TRABAJOS FIN DE GRADO

Universidad de Jaén

Escuela Politécnica Superior de Jaén

UJA-GPT Un chatbot con generación aumentada por recuperación para la comunidad universitaria

Autor: Pablo Calahorro Jiménez

Grado: Ingeniería informática

Director: Arturo Montejo Ráez
Departamento del director: Informática

Fecha: 01/07/2024

CREA

ÍNDICE

ÍNDICE TABLAS Y FIGURAS

Agradecimientos

Quisiera expresar mi más profundo agradecimiento a todas las personas que han contribuido a la realización de este Trabajo de Fin de Grado.

En primer lugar, agradezco a mi tutor, Arturo, por su constante apoyo, orientación y valiosos consejos a lo largo de este proceso. Su experiencia y dedicación han sido fundamentales para la culminación de este trabajo, además de agradecer el ofrecimiento de un tema que me hizo recuperar la curiosidad y el interés que creía perdidos.

También quisiera gradecer la predisposición de los miembros de los distintos equipos de los servicios universitarios que han participado en el diseño, a Enrique, a Nuria, a Juan Carlos, a José María… la paciencia que han demostrado en las reuniones. Espero que en el futuro este trabajo repercuta en facilitarles el trabajo.

Agradezco también a todos mis profesores y compañeros de la escuela politécnica superior de Jaén, quienes han compartido conmigo sus conocimientos y experiencias a lo largo de estos años de estudio.

A mi pareja Paloma, por su comprensión y paciencia durante los momentos de estrés y dedicación que este trabajo ha requerido, sobre todo durante la enfermedad de nuestra perrita. Su apoyo emocional ha sido determinante para mantenerme enfocado y motivado.

Finalmente, quiero dedicar este trabajo a mi familia. A mis padres, por su amor incondicional, apoyo constante y sacrificios para brindarme las oportunidades necesarias para mi formación académica. A mi hermana Elena, por su aliento, palabras de ánimo y regañinas.

Sin todo el apoyo, este proyecto no hubiera sido posible.

1. Especificación del trabajo

1.1 Introducción

En esta introducción se va a presentar la idea que se pretende diseñar e implementar, pero sin demasiado detalle, puesto que, en puntos posteriores de este mismo capítulo, y en capítulos posteriores, se irán profundizando tanto el diseño como la implementación.

Como idea general, se pretende es **potenciar** la capacidad de la inteligencia artificial para emular el lenguaje humano, con **información administrativa** de la universidad para elaborar de forma automática respuestas a las preguntas de los miembros de la comunidad universitaria, principalmente estudiantes.

1.1.1 Contexto administrativo

En las universidades, o en cualquier otra organización de similares dimensiones, surgen numerosos retos acerca de cómo atender de forma eficiente y efectiva las diversas y numerosas demandas de usuarios del ente público, o análogamente, clientes del ente privado.

Durante el desarrollo del presente proyecto, se va a poner el foco en los **usuarios** de distintos servicios como ayuda al estudiante, relaciones internacionales, o el servicio de gestión académica de la Universidad de Jaén, así como en **el personal que trabaja** en dichos servicios.

Con el objetivo de conseguir un poco de perspectiva sobre **para qué** se desarrolla un sistema de apoyo a los servicios anteriores, es importante enumerar algunas áreas de gestión universitaria de las que se encargan:

- Acceso a la universidad
- Matrícula de nuevo estudiantado o en continuidad de estudios
- Modificación de matrícula
- Cambios de grupos
- Recogida y expedición de títulos
- Reconocimiento de estudios previos
- Movilidad internacional entrante – saliente
- Orientación laboral

Para la consecución de los objetivos del proyecto, es de una importancia capital, como desarrollador, tener una perspectiva clara y completa del funcionamiento administrativo de la universidad, lo que supondrá un trabajo constante con los trabajadores de los servicios para conseguir los objetivos

definidos en el presente trabajo (apartado 1.3). En realidad, aunque la relación con los trabajadores es puramente académica, serían los clientes si esto se tratase de una relación comercial.

1.1.2 Contexto académico

Desde que aparece chat-GPT[1], como gran hito, la relación de la población universitaria con la inteligencia artificial ha sufrido un cambio vertiginoso, hasta el punto de estar completamente integrada en la cotidianidad por una buena parte del estudiantado y profesorado, aunque no tanto por parte de las áreas de gestión.

Por poner un ejemplo de lo anterior, se aprecia una diferencia notable a la hora de programar entre:

1. Documentarse a través de los tradicionales stack overflow[2], w3schools[3] y similares.
2. Preguntar a una inteligencia artificial cómo se realiza una tarea concreta en un lenguaje, o cómo se interpreta un determinado error, lo que ahorra entre dos minutos y hasta varias horas según la complejidad de la tarea sobre la que se necesite ayuda.

   Esto hace que se plantee la siguiente cuestión, si tan bien ha venido para:

- Como alumno: hacer un trabajo, o ayudar con la realización de un ejercicio.
- Como profesor: prepararse una clase o realizar un mapa de contenidos.

¿Por qué no se utilizan este tipo de sistemas para fines burocráticos o administrativos?

Existen numerosas respuestas, pero en las pruebas preliminares nos damos cuenta de algunas cuestiones llamativas:

1. La inteligencia artificial no controla si las fuentes de información que se consultan en el proceso de recuperación de información son válidas o no. La prioridad de un sistema de este tipo, es, normalmente, contestar a toda costa.
2. Cada contexto administrativo o burocrático marca unas reglas que van más allá del lenguaje.
3. Esto hace necesario un consistente trabajo humano para preparar y fiscalizar a los sistemas de recuperación de información

[1] https://chatgpt.com/
[2] https://stackoverflow.com/
[3] https://www.w3schools.com/

Un par de ejemplos significativos para ilustrar esto:

1. A nivel de lenguaje convalidar y homologar son prácticamente sinónimos, a nivel burocrático no son sinónimos en absoluto.
2. Normalmente los sistemas de este tipo no son capaces de filtrar qué información es la relevante en su contexto de uso, ese trabajo aún es humano.

1.2 Estructura del documento

El presente documento se organiza en cuatro grandes secciones.

1. Especificación.

   En esta, se da contexto, se explican objetivos, planificación, metodología y tecnologías seleccionadas. Además, se realiza un análisis de riesgos. También se analiza la situación actual y se define el alcance del proyecto.

2. Diseño

   En esta sección se detalla el proceso de recuperación implementado, la estructura del proyecto, los datos generados por el funcionamiento del sistema, y los distintos flujos de información y retroalimentación cliente-usuario.

3. Funcionamiento y Experimentación

   En esta sección se explican distintas pruebas de funcionamiento y se detalla el post procesamiento de los datos generados por el sistema durante su funcionamiento.

4. Conclusiones

En esta sección se elaboran unas conclusiones en torno a tres grandes ejes, el trabajo con el cliente y la satisfacción de los usuarios, el futuro de los bots expertos, escalabilidad del sistema. Además se concluye con una comparación UJAGPT vs ADA y una reflexión personal.

1.3 Objetivos del trabajo

Objetivos generales:

1. Reducir la carga de trabajo en concepto de atención a usuarios de los equipos que conforman los servicios mencionados en los puntos anteriores.
2. Generar respuestas satisfactorias e inmediatas ante las preguntas y dudas de los usuarios.
3. Dar a cada usuario una atención lo más personalizada y específica posible en el contexto de la gestión académica.

Objetivos específicos:

1. Definir una base de conocimiento completa y lo más fácil de gestionar posible.
2. Obtener respuestas claras y completas a cada cuestión planteada por parte de los usuarios.
3. Adjuntar a cada respuesta elaborada por el sistema, las fuentes consultadas de una forma ordenada y accesible para el usuario.
4. Definir un conjunto de métricas para validar el sistema
5. Validar los resultados a través de los trabajadores mediante un sistema y a través de los usuarios mediante otro sistema sensiblemente distinto, dado que los criterios de un trabajador y de un usuario difieren notablemente, también difieren las métricas utilizadas.
6. Utilizar las validaciones para determinar los parámetros de funcionamiento más óptimos.
7. Identificar resultados y comportamientos no deseados a través de las validaciones anteriores, así como sus causas y dar solución a cada incidencia identificada.

1.4 Antecedentes y estado del arte

A continuación, se van a listar y definir brevemente los conceptos y elementos clave en relación con el desarrollo del proyecto.

1.4.1 RAG, definición y esquema básico

Técnica que combina la generación de texto mediante modelos de lenguaje con la recuperación de información relevante de una base de datos o corpus de documentos [5]. Esta técnica tiene el objetivo de mejorar la precisión y relevancia de las respuestas generadas por los modelos de lenguaje.

1. Recuperación de Información (Retrieval):
    - Base de Datos: RAG se basa en un conjunto predefinido de documentos o una base de datos de la cual puede recuperar información. En el presente proyecto a partir de un conjunto de documentos descargados de un conjunto acotado de direcciones url generaremos una base de datos vectorial.
    - Mecanismo de Recuperación: Utiliza un mecanismo de recuperación de información para identificar y extraer documentos o fragmentos relevantes de la base de datos en respuesta a una consulta o pregunta. En nuestro caso se realizará a través de un índice que indexa fragmentos de los documentos antes mencionados
2. Generación de Texto (Generation):
    - Modelo de Lenguaje: Emplea un modelo de lenguaje avanzado que puede generar texto coherente y contextual a partir de un prompt dado.
    - Fusión de Información: El modelo de generación integra la información recuperada con su capacidad de generación de texto para producir respuestas más precisas y contextualmente adecuadas.

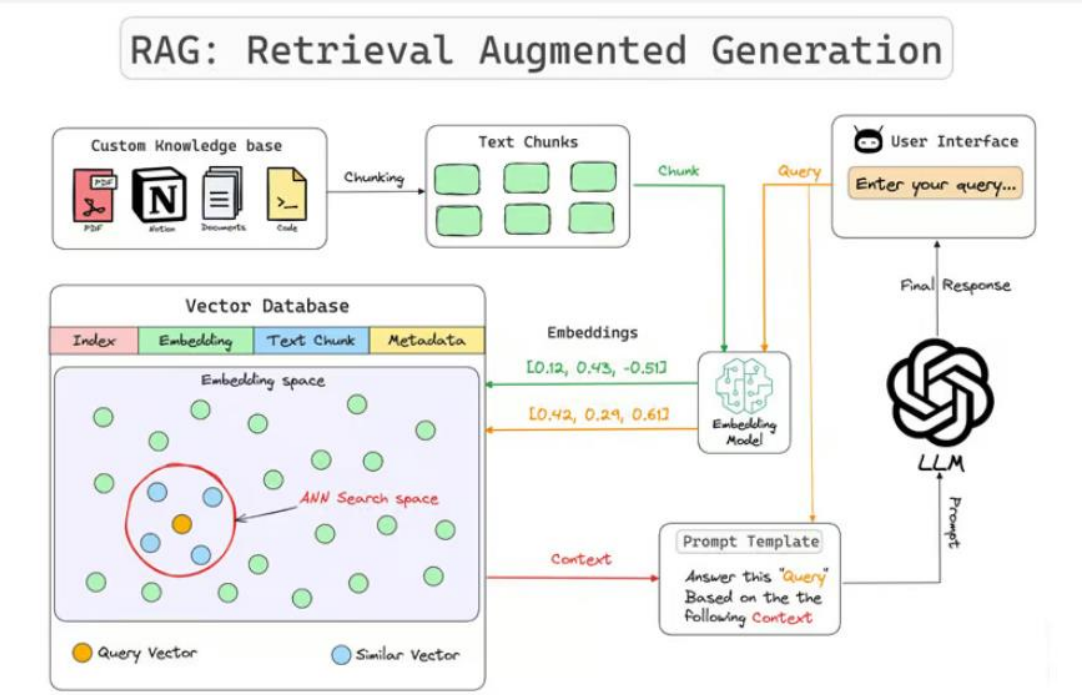


Figura 1. esquema general de un sistema RAG[4]

1.4.2 Componentes de un sistema RAG

A continuación, se enumeran y describen brevemente, los componentes de un sistema RAG [5]:

1. **Base de datos vectorial**. Se almacenan los datos, en el presente caso texto, pero también imágenes o audio, a través de representaciones numéricas, lo que permite optimizar y simplificar los procesos de recuperación de información. **Ofrecen la posibilidad de realizar búsquedas por similitud**.
2. **Embeddings**. Se ha comentado a la ligera anteriormente, aquello de "representaciones numéricas de los datos". Se podría decir que aquí encontramos el núcleo de todo el proceso que se está tratando: **conseguir capturar numéricamente las relaciones semánticas que hay en / entre los datos**. Se podría elaborar todo un proyecto analizando este concepto.
3. **Base de conocimiento**. Se trata de la colección de documentos a partir de la cual se construye la base de datos vectorial.
4. **LLM**. Modelo de lenguaje de gran tamaño que ha sido entrenado con gigantescas cantidades de datos textuales para comprender y generar texto de manera coherente y contextualmente relevante.

[4] https://www.consultor365.com/ia/explicacion-rag/

5. **Prompt Template**. Define el contexto de la interacción, esto ayuda al sistema a generar respuestas adecuadas según el fin para el que esté ideado.
6. **User Interface**. Cualquier tecnología que permita a un usuario no experto trasladar una consulta al sistema, para posteriormente recibir una respuesta y las fuentes de las que se ha extraído la misma.

1.4.3 Loro estocástico

El término "loro estocástico" [1] se usa de manera normalmente crítica para describir modelos de lenguaje avanzados como GPT-3 o GPT-4. La idea es que estos modelos, a pesar de su capacidad para generar texto coherente y a menudo convincente, no entienden realmente el contenido que producen. En cambio, funcionan mediante la predicción estadística de palabras y frases basadas en los datos con los que fueron entrenados, de manera similar a cómo un loro repite palabras sin comprenderlas.

- **Estocástico** se refiere a los procesos que son aleatorios o probabilísticos. En este contexto, los modelos de lenguaje generan texto basado en probabilidades calculadas a partir de vastas cantidades de datos de entrenamiento.
- **Loro** sugiere repetición sin comprensión. Los modelos de lenguaje no tienen una comprensión profunda del significado, sino que generan texto que estadísticamente tiene sentido dado el contexto de entrada.

Por lo tanto, un "loro estocástico" en IA es un modelo que produce texto coherente, pero carece de entendimiento real del mismo.

Los modelos de lenguaje sin contexto ni una correcta validación de la interacción, se comportan como loros basados en principios probabilísticos para emular la relación del ser humano con el lenguaje.

1.4.4 Dimensión de sistemas RAG

A continuación, se exponen una serie de circunstancias relacionadas con el consumo de los sistemas RAG.

RAG de menor dimensión

La puesta en marcha y mantenimiento de los sistemas RAG de altas capacidades, conlleva un consumo significativo de recursos, tanto en términos de energía como de materiales, particularmente tierras raras. A continuación, se presentan algunas ideas a modo de justificación de por qué parece prometedor aplicar una técnica **divide y vencerás** a gran escala con los sistemas RAG y por qué no parece recomendable mantener sistemas con índices excesivamente grandes.

Consumo de Energía de Centros de Datos

Según un informe de la Agencia Internacional de Energía de 2024 [4], los centros de datos a nivel mundial consumieron aproximadamente 200 teravatios-hora (TWh) de electricidad en 2018, lo que representa alrededor del 1% del consumo mundial de electricidad. Este consumo ha ido en aumento debido a la creciente demanda de procesamiento de datos, impulsada en gran parte por el uso de tecnologías avanzadas como los sistemas RAG.

Proporción de Energía Fósil

La fuente de energía para los centros de datos varía según la región. A nivel global, una parte significativa de la electricidad todavía proviene de combustibles fósiles, aunque esta proporción está disminuyendo con el aumento de las fuentes renovables. Según datos de la IEA, aproximadamente el 60% de la electricidad a nivel mundial proviene de combustibles fósiles (carbón, gas natural y petróleo).

1.4.5 Resumen

Más allá de consideraciones mediáticas, políticas o ideológicas, si no se mantiene un exhaustivo control sobre la información indexada por un sistema de IA generativa, se corre un serio riesgo, por ejemplo, de recuperar plazos y procedimientos obsoletos o erróneos.

Si no se mantienen correctamente **actualizadas y supervisadas las fuentes** de información de las que se recuperan las respuestas a las cuestiones de los usuarios en cada ámbito concreto, se incurriría en el riesgo mencionado anteriormente, vigilar fuentes de datos excesivamente grandes es muy costoso. Esta es una razón de peso, junto la cuestión material y energética antes mencionada (1.4.4), para aplicar el clásico **"divide y vencerás"**. Si se definen correctamente dominios de conocimiento pequeños con fuentes de datos manejables:

- **La recuperación por consulta es más eficiente**
- **La recuperación es más eficaz, se falla menos y se indexa menos "basura"**

1.5 Descripción de la situación de partida

1.5.1 Descripción del entorno actual en la UJA

Se pretende desarrollar y desplegar un sistema basado en RAG que asista al servicio de gestión académica de la universidad de Jaén, del que dependen directamente los procesos más relevantes en el desarrollo del normal funcionamiento de la organización como es acceso a la universidad a través de distintas vías, echar la matrícula, participación en programas de movilidad o la expedición de títulos.

El **impacto** se podrá apreciar como una importante reducción en la cantidad de atenciones, telefónicas o presenciales, que realiza diariamente el servicio, minimizándolas gracias a un sistema basado en RAG con interfaz, tipo "bot experto" que atenderá a los usuarios finales, es decir los estudiantes, a través de una interacción similar a chat-GPT y en relación a cualquier cuestión administrativa y académica de la organización.

Además, se realizará una **comparación exhaustiva con el actual sistema de atención al estudiante basado en inteligencia artificial ADA**[5], con el fin de mejorar la calidad de la atención que reciben los estudiantes.

[5] https://www.ujaen.es/estudios/acceso-y-matricula/ada

1.5.2 Deficiencias y carencias identificadas

En este apartado se van a mostrar algunos ejemplos de interacción con el actual sistema de atención al estudiante como motivación principal del trabajo.

En las siguientes imágenes observamos que

1. Sólo contesta en castellano
2. Advierte que la información no es vinculante
3. Es más bien un asistente a la navegación por la web de la universidad que un sistema de IA generativa.

