Universidad ORT Uruguay
Facultad de Ingeniería

Chat Your Data

Chatbot de integración de datos con Inteligencia Artificial

Entregado como requisito para la obtención del título de Licenciatura en Sistemas.

Noa Brenner – 231045

Paula Hernández - 201886

Ignacio Vallarino - 203645

Matías Wilner – 231073

Tutor: Eduardo Mangarelli

2023

Declaración de autoría

Nosotros, Noa Brenner, Paula Hernández, Ignacio Vallarino y Matías Wilner, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

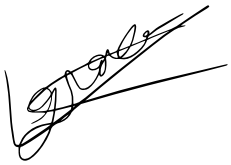
- La obra fue producida en su totalidad mientras realizábamos el proyecto final de la carrera: Licenciatura en sistemas;
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Noa Brenner

19 de Octubre de 2023

Paula Hernández

19 de Octubre de 2023



Ignacio Vallarino

19 de Octubre de 2023

Matías Wilner

19 de Octubre de 2023

Agradecimientos

En primer lugar, deseamos expresar nuestro agradecimiento a nuestro tutor, Eduardo Mangarelli, por su valiosa compañía y orientación a lo largo de este proceso. También queremos extender nuestros agradecimientos a la Universidad ORT Uruguay por el apoyo constante que nos brindaron durante nuestra formación, así como a nuestros profesores, personal de bedelía y otros funcionarios que contribuyeron a nuestra educación. En particular, queremos agradecer al CIE por impulsarnos con la idea, a los revisores y a todas las personas dispuestas a ayudarnos en este proyecto: Martín Solari, Gastón Mousques, Álvaro Ortas, Ximena Scasso y Enrique Topolansky. Su colaboración fue fundamental para validar diferentes aspectos del proyecto, realizar correcciones y recibir valiosas recomendaciones.

A su vez, queremos agradecerle al Banco Interamericano de Desarrollo (BID), principalmente a Alan Kind, quien, desde los primeros días en que Chat Your Data era tan solo una idea, confió en nuestro equipo. Su apoyo y confianza incondicional fueron de gran ayuda para llevar a cabo este proyecto. Además, deseamos agradecer a todas las personas y empresas interesadas en el proyecto por sus opiniones y aportaciones cruciales, entre ellas, Pablo Darscht, Sylvia Chebi, Ernesto Krawchik y más.

Por último, pero no menos importante, queremos agradecer a nuestros seres más cercanos, familias y amigos por el apoyo, la motivación y la paciencia que nos brindaron a lo largo de este viaje.

A todos ellos, muchas gracias.

Abstract

En un mundo cada vez más digitalizado, la gestión de información se convirtió en un desafío crítico para las organizaciones. La acumulación de datos ha llevado a una creciente complejidad en su organización y acceso. El proyecto "Chat Your Data" busca revolucionar la forma en que las organizaciones gestionan y generan su información.

Este proyecto se desarrolló en un contexto donde la inteligencia artificial (IA) está transformando industrias enteras. Chat Your Data se propuso como una plataforma que permite a los usuarios interactuar conversacionalmente con la información de la organización, haciendo preguntas y recibiendo respuestas instantáneas, como si tuvieran un asistente virtual personal.

La primera etapa del proyecto implicó una investigación exhaustiva sobre el problema en cuestión y la inteligencia artificial generativa. Durante este período de exploración, se adquirió una comprensión profunda del potencial y los desafíos de la IA en la gestión de información. Para gestionar esta etapa, se adoptó el marco de trabajo Kanban, que facilitó la organización y priorización de tareas en un entorno de rápido cambio.

La segunda etapa se centró en el desarrollo de un Producto Mínimo Viable (MVP) que representara la visión de Chat Your Data. En esta instancia se utilizó una adaptación del marco Scrum para guiar el proceso de desarrollo. El resultado fue una arquitectura de microservicios robusta y extensible que permitió una comunicación eficiente entre los componentes del sistema, utilizando tecnologías como gRPC y REST API.

La validación de la solución se llevó a cabo en colaboración con el Banco Interamericano de Desarrollo (BID) como *beta tester*, lo que proporcionó valiosos comentarios y permitió adaptar la plataforma a las necesidades reales de una organización de gran tamaño.

Chat Your Data representa un paso hacia el futuro de la gestión de información organizacional, aprovechando la IA para cambiar la forma en que se interactúa con los datos. Este proyecto busca adoptar la innovación en un mundo impulsado por la información, donde la tecnología seguirá mejorando la eficiencia y la productividad en las organizaciones.

Palabras clave

MVP, *Chatbot*, Inteligencia Artificial Generativa, Machine Learning, NLP, ChatGPT, LLM, *Embedding*, Base de Datos Vectorial, *Bot, Prompt*, *Agent*, *Tool*, Memoria, LangChain, Python, React, Nest, Redis, Firebase, Scrum, Kanban.

Índice

Glosario

MVP (Producto Mínimo Viable): La versión más básica de un producto o servicio que aún cumple con los requisitos mínimos para ser lanzado y utilizado por los usuarios.

***Beta tester*:** Persona o entidad que participa en la fase de pruebas beta de un producto o servicio antes de su lanzamiento oficial al público en general. Prueban el producto y dan retroalimentación sobre su funcionamiento.

Chatbot: Programa informático diseñado para interactuar y conversar con personas, en general utilizando lenguaje natural.

Inteligencia Artificial Generativa: Un subcampo de la inteligencia artificial que se enfoca en la creación de modelos capaces de generar contenido nuevo y original, como texto, imágenes o música, a partir de datos de entrada.

Machine Learning: Un enfoque de la inteligencia artificial que utiliza algoritmos y modelos para permitir a las computadoras aprender y mejorar automáticamente a partir de datos y experiencias previas.

NLP (Procesamiento de Lenguaje Natural): Rama de la inteligencia artificial que se enfoca en permitir que las computadoras comprendan, interpreten y generen lenguaje humano de manera natural.

LLMs (*Large Language Models):* Son modelos de inteligencia artificial entrenados para comprender y generar lenguaje humano, como GPT-3.

GPT (*Generative Pre-trained Transformer*): Un modelo de inteligencia artificial que genera texto coherente y contextual basado en el contexto proporcionado, utilizado en aplicaciones de procesamiento de lenguaje natural (*NLPs*).

ChatGPT: Un modelo de lenguaje desarrollado por OpenAI que usa la tecnología *GPT* para generar respuestas de texto en conversaciones de *chat*.

Token en el área de NLP: Un *token* en el Procesamiento de Lenguaje Natural es una unidad individual de texto, que puede ser una palabra o incluso una parte de una palabra.

***Embeddings**:* Representaciones numéricas de datos, como palabras o imágenes, que se usan en algoritmos de aprendizaje automático para capturar características y relaciones entre ellos.

Base de Datos Vectoriales: Una base de datos que almacena información en forma de vectores numéricos, lo que facilita el procesamiento y análisis de datos complejos.

Similarity Search (búsqueda de similitud): Mecanismo que busca elementos similares en una base de datos en función de su similitud con un elemento de consulta dado.

Bot: Un término abreviado para "robot de *software*" o "agente de *software*", que se refiere a un programa informático que puede realizar tareas automatizadas en nombre de los usuarios.

Prompt: Una instrucción o pregunta proporcionada a un modelo de lenguaje como entrada para solicitar una respuesta específica.

Prompt Engineering*:* (La práctica de diseñar de forma efectiva las preguntas o instrucciones que se hacen a modelos de lenguaje para obtener respuestas deseadas.

Agent: Un agente es un programa o sistema informático diseñado para percibir su entorno, tomar decisiones y realizar acciones para lograr un objetivo o conjunto de objetivos específicos.

Tool: Una herramienta o software utilizado para realizar tareas específicas o facilitar procesos en computación o trabajo.

React: Biblioteca de JavaScript para construir interfaces de usuario interactivas y dinámicas en aplicaciones web.

LangChain: Framework diseñado para simplificar la creación de aplicaciones utilizando *Large Language Models (LLM),* permite a los desarrolladores combinar modelos *LLM* como *GPT-4* con recursos externos de datos.

Python: Un lenguaje de programación de alto nivel ampliamente utilizado que se caracteriza por su simplicidad y legibilidad.

NestJS: Un marco de desarrollo para la creación de aplicaciones web en *Node.js.*

Redis: Un sistema de almacenamiento en caché de datos en memoria que se utiliza para acelerar la recuperación de datos en aplicaciones web y bases de datos.

Firebase: Una plataforma de desarrollo de aplicaciones móviles y web que proporciona herramientas y servicios para el desarrollo rápido de aplicaciones.

SCRUM: Un marco de trabajo ágil utilizado en la gestión de proyectos que se centra en la colaboración y la adaptabilidad.

KANBAN: Un sistema de gestión visual utilizado para mejorar la eficiencia en la producción y la gestión de proyectos.

Firestore: Una base de datos en tiempo real y en la nube proporcionada por *Firebase*.

Firebase Auth: Un servicio de *Firebase* que ofrece autenticación de usuarios y funciones relacionadas con la seguridad en aplicaciones web y móviles.

1. Introducción

El documento a continuación tiene como objetivo describir el proyecto de fin de carrera Chat Your Data, realizado como requisito para obtener el título de Licenciado/a en Sistemas de la Universidad ORT Uruguay. El proyecto fue desarrollado por Noa Brenner, Paula Hernández, Ignacio Vallarino y Matías Wilner, y la tutoría estuvo a cargo de Eduardo Mangarelli.

1.1. Surgimiento de Chat Your Data

En febrero de 2023, Noa Brenner, Paula Hernández, Ignacio Vallarino y Matías Wilner decidieron comenzar con su proyecto final de carrera. El equipo estaba motivado para llevar a cabo un proyecto propio que fuera desafiante, motivador y que abordara un problema real, además de contar con clientes interesados y cumplir con los requisitos de aprobación.

A partir de experiencias laborales y conversaciones cercanas, se percataron que todos enfrentaban dificultades para acceder y comprender la información disponible. Este desafío era compartido, ya que incluso las organizaciones o empresas lidiaban con la falta de accesibilidad a datos cruciales por parte de sus empleados. A medida que el equipo interactuó con diferentes actores interesados, se hizo evidente que esta era una necesidad. Paralelamente, la tecnología de ChatGPT y la inteligencia artificial generativa alcanzaban un punto culminante de desarrollo. Fue en este contexto que surgió la idea de Chat Your Data.

1.2. Objetivos del Proyecto

Chat Your Data tiene como objetivo desarrollar un MVP (Producto Mínimo Viable) de una plataforma de *chatbot* con inteligencia artificial, para el manejo de datos organizacionales. El foco principal del proyecto es brindar una solución conversacional, en donde los usuarios pueden interactuar, en lenguaje natural, con los datos de su organización facilitando el acceso a la información, generación de contenido, la creación de *insights* y la toma de decisiones basadas en sus grandes cantidades de datos.

A continuación, se detallan los objetivos globales del proyecto, como también los objetivos académicos y los objetivos del producto.

1.2.1. Proyecto

1.2.1.1. Desarrollar una plataforma de *chatbot* con IA

El principal objetivo del proyecto es desarrollar un MVP de una plataforma de *chatbot* que utiliza técnicas de IA para permitir al usuario tener conversaciones basadas en los datos organizacionales. El *chatbot* será capaz de comprender preguntas complejas sobre los datos y generar respuestas relevantes y precisas, así como generar contenido en función de las consultas realizadas.

1.2.1.2. Integración de tecnologías avanzadas

Aprovechar las tecnologías disponibles en el campo de la inteligencia artificial, como LLMs, *frameworks*, procesamiento de texto en lenguaje natural y técnicas de análisis de datos. Estas tecnologías permiten mejorar la interpretación y la respuesta a las consultas de los usuarios.

1.2.1.3. Adaptabilidad y extensibilidad

Desarrollar una arquitectura flexible y extensible, que facilite la incorporación de nuevas funcionalidades, interfaces de usuario y herramientas de inteligencia artificial, buscando que el sistema pueda adaptarse fácilmente a las necesidades cambiantes de los usuarios.

1.2.2. Producto

1.2.2.1. Mejora de la experiencia del usuario

Proporcionar una experiencia de usuario mejorada en las empresas u organizaciones al permitir un fácil y rápido acceso a la información organizacional mediante interacciones conversacionales intuitivas y eficientes. Los usuarios podrán obtener respuestas rápidas y precisas a sus consultas, lo que facilitará su forma de acceder a los datos.

1.2.2.2. Generar valor para la organización

Se busca agregar valor a la organización, al proporcionar funcionalidades que ayuden a crear *insights*, soportar la toma de decisiones basadas en datos, generar contenido y ahorrar tiempo en el acceso a la información organizacional. Esto permitirá a la organización aprovechar al máximo sus datos y el tiempo de sus trabajadores.

1.2.2.3. Generar un producto de nivel profesional

Desarrollar un producto que satisfaga las necesidades de nuestros clientes interesados, y cumpla con las características de usabilidad y eficiencia.

1.2.3. Académicos

1.2.3.1. Aprender nuevas tecnologías

El proyecto brinda la oportunidad de aprender sobre las nuevas tecnologías en el campo de la inteligencia artificial, como modelos de lenguajes, *chatbots*, modelos de aprendizaje automático.

1.2.3.2. Aplicación de los conocimientos previos

Además de aprender nuevas tecnologías, el equipo busca aplicar los conceptos y técnicas que se aprendió a lo largo de la carrera para crear un proyecto sólido. Se utilizaron las herramientas y metodologías de ingeniería de *software* para llegar a resultados óptimos en el desarrollo del proyecto.

1.2.3.3. Conseguir la excelencia académica

Como equipo, se apunta a culminar la carrera académica con un trabajo de excelencia, manteniendo el nivel de rendimiento y dedicación que se demostró hasta ahora.

1.2.4. Equipo

1.2.4.1. Buena relación entre los integrantes

La buena relación entre el equipo es fundamental para poder desarrollar un proyecto de tal magnitud.

1.2.4.2. Crear un emprendimiento

El equipo buscó crear un sistema o producto que tenga el potencial de trascender más allá del proyecto académico.

1.2.4.3. Disfrutar del camino

El equipo valora la importancia de disfrutar del proceso de desarrollo del proyecto y aprovechar al máximo las oportunidades de aprendizaje que se presenten.

1.3. Estructura del Documento

1. **Introducción**: Se presentan los aspectos esenciales del proyecto, incluyendo sus integrantes, origen y objetivos.
2. **El problema y su solución**: Explora en detalle el problema en cuestión y su contexto, así como la solución propuesta.
3. **Justificación del problema y solución**: Se muestra la justificación del problema y su solución, con un análisis del negocio detallado.
4. **Marco metodológico**: Se habla sobre las decisiones clave adoptadas por el equipo, como el ciclo de vida del proyecto, las metodologías utilizadas, los roles y las etapas del proceso, tomadas en base a las características del equipo y proyecto.
5. **Investigación**: Se ofrece una introducción de los conceptos relacionados con la inteligencia artificial, su comprensión y aplicaciones relevantes.
6. **Ingeniería de requerimientos**: Se describe el proceso de la obtención y análisis de requerimientos, las herramientas y técnicas que se utilizaron.
7. **Arquitectura**: Se busca explicar la arquitectura que fue utilizada en el proyecto, justificando las decisiones de diseño tomadas para definir la arquitectura, y las tecnologías utilizadas.
8. **Gestión del proyecto**: Se describe cómo trabajó el equipo durante todo el proyecto, además de explicar la gestión de riesgos y gestión de la comunicación.
9. **Gestión de calidad**: Se definen los objetivos de calidad, y se describe cómo el equipo gestionó la calidad del producto y el proceso.
10. **Gestión de la configuración**: Se mencionan los elementos de configuración de *software* y se explica cómo se trabajó, tanto en la configuración del *software*, como en las herramientas usadas para la documentación.
11. **Conclusiones y lecciones aprendidas**: Se describen las lecciones aprendidas, las conclusiones al terminar el proyecto y cuáles son los próximos pasos.
12. **Referencias bibliográficas**: Se presentan las referencias bibliográficas.
13. **Anexo**: Se incluye información de apoyo al documento.

2. Problema y su solución

2.1. El problema

2.1.1. Identificación del problema

La dificultad de muchas empresas u organizaciones para integrar, analizar y hacer consultas eficientes sobre sus datos es un problema existente. En especial, gestionar grandes cantidades de documentos puede ser un desafío para muchas organizaciones. Los documentos representan una fuente vital de información en cualquier entidad, y su manejo ineficiente puede dar lugar a problemas significativos, especialmente si la información está dispersa en diferentes lugares.

En consecuencia, los trabajadores se encuentran en situaciones en las que se dificulta hallar la información necesaria en el momento preciso, y en ocasiones, incluso puede no encontrarse en absoluto. A esto se suma la complejidad y poca claridad de las interfaces utilizadas para acceder y analizar estos datos, lo que demanda la intervención de personal especializado en tecnología para resolver esta dificultad.

Es importante destacar que el acceso a datos de alta calidad tiene una gran importancia en actividades esenciales como el análisis del negocio, la creación de campañas de *marketing*, la generación de oportunidades comerciales (*leads*) y todos los procesos que una organización lleva a cabo. La falta de información detallada y actualizada dificulta el desarrollo de las operaciones diarias y no logra alcanzar la eficiencia deseada.

2.1.2. Dimensionamiento y relevancia del problema

El manejo ineficiente de documentos puede generar repercusiones negativas en cualquier empresa u organización, manifestándose especialmente en la pérdida de información valiosa y la duplicación innecesaria de esfuerzos para encontrarla y consolidarla.

Se consume más tiempo en la búsqueda de datos en documentos que en la realización efectiva de tareas, desencadenando un efecto perjudicial tanto en la capacidad de toma de decisiones basadas en datos como en la identificación de oportunidades de negocio esenciales debido a la dificultad de localizar y consolidar la información relevante. Todo esto puede hacer que la competitividad y éxito de la organización se vea afectada.

Es importante tener en cuenta que, a medida que la cantidad de datos en documentos crece, la necesidad de una solución más efectiva y eficiente es cada vez más necesaria.

Por todo lo dicho anteriormente, es esencial para cualquier organización encontrar una manera de gestionar y utilizar sus datos de manera más efectiva y eficiente que le permita gestionar mejor su operación, tomar decisiones con mejor calidad y completitud de datos y mantener su productividad de forma de asegurar su éxito a largo plazo.

2.1.3. ¿Quién tiene el problema?

En la actualidad, la presencia de grandes volúmenes de archivos afecta a múltiples departamentos y áreas dentro de las organizaciones y empresas, incluyendo ventas, finanzas, relaciones con clientes, relaciones con proveedores, operaciones y recursos humanos, entre otras. Estos archivos abarcan áreas críticas en todas estas funciones organizacionales, generando un desafío común para diversas áreas de la organización.

La complejidad de integrar, analizar y extraer datos de manera eficiente se extiende a todos los sectores organizacionales. Los grandes volúmenes de archivos impactan significativamente en la funcionalidad y eficiencia de estas áreas en todos los sectores organizacionales. Este problema no se limita a una industria o sector específico, sino que es una preocupación generalizada que afecta a toda la comunidad empresarial.

2.1.4. Consecuencias de la ineficiencia en la gestión de datos

2.1.4.1. Impacto en la eficiencia de la organización

Este problema tiene un gran impacto en la eficiencia de las organizaciones, ya que hace difícil obtener de manera oportuna información necesaria para la toma de decisiones. La pérdida de información valiosa y duplicación innecesaria de esfuerzos para consolidar datos dispersos, tienen como consecuencia una pérdida de tiempo y recursos considerables.

Además, cuando los datos no se gestionan eficazmente, existe un mayor riesgo de cometer errores y tomar decisiones incorrectas. La falta de acceso a datos precisos puede llevar a decisiones basadas en información desactualizada o incompleta.

2.1.4.2. Complejidad de acceso a datos

Las organizaciones cuentan con sistemas de análisis de datos que presentan interfaces complejas que requieren personal especializado. Esto puede crear barreras para que todos los miembros de la organización accedan a los datos de manera sencilla, lo que afecta su productividad y limita su capacidad para aprovechar al máximo sus recursos.

2.1.4.3. Falta de información clara y consolidación de datos

El esfuerzo requerido para consolidar datos y extraer información valiosa puede resultar en la falta de claridad en los datos disponibles. Esto puede llevar a que la organización no atienda de manera proactiva las necesidades o problemas de sus clientes, y no obtenga indicadores de calidad para definir acciones rápidas y eficaces.

2.1.4.4. Disminución de la productividad

La búsqueda manual y la manipulación de datos consumen un tiempo valioso que podría destinarse a tareas más productivas. La ineficiencia en la gestión de datos puede disminuir la productividad general de los empleados y retrasar los proyectos.

2.1.5. Validación del problema

Como respuesta a la problemática identificada, el equipo implementó varias estrategias para validar tanto el problema como el potencial de una solución. El equipo optó por adoptar enfoques variados, abarcando a usuarios afectados por el problema, usuarios potenciales y organizaciones que buscan soluciones de esta índole. Como parte de esta fase de validación, se llevaron a cabo encuestas y entrevistas, dirigidas a individuos de organizaciones de diferentes dimensiones y sectores.

Las encuestas se diseñaron con el propósito de recopilar información detallada acerca de las dificultades que enfrentan en la gestión de volúmenes sustanciales de datos. Es fundamental enfatizar en primer lugar que las respuestas obtenidas de estas encuestas revelaron una tendencia marcada: la mayoría de los empleados encuestados experimentan desafíos significativos en la gestión de sus amplias cantidades de datos.

Continuando con el proceso de validación, el equipo llevo a cabo entrevistas con figuras influyentes en organizaciones de variadas industrias y tamaños, tales como GeneXus, CloudGaia, el Banco Interamericano de Desarrollo (BID), el Palacio Legislativo, entre otras.

Luego de estas conversaciones, el equipo concluyó que la problemática identificada era de hecho una preocupación real en el entorno organizacional, y que se requería una solución inmediata para atenderla de manera efectiva. En el próximo capítulo se detallará a fondo la justificación del problema.

Por otro lado, en el capítulo 6 Ingeniería de requerimientos se encuentra en detalle las conclusiones y evidencias acerca de las distintas etapas de validación e investigación realizada para el problema y solución, así como el proceso iterativo con el *beta tester* para asegurar de llegar a una solución que se ajuste a las necesidades reales.

2.2. Solución

2.2.1. Descripción de la solución

Chat Your Data es una plataforma de *chatbot* con inteligencia artificial, que recupera información relevante de múltiples fuentes y permite a las organizaciones realizar consultas sobre sus datos, obteniendo respuestas instantáneas y generando contenido basado en esos datos. Facilita la integración, análisis y consultas eficientes sobre grandes cantidades de datos organizacionales.

Esta solución innovadora está diseñada para abordar la demanda creciente de las organizaciones en la gestión de grandes volúmenes de datos. A través de su plataforma de *chatbot* con inteligencia artificial, Chat Your Data proporciona a las organizaciones la capacidad de realizar consultas sobre sus datos, obteniendo respuestas instantáneas y generando contenido único en base a la información recopilada. Surge como una respuesta directa a la necesidad de acceder a información relevante y contextualizada de forma ágil y efectiva.

2.2.1.1. Descripción funcional de la solución

A continuación, se muestra una descripción funcional de alto nivel que ofrece una visión general de las funcionalidades principales de Chat Your Data, destacando cómo estas funciones satisfacen las necesidades y perfiles de los usuarios.

Subida, eliminación y vista de archivos: Los usuarios pueden cargar archivos relevantes al sistema, permitiendo la incorporación de datos en el entorno de Chat Your Data. De la misma forma, pueden eliminar estos archivos y abrirlos para ver su contenido.

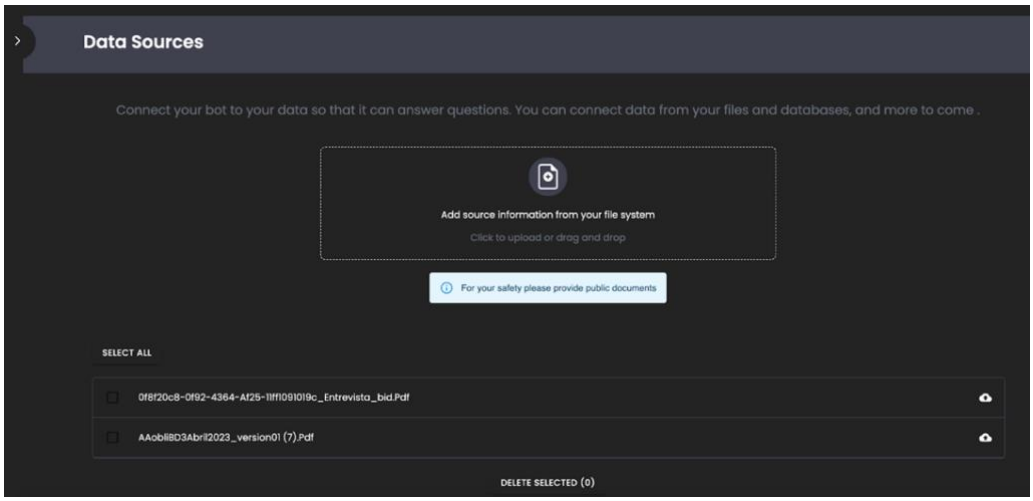


Ilustración 1- Visualización y subida de archivos

Interacción con el *chatbot*: Los usuarios pueden hacer preguntas específicas sobre los archivos que han cargado. El *chatbot* proporciona respuestas rápidas y precisas, facilitando la obtención de información valiosa.

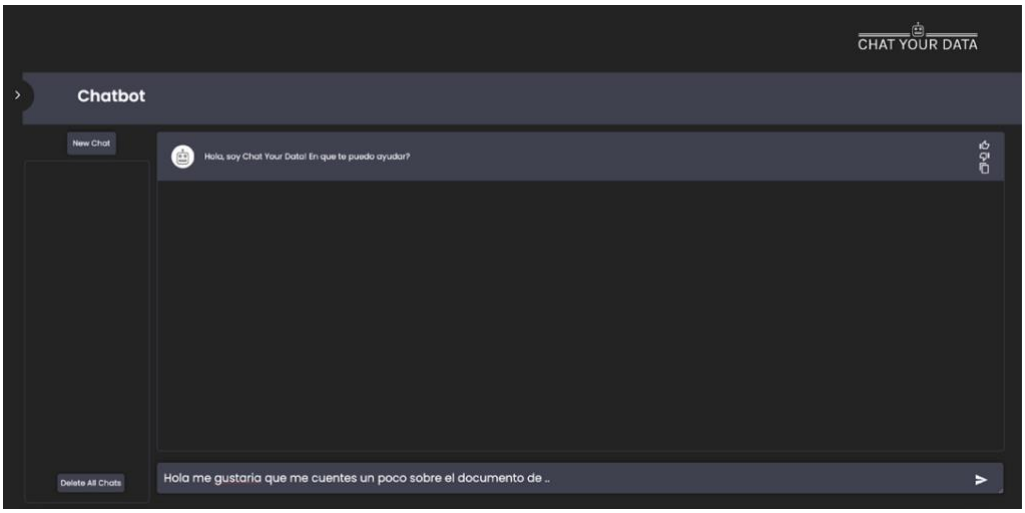


Ilustración 2- Interacción con el *chat*

Ver referencias: Los usuarios pueden visualizar las referencias y fuentes utilizadas en la respuesta que proporciona el *chatbot*, además de hacer *click* y que se abra el documento.

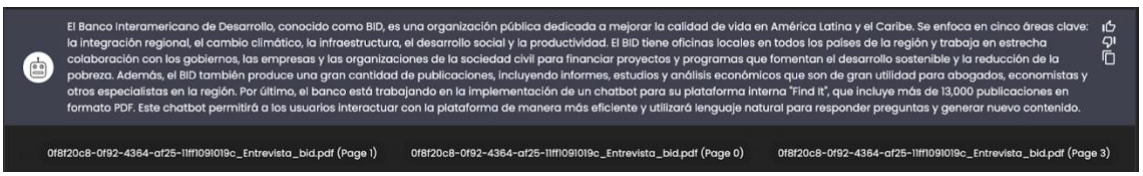


Ilustración 3- Referencias en el *chat*

Acceso a conversaciones anteriores: Se ofrece la posibilidad de acceder a conversaciones previas, lo que permite revisar y retomar discusiones pasadas para una mejor comprensión y seguimiento.

Crear/editar nombres - eliminar conversaciones: Los usuarios pueden crear nuevas conversaciones, personalizar los nombres de las conversaciones y eliminarlas.

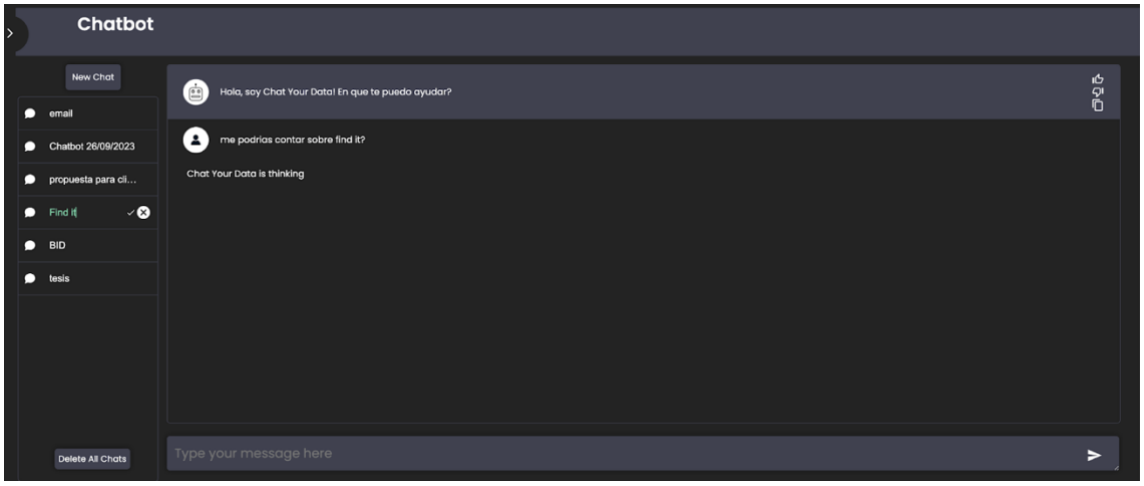


Ilustración 4- Visualización de conversaciones

***Feedback* y calificación de respuestas**: Los usuarios pueden calificar las respuestas del *chatbot* como buenas o malas. Además, tienen la opción de agregar comentarios adicionales para proporcionar retroalimentación detallada. Esto contribuye a mejorar continuamente la calidad de las respuestas.

Función de copiar: La plataforma permite a los usuarios copiar las respuestas generadas por el *chatbot*, facilitando compartir y usar la información dentro y fuera del sistema.

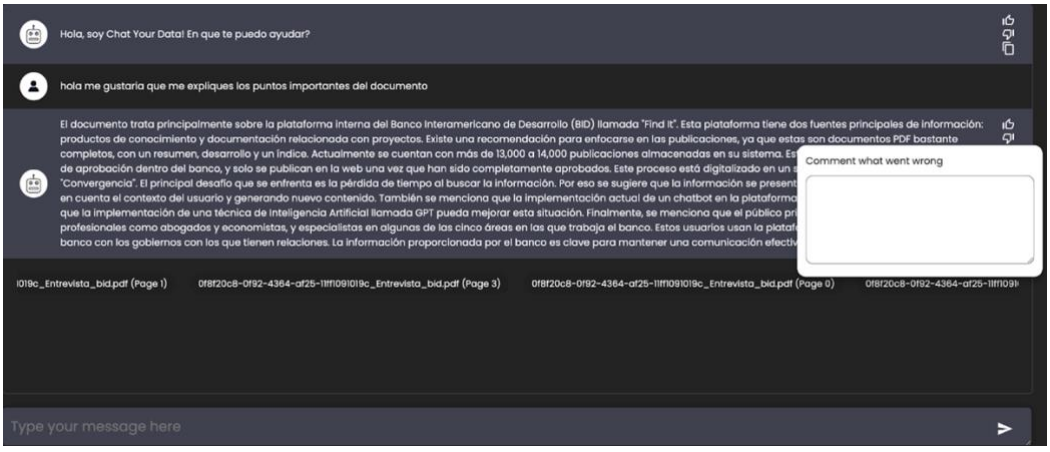


Ilustración 5- Visualización de *feedback* de respuestas

2.2.1.2. Propuesta de valor

Permitir a cualquier empleado de cualquier rol en una organización encontrar la información que necesita a través de simples preguntas o indicaciones en lenguaje natural, de una manera eficiente y contextualizada, mejorando así la toma de decisiones y simplificando la consulta y gestión de datos, además de facilitar la generación de contenido.

1. **Acceso eficiente**: Los empleados pueden acceder a los datos de manera rápida, eficiente y sencilla, lo que maximiza la utilidad de la información en el día a día.
2. **Centralización de datos**: La plataforma centraliza diversas fuentes de datos dispersas, permitiendo la integración de ellos, brindando un acceso único y simplificado el acceso a la información.
3. **Toma de decisiones informadas**: La disponibilidad de datos precisos y actualizados facilita la toma de decisiones fundamentadas y estratégicas en todos los niveles de la organización. Facilita el análisis sobre grandes cantidades de datos.
4. **Generación de contenido único**: La solución permite la creación de contenido único y especializado a partir de los datos, impulsando la personalización y la relevancia en la comunicación.

2.2.1.3. Características destacadas de Chat Your Data

La plataforma recupera información relevante de diversas fuentes y la presenta de manera comprensible y relevante para el usuario, lo que simplifica la búsqueda de datos y agiliza la toma de decisiones. También facilita la integración, el análisis y las consultas de grandes cantidades de datos, lo que mejora la eficiencia en el uso de la información.

1. **Interfaz intuitiva**: Chat Your Data ofrece una interfaz fácil de entender y utilizar para usuarios de todos los niveles. Esto garantiza que la gestión de datos sea accesible y cómoda para cada miembro de la organización.
2. **Procesamiento de lenguaje natural**: La plataforma comprende y responde a preguntas y comandos en lenguaje natural, lo que agiliza la interacción con los datos y promueve una experiencia conversacional.
3. **Tecnologías de IA avanzadas**: Chat Your Data emplea las últimas tecnologías de inteligencia artificial para brindar soluciones de vanguardia, mejorando la eficiencia y precisión en la gestión de datos.

4. **Generación automatizada de contenido**: La solución posibilita la generación automatizada de contenido, como correos electrónicos, informes y análisis, reduciendo el esfuerzo manual y aumentando la eficacia en la comunicación.
5. **Soporte multilingüe**: Chat Your Data se adapta a documentos y consultas en español e inglés, lo que permite una experiencia de usuario más amplia y efectiva.
6. **Variedad de formatos de documentos**: La plataforma acepta documentos de texto en formatos como PDF, en un futuro aceptando otros formatos como CSV y otros comúnmente utilizados en el ámbito organizacional, brindando flexibilidad en la incorporación de datos desde diversas fuentes.

2.2.1.4. Tecnologías claves que hacen posible esta solución

- **Procesamiento de lenguaje natural (NLP)**: El procesamiento de lenguaje natural (NLP) es una tecnología que permite a las computadoras comprender y procesar el lenguaje humano [1]. Chat Your Data utiliza tecnologías avanzadas de procesamiento de lenguaje natural mediante LangChain, que permite que la inteligencia artificial maneje diálogos de manera efectiva. y convierta el lenguaje humano en vectores matemáticos para un análisis más profundo [2].
- **Large Language Models (LLM)**: Son sistemas de inteligencia artificial que fueron entrenados en grandes cantidades de texto y pueden generar respuestas coherentes y contextualizadas en función de las consultas de los usuarios [3]. Chat Your Data aprovecha los modelos de lenguaje avanzados, como GPT-3 y GPT-4 de OpenAI, para brindar respuestas precisas y contextualizadas a las consultas de los usuarios.
- **Base de datos vectorial**: Chat Your Data utiliza una base de datos vectorial. Esta es una estructura de almacenamiento que guarda eficientemente los *embeddings* generados durante el proceso de e*mbedding* de documentos. Esto permite un acceso rápido y eficiente a los datos, dándole agilidad y efectividad a la plataforma [4].

En el capítulo 5 Investigación, se profundiza en detalle acerca de las tecnologías y conceptos clave abordados en el proyecto.

En el próximo capitulo, se hará un análisis sobre la justificación del problema y su solución, el análisis de mercado, y el desarrollo de un plan de negocio.

3. Justificación del problema y solución

3.1. Técnicas y metodologías utilizadas para justificación de solución y problema

3.1.1. Design Thinking

El equipo trabajó con una adaptación de Design Thinking, siendo este un enfoque centrado en el usuario que se utiliza para comprender, idear y resolver problemas [5]. A través de Design Thinking, el equipo abordó tanto la identificación del problema como la propuesta de solución, asegurando que ambas etapas estuvieran en sintonía con las necesidades reales de las organizaciones y sus usuarios.

- **Empatizar**: En esta fase, el equipo se centró en comprender a fondo las necesidades y desafíos de las organizaciones en la gestión de datos. Se llevaron a cabo encuestas y entrevistas con diferentes organizaciones para obtener información directa de los usuarios y entender sus experiencias. Esta etapa permitió al equipo obtener una comprensión profunda de los problemas que enfrentan las organizaciones en relación con la integración, análisis y consulta eficiente de sus datos. Además, se realizó un análisis del entorno, evaluando las tendencias tecnológicas, las fuerzas del mercado y otras influencias externas que podrían impactar en el problema.
- **Definir**: Basándose en los conocimientos adquiridos en la fase de Empatizar, el equipo definió claramente el problema central que buscaban resolver: la dificultad de las organizaciones para acceder y utilizar eficientemente sus datos. Se identificaron los principales desafíos que enfrentan las organizaciones en este proceso.
- **Idear**: En esta etapa, se generaron ideas creativas y soluciones innovadoras para abordar el problema identificado. Se llevaron a cabo sesiones de lluvia de ideas en equipo, charlas con expertos, explorando diferentes enfoques para mejorar la gestión y el acceso a los datos. El objetivo era generar una variedad de conceptos que podrían ser potencialmente implementados en la solución final.
- **Prototipar**: Con la visión más clara, el equipo creó prototipos y *mockups* de la solución propuesta. Estos prototipos permitieron visualizar cómo funcionaría la plataforma y cómo los usuarios interactuarían con ella. Se realizaron pruebas de usabilidad y se recopiló *feedback* para iterar y mejorar los prototipos.

- **Testear**: Se llevaron a cabo pruebas con usuarios reales para evaluar la eficacia y la usabilidad de los prototipos. A través de estas pruebas, el equipo pudo identificar áreas de mejora y realizar ajustes basados en los comentarios de los usuarios.
- **Implementar**: Finalmente, se procedió a la implementación de Chat Your Data como solución basada en las iteraciones y mejoras derivadas de las etapas anteriores. En esta etapa se fueron realizando diferentes validaciones para poder ir evaluando el sistema y realizando mejoras.

A lo largo de este proceso de Design Thinking, el equipo buscó mantener un enfoque centrado en el usuario, buscando entender sus necesidades y diseñar una solución que abordara sus desafíos de forma efectiva y eficiente.

3.1.2. Análisis del Negocio

3.1.2.1. Lean Startup Canvas

El equipo decidió realizar un Lean Startup Canvas para modelar el modelo de negocios por varias razones que se alineaban con las necesidades y objetivos del proyecto. El Lean Startup Canvas proporcionó un enfoque efectivo para abordar los desafíos específicos enfrentados en el proyecto, especialmente en el contexto de desarrollar un MVP dentro de plazos ajustados.

El Lean Canvas es una herramienta útil y de valor principalmente para emprendimientos y *startups*, en donde la velocidad y la adaptabilidad son aspectos clave para el éxito. Este, al enfocarse en la esencia del negocio en vez de una planificación detallada, permite una mayor agilidad y la posibilidad de ajustarse y cambiarse constantemente, a medida que se obtiene *feedback* de los usuarios [6].

A continuación, se explica las principales razones de usar Lean Canvas:

Enfoque en validación temprana

El enfoque en la validación temprana del problema y la solución fue una de las razones principales de la elección del uso del Lean Canvas. Esto fue relevante en el contexto del proyecto final de carrera, donde los plazos eran ajustados, y se requería una validación y desarrollo del proyecto rápida y eficiente.

Además, el Lean Canvas se enfoca en la esencia del negocio en lugar de enfocarse en una planificación detallada. Esto permitió concentrarse en los aspectos fundamentales del proyecto y adaptarse ágilmente a medida que se conocía y se obtenía *feedback* de los usuarios.

Iteración Continua

El Lean Canvas se enfoca en la iteración continua y en el ajuste constante del modelo de negocio en función de los resultados y el *feedback* de los usuarios. Dado que se está desarrollando un MVP, fue fundamental tener la flexibilidad de adaptar el modelo de negocio en función de lo que el equipo aprendiera a lo largo del proceso de desarrollo.

Mayor enfoque en el cliente

El Lean Canvas ayuda a entender profundamente lo que los clientes buscan y necesitan. Esto encajaba perfectamente con el objetivo del equipo, ya que se quería asegurar que el producto final satisficiera las demandas reales de los usuarios y aporte un valor significativo.

Ágil y adaptable

La estructura del Lean Canvas se alinea eficazmente con la naturaleza dinámica y adaptable del desarrollo de software. Esto es importante particularmente en el ámbito de las tecnologías de inteligencia artificial, donde los avances emergentes pueden transformar aspectos cruciales del proyecto. Gracias a la flexibilidad del Lean Canvas, se pudo enfrentar estos cambios con eficiencia, adaptando el modelo de negocio según las necesidades.

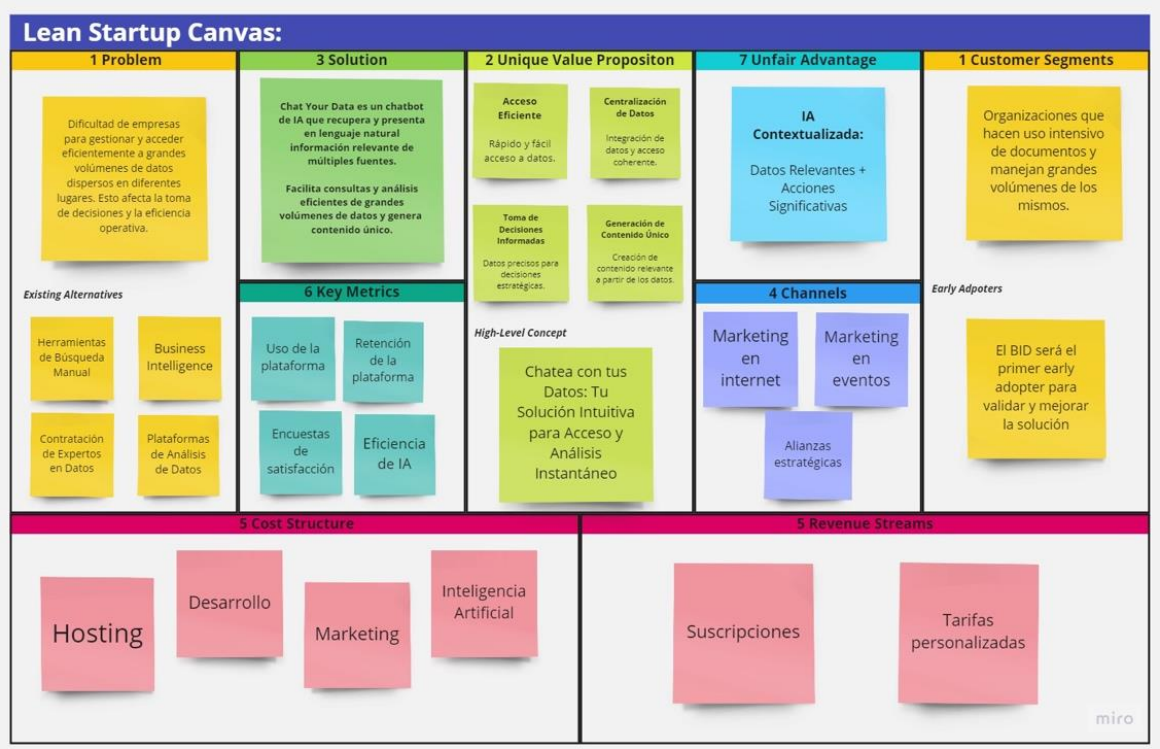


Ilustración 6- *Lean Startup Canvas*

3.1.2.1.1. "*Problem-Solution fit*" y "*Product-Market fit*"

Para entender la solución, es crucial entender dos conceptos clave que guiaron al equipo a la creación de Chat Your Data: el "*Problem-Solution fit*" y "*Product-Market fit*".

- ***Problem-Solution Fit***: Este término se refiere a la alineación entre un problema específico y la solución desarrollada para abordarlo [7]. El equipo buscó identificar y comprender a fondo el desafío común de la gestión de datos en las organizaciones. Este encaje entre el problema y la solución fue la base para la creación de Chat Your Data.
- ***Product-Market Fit***: Una vez que el equipo logró el *Problem-Solution fit*, el siguiente paso fue asegurarse de que la solución sea una respuesta a esas necesidades e intereses del mercado [8]. Este es el *Product-Market fit*, el punto en el cual el producto satisface las demandas del mercado y se convierte en una solución que da valor a un grupo específico de usuarios.

Estos dos conceptos no sólo guiaron el proceso de desarrollo del equipo, sino que también justifican las decisiones a lo largo del proyecto. A continuación, se explorará cómo se aplicaron estos conceptos en la creación de Chat Your Data y cómo la solución refleja un equilibrio perfecto entre el problema, la solución y las necesidades del mercado.

Evidencia del *Problem-Solution Fit* y validación del problema

Para el equipo, el entendimiento profundo del problema fue fundamental en la creación de Chat Your Data. La validación de este problema se basa en los resultados de encuestas y entrevistas realizadas.

Las validaciones dieron un resultado claro: la mayoría de los empleados experimentan dificultades significativas en la gestión de grandes volúmenes de datos. La complejidad de integrar, analizar y acceder eficientemente a estos datos es un obstáculo común en diversas organizaciones. Esta validación aseguró la alineación directa entre el problema que se está abordando y la solución que se presenta.

Product-Market fit y validación de la solución

La validación de la solución fue respaldada por la respuesta positiva de los encuestados. Las encuestas incluyeron preguntas sobre la viabilidad de adquirir una solución como Chat Your Data, confirmando el interés por parte de los encuestados y entrevistados.

Esta validación se extiende más allá de las encuestas. Al comprender las necesidades compartidas por diversas organizaciones, se asegura la posición en el mercado, lo que se denomina el *Product-Market fit*. Los comentarios positivos de los encuestados y entrevistados, en términos de su disposición a adoptar la solución, demuestran que el producto está alineado con las expectativas del mercado.

Además, al ver interés de empresas como lo son Genexus y CloudGaia se refuerza aún más la creencia de que se está atendiendo una necesidad real y valiosa en el mercado. Principalmente, el apoyo del BID fue de gran respaldo. Aunque es una organización con características distintas, como por ejemplo que documenta exhaustivamente todos sus proceso, fue de gran ayuda para los casos que se probaron. Este respaldo externo baja a tierra la idea de que se llegó al *Product-Market fit* al desarrollar Chat Your Data como una solución efectiva y deseada por las empresas.

A continuación, se muestra cómo se evaluó la segmentación de los clientes:

Segmentación de clientes utilizando el modelo TAM SAM SOM [9]

TAM (Total Addressable Market)

- El TAM es el tamaño total del mercado al que se dirige, considerando la demanda máxima de un producto o servicio en una industria específica.
- El TAM para Chat Your Data abarca a todas las organizaciones, a nivel mundial, que manejan grandes volúmenes de datos provenientes de diversas fuentes, en diferentes formatos.
- Estas organizaciones enfrentan desafíos al acceder y analizar eficientemente sus datos. Incluye organizaciones de diversas industrias y tamaños que comparten esta problemática de gestión de datos.

SAM (Serviceable Available Market)

- El SAM representa el mercado al que el equipo puede alcanzar. Es la parte del TAM limitada por factores como la ubicación o la especialización, pero considerando la demanda máxima. Es la dimensión del mercado al que el producto apunta, considerando la demanda máxima de un producto en una industria específica.
- Para Chat Your Data, este conjunto abarca a las organizaciones que no solo tienen desafíos en la gestión de sus datos, sino que tienen la disposición y la capacidad para adoptar una solución como Chat Your Data. Además, son organizaciones de habla hispana o inglesa. En definitiva, el SAM de Chat Your Data se define como las organizaciones medianas y grandes que tienen un enfoque en la tecnología y la innovación.

SOM (Serviceable Obtainable Market)

- El SOM, por otro lado, se refiere al conjunto del mercado al que se puede acceder a corto plazo con los recursos actuales. Estos recursos incluyen aspectos como el equipo, el capital, la ubicación, la popularidad o la competencia en el mercado. El SOM se define por el conjunto limitado del SAM en los que el equipo se va a enfocar en el inicio y en los que se tiene la capacidad de llegar fácilmente.

- Para Chat Your Data, el SOM incluye a las organizaciones ubicadas en Uruguay o que tienen relación con el país, a las cuales Chat Your Data puede acceder, a través de contactos o conexiones previas.

Segmentación final de los clientes

En resumen, la segmentación final de Chat Your Data se enfoca en aquellas organizaciones que hacen uso intensivo de datos en formato texto y manejan grandes volúmenes de documentos. Este enfoque horizontal permite abordar una amplia variedad de sectores y organizaciones que comparten esta necesidad en la gestión de datos y en la toma de decisiones, sin estar restringido a una industria o dominio específico, sino dirigiéndose a aquellas organizaciones y empresas que se caracterizan por su intensivo uso de documentos.

Beta testers

Los *beta testers* no pertenecen oficialmente al modelo TAM SAM SOM, pero juegan un papel esencial en el proceso inicial del producto.

Un *beta tester* tiene las siguientes características:

- Está dispuesto a probar un producto o servicio antes de su lanzamiento oficial.
- Busca activamente problemas y errores en el producto y proporciona *feedback*.
- Su objetivo principal es contribuir a mejorar el producto antes de su lanzamiento.
- Tiene acceso temprano al producto para probarlo en condiciones reales.

En resumen, un *beta tester* se centra en probar, identificar problemas y contribuir a la mejora del producto en desarrollo. [10] Estos *beta testers* son especialmente importantes en las etapas tempranas, ya que su *feedback* y adopción temprana permiten afinar la solución y validar su relevancia y encaje en el mercado. Además, permite entrar en un ambiente real y obtener información valiosa de cómo la solución podría resolver los problemas de manera efectiva.

El BID (Banco Interamericano de Desarrollo) [11] es el *beta tester*, desempeñando un papel fundamental en las primeras etapas de adopción y validación del producto. En conjunto con el BID, se llevaron a cabo pruebas y validaciones para afinar y mejorar el sistema. Además, el equipo optó por realizar pruebas con actores de menor escala con el propósito de obtener una variedad de opiniones y perspectivas.

En la sección 6 Ingeniería de requerimientos, se encuentra un análisis detallado del proceso iterativo llevado a cabo con el BID. Además, se describe tanto el funcionamiento y objetivos del BID como sus necesidades que Chat Your Data satisface, junto con una detallada explicación sobre la comunicación y la colaboración que tuvo lugar durante este proceso.

3.1.2.1.2. Alternativas existentes

Las alternativas actuales para abordar el problema de la gestión ineficiente de sus grandes cantidades de datos incluyen:

Herramientas de búsqueda manual: La forma de muchas organizaciones de acceder a grandes cantidades de datos es realizando búsquedas manuales en múltiples sistemas y fuentes, lo que es lento y propenso a errores.

Business intelligence: Estas herramientas permiten análisis de datos, pero generalmente requieren conocimientos técnicos y tiempo para configurarlas. Es importante destacar que estas soluciones generalmente se centran en datos numéricos, lo que no aborda el desafío de la gestión de datos textuales.

Contratación de expertos en datos: Algunas organizaciones eligen contratar personal especializado en gestionar y analizar los datos, lo que puede ser costoso y además solo depende de los mismos y no de cualquier tipo de empleado.

Plataformas de análisis de datos: Ofrecen análisis automatizados, pero generalmente no tienen la capacidad de comprender preguntas en lenguaje natural y brindar respuestas contextuales. Además, estas plataformas tienden a centrarse en datos estructurados y cuantitativos, lo que limita su utilidad en la gestión de datos de tipo texto.

3.1.2.1.3. Ventaja competitiva

La diferenciación de Chat Your Data se encuentra en su tecnología de inteligencia artificial generativa y contextual.

Con Chat Your Data, para obtener una información específica que se encuentra en uno de los múltiples documentos de la organización, en lugar de tener que utilizar y entender complejas interfaces y consultas, simplemente se puede formular una pregunta en lenguaje natural, transformando la forma en la que se interactúa con la información.

La IA de la plataforma capta la solicitud, extrae los datos necesarios y proporciona al usuario una respuesta clara, facilitando a los usuarios tomar decisiones informadas. Todo esto mejora la eficiencia, y por consecuencia el éxito de la organización.

A continuación, se detallan los aspectos de diferenciación de Chat Your Data:

Respuestas altamente contextualizadas

El aspecto principal de diferenciación de Chat Your Data es su capacidad de ofrecer respuestas altamente contextualizadas en función de los documentos subidos por el usuario y el contexto propio de cada conversación con la IA. En la evolución continua de Chat Your Data, la visión es llevar la contextualización a un nivel todavía más superior al integrar la información de los usuarios, sus roles y la organización a la que pertenecen. A medida que se almacenan y procesan esos datos, no solo se tiene en cuenta la pregunta o la solicitud actual, sino que también se comprende quién es el usuario, en que área de la organización trabaja, cuál es su función, en qué industria se especializa la organización, y qué necesita específicamente según su contexto. Esto llevará la contextualización a un nuevo nivel, permitiendo respuestas aún más precisas, relevantes y contextualizadas.

Flexibilidad de integración

Chat Your Data es desarrollado con la capacidad de expandir sus funcionalidades, más allá de las principales, debido a su posible integración con herramientas para distintos tipos de tareas, como por ejemplo traducción de idiomas, cálculos matemáticos avanzados, búsqueda en la web, entre otros, según las necesidades específicas de los usuarios.

Especialización en datos textuales

A diferencia de las alternativas existentes que se centran principalmente en datos numéricos, Chat Your Data se especializa en datos textuales. Esto permite a los usuarios obtener información eficientemente, formulando preguntas en lenguaje natural y transformando la forma en que interactúan con la información.

Soporte técnico integral

El éxito de cualquier solución va más allá del producto en sí; el soporte técnico y la forma de llegar a sus clientes son partes esenciales de la propuesta de valor. El compromiso de Chat Your

Data es proveer un soporte técnico excepcional, creando una experiencia de usuario personalizada en todos los aspectos, pudiendo estar a disposición en caso de cualquier problema.

3.1.2.1.4. Canales, *Cost Structure*, *Revenue Streams* y *Key Metrics*

En esta sección, se explorarán en detalle los aspectos cruciales que sustentan la viabilidad y el éxito del proyecto. Para ello, se analizaron los canales de distribución y promoción que serán utilizados para alcanzar al público objetivo. También se examinó la estructura de costos, es decir, los costos que involucran la creación y operación del producto. Además, se analizaron las fuentes de ingresos que se espera generar a través del modelo de negocio.

Comparar la estructura de costos con las fuentes de ingresos, permite evaluar la viabilidad del producto y brindar claridad sobre la capacidad del producto para generar beneficios sostenibles y respaldar el emprendimiento a largo plazo.

Finalmente, se definieron las métricas clave que se utilizaran para medir el rendimiento y la efectividad del producto en el mercado. Esta sección brinda una comprensión detallada de cómo se pretende llevar a cabo y sostener la propuesta de valor.

***Channels*:**

- *Marketing* en línea dirigido a organizaciones que enfrentan problemas relacionados con la gestión de datos.
- Participación en eventos y ferias relacionadas con tecnología e inteligencia artificial.
- Alianzas estratégicas con empresas tecnológicas y consultoras para ampliar el alcance.

Cost Structure:

- Desarrollo y mejora continua del *software*.
- Costos de IA – uso de modelos de OpenAI
- Costo de bases de datos.
- Gastos de *marketing* y publicidad.
- Costos operativos para mantener la infraestructura y la atención al cliente.

Key Metrics:

- Número de organizaciones que utilizan la plataforma.
- Tasa de retención de clientes.
- Nivel de satisfacción del cliente.
- Tiempo de respuesta de la IA y precisión de las respuesta.
- Cantidad de *feedbacks* positivos sobre cantidad total de *feedbacks* de las respuestas.
- Cantidad de interacciones por usuario.

Revenue Streams:

- Modelo de suscripción mensual o anual para las organizaciones que utilizan la plataforma.
- Tarifas personalizadas para organizaciones con necesidades específicas.
- Potencialmente, acuerdos de licencia con empresas tecnológicas interesadas en integrar la tecnología Chat Your Data en sus productos.

Modelos de suscripción - Planes

Chat Your Data tiene previsto ofrecer varios modelos de suscripción a sus potenciales clientes. Aunque las características específicas de cada plan ya están delineadas, es crucial entender que los precios aún no están fijados. La razón principal es la fluctuación en los costos relacionados con la inteligencia artificial en los últimos meses. Antes de establecer los precios definitivos, el equipo esperará hasta acercarse más al lanzamiento y tener una base de clientes potenciales. Para más detalles sobre las funcionalidades de cada plan, consultar la sección 13.2 Modelos de Suscripción del anexo.

- **Plan Personal**: Este plan permite a los usuarios acceder a una funcionalidad básica de Chat Your Data de forma gratuita. Incluye características esenciales para un usuario individual.
- **Plan Standard**: El Plan Standard ofrece un conjunto más amplio de características y funcionalidades que son ideales para organizaciones pequeñas y medianas que buscan un mayor control sobre sus datos y análisis.
- **Plan Pro**: El Plan Pro está diseñado para organizaciones más grandes, ofreciendo una mayor cantidad de capacidades y características adicionales que el Plan Standard.
- **Plan Enterprise**: Este plan es la opción más completa y está dirigido a empresas y organizaciones grandes con requisitos de datos y análisis a gran escala. Proporciona un conjunto completo de características como también servicio de soporte y personalización.

3.1.2.2. Análisis del entorno

El equipo optó por hacer un análisis del entorno ya que proporciona una visión integral de los factores que rodean el proyecto, tomar decisiones informadas, anticipar cambios y a adaptarse a las condiciones cambiantes del entorno. El equipo realizó tanto el análisis del macroentorno como del microentorno. En el anexo 13.1 Análisis del entorno se puede encontrar este análisis detallado, donde se realiza un análisis de PESTEL, 5 fuerzas de Porter y FODA. A continuación, se presentan las tendencias tecnológicas que se destacan como fuerzas impulsoras para la transformación y la mejora de los procesos.

3.1.2.2.1. Tendencias tecnológicas

- **Inteligencia artificial**: El campo de la IA está en constante evolución, y se espera que las aplicaciones de la IA sigan creciendo en diferentes industrias. Esta tecnología está revolucionando cómo las empresas procesan y analizan grandes cantidades de datos. Con tecnologías como Machine Learning y el procesamiento del lenguaje natural, se puede obtener información muy importante de los procesos, logrando mejorar la toma de decisiones, y ser más eficientes.
- **Automatización**: Las tecnologías de automatización, como la inteligencia artificial, los *chatbots*, y el Internet de las Cosas (IoT) se están volviendo cada vez más importantes para los emprendedores que buscan mejorar la eficiencia de sus operaciones y la satisfacción del cliente. Las empresas logran reducir los errores humanos y mejorar la eficiencia, aumentando la productividad y rentabilidad.
- **Nube y servicios de alojamiento**: La creciente adopción de servicios de alojamiento en la nube está permitiendo que las empresas tengan acceso a una variedad de herramientas y recursos tecnológicos a un costo menor. Además, permite la colaboración en tiempo real, generando mayor eficiencia en los proyectos.

3.2. Conclusión

En resumen, todas estas herramientas permitieron al equipo identificar y describir de manera precisa el problema que se quiere resolver, se logra entender las necesidades y desafíos que se enfrentan los usuarios.

4. Marco Metodológico

4.1. Contexto

4.1.1. Características del equipo

El equipo se destaca por su buena relación y actitud positiva. La colaboración es una parte fundamental de su enfoque de trabajo. Además, cada miembro del equipo cuenta con experiencia laboral previa en empresas de *software* y está familiarizado con metodologías ágiles y diversas tecnologías. Aunque no poseen experiencia previa en inteligencia artificial, su alto nivel de motivación y compromiso contribuye a un ambiente de trabajo productivo y a la capacidad para superar desafíos de manera efectiva.

4.1.2. Características del proyecto

El proyecto se caracteriza por su complejidad en muchos aspectos. En primer lugar, se trata de un proyecto de tipo emprendimiento, lo que implica que el equipo ha tenido que identificar un problema, validar una solución y desarrollarla. Este tipo de proyectos suele requerir adaptabilidad y ajustes constantes. A pesar de esta complejidad, el equipo tenía un conocimiento claro de los requerimientos y los objetivos que buscaba alcanzar.

Además, el proyecto aborda un tema altamente dinámico, ya que coincidió con el surgimiento del auge en inteligencia artificial generativa. Esto implica que el dominio de la IA no solo es relativamente nuevo, sino que también estaba en constante evolución mientras se avanzaba en el proyecto.

Por lo tanto, el equipo se enfrentó a un proceso de aprendizaje desde cero, entrando en un mundo nuevo y altamente interesante, pero también extremadamente cambiante. La complejidad técnica del proyecto fue un desafío adicional, ya que implicaba la implementación de tecnologías avanzadas y en constante evolución.

4.2. Ciclo de vida

A partir de las características del proyecto y del equipo, se eligió por adoptar un enfoque de ciclo de desarrollo iterativo e incremental, que se ajusta de manera dinámica a las necesidades cambiantes a lo largo del proceso.

Los ciclos iterativos e incrementales son esenciales en este enfoque, permitiendo la repetición de actividades en fases específicas (iteraciones) para aprovechar el progresivo entendimiento del producto por parte del equipo y desarrollarlo gradual y efectivamente [12]. La flexibilidad permite gestionar cambios en los requisitos y necesidades del proyecto. Al final de cada iteración, se obtiene un entregable o conjunto de entregables completados. Las iteraciones siguientes pueden perfeccionar los entregables o bien introducir nuevas funcionalidades, lo que facilita la adaptación constante a cambios tecnológicos y lecciones aprendidas. Esta metodología asegura un desarrollo ágil y orientado a resultados.

En un entorno en constante cambio, donde los objetivos y alcances pueden variar, este enfoque da al equipo la estructura necesaria para manejar los cambios de manera efectiva, garantizando que el producto final satisfaga de manera óptima las necesidades identificadas en cada etapa del proceso.

4.3. Metodologías ágiles

Las metodologías ágiles representan enfoques de gestión de proyectos que se centran en la entrega continua de productos de alta calidad, destacándose por su enfoque en la adaptabilidad, flexibilidad, colaboración y comunicación constante entre los miembros del equipo [13]. El equipo reconoce que el proyecto se encuentra en un campo sumamente dinámico y cambiante, donde se encuentran nuevas tecnologías y avances de manera constante. Por lo tanto, es fundamental la necesidad de adaptarse ágilmente a estos cambios en evolución, minimizando cualquier impacto.

El concepto de agilidad en la gestión de proyectos se resume en varios aspectos clave: la entrega temprana de valor de negocio, la mejora continua del producto y de los procesos utilizados, la flexibilidad en el alcance, la participación activa del equipo y la entrega de productos probados que satisfacen las necesidades del cliente. Estos principios refuerzan aún más la decisión de optar por las metodologías ágiles, ya que encajan perfectamente con los desafíos cambiantes y la naturaleza evolutiva del proyecto.

Cabe destacar que todos los miembros del equipo cuentan con experiencia previa en estas metodologías, lo cual refuerza aún más su elección. Además, el equipo entiende la importancia del proceso de colaboración mutua y la comunicación constante, aspectos que se alinean de manera clara con las metodologías ágiles.

Por esto, el equipo decidió adaptar dos marcos de trabajo que permiten implementar agilidad: Kanban y Scrum.

4.3.1. Kanban

El enfoque Kanban ha sido adoptado para determinadas etapas críticas que no están directamente relacionadas con el desarrollo del código, como: investigación, validación de la idea, planificación, diseño y pruebas.

Su elección se basó en varias ventajas que aportan al proceso [14], explicadas en el anexo 13.3.1 Características de Kanban.

4.3.1.1. Implementación de la adaptación Kanban

Kanban representa un marco visual para la gestión de tareas, fundamentada en el uso de tarjetas o "*post-its*" que representan cada tarea. Estas tarjetas se desplazan a través de un tablero Kanban que presenta diversas columnas, representando distintas etapas del proceso. La finalidad de Kanban radica en visualizar el flujo de trabajo y supervisar la cantidad de tareas en proceso, permitiendo identificar posibles problemas y optimizar la eficacia del procedimiento [15].

La principal ventaja de utilizar Kanban en estas etapas radica en su enfoque en la visualización y gestión del flujo de trabajo, lo cual brinda una comprensión más clara y completa del progreso del proyecto. Además, este marco permite priorizar tareas y asignar responsabilidades de manera sencilla, facilitando la coordinación del equipo y la toma de decisiones. Otra ventaja significativa de Kanban es su adaptabilidad y capacidad para ajustarse a cambios y modificaciones en el proyecto, lo que brinda la posibilidad de responder rápidamente a situaciones imprevistas o ajustes en los requisitos.

Dado que el proyecto involucra inteligencia artificial y procesamiento de lenguaje natural (NLP), áreas que conllevan incertidumbre y posibles cambios a lo largo del camino, se optó por utilizar Kanban en todas las etapas que no están directamente relacionadas con el desarrollo de código, como la investigación, validación de la idea, planificación, diseño, pruebas, entre otras.

En el anexo 13.3.2 Elementos de Kanban se encuentra al detalle la explicación de cómo se implementó el Kanban Board y el uso de las columnas.

En el tablero mostrado a continuación se puede ver que en cada tarjeta se agregaba un título descriptivo, se marcaba quién era el encargado y además el equipo decidió agregar iconos, que funcionan como etiquetas (si era investigación, validación). Dentro de la misma se agregaba todo tipo de información necesaria (*links*, búsquedas, etc.).

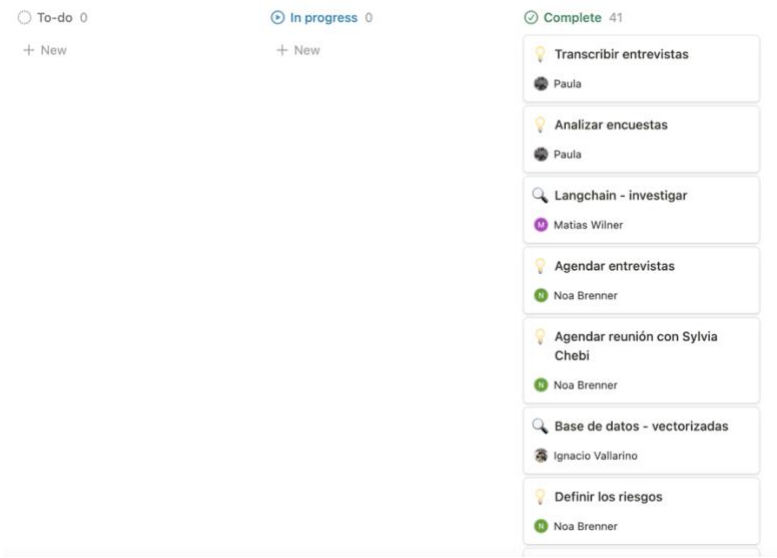


Ilustración 7- Visualización del Kanban *Board*

En resumen, la elección de una adaptación de Kanban como marco de gestión proporcionó una visualización clara del flujo de trabajo, facilitando la priorización de tareas, la asignación de responsabilidades y la adaptación a cambios. Esto ha sido especialmente beneficioso en un proyecto con componentes de IA y NLP, donde la incertidumbre y la necesidad de agilidad son altas.

4.3.2. Scrum

"*Scrum es un marco de trabajo liviano que ayuda a las personas, equipos y organizaciones a generar valor a través de soluciones adaptativas para problemas complejos.*" [16]

Scrum, al igual que el ciclo de vida seleccionado, persigue un enfoque iterativo e incremental, con el propósito de optimizar la previsibilidad, controlar riesgos y abordar desafíos complejos. Basado en cinco valores fundamentales: foco en un objetivo claro, compromiso, respeto, apertura al *feedback* y coraje para asumir riesgos, Scrum proporciona un marco sólido.

En esta línea, el equipo optó por adaptar los principios de Scrum al proceso de desarrollo del producto (es decir, el código principalmente).

Los motivos principales de elección se basan en ciertas características de Scrum, que se encuentran en el anexo 13.3.3 Características de Scrum.

4.3.2.1. Implementación de la adaptación de Scrum

La elección de una adaptación de Scrum fue tomada, en particular, para la fase de desarrollo de código debido a sus notables beneficios. Una de las ventajas principales de Scrum radica en su capacidad para adaptarse a cambios y ajustes en el proyecto. Dado que el proyecto está en un entorno dinámico y tecnológicamente cambiante, Scrum permite la agilidad necesaria para responder rápidamente a cambios en los requisitos y mantener al equipo alineado con los avances más recientes.

Durante la etapa de desarrollo de código, se implementó una adaptación de Scrum. A continuación, se explicará cómo se implementó este marco.

4.3.2.2. Herramienta principal

Para implementar Scrum se optó por utilizar Jira como herramienta principal. Jira facilitó la gestión de tareas y asignación de responsabilidades en el equipo, creando el *product backlog*, *sprint backlog*, y teniendo toda la información en el *board*. A su vez, permitió al equipo visualizar y planificar el *roadmap*, generar gráficas de velocidad y crear *burndown charts*. En resumen, Jira fue fundamental para una implementación exitosa de Scrum en el proyecto.

4.3.2.3. Sprints

Se utilizó este marco durante 18 semanas divididas en 9 *sprints*. En el anexo 13.3.3.1 Sprint se encuentra una definición de que es un *sprint*. La duración de estos fue de 2 semanas cada uno, dando un equilibrio entre tener suficiente tiempo para avanzar en el trabajo y mantener la agilidad necesaria para responder a los cambios y realizar mejoras de manera continua.

4.3.2.4. Scrum Board

Para implementar esta adaptación, el equipo se basó en el *"Scrum Board"*, o tablero de Scrum, es una herramienta visual utilizada para gestionar y dar seguimiento al flujo de trabajo del equipo [17]. En este caso, se utilizó la plataforma Jira para el *Scrum Board*. El tablero permite

organizar y visualizar el *Product Backlog*, el estado actual del *sprint* y el estado actual de *sprints* anteriores.

En el capítulo 8 Gestión del proyecto se encuentra con más profundidad la implementación de la adaptación de Scrum.

