



**TÍTULO DEL TFG: AUTOMATIZACIÓN DE LA CATALOGACIÓN DE CENTRALES
TELEFÓNICAS EN LA COMUNIDAD DE MADRID MEDIANTE INTELIGENCIA
ARTIFICIAL**

MODALIDAD DEL TFG: CONVENCIONAL

CONVOCATORIA: ORDINARIA

ALUMNO / A: ADRIAN TORAL ONTORIA

TUTOR / A: JAVIER ALGARRA

CO-TUTOR / A: ALFONSO CASTRO

GRADO: INGENIERÍA DEL SOFTWARE

Contenido

RESUMEN	5
ABSTRACT	6
1. INTRODUCCIÓN	7
1.1 Motivación y contexto	7
1.2 Planteamiento del problema	8
1.3 Antecedentes históricos	9
1.4 Relevancia del estudio	10
1.4.1 Metodología general.....	11
1.5 Objetivos del trabajo.....	12
2. Estado de la cuestión	14
2.1 Marco Teórico del trabajo	16
2.1.1 Conceptos de patrimonio arquitectónico.....	16
2.1.2 Bases de datos y su uso en la gestión del patrimonio	17
2.1.3 Procesamiento de lenguaje natural.....	18
2.2 Trabajos relacionados.....	19
2.2.1 La inteligencia artificial y su importancia en el patrimonio y las industrias culturales	19
2.2.2 Aplicación de algoritmos de inteligencia artificial en la conservación del arte y patrimonio cultural	20
2.2.3 Uso de la Inteligencia Artificial en la preservación del patrimonio cultural	20
2.2.4 PatrimoniIndustrial.cat: catálogo digital del patrimonio industrial catalán	20
3. ASPECTOS METODOLÓGICOS	21
3.1 Metodología	21
3.1.1 Product Backlog.....	22
3.1.2 Tareas e historias de usuario del Product Backlog	23
3.2 Tecnologías empleadas.....	26
3.2.1 Git	27
3.2.2 PyCharm.....	27
3.2.3 Python como lenguaje de programación.....	28
3.2.4 Python como intérprete.....	28
3.2.5 Qué es un ORM y su utilidad en la gestión de bases de datos	28
3.2.6 Limitaciones de los ORMs convencionales con Microsoft Access .	29
3.2.7 Microsoft Access y archivos .mdb	30
3.2.8 Py ODBC	30

3.2.9	Obsidian	31
3.3	Versiones de las tecnologías.....	31
4.	DESARROLLO DEL TRABAJO	31
4.1	Ingeniería inversa.....	32
4.2	Creación del ORM.....	35
4.3	Obtención de los datos.....	44
4.4	Modelo de inteligencia artificial.....	45
5.	CONCLUSIONES.....	50
5.1	Objetivos cumplidos	52
5.2	Futuros desarrollos.....	53
6.	REFERENCIAS.....	56
6.1	Bibliografía.....	56
6.2	Webgrafía.....	58
7.	ANEXOS	60

RESUMEN

Este Trabajo de Fin de Grado aborda la catalogación automatizada de centrales telefónicas en la Comunidad de Madrid mediante la implementación de agentes de inteligencia artificial. El objetivo principal es optimizar y automatizar el proceso de recopilación y organización de información histórica y arquitectónica relevante, integrándola eficientemente en el sistema de gestión de patrimonio industrial ETNOCAM, utilizado por la Dirección General de Patrimonio Cultural. Para ello, se lleva a cabo una fase inicial de ingeniería inversa sobre la base de datos actual del sistema, asegurando una correcta modelación y estructuración que facilite la interoperabilidad con el nuevo sistema automatizado.

El proyecto se estructura en tres fases esenciales: primero, el desarrollo de un sistema basado en inteligencia artificial que extrae automáticamente información clave de fuentes documentales especializadas; segundo, la creación de un software específico para la carga automatizada y estructurada de fichas patrimoniales; y tercero, una fase final de validación manual que asegura la calidad y precisión de los datos mediante la verificación de planos, fotografías y documentos históricos. Como resultado, este trabajo busca facilitar significativamente la gestión, conservación y documentación del patrimonio arquitectónico industrial madrileño, proporcionando una herramienta eficaz y sostenible frente a la progresiva desaparición de estas infraestructuras debido a la modernización de las telecomunicaciones.

ABSTRACT

This Final Degree Project addresses the automated cataloging of telephone exchanges in the Community of Madrid through the implementation of artificial intelligence agents. Its primary objective is to optimize and automate the collection and organization of relevant historical and architectural information, integrating it efficiently into the industrial heritage management system ETNOCAM, managed by the General Directorate of Cultural Heritage. To achieve this, an initial phase of reverse engineering on the existing system's database is conducted, ensuring proper modeling and structuring to guarantee interoperability with the new automated system.

The project is structured into three fundamental phases: firstly, the development of an artificial intelligence-based system that automatically extracts critical information from specialized documentary sources; secondly, the creation of specific software for automated and structured loading of heritage records; and finally, a manual validation phase to ensure the quality and accuracy of the data through verification of plans, photographs, and historical documents. Consequently, this work aims to significantly facilitate the management, preservation, and documentation of Madrid's industrial architectural heritage, providing an effective and sustainable tool to counteract the gradual disappearance of these infrastructures due to telecommunications modernization.

1. INTRODUCCIÓN

1.1 Motivación y contexto

La arquitectura telefónica constituye un elemento fundamental del patrimonio industrial y urbano en la Comunidad de Madrid. Durante el siglo XX, las centrales telefónicas fueron concebidas no solo como infraestructuras esenciales para las telecomunicaciones, sino también como edificios con un significativo valor arquitectónico, integrándose armónicamente en el paisaje urbano.

Este enfoque, que otorga a ciertas infraestructuras un papel simbólico y representativo dentro de la ciudad, ha sido abordado en estudios sobre arquitectura institucional, destacando cómo su diseño responde no solo a criterios funcionales, sino también a intenciones estéticas y sociales. Sin embargo, el rápido avance tecnológico y la transición de la telefonía tradicional basada en cobre hacia redes modernas de fibra óptica han dejado muchas de estas infraestructuras en desuso, aumentando considerablemente el riesgo de deterioro y desaparición (García Algarra, 2012).

A pesar de su notable importancia histórica, arquitectónica y cultural, el proceso de catalogación y conservación de estas infraestructuras se ha realizado de manera predominantemente manual, lo que implica un procedimiento lento, laborioso y propenso a errores. Actualmente, la Dirección General de Patrimonio Cultural de la Comunidad de Madrid administra ETNOCAM, una base de datos destinada a registrar y estudiar estos edificios para facilitar su conservación.

Sin embargo, ETNOCAM es una aplicación antigua y cerrada, que carece de una interfaz de programación de aplicaciones (API), lo cual dificulta significativamente su actualización y modernización. Además, la incorporación y actualización de registros en esta base requieren un esfuerzo considerable, consumiendo grandes cantidades de tiempo y recursos humanos.

Este proyecto nace con la motivación específica de resolver estos problemas mediante la implementación de tecnologías avanzadas de inteligencia artificial, buscando optimizar y automatizar el proceso de catalogación de las centrales telefónicas. Además, este trabajo responde a un encargo específico por parte de la Comunidad de Madrid (CAM), lo que le otorga un valor añadido al tratarse de una iniciativa con aplicación real y directa sobre la gestión patrimonial regional.

El objetivo central es desarrollar un sistema basado en IA que pueda extraer información relevante automáticamente desde diversas fuentes documentales especializadas, estructurándola adecuadamente para su incorporación en los sistemas patrimoniales existentes. Esto permite dar valor añadido a la base de datos existente, desligándola de la interfaz de usuario obsoleta y facilitando futuras actualizaciones y mejoras tecnológicas.

Además de aumentar la eficiencia administrativa y técnica en la gestión del patrimonio arquitectónico, este proyecto tiene la capacidad real de preservar eficazmente un legado histórico que actualmente enfrenta amenazas significativas. La aplicación de estas tecnologías innovadoras pretende establecer una plataforma robusta y escalable, que pueda servir como referencia en futuros proyectos tanto dentro como fuera de la Comunidad de Madrid, contribuyendo de forma tangible al ámbito de la conservación patrimonial.

1.2 Planteamiento del problema

La Comunidad de Madrid posee un vasto y valioso patrimonio arquitectónico relacionado con las telecomunicaciones, compuesto principalmente por centrales telefónicas que han sido fundamentales en el desarrollo histórico y tecnológico de la región. Sin embargo, los rápidos avances tecnológicos en telecomunicaciones han provocado que muchas de estas infraestructuras queden obsoletas, lo que ha llevado a su abandono, transformación o, en el peor de los casos, demolición.

La conservación efectiva y la catalogación precisa de estos edificios enfrentan actualmente importantes desafíos. La información relevante se encuentra fragmentada y dispersa en diversas fuentes documentales, careciendo de un sistema centralizado o automatizado para recopilarla eficientemente. Además, la gestión actual depende en gran medida del trabajo manual de introducción de datos en sistemas existentes como INPHIS, lo que implica una considerable inversión de tiempo y esfuerzo, incrementando además el riesgo de errores y pérdida de información crítica.

Ante este contexto, surge la necesidad real y urgente de implementar soluciones tecnológicas que permitan automatizar y optimizar los procesos de extracción, organización y carga de información en bases de datos patrimoniales. Este trabajo propone precisamente la creación de un sistema basado en inteligencia artificial capaz de agilizar la recopilación y validación de datos sobre estas infraestructuras, integrándolos de manera efectiva en los sistemas existentes.

Este proyecto tiene una utilidad real y tangible para la Comunidad de Madrid, ya que no solo facilita significativamente la tarea administrativa relacionada con la catalogación, sino que además garantiza la preservación de información histórica y arquitectónica que podría perderse sin una adecuada gestión. La implementación efectiva de esta solución tecnológica contribuirá directamente a proteger el legado arquitectónico industrial madrileño, proporcionando además una plataforma robusta para futuras iniciativas en el ámbito de la conservación y el patrimonio cultural.

1.3 Antecedentes históricos

Las centrales telefónicas han desempeñado un papel crucial en la evolución de las telecomunicaciones, actuando no solo como infraestructuras técnicas para la conmutación de llamadas, sino también como elementos emblemáticos y reconocibles del paisaje urbano. En España, el desarrollo de la arquitectura telefónica estuvo estrechamente vinculado con la expansión de la Compañía Telefónica Nacional de España (CTNE), influenciado por tendencias arquitectónicas internacionales y por la necesidad de integración urbana.

En la Comunidad de Madrid, numerosas centrales telefónicas fueron diseñadas por arquitectos destacados, lo que ha conferido a estos edificios un valor añadido desde el punto de vista del patrimonio industrial. Sin embargo, la progresiva sustitución tecnológica desde la telefonía tradicional basada en cobre hacia la fibra óptica ha reducido drásticamente la necesidad operativa de estas instalaciones, poniendo en riesgo su conservación. Algunos edificios, como la central de la calle Hermosilla, han sido demolidos en 2022 (de la Cruz, 2022), mientras que otros permanecen en desuso o corren peligro de transformación sin contar con registros adecuados que aseguren su protección.

1.4 Relevancia del estudio

La creciente desaparición de estas infraestructuras representa un desafío crucial para la conservación del patrimonio industrial y arquitectónico madrileño. Si bien algunas centrales han sido declaradas Bien de Interés Cultural (BIC), como la Central de Satélites de Buitrago del Lozoya, la mayoría carece de una protección adecuada, y su catalogación es insuficiente o inexistente. Para revertir esta situación y asegurar su preservación, es imprescindible documentar adecuadamente estos edificios mediante bases de datos especializadas.

La Dirección General de Patrimonio Cultural gestiona actualmente ETNOCAM, una plataforma destinada a registrar bienes de interés patrimonial. Sin embargo, la incorporación y actualización de los datos en esta plataforma aún dependen de procedimientos manuales, lo que limita considerablemente su eficacia operativa. La implementación de inteligencia artificial permitirá automatizar eficazmente la extracción, organización e integración de información, mejorando el acceso y reduciendo drásticamente la carga administrativa.

La relevancia de este estudio radica en la propuesta de una solución tecnológica innovadora capaz de transformar profundamente la gestión del patrimonio arquitectónico, mejorando sustancialmente la eficiencia administrativa, facilitando el acceso a información vital para investigadores y gestores culturales, y proporcionando una plataforma de referencia para futuras iniciativas similares tanto dentro como fuera de la Comunidad de Madrid.

1.4.1 Metodología general

Para abordar el desafío de la catalogación de las centrales telefónicas en la Comunidad de Madrid, se propone una metodología estructurada en tres fases principales:

1.4.1.1 Extracción automatizada de información

Se desarrollará un sistema basado en inteligencia artificial capaz de analizar fuentes documentales especializadas y extraer información clave sobre las centrales telefónicas. Este proceso incluirá la identificación de su ubicación, características arquitectónicas y autores, permitiendo una recopilación de datos más eficiente y precisa.

1.4.1.2 Integración con bases de datos patrimoniales

Se llevará a cabo un proceso de ingeniería inversa sobre la base de datos existente con el objetivo de modelar su estructura y asegurar una interoperabilidad óptima con el sistema ETNOCAM. Esto garantizará que los nuevos registros se integren correctamente y puedan ser gestionados de manera efectiva dentro de la plataforma patrimonial.

