

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA



TRABAJO DE FIN DE GRADO

iaPWeb. Análisis de las Inteligencias Artificiales
generativas de código para Programación Web

Titulación: Grado en Ingeniería en Tecnologías de la Telecomunicación
Mención: Telemática
Autor: ALEXANDER CASTILLA CAMPENY
Tutor: MIGUEL ÁNGEL QUINTANA SUÁREZ
Fecha: Junio 2025

Resumen

Este trabajo aborda un estudio exploratorio y evaluativo sobre la aplicación de modelos de Inteligencia Artificial Generativa (IAG) para la generación automática de código, aplicándolos específicamente al desarrollo de una aplicación web en el entorno de la asignatura Programación Web del Grado en Ingeniería en Tecnologías de la Telecomunicación. Dentro de esta asignatura se utilizan tecnologías fundamentales para el desarrollo de aplicaciones web, como son: HTML, CSS, PHP, SQL y JavaScript. El objetivo principal es analizar y evaluar modelos de IA para la generación automática de código, buscando no solo construir una aplicación funcional, sino también explorar cómo los modelos de IA pueden optimizar el flujo de trabajo de los estudiantes y fomentar el aprendizaje práctico de técnicas modernas de desarrollo web. Se investigan, evalúan y documentan las capacidades de diferentes modelos de IA, identificando conductas y mejores prácticas en el diseño de prompts para maximizar su efectividad en este contexto, sin perder de vista el proceso enseñanza-aprendizaje.

Los resultados, del análisis y evaluación de los modelos de IA, mostraron contrastes en su rendimiento. ChatGPT o3-mini-high destacó como la opción más confiable al completar con éxito todas las fases analizadas, aunque Claude y Qwen resultaron ser alternativas útiles. En términos de velocidad y consistencia, Claude fue el más rápido, aunque presentó una alta tasa de errores, mientras que CGPTo3mh ofreció un mejor equilibrio. Se identificaron áreas comunes donde los modelos presentaron dificultades para la implementación de funcionalidades JavaScript, seguridad, consultas SQL y optimización front-end. Un aspecto central del trabajo fue la identificación de estrategias efectivas para la elaboración de prompts, como la estructura jerárquica, la contextualización, la especificidad técnica, la separación de responsabilidades y el enfoque en buenas prácticas.

Este trabajo también estudia el impacto del uso de la IA en el proceso educativo de la asignatura, señalando cómo puede afectar los objetivos docentes, resultados de aprendizaje y competencias del alumno. También se plantean diferentes escenarios para la inclusión de la IA en la metodología docente, tanto desde un uso limitado como apoyo, hasta una integración completa que requeriría redefinir los objetivos para formar a los estudiantes en la gestión y supervisión de herramientas de IA.

Abstract

This paper addresses an exploratory and evaluative study on the application of Generative Artificial Intelligence (GAI) models for automatic code generation, specifically applying them to the development of a web application within the Web Programming course of the Bachelor's Degree in Telecommunications Technology Engineering. This course utilizes fundamental technologies for web application development, such as HTML, CSS, PHP, SQL, and JavaScript. The main objective is to analyze and evaluate AI models for automatic code generation, seeking not only to build a functional application but also to explore how AI models can optimize students' workflow and foster practical learning of modern web development techniques. The capabilities of different AI models are investigated, evaluated, and documented, identifying behaviors and best practices in prompt design to maximize their effectiveness in this context, while keeping in mind the teaching-learning process.

The results of the analysis and evaluation of the AI models showed contrasting performance. ChatGPT o3-mini-high emerged as the most reliable option, successfully completing all the phases analyzed, although Claude and Qwen proved to be useful alternatives. In terms of speed and consistency, Claude was the fastest, although it had a high error rate, while CGPTo3mh offered a better balance. Common areas where the models presented difficulties in implementing JavaScript functionality, security, SQL queries, and front-end optimization were identified. A central aspect of the work was the identification of effective strategies for creating prompts, such as hierarchical structure, contextualization, technical specificity, separation of responsibilities, and a focus on best practices.

This work also studies the impact of the use of AI on the educational process of the subject, highlighting how it can affect teaching objectives, learning outcomes, and student competencies. Different scenarios for the inclusion of AI in teaching methodology are also proposed, ranging from limited use as support to full integration, which would require redefining objectives to train students in the management and supervision of AI tools.

Tabla de contenido

Resumen.....	1
Abstract.....	2
Tabla de contenido.....	1
Índice de Tablas.....	3
Índice de Figuras.....	4
Capítulo 1. Introducción y Objetivos.....	5
1.1 Introducción.....	5
1.2 Antecedentes.....	6
1.3 Objetivos.....	8
1.4 Estructura del documento.....	9
1.5. Uso de IA Generativas.....	10
Capítulo 2. Tecnologías asociadas al uso de la IA.....	11
2.1 Contextualización.....	11
2.1.1 Inteligencia Artificial.....	11
2.1.2 Inteligencia Artificial Generativas.....	12
2.1.3 Características de las IA.....	13
2.2 Desarrollos más importantes en IA generativa.....	16
2.2.1 OpenAI.....	16
2.2.2 DeepSeek.....	17
2.2.3 Microsoft.....	18
2.2.4 Anthropic.....	19
2.2.5 Mistral.....	19
2.2.6 Meta.....	20
2.2.7 Google.....	21
2.3 Programación asistida por Inteligencia Artificial.....	22
2.3.1 IA conversacional.....	23
2.3.2 IDEs con integración de IA.....	25
2.4 Análisis de modelos de IA.....	25
2.4.1 Características generales.....	26
2.4.2 Razonamiento.....	26
2.4.3 Coste de uso.....	28
2.4.4 Métricas de desempeño.....	29
Capítulo 3. Modelos de IA OpenSource.....	31
3.1 Definiendo la IA de Código Abierto.....	31
3.2 La Evolución Histórica de la IA de Código Abierto.....	32
3.3 Tipos de Modelos de IA de Código Abierto.....	33
3.3.1 Modelos de IA Generativa.....	33
3.3.2 Modelos de Embebido (Embeddings).....	33
3.3.3 Modelos de Generación de Vídeo e Imagen.....	34
3.3.4 Modelos de Generación de Código.....	35
3.4 Análisis de modelos.....	36
3.4.1 Características generales.....	36

3.4.2 Razonamiento.....	37
3.4.3 Relación Precio.....	38
3.4.4 Programación.....	39
3.5 Entornos de ejecución.....	41
3.5.1 Requisitos de Hardware.....	41
3.5.2 Requisitos de Software.....	42
3.5.3 Plataformas para la Ejecución Local de IA.....	43
Capítulo 4. Evaluación Inicial de los Modelos.....	45
4.1 Modelos Elegidos y Plataforma de Ejecución.....	45
4.1.1 Descripción de los Modelos iniciales seleccionados.....	46
4.2 Diseño y resultados de las Pruebas Iniciales.....	50
4.2.1 Diseño de las Pruebas Iniciales.....	51
4.2.2 Resultados de las Pruebas Iniciales.....	52
4.3 Métricas análisis de los Modelos.....	56
4.3.1 Métricas aplicadas a los modelos.....	57
4.3.2 Métricas aplicadas a los prompts.....	59
4.4 Requisitos del ABP de la asignatura en prompts.....	59
4.4.1 El uso de la IA en la asignatura.....	60
Capítulo 5. Análisis y Resultados.....	62
5.1 Fases y Tareas logradas con los Modelos.....	62
5.2 Resultados de los modelos.....	63
5.3 Resultados de la ejecución de los Prompts.....	66
5.3.1 Estrategias identificadas.....	68
5.3.2 Alineación con el proceso docente de la asignatura de Programación Web.....	72
5.3.3 Inclusión de la generación de prompts en la asignatura.....	73
5.3.4 Escenarios de Inclusión de IA.....	77
Capítulo 6. Conclusiones y Líneas futuras.....	80
6.1 Conclusiones que afectan a la acción docente.....	80
6.2 Conclusiones.....	80
6.3 Líneas futuras.....	82
Bibliografía.....	83
Presupuesto.....	88
Objetivos de Desarrollo Sostenible.....	92
ANEXOS.....	1
A1 Guía de instalación del entorno Ollama-OpenWebUI.....	2
A1.1 Especificaciones de Hardware Recomendadas.....	2
A1.2 Proceso de Instalación (Ollama, Open WebUI).....	2
1.2.1 Instalación de Ollama.....	2
1.2.2 Instalación de Open WebUI.....	4
A1.3 Configuración del Entorno de Ollama.....	7
A2. Código generado en las pruebas iniciales.....	10
A3. Especificaciones del CMS.....	13
A4. Repositorios del código generado y prompts evaluados.....	19
A5. Prompts e IAs usadas para la redacción normativa ULPGC.....	24

Índice de Tablas

Tabla 1. Categorización general	14
Tabla 2. Categorización de Hintze	14
Tabla 3. Otros tipos de IA	15
Tabla 4. Diversos modelos de IA generativa	33
Tabla 5. Diversos modelos de IA embebida	34
Tabla 6. Diversos modelos de IA generación de video e imágenes	35
Tabla 7. Diversos modelos de IA generación de código	36
Tabla 8. Requerimientos de hardware para modelos locales [82]	41
Tabla 9. Modelos iniciales seleccionados	46
Tabla 10. Plan de Implementación	57
Tabla 11. Sistema de evaluación de prompts	59
Tabla 12. Fases implementadas con Claude 3.7 Sonnet	60
Tabla 13. Fases y subfases donde se empleo los modelos	62
Tabla 14. Recuento de fases y subfases no alcanzadas	62
Tabla 15. Estadísticas de los modelos	63
Tabla 16. Información de los prompts empleados con Phi-4	66
Tabla 17. Información de los prompts empleados con Qwen 2.5-coder	67
Tabla 18. Información de los prompts empleados con DeepSeek-R1	67
Tabla 19. Información de los prompts empleados con Chat GPT o3-mini-high	67
Tabla 20. Diagrama DAFO – Inclusión de IA	73
Tabla 21. Tabla de amortización de los recursos hardware	89
Tabla 22. Valores del factor de corrección en función a las horas trabajadas	90
Tabla 23. Presupuesto del trabajo tarifado y amortización de los recursos materiales	91
Tabla 24. Aplicación de impuestos a los costes	91
Tabla 25. Objetivos de desarrollo sostenible	92
Tabla 26. Requerimientos hardware para Ollama	A2
Tabla 27. Comparativa de Prueba 1 de Exploración Creativa	A10
Tabla 28. Comparativa de Prueba 2 de Exploración Creativa	A11
Tabla 29. Comparativa de Prueba 3 de Resolución de Objetivos	A11
Tabla 30. Comparativa de Prueba 4 de Resolución de Objetivos	A12
Tabla 31. Resumen de Tecnologías y su Propósito en el Desarrollo Web	A13
Tabla 32. Funcionalidades y Roles de Usuario	A15
Tabla 33. Bloques Funcionales	A18

Índice de Figuras

Figura 1 Gráfica comparativa de modelos pequeños	13
Figura 2 Línea temporal con eventos importantes	14
Figura 3 Logo de ChatGPT	17
Figura 4 Logo de DeepSeek	18
Figura 5 Logo de Github Copilot	19
Figura 6 Logo de Claude	20
Figura 7 Logo de Mistral	20
Figura 8 Logo de Meta	21
Figura 9 Logo de Gemini	22
Figura 10 Integración de la IA en la interfaz de IntelliJ IDEA	26
Figura 11 Gráfica del índice de inteligencia de modelos de IA en la nube	27
Figura 12 Gráfica de la velocidad de respuesta de modelos de IA en la nube	27
Figura 13 Gráfica Comparativa sobre MMLU-Pro	28
Figura 15 Gráfica Comparativa sobre GPQA	28
Figura 16 Gráfica Comparativa Inteligencia/Precio	29
Figura 17. Gráfica Comparativa de precio de entrada y salida	30
Figura 18. Gráfica Comparativa de HumanEval	30
Figura 19. Gráfica Comparativa de SciCode	31
Figura 20. Gráfica Comparativa de LiveCodeBench	31
Figura 21 Gráfica del índice de inteligencia de modelos de IA locales	38
Figura 22 Gráfica de la velocidad de respuesta de modelos de IA locales	38
Figura 23 Gráfica Comparativa sobre MMLU-Pro de modelos locales	38
Figura 24 Gráfica Comparativa sobre Humanity's Last Exam de modelos locales	39
Figura 25 Gráfica Comparativa sobre GPQA de modelos locales	39
Figura 26. Gráfica Comparativa de precio de entrada y salida de modelos locales	40
Figura 27 Gráfica Comparativa Inteligencia/Precio de modelos locales	40
Figura 28. Gráfica Comparativa de HumanEval de modelos locales	41
Figura 29. Gráfica Comparativa de SciCode de modelos locales	41
Figura 30. Gráfica Comparativa de LiveCodeBench de modelos locales	41
Figura 31. Gráfico de barras de los resultados de la prueba 1.	53
Figura 32. Gráfico de barras de los resultados de la prueba 2.	54
Figura 33. Gráfico de barras de los resultados de la prueba 3.	55
Figura 34. Gráfico de barras de los resultados de la prueba 4.	55
Figura 35. Capturas de los resultados de la prueba 1.	56
Figura 36. Capturas de los resultados de la prueba 2.	56
Figura 37. Login de Open-WebUI	A5
Figura 38. Descargar un modelo	A5
Figura 39. Navegador con modelo cargado	A5
Figura 40. Mensaje de conexión entre Ollama y Open-WebUI	A7
Figura 42. Lista de modelos de IA cargados en Ollama	A8
Figura 43. Diagrama Conceptual de la Arquitectura	A13

Capítulo 1. Introducción y Objetivos

1.1 Introducción

Este Trabajo Fin de Grado, enmarcado en la Escuela de Ingeniería de Telecomunicación y Electrónica (EITE) y la titulación de Grado en Ingeniería en Tecnologías de la Telecomunicación (GITT) con mención en Telemática, aborda un estudio exploratorio y evaluativo sobre la aplicación de modelos de Inteligencia Artificial Generativa (IAG) para la generación automática de código.

En los últimos años, la aparición e implementación de la IA en diversos campos, incluida la programación web, ha dado lugar a un nuevo paradigma a nivel mundial, caracterizado por herramientas innovadoras que amplían aplicaciones y capacidades. La IA ha transformado significativamente tanto el desarrollo del lado del servidor como del lado del cliente en el ecosistema web. Mientras que en el lado del servidor se utiliza para optimizar la gestión de bases de datos, administrar la carga de servidores y mejorar la seguridad aplicando IAs, en el lado del cliente ha revolucionado la experiencia del usuario a través de chatbots interactivos y asistentes virtuales

Este proyecto se centra en analizar y evaluar modelos de IA para la generación automática de código, aplicándolos específicamente al desarrollo de una aplicación web en el entorno de la asignatura Programación Web del Grado en Ingeniería en Tecnologías de la Telecomunicación, utilizando tecnologías como HTML, CSS, PHP, SQL, y JavaScript, fundamentales en el desarrollo de app web. El objetivo principal no solo es construir una aplicación funcional, sino también explorar cómo los modelos de IA pueden optimizar el flujo de trabajo de los estudiantes, y fomentar el aprendizaje práctico de técnicas modernas de desarrollo web. Se pretende investigar, evaluar y documentar las capacidades de diferentes modelos de IA, identificando conductas y mejores prácticas en el diseño de prompts para maximizar su efectividad en este contexto, sin perder de vista el proceso enseñanza y aprendizaje del estudiante.

La motivación principal de este trabajo surge de la creciente influencia y el rápido progreso de la inteligencia artificial generativa, prompts “avanzados” y su aplicación en el ámbito del desarrollo de software. La programación asistida por IA, *vibe coding*, se ha consolidado como un paradigma en rápida evolución, integrando técnicas de IA para aumentar y automatizar diversas etapas del proceso de programación. La incorporación de la IA tiene una importancia significativa en la ingeniería de software moderna,

ofreciendo el potencial de transformar la forma en que se concibe, desarrolla y mantiene el software.

El uso de IA en la enseñanza de Programación Web puede transformar el proceso educativo. Este enfoque se basa en el análisis de mejores prácticas para optimizar el uso de prompts en modelos de IA, fomentando la creatividad y la resolución de problemas. La IA conversacional, por ejemplo, ofrece una forma más natural e interactiva para que los desarrolladores obtengan asistencia, permitiendo hacer preguntas y recibir orientación en lenguaje natural, lo que puede automatizar tareas repetitivas, gestionar flujos de trabajo y proporcionar acceso rápido a la información, aumentando potencialmente la productividad de los desarrolladores. Los modelos de lenguaje extensos (LLMs) también desempeñan un papel cada vez más importante, yendo más allá de la simple finalización de código para incluir tareas como traducción de código, ayuda en la depuración, diseño front-end e interacción con APIs y bases de datos.

Además, el análisis detallado de los "prompts" es crucial, ya que la forma en que se redactan y estructuran las instrucciones para los modelos de IA puede tener un impacto significativo en la calidad del código generado. Comprender esta relación es fundamental para identificar las mejores prácticas para maximizar la eficiencia y la precisión de las herramientas de generación automática de código. Explorar y analizar el uso de las diferentes implementaciones de la inteligencia artificial en la generación automática de código, e identificar conductas y mejores prácticas en el diseño de prompts para maximizar su efectividad, son objetivos clave de esta investigación.

En el contexto específico de la asignatura Programación Web del GITT, este proyecto busca explorar cómo esta tecnología puede optimizar el flujo de trabajo de estudiantes y docentes, permitiendo a los estudiantes automatizar tareas repetitivas, lo que les facilitaría centrarse en diseñar soluciones innovadoras, colaborar con tecnologías avanzadas y entender los principios éticos y técnicos necesarios en un mercado laboral impulsado por la IA.

1.2 Antecedentes

En los últimos años, la aparición e implementación de la inteligencia artificial en diversos campos y disciplinas ha dado lugar a un nuevo paradigma a nivel mundial. Este fenómeno se caracteriza por la constante aparición de herramientas innovadoras que amplían sus aplicaciones y capacidades con el tiempo. Aunque los resultados pueden variar, el uso principal de la IA se centra en la gestión de datos y las acciones que se pueden realizar con ellos. Algunos ejemplos incluyen la creación y edición de contenido

multimedia [1], así como el análisis y la búsqueda de grandes volúmenes de información en internet¹.

Dentro del ámbito de la programación la integración de la IA ha revolucionado el desarrollo de software [2], particularmente con modelos como GPT-4 de OpenAI [3] y DeepSeek R1 de DeepSeek [4]. Estos modelos han demostrado capacidades sobresalientes en la generación de código y asistencia a programadores, lo que mejora la productividad y reduce los errores en el desarrollo de aplicaciones.

Centrándonos en el ecosistema web, la IA ha transformado significativamente tanto el desarrollo del lado del servidor como del lado del cliente. En el lado del servidor, la IA se utiliza para optimizar la gestión de bases de datos, administrar la carga de servidores y mejorar la seguridad aplicando IAs como Deepcode [5], la cual encuentra vulnerabilidades en el código. Por otro lado, en el lado del cliente, la IA ha revolucionado la experiencia del usuario a través de chatbots interactivos y asistentes virtuales que personalizan el contenido en tiempo real. Además, el uso de estilos avanzados en combinación con HTML5 y JavaScript, junto con bibliotecas de IA como TensorFlow.js [6], permite a los desarrolladores crear aplicaciones web más dinámicas e intuitivas. Esto no solo mejora la estética y funcionalidad de las páginas, sino que también permite implementar características como reconocimiento de voz y procesamiento de imágenes directamente en el navegador. En conjunto, estos avances están redefiniendo la forma en que interactuamos con la web, haciéndola más inteligente y accesible.

Debido a la constante evolución y multiplicación de productos de esta tecnología se vuelve cada vez mayor la necesidad de probar y comparar para determinar su potencial, destacando algunos modelos: GPT-4, por ejemplo, ha mostrado un rendimiento comparable al de los humanos en algunas tareas de programación, aunque los resultados indican que aún es necesario mantener una supervisión humana para validar la exactitud del código generado. Este modelo no solo genera código, sino que también puede refactorizar código existente, mejorando su calidad según varias métricas establecidas de calidad de software.

Por otro lado, DeepSeek R1, destacado por su accesibilidad y costo-eficiencia, presenta una opción viable tanto para uso local como en la nube, ofreciendo a los desarrolladores flexibilidad según las necesidades del proyecto. Esto incluye la implementación de

¹ Nvidia y Bezos invierten en la “startup” Perplexity AI y la valoran en 520 millones de dólares:

<https://www.computerworld.es/article/2104378/nvidia-y-bezos-invierten-en-la-startup-perplexity-ai-y-la-valoran-en-520-millones-de-dolares.html>

modelos en la nube para escalar y acceder remotamente, o la ejecución local para mayor control y seguridad de los datos.

La evaluación de las capacidades de generación de código por parte de las inteligencias artificiales (IA) se realiza mediante métricas que analizan tanto la calidad sintáctica como la funcionalidad y eficiencia del código. Entre estas métricas destacan la tasa de errores de compilación, que mide la validez sintáctica, y la tasa de éxito en pruebas unitarias, que evalúa si el código cumple con los requisitos funcionales mediante un conjunto de pruebas predefinidas [7]. Para medir la calidad estructural, se utilizan indicadores como la complejidad ciclomática, que cuantifica la dificultad lógica del código, y la cobertura de código, que mide la proporción de líneas ejecutadas durante las pruebas [8]. Los benchmarks especializados, como HumanEval y CodeXGLUE, complementan estas métricas al incluir problemas de programación diseñados para probar tanto la exactitud funcional como la capacidad del modelo para generalizar a nuevos problemas [7]. Este enfoque integral permite identificar fortalezas y limitaciones en la generación de código por IA, contribuyendo a su mejora continua.

La aplicación de estas herramientas en el contexto educativo tiene un potencial inmenso. En el Grado en Ingeniería en Tecnologías de la Telecomunicación (GITT), el desarrollo de un Sistema de Gestión que aproveche estas tecnologías puede facilitar tanto la enseñanza como el aprendizaje de la Programación Web [9]. Este proyecto tiene como objetivo no solo construir una app funcional, sino también explorar cómo los modelos de IA pueden optimizar el flujo de trabajo de los desarrolladores y estudiantes, y fomentar el aprendizaje práctico de técnicas modernas de desarrollo web.

A lo largo del desarrollo del proyecto, también se analizarán los "prompts" utilizados, ya que la forma en la que se redactan y estructuran las instrucciones para los modelos de IA puede tener un impacto significativo en la calidad del código generado. Las métricas y métodos de estudio que se usaran para el análisis son: tasa de errores de compilación, tasa de éxito en pruebas unitarias, cobertura de código, calidad del código y tiempo de ejecución. Este análisis contribuirá a identificar las mejores prácticas para maximizar la eficiencia y la precisión de las herramientas de generación automática de código.

1.3 Objetivos

Por tanto, los objetivos perseguidos son analizar y evaluar modelos de IA para la generación automática de código, aplicándolos al desarrollo de una aplicación web en el entorno de la asignatura Programación Web, utilizando tecnologías como HTML, CSS, PHP, SQL, JavaScript, etc. El uso de inteligencias artificiales en la enseñanza de

Programación Web transformará el proceso educativo. Este enfoque se basa en el análisis de mejores prácticas para optimizar el uso de prompts en modelos de IA, fomentando la creatividad y la resolución de problemas.

Para lograr estos objetivos se han definido siguientes objetivos específicos:

OBJ-1.Explorar y analizar el uso de modelos de inteligencia artificial en la generación automática de código: Investigar, evaluar y documentar las capacidades de diferentes modelos de IA, identificando conductas y mejores prácticas en el diseño de prompts para maximizar su efectividad.

OBJ-2.Desarrollar una app web modular y escalable con IA: Crear una aplicación web funcional para la asignatura de Programación Web del Grado en Ingeniería en Tecnologías de la Telecomunicación (GITT), con una arquitectura diseñada para facilitar los conocimientos de la asignatura mediante el uso de IAs generativas.

OBJ-3.Analizar los prompts y evaluar: identificar y documentar buenas prácticas en la creación de prompts para la generación de código relacionado con las tecnologías que forman parte de la programación web.

1.4 Estructura del documento

La estructura de esta memoria está dividida en capítulos los cuales contienen la información y datos obtenidos en el desarrollo de este trabajo.

La memoria se organiza en **capítulo 1**, "Introducción y Objetivos", presenta el trabajo, sus antecedentes en el contexto de la aparición e implementación de la inteligencia artificial y las motivaciones que lo impulsan. Define los objetivos específicos que se buscan alcanzar con la investigación y el desarrollo y describe brevemente los contenidos y la organización de la memoria.

En el **capítulo 2**, "Tecnologías asociadas al uso de la IA", se realiza una revisión de aspectos importantes de la Inteligencia Artificial para un mejor entendimiento, comprensión de su impacto y valor que puede añadir a la asignatura Programación Web del GITT. Se abordan conceptos clave como la Inteligencia Artificial Generativa y la Programación asistida por Inteligencia Artificial, incluyendo el uso de IA conversacional e integración de IA en IDEs, analizando diferentes modelos de IA generales y especializados para código.

El **capítulo 3**, "Modelos de IA OpenSource", explora la Inteligencia Artificial de Código Abierto. Describe sus características fundamentales que garantizan las libertades de uso, estudio, modificación y distribución, la evolución histórica de la IA de código abierto y los

tipos de modelos de IA de código abierto existentes, incluyendo modelos de generación de texto, imagen, vídeo y código. También se incluye un análisis de modelos y los entornos de ejecución requeridos, detallando requisitos de hardware y software.

El **capítulo 4**, "Evaluación Inicial de los Modelos" detalla la evaluación inicial de un grupo de modelos seleccionados bajo los criterios de la asignatura de Programación Web. Se describen los criterios de selección de los modelos y las herramientas de ejecución, el diseño de las pruebas iniciales, divididas en escenarios de Exploración Creativa y Resolución de Objetivos, y se presentan los resultados de las pruebas iniciales. Se describe la elaboración de un proyecto (un Sistema de Gestión de Contenidos) utilizando inicialmente uno de los modelos de IA, y se plantea la discusión sobre el dilema de la IA en asignaturas prácticas, explorando diferentes escenarios de inclusión de la IA en la docencia y su impacto en objetivos y competencias.

Continuando con el estudio, el **capítulo 5**, "Análisis y Resultados" presenta los resultados obtenidos al recrear el proyecto del capítulo anterior con el resto de IAs seleccionadas, partiendo de prompts mejorados. Se detallan las fases y tareas logradas con cada uno de los modelos y se analizan los resultados cuantitativos y cualitativos de los modelos evaluados, incluyendo su rendimiento global, puntos fuertes por sistema y áreas de mejora comunes, así como los resultados específicos obtenidos del análisis de los prompts utilizados.

Por último, el **capítulo 6**, "Conclusiones" se recogen las conclusiones del proyecto. En esta sección, se sintetizan los hallazgos principales derivados de la investigación y la experimentación, incluyendo diferentes planteamientos educativos sobre cómo la IA podría integrarse en la enseñanza de la programación web, y se ofrecen perspectivas sobre posibles futuros desarrollos y líneas de investigación.

1.5. Uso de IA Generativas

En cumplimiento de las "Recomendaciones sobre uso de la IAGen en la UPGC" [10], aprobadas por el Consejo de Gobierno Extraordinario de la ULPGC el 6 de junio de 2024 [11], este TFG ha empleado inteligencia artificial generativa como herramienta de apoyo, usándose de manera equivalente a un navegador web para contrastar fuentes de información, generar ideas y estructurar contenidos. Como ejemplo, se utilizó Google NotebookLM para extraer información exclusivamente de los documentos oficiales de prácticas y proyecto docente de Programación Web. Además se ha realizado uso amplio de las IAs como parte específica del contenido y desarrollo de los objetivos de este TFG.

Capítulo 2. Tecnologías asociadas al uso de la IA

La aparición e integración de la inteligencia artificial en numerosos ámbitos ha cambiado los paradigmas de varias maneras: productividad, calidad y automatización. En este capítulo se revisa aspectos importantes de esta tecnología para un mejor entendimiento, comprensión de su impacto y valor que puede añadir la asignatura Programación Web del Grado en Ingeniería en Tecnologías de la Telecomunicación (GITT), realizando un análisis de los modelos de inteligencia artificial más atractivos para realización de esta asignatura.

2.1 Contextualización

La inteligencia artificial(IA)² se define como el conjunto de técnicas, algoritmos y sistemas que permiten a las máquinas imitar funciones cognitivas humanas, como el aprendizaje, el razonamiento y la resolución de problemas.

2.1.1 Inteligencia Artificial

La función de la IA se centra en mejorar la eficiencia y autonomía de tareas complejas y toma de decisiones de los sistemas a través del procesamiento y análisis de grandes volúmenes de datos. Volviéndose una herramienta importante en la automatización de procesos, la mejora de la interacción entre humanos y sistemas y el desarrollo de aplicaciones capaces de aprender y adaptarse continuamente.

Los inicios de la inteligencia artificial se remontan a orígenes que combinan tradiciones mitológicas y filosóficas, con los primeros avances en matemáticas y computación, donde ya se especulaba con la creación de seres artificiales [12].

Los primeros intentos formales de creación se consolidaron en la década de 1940, con la formulación del modelo de neuronas artificiales por McCulloch y Pitts³ y el influyente ensayo de Alan Turing⁴. A lo largo de los años, se han sucedido distintos auges en el campo: desde el entusiasmo por los sistemas expertos en los años 80 hasta el resurgimiento del aprendizaje profundo a partir de 2012⁵, tras los periodos conocidos como "*inviernos de la IA*" [13] que marcaron una pausa en la financiación e investigación.

² Según la Real Academia Española (RAE), es la disciplina científica que crea programas informáticos capaces de ejecutar operaciones comparables a las de la mente humana.

³ McCulloch, Warren S., and Walter H. Pitts. "Un cálculo lógico de las ideas inmanentes en la actividad nerviosa." Margaret A. Boden (Comp.)(1994). Filosofía de la Inteligencia Artificial. Fondo de Cultura Económica (1943)

⁴ A. M. Turing, "Computing Machinery and Intelligence"

⁵ Una red neuronal convolucional se utiliza para ganar el concurso de reconocimiento de imágenes sobre ImageNet <https://en.wikipedia.org/wiki/ImageNet>

2.1.2 Inteligencia Artificial Generativas

Un avance significativo ocurrió en 2014 con la introducción de las Redes Generativas Adversarias (GAN) por Ian Goodfellow⁶ y sus compañeros, capaces de crear contenido realista, como imágenes y videos, que antes solo podían ser producidos por humanos.

En este contexto de rápido progreso, la aparición de ChatGPT por parte de OpenAI en noviembre de 2022 marcó un hito trascendental [14]. La presentación de ChatGPT, un LLM con una interfaz de chatbot accesible, capturó la atención global debido a su habilidad para mantener conversaciones coherentes y generar respuestas detalladas sobre una amplia gama de temas. Este evento no solo demostró el potencial de la IA conversacional, sino que también provocó una intensa reflexión en la comunidad mundial sobre sus implicaciones en diversos ámbitos y disciplinas, acuñando IA como el uso de IA generativa/conversacional.

El panorama actual de la inteligencia artificial se caracteriza por avances continuos y significativos en diversas áreas. En 2024 y 2025, se han observado mejoras notables en el rendimiento de los modelos de IA en benchmarks exigentes, así como una creciente integración de la IA en la vida cotidiana [15]. Los modelos más pequeños están alcanzando niveles de rendimiento que antes solo eran posibles con modelos mucho más grandes, lo que indica una mayor eficiencia y accesibilidad (Ver Figura 1). Además, el costo de consultar modelos de IA con capacidades similares a las de GPT-3.5 ha disminuido drásticamente [16].

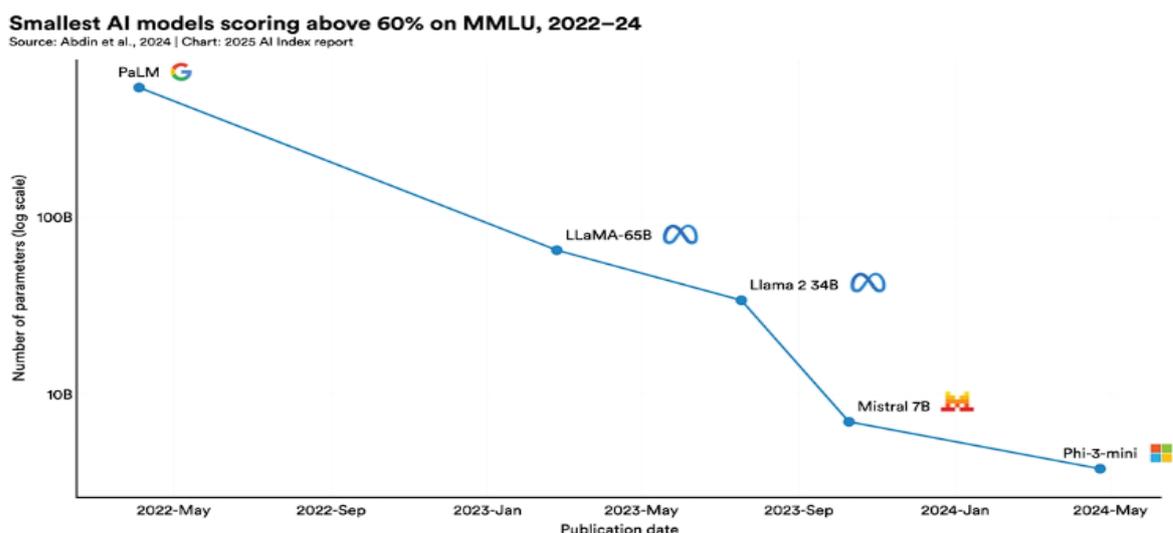


Figura 1 Gráfica comparativa de modelos pequeños [15]

⁶ I. J. Goodfellow et al., «Generative Adversarial Networks»

Historia de la IA

Eventos importantes

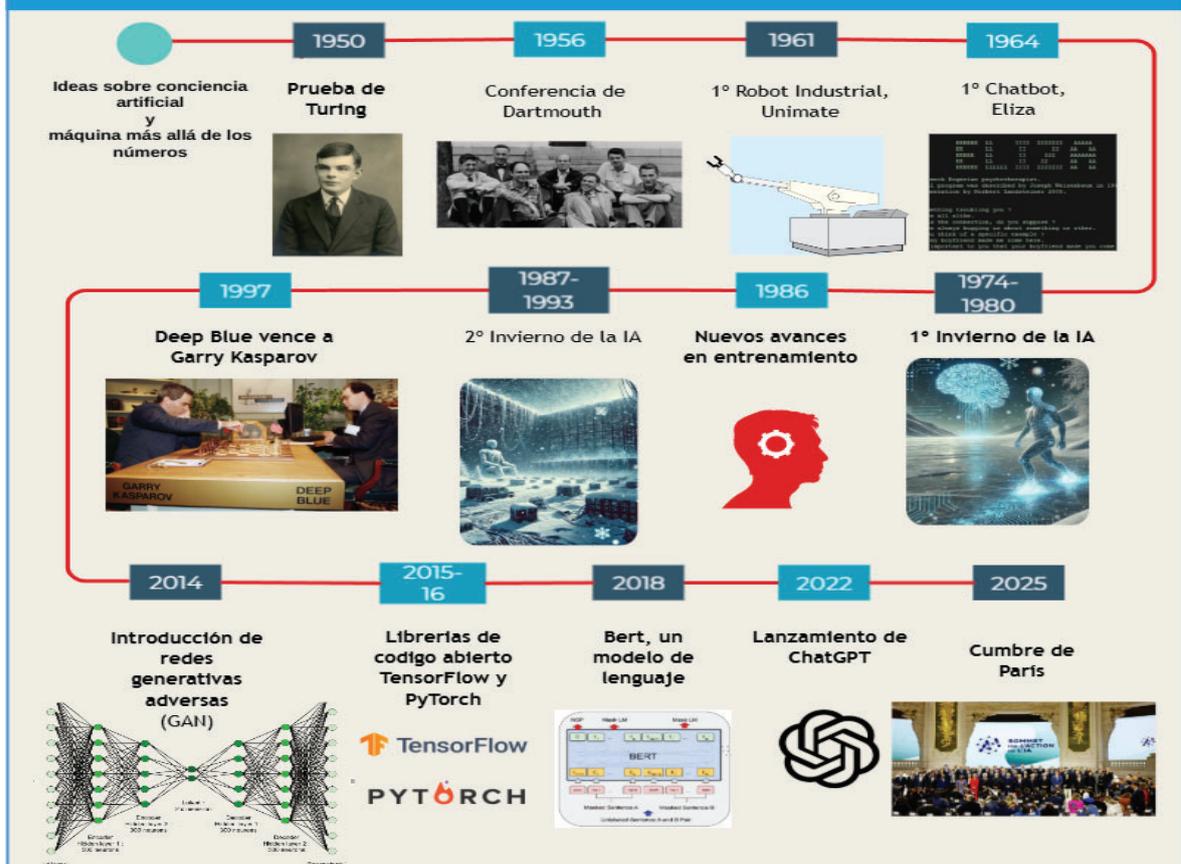


Figura 2 Línea temporal con eventos importantes, Elaboración propia

2.1.3 Características de las IA

Una de las tendencias más destacadas es el auge de la IA multimodal, que puede procesar e integrar información de diferentes tipos de datos como texto, imágenes, audio y video [17]. Los agentes de IA, sistemas autónomos capaces de planificar y actuar para lograr objetivos, también están ganando terreno. Estos agentes tienen el potencial de transformar la forma en que trabajamos y realizamos diversas tareas

Actualmente, la IA ha pasado de ser teórica a una tecnología presente en reconocimiento, procesamiento de lenguaje y generación de contenidos, ocupando un papel central en la era digital, con el constante desarrollo de servicios que la incorporan y respaldo financiero de diversos organismos⁷. Todo bajo la constante carrera protagonizada por los países

⁷ La UE pone en marcha la iniciativa InvestAI para movilizar 200 000 millones EUR de inversión en inteligencia artificial: <https://digital-strategy.ec.europa.eu/es/news/eu-launches-investai-initiative-mobilise-eu200-billion-investment-artificial-intelligence>

tecnológicamente líderes por dominar esta tecnología, mientras otros plantean como restringirla y controlarla (Ver Figura 2, 2025 cumbre de París).