4.3.3. Conclusiones

La combinación de Kanban y Scrum fue muy efectiva para el desarrollo del proyecto. Esta combinación permitió adaptarse a los cambios y ajustes en el proyecto, reducir los riesgos del contexto, asegurar la entrega continua de incrementos de *software* y obtener validación constante del *beta tester*. Además, permitió fomentar la colaboración y la comunicación constante entre los miembros del equipo, lo que fue fundamental para lograr un producto final de alta calidad y satisfacción del cliente.

A continuación, se explicará en qué etapa se utilizó cada marco.

4.4. Etapas del proyecto

La ejecución del proyecto se dividió en tres etapas fundamentales. Cada una de estas etapas fue diseñada para abordar diferentes aspectos y necesidades del proyecto de manera progresiva y eficiente.

La primera etapa, enfocada en la investigación, se separó en dos fases distintivas. En primer lugar, se dedicó esfuerzo a la identificación, exploración y validación exhaustiva del problema en cuestión. Durante este período, se profundizó en la comprensión de la problemática, se confirmó su existencia y se evaluó su impacto. A continuación, se avanzó hacia una fase dedicada al aprendizaje y la investigación en el campo de la inteligencia artificial. En esta fase, se asimilaron nuevas herramientas, tecnologías y conceptos fundamentales para la concreción del proyecto. Es relevante destacar que, en ambas fases, se optó por seguir una adaptación del marco de trabajo Kanban como guía.

Una vez finalizada esta primera etapa, se dio inicio a la fase de desarrollo del producto, que a su vez se dividió en dos subetapas: planificación y ejecución de la solución. Durante estas subetapas, el equipo adoptó el marco de trabajo Scrum, que permitió una mayor agilidad y adaptabilidad en el proceso de desarrollo. Asimismo, durante esta fase, se puso énfasis en la validación y el *testing* del producto, garantizando su funcionalidad y calidad.

La tercera etapa se centró en la finalización de la documentación completa y la implementación de mejoras en el producto. Esta fase final, se utilizó para consolidar el proyecto en su totalidad, asegurando tanto la parte de documentación como la optimización del funcionamiento del producto.

4.5. Roles

Durante la fase de investigación, cada miembro del equipo se dedicó al estudio de nuevas tecnologías y al análisis detallado del problema. Una vez establecida una base común de conocimiento, se definieron roles específicos para mejorar la coordinación y gestión del grupo. Esta asignación consideró la experiencia, las preferencias individuales y las áreas donde cada integrante mostró mayor entendimiento. La designación de roles se puede encontrar en el anexo 13.3.4 Designación de roles.

5. Investigación

Cuando el equipo comenzó este proyecto, se enfrentó a un desafío importante: adentrarse en tecnologías y conceptos que ninguno de sus miembros conocía previamente. Además, estaban lidiando con un campo que estaba en pleno auge, con tendencias y avances que evolucionaban constantemente.

Ante este panorama, los integrantes del equipo decidieron dedicar tiempo en aprender. Tomaron cursos, leyeron en internet, asistieron a charlas y conferencias, y se reunieron con personas expertas. Dedicaron tiempo entendiendo las tecnologías emergentes, explorando las últimas tendencias y familiarizándose con los conceptos clave.

A continuación, se presentarán algunos de los conceptos clave que el equipo adquirió durante esta fase de investigación. Estos conocimientos no solo les permitieron comprender mejor las decisiones que llevaron a Chat Your Data a su estado actual, sino que también se convirtieron en una base fundamental para enfrentar los desafíos que encontraron en el camino.

5.1. Introducción a la Inteligencia Artificial

Desde el final del 2022, el concepto de inteligencia artificial ("IA" de aquí en adelante) comenzó a resonar fuertemente a partir del surgimiento de ChatGPT. A partir de esta herramienta se pudo experimentar el poder que puede llegar a tener la IA, y cómo puede cambiar radicalmente la vida de las personas. Tal es así, que es la primera vez en la historia que cientos de expertos en IA, empresarios tecnológicos y otras eminencias como Elon Musk y Yuval Harari, firmaron una carta abierta pidiendo a todos los laboratorios de IA que suspendieran inmediatamente durante al menos 6 meses el entrenamiento de sistemas de IA. La carta solicitaba una pausa en el desarrollo de la IA generativa para considerar los riesgos del rápido despliegue de los modelos de IA generativa en la sociedad [18].

En un mundo donde las máquinas no solo ejecutan tareas, sino que también comprenden, aprenden y crean, la inteligencia artificial se ha transformado en una herramienta indispensable. En este capítulo, sé mostrarán las capas de la IA, desde su definición, hasta sus aplicaciones más avanzadas, como la IA generativa y la arquitectura Transformer. Se explorará cómo los modelos de lenguaje (LM) han evolucionado para comprender y generar lenguaje de una manera que, en ocasiones, no es distinguible con el del humano.

La IA se define como un campo de estudio dentro de la informática que busca crear sistemas capaces de realizar tareas que, de requerir inteligencia cuando son realizadas por humanos, implican habilidades como entender el lenguaje natural, reconocer patrones, resolver problemas y aprender de la experiencia. Esta tecnología aspira a diseñar máquinas que puedan mejorar su rendimiento a lo largo del tiempo y adaptarse para realizar tareas específicas, apoyando o incluso automatizando decisiones y procesos que tradicionalmente han dependido de la intervención humana. La IA explora el desarrollo de algoritmos que pueden procesar información de una manera similar a cómo lo harían los humanos, pero es crucial notar que, aunque los sistemas de IA pueden simular la inteligencia humana, carecen de consciencia y entendimiento genuino [19]. En el anexo 13.4 Evolución de la inteligencia artificial se explica la evolución y como es hoy en día.

5.1.1. Machine Learning

Machine Learning es una rama de la inteligencia artificial que se enfoca en desarrollar sistemas que aprenden de los datos. En vez de estar explícitamente programados para llevar a cabo una tarea, estos sistemas utilizan los datos para hacer predicciones o tomar decisiones sin necesidad de programación específica para esa función [20] .

Existen diferentes tipos de Machine Learning:

- ***Supervised Learning* (Aprendizaje Supervisado)**: Implica que los datos son etiquetados. El *dataset* que se le envía contiene *labels* (una etiqueta por cada dato) y un *target* (que es lo que se quiere predecir).
- ***Unsupervised Learning* (Aprendizaje no Supervisado):** Los datos no están etiquetados. Lo que se le solicita es encontrar patrones. De cada registro no se tiene un *target*. Lo que hace el algoritmo es descubrir patrones entre los datos.
- ***Reinforced Learning* (Aprendizaje por Reforzamiento):** El modelo aprende a tomar una serie de decisiones mediante recompensas y penalizaciones, tratando de maximizar la recompensa a lo largo del tiempo [21].

Los Large Language Models (LLMs*)*, explicados en detalle más adelante, son fundamentados y construidos sobre principios de Machine Learning. De esta manera, los LLMs pueden ser entrenados en un modo no supervisado a través de diferentes corpus de texto para entender las distribuciones de probabilidad de las palabras en el lenguaje. Por otro lado, se puede realizar

un *fine-tuning* sobre un LLM, siendo así un aprendizaje supervisado. Este proceso está explicado en la sección 13.6 Fine-Tuning del anexo.

5.1.2. Redes Neuronales

Las redes neuronales se inspiran en la estructura y funcionamiento de las redes neuronales biológicas, específicamente en cómo los cerebros humanos procesan la información. Están compuestas por unidades de procesamiento denominadas “neuronas” o “nodos”, estructuradas en diversas capas: de entrada, ocultas, y de salida [22]. Se puede encontrar más información en la sección 13.7 Redes Neuronales del anexo.

5.1.3. *Deep Learning* (Aprendizaje Profundo)

Deep Learning, o Aprendizaje Profundo, se posiciona como una subcategoría del Machine Learning. Este enfoque particulariza en el uso de redes neuronales profundas, es decir, redes neuronales con múltiples capas entre la entrada y la salida [23]. Se puede encontrar más información en la sección 13.8 Deep Learning del anexo.

5.1.4. Inteligencia artificial generativa

La Inteligencia artificial generativa es una rama de la inteligencia artificial que se enfoca en la generación de contenido original a partir de datos existentes. Se basa en métodos de aprendizaje automático que recogen información sobre determinados elementos y, posteriormente, utilizan sus datos para generar ideas nuevas y realistas [24].

Una vez que el equipo había asimilado este concepto, tomó la decisión de utilizar esta tecnología. Reconociendo que no solo buscaban un *chat* que respondiera en base a los datos, sino también uno que pudiera crear contenido original a partir de los mismos.

A continuación, se verá cómo se logra la inteligencia artificial generativa a partir de la arquitectura Transformer.

5.1.5. Arquitectura Transformer

La comunicación humana, compleja y multifacética (es decir, que tiene muchas dimensiones), permaneció durante mucho tiempo como un campo fértil para la exploración dentro del dominio de la IA. En 2017, el artículo "*Attention is All You Need*" de Vaswani y su equipo, se introdujo una nueva arquitectura de red neuronal para tareas de procesamiento del lenguaje natural

(Natural Language Processing [NLP]). Esta arquitectura, denominada Transformer, se basa en el mecanismo de atención, que permite al modelo aprender dependencias a largo plazo entre diferentes partes de una secuencia de entrada. Esencialmente, se utilizan mecanismos de autoatención que ponderan la importancia de las distintas palabras de la entrada al generar cada palabra de salida [25].

Este proceso comprende varios pasos:

1. **Tokenización**: El texto de entrada (*prompt*) se divide en partes más pequeñas, generalmente palabras o subpalabras (llamados *tokens*). Posteriormente, estos *tokens* se convierten en representaciones numéricas (vectores), que se utilizan como entrada para el modelo.
2. ***Embedding***: La entrada tokenizada pasa por una capa de incrustación (*embedding*) que traduce esos *tokens* a un espacio de mayor dimensión. Esto permite al modelo aprender y representar relaciones complejas entre diferentes palabras.
3. **Atención**: Permite que la arquitectura Transformer manipule y comprenda las dependencias y relaciones entre diferentes palabras en una secuencia, proporcionando una comprensión contextual.
4. **Predicción**: Finalmente, el modelo predice el siguiente *token* basándose en el contexto de los *tokens* anteriores. Este paso se repite hasta que se cumpla una condición de finalización, normalmente la longitud máxima del texto de salida o un *token* especial.

En el anexo 13.5 Arquitectura Transformer, se explorarán con mayor detalle estos conceptos.

5.2. Lenguaje Natural y Modelos de Lenguaje

5.2.1. Natural Language Processing (NLP)

El campo de la inteligencia artificial abarca diversas áreas y dominios de investigación. Uno de sus aspectos más destacados es el Procesamiento del Lenguaje Natural (NLP, por sus siglas en inglés). El concepto original de la inteligencia artificial, que se remonta a la definición de Alan Turing sobre máquinas pensantes hace más de 60 años, sentó las bases para el desarrollo del NLP [26].

El NLP es un subcampo especializado que se concentra en la interacción entre las computadoras y los seres humanos utilizando el lenguaje natural. Su principal objetivo es permitir que las

máquinas lean, comprendan y atribuyan significado al lenguaje humano de manera eficaz, añadiendo valor a esta interacción.

Un concepto relacionado dentro del NLP es la Generación del Lenguaje Natural (NLG, por sus siglas en inglés). Se refiere a un conjunto de modelos matemáticos que no solo comprenden el lenguaje humano en sus diversas formas y lenguas, sino que también han mejorado significativamente en la tarea de generar lenguaje [27].

En los últimos cinco años, se ha observado un progreso destacado en los denominados Modelos de Lenguaje Grandes o Modelos de Lenguaje Fundamentales. Las innovaciones en su arquitectura han potenciado su habilidad para procesar y entender el lenguaje con mayor precisión.

5.2.2. Language Models (LM) y Large Language Models (LLMs)

Dentro del dominio del Procesamiento del Lenguaje Natural (NLP, por sus siglas en inglés), existen múltiples modelos y algoritmos diseñados para interpretar, generar, procesar y modificar el lenguaje. Estos algoritmos se conocen como Modelos de Lenguaje (LM). Su función principal es analizar grandes cantidades de texto para aprender relaciones entre palabras y frases, permitiendo comprender y generar información en formatos como libros, artículos y sitios web. Los LMs, como BERT y BARD, han sido clave para predecir palabras, comprender el lenguaje y categorizar oraciones [28].

En 2022, se experimentó un avance significativo con la introducción de los Modelos de Lenguaje Grandes (LLMs). Estos son esencialmente Modelos de Lenguaje que se han ampliado enormemente en términos de datos y capacidad. Su denominación "Large" se refiere a la enorme cantidad de datos con los que se entrenan. Un LLM puede entrenarse con hasta 10 billones de palabras, lo que le permite detectar y aprender una gran gama de patrones y estructuras lingüísticas. Gracias a su extenso entrenamiento con textos variados de Internet, los LLMs pueden generar contenido sobre una amplia variedad de temas en diferentes idiomas.

Un ejemplo prominente de estos LLMs es GPT-3 o GPT-3.5, que revolucionó el mundo de la inteligencia artificial en 2022. Este modelo, basado en la tecnología de Transformers, ha demostrado una capacidad sin precedentes para comprender y generar lenguaje.

5.2.3. Modelos de IA destacados: GPT-3 y GPT-4

GPT-3 y GPT-4 son Large Language Models (LLM) desarrollados por OpenAI. Son modelos de aprendizaje automático entrenados en un conjunto de datos masivo de texto y código. GPT-3 fue lanzado en 2020 y GPT-4 en 2023.

GPT-3*:* es un modelo de lenguaje generativo preentrenado. Esto significa que puede generar texto, traducir idiomas, escribir diferentes tipos de contenido creativo y responder a sus preguntas de manera informativa. Sin embargo, GPT-3 también tiene algunas limitaciones. Por ejemplo, puede generar texto que sea sesgado o inexacto [29].

GPT-4: es una versión mejorada de GPT-3. Es más grande, más preciso y más adaptable que su predecesor. GPT-4 también es capaz de entender y responder a las consultas del usuario de manera más contextual.

Dentro del anexo en la sección 13.10 Diferencias de GPT-3 y GPT-4, se puede encontrar un análisis detallado de las principales diferencias entre GPT-3 y GPT-4.

5.3. Técnicas de generación de contenido en IA

Una característica distintiva de los LLMs en comparación con la programación convencional radica en cómo se obtienen resultados. En lugar de depender de una lógica de código complicada, estos modelos trabajan con comandos dados en lenguaje natural. Estos comandos, conocidos como "*prompts*", son instrucciones, preguntas o fragmentos de texto que facilitan la comunicación con sistemas basados en inteligencia artificial.

En términos simples, un *prompt* es una solicitud dada al modelo para que realice una tarea específica. Esencialmente, es una manera de decirle a la máquina lo que se espera de ella usando palabras comunes. Al estructurar correctamente estos *prompts*, es posible lograr resultados sorprendentes, precisos y creativos.

El arte y la ciencia de diseñar estos *prompts* han dado origen a un campo especializado: la Ingeniería de Prompts (Prompt Engineering). Al igual que en la programación tradicional, donde el código se prueba y se refina, en el Prompt Engineering se experimenta con diferentes formulaciones para optimizar la comunicación y mejorar los resultados obtenidos de los sistemas de IA. Esta similitud en la metodología es lo que le otorga el término "*Engineering*" a esta práctica.

5.3.1.1. Prompt Engineering

Como se mencionó anteriormente, Prompt Engineering es el proceso de diseñar y probar *prompts* que permitan a los LLMs generar resultados precisos, creativos y relevantes.

El prompt engineering es una práctica importante para aprovechar al máximo el potencial de los LLM. Los *prompts* bien diseñados pueden ayudar a los LLM a realizar tareas que de otro modo serían difíciles o imposibles. Hoy en día, gracias a esta disciplina es que se pueden construir aplicaciones con inteligencia artificial más fácilmente [30].

En el contexto de los modelos de lenguaje de inteligencia artificial como ChatGPT, el término alucinaciones se utiliza para describir situaciones en las que el modelo genera información o detalles que no están respaldados por sus datos de entrenamiento o que son falsos. Un ejemplo de alucinación sería cuando un usuario pregunta: "¿Cuál es la capital de Australia?" y ChatGPT responde: "La capital de Australia es Sydney," cuando en realidad la respuesta correcta es Canberra. Por esta razón, es fundamental saber cómo escribir los *prompts* para que los resultados sean los más precisos posibles. Además, el concepto de "temperatura" también influye en las alucinaciones que el modelo genera. En la próxima sección, se explicará en detalle este concepto.

5.3.1.2. Chain of Thought

La cadena de pensamiento (Chain of Thought) se refiere a la secuencia de pasos que un modelo de aprendizaje automático utiliza para llegar a una conclusión. Es un método de Prompt Engineering recientemente desarrollado que alienta al LLM a explicar su razonamiento. De esta manera, la persona que formula el *prompt* le indica al modelo cómo debe pensar. Esta forma de explicación del razonamiento suele traducirse en resultados más precisos [31]. En la sección 13.11 Diferencia entre Standard Prompting y Chain of Thought Prompting del anexo se encuentra un ejemplo comparativo entre un *prompt* normal o al usar este método.

5.3.2. Control de la creatividad: Uso de la temperatura en IA generativa

La temperatura es un parámetro que toma valores entre 0 y 1, que se utiliza para controlar la aleatoriedad de las respuestas generadas por un modelo de lenguaje. A una temperatura baja, el modelo generará respuestas más precisas y coherentes, pero también puede ser menos variado

y creativo. A una temperatura alta, el modelo generará respuestas más creativas y originales, pero también puede ser menos preciso y coherente.

La temperatura se utiliza en los modelos de lenguaje generativo, como los utilizados por los *chatbots* o los generadores de texto creativo. Estos modelos se entrenan en un conjunto de datos de texto y código, y luego pueden generar texto nuevo a partir de ese conjunto de datos. La temperatura se utiliza para controlar la probabilidad de que el modelo genere texto que sea similar al texto del conjunto de datos de entrenamiento o que sea más creativo y original [32].

Por ejemplo, a una temperatura baja, un *chatbot* podría generar la siguiente respuesta a la pregunta "¿Cuál es tu color favorito?" Mi color favorito es el azul.

A una temperatura alta, el *chatbot* podría generar la siguiente respuesta a la misma pregunta: Mi color favorito es el color del cielo en un día soleado. Es un azul brillante y vivo que me hace sentir feliz.

La temperatura es una herramienta importante que puede utilizarse para controlar el equilibrio entre la precisión y la creatividad en las respuestas generadas por un modelo de lenguaje.

5.4. Desarrollo de *bots* y cadenas de *bots*

5.4.1. Definición y funcionalidades de un *bot*

El término *bot*, originario de robot, es un programa que realiza tareas repetitivas, predefinidas y automatizadas. Los *bots* están diseñados para imitar o sustituir una tarea o acción humana. Operan de forma automatizada, por lo que pueden trabajar mucho más rápido que una persona [33].

Chatbot: Un *chatbot* es un programa de inteligencia artificial que puede simular una conversación (o un *chat*) con un usuario en lenguaje natural a través de aplicaciones de mensajería, sitios web, aplicaciones móviles o por teléfono [34].

5.4.2. Creación y utilización de cadenas de *bots*

Una cadena de *bots* es un sistema de *bots* interconectados que trabajan juntos para completar tareas. Cada *bot* de la cadena tiene un conjunto específico de habilidades y conocimientos, y pueden comunicarse entre sí para compartir información y colaborar en soluciones.

La principal ventaja de utilizar cadenas de *bots* radica en la precisión que ofrecen. Si los *bots* son capaces de llevar a cabo tareas con un alto grado de exactitud -lo que reduce significativamente el riesgo de cometer errores- la implementación de una cadena de *bots* tiende a mejorar aún más esta precisión. A su vez, al utilizar cadenas de *bots* se puede prever alucinaciones, lo cual es fundamental en estos casos [35].

5.5. Memoria

5.5.1. Importancia de la memoria en modelos de IA

En el área de la inteligencia artificial generativa, la memoria es la habilidad que tiene un LLM (en este caso un *chatbot*) de recordar la conversación pasada con un usuario. Esto es fundamental para que la conversación sea lo más humana, natural y fluida posible. Para esto es necesario pasarle contexto de las conversaciones anteriores.

Existen diferentes maneras en las que se le puede incluir memoria a los LLMs, a continuación, se verán algunas técnicas para poder incluir el contexto a un LLM [36].

5.5.2. Tipos de Memoria

Algunos de los tipos de memoria más comunes son:

Memoria de estado: Este tipo de memoria almacena información sobre el estado actual de la conversación. Esto incluye información como el tema de la conversación, las entidades que se han mencionado, y cualquier otra información relevante.

Memoria de contexto: Es un tipo de memoria que almacena información sobre el contexto de la conversación. Esto incluye información como el entorno en el que se está llevando a cabo la conversación, las emociones del usuario, y cualquier otra información relevante.

5.5.2.1. Entity Memory

EntityMemory es un tipo de memoria que almacena información sobre las entidades que se han mencionado a lo largo de la conversación. Una entidad es un objeto o concepto que se menciona durante la conversación. Por ejemplo, una persona, un lugar, un evento, o un concepto abstracto.

La EntityMemory es un tipo de memoria de estado. Esto significa que almacena información sobre el estado actual de la conversación. En el caso de la EntityMemory, la información almacenada incluye un resumen de cada entidad que se menciona en el *chat* [37].

5.6. Bases de datos en inteligencia artificial

5.6.1. Introducción a bases de datos vectoriales

Las bases de datos vectoriales son esenciales para la gestión de datos en la actualidad. Estas bases de datos tienen la capacidad de almacenar y recuperar vectores, incluso en dimensiones muy altas, lo que las convierte en un recurso valioso para diversos casos de uso de búsqueda vectorial, como la búsqueda visual, semántica y multimodal [4].

Lo más interesante es que en la era de la inteligencia artificial (IA), las bases de datos vectoriales han evolucionado aún más. Se han fusionado con modelos de texto de IA generativa para crear agentes inteligentes que ofrecen experiencias de búsqueda conversacional. Además, desempeñan un papel crucial en la prevención de "alucinaciones" por parte de los modelos de IA generativa, lo que garantiza que los *chatbots* proporcionen respuestas veraces y precisas [37].

Estas bases de datos no solo potencian la innovación, sino que también simplifican el desarrollo de aplicaciones de IA y la operatividad de las cargas de trabajo relacionadas con la IA. En resumen, las bases de datos vectoriales son una tecnología versátil y fundamental en el mundo actual de la gestión de datos y la inteligencia artificial.

5.6.2. Aplicación de *Chunks*

Los *chunks* son una unidad de información, formada a partir de la agrupación de varios elementos de un documento como palabras, frases o conceptos.

En el procesamiento de texto la “división en *chunks*” es una técnica clave. Es el proceso de dividir un texto en fragmentos más pequeños, para facilitar su procesamiento.

Los *chunks* también se utilizan en los modelos de lenguaje generativo, para generar texto nuevo, poemas, artículos o código. Con *chunks*, estos pueden representar el conocimiento que tienen del lenguaje y del mundo, generando texto coherente y que tiene sentido [38].

5.6.2.1. Recursive Character Text Splitting

Recursive Character Text Splitter divide los datos del documento de manera recursiva para mantener juntos todos los párrafos, oraciones y palabras tanto como sea posible. Los caracteres predeterminados proporcionados son ["\n\n", "\n", " ", ""] [39].

El divisor recursivo de caracteres funciona de la siguiente manera:

- Divide el documento en fragmentos según los separadores.
- Si el tamaño de un fragmento es mayor que el tamaño máximo permitido, el fragmento se divide en subfragmentos de igual tamaño.
- El proceso se repite hasta que todos los fragmentos tengan un tamaño menor o igual al tamaño máximo permitido.

Se puede decir que comienza por intentar dividir el texto grande utilizando el primer carácter "\n\n". Si la división inicial por "\n\n" sigue siendo grande, pasa al siguiente carácter que es "\n" y trata de dividirlo por él. Si aún es más grande que el tamaño de fragmento especificado continúa moviéndose al siguiente carácter en el conjunto hasta que obtenga una división que sea menor que el tamaño de fragmento especificado [40].

5.6.3. Búsqueda y recuperación de información en bases de datos vectoriales

5.6.3.1. Búsqueda en una base de datos vectorial por “Similarity Search”

Similarity Search (búsqueda de similitud), es un mecanismo que permite obtener los elementos más semejantes a un elemento de consulta dentro de un conjunto de datos. Dentro de las bases de datos vectoriales, la búsqueda de similitud es fundamental dado que ayuda a navegar en espacios de alta dimensión para obtener vectores que son más similares a un vector de consulta dado, según una métrica definida [41].

En un espacio vectorial, cada elemento, ya sea un documento, imagen o cualquier otra forma de datos, está representado como un vector de números. La similitud entre dos vectores se mide típicamente usando métricas de distancia. El objetivo es encontrar vectores que estén más cerca del vector de consulta.

En un escenario práctico, se cuenta con una base de datos de documentos de textos, como artículos de noticias, que previamente han sido transformados en vectores mediante la aplicación de un modelo de *embedding* de texto. Cada uno de estos vectores se encuentra almacenado dentro de la base de datos vectorial. Luego, se hace una búsqueda por el título del articulo "Avances de la inteligencia artificial en la salud", en donde es sometido al proceso de conversión en un vector número, utilizando el mismo modelo de *embedding* que se utilizó anteriormente.

El proceso de búsqueda de similitud se desarrolla de la siguiente manera [42]:

1. Se calcula la similitud entre el vector de consulta y cada uno de los vectores almacenados en la base de datos, utilizando una métrica seleccionada.
2. Se procede a la recuperación de los vectores que han obtenido puntuaciones de similitud que superan un umbral previamente definido o que se ubican en el grupo de las primeras *N* coincidencias.
3. Se extraen los documentos originales correspondientes a los vectores seleccionados y se presentan como los resultados de la búsqueda.

En el anexo, en la sección 13.12 Mecanismo de funcionamiento de Similarity Search en Bases de Datos Vectoriales, se explaya más sobre este concepto.

5.7. Artefactos de la inteligencia artificial generativa

5.7.1. *Chain*

Un *chain* es una secuencia de herramientas o LLMs que se utilizan para realizar una tarea compleja. Un *chain* se compone de una serie de pasos, cada uno de los cuales es realizado por una herramienta diferente. Los pasos del *chain* están conectados entre sí, lo que significa que el resultado de un paso se utiliza como entrada para el siguiente paso, y asi sucesivamente hasta llegar al último paso de la *chain*. La idea es que cada paso de la *chain* se encargue de una tarea en particular para realizar una tarea. Esto sirve para poder mejorar las respuestas o los resultados de una IA [43].

5.7.2. *Agent*

Un *agent* es una entidad, ya sea un programa o sistema, que actúa autónomamente en un entorno para alcanzar objetivos predeterminados. Estos *agents* son capaces de percibir su entorno, tomar decisiones basadas en esa percepción y realizar acciones con el fin de lograr sus metas.

Mientras que algunos *agents* son diseñados para tareas específicas y sencillas, como jugar a un juego determinado, otros son más complejos y pueden gestionar múltiples funciones, como controlar robots o asistir a usuarios en la obtención de información [44].

5.7.2.1. Agent Scratchpad

El Agent Scratchpad es un área de almacenamiento temporal utilizado por un agente conversacional de inteligencia artificial para guardar información durante una conversación. El *scratchpad* puede ser usado para almacenar información como el estado actual de la conversación, las preferencias de los usuarios, las ideas y pensamientos propios del agente [45].

5.7.3. Introducción a MRKL y MRKL *Agents*

MRKL*: Modular Reasoning, Knowledge & Language system* (Sistema Modular de Razonamiento, Conocimiento y Lenguaje). Es una arquitectura modular que combina *Large Language Models* (LLMs), *external knowledge sources* (Fuentes de Conocimiento Externas) y *discrete reasoning* (Razonamiento Discreto).

Esta arquitectura está diseñada para superar las limitaciones de los LLMs y mejorar sus capacidades de razonamiento, permitiéndoles realizar deducciones lógicas, inferencias y toma de decisiones basadas en el contexto proporcionado.

Un agente MRKL típicamente consta de tres partes: un conjunto de módulos (por ejemplo, una calculadora, una API meteorológica, una base de datos, etc.), un enrutador que determina cómo conectar las consultas en lenguaje natural entrantes hacia el módulo adecuado y un LLM que actúa como el enrutador [46].

Por ejemplo, si un sistema MRKL recibe la pregunta "¿Cómo está el clima en Nueva York?", el LLM puede extraer la información necesaria de la pregunta, indicar al Sistema MRKL que utilice una API meteorológica para obtener la información y formular una respuesta. El sistema MRKL reconocería el comando para la API meteorológica y proporcionaría la información relevante.

5.7.3.1. Arquitectura MRKL y su Aplicación en la Toma de Decisiones

La arquitectura de un agente MRKL está diseñada de manera modular, lo que le permite combinar las capacidades de razonamiento de los LLMs con fuentes de conocimiento externas y módulos de razonamiento discreto [47].

A continuación, se presenta un desglose detallado de la arquitectura MRKL:

1. **Large Language Model (LLM):** El *LLM* actúa como la interfaz principal para interactuar con el usuario. Procesa las entradas y salidas en lenguaje natural y también es responsable de enviar las consultas al módulo apropiado según el contexto.
2. **Módulos (*Tools*)**: Estos son componentes separados que realizan tareas específicas. Los módulos pueden ser desde una calculadora para operaciones matemáticas, una API meteorológica para obtener datos meteorológicos, una base de datos para almacenar y recuperar información, hasta una herramienta de búsqueda en la web para encontrar información en línea. Los módulos amplían las capacidades del LLM y permiten que el agente MRKL realice tareas que van más allá del alcance del LLM.
3. **Enrutador**: Este componente decide qué módulo debe manejar una consulta dada. El enrutador está entrenado para extraer la información correcta de la salida del LLM y dirigirla al módulo correcto. El proceso de enrutamiento es fundamental para que el agente MRKL funcione de manera efectiva, ya que garantiza que las consultas sean manejadas por el módulo más adecuado.

En cuanto a la toma de decisiones, un agente *MRKL* utiliza el LLM para comprender el contexto y la intención de una consulta. Sobre la base de esta comprensión, el LLM genera un plan de enrutamiento, que luego es ejecutado por el enrutador. El enrutador dirige la consulta al módulo adecuado para su procesamiento. Una vez que el módulo ha procesado la consulta, el resultado se devuelve al LLM, que luego formula una respuesta en lenguaje natural.

Por ejemplo, si se le pide a un agente MRKL que calcule la suma de dos números, el LLM comprendería la intención de la consulta y la dirigiría hacia un módulo de calculadora. El módulo de calculadora realizaría el cálculo y devolvería el resultado al LLM, que luego generaría una respuesta para el usuario.

5.8. LangChain*:* Un marco de trabajo para la IA generativa

Se llevó a cabo una investigación exhaustiva para identificar las herramientas potenciales que podrían ser aplicables en el proyecto, lo que incluyó la exploración de opciones como Haystack, LangChain, entre otras. A continuación, se detallará la herramienta seleccionada para su implementación, LangChain.

5.8.1. Introducción y aplicaciones de LangChain

LangChain es un *framework* diseñado para simplificar la creación de aplicaciones utilizando Large Language Models (LLM), permite a los desarrolladores combinar modelos LLM como GPT-4 con recursos externos de datos [48].

Sus principales beneficios son:

- Agilidad: Permite a los desarrolladores crear aplicaciones rápidamente y sin tener que empezar desde cero.
- Flexibilidad: Permite a los desarrolladores adaptar las aplicaciones a sus necesidades específicas.
- Escalabilidad: Permite a las aplicaciones crecer y adaptarse a las necesidades cambiantes.

Así, LangChain facilita la creación de aplicaciones de inteligencia artificial, proporcionando algoritmos, herramientas e implementaciones que aceleran el desarrollo de estas aplicaciones. LangChain convierte el desafío en una solución, ofreciendo un conjunto de herramientas y métodos para aprovechar al máximo la inteligencia artificial.

5.9. Otros conceptos clave

Además de lo explicado anteriormente, el equipo también realizó investigaciones sobre otras herramientas que, aunque no están directamente relacionadas con la inteligencia artificial, son esenciales para el proyecto en su conjunto. Estas herramientas incluyen Firebase (Firebase Auth, Guards, Firebase Firestore, etc.), Docker y AWS (S2, Container Registry, Task, Services, Clusters, Elastic IP). Se pueden encontrar enlaces a estos conceptos en el anexo, en la sección 13.13 Otros conceptos claves .

6. Ingeniería de requerimientos

6.1. Obtención y análisis de los requerimientos

6.1.1. Herramientas para la obtención de requerimientos

Es fundamental identificar a los *stakeholders* y comprender los diferentes tipos de usuarios que sufren el problema. En este contexto, los *stakeholders* son aquellos individuos y equipos dentro de las organizaciones que enfrentan desafíos al acceder y aprovechar eficazmente los datos en formato texto. Para lograr conocer al usuario, el equipo implementó herramientas específicas destinadas a comprender a los usuarios y obtener los requisitos necesarios.

6.1.1.1. Encuestas

La fase inicial de análisis de requerimientos, que corresponde a la etapa de empatía en el enfoque de Design Thinking, se llevó a cabo mediante encuestas y entrevistas dirigidas a personas de diversas organizaciones con el objetivo de entender sus necesidades y expectativas en cuanto al manejo de grandes cantidades de datos, así como sus expectativas respecto a una plataforma para integrar y gestionar esa información.

En un primer plano, el equipo se dedicó a realizar encuestas a individuos pertenecientes a organizaciones de diferentes tamaños y sectores. Esta recopilación de información le permitió al equipo entender más a fondo los desafíos que estas tienen con la administración diaria de grandes cantidades de datos.

Algunos de los cuestionamientos fundamentales abordados en estas encuestas comprendieron las siguientes preguntas. El *link* a la encuesta se encuentra en la sección 13.15.1 Link de la encuesta del anexo.

- ¿Ha tenido problemas para manejar grandes cantidades de datos en su organización?
- ¿Ha utilizado alguna plataforma de *chatbot* con IA para manejar datos organizacionales?
- ¿Qué características espera encontrar en una plataforma de *chatbot* con IA para manejar datos organizacionales?
- ¿Cuáles son las principales dificultades que encuentra en el manejo de grandes cantidades de datos en su organización?

Los resultados de estas encuestas permitieron sacar conclusiones que son de utilidad para el proyecto.

En primer lugar, es importante destacar que las respuestas a las encuestas indicaron que la mayoría de las organizaciones experimentan problemas para manejar sus grandes cantidades de datos, indicando así que existe la necesidad en el mercado para una solución que pueda ayudar a abordar estos problemas.

A su vez, se obtuvo que en la mayoría de las organizaciones no habían incursionado previamente en el uso de plataformas de *chatbot* que utilicen IA para el manejo de datos organizacionales, lo que afirma la oportunidad de mercado para nuestra plataforma. Además, un 83% mencionó que si le gustaría una plataforma de *chat* con inteligencia artificial personalizada para su organización.

Ilustración 8- Evidencia de resultado de encuesta

Además, se le preguntó a los encuestados, ¿qué beneficios piensas que podrías obtener si la IA tiene acceso a los datos de tu organización?

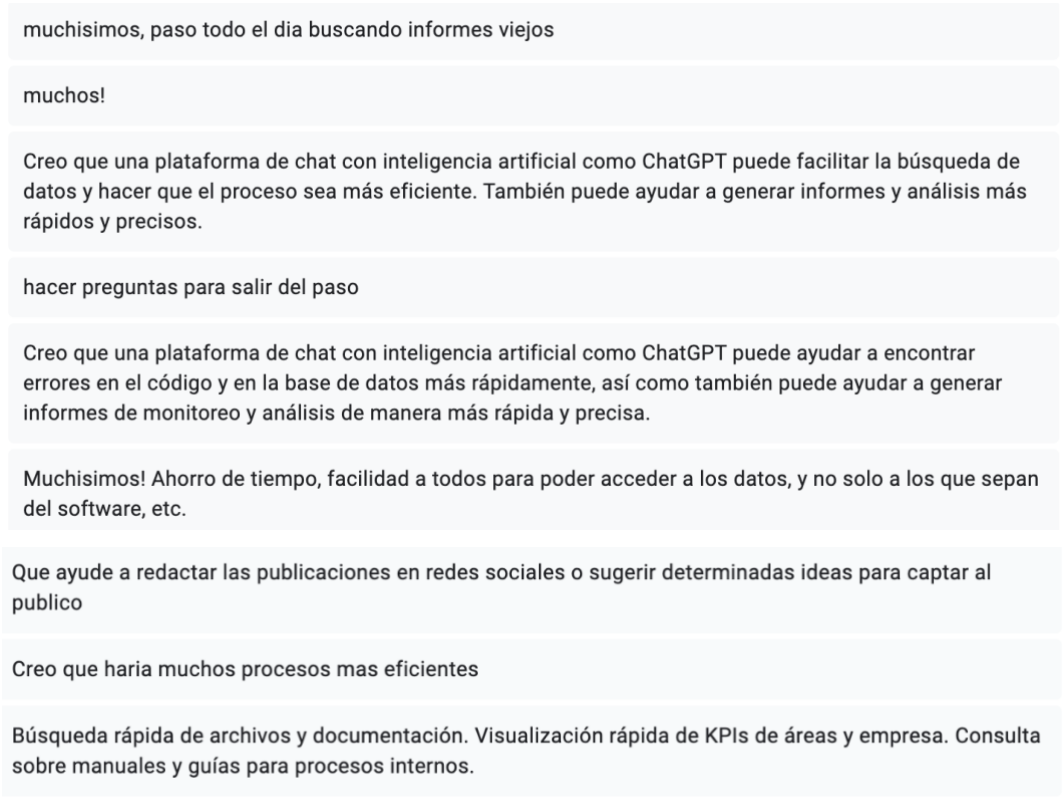


Ilustración 9- Evidencia de resultados de encuesta

También se le preguntó a los encuestados que preguntas le harían al *chat* para hacer su trabajo más eficiente y estas fueron algunas de sus respuestas.

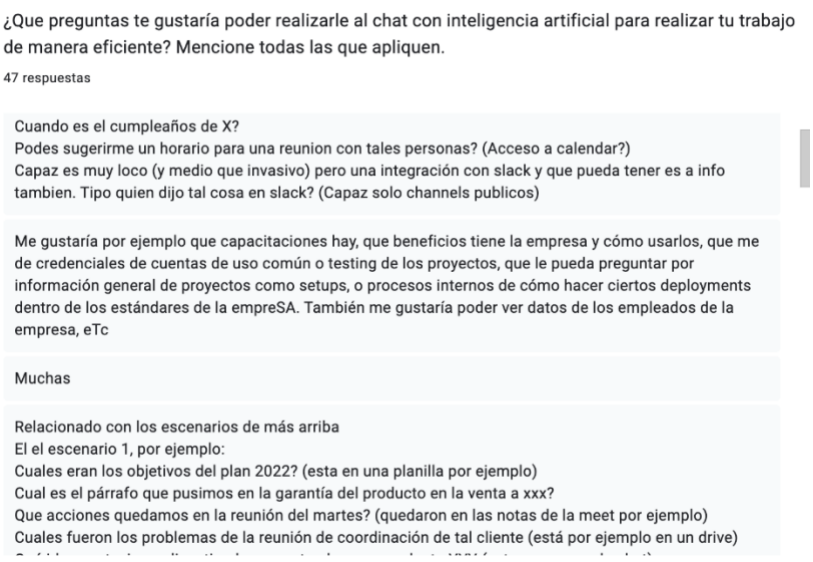


Ilustración 10- Evidencia de resultados de la encuesta

Otra conclusión destacada de las encuestas fue obtener las características más esperadas en una plataforma de *chatbot* para manejar datos organizacionales. Estas son en su mayoría, la

posibilidad de realizar consultas en lenguaje natural, la generación de contenido único y la rapidez de respuesta.

La principal dificultad que enfrentan las organizaciones encuestadas en su manejo de grandes cantidades de datos es la falta de tiempo para procesar la información. Esto sugiere que la plataforma debe ser capaz de procesar los datos de forma rápida y eficiente, para que quien use la plataforma pueda ahorrar tiempo y esfuerzo.

Otra gran dificultad a la que se enfrentan la mayoría de las personas encuestadas es la falta de habilidades técnicas de los empleados para analizar y gestionar sus grandes cantidades de datos, lo que nos indica que la plataforma debe ser fácil de usar y minimizar los requerimientos técnicos. Para lograr esto, una buena interfaz de usuario es crucial.

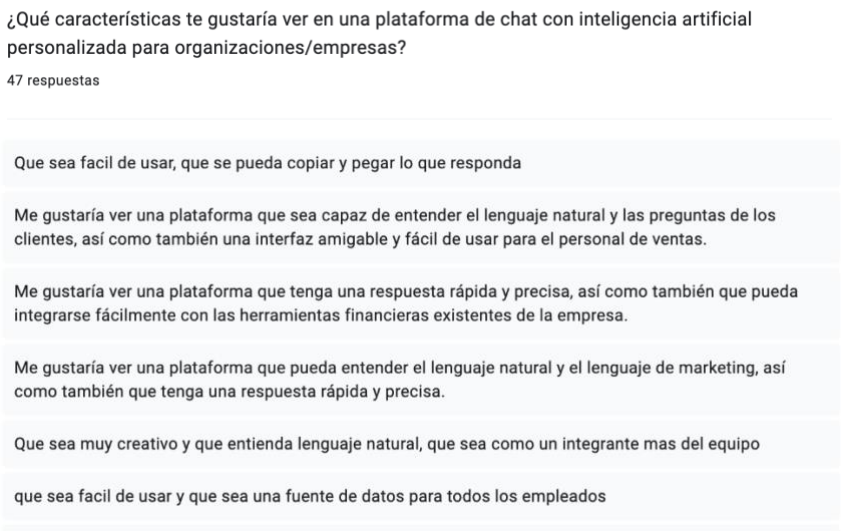


Ilustración 11- Evidencia de resultados de encuesta

Como conclusión general del resultado de las encuestas, el equipo logró observar que está clara la necesidad de una solución que les ayude a manejar sus grandes cantidades de datos. Las respuestas obtenidas a través de las encuestas le brindaron al equipo una visión clara de las características principales que las personas buscan, entre ellas, la capacidad de realizar consultas en lenguaje natural y la generación de contenidos únicos. También es importante destacar que la plataforma debe ser capaz de procesar los datos rápida y eficientemente para ahorrar tiempo y esfuerzo a los usuarios, y que debe ser fácil de usar para requerir la mínima cantidad posible de conocimientos técnicos. En el anexo 13.15.2 Resultados de la encuesta se encuentran todos los demás resultados.

6.1.1.2. Entrevistas

Además de las encuestas, también se realizaron entrevistas a posibles clientes de la plataforma para complementar la investigación. Entre las más principales, Mayda Kurdian, *Research and Development Manager* de K2B, Alan Kind, director de unidad del BID (Banco Interamericano de Desarrollo), Laura Sánchez, Gerente de Proyectos en una empresa de consultoría, Matías Sobol, *Tech Leader* en empresa de *software*, Camila Pakciarz, asesora en empresa de servicios financieros; ProCapital y Ernesto Krawchik, Managing Director de CloudGaia.

A continuación, se detallan algunas de las ideas principales que el equipo encontró importantes. En el anexo 13.14.2 Transcripción de entrevista a Mayda Kurdian se puede encontrar una de las entrevistas transcriptas.

Manejo de grandes volúmenes de datos

Todos los entrevistados coinciden en los desafíos que enfrentan con respecto a la gestión de datos en sus respectivas organizaciones. La dispersión de datos, la dificultad para encontrar información relevante y el tiempo dedicado a la búsqueda y procesamiento de datos son temas recurrentes.

Mayda Kurdian mencionó que la empresa en la que trabaja maneja grandes cantidades de datos y que han experimentado dificultades para procesar y analizar la información de forma efectiva, ya que está distribuida en distintas fuentes, lo que les genera una gran pérdida de tiempo.

Alan Kind también compartió la perspectiva de Mayda. Afirmó que en su organización ha tenido dificultades para manejar grandes cantidades de datos, tanto así que anteriormente intentaron crear un *chatbot* de ayuda a este problema, pero no tuvo éxito. El motivo por el cual no tuvo éxito se debió a que no cumplió con las expectativas.

Laura Sánchez trabaja en una empresa de consultoría y tiene la responsabilidad de gestionar proyectos. En cuanto al proceso de búsqueda y acceso a la información dentro de su empresa, Laura explicó que depende del tipo de información, pero en su mayoría se encuentra en algún sistema o repositorio. Sin embargo, la dificultad radica en que la información está dispersa y es necesario recopilar manualmente los distintos componentes de la información. En una semana típica de trabajo, Laura dedica de 6 a 10 horas a la búsqueda de información. Utiliza plataformas como *CRM, ERP y Google Drive* para buscar información.

Matías Sobol trabaja en una empresa de *software* y tiene la responsabilidad de dirigir proyectos. Según el, el proceso de búsqueda y acceso a la información en su empresa implica utilizar Notion donde se publica información técnica, así como intercambiar información a través de correos privados.

Matías también realiza investigaciones en sitios web relacionados con temas técnicos. A veces, encuentra dificultad para encontrar casos específicos debido a que la información la tienen las personas y no está centralizada. Matías dedica más de 3 horas por semana a buscar información. Utiliza Notion y la web en general como plataformas para buscar información.

Camila Pakciarz trabaja en una empresa de servicios financieros en el área de cuentas. En su empresa, el proceso de búsqueda y acceso a la información se realiza mediante un buscador interno similar a *Google*. Camila encuentra dificultad en saber qué palabra clave utilizar para buscar la información y también en el hecho de que los mensajes de error o la información están en diferentes idiomas dependiendo de quién los reportó. En una semana típica de trabajo, Camila dedica aproximadamente 2 horas diarias buscar información. Utiliza sistemas internos y *Google* como plataformas para buscar información.

Experiencia en el uso de herramientas de inteligencia artificial

En cuanto a su experiencia en el uso de herramientas de inteligencia artificial, Matías compartió que ha tenido la oportunidad de utilizar soluciones de *chat* con IA en el pasado, y su experiencia con ChatGPT fue sumamente positiva. Laura no utilizó soluciones conversacionales con IA anteriormente, pero está entusiasmada con probar. Por otro lado, Mayda Kurdian, Alan Kind y Ernesto Krawchik también han experimentado el uso de aplicaciones con inteligencia artificial y se han mostrado entusiasmados por las posibilidades que ofrecen.

Importancia de plataforma de *chatbot* con IA

Tanto Mayda, Alan y Ernesto compartieron sus perspectivas sobre la importancia fundamental de contar con una plataforma de *chatbot* con IA en el entorno organizacional. Mayda destacó la importancia de tener una plataforma de *chatbot* con IA para manejar datos organizacionales, ya que esto podría ayudar a automatizar muchas de las tareas que hoy en día se realizan manualmente. Para Alan, esta solución se presenta como una respuesta viable a problemas complejos relacionados con la organización y el procesamiento de grandes cantidades de datos

organizacionales. A su vez, Ernesto también cree que una plataforma con IA puede ser de gran utilidad en CloudGaia y su integración con Salesforce.

Características necesarias de Chat Your Data

Durante las entrevistas, se destacaron una serie de características cruciales que se consideran esenciales en una plataforma como Chat Your Data: facilidad de uso e intuitividad, personalización, creación de contenido nuevo, etc.

Mayda y Camila mencionaron la necesidad de una plataforma que sea fácil de usar y que no requiera habilidades técnicas avanzadas. Según Mayda, muchos de los empleados de la empresa no tienen experiencia en tecnología, por lo que una solución intuitiva sería muy importante. También destacó la importancia de la personalización, ya que cada empresa tiene necesidades únicas y una solución que pueda adaptarse a ellas sería más efectiva.

Además, Alan mencionó que, dada su experiencia, las características más importantes en una plataforma de *chatbot* con IA para manejar datos organizacionales son la capacidad de procesar grandes cantidades de datos eficientemente, y la integración con múltiples fuentes de datos. También mencionó la importancia de la seguridad de los datos organizacionales, y la necesidad de una solución escalable y adaptable que pueda ir evolucionando a la vez de las necesidades de la organización.

Además, se llevaron a cabo reuniones con personas influyentes en la industria, como Sylvia Chebi, Cofounder and Executive Director - ThalesLab y Sergio Fogel, Cofounder de dLocal. Ambos brindaron valiosos consejos y orientación sobre cómo abordar el problema y la solución propuesta. Recomendaron realizar un exhaustivo análisis del negocio, además de destacar de manera positiva el enfoque y análisis con Lean Canvas del equipo. También enfatizaron la importancia de presentar prototipos a los posibles usuarios para obtener retroalimentación y perfeccionar la propuesta.

6.1.1.3. User Persona

Antes de profundizar en los detalles de la solución y cómo esta aborda las necesidades de los usuarios, el equipo llevó a cabo un proceso de desarrollo de User Persona para comprender más sobre quiénes son sus usuarios y cuáles son sus desafíos.

El desarrollo de User Persona es una herramienta valiosa en el diseño de productos centrados en el usuario. Esta técnica se basa en crear perfiles ficticios, basados en investigaciones y datos reales [49]. El equipo se basó en la información previamente recopilada en entrevistas, encuestas y análisis de mercado para crear estos perfiles.

Esto permitió al equipo visualizar a los usuarios como individuos reales, lo que es de gran ayuda para comprender las necesidades, comportamientos, deseos y frustraciones de los mismos y llegar a una solución relevante para los usuarios.

A continuación, se detalla a Juana Pérez, una coordinadora de *marketing* en una empresa de *e-commerce*. Su perfil refleja a un usuario típico que enfrenta desafíos de acceso a información y toma de decisiones basadas en datos, similares a los de muchas otras personas en roles parecidos.

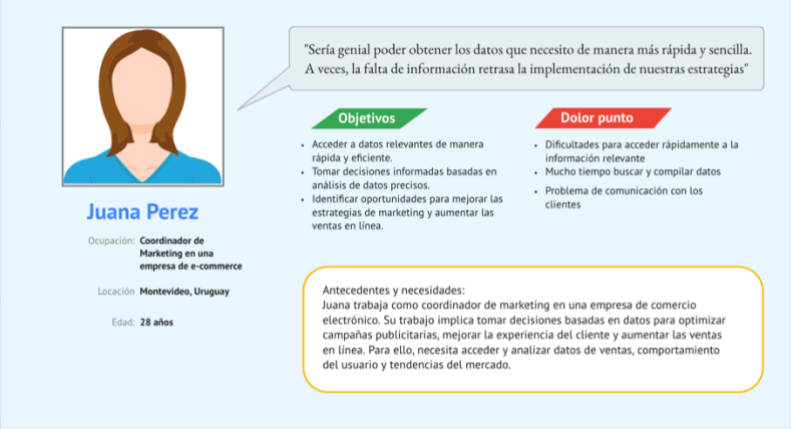


Ilustración 12- User Persona

6.1.1.4. Customer Journey

Para comprender cómo Chat Your Data puede abordar las necesidades y desafíos de sus usuarios, es esencial analizar cómo interactúan los usuarios con la plataforma de Chat Your Data en su día a día. El proceso de ilustrar el Customer Journey es de gran ayuda al equipo para tener una visión detallada de las diferentes etapas que atravesarían los usuarios al utilizar la plataforma.

El Customer Journey es una representación visual de las interacciones que los usuarios tienen con la plataforma desde el primer contacto hasta la finalización de una tarea o un objetivo. Al mapear estas etapas, se puede identificar puntos de dolor, oportunidades de mejora y áreas donde ofrecer una mejor experiencia [50].

En la sección 13.16 Customer Journey del anexo se detalla uno de los usuarios clave, Ana, quien trabaja en el departamento de desarrollo de negocios y necesita acceder y analizar datos relevantes de forma rápida y eficiente.

Este nos permite visualizar cómo la interacción de Ana con Chat Your Data simplificará su proceso de toma de decisiones, aumentará su productividad y la ayudará a superar los desafíos que enfrenta a diario.

6.1.1.5. Mapa de Empatía

El equipo realizó un mapa de empatía para comprender con mayor profundidad las necesidades de los usuarios. Este brinda una visión completa de su experiencia, entendiendo sus pensamientos, emociones, preocupaciones y entorno [51]. Esta perspectiva del usuario es esencial para diseñar una solución lo más relevante posible para el usuario. A continuación, se detalla el mapa de empatía para el potencial usuario de Chat Your Data; Marina Santos, en donde se explora detalladamente cada aspecto del mismo.

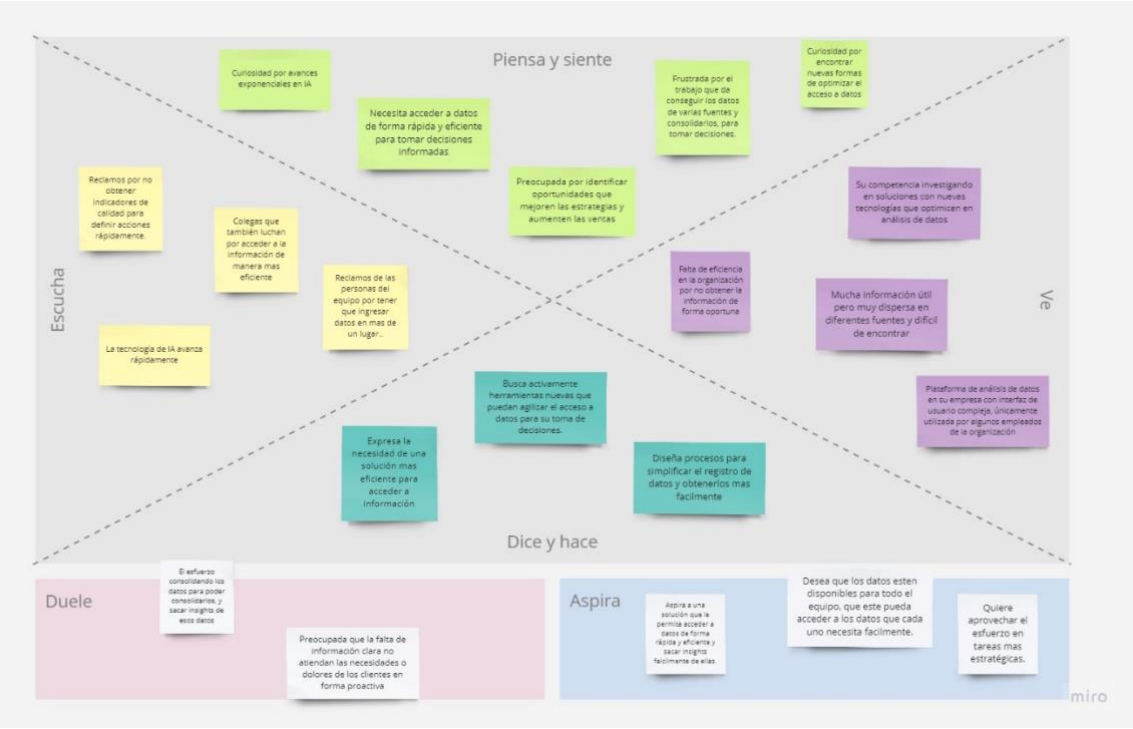


Ilustración 13- Mapa de empatía

La utilización de herramientas como User Persona, Customer Journey y Mapa de Empatía ayudaron al equipo a comprender aún mejor las necesidades de los usuarios y su experiencia y por consecuencia definir requerimientos más detallados y precisos que abordaran de manera

efectiva las necesidades reales de los usuarios. Toda esta información obtenida fue muy útil para el desarrollo del proyecto y ayudó al equipo a llegar a una solución que aborde de manera óptima el desafío de la gestión de datos organizacionales.

6.1.2. Conclusiones

Es fundamental destacar que todos los entrevistados coincidieron en que enfrentan desafíos significativos en lo que respecta a la gestión de sus datos organizacionales. Estos problemas varían desde la dispersión de los datos hasta la dificultad en su búsqueda, análisis y uso efectivo.

Los entrevistados destacaron que la plataforma tendría un impacto positivo en la productividad y eficiencia de la organización al reducir los tiempos de búsqueda y análisis de información. Sin embargo, también se expresaron preocupaciones relacionadas con la confidencialidad de la información, la confiabilidad de los datos y la calidad del contenido en el que se basa la inteligencia artificial.

Los entrevistados incluso mostraron que están dispuestos a pagar por un chat con inteligencia artificial para acceder a los datos de su organización.

Además, el interés compartido en las soluciones de inteligencia artificial subraya aún más la relevancia de Chat Your Data como una plataforma que promete abordar los desafíos que enfrentan las organizaciones en la gestión de datos y la toma de decisiones. El testimonio positivo de aquellos que han tenido experiencias previas con tecnologías similares refuerza la confianza en las capacidades y el potencial de Chat Your Data en el mercado organizacional.

En conclusión, la información recopilada de las encuestas, entrevistas y el resto de los métodos explicados anteriormente refleja una clara demanda en el mercado de una solución que ayude a las organizaciones a manejar sus grandes cantidades de datos de manera efectiva. Se destaca la importancia de desarrollar un *chatbot* con inteligencia artificial que sea fácil de usar, personalizable y seguro. Esta solución debe tener la capacidad de procesar eficientemente grandes volúmenes de datos, integrarse con diversas fuentes de información, ser escalable y adaptable, y garantizar la seguridad de los datos organizacionales.

6.2. Especificación de requerimientos

La etapa de Especificación de Requerimientos es crucial en el proceso de desarrollo de cualquier proyecto, ya que implica la definición clara y detallada de lo que se espera lograr con

el producto final. En esta fase, se establecen las funcionalidades, características y comportamientos clave que el producto debe tener para cumplir con los objetivos establecidos.

6.2.1. Prototipos

Para llevar a cabo esta etapa, el equipo utilizó una herramienta de prototipado, que es una técnica efectiva para visualizar y comunicar cómo se verá y funcionará el producto antes de su desarrollo completo. Esta fase se enmarca también dentro del enfoque de Design Thinking.

6.2.1.1. Prototipación guiada por el usuario

Para obtener una comprensión más profunda de las necesidades y experiencias de los usuarios, el equipo llevó a cabo una prototipación guiada por el usuario. Esta técnica permitió a los usuarios dibujar soluciones que abordan sus problemas específicos.

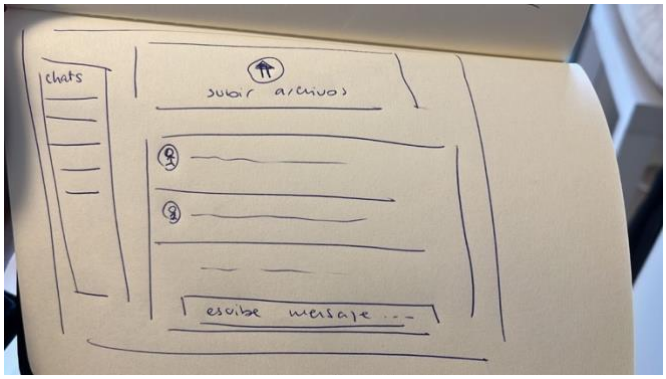


Ilustración 14- Prototipación guiada por el usuario

Luego de esta colaboración, el equipo identificó los siguientes aspectos:

Proceso de importación: Los usuarios destacaron la importancia de una interfaz intuitiva para importar los documentos. Estos valoran una navegación clara y una presentación visual de los pasos a seguir.

Visualización del *chat*: Los usuarios enfatizaron la necesidad de una representación minimalista y clara del *chat* y su conversación. Valoran que el *chat* sea la sección principal en la aplicación, lo más fácil de acceder.

6.2.1.2. Prototipo mockup interactivo

El equipo optó por crear un prototipo *mockup* interactivo utilizando la herramienta *Uizard*. Un prototipo mockup interactivo es una representación visual dinámica de la interfaz de usuario de la aplicación o producto en desarrollo. A través de esta herramienta, el equipo pudo diseñar de manera detallada la disposición de elementos en la interfaz, como botones, campos de entrada y áreas de visualización, y simular interacciones simples entre estos elementos.

La elección de *Uizard* como herramienta de prototipado permitió al equipo crear prototipos *mockup* interactivos de manera eficiente y efectiva. La herramienta ofrece una interfaz intuitiva, funciones con IA y fácil de usar que facilitaron al equipo.

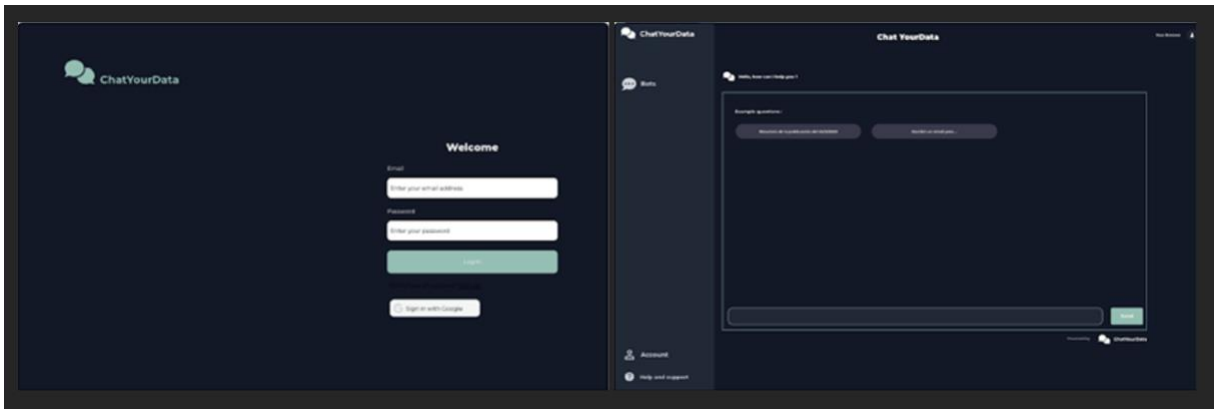


Ilustración 15 – Evidencia del prototipo

6.2.1.3. Prototipo Funcional

Se implementó un prototipo funcional basado en los comentarios y prioridades de los usuarios. Esto permitió validar el flujo de trabajo y la usabilidad de la plataforma. Del mismo, se obtuvieron las siguientes conclusiones:

- **Interacción en tiempo real:** Los usuarios valoraron altamente la capacidad de interactuar con los datos en tiempo real. La actualización instantánea de visualizaciones y resultados fue una característica importante.
- **Flujo de trabajo intuitivo:** La secuencia de pasos para importar, analizar y visualizar datos fue afinada para que los usuarios no encuentren obstáculos en su flujo de trabajo.

Mejoras y ajustes identificados: Tras analizar las sesiones de prototipación y recopilar los comentarios de los usuarios, se identificaron las siguientes áreas clave para mejoras:

- **Claridad en las acciones:** Se resaltó la necesidad de hacer que las acciones y botones sean descriptivos y sin ambigüedades.
- **Visibilidad del progreso:** Los usuarios valoraron la inclusión de indicadores de progreso claros para tareas que requieren tiempo de procesamiento.

Estos resultados de la prototipación han sido fundamentales para el desarrollo continuo, asegurando que la plataforma se ajuste a las necesidades y expectativas de los usuarios.

6.3. Validación de Prototipos

A partir de estos primeros prototipos, surgieron oportunidades de mejora tanto en términos de usabilidad como de funcionalidades. Estas observaciones permitieron al equipo iterar y evolucionar los prototipos, con el objetivo de alcanzar una interfaz que fuera verdaderamente intuitiva y de fácil uso para los usuarios.

Durante este proceso de mejora, el equipo se inspiró en diversas interfaces exitosas, como la de ChatGPT y otros sistemas de *chat* con IA, que se caracterizan por su simplicidad y accesibilidad.

La iteración constante y la retroalimentación del equipo y de posibles usuarios jugaron un papel crucial en la evolución de los prototipos. Cada mejora implementada se basó en la retroalimentación recibida y en la observación de cómo los usuarios interactuaban con la interfaz. Esto permitió afinar la plataforma a medida que se identificaban y abordaban los posibles obstáculos y áreas de mejora.

6.3.1. Evaluación de la usabilidad en los prototipos

El proceso de evaluación de usabilidad se llevó a cabo permitiendo a los usuarios interactuar con la plataforma sin asistencia, observando cómo se desenvolvían durante su uso.

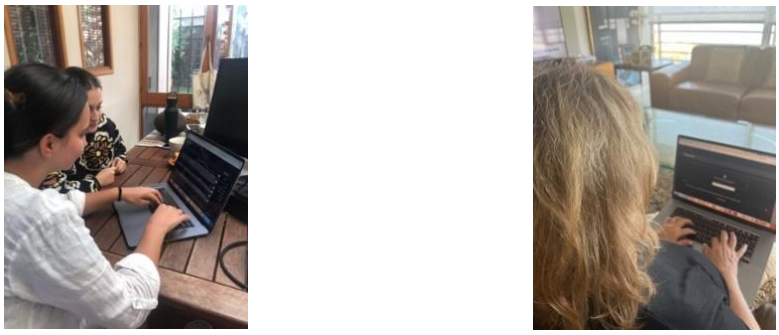


Ilustración 16- Evaluación de la usabilidad

6.3.1.1. Encuestas de satisfacción y retroalimentación

Con el fin de obtener una comprensión más completa de la experiencia del usuario, se llevaron a cabo encuestas de satisfacción. Estas encuestas proporcionaron información valiosa sobre cómo los usuarios se sintieron al utilizar la plataforma y cómo percibieron su funcionalidad y usabilidad.

Además, la retroalimentación recopilada a través de estas encuestas fue esencial para identificar áreas específicas de mejora. Las opiniones y comentarios de los usuarios nos brindaron una visión directa de sus necesidades y expectativas, lo que permitió realizar ajustes y refinamientos en la plataforma para lograr una experiencia óptima.

6.4. Lista de requerimientos

6.4.1. Historias de usuario

El equipo decidió escribir los requerimientos en formato de historias de usuario. Una historia de usuario es una descripción breve, informal y en lenguaje sencillo de lo que un usuario quiere hacer dentro de un producto de *software* para obtener algo que considera valioso [52].

El equipo tomó esta decisión por varias razones. En relación a los usuarios, esta elección coloca a estos individuos en el centro de las conversaciones relacionadas con la incorporación o modificación de elementos en un producto de *software*. La utilización de historias de usuario también ayuda a comprender el contexto y la justificación de las creaciones y cómo se agrega valor al producto, teniendo en cuenta las necesidades de los usuarios o clientes.

Además, el equipo notó que anotar los criterios de aceptación antes de implementar la historia, hizo que el equipo sea más productivo al evitar la repetición de trabajos como resultado de malentendidos.

Las características de una buena historia de usuario son las siguientes:

- El título debe describir una actividad.
- La narrativa debe incluir un rol, una característica y un beneficio.
- El título del escenario debería decir qué es diferente.
- El escenario debe describirse en términos de hechos, eventos y resultados.
- Los datos deben definir todo, y no más que, el contexto requerido.

- El evento debe describir la característica.
- La historia debe ser lo suficientemente pequeña como para caber en una iteración.

El equipo buscó que las historias de usuario cumplan con el concepto de INVEST, explicado en el anexo 13.17 INVEST.

6.4.2. Requerimientos principales funcionales

User Story RF-A01 – Carga de Archivos

Como usuario administrador quiero poder cargar archivos de datos de formato texto al sistema, para poder hacer consultas sobre los mismos.

Criterios de Aceptación:

- El sistema debe permitir a los administradores seleccionar y cargar un archivo de datos de formato texto.
- El sistema debe verificar que el archivo cargado cumpla con el formato requerido y notificar al usuario si no es compatible.
- El sistema debe almacenar el archivo cargado en el repositorio centralizado.
- El sistema debe proporcionar una confirmación al usuario de que la carga del archivo ha sido exitosa.

User Story RF-A02 – Almacenamiento de Archivos

Como administrador, quiero que los archivos cargados se almacenen de forma segura y estructurada en un repositorio centralizado, para facilitar su posterior acceso y uso por parte del equipo.

Criterios de Aceptación:

- Los archivos cargados se almacenan de forma segura y estructurada en el repositorio centralizado.
- Cada archivo almacenado recibe un identificador único para su fácil recuperación.
- Recibo una confirmación de que el almacenamiento del archivo ha sido exitoso.

User Story RF-A03 – Consulta sobre Archivos mediante Chatbot

Como usuario, quiero poder hacer preguntas en lenguaje natural al *chatbot* sobre los archivos subidos al sistema, para obtener respuestas relevantes basadas en la información contenida en los archivos.

Criterios de Aceptación:

- Puedo formular preguntas en lenguaje natural al *chatbot* sobre los archivos.
- El *chatbot* procesa y comprende la pregunta formulada.
- El *chatbot* busca la respuesta en los archivos del repositorio que sean relevantes para la pregunta.
- El *chatbot* proporciona una respuesta basada en la información de los archivos en lenguaje natural y comprensible.

User Story RF-A04 – Generación de Contenido Útil por parte del *Chatbot*

Como usuario, quiero poder pedirle al *chatbot* que genere contenido útil basado en la información de los archivos, como la creación de correos electrónicos o informes, para facilitar tareas relacionadas a la información contenida en los archivos.

Criterios de Aceptación:

- Puedo solicitar al *chatbot* que genere contenido basado en la información de los archivos.
- El *chatbot* procesa la solicitud y selecciona la información relevante de los archivos.
- El *chatbot* genera el contenido solicitado, como un correo electrónico o informe, utilizando la información de los archivos de manera coherente y comprensible.

User Story RF-A05 – Eliminación de Archivos

Como administrador, quiero poder eliminar archivos del sistema cuando ya no sean necesarios o estén desactualizados, para mantener una gestión eficiente de los recursos y garantizar integridad en los datos.

Criterios de Aceptación:

- Puedo seleccionar un archivo de datos previamente cargado para eliminarlo.
- El sistema verifica si tengo los permisos necesarios para eliminar el archivo.

- El archivo seleccionado se elimina del repositorio centralizado.
- Recibo una confirmación de que la eliminación del archivo ha sido exitosa.

User Story RF-A06 – Visualización de Archivos

Como administrador o empleado autorizado, quiero poder visualizar el contenido de los archivos almacenados, para acceder a la información relevante y realizar análisis o consultas necesarias.

Criterios de Aceptación:

- Puedo seleccionar un archivo de datos previamente cargado para visualizar su contenido.
- El sistema busca y recupera el archivo almacenado correspondiente.
- Se muestra en pantalla el contenido del archivo de manera legible y comprensible.

User Story RF-A07 – Ver Conversaciones

Como usuario, quiero poder ver la lista de conversaciones almacenadas en la plataforma para acceder rápidamente a conversaciones previas.

Criterios de Aceptación:

- En la interfaz de usuario, se muestra una lista de conversaciones previas del usuario de manera clara y organizada.
- El usuario puede hacer clic en una conversación de la lista para acceder a ella y ver el historial de mensajes.
- Las conversaciones se mantienen persistentes en la lista de conversaciones, lo que permite al usuario acceder a ellas en futuras sesiones.