1.4.1.3 Validación y carga de datos

Se implementará un sistema de carga automatizada de fichas patrimoniales, optimizando el ingreso de información en la base de datos. Además, se incluirá una fase de validación manual en la que se incorporarán planos, fotografías y documentos históricos, asegurando la calidad y fiabilidad de los datos almacenados.

1.5 Objetivos del trabajo

El propósito central de este trabajo es desarrollar un sistema basado en inteligencia artificial que permita automatizar eficazmente la catalogación de centrales telefónicas en la Comunidad de Madrid, optimizando significativamente los procesos de conservación y documentación del patrimonio arquitectónico industrial.

Para cumplir con este objetivo general, se plantean los siguientes objetivos específicos:

- Realizar una ingeniería inversa exhaustiva sobre la base de datos actual gestionada por la Comunidad de Madrid, identificando y documentando de manera clara y precisa todas sus tablas, columnas, tipos de datos y relaciones existentes. Este análisis permitirá diseñar un esquema de datos eficiente, facilitando la compatibilidad e integración con el sistema de gestión patrimonial existente (ETNOCAM).
- Desarrollar un ORM (Object-Relational Mapping) personalizado que permita realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre la base de datos Microsoft Access, simplificando considerablemente la interacción con la base de datos y mejorando tanto la eficiencia como la seguridad del proceso.
- Implementar un sistema que permita la importación y exportación de datos en formatos estándar (.csv y .json) a través del ORM desarrollado, facilitando así la integración y el análisis de la información patrimonial con herramientas externas.

- Integrar un modelo de inteligencia artificial basado en técnicas avanzadas de procesamiento de lenguaje natural y aprendizaje automático (fine tuning) utilizando la plataforma OpenAI, capaz de extraer automáticamente información clave sobre las centrales telefónicas a partir de fuentes documentales especializadas, estructurando y organizando esta información de manera eficiente y precisa para su incorporación automática en la base de datos.
- Crear un mecanismo de validación y supervisión que asegure la precisión, coherencia y calidad de la información recopilada y catalogada, reduciendo al máximo la intervención manual y minimizando errores o inconsistencias en los registros incorporados al sistema.
- Establecer un entorno de documentación técnica exhaustiva y accesible, utilizando herramientas especializadas como Obsidian, garantizando la transparencia y trazabilidad del desarrollo tecnológico y facilitando futuras ampliaciones y mejoras.
- Evaluar rigurosamente la eficacia del sistema desarrollado mediante pruebas específicas que midan tanto la precisión en la catalogación como el tiempo de optimización respecto al proceso manual tradicional. A partir de estas evaluaciones, elaborar un informe detallado con recomendaciones concretas orientadas a mejorar aún más la conservación patrimonial, incluyendo propuestas de optimización y posibles aplicaciones futuras de esta tecnología en otros ámbitos de gestión cultural.

Mediante estos objetivos específicos, el trabajo busca proporcionar una solución tecnológica innovadora que no solo facilite y optimice el proceso de gestión patrimonial, sino que también sirva como referente metodológico y tecnológico para futuras iniciativas similares en la Comunidad de Madrid y en otras regiones interesadas en la preservación y documentación del patrimonio arquitectónico

2. Estado de la cuestión

La integración de la inteligencia artificial (IA) en la catalogación y gestión del patrimonio arquitectónico representa un avance significativo que está transformando el ámbito de la conservación cultural. Numerosos estudios recientes han abordado cómo las técnicas avanzadas de IA, especialmente aquellas relacionadas con el procesamiento de lenguaje natural (PLN) y aprendizaje automático, pueden contribuir eficazmente a la documentación y preservación del patrimonio.

Según Fernández, García y Ramírez (2023), la inteligencia artificial ofrece herramientas innovadoras para automatizar procesos complejos y repetitivos en la digitalización del patrimonio cultural. Este estudio destaca particularmente el potencial del aprendizaje automático para analizar documentos históricos y extraer información relevante con alta precisión, permitiendo una gestión más efectiva y accesible de archivos históricos.

Mendoza y Pérez (2021) profundizan en la aplicación específica del PLN en la clasificación y catalogación de documentos históricos, identificando patrones lingüísticos que facilitan la interpretación automática de textos antiguos. Este enfoque ha resultado especialmente útil para manejar grandes volúmenes de información documental, optimizando considerablemente el proceso tradicional manual de catalogación.

En el contexto específico del patrimonio arquitectónico industrial, Martínez, Pérez y Sánchez (2021) resaltan el desafío que supone la documentación y conservación de edificios con valor patrimonial que corren riesgo debido a la transformación urbana. La automatización propuesta mediante inteligencia artificial podría responder eficazmente a esta problemática al acelerar significativamente los procesos de identificación y catalogación de bienes arquitectónicos.

Adicionalmente, Rodríguez y Sánchez (2022) abordan los retos técnicos relacionados con la interoperabilidad de bases de datos patrimoniales, destacando que una gestión eficiente de los datos requiere la estandarización y normalización de formatos para asegurar la compatibilidad entre sistemas heterogéneos. La aplicación de técnicas automatizadas de IA puede facilitar este proceso, mejorando así la accesibilidad y coherencia de la información patrimonial.

Finalmente, proyectos concretos como el catálogo digital georreferenciado sobre patrimonio industrial catalán "PatrimoniIndustrial.cat" (Álvarez, s. f.) demuestran la eficacia de combinar métodos tradicionales de catalogación con tecnologías digitales avanzadas. Aunque este proyecto no integra directamente técnicas avanzadas de IA, su metodología y estructura ofrecen una base sólida sobre la que implementar estas tecnologías emergentes. Además, diversos estudios destacan cómo la digitalización y el uso de herramientas geoespaciales mejoran significativamente la accesibilidad, análisis y preservación del patrimonio cultural (López-Menchero Bendicho y Herráez, 2019).

En conclusión, los estudios revisados ponen de manifiesto que la aplicación de la inteligencia artificial en la gestión patrimonial no solo agiliza los procesos administrativos y documentales, sino que también mejora significativamente la precisión y accesibilidad de la información, representando así un paso esencial para la conservación sostenible del patrimonio arquitectónico e industrial.

2.1 Marco Teórico del trabajo

El marco teórico del presente trabajo tiene por objetivo definir y aclarar los conceptos fundamentales que sustentan el proyecto, proporcionando un contexto sólido que permita comprender tanto la problemática identificada como la solución tecnológica propuesta.

En este apartado se abordarán de manera sistemática conceptos esenciales relacionados con el patrimonio arquitectónico industrial, las técnicas avanzadas de inteligencia artificial aplicadas a la documentación patrimonial y el uso de sistemas modernos de gestión de bases de datos, particularmente mediante ORMs y archivos .mdb de Microsoft Access.

2.1.1 Conceptos de patrimonio arquitectónico

El patrimonio arquitectónico comprende el conjunto de edificaciones caracterizadas por su valor histórico, artístico o cultural, que reflejan y testimonian la evolución social y tecnológica de una época específica. Dentro del patrimonio arquitectónico, se destaca el patrimonio industrial, el cual incluye construcciones directamente vinculadas con actividades productivas, transporte y telecomunicaciones, elementos esenciales para el desarrollo económico e industrial de una región (García y López, 2020).

Las centrales telefónicas constituyen un ejemplo relevante de patrimonio industrial, ya que desempeñaron un rol fundamental en el desarrollo de las comunicaciones durante el siglo XX. Más allá de su función operativa, estos edificios se diseñaron considerando cuidadosamente criterios arquitectónicos específicos para asegurar su integración visual y funcional en el paisaje urbano.

No obstante, la progresiva transición tecnológica desde la telefonía tradicional (basada en cobre) hacia redes más modernas de fibra óptica ha provocado que muchas de estas infraestructuras queden obsoletas, aumentando significativamente el riesgo de deterioro o demolición por falta de una adecuada catalogación y protección (Martínez et al., 2021).

En España, la identificación, catalogación y protección legal del patrimonio arquitectónico se encuentra regulada principalmente por la Ley de Patrimonio Histórico Español de 1985, que establece directrices claras para la declaración de bienes de interés cultural (BIC).

En la Comunidad de Madrid, la Dirección General de Patrimonio Cultural es responsable del registro, catalogación y gestión del patrimonio arquitectónico mediante sistemas específicos como ETNOCAM (Dirección General de Patrimonio Cultural, 2022), que actúa como repositorio centralizado para la documentación patrimonial en la región.

2.1.2 Bases de datos y su uso en la gestión del patrimonio

Las bases de datos patrimoniales son fundamentales para el almacenamiento, gestión y consulta eficiente de información histórica y arquitectónica. En España, plataformas como ETNOCAM e INPHIS constituyen ejemplos relevantes en el ámbito patrimonial, facilitando la centralización y organización sistemática de información crítica para la conservación y gestión documental del patrimonio arquitectónico.

Uno de los principales retos enfrentados por estas plataformas es la interoperabilidad (Rodríguez y Sánchez, 2022), definida como la capacidad de integrar información de diversas fuentes sin comprometer la coherencia, calidad y accesibilidad de los datos almacenados.

Actualmente, la mayoría de los datos patrimoniales se generan e introducen de manera manual, creando estructuras de información heterogéneas que dificultan considerablemente la integración en sistemas automatizados avanzados.

Para enfrentar este desafío, se utilizan técnicas específicas de normalización de datos. Estas técnicas permiten transformar y estandarizar los datos provenientes de diversas fuentes, garantizando así que la información extraída mediante herramientas de inteligencia artificial pueda incorporarse de forma precisa en las bases de datos patrimoniales como ETNOCAM e INPHIS. Esto asegura una gestión eficaz y optimizada del patrimonio arquitectónico.

2.1.3 Procesamiento de lenguaje natural

La inteligencia artificial (IA) se ha consolidado como una herramienta clave para la gestión y conservación del patrimonio arquitectónico (Fernández et al., 2023), gracias a su capacidad de automatizar procesos complejos como la extracción de datos, clasificación documental y análisis histórico.

En particular, el procesamiento de lenguaje natural (PLN) representa una de las técnicas más prometedoras en este campo, ya que permite a los sistemas computacionales interpretar, estructurar y extraer información relevante de documentos textuales con gran precisión (Mendoza y Pérez, 2021). El PLN se ha aplicado con éxito en diversos contextos culturales, como la digitalización y catalogación automatizada de documentos históricos en bibliotecas, museos y archivos (Gómez y Ramírez, 2022).

En este proyecto específico, la combinación de técnicas avanzadas de inteligencia artificial y PLN tiene como propósito principal automatizar la extracción y estructuración de información relevante a partir de documentos históricos relacionados con las centrales telefónicas. El objetivo es generar registros estructurados compatibles con bases de datos patrimoniales, facilitando así su gestión, acceso y preservación futura.

No obstante, es necesario destacar que el uso de IA y PLN presenta desafíos significativos, como la validación manual imprescindible para garantizar la exactitud y coherencia de los datos extraídos. Además, la interpretación de documentos históricos que emplean lenguaje técnico especializado o arcaico puede requerir ajustes específicos en los modelos de PLN, con el fin de mejorar su precisión y rendimiento en contextos altamente especializados como el patrimonial.

2.2 Trabajos relacionados

En esta sección se presentan diversas investigaciones recientes que analizan la utilización de inteligencia artificial para apoyar la conservación, catalogación y difusión del patrimonio arquitectónico e industrial. El objetivo es identificar tecnologías innovadoras y analizar críticamente sus fortalezas y limitaciones para establecer el contexto en el que se desarrolla este proyecto.

2.2.1 La inteligencia artificial y su importancia en el patrimonio y las industrias culturales

Diversas investigaciones han mostrado el potencial de la inteligencia artificial para procesar grandes cantidades de información patrimonial, especialmente en la clasificación automática y análisis detallado de fotografías históricas y documentos antiguos (Acento, s. f.). La principal ventaja reside en la optimización y aceleración de tareas que habitualmente demandan gran esfuerzo manual. Además, estos sistemas ofrecen la posibilidad de personalizar experiencias culturales mediante recomendaciones basadas en los intereses específicos de los usuarios, aumentando la accesibilidad y el impacto del patrimonio cultural.

2.2.2 Aplicación de algoritmos de inteligencia artificial en la conservación del arte y patrimonio cultural

En estudios recientes se ha abordado cómo combinar algoritmos de inteligencia artificial con tecnologías avanzadas puede mejorar significativamente la conservación del patrimonio cultural (Revista ASRI, s. f.). Estas investigaciones destacan la eficacia de la IA en tareas prácticas como la restauración predictiva, la monitorización automatizada del deterioro y la administración eficiente de grandes colecciones patrimoniales. Estos resultados son potencialmente aplicables a la gestión y conservación del patrimonio arquitectónico industrial, contribuyendo a una catalogación más precisa y eficiente.

