Sistema de Recuperación Aumentada por Generación (RAG) para la optimización de la gestión y consulta de documentos en Fintic

Simón Sánchez Maya

Asesor

Andrés Felipe Hernández Giraldo

Universidad Nacional Abierta y a Distancia UNAD

Escuela de Ciencias Básicas, Tecnología e Ingeniería ECBTI

Especialización en Ciencia de Datos y Analítica

2026

Resumen

El presente proyecto de grado propone el diseño e implementación de un sistema de Recuperación Aumentada por Generación (RAG) para mejorar la gestión y consulta de la información documental en Fintic S.A.S. La creciente complejidad y volumen de datos en el entorno empresarial actual demandan soluciones eficientes que permitan el acceso rápido y seguro a la información relevante. Este sistema RAG combinará la capacidad de recuperar información precisa de una base de conocimiento documental interna con la habilidad de generar respuestas coherentes y contextualizadas, utilizando modelos de lenguaje de gran escala (LLMs). La arquitectura propuesta incluirá un portal web con una interfaz de chat intuitiva, una API intermedia para la orquestación de las solicitudes y la generación de *embeddings*, y el motor RAG encargado de la recuperación y generación. Se espera que esta implementación optimice los procesos de búsqueda, reduzca los tiempos de consulta y mejore la toma de decisiones al proporcionar acceso ágil y preciso a la información empresarial.

Evitar o disminuir las alucinaciones de los LLM y mantener la seguridad corporativa al no exponer información confidencial hacia modelos públicos, son objetivos primarios de las empresas que están en un proceso de adopción de la inteligencia artificial como herramienta de uso colaborativo. Para esto, el uso combinado de RAGs y LLMs como Gemini, BM25, entre otros, demuestran rendimientos costo eficientes en la extracción de información directamente desde los documentos. (Solano Cohen, 2023)

***Palabras clave*:** RAG (Retrieval Augmented Generation), Gestión Documental, Inteligencia Artificial, Modelos de Lenguaje (LLMs), Bases de Datos Vectoriales.

Abstract

This graduation project proposes the design and implementation of a Retrieval Augmented Generation (RAG) system to enhance document management and information retrieval within Fintic S.A.S. The increasing complexity and volume of data in today's business environment demand efficient solutions for quick and secure access to relevant information. This RAG system will combine the ability to retrieve precise information from an internal document knowledge base with the capacity to generate coherent and contextualized responses, leveraging large language models (LLMs). The proposed architecture will include a web portal with an intuitive chat interface, an intermediary API for request orchestration and embedding generation, and the RAG engine responsible for retrieval and generation. This implementation is expected to optimize search processes, reduce query times, and improve decision-making by providing agile and accurate access to enterprise information.

Avoiding or minimizing LLM delusions and maintaining corporate security by not exposing confidential information to public models are primary objectives for companies adopting artificial intelligence as a collaborative tool. To achieve this, the combined use of RAGs and LLMs such as Gemini and BM25, among others, demonstrates cost-effective performance in extracting information directly from documents. (Solano Cohen, 2023)

Keywords**:** RAG (Retrieval Augmented Generation), Document Management, Artificial Intelligence, Language Models (LLMs), Vector Databases.

Tabla de Contenido

Lista de Tablas

Lista de Figuras

Introducción

En el dinámico y cada vez más complejo entorno empresarial actual, la información se ha consolidado como uno de los activos más valiosos para cualquier organización. La capacidad de acceder, gestionar y utilizar esta información de manera eficiente y segura es fundamental para la toma de decisiones estratégicas, la optimización de operaciones y el mantenimiento de una ventaja competitiva. Sin embargo, la proliferación de datos y documentos en diversos formatos y ubicaciones presenta un desafío significativo para muchas empresas, incluida Fintic S.A.S., donde la dispersión y el volumen de la información pueden dificultar la recuperación rápida y precisa de conocimiento relevante.