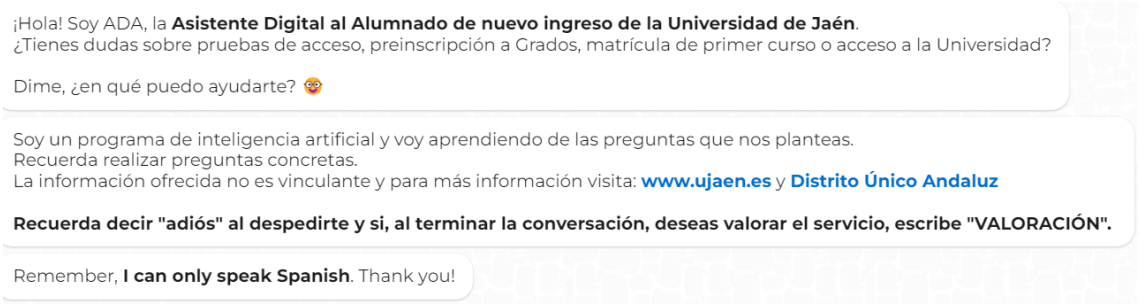


Imagen 1. ADA presentación

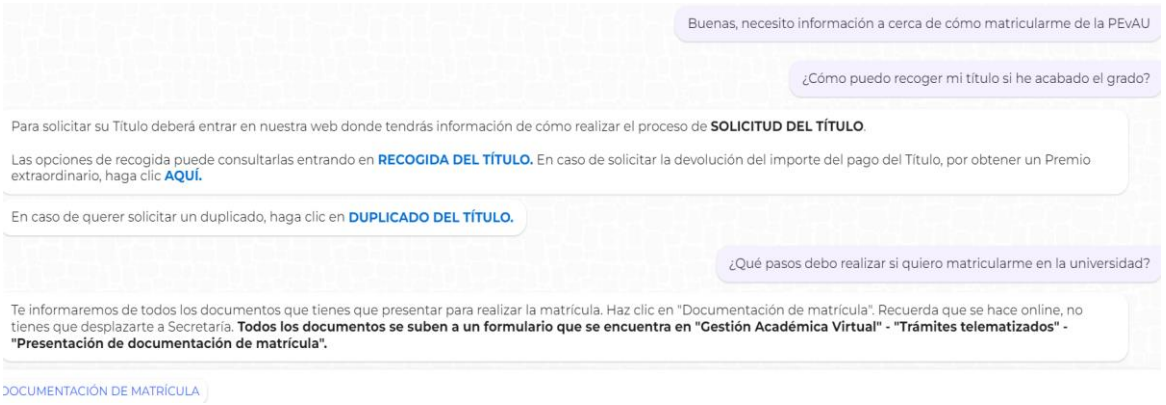


Imagen 2. ADA antes de modificaciones

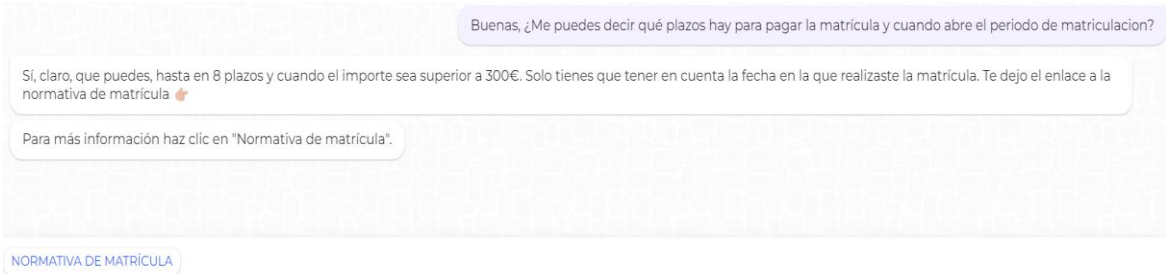


Imagen 3. ADA tras modificaciones

En las imágenes anteriores, se observa una deficiencia, le pregunto que “si puede decirme”, y en la respuesta me dice que “puedo matricularme”, la información es correcta pero el enlace que adjunta no contiene la información concreta. **Demasiado genérico**.

Entre las dos imágenes anteriores, la 2 y la 3, ha transcurrido bastante tiempo. Se aprecia que se ha implementado una mejora en el sistema ADA de manera que ahora si se está utilizando un modelo de lenguaje mejor entrenado que el anterior pero **no se aprecia un mecanismo de recuperación demasiado efectivo**.

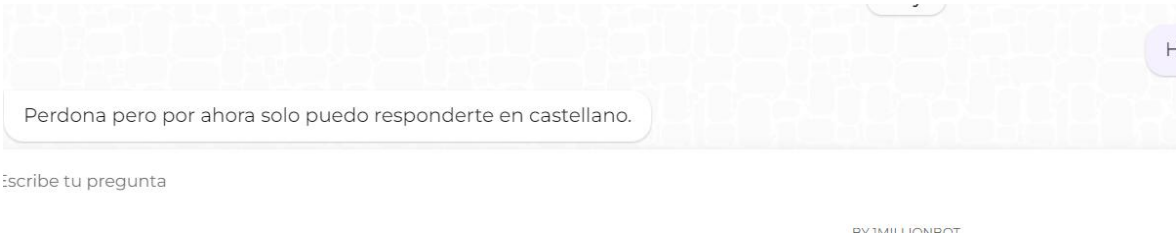


Imagen 4. ADA idiomas

Además, en la imagen 4 queda evidencia del **problema del actual sistema ADA con los idiomas**.

1.6 Requisitos iniciales

A continuación, se enumeran los requisitos básicos del sistema:

1. El sistema debe permitir a los usuarios realizar consultas a través de un chat, debe llegar a soportar cargas de en torno a 30 mil usuarios, peor caso estimado por la cantidad de estudiantes matriculados y alumnos que realizan pruebas de acceso.
2. Debe responder a consultas sobre cualquier cuestión referente a asuntos administrativos de los estudiantes de la UJA.
3. El sistema debe contestar en el idioma en el que se realiza la consulta, en caso de no ser así, debe permitir la traducción automática a demanda del usuario
4. El sistema debería permitir al equipo del servicio de gestión académica manipular y actualizar las fuentes de información del sistema RAG sin intervención de ningún desarrollador o técnico informático. Es decir, a través de una interfaz de retroalimentación
5. El sistema debe permitir la validación cruzada entre parámetros y métricas extraídas de la interacción con los usuarios
6. El punto anterior debería implementarse a través de un proceso de aprendizaje automático
7. El sistema debe cumplir todas las normas y regulaciones referentes a protección de datos
8. El sistema debe funcionar, aunque reciba mensajes con emoticonos, caracteres orientales o cirílicos.

1.7 Alcance

Descripción

Desarrollar un sistema que permita asesorar a los alumnos de la universidad sobre todo tipo de trámites administrativos al mismo tiempo que descarga de trabajo al equipo administrativo de la organización

Entregables

- Sistema basado en RAG con interfaz de chat para usuarios finales
- Manual de despliegue
- Manual de usuario
- Documentación sobre experimentos realizados

Características

- Mantendrá el contexto de la conversación reciente para optimizar la recuperación de información
- Contestará en el idioma en el que se realice la consulta
- Permitirá mantener un log de forma anónima con el fin de dejar registro, por ejemplo, de qué información se recupera con más frecuencia
- Permitirá, de forma paralela, generar un archivo JSON que va guardando datos sobre las interacciones respecto de un conjunto de consultas definidas por los trabajadores del servicio

Exclusiones

- No se implementará la interfaz gráfica necesaria para que el equipo de administración pueda manipular las fuentes de información del sistema RAG.
- No manejará consultas con emoticonos, o caracteres orientales.
- No se hará un subsistema de parametrización automática.

Restricciones

- El proyecto debe completarse en 4 meses
- El presupuesto máximo es de 15.000 euros

Suposiciones

- Los usuarios deben tener acceso a internet
- La organización debe disponer de servidores propios en los que instalar el sistema

Stakeholders

- Equipo de desarrollo. Pablo Calahorro Jiménez
- Gerente del proyecto. Pablo Calahorro Jiménez
- Usuarios finales. estudiantes de la universidad de Jaén

- Cliente. Universidad de Jaén. Jefes de servicio
    - Servicio de relaciones internacionales, Emilio Ayala
    - Servicio de gestión académica, Enrique Jerez
    - Servicio de ayuda al estudiante, Fernando Valverde
    - Servicio de deportes, Álvaro Trujillo

1.8 Alternativas y viabilidad

Se van a enumerar y describir brevemente las alternativas entre las que se han seleccionado las tecnologías y herramientas para la implementación

1.8.1 Tecnologías de indexación

Imagen 5. Alternativas acceso datos no estructurados

- **Llama-index[6] versión 0.10.46**

  Llama-Index es una tecnología que proporciona una infraestructura ligera para la construcción y el análisis de modelos de lenguaje natural [5]. Está diseñada para ser eficiente en términos de recursos y puede integrarse fácilmente en aplicaciones que requieren análisis de texto o generación de lenguaje natural

- Elasticsearch[7] versión 8.14.1

  Elasticsearch es un motor de búsqueda y análisis de texto de código abierto, basado en el motor de búsqueda Apache Lucene. Permite almacenar, buscar y analizar grandes volúmenes de datos en tiempo real. Es conocido por su capacidad de búsqueda distribuida, escalabilidad y flexibilidad en el manejo de datos estructurados y no estructurados

- Weaviate[8] versión 1.24.19

  Weaviate es una base de datos de gráficos vectoriales de código abierto que permite el almacenamiento, búsqueda y administración de datos semánticos. Utiliza vectores para representar datos y ofrece capacidades avanzadas de búsqueda y análisis semántico, integrándose con modelos de machine learning para mejorar la precisión y relevancia de las consultas

- Annoy versión[9] 1.17.3

  Annoy es una biblioteca de código abierto desarrollada por Spotify para la búsqueda de vecinos más cercanos aproximados en grandes conjuntos de datos de alta dimensión. Es eficiente en términos de memoria y rendimiento, y es ampliamente utilizada en aplicaciones que requieren búsquedas rápidas y

[6] https://www.llamaindex.ai/
[7] https://www.elastic.co/es/elasticsearch
[8] https://weaviate.io/
[9] https://pypi.org/project/annoy/

aproximadas, como sistemas de recomendación y recuperación de información basada en vectores.

De estas cinco grandes candidatas, nos hemos quedado con la que ofrece una integración más sencilla con el resto de aspectos del sistema, **llama-index**. Elegir llama-index nos ahorra, además, la elección del modelo de lenguaje, dado que trae GPT 4.0.

1.8.2 Almacenamiento

Imagen 6. Alternativas base de datos

- MongoDB[10] versión 7.0.55
- ChromaDB[11] versión 0.5.3
- Pinecone[12] versión 4.1.0

Se van a enumerar las características más determinantes de cara a elegir.

- Pinecone y ChromaDB están diseñados específicamente para búsquedas y **almacenamiento de vectores**, orientados a aplicaciones de aprendizaje automático, recuperación de información, inteligencia artificial en general.
- MongoDB es una base de datos generalista NoSQL orientada a documentos, adecuada para una amplia gama de aplicaciones de datos no estructurados.
- Pinecone es un servicio gestionado, eliminando la necesidad de gestionar la infraestructura.
- ChromaDB ofrece flexibilidad de despliegue.
- MongoDB requiere gestión de infraestructura, aunque existen servicios gestionados como MongoDB Atlas.

En resumen, chromaDB y Pinecone son las más convenientes dado que se va a trabajar con vectores (llama-index) en un sistema de IA generativa.

Basta ver las versiones, 0.5 y 4.1 para adivinar que Pinecone sería una opción más asequible en términos de esfuerzo, se encontraría más material y mejor documentado, pero chromadb [7], aunque aún es joven, promete mucho y ofrece un RAG para su documentación que, en estos primeros compases del trabajo, está siendo de gran utilidad. **Por lo tanto, se ha decidido utilizar como librería de almacenamiento, chromaDB**.

[10] https://www.mongodb.com/
[11] https://docs.trychroma.com/
[12] https://www.pinecone.io/

1.8.3 Interfaz de usuario

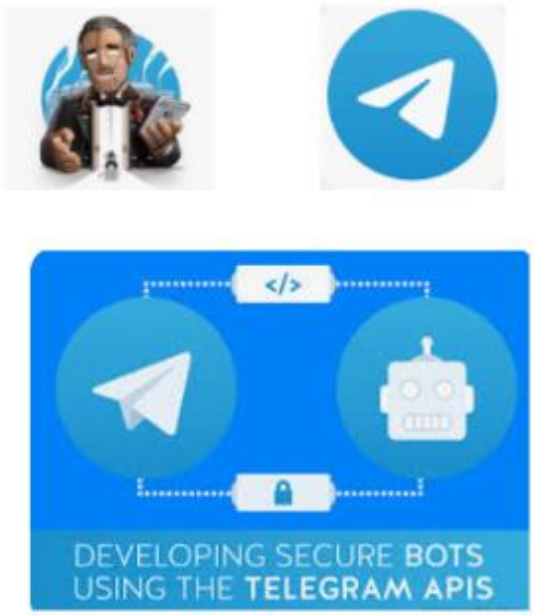


Imagen 7. Telegram como interfaz[13]

Telebot[14] **versión 4.2**, para interactuar con la API de telegram[15]. Telebot es una implementación tanto síncrona como asíncrona de telegram Bot API[16] versión 7.5.

Dado que se dispone de poco tiempo y recursos, se recorta aquí, gracias a la elección de telebot, **obteniendo unos resultados muy óptimos a cambio de muy poco esfuerzo para lanzar un sistema completamente funcional**.

Aunque se da una **gran desventaja** con respecto a otros sistemas equivalentes, un mismo usuario no puede abrir varios hilos de conversación distintos, salvo que disponga de varios dispositivos con varias cuentas de telegram.

13 https://nordicapis.com/developing-secure-bots-using-the-telegram-apis/
14 https://pytba.readthedocs.io/en/latest/index.html#indices-and-tables
15 https://telegram.org/
16 https://core.telegram.org/bots/api

1.8.4 Tecnologías análisis de datos

Imagen 8. Librerías científicas

1. **Scikit learn**

Scikit-learn[17] es una biblioteca de Python diseñada para la minería de datos y el análisis de datos. Proporciona una amplia gama de herramientas eficientes para el aprendizaje automático, como algoritmos de clasificación, regresión, clustering, reducción de dimensionalidad, y más. Está construida sobre NumPy, SciPy y matplotlib.

2. **NumPy**

NumPy[18] es una biblioteca fundamental para la computación científica en Python. Proporciona soporte para arrays y matrices multidimensionales, junto con una colección de funciones matemáticas de alto nivel para operaciones rápidas y eficientes en estos arrays. Es la base sobre la cual se construyen muchas otras bibliotecas científicas de Python.

3. **Pandas**

Pandas[19] es una biblioteca de Python que proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento y fáciles de usar. Introduce dos estructuras de datos principales: Series (una estructura unidimensional) y DataFrame (una estructura bidimensional similar a una tabla de una base de datos). Pandas facilita la manipulación, análisis y visualización de datos.

4. **SciPy**

SciPy[20] es una biblioteca de Python utilizada para la computación científica y técnica. Construida sobre NumPy, proporciona rutinas eficientes y fáciles de usar para la integración numérica, álgebra lineal, optimización, interpolación, transformadas de Fourier, procesamiento de señales y otras tareas comunes en la ciencia y la ingeniería.

5. **StatsModels**

StatsModels[21] es una biblioteca de Python que proporciona clases y funciones para la estimación de muchos modelos estadísticos diferentes, así

[17] https://scikit-learn.org/stable/
[18] https://numpy.org/
[19] https://pandas.pydata.org/
[20] https://scipy.org/
[21] https://www.statsmodels.org/stable/index.html

como para realizar pruebas estadísticas y la exploración de datos (Stats Models s.f.). Ofrece una amplia gama de modelos estadísticos (como regresión lineal, modelos de series temporales y modelos de probabilidad) y herramientas de análisis para trabajar con datos estadísticos.

6. Matplotlib

Matplotlib[22] es una biblioteca de visualización de datos en Python que produce gráficos en 2D con calidad de publicación. Es altamente configurable y se puede utilizar para crear una amplia variedad de gráficos, como líneas, barras, histogramas, dispersión, gráficos tridimensionales, entre otros. Matplotlib es una herramienta fundamental para el análisis de datos y la creación de gráficos informativos y atractivos.

Para obtener correlaciones estadísticas entre parámetros y métricas en Python, la biblioteca más adecuada sería **Pandas** junto con **SciPy** y **Statsmodels**. Estas bibliotecas ofrecen diversas herramientas para calcular y analizar correlaciones estadísticas de manera eficiente.

[22] https://matplotlib.org/

1.9 Descripción de la solución propuesta

Se propone alojar el proyecto en un repositorio de git de forma que se descargue e instale en un servidor de forma sencilla, las funcionalidades, a grandes rasgos, son:

1. **Preprocesamiento**. Elimina del corpus la información mal seleccionada, direcciones duplicadas o direcciones no accesibles. Esto no sería necesario en caso de tener una interfaz para el cliente.
2. **Creación de base de datos e índices** a partir de las fuentes seleccionadas y previamente filtradas. Utiliza la información textual para crear un índice de vectores dentro de una base de datos vectorial.
3. **Manejador de mensajes**, un hilo por cada mensaje recibido en telegram. Para cada mensaje recibido por el bot, se realizará un procesamiento independiente.
4. **Post procesamiento**. Aquellos usuarios que participen en el proceso de evaluación y validación del sistema, habrán generado un archivo JSON que asocia métricas a parámetros del sistema durante cada consulta realizada.
5. No será necesario implementar clase alguna externa a las que encontramos en las librerías utilizadas.
6. Se utilizará un hashmap, implementado a partir de un map de python para mantener el contexto de cada conversación.
7. Se utilizará un archivo JSON para mantener accesibles los metadatos de la colección creada.
8. Se utilizará otro archivo JSON para almacenar los datos de validación de los parámetros de funcionamiento y las métricas.

1.10 Material y metodología

Se trabaja con dos ordenadores portátiles, uno para desarrollar el código y otro para abrir telegram desktop[23] y hacer pruebas, además del material utilizado -teléfonos móviles u ordenadores personales- por los usuarios finales que participan en el proceso de validación.