2.2.3 Uso de la Inteligencia Artificial en la preservación del patrimonio cultural

Una iniciativa destacada emplea modelos avanzados de inteligencia artificial para automatizar la extracción de información relevante a partir de imágenes digitales de objetos patrimoniales (Jornal da Unicamp, 2024). Este enfoque mejora sustancialmente la precisión y rapidez del proceso de catalogación digital, facilitando no solo la conservación sino también el acceso público a estos recursos patrimoniales. La capacidad del sistema para extraer automáticamente características clave ofrece importantes beneficios prácticos para la gestión del patrimonio arquitectónico industrial.

2.2.4 PatrimoniIndustrial.cat: catálogo digital del patrimonio industrial catalán

El proyecto PatrimoniIndustrial.cat se centra en la creación de un catálogo digital interactivo y georreferenciado sobre el patrimonio industrial en Cataluña (Álvarez, s. f.). Aunque este proyecto no implementa directamente técnicas de inteligencia artificial, su enfoque meticuloso y estructurado en la catalogación digital proporciona un punto de partida sólido para integrar futuras aplicaciones avanzadas de IA. La combinación de estos métodos tradicionales con nuevas tecnologías digitales puede mejorar considerablemente la gestión, visibilidad y accesibilidad del patrimonio arquitectónico industrial.

3. ASPECTOS METODOLÓGICOS

Para el desarrollo del presente trabajo se ha utilizado la metodología ágil Scrum (Schwaber y Sutherland, 2020) caracterizada por una gestión iterativa e incremental que permite una adaptación flexible frente a cambios imprevistos y optimiza continuamente la entrega de resultados parciales en cada fase del proyecto.

Dada la naturaleza tecnológica y experimental del trabajo, especialmente en lo referido a la integración de inteligencia artificial y gestión de bases de datos patrimoniales, Scrum ha resultado especialmente adecuada para ajustar continuamente el desarrollo en función de pruebas, validaciones y retroalimentación constante.

3.1 Metodología

El desarrollo del proyecto se ha estructurado mediante ciclos iterativos conocidos como *sprints*, cuya duración ha variado entre dos y seis semanas, dependiendo de la complejidad específica de las tareas definidas. Al inicio del proyecto, se elaboró un *product backlog*, una lista ordenada y priorizada de las tareas esenciales necesarias para alcanzar los objetivos planteados. En cada *sprint*, se seleccionaban aquellas tareas de mayor prioridad y relevancia en función de los objetivos inmediatos del proyecto.

La dinámica habitual dentro de cada sprint consistió en una planificación inicial clara, reuniones semanales de seguimiento y coordinación con el tutor para evaluar el progreso, identificar obstáculos y resolver posibles bloqueos, además de revisiones sistemáticas al finalizar cada iteración. Estas revisiones permitieron evaluar los avances logrados, realizar ajustes necesarios y definir estrategias de mejora para siguientes ciclos de desarrollo.

Scrum define claramente tres roles fundamentales dentro de su estructura metodológica: el *Scrum Master*, quien facilita el proceso eliminando impedimentos y asegurando una dinámica efectiva de trabajo; el *Product Owner*, encargado de definir, priorizar y ajustar continuamente las tareas del *product backlog* asegurando que el trabajo aporte valor; y los desarrolladores, responsables directos de la implementación de cada tarea.

Debido al carácter individual del desarrollo técnico del proyecto, asumí personalmente los roles de *Scrum Master* y desarrollador. Como *Scrum Master*, gestioné eficientemente el tiempo y los recursos disponibles, asegurando la ejecución ordenada y efectiva de cada sprint. Como desarrollador, fui responsable de la implementación técnica de las funcionalidades, realizando pruebas y validaciones, así como ajustes basados en los resultados obtenidos.

Por otro lado, Javier Algarra, mi tutor e investigador principal del proyecto en representación de la Comunidad de Madrid, asumió el rol de *Product Owner*, identificando claramente los objetivos del proyecto, definiendo y priorizando adecuadamente las tareas del backlog y guiando el desarrollo a través de revisiones periódicas y retroalimentación continua tras cada demostración.

3.1.1 Product Backlog

El *product backlog* es una herramienta esencial en Scrum que consiste en una lista dinámica y continuamente actualizada de tareas y requisitos, representados como historias de usuario o elementos específicos a desarrollar durante el proyecto. Cada elemento del *backlog* se describe con suficiente detalle para garantizar su efectiva implementación, abarcando aspectos desde nuevas funcionalidades hasta correcciones técnicas, optimización de procesos y ajustes necesarios para asegurar una correcta integración con bases de datos y sistemas externos.

Durante todo el desarrollo del proyecto, el *product backlog* se ha revisado y ajustado periódicamente en función del progreso realizado, la identificación de nuevas necesidades y los resultados obtenidos en cada sprint. Esta gestión dinámica ha permitido adaptarse eficazmente a los desafíos técnicos y operativos que surgían durante el desarrollo, garantizando que las tareas de mayor impacto y prioridad se implementaran en primer lugar y que cualquier mejora adicional se realizara de manera eficiente y oportuna.

Gracias al uso del *product backlog* como mecanismo central de planificación y seguimiento, el desarrollo del proyecto se ha mantenido estructurado, transparente y orientado claramente hacia los objetivos establecidos, facilitando así la gestión efectiva del tiempo y la toma ágil de decisiones durante todas las fases del trabajo.

3.1.2 Tareas e historias de usuario del Product Backlog

3.1.2.1 Historias de Usuario

3.1.2.1.1 Fase 1: Ingeniería Inversa de la Base de Datos

Como desarrollador, quiero extraer con precisión la estructura de la base de datos de la Comunidad de Madrid utilizando Microsoft Access para comprender claramente la organización de tablas, columnas y tipos de datos.

Como analista, quiero contar con documentación completa y clara en formato *Markdown* sobre la estructura y relaciones entre tablas, facilitando la identificación de posibles mejoras y evitando redundancias.

Como administrador de bases de datos, quiero verificar y asegurar que los tipos de datos estén definidos correctamente para prevenir errores durante la inserción de información, especialmente en casos específicos como números de teléfono.

3.1.2.1.2 Fase 2: Creación del ORM

Como desarrollador, quiero contar con un ORM personalizado que abstraiga eficazmente la complejidad inherente al acceso y manejo de la base de datos en Microsoft Access, para enfocar el esfuerzo en la lógica del negocio.

Como usuario final (otro equipo de desarrollo), quiero disponer de métodos sencillos para realizar operaciones CRUD, facilitando así la integración de nuevas funcionalidades sin conocer en detalle la estructura interna.

Como administrador de bases de datos, quiero asegurar que el ORM cumpla con patrones de diseño como Singleton o Factory, garantizando la mantenibilidad y escalabilidad del sistema.

Como desarrollador, deseo que el ORM genere y ejecute consultas SQL mediante una sintaxis familiar para reducir la curva de aprendizaje y facilitar futuras modificaciones.

3.1.2.1.3 Fase 3: Obtención de Datos

Como analista de datos, quiero importar y exportar datos en formatos .csv y .json mediante el ORM, facilitando el análisis y reutilización de la información en otras plataformas.

Como usuario final, quiero validar y limpiar los datos obtenidos de fuentes externas antes de su inserción para evitar inconsistencias o duplicaciones.

Como responsable de cumplimiento normativo, quiero asegurar que los datos recopilados respeten las regulaciones legales aplicables, protegiendo la privacidad y derechos de autor.

3.1.2.1.4 Fase 4: Modelo de Inteligencia Artificial

Como analista, quiero implementar un modelo de inteligencia artificial para automatizar eficientemente la recopilación y estructuración de nueva información sobre los bienes culturales.

Como desarrollador, quiero adaptar un modelo de IA existente mediante fine tuning para que se ajuste específicamente a las características y requerimientos del proyecto, generando resultados compatibles con el ORM.

Como administrador de la base de datos, quiero que el modelo de IA pueda conectarse a fuentes de datos en tiempo real para mantener la información permanentemente actualizada.

Como responsable del proyecto, deseo definir claramente las reglas y comportamientos del modelo de IA para garantizar que genere resultados estructurados y útiles en formato .csv.

3.1.2.2 Tareas Relacionadas

- Revisar detalladamente la documentación oficial de Microsoft Access.
- Investigar y adaptar consultas SQL específicas de Access para ingeniería inversa.
- Extraer mediante consultas SQL la lista completa de tablas, columnas y tipos de datos de la base de datos.
- Documentar en Markdown cada tabla, campo, relación y restricción existente.
- Definir clases en el ORM para representar fielmente las tablas y columnas de la base de datos.
- Implementar métodos CRUD claramente definidos en el ORM.
- Desarrollar la lógica necesaria para la conversión personalizada de tipos de datos.
- Crear y validar un sistema para traducir consultas en una sintaxis similar a SQL tradicional hacia sentencias SQL específicas de Access.

- Implementar funciones para importar y exportar registros en formatos .csv y .json con validaciones esenciales.
- Desarrollar un módulo específico para validar la calidad de los datos (tipos, duplicados y coherencia).
- Analizar y configurar correctamente la API de OpenAI con credenciales específicas para el proyecto.
- Preparar instrucciones detalladas para guiar al modelo IA.
- Realizar fine tuning del modelo IA con ejemplos específicos y reglas claras.
- Realizar pruebas prácticas del modelo IA verificando la compatibilidad total con el ORM.
- Configurar entornos diferenciados para desarrollo, pruebas y producción.
- Definir planes detallados para monitoreo continuo y control de versiones.
- Realizar una revisión exhaustiva de cumplimiento normativo sobre los datos recolectados.

3.2 Tecnologías empleadas

Para llevar a cabo el desarrollo de este proyecto, se ha realizado una cuidadosa selección de tecnologías específicas con el objetivo de facilitar la implementación y lograr satisfactoriamente los objetivos planteados inicialmente. La elección de estas herramientas no ha sido aleatoria, sino que ha surgido tras un análisis profundo y comparativo de diferentes alternativas, considerando aspectos fundamentales como la eficiencia operativa, la compatibilidad con los requerimientos específicos del proyecto y mi propia experiencia previa con cada una de ellas.

Cada tecnología ha sido seleccionada estratégicamente con el propósito de optimizar el flujo de trabajo general, mejorar notablemente la escalabilidad del sistema y asegurar una integración eficiente con las bases de datos patrimoniales destinadas a la catalogación de centrales telefónicas. Además, la experiencia adquirida en proyectos anteriores ha sido decisiva para tomar decisiones informadas respecto a las herramientas tecnológicas que finalmente han sido adoptadas.

A lo largo del desarrollo del proyecto, se han utilizado desde tecnologías básicas y fundamentales para la construcción del sistema hasta soluciones más avanzadas orientadas hacia la automatización del procesamiento de datos y la integración efectiva con sistemas patrimoniales. Estas herramientas tecnológicas han permitido agilizar significativamente el desarrollo del proyecto, asegurar la fiabilidad del sistema implementado y facilitar en gran medida su mantenimiento futuro.

3.2.1 Git

Git es un sistema distribuido de control de versiones, cuya principal función consiste en gestionar eficazmente los cambios realizados sobre el código. Su uso ha resultado esencial para mantener un registro exhaustivo y detallado de todas las modificaciones efectuadas, permitiendo así la reversión de errores, la organización estructurada del trabajo mediante ramas independientes, y realizar copias de seguridad frecuentes del código fuente. De este modo, Git ha permitido experimentar nuevas funcionalidades sin comprometer la estabilidad de la versión principal y documentar de forma clara cada fase del proceso de desarrollo.

3.2.2 PyCharm

Para la escritura y depuración del código, se ha empleado PyCharm, un entorno de desarrollo integrado (IDE) especializado en Python. PyCharm ofrece un conjunto avanzado de herramientas que han facilitado enormemente la organización del código, la detección temprana de errores y la optimización general del proceso de desarrollo. Algunas de sus características más útiles para este proyecto han sido el resaltado de sintaxis, el autocompletado inteligente, la integración directa con Git y la posibilidad de ejecutar scripts en un entorno controlado dentro del propio IDE.

3.2.3 Python como lenguaje de programación

Python fue seleccionado como lenguaje principal para el desarrollo del proyecto debido principalmente a su gran versatilidad y facilidad de aprendizaje y uso. Gracias a su extensa comunidad de desarrolladores y a la amplia variedad de bibliotecas disponibles, Python ha facilitado la realización de tareas complejas como la conexión con bases de datos o el procesamiento eficiente de datos. Además, su sintaxis limpia y legible ha contribuido significativamente a mejorar la mantenibilidad del código a largo plazo.

3.2.4 Python como intérprete

El intérprete de Python ha sido el responsable de ejecutar el código fuente, transformando las instrucciones escritas en el lenguaje Python a instrucciones ejecutables por el sistema operativo en tiempo de ejecución. La selección adecuada del intérprete ha sido clave para asegurar la compatibilidad completa con las bibliotecas necesarias y garantizar un rendimiento óptimo durante la ejecución de los scripts desarrollados.

3.2.5 Qué es un ORM y su utilidad en la gestión de bases de datos

Un Object-Relational Mapper (ORM) es una herramienta diseñada para facilitar la interacción con bases de datos relacionales mediante un enfoque orientado a objetos, eliminando así la necesidad de escribir manualmente consultas SQL.

