Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA
DE SEGOVIA

Grado en Ingeniería Informática
de Servicios y Aplicaciones

Desarrollo de un sistema para la autoevaluación de modelos conceptuales de datos utilizando inteligencia artificial generativa

---

Alumno: Inés de Castro Heras

Tutores: Miguel Ángel Martínez Prieto
Clara Gándara González

Fecha: 26 de junio de 2025

Desarrollo de un sistema para la autoevaluación de modelos conceptuales de datos utilizando inteligencia artificial generativa

Inés de Castro Heras

26 de junio de 2025

A mi madre.
Sin ella nada habría sido posible.

Agradecimientos

Quisiera agradecer profundamente a todas las personas que me han acompañado no sólo en la realización de este Trabajo de Fin de Grado sino también durante estos últimos cinco años, sin vosotros nunca habría llegado hasta aquí.

En primer lugar, agradezco a mis tutores, Miguel Ángel Martínez Prieto y Clara Gándara González, por su constante apoyo, orientación y paciencia durante todo el proceso. Su dedicación y esfuerzo han sido fundamentales en este proyecto.

A mi familia, en especial a mi madre y a mi hermano, por su amor incondicional y por motivarme a seguir adelante incluso en los momentos más difíciles. Siempre les estaré agradecida por ser el pilar fundamental de mi vida.

A todos mis amigos, en particular los que me ha dado esta carrera tan bonita, por compartir este camino, por su compañía, consejos y por los buenos momentos que me han ayudado a sobrellevar el esfuerzo diario. Me habéis hecho muy feliz estos años y espero que siga siendo así toda la vida.

Por último, a Nico por haber sido y seguir siendo mi mayor apoyo, mi lugar seguro y mi casa cuando estaba muy lejos de ella.

A todos, gracias de corazón.

Resumen

El objetivo de este proyecto es el desarrollo de LearnER, una aplicación basada en inteligencia artificial generativa dirigida al aprendizaje y evaluación de diagramas Entidad-Relación (E-R), fundamentales en el diseño conceptual de bases de datos. LearnER permite a los estudiantes cargar sus propios diagramas, reconoce automáticamente todos sus componentes y genera un *feedback* personalizado que ayuda al alumno a identificar sus errores y mejorar su comprensión de manera autónoma. La aplicación ha demostrado una alta capacidad para identificar correctamente los elementos de los modelos E-R manuscritos, con una precisión media superior al 90 %. Además, reproduce la calificación humana con un error medio de apenas 0,14 sobre 1, lo que demuestra una fiabilidad bastante alta en la corrección automática.

Palabras claves: inteligencia artificial generativa, diagramas entidad-relación, bases de datos, *feedback* automático, aprendizaje personalizado.

Abstract

The aim of this project is to develop LearnER, a generative-AI-based application designed for the learning and assessment of Entity–Relationship (E–R) diagrams, which are fundamental to the conceptual design of databases. LearnER allows students to upload their own diagrams, automatically recognises all their components, and provides personalised *feedback* that helps learners identify their mistakes and enhance their understanding independently.

The application has shown a strong ability to correctly identify the elements of handwritten E–R models, achieving an average accuracy above 90 %. Moreover, it reproduces human grading with a mean error of only 0.14 on a 1-point scale, demonstrating a high degree of reliability in automatic assessment.

Keywords: generative artificial intelligence, entity-relationship diagrams, databases, automatic feedback, personalised learning. .

Índice general

Índice de figuras

Índice de tablas

Parte I

Descripción del proyecto

Capítulo 1

Introducción

En la actualidad, la inteligencia artificial (IA) ha transformado radicalmente diversos aspectos de nuestras vidas. La IA generativa, en particular, ha demostrado ser un recurso muy útil para tareas como crear código, resolver problemas matemáticos o incluso mantener conversaciones prácticamente iguales a las que podrías tener con una persona. En particular, el avance exponencial de estas tecnologías ha hecho que se convierta en una herramienta muy poderosa para el aprendizaje, permitiendo que este sea más eficiente y se adapte mejor a las necesidades individuales de cada persona.

En este contexto, este proyecto propone el desarrollo de una aplicación orientada a estudiantes de bases de datos, cuyo propósito principal es facilitar el aprendizaje del diseño conceptual a través del diagrama entidad-relación. Un diagrama entidad-relación es una representación gráfica del diseño conceptual de una base de datos, en el que se modelan entidades, atributos y relaciones para estructurar la información de manera organizada y comprensible. En la Figura 1.1 se presenta un ejemplo de diagrama ER manuscrito. Estos diagramas son fundamentales en la etapa de diseño de bases de datos, ya que permiten a los desarrolladores y analistas definir correctamente la estructura de los datos antes de su implementación en un sistema de gestión de bases de datos.

Aunque el desarrollo de un modelo ER pueda parecer sencillo, a los estudiantes suele resultarles una tarea muy complicada al principio, especialmente porque deben identificar los requisitos de información y luego abstraerlos en forma de entidades, relaciones y atributos. Por este motivo, esta aplicación, llamada LearnER, será de gran valor tanto para estudiantes, como para profesores de asignaturas de bases de datos ya que favorecerá el aprendizaje de los alumnos. La aplicación propuesta permitirá a los estudiantes cargar tanto su propio diagrama entidad-relación como el proporcionado por el profesor, que actúa como referencia. Utilizando IA Generativa se analizarán ambos diagramas y se generará un *feedback* que ayudará al estudiante a evaluar y corregir su diseño conceptual.

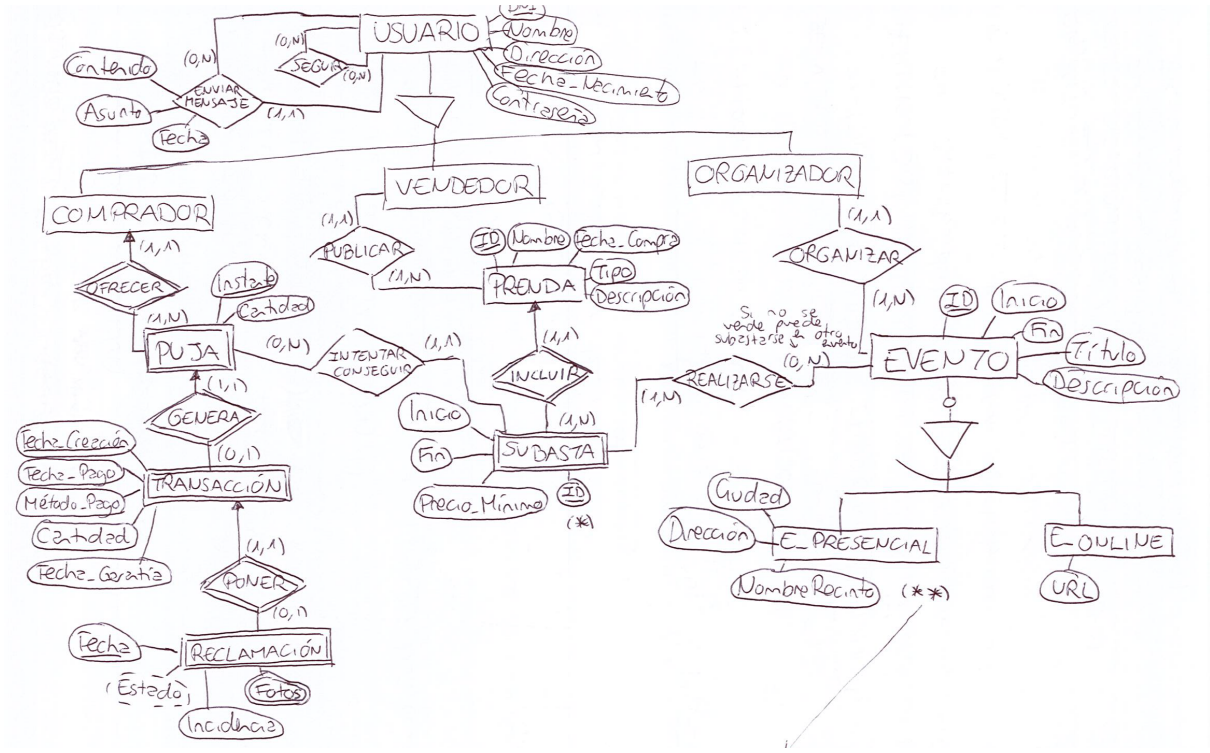

Figura 1.1: Ejemplo de diagrama ER manuscrito.

1.1. Planteamiento del problema (*Problem Statement*)

El planteamiento del problema o *problem statement* especifica una descripción de alto nivel del problema que se va a abordar en el TFG. Para ello, es necesario identificar la situación a la que se aspiraría en un entorno ideal y compararla con la realidad actual, ya que eso ayuda a determinar la brecha existente y las consecuencias que esta tiene en el entorno de negocio del proyecto. Sobre este análisis inicial, se determina la propuesta que se hace en este TFG y que consolida su objetivo principal.

- IDEAL: El aprendizaje del diseño conceptual de datos requiere de una práctica continuada por parte del estudiante. Lo ideal sería que los alumnos, tras realizar un ejercicio recibieran un *feedback* detallado que les ayude a identificar sus errores y a fijar los conocimientos adquiridos. Esto favorecería notablemente a su aprendizaje.

- REALIDAD: Para un profesor es inviable corregir todos los ejercicios que hacen todos sus alumnos y en caso de poder hacerlo, probablemente esto le llevaría mucho tiempo, haciendo que los alumnos no reciban *feedback* de parte de su trabajo o lo reciban después de demasiado tiempo lo que puede ser mucho menos útil para ellos, pues en la mayoría de los casos incluso habrán olvidado lo que ellos mismos pensaron.

- CONSECUENCIAS: Como consecuencia de lo que hemos planteado los profesores tienen una alta carga de trabajo y los alumnos tienen más dificultades en su aprendizaje al no recibir *feedback* de su trabajo o recibirlo después de demasiado tiempo.

- PROPUESTA: Como solución al problema planteado, se propone una aplicación basada en inteligencia artificial generativa que sea capaz de proporcionar *feedback* inmediato para los estudiantes sobre los modelos ER que construyan.

1.2. Objetivos

En este proyecto se propone el desarrollo de un tutor inteligente para apoyar el aprendizaje del diseño conceptual de bases de datos, proporcionando retroalimentación automatizada sobre modelos entidad-relación creados por los estudiantes. Se objetivos del proyecto en la Tabla 1.1.

OBJ-ID	**Objetivos**
OBJ-01	Implementar un módulo de interpretación de modelos entidad-relación manuscritos utilizando IA Generativa.
OBJ-02	Desarrollar un sistema de comparación flexible de modelos ER.
OBJ-03	Diseñar un mecanismo de generación de *feedback* estructurado, alineado con criterios de aceptación predefinidos, para proporcionar una evaluación objetiva del desempeño del estudiante.
OBJ-04	Implementar un sistema de generación de *feedback* cualitativo con un modelo de lenguaje (LLM), que explique el estado del aprendizaje del estudiante y ofrezca recomendaciones personalizadas para mejorar.

Tabla 1.1: Objetivos.

1.3. Condicionantes

En este apartado se detallarán los condicionantes del proyecto, es decir, posibles limitaciones que afecten a su desarrollo. Estas restricciones se detallan en la Tabla 1.2.

COND-ID	Condicionantes
COND-01	El esfuerzo medido en tiempo de este proyecto debe ser de aproximadamente 300 o 360 horas.
COND-02	Las correcciones generadas por la inteligencia artificial pueden ser poco detalladas o incluso erróneas.
COND-03	El modelo de lenguaje puede no ser capaz de identificar todos los elementos de los diagramas entidad relación.
COND-04	El análisis de las imágenes puede consumir más *tokens* de los esperados suponiendo un gasto demasiado elevado para poder afrontarlo.
COND-05	La API de OpenAI reserva un número máximo de *tokens* tanto para procesar la entrada como para generar la salida y este número podría ser demasiado bajo para las tareas que necesitamosllevar a cabo.

Tabla 1.2: Restricciones.

1.4. Alcance del proyecto

En esta subsección se definirá el alcance utilizando un diagrama de flujo o *workflow* que definirá el flujo de datos sobre el que se construirá nuestro sistema. El fujo consta de 6 etapas como se aprecia en la Figura 1.2.

A continuación, se explicarán de forma detallada las etapas del workflow de la Figura 1.2. Cabe destacar que todas las etapas que demanden el uso de IA generativa se llevarán a cabo a través de los recursos proporcionados por la API de OpenAI 3.4.1.

1. **Carga de imágenes y preprocesamiento:** En esta etapa el usuario sube la imagen de su diagrama ER, el sistema la guarda y después le aplica técnicas de preprocesamiento que facilitaran el procesamiento de la imagen en etapas posteriores.

2. **Identificación de las entidades:** Se identifican las entidades y sus atributos mediante inteligencia artificial generativa.

3. **Validación de las entidades:** El usuario valida mediante una interfaz web las entidades y sus atributos, permitiendole hacer todos los cambios que necesite.

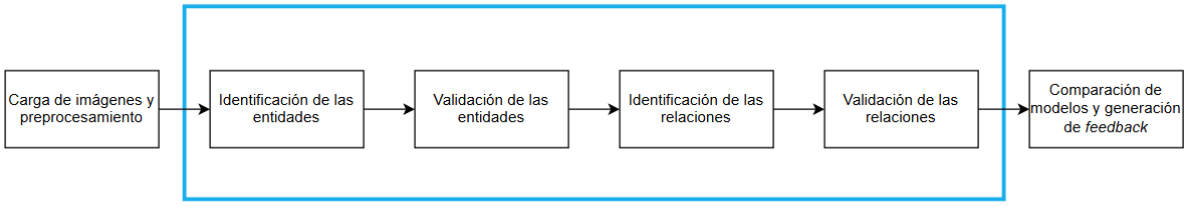

Figura 1.2: *Workflow* del sistema.

4. **Identificación de las relaciones:** Se identifican las relaciones, sus participantes y sus atributos mediante inteligencia artificial generativa.

5. **Validación de las relaciones:** El usuario valida mediante una interfaz web las relaciones, sus participantes y sus atributos, permitiendole hacer todos los cambios que necesite.

6. **Comparación de modelos y generación de *feedback*:** Finalmente, utilizando inteligencia artificial generativa se comparan por un lado los modelos del profesor y del alumno y por otro lado el modelo del alumno y el enunciado, para obtener dos tipos de *feedback*: un *feedback* cualitativo y un *feedback* cuantitativo.

1.5. Estructura de la memoria

Esta memoria se organiza en cuatro partes bien diferencias, de acuerdo con las características de este proyecto.

La primera parte, **"Descripción del proyecto"**, presenta el contexto general, la planificación y los antecedentes necesarios para entender el desarrollo del trabajo. Incluye los siguientes capítulos:

- **Capítulo 1. Introducción:** Expone el planteamiento del problema, los objetivos del proyecto, los condicionantes existentes, el alcance del proyecto y, finalmente, describe la estructura de la memoria.

- **Capítulo 2. Gestión del proyecto:** Detalla la metodología seguida, la planificación temporal del proyecto dividida en *sprints*, el presupuesto estimado y un balance tanto temporal como económico.

- **Capítulo 3. Antecedentes:** Describe el entorno de negocio en el que se enmarca el proyecto, el estado del arte con los trabajos relacionados, y se realiza una revisión sobre la Inteligencia Artificial Generativa y los fundamentos técnicos relevantes.

La segunda parte, **"Desarrollo de la solución"**, aborda el análisis, el diseño, la implementación y las pruebas del sistema desarrollado. Está compuesta por los siguientes capítulos:

- **Capítulo 4. Análisis:** Identifica los actores del sistema, los requisitos de usuario y los requisitos funcionales y no funcionales.

- **Capítulo 5. Diseño:** Presenta la arquitectura física y lógica de la solución, el modelo lógico de datos y el diseño de la interfaz de usuario mediante bocetos.

- **Capítulo 6. Implementación:** Explica las herramientas utilizadas, el desarrollo de los *prompts* utilizados, la generación de *feedback* y el estilo *frontend* implementado.

- **Capítulo 7. Pruebas:** Incluye la descripción de las pruebas realizadas para validar el funcionamiento de la solución.

La tercera parte, **"Resultados"**, expone los resultados de las pruebas llevadas a cabo para la aceptación del sistema construido y expone las conclusiones y trabajo futuro de este proyecto. Esta parte está formada por los siguientes capítulos:

- **Capítulo 8. Aceptación:** Evalúa la aceptación del sistema tanto en la identificación de elementos como en la evaluación del modelo.
- **Capítulo 9. Conclusiones y trabajo futuro:** Presenta las principales conclusiones desde la perspectiva del proyecto y personal, así como posibles líneas de trabajo a futuro.

Finalmente, la cuarta parte incluye los **Apéndices** (con el manual de instalación y el manual de usuario) y la **Bibliografía**.

Capítulo 2

Gestión del proyecto

En este capítulo se describe la metodología Asap (Agile Student Academic Projects), que ha sido empleada para desarrollar el proyecto. Además, se detalla cómo se ha llevado a cabo una planificación incremental utilizando dicha metodología, y se presentan tanto los presupuestos como el balance temporal del proyecto.

La sección 2.1 está dedicada a explicar la metodología aplicada en el desarrollo del proyecto. En la sección 2.2 se expone la planificación realizada antes de su inicio, en la sección 2.3 se detallan los presupuestos contemplados, y finalmente, la sección 2.4 recoge el balance tanto temporal como económico.

2.1. Metodología

Para la realización de este proyecto hemos decidido utilizar la metodología Asap (Agile Student Academic Projects) [37], ya que es una metodolgía diseñada para facilitar que los estudiantes realicen sus Trabajos Fin de Grado (TFG) de manera efectiva y profesional. Asap adapta prácticas ágiles del ámbito profesional (como las utilizadas en frameworks como Scrum [56] o eXtreme Programming [7]) para proporcionar al estudiante una estructura de trabajo clara y un feedback constante que le ayude a alcanzar los objetivos de su proyecto. Los motivos principales por los que hemos elegido esta metodología son, en primer lugar, que las metodologías ágiles son muy utilizadas actualmente en el ámbito laboral. El segundo motivo, es, de hecho, la razón de que se usen tanto estás metodologías en el ámbito laboral y es que permiten trabajar en mejora constante para obtener resultados de muy alta calidad. Asap tiene tres componentes clave inspirados en los componentes de Scrum [56] que se detallarán en las siguientes subsecciones que son los roles, los eventos y los artefactos.

2.1.1. Roles

A continuación vamos a explicar los roles que participarán en el proyecto y el papel que jugarán en él.

- **Estudiante:** Es el eje central del proyecto y tiene un rol activo en todas las fases del mismo. Su responsabilidad principal es identificar y ejecutar las tareas necesarias para alcanzar los objetivos establecidos. Además, debe mantener una comunicación fluida y efectiva con los demás participantes, en especial con su tutor y la comunidad académica.

- **Tutor:** Desempeña un papel clave en la supervisión y orientación del estudiante a lo largo del proceso. Su función es proporcionar un entorno que favorezca el desarrollo académico del estudiante, fomentando la comunicación abierta y ofreciendo retroalimentación periódica que facilite la consecución de los objetivos del proyecto.
- **Comunidad:** Representa un apoyo fundamental en el desarrollo del TFG. Está conformada por compañeros que se encuentran en situaciones similares a las del estudiante, así como por profesores y expertos en la materia. Su propósito es brindar orientación y compartir conocimientos en aquellas áreas donde el estudiante lo requiera.
- **Tribunal:** Es el encargado de la evaluación final del TFG. Su función es valorar si los resultados obtenidos cumplen con los objetivos de aprendizaje definidos en la titulación.

2.1.2. Eventos

El desarrollo del proyecto se organizara en *sprint* de aproximadamente un mes de duración. Cada uno de ellos tendrá un objetivo específico, propuesto por el profesor y consensuando con el estudiante según el progreso alcanzado hasta el momento, las tareas pendientes y las circunstancias particulares del estudiante. De esta forma, se facilita la planificación, el seguimiento del trabajo y la evaluación continua del proyecto.

Cada *sprint* incluye cuatro eventos clave que se detallarán a continuación:

- **Planificación:** Al inicio de cada *sprint,* se establece el objetivo del mismo y se identifican las tareas necesarias para alcanzarlo. Este evento permite al estudiante estructurar su trabajo y definir un plan de acción que le permitirá organizarse en las semanas siguientes. Toda la planificación se lleva a cabo en el cuaderno de trabajo.
- **Sincronización:** Se realizan reuniones de sincronización semanales para dar seguimiento al progreso del estudiante. Serán reuniones cortas, de aproximadamente 15 minutos, en las que se discuten los avances, se identifican y resuelven posibles bloqueos y, si es necesario, se ajusta la planificación.
- **Comunicación de progresos:** Al finalizar cada *sprint,* el estudiante presenta el trabajo realizado hasta el momento. Este evento no solo permite recibir retroalimentación valiosa, sino que también prepara al estudiante para la defensa final del TFG. Además, el estudiante participa como miembro de la comunidad, proporcionando comentarios y sugerencias a otros compañeros.
- **Retrospectiva:** Marca el cierre del *sprint* y sirve para reflexionar sobre el trabajo realizado. Se analizan los aspectos positivos y las áreas de mejora, con el objetivo de optimizar la metodología de trabajo en los siguientes *sprints.* La experiencia de cada participante contribuye al aprendizaje global de la comunidad.

2.1.3. Artefactos

Durante el desarrollo del Trabajo de Fin de Grado (TFG), se emplean dos artefactos fundamentales para documentar y evaluar el progreso del proyecto. Estos artefactos permiten visualizar los avances realizados y facilitar la mejora continua a lo largo del proceso.

- **Incremento:** Representa el resultado del trabajo al final de cada *sprint* y refleja el progreso alcanzado en el proyecto. El incremento debe incluir al menos el producto del TFG en el estado en el que se encuentre en ese momento, la memoria del proyecto, la presentación utilizada en la comunicación de progresos y el cuaderno de trabajo actualizado.
- **Cuaderno de trabajo:** es el elemento principal para la planificación y el seguimiento del trabajo. En él registrarás las tareas a realizar en cada *sprint*, determinando sus plazos de ejecución a priori y anotando las fechas reales de finalización de cada una de ellas. Asimismo, esta herramienta permitirá llevar registro de las tareas realizadas y el tiempo invertido en ellas, ayudándo al alumno a gestionar mejor tu esfuerzo y a mantener un registro detallado del progreso.
- **Retroalimentación:** Recoge todos los comentarios y sugerencias recibidos al final de cada *sprint,* tanto del tutor como de los miembros de la comunidad. La retroalimentación es un elemento clave para mejorar tanto el producto como el proceso de desarrollo del TFG, por lo que debe ser integrada de manera efectiva en los *sprint* siguientes.

2.1.4. Entorno de trabajo

El entorno de trabajo es fundamental para el desarrollo del TFG. Por ello, utilizaremos diversas herramientas para facilitar la colaboración, la organización y el seguimiento del proyecto.

- **Espacio de trabajo compartido:** ofrece los recursos necesarios para compartir todos los contenidos generados en el TFG y asegurar la comunicación entre todos los participantes en el proyecto. Esto se ha implementado en *Teams* mediante un *chat* con el alumno y sus tutores y otro con toda la comunidad. De esta forma, se iban planteando las dudas y respondiendo de forma rápida y los archivos estaban accesibles en todo momento.

Este entorno colaborativo proporciona todo lo necesario para gestionar el TFG de manera efectiva, asegurando que el alumno siempre tenga acceso a los recursos y al apoyo necesario para alcanzar sus objetivos.

2.2. Planificación temporal

Asap divide el proyecto en *sprints* de duración comparable. Concretamente, este TFG se ha organizado 4 *sprints* con las siguientes fechas de inicio y finalización:

- *Sprint* #1: 24/02/2025 - 21/03/2025
- *Sprint* #2: 21/03/2025 - 25/04/2025
- *Sprint* #3: 25/04/2025 - 23/05/2025
- *Sprint* #4: 23/05/2025 - 20/06/2025

Al comienzo de cada *sprint*, se llevará a cabo una planificación en la que se definen las fechas previstas de inicio y finalización de cada actividad a desarrollar en dicho *sprint*. Las actividades asignadas para cada *sprint* son determinadas previamente en la reunión de inicio y se presentarán en los apartados siguientes.

A continuación, como se indicaba anteriormente, en la planificación se definen las actividades completadas en cada *sprint,* junto con un diagrama de Gantt por *sprint,* que permitirá visualizar la distribución de tareas planificadas antes de comenzar cada *sprint.*

2.2.1. *Sprint* #1

El proyecto está dividido en 5 paquetes de trabajo y cada uno de ellos comprende la realización de diferentes actividades. A continución, se detallan tanto los paquetes de trabajo activos en este *sprint* como las actividades planificadas para alcanzar su objetivo.

- **PT1-Proyecto**

  Este paquete de trabajo aborda la gestión de proyectos en el ámbito de la Ingeniería Informática, comprende 7 actividades que se explican a continuación:

  - **H1.1-Planteamiento del problema:** Construir el *problem statement* del proyecto para determinará el "gap" existente entre la situación en la que se inicia el proyecto y el resultado esperado a su finalización..
  - **H1.2-Objetivos:** Determinar y especificar los objetivos del proyecto que deben guiar todas las actividades del proyecto y proporcionar criterios claros para evaluar su éxito.
  - **H1.3-Condicionantes:** Identificar todos aquellos aspectos que pueden condicionar el desarrollo del proyecto.
  - **H1.4-Metodología:** Justificar la capacidad de la metodología Asap para ayudar a desarrollar este proyecto de forma estructurada y sistemática.
  - **H1.5-Planificación:** Establecer un plan de trabajo, que incluya todas las tareas necesarias para completar el proyecto. La lanificación debe considerar los plazos, los recursos necesarios y las posibles contingencias.
  - **H1.6-Balance:** Esta actividad se realiza al final de cada *sprint* con el objetivo de evaluar el uso de los recursos utilizados respecto a los resultados obtenidos, analizando las desviaciones que se han producido durante el desarrollo del proyecto respecto a lo planificado.
  - **H.7-Interacción:** Esta actividad se lleva a cabo durante todo el *sprint* y busca mantener una interacción efectiva y continua con todos los participantes en tu proyecto.

- **PT2-Antecedentes**

  Este paquete de trabajo aborda la recopilación y análisis de información relevante para establecer el contexto en el que se enmarca el proyecto. Las actividades que comprende son las siguientes:

  - **H2.1-Entorno de negocio:** Comprender el entorno de negocio del proyecto e identificar los *stakeholders.*
  - **H2.2-Estado del arte:** Adquierir una visión sólida de cómo otras soluciones existentes abordan problemas comparables al planteado en tu proyecto.
  - **H2.3-Fundamentos teóricos:** Obtener un conocimiento profundo de las teorías, principios y conceptos que son relevantes para el proyecto.

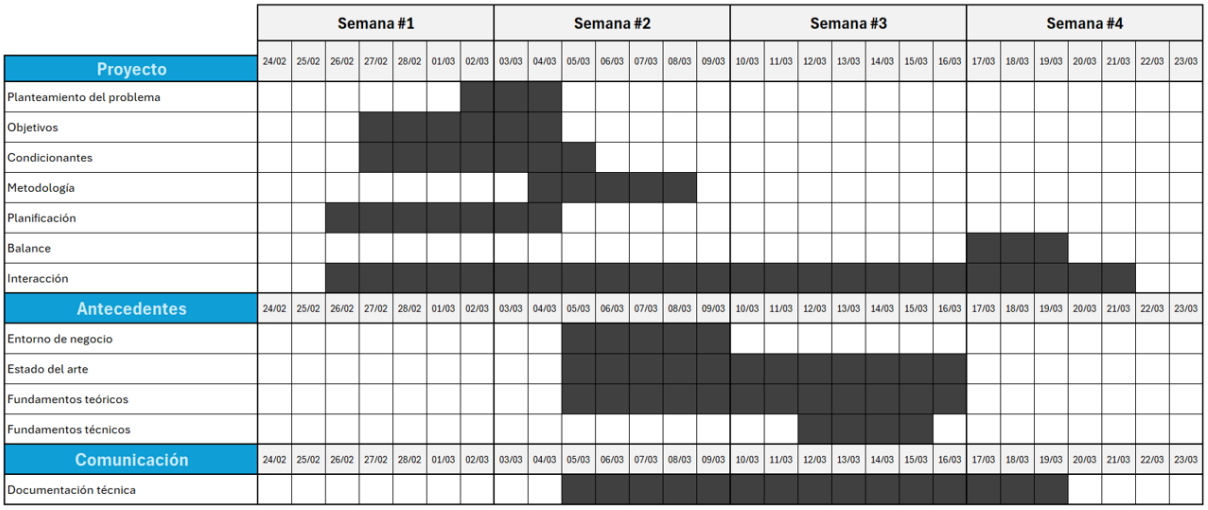

Figura 2.1: Diagrama de Gantt del primer *sprint* .

- **H2.4-Fundamentos técnicos:** Obtener una visión profunda de las técnicas, herramientas y/o tecnologías utilizadas habitualmente para construir productos similares en proyectos comparables.

- **PT5-Comunicación**

  Este objetivo aborda la comunicación efectiva del trabajo que se hs realizado en el proyecto y los resultados que se han obtenido gracias a ello.

  - **H5.1-Documentación técnica:** Redactar de la memoria técnica del proyecto, en la que se presentará el trabajo realizado de manera clara, estructurada y completa, de acuerdo con los criterios de aceptación establecidos en cada uno de los objetivos del proyecto.
  - **H5.2-Presentación:** Esta actividad de aprendizaje no se abordará durante este primer *sprint* .

En la Figura 2.1 se muestra el diagrama de Gantt del primer *sprint* .

2.2.2. *Sprint* #2

A continución, se detallan tanto los paquetes de trabajo activos en este *sprint* como las actividades planificadas para alcanzar su objetivo.

- **PT1-Proyecto**

  Este paquete de trabajo aborda la gestión de proyectos en el ámbito de la Ingeniería Informática, comprende 7 actividades que se explican a continuación:

  - **H1.1-Planteamiento del problema:** Construir el *problem statement* del proyecto para determinará el "gap" existente entre la situación en la que se inicia el proyecto y el resultado esperado a su finalización..

- **H1.2-Objetivos:** Determinar y especificar los objetivos del proyecto que deben guiar todas las actividades del proyecto y proporcionar criterios claros para evaluar su éxito.
- **H1.3-Condicionantes:** Identificar todos aquellos aspectos que pueden condicionar el desarrollo del proyecto.
- **H1.4-Metodología:** Justificar la capacidad de la metodología Asap para ayudar a desarrollar este proyecto de forma estructurada y sistemática.
- **H1.5-Planificación:** Establecer un plan de trabajo, que incluya todas las tareas necesarias para completar el proyecto. La lanificación debe considerar los plazos, los recursos necesarios y las posibles contingencias.
- **H1.6-Balance:** Esta actividad se realiza al final de cada *sprint* con el objetivo de evaluar el uso de los recursos utilizados respecto a los resultados obtenidos, analizando las desviaciones que se han producido durante el desarrollo del proyecto respecto a lo planificado.
- **H.7-Interacción:** Esta actividad se lleva a cabo durante todo el *sprint* y busca mantener una interacción efectiva y continua con todos los participantes en tu proyecto.

- **PT2-Antecedentes**

  Este paquete de trabajo aborda la recopilación y análisis de información relevante para establecer el contexto en el que se enmarca el proyecto. Las historias que comprende son las siguientes:

  - **H2.1-Entorno de negocio:** Comprender el entorno de negocio del proyecto e identificar los *stakeholders*.
  - **H2.2-Estado del arte:** Adquierir una visión sólida de cómo otras soluciones existentes abordan problemas comparables al planteado en tu proyecto.
  - **H2.3-Fundamentos teóricos:** Obtener un conocimiento profundo de las teorías, principios y conceptos que son relevantes para el proyecto.
  - **H2.4-Fundamentos técnicos:** Obtener una visión profunda de las técnicas, herramientas y/o tecnologías utilizadas habitualmente para construir productos similares en proyectos comparables.

- **PT3-Desarrollo**

  Este paquete de trabajo aborda las competencias relacionadas con el proceso de desarrollo del proyecto, desde la comprensión y análisis del problema hasta la implementación y prueba del producto final. Las historias que comprende son las siguientes aunque la última de ellas no se abordará en este *sprint*:

  - **H3.1-Análisis del problema:** Comprender el problema que se aborda en el proyecto y especificarlos de acuerdo con los requisitos que lo caracterizan.
  - **H3.2-Diseño de la solución:** Diseñar una solución que satisfaga los objetivos del proyecto, de acuerdo con los requisitos y condicionantes identificados.
  - **H3.3-Construcción del producto:** Construir el producto que resuelve el problema planteado en el proyecto, de acuerdo con sus objetivos, condicionantes y requisitos.

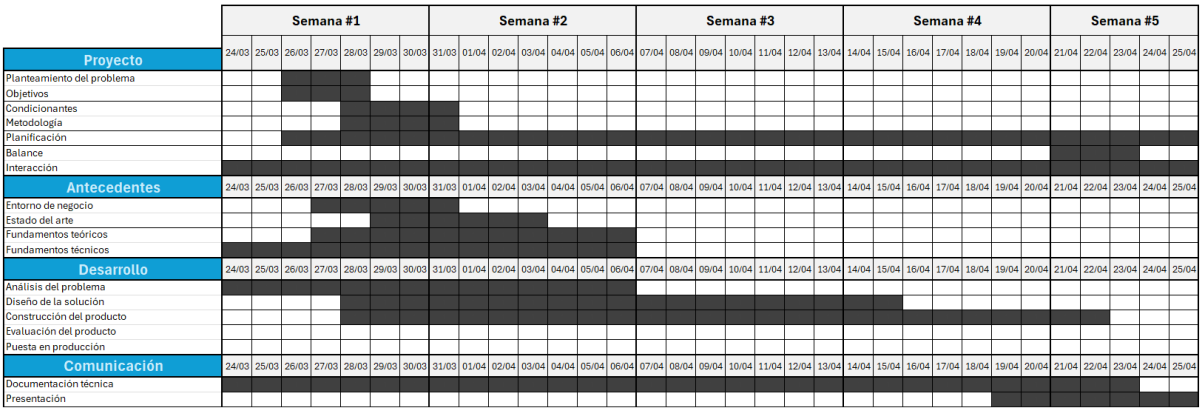

Figura 2.2: Diagrama de Gantt del segundo *sprint* .

- **H3.4-Evaluación del producto:** Evaluar que el producto desarrollado cumple con los requisitos especificados y funciona correctamente en las condiciones previstas.
- **H3.5-Puesta en producción:** Determinar el proceso de puesta en producción del producto construido..

- **PT5-Comunicación**

  Este paquete de trabajo aborda la comunicación efectiva del trabajo que se ha realizado en el proyecto y los resultados que se han obtenido gracias a ello.

  - **H5.1-Documentación técnica:** Redactar de la memoria técnica del proyecto, en la que se presentará el trabajo realizado de manera clara, estructurada y completa, de acuerdo con los criterios de aceptación establecidos en cada uno de los objetivos del proyecto.
  - **H5.2-Presentación:** se centra en tu capacidad para presentar y defender tu proyecto de manera efectiva, demostrando con ello tu comprensión del trabajo realizado, tu capacidad para comunicar ideas de manera clara y tu habilidad para responder a preguntas sobre el proyecto.

En la Figura 2.2 se muestra el diagrama de Gantt del segundo *sprint* .

2.2.3. *Sprint* #3

A continución, se detallan tanto los paquetes de trabajo activos en este *sprint* como las actividades planificadas para alcanzar su objetivo.

- **PT1-Proyecto**

  Este paquete de trabajo aborda la gestión de proyectos en el ámbito de la Ingeniería Informática, comprende 7 actividades que se explican a continuación:

  - **H1.1-Planteamiento del problema:** Construir el *problem statement* del proyecto para determinará el "gap" existente entre la situación en la que se inicia el proyecto y el resultado esperado a su finalización..

- **H1.2-Objetivos:** Determinar y especificar los objetivos del proyecto que deben guiar todas las actividades del proyecto y proporcionar criterios claros para evaluar su éxito.
- **H1.3-Condicionantes:** Identificar todos aquellos aspectos que pueden condicionar el desarrollo del proyecto.
- **H1.4-Metodología:** Justificar la capacidad de la metodología Asap para ayudar a desarrollar este proyecto de forma estructurada y sistemática.
- **H1.5-Planificación:** Establecer un plan de trabajo, que incluya todas las tareas necesarias para completar el proyecto. La lanificación debe considerar los plazos, los recursos necesarios y las posibles contingencias.
- **H1.6-Balance:** Esta actividad se realiza al final de cada *sprint* con el objetivo de evaluar el uso de los recursos utilizados respecto a los resultados obtenidos, analizando las desviaciones que se han producido durante el desarrollo del proyecto respecto a lo planificado.
- **H.7-Interacción:** Esta actividad se lleva a cabo durante todo el *sprint* y busca mantener una interacción efectiva y continua con todos los participantes en tu proyecto.

- **PT2-Antecedentes**

  Este paquete de trabajo aborda la recopilación y análisis de información relevante para establecer el contexto en el que se enmarca el proyecto. Las historias que comprende son las siguientes:

  - **H2.1-Entorno de negocio:** Comprender el entorno de negocio del proyecto e identificar los *stakeholders*.
  - **H2.2-Estado del arte:** Adquierir una visión sólida de cómo otras soluciones existentes abordan problemas comparables al planteado en tu proyecto.
  - **H2.3-Fundamentos teóricos:** Obtener un conocimiento profundo de las teorías, principios y conceptos que son relevantes para el proyecto.
  - **H2.4-Fundamentos técnicos:** Obtener una visión profunda de las técnicas, herramientas y/o tecnologías utilizadas habitualmente para construir productos similares en proyectos comparables.

- **PT3-Desarrollo**

  Este paquete de trabajo aborda las competencias relacionadas con el proceso de desarrollo del proyecto, desde la comprensión y análisis del problema hasta la implementación y prueba del producto final. Las historias que comprende son las siguientes aunque la última de ellas no se abordará en este *sprint*:

  - **H3.1-Análisis del problema:** Comprender el problema que se aborda en el proyecto y especificarlos de acuerdo con los requisitos que lo caracterizan.
  - **H3.2-Diseño de la solución:** Diseñar una solución que satisfaga los objetivos del proyecto, de acuerdo con los requisitos y condicionantes identificados.
  - **H3.3-Construcción del producto:** Construir el producto que resuelve el problema planteado en el proyecto, de acuerdo con sus objetivos, condicionantes y requisitos.

- **H3.4-Evaluación del producto:** Evaluar que el producto desarrollado cumple con los requisitos especificados y funciona correctamente en las condiciones previstas.
- **H3.5-Puesta en producción:** Determinar el proceso de puesta en producción del producto construido..

- **PT4-Análisis**

  Este paquete de trabajo aborda las competencias relacionadas con el proceso de desarrollo del proyecto, desde la comprensión y análisis del problema hasta la implementación y prueba del producto final. Las historias que comprende son las siguientes aunque la última de ellas no se abordará en este *sprint*:

  - **H3.1-Análisis del problema:** Comprender el problema que se aborda en el proyecto y especificarlos de acuerdo con los requisitos que lo caracterizan.
  - **H3.2-Diseño de la solución:** Diseñar una solución que satisfaga los objetivos del proyecto, de acuerdo con los requisitos y condicionantes identificados.
  - **H3.3-Construcción del producto:** Construir el producto que resuelve el problema planteado en el proyecto, de acuerdo con sus objetivos, condicionantes y requisitos.
  - **H3.4-Evaluación del producto:** Evaluar que el producto desarrollado cumple con los requisitos especificados y funciona correctamente en las condiciones previstas.
  - **H3.5-Puesta en producción:** Determinar el proceso de puesta en producción del producto construido..

- **PT5-Comunicación**

  Este paquete de trabajo aborda la comunicación efectiva del trabajo que se ha realizado en el proyecto y los resultados que se han obtenido gracias a ello.

  - **H5.1-Documentación técnica:** Redactar de la memoria técnica del proyecto, en la que se presentará el trabajo realizado de manera clara, estructurada y completa, de acuerdo con los criterios de aceptación establecidos en cada uno de los objetivos del proyecto.
  - **H5.2-Presentación:** se centra en tu capacidad para presentar y defender tu proyecto de manera efectiva, demostrando con ello tu comprensión del trabajo realizado, tu capacidad para comunicar ideas de manera clara y tu habilidad para responder a preguntas sobre el proyecto.

- **PT6-Aceptación**

  Este paquete de trabajo aborda el diseño y ejecución de un plan que permita evaluar el cumplimiento de los objetivos establecidos del proyecto.

  - **H6.1-Métricas de éxito:** Establecer las métricas necesarias para evaluar el éxito del proyecto, asegurando su alineamiento con los objetivos establecidos y su capacidad para evaluarlos de forma precisa y objetiva.
  - **H6.2-Plan de aceptación:** Diseñar y ejecutar un plan de aceptación estructurado y riguroso, que asegure una evaluación precisa del proyecto.

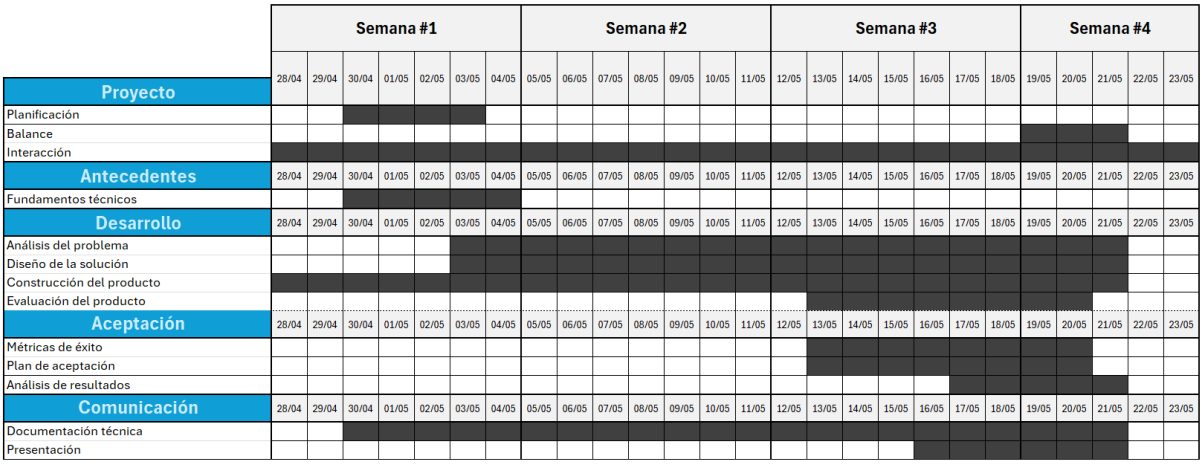

Figura 2.3: Diagrama de Gantt del tercer *sprint* .

- **H6.3-Análisis de resultados:** Interpretar los resultados obtenidos en el proceso de aceptación y, en base a ellos, determinar el éxito del proyecto.

En la Figura 2.3 se muestra el diagrama de Gantt del tercer *sprint* .

2.2.4. *Sprint* #4

En este último *sprint* se abordarán todos los paquetes de trabajo y actividades contemplados.

- **PT1-Proyecto**

  Este paquete de trabajo aborda la gestión de proyectos en el ámbito de la Ingeniería Informática, comprende 7 actividades que se explican a continuación:

  - **H1.1-Planteamiento del problema:** Construir el *problem statement* del proyecto para determinará el "gap" existente entre la situación en la que se inicia el proyecto y el resultado esperado a su finalización..
  - **H1.2-Objetivos:** Determinar y especificar los objetivos del proyecto que deben guiar todas las actividades del proyecto y proporcionar criterios claros para evaluar su éxito.
  - **H1.3-Condicionantes:** Identificar todos aquellos aspectos que pueden condicionar el desarrollo del proyecto.
  - **H1.4-Metodología:** Justificar la capacidad de la metodología Asap para ayudar a desarrollar este proyecto de forma estructurada y sistemática.
  - **H1.5-Planificación:** Establecer un plan de trabajo, que incluya todas las tareas necesarias para completar el proyecto. La lanificación debe considerar los plazos, los recursos necesarios y las posibles contingencias.
  - **H1.6-Balance:** Esta actividad se realiza al final de cada *sprint* con el objetivo de evaluar el uso de los recursos utilizados respecto a los resultados obtenidos, analizando las desviaciones que se han producido durante el desarrollo del proyecto respecto a lo planificado.

- **H.7-Interacción:** Esta actividad se lleva a cabo durante todo el *sprint* y busca mantener una interacción efectiva y continua con todos los participantes en tu proyecto.

- **PT2-Antecedentes**

  Este paquete de trabajo aborda la recopilación y análisis de información relevante para establecer el contexto en el que se enmarca el proyecto. Las historias que comprende son las siguientes:

  - **H2.1-Entorno de negocio:** Comprender el entorno de negocio del proyecto e identificar los *stakeholders*.
  - **H2.2-Estado del arte:** Adquirir una visión sólida de cómo otras soluciones existentes abordan problemas comparables al planteado en tu proyecto.
  - **H2.3-Fundamentos teóricos:** Obtener un conocimiento profundo de las teorías, principios y conceptos que son relevantes para el proyecto.
  - **H2.4-Fundamentos técnicos:** Obtener una visión profunda de las técnicas, herramientas y/o tecnologías utilizadas habitualmente para construir productos similares en proyectos comparables.

- **PT3-Desarrollo**

  Este paquete de trabajo aborda las competencias relacionadas con el proceso de desarrollo del proyecto, desde la comprensión y análisis del problema hasta la implementación y prueba del producto final. Las historias que comprende son las siguientes aunque la última de ellas no se abordará en este *sprint*:

  - **H3.1-Análisis del problema:** Comprender el problema que se aborda en el proyecto y especificarlos de acuerdo con los requisitos que lo caracterizan.
  - **H3.2-Diseño de la solución:** Diseñar una solución que satisfaga los objetivos del proyecto, de acuerdo con los requisitos y condicionantes identificados.
  - **H3.3-Construcción del producto:** Construir el producto que resuelve el problema planteado en el proyecto, de acuerdo con sus objetivos, condicionantes y requisitos.
  - **H3.4-Evaluación del producto:** Evaluar que el producto desarrollado cumple con los requisitos especificados y funciona correctamente en las condiciones previstas.
  - **H3.5-Puesta en producción:** Determinar el proceso de puesta en producción del producto construido..