Como metodología, se apuesta por realizar un **desarrollo incremental (1.12)**, esta decisión se debe a:

1. Detección temprana de errores
2. Retroalimentación con el cliente
3. Retroalimentación con el usuario
4. Más calidad con menos esfuerzo

1.11 Pila de tecnologías utilizadas

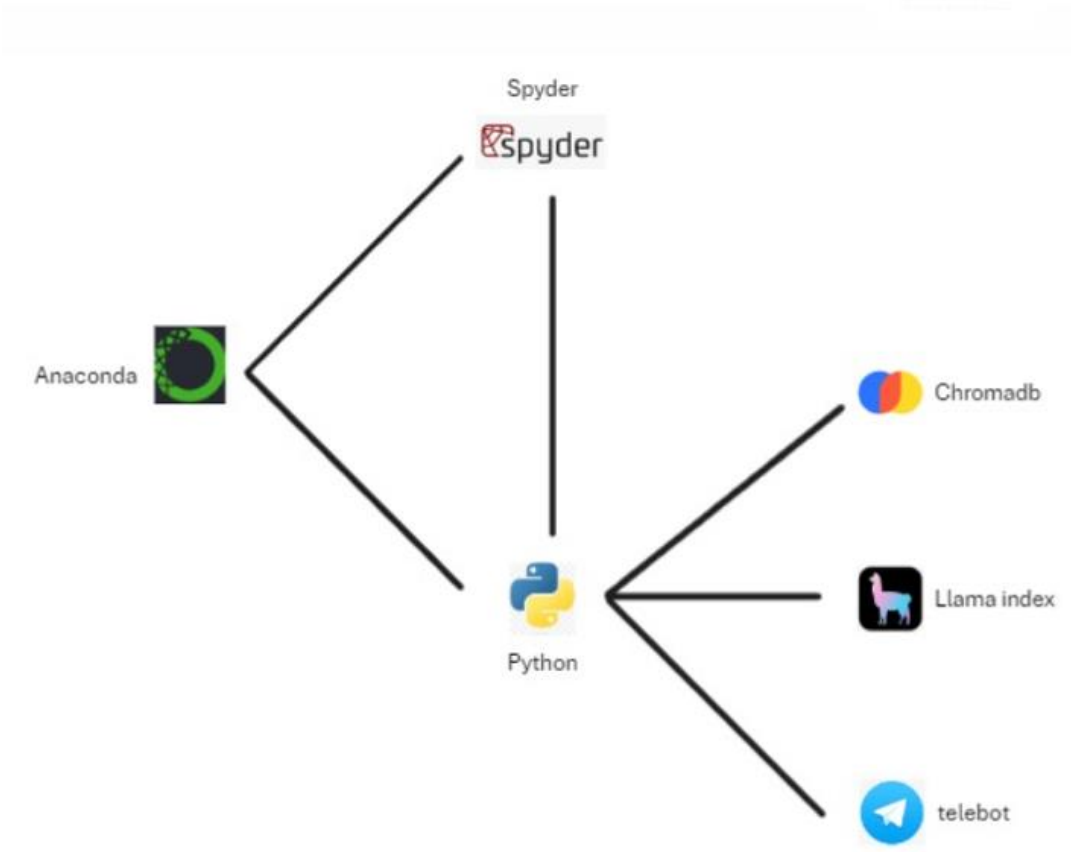


Figura 2. Pila de tecnologías

[23] https://desktop.telegram.org/

Entorno Virtual de Anaconda [24]

Descripción: Anaconda es una distribución de Python y R para la ciencia de datos y el aprendizaje automático que simplifica la gestión de paquetes y entornos. Utiliza conda como su sistema de gestión de paquetes y entornos virtuales.

- Propósito: Anaconda facilita la creación y gestión de entornos virtuales aislados, asegurando que las dependencias del proyecto no entren en conflicto con otras aplicaciones.
- Integración: Se utiliza para gestionar las bibliotecas de Python necesarias para el proyecto, incluyendo llama-index, chromadb y telebot, asegurando la compatibilidad y la reproducibilidad del entorno de desarrollo.

Llama-index

Descripción: llama-index es una biblioteca especializada en la construcción y manipulación de índices de vectores, optimizados para consultas rápidas y eficientes.

- Propósito: Se emplea para indexar y buscar información de manera eficiente dentro de grandes conjuntos de datos [6].
- Integración: Integrado en el backend del proyecto para proporcionar capacidades de búsqueda y recuperación de datos, permitiendo consultas rápidas y precisas sobre el contenido almacenado.

Chromadb

Descripción: chromadb es una base de datos orientada a grafos diseñada para manejar grandes volúmenes de datos interconectados. Proporciona capacidades avanzadas para almacenar, consultar y analizar relaciones complejas entre datos [7].
Uso en el Proyecto:

- Propósito: Utilizada para almacenar y gestionar datos estructurados como grafos, permitiendo consultas avanzadas y análisis sobre las relaciones entre los datos.
- Integración: Implementada como la base de datos principal del proyecto, facilitando la gestión de datos interconectados y mejorando el rendimiento de las consultas complejas.

[24] https://www.anaconda.com/

Telebot

Descripción: telebot es una biblioteca de Python para interactuar con la API de telegram, permitiendo la creación de bots de telegram de manera sencilla y eficiente.

- Propósito: Desarrollar un bot de telegram que interactúe con los usuarios, proporcionando funcionalidades específicas como notificaciones, respuestas automatizadas y consultas de datos.
- Integración: Implementado para manejar la comunicación entre el sistema y los usuarios finales, utilizando la API de telegram para recibir y responder mensajes, y para integrar las funcionalidades de indexación y búsqueda proporcionadas por llama-index y chromadb.

1.12 Metodología de desarrollo de software

La metodología de **desarrollo incremental** es una estrategia de gestión y desarrollo de proyectos de software en la que el sistema se construye, prueba y entrega en partes pequeñas y manejables llamadas incrementos [3]. Cada incremento representa una versión completa del producto que incluye una porción de la funcionalidad total esperada. A lo largo del proyecto, estos incrementos se combinan para formar el sistema completo.

Se concretará cada incremento en el apartado de planificación temporal (1.14)

Características del desarrollo incremental

1. Desarrollo por Etapas:
    - El proyecto se divide en pequeños subproyectos (incrementos), cada uno de los cuales se centra en un conjunto específico de funcionalidades.
    - Cada incremento es una versión operativa del producto, aunque sea una versión limitada.
2. Entrega Frecuente:
    - Los incrementos se desarrollan en iteraciones cortas, generalmente de dos a cuatro semanas.
    - Al final de cada iteración, se entrega un incremento que puede ser evaluado y probado por los interesados.
3. Feedback Continuo:
    - Los usuarios y otros interesados proporcionan feedback sobre cada incremento entregado.
    - Este feedback se utiliza para ajustar y mejorar los incrementos posteriores, asegurando que el producto final cumpla con las expectativas y necesidades del cliente.
4. Flexibilidad y Adaptabilidad:
    - La metodología permite realizar cambios en los requisitos y prioridades a medida que se desarrolla el proyecto.
    - Los cambios se pueden incorporar en incrementos futuros sin afectar negativamente al trabajo ya realizado.

1.13 Análisis de riesgos

A continuación, se enumeran los riesgos que, a priori, se vislumbran más obvios.

1. **Gestión del Entorno**

   - Configuración del Entorno

   Riesgo: Configuración incorrecta del entorno virtual de Anaconda.

   - Actualización de Bibliotecas

   Riesgo: Actualizaciones de bibliotecas pueden introducir incompatibilidades.

2. **Rendimiento**

   - Escalabilidad del Sistema

   Riesgo: Problemas de rendimiento con grandes volúmenes de datos en llama-index y chromadb.

   - Latencia en Respuestas del Bot

   Riesgo: Alta latencia en la respuesta del bot de telegram.

3. **Operación del Sistema**

   - Fiabilidad del Bot

   Riesgo: El bot de telegram se desconecta o falla frecuentemente.

   - Mantenimiento y Actualización

   Riesgo: Dificultades en el mantenimiento y actualización del sistema sin interrumpir el servicio.

4. **Integración y Compatibilidad**

   - Integración de Múltiples Tecnologías

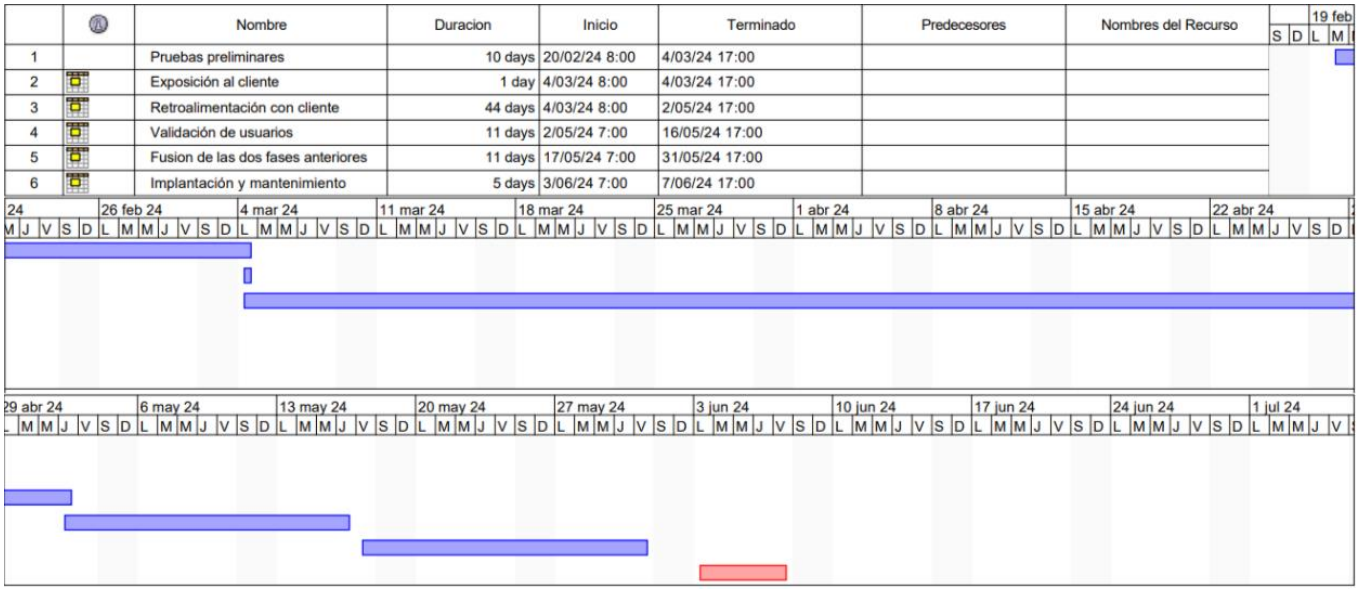
   Riesgo: Problemas de integración entre Anaconda, llama-index, chromadb y telebot.

   - Compatibilidad de Versiones

   Riesgo: Incompatibilidades entre versiones de las bibliotecas utilizadas.

1.14 Planificación temporal

A continuación, se muestra la planificación con sus correspondientes tareas o fases del proyecto, diagrama de Gantt, así como una breve explicación de cada tarea. Para la elaboración del diagrama se ha utilizado Project libre [25].



	Nombre	Duracion	Inicio	Terminado	Predecesores	Nombres del Recurso
1	Pruebas preliminares	10 days	20/02/24 8:00	4/03/24 17:00		
2	Exposición al cliente	1 day	4/03/24 8:00	4/03/24 17:00		
3	Retroalimentación con cliente	44 days	4/03/24 8:00	2/05/24 17:00		
4	Validación de usuarios	11 days	2/05/24 7:00	16/05/24 17:00		
5	Fusion de las dos fases anteriores	11 days	17/05/24 7:00	31/05/24 17:00		
6	Implantación y mantenimiento	5 days	3/06/24 7:00	7/06/24 17:00		

Diagrama 1. Planificación temporal

- Fase 0: Formación. Comienza 4 de diciembre, fecha límite 19 febrero

El desarrollo comienza por una primera fase, previa a la elaboración del proyecto. Es decir, formalmente se comienza el proyecto el día 4 de marzo, pero previamente se han dedicado cerca de tres meses a comprender y asimilar cada concepto y tecnología utilizados.

Este esfuerzo en concepto de “horas de formación” no se tendrá en cuenta en la planificación temporal ni en la estimación de tamaño y esfuerzo, ni en las 300 horas estimadas para elaborar un TFG, que previo a redactar este documento, están más que sobrepasadas.

- **Fase 1: Pruebas preliminares**. Comienza 20 febrero, fecha límite 3 marzo

Antes de hacer partícipe al cliente, se harán una serie de pruebas preliminares con las tecnologías más allá del concepto teórico, esto ayudará al desarrollador a comprender aspectos que, en el marco teórico, parecían ideas vagas, además de ir comprendiendo el impacto de algunos de los parámetros del sistema y en su funcionamiento.

[25] https://www.projectlibre.com/

Haciendo referencia al análisis de riesgos anterior, se debe tener mucho cuidado con el proceso de configuración del entorno virtual de anaconda y con las dependencias en el momento de instalar librerías.

- **Fase 2: Exposición al cliente**. Día 4 de marzo

Se concreta una reunión con Enrique Jerez y algunos miembros del servicio con el fin de explicar el proyecto y conocer la realidad concreta del funcionamiento del servicio de gestión académica.

Se pretende con esta reunión, trazar la hoja de ruta para construir una base de conocimiento lo más sólida posible en base a las consultas y problemas que plantean los usuarios por teléfono o de forma presencial

- **Fase 3: Retroalimentación con cliente**. Comienza 4 marzo, fecha límite 3 de mayo

Tras la presentación, se mantendrán una serie de reuniones para identificar las fuentes más relevantes según el área de gestión, más concretamente:

  - Matrícula
  - Acceso
  - Reconocimiento
  - Ayuda al estudiante
  - Títulos
  - Relaciones internacionales

Esto permitirá al desarrollador preparar una base de conocimiento con la que comenzar la validación de los usuarios finales. Además, se evaluará la calidad de las fuentes de información existentes en la organización.

- **Fase 4: Validación con usuarios**. Primera mitad del mes de mayo

En esta fase el esfuerzo se centrará en capacitar a los usuarios para evaluar al sistema, y en diseñar e implementar los mecanismos que permita la comunicación entre usuarios finales y el cliente.

En esta fase tendrán más peso la interacción y la evaluación de los usuarios, esto permitirá además depurar errores y bugs.

- **Fase 5: Iteraciones de las fases 3 y 4**. Segunda mitad del mes de mayo

Durante un mes se irán repitiendo las 2 fases anteriores hasta tener un sistema completamente funcional y suficientemente robusto y fiable, para alojar en un repositorio de GIT

- **Fase 6: Implantación y mantenimiento**. Comienza a primeros de junio. Duración no determinada

En esta última fase, los esfuerzos se centrarán en adaptar el desarrollo realizado hasta ahora de manera local y en pruebas, a un típico proyecto de GIT, con el fin de que la organización pueda descargarlo a un servidor y hacerlo funcionar 24/7

2. Diseño

Exposición de los elementos más relevantes del diseño, partiendo de la Figura 1 como esquema estandarizado, se va a concretar el funcionamiento a través de la figura 3. Los diagramas de flujo se han elaborado utilizando draw.io[26].

2.1 Arquitectura y flujo de funcionamiento

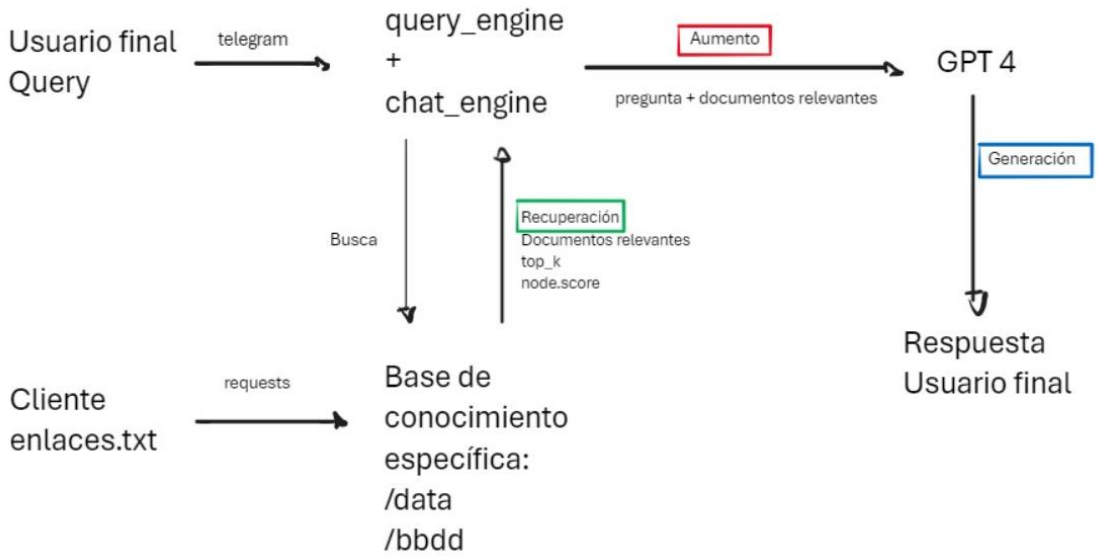


Figura 3. Flujo de funcionamiento general

Un usuario final, idealmente, un estudiante de la universidad de Jaén, realiza una pregunta al chat de **telegram**.

1. Dicha pregunta se cruza con la información indexada en chromadb a través de un **motor de consulta** contextualizado en un **motor de chat**. Los parámetros de la **recuperación** se asignan de manera aleatoria dentro de un rango razonable cada vez que se reinicia el sistema con el objetivo de enriquecer el proceso de validación.
2. Si se recupera información relevante, dicha información se **aumenta**, devolviendo al usuario una respuesta que sintetiza la información, así como una relación de las fuentes de las que se ha recuperado dicha información
3. Se **genera** una respuesta que es devuelta a través de telegram, si no se han consultado fuentes lo suficientemente relevantes, no mandará enlace alguno, y se anotará una incidencia. Esto es de vital importancia para **mantener la retroalimentación usuario-cliente**.

[26] https://app.diagrams.net/

2.2 Parámetros y métricas del sistema

Cada vez que arranca el sistema, se produce una **asignación a los parámetros de configuración pseudoaleatoria**, con el fin de dar riqueza al experimento estadístico que en gran parte motiva este trabajo.

A continuación, se va a explicar el papel de cada parámetro y cada métrica que los usuarios evalúan cuando deciden participar en el proceso de validación.

- Los parámetros se quedan fijados en el fichero config.json
- Las métricas se almacenan en el fichero evaluaciones.json

2.2.1 Parámetros

- **Chunk_size. Potencia de 2 entre la 10 y la 15**

En sistemas de este tipo, se refiere a la cantidad de información que se almacena como una unidad de información, se le puede denominar nodo o bloque según a qué aspecto del sistema se quiera hacer referencia[27].

- **Chunk_overlap. Una octava parte de la potencia de 2 fijada anteriormente**

Se refiere a la cantidad de información que comparten dos bloques consecutivos, una relación size ←→ overlap bien ajustada según el tamaño de los documentos de base, aporta al buen funcionamiento del sistema.

- **Top_k. Numero entero entre 1 y 3**

Número de nodos suficientemente similares a la consulta que se recuperan para generar la respuesta.

- **Node_score. Número real entre 0’7 y 0’8**

Basándonos en el concepto de loro estocástico presentado anteriormente, es la probabilidad a partir de la que se considera un bloque de información suficientemente similar, o lo bastante relevante.