Los ORMs actúan como una capa de abstracción entre la aplicación y la base de datos, permitiendo definir modelos de datos como objetos dentro del lenguaje utilizado. Esto simplifica considerablemente la manipulación y gestión de registros, evitando la necesidad de usar sentencias SQL directamente (Fowler, 2003).

En la práctica, un ORM convierte operaciones realizadas sobre objetos en instrucciones SQL comprensibles por la base de datos, permitiendo gestionar estructuras de datos en memoria de forma sencilla y abstracta, sin tener que preocuparse por detalles específicos del motor de base de datos subyacente.

Entre las principales ventajas del uso de ORMs están la abstracción y simplificación del acceso a los datos, evitando errores al reducir consultas SQL manuales; la portabilidad, permitiendo utilizar un mismo ORM con diversos motores de base de datos; mayor seguridad mediante consultas parametrizadas que reducen riesgos de inyección SQL; y una gestión eficiente de las relaciones entre modelos sin interactuar directamente con tablas.

3.2.6 Limitaciones de los ORMs convencionales con Microsoft Access

Aunque los ORMs están ampliamente extendidos en aplicaciones con bases de datos relacionales, su implementación puede presentar dificultades en entornos donde la base de datos no sigue los estándares modernos o carece de soporte adecuado (Ambler, 2012).

En este proyecto, la base de datos utilizada es Microsoft Access, específicamente en formato .mdb, lo cual presenta varios desafíos en su integración con un ORM tradicional.

En primer lugar, los ORM estándar no ofrecen soporte directo para Access, lo cual impide su aplicación inmediata. Además, la comunicación con Microsoft Access implica utilizar controladores ODBC (Open Database Connectivity), añadiendo una capa adicional de complejidad. Otro desafío radica en la estructura heredada de esta base de datos, que no sigue estrictamente las normas modernas de normalización, complicando su adaptación a ORMs convencionales.

Debido a estas limitaciones, se decidió desarrollar un ORM personalizado capaz de interactuar eficientemente con Access sin perder las ventajas propias de la programación orientada a objetos, facilitando así la gestión y manipulación de datos.

3.2.7 Microsoft Access y archivos .mdb

Microsoft Access es un sistema de gestión de bases de datos relacional (RDBMS) desarrollado por Microsoft, ampliamente utilizado en entornos pequeños y medianos gracias a su facilidad de uso e integración con el ecosistema de Microsoft Office.

Este sistema permite almacenar datos en archivos con extensión .mdb (Microsoft, 2022) los cuales incluyen tanto la estructura de la base de datos como los datos y consultas predefinidas.

Históricamente, los archivos .mdb han sido empleados en aplicaciones de escritorio y sistemas de gestión documental, aunque su popularidad ha disminuido en favor de soluciones más robustas como SQL Server o PostgreSQL.

Sin embargo, muchas administraciones públicas y empresas continúan utilizando bases de datos en este formato debido a su compatibilidad con sistemas heredados, lo que justifica su persistencia en determinados entornos de trabajo.

3.2.8 Py ODBC

Para conectar Python con Microsoft Access he utilizado Py ODBC (Golke, 2024), una biblioteca que permite interactuar con bases de datos mediante el estándar ODBC (Open Database Connectivity).

Gracias a PyODBC, ha sido posible ejecutar consultas SQL desde Python, acceder a los datos almacenados en Access y manipular la información de manera automatizada. Su uso ha sido fundamental para integrar el sistema de inteligencia artificial con la base de datos patrimonial, permitiendo la carga y extracción de información de forma eficiente.

3.2.9 Obsidian

Para la gestión de notas y documentación del proyecto he utilizado Obsidian, una herramienta de toma de apuntes basada en archivos Markdown que permite organizar y estructurar la información de manera eficiente. Su uso ha sido clave para documentar avances, registrar decisiones técnicas y mantener un repositorio accesible de información relevante para el desarrollo.

Gracias a su funcionalidad de enlaces entre notas y su flexibilidad en la organización de contenidos, he podido estructurar de forma clara los requerimientos del proyecto, diagramas de arquitectura y registros de pruebas, facilitando la trazabilidad y el seguimiento de las diferentes fases del trabajo.

3.3 Versiones de las tecnologías

Python	^3.11
pyodbc	^5.2.0
Obsidian	1.8.7
Git	2.42.0
Pycharm	2024.3.1
Microsoft Access	16.0.1

4. DESARROLLO DEL TRABAJO

El desarrollo del proyecto se estructura en cuatro fases principales claramente diferenciadas: la ingeniería inversa realizada sobre la base de datos proporcionada por la Comunidad de Madrid, la creación del ORM (Object-Relational Mapping), la obtención y almacenamiento de datos en la base de datos, y finalmente, el desarrollo de un modelo de inteligencia artificial orientado a facilitar futuras tareas de obtención de datos.

4.1 Ingeniería inversa

El proceso de ingeniería inversa aplicado a la base de datos mencionada consistió en extraer y analizar detalladamente la estructura existente utilizando Microsoft Access como herramienta de gestión de bases de datos. Aunque previamente existía conocimiento general sobre Access, su empleo específico presentó diversos obstáculos, principalmente relacionados con su interfaz poco intuitiva y su complejidad operativa.

Los principales desafíos encontrados durante este proceso incluyeron la dificultad para extraer de forma directa y sencilla toda la información necesaria para replicar correctamente la estructura original de la base de datos utilizada por la Comunidad de Madrid. Estos obstáculos fueron resueltos recurriendo directamente al uso de consultas SQL (Structured Query Language), lo que permitió obtener la información requerida con mayor precisión y eficiencia.

Para garantizar la correcta utilización de SQL dentro del entorno de Access, se consultó la documentación oficial (Microsoft, s. f.), debido a las posibles particularidades específicas del lenguaje adaptado por esta suite.

Este proceso de obtención y reconstrucción de la estructura de la base de datos conllevó una curva de aprendizaje algo peculiar, no debido al propio lenguaje SQL, sino más bien a las particularidades inherentes a la suite de Microsoft, la cual parece estar especialmente orientada hacia usuarios con cierto grado de familiarización previa.

El resultado de este trabajo se concretó en la creación de un índice exhaustivo que detalla todas las tablas existentes en dicha base de datos, así como las columnas pertenecientes a cada tabla, especificando claramente el tipo de dato almacenado en cada columna.

Este detalle resulta sumamente relevante debido a las implicaciones prácticas que tiene al momento de interactuar con la base de datos. Por ejemplo, en un caso práctico sencillo, si la base de datos almacena números de teléfono como números enteros, intentar añadir un prefijo como '+34' generaría errores o excepciones indeseadas que afectarían negativamente la experiencia de usuario.

Otro aspecto de gran importancia que fue posible extraer durante este proceso fueron las relaciones existentes entre las distintas tablas de la base de datos. Estas relaciones permiten conectar los datos evitando redundancias y facilitando referencias entre diferentes tablas. Cabe destacar que, en este caso concreto, las relaciones fueron creadas de forma manual por los desarrolladores originales, en lugar de utilizar las capacidades automáticas proporcionadas por los sistemas de gestión de bases de datos relacionales.

Esto se deduce ya que en la base de datos relacionales se guardan referencias a otras tablas usando *claves foráneas* o *foreign keys*, las cuales permiten saber cómo se relacionan los datos, sin embargo, en esta base de datos no hay *claves foráneas*, simplemente son cadenas de texto a las cuales se les asigna una relación, pero no a nivel de base de datos, sino a nivel de programa, lo cual puede dar varios problemas de referencias o incluso incongruencia en los datos.

Esta decisión podría claramente mejorarse o al menos optimizarse mediante una adecuada documentación y aplicación de las mejores prácticas recomendadas en el diseño de bases de datos relacionales, como se expone en la literatura sobre diseño de bases de datos (VidelCloud, 2017).

En la siguiente imagen se puede apreciar una representación clara y sencilla de la estructura mencionada, mostrando el nombre de cada tabla, sus respectivas columnas, el tipo de dato almacenado en cada columna, así como la cantidad de bytes necesarios para su almacenamiento en memoria. Cabe mencionar, además, que he documentado detalladamente esta estructura en archivos markdown, incluyendo explicaciones claras sobre el significado y función específica que tiene cada columna dentro del programa INPHIS, ya que en algunos casos los nombres empleados para estas columnas podrían resultar poco descriptivos o intuitivos.

```

1  {
2  >   "CULT_BIENES": [ 60 elements... ],
244 >   "CULT_BIENES_BIBLIO": {
245 >     "OBJECTID": {
246 >       "type": "<class 'int'>",
247 >       "bytes": 10
248 >     },
249 >     "CD_CODIGO": { adriantoral, 27/02/2025 14:00 · fe
250 >       "type": "<class 'str'>",
251 >       "bytes": 25
252 >     },
253 >     "TL_ANNO": {"type": "<class 'str'>" ...},
257 >     "CL_TITULO": {"type": "<class 'str'>" ...},
261 >     "CL_EDITORIAL": {"type": "<class 'str'>" ...},
265 >     "CL_LUGAR_EDICION": {"type": "<class 'str'>" ...},
269 >     "TL_EXTENSION": {"type": "<class 'str'>" ...},
273 >     "CL_AUTORIA": {"type": "<class 'str'>" ...},
277 >     "NM_ORDEN": {"type": "<class 'int'>" ...}
281 >   },
282 >   "CULT_BIENES_CATALOGO": [ 12 elements... ],
332 >   "CULT_BIENES_DEPOSITO": [ 6 elements... ],
358 >   "CULT_BIENES_DOC_LEGAL": [ 9 elements... ],
396 >   "CULT_BIENES_INVESTIGA": [ 8 elements... ],
430 >   "CULT_BIENES_PICKLIST": [ 5 elements... ],
452 >   "CULT_BIENES_POLIGONO": [ 6 elements... ],
478 >   "CULT_BIENES_PRO_PATRI": [ 6 elements... ],
504 >   "CULT_BIENES_PRO_URBAN": [ 7 elements... ],
534 >   "CULT_ENTORNO": [ 7 elements... ],
564 >   "CULT_POLIGONO": [ 6 elements... ],
590 >   "CULT_VAR_ACTIVIDAD": [ 4 elements... ],
608 >   "CULT_VAR_CAUSAS_ESTADO": [ 4 elements... ],
626 >   "CULT_VAR_CLASIF_SUELO": [ 4 elements... ],
644 >   "CULT_VAR_CONSTRUCCION": [ 4 elements... ],
662 >   "CULT_VAR_CULTURA": [ 4 elements... ],
680 >   "CULT_VAR_DOMINIO": [ 4 elements... ],
698 >   "CULT_VAR_ESTADO": [ 4 elements... ],
716 >   "CULT_VAR_ESTILO": [ 4 elements... ],
734 >   "CULT_VAR_INTERES": [ 4 elements... ],
752 >   "CULT_VAR_INVESTIGACIONES": [ 4 elements... ],
770 >   "CULT_VAR_LUGAR_DEPOSITO": [ 4 elements... ],
788 >   "CULT_VAR_MAT_ARQUEOLOG": [ 4 elements... ],
806 >   "CULT_VAR_MUEBLES": [ 4 elements... ],
824 >   "CULT_VAR_MUNICIPIOS": [ 4 elements... ],
842 >   "CULT_VAR_PERIODO": [ 4 elements... ],
860 >   "CULT_VAR_PROT_BICS": [ 4 elements... ],
878 >   "CULT_VAR_PROT_CATA": [ 4 elements... ],
896 >   "CULT_VAR_PROT_FIGURA": [ 4 elements... ],
914 >   "CULT_VAR_PROT_PLANEA": [ 4 elements... ],
932 >   "CULT_VAR_TAXONOMIA": [ 4 elements... ],
950 >   "CULT_VAR_TIPO_ACTO": [ 4 elements... ],
968 >   "CULT_VAR_TIPOLOGIA": [ 4 elements... ]
986 > }

```

Figura 1. Índice creado a partir de la base de datos, muestra todas las tablas y columnas existentes en la base de datos, así como su tipo de valor y los bytes que ocupa en memoria.

4.2 Creación del ORM

Una vez definida la estructura de la base de datos, se continuo con el desarrollo del ORM. Antes de iniciar la programación propiamente dicha, consideré fundamental documentarme sobre enfoques previos adoptados por profesionales en el ámbito del desarrollo de software.

Los aspectos en los que centré principalmente mi atención para la primera versión del ORM fueron el sistema de modelado de datos, la generación sencilla e intuitiva de consultas, y los procesos de serialización y deserialización de los datos. Estos tres puntos fueron considerados esenciales para disponer de un Producto Mínimo Viable (MVP) que pudiera ser claramente presentado. A continuación, detallo cada uno de estos aspectos para ofrecer un contexto más preciso.

El sistema de modelado permite definir claramente la estructura de una base de datos utilizando clases dentro del código, proporcionando un nivel de abstracción que permite al usuario final utilizar la aplicación sin preocuparse por la implementación concreta de la conexión con la base de datos. De esta forma, el usuario puede interactuar fácilmente con el sistema mediante funciones específicas que garantizan el correcto funcionamiento interno del ORM.