Este proyecto de grado ejecutado como un proyecto aplicado, aborda esta problemática mediante la propuesta de diseño e implementación de un Sistema de Recuperación Aumentada por Generación (RAG). A diferencia de los sistemas de búsqueda tradicionales, que a menudo se limitan a la recuperación de documentos completos, o de los modelos generativos puros, que pueden "alucinar" información, el enfoque RAG combina lo mejor de ambos mundos. Permitirá a los usuarios de Fintic S.A.S. realizar consultas en lenguaje natural y recibir respuestas coherentes y contextualizadas, fundamentadas directamente en la vasta base de conocimiento documental interna de la empresa.

El sistema se concebirá bajo una Arquitectura Orientada a Servicios (SOA), integrando un portal web intuitivo con un chat conversacional, una API intermedia para la orquestación inteligente de solicitudes y la generación de *embeddings*, y un robusto motor RAG. Este último será el cerebro que recuperará fragmentos de información altamente relevantes de los documentos de la empresa (sean operativos, contables, comerciales, etc.) y los utilizará para guiar a Modelos de Lenguaje de Gran Escala (LLMs) en la formulación de respuestas precisas.

La implementación de este sistema no solo busca optimizar los procesos internos de búsqueda y reducir drásticamente los tiempos de consulta, sino también garantizar la veracidad de la información y proteger los datos sensibles al evitar su exposición a plataformas de inteligencia artificial públicas. En última instancia, este proyecto aspira a transformar la manera en que Fintic S.A.S. interactúa con su propio conocimiento, empoderando a sus colaboradores con acceso instantáneo y seguro a la inteligencia colectiva de la organización.

Planteamiento del Problema

La empresa Fintic S.A.S., como muchas organizaciones modernas, gestiona un volumen creciente y complejo de información documental esencial para sus operaciones (aspectos contables, comerciales, operativos, legales, etc.). Actualmente, la gestión y el acceso a esta vasta base de conocimiento se realiza mediante sistemas de almacenamiento (Sharepoint y One Drive) y búsqueda tradicionales, los cuales presentan limitaciones significativas que impactan negativamente la eficiencia operativa y la toma de decisiones.

Al no tener aún una adopción clara de inteligencia artificial, el equipo administrativo y comercial se encuentra cargando el 70% de los documentos oficiales y confidenciales en diferentes modelos LLM públicos o en su versión de suscripción como ChatGPT y Gemini para apoyarse en gestiones derivadas a estos documentos.

Los sistemas de gestión y búsqueda actuales son ineficientes, pues se limitan a la búsqueda literal por palabras clave y no logran comprender la intención semántica de las consultas complejas de los usuarios, lo que resulta en una inversión del 40% de tiempo por parte de los colaboradores en la obtención, análisis y exploración de la información, por tanto representa una consecuente reducción en la eficiencia operativa . A esto se suma la necesidad crítica de garantizar la veracidad y la seguridad de los datos; la falta de un mecanismo interno inteligente obliga a considerar la exposición de información sensible a servicios de inteligencia artificial públicos, o a tolerar el riesgo de respuestas no verificadas o "alucinadas" si se utilizan modelos generativos no anclados al conocimiento propietario.

Pregunta de Investigación

¿Cómo pueden las empresas y organizaciones implementar soluciones de Inteligencia Artificial (IA), particularmente Grandes Modelos de Lenguaje (LLM) y Generación Aumentada

por Recuperación (RAG), para procesar y analizar grandes volúmenes de información de manera rápida, eficiente y rentable, asegurando al mismo tiempo la privacidad y veracidad de los datos, y optimizando los costos de implementación y operación?