[27] https://gustavo-espindola.medium.com/divisi%C3%B3n-y-superposici%C3%B3n-de-chunks-comprendiendo-el-proceso-112816192d08#:~:text=Entonces%2C%20la%20superposici%C3%B3n%20de%20fragmentos,informaci%C3%B3n%20data%20relevante%20al%20modelo

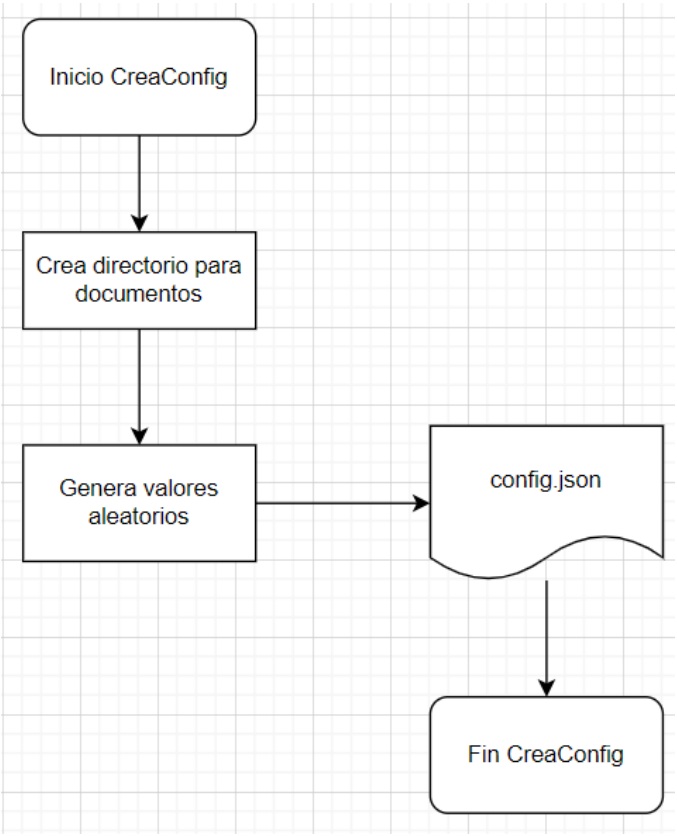


Diagrama 2. Genera configuración.

```
{
    "chunk_size": 4096,
    "chunk_overlap": 512,
    "top_k": 2,
    "node_score": 0.7017350333269053,
    "llm": 2,
    "api_key": "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",
    "bot_token": "7033107568:AAFokjJOuHiGhHCc3dZMYXcLmLY5YrJFoF8"
}
```

Imagen 9. Contenido de config.json

2.2.2 Métricas

Los valores de las métricas son enteros entre 1 y 5. Un 1 representa baja satisfacción del usuario y un 5 alta satisfacción

- **Rel_query_links**

Con esta métrica, un usuario mide la relación que hay entre la respuesta generada y las fuentes que el sistema ha considerado relevantes

- **Rel_query_response**

Con esta métrica, un usuario mide lo útil que es la información que el sistema ha considerado relevante para responder a la consulta, se pretende evaluar de manera independiente con respecto a la métrica anterior, aquí se mide la relación entre consulta y respuesta dejando al margen las fuentes

- **Context**

Con esta métrica, un usuario mide lo bien o mal que el sistema ha capturado el contexto, es muy útil cuando se pretende evaluar una secuencia de mensajes que componen la conversación con el sistema, más que una consulta de manera aislada.

- **Time**

El usuario mide si el tiempo de respuesta es satisfactorio.

- **Correction**

El usuario mide lo correcto que es el sistema al generar respuestas, si utiliza los distintos idiomas de forma consistente y se ajusta a las reglas gramaticales y semánticas, en resumen, se considera perfecto si se tiene la sensación de estar hablando con un ser humano.

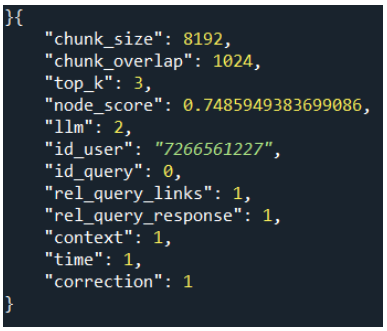


Imagen 10. Ejemplo de una evaluación

2.3 Elementos del sistema

El directorio del proyecto queda con esta estructura, a falta de realizar la versión final que quedaría alojada en el repositorio.

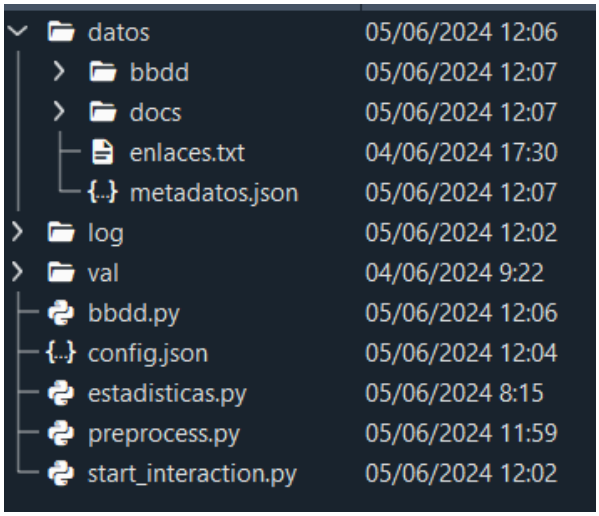


Imagen 11. Estructura del proyecto

2.3.1 Las fuentes: enlaces.txt

La elaboración y mantenimiento de este elemento del sistema es uno de los puntos más críticos. Se ha diseñado un proceso de retroalimentación muy sencillo**. Cada vez que, para una pregunta no se recupera información suficientemente relevante, dicha pregunta es volcada en un archivo, concretamente, dentro de la jerarquía del proyecto "log/incidencias.json"**

El objetivo de este mecanismo, es que los diversos equipos de gestión y administración de la universidad, puedan comprobar en tiempo real:

- Qué información se quiere recuperar
- Pero no se encuentra alojada en la base de conocimiento

De esta manera, se puede reindexar la información a medida que se identifiquen un número significativo de incidencias, **añadiendo nuevas direcciones a enlaces.txt**

Ahora se va a **describir el procedimiento seguido para construir este archivo**, aunque parezca un elemento sencillo o simple es, sin duda, el **aspecto central** de todo el sistema de recuperación de información. La versión definitiva de este archivo, así como las sucesivas reuniones con los equipos de gestión de la universidad que han llevado a la misma, son de especial relevancia.

Es poco frecuente que en un trabajo de fin de grado se trabaje con un cliente real y merece ser puesto de relieve todo el trabajo y esfuerzo realizados.

El procedimiento en cuestión se ajusta a las fases 2 3 y 4 de la planificación temporal expuesta en el punto 1.14 de este documento.

1. En la fase 2 se mantuvo una reunión con Enrique Jerez, uno de los jefes de servicio de la universidad, que recibió de buen grado la propuesta de participar en la elaboración de la base de conocimiento del sistema, se transmitió la importancia de la participación de los verdaderos **protagonistas** de este proyecto, por su **conocimiento de la institución**, que son los distintos servicios de la universidad.
2. Durante la fase 3, se necesitaron dos reuniones para transmitir correctamente la finalidad y bases técnicas del sistema, las áreas de gestión que accedieron a participar en el proceso son:
    a. Acceso
    b. Títulos
    c. Reconocimientos
    d. Matrícula
    e. Además, se incorporó información referente a relaciones internacionales, ayuda al estudiante, emprendimiento y deportes, aunque no se ha trabajado directamente con los responsables, por lo que no se tiene un organigrama tan claro como el de las otras áreas.
3. Durante la fase 4, se mantuvieron otras tres reuniones en las que se esclarecieron varios aspectos muy delicados
    a. La **recuperación** de información es muy **dependiente** de cómo esté redactado el cuerpo de texto de cada documento HTML.
    b. Era de vital importancia definir una **etiqueta** para cada enlace, con el fin de colocar dicha etiqueta en el **botón** en cuestión al recuperar las fuentes relevantes para construir una respuesta.
    c. Había serios **problemas** para coordinar el **idioma** de la respuesta con el de la pregunta.
4. A partir de aquí se pretendía continuar trabajando, pero dado que la naturaleza de este proyecto no deja de ser académica, se dejó aparcado con el fin de elaborar la presente documentación. Además, justo la época de la PEvAU es un “vacío administrativo”, **con mucha normativa**, plazos, precios y otros **aspectos pendientes de aprobar por parte del consejo de gobierno**, por lo que seguir trabajando carecía de sentido hasta tener la información del curso siguiente, 2024 – 2025, actualizada.

5. Se procedió a incorporar a usuarios finales, estudiantes actuales y egresados, a la validación del sistema, utilizando el enlace de telegram, y se incorporó información referente a
    a. Reserva de instalaciones deportivas
    b. Aprobar una asignatura en compensatoria
    c. Se depuraron numerosos bugs y errores gracias a las interacciones capciosas de algunos de los participantes, como la búsqueda de emojis en un modelo de lenguaje o de caracteres no latinos.

2.3.2 Los datos

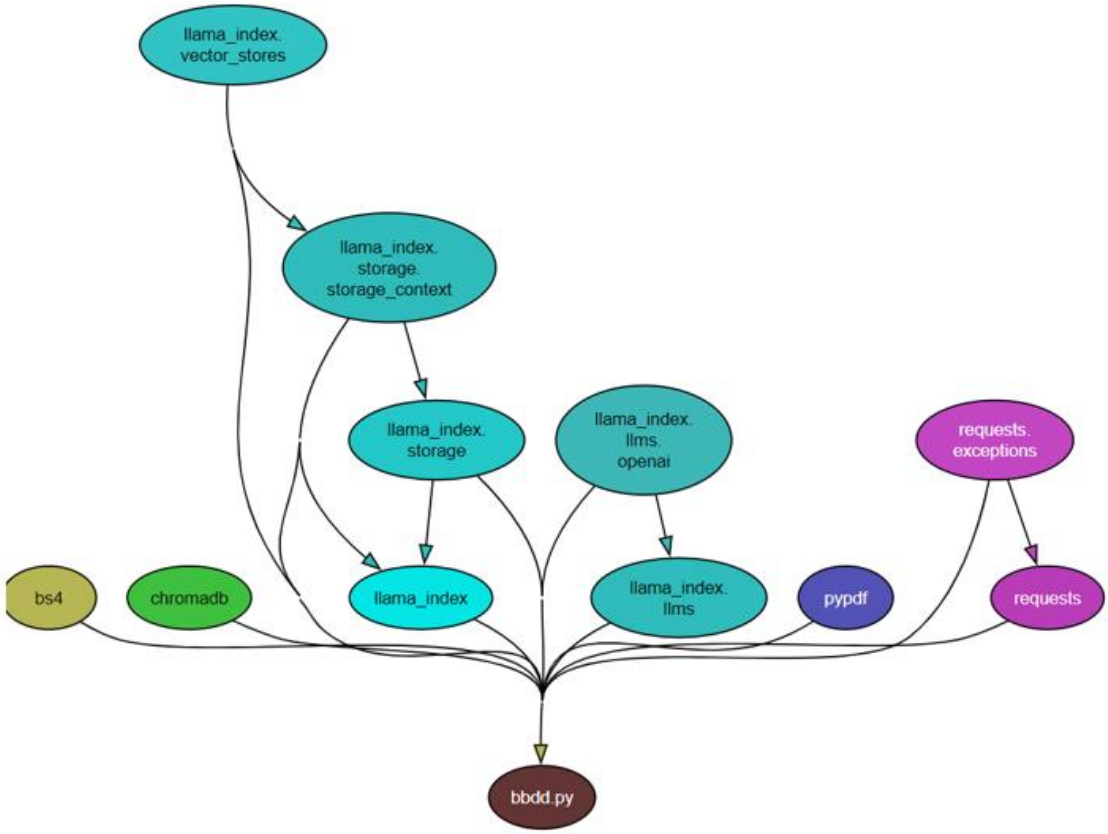


Diagrama 3. Dependencias bbdd.py

Paso 1 enlaces → documentos

Procesando cada línea de enlaces.txt, se va comprobando:

- que son direcciones accesibles
- que no están repetidas

Esto último no es raro dado que distintas áreas de gestión de la universidad comparten normativas y procedimientos. Una vez filtradas, las direcciones son descargadas a través de requests. **Cada archivo se crea con un identificador único en su propio nombre de fichero, crucial para los metadatos**.

- Las direcciones que apuntan a archivos pdf, son descargadas directamente
- Las direcciones que apuntan a distintas vistas de la web de la universidad, son descargadas tras un procesamiento a través de beautifulsoup.

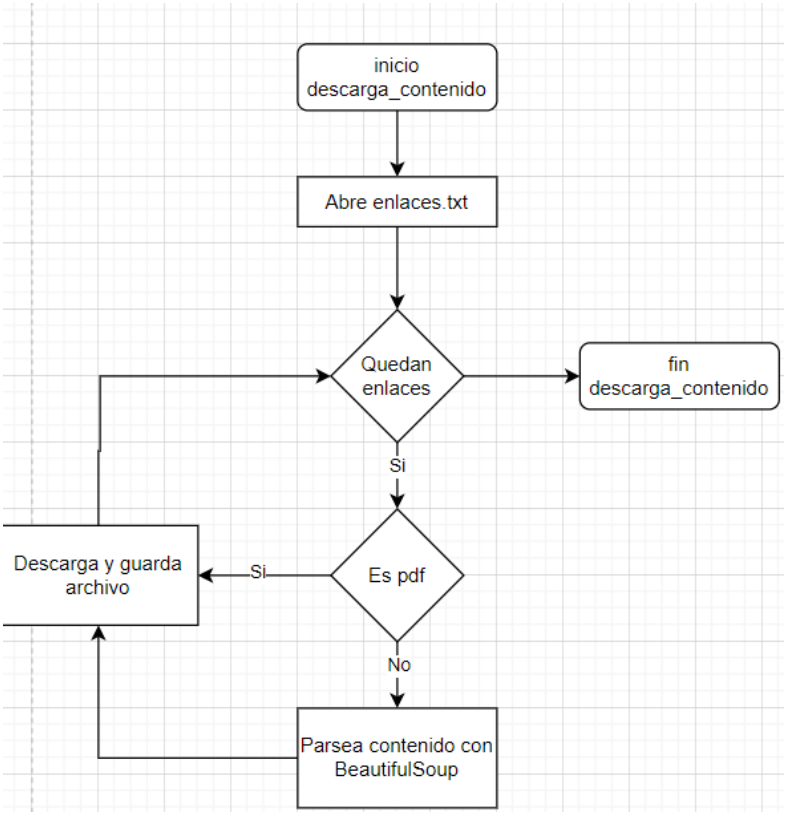


Diagrama 4. Flujo de descarga corpus

El resultado de los **requests**, se guardan en el directorio **/datos/docs**

Paso 2 documentos → índices

A partir de los documentos descargados, se crea una colección dentro de una base de datos vectorial, en la que se almacena un índice de vectores de chromadb, este se guarda como colección UJA-DATA en la base de datos. Los parámetros de configuración que se utilizan para el service_context se toman del fichero config.json

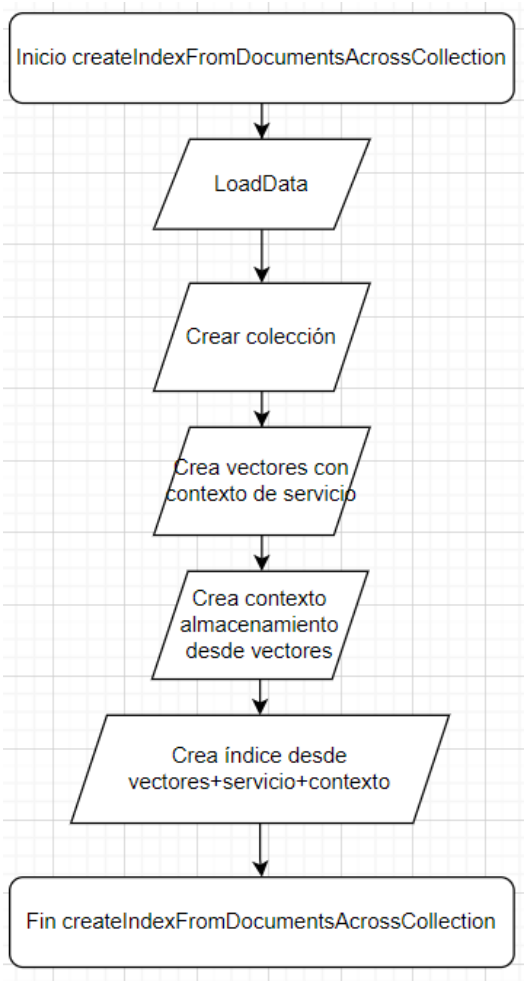


Diagrama 5. Flujo creación y configuración vectores

¿Qué es service_context? ¿Qué es storage_context?

- La primera es una estructura fundamental para configurar el pipeline de indexación, modelo de lenguaje, modelo de incrustación, node parser, etc.
- La segunda se utiliza para especificar cómo y dónde se almacenan las incrustaciones de los documentos.