Más adelante detallaré específicamente mi caso de uso con la base de datos de la Comunidad de Madrid, pero inicialmente, para clarificar mejor este concepto, presento un ejemplo general: en el supuesto caso en el que una base de datos contenga una tabla de usuarios, el desarrollador no necesita conocer los detalles del almacenamiento de estos datos, simplemente desea almacenarlos. Mi rol como responsable de la base de datos consiste en ofrecer piezas de código que permitan crear, modificar, leer y eliminar usuarios, como se ilustra a continuación:

```

class User:
    """
    Modelo de usuario con funciones CRUD.
    """

    def __init__( self, id: int, name: str, email: str, created_at: str ):
        self.id = id
        self.name = name
        self.email = email
        self.created_at = created_at

    @classmethod
    def create ( cls, name: str, email: str ):
        """Crea un nuevo usuario y lo guarda en la base de datos."""
        pass

    @classmethod
    def read ( cls, user_id: int = None ):
        """Lee uno o todos los usuarios dependiendo del parámetro."""
        pass

    @classmethod
    def update ( cls, user_id: int, name: str = None, email: str = None ):
        """Actualiza los datos de un usuario existente."""
        pass

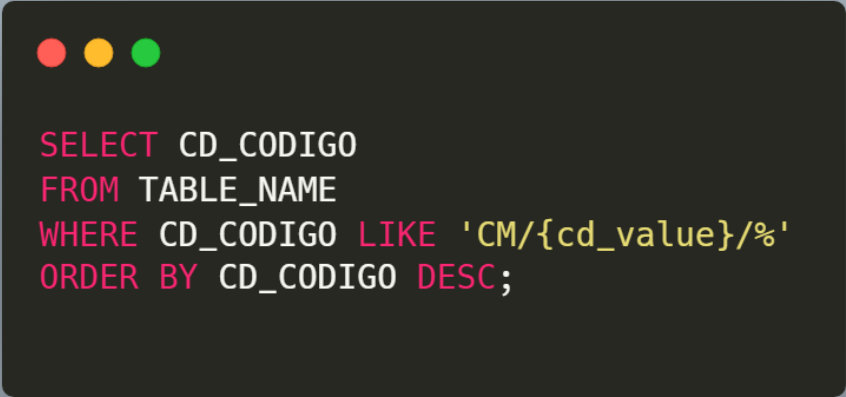
    @classmethod
    def delete ( cls, user_id: int ):
        """Elimina un usuario por su ID."""
        pass

```

Figura 2. Ejemplo de una clase de un ORM que abstrae al desarrollador para interactuar con la base de datos, simplemente se le exponen unas funciones a las que puede llamar para manipular la base de datos sin conocer el trasfondo.

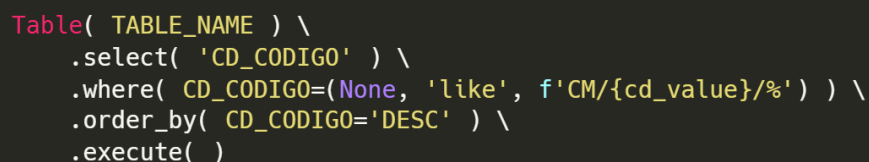
Este fragmento de código permite al desarrollador llamar a funciones específicas que interactúan con la base de datos sin necesidad de conocer los detalles técnicos del proceso. Considero que esta abstracción es un elemento clave en el diseño de un ORM personalizado, ya que facilita considerablemente su uso y mantenimiento futuro al reducir la responsabilidad técnica del desarrollador.

El siguiente paso fue diseñar un sistema de consultas que permitiese interactuar con la base de datos de manera similar al lenguaje SQL, ampliamente conocido y utilizado por desarrolladores. Al imitar la estructura de consultas SQL, el ORM gana adaptabilidad, permitiendo que cualquier persona familiarizada mínimamente con SQL pueda emplearlo con facilidad. A continuación, muestro un ejemplo comparativo entre una consulta SQL tradicional y su equivalente en mi sistema personalizado:



```
SELECT CD_CODIGO
FROM TABLE_NAME
WHERE CD_CODIGO LIKE 'CM/{cd_value}/%'
ORDER BY CD_CODIGO DESC;
```

Figura 3. Consulta SQL básica para obtener CD_CODIGO de la tabla TABLE_NAME con una condición de que CD_CODIGO tiene que ser parecido a CM/{}/% y ordenado de manera descendente.

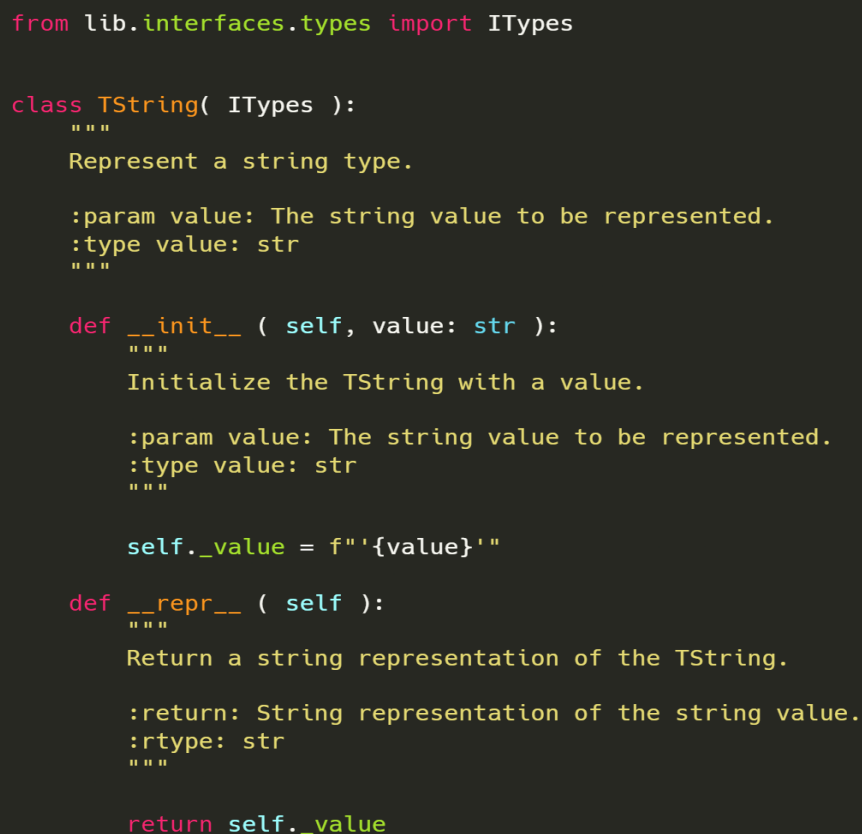


```
Table( TABLE_NAME ) \
.select( 'CD_CODIGO' ) \
.where( CD_CODIGO=(None, 'like', f'CM/{cd_value}/%') ) \
.order_by( CD_CODIGO='DESC' ) \
.execute( )
```

Figura 4. Consulta con el sistema de consultas creado haciendo analogía a la consulta previa.

Puede observarse claramente una similitud en la estructura y formato de ambas consultas, debido a que mi implementación se basa directamente en SQL. Esto resulta especialmente importante, ya que garantiza una curva de aprendizaje muy baja para futuros desarrolladores encargados del mantenimiento del código.

Internamente, este sistema de consultas transforma las instrucciones del ORM a código SQL puro. Por lo tanto, es crucial manejar adecuadamente la inserción de datos, especialmente aquellos que necesitan ir entre comillas en SQL, como las descripciones textuales. Para resolver este aspecto, utilicé métodos especiales que ofrece Python, como `__str__` y `__repr__`, que permiten modificar la representación de objetos cuando son convertidos a cadenas o representados mediante instancias.



```
from lib.interfaces.types import ITypes

class TString( ITypes ):
    """
    Represent a string type.

    :param value: The string value to be represented.
    :type value: str
    """

    def __init__( self, value: str ):
        """
        Initialize the TString with a value.

        :param value: The string value to be represented.
        :type value: str
        """

        self._value = f'"{value}"'

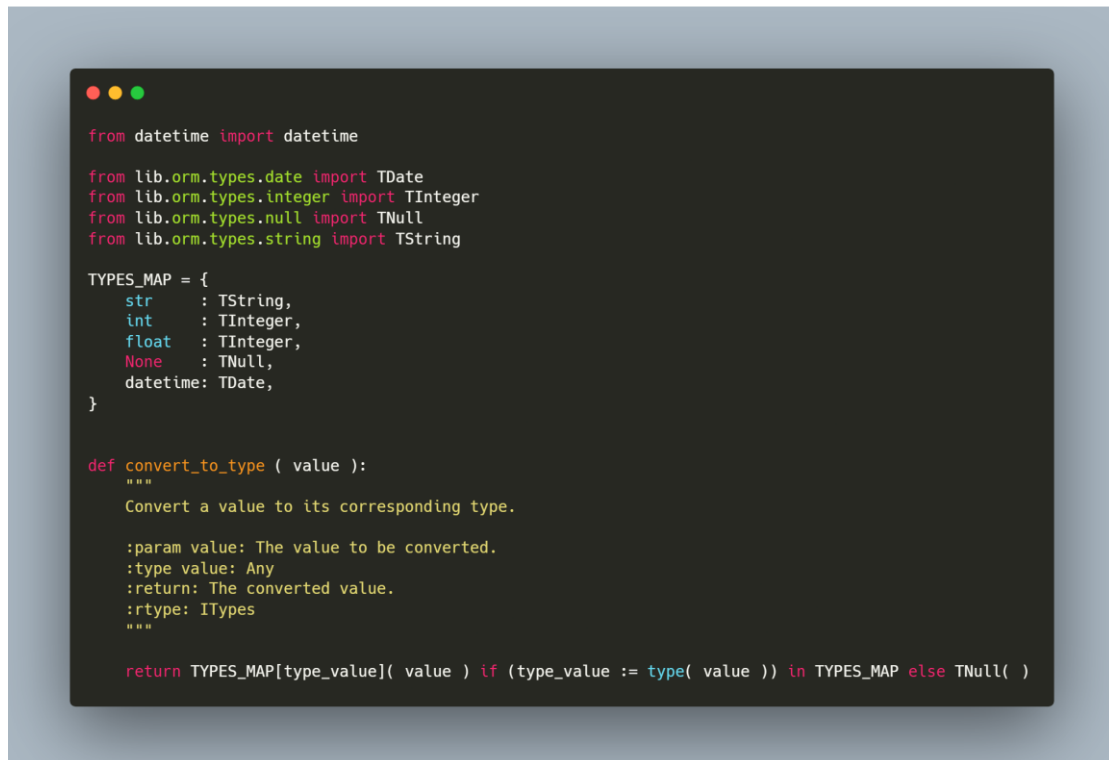
    def __repr__( self ):
        """
        Return a string representation of the TString.

        :return: String representation of the string value.
        :rtype: str
        """

        return self._value
```

Figura 5. Representación de un tipo propio del ORM que facilita el guardado en base de datos.

A continuación, se presenta un ejemplo que ilustra cómo debe representarse correctamente un tipo string en la base de datos, agregando comillas automáticamente. Adicionalmente, desarrollé una función específica para transformar los tipos originales de Python al tipo personalizado TString, facilitando así la conversión automática requerida por la base de datos:



```
from datetime import datetime

from lib.orm.types.date import TDate
from lib.orm.types.integer import TInteger
from lib.orm.types.null import TNull
from lib.orm.types.string import TString

TYPES_MAP = {
    str      : TString,
    int      : TInteger,
    float    : TInteger,
    None     : TNull,
    datetime: TDate,
}

def convert_to_type ( value ):
    """
    Convert a value to its corresponding type.

    :param value: The value to be converted.
    :type value: Any
    :return: The converted value.
    :rtype: ITypes
    """

    return TYPES_MAP[type_value]( value ) if (type_value := type( value )) in TYPES_MAP else TNull( )
```

Figura 6. Mapeo de tipos de los propios de Python a los usados en el sistema ORM, asegurando que los tipos usados en las consultas sean seguros y controlados.

Esta función mapea el tipo estándar de Python al tipo personalizado del ORM. En caso de no existir dicha correspondencia, simplemente devuelve un valor nulo, omitiendo su procesamiento. Para garantizar la reutilización de código y facilitar su mantenimiento, implementé diversos patrones de diseño como Singleton, Factory y Herencia, entre otros. En la siguiente imagen se muestra claramente la interfaz implementada en cada modelo de la base de datos.