- **PT4-Análisis**

  Este paquete de trabajo aborda las competencias relacionadas con el proceso de desarrollo del proyecto, desde la comprensión y análisis del problema hasta la implementación y prueba del producto final. Las historias que comprende son las siguientes aunque la última de ellas no se abordará en este *sprint*:

  - **H3.1-Análisis del problema:** Comprender el problema que se aborda en el proyecto y especificarlos de acuerdo con los requisitos que lo caracterizan.
  - **H3.2-Diseño de la solución:** Diseñar una solución que satisfaga los objetivos del proyecto, de acuerdo con los requisitos y condicionantes identificados.

- **H3.3-Construcción del producto:** Construir el producto que resuelve el problema planteado en el proyecto, de acuerdo con sus objetivos, condicionantes y requisitos.
- **H3.4-Evaluación del producto:** Evaluar que el producto desarrollado cumple con los requisitos especificados y funciona correctamente en las condiciones previstas.
- **H3.5-Puesta en producción:** Determinar el proceso de puesta en producción del producto construido..

- **PT5-Comunicación**

  Este paquete de trabajo aborda la comunicación efectiva del trabajo que se ha realizado en el proyecto y los resultados que se han obtenido gracias a ello.

  - **H5.1-Documentación técnica:** Redactar de la memoria técnica del proyecto, en la que se presentará el trabajo realizado de manera clara, estructurada y completa, de acuerdo con los criterios de aceptación establecidos en cada uno de los objetivos del proyecto.
  - **H5.2-Presentación:** se centra en tu capacidad para presentar y defender tu proyecto de manera efectiva, demostrando con ello tu comprensión del trabajo realizado, tu capacidad para comunicar ideas de manera clara y tu habilidad para responder a preguntas sobre el proyecto.

- **PT6-Aceptación**

  Este paquete de trabajo aborda el diseño y ejecución de un plan que permita evaluar el cumplimiento de los objetivos establecidos del proyecto.

  - **H6.1-Métricas de éxito:** Establecer las métricas necesarias para evaluar el éxito del proyecto, asegurando su alineamiento con los objetivos establecidos y su capacidad para evaluarlos de forma precisa y objetiva.
  - **H6.2-Plan de aceptación:** Diseñar y ejecutar un plan de aceptación estructurado y riguroso, que asegure una evaluación precisa del proyecto.
  - **H6.3-Análisis de resultados:** Interpretar los resultados obtenidos en el proceso de aceptación y, en base a ellos, determinar el éxito del proyecto.

- **PT7-Evaluación**

  El producto de aprendizaje no sólo especifica el trabajo a realizar en el TFG del alumno, sino que también determina la base sobre la que se plantean los diferentes instrumentos de evaluación contemplados en ASAP, tanto los que usa el estudiante, como los del tutor.

  - **H7.1-Estudiante:** Al finalizar el trabajo relacionado con cada actividad el estudiante tenndrá que valorar cualitativamente la calidad del resultado producido.
  - **H7.2-Profesor:** Evaluación por parte del tutor al alumno.

En la Figura 2.4 se muestra el diagrama de Gantt del cuarto *sprint* .

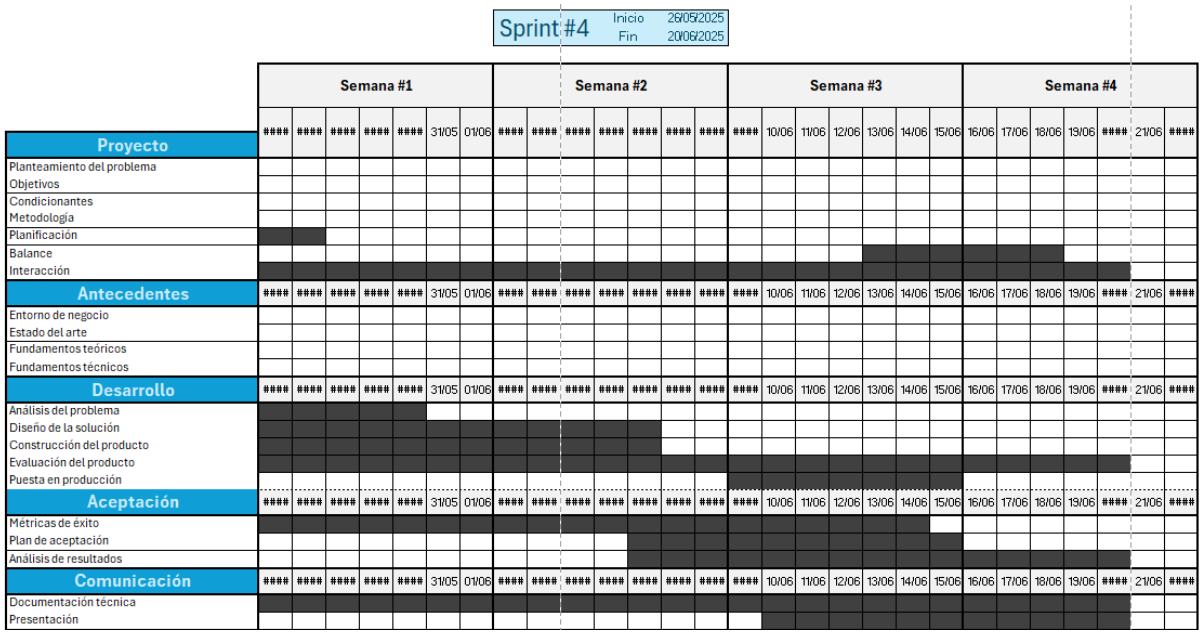

Figura 2.4: Diagrama de Gantt del cuarto *sprint* .

2.3. Presupuestos

En esta sección se desglosa el presupuesto necesario para llevar a cabo el proyecto, dividiendo los costes derivados del *hardware*, del *software* y de los recursos humanos. Se mostrarán los cálculos individuales, seguidos del coste total estimado del proyecto.

2.3.1. *Hardware*

Los recursos de *hardware* necesarios han sido principalmente un ordenador portátil, acceso a una red Wi-Fi estable y una máquina virtual facilitada por la Universidad de Valladolid (UVa), utilizada para desplegar la aplicación.

Aunque en este caso la máquina virtual ha sido proporcionada por la UVa, vamos a suponer para el presupuesto que esto no es así para hacer una estimación más precisa:

Herramienta	**Coste total (€)**	**Vida útil (meses)**	**Porcentaje uso (%)**	**Uso en meses**
Ordenador portátil	1099	$4,5 \times 12 = 54$	65	5
Wi-Fi	20 (mensuales)	-	15	5
Máquina virtual despliegue	25 (mensuales)	-	100	1

Tabla 2.1: Presupuesto estimado de *hardware*.

Se empleará la fórmula siguiente para estimar el coste real del *hardware*, tomando la vida útil

como 1 en el caso de no tener.

$$\text{Coste (euros)} = \frac{\text{Coste total (euros)}}{\text{Vida útil (meses)}} \times \text{Porcentaje uso} \times \text{Tiempo uso (meses)} \tag{2.1}$$

Finalmente, en euros, obtenemos los valores de la Tabla 2.7.

Herramienta	**Coste real (€)**
Ordenador portátil	66,14 €
Wi-Fi	15 €
Máquina virtual despliegue	25 €
Total	106,14 €

Tabla 2.2: Estimación de costes reales del *hardware*.

2.3.2. *Software*

La mayoría de herramientas *software* que se estima que se utilizarán en este proyecto son gratuitas a excepción de la API de Openai. En la Tabla 2.3 se presentan las herramientas empleadas.

Herramienta	**Coste mensual**	**Porcentaje uso**	**Uso en meses**
API OpenAI	10	100	4
Teams	0	100	5
Overleaf	0	70	5
Paquete Office	0	60	5
Visual Studio Code	0	100	4
Github	0	100	4
Drawio	0	100	4

Tabla 2.3: Presupuesto estimado de *software*.

Usando la fórmula de (2.1) para la API de Openai, pues es la única herramienta *software* de pago tenemos que

$$\text{Coste } \textit{software} = \text{Coste real API de Openai} = 40€. \tag{2.2}$$

2.3.3. Recursos humanos

Este apartado cuantifica el coste humano considerando los diferentes roles desempeñados por el estudiante durante las 330 horas de duración estimadas del proyecto(media aritmética de 300 horas y 360 horas que son las horas que la UVa estima que el alumno debe tardar en realizar el TFG).A continuación se detallan los roles que participan en el proyecto, aunque en la práctica el alumno realizará todos ellos.

- **Analista de sistemas**: se ocupa de estudiar y definir los requisitos del proyecto.
- **Arquitecto de software**: se encarga de estructurar y planificar la solución tecnológica del proyecto.
- **Desarrollador de frontend**: desarrolla la interfaz y la experiencia de usuario de la aplicación.
- **Desarrollador de backend**: implementa la lógica y los procesos internos de la aplicación utilizando Python.
- **Tester de software**: verifica que el software funcione correctamente y cumpla con los requisitos.
- **Documentalista**: elabora la documentación técnica y descriptiva del proyecto.

La Tabla 2.8 detalla el salario medio anual de cada rol y su porcentaje de participación en el proyecto.

Rol	Salario anual(€)	Participación en el proyecto (%)
Analista de sistemas	33.000	10 %
Arquitecto de *software*	45.000	8 %
Desarrollador de *frontend*	31.969	15 %
Desarrollador de *backend*	34.497	42 %
Tester software	41.200	10 %
Documentalista	23.585	15 %

Tabla 2.4: Presupuesto de *recursos humanos*.

Suponemos que una persona trabaja aproximadamente 1.824 horas anuales, con este dato y la información de la Tabla 2.8 calcularemos el presupuesto para cada uno de los roles, utilizando la fórmula siguiente:

$$\text{Salario rol} = \frac{\text{Salario anual}}{\text{Horas al año}} \times \text{Horas totales proyecto} \times \text{Porcentaje participación} \tag{2.3}$$

Para determinar el coste total que representa cada rol para la empresa, es necesario añadir al salario la cotización que la empresa abona a la seguridad social por cada trabajador, la cual hemos estimado en un 32,5 %. De esta manera, el coste final se calcula con la fórmula siguiente:

$$\text{Coste rol} = \text{Salario rol} + 32,5\,\% \times \text{Salario rol} = 1,325 \times \text{Salario rol} \tag{2.4}$$

Rol	**Salario rol (€)**	**Coste rol (€)**
Analista de sistemas	597,4	791,56
Arquitecto de *software*	651,32	863
Desarrollador de *frontend*	867,58	1.149,54
Desarrollador de *backend*	2.621,32	3.473,25
Tester software	745,4	987,66
Documentalista	640,05	848,07

Tabla 2.5: Salario y coste empresarial por rol en el proyecto.

Por tanto, si sumamos todos los costes por rol llegamos a que el presupuesto estimado para los recursos humanos es de 8.113,08 €.

2.3.4. Presupuesto total

Finalmente, sumando los costes *software*, *hardware* y de recursos humanos calculados en las subsecciones anteriores, llegamos a que el presupuesto total estimado es:

$$\text{Presupuesto total} = 106,14 + 40 + 8{,}113,08 = \mathbf{8.259{,}22\,€} \tag{2.5}$$

2.4. Balance temporal y económico

En esta sección se expondrán el balance temporal y el balance económico cuya función es comparar respectivamente el tiempo y el coste planificados, con el tiempo y el coste que realmente han sido necesarios para llevarlo a cabo.

2.4.1. Balance temporal

El objetivo de esta sección es comparar la planificación que se realizó al inicio de cada *sprint* con la distribución que se ha realizado en la práctica. Para ello nos apoyaremos de un cronograma que contrastará la planificación de cada *sprint* respecto al trabajo realizado. A este respecto, se colorearán en verde las celdas correspondientes al trabajo realizado en las fechas planificadas y en naranja aquellas tareas que se hayan hecho en fechas diferentes a las inicialmente estimadas. A continuación se muestran las horas de trabajo dedicadas a cada *sprint*, además de las horas dedicadas en la última semana (después del resto de *sprints*):

- *Sprint 1*: 63 horas
- *Sprint 2*: 104,42 horas
- *Sprint 3*: 93,33 horas
- *Sprint 4*: 98,75 horas
- *Semana extra*: 29 horas

Esto hace un total de 388,5 horas, lo cual supera el límite superior considerado para el proyecto completo.

Balance temporal del *sprint* 1

Durante este primer *sprint* se ha seguido la planificación de forma bastante correcta para ser la primera vez que la alumna planificaba un *sprint* de este estilo; ha habido algunos desvíos y tareas realizadas fuera de tiempo, como vemos en el cronograma de la Figura 2.5 pero creo que tras el aprendizaje de este *sprint* la planificación del siguiente será mucho más precisa. Estas desviaciones se deben principalmente a la inexperiencia de la alumna realizando planificaciones de este estilo y a su gran carga de trabajo. En total, en este *sprint* se han invertido 63 horas de trabajo.

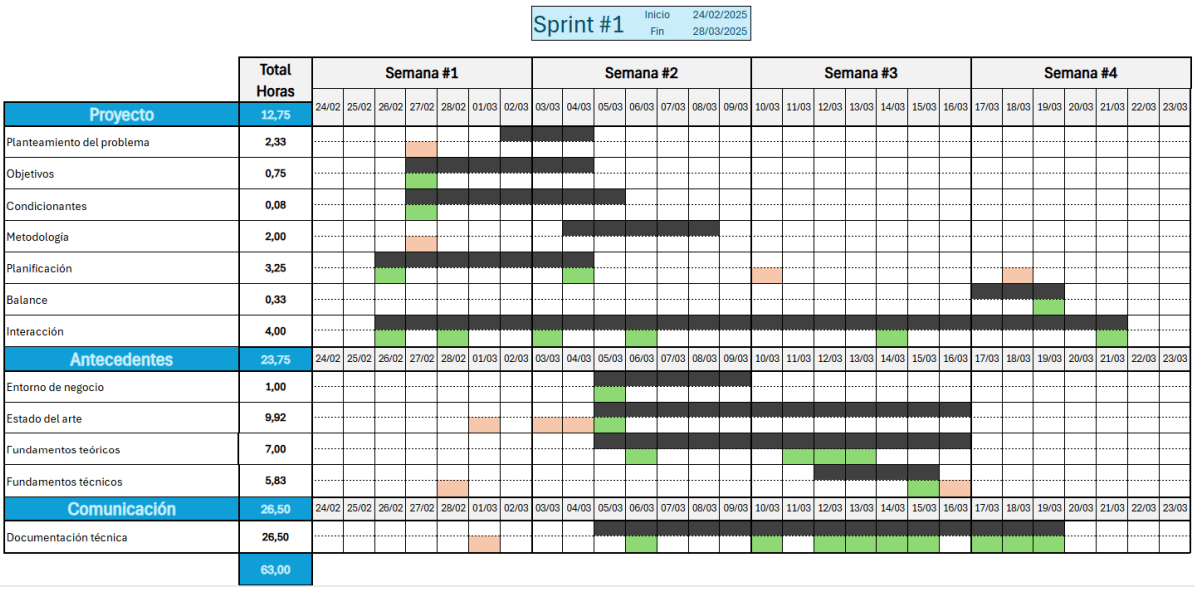

Figura 2.5: Cronograma del *sprint* 1.

Balance temporal del *sprint* 2

Durante este segundo *sprint* se ha seguido la planificación de forma más fiel a la anterior y apenas se han realizado tareas fuera del plazo planificado como vemos en el cronograma de la Figura 2.6. Este ha sido un *sprint* con una mayor carga de trabajo, en total se han dedicado 104, 42 horas, y vemos en el cronograma que hubo un parón en el trabajo entre el 11 y el 19 de abril que corresponde con las vacaciones de Semana Santa.

Balance temporal del *sprint* 3

Durante este tercer *sprint* se han acumulado un total de 93,33 horas de trabajo. Como se aprecia en el cronograma de la Figura 2.7, la mayoría de las tareas realizadas se han completado dentro del plazo planificado, con un alto grado de adherencia a la planificación inicial.

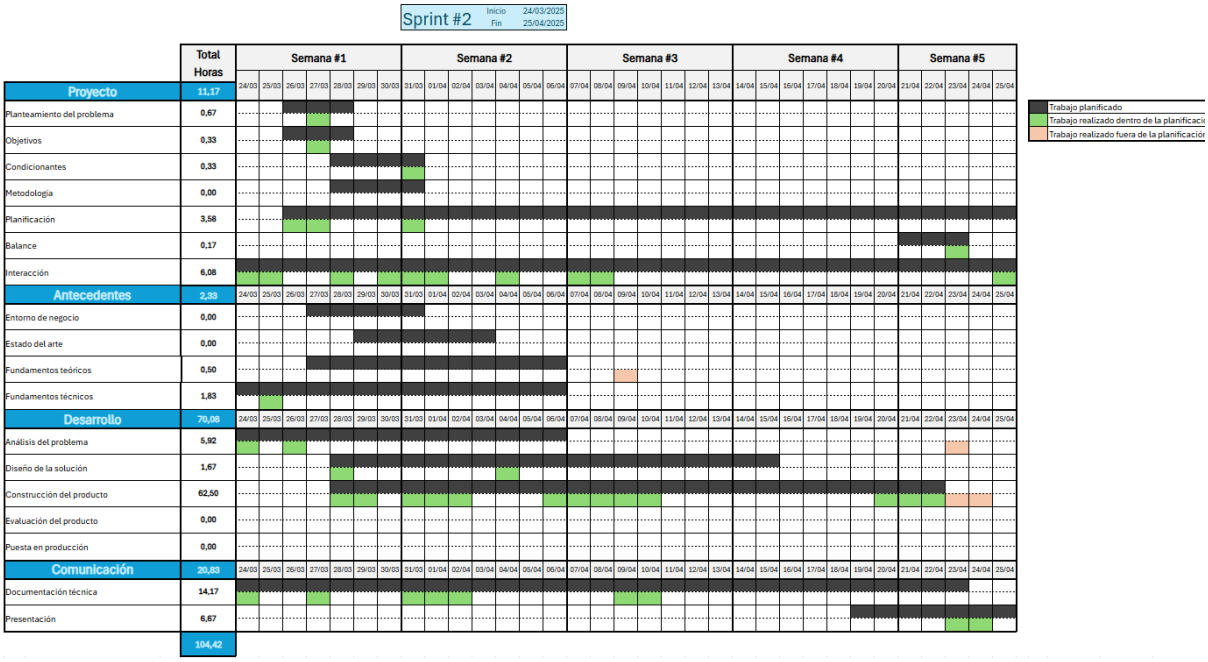

Figura 2.6: Cronograma del *sprint* 2.

Cabe destacar que algunas tareas planificadas, especialmente dentro de las actividades de "Análisis del problema", "Diseño de la solución", "Evaluación del producto" y "Aceptación", no llegaron a realizarse. Esta decisión fue tomada deliberadamente para priorizar el avance en la construcción del producto, con el objetivo de dejar la aplicación en un estado muy avanzado, casi finalizado, al cierre del *sprint*.

Balance temporal del *sprint* 4

Durante este cuarto *sprint* se han acumulado un total de 98,75 horas de trabajo. Como se aprecia en el cronograma de la Figura 2.7, la mayoría de las tareas realizadas se han completado dentro del plazo planificado, con un alto grado de adherencia a la planificación inicial. Ha habido tareas realizadas fuera de la planificación que son la construcción del producto y el análisis del problema. La primera se debe a que los resultados de la aceptación llevaron a hacer cambios en la aplicación y la segunda se debe a que las correcciones del profesor llevaron a realizar de nuevo el diagrama de casos de uso y las especificación de los casos de uso.

Balance temporal tras el *sprint* 4

Tras los 4 *sprints* no se requirió de otro quinto *sprint* pero sí hubo una semana más de trabajo. En esta semana se han dedicado al TFG 29 horas extra.

2.4.2. Balance económico

En este apartado se contrasta el presupuesto inicial presentado en la Sección 2.3 con los gastos reales al final del proyecto. Al igual que en el presupuesto, vamos a dividir los gastos en *hardware*, *software* y recursos humanos.

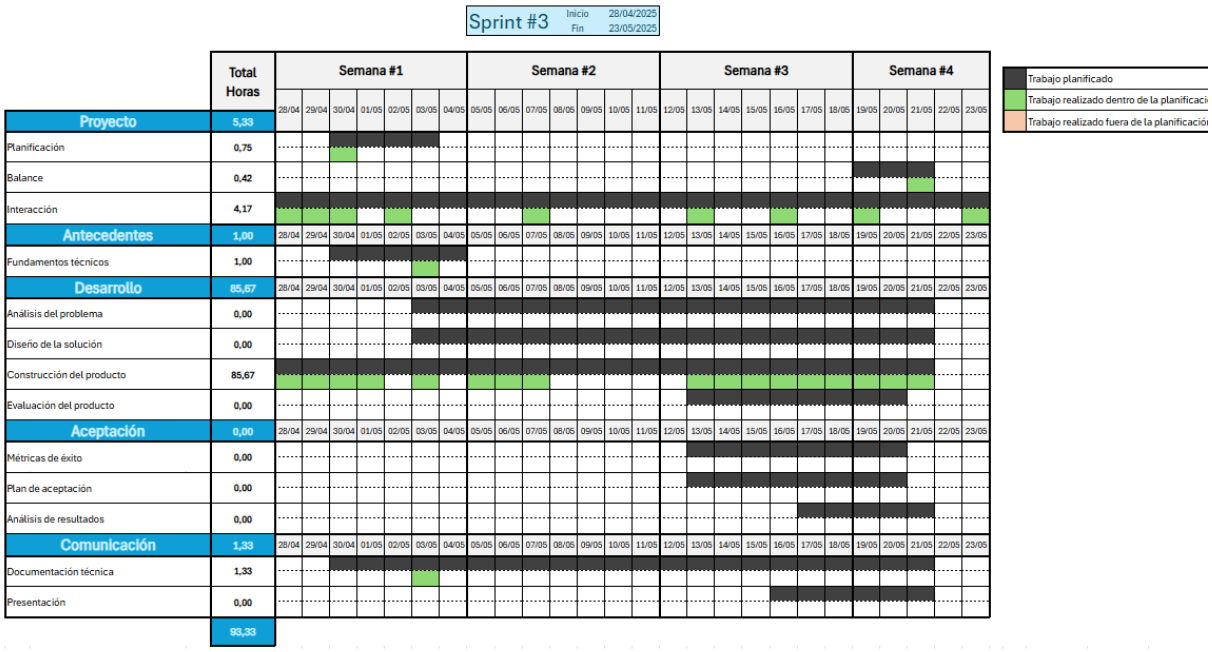

Figura 2.7: Cronograma del *sprint* 3.

Hardware

Para calcular el coste del *hardware* debemos tener en cuenta que el proyecto se alargó una semana, luego duró 5,25 meses en vez de 5 meses. Los gastos se detallan en la Tabla 2.6. Aplicando

Herramienta	**Coste total (€)**	**Vida útil (meses)**	**Porcentaje uso (%)**	**Uso en meses**
Ordenador portátil	1 099	$4,5 \times 12 = 54$	65	5,25
Wi-Fi	20 (mensuales)	1	15	5,25
Máquina virtual de despliegue	25 (mensuales)	1	100	1,25

Tabla 2.6: Coste final de *hardware*.

la ecuación (2.1) (tomando la vida útil como 1 cuando no procede):

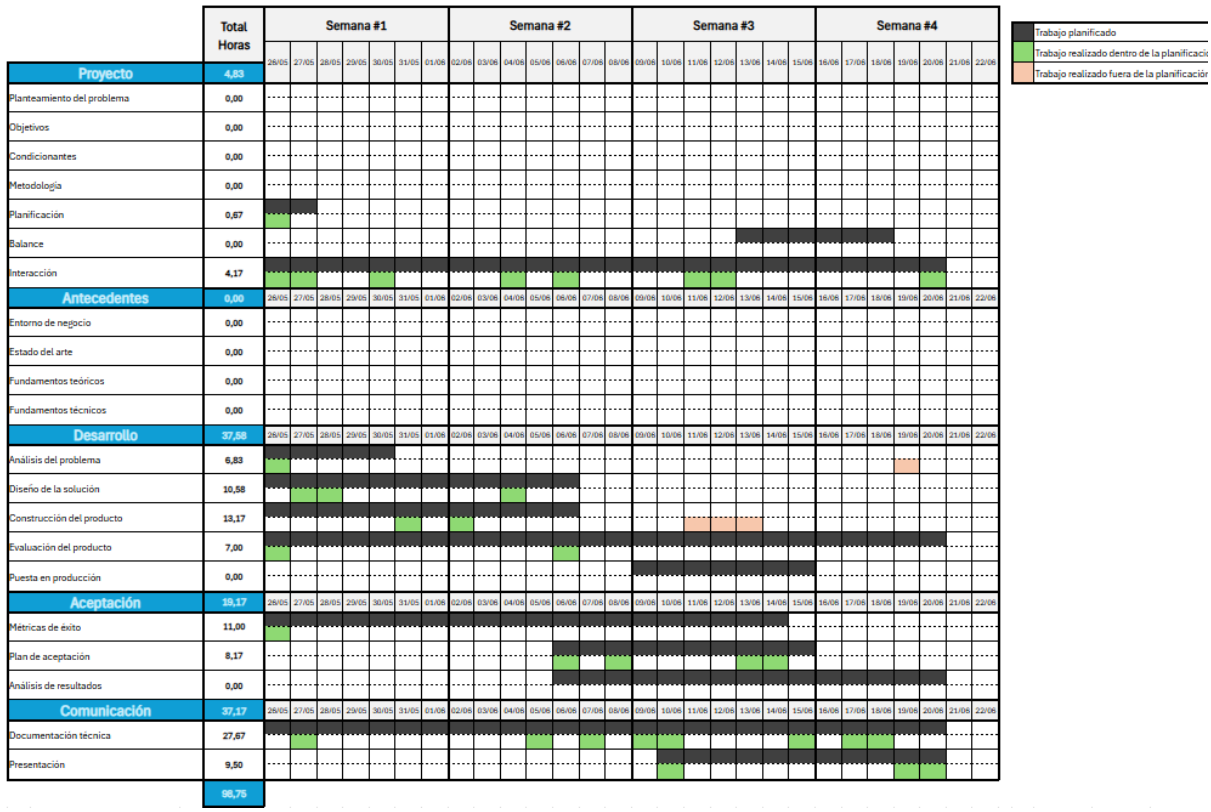

Figura 2.8: Cronograma del *sprint* 4.

Herramienta	**Coste real (€)**
Ordenador portátil	69,45
Wi-Fi	15,75
Máquina virtual de despliegue	31,25
Total	**116,45**

Tabla 2.7: Costes reales del *hardware*.

Por tanto, en *hardware* se han gastado 116,45 euros, en comparación con los 106,14 euros presupuestados.

Software

Como se indicó en los presupuestos la única herramienta software de pago con la que hemos trabajado es OpenAI para la que se presupuestaron 40 €, sin embargo, consultando la API solo se han gastado 15,74 €.

Recursos humanos

Este apartado cuantifica el coste humano considerando los distintos roles desempeñados por el estudiante durante las 390 horas (son 388,5 horas pero se ha redondeado) efectivas del proyecto.

La Tabla 2.8 recoge los salarios medios anuales y la participación de cada rol:

Rol	Salario anual (€)	Participación (%)
Analista de sistemas	33.000	10
Arquitecto de *software*	45.000	8
Desarrollador *frontend*	31.969	15
Desarrollador *backend*	34.497	42
Tester de *software*	41.200	10
Documentalista	23.585	15

Tabla 2.8: Presupuesto de *recursos humanos*.

Con 1.824 horas al año y aplicando las ecuaciones (2.3) y (2.4):

Rol	**Salario rol (€)**	**Coste rol (€)**
Analista de sistemas	705,59	934,91
Arquitecto de *software*	769,74	1.019,90
Desarrollador *frontend*	1.025,32	1.358,55
Desarrollador *backend*	3.097,92	4.104,75
Tester de *software*	880,92	1.167,22
Documentalista	756,43	1.002,27

Tabla 2.9: Coste real de los recursos humanos en el proyecto.

El presupuesto total destinado a recursos humanos asciende a **9.587,59 €**.

Comparativa global

Se detalla la desviación de cada tipo de gasto y el impacto global sobre el proyecto en la Tabla 2.10.

El proyecto finalizó con un **sobrecoste del 17,7 %** (1.460,56 €) respecto a la previsión inicial, debido principalmente a la semana adicional de trabajo y al aumento de horas efectivas en general. El incremento de una semana de trabajo y de 60 horas adicionales provocó un sobrecoste global del 17,7 %, concentrado mayoritariamente en los gastos de recursos humanos. La desviación en *hardware* es menor ($\leq$10 %) y refleja los gastos de utilizar los recursos *hardware* una semana más. Finalmente, la desviación de los costes *software* es de -60,7 % lo cual es muy positivo ya que se ha ahorrado mucho en este tipo de gastos.

Este balance económico permite justificar las diferencias con el presupuesto inicial y cuantificar con precisión la repercusión de la extensión temporal del proyecto.

Partida	**Presupuesto** (€)	**Gasto real** (€)	**Desviación** (€)	**Desviación (%)**
Hardware	106,14	116,45	+10,31	+9,7 %
Software	40,00	15,74	-24,26	-60,7 %
Recursos humanos	8.113,08	9.587,59	+1.474,51	+18,2 %
Total	**8.259,22**	**9.719,78**	**+1.460,56**	**+17,7 %**

Tabla 2.10: Comparativa entre presupuesto inicial y gasto real (software corregido).

Capítulo 3

Antecedentes

En este capítulo se presentan los conocimientos fundamentales necesarios para comprender el impacto del proyecto y el enfoque seguido para desarrollar el sistema informático.

En la Sección 3.1 se comentan en primer lugar los *stakeholders* y después se describe el diseño conceptual de los datos. A continuación, en la Sección 3.2 se expone el estado del arte relacionado con el proyecto. La Sección 3.3 está dedicada a la intelligencia artificial generativa. Finalmente, en la Sección 3.4.1 se hablará sobre la API de OpenAI.

3.1. Entorno de negocio

En esta Sección profundizaremos en el dominio del proyecto, para ello se comenzará explicando de forma muy detallada el diseño conceptual en la Sección 3.1.1 y finalmente se explicará quiénes son los *stakeholders* en la Sección 3.1.2.

3.1.1. Diseño conceptual

El diseño conceptual de una base de datos se define como"el proceso de construcción de un modelo de los datos utilizados en una empresa, independientemente de cualquier consideración física" [12]. El primer paso para realizar el diseño de una base de datos será comprender correctamente qué es lo que el cliente nos está pidiendo, o dicho de una forma más técnica, cuáles son los requisitos de información del proyecto que vamos a realizar. El diseño es un proceso iterativo, se repiten los diseños varias veces hasta que estos modelen correctamente el "mini-mundo" que debe representar la base de datos. Para asegurar que se modela correctamente el "mini-mundo", se irán validando los resultados obtenidos con el cliente para poder identificar y corregir posibles errores.

Para realizar el diseño conceptual existen dos técnicas principales que son la realización de diagramas Entidad-Relación y la creación de un diccionario de datos. Este proyecto se centrará en la primera de ellas.

Es fundamental que esta parte del diseño se realice correctamente ya que los diseños lógico y físico se construirán sobre ella, esto muestra la importancia de que los alumnos aprendan a hacer modelos ER de forma correcta, y por tanto, la importancia de este proyecto.

Modelo ER

El modelo ER [12] plantea una descripción visual del diseño conceptual de una base de datos que describe la realidad en términos de entidades, relaciones y atributos; y que se explicarán a continuación utilizando como referencia el diagrama de la Figura 3.1.

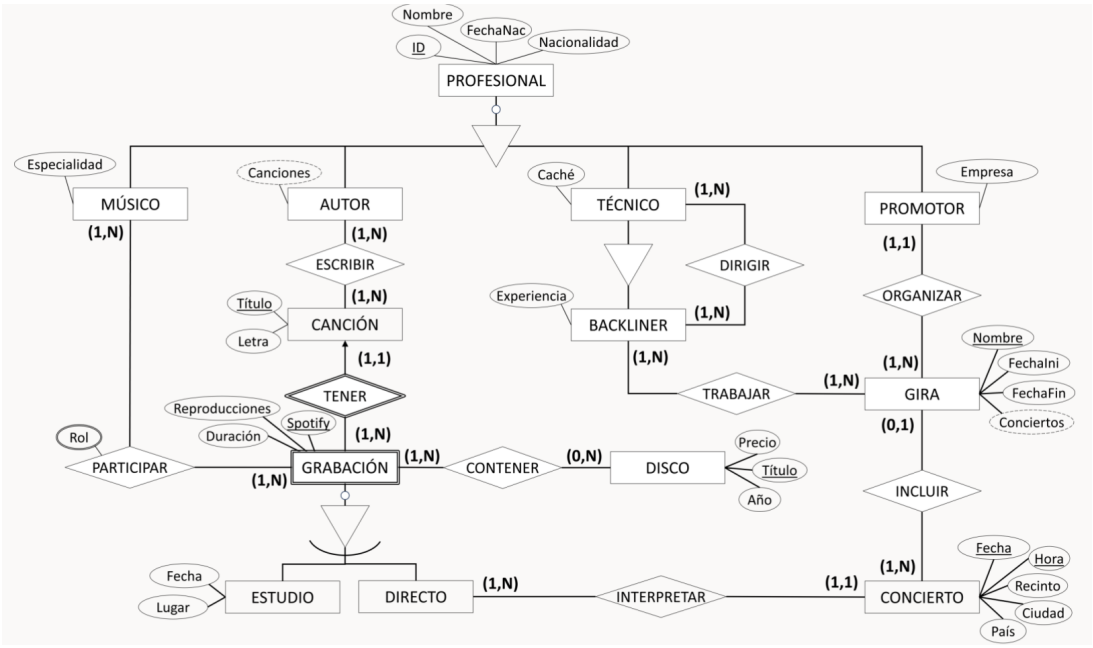

Figura 3.1: Ejemplo de diagrama entidad-relación.

Una **entidad** describe a un conjunto de elementos del mundo real que comparten las mismas propiedades y cuya existencia es independiente. Las entidades pueden ser fuertes, si su existencia no depende de ningún otro tipo de entidad, o débiles en caso contrario. Las entidades débiles pueden depender de una o más entidades fuertes a las que se les llamará "entidades padre".

En el modelo entidad-relación las entidades se representan con un rectángulo simple, si la entidad es fuerte, y doble, si la entidad es débil, dentro del rectángulo se indica el nombre de la entidad. Fijándonos en el modelo ER del ejemplo de la Figura 3.1 observamos que CANCIÓN sería una entidad fuerte mientras que GRABACIÓN sería una entidad débil que depende de la entidad CANCIÓN, lo cual tiene sentido pues para hacer una grabación en este contexto deben existir canciones que grabar. Cada entidad tiene varios **atributos** que la describen, y cada atributo tiene un dominio que establece el conjunto de posibles valores que puede tomar. En el diagrama entidad-relación, los atributos se representan rodeados por elipses y unidos con una línea a la entidad a la que están asociados. Hay diferentes tipos de atributos: en primer lugar, tendríamos los atributos simples, que se representan rodeados por una elipse simple. Por ejemplo, en el caso de la entidad DISCO sus atributos serían Precio, Título y Año y serían atributos simples. Otra clase de atributos serían los atributos multivaluados, en el diagrama entidad-relación estos estarían representados por una elipse doble y su particularidad es que pueden tomar varios valores. En el ejemplo de la Figura 3.1 encontramos el atributo Rol de la relación PARTICIPAR.

Por último, tenemos los atributos derivados, que son los atributos cuyo valor se calcula utilizando el valor de otros atributos, de la entidad a la que corresponden o incluso de otra entidad o relación. Un ejemplo de atributo derivado sería el atributo Conciertos de la entidad GIRA, que representa el número de conciertos que se han realizado en una gira y que en este caso se calcularía contando el número de conciertos que estan "incluidos" en cada gira. Además, observando el diagrama entidad-relación de la figura, podemos ver que algunos atributos están subrayados, estos atributos se dice que son identificadores de la entidad, en el caso de que la entidad sea fuerte, o discriminantes de la entidad, en el caso de que la entidad sea débil. Los identificadores de las entidades fuertes identifican unívocamente cada una de sus instancias ya que tienen valores únicos y están formadas por el mínimo número de atributos que asegura esto. En el caso de las débiles sucedería lo mismo, pero con la unión del discriminante y del identificador de su entidad padre.

Las diferentes entidades están unidas entre sí por **relaciones**. Las relaciones se representan en el modelo entidad-relación como rombos que contienen en su interior el nombre de la relación, y del rombo salen líneas hacia las entidades involucradas en la relación que pueden ser solo una (relaciones unarias), dos (relaciones binarias) o N (relaciones N-arias). Estas relaciones modelan las distintas interacciones entre las entidades en el "mini-mundo" en el que se está trabajando.

Las relaciones pueden ser débiles, si unen una entidad débil con su entidad padre o fuertes en caso contrario. Las relaciones débiles se representan por un rombo con doble línea y un ejemplo de estas relaciones sería la relación TENER entre GRABACIÓN y CANCIÓN, en este caso, GRABACIÓN sería la entidad débil o hija, mientras que CANCIÓN sería la entidad fuerte o padre; las relaciones fuertes se representan por un rombo con línea simple y un ejemplo en este diagrama entidad-relación, sería la relación INCLUIR entre GIRA y CONCIERTO, en este caso GIRA y CONCIERTO son ambas entidades fuertes. Otro ejemplo diferente sería la relación CONTENER entre GRABACIÓN y DISCO, aunque GRABACIÓN sea una entidad débil su entidad padre no es DISCO sino CANCIÓN y por tanto la relación es fuerte. Además, las relaciones tienen multiplicidades que nos indican cuántas instancias de una entidad pueden estar relacionadas con una instancia del resto de entidades de la relación. Las multiplicidades se representan por un par de números por cada entidad que participe en la relación: el primero representa la participación de la entidad y será 0 si la participación no es obligatoria y 1 en caso contrario; el segundo número, representa la cardinalidad y será 1 si aparece una vez como máximo, cualquier otro número si aparece como máximo ese número de veces y N si no hay un número máximo concreto de veces que puede aparecer. Para leer correctamente las multiplicidades deben leerse en orden: primero la entidad, después la relación, luego la multiplicidad y finalmente la otra entidad, por ejemplo, una GIRA (primera entidad) la organiza (relación ORGANIZAR) un único (pues la multiplicidad es (1,1), es decir mínimo uno y máximo uno) PROMOTOR (segunda entidad). Podemos leerlo comenzando por PROMOTOR y tendríamos que un PROMOTOR organiza una o más GIRAS.

Además, las relaciones también podrán tener uno o más atributos propios, como es el caso de la relación PARTICIPAR que tiene el atributo multivaluado Rol.

3.1.2. *Stakeholders*

En esta Sección hablaremos sobre los *stakeholders*, o lo que es lo mismo, los interesados del proyecto. A continuación, se presentan los diferentes *stakeholders* indicando su rol, su grado de influencia en el proyecto y sus intereses y necesidades.

1. **Alumno**
   - **Rol:** Usuario principal de la aplicación.
   - **Influencia:** Alta.
   - **Intereses y necesidades:**
     - Aprender a diseñar bases de datos a nivel conceptual. Esto se materializa en lo siguiente:
       - Subir su diagrama entidad-relación para recibir *feedback*.
       - Mejorar sus habilidades en modelado de bases de datos.
       - Recibir retroalimentación clara y útil para corregir errores.

2. **Profesor**
   - **Rol:** Facilitador del proceso de aprendizaje.
   - **Influencia:** Alta.
   - **Intereses y necesidades:**
     - Asegurar que los estudiantes comprendan el diseño conceptual de bases de datos.
     - Ahorrar tiempo en la revisión manual de diagramas.

3.2. Estado del arte

Si bien el modelo Entidad-Relación (ER) es fundamental en la enseñanza de bases de datos, los estudiantes suelen enfrentar dificultades al aprender a construir correctamente estos diagramas. Entre los problemas más comunes destacan: confusión entre entidades y atributos, dificultad para identificar relaciones adecuadas, errores al asignar cardinalidades y problemas al traducir requisitos del mundo real a componentes del modelo ER. Estas dificultades suelen generar errores recurrentes en los diagramas realizados por los alumnos, dificultando su comprensión y limitando su capacidad para diseñar bases de datos correctas. Por esta razón, es relevante desarrollar herramientas que apoyen el proceso de aprendizaje, ofreciendo retroalimentación inmediata y ayudando a los estudiantes a identificar y corregir sus errores en la construcción de modelos ER. En esta Sección se analizará el estado del arte, se explorarán investigaciones previas y soluciones existentes con el objetivo de contextualizar el problema y destacar la importancia de este trabajo, indicando los puntos nuevos que aporta en comparación con trabajos similares. En primer lugar, se plantearán artículos y proyectos que tengan que ver con nuestro proyecto después, en la Sección 3.2.9, se discutirán las diferencias y similitudes entre estos.

3.2.1. MonstER Park

MonstER Park [54] es un juego diseñado para enseñar los fundamentos de los diagramas ER de manera interactiva y entretenida, a través de distintas historias. Al comenzar el juego, el jugador verá una pantalla similar a la mostrada en la Imagen 3.2, donde aparecerán personajes planteando preguntas o contando una historia. En la parte inferior de la pantalla, el usuario deberá seleccionar el tipo de elemento que desea insertar (entidad, atributo, relación, etc.), así como asignarle un nombre adecuado para resolver cada desafío y avanzar en el juego.

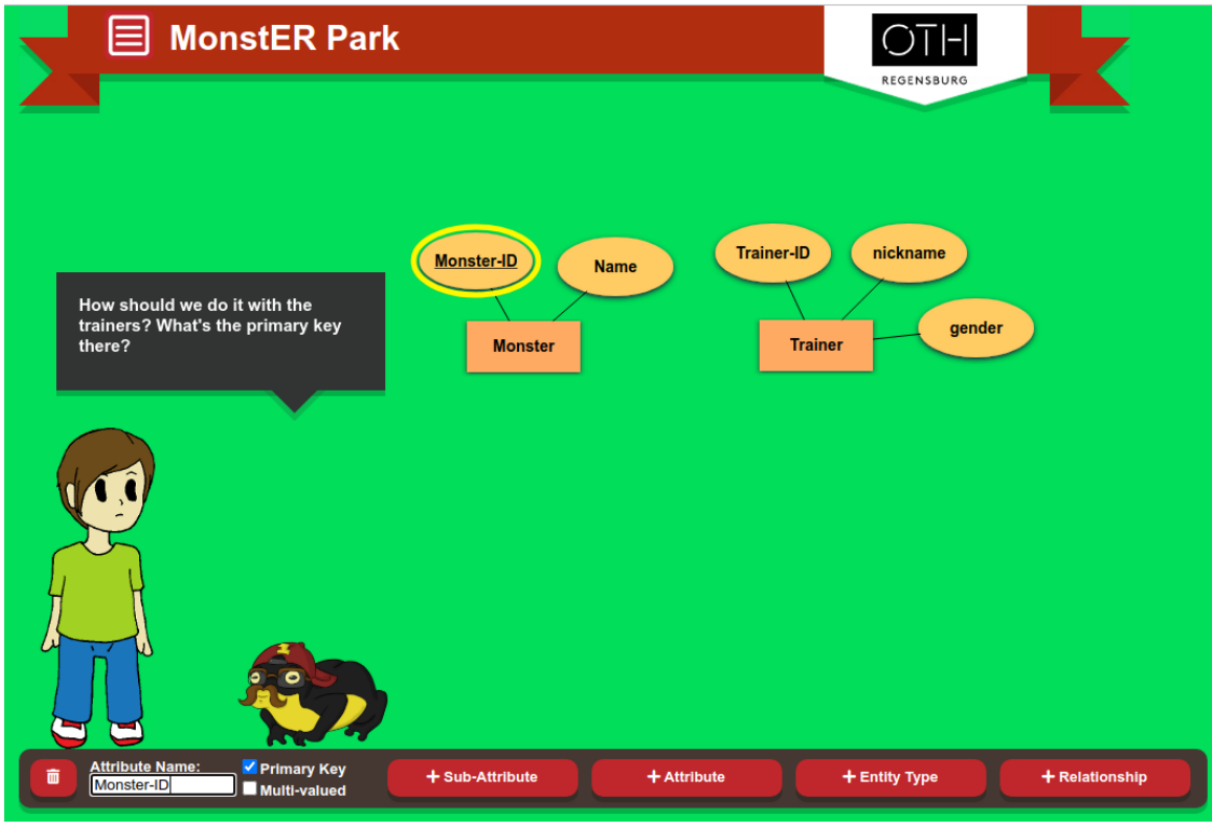

Figura 3.2: Interfaz de usuario del juego MonstER Park [54].

Una vez añadido el objeto correspondiente, este se representará visualmente: dentro de un rectángulo si se trata de una entidad, en un óvalo si es un atributo, y así sucesivamente. Cada vez que se introduce un concepto nuevo, el juego genera una explicación asociada. Además, el jugador recibe retroalimentación inmediata sobre sus errores y aciertos, lo que facilita el aprendizaje.

No obstante, la propuesta presenta una limitación específica. Aunque el programa cuenta con un conjunto de respuestas predefinidas para cada objeto, existe la posibilidad de que el usuario ingrese un término equivalente que, al no estar registrado en la lista de respuestas aceptadas, sea marcado como incorrecto. Por ejemplo, si el usuario nombra una entidad como “COCHE”, pero el sistema solo reconoce “AUTOMÓVIL” como respuesta válida, el programa podría indicarle un error, a pesar de que ambos términos sean sinónimos

3.2.2. AutoER

AutoER [17] es una herramienta diseñada para ayudar a los estudiantes a comprender el diseño de diagramas de bases de datos mediante una visualización interactiva y retroalimentación instantánea. Su propósito es guiar a los estudiantes a identificar y extraer los distintos elementos de un modelo ER a partir de un enunciado. Para ello, los estudiantes responden preguntas creadas por el profesor o generadas automáticamente, interactuando con el texto para construir el diagrama. En lugar de dibujar manualmente los componentes del modelo, como sucede en otros programas de diseño UML, los usuarios simplemente seleccionan el tipo de elemento que desean agregar (entidad, atributo, relación), y el sistema genera el diagrama de manera dinámica según sus respuestas.