Este índice con estas estructuras almacenadas, será cargado posteriormente en el manejador de mensajes del bot de telegram

Paso 3 índices→metadatos

Asociamos la etiqueta asignada por el cliente a cada url en los metadatos de cada nodo de la colección, este mecanismo es el que nos permitirá **recuperar fuentes para el usuario de forma cómoda y usable** a nivel de interfaz

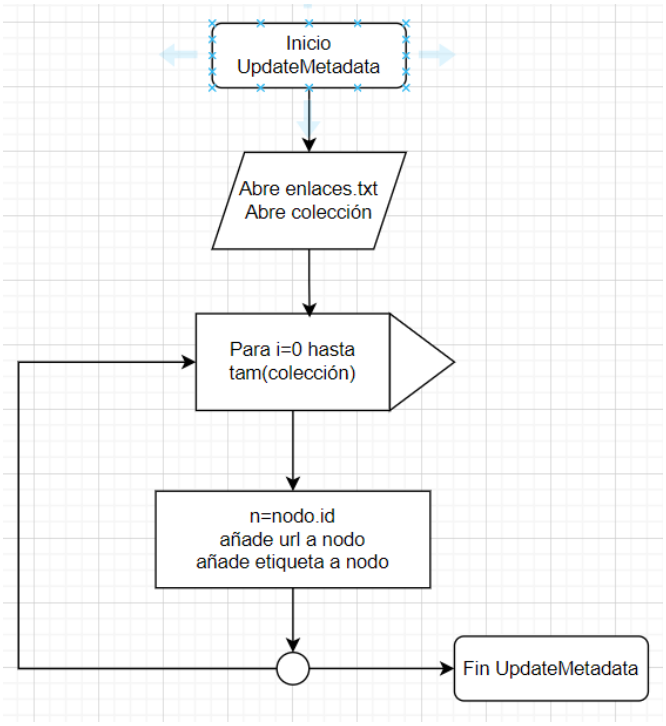


Diagrama 6. Flujo metadatos

2.3.3 El retriever

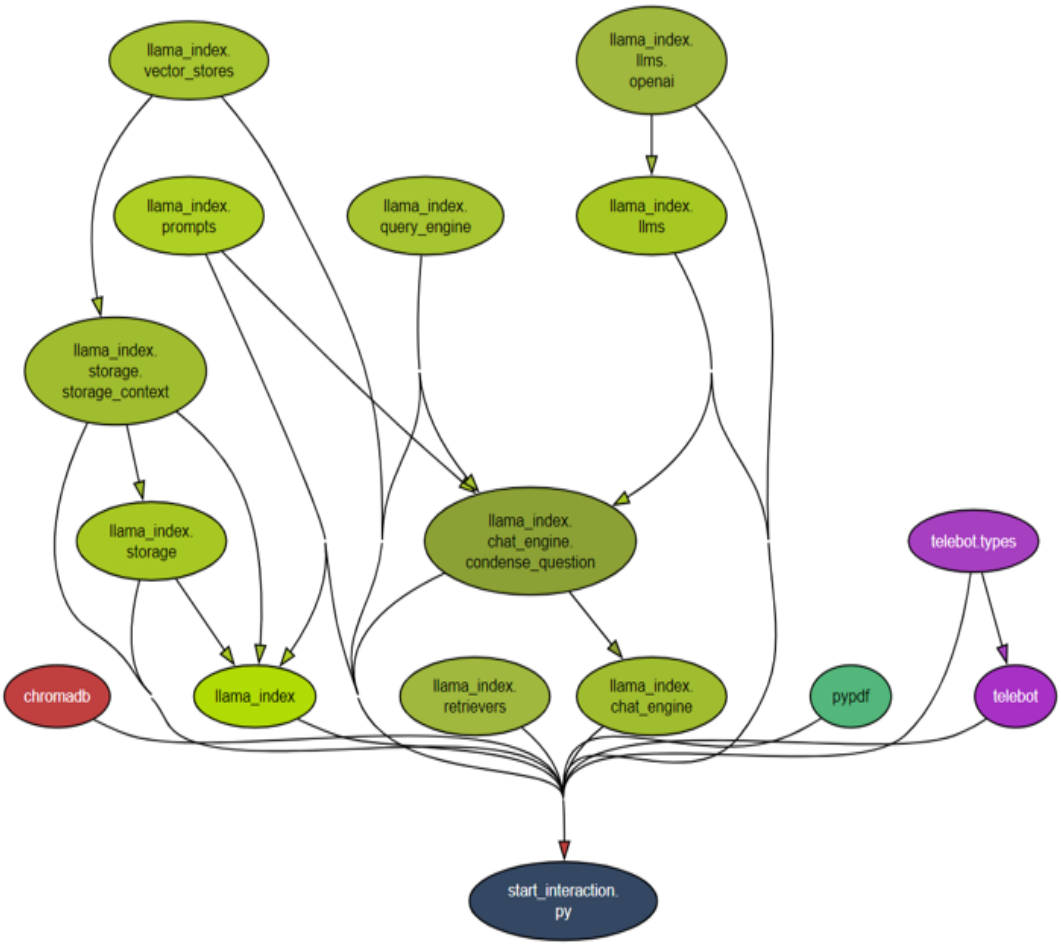


Diagrama 7. Dependencias start_interaction.py

Visión general

A partir de la colección creada anteriormente, creamos un índice, y a partir del índice, **un motor de consulta, que es, esencialmente, el encargado de recuperar información** a través de un criterio de similitud, trata de recuperar más probable según la consulta en las fuentes proporcionadas. Dicho motor será la herramienta de consulta para cada motor de conversación creado, uno para cada usuario de manera independiente. Inicialmente, el mapa de conversaciones estará vacío.

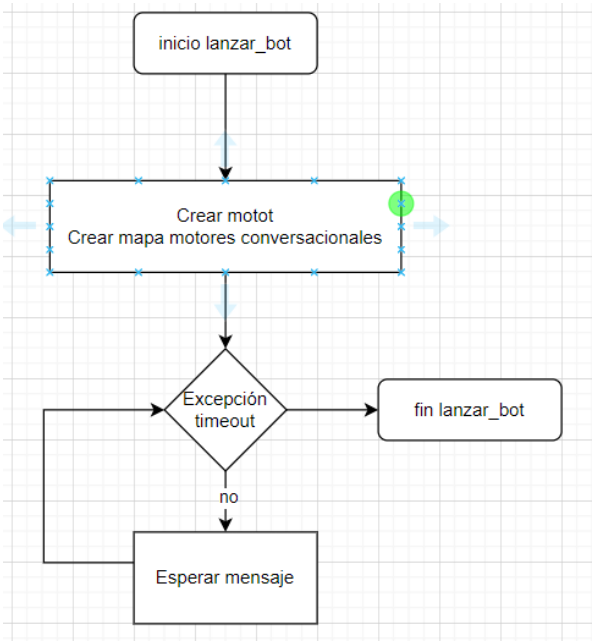


Diagrama 8. Lanzar bot

Colección → índice

Se crea el índice de búsqueda utilizando los mismos parámetros de indexación y almacenamiento que se usaron en el momento de la creación, por recomendación de la propia librería en su documentación

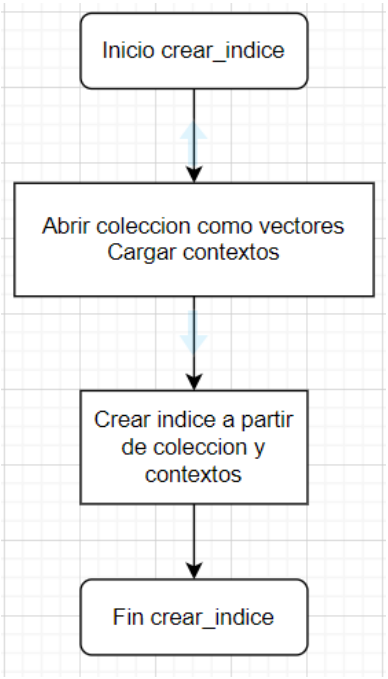


Diagrama 9. Creación de índice en el retriever

Índice → motor de consulta

El retriever se configura para recuperar los k-mejores nodos y sus correspondientes k-mejores fuentes. Una vez creado el índice, como se aprecia en el diagrama anterior, se crea un motor de consulta. A partir de este elemento, se crea un motor de chat único, asociando una plantilla a cada usuario que se va modificando con el contenido de cada chat de forma independiente.

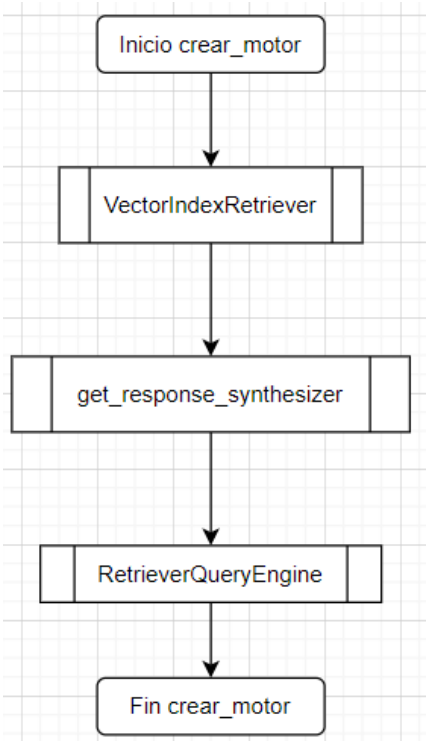


Diagrama 10. De índice a query_engine

2.3.4 El log del sistema

- Para tener un buen registro de lo que ocurre durante la ejecución del sistema de recuperación, se ha guardado la interacción **de cada usuario de forma anónima** gracias al ID que ofrece la API de telegram, un json para cada usuario.
- Cada vez que se recupera una **respuesta sin fuentes** lo suficientemente relevantes, la consulta se almacena como una **incidencia** junto con el parámetro en cuestión en incidencias.json
- Cada vez que un usuario decide **participar** en el proceso de evaluación anteriormente explicado, el bot almacena los parámetros activos junto con las métricas del usuario (imagen 12).

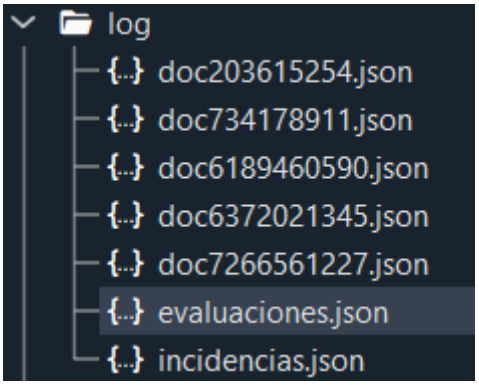


Imagen 12. LOG del sistema

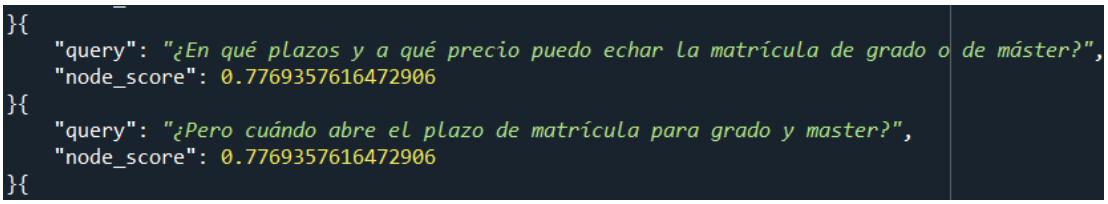


Imagen 13. Contenido de incidencias.json

2.3.5 Interacción de los usuarios con el recuperador

A continuación, se explicarán las tres mecánicas de interacción que implementa la interfaz

Mensaje de bienvenida

Cuando un usuario abre por primera vez el chat con UJA-GPT, se encuentra con un mensaje de bienvenida, así como con una invitación a participar en el proceso de validación, adjuntando algunos archivos explicativos sobre dicho proceso.

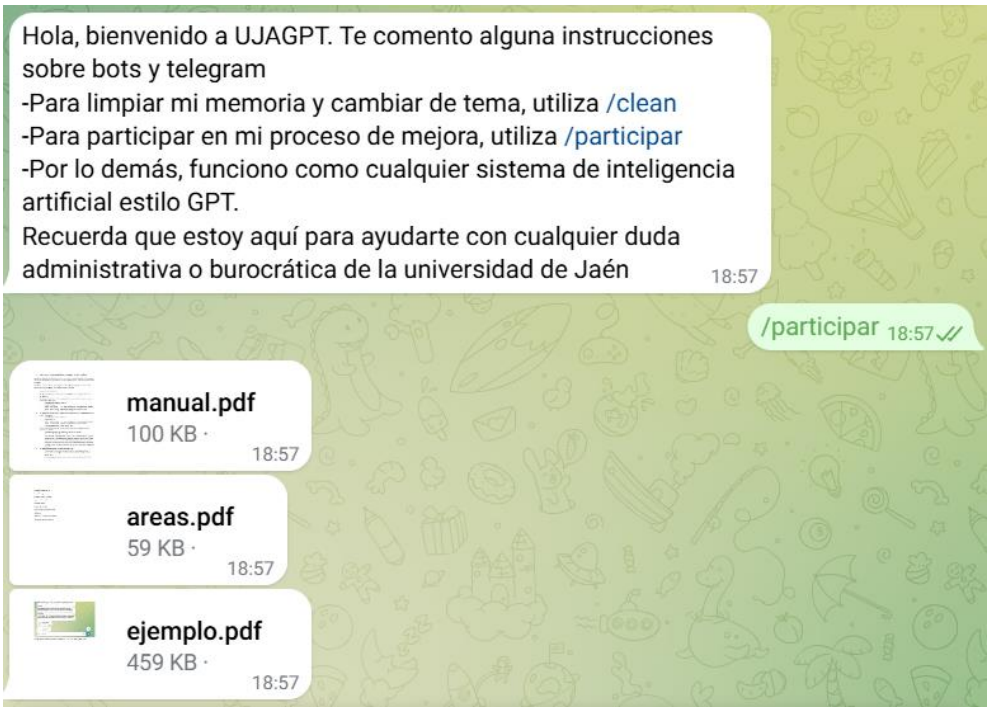


Imagen 14. Mensaje de bienvenida

Evaluación de una interacción

Cada vez que un usuario desea evaluar la calidad del sistema, el mensaje se encabeza con /evaluar, así, el sistema captura el mensaje, que debe tener un **formato adecuado**, en el archivo evaluaciones.json antes mencionado. Como se observa en el código, la evaluación se añade asociando los parámetros activos, a las puntuaciones dadas por el usuario, lo que **permitirá establecer relaciones estadísticas parámetro-puntuación** que, de cara al futuro, podrían permitir implementar sistemas a partir del actual, que aprendan de sí mismos según los usuarios los evalúen.

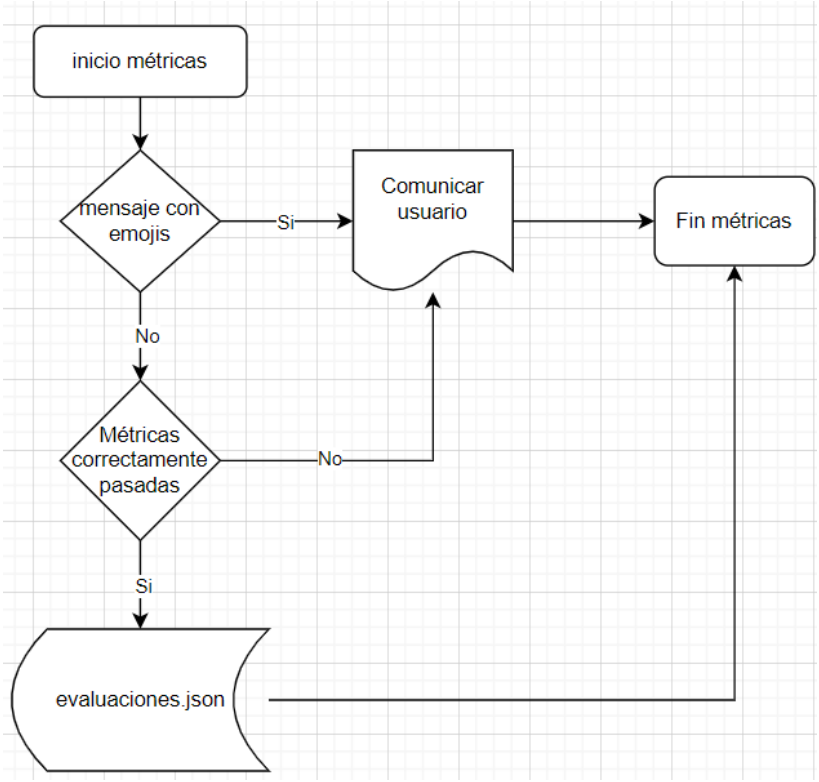


Diagrama 11. Evaluación de la interacción por parte de un usuario

La recuperación de información

Cada mensaje recibido, se maneja a través de la siguiente función, se lanza un hilo para cada mensaje de manera independiente. Bajo los diagramas de flujo, se van a enumerar y comentar los aspectos más relevantes.