User Story RF-A08 – Crear Nuevo Chat

Como usuario, quiero tener la capacidad de crear un nuevo *chat* para iniciar una conversación con otros usuarios o el *chatbot*.

Criterios de Aceptación:

- En la interfaz de usuario, se ve una opción visible para iniciar un nuevo *chat*.
- Al crear un nuevo *chat*, el usuario ve el nuevo *chat* iniciado.

- El usuario puede enviar mensajes en el nuevo *chat* y recibir respuestas de los destinatarios.

User Story RF-A09 – Eliminar *Chat*

Como usuario, quiero tener la posibilidad de eliminar *chats* de mi lista de conversaciones cuando ya no sean necesarios o relevantes.

Criterios de Aceptación:

- En la lista de conversaciones, cada conversación tiene una opción para eliminarla de la lista.
- Al seleccionar la opción de eliminar, se muestra una confirmación para que el usuario confirme la eliminación.
- Una vez confirmada la eliminación, la conversación desaparece de la lista de conversaciones del usuario.

User Story RF-A10 – Dar *Feedback* a Respuestas

Como usuario, quiero poder dar *feedback* sobre las respuestas del *chatbot* para mejorar la calidad del servicio.

Criterios de Aceptación:

- Cada respuesta proporcionada por el *chatbot* tiene la opción para que el usuario de *feedback*.
- El *feedback* puede diferenciar entre las respuestas satisfactorias y no satisfactorias.
- Si el usuario selecciona que una respuesta no fue satisfactoria, se abre una opción que le permite enviar comentarios adicionales sobre por qué no fue satisfactoria.

User Story RF-A11 – Mostrar Referencias en Respuestas del *Chatbot*

Como usuario, quiero poder ver las referencias utilizadas por el *chatbot* en sus respuestas, para tener acceso a la fuente de la información brindada.

Criterios de Aceptación:

- Cada respuesta generada por el *chatbot* incluye un enlace o referencia a la fuente de la información.
- El enlace o referencia a la fuente es claramente visible dentro de la respuesta del *chatbot*.
- Los enlaces o referencias se presentan de manera legible y con un formato intuitivo, lo que permite a los usuarios acceder fácilmente a la información original.

- Al *clickear* los enlaces, estos redirigen al usuario a la fuente de la información de donde se obtuvo la información.

6.4.3. Requerimientos principales no funcionales

RNF-1 – Precisión

El sistema debe proporcionar un 90% de respuestas precisas a las consultas de los usuarios. Lo que significa que deben ser correctas y relevantes para las preguntas.

RNF-2 – Contextualización

Al menos el 90% de las respuestas del sistema deben tener en cuenta el contexto de la consulta del usuario, lo que implica que las respuestas deben adaptarse a la información previamente proporcionada por el usuario y ser coherentes con el tema de la conversación.

RNF-3 – Usabilidad

La interfaz de usuario del sistema debe ser intuitiva y fácil de usar, siguiendo los principios de diseño centrado en el usuario. Además, el sistema debe proporcionar retroalimentación adecuada a los usuarios sobre el estado de sus acciones, como mensajes de error claros y confirmaciones cuando sea necesario.

RNF-3 – Rendimiento

El sistema debe responder a las consultas de los usuarios en un tiempo promedio no superior a 10 segundos, para una experiencia de usuario ágil.

RNF-4 – Extensibilidad

El sistema debe ser extensible para poder implementar nuevas tecnologías de inteligencia artificial, como *LLMs*, nuevos *frameworks*, bases de datos vectoriales, etc.

RNF-5 – Interoperabilidad

El sistema debe ser compatible con los navegadores Google Chrome, Firefox, Safari y Microsoft Edge, en sus versiones más recientes. Además, el sistema debe de adaptarse a distintas dimensiones de pantallas como dispositivos móviles, tabletas y computadoras,

adaptándose a diferentes resoluciones y tamaños de pantalla para garantizar una experiencia uniforme en una variedad de dispositivos.

6.5. Validación de problema y solución

6.5.1. *Beta tester*

Tal como se mencionó previamente, se dedicó un gran esfuerzo a la validación exhaustiva del problema, buscando comprenderlo en su profundidad. A medida que se lograba validar el problema con diversas organizaciones de interés, se tomó la decisión de involucrar a una de estas organizaciones para llevar a cabo una validación integral del proyecto. Con este fin, el equipo optó por contar con un *beta tester,* siendo el Banco Interamericano de Desarrollo (BID) la entidad seleccionada para desempeñar este papel.

El Banco Interamericano de Desarrollo (BID) es una institución financiera internacional que promueve el desarrollo económico y social en América Latina y el Caribe. Con sede en Washington D.C., el BID trabaja en estrecha colaboración con los países miembros y otras organizaciones para impulsar proyectos de desarrollo sostenible en áreas clave como infraestructura, educación, salud y medio ambiente.

Tiene un papel fundamental en el fomento del crecimiento económico, la reducción de la pobreza y la promoción de la igualdad de oportunidades en la región. A través de su apoyo financiero y técnico, el BID brinda asistencia a gobiernos, empresas y organizaciones de la sociedad civil para impulsar el progreso y mejorar la calidad de vida de millones de personas en América Latina y el Caribe.

El BID se destaca por su enfoque en el desarrollo sostenible, la innovación y la inclusión social. Trabaja en estrecha colaboración con sus países miembros para identificar desafíos y oportunidades, y diseñar soluciones adaptadas a las necesidades específicas de cada país. Además, el BID promueve la colaboración regional y el intercambio de conocimientos para potenciar el desarrollo y fortalecer la integración en América Latina y el Caribe.

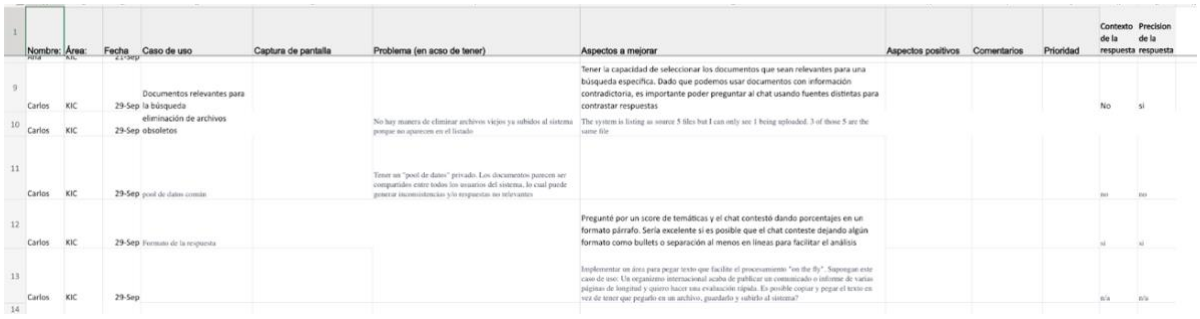
Como *beta tester* del proyecto Chat Your Data, busca mejorar su capacidad de acceder y utilizar de manera eficiente la información contenida en sus datos. A través de esta colaboración, el BID pretende optimizar la toma de decisiones informadas, agilizar los procesos internos y maximizar el impacto de sus iniciativas en la región.

6.5.2. Colaboración con el Banco Interamericano de Desarrollo

Inicialmente, la comunicación comenzó con Alan Kind, quien actuó como intermediario entre el equipo y los profesionales del BID interesados en el proyecto. Se gestionó un acuerdo entre el BID, el equipo de Chat Your Data y la Universidad ORT Uruguay para poder llevar a cabo esta colaboración. Las interacciones iniciales involucraron reuniones estratégicas en las que se presentó la idea del proyecto. Una vez que se alcanzó un acuerdo mutuo, se dio a conocer al equipo completo, que incluía a Mariana Gutiérrez, Lucía Latorre, Alan Kind, Ana Laura Sposito y Kyle Strand. A partir de ese momento, se les brindó acceso a nuestra plataforma, lo que les permitió comenzar las pruebas. Luego, una vez que la retroalimentación llegó a una etapa avanzada, se sumó al equipo Carlos Savignano, el *tech advisor* del equipo.

Durante el proceso de prueba, el equipo preparó un documento Excel, donde el BID mantuvo un registro detallado de sus experiencias. Este registro incluyó información sobre las acciones que realizaron, las respuestas que obtuvieron, capturas de pantalla, fechas y descripciones detalladas, y si la respuesta fue precisa y contextualizada. Esta documentación fue esencial para evaluar el rendimiento y la efectividad de la solución.

A continuación, se presentan algunas de las entradas redactadas por Carlos, donde comparte sus comentarios y opiniones. El equipo asignaba una prioridad y comentario a cada problema o funcionalidad pendiente, categorizándola según su prioridad. Esta clasificación era fundamental para mantener al equipo del BID informado sobre las áreas en las que se estaba trabajando y no tener que esperar hasta la próxima reunión para saber el estado actual de esos problemas.



	Nombre:	Área:	Fecha	Caso de uso	Captura de pantalla	Problema (en acso de tener)	Aspectos a mejorar	Aspectos positivos	Comentarios	Prioridad	Contexto de la respuesta	Precision de la respuesta
9	Carlos	KIC	29-Sep	Documentos relevantes para la búsqueda			Tener la capacidad de seleccionar los documentos que sean relevantes para una búsqueda específica. Dado que podemos usar documentos con información contradictoria, es importante poder preguntar al chat usando fuentes distintas para contrastar respuestas				No	si
10	Carlos	KIC	29-Sep	eliminación de archivos obsoletos		No hay manera de eliminar archivos viejos ya subidos al sistema porque no aparecen en el listado	The system is listing as source 5 files but I can only see 1 being uploaded. 3 of those 5 are the same file					
11	Carlos	KIC	29-Sep	pool de datos común		Tener un "pool de datos" privado. Los documentos parecen ser compartidos entre todos los usuarios del sistema, lo cual puede generar inconsistencias y/o respuestas no relevantes					no	no
12	Carlos	KIC	29-Sep	Formato de la respuesta			Pregunté por un score de temáticas y el chat contestó dando porcentajes en un formato párrafo. Sería excelente si es posible que el chat conteste dejando algún formato como bullets o separación al menos en líneas para facilitar el análisis				si	si
13	Carlos	KIC	29-Sep				Implementar un área para pegar texto que facilite el procesamiento "on the fly". Supongan este caso de uso: Un organismo internacional acaba de publicar un comunicado o informe de varias páginas de longitud y quiero hacer una evaluación rápida. Es posible copiar y pegar el texto en vez de tener que pegarlo en un archivo, guardarlo y subirlo al sistema?				n/a	n/a
14												

Ilustración 17- Excel de validación

Para mejorar aún más la retroalimentación, se implementó una función que permitía a los usuarios proporcionar comentarios directamente en la plataforma. Esto incluía la opción de

calificar una respuesta como buena o mala, y la posibilidad de dejar comentarios detallados en caso de una respuesta insatisfactoria. Esta función se diseñó pensando en la facilidad de uso y la utilidad para cualquier usuario que utilizara la plataforma.

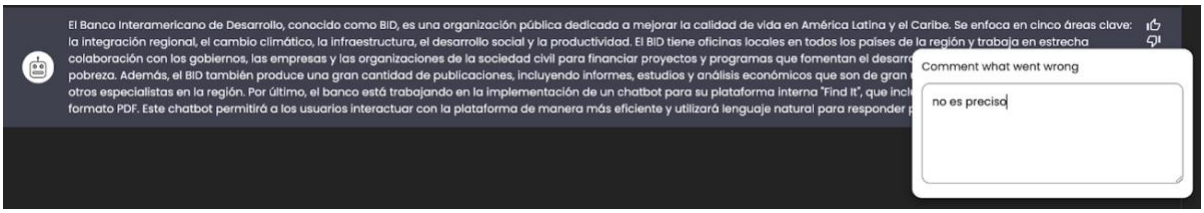


Ilustración 18- *Feedback* en el sistema

El equipo mantuvo reuniones programadas cada dos semanas con el equipo del BID, donde se discutieron sus experiencias y se recopiló información valiosa. Durante estas reuniones, se analizó la selección de datos a utilizar, priorizando documentos públicos como publicaciones. Con el tiempo, el equipo del BID expresó un creciente interés en explorar aún más las capacidades del producto, especialmente en lo que respecta a la generación de informes y nuevos reportes, destacando este aspecto como uno de los más atractivos de la solución.

Esta colaboración estrecha y continua con el *beta tester* desempeñó un papel fundamental en la mejora constante de la solución, garantizando que estuviera alineada con las necesidades y expectativas del mercado.

Ilustración 19- Reunión con el BID

7. Arquitectura

7.1. Introducción a la Arquitectura

En este capítulo, se explorará en detalle la arquitectura de Chat Your Data. La arquitectura es un componente crucial de cualquier sistema de *software*, y en este proyecto, desempeña un papel fundamental en la creación de una solución conversacional que permita a los usuarios interactuar de manera natural con los datos de su organización.

La importancia de la arquitectura de Chat Your Data radica en su capacidad para respaldar los atributos de calidad clave que son fundamentales para el éxito del proyecto. Estos atributos incluyen la precisión en las respuestas generadas, la contextualización de la información, la mantenibilidad a lo largo del tiempo, la extensibilidad para futuras mejoras, la usabilidad en la interacción con los usuarios, el rendimiento óptimo y la interoperabilidad con otros sistemas.

A lo largo de este capítulo, se desglosarán los componentes principales de la arquitectura y se explorarán las decisiones de diseño tomadas para abordar estos atributos de calidad. Las secciones posteriores proporcionarán diagramas de clases y descripciones más detalladas de los componentes clave, como el *frontend*, el *backend* y el módulo de inteligencia artificial PluginAI. Al mismo tiempo, se incluye un análisis comparativo de las decisiones que se tomaron en cuanto a las herramientas y tecnologías a utilizar.

Chat Your Data se compone de tres servicios principales: el *frontend*, el *backend* y el PluginAI. Cada uno de estos componentes desempeña un papel específico en el funcionamiento del sistema y se comunica de manera sincronizada para proporcionar una experiencia de usuario fluida y efectiva. Se profundizará en cada uno de estos servicios y sus componentes, examinando su arquitectura interna y su función en el proyecto. Además, se discutirá la implementación de una arquitectura basada en microservicios y cómo esta elección impacta en la escalabilidad y la modularidad del sistema.

La arquitectura de Chat Your Data es el pilar que sustenta la visión de un sistema conversacional de gestión de datos organizacionales, y este capítulo ofrecerá una visión completa de su estructura y diseño, así como de las decisiones clave que lo respaldan.

7.2. Arquitectura basada en microservicios (Arquitectura a alto nivel)

Dentro de la gran gama de patrones arquitectónicos disponibles en el dominio de la tecnología de la información, la arquitectura basada en microservicios ha emergido como una de las más resilientes y versátiles, especialmente en contextos que demandan escalabilidad y adaptabilidad. Tal es el caso de Chat Your Data, cuyo diseño arquitectónico se fundamenta precisamente en este enfoque.

Los microservicios son un estilo de arquitectura y un modo de programar *software* que se enfoca en dividir las aplicaciones en sus elementos más pequeños e independientes entre sí [53]. Cada uno de estos elementos o procesos es un microservicio, y a diferencia del enfoque tradicional y monolítico de las aplicaciones, los microservicios son elementos independientes que funcionan en conjunto para llevar a cabo las mismas tareas. Cada sección concentra sus esfuerzos en funciones concretas, optimizando el rendimiento y la gestión de recursos.

Este capítulo no solo busca explicar cómo se estructura Chat Your Data desde una perspectiva de microservicios, sino también por qué se eligió este enfoque particular. Es esencial entender que la elección no fue simplemente una consecuencia de seguir tendencias tecnológicas actuales, sino que fue una decisión estratégica para garantizar que el sistema responda efectivamente a los atributos de calidad establecidos desde el inicio del proyecto.

A continuación, se detallan cada uno de los microservicios que conforman el sistema. Cada componente, ya sea el *frontend*, el *backend* o el PluginAI, será desglosado para ofrecer una clara comprensión de su papel, responsabilidades y la manera en la que se interconecta con el resto del ecosistema. Además, se pondrá especial énfasis en las ventajas que esta estructura aporta en términos de escalabilidad, mantenibilidad y flexibilidad.

7.2.1. Justificación de la arquitectura basada en microservicios

La elección de una arquitectura resulta de un análisis detallado que busca equilibrar las necesidades inmediatas del sistema con proyecciones de crecimiento a futuro. En el caso de Chat Your Data, la elección de utilizar una arquitectura basada en microservicios surgió de un análisis estratégico. Las razones clave detrás de esta elección se detallan a continuación, ilustrando cómo cada una contribuye al cumplimiento de los atributos de calidad y a la visión a largo plazo del sistema:

- **Escalabilidad**: En un contexto digital que cambia rápidamente, la adaptabilidad es esencial. Los microservicios permiten que cada componente se escale de forma independiente, adaptándose a las demandas fluctuantes. Por ejemplo, si Chat Your Data ve un aumento en las solicitudes dirigidas al PluginAI, este módulo se puede expandir sin afectar a otros.
- **Mantenibilidad**: En un marco modular, las modificaciones en un componente tienen menos probabilidades de afectar a otros. Esta independencia facilita la implementación de actualizaciones o correcciones, algo esencial para un sistema que se anticipa en constante evolución.
- **Flexibilidad**: La arquitectura de microservicios respeta la elección de herramientas, lenguajes o *frameworks* que se consideren más apropiados para cada función, asegurando la optimización de cada componente.
- **Desarrollo paralelo**: Esta estructura promueve el desarrollo simultáneo. Varios equipos pueden trabajar en diferentes componentes al mismo tiempo, acelerando la evolución y entrega del sistema.
- **Precisión y contextualización**: Localizando la inteligencia artificial en PluginAI y adoptando modelos avanzados, Chat Your Data proporciona respuestas no solo precisas, sino también contextualmente relevantes.
- **Performance**: La elección de gRPC para la comunicación entre el *backend* y el PluginAI y el uso de Redis y NoSQL garantizan un flujo de datos rápido y fiable.
- **Interoperabilidad y usabilidad:** Una clara separación entre el *frontend* y el *backend* facilita una experiencia de usuario intuitiva. Esta división también asegura que Chat Your Data pueda integrarse con una variedad de clientes y plataformas.

Por lo tanto, este enfoque basado en microservicios no solo se ocupa de las demandas actuales de Chat Your Data, sino que también posiciona al sistema para enfrentar desafíos y oportunidades futuros.

7.2.2. Arquitectura a alto nivel

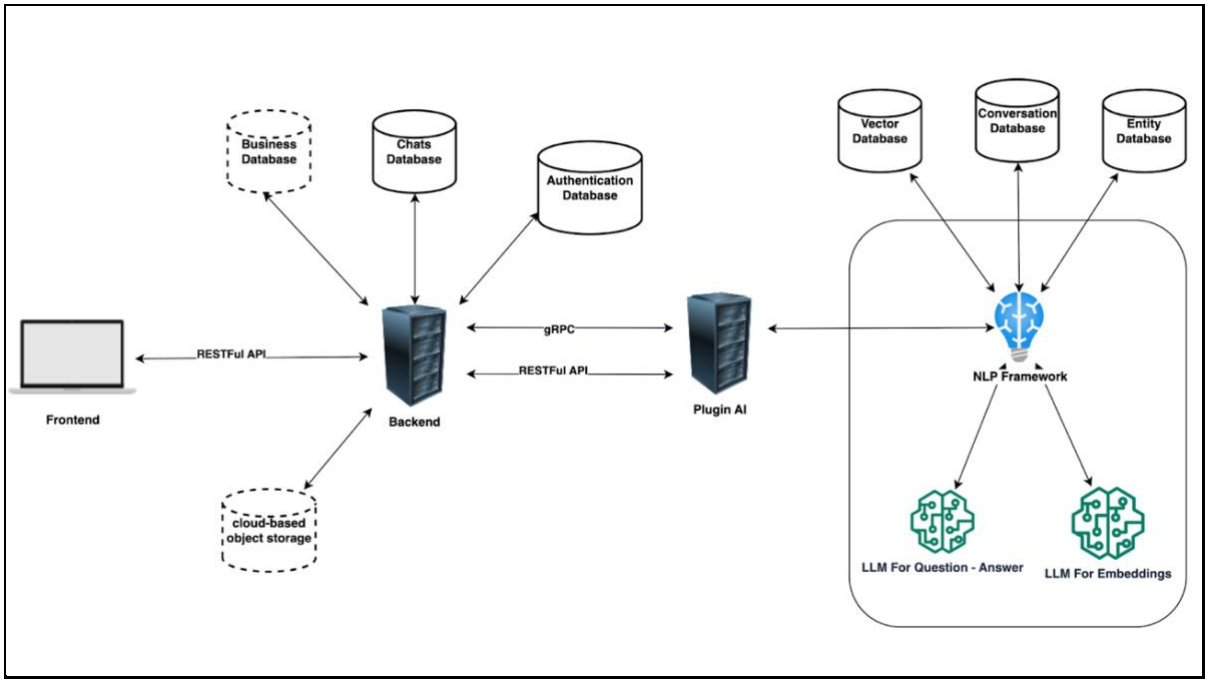


Ilustración 20- Arquitectura a alto nivel

Este capítulo se dedica a analizar y describir la arquitectura de alto nivel de la plataforma Chat Your Data. La estructura arquitectónica propuesta destaca por su modularidad, escalabilidad y enfoque orientado al usuario. A continuación, se desglosarán los componentes clave y se explicará su función dentro del ecosistema general del sistema.

Componentes y Descripción

- ***Frontend***: Representa la interfaz gráfica de usuario (GUI) de la plataforma Chat Your Data. Esta capa es crucial para proporcionar una buena experiencia de usuario, ofreciendo puntos de interacción claramente definidos y optimizados.
- ***Backend***: Funciona como la lógica de negocio central y el orquestador entre las distintas capas y componentes. Esta capa asume la responsabilidad de procesar las peticiones del usuario y coordinar con las bases de datos y otros sistemas subyacentes.
- ***Business Database*:** Aunque aún no implementada, está diseñada para consolidar información pertinente relacionada con organizaciones y usuarios, sirviendo como un recurso centralizado para tales datos.

- ***Cloud-based object storage*:** Previsto como el repositorio principal para documentos cargados por los usuarios. Esta solución en la nube promete escalabilidad y eficiencia en el almacenamiento de datos a gran escala. Aún no está implementada.
- ***Chats Database*:** Almacena metadatos y detalles de todas las sesiones de *chat*, incluyendo interacciones específicas y *feedback* del usuario, constituyendo una fuente invaluable para análisis y mejoras futuras.
- ***Authentication Database*:** Es una infraestructura crítica de seguridad que contiene credenciales y *tokens* de autenticación, garantizando la integridad y confidencialidad del sistema.
- **PluginAI:** Esta entidad encapsula la funcionalidad relacionada con la inteligencia artificial. Su nomenclatura "*Plugin*" refleja un diseño modular que permite adaptabilidad a las innovaciones futuras en el dominio de la inteligencia artificial.
- **NLP *Framework*:** Proporciona un conjunto cohesivo de herramientas y protocolos diseñados específicamente para facilitar la comunicación y la integración con LLM y bases de datos vectoriales.
- ***Vector Database*:** Facilita el almacenamiento eficiente de representaciones vectoriales (*embeddings*) de documentos, optimizando la recuperación y la consulta de tales datos.
- ***Conversation Database*:** Mantiene registros detallados de conversaciones pasadas, permitiendo que los LLMs accedan a contextos previos y generen respuestas más coherentes.
- ***Entity Database*:** Conserva entidades identificables derivadas de las interacciones, proporcionando una rápida referencia para respuestas contextualizadas en conversaciones futuras.
- **LLM *Question Answering y* LLM *Embedding***: LLM que desempeñan roles específicos en la comunicación y procesamiento de datos, respectivamente.

Existen numerosas bases de datos que sirven a distintos fines dentro de Chat Your Data. La división de estas sirve para poder utilizar la mejor tecnología para cada una. Más adelante, se detallará qué tecnología utilizan.

7.2.3. Separación en tres componentes: *frontend*, *backend* y PluginAI

Como ya se mencionó anteriormente, el diseño arquitectónico de Chat Your Data se caracteriza por una clara separación en tres componentes fundamentales: el *frontend*, el *backend* y el PluginAI. Esta distribución no solo refleja la adaptación al paradigma de microservicios, sino

que también facilita una gestión eficiente del sistema, al dividir las responsabilidades y funciones del *software*. Además, cada uno de estos componentes se aloja estratégicamente en repositorios separados, para así cumplir con las buenas prácticas de arquitectura de *software*.

Cada componente contiene sus propios archivos ".env". Estos archivos se utilizan para configurar una aplicación proporcionando variables específicas del entorno. Estos archivos son de texto plano y suelen contener pares claves-valor. Dentro de estos archivos se alberga información de API_KEYS como las rutas entre *frontend*, *backend* y PluginAI.

7.2.3.1. Frontend

- **Responsabilidad principal:** Actúa como la interfaz de interacción entre los usuarios y Chat Your Data. Es el punto de contacto inicial, donde se capturan las solicitudes de los usuarios y se presentan las respuestas generadas por el sistema.
- **Características clave:** Está diseñado para ser intuitivo y amigable, ofreciendo una experiencia de usuario fluida y efectiva. Su diseño adaptable permite que los usuarios accedan desde diferentes dispositivos y plataformas sin comprometer la funcionalidad o la estética.
- **Comunicación:** Se comunica directamente con el b*ackend*, enviando solicitudes de los usuarios y recibiendo datos procesados para su visualización.

7.2.3.2. Backend

- **Responsabilidad principal:** Funciona como el mediador entre el f*rontend* y el PluginAI. Gestionando funciones como autenticación de usuarios, almacenamiento de datos y procesamiento de solicitudes, es el núcleo operativo de Chat Your Data.
- **Características clave:** Además de facilitar la interacción entre el f*rontend* y el PluginAI, el b*ackend* es responsable de mantener la integridad de los datos, garantizar la seguridad y administrar la lógica de negocio.
- **Comunicación:** Mientras que interactúa con el *frontend* a través de una *REST API*, se comunica con el PluginAI principalmente mediante *gRPC*, lo que garantiza una transmisión eficiente y segura de datos.

7.2.3.3. PluginAI

- **Responsabilidad principal:** Este componente es el cerebro del sistema. Procesa las solicitudes de los usuarios, analiza datos y genera respuestas pertinentes y contextualizadas.
- **Características clave:** Se compone de diversos algoritmos y modelos de aprendizaje automático, lo que permite a Chat Your Data entender y responder de manera efectiva a las consultas complejas de los usuarios.
- **Comunicación:** Recibe las solicitudes procesadas desde el *backend*, a las cuales responde después de procesar la información y determinar la mejor respuesta.

7.2.4. Descripción de la comunicación entre componentes

En el amplio ecosistema de las aplicaciones modernas, la interacción entre sus distintos componentes es el vínculo que une todo el sistema, haciendo que funcione como una entidad cohesiva. La arquitectura de Chat Your Data no es una excepción. Su eficacia y eficiencia están ligadas a cómo se lleva a cabo la comunicación entre los diferentes módulos y componentes del sistema. Entender estas interacciones es crucial, no solo para garantizar el funcionamiento óptimo en el presente, sino también para planificar y adaptarse a las futuras demandas y expansiones.

A lo largo de este capítulo, se profundizará en los mecanismos de comunicación que adopta Chat Your Data, abordando desde los protocolos empleados hasta las especificidades de su implementación. Para facilitar una comprensión visual, se presentarán diagramas de secuencia que ilustrarán el flujo de información entre los componentes en diferentes escenarios operativos.

La elección de los protocolos y mecanismos de comunicación no es aleatoria, y detrás de cada decisión hay una serie de consideraciones técnicas y prácticas que buscan optimizar el rendimiento, la seguridad y la escalabilidad del sistema. En este sentido, dentro del anexo 13.18 Protocolos de Comunicación se discutirán dos protocolos clave que han sido adoptados: API REST y gRPC.

El sistema generalmente adopta un enfoque de comunicación asíncrona. Esto significa que las solicitudes enviadas no requieren una respuesta inmediata y pueden esperar una respuesta sin bloquear otras operaciones. Esta naturaleza asíncrona suele ser más adaptable y escalable,

especialmente para sistemas que manejan múltiples tareas o que esperan grandes volúmenes de tráfico.

7.2.4.1. Creación de un *prompt*

Cuando se da el suceso de un *prompt* ocurren ciertas interacciones entre los diferentes componentes. A continuación, se presenta un diagrama de secuencia en donde se puede apreciar en un alto nivel que es lo que sucede en cada componente.

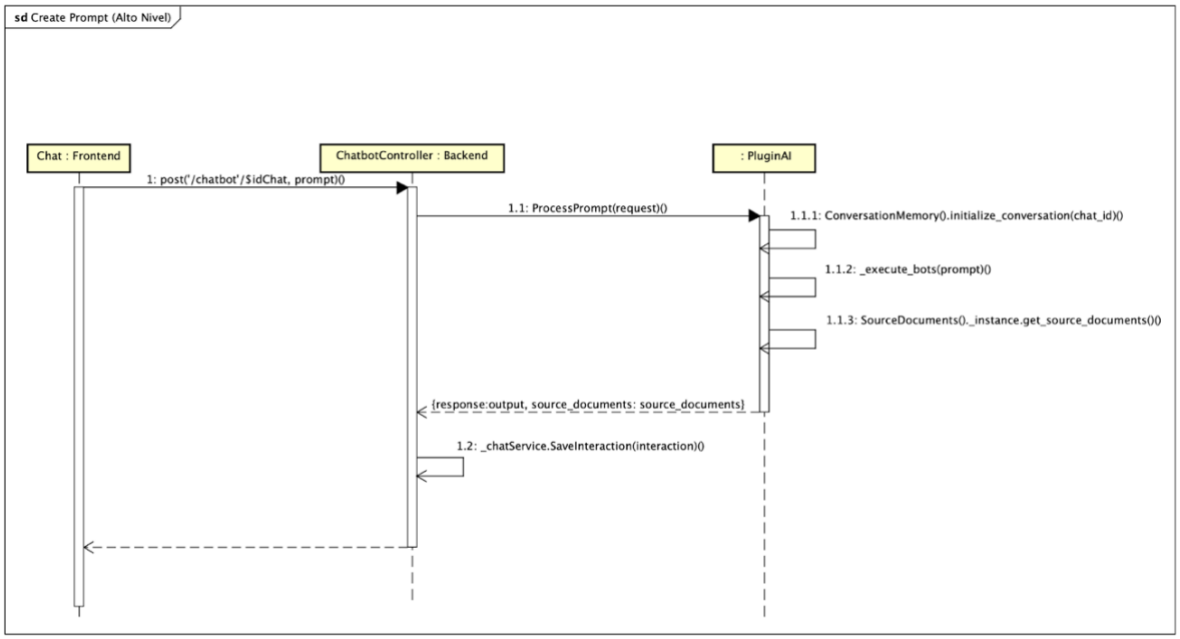


Ilustración 21- Diagrama de secuencia de *Create Prompt*

Para poder solicitar un *prompt* al sistema de Chat Your Data los clientes *(frontend)* deben de cumplir con un *token* necesario para poder realizar consultas. Si bien se hablará más adelante a modo de resumen, un *token* es un mecanismo de seguridad que se utiliza en el desarrollo de *software* para autenticar y autorizar solicitudes de manera segura y confiable.

Además del *token* el *frontend* debe proporcionar el *prompt* y un identificador del *chat* (*"chatId"*). Este identificador lo puede obtener solicitando al *backend* mediante otra consulta. Una vez que se valida el *token*, el cual es una tarea realizada por el *backend*, el *backend* se comunica mediante gRPC con PluginAI para enviarle el *prompt* que hizo el usuario junto con el "*chatId"*. El funcionamiento de la comunicación mediante gRPC se encuentra explicada detalladamente en el anexo 13.18.2 gRPC.

Dentro de PluginAI se ejecutan varias funciones para poder dar la respuesta. Luego de que PluginAI genere la respuesta, esta información es enviada por gRPC al *backend*, donde este se encarga de guardar la interacción, para que el usuario pueda tener su historial de conversaciones.

Finalmente, la respuesta es enviada al *frontend* junto con la *completion* y los *source_documents*.

7.2.4.2. Subida de archivos

Cuando se da la subida también es necesario contar con un *token* para poder realizar la solicitud. Hay un *endpoint* en el *backend* al cual se le pueden enviar varios archivos para que Chat Your Data los pueda entender. Este proceso de entender es conocido técnicamente como el proceso de e*mbedding.*

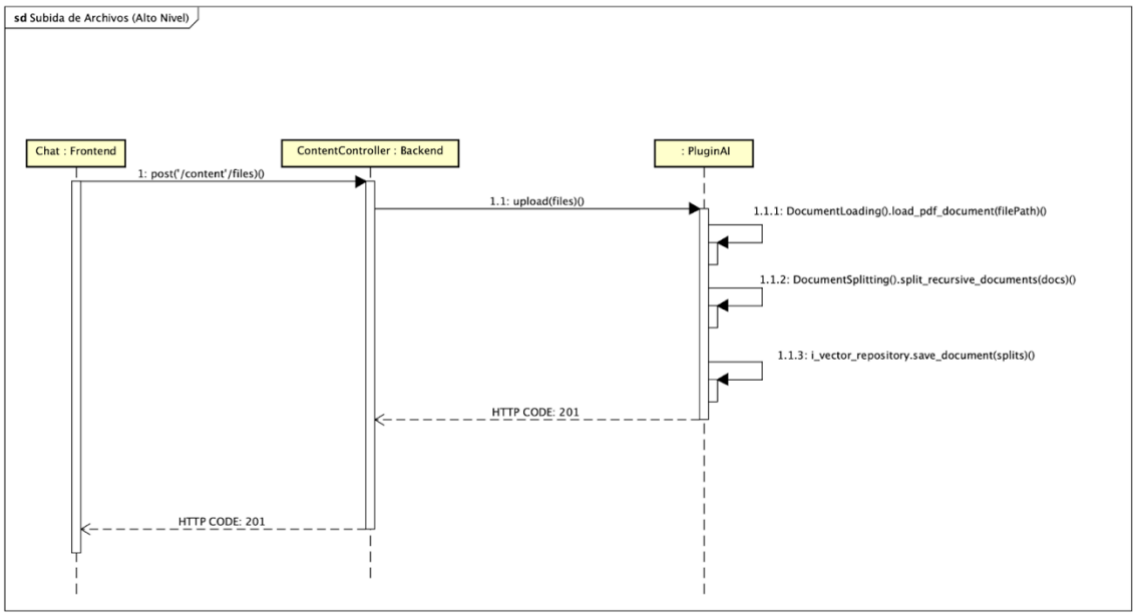


Ilustración 22- Diagrama de secuencia de subida de archivos

Lo que va sucediendo dentro de PluginAI son las siguientes funciones:

- ***DocumentLoading().load_pdf_document(file_path)*:** Aquí está extrayendo todo el texto del PDF subido.
- ***DocumentSplitting().split_recursive_documents(docs)*:** Este proceso genera diferentes "*chunks"* para que sean guardados luego en la base de datos vectorial.
- ***i_vector_respository.save_documents(splits)*:** Aquí se utiliza la interfaz implementada por la base de datos ChromaDB en donde los *chunks* son guardados en la base de datos vectorial.

7.2.4.3. Envío de feedback

Algo primordial y necesario para que Chat Your Data mejore sus respuestas es el poder generar una base de datos de aquellas respuestas que fueron buenas y malas. Es decir, que a partir de un *prompt* que hizo un usuario Chat Your Data generó una *completion* la cual el usuario tiene la posibilidad de marcar como buena y mala. Esto resulta necesario para poder realizar el proceso de *fine-tuning*. Este concepto se detalla en el anexo 13.6 Fine-Tuning.

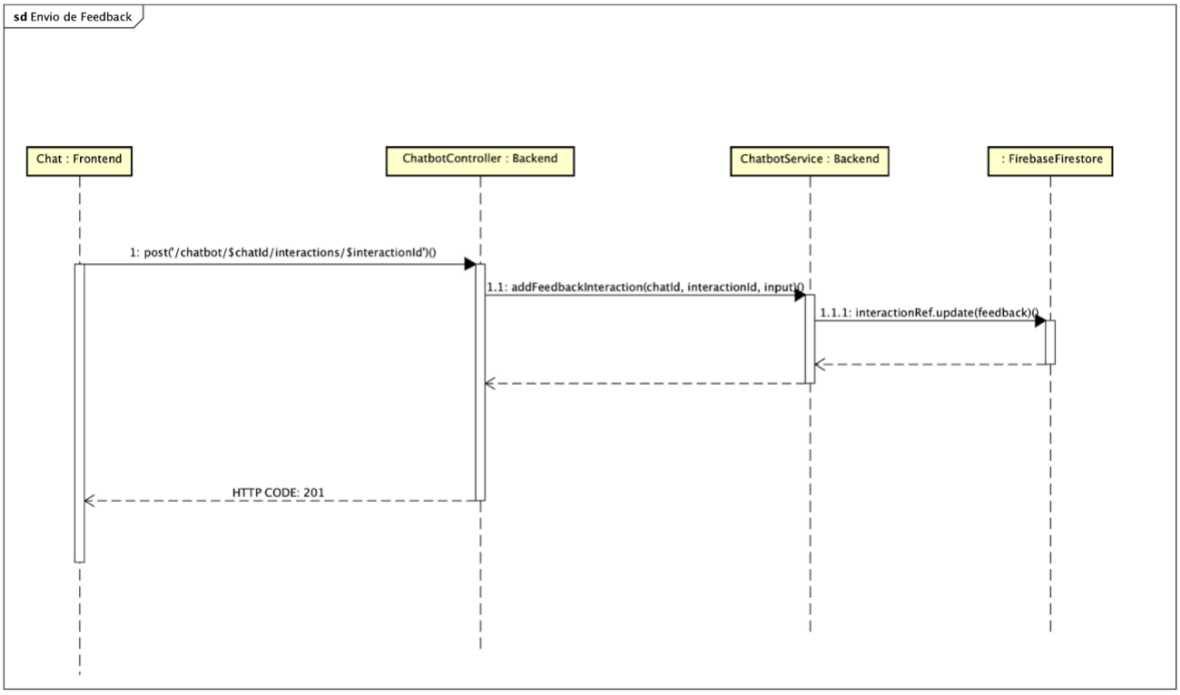


Ilustración 23- Diagrama de secuencia de envio de *feedback*

Para el envío del *feedback* de una interacción entre el usuario y Chat Your Data se tiene que tener el identificador del *chat* y el identificador de la interacción. En este proceso no participa PluginAI dado que no es su responsabilidad y no es necesaria su intervención.

Como se verá en los próximos capítulos, se está utilizando "Firebase Firestore" para el alojamiento del guardado de los *chats*.

7.2.4.4. Evolución y adaptabilidad

Un factor crítico para la longevidad y la adaptabilidad de cualquier sistema es su capacidad para evolucionar. Chat Your Data ha implementado interfaces, lo que facilita la adaptación a cambios o actualizaciones. Estas interfaces actúan como contratos que permiten modificar la implementación subyacente sin afectar otros componentes que dependen de ella.

7.2.4.5. Conclusión

La arquitectura de Chat Your Data ha sido diseñada pensando en la comunicación efectiva entre componentes. Desde la elección de protocolos hasta mecanismos de autenticación y adaptabilidad, cada decisión refleja un compromiso con las buenas prácticas arquitectónicas y con la entrega de un servicio de calidad a los usuarios. A medida que Chat Your Data continúa evolucionando, es esencial mantener y mejorar estas prácticas de comunicación para asegurar su posición destacada en el mundo de las soluciones conversacionales.

7.3. Plataforma, tecnologías y decisiones de diseño

En esta sección, se profundizará en detalle en la elección de tecnologías para la arquitectura planteada en los capítulos anteriores. Cada elección se basa en criterios específicos para garantizar los objetivos de Chat Your Data y realizar el proyecto dentro del tiempo establecido.

Para esta sección el equipo discutió entre sí y mantuvo conversaciones con referentes en el tema, además de utilizar la tecnología de ChatGPT para que indicara cuáles podrían llegar a ser las tecnologías a utilizar en base al equipo, tiempo y alcance.

7.3.1. Elección de tecnologías y justificación

7.3.1.1. Frontend (React)

Para la elección de la tecnología adecuada para el *frontend* de la plataforma Chat Your Data, se consideraron diversas opciones, examinando sus ventajas, desventajas y peculiaridades. A continuación, se muestra el proceso de toma de decisiones que terminó en la elección de React.

Comparativa de tecnologías *frontend*

Lenguaje o *Framework*	**Ventajas**	**Desventajas**	**Aspectos a Considerar**
React	-Gran comunidad. -Alto rendimiento. -Arquitectura basada en componentes.	-Curva de aprendizaje más profunda.	Elegir las bibliotecas adecuadas para gestión de estado y UI.

Vue.js	-Curva de aprendizaje sencilla. -Alto rendimiento. -Flexibilidad y modularidad.	-Comunidad más pequeña que React.	Elegir las bibliotecas adecuadas para gestión de estado y UI.
Angular	-Framework completo. -Tipado fuerte. -Gran comunidad.	-Curva de aprendizaje pronunciada. -Mas código repetitivo.	Es necesario adherirse a las mejores prácticas de Angular.

Tabla 1- Comparativa de tecnologías *frontend*

Justificación de la Elección

La elección de React como la tecnología principal para el desarrollo del *frontend* se basó en varios factores clave:

- **Experiencia Previa del Equipo:** Dos miembros del equipo ya tenían experiencia trabajando con React, lo que supone una ventaja en términos de velocidad de desarrollo y familiaridad con el *framework*.
- **Gran Comunidad:** React, desarrollado y mantenido por Facebook, cuenta con una gran comunidad que ofrece un inmenso repositorio de recursos, tutoriales y soluciones a problemas comunes. Esta fortaleza comunitaria promete respuestas rápidas a desafíos técnicos y acceso a una variedad de bibliotecas complementarias.
- **Arquitectura Basada en Componentes:** React adopta un enfoque modular mediante una arquitectura basada en componentes. Esto permite una reutilización eficiente del código, mejora la mantenibilidad y facilita la creación de interfaces de usuario escalables y cohesivas.
- **Alto Rendimiento:** Gracias al *Virtual DOM* (*DOM virtual*) de React, las actualizaciones y renderizaciones de la interfaz son optimizadas, lo que se traduce en un alto rendimiento y una experiencia de usuario fluida.
- **Flexibilidad:** React no es opinativo y otorga libertad para elegir bibliotecas y estructuras que mejor se adapten a las necesidades del proyecto, desde la gestión de estado hasta el diseño de la interfaz.

Conclusión

Teniendo en cuenta las ventajas que ofrece, el conocimiento previo del equipo y la flexibilidad para adaptarse a las demandas cambiantes del proyecto, React fue la opción más adecuada para el *frontend* de Chat Your Data. Esta decisión, respaldada por la solidez y popularidad de React en la industria, busca garantizar un desarrollo ágil, un rendimiento optimizado y una experiencia de usuario excepcional.

7.3.1.2. Backend (Nest.js)

Para determinar la tecnología más apropiada para el *backend* de la plataforma Chat Your Data, se evaluaron diversas opciones, considerando sus ventajas, desventajas y características específicas. A continuación, se presenta el análisis que condujo a la elección de Nest.js como solución de *backend*.

Comparativa de tecnologías *backend*

Lenguaje o *Framework*	**Ventajas**	**Desventajas**	**Aspectos a Considerar**
Nest.js	-Extensión de Node.js. -Integración con TypeScript. -Arquitectura modular.	-Relativa juventud comparado con otros *frameworks*. -Requiere familiaridad con los decoradores.	Aprovecha la amplia gama de módulos y herramientas.
Elixir (Phoenix)	-Alto rendimiento. -Tolerancia a fallos. -Soporte de concurrencia.	-Comunidad más pequeña.	Requiere comprensión de programación funcional y el modelo Actor.
Express	-JavaScript en el backend. -Amplia comunidad. -Buen rendimiento.	-Monohilo (aunque basado en eventos). -Problemas con callbacks.	Considerar el uso de TypeScript para un tipado fuerte.

Golang	-Alto rendimiento. -Soporte de concurrencia. -Tipado estático.	-Comunidad más pequeña. -Librería de terceros limitados. -Alta curva de aprendizaje.	Requiere comprensión de características idiosincrasias únicas de Go.

Tabla 2- Comparativa de tecnologías *backend*

Justificación de la Elección

Después de considerar diversas tecnologías, se decidió optar por Nest.js, una extensión de *Node.js*, por las siguientes razones clave:

- **Experiencia Interna:** Un miembro del equipo trabaja diariamente con Nest.js, lo que representa una ventaja significativa en términos de velocidad de desarrollo y familiaridad con la tecnología.
- **Arquitectura Modular:** Nest.js ofrece una arquitectura modular que facilita la organización y la escalabilidad del código. Esta característica es especialmente valiosa para proyectos que pueden crecer en complejidad con el tiempo.
- **Integración de TypeScript:** Nest.js se integra perfectamente con TypeScript, proporcionando tipado fuerte, lo que puede ayudar a prevenir errores en tiempo de ejecución y mejorar la calidad del código.
- **Decoradores y Controladores:** Uno de los rasgos distintivos de Nest.js son sus decoradores y controladores, que permiten una organización clara del código y una definición sencilla de rutas y endpoints.
- **Amplio Ecosistema:** Nest.js cuenta con un conjunto de módulos y herramientas que pueden ser fácilmente integrados, facilitando la incorporación de funcionalidades adicionales como la autenticación, la autorización, y la interacción con bases de datos, entre otras.

Conclusión

Basándose en la experiencia interna y las características intrínsecas de Nest.js, se llegó a la conclusión de que este *framework* era la opción más adecuada para el desarrollo del *backend* de Chat Your Data. Esta decisión se fundamenta en la combinación de flexibilidad,

escalabilidad y eficiencia que ofrece Nest.js, prometiendo un desarrollo robusto y ágil que se alinee con los objetivos del proyecto.

7.3.1.3. PluginAI (Python)

Dentro del espectro de tecnologías disponibles para el desarrollo de sistemas con capacidades de inteligencia artificial, se realizó un estudio comparativo riguroso para identificar la herramienta más adecuada para la construcción de PluginAI. La siguiente sección detalla este análisis y la justificación detrás de nuestra elección.

Comparativa de tecnologías para PluginAI

Lenguaje o *Framework*	**Ventajas**	**Desventajas**	**Aspectos a Considerar**
Python	-Versátil y ampliamente utilizado. -Rico ecosistema de bibliotecas y herramientas. -Fácil de aprender y comprender. -Adecuado para crear plugins modulares y reutilizables.	-Tiempo de ejecución más lento en comparación con lenguajes compilados.	Python es un lenguaje popular y versátil, adecuado para una amplia gama de tareas, con un ecosistema extenso que lo posiciona fuertemente para la construcción de sistema de plugins.
JavaScript (Node.js)	-Altamente popular. -Rico ecosistema de bibliotecas y herramientas. -Fácil de aprender para aquellos familiarizados con JavaScript en el frontend. -Asíncrono y orientado a eventos.	-Monohilo, puede no ser optimo para tareas intensivas de CPU.	Node.js es una elección popular para sistemas de backend. Su naturaleza orientada a eventos lo hace apto para un sistema de plugins, pero hay que tener precaución con tareas intensivas de CPU.

Golang	-Alto rendimiento gracias a su naturaleza compilada. -Soporte para concurrencia con *goroutines*.	-Curva de aprendizaje mas pronunciada.	Golang destaca por su rendimiento y soporte para la concurrencia.
Rust	-Excelente rendimiento y seguridad de memoria. -Soporte para la concurrencia.	-Curva de aprendizaje pronunciada.	Rust es potente en términos de rendimiento y seguridad.

Tabla 3- Comparativa de tecnologías para PluginAI

Justificación de la elección de Python para PluginAI

La decisión de optar por Python como lenguaje base para PluginAI se basó en varios factores críticos:

- **Especialización en Inteligencia Artificial:** Python se ha posicionado como el lenguaje de programación por excelencia en el ámbito de la inteligencia artificial. Las bibliotecas y herramientas más reconocidas en IA, como TensorFlow, PyTorch y Scikit-Learn, ofrecen soporte extensivo para Python, lo que garantiza una implementación fluida de diversas tareas relacionadas con IA.
- **Simplicidad y Legibilidad:** El diseño de la sintaxis de Python prioriza la claridad, lo que facilita la lectura y comprensión del código. Esta característica es esencial para garantizar que el componente de plugins sea mantenible y extensible a lo largo del tiempo.
- **Adaptabilidad:** Python se puede integrar fácilmente con varios sistemas y bibliotecas. Su naturaleza dinámica y su extensa biblioteca estándar son ideales para crear *plugins* que funcionen de manera fluida con diferentes formatos de archivos y herramientas.
- **Comunidad Activa:** La comunidad de desarrolladores de Python es grande y activa, lo que asegura una evolución constante del lenguaje y su ecosistema. Este soporte comunitario se traduce en una abundancia de recursos, lo que facilita el aprendizaje y adaptación a las necesidades del proyecto.

Si bien el equipo no contaba con experiencia previa en Python, se reconoció que enfrentar este desafío valdría la pena debido a los beneficios inherentes del lenguaje. En términos de desarrollo de sistemas con capacidades de inteligencia artificial, Python se destaca como una

elección insuperable, y se decidió invertir tiempo y recursos en familiarizarse con él para garantizar el éxito de PluginAI.

7.3.1.4. NLP Framework (LangChain)

El equipo necesitaba de un *framework* que le ayudara en su funcionalidad principal, que consiste en poder generar *embeddings* a partir de un documento, y que luego un *chatbot* pudiese obtener información de los documentos a partir del *prompt* de un usuario.

En la primera etapa de investigación, surgieron herramientas como Haystack y LangChain. Con este marco de tecnología se encontraba el equipo ante la elección de las dos tecnológicas:

Aspecto	LangChain	Haystack
Facilidad de Uso	Alta	Moderada
Documentación	Completa	Amplia
Comunidad de Desarrolladores	Menos conocido	Creciente

Tabla 4- Análisis de herramientas

Se había elegido Haystack como una posible alternativa. Sin embargo, luego de estar analizando más a futuro las funcionalidades de Chat Your Data, Haystack no iba a brindar el set de herramientas requerido.

Es por ello que **LangChain** se destacó entre las opciones evaluadas por sus capacidades avanzadas en áreas cruciales como la construcción de *chatbots* y la gestión de preguntas y respuestas (*GQA*). Además, presento una característica distintiva y valiosa: la habilidad de encadenar complejos flujos de trabajo de manera única, lo cual se alinea perfectamente con los objetivos de automatización y eficiencia de Chat Your Data. Dentro del anexo 13.19 Capacidades de Langchain se encuentran las ventajas de LangChain y sus casos de uso.

7.3.1.5. LLM Question answering (GPT-3 y GPT-4)

La selección de los LLM para el proyecto se llevó a cabo con un análisis cuidadoso de las opciones disponibles en el mercado. Aunque la elección de un LLM no era una prioridad inicial para el proyecto, el equipo reconoció la importancia de elegir una tecnología robusta y confiable para garantizar un rendimiento óptimo.

Las opciones consideradas por el equipo incluyeron GPT-3, PaLM y LLaMA. Sin embargo, los modelos GPT-3 y GPT-4 de OpenAI se destacaron por varias razones. En primer lugar, la reputación y el reconocimiento de OpenAI en el ámbito de la inteligencia artificial brindaron una confianza adicional al equipo. Además, la popularidad de estos modelos, especialmente a raíz del auge de soluciones como ChatGPT, sugirió una amplia aceptación y confiabilidad dentro de la comunidad.

Otro factor decisivo fue que los modelos de OpenAI ya eran ampliamente utilizados en las documentaciones y ejemplos proporcionados por LangChain. Esta familiaridad simplificó la decisión y permitió una integración más fluida con el proyecto existente.

A pesar de la decisión de adoptar los modelos de OpenAI en esta etapa del proyecto, el equipo mantiene una postura abierta hacia la incorporación de otros LLM en el futuro. Reconociendo la naturaleza dinámica del campo de la inteligencia artificial, el equipo está comprometido con la revisión y adaptación constantes para garantizar que el proyecto siga siendo relevante y esté alineado con los avances tecnológicos más recientes.

7.3.1.6. LLM Embedding (Text-embedding-ada-002)

Los LLM *embedding* se centran en representar palabras, frases o inclusos párrafos en un espacio vectorial.

Text-embedding-ada-002 es un modelo de *embedding* nuevo y mejorado desarrollado por OpenAI. Tiene varias ventajas en comparación con sus predecesores y otros modelos de *embedding*, lo que llevo al equipo a elegirlo como modelo.

1. Rendimiento mejorado: Supera a todos los antiguos modelos de *embedding* en tareas como la búsqueda de texto, búsqueda de código, y la similitud de frases.
2. Contexto más largo: La longitud del contexto del modelo se incrementó, lo que lo hace ideal para trabajar con documentos largos.
3. Precio reducido: El precio de los nuevos se ha reducido en comparación con los antiguos modelos. Esto lo convierte en una opción más asequible sin comprometer el rendimiento.

7.3.1.7. Chats Database y Authentication Database (Firebase)

Dentro de las decisiones tecnológicas necesarias para la creación de un MVP robusto y eficiente, la selección de sistemas de autenticación y bases de datos representa un componente

crucial. Estas elecciones no solo influyen en la funcionalidad y experiencia del usuario, sino que también determinan la escalabilidad y adaptabilidad del proyecto en etapas posteriores. En el caso de este proyecto, aunque se consideraron diversas opciones, Firebase destacó por encima del resto.

Para poder autenticar usuarios, JWT (JSON Web tokens) surgió como una alternativa. A pesar de que JWT goza de reconocimiento y confianza en la industria por su robustez, adoptarlo habría implicado la necesidad de establecer una base de datos adicional a desplegar.

En contraste, Firebase era una solución más consolidada, al fusionar los servicios de autenticación y almacenamiento de datos bajo una misma plataforma. Esta aproximación no solo ofrecía una implementación más directa, sino que prometía una experiencia de usuario uniforme y segura. Además, la decisión fue respaldada por la experiencia previa con Firebase que tenían varios miembros del equipo.

Al tener en cuenta estas consideraciones, el equipo tomó la decisión estratégica de adoptar Firebase, buscando la cohesión tecnológica, la optimización en el desarrollo y la capacidad de adaptarse a los cambios y necesidades futuras del proyecto.

El sistema de autenticación de Firebase se distingue por su intuición y eficacia. Está diseñado de forma que la configuración y gestión de sistemas de inicio de sesión resulta sencilla, y lo más relevante, cumple con estándares de seguridad de alto nivel, garantizando a los usuarios una experiencia impecable y protegida.

Una vez resuelta la autenticación, apareció otro desafío: gestionar y almacenar las interacciones entre usuarios e inteligencia artificial. Estas conversaciones no son solo valiosas para ofrecer un registro histórico al usuario, sino que constituyen una fuente de información crucial para el perfeccionamiento del sistema a través del *fine-tuning*.

Ante esta necesidad, y en línea con la elección previa, el equipo se decidió por Firebase Firestore. Su capacidad para proporcionar una base de datos NoSQL de alta eficiencia y su perfecta integración con el sistema de autenticación de Firebase, ofrecieron un entorno óptimo y seguro para registrar y analizar las interacciones.

Los beneficios que Firebase aportó al proyecto fueron claros y tangibles:

1. **Desarrollo ágil**: Al ser una solución integrada, Firebase acelera el tiempo de desarrollo, facilitando la rápida instauración de funcionalidades vitales.
2. **Escalabilidad**: La flexibilidad de Firebase garantiza su adaptabilidad conforme el proyecto expande sus horizontes.
3. **Reducción de *Overhead***: Usar Firebase eliminó la necesidad de gestionar servidores adicionales, permitiendo al equipo centrarse en el núcleo del producto.
4. **Integración holística**: Tener autenticación y gestión de datos en un solo lugar elimina problemas de compatibilidad y unifica el flujo de trabajo.

Con todo esto, la elección de Firebase no solo coincidió con la experiencia del equipo, sino que también representó una visión estratégica de desarrollo. Este enfoque se centró en superar desafíos y brindar un valor óptimo al usuario final.

7.3.1.8. Conversation Database y Entity Database (Redis)

Al diseñar sistemas de *chat* eficientes, mantener un registro contextualizado de las conversaciones es esencial. Para el proyecto Chat Your Data, esto se tradujo en la necesidad de almacenar *entities* e interacciones entre el usuario y el sistema. En el capítulo del componente de PluginAI se explicará en detalle cómo funciona el sistema de *entities*.

En la fase de selección tecnológica, el equipo se enfrentó a una difícil decisión: optar por *Redis* o *SQLite*. Ambas herramientas ofrecen soluciones viables para implementar el contexto conversacional utilizando *entities* con LangChain.

Tras un análisis exhaustivo, el equipo se inclinó por *Redis*. Varios factores influyeron en esta elección. Primero, la alta performance de *Redis*, que, al almacenar datos en memoria, garantiza un acceso casi inmediato a la información. Esta característica es crítica para garantizar que las respuestas del *chatbot* sean ágiles, sin retrasos innecesarios debido a consultas a la base de datos. Además, Redis Enterprise Cloud surgió como un beneficio adicional, ya que ofrece un servicio de hosting especializado para bases de datos Redis. Esta solución elimina la necesidad de desplegar y gestionar la infraestructura por separado, simplificando considerablemente el proceso.

El equipo también optó por Redis para almacenar el registro de conversaciones. Aunque comparte muchas ventajas con SQLite, la decisión de utilizar Redis para ambos propósitos

promovió la cohesión y uniformidad en el manejo de datos. No obstante, es importante mencionar una limitación inherente a Redis: su naturaleza volátil, ya que almacena datos en memoria. Esto significa que, después de cierto tiempo, la información puede ser eliminada. De igual forma, Redis ofrece la posibilidad de persistir los datos, sin embargo, dado que el hecho de persistir los datos repercute en los gastos de la base de datos de Redis, en esta fase de MVP la prioridad estaba enfocada en la performance y agilidad.

En resumen, la elección de Redis para el Chat Your Data reflejó una decisión estratégica que prioriza la eficiencia y la facilidad de implementación, preparando el proyecto para futuras optimizaciones conforme el proyecto evolucione.

7.4. Arquitectura interna de cada componente

7.4.1. *Frontend*

La aplicación de *frontend* fue desarrollada en ReactJS junto con TypeScript. Como ya se explicó anteriormente, la elección se basó en el rendimiento de React, y la capacidad de reutilizar componentes eficientemente.

La organización de componentes se encuentra del anexo 13.20 Organización de componentes en React.

7.4.1.1. Comunicación con el *Backend*

Para garantizar una comunicación fluida entre el *frontend* y el *backend*, se decidió utilizar Axios para realizar solicitudes *HTTP* y gestionar las respuestas del servidor de forma efectiva.

7.4.1.2. Uso de React Hooks

Se utilizaron los React Hooks en el desarrollo del *frontend*. Estos permiten manejar el ciclo de vida y el estado de los componentes de forma más limpia y modular. Algunos de los React Hooks clave que se utilizaron son:

- useState: Para gestionar el estado local de los componentes.
- useEffect: Para realizar efectos secundarios en componentes funcionales.
- useContext: Para acceder a datos globales en toda la aplicación.
- useHistory: Para navegar entre vistas en aplicaciones de una sola página.

Librerías externas principales:

- Mui material: Material-UI, una biblioteca popular de componentes de interfaz de usuario para React basada en el diseño de Material Design.
- Firebase: Firebase, una plataforma de desarrollo de aplicaciones en la nube que proporciona herramientas y servicios para desarrollar aplicaciones *web* y móviles, como también para gestionar las sesiones (inicio de sesión y crear cuenta).
- Axios: Una biblioteca para realizar solicitudes HTTP desde el navegador o Node.js. React-Query y Redux: Utilizados para el estado global y la gestión de datos de la aplicación.
- Styled-Components: Una biblioteca que permite definir estilos CSS en los componentes React.
- Typescript: Un *superset* de JavaScript que agrega tipado estático a la aplicación.

7.4.1.3. Estructura de features

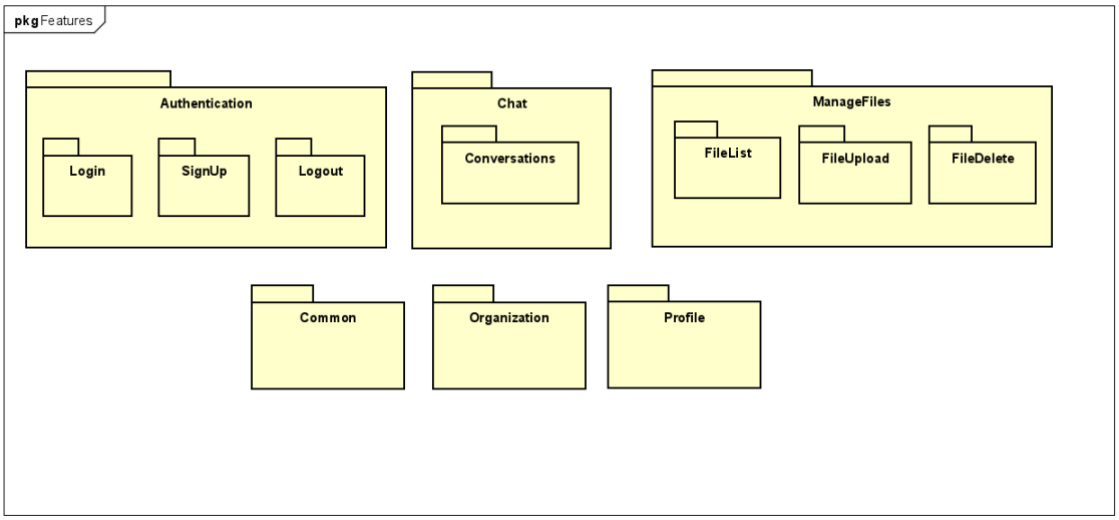


Ilustración 24- Diagrama de paquetes de estructura de *features*

7.4.2. *Backend*

Como se mencionó anteriormente, para el desarrollo del *backend*, se optó por el uso de NestJS. Este *framework* se basa en una arquitectura modular que se alinea perfectamente con la estructura de microservicios de Chat Your Data. La capacidad de dividir el código en módulos facilita la gestión y el desarrollo de componentes individuales de manera independiente. Esto es crucial para la escalabilidad de nuestro sistema, ya que se puede agregar nuevos servicios y funcionalidades sin afectar negativamente a otras partes de la aplicación.

7.4.2.1. Arquitectura en módulos

La arquitectura de Chat Your Data se destaca por su enfoque en la modularidad, una estrategia diseñada para maximizar la extensibilidad del código y permitir un crecimiento fluido a medida que el proyecto evoluciona. Esta división en módulos es una decisión estratégica que garantiza una mayor flexibilidad y facilidad en el desarrollo de nuevas funcionalidades y características en la plataforma. La justificación del porque una arquitectura en módulos se encuentra del anexo 13.21 Justificación de arquitectura en módulos en *backend*.

Organización en Módulos Principales

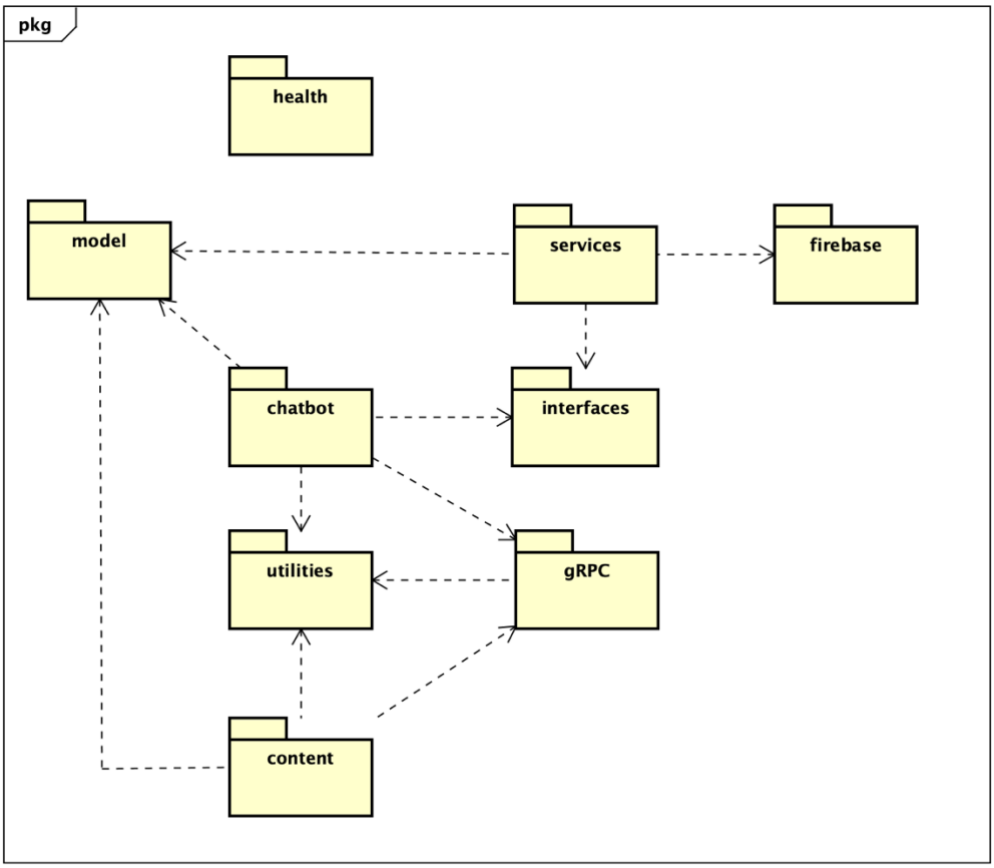


Ilustración 25- Organización en módulos principales

Los componentes del *backend* se organizan en carpetas dentro del repositorio, y se dividen principalmente en dos módulos esenciales: "*Content*" y "*Chatbot*". Cada uno de estos módulos tiene una función claramente definida en la plataforma.

7.4.2.1.1. Módulo "Content"

Este módulo se encarga de todas las operaciones relacionadas con la gestión de contenido, incluyendo la Alta, Baja y Modificación (ABM) de documentos. La separación de esta

funcionalidad en un módulo independiente permite un enfoque dedicado en el manejo de documentos, lo que facilita futuras expansiones y mejoras en esta área. Dentro del anexo 13.22.1 Módulo Content se encuentran los diagramas.

7.4.2.1.2. Módulo "Chatbot"

Este módulo es el núcleo de la interacción entre el usuario y la inteligencia artificial. Todas las actividades relacionadas con el *chat* y las conversaciones se gestionan aquí. La modularización asegura que las características de *chat* puedan ser desarrolladas y escaladas sin afectar otros aspectos del sistema. Dentro del anexo 13.22.2 Módulo Chatbot se encuentran los diagramas.

7.4.2.1.3. Módulos de comunicación externa

Además de los módulos principales, Chat Your Data incluye dos módulos que gestionan la comunicación con servicios externos: Firebase y gRPC. Estos módulos permiten la integración sin problemas con sistemas externos y servicios, lo que es esencial para una plataforma que debe interactuar con múltiples tecnologías y fuentes de datos. Dentro del anexo 13.22.3 Módulos de comunicación Externa se encuentran los diagramas.

7.4.2.1.4. Módulos de utilidades globales

El conjunto de módulos restantes se enfoca en proporcionar utilidades globales para toda la aplicación. Esto incluye la definición de modelos, interfaces y controladores de estado de la aplicación. Estos módulos desempeñan un papel fundamental en la estandarización y reutilización de código en toda la plataforma. Dentro del anexo 13.22.4 Módulos de utilidades globales se encuentran los diagramas.

7.4.2.2. Modelo de tres capas

El modelo de arquitectura que divide una aplicación en tres capas principales: controladores, capa de negocio (o lógica de negocio), y capa de datos, que se conoce comúnmente como "arquitectura de tres capas" o "modelo de tres capas" [54]. Este enfoque arquitectónico se utiliza ampliamente en el desarrollo de *software* para separar las preocupaciones y promover la modularidad y la reutilización del código. El equipo hace uso de este modelo de tres capas internamente para cada submódulo del *backend*.

- Capa de Presentación (Controladores): Esta capa se encarga de la interfaz de comunicación con el *frontend* para la interacción con el usuario final. Los controladores manejan las solicitudes del usuario y gestionan la navegación y la presentación de datos.
- Capa de Negocio (Lógica de Negocio): En esta capa, se encuentra la lógica de negocio central de la aplicación. Aquí se procesan los datos, se aplican reglas comerciales y se gestionan las operaciones específicas de la aplicación. La capa de negocio es independiente de la capa de presentación y la capa de datos, lo que facilita las pruebas unitarias y la reutilización del código.
- Capa de Datos: La capa de datos se encarga de la gestión y el acceso a los datos. Esto incluye la interacción con bases de datos, almacenamiento de datos, conexiones con servicios externos y operaciones relacionadas con la persistencia de datos. Separar esta capa permite cambiar el almacenamiento de datos sin afectar la lógica de negocio y la presentación.

Aplicación en Chat Your Data

Este enfoque de tres capas es utilizado por los módulos principales explicados anteriormente: *Chatbot* y *Content*. Por ejemplo, en el módulo *Chatbot*, hay un *ChatbotController* y un *ChatbotService*, que se comunica con la capa de datos en la clase *ChatFirebaseService*. Es altamente escalable y es aplicable a los futuros módulos de la aplicación, como *User*, *Organization* y otros. Cada nuevo módulo puede seguir la misma estructura arquitectónica para garantizar la consistencia en toda la aplicación. Los módulos son claramente distinguibles ya que en la gran mayoría de los casos cada módulo representa a una clase del dominio del sistema. El diagrama de clases aplicable a lo mencionado será mostrado más adelante.

Por ejemplo, si se implementa el módulo *User*, se crearían controladores para manejar las solicitudes relacionadas con la gestión de usuarios, servicios de negocio para aplicar reglas comerciales específicas de usuarios y componentes de acceso a datos para interactuar con la base de datos o servicios externos relacionados con la gestión de usuarios.

La arquitectura de tres capas a nivel de módulo garantiza una separación clara de responsabilidades, facilita el desarrollo y mantenimiento de cada componente y promueve la reutilización del código en toda la aplicación, lo que es esencial tanto para los módulos existentes como para los futuros.

7.4.2.3. Inversión de dependencias - Inyección de dependencias

La Inversión de Control (IoC) y la Inyección de Dependencias (DI) son técnicas fundamentales en el diseño de *software* que permiten la creación de sistemas altamente flexibles y fácilmente adaptables a cambios futuros en las implementaciones y proveedores de servicios [55]. Estos conceptos se basan en el principio de que los componentes no deben controlar directamente sus dependencias, sino que el control debe invertirse y delegarse a un contenedor o sistema externo.

En Chat Your Data, se aplica activamente la IoC y la DI mediante el uso de interfaces y sus implementaciones. Un ejemplo de esto se encuentra en la carpeta de interfaces, que contiene dos archivos principales: *chat.interface* y *user.interface*. Estas interfaces desempeñan un papel clave en el acceso y la modificación de datos relacionados con usuarios y chats a nivel de la plataforma.