Justificación

En el panorama empresarial actual, la información es un activo estratégico que impulsa la toma de decisiones, la eficiencia operativa y la innovación. En Fintic S.A.S., la creciente generación y acumulación de documentos de diversa índole (operativos, contables, comerciales, etc.) ha llevado a un volumen de información que, si bien es valioso, presenta desafíos significativos en términos de su gestión, acceso y aprovechamiento. Los métodos tradicionales de búsqueda y consulta se ven sobrepasados, resultando en pérdida de eficiencia operativa a causa del tiempo invertido por los colaboradores en buscar información específica se eleva, disminuyendo la productividad y retrasando la ejecución de tareas críticas. Así mismo, la dificultad en el acceso al conocimiento vital para la operación de la empresa se encuentra disperso y, en ocasiones, de difícil acceso, limitando la transferencia de conocimiento entre equipos y la capitalización de la experiencia interna. Finalmente, la vulnerabilidad y exposición de datos al utilizar los colaboradores herramientas de inteligencia artificial públicas para procesar y consultar información empresarial presenta un riesgo latente de exposición de datos sensibles, comprometiendo la seguridad y la privacidad de la información.

La implementación de un Sistema de Recuperación Aumentada por Generación (RAG) emerge como una solución innovadora y estratégica para abordar estas problemáticas, estas técnicas han demostrado ser eficaces para integrar información actualizada, mitigar las alucinaciones y mejorar la calidad de las respuestas, especialmente en dominios especializados (Wang, y otros, 2024). A diferencia de las soluciones de búsqueda convencionales, que simplemente devuelven documentos, o de los Modelos de Lenguaje de Gran Escala (LLMs) puros, que pueden generar respuestas plausibles pero potencialmente incorrectas ("alucinaciones"), el RAG combina lo mejor de ambos mundos. Este sistema no solo recuperará

información precisa y contextualizada directamente de la base de conocimiento interna de Fintic S.A.S., sino que también utilizará esa información para generar respuestas coherentes y fáciles de entender. En este contexto, la adopción de arquitecturas RAG de nivel empresarial permite transformar los datos no estructurados en decisiones estratégicas mediante una recuperación de conocimiento más precisa y contextual (Varma, Shivam, Natajaran, Banerjee, & Roy, 2025).

Objetivos

Objetivo General

Implementar un sistema de Recuperación Aumentada por Generación (RAG) que facilite la gestión, consulta y acceso rápido y seguro a la información contenida en los documentos de Fintic S.A.S, optimizando la eficiencia operativa y la toma de decisiones.

Objetivos Específicos

Categorizar las fuentes de información documental relevantes dentro de Fintic S.A.S que serán integradas al sistema RAG para conocer el tipo de datos y tipo de información a custodiar.

Diseñar la arquitectura técnica del sistema RAG, incluyendo el portal web, la API intermedia y el módulo central de recuperación y generación con lenguajes de programación y frameworks que permiten una escalabilidad del producto y una inversión en nube sostenible.

Implementar una base de datos vectorial para almacenar y gestionar eficientemente los embeddings generados, permitiendo búsquedas semánticas rápidas y precisas. Integrar LLMs con el módulo de recuperación para generar respuestas coherentes y contextualizadas basadas en la información recuperada de los documentos.

Desarrollar un ecosistema digital integral basado en arquitectura RAG, integrando una interfaz de usuario intuitiva, una API de alto rendimiento y una infraestructura de datos robusta, empleando tecnologías escalables que garanticen una experiencia de usuario fluida y una eficiencia de costos operativa en la nube.

Marco de Referencia

Estado del Arte

El desarrollo de sistemas de procesamiento de lenguaje natural (NLP) ha experimentado una evolución significativa hacia la integración de fuentes externas de conocimiento. La base de esta tendencia fue consolidada por Lewis P. quienes introdujeron el concepto de Retrieval-Augmented Generation (RAG). Este enfoque híbrido combina modelos paramétricos (como los modelos de lenguaje pre-entrenados) con componentes no paramétricos (memoria densa de vectores), permitiendo que los modelos de IA realicen tareas intensivas en conocimiento con mayor precisión que los modelos tradicionales (Lewis, y otros, 2020).