Clasificación de la IA

Existen diversas categorías para etiquetar IAs, un ejemplo es según su nivel de inteligencia en una tarea específica (IA débil o estrecha), cualquier ámbito intelectual (IA fuerte o general) o casi todos los campos: social, intelectual y creativo (ASI o Superinteligencia artificial) [18]. (Ver Tabla 1)

Tabla 1. Categorización general

Tipo de IA	Característica Principal	Ejemplo
IA Débil o Estrecha (ANI)	Diseñada para realizar tareas específicas sin comprender el contexto general.	Asistentes virtuales como Siri o Alexa.
IA Fuerte o General (AGI)	Hipotética IA con capacidad de razonar y aprender como un humano en diferentes dominios.	No existe actualmente.
Superinteligencia Artificial (ASI)	IA teórica que superaría la inteligencia humana en todos los aspectos.	No existe actualmente.

Otros autores, en concreto Stuart J. Russell y Peter Norvig [13], diferencian las IAs en base si actúan o piensan racionalmente o como humanos. Una visión más biológica es la que propone Arend Hintze [19] (Ver Tabla 2) donde son diferenciadas según las capacidades mentales que poseen: memoria, raciocinio y emociones, siendo las emociones el punto más avanzado.

Tabla 2. Categorización de Hintze

Tipo de IA	Característica Principal	Ejemplo
IA Reactiva	No tiene memoria ni capacidad de aprendizaje, solo responde a estímulos actuales.	Deep Blue, computadora de ajedrez de IBM.
IA de Memoria Limitada	Puede aprender de experiencias pasadas en un rango limitado.	Modelos de predicción del clima.
Teoría de la Mente	Aún en investigación; busca comprender emociones e intenciones humanas para interactuar socialmente.	Robots sociales en desarrollo.
IA Autoconsciente	Hipotética IA con autoconciencia y capacidad de reflexión.	No existe actualmente.

Al intentar recrear aspectos humanos, las clasificaciones de IAs son extensas y en constante expansión, pues nuevos modelos logran niveles antes hipotéticos y emergen nuevos campos en un ámbito aún no consolidado.

Otras etiquetas relacionadas con aspectos como su capacidad de obtener datos mediante distintos tipos de fuentes (multimodal) [20], crear nuevos datos mediante el aprendizaje

de patrones (generativa) [21] o aspectos de diseño como su impacto y alineación con valores humanos (IA amigable)⁸. (Ver tabla 3)

Tabla 3. Otros tipos de IA

Tipo de IA	Característica Principal	Ejemplo
Basada en Reglas	Utiliza reglas predefinidas para tomar decisiones.	Sistemas expertos.
Basada en Aprendizaje Automático	Aprende patrones a partir de datos sin reglas predefinidas.	Algoritmos de Machine Learning.
Generativa	Crea contenido nuevo basado en datos previos.	DALL-E, ChatGPT.
Multimodal	Procesa e integra texto, imagen y audio.	GPT-4, Gemini.
Basada en Redes Neuronales	Emula patrones neuronales humanos.	Deep Learning, CNNs, RNNs.
Simbólica	Usa lógica y reglas para representar conocimiento explícitamente.	Lógica difusa, sistemas de inferencia.
Basada en Algoritmos Evolutivos	Optimiza soluciones con técnicas basadas en evolución biológica.	Algoritmos genéticos.
Probabilística	Toma decisiones con incertidumbre usando teoría de probabilidad.	Redes Bayesianas.

Tokens y Parámetros

Para un mejor entendimiento, particularmente en los modelos de lenguaje de gran escala, existen dos conceptos que definen su estructura interna: los tokens y los parámetros [21].

Podemos pensar en los tokens como las piezas básicas del lenguaje que el modelo manipula, similar a cómo un niño aprende primero con bloques de construcción antes de crear estructuras más complejas. Estos tokens pueden ser palabras completas o fragmentos de ellas, y constituyen el vocabulario fundamental con el que el modelo interactúa.

Por otro lado, los parámetros son como los "conocimientos adquiridos" del modelo: representan los valores internos que se ajustan durante el proceso de entrenamiento, permitiendo reconocer y generar patrones lingüísticos cada vez más sofisticados.

Ambas características son aspectos importantes para la generación por parte de este tipo de IAs. Individualmente el número de tokens que el modelo es capaz de procesar afecta a su habilidad de contexto, mientras a mayor número de parámetros es más capaz de aprender patrones y matices más complejos en el lenguaje. En conjunto, mientras más tokens útiles se puedan extraer del contexto y mayor sea la capacidad (número de parámetros) para procesarlos, mejores serán las respuestas generadas por la IA. Esta

⁸ Eliezer Yudkowsky

sinergia es clave para que un modelo produzca resultados sofisticados y adaptados al contexto de la consulta.

2.2 Desarrollos más importantes en IA generativa

En esta sección se recogen algunas de las empresas más importantes por el impacto de sus IAs o por ser empresas tecnológicas líderes a fecha de la redacción.

2.2.1 OpenAI



Figura 3 Logo de ChatGPT

GPT-4o [22], el modelo multimodal de OpenAI, representa una evolución significativa en el campo de la programación asistida por inteligencia artificial, construyendo sobre los fundamentos establecidos por su predecesor GPT-4 Turbo. Su arquitectura omni, que incorpora un entrenamiento especializado en código y matemáticas, abarca un amplio espectro de capacidades técnicas: desde la generación de código funcional en diversos lenguajes de programación como Python y JavaScript, hasta el análisis y depuración contextualizada de código existente, incluyendo la proposición de optimizaciones y adherencia a mejores prácticas. El modelo se distingue por su capacidad de procesar consultas técnicas que combinan múltiples modalidades, incluyendo texto e imágenes de interfaces o diagramas, lo que facilita una interacción más natural y efectiva. En el ámbito práctico, estas capacidades se traducen en aplicaciones concretas como asistencia avanzada en entornos de desarrollo integrado, traducción fluida entre lenguajes de programación, explicaciones pedagógicas de conceptos complejos como algoritmos de aprendizaje automático, y análisis comprehensivo de repositorios mediante integración con herramientas de desarrollo. Además, el modelo implementa salvaguardas éticas robustas para prevenir usos indebidos, mientras mantiene un alto nivel de eficiencia operativa y precisión en sus respuestas, especialmente en contenido técnico y documentación.

OpenAI o1 [23], marca un hito revolucionario en la inteligencia artificial aplicada al desarrollo de software gracias a su innovador sistema de reflexión aumentada, que le permite analizar problemas de programación con una profundidad contextual sin precedentes. Este avance se materializa en cuatro capacidades fundamentales que transforman el proceso de desarrollo: una sofisticada generación de código que incluye validación en tiempo real mediante simulaciones de ejecución previas a la implementación; un sistema de refactorización inteligente que optimiza tanto el

rendimiento como la legibilidad del código sin comprometer su funcionalidad; una asistencia técnica multinivel que abarca desde la depuración de scripts hasta el diseño de arquitecturas orientadas a la nube; y un análisis predictivo de dependencias que anticipa y previene conflictos en la actualización de sistemas heredados. Lo que verdaderamente distingue a este modelo es su capacidad para modelar procesos mentales a través de cadenas de razonamiento verificables, permitiendo a los desarrolladores comprender y seguir la lógica detrás de cada solución propuesta, lo que facilita una colaboración más efectiva entre la máquina y el programador.

OpenAI o3-mini [24], es una evolución de o1, representa un avance significativo en IA aplicada al desarrollo de software, destacando como la opción óptima para tareas técnicas exigentes. Diseñado específicamente para STEM, ofrece capacidades sobresalientes en programación gracias a su arquitectura optimizada para gestión de código competitivo (Codeforces), ingeniería de software compleja (SWE-bench Verified) y resolución algorítmica. Destaca su sistema de esfuerzo de razonamiento ajustable (bajo, medio, alto), que permite priorizar velocidad (7.7s de latencia promedio) o precisión, superando en rendimiento a modelos anteriores: 39% menos errores críticos y 56% de preferencia en evaluaciones frente a o1-mini. Integra herramientas como búsqueda contextual con fuentes actualizadas, generación estructurada de código y compatibilidad con function calling, siendo ideal para flujos de trabajo que requieren iteración rápida. Su enfoque de seguridad deliberativa asegura respuestas alineadas éticamente, incluso en escenarios de alta complejidad.

2.2.2 DeepSeek



Figura 4 Logo de DeepSeek

DeepSeek-R1 [4], representa un avance significativo en el campo de la inteligencia artificial aplicada a la programación, demostrando un nivel de competencia que se acerca notablemente al de los programadores humanos expertos. Este modelo ha alcanzado un hito destacable al obtener un rating de 2,029 en la plataforma Codeforces, una métrica que lo sitúa por encima del 96.3% de los participantes humanos, lo que demuestra su capacidad excepcional para resolver problemas algorítmicos complejos. Su excelente desempeño se extiende también a otros entornos de evaluación estándar de la industria, como LiveCodeBench y SWE Verified, donde ha demostrado una notable versatilidad en la generación y análisis de código a través de múltiples lenguajes de programación. Lo que distingue especialmente a DeepSeek-R1 es su naturaleza de código abierto, que permite a la comunidad de desarrolladores no solo examinar y comprender su

funcionamiento interno, sino también modificar y adaptar el modelo según sus necesidades específicas. Además, su arquitectura ha sido diseñada para permitir su despliegue en infraestructura local, otorgando a las organizaciones un control completo sobre sus datos y procesos de desarrollo, una característica particularmente valiosa para empresas con requisitos estrictos de privacidad y seguridad. Estas capacidades sobresalientes, resultado de un sofisticado proceso de entrenamiento por refuerzo, han dotado al modelo de una comprensión profunda de los principios de programación, permitiéndole no solo asistir a los desarrolladores en la creación de soluciones innovadoras, sino también contribuir significativamente a la optimización de los procesos de ingeniería de software, marcando así un nuevo estándar en la colaboración entre inteligencia artificial y desarrollo de software.

2.2.3 Microsoft



Figura 5 Logo de Github Copilot

GitHub **Copilot** [25] de Microsoft representa una innovación transformadora en el campo del desarrollo de software, materializando la visión de un asistente de programación impulsado por inteligencia artificial que fundamentalmente redefine la manera en que los desarrolladores interactúan con el código. En el núcleo de esta herramienta, desarrollada mediante la colaboración entre GitHub y OpenAI, se encuentra un sofisticado modelo de lenguaje basado en OpenAI Codex, que ha sido entrenado con millones de líneas de código público, permitiéndole comprender y generar código en una amplia gama de lenguajes de programación. La verdadera fortaleza de Copilot radica en su capacidad para integrarse perfectamente en los entornos de desarrollo más utilizados por la comunidad de programadores, como Visual Studio Code, Visual Studio, Neovim y la suite completa de IDEs de JetBrains, donde ofrece sugerencias contextuales en tiempo real que van desde simples autocompletados hasta la generación de funciones completas. Una de sus características más notables es su habilidad para transformar descripciones en lenguaje natural, expresadas como comentarios, en código ejecutable funcional, automatizando así tareas repetitivas y permitiendo a los desarrolladores concentrarse en aspectos más creativos y estratégicos de su trabajo. Sin embargo, es crucial entender que Copilot funciona como un colaborador inteligente más que como un reemplazo del juicio humano, siendo fundamental que los desarrolladores mantengan un papel activo en la revisión y refinamiento de las sugerencias generadas para asegurar que el código final cumpla con los estándares de calidad y seguridad requeridos.

2.2.4 Anthropic



Figura 6 Logo de Claude

La familia de modelos **Claude 3** [26] representa un avance significativo en el campo de la inteligencia artificial aplicada a la programación, destacándose por su capacidad para abordar de manera integral el ciclo completo del desarrollo de software. Esta nueva generación de modelos ha demostrado una notable evolución en sus capacidades fundamentales, abarcando desde la generación precisa de código hasta la depuración avanzada y la edición contextual inteligente. Su excelencia se evidencia de manera cuantitativa en evaluaciones estandarizadas de la industria como HumanEval, APPS y MBPP, donde ha superado significativamente el rendimiento de sus predecesores gracias a mejoras sustanciales en su capacidad de razonamiento y su comprensión profunda del lenguaje técnico especializado. Un aspecto particularmente innovador de Claude 3 es su arquitectura multimodal, que permite al modelo procesar e interpretar simultáneamente información tanto textual como visual, facilitando así el análisis comprehensivo de diagramas técnicos, estructuras de código y representaciones visuales de arquitecturas de software en una amplia gama de lenguajes de programación. Esta combinación de capacidades avanzadas posiciona a Claude 3 como una herramienta transformadora en el panorama del desarrollo de software, ofreciendo no solo la automatización eficiente de procesos rutinarios sino también un apoyo sofisticado en la resolución de tareas complejas que requieren un alto nivel de comprensión y razonamiento técnico.

De esta familia destaca **Claude 3.5 Sonnet**, modelo mejorado de Claude 3 Sonnet, que destaca por su relación inteligencia/coste, siendo más *barato* en tokens frente a su hermano más pesado, Opus, y a su vez más inteligente que este.

2.2.5 Mistral



Figura 7 Logo de Mistral

Con respecto a las IAs europeas está la familia de modelos de código abierto desarrollada por la empresa francesa, Mistral AI [27], representa una evolución significativa en el campo de los modelos de lenguaje. Los modelos de **Mistral AI** han mostrado una evolución significativa en sus capacidades de programación. Todo comenzó con Mistral 7B [28], que ya destacó en la generación de código a pesar de ser

un modelo relativamente pequeño de 7 mil millones de parámetros. Su arquitectura innovadora con Grouped-Query Attention le permitió competir eficientemente con modelos mucho más grandes en tareas de programación.

La familia **Codestral** representa la especialización específica de Mistral AI en programación. El Codestral 22B [29] se diseñó específicamente para tareas de desarrollo de software, mientras que su variante Codestral Mamba incorporó una arquitectura optimizada que le permite generar código de manera más eficiente. Una característica especialmente notable de Codestral Mamba es su capacidad para trabajar con más de 80 lenguajes de programación diferentes, lo que lo hace extremadamente versátil para diferentes entornos de desarrollo.

El **Mixtral 8x7B** [30], aunque no está específicamente orientado a la programación, incorporó avances significativos que mejoran su rendimiento en tareas de código. Su arquitectura de mezcla de expertos dispersos (SMoE) le permite procesar información de manera más eficiente que modelos anteriores, superando incluso a GPT-3.5 y Llama 2 70B en varias evaluaciones de programación.

El desarrollo más reciente, **Mistral Large 2** [31], con sus 123 mil millones de parámetros y una ventana de contexto de 128,000 tokens, representa un salto cualitativo en capacidades de programación. Su extensa ventana de contexto resulta especialmente útil para trabajar con archivos de código largos o múltiples archivos simultáneamente, permitiendo mantener un mejor entendimiento del contexto global del proyecto.

2.2.6 Meta



Figura 8 Logo de Meta

La familia de modelos **LLaMA** [32], desarrollada por Meta, representa una evolución en el panorama de los modelos de lenguaje de gran escala. Los modelos LLaMA iniciales, introducidos en 2023, establecieron nuevos estándares arquitectónicos al incorporar mejoras como la pre-normalización RMSNorm en cada subcapa del transformador, la función de activación SwiGLU en lugar de ReLU para optimizar el rendimiento, y las incrustaciones posicionales rotatorias (RoPE) integradas en todas las capas. Una característica distintiva de LLaMA radica en su entrenamiento exclusivo con conjuntos de datos públicos, aproximadamente 1.4 billones de tokens que incluyen CommonCrawl en inglés (67%), C4 (15%), datos de GitHub (4.5%), Wikipedia (4.5%), Gutenberg y Books3 (4.5%), ArXiv (2.5%) y Stack Exchange (2%).

En términos de rendimiento, los modelos LLaMA han demostrado su competitividad en evaluaciones exhaustivas de zero-shot y few-shot across 20 benchmarks especializados. LLaMA-65B supera consistentemente a Chinchilla-70B y PaLM-540B en tareas de razonamiento de sentido común, mientras que en generación de código, LLaMA-13B y versiones superiores superan a LaMDA 137B en los benchmarks HumanEval y MBPP. La implementación de finetuning de instrucciones (LLaMA-I) en LLaMA-65B ha evidenciado mejoras sustanciales en el rendimiento MMLU y una capacidad superior para seguir instrucciones complejas [32].

Llama 2 marcó una evolución de la arquitectura original y sirvió como fundamento para desarrollos especializados como Code Llama [33]. La transición hacia Llama 3 introdujo modelos de hasta 405B parámetros con una ventana de contexto expandida a 128K tokens, mejorando significativamente las capacidades multilingües, de codificación, razonamiento y uso de herramientas [34].

El salto más reciente se muestra en los modelos **Llama 4** [35] Scout y Maverick, introducidos en abril de 2025, que amplían las capacidades de la familia de modelos al enfocarse en la creación de contenido, la comprensión visual y el procesamiento multimodal, permitiendo la integración y conversión fluida de diversos formatos de datos (texto, video, imágenes y audio) dentro de un único modelo. Llama 4 Scout incorpora 17 mil millones de parámetros activos dentro de un total de 109 mil millones, organizados mediante 16 "expertos" especializados, con una ventana de contexto revolucionaria de 10 millones de tokens que supera ampliamente los 128K de su predecesor. Llama 4 Maverick escala esta arquitectura a 400 mil millones de parámetros distribuidos en 128 expertos, implementando una arquitectura de mezcla de expertos que activa selectivamente los parámetros necesarios por operación. Ambos modelos integran capacidades multimodales nativas mediante técnicas de fusión temprana de texto, imagen y video, aprovechando la tecnología propietaria MetaCLIP para la codificación visual avanzada .

2.2.7 Google



Figura 9 Logo de Gemini

La familia **Gemini** [36], desarrollada por Google DeepMind, representa el enfoque de la empresa en la creación de modelos multimodales integrados. Lanzada en 2023, esta serie establece un nuevo paradigma al permitir la comprensión y operación nativa a

través de texto, imágenes, audio y video dentro de una arquitectura unificada [37]. Esta capacidad multimodal marca un diferenciador estratégico en la búsqueda de sistemas de IA más generales y versátiles.

Gemini Ultra [36] es el modelo más potente de su generación en la familia, logrando resultados de vanguardia en 30 de 32 benchmarks evaluados. Su capacidad de alcanzar un rendimiento a nivel de experto humano en MMLU con puntuaciones superiores al 90%. Las capacidades de razonamiento multimodal de Ultra permiten tareas complejas como la verificación de soluciones de física escritas a mano, identificando errores específicos en procesos de razonamiento matemático.

Gemini 1.5 Pro [38] introdujo una arquitectura de mezcla de expertos optimizada para la eficiencia computacional y el procesamiento de contextos extremadamente largos. Esta capacidad mejoró significativamente el estado del arte en tareas de Preguntas y Respuestas sobre documentos extensos y contenido audiovisual, demostrando una recuperación de información casi perfecta en contextos largos.

Gemini 2.0 Flash y Flash Thinking [39] representan la transición hacia los modelos de razonamiento de primera generación de Google. Flash sirve como núcleo para sistemas agénticos avanzados, ejemplificado en el Project AMIE para aplicaciones médicas, donde extiende las capacidades multimodales al mundo físico con comprensión espacial y temporal mejorada.

Gemini 2.5 Pro Experimental [37], el modelo más sofisticado de DeepMind para abordar desafíos de alta complejidad. Lidera actualmente el LMArena y benchmarks especializados en matemáticas y ciencias como GPQA y AIME 2025, destacando por sus capacidades de razonamiento avanzado y codificación especializada para crear aplicaciones web visualmente atractivas y sistemas de código agéntico. Su innovación principal radica en el "Deep Think mode", que permite al modelo evaluar múltiples hipótesis antes de generar respuestas, optimizando la precisión en casos de uso altamente complejos.

2.3 Programación asistida por Inteligencia Artificial

La programación asistida por IA se ha consolidado como un paradigma en rápida evolución dentro del campo del desarrollo de software. Esta disciplina integra técnicas de inteligencia artificial para aumentar y automatizar diversas etapas del proceso de programación. La creciente incorporación de la IA tiene una importancia significativa en la ingeniería de software moderna, ofreciendo el potencial de transformar la forma en que se

concibe, desarrolla y mantiene el software Destacándose dos vertientes: el uso de IAs especializadas en código o modelos generales a través de interfaces de chat y la integración de las IAs en los IDEs, ambos generan código y asisten a desarrolladores, aumentando productividad y reduciendo errores.

2.3.1 IA conversacional

La IA conversacional [40] ofrece una forma más natural e interactiva para que los desarrolladores obtengan asistencia. Tiene el potencial de ayudar en la recopilación de requisitos de las partes interesadas y la gestión de tareas relacionadas con el proyecto. Los LLMs pueden procesar la entrada en lenguaje natural de diversas fuentes para ayudar a definir el alcance del proyecto y realizar un seguimiento del progreso. En lugar de depender únicamente de la finalización de código o el análisis estático, los desarrolladores pueden hacer preguntas y recibir orientación en lenguaje natural. Los chatbots pueden automatizar tareas repetitivas, gestionar flujos de trabajo y proporcionar acceso rápido a la información, lo que potencialmente aumenta la productividad de los desarrolladores. Esta capacidad para agilizar el proceso de desarrollo convierte a la IA conversacional en una valiosa herramienta.

Existe una tendencia creciente a integrar chatbots directamente en los IDEs y otras herramientas de desarrollo para proporcionar asistencia [41]. Esta integración hace que la IA conversacional sea fácilmente accesible dentro del entorno familiar del desarrollador, lo que reduce el cambio de contexto y facilita el aprovechamiento de la asistencia de la IA.

Modelos de IA Generales

Los LLMs, como GPT-4, desempeñan un papel cada vez más importante en la programación que va más allá de la simple finalización de código [42]. Estos modelos versátiles tienen una amplia gama de aplicaciones potenciales en el desarrollo de software. Una de ellas es la traducción de código entre diferentes lenguajes de programación. Esta capacidad puede ser valiosa para el desarrollo multiplataforma y la migración de sistemas heredados. Los LLMs también pueden aprender de los errores, analizando los mensajes de error y refinando iterativamente su código. Esto sugiere un potencial para que los LLMs ayuden en la depuración y mejoren el código con el tiempo.

Además, los LLMs pueden manejar tareas de programación más amplias, como el diseño front-end, la interacción con APIs y las operaciones de bases de datos. Su amplio conjunto de datos de entrenamiento les permite tener conocimientos en diferentes dominios y tecnologías de programación. Los LLMs también pueden ayudar en el diálogo

exploratorio y la recopilación de requisitos, facilitando la comunicación entre desarrolladores y usuarios finales para comprender mejor los requisitos del software [43]. Sus capacidades de comprensión del lenguaje natural pueden ayudar a cerrar la brecha entre los desarrolladores técnicos y las partes interesadas no técnicas. En esencia, los LLMs son herramientas poderosas con un potencial significativo para mejorar diversos aspectos del proceso de desarrollo de software.

Modelos de Inteligencia Artificial Especializados para Código

Dentro del ámbito de la programación asistida por IA, existen modelos de inteligencia artificial especializados diseñados para tareas de programación particulares. Los modelos de generación de código, como Codex de OpenAI y la serie GPT (GPT-2, GPT-3, GPT-4), destacan por su capacidad para traducir lenguaje natural o código parcial en fragmentos de código completos [44]. Estos modelos representan un avance significativo en la accesibilidad de la programación, ya que potencialmente permiten a personas con conocimientos de programación limitados generar código funcional a partir de descripciones en lenguaje natural.

Un ejemplo del uso de estos modelos especializados es el análisis estático y dinámico, donde herramientas como DeepCode e IntelliCode utilizan IA para identificar errores, vulnerabilidades de seguridad y problemas de rendimiento. Mientras el análisis estático examina el código sin ejecutarlo, el dinámico lo evalúa durante la ejecución [45]. Esta capacidad proactiva de detección supera las limitaciones de las revisiones humanas tradicionales, identificando problemas que podrían pasar desapercibidos.

Una vez detectados los problemas, la IA no se limita a señalarlos, sino que también los corrige automáticamente. Las técnicas de corrección de código emplean modelos especializados que aprovechan el aprendizaje por refuerzo y el ajuste fino en conjuntos de datos de errores y correcciones conocidas [46]. Esta automatización reduce significativamente el tiempo dedicado a la depuración manual.

Para lograr esta comprensión profunda del código, algunos modelos de IA utilizan representaciones estructuradas como el Árbol de Sintaxis Abstracta (AST) [47]. Esta aproximación basada en árboles permite a la IA comprender los aspectos estructurales y semánticos del código de manera más efectiva que el simple procesamiento de texto plano, proporcionando un contexto rico para el análisis y la generación de código.

Finalmente, los mecanismos de atención refinan todo este proceso al permitir que los modelos se enfoquen selectivamente en las partes más relevantes del contexto del

código durante las predicciones [47]. Esta capacidad de atención selectiva resulta en sugerencias más precisas y contextualmente apropiadas.

2.3.2 IDEs con integración de IA

Existe una creciente tendencia a integrar funciones avanzadas de IA en los IDEs convencionales, como VS Code (por ejemplo, GitHub Copilot, Agent Mode) e IntelliJ IDEA [48] (ver Figura 10).

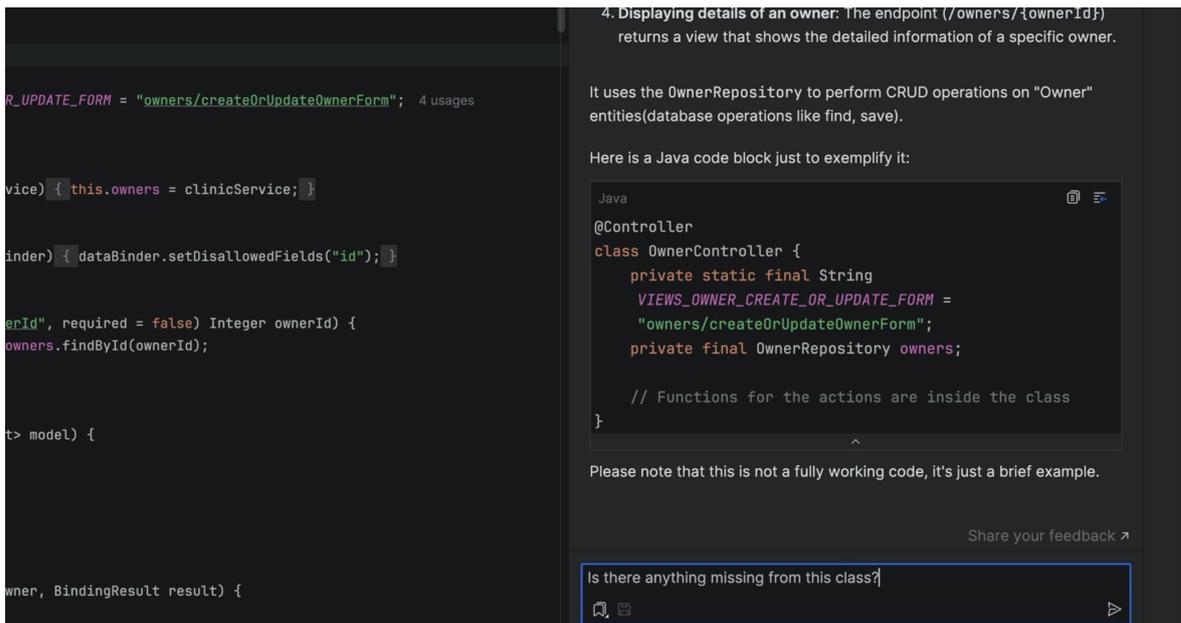


Figura 10 Integración de la IA en la interfaz de IntelliJ IDEA

La principal característica de estas herramientas se centra en la interpretación y control del código en tiempo real, abarcando aspectos como la redacción del código proporcionando recomendaciones automáticas, completación líneas de código y la generación de funciones completas [49]. Además de aspectos de mantenimiento como la refactorización y detección inteligente de errores durante la escritura del código y ofreciendo sugerencias para su corrección [45].

2.4 Análisis de modelos de IA

En este apartado se mostrarán gráficas elaboradas a partir de los datos disponibles de los modelos en la plataforma Artificial Analysis [50], un analista independiente de modelos de IAs y proveedores de APIs.

2.4.1 Características generales

En esta sección se muestran gráficas sobre aspectos comunes relacionados que pueden interesar a usuarios comunes:

Índice de Inteligencia

Esta métrica es propia de Artificial Analysis, donde se evalúan diversos aspectos relacionados con la inteligencia mediante un conjunto de datos: conocimiento, matemáticas, programación, razonamiento general, razonamiento científico y cuantitativo.(Figura 11)

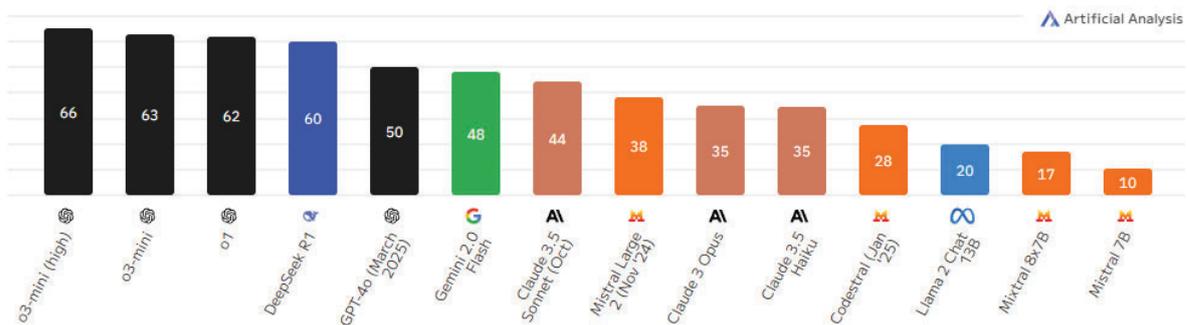


Figura 11 Gráfica del índice de inteligencia de modelos de IA en la nube [50]

Velocidad de salida

La siguiente gráfica representa la velocidad de tokens generados por segundo, liderado por modelos de OpenAI, Mistral y Gemini 2.0 siendo el principal atractivo de las empresas. En el caso contrario está DeepSeek R1, que destaca de ser el más lento, posiblemente debido a su bajo coste. (Figura 12)

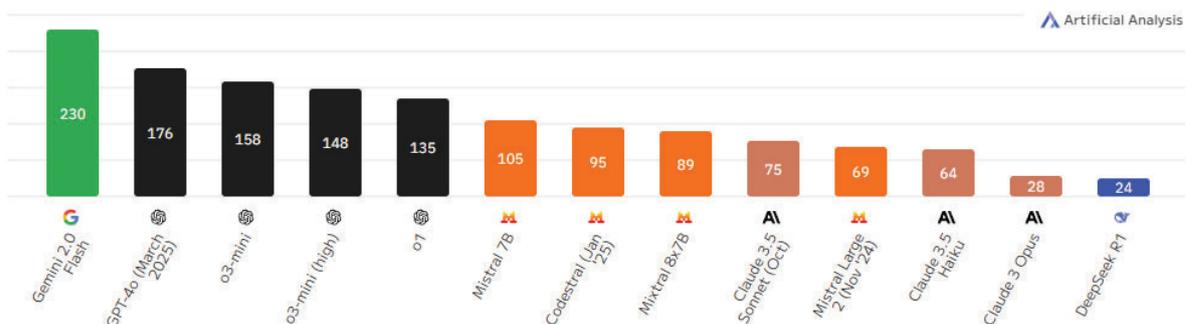


Figura 12 Gráfica de la velocidad de respuesta de modelos de IA en la nube [50]

2.4.2 Razonamiento

En este apartado se muestran una serie de gráficas relacionadas con el razonamiento y conocimiento, tanto general como científico. Como se podría esperar, los modelos en los primeros puestos son los que tienen capacidades de razonamiento, destacando DeepSeek R1 en conocimiento general (Figuras 13 y 14), mientras que los modelos

razonantes de OpenAI en conocimiento científico (Figura 15). Además se destaca la cercanía de Claude 3.5 Sonnet y Gemini 2.0 a los modelos razonantes y el posicionamiento en los últimos puestos por parte de los modelos de Mistral.

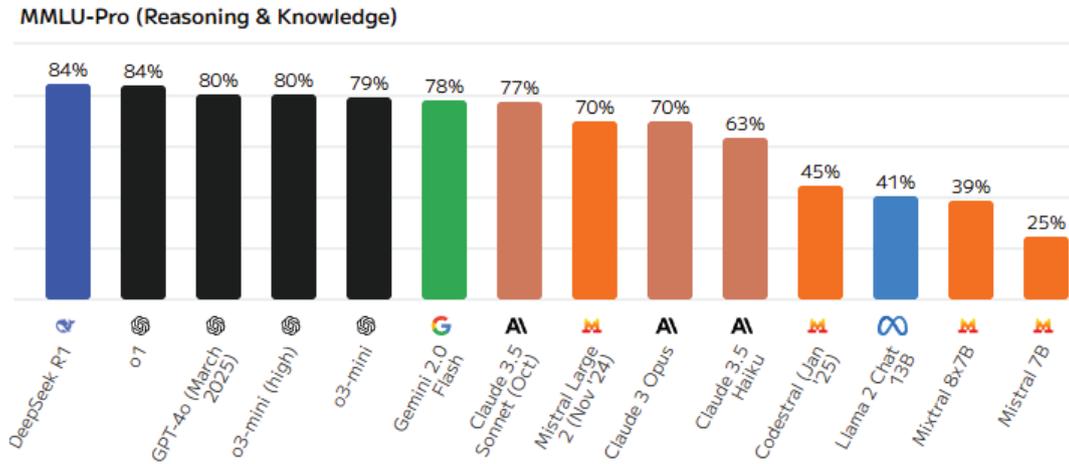


Figura 13 Gráfica Comparativa sobre MMLU-Pro [50]

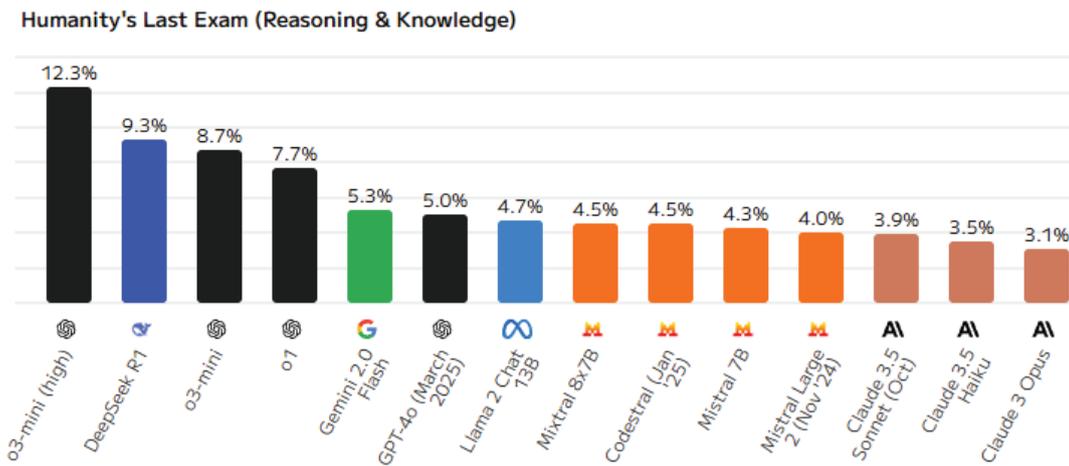


Figura 14 Gráfica Comparativa sobre Humanity's Last Exam [50]

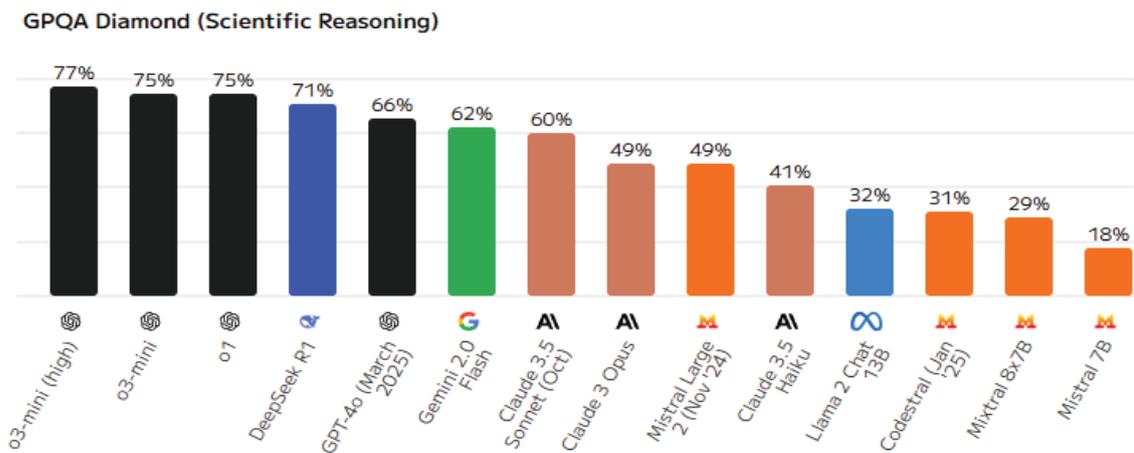


Figura 15 Gráfica Comparativa sobre GPQA [50]

2.4.3 Coste de uso

En el ámbito económico, determinar los costes de los procesos e implementar estrategias para optimizarlos resulta fundamental en proyectos y empresas. Los servicios vinculados a la IA no son la excepción. Comprender y cuantificar el coste asociado al uso de esta tecnología es clave para cualquier proceso que la incorpore. Con este objetivo, se presentan una serie de gráficos que analizan la relación entre capacidad y coste en el desarrollo y aplicación de sistemas de IA.