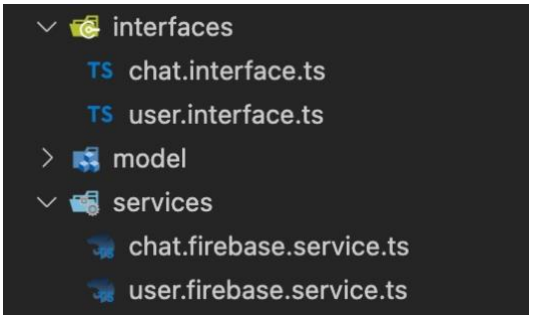


Ilustración 26- Evidencia de uso de interfaces y servicios

Las interfaces *chat.interface* y *user.interface* se diseñaron para definir claramente los contratos que deben cumplir las clases que se encargan de interactuar con los datos de usuarios y *chats*. Estos contratos especifican los métodos y las operaciones que deben estar disponibles para acceder y modificar estos datos, independientemente de la implementación subyacente o del proveedor de servicios utilizado.

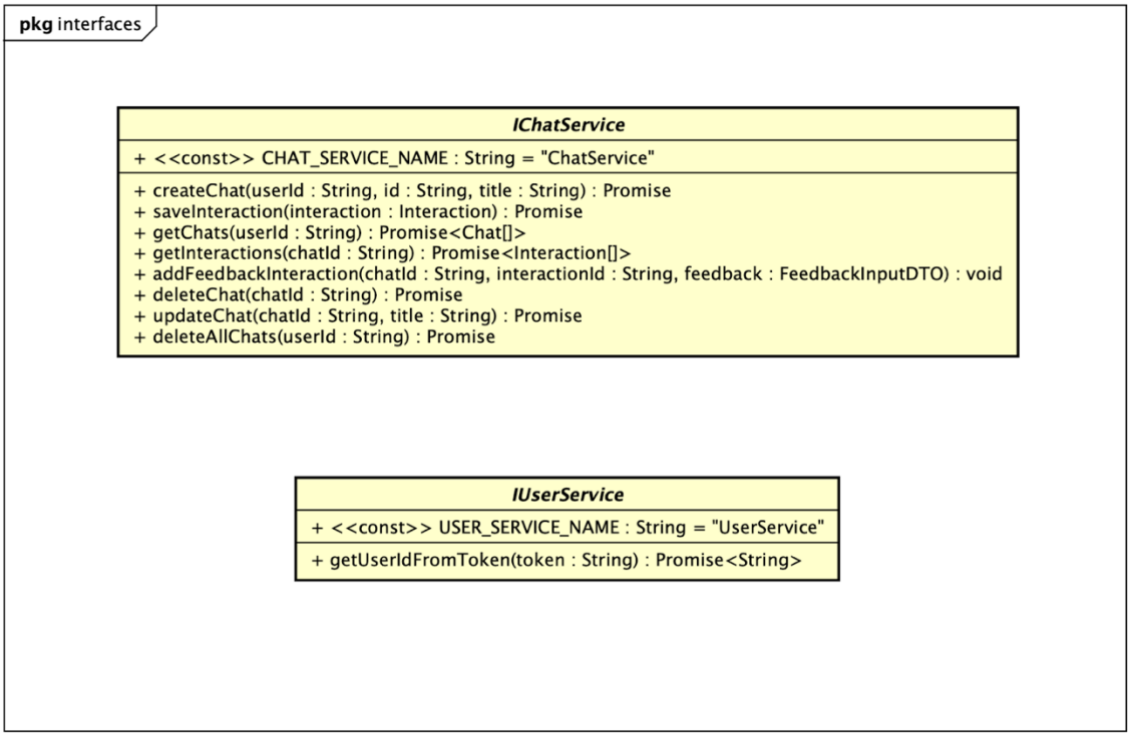


Ilustración 27- Uso de interfaces

Las clases *chat.firebase.service* y *user.firebase.service* se crearon como implementaciones concretas de estas interfaces. Estas implementaciones están diseñadas específicamente para interactuar con *Firebase*, que es el proveedor de servicios actualmente utilizado para almacenar y gestionar datos de usuarios y *chats* en la plataforma.

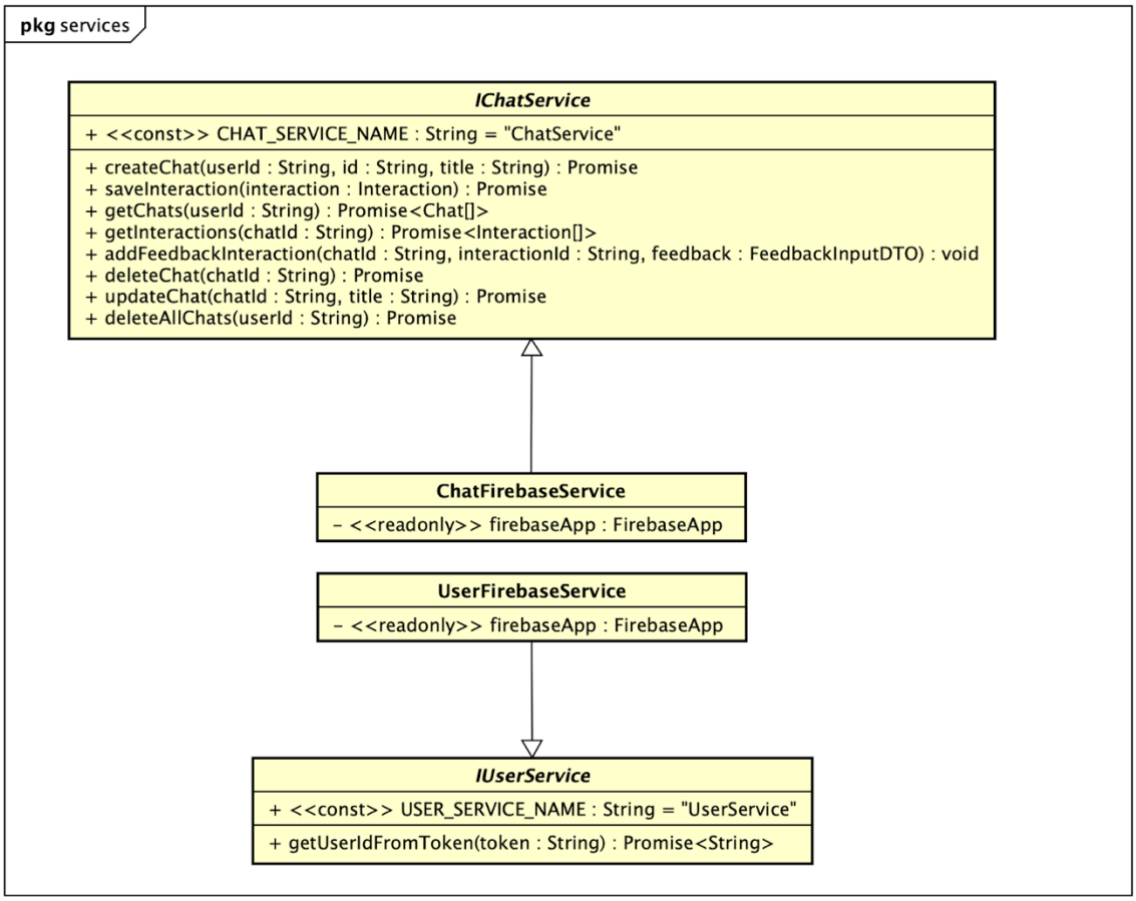


Ilustración 28- Uso de servicios

La ventaja principal de esta implementación basada en interfaces es que proporciona una capa de abstracción entre la lógica de negocio y el proveedor de servicios subyacente (en este caso, Firebase). Esto significa que, si en el futuro se decide cambiar a otro proveedor o tecnología de almacenamiento de datos, la transición es considerablemente más sencilla.

La flexibilidad y facilidad de cambio se logran gracias a la capacidad de simplemente crear nuevas implementaciones de las interfaces *chat.interface* y *user.interface* que se adapten al nuevo proveedor o tecnología, sin necesidad de modificar la lógica de negocio existente cumpliendo con el principio Open/Closed Principle. Esto respalda directamente el atributo de calidad de "Extensibilidad" en la arquitectura de la plataforma. La aplicación de la Inversión de Control e Inyección de Dependencias a través de interfaces en Chat Your Data proporciona una sólida base técnica para futuras expansiones y adaptaciones en la gestión de datos de usuarios y *chats*, al tiempo que mantiene una alta flexibilidad y modularidad en la arquitectura del sistema.

7.4.2.4. Uso de *guards*

En la arquitectura de Chat Your Data los *guards* desempeñan un rol central en garantizar la seguridad y la autenticación de las solicitudes de los usuarios. Estos componentes se utilizan para controlar el acceso a las rutas y los controladores de la aplicación, asegurando que solo aquellos usuarios autenticados y autorizados puedan acceder a recursos y funcionalidades específicas.

Además, los *guards* proporcionan una capa de seguridad uniforme en toda la aplicación, lo que simplifica el control de acceso y garantiza que las políticas de seguridad se apliquen de manera coherente en toda la plataforma.

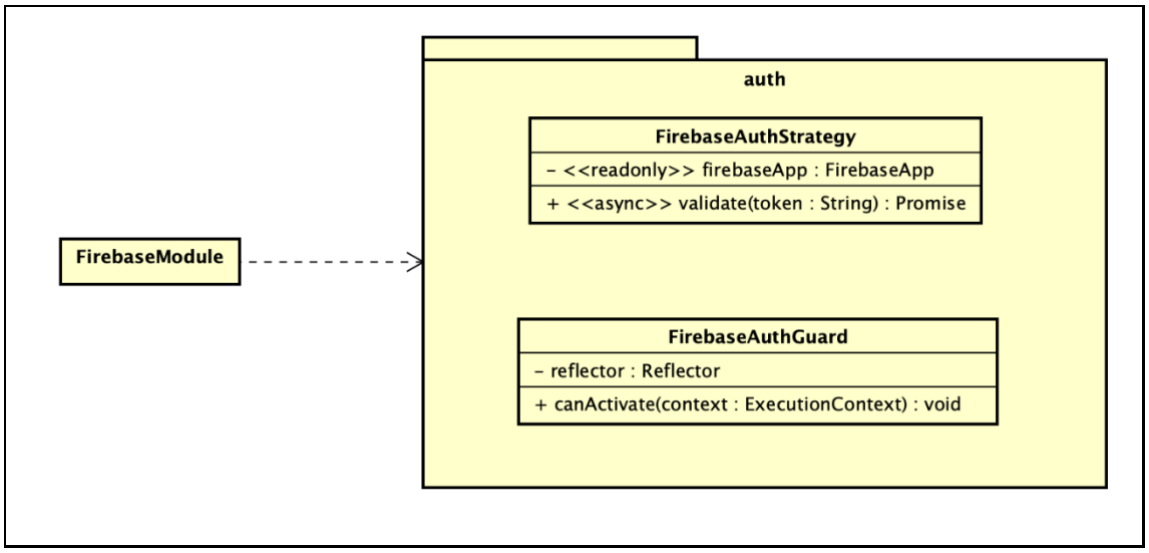


Ilustración 29- Diagrama de autenticación

Para el uso de los *guards* se hace un uso de 'AuthGuard' y 'PassportStrategy', dos elementos clave para garantizar la seguridad y la autenticación de las solicitudes de los usuarios.

- 'AuthGuard': Este componente esencial se utiliza para controlar el acceso a rutas y controladores en la aplicación. A través de su configuración y extensibilidad, se pueden aplicar políticas de seguridad específicas para determinar quién puede acceder a ciertas funcionalidades y recursos.
- 'PassportStrategy': *Passport* es un módulo ampliamente utilizado en la comunidad de Node.js para la autenticación. 'PassportStrategy' se usa para implementar estrategias de autenticación personalizadas, como la validación de *tokens*, la autorización basada en roles y más.

Dentro del anexo 13.23 Extensibilidad y autenticación en Chat Your Data, se puede observar la extensibilidad de la autenticación y como se implementa dentro los controladores.

En resumen, el uso de *Guards* como 'FirebaseAuthGuard' en Chat Your Data garantiza un alto nivel de seguridad y autenticación en la plataforma, protegiendo los recursos y las funcionalidades de acceso restringido. Estos *Guards* trabajan en conjunto con la estrategia de autenticación 'FirebaseAuthStrategy' para validar la identidad de los usuarios a través de 'Firebase Authentication', lo que contribuye a la robustez y la confiabilidad de la plataforma.

7.4.3. PluginAI

7.4.3.1. Introducción - Componente PluginAI

El componente 'PluginAI' desempeña un papel fundamental dentro de la arquitectura de Chat Your Data. Su función principal es habilitar y potenciar la inteligencia artificial en la plataforma, lo que le permite interactuar de manera conversacional con los usuarios, comprender y procesar documentos, y generar contenido nuevo de manera eficiente. En resumen, 'PluginAI' es el "cerebro" detrás de la conversación y el procesamiento de datos en la plataforma.

7.4.3.2. Arquitectura interna de PluginAI

Para comprender mejor cómo 'PluginAI' cumple con su función, es útil explorar sus componentes internos organizados en varios paquetes:

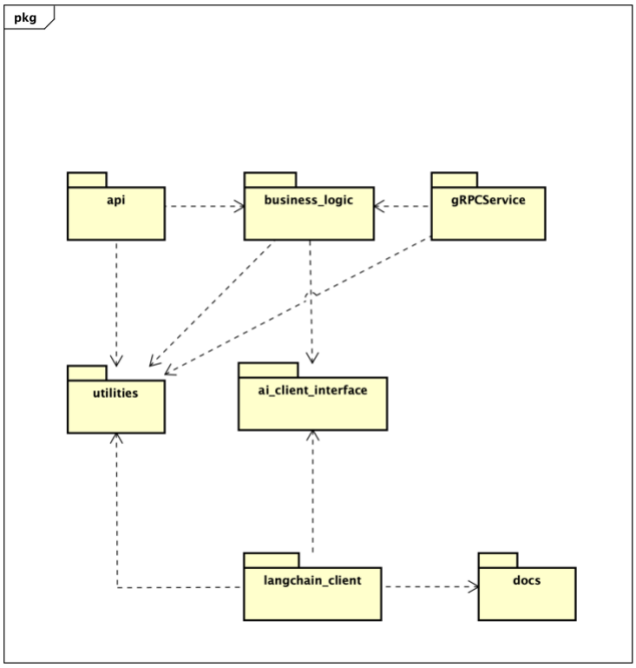


Ilustración 30- Arquitectura interna de PluginAI

- **‘gRPCService’**: Este paquete se encarga de la comunicación entre ‘PluginAI’ y el *backend* central utilizando gRPC. Es uno de los canales de entrada al sistema. Proporciona una forma eficiente, rápida y confiable de transmitir la información entre sí. Como se mencionó anteriormente, la velocidad de respuesta es importante para la experiencia del usuario, y gRPC provee una forma de comunicación rápida y sencilla para el procesamiento de *prompts*.
- **‘api’**: Este paquete contiene la *API* pública que expone las capacidades de ‘PluginAI’, permitiendo que el *backend* central (*server*) interactúe con él de manera controlada y segura. Esta *API*, si bien es extensible a futuro, por el momento se utiliza únicamente para todo lo que no es procesamiento de un *prompt*, donde la velocidad no es requisito primordial.
- **‘business_logic’:** Este paquete se encarga de recibir las órdenes del ‘Controller’ y comunicarse con la interfaz de IA para obtener procesar lo solicitado.
- **‘utilities’:** Contiene utilidades y herramientas comunes que se utilizan en toda la arquitectura de ‘PluginAI’. Estas utilidades facilitan la implementación y el mantenimiento de la plataforma.
- **‘ai_client_interface’:** Este paquete proporciona una interfaz que permite la interacción con proveedores de servicios de inteligencia artificial. Facilita la integración de capacidades de IA externas a través de una abstracción coherente.

- **‘langchain_client’:** LangChain es un marco de trabajo especializado en el procesamiento de lenguaje natural (*NLP*) y *embeddings*. Este paquete se encarga de interactuar con LangChain para el procesamiento de documentos, la generación de *embeddings*, el procesamiento de *prompts*, entre otros.
- **‘docs’:** Contiene los documentos que el usuario sube a la plataforma.

A grandes rasgos, lo mencionado anteriormente es una breve explicación del rol de PluginAI en el sistema global, y las funciones de cada uno de sus componentes. A continuación, se profundizará en cada uno de estos paquetes, explorando su funcionalidad y su contribución al sistema en detalle.

7.4.3.3. Canales de comunicación - API y gRPC

El primer canal de comunicación de la aplicación es mediante *gRPC* que se encarga de la interacción en tiempo real entre los usuarios y el componente de IA para el procesamiento de *prompts*. Este canal permite a los usuarios enviar solicitudes al componente y recibir respuestas (*completion*) generadas por la IA de manera eficiente. En el anexo 13.25.1.1 gRPC se encuentra el diagrama del paquete.

El funcionamiento de este canal se basa en la definición de un servicio gRPC llamado ‘PromptService’, que contiene una operación llamada ‘ProcessPrompt’. Esta operación recibe dos parámetros: ‘chat_id’, que identifica al *chat* entre el usuario y la IA, y ‘*prompt’*, que representa la solicitud del usuario en formato de texto.

```
message PromptRequest {
  string chat_id = 1;
  string prompt = 2;
}
```

Ilustración 31- *Prompt Request gRPC*

Cuando un usuario envía un *prompt* a través del canal gRPC, el sistema PluginAI procesa la solicitud utilizando la lógica de negocio de ‘ProcessPromptLogic’. Este componente se encarga de determinar cómo debe manejarse la solicitud, qué información debe recuperarse de la base de datos vectorial y cómo generar una respuesta relevante y precisa.
Una vez que la lógica de negocio ha procesado la solicitud del usuario y ha generado una respuesta, esta se envía de vuelta al cliente gRPC en formato JSON a través de la respuesta ‘CompletionAnswer’, que incluye tanto la *completion* como los *source documents*. Los *source*

documents son los documentos de donde la IA saco información de la base de datos vectorial para dar una respuesta.

```
message CompletionAnswer {
  string completion = 1;
}
```

Ilustración 32- Completion Answer

El canal de comunicación gRPC facilita la interacción en tiempo real y de la forma más rápida posible entre el *backend* y el componente de PluginAI. Permite a los usuarios enviar solicitudes de *prompts* y recibir respuestas generadas por la IA de manera eficiente, brindando una experiencia conversacional fluida y enriquecida.

El segundo canal de comunicación es la 'API REST', responsable de la interacción con el componente para la gestión de documentos y otras operaciones relacionadas con la base de datos vectorial, además de otras cuestiones que tienen que ver con el *chat*, pero no en el procesamiento de un *prompt*, sino como eliminar un *chat*. Si bien aún no está implementado por cuestiones de tiempo y prioridades, en el corto plazo existirá un *ChatController* para las cuestiones mencionadas anteriormente relacionadas a los chats. Actualmente, la API tiene dos 'controllers'. El primero es el ContentController, que se encarga de todo lo relacionado a documentos. El segundo es HealthController, que similarmente a lo explicado en el capítulo de la arquitectura del *backend* central, sirve para determinar el estado del sistema. En el anexo 13.25.1.2 API se encuentra el diagrama del paquete de API.

Los endpoints del *ContentController* son los siguientes:

- Alta de Documentos (*POST /content*): Cuando un usuario desea cargar un nuevo documento en la plataforma, realiza una solicitud a este *endpoint*. El documento se carga en el servidor y se almacena de manera asíncrona en el sistema de archivos. Además, se inicia el proceso de procesamiento de este documento para crear *embeddings* que lo representen.
- Baja de Documentos (*DELETE /content*): Este *endpoint* se utiliza para eliminar documentos previamente cargados en la base de datos vectorial. El cliente proporciona la información sobre los documentos que se deben eliminar, y el sistema se encarga de ejecutar la operación correspondiente.

- Consulta de Documentos (*GET /content*): Para obtener una lista de los documentos que han sido cargados en la plataforma, los usuarios pueden realizar una solicitud a este *endpoint*. La plataforma responde con la lista de nombres de archivos disponibles.
- Descarga de Documentos (*GET /content/document*): Cuando un usuario desea obtener un documento específico que ha sido cargado en la plataforma, puede utilizar este *endpoint* proporcionando el nombre del archivo deseado. La plataforma responde con el contenido del documento.

La existencia de la API como un canal alternativo de comunicación se justifica principalmente por las diferencias en las necesidades de velocidad de respuesta. Mientras que gRPC ofrece una rapidez esencial para simular una conversación fluida y natural con la IA, similar a hablar con otro ser humano, la API se adecúa mejor a otros propósitos de la aplicación. Esta API presenta ventajas notables: facilita y agiliza el desarrollo de nuevas funciones y *endpoints*, en comparación con gRPC. Es particularmente útil porque la transferencia de archivos en gRPC resulta más compleja. Además, dada la ambición del proyecto de evolucionar y adaptarse rápidamente, se prioriza la implementación rápida de características innovadoras para distinguirse de los competidores, ya sean directos o indirectos.

7.4.3.4. Paquete business_logic

El paquete business_logic se encarga de coordinar las operaciones entre los canales de comunicación, ya sea a través de gRPC o la API REST, y la interfaz de inteligencia artificial que procesa las solicitudes de los usuarios. En el anexo 13.25.2 Paquete de business_logic se encuentra el diagrama del paquete de business_logic.

Dentro del paquete business_logic, se encuentran dos clases principales:

- ProcessPromptLogic: Se encarga de procesar las solicitudes recibidas a través del canal de comunicación gRPC. Su función principal es recibir los *prompts* enviados por los usuarios a través de gRPC y coordinar su procesamiento. Una vez que el *prompt* ha sido procesado por la interfaz de inteligencia artificial, la clase ProcessPromptLogic se encarga de enviar la respuesta generada de regreso a través del canal gRPC correspondiente. En resumen, esta clase actúa como un intermediario entre los usuarios y la IA para el procesamiento de *prompts* y *completions*.

- ContentLogic: Se encarga de manejar las solicitudes provenientes de la API REST. Su principal responsabilidad es recibir solicitudes relacionadas con la gestión de documentos, como la carga, eliminación y consulta de archivos. Después de procesar las solicitudes y obtener resultados, ContentLogic proporciona las respuestas adecuadas a través de la API REST.

Estas clases existen para intermediar entre los canales de comunicación explicados anteriormente y la interfaz de inteligencia artificial. A medida que el componente crece y se agregan nuevos controladores en la API, o nuevas formas de comunicación, se implementa una clase nueva correspondiente a ese caso de uso, generando un sistema altamente mantenible y extensible.

7.4.3.5. Inversión de dependencias / Inyección de dependencias

En la arquitectura de PluginAI de Chat Your Data, se aplican los principios fundamentales de inversión de dependencias e inyección de dependencias para lograr una estructura de *software* flexible y extensible. Estos principios son cruciales para mantener la modularidad y la escalabilidad del sistema, especialmente en lo que respecta a la interacción con la inteligencia artificial.

El Principio de Inversión de Dependencias (DIP) establece que los módulos de alto nivel no deben depender de módulos de bajo nivel, sino de abstracciones compartidas. Además, las abstracciones no deben depender de los detalles de implementación; en cambio, los detalles deben depender de las abstracciones. En el contexto de Chat Your Data, las interfaces IContentAI e IProcessPromptAI actúan como abstracciones que definen un contrato sin entrar en los detalles de cómo se implementa la IA.

La inyección de dependencias es una técnica que implementa el DIP y permite que las dependencias requeridas por un componente se inyecten desde el exterior en lugar de que el componente las cree internamente. En el caso de las interfaces IContentAI e IProcessPromptAI, la inyección de dependencias se utiliza para proporcionar una implementación específica de IA a los módulos ContentLogic y ProcessPromptLogic. Esto significa que la implementación de IA puede intercambiarse fácilmente simplemente proporcionando una implementación diferente que cumpla con las mismas interfaces.

Dentro del paquete ai_client_interface, se encuentran las interfaces IContentAI e IProcessPromptAI. Estas interfaces desempeñan un papel fundamental al definir un conjunto de métodos que deben ser implementados por cualquier proveedor de servicios de IA. En el anexo 13.25.3 Paquete de ai_client_interface se encuentra el diagrama del paquete.

Hay que considerar que esta arquitectura fue diseñada al principio de este trabajo final de carrera, por lo que se consideró -y se sigue considerando- que era necesario tener una interfaz que pueda contar con diferentes implementaciones para IA, pudiendo cambiar lo más rápido posible de ser necesario. Esto era sumamente importante ya que las tecnologías en el campo de la IA estaban montadas en una ola muy grande y que tuvo un *boom* muy fuerte, por lo que la volatilidad de la misma era grande y la probabilidad de que surjan nuevas y mejores era bastante alta. Aunque actualmente solo existe una implementación con LangChain, el uso de estas interfaces permite que el componente sea altamente extensible. Esto significa que, en el futuro, otras implementaciones de IA pueden reemplazar fácilmente la implementación existente sin requerir cambios significativos en otros componentes del sistema.

7.4.3.6. Paquete de implementación - LangChain client

7.4.3.6.1. Introducción

LangChain es un marco de trabajo diseñado para simplificar la creación de aplicaciones utilizando e interactuando con LLMs, como los populares GPT o LLaMA. Es una biblioteca que permite crear y combinar diferentes modelos de inteligencia artificial, agentes y solicitudes de manera estructurada. Se utiliza para la integración de modelos de lenguaje en general, incluyendo análisis y resumen de documentos, *chatbots* y análisis de código. Asimismo, permite crear cadenas lógicas entre uno o más LLMs, lo que proporciona una gran utilidad. LangChain es modular y fácil de usar, lo que permite personalizar cadenas existentes y crear nuevas. En el anexo 13.25.4 Paquete de langchain_client se encuentra el diagrama del paquete.

Este paquete representa lo que es la implementación de todo lo relacionado a IA en el proyecto, desde la interacción con la base de datos vectorial hasta el procesamiento de los *prompts* que ingresan los usuarios. A alto nivel, el paquete langchain_client contiene los siguientes componentes:

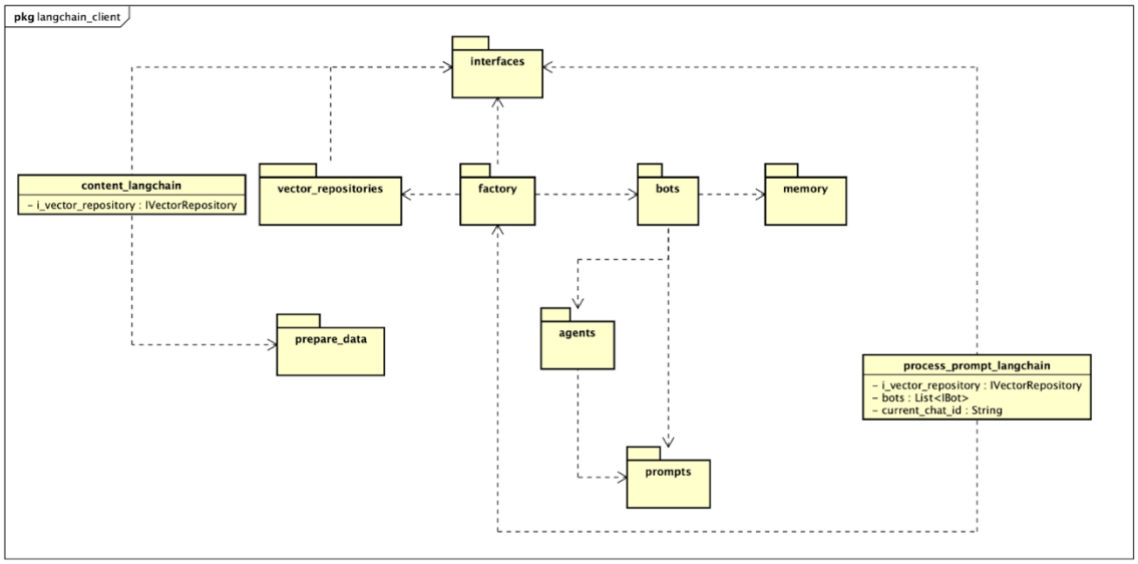


Ilustración 33- Diagrama de paquete de LangChain

- interfaces: contiene la interfaz i_vector_repository, para la definición de los métodos y propiedades que deben implementar la base de datos vectorial que se utilice. Esto permite que los componentes de LangChain sean reutilizables y extensibles.
- vector_repositories: contiene la implementación de Chroma como base de datos vectorial, y el código relacionado a la generación de *embeddings*. Los *embeddings* de texto son vectores que representan el significado de un texto.
- *factory*: es responsable de la creación de instancias de los componentes de LangChain. Esto facilita la creación de nuevos *bots* y agentes.
- *bots*: los *bots* son los componentes que efectivamente interactúan y se comunican con los modelos de IA.
- *memory*: almacena el estado de la conversación entre el *bot* y el usuario. Esto permite al *bot* tener contexto de la conversación que se tuvo con el usuario.
- *agents*: tienen la capacidad de delegar la realización de tareas específicas, como la búsqueda en la base de datos vectorial u operaciones matemáticas. Los *agents* se pueden utilizar para ampliar las capacidades de los *bots*.
- *prompts*: son las indicaciones que se le dan al *bot* para generar texto.
- *prepare_data*: contiene las herramientas para preparar los datos para su uso con LangChain. Esto incluye tareas como la carga de documentos y su separación en *chunks* para el guardado en la base de datos vectorial.

7.4.3.6.2. Manejo de contenido

La clase `ContentLangChain`, que implementa la interfaz `IContentAI`, gestiona la comunicación con la base de datos vectorial, específicamente con ChromaDB, para llevar a cabo operaciones tales como acceso, creación, modificación y eliminación de archivos.

La información a almacenar en dicha base de datos es textual, categorizándose como información desestructurada. Este tipo de información, por su naturaleza sin formato organizado o predefinido, se torna compleja de analizar y las bases de datos tradicionales suelen encontrar dificultades para almacenarla y procesarla eficientemente. Para habilitar el procesamiento de esta información por parte de la inteligencia artificial, la data desestructurada se convierte y se almacena en formatos numéricos mediante el uso de vectores. El método de transformación (reconocido como *embedding*) se abordará en las secciones siguientes de este capítulo.

Esta información, ahora en formato de vectores numéricos, se almacena en una base de datos vectorial. Este modelo de base de datos es especialmente útil para realizar búsquedas por similitud, permitiendo identificar relaciones entre vectores.

Para cada documento gestionado, se invoca la función `upload_content` de esta clase, detallada en el anexo 13.25.5 Método upload_content. El proceso para convertir la información desde el formato de entrada original (como archivos PDF o textos en general) hasta el formato de salida vectorial involucra los siguientes pasos, los cuales serán explicados en detalle en capítulos posteriores:

1. Carga del documento.
2. División del documento en segmentos (*chunks*).
3. Procesamiento de *embeddings* de los segmentos, transformándolos en vectores numéricos.
4. Almacenamiento de los vectores numéricos en la base de datos vectorial.

Document Loading y Document Splitting

El Document Loading y el Document Splitting representan los pasos 1 y 2 mencionados en el proceso general. Las clases responsables de estos procesos son las representadas visualmente en la siguiente imagen, ambas contenidas en el paquete *prepare_data*.

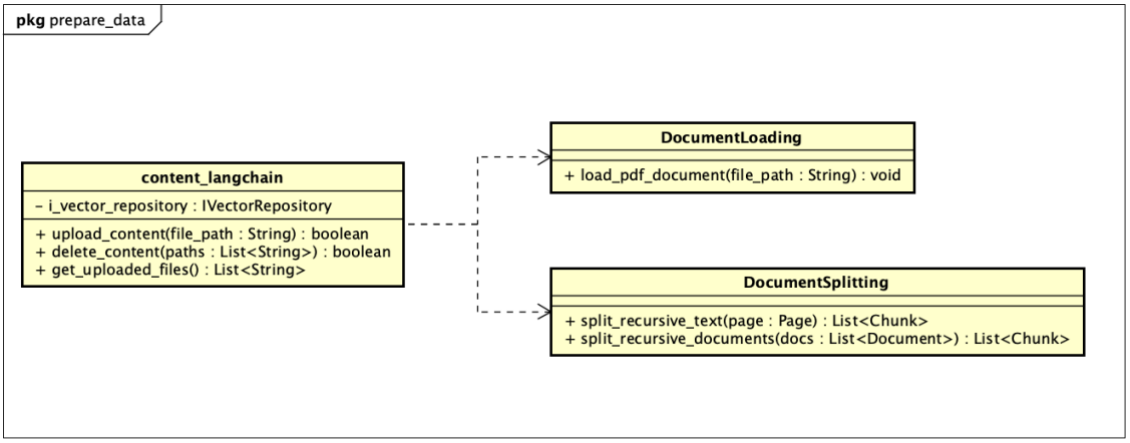


Ilustración 34- Diagrama de paquetes *prepare_data*

Document Loading es el proceso por el cual se recibe un documento en su formato original (PDF u otro) y se obtienen los documentos en un formato que el Document Splitting pueda entender.

Para esta parte del proceso fue usada la librería PyPDFLoader de LangChain. Esta librería recibe la ubicación del directorio dentro del repositorio donde se encuentran ubicados los documentos, y un nombre del archivo, que completa la dirección del documento. Es importante aclarar que esto es algo con planes a mejorar a futuro. Los documentos no deberían estar guardados en el repositorio del proyecto temporalmente, sino que deberían ser cargados desde un repositorio externo o guardarlos en el proyecto, pero en otro espacio con mayor seguridad. Por razones de prioridades se decidió no implementar esto por el momento. Lo que retorna la librería son los documentos en un formato que el Document Splitting pueda entender.

Document Splitting es el proceso en el cual los documentos son separados en *chunks* para luego ser vectorizados en el proceso de *embedding*. Dividir los documentos en fragmentos es necesario para procesarlos de manera eficiente. Los modelos de aprendizaje automático que se utilizan para procesar documentos suelen estar entrenados en datos fragmentados. Esto se debe a que los fragmentos son más fáciles de procesar que los documentos completos. Además, dividir los documentos en fragmentos permite aprovechar el paralelismo. Esto significa que se pueden procesar múltiples fragmentos al mismo tiempo, lo que puede acelerar el procesamiento de documentos.

La función split_recursive_documents en la clase DocumentSplitting es llamada desde la clase ContentLanchain para obtener los *chunks* del documento correspondiente.

```
def split_recursive_documents(self, docs):
    r_splitter = RecursiveCharacterTextSplitter(
        chunk_size=Constants.CHUNK_SIZE_EMBEDDING,
        chunk_overlap=Constants.CHUNK_OVERLAP,
        separators=["\n\n", "\n", "\. ", " ", ""]
    )
    splits = r_splitter.split_documents(docs)
    return splits
```

Ilustración 35- Función *split_recursive_documents*

Internamente, esta función utiliza la herramienta RecursiveCharacterTextSplitter de *LangChain*, que divide los documentos en fragmentos según una lista de separadores. En este caso, los separadores son [\n\n, \n, ., , y ""], ordenados según dé prioridad. El objetivo de esto es lograr que la división de los *chunks* tenga cierto sentido, es decir que un *chunk* sea lo más cercano posible a una sección del texto, en su defecto a un párrafo, en su defecto a un enunciado, y así sucesivamente. Por supuesto que esto depende del tamaño máximo de los *chunks*, que está determinado por el parámetro *chunk_size*. El parámetro *chunk_overlap* determina la superposición entre los fragmentos. El valor de esta superposición tiene como objetivo asegurar la continuidad del texto, y la correlación entre un chunk y el siguiente.

Base de datos vectorial

Una **base de datos vectorial** se especializa en almacenar y gestionar datos representados en forma de vectores, los cuales describen objetos según sus características a través de representaciones matemáticas. La habilidad de las bases de datos vectoriales para almacenar y recuperar vectores como puntos de alta dimensión añade potencialidades significativas para realizar búsquedas eficaces y rápidas de vecinos cercanos en un espacio multidimensional. Aunque se explicó en detalle la funcionalidad de estas búsquedas en la sección 5.6 Bases de datos en inteligencia artificial, esencialmente, el texto de búsqueda se vectoriza y se identifican los vectores matemáticamente más cercanos al vector de búsqueda, lo que produce resultados alineados con la información buscada.

En la implementación de LangChain de PluginAI, una interfaz facilita la comunicación con bases de datos vectoriales. El diagrama de este paquete se encuentra en el anexo 13.25.6 Paquete de vector_repositories. Alineado con las prácticas adoptadas en todo el proyecto, se optó por usar interfaces para la IA generativa y aspectos relacionados, en vista de su volatilidad y cambio en el corto plazo, asegurando así la mantenibilidad y extensibilidad de la aplicación.

ChromaRepository, una clase responsable de la comunicación con la base de datos ChromaDB a través de LangChain, implementa la interfaz mencionada. ChromaDB es una base de datos de vectores de código abierto utilizada para almacenar y recuperar vectores de incrustación, especialmente beneficiosa para sistemas de IA generativa y NLP, por su eficiencia en recuperación de información y capacidad para búsquedas por similitud.

Esta clase implementa el patrón Singleton, descrito en el anexo 13.24 Patrón Singleton. Para inicializar ChromaDB, son necesarios *embedding_function* y *persist_directory* para gestionar *embeddings* y establecer la ubicación de almacenamiento de los vectores, respectivamente.

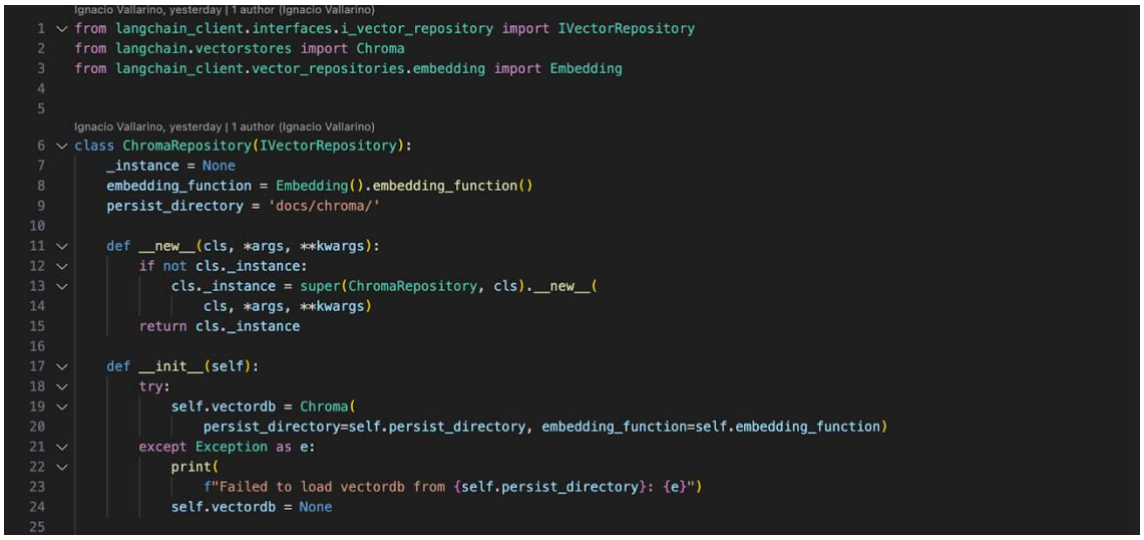


Ilustración 36- Clase *ChromaRepository*

La creación de *embeddings*, que convierte palabras en números (vectores) para ser procesadas por computadoras, se realiza entrenando un modelo para predecir palabras basadas en su contexto en grandes conjuntos de texto. Este método es crucial para que la IA pueda procesar y buscar información eficientemente, con más detalles disponibles en el anexo 13.5.2 Embeddings.

Tras el Document Splitting, el proceso de *embeddings* ocurre al almacenar los *splits* (documentos divididos en *chunks*) en ChromaDB.

El almacenamiento de la base de datos ChromaDB se realiza en un directorio del repositorio del proyecto, lo cual, como se explicó en capítulos anteriores, es un aspecto a mejorar en el corto plazo. El contenido de la base de datos vectorial idealmente debería estar alojado en la nube o localmente en el sistema donde se utiliza la plataforma, pero, por cuestiones de tiempo y prioridades, y siendo este un MVP, se optó por no abordar esto por el momento.

Es vital mencionar que la base de datos vectorial es común para todos los usuarios de Chat Your Data. Esto significa que los documentos a los que accede cada usuario de la aplicación son idénticos. Pensando en la futura evolución del proyecto y en los potenciales requerimientos de los clientes, se podría considerar implementar reglas de acceso que permitan a los empleados de una organización acceder a documentos específicos, dependiendo de su nivel de acceso, área, entre otros aspectos.

7.4.3.6.3. Procesamiento de un *prompt*

ProcessPromptLangchain es la responsable de coordinar el procesamiento de una solicitud de un usuario, lo que se conoce como *prompt*. Similarmente a la clase ContentLangchain, ProcessPromptLangchain implementa la interfaz IProcessPromptAI, por las mismas razones detalladas anteriormente en el capítulo de 'Manejo de contenido'.

```
def process_prompt(self, chat_id: str, prompt: str) -> dict:
    log.ConsoleLog("Calling ask_ai method").info()
    if (self.current_chat_id != chat_id):
        self.current_chat_id = chat_id
        # self.bots[0].clear_chat_memory()
        self.bots[0].set_current_chat_id(chat_id)
        self.save_current_chat_id()

    ConversationMemory().initialize_conversation(chat_id)

    output = self._execute_bots(prompt)

    source_documents = SourceDocuments()._instance.get_source_documents()

    response_chat_your_data = {"response": output,
                               "source_documents": source_documents}

    return response_chat_your_data
```

Ilustración 37- Definición de process_prompt

A continuación, se presenta un diagrama de secuencia que representa visualmente a la función process_prompt. Los detalles técnicos de implementación se explican en los capítulos siguientes.

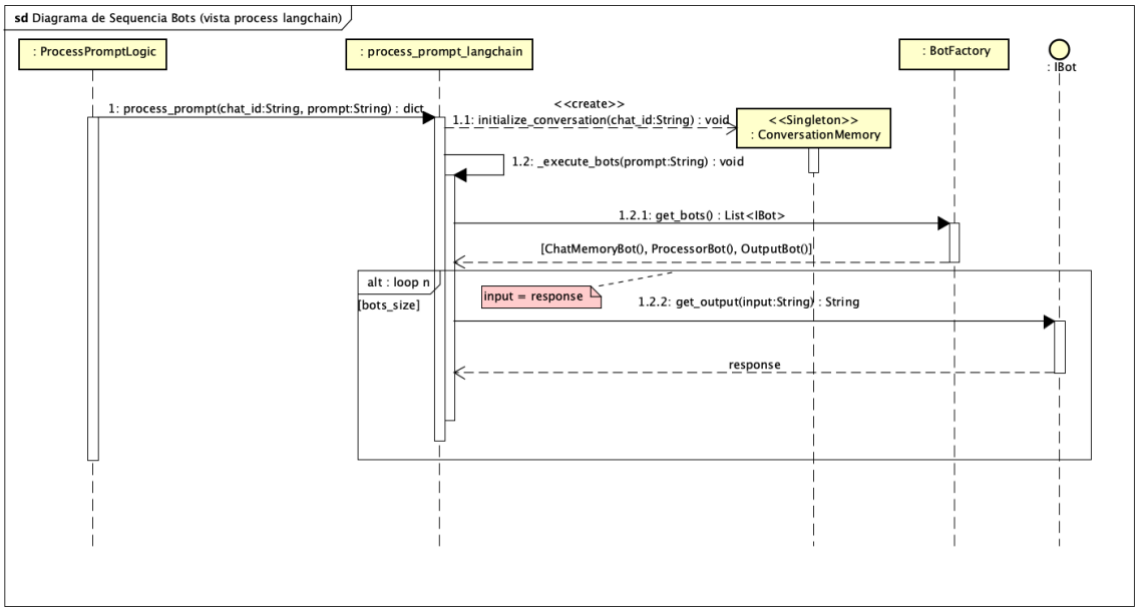


Ilustración 38- Diagrama de secuencia de la función process_prompt

La primera función es de inicializar la memoria del *chat* para el *chatId* que recibe como parámetro. Si bien más adelante se explicará a fondo que es la memoria, básicamente es lo que provee el contexto de la conversación. El parámetro *chatId* es el identificador del *chat*, necesario para distinguir entre los distintos *chats* y que el sistema sea capaz de mantener varios al mismo tiempo.

Una vez inicializada la memoria del *chat*, se ejecuta la función *execute_bots*, que se encuentra también en la clase ProcessPromptLangchain. Más adelante se ahondará acerca de que se ejecuta internamente al llamar a esta función. A grandes rasgos, se inicia un *loop* de *bots* que se comunican entre sí, usando el *prompt* del usuario, para darle la respuesta más precisa posible, considerando el contexto de la conversación, el idioma del usuario, entre otras variables.

La variable *source_documents* contiene las referencias a los documentos (en caso de que corresponda) de donde se sacó la información de la base de datos vectorial para generar una respuesta. Esto surge a partir del requerimiento de poder visualizar las fuentes para la verificación humana u obtener un contexto más completo.

Entre la respuesta generada por los *bots* y los *source_documents*, se genera un diccionario que es lo que se le entrega al *backend* para que luego el usuario lo vea en un formato visual amigable gracias al *frontend* de la aplicación.

7.4.3.6.4. *Bots Chain*

El concepto de *chain of bots* en el marco del desarrollo de sistemas de IA generativa se refiere a la creación de una cadena de *bots* que trabajan juntos para generar una respuesta coherente y relevante. Cada *bot* en la cadena se especializa en un área específica del conocimiento o del lenguaje, y se encarga de generar una parte de la respuesta. La respuesta final se genera a partir de la combinación de las respuestas de cada *bot* en la cadena.

Aunque cada *bot* en la *chain of bots* involucra un costo adicional por utilizar modelos de inteligencia artificial, se ha demostrado que esto mejora significativamente la calidad de las respuestas [56]. Los sistemas de IA generativa "públicos", como ChatGPT y Bard, en ocasiones enfrentan desafíos en cuanto a fiabilidad debido a las "alucinaciones", que son respuestas sin fundamentación en datos reales, pudiendo degradar la experiencia del usuario. Es vital minimizar estas alucinaciones y aumentar la precisión, especialmente cuando el sistema es utilizado por trabajadores en diversas organizaciones para asistir en la toma de decisiones. La finalidad de la *chain of bots* es abordar eficazmente este desafío.

Los *bots* que tiene actualmente la aplicación, surgen de tres características principales que se entiende que son fundamentales para el éxito del proyecto: la contextualización de la IA acerca de la conversación con el usuario, la precisión y calidad de la respuesta, y el formato conversacional: que la conversación sea lo más “humana” posible. A partir de cada una de las características mencionadas surge cada uno de los *bots*: ChatMemoryBot, ProcessorBot y OutputBot respectivamente. A continuación, se presenta visualmente mediante un diagrama de secuencia como es el proceso de comunicación entre los *bots*.

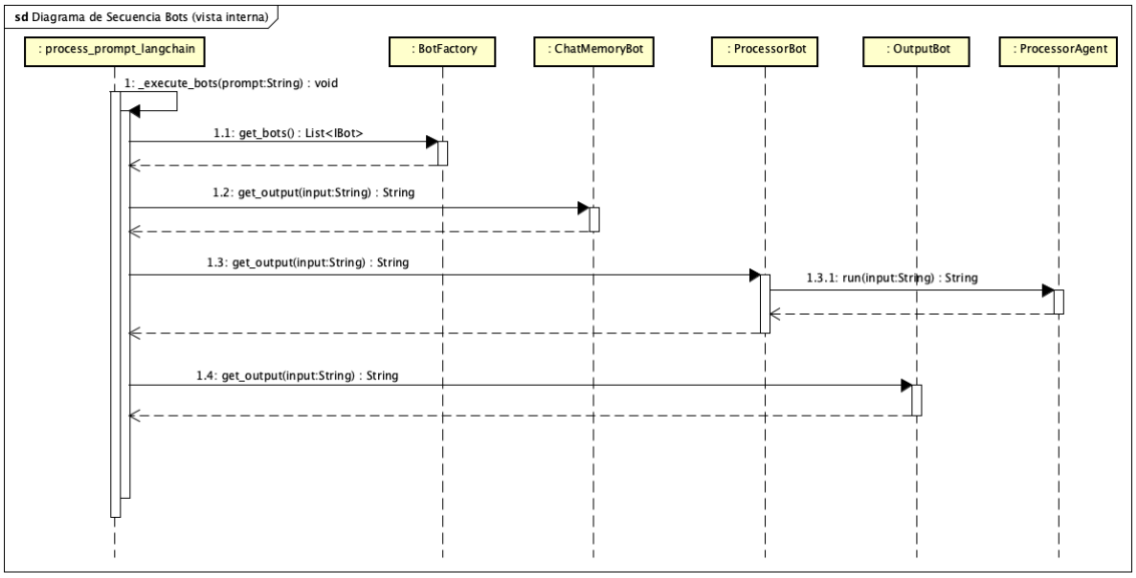


Ilustración 39- Diagrama de secuencia *Bots*

La comunicación entre los *bots* está centralizada por el método *execute_bots* de la clase ProcessPromptLangchain. Aquí se destacan dos características principales en la comunicación entre ellos:

1. El *output* de un *bot* es el *input* del siguiente *bot*. Esto quiere decir que el *prompt* de un *bot* es la *completion* del anterior. Por supuesto que el *prompt* del primer *bot* es el original solicitado por el usuario. Más adelante se explicará cómo es el formato de los *outputs* y de los *inputs* de cada *bot.*
2. Todos los *bots* conocen el *input* original del usuario. Esto tiene el mismo objetivo: aumentar al máximo posible la contextualización, la precisión de la respuesta, y la naturalidad de la conversación.

A continuación, se presentan los componentes principales de la cadena de *bots*.

Bot Factory

Dadas las características del proyecto, y la nula experiencia previa del equipo desarrollando sistemas de IA generativa, era una situación bastante probable que de antemano no se sepa con certeza que *bots* iba a ser necesario tener en el sistema, de hecho, fue algo que fue cambiando durante el desarrollo del trabajo hasta llegar a esta última versión. Por esto, se implementó la clase BotFactory.

La clase BotFactory es la clase que implementa el patrón *Factory Method*, que es un patrón de diseño creacional que proporciona una interfaz para crear objetos en una superclase, pero permite a las subclases modificar el tipo de objetos que se crearan [57]. Este patrón es particularmente útil cuando una clase no puede anticipar el tipo de objetos que necesita crear, o cuando una clase quiere sus subclases especifiquen los objetos que crea.

BotFactory es responsable de crear y devolver una lista de objetos *IBot*. Estos objetos se crean en un orden especifico, que es crucial para la *Bots Chain* funcione correctamente.

```python
class BotFactory:
    def __init__(self):
        self.bots = [ChatMemoryBot(), ProcessorBot(), OutputBot()]

    def get_bots(self):
        return self.bots
```

Ilustración 40- Clase *BotFactory*

El uso del patrón *Factory Method* en este contexto ofrece varias ventajas. En primer lugar, proporciona una ubicación centralizada para la creación de objetos IBot, lo que simplifica el mantenimiento del código y mejorar su legibilidad. Si hay que añadir un nuevo *bot* solo se hace la construcción del *bot* haciendo la implementación de IBot y se agrega a la BotFactory.

Otra ventaja, es que al encapsular el proceso de creación de los *bots* dentro de BotFactory, la clase cliente ProcessPromptLangchain se desacopla de las clases específicas del *bot*. Esto significa que la clase cliente puede trabajar con cualquier objeto *bot* que implemente la interfaz IBot, sin la necesidad de saber cómo se crean estos objetos. Este nivel de abstracción permite ampliar y modificar el sistema con un impacto mínimo en la clase cliente.

Interfaz IBot

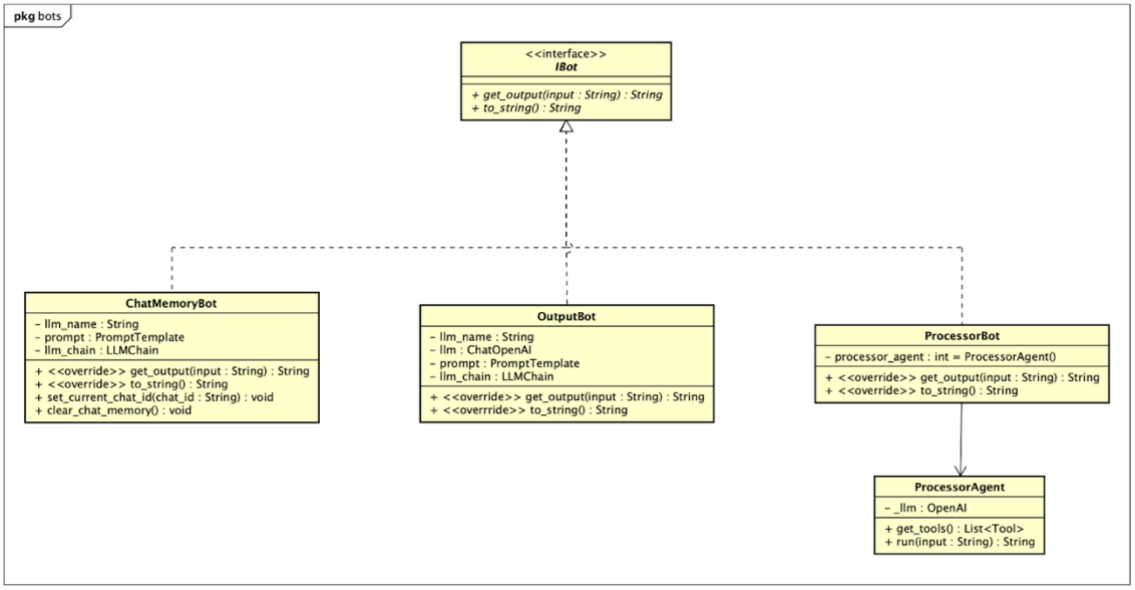


Ilustración 41- Interfaz *IBot*

De la mano con el patrón Factory Method, la clase *IBot* representa la interfaz de los *bots* del sistema. Esta interfaz tiene dos métodos a ser implementados por las clases concretas: *get_output* y *toString*. Esta última retorna el nombre del *bot*, mientras que la función *get_output* es responsable de ejecutar el *bot*. La implementación de cada *bot* es independiente y como se mencionó anteriormente, el *output* de uno es el *input* del siguiente. Las instrucciones acerca de cómo debe ser este *output*, y como es el *input* son hechas en el *prompt*, aplicando Prompt Engineering, técnica que será detallada más adelante en este capítulo.

De acuerdo a la implementación actual, el primer *bot* de la cadena es el ChatMemoryBot. Este *bot* tiene como función recibir el *prompt* original y generar un *output* que contiene información relevante, como el contexto de la conversación, entre otros, para que el siguiente *bot* pueda dar una respuesta completa.

El segundo *bot* (ProcessorBot) recibe un *input* con el mismo formato que el *output* del ChatMemoryBot y procesar la solicitud del usuario a través de un Agent (explicado a continuación) y generar una respuesta usando las *tools* (herramientas) a su disposición. Este *bot* retornará un *output* que será una respuesta intermedia del sistema al usuario a partir de su solicitud.

El tercer y último *bot*, el OutputBot, es responsable de recibir la respuesta intermedia del *bot* anterior y presentarla de una forma amigable.

ChatMemoryBot

El primer *bot* de la cadena es ChatMemoryBot. Este *bot* tiene dos responsabilidades principales que están altamente relacionadas:

- Proveer el contexto de la conversación a partir de la memoria del *chat*.
- Establecer cuál es la tarea a desarrollar por el siguiente *bot*.

Este contexto es esencial para que las respuestas de la IA sean relevantes y precisas, comunicándose con la clase ConversationMemory para interactuar con las bases de datos de memoria.

Utilizando técnicas de Prompt Engineering, ChatMemoryBot establece detalladamente qué debe hacer el siguiente *bot*, cómo debe responder y en qué formato entregar la respuesta.

Más adelante se irá al detalle de cómo es el *prompt* indicado para este *bot*. Básicamente se compone de una explicación del último historial de la conversación con el usuario, más otras informaciones relevantes de la conversación, y un formato específico en el que debe responder, compuesto por los siguientes puntos:

- Input original del usuario
- Resumen del input del usuario
- Resumen del contexto de la conversación
- Siguiente tarea (a desarrollar por el próximo bot)
- Lenguaje utilizado por el usuario

Por último, se le indica que esta respuesta no está pensada para ser entregada al usuario final, si no que para otra IA.

La *ConversationChain* de LangChain es una clase que se utiliza para llevar a cabo conversaciones entre humanos y sistemas de IA. La *ConversationChain* está compuesta por tres componentes principales:

1. LLM: modelo de lenguaje de IA
2. Prompt: serie de instrucciones para la IA

3. Memory: contexto de la conversación

```
def get_output(self, input: str):
    try:
        ConversationMemory().save_context_input(input)

        memory = ConversationMemory()._instance._memory

        conversation = ConversationChain(
            llm=self.llm, verbose=True, prompt=self.prompt, memory=memory)
        prediction = conversation.predict(input=input)

        ConversationMemory().remove_last_interaction()

        return prediction

    except Exception as e:
        print(e)
        return "Sorry, there was an error."
```

Ilustración 42- Definición de *get_output*

El método *predict* de la ConversationChain de LangChain se comunica con la IA con los ítems explicados y obtiene una respuesta, que es el *output* de este *bot* para que lo utilice el siguiente: ProcessorBot.

ProcessorBot

El ProcessorBot es llamado desde el *loop* de la cadena de *bots* y recibe como *input* el *output* del anterior. Su función es la que su nombre describe: procesar el *input* y generar una respuesta.

El ProcessorBot es únicamente responsable de inicializar y ejecutar el ProcessorAgent. Un *agent*, como se describió en el capítulo de investigación, es una entidad inteligente que actúa como intermediario entre el usuario y las herramientas disponibles en la plataforma. Los agentes utilizan un Large Language Model (LLM) para determinar qué acciones tomar y en qué orden.

ProcessorAgent

Continuando con lo presentado anteriormente acerca de que es un *agent*, es una entidad que tiene acceso a una serie de herramientas y recursos, y que utiliza esta información para tomar decisiones y realizar tareas específicas. Los agentes en LangChain se utilizan para interactuar con el mundo real y para realizar tareas automatizadas, como responder preguntas, generar texto, traducir idiomas, resumir texto, entre otras. Los agentes en LangChain se basan en Large Language Models (LLM) y utilizan técnicas de procesamiento de lenguaje natural para realizar sus tareas.

Las herramientas a las que tiene acceso, conocidas como *tools*, son las que le permiten realizar tareas específicas y especializadas. Por ejemplo, traducción de idiomas, cuentas matemáticas, etc. Estas herramientas pueden ser reusadas de las librerías de LangChain o pueden ser auto implementadas. El *agent* decide qué herramienta utilizar a partir de la descripción que se le da a la *tool* al declararla. Más adelante se verá algunos ejemplos.

Su funcionamiento básicamente se basa en recibir un *input*, tomar una decisión acerca de qué tarea debe hacer, que puede ser usar una herramienta para generar una respuesta, o darle una respuesta final al usuario, y finalmente la ejecución de esa tarea.

Para el desarrollo de Chat Your Data, se implementó un Custom MRKL Agent. Este es un tipo de *agent* que utiliza la tecnología MRKL (*Modular Reasoning, Knowledge and Language system*) para realizar tareas de procesamiento de lenguaje natural. El MRKL Agent se compone de tres partes: herramientas, LLMChain y razonamiento discreto. Las herramientas son las herramientas que el *agent* tiene disponibles para utilizar, la LLMChain es la cadena de modelos de lenguaje grandes que produce el texto que se utiliza para la generación de respuestas, y el razonamiento discreto es el proceso de toma de decisiones que utiliza el agente para seleccionar la herramienta y el modelo de lenguaje adecuados para la tarea en cuestión. Permite realizar tareas que requieren razonamiento, como la resolución de problemas, la generación de texto y la respuesta a preguntas.

Además, el MRKL *agent* es un agente autónomo porque puede tomar decisiones sin la intervención humana. Esto le permite realizar tareas de forma independiente.

Debido a la variedad de tareas que puede realizar, su uso en aplicaciones como *chatbots*, asistentes virtuales y motores de búsqueda semánticos es muy conveniente.

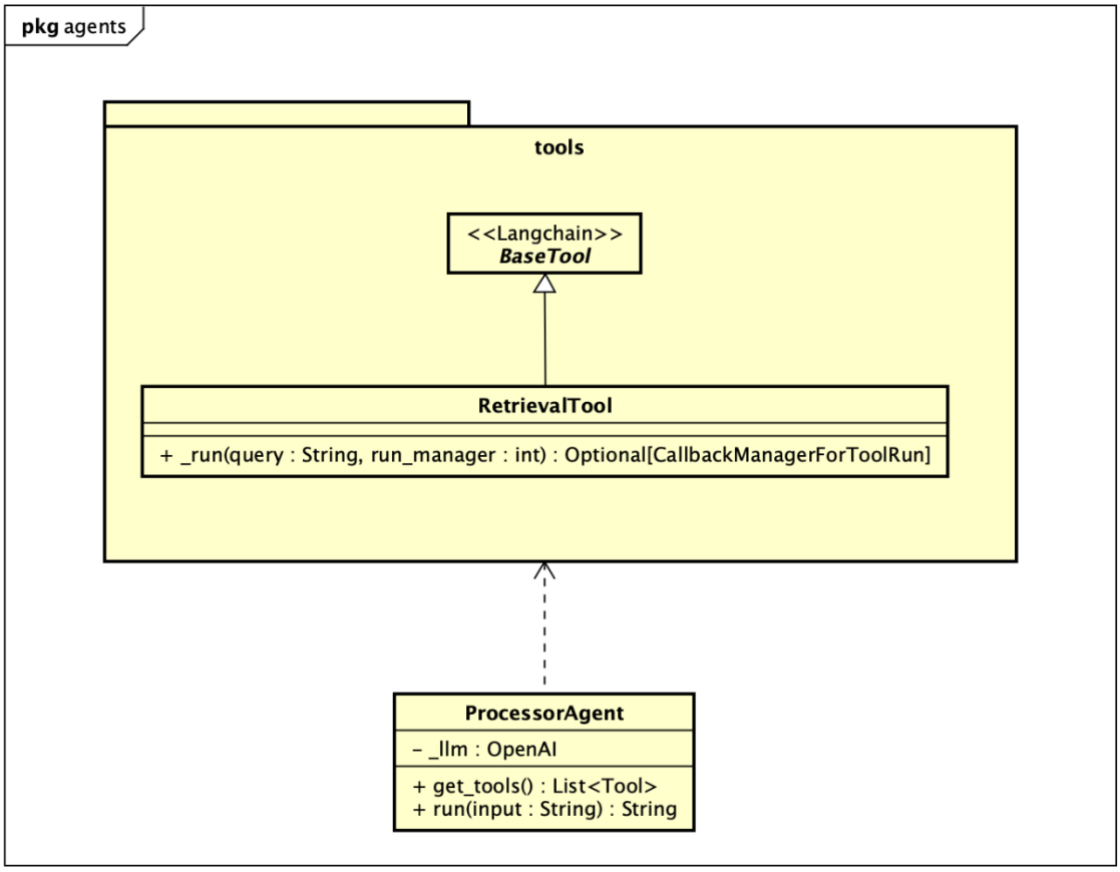


Ilustración 43- Diagrama de *Agents*

La clase *ProcessorAgent* cuenta con dos funciones; la primera de ellas es *get_tools*, donde se puede definir y obtener las herramientas a las que tiene acceso el *agent*.

```
def get_tools(self):
    load_dotenv()
    SERP_API_KEY = os.getenv("SERP_API_KEY")
    search = SerpAPIWrapper(serpapi_api_key=SERP_API_KEY)
    llm_math_chain = LLMMathChain.from_llm(llm=self._llm, verbose=True)
    return [
        Tool(
            name="Search",
            func=search.run,
            description="useful for when you need to answer questions about current events that you couldn't be able to answer with the Database Retrieval Tool."
        ),
        Tool(
            name="Calculator",
            func=llm_math_chain.run,
            description="useful for when you need to answer questions about math"
        ),
        RetrievalTool()
    ]
```

Ilustración 44- Definición *get_tools*

Las *tools* con las que cuenta el ProcessorAgent son ‘Search’, ‘Calculator’ y ‘Retrieval’:

- *RetrievalTool:* responsable de buscar en la base de datos vectorial, donde se encuentran los documentos y la información que subió la empresa u organización.

- *Search*: se usa la *SerpAPI* para búsquedas en la web. Permite que el usuario pueda consultar sobre temas que el LLM no sepa ni se encuentren en la base de datos vectorial, como noticias recientes o cualquier tipo de búsqueda a hacerse en Google. Considerando que la prioridad de búsqueda debe ser dentro de los documentos de cada organización, en la descripción de esta herramienta se declara cuando es conveniente usarla, y se hace énfasis en que solo la use si no encontró información en la RetrievalTool.
- *Calculator*: una falla conocida de los LLMs como GPT, LLaMa u otros es la capacidad de responder de forma precisa a preguntas matemáticas [58] , en los que los cálculos sean un poco complejos. Por esto, se recomienda que al desarrollar *chatbots* se implementen este tipo de herramientas para mitigar este problema.

A continuación, en la siguiente sección se discutirá sobre la *RetrievalTool*, responsable de la búsqueda en la base de datos vectorial.

Es importante destacar que la capacidad de crecer y extenderse de este componente para lograr procesar cada vez tareas más diversas resulta muy fácil, ya que simplemente se debe importar una herramienta desde LangChain o implementar una propia y agregarla en el array de *tools*. A futuro hay ciertas herramientas que tienen cierta prioridad para Chat Your Data, entendiendo que este producto será usado en el ámbito organizacional. Estas herramientas incluyen la búsqueda en una base de datos de tipo SQL, la posibilidad de resumir textos grandes, como artículos, reportes o investigaciones, y la posibilidad de analizar archivos de tipo CSV, comúnmente usados en este ámbito. En los próximos pasos del equipo, estas cuestiones serán abordadas para que el producto sea lo más completo posible.

La segunda función del ProcessorAgent es la función *run*, que básicamente ejecuta el *agent*. Esta función inicializa el agente con dos componentes principales: la lista de *tools* y una LLMChain, y luego lo ejecuta.

```
def run(self, input: str):
    tools = self.get_tools()

    prefix = ProcessorAgentPrompt.PREFIX
    suffix = ProcessorAgentPrompt.SUFFIX
    format_instructions = ProcessorAgentPrompt.FORMAT_INSTRUCTIONS

    prompt = ZeroShotAgent.create_prompt(tools, prefix, suffix, format_instructions=format_instructions, input_variables=[
                                         "input", "agent_scratchpad"])

    llm_chain = LLMChain(llm=OpenAI(temperature=0), prompt=prompt)

    tool_names = [tool.name for tool in tools]

    agent = ZeroShotAgent(llm_chain=llm_chain, allowed_tools=tool_names)

    agent_executor = AgentExecutor.from_agent_and_tools(
        agent=agent, tools=tools, verbose=True)

    return agent_executor.run(input)
```

Ilustración 45- Definición *run*

Sumado a lo explicado anteriormente, se introduce el concepto de *Chain of Thought*, que, si bien es utilizado en los otros *bots* a partir de las *prompts* correspondientes a cada uno, aquí es donde se hace especial énfasis en el uso de este.

El concepto de *Chain of Thought* se refiere a una técnica de generación de *prompts* que se utiliza en sistemas de inteligencia artificial. Consiste en proporcionar instrucciones a un LLM de manera de "paso a paso", lo que hace que las salidas o respuestas de la IA se construyan unas sobre otras, como una cadena de pensamiento. La técnica de *Chain of Thought* se utiliza para mejorar la coherencia y relevancia de las respuestas generadas por un LLM y para hacer que las conversaciones sean más naturales y fluidas. En la sección 5.3.1.2 Chain of Thought se puede conocer más acerca de esta técnica.

Este concepto es aplicado claramente en el *prompt* del ProcessorAgent, dividida en *prefix*, *suffix* y *format_instructions*:

```
FORMAT_INSTRUCTIONS = """Use the following format:

    Question: the input question you must answer
    Thought: you should always think about what to do
    Action: the action to take, should be one of [{tool_names}]
    Action Input: the input to the action. Be sure to add all the information in a very detailed manner from the "Next Task" provided in the "Conversat
    Observation: the result of the action
    ... (this Thought/Action/Action Input/Observation can repeat N times)
    Thought: I now know the final answer
    Final Answer: the final answer to the original input question"""
```

Ilustración 46- *Format Instructions*

Además de esto, se puede distinguir que al momento de crear el *prompt* para el *agent* se distinguen dos variables de entrada; el *input* (que como se mencionó anteriormente es el *output* del ChatMemoryBot, que contiene el contexto y la tarea a realizar por este *bot*), y el *agent_scratchpad*. El *agent scratchpad* es una herramienta que se utiliza en LangChain para almacenar información temporal durante una conversación entre un usuario y un agente. El

scratchpad es una especie de pizarrón o cuaderno virtual donde el *agent* escribe notas, ideas o cualquier otra información relevante que pueda ser útil para responder a la solicitud del usuario. Es útil en conversaciones largas o complejas, donde puede ser difícil recordar toda la información relevante, pero es especialmente favorable en casos como el del *ProcessorAgent* de la plataforma, donde es utilizado para almacenar información que se ha obtenido de fuentes externas, como bases de datos o motores de búsqueda. En la sección 5.7.2.1 Agent Scratchpad se puede obtener más información.

RetrievalTool

La RetrievalTool es la responsable de buscar información relacionada a lo que pide el usuario en la base de datos vectorial. Implementa la BaseTool, que es una interfaz que provee LangChain para la creación de *tools* [59].

```
class RetrievalTool(BaseTool):
    name = "RetrievalTool"
    description = """useful for when you need to search information you don't know about.
    Note that in case there is nothing related to the user question,
    the tool will return 'I did not find anything'. You should give priority to using it."""
    llm_name = GPT_3_TURBO
    llm = ChatOpenAI(model_name=llm_name, temperature=0)
    prompt = PromptTemplate.from_template(RETRIEVAL_PROMPT)

    def _run(self, query: str, run_manager: Optional[CallbackManagerForToolRun] = None):

        # Run chain
        qa_chain = RetrievalQA.from_chain_type(
            llm=self.llm,
            retriever=VectorDataBaseFactory.vector_database().get_retriever(),
            return_source_documents=True,

            chain_type_kwargs={"prompt": self.prompt},
            verbose=True,
        )

        result = qa_chain({"query": query})
        # TODO: If the retrieval dint't found any information
        SourceDocuments()._instance.save_source_documents(
            result["source_documents"])

        source_documents = result["source_documents"]

        if len(source_documents) == 0:
            return query

        return result["result"]
```

Ilustración 47- Clase *RetrievalTool*

La principal distinción que se puede hacer en los componentes de la clase RetrievalTool es la descripción. Como se explicó anteriormente, la descripción en una *tool* es la instrucción que se

le da al *agent* sobre cuándo debe usarla. En este caso, se agrega dos consideraciones que debe tener. La primera es una indicación acerca de cómo responde la herramienta en el caso de que no encuentre información relacionada en base al *prompt* del usuario. La segunda, es que se le debe dar prioridad al uso de la misma, como se señaló previamente.