Desde el lado del profesor, AutoER ofrece amplias opciones de configuración como se muestra en la Figura 3.4. Permite personalizar la interfaz que se presenta a los estudiantes, ajustar los criterios de evaluación, modificar la retroalimentación proporcionada y definir los parámetros para la generación de preguntas. Además, los profesores pueden habilitar un entorno de práctica donde los estudiantes puedan familiarizarse con la herramienta antes de realizar evaluaciones formales.

En la Figura 3.3 se puede observar la interfaz del sistema y el proceso de respuesta de un estudiante. El enunciado del problema incluye un formato interactivo que permite seleccionar palabras o frases clave para incorporarlas al diagrama. Los menús contextuales facilitan la adición de entidades y atributos, y a medida que se avanza en la construcción del modelo, es posible modificarlo agregando o eliminando elementos como entidades, atributos, claves y relaciones. Para evaluar la respuesta, el sistema convierte el diagrama en una representación textual y genera comentarios en función de su precisión.

3.2.3. DiagrammER

Este trabajo [30] presenta una aplicación web concebida como una herramienta de apoyo en la enseñanza de asignaturas de grado sobre bases de datos. Su finalidad es proporcionar a los estudiantes un entorno práctico en el que puedan desarrollar sus habilidades en modelado de datos mediante diagramas Entidad-Relación (ER).

La plataforma está diseñada para su uso tanto por parte del profesorado como del alumnado. Los docentes pueden preparar ejercicios y ejemplos, algunos de los cuales incluyen diagramas ER para ilustrar conceptos clave. Los estudiantes, por su parte, pueden crear y modificar sus propios diagramas en cualquier momento, así como acceder a los ejercicios diseñados por el profesor para resolverlos y así poder reforzar su aprendizaje.

En la Figura 3.5 se puede visualizar la interfaz de usuario de DiagrammER y observar como añadir entidades, relaciones, modificar el tipo de letra utilizado etc.

3.2.4. Simanjuntak

Este artículo [58] presenta un sistema de calificación automática para diagramas de entidad-relación utilizando archivos en formato XML, con dos enfoques principales. El primero se basa en la medición de similitud a través del algoritmo Tree Edit Distance como vemos en la Figura 3.6, que convierte los diagramas ER creados por los estudiantes y la solución del profesor en archivos XML para luego compararlos. Este algoritmo calcula la cantidad mínima de modificaciones necesarias para transformar un árbol en otro, permitiendo así generar un puntaje de similitud

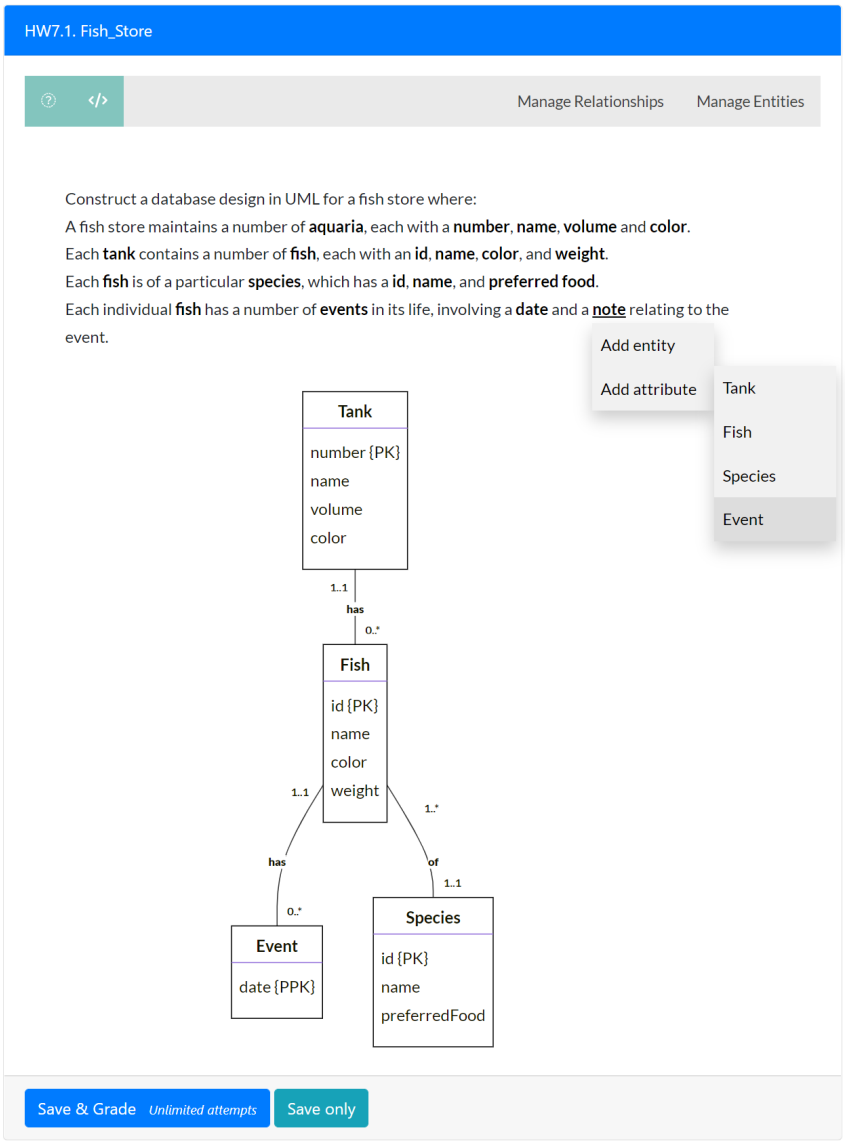

Figura 3.3: Respuesta a una pregunta por parte del estudiante en AutoER. [17]

y proporcionar retroalimentación sobre diferencias estructurales, atributos, relaciones y nivel de normalización.

El segundo enfoque utiliza técnicas de aprendizaje automático para calificar los diagramas ER. Se extraen características clave como entidades, atributos, relaciones y niveles de normalización, que luego sirven para entrenar un modelo basado en ejemplos previamente calificados por expertos. De esta manera, el modelo aprende a evaluar y asignar calificaciones de manera automática, comparando sus resultados con los de evaluaciones humanas para mejorar su precisión con el tiempo.

El objetivo principal del sistema es agilizar la evaluación de diagramas ER, garantizando calificaciones consistentes y ofreciendo retroalimentación inmediata tanto para estudiantes como para docentes en entornos educativos y de desarrollo de software.

pl-ubco-umldemo / questions / Fish_Store / question.html

Edit | Upload | Download | Rename | Delete

```
<pl-uml-element random="False" max-grade = "10">
    <uml-question>Construct a database design in UML for a fish store where:
        A fish store maintains a number of [aquaria](tank), each with a [number](number), [name](name), [volume](volume) and [color](color).
        Each [tank](tank) contains a number of [fish](fish), each with an [id](id), [name](name), [color](color), and [weight](weight).
        Each [fish](fish) is of a particular [species](species), which has a [id](id), [name](name), and [preferred food](preferredFood).
        Each individual [fish](fish) has a number of [events](event) in its life, involving a [date](date) and a [note](note) relating to the event.</uml-q
    <uml-answer>[Tank|number {PK};name;volume;color]
[Fish|id {PK};name;color;weight]
[Species|id {PK};name;preferredFood]
[Event|date {PPK};note]
[Tank] 1..1 - 0..* [Fish]
[Fish] 1..* - 1..1 [Species]
[Fish] 1..1 - 0..* [Event]</uml-answer>
    <uml-marking entity-name="0.2" entity-attributes="0.1" entity-key="0.2" extra-entity-penalty="0.25" weak_entity="0.5" relationship="0.5" cardinality="0
</pl-uml-element>
```

Figura 3.4: Creación de una pregunta por un profesor [17].

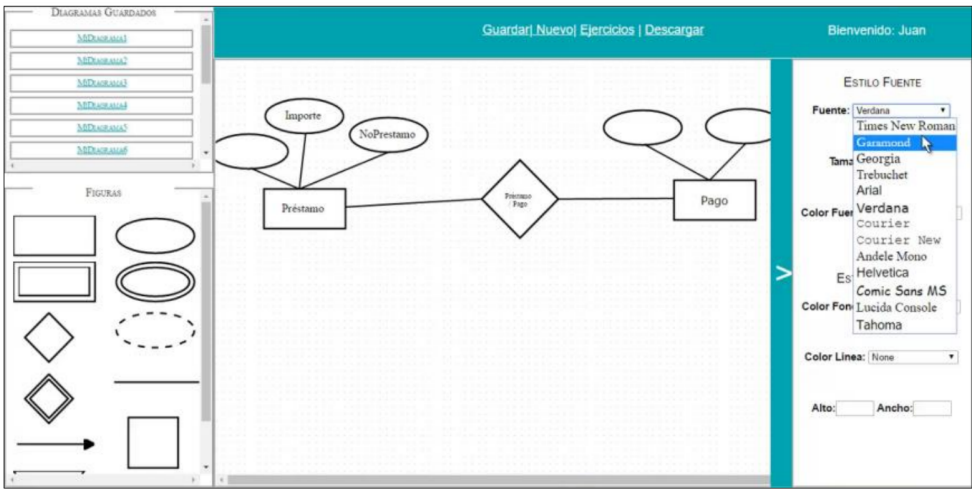

Figura 3.5: Interfaz de usuario de DiagrammER [30].

3.2.5. SAT

En este proyecto [18], se ha empleado un modelo basado en redes neuronales para identificar los distintos elementos (clases y relaciones) de un diagrama UML y extraer el texto asociado a cada uno de ellos. El proceso comienza con un preprocesamiento de la imagen, que incluye ajustes de tamaño, brillo y contraste, eliminación de ruido y detección de contornos. Posteriormente, se utiliza un detector de objetos basado en la arquitectura Faster-RCNN para localizar y detectar los diversos componentes del diagrama. Después, se aplican redes neuronales recurrentes convolucionales para analizar el texto de cada componente. Finalmente, se replica la estructura y el texto en formato digital utilizando una biblioteca gráfica.

Para entrenar el modelo, se utilizó un conjunto de diagramas UML dibujados a mano y otro generado digitalmente. Para evaluar su rendimiento, se probaron tres tipos de entradas: diagramas UML dibujados a mano, diagramas con tinta inteligente y diagramas digitales. Los

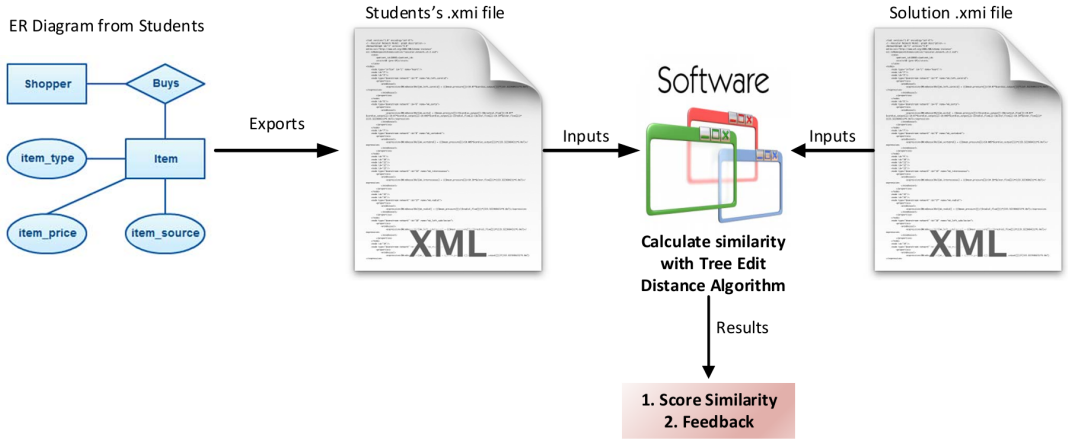

Figura 3.6: Evaluación de la similitud del diagrama ER utilizando el algoritmo de Tree Edit Distance [58].

resultados obtenidos muestran que el modelo presenta un mejor desempeño con los diagramas digitales, seguido por los de tinta inteligente, y, finalmente, con los realizados a mano.

3.2.6. DBReader

Este proyecto [38] proporciona una herramienta de autoevaluación de modelos ER que porporciona *feedback* instantáneo de un modelo ER. Para ello, el estudiante proporciona la imagen de su modelo ER y la aplicación le proporciona *feedback* partiendo de una solución de referencia. DBReader utiliza visión computacional y reconocimiento óptico de caracteres para interpretar el modelo ER, de acuerdo con las fases incluidas en el recuadro azul del *workflow* mostrado en la Figura 3.7.

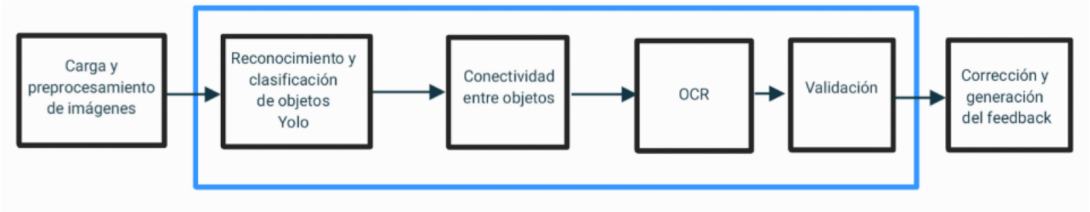

Figura 3.7: Workflow del proyecto [38].

Cuando se introduce la imagen, el sistema reconoce las entidades, Figura 3.8, las relaciones Figura 3.9 y sus atributos. Finalmente, una vez extraído el modelo, el sistema lo compara respecto a una solución de referencia y le entrega al estudiante una retroalimentación detallada por criterios de evaluación, como se muestra en la Figura 3.10.

3.2.7. LIJ

Este proyecto [33] se centra en desarrollar una solución asistida por inteligencia artificial que facilite el diseño de arquitecturas de software siguiendo la metodología ADD 3.0. Esto significa

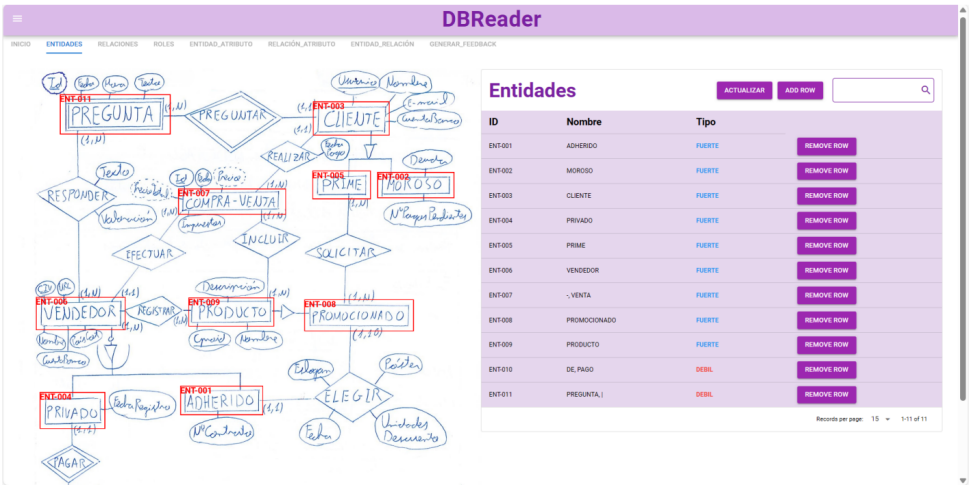

Figura 3.8: Interfaz de usuario del sistema donde se ve la lista de entidades [38].

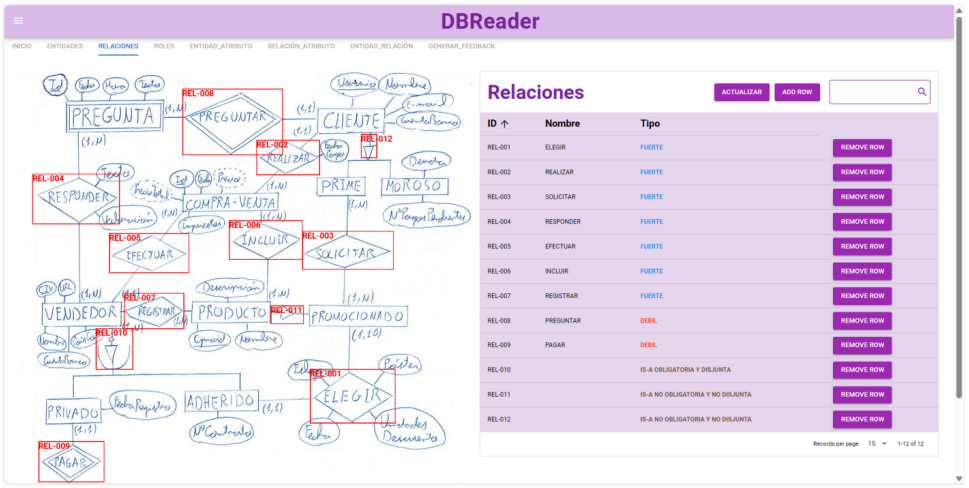

Figura 3.9: Interfaz de usuario del sistema donde se ve la lista de relaciones [38].

que, en lugar de abordar el diseño arquitectónico de forma *ad-hoc*, se utiliza un proceso estructurado y basado en atributos de calidad y requisitos del sistema. El proceso se divide en iteraciones bien definidas que parten de la identificación de *drivers* clave y atributos de calidad, para luego seleccionar patrones, tácticas y conceptos de diseño que aseguren una implementación coherente y alineada con los objetivos del sistema.

Para ello se plantea la construcción de varias herramientas especializadas en tareas específicas, y que sean capaces de comunicarse entre sí para construir resultados mucho más elaborados que son:

- ***Researcher*:** herramienta para apoyar a los arquitectos de *software* que necesiten comprender y aplicar la metodología ADD 3.0 y las tácticas de arquitectura enfocadas en desempeño y disponibilidad.

- ***Classifier*:** es un sistema basado en aprendizaje automático (Machine Learning) diseñado

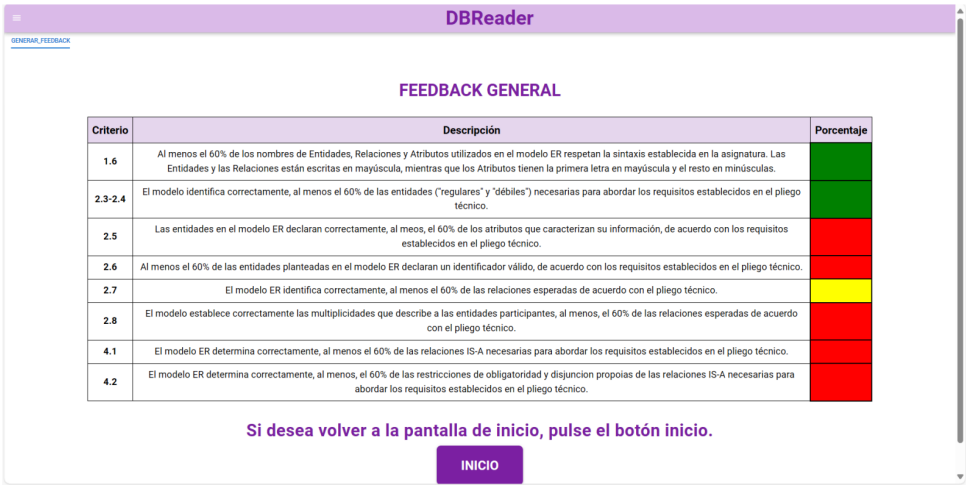

Figura 3.10: Interfaz de usuario del sistema donde se ve el feedback recibido [38].

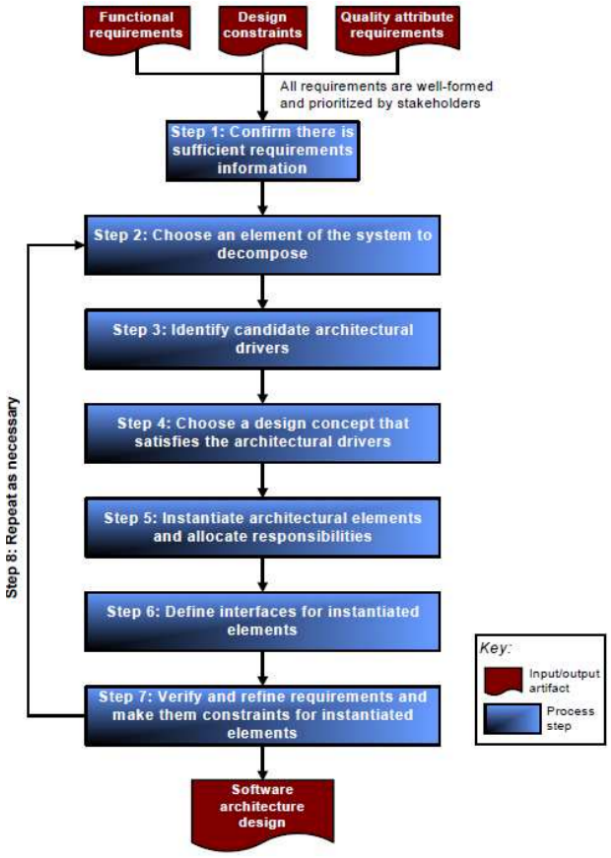

Figura 3.11: Etapas del proceso de la metodología ADD 3.0 [35].

específicamente para la clasificación de diagramas de arquitectura de software.

- ***Describer*:** herramienta diseñada para proporcionar explicaciones detalladas sobre tácticas

específicas de arquitectura de software.

- ***Creator*:** se utiliza para generar código XML que permita la creación de diagramas de arquitectura de software.

3.2.8. GAI4DB

En este artículo [10] se pone en valor el uso de la Inteligencia Artificial Generativa en entornos educativos, en concreto, para que los alumnos aprendan a realizar consultas SQL. Se propone un *framework* para ello, que se resume de forma visual en la Figura 3.12 y se hace una comparativa entre distintas herramientas del IA Generativa para el desarrollo de habilidades SQL. Este *framework* propone el uso de herramientas de inteligencia artificial generativa (GAI) en la enseñanza de SQL, asumiendo que los estudiantes tienen conocimientos básicos sobre bases de datos relacionales, el proceso puede resumirse con los siguientes pasos:

1. **Configuración del entorno**: El docente prepara el servidor de bases de datos y los ejercicios.
2. **Interacción con la GAI**: Los estudiantes consultan dudas y reciben pistas sobre SQL.
3. **Respuestas de la GAI**: La herramienta proporciona explicaciones y sugerencias para resolver los ejercicios.
4. **Redacción de consultas SQL**: Los estudiantes escriben sus consultas basándose en la ayuda recibida.
5. **Ejecución y prueba**: Se ejecutan las consultas en la base de datos para verificar su validez.
6. **Revisión de resultados**: Se analizan los resultados obtenidos para identificar posibles errores.
7. **Corrección y validación**: Los estudiantes ajustan sus consultas hasta obtener los resultados esperados.
8. **Entrega del ejercicio**: Se envían las soluciones al docente para su evaluación y retroalimentación.

Este enfoque permite un aprendizaje más interactivo y autónomo, facilitando la comprensión de SQL mediante asistencia en tiempo real.

3.2.9. Discusión

En esta Sección se discutirán y compararán los artículos planteados anteriormente de acuerdo a 10 dimensiones que se explican en la Tabla 3.1.

Para que la comparativa se pueda hacer de manera más visual y más fluida organizamos los artículos presentados anteriormente en la Tabla 3.2.

Finalmente, en la Tabla 3.3 presentamos de forma muy visual la comparativa realizada entre los diferentes artículos y nuestro proyecto a partir de las dimensiones antes mencionadas.

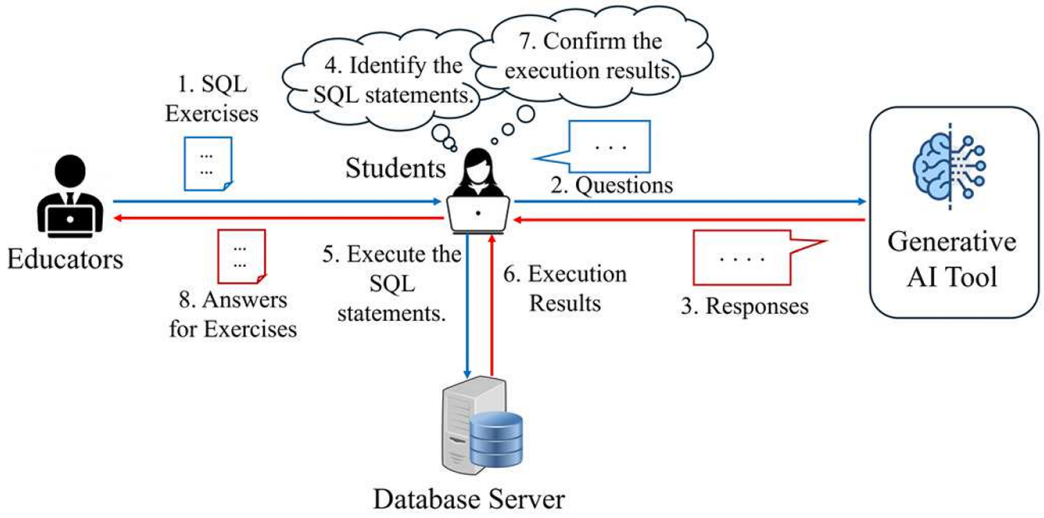

Figura 3.12: *Framework* propuesto en [10].

Artículo	**Dim-01**	**Dim-02**	**Dim-03**	**Dim-04**	**Dim-05**	**Dim-06**	**Dim-07**	**Dim-08**	**Dim-09**	**Dim-10**
ART-01		✓			✓	✓	✓			
ART-02		✓			✓		✓			
ART-03		✓			✓					
ART-04		✓	✓	✓	✓					
ART-05			✓					✓		
ART-06		✓	✓		✓			✓	✓	
ART-07	✓									✓
ART-08										✓
LearnER	✓	✓	✓		✓			✓	✓	✓

Tabla 3.3: Comparativa usando las dimensiones de la Tabla 3.1 de los artículos presentados en la Tabla 3.2 con nuestro proyecto.

Los primeros 4 artículos proponen una aplicación web que permite al usuario crear y/o corregir diagramas ER y le proporciona un feedback inmediato, además cuentan con ejercicios extra para practicar. El artículo ART-03 además permite calificar los exámenes de los estudiantes.

Es importante destacar que los únicos proyectos que trabajan con inteligencia artificial generativa (IAG) son el ART-07, el ART-08 y nuestro proyecto. El ART-08 plantea un *framework* muy interesante con el que trabajar para utilizar IAG en el aprendizaje de SQL y el ART-07 propone sobre un sistema que se aplica a la ingeniería de software en general, pero ninguno de ellos aborda el aprendizaje del diagrama ER.

Por otra parte, el proyecto ART-06 proporciona una aplicación web que es capaz de leer imágenes, compararlas y dar un feedback cuantitativo al estudiante pero no es capaz de proporcionar

ID	Dimensión	Descripción
Dim-01	Feedback cualitativo	Capacidad de la aplicación de proporcionar al estudiante un feedback cualitativo, no solo cuantitativo, explicando errores.
Dim-02	Feedback inmediato	Capacidad de la aplicación de proporcionar feedback de manera inmediata al estudiante.
Dim-03	Diagramas externos	Evalúa si la aplicación incorpora diagramas ER externos en su funcionalidad.
Dim-04	Uso en exámenes	Verifica si la herramienta se utiliza en la realización de exámenes.
Dim-05	Uso para practicar	Examina si la aplicación se utiliza para prácticas en la creación de diagramas ER a partir de un enunciado.
Dim-06	Explicación de conceptos	Indica si la herramienta incluye explicaciones detalladas de conceptos durante su uso.
Dim-07	Generación de preguntas	Evalúa si la aplicación genera preguntas relacionadas con el contenido para la creación de los diagramas ER.
Dim-08	Lectura de imágenes	Verifica si la herramienta puede interpretar y analizar imágenes externas a ella.
Dim-09	Comparación de imágenes	Indica si la aplicación realiza comparaciones entre diferentes imágenes.
Dim-10	Uso de IA Generativa	Indica si la aplicación utiliza Inteligencia Artificial Generativa para alguna de sus funcionalidades.

Tabla 3.1: Tabla que muestra las dimensiones con las que se van a evaluar los artículos presentados en la Sección 3.2.

un feedback cualitativo, ni utiliza IAG.

En nuestro proyecto, se plantea el desarrollo de una aplicación web capaz de procesar imágenes de diagramas ER identificando y analizando sus elementos. Luego, comparará el diagrama proporcionado por el estudiante con el del profesor, ofreciendo así 2 feedbacks diferentes e inmediatos: un feedback cuantitativo y un feedback cualitativo.

Lo innovador de nuestra propuesta es que, de todos los proyectos analizados es el único que aprovecha las ventajas de la IAG para evaluar diagramas ER y proporciona *feedback* inmediato al estudiante,.

ID	Artículo	Cita
ART-01	MonstER Park	[54]
ART-02	AutoER	[17]
ART-03	DiagrammER	[30]
ART-04	Simanjuntak	[58]
ART-05	SAT	[18]
ART-06	DBReader	[38]
ART-07	LIJ	[33]
ART-08	GAI4DB	[10]

Tabla 3.2: Tabla con la relación de los artículos del estado del arte con sus IDs y su cita.

3.3. Inteligencia artificial generativa

En esta Sección se comentará muy detalladamente la inteligencia artificial generativa (IAG). En primer lugar, comentaremos los fundamentos del la IAG en la Sección 3.3.1, continuaremos hablando en la Sección 3.3.2 sobre las arquitecturas de los modelos, después pasaremos a comentar sobre los LLMs en la Sección 3.3.3, posteriormente sobre el *workflow* general en la Sección 3.3.4, luego sobre las limitaciones de la IAG en la Sección 3.3.5 y finalmente sobre la ingeniería de *prompting* en la Sección 3.3.6.

3.3.1. Fundamentos

En [48] se define la inteligencia artificial como un campo de la informática que se enfoca en crear sistemas que puedan realizar tareas que normalmente requieren inteligencia humana, como el aprendizaje, el razonamiento y la percepción. Este término engloba a otros términos que serán de utilidad en este TFG como se observa en la Figura 3.13 y que se explicarán a continuación.

Aprendizaje automático

El aprendizaje automático o Machine Learning (ML) es un subcampo de la inteligencia artificial (IA) que se enfoca en el desarrollo de algoritmos capaces de resolver tareas de manera autónoma mediante la exposición a datos, sin necesidad de ser programados explícitamente. En otras palabras, estos algoritmos aprenden a partir de la información disponible y mejoran su funcionamiento en función de la experiencia.

Existen distintos enfoques de aprendizaje automático [38] según la naturaleza de los datos y el objetivo del análisis:

- **Aprendizaje supervisado o *Supervised Learning*:** El propósito de este tipo de aprendizaje es entrenar modelos utilizando un conjunto de datos etiquetados para realizar predicciones o clasificar datos no observados previamente. En la programación tradicional, se introducen datos y un programa en un ordenador, que luego genera una salida. Sin embargo, en el aprendizaje automático, se proporciona un conjunto de datos de entrada y un

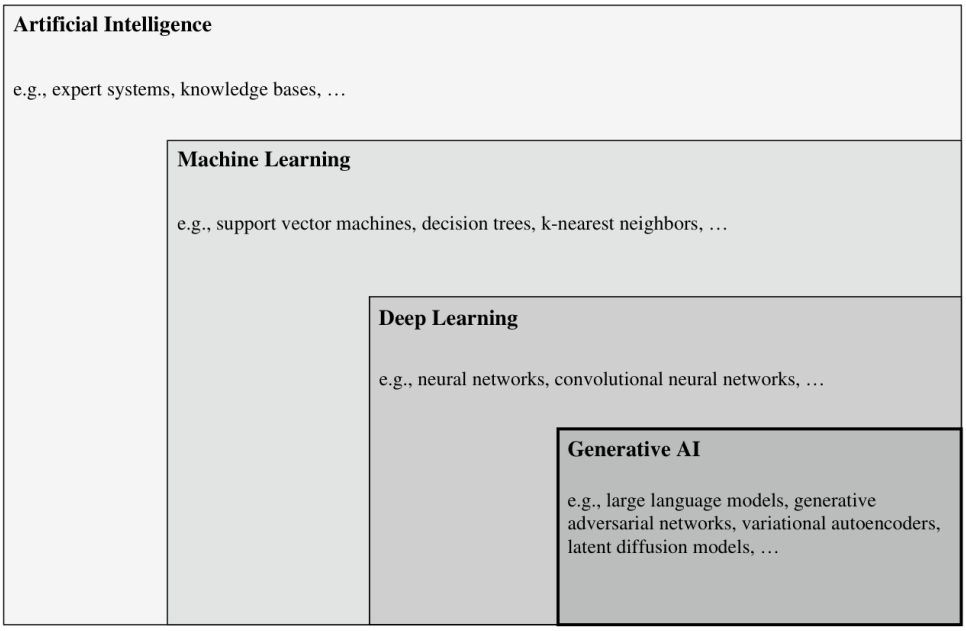

Figura 3.13: Jerarquía de términos dentro de la IA [6]

conjunto de datos de salida, y el ordenador produce como resultado un modelo (programa) [23].

- **Aprendizaje no supervisado o *Unsupervised Learning*:** Su objetivo es visualizar, preprocesar o identificar subgrupos dentro de los datos sin emplear etiquetas. Existen dos enfoques principales en el aprendizaje no supervisado: el clustering o agrupamiento, que consiste en identificar conjuntos de datos similares, y la reducción de dimensionalidad, que busca disminuir la cantidad de variables manteniendo la información esencial [21].

  En la Figura 3.14 encontramos una comparativa visual entre el aprendizaje supervisado y el no supervisado. En la parte superior, el aprendizaje supervisado utiliza un conjunto de datos acompañado de etiquetas que indican la categoría de cada elemento. Un modelo procesa esta información para aprender a clasificar correctamente nuevos datos en función de sus características. En cambio, en la parte inferior, el aprendizaje no supervisado trabaja únicamente con datos sin etiquetas. Aquí, el modelo analiza los datos y encuentra patrones o estructuras por sí mismo, agrupando elementos similares sin conocer previamente sus categorías.

- **Aprendizaje por refuerzo o Reinforcement Learning:** En este caso, la máquina aprende a partir de su propia experiencia, interactuando con el entorno hasta lograr un comportamiento óptimo. A partir de la información disponible, lleva a cabo acciones que repetirá y reforzará en función de las recompensas obtenidas, las cuales pueden ser positivas o negativas [5]. La Figura 3.15 ilustra un ejemplo de aprendizaje por refuerzo, donde un agente interactúa con un entorno, recibiendo un estado y tomando acciones. Un intérprete evalúa el desempeño del agente y proporciona una recompensa, que guía el aprendizaje para mejorar futuras decisiones.

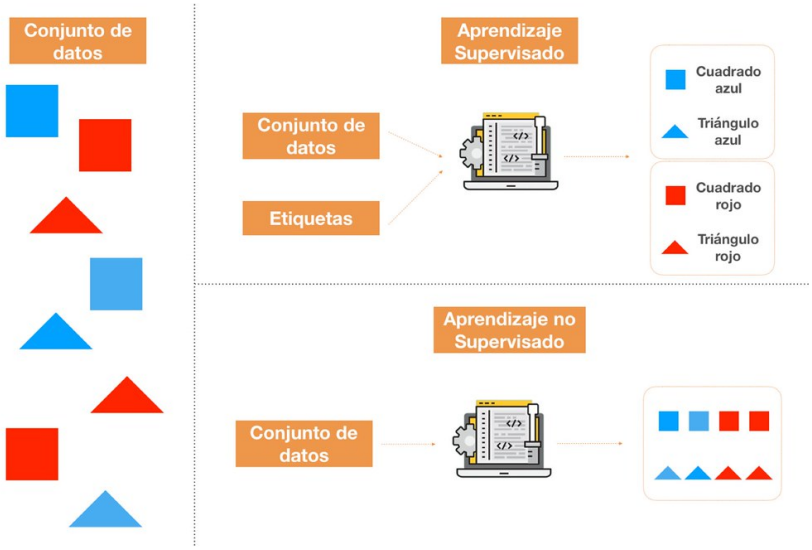

Figura 3.14: Comparativa del aprendizaje automático supervisado y no supervisado [19].

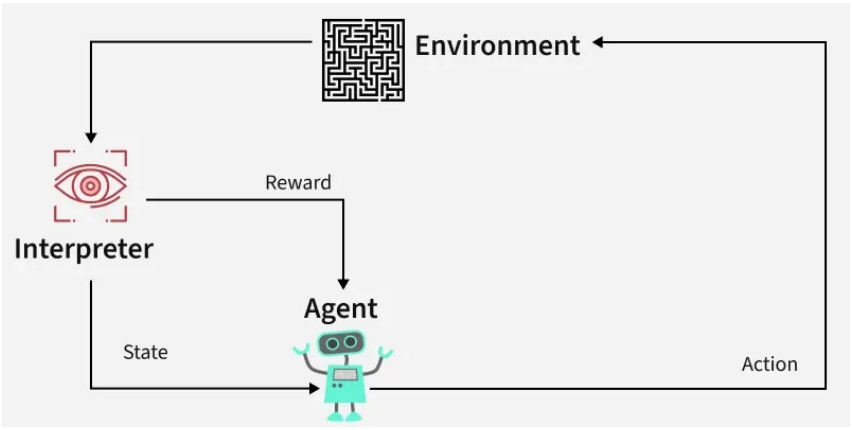

Figura 3.15: Funcionamiento del aprendizaje automático por refuerzo [**imgrefuerzo**]

- **Aprendizaje semi-supervisado o *Semi-Supervised Learning*:** Este enfoque combina el uso de datos etiquetados y no etiquetados en el proceso de aprendizaje [22]. La Figura 3.16 muestra un ejemplo de aprendizaje semi-supervisado en el que vemos que primero, el modelo se entrena con una pequeña cantidad de datos etiquetados y luego usa datos sin etiquetas para mejorar su precisión a través de un algoritmo, generando un modelo más efectivo.

- **Aprendizaje por transferencia o *Transfer Learning*:** Se basa en aprovechar patrones o modelos previamente entrenados en una tarea para aplicarlos a la resolución de un problema específico [65]. La Figura 3.17 presenta un ejemplo de aprendizaje por transferencia, donde un modelo previamente entrenado para una tarea (como identificar perros) se reutiliza y ajusta para una nueva tarea (como identificar gatos), en lugar de entrenarlo desde cero, lo que ahorra tiempo y recursos.

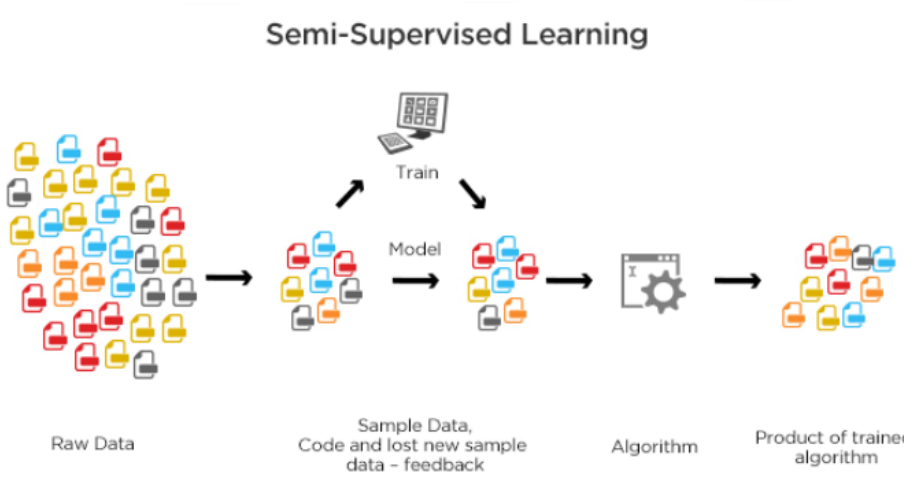

Figura 3.16: Funcionamiento del aprendizaje automático semi-supervisado [28]

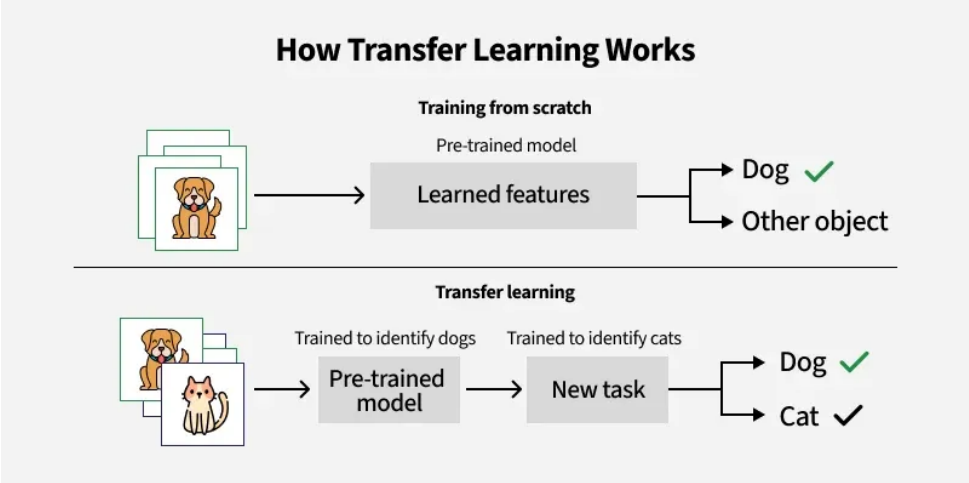

Figura 3.17: Ejemplo del aprendizaje automático por transferencia [61]

Deep learning

El *deep learning* [24] es "un subconjunto del *machine learning* que utiliza redes neuronales multicapa, llamadas redes neuronales profundas, para simular el complejo poder de toma de decisiones del cerebro humano. Algunas formas de *deep learning* impulsan la mayoría de las aplicaciones de inteligencia artificial (IA) en nuestra vida actual".

La diferencia fundamental entre *deep learning* y *machine learning* radica en la arquitectura de la red neuronal utilizada. Mientras que los modelos tradicionales de *machine learning* emplean redes neuronales simples con una o dos capas computacionales, los modelos de *deep learning* cuentan con tres o más capas, llegando en muchos casos a cientos o incluso miles de capas para entrenar los modelos. Se puede visualizar la diferencia en la Figura 3.18. En ella vemos como en el *machine learning*, primero se realiza una extracción manual de características antes de que una red neuronal haga la clasificación y en el *deep learning*, una red neuronal profunda realiza tanto

la extracción de características como la clasificación automáticamente, sin intervención humana.

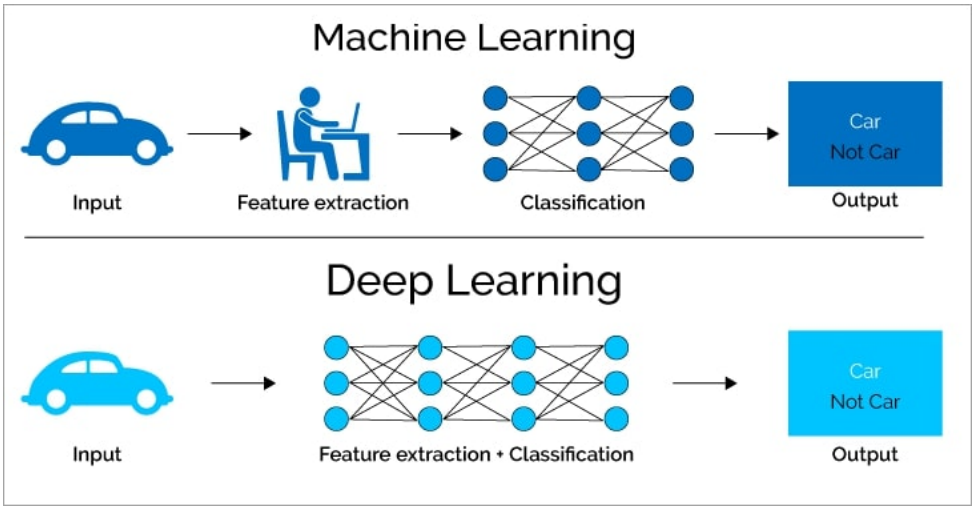

Figura 3.18: Comparativa entre el Machine Learning y el Deep Learning [63].

El *deep learning* es una rama de la inteligencia artificial que impulsa diversas aplicaciones y servicios enfocados en la automatización, permitiendo la ejecución de tareas analíticas y físicas sin intervención humana. Gracias a esto, es posible el desarrollo de productos y servicios de uso cotidiano, como asistentes digitales, controles remotos de TV con reconocimiento de voz, detección de fraudes con tarjetas de crédito, vehículos autónomos e inteligencia artificial generativa.

3.3.2. Arquitecturas de modelos de IA generativa y su evolución

Podemos definir la IAG como "una inteligencia artificial (IA) que puede crear contenido original (como texto, imágenes, video, audio o código de software) en respuesta a una instrucción o un mensaje del usuario" [26]. En los últimos años, la inteligencia artificial generativa ha experimentado un avance significativo gracias al desarrollo de diversas arquitecturas de modelos de *deep learning*. A continuación, se describen las principales arquitecturas que han impulsado esta evolución [26].

- **Autocodificadores variacionales (VAE):** Los autocodificadores [26] son modelos de *deep learning* compuestos por dos redes neuronales: una que comprime datos y otra que los reconstruye. Aunque pueden generar contenido, su principal utilidad radica en la compresión y recuperación de información. Los Autocodificadores Variacionales (VAE), introducidos en 2013, mejoraron esta técnica al permitir la generación de múltiples variaciones de datos con mayor precisión permitiendo la detección de anomalías y la generación de lenguaje natural.

- **Redes generativas adversarias (GAN):** Las GAN [26], introducidas en 2014, también constan de dos redes neuronales: un generador, que crea nuevos contenidos, y un discriminador, que evalúa la precisión y la calidad de los datos generados. Estos algoritmos adversarios ayudan al modelo a generar resultados cada vez de mayor calidad. Habitualmente, se utilizan para la generación de imágenes y vídeos.