Específicamente, se definió un modelo como la combinación lógica de una tabla completa con una fila individual, agrupando la funcionalidad de lectura y escritura en una sola interfaz para simplificar la implementación y comprensión.

```

from abc import abstractmethod
from typing import Any, Type

from lib.orm.tables.table import Table
from lib.orm.utils.build_query import build_query_columns

class IModels:
    """
    Interface for models.
    """

    @staticmethod
    @abstractmethod
    def create ( *args, **kwargs ):
        """
        Abstract method to create a new instance of the model.
        """

        pass

    @staticmethod
    @abstractmethod
    def read ( *args, **kwargs ):
        """
        Abstract method to read existing instances of the model.
        """

        pass

    def _save (
        self,
        table_name: str,
        model_cls: Type['IModels'],
        is_update: bool,
        update_key_value: tuple[str, Any],
        read_value: str,
        ignore: list[str]
    ):
        """
        Save the current model instance to the database.

        :param table_name: The name of the table where the model instance will be saved.
        :type table_name: str
        :param model_cls: The class of the model being saved.
        :type model_cls: Type['IModels']
        :param is_update: Flag indicating whether the operation is an update.
        :type is_update: bool
        :param update_key_value: A tuple containing the key and value for the update condition.
        :type update_key_value: tuple[str, Any]
        :param read_value: The value used to read the model instance after saving.
        :type read_value: str
        :param ignore: A list of fields to ignore during the save operation.
        :type ignore: list[str]
        """

        data = build_query_columns( self.__dict__, ignore )

        if is_update:
            Table( table_name ) \
                .update( **data ) \
                .where( **{ update_key_value[0]: (None, '=', update_key_value[1]) } ) \
                .execute( )
        else:
            Table( table_name ) \
                .insert( **data ) \
                .execute( )

        self.__dict__.update( model_cls.read( read_value ).pop( ).__dict__ )

    @abstractmethod
    def save ( self ):
        """
        Abstract method to save the current model.
        """

        pass

    @abstractmethod
    def delete ( self ):
        """
        Abstract method to delete the current model instance.
        """

        pass

```

Figura 7. Interfaz del modelo a implementar en todas las tablas de la base de datos.

Este modelo implementa funciones que respetan los principios CRUD (Create, Read, Update, Delete), facilitando a los desarrolladores la creación y actualización de registros mediante un único método denominado ``save``, que distingue automáticamente entre ambas operaciones. Para ofrecer un contexto práctico, en la siguiente imagen se ilustra la aplicación concreta sobre una tabla denominada `CULT_BIENES`, que almacena información relevante sobre bienes culturales.

```

import datetime
from dataclasses import dataclass
from typing import Optional

from lib.interfaces.models import IModels
from lib.orm.tables.table import SELECT_ALL, Table

TABLE_NAME = 'CULT_BIENES'
IGNORE = [
    'objectid',
    'shape',
    'shape_length',
    'shape_area'
]

@dataclass
class CultBienesModel( IModels ):
    """
    Model to represent a cult_bien.
    """

    objectid: Optional[int] = None
    shape: Optional[bytearray] = None
    cd_codigo: Optional[str] = None
    tl_nombre: Optional[str] = None
    tl_dircalles: Optional[str] = None
    nm_dirnum: Optional[int] = None
    tl_localidad: Optional[str] = None
    tl_otros_nombres: Optional[str] = None
    cd_cod_ant: Optional[str] = None
    nm_utm_x: Optional[float] = None
    nm_utm_y: Optional[float] = None
    tl_geo_lon: Optional[str] = None
    tl_geo_lat: Optional[str] = None
    nm_altitud: Optional[int] = None
    nm_extension: Optional[int] = None
    cl_accesos: Optional[str] = None
    cl_des_general: Optional[str] = None
    nm_cronologia_inicio: Optional[int] = None
    nm_cronologia_fin: Optional[int] = None
    cl_just_atribucion: Optional[str] = None
    cl_des_bien: Optional[str] = None
    cl_des_muebles: Optional[str] = None
    cl_fuentes_escritas: Optional[str] = None
    cl_fuentes_carto: Optional[str] = None
    cl_fuentes_icono: Optional[str] = None
    cl_fuentes_orales: Optional[str] = None
    cl_uso_estado: Optional[str] = None
    tl_estado_porc_extraido: Optional[str] = None
    tl_figura2: Optional[str] = None
    tl_figura3: Optional[str] = None
    tl_figura4: Optional[str] = None
    tl_figura5: Optional[str] = None
    cl_observaciones: Optional[str] = None
    tl_autor: Optional[str] = None
    tl_supervisor: Optional[str] = None
    fc_autor_fecha_cumplimenta: Optional[datetime.datetime] = None
    fc_super_fecha_cumplimenta: Optional[datetime.datetime] = None
    geometry1_sk: Optional[str] = None
    tl_adjunto: Optional[str] = None
    id_referencia: Optional[int] = None
    tl_fecha_referencia: Optional[str] = None
    cd_yac_referencia: Optional[str] = None
    cd_catalogo_regional: Optional[str] = None
    cd_catalogo_urbanistico: Optional[str] = None
    fc_inscripcion_catalogo: Optional[datetime.datetime] = None
    tl_proteccion_arq_regional: Optional[str] = None
    tl_dir_postal: Optional[str] = None
    tl_dir_poligono: Optional[str] = None
    tl_referencia_catastral: Optional[str] = None
    cl_historia_bien: Optional[str] = None
    cl_obras_usos: Optional[str] = None
    tl_arca: Optional[str] = None
    cl_otros_codigos: Optional[str] = None
    fc_fecha_modificacion: Optional[datetime.datetime] = None
    shape_length: Optional[float] = None
    shape_area: Optional[float] = None
    geometry_bk: Optional[str] = None
    geometry_x_bk: Optional[float] = None
    geometry_y_bk: Optional[float] = None
    geometry_area_bk: Optional[float] = None

```

```

@staticmethod
def generate_cd_codigo ( cult_var_municipios_cd_values: list[str] ):
    """
    Generate a code for a new cult_bien.

    :param cult_var_municipios_cd_values: A list of municipality codes.
    :return: A generated code for the cult_bien.
    :rtype: str
    """

    cd_value = '000' if len( cult_var_municipios_cd_values ) > 1 else
cult_var_municipios_cd_values[0]
    data = Table( TABLE_NAME ) \
        .select( 'CD_CODIGO' ) \
        .where( CD_CODIGO=(None, 'like', f'CM/{cd_value}/%' ) ) \
        .order_by( CD_CODIGO='DESC' ) \
        .execute( )

    last_cd_codigo = data[0][0] if data else f'CM/{cd_value}/0000'
    prefijo, cd_value, id_contador = last_cd_codigo.split( '/' )

    return f'{prefijo}/{cd_value}/{str( int( id_contador ) + 1 ).zfill( 4 )}'

@staticmethod
def create ( tl_nombre: str, cult_var_municipios_cd_values: list[str] ):
    """
    Create a new instance of CultBienesModel.

    :param tl_nombre: The name of the cult_bien.
    :param cult_var_municipios_cd_values: A list of municipality codes.
    :return: An instance of CultBienesModel.
    :rtype: CultBienesModel
    """

    return CultBienesModel(
        cd_codigo=CultBienesModel.generate_cd_codigo( cult_var_municipios_cd_values ),
        tl_nombre=tl_nombre
    )

@staticmethod
def read ( cd_codigo: str ):
    """
    Read an existing cult_bien.

    :param cd_codigo: The code of the cult_bien.
    :return: A list of instances of CultBienesModel.
    :rtype: list[CultBienesModel]
    """

    data = Table( TABLE_NAME ) \
        .select( SELECT_ALL ) \
        .where( CD_CODIGO=(None, '=', cd_codigo) ) \
        .execute( )

    if not len( data ): raise ValueError( f'{cd_codigo} not found in {TABLE_NAME}' )
    return [CultBienesModel( *row ) for row in data]

def save ( self ):
    """
    Save the current cult_bien.
    """

    return self._save(
        TABLE_NAME,
        CultBienesModel,
        bool( self.objectid ),
        ('OBJECTID', self.objectid),
        self.cd_codigo,
        IGNORE
    )

def delete ( self ):
    """
    Delete the current cult_bien.

    :raises ValueError: If OBJECTID is not set.
    """

    if not self.objectid: raise ValueError( 'OBJECTID is required to delete' )
    return Table( TABLE_NAME ) \
        .delete( ) \
        .where( OBJECTID=(None, '=', self.objectid) ) \
        .execute( )

```

Figura 8. Ejemplo de un modelo de la base de datos implementando las funciones definidas en la interfaz previa.

Este ejemplo representa un caso complejo, debido a la necesidad de un identificador manual llamado `CD_CODIGO`. Este identificador combina varios elementos: un prefijo `CM/` (presumiblemente Comunidad de Madrid), un `cd_value` que indica la región del bien cultural (por ejemplo, 001, aunque si se seleccionan múltiples regiones se asigna 000), y finalmente un contador secuencial.

Por ejemplo, al añadir el primer registro en la región de Aravaca, el `CD_CODIGO` resultante sería `CM/001/0001`. Todo este proceso se ejecuta automáticamente mediante la función `generate_cd_codigo` durante la creación de registros.

Finalmente, tras implementar esta estructura en todas las tablas necesarias, la primera versión del ORM es plenamente funcional en términos de operaciones CRUD.

4.3 Obtención de los datos

Respecto al proceso de obtención de los datos, cabe mencionar que fue llevado a cabo, como parte del proyecto encargado por la CAM por dos estudiantes del Ciclo Formativo de Grado Superior, Alejandro Tineo y Alejandro López en Desarrollo de Aplicaciones Web, quienes realizaron un análisis exhaustivo utilizando fuentes fiables. De estas fuentes extrajeron información relevante y fundamental, como imágenes de los bienes culturales, descripciones detalladas de los mismos, coordenadas geográficas precisas de sus ubicaciones, entre otros elementos clave.

Por otro lado, mi responsabilidad consistió en la incorporación efectiva de estos datos en las tablas correspondientes de la base de datos mediante la utilización del ORM que previamente he explicado.

Actualmente, el ORM implementado permite realizar operaciones tanto de importación como exportación de datos en formatos estándar como `.csv` y `.json`. Esta funcionalidad resulta esencial y muy valiosa, especialmente si los datos almacenados se desean utilizar posteriormente en análisis estadísticos, investigaciones adicionales o incluso integrarlos en otras bases de datos externas.

4.4 Modelo de inteligencia artificial

Durante el desarrollo del trabajo, se planteó inicialmente la creación de un modelo de inteligencia artificial que fuese capaz de facilitar y automatizar parcialmente la recolección de datos. Para abordar esta idea, se consideraron dos opciones fundamentales:

- La primera opción contemplaba desarrollar un modelo completamente nuevo desde cero, que fuese entrenado específicamente para extraer conclusiones sobre las centrales para las que se dispusieran de datos previos. Esta opción se descartó rápidamente debido a sus claras limitaciones prácticas: principalmente, la ausencia significativa de datos históricos o previos sobre dichas centrales. Además, entrenar un modelo únicamente con datos limitados a las centrales conocidas reduciría notablemente su utilidad práctica, ya que un modelo de inteligencia artificial debería idealmente poder generalizar conocimientos hacia nuevos casos no entrenados.
- La segunda opción fue considerar un modelo preexistente que pudiera adaptarse mediante la técnica conocida como fine tuning o ajuste fino. El fine tuning, explicado de forma simple, consiste en adaptar un modelo previamente entrenado para que pueda interpretar nuevas entradas o información, relacionándolas eficientemente con los conocimientos que ya poseía. Esta opción resultó mucho más viable y prometedora.

Finalmente, se decidió utilizar una API (Interfaz de Programación de Aplicaciones) proporcionada por OpenAI, reconocida empresa en el ámbito de la inteligencia artificial. OpenAI ofrece planes específicos que permiten la creación personalizada de modelos propios a partir de los modelos ya existentes en su plataforma. Aunque este método no es exactamente igual al fine tuning clásico, proporciona funcionalidades similares.

Un aspecto particularmente ventajoso de este plan es la capacidad del modelo para conectarse a datos en tiempo real disponibles en Internet, asegurando así que la información obtenida siempre esté actualizada y refleje la realidad actual. OpenAI facilita una interfaz intuitiva que permite configurar adecuadamente el modelo, especificando con precisión las instrucciones necesarias para garantizar su funcionamiento según los requerimientos establecidos.