Con el auge de los Modelos de Lenguaje de Gran Escala (LLMs), la arquitectura RAG se ha estandarizado. Diferentes fuentes proporcionan una visión panorámica de esta evolución a través de un estudio exhaustivo, clasificando los sistemas RAG en paradigmas ingenuos (Naive), avanzados y modulares. Según estos autores, el enfoque ha pasado de una simple recuperación de documentos a procesos iterativos y dinámicos que mejoran la coherencia de las respuestas (Gao, y otros, 2023). En esta misma línea de optimización técnica, han centrado sus esfuerzos en la búsqueda de "mejores prácticas", evaluando diversos flujos de trabajo para determinar las configuraciones más eficientes en términos de recuperación y generación, estableciendo un estándar para la implementación industrial de estas herramientas (Wang, y otros, 2024).

La literatura reciente demuestra que el potencial de RAG trasciende el chat generalista, moviéndose hacia aplicaciones de nicho: En la inteligencia de negocios se presenta un nuevo concepto denominado Business-RAG, una implementación diseñada específicamente para la extracción de información estratégica en contextos empresariales, demostrando cómo la arquitectura RAG puede transformar datos corporativos no estructurados en insights accionables

(Arslan & Cruz, 2024). Bajo esta misma premisa de especialización, se ha explorado el uso de LLMs y técnicas de aprendizaje automático para la automatización de tareas críticas y evaluación de contenidos complejos, lo que representa una evolución en la precisión de los sistemas de respuesta automatizados (Aarathisree, Sarkar, Dammala, Panday, & Sharma, 2024).

Así mismo, en la detección de fraude y seguridad, se encuentra una aplicación innovadora donde extienden el uso de RAG a la detección de deepfakes de audio. Investigaciones demuestran que recuperar muestras de referencia puede mejorar drásticamente la capacidad de los modelos para distinguir entre voces reales y sintéticas (Kang, y otros, 2024).

En otras instancias, el sector gubernamental no se queda atrás. En el ámbito de la gobernanza de datos, exploran cómo la IA para recuperación de datos puede mejorar la experiencia del usuario en portales gubernamentales de datos abiertos, proponiendo reflexiones sobre la accesibilidad y la transparencia de la información pública (Teixeira de Oliveira, Nascimento Silva, & Mafra Pereira, 2024). Esta capacidad de adaptación resulta particularmente relevante para el sector público y las organizaciones con marcos normativos estrictos. Al respecto, se destaca que la implementación de técnicas RAG sobre modelos de lenguaje permite optimizar la extracción de conocimiento y la generación de documentos oficiales, garantizando que el contenido generado se adhiera estrictamente a las bases de datos de las entidades (Collado Alonso, 2024).

Marco Contextual

Fintic S.A.S es una empresa de desarrollo de software enfocada en LowCode, desarrollo personalizado y agentes de inteligencia artificial con 8 años de experiencia en el mercado. Es una empresa que aunque pequeña con 30 personas integradas en su equipo de trabajo, se enfrenta a

una situación compleja en la administración documental resultante en las negociaciones de proyecto o cotizaciones e incluso en los artefactos resultado del desarrollo de dichos proyectos.

Actualmente toda su administración documental se encuentra en Sharepoint, One Drive y correo electrónico generando dispersión entre el conocimiento empresarial generado. Los principales beneficiados de este sistema serán desde el área de dirección y gerencia hasta el equipo de operación, quienes son los que más precisan acceder a las fuentes de información de consulta y los que más usan inteligencia artificial con modelos públicos para trabajar información sensible de la organización.

Las tecnologías, lenguajes de programación y librerías que Fintic a lo largo del tiempo se ha experimentado, permiten tener un abanico de opciones autilizar en la implementación del sistema.

- Lenguaje de Programación: Python (por su robustez en ciencia de datos y PLN).
- Frameworks de Desarrollo Web (para el portal y la API):
- Frontend: React, Angular o Vue.js (se elegirá uno según la familiaridad del equipo y los requisitos).
- Backend (API): Flask o FastAPI (por su ligereza y eficiencia para APIs).
- Librerías de Procesamiento del Lenguaje Natural:
  - Hugging Face Transformers (para *embeddings* y la interacción con LLMs).
  - NLTK o SpaCy (para procesamiento básico de texto).
  - LangChain o LlamaIndex (para orquestación del RAG y manejo de *prompts*).
- Bases de Datos Vectoriales:
  - Opción 1: Pinecone por su capa gratuita extendida y facilidad de integración.