- **Modelos de difusión:** [26] fueron introducidos en 2014 y funcionan añadiendo primero ruido a los datos de entrenamiento hasta que son aleatorios e irreconocibles, y después entrenando al algoritmo para que difumine iterativamente el ruido hasta revelar un resultado deseado. Estos modelos tardan más tiempo en entrenarse que los anteriores, pero ofrecen más control en los resultados.
- **Transformadores:** [26] aparecen por primera vez en 2017 y evolucionan el paradigma codificador-decodificador para permitir un gran paso adelante en la forma en que se entrenan los modelos fundacionales, y en la calidad y gama de contenidos que pueden producir. Estos modelos destacan en el procesamiento del lenguaje natural (PLN) y la comprensión del lenguaje natural (CLN), y pueden generar secuencias de datos más largas y con mayor precisión y calidad que otros modelos, además también pueden entrenarse o ajustarse para utilizar herramientas. Actualmente, muchas herramientasde IA generativa como ChatGPT y GPT-4, Copilot, BERT, Bard o Midjourney se basan en estos modelos [27].

3.3.3. LLMs

Los Modelos de Lenguaje de Gran Tamaño o *Large Language Models* (LLMs) son modelos de inteligencia artificial entrenados con enormes cantidades de datos textuales y diseñados para comprender, generar y manipular lenguaje humano natural. Están construidos utilizando arquitecturas de *deep learning*, concretamente la arquitectura de *transformers*, que les permite manejar secuencias de texto muy largas, modelar dependencias complejas y generar respuestas coherentes y contextualmente relevantes [8].

Los LLMs se entrenan a través del aprendizaje no supervisado o auto-supervisado, utilizando grandes textos y técnicas como el modelado de lenguaje causal o enmascarado. Una vez entrenados, pueden adaptarse a tareas específicas mediante técnicas que se explicarán en la Sección 3.3.4 como el *fine-tuning* o el aprendizaje por refuerzo con retroalimentación humana (RLHF).

A continuación, se presenta una revisión de algunos de los LLMs más destacados actualmente:

- **GPT-4 / GPT-4o (OpenAI)**: Modelos avanzados desarrollados por OpenAI, sucesores de GPT-3.5, capaces de razonar, generar texto, comprender imágenes y manejar múltiples modalidades. GPT-4o, el más reciente, incorpora capacidades multimodales con una mejora sustancial en coste y velocidad [1, 29].
- **Claude (Anthropic)**: Claude es un modelo enfocado en la seguridad, interpretabilidad y alineación con valores humanos. Está diseñado para evitar respuestas dañinas y es altamente competitivo en tareas de comprensión y generación de texto [4].
- **Gemini (Google DeepMind)**: La serie de modelos Gemini, anteriormente conocida como Bard, está optimizada para razonamiento avanzado, programación y tareas de comprensión de lenguaje. Ofrece una fuerte integración con el ecosistema Google [13].
- **LLaMA 2 (Meta)**: Modelo de código abierto desarrollado por Meta, orientado a proporcionar acceso más amplio a modelos de alto rendimiento. Ha sido entrenado con datasets cuidadosamente seleccionados para ofrecer resultados competitivos a bajo coste [2].
- **Mistral / Mixtral**: Mistral AI ha propuesto modelos ligeros y eficientes, incluyendo el modelo mixto Mixture-of-Experts (MoE), que permite un equilibrio entre capacidad computacional y rendimiento [3].

- **Command R / R+ (Cohere)**: Especializados en recuperación de información y generación aumentada por recuperación (RAG). Ofrecen un gran rendimiento en tareas específicas como chatbots empresariales y asistencia documental [11].

Estos modelos han sido entrenados con cantidades masivas de datos, algunos superando los 100 mil millones de parámetros, y representan el estado del arte en tareas como resumen automático, generación creativa, traducción, razonamiento lógico, codificación y mucho más.

3.3.4. Workflow general de la Inteligencia Artificial Generativa

Como se ve en la Figura 3.13, la IA generativa se basa en sofisticados modelos de *deep learning* y funciona en tres fases:

1. **Formación y entrenamiento**, para crear un modelo fundacional que pueda servir de base a varias aplicaciones de IA generativa.

2. **Ajuste**, para adaptar el modelo fundacional a una aplicación específica de IA generativa.

3. **Generación**, evaluación y reajuste, para evaluar el resultado de la aplicación de IA generativa y mejorar continuamente su calidad y precisión.

A continuación se detallarán con más precisión estas fases [26].

Formación y entrenamiento

La IA generativa parte de un modelo fundacional, es decir, de un modelo de *deep learning* que sirve de base para varios tipos diferentes de aplicaciones de IA generativa. Los modelos fundacionales más comunes hoy en día son los modelos de lenguaje de gran tamaño (LLM), creados para aplicaciones de generación de texto, pero también existen modelos fundacionales para la generación de imagen, vídeo y música, así como modelos fundacionales multimodales compatibles con varios tipos de generación de contenido.

Para crear un modelo fundacional, se entrena un algoritmo de *deep learning* en enormes volúmenes de datos brutos, no estructurados y sin etiquetar; por ejemplo, terabytes de datos extraídos de Internet o de alguna otra fuente de datos enorme. Durante el entrenamiento, el algoritmo realiza y evalúa millones de tareas que consisten en "rellenar los espacios en blanco", intentando predecir el siguiente elemento de una secuencia (por ejemplo, la siguiente palabra de una frase, el siguiente elemento de una imagen, el siguiente comando de una línea de código) y ajustándose continuamente para minimizar la diferencia entre sus predicciones y los datos reales (o resultado "correcto").

El resultado de este entrenamiento es una red neuronal que puede generar contenido de forma autónoma en respuesta a las entradas o instrucciones.

Este proceso de entrenamiento requiere muchos cálculos, tiempo y dinero: requiere miles de unidades de procesamiento gráfico (GPU) agrupadas y semanas de procesamiento, todo lo cual cuesta millones de dólares. Los proyectos de modelos fundacionales de código abierto, como Llama-2 de Meta, permiten a los desarrolladores de IA generativa evitar este paso y sus costes.

Ajuste

Se dice que los modelos fundacionales son generalistas, es decir, saben mucho sobre muchos tipos de contenido, pero a menudo no pueden generar resultados específicos con la precisión o fidelidad deseadas. Para ello, el modelo debe ajustarse a una tarea específica de generación de contenidos. Esto puede hacerse de varias maneras.

- La **afinación o *fine tuning*** es el proceso de adaptar un modelo previamente entrenado para tareas o casos de uso específicos. Se ha convertido en una técnica fundamental de de *deep learning*, especialmente en el proceso de entrenamiento de modelos fundacionales utilizados para la IA generativa [25]. Implica alimentar el modelo con datos etiquetados específicos de la aplicación de generación de contenidos, preguntas o instrucciones que la aplicación es probable que reciba, y las correspondientes respuestas correctas en el formato deseado. Por ejemplo, si un equipo de desarrollo está intentando crear un *chatbot* de atención al cliente, crearía cientos o miles de documentos con preguntas del servicio de atención al cliente etiquetadas y respuestas correctas, y luego alimentaría el modelo con esos documentos. Este proceso requiere mucha mano de obra.
- El **aprendizaje por refuerzo a partir de la retroalimentación humana** (RLHF), consiste en que los usuarios humanos responden a los contenidos generados con evaluaciones que el modelo puede utilizar para actualizarlo y dotarlo de mayor precisión o relevancia. A menudo, el RLHF implica que las personas "puntúen" diferentes resultados en respuesta a una misma instrucción. Pero puede ser tan sencillo como hacer que las personas escriban o respondan a un chatbot o asistente virtual, corrigiendo su resultado.

Generación, evaluación, más ajuste

En esta fase, los desarrolladores y los usuarios evalúan continuamente los resultados de sus aplicaciones de IA generativa y ajustan aún más el modelo para aumentar su precisión o relevancia.

Otra opción para mejorar el rendimiento de una aplicación de IAG es la *RAG*, o generación aumentada por recuperación [50], es un enfoque de inteligencia artificial que combina la generación de texto con la capacidad de buscar información en bases de datos especializadas. A diferencia de los modelos anteriores, que responden únicamente con base en su entrenamiento previo, RAG accede a fuentes externas, lo que garantiza respuestas más precisas, actualizadas y contextualizadas.

El modelo opera en dos etapas clave:

1. **Recuperación de información:** Emplea motores de búsqueda internos o bases de conocimiento específicas para localizar los datos más relevantes en función de la consulta.
2. **Generación de texto:** Una vez obtenida la información, el modelo elabora una respuesta en lenguaje natural, integrando los datos de manera coherente y fluida.

3.3.5. Limitaciones y probemas de la IAG

La IAG ha evolucionado mucho en un periodo muy corto de tiempo pero aún así presenta todavía ciertas limitaciones y problemas como los que se presentan a continuación [26].

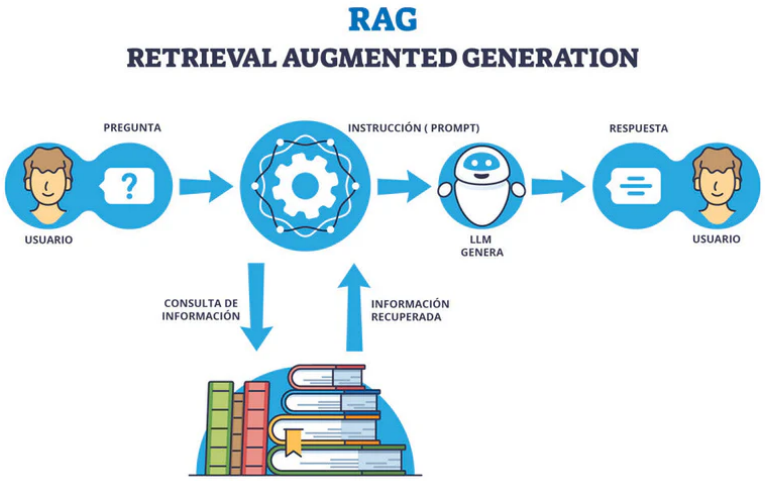

Figura 3.19: Funcionamiento de un RAG [50].

- **Alucinaciones:** Puede generar información falsa o inexacta pero convincente. Los desarrolladores están creando unas medidas preventivas denominadas barreras.
- **Falta de coherencia:** Los resultados pueden variar con la misma entrada, lo que dificulta su uso en contextos donde se requiere consistencia. La ingeniería de *prompting* ayuda a mejorar la estabilidad de las respuestas.
- **Sesgo:** Puede reflejar prejuicios de los datos de entrenamiento, lo que lleva a respuestas injustas o sesgadas. Para mitigar esto, se trabaja en mejorar la diversidad de datos y se realizan evaluaciones constantes.
- **Falta de explicabilidad:** Los modelos son "cajas negras", lo que dificulta entender cómo toman las decisiones. Se buscan técnicas de IA explicable para aumentar la transparencia y la confianza.
- **Riesgos de seguridad y privacidad:** Puede ser usada para generar contenido malicioso, como phishing o identidades falsas. .
- **Deepfakes:** La manipulación de imágenes, videos y audios con IA puede ser usada para desinformación y fraudes. Se están desarrollando herramientas para detectarlos y se fomenta la educación digital para prevenir su impacto.

3.3.6. Ingeniería de *prompting*

Un *prompt* es "una instrucción o texto inicial proporcionado a una herramienta de IA generativa para dirigir la generación de respuestas o resultados específicos. Esta entrada de información establece el contexto y la tarea que se espera que la herramienta complete, permitiendo al usuario comunicarse de manera efectiva con la IA en un lenguaje natural" [55]. La creación de instrucciones efectivas para que un modelo genere contenido es un proceso conocido como **ingeniería de**

prompting [45]. Debido a que el contenido generado por un modelo es no determinista, construir un *prompt* que genere el tipo correcto de contenido es una combinación de arte y ciencia. A continuación se presentan algunas pautas generales para hacer un buen *prompt*:

- Proporcionar instrucciones detalladas para evitar ambigüedades en las respuestas.
- Proporcionar ejemplos al modelo del tipo de entradas y el tipo de salidas que deseas para esa entrada; esta técnica se llama *few-shot learning* [45].
- Describir la tarea en términos de objetivos y resultados deseados, en lugar de proporcionar instrucciones paso a paso sobre cómo realizar la tarea.
- Ir probando pequeñas modificaciones en los *prompts* suele ser suficiente para obtener los resultados que queremos del modelo, pero también se puede llevar a cabo *fine tuning* para personalizar modelos base para un caso de uso particular.

3.4. Fundamentos técnicos

3.4.1. API de OpenAI

La API de OpenAI [42] proporciona acceso a potentes modelos de inteligencia artificial diseñados para realizar tareas como generación de texto, asistencia conversacional, análisis de contenido, transcripción de audio, generación de imágenes entre otras. En esta Sección profundizaremos más sobre esta API comenzando por comentar en la Sección 3.4.1 los modelos disponibles, continuando con los *tokens* en la Sección 3.4.1, después hablaremos sobre como utilizar *prompts* con imágenes en la Sección 3.4.1 y finalmente se explicará en la Sección 3.4.1 como se utiliza.

Gracias a esta API, se podrá integrar capacidades avanzadas de inteligencia artificial en el proyecto sin necesidad de entrenar o mantener un modelo complejo. OpenAI se encarga de toda la infraestructura y optimización del modelo, permitiendo a los usuarios centrarse en la implementación y el uso eficiente de estas herramientas.

Modelos disponibles

OpenAI ofrece diferentes modelos [39] para adaptarse a las necesidades de los distintos usuarios. Cada modelo tiene características específicas en términos de capacidad, costo y rendimiento, a continuación se comentarán algunos de los modelos más relevantes:

Para el caso particular de este proyecto los modelos que podrían ser de utilidad serían los capaces de procesar imágenes, ya que necesitamos que el modelo sea capacidad de procesar los diagramas ER y por tanto serían: OpenAI o1, OpenAI o3-mini, GPT-4o y GPT-4.5

Tokens

En la API de OpenAI, los *tokens* son las unidades básicas que el modelo utiliza para procesar y generar texto. Un token puede ser tan pequeño como un carácter o tan grande como una palabra completa. Por ejemplo, la palabra "inteligencia" podría descomponerse en varios *tokens*, mientras que una palabra corta como "es" podría ser un solo *token*. Como regla general, en inglés, un *token* equivale aproximadamente a 4 caracteres o 0,75 palabras [46].

La gestión eficiente de *tokens* es crucial, ya que los modelos de OpenAI tienen límites en la cantidad de *tokens* que pueden procesar en una sola interacción. Estos límites varían según el modelo específico que se utilice [47]. Además, el coste de uso de la API se basa en la cantidad de *tokens* procesados, por lo que es importante optimizar su uso para controlar los gastos [43].

El procesamiento de imágenes en la API de OpenAI implica un consumo de *tokens* que depende de las dimensiones y el nivel de detalle de la imagen. Para calcular el número de *tokens* que una imagen consume, se utiliza la siguiente fórmula:

$$\text{Total de } \textit{tokens} = 85 + 170 \times n$$

donde n es el número de *tiles* necesarios para cubrir la imagen. Un *tile* es una Sección de 512x512 píxeles en la que se divide la imagen para su procesamiento. Cada *tile* adicional después de la primera incrementa el consumo en 170 *tokens*. Por ejemplo, una imagen de 1024x1024 píxeles se divide en 4 *tiles*, resultando en un consumo total de *tokens* de $85 + (170 \times 4) = 765$ *tokens* [40]. Cada solicitud que se hace a la API consume *tokens*, incluyendo:

- **Tokens de entrada:** Los que forman parte del mensaje o prompt que se envía.
- **Tokens de salida:** Los generados en la respuesta de la IA.

Cada modelo tiene un límite máximo de *tokens* por solicitud que incluye tanto la entrada como la salida generada. Si el mensaje de entrada y la respuesta combinados superan este límite, la API recortará el contenido o fallará la solicitud por lo que se debe dividir las consultas largas en varias solicitudes más pequeñas.

Prompts con imágenes

Para este proyecto necesitaremos incluir imágenes en los *prompts* para poder procesar y analizar las imágenes de los diagramas ER.

La "visión" es la capacidad de un modelo de usar imágenes en los *prompts* y generar respuestas basadas en los datos que contienen. En la API de OpenAI, las imágenes pueden proporcionarse como una URL o proporcionando una imagen codificada en Base64. Además, al proporcionar una imagen puede indicarse al modelo el nivel de detalle que debe usar al procesar y comprender la imagen utilizando el parámetro *detail* en la solicitud a la API, de esta forma se pueden ahorrar *tokens* o llegar a un nivel más detallado de procesamiento dependiendo de las necesidades.

Acceso y utilización de la API de OpenAI

Para acceder a la API de OpenAI [47], en primer lugar, es necesario registrarse en la plataforma de OpenAI y obtener una clave de API, que servirá para autenticar las solicitudes y debe mantenerse segura para evitar accesos no autorizados.

Para facilitar la integración, OpenAI proporciona una biblioteca oficial en varios lenguajes de programación, como Python y JavaScript y una vez instalada, se pueden realizar solicitudes a la API mediante peticiones HTTP, generalmente de tipo POST, enviando los datos adecuados al endpoint correspondiente.

Como vemos en la Figura 3.20, el usuario envía una solicitud (request) a la API de OpenAI, incluyendo datos como el modelo y parámetros. La API procesa la solicitud y la envía a un servidor. Luego, la respuesta (response) viaja de nuevo a la API y finalmente llega al usuario con la información solicitada.

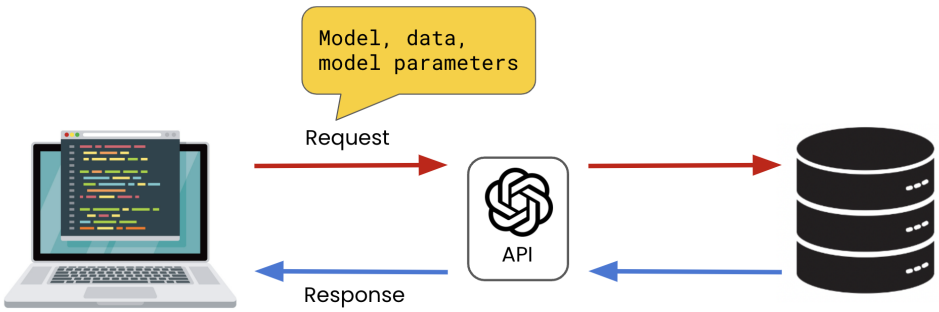

Figura 3.20: Funcionamiento de la API de OpenAI [66].

Como se ve en la Figura 3.21, es posible definir parámetros como el modelo a utilizar, el mensaje de entrada y opciones adicionales como la temperatura, que afecta la creatividad de las respuestas generadas. Las respuestas de la API contienen información estructurada que debe procesarse según las necesidades del proyecto, y es recomendable manejar posibles errores, como límites de uso o fallos en la conectividad.

3.4.2. FastAPI

FastAPI es un *framework* moderno y de alto rendimiento para la construcción de APIs web en Python. Diseñado para aprovechar las anotaciones de tipo de Python 3.8+, permite una validación automática de datos, generación de documentación interactiva y una integración sencilla con estándares como OpenAPI y JSON Schema [51].

Entre sus características destacadas se incluyen:

- **Alto rendimiento**: Comparable con *frameworks* como NodeJS y Go, gracias a su arquitectura basada en Starlette y Pydantic.
- **Desarrollo rápido**: Permite aumentar la velocidad de desarrollo en un 200 % a 300 % y reduce errores humanos en un 40 % [51].
- **Documentación automática**: Genera interfaces interactivas como Swagger UI y ReDoc sin configuración adicional [52] como se ve en la Figura 3.22.

FastAPI es especialmente interesante para este proyecto porque facilita la conexión entre el *frontend* y el *backend* de forma rápida y sencilla. Además, la documentación automática es muy cómoda y útil.

3.4.3. LangChain

LangChain es un *framework* de código abierto diseñado para facilitar la integración de modelos de lenguaje de gran tamaño (LLMs) en aplicaciones. Su objetivo principal es permitir la

```python
from openai import OpenAI

client = OpenAI()
response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {
            "role": "user",
            "content": [
                {
                    "type": "text",
                    "text": "What are in these images? Is there any difference between them?",
                },
                {
                    "type": "image_url",
                    "image_url": {
                        "url": "https://....jpg",
                    },
                },
                {
                    "type": "image_url",
                    "image_url": {
                        "url": "https://....png",
                    },
                },
            ],
        }
    ],
    max_tokens=300,
)
print(response.choices[0])
```

Figura 3.21: Ejemplo de código en Python para hacer una llamada a la API de OpenAI con imágenes [41].

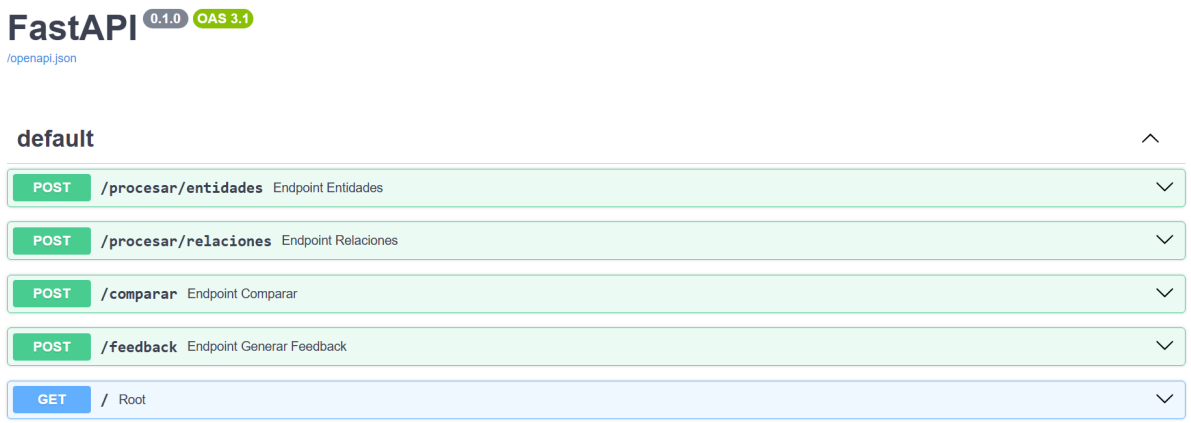

Figura 3.22: Interfaz de documentación generada automáticamente por FastAPI.

construcción de aplicaciones contextualmente conscientes que puedan razonar y tomar decisiones basadas en datos y herramientas externas [9].

Sus características principales incluyen:

- **Composición modular**: Permite combinar múltiples componentes como LLMs, herramientas externas y fuentes de datos en cadenas lógicas.
- **Integración con múltiples proveedores**: Soporta modelos de OpenAI, Anthropic, Google, entre otros.
- **Soporte para RAG (Retrieval-Augmented Generation)**: Facilita la implementación de aplicaciones que combinan recuperación de información y generación de texto [32].
- **Capacidad de citación**: Permite que las respuestas generadas incluyan referencias a las fuentes de información utilizadas [31].

Nombre del modelo	**Descripción general**	**Caso de uso ideal**	**Longitud de contexto**
OpenAI o1	Modelo de razonamiento avanzado que admite herramientas, salidas estructuradas y procesamiento de imágenes.	Problemas complejos y de múltiples pasos.	200k *tokens*
OpenAI o3-mini	Modelo de razonamiento compacto y rentable, optimizado para tareas de codificación, matemáticas y ciencia. Soporta herramientas y salidas estructuradas.	Tareas de codificación, matemáticas y ciencia.	200k *tokens*
GPT-4.5	El modelo GPT más grande, diseñado para tareas creativas y planificación agente, actualmente en vista previa de investigación.	Tareas creativas y planificación agente.	128k *tokens*
GPT-4o	Modelo de alta inteligencia para tareas complejas.	Tareas complejas.	128k *tokens*
GPT-4o mini	Modelo pequeño y asequible para tareas rápidas y cotidianas.	Tareas rápidas y cotidianas.	128k *tokens*
GPT-3.5	Opción más asequible, ideal para tareas que no requieren la máxima potencia (solo admite texto).	Tareas de baja exigencia en potencia.	–
Whisper	Modelo de transcripción de audio a texto.	Convertir grabaciones en texto con alta precisión.	–
DALL·E	Modelo de generación de imágenes basado en descripciones textuales.	Generación de imágenes a partir de descripciones textuales.	–
GPT-4.1	Modelo sucesor de GPT-4o con ventana de contexto ultra-larga, mejoras en codificación e instrucción, soporte multimodal.	Procesamiento de grandes códigos, documentos legales o bases de datos extensas; tareas complejas que requieren seguimiento preciso de instrucciones.	1.000.000 *tokens*
OpenAI o3	Modelo de razonamiento profundo con "cadena de pensamiento privada", acceso autónomo a herramientas (web, código, imágenes), sobresaliente en matemáticas, ciencia y programación.	Análisis de problemas avanzados, investigación técnica profunda, resolución paso a paso con uso de herramientas integradas.	1.000.000 *tokens*

Tabla 3.4: Comparativa de modelos

Parte II

Desarrollo de la solución

Capítulo 4

Análisis

El análisis de requisitos es una etapa esencial del desarrollo de software, cuyo objetivo es identificar y documentar las necesidades de los usuarios y otros *stakeholders*. A partir de esta información, se definen especificaciones que guían el diseño y la implementación del sistema [59].

Este capítulo se organiza en cinco secciones: identificación de actores del sistema 4.1, definición de requisitos de usuario 4.2 y funcionales 4.3, especificación de los requisitos de información 4.4, y descripción de los requisitos no funcionales 4.5.

4.1. Actores

Los actores son los diferentes tipos de usuario que interactuarán con la aplicación, en este caso, el único actor sería el alumno. El alumno representaría a un estudiante de una asignatura de bases de datos cuyo objetivo es aprender a realizar diagramas ER correctamente.

4.2. Requisitos de usuario

Para identificar y describir los requisitos de usuario, se ha optado por utilizar la técnica de casos de uso. Esta metodología permite modelar de forma estructurada las interacciones entre el sistema y los actores involucrados.

En la Figura 4.1 se muestra el diagrama de casos de uso del sistema *LearnER*, donde se identifica como único actor al estudiante como se comentó en la sección anterior. Este puede realizar una acción principal que es evaluar un modelo ER.

A continuación, se muestran las especificaciones de los diferentes casos de uso identificados en el sistema *LearnER*, representados previamente en el diagrama de la Figura 4.1. La Tabla 4.1 contiene la especificación del caso de uso CU-01 “Evaluar modelo”; la Tabla 4.2 detalla el caso de uso CU-02 “Validar entidades”; la Tabla 4.3 recoge el caso de uso CU-03 “Añadir entidad”; la Tabla 4.4 presenta el caso de uso CU-04 “Modificar entidad”; la Tabla 4.5 documenta el caso de uso CU-05 “Eliminar entidad”; la Tabla 4.6 describe el caso de uso CU-06 “Validar relaciones”; la Tabla 4.7 muestra el caso de uso CU-07 “Añadir relación”; la Tabla 4.8 detalla el caso de uso CU-08 “Modificar relación”; la Tabla 4.9 describe el caso de uso CU-09 “Eliminar relación” y finalmente, la Tabla 4.10 describe el caso de uso CU-010 “Visualizar *feedback*”.

El diagrama de secuencia de la Figura 4.2 representa la interacción entre el alumno (como único actor de la aplicación) y los componentes principales de la aplicación en el contexto del

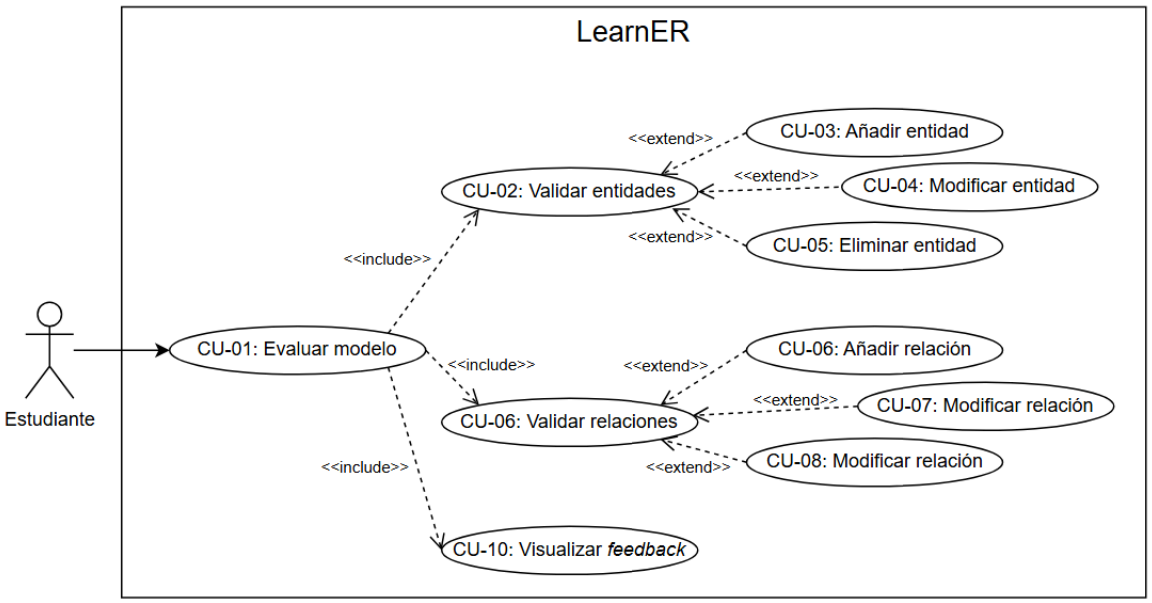

Figura 4.1: Diagrama de casos de uso.

caso de uso CU-01: Evaluar modelo. Un diagrama de secuencia es un tipo de diagrama UML que describe el flujo de mensajes intercambiados entre los distintos participantes de un sistema a lo largo del tiempo, permitiendo visualizar el orden de las operaciones y la colaboración necesaria para completar una funcionalidad concreta.

En este caso, el diagrama ilustra cómo el estudiante selecciona un ejercicio, sube su modelo entidad-relación, la aplicación procesa la imagen, permite la validación manual de entidades y relaciones, y finalmente evalúa el modelo (todo esto con ayuda de la API de OpenAI), mostrando al estudiante el *feedback* correspondiente.

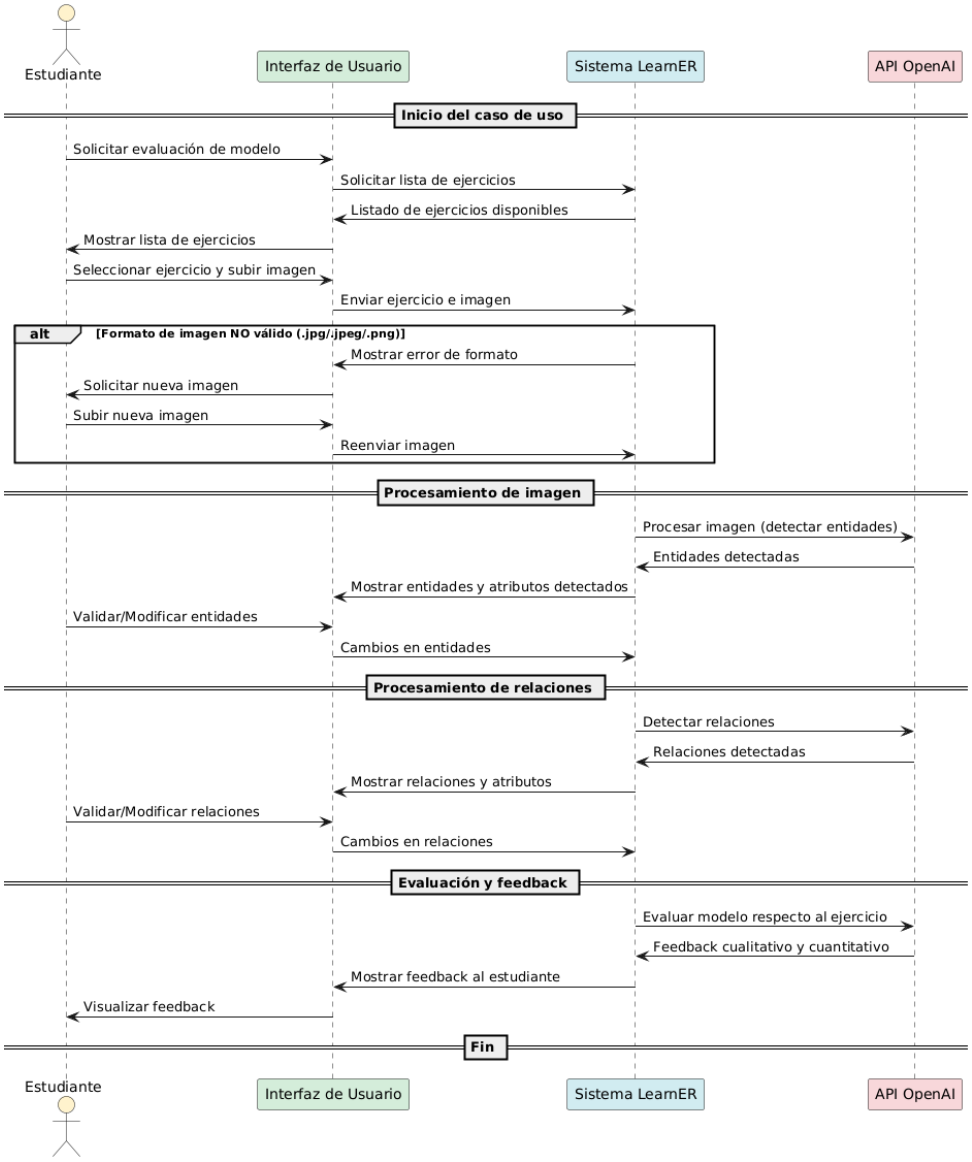

Figura 4.2: Diagrama de secuencia del CU-01: Evaluar modelo.

CU-01: Evaluar modelo		
Descripción	Describe el proceso mediante el cual el estudiante solicita la evaluación de un modelo ER, selecciona un supuesto práctico, sube la imagen de su diagrama y el sistema procesa la imagen, valida el modelo y muestra el *feedback*.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita evaluar un modelo ER.
	Paso 2	El sistema presenta los supuestos prácticos para los que se puede solicitar la evaluación.
	Paso 3	El estudiante elige el supuesto deseado y proporciona la imagen que contiene el modelo entidad-relación.
	Paso 4	El sistema procesa la imagen y muestra las entidades identificadas con sus atributos correspondientes.
	Paso 5	Se realiza el caso de uso CU-02. Validar entidades.
	Paso 6	El sistema procesa la imagen y muestra las relaciones identificadas con sus multiplicidades y atributos correspondientes.
	Paso 7	Se realiza el caso de uso CU-03. Validar relaciones.
	Paso 8	El sistema almacena el modelo obtenido a partir de la imagen procesada.
	Paso 9	El sistema evalúa el modelo proporcionado respecto a la solución del supuesto práctico.
	Paso 10	El sistema almacena el resultado de la evaluación.
	Paso 11	Se realiza el caso de uso CU-04. Visualizar *feedback*.
	Paso 12	El caso de uso finaliza satisfactoriamente.
Excepciones	E 1	El caso de uso vuelve al paso 3 si el formato de la imagen no es .jpg, .jpeg o .png.

Tabla 4.1: Especificación del caso de uso Evaluar modelo.

CU-02:Validar entidades		
Descripción	Este caso de uso permite al estudiante validar las entidades generadas por el sistema.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El sistema muestra las entidades identificadas en la imagen.
	Paso 2.a	Si el estudiante solicita añadir una nueva entidad, se realiza el caso de uso CU-03 Añadir entidad.
	Paso 2.b	Si el estudiante solicita modificar una entidad existente, se realiza el caso de uso CU-04 Modificar entidad.
	Paso 2.c	Si el estudiante solicita eliminar una entidad existente, se realiza el caso de uso CU-05 Eliminar entidad.
	Paso 3	El caso de uso finaliza cuando el estudiante confirma que todas las entidades han sido validadas.

Tabla 4.2: Especificación del caso de uso Validar entidades

CU-03: Añadir entidad		
Descripción	Permite al estudiante crear una nueva entidad y sus atributos dentro del modelo ER.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita añadir una nueva entidad.
	Paso 2	El sistema presenta los campos necesarios para describir la entidad (nombre, atributos, clave primaria, etc.).
	Paso 3	El sistema almacena la entidad creada.
	Paso 4	El caso de uso finaliza satisfactoriamente.

Tabla 4.3: Especificación del caso de uso Añadir entidad

CU-04: Modificar entidad		
Descripción	Permite al estudiante actualizar los datos de una entidad existente.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita modificar una entidad existente.
	Paso 2	El sistema presenta los campos actuales de la entidad para que el estudiante los edite.
	Paso 3	El sistema almacena los cambios realizados.
	Paso 4	El caso de uso finaliza satisfactoriamente.

Tabla 4.4: Especificación del caso de uso Modificar entidad

CU-05: Eliminar entidad		
Descripción	Permite al estudiante borrar una entidad del modelo ER.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita eliminar una entidad existente.
	Paso 2	El sistema presenta los datos de la entidad para confirmación.
	Paso 3	El sistema elimina la entidad seleccionada.
	Paso 4	El caso de uso finaliza satisfactoriamente.

Tabla 4.5: Especificación del caso de uso Eliminar entidad

CU-06: Validar relaciones		
Descripción	Este caso de uso permite al estudiante validar las relaciones identificadas en la imagen, así como añadir, modificar o eliminar relaciones según sea necesario.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El sistema muestra las relaciones identificadas en la imagen.
	Paso 2.a	Si el estudiante solicita añadir una nueva relación, se realiza el caso de uso CU-07 Añadir relación.
	Paso 2.b	Si el estudiante solicita modificar una relación existente, se realiza el caso de uso CU-08 Modificar relación.
	Paso 2.c	Si el estudiante solicita eliminar una relación existente, se realiza el caso de uso CU-09 Eliminar relación.
	Paso 3	El caso de uso finaliza cuando el estudiante confirma que todas las relaciones han sido validadas.

Tabla 4.6: Especificación del caso de uso Validar relaciones

CU-06: Añadir relación		
Descripción	Permite al estudiante crear una nueva relación entre entidades, incluyendo sus multiplicidades y atributos.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita añadir una nueva relación.
	Paso 2	El sistema presenta los campos necesarios para describir la relación (entidades participantes, cardinalidades, atributos, etc.).
	Paso 3	El sistema almacena la relación creada.
	Paso 4	El caso de uso finaliza satisfactoriamente.

Tabla 4.7: Especificación del caso de uso Añadir relación

CU-07: Modificar relación		
Descripción	Permite al estudiante actualizar los datos de una relación existente.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita modificar una relación existente.
	Paso 2	El sistema presenta los campos actuales de la relación para que el estudiante los edite.
	Paso 3	El sistema almacena los cambios realizados.
	Paso 4	El caso de uso finaliza satisfactoriamente.

Tabla 4.8: Especificación del caso de uso Modificar relación

CU-08: Eliminar relación		
Descripción	Permite al estudiante borrar una relación del modelo ER.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita eliminar una relación existente.
	Paso 2	El sistema presenta los datos de la relación para confirmación.
	Paso 3	El sistema elimina la relación seleccionada.
	Paso 4	El caso de uso finaliza satisfactoriamente.

Tabla 4.9: Especificación del caso de uso Eliminar relación

CU-10: Visualizar *feedback*		
Descripción	Describe el proceso mediante el cual el estudiante visualiza el *feedback* generado.	
Actor	Estudiante.	
Secuencia normal	Paso 1	El estudiante solicita visualizar el *feedback* generado.
	Paso 2	El sistema obtiene los resultados de la evaluación y los presenta.
	Paso 3	El caso de uso finaliza satisfactoriamente.

Tabla 4.10: Especificación del caso de uso Visualizar *feedback*.

4.3. Requisitos funcionales

Los requisitos funcionales describen qué tendrán que implementar los desarrolladores para satisfacer los requisitos de usuario, describen el comportamiento del producto software bajo unas condiciones especificadas. En la Tabla 4.11 se muestran los requisitos funcionales de esta aplicación.

ID	Descripción
RF-01	La aplicación deberá permitir a los alumnos seleccionar el ejercicio que quieren corregir.
RF-02	La aplicación deberá permitir la subida de diagramas ER cuando el alumno solicite una corrección.
RF-03	La aplicación deberá conectar con la API de OpenAI para identificar los componentes del diagrama ER del alumno.
RF-04	La aplicación permitirá crear, modificar, actualizar y acceder a la información de las entidades.
RF-05	La aplicación permitirá crear, modificar, actualizar y acceder a la información de las relaciones.
RF-06	La aplicación permitirá crear, modificar, actualizar y acceder a la información de los atributos.
RF-07	La aplicación evaluará el modelo proporcionado por el estudiante respecto a la solución disponible, de acuerdo con los criterios de aceptación establecidos.
RF-08	La aplicación obtendrá retroalimentación contextualizada en el ámbito de negocio del supuesto práctico.

Tabla 4.11: Requisitos funcionales.

4.4. Requisitos de información

En este apartado se definen los requisitos de información necesarios para el desarrollo del sistema. Los requisitos de información permiten identificar y describir de forma estructurada los datos que debe gestionar el sistema, así como las relaciones existentes entre ellos. Para representar estos requisitos se empleará un modelo conceptual de datos (Figura 4.3), que estará compuesto por un diagrama entidad-relación (ER) y un diccionario de datos. El diagrama ER facilita una representación gráfica de las entidades, atributos y relaciones entre los datos, permitiendo una comprensión visual y global del modelo. El diccionario de datos complementa al diagrama proporcionando una descripción detallada de cada elemento del modelo, especificando su significado, estructura y restricciones.

A continuación, se presenta el diccionario de datos en las imágenes 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13, 4.14, 4.15, 4.16 y 4.17.

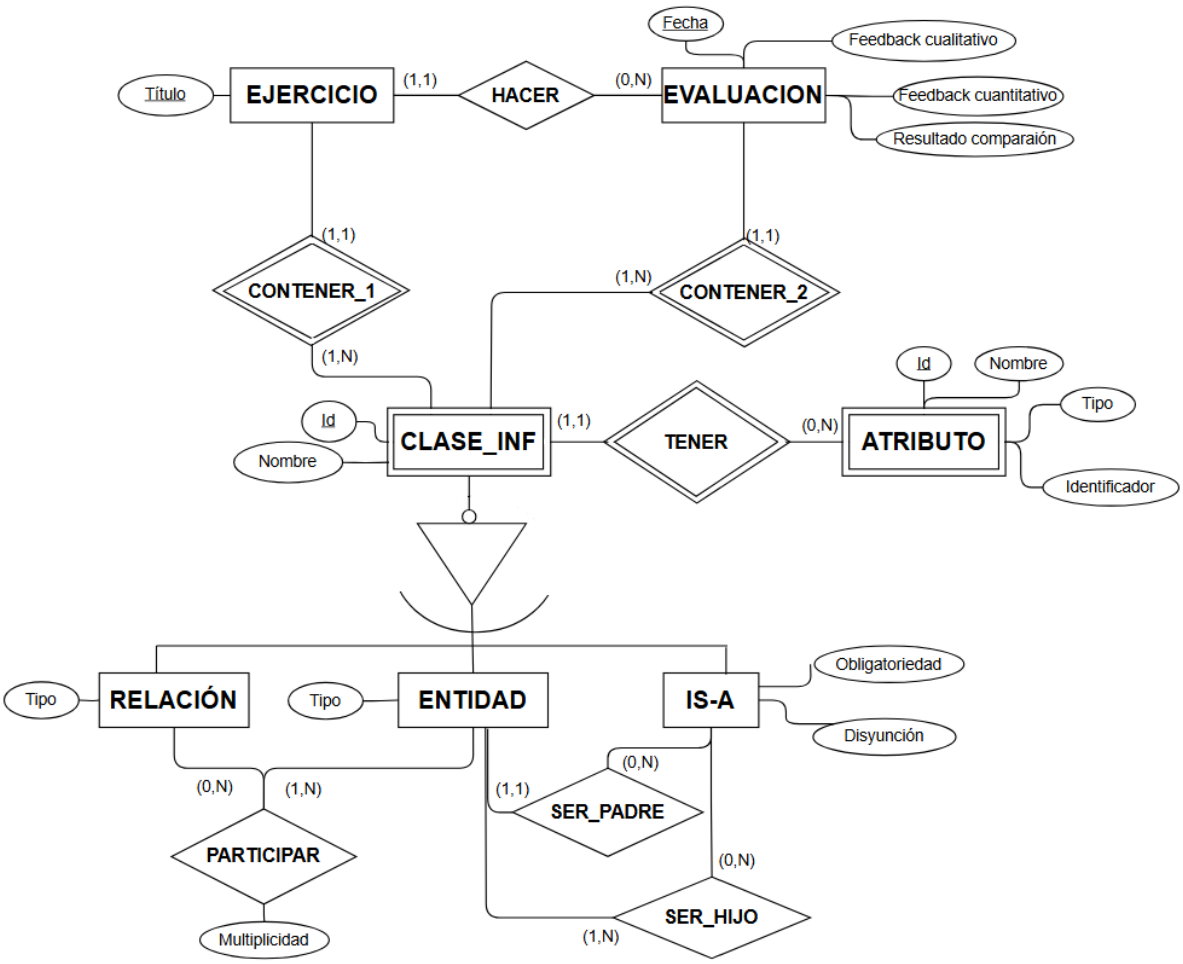

Figura 4.3: Modelo entidad-relación del sistema.

ID	ENT-01
Nombre:	EVALUACIÓN
Definición:	La entidad EVALUACIÓN representa la solución a un ejercicio subida por un estudiante.

Atributos:	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
	A01.01	Fecha	Representa la fecha exacta de realización de la entrega del ejercicio.	Timestamp	Formato: Año/Mes /Día/Hora/Min/Seg	No	No	No	Sí	No	No
	A01.02	Feedback cualitativo	Texto con el feedback cualitativo generado por la aplicación.	String	N/A	No	No	No	No	No	No
	A01.03	Feedback cuantitativo	Json con el feedback cuantitativo generado por la aplicación. Las claves son los criterios de calificación.	String	Formato json	No	No	No	No	No	No
	A01.04	Resultado comparación	CSV con la información de la comparación del json del alumno y el profesor	String	Formato CSV	No	No	No	No	No	No

Identificadores:	ID	Nombre Atributo
	A01.01	Fecha

Figura 4.4: Diccionario de datos de la entidad EVALUACIÓN.

ID	ENT-02
Nombre:	EJERCICIO
Definición:	La entidad EJERCICIO representa la solución del profesor a un ejercicio.

Atributos:	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
	A02.01	Título	Título del ejercicio.	String	N/A	No	No	No	Si	No	No

Identificadores:	ID	Nombre Atributo
	A02.01	Título

Figura 4.5: Diccionario de datos de la entidad EJERCICIO.