Para la búsqueda en la base de datos vectorial se utiliza la clase *RetrievalQA*, que se usa para recuperar documentos de un Retriever y luego usar *chain* de tipo QA para responder una pregunta basada en los documentos recuperados. Internamente, realiza el siguiente proceso:

1. Recibe el *input* del usuario y lo vectoriza mediante el proceso de *embedding*.
2. Busca la coincidencia de este vector en la base de datos vectorial, usando como *retriever* el que se indique (en este caso ChromaDB).
3. Genera una respuesta usando un LLM.

El Retriever que se usa en esta clase es el que implementa la clase ChromaRepository, en la función *get_retriever():*

```
def get_retriever(self):
    return self.vectordb.as_retriever(search_type="similarity_score_threshold", search_kwargs={'score_threshold': 0.6})
```

Ilustración 48- Definición *get_retriever*

Aquí se especifica que tipo de búsqueda se debe utilizar y que nivel de similitud es el mínimo para considerar al fragmento de texto o al documento como relacionado a lo que pide el usuario, de acuerdo a una puntuación que se le asigna.

La variable return_source_documents asignada a verdadero, es la indicación de que se quiere obtener los fragmentos de texto de los documentos de donde se obtuvo la información para generar la respuesta que generó la QA *chain*. Como se mencionó anteriormente, esto es importante para los usuarios que quieran profundizar sobre la respuesta que dio el *chatbot*.

Cuando se obtiene la respuesta, se guardan en la instancia de SourceDocuments (detallado en las siguientes secciones) que irán incluidos en la respuesta que se le dará al usuario.

En un principio, se entendió que la acción de la búsqueda en la base de datos vectorial debería ser un *bot* más dentro de la *chain of bots* explicada previamente, ya que se consideraba que era un proceso fundamental en la generación de la respuesta de Chat Your Data. A partir de ver la experiencia de los usuarios del *beta tester*, el BID (Banco Interamericano de Desarrollo), el

equipo se dio cuenta que muchas veces se estaba haciendo una consulta a la base de datos vectorial cuando no era necesario, ya que las *prompts* que ingresaban los usuarios no incluían preguntas relacionadas a información existente en la base de datos vectorial. Esto generaba que el tiempo de respuesta del sistema sea más largo, y que los *bots* alucinen inventando información que no era correcta. Por estas razones, se decidió transformarlo en una Tool, donde el ProcessorAgent es quien decide si debe recurrir a la herramienta o no.

OutputBot

El OutputBot es el responsable de generar una respuesta final, que será la *completion* al *prompt* ingresado por el usuario. Este *bot* es la última instancia de la *chain of bots*, por lo que recibe como input el output del ConversationBot, para comprender el contexto de la conversación, y el *output* de *ProcessorBot*, para contar con la información necesaria para dar respuesta completa y precisa.

El funcionamiento del *bot* se basa en una LLM chain (explicada anteriormente), especificándole un *prompt* donde se hace uso de la técnica de *Chain of Thought*, para maximizar la calidad de la respuesta. La única diferencia, es el LLM que se usa para correr la *chain* de este *bot*. En este caso, se optó por el uso de GPT-4, a pesar de su costo mayor a GPT-3.5, usado en el resto del sistema. Esta decisión se fundamenta en que este *bot* cobra especial relevancia, considerando que lo que genere será la respuesta que vea el usuario que generó Chat Your Data.

Además de esto, se destaca que el OutputBot también es responsable de llamar a la función *save_context* de la ConversationMemory, explicada a continuación. Este método cumple con la función de guardar la interacción entre el usuario y la IA, almacenando el *prompt* original del usuario, y la *completion* final generada por el sistema.

7.4.3.6.5. *Memory*

El concepto de memoria en el área de IA generativa significa la habilidad que tiene un sistema (en este caso un *chat*) de recordar la conversación pasada con un usuario. Recordemos que el objetivo de esta herramienta es que la conversación sea lo más “humana”, natural y fluida posible, por lo tanto, resulta fundamental que el sistema recuerde de que se viene hablando mientras el usuario hace preguntas o ingresa distintos *prompts*.

Un sistema de memoria en IA tiene dos acciones básicas: lectura y escritura. El flujo básico es el siguiente:

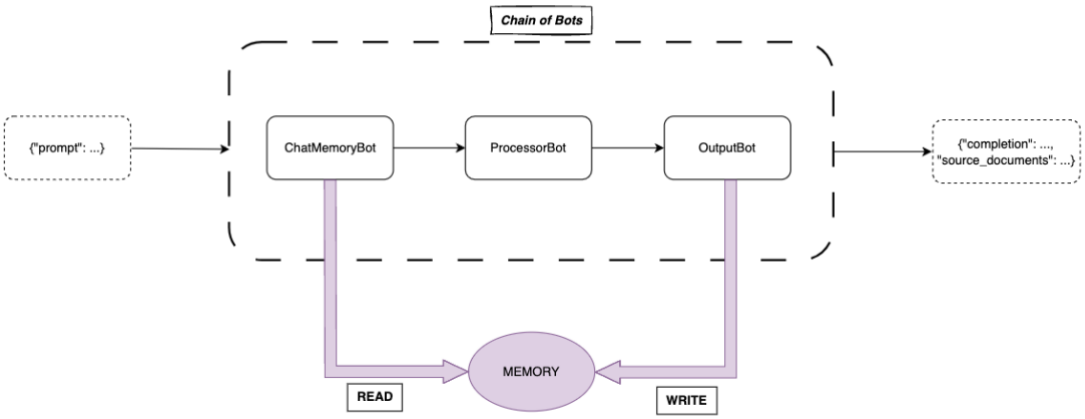


Ilustración 49 - Flujo de lectura y escritura en memoria

En este caso, la lectura se ejecuta antes de correr la *chain* del ChatMemoryBot, y la escritura se ejecuta en el último *bot* de la *chain of bots* antes de darle una respuesta final al usuario, luego de ser generada la respuesta del OutputBot.

Dentro del componente PluginAI, la memoria está gestionada por el paquete *memory*, que contiene la clase ConversationMemory, responsable de interactuar con todo el sistema de memoria de Chat Your Data.

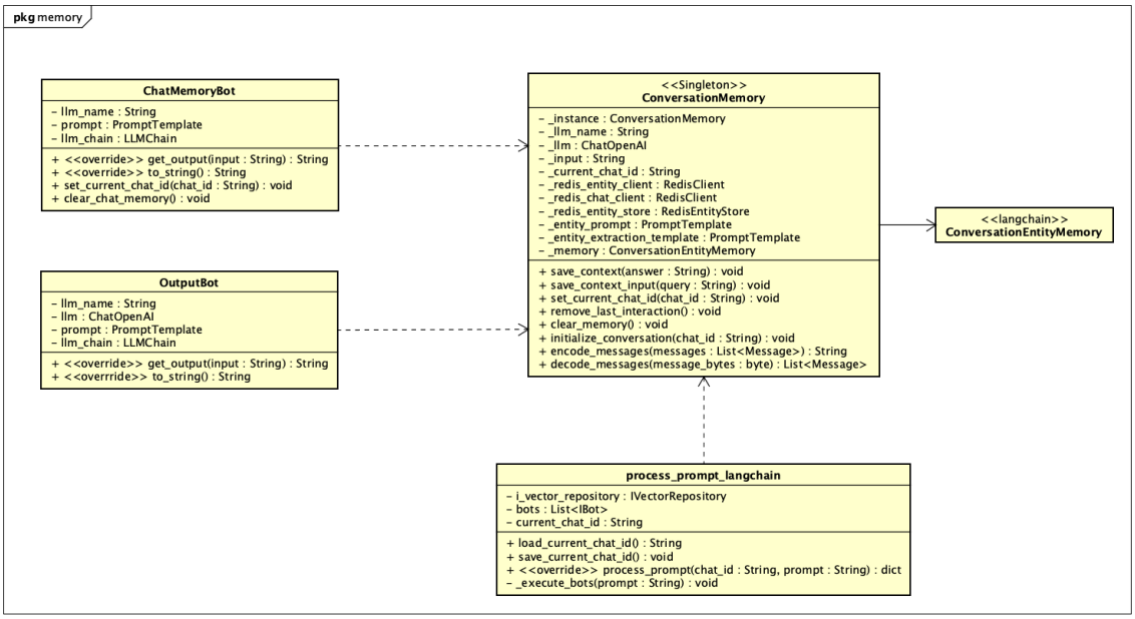


Ilustración 50- Diagrama de *memory*

Como se mencionó previamente, ChatMemoryBot y OutputBot acceden a las funciones de la clase ConversationMemory. De la misma forma lo hace la clase ProcessPromptLangchain, que

mediante la función *initialize_conversation()*, ingresa el *current_chat_id*, para que el sistema pueda consultar a la memoria de cada *chat* cuando se interactúa con el mismo.

El tipo de memoria que se utiliza en Chat Your Data es ConversationEntityMemory que provee LangChain. La ConversationEntityMemory es un tipo de memoria que almacena información sobre el estado actual de la conversación. Esta información incluye dos características relevantes:

- Las últimas interacciones entre el usuario y la IA, para seguir el hilo de la conversación y generar respuestas relevantes.
- Las entidades que se han mencionado a lo largo de toda la conversación. Se guarda un resumen de cada entidad que se menciona en el *chat*

La *memory* comunica información al ChatMemoryBot a través del *prompt*, pero enfrenta limitantes como costo de almacenamiento, performance y, crucialmente, el límite de *tokens* en los LLMs. Aunque idealmente guardaría todo el historial de conversación, para manejar estas restricciones, la ConversationEntityMemory se utiliza para retener textualmente las últimas 3 interacciones y las entidades mencionadas durante toda la conversación, asegurando un contexto reciente sin exceder los límites de *tokens*.

Las entidades son aquellas personas u objetos que sean mencionados en la conversación. A modo de ejemplo, a partir del *prompt* :"Hola! Soy Nacho y tengo dos perros, Paco y Holly! Paco es un Golden y Holly es un San Bernardo", se generan las siguientes entidades:

Entidad	Descripción
Golden	Nacho has a golden retriever named Paco.
Paco	Paco is a male golden retriever owned by Nacho.
Nacho	Nacho is a person who owns two dogs named Paco, a golden retriever, and Holly, a San Bernardo.
Holly	Holly is one of Nacho's two dogs, a female San Bernardo.
San Bernardo	Holly, owned by Nacho, is a female San Bernardo.

Tabla 5- Tabla de entidades

En la clase ConversationMemory se implementa el patrón Singleton, con el objetivo de garantizar que esta clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. Además, contiene una variable global que contiene el *chatId* actual llamada *current_chat_id*. El objetivo de esta variable es asegurar que el sistema acceda a la memoria correspondiente al *chatId* con el que se está interactuando en el momento.

La ConversationEntityMemory de Chat Your Data se nutre de dos bases de datos distintas, ambas implementadas usando la tecnología Redis y siguiendo el formato de clave-valor:

1. Base de datos de entidades: almacena entidades mencionadas en el *chat* y las resume mediante ConversationEntityMemory. Un LLM recibe instrucciones sobre como identificar y resumir entidades del *input* del usuario, generando un diccionario de entidades y sus descripciones para almacenar en la base de datos. Cada entidad, identificada por un *string* combinado con el *chatId* y el nombre de le entidad, se actualizará conforme se menciona, incorporando nueva información.

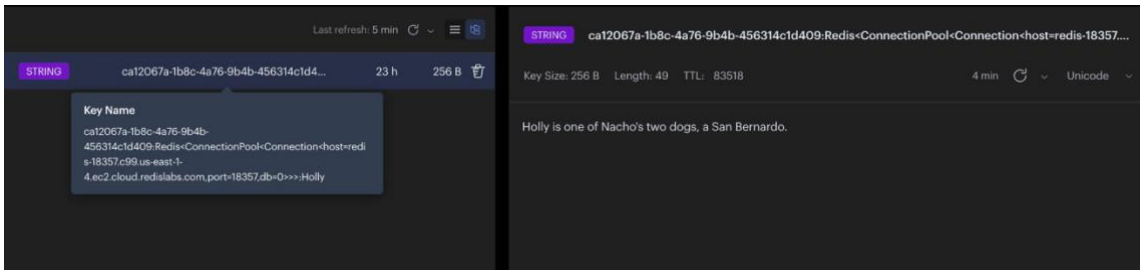


Ilustración 51- Base de datos de entidades

2. Base de datos de interacciones: almacena las últimas tres interacciones entre el usuario y la IA de cada *chat*. En este caso, la clave es el *chatId*, y el valor es un objeto de tipo JSON que contiene el *prompt* ingresado por el humano y la *completion* generada por la IA.

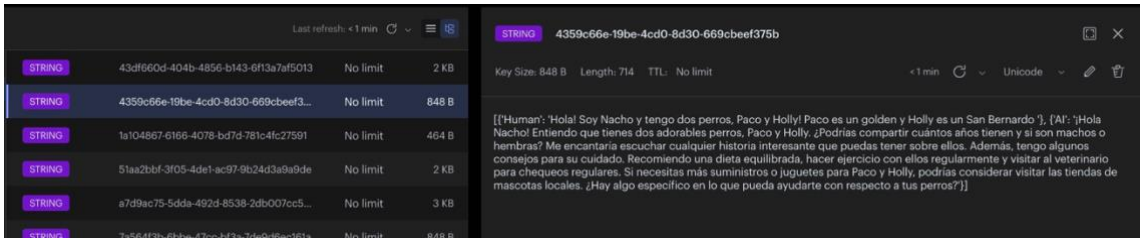


Ilustración 52- Base de datos de interacciones

7.4.3.6.6. Source Documents

Como requerimiento de negocio, surge la necesidad de referenciar los documentos de los cuales la IA obtuvo la información para generar una respuesta a el *prompt* del usuario. Para esto fue

necesario que, al momento de consultar en la base de datos vectorial, se ingrese la variable *return_source_documents* en true. Esta información específica se guarda en la clase SourceDocuments, que es la responsable de la gestión de esta información que luego será leída en *process_prompt_langchain* para dar una respuesta final al usuario.

La clase SourceDocuments implementa el patrón Singleton, con el objetivo de asegurar que una clase tenga una única instancia y proporciona acceso global a ella.

7.4.3.6.7. *Prompts*

Los *prompts* juegan un papel crucial en la *chain of bots* y al interactuar con LLMs al servir como instrucciones que incitan respuestas de los sistemas de IA, guiando y personalizando su generación de texto. La técnica de Prompt Engineering busca mejorar la calidad de estas respuestas a través del diseño de *prompts* que conduzcan efectivamente los procesos del modelo, permitiendo respuestas más pertinentes y precisas. En la sección 5.3.1.1 Prompt Engineering se puede profundizar sobre este tema.

Para favorecer la mantenibilidad del sistema y el orden del código, se optó por separar en un paquete llamado *prompts* los textos de entrada para guardar los *prompts* de cada llamada a un LLM y luego importarlos en cada lugar que sea necesario.

El paquete *prompts*, el cual se puede encontrar en el anexo 13.25.7 Paquete de Prompts, tiene 5 *prompts*:

- RetrievalPrompt: se utiliza en la RetrievalTool para darle instrucciones al *retriever* de Chroma acerca de qué información debe buscar en la base de datos vectorial y como presentar la respuesta para el siguiente *bot*.
- EntityPrompt: contiene las instrucciones acerca de cómo extraer y resumir las entidades que se encuentran en una conversación para ser posteriormente guardadas en la base de datos de entidades.
- ChatMemoryPrompt: explica las instrucciones que debe usar la *chain* del ChatMemoryBot para recibir el input original del usuario y el contexto conversacional para procesarlo y generar el output de este *bot* en el formato que también se especifica en el ChatMemoryPrompt.
- OutputPrompt: contiene la información acerca de cómo generar una respuesta final a partir del input original del usuario y lo generado por los *bots* anteriores.

- ProcessorAgentPrompt: contiene las instrucciones acerca de cómo utilizar las *tools* para procesar el input del usuario usando como referencia el contexto de la conversación.

Los *prompts* son generados utilizando la librería de LangChain llamada PromptTemplate. Esto es una herramienta que se utiliza para crear *prompts* con entradas dinámicas de manera más fácil y eficiente. El *prompt template* es una plantilla fija que se utiliza para crear *prompts* personalizados y precisos. La plantilla se compone de un texto básico y una serie de variables que se pueden reemplazar con entradas dinámicas. A modo de ejemplo, en el ChatMemoryPrompt estas variables son: el *prompt* original del usuario, el historial conversacional y las entidades relacionadas.

7.4.3.6.8. Discusión de decisiones de implementación de IA

Durante el desarrollo, las decisiones relacionadas con la implementación de la Inteligencia Artificial (IA) surgieron como una de las áreas más críticas y desafiantes. Estas decisiones, fruto de profundos análisis y reflexiones, tuvieron un impacto directo en la funcionalidad, eficiencia y adaptabilidad del sistema.

Uno de los primeros desafíos que se abordó fue la definición de parámetros para el *Chunk Splitter*. La determinación del tamaño del fragmento (*chunk size*) y el solapamiento entre fragmentos (*chunk overlap*) no fue trivial. Ambos parámetros jugaron un papel crucial en determinar cómo se procesaría y entendería la información durante el proceso de *embedding*, impactando tanto en la velocidad de procesamiento como en la calidad del análisis contextual.

La decisión sobre el ChatMemory también resultó ser vital. La forma en que se conserva el contexto de las conversaciones afecta directamente la capacidad del sistema para ofrecer respuestas coherentes y relevantes a lo largo del tiempo. La estructuración y gestión del ChatMemory debía asegurar una interacción fluida, manteniendo a la vez la relevancia de la información pasada y presente.

Otro punto fue la implementación del Agent. El agente tiene la responsabilidad de procesar y elegir qué herramientas (*Tools*) se tienen que utilizar dependiendo del *prompt* del usuario. Existen varias implementaciones de Agent, cada uno con distintas fortalezas y debilidades.

Finalmente, el diseño y la elección de los "tipos de *chains*" dentro de Chat Your Data fueron fundamentales. Estas cadenas determinan cómo se procesa y se relaciona la información, actuando como la columna vertebral del análisis y la generación de respuestas.

Chunk splitter, chunk size y chunk overlap

La fragmentación "*chunk*" es el proceso de dividir grandes fragmentos de textos en segmentos más pequeños. Es una técnica que ayuda a optimizar la relevancia del contenido que se busca en una base de datos vectorial. La idea de la fragmentación es asegurarse que se está haciendo *embedding* a una pieza del contenido con el menor ruido posible y que sigue siendo semánticamente relevante.

El establecimiento de un tamaño de fragmento y el tipo de fragmento a utilizar requiere tener en cuenta varios factores. Para ello, el equipo se planteó las siguientes preguntas:

- ¿Cuál es la naturaleza del contenido que se indexa?
    - Documentos en cantidad y grandes siguiendo un formato estructurado para que pueda ser entendido por personas.
- ¿Qué modelo de *embedding* se está utilizando?
    - Text-Embedding-Ada-002xt
- ¿Cuáles son las consultas que harán los usuarios?
    - Consultarán sobre información que puede estar incluida dentro de varios documentos.
- ¿Cómo se utilizarán los resultados obtenidos?
    - Van a ser procesados por un LLM para darle una respuesta final al usuario.

A partir de estas preguntas, se determinaron el tamaño de fragmento y el tipo de fragmento a utilizar.

Tipo de Fragmento

Existen diferentes métodos para realizar el proceso de *chunking*, y cada uno de estos son apropiados dependiendo de la situación.

En Chat Your Data se utilizó de Recursive Chunking debido garantiza una estructura jerárquica, que puede ser beneficiosa cuando la información se encuentra en capas o subtemas. De esta manera, logra dividir un tema complejo en los subtemas que lo componen, cada uno de los cuales puede dividirse aún más si es necesario.

Por otro lado, permite la preservación del contexto, en donde en los temas detallados, el contexto es crucial para entender la información. El Recursive Chunking, por su naturaleza, preserva el contexto mejor que otros métodos que podrían cortar límites importantes en el texto.

También, mejora la comprensión al dividir temas complejos en trozos más pequeños y manejables.

Tamaño de Fragmento

Los parámetros que resumen el tamaño del fragmento son "*CHUNK_SIZE_EMBEDDING"* y *"CHUNK_OVERLAP"*. El parámetro "*CHUNK_SIZE_EMBEDDING"* determina el número de caracteres que comprenderá cada fragmento, mientras que *"CHUNK_OVERLAP"* designa la cantidad de caracteres comunes entre fragmentos adyacentes. Estas configuraciones tienen como objetivo principal segmentar grandes cantidades de texto en unidades más manejables, asegurando que no pierdan características relevantes en los límites entre fragmentos.

Para el caso de Chat Your Data se utilizó un tamaño de fragmento de 1500 y un solapamiento de 150.

Beneficios

- Conservación del contexto: Un fragmento de 1500 asegura que se incluya más contexto en cada segmento. Esto es crucial para comprender el significado semántico del texto. La superposición de 150 caracteres garantiza una transición coherente entre fragmentos, minimizando la perdida de contexto.
- Adecuado para textos estructurados: En base a los documentos que en su mayoría se van a subir para el uso de Chat Your Data contienen una estructura bien definida.
- Optimización para modelos: Para el modelo de Text-embedding-ada-002 el cual se utiliza se ofrecen mejores resultados con fragmentos más extensos.

Desafíos

- Inadecuado para textos menos estructurados: Textos como novelas o libros podrían no beneficiarse de fragmentos tan extensos y, en cambio, podrían requerir divisiones más pequeñas.
- Eficiencia en contenidos cortos: En el caso de contenidos breves, como tuits o mensajes instantáneos, un tamaño de fragmento grande podría resultar redundante o ineficiente.

Tipo de ChatMemory

Un atributo de calidad esencial es la contextualización. Para lograrla, se desarrolló la capacidad de conservar la conversación entre el usuario y "Chat Your Data".

Como se discutió en capítulos previos, para que un LLM conserve el contexto de una conversación, es imperativo incluirlo en el *prompt*. No obstante, existe un límite en la longitud del *prompt*, lo que impide incorporar una conversación completa. Aunque se pueden incluir las interacciones más recientes, si el usuario consulta sobre un tema previamente discutido, esa información no se considera, resultando en la pérdida de contextualización.

Para abordar este desafío, se optó por emplear el concepto de Entities (Entidades). Las entidades son resúmenes de los conceptos o temas clave discutidos durante la conversación. Así, a medida que avanza la conversación, se generan distintas entidades, y dentro de estas, se crea un resumen de lo más relevante. La generación de estas entidades requiere un LLM especializado en sintetizar la información, enfocándose en los puntos esenciales. Esto permite condensar más información en menos palabras, optimizando el uso del espacio en el *prompt*.

Por lo tanto, al formular un *prompt*, un LLM determina qué entidades deben incluirse para que el ChatMemoryBot, encargado de comprender el contexto conversacional, pueda ofrecer la mejor contextualización posible. De esta manera, se asegura que el ChatMemoryBot contenga solo las entidades esenciales, eliminando el ruido de entidades irrelevantes y reduciendo el riesgo de interpretaciones erróneas. Además, esto permite aprovechar los *tokens* disponibles para información más pertinente.

Adicionalmente, se reconoció la importancia de conservar las últimas tres interacciones. Esto permite al ChatMemoryBot comprender el tema de discusión anterior y contar con información adicional, complementando las entidades como resúmenes.

Custom agent

Los *agents* permiten tener una cadena de LLMs que va a depender exclusivamente del *prompt* del usuario. De esta manera, el *agent* a partir de un conjunto de *Tools*, puede decidir a qué *Tool* llamar dependiendo del *prompt* que haga el usuario. Este es el principio fundamental detrás del concepto de Agent.

Existen diversos tipos de *Agents*, y la elección de uno sobre otro depende del contexto o escenario especifico en el que se vaya a implementar.

El *Agent* utilizado para Chat Your Data es un *Custom MRKL* agent, en donde MRKL es un acrónimo de *Modular Reasoning, Knowledge & Lenguaje System* (Sistema Modular de Razonamiento, Conocimiento y Lenguaje). Esta arquitectura modular fusiona LLMs con fuentes de conocimiento externo y capacidades de razonamiento discreto, lo que resulta ideal para el caso de Chat Your Data.

El principal valor de este agente radica en su sistema MRKL, que está diseñado para gestionar consultas de alta complejidad. Su versatilidad es notable, ya que puede manejar consultas que involucran múltiples herramientas, convirtiéndolo en una opción ideal para tareas que demandan una variedad de funcionalidades.

Tipos de *chains*

Para el desarrollo de Chat Your Data el utilizar de manera aislada los LLM puede resultar insuficiente cuando se busca construir un *software* de mayor complejidad, donde es necesario encadenar varios LLMs para lograr una funcionalidad avanzada.

Las cadenas o *chains* permiten amalgamar varios LLMs para concebir una aplicación unificada y más potente. Para ello, LangChain provee varios tipos de cadenas, siendo LLMChain uno de los más comunes y utilizados.

El LLMChain es una secuencia de llamadas a un LLM, donde cada llamada en la cadena está determinada por una plantilla de *prompt* y la salida de la llamada anterior.

Por otro lado, RetrivalQA es otro tipo de cadena que ha sido diseñado específicamente para tareas de preguntas a respuestas. Este tipo de cadena utiliza un *retriever* para obtener documentos relevantes basados en *prompt* del usuario y, posteriormente, emplea un LLM para generar una respuesta basada en esos documentos.

En el marco de Chat Your Data, RetrievalQA se presenta como una herramienta esencial para gestionar las consultas sobre los documentos subidos por los usuarios, permitiendo que el sistema recupere información pertinente de los documentos y genere respuestas precisas y contextualizadas a las preguntas de los usuarios.

7.5. Seguridad

7.5.1. Mecanismos de autenticación

La garantía de autenticación es un pilar fundamental en el ámbito de la seguridad de un sistema. En el proyecto Chat Your Data, la integridad del acceso está asegurada a través de Firebase Authentication en la interfaz de usuario. Cuando un individuo pretende ingresar al sistema, se le requiere proporcionar sus credenciales, que, una vez verificadas, generan un *token* de autorización. Este distintivo digital, que actúa como evidencia de la identidad del usuario y sus respectivos permisos, se comunica al *backend* en todas las solicitudes subsiguientes. Aquí, una herramienta denominada FirebaseGuard se encarga de examinar la validez del *token*. En el caso de una verificación exitosa, no solo se otorga acceso a las funciones solicitadas, sino que el sistema también puede reconocer y monitorizar las acciones del usuario en cuestión.

7.5.2. Seguridad de la información en tránsito

Aunque la implementación de medidas avanzadas de cifrado aún está pendiente, es importante recalcar que la validación mediante FirebaseGuard proporciona una protección robusta en la transmisión de datos entre el *frontend* y el *backend*. Además, es imperativo considerar la adopción de protocolos como HTTPS en futuras iteraciones, para asegurar que toda la información intercambiada esté cifrada y resguardada de posibles interceptaciones.

7.5.3. Disponibilidad y Monitorización del Sistema

Una alta disponibilidad es vital para garantizar que los usuarios tengan acceso constante y confiable al sistema. Chat Your Data ha incorporado un mecanismo de verificación de salud mediante un *endpoint* llamado *health* en el *backend* y en el PluginAI. Este punto de acceso actúa como un barómetro de la salud del sistema: si al consultar este *endpoint* se recibe un código 200, es una indicación clara de que el servidor en cuestión está en línea y funcionando de manera óptima. Además, implementar herramientas de monitorización y alertas en tiempo real puede ser una adición valiosa para futuras versiones, asegurando una respuesta rápida ante cualquier irregularidad o fallo potencial.

7.6. *Deployment* en AWS

Para garantizar la accesibilidad y pruebas del sistema por parte del BID y otros usuarios, el proyecto Chat Your Data fue desplegado en Amazon Web Services (AWS). A pesar de la inicial

falta de experiencia del equipo en este ámbito, se logró una implementación funcional y optimizada del sistema en la plataforma de nube.

Dentro del anexo 13.26 Deployment en AWS se encuentra la implementación de Chat Your Data dentro de AWS.

7.7. Consideraciones futuras

A partir del trabajo realizado y de haber trabajado en la arquitectura con todas las consideraciones explicadas en las secciones anteriores, el equipo entiende que hay ciertos aspectos que deben ser mejorados o implementados pensando de cara al futuro del emprendimiento.

7.7.1. Estrategia para el alojamiento de documentos

El primero de ellos es el alojamiento de documentos. Como se explicó anteriormente, el sistema guarda los documentos en el repositorio propio. Las razones fueron explicadas en detalle anteriormente, pero esta fuente de documentos debería ser un repositorio externo o estar alojados en una base de datos documental en la nube.

Esta consideración se basa en la necesidad de mejorar la escalabilidad, redundancia, disponibilidad y acceso remoto de los documentos, así como aprovechar las funcionalidades avanzadas que ofrecen las soluciones de almacenamiento en la nube.

La migración hacia un repositorio externo requeriría una planificación cuidadosa. Será esencial llevar a cabo una evaluación detallada de las necesidades específicas de almacenamiento y acceso de la plataforma, así como de las implicaciones en términos de seguridad y costos. Esta posible evolución en la gestión de documentos refleja el compromiso de Chat Your Data con la adaptabilidad y la mejora continua de su arquitectura para satisfacer las demandas cambiantes de los usuarios y las organizaciones.

7.7.2. Sistema concurrente pudiendo soportar muchos usuarios al mismo tiempo

Uno de los desafíos clave en la evolución de Chat Your Data es la capacidad de soportar eficazmente un gran número de usuarios concurrentes. Si bien la arquitectura actual ha sido diseñada para un uso eficiente, el crecimiento futuro de la plataforma podría requerir mejoras

significativas en términos de escalabilidad y rendimiento para mantener una experiencia de usuario óptima.

La planificación para el futuro en este sentido implica la implementación de técnicas avanzadas de escalabilidad horizontal y vertical. Esto podría incluir la distribución de cargas de trabajo a través de múltiples servidores o la adopción de servicios de administración de recursos en la nube para garantizar la disponibilidad y el rendimiento. Además, se debería considerar la implementación de estrategias de almacenamiento en caché y optimización de consultas para minimizar la carga en la base de datos y acelerar las respuestas en tiempo real. La planificación también abordaría la supervisión constante del rendimiento y la escalabilidad del sistema, lo que permitiría una expansión proactiva a medida que la demanda de usuarios aumente.

7.7.3. Gestión de usuarios y organizaciones

Una de las futuras expansiones fundamentales de Chat Your Data se centra en la administración de organizaciones y usuarios. Para ello, se planea la implementación de una base de datos SQL robusta y escalable que permita el almacenamiento eficiente de información relacionada con múltiples organizaciones, sus usuarios y roles. Esta base de datos proporcionará una estructura organizada para gestionar la información de manera más efectiva, facilitando la creación, modificación y eliminación de organizaciones y la asignación de usuarios a estas. Además, se incorporará una capa de lógica de negocio dedicada para garantizar la coherencia y seguridad de los datos, así como la gestión de autorizaciones y autenticación de usuarios.

Esta información que se empezará a guardar constituye la piedra fundamental del conocimiento de los clientes y potenciales clientes. Guardar la información de los clientes, obteniendo información sobre el uso de la aplicación, la información de las organizaciones, los usuarios, etc. genera el potencial de analizar estos datos para obtener métricas relacionadas y así mejorar el servicio.

Por último, esta información es la clave para generar un diferencial en el negocio: la alta contextualización para las respuestas del *chat*. El objetivo será que a partir de la información que se tenga de los usuarios en la base de datos recientemente mencionada, se pueda personalizar la respuesta del *bot* mediante Prompt Engineering, contextualizando al *chain of bots* sobre la información del usuario: nombre, rol, área, especialización, entre otros, así como la información de la organización: nombre, descripción, actividad, industria, etc. Esto generará

que las respuestas del sistema sean aún más personalizadas, contextualizadas y precisas, finalizando en una mejor experiencia de los clientes.

7.7.4. Alojamiento de base de datos vectorial

Otro aspecto crítico de la planificación futura de Chat Your Data se refiere al alojamiento de la base de datos vectorial. Actualmente, la plataforma almacena los vectores de documentos en el repositorio propio. Sin embargo, a medida que la plataforma crece y se expande, surge la necesidad de considerar opciones más escalables y flexibles para el almacenamiento de estos datos. En un futuro próximo, se planea migrar hacia una solución de alojamiento externo o en la nube, lo que permitirá gestionar grandes volúmenes de datos de manera más eficiente y garantizar la disponibilidad de la información.

Esta migración hacia un alojamiento externo requerirá una cuidadosa planificación y la adopción de las mejores prácticas de seguridad y rendimiento. Además, se buscarán soluciones que faciliten la administración y el acceso a los datos vectoriales, lo que contribuirá a mejorar la velocidad de respuesta de la plataforma y la capacidad de procesamiento de consultas de los usuarios.

7.7.5. Seguridad

Como se ha mencionado anteriormente, el atributo de calidad de la seguridad de la plataforma y de la información contenida es una cuestión que se dejó de lado por obvias razones: el desarrollo de un MVP en el marco de un trabajo final de carrera con una duración corta generó que se deban establecer ciertas prioridades dentro de las cuales este tema no estaba definido. A pesar de esto, y pensando en la posibilidad de tener clientes “reales”, la seguridad de los datos es de suma importancia en Chat Your Data, especialmente considerando que se trata de información confidencial de empresas u organizaciones. En este sentido, se planea una mejora continua en las medidas de seguridad implementadas en la plataforma. En el futuro, se prevé fortalecer aún más la seguridad de las bases de datos a implementar, así como desarrollar medidas adicionales de cifrado y autenticación de usuarios en el acceso a la plataforma. También se prevé la incorporación de sistemas de monitoreo y detección de amenazas avanzados.

Además, se explorará la posibilidad de obtener certificaciones de seguridad reconocidas a nivel industrial, lo que demostrará el compromiso de Chat Your Data con la protección de los datos

de sus usuarios. Estas certificaciones serán un valioso respaldo para las organizaciones que confían en la plataforma para gestionar su información.

7.7.6. Extensión de funcionalidades del sistema

Con el objetivo de brindarle a las organizaciones la mayor cantidad de soluciones posible, está puesto el foco en extender las capacidades del sistema. La arquitectura construida permite -como se explicó en detalle en las secciones anteriores- una alta extensibilidad del sistema, basada en la implementación de nuevas *tools* que generen nuevas funcionalidades.

En particular, se tiene como prioridad implementar herramientas que permitan consultar bases de datos de tipo SQL, leer de forma correcta archivos CSV (u otros) y resumir documentos extensos. Si bien estos son los prioritarios, la lista es larga y todos cumplen con un objetivo distinto y favorable para los clientes. Otros ejemplos pueden ser: acceso a la información de otras aplicaciones empresariales como CRMs, ERPs, Email, Google Drive, Slack, Microsoft Teams, redes sociales como Instagram, Twitter, Linkedin, etc.

7.7.7. Facilitar la actualización de documentos

La presencia de documentos desactualizados en las organizaciones plantea un desafío notorio en cuanto a la gestión y fiabilidad de la información. Es por ello, que se tiene que diseñar e implementar un sistema que permita a los usuarios fácilmente actualizar sus documentos.

7.7.8. Manejar la posible contradicción entre documentos

Se entiende que pueden existir contradicciones en la información contenida en los documentos que una organización tenga, debido a actualizaciones, obsolescencia u otras razones. Esto puede ocurrir a medida que la organización va usando la plataforma y va subiendo nuevo contenido mientras ejecuta sus operaciones. Por esto, se mejorará el algoritmo para que pueda distinguir la cuestión mencionada anteriormente y devuelva la "última" información sobre el tema.

7.8. Conclusión del Capítulo

El proceso de desarrollo de Chat Your Data estuvo repleto de desafíos y aprendizajes, y le brindó al equipo valiosas lecciones en el campo de la arquitectura de *software* y la implementación de tecnología de vanguardia. A medida que se concluye este capítulo, surgen varias reflexiones sobre las experiencias y conocimientos adquiridos a lo largo del camino.

Uno de los logros más significativos ha sido la comprensión profunda de la importancia de una arquitectura de microservicios. El equipo experimentó en sus ventajas, como la escalabilidad, la flexibilidad y la capacidad de despliegue independiente de los servicios. Al mismo tiempo, surgieron algunos desafíos relacionados con la gestión y la coordinación de estos servicios, lo que ha enseñado sobre la importancia de un diseño cuidadoso y una comunicación efectiva entre los equipos.

La implementación de Chat Your Data en la nube de AWS ha proporcionado una valiosa experiencia en el despliegue y la administración de servicios en un entorno de producción. El equipo aprendió a aprovechar las capacidades de escalabilidad y seguridad que ofrece la nube, lo que ha sido fundamental para garantizar un rendimiento óptimo y la protección de los datos de los usuarios.

La integración de gRPC y API en el sistema ha permitido lograr una comunicación eficiente entre los componentes. Se distinguieron las fortalezas de cada enfoque y cómo pueden ser aplicados en diferentes contextos. Esta versatilidad ha demostrado la importancia de seleccionar la tecnología adecuada para cada tarea y cómo estas tecnologías pueden complementarse entre sí para brindar un sistema coherente y eficaz.

Uno de los aspectos más emocionantes y desafiantes de este proyecto fue la incursión en el campo de la inteligencia artificial. El equipo trabajó con modelos de lenguaje de última generación (LLMs), bases de datos vectoriales y técnicas de procesamiento de lenguaje natural (NLP). Esta experiencia brindó una visión más profunda de las capacidades y limitaciones de la inteligencia artificial y preparó al equipo para el futuro de Chat Your Data, así como futuros proyectos relacionados con esta tecnología en constante evolución.

Actualmente se está hablando mucho acerca de la importancia mirando hacia el futuro del Prompt Engineering como una habilidad para la vida personal y profesional de todas las personas. Comprender cómo diseñar y formular *prompts* para interactuar con los modelos de lenguaje proporcionó una habilidad valiosa que se considera esencial en el panorama tecnológico actual y futuro.

Además, se reconoció el potencial transformador de las herramientas de inteligencia artificial en el mundo organizacional. La capacidad de automatizar tareas, generar contenido y ofrecer respuestas rápidas a través del *chat* tiene un impacto significativo en la eficiencia operativa y la toma de decisiones dentro de las organizaciones. Chat Your Data es un testimonio del

compromiso del equipo con la excelencia técnica y la creación de soluciones que empoderen a las organizaciones a través de la inteligencia artificial.

Cada desafío superado y cada lección aprendida han contribuido al crecimiento y desarrollo del equipo. Mirando hacia el futuro del emprendimiento y de otros proyectos, el equipo está preparado para enfrentar el constante cambio en el área de inteligencia artificial y su potencial, así como los desafíos y oportunidades que presentará.

8. Gestión del proyecto

8.1. Marco de gestión

Como se mencionó previamente, el equipo dividió el trabajo en tres etapas principales. Inicialmente, se llevó a cabo una fase de investigación que se inició en febrero de 2023. Durante esta etapa, se investigó y validó el problema, al mismo tiempo que se determinó la solución adecuada. Además, se llevó a cabo una investigación en el campo de la inteligencia artificial. Esta fase abarcó desde finales de febrero hasta finales de mayo, con una duración de tres meses, donde se adaptó el marco de Kanban.

Luego, se siguió a la etapa de desarrollo del producto, que no solo involucró la codificación, sino que también se continuó validando la solución y realizando pruebas con diversos actores. Se tuvo que seguir investigando sobre tecnologías durante el camino, ya que estas eran completamente nuevas para el equipo. Por lo tanto, por más que se le llame etapa de desarrollo, también hubo tiempo dedicado a la investigación, y validación. Esta etapa comenzó el 22 de mayo y finalizó el 25 de septiembre. En esta etapa se adaptó el marco de Scrum, abarcando nueve iteraciones (*sprints*) de dos semanas cada una, lo que equivale a cuatro meses.

Finalmente, se llevó a cabo la tercera etapa, con una duración de un mes aproximadamente, durante la cual se realizaron mejoras, pruebas y se completó la redacción de la documentación.

8.2. Herramientas para la gestión

8.2.1. Toggl

Para medir y registrar el esfuerzo dedicado por el equipo a lo largo de todo el proyecto, se empleó la herramienta Toggl. Esta aplicación fue utilizada en todas las etapas del proyecto y permitió detallar el tiempo dedicado a tareas específicas. Para una mayor precisión, se asignaron etiquetas a cada tarea, como 'Capacitación', 'Investigación', 'Reuniones' (distinguiendo entre reuniones internas, con el *beta tester* y con el tutor), entre otras. Además de las etiquetas, se podía agregar una descripción detallada de la tarea en curso. Cada miembro del equipo iniciaba el contador de tiempo al comenzar una tarea y lo detenía al completarla.

El uso de esta herramienta proporcionó información importante sobre la distribución del tiempo, lo que permitió identificar áreas que requerían mayor atención y evaluar la eficiencia

del equipo. Además, se convirtió en un recurso esencial para las métricas de seguimiento del equipo y facilitó la identificación de oportunidades de mejora en la gestión del tiempo.

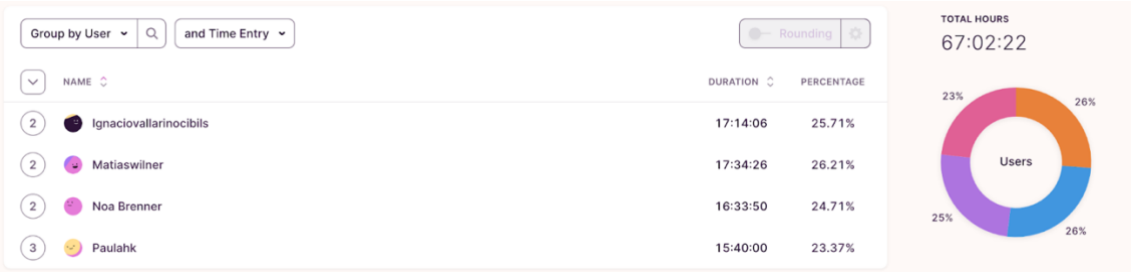


Ilustración 53- Evidencia de uso de Toggl

8.2.2. Notion

Durante la etapa de investigación, el equipo optó por utilizar la plataforma Notion como su herramienta principal. Notion ofrece la capacidad de mantener toda la información ordenada y accesible de manera eficiente. Se implementaron categorías temáticas en Notion, como investigación, notas de reuniones, cursos, entre otros, lo cual permitió al equipo organizar y estructurar la información de manera sencilla. Además de su funcionalidad de organización, Notion proporcionó el tablero de Kanban integrado en la plataforma, que se abordará a continuación.

8.2.3. Jira

Como se mencionó anteriormente, para la etapa de desarrollo, se utilizó Jira. Esta herramienta permite trabajar con Scrum, siendo una herramienta fácil de usar e intuitiva. Además, permitió planificar y gestionar los *sprints* de trabajo, gestionando las incidencias. También brinda una variedad de reportes que fueron de gran utilidad para poder analizar la evolución y lograr planificar mejor.

8.3. Etapa de investigación

Como se mencionó anteriormente, durante la fase de investigación, se hizo uso de la plataforma Notion, y a continuación, se detalla su estructura organizativa.

En primer lugar, se estableció un calendario que sirvió para programar y registrar las próximas reuniones y entrevistas. Además, se dispuso de un espacio dedicado exclusivamente a las notas de las reuniones, donde se documentaron los detalles de las discusiones y las decisiones clave

tomadas en cada encuentro. Otra sección importante de Notion se destinó a la investigación en profundidad sobre temas cruciales como la inteligencia artificial, LangChain, *prompts* y cualquier otra información relevante. En este espacio, se registraron apuntes de cursos, lecturas, videos, junto con las fuentes de donde se extrajo esta información y las lecciones aprendidas en el proceso. En la sección 13.27 Organización del Notion se puede encontrar una imagen de cómo se organizó.

Notion también desempeñó un papel fundamental en la recopilación y organización de datos relacionados con el problema en estudio, así como las decisiones críticas y posibles soluciones. Toda esta información fue un punto de partida para abordar temas técnicos, como la arquitectura, la ingeniería de *software* y la ingeniería de requerimientos. Cabe destacar que varios de estos apuntes resultaron fundamentales en el proceso de documentación del proyecto.

8.3.1. Esfuerzo

Se utilizó este marco desde el 10 de febrero hasta el 22 de mayo, siendo estos 3 meses y 12 días, dedicando 1033 horas. En el anexo 13.28 Cursos se encuentran algunos detalles de las temáticas, como por ejemplo qué cursos se vieron, páginas interesantes, etc.

Tema	Dedicación (horas)	Descripción
Machine Learning y NLP	200	Investigación en profundidad sobre los conceptos y aplicaciones de *Machine Learning* y Procesamiento de Lenguaje Natural (*NLP*)
Inteligencia Artificial	75	Estudio y análisis detallado de los fundamentos y enfoques de la Inteligencia Artificial, incluyendo sus implicaciones y posibles aplicaciones
Herramientas y tecnologías	150	Estudio y análisis de distintas tecnologías y herramientas
Herramienta: LangChain	100	Investigación exhaustiva de LangChain, explorando su funcionalidad, usos potenciales y limitaciones

Tecnologías: Python	23	Estudio de la programación en Python, incluyendo bibliotecas y *frameworks* relevantes para el desarrollo de soluciones basadas en inteligencia artificial
Problema	198	Análisis profundo del problema, análisis del negocio
Solución	190	Identificación de posibles soluciones y enfoques para abordarlo de manera efectiva
Entrevistas y reuniones	97	Realización de entrevistas y reuniones con los *stakeholders* clave del proyecto para comprender sus necesidades, desafíos y expectativas

Tabla 6- Tabla de esfuerzos en la etapa de investigación

8.4. Etapa de desarrollo

8.4.1. Scrum

Como se mencionó anteriormente, para la etapa de desarrollo de *software*, el equipo decidió utilizar Scrum, realizando una adaptación de este *framework*.

La elección de este marco fue justificada en los capítulos anteriores. A continuación, se describirá en profundidad sobre cómo se implementó este marco, el equipo scrum, sus artefactos y eventos. Para esto se siguió el ciclo de Scrum, explicado en el marco. En la sección 13.29.1 Ciclo de Scrum, se encuentra una imagen donde se muestra el ciclo sugerido por Scrum.

8.4.1.1. Equipo Scrum

Scrum Master

Es el responsable del proceso de Scrum, su función consiste en gestionar y facilitar el proceso de Scrum, asegurando que se sigan los principios y prácticas para alcanzar los objetivos del proyecto. Por ejemplo, apoyar a los *developers* para que sean autoorganizados, o facilitar los eventos de Scrum según se necesite. Esta persona trabaja con el equipo y con los *stakeholders*, los interesados del proyecto. Noa Brenner fue designada como Scrum Master del proyecto, ya que era la encargada de gestión.

Developers

Son los responsables de construir y entregar el incremento, como también de generar y mantener la definición de terminado (Definition of Done). Los *developers* son quienes definen quienes ejecutan qué tareas, y estiman las tareas a realizar. Rol ocupado por todos los integrantes del equipo.

Product Owner

Paula se desempeñó como el Product Owner, quien es responsable de maximizar el valor del producto resultante del trabajo. Es responsable del incremento y de la gestión efectiva del Product Backlog, como también de entender y transmitir las necesidades de los usuarios a los *developers*. También actúa como representante de los posibles interesados en el proyecto, garantizando que se cumplan sus necesidades y expectativas.

8.4.1.2. Artefactos

Los artefactos de Scrum representan el trabajo o el valor. Su función es maximizar la transparencia, siendo este uno de los pilares de Scrum. Cada artefacto contiene un compromiso para garantizar que proporcione información que mejore la transparencia.

- Para el ***Product Backlog***, es el **Objetivo del Producto** (*Product Goal*).
- Para el ***Sprint Backlog***, es el **Objetivo del *Sprint*** (*Sprint Goal*).
- Para el ***Product Increment*** es la **Definición de Terminado** (*Definition of Done*)

En el anexo 13.29.2 Artefactos de Scrum se explica detallado estos tres artefactos y cuando fueron definidos en el proyecto.

A continuación, se explica cuál fue la Definition of Done del equipo, siendo este el compromiso del Product Backlog.

Definition of Done

El Definition of Done (Definición de Terminado) se estableció como un conjunto de criterios que debían cumplirse para considerar que una historia de usuario estaba lista para ser marcada como "terminada". En resumen, una *user story* podía considerarse completa cuando:

- Era integrada en la rama *develop* mediante un *pull request* que había sido aprobado por otro miembro del equipo.
- Cumplía con los criterios de aceptación específicos definidos para esa historia en particular.
- Se desplegaba de manera correcta en el ambiente de pruebas.

8.4.1.3. Eventos de Scrum

Sprint Planning

Durante este evento, se establece el objetivo del próximo sprint (Sprint Goal) para la siguiente iteración. Dicho objetivo debe ser único y se deriva del Product Backlog. Los participantes en esta reunión abarcan a todos los miembros del equipo Scrum (Product Owner*, developers y* Scrum Master). En este encuentro, se identifican las tareas que serán incorporadas al Sprint Backlog.

Siguiendo la dinámica de *sprints* de dos semanas, con cada ciclo comenzando cada dos lunes, el proceso del Sprint Planning se llevaba a cabo antes de iniciar el siguiente *sprint*.

Durante esta reunión, en primer lugar, se seleccionaban las historias de usuario que serían abordadas en el siguiente ciclo, extrayéndolas del Product Backlog. Después se realizaba una re-estimación de *story points*, en donde se utilizaba la herramienta Planning Poker, donde todos los integrantes del equipo podían opinar y con esto llegar a una estimación en conjunto. Además, se tenía en cuenta cuánto trabajo podía dedicar el equipo para esa *sprint*. Dicha elección se basaba en la velocidad del equipo y el nivel de compromiso en ese momento. La disponibilidad del equipo podía variar debido a factores laborales, académicos o personales, y estas circunstancias se tomaban en cuenta al determinar las historias de usuario a incluir.

Sprint Daily

Esta instancia representa un punto de comunicación fundamental para evaluar el progreso hacia el Sprint Goal y ajustar, si es necesario, el Sprint Backlog. Durante este evento, el equipo tiene la oportunidad de presentar su estado de avances y abordar cualquier asunto relevante.

Dado que todos los miembros del equipo trabajaban en conjunto y estaban en constante comunicación, se optó por una adaptación en este evento, llevando a cabo tres reuniones semanales (lunes, miércoles y viernes). En cada una de estas sesiones, al comienzo, se destinaban 15 minutos para permitir que los integrantes dialogaran sobre posibles

impedimentos o bloqueos que pudieran estar enfrentando, buscando soluciones colaborativas. Posteriormente, una vez transcurridos los 15 minutos, la reunión continuaba abordando otros temas de importancia que demandaban deliberación y toma de decisiones por parte del equipo.

Sprint Review

El evento de revisión del *sprint* constituye una instancia de evaluación del incremento logrado y de adaptación del *backlog* mediante la muestra de los logros obtenidos durante el período del *sprint*. Estas reuniones tienen como objetivo abordar cuestiones como la finalización de tareas, los desafíos enfrentados y el estado actual del backlog. En esta dinámica, el Product Owner desempeña un papel fundamental, brindando retroalimentación y evaluando si el incremento se cumplió adecuadamente.

Inicialmente, estas sesiones no contaron con la participación de un Product Owner externo; en su lugar, se le presentaban los avances al tutor, quien junto al equipo evaluaba los resultados del *sprint*. Luego una vez que el equipo comenzó a tener reuniones fijas cada dos semanas con el equipo del BID, se comenzó a hacer con ellos.

Sprint Retrospective

En la instancia de inspección y adaptación, se aborda la evaluación del desempeño del equipo a través de la transparencia de los acontecimientos ocurridos. En este evento, se siguen los pasos delineados en el libro "Agile Retrospectives - Making Good Teams Great" de Esther Derby, para lograr una estructura efectiva: se inicia con la apertura del escenario, seguido de la recolección de datos, la generación de percepciones, la toma de decisiones y finaliza con un cierre. Esta metodología fomenta la divergencia inicial para luego converger y establecer un plan de acción. El propósito principal es generar un plan de acción único, o en su defecto, un máximo de tres, junto con la asignación de responsables encargados de su implementación. La finalidad de este evento es mejorar el proceso de futuras iteraciones y evaluar el desempeño del trabajo realizado. La participación en esta reunión es extendida a todos los miembros del equipo de Scrum y se lleva a cabo al culminar un sprint. Se muestra evidencia en el anexo 13.30 Evidencia Retro .

8.4.1.4. Estimación

El equipo llevó a cabo la estimación de las tareas después de definir y validar el *backlog*. Además, se realizaron re-estimaciones durante las Sprint Plannings para garantizar una

planificación precisa. La estimación se basó en la escala de Fibonacci, donde el número más bajo era 1 y el más alto era 8. Si una tarea excedía este número, se desglosaba en tareas más pequeñas para facilitar la valoración en términos de esfuerzo y tiempo.

Para llevar a cabo la estimación, el equipo utilizó la técnica conocida como Planning Poker. Este enfoque involucra a todo el equipo en un proceso iterativo para desarrollar estimaciones precisas. Se selecciona una carta al azar que representa una historia de usuario y se discute en detalle. Cada desarrollador escribe su estimación de manera independiente y sin mostrarla a los demás. En estas estimaciones, los desarrolladores consideran todas las acciones necesarias para completar la historia. Después de que todos completan sus estimaciones, se revelan y se discuten. Luego, se repite el proceso de escritura y discusión de las estimaciones, lo que a menudo lleva a una convergencia en las estimaciones [60].

Es importante destacar que no era necesario que todas las estimaciones sean idénticas. Lo esencial era que el equipo argumente y comprenda las decisiones de manera colaborativa, lo que garantiza que las estimaciones reflejen de manera precisa el esfuerzo requerido para completar cada tarea.

8.4.1.5. Uso de *story points*

Al final de una iteración, el equipo contaba el número de puntos de historia que completaron. Luego usa este número como un pronóstico de cuántos puntos de la historia completarán en las próximas iteraciones de la misma longitud.

8.4.2. Descripción de los *sprints* realizados

Para cada *sprint* se fijaba un objetivo, y un entregable, además se cuenta con la fecha y el esfuerzo realizado. A continuación, se muestra algún caso de ejemplo:

***Sprint* 7**

Sprint	7
Fecha	14/8/2023 - 28/8/2023
Objetivo	Creación del contexto conversacional y optimizar las respuestas generadas.

Esfuerzo	43 sp
Entregable	Se generó el segundo *release* cumpliendo el objetivo, y se preparó al sistema para la función de guardar conversaciones.

Tabla 7- Descripción del *sprint* 7

Evidencia de *retro meeting* del *sprint* 7

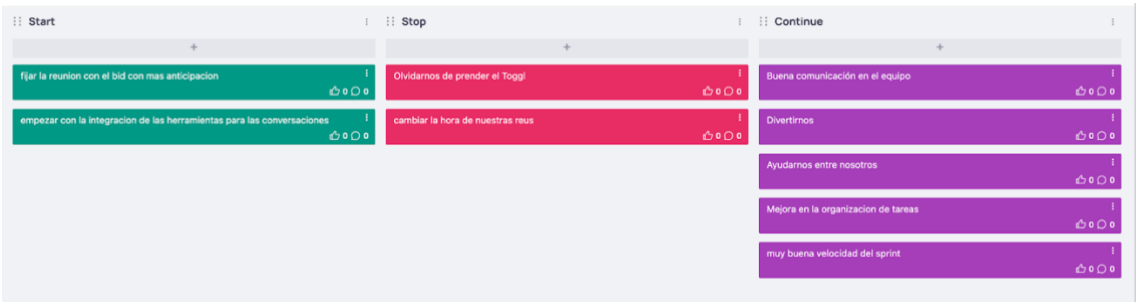


Ilustración 54- Evidencia de *retro meeting* - *sprint* 7

***Sprint* 8**

Sprint	8
Fecha	28/8/2023 - 11/9/2023
Objetivo	Integración de Redis y Firebase Store para el manejo de las conversaciones.
Esfuerzo	35 sp
Entregable	Implementación de la integración de Redis y Firebase Store para gestionar y almacenar las conversaciones, brindando al usuario la capacidad de acceder a conversaciones anteriores y continuarlas desde donde las dejó.

Tabla 8- Descripción de sprint 8

8.4.3. Métricas de gestión

8.4.3.1. Desvío de estimaciones

El desvío de estimaciones se refiere a la comparación entre los *story points* que el equipo planeó completar en un *sprint* y los que realmente logró completar. Si existe un desvío significativo entre lo planeado y lo ejecutado, esto puede indicar que las estimaciones iniciales necesitan ser ajustadas o que hay factores imprevistos que afectan la capacidad de ejecución del equipo. Analizar este desvío de manera regular hizo que el equipo pueda mejorar su capacidad de planificación y gestión de *sprints*.

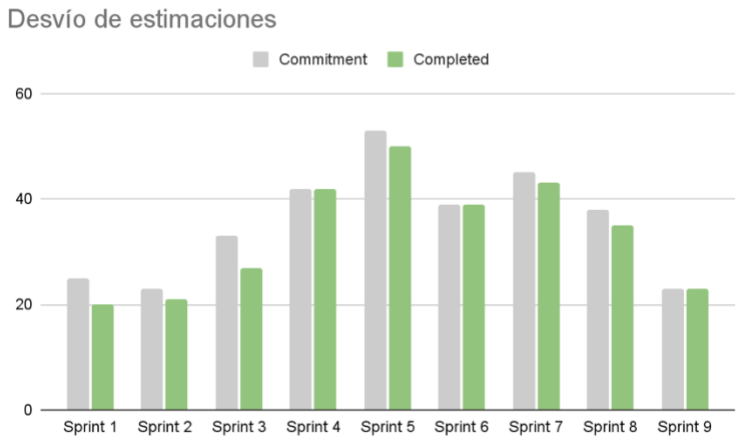


Ilustración 55- Desvío de estimaciones

La barra gris representa los *story points* que el equipo se comprometió a completar al comienzo del *sprint*, mientras que la barra verde muestra los *story points* efectivamente implementados. Durante los primeros *sprints*, los *story points* asignados eran menores debido a la fase inicial de configuración del proyecto, lo que llevó a estimaciones menos precisas. Con el tiempo, el equipo adquirió más experiencia tanto con las tecnologías utilizadas como con el producto en desarrollo, lo que permitió estimaciones y planificaciones más precisas.

En el *sprint* 3, se observa una mayor desviación, principalmente debido a desafíos relacionados con la tecnología Python y la configuración de LangChain, una herramienta nueva que aún no se dominaba completamente. Sin embargo, en el *sprint* 4, el equipo logró equilibrar y completó la integración de manera exitosa.

En el *sprint* 8, se registra otra ligera desviación. Esto se debió a la necesidad de realizar correcciones en el producto final y la implementación de mejoras que no estaban completamente definidas al inicio del *sprint*.

8.4.3.2. Velocidad del equipo

La velocidad del equipo se define como la cantidad de puntos terminados por *sprint* [61]. Al comenzar cada nuevo *sprint*, el equipo utilizaba esta métrica como referencia para planificar el siguiente. Mantenerse en línea con la velocidad esperada permitía al equipo cumplir con el alcance en la fecha establecida. Se esperaba que la velocidad del equipo aumentara a medida que avanzaba el proyecto, ya que se adquiría más experiencia y las partes más difíciles se abordaban al principio del desarrollo.

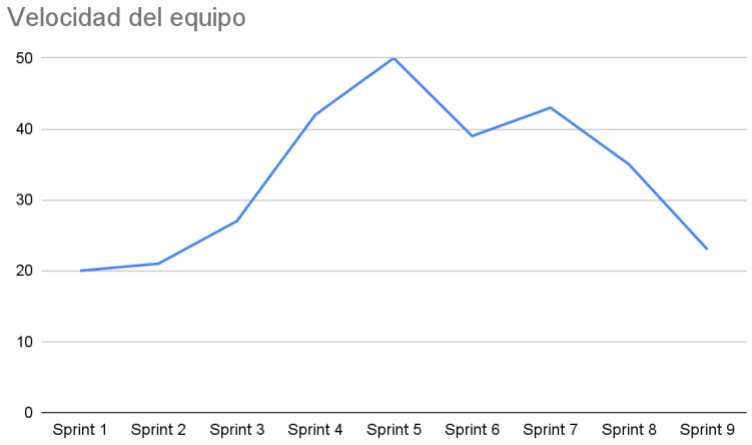


Ilustración 56- Gráfico de velocidad del equipo

El análisis de la velocidad del equipo muestra una evolución interesante a lo largo de los *sprints*. Inicialmente, en los primeros *sprints*, la velocidad no fue alta, lo que se puede atribuir al tiempo dedicado a la configuración de tecnologías y herramientas. Este proceso de configuración también se refleja en la desviación observada en el *sprint* 3.

A partir del *sprint* 4, se observa un aumento significativo en la velocidad, que coincidió con una fecha crítica de entrega planificada para el 31 de julio. En los *sprints* 4 y 5, el equipo trabajó intensamente en la integración del *chatbot* de IA con documentos y otras funcionalidades clave. Sin embargo, en los *sprints* siguientes, como el 6, hubo una pequeña disminución de velocidad, ya que se centraron en la mejora del contexto conversacional del *chat* y en la respuesta del mismo. Esto implicaba la integración de más *bots* y se anticipaba como una tarea más compleja.

En el *sprint* 7, el equipo logró una mejor comprensión de cómo abordar esta tarea, lo que resultó en una mejora en la velocidad. Sin embargo, en el *sprint* 8, la velocidad disminuyó nuevamente, ya que se trabajaba en el guardado de conversaciones y en mejoras generales del sistema.

Finalmente, en la iteración 9, se observa una velocidad menor, ya que el equipo estaba cumpliendo con el alcance del proyecto, es decir no quedaban tantas historias de usuario en el *product backlog* y se estaba enfocando en actividades de documentación. El promedio de la velocidad del equipo a lo largo del desarrollo fue de 33,3 puntos de historia por *sprint*.

8.4.3.3. Burndown chart

Burndown chart se convirtió en una herramienta fundamental para el equipo, ya que permite medir la cantidad de requerimientos del *backlog* que aún quedan por completar al comienzo de cada *sprint*. Esta visualización se volvió esencial en la planificación de los *sprints*, ya que proporcionaba una vista clara del progreso del equipo y permitía evaluar si se iba a alcanzar el alcance previsto en el tiempo establecido, en caso contrario el equipo debía tomar acciones para corregir esas desviaciones.

Además, Jira permite a los equipos ver de manera visual y actualizada cuánto trabajo queda por hacer a medida que avanzan en el *sprint*. Esto se logra a través de un gráfico de progreso que muestra de manera clara el porcentaje de trabajo en proceso, el trabajo completado y el trabajo que aún no ha comenzado, proporcionando una representación visual instantánea del progreso del *sprint*.

Ilustración 57- Gráfico de progreso del *sprint*

8.4.3.4. Retrabajo

Para medir el retrabajo, el equipo utilizaba Jira para identificar las tareas que se consideraban errores o problemas. Esto se señalaba en Jira utilizando la etiqueta *bug*. Cuando surgían problemas relacionados con herramientas, integraciones o errores en el sistema, se registraba una tarea específica para abordarlos.

Con esta información disponible, se pudo generar la siguiente gráfica:

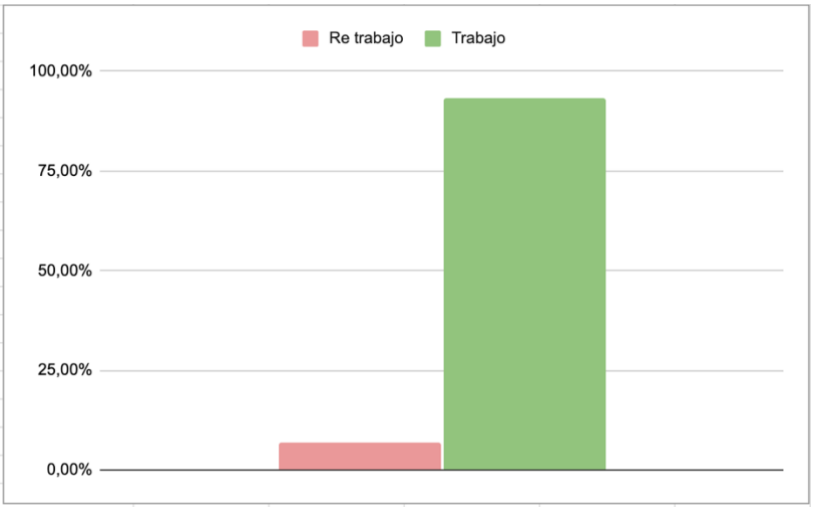


Ilustración 58- Gráfico de retrabajo

Se puede observar que se experimentó un bajo índice de retrabajo, lo que indica una efectiva planificación, organización y la capacidad de mitigar errores a través de prácticas como la revisión de código y la adhesión a buenas prácticas, entre otras.

8.4.4. Plan de *releases*

Para organizar el trabajo de mejor manera se planificaron distintos *releases*, lo que permitió al equipo definir entregas intermedias, marcar objetivos medibles y ayudar a realizar un producto incremental.

Se planificaron cuatro *releases* con fechas establecidas y objetivos principales, así como funcionalidades claves. El primero se llevó a cabo al finalizar el quinto *sprint*, el 31 de julio, en el que el objetivo era tener una primera versión con las funcionalidades elementales, y para esto se implementaron las funcionalidades básicas, como el *chatbot* con IA que pueda responder sobre un archivo y la capacidad de cargar documentos.

El segundo *release* tuvo lugar al finalizar el séptimo *sprint*, el 28 de agosto. Durante estas dos iteraciones, el equipo se enfocó en mejorar las respuestas del *chat*, la capacidad de tener

contexto conversacional, el manejo de archivos y autenticación, siendo el objetivo obtener una versión mejorada incorporando funcionalidades complementarias.

Luego, se decidió seguir realizando *releases* al finalizar cada iteración, lo que significa que se hicieron al finalizar el octavo y noveno *sprint*. Estos últimos *releases*, que culminaron el 25 de septiembre, incorporaron las últimas características, como el guardado de conversaciones para cada usuario, y la creación de nuevos *chats*, reiniciando el contexto conversacional y mejoras del contexto conversacional utilizando entidades, brindando la versión más completa de Chat Your Data.

De todas maneras, al equipo se le informó de parte de ORTsf que se podía continuar con el desarrollo del código hasta la fecha de la defensa, por ende, el equipo decidió realizar otro *release* más con unos últimos detalles.

8.5. Gestión de riesgos

La gestión de riesgo implica anticipar riesgos que pueden alterar el calendario del proyecto o la calidad del *software* a entregar y posteriormente tomar acciones para evitar los riesgos. Se puede considerar un riesgo, algo que se prefiere que no ocurra, los mismos pueden ser amenazas al proyecto, al equipo o al *software*. La gestión de riesgos es importante principalmente para este tipo de proyectos, que cuentan con un nivel grande de incertidumbre, que los requerimientos pueden cambiar, o las tecnologías son nuevas. Debido a esto es necesario anticipar los riesgos, comprender el efecto de los mismos, y saber cómo evitarlos. De esta manera, el equipo decidió diseñar planes de contingencia, para poder tomar acciones inmediatas en caso de que ocurran, y planes de mitigación para evitar que los mismos ocurran.

El equipo decidió dividir los riesgos en las siguientes categorías: Interesados, Producto, Equipo y Tecnológicos. Para el proceso de gestión de riesgos, el equipo se basó en las siguientes etapas: identificación del riesgo, análisis de riesgos, planeación del riesgo y monitorización del riesgo.

En la etapa de identificación del riesgo, no solo se identificaron los riesgos, sino también se realizó un juicio acerca de la probabilidad y gravedad de dicho riesgo (análisis de riesgos). El equipo decidió tomar una estrategia para esta etapa. Los riesgos se ponderaban de acuerdo a la probabilidad de ocurrencia, asignándoles un valor entre 0 y 1 — siendo 1 la mayor probabilidad de ocurrencia, mientras que el impacto sobre el proyecto se ponderaba con valor entre 0 al 5 — siendo 5 el de mayor impacto. Teniendo ambos valores, se multiplicaban los dos valores

llegando a la magnitud del riesgo. Dicho resultado permitió realizar la matriz de probabilidad e impacto que clasifica los riesgos según la importancia: "alta", "moderada" o "baja". Con esta matriz, el equipo concluyó que si el resultado estaba entre 0 y 1 se debe aceptar el riesgo y no se deben tomar medidas, si esta entre 1 y 2 se debe tomar medidas para mitigarlo, si es mayor a 2 el riesgo debe ser evitado. En esta parte, entra la siguiente etapa del proceso, la planeación del riesgo, donde se consideraba que acciones se puede tomar para minimizar/evitar el riesgo, lo que antes se mencionó: planes de mitigación y contingencia. La monitorización del riesgo es el proceso para comprobar que no han cambiado los riesgos, es por esto que se debe valorar y revisar regularmente, para saber si ese riesgo es más o menos probable. El equipo decidió hacer una reunión cada dos semanas para poder evaluar los riesgos, sus probabilidades y decidir si hay nuevos riesgos. En las primeras etapas del proyecto, se decidió hacer la reunión una vez por mes, ya que los riesgos no cambiaban demasiado sus probabilidades, debido a que el equipo se encontraba en la parte de investigación. Una vez comenzado el desarrollo, se decidió volver a hacer cada dos semanas.

Para registrar toda esta información, se creó una planilla que contenía todos los campos mencionados anteriormente. Se encuentra una evidencia en 13.34 Evidencia de Excel de riesgos. Se generaba una nueva hoja para cada instancia en la que se necesitaba documentar los datos correspondientes. En el anexo 13.31 Impacto y 13.32 Probabilidad de ocurrencia, se encuentran las tablas de impacto, y de probabilidad de ocurrencia, con sus detalles y valores.

8.5.1. Matriz de probabilidad de impacto

					Probabilidad						
Magnitud											
		0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
	1	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
	2	0	0,2	0,4	0,6	0,8	1	1,2	1,4	1,6	1,8	2
Impacto	3	0	0,3	0,6	0,9	1,2	1,5	1,8	2,1	2,4	2,7	3
	4	0	0,4	0,8	1,2	1,6	2	2,4	2,8	3,2	3,6	4
	5	0	0,5	1	1,5	2	2,5	3	3,5	4	4,5	5

Ilustración 59- Matriz de probabilidad de impacto

Como se mencionó anteriormente, se tenía un plan de respuesta para cada uno de los riesgos. En el anexo 13.33 Riesgos se pueden encontrar todos los riesgos que el equipo identificó a lo largo del proyecto.