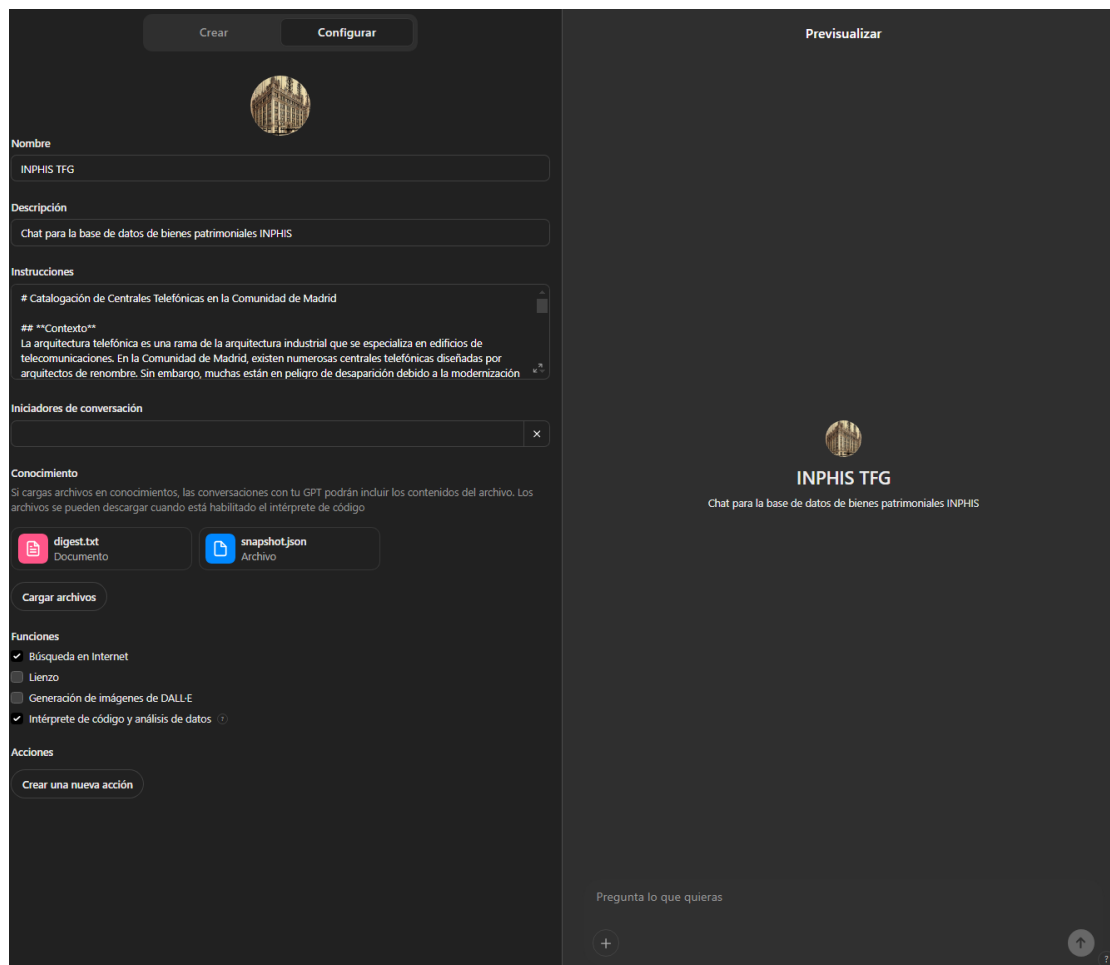


Figura 9. Interfaz de OpenAI para crear el modelo de inteligencia artificial personalizado.

El proceso para lograr esto no implica procedimientos extremadamente complejos, sino más bien se basa en proporcionar al modelo instrucciones claras sobre cómo debe comportarse ante determinados casos específicos de uso, estableciendo así una serie de reglas y comportamientos esperados. En el caso particular del presente trabajo, se requería específicamente que el modelo fuese capaz de recibir entradas referentes a una central concreta y que, posteriormente, devolviera automáticamente un archivo .csv perfectamente importable mediante el ORM desarrollado.

Para alcanzar este objetivo, proporcioné al modelo un resumen comprensible de la estructura previamente documentada de la base de datos, presentado en un formato accesible para su interpretación y comprensión. Además, suministré ejemplos concretos de datos de prueba que permitieron al modelo identificar claramente qué información específica debía almacenarse en cada campo de la base de datos. Junto con esta documentación, también establecí explícitamente las reglas operativas del modelo, definiendo claramente cómo debía actuar y qué resultados concretos debía producir ante diferentes tipos de solicitudes.

```
# Catalogación de Centrales Telefónicas en la Comunidad de Madrid

## **Contexto**
La arquitectura telefónica es una rama de la arquitectura industrial que se especializa en edificios de telecomunicaciones. En la Comunidad de Madrid, existen numerosas centrales telefónicas diseñadas por arquitectos de renombre. Sin embargo, muchas están en peligro de desaparición debido a la modernización tecnológica y la revalorización del suelo.

Este proyecto busca catalogar estos edificios singulares para la Dirección General de Patrimonio Cultural en la base de datos ETNOCAM, garantizando su correcta documentación y preservación.

## **Objetivo del GPT**
Tu objetivo es ayudar en la recopilación de datos sobre estos bienes culturales. Para ello, deberás:
1. Buscar y extraer información exclusivamente de fuentes verificadas en internet, como bases de datos oficiales de patrimonio, ministerios de cultura o registros históricos.
2. No utilizar `snapshot.json` como fuente de datos, sino únicamente como referencia para comprender la estructura y el tipo de valores aceptados en la base de datos.
3. Completar fichas en la base de datos siguiendo la estructura definida en el archivo `digest.txt`.
4. Generar salidas en formato CSV con los datos estructurados.

## **Rol del GPT**
Eres INPHIS TFG, un experto en patrimonio cultural y búsqueda de información en internet. Tu tarea es proporcionar información detallada y verificada sobre los bienes culturales que se te consulten.

### **Tipos de entrada de datos**
Puedes recibir varios tipos de entradas:
1. Consultas directas: preguntas específicas como "Háblame sobre {bien_cultural}" o "Dame una lista de los bienes culturales en {ubicación}". En estos casos, realiza una búsqueda en internet para obtener información actualizada y fiable antes de generar la respuesta.
2. Consultas de listados: solicitudes como "Lista de bienes culturales en {ciudad}", donde debes acceder a fuentes oficiales en internet y verificar la información antes de generar una respuesta. Asegúrate de filtrar correctamente por TL_LOCALIDAD y no devolver datos de otra ubicación.
3. Archivos CSV: listados de bienes con algunas columnas ya completadas. Tu tarea es completar la información faltante siguiendo la estructura de la base de datos.

Ejemplo de CSV de entrada:
'''csv
cd_codigo;tl_nombre;tl_dircalle;nm_dirnum
004-001;Casa de la Cultura;Av. 9 de Octubre;505
'''

- En este caso, debes rellenar todas las columnas restantes (excepto aquellas con descripción *None* en `digest.txt`).

## **Formato de Salida**
Siempre debes entregar los datos en formato CSV con la siguiente estructura:
'''csv
{COLUMNAS_DE_LA_TABLA_SEPARADAS_POR_,}
{DATOS_SEPARADOS_POR_,}
'''

Ejemplo de salida esperada:
'''csv
cd_codigo,tl_nombre,tl_dircalle,nm_dirnum,tl_localidad,nm_utm_x,nm_utm_y,tl_geo_lon,tl_geo_lat,nm_altitud,cl_des_general,tl_autor,nm_cronologia_inicio,nm_cronologia_fin
NAV/001/0001,Castillo de Olite,Calle del Palacio,1,Olite,607000,4748000,-1.650278,42.486389,400,Fortaleza medieval reconstruida en el siglo XX,Teobaldo I de Navarra,1269,1274
'''

Importante: No agregues texto explicativo, solo devuelve los datos en CSV con las columnas y datos requeridos en `digest.txt`.

### **Pautas para responder a consultas generales**
- Siempre realiza una búsqueda en internet antes de generar la respuesta. No utilices datos internos como referencia de conocimiento.
- Si la consulta es sobre una ubicación distinta a Madrid, no devuelvas datos de Madrid. Asegúrate de que los bienes listados pertenezcan a la ubicación solicitada.
- Si no encuentras datos en internet sobre la ubicación solicitada, responde indicando que no se encontraron datos en lugar de devolver información incorrecta.

Ejemplo cuando no hay datos:
'''csv
cd_codigo,tl_nombre,tl_dircalle,nm_dirnum,tl_localidad,nm_utm_x,nm_utm_y,tl_geo_lon,tl_geo_lat,nm_altitud,cl_des_general,tl_autor,nm_cronologia_inicio,nm_cronologia_fin
No se encontraron datos para esta consulta.
'''

- No generes datos aleatorios ni reutilices información de otras ubicaciones.
- Siempre valida el formato del CSV antes de devolver la respuesta para asegurarte de que todas las columnas obligatorias están presentes y cumplen con la estructura esperada.
```

Figura 10. Prompt usado para entrenar al modelo personalizado de inteligencia artificial.

Tras esta configuración completa del modelo, realicé pruebas específicas para verificar su rendimiento. El resultado de una de estas pruebas se muestra en el siguiente ejemplo, donde es evidente que el modelo puede estructurar correctamente la información solicitada en un formato .csv. Dicho archivo resultante puede ser importado directamente en la base de datos mediante la utilización del ORM anteriormente desarrollado.

Edificio Telefónica

Aquí tienes un listado de centrales telefónicas en la Comunidad de Madrid basado en los datos disponibles:

CSV

Copiar

```

cd_codigo,tl_nombre,tl_dircalle,nm_dirnum,tl_localidad,nm_utm_x,nm_utm_y,tl_geo_lon,tl_geo_la
CM/079/0029,CENTRAL TELEFÓNICA MADRID GRAN VÍA,Gran Vía,28,Madrid,440453.0,4474656.0,-3.70194
CM/096/0001,CENTRAL TELEFÓNICA NAVALCARNERO,San Roque,46,Navalcarnero,,,,-4.013078,40.285943,
CM/079/0001,CENTRAL TELEFÓNICA MADRID BELLAS VISTAS,La Rábida,19,Madrid,,,,-3.711754,40.45437
CM/079/0002,CENTRAL TELEFÓNICA MADRID PAVONES,Hacienda de Pavones s/n,,Madrid,,,,-3.635820,40
CM/079/0003,CENTRAL TELEFÓNICA MADRID OSUNA,Otoño,14,Madrid,451657.0,4477446.0,-3.570794,40.4
CM/079/0004,CENTRAL TELEFÓNICA MADRID MOSCARDÓ,Santa María de la Cabeza (Paseo),117,Madrid,43
CM/079/0005,CENTRAL TELEFÓNICA MADRID ALMENDRALES,Cristo de la Victoria,84,Madrid,440358.0,44
CM/079/0006,CENTRAL TELEFÓNICA MADRID CONCEPCIÓN,Demetrio López,2,Madrid,,,,-3.656136,40.4372
CM/079/0007,CENTRAL TELEFÓNICA MADRID MORATALAZ,Marroquina,43,Madrid,,,,-3.648648,40.408132,
CM/079/0008,CENTRAL TELEFÓNICA TORREJÓN DE ARDOZ,Avenida de La Constitución,11,Torrejón de Ar

```

Si necesitas más información o quieres un formato diferente, dime cómo lo prefieres.

dame un listado de edificios de telefonica en galicia

A continuación, se presenta un listado de edificios de Telefónica en Galicia, incluyendo sus características principales:

CSV

Copiar

```

cd_codigo,tl_nombre,tl_dircalle,nm_dirnum,tl_localidad,nm_utm_x,nm_utm_y,tl_geo_lon,tl_geo_la
GAL/150/0001,Central Telefónica de A Coruña,Calle San Andrés, ,A Coruña, , , -8.409,43.368, ,E
GAL/362/0001,Central Telefónica de Vigo,Calle Urzáiz, ,Vigo, , , -8.711,42.233, ,Edificio de e
GAL/157/0001,Central Telefónica de Santiago de Compostela,Avenida de Rodríguez de Viguri,49,Si
GAL/270/0001,Central Telefónica de Foz,Calle Salvador de Madariaga, ,Foz, , , -7.266,43.567, ,
GAL/150/0002,Central Telefónica de O Castrillón,Calle Montinho, ,A Coruña, , , -8.407,43.343, ,
GAL/362/0002,Centro de Talento y Tecnología de Vigo,Calle del Calvario, ,Vigo, , , -8.712,42.2
GAL/156/0001,Central Telefónica de Mugardos,Rúa Castelao, ,Mugardos, , , -8.255,43.459, ,Edifi

```

Figura 11. Ejemplo de una salida del modelo ya entrenado, devuelve un .csv el cual puede ser integrado en la base de datos usando el ORM.

5. CONCLUSIONES

Este Trabajo de Fin de Grado ha permitido abordar con éxito el reto de automatizar y optimizar la catalogación de centrales telefónicas en la Comunidad de Madrid mediante el uso de tecnologías avanzadas de inteligencia artificial y desarrollo de software especializado. Se han alcanzado satisfactoriamente todos los objetivos planteados, ofreciendo soluciones concretas a las limitaciones y desafíos existentes en la gestión tradicional del patrimonio arquitectónico industrial.

En primer lugar, se realizó una exhaustiva ingeniería inversa sobre la base de datos existente, lo que permitió obtener una representación detallada y precisa de su estructura. Esta fase proporcionó un entendimiento profundo sobre las particularidades técnicas del sistema utilizado por la Comunidad de Madrid, facilitando así posteriores procesos de desarrollo y garantizando la interoperabilidad con el nuevo sistema automatizado.

Además, la creación de un ORM personalizado supuso un hito importante dentro del proyecto, permitiendo abstraer eficazmente la complejidad inherente al acceso y manejo de datos en Microsoft Access. Este desarrollo simplifica notablemente las operaciones CRUD y facilita significativamente la interacción del usuario final con la base de datos, reduciendo errores operativos y agilizando las tareas administrativas.

Otro logro destacado del trabajo ha sido la integración exitosa de un modelo de inteligencia artificial basado en la plataforma OpenAI, ajustado específicamente a las necesidades del proyecto mediante técnicas de fine tuning. Este modelo ha demostrado ser altamente eficaz para automatizar la extracción y organización de información clave a partir de fuentes documentales, generando resultados en formato .csv compatibles directamente con el ORM desarrollado. La automatización conseguida supone un ahorro considerable de tiempo y esfuerzo frente a los métodos manuales tradicionales.

Desde un punto de vista práctico, la solución tecnológica propuesta tiene una utilidad real e inmediata para la Dirección General de Patrimonio Cultural de la Comunidad de Madrid, mejorando sustancialmente su capacidad para gestionar, conservar y documentar el patrimonio arquitectónico. La automatización y precisión alcanzadas proporcionan una herramienta robusta que permitirá preservar de forma efectiva un legado histórico y arquitectónico valioso, evitando la pérdida irreparable de información crítica.