Inteligencia

En muchos casos, más puede ser caro, pero en el caso de la IA no siempre es así, como se muestra en la grafica no existe una relacion lineal entre la inteligencia y su coste (Dolar por 1000 mil tokens), destacando el cuadrante de color verde con DeepSeek R1 y o3-mini, donde superan en inteligencia a otros modelos pero a su vez siendo la opcion más barata. Siendo la otra cara de la moneda o1, siendo con diferencia el más caro pese su *intelecto*. (Figura 16)

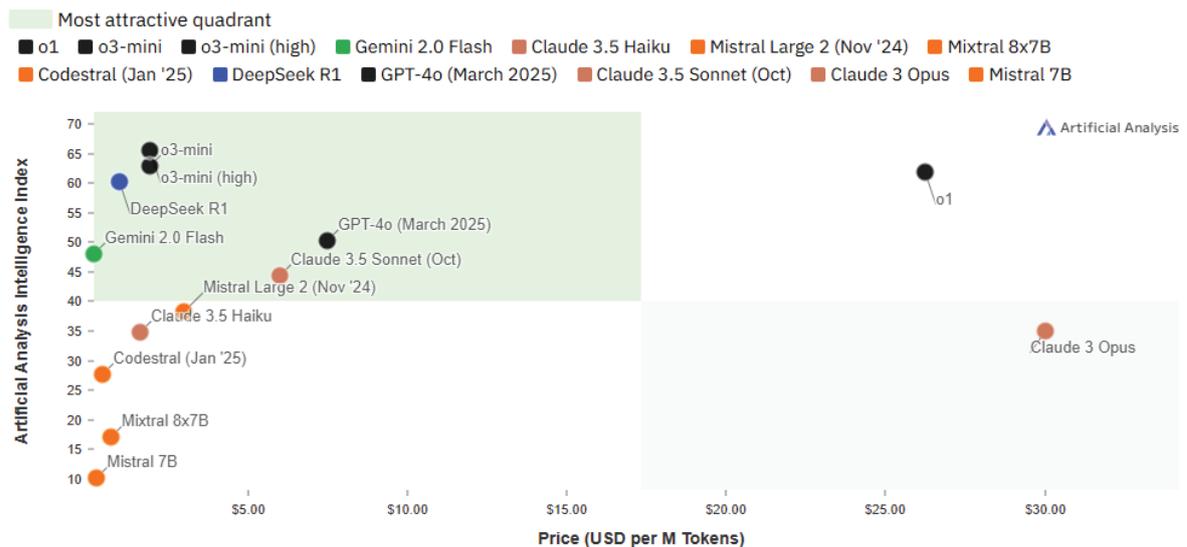


Figura 16 Gráfica Comparativa Inteligencia/Precio [50]

Entradas y Salidas

Desde un punto de vista más orientado al uso de APIs, el escenario es distinto al anterior, donde destaca algunos modelos de Mistral y Gemini 2.0, aunque DeepSeek R1 y o3-mini siguen superando a la media, y o1 sigue con el título del más caro. (Figura 17)

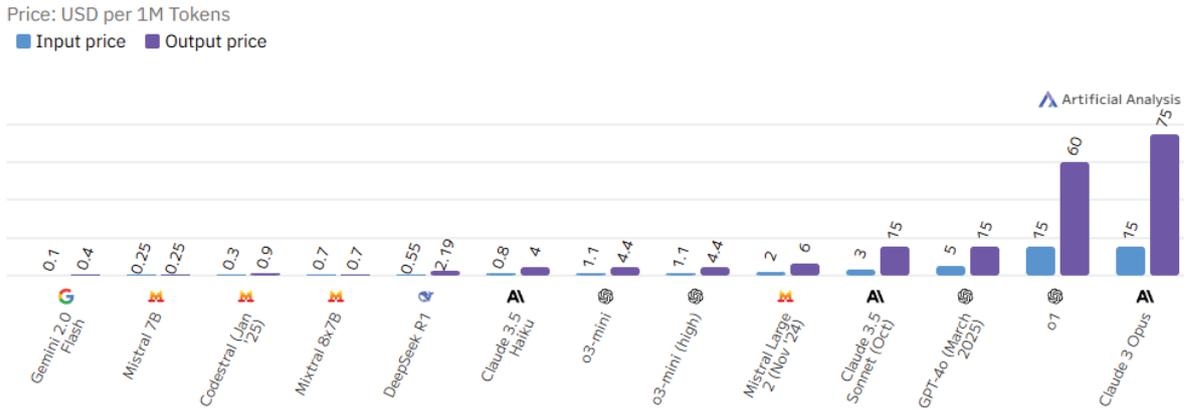


Figura 17. Gráfica Comparativa de precio de entrada y salida [50]

2.4.4 Métricas de desempeño

Un aspecto fundamental para la programación web es el código y las distintas acciones que lo rodean: explicación, documentación, generación, mantenimiento, etc. Debido a esto es fundamental comparar los modelos en función de su habilidad para el código. Como se aprecia en las siguientes gráficas, los modelos razonantes lideran los primeros puestos, seguidos por Claude 3.5 Sonnet presentando un desempeño cercano a estos y menos de cerca Gemini 2.0 (Figuras 18, 19 y 20).

Se debe tener en cuenta que en el contexto de la asignatura de programación web los componentes base son html, css y javascript en la parte frontend y php en backend, los cuales no son evaluados en los benchmark (python) que se muestran a continuación, con lo que se debe tomar con cautela los resultados mostrados e interpretarse como una evaluación de habilidades relacionadas con la programación y no como el desempeño generativo de los modelos en varios lenguajes de código.

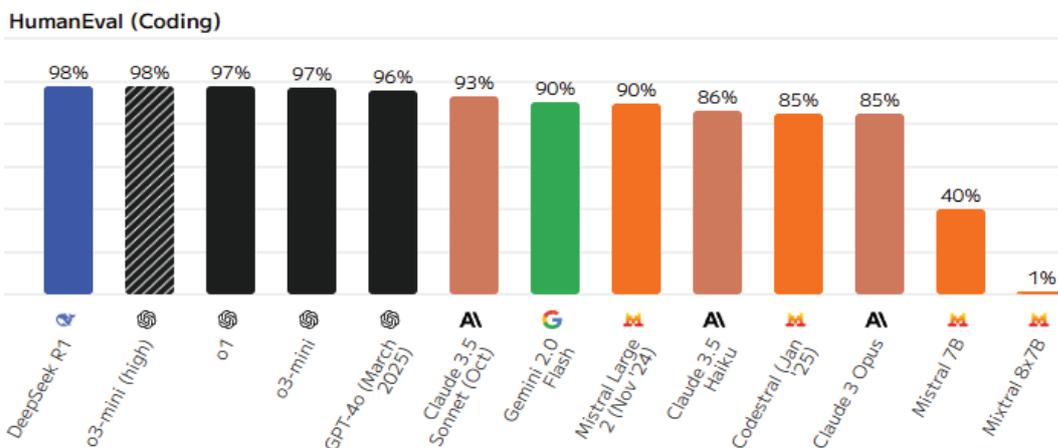


Figura 18. Gráfica Comparativa de HumanEval [50]

SciCode (Coding)

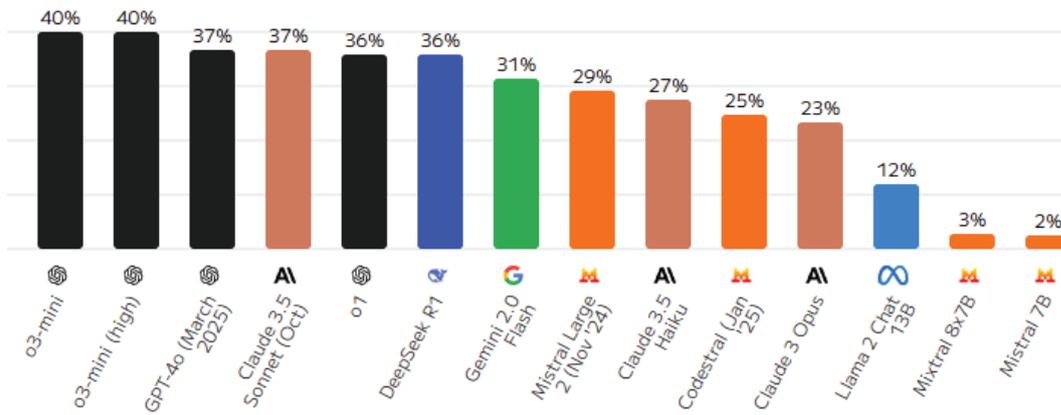


Figura 19. Gráfica Comparativa de SciCode [50]

LiveCodeBench (Coding)

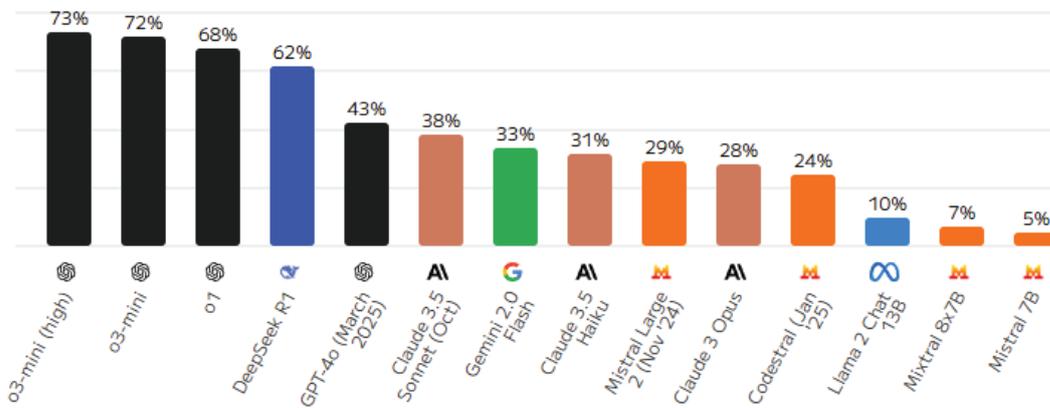


Figura 20. Gráfica Comparativa de LiveCodeBench [50]

Capítulo 3. Modelos de IA OpenSource

3.1 Definiendo la IA de Código Abierto

La creciente influencia de la inteligencia artificial en la sociedad ha generado un interés en los modelos de IA de código abierto. Estos sistemas, cuyo código fuente está disponible públicamente para su uso, estudio, modificación y distribución, fomentan un entorno colaborativo y transparente que impulsa la innovación, reduce las barreras de entrada y respalda un desarrollo ético de la IA [51]. La comunidad de aprendizaje automático aspira a valores inherentes como la colaboración, la revisión por pares, la transparencia, la fiabilidad, la flexibilidad y la accesibilidad [52], convirtiendo a la IA de código abierto en un elemento crítico para democratizar el acceso a tecnologías avanzadas [53].

La Open Source Initiative (OSI) define la IA de código abierto como un sistema que otorga las libertades de usar, estudiar, modificar y compartir tanto el sistema como sus elementos discretos. Wikipedia complementa esta definición, extendiendo estos atributos a cada componente del sistema, incluidos los conjuntos de datos, el código y los parámetros del modelo [54]. La definición 1.0 de IA de código abierto de la OSI requiere que un sistema se ponga a disposición bajo términos que garanticen estas libertades fundamentales, aplicándose tanto a modelos base como entrenados, pesos, parámetros y otros elementos estructurales [55].

Los modelos de IA de código abierto se caracterizan por cuatro libertades centrales: usar el sistema para cualquier propósito sin solicitar permiso; estudiar su funcionamiento e inspeccionar sus componentes; modificar el sistema para cualquier propósito, incluyendo el cambio de su salida; y compartir el sistema con o sin modificaciones [53]. Estas libertades se aplican tanto al sistema completo como a sus elementos estructurales [55].

Para ejercer estas libertades, los requisitos incluyen acceso al código fuente completo utilizado para entrenar y ejecutar el sistema, los parámetros del modelo, información detallada sobre los datos de entrenamiento, y licencias apropiadas. La definición de la OSI exige "información suficientemente detallada sobre los datos utilizados para entrenar el sistema, de modo que una persona cualificada pueda construir un sistema sustancialmente equivalente", aunque no requiere compartir los datos de entrenamiento reales [51]. Los modelos se publican frecuentemente bajo licencias FOSS como Apache License, MIT License y GNU General Public License [54].

3.2 La Evolución Histórica de la IA de Código Abierto

Los principios de código abierto en el software, basados en el libre intercambio y uso de información a través de redes colaborativas, han sentado las bases para su aplicación en el campo de la inteligencia artificial [56]. La IA de código abierto ha sido una parte integral de la inteligencia artificial desde sus inicios, y los valores que sustentan el código abierto se alinean con las aspiraciones de la comunidad de aprendizaje automático desde su concepción [52]. La historia de la IA de código abierto está intrínsecamente ligada al desarrollo de las tecnologías de IA y al crecimiento del movimiento del software de código abierto [54]. La Free Software Foundation (FSF) desempeñó un papel fundamental en el establecimiento de los principios del software libre, que influyeron en la IA de código abierto [57].

A lo largo de la historia de la IA de código abierto, se han producido hitos clave y han surgido proyectos influyentes. Las primeras bibliotecas de ML de código abierto, como Scikit-learn, OpenCV, Torch y Theano, fueron fundamentales para la investigación académica. La revolución del aprendizaje profundo se vio impulsada por entornos de código abierto como TensorFlow y PyTorch. La publicación de modelos de código abierto influyentes, como BERT de Google, avanzó significativamente la investigación en PLN. Hugging Face se ha convertido en un centro central para compartir y distribuir modelos de IA de código abierto, particularmente en PLN [54]. El lanzamiento por parte de Meta de la familia de modelos LLaMA marcó un paso significativo para hacer accesibles los modelos de lenguaje grandes [58].

Las iniciativas de código abierto han tenido un impacto profundo en la progresión y accesibilidad de las tecnologías de IA a lo largo del tiempo. El código abierto ha brindado una accesibilidad generalizada a las herramientas de aprendizaje automático y ha desempeñado un papel crucial en el desarrollo y la adopción de modelos de lenguaje grandes, democratizando el acceso a tecnologías lingüísticas avanzadas. Los modelos de traducción automática de código abierto han allanado el camino para la compatibilidad multilingüe en diversas aplicaciones [54]. El auge de los modelos fundacionales de código abierto está dando lugar a un ecosistema próspero en la IA generativa [59]. Además, los esfuerzos de código abierto han cambiado el equilibrio de poder en la industria de la IA, reduciendo las barreras de entrada y acelerando la innovación [60].

3.3 Tipos de Modelos de IA de Código Abierto

3.3.1 Modelos de IA Generativa

Los modelos de IA generativa de código abierto ofrecen flexibilidad y control a los desarrolladores, permitiéndoles modificar y ampliar el código según sea necesario. Se benefician del apoyo y la colaboración de la comunidad, lo que fomenta la innovación y el desarrollo rápido [61]. Las opciones de código abierto proporcionan transparencia, lo que permite el escrutinio del código subyacente y la información de los datos de entrenamiento [62]. La naturaleza de código abierto mejora la adaptabilidad, el desarrollo colaborativo y la transparencia de los modelos de IA generativa.

El panorama del código abierto para la IA generativa es rico y diverso, y ofrece modelos para diversas modalidades y tareas, a menudo rivalizando con las alternativas propietarias. Algunos ejemplos destacados:

Tabla 4. Diversos modelos de IA generativa

Modelo	Descripción
GPT-NeoX	Desarrollado por EleutherAI, una alternativa de código abierto a modelos como GPT-3, conocido por generar contenido coherente de formato largo.
BLOOM	Un modelo multilingüe que admite más de 50 idiomas, promoviendo la inclusión y accesibilidad para investigadores de todo el mundo.
OPT (Open Pretrained Transformer)	Desarrollado por Meta, diseñado para la generación de texto a gran escala y la investigación, manejando tareas complejas de PLN.
T5 (Text-to-Text Transfer Transformer)	De Google, versátil y adaptable para diversas tareas de PLN, con puntos de control preentrenados disponibles.

3.3.2 Modelos de Embebido (Embeddings)

Los modelos de embebido convierten el texto en representaciones vectoriales que capturan el significado semántico detrás de las palabras [63]. Estos vectores representan las características esenciales de la entrada, preservando las relaciones semánticas y la información estructural [64]. Los modelos de embebido son cruciales para tareas como la búsqueda semántica, la recuperación de información, la agrupación, la clasificación y los sistemas de recomendación [63]. Son fundamentales para permitir que la IA comprenda el significado del texto y otros datos, facilitando una amplia gama de aplicaciones posteriores.

Los modelos de embebido de código abierto a menudo igualan o superan la calidad de las opciones de código cerrado en las tablas de clasificación de referencia como MTEB

[65]. Las características clave incluyen el tamaño del modelo (número de parámetros), el uso de memoria, las puntuaciones de rendimiento de MTEB, las dimensiones del embebido y el tamaño del contexto (tokens de entrada máximos) [66]. El rendimiento puede variar según la tarea específica (recuperación, clasificación, agrupación, etc.) y el idioma de los datos [67]. La evaluación de los modelos de embebido de código abierto requiere considerar varias métricas y su idoneidad para casos de uso específicos, ya que el rendimiento de referencia en conjuntos de datos generales puede no siempre traducirse en un rendimiento óptimo en datos especializados.

Algunos ejemplos de modelos de embebido de código abierto conocidos incluyen:

Tabla 5. Diversos modelos de IA embebida

Modelo	Descripción
all-mpnet-base-v2	Modelo de sentence-transformers para codificar oraciones y párrafos cortos; entrenado con más de mil millones de pares de oraciones.
Modelos BGE (p. ej., bge-en-icl)	De la Academia de IA de Beijing (BAAI); alta calidad y multilingües. BGE-M3 destaca por su multifuncionalidad y compatibilidad con más de 100 idiomas.
Modelos GTE (p. ej., gte-base)	De Alibaba Group, con buen rendimiento en recuperación multilingüe.
Modelos E5 (p. ej., e5-small-v2, e5-mistral-7b-instruct)	Enfocados en preentrenamiento contrastivo débilmente supervisado; destacan por eficiencia y rendimiento.
Nomic Embed (p. ej., nomic-embed-text)	Modelos compactos y eficientes para texto, con versiones multimodales disponibles.
Nomic Embed Code	Entrenado específicamente para código; ofrece alto rendimiento.
SciBERT	Modelo BERT preentrenado en texto científico; vocabulario enfocado en la ciencia.
BioSentVec	Vectores sent2vec preentrenados para texto biomédico.

3.3.3 Modelos de Generación de Vídeo e Imagen

El campo de la IA para la generación de contenido visual ha avanzado rápidamente. El aprendizaje profundo ha impulsado un aumento en los modelos generativos para la síntesis de imágenes y vídeos [68]. Los modelos de difusión de texto a imagen han llevado la generación de imágenes a la vanguardia de la investigación en aprendizaje automático [69]. La IA generativa está revolucionando la creación de vídeos al automatizar la generación de contenido realista y de alta calidad [70].

Los modelos de código abierto ofrecen personalización, portabilidad, transparencia y pueden mitigar riesgos como el sesgo [71]. Los modelos de generación de imágenes a menudo utilizan modelos de difusión, redes generativas adversarias y modelos autorregresivos [72]. Los modelos de generación de vídeo deben manejar la propiedad espacio-temporal de los vídeos y requieren importantes recursos computacionales [70].

Las características clave incluyen el número de parámetros, los datos de entrenamiento, la resolución, la velocidad de fotogramas y la capacidad de manejar indicaciones de texto e imagen [73]. Los modelos visuales de código abierto se caracterizan por sus diversas arquitecturas y los esfuerzos continuos para mejorar la calidad y la eficiencia de la salida, especialmente para la generación de vídeo.

Ejemplos notables de modelos de código abierto en esta categoría incluyen:

Tabla 6. Diversos modelos de IA generación de video e imágenes

Categoría	Modelo	Descripción
Generación de Imágenes	Stable Diffusion	Modelo prominente de código abierto para generar imágenes fotorrealistas a partir de texto; altamente personalizable y con un gran ecosistema.
	DeepFloyd IF	Combina imágenes realistas con una fuerte comprensión del lenguaje.
	FLUX.1	De Black Forest Labs, supera a modelos propietarios en calidad y adherencia a las indicaciones.
	Animagine XL	Diseñado para generar imágenes de estilo anime de alta calidad.
	Playground 2.5	Basado en Stable Diffusion XL, entrenado para reflejar el estilo visual de Midjourney.
Generación de Vídeo	HunyuanVideo	De Tencent, modelo de código abierto con 13B de parámetros; ofrece calidad cinematográfica y precisión en movimiento.
	Mochi (Mochi 1)	De Genmo, modelo de difusión con 10B de parámetros, destaca por su fidelidad de movimiento y adherencia a indicaciones.
	Wan2.1	De Alibaba, conjunto completo para generación, edición y conversión texto a vídeo; incluye soporte multilingüe.
	LTX Video	De Lightricks, optimizado para velocidad y eficiencia; capaz de generación en tiempo real en GPUs asequibles.
	CogVideoX	De THUDM, modelo basado en transformadores para generación de vídeo de alta calidad desde descripciones textuales.

3.3.4 Modelos de Generación de Código

Los modelos de generación de código de código abierto se entrenan con grandes cantidades de código disponible públicamente, a menudo de proyectos de código abierto [74]. Las características clave incluyen el número de parámetros, la gama de lenguajes de programación compatibles, la capacidad de manejar indicaciones en lenguaje natural y el rendimiento en puntos de referencia como HumanEval [75]. Algunos modelos ofrecen funciones como finalización de código, asistencia para la depuración, traducción de código y documentación automatizada [76].

Ejemplos de modelos de generación de código de código abierto incluyen:

Tabla 7. Diversos modelos de IA generación de código

Modelo / Plataforma	Descripción
Code Llama	De Meta, basado en Llama 2; diseñado para generación, comprensión y finalización de código en múltiples lenguajes, con variantes según tamaño de parámetros.
DeepSeek-Coder	De DeepSeek AI, especializado en generación y relleno de código; destaca por su rendimiento en HumanEval.
Qwen-Coder (Qwen2.5-Coder-Instruct)	De Alibaba Cloud; adaptado a tareas de código, logra altos puntajes en HumanEval y soporta múltiples lenguajes.
Phi-3.5-mini-instruct	De Microsoft, modelo ligero y multilingüe para generación, depuración y comprensión de código.
WaveCoder-Ultra	De Microsoft, enfocado en generación, resumen, traducción y reparación de código.
CodeGen	De Salesforce; entrenado con datos de lenguaje natural y programación, con buen rendimiento en HumanEval.
PolyCoder	De la Universidad Carnegie Mellon, sobresale en programación en C, superando a otros modelos en ese lenguaje.

3.4 Análisis de modelos

En este apartado se mostrarán una serie de gráficas elaboradas a partir de los datos disponibles en Artificial Analysis de los modelos con posibilidad de ser ejecutados localmente. Los modelos mostrados pertenecen a empresas en diferentes sectores como el tecnológico (Google con la serie Gemma, Microsoft con la serie Phi y Meta con la serie Llama), la inteligencia artificial (Mistral y DeepSeek con modelos del mismo nombre de la empresa) y algunas de otros sectores (Alibaba con la serie Qwen).

3.4.1 Características generales

En esta sección se muestran gráficas sobre aspectos comunes relacionados que pueden interesar a usuarios comunes:

Índice de Inteligencia

Esta métrica es propia de Artificial Analysis, donde se evalúan diversos aspectos relacionados con la inteligencia mediante un conjunto de datos: conocimiento, matemáticas, programación, razonamiento general, razonamiento científico y cuantitativo. (Ver Figura 21)



Figura 21 Gráfica del índice de inteligencia de modelos de IA locales [50]

Velocidad de salida

La siguiente gráfica representa la velocidad de tokens generados por segundo, liderado por modelos de Meta (Llama), Mistral (Codestral) y Alibaba (Qwen) siendo el principal atractivo de las empresas. En el caso contrario está DeepSeek R1 y los modelos de Microsoft, que destaca de ser el más lento, posiblemente en relación a su bajo coste. (Ver Figura 22)

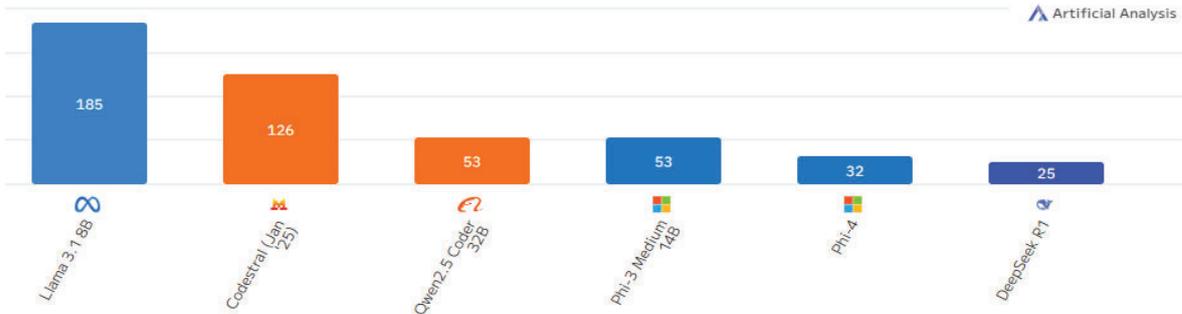


Figura 22 Gráfica de la velocidad de respuesta de modelos de IA locales [50]

3.4.2 Razonamiento

En este apartado se muestran una serie de gráficas relacionadas con el razonamiento y conocimiento, tanto general como científico. Como se podría esperar, los modelos en los primeros puestos son los que tienen capacidades de razonamiento, destacando DeepSeek R1 en conocimiento general, mientras que los puestos de modelos sin razonamiento los posee Alibaba y Microsoft (ver Figuras 23 y 25).

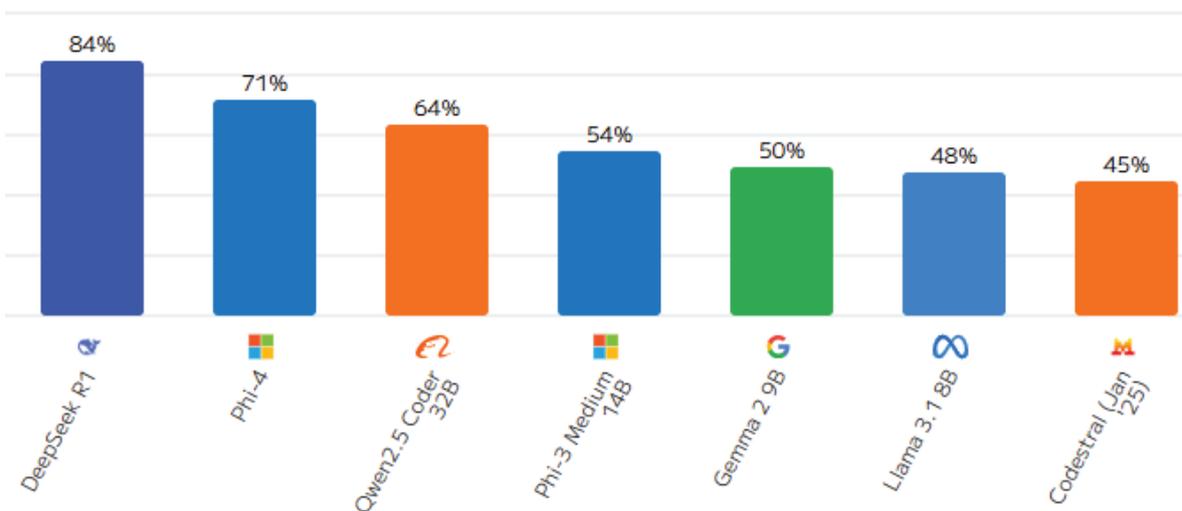


Figura 23 Gráfica Comparativa sobre MMLU-Pro de modelos locales [50]

Humanity's Last Exam (Reasoning & Knowledge)

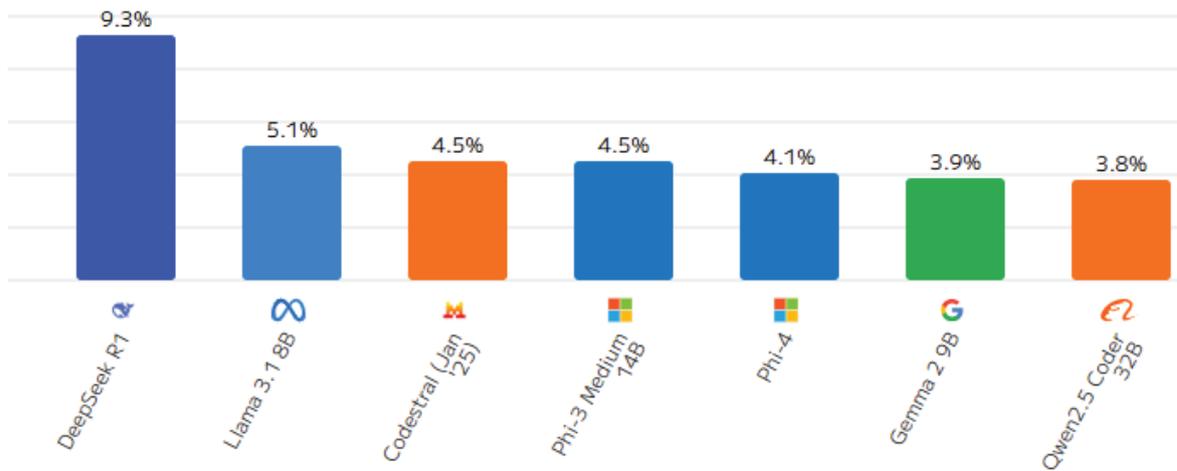


Figura 24 Gráfica Comparativa sobre Humanity's Last Exam de modelos locales [50]

GPQA Diamond (Scientific Reasoning)

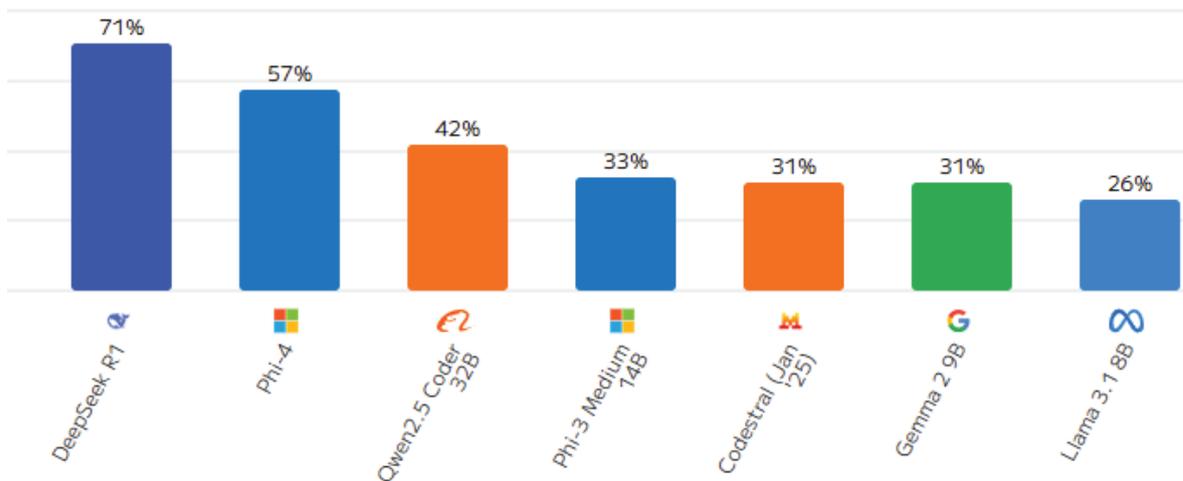


Figura 25 Gráfica Comparativa sobre GPQA de modelos locales [50]

3.4.3 Relación Precio

Aunque la implementación en local no requiera de servicios externos, conviene estudiar los gastos de un servicio de API para determinar si es más rentable que la preparación de un entorno que ejecute IA local.

Entradas y Salidas

En el contexto del coste de la entrada y las salidas, en los modelos de menor coste relación entrada/salida destacan Llama, Gemma y Qwen, mientras que los modelos de Microsoft junto Codestral se encuentran entre los modelos de menor y mayor coste. Por último, el modelo con diferencia más caro es DeepSeek-R1. (Figura 26)

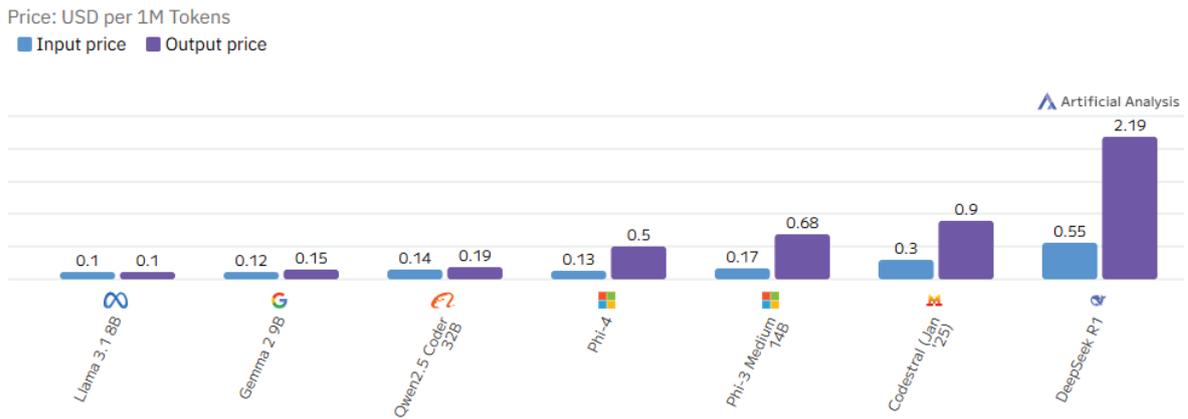


Figura 26. Gráfica Comparativa de precio de entrada y salida de modelos locales [50]

Inteligencia

En el caso de la relación de inteligencia/coste, ningún modelo se encuentra dentro del cuadrante *atractivo* (mejor relación inteligencia/coste), los modelos más cercanos al son Phi-4 y Qwen, mientras que el resto se alejan por coste (DeepSeek) o por inteligencia (Gemma). (Figura 27)

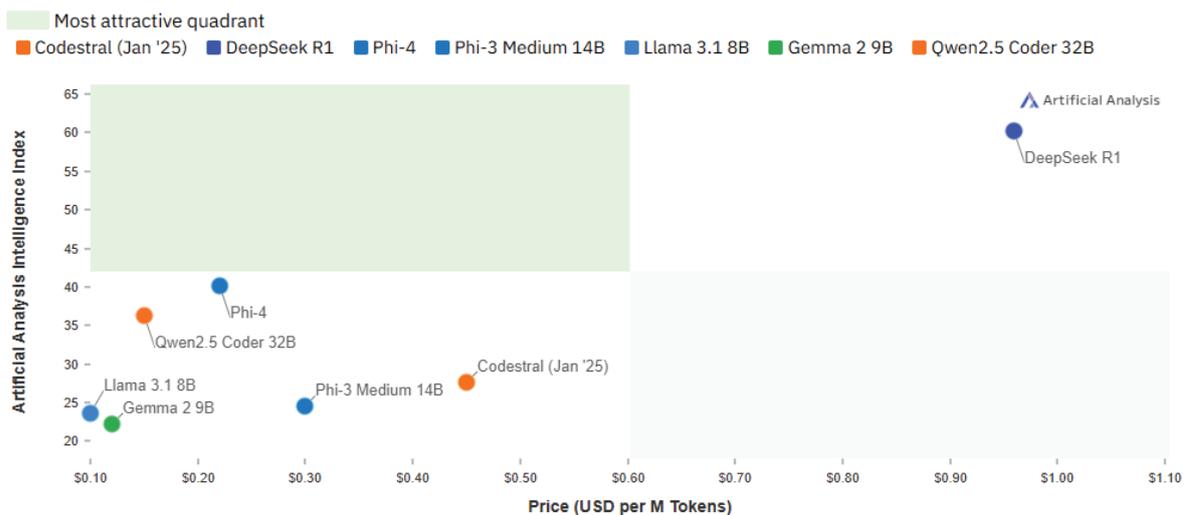


Figura 27 Gráfica Comparativa Inteligencia/Precio de modelos locales [50]

3.4.4 Programación

Como se aprecia en las siguientes gráficas, DeepSeek lidera los benchmarks relacionadas con programación, seguido por los modelos Qwen y Phi-4 que dependiendo de las pruebas tienen un desempeño parecido a DeepSeek pese la diferencia de carecer de razonamiento (Figuras 28 a 30).