A continuación, se muestra el ejemplo de un riesgo con todos los campos que se analizaban cada vez que se hacía el control. Este es el ejemplo del riesgo "No conocer las tecnologías" para la fecha 11/05.

Fecha: 11/05

Fecha	11/05
Riesgo	No conocer las tecnologías
Tipo de riesgo	Tecnológico
Descripción	El equipo no cuenta con un gran conocimiento claro de inteligencia artificial, y Machine Learning
Plan de mitigación	Realizar una etapa de investigación para aprender del tema.
Plan de contingencia	Definir un miembro del equipo responsable para dedicarse específicamente a investigar y estudiar más sobre las tecnologías que estaban dando problema. Si era necesario acudir a alguien experto.
Impacto	4
Probabilidad de ocurrencia	0,5
Score	2

Tabla 9- Análisis del riesgo - 11/05

Una vez ya identificados, se monitoreaban cada un mes o dos semanas, como se mencionó anteriormente. Este control se realizaba utilizando Google Sheets, donde se tenía una hoja para cada fecha correspondiente. De esta manera, se copiaban todos los riesgos anteriores y se analizaba cada uno de ellos nuevamente y se veía si había que considerar algún riesgo nuevo. Esta manera de realizar el control permitió llevar un registro claro sobre cómo evolucionaban los riesgos, pudiendo hacer gráficos para visualizar de mejor manera.

8.5.2. Gráficos de los riesgos y su evolución

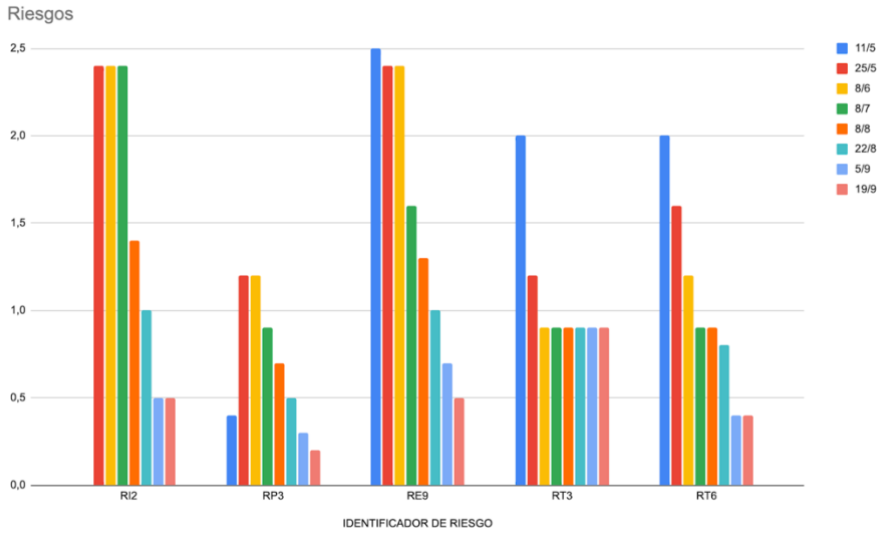


Ilustración 60- Gráfico de los riesgos y su evolución

En el gráfico de riesgos, se puede identificar y analizar algunos de los riesgos claves que fueron los más significativos en el proyecto.

RI2 - Dificultad de comunicación con el potencial *beta tester:* Este riesgo inicialmente planteaba la preocupación de que la comunicación con el potencial *beta tester* no fuera efectiva y que sus expectativas no se comprendieran correctamente. En un principio, el riesgo era más alto debido a la falta de comunicación continua, ya que el equipo solo se comunicaba con Alan Kind mientras se encontraba en la etapa de cerrar el acuerdo.

Para abordar este riesgo, el equipo tomó medidas al programar reuniones regulares cada dos semanas. Esto permitió establecer una comunicación más fluida y una comprensión mutua más clara con el potencial *beta tester*. Esta comunicación continua contribuyó significativamente a reducir el riesgo de malentendidos y a garantizar que las expectativas del potencial *beta tester* se alinearan adecuadamente con el progreso del proyecto. Con el tiempo, este riesgo disminuyó a medida que se fortaleció la relación y la comunicación con el potencial *beta tester*.

RP3 - No conocer al usuario tipo: Este riesgo se relaciona con la falta de comprensión del usuario típico del producto. La gestión de este riesgo fue fundamental, ya que una comprensión insuficiente del usuario podría haber llevado a un producto que no satisficiera sus necesidades.

Para mitigar este riesgo, se incluyeron una serie de acciones específicas. En primer lugar, se llevaron a cabo entrevistas y encuestas con posibles usuarios para comprender sus problemas y necesidades. Además, se realizaron reuniones regulares con usuarios del equipo del BID para discutir el progreso del proyecto y recibir retroalimentación. Otra medida importante fue la realización de validaciones, ya sea con prototipos, o implementar una función en el sistema mismo para que se pueda dar *feedback* a la respuesta. Esto permitió identificar cualquier problema o desviación temprana y realizar ajustes en consecuencia.

RE9 - El equipo no logra organizarse adecuadamente: Este riesgo se refiere a la posibilidad de que el equipo no logre organizarse de manera efectiva para llevar a cabo el proyecto, lo que podría conducir a una mala estimación de las tareas y una planificación deficiente, lo que resultaría en la falta de cumplimiento de los plazos establecidos. Al principio, este riesgo era elevado, dado que el equipo estaba empezando a trabajar en un proyecto nuevo y no estaba completamente familiarizado con las metodologías y la dinámica requerida.

Para mitigar este riesgo, el equipo implementó herramientas de gestión de proyectos y llevó a cabo una planificación detallada de los plazos y las tareas. Se puso un énfasis especial en establecer y comunicar con claridad los objetivos, los entregables y los plazos del proyecto. Se fomentó una comunicación constante entre los miembros del equipo. Con el tiempo, a medida que el equipo se familiarizó con el proyecto y adquirió experiencia en la dinámica de trabajo, este riesgo disminuyó significativamente.

RT3 - Tecnologías investigadas no adecuadas: Existe el riesgo de que las tecnologías investigadas no sean las adecuadas para el proyecto. Este riesgo se manejó a medida que el equipo adquirió experiencia y conocimiento en las tecnologías necesarias. Como plan de mitigación, el equipo hizo pruebas de concepto constantemente para validar que sean adecuadas. En cuanto al plan de contingencia, se buscaron nuevas tecnologías, y si se estaba en una etapa muy avanzada del proyecto, adaptar las conocidas para el sistema.

RT6 - No conocer las tecnologías: Este riesgo se refiere a la falta de conocimiento sobre tecnologías específicas, lo que podría resultar en retrasos en el desarrollo, estimaciones incorrectas de tareas o dificultades para abordar problemas técnicos. El equipo afrontó este riesgo a través de un enfoque constante en el aprendizaje y la capacitación continua en estas tecnologías específicas. La estrategia de mitigación implicaba que el equipo se comprometiera

a adquirir el conocimiento necesario sobre estas tecnologías. Esta dedicación al aprendizaje contribuyó significativamente a la reducción del riesgo a lo largo del tiempo.

Además del plan de mitigación, se desarrolló un plan de contingencia para abordar situaciones en las que el equipo se encontrara con problemas técnicos que no pudiera resolver de inmediato debido a la falta de conocimiento. El plan de contingencia consistía en que, en caso de surgir un problema técnico desconocido, un miembro del equipo se dedicaría a investigar y, si fuera necesario, buscaría asesoramiento de expertos técnicos externos para resolver el problema de manera eficaz.

RT5 - Actualización de las tecnologías que afecten el desempeño del producto: Este riesgo se relaciona con la posibilidad de que las actualizaciones de tecnología, como cambios en sistemas operativos o actualizaciones de *software*, puedan afectar negativamente el rendimiento del producto. Para mitigar este riesgo, el equipo implementó un enfoque de seguimiento de las actualizaciones que eran relevantes para el proyecto. Además, se establece que el equipo antes de implementar cualquier actualización, se harían pruebas en un entorno de desarrollo, para evitar que afecte el rendimiento del producto.

Como plan de contingencia para este riesgo, el equipo decidió que, si se encontraba en una etapa temprana del proyecto, se adaptaría a las tecnologías. Si el proyecto por otro lado se encuentra en una etapa más avanzada, se buscarán soluciones alternativas dentro de esas tecnologías, como realizar ajustes necesarios para que estas no afecten negativamente el producto.

En resumen, a lo largo del proyecto, el equipo demostró una efectiva gestión y mitigación de estos riesgos, lo que desempeñó un papel fundamental en el éxito general del proyecto.

8.6. Gestión de la Comunicación

Desde el inicio del proyecto, se tomó la decisión de establecer un enfoque claro y eficiente para la comunicación, tanto entre los miembros del equipo y con el tutor, como con los interesados externos. A continuación, se explica la gestión de comunicación con los interesados. En el anexo 13.35 Gestión de comunicación se detalla la misma con el equipo y con el tutor.

8.6.1. Comunicación con los interesados

En esta etapa se asignaron responsables para la comunicación con cada interesado. En el caso de GeneXus, se estableció la comunicación entre Paula Hernandez y Mayda Kurdian, Research and Development Manager de K2B, por medio de Whatsapp, fijando después reuniones por Google Meet. En el caso de CloudGaia, Noa estableció el contacto con Ernesto Krawchik, mediante Whatsapp y *mail*, además de asignando reuniones por Meet, lo mismo con el Palacio Legislativo, siendo Ignacio el responsable.

Con respecto al BID (*beta tester*), en las etapas iniciales de la interacción con el Banco Interamericano de Desarrollo la comunicación se estableció principalmente entre Noa y Alan Kind. A medida que avanzaba el proyecto, se buscó ampliar y fortalecer esta comunicación para garantizar una interacción efectiva con los interesados clave. Este enfoque aseguró que las partes involucradas estuvieran al tanto de los avances y resultados del proyecto en todo momento. Para esto fijaron reuniones con los integrantes del equipo y con representantes del BID. Por lo general, lo que se acostumbraba a hacer era que cuando se terminaba la reunión, se pensaba cuando era necesario tener otra (en una semana, dos semanas, etc.), estableciendo la fecha para la siguiente reunión en ese mismo momento. Las reuniones se realizaban con integrantes del equipo del BID, asignados para ayudar y trabajar en conjunto. Este equipo estaba conformado por Mariana Gutierrez, Lucia Latorre, Alan Kind, Ana Laura Sposito y Kyle Strand, los cuales fueron de gran ayuda para todo el proceso. Además, cualquier cosa que se necesitaba, se tenía una cadena de *mails* donde el equipo se comunicaba por cualquier cosa necesaria. En el anexo 13.35.3 Evidencia de reunión con el BID se encuentra una imagen de una de las reuniones.

Complementariamente, para la validación con el BID, se realizó un documento de *Excel* compartido, en donde se realizó un *template* para que los integrantes del BID puedan ir dejando su *feedback* y comentarios.

9. Gestión de Calidad

9.1. Introducción

Un aspecto fundamental de Chat Your Data es su adaptación al lenguaje natural, y el poder brindar contenido de valor y calidad al usuario. La capacidad de proporcionar respuestas coherentes y contextualmente relevantes, como también una buena usabilidad, son aspectos clave para alcanzar el mejor nivel de calidad de la plataforma.

Sin embargo, en un contexto tan dinámico y cambiante como el de la inteligencia artificial, la medición de la calidad puede ser un desafío. Esto se debe a las siguientes razones:

- **La naturaleza dinámica de la inteligencia artificial** y la evolución constante de modelos, algoritmos y capacidades, muestran que la calidad de las respuestas puede variar con el tiempo, lo que hace difícil la medición precisa en un momento específico. Lo que se considera una respuesta de alta calidad hoy podría no serlo en el futuro si los modelos y algoritmos utilizados evolucionan.
- **La subjetividad de la calidad** es otro aspecto que agrega complejidad. Lo que una persona considera una respuesta adecuada, puede no serlo para otra. Además, en contextos más convencionales de desarrollo tecnológico, determinar si una respuesta es 'correcta' puede ser tan simple como un 'sí' o un 'no'. Sin embargo, en este caso, la calidad de la respuesta se evalúa a través de una amplia gama de opiniones que los usuarios pueden tener sobre la interacción, lo que va desde 'mala' hasta 'excelente'.
- **La variedad de casos de uso** para diferentes tipos de información hace que lo que pueda ser una respuesta de alta calidad en un contexto pueda no serlo en otro. Esto hace que evaluar la calidad requiera diferentes enfoques para cada contexto.
- **La calidad depende de los datos de entrenamiento utilizados.** Si estos datos no son precisos, pueden influir en la calidad de las respuestas generadas por la IA. Por esta razón, la calidad de las interacciones y respuestas está altamente relacionada con la calidad de los datos que aportan los usuarios a la plataforma.

A pesar de estos desafíos en la medición de calidad en sistemas de inteligencia artificial, el equipo reconoció la importancia de identificar atributos de calidad específicos, para garantizar la calidad del sistema. Además, se explicará el proceso y las actividades para llegar a los objetivos de calidad, que incluye el uso de estándares, pruebas y evaluaciones para el

aseguramiento del logro de estos objetivos, y un plan de calidad que se puede encontrar en el anexo 13.37 Plan de calidad. A continuación, se describen brevemente estos atributos.

9.2. Definición de calidad para Chat Your Data

Para Chat Your Data, la calidad se define como un conjunto de criterios que son fundamentales para ofrecer la mejor experiencia a los usuarios. Estos atributos de calidad incluyen:

- **Precisión**: El *software* debe proporcionar respuestas precisas a las consultas de los usuarios. Esto significa que cada respuesta debe ser correcta y relevante para la pregunta formulada. La precisión es esencial para construir la confianza del usuario en la plataforma.
- **Contextualización**: Un aspecto clave de Chat Your Data es ofrecer respuestas contextualizadas que consideren el contexto de la consulta del usuario. Esto implica comprender las conversaciones previas y adaptar las respuestas en consecuencia, lo que mejora significativamente la experiencia del usuario.
- **Usabilidad**: Chat Your Data busca proporcionar una interfaz intuitiva y amigable que permita a los usuarios interactuar de manera efectiva y sin dificultades.
- **Rendimiento**: La plataforma debe ser performante, lo que significa que debe ser capaz de manejar una carga de trabajo razonable sin degradación significativa en la velocidad o capacidad de respuesta. Esto asegura una experiencia fluida para los usuarios, incluso en momentos de alta demanda.
- **Interoperabilidad**: La interoperabilidad se refiere a la capacidad de funcionar de manera efectiva y sin problemas con otras tecnologías y sistemas. Esto garantiza que los usuarios puedan integrar la plataforma en entornos fácilmente.
- **Mantenibilidad**: La plataforma debe ser mantenible y extensible a medida que evoluciona. Esto implica la capacidad de realizar actualizaciones y adoptar nuevas tecnologías de forma eficiente. La mantenibilidad es crucial para garantizar que Chat Your Data viva dentro de un entorno tecnológico que está en constante cambio.
- **Extensibilidad**: La plataforma debe ser extensible para poder adaptarse y evolucionar con el tiempo, incorporando nuevas funcionalidades o modificando las existentes de manera eficiente y sin causar interrupciones en el funcionamiento general.

Al momento de planificar los atributos de calidad que se consideraban importantes para el proyecto, se decidió omitir algunos que, reconociendo su importancia, se postergó la

implementación de soluciones que permitan cumplir con esos atributos. Estos atributos de calidad omitidos son los detallados a continuación:

- **Seguridad:** Este atributo se refiere a la protección de los datos y la prevención de accesos no autorizados.
- **Testeabilidad:** Es la facilidad con la que se puede probar el sistema para garantizar que cumple con los requisitos.
- **Confiabilidad:** Implica que el sistema funcione de manera consistente y predecible.
- **Disponibilidad**: Refiere a la capacidad del sistema para estar en funcionamiento y accesible cuando los usuarios lo necesitan.

9.2.1. Justificación de los atributos de calidad seleccionados

El equipo eligió los atributos de calidad detallados anteriormente como los más importantes, considerando las necesidades de los usuarios y la demanda del mercado. Si bien otros atributos como la seguridad, testeabilidad, confiabilidad y disponibilidad son importantes, se optó por priorizar los primeros atributos de calidad mencionados por las siguientes razones:

- **Relevancia para el usuario**: Los atributos seleccionados impactan directamente en la experiencia de usuario final, y son aspectos cruciales para brindar respuestas útiles y la satisfacción de los usuarios.
- **Factibilidad en el contexto actual**: Dada la complejidad y las restricciones del tiempo del proyecto y que el mismo es un *MVP* (mínimo producto viable), el equipo identificó estos atributos como los más alcanzables en la fase actual. Esto permitirá brindar un mínimo producto viable que cumpla con los aspectos esenciales, antes de considerar atributos adicionales.
- **Flexibilidad en el futuro**: Aunque estos atributos son la principal prioridad, se reconoce la importancia de otros factores, como la seguridad de los datos. Estos aspectos se abordarán en etapas futuras del producto.

Esta selección de atributos permite ofrecer un producto de calidad, que cumpla con los requerimientos esenciales y brinde una base sólida para futuras mejoras y expansiones.

9.3. Objetivos de calidad

En esta sección, se establecen los objetivos de calidad para Chat Your Data. Se utiliza el enfoque SMART (específico, medible, alcanzable, relevante y oportuno) para su definición [62].

- **Precisión**:
  - Objetivo: El *software* debe proporcionar respuestas precisas al 90% de las consultas de los usuarios.
- **Contextualización:**
  - Objetivo: Al menos el 90% de las respuestas deben tener en cuenta el contexto de la consulta del usuario.
- **Usabilidad**:
  - Objetivos:
    - La interfaz de usuario del sistema debe ser intuitiva y fácil de usar, siguiendo los principios de diseño centrado en el usuario.
    - El sistema debe proporcionar retroalimentación adecuada a los usuarios sobre el estado de sus acciones (por ejemplo, mensajes de error o confirmación).
- **Rendimiento:**
  - Objetivos:
    - El sistema debe responder a las consultas de los usuarios en un tiempo promedio no superior a 10 segundos.
    - El sistema debe ser capaz de manejar simultáneamente hasta 1.000 usuarios activos sin degradar significativamente su rendimiento.
- **Extensibilidad**:
  - Objetivos:
    - El sistema debe ser extensible para poder agregar nuevas funcionalidades (*Tools*) fácilmente.
    - El sistema debe ser extensible para poder implementar nuevas tecnologías de inteligencia artificial, como *LLMs*, nuevos *frameworks*, bases de datos vectoriales, etc.

- **Interoperabilidad**:
    - Objetivos:
        - El sistema debe ser compatible con los navegadores web más comunes (Google Chrome, Firefox, Safari, Microsoft Edge) en sus versiones más recientes.
        - El sistema debe ser accesible y funcional en dispositivos móviles, tabletas y computadoras de escritorio, adaptándose a diferentes resoluciones y tamaños de pantalla.
- **Mantenibilidad:**
    - Objetivos:
        - El código fuente del sistema debe estar organizado y documentado, siguiendo las mejores prácticas de la industria para facilitar futuras actualizaciones, correcciones de errores y mejoras en la funcionalidad.
        - El sistema debe permitir actualizaciones sin interrupciones significativas en el servicio, minimizando el tiempo de inactividad y las molestias para los usuarios.

9.4. Aseguramiento de la calidad

El aseguramiento de la calidad es el proceso de garantizar que el *software* cumple con los objetivos de calidad establecidos [63]. En esta sección, se describirá cómo el equipo aplicó estándares y actividades para alcanzar los objetivos de calidad previamente definidos.

9.4.1. Estándares de calidad

9.4.1.1. Estándares de código de fuente - *Backend*

- **Estructura de carpeta modular:** El código se estructura en módulos con controladores, servicios y módulos. Esta estructura facilita la gestión y el mantenimiento, permitiendo expandirse y agregar características fácilmente.
- **Inyección de dependencias**: El código utiliza inyección de dependencias para garantizar que los componentes sean acoplados y reutilizables.
- **Nomenclatura para nombres de clases**: El *software* sigue una nomenclatura coherente para los nombres de las clases, lo que facilita la comprensión del código.

- **Alta cohesión y bajo acoplamiento:** Se promueve la alta cohesión (módulos que realizan una única tarea) y el bajo acoplamiento (dependencias mínimas entre módulos) para un código más limpio y mantenible.

9.4.1.2. Estándares de código de fuente - *Frontend*

- **TypeScript**: En el *frontend* se utiliza TypeScript, un *superset* de JavaScript que permite definir tipos de datos precisos y verificar la integridad del código. Esto hace que el código sea más robusto y menos propenso a errores.
- **ESLint y Prettier:** Para garantizar la coherencia en el código y mantener un estilo de codificación limpio y uniforme, se utilizó ESLint y Prettier. ESLint ayuda a identificar y corregir problemas potenciales en el código, mientras que Prettier formatea el código para que cumpla con las reglas establecidas. Esto asegura que los nombres de variables sean descriptivos, una correcta indentación, etc., haciendo que se cumplan las convenciones de codificación establecidas de forma consistente.
- **React Hooks**: El *frontend* se desarrolló en React y se hace uso de *React Hooks* para gestionar el estado y el ciclo de vida de los componentes de forma eficiente. Los *hooks*, como useState, permiten manejar el estado en componentes funcionales, lo que ayuda a la reutilización del código y simplifica la lógica de los componentes. El concepto de React Hooks se encuentra detalladamente en la sección 7.4.1.2 Uso de React Hooks.
- **Nombres descriptivos:** Se utilizan nombres descriptivos para variables, funciones y componentes. Esto mantiene la claridad y legibilidad, facilitando a que cualquiera que lea el código lo pueda comprender de forma sencilla.
- **Componentes React en archivos separados:** El proyecto se organiza en componentes React para mejorar la modularidad y la mantenibilidad del código. Cada componente está en su propio archivo, lo que facilita su reuso y su búsqueda.

9.4.1.3. Estándares para el proceso de desarrollo

Para el proceso de desarrollo, se optó por la metodología Git Flow con el propósito de establecer un estándar coherente y eficiente en la gestión de ramas. Esta elección garantizó un enfoque estructurado en la administración de versiones y funcionalidades del proyecto. Se encuentra detalladamente descrito en el capítulo 10 Gestión de la Configuración.

Revisión de código: El equipo siguió un proceso de revisión de código en el que los miembros del equipo revisan y aprueban los cambios de código antes de la fusión en la rama principal. Se encuentra explicado con más detalle en el capítulo de 10 Gestión de la Configuración.

9.4.1.4. Estándares de documentación

Para la elaboración de la documentación, el equipo siguió rigurosamente los estándares establecidos por la Facultad de Ingeniería de la ORT. En particular, se hizo referencia a la "Guía de Entrega Final" y a los documentos 303 y 302. El documento 303 proporciona una lista de verificación de formato, mientras que el documento 302 establece las normas de presentación aplicables a trabajos finales de carrera. Estos recursos sirvieron como referencia clave para garantizar la calidad y coherencia de la documentación del proyecto.

9.4.2. Actividades para garantizar la calidad

9.4.2.1. Pruebas de usabilidad

Para cumplir con el objetivo de usabilidad (facilidad de uso para el 90% de los usuarios), se realizaron pruebas con usuarios reales, evaluando la facilidad de uso y la comprensión de las respuestas del *software*.

Además, el sistema sigue las heurísticas de Nielsen, un conjunto de principios reconocidos para evaluar la usabilidad del sistema [64]. A continuación, se explican con detalle algunos de ellos.

- **Visibilidad del estado del sistema**: Se busca que los usuarios puedan identificar fácilmente en qué parte del sistema se encuentran, resaltando la ubicación en el menú y proporcionando títulos claros para las secciones visitadas. Por ejemplo, si el usuario navega a "*Manage Files*", se muestra un título grande y en el menú marcado.
- **Diseño minimalista:** La plataforma está diseñada de forma de brindar una experiencia de usuario minimalista, presentando solo los componentes esenciales necesarios para su uso.
- **Consistencia y estándares:** Se mantiene una consistencia visual en toda la plataforma, siguiendo una paleta de colores compuesta por negro, gris, celeste y blanco, así como una tipografía uniforme.
- **Flexibilidad y eficiencia de uso:** El diseño de la plataforma se adapta tanto a usuarios novatos como a usuarios experimentados, garantizando una experiencia de usuario satisfactoria para ambos perfiles.

- **Prevención de errores:** Se implementaron medidas para evitar que los usuarios cometan errores, lo que contribuye a una experiencia de usuario más fluida y libre de problemas.

9.4.2.2. Interoperabilidad

Para cumplir con el objetivo de interoperabilidad se hizo el sistema *responsive*, para que se adapte al dispositivo por el cual el usuario quiere acceder.

Para ello, se realizaron pruebas en diferentes dispositivos, como escritorio, móvil y *Tablet*, estando la evidencia en el anexo 13.36 Evidencia de interoperabilidad.

9.4.2.3. Precisión y contextualización

Asegurar que las respuestas generadas presenten precisión y una adecuada contextualización involucra la implementación estratégica de tecnologías de inteligencia artificial, específicamente mediante la utilización de *chains* y la integración de un tipo particular de memoria conocida como Entity Memory.

El *chain of bots* en el contexto del desarrollo de sistemas de IA generativa se alude a la concepción de una cadena de *chatbots* que colaboran para dar una respuesta que no solo sea coherente, sino también pertinente. Cada *chatbot* en la cadena tiene una especialización en un dominio particular del conocimiento o del lenguaje, siendo responsable de originar una fracción de la respuesta, mejorando así la precisión. La respuesta final, entonces, es una conjunción de las respuestas de cada *chatbot* que compone la cadena siendo esta una respuesta optima. Este concepto se explica más detalladamente en la sección 5.7.1 *Chain*, y el cómo fue implementado en el sección 7.4.3.6.4 *Bots Chain*.

Por otro lado, Entity Memory refiere a una modalidad de memoria que resguarda información respecto a las entidades mencionadas durante el curso de la conversación. Una entidad puede ser entendida como un objeto o concepto mencionado en la charla, que podría ser una persona, un lugar, un evento, o un concepto abstracto, por ejemplo. Esto permite una mayor contextualización. Este concepto se explica detalladamente en la sección 5.5.2.1 Entity Memory y como se implementó en la sección 7.4.3.6.5 *Memory*.

9.4.2.4. Rendimiento

La optimización del rendimiento en sistemas tecnológicos es crítica para asegurar una experiencia de usuario ágil y satisfactoria. En este contexto, la implementación de gRPC para la comunicación en el *backend* se realizó con el objetivo de garantizar una comunicación más eficiente y robusta entre los servicios. gRPC, con su habilidad para crear conexiones rápidas, confiables y seguras, facilita notoriamente las operaciones en sistemas distribuidos, lo que incide directamente en un rendimiento superior y en la disminución de latencias.

En paralelo, la integración de Redis se llevó a cabo con la intención de potenciar la performance, especialmente en lo que respecta al acceso de los datos. Redis, al ser una base de datos en memoria, ofrece tiempos de respuesta extremadamente rápidos, permitiendo realizar operaciones de lectura y escritura de manera casi instantánea. Esto se traduce en una mejora sustancial en el manejo de datos, especialmente cuando se trata de gestionar grandes volúmenes de información o realizar operaciones de alta concurrencia.

Adicionalmente, se optó por utilizar Firebase Firestore, una base de datos NoSQL, buscando optimizar las operaciones de búsqueda de datos. Firestore proporciona una solución eficaz y escalable para gestionar los datos de manera estructurada y proporcionar búsquedas eficientes, incluso en entornos de alta demanda y con requisitos complejos de consulta de datos. Su arquitectura NoSQL permite una estructuración de datos más flexible y una escalabilidad que se adapta de manera eficiente a las necesidades cambiantes de la aplicación.

9.4.2.5. Extensibilidad y mantenibilidad

Asegurar la extensibilidad y mantenibilidad de un sistema de *software* es esencial para su evolución y adaptabilidad a futuras necesidades y desafíos. En este contexto, el empleo de principios y patrones de diseño se convierte en fundamental.

La adhesión a los principios SOLID y la práctica del Clean Code se implementaron con el objetivo de fomentar un código fuente que no solo sea robusto y confiable, sino también comprensible y fácil de modificar. Los principios SOLID, que incluyen Responsabilidad Única, Abierto/Cerrado, Sustitución de Liskov, Segregación de Interfaz e Inversión de Dependencias, proporcionan una base sólida para escribir *software* que sea fácil de mantener y escalar [65]. Paralelamente, el Clean Code persigue la claridad, la simplicidad y la organización del código,

facilitando así que los desarrolladores entiendan, expandan y modifiquen el código de manera efectiva.

En cuanto a los patrones de diseño como Singleton, Strategy y Factory Method, estos fueron aplicados con el propósito de resolver problemas comunes de diseño de *software* de manera eficiente y proporcionar esquemas que facilitan la reutilización del código.

Además, la inyección de dependencias fue utilizada como técnica para aumentar la modularidad y promover un acoplamiento débil entre las clases. Esta práctica permite que las dependencias de un objeto se suministren desde el exterior, facilitando la sustitución y manejo de componentes, y mejorando así la testabilidad y mantenibilidad del sistema.

9.4.3. Métricas de calidad

A continuación, se detallan las métricas utilizadas para evaluar tanto el producto final como el proceso de desarrollo.

9.4.3.1. Precisión y contextualización

Porcentaje de respuestas precisas: Esta métrica representa el porcentaje de respuestas generadas por el *chatbot* que se consideran precisas y cumplen con las expectativas del usuario. Para evaluarla, durante el proceso de validación con el *beta tester* (BID) y en las validaciones con otros usuarios, se solicita a los usuarios que asignen una puntuación del 1 al 5 para calificar la precisión de cada respuesta. Además, el equipo realizó pruebas exhaustivas, con sus familiares y cercanos, donde se probaba la precisión. El resultado fue de 88%. Es importante destacar que, aunque el objetivo inicial no se cumplió por completo, el equipo está trabajando continuamente en mejorarlo. Con este fin, se incorporó un *bot* más en la cadena de *bots* así como la implementación de un *agent*, como se explicó en el capítulo 7 Arquitectura, para aumentar la precisión. La medición de precisión es un desafío, ya que lo que se considera una respuesta 'precisa' puede variar según el usuario. Esto se debe a la subjetividad de la calidad, un aspecto que se mencionó en la introducción del capítulo.

Porcentaje de respuestas contextualizadas: Esta métrica evalúa el porcentaje de respuestas que tienen en cuenta el contexto de la consulta del usuario. De manera similar a la métrica de precisión, los usuarios otorgan una calificación del 1 al 5 para evaluar la contextualización de cada respuesta. El resultado fue del 93% de las respuestas contextualizadas, cumpliendo con el objetivo deseado.

A continuación, se presenta una parte del archivo Excel que se compartió con algunos usuarios que utilizaron Chat Your Data para su evaluación. Los usuarios debían registrar el caso de uso, describir el problema encontrado y calificar la contextualización y precisión de las respuestas proporcionadas por el *chatbot*. A la vez, el equipo agregaba su prioridad en resolverlo, para hacerle saber al usuario que se estaba trabajando en eso.

Nombre:	Fecha	Caso de uso	Captura de pantalla	Problema (en caso de tener)	Aspectos a mejorar	Aspectos positivos	Respuesta esta contextualizada	Respuesta es precisa	Prioridad
Micaela	19-ago	Generación de contenido		Solicité que me escriba un email y me respondio el contenido, pero no en formato de email	Necesita mejorar la generación de contenido	Respuesta rápida.	4	2	Alto
Pablo	21-ago	Subida documentos		Cuando ocurre un error, la plataforma no proporciona información útil.	Retroalimentación sobre errores	Interfaz atractiva.	4	4	Alto
Paula	23-ago	Estadísticas de documentos		Me gustaría ver estadísticas más detalladas de los documentos.	Considerar agregar estadísticas más completas.	Interfaz atractiva.	3	4	Alto
Diego	29- ago	Compatibilidad de idioma		Le hable en espanol y me respondio parte en espanol y parte en ingles	Mejorar la comprensión en múltiples idiomas.	Respuesta rápida.	2	2	Medio
María	30-ago	Referencias		Me respondio perfecto pero las refernecias que me dio no fueron exactas	Referencias	Diseño intuitivo.	4	4	Medio
Juan	1/9	Análisis financiero		Realicé una pregunta sobre análisis financiero y la respuesta fue inexacta.	Mejorar la precisión en análisis financieros.	Interfaz sencilla.	4	2	Bajo
Elena	3-sept	Pregunte sobre mi documento		Las respuestas demoran mucho en cargar	demora en cargar	Fácil navegación.	3	4	Alto
Miguel	6-sept	-		A veces, no comprende bien mis preguntas ambiguas.	Mejorar la comprensión de preguntas ambiguas.	Facil de usar	4	3	Alto

Ilustración 61- Evidencia del Excel de validación

Profundidad de contextualización: Esta métrica cuantifica la cantidad de información contextual que se proporciona en cada respuesta, lo cual es esencial para lograr una comunicación efectiva. Se mide la cantidad de detalles relevantes que se incorporan en las respuestas generadas por el *chatbot* para enriquecer la interacción con el usuario.

Para evaluar esta métrica, el equipo realizó conversaciones extensas con el *chat* para determinar hasta qué punto podía mantener el contexto en las interacciones. Esto implicó hacer preguntas sobre diferentes temas y verificar si el *chatbot* podía volver al tema anterior sin problemas.

9.4.3.2. Rendimiento

Tiempo de respuesta: Se implementaron mediciones de tiempo de respuesta a nivel de *backend* para analizar y optimizar el rendimiento del sistema. Para realizar estas mediciones, el equipo utilizó la librería Stopwatch en NestJS, que permitió cronometrar las respuestas generadas por el *chatbot*.

Se identificó que, en algunos casos, las respuestas experimentaban demoras de más de 10 segundos. Estas demoras se debían a la priorización del equipo en lograr respuestas altamente contextuales y precisas en lugar de enfocarse en la velocidad de respuesta. Para mejorar la calidad de las respuestas, se decidió implementar una cadena de *bots* que contribuye a una

mayor precisión, pero puede aumentar el tiempo de respuesta. Además, dado que este proyecto es un MVP, aún no se ha implementado la tecnología de *streaming* [66], la cual se espera que mejore significativamente la velocidad de respuesta en futuras iteraciones.

El objetivo inicial del proyecto consistía en desarrollar un sistema capaz de manejar simultáneamente hasta 1.000 usuarios activos sin degradar significativamente su rendimiento. Sin embargo, se priorizó la funcionalidad y la calidad del *chatbot* en sí mismo, centrándose en garantizar que pudiera proporcionar respuestas precisas y contextuales a las consultas de los usuarios. Si bien en la etapa actual del proyecto no se ha logrado la capacidad completa de manejar 1.000 usuarios activos simultáneamente, es importante destacar que este objetivo sigue siendo una meta a largo plazo. El equipo reconoce la importancia de la escalabilidad y está trabajando en futuras iteraciones del sistema para abordar esta limitación y mejorar el rendimiento en términos de capacidad de usuarios concurrentes.

9.4.3.3. Métricas de mantenibilidad

Calidad del código: Se utilizaron herramientas de análisis de código estático, ESLint y Prettier, para evaluar cuantitativamente la calidad del código fuente. Esto ayuda a mantener un código organizado y de alta calidad, facilitando futuras actualizaciones y mejoras.

9.4.3.4. Métricas de usabilidad

Porcentaje de usuarios satisfechos: Para medir la usabilidad del *software*, se llevó a cabo una evaluación que se centró en el grado de satisfacción de los usuarios con la plataforma. Se realizó una encuesta de satisfacción a una muestra representativa de usuarios, en la cual se plantearon preguntas específicas sobre la facilidad de uso, la funcionalidad y la estética de la plataforma. Las respuestas recopiladas en la encuesta se sometieron a análisis con el fin de determinar el índice de satisfacción de los usuarios.

Preguntas de Evaluación:

1. ¿Qué tan sencillo e intuitivo encontró el uso de la aplicación?
2. ¿Comprendió de manera clara cómo utilizar la aplicación?
3. ¿Siente que se mantiene bien informado acerca de lo que está ocurriendo en la aplicación?
4. ¿Las funcionalidades de la plataforma le resultaron claras o percibió alguna carencia en alguna de ellas?

Resultados de la Evaluación:

Según los resultados de la encuesta, se observa que el 75% de los usuarios expresaron satisfacción con la plataforma, otorgando calificaciones de 4 o 5 en la encuesta. Si se considera que el 3 es un buen resultado, entonces un 92% de los usuarios quedó satisfecho. Solo un 8.3% de los encuestados seleccionó la calificación 2, y ningún usuario calificó con 1.

En cuanto a la pregunta sobre qué tan fácil fue usar la plataforma, el equipo logró cumplir con su objetivo de alcanzar un nivel de satisfacción del 90% en usabilidad, considerando únicamente las calificaciones de 4 y 5.

Satisfacción de la plataforma

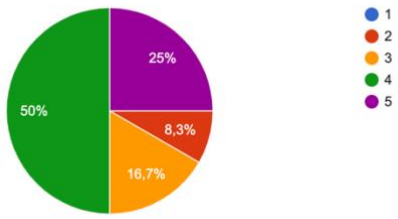


Ilustración 62- Resultado de la satisfacción de los usuarios

Facilidad de uso de la plataforma

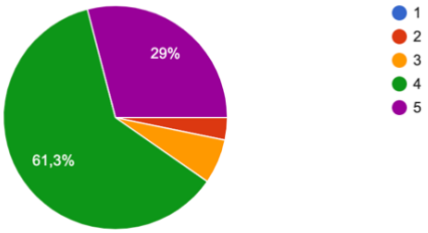


Ilustración 63- Resultado de facilidad de uso de la plataforma

Tiempo de Aprendizaje: Se determinó cuánto tiempo lleva a los usuarios aprender a usar la plataforma, lo que ayudó a identificar áreas de mejora en la experiencia del usuario.

Para este propósito, se seleccionó un grupo de usuarios sin experiencia previa en la plataforma. A estos usuarios se les pidió que hagan una serie de tareas básicas específicas, y se registró el tiempo que les tomó completar cada una de estas tareas.

Algunas de las tareas que se les asignaron incluyeron:

- Subir un documento a la plataforma y abrir dicho documento.
- Crear una nueva conversación en el *chat*.
- Formular preguntas al *chat* y proporcionar retroalimentación sobre las respuestas.
- Formular preguntas al *chat* y copiar las respuestas.
- Hacer una pregunta al *chat* relacionada con la generación de contenido.

Luego se decidió por hacer la prueba de todo el flujo básico de actividades, que abarcó desde el comienzo de la página, el inicio de sesión hasta la generación de contenido, habiendo pasado por el proceso de subida de archivos.

Resultados de la evaluación:

El tiempo promedio de aprendizaje requerido para completar el flujo básico de actividades, que incluye el inicio de sesión, la carga de 5 archivos, la formulación de 5 preguntas básicas sobre los archivos y la generación de contenido fue de 4 minutos y 45 segundos (4:45).

9.4.3.5. Métricas del proceso

El equipo utilizó métricas para evaluar el proceso del proyecto. Entre ellas se incluyen la Velocidad del Equipo, el Desvío de Estimaciones, Burndown Chart y el Re-Trabajo. Las métricas de gestión se encuentran detalladas en la sección 8.4.3 Métricas de gestión.

10. Gestión de la Configuración

10.1. Introducción

A continuación, se presentará el enfoque adoptado para la gestión de la configuración del *software* (SCM) durante el desarrollo del proyecto.

Es inherente a los sistemas de software experimentar cambios tanto en su estructura como en sus requerimientos a lo largo de su ciclo de vida. Esto implica la constante aparición de nuevas versiones que deben ser adecuadamente gestionadas y mantenidas. Dado que el proyecto se encuentra en constante evolución y cuenta con un equipo activamente trabajando en él, la adecuada gestión de su configuración se vuelve esencial.

En la administración de la configuración, según el libro "Ingeniería de *Software* 9 - Pearson" de Sommerville, se pueden encontrar cuatro actividades:

Administración del cambio: Hacer seguimiento de las peticiones de cambio al *software*, de parte de los potenciales clientes y desarrolladores. La administración del cambio tiene la intención de asegurar que la evolución del sistema sea un proceso gestionado en el que se da prioridad a los cambios más urgentes, analizando los costos, y beneficios.

Gestión de versiones: Hacer seguimiento de las numerosas versiones de los componentes del sistema. También incluye asegurar que los cambios hechos por diferentes desarrolladores no interfieran entre sí.

Construcción del sistema: El proceso de juntar los componentes del programa, datos y librerías, y luego compilarlos y vincularlos para crear un sistema ejecutable.

Gestión de *releases*: Preparar el *software* para la entrega externa y hacer un seguimiento de las versiones que se entregaron al cliente.

El equipo decidió hacer hincapié en algunas de estas actividades.

10.2. Elección de herramientas

10.2.1. Desarrollo del *software*: Git

En lo que respecta a la elección de la herramienta para el seguimiento, versionado y almacenamiento del desarrollo de *software*, se determinó que Git sería la opción más adecuada.

Dicha elección se fundamentó en diversas razones clave:

Experiencia del equipo: Todos los integrantes del equipo cuentan con experiencia previa en el uso de Git, lo que facilita la implementación y el manejo eficiente de la herramienta.

Amplia utilización: Git es una de las herramientas más utilizadas en la industria del *software*, lo que garantiza su confiabilidad y compatibilidad con las mejores prácticas en el desarrollo.

Colaboración simplificada: Git ofrece una plataforma robusta para la colaboración en el desarrollo de *software*, permitiendo a los miembros del equipo trabajar juntos de manera eficiente y efectiva.

Trabajo independiente: Git permite que cada miembro del equipo trabaje de manera independiente en su propio entorno, facilitando la división de tareas y la realización paralela de actividades.

Por consiguiente, se determinó que GitHub sería la plataforma elegida como proveedor de servicios de repositorios.

10.2.2. Documentación

El equipo optó por utilizar diversas herramientas para gestionar la documentación del proyecto. Todas estas herramientas se eligieron para fomentar la colaboración y facilitar el flujo de información dentro del equipo.

En las etapas iniciales, como se mencionó previamente, se empleó Notion. Esta plataforma se utilizó para registrar todo el proceso de investigación, validación y descubrimiento, incluyendo notas de llamadas y reuniones, así como cursos, videos y apuntes relevantes para futuras referencias. Los contenidos se etiquetaron y organizaron en distintas secciones para mantener una estructura organizada.

A medida que el proyecto avanzó y se procedió a redactar la documentación más estructurada y formal, se trasladó a Google Drive. En esta herramienta, se creó una carpeta global para el proyecto que contenía múltiples subcarpetas, como una por capítulo y otra para información importante, entre otras.

Además, se aprovecharon aplicaciones como Google Docs y Google Sheets para tareas específicas, como la gestión de riesgos, la validación con el *beta tester*, la creación de presentaciones y la implementación de formularios para validar el problema y el prototipo.

Toda esta información se migró posteriormente a Microsoft Word para redactar la versión final de la documentación. El equipo consideró que Word ofrecía una mayor flexibilidad para cumplir con los requisitos y el estándar de documentos formales.

10.3. Elementos de configuración

10.3.1. Desarrollo de *software*

Elementos de configuración	**Nombre de repositorio**
Backend PluginAI	PluginAI
Backend Server	ChatYourDataAPI
Frontend	chatYourData-frontend

Tabla 10- Elementos de configuración - desarrollo de *software*

PluginAI

Este componente es el núcleo de la inteligencia del sistema. Su función principal radica en procesar las solicitudes de los usuarios, analizar datos y generar respuestas pertinentes y contextualizadas.

Backend

Actúa como el intermediario entre el *frontend* y el componente de inteligencia artificial (PluginAI). Su responsabilidad principal incluye la gestión de funciones como la autenticación de usuarios y el almacenamiento de datos.

Frontend

Sirve como la interfaz de interacción primaria entre los usuarios y Chat Your Data. Es el punto de entrada inicial, donde se recopilan las solicitudes de los usuarios y se presentan las respuestas generadas por el sistema.

10.4. Importancia de repositorios separados en la arquitectura de *software*

Dentro del ámbito de la arquitectura de *software*, las buenas prácticas no solo se centran en cómo se diseña la estructura interna de un sistema, sino también en cómo se gestiona y organiza el código fuente. A continuación, se exploran las ventajas de mantener los componentes en repositorios separados:

1. **Aislamiento y seguridad**: Los repositorios independientes brindan una capa adicional de protección. Al limitar el acceso según la necesidad específica, se minimizan las vulnerabilidades y se protege cada componente de accesos no autorizados o inadvertidos.
2. **Integración y despliegue continuo (CI/CD)**: La separación en repositorios facilita la implementación de *pipelines* de integración y despliegue diseñados específicamente para cada componente. Esta modularidad ayuda a tener ciclos de despliegue más ágiles y menos propensos a errores.
3. **Especialización del equipo**: La organización en repositorios separados permite que los equipos especializados se centren en áreas específicas del proyecto. Esto hace que el desarrollo sea más eficiente, alineado con las mejores prácticas de desarrollo ágil.
4. **Escalabilidad y mantenibilidad**: Un diseño arquitectónico que favorece repositorios separados permite escalar o modificar un componente de forma independiente, sin afectar o depender de otros. Esta independencia facilita el mantenimiento y mejora la adaptabilidad del sistema.
5. **Control de versiones y dependencias**: Gestionar versiones y dependencias se vuelve más manejable cuando cada componente tiene su propio repositorio. Las actualizaciones y cambios se rastrean con precisión, evitando conflictos y dependencias cruzadas.
6. **Transparencia y responsabilidad**: Con un historial de cambios específico para cada componente, se promueve la transparencia en el desarrollo. Este registro detallado refuerza la responsabilidad del equipo, facilitando la detección y corrección de errores.

En resumen, la segmentación en repositorios separados se presenta como una clara demostración de su adhesión a las mejores prácticas en arquitectura de *software*. Esta estructura no solo resalta su robustez y pensamiento arquitectónico, sino que también garantiza una gestión óptima, mayor seguridad y una escalabilidad fluida.

10.5. Forma de trabajo

La adopción de un enfoque estándar, conocido como GitFlow [67], en la gestión de repositorios fue fundamental para mantener la organización y la colaboración efectiva del equipo. Se establecieron dos ramas principales en el flujo de trabajo: la rama *main* (principal) y la rama *develop* (desarrollo). La rama *main* representaba la versión estable y funcional del proyecto, mientras que la rama *develop* servía como espacio para la integración continua de cambios y nuevas características.

Además de las ramas principales, se implementaron categorías específicas para la creación de otras ramas, como *fix* (corrección) y *feature* (característica). Las ramas de *fix* se utilizaron para abordar correcciones de errores y problemas identificados, asegurando que la rama principal *main* se mantuviera confiable y libre de problemas. Por otro lado, las ramas de *feature* se crearon para desarrollar nuevas características o funcionalidades del proyecto de manera aislada, lo que permitía un enfoque modular y una mayor claridad en el desarrollo de cada elemento.

Para integrar los cambios en las ramas de *feature* y *fix* en las ramas principales, se adoptó la práctica de *pull request* (solicitudes de extracción). Las solicitudes de extracción se utilizaron para revisar y validar los cambios antes de fusionarlos en la rama *develop*. Luego de que el *pull request* estuviese corregido, estaba pronto para hacer el *merge*. Esto aseguraba una revisión y aprobación por parte del equipo antes de incorporar nuevos elementos al proyecto, manteniendo la calidad y la coherencia del código. Una vez que está pronto en la rama *develop*, y se decide subir a producción, se hacía *merge* con la rama *main*, siendo esta la que contiene el código en el ambiente de producción.

El equipo implementó la norma de que un *pull request* debía ser revisado y aprobado por un compañero antes de incorporar esa rama con la rama *develop*. Esto garantizaba una revisión exhaustiva y la aprobación antes de incorporar los cambios, lo que contribuyó a mantener la calidad y la coherencia del código.

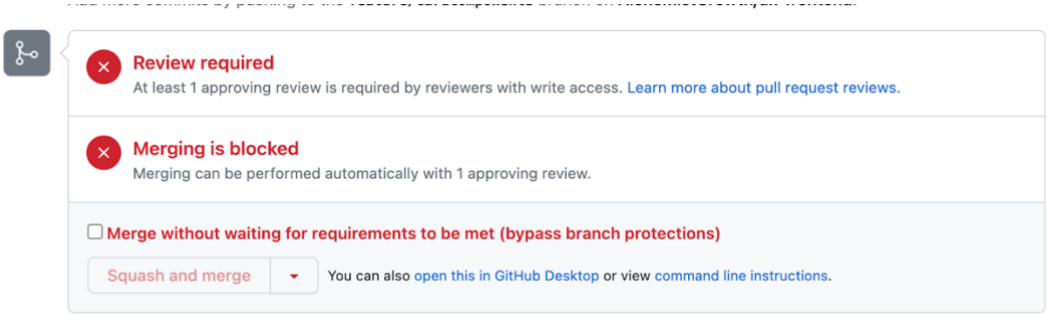


Ilustración 64- Evidencia de uso de *pull request*

10.6. Ambientes de desarrollo

Durante el proceso de desarrollo, el equipo gestionó dos ambientes distintos. El primero, conocido como *staging*, reflejaba la última versión del código y correspondía al estado actual de la rama *develop*. Este entorno se utilizaba para pruebas y ajustes antes de llevar a cabo los releases.

Por otro lado, el ambiente de producción representaba la rama *main*, que contenía la versión más estable y probada del sistema. La actualización de esta rama solo ocurría con los *releases* programados, después de completar pruebas internas exhaustivas.

11. Conclusiones y lecciones aprendidas

11.1. Estado actual

En el estado actual del proyecto, el equipo se siente sumamente satisfecho con los logros alcanzados hasta el momento. Se ha logrado despertar un alto interés entre potenciales clientes, lo que demuestra la relevancia y la demanda de la solución en el mercado. Inicialmente, se tomó la decisión de enfocarse en un único *beta tester*, el BID, reconocido por su importancia. Sin embargo, a medida que avanzaba el proyecto, surgieron numerosas organizaciones interesadas en la propuesta y manifestaron su deseo de incorporarla. Este nivel de interés en el proyecto muestra la validez de la solución y su potencial en el mercado.

El producto actualmente está en funcionamiento, lo que representa un logro significativo en el desarrollo de esta solución. Si bien se reconoce que todavía hay funcionalidades por implementar y mejoras, es importante destacar que se alcanzó a lograr un primer MVP completamente funcional.

Este MVP ha permitido al equipo probar y validar conceptos clave, así como demostró la viabilidad técnica de la solución y su capacidad para cumplir con los objetivos iniciales del proyecto.

11.2. Pasos a seguir

A continuación, se describen los siguientes pasos a seguir en el proyecto:

11.2.1. A nivel de producto

Finalizar las funcionalidades de usuarios y organizaciones: Se completarán las funcionalidades relacionadas con usuarios y organizaciones. Esto abarca desde permitir que los administradores de organizaciones inviten a nuevos usuarios hasta ofrecer a los usuarios la capacidad de editar su perfil personal, y proporcionar a los administradores de organizaciones la opción de editar la información relacionada con la organización que gestionan.

Desarrollar una *landing page*: Se planifica la creación de una *landing page* que permita a personas y organizaciones acceder al servicio y elegir el plan que mejor se adapte a sus necesidades.

Integración de pagos: Se llevará a cabo la integración de un sistema de pagos que permita a los usuarios seleccionar y pagar por el plan que elijan en la *landing page*.

Pruebas y ajustes: Se realizarán pruebas exhaustivas de todas las funcionalidades implementadas hasta el momento, asegurando que el sistema funcione sin problemas y cumpla con los requisitos del usuario.

Afinar las respuestas: Se realizará un proceso de *fine tuning* de las respuestas generadas por la plataforma. Esto implica revisar y mejorar la calidad de las respuestas proporcionadas, garantizando así una experiencia de usuario excepcional y precisa. Se basará en la retroalimentación de los usuarios y la mejora continua del modelo de lenguaje utilizado en la plataforma.

Desarrollo adicional de funcionalidades: En base a la retroalimentación de los usuarios y las necesidades emergentes, se considerará el desarrollo adicional de características y mejoras en la plataforma.

Mejorar el rendimiento: Uno de los aspectos críticos que requiere atención es el rendimiento de la plataforma. Esto abarca tanto la velocidad de respuesta como la capacidad de manejar múltiples usuarios concurrentemente. Por lo tanto, se enfocará en optimizar el rendimiento para permitir que más de 1.000 usuarios utilicen la plataforma simultáneamente y que no se experimenten demoras significativas en las respuestas.

Implementación de seguridad: Se implementarán medidas de seguridad adicionales para garantizar la protección de los datos y la información del usuario.

11.2.2. A nivel de mercado

Comunicación activa con organizaciones interesadas: El equipo seguirá enfocado en la búsqueda proactiva de nuevas organizaciones interesadas en unirse a la plataforma y en la realización de demostraciones de la plataforma para mostrar sus capacidades y beneficios.

Presencia en redes: El equipo busca aumentar la presencia en redes sociales y otros canales digitales para aumentar la visibilidad de la solución y llegar a un público más amplio. Esto incluirá la creación de contenido relevante, la participación en conversaciones relevantes y la interacción con posibles clientes en línea.

Participación en eventos y conferencias: El equipo considerará la participación en eventos y conferencias de la industria como una oportunidad para establecer contactos, presentar la plataforma y generar interés entre clientes potenciales.

11.3. Cumplimiento de objetivos

El equipo valora profundamente la experiencia del proyecto como un período de crecimiento tanto personal como académico. Desde la identificación del problema hasta el desarrollo de la solución ideada, presentando desafíos significativos que fueron superados con éxito.

11.3.1. A nivel de proyecto

El equipo se siente muy contento de haber alcanzado los objetivos clave, que incluyen la creación de un Mínimo Producto Viable (MVP) que es tanto extensible como adaptable. Esto genera que el sistema se pueda adaptar fácilmente a las necesidades de los usuarios. Además, el equipo aprovechó las últimas tecnologías para desarrollar un sistema funcional. Este logro representa un valioso aprendizaje que servirá a lo largo de la vida.

11.3.2. A nivel de producto

El equipo considera que identificó un problema común en las organizaciones y que Chat Your Data puede aportar un valor significativo a estas organizaciones. En primer lugar, se desarrolló un producto de calidad profesional, aunque aún hay margen para mejoras. Este producto ya es funcional y utilizable, lo que, en la opinión del equipo, es una gran satisfacción. Además, no sólo agrega valor a las organizaciones, sino que también mejora la experiencia de empleados y usuarios en general.

11.3.3. A nivel académico

El equipo se siente muy contento por la oportunidad de aprender nuevas tecnologías durante el desarrollo del proyecto. Estas nuevas habilidades técnicas no solo se adquirieron con éxito, sino que también se implementaron de manera efectiva en el proyecto, enriqueciendo así la capacidad del equipo para abordar desafíos tecnológicos en el futuro. Además, la aplicación de conocimientos previos contribuyó en la creación de un proyecto sólido.

11.3.4. A nivel de equipo

El equipo considera que el éxito alcanzado se debe en gran medida a la efectiva división de roles, al compromiso de sus miembros y a la relación entre los integrantes. Desde el inicio, se enfocó en disfrutar del trabajo conjunto y mantener una comunicación efectiva y fluida. El equipo está sumamente contento de haber creado un emprendimiento y con ganas de seguirlo en el futuro.

11.4. Lecciones aprendidas

A lo largo del proyecto Chat Your Data, el equipo adquirió una serie de lecciones fundamentales.

En primer lugar, los integrantes comprendieron la importancia del trabajo en equipo y cómo la colaboración efectiva puede superar incluso los desafíos más complejos. Además, llevarse bien entre los integrantes, la organización y la división eficiente de las tareas se convirtieron en un pilar fundamental del buen funcionamiento del equipo. Por otro lado, basarse en un marco de trabajo ágil y gestionar el proyecto fueron esenciales para maximizar la productividad y mantener el enfoque en los objetivos del proyecto. Trabajar en colaboración con una organización de la importancia y tamaño del BID, y con un propósito que beneficia al mundo, en especial a América Latina, fue un gran honor.

Uno de los mayores logros del equipo fue la creación de una plataforma *"end to end"*, abarcando desde la interfaz de usuario hasta la gestión de bases de datos y la implementación del despliegue por cuenta propia. A lo largo de este proyecto, se enfrentaron a diversos desafíos, incluido el aprendizaje sobre inteligencia artificial y campos en evolución y la complejidad del despliegue de la aplicación en AWS.

Como se mencionó en la sección de Lecciones Aprendidas del capítulo de Arquitectura, uno de los grandes aprendizajes fue la capacidad de adquirir conceptos y herramientas por cuenta propia, especialmente en el campo de la IA, superando el temor inicial a estos conceptos. Además, la comprensión de que todas estas tecnologías estaban en constante cambio y evolución fortaleció la capacidad del equipo para adaptarse, siendo esto fundamental en un equipo. [33]

El proyecto también proporcionó una visión profunda sobre cómo identificar problemas del mundo real y enfocarse en soluciones que realmente beneficien a quienes enfrentan esos problemas. La importancia de centrarse en el problema en lugar de solo en el producto en sí se convirtió en una lección clave.

En paralelo, la gestión de la planificación y la gestión de riesgos demostraron ser cruciales para el éxito del proyecto, proporcionando una base sólida para la toma de decisiones durante todo el proceso.

Por último, pero no menos importante, el equipo desarrolló una amplia variedad de habilidades valiosas, desde liderar un proyecto completo hasta la capacidad de aprender y aplicar nuevos conceptos y tecnologías de manera independiente, superando desafíos constantemente.

12. Referencias Bibliográficas

[1] IBM, «Natural Language Processing (NLP),» Marzo 2023. [En línea]. Available: www.ibm.com/topics/natural-language-processing. [Ultimo Acceso: 16 de Marzo 2023]

[2] LangChain, «Introduction - LangChain,» Marzo 2023. [En línea]. Available: https://python.langchain.com/docs/get_started/introduction. [Ultimo Acceso: 24 de Marzo 2023]

[3] AWS, «¿Qué es un modelo de lenguaje grande?,» [En línea]. Available: https://aws.amazon.com/es/what-is/large-language-model/. [Ultimo Acceso: 22 de Marzo 2023]

[4] AWS, «¿Qué es una base de datos vectorial?,» Marzo 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/vector-databases/. [Ultimo Acceso: 22 de Marzo 2023]

[5] DesignThinking, «Introduccion a design thinking,» Marzo 2023. [En línea]. Available: https://www.designthinking.es/inicio/index.php. [Ultimo Acceso: 23 de Marzo 2023]

[6] E. Ries, El método Lean Startup, 2011. [Ultimo Acceso: 07 de Abril 2023]

[7] G. D. Backer, «Problem-Solution Fit: What Is It + How To Get It,» Marzo 2023. [En línea]. Available: https://gustdebacker.com/problem-solution-fit/. [Ultimo Acceso: 08 de Abril 2023]

[8] G. d. Backer, «Product-Market Fit: What Is It + How To Get It,» Marzo 2023. [En línea]. Available: https://gustdebacker.com/product-market-fit/. [Ultimo Acceso: 10 de Abril 2023]

[9] G. d. Backer, «TAM, SAM, SOM: Understanding the Size and Structure of Your Market,» Marzo 2023. [En línea]. Available: https://gustdebacker.com/tam-sam-som-market/. [Ultimo Acceso: 12 de Abril 2023]

[10] ProductPlan, «Beta Test,» Abril 2023. [En línea]. Available: www.productplan.com/glossary/beta-test/. [Ultimo Acceso: 05 de Mayo 2023]

[11] BID, «Banco Interamericano de Desarrollo (BID),» Abril 2023. [En línea]. Available: https://www.iadb.org/es/quienes-somos/acerca-del-bid. [Ultimo Acceso: 22 de Abril 2023]

[12] iTMPlatform, «Ciclos de vida clásico, iterativo y ágil,» Abril 2023. [En línea]. Available: https://www.itmplatform.com/es/blog/ciclos-de-vida-clasico-iterativo-y-agil. [Ultimo Acceso: 22 de Abril 2023]

[13] IebSchool, «Metodologias agiles Scrum,» Abril 2023. [En línea]. Available: https://www.iebschool.com/blog/que-son-metodologias-agiles-agile-scrum/. [Ultimo Acceso: 23 de Abril 2023]

[14] Fhios, «Kanban - pro y contras,» Abril 2023. [En línea]. Available: ttps://www.fhios.es/metodologia-kanban-pros-y-contras/. [Ultimo Acceso: 24 de Abril 2023]

[15] KanbanTool, «Metodologia Kanban,» Abril 2023. [En línea]. Available: https://kanbantool.com/es/metodologia-kanban. [Ultimo Acceso: 25 de Abril 2023]

[16] K. S. &. J. Sutherland, La Guía de Scrum, 2020. [Ultimo Acceso: 27 de Abril 2023]

[17] Atlassian, «Scrum board - Software,» [En línea]. Available: https://www.atlassian.com/software. [Ultimo Acceso: 20 de Abril 2023]

[18] FutureOfLife, «Pause Giant AI Experiments,» Abril 2023. [En línea]. Available: https://futureoflife.org/open-letter/pause-giant-ai-experiments/. [Ultimo Acceso: 22 de Abril 2023]

[19] CloudGoogle, «What is artificial intelligence?,» Abril 2023. [En línea]. Available: https://cloud.google.com/learn/what-is-artificial-intelligence?hl=es-419. [Ultimo Acceso: 28 de Abril 2023]

[20] AWS, «Que es machine learning?,» Abril 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/machine-learning/. [Ultimo Acceso: 28 de Abril 2023]

[21] J. Brownlee, «Supervised and Unsupervised Machine Learning Algorithms,» Mayo 2023. [En línea]. Available: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/. [Ultimo Acceso: 02 de Mayo 2023]

[22] AWS, «What is neural network?,» Abril 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/neural-network/. [Ultimo Acceso: 02 de Mayo 2023]

[23] DeepLearning, «Deep Learning,» [En línea]. Available: https://www.deeplearning.ai/. [Ultimo Acceso: 03 de Mayo 2023]

[24] CloudGoogle, «Generative AI,» [En línea]. Available: https://cloud.google.com/use-cases/generative-ai?hl=es. [Ultimo Acceso: 05 de Mayo 2023]

[25] P. Neurips, «Attention Is All You Need,» Mayo 2023. [En línea]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf. [Ultimo Acceso: 06 de Mayo 2023]

[26] AWS, «What is NLP?,» Abril 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/nlp/. [Ultimo Acceso: 08 de Mayo 2023]

[27] Qualtrics, «Natural Language Generation,» [En línea]. Available: https://www.qualtrics.com/experience-management/customer/natural-language-generation/. [Ultimo Acceso: 08 de Mayo 2023]

[28] Elastic, «Large Language Models,» Abril 2023. [En línea]. Available: https://www.elastic.co/es/what-is/large-language-models. [Ultimo Acceso: 16 de Abril 2023]

[29] Chatgpt, «GPT-3,» Mayo 2023. [En línea]. Available: https://chatgpt.ch/es/diferencias-entre-gpt-3-5-y-gpt-4/. [Ultimo Acceso: 13 de Mayo 2023]

[30] PromptingGuide, «Prompt Engineering,» Mayo 2023. [En línea]. Available: https://www.promptingguide.ai/es. [Ultimo Acceso: 16 de Mayo 2023]

[31] L. Prompting, «Chain of Thought,» Mayo 2023. [En línea]. Available: https://learnprompting.org/es/docs/intermediate/chain_of_thought. [Ultimo Acceso: 16 de Mayo 2023]

[32] Upnify, «Temperatura IA,» Abril 2023. [En línea]. Available: https://upnify.com/es/diccionario-ventas/temperatura-ia.html. [Ultimo Acceso: 02 de Mayo 2023]

[33] AWS, «Bot,» Mayo 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/bot/. [Ultimo Acceso: 18 de Mayo 2023]

[34] Oracle, «What is a chatbot?,» Abril 2023. [En línea]. Available: https://www.oracle.com/chatbots/what-is-a-chatbot/. [Ultimo Acceso: 16 de Abril 2023]

[35] kbdhunga, «Cadena de bots,» Junio 2023. [En línea]. Available: https://medium.com/@kbdhunga/exploring-langchain-chains-agents-a-quick-overview-9d0a8c4d7ba0. [Ultimo Acceso: 20 de Junio 2023]

[36] LangChain, «Memory,» [En línea]. Available: https://js.langchain.com/docs/modules/memory/. [Ultimo Acceso: 19 de Junio 2023]

[37] Deloitte, «Responsible use of generative ai,» Junio 2023. [En línea]. Available: https://www2.deloitte.com/us/en/pages/consulting/articles/responsible-use-of-generative-ai.html. [Ultimo Acceso: 17 de Abril 2023]

[38] TowardsDataScience, «Chunks,» Junio 2023. [En línea]. Available: https://towardsdatascience.com/chunking-in-nlp-decoded-b4a71b2b4e24. [Ultimo Acceso: 19 de Junio 2023]

[39] Dev.to, «Recursive Character Text Splitting,» Junio 2023. [En línea]. Available: https://dev.to/eteimz/understanding-langchains-recursivecharactertextsplitter-2846. [Ultimo Acceso: 20 de Junio 2023]

[40] DocsN8n, «Recursive Character Text Splitting,» Junio 2023. [En línea]. Available: https://docs.n8n.io/integrations/builtin/cluster-nodes/sub-nodes/n8n-nodes-langchain.textsplitterrecursivecharactertextsplitter/. [Ultimo Acceso: 26 de Abril 2023]

[41] Pinecone, «What is similarity search?,» Junio 2023. [En línea]. Available: https://www.pinecone.io/learn/what-is-similarity-search/. [Ultimo Acceso: 22 de Junio 2023]

[42] LabelBox, «How vector similarity search works?,» Junio 2023. [En línea]. Available: https://labelbox.com/blog/how-vector-similarity-search-works/. [Ultimo Acceso: 24 de Junio 2023]

[43] Perplexity, «Chain,» Junio 2023. [En línea]. Available: www.perplexity.ai/search. [Ultimo Acceso: 26 de Junio 2023]

[44] GeeksForGeeks, «Agents artificial intelligence,» Junio 2023. [En línea]. Available: https://www.geeksforgeeks.org/agents-artificial-intelligence/. [Ultimo Acceso: 27 de Junio 2023]

[45] Pinecone, «LangChain Agents,» Junio 2023. [En línea]. Available: https://www.pinecone.io/learn/series/langchain/langchain-agents/#:~:text=The%20agent_scratchpad%20is%20where%20we,enabling%20continuity%20in%20agent%20actions.. [Ultimo Acceso: 20 de Junio 2023]

[46] LangChain, «MRKL Agents,» Junio 2023. [En línea]. Available: https://python.langchain.com/docs/modules/agents/how_to/custom_mrkl_agent. [Ultimo Acceso: 03 de Julio 2023]

[47] Arxiv, «MRKL Systems,» Junio 2023. [En línea]. Available: https://arxiv.org/abs/2205.00445. [Ultimo Acceso: 04 de Julio 2023]

[48] LangChain, «Introducción de LangChain,» Mayo 2023. [En línea]. Available: https://python.langchain.com/docs/get_started/introduction. [Ultimo Acceso: 05 de Mayo 2023]

[49] Hotjar, «User Personas,» Mayo 2023. [En línea]. Available: www.hotjar.com/blog/user-personas. [Ultimo Acceso: 08 de Mayo 2023]

[50] Bynder, «Customer Journey,» Junio 2023. [En línea]. Available: https://www.bynder.com/en/glossary/customer-journey/. [Ultimo Acceso: 15 de Junio 2023]

[51] Rdstation, «Mapa de empatia,» Junio 2023. [En línea]. Available: www.rdstation.com/blog/es/mapa-de-empatia. [Ultimo Acceso: 18 de Junio 2023]

[52] Atlassian, «User Stories,» Junio 2023. [En línea]. Available: https://www.atlassian.com/es/agile/project-management/user-stories. [Ultimo Acceso: 18 de Junio 2023]

[53] AWS, «Microservices,» Julio 2023. [En línea]. Available: https://aws.amazon.com/es/microservices/. [Ultimo Acceso: 19 de Julio 2023]

[54] IBM, «Three tier architecture,» Julio 2023. [En línea]. Available: https://www.ibm.com/mx-es/topics/three-tier-architecture. [Ultimo Acceso: 04 de Julio 2023]

[55] Perplexity, «Que es la inversión de dependencias? y sus ventajas?,» Agosto 2023. [En línea]. Available: www.perplexity.com. [Ultimo Acceso: 01 de Agosto 2023]

[56] kbdhunga, «Chain Agents,» Agosto 2023. [En línea]. Available: https://medium.com/@kbdhunga/exploring-langchain-chains-agents-a-quick-overview-9d0a8c4d7ba0. [Ultimo Acceso: 05 de Agosto 2023]

[57] RefactoringGuru, «Factory Method,» Agosto 2023. [En línea]. Available: https://refactoring.guru/design-patterns/factory-method. [Ultimo Acceso: 10 de Agosto 2023]

[58] Hipertextual, «Calculator,» Agosto 2023. [En línea]. Available: https://hipertextual.com/2023/02/matematicas-peores-enemigas-chatgpt. [Ultimo Acceso: 12 de Agosto 2023]

[59] LangChain, «Tools - Agents,» [En línea]. Available: https://python.langchain.com/docs/modules/agents/tools/custom_tools#subclassing-the-basetool. [Ultimo Acceso: 14 de Agosto 2023]

[60] MountainGoatSoftware, «Planning poker,» Junio 2023. [En línea]. Available: https://www.mountaingoatsoftware.com/agile/planning-poker. [Ultimo Acceso: 15 de Junio 2023]

[61] NimbleWork, «What is velocity?,» Junio 2023. [En línea]. Available: https://www.nimblework.com/agile/what-is-velocity/. [Ultimo Acceso: 18 de Junio 2023]

[62] Asana, «Smart Goals,» Junio 2023. [En línea]. Available: https://asana.com/es/resources/smart-goals. [Ultimo Acceso: 27 de Junio 2023]

[63] Orienteed, «Aseguramiento de la calidad,» [En línea]. Available: https://orienteed.com/es/aseguramiento-de-la-calidad-software-qa/. [Ultimo Acceso: 07 de Julio 2023]

[64] Teacuplab, «Principios de nielsen,» Junio 2023. [En línea]. Available: https://www.teacuplab.com/es/blog/los-10-principios-heuristicos-de-nielsen-explicados-con-ejemplos/. [Ultimo Acceso: 15 de Junio 2023]

[65] DigitalOcean, «Principios de Programación Orientada a Objetos,» Junio 2023. [En línea]. Available: https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design. [Ultimo Acceso: 19 de Junio 2023]

[66] AprendeInformaticas, Agosto 2023. [En línea]. Available: https://aprendeinformaticas.com/streams/. [Ultimo Acceso: 24 de Agosto 2023]

[67] KeepCoding, «Que es gitflow?,» Junio 2023. [En línea]. Available: https://keepcoding.io/blog/que-es-gitflow/. [Ultimo Acceso: 14 de Junio 2023]

[68] ComisiónEuropea, Marzo 2023. [En línea]. Available: https://commission.europa.eu/law/law-topic/data-protection/data-protection-eu_es. [Ultimo Acceso: 14 de Marzo 2023]

[69] ParlamentoEuropeo, Marzo 2023. [En línea]. Available: https://www.europarl.europa.eu/news/en/headlines/society/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence#:~:text=It%20says%20that%20AI%20systems,world's%20first%20rules%20on%20AI.. [Ultimo Acceso: 16 de Marzo 2023]

[70] SchoolIeb, «5 fuerzas de Porter,» Abril 2023. [En línea]. Available: https://www.iebschool.com/blog/las-5-fuerzas-porter-marketing-digital/. [Ultimo Acceso: 27 de Abril 2023]

[71] Asana, «FODA,» Marzo 2023. [En línea]. Available: https://asana.com/es/resources/swot-analysis. [Ultimo Acceso: 18 de Marzo 2023]

[72] KanbanTool, «Kanban,» Marzo 2023. [En línea]. Available: https://kanbantool.com/es/metodologia-kanban. [Ultimo Acceso: 26 de Marzo 2023]

[73] Atlassian, «Scrum,» Marzo 2023. [En línea]. Available: https://www.atlassian.com/es/agile/scrum#:~:text=%C2%BFQu%C3%A9%20es%20scrum%3F,de%20valores%2C%20principios%20y%20pr%C3%A1cticas.. [Ultimo Acceso: 12 de Marzo 2023]

[74] Atlassian, «Sprints,» [En línea]. Available: https://www.atlassian.com/agile/scrum/sprints. [Ultimo Acceso: 19 de Junio 2023]

[75] OpenAI, «Embeddings,» Abril 2023. [En línea]. Available: https://platform.openai.com/docs/guides/embeddings. [Ultimo Acceso: 22 de Abril 2023]

[76] OpenAI, «Fine-tuning,» Junio 2023. [En línea]. Available: https://platform.openai.com/docs/guides/fine-tuning. [Ultimo Acceso: 15 de Junio 2023]

[77] AWS, «Redes neuronales,» Mayo 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/neural-network/. [Ultimo Acceso: 10 de Mayo 2023]

[78] Microsoft, «Deep Learning,» Abril 2023. [En línea]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/concept-deep-learning-vs-machine-learning?view=azureml-api-2. [Ultimo Acceso: 26 de Abril 2023]

[79] SendSteps, «Diferencias entre GPT3 y GPT4,» Junio 2023. [En línea]. Available: https://www.sendsteps.com/es/blog/gpt-4-frente-a-gpt-3-las-diferencias-que-debe-conocer/. [Ultimo Acceso: 14 de Junio 2023]

[80] LearnPrompting, «Chain of Thought,» Mayo 2023. [En línea]. Available: https://learnprompting.org/es/docs/intermediate/chain_of_thought. [Ultimo Acceso: 08 de Mayo 2023]

[81] MuyAgil, «INVEST,» Junio 2023. [En línea]. Available: https://muyagile.com/escribe-mejor-tus-historias-de-usuario-invest/. [Ultimo Acceso 25 de Junio 2023]

[82] RedHat, «What is Rest API?,» Junio 2023. [En línea]. Available: https://www.redhat.com/es/topics/api/what-is-a-rest-api. [Ultimo Acceso: 21 de Junio 2023]

[83] Grpc, «GRPC,» Junio 2023. [En línea]. Available: https://grpc.io/. [Ultimo Acceso: 02 de Agosto 2023]

[84] LangChain, «LangChain,» Junio 2023. [En línea]. Available: https://python.langchain.com/docs/get_started/introduction. [Ultimo Acceso: 05 de Agosto 2023]

[85] RefactoringGuru, «Singleton,» Julio 2023. [En línea]. Available: https://refactoring.guru/es/design-patterns/singleton. [Ultimo Acceso: 08 de Agosto 2023]

[86] Docker, «Docker,» Agosto 2023. [En línea]. Available: https://www.docker.com/. [Ultimo Acceso: 19 de Agosto 2023]

[87] AWS, «ECR,» Agosto 2023. [En línea]. Available: https://aws.amazon.com/es/ecr/. [Ultimo Acceso: 22 de Agosto 2023]

[88] AWS, «ECS,» Agosto 2023. [En línea]. Available: https://aws.amazon.com/es/ecs/. [Ultimo Acceso: 22 de Agosto 2023]

[89] OpenAI, «ChatGPT: Un modelo de lenguaje generativo de gran escala para la conversación,» 23 Febrero 2023. [En línea]. Available: https://openai.com/blog/chatgpt/. [Ultimo Acceso: 12 de Febrero 2023]

[90] E. Ries, El método Lean Startup, 2011. [Ultimo Acceso: 06 de Junio 2023]

[91] D. Thinking, «Introducción a Design Thinking,» Marzo 2023. [En línea]. Available: https://www.designthinking.es/inicio/index.php. [Ultimo Acceso: 18 de Marzo 2023]

[92] ProductPlan, «Beta Test,» Marzo 2023. [En línea]. Available: https://www.productplan.com/glossary/beta-test/. [Ultimo Acceso: 22 de Marzo 2023]

[93] Arimetrics, «¿Qué es un lead?,» Marzo 2023. [En línea]. Available: https://www.arimetrics.com/glosario-digital/lead. [Ultimo Acceso: 25 de Marzo 2023]

[94] E. Ries, «What Is an MVP?,» Marzo 2023. [En línea]. Available: https://leanstartup.co/resources/articles/what-is-an-mvp/. [Ultimo Acceso: 26 de Marzp 2023]

[95] OpenAI, «ChatGPT: Un modelo de lenguaje generativo de gran escala para la conversación,» Febrero 2023. [En línea]. Available: https://openai.com/blog/chatgpt. [Ultimo Acceso: 03 de Marzo 2023]

[96] A. W. Services, «¿Qué es una base de datos vectorial?,» Marzo 2023 . [En línea]. Available: https://aws.amazon.com/es/what-is/vector-databases/. [Ultimo Acceso: 06 de Abril 2023]

[97] Hubspot, «Análisis Pestel,» Marzo 2023. [En línea]. Available: https://blog.hubspot.es/marketing/crear-analisis-pestel. [Ultimo Acceso: 14 de Marzo 2023]

[98] Hubspot, «Fuerzas de Porter,» Marzo 2023. [En línea]. Available: https://blog.hubspot.es/marketing/fuerzas-de-porter. [Ultimo Acceso: 15 de Marzo 2023]

[99] Hubspot, «Análisis FODA,» Marzo 2023. [En línea]. Available: https://blog.hubspot.es/marketing/analisis-foda. [Ultimo Acceso: 19 de Marzo 2023]

[100] C. Google, «what is artificial intelligence,» Marzo 2023. [En línea]. Available: https://cloud.google.com/learn/what-is-artificial-intelligence?hl=es-419. [Ultimo Acceso: 22 de Marzo 2023]

[101] IBM, «Automatización,» Marzo 2023. [En línea]. Available: https://www.ibm.com/es-es/topics/automation. [Ultimo Acceso: 24 de Marzo 2023]

[102] G. d. Backer, «TAM - SAM - SOM Market,» Abril 2023. [En línea]. Available: https://gustdebacker.com/tam-sam-som-market/.. [Ultimo Acceso: 23 de Abril 2023]

[103] C. Google, «Que es inteligencia artificial?,» Abril 2023. [En línea]. Available: https://cloud.google.com/learn/what-is-artificial-intelligence?hl=es-419. [Ultimo Acceso: 26 de Abril 2023]

[104] OpenAI, «GPT-4,» [En línea]. Available: https://openai.com/gpt-4. [Ultimo Acceso: 26 de Marzo 2023]

[105] Medium, «Integrate Entity Memory,» Junio 2023. [En línea]. Available: https://sonery.medium.com/integrate-entity-memory-into-your-chatgpt-conversations-c8069bb43be6. [Ultimo Acceso: 26 de Junio 2023]

[106] CloudFlare, «Token Based Authentication,» Julio 2023. [En línea]. Available: https://www.cloudflare.com/es-es/learning/access-management/token-based-authentication/. [Ultimo Acceso: 12 de Julio 2023]

13. Anexo

13.1. Análisis del entorno

13.1.1. Análisis del macroentorno utilizando el modelo de PESTEL

Para analizar el macroentorno, se realiza un análisis PESTEL para examinar los factores políticos, económicos, socio-culturales, tecnológicos, ambientales, legales y éticos que afectan a Chat Your Data.