ID	ENT-03
Nombre:	CLASE_INF
Definición:	La entidad CLASE_INF representa los componentes de un modelo ER.
Reglas:	Hay 3 tipos de CLASE_INF: RELACIÓN, ENTIDAD e IS-A

Atributos:	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
	A03.01	Nombre	Nombre del componente.	String		No	No	No	No	No	No
	A03.02	ID	Identificador del componente	String		No	No	No	Si	No	No

Identificadores:	ID	Nombre Atributo
	A03.02	ID
	A02.01	Título
	A01.01	Fecha

Figura 4.6: Diccionario de datos de la entidad CLASE_INF.

ID	ENT-04
Nombre:	RELACIÓN
Definición:	La entidad RELACIÓN representa a las relaciones del modelo ER.

Atributos:	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
	A04.01	Tipo	Representa la categoría de la relación.	String	Puede ser "débil" o "fuerte"	No	No	No	No	No	No

Identificadores:	ID	Nombre Atributo
	A03.02	ID
	A02.01	Título
	A01.01	Fecha

Figura 4.7: Diccionario de datos de la entidad RELACIÓN.

ID	**ENT-05**
Nombre:	**ENTIDAD**
Definición:	La entidad ENTIDAD representa a las entidades del modelo ER.

	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
Atributos:	A05.01	Tipo	Representa la categoría de la entidad.	String	Puede ser "débil" o "fuerte"	No	No	No	No	No	No

	ID	Nombre Atributo
Identificadores:	A03.02	ID
	A02.01	Título
	A01.01	Fecha

Figura 4.8: Diccionario de datos de la entidad ENTIDAD.

ID	**ENT-06**
Nombre:	**IS-A**
Definición:	La entidad IS-A representa a las relaciones de tipo IS-A del modelo

	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
Atributos:	A06.01	Obligatoridad	Indica si las instancias de la superclase tienen que pertenecer a alguna subclase.	Boolean		No	No	No	No	No	No
	A06.02	Disjunción	Indica si las instancias de las superclases pueden pertenecer a más de una subclase.	Boolean		No	No	No	No	No	No

	ID	Nombre Atributo
Identificadores:	A03.02	ID
	A02.01	Título
	A01.01	Fecha

Figura 4.9: Diccionario de datos de la entidad IS-A.

ID	**ENT-07**
Nombre:	**ATRIBUTO**
Definición:	La entidad ATRIBUTO representa los atributos de un modelo ER.

	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
Atributos:	A07.01	Nombre	Nombre del atributo.	String		No	No	No	No	No	No
	A07.02	Identificador	Indica si el atributo es un identificador del componente al que pertenece	Boolean		No	No	No	No	No	No
	A07.03	Tipo	Representa la categoría del atributo.	String	Puede ser "compuesto", "derivado", "multivaluado" o "simple"	No	No	No	No	No	No
	A07.04	ID	Identificador del atributo.	String		No	No	No	Si	No	No

	ID	Nombre Atributo
Identificadores:	A07.04	ID
	A02.01	Título
	A01.01	Fecha

Figura 4.10: Diccionario de datos de la entidad ATRIBUTO.

ID	**ENT-01**
Nombre:	**EVALUACIÓN**
Definición:	La entidad EVALUACIÓN representa la solución a un ejercicio subida por un estudiante.

	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
Atributos:	A01.01	Fecha	Representa la fecha exacta de realización de la entrega del ejercicio.	Timestamp	Formato: Año/Mes /Día/Hora/Min/Seg	No	No	No	Sí	No	No
	A01.02	Feedback cualitativo	Texto con el feedback cualitativo generado por la aplicación.	String	N/A	No	No	No	No	No	No
	A01.03	Feedback cuantitativo	Json con el feedback cuantitativo generado por la aplicación. Las claves son los criterios de calificación.	String	Formato json	No	No	No	No	No	No
	A01.04	Resultado comparación	CSV con la información de la comparación del json del alumno y el profesor	String	Formato CSV	No	No	No	No	No	No

	ID	Nombre Atributo
Identificadores:	A01.01	Fecha

Figura 4.11: Diccionario de datos de la relación HACER.

ID	**REL_02**
Nombre:	**CONTENER_1**
Definición:	Un ejercicio describe una o má clase_inf y una clase_inf está descrita en un ejercicio.

	ID	Nombre Entidad	Participación	Cardinalidad
Entidades	ENT-02	EJERCICIO	1	N
	ENT-03	CLASE_INF	1	1

Figura 4.12: Diccionario de datos de la relación CONTENER_1.

ID	**REL_03**
Nombre:	**CONTENER_2**
Definición:	Una evaluación describe una o más clase_inf y una clase_inf puede estar descrita en una o más evaluaciones.

	ID	Nombre Entidad	Participación	Cardinalidad
Entidades	ENT-01	EVALUACIÓN	1	N
	ENT-03	CLASE_INF	1	N

Figura 4.13: Diccionario de datos de la relación CONTENER_2.

ID	**REL_03**
Nombre:	**PARTICIPAR**
Definición:	Una relación participa en una o más entidades, una entidad puede participar en ninguna, una o más relaciones.

	ID	Nombre Entidad	Participación	Cardinalidad
Entidades	ENT-07	RELACIÓN	1	N
	ENT-08	ENTIDAD	0	N

	ID	Nombre	Definición	Tipo	Reglas	Derivado	Multivaluado	Compuesto	Único	Inicial	Nulo
Atributos:	E03.01	Multiplicidad	Valor de la multiplicidad de la relación.	String		No	No	No	No	No	No

Figura 4.14: Diccionario de datos de la relación PARTICIPAR.

ID	**REL_04**
Nombre:	**SER_PADRE**
Definición:	Una IS-A puede ser clase padre de una o varias entidades y una entidad puede ser padre de ninguna, una o más IS-A.

	ID	Nombre Entidad	Participación	Cardinalidad
Entidades	ENT-05	ENTIDAD	0	N
	ENT-06	IS-A	1	1

Figura 4.15: Diccionario de datos de la relación SER_PADRE.

ID	**REL_05**
Nombre:	**SER_HIJA**
Definición:	Una IS-A puede ser clase hija de una o varias entidades y una entidad puede ser hija de ninguna, una o más IS-A.

	ID	Nombre Entidad	Participación	Cardinalidad
Entidades	ENT-05	ENTIDAD	0	N
	ENT-06	IS-A	1	N

Figura 4.16: Diccionario de datos de la relación SER_HIJA.

ID	**REL_09**
Nombre:	**TENER**
Definición:	Una clase_inf puede poseer uno o más atributos y un atributo puede ser poseido por una clase_inf.

	ID	Nombre Entidad	Participación	Cardinalidad
Entidades	ENT-03	CLASE_INF	1	N
	ENT-07	ATRIBUTO	1	1

Figura 4.17: Diccionario de datos de la relación TENER.

4.5. Requisitos no funcionales

Los requisitos no funcionales de un sistema informático describen características técnicas del sistema necesarias para su correcto funcionamiento. En la Tabla 4.12 se presentan los requisitos no funcionales de esta aplicación.

ID	**Descripción**
RNF-01	La aplicación deberá soportar imágenes en formato JPG, JPEG y PNG.

Tabla 4.12: Requisitos no funcionales

Capítulo 5

Diseño

En este capítulo se presentará en detalle el diseño del sistema, abordando los distintos aspectos que conforman su estructura y funcionamiento. En primer lugar, se presenta la arquitectura física en la Sección 5.1, a continuación, en la Sección 5.2, se desarrolla la arquitectura lógica, después el modelo lógico de datos en la Sección 5.3 y finalmente se trata el diseño de la interfaz de usuario en la Sección 5.4.

5.1. Arquitectura física

La arquitectura física muestra los componentes físicos de la aplicación a desarrollar y las relaciones entre ellos. En este caso se ha optado por utilizar una arquitectura física de dos capas como se aprecia en la Figura 5.1. La primera de ellas será la capa cliente que incluye los dispositivos que utilizan los usuarios para acceder a la aplicación y la segunda capa es la capa de aplicación que contiene el servidor que ejecuta la lógica de la aplicación.

En la Figura 5.2 se presenta el diagrama de despliegue del sistema, en él se detallan los principales dispositivos y entornos de ejecución involucrados, así como la interacción entre los diferentes componentes del sistema, desde el cliente hasta la integración con servicios externos.

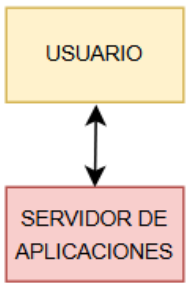

Figura 5.1: Diagrama de arquitectura física del sistema.

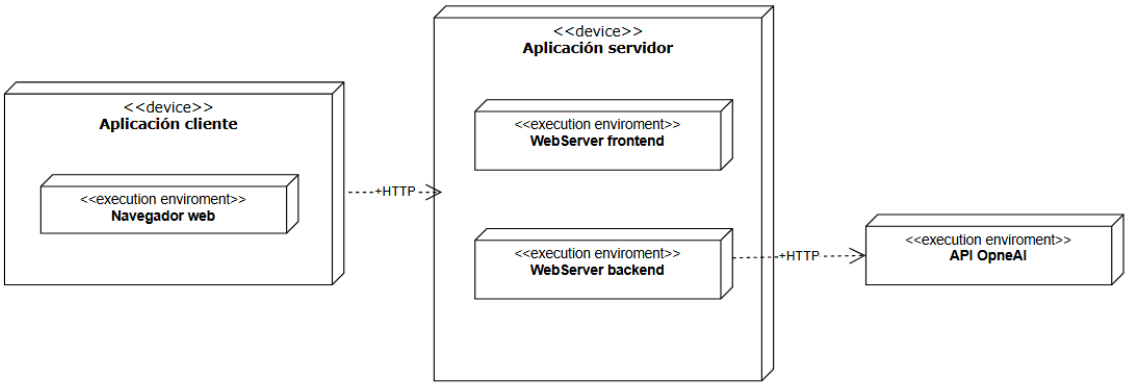

Figura 5.2: Diagrama de despliegue del sistema.

5.2. Arquitectura lógica

La arquitectura lógica de la aplicación sigue el patrón Modelo-Vista-Controlador (MVC) [15], una estructura ampliamente utilizada que facilita la organización del software al dividirlo en tres componentes principales:

- **Modelo**: es la parte de la aplicación que gestiona los datos, la lógica de negocio y el funcionamiento del sistema.
- **Vista**: representa la interfaz de usuario. Su función es mostrar los datos al usuario y recoger sus acciones o entradas.
- **Controlador**: actúa como intermediario entre la vista y el modelo. Recibe las acciones del usuario desde la vista, interpreta esas acciones y coordina las respuestas adecuadas.

En el diagrama de arquitectura lógica de la aplicación que se presenta en la Figura 5.3, esta estructura se refleja de la siguiente manera:

- La **capa de presentación** corresponde a la vista. Está formada por las distintas páginas de la aplicación, como la página de inicio, la de subida de archivos, la de entidades, la de relaciones y la de *feedback*. Todo lo que el usuario visualiza y manipula pertenece a esta capa.
- El **orquestador**, ubicado en el centro de la capa de negocio, es el controlador del sistema. Este componente, implementado con FastAPI, se encarga de recibir las solicitudes procedentes de la interfaz de usuario, procesarlas y coordinar la ejecución de las operaciones necesarias en la lógica de negocio.
- La **capa de negocio** excluyendo el orquestador y el almacenamiento representan el modelo. Aquí se encuentran los servicios responsables de procesar la información: la generación de entidades y relaciones, la comparación de resultados, la generación de *feedback* y el cálculo

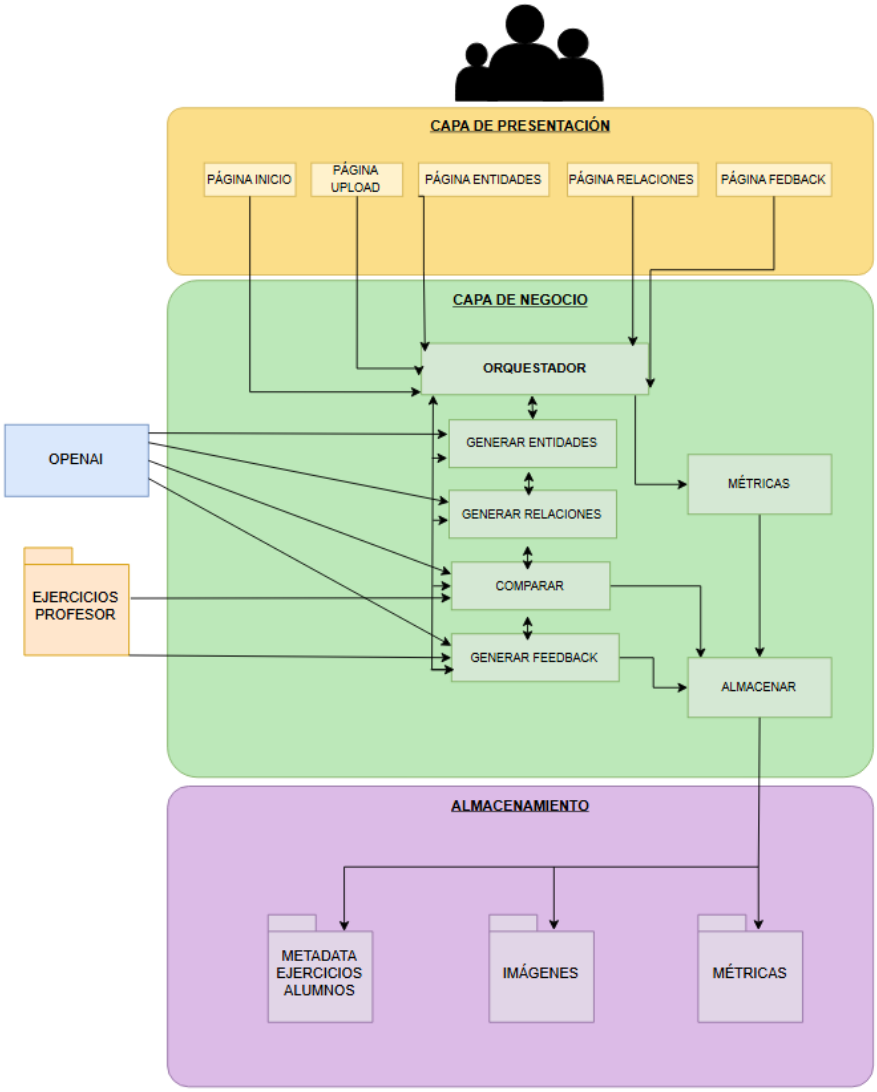

Figura 5.3: Diagrama de la arquitectura lógica del sistema.

de métricas. A excepción del calculo de métricas, todas estas tareas incluyen a la API de OpenAI. Además, para las tareas de comparar y generar *feedback* se extrae información de una carpeta local en la que se encuentra la solución de cada ejercicio del profesor y el enunciado.

5.3. Modelo lógico de datos

A continuación se presenta el modelo lógico de datos que incluye los esquemas lógicos utilizados para darle soporte a los diferentes conjuntos de datos que se crearán y manipularán en la aplicación.

En primer lugar, se comenta la forma en la que se persistirán los componentes identificados en el modelo ER. Como se ve en la Figura 5.4, el modelo se guarda en un documento JSON,

```
[ {json_entidades},
  {json_relaciones}
]
```

Figura 5.4: Estructura de datos del modelo ER.

```
{
  "id": id_entidad,
  "name": nombre_entidad,
  "type": tipo_entidad,
  "attributes": [lista_atributos]
}
```

Figura 5.5: Estructura de datos de una entidad.

con dos elementos que almacenan, respectivamente, la información correspondiente a entidades y relaciones.

Cada entidad se describe como un nuevo elemento en el que se registra su id, el nombre, tipo y una lista con los atributos, que los caracterizan, como se ilustra en la Figura 5.5. Como vemos en la Figura 5.6, de cada atributo se registra su id, nombre, el tipo y un valor booleano que indica si el atributo es identificador o no.

Por otra parte, de cada relación se almacena su información relevante incluyendo el id, el nombre, el tipo, una lista con los participantes y una lista con los atributos, como se ve en la Figura 5.7. En el caso de las relaciones IS-A se guardan de la misma forma con la diferencia de que el nombre de la relación es IS-A. Finalmente, cabe destacar que los atributos se almacenan de forma similar a cómo se ha indicado para el caso de las entidades y para cada participante, se guarda el nombre de la entidad y la multiplicidad que le corresponde, como se ve en la Figura 5.8.

En segundo lugar, se detalla el formato en el que se guarda el resultado de comparar el

```
{
  "id": id_atributo,
  "name": nombre_atributo,
  "type": tipo_atributo,
  "PK": true/false
}
```

Figura 5.6: Estructura de datos de un atributo.

```
{
  "id": id_relacion,
  "name": nombre_relacion,
  "type": tipo_relacion,
  "attributes": [lista_atributos],
  "participants": [lista_participantes]
}
```

Figura 5.7: Estructura de datos de una relación.

```
{
  "entity": nombre_entidad,
  "multiplicity": multiplicidad
}
```

Figura 5.8: Estructura de datos de un participante.

modelo del profesor con el modelo del alumno. Para guardar esta "comparación" se utilizará un archivo CSV en el que tendremos una fila por cada entidad, relación o IS-A que el profesor tenga en su modelo. La estructura de las filas correspondientes a una entidad se observa en la Figura 5.9. La primera columna contiene el identificador de la entidad según el modelo del profesor. La segunda columna indica si la entidad es fuerte o débil, utilizando el valor 0 para entidades fuertes y 1 para entidades débiles. A continuación, la tercera columna refleja si la entidad del alumno representa correctamente la del profesor, incluso si los nombres difieren, mediante un valor 1 para la coincidencia conceptual y 0 en caso contrario. La cuarta columna indica, también con valores 1 o 0, si el tipo de la entidad (fuerte o débil) está correctamente modelado por el alumno. En la quinta columna se señala si el identificador de la entidad coincide conceptualmente en ambos modelos, nuevamente empleando 1 para indicar coincidencia y 0 para discrepancia. Las dos últimas columnas recogen, respectivamente, el número de atributos correctamente modelados en la entidad del alumno respecto al profesor, y el número de atributos incorrectos o ausentes en el modelo del alumno.

En cuanto a las filas que representan relaciones (excluyendo las de tipo IS-A), se observa en la Figura 5.10. La primera columna es el identificador de la relación según el modelo del profesor. La segunda columna recoge si la relación está presente y correctamente modelada por el alumno, con independencia de la coincidencia exacta del nombre, utilizando el valor 1 si la relación aparece y 0 si no. La tercera columna indica si el tipo de relación (por ejemplo, binaria, ternaria, etc.) está correctamente representado, también con valores 1 o 0. A continuación, la cuarta y la quinta columna contienen, respectivamente, el número de atributos correctamente

```
ID;esDebil;correccionNombre;correccionTipo;correccionPK;númeroAtributosCorrectos;númeroAtributosIncorrectos
```

Figura 5.9: Estructura de las líneas correspondientes a una entidad en el CSV de comparación de modelos.

```
ID;correccionNombre;correccionTipo;númeroAtributosCorrectos;númeroAtributosIncorrectos;correccionPrimeraMultiplicidad;correccionSegundaMultiplicidad
```

Figura 5.10: Estructura de las líneas correspondientes a una relación en el CSV de comparación de modelos.

```
ID;correcciónAparece;correcciónTipo
```

Figura 5.11: Estructura de las líneas correspondientes a una IS-A en el CSV de comparación de modelos.

modelados en la relación del alumno y el número de atributos incorrectos o ausentes. Por último, las dos columnas finales reflejan si las multiplicidades de los participantes en la relación están correctamente modeladas, siendo 1 si la multiplicidad coincide y 0 si existe algún error, tanto para el primer como para el segundo participante.

Para las filas que corresponden a relaciones de tipo IS-A se observa en la Figura 5.11. La primera columna recoge el identificador de la relación IS-A en el modelo del profesor. La segunda columna indica si esta relación IS-A aparece correctamente representada en el modelo del alumno, con un valor de 1 en caso afirmativo y 0 en caso contrario. La tercera columna señala si el tipo de generalización o especialización (por ejemplo, si es exclusiva o inclusiva, total o parcial) ha sido correctamente modelado, empleando nuevamente 1 o 0 según corresponda.

En todos los casos, las filas del archivo reflejan únicamente los elementos que aparecen en el modelo del profesor, y los valores 1 y 0 se emplean sistemáticamente para indicar si cada aspecto está correctamente representado o no en el modelo del alumno.

Finalmente, como se ve en la Figura 5.12 se tienen cuatro últimas filas que se utilizan para contar el número de entidades fuertes, entidades débiles, relaciones y relaciones IS-A respectivamente que el alumno ha representado en su modelo y son erróneas, es decir, no corresponden con ningún componente del modelo del profesor.

```
EntidadesFuertesMal;numEntidadesFuertesMal

EntidadesDebilesMal;numEntidadesDebilesMal

RelacionesMal;numRelacionesMal

ISAMal;numISAMal
```

Figura 5.12: Estructura de las líneas que cuentan el número de elementos incorrectos en el modelo del alumno.

5.4. Interfaz de usuario

Antes de comenzar con el desarrollo *frontend* de la aplicación se hicieron bocetos de cada página y componente importante para poder visualizar el diseño de la interfaz de usuario deseado.

A continuación, se presenta una breve descripción de cada uno de los bocetos realizados:

- **Selección de ejercicio (Figura 5.13):** Muestra la página donde el usuario puede seleccionar el ejercicio que desea corregir. Este boceto es útil para definir la navegación inicial de la aplicación y cómo se presentan las distintas opciones de ejercicios disponibles.
- **Subida de ejercicio (Figura 5.14):** Representa la interfaz en la que el usuario sube su ejercicio.
- **Edición de entidades (Figura 5.15):** Ilustra la página donde se editan las entidades del ejercicio. El objetivo es mostrar cómo se podrán visualizar, añadir, eliminar y modificar las entidades detectadas, lo que resulta esencial para la corrección y mejora del ejercicio por parte del usuario.
- **Desplegable de entidades (Figura 5.16):** Este boceto muestra el desplegable que se abre al seleccionar una entidad en la página de entidades. Aquí se detalla la información adicional y las acciones disponibles para cada entidad, facilitando su gestión individual.
- **Edición de relaciones (Figura 5.17):** Representa la página para editar tanto las relaciones normales como las relaciones IS-A entre entidades. Esta interfaz es importante para definir cómo el usuario puede visualizar, añadir, eliminar y modificar las relaciones (tanto normales, como IS-A).
- **Desplegable de relaciones normales (Figura 5.18):** Muestra el desplegable que aparece al seleccionar una relación normal (que no es IS-A) en la página de relaciones. Permite acceder y editar información específica sobre la relación seleccionada.
- **Desplegable de relaciones IS-A (Figura 5.19):** Similar al anterior, pero en este caso enfocado en las relaciones de tipo IS-A. Aquí se puede consultar y modificar información específica de estas relaciones jerárquicas.
- **Página de *feedback* (Figura 5.20):** Boceto de la página donde se mostrará al usuario tanto el *feedback* cualitativo como el cuantitativo respecto a su ejercicio. Esta sección es clave para que el usuario reciba información útil sobre su desempeño y áreas de mejora.

Figura 5.13: Boceto de la página en la que se selecciona el ejercicio que se quiere corregir.

Figura 5.14: Boceto de la página en la que el usuario sube su ejercicio.

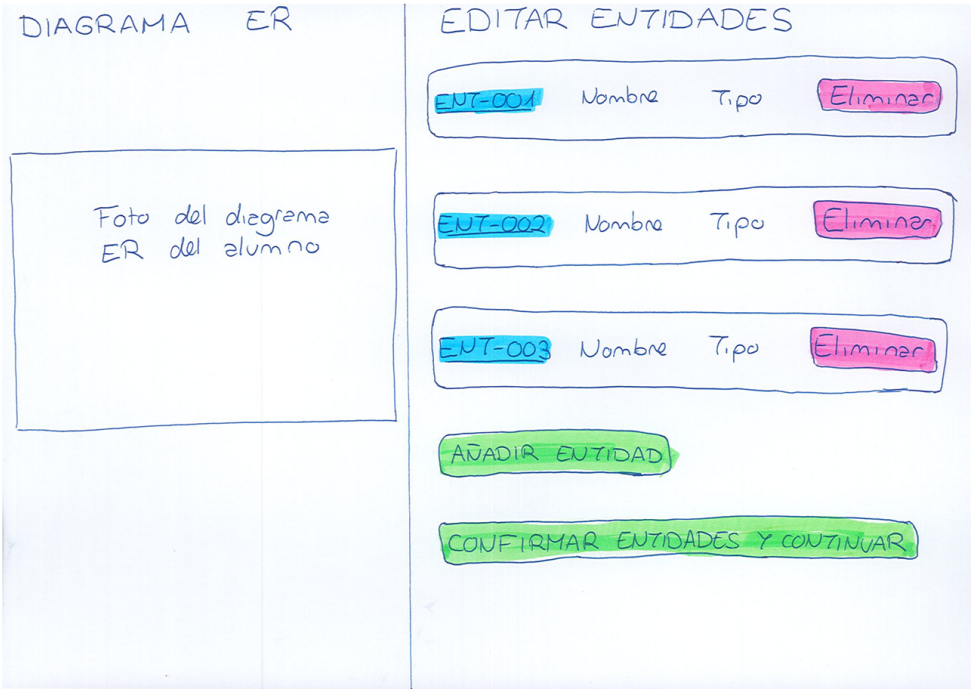

Figura 5.15: Boceto de la página en la que se editan las entidades.

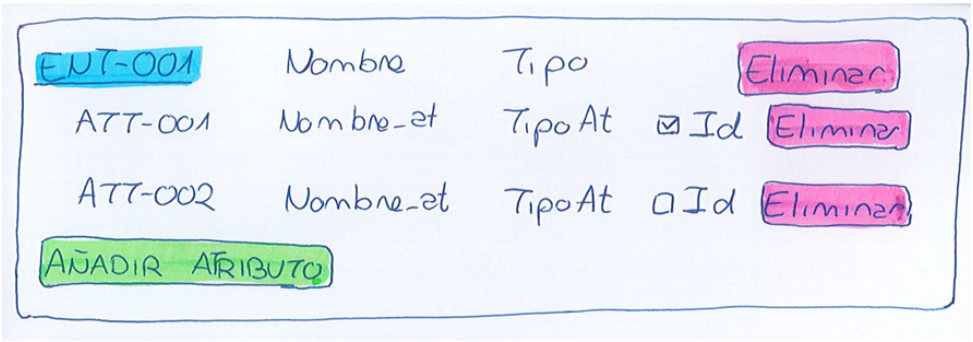

Figura 5.16: Boceto del desplegable de la página de entidades que se abre al pinchar en una entidad.

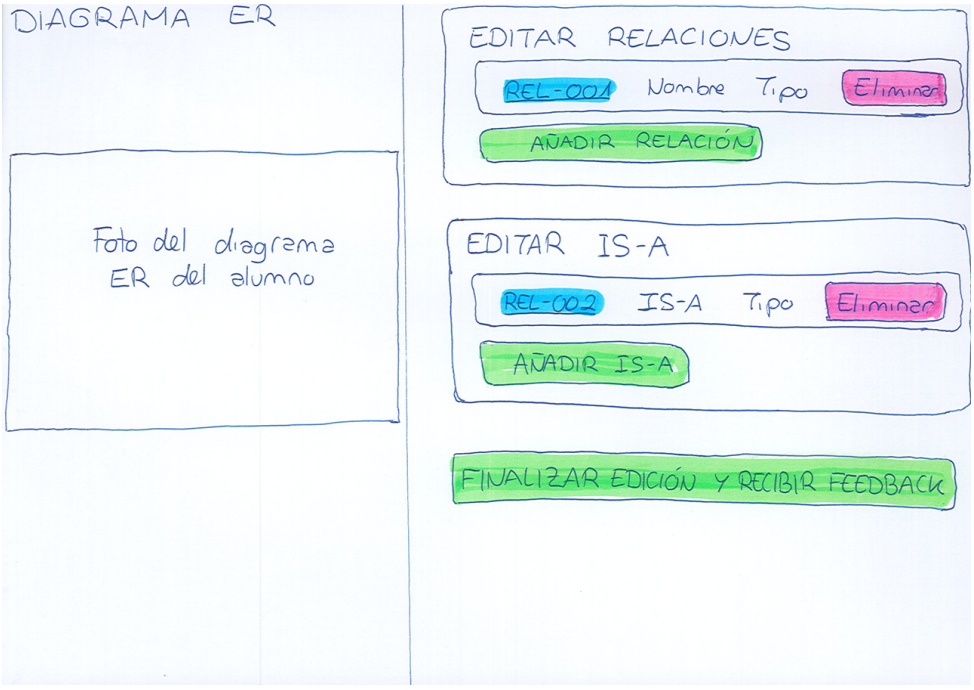

Figura 5.17: Boceto de la página en la que se editan las relaciones normales y las relaciones IS-A.

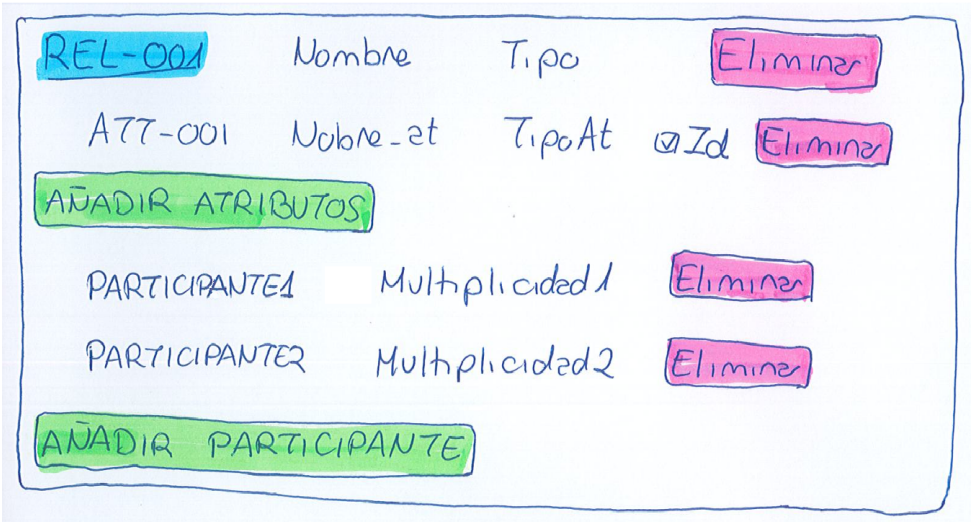

Figura 5.18: Boceto del desplegable de la página de relaciones que se abre al pinchar en una relación normal (no IS-A).

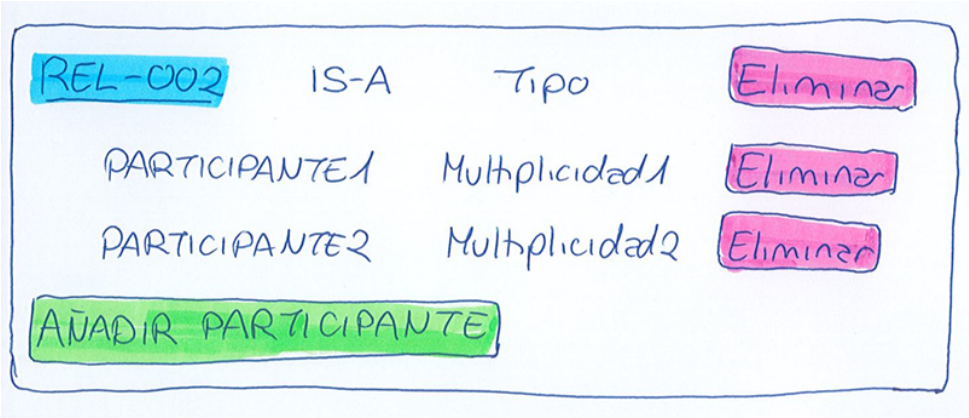

Figura 5.19: Boceto del desplegable de la página de relaciones que se abre al pinchar en una relación IS-A.

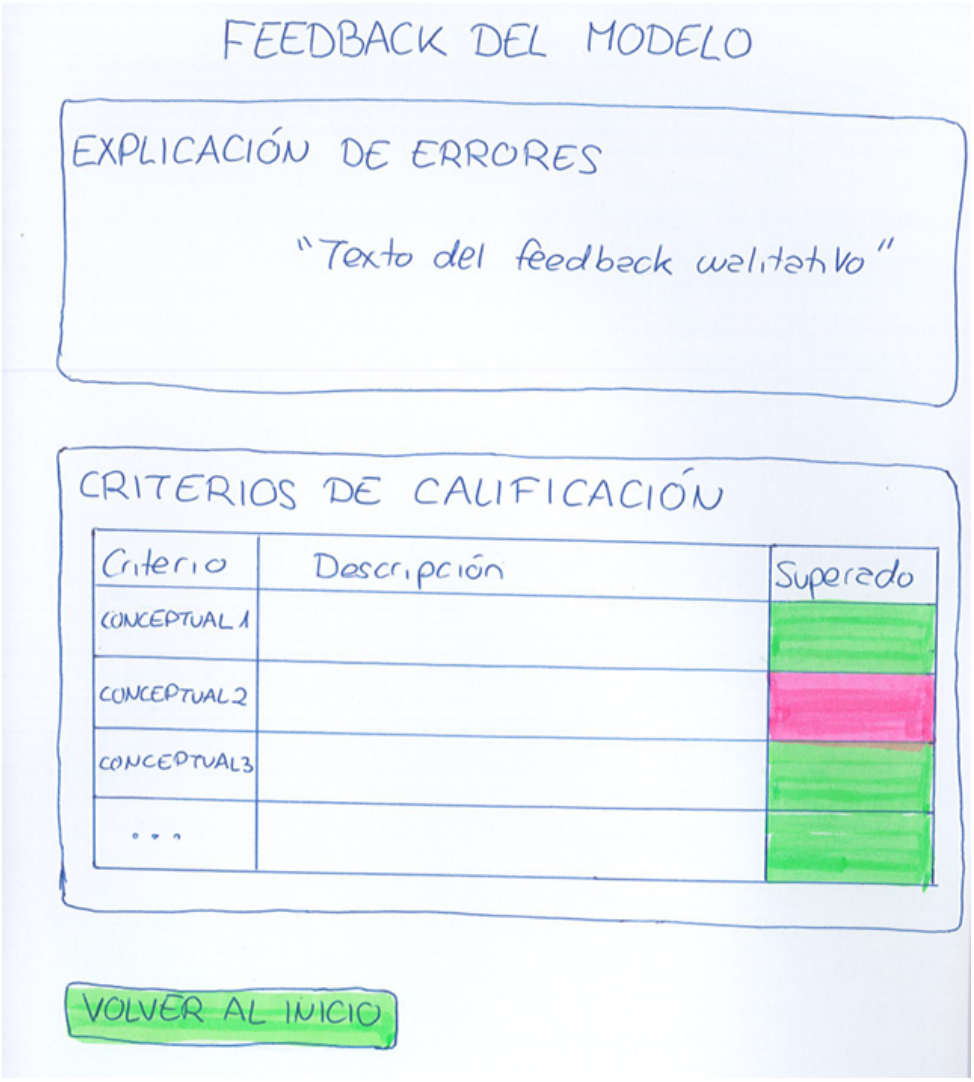

Figura 5.20: Boceto de la página en la que se mostrará el *feedback* (tanto el cualitativo como el cuantitativo).

Capítulo 6

Implementación

En este capítulo se detallarán algunos aspectos relevantes de la implementación de la aplicación. En primer lugar, en la Sección 6.1 se comentarán las herramientas utilizadas para la implementación de la aplicación, en la Sección 6.3 se explicarán con detalle los *prompts* utilizados para hacer las llamadas a la API de Openai, después se explicará como se genera el *feedback* cualitativo en la Sección 6.4 y finalmente se hablará sobre el estilo del *frontend* en la Sección 6.5.

6.1. Herramientas

Para el desarrollo de este proyecto se han seleccionado un conjunto de tecnologías que han permitido construir una aplicación web eficiente y moderna, abarcando tanto el *backend* como el *frontend*. En primer lugar, para la parte del *backend* se ha utilizado Python ya que proporciona numerosas librerías interesantes con las que nos ha sido muy útil trabajar. Una de ellas es LangChain con la que hemos gestionado interacciones con la API de OpenAI. Sobre esta base, se ha empleado el framework FastAPI, ya que facilita la creación de APIs de manera rápida y sencilla, ofreciendo además un alto rendimiento y documentación automática.

En cuanto al *frontend*, se ha optado por el desarrollo de la interfaz de usuario utilizando React junto con TypeScript. Esta combinación resulta especialmente potente, ya que React facilita la construcción de interfaces interactivas y dinámicas, mientras que TypeScript aporta un sistema de tipos que contribuye a prevenir errores y mejora el mantenimiento del código. Para el diseño de la aplicación, se ha utilizado TailwindCSS, un framework de funciones CSS que permite aplicar estilos de forma rápida, obteniendo un resultado visual atractivo y profesional.

Además, para establecer la conexión con la API de OpenAI se ha utilizado una clave de API proporcionada por la propia plataforma, la cual se almacena de forma segura mediante variables de entorno para evitar exponer datos sensibles en el código fuente. La autenticación se realiza enviando dicha clave en las cabeceras de las peticiones HTTP. La configuración básica consiste en inicializar un objeto ChatOpenAI de LangChain, especificando el modelo a utilizar (por ejemplo, gpt-4.1 o o3) y los parámetros deseados (como la temperatura). A partir de ahí, el *backend* de FastAPI expone los *endpoints* necesarios que reciben las peticiones del *frontend*, procesan los datos y devuelven la respuesta generada por el modelo de OpenAI.

En la Figura 6.1 se proporciona un esquema de como se relacionan las tecnologías para poder visualizar las herramientas con más claridad.

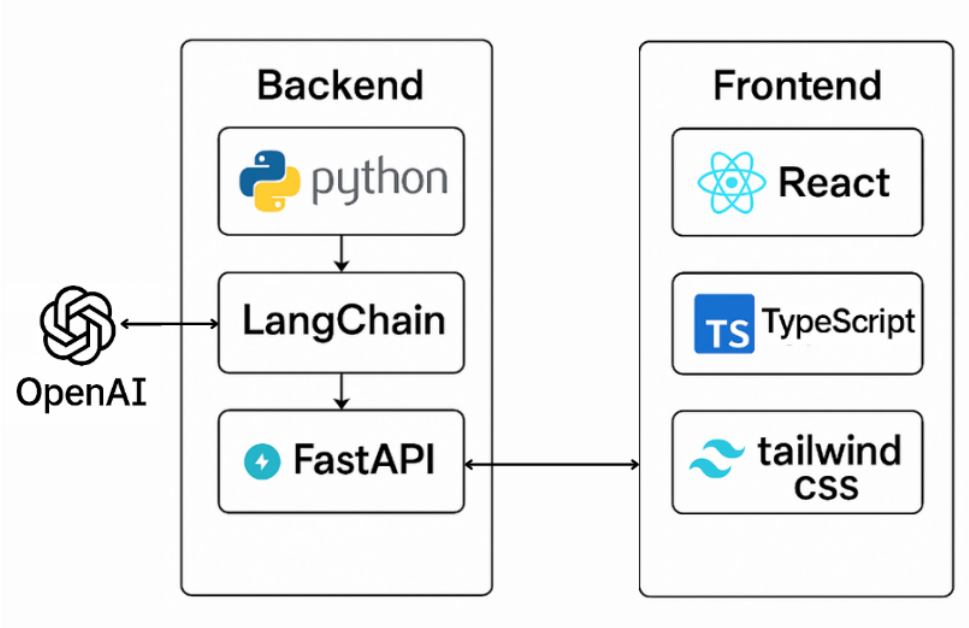

Figura 6.1: Esquema de las herramientas utilizadas en la aplicación y sus relaciones.

6.2. Preprocesado de las imágenes

El preprocesado de imágenes es una etapa esencial para garantizar que los modelos de inteligencia artificial puedan analizar correctamente los diagramas entidad-relación (ER) manuscritos, especialmente cuando provienen de fotografías. Estas imágenes suelen contener imperfecciones como sombras, fondos irregulares o ruido, lo que dificulta la interpretación automática. Por ello, se implementan una serie de transformación para optimizar los diagramas antes de enviarlos a la API de OpenAI y que así, los resultados de la identificación de los componentes sean mejores. A continuación, se describen las etapas aplicadas y cómo cada una de ellas contribuye a una mejor extracción de la información relevante:

1. **Conversión a escala de grises:** La fotografía del diagrama ER se transforma en una imagen en escala de grises. De este modo, se eliminan los colores del papel o la tinta que pueden distraer o generar ambigüedad. Así, el contraste entre el fondo y los trazos manuscritos se acentúa, facilitando el análisis estructural.

2. **Binarización adaptativa:** Se aplica un umbral adaptativo (*adaptive thresholding*) para convertir la imagen a blanco y negro. Esta técnica ajusta dinámicamente el umbral en función de las condiciones locales de la imagen, permitiendo separar eficazmente los trazos manuscritos del fondo, incluso en presencia de sombras o cambios de iluminación típicos de una foto.

3. **Eliminación de ruido:** Mediante operaciones morfológicas de apertura (*morphological opening*), se eliminan manchas, puntos y otros artefactos generados por imperfecciones del papel, la cámara o el entorno. Así se consigue que los elementos principales del diagrama, como entidades, relaciones y atributos, sean mucho más nítidos y reconocibles.

4. **Recorte automático:** Se detectan los contornos relevantes y se recorta automáticamente la imagen para centrarse exclusivamente en la región donde se encuentra el diagrama manuscrito. De esta forma, se descartan márgenes vacíos o elementos ajenos, asegurando que la API de OpenAI procese sólo la información esencial.

5. **Redimensionado:** Finalmente, la imagen recortada se ajusta a un ancho estándar (de 1200 píxeles), manteniendo la proporción original. Esto garantiza que la resolución sea la adecuada para el análisis, evitando tanto imágenes demasiado grandes (que podrían ralentizar el proceso) como demasiado pequeñas (que perderían detalle).

Gracias a este preprocesado, los diagramas ER manuscritos se presentan de forma clara, centrada y sin ruido ante la API de OpenAI, facilitando un reconocimiento mucho más preciso y eficiente tanto del texto como de la estructura del diagrama.

6.3. *Prompts*

Durante esta sección se irán explicando uno a uno los *prompts* utilizados en las llamadas a la API de OpenAI, detallando su estructura y el porqué de algunas de sus partes. Es importante tener en cuenta que para llegar a los *prompts* actuales se han ido haciendo numerosas pruebas para obtener resultados lo más parecidos posible a los que se necesitaban. Además del *prompt*, son importantes otros detalles como utilizar un modelo adecuado y otros parámetros que se detallan en la llamada a la API como *temperature*, *top_p*, *presence_penalty* o *frequency_penalty*:

- **temperature** [47]: Controla la aleatoriedad de las respuestas generadas por el modelo. Valores bajos producen respuestas más deterministas, mientras que valores altos fomentan mayor creatividad y variedad.

- **top_p** [47]: Define la probabilidad acumulada utilizada para el muestreo de palabras. Un valor de 1.0 incluye todas las palabras posibles; valores menores restringen las opciones a las más probables, reduciendo la imprevisibilidad.

- **presence_penalty** [47]: Penaliza la aparición de nuevos temas en la respuesta. Valores positivos incentivan al modelo a hablar de temas diferentes, mientras que valores negativos lo animan a repetir temas ya mencionados.

- **frequency_penalty** [47]: Penaliza la repetición de palabras en la respuesta generada. Un valor alto fuerza al modelo a ser más variado en el vocabulario; valores bajos o negativos permiten más repeticiones.

Con respecto al modelo se comenzó tabajando con el modelo *gpt-4o* pero finalmente en Abril de 2025 se lanzó el modelo *gpt-4.1* [47] que mostró numerosas ventajas sobre el anterior. En parte, esto se debe a que el *gpt-4.1* admite hasta 1 millón de *tokens* de contexto mientras que *gpt-4o* admite 128.000 *tokens*, lo cual es una cifra bastante alta pero notablemente inferior. El aspecto de los *tokens* de contexto es muy relevante en esta aplicación pues se trabaja además de con *tokens* largos, con imágenes con muchos componentes por analizar, con jsons muy largos (ya sea generarlos o analizarlos) y con enunciados muy largos de los ejercicios. Después se observó que para la tarea de comparación era mejor utilizar el modelo *o3* [47] que tiene el mismo número de *tokens* de contexto que el *gpt-4.1* pero es un modelo de razonamiento en vez de un modelo de

generación como los anteriores. En caso de que no se indique explícitamente que se ha utilizado el modelo *o3*, se da por hecho que el modelo utilizado para las llamadas a la API de OpenAI es el *gpt-4.1*.

6.3.1. *Prompt* de generación de entidades

En primer lugar, comenzaremos comentando el *prompt* de generación de entidades, su función es partiendo de una imagen de un diagrama ER generar un json con la información de las entidades y con la forma comentada en la Sección 5.3. A continuación se muestra el *prompt*:

```
Analiza la imagen adjunta de un diagrama entidad-relación que sigue la notación
de Chen. Identifica las entidades, clasificándolas como ‘‘fuertes’’ o ‘‘débiles’’. En
caso de que la imagen no corresponda con la de un diagrama ER, responde con un 0
únicamente, nada más. Una entidad se considera:

- ‘‘débil’’ si está representada por un rectángulo con doble línea.
- ‘‘fuerte’’ si está representada por un rectángulo con una sola línea.

Genera un JSON con la siguiente estructura:

{
  "entities": [
    {
      "id": "ENT-001",
      "name": "EjemploEntidad",
      "type": "fuerte o débil",
      "attributes": [
        {
          "id": "ATT-001",
          "name": "EjemploAtributo",
          "type": "simple, derivado o multivaluado",
          "PK": true
        }
      ]
    }
  ]
}
No incluyas explicaciones. Solo JSON válido.
```

Al inicio, se comienza explicando de forma clara qué debe de hacer el modelo. Después, se le solicita que responda con un "0" en caso de que la imagen no corresponda con un modelo ER, esto se hace para poder mostrar un error en el *frontend* y que el usuario suba una imagen válida. A continuación se observa como se le explica con detalle como reconocer una entidad débil y una fuerte, esto se añadió porque se observó que había problemas para identificar correctamente el tipo de una entidad. Luego, se proporciona un pequeño ejemplo de como debe ser el json de salida ya que los ejemplos hacen que el modelo comprenda exactamente cómo queremos que sea la salida, proporcionando resultdos mucho mejores, y finalmente se le informa de que no debe

incluir explicaciones, únicamente el json ya que esto es necesario para poder procesarlo de forma automática.