HumanEval (Coding)

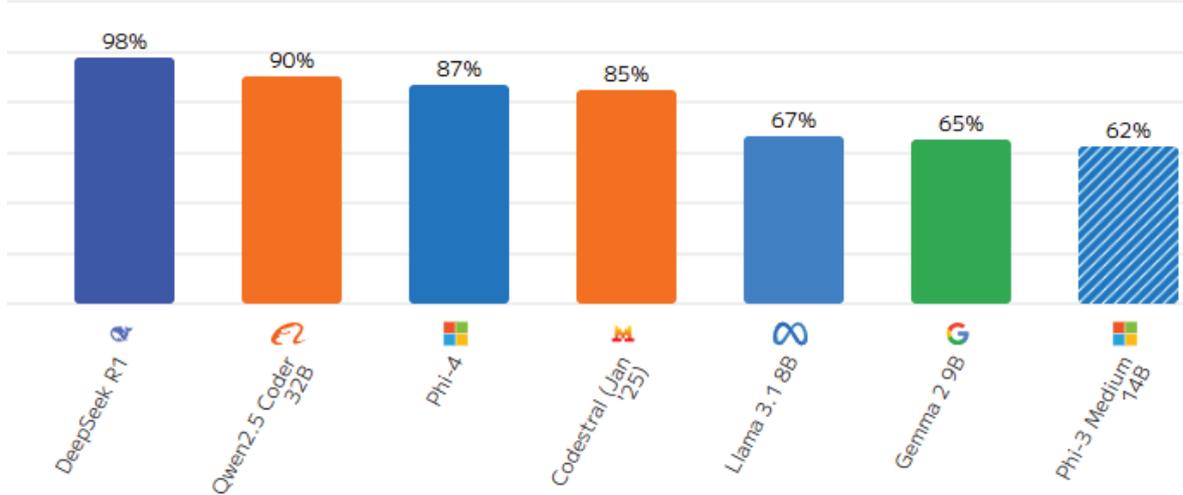


Figura 28. Gráfica Comparativa de HumanEval de modelos locales [50]

SciCode (Coding)

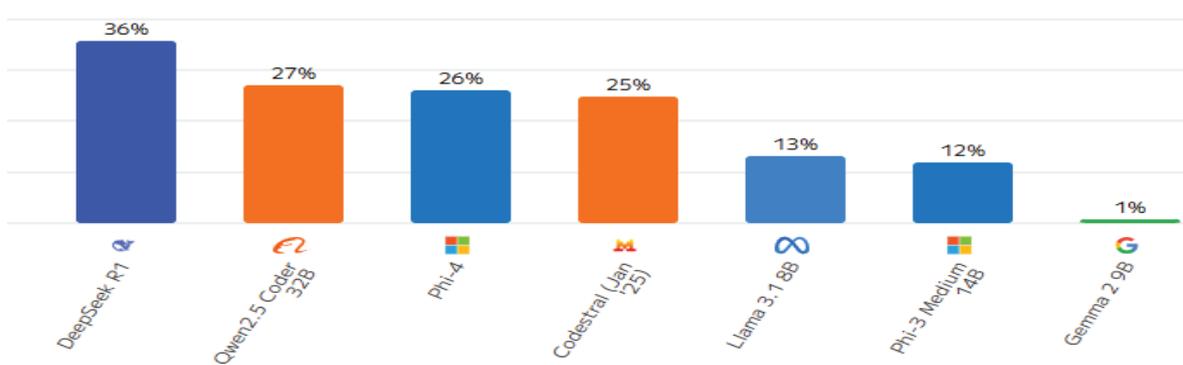


Figura 29. Gráfica Comparativa de SciCode de modelos locales [50]

LiveCodeBench (Coding)

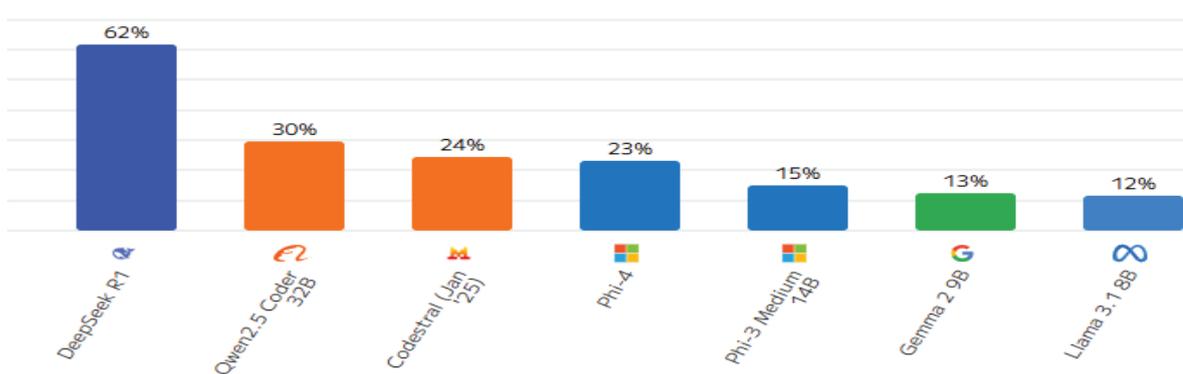


Figura 30. Gráfica Comparativa de LiveCodeBench de modelos locales [50]

3.5 Entornos de ejecución

Para establecer un entorno de ejecución de IA local funcional, se deben considerar varios requisitos esenciales, tanto en términos de hardware como de software [77], [78], [79].

3.5.1 Requisitos de Hardware

- **Unidad Central de Procesamiento (CPU):** Si bien la Unidad de Procesamiento Gráfico es a menudo el componente más crítico para las cargas de trabajo de IA, una CPU sigue siendo esencial para la gestión de las canalizaciones de datos, las tareas de preprocesamiento y la coordinación general de las operaciones del sistema. Se recomiendan procesadores multinúcleo con altas velocidades de reloj para un manejo eficiente de estas tareas.
- **Unidad de Procesamiento Gráfico (GPU):** Una GPU de alto rendimiento es con frecuencia el componente más crucial para ejecutar modelos de lenguaje extenso (LLMs) de manera eficiente a nivel local. Los modelos de IA, especialmente las redes neuronales profundas, dependen en gran medida del procesamiento paralelo, lo que hace que las GPU sean muy adecuadas para acelerar estas computaciones. La cantidad de Memoria de Video (VRAM) es particularmente importante, ya que determina el tamaño de los modelos que se pueden ejecutar sin problemas. Los diferentes tamaños de modelos tienen diferentes requisitos de VRAM; por ejemplo, un modelo con 7 mil millones de parámetros puede requerir entre 8 y 16 GB de VRAM, mientras que los modelos más grandes pueden necesitar 24 GB o más. Las GPU de NVIDIA son a menudo preferidas en la comunidad de IA debido a su amplio soporte para CUDA y su compatibilidad con entornos de aprendizaje profundo populares como PyTorch y TensorFlow.

Tabla 8. Requerimientos de hardware para modelos locales [82]

Tamaño del Modelo LLM	Requisito de VRAM	RAM Recomendada	Casos de Uso Típicos
7B parámetros	8–16 GB	16–32 GB	Prototipado y experimentos
13B parámetros	16–24 GB	32–64 GB	Análisis de texto, generación de contenido
30B parámetros	24–48 GB	64–128 GB	Tareas complejas e investigación avanzada
65B+ parámetros	48+ GB	128+ GB	Modelos de vanguardia

- **Memoria de Acceso Aleatorio (RAM):** Una cantidad suficiente de RAM es esencial para un procesamiento fluido de la IA, especialmente cuando se trabaja con modelos y conjuntos de datos grandes. Para tareas básicas de IA, se sugiere un mínimo de 16 GB de RAM, pero para modelos más grandes y para el ajuste fino (fine-tuning), se

recomienda 32 GB o más. Para un rendimiento óptimo con modelos de tamaño mediano y para el ajuste fino, se considera ideal 64 GB de RAM DDR5.

- Almacenamiento: Las soluciones de almacenamiento rápidas, como las unidades de estado sólido NVMe (SSDs), son cruciales para cargar rápidamente los modelos de IA y manejar grandes conjuntos de datos. Se recomienda un mínimo de 10 GB de espacio libre en disco para la instalación de las herramientas de IA, pero los modelos más grandes y los conjuntos de datos requerirán significativamente más espacio. Para un rendimiento óptimo, se prefieren los SSDs a los discos duros tradicionales (HDDs) debido a sus velocidades de lectura/escritura mucho más rápidas, lo que reduce los tiempos de carga de los modelos y mejora la capacidad de respuesta general del sistema.
- Refrigeración: Las cargas de trabajo de IA de alto rendimiento generan una cantidad significativa de calor. Por lo tanto, es necesario un sistema de enfriamiento robusto para evitar la limitación térmica y garantizar un rendimiento sostenido. Las soluciones de enfriamiento pueden variar desde enfriadores de aire de alta calidad hasta sistemas de enfriamiento líquido más avanzados, según las demandas específicas de la CPU y la GPU utilizadas en el sistema.

3.5.2 Requisitos de Software

Esta sección detalla los componentes de software fundamentales necesarios para implementar y ejecutar sistemas de inteligencia artificial de manera local. A diferencia de las soluciones basadas en la nube, la ejecución local de IA requiere una configuración específica del entorno de software que incluye desde el sistema operativo base hasta las bibliotecas especializadas. Los siguientes requisitos constituyen el ecosistema mínimo para garantizar un funcionamiento óptimo de las aplicaciones de IA en infraestructura propia.

- Sistema Operativo: Los sistemas operativos compatibles para la ejecución local de IA incluyen Windows (se recomienda Windows 10 o posterior con el Subsistema de Windows para Linux 2 - WSL2), macOS (versión 11 o posterior) o una distribución de Linux (Ubuntu es a menudo la preferida debido a su amplio soporte para herramientas y controladores de IA). Es importante asegurarse de que el sistema operativo esté actualizado para evitar problemas de compatibilidad con el software de IA.
- Empaquetación: Docker es una herramienta popular para empaquetar los componentes de IA en entornos aislados llamados contenedores. Esto simplifica la implementación, garantiza la portabilidad entre diferentes sistemas y ayuda a

gestionar las dependencias de software. Docker permite a los investigadores ejecutar diferentes herramientas y modelos de IA sin conflictos entre sus requisitos.

- Bibliotecas de IA: Dependiendo de las tareas específicas de IA que se vayan a realizar, puede ser necesario instalar software adicional. Esto a menudo incluye Python, un lenguaje de programación ampliamente utilizado en el campo de la IA. Para aquellos que utilizan GPUs NVIDIA, es posible que se requiera la instalación de CUDA y cuDNN para permitir que las aplicaciones de IA aprovechen la potencia de procesamiento paralelo de la GPU. Además, los entornos de aprendizaje profundo como PyTorch o TensorFlow son fundamentales para construir, entrenar y ejecutar modelos de IA. Finalmente, bibliotecas como Hugging Face Transformers proporcionan acceso a una vasta colección de modelos pre-entrenados y herramientas para diversas tareas de Procesamiento del Lenguaje Natural (PNL), lo que facilita el trabajo con LLMs a nivel local.

3.5.3 Plataformas para la Ejecución Local de IA

Existe una variedad de plataformas disponibles para la ejecución local de modelos de Inteligencia Artificial, cada una con sus propias fortalezas y debilidades, que se adaptan a diferentes necesidades y niveles de experiencia técnica. Se destacan algunas de las plataformas más importantes:

- Llama.cpp [80]: Un proyecto de código abierto escrito en C/C++ diseñado específicamente para ejecutar modelos LLaMA (Large Language Model Meta AI) de manera eficiente en hardware local. Es conocido por su ligereza y su capacidad para funcionar tanto en CPUs como en GPUs, lo que lo hace accesible incluso para usuarios con recursos computacionales limitados.
- Hugging Face Transformers: Una biblioteca de Python muy popular que proporciona acceso a una vasta colección de modelos pre-entrenados para diversas tareas de procesamiento del lenguaje natural [78]. Si bien Hugging Face también ofrece una plataforma en la nube, su biblioteca Transformers permite a los usuarios descargar y ejecutar modelos localmente, lo que proporciona flexibilidad y control sobre los datos [81].
- ONNX (Open Neural Network Exchange): Un formato estándar para representar modelos de aprendizaje automático, lo que permite la interoperabilidad entre diferentes entornos de trabajo de IA [80]. Esto significa que un modelo entrenado en un entorno como PyTorch se puede convertir al formato ONNX y luego ejecutar utilizando otro programa o incluso en hardware especializado que admita ONNX.

Ejecución de modelos de IA con Interfaz completa integrada

- LM Studio: Esta aplicación proporciona una interfaz gráfica de usuario (GUI) intuitiva que simplifica enormemente el proceso de descubrir, descargar y ejecutar LLMs localmente [80]. LM Studio admite una variedad de formatos de modelos, incluido el formato gguf, y permite a los usuarios interactuar con los modelos a través de una interfaz de chat. También ofrece una función de servidor de inferencia local para desarrolladores que desean integrar LLMs en sus propias aplicaciones [82]. Su facilidad de uso lo convierte en una excelente opción para principiantes y usuarios ocasionales.
- GPT4All: Este es un ecosistema de código abierto impulsado por la comunidad que ofrece una aplicación de escritorio para ejecutar varios modelos de lenguaje de código abierto a nivel local [80]. Proporciona una interfaz sencilla para interactuar con los modelos y es una opción popular entre los entusiastas de la IA y los investigadores que priorizan la privacidad.
- Jan.ai [83]: Similar a LM Studio, Jan.ai es otra aplicación fácil de usar con una GUI que permite a los usuarios ejecutar LLMs localmente . Ofrece funciones como la selección de modelos, la aceleración por GPU (si está disponible) y la capacidad de ampliar la funcionalidad con extensiones. Su interfaz intuitiva y su enfoque en la facilidad de uso lo convierten en una opción atractiva para aquellos que prefieren una experiencia más visual.

Ejecución de modelo IA con Interfaz básica

- Ollama: Esta plataforma está diseñada específicamente para simplificar la implementación y la gestión local de LLMs [84]. Ollama permite a los usuarios descargar y ejecutar modelos con un solo comando a través de su interfaz de línea de comandos (CLI). También proporciona una API (incluida una API compatible con OpenAI) que permite a los desarrolladores integrar fácilmente LLMs en otras aplicaciones y flujos de trabajo. Ollama admite una amplia gama de modelos populares y facilita la creación de modelos personalizados a partir de los existentes [85].
- llamafile [86]: Esta herramienta ofrece una solución única para empaquetar los pesos de los LLMs y las herramientas necesarias para ejecutarlos en archivos ejecutables únicos . Esto hace que los LLMs sean extremadamente accesibles y fáciles de distribuir, ya que se pueden ejecutar con un solo archivo ejecutable. llamafile también proporciona, de forma predeterminada, un punto final compatible con la API de OpenAI, lo que facilita aún más la integración con las herramientas y flujos de trabajo existentes.

Capítulo 4. Evaluación Inicial de los Modelos

Tras la revisión bibliográfica vista en los capítulos anteriores sobre la inteligencia artificial, se procedió a evaluar un grupo de modelos bajo los criterios de la asignatura de Programación web.

4.1 Modelos Elegidos y Plataforma de Ejecución

La selección inicial de modelos para este estudio se ha realizado siguiendo criterios específicos de accesibilidad y capacidad. Se ha dado preferencia a aquellos modelos disponibles tanto a través de interfaces web como en implementaciones locales, excluyendo deliberadamente las extensiones para entornos de programación, ya que estas últimas frecuentemente requieren suscripciones y tienden a generar código que meramente imita o reinterpreta el trabajo del programador original.

Un criterio principal en la selección ha sido la capacidad de razonamiento y resolución de problemas lógicos, consideradas habilidades esenciales para la generación de código. La investigación se ha enfocado principalmente en modelos de alto impacto en la industria, como ChatGPT y el reciente Deepseek, así como en aquellos desarrollados por empresas líderes en el sector. Además, se ha priorizado la inclusión de los modelos más actualizados y accesibles hasta la fecha en el que se realizó la búsqueda bibliográfica.

Justificación de las Herramientas Seleccionadas

Para la implementación de los modelos locales se configuró un entorno siguiendo los requerimientos detallados en el capítulo 3 y cuya guía de instalación y configuración se encuentran en el anexo (A1), a continuación se detallan aspectos de las herramientas usadas:

- **Ollama**: Ofrece un enfoque sencillo para la ejecución de modelos, optimizado para entornos locales y con la posibilidad de interactuar con modelos open-source.
- **Open-WebUI** [87]: Proporciona una interfaz web intuitiva y flexible para la gestión e interacción con modelos de lenguaje, facilitando su integración y personalización.

Ambas herramientas destacan por su comunidad, documentación disponible y licencias abiertas que permiten su uso y extensión. Ollama proporciona un backend robusto y eficiente para servir LLMs localmente, manejando la gestión de modelos y proporcionando una API. OpenWebUI actúa como un frontend intuitivo, aprovechando la API de Ollama para ofrecer una experiencia de usuario perfecta para interactuar con estos modelos.

Las especificaciones del ordenador usado para la ejecución de modelos locales: Ordenador de sobremesa con procesador Intel(R) Core(TM) i7-9700 3.00GHz, tarjeta gráfica MSI NVIDIA GeForce RTX 4090 de 24GB y 64GB de RAM DDR4 (4 módulos de 16GB).

4.1.1 Descripción de los Modelos iniciales seleccionados

En este apartado se revisan y se amplían las características de los modelos vistos anteriormente en los antecedentes y que cumplen los requisitos establecidos en la introducción del capítulo. (Ver tabla 9)

Tabla 9. Modelos iniciales seleccionados

Web	Gratis (Acceso Limitado)	ChatGPT-4o
		Claude Sonnet 3.5
	Pago	OpenAI o1
		OpenAI o3-mini
Local	Gratis (OpenSource)	OpenAI o3-mini-high
		DeepSeek-R1 14B
		Codestral 22B
		Phi-4 14B

ChatGPT-4o

Características Técnicas:

- **Arquitectura:** Basado en la estructura Transformer, una técnica de procesamiento de lenguaje natural que convierte palabras en vectores numéricos para su análisis mediante redes neuronales.
- **Algoritmos Base:** Utiliza algoritmos de redes neuronales artificiales. Estos algoritmos permiten manejar grandes volúmenes de datos y realizar tareas como reconocimiento de imágenes y procesamiento de lenguaje natural.

Capacidades:

- **Multimodalidad:** A diferencia de versiones anteriores, ChatGPT-4o puede procesar y generar no solo texto, sino también imágenes y audio, facilitando interacciones más naturales y versátiles.
- **Mejora en el razonamiento:** Presenta avances en tareas complejas de razonamiento, mostrando un rendimiento superior en áreas como programación competitiva y matemáticas.

OpenAI o1

Características Técnicas:

- **Arquitectura:** Basado en la estructura Transformer, emplea técnicas avanzadas de aprendizaje por refuerzo y un algoritmo de optimización específico. Este

enfoque le permite generar cadenas de pensamiento internas antes de producir una respuesta final, mejorando su capacidad de razonamiento.

- **Algoritmos Base:** Utiliza aprendizaje por refuerzo para refinar sus procesos de pensamiento, explorando diferentes estrategias y corrigiendo errores.

Capacidades:

- **Razonamiento Avanzado:** o1 ha demostrado un rendimiento superior en tareas que requieren razonamiento complejo. Resolviendo el 83% de los problemas en el examen de la Olimpiada Internacional de Matemáticas, superando a modelos anteriores como GPT-4o.

OpenAI o3-mini

Características Técnicas:

- **Arquitectura y Algoritmos Base:** Aunque OpenAI no ha revelado detalles específicos sobre la arquitectura interna de o3-mini, se sabe que emplea técnicas de aprendizaje por refuerzo para mejorar sus habilidades de razonamiento. Permitiendo al modelo planificar y ejecutar pasos intermedios de razonamiento para resolver problemas complejos.
- **Compatibilidad con Funciones y Salidas Estructuradas:** El modelo admite llamadas a funciones y generación de salidas estructuradas, facilitando su integración en diversas aplicaciones y flujos de trabajo automatizados.

Capacidades:

- **Rendimiento en Tareas STEM:** o3-mini ha demostrado un rendimiento superior en evaluaciones de matemáticas, programación y ciencia, igualando o superando a modelos anteriores como o1.

OpenAI o3-mini-high

Características Técnicas:

- **Arquitectura y Algoritmos Base:** Al igual que o3-mini, o3-mini-high se basa en técnicas de aprendizaje por refuerzo y utiliza cadenas de pensamiento para descomponer problemas complejos en pasos intermedios.
- **Niveles de Esfuerzo de Razonamiento:** o3-mini-high representa el nivel más alto de esfuerzo de razonamiento disponible en la serie o3-mini. Diseñado para tareas que requieren un análisis más profundo y detallado, proporcionando respuestas más precisas a costa de un mayor tiempo de procesamiento.

Capacidades:

- **Rendimiento en Tareas STEM:** ha demostrado un rendimiento sobresaliente en evaluaciones de matemáticas, programación y ciencia. En pruebas como AIME

2024, alcanzó una precisión del 83.6%, y en GPQA Diamond, logró un 77.0% de precisión, superando a modelos anteriores como o1 en estas áreas.

- **Mejora en Programación Competitiva:** En tareas de programación competitiva, o3-mini-high obtuvo una puntuación Elo de 2073 en Codeforces, destacando su capacidad para resolver problemas de codificación de alta complejidad.

DeepSeek-R1 14B

Características Técnicas:

- **Arquitectura y Algoritmos Base:** Se basa en una arquitectura LLM y ha sido entrenado utilizando técnicas de aprendizaje por refuerzo. A diferencia de enfoques tradicionales que dependen de datos etiquetados manualmente, R1 emplea un método de auto-verificación, permitiéndole refinar sus habilidades de razonamiento de manera autónoma.
- **Entrenamiento y Eficiencia:** El modelo fue entrenado con aproximadamente 2,000 chips Nvidia H800, con un costo estimado de \$5.6 millones, lo que representa una fracción del coste asociado con modelos similares de otros líderes de la industria.

Capacidades:

- **Rendimiento en Tareas de Razonamiento:** Ha demostrado capacidades avanzadas en tareas de razonamiento lógico, matemáticas y programación, rivalizando con modelos como OpenAI o1 en diversas evaluaciones.
- **Posibilidad de uso local:** Disponible.

Claude 3.5 Sonnet

Características Técnicas:

- **Arquitectura y Algoritmos Base:** Aunque Anthropic no ha divulgado detalles específicos sobre la arquitectura interna, se sabe que emplea técnicas avanzadas de aprendizaje automático para mejorar sus capacidades de razonamiento y comprensión del lenguaje. Siendo entrenado para interpretar y generar contenido de alta calidad, manteniendo un tono natural y coherente.
- **Ventana de Contexto y Eficiencia:** Maneja una ventana de contexto de 200,000 tokens, permitiendo procesar y generar respuestas en contextos extensos sin pérdida de coherencia. Además, opera al doble de velocidad que su predecesor, Claude 3 Opus, con un rendimiento mejorado a una fracción del costo.

Capacidades:

- **Rendimiento en Tareas de Programación:** Ha demostrado una notable mejora en tareas de codificación, alcanzando un 49.0% en el benchmark SWE-bench Verified, superando a modelos anteriores y a otros disponibles públicamente.

Codestral 22B

Características Técnicas:

- **Arquitectura y Algoritmos Base:** Se basa en LLM optimizada para la generación de código. Su entrenamiento abarcó un conjunto de datos diverso que incluye más de 80 lenguajes de programación, permitiéndole comprender y generar código en múltiples entornos.
- **Ventana de Contexto Ampliada:** Una de las características destacadas de Codestral 22B es su ventana de contexto de 32.000 tokens. Permittiéndole manejar tareas de completado de código de largo alcance con mayor eficacia.

Capacidades:

- **Dominio Multilenguaje:** La capacidad de para generar y completar código en más de 80 lenguajes lo hace versátil para diversos proyectos de desarrollo.
- **Mecanismo de Relleno en el Medio (FIM):** Puede predecir tokens intermedios entre un prefijo y un sufijo, facilitando la finalización de segmentos de código incompletos y mejorando la eficiencia en el desarrollo.
- **Posibilidad de uso local:** Disponible.

Phi-4 14B

Características Técnicas:

- **Arquitectura:** Se basa en una arquitectura de transformador de solo decodificador. Esta arquitectura sigue de cerca a la de phi-3-medium, con cambios mínimos.
- **Tokenizador:** Emplea el tokenizador tiktoken, que soporta mejor varios idiomas, con un tamaño de vocabulario acolchado de 100.352.

Capacidades:

- **Enfoque en la Calidad de Datos:** Fue desarrollado con un entrenamiento centrado en la alta calidad de los datos. Incorporaron datos sintéticos de forma estratégica a lo largo del entrenamiento.
- **Datos Sintéticos Innovadores:** Utiliza técnicas innovadoras para generar datos sintéticos, incluyendo multi-agent prompting, flujos de auto-revisión e inversión de instrucciones. Estos métodos buscan inducir habilidades de razonamiento y resolución de problemas más fuertes. Los datos sintéticos constituyen la mayor parte de los datos de entrenamiento.
- **Posibilidad de uso local:** Disponible.

Qwen2.5-Coder 32B

Características Técnicas:

- **Arquitectura y Tamaños:** La arquitectura se deriva directamente de Qwen2.5. La serie consta de varios tamaños: 0.5B, 1.5B, 3B, 7B, 14B y 32B parámetros.
- **Tokenización:** Hereda el vocabulario de Qwen2.5, pero introduce varios tokens especiales para ayudar a comprender mejor el código.
- **Qwen2.5-Coder-Data** es un conjunto de datos de preentrenamiento específico para codificación, compuesto por cinco categorías principales: código fuente de GitHub que abarca 92 lenguajes de programación, datos de text-code grounding curados de Common Crawl mediante un sistema de filtrado jerárquico, datos sintéticos generados y validados por CodeQwen1.5, datos matemáticos del corpus Qwen2.5-Math, y texto general de alta calidad del corpus Qwen2.5 para mantener capacidades lingüísticas generales.

Capacidades:

- **Excelencia general:** Alcanza rendimiento estado del arte en más de 10 benchmarks, superando a modelos del mismo tamaño.
- **Generación de código:** Destaca en HumanEval, MBPP y BigCodeBench-Complete, posicionándose como el modelo de código abierto más potente, aproximándose al rendimiento de GPT-4o.
- **Comprensión de tablas:** El modelo de 32B lidera el benchmark TableBench para responder preguntas sobre datos tabulares.

4.2 Diseño y resultados de las Pruebas Iniciales

Se plantean como pruebas varios escenarios mediante un grupo de prompts iniciales, basados en los documentos a los que puede tener un estudiante de Programación Web a la hora de realizar las prácticas.⁹ Dividiéndose en las pruebas en tipos dos:

- **Exploración Creativa:** son escenarios donde se evalúa la capacidades creativas-innovadoras de los modelos, usarán un número definido de prompts para la creación de elementos y comparará el resultado final.
- **Resolución de Objetivos:** son escenarios que evalúan las capacidades de resolución de problemas y cumplir objetivos, contabilizando el número de prompts necesarios para llegar al resultado deseado.

⁹ En el anexo (A2) se incluyen enlaces repositorios de Github del código generado.

4.2.1 Diseño de las Pruebas Iniciales

Exploración Creativa

Estos escenarios evalúan la creación de soluciones novedosas con prompts fijos, considerando calidad estética, originalidad y adecuación al contexto web.

Prueba 1: Diseño de una Landing Page Innovadora

***Prompt inicial:** Quiero que crees una Landing Page Innovadora para un mercado local, debe crearse con HTML5, CSS3 y JavaScript básico, puedes usar Bootstrap para las animaciones. Debe tener un apartado de presentación, otra sección de puestos y otra sección de productos, todas las secciones deben mostrar datos relacionados con la sección con datos.*

- **Lenguajes y tecnologías esperados:**
 - *Frontend:* HTML5, CSS3, JavaScript básico y Bootstrap.
- **Número de prompts a utilizar tras el inicial: 5**

Prueba 2: Diseño de un Portafolio Personal Automatizado

***Prompt inicial:** Diseña un portafolio personal automatizado con HTML5, CSS3 y JavaScript. Dale un diseño moderno e introduce datos genéricos como ejemplo de lo que se puede poner.*

- **Lenguajes y tecnologías esperados:**
 - *Frontend:* HTML5, CSS3 y JavaScript.
- **Número de prompts a utilizar tras el inicial: 5**

Resolución de Objetivos

Estos escenarios plantean un reto concreto: la solución completa de un problema, midiendo las iteraciones/prompts, eficiencia del código y cumplimiento funcional.

Prueba 3: Implementación de un CRUD con Laravel

***Prompt inicial:** Desarrolla una aplicación web utilizando Laravel que permita gestionar un conjunto de productos mediante las operaciones básicas de un CRUD (Crear, Leer, Actualizar, Eliminar). La aplicación deberá contar con una interfaz intuitiva y responsiva, y conectarse a una base de datos relacional (por ejemplo, MySQL) para almacenar la información de los productos.*

- **Lenguajes y tecnologías esperados:**
 - *Backend:* PHP utilizando el framework Laravel.
 - *Base de datos:* MySQL (u otro sistema relacional).
 - *Frontend:* HTML5, CSS3 y JavaScript para la interfaz de usuario.

Prueba 4: Desarrollo de un Sistema de Chat en Tiempo Real

Prompt inicial: *Desarrolla un sistema de chat utilizando PHP con Laravel en el backend, mientras que en el frontend se implementan interfaces dinámicas y responsivas con HTML, CSS y JavaScript que permitan unirse a salas de chat y enviar mensajes, garantizando una experiencia fluida.*

- **Lenguajes y tecnologías esperados:**
 - *Backend:* PHP con Laravel, con posible integración de servicios para WebSockets.
 - *Frontend:* HTML, CSS y JavaScript para la interfaz en tiempo real.

4.2.2 Resultados de las Pruebas Iniciales

Para la ejecución del código generado se utilizaron las mismas herramientas que las definidas en la asignatura Programación Web. Entre ellas destaca el uso de Laragon como plataforma de los sistemas y servidores web (apache, mysql, php) y Laravel como framework para la construcción de aplicaciones web.

A cada modelo se le suministró el mismo prompt inicial, por cada uno de las pruebas definidas anteriormente. Además se incrementó el número de prompts donde fue necesario hasta asegurar un resultado comparable. Principalmente se usó la interfaz web para interactuar con los modelos, aquellos que carecen de accesibilidad desde el navegador web se empleó en el entorno descrito en el apartado 4.1, *Justificación de las Herramientas Seleccionadas*. En la figura 35 y 36 se pueden ver las páginas webs generadas para las pruebas 1 y 2.

Se usaron las siguientes métricas:

- Calidad y legibilidad del código generado (CyL): 0-10
- Originalidad en el diseño (OD)(uso de paleta de colores, animaciones, etc.): 0-10
- Grado de cumplimiento (GC): 0-10
- Complejidad de diseño (CD): 0-10
- Número de prompts solo para las pruebas de Resolución de objetivos.