13.1.1.1. Tecnológicos

- **Inteligencia artificial:** El campo de la inteligencia artificial (IA) se encuentra en constante evolución, y se espera que las aplicaciones de la IA sigan creciendo en diferentes industrias. Esta tecnología está revolucionando cómo las empresas procesan y analizan grandes cantidades de datos.
- **Automatización:** Las tecnologías de automatización, como la inteligencia artificial, los *chatbots*, y el internet de las cosas (IoT) se están volviendo cada vez más importantes para los emprendedores que buscan mejorar la eficiencia de sus operaciones y la satisfacción del cliente. Las empresas logran reducir los errores humanos y mejoran la eficiencia, aumentando la productividad y rentabilidad.
- **Nube y servicios de alojamiento:** La creciente adopción de servicios de alojamiento en la nube está permitiendo que las empresas tengan acceso a una variedad de herramientas y recursos tecnológicos a un costo más bajo, lo que puede ayudar a nivelar la competencia para los nuevos emprendimientos. Además, permite la colaboración en tiempo real, generando mayor eficiencia en los proyectos.

13.1.1.2. Sociales y culturales

- **Digitalización de la vida cotidiana:** Las personas están cada vez más conectadas digitalmente, y esto ha llevado a cambios en la forma en que se comunican, trabajan, compran y consumen productos y servicios.
- **Trabajo remoto:** La pandemia ha llevado a un aumento en el trabajo remoto, lo que ha cambiado la forma en que las empresas operan y los empleados trabajan.

- **Cambio demográfico:** La población está envejeciendo y la diversidad en la fuerza laboral está creciendo, lo que puede afectar la forma en que las empresas contratan y gestionan a los empleados.
- **Cambios en la fuerza laboral:** La cultura laboral está evolucionando, con una mayor preferencia por la flexibilidad y el trabajo remoto, lo que está creando nuevas oportunidades y desafíos para los emprendedores.

13.1.1.3. Económicas

- **Condiciones del mercado global**: La pandemia de COVID-19 ha tenido un gran impacto en el mercado global, y ha acelerado la adopción de tecnologías digitales en todas las industrias.
- **Tendencia a la globalización**: La tendencia hacia la globalización también significa que hay una creciente necesidad de herramientas de comunicación y colaboración que permitan a las empresas trabajar de manera eficiente en diferentes países y culturas.
- **Costo creciente de la mano de obra:** El aumento de los costos laborales puede impulsar la adopción de tecnologías de automatización.
- **Demanda en crecimiento de servicio al cliente:** La creciente demanda de un servicio al cliente eficiente y efectivo puede impulsar la implementación de soluciones de *chatbot* y automatización.
- **Aumento de la computación en la nube:** La expansión de la computación en la nube puede influir en la implementación de soluciones basadas en la nube que utilizan tecnologías de inteligencia artificial.

13.1.1.4. Políticos

- **Regulaciones gubernamentales en la industria:** Las regulaciones gubernamentales pueden impactar en la operación de las organizaciones, incluyendo aquellas relacionadas con la privacidad de datos y seguridad**,** como también en el uso de inteligencia artificial.
- **Apoyo gubernamental para la innovación y desarrollo en IA:** La disposición de los gobiernos para respaldar la investigación y el desarrollo en inteligencia artificial es un factor clave, ya que puede influir en la disponibilidad de fondos e incentivos para la innovación.
- **Regulaciones gubernamentales sobre el uso de IA en el trabajo:** Surgen regulaciones que afectan el uso de la inteligencia artificial en el lugar de trabajo, incluyendo temas como la automatización y las políticas de recursos humanos.

13.1.1.5. Ecológicos

- **Mayor atención en la sostenibilidad:** Las empresas están tomando medidas para reducir su impacto ambiental y aumentar su responsabilidad social.
- **Economía circular:** La economía circular es una tendencia ambiental que se enfoca en reducir los residuos y maximizar el uso de los recursos. Las empresas pueden considerar cómo su *software* puede ayudar a las empresas a adoptar prácticas más sostenibles y reducir su impacto ambiental

13.1.1.6. Legales

- **Regulación de protección de datos:** Las regulaciones sobre privacidad y protección de datos, como el Reglamento General de Protección de Datos (RGPD) en la Unión Europea, están teniendo un impacto significativo en la forma en que las empresas pueden recopilar, almacenar y utilizar datos.
- **Regulación de inteligencia artificial:** Algunos países y regiones están implementando regulaciones específicas para la inteligencia artificial, especialmente en áreas como la ética, la transparencia y la responsabilidad. Un ejemplo es la Ley de Inteligencia Artificial (AIA) de la Comisión Europea, que clasifica los sistemas de IA en niveles de riesgo, desde inaceptable hasta mínimo. Las regulaciones se centran en la transparencia, la prevención del sesgo y la discriminación, la privacidad, la seguridad y la importancia del control humano. Estas regulaciones están en desarrollo en varios países y regiones, incluyendo Estados Unidos, China, India y Japón, pero aún se encuentran en sus primeras etapas. Estas regulaciones presentan un gran desafío, ya que se busca equilibrar los beneficios de la IA con los riesgos, pero promoviendo la innovación [68].
- **Regulación de propiedad intelectua**l: Las empresas que utilizan *chatbots* con inteligencia artificial deben considerar las leyes de propiedad intelectual y cómo pueden afectar a su *software* [69].

13.1.2. Análisis del Microentorno

13.1.2.1. Análisis del Micro Entorno utilizando el Modelo de las Cinco Fuerzas de Porter [70]

13.1.2.1.1. Poder de Negociación de Proveedores:

El poder de negociación de los proveedores es moderado. Hoy en día, hay una serie de proveedores para elegir, permitiendo cambiar fácilmente de proveedor si no se está satisfecho con el servicio que se está recibiendo.

Además, los productos y servicios proveedores no son necesariamente únicos o difíciles de sustituir. Se encuentran fácilmente proveedores alternativos que ofrecen productos o servicios similares a precios competitivos.

Sin embargo, hay algunos proveedores que tienen un poder de negociación más alto que otros. Por ejemplo, los proveedores de *hardware* y *software* clave pueden tener más poder de negociación que los proveedores de datos o servicios de nube, ya que de los segundos hay más alternativas que ofrecen el mismo servicio de la misma calidad.

13.1.2.1.2. Rivalidad entre Competidores Existentes:

Hoy en día, hay muchos competidores en el mercado de la inteligencia artificial y en análisis de datos. Sin embargo, se cree que el proyecto cuenta con una propuesta de valor con una ventaja competitiva. Muchos de los competidores en el mercado requieren que los usuarios aprendan un *software* específico, y tengan habilidades técnicas específicas para usar su plataforma de análisis de datos. Además, muchas veces estas soluciones no son muy intuitivas y fáciles de usar, lo que puede generar frustración y menor uso por parte de los usuarios. El emprendimiento se diferencia de sus competidores al ofrecer una solución fácil de usar e innovadora para integrar y analizar datos empresariales de forma centralizada. Además, ofrece una solución a medida para cada organización. Esto implica que las organizaciones contarán con una solución eficiente para procesar volúmenes muy grandes de datos y obtener información valiosa para tomar decisiones.

13.1.2.1.3. Amenaza de Nuevos Jugadores

A medida que el proyecto fue avanzando, los nuevos jugadores fueron siendo cada vez más. A pesar de que existen una serie de barreras de entrada, como la necesidad de experiencia técnica y acceso a capital, los nuevos jugadores en la industria pueden representar una amenaza para los emprendimientos establecidos, ya que pueden aportar nuevas ideas y tecnologías disruptivas. Estos nuevos jugadores pueden ser de diferentes perfiles, como por ejemplo otras *start ups* que se centren exclusivamente en soluciones de integración de datos con un *chat*, incluso con tecnologías más avanzadas o con un enfoque más innovador. También podrían ser empresas de análisis de datos o de automatización de procesos, que ya estén familiarizadas con el tema, y que vean oportunidades en pivotar su modelo de negocios al integrar la inteligencia artificial.

Algunos de ellos son: Data Chat: una plataforma de análisis de datos que utiliza inteligencia artificial y GPT-3 para permitir a sus usuarios hacer consultas en lenguaje natural y analizar datos. ChatThing: combinado con un CRM y ChatGPT, proveen una interfaz de *chat* que permite preguntar y dar instrucciones en lenguaje natural. Chatbotkit: similar agregando integración con web, Discord, Slack. BunkerDB: un asistente de marketing virtual, a través de un *chat* conversacional.

A continuación, se encuentra un análisis de estos nuevos jugadores:

Estos nuevos jugadores fueron apareciendo a medida que el proyecto avanzaba, algunos con diferentes servicios, y otros similares a Chat Your Data. Un dato llamativo es que al momento no se encontró ningún jugador en Latinoamérica. A continuación, se muestran los más interesantes.

Chat Thing

Nombre	Chat Thing		
Planes	***Standard*** • *6 chatbots* • *3 data sources* • *3,000,000 storage tokens*	***Pro*** • *50 chatbots* • *10 data sources per bot* • *10,000,000 storage*	***Enterprise*** • *Unlimited chatbots* • *Unlimited sources per bot data* • *40,000,000 storage*

	49 usd/month	*99 usd/month*	*299 usd/month*
Servicio	Similar a Chat Your Data , con integraciones		
Link	https://chatthing.ai/		
Origen	Reino Unido abril 2023		

Chatbotkit

Nombre	Chatbotkit		
Planes	***Basic*** • *10 datasets* • *10 integrations* • *2M tokens* *25usd/month*	***Pro*** • *100 datasets* • *100 integrations* • *4M tokens* *65usd/month*	***Team*** • *1K Datasets* • *100 integrations* • *20M tokens* *365 usd/month*
Servicio	Chatbots con conexión con Discord, Web, Slack		
Link	https://chatbotkit.com		
Origen	Reino Unido marzo 2023		

Data Chat

Nombre	Data Chat		
Planes	*Free* • *5 concurrent Sessions* • *10M cell Datasets* • *100mb file storage*	*Enterprise* • *Unlimited storage* • *Unlimited sessions*	

Servicio	Enfocado en análisis de datos		
Link	https://datachat.ai		
Origen	Estados Unidos 2023		

13.1.2.1.4. Amenaza de productos o servicios sustitutos

La amenaza de productos y servicios sustitutos es una fuerza importante en cualquier industria, ya que pueden limitar la demanda y los ingresos de un emprendimiento. Los productos y servicios sustitutos pueden incluir otras plataformas de inteligencia artificial o *software* de análisis de datos. Estas plataformas proveen herramientas para analizar e integrar y visualizar datos de empresas. También, servicios de consultoría de datos que ayudan a las empresas a integrar sus datos, otras soluciones de IA y machine learning o métodos tradicionales de análisis de datos, como la hoja de cálculo y la revisión manual de documentos.

A pesar de la existencia de estas alternativas, la amenaza de productos sustitutos es baja. No hay otros productos o servicios que puedan sustituir perfectamente a los *chatbots* con inteligencia artificial. Sin embargo, es necesaria la innovación y la mejora constante para mantenerse competitivos.

13.1.2.1.5. Poder de Negociación de los Clientes

El poder de negociación de los clientes en el mercado se encuentra en crecimiento, debido a el constante ingreso de nuevos jugadores al mercado, lo que le da el poder a los clientes de cambiar fácilmente de producto si no están satisfechos con el servicio que están recibiendo.

13.1.3. Análisis FODA

En este análisis FODA, se evalúan las fortalezas y debilidades internas de Chat Your Data, junto con las oportunidades y amenazas que enfrenta en su entorno competitivo. Este análisis ayuda a tener una visión clara de la posición estratégica de la empresa en la era de la inteligencia artificial y los datos organizacionales [71].

13.1.3.1. Fortalezas:

Propuesta de valor: Chat Your Data se destaca en su industria al ofrecer una solución fácil de usar e innovadora para la integración y análisis de datos organizacionales de manera centralizada. Esto atrae a clientes que buscan soluciones intuitivas y personalizadas.

Adaptación a las tendencias: Chat Your Data está bien posicionada para aprovechar las tendencias tecnológicas y sociales, desde el uso de tecnologías innovadoras como el procesamiento del lenguaje natural, como también la creciente demanda de automatización y la digitalización de la vida cotidiana.

Asociaciones estratégicas: La colaboración con organizaciones reconocidas internacionalmente, como el BID, mejora la imagen de la empresa y puede facilitar futuras inversiones.

13.1.3.2. Oportunidades:

Mercado en evolución: La inteligencia artificial sigue avanzando y más empresas usan tecnología digital, lo que significa que Chat Your Data puede crecer en diferentes industrias.

Financiación disponible: La disponibilidad de capital puede respaldar el desarrollo y la expansión del negocio, especialmente en un entorno donde se valora la innovación tecnológica.

Globalización: La tendencia hacia la globalización crea una creciente necesidad de herramientas de comunicación y colaboración eficientes en diferentes países y culturas, lo que puede hacer cada vez más grande la demanda de soluciones de integración de datos.

13.1.3.3. Debilidades:

Dependencia tecnológica: Chat Your Data depende de proveedores externos de tecnología y servicios para su plataforma, lo que podría afectar negativamente en caso de problemas de los proveedores.

Costos tecnológicos: La implementación de tecnologías avanzadas, como la inteligencia artificial, podría representar un desafío financiero tanto para Chat Your Data como para sus clientes.

13.1.3.4. Amenazas:

Competencia emergente: La entrada de nuevos competidores, especialmente startups con tecnologías avanzadas, podría ser una amenaza para la posición de Chat Your Data en el mercado.

Productos y servicios sustitutos: La disponibilidad de otras plataformas de inteligencia artificial y *software* de análisis de datos podría limitar la demanda de los servicios de Chat Your Data.

Regulaciones cambiantes: Las regulaciones en torno a la inteligencia artificial y la privacidad de datos están en constante cambio, lo que podría requerir ajustes en la operación de la empresa.

Ética y responsabilidad: Las preocupaciones éticas en torno a la inteligencia artificial y la responsabilidad en caso de errores pueden plantear desafíos legales y de reputación.

13.2. Modelos de Suscripción

13.2.1. Características de los Planes

Para cada plan, se ofrecerán las siguientes características:

Se explican estos conceptos para comprender la tabla.

Storage tokens: Los *tokens* de almacenamiento se utilizan cuando se suben documentos a la base de datos vectorial. 1000 *tokens* de almacenamiento equivalen aproximadamente a unas 750 palabras. Cada vez que suba un nuevo documento, utilizará más *tokens* de almacenamiento.

Interaction tokens: Los *tokens* de mensaje se utilizan cada vez que se envía un *prompt* a un *bot* y se recibe una *completion*. Cada mensaje utiliza entre 1000 y 4000 *tokens* de mensaje según las preguntas y la respuesta de los *bots*. Esto significa que 200.000 *tokens* de mensajes equivalen aproximadamente a 100 mensajes.

	Cantidad de usuarios base	Cantidad de storage *tokens*	Cantidad de interaction *tokens*	Cantidad de *chatbots*	*Features*	Modelo de IA
Free (personal)	1	3.000.000	2.000.000	1	En tabla *Features*	GPT-3.5
Standard (personal y organizaciones chicas)	10	10.000.000	10.000.000	1 por usuario	En tabla Features	GPT-3.5
Pro (organizaciones > 50 empleados)	50	Ilimitado	Ilimitado	1 por usuario	En tabla Features	Híbrido (GPT-4 para el output, GPT-3.5 para el resto)
Enterprise (organizaciones > 100 empleados)	Ilimitado	Ilimitado	Ilimitado	Ilimitado	Todas (en tabla Features)	GPT-4

Tabla 11- Características de los planes

13.2.2. *Features*

	Free	*Standard*	*Pro*	*Enterprise*
Compartir conversación	X	X	X	X
Guardar conversaciones		X	X	X
Ver referencias		X	X	X
Chats multipersonales			X	X
Búsqueda de Google embebida (SerpApi)			X	X
Resumen de textos grandes			X	X
Tipos de archivos	PDF	PDF y CSV	PDF y CSV	Todos
Soporte en línea personalizado				X
Integración con herramientas externas (GCalendar, Slack, Teams, LinkedIn)				X

Tabla 12- Tabla de *features* del modelo de suscripción

13.3. Metodologías Agiles

13.3.1. Características de Kanban

- **Flexibilidad en la definición de tareas:** Kanban brinda la posibilidad de definir tareas sin la rigidez de establecer tiempos de entrega. Esta característica se adapta de manera óptima a contextos cambiantes y permite una gestión ágil de las tareas a medida que surgen.
- **Implementación sencilla:** La simplicidad inherente a Kanban es un factor determinante. Requiere una mínima preparación u organización previa, lo que facilita su adopción sin imponer barreras significativas al equipo.
- **Visualización del flujo de trabajo:** La visualización del flujo de trabajo es un punto clave que Kanban aborda de manera efectiva. Al proporcionar una representación gráfica de las tareas en diferentes etapas, brinda claridad y transparencia sobre el progreso y las áreas de enfoque.

- **Fomento de la mejora continua:** Al proporcionar una visión clara del proceso y el flujo de trabajo, se facilita la identificación de áreas de optimización y refinamiento constante [72].

13.3.2. Elementos de Kanban

13.3.2.1. Kanban board

Inicialmente, se llevó a cabo una sesión de lluvia de ideas con el objetivo de identificar todas las tareas y temas pertinentes en el proyecto. Dado el carácter desafiante y novedoso del tema, muchas de estas tareas se relacionaban con la investigación, pruebas y revisión de cursos, entre otros aspectos. Una vez finalizada la lluvia de ideas, las tareas fueron ubicadas en la columna *To do* para que cada miembro del equipo pudiera seleccionar aquellas en las que empezar a trabajar. En casos particulares, cuando se trataba de tareas extensas, se asignó más de un miembro del equipo para colaborar en conjunto. Además, se buscó no tener un exceso de tareas en progreso al mismo tiempo, permitiendo mantener un flujo constante y eficiente.

Asimismo, todos los integrantes del equipo tenían la capacidad de agregar nuevas "tarjetas" durante la fase de investigación, dado que en muchas ocasiones surgían conceptos nuevos que requerían comprensión posterior. Para mantener un control efectivo de las tareas, se llevaban a cabo reuniones tres veces por semana, donde se compartían actualizaciones sobre la investigación y se realizaba un seguimiento del tablero.

La gestión de estas tareas y su estado se realizaba a través de un tablero que incluía 3 columnas representativas de los distintos estados de las tareas, siendo estas *To do, In Progress* y *Done*. Este tablero fue implementado utilizando la herramienta Notion, la cual proporcionó la flexibilidad necesaria para editar, asignar responsables, añadir etiquetas temáticas y aprovechar otras funciones útiles para la gestión integral del proyecto.

13.3.2.2. Columnas

To Do (Por hacer): Esta columna incluye las tareas que aún no han sido iniciadas y, en su mayoría, no han sido asignadas a ningún miembro del equipo. Esto permite que los integrantes del equipo puedan acceder a estas tareas una vez que hayan finalizado las que estaban trabajando.

In Progress (En progreso): En esta columna se encuentran las tareas en las que el equipo está trabajando actualmente. Cada tarea está asignada a un miembro específico del equipo, lo que permite una clara identificación de quién está trabajando en cada tarea. Esta columna es fundamental para visualizar el progreso del proyecto y asegurarse de que todas las tareas estén en curso de manera adecuada.

Done (Completado): Esta columna muestra las tareas que han sido finalizadas y completadas exitosamente. Una vez que una tarea ha sido completada, se mueve a esta columna, lo que permite al equipo tener una visión clara de las tareas que se han concluido.

13.3.3. Características de Scrum

- **Iteraciones cortas**: Dada la complejidad y volatilidad del contexto, la implementación de *sprints* de corta duración es una práctica fundamental. Esta práctica favorece una comunicación continua, proporcionando un canal directo para abordar desafíos y, sobre todo, para mitigar posibles riesgos.
- **Equipos auto-organizados**: Se busca fomentar la auto-organización del equipo, permitiendo la libertad de estructurar el enfoque y elegir las tareas que resulten más interesantes. Esta autonomía impulsa un compromiso más profundo y una mayor motivación en el equipo.
- **Comunicación constante**: Se promueve la colaboración activa y la conexión entre los miembros del equipo, así como con las partes interesadas. La comunicación continua lleva a una relación efectiva y fluida, contribuyendo a un entendimiento más sólido y a una alineación constante con los objetivos del proyecto [73].

13.3.3.1. Sprint

Son eventos de duración fija de un mes o menos para crear consistencia (en el caso del proyecto se decidió que fuesen de 2 semanas). Cada nuevo *sprint* comienza inmediatamente después de que finaliza el anterior. Durante un *sprint*, se realiza todo el trabajo necesario para alcanzar el Objetivo del Producto, incluyendo la planificación del *sprint*, las reuniones diarias de seguimiento, la revisión del *sprint* y la retrospectiva del *sprint*. Esta estructura asegura la inspección y adaptación periódica del progreso hacia el Objetivo del Producto al menos una vez al mes [74].

13.3.4. Designación de roles

Rol	Descripción	Responsable
Ingeniero en requerimientos	Encargado de identificar y documentar los requerimientos y comprender necesidades del producto, utilizando como input los comentarios del *beta tester*.	Paula
Encargado de Gestión	Responsable de la coordinación y planificación del proyecto, asignar tareas y facilitar la comunicación del equipo.	Noa
Arquitecto y encargado en infraestructura	Responsable de diseñar la Arquitectura del proyecto, establecer la infraestructura y asegurar la integración y la escalabilidad del proyecto	Ignacio
Encargado de aseguramiento de la calidad	Encargado de garantizar la calidad del producto final, realizar pruebas para identificar errores, asegurar que cumpla con los estándares de calidad establecidos.	Matías

Tabla 13 - Designación de Roles

13.4. Evolución de la inteligencia artificial

La inteligencia artificial comienza a inicios de la década de 1940, cuando el matemático y filósofo británico Alan Turing propuso la idea de una "máquina universal" que pudiera procesar información. Esto sentó las bases de la computación moderna y estableció los fundamentos para la IA.

A continuación, se desarrollan algunos de los hitos más importantes:

- 1950-1970: Desarrollo de métodos y algoritmos de IA, incluyendo el Perceptron, una red neuronal artificial capaz de reconocer patrones en imágenes y textos. Se creo ELIZA, un

programa que simulaba una terapeuta y podía mantener conversaciones simples con usuarios. ELIZA se considera uno de los primeros *chatbots* de la historia.

- 1970-1990: Los sistemas de procesamiento de lenguaje natural (NLP) se convirtieron en un campo de investigación, con el objetivo de permitir que las computadoras pudieran comprender el lenguaje humano. Se desarrollaron los algoritmos de aprendizaje profundo que permiten entrenar redes neuronales con varias capas para reconocer patrones complejos en imágenes, sonidos y texto. Estos algoritmos fueron los que dieron lugar a las redes neuronales profundas, que son fundamentales en muchas de las aplicaciones de IA modernas.
- 1990 - 2000: De la mano con el desarrollo de la computación moderna, y la creación de redes neuronales, la IA comenzó a ganar en juegos más complejos, como el ajedrez y el *Go*.
- 2000 - 2010: IBM presentó Watson, un sistema de IA capaz de responder preguntas en lenguaje natural, demostrando que la IA podía manejar grandes cantidades de datos no estructurados y responder preguntas de manera precisa. Google, quien se convierte en uno de los pioneros en IA crea DeepMind en donde desarrollo un sistema que aprendió a jugar videojuegos de la consola Atari 2600 sin conocimiento previo del juego.
- 2010 - Actualidad: Se comienza con la "cuarta revolución industrial", en donde la IA forma parte. Esto se observa con la llegada de "Siri", "Google Now" y "Cortana" como asistentes. La IA tiene enormes avances en el desarrollo de sistemas basados en el aprendizaje profundo o *deep learning*. Elon Musk y otros anuncian una inversión de un billón de dólares en Open AI. Se crea la arquitectura Transformer. Lanzamiento de ChatGPT.

[29]

13.5. Arquitectura Transformer

13.5.1. Tokenización

La tokenización es el proceso de descomponer el texto en componentes más pequeños, denominados *tokens*. En el contexto de los modelos lingüísticos de IA, estos *tokens* suelen corresponder a palabras o subpalabras. Los *tokens* pueden ser tan breves como un carácter o tan extensos como una palabra, e incluso más largos en idiomas sin espacios.

Existen varios tipos de tokenización, entre los que se incluyen:

1. **Tokenización de palabras**: Este es el método más básico, en el que el texto se divide en palabras individuales. Por ejemplo, la frase "I love AI" se dividiría en ["I", "love", "AI"].
2. **Tokenización de subpalabras:** Este método descompone las palabras en componentes semánticos más pequeños, o subpalabras. Resulta especialmente útil para tratar palabras fuera del vocabulario y para lenguas en que las palabras pueden formarse a partir de raíces más pequeñas. Por ejemplo, la palabra "unhappiness" se puede dividir en ["un", "happiness"].
3. **Tokenización basada en byte-pair encoding (BPE)**: Este es un método híbrido que inicialmente divide el texto en caracteres y luego combina los caracteres o secuencias de caracteres más comunes en un solo *token*. Este método es muy útil para lidiar con palabras raras o desconocidas en la fase de entrenamiento.

Una vez que los *tokens* son generados, cada uno de ellos se mapea a un vector en un espacio de mayor dimensión. Este proceso se conoce como *embedding*.

13.5.2. *Embeddings*

El proceso de *embedding* actúa como un traductor, convirtiendo *tokens* a un espacio vectorial de mayor dimensión y, por ende, permitiendo que el modelo de IA aprenda y represente relaciones complejas entre diferentes palabras. Este espacio, no solo representa la semántica de las palabras, sino que también facilita al modelo comprender similitudes y relaciones entre palabras desde un punto de vista gramatical y contextual. En este espacio, cada palabra o *token* se transfigura en un vector único [75].

Una representación n-dimensional de palabras

Los *embeddings* crean una representación n-dimensional del texto, estableciendo un vínculo crucial entre la comprensión humana y la computacional del lenguaje. En este entorno, las palabras que comparten similitudes semánticas tienden a posicionarse cerca unas de otras, fomentando así un mapeo semántico coherente.

Conversión a vectores

Posterior a la *tokenización*, cada *token* se convierte en un vector. Inicialmente, estos vectores pueden generarse de manera aleatoria y son refinados durante el preentrenamiento de modelos, como GPT-4, al aprender a predecir el siguiente *token* en una secuencia. A medida que el

modelo procesa vastas cantidades de texto, estos vectores se optimizan para encapsular de manera precisa las relaciones semánticas y sintácticas entre palabras.

En el ámbito del procesamiento del lenguaje natural (*NLP*), esta conversión de palabras a representaciones numéricas es imperativa para permitir que modelos de aprendizaje automático, como *GPT-4*, puedan interpretar y generar texto de manera coherente.

13.5.3. Atención (contextualización)

La capacidad de enfocar de manera diferencial diversas partes de la información de entrada es clave en la arquitectura Transformer, y esto es realizado mediante el mecanismo de "Atención". Este concepto se centra en permitir que el modelo tenga una percepción precisa del significado de las palabras, no solamente basada en ellas mismas, sino también considerando el contexto proporcionado por las palabras circundantes.

Para ilustrarlo con un ejemplo, la palabra "banco" en las frases "Voy al banco a sacar dinero" y "Estoy sentado en el banco de la plaza" tiene significados distintos: una institución financiera en el primer caso y un objeto físico para sentarse en el segundo. El mecanismo de atención es capaz de discernir entre estos diferentes significados mediante la ponderación del contexto proporcionado por las palabras adyacentes, permitiendo una interpretación adecuada en diferentes situaciones.

Un vistazo detallado al mecanismo de atención

El mecanismo de atención funciona asignando diferentes pesos a las palabras en una secuencia. Así, al procesar una palabra específica, el modelo puede "enfocarse" más en aquellas palabras circundantes que son más relevantes para determinar su significado en el contexto particular. Este cálculo de pesos se realiza utilizando las representaciones vectoriales (*embeddings*) de las palabras, ayudando al modelo a formar una interpretación contextualizada y coherente de la palabra o frase en cuestión.

Por ejemplo, en la oración "El gato cazó al ratón", al interpretar la palabra "cazó", el mecanismo de atención facilita que el modelo otorgue una importancia relevante a las palabras "gato" y "ratón", comprendiendo la dinámica entre los tres *tokens* y logrando una interpretación precisa de los actores y las acciones involucradas.

Implementación en la Arquitectura Transformer

La atención se implementa en la arquitectura Transformer a través de lo que se conoce como "cabezas de atención múltiple". Esto permite al modelo no solo entender las relaciones entre diferentes palabras en una secuencia, sino también capturar diferentes tipos de relaciones y dependencias al mismo tiempo. Cada cabeza de atención tiene la capacidad de enfocarse en distintos aspectos del texto, por ejemplo, la semántica o la sintaxis, proporcionando así una interpretación del lenguaje rica y diversa.

La habilidad de los modelos Transformer para gestionar estas relaciones contextuales complejas y variadas es uno de los factores clave que ha llevado a la amplia adopción y éxito de esta arquitectura en numerosas aplicaciones de procesamiento del lenguaje natural (NLP).

Este mecanismo de atención múltiple es crucial para que los modelos Transformer puedan generar respuestas relevantes y coherentes, ya que proporciona una forma de considerar diversas perspectivas o interpretaciones de una entrada de texto simultáneamente.

En la sección 13.9 Arquitectura Transformer se puede encontrar un ejemplo de cómo funciona.

13.5.4. Predicción

La predicción juega un papel central, habilitando al modelo para anticipar de manera coherente y contextual la palabra o *token* subsiguiente en una secuencia lingüística. Este componente es crucial tanto en la generación de texto, como en la traducción automática, y otras aplicaciones que requieren de una generación de secuencias de salida lógica y contextualizadas.

Aprendizaje de secuencias y mapeo de entrada/salida

En una aproximación más detallada, durante el entrenamiento, los modelos Transformer aprenden a mapear secuencias de entrada (por ejemplo, una frase) a secuencias de salida correspondientes (tal como una traducción de la frase), haciendo mucho más que establecer correspondencias simples entre palabras. De esta manera, el modelo asimila y entiende las relaciones entre las palabras en las secuencias de entrada y salida para generar mapeos significativos.

Decodificación autorregresiva en la predicción

En modelos como el *GPT*, un método denominado "decodificación autoregresiva" es empleado en el proceso de predicción. Aquí, una vez que una palabra es predicha, esta se reincorpora a la secuencia de entrada para ayudar a predecir las palabras siguientes. En esencia, las predicciones previas se utilizan como contexto para las futuras, asegurando que las secuencias generadas mantengan una coherencia y relevancia textual.

13.6. Fine-Tuning

El proceso de "fine-tuning" (ajuste fino) en el contexto de LLMs, involucra una adaptación específica del modelo pre-entrenado a una tarea particular o a un dominio específico, utilizando un conjunto de datos adicional que es más preciso o pertinente para la tarea en cuestión.

1. Pre-entrenamiento del modelo:

Antes del fine-tuning, un LLM es típicamente pre-entrenado en un gran corpus de texto. Durante este pre-entrenamiento, el modelo aprende a entender y generar texto coherente, así como a capturar varias relaciones semánticas y sintácticas entre palabras y frases, al predecir palabras faltantes en las oraciones, por ejemplo.

2. Creación del dataset específico de tarea:

El *dataset* se compone de pares de preguntas y respuestas, en los que las preguntas sirven como entrada y las respuestas correspondientes como la salida deseada. Este conjunto de datos específico se emplea para refinar las capacidades del LLM en el contexto de la tarea de generación de respuestas.

3. *Fine-Tuning:*

- Inicialización del modelo: El LLM, ya pre-entrenado, es inicializado con los pesos aprendidos durante el pre-entrenamiento. Estos pesos contienen el conocimiento del lenguaje adquirido durante esa fase inicial.

- Adaptación al dataset específico: Se ajustan los pesos del LLM utilizando el *dataset* de preguntas y respuestas mediante aprendizaje supervisado. En cada iteración, el modelo predice una respuesta basándose en la pregunta dada, luego se compara la predicción del modelo con la respuesta real, y se usa esta diferencia (el error) para actualizar los pesos del modelo mediante retropropagación.

- Optimización: Este proceso se repite, optimizando iterativamente los pesos del modelo para minimizar el error en las predicciones del LLM en el *dataset* de tareas específicas.

Evaluación y Ajuste:

Después de la fase de *fine-tuning*, el modelo es evaluado para asegurar que las respuestas generadas son coherentes y relevantes a las preguntas proporcionadas. Se pueden realizar ajustes adicionales basados en las necesidades específicas del dominio o en las métricas de rendimiento deseadas.

Este proceso de *fine-tuning* es crucial para adaptar modelos de lenguaje preentrenados, que tienen un conocimiento del lenguaje general, a tareas específicas o dominios, permitiéndoles proporcionar respuestas y soluciones más relevantes y precisas [76].

13.7. Redes Neuronales

Componentes y estructura

- Neurona: Elemento fundamental que realiza cálculos sencillos sobre los datos de entrada y pasa el resultado a la siguiente capa a través de conexiones ponderadas.

- Pesos: Valores numéricos que modulan las conexiones entre neuronas. Se ajustan durante el entrenamiento para mejorar las predicciones del modelo.

- Función de activación: Transforma la entrada acumulativa de una neurona antes de pasarla a la siguiente capa. Ayuda a la red a aprender patrones complejos y no lineales.

Capas de Neuronas:

- Capa de Entrada: Recibe los datos iniciales para el procesamiento.
- Capas Ocultas: Realizan la mayoría de los cálculos necesarios para el aprendizaje.
- Capa de Salida: Produce la predicción o clasificación final del modelo.

Proceso de aprendizaje

- Entrenamiento: Las redes neuronales aprenden ajustando los pesos de las conexiones en base a la diferencia entre la predicción generada y los valores reales, utilizando algoritmos como el Descenso del Gradiente y técnicas como la Retropropagación del Error.

- Predicción: Una vez entrenada, la red neuronal puede hacer predicciones al procesar nuevas entradas a través de la red, propagando los datos a través de las capas y produciendo una salida.

Aplicaciones prácticas

- Clasificación de imágenes: Identificación y categorización de objetos dentro de imágenes.

- Reconocimiento de voz: Interpretación y transcripción de datos auditivos.

- Análisis de texto: Comprender y generar respuestas o clasificaciones basadas en datos textuales.

[77]

13.8. Deep Learning

- Estructura: Similar a las redes neuronales, las profundas también se estructuran en capas de neuronas, pero en este caso, hay un número significativo de capas ocultas entre las capas de entrada y salida.

- Capacidad: El uso de múltiples capas ocultas permite a las redes neuronales profundas aprender representaciones de datos más complejas y abstractas.

Elementos clave

- Aprendizaje jerárquico: Las redes neuronales profundas aprenden de manera jerarquizada, con neuronas en las capas iniciales aprendiendo características simples y neuronas en capas más profundas construyendo representaciones más complejas.

- Aprendizaje autónomo de características: Reducen la necesidad de ingeniería manual de características, ya que son capaces de aprender características relevantes directamente de los datos.

Métodos y modelos notables

- Redes neuronales convolucionales (CNNs): Especialmente robustas en el procesamiento de datos visuales, como imágenes.

- Redes neuronales recurrentes (RNNs): Destacan en el manejo de secuencias de datos, como series temporales o secuencias lingüísticas.

- Autoencoders: Útiles en la compresión de datos y la reducción de la dimensionalidad.

Aplicaciones

- Visión por computadora: Desde el reconocimiento facial hasta la clasificación de imágenes.

- Procesamiento del lenguaje natural: Comprende desde la traducción automática hasta la generación de texto.

- Reconocimiento de voz: Facilitando la interacción humano-computadora mediante el habla.

- Juegos y Simulaciones: Desarrollando agentes que pueden aprender estrategias complejas a través del juego.

Desafíos y Consideraciones:

- Necesidad de datos: Los modelos de *deep learning* requieren grandes volúmenes de datos para entrenarse eficazmente.

- Computacionalmente intensivo: Requieren *hardware* especializado y son más exigentes en términos de recursos computacionales.

- Interpretabilidad: Los modelos a menudo son criticados por ser "cajas negras", dada la dificultad para interpretar cómo hacen sus predicciones.

[78]

13.9. Arquitectura Transformer

Ejemplo de cómo funciona la atención dentro de la arquitectura Transformer

Para comprender aún más cómo funciona la Atención dentro de la arquitectura Transformer se puede ver el siguiente ejemplo: "El león, feroz y majestuoso, cazó a la gacela en la sabana africana."

Cuando la arquitectura *Transformer* procesa esta oración, lo hace utilizando varias "cabezas de atención" simultáneamente, y cada una de ellas puede enfocarse en diferentes aspectos o relaciones dentro de la oración.

- Cabeza de Atención 1: Podría enfocarse en la relación sujeto-verbo-objeto para entender la acción principal en la oración. En este caso, podría asociar "El león" con "cazó" y "a la gacela" para formar una comprensión básica de la acción que se está llevando a cabo.

- Cabeza de Atención 2: Podría centrarse en las descripciones y adjetivos de los sustantivos para obtener una comprensión más rica del contexto. Aquí, podría asociar "feroz y majestuoso" con "El león" para formar una imagen más detallada del sujeto que está realizando la acción.

- Cabeza de Atención 3: Podría estar atenta a la información de ubicación o contexto ambiental contenida en la oración. Podría vincular "en la sabana africana" con la acción de cazar para entender el escenario donde ocurre la acción.

- Cabeza de Atención 4: Podría centrarse en las relaciones gramaticales, identificando, por ejemplo, que "feroz y majestuoso" están trabajando como adjetivos para "El león" y "africana" para "sabana", garantizando que el modelo tenga una comprensión gramatical clara de la oración.

Cada una de estas "cabezas" procesa la oración de manera simultánea y atiende a diferentes aspectos de la información contenida en ella. Posteriormente, las representaciones de todas las cabezas se combinan y se procesan en las capas subsecuentes de la red para formar una comprensión cohesiva y rica de la oración completa.

13.10. Diferencias de GPT-3 y GPT-4

Principales diferencias entre GPT-3 y GPT-4

- Tamaño: GPT-4 tiene un tamaño significativamente mayor que GPT-3. Esto significa que puede aprender y retener más información.
- Precisión: GPT-4 es más preciso que GPT-3. Esto significa que es menos probable que genere texto que sea sesgado o inexacto.
- Adaptabilidad: GPT-4 es más adaptable que GPT-3. Esto significa que puede ser ajustado con mayor facilidad para satisfacer las necesidades específicas de una aplicación o usuario.

Habiendo entendido los conceptos básicos de inteligencia artificial, se comenzó a profundizar en cómo se podría llevar a cabo un *chatbot* con IA, explorando conceptos específicos y herramientas necesarias para su desarrollo. [79]

13.11. Diferencia entre Standard Prompting y Chain of Thought Prompting

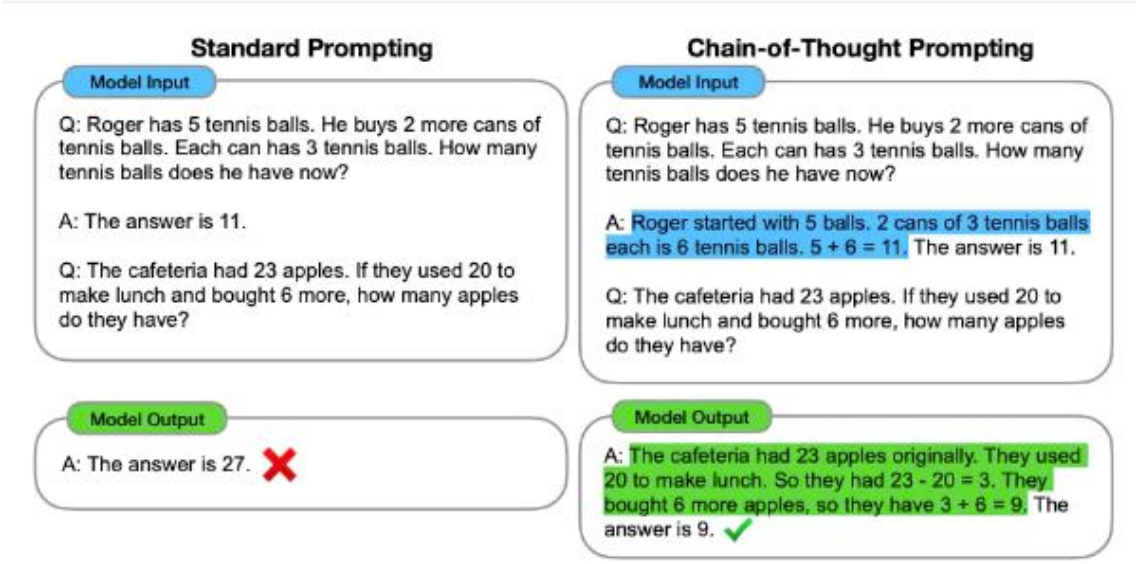


Ilustración 65 - Diferencia entre Standard Prompting y Chain of Thought Prompting

Extraída de LearnPrompting.org [80]

13.12. Mecanismo de funcionamiento de Similarity Search en Bases de Datos Vectoriales

Paso 1: Representación Vectorial

Los elementos en la base de datos se traducen en vectores numéricos mediante un proceso conocido como *embedding*. Esto podría implicar convertir palabras, documentos o imágenes en una forma en que estén representados por puntos en un espacio multidimensional.

Paso 2: Definición de Similitud

Se elige una métrica de similitud o distancia de acuerdo a la aplicación. Métricas comunes incluyen:

- Distancia Euclidiana: Mide la distancia en línea recta entre dos puntos en el espacio euclidiano.
- Similitud Coseno: Mide el coseno del ángulo entre dos vectores, a menudo utilizado para datos de texto.
- Distancia Manhattan: La distancia entre dos puntos medida a lo largo de ejes en ángulos rectos.

Paso 3: Vector de Consulta

Cuando se realiza una consulta, el elemento de interés también se convierte en un vector utilizando el mismo proceso de *embedding*. Este vector se utiliza luego para buscar en la base de datos de vectores y encontrar elementos similares.

Paso 4: Algoritmo de Búsqueda

Un algoritmo de búsqueda navega a través de la base de datos para identificar los vectores que son más similares al vector de consulta. Algoritmos de búsqueda eficientes, como árboles k-d, árboles de bola, o Vecinos Más Cercanos Aproximados (ANN por sus siglas en inglés), pueden ser empleados para mejorar la velocidad y eficiencia de la búsqueda, especialmente en grandes bases de datos.

13.13. Otros conceptos claves

Firebase:

https://firebase.google.com/docs/auth?hl=es-419
https://firebase.google.com/docs/storage?hl=es-419

Firebase es una plataforma en la nube desarrollada por Google para el desarrollo de aplicaciones móviles y web. Firebase ofrece una variedad de servicios que pueden ayudar a los desarrolladores a crear aplicaciones más rápidamente y fácilmente, incluyendo:

- Base de datos: Firebase Realtime Database y Firebase Firestore, base de datos NoSQL de tiempo real que permite a los desarrolladores almacenar datos de forma segura y escalable.
- Autenticación: Firebase Authentication proporciona una forma sencilla de autenticar a los usuarios de una aplicación.
- Notificaciones: Firebase Cloud Messaging permite a los desarrolladores enviar *push notifications* a los usuarios de sus aplicaciones.

- Almacenamiento: Firebase Cloud Storage proporciona un almacenamiento seguro y escalable para los archivos de una aplicación.

Redis

https://redis.io/

Redis es un sistema de gestión de bases de datos en memoria que ofrece un conjunto de estructuras de datos flexibles y potentes. Es una opción popular para aplicaciones que requieren alta velocidad, escalabilidad y disponibilidad.

Es una base de datos NoSQL, lo que significa que no utiliza un modelo de datos relacional tradicional. En cambio, Redis utiliza un modelo de datos flexible que admite una variedad de estructuras de datos, como cadenas, *hash*, listas, conjuntos, conjuntos ordenados, bitmaps, *hyperloglogs*, índices geoespaciales y flujos.

13.14. Entrevistas

13.14.1. Preguntas de las entrevistas

¿En qué tipo de organización trabajas y cuáles son tus responsabilidades?
¿Cómo suele ser el proceso de búsqueda y acceso a la información dentro de tu organización?
¿Qué problemas o dificultades has encontrado en el proceso de búsqueda y acceso a la información dentro de tu organización?
¿Cuánto tiempo te lleva encontrar la información que necesitas?
¿Hay alguna información que te resulta especialmente difícil de encontrar? ¿Qué tipo de información es?
¿Cuánto tiempo pasas buscando información en una semana típica de trabajo?
¿En qué tipo de plataformas o herramientas buscas información? (por ejemplo, Google, Drive, Dropbox, CRM, ERP, etc.)
¿Has utilizado alguna solución de chat IA anteriormente? ¿Cómo fue tu experiencia?
¿Estás familiarizado con la tecnología de chat IA personalizada?
¿Qué características o funcionalidades te gustaría ver en una plataforma de chat IA personalizada para organizaciones?
¿Qué impacto crees que tendría una plataforma de chat IA personalizada en la productividad y eficiencia de tu organización?

¿Qué preocupaciones o desafíos tendrías al implementar una plataforma de chat IA personalizada en tu organización?
¿Estarías dispuesto a pagar por un chat con inteligencia artificial para hacer consultas de los datos de tu organización?
Recomendaciones, preguntas

13.14.2. Transcripción de entrevista a Mayda Kurdian

Mayda Kurdian, 59 años, empresa K2B:

¿En qué tipo de organización trabajas y cuáles son tus responsabilidades?

Trabajo en la empresa K2B que construye e implementa el producto K2B. K2B es un ERP para medianas y grandes empresas, presente en gran parte de Latinoamérica. Soy la R&D manager del producto, liderando un equipo de 13 ingenieros y técnicos en informática.

También soy *Product Manager* del Equipo K2BTools compuesto por 3 ingenieros que desarrolla herramientas para acelerar el desarrollo con GeneXus.

¿Cómo suele ser el proceso de búsqueda y acceso a la información dentro de tu empresa?

Es un proceso muy ineficiente y engorroso. Manejamos una cantidad importante de información y de diferentes tipos. Algunos ejemplos: desde especificaciones de nuevas funcionalidades para el producto, documentación de las capacidades del producto en cada una de sus versiones, datos de clientes para saber que versiones tienen instalado y cómo usan el producto, datos de *issues*, comunicaciones dentro del equipo, datos del equipo como licencias, etc. Tenemos diferentes sistemas o soporte para cada tipo de información, y el proceso de saber dónde está esa información y acceder a ella es engorroso, consume tiempo e incrementa la carga cognitiva de las personas. Además, somos un equipo horizontal y generalista, por lo que casi todos deben acceder a casi todo. Esto profundiza más aún el problema

¿Qué problemas o dificultades has encontrado en el proceso de búsqueda y acceso a la información dentro de tu empresa? ¿Cuánto tiempo te lleva encontrar la información que necesitas? ¿Hay alguna información que te resulta especialmente difícil de encontrar? ¿Qué tipo de información es?

Parte de la respuesta fue contestada en la pregunta anterior. A mí en particular que debo acceder a toda la información, que tomo decisiones basada en ella, que tengo que consolidarla y

presentarla a otros, me lleva bastante tiempo. No tengo medido el tiempo, pero teniendo en cuenta todo lo anterior podría decir que eso me lleva casi un día entero de trabajo a la semana.

¿En qué tipo de plataformas o herramientas buscas información? (por ejemplo, Google, Drive, Dropbox, CRM, ERP, etc.)

CRM, ERP, Sistema de *issue tracking*, WIKI con toda la documentación del producto, Google Drive (docs, slides, sheets), Excel, Google Chat, Sistema de Proyectos interno, Sistema de RRHH.

¿Has utilizado alguna solución de chat IA anteriormente? ¿Cómo fue tu experiencia?

No hemos utilizado soluciones de chat IA aún, pero si he usado como usuaria de diferentes sitios, y la experiencia hasta ahora ha sido mala. Nunca he encontrado solución a mis problemas o respuestas a mis preguntas en esos chats. En muchos de ellos me he encontrado en un *loop* constante.

¿Estás familiarizado con la tecnología de chat IA personalizada?

He comenzado a interiorizarme con la *tech* de chats personalizados basados en IA desde principios del 2022 y a *full* desde la liberación de ChatGPT.

¿Qué características o funcionalidades te gustaría ver en una plataforma de chat IA personalizada para empresas?

Preguntarle lo que sea de mi organización y me responda teniendo en cuenta toda la documentación de mi organización. ¡Eso sería grandioso!

Sería interesante si me pudiera dar alguna medida de la precisión de la respuesta, de ser posible. También, siendo menos ambiciosa que me responda con algún *link* a un formulario de un sistema con filtros instanciados o documento donde pueda encontrar la información.

También que, dada cierta información específica, ejemplo si tengo en una planilla los *issues* de los últimos meses, con todas sus características: *issue*, cliente, fecha, estado, etc., pudiera hacerle preguntas sobre esa información para analizarla y tomar decisiones.

Ejemplos: clientes que reportaron más *issues*, comparativos de diferentes periodos, tiempos medios de resolución, pendientes y corregidos. Cliente que reporta más, etc.

O sea determinada una fuente de datos específica, que el chat me permita analizar esa información, encontrar patrones, problemas y por qué no, posibles soluciones o acciones.
También generar un resumen de esa información y crear una presentación con esto.

¿Qué impacto crees que tendría una plataforma de chat IA personalizada en la productividad y eficiencia de tu empresa?

¡Enorme! No sólo eficiencia, sino que mejoraría enormemente la calidad de las decisiones.
Mejoraría desde los procesos internos, la atención a los clientes en forma mucho más personalizada y con información más completa y confiable. Además, no solo me gustaría todo lo dicho para mi organización sino para nuestro producto. El ERP tiene una cantidad de información de la organización y cada una de ellas requiere diferentes reportes e información, porque las empresas donde implantamos son de diferentes rubros, pero además cada una configura y personaliza el ERP de diferentes formas. La personalización de qué información y preguntas son importantes para cada una es algo que tiene un importante costo y no es muy reutilizable.

¿Qué preocupaciones o desafíos tendrías al implementar una plataforma de chat IA personalizada en tu empresa?

Quizás sobre la seguridad y confidencialidad de cierta información.

¿Estarías dispuesto a pagar por un chat con inteligencia artificial para hacer consultas de los datos de tu empresa?

Dado que somos una empresa de tecnología y ya estamos experimentando con IA no solo para usarla en la organización sino para incluirla en nuestros productos, creo que la respuesta es sí.

A nivel técnico les puedo decir lo que aprendí y experimenté hasta ahora, desde mi corta experiencia en esto:

- que el *prompting* es clave y que se aprende haciendo/probando,
como siempre, pero en esto más que nunca
- que no descarten usar técnicas de *embedding* si es necesario
- que aunque la *tech* aun no sea perfecta esto está evolucionando a un ritmo increíble y se va a ir perfeccionando

- que si ven necesario no descarten tener *test* automáticos con los *prompts* que ya hayan definido, para hacer pruebas de regresión cuando cambian cosas en la programación o ante nuevas versiones de la IA que usen

- que piensen todo lo que hagan como asistentes, porque no es determinístico y siempre requiere de alguien que valide, corrija, transforme o de el *ok*.

¡Y estoy dispuesta a ayudarlos en lo que necesiten!

13.15. Encuesta

13.15.1. *Link* de la encuesta

https://forms.gle/q5w43H1QQmZbt6ng8

13.15.2. Resultados de la encuesta

A continuación, se muestran algunas de las respuestas que no fueron mostradas anteriormente.

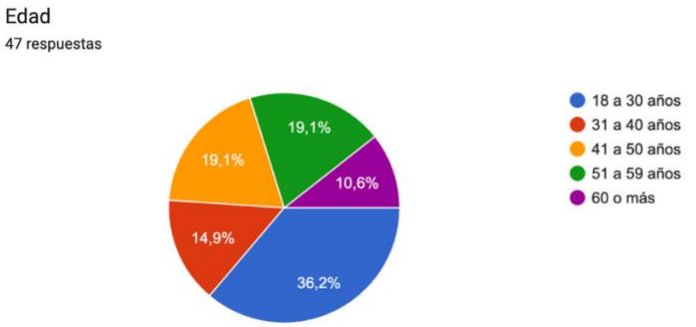


Ilustración 66- Resultados de la encuesta - Edad

Ilustración 67- Resultado de la encuesta - Cómo se buscan los datos?

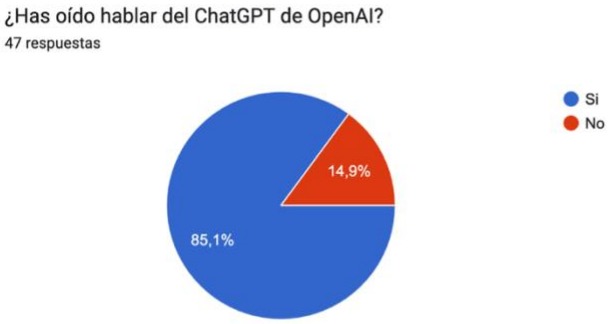


Ilustración 68- Resultado de la encuesta - Sobre ChatGPT

¿Qué características te gustaría ver en una plataforma de chat con inteligencia artificial personalizada para organizaciones/empresas?

47 respuestas

Accesible, facil de entender, rapida.
Me gustaría ver una plataforma que sea fácil de usar y que tenga una respuesta rápida y precisa.
Poder consultarle en lenguaje natural, que sea facil, rapida y amigable.
Que entienda bien el lenguaje natural, como si estuviera escibiendo un whatsapp
Me gustaria que la herramienta pueda ayudar al area financiera a detectar irregularidades en reportes
Que sea facil de usar hasta para los empleados mayores
No se
Que sea facil de usar, que se pueda copiar y pegar lo que responda
Me gustaría ver una plataforma que sea capaz de entender el lenguaje natural y las preguntas de los clientes, así como también una interfaz amigable y fácil de usar para el personal de ventas.
Lectura de contratos y documentos. Organizar carpetas e información. Interpretar grandes bases de datos.
Que sea amigable y que pueda entender lenguaje natural.
no seria util
Como mencione anteriormente, que pueda generar búsquedas en lenguaje natural.
No lo considero importante, pero me gustaría que fuese rápida.
facil, rapida, e informacion citada, de donde es sacada la informacion
Que sea rápida y segura
Que se pueda subir imagenes y inteprete videos

Ilustración 69- Resultado de la encuesta

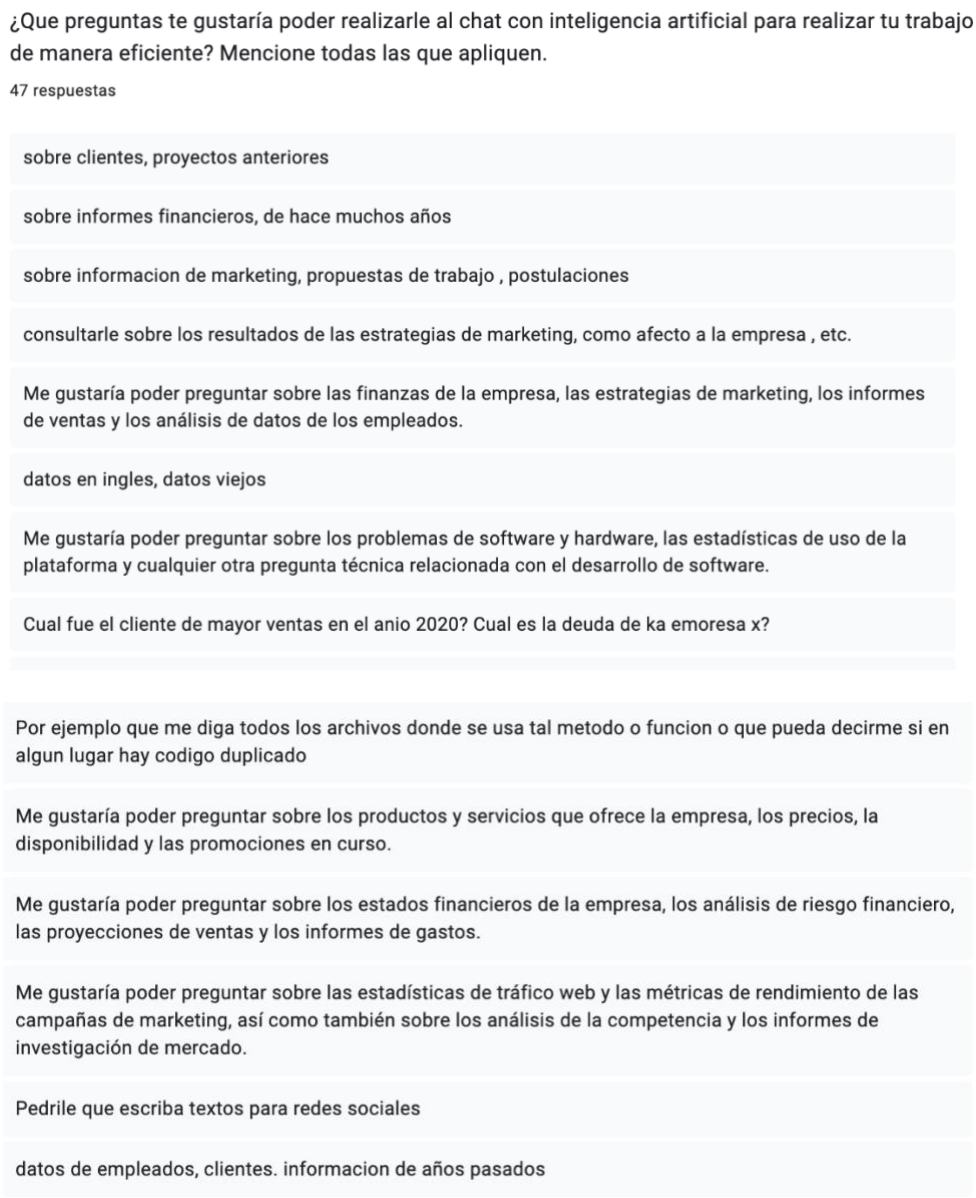


Ilustración 70- Resultados de la encuesta

13.16. Customer Journey

Nombre del usuario: Ana

Rol: Empleado de un departamento de desarrollo de negocios

Etapa 1: Identificación de una nueva oportunidad de negocio

Ana se entera de una nueva oportunidad de negocio y necesita presentar una propuesta sólida para su desarrollo. Sabe que esto requerirá una gran cantidad de información que se encuentra dispersa en diversos documentos y fuentes de la empresa.

Etapa 2: Búsqueda y recopilación de información

Ana comienza a buscar información relevante para su propuesta en distintos documentos y sistemas de la empresa. Esto implica abrir múltiples archivos y revisar correos electrónicos antiguos para obtener los datos necesarios.

Etapa 3: Comunicación con el *chatbot*

Ana decide probar Chat Your Data para facilitar el proceso. Abre la aplicación y hace una pregunta al *chatbot* sobre los datos necesarios para su propuesta. El *chatbot* responde instantáneamente con la información que Ana necesita, evitando la necesidad de buscar en varios lugares.

Etapa 4: Obtención de información adicional

Ana se da cuenta de que necesita información adicional para completar su propuesta. En lugar de buscar nuevamente, le pregunta al *chatbot* sobre los detalles faltantes. El *chatbot* le proporciona rápidamente los datos requeridos.

Etapa 5: Creación automática de la propuesta

Ana tiene toda la información que necesita gracias al *chatbot*. Decide hacer otra consulta a de Chat Your Data pidiéndole que le genere la propuesta de negocio basada en los datos recopilados. El *chatbot* crea una propuesta bien estructurada en función de los parámetros proporcionados por Ana.

Etapa 6: Revisión y finalización

Ana revisa la propuesta generada por el *chatbot* y realiza algunos ajustes. Al final, está satisfecha con el resultado y presenta la propuesta completa para la nueva oportunidad de negocio.

13.17. INVEST

Independiente: En la medida de lo posible, se debe tener cuidado de no introducir dependencias entre historias. Las dependencias entre historias conducen a problemas de priorización y planificación.

Negociable: Los detalles de una historia de usuario se establecen en colaboración entre el cliente y el equipo que la implementará. Esta colaboración incluye negociar el alcance: qué incluirá y qué no incluirá la implementación.

Valiosa: Una buena historia de usuario tiene valor para el cliente / usuario. Sin ese valor, no tiene sentido poner ningún esfuerzo en la historia.

Estimable: Es importante que los desarrolladores puedan estimar (o al menos adivinar) el tamaño de una historia o la cantidad de tiempo que tomará convertir una historia en código funcional. Si no puede estimar una historia, significa que aún no comprende el alcance lo suficientemente bien o que el alcance es demasiado grande para estimarlo fácilmente. No se necesitan estimaciones exactas, pero cuando se puede estimar una historia, también es más negociable.

Pequeña (*Small*): Es preferible que el esfuerzo para implementar una historia de usuario sea pequeño; horas, días… Las historias más pequeñas son más fáciles de estimar. Las grandes historias son más difíciles de estimar y, por lo tanto, menos negociables. La determinación final de si una historia tiene el tamaño adecuado se basa en el equipo, sus capacidades y las tecnologías en uso.

Comprobable (Testable): Si el cliente, en colaboración y ayudado por el equipo de implementación, no puede decir cómo verificar que se ha implementado lo que quiere, todavía no ha creado suficiente claridad sobre la historia.
[81]

13.18. Protocolos de Comunicación

13.18.1. API REST

Representational State Transfer o REST es un estilo arquitectónico que utiliza un conjunto de principios y restricciones para crear servicios web. La simplicidad de REST radica en su

capacidad para construir interfaces escalables y fáciles de usar basadas en métodos HTTP estándar. Esta elección permite una integración sencilla con el *frontend* y una amplia adopción en la comunidad de desarrolladores [82].

13.18.2. gRPC

Google Remote Procedure Call o gRPC es un protocolo de comunicación de alto rendimiento desarrollado por Google. Está diseñado para conectar sistemas distribuidos de manera eficiente, segura y rápida [83]. Gracias a sus características, como el uso de Protocol Buffers y la transmisión bidireccional, gRPC se vuelve una elección ideal para operaciones que requieren una transmisión de datos rápida y segura, como las interacciones con el PluginAI en Chat Your Data.

Para poder realizar una consulta con *gRPC* es necesario establecer una comunicación en común, y esto se logra mediante el *".proto"* el cual es un archivo que define los métodos que ambos servidores deben de implementar. A continuación, el *"prompt.proto"* de Chat Your Data:

```
syntax = "proto3";

package prompt;

service PromptService {
  rpc ProcessPrompt(PromptRequest) returns (CompletionAnswer) {}
}

message PromptRequest {
  string chatId = 1;
  string prompt = 2;
}

message CompletionAnswer {
  string completion = 1;
}
```

Ilustración 71- Archivo .proto

13.19. Capacidades de Langchain

13.19.1. Ventajas de LangChain

1. Capacidades avanzadas: El *framework* ofrece funcionalidades avanzadas en la creación de *chatbots*, gestión de preguntas y respuestas, lo que permitirá una implementación eficaz de estas funciones en la solución propuesta.