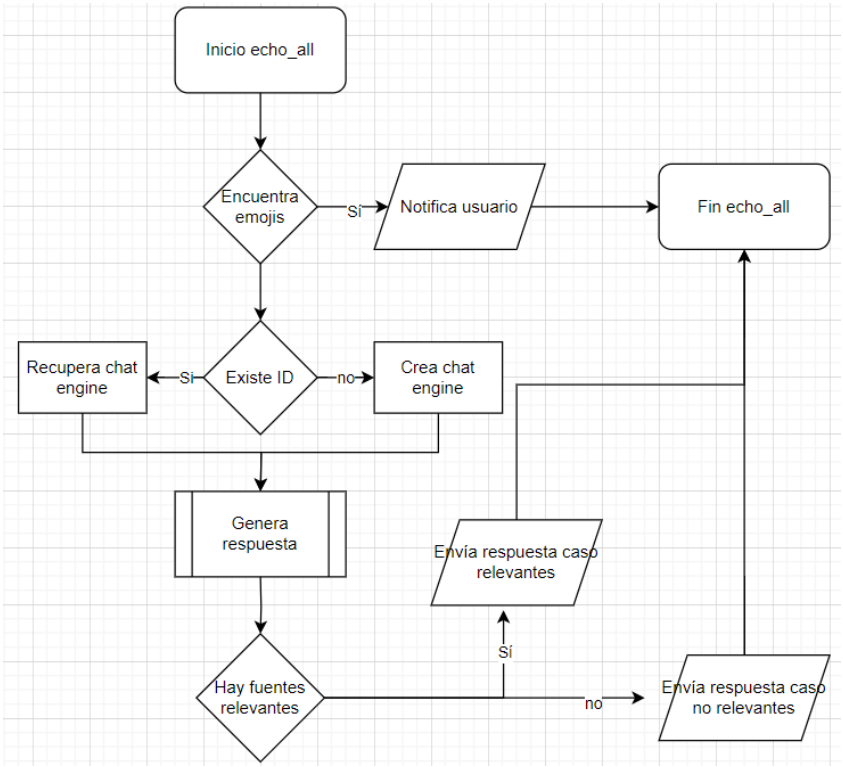


Diagrama 12. FLUJO PRINCIPAL echo_all

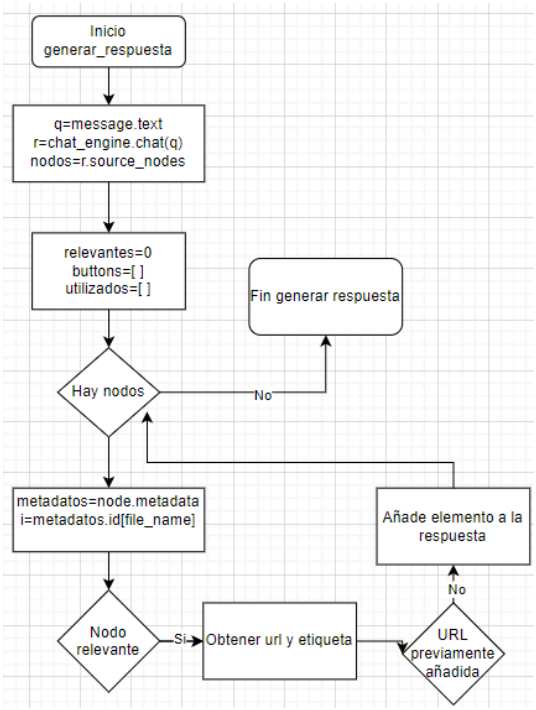


Diagrama 13. Flujo de la generación de una respuesta

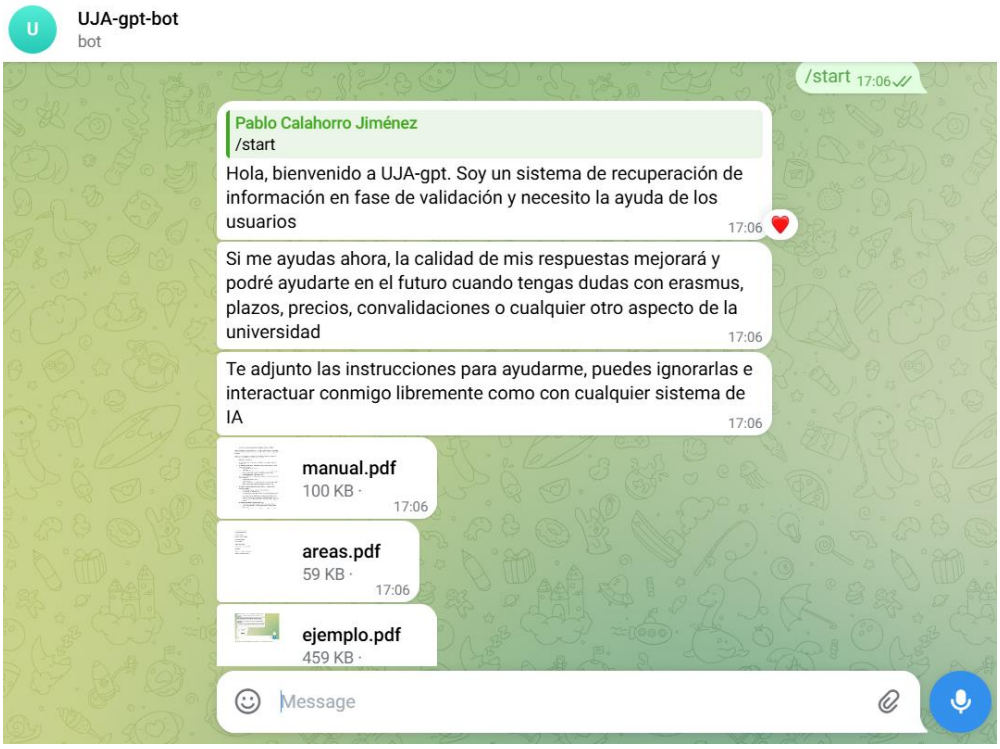
- Cada mensaje tiene asociado el ID del chat en el que ha sido recibido. Esto es crucial dado que nos permite almacenar cada motor de chat en el proceso de recuperación (mapa_motores), **manteniendo el contexto de la conversación de cada usuario por separado**.
- Si el ID del chat no se encuentra en mapa_motores, quiere decir que este usuario no tiene contexto previo, se crea uno vacío con la promp template pertinente.
- Una vez extraído el texto de la pregunta, y asociado este al contexto de la conversación a través del chat_engine, el método **chat_engine.chat()** es el que devuelve la respuesta y las fuentes relevantes.
- Las fuentes relevantes se extraen de **source_nodes**, asociadas al dato devuelto por el método anterior. Esto nos permite, gracias a los metadatos, saber en qué enlace de todos los que componen nuestra base de conocimiento, se ha recuperado la información
- ¿Qué ocurre si las fuentes **no son lo "suficientemente relevantes"**? Se recorren todas chequeando que no se queden por debajo de la calidad esperada, de esta manera, si las fuentes no tienen suficiente similitud con la consulta enviada a chat_engine.chat(), el **contador relevantes** se quedará a 0, por lo que se hará un volcado como incidencia. La respuesta se enviará al usuario de todas maneras por si lo que se pretende es echar el rato charlando con la inteligencia artificial, como es habitual por parte de usuarios de este tipo de sistemas.
- En caso de encontrar fuentes relevantes, se recuperarán las **etiquetas** para crear botones que enlacen al usuario con las fuentes relevantes, además de enviar la respuesta generada como cabecera de las fuentes. En este caso de almacena la interacción en el json creado para este usuario.
- Para reiniciar el contexto de la interacción es necesario que el usuario evalúe la interacción, se pretende incentivar la participación de los usuarios en la validación del sistema. También se reinicia cuando se relanza el sistema.

3. Experimentación

3.1 Pruebas de funcionamiento

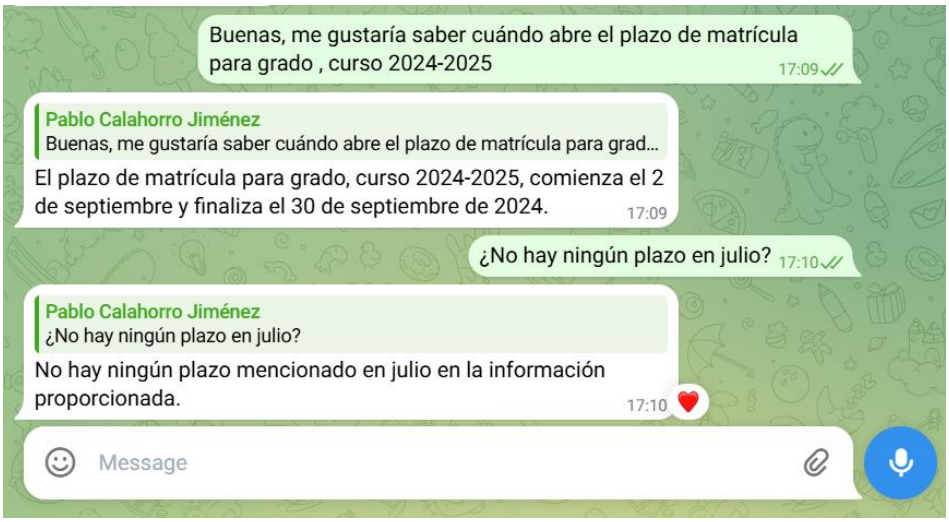
A continuación, se van a mostrar algunas de las interacciones más relevantes según una serie de criterios, como detección de errores, funcionamientos inesperados o un gran resultado en la recuperación.

Prueba 1. La bienvenida



Este es el mensaje de bienvenida que envía el sistema cuando se abre el chat, ha sido un fracaso dado que la participación de usuarios en el proceso de validación no ha dado como resultado datos consistentes ni estadísticamente coherentes

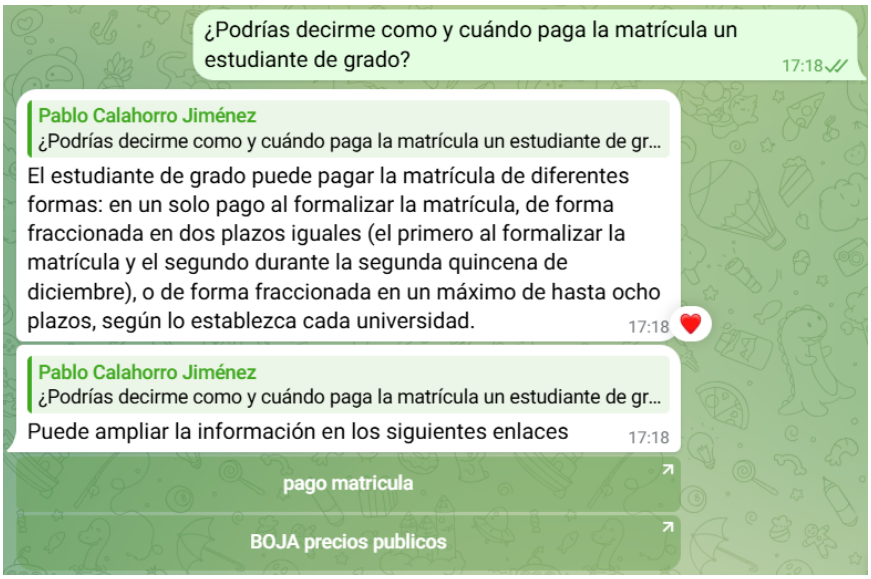
Prueba 2. Pregunta simple con una alta puntuación de similitud

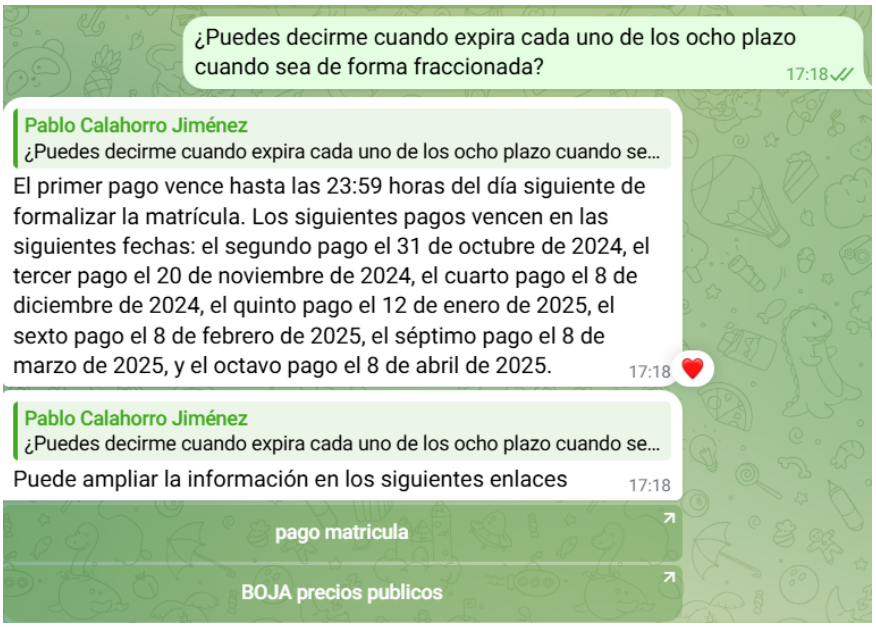


En la imagen anterior, se observa una respuesta que cuadra a la perfección con la información recogida en la web y en la normativa. Sin embargo, no adjunta las fuentes relevantes.

¿por qué ocurre esto? Porque durante la ejecución de la recuperación anterior, la puntuación de similitud para recuperar el documento asociado al nodo, estaba muy cerca de 0’8, lo que supone un desajuste con la realidad, dado que hay fuentes muy relevantes que se han quedado cerca del corte pero fuera.

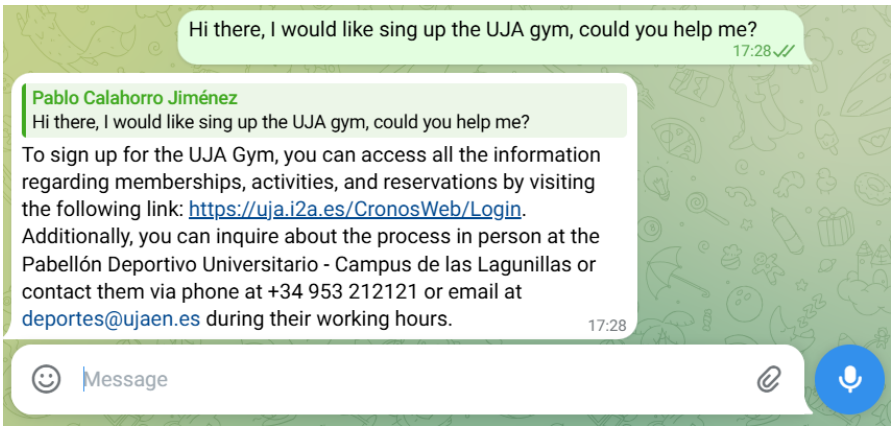
Prueba 3. Pago de la matrícula





Esta prueba sirve **como ejemplo perfecto de funcionamiento** en todos los sentidos, fuentes y construcción de respuesta, ahorrando a los usuarios la necesidad de navegar por las distintas vistas web de la universidad para encontrar la información, pero enviando dichas direcciones url por si el usuario desea ampliar la información.

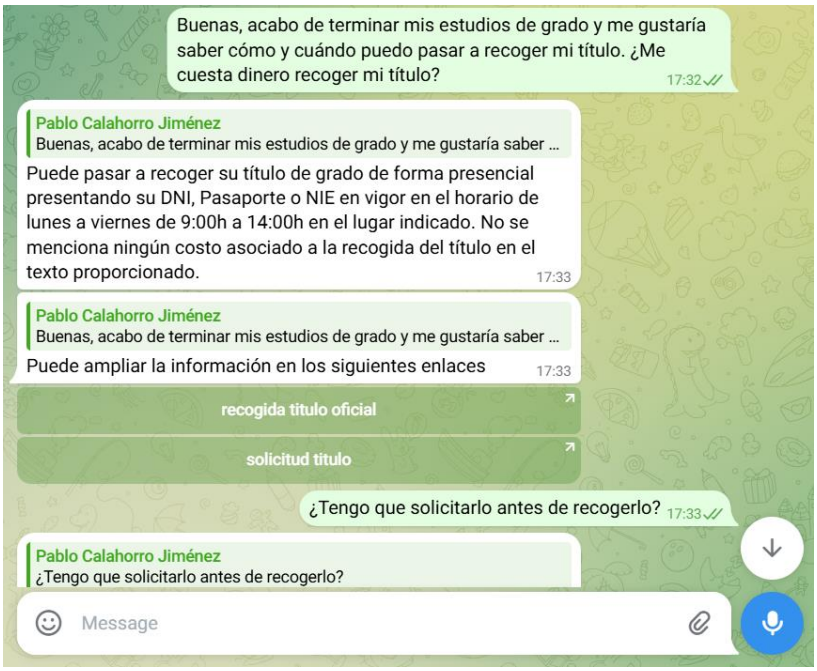
Prueba 4. Un estudiante extranjero quiere hacer uso del gimnasio



Como vemos en la imagen anterior, el sistema contesta en el idioma en el que se le hace la pregunta, funcionalidad que se ha conseguido a través de un adecuado manejo de las **prompt templates.**

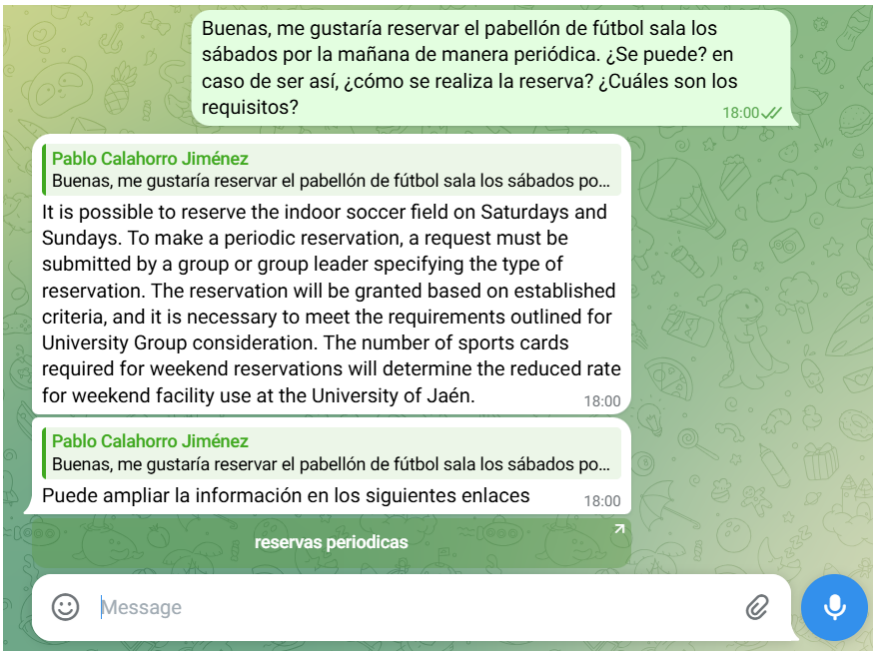
Hay otra línea que se estaba trabajando intentando entrar a la configuración de idioma del usuario para adaptar el la prompt del chat engine en el momento en el que se abre el chat, sin necesidad de que el usuario envíe mensaje alguno, pero telebot es una librería bastante complicada de manejar a ese nivel.

Prueba 5. He terminado la carrera y quiero recoger mi título



La recuperación de fuentes y elaboración de respuestas son perfectas, pero hay un defecto en la información de la web, no se menciona en ninguno de los enlaces nada relacionado con pagar tasas para poder recoger tu título, pregunta que es muy habitual. Posteriormente el sistema contestó que el plazo máximo desde solicitud hasta estar disponible es de 6 meses. Es decir, todo funciona perfecto en base a la información que hay, pero hay un pequeño defecto en la información de la web, no se encuentra el precio de las tasas de recogida.

Prueba 6. Reserva periódica de instalaciones deportivas



Se ha seleccionado esta prueba de funcionamiento para que quede demostrado que **no siempre funciona** el mecanismo de adecuación de idioma consulta – respuesta. Aunque este dato no se ha recogido de forma explícita en ningún recurso del LOG del sistema, con una estadística manual, aproximadamente el 80 % de las veces sí se ajusta el idioma correctamente.

3.2 Evaluación del sistema

A continuación, se presentan los resultados obtenidos tras ejecutar dos scripts que realizan el post procesamiento estadístico sobre las evaluaciones que realizaron algunos de los usuarios tras interactuar con el sistema de recuperación.

3.2.1 Medidas de rendimiento

Los parámetros y las métricas utilizadas para los estudios estadísticos han sido descritas en el punto 2.2. Se va a mostrar una tabla con todos resultados, así como un breve comentario para cada resultado de cada métrica y un histograma de frecuencias para cada métrica.

	Métrica 1	Métrica 2	Métrica 3	Métrica 4	Métrica 5
Media	3.39	3.42	3.91	4.54	4.1
Mediana	4	4	4	5	5
Moda	5	5	5	5	5
Desviación	1.55	1.62	1.35	0.69	1.42
Asimetría	-0.41	-0.41	-1.02	-1.89	-1.27
Curtosis	-1.36	-1.47	-0.25	4.9	0.01
P-valor SW	0	0	0	0	0

Tabla 1. Medidas del rendimiento

- **Métrica 1. rel_query_links**

  i. Media (3.39): La media indica que, en promedio, la métrica evaluada por los usuarios es ligeramente por encima del punto medio de la escala, sugiriendo una valoración más positiva que negativa.

  ii. Mediana (4.0): La mediana, que es superior a la media, indica que al menos el 50% de las evaluaciones son de 4 o más, mostrando una tendencia hacia valoraciones positivas.

  iii. Moda (5): La moda siendo 5 muestra que la evaluación más común es la máxima posible, lo que sugiere que muchos usuarios consideran la métrica muy buena.

  iv. Varianza (2.40) y Desviación Estándar (1.55): Estos valores indican una dispersión moderada en las evaluaciones, lo que implica que las opiniones de los usuarios están distribuidas en un rango amplio alrededor de la media.

  v. Asimetría (-0.41): La asimetría negativa sugiere una ligera inclinación de la distribución hacia la izquierda, es decir, hay una mayor concentración de valores altos.

  vi. Curtosis (-1.36): La curtosis negativa indica que la distribución tiene colas más ligeras y una forma más achatada que la distribución normal, sugiriendo menos valores extremos.

  vii. Prueba de Shapiro-Wilk (Estadístico=0.83, p-valor=1.32e-12): El p-valor extremadamente bajo indica que la distribución de los datos no es normal.