Además, las llamadas a la API con este *prompt* se hacen con una temperatura de 0, esto impide que el modelo "alucine" y se ciña a lo que detecta en la imagen.

6.3.2. *Prompt* de generación de relaciones

Este *prompt* es muy similar al de generación de entidades, pero una de las principales diferencias es que además de recibir la imagen del diagrama ER, recibe el json de entidades que se generó con el *prompt* anterior. Esto se debe a que inicialmente se observó que si se intentaba generar el json del modelo entero con un solo *prompt*, había muchos elementos que el modelo no era capaz de identificar (debido a que se agotaban los *tokens* de contexto) así que se decidió dividir la tarea en dos fases, esto implica que para la generación de las relaciones se conozca cuales son las entidades y parte de su información para poder generar correctamente el json de relaciones. A continuación se presenta dicho *prompt*:

```
Usa el siguiente JSON de entidades: json_entidades
Y analiza la imagen adjunta de un diagrama entidad-relación (notación de Chen)
para generar un JSON con todas las relaciones.
Identifica los siguientes tipos de relaciones:

- Relaciones débiles: representadas con rombos de doble línea. Siempre están
  conectadas a una entidad débil.
- Relaciones fuertes: representadas con rombos de una sola línea.
- Relaciones IS-A: representadas por un triángulo invertido, deben tener el
  nombre ‘‘IS-A’’ y en el campo ‘‘type’’ deben especificar si son:
    • ‘‘obligatoria y disjunta’’ (triángulo invertido + circulito encima
      (obligatoria) + línea curva debajo (disjunta))
    • ‘‘obligatoria y no disjunta’’ (triángulo invertido + un circulito
      encima (obligatoria))
    • ‘‘no obligatoria y disjunta’’ (triángulo invertido + línea curva debajo
      (disjunta))
    • ‘‘no obligatoria y no disjunta’’ (solo un triángulo invertido SIN
      circulito y SIN curva debajo (también es una relación IS-A válida))

  Además, para las relaciones IS-A el campo ‘‘multiplicity’’ de todas las
  entidades del campo ‘‘participants’’ debe ir vacío.

  El primer participante de una relación IS-A debe corresponder con la
  entidad ‘‘padre’’ y las demás deben ser las entidades ‘‘hijas’’.

Importante: quiero que escribas la multiplicidad separada por comas, por ejemplo:
0,1; 1,1; 1,N o 0,N.
Importante: recuerda que las relaciones suelen tener como nombre un verbo en
infinitivo.
Genera un JSON con esta estructura:
```

```
{
  "relationships": [
    {
      "id": "REL-001",
      "name": "NombreRelacion",
      "type": "fuerte, débil o descripción IS-A",
      "participants": [
        {
          "entity": "NombreEntidad1",
          "multiplicity": "1..n"
        },
        {
          "entity": "NombreEntidad2",
          "multiplicity": "0..n"
        }
      ],
      "attributes": [
        {
          "id": "ATT-003",
          "name": "AtributoRelacion",
          "type": "simple",
          "PK": false
        }
      ]
    }
  ]
}
No incluyas explicaciones. Solo JSON válido.
```

Se observa que la estructura es muy similar a la del prompt anterior, en este caso se observó que no reconocía bien los tipos de relaciones y los subtipos de las relaciones IS-A así que eso es lo que se le ha explicado con más detalle. También, gracias a numerosas pruebas se detectó que en numerosas ocasiones las multiplicidades las ponía con puntos suspensivos en vez de con comas y que los nombres de las relaciones los cambiaba por verbos conjugados (por ejemplo, recibe en vez de recibir) luego ambas cosas se han puesto como importantes. Finalmente, encontramos el ejemplo del json de salida y la solicitud de recibir únicamente el json sin texto complementario.

Además, las llamadas a la API con este *prompt* se hacen con una temperatura de 0, esto impide que el modelo "alucinez se ciña a lo que detecta en la imagen.

6.3.3. *Prompt* de comparación

A continuación, se comenta la primera versión del *prompt* encargado de comparar el modelo del profesor con el modelo del alumno, su objetivo es devolver un texto correspondiente a un csv corrigiendo diferentes elementos del modelo del alumno basándose en el modelo del profesor. El *prompt* es el siguiente:

Eres un experto en modelado de bases de datos y tu tarea es actuar como tutor automático. Se te proporcionan dos modelos entidad-relación (ER), representados en formato JSON. Uno corresponde al modelo correcto realizado por el profesor, y el otro corresponde al modelo realizado por un alumno. Ambos modelos pueden utilizar diferentes nombres de entidades, atributos o relaciones, pero lo importante es que representen los mismos conceptos desde el punto de vista del modelado.
Tu tarea es comparar ambos modelos y devolver una tabla con los resultados de la comparación, en formato CSV (separado por punto y coma ‘;‘).
IMPORTANTE:

- NO incluyas ninguna explicación, comentario, encabezado ni texto adicional.
- Solo responde con texto plano en formato CSV válido, ya que será procesado automáticamente. Cualquier otro tipo de contenido producirá errores.

Criterios de comparación
Debes identificar equivalencias entre elementos del modelo del profesor y del alumno aunque los nombres no coincidan exactamente. Usa las siguientes reglas:
Entidades

- Considera que dos entidades son equivalentes si tienen atributos similares y cumplen roles similares en las relaciones, aunque sus nombres cambien.
- Compara sus atributos, si son identificadores, y si están correctamente modeladas como fuertes o débiles.
- Los nombres de los atributos tampoco tienen por qué coincidir exactamente, simplemente basta con que representen lo mismo. Por ejemplo: f_inicio, fechaComienzo, Inicio y Comienzo podrían representar al mismo atributo aunque los
  nombres sean diferentes.

Relaciones (que no sean IS-A)

- Para cada relación en el modelo del profesor, identifica su equivalente en el modelo del alumno aunque el nombre cambie. Para determinar si una relación es equivalente:
  - Verifica si las entidades participantes en la relación del profesor también aparecen en alguna relación del alumno (aunque con nombres distintos pero roles similares).
  - Si hay coincidencias en los atributos y los participantes son equivalentes, considera que se trata de la misma relación aunque el nombre sea distinto.
  - No te bases solo en el nombre para emparejar relaciones: prioriza la coincidencia de participantes y estructura.

- Ejemplo: una relación ‘‘OFRECER’’ con participantes ‘‘COMPRADOR’’ y ‘‘PUJA’’ puede considerarse equivalente a una relación ‘‘HACER’’ si esta también conecta ‘‘COMPRADOR’’ y ‘‘PUJA’’, aunque los nombres difieran.

Relaciones tipo IS-A

- Compara las jerarquías tipo IS-A y determina si están correctamente representadas aunque haya diferencias de nombre o estilo.
- Evalúa si la relación aparece y si el tipo de generalización/especialización es correcto.

Formato esperado

1. Entidades (que no sean relaciones ni IS-A):
   ID;esDebil;correccionNombre;correccionTipo;correccionPK;
   númeroAtributosCorrectos;númeroAtributosIncorrectos
   esDebil = será 0 si la entidad es fuerte y 1 si la entidad es débil
   correccionPK = será 1 si la entidad del profesor tiene los mismos atributos PK que la entidad del alumno
2. Relaciones (que no sean IS-A):
   ID;correccionNombre;correccionTipo;númeroAtributosCorrectos;
   númeroAtributosIncorrectos;correccionPrimeraMultiplicidad;
   correccionSegundaMultiplicidad
   correccionNombre realmente será 1 si la relación aparece y 0 en caso contrario
3. Relaciones tipo IS-A:
   ID;correcciónAparece;correcciónTipo
   Solo incluye líneas IS-A si existen en el modelo del profesor.
4. Todas las correcciones deben expresarse como:
   - 1 si el aspecto está correctamente representado.
   - 0 si no lo está.
5. Entidades fuertes incorrectas:
   EntidadesFuertesMal;numEntidadesFuertesMal
   numEntidadesFuertesMal: será un único número que represente el número de entidades fuertes que están en el json del alumno y no se relacionan con ninguna entidad del json del profesor
6. Entidades débiles incorrectas:
   EntidadesDebilesMal;numEntidadesDebilesMal
   numEntidadesDebilesMal: será un único número que represente el número de entidades débiles que están en el json del alumno y no se relacionan con ninguna entidad del json del profesor

```
7. Relaciones incorrectas (que no sean IS-A):
   RelacionesMal;numRelacionesMal
   numRelacionesMal: será un único número que represente el número de relaciones
   que están en el json del alumno y no se relacionan con ninguna relación del
   json del profesor

8. IS-A incorrectas:
   ISAMal;numISAMal
   numISAMal: será un único número que represente el número de relaciones tipo
   IS-A que están en el json del alumno y no se relacionan con ninguna relación
   tipo IS-A del json del profesor

El resultado debe cubrir todos los elementos (entidades, relaciones, IS-A)
presentes en el modelo del profesor más los números que cuentan los elementos
incorrectos del json del alumno.
A continuación, se proporcionarán dos objetos JSON:
   Modelo profesor: {json_profesor}
   Modelo alumno: {json_alumno}
Solo responde con el contenido CSV. No incluyas encabezados, explicaciones, texto
adicional ni comentarios.
```

En primer lugar, se comienza asignando un "rol" al modelo, de esta forma contestará de forma más precisa a la pregunta y justo después se detalla la tarea que debe realizar. En este caso era muy importante dejar claro que aunque un elemento en el modelo del profesor y otro elemento del modelo del alumno tuvieran distinto nombre, podían representar lo mismo (esto se detalla en el apartado de criterios de comparación). Después, en el apartado de formato esperado, se explica de forma muy detallada cómo debe ser el formato del csv. Finalmente, se proporcionan los jsons y se recalca que la respuesta no debe contener explicaciones ni texto adicional.

En este caso,la temperatura tiene un valor de 0.3 (de esta forma el modelo no será tan estricto pero tampoco tendrá apenas margen para alucinar), el top_p será 1 y el presence_penalty y el frequency_penalty serán 0. Con este *prompt*, en las pruebas de aceptación se vio que los resultados no eran suficientemente buenos así que se decidió implementar otras tres mejoras:

1. Dividir el *prompt* en 3 partes para hacer 3 llamadas: una para hacer la comparación para las entidades, otra para las relaciones normales y otra para las relaciones IS-A.

2. Enviar en el *prompt* solo el fragmentos de json que tenía que ver con los elementos que se iban a comparar.

3. Empezar a trabajar con el modelo *o3* en vez del *gpt-4.1* ya que es un modelo de razonamiento en vez de un modelo de generación.

A continuación, se adjuntan los *prompts* finales que se han utilizado en la aplicación para realizar la comparación del modelo. *Prompt* para comparar las entidades:

```
Eres un experto en modelado de bases de datos y tu tarea es actuar como tutor
automático. Se te proporcionan dos modelos entidad-relación (ER), representados en
formato JSON. Uno corresponde al modelo correcto realizado por el profesor, y el
```

otro al modelo realizado por un alumno. Ambos pueden tener diferentes nombres en entidades o atributos.

Tu tarea: Compara solo las ENTIDADES de ambos modelos y devuelve los resultados en formato CSV (separado por punto y coma ‘;‘), sin encabezados ni texto adicional. Debe haber una línea por cada entidad del profesor, y para cada entidad del profesor debe comprobarse si el alumno modela como el profesor cada uno de los aspectos.

Debes identificar equivalencias entre elementos del modelo del profesor y del alumno aunque los nombres no coincidan exactamente. Usa las siguientes reglas:
Criterios:

1. Considera que dos entidades son equivalentes si tienen atributos similares y cumplen roles similares en las relaciones, aunque sus nombres cambien.
2. Compara sus atributos, si son clave primaria, y si están correctamente modeladas como fuertes o débiles.
3. Los nombres de los atributos tampoco tienen por qué coincidir exactamente, simplemente basta con que representen lo mismo. Por ejemplo: f_inicio, fechaComienzo, Inicio y Comienzo podrían representar al mismo atributo aunque los nombres sean diferentes.
4. Solo responde líneas de entidades, más el resumen de entidades fuertes/débiles incorrectas.

Formato de salida:
ID;esDebil;correccionNombre;correccionTipo;correccionPK;
númeroAtributosCorrectos;númeroAtributosIncorrectos
EntidadesFuertesMal;numEntidadesFuertesMal
EntidadesDebilesMal;numEntidadesDebilesMal

1. esDebil será 0 si la entidad es débil y 1 si la entidad es fuerte
2. Todas las correcciones deben expresarse como:
   a) 1 si el aspecto está correctamente representado.
   b) 0 si no lo está
3. numEntidadesFuertesFuertesMal: será un único número que represente el número de entidades fuertes que están en el json del alumno y no se relacionan con ninguna entidad del json del profesor.
4. numEntidadesDebilesMal: será un único número que represente el número de entidades débiles que están en el json del alumno y no se relacionan con ninguna entidad del json del profesor

```
MUY IMPORTANTE!!! El resultado debe cubrir TODAS las entidades presentes en
el modelo del profesor (ninguna línea por las del alumno) más los números que
cuentan las entidades incorrectas del json del alumno.

No incluyas encabezados ni explicaciones. Solo CSV.

Modelo profesor:
{json_profesor}

Modelo alumno:
{json_alumno}
```

Prompt para comparar las relaciones:

```
Eres un experto en modelado de bases de datos y tu tarea es actuar como tutor
automático. Se te proporcionan dos modelos ER en JSON (profesor y alumno). Ambos
pueden usar diferentes nombres para relaciones o entidades.

Tu tarea: Compara solo las RELACIONES (que no sean IS-A) de ambos modelos y
devuelve los resultados en formato CSV (separado por punto y coma ';'), sin
encabezados ni texto adicional. Debe haber una línea por cada relación del
profesor, y para cada relación del profesor debe comprobarse si el alumno modela
como el profesor cada uno de los aspectos.

Debes identificar equivalencias entre elementos del modelo del profesor y del
alumno aunque los nombres no coincidan exactamente. Usa las siguientes reglas:
Criterios:
Para cada relación en el modelo del profesor, identifica su equivalente en el
modelo del alumno aunque el nombre cambie. Para determinar si una relación
es equivalente:

  1. Verifica si las entidades participantes en la relación del profesor también
     aparecen en alguna relación del alumno (aunque con nombres distintos pero
     roles similares).

  2. Si hay coincidencias en los atributos y los participantes son equivalentes,
     considera que se trata de la misma relación aunque el nombre sea distinto.

  3. Las multiplicidades pueden estar en distinto orden, pero deben coincidir en
     contenido.

  4. No te bases solo en el nombre para emparejar relaciones: prioriza la
     coincidencia de participantes y estructura.

Ejemplo: una relación 'OFRECER' con participantes 'COMPRADOR' y 'PUJA' puede
considerarse equivalente a una relación 'HACER' si esta también conecta 'COMPRADOR'
y 'PUJA', aunque los nombres difieran.
```

1. Solo responde líneas de relaciones, más el resumen de relaciones incorrectas.

Formato de salida:
ID;correccionNombre;correccionTipo;númeroAtributosCorrectos;
númeroAtributosIncorrectos;correccionPrimeraMultiplicidad;
correccionSegundaMultiplicidad

RelacionesMal;numRelacionesMal

1. correccionNombre será 1 si la relación aparece y 0 en caso contrario
2. Todas las correcciones deben expresarse como:
   a) 1 si el aspecto está correctamente representado.
   b) 0 si no lo está.
3. numRelacionesMal: será un único número que represente el número de relaciones que están en el json del alumno y que no se relacionan con ninguna relación del json del profesor, debes estar seguro de que efectivamente no corresponden a ninguna relación del profesor, recuerda que pueden tener nombres diferentes pero representar la misma relación.
4. Importante: recuerda lo que dijimos de cómo identificar las relaciones.

MUY IMPORTANTE!!! El resultado debe cubrir TODAS las relaciones presentes en el modelo del profesor (ninguna línea por las del alumno) más el número que cuenta las relaciones incorrectas del json del alumno.

No incluyas encabezados ni explicaciones. Solo CSV.

Modelo profesor:
{json_profesor}

Modelo alumno:
{json_alumno}

Prompt para comparar las relaciones IS-A:

Eres un experto en modelado de bases de datos y tu tarea es actuar como tutor automático. Se te proporcionan dos modelos ER en JSON (profesor y alumno). Ambos pueden tener diferentes nombres para las relaciones IS-A o las entidades participantes.

Tu tarea: Compara solo las relaciones tipo IS-A de ambos modelos y devuelve los resultados en formato CSV (separado por punto y coma ‘;‘), sin encabezados ni texto adicional. Debe haber una línea por cada IS-A del profesor, y para cada IS-A del profesor debe comprobarse si el alumno modela como el

profesor cada uno de los aspectos.

Debes identificar equivalencias entre elementos del modelo del profesor y del alumno aunque los nombres no coincidan exactamente. Usa las siguientes reglas:
Criterios:

1. Evalúa si la relación IS-A aparece y si el tipo de generalización/especialización es correcto.

2. Compara las jerarquías tipo IS-A y determina si están correctamente representadas aunque haya diferencias en el nombre de algún participante.

3. Evalúa si la relación aparece y si el tipo de generalización/especialización es correcto.

4. Solo responde líneas de relaciones IS-A, más el resumen de IS-A incorrectos.

Formato de salida:
ID;correcciónAparece;correcciónTipo
ISAMal;numISAMal

1. Todas las correcciones deben expresarse como:

   a) 1 si el aspecto está correctamente representado.

   b) 0 si no lo está.

2. numISAMal: será un único número que represente el número de relaciones tipo IS-A que están en el json del alumno y no se relacionan con ninguna relación tipo IS-A del json del profesor

MUY IMPORTANTE!!! El resultado debe cubrir TODAS las relaciones presentes en el modelo del profesor (ninguna línea por las del alumno) más el número que cuenta las relaciones incorrectas del json del alumno.

No incluyas encabezados ni explicaciones. Solo CSV.

Modelo profesor:
{json_profesor}

Modelo alumno:
{json_alumno}

6.3.4. *Prompt* para evaluar si la notación utilizada por el estudiante es correcta

Este *prompt* se utiliza para evaluar uno de los criterios de calificación con los que se trabaja cuyo objetivo es básicamente evaluar si la notación que ha utilizado el estudiante para representar el diagrama ER es correcta, para ello se le envía al modelo la imagen del diagrama ER junto con el *prompt* que se presenta a continuación:

```
Evalúa si el siguiente diagrama entidad-relación cumple en general con la notación
de Chen:
Si visualmente se identifican en general las entidades, atributos y relaciones con
sus formas respectivas (aunque haya leves inconsistencias), responde con 0.
Solo responde 1 si la notación está mayormente mal aplicada. No expliques tu
respuesta ni añadas ningún otro texto o caracter ya que el resultado se
procesará automáticamente y sino dará error.
```

En primer lugar, se explica la tarea que debe realizar, y se le dice el formato con el que debe contestar, finalmente se recalca que es importante que no añada a la respuesta nada más que el 0 o el 1. En este caso, se observó que cuando "obligas" al modelo a responder con sí o no (0 o 1), el modelo tiende a curarse en salud y responder que no por lo que tuvo que destacarse que a no ser que claramente el modelo de la imagen no siguiera la notación de Chen la respuesta debía ser 0.

Además, las llamadas a la API con este *prompt* se hacen con una temperatura de 0, esto impide que el modelo "alucine" y se ciña a lo que detecta en la imagen.

6.3.5. *Prompt* generación de *feedback* cualitativo

Este *prompt* se utiliza para generar un texto que explique al alumno las cosas positivas y negativas de su modelo y le de sugerencias para corregirlo pero sin llegar a decirle la respuesta, esto se hace así para ayudar al alumno a pensar por sí mismo y para que aprenda a corregir sus propios errores. El *prompt* es el siguiente:

```
Actúa como un experto en diseño de bases de datos y profesor de modelado
entidad-relación.
Recibirás dos entradas:
Una descripción de negocio (requisitos funcionales en lenguaje natural).
Un diagrama ER del estudiante representado como un JSON estructurado.
Tu tarea es:
```

- Analizar el texto de negocio y extraer las entidades, relaciones, atributos y restricciones que deberían estar representados en el modelo.
- Comparar lo que debería estar representado, según los requisitos, con lo que el estudiante modeló en su JSON.
- Detectar:
    - Entidades o relaciones importantes que faltan.

    - Entidades, relaciones o atributos incorrectamente modelados (tipo de entidad, tipo de relación, errores de clave primaria, multiplicidades erróneas, atributos mal definidos...).
    - Elementos que sobran (innecesarios o fuera de los requisitos).
    - Errores conceptuales en la modelización.
- Explicar de forma detallada qué aspectos del modelo no cumplen con los requisitos del negocio, sin proporcionar la solución correcta. El objetivo es que el estudiante identifique y corrija sus errores por sí mismo.

Consideraciones importantes sobre relaciones IS-A:

- Si los requisitos describen tipos o categorías de una entidad general, debe modelarse una relación IS-A.
- Verifica que los atributos comunes estén en la entidad padre y los atributos específicos en las entidades hijas.
- Revisa si la especialización es disjunta o solapada y total o parcial, y que estas restricciones estén claramente indicadas.
- Comprueba que las entidades hijas hereden correctamente la clave primaria de la entidad padre.

Cuando en el JSON se declare una relación con ‘‘name’’: ‘‘IS-A’’, debes interpretar que las entidades participantes son una jerarquía de generalización/especialización. Interpretación obligatoria de las relaciones IS-A en el JSON:

- Cada relación con ‘‘name’’: ‘‘IS-A’’ indica una jerarquía de generalización/especialización.
    - El primer participante es la entidad padre.
    - Los demás participantes son entidades hijas, aunque en su definición aparezcan con ‘‘type’’: ‘‘fuerte’’.
- Cuando exista esta relación:
    - Considera que las hijas heredan la clave primaria de la entidad padre, salvo que declaren expresamente otra PK adicional.
    - No señales como error que las hijas estén tipificadas como ‘‘fuerte’’ ni que carezcan de atributos propios: pueden ser meros roles sin atributos adicionales.
- Solo debe marcarse un problema si:
    - se declara una PK distinta en la hija sin justificación,
    - faltan atributos comunes en el padre,
    - o la relación IS-A contradice las restricciones (disjunta/solapada, total/parcial) especificadas en el atributo ‘‘type’’.

```
Muy importante: Responde únicamente rellenando la siguiente estructura. No añadas
texto fuera de ella, ni saludos, ni conclusiones adicionales.
Estructura obligatoria de la respuesta:
```

- `Aspectos Correctos: Redacta un párrafo general destacando los aciertos del modelo del estudiante, sin enumerar elementos específico`
- `Errores Detectados: Indica cada error detectado, explicando qué parte del modelo no es coherente con el texto de negocio y por qué. No proporciones la solución correcta.`
- `Sugerencias y Mejoras: Ofrece consejos generales o preguntas orientadoras para que el estudiante pueda reflexionar y mejorar su modelo.`

```
Datos de entrada:
   Descripción de Negocio:
{enunciado}
   Aquí se proporciona el JSON del diagrama del estudiante:
{json_alumno}
Realiza el análisis en base a la descripción de negocio y responde únicamente
siguiendo la estructura indicada.
```

Al igual que en el *prompt* de comparación, se comienza asignando un "rol" al modelo, de esta forma contestará de forma más precisa a la pregunta y justo después se detallan las entradas que recibirá (el modelo del alumno y el enunciado del ejercicio) y la tarea que debe realizar. En este caso se observó que no corregía bien las relaciones IS-A así que se le proporciona información extra sobre como hacerlo correctamente. Después, se explica de forma muy detallada como debe ser el formato de la salida, explicando las 3 partes que debe de contener la respuesta y que tipo de información debe de incluir cada una. Finalmente, se proporciona el enunciado del ejercicio y el json del alumno y se recalca que la respuesta debe de seguir la estructura indicada.

Además, las llamadas a la API con este *prompt* se hacen con una temperatura de 0, esto impide que el modelo "alucinez se ciña a su tarea.

6.4. Generación del *feedback* cuantitativo

La evaluación de los modelos ER se realiza de acuerdo con los criterios de aceptación establecidos en la asignatura de Sistemas de Bases de Datos del Grado en Ingeniería Informática de Servicios y Aplicaciones, impartida en la Universidad de Valladolid. La evaluación de estos modelos en la asignatura se basa en 11 criterios de aceptación, los cuales permiten medir de manera independiente los distintos aspectos que definen dicho modelo y que se detallan a continuación:

- **CONCEPTUAL-2.2.** Soy capaz de caracterizar las entidades presentes en el "mini-mundo" de una base de datos.
- **CONCEPTUAL-2.3.** Soy capaz de decidir cuándo una entidad debe modelarse como débil, de acuerdo con su dependencia existencial de una entidad fuerte.
- **CONCEPTUAL-2.4.** Soy capaz de caracterizar los atributos que describen cualquier entidad (o relación) presente en el "mini-mundo" de una base de datos.

- **CONCEPTUAL-2.5.** Soy capaz de elegir el identificador de una entidad, a partir de sus atributos descriptivos.
- **CONCEPTUAL-2.6.** Soy capaz de caracterizar las relaciones existentes entre cualquier entidad presente en el "mini-mundo" de una base de datos.
- **CONCEPTUAL-2.7.** Soy capaz de determinar la participación y las cardinalidades propias de relaciones de cualquier grado.
- **CONCEPTUAL-2.8.** Soy capaz de construir el modelo entidad-relación que establece el diseño conceptual de una base de datos.
- **CONCEPTUAL-2.9.** Soy capaz de utilizar una notación estandarizada para expresar el diseño conceptual (y mantener un uso consistente de la misma durante todo el ciclo de vida).
- **CONCEPTUAL-4.1.** Soy capaz de caracterizar las relaciones generalización-especialización (IS-A) existentes en el 'mini-mundo'de una base de datos.
- **CONCEPTUAL-4.2.** Soy capaz de determinar la obligatoriedad y la disyunción propias de relaciones IS-A.
- **CONCEPTUAL-4.3.** Soy capaz de construir el modelo entidad-relación extendido que establece el diseño conceptual de una base de datos.

Utilizando los datos del CSV generado tras la llamada a la API de OpenAI con el *prompt* presentado en la subsección 6.3.3 y el resultado del *prompt* mostrado en la subsección 6.3.4, se procede a evaluar si se ha superado o no cada criterio. Para ello, se tienen en utilizan los umbrales de evaluación y otras observaciones de cada criterio, que se detallan a continuación.

- **CONCEPTUAL-2.2.** El modelo ER identifica correctamente, al menos, el 60 % de las entidades "regulares" necesarias para abordar los requisitos establecidos en el pliego técnico. Cada entidad incorrecta restará en la misma proporción que las entidades correctas.
- **CONCEPTUAL-2.3.** El modelo ER identifica correctamente, al menos, el 60 % de las entidades "débiles" necesarias para abordar los requisitos establecidos en el pliego técnico. Cada entidad incorrecta restará en la misma proporción) que las entidades correctas.
- **CONCEPTUAL-2.4.** Las entidades planteadas en el modelo ER declaran correctamente, al menos, el 60 % de los atributos que caracterizan su información, de acuerdo con los requisitos establecidos en el pliego técnico. Cada atributo incorrecto restará en la misma proporción que los atributos correctos.
- **CONCEPTUAL-2.5.** Al menos el 60 % de las entidades planteadas en el modelo ER declara un identificador válido, de acuerdo con los requisitos establecidos en el pliego técnico. Cada identificador incorrecto restará en la misma proporción que los identificadores correctos y, además, invalidará la entidad correspondiente.
- **CONCEPTUAL-2.6.** El modelo ER identifica correctamente, al menos, el 60 % de las relaciones esperadas de acuerdo con el pliego técnico. Cada relación incorrecta restará en la misma proporción que las relaciones correctas.

- **CONCEPTUAL-2.7.** El modelo establece correctamente las multiplicidades que describen a las entidades participantes en, al menos, el 60 % de las relaciones esperadas de acuerdo con el pliego técnico. Cada multiplicidad incorrecta restará en la misma proporción que las multiplicidades correctas.
- **CONCEPTUAL-2.8 / 4.3.** El modelo ER utiliza los componentes de forma semánticamente correcta, de acuerdo con los principios establecidos en la asignatura. El uso de cualquier "artefacto" diferente a los establecidos supondrá la no aceptación de este criterio.
- **CONCEPTUAL-2.9.** Al menos el 75 % de los nombres de entidades, relaciones y atributos utilizados en el modelo ER respetan la sintaxis establecida en la asignatura.
- **CONCEPTUAL-4.1.** El modelo ER identifica correctamente, al menos, el 50 % de las relaciones IS-A esperadas de acuerdo con el pliego técnico. Cada relación incorrecta IS-A restará en la misma proporción que las relaciones IS-A correctas.
- **CONCEPTUAL-4.2.** El modelo establece correctamente, al menos, el 50 % de las restricciones de obligatoriedad y disyunción propias de las relaciones IS-A esperadas de acuerdo con el pliego técnico. Cada par de restricciones de obligatoriedad y disyunción incorrecto restará en la misma proporción que las relaciones IS-A correctas.

6.5. Estilo *frontend*

Para el *frontend*, se ha decidido utilizar un estilo tipo *hacker* pensando especialmente en el público objetivo de la aplicación, que son los estudiantes universitarios de informática. Este estilo es muy original y la temática es muy popular, lo que llama más la atención de los estudiantes y por tanto, hará que más gente la use. Además tiene otra ventaja para los usuarios y es que las interfaces con fondos de color negro son mejores para la vista lo cual es muy importante porque frecuentemente los estudiantes de informática pasan mucho tiempo frente a pantallas.

A continuación se describen brevemente las distintas páginas y componentes principales de la interfaz, mostrando su contenido y finalidad:

- **Página de inicio (Figura 6.2):** Permite al alumno seleccionar el ejercicio que desea corregir. Es la puerta de entrada al resto de funcionalidades de la aplicación.
- **Subida de diagrama ER (Figura 6.3):** Interfaz donde el usuario puede cargar la imagen del diagrama de entidad-relación (ER) que se quiere corregir, iniciando así el proceso de corrección automática.
- **Confirmación de imagen cargada (Figura 6.4):** Pantalla que confirma al usuario que la imagen del diagrama ER ha sido correctamente seleccionada antes de continuar con el análisis.
- **Gestión de entidades (Figura 6.5):** Página donde se muestran las entidades detectadas por el sistema y permite al usuario modificarlas o corregirlas. Incluye la imagen del diagrama ER para facilitar la edición.
- **Detalles de entidad (Figura 6.6):** Desplegable que aparece al seleccionar una entidad para consultar o editar sus detalles individuales.

- **Edición y confirmación de entidades (Figura 6.7):** Apartado para añadir nuevas entidades o confirmar que la información de las entidades ya es correcta, facilitando el flujo de trabajo del usuario.
- **Gestión de relaciones (Figura 6.8):** Página para visualizar y modificar las relaciones identificadas por el sistema, también mostrando la imagen del diagrama ER para referencia visual.
- **Detalles de relación (Figura 6.9):** Desplegable que muestra los detalles de una relación seleccionada, permitiendo editar la información correspondiente.
- **Relaciones IS-A (Figura 6.10):** Sección que muestra y permite modificar las relaciones de tipo IS-A, diferenciadas de las relaciones normales para mayor claridad.
- **Detalles de relación IS-A (Figura 6.11):** Desplegable específico para consultar y modificar los detalles de una relación IS-A seleccionada.
- ***Feedback* cualitativo (Figura 6.12):** Parte de la página de *feedback* que proporciona explicaciones y sugerencias personalizadas para que el usuario entienda los errores y áreas de mejora.
- ***Feedback* cuantitativo (Figura 6.13):** Sección de la página de *feedback* que muestra una tabla con los criterios de evaluación, indicando visualmente si cada uno ha sido superado o no.

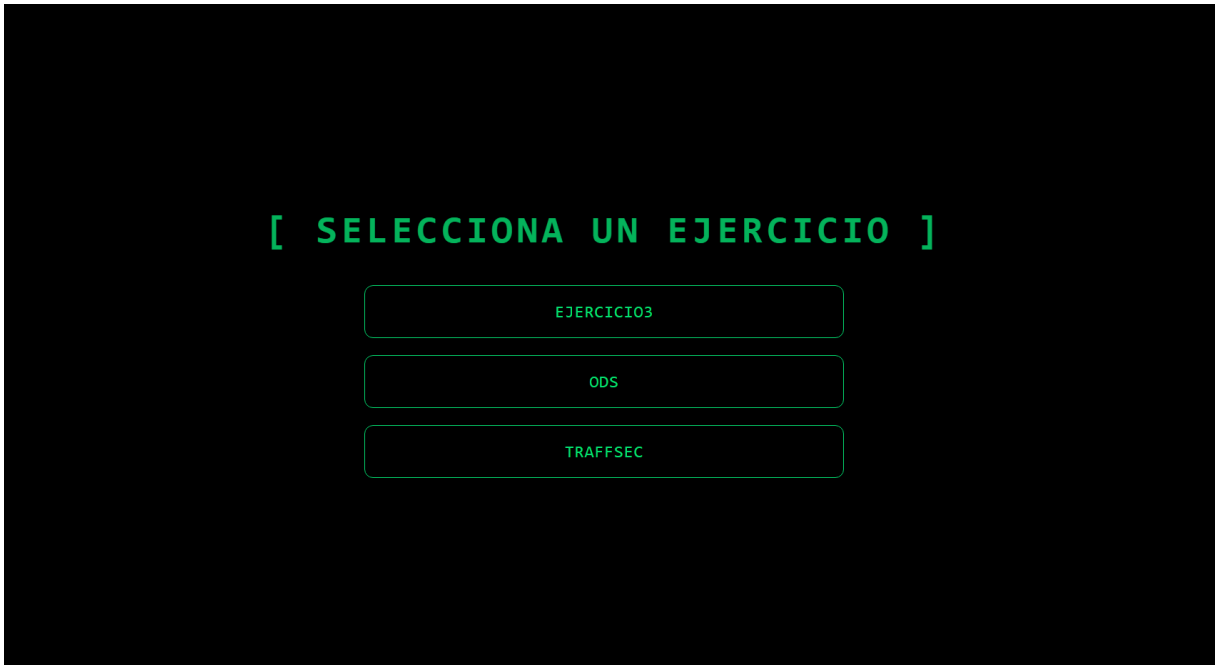

Figura 6.2: Página de inicio.

Figura 6.3: Subida de diagrama ER.

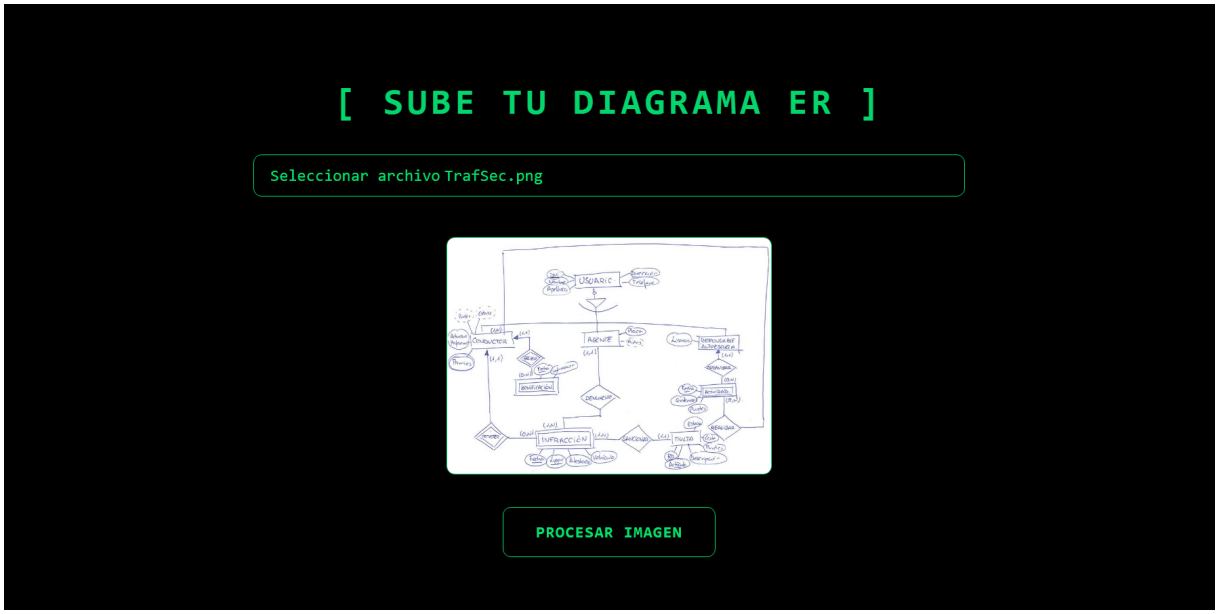

Figura 6.4: Confirmación de imagen cargada.

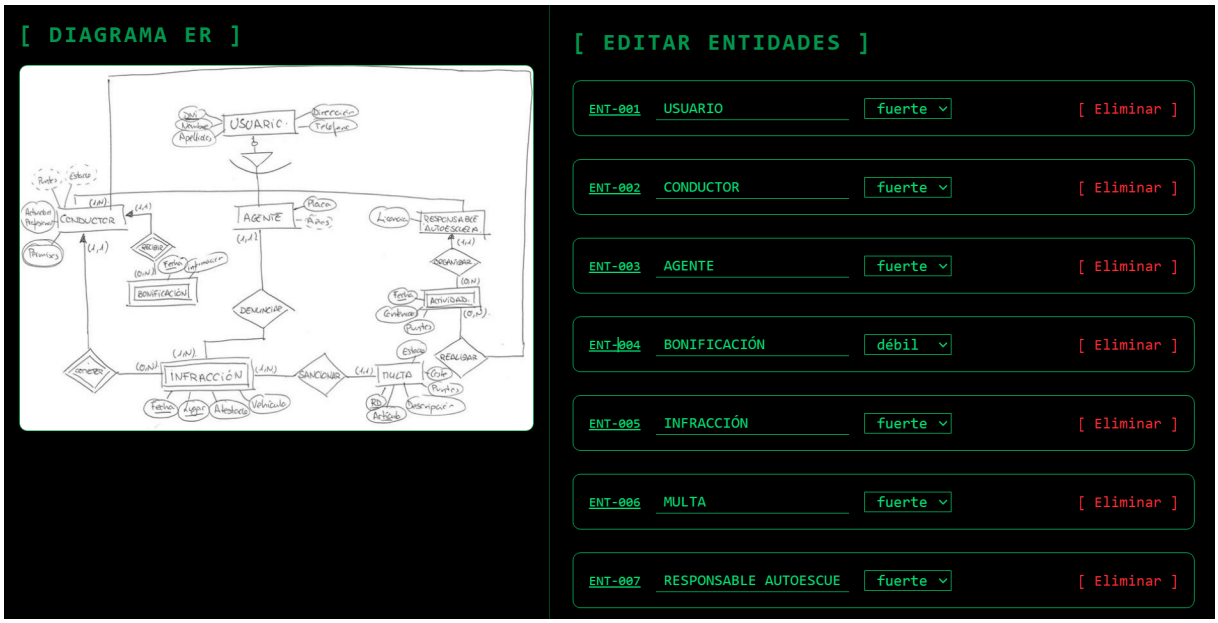

Figura 6.5: Gestión de entidades.

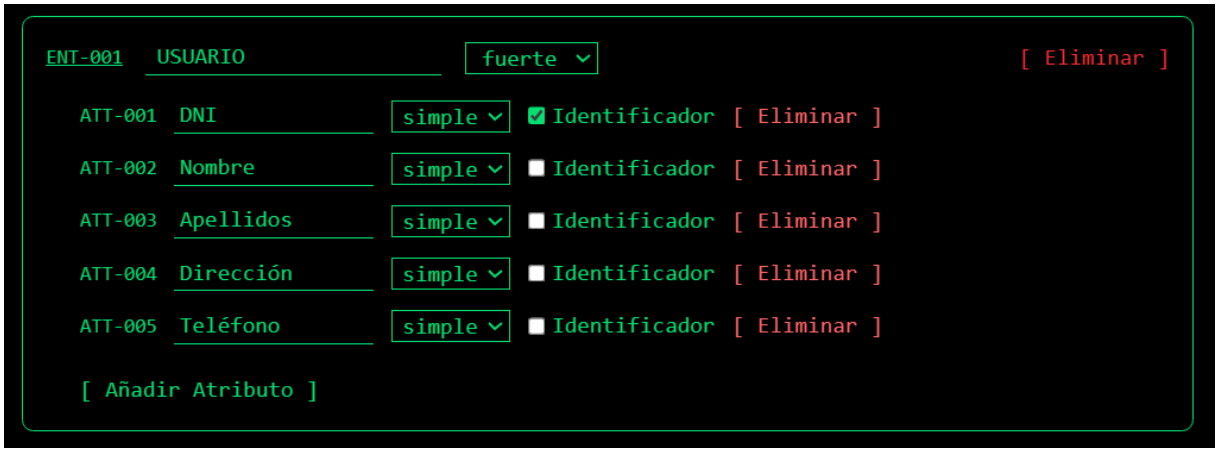

Figura 6.6: Detalles de entidad.

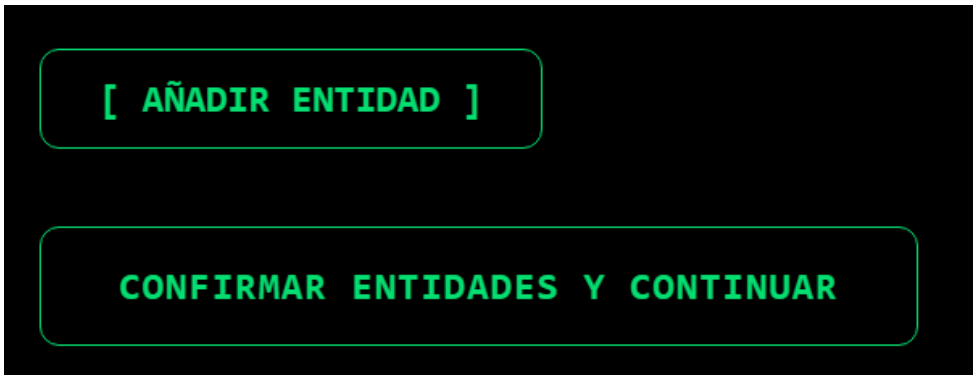

Figura 6.7: Edición y confirmación de entidades.

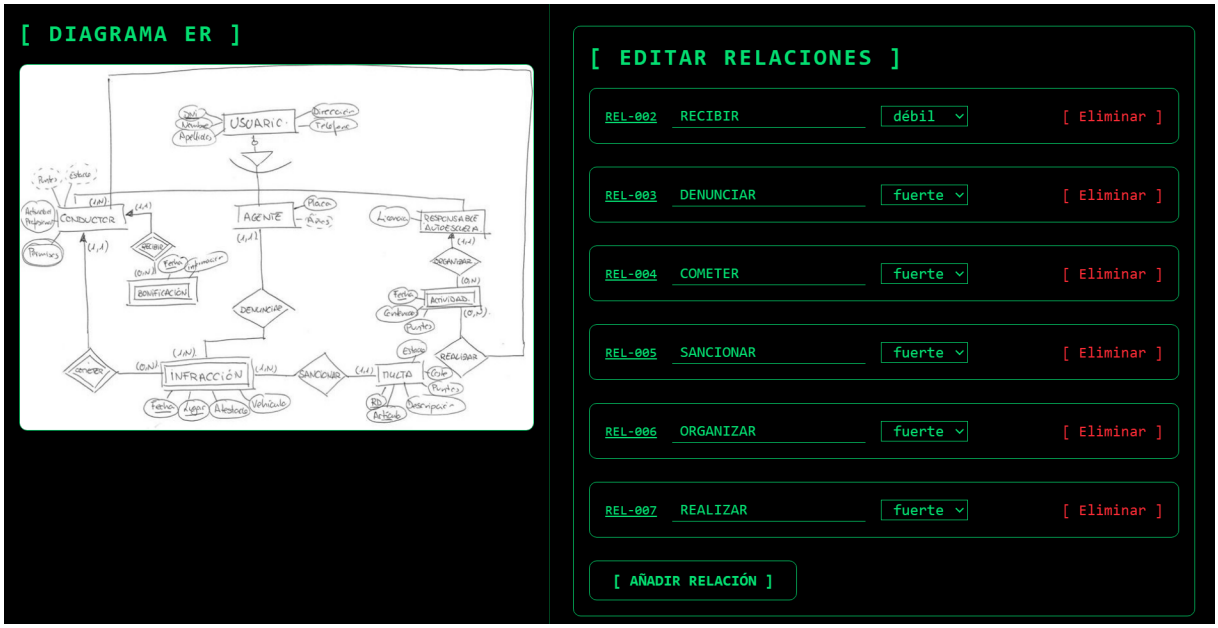

Figura 6.8: Gestión de relaciones.

Figura 6.9: Detalles de relación.

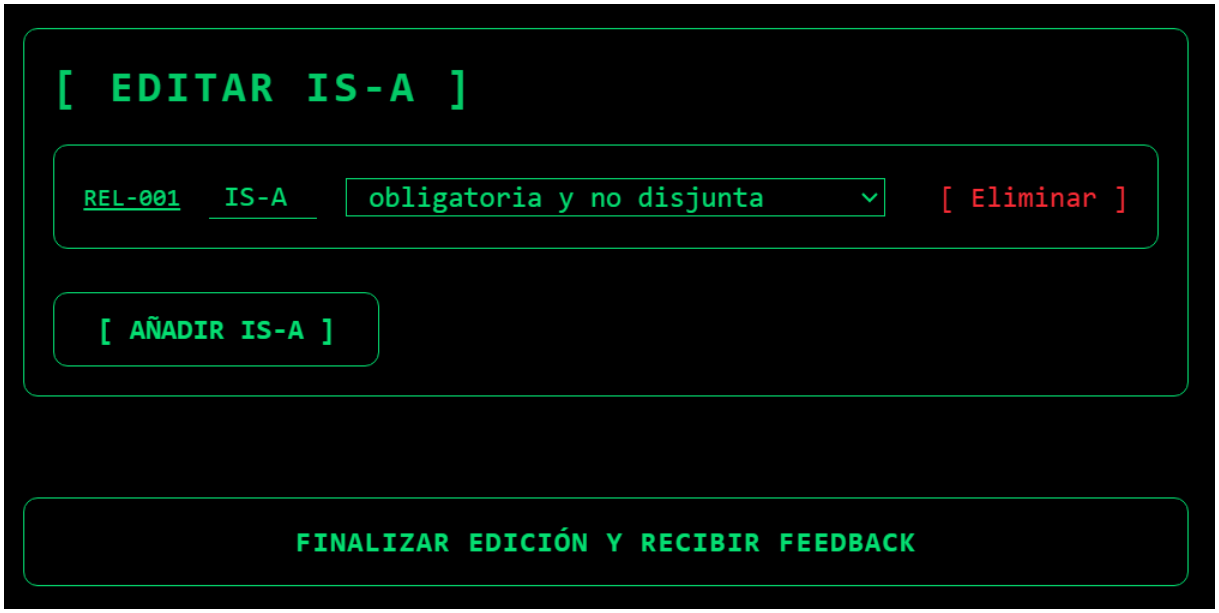

Figura 6.10: Relaciones IS-A.