- Opción 2: Se deja como posibilidad de uso Supabase Vector Search o Mongo Vector Database, que en primera intancia son descartas por los pasos previos para instalación, desarrollo, mantenimiento y costos.

- Modelos de Lenguaje de Gran Escala (LLMs): Modelos de acceso abierto (e.g., Llama 3, Mistral) para despliegue local o en entornos controlados. APIs de modelos comerciales (e.g., OpenAI GPT series, Google Gemini) para complementar o como alternativa si la infraestructura lo permite y el presupuesto es viable.
- Contenedorización: Docker (para facilitar el despliegue y la gestión de dependencias).
- Control de Versiones: Git y GitHub (para la gestión colaborativa del código).

Marco Teórico

El marco teórico de este proyecto se fundamentará en los pilares de la Ciencia de Datos, el Procesamiento del Lenguaje Natural (PLN), los Modelos de Lenguaje de Gran Escala (LLMs) y los Sistemas de Recuperación de Información (SRI), con un enfoque particular en la arquitectura RAG.

La ciencia de datos es un campo interdisciplinario que utiliza métodos científicos, procesos, algoritmos y sistemas para extraer conocimiento o *insights* de datos estructurados y no estructurados. Para este proyecto, la ciencia de datos es fundamental en la preparación y procesamiento de los documentos, la generación y gestión de *embeddings*, y la evaluación del rendimiento del sistema RAG.

El PLN es una rama de la inteligencia artificial que se enfoca en la interacción entre las computadoras y el lenguaje humano. Conceptos clave incluyen:

- Tokenización: Proceso de dividir el texto en unidades más pequeñas (tokens).

- Normalización de texto: Convertir el texto a una forma estándar.
- Vectorización de texto (*Embeddings*): Transformar palabras o frases en vectores numéricos densos que capturan su significado semántico. Modelos como Word2Vec, GloVe y, más recientemente, los *embeddings* de modelos transformadores (ej., BERT, Sentence-BERT) son cruciales para representar la información documental de manera que sea comprensible para los algoritmos.

Los LLMs son redes neuronales profundas entrenadas en vastas cantidades de datos de texto para comprender, generar y manipular el lenguaje humano. Son el componente generativo del sistema RAG.

Los SRI (Sistemas de Recuperación de Información) son sistemas diseñados para buscar información relevante en grandes colecciones de documentos. Componentes importantes incluyen:

- Indexación: Proceso de organizar la información para facilitar la búsqueda.
- Modelos de recuperación: Algoritmos que determinan la relevancia de los documentos frente a una consulta (ej., TF-IDF, BM25, modelos basados en *embeddings*).
- Bases de Datos Vectoriales: Bases de datos especializadas para almacenar y buscar eficientemente vectores de alta dimensión, como los *embeddings* de texto. Ejemplos incluyen Pinecone, FAISS, Weaviate, Milvus.

El RAG es una arquitectura que combina la capacidad de recuperación de información de los SRI con la capacidad generativa de los LLMs. La justificación de su uso radica en:

- Mitigación de "alucinaciones": Al fundamentar las respuestas en documentos reales, se reduce la tendencia de los LLMs a generar información incorrecta o ficticia.

- Contextualización y relevancia: Asegura que las respuestas estén directamente relacionadas con la información disponible en la base de conocimiento interna de la empresa.
- Actualización de conocimiento: Permite que el sistema acceda a información más reciente que no estaba presente durante el entrenamiento inicial del LLM.