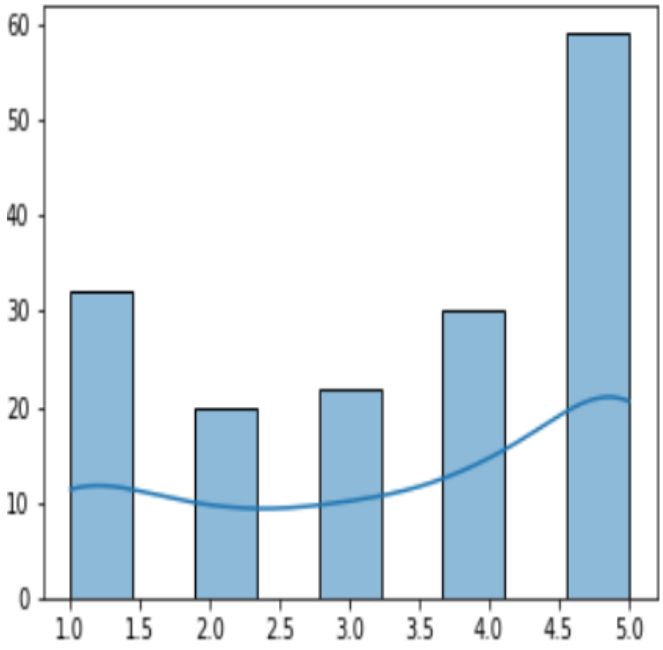


Gráfico 1. Histograma de la métrica 1

- **Métrica 2 rel_query_response**

  i. Media (3.42): La media indica que, en promedio, la métrica evaluada por los usuarios está ligeramente por encima del punto medio de la escala, sugiriendo una valoración generalmente positiva.

  ii. Mediana (4.0): La mediana, que es superior a la media, muestra que al menos el 50% de las evaluaciones son de 4 o más, lo que indica una tendencia hacia valoraciones más altas.

  iii. Moda (5): La moda siendo 5 significa que la evaluación más común es la máxima posible, sugiriendo que muchos usuarios consideran la métrica muy buena.

  iv. Varianza (2.63) y Desviación Estándar (1.62): Estos valores reflejan una dispersión moderada en las evaluaciones, implicando que las opiniones de los usuarios están distribuidas en un rango amplio alrededor de la media.

  v. Asimetría (-0.42): La asimetría negativa sugiere una ligera inclinación de la distribución hacia la izquierda, es decir, hay una mayor concentración de valores altos.

  vi. Curtosis (-1.48): La curtosis negativa indica que la distribución tiene colas más ligeras y una forma más achatada que la distribución normal, sugiriendo una menor presencia de valores extremos.

  vii. Prueba de Shapiro-Wilk (Estadístico=0.80, p-valor=8.74e-14): El p-valor extremadamente bajo indica que la distribución de los datos no es normal.

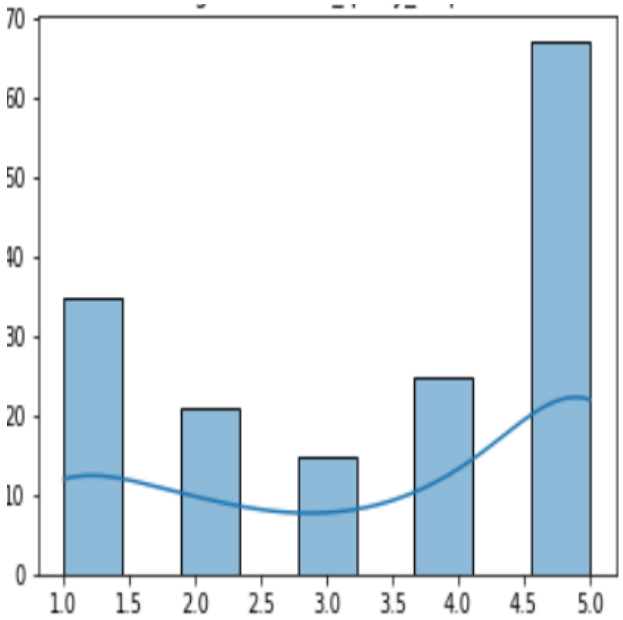


Gráfico 2. Histograma de la métrica 2

- **Métrica 3 context**

  i. Media (3.91): La media indica que, en promedio, la métrica evaluada por los usuarios es bastante alta, sugiriendo una valoración generalmente positiva.
  ii. Mediana (4.0): La mediana, siendo ligeramente superior a la media, muestra que al menos el 50% de las evaluaciones son de 4 o más, lo que indica una tendencia clara hacia valoraciones altas.
  iii. Moda (5): La moda siendo 5 significa que la evaluación más común es la máxima posible, indicando que muchos usuarios consideran la métrica muy buena.
  iv. Varianza (1.83) y Desviación Estándar (1.35): Estos valores reflejan una dispersión moderada en las evaluaciones, lo que implica que las opiniones de los usuarios están distribuidas en un rango considerable alrededor de la media.
  v. Asimetría (-1.02): La asimetría negativa indica una inclinación de la distribución hacia la izquierda, es decir, hay una mayor concentración de valores altos.
  vi. Curtosis (-0.26): La curtosis negativa indica que la distribución tiene colas más ligeras y una forma algo más achatada que la distribución normal, sugiriendo una menor presencia de valores extremos.
  vii. Prueba de Shapiro-Wilk (Estadístico=0.77, p-valor=8.86e-15): El p-valor extremadamente bajo indica que la distribución de los datos no es normal.

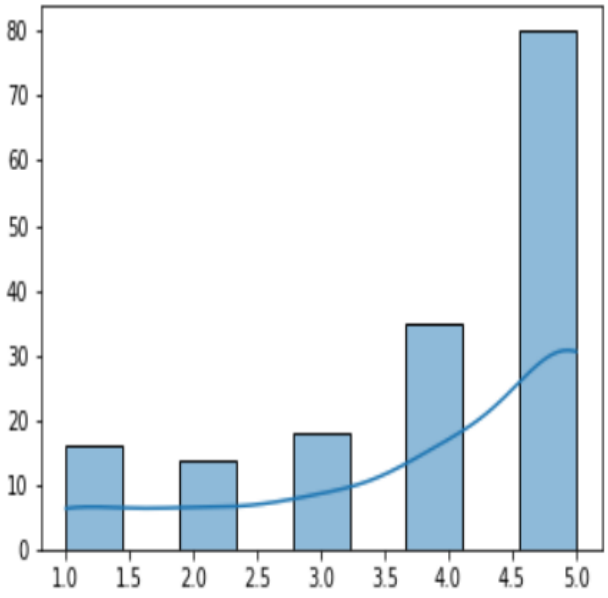


Gráfico 3. Histograma de la métrica 3

- **Métrica 4 time**

  i. Media (4.55): La media muy alta sugiere que, en promedio, la métrica evaluada por los usuarios es excelente.
  ii. Mediana (5.0): La mediana muestra que al menos el 50% de las evaluaciones son la puntuación máxima, lo que indica una valoración muy positiva.
  iii. Moda (5): La moda siendo 5 indica que la evaluación más común es la máxima posible, reforzando la tendencia de evaluaciones muy favorables.
  iv. Varianza (0.48) y Desviación Estándar (0.70): Estos valores reflejan una dispersión baja en las evaluaciones, implicando que las opiniones de los usuarios están bastante concentradas alrededor de la media.
  v. Asimetría (-1.89): La asimetría negativa significativa indica una fuerte inclinación de la distribución hacia la izquierda, es decir, una mayor concentración de valores altos.
  vi. Curtosis (4.90): La curtosis positiva elevada sugiere que la distribución tiene colas más pesadas y un pico más alto que una distribución normal, indicando una alta frecuencia de evaluaciones en los valores extremos.
  vii. Prueba de Shapiro-Wilk (Estadístico=0.65, p-valor=5.18e-18): El p-valor extremadamente bajo indica que la distribución de los datos no es normal.

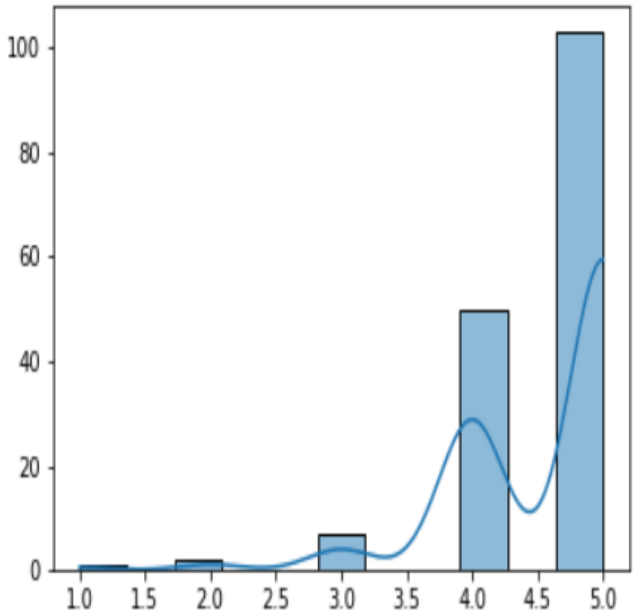


Gráfico 4. Histograma de la métrica 4

- **Métrica 5 correction**

  i. Media (4.10): La media sugiere que, en promedio, la métrica evaluada por los usuarios es bastante alta, indicando una valoración generalmente positiva.
  ii. Mediana (5.0): La mediana, siendo igual a la moda, muestra que al menos el 50% de las evaluaciones son la puntuación máxima, lo que indica una tendencia clara hacia valoraciones muy altas.
  iii. Moda (5): La moda siendo 5 indica que la evaluación más común es la máxima posible, lo cual refuerza la percepción de que muchos usuarios consideran la métrica excelente.
  iv. Varianza (2.02) y Desviación Estándar (1.42): Estos valores indican una dispersión moderada en las evaluaciones, lo que implica que las opiniones de los usuarios están distribuidas en un rango amplio alrededor de la media.
  v. Asimetría (-1.27): La asimetría negativa indica una inclinación de la distribución hacia la izquierda, sugiriendo una concentración de valores altos.
  vi. Curtosis (0.01): La curtosis cercana a cero sugiere que la distribución tiene colas leves y una forma cercana a la distribución normal, indicando una distribución de valores más uniforme alrededor de la media.
  vii. Prueba de Shapiro-Wilk (Estadístico=0.65, p-valor=4.40e-18): El p-valor extremadamente bajo indica que la distribución de los datos no es normal.

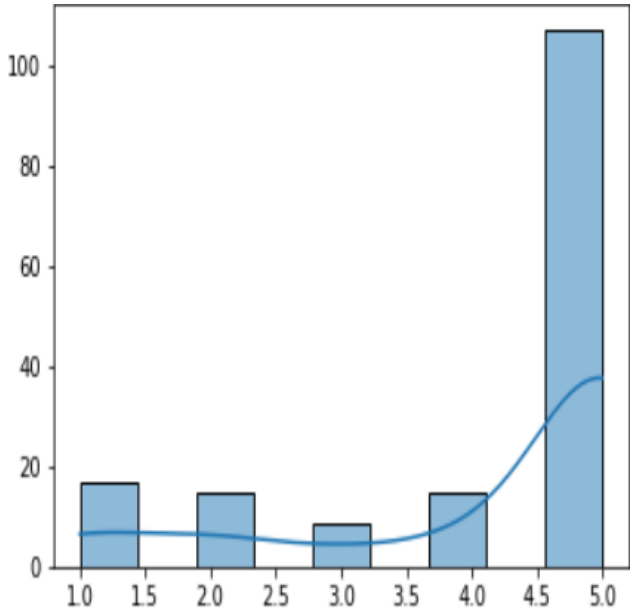


Gráfico 5. Histograma de la métrica 5

3.2.2 Mapa de correlaciones

A continuación, se muestra un mapa de correlaciones con los parámetros y las métricas.

Un mapa de correlaciones, o heatmap de correlación, es una representación visual de la matriz de correlación entre múltiples variables.

¿Para qué sirve?

- Identificación de Relaciones: Ayuda a identificar relaciones lineales entre variables. Si dos variables están correlacionadas, el cambio en una podría estar asociado con el cambio en la otra.
- Selección de Variables: En análisis de datos y modelos predictivos, ayuda a seleccionar variables relevantes y evitar multicolinealidad (cuando dos o más variables independientes en un modelo están altamente correlacionadas).
- Exploración de Datos: Es una herramienta exploratoria para entender mejor los datos antes de aplicar modelos más complejos.
- Detección de Patrones: Facilita la detección de patrones que pueden no ser evidentes con una simple inspección de los datos.

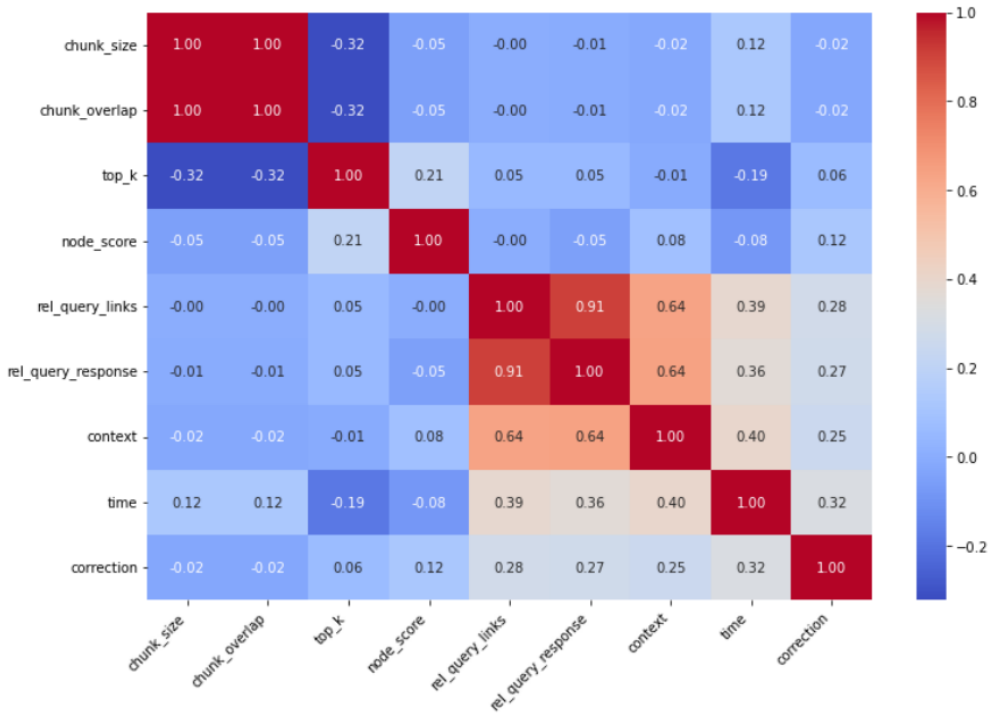


Gráfico 6. Mapa de correlaciones 1

¿Cómo se interpretan estos números?

- **1**: Correlación positiva perfecta. Aumentar en una variable siempre se asocia con un aumento proporcional en la otra.
- **-1**: Correlación negativa perfecta. Aumentar en una variable siempre se asocia con una disminución proporcional en la otra.
- **0**: No hay correlación. Los cambios en una variable no están asociados con cambios en la otra.

Podemos concluir del mapa anterior, aunque en el punto 4 de este documento se detallará más, que el proceso de validación con usuarios finales ha sido un fracaso, dado que manualmente se han extraído fuertes relaciones entre parámetros y métricas hasta perfeccionar el funcionamiento del sistema mediante ensayo-error, pero los datos procesados no avalan dicha hipótesis porque no se ha evaluado al sistema con coherencia por parte de los participantes que han ido sumándose al proceso de validación.

El fracaso del proceso de validación, se puede apreciar en las esquinas superior derecha e inferior izquierda del mapa anterior, dado que los números que deberían estar por encima de o cerca de 0’5, no se alejan a penas de 0, lo que revela nula correlación lineal entre parámetros y métricas.

Esto no se corresponde con la evolución de funcionamiento del sistema, que era lo que se pretendía ilustrar a través del mapa de correlaciones.

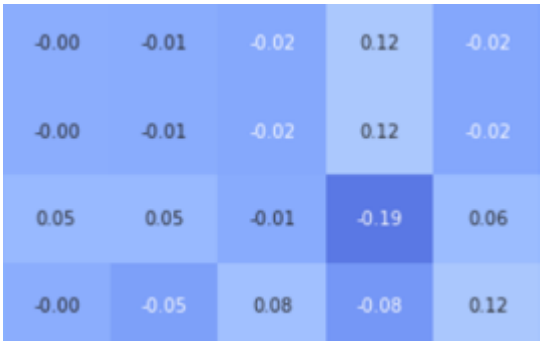


Gráfico 7. Mapa de correlaciones 2

3.2.3 Conclusión y subsistema estadístico

Podemos concluir que las métricas están bien valoradas por los usuarios en general, pero se ha fracasado en el intento de establecer correlaciones entre el valor de los parámetros de funcionamiento y las métricas de medida de rendimiento

A continuación, se expone brevemente, a través de un diagrama de flujo, el subsistema estadístico que ha procesado los datos recogidos. En el script se han utilizado las librerías descritas en el punto 1.8.4

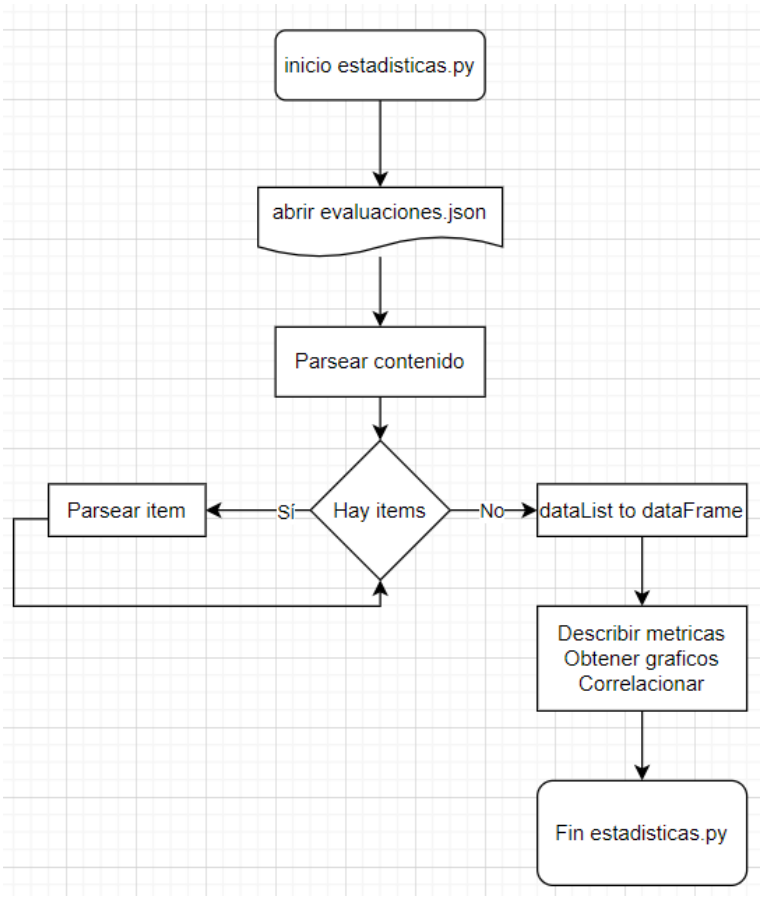


Diagrama 14. Diagrama de flujo de estadísticas

4. Conclusiones

Durante este capítulo, se van a exponer las conclusiones del trabajo realizado divididas en 4 ejes principales. Potencial, escalabilidad, experimentación comparación con ADA y una reflexión final.