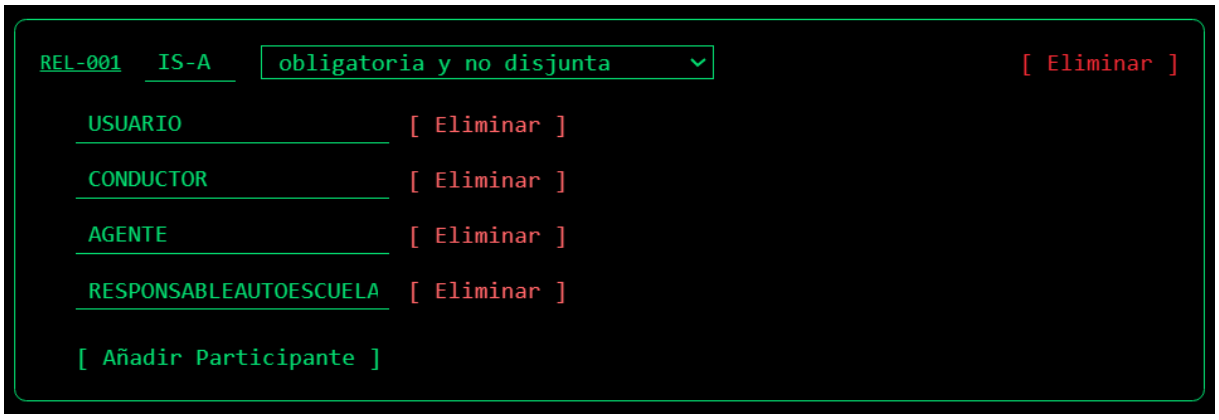

Figura 6.11: Detalles de relación IS-A.

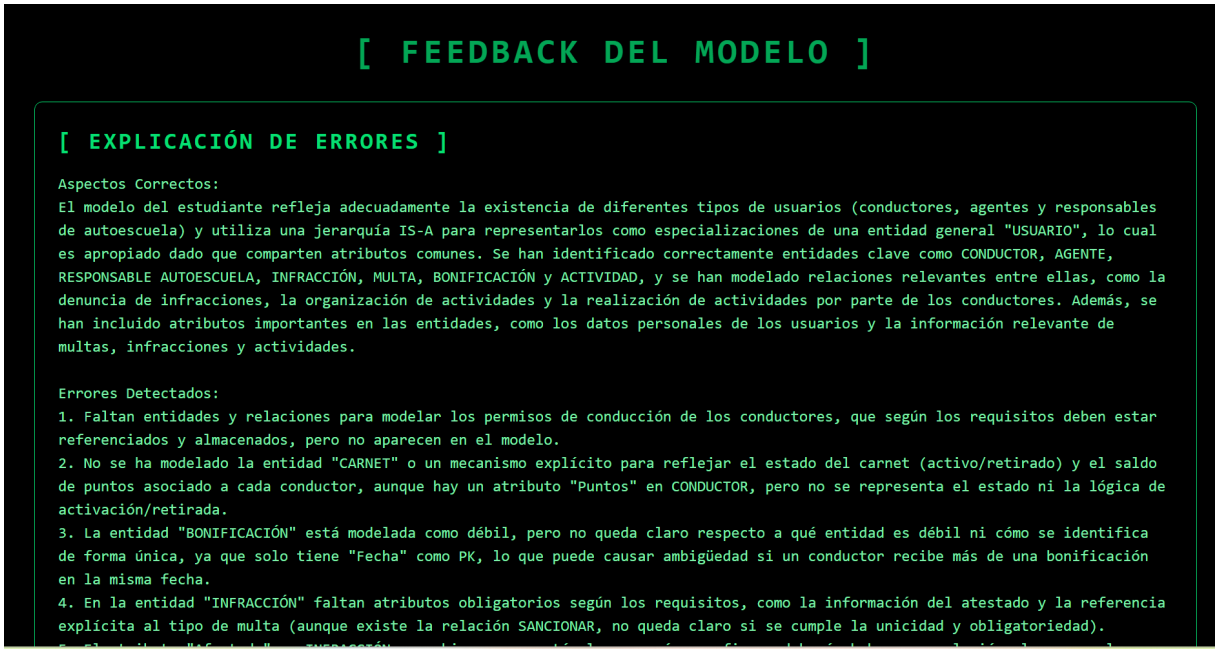

Figura 6.12: *Feedback* cualitativo.

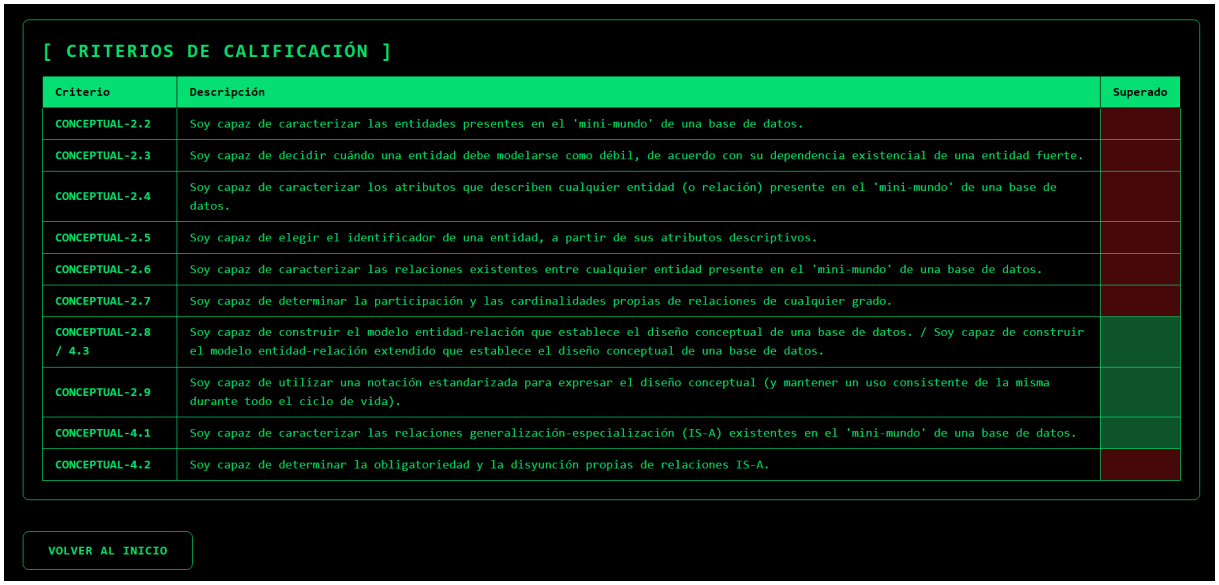

Figura 6.13: *Feedback* cuantitativo.

Capítulo 7

Pruebas

El objetivo de estas pruebas es verificar la robustez del sistema, para ello se pondrá el foco en la gestión de entradas incorrectas en la etapa de carga de archivos por parte del alumno ya que realmente un alumno puede escribir cualquier cosa en su diagrama ER, por muy incorrecta que sea, y la aplicación debe permitirlo. A continuación, se presentan las pruebas que se han llevado a cabo.

P-01	**Subida de un archivo que no es una imagen**
Objetivo	Comprobar que el sistema rechaza archivos que no son imágenes válidas.
Precondición	El usuario accede a la interfaz de subida de diagrama ER.
Pasos	1. Pulsar el botón para seleccionar archivo. 2. Seleccionar un archivo PDF (u otro formato no soportado, por ejemplo .docx). 3. Pulsar "Procesar imagen".
Resultado esperado	El sistema muestra un mensaje de alerta indicando que no se pudo procesar.
Resultado obtenido	CORRECTO (ver Figura 7.1 para ver el mensaje de error).

Tabla 7.1: Especificación de la prueba 1.

Figura 7.1: Alerta que salta en la aplicación cuando se intenta subir una imagen que no corresponde con un diagrama ER o cuando se intenta subir una archivo que no es una imagen.

P-02	**Subida de imagen que no representa un diagrama ER**
Objetivo	Comprobar el comportamiento del sistema ante la subida de una imagen que no contiene un diagrama ER.
Precondición	El usuario accede a la interfaz de subida de diagrama ER.
Pasos	1. Pulsar el botón para seleccionar archivo. 2. Seleccionar una imagen válida (por ejemplo, .jpg o .png) que no sea un diagrama ER. 3. Pulsar "Procesar imagen".
Resultado esperado	El sistema procesa la imagen, pero al identificar que no es un modelo ER lanza una alerta. No se genera un modelo y se ofrece la posibilidad de intentar con otra imagen.
Resultado obtenido	CORRECTO (ver Figura 7.1 para ver el mensaje de error).

Tabla 7.2: Especificación de la prueba 2.

Parte III

Resultados

Capítulo 8

Aceptación

La aceptación de este proyecto se divide en dos partes principales, la aceptación de la parte de la identificación de los elementos del modelo ER y la parte de la aceptación de la evaluación del modelo. Durante este capítulo se tratarán ambas partes con detalle.

8.1. Aceptación de la identificación de elementos del modelo ER del estudiante

Esta parte de la aceptación se encarga de ver cómo de bien es capaz el sistema de identificar las diferentes partes del diagrama ER del alumno incluyendo entidades, relaciones, IS-A, atributos y participantes entre otros.

8.1.1. Métricas

Para evaluar el rendimiento de la aplicación en la identificación de los distintos elementos de un modelo Entidad-Relación se han utilizado métricas ampliamente reconocidas en la evaluación de modelos de clasificación y extracción de información: precisión, exhaustividad (*Recall*) y la métrica F1 (*F1-score*). Estas métricas permiten cuantificar el acierto del sistema tanto en la detección correcta de los elementos esperados como evitando falsos positivos.

Antes de definir las métricas que vamos a utilizar, necesitamos definir lo que son los *True Positives*, los *False Positives* y los *False Negatives*.

- ***True Positives* (TP)** [57]: Elementos detectados correctamente, es decir, presentes tanto en el conjunto de referencia (que en este caso sería la imagen del diagrama ER del alumno) como en el resultado obtenido por el sistema.
- ***False Positives* (FP)** [57]: Elementos identificados por el sistema que en realidad no existen en el conjunto de referencia.
- ***False Negatives* (FN)** [57]: Elementos que existen en el conjunto de referencia pero que el sistema no ha detectado.

A partir de estas cantidades, se calculan las métricas [20] con las que se va a trabajar y que se explican a continuación.

- **Precisión**: proporción de elementos detectados que son realmente correctos.

$$\text{Precisión} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{8.1}$$

- ***Recall*** (Exhaustividad): proporción de elementos relevantes que han sido identificados correctamente.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{8.2}$$

- ***F1-score***: media armónica entre la precisión y la exhaustividad.

$$F1 = 2 \times \frac{\text{Precisión} \times \text{Recall}}{\text{Precisión} + \text{Recall}} \tag{8.3}$$

En el contexto de este trabajo, las métricas anteriores se han adaptado para evaluar la extracción automática de modelos ER. Para cada tipo de elemento (entidades, atributos, relaciones, jerarquías ISA, participantes, etc.) y para cada una de sus propiedades relevantes (por ejemplo, aparición, nombre, tipo, clave primaria o multiplicidad), se calculan de forma individual los valores de TP, FP y FN a partir de los datos recogidos por la aplicación y con ello se calculan las métricas que se acaban de presentar.

Campo	**Significado**
numentidadesdetectadas	Total de entidades identificadas por la aplicación.
entidadesborradas	Entidades identificadas por la aplicación pero que no existen en el modelo (el estudiante las elimina).
entidadesanadidas	Entidades no identificadas por la aplicación pero que existen en el modelo (el estudiante las añade).
nombresentidadesmodificadas	Entidades cuyo nombre se detectó incorrectamente (el estudiante lo corrige).
tiposentidadesmodificadas	Entidades cuyo tipo se detectó incorrectamente (el estudiante lo corrige)
entnumatributosdetectados	Total de atributos de entidades detectados por la aplicación.
entatributosborrados	Atributos de entidades identificados por la aplicación pero que no existen en el modelo (el estudiante los elimina).
entatributosanadidos	Atributos de entidades no identificados por la aplicación pero que existen en el modelo (el estudiante los añade).
entnombreatributosmodificados	Atributos cuyo nombre se detectó incorrectamente (el estudiante lo corrige).
enttiposatributosmodificados	Atributos de entidades cuyo tipo se detectó incorrectamente (el estudiante lo corrige).
entpksatributosmodificados	Atributos designados incorrectamente como identificadores (el estudiante lo corrige).

Tabla 8.1: Significado de los campos asociados a elementos y propiedades de entidades en el modelo ER.

Campo	**Significado**
numrelacionesdetectadas	Total de relaciones detectadas por la aplicación.
relacionesborradas	Relaciones identificadas por la aplicación pero que no existen en el modelo (el estudiante las elimina).
relacionesanadidas	Relaciones no identificadas por la aplicación pero que existen en el modelo (el estudiante las añade).
nombrerelacionesmodificadas	Relaciones cuyo nombre se detectó incorrectamente (el estudiante lo corrige).
tiposrelacionesmodificadas	Relaciones cuyo tipo se detectó incorrectamente (el estudiante lo corrige).
relnumatributosdetectados	Total de atributos de relaciones detectados por la aplicación.
relatributosborrados	Atributos de relaciones identificados por la aplicación pero que no existen en el modelo (el estudiante los elimina).
relatributosanadidos	Atributos de relaciones no identificados por la aplicación pero que existen en el modelo (el estudiante los añade).
relnombreatributosmodificados	Atributos de relaciones cuyo nombre se detectó incorrectamente (el estudiante lo corrige).
reltiposatributosmodificados	Atributos de relaciones cuyo tipo se detectó incorrectamente (el estudiante lo corrige).
relpksatributosmodificados	Número de atributos de relación detectados con PK (clave primaria) incorrecto.
relnumparticipantesdetectados	Total de participantes en relaciones detectados por la aplicación.
relparticipantesborrados	Participantes en relaciones identificadas por la aplicación pero que no existen en el modelo (el estudiante las elimina).
relparticipantesanadidos	Participantes en relaciones no identificadas por la aplicación pero que existen en el modelo (el estudiante las añade).
relnombreparticipantesmodificados	Participantes en relaciones detectados con nombre incorrecto (el estudiante las corrige).
relmultiplicidadesparticipantesmodificadas	Participantes en relaciones identificadas con multiplicidades incorrectos (el estudiante las corrige).

Tabla 8.2: Significado de los campos asociados a elementos y propiedades de relaciones en el modelo ER.

Campo	**Significado**
numISAdetectadas	Total de jerarquías ISA detectadas por la aplicación.
ISAborradas	Relaciones IS-A identificadas por la aplicación pero que no existen en el modelo (el estudiante las elimina).
ISAanadidas	Relaciones IS-A no identificadas por la aplicación pero que existen en el modelo (el estudiante las añade).
tiposISAmodificadas	Número de jerarquías ISA detectadas con tipo incorrecto.
ISAnumparticipantesdetectados	Total de participantes en ISA detectados por la aplicación.
ISAparticipantesborrados	Participantes en relaciones IS-A identificadas por la aplicación pero que no existen en el modelo (el estudiante las elimina).
ISAparticipantesanadidos	Participantes en relaciones IS-A no identificadas por la aplicación pero que existen en el modelo (el estudiante las añade).
ISAnombreparticipantesmodificados	Número de participantes en ISA detectados con nombre incorrecto.

Tabla 8.3: Significado de los campos asociados a relaciones de tipo IS-A (jerarquía) en el modelo ER.

Partiendo de los datos extraídos del sistema presentados en las tablas 8.1 (datos sobre en-

tidades), 8.2 (datos sobre relaciones) y 8.3 (datos sobre relaciones de tipo IS-A), los valores de TP, FP y FN se obtienen de la siguiente manera para cada componente o propiedad del modelo, como se resume en las tablas 8.4 (para entidades), 8.5 (para relaciones) y 8.6 (para relaciones IS-A).

Elemento / Propiedad	**TP**	**FP**	**FN**
Entidades	numentidadesdetectadas − entidadesborradas	entidadesborradas	entidadesanadidas
Nombre de entidad	numentidadesdetectadas − nombresentidadesmodificadas	nombresentidadesmodificadas	nombresentidadesmodificadas
Tipo de entidad	numentidadesdetectadas − tiposentidadesmodificadas	tiposentidadesmodificadas	tiposentidadesmodificadas
Atributos de entidad	entnumatributosdetectados − entatributosborrados	entatributosborrados	entatributosanadidos
Nombre de atributo en entidad	entnumatributosdetectados − entnombreatributosmodificados	entnombreatributosmodificados	entnombreatributosmodificados
Tipo de atributo en entidad	entnumatributosdetectados − enttiposatributosmodificados	enttiposatributosmodificados	enttiposatributosmodificados
PK de atributo en entidad	entnumatributosdetectados − entpksatributosmodificados	entpksatributosmodificados	entpksatributosmodificados

Tabla 8.4: Definición de TP, FP y FN para elementos y propiedades relacionadas con entidades del modelo ER.

Elemento / Propiedad	**TP**	**FP**	**FN**
ISA (Jerarquía)	numISAdetectadas − ISAborradas	ISAborradas	ISAanadidas
Tipo de ISA	numISAdetectadas − tiposISAmodificadas	tiposISAmodificadas	tiposISAmodificadas
Participantes en ISA	ISAnumparticipantesdetectados − ISAparticipantesborrados	ISAparticipantesborrados	ISAparticipantesanadidos
Nombre de participante en ISA	ISAnumparticipantesdetectados − ISAnombreparticipantesmodificados	ISAnombreparticipantesmodificados	ISAnombreparticipantesmodificados

Tabla 8.6: Definición de TP, FP y FN para elementos y propiedades relacionadas con relaciones de tipo IS-A en el modelo ER.

En el caso de las propiedades de los elementos (por ejemplo, el nombre o tipo de una entidad), cada error de extracción se cuenta tanto como falso positivo como falso negativo, siguiendo la práctica habitual en aceptación de tareas de extracción de información como es este caso.

8.1.2. Análisis de los resultados

Se han realizado pruebas sobre 11 modelos ER (10 procedentes de exámenes de 2024 y uno adicional de un ejercicio externo), cuyos resultados se detallan en las tablas 8.7, 8.8 y 8.9.

Elemento / Propiedad	**TP**	**FP**	**FN**
Relaciones	numrelacionesdetectadas − relacionesborradas	relacionesborradas	relacionesanadidas
Nombre de relación	numrelacionesdetectadas − nombrerelacionesmodificadas	nombrerelacionesmodificadas	nombrerelacionesmodificadas
Tipo de relación	numrelacionesdetectadas − tiposrelacionesmodificadas	tiposrelacionesmodificadas	tiposrelacionesmodificadas
Atributos en relaciones	relnumatributosdetectados − relatributosborrados	relatributosborrados	relatributosanadidos
Nombre de atributo en relación	relnumatributosdetectados − relnombreatributosmodificados	relnombreatributosmodificados	relnombreatributosmodificados
Tipo de atributo en relación	relnumatributosdetectados − reltiposatributosmodificados	reltiposatributosmodificados	reltiposatributosmodificados
PK de atributo en relación	relnumatributosdetectados − relpksatributosmodificados	relpksatributosmodificados	relpksatributosmodificados
Participantes en relaciones	relnumparticipantesdetectados − relparticipantesborrados	relparticipantesborrados	relparticipantesanadidos
Nombre de participante en relación	relnumparticipantesdetectados − relnombreparticipantesmodificados	relnombreparticipantesmodificados	relnombreparticipantesmodificados
Multiplicidad de participante en relación	relnumparticipantesdetectados − relmultiplicidadesparticipantesmodificadas	relmultiplicidadesparticipantesmodificadas	relmultiplicidadesparticipantesmodificadas

Tabla 8.5: Definición de TP, FP y FN para elementos y propiedades relacionadas con relaciones del modelo ER.

Elemento	**Propiedad**	**TP**	**FP**	**FN**	**Precisión**	**Recall**	**F1**
Entidad	**Presencia**	133	6	2	0.9568	0.9852	0.9708
	Nombre	138	1	1	0.9928	0.9928	0.9928
	Tipo	116	23	23	0.8345	0.8345	0.8345
Atributo en entidad	**Presencia**	342	7	23	0.9799	0.9370	0.9580
	Nombre	335	14	14	0.9599	0.9599	0.9599
	Tipo	341	8	8	0.9771	0.9771	0.9771
	PK	316	33	33	0.9054	0.9054	0.9054

Tabla 8.7: Métricas para Entidades

Elemento	Propiedad	TP	FP	FN	Precisión	Recall	F1
Relación	**Presencia**	115	2	1	0.9829	0.9914	0.9871
	Nombre	116	1	1	0.9915	0.9915	0.9915
	Tipo	99	18	18	0.8462	0.8462	0.8462
Atributo en relación	**Presencia**	12	2	9	0.8571	0.5714	0.6857
	Nombre	9	5	5	0.6429	0.6429	0.6429
	Tipo	14	0	0	1.0000	1.0000	1.0000
	PK	13	1	1	0.9286	0.9286	0.9286
Participante en relación	**Presencia**	236	0	1	1.0000	0.9958	0.9979
	Nombre	224	12	12	0.9492	0.9492	0.9492
	Multiplicidad	207	29	29	0.8771	0.8771	0.8771

Tabla 8.8: Métricas para Relaciones

Elemento	Propiedad	TP	FP	FN	Precisión	Recall	F1
ISA	**Presencia**	18	0	4	1.0000	0.8182	0.9000
	Tipo	7	11	11	0.3889	0.3889	0.3889
Participante en ISA	**Presencia**	61	0	0	1.0000	1.0000	1.0000
	Nombre	61	0	0	1.0000	1.0000	1.0000

Tabla 8.9: Métricas para IS-A

A continuación se presenta un análisis crítico de las métricas obtenidas. En el análisis se ha utilizado solo el F1 porque combina precisión y *recall* en un único valor, lo que ayuda a ver de un vistazo cómo de bien funciona el sistema en general, sin tener que repasar varias cifras cada vez. Además, esto facilita comparar diferentes aspectos aunque tengan distintos tipos de errores, y así la lectura del análisis resulta más clara y directa.

- **Entidades:** La detección de entidades presenta unos resultados muy competitivos. El sistema acierta de forma generalizada cuando tiene que decidir si una entidad está presente (**F1 de 0.9708**) y también con los nombres (**F1 de 0.9928**), lo que muestra que es bastante fiable en estas tareas. Sin embargo, al clasificar el tipo de entidad (fuerte/débil), el rendimiento baja (**F1 de 0.8345**). Esto suele pasar porque distinguir entre tipos de entidades depende mucho de detalles gráficos, como el doble contorno, que pueden perderse si el diagrama está dibujado de forma irregular, y ahí el modelo tiende a fiarse más del contexto textual, que a veces no ayuda.
- **Atributos en entidad:** El sistema reconoce bien los atributos dentro de las entidades (**F1 de 0.9580**) y también es bastante bueno identificando sus nombres y tipos (**F1** por encima de 0.95). El mayor reto aparece con las claves primarias (**F1 de 0.9054**), donde aún se pueden mejorar resultados. Esto se debe a que un atributo es identificador si está

subrayado, este detalle puede pasar desapercibido si la línea no es muy clara o si el diagrama ER es demasiado grande.

- **Relaciones:** La detección de las relaciones y sus nombres presenta un comportamiento muy competitivo (**F1 de 0.9871 y 0.9915**), aunque la efectividad se reduce ligeramente a la hora de clasificar el tipo de relación (**F1 de 0.8462**). Muchas veces esto se debe a que la diferencia entre tipos de relación depende de detalles como el doble contorno del rombo, detalles que se pueden perder si el diagrama no es muy claro. Además, los atributos que pertenecen a las relaciones son una de las partes más flojas, tanto en presencia como en nombre (**F1 de 0.6857 y 0.6429**). Aquí influye que suelen estar en los márgenes, conectados con líneas muy finas que se confunden con facilidad, y si la imagen es densa o no tiene buena calidad, el modelo puede ignorarlos o confundirlos con otros elementos. En muchas de las ocasiones en las que no se identifican los atributos de una relación es por que el sistema suele ponerlos como atributos de entidades cercanas. También es cierto que hay pocos atributos de relación en los ejemplos, así que le cuesta más generalizar.

- **Participantes en relación:** La parte de identificar participantes en las relaciones la hace prácticamente perfecta (**F1 de 0.9979**), pero la cosa cambia con la multiplicidad (**F1 de 0.8771**). La razón es que las cardinalidades se suelen poner en letra pequeña y pegadas a las líneas; si están muy juntas o poco claras, el modelo puede no saber a qué participante se refiere cada número.

- **IS-A:** En las jerarquías IS-A, el sistema detecta bien su presencia (**F1 de 0.9000**), pero falla mucho al identificar el tipo (**F1 de 0.3889**). Este problema ocurre porque las pistas para distinguir entre tipos de IS-A suelen ser gráficas (círculo, arcos, etc.) y no siempre están dibujadas igual o con claridad en los diagramas de los estudiantes, así que el modelo no termina de aprenderlas bien.

- **Participantes en IS-A:** Aquí no hay ningún problema: el sistema acierta siempre (**F1 de 1.0000**) tanto en presencia como en nombre, así que identifica perfectamente las entidades que participan en jerarquías.

En definitiva, el sistema destaca en la detección de entidades y relaciones, así como en sus nombre. Los puntos que peor lleva son la clasificación de tipos (en entidades, relaciones e IS-A), los atributos dentro de relaciones y la asignación de multiplicidades. Estos errores suelen deberse a que el modelo de IA, como GPT-4.1, que es el que usa el sistema para realizar esta tarea, tienden a fijarse más en el texto que en detalles gráficos, y es sensible a cómo de limpios o claros estén dibujados los diagramas.

8.2. Aceptación de la evaluación cuantitativa del modelo

Para hacer la aceptación de la evaluación cuantitativa del modelo, es decir, ver cómo de bien hace el *feedback* cuantitativo, el sistema, la metodología usada ha sido la siguiente: se comenzó haciendo pruebas con 10 modelos ER correspondientes a soluciones del examen de 2024 de alumnos de la asignatura de Sistemas de Bases de Datos del Grado en Ingeniería Informática de Servicios y Aplicaciones, impartida en la Universidad de Valladolid. Una vez obtenidas las notas de cada criterio para cada uno de los estudiantes, se guardan en un Excel y se comparan

con las notas que puso el profesor de la asignatura en su momento. Se comenzó realizando la aceptación para cuando se utilizaba el modelo GPT-4.1 para hacer la comparación de modelos necesaria para la obtención del *feedback* cuantitativo, pero se observó que los resultados no eran demasiado buenos. Por tanto, tras una investigación exhaustiva de los modelos disponibles en la API de OpenAI se decidió trabajar con el modelo de razonamiento o3. En las imágenes 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8 y 8.9 se muestran varias gráficas que comparan los resultados obtenidos en la corrección realizada por el profesor con los proporcionados por la aplicación, utilizando dos modelos distintos (GPT 4.1 y o3). En cada figura, la gráfica de la izquierda corresponde al modelo GPT 4.1 y la de la derecha al modelo o3. De esta forma es más sencillo comparar los modelos y ver cual de ellos es más preciso en esta tarea.

Una visión general de estas gráficas, nos lleva a que el modelo o3 hace la tarea en general mejor que el GPT-4.1 por lo que finalmente es el que se ha mantenido en la aplicación. Por tanto, a partir de ahora nos centraremos en analizar únicamente los resultados obtenidos utilizando el modelo o3.

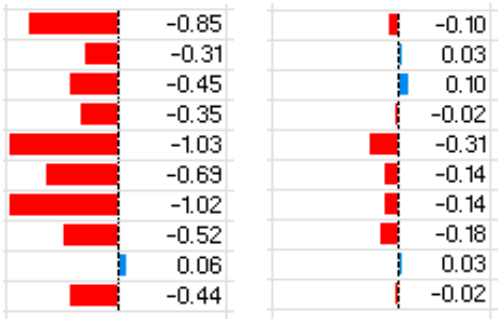

Figura 8.1: Comparativa entre modelos para los criterios 2.2 y 2.3 (Entidades fuertes y Entidades débiles).

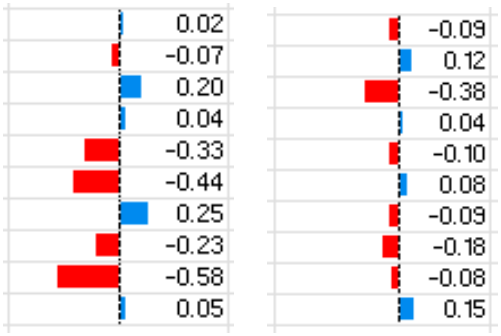

Figura 8.2: Comparativa entre modelos para el criterio 2.4 (Atributos).

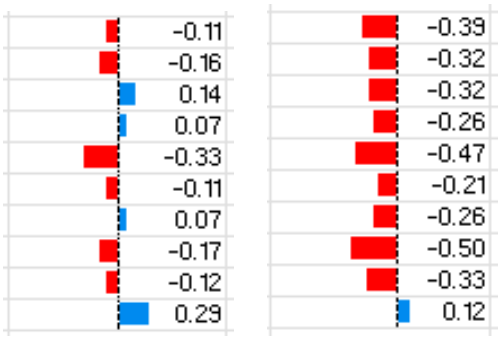

Figura 8.3: Comparativa entre modelos para el criterio 2.5 (Identificadores).

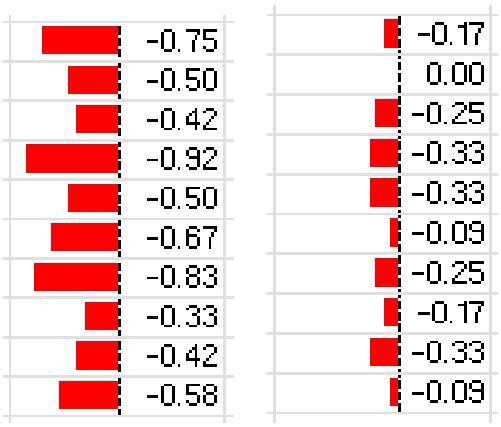

Figura 8.4: Comparativa entre modelos para el criterio 2.6 (Relaciones).

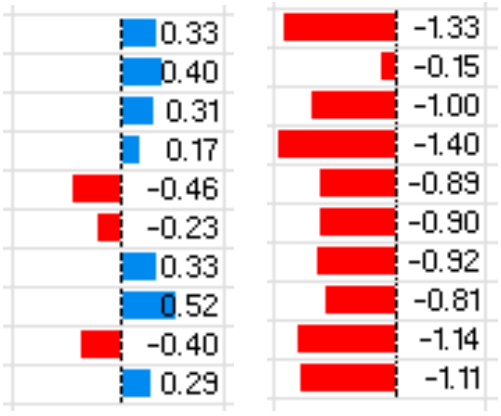

Figura 8.5: Comparativa entre modelos para el criterio 2.7 (Multiplicidades).

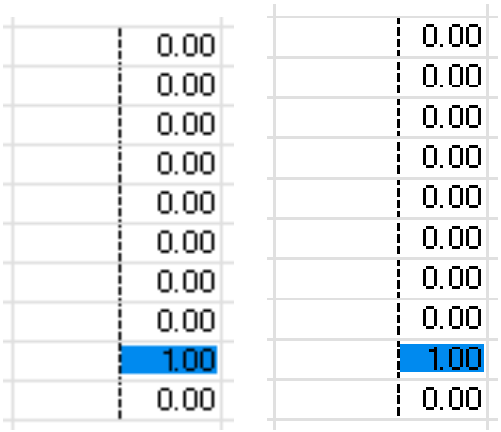

Figura 8.6: Comparativa entre modelos para el criterio 2.8/4.3 (Modelo y obligatoriedad/disyunción IS-A).

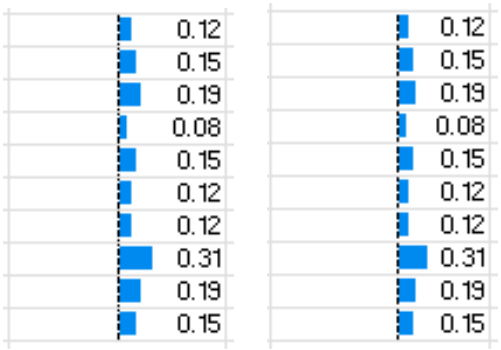

Figura 8.7: Comparativa entre modelos para el criterio 2.9 (Notación).

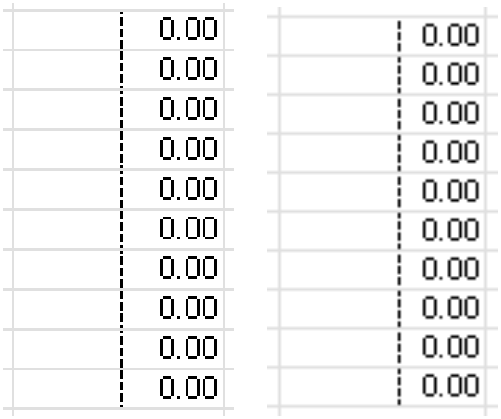

Figura 8.8: Comparativa entre modelos para el criterio 4.1 (Relaciones IS-A).

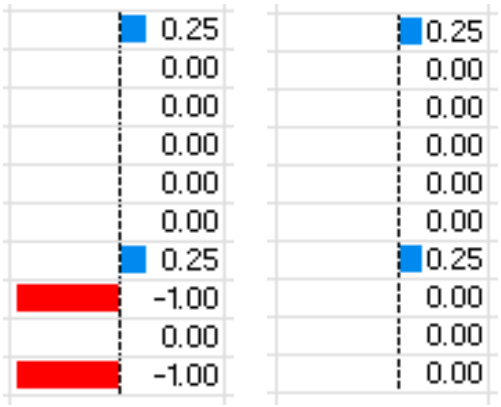

Figura 8.9: Comparativa entre modelos para el criterio 4.2 (Obligatoriedad y disyunción de relaciones IS-A).

A partir de ahora nos centraremos en analizar únicamente los resultados obtenidos utilizando el modelo o3.

8.2.1. Métricas

Para analizar las diferencias entre los resultados obtenidos por el profesor en la evaluación de modelos y los resultados obtenidos por LearnER se utilizarán las métricas que se presentan a continuación.

- **Media** (μ) [53].

$$\mu = \frac{1}{n}\sum_{i=1}^{n} e_i,$$

  donde e_i es la diferencia entre ambas notas para el diagrama i. Detecta el sesgo: los valores positivos indican que el sistema tiende a sobre puntuar y los valores negativos indican que el sistema tiende a infra-puntuar.

- **Error absoluto medio (MAE)** [62].

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |e_i|.$$

  Elimina el signo del error para medir el tamaño medio de las diferencias, evitando cancelaciones entre sobre-estimaciones y sub-estimaciones.

- **Varianza** (σ^2) [16].

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(e_i - \mu)^2.$$

  Evalúa la dispersión de los errores alrededor de la media. Un σ^2 pequeño indica que casi todos los e_i se agrupan cerca de μ, lo que implica que el sistema se comporta con regularidad—aunque la media fuese distinta de cero. Por el contrario, un valor alto revela que

algunos diagramas reciben calificaciones muy alejadas del patrón general, es decir, la herramienta resulta impredecible. Su raíz cuadrada (σ)—la desviación típica—se expresa en las mismas unidades que la nota, lo que facilita comunicar al lector cuántos puntos "suele" desviarse el sistema respecto al profesor.

8.2.2. Análisis de los resultados

En la Tabla 8.10 se presentan los resultados obtenidos a partir de los datos y, posteriormente, se hace un análisis detallado.

Criterio	**Media**	**MAE**	**Varianza**
CONCEPTUAL-2.2+2.3	−0,074	0,105	0,013
CONCEPTUAL-2.4	−0,053	0,129	0,022
CONCEPTUAL-2.5	−0,294	0,317	0,026
CONCEPTUAL-2.6	−0,200	0,200	0,013
CONCEPTUAL-2.7	−0,964	0,964	0,106
CONCEPTUAL-2.8	0,100	0,100	0,090
CONCEPTUAL-2.9 / 4.3	0,158	0,158	0,004
CONCEPTUAL-4.1	0,000	0,000	0,000
CONCEPTUAL-4.2	0,050	0,050	0,010

Tabla 8.10: Media, error absoluto medio y varianza de las diferencias de las notas.

Análisis por métricas

En cuanto a la media de las diferencias, el sistema coincide prácticamente con el profesor en los criterios 4.1 y 4.2, donde el sesgo se reduce a cero o a apenas cinco centésimas de sobrepuntuación. Los criterios fusionados 2.2 + 2.3 y el criterio 2.4 presentan desviaciones pequeñas, del orden de una décima negativa sobre la escala de uno. En 2.5 y 2.6 la discrepancia se intensifica: el sistema resta de media entre dos y tres décimas. El caso más marcado es 2.7, donde la media alcanza casi un punto completo por debajo de la valoración humana, lo que indica una infra-puntuación constante. En sentido opuesto, 2.8 y el bloque 2.9 / 4.3 muestran ligero sesgo positivo, alrededor de una décima y media, señalando una tendencia a añadir nota.

Si se observa el error absoluto medio, puede verse que 4.1 y 4.2 presentan una coincidencia casi perfecta, pues el MAE es nulo o de sólo cinco centésimas. Los bloques 2.2 + 2.3 y 2.4 se mantienen en torno a una décima, mientras que 2.5 y 2.6 ascienden a tres y dos décimas respectivamente, reflejando una discrepancia que el profesor notaría con frecuencia. El criterio 2.7 vuelve a destacar: el sistema se desvía de la nota del docente en prácticamente un punto en cada diagrama, lo que convierte a este apartado en el de mayor distancia absoluta. Por último, 2.8 y 2.9 / 4.3 presentan errores medios idénticos a su sesgo, confirmando que sus diferencias, aun siendo positivas, son uniformes.

Respecto a la varianza, los errores en 4.1 son completamente uniformes, mientras que en 2.9 / 4.3 y en 2.2 + 2.3 la dispersión es mínima y los valores se agrupan muy cerca de la media. 2.6

comparte esa regularidad pese a su sesgo negativo. En los criterios 2.4 y 2.5 la varianza crece, indicando que se alternan desviaciones más dispersas alrededor de sus respectivas medias. La mayor dispersión aparece otra vez en 2.7, donde la variabilidad alcanza décimas completas, y en menor medida en 2.8; ello indica que, además de la diferencia sistemática, los errores son muy heterogéneos dentro de estos criterios.

Análisis por criterio

Durante este análisis, muchas veces se justificarán los errores hablando de la IA; esto se debe a que, como se comentó, la evaluación de los modelos se hace utilizando inteligencia artificial generativa.

- **CONCEPTUAL-2.2 + 2.3:** este criterio evalúa la capacidad de caracterizar correctamente las entidades presentes en el "mini-mundo" de la base de datos y decidir cuándo una entidad debe modelarse como fuerte o como débil. El sesgo medio es de $-0{,}074$, el error absoluto medio es 0,105 y la varianza 0,013. El sistema tiende a restar ligeramente nota respecto al profesor y lo hace de manera bastante consistente. Esta diferencia podría deberse a que para la la IA el tipo de una entidad (fuerte o débil) del alumno debe coincidir con el tipo de la entidad del profesor, en caso contrario determinará que el tipo de la entidad está mal. Sin embargo, el profesor corrige teniendo en cuenta la visión global del modelo, sabiendo que en ciertas ocasiones, el alumno puede tener un tipo diferente para cierta entidad y que aún así este correcto.

- **CONCEPTUAL-2.4:** este criterio evalúa la capacidad de identificar los atributos que describen entidades o relaciones. El sesgo es de $-0{,}053$, el MAE 0,129 y la varianza 0,022. El sistema presenta un leve sesgo negativo con un error algo mayor que en el criterio anterior. Esto podría estar relacionado con una menor tolerancia del modelo ante sinónimos o formas alternativas de expresar atributos que el profesor sí acepta.

- **CONCEPTUAL-2.5:** este criterio evalúa la elección del identificador adecuado de una entidad. El sesgo es notablemente negativo ($-0{,}294$), con un MAE de 0,317 y varianza 0,026. La IA hace una sub-puntuación de este criterio, esto es producto tanto de los resultados de los criterios 2.2+2.3 como del de 2.4. En primer lugar, en muchas ocasiones una entidad débil puede convertirse en fuerte si le añadimos un id propio, por tanto, que el alumno haya puesto un tipo de entidad diferente al profesor, aunque esté correcto podría hacer que el identificador fuera correcto. Por otro lado, si la IAG no ha detectado correctamente el nombre de un atributo que es identificador, entonces no solo pondrá que el atributo está mal sino también que el identificador está mal. Otra razón es, que para una misma entidad, podría haber diferentes identificadores válidos (por ejemplo, para una entidad con DNI y Número de la seguridad social, cualquiera de los dos puede servir como identificador), pero si el alumno elige un identificador diferente al profesor, es posible que se evlue mal.

- **CONCEPTUAL-2.6:** este criterio evalúa la caracterización de las relaciones existentes entre entidades. El sesgo es de $-0{,}200$, el MAE 0,200 y la varianza 0,013. La infrapuntuación es sistemática, pero la baja varianza indica que el comportamiento del sistema es consistente. Es posible que la IA sea más estricta al aceptar relaciones (en gran parte por lo que se comentó para las entidades sobre los tipos en el CONCEPTUAL-2.2+2.3). Además,

suele haber mucha diferencia entre los nombres que den a las relaciones distintas personas, pudiendo estar todos bien. Esto dificulta notablemente el trabajo de la IA para ver si una entidad es correcta o no.

- **CONCEPTUAL-2.7:** este criterio evalúa la determinación correcta de las participaciones y cardinalidades de las relaciones. Los resultados son los peores de todo el conjunto: el sesgo es de $-0{,}964$, el MAE 0,964 y la varianza 0,106. Esto indica una infrapuntuación casi total y un comportamiento muy irregular. La causa probable es la dificultad de los modelos de lenguaje para interpretar correctamente cardinalidades representadas numéricamente, ya que estas se alejan del tipo de información textual que domina la IA. Además, hay numerosas multiplicidades y en muchos casos no hay solo una opción correcta: el profesor evalúa viendo si el resultado del alumno es correcto mientras que la IA lo hace comparando ambos ejercicios.
- **CONCEPTUAL-2.8:** este criterio evalúa la capacidad de construir el modelo entidad-relación completo que define el diseño conceptual. El sesgo es positivo (0,100), el MAE 0,100 y la varianza relativamente alta (0,090). El sistema tiende a otorgar un poco más de nota de la que concede el profesor, y lo hace de forma variable. Esto puede deberse a que el modelo premia aspectos generales como la consistencia global o el orden del diagrama, incluso en casos donde el contenido presenta algunas pequeñas carencias.
- **CONCEPTUAL-2.9 / 4.3:** este criterio evalúa el uso de notación estandarizada y la construcción del modelo E-R extendido. El sesgo es de 0,158, el MAE 0,158 y la varianza muy baja (0,004). El sistema sobrepuntuó de manera uniforme, lo que sugiere que tiene un umbral de reconocimiento de notación más laxo que el profesor.
- **CONCEPTUAL-4.1:** este criterio evalúa este criterio evalúa la caracterización de las relaciones IS-A. El sistema y el profesor coincidieron plenamente, con sesgo, MAE y varianza nulos. Esto indica que la aplicación evalúa de forma idéntica al profesor este aspecto.
- **CONCEPTUAL-4.2:** este criterio evalúa la correcta determinación de la obligatoriedad y disyunción en relaciones IS-A. El sesgo es de 0,050, el MAE 0,050 y la varianza 0,010. El sistema tiende a conceder ligeramente más nota que el profesor y lo hace con una dispersión reducida. Esto podría deberse a una interpretación más generosa por parte de la IA en situaciones ambiguas. Igualmente, estos resultados son muy bajos lo que indica que el sistema es muy preciso evaluando este aspecto.

En conjunto, la herramienta reproduce con notable fidelidad la evaluación del profesor en la mayoría de los criterios. Los criterios 4.1, 4.2, 2.2+2.3 y 2.4 exhiben sesgos y errores absolutos del orden de una décima o menos, con una dispersión muy baja, lo que confirma un comportamiento coherente y prácticamente indistinguible de la corrección manual. Los criterios 2.9 / 4.3 y 2.6 presentan discrepancias algo mayores, pero todavía razonables: la desviación media ronda las dos décimas y la variabilidad es contenida. El desempeño se degrada de forma apreciable en 2.5 y, sobre todo, en 2.7. En el primero, el sistema resta casi un tercio de punto por diagrama y el error absoluto medio supera las tres décimas; en el segundo, la infra-puntuación media es muy alta y va acompañada de la mayor varianza, lo que evidencia un comportamiento muy irregular.

Capítulo 9

Conclusiones y trabajo futuro

En este capítulo se presentan tanto las conclusiones del proyecto y como las conclusiones personales (Sección 9.1). También, se describen diferentes líneas abiertas que se podrían estudiar en un futuro para mejorar y/o ampliar la aplicación (Sección 9.2).

9.1. Conclusiones

En esta sección se presentan, en primer lugar, las conclusiones del proyecto, explicando su utilidad de cara al futuro. Además, se analiza si se han alcanzado los objetivos y se incluye una breve reflexión sobre el papel que ha desempeñado la metodología ASAPen el desarrollo de este trabajo (Sección 9.1.1). Después, se presenta la perspectiva personal en la que se reflexiona sobre el desarrollo y aprendizajes proyecto en general (Sección 9.1.2).