2. Encadenamiento único: La capacidad de encadenar complejos flujos de trabajo facilita la creación de procesos automatizados y la integración de diferentes etapas en el procesamiento de la información.
3. Documentación completa: La documentación detallada y bien estructurada de LangChain facilita el entendimiento y la implementación de sus herramientas y funcionalidades.
4. Buenas reseñas: Las reseñas positivas brindan confianza en la estabilidad y la eficiencia del *framework*.
5. Variedad de herramientas: LangChain proporciona una amplia gama de herramientas como Selectores de Ejemplo, *Prompts*, Almacenes de Vectores, Cargadores de Documentos, Analizadores de Salida, Divisores de Texto, y Modelos, lo que enriquece las posibilidades de desarrollo e implementación.
6. Conexiones inteligentes: La posibilidad de establecer conexiones inteligentes con diversas bases de datos y tecnologías es una ventaja significativa que facilita la integración y el manejo de datos.
   [84]

13.19.2. Casos de uso de LangChain:

LangChain trasciende la funcionalidad básica de un *framework* NLP, proporcionando una plataforma robusta para:

1. Análisis de documentos: Permite el análisis profundo de documentos para extraer, procesar y entender la información contenida en ellos.
2. Búsqueda personalizada: Facilita la creación de sistemas de búsqueda avanzados y personalizados.
3. *Chatbots*: Proporciona las herramientas necesarias para desarrollar *chatbots* sofisticados y eficientes.

13.20. Organización de componentes en React

Componentes React

React se basa en la composición de componentes, lo que permite crear y mantener de manera eficiente la interfaz de usuario de la aplicación. La modularidad de los componentes React hace que el código sea reutilizable y mantenible.

Organización de principales Componentes

Carpeta "***src***": Todos los componentes y archivos relacionados se encuentran dentro de esta carpeta principal.

Carpeta "***features***": Aquí se incluyen los componentes principales relacionados con las diferentes características o módulos de la aplicación. Cada característica importante tiene su propia carpeta dentro de "*features*". Por ejemplo:

- ***Authentication***: Esta carpeta contiene componentes relacionados con la autenticación, como el inicio de sesión, cierre de sesión y el registro.
- ***Chat***: Contiene los componentes relacionados con la funcionalidad de *chat*.
- ***ManageFiles***: Contiene componentes relacionados con la gestión de archivos.
- ***Organization***: Componentes relacionados con la organización en la aplicación.
- ***Profile***: Para la gestión del perfil del usuario.

Componentes Reutilizables: Se manejaron componentes reutilizables, utilizados a lo largo de la aplicación, como Botones, *Modals*, Texto.

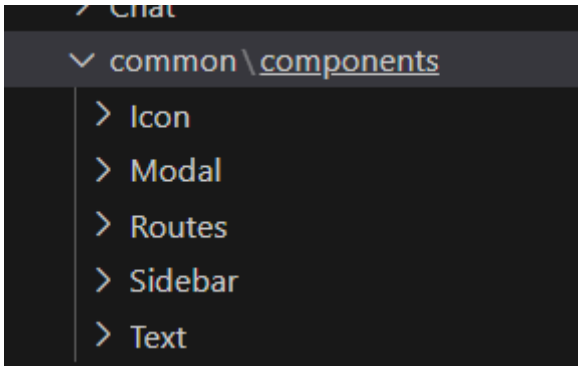


Ilustración 72- Imagen de carpeta common/components

Se utilizó *React Query* y *Redux* para administrar el estado de la aplicación en el *frontend*. React Query brinda un sistema eficiente para manejar los datos de forma sincronizada y hacer consultas al *backend*. Redux se utiliza para gestionar el estado global de la aplicación, lo que permite un flujo de estados coherente.

Comunicación entre componentes (Gestión del Estado): Se utilizó *React Query* y *Redux* para administrar el estado de la aplicación en el *frontend*. *React Query* brinda un sistema eficiente para manejar los datos de forma sincronizada y hacer consultas al *backend*. *Redux* se utiliza para gestionar el estado global de la aplicación, lo que permite un flujo de estados coherente.

Enrutamiento: Se utiliza React Router para gestionar la navegación entre diferentes vistas o páginas. Las rutas están configuradas en un archivo específico llamado "Routes.ts". Esto permite definir rutas para todas las características y dirigir a los usuarios a la vista correcta según sus acciones.

Estilos y Styled Components: Para garantizar una buena experiencia de usuario, se utilizaron Styled Components para gestionar los estilos de la aplicación.

Tipado Seguro con TypeScript: Para mejorar la calidad y la seguridad del código, se optó por utilizar *TypeScript*. Este lenguaje de programación introduce tipos estáticos en *JavaScript*, lo que facilita la detección temprana de errores y proporciona una documentación clara y concisa del código.

13.21. Justificación de arquitectura en módulos en *backend*

La elección de una arquitectura modular se justifica por la necesidad de anticiparse a futuras expansiones del sistema. Aunque en la actualidad el repositorio del servidor puede considerarse relativamente pequeño, esta estructura modular está diseñada pensando en el crecimiento a mediano y largo plazo. La plataforma está en una fase de desarrollo temprana, y se espera un aumento significativo en la funcionalidad en los próximos ciclos de desarrollo.

Una de las principales ventajas de la arquitectura modular es el aislamiento de responsabilidades. Cada módulo se dedica a una función o conjunto de funciones específicas, lo que permite una comprensión más clara y un mantenimiento más sencillo de cada componente. Por ejemplo, el módulo "Content" se ocupa exclusivamente de las operaciones relacionadas con documentos, mientras que el módulo "Chatbot" se centra en la interacción humano-IA. Esto facilita la localización y corrección de errores, así como la implementación de nuevas características sin afectar otros aspectos del sistema.

La arquitectura modular promueve la reutilización de código. Los módulos, especialmente aquellos dedicados a utilidades globales, como modelos e interfaces, pueden ser compartidos y utilizados en diferentes partes de la plataforma. Esta reutilización no sólo acelera el desarrollo, sino que también garantiza la consistencia en toda la aplicación. Los módulos "Content" y "Chatbot" pueden emplear componentes comunes, reduciendo así la duplicación de esfuerzos y el riesgo de inconsistencias.

La modularidad es esencial para la escalabilidad de la plataforma. La capacidad de agregar nuevos módulos y componentes sin interrumpir los existentes permite a Chat Your Data adaptarse a las crecientes demandas tecnológicas. Por ejemplo, cuando se desarrolle la gestión de usuarios, roles y permisos, estos pueden integrarse como nuevos módulos sin alterar la funcionalidad existente. Esta escalabilidad prepara para lograr el crecimiento a largo plazo y garantiza que el sistema pueda abordar futuros desafíos tecnológicos y funcionales.

La arquitectura modular reduce el acoplamiento entre componentes. Cada módulo tiene interfaces claras y definidas para la comunicación con otros módulos. Esto significa que un cambio en un módulo específico tiene un impacto mínimo en otros componentes. La reducción del acoplamiento mejora la estabilidad y la robustez del sistema, lo que es esencial para garantizar que las actualizaciones y mejoras puedan implementarse sin generar efectos secundarios no deseados.

En resumen, la arquitectura en módulos de Chat Your Data es una elección estratégica que potencia la extensibilidad del código y prepara la plataforma para un crecimiento sostenible a medida que evoluciona y se enfrenta a nuevos desafíos tecnológicos y funcionales.

13.22. Diagramas *backend*

13.22.1. Módulo *Content*

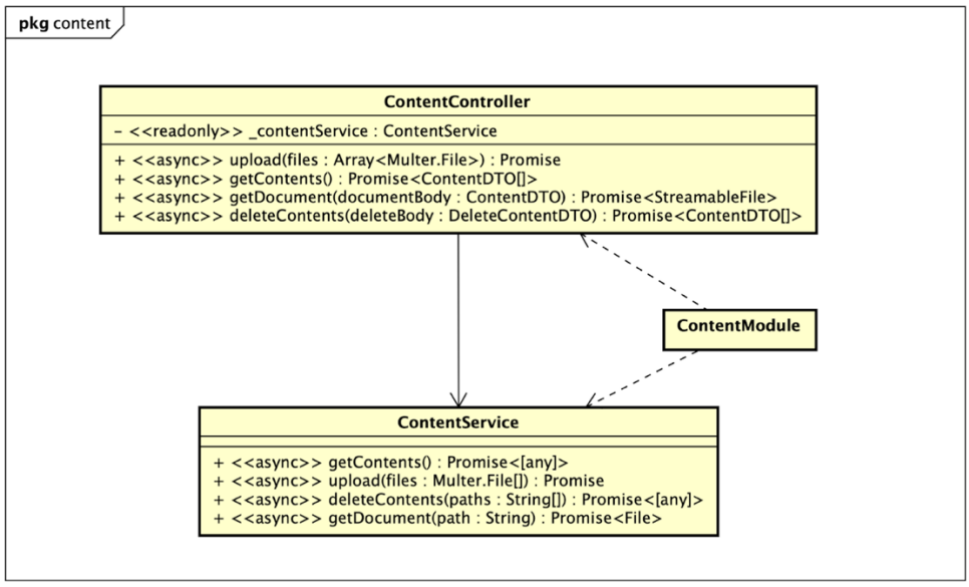


Ilustración 73- Diagrama módulo Content

13.22.2. Módulo *Chatbot*

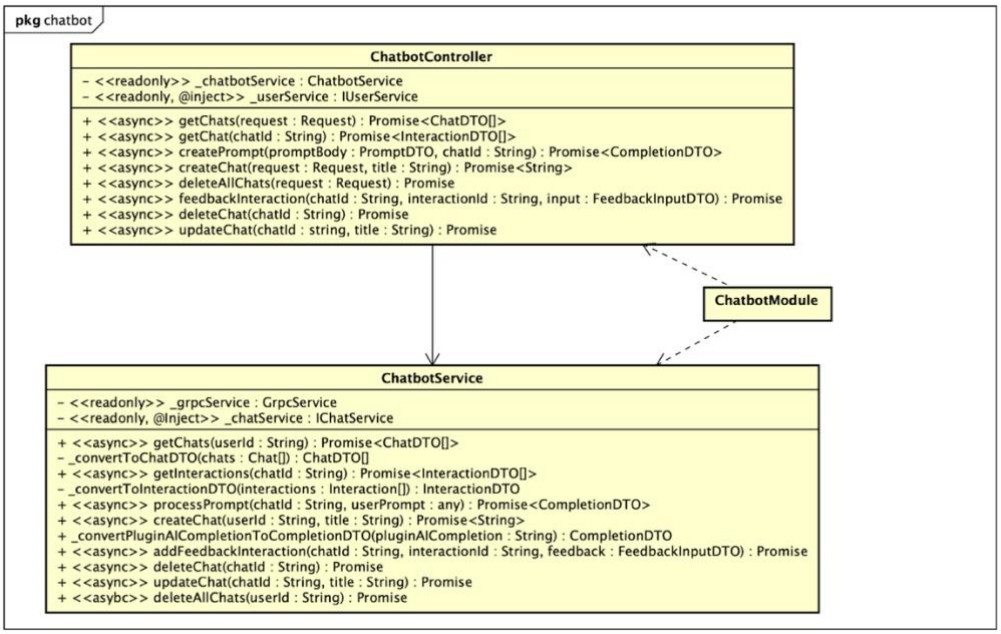


Ilustración 74- Diagrama de módulo *Chatbot*

13.22.3. Módulos de comunicación Externa

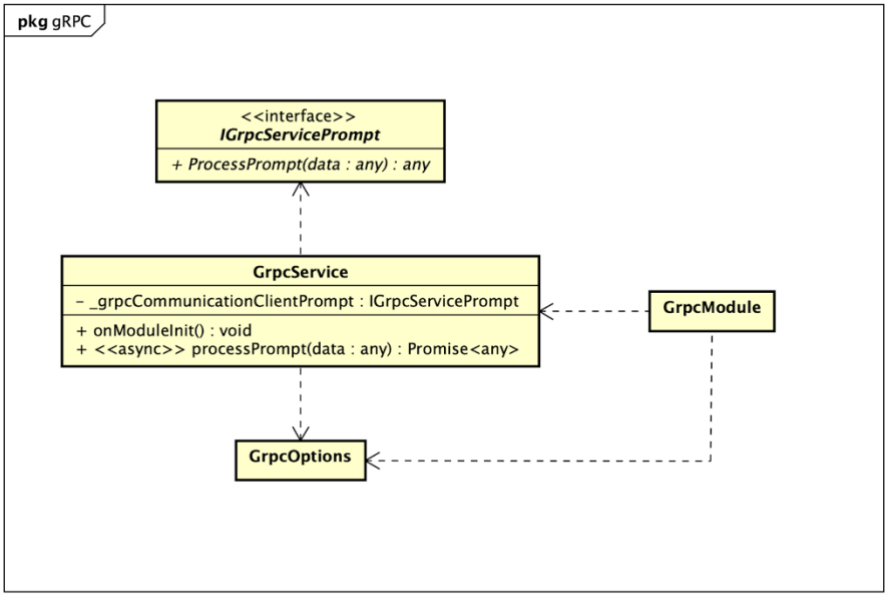


Ilustración 75- Módulos de comunicación externa

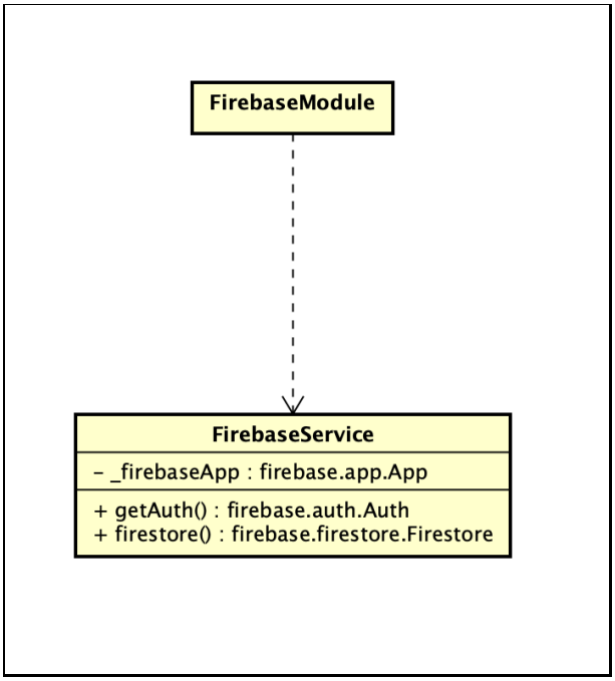


Ilustración 76- Modulo de Firebase

13.22.4. Módulos de utilidades globales

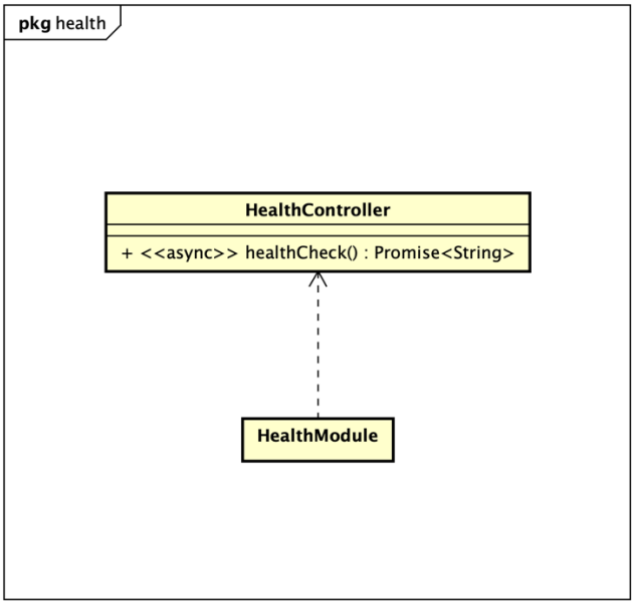


Ilustración 77- Modulo de utilidades globales: Health

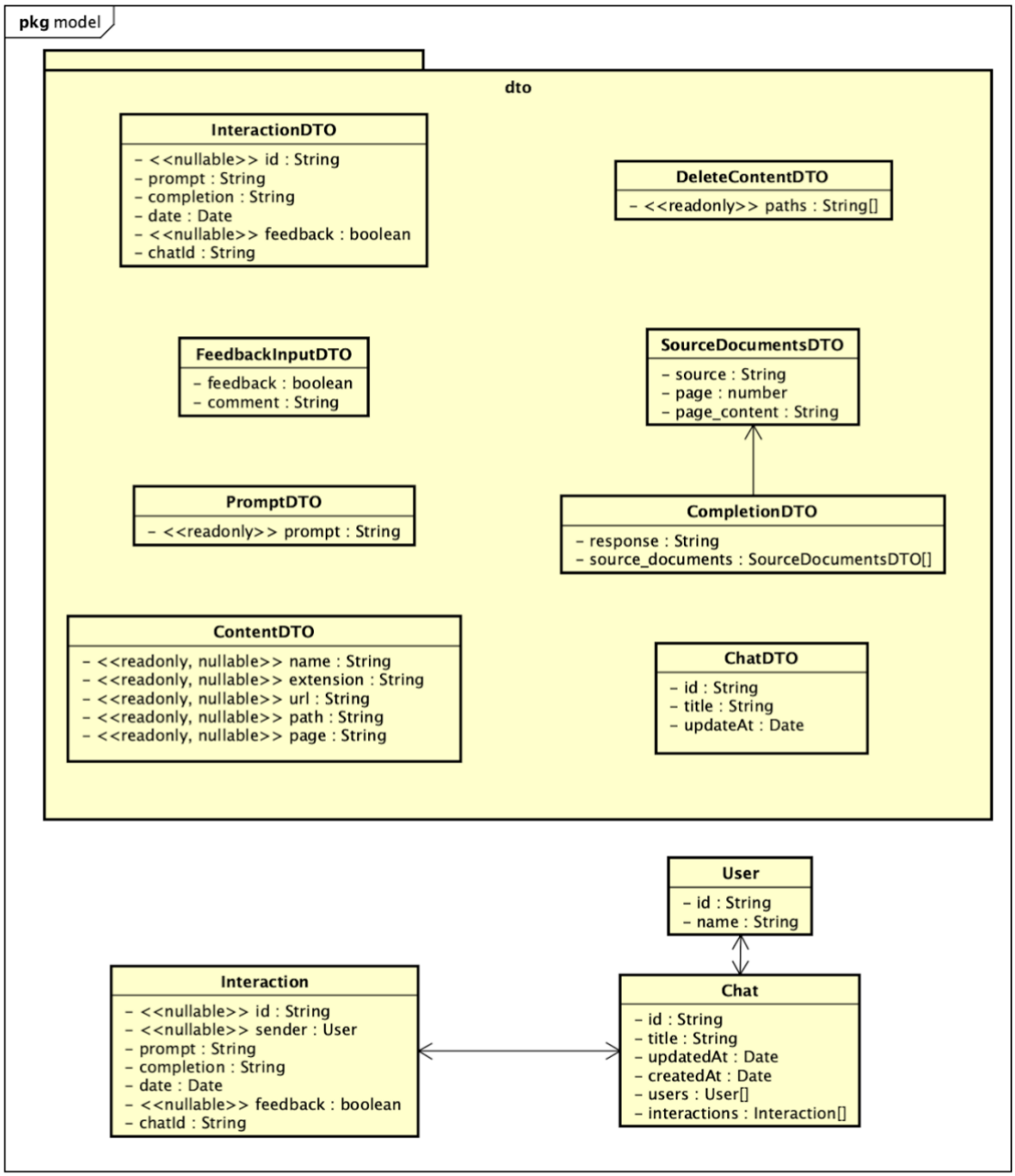


Ilustración 78- Diagrama de DTO

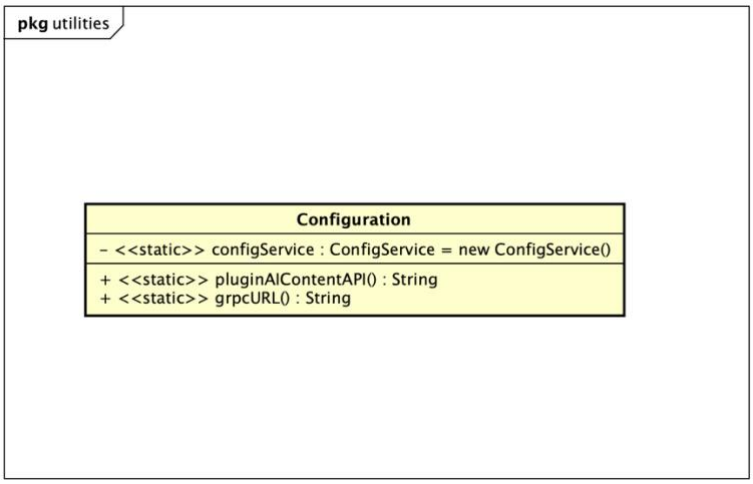


Ilustración 79- Diagrama de configuración

13.23. Extensibilidad y autenticación en Chat Your Data

En Chat Your Data, se ha optado por extender los *guards* y las estrategias existentes para abordar requisitos específicos de seguridad sin comprometer la implementación existente. Esto significa que la arquitectura es altamente adaptable a futuros cambios en los requisitos de seguridad o en los proveedores de autenticación.

Estos *guards* son implementados en Chat Your Data por FirebaseAuthGuard y la estrategia FirebaseAuthStrategy. Estos componentes se encargan de la autenticación a través de Firebase Authentication, pero su diseño permite que se agreguen estrategias adicionales o se cambien proveedores de autenticación en el futuro sin afectar significativamente el código existente.

```
import { ExecutionContext, Injectable } from "@nestjs/common";
import { AuthGuard } from "@nestjs/passport";
import { Reflector } from "@nestjs/core";

@Injectable()
export class FirebaseAuthGuard extends AuthGuard("firebase-auth") {
  constructor(private reflector: Reflector) {
    super();
  }
```

Ilustración 80- FirebaseAuthGuard *class*

```
import { PassportStrategy } from "@nestjs/passport";
import { Injectable, UnauthorizedException } from "@nestjs/common";
import { Strategy, ExtractJwt } from "passport-firebase-jwt";
import { FirebaseApp } from "../firebase.app.service";

@Injectable()
export class FirebaseAuthStrategy extends PassportStrategy(
  Strategy,
  "firebase-auth"
) {
  constructor(private readonly firebaseApp: FirebaseApp) {
    super({
      jwtFromRequest: ExtractJwt.fromAuthHeaderAsBearerToken(),
    });
  }
```

Ilustración 81 - FirebaseAuthStrategy *class*

Para garantizar la autenticación en todos los controladores de la aplicación, se aplica el FirebaseAuthGuard a nivel de controlador. A continuación, se muestra un ejemplo de cómo se utiliza en el controlador ContentController:

```
@Controller("content")
@UseGuards(FirebaseAuthGuard)
export class ContentController {
```

Ilustración 82- Aplicación UseGuards en ContentController

Esto asegura que todas las rutas y métodos dentro de ContentController requieran autenticación mediante Firebase antes de acceder a ellos.

13.24. Patrón Singleton

El patrón de diseño Singleton se enfoca en restringir la creación de objetos de una clase, asegurando que solo exista una única instancia de la misma. Además, provee un punto de acceso global a dicha instancia, garantizando que todas las partes del *software* se comuniquen con esa única instancia. [85]

13.24.1. Características principales

- Una Sola Instancia:
    - Solo se permite crear una instancia de la clase.
- Acceso Global:

    - Se facilita un mecanismo para acceder a esa única instancia desde cualquier punto del código, de manera controlada y predecible.
- Inicialización Perezosa (Lazy Initialization):
    - La creación de la instancia única generalmente no ocurre hasta que se solicita por primera vez, lo que puede ayudar a reducir el uso de recursos y mejorar los tiempos de carga si la instancia es costosa de crear o inicializar.

13.24.2. Ventajas

- Control de Instancia:
    - Garantiza que una clase sea instanciada una sola vez.
- Acceso Global:
    - Ofrece un único punto de acceso a la instancia a través de la aplicación.
- Reducción de Redundancia
    - Evita la creación de múltiples instancias innecesarias, economizando recursos.

13.24.3. Desventajas

- Problemas de *concurrency*:
    - En entornos con multihilos, puede haber desafíos para garantizar que la instancia se cree una única vez si varios hilos intentan crearla simultáneamente.
- Problemas de *testing*:
    - La instancia única y el acceso global pueden complicar las pruebas unitarias al acoplar diferentes partes del código.
- Acoplamiento Global:
    - Al ser un acceso global, diferentes módulos del *software* pueden depender en exceso de él, incrementando el acoplamiento en el sistema.

13.25. Diagramas PluginAI

13.25.1. Canales de comunicación

13.25.1.1. gRPC

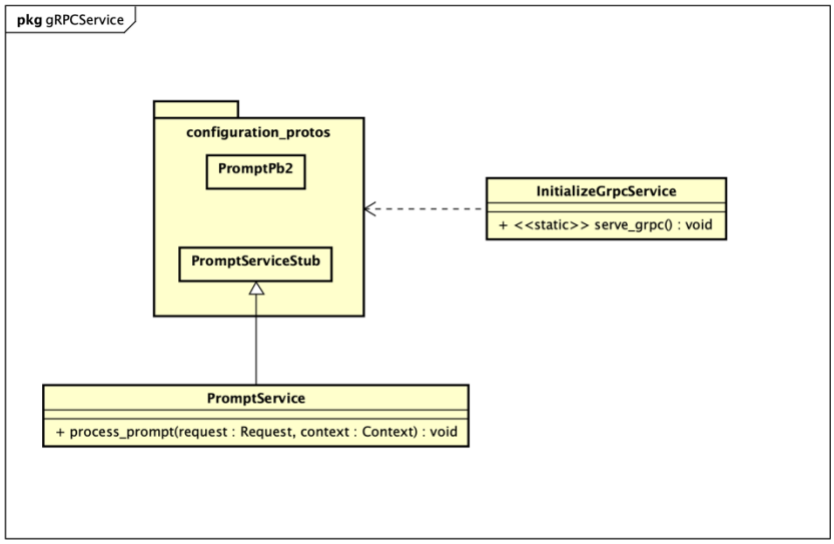


Ilustración 83- Diagrama - Canales de comunicación: gRPC

13.25.1.2. API

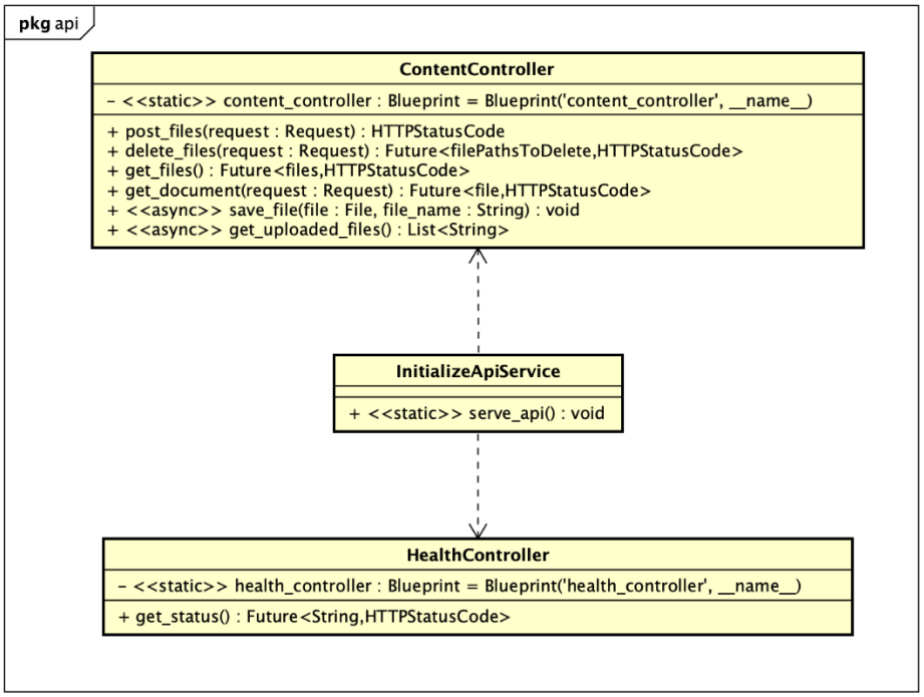


Ilustración 84- Diagrama - Canales de comunicación: API

13.25.2. Paquete de business_logic

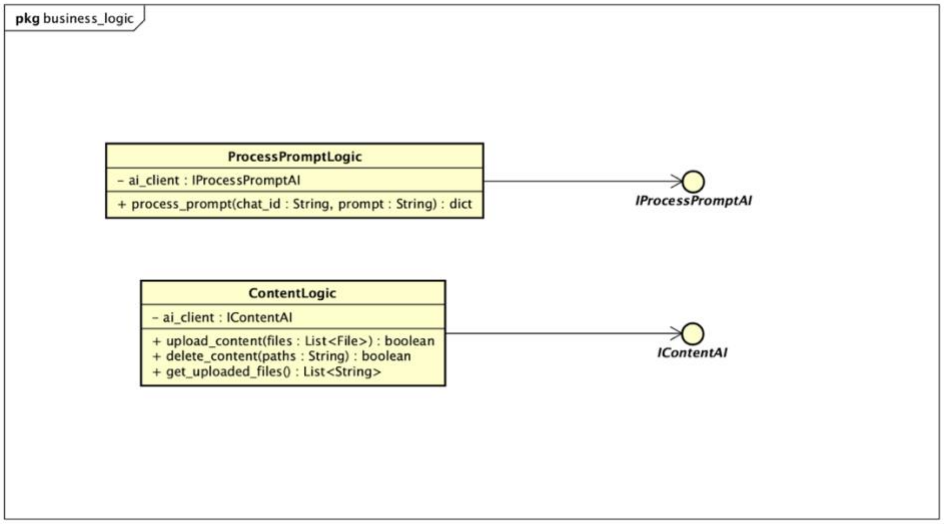


Ilustración 85- Diagrama de paquete de business_logic

13.25.3. Paquete de ai_client_interface

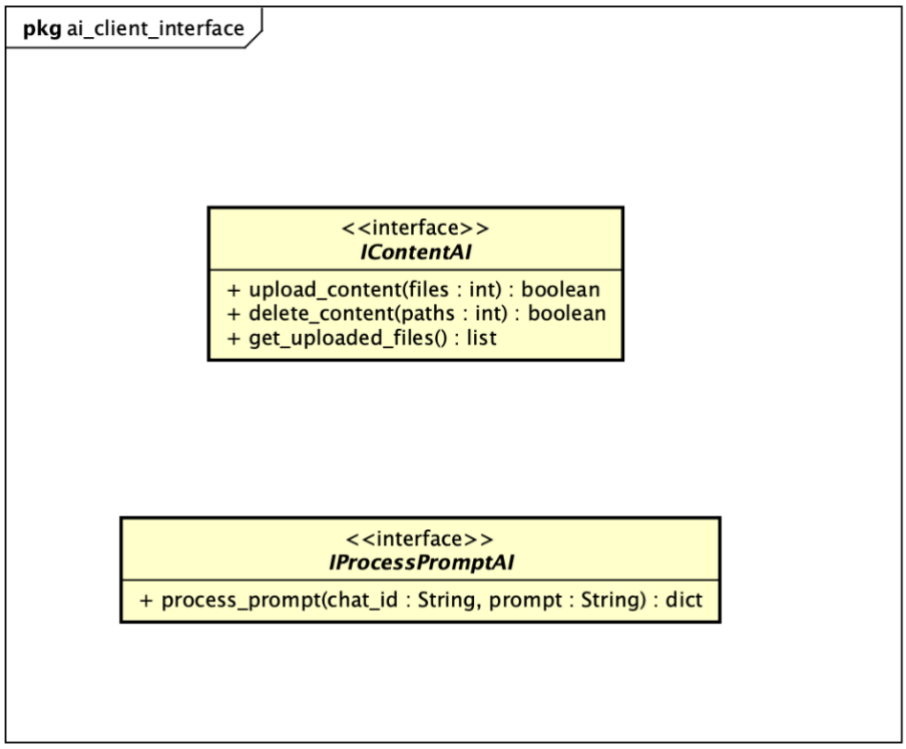


Ilustración 86- Diagrama de paquete de ai_client_interface

13.25.4. Paquete de langchain_client

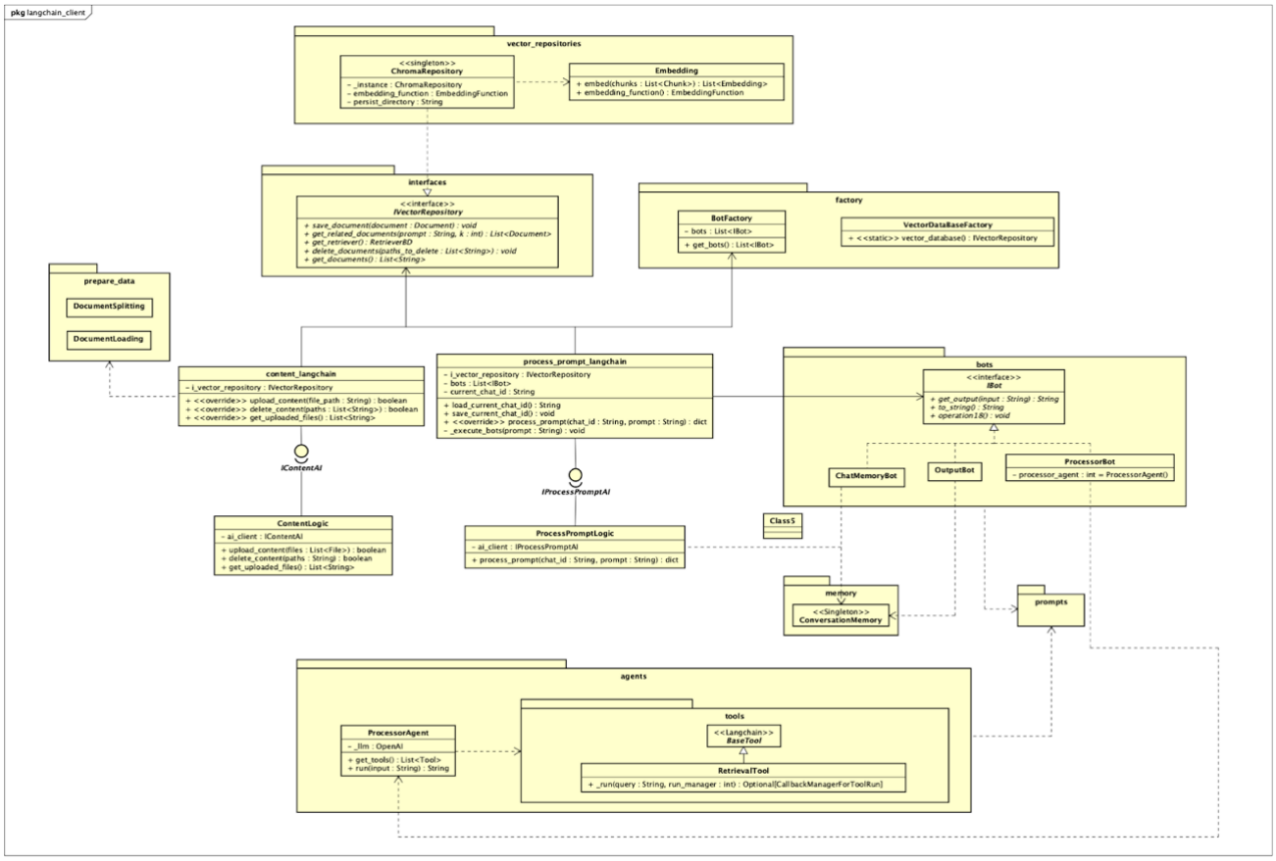


Ilustración 87- Diagrama de langchain_client

13.25.5. Método upload_content

```
def upload_content(self, file_path) -> bool:
    log.ConsoleLog("Calling upload content method from Langchain").info()

    docs = DocumentLoading().load_pdf_document(file_path)
    splits = DocumentSplitting().split_recursive_documents(docs)

    self.i_vector_repository.save_document(splits)

    return True
```

Ilustración 88- Método upload_content de la clase content_langchain

13.25.6. Paquete de vector_repositories

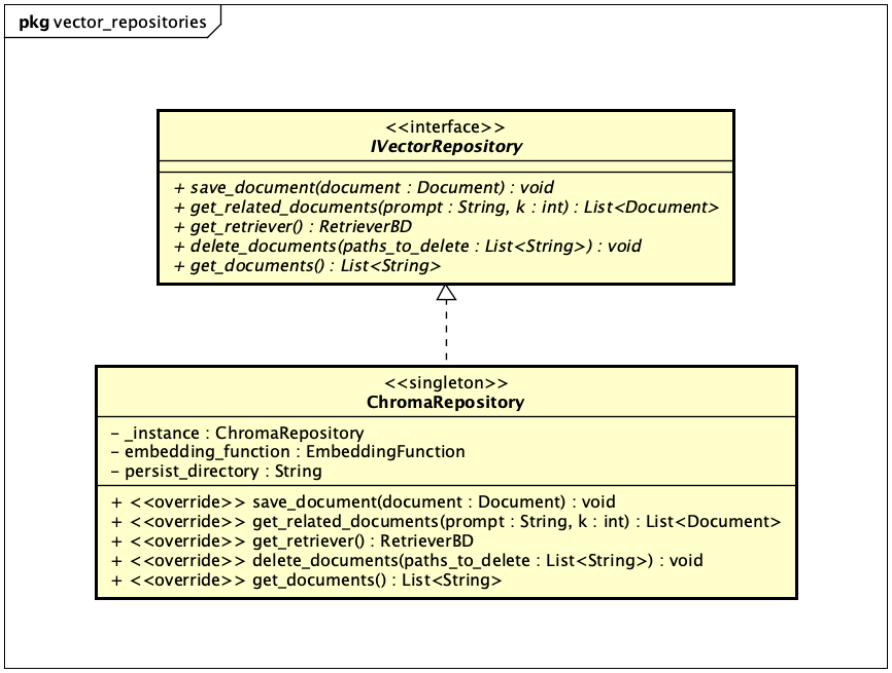


Ilustración 89- Diagrama de paquete de vector_repositories

13.25.7. Paquete de Prompts

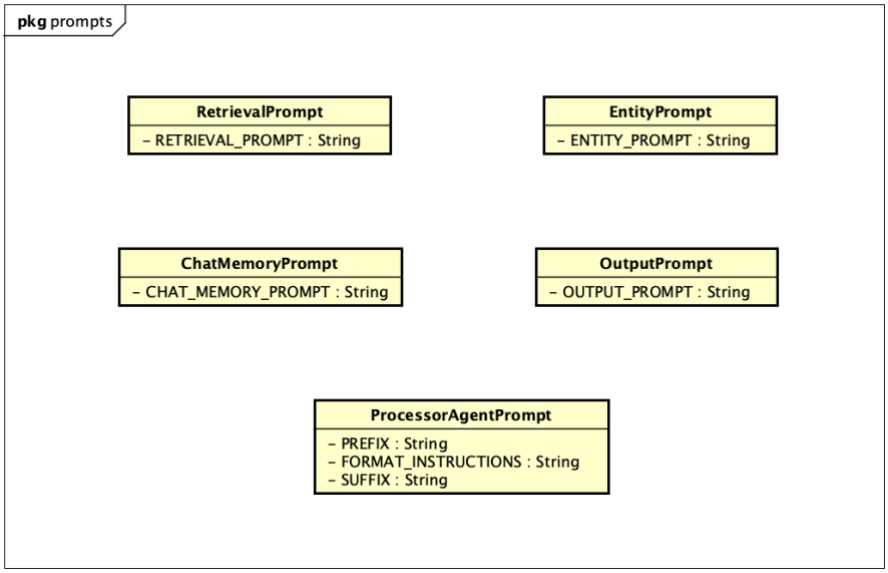


Ilustración 90- Diagrama de paquetes de Prompts

13.26. *Deployment* en AWS

13.26.1. Docker

Docker emergió como el pilar fundamental en la fase de despliegue. Esta herramienta permitió contenerizar y encapsular las aplicaciones, asegurando uniformidad y transferibilidad entre diversos ambientes [86]. Al usar Docker, el equipo podía estar seguro de que si una imagen funciona correctamente en un ambiente de desarrollo, lo haría igualmente en producción.

13.26.2. Elastic Container Registry (ECR)

ECR de AWS, más que un simple repositorio, facilita el almacenamiento, administración y despliegue de imágenes Docker. Se incluyeron las imágenes para el *frontend*, PluginAI, y *backend* en ECR, estableciendo así un puente esencial para la siguiente etapa [87].

13.26.3. Amazon Elastic Container Service (ECS)

Amazon Elastic Container Service (ECS) es una herramienta altamente escalable y de alto rendimiento de orquestación de contenedores. Ofrece soporte nativo para Docker, lo que permite ejecutar y gestionar contenedores a cualquier escala [88]. Para Chat Your Data, ECS proporcionó la infraestructura y la gestión necesaria para un despliegue sin fisuras y eficiente. Aquí detallamos más a fondo cada uno de sus componentes empleados:

13.26.3.1. *Clusters*

Los clústeres en ECS son agrupaciones lógicas de recursos, y, en este caso, se crearon tres distintos. Estos clústeres representan el entorno operativo para las imágenes contenerizadas y definen la capacidad computacional utilizada para ejecutar las tareas.

Implementación: Cada clúster fue diseñado meticulosamente para hospedar una de las imágenes generadas anteriormente, asegurando que cada servicio tuviera el entorno más apropiado para su función.

Beneficios: La creación de clústeres individualizados para cada imagen permitió una mejor gestión de recursos, facilitando la escalabilidad y el aislamiento de cada servicio.

13.26.3.2. *Task definitions*

Las definiciones de tarea en ECS especifican la estructura, los requerimientos y las configuraciones para cada contenedor. Son, en esencia, la 'receta' que ECS utiliza para ejecutar las aplicaciones.

Implementación: Para cada imagen (frontend, PluginAI, servidor), se creó una definición de tarea que detalla los recursos, configuraciones y otros aspectos esenciales.

Beneficios: Gracias a estas definiciones de tarea, se pudo lograr una ejecución coherente y predecible de los contenedores, garantizando la replicabilidad del despliegue en cualquier momento.

13.26.3.3. Variables de entorno

Las variables de entorno permiten parametrizar y personalizar la ejecución de aplicaciones dentro de contenedores.

Implementación: Se configuraron específicas variables de entorno dentro de cada contenedor, determinando, por ejemplo, conexiones a bases de datos, rutas de servicios externos y configuraciones específicas del sistema.

Beneficios: Esta capa de personalización garantizó que las aplicaciones pudieran interactuar adecuadamente con los recursos y servicios externos, así como adaptarse a cualquier cambio en el entorno sin necesidad de modificar el código.

13.26.3.4. Orden de despliegue

La secuencia en la que se desplegaron las imágenes fue crucial debido a las dependencias entre los servicios.

Implementación: Se priorizó el despliegue del *plugin_ai* para obtener primero su dirección IP. Esta IP se utilizó posteriormente para configurar el servidor, garantizando que pudiera comunicarse con el plugin sin problemas.

Beneficios: Este enfoque secuencial evitó problemas de conexión y garantizó una comunicación ininterrumpida entre los servicios, optimizando la experiencia del usuario.

13.26.3.5. Ejecución

La fase final del despliegue en ECS involucra el lanzamiento efectivo de los contenedores basándose en las definiciones de tarea.

Implementación: Las definiciones de tarea establecidas anteriormente se asignaron a sus respectivos *clusters*, activando la ejecución de los contenedores.

Beneficios: Esta ejecución orquestada permitió que cada contenedor se ejecutara en el entorno más adecuado, aprovechando al máximo los recursos asignados y garantizando un rendimiento óptimo.

13.26.4. EC2 (Elastic Compute Cloud)

Una vez activados, los *clusters* residieron en instancias específicas de EC2. Estas instancias no solo proporcionaron las direcciones IP públicas necesarias para la accesibilidad, sino también el poder computacional necesario para un rendimiento óptimo.

13.26.5. Security Groups

La integridad y la seguridad son primordiales. Para ello, se recurrió a los Security Groups de AWS, que se configuraron para canalizar y regular el tráfico entrante y saliente. Al limitar el acceso únicamente a puertos esenciales, se mantuvo la accesibilidad de las aplicaciones al tiempo que se fortaleció su seguridad.

13.27. Organización del Notion

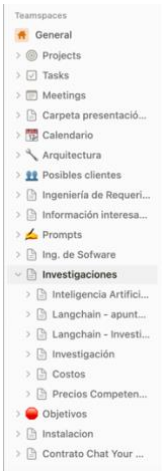


Ilustración 91- Organización del Notion

13.28. Cursos

https://www.youtube.com/watch?v=aywZrzNaKjs&t=104s

https://www.deeplearning.ai/short-courses/langchain-chat-with-your-data/

https://www.deeplearning.ai/short-courses/langchain-for-llm-application-development/

13.29. Scrum

13.29.1. Ciclo de Scrum

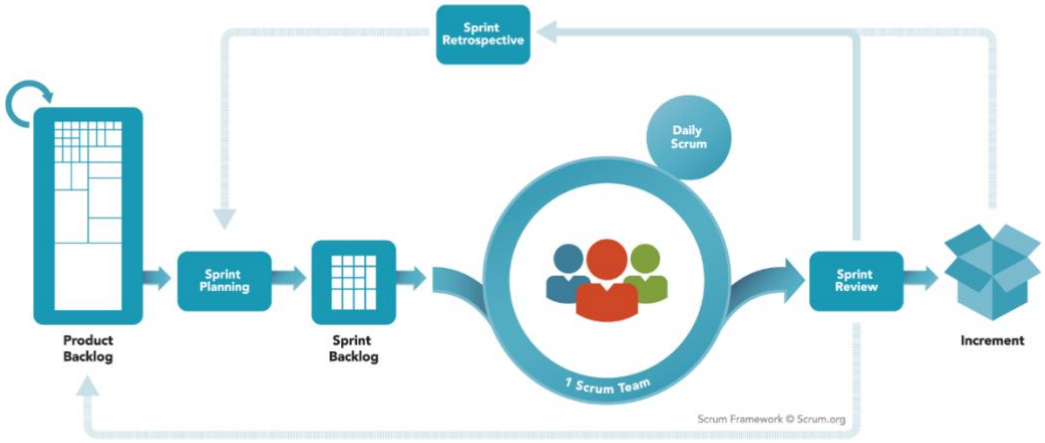


Ilustración 92- Ciclo de Scrum

Extraída de Scrum Framework, scrum.org

13.29.2. Artefactos de Scrum

Product Backlog:

El Product Backlog es la lista de todas las tareas e historias de usuario ordenadas que deben ser completadas a lo largo del proyecto. Esta lista puede cambiar y evolucionar constantemente a medida que se descubren nuevos requerimientos o se obtiene retroalimentación de los usuarios. Fue definido luego del proceso de ingeniería de requisitos.

Sprint Backlog:

El Sprint Backlog consiste en un conjunto de elementos del Product Backlog seleccionados específicamente para el sprint en curso. Además, incluye un plan de acción detallado para entregar el incremento deseado durante el Sprint. Se define después de cada Sprint Planning y se enfoca en las historias de usuario que se desarrollarán durante el próximo sprint. Proporciona una visión clara de las tareas y de lo que el equipo se ha comprometido a completar en el sprint actual.

Product Increment:

El Product Increment es el entregable de este sprint más los anteriores. Este entregable debe cumplir con la Definición de Terminado (Definition of Done).

13.30. Evidencia Retro *meeting*

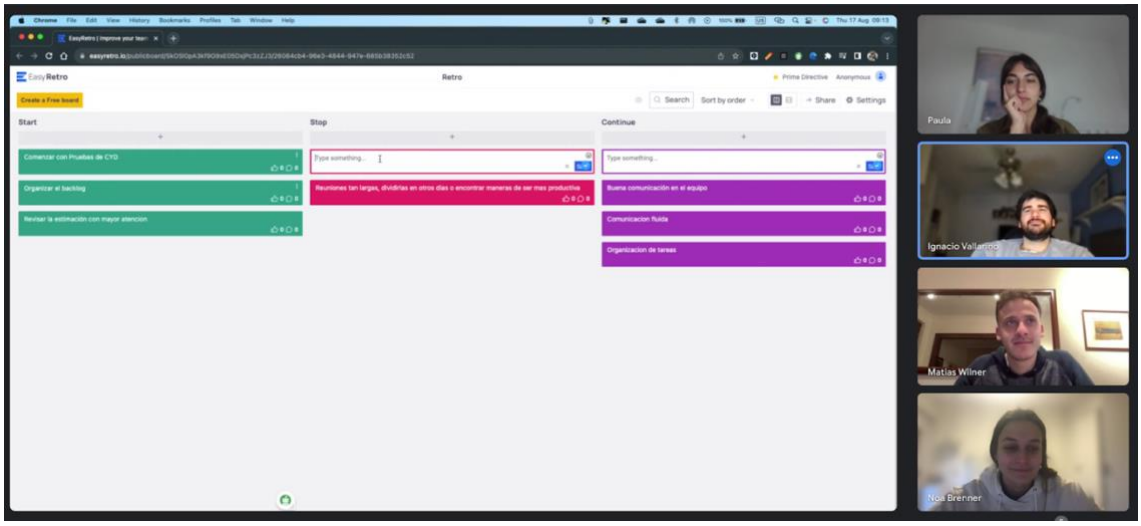


Ilustración 93- Evidencia de Retro *meeting*

13.31. Impacto

Valor	Impacto
0	No genera impacto
1	Impacto muy bajo
2	Impacto bajo
3	Impacto medio
4	Impacto alto
5	Impacto muy alto

Tabla 14 - Tabla de impacto

13.32. Probabilidad de ocurrencia

Probabilidad	Descripción
0	No hay probabilidad de que ocurra
0,1 - 0,2	Probabilidad muy baja de ocurrencia
0,3 - 0,4	Probabilidad baja de ocurrencia
0,5 - 0,6	Probabilidad media de ocurrencia
0,7- 0,8	Probabilidad alta de ocurrencia
0,9	Probabilidad muy alta de ocurrencia
1	Se convirtió en problema

Tabla 15 - Tabla de probabilidad de ocurrencia

13.33. Riesgos

Identificador de riesgo	**Riesgo**	**Tipo de riesgo**	**Descripción**
RI1	Pérdida del potencial cliente	Interesados	El beta tester decide no continuar trabajando con el equipo.
RI2	Dificultades de Comunicación con el potencial beta tester	Interesados	Falta de comunicación efectiva con el beta tester.
RI3	Retrasos en retroalimentación del potencial beta tester	Interesados	beta tester demora en proporcionar retroalimentación o aprobaciones necesarias.

RI4	Satisfacción del potencial beta tester	Interesados	El producto no cumpla con las expectativas o necesidades del potencial beta tester
RI5	Tener mala relacion con el potencial beta tester	Interesados	Existe el riesgo de que se desarrolle una mala relación con el potencial beta tester, lo cual puede afectar la comunicación, la satisfacción del beta tester y la colaboración efectiva.
RP1	No manejar intenciones inesperadas	Producto	Que el chat no esté capacitado a responder solicitudes inesperadas y falle debido a esto.
RP2	La etapa de investigación retrase demasiado el proyecto	Producto	Que el tiempo de desarrollo no sea el suficiente por la extensión de la etapa de investigación.
RP3	No conocer al usuario tipo	Producto	No lograr cumplir con las expectativas de los usuarios porque no se conoce al usuario
RE1	Falta de habilidades o experiencia	Equipo	Los miembros del equipo carecen de las habilidades o experiencias necesarias para el proyecto.
RE2	Comunicación inadecuada	Equipo	Falta de comunicación efectiva entre los miembros del equipo.
RE3	Sobrecarga de trabajo	Equipo	Miembros del equipo con demasiadas responsabilidades o trabajando en múltiples proyectos.

RE4	Abandono del tutor	Equipo	Existe el riesgo de que el tutor designado para el proyecto decida abandonar su rol y ya no participar en el proyecto. Esto puede deberse a cambios en su disponibilidad, falta de interés u otras circunstancias.
RE5	No seguir los procesos de ingeniería de *software* y prácticas de gestión adecuadamente	Equipo	Existe el riesgo de que el equipo no siga adecuadamente los procesos de ingeniería de *software* y las prácticas de gestión establecidas para el proyecto. Esto puede resultar en retrasos, errores en el desarrollo o incumplimiento de los estándares de calidad establecidos.
RE6	Tener mala relación entre el equipo	Equipo	riesgo de que se desarrolle una mala relación entre los miembros del equipo, lo cual puede afectar negativamente el ambiente de trabajo, la colaboración y la productividad.
RE7	Pérdida de un miembro del equipo	Equipo	riesgo de que se produzca la pérdida de un miembro del equipo, ya sea por renuncia, enfermedad u otras circunstancias imprevistas.
RE8	Que el equipo no sepa cumplir con sus roles	Equipo	Existe el riesgo de que los miembros del equipo no sean capaces de cumplir adecuadamente con los roles y responsabilidades asignados.

RE9	Que el equipo no logre organizarse adecuadamente	Equipo	Existe el riesgo de que el equipo no logre organizarse de manera adecuada para cumplir con las tareas y los plazos establecidos. Esto puede resultar en retrasos, falta de coordinación o dificultades para priorizar y distribuir las tareas de manera efectiva.
RT1	No conseguir acceso a servicios pagos	Tecnológico	Aumento de los costos del proyecto por encima del presupuesto previsto.
RT3	Tecnologías investigadas no sean las adecuadas para el proyecto	Tecnológico	Al ser tecnologías nuevas, puede ocurrir que las seleccionadas no sean las adecuadas
RT4	No poder probar todos los casos	Tecnológico	Son infinitos casos, no se puede realizar pruebas para todos los casos.
RT5	Actualización de tecnologías que afecten el desempeño del producto	Tecnológico	Es un tema muy reciente, donde aparecen tecnologías nuevas constantemente.
RT6	No conocer las tecnologías	Tecnológico	El equipo comienza el proyecto sin conocimientos claros de inteligencia artificial, y machine learning

Tabla 16- Tabla de riesgos

13.34. Evidencia de Excel de riesgos

IDENTIFICADOR DE RIESGO	RIESGO	IMPACTO	PROBABILIDAD DE OCURRENCIA	SCORE	TIPO DE RIESGO	DESCRIPCIÓN	PLAN DE MITIGACION	PLAN DE CONTINGENCIA
RI1	Perdida del potencial cliente	4	0,4	1,6	Interesados	Cliente decide no continuar trabajando con el equipo.	Tener una comunicacion efectiva	Mantener una relación sólida con el cliente y estar atentos a las señales de insatisfacción o inquietudes.
RI2	Dificultades de Comunicación con el potencial cliente	4	0,6	2,4	Interesados	Falta de comunicación efectiva con el cliente.	Tener contacto fluido y constante con el cliente	Establecer canales de comunicación claros y reuniones regulares para garantizar una comunicación efectiva.
RI3	Retrasos en retroalimentación del potencial Cliente	2	0,2	0,4	Interesados	Cliente demora en proporcionar retroalimentación o aprobaciones necesarias.	Marcar instancias fijas de retroalimentacion (una cada dos semanas)	Establecer plazo alternativo para la retroalimentación y seguimiento de las entregas pendientes de aprobación.
RI4	Satisfacción del potencial cliente	2	0,1	0,2	Interesados	El producto no cumpla con las expectativas o necesidades del cliente	Aplicar los cambios que el cliente fue sugiriendo mediante los feedbacks	Demos constantes para que pueda ir viendo el proceso
RI5	Tener mala relacion con el potencial cliente	2	0,1	0,2	Interesados	Existe el riesgo de que se desarrolle una mala relación con el cliente, lo cual puede afectar la comunicación, la satisfacción del cliente y la colaboración efectiva.	Establecer una comunicación abierta y regular con el cliente desde el inicio del proyecto. Entender las necesidades del cliente,aceptar los comentarios y tomarlos	Asignar a un responsable del equipo para tratar los problemas de comunicacion con el cliente y encontrar soluciones.
RP1	No manejar intenciones inesperadas	1	0,6	0,6	Producto	Que el chat no esté capacitado a responder solicitudes inesperadas y falle debido a esto.	Prototipos y validaciones	Tener una respuesta preparada por si hay respuestas que no puede contestar
RP2	La etapa de investigación retrase demasiado el proyecto	4	0,3	1,2	Producto	Que el tiempo de desarrollo no sea el suficiente e por la extensión de la etapa de investigación.	Ponerse deadlines para determinados temas	Aumentar horas de desarrollo
RP3	No conocer al usuario tipo	4	0,3	1,2	Producto	No lograr cumplir con las expectativas de los usuarios porque no se conoce al usuario	Hacer relevamiento de cuales son los usuarios finales	Dedicar tiempo en investigar al usuario
RE1	Falta de habilidades o experiencia	3	0,3	0,9	Equipo	Los miembros del equipo carecen de las habilidades o experiencias necesarias para el proyecto.	Realizar capacitaciones para desarrollar las habilidades necesarias en el equipo. Asignar tareas de acuerdo a las habilidades de cada miembro del equipo.	Buscar asesoramiento externo y asignar tareas según las habilidades de cada miembro. Reasignar tareas o buscar apoyo en el equipo para mitigar el impacto.
RE2	Comunicación inadecuada	4	0	0	Equipo	Falta de comunicación efectiva entre los miembros del equipo.	Implementar las herramientas de comunicación y establecer reuniones regulares para discutir el progreso.	Designar a un facilitador de comunicacion en el equipo, que se se encargue de abordar los problemas de comunicacion. Considerar establecer nuevos canales de comunicacion.
RE3	Sobrecarga de trabajo	2	0	0	Equipo	Miembros del equipo con demasiadas responsabilidades o trabajando en múltiples proyectos.	Monitorear regularmente la carga de trabajo y distribuir las tareas equitativamente. Priorizar las tareas y posponer las no escenciales. Comunicar al equipo cualquier sobrecarga o dificultades	Monitorear la carga de trabajo y redistribuir tareas si es necesario para evitar la fatiga y el agotamiento.
						Existe el riesgo de que el tutor designado		

Ilustración 94- Evidencia de Excel de riesgos

13.35. Gestión de comunicación

13.35.1. Comunicación interna del equipo

Para garantizar una comunicación fluida y efectiva entre los integrantes del equipo, se emplearon varias herramientas y canales. Entre ellos, se destacan el uso de aplicaciones como WhatsApp, donde se tenía un grupo con todos los integrantes, siendo este el utilizado diariamente y Google Meet, que permitieron mantener un contacto constante. Además, se llevaron a cabo reuniones presenciales, las cuales el equipo encontró especialmente productivas, enriquecedoras y divertidas.

13.35.2. Comunicación con el tutor y la universidad

En lo que respecta a la comunicación con el tutor y otros representantes de la universidad, se optó por utilizar la plataforma Microsoft Teams para mantener un canal de comunicación directo y organizado. Además, se estableció la práctica de coordinar reuniones puntuales a través de correo electrónico y programar encuentros mediante Google Meet. Esta metodología permitió abordar cuestiones específicas del proyecto, como la arquitectura (Gastón Mousques), los requerimientos (Alvaro Ortas) y la gestión de proyectos (Martin Solari), con la participación de expertos en cada área.

13.35.3. Evidencia de reunión con el BID

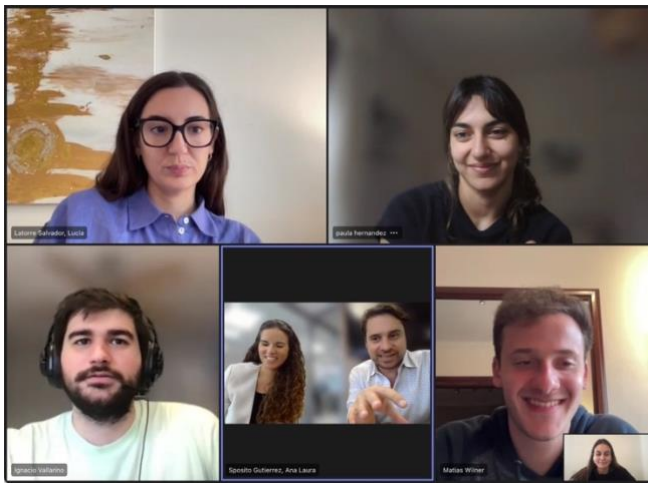


Ilustración 95- Reunión con el BID

13.36. Evidencia de interoperabilidad

13.36.1. Escritorio

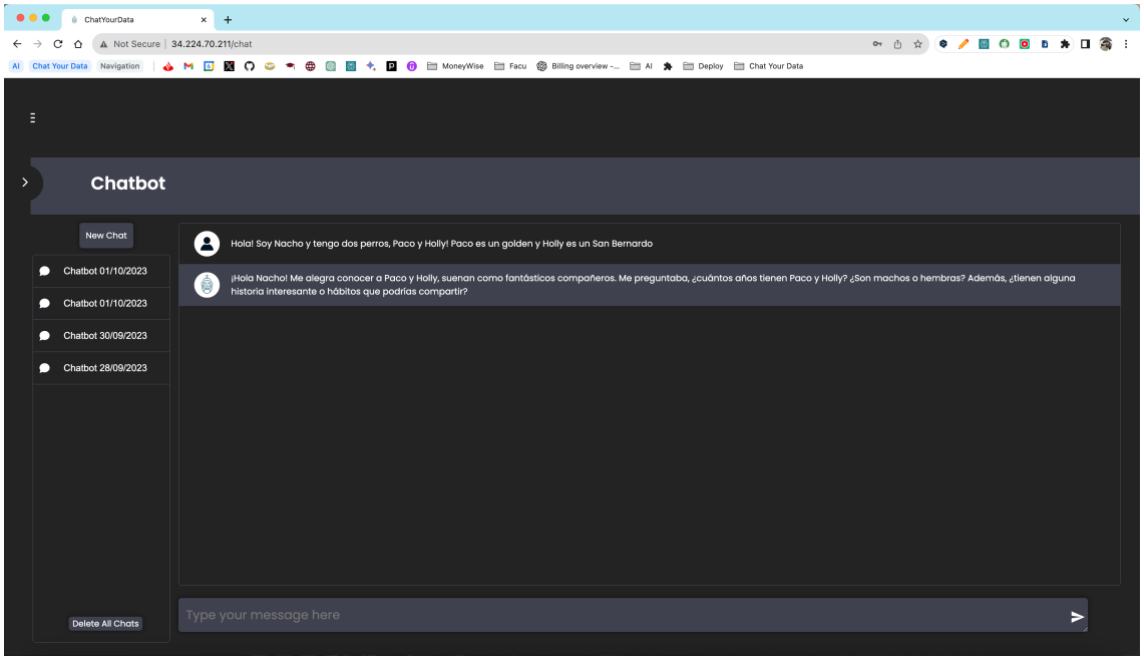


Ilustración 96 - Evidencia de Escritorio

13.36.2. Móvil

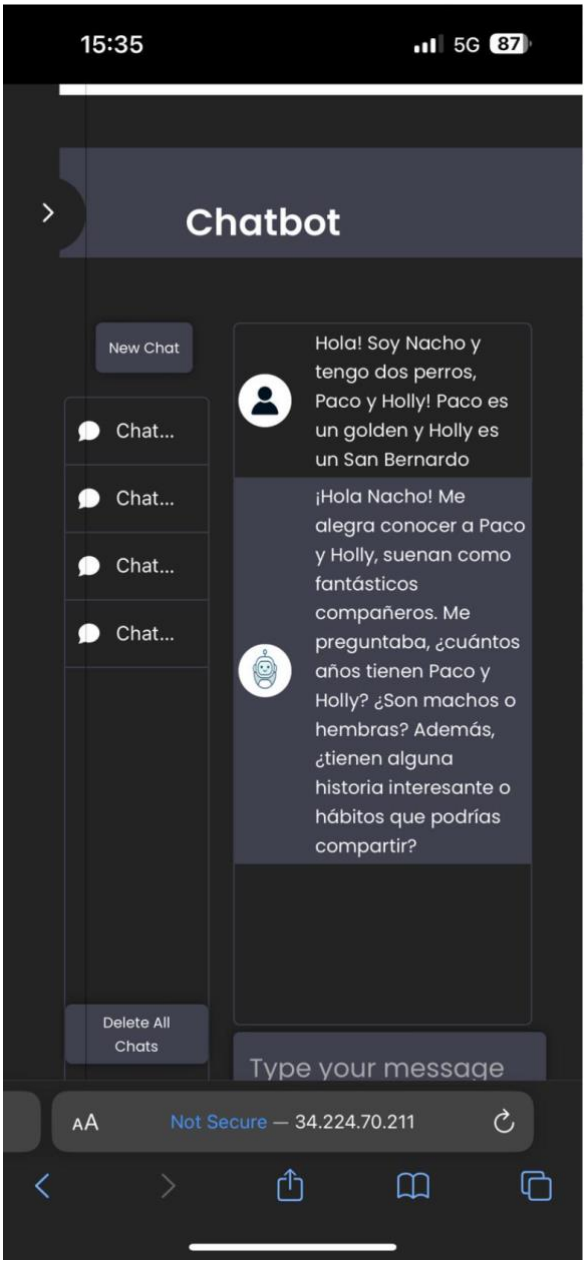


Ilustración 97 - Evidencia de Móvil

13.36.3. Evidencia *tablet*

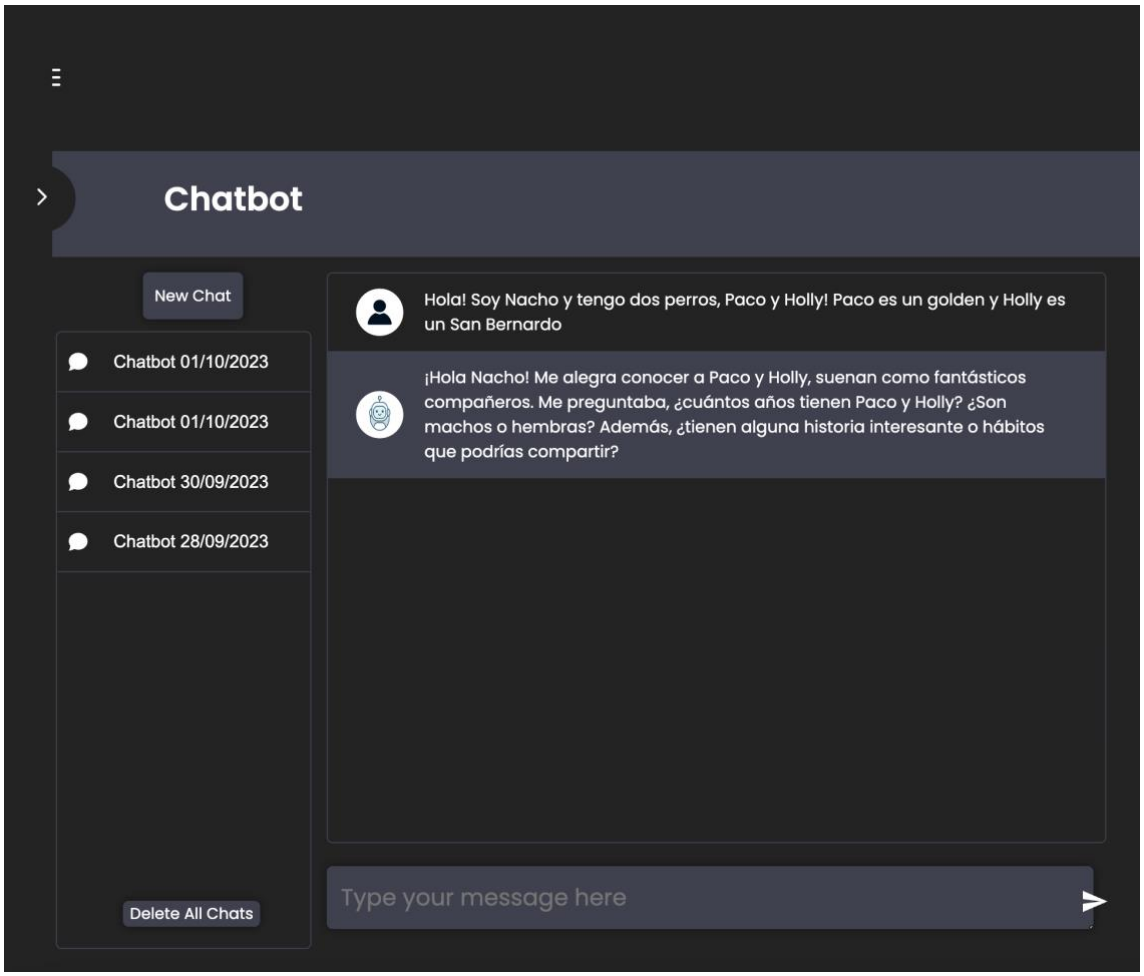


Ilustración 98 - Evidencia *Tablet*

13.37. Plan de calidad

Para el plan de calidad, se decidió poner objetivos, para cada etapa del proyecto, definiendo las actividades, participantes y el responsable.

Etapa	**Actividad**	**Producto- Resultado**	**Responsable**
Investigación	Identificación, exploración y validación del problema	Documento de problemática validada y confirmada	Equipo
Investigación	Aprendizaje e investigación en inteligencia artificial	Documento con nuevas herramientas,	Equipo

		tecnologías y conceptos asimilados	

Etapa	Actividad	Producto- Resultado	Responsable
Desarrollo del producto – planificación	Actividades de relevamiento	Requerimientos	Ingeniero de requerimientos
desarrollo del producto - planificación	Prototipado	Requerimientos validados	Ingeniero de requerimientos
Desarrollo del producto- planificación	Descripción de requerimientos	*Product Backlog*	Ingeniero de requerimientos
Desarrollo del producto - planificación	Priorización de requerimientos	*Product Backlog*	Ingeniero de requerimientos
Desarrollo del producto - planificación	Diseño de la arquitectura	Documento de arquitectura	Arquitecto
Desarrollo del producto - planificación	Análisis de requerimientos no funcionales	Atributos de calidad	Arquitecto
Desarrollo del producto - planificación	Decisión de tecnologías	Selección de las herramientas	Arquitecto
Desarrollo del producto - planificación	Diseño de la solución	Documento de diseño	Arquitecto
Desarrollo del producto - planificación	Patrones y buenas prácticas	Documento de diseño	Arquitecto
Desarrollo del producto - planificación	Interacciones entre los módulos	Diagramas	Arquitecto

Desarrollo del producto - ejecución	Revisión de código	*pull request reviews*	Desarolladores

Etapa	Actividad	Producto- Resultado	Responsable
Pruebas	Probar y validar	Documento de evidencia	Desarrolladores
Pruebas	*Testing*	Plan de *testing*	Desarrolladores

Tabla 17 - Plan de calidad de las etapas