Marco Conceptual

- Generación Aumentada por Recuperación (RAG): Arquitectura de inteligencia artificial que optimiza la salida de un Modelo de Lenguaje de Gran Escala (LLM) consultando una base de conocimientos externa y confiable antes de generar una respuesta.
- Modelo de Lenguaje de Gran Escala (LLM): Algoritmo de aprendizaje profundo entrenado en conjuntos de datos masivos, capaz de reconocer, resumir, traducir, predecir y generar contenido de texto.
- Embeddings (Incrustaciones): Representaciones numéricas en forma de vectores que capturan el significado semántico de las palabras o documentos, permitiendo que las máquinas calculen la similitud entre ellos.
- Base de Datos Vectorial: Sistema de almacenamiento diseñado para indexar y buscar vectores de alta dimensionalidad de manera eficiente, utilizado en la fase de "recuperación" de RAG.
- Alucinación (en IA): Fenómeno en el cual un modelo de lenguaje genera información que parece coherente pero es fácticamente incorrecta o no tiene sustento en los datos de entrenamiento.

Marco Normativo

- Protección de Datos Personales.

- Ley 1581 de 2012: Es la norma general para la protección de datos personales en Colombia.
- Decreto 1377 de 2013: Reglamenta la ley anterior, estableciendo los requisitos para obtener el consentimiento informado del titular del dato y las políticas de tratamiento de la información.

- Habeas Data Financiero.
  - Ley 1266 de 2008: Esta ley regula el manejo de esos datos específicos acerca de información financiera, crediticia o comercial.

- Ética y Gobernanza de Inteligencia Artificial.
  - Marco Ético para la Inteligencia Artificial en Colombia (2020): Documento de política pública expedido por el Gobierno Nacional que establece lineamientos sobre transparencia, privacidad, seguridad y no discriminación en el desarrollo de sistemas de IA.
  - CONPES 3975 de 2019: Política Nacional para la Transformación Digital e Inteligencia Artificial, que busca fomentar el uso de estas tecnologías bajo estándares de seguridad y eficiencia.

- Seguridad de la Información.
  - Ley 1273 de 2009: Conocida como la "Ley de Delitos Informáticos". Protege la confidencialidad, la integridad y la disponibilidad de los datos y de los sistemas informáticos.

Metodologías

Para la ejecución del proyecto, se llevará a cabo una investigación aplicada partiendo de recursos de apoyo en el manejo de RAGs o la incorporación de herramientas como NotebookLLM en la adopción de procesos con IA en las organizaciones.

El enfoque del proyecto aplicado será mixto, ya que se desea impactar métricas cuantitativas y medibles acerca del comportamiento con el uso del sistema RAG a incorporar en los procesos de Fintic S.A.S, por otro lado, una toma cualitativa para la fase de requisitos (entrevistas a usuarios, análisis de flujos de trabajo) para la evaluación de la experiencia de usuario (UX) y la relevancia de las respuestas generadas.

Durante la ejecución del proyecto, se propone un desarrollo ágil e iterativo incorporando la metodología Kanban en 5 diferentes fases o etapas para lograr los objetivos en el tiempo estipulado en el cronograma de trabajo. La herramienta GitHub, será participe del proceso en la planeación de tareas y actividades, además de la custodia del código generado para la implementación. Las diferentes etapas que se prevén para la correcta puesta en marcha son:

Fase 1 Planificación y Recopilación de Requisitos

En esta fase se pretende recuperar por medio de entrevistas y sesiones conjuntas con el equipo de trabajo, requerimientos a alto nivel que permitan darle un alcance cerrado a la solución, teniendo en cuenta limitaciones de sistemas o supuestos que no se van a realizar por tener un tiempo acotado y que por tanto no pertenecen a los entregables de la solución. Las actividades principales son:

- Definición detallada de los documentos a integrar, extensiones y tipo de media. Se hace énfasis que documentos de tipo audio, video, imágenes, presentaciones de power point no

serán incluidos en la vectorización de datos o generación de *embeddings* por su complejidad de manejo y descomposición.