Los resultados de las pruebas se recogen en el apéndice A2, y se muestran en las figuras de la 31 a la 34:

Prueba 1

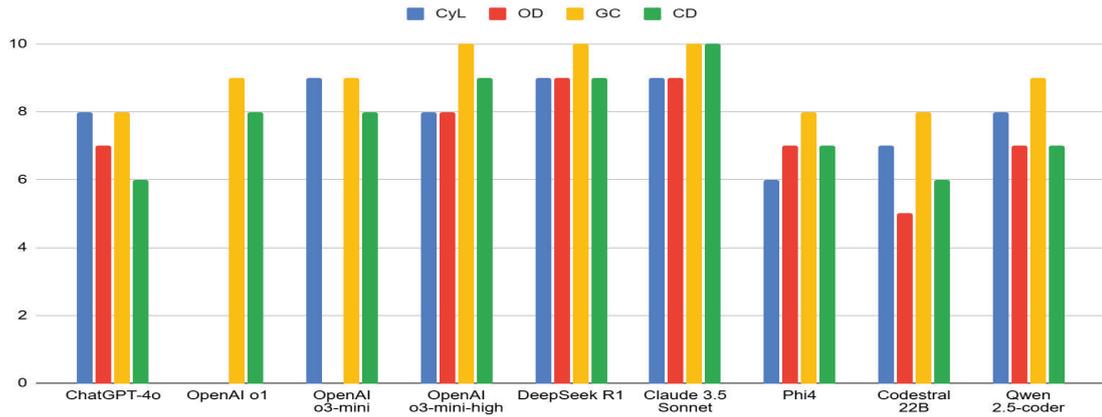


Figura 31. Gráfico de barras de los resultados de la prueba 1. Elaboración propia

Prueba 2

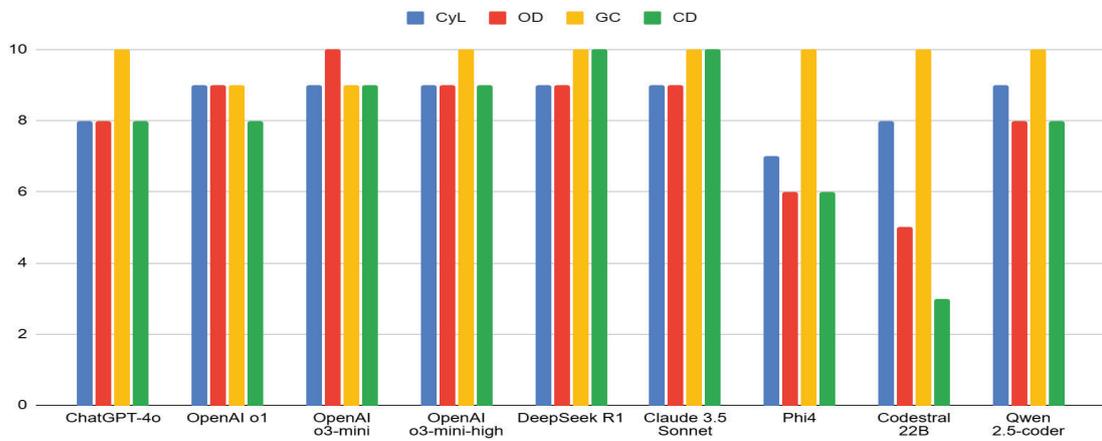


Figura 32. Gráfico de barras de los resultados de la prueba 2. Elaboración propia

Prueba 3

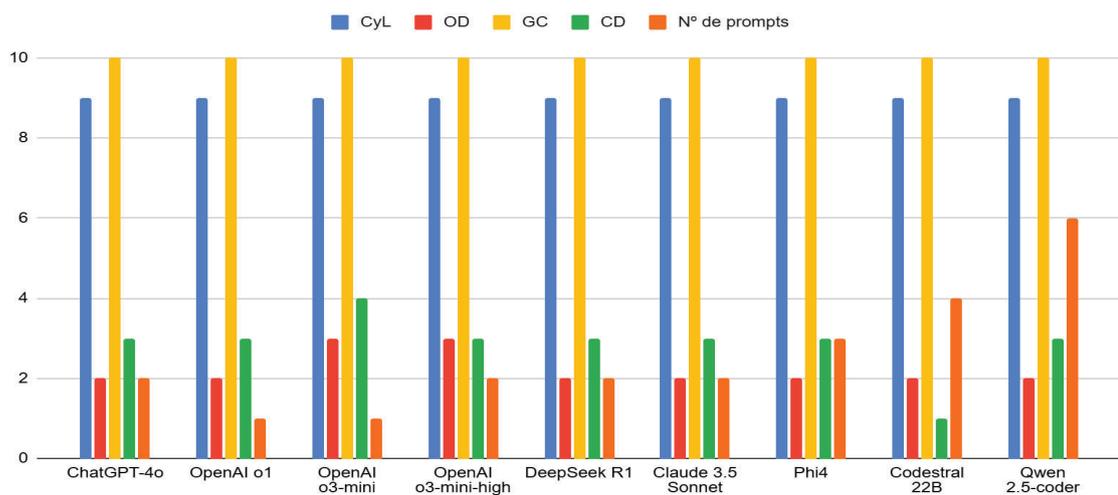


Figura 33. Gráfico de barras de los resultados de la prueba 3. Elaboración propia

Prueba 4

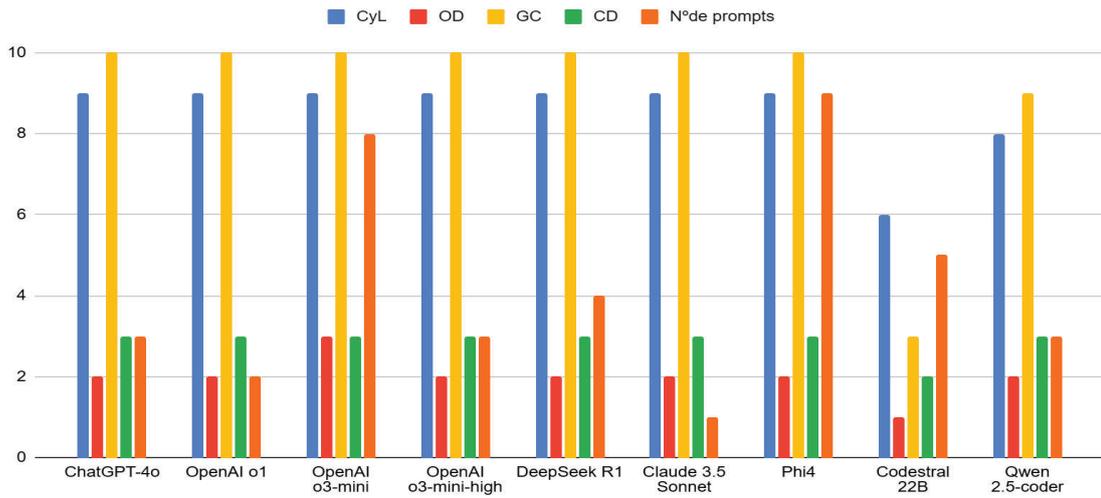


Figura 34. Gráfico de barras de los resultados de la prueba 4. Elaboración propia

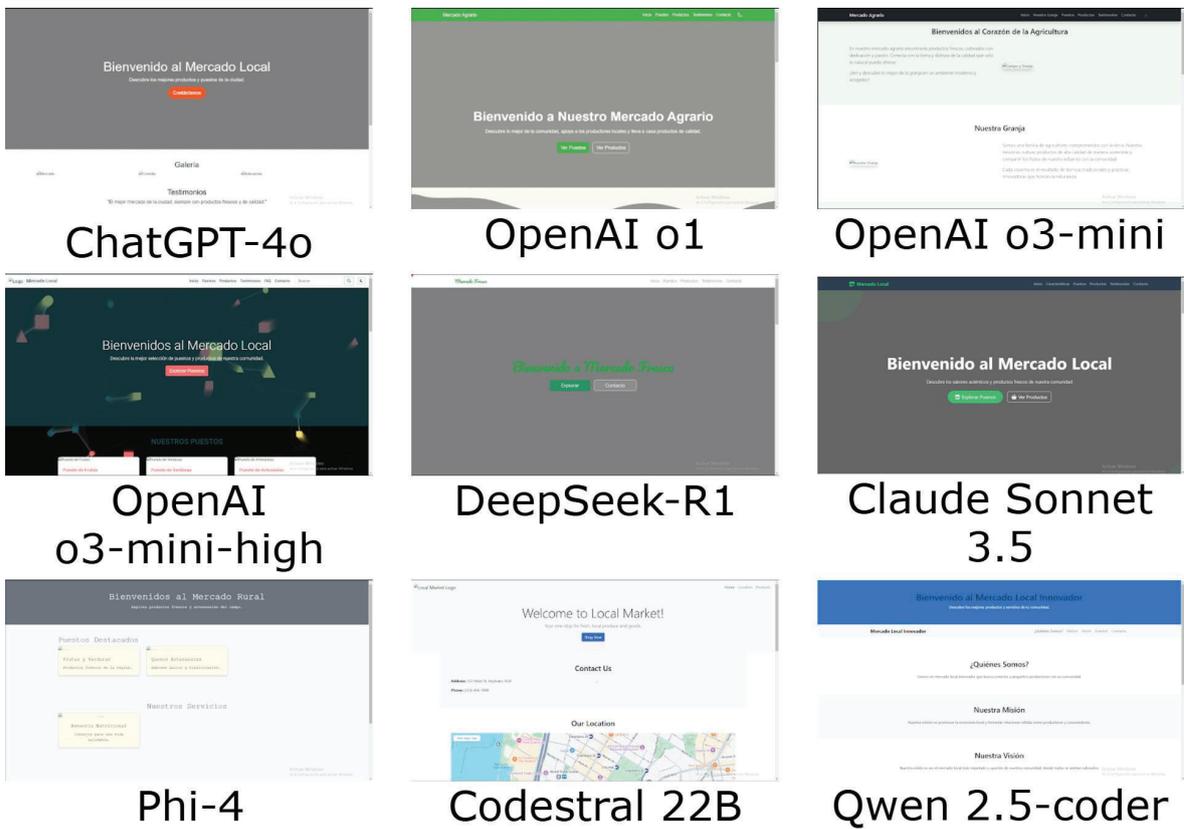


Figura 35. Capturas de los resultados de la prueba 1. Elaboración propia

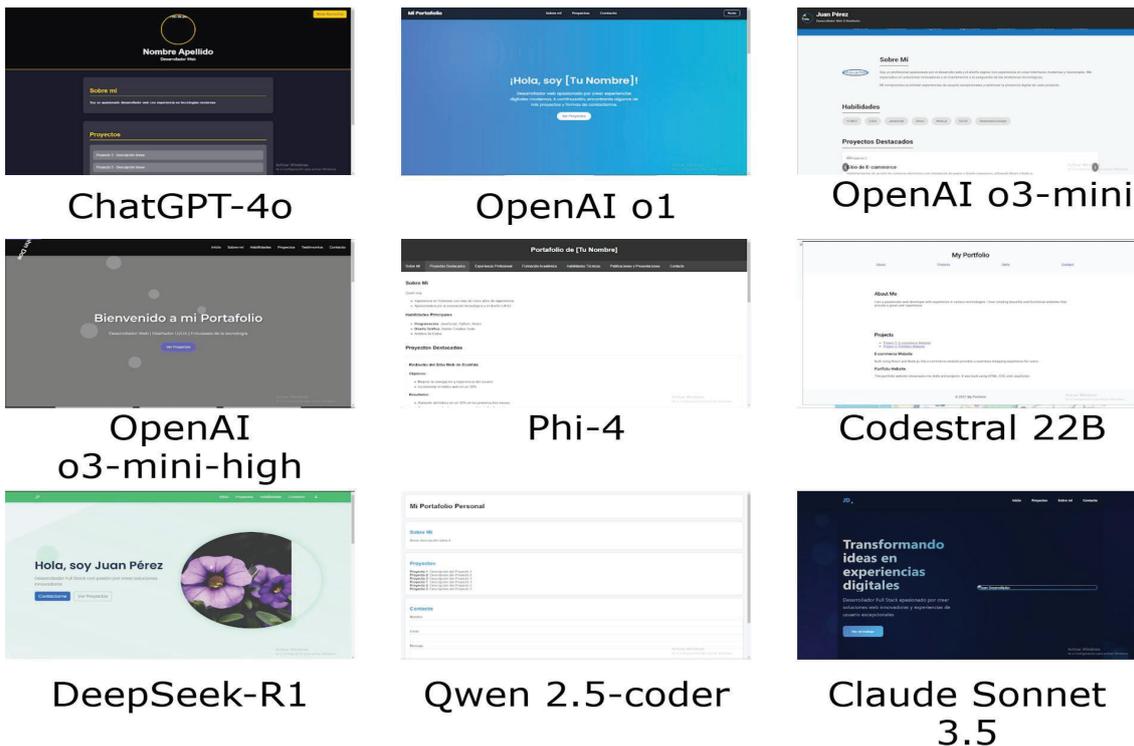


Figura 36. Capturas de los resultados de la prueba 2. Elaboración propia

Análisis de los Resultados

A continuación se presenta una lista de los modelos elegidos en base a los resultados de las pruebas realizadas y sus características:

- **OpenAI o3-mini-high:** Destaca entre los modelos de la compañía por su superior resolución de problemas y diseño de interfaces, entregando soluciones precisas sin elementos innecesarios o añadiendo implementaciones efectivas, superando las limitaciones de su versión base. Accedido mediante web.
- **DeepSeek R1:** Comparable en rendimiento a o3-mini-high, con excelente capacidad para resolver problemas y crear interfaces. Su principal inconveniente es la sobreocupación del servidor en su interfaz web, solucionable mediante implementación local, limitada únicamente por el hardware disponible.
- **Claude Sonnet 3.5:** Aunque carece de capacidades explícitas de razonamiento, iguala o supera a modelos como ChatGPT-4o en ciertas tareas gracias a esta característica, permitiéndole enfocarse precisamente en lo solicitado y ampliar contenido solo cuando se le indica. Accedido mediante web. A fecha de redacción ha sido cambiado a 3.7.
- **Phi4:** Este modelo ha demostrado tener los conocimientos y capacidades necesarias para resolver problemas, siendo su punto fuerte durante las pruebas. Siendo su punto débil la creación de interfaces interesantes. Ejecutado localmente.

- **Qwen 2.5-coder:** En general el modelo ha solucionado todas las pruebas de resolución de problemas, sin embargo no destaca en la creación de interfaces complejas visualmente. Ejecutado localmente.

Durante la fase de pruebas, se identificaron varios aspectos significativos sobre el comportamiento de diferentes modelos de IA:

- GPT-4: Respuestas precisas pero básicas en diseño y frontend, careciendo de creatividad.
- DeepSeek R1: Problemas frecuentes de disponibilidad con mensajes de servidor ocupado.
- Codestral: Buen conocimiento teórico pero dificultades con implementaciones prácticas; mejor como consultor que como implementador.
- Claude 3.7 Sonnet: Resultados atractivos visualmente pero confusos al mostrar solo secciones modificadas del código.
- Modelos locales (Phi-4, Codestral, Qwen): Tienden a generar código desactualizado.
- En pruebas CRUD: Todos siguen pasos similares, diferenciándose en cantidad de código y manejo de posibles problemas.

La primera IA seleccionada para evaluación será Claude, debido a su rendimiento en programación comparado con opciones de pago como o3 y modelos locales como DeepSeek. Además, las características del servicio, la limitación en el tamaño de los chats, facilita la segmentación del código generado y los prompts utilizados, para mejor un análisis estructurado del proceso de desarrollo.

4.3 Métricas análisis de los Modelos

Con este objetivo en mente, se definieron las especificaciones de un proyecto acorde al trabajo que se podría llevar a cabo en las prácticas de la asignatura y la metodología ABP. Las especificaciones están recogidas en el anexo (A3).

El proyecto en cuestión sería un Sistema de Gestión de Contenidos (CMS) orientado a un mercado local en San Mateo, Gran Canaria. El sistema tiene como objetivo principal fomentar la producción y consumo sostenible mediante la promoción de productos artesanales y alimentos de proximidad, la elaboración del proyecto que se dividió en las siguientes fases:

Tabla 10. Plan de Implementación

Fase 1: Configuración y estructura básica	1.1 Configuración del entorno Laravel 1.2 Implementación de migraciones de base de datos 1.3 Creación de modelos y relaciones
Fase 2: Autenticación y perfiles	2.1 Sistema de registro e inicio de sesión 2.2 Gestión de perfiles por rol 2.3 Implementación de middleware de autorización
Fase 3: Gestión de productos	3.1 CRUD de productos para vendedores y proveedores 3.2 Gestión de stock 3.3 Categorización y búsqueda
Fase 4: Tienda virtual	4.1 Implementación del carrito de compra 4.2 Proceso de checkout 4.3 Integración de pasarela de pago
Fase 5: Red social	5.1 Sistema de mensajería privada 5.2 Creación de foros y temas
Fase 6: Integraciones API	6.1 Desarrollo de API interna 6.2 Integración con API de clima 6.3 Implementación de AJAX para actualización dinámica

4.3.1 Métricas aplicadas a los modelos

A continuación se presentan listas de métricas con sus respectivas justificaciones que se aplicaron a los modelos, divididas en aspectos cuantificables de los prompts y respuestas, aspectos arquitectónicos del código generado y un acercamiento a la programación web.

Aspectos cuantificables

Estas métricas se centran en aspectos cuantificables de la generación de código, tanto a nivel de prompt como a nivel de la respuesta.

Tasa de errores de código

Esta medida evalúa la probabilidad de generar código sintácticamente incorrecto o no funcional que requiera revisiones manuales. Se calcula como la proporción de código erróneo (que falla en compilación o ejecución) respecto al total generado.

Fórmula:

$$Tasadeerroresdecódigo(\%) = \left(\frac{NResultados_{Erroneos}}{NResultados_{Total}} \right) * 100$$

Tiempo de Respuesta y Velocidad de Generación

Es crucial evaluar la velocidad de generación de código para determinar su idoneidad en desarrollo en tiempo real o prototipado. El análisis incluye rendimiento promedio, valores mínimos y máximos, destacando diferencias entre modelos locales y basados en web.

Aspectos cualitativos

Estas métricas se centran en aspectos cualitativos de la generación de código, principalmente a nivel de la respuesta.

Coherencia y Consistencia en el Estilo

Un código con nomenclatura, estructura y formato consistentes facilita mantenimiento y comprensión, especialmente en proyectos colaborativos o de largo plazo. Los modelos se evaluarán de 0-10 según estructuración y simplicidad, valorando especialmente aquellos que logran grandes resultados con código simple.

Calidad de la Documentación y Comentarios

La inclusión de comentarios y documentación mejora la legibilidad del código y facilita su integración y evolución, sirviendo como guía para futuros desarrolladores. Los modelos se evaluarán de 0-10 según la inclusión y calidad de comentarios, valorando especialmente aquellos que los introduzcan por iniciativa propia.

Modularidad y Organización del Código

La generación de código modular y organizado permite mayor escalabilidad y reutilización, facilitando adaptación a diversos proyectos o entornos. Los modelos se evaluarán de 0-10 según la organización y modularidad, valorando especialmente aquellos que estructuren el código por apartados/archivos, ya sea por iniciativa propia o invitando al usuario a hacerlo.

Aspectos relacionados con la asignatura

Estás métricas se centran en aspectos más cercanos a la asignatura de Programación web.

Validez y Conformidad del HTML/CSS

Cumplir con los estándares W3C en código HTML y CSS asegura correcta visualización y compatibilidad entre navegadores. Los modelos se evaluarán de 0-10 en el uso de HTML5 y CSS3, valorando el uso adecuado de clases, identificadores y elementos.

Funcionalidad y Manejo de Eventos en JavaScript

Se evaluará de 0-10 la capacidad de cada modelo para crear/integrar código JavaScript que manipule el DOM, gestione eventos y proporcione animaciones e interactividad con gestos del usuario, asegurando experiencias fluidas y dinámicas.

Eficiencia y Seguridad en Consultas SQL

Se evaluará de 0-10 la capacidad de cada modelo para generar consultas SQL optimizadas y seguras (prevención de inyecciones, parametrización) y crear elementos que las utilicen correctamente, garantizando funcionalidad y seguridad en bases de datos para aplicaciones web.

Optimización y buenas prácticas del Front-End

Se evaluará de 0-10 cómo cada modelo implementa prácticas de optimización (minimización, lazy loading) y SEO/accesibilidad (etiquetas meta/alt, etc.) en código

front-end, valorando la reducción de redundancias para mejorar rendimiento, posicionamiento y usabilidad para personas con discapacidades.

Robustez del Código en Lenguajes del Lado del Servidor (PHP, etc.)

Se evaluará de 0-10 cómo cada modelo genera código servidor que implementa buenas prácticas de seguridad y prevención de vulnerabilidades y ataques como CRC.

4.3.2 Métricas aplicadas a los prompts

Se ha creado un método sistemático para evaluar la efectividad de los prompts con categorías como claridad, especificidad, precisión técnica y calidad del código generado. Mediante un sistema de puntuación (1-5) para evaluar diferentes aspectos de los prompts. (Ver tabla 11)

Tabla 11. Sistema de evaluación de prompts

Categoría	Descripción	Criterios de Puntuación
Claridad	Nivel de claridad y ausencia de ambigüedad	1: Altamente ambiguo y confuso 2: Varios elementos poco claros 3: Mayormente claro con alguna ambigüedad 4: Claro con mínima ambigüedad 5: Perfectamente claro y sin ambigüedades
Especificidad	Nivel de detalle y precisión en los requisitos	1: Extremadamente vago 2: Sólo información básica 3: Detalle moderado 4: Requisitos detallados 5: Especificaciones precisas y completas
Precisión Técnica	Uso correcto de términos y conceptos técnicos	1: Errores técnicos graves 2: Varias imprecisiones técnicas 3: Problemas menores técnicos 4: Mayormente preciso 5: Totalmente preciso
Provisión de Contexto	Información de fondo y contexto del proyecto	1: Sin contexto 2: Contexto mínimo 3: Contexto básico relevante 4: Buena información contextual 5: Contexto completo
Calidad de la Estructura	Organización y flujo lógico del prompt	1: Estructura caótica 2: Organización deficiente 3: Estructura adecuada 4: Bien organizado 5: Estructura lógica excelente
Calidad del Código Generado	Calidad del código resultante	1: Código no funcional 2: Funcional pero de baja calidad 3: Calidad aceptable 4: Buena calidad con problemas menores 5: Código excelente y listo para producción

4.4 Requisitos del ABP de la asignatura en prompts

En esta sección se recoge algunos resultados del uso de Claude Sonnet 3.7, identificándose aspectos relacionados con la asignatura y sobre cómo es afectada con el uso de la IA.

En la tabla 12 se recogen todas las fases de desarrollo donde se empleó Claude Sonnet 3.7, a través de su interfaz de chat, mandando distintos prompts según la fase que se estuviera desarrollando adjuntando un documento con las especificaciones del CMS y su base de datos. Como se puede observar se logró implementar todas las funcionalidades excepto la integración de la pasarela de pago.

Tabla 12. Fases implementadas con Claude 3.7 Sonnet

Fase	SubFase	Funcionalidad	Logrado
Fase 1: Configuración y estructura básica	1.1	Configuración del entorno Laravel	Sí
	1.2	Implementación de migraciones de base de datos	Sí
	1.3	Creación de modelos y relaciones	Sí
Fase 2: Autenticación y perfiles	2.1	Sistema de registro e inicio de sesión	Sí
	2.2	Gestión de perfiles por rol	Sí
	2.3	Implementación de middleware de autorización	Sí
Fase 3: Gestión de productos	3.1	CRUD de productos para vendedores y proveedores	Sí
	3.2	Gestión de stock	Sí
	3.3	Categorización y búsqueda	Sí
Fase 4: Tienda virtual	4.1	Implementación del carrito de compra	Sí
	4.2	Proceso de checkout	Sí
	4.3	Integración de pasarela de pago	No
Fase 5: Red social	5.1	Sistema de mensajería privada	Sí
	5.2	Creación de foros y temas	Sí
Fase 6: Integraciones API	6.1	Desarrollo de API interna	Sí
	6.2	Integración con API de clima	Sí
	6.3	Implementación de AJAX para actualización dinámica	Sí

Para la elaboración del CMS se requirió de un periodo aproximado al de una semana y media, en comparativa al periodo de clases, quince semanas, supone un periodo de tiempo bastante reducido. Además de que los problemas que retrasaron el proceso de programación son principalmente por el uso del modelo mediante una suscripción gratuita, lo cual condiciona el tamaño de las conversaciones y obliga a abrir conversaciones del modelo sin el contexto anterior.

4.4.1 El uso de la IA en la asignatura

Con este primer uso de la IA en la asignatura "Programación Web", la cual sigue un enfoque de Aprendizaje Basado en Proyectos (ABP), se puede intuir cómo la inteligencia artificial, ya sea directa o indirectamente, influye en el desarrollo del proyecto y la adquisición de habilidades y conocimientos del alumno, cumpliéndose de forma diferente los objetivos, competencias y habilidades.

La asignatura usa un enfoque metodológico de ABP, se centra en el alumno como parte principal, promoviendo el aprendizaje autónomo y el trabajo en grupo para desarrollar un portal web usando tecnologías (Laravel, CMS, MySQL, etc...). Las prácticas cubren

aspectos sobre: HTML, CSS, bases de datos, PHP, JavaScript, comercio electrónico y redes sociales.

En un primer vistazo, un proyecto realizado principalmente con IA comprometería gravemente las competencias de implementación técnica, diseño detallado, resolución de problemas de código, manejo de tecnologías específicas y redacción técnica autónoma establecidas en la guía docente. Aunque podrían mantenerse habilidades de comunicación, planificación y trabajo en equipo, el principio fundamental del ABP (desarrollo activo y autónomo por parte del estudiante) resultaría alterado significativamente. Estos aspectos se desarrollan con más detalle en el capítulo 5

Capítulo 5. Análisis y Resultados

A partir de la experiencia obtenida con Claude descrita en el apartado 4 del capítulo 4, se procedió a recrear el proyecto con el resto de IAs seleccionadas. Para este conjunto de IAs se inició cada fase con un prompt mejorado que engloba en su contexto toda la información referente que se tuvo que aportar a Claude para realizar la fase correspondiente: partes de código necesario, especificaciones concretas, errores, formato deseado, etc.... Todos los prompts generados se encuentran detallados en el anexo 4.

5.1 Fases y Tareas logradas con los Modelos

La tabla 13 compara la capacidad de cinco modelos: Phi-4 (Phi4), DeepSeek-R1 (DSR1), ChatGPT o3-mini-high (CGPTo3mh), Qwen2.5-coder (QWEN2.5) y Claude Sonnet 3.5 (Claude) para completar con éxito distintas fases (de la Fase 1 a la Fase 6.3) en la tabla 14 se presenta un resumen del número de fases alcanzadas con o sin éxito.

Tabla 13. Fases y subfases donde se empleo los modelos

Fases/IAs	Locales			Web	
	Phi4	DSR1	QWEN2.5	CGPTo3mh	Claude
Fase 1	si	si	si	si	si
Fase 1.1	si	si	si	si	si
Fase 1.2	si	si	si	si	si
Fase 1.3	si	si	si	si	si
Fase 2	si	si	si	si	si
Fase 2.1	si	si	si	si	si
Fase 2.2	si	si	si	si	si
Fase 2.3	si	si	si	si	si
Fase 3	si	si	si	si	si
Fase 3.1	si	si	si	si	si
Fase 3.2	si	si	si	si	si
Fase 3.3	si	si	si	si	si
Fase 4	no	no	no	si	no
Fase 4.1	no	no	si	si	si
Fase 4.2	no	no	si	si	si
Fase 4.3	no	no	no	si	no
Fase 5	no	si	si	si	si
Fase 5.1	no	si	si	si	si
Fase 5.2	si	si	si	si	si
Fase 6	no	no	si	si	si
Fase 6.1	si	si	si	si	si
Fase 6.2	si	no	si	si	si
Fase 6.3	no	no	si	si	si

Comparación directa

Tabla 14. Recuento de fases y subfases no alcanzadas

Modelo	Phi4	DSR1	QWEN2.5	CGPTo3mh	Claude
Fases fallidas	3	2	1	0	1
Subfase fallidas	5	5	1	0	1

Las fases del 1 al 3 no han supuesto problema alguno a los modelos., sin embargo las partes que incluyen servicios externos o mayor uso de javascript supuso una división entre los modelos:

- CGPTo3mh tienen el mejor rendimiento, completando todas las fases.
- Claude y QWEN2.5 fallan solo en una subfase (Fase 4.3).
- DSR1 falla en 5 subfases (Fases 4.1-3 y 6.2-3).
- Phi4 falla en 5 subfases (Fases 4.1-3, 5.1 y 6.3).

Tanto Qwen como Claude tuvieron problemas en implementar la pasarela de pago, mientras que los modelos Phi-4 y DeepSeek fueron incapaces de implementar la Fase 4 entera. Además de fallar en el uso de APIs y en el caso concreto de Phi-4 implementar los foros.

Por tanto ChatGPT o3-mini-high se corona como la opción más confiable a la hora de realizar el proyecto al completar con éxito todas las fases, mientras que los modelos Claude y Qwen se convierten en un alternativa útil. Caso contrario con los casos de DeepSeek y Phi-4 los cuales han demostrado grandes dificultades a la hora de implementar fases más complejas.

5.2 Resultados de los modelos

En este apartado se muestra en la siguiente tabla los resultados promedios obtenidos con las IAs teniendo en cuenta las métricas definidas en el capítulo anterior.

Tabla 15. Estadísticas de los modelos

Categoría	Métrica	Phi4	DSR1	CGPTo3mh	QWEN2.5	Claude
Aspectos Cuantificables (Valor menor es mejor)	Tasa de errores de código (%)	27,42	20,00	23,81	11,90	61
	Tiempo de Respuesta (s)	18,62	39	7	56	4
	Tiempo de Respuesta Mínimo (s)	8	14	3	32	1
	Tiempo de Respuesta Máximo (s)	48	1m 39	16	5m 2	13
Aspectos cualitativos (0-10) (Valor mayor es mejor)	Coherencia y Consistencia en el Estilo	8,5	8	9,17	9	9
	Calidad de la Documentación y Comentarios	6,67	6,17	8	7	4
	Modularidad y Organización del Código	8,67	7,33	9,67	9	9
Aspectos relacionados con la asignatura (0-10) (Valor mayor es mejor)	Validez y Conformidad del HTML/CSS	7,2	8,6	9	8	9
	Funcionalidad y Manejo de Eventos en JavaScript	6,33	6,25	7,75	7,25	7
	Eficiencia y Seguridad en Consultas SQL	7,5	6,6	8,83	8,17	8
	Optimización y Buenas Prácticas del Front-End	6,4	6,17	8,4	7,2	9
	Robustez del Código en Lenguajes del Lado del Servidor (PHP, etc.)	8,33	7,2	9,17	8,5	9

Valoraciones de velocidad y fiabilidad

Tasa de errores

Se muestra contrastes, con Claude como el más problemático (61%) debido principalmente a limitaciones de su servicio al restringir chats, muy por encima de sus competidores; en contraste, GWEN2.5 demuestra solidez con apenas 11,9% de fallos, estableciéndose como referente en estabilidad; el segundo lugar en fiabilidad lo ocupa DSR1 (20%), seguido por CGPTo3mh (23,8%); mientras Phi-4 se sitúa en posición intermedia pero con tendencia problemática (27,4%).

Velocidades promedio

El análisis temporal revela grandes diferencias entre los modelos locales y los accesibles mediante la internet, con Claude destacándose como el más veloz al completar operaciones en solo 4 segundos, seguido por CGPTo3mh con un competitivo tiempo de 7 segundos; ambos establecen un estándar de alta eficiencia que contrasta marcadamente con sus competidores. En nivel intermedio está Phi-4 (18,6 segundos), ofreciendo una respuesta aceptable pero significativamente más lenta, quintuplicando el tiempo de Claude. En el extremo más lento se ubican DSR1 y GWEN2.5, con tiempos considerablemente elevados de 39 y 56 segundos respectivamente, evidenciando un compromiso entre la precisión de GWEN2.5 (con su baja tasa de errores) y la rapidez que caracteriza a los sistemas líderes en velocidad.

Consistencia (mínimos y máximos)

El análisis de estabilidad temporal muestra que Claude destaca no solo por su rapidez absoluta (1-13 segundos), sino especialmente por su notable consistencia, ofreciendo tiempos de respuesta altamente predecibles; similarmente, CGPTo3mh mantiene un rango acotado (3-16 segundos), convirtiéndolo también en opción fiable. Por otro lado, los demás sistemas presentan cambios significativos: Phi-4 muestra variaciones considerables, DSR1 puede extenderse hasta 1:39 minutos en escenarios desfavorables; y el caso más extremo, GWEN2.5, pese a su mínimo de 32 segundos (superior al máximo de Claude), puede alcanzar tiempos extraordinariamente prolongados de hasta 5:21 minutos, revelando una inconsistencia temporal potencialmente problemática en entornos que requieren respuestas rápidas.

Conclusiones prácticas

Según las prioridades, Claude destaca como la mejor opción en velocidad y consistencia temporal, aunque esta eficiencia conlleva un compromiso con su elevada tasa de errores; alternativamente, CGPTo3mh ofrece más equilibrio entre rapidez y precisión media. Para entornos que requieren entre fiabilidad y tiempo de respuesta, Phi-4 emerge como solución versátil, sin destacar pero manteniendo desempeño aceptable en ambas dimensiones. Cuando la precisión es prioritaria sobre la velocidad, GWEN2.5 destaca con su mínima tasa de errores, aunque exige aceptar tiempos significativamente más largos y variables. Finalmente, DSR1 presenta perfil menos competitivo, con fiabilidad moderada y lentitud considerable, limitando su idoneidad a casos específicos donde sus características particulares justifiquen estas desventajas comparativas.

Comparativa entre Modelos

A partir de los puntajes que cada sistema obtuvo en las distintas dimensiones de calidad, podemos extraer estas conclusiones generales:

Rendimiento global

El análisis comparativo demuestra que Claude y CGPTo3mh son los modelos más consistentes y mejor evaluados, con Claude logrando un sobresaliente 9,0 promedio en casi todas las categorías sin bajar de 8 en ningún aspecto, mientras CGPTo3mh mantiene un sólido 8,7 con fortaleza en coherencia y robustez. GWEN2.5 ocupa el tercer lugar con puntuaciones entre 7,0-9,0, mostrando competencia en front y back-end. Por su parte, DSR1 y Ph4 muestran rendimientos más modestos (promedios 6-7), presentando DSR1 debilidades en seguridad y documentación, mientras Ph4 evidencia marcadas inconsistencias, especialmente en funcionalidad JavaScript y desarrollo front-end (6,3).

Puntos fuertes por sistema

El análisis de fortalezas muestra que Claude destaca con puntuaciones excepcionales en coherencia, modularidad, validación, sobresaliendo en optimización front-end y robustez back-end; mientras CGPTo3mh lidera en coherencia y robustez back-end, con excelencia adicional en modularidad y validación HTML/CSS; por su parte, GWEN2.5 presenta un perfil equilibrado con destacando en modularidad y coherencia, manteniendo niveles competentes pero más discretos en funcionalidad JavaScript y documentación.

Áreas de mejora comunes

En el análisis técnico, la funcionalidad JavaScript es un desafío general con CGPTo3mh liderando modestamente (7,75) mientras Phi-4 y DSR1 muestran deficiencias significativas (6,2-6,3); situación similar ocurre en seguridad y consultas SQL, donde CGPTo3mh y GWEN2.5 implementan mejores prácticas (8-8,8) frente a puntuaciones mediocres de Phi-4 y DSR1 (6,6-7,5); en optimización front-end existe clara jerarquía con Claude dominando (9), seguido por GWEN2.5 (7,2), mientras DSR1 y Phi-4 evidencian considerable margen de mejora con puntuaciones preocupantemente bajas (6,1-6,4).

Conclusiones prácticas

En conclusión, Claude emerge como solución óptima para entornos que priorizan cohesión y limpieza del código, ofreciendo el perfil más equilibrado con mínimas deficiencias técnicas; CGPTo3mh representa la elección ideal cuando robustez de back-end y estructuración modular son requisitos críticos; GWEN2.5 se posiciona como alternativa útil y versátil para equipos que requieren rendimiento sólido tanto en front-end como en seguridad, sin alcanzar la excelencia de los líderes; mientras DSR1 y Phi-4 pueden resultar adecuados en contextos donde prime simplicidad o eficiencia

computacional, aunque demandan intervenciones manuales significativas para mejorar su desempeño en JavaScript, seguridad y optimización, haciéndolos menos recomendables para implementaciones exigentes sin refinamiento posterior.

Tipos de Errores Frecuentes

Los modelos de IA presentan fallos comunes en desarrollo Laravel: **rutas y vistas** inexistentes (Claude, CGPTo3mh) o referencias incorrectas (GWEN2.5, Phi4), mientras DSR1 se enfoca en relaciones de datos; **dependencias** problemáticas especialmente en JavaScript (CGPTo3mh, Phi4) e instalación (GWEN2.5); **JavaScript y APIs** con uso incorrecto (CGPTo3mh, DSR1) versus integración genérica (Claude, GWEN2.5); **base de datos y migraciones** deficientes (CGPTo3mh, DSR1) con problemas adicionales de seeders y conexión (Phi4); y **organización del código** con falta de convenciones de directorio y documentación que dificulta el mantenimiento (CGPTo3mh).

5.3 Resultados de la ejecución de los Prompts

Las siguientes tablas muestran los datos de los distintos prompts usados en los modelos después de Claude. Como se puede observar los campos referentes a la asignatura varían al tomarse en cuenta tanto el prompt como la respuesta, debido a esto cada modelo al mismo prompt dio distintos resultados. Este hecho se refleja también en el número de iteraciones, pues cada modelo requirió un número distinto para completar las fases.

En las tablas se incluyen también versiones de los prompts creados a partir del prompt inicial y las iteraciones que se tuvieron que hacer, obteniéndose prompts con mayor contexto y especificación. Con estas versiones se pretende dar a entender la necesidad de la ingeniería de prompt, mostrándose casos donde con menor número de caracteres e iteraciones se pueden obtener resultados iguales o mejores.

Phi-4

Tabla 16. Información de los prompts empleados con Phi-4

Fases	1		2		3		4		5		6	
Tecnología	PHP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL	
Resultados de aprendizaje vinculados	R3, R4, R6, R10, R14, R15		R1-R6, R10, R12, R15		R1, R3-R7, R10-R15		R1, R3, R4, R6, R7, R10, R14, R15		R1-R6, R10-R15		R1-R4	
Competencias relacionadas	CB2, CG5, CT1, CT2, CETM6, CETM7		CB2, CB3, CB5, CG5, CETM1, CETM6, CETM7		CB2-CB4, CG5, CETM1, CETM4, CETM6, CETM7, CT1, CT2		CB3-CB5, CG5, CETM1, CETM6, CETM7, CT1, CT2		CETM1, CETM4, CETM6, CETM7, CT1, CT2, CB3-CB5, CR2, CR3, CG5		CETM1, CETM4, CETM6, CETM7, CG5, CB3-CB5, CT1, CT2,	
Objetivos de la asignatura	OBJ-1-OBJ-3		OBJ-1-OBJ-3		OBJ-1-OBJ-4		OBJ-1-OBJ-3		OBJ-1, OBJ-2, OBJ-4		OBJ-1-OBJ-3	
Iteraciones	1		10		28		1		9		2	
Versión	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1
Long.*	2108	1856	9043	2302	30110	2523	2018	2131	8877	2323	3780	2669
Punt.	5	5	5	5	4.83	5	4.83	5	4.83	5	4.67	5

Qwen 2.5 coder

Tabla 17. Información de los prompts empleados con Qwen 2.5-coder

Fases	1		2		3		4		5		6	
Tecnología	PHP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL	
Resultados de aprendizaje vinculados	R1- R4, R6, R10, R12, R14, R15		R2-R4, R15		R3, R4, R12, R15		R1, R3-R6, R10, R15		R1-R6, R10-R15		R1-R15	
Competencias relacionadas	CETM6,CETM7 CG5, CB4, CB5, CT1, CT2		CETM1, CETM6, CG5, CB4, CT2		CETM-6, CETM-7, CG-5, CB-2, CB-5, CT-1, CT-2		CETM6, CETM7, CB2, CB3, CB5, CG5, CT1, CT2		CETM1, CETM4, CETM6, CETM7, CG5: CB3-CB5, CT1, CT2: CR2, CR3		CETM-1,CETM-4, CETM-6, CETM-7, CG-5, CB-3-CB-5, CT-1-CT-3	
Objetivos de la asignatura	OBJ-1-OBJ-3		OBJ-1-OBJ-3		OBJ-1-OBJ-3		OBJ-2, OBJ-4, OBJ-5		OBJ-1, OBJ-2, OBJ-4		OBJ-1, OBJ-2, OBJ-4, OBJ-5	
Iteraciones	2		5		5		13		4			
Versión	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1
Long.*	1322	1444	8511	1896	8344	2468	10028	1810	6593	3153	24323	1642
Punt.	5	5	4,83	5	5	5	4,5	5	4,83	5	4,3	5

DeepSeek-R1

Tabla 18. Información de los prompts empleados con DeepSeek-R1

Fases	1		2		3		4		5		6	
Tecnología	PHP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL	
Resultados de aprendizaje vinculados	R2, R4, R6, R12, R15		R1, R3-R6, R11, R12, R14, R15		R3, R4, R15		R3, R4, R14, R15		R2-R4, R12, R15		R1-R6, R15	
Competencias relacionadas	CETM-1,CETM-6, CB-2, CB-3, CG-1, CG-5		CETM-1,CETM-4, CETM-6,CETM-7, CG-5, CR-2,CR-3, CT-1, CT-2		CETM1, CETM7, CG5, CB5		CETM-6, CETM-7, CG-5, CB-2, CT-1		CETM1, CETM6, CETM7, CT1, CB5		CETM1, CETM4, CETM6, CETM7, CG5, CB2, CB4, CB5, CT1-CT3	
Objetivos de la asignatura	OBJ-2, OBJ-3		OBJ-1-OBJ-3		OBJ-1, OBJ-2		OBJ-1-OBJ-3		OBJ-1, OBJ-2		OBJ-1-OBJ-3	
Iteraciones	21		11		19		3		16		4	
Versión	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1
Long.*	19865	2243	7378	2319	28090	1725	9562	2457	10972	2084	3875	1990
Punt.	3,67	5	5	5	4	4,83	5	5	4	5	4,83	5

Chat GPT o3-mini-high

Tabla 19. Información de los prompts empleados con Chat GPT o3-mini-high

Fases	1		2		3		4		5		6	
Tecnología	PHP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP/SQL		HTML/CSS/JS/P HP	
Resultados de aprendizaje vinculados	R1-R6,R10-R12, R14, R15		R1-R6, R8, R10-R15		R1-R15		R1, R3-R7, R10-R15		R1-R6, R8, R10-R12, R14, R15		R2-R4, R15	
Competencias relacionadas	CETM1,CETM4, CETM6,CETM7, CG5, CB2, CB3, CB5, CT1-CT3		CETM-1,CETM-4, CETM-6,CETM-7, CG-5, CB-2-CB-5, CT-1, CT-2		CETM-1,CETM-4, CETM-6,CETM-7, CG-5, CB-4,CB-5, CT-1, CT-2, CR-2, CR-3		CETM-1,CETM-6, CETM-7, CG-5, CB-2, CB-4, CT-1, CT-2		CETM-1,CETM-4, CETM-6,CETM-7, CG-5, CR-2, CR-3, CB-3-CB-5, CT-1, CT-2		CETM-1,CETM-4, CETM-6, CETM-7	
Objetivos de la asignatura	OBJ-1-OBJ-4		OBJ-1-OBJ-4		OBJ-1-OBJ-3		OBJ-1-OBJ-3		OBJ-1, OBJ-2, OBJ-4		OBJ-1-OBJ-3	
Iteraciones	0		6		9		17		1		4	
Versión	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1	V1	V1.1
Long.*	1199	2002	1952	2860	4923	1703	9027	2982	2168	2247	3805	2643
Punt.	5	5	5	5	5	5	5	5	5	5	5	5

5.3.1 Estrategias identificadas

En las tablas anteriores (tablas de la 16 a la 19) se puede observar la cantidad de caracteres de los prompts. En el caso de la versión V1 incluye las iteraciones, entendiéndose por iteraciones los prompts que se tuvieron que hacer seguido al prompt inicial para alcanzar el objetivo deseado. Con éstos se han redactado prompts con toda la información clave que se tuvo que dar en las iteraciones, con el objetivo de obtener prompts completos que eviten la necesidad de más consultas al modelo.