Finalmente, este trabajo establece una sólida base tecnológica que permitirá futuras ampliaciones y mejoras, sirviendo como plataforma estratégica para proyectos académicos y profesionales posteriores. La continuidad de este proyecto es especialmente prometedora, abriendo posibilidades para el desarrollo de nuevas aplicaciones web o de escritorio orientadas a la conservación del patrimonio, tanto en la Comunidad de Madrid como en otras regiones interesadas.

Sin embargo, esta solución no es escalable a largo plazo debido a que toda la ejecución se lleva a cabo en un entorno local, es decir, la base de datos y las aplicaciones desarrolladas se encuentran en un mismo dispositivo, el cual almacena toda la información. Este planteamiento presenta limitaciones claras; por ejemplo, supongamos una situación en la que al menos dos usuarios estén interactuando simultáneamente con la base de datos, añadiendo o modificando registros.

En este escenario, surgen preguntas importantes: ¿cómo podrían asegurarse los usuarios de que un registro que intentan agregar no ha sido ya introducido por otro usuario previamente en su propio equipo? ¿Cómo evitar la duplicidad en los datos o cómo facilitar la compartición eficiente de información?

Ante estos desafíos, la solución más adecuada consiste en desacoplar la base de datos de los dispositivos personales y alojarla en un servidor externo accesible a través de internet. Esta alternativa permite mantener la base de datos constantemente actualizada y accesible para múltiples usuarios de manera simultánea.

Sin embargo, exponer la base de datos en internet también plantea nuevos retos, especialmente relacionados con la ciberseguridad. Afortunadamente, dado que en este caso se trata de información pública, los riesgos asociados podrían gestionarse eficazmente mediante medidas básicas de seguridad y protección adecuadas.

A nivel personal, la realización de este proyecto ha significado una enriquecedora experiencia académica y profesional. Me ha permitido aplicar y ampliar significativamente mis conocimientos técnicos en ingeniería del software e inteligencia artificial, además de brindarme una valiosa oportunidad para contribuir activamente a un campo relevante como es la conservación patrimonial.

5.1 Objetivos cumplidos

A lo largo del desarrollo de este proyecto se han alcanzado satisfactoriamente los objetivos propuestos inicialmente, destacando especialmente los siguientes logros específicos:

- Se ha llevado a cabo con éxito el proceso de ingeniería inversa sobre la base de datos proporcionada por la Comunidad de Madrid, obteniendo una estructura clara y detallada que incluye todas las tablas, columnas, tipos de datos y relaciones existentes. Esta estructura fue documentada exhaustivamente mediante archivos en formato Markdown, facilitando significativamente la comprensión y el futuro mantenimiento del sistema.
- Se ha desarrollado un ORM personalizado completamente funcional que permite realizar operaciones CRUD (Create, Read, Update, Delete) sobre la base de datos Microsoft Access. Este ORM ha logrado abstraer eficazmente la complejidad inherente al acceso directo a la base de datos, simplificando la interacción y mejorando considerablemente la eficiencia y seguridad del proceso.

- Se ha logrado implementar la capacidad de importar y exportar datos desde y hacia formatos estándar como .csv y .json mediante el ORM desarrollado. Esta característica es crucial para la integración y análisis de los datos con otras herramientas externas, facilitando la flexibilidad en el manejo de información patrimonial.
- Se ha desarrollado e integrado exitosamente un modelo de inteligencia artificial basado en la plataforma OpenAI, adaptado mediante técnicas de ajuste fino (fine tuning). Este modelo es capaz de automatizar parcialmente la recolección y estructuración de información sobre centrales telefónicas, generando resultados en formato .csv compatibles con el ORM, lo que optimiza significativamente el proceso tradicional de catalogación manual.
- Se ha establecido un entorno de documentación técnica clara y accesible utilizando la herramienta Obsidian, garantizando la trazabilidad, transparencia y facilidad en la gestión de información técnica y decisiones de desarrollo del proyecto.

Estos objetivos cumplidos representan un avance significativo hacia la automatización y optimización del proceso de gestión y catalogación del patrimonio arquitectónico industrial de las centrales telefónicas en la Comunidad de Madrid, sentando además las bases para futuras ampliaciones y mejoras tecnológicas del sistema desarrollado.

5.2 Futuros desarrollos

El presente Trabajo de Fin de Grado representa el inicio de un proyecto de mayor alcance en colaboración con la Comunidad de Madrid, estableciendo una base tecnológica sólida sobre la cual se podrán desarrollar futuras aplicaciones tanto web como de escritorio. Estas aplicaciones estarán destinadas principalmente a optimizar el mantenimiento, gestión y conservación de bienes culturales, en especial las centrales telefónicas catalogadas.

Entre las posibles mejoras y ampliaciones planteados a futuro se encuentran:

- Extender la funcionalidad del ORM desarrollado para integrarse con múltiples plataformas y entornos tecnológicos, permitiendo así una gestión más flexible, ágil y eficiente del patrimonio cultural desde diversos dispositivos y sistemas operativos.
- Diseñar e implementar una interfaz web o aplicación de escritorio intuitiva y amigable para usuarios no técnicos, que permita consultar, actualizar y gestionar eficazmente la información almacenada sobre los bienes culturales. Esto facilitará considerablemente las tareas administrativas, reduciendo significativamente el esfuerzo manual y el tiempo requerido en comparación con métodos tradicionales.
- Mejorar el modelo de inteligencia artificial implementado, ampliando su capacidad para procesar una mayor variedad de documentos históricos, imágenes y otros recursos documentales. Este proceso implica una fase adicional de entrenamiento y ajuste del modelo para aumentar su precisión y aplicabilidad, logrando así una automatización más efectiva y ahorrando importantes recursos humanos y temporales.
- Una vez desarrollada una API capaz de interactuar con la base de datos, es posible entrenar un modelo de inteligencia artificial para que realice llamadas automatizadas a las distintas rutas disponibles en dicha API. Esto significa que podemos utilizar la inteligencia artificial para recopilar datos y, posteriormente, almacenarlos de manera automática en la base de datos. Este enfoque resulta especialmente útil, ya que elimina la necesidad de interacción directa del usuario con el ORM, simplificando así el proceso y aumentando la eficiencia.
- Utilizar la infraestructura tecnológica desarrollada como base para futuros proyectos académicos, particularmente para los trabajos de fin de grado de otros estudiantes, como es el caso de los alumnos Alejandro Tineo y Alejandro López. Estos estudiantes podrían aprovechar esta plataforma tecnológica para extender sus investigaciones, desarrollar nuevas funcionalidades y aportar mejoras

adicionales al sistema, fortaleciendo así la continuidad y evolución del proyecto.

La importancia de continuar con el desarrollo de este proyecto radica en su potencial para transformar profundamente el ámbito de la conservación patrimonial. Al automatizar procesos tradicionalmente lentos y laboriosos, esta iniciativa podría ahorrar enormes cantidades de tiempo y esfuerzo humano, incrementando sustancialmente la eficacia y calidad de la gestión cultural.

6. REFERENCIAS

6.1 Bibliografía.

Fernández, J., García, P., & Ramírez, L. (2023). Aplicaciones de inteligencia artificial en la digitalización del patrimonio cultural. *Journal of Digital Heritage*, 18(3), 45-61.

Fowler, M. (2003). *Patterns of Enterprise Application Architecture*. Addison-Wesley.

García Algarra, J. (2012). De Gran Vía al Distrito C: el patrimonio arquitectónico de Telefónica

García, M., & López, A. (2020). Patrimonio arquitectónico industrial en España: Historia y conservación. *Revista de Patrimonio Cultural*, 25(2), 89-105.

Gómez, R., & Ramírez, S. (2022). Procesamiento de lenguaje natural en la documentación de archivos históricos. *Digital Humanities Review*, 30(4), 112-130.

Ley 16/1985, de 25 de junio, del Patrimonio Histórico Español. (1985). Boletín Oficial del Estado. <https://www.boe.es/buscar/act.php?id=BOE-A-1985-12534>

López-Menchero Bendicho, V. M., & Herráez, J. (2019). Digitalización del patrimonio cultural: Modelos, tecnologías y aplicaciones. Universidad de Castilla-La Mancha. https://doi.org/10.18239/jor_03.2019.08

Martínez, C., Pérez, R., & Sánchez, D. (2021). Transformaciones urbanas y desaparición del patrimonio industrial en Madrid. *Urban Heritage Journal*, 15(1), 77-95.

Mendoza, J., & Pérez, T. (2021). Machine learning y PLN en la clasificación de documentos históricos. *Artificial Intelligence Journal*, 19(2), 54-72.

Microsoft. (2022). Microsoft Access Documentation. Microsoft.

Rodríguez, F., & Sánchez, L. (2022). Interoperabilidad de bases de datos patrimoniales: Problemas y soluciones. *Heritage Data Science*, 14(1), 99-120.

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide: The definitive guide to Scrum: The rules of the game*.

6.2 Webgrafía.

Acento. (s. f.). La inteligencia artificial y su importancia en el patrimonio y las industrias culturales: retos y desafíos. Recuperado de <https://acento.com.do/cultura/la-inteligencia-artificial-y-su-importancia-en-el-patrimonio-cultural-y-las-industrias-culturales-retos-y-desafios-9426133.html>

Ambler, S. W. (2012). Mapping objects to relational databases: O/R mapping in detail. Agile Data. Recuperado de <https://www.agiledata.org/essays/mappingObjects.html>

Comunidad de Madrid. (2024, 5 de mayo). La Comunidad de Madrid declara Bien de Interés Cultural la Estación de Comunicaciones por Satélite de Buitrago del Lozoya. Recuperado de <https://www.comunidad.madrid/notas-prensa/2024/05/03/comunidad-madrid-declara-bien-interes-cultural-estacion-comunicaciones-satelite-buitrago-lozoya>

de la Cruz, L. (2022, 26 de noviembre). Demolida la estación telefónica más antigua de Madrid en el barrio de Salamanca. elDiario.es. Recuperado de https://www.eldiario.es/madrid/somos/demolida-estacion-telefonica-antigua-madrid-barrio-salamanca_1_9745287.html

Dirección General de Patrimonio Cultural. (2022). ETNOCAM: Sistema de catalogación del patrimonio en la Comunidad de Madrid. Gobierno de la Comunidad de Madrid.

Golke, M. (2024). pyodbc 5.0.1 documentation. PyODBC. Recuperado de <https://github.com/mkleehammer/pyodbc>

Jornal da Unicamp. (2024, 23 de septiembre). La Inteligencia Artificial puede ayudar a preservar el patrimonio cultural. Recuperado de <https://jornal.unicamp.br/es/noticias/2024/09/23/inteligencia-artificial-pode-ajudar-na-preservacao-do-patrimonio-cultural/>

Microsoft. (s. f.). Data Manipulation Language (DML) – Access Desktop Database Reference. Recuperado de <https://learn.microsoft.com/es-es/office/client-developer/access/desktop-database-reference/data-manipulation-language>

PatrimoniIndustrial.cat. (s. f.). Catálogo digital georreferenciado sobre el patrimonio industrial catalán. Recuperado de <http://www.patrimoniindustrial.cat/es/>

Revista ASRI. (s. f.). Tecnologías para la conservación del arte y patrimonio cultural: algoritmos de inteligencia artificial en aplicaciones. Recuperado de <https://revistaasri.com/article/view/6741/7327>

VidelCloud. (2017, 5 de marzo). Buenas prácticas para el diseño de base de datos. Recuperado de <https://videlcloud.wordpress.com/2017/03/05/buenas-practicas-para-el-diseno-de-base-de-datos/>

Álvarez, M. (s. f.). PatrimoniIndustrial.cat: un catálogo digital georreferenciado sobre el patrimonio industrial catalán. BiD: textos universitaris de biblioteconomia i documentació. Recuperado de <https://bid.ub.edu/es/43/alvarez.htm>

7. ANEXOS

GitHub con el código: <https://github.com/adriantoral/tfg>

Glosario de siglas y acrónimos:

- AI / IA – Inteligencia Artificial (Artificial Intelligence).
- API – Interfaz de Programación de Aplicaciones (Application Programming Interface).
- CSV – Valores Separados por Comas (Comma-Separated Values).
- DML – Lenguaje de Manipulación de Datos (Data Manipulation Language).
- ETNOCAM – Base de datos etnográfica de la Comunidad de Madrid.
- GUI – Interfaz Gráfica de Usuario (Graphical User Interface).
- JSON – Notación de Objetos de JavaScript (JavaScript Object Notation).
- NLP / PLN – Procesamiento de Lenguaje Natural (Natural Language Processing).
- ORM – Mapeo Objeto-Relacional (Object-Relational Mapping).
- PDF – Formato de Documento Portátil (Portable Document Format).
- PLN – Procesamiento de Lenguaje Natural.
- SQL – Lenguaje de Consulta Estructurado (Structured Query Language).
- UI – Interfaz de Usuario (User Interface).
- URL – Localizador Uniforme de Recursos (Uniform Resource Locator).