4.1 Potencial

Es habitual escuchar que las instituciones públicas no atienden adecuadamente al ciudadano de a pie, se pueden agrupar las reclamaciones en dos grandes grupos

1. Falta de personal
2. Laberintos burocráticos

Se pueden atacar ambos problemas de un golpe gracias a sistemas parecidos al que se ha implementado en este trabajo de fin de grado. Dado que:

1. La capacidad de modelos de lenguaje como GPT 4 para “traducir” el lenguaje administrativo a lenguaje cotidiano es extraordinariamente prometedora.
2. La capacidad para atender a miles de usuarios con una inversión de recursos pequeña ha quedado más que probada. Como nota técnica final, de cara a un futuro despliegue repositorio-servidor, **el peso de la aplicación en local ronda los 3’5 GBytes**, código, datos y post procesamiento.

Por lo tanto, los trabajadores de la administración pública se pueden dedicar a

1. Solventar las casuísticas más delicadas que se den en cada área de gestión del estado, dejando las triviales para el sistema bot experto.
2. Revisar y retroalimentar los sistemas RAG de atención al ciudadano para garantizar que la información de las bases de conocimiento de la administración, esté siempre adecuadamente actualizadas y completas.

Se puede afirmar que la inteligencia artificial, siempre correctamente supervisada y validada, va a ser de gran ayuda en ámbitos en los que no se busca el beneficio, más bien la calidad del servicio y el correcto funcionamiento de una institución.

4.2 Escalabilidad

Dada la dimensión que debe tener un trabajo de fin de grado, se ha decidido no desarrollar al completo todos los aspectos del sistema. A continuación, se van a enumerar algunos de las propuestas más interesantes:

1. Interfaz para el cliente con autentificación para administradores.
2. Limpieza de estructuras de datos a través de hilos para reiniciar el contexto de cada chat de cada usuario activo, sin tener que evaluar la interacción con el sistema o resetear la interacción como ocurre actualmente.
3. Estudio estadístico temporal para evaluar cómo evolucionan las métricas y los parámetros en el tiempo, de esta forma evaluar crecimientos y decrecimientos de la satisfacción según varían los parámetros.
4. Agrupar datos de las evaluaciones por ID de usuario, de forma que se puedan generar muestras no pareadas.
5. Reseteo de parámetros del sistema automático cada cierto intervalo de tiempo buscando acercarse a los valores más óptimos según revele la serie histórica de correlaciones parámetros-métricas. Es decir, que el sistema aprenda de sí mismo a medida que los usuarios participen en la evaluación del sistema.
6. Diseñar un experimento dilatado en el tiempo, con usuarios reales y estadísticamente serio, si es necesario pagando a una muestra heterogénea y representativa de estudiantes, para obtener unos datos estadísticamente coherentes y consistentes.
7. Pasar el proyecto, actualmente en fase de desarrollo, a una versión de producción instalable en un servidor y robusta ante errores, alojada en un repositorio, dado que todos los intentos que se han hecho de hacer funcionar el sistema en un servidor 24/7 han terminado en una caída del sistema o en problemas con las dependencias al gestionar con pip en lugar de con conda. En este punto es donde más he notado mi falta de experiencia en todo lo relacionado con la elaboración de un proyecto software.

4.3 Experimentación

Conclusiones sobre datos de estadísticas

1. Como se ha expuesto brevemente en el apartado 3.2 y en el 4.2, los datos extraídos de la validación de usuarios durante uso real del sistema, **carecen de consistencia dado que dicha participación ha resultado algo forzada**. Si no se realiza una recogida de datos bien diseñada y estructurada no se va a conseguir desarrollar un estudio estadístico en el que se extraigan correlaciones significativas, que era uno de los objetivos de este proyecto. Esta tarea daría para todo un trabajo de fin de grado por sí misma.
2. Sin embargo, la satisfacción de los usuarios sobre medidas de rendimiento es, en general y según los datos obtenidos, asimétrica a la izquierda y con modas y medianas cercanas al máximo, de lo que se deduce **que la satisfacción de los usuarios con el funcionamiento del sistema es alta**.
3. Se debería también, diseñar un experimento en el que sean los mayores conocedores del servicio quienes evalúen la calidad de las fuentes de información y las respuestas, es decir, deberían ayudar a diseñar y ejecutar un experimento los propios trabajadores de los distintos servicios con los que se ha trabajado.
4. En resumen, si se **desplegasen un trabajo experimental hacia usuario y otro hacia cliente** bien diseñados y con un equipo multidisciplinar de desarrollo y ciencia de datos, se obtendría una base científica para escalar el proyecto a cualquier otro ámbito de gestión y administración pública de forma que cada implantación ayudaría a implantaciones posteriores.

Conclusiones sobre el comportamiento del sistema

1. Se comenzó trabajando con muchos parámetros que venían definidos por defecto lo que supuso un comienzo algo tortuoso en cuanto a resultados
2. A medida que se realizaban pruebas, se iba comprendiendo el impacto de determinados parámetros en la recuperación de información, y se realizaron experimentos de ensayo y error que iban arrojando luz.
3. Se fueron fijando tamaños de fragmento y superposiciones según el tamaño medio de los archivos, además de fijar un límite de relevancia y un límite en el número de fuentes relevantes, con el fin de evitar un bombardeo de enlaces al usuario.
4. El manejo rudimentario de las prompt templates también supuso una notable mejora referente al idioma en el que se contestaba a las consultas

5. **Estas decisiones impactaron en el comportamiento del sistema muy positivamente, aunque queda mucho por explorar y mejorar, como despliegue, correcto manejo de prompt templates o ajuste de parámetros del sistema más óptimo.**

4.4 UJAGPT vs ADA

Como quedó patente en el punto 1.5, las capturas a la interacción con ADA cuando está ampliada la interfaz, son de calidad muy deficiente para incorporarlas al documento, a diferencia de lo que ocurre con UJAGPT lo que se aprecia en el punto 3.5. Las capturas de telegram se visualizan en el documento con mucha calidad. Utilizando la versión minimizada de ADA es bastante más presentable.

Ronda 1

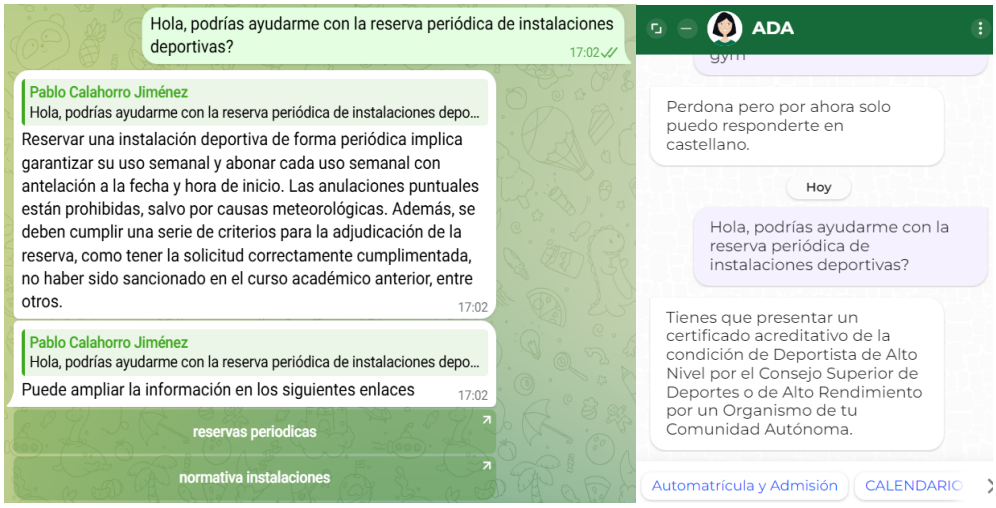


Imagen 15. UJAGPT vs ADA ronda 1

Ronda 2

Imagen 16. UJAGPT vs ADA ronda 2

Ronda 3

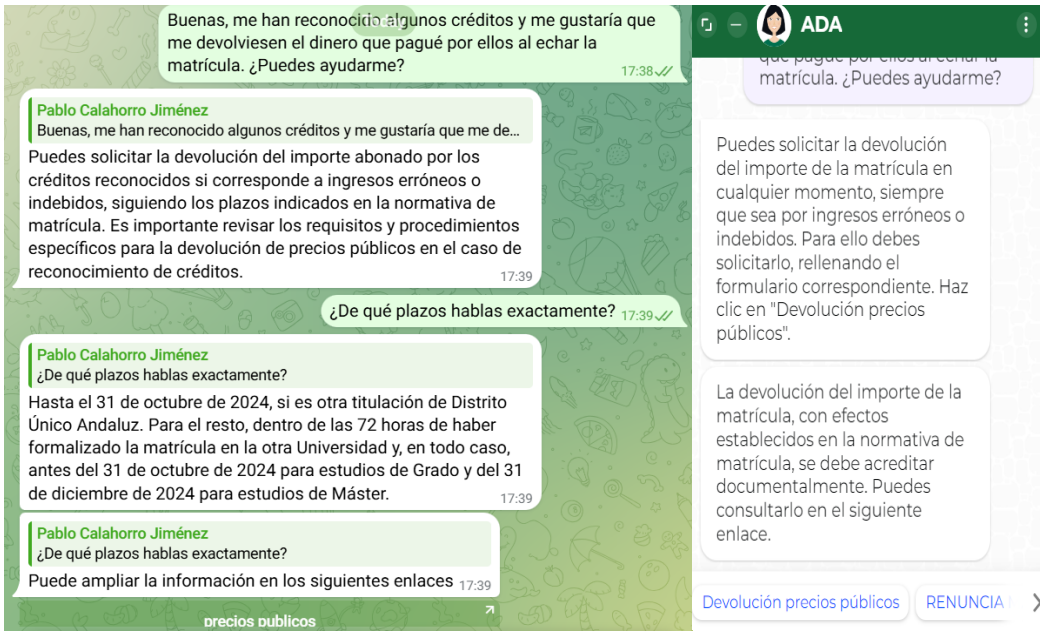


Imagen 17. UJAGPT vs ADA ronda 3

Ronda 4

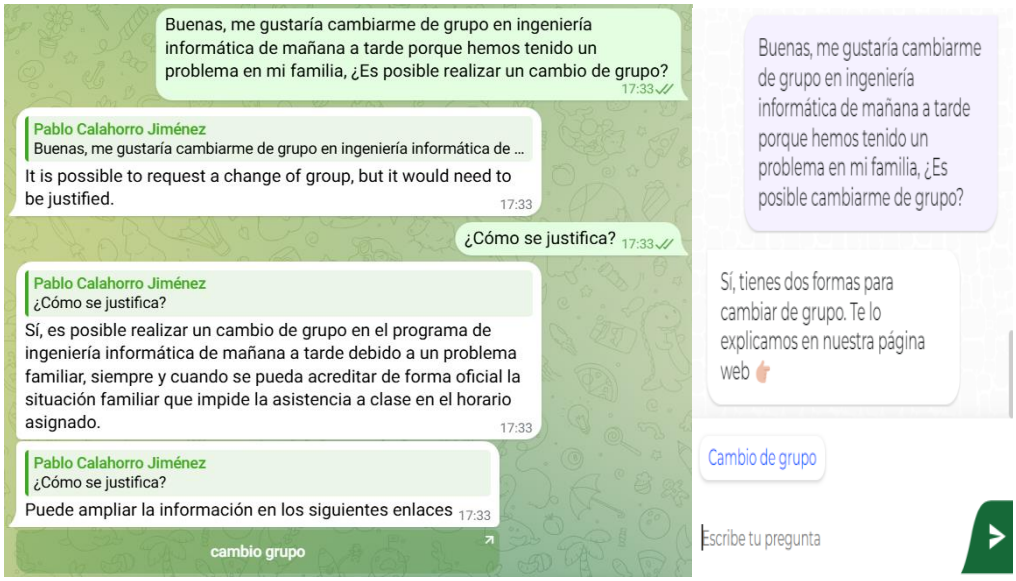


Imagen 18. UJAGPT vs ADA ronda 4

Comentarios sobre la comparación. Se va a analizar fundamentalmente el comportamiento, dado que no se dispone de información a cerca de la implementación de ADA

1. En la ronda 1 queda patente que el **corpus** de documentos de UJAGPT es bastante **más amplio** que el ADA. ADA relaciona “deporte” con lo más similar que tiene cargado, que es el acceso a la universidad para deportistas de alto nivel. En cambio, UJAGPT sí dispone de documentación a cerca de la normativa y uso de instalaciones deportivas. En las rondas posteriores se ha utilizado la base de conocimiento de ADA, que es **acceso y matrícula**, como referencia porque carece de sentido comparar la recuperación en dos sistemas cuando no se utiliza la misma información como referencia.
2. En la ronda 2 se observa lo que puede ocurrir con UJAGPT cuando nunca has chateado con él. Si no tiene mensajes previos, UJAGPT suele escribir el primer mensaje en inglés, lo que es más probable, si no encuentra fuentes lo suficientemente relevantes. En cuanto al contenido, esta ronda está bastante igualada, UJAGPT contesta de forma más genérica, aunque no en el idioma de la pregunta, dado que la pregunta es muy ambigua, sin embargo, ADA contesta de forma muy concreta, presuponiendo que el usuario quería dicha información y no adaptando la respuesta a la ambigüedad de la pregunta.
3. En la tercera se aprecia el **potencial conversacional** de UJAGPT, que conforme obtiene información y se le concretan las cuestiones, es muy preciso en la recuperación. En esta pregunta, la recuperación de ADA no tiene nada que envidiar a la de UJAGPT, pero su respuesta es algo más robótica, da menos la sensación de humano, se podría sospechar que ADA no incorpora GPT 4. Hace muy poco se publicó un experimento [2] en el que GPT 4 había superado el test de Turing en más del 50% de las interacciones con usuarios.
4. En la cuarta ronda se confirma lo que se venía comentando sobre el contexto y la relevancia.
5. Sobre el post procesamiento de datos generados en el funcionamiento no se pueden realizar comparaciones dado que no se dispone de información sobre ADA.

	ADA	UJAGPT
Servicio gestión académica	X	X
Servicio de deportes		X
Servicio RR.II.		X
Servicio ayuda al estudiante		X
Sistema internacionalizado		X
LLM Turing superado		X
Contexto conversaciones	X	X

Tabla 2. ADA vs UJAGPT

En resumen, sobre la comparación

- ADA es más robótica
- UJAGPT posee un corpus de documentos más amplio. Información sobre al menos 4 servicios distintos dentro de la universidad. ADA posee información sobre un área de un servicio.
- UJAGPT es capaz de contestar en varios idiomas además de identificar idioma de manera automática
- UJAGPT es dependiente del contexto para ajustar idioma y para optimizar relevancia, suele mostrar más potencial en conversaciones.
- ADA es muy óptima, incluso más que UJAGPT, en recuperar información del área de acceso y matrícula, dentro del servicio de gestión académica, además sólo en castellano.

Para concluir la comparación, si la universidad de Jaén, o la escuela politécnica superior, están pagando por tener activo el sistema ADA, se aconseja la extinción de dicho contrato y su sustitución por el actual sistema UJAGPT.

4.5 Reflexión final

Para concluir este documento, se va a exponer una breve reflexión acerca del papel del ser humano en el buen o mal desempeño de la inteligencia artificial, más concretamente, en sistemas equivalentes al que se ha implementado. Se estructura la reflexión en 5 puntos, con los que se pretende trascender el aspecto técnico y académico del documento:

1. La inteligencia artificial no tiene conciencia de sí misma, en este tipo de sistemas, como se expuso en el punto 1.4.3, se ejecutan mecanismos **estocásticos [1]**. Por lo tanto, el buen funcionamiento alejado de la aleatoriedad depende directamente del correcto diseño y despliegue por parte de un equipo humano con formación específica en la materia.
2. Los sistemas de recuperación encarnan un **reto** del que, en mi opinión, no somos conscientes. Más que un reto, encarna un **enorme riesgo** en materia energética y geológica [4], dado que no podemos desplegar capacidades de cómputo cada vez mayores, tendiendo al infinito, en un planeta que es finito. Los sistemas **tipo bot experto** ofrecen muchísimas ventajas sobre los sistemas de recuperación de propósito general, pero una de ellas es, sin duda, el menor consumo y desperdicio de recursos.
3. Otro reto, es el de fiscalizar **qué información** permitimos que forme parte de los corpus de este tipo de sistemas.
    a. Imaginemos que existen **agencias** de noticias que acumulan **condenas** por **publicar información falsa**.
    b. Imaginemos que existen **científicos, ingenieros y académicos** respaldados por el **lobby** de la industria petrolera, o por el lobby de la nuclear, o el lobby de las renovables.
4. Ahora, imaginemos que la anterior agencia de noticias, o que los autores “respaldados” por un lobby, terminan con su información falsa o sesgada, en artículos, estudios o libros, y estos documentos a su vez, incorporados en el corpus de sistemas de recuperación de información, e imaginemos que estos sistemas son ampliamente utilizados por parte de la población.
5. Pongámonos en el lugar de un ciudadano, un **usuario, que no conoce ningún aspecto técnico** tras este tipo de sistemas. Que incluso llega a pensar que es técnicamente imposible que estos sistemas mientan y sesguen por interés, porque ¿cómo íbamos a permitir eso los encargados de implementar y desplegar estos sistemas? Este reto también se puede afrontar con **el enfoque de bot expertos** sobre los RAG de propósito general, dado que es más fácil fiscalizar un corpus pequeño.

Bibliografía

[1] Emily M. Bender, Timnit Gebru 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? [Archivo PDF]. https://dl.acm.org/doi/pdf/10.1145/3442188.3445922

[2] Cameron R. Jones, Benjamin K. Bergen 2024. People cannot distinguish GPT-4 from a human in a Turing test [Archivo PDF]. https://arxiv.org/pdf/2405.08007

[3] Roger S. Pressman 2014. Software Engineering a practitioner's approach. McGraw Hill

[4] Agencia internacional de la energía. Electricity 2024 – Analysis and forecast to 2026. https://iea.blob.core.windows.net/assets/6b2fd954-2017-408e-bf08-952fdd62118a/Electricity2024-Analysisandforecastto2026.pdf

[5] Llama-index. High level concepts. https://docs.llamaindex.ai/en/stable/getting_started/concepts/ recuperado el 05 / 02 / 2024

[6] Building Performant RAG Applications for Production. https://docs.llamaindex.ai/en/stable/optimizing/production_rag/ recuperado el 05 / 02 / 2024

[7] Chroma. Getting Started. https://docs.trychroma.com/getting-started recuperado el 06 / 02 / 2024