Las estrategias presentadas son fundamentales para la ingeniería de prompt en el desarrollo de software, ayudando a guiar a los LLMs para generar código de alta calidad y bien estructurado. Estas técnicas, desde la organización jerárquica hasta la especificación detallada y el énfasis en buenas prácticas, permiten formular prompts que imparten principios de diseño y calidad a la IA. A continuación, se detallan las estrategias identificadas durante el desarrollo con diversas IAs:

1. Estructura Jerárquica y Modular

El formateo efectivo de prompts consiste en estructurar el contenido utilizando encabezados, separadores (####, ---), fases numeradas y secciones delimitadas con títulos descriptivos para organizar la información de manera clara. Esta estructuración incluye el uso de numeraciones (1, 2, 3) para bloques principales, sublistas con viñetas o letras para requisitos secundarios, y la organización modular donde cada fase o funcionalidad tiene su propio objetivo, requerimientos y entregables específicos, facilitando así una comprensión y seguimiento ordenado del contenido.

2. Contextualización del Proyecto y Objetivos

El diseño estructurado de prompts efectivos integra una descripción del dominio funcional (mercado, roles, flujos de interacción) estratégicamente posicionada para enmarcar todas las tareas, incorpora secciones introductorias como "Contexto" y "Objetivo" antes de los detalles técnicos, refuerza la meta general mediante su repetición al inicio y final, define claramente los límites del proyecto especificando qué elementos están dentro y fuera del alcance, y explicita el contexto de dominio junto con la tipología de usuarios involucrados para garantizar una comprensión integral del ecosistema, todo ello para situar al modelo en el escenario adecuado y alinear sus respuestas con los objetivos específicos.

3.Especificidad en el Formato y Detalle Técnico

Los prompts efectivos para desarrollo incluyen indicaciones exactas sobre qué generar (comandos o archivos específicos), especifican convenciones de nombres y estilos de codificación preferidos, solicitan código funcional sin explicaciones innecesarias, indican versiones tecnológicas precisas, utilizan la sintaxis real del entorno específico para eliminar ambigüedades, alternan entre guía textual e indicaciones de consola, incorporan fragmentos de código organizados, entregan ejemplos concretos como "micro-instrucciones", e incluyen bloques de código en Markdown que sirven como plantillas literales para elementos como rutas, relaciones o estructuras de tabla, todo ello proporcionando una guía clara y ejecutable para los desarrolladores.

4.Separación de Responsabilidades y Capas

La estructuración eficaz de prompts para desarrollo web establece una separación clara entre el backend (modelo de datos, lógica de negocio) y el frontend (Blade, AJAX, scripts), implementa una división sistemática por capas y componentes arquitectónicos (Modelos, Controladores, Vistas, etc.), aborda el tratamiento de entidades en capas separadas y bien definidas para mantener la coherencia del sistema, y enfatiza la adherencia al patrón MVC respetando meticulosamente la distinción entre datos, lógica de negocio y presentación, lo que facilita la mantenibilidad, escalabilidad y comprensión del código a lo largo del desarrollo del proyecto.

5.Enfoque en Buenas Prácticas y Convenciones

Los prompts efectivos para desarrollo Laravel deben incorporar las convenciones específicas del framework (nomenclatura estandarizada, softDeletes, relaciones constrained), patrones de diseño fundamentales (MVC, middleware, eventos/Listeners, principios SOLID), convenciones de nombres precisas (snake_case para tablas, PascalCase para modelos), optimizaciones técnicas clave (caché, manejo de errores, logs, minimización de AJAX) y enfatizar consistentemente la importancia de mantener código limpio y mantenible que facilite la escalabilidad y colaboración del proyecto.

6.Validación y Calidad del Software

Los prompts completos de desarrollo Laravel deben incorporar validaciones robustas del servidor (con `$request->validate()`), protección CSRF y restricciones específicas como `stock ≥ 0`), exigir la creación de seeders y factories para datos de prueba, solicitar pruebas unitarias y de rendimiento junto con documentación técnica completa, promover arquitectura RESTful con uso eficiente de Eloquent y separación adecuada de capas, establecer criterios de evaluación específicos (correctitud, eficiencia, claridad, profesionalismo) y enfatizar desde el inicio la importancia de implementar buenas

prácticas, validaciones exhaustivas y testing completo para garantizar la calidad integral del código y facilitar el desarrollo colaborativo.

7. Definición de entregables por etapas del desarrollo

Los prompts efectivos para proyectos Laravel deben incluir terminología precisa sobre entregables específicos (migraciones, modelos, controladores, vistas, etc), proporcionar listas detalladas con nombres exactos de archivos requeridos, incorporar secciones explícitas como "Archivos Esperados" que funcionen como checklist verificable, especificar todos los ficheros a producir (README.md, base de datos SQLite, seeders, etc) y definir claramente el formato de entrega incluyendo rutas con nombre adecuado, capturas en README, ejemplos funcionales de datos de prueba y otros elementos que aseguren la completitud y calidad del proyecto final.

8. Lenguaje Imperativo y Directo

Los prompts efectivos para desarrollo con asistentes de IA utilizan frases directivas concretas ("Quiero que configures...", "Genera...", "Crea...") con oraciones cortas centradas en resultados específicos, incluyen instrucciones explícitas como "Evita errores comunes" o "Si hay ambigüedades, solicita aclaración" para transferir responsabilidad de calidad al asistente, solicitan explicaciones breves del propósito de cada paso (no solo código sino el "porqué"), y requieren análisis comparativos con justificaciones técnicas cuando existen múltiples enfoques posibles (Livewire vs Inertia vs Blade, Stripe vs Redsys), lo que maximiza la precisión y conduce a soluciones más informadas y adaptadas al contexto específico del proyecto.

9. Uso de Elementos Visuales e Iconografía

Los prompts profesionales y visualmente efectivos incorporan emoticonos estratégicamente posicionados (, , , ) para marcar bloques principales de contenido y facilitar la navegación rápida, implementan caracteres visuales específicos (, , , ) como puntos de referencia identificables para categorizar temas, utilizan emojis como "anclas" visuales intuitivas ( para objetivos,  para requisitos,  para secciones clave) que mejoran la comprensión conceptual, y complementan estos elementos con el uso juicioso de negritas y mayúsculas en títulos, creando una jerarquía visual clara que organiza la información y mejora significativamente la usabilidad del documento técnico.

10. Cobertura Completa del Ciclo de Vida y Progresión

Los prompts de desarrollo efectivos implementan reutilización estratégica de patrones entre fases del proyecto para profundizar progresivamente en distintos módulos

funcionales (carrito de compras, características sociales, integraciones API/AJAX), establecen scaffolding progresivo que construye el sistema incrementalmente desde componentes generales (autenticación) hacia funcionalidades específicas (módulos sociales, optimización algorítmica), y consistentemente emparejan descripción contextual del negocio con tareas técnicas precisas (Contexto + Tarea), proporcionando tanto visión global del proyecto como instrucciones específicas para cada componente, facilitando implementación coherente y alineada con objetivos del negocio.

11.Control de Acceso y Autorización Basado en Roles

Los prompts completos para desarrollo de sistemas multi-usuario definen claramente roles del sistema (cliente, vendedor, proveedor) con sus reglas de negocio específicas, incluyen instrucciones precisas para implementar middleware de autorización (CheckRole) y protección de rutas según permisos diferenciados, especifican políticas de autorización necesarias (ProductPolicy) para regular acceso a recursos, detallan controles de visibilidad específicos para cada tipo de usuario y configuran redirecciones post-login personalizadas según rol, garantizando una arquitectura de seguridad robusta y experiencia de usuario coherente con las responsabilidades y privilegios de cada perfil del sistema.

12.Integración Frontend–Backend

Los prompts técnicos detallados para desarrollo web especifican las tecnologías frontend requeridas (Blade + Bootstrap + jQuery/AJAX en Laravel), proporcionan indicaciones precisas sobre implementación de JavaScript, gestión de spinners y manejo visual de errores para mejorar UX, establecen directrices para diseño de rutas RESTful, consumo eficiente de APIs internas/externas, estrategias de caché y técnicas de despliegue óptimo de datos dinámicos, creando una base completa para construir interfaces responsivas y funcionales perfectamente integradas con la arquitectura backend.

13.Flexibilidad con Aspectos Opcionales

Los prompts bien estructurados para desarrollo incorporan secciones claramente definidas de funcionalidad opcional (implementaciones AJAX, mejoras UX) que permiten flexibilidad en el alcance del proyecto, incluyen menciones estratégicas de potenciales mejoras adicionales (políticas de datos, opciones de desistimiento, diseño responsive) que enriquecen la solución, y ofrecen apartados específicos etiquetados como "Extras", permitiendo implementación escalable ajustable tanto a restricciones de tiempo como a objetivos de excelencia técnica según las circunstancias particulares del proyecto.

5.3.2 Alineación con el proceso docente de la asignatura de Programación Web

En este apartado se presentan aspectos sobre cómo el uso de la IA afecta a elementos del proyecto docente como son los objetivos docentes, los resultados de aprendizaje y las competencias, además de un balance sobre su inclusión (Tabla 24).

Aspectos afectados

Los **objetivos docentes** se ven afectados si la IA actúa como creadora principal en lugar de asistente:

- El aprendizaje de tecnologías web perdería la experiencia práctica.
- El desarrollo completo de sistemas con base de datos reduciría la responsabilidad del estudiante.
- La familiarización con herramientas de e-commerce sería limitada sin implementación directa.
- La comprensión de conceptos para redes sociales disminuiría sin interacción práctica con herramientas de blogs y muros.

En todos los casos, el nivel de impacto depende del grado de dependencia en la IA versus su uso como apoyo al aprendizaje.

Los **resultados de aprendizaje** se verían comprometidos por el uso excesivo de IA:

- La comprensión de relaciones entre elementos web disminuiría por la generación automática.
- La interiorización de arquitecturas como MVC se reduciría.
- el dominio práctico de tecnologías, lenguajes y protocolos web sería menor.
- Las habilidades para diseñar e implementar servicios web no se desarrollarían adecuadamente.
- La comunicación escrita y oral perdería autenticidad cuando el estudiante presente trabajo no realizado personalmente.
- La interacción con documentación técnica en inglés disminuiría si la IA la sintetiza; y, fundamentalmente, la metodología de Aprendizaje Basado en Proyectos (ABP) que promueve la autonomía se vería contradicha por la dependencia en la generación automatizada.

Las **competencias** se verían afectadas significativamente por el uso intensivo de IA:

- Las competencias básicas (CB3-CB5) de resolución de problemas.
- La gestión de información y aplicación de conocimientos se comprometerían cuando la IA asume los desafíos técnicos de programación web.
- La comunicación (CG5) perdería autenticidad, especialmente en el componente escrito.
- Las competencias transversales (CT1-CT2) de trabajo en equipo se cambiaría, mientras que la búsqueda de información y desarrollo del espíritu crítico se limitarían al reducirse la experiencia directa
- Las competencias de resolución de problemas de ingeniería telemática (CR1-CR3) disminuirían al delegar el diseño e implementación.
- Las competencias técnicas específicas de Telemática (CETM1, CETM4, CETM6, CETM7) no se adquirirían correctamente al sustituir la experiencia práctica directa requerida por la metodología de ABP.

Tabla 20. Diagrama DAFO – Inclusión de IA

Fortalezas (Interno +)	Debilidades (Interno -)
<ul style="list-style-type: none"> • Simula entornos profesionales actuales, aumentando la empleabilidad. • Ahorra tiempo en tareas repetitivas, permitiendo centrar la docencia en arquitectura, diseño y pruebas. • Ofrece feedback personalizado 24/7 que refuerza el aprendizaje autónomo (ABP). 	<ul style="list-style-type: none"> • Requiere rediseñar temario, rúbricas y sistemas de evaluación para garantizar la adquisición de competencias. • Riesgo de dependencia excesiva y menor asimilación de fundamentos si no se guía adecuadamente. • Posibles sesgos o alucinaciones; demanda supervisión
Oportunidades (Externo +)	Amenazas (Externo -)
<ul style="list-style-type: none"> • Proliferación de herramientas educativas y APIs que pueden integrarse con coste bajo. • Acceso a fondos y convocatorias para innovación docente y transformación digital. • Fomenta interdisciplina y abre nuevas líneas de investigación para TFG/TFM. 	<ul style="list-style-type: none"> • Evolución vertiginosa de las herramientas que puede dejar obsoletos contenidos rápidamente. • Futuras restricciones regulatorias o normativas institucionales sobre uso de IA. • Riesgo de plagio masivo y pérdida de credibilidad si no se controlan las autorías.

5.3.3 Inclusión de la generación de prompts en la asignatura

Como se puede observar en las tablas de la sección 5.3 (tablas de la 20 a la 23), se evaluaron los distintos prompts y sus versiones mejoradas respecto su afinidad con los objetivos, resultados de aprendizaje y competencias de la asignatura. Con esto se puede identificar qué aspectos de la evaluación y objetivos son fomentados mediante la redacción de los prompts.

Que un alumno sea capaz de elaborar prompts siguiendo estas estrategias demuestra una serie de resultados de aprendizaje, contribuye al logro de ciertos objetivos de la asignatura y evidencia el desarrollo de diversas competencias.

Contribución en los Resultados de Aprendizaje:

- **R1: Comprende la relación entre los elementos estudiados.** Al estructurar prompts de forma jerárquica y modular, y contextualizar el proyecto definiendo su

dominio funcional y objetivos, el alumno demuestra comprender cómo se interrelacionan las distintas partes de un sistema web y sus requisitos.

- **R2: Conoce diferentes arquitecturas o modelos de software de comunicación.** Aunque no es el foco principal, al especificar tecnologías como Laravel (que sigue el patrón MVC) e interactuar con bases de datos y, potencialmente, servicios externos, el alumno aplica implícitamente conocimientos sobre arquitecturas de software.
- **R3: Conoce, distingue y utiliza diferentes tecnologías, lenguajes, protocolos y estándares propios de la computación web.** La especificidad en el detalle técnico de los prompts, indicando lenguajes, frameworks y versiones específicas, demuestra un conocimiento y manejo de estas tecnologías fundamentales en programación web.
- **R4: Diseña, implementa, válida y ofrece servicios web.** El proceso completo de guiar a la IA a través de prompts para desarrollar las fases del proyecto, definiendo requisitos, solicitando código funcional, validaciones y pruebas, se alinea directamente con la capacidad de diseñar e implementar servicios web.
- **R5: Muestra interés por los recursos que ofrece la programación de las telecomunicaciones.** El uso proactivo de herramientas de IA para la generación de código como recurso en el proceso de desarrollo web demuestra interés en las capacidades que ofrecen estas tecnologías para potenciar la programación.
- **R6: Comunica de forma escrita las soluciones de los problemas que se plantean en teoría.** La redacción de prompts detallados, claros y estructurados para guiar a la IA sobre cómo generar código funcional es una forma de comunicar por escrito una solución técnica a un problema de programación. El uso de lenguaje imperativo y directo, así como elementos visuales, mejora la efectividad de esta comunicación.
- **R8: Busca y comprende manuales, artículos y especificaciones en inglés sobre la materia.** Para poder redactar prompts con la precisión técnica requerida, especificando convenciones, versiones y sintaxis, el alumno debe haber buscado y comprendido información técnica relevante, a menudo disponible en inglés.
- **R10: Redacta una memoria sobre conceptos de la materia.** Aunque el prompt no es una memoria en sí, la capacidad de estructurar información de manera lógica y organizada, como se hace en los prompts jerárquicos y modulares, es una habilidad transferible y necesaria para redactar documentos técnicos como memorias.
- **R11: Identifica los objetivos del grupo de trabajo y las responsabilidades de cada miembro, asumiendo su compromiso con la tarea asignada.** Si bien este

resultado se enfoca en trabajo en grupo, la capacidad de definir claramente los objetivos y entregables de cada "fase" del proyecto en los prompts puede verse como una analogía a definir tareas y responsabilidades.

- **R12: Utiliza los recursos disponibles para buscar la información necesaria. Valora la propiedad intelectual y cita adecuadamente las fuentes.** Similar a R8, la preparación para escribir prompts técnicos implica el uso de recursos. Aunque las fuentes no se citan *en el prompt*, el contexto del proyecto completo probablemente sí lo requiere.
- **R14: Realiza las tareas encomendadas por el profesor en tiempo y forma.** La elaboración de prompts efectivos como parte de un proyecto es una tarea asignada que se debe completar para avanzar en el desarrollo.
- **R15: Aplica las competencias adquiridas en la realización de una tarea de forma autónoma.** Diseñar y refinar prompts de manera independiente para lograr que la IA genere el código deseado es un ejemplo claro de aplicar competencias de forma autónoma en la resolución de problemas de programación.

Contribución a los Objetivos de la Asignatura:

- **OBJ-1: Que los estudiantes aprendan las distintas tecnologías web, tanto del lado del cliente como del servidor.** Al especificar en los prompts el uso de HTML, CSS, JavaScript, PHP y SQL, así como frameworks asociados, el alumno consolida su conocimiento sobre estas tecnologías al aplicarlas en un contexto práctico (instruyendo a la IA).
- **OBJ-2: Que los estudiantes desarrollen un proyecto de diseño e implementación de un sistema web completo con acceso a base de datos.** La elaboración de prompts para cada fase del desarrollo del CMS, incluyendo la definición de la base de datos y su uso, contribuye directamente a la experiencia en el diseño e implementación de un sistema web completo, aunque la codificación la realice la IA. La habilidad recae en guiar el proceso.
- **OBJ-3: Que los estudiantes conozcan distintas herramientas que permiten el desarrollo de una tienda de comercio electrónico.** Similar al OBJ-2,. la elaboración de prompts para estas funcionalidades familiariza al alumno con los requisitos de tales sistemas.
- **OBJ-4: Que los estudiantes conozcan los conceptos y herramientas para el desarrollo básico de redes sociales.** Similar a OBJ-3, la elaboración de prompts para estas funcionalidades expone al alumno a conceptos relevantes para redes sociales.

- **OBJ-5: Que los estudiantes aprendan los conceptos básicos de Web semántica.** En la elaboración de prompts para las partes o el proyecto, implican o podrían implicar conceptos de Web semántica.

Contribución a las Competencias:

- **CB-2: Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.** Utilizar la IA como herramienta de forma profesional, estructurando el trabajo a través de prompts efectivos y resolviendo los problemas que surgen, demuestra la aplicación profesional de conocimientos de programación web.
- **CB-3: Que los estudiantes tengan la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética.** El proceso de evaluar la calidad del código generado, analizar los resultados, e identificar peculiaridades en el comportamiento de los modelos requiere reunir e interpretar los "datos" proporcionados por la IA y emitir juicios sobre su validez y calidad.
- **CB-4: Que los estudiantes puedan transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado.** La redacción de prompts claros, específicos y bien estructurados es un ejercicio de transmisión de información técnica y de requisitos de manera efectiva a un "público" (IA) que requiere precisión para generar los resultados esperados.
- **CB-5: Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.** La experimentación, el análisis de resultados, la identificación de estrategias efectivas para interactuar con la IA y la adaptación ante los desafíos fomentan el desarrollo de habilidades de aprendizaje autónomo.
- **Otras Competencias (CG5, CT1, CT2, CR2, CR3, CETM1, CETM4, CETM6, CETM7).** Estas competencias también están vinculadas a los prompts y fases del proyecto en las tablas. El proceso de planificación, ejecución y evaluación del proyecto (guiado por prompts) involucra habilidades generales, transversales y específicas del ámbito de la telecomunicación/ingeniería y aspectos éticos/regulatorios relacionados con la IA.

Al redactar prompts efectivos, un alumno no solo adquiere una habilidad técnica específica para interactuar con IA, sino que también demuestra y refuerza su conocimiento de las tecnologías web, su capacidad para diseñar y gestionar proyectos de software, sus habilidades de comunicación técnica, su pensamiento crítico al evaluar resultados y su capacidad de aprendizaje autónomo, todo ello alineado con los objetivos y competencias de la asignatura. Este enfoque transforma el rol del estudiante de "codificador manual" a "arquitecto/integrador" que guía herramientas avanzadas.

Sin embargo, en el caso de que los prompts sean obtenidos desde un repositorio o desarrollados por un compañero las repercusiones serían totalmente diferentes, tal como se indican en el apartado anterior.

5.3.4 Escenarios de Inclusión de IA

Con lo comentado en los apartados 5.3.2 y 5.3.3 sobre los efectos que puede tener la IA y redacción de prompts en la asignatura se ha planteado dos metodologías para la integración de la IA, diferenciándose en su efecto en los objetivos docentes.

Sin Inclusión de la IA en los objetivos docentes

Dado que los objetivos actuales y resultados de aprendizaje se centran en la aplicación directa y manual de tecnologías y herramientas web y en el desarrollo personal de las tareas de diseño e implementación, el uso de la IA debería limitarse a roles de apoyo que no sustituyan la ejecución por parte del estudiante.

Si la IA se utilizara de forma que el estudiante no tuviera que realizar las tareas de programación, diseño de base de datos, implementación de lógica, etc., de forma autónoma, los objetivos, habilidades y competencias asociados a esas tareas no se cumplirían en la forma esperada por la guía docente. Teniéndose que restringirse a:

- **Asistencia en la búsqueda y comprensión de documentación:** Ayudar a los estudiantes a encontrar y entender manuales, artículos y especificaciones en inglés. La IA sintetizaría la información o explicaría conceptos, pero el estudiante seguiría siendo responsable de aplicar ese conocimiento.
- **Ayuda en la depuración de código:** La IA podría sugerir posibles errores en el código escrito por el estudiante o proponer correcciones. Esto podría facilitar la resolución de problemas, pero el estudiante debe entender el error y la solución para aplicarla de forma significativa.
- **Generación de ejemplos o fragmentos de código para análisis:** La IA podría generar pequeños bloques de código que el estudiante debería analizar, modificar e integrar manualmente en su proyecto. Esto refuerza la comprensión y la

capacidad de utilizar tecnologías, siempre y cuando el estudiante no use el código generado directamente sin entenderlo.

- **Revisión y mejora de la comunicación escrita:** Utilizar la IA para corregir la gramática, el estilo o la redacción de los documentos. Esto apoyaría las habilidades de comunicación escrita.
- **Lluvia de ideas:** La IA ayudaría a generar ideas para las especificaciones del proyecto o los casos de uso, facilitando el trabajo en equipo y la planificación.
- **Validación básica o pruebas unitarias:** La IA ayudaría a generar casos de prueba o verificar la sintaxis, apoyando la validación e implementación.

Inclusión de la IA como parte de la metodología docente

La integración profunda de IA requeriría redefinir completamente los objetivos de la asignatura, transformando el enfoque desde la implementación manual hacia la gestión y supervisión de herramientas de IA en el desarrollo.

Los objetivos, habilidades y competencias deberían orientarse hacia:

Objetivos docentes:

Formar profesionales capaces de incorporar eficientemente herramientas de inteligencia artificial en todas las fases del desarrollo web, dominando técnicas avanzadas de prompt engineering para maximizar los resultados de estos sistemas, mientras adquieren una comprensión profunda de sus fundamentos, potencialidades y restricciones, permitiéndoles evaluar con pensamiento crítico el impacto real de estas tecnologías en sus proyectos profesionales.

Resultados de Aprendizaje:

Los resultados de aprendizaje se orientan a que los estudiantes posean comprensión profunda de las interrelaciones entre componentes generados mediante IA en entornos web, la aplicación estratégica de estas tecnologías en diversos contextos técnicos, y el enfoque en arquitecturas de alto nivel con verificación meticulosa de los resultados automatizados. Además, desarrollarán habilidades para documentar razonadamente sus decisiones tecnológicas, mantenerse actualizados sobre avances en herramientas de IA y sus interfaces, gestionar con autonomía proyectos potenciados por inteligencia artificial, y cultivar un pensamiento crítico que les permita evaluar, refinar y personalizar soluciones generadas automáticamente para su óptima integración en proyectos web profesionales.

Competencias:

Las competencias evolucionan hacia la resolución de problemas de integración de IA, gestión de información generada automáticamente y aplicación de conocimiento para

guiar/validar el desarrollo asistido, incluyendo habilidades de comunicación sobre procesos con IA, trabajo en equipo que incorpora estas herramientas, evaluación crítica de resultados automatizados, aplicación de IA a problemas de ingeniería telemática, y transición desde la programación manual hacia la comprensión, validación, modificación e integración de componentes generados por IA en arquitecturas web completas.

Capítulo 6. Conclusiones y Líneas futuras

6.1 Conclusiones que afectan a la acción docente

En esta sección, se recogen algunas conclusiones que se podrían tener en cuenta para el desarrollo de la asignatura de Programación Web tras los datos mostrados en el capítulo 5, esto implica la adecuación de la metodología docente en los siguientes aspectos:

- **Educación mediante ingeniería inversa:** Utilizar modelos de IA con alta coherencia, consistencia en el estilo y buen uso de comentarios puede guiar al estudiante a comprender y analizar código de calidad.
- **Educación basada en la eficiencia de la IA:** Emplear modelos que resuelven las fases con menos iteraciones puede liberar tiempo para profundizar en conceptos que requieran mayor atención.
- **Educación a través de la resolución de problemas:** El uso de modelos que encontraron más dificultades o requirieron múltiples iteraciones para superar las fases puede fomentar la autonomía del estudiante al enfrentarse a los desafíos que la IA no resolvió fácilmente. El desarrollo con Claude ejemplifica esto, donde la falta de especificación inicial en los *prompts* llevó a numerosas iteraciones y a la resolución de problemas emergentes por parte del estudiante.
- **Educación mediante la elaboración de *prompts* contextualizados:** Antes de generar código con la IA, se propone que el estudiante investigue, estructure y razone sobre los aspectos del problema a resolver. Esto implica adquirir y comprender el conocimiento necesario para luego comunicarlo eficazmente a la IA, permitiendo la generación de código más preciso y el logro del resultado esperado. Los *prompts* iniciales dirigidos a modelos posteriores a Claude ilustran esta metodología, al incorporar el conocimiento previo relevante para cada fase (formato, distribución de archivos, nomenclatura, buenas prácticas, etc.).

6.2 Conclusiones

Tras los resultados expuestos en el capítulo 5 y los elementos detallados en los anexos, se puede concluir que el trabajo cumple con los objetivos definidos en el capítulo 1. Con este trabajo se propuso analizar y evaluar modelos de Inteligencia Artificial (IA) para la generación automática de código, aplicándolos específicamente al desarrollo de una aplicación web para la asignatura Programación Web.

Esto se aborda en varias partes del documento, incluyendo una revisión de aspectos importantes de la IA en el Capítulo 2 y la evaluación preliminar en el Capítulo 4, culminando en el análisis detallado de los resultados en el Capítulo 5.

Los objetivos específicos definidos al comienzo se puede desglosar como sigue:

- **OBJ-1: Explorar y analizar el uso de modelos de inteligencia artificial en la generación automática de código.** Esto implica investigar, evaluar y documentar las capacidades de diferentes modelos de IA, identificando conductas y mejores prácticas en el diseño de prompts para maximizar su efectividad. El TFG describe la revisión de modelos importantes como GPT-4, DeepSeek R1, OpenAI o1, o3-mini, o3-mini-high, Claude 3.5 Sonnet, Copilot, Mixtral 8x7B, y Mistral Large 2, entre otros. Se realizó una evaluación preliminar de un grupo de modelos seleccionados, y posteriormente se llevó a cabo un análisis más profundo en el Capítulo 5. Se evaluaron modelos basándose en métricas como tasa de errores de código, tiempo de respuesta y calidad del código generado.
- **OBJ-2: Desarrollar una app web modular y escalable con IA.** El proyecto implicó la creación de una aplicación web funcional, un Sistema de Gestión de Contenidos (CMS) orientado a un mercado local, diseñado para facilitar el aprendizaje de la asignatura Programación Web mediante el uso de IAs generativas. Se detallan las especificaciones del CMS, incluyendo la capa de presentación (Frontend), y la capa de lógica de negocio (Backend) con Laravel y una estructura basada en MVC. El desarrollo de este proyecto se inició con Claude Sonnet 3.7 y se recreó posteriormente con otros modelos seleccionados. La Tabla 15 muestra las fases y funcionalidades logradas con Claude 3.7 Sonnet, indicando que se implementaron la mayoría de las funcionalidades, a excepción de la integración de una pasarela de pago. Esto demuestra el desarrollo de una app web funcional asistida por IA.
- **OBJ-3: Analizar los prompts y evaluar.** Este objetivo se centra en identificar y documentar buenas prácticas en la creación de prompts para la generación de código relacionado con las tecnologías de programación web. Se creó un entorno sistemático para evaluar la efectividad de los prompts, incluyendo categorías como claridad, especificidad, precisión técnica, provisión de contexto, calidad de la estructura y calidad del código generado. En la sección 5.3, se presenta información sobre los prompts utilizados con diferentes modelos, y en la sección 5.3.1, se detallan las estrategias identificadas para la elaboración efectiva de prompts, como la estructura jerárquica, contextualización, especificidad técnica,

separación de responsabilidades, enfoque en buenas prácticas, validación, entrega de entregables concretos y lenguaje imperativo.

6.3 Líneas futuras

Algunas líneas futuras que se derivan de este trabajo podrán incluir:

- Evaluación continua de nuevos modelos de IA a medida que surgen y evolucionan, comparando su rendimiento y aplicabilidad a tareas de programación web y escenarios educativos.
- Investigación más profunda en el diseño de prompts, desarrollando metodologías estandarizadas o herramientas para optimizar la interacción entre estudiantes y modelos de IA.
- Desarrollo e implementación de estrategias pedagógicas específicas que integren el uso de IA generativa de código en el plan de estudios de Programación Web, basándose en los efectos observados y los escenarios de inclusión propuestos.
- Análisis del impacto a largo plazo del uso de herramientas de generación de código por IA en la adquisición de competencias de programación por parte de los estudiantes y en su preparación para el mercado laboral.
- Aplicación de los hallazgos a otras áreas o asignaturas dentro del Grado en Ingeniería en Tecnologías de la Telecomunicación o en otras disciplinas de ingeniería.

