trabajo de Fin de Grado.

B. Preguntas realizadas en las pruebas con usuarios

Durante la fase de evaluación del sistema se llevó a cabo un test de usuarios estructurado en distintos bloques, con el objetivo de recoger información socio-demográfica, así como valoraciones cuantitativas y cualitativas sobre el uso de la plataforma MINDSCAPE. A continuación, se detallan las preguntas planteadas:

Consentimiento y aspectos éticos

- Confirmación de la lectura y comprensión de la información del estudio.
- Aceptación del consentimiento informado para participar en la investigación.
- Reconocimiento del carácter experimental del sistema y de que no sustituye atención psicológica o terapéutica profesional.

Datos generales

- Nombre
- Género
- Rango de edad
- Experiencia previa con la tecnología
- Experiencia previa en terapia o arteterapia

Evaluación del sistema

- ¿Cómo evaluarías la facilidad de uso del sistema?, escala 1–5 (Muy difícil – Muy fácil).
- ¿Cómo evaluarías las interfaces vistas? (respuesta abierta)
- ¿Cómo evaluarías la coherencia conversacional del modelo de IA?, escala 1–5 (Poco coherente – Muy Coherente)
- ¿Qué sensaciones te han transmitido las respuestas del modelo de IA? (respuesta abierta).
- ¿Cómo te han parecido las preguntas formuladas por el modelo de IA?, escala 1–5. (Poco claras y difíciles de comprender – Muy claras y fáciles de responder)
- ¿Cómo evaluarías el sistema de finalización de la conversación? (respuesta abierta)
- ¿Qué cambios o mejoras añadirías al sistema? (respuesta abierta)

Este conjunto de preguntas permitió obtener una visión global de la usabilidad del sistema, la experiencia conversacional y la percepción subjetiva del comportamiento del modelo de inteligencia artificial por parte de los participantes.