9.1.1. Perspectiva del proyecto

- **Objetivos específicos**: A continuación, se va a comprobar que efectivamente los 4 objetivos definidos al inicio del proyecto se han cumplido. El primero de ellos (OBJ-01) consistía en interpretar modelos entidad-relación manuscritos utilizando IA Generativa, en la Sección 8.1 se comprobó que este objetivo estaba superado satisfactoriamente utilizando diferentes métricas. Los objetivos dos y tres (OBJ-02 y OBJ-03) en la práctica van juntos, pues ambos tienen como finalidad la generación del *feedback* cuantitativo. En la Sección 8.2, se ha probado que ambos objetivos efectivamente se han cumplido con éxito. El último de los objetivos (OBJ-04), buscaba generar un *feedback* cualitativo que ayudara al estudiante a comprender sus errores y le diera sugerencias de mejora. Aunque la aceptación de esta funcionalidad con estudiantes quedaba fuera del alcance del proyecto, durante las diferentes pruebas se ha comprobado que efectivamente los consejos que se le dan al usuario son útiles y realmente sí que le ayudan a mejorar sus habilidades para construir diagramas ER.

- **Utilidad para el futuro**: Esta aplicación puede resultar realmente útil en entornos reales, siendo una herramienta muy poderosa para los estudiantes que deseen mejorar su capacidad de creación de diagramas ER. Además, OpenAI publica modelos cada vez más poderosos luego, simplemente con mínimos cambios en las llamadas a la API podría aumentar notablemente la precisión tanto de la identificación de los componentes del diagrama como de la generación de ambos tipos de *feedback*. En resumen, la herramienta es útil actualmente,

pero tendrá muy buen envejecimiento, pues la IA generativa mejora exponencialmente, y con ella, la aplicación

- **Desarrollo del proyecto/metodología de trabajo:** La implementación de la metodología Asapresultó fundamental para el desarrollo exitoso de este proyecto. Esta metodología permitió mantener una dinámica de trabajo estable y continua, además de facilitar la detección y resolución temprana de errores a medida que aparecían. Gracias a esto, fue posible avanzar en el proyecto sin enfrentar interrupciones importantes.

9.1.2. Perspectiva personal

El desarrollo de este proyecto ha supuesto numerosos aprendizajes y también algunos contratiempos, que he tenido que resolver bajo cierta presión. Entre los aspectos positivos, destaco que me ha permitido comprender lo que implica llevar a cabo un proyecto real, así como la importancia de las metodologías ágiles en la dinámica de trabajo. Asimismo, he adquirido conocimientos sobre la inteligencia artificial generativa, un campo que no se aborda en el plan de estudios del grado pero que, en mi opinión, tendrá un papel crucial en el futuro. Considero que la capacidad para crear *prompts* precisos será muy útil para mi desarrollo profesional.

No obstante, también he enfrentado dificultades que, en ocasiones, han empañado los logros alcanzados. Por ejemplo, durante la fase de aceptación, detecté que la aplicación no evaluaba correctamente los ejercicios, lo que me obligó a modificar tanto el *prompt* como el modelo utilizado. Otro reto fue la visualización de la aplicación con el proyector, lo que me llevó a adaptar toda la paleta de colores de la interfaz para la presentación del TFG (aunque la versión final mantiene los colores originales).

En conclusión, considero que los aspectos positivos han superado a los negativos y, tras la realización de este trabajo, que en general he disfrutado mucho, me siento mucho mejor preparada para afrontar los retos que me deparará mi futura vida profesional.

9.2. Trabajo futuro

En el marco de este proyecto, desarrollamos una aplicación capaz de corregir el modelo ER elaborado a mano por un estudiante ofreciendo retroalimentación cualitativa y cuantitativa de forma instantánea. Si bien hemos alcanzado los cuatro objetivos planteados al inicio del proyecto, todavía existen diversos puntos que pueden mejorarse. A continuación, señalamos las principales líneas de trabajo que permanecen abiertas:

1. Añadir al profesor como actor de la aplicación, permitiéndole subir las soluciones y enunciados de los ejercicios y visualizar el progreso de sus alumnos.

2. Almacenar todos los datos que a día de hoy se almacenan en local o no se almacenan en una base de datos.

3. Añadir la funcionalidad de ayudar al profesor a corregir exámenes, permitiendo al profesor subir todos los diagramas a la vez y generando un archivo descargable con los resultados. Aunque la Unión Europea prohibe este tipo de herramientas para corregir directamente los exámenes, podría serle de ayuda para corroborar sus resultados.

4. Mejorar los resultados de la evaluación de los modelos.

5. Tras haber comprobado que la IA generativa hace un buen trabajo con los diagramas ER, no es descabellado querer ampliar el alcance de la aplicación a otros tipos de diagramas *software* como diagramas de clases o diagramas de casos de uso.

6. Incluir un chat en la aplicación que, utilizando IA generativa, pueda contestar a dudas del estudiante sobre diagramas ER en general o sobre el suyo en particular.

7. Añadir opciones como el modo daltónio o el modo para sordos a la aplicación para que sea más inclusiva y pueda llegar a más alumnos.

Parte IV

Apéndices

Apéndice A

Manual de Instalación

A continuación, se presenta un manual de instalación detallado para desplegar y ejecutar la aplicación Learner desarrollada como parte de este Trabajo de Fin de Grado. El objetivo de este manual es guiar al usuario a través de todos los pasos necesarios, desde la instalación de las herramientas requeridas hasta la ejecución final de la aplicación, de manera que cualquier persona con conocimientos básicos de informática pueda completar el proceso satisfactoriamente.

A.1. Requisitos previos

Para poder instalar y ejecutar correctamente la aplicación, es necesario disponer de las siguientes herramientas:

A.1.1. Python

La aplicación requiere una versión de Python comprendida entre la 3.10.x y la 3.12.x.

- Puedes descargar Python desde la página oficial: `https://www.python.org/downloads/`

Durante la instalación de Python, asegúrate de marcar la opción "*Add Python to enviroment variables*" (o "Agregar Python a las variables de entorno"), como se muestra en la Figura A.1. Esto permitirá que Python y el gestor de paquetes `pip` puedan ser utilizados desde la línea de comandos (terminal). Si se omite este paso, será necesario añadir Python manualmente a las variables de entorno del sistema.

A.1.2. Node.js

La parte *frontend* de la aplicación requiere Node.js para gestionar las dependencias y ejecutar el servidor de desarrollo.

- Puedes descargar Node.js desde la página oficial: `https://nodejs.org/`

En el proceso de instalación de Node.js, asegúrate de que la opción **"Add to PATH"** esté habilitada. Esto permitirá que los comandos `node` y `npm` sean accesibles desde cualquier terminal.

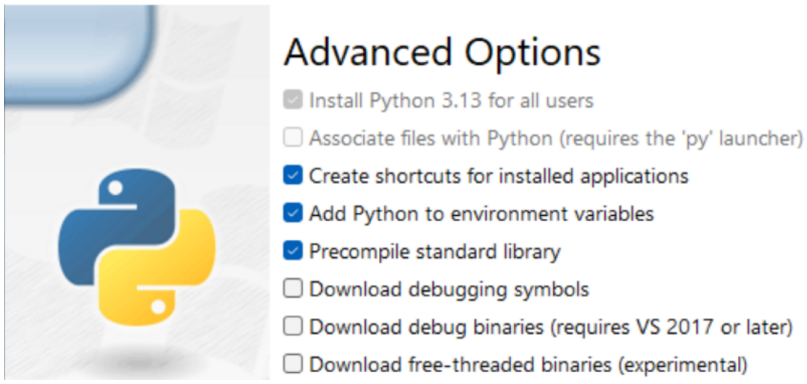

Figura A.1: Opciones avanzadas para la instalación de Python incluyendo la de "*Add Python to enviroment variables*".

A.2. Instalación de dependencias

Una vez instalados Python y Node.js, sigue los siguientes pasos para instalar las dependencias necesarias tanto en el *backend* como en el *frontend*.

A.2.1. Instalación de dependencias del *backend*

1. Abre una terminal o línea de comandos.
2. Navega hasta la carpeta principal del proyecto (donde se encuentra el archivo `requirements.txt`).
3. Ejecuta el siguiente comando para instalar todas las dependencias requeridas por el *backend*:

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

Esto descargará e instalará automáticamente todos los paquetes necesarios especificados en el archivo `requirements.txt`.

A.2.2. Instalación de dependencias del *frontend*

1. Abre otra terminal (o usa la misma, si lo prefieres).
2. Navega hasta la carpeta del *frontend* de tu proyecto. Si estás en la carpeta principal del proyecto, puedes hacerlo ejecutando:

   ```
   cd frontend
   ```

3. Una vez dentro de la carpeta del *frontend*, ejecuta el siguiente comando para instalar las dependencias de Node.js:

   ```
   npm install
   ```

A.3. Ejecución de la aplicación

A.3.1. Despliegue del *backend*

1. Desde la carpeta principal del proyecto, ejecuta el siguiente comando para iniciar el servidor *backend* (FastAPI con Uvicorn):

   ```
   python -m uvicorn app.main:app --reload
   ```

A.3.2. Despliegue del *frontend*

1. Abre una terminal en la carpeta `frontend` (si no lo hiciste antes, puedes ejecutar `cd frontend` desde la principal).

2. Ejecuta el siguiente comando para iniciar el servidor de desarrollo del *frontend*:

   ```
   npm run dev
   ```

Al completar este paso, la terminal mostrará un mensaje indicando el puerto donde se ha desplegado la aplicación *frontend* (por defecto, suele ser `http://localhost:5173/`).

A.3.3. Acceso a la aplicación

Por último, abre tu navegador web y accede a la siguiente dirección para comenzar a utilizar la aplicación:

`http://localhost:5173/`

A partir de este momento, la aplicación estará lista para su uso.

Apéndice B

Manual de Usuario

Se presenta a continuación el manual de usuario de la aplicación que explica detalladamente como se usa apoyándose en capturas reales. La aplicación es muy sencilla de usar.

1. En primer lugar, nada más desplegar la aplicación, nos encontramos con la página principal que, como vemos en la Figura B.1, nos solicita elegir el ejercicio que se quiere corregir. Para elegirlo debemos hacer *click* en el botón del ejercicio correspondiente.

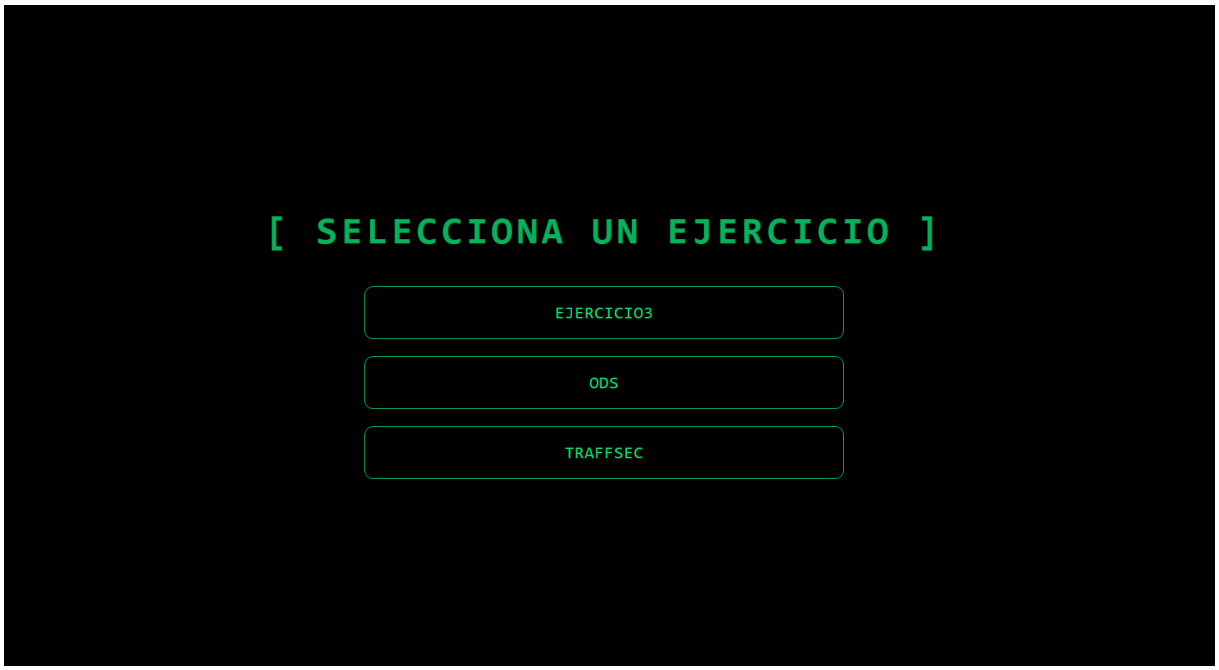

Figura B.1: Página de inicio.

2. Tras elegir el ejercicio, llegamos a una pantalla (Figura B.2) que nos solicita subir el diagrama ER, basta con pulsar en el rectángulo de "Seleccionar archivo" para que se nos abra una ventana con el explorador de archivos que nos permita seleccionar un archivo.

Figura B.2: Subida de diagrama ER.

3. Una vez seleccionada la imagen, se desbloquea el botón de "Procesar imagen" que pulsaremos para continuar (Figura B.3).

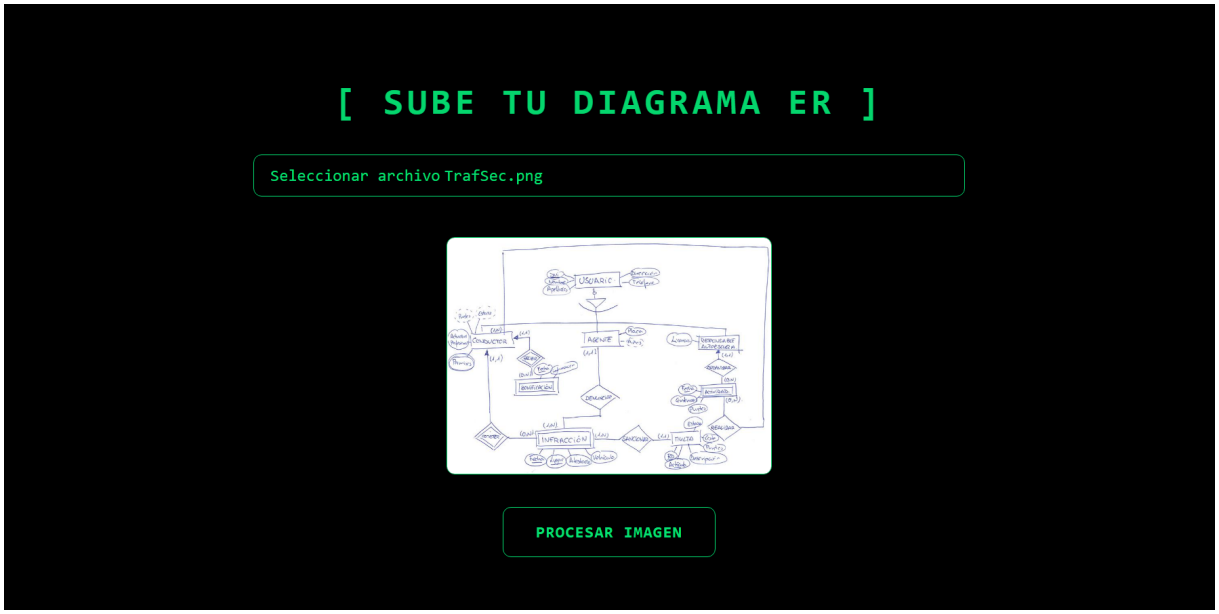

Figura B.3: Confirmación de imagen cargada.

4. Ahora nos saldrá una pantalla con todas las entidades (Figura B.4) que ha captado la aplicación, pulsando los botones podemos eliminar o añadir las entidades que queramos.

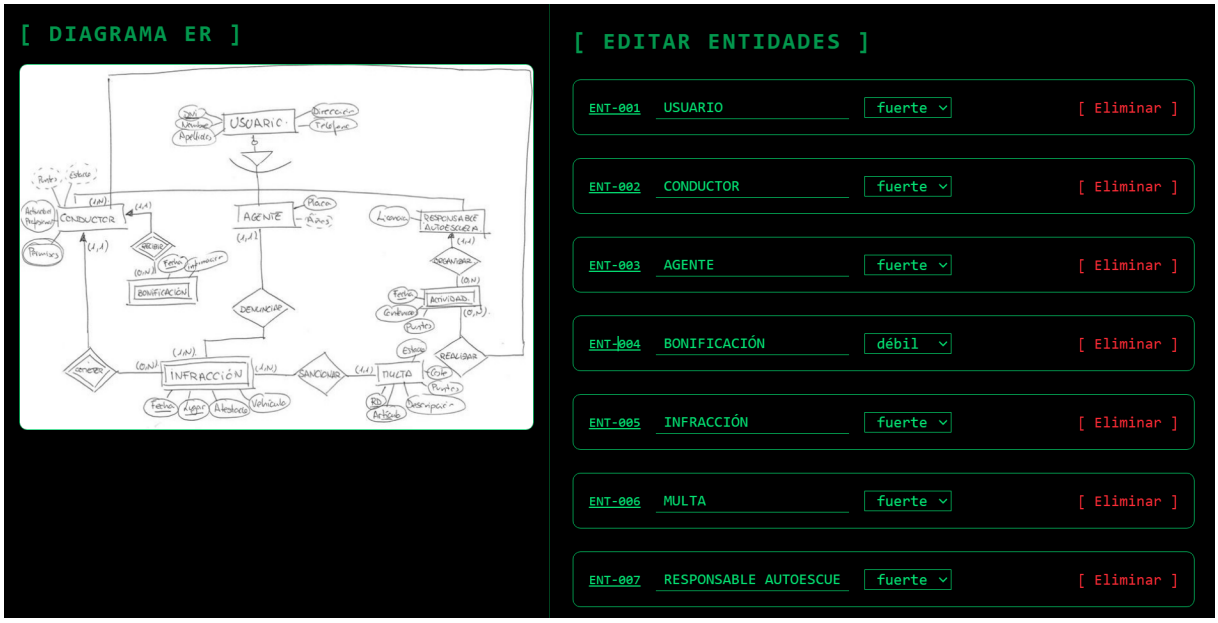

Figura B.4: Gestión de entidades.

5. Haciendo *click* en el id de una entidad se abre un desplegable (Figura B.5) que permite ver, añadir y modificar todos los detalles de los atributos de esa entidad.

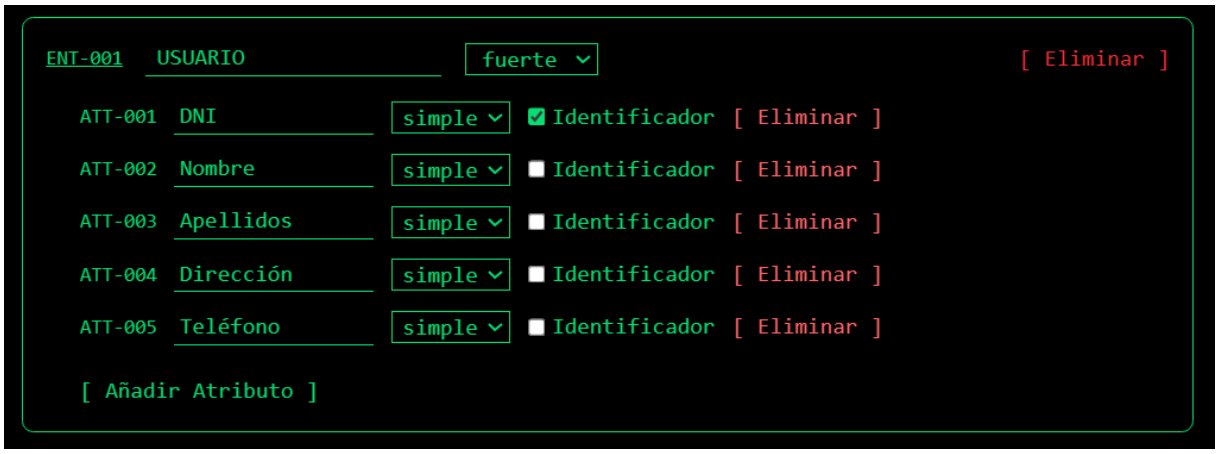

Figura B.5: Detalles de entidad.

6. Una vez se han hecho todos los cambios necesarios en las entidades se pulsa en el botón de "Confirmar entidades" (Figura B.6).

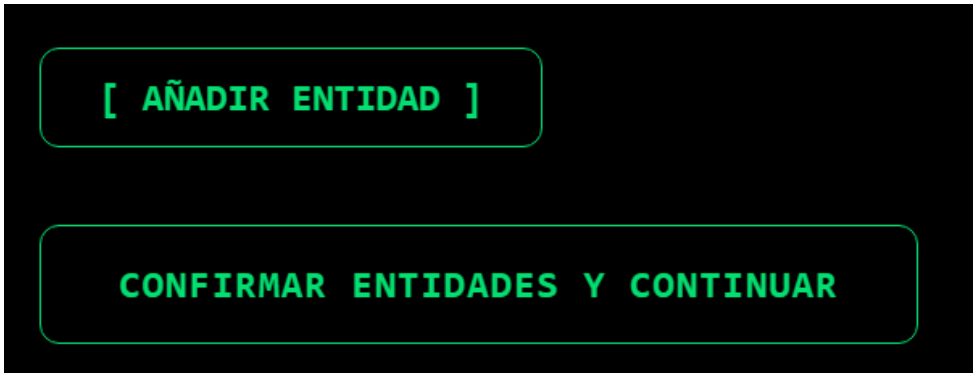

Figura B.6: Edición y confirmación de entidades.

7. Ahora nos saldrá una pantalla con todas las relaciones y relaciones IS-A (Figura B.7) que ha captado la aplicación, pulsando los botones podemos eliminar o añadir las relaciones o relaciones IS-A que queramos.

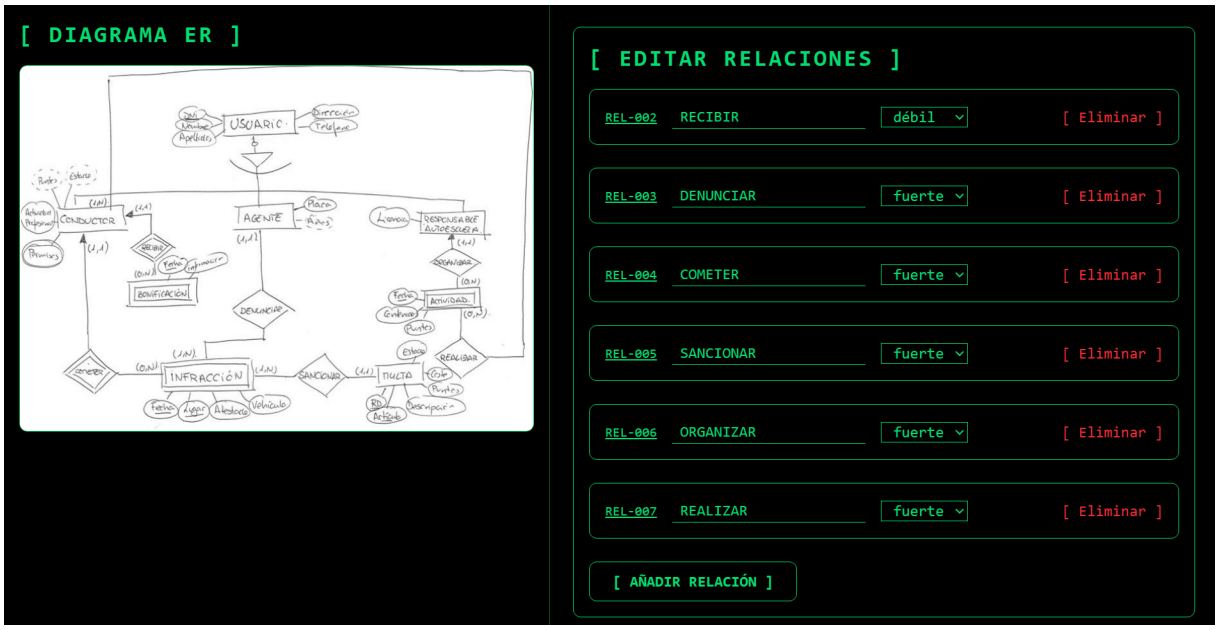

Figura B.7: Gestión de relaciones.

8. Haciendo *click* en el id de una relación se abre un desplegable (Figura B.8) que permite ver, añadir y modificar todos los detalles de los atributos y participantes de esa relación.

Figura B.8: Detalles de relación.

9. Haciendo *click* en el id de una relación IS-A se abre un desplegable (Figura B.9) que permite ver, añadir y modificar todos los detalles de los participantes de esa IS-A.

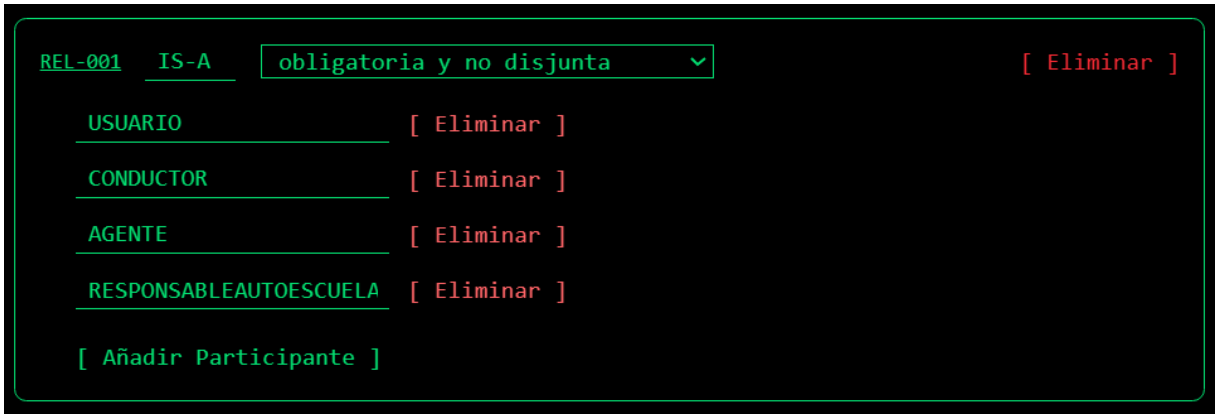

Figura B.9: Detalles de relación IS-A.

10. Una vez se han hecho todos los cambios necesarios en las relaciones se pulsa en el botón de "Finalizar edición y recibir feedback" (Figura B.10).

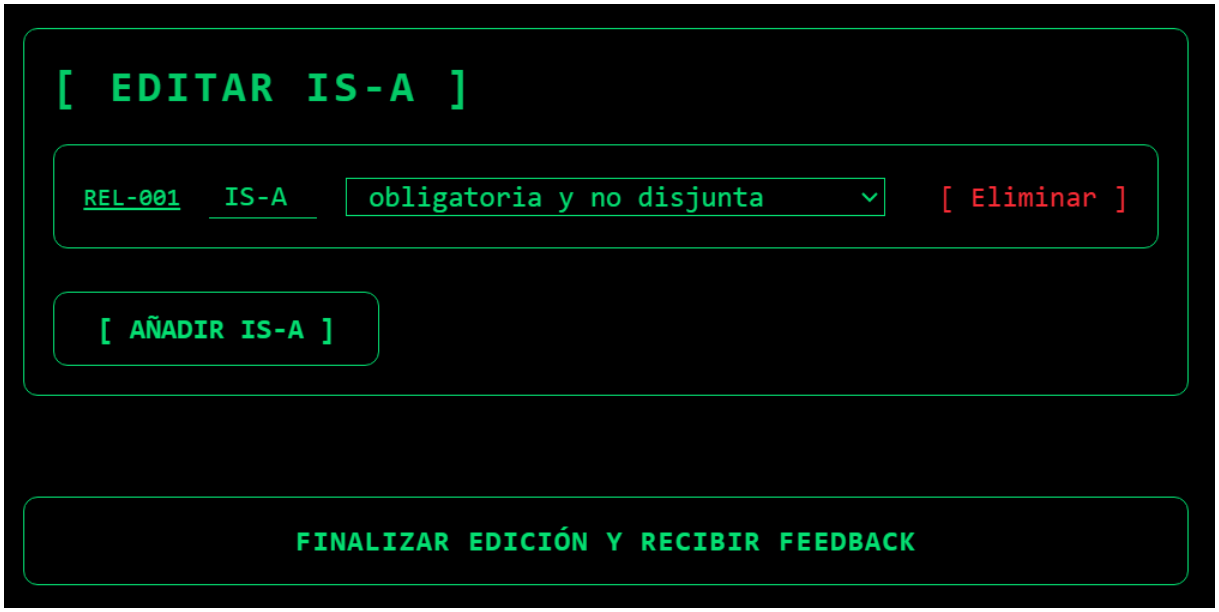

Figura B.10: Relaciones IS-A.

11. Finalmente, llegamos a la página de feedback que nos muestra tanto el *feedback* cualitativo (Figura B.11) como el *feedback* cuantitativo (Figura B.12). Pulsando el botón "Volver al inicio" volveríamos a la página principal que nos permitiría volver a corregir un ejercicio.

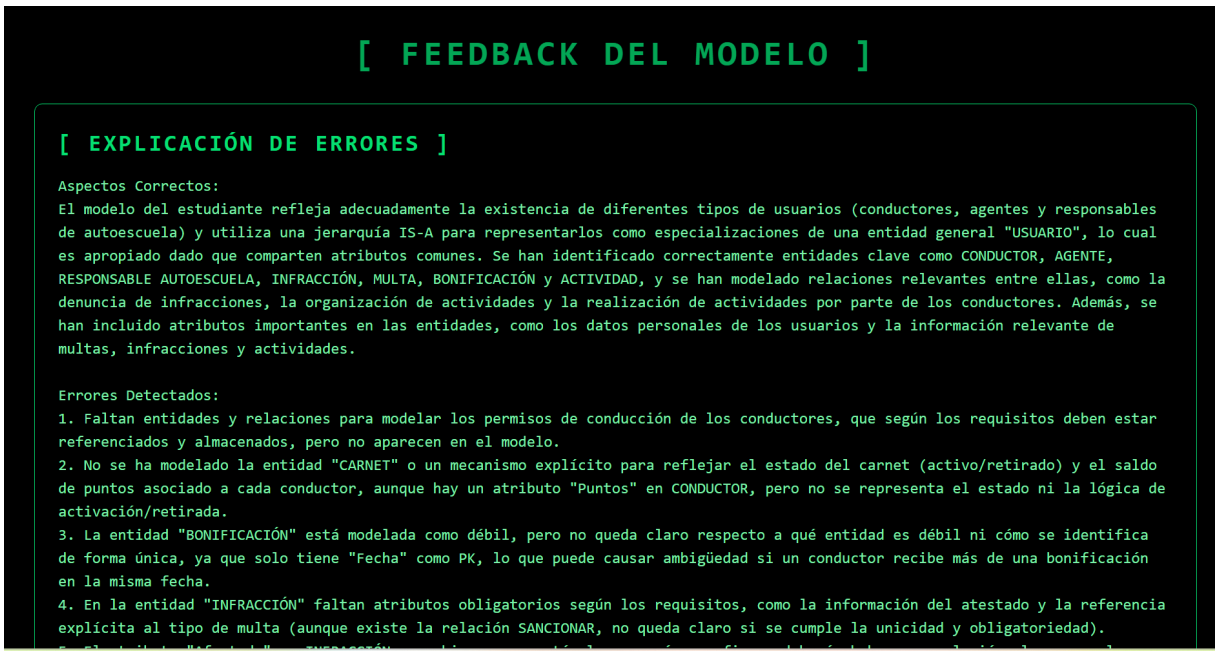

Figura B.11: *Feedback* cualitativo.

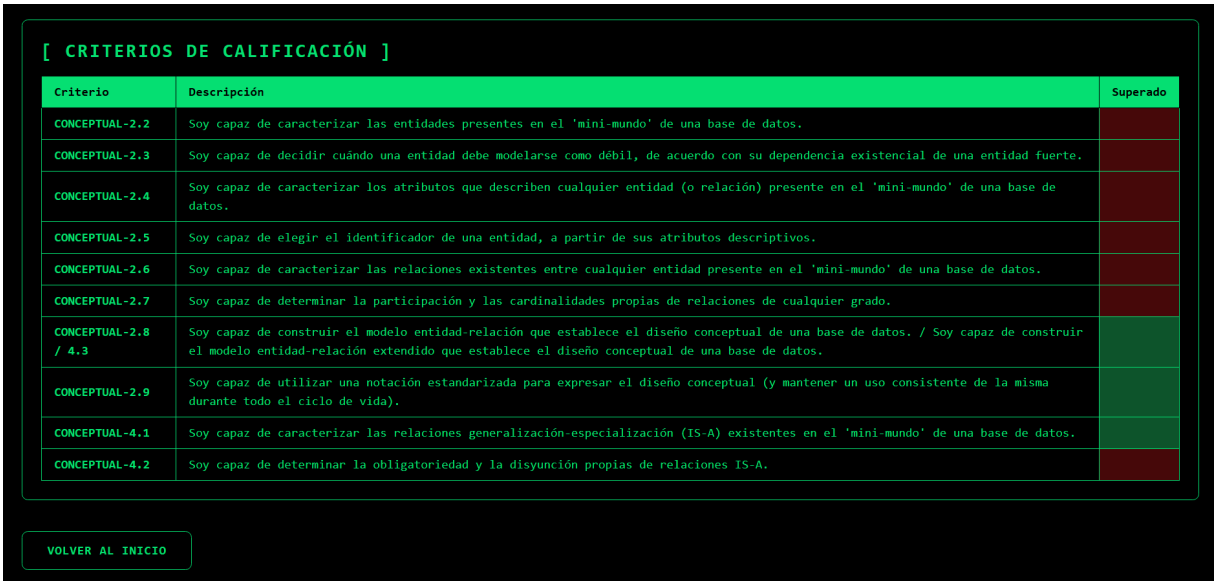

Figura B.12: *Feedback* cuantitativo.

Bibliografía

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat et al. «Gpt-4 technical report». En: *arXiv preprint arXiv:2303.08774* (2023).

[2] Meta AI. *LLaMA 2: Open Foundation and Fine-Tuned Chat Models*. Accedido por última vez el 9 de Abril de 2025. 2023. URL: https://ai.meta.com/llama.

[3] Mistral AI. *Mistral 7B and Mixtral Models*. Accedido por última vez el 9 de Abril de 2025. 2023. URL: https://mistral.ai/news/announcing-mistral-7b/.

[4] Anthropic. *Introducing Claude*. Accedido por última vez el 9 de Abril de 2025. 2023. URL: https://www.anthropic.com/index/claude.

[5] *Aprendizaje por refuerzo y Optimización*. Accedido última vez el 15 de Febrero del 2025. Instituto de ingeniería del conocimiento. URL: https://www.iic.uam.es/inteligencia-artificial/aprendizaje-por-refuerzo/.

[6] Leonardo Banh y Gero Strobel. «Generative artificial intelligence». En: *Electronic Markets* 33.1 (2023), pág. 63.

[7] Kent Beck y Cynthia Andres. *Extreme Programming Explained: Embrace Change*. 2nd. Addison-Wesley, 2004. ISBN: 978-0321278654.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell et al. «Language models are few-shot learners». En: *Advances in neural information processing systems* 33 (2020), págs. 1877-1901.

[9] Harrison Chase. *LangChain Documentation*. Consultado el 3 de mayo de 2025. 2025. URL: https://www.langchain.com/.

[10] Somchai Chatvichienchai. «Effective Development of Database Manipulation Skills Using Generative AI Tools». En: *2025 19th International Conference on Ubiquitous Information Management and Communication (IMCOM)*. 2025, págs. 1-6. DOI: 10.1109/IMCOM64595.2025.10857538.

[11] Cohere. *Introducing Command R+*. Accedido por última vez el 9 de Abril de 2025. 2024. URL: https://docs.cohere.com/docs/command-r-models.

[12] Thomas M Connolly y Carolyn E Begg. *Database systems: a practical approach to design, implementation, and management*. Pearson Education, 2005.

[13] Google DeepMind. *Gemini: Our Most Capable Model Yet*. Accedido por última vez el 9 de Abril de 2025. 2023. URL: https://deepmind.google/technologies/gemini/.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee y Kristina Toutanova. «Bert: Pre-training of deep bidirectional transformers for language understanding». En: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, págs. 4171-4186.

[15] Yenisleidy Fernández Romero y Yanette Díaz González. «Patrón Modelo-Vista-Controlador». En: *Telem@ tica (La Habana)* 11.1 (2012), págs. 47-57.

[16] Ronald A. Fisher. *Statistical Methods for Research Workers*. Edinburgh: Oliver y Boyd, 1925.

[17] Sarah Foss, Tatiana Urazova y Ramon Lawrence. «Automatic Generation and Marking of UML Database Design Diagrams». En: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1*. 2022, 626–632. ISBN: 9781450390705. DOI: 10.1145/3478431.3499376.

[18] Sandip Gautam. «Parsing Structural and Textual Information from UML Class diagrams to assist in verification of requirement specifications». En: (2022).

[19] Ligdi Gonzalez. *Diferencia entre aprendizaje supervisado y no supervisado*. Accedido última vez el 15 de Febrero del 2025. UNIR. Junio 15, 2018. URL: https://aprendeia.com/diferencia-entre-aprendizaje-supervisado-y-no-supervisado/.

[20] Cyril Goutte y Eric Gaussier. «A probabilistic interpretation of precision, recall and F-score, with implication for evaluation». En: *European conference on information retrieval*. Springer. 2005, págs. 345-359.

[21] IBM. *¿Qué es el aprendizaje no supervisado?* URL: https://www.ibm.com/es-es/topics/unsupervised-learning.

[22] IBM. *¿Qué es el aprendizaje semisupervisado?* Accedido última vez el 15 de Febrero del 2025. URL: https://www.ibm.com/es-es/topics/semi-supervised-learning.

[23] IBM. *¿Qué es el aprendizaje supervisado?* URL: https://www.ibm.com/es-es/topics/supervised-learning.

[24] IBM. *¿Qué es el deep learning?* Accedido última vez el 15 de Febrero del 2025. URL: https://www.ibm.com/es-es/topics/deep-learning.

[25] IBM. *¿Qué es el fine-tuning?* Accedido última vez el 15 de Febrero del 2025. URL: https://www.ibm.com/es-es/think/topics/fine-tuning.

[26] IBM. *¿Qué es la IA generativa?* Accedido última vez el 16 de Febrero del 2025. URL: http://ibm.com/es-es/think/topics/generative-ai.

[27] IBM. *¿Qué es un modelo de transformador?* Accedido última vez el 15 de Febrero del 2025. URL: https://www.ibm.com/es-es/think/topics/transformer-model.

[28] *Introduction to Semi-Supervised Learning*. Accedido última vez el 15 de Febrero del 2025. 26 Octubre, 2021. URL: https://teksands.ai/blog/semi-supervised-learning.

[29] Raisa Islam y Owana Marzia Moushi. «Gpt-4o: The cutting-edge advancement in multimodal llm». En: *Authorea Preprints* (2024).

[30] Carlos Jaimez-González y Jazmín Martínez-Samora. «DiagrammER: A web application to support the teaching-learning process of database courses through the creation of ER diagrams». En: *International Journal of Emerging Technologies in Learning (iJET)* 15.19 (2020), págs. 4-21.

[31] LangChain. *Citations | LangChain*. Consultado el 3 de mayo de 2025. 2025. URL: https://js.langchain.com/v0.1/docs/use_cases/question_answering/citations/.

[32] LangChain. *How to get a RAG application to add citations*. Consultado el 3 de mayo de 2025. 2025. URL: https://python.langchain.com/docs/how_to/qa_citations/.

[33] Juan Santiago Latorre Munar, Sebastián Ibáñez Capacho y Samuel Alejandro Jiménez Ramírez. *Inteligencia Artificial Generativa para Arquitectura de Software*. Universidad de los Andes, Colombia. 2024.

[34] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer y Veselin Stoyanov. «Roberta: A robustly optimized bert pretraining approach». En: *arXiv preprint arXiv:1907.11692* (2019).

[35] Humberto Cervantes Maceda. *Arquitecturas y Calidad de Software: ADD*. Universidad de los Andes, Colombia. Abril 2008.

[36] Christopher D. Manning, Prabhakar Raghavan e Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. ISBN: 9780521865715.

[37] Miguel A. Martinez-Prieto, Jorge Silvestre, Aníbal Bregón, Patricia Baz Domínguez, Clara Gándara González, Paula Mielgo Martín e Irene Peñas Pérez. «Una metodología basada en prácticas ágiles para la realización de Trabajos Fin de Grado». En: *Actas de las JENUI - Vol. 8 (2023)*. 2023, págs. 367-374.

[38] Clara Gándara Martínez. *Desarrollo de un sistema para la autoevaluación de modelos conceptuales de datos utilizando técnicas de visión computacional*. Accedido última vez el 31 de Marzo del 2025. Universidad de Valladolid. 2024. URL: https://uvadoc.uva.es/handle/10324/68631.

[39] OpenAI. *API PRICING*. Accedido última vez el 16 de Febrero del 2025. URL: https://openai.com/api/pricing/.

[40] OpenAI. *How do I calculate image tokens in GPT-4 vision*. Accedido última vez el 16 de Febrero del 2025. 2025. URL: https://community.openai.com/t/how-do-i-calculate-image-tokens-in-gpt4-vision/492318.

[41] OpenAI. *Images Guide*. Accedido última vez el 4 de Abril del 2025. 2025. URL: https://platform.openai.com/docs/guides/images.

[42] OpenAI. *OpenAI API*. Accedido última vez el 16 de Febrero del 2025. URL: https://openai.com/index/openai-api/.

[43] OpenAI. *Pricing*. Accedido última vez el 16 de Febrero del 2025. 2025. URL: https://openai.com/api/pricing/.

[44] OpenAI. *Prompt Engineering Guide*. Accedido última vez el 16 de Febrero del 2025. 2025. URL: https://platform.openai.com/docs/guides/prompt-engineering.

[45] OpenAI. *Text generation and prompting*. Accedido última vez el 16 de Febrero del 2025. URL: https://platform.openai.com/docs/guides/text?api-mode=chat.

[46] OpenAI. *What are tokens and how to count them*. Accedido última vez el 16 de Febrero del 2025. 2025. URL: https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them.

[47] *OpenAI developer platform*. Accedido última vez el 16 de Febrero del 2025. URL: https://platform.openai.com/docs/overview.

[48] *Qué es la Inteligencia Artificial*. Accedido última vez el 12 de Febrero del 2025. Gobierno de España. Abril 19, 2023. URL: https://planderecuperacion.gob.es/noticias/que-es-inteligencia-artificial-ia-prtr.

[49] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li y Peter J Liu. «Exploring the limits of transfer learning with a unified text-to-text transformer». En: *Journal of machine learning research* 21.140 (2020), págs. 1-67.

[50] *RAG (Retrieval-Augmented Generation): Revolucionando la Generación de Contenidos con Inteligencia Artificial*. Accedido última vez el 15 de Febrero del 2025. Arroba System. URL: https://arrobasystem.com/blogs/blog/retrieval-augmented-generation-revolucionando-la-generacion-de-contenidos-con-inteligencia-artificial.

[51] Sebastián Ramírez. *FastAPI Documentation*. Consultado el 3 de mayo de 2025. 2025. URL: https://fastapi.tiangolo.com/.

[52] Sebastián Ramírez. *Metadata and Docs URLs - FastAPI*. Consultado el 3 de mayo de 2025. 2025. URL: https://fastapi.tiangolo.com/tutorial/metadata/.

[53] John A. Rice. *Mathematical Statistics and Data Analysis*. 3rd. Belmont, CA: Cengage Learning, 2006.

[54] Johannes Schildgen. «MonstER Park - The Entity-Relationship-Diagram Learning Game». En: *ER Forum, Demo and Posters 2020 co-located with 39th International Conference on Conceptual Modeling (ER 2020)*. Vol. 2716. CEUR Workshop Proceedings. CEUR-WS.org, 2020, págs. 150-157. URL: https://ceur-ws.org/Vol-2716/paper13.pdf.

[55] PANAMERICAN Business School. *¿Qué es un prompt en IA y para qué sirve?* Accedido última vez el 16 de Febrero del 2025. URL: https://platform.openai.com/docs/guides/text?api-mode=chat.

[56] Ken Schwaber y Jeff Sutherland. *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game*. Última actualización en noviembre de 2020. 2020. URL: https://scrumguides.org/scrum-guide.html.

[57] Carsten Schwenke y AG Schering. «True positives, true negatives, false positives, false negatives». En: *Wiley StatsRef: Statistics Reference Online* (2014).

[58] Humasak Simanjuntak. «Proposed framework for automatic grading system of ER diagram». En: *2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE)*. 2015, págs. 141-146. DOI: 10.1109/ICITEED.2015.7408930.

[59] Ian Sommerville. *Software Engineering*. 9th. Addison-Wesley, 2011. ISBN: 9780137035151.

[60] C.J. Van Rijsbergen. «Information Retrieval: A Review». En: *Journal of Documentation* 35.4 (1979), págs. 249-287.

[61] *What is Transfer Learning?* Accedido última vez el 15 de Febrero del 2025. 13 Febrero, 2025. URL: https://teksands.ai/blog/semi-supervised-learning.

[62] Cort J. Willmott y Kenji Matsuura. «Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in Assessing Average Model Performance». En: *Climate Research* 30.1 (2005), págs. 79-82.

[63] Arne Wolfewicz. *Deep Learning vs. Machine Learning – What's The Difference?* Accedido última vez el 15 de Febrero del 2025. 30 Septiembre, 2024. URL: https://levity.ai/blog/difference-machine-learning-deep-learning.

[64] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov y Quoc V Le. «Xlnet: Generalized autoregressive pretraining for language understanding». En: *Advances in neural information processing systems* 32 (2019).

[65] *¿Qué es el transfer learning y qué ventajas tiene?* Accedido última vez el 15 de Febrero del 2025. UNIR. Noviembre 02, 2023. URL: https://www.unir.net/ingenieria/revista/transfer-learning/.

[66] *¿Qué es la API de OpenAI?* Accedido última vez el 16 de Febrero del 2025. URL: https://campus.datacamp.com/es/courses/working-with-the-openai-api/introduction-to-the-openai-api?ex=1.