Bibliografía

- [1] P. E. C. Sánchez, R. E. R. Ibañez, y J. L. L. Pabón, «Impacto del uso de inteligencia artificial para la creación de contenidos digitales en estudiantes del pregrado en ingeniería multimedia», *Rev. Colomb. Tecnol. Av. RCTA*, vol. 2, n.º 44, Art. n.º 44, ago. 2024, doi: 10.24054/rcta.v2i44.3046.
- [2] «Uso de herramientas de IA generativa para la automatización del desarrollo de software: un caso de estudio». Accedido: 24 de febrero de 2025. [En línea]. Disponible en:
<https://citt.itsm.edu.mx/ingeniantes/articulos/ingeniantes11no1vol3/19.pdf>
- [3] «gpt-4.pdf». Accedido: 17 de mayo de 2025. [En línea]. Disponible en:
<https://cdn.openai.com/papers/gpt-4.pdf>
- [4] «DeepSeek-R1/DeepSeek_R1.pdf at main · deepseek-ai/DeepSeek-R1», GitHub. Accedido: 24 de febrero de 2025. [En línea]. Disponible en:
https://github.com/deepseek-ai/DeepSeek-R1/blob/main/DeepSeek_R1.pdf
- [5] «Fix code vulnerabilities automatically | Snyk User Docs». Accedido: 24 de febrero de 2025. [En línea]. Disponible en:
<https://docs.snyk.io/scan-with-snyk/snyk-code/manage-code-vulnerabilities/fix-code-vulnerabilities-automatically>
- [6] D. Smilkov *et al.*, «TensorFlow.js: Machine Learning for the Web and Beyond», 28 de febrero de 2019, *arXiv*: arXiv:1901.05350. doi: 10.48550/arXiv.1901.05350.
- [7] M. Chen *et al.*, «Evaluating Large Language Models Trained on Code», 14 de julio de 2021, *arXiv*: arXiv:2107.03374. doi: 10.48550/arXiv.2107.03374.
- [8] C. Ebert, J. Cain, G. Antoniol, S. Counsell, y P. Laplante, «Cyclomatic Complexity», *IEEE Softw.*, vol. 33, n.º 6, pp. 27-29, nov. 2016, doi: 10.1109/MS.2016.147.
- [9] «pdf.pdf». Accedido: 24 de febrero de 2025. [En línea]. Disponible en:
https://www2.ulpgc.es/aplicaciones/proyectosdocentes/pdf.php?id_proyecto=71651&NUEVA=1
- [10] «2024_05_28_recomendaciones_sobre_el_uso_de_la_ia_en_la_ulpgc.pdf». Accedido: 31 de mayo de 2025. [En línea]. Disponible en:
https://si.ulpgc.es/sites/default/files/documentos/2024_05_28_recomendaciones_sobre_el_uso_de_la_ia_en_la_ulpgc.pdf
- [11] «boulpgc_no10_de_10_de_junio_de_2024.pdf». Accedido: 31 de mayo de 2025. [En línea]. Disponible en:
https://www.ulpgc.es/sites/default/files/ArchivosULPGC/boulpgc/BOULPGC/boulpgc_no10_de_10_de_junio_de_2024.pdf
- [12] «Beranek et al. - HISTORY OF ARTIFICIAL INTELLIGENCE P. McCorduck (Session Chairman), Univ. of Pittsburgh!». Accedido: 15 de mayo de 2025. [En línea]. Disponible en: <https://www.ijcai.org/Proceedings/77-2/Papers/083.pdf>
- [13] S. J. Russell y P. Norvig, *Artificial intelligence: a modern approach*, Fourth edition, Global edition. en Prentice Hall series in artificial intelligence. Boston: Pearson, 2022. [En línea]. Disponible en:
http://lib.yzu.am/disciplines_bk/efdd4d1d4c2087fe1cbe03d9ced67f34.pdf
- [14] J. Dempere, K. Modugu, A. Hesham, y L. K. Ramasamy, «The impact of ChatGPT on higher education», *Front. Educ.*, vol. 8, sep. 2023, doi: 10.3389/educ.2023.1206936.
- [15] «AI Index 2025: State of AI in 10 Charts | Stanford HAI». Accedido: 18 de mayo de 2025. [En línea]. Disponible en:
<https://hai.stanford.edu/news/ai-index-2025-state-of-ai-in-10-charts>
- [16] «The 2025 AI Index Report | Stanford HAI». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://hai.stanford.edu/ai-index/2025-ai-index-report>
- [17] «Las Tendencias Principales en IA Generativa de 2024 y proyecciones para 2025», Big Data & AI World Madrid 2024. Accedido: 26 de mayo de 2025. [En línea]. Disponible en:

- <https://www.bigdataworld.es/noticias/las-tendencias-principales-en-ia-generativa-de-2024-y-proyecciones-para-2025>
- [18] «Tipos de inteligencia artificial | IBM». Accedido: 15 de mayo de 2025. [En línea]. Disponible en: <https://www.ibm.com/es-es/think/topics/artificial-intelligence-types>
- [19] A. Hintze, «Understanding the four types of AI, from reactive robots to self-aware beings», *The Conversation*. Accedido: 15 de mayo de 2025. [En línea]. Disponible en: <http://theconversation.com/understanding-the-four-types-of-ai-from-reactive-robots-to-self-aware-beings-67616>
- [20] T. Baltrušaitis, C. Ahuja, y L.-P. Morency, «Multimodal Machine Learning: A Survey and Taxonomy», *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, n.º 2, pp. 423-443, feb. 2019, doi: 10.1109/TPAMI.2018.2798607.
- [21] S. J. Russell y P. Norvig, *Artificial intelligence: a modern approach*, Fourth edition, Global edition. en Prentice Hall series in artificial intelligence. Boston: Pearson, 2022.
- [22] «GPT-4o System Card». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://openai.com/index/gpt-4o-system-card/>
- [23] «OpenAI o1 System Card». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://openai.com/index/openai-o1-system-card/>
- [24] «OpenAI o3-mini System Card». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://openai.com/index/o3-mini-system-card/>
- [25] «documentación GitHub Copilot - Documentación de GitHub», GitHub Docs. Accedido: 24 de febrero de 2025. [En línea]. Disponible en: https://docs-internal.github.com/_next/data/qVwjwfskFzR399ehBAeYy/es/free-pro-team%40latest/copilot.json?versionId=free-pro-team%40latest&productId=copilot
- [26] «Claude-3-Model-Card.pdf». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://assets.anthropic.com/m/61e7d27f8c8f5919/original/Claude-3-Model-Card.pdf>
- [27] «About us | Mistral AI». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://mistral.ai/about>
- [28] A. Q. Jiang *et al.*, «Mistral 7B», 10 de octubre de 2023, *arXiv*: arXiv:2310.06825. doi: 10.48550/arXiv.2310.06825.
- [29] «Codestral 25.01 | Mistral AI». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://mistral.ai/news/codestral-2501>
- [30] A. Q. Jiang *et al.*, «Mixtral of Experts», 8 de enero de 2024, *arXiv*: arXiv:2401.04088. doi: 10.48550/arXiv.2401.04088.
- [31] «Large Enough | Mistral AI». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://mistral.ai/news/mistral-large-2407>
- [32] H. Touvron *et al.*, «LLaMA: Open and Efficient Foundation Language Models», 27 de febrero de 2023, *arXiv*: arXiv:2302.13971. doi: 10.48550/arXiv.2302.13971.
- [33] «Code Llama: Open Foundation Models for Code». Accedido: 23 de mayo de 2025. [En línea]. Disponible en: <https://arxiv.org/pdf/2308.12950>
- [34] «Daily Papers - Hugging Face». Accedido: 23 de mayo de 2025. [En línea]. Disponible en: <https://huggingface.co/papers>
- [35] M. Derrick, «How Meta's new AI Models Will Shake up the Global AI Race». Accedido: 23 de mayo de 2025. [En línea]. Disponible en: <https://technologymagazine.com/ai-and-machine-learning/what-metas-new-ai-models-mean-for-the-global-ai-race>
- [36] G. Team *et al.*, «Gemini: A Family of Highly Capable Multimodal Models», 9 de mayo de 2025, *arXiv*: arXiv:2312.11805. doi: 10.48550/arXiv.2312.11805.
- [37] «Gemini 2.5: Our most intelligent AI model», Google. Accedido: 23 de mayo de 2025. [En línea]. Disponible en: <https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/>
- [38] G. Team *et al.*, «Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context», 25 de abril de 2024, *arXiv*: arXiv:2403.05530. doi: 10.48550/arXiv.2403.05530.

- [39] «Expanding Gemini 2.5 Flash and Pro capabilities», Google Cloud Blog. Accedido: 23 de mayo de 2025. [En línea]. Disponible en: <https://cloud.google.com/blog/products/ai-machine-learning/expanding-gemini-2-5-flash-and-pro-capabilities>
- [40] «What is Conversational AI? | IBM». Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://www.ibm.com/think/topics/conversational-ai>
- [41] «Enhancing Software Development with Context-Aware Conversational Agents: A User Study on Developer Interactions with Chatbots». Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://arxiv.org/html/2505.08648v1>
- [42] W. Hou y Z. Ji, «Comparing Large Language Models and Human Programmers for Generating Programming Code», *Adv. Sci.*, vol. 12, n.º 8, p. 2412279, dic. 2024, doi: 10.1002/adv.202412279.
- [43] «On Program Synthesis and Large Language Models – Communications of the ACM». Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://cacm.acm.org/opinion/on-program-synthesis-and-large-language-models/>
- [44] T. Hliš, L. Četina, T. Beranič, y L. Pavlič, «Evaluating the Usability and Functionality of Intelligent Source Code Completion Assistants: A Comprehensive Review», *Appl. Sci.*, vol. 13, n.º 24, Art. n.º 24, ene. 2023, doi: 10.3390/app132413061.
- [45] «Is There a Future for Software Engineers? The Impact of AI [2025]». Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://brainhub.eu/library/software-developer-age-of-ai>
- [46] «Making AI-generated code more accurate in any language», MIT News | Massachusetts Institute of Technology. Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://news.mit.edu/2025/making-ai-generated-code-more-accurate-0418>
- [47] «Survey of Code Completion using DeepLearning», ResearchGate. Accedido: 17 de mayo de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/366503090_Survey_of_Code_Completion_using_DeepLearning
- [48] «Assisted Development: The Role of AI in the IDEs | Baufest». Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://baufest.com/en/ai-redefining-ides-with-automation-cloud-and-vibe-coding/>
- [49] E. Frank y O. Godwin, «Enhancing Developer Productivity: a Study on GitHub Copilot's Code Completion Capabilities».
- [50] «AI Model & API Providers Analysis | Artificial Analysis». Accedido: 17 de mayo de 2025. [En línea]. Disponible en: <https://artificialanalysis.ai>
- [51] «OSI releases first candidate definition of 'open source AI' for final comments», Osler, Hoskin & Harcourt LLP. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.osler.com/en/insights/updates/osi-releases-first-candidate-definition-of-open-source-ai-for-final-comments/>
- [52] «What is Open Source Model», Iguazio. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.iguazio.com/glossary/open-source-model/>
- [53] «Adapting the Definition of Open Source to AI: the Quest for the Holy Grail – LFAI & Data». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://lfaidata.foundation/blog/2024/02/14/adapting-the-definition-of-open-source-to-ai-the-quest-for-the-holy-grail/>
- [54] «Open-source artificial intelligence», *Wikipedia*. 29 de abril de 2025. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Open-source_artificial_intelligence&oldid=1287964729
- [55] «The Open Source AI Definition – 1.0», Open Source Initiative. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://opensource.org/ai/open-source-ai-definition>
- [56] «The Need for Open Source Software in Machine Learning.», *ResearchGate*, feb. 2025, Accedido: 18 de mayo de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/220319831_The_Need_for_Open_Source_S

- oftware_in_Machine_Learning
- [57] «The Impact of Open-Source Software on Artificial Intelligence», *ResearchGate*, doi: 10.46754/jmsi.2023.12.006.
- [58] «8 Top Open-Source LLMs for 2024 and Their Uses». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.datacamp.com/blog/top-open-source-llms>
- [59] «The Fight for Open Source in Generative AI - Network Law Review». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.networklawreview.org/open-source-generative-ai/>
- [60] «The Rise and Impact of Open-Source AI Initiatives (Part 1 of 4) | Xantage». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.xantage.co/xantage-blog-posts/the-rise-and-impact-of-open-source-ai-initiatives>
- [61] «Open Source AI vs. Proprietary AI: Pros and Cons for Developers | Novus». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.novusasi.com/blog/open-source-ai-vs-proprietary-ai-pros-and-cons-for-developers>
- [62] «Op-ed: The Benefits Of Open Source AI – The Fieldston News». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://fieldstonnews.com/home/2024/05/op-ed-the-benefits-of-open-source-ai/>
- [63] «The Best Embedding Models for Retrieval-Augmented Generation (RAG)». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://writingmate.ai/blog/the-best-embedding-models>
- [64] Shubham, «Beginner's Guide to Embedding Models». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://learnopencv.com/embedding-models-explained/>
- [65] «The best open-source embedding models | Baseten Blog», Baseten. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.baseten.co/blog/the-best-open-source-embedding-models/>
- [66] A. Joshi, «How to Choose the Best Embedding Model for Your LLM Application | MongoDB». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.mongodb.com/developer/products/atlas/choose-embedding-model-rag/>
- [67] «A Guide to Open-Source Embedding Models». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.bentoml.com/blog/a-guide-to-open-source-embedding-models>
- [68] «A Comprehensive Survey of Image and Video Generative AI: Recent Advances, Variants, and Applications», *ResearchGate*. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/377751387_A_Comprehensive_Survey_of_Image_and_Video_Generative_AI_Recent_Advances_Variants_and_Applications
- [69] C. Zhang, C. Zhang, M. Zhang, I. S. Kweon, y J. Kim, «Text-to-image Diffusion Models in Generative AI: A Survey», 8 de noviembre de 2024, *arXiv*: arXiv:2303.07909. doi: 10.48550/arXiv.2303.07909.
- [70] «A Survey on Generative AI and LLM for Video Generation, Understanding, and Streaming». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://arxiv.org/html/2404.16038v1>
- [71] «The Case for Open-Source Generative AI in Government». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.boozallen.com/insights/ai-research/the-case-for-open-source-generative-ai-in-government.html>
- [72] «A Guide to Open-Source Image Generation Models». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.bentoml.com/blog/a-guide-to-open-source-image-generation-models>
- [73] «Top open-source text-to-video AI models», *Modal*. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://modal.com/blog/text-to-video-ai-article>
- [74] «What is AI code generation?», *GitHub*. Accedido: 18 de mayo de 2025. [En línea].

- Disponible en: <https://github.com/resources/articles/ai/what-is-ai-code-generation>
- [75] «Top 5 Open Source Models for Coding», Athina AI Hub. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://hub.athina.ai/blogs/top-open-source-models-for-code-generation-in-2025/>
- [76] «CODEGEN: A Transformative Open-Source Language Model for Versatile Program Synthesis | DigitalOcean». Accedido: 18 de mayo de 2025. [En línea]. Disponible en: <https://www.digitalocean.com/community/tutorials/codegen-open-source-language-model-for-code-generation>
- [77] «Best Hardware for Running Large Language Models LLMs». Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://rational.co.in/best-hardware-for-running-large-language-models-llm/>
- [78] «Research infinity - Hardware requirements for running the large language model Deepseek R1 locally.» Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.rnfinity.com/news-show/Hardware-requirements-for-running-large-language-model-Deepseek-R1-on-a-local-machine>
- [79] «¿Qué es Docker? | IBM». Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.ibm.com/es-es/think/topics/docker>
- [80] «Top 10 LLM Tools to Run Models Locally in 2025 - AI Tools». Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.godofprompt.ai/blog/top-10-llm-tools-to-run-models-locally-in-2025>
- [81] «What is ai infrastructure? | IBM». Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.ibm.com/think/topics/ai-infrastructure>
- [82] A. G, «The 6 Best LLM Tools To Run Models Locally». Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://getstream.io/blog/best-local-llm-tools/>
- [83] S. Burton, «Install an AI LLM on Your Computer: A Step-by-Step Guide», Adventures in CRE. Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.adventuresinre.com/how-to-install-llm-locally/>
- [84] A. Muhammad, «What is Ollama? Understanding how it works, main features and models», Hostinger Tutorials. Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.hostinger.com/tutorials/what-is-ollama>
- [85] «Part 2: Ollama Advanced Use Cases and Integrations - Cohorte Projects». Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://www.cohorte.co/blog/ollama-advanced-use-cases-and-integrations>
- [86] «Seven Ways of Running Large Language Models (LLMs) Locally (April 2024)», kleiber.me – Ingo Kleiber. Accedido: 19 de mayo de 2025. [En línea]. Disponible en: <https://kleiber.me/blog/2024/01/07/seven-ways-running-llm-locally/>
- [87] «GitHub - open-webui/open-webui: User-friendly AI Interface (Supports Ollama, OpenAI API, ...)». Accedido: 24 de febrero de 2025. [En línea]. Disponible en: <https://github.com/open-webui/open-webui>

Presupuesto

En este apartado se exponen los costes correspondientes a la elaboración del proyecto. Teniendo en cuenta el contexto académico de este Trabajo Fin de Grado, se ha tomado como guía las directrices del Colegio Oficial de Graduados e Ingenieros Técnicos de Telecomunicación (COITT). Sin embargo, a la hora de la valoración de proyectos en el ejercicio libre de la profesión debe tenerse en cuenta “El Ministerio de Economía y Hacienda remitió a todos los colegios profesionales una nota en la que se nos recordaba que, siguiendo directivas europeas, se debían eliminar los baremos orientativos de honorarios que tradicionalmente veníamos publicando.”¹⁰

Los costes se han dividido en las siguientes secciones:

- Materiales usados
- Trabajo tarifado por tiempo empleado
- Costes asociados a la redacción del Trabajo Fin de Grado
- Gastos derivados de los impuestos

P1. Materiales usados

En este apartado, se examina la utilización de los elementos físicos y lógicos que sustentan el desarrollo del proyecto. Para calcular su coste, se ha empleado un método de depreciación lineal, el cual distribuye equitativamente la pérdida de valor de estos elementos a lo largo de su vida útil estimada. Si bien la vida útil estándar considerada es de 4 años, la duración real de este Trabajo de Fin de Grado ha sido de cuatro meses, lo que ha requerido un ajuste proporcional en los cálculos para reflejar este periodo específico.

Equipos

El Trabajo de Fin de Grado se completó en un periodo de cuatro meses, un intervalo temporal significativamente menor en comparación con el periodo de tres años que comúnmente se utiliza como base para calcular la depreciación del equipamiento físico. En consecuencia, los gastos de amortización presentados corresponden exclusivamente al valor de uso durante estos cuatro meses específicos.

Debido a la diferencia temporal del periodo de elaboración del TFG y el de vida útil del material usado, se ha tomado la relación entre los dos periodos para el cálculo de la amortización:

¹⁰ <https://www.telecos.zone/libre-ejercicio/honorarios-profesionales-le>

Vida útil (meses) = 4 años * 12 meses/1 año = 48 meses

Valor amortizado = 4 meses/48 meses * Valor de adquisición

La Tabla 21 detalla los elementos de hardware fundamentales utilizados en el proyecto, indicando tanto su precio de compra original como la porción de su valor que se ha depreciado durante el periodo de desarrollo.

Tabla 21 Tabla de amortización de los recursos hardware

Elemento	Valor de adquisición	Amortización
Portátil	259,28 €	21,60 €
Ordenador de sobremesa	2.329,58 €	194,13 €
Total		215,73 €

El coste total de amortización de los materiales físicos es **doscientos quince euros con setenta y tres céntimos (215,73)**, el cual coincide con el portátil usado para la redacción ya que el ordenador de sobremesa usado para la carga de modelos de IA fue facilitado por la universidad.

Software

Para el software implementado en este Trabajo de Fin de Grado, la amortización se calcula considerando una utilización de 4 meses dentro de un ciclo de vida útil de 4 años.

Las herramientas de software seleccionadas han sido principalmente de naturaleza gratuita y open source:

- Docker Desktop: Al ser usado su versión sin suscripción y no requerir pago de licencia todos los costes de amortización serán nulos.
- OpenWebUI y Ollama: Al ser Software Libre no requiere el pago de licencias por lo que todos los costes de amortización serán nulos.
- Windows 10 y 11: Al venir ya instalados en las máquinas usadas no requiere el pago de licencias por lo que todos los costes de amortización serán nulos.
- Google Drive: el ecosistema de ofimática gratuito de Google no requiere el pago de suscripciones por lo que todos los costes de amortización serán nulos.
- ChatGPT Plus: para la utilización de ciertos modelos de OpenAI se pago la suscripción mensual de pago, un total de **noventa y tres euros con ochenta y cinco céntimos (93,85)** durante el periodo del trabajo.

En consecuencia, el coste total derivado del software utilizado es de **noventa y tres euros con ochenta y cinco céntimos (93,85€)**.

P2. Trabajo por tiempo empleado

En la ejecución de este proyecto se han dedicado aproximadamente 300 horas distribuidas entre las fases de diseño, desarrollo y creación de la documentación. De acuerdo con las directrices del COITT, la valoración económica del trabajo realizado puede determinarse mediante la siguiente fórmula:

$$H = C_t * 74.88 * H_n + C_t * 96.72 * H_e$$

- H : Importe total de honorarios correspondientes al proyecto
- C_t : Coeficiente de ajuste en función de las horas empleadas
- H_n : Horas desarrolladas durante jornada laboral ordinaria
- H_e : Horas desarrolladas en horario extraordinario (para este proyecto su valor es 0 al no haberse registrado)

Considerando los baremos establecidos por el COITT, el coeficiente de ajuste correspondiente a las horas empleadas, según se especifica en la Tabla 22, equivale a 0,60.

Tabla 22 Valores del factor de corrección en función a las horas trabajadas

Horas empleadas	Factor de corrección C_t
$X < 36$	1
$36 < X < 72$	0,90
$72 < X < 108$	0,80
$108 < X < 144$	0,70
$144 < X < 180$	0,65
$180 < X < 360$	0,60
$360 < X < 540$	0,55

Conforme a dicha tabla, al completarse este proyecto en 300 horas, corresponde aplicar el coeficiente de ajuste con valor de 0.60. En base a esto, la expresión matemática anterior se establece de la siguiente forma:

$$H = 0.6 * 74.88 * 300 + 0.6 * 96.72 * 0 = 13478.40 \text{ €}$$

Los honorarios resultantes del tiempo invertido en el proyecto, sin incluir impuestos, alcanzan un total de **trece mil cuatrocientos setenta y ocho euros con cuarenta céntimos (13.478,40 €)**

Redacción de documentación

Con respecto al coste de la redacción del documento se utiliza la ecuación:

$$H = 0.07 * P * C_n$$

- R : Honorarios correspondientes a la elaboración del documento.
- P : presupuesto total del trabajo.
- C_n : coeficiente de ponderación determinado según el presupuesto.

El importe final resulta de la adición de los costes laborales facturados por tiempo invertido, previamente calculados, y la depreciación de recursos materiales cuyo total se detalla en la Tabla 23.

Tabla 23 Presupuesto del trabajo tarifado y amortización de los recursos materiales

Descripción	Costes
Amortización de recursos materiales	223,55 €
Trabajo tarifado por tiempo empleado	13.478,40 €
Total	13.701,95 €

Dado que el coeficiente de ponderación para presupuestos inferiores a 30.050,00€ se establece por el COITT en 1.00, el coste de elaboración documental del TFG resulta:

$$H = 0,07 * 13.701,95 * 1 = 959,14€$$

En definitiva, el coste de redacción del proyecto alcanza un importe de **novecientos cincuenta y nueve euros con catorce céntimos (959,14 €)**.

P3. Aplicación de impuestos y coste total

A este Trabajo de Fin de Grado se le aplica el Impuesto General Indirecto Canario (IGIC), equivalente al 7% del importe presupuestario. El presupuesto global del proyecto se detalla en la Tabla 24.

Tabla 24 Aplicación de impuestos a los costes

Concepto	Costes
Amortización de recursos materiales	223,55 €
Trabajo tarifado por tiempo empleado	13.478,40 €
Redacción de documentación	959,14€
Subtotal (Sin IGIC)	14.661,09 €
IGIC (7%)	1.026,27 €
Total	15.687,37 €

El Trabajo de Fin de Grado denominado "iaPWeb. Análisis de las Inteligencias Artificiales generativas de código para Programación Web", desarrollado en la Escuela de Ingeniería de Telecomunicaciones y Electrónica de la Universidad de las Palmas de Gran Canaria, presenta un coste total de desarrollo de **quince mil con seiscientos ochenta y siete euros con treinta y siete céntimos (15.687,37 €)**, que corresponde a la suma de los importes asignados a los conceptos anteriormente especificados.



Firmado: Alexander Castilla Campeny

Fecha: 04/06/2025

Objetivos de Desarrollo Sostenible

Grado de relación del TFG con los objetivos de desarrollo sostenible

Tabla 25. Objetivos de desarrollo sostenible

ODS	Grado de relación con los ODS			
	0 No procede	1 Bajo	2 Medio	3 Alto
ODS 1 Fin de la Pobreza	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 2 Hambre cero	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 3 Salud y Bienestar	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 4 Educación de calidad	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ODS 5 Igualdad de género	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 6 Agua limpia y saneamiento	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 7 Energía Asequible y no contaminante	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 8 Trabajo decente y crecimiento económico	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ODS 9 Industria, Innovación e Infraestructuras	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ODS 10 Reducción de las desigualdades	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ODS 11 Ciudades y comunidades sostenibles	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 12 Producción y consumo sostenibles	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 13 Acción por el clima	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 14 Vida submarina	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 15 Vida de ecosistemas terrestres	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 16 Paz, justicia e instituciones sólidas	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ODS 17 Alianzas para lograr objetivos	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Justificación del alineamiento del TFT con los ODS con los que se ha indicado que tiene un mayor grado de relación:

ODS 4: Educación de Calidad

Se exploran y proponen metodologías para integrar la IA en la enseñanza de la programación, analizando cómo afecta los objetivos docentes, resultados de aprendizaje y competencias de los estudiantes. Esto se alinea directamente con el objetivo de mejorar la calidad de la educación y la adquisición de competencias relevantes para el mercado laboral

ODS 8: Trabajo Decente y Crecimiento Económico

El trabajo analiza la programación asistida por IA como un paradigma en evolución que puede aumentar la productividad de los desarrolladores. El CMS para el mercado local busca fomentar la producción mediante la promoción de productos artesanales, apoyando el trabajo de artesanos y vendedores locales. Además, se discute cómo el uso de IA en la educación prepara a los estudiantes para un mercado laboral impulsado por la IA, lo que está relacionado con el desarrollo de habilidades para el trabajo decente.

ODS 9: Industria, Innovación e Infraestructura

El proyecto se centra en el análisis de las Inteligencias Artificiales generativas de código, que representan una innovación significativa en la industria del software. Se revisan los desarrollos más importantes en IA generativa. y modelos de IA OpenSource, analizando sus características y los entornos de ejecución (infraestructuras) requeridos. El CMS desarrollado también puede considerarse una pieza de infraestructura digital para el mercado local.

ODS 10: Reducción de las Desigualdades

El trabajo destaca la importancia de la IA de código abierto (Open Source) como un elemento crítico para democratizar el acceso a tecnologías avanzadas, lo cual contribuye a reducir la desigualdad en el acceso a herramientas de IA.

ANEXOS

A1 Guía de instalación del entorno Ollama-OpenWebUI

A1.1 Especificaciones de Hardware Recomendadas

A continuación, se presentan los requisitos de hardware sugeridos para obtener un rendimiento óptimo con Ollama¹¹:

Tabla 26. Requerimientos hardware para Ollama

Hardware	Mínimo / Sugerido	Explicación
RAM	16 GB (sugerido)	Permite ejecutar modelos de hasta 7B (7 mil millones de parámetros).
Espacio en Disco	12 GB (mínimo)	Requiere espacio adicional según el número y tamaño de los modelos instalados.
Procesador	4 núcleos (mínimo), 8 núcleos (para 13B)	Procesadores modernos recomendados para un buen rendimiento, especialmente con modelos más grandes.
GPU (opcional)	No obligatoria	Mejora notablemente el rendimiento con modelos grandes, especialmente en ejecución y entrenamiento personalizado.

Dependencias de Software Necesarias

- **Administrador de paquetes** (ej. `apt`, `yum`, `brew`).
- **Python 3.9 o superior** (para diversas librerías y scripts de automatización).
- **Docker** (opcional, recomendado para aislar entornos).
- **Controladores de GPU y CUDA (opcional)**: Instalar los controladores de tarjeta gráfica (NVIDIA/AMD) y CUDA (NVIDIA) para la aceleración de modelos.

Sistemas Operativos Compatibles

- **Linux** (Ubuntu/Debian, Fedora, CentOS).
- **macOS** (versión 11 o superior).
- **Windows 10/11** (se recomienda instalar mediante Docker WSL¹² para simplificar).

A1.2 Proceso de Instalación (Ollama, Open WebUI)

1.2.1 Instalación de Ollama

Instalación General

1. Actualizar el sistema operativo con los últimos parches y paquetes.
2. Descargar el paquete de Ollama¹³ en su página web o instalarlo con el método apropiado para el sistema operativo.
3. Verificar la versión instalada mediante el comando correspondiente:

¹¹Requerimientos para Ollama-OpenWebUI:

<https://github.com/open-webui/open-webui/discussions/736>

¹² Documentación sobre WSL: <https://learn.microsoft.com/es-es/windows/wsl/>

¹³ Pagina de descarga de Ollama: <https://ollama.com/download>

```
ollama --version
```

También se puede acceder a Ollama vía navegador en localhost:11434 (API).

4. Configuración Inicial (opcional): En Docker, establecer variables de entorno¹⁴.
5. Verificación del sistema

Ejecutar el comando de prueba para cargar un modelo simple:

```
ollama pull [nombre_del_modelo] # Descarga el modelo
ollama run [nombre_del_modelo] # Carga el modelo (o lo descarga si no está)
```

Revisar que no aparezcan errores y que el modelo responda a consultas.

Instalación en Windows¹⁵

Requisitos del Sistema

- Windows 10 22H2 o más reciente (Home o Pro).
- En caso de una GPU NVIDIA, controladores NVIDIA¹⁶ 452.39 o más nuevos.
- En caso de una GPU NVIDIA, controladores CUDA¹⁷.
- En caso de una GPU AMD, controlador AMD Radeon¹⁸.

Pasos de instalación

- Descarga el instalador (.exe) de la versión de Windows.
- Ejecuta el archivo .exe y seguir las instrucciones.
- Verificar la instalación con el siguiente comando en la terminal de PowerShell o CMD:

```
ollama --version
```

- (Opcional) Instalación mediante WSL o Docker para proyectos más avanzados.
- (Opcional) Configurar variables de entorno en *Panel de Control* → *Sistema* → *Configuración avanzada del sistema* → *Variables de entorno*.

Instalación en Linux¹⁹

1. Actualizar el sistema

```
sudo apt update
sudo apt upgrade
```

2. Instalar dependencias

¹⁴ Variables de Entorno de Ollama:

<https://hostkey.com/documentation/technical/gpu/ollama/#introduction-to-ollama>

¹⁵ Documentación de Ollama para Windows:

<https://github.com/ollama/ollama/blob/main/docs/windows.md>

¹⁶ Drivers de Nvidia: <https://www.nvidia.com/es-es/drivers/>

¹⁷ Drivers CUDA de Nvidia: <https://developer.nvidia.com/cuda-downloads>

¹⁸ Drivers de AMD: <https://www.amd.com/en/support/download/drivers.html>

¹⁹ Documentación de Ollama para Linux:

<https://github.com/ollama/ollama/blob/main/docs/linux.md>

```
sudo apt install python3 python3-pip git
```

3. Descargar e extraer Ollama

```
curl -L https://ollama.com/download/ollama-linux-amd64.tgz -o ollama-linux-amd64.tgz ##Descargar  
sudo tar -C /usr -xzf ollama-linux-amd64.tgz ##Extraer
```

4. Empezar Ollama

```
ollama serve
```

5. Verificar la instalación

```
ollama -v
```

6. Instalar y cargar modelos

```
ollama pull [nombre del modelo] ##Descargar modelo  
ollama run [nombre del modelo] ##Cargar modelo
```

Configurar el servicio Systemd (opcional para arranque automático).

```
sudo systemctl start ollama ##Iniciar el servicio  
sudo systemctl enable ollama ##Habilitar el servicio
```

Instalar CUDA para GPUs NVIDIA (opcional).

```
nvidia-smi
```

1.2.2 Instalación de Open WebUI

Procedimiento General

1. Clonar el repositorio oficial²⁰:

```
git clone https://github.com/open-webui/open-webui.git
```

2. Instalar las dependencias requeridas (descritas en el README).
3. Configurar variables de entorno si fuera necesario.

Configuración Básica (opcional)

- Ajustar el archivo de configuración local para indicar la carpeta de modelos.
- Definir los puertos de escucha y los parámetros de seguridad, si aplica.

Pruebas de Funcionamiento

- Iniciar la interfaz web y accede a localhost:8080.

²⁰ Repositorio de OpenWebUI: <https://github.com/open-webui/open-webui?tab=readme-ov-file>



Figura 37. Login de Open-WebUI

- Registrarse en Open-WebUI (la primera cuenta registrada es la administradora)
- Descargar un modelo con Ollama

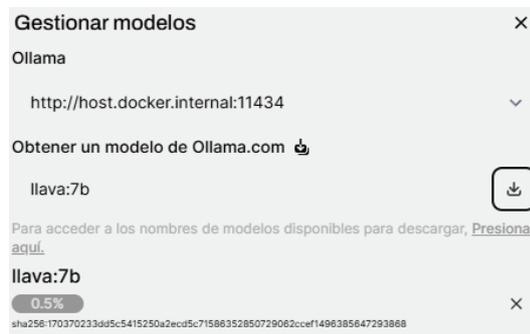


Figura 38. Descargar un modelo

- Carga un modelo de prueba para validar el correcto funcionamiento.

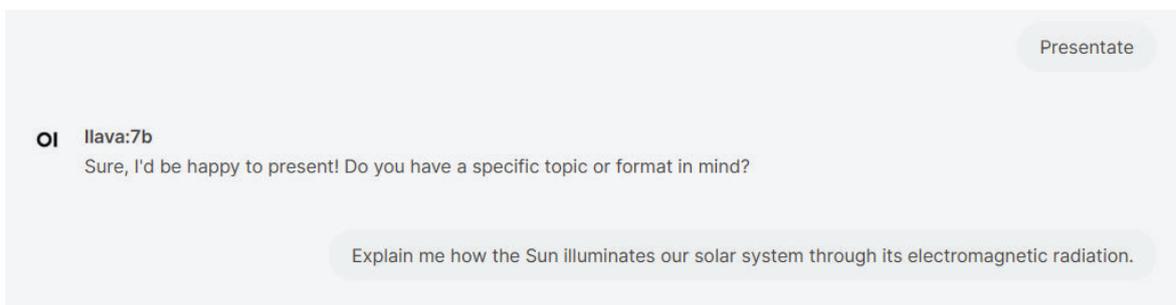


Figura 39. Navegador con modelo cargado

Instalación según el SO

Instalación con uv

macOS/Linux:

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

Windows:

```
$env:DATA_DIR="C:\open-webui\data"; uvx --python 3.11 open-webui@latest serve
```

Cargar Open-WebUI

macOS/Linux:

```
DATA_DIR=~/.open-webui uvx --python 3.11 open-webui@latest serve
```

Windows:

```
$env:DATA_DIR="C:\open-webui\data"; uvx --python 3.11 open-webui@latest serve
```

Instalación con Python pip

Asegurar de usar Python 3.11 para evitar incompatibilidades:

```
pip install open-webui
open-webui serve
```

Acceder a la interfaz local `localhost:8080` para utilizar Open-WebUI.

Inicio Rápido con Docker

Esta instalación permite tener la aplicación en un entorno aislado para mejor gestión.

Instalación con la Configuración Predeterminada

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v
open-webui:/app/backend/data --name open-webui --restart always
ghcr.io/open-webui/open-webui:main
```

Si Ollama está en un Servidor Diferente

```
docker run -d -p 3000:8080 -e OLLAMA_BASE_URL=https://example.com -v open-webui:/app/backend/data
--name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Soporte GPU NVIDIA²¹

```
docker run -d -p 3000:8080 --gpus all --add-host=host.docker.internal:host-gateway -v
open-webui:/app/backend/data --name open-webui --restart always
ghcr.io/open-webui/open-webui:cuda
```

Uso Exclusivo de OpenAI API

```
docker run -d -p 3000:8080 -e OPENAI_API_KEY=your_secret_key -v open-webui:/app/backend/data
--name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Consejos

- En caso de problemas de conexión, revisar la guía oficial²² de la Documentación de Open-WebUI.
- Incluir al comando `docker run -v open-webui:/app/backend/data` para evitar pérdida de datos.

²¹ Para mejorar la experiencia de uso y acelerar el proceso de los modelos se ha optado por la instalación con soporte GPU.

²² Documentación de OpenWebUI: <https://docs.openwebui.com/>

Instalar Open-WebUI con Soporte Integrado de Ollama

Este método de instalación utiliza una única imagen de contenedor que incluye Open-WebUI junto con Ollama, lo que permite una configuración simplificada con un solo comando. Elegir el comando adecuado según las características:

Con Soporte GPU:

```
docker run -d -p 3000:8080 --gpus=all -v ollama:/root/.ollama -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:ollama
```

Solo CPU:

```
docker run -d -p 3000:8080 -v ollama:/root/.ollama -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:ollama
```

Ambos comandos ofrecen una instalación integrada y sencilla de Open-WebUI y Ollama, asegurando que todo funcione de inmediato. Una vez finalizada la instalación, se puede acceder a la interfaz en `localhost:3000`.

A1.3 Configuración del Entorno de Ollama

Para que Ollama y Open-WebUI funcionen de manera óptima, es posible configurar ciertas variables de entorno²³ que definen aspectos como la dirección IP, la ubicación de los modelos, la memoria de la GPU, entre otros aspectos.

- Declarar estas variables en el shell o en archivos de configuración (p. ej. `.bashrc`, `.zshrc`, `.env`), en Linux, en `/etc/systemd/system/ollama.service`. o en Docker el archivo `docker-compose.yaml`
- Ajustar cada variable según las necesidades o déjalas con el valor por defecto si no lo requiere.

Validación de Conexión

1. Verificación de Conectividad

Un método sencillo para comprobar la conexión entre Ollama y Open-WebUI es usar el apartado de conexiones de Open-WebUI donde está definido la url a Ollama y darle al botón de comprobación.



Figura 40. Mensaje de conexión entre Ollama y Open-WebUI

2. Comprobación del uso de GPU

²³ <https://docs.openwebui.com/getting-started/env-configuration>

Si se han configurado e instalado correctamente lo necesario para el uso de la GPU en la carga de modelos, al ejecutar `ollama ps` en la consola debería salir algo parecido:

```
# ollama ps
NAME          ID              SIZE    PROCESSOR    UNTIL
llava:7b     8dd30f6b0cb1  7.0 GB  100% GPU     4 minutes from now
```

Figura 42. Lista de modelos de IA cargados en Ollama

Problemas Encontrados y Soluciones

1. Descripción de Obstáculos Durante la Instalación

- Durante los primeros intentos se intentó la instalación de Open WebUI con versiones más actuales de Python, resultando en problemas de compatibilidad.
- Durante la instalación de Open WebUI en docker hubo problemas con la conexión con Ollama, debido a la configuración de redes de docker.
- En la comprobación del acceso a Open WebUI desde otro ordenador se experimentó un ralentización de la interfaz web.

2. Soluciones Implementadas

- Uso de entornos virtuales y versiones correspondientes de Python.
- Configurando ambos contenedores en la misma red de Docker.
- Se configuró el contenedor de Open WebUI permitiendo la conexión externa.

3. Recomendaciones para Futuros Despliegues

- Verificar las versiones de todas las dependencias antes de iniciar.
- Emplear contenedores para aislar la instalación.
- Mantener una bitácora de cambios y ajustes.
- Planear previamente la distribución de la red de contenedores y Docker

Material Recomendado

Para futuras instalaciones y uso de las herramientas se recomiendan las siguientes opciones para una mejor gestión, mantenimiento e inclusión de funcionalidades:

- Contenedor Portainer: Facilita de manera significativa la administración de contenedores Docker gracias a su interfaz intuitiva, ideal para usuarios que buscan simplificar despliegues y monitorización en entornos locales.
- Material y herramientas de Open-WebUI: Esta plataforma, respaldada por una comunidad activa, amplía las funcionalidades de los modelos de IA mediante código Python integrable. Permite conectar los modelos a navegadores, APIs externas y herramientas locales (ej. calculadoras), optimizando su versatilidad en tareas específicas. <https://openwebui.com/tools?query=>

- Prompts predefinidos: La comunidad ofrece plantillas de prompts estructurados, diseñados para automatizar procesos repetitivos y mejorar la calidad de las respuestas, superando en muchos casos los resultados obtenidos mediante prompts genéricos. <https://openwebui.com/prompts?query=>
- Modelos personalizados: Similar a funciones premium como ChatGPT Plus, Open-WebUI permite descargar y ajustar modelos de IA adaptados a necesidades particulares, eliminando la dependencia de suscripciones pagas. <https://openwebui.com/models?query=>
- Gestión de conocimientos: Aunque no existe un apartado dedicado, la comunidad habilita la configuración de flujos de trabajo e inyección de información contextual (ej. protocolos empresariales o datos técnicos), lo que enriquece la capacidad de respuesta del modelo según contextos definidos.

A2. Código generado en las pruebas iniciales

Los siguientes repositorios contienen el código generado durante las pruebas iniciales cuyas evaluaciones se recogen en las tablas de este anexo y se muestran en las gráficas del apartado 4.2.1 del capítulo 4. Los repositorios tienen archivos con el código generado, cada archivo tiene el nombre de la IA que lo generó.

En el caso de las pruebas 1 y 2 son archivos html debido a que son pruebas sobre aspectos de interfaz y estética.

Prueba 1

https://github.com/AlexCastillaC/TFG_codigo_pruebas_prueba1

Listado de archivos:

- ChatGPT-4o.html
- Claude 3.5 Sonnet.html
- Codestral 22B.html
- DeepSeek-R1.html
- OpenAI o1.html
- OpenAI o3-mini-high.html
- OpenAI o3-mini.html
- Phi4.html
- Qwen 2.5-coder.html

Tabla 27 Comparativa de Prueba 1 de Exploración Creativa

IA/Métrica	CyL	OD	GC	CD
ChatGPT-4o	8	7	8	6
OpenAI o1	8.5	7.5	9	8
OpenAI o3-mini	9	7.5	9	8
OpenAI o3-mini-high	8	8	10	9
DeepSeek R1	9	9	10	9
Claude 3.5 Sonnet	9	9	10	10
Phi4	6	7	8	7
Codestral 22B	7	5	8	6
Qwen 2.5-coder	8	7	9	7

Prueba 2

https://github.com/AlexCastillaC/TFG_codigo_pruebas_prueba2

Listado de archivos:

- ChatGPT-4o.html
- Claude 3.5 Sonnet.html
- Codestral 22B.html

- DeepSeek-R1.html
- OpenAI o1.html
- OpenAI o3-mini-high.html
- OpenAI o3-mini.html
- Phi4.html
- Qwen 2.5-coder.html

Tabla 28 Comparativa de Prueba 2 de Exploración Creativa

IA/Métrica	CyL	OD	GC	CD
ChatGPT-4o	8	8	10	8
OpenAI o1	9	9	9	8
OpenAI o3-mini	9	10	9	9
OpenAI o3-mini-high	9	9	10	9
DeepSeek R1	9	9	10	10
Claude 3.5 Sonnet	9	9	10	10
Phi4	7	6	10	6
Codestral 22B	8	5	10	3
Qwen 2.5-coder	9	8	10	8

Por otro lado, los archivos de las pruebas 3 y 4 son archivos txt divididos internamente según los elementos de Laravel generados al ser pruebas orientadas con la obtención de objetivos e implementación de funcionalidades.

Prueba 3

https://github.com/AlexCastillaC/TFG_codigo_pruebas_prueba3

Listado de archivos:

- ChatGPT-4o.txt
- Claude 3.5 Sonnet.txt
- Codestral 22B.txt
- DeepSeek-R1.txt
- OpenAI o1.txt
- OpenAI o3-mini-high.txt
- OpenAI o3-mini.txt
- Phi4.txt
- Qwen 2.5-coder.txt

Tabla 29 Comparativa de Prueba 3 de Resolución de Objetivos

IA/Métrica	CyL	OD	GC	CD	Nº de prompts
ChatGPT-4o	9	2	10	3	2
OpenAI o1	9	2	10	3	1
OpenAI o3-mini	9	3	10	4	1
OpenAI o3-mini-high	9	3	10	3	2

DeepSeek R1	9	2	10	3	2
Claude 3.5 Sonnet	9	2	10	3	2
Phi4	9	2	10	3	3
Codestral 22B	9	2	10	1	4
Qwen 2.5-coder	9	2	10	3	6

Prueba 4

https://github.com/AlexCastillaC/TFG_codigo_pruebas_prueba4

Listado de archivos:

- ChatGPT-4o.txt
- Claude 3.5 Sonnet.txt
- Codestral 22B.txt
- DeepSeek-R1.txt
- OpenAI o1.txt
- OpenAI o3-mini-high.txt
- OpenAI o3-mini.txt
- Phi4.txt
- Qwen 2.5-coder.txt

Tabla 30 Comparativa de Prueba 4 de Resolución de Objetivos

IA/Métrica	CyL	OD	GC	CD	Nºde prompts
ChatGPT-4o	9	2	10	3	3
OpenAI o1	9	2	10	3	2
OpenAI o3-mini	9	3	10	3	8
OpenAI o3-mini-high	9	2	10	3	3
DeepSeek R1	9	2	10	3	4
Claude 3.5 Sonnet	9	2	10	3	1
Phi4	9	2	10	3	9
Codestral 22B	6	1	3	2	5
Qwen 2.5-coder	8	2	9	3	3

A3. Especificaciones del CMS

Para la elaboración de las especificaciones se han tomado como punto de partida el proyecto docente de la asignatura así como los enunciados de las diferentes prácticas.

Tecnologías y Frameworks

El desarrollo del CMS se realizó utilizando las siguientes tecnologías y frameworks (Ver figura 43 y tabla 30):

Tabla 31. Resumen de Tecnologías y su Propósito en el Desarrollo Web

Tecnología/Framework	Propósito
HTML5	Estructura básica de las páginas web
CSS3	Gestión del diseño, estilo y apariencia del sitio web
JavaScript	Programación del lado del cliente para funcionalidades interactivas
AJAX	Operaciones de acceso a datos mediante servicios web sin recargar páginas
PHP	Desarrollo de aplicaciones del lado del servidor
MySQL	Gestión de la base de datos relacional
Laravel	Framework PHP para el desarrollo web, incluidas migraciones y seeders
Eloquent ORM	Creación y gestión de modelos de datos
Bootstrap	Biblioteca de componentes para el desarrollo web responsive

Arquitectura del sistema

El CMS para el Mercado Local San Mateo sigue una arquitectura basada en MVC (Modelo-Vista-Controlador) utilizando Laravel como framework backend. Esta arquitectura modular permite un desarrollo escalable y mantenible del CMS, asegurando un flujo lógico en la interacción entre usuarios y datos. La arquitectura del sistema está dividida en varios componentes clave (Ver figura 43):

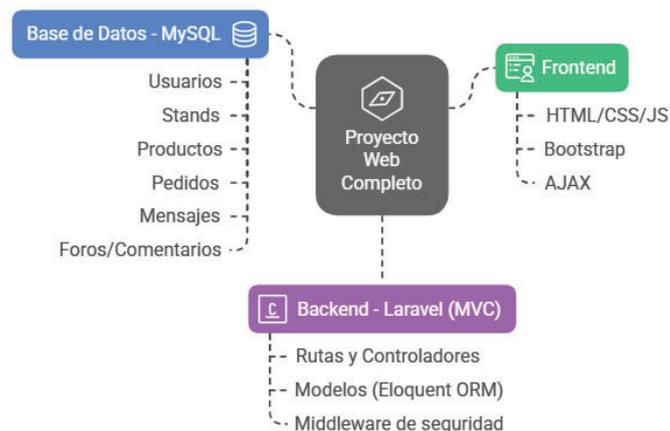


Figura 43. Diagrama Conceptual de la Arquitectura (Elaborado con IA²⁴)

²⁴ <https://app.napkin.ai/>

Capa de Presentación (Frontend)

- **Tecnologías utilizadas:** HTML5, CSS3, JavaScript, Bootstrap.
- **Responsabilidad:** Interfaz de usuario intuitiva y accesible para clientes, vendedores y proveedores.
- **Funcionalidades principales:**
 - Páginas de inicio, productos, carrito de compras y perfil de usuario.
 - Formularios de autenticación y gestión de cuenta.
 - Visualización de la red social integrada (foros, comentarios, mensajería).

Capa de Lógica de Negocio (Backend)

- **Framework:** Laravel (PHP)
- **Responsabilidad:** Procesar la lógica de negocio y gestionar la comunicación entre la interfaz de usuario y la base de datos.
- **Estructura basada en MVC:**
 - **Modelos:** Representan las entidades de la base de datos (usuarios, productos, pedidos, mensajes, etc.).
 - **Vistas:** Plantillas Blade que generan el HTML dinámico basado en datos del backend.
 - **Controladores:** Procesan las solicitudes del usuario, interactúan con los modelos y devuelven respuestas.

Capa de Persistencia (Base de Datos)

- **Gestor de base de datos:** MySQL.
- **Estructura basada en relaciones entre entidades:**
 - Usuarios (clientes, vendedores y proveedores).
 - Stands (puestos de vendedores).
 - Productos (asociados a stands o proveedores).
 - Pedidos (cliente a vendedor, con múltiples productos).
 - Mensajes (sistema de mensajería privada).
 - Foros y Comentarios (red social para la comunidad del mercado).
- Manejo de relaciones mediante **Laravel Eloquent ORM**.
- **Migraciones** y **seeders** para la generación y actualización de la base de datos.

Comunicación entre Componentes

- **Routing:** Laravel gestiona las rutas de la aplicación para diferenciar endpoints públicos y protegidos por autenticación.
- **Middleware:** Controla la seguridad y autorización de accesos según los roles de usuario.
- **API REST:** Exposición de datos mediante endpoints JSON para funciones como búsqueda de productos, gestión de pedidos y red social.

Seguridad y Autenticación

- Autenticación de usuarios con **Laravel Auth**.
- Middleware para roles (cliente, vendedor, proveedor).
- Protección de datos con encriptación de contraseñas y tokens CSRF.
- Políticas de acceso a recursos según permisos.

Integraciones y Extensibilidad

- **Integración de APIs externas:** Servicios como información meteorológica y métodos de pago.
- **API interna:** Posibilita comunicación entre módulos y futuras integraciones con otras plataformas.
- **Sistema modular:** Permite añadir nuevas funcionalidades sin afectar la estructura existente.

Roles de Usuario

El sistema contempla tres roles de usuario principales, cada uno con funcionalidades específicas (Ver tabla 31):

Tabla 32. Funcionalidades y Roles de Usuario

Funcionalidad / Capacidad	Cliente	Vendedor	Proveedor
Registro y gestión de perfil personal	X	X	X
Participación en foros del mismo rol	X	X	X
Comunicación privada con otros usuarios	X	X	X
Compra de productos ofertados por vendedores	X		
Gestión de información del puesto (nombre, clasificación, localización)		X	X
Venta de productos a clientes		X	
Compra de productos a proveedores		X	
Venta de productos exclusivamente a vendedores			X
Gestión de stock y precios de productos		X	X

Descripción de tablas principales

Tabla users (Usuarios): Almacena la información de los usuarios registrados en el sistema.

Campos principales:

- **id** (INT, PK) → Identificador único del usuario.
- **name** (VARCHAR) → Nombre completo del usuario.
- **email** (VARCHAR, UNIQUE) → Correo electrónico.
- **password** (VARCHAR) → Contraseña encriptada.
- **phone** (VARCHAR, NULL) → Teléfono de contacto.
- **role** (ENUM: **cliente**, **vendedor**, **proveedor**) → Rol del usuario.
- **profile_picture** (VARCHAR, NULL) → Imagen de perfil.
- **created_at**, **updated_at** (TIMESTAMP) → Control de fechas.

Relaciones:

- Un usuario (**vendedor/proveedor**) puede tener **cero o un puesto** (**stands**).
- Un usuario (**vendedor/proveedor**) puede publicar **múltiples productos** (**products**).

- Un usuario (**cliente/vendedor**) puede realizar **múltiples pedidos** (*orders*).
- Un usuario puede enviar/recibir múltiples **mensajes privados** (*messages*).
- Un usuario puede crear múltiples **foros y comentarios**.

Tabla *stands* (Puestos de Venta): Define los puestos de los vendedores en el mercado.

Campos principales:

- *id* (INT, PK) → Identificador único del puesto.
- *user_id* (INT, FK) → Relación con el vendedor propietario.
- *name* (VARCHAR) → Nombre del puesto.
- *description* (TEXT, NULL) → Descripción del puesto.
- *location* (VARCHAR) → Ubicación en el mercado.
- *category* (VARCHAR, NULL) → Categoría del puesto (Ej. frutas, artesanía).

Relaciones:

- Un **puesto pertenece a un usuario (vendedor/proveedor)** (*users*).
- Un **puesto puede contener múltiples productos** (*products*).

Tabla *products* (Productos): Representa los productos ofrecidos por vendedores o proveedores.

Campos principales:

- *id* (INT, PK) → Identificador único del producto.
- *user_id* (INT, FK) → Propietario del producto (vendedor o proveedor).
- *stand_id* (INT, FK, NULL) → Relación con un puesto de venta (opcional).
- *name* (VARCHAR) → Nombre del producto.
- *description* (TEXT, NULL) → Descripción del producto.
- *price* (DECIMAL) → Precio del producto.
- *stock* (INT) → Cantidad disponible.
- *category* (VARCHAR, NULL) → Categoría del producto.
- *is_available* (BOOLEAN) → Indica si el producto está activo.

Relaciones:

- Un **producto pertenece a un usuario (vendedor/proveedor)** (*users*).
- Un **producto puede estar asociado a un puesto** (*stands*).
- Un **producto puede estar en múltiples pedidos** (*order_items*).

Tabla *orders* (Pedidos de Clientes a Vendedores): Registra las compras realizadas en el sistema.

Campos principales:

- *id* (INT, PK) → Identificador único del pedido.
- *client_id* (INT, FK) → Usuario que realizó la compra.
- *vendor_id* (INT, FK) → Usuario que vendió los productos.

- `total` (DECIMAL) → Monto total del pedido.
- `status` (ENUM: `pendiente`, `procesando`, `enviado`, `entregado`, `cancelado`).
- `payment_method` (VARCHAR) → Método de pago.
- `payment_status` (ENUM: `pendiente`, `pagado`, `reembolsado`).
- `shipping_address` (TEXT, NULL) → Dirección de entrega.

Relaciones:

- Un **pedido pertenece a un cliente** (`users`).
- Un **pedido es atendido por un vendedor** (`users`).
- Un **pedido contiene múltiples productos** (`order_items`).

Tabla **order_items** (Detalles de Pedido): Relaciona los pedidos con los productos comprados.

Campos principales:

- `id` (INT, PK) → Identificador único del ítem.
- `order_id` (INT, FK) → Pedido al que pertenece.
- `product_id` (INT, FK) → Producto comprado.
- `quantity` (INT) → Cantidad del producto.
- `price` (DECIMAL) → Precio unitario en la compra.

Relaciones:

- Un **ítem de pedido pertenece a un pedido** (`orders`).
- Un **ítem de pedido hace referencia a un producto** (`products`).

Tabla **forums** (Foros de Discusión): Espacios de interacción entre usuarios del sistema.

Campos principales:

- `id` (INT, PK) → Identificador único del foro.
- `user_id` (INT, FK) → Usuario que creó el foro.
- `title` (VARCHAR) → Título del foro.
- `description` (TEXT, NULL) → Descripción del foro.
- `role_access` (ENUM: `cliente`, `vendedor`, `proveedor`).

Relaciones:

- Un **foro pertenece a un usuario** (`users`).
- Un **foro puede tener múltiples temas** (`topics`).

Tabla **messages** (Mensajes Privados entre Usuarios): Gestión de la comunicación privada en la plataforma.

Campos principales:

- `id` (INT, PK) → Identificador único del mensaje.
- `sender_id` (INT, FK) → Remitente del mensaje.

- `receiver_id` (INT, FK) → Destinatario del mensaje.
- `content` (TEXT) → Contenido del mensaje.
- `read_at` (TIMESTAMP, NULL) → Fecha de lectura.

Relaciones:

- Un mensaje tiene un remitente (`users`).
- Un mensaje tiene un destinatario (`users`).

Relaciones entre entidades

Las principales relaciones en el modelo son:

1. **Usuarios-Stands**→ Un vendedor o proveedor puede tener un puesto (1:0/1).
2. **Usuarios-Productos**→ Un vendedor o proveedor puede tener varios productos (1:N).
3. **Usuarios-Pedidos**→Un cliente puede hacer múltiples pedidos (1:N).
4. **Pedidos-Productos**→Un pedido puede contener múltiples productos (M:N a través de `order_items`).
5. **Usuarios-Foros**→Un usuario puede crear múltiples foros (1:N).
6. **Usuarios-Mensajes**→Un usuario puede enviar/recibir múltiples mensajes (1:N)

Requerimientos funcionales

En la siguiente tabla se muestran las funcionalidades que serán definidas para el proyecto divididas en categorías.

Tabla 33. Bloques Funcionales

Autenticación y Gestión de Perfiles	Registro de usuarios con validación
	Inicio de sesión seguro
	Gestión de perfil (foto, nombre, teléfono, etc.)
	Panel de control específico según rol
Tienda Virtual	Catálogo de productos
	Carrito de compra
	Pasarela de pago
	Gestión de stock
	Historial de compras/ventas
Red Social	Mensajería privada entre usuarios
	Foros específicos por rol
Gestión de Productos	Alta, baja y modificación de productos
	Categorización, búsqueda y filtrado de productos
	Gestión de precios
Integración API	Mostrar información meteorológica de San Mateo
	API propia para consumo interno
	Posibles integraciones con servicios externos relevantes

A4. Repositorios del código generado y prompts evaluados

Este anexo contiene los enlaces a los repositorios que contienen el código generado por las IAs seleccionadas tras las pruebas iniciales del apartado 4.2.1 y cuya evaluación se muestra en el apartado 5.2. Además de los repositorios con las hojas de evaluación de los prompts que se ejecutaron para generar código, que contienen los prompts usados, su versión nueva y la evaluación de ambos.

Código

Estos repositorios contienen los proyectos hechos en Laravel cuyo código fue generado por IA, los directorios donde se encuentra la gran cantidad del código generado son:

IAs

- Claude Sonnet 3.5: https://github.com/AlexCastillaC/TFG_CMS_Claude
- Phi-4: https://github.com/AlexCastillaC/TFG_CMS_Phi4
- ChatGPT o3-mini-high: http://github.com/AlexCastillaC/TFG_CMS_ChaGPTo3mh
- Qwen 2.5-coder: https://github.com/AlexCastillaC/TFG_CMS_Qwen-2.5-coder
- DeepSeek-R1: https://github.com/AlexCastillaC/TFG_CMS_DeepSeek-R1

Listado de archivos de ejemplo:

```
TFG_CMS_DeepSeek-R1-main
├── app
│   ├── Http
│   │   ├── Controllers
│   │   │   ├── ApiController.php
│   │   │   ├── AuthController.php
│   │   │   ├── CartController.php
│   │   │   ├── CheckoutController.php
│   │   │   ├── ClientController.php
│   │   │   ├── ClimaApiController.php
│   │   │   ├── CommentController.php
│   │   │   ├── Controller.php
│   │   │   ├── ForumController.php
│   │   │   ├── HomeController.php
│   │   │   ├── MessageController.php
│   │   │   ├── PaymentController.php
│   │   │   ├── ProductController.php
│   │   │   ├── ProviderController.php
│   │   │   ├── ShoppingCartController.php
│   │   │   ├── StandController.php
│   │   │   ├── StripeController.php
│   │   │   ├── TopicController.php
│   │   │   └── VendorController.php
```

```

| | | └─ Auth
| | |   └─ ConfirmPasswordController.php
| | |   └─ ForgotPasswordController.php
| | |   └─ LoginController.php
| | |   └─ RegisterController.php
| | |   └─ ResetPasswordController.php
| | |   └─ VerificationController.php
| | └─ Middleware
| |   └─ CheckRole.php
| └─ Models
|   └─ Comment.php
|   └─ Forum.php
|   └─ Message.php
|   └─ Order.php
|   └─ OrderItem.php
|   └─ Product.php
|   └─ Stand.php
|   └─ Topic.php
|   └─ User.php
|
└─ database
  └─ migrations
    └─ 2014_10_12_000000_create_users_table.php
    └─ 2014_10_12_100000_create_password_resets_table.php
    └─ 2014_10_12_100000_create_password_reset_tokens_table.php
    └─ 2019_08_19_000000_create_failed_jobs_table.php
    └─ 2019_12_14_000001_create_personal_access_tokens_table.php
    └─ 2025_04_07_155415_create_stands_table.php
    └─ 2025_04_07_160133_create_products_table.php
    └─ 2025_04_07_160730_create_orders_table.php
    └─ 2025_04_07_161050_create_order_items_table.php
    └─ 2025_04_07_161235_create_forums_table.php
    └─ 2025_04_07_162333_create_topics_table.php
    └─ 2025_04_07_162512_create_comments_table.php
    └─ 2025_04_07_163247_create_messages_table.php

└─ public
  └─ js
    └─ feature-clima.js
    └─ feature-productos.js
  └─ resources
    └─ views
      └─ home.blade.php
      └─ shopping-cart.blade.php
      └─ welcome.blade.php
      └─ auth
        └─ login.blade.php
        └─ register.blade.php

```

```

| | | |— verify.blade.php
| | | |— passwords
| | | |— confirm.blade.php
| | | |— email.blade.php
| | | |— reset.blade.php
| | |— cart
| | | |— summary.blade.php
| | |— checkout
| | | |— index.blade.php
| | |— layouts
| | | |— app.blade.php
| | |— products
| | | |— create.blade.php
| | | |— edit.blade.php
| | | |— index.blade.php
| | | |— stock.blade.php
| | |— profile
| | | |— client.blade.php
| | | |— provider.blade.php
| | | |— vendor.blade.php
| | |— social
| | | |— messaging
| | | |— index.blade.php
| | | |— show.blade.php
| | | |— threads
| | | |— create-topic.blade.php
| | | |— create.blade.php
| | | |— index.blade.php
| | | |— show.blade.php
| | | |— topic-show.blade.php
| | |— stands
| | | |— create.blade.php
| | | |— edit.blade.php
| | | |— index.blade.php
| | | |— Seestands.blade.php
|
|— routes
| |— api.php
| |— web.php

```

Prompts

Estos repositorios contienen las hojas de análisis de los prompts usados para generación de código, en el caso de Claude al ser la primera IA en usarse en el trabajo sus hojas de análisis están orientadas a aspectos concretos de Laravel, mientras que el resto de los modelos a las fases donde se usó:

Claude Sonnet 3.5: https://github.com/AlexCastillaC/Prompts_Claude-Sonnet-3.5

Listado de archivos:

- P001-Laravel Environment.pdf
- P002-AuthenticationUser.pdf
- P003-ProductManagement.pdf
- P004-ControllertoRoutes.pdf
- P005-ModelToController.pdf
- P006-Create_EditStands.pdf
- P007-ControllertoViews.pdf
- P008-ChageViewElements.pdf
- P009-ModifyCategoryCode.pdf
- P0010-ProductMiddleware.pdf
- P0011-RefactoringCode.pdf
- P0012-Controllerfunctiontoview.pdf
- P0013-ProductsEdit.pdf
- P0014-ProviderProductManagment.pdf
- P0015-Fase4.pdf
- P0016-CartViews.pdf
- P0017-OrdersViews.pdf
- P0018-OrderShow.pdf
- P0019-SocialMediaPart.pdf
- P0020-APIINTEGRATION.pdf
- P0021-TOASTNOTIFICATION.pdf

Phi-4: https://github.com/AlexCastillaC/Prompts_Phi-4

Listado de archivos:

- P001-1 de Phi4.pdf
- P002-2 de Phi4.pdf
- P003-Fase3.pdf
- P004-Fase4.pdf
- Fase 5 Phi4.pdf
- Fase 6 Phi4.pdf

ChatGPTo3-mini-high: https://github.com/AlexCastillaC/Prompts_ChatGPT-o3-mini-high

Listado de archivos:

- P001-Fase 1.pdf
- Fase 2.pdf
- Fase 3.pdf
- P004-Fase 4.pdf
- Fase 5.pdf
- Fase 6.pdf

Qwen 2.5-coder: https://github.com/AlexCastillaC/Prompts_Qwen-2.5-coder

Listado de archivos:

- P001-Fase 1 Qwen.pdf
- P002-Fase 2 Qwen.pdf
- P003-Fase 3 Qwen.pdf
- P004-Fase 4 Qwen.pdf
- P005-Fase 5 Qwen.pdf
- P006-Fase 6 Qwen.pdf

DeepSeek-R1: https://github.com/AlexCastillaC/Prompts_DeepSeek-R1

Listado de archivos:

- P001-Fase 1 DeepSeek-R1.pdf
- P002-Fase 2 DeepSeek-R1.pdf
- P003-Fase 3 DeepSeek-R1.pdf
- P004-Fase 4 DeepSeek-R1.pdf
- P005-Fase 5 DeepSeek-R1.pdf
- P006-Fase 6 DeepSeek-R1.pdf

A5. Prompts e IAs usadas para la redacción normativa ULPGC

IA Generativas utilizadas durante la elaboración de este documento:

- Claude 3.5, 3.7 y 4 Sonnet
- ChatGPT 4o, Open AI o1, o3 mini y o3 mini-high
- Napkin AI
- Gemini 2.5 Flash
- DeepSeek-R1

Ejemplos de prompts usados:

Claude

Yo: Dame una lista detallada de todo lo que puedo hacer con Claude en versión gratuita

Respuesta: Te ayudo a entender lo que puedes hacer conmigo. Ten en cuenta que no conozco los detalles específicos de los planes o límites gratuitos actuales, así que te sugiero consultar <https://support.anthropic.com> para esa información.

Lo que sí puedo decirte es que puedo ayudarte con una amplia gama de tareas como:

1. Análisis y explicaciones
 - Analizar textos y documentos
 - Explicar conceptos complejos de forma clara y accesible
 - Ayudar con tareas académicas y de investigación
 - Ofrecer diferentes perspectivas sobre un tema
2. Programación y tecnología
 - Escribir, revisar y depurar código en múltiples lenguajes
 - Explicar conceptos de programación
 - Ayudar con problemas técnicos
 - Generar documentación técnica
3. Escritura y creatividad
 - Ayudar con la redacción y edición de textos
 - Escribir historias creativas
 - Generar ideas y lluvia de ideas
 - Traducir entre diferentes idiomas
4. Matemáticas y análisis cuantitativo
 - Resolver problemas matemáticos paso a paso

- Ayudar con estadísticas y análisis de datos
 - Explicar conceptos matemáticos
5. Asistencia práctica
- Ayudar a organizar ideas y proyectos
 - Ofrecer sugerencias y recomendaciones
 - Responder preguntas sobre diversos temas
 - Analizar imágenes y ayudar a entender su contenido

¿Hay algún tipo específico de tarea con la que te gustaría que te ayudara?

DeepSeek-R1

Yo: Mejora la redacción del texto: En proyectos y empresas determinar el coste de sus procesos e investigar formas de abaratar los es fundamental en la economía, en los servicios relacionados con la IA no es una excepción. Entender y cuantificar el coste del uso de inteligencia artificial será fundamental en todo proceso que utilice esta herramienta, con este objetivo se muestran una serie de gráficas que muestran una relación capacidad/coste en la IA.

Respuesta: ****Texto mejorado:****En el ámbito económico, determinar los costes de los procesos e implementar estrategias para optimizarlos resulta fundamental en proyectos y empresas. Los servicios vinculados a la inteligencia artificial (IA) no son la excepción. Comprender y cuantificar el coste asociado al uso de esta tecnología se convierte en un pilar esencial para cualquier proceso que la incorpore. Con este objetivo, se presentan una serie de gráficos que analizan la relación entre capacidad y coste en el desarrollo y aplicación de sistemas de IA.

****Detalles de las mejoras:****

1. ****Estructura más fluida:**** Se reorganizaron las ideas para una lectura coherente, vinculando economía, proyectos empresariales y IA.
2. ****Precisión léxica:**** Se reemplazaron términos vagos (*formas de abaratar los*) por otros más específicos (*implementar estrategias para optimizarlos*).
3. ****Eliminación de redundancias:**** Se evitó repetir "muestran" y se ajustó el enfoque hacia el análisis (*gráficos que analizan*).
4. ****Tono formal y técnico:**** Se emplearon expresiones como *pilar esencial* y *sistemas de IA* para profesionalizar el texto.
5. ****Claridad en el propósito:**** Se enfatizó el vínculo entre los gráficos y su utilidad práctica en la toma de decisiones.

Napkin AI

Yo:

Tecnología/Framework	Propósito
HTML5	Estructura básica de las páginas web
CSS3	Gestión del diseño, estilo y apariencia del sitio web
JavaScript	Programación del lado del cliente para funcionalidades interactivas
AJAX	Operaciones de acceso a datos mediante servicios web sin recargar páginas
PHP	Desarrollo de aplicaciones del lado del servidor
MySQL	Gestión de la base de datos relacional
Laravel	Framework PHP para el desarrollo web, incluidas migraciones y seeders
Eloquent ORM	Creación y gestión de modelos de datos
Bootstrap	Biblioteca de componentes para el desarrollo web responsive

Elaboró:



ChatGPT-4o

Yo: Búscame cuales son los mejores inteligencias artificiales que se puedan acceder vía web para programar en lenguajes relacionados con la programación web: HTML, CSS, PHP, Javascript y SQL

Respuesta: Por supuesto, a continuación te presento algunas de las mejores herramientas de inteligencia artificial accesibles vía web que pueden asistirte en la programación de lenguajes relacionados con el desarrollo web, como HTML, CSS, PHP, JavaScript y SQL:

1. **GitHub Copilot:** Desarrollado por GitHub en colaboración con OpenAI, Copilot actúa como un programador de pares virtual impulsado por IA. Sugiere líneas completas o bloques de código mientras escribes, acelerando el proceso de codificación. Es compatible con una amplia gama de lenguajes de programación, incluyendo JavaScript y PHP. (unite.ai)

2. **Tabnine:** Este asistente de IA ayuda a los desarrolladores sugiriendo fragmentos de código y completando líneas de código en tiempo real. Es compatible con múltiples lenguajes, como JavaScript, HTML y CSS, y se integra fácilmente con diversos entornos de desarrollo integrados (IDEs). (geekflare.com)
3. **AskCodi:** Una herramienta impulsada por IA que asiste en la generación de código, documentación y pruebas. Soporta una variedad de lenguajes de programación, incluyendo PHP y JavaScript, y se integra con editores populares como Visual Studio Code y Sublime Text. (geekflare.com)
4. **AI Helper Bot:** Un generador de consultas SQL basado en IA diseñado para simplificar la creación de consultas, incluso para usuarios sin experiencia previa en SQL. Utiliza GPT-4 para generar consultas precisas y admite múltiples idiomas. (copilot.live)
5. **CodeWP:** Un generador de código impulsado por IA diseñado específicamente para desarrolladores de WordPress. Ayuda en la generación de código PHP y JavaScript, agilizando los flujos de trabajo y mejorando la seguridad en el desarrollo de WordPress. (copilot.live)

Estas herramientas pueden mejorar significativamente tu eficiencia y productividad al trabajar con lenguajes de programación web. Te recomiendo explorar cada una para determinar cuál se adapta mejor a tus necesidades específicas.

Para obtener una visión más detallada de algunas de estas herramientas, puedes consultar el siguiente video:

[5 herramientas IA que PROGRAMAN POR TI \[GRATIS\]](#)

OpenAI o1

Yo: Dame información sobre los parámetros que se pueden ajustar en Open-WebUI para la respuesta de los modelos

Respuesta: Open-WebUI permite ajustar numerosos parámetros que afectan directamente la forma en que los modelos generan sus respuestas. Estos ajustes se configuran principalmente al editar el “modelfile” en la sección de Models y se dividen en dos grandes categorías:

1. Parámetros Generales del Modelfile

- **System Prompt:** Define el comportamiento base del modelo al iniciar la conversación.
- **Prompt Suggestions:** Sugerencias predefinidas que aparecen al iniciar un chat.
- **Documents, Tools, Filters y Actions:** Permiten integrar funcionalidades adicionales y adaptar la interacción según la tarea.

2. Parámetros de Generación de Texto

Estos parámetros se pasan a la API subyacente (por ejemplo, a Ollama u otros motores) y permiten afinar la respuesta del modelo:

- **num_ctx:** Define la longitud del contexto (número máximo de tokens que el modelo "recuerda").
- **Temperature:** Controla el grado de aleatoriedad; valores bajos hacen respuestas más deterministas, mientras que valores altos fomentan la creatividad.
- **Top-K:** Limita la selección de tokens a los K más probables en cada paso.
- **Top-P (Nucleus Sampling):** Selecciona tokens hasta cubrir un porcentaje acumulado de probabilidad, ajustando así la diversidad de la salida.
- **Stop Sequences:** Permite definir secuencias de texto en las que se detendrá la generación (por ejemplo, para separar roles o finalizar una respuesta).
- **Max Tokens (num_predict):** Establece el número máximo de tokens a generar en la respuesta.
- **Seed:** Para lograr reproducibilidad en las respuestas fijando una semilla.
- **Parámetros Avanzados:**
 - Opciones como **mirostat** (y sus ajustes como Eta y Tau) que actúan como "termostatos" de creatividad,
 - Penalizaciones de frecuencia para evitar repeticiones,
 - Configuraciones como **repeat last N**, **num_batch** y **num_keep** que permiten ajustar cómo se mantiene el contexto y se gestionan múltiples salidas.
- **Ajustes específicos para Ollama:** Por ejemplo, habilitar o deshabilitar el uso de archivos de memoria mapeada (**use_mmap**) o bloquear la memoria (**use_mlock**), así como definir el número de hilos (**num_thread**) o GPUs (**num_gpu**) a utilizar.

Estos ajustes permiten personalizar el comportamiento del modelo para diferentes escenarios, optimizando tanto la calidad como el rendimiento de la generación de texto.

Para más detalles técnicos y ejemplos de cada parámetro, puedes consultar la [documentación oficial de Open-WebUI sobre models](#) y la discusión avanzada de parámetros en GitHub (github.com)).

OpenAI-o3-mini

Yo: Dame un párrafo explicando que es Github Copilot y sus características en programación

Respuesta: GitHub Copilot es un asistente de programación basado en inteligencia artificial desarrollado por GitHub y OpenAI que ayuda a los desarrolladores a escribir código de forma más rápida y eficiente, sugiriendo autocompletados y generando funciones completas en tiempo real a partir del contexto del proyecto (es.wikipedia.org). Utiliza modelos avanzados de lenguaje, como OpenAI Codex, entrenados con millones de líneas de código público, lo que le permite comprender diversos lenguajes (Python, JavaScript, TypeScript, Ruby, Go, entre otros) y se integra en entornos de desarrollo populares como Visual Studio Code, Visual Studio, Neovim y los IDEs de JetBrains (docs.github.com). Además, transforma comentarios en código ejecutable y automatiza tareas repetitivas, potenciando la creatividad y productividad de los programadores, aunque siempre es necesario que el desarrollador revise y ajuste las sugerencias para garantizar la calidad y seguridad del código (xataka.com).

Gemini 2.5 Flash

Yo: Hay algo estipulado por el colegio Colegio Oficial De Ingenieros Técnicos De Telecomunicaciones?

Respuesta: Sí, el **Colegio Oficial de Ingenieros Técnicos de Telecomunicación (COITT)** ha tenido históricamente criterios orientativos de honorarios. Sin embargo, debido a la normativa de defensa de la competencia, los colegios profesionales en España tienen **limitaciones para establecer baremos de honorarios obligatorios u orientativos**, más allá de lo permitido para la tasación de costas en procedimientos judiciales.

Según la información que he encontrado:

- El COITT, en su web, menciona que los **honorarios son libres y responden al libre acuerdo entre el profesional y su cliente**. Anteriormente publicaban baremos orientativos, pero esto se ha eliminado siguiendo directivas europeas.

- El Colegio sí enumera algunos **conceptos a tener en cuenta en el cálculo de honorarios**, como costes directos del ingeniero y colaboradores, gastos generales, derechos de visado, etc. Esto proporciona una guía sobre qué factores considerar al presupuestar, pero no establece tarifas específicas.
- La **Comisión Nacional de los Mercados y la Competencia (CNMC)** ha insistido en que los colegios profesionales no deben establecer baremos orientativos de honorarios, con la excepción de criterios para la tasación de costas.

Por lo tanto, **no existe una guía de tarifas estipulada por el COITT para la redacción de documentos como el presupuesto de un TFG** que puedas seguir de forma oficial. Los honorarios en el ámbito profesional se basan en el acuerdo libre entre las partes.

En tu caso, como estudiante valorando tu propio trabajo, las referencias que te di anteriormente (tarifas de redacción técnica o consultoría junior como punto de partida) podrían ser más útiles para estimar el valor de tu tiempo.

¿Te gustaría que intentáramos acotar una estimación basada en el tiempo que crees que te llevará redactar esta sección?