- Análisis de los flujos de trabajo actuales y necesidades de los usuarios.
- Selección definitiva de tecnologías y herramientas.

Fase 2 Diseño de Arquitectura y Base de Datos

Definir la cualidades arquitectónicas, de red y bases de datos bajo la cual se construye el sistema de información es el objetivo de esta fase, donde especificar la tecnología, servidores sugeridos, base de datos a usar son aspectos que ayudan a definir el tipo de recursos humanos y tecnológicos necesarios para darle mantenimiento o escalabilidad a la solución, además, de servir de guía para la puesta en producción y hacer una proyección de costos de uso.

- Diseño detallado de la API con documentación en Swagger.
- Diseño del esquema de la base de datos vectorial y la integración entre componentes.
- Diseño del proyecto del *Front-End*.
- Generación del diagrama de arquitectura de referencia de la solución.

Fase 3 Desarrollo Iterativo

Para asegurar el cumplimiento del cronograma se ejecutará el desarrollo del proyecto con entregas incrementales bajo la metodología ágil Kanban. Las principales historias de usuario que pueden ser desarrollas en esta fase son:

- Ingesta de Documentos: Implementación del pipeline de extracción, limpieza y generación de *embeddings*.
- Módulo de Recuperación: Desarrollo de la lógica de búsqueda en la base de datos vectorial.

- Módulo de Generación: Integración con LLMs y construcción de *prompts*.
- API Intermedia: Desarrollo de los *endpoints* para la comunicación entre el *frontend* y el *backend* RAG.
- Interfaz de Usuario (Chat): Desarrollo del frontend interactivo.

Fase 4 Pruebas y Evaluación

Para garantizar un desarrollo seguro manteniendo lineamientos y estándares internacionales como OWASP, se ejecutarán las siguientes actividades durante la implementación de código y entrega del proyecto, sin embargo, estas no están sujetas a documentación ni uso de herramientas externas como SonarQ que puedan generar un alza en los costos de implementación:

- Pruebas unitarias, de integración y funcionales.
- Evaluación de la calidad de las respuestas generadas (precisión, coherencia, relevancia).
- Pruebas de rendimiento y escalabilidad.

Para esta fase, se tendrán en cuenta las siguientes métricas de calidad asegurando que lo implementación apunta a la optimización de tiempo en los procesos operativos de consulta en documentos, seguridad y confidencialidad de la información, rendimiento y costos. Estas métricas si serán artefacto de entrega como conclusión y evidencia de los aportes de la solución a la organización.

- Precisión de recuperación. Mide la proporción de fragmentos de documentos recuperados que son verdaderamente relevantes para la consulta del usuario. Número entero con la cantidad de vectores usados.

- Coherencia. Evalúa la fluidez, la gramática y la estructura lógica de la respuesta generada. Porcentaje de cumplimiento medido contra una IA.
- Relevancia contextual. Mide si la respuesta final aborda directamente y satisface la necesidad de información planteada en la consulta original. Medición cualitativa en set de pruebas con usuarios finales.
- Tiempo de respuesta. El tiempo transcurrido desde que el usuario envía la consulta hasta que recibe la respuesta final. Milisegundos medidos con herramientas de inspección de red.
- Tasa de acierto. Número de consultas que el sistema puede procesar por unidad de tiempo. Número de consultas por segundo.
- Reducción del tiempo de búsqueda. Medir el tiempo que tomaba a los empleados encontrar una respuesta clave usando los métodos anteriores vs. el tiempo que toma con el sistema RAG. Entregar una expresión en porcentaje del tiempo reducido.
- Tiempo de generación de *embeddings*. Tiempo en segundos que toma la generación de información vectorizada y guardada en la base de datos.

Fase 5 Documentación y Despliegue

La transferencia de conocimiento será parte del proyecto aplicado ya que es una entrega funcional a la empresa Fintic S.A.S contemplando las siguientes actividades:

- Elaboración de la documentación técnica.
- Orientación en las sugerencias de despliegue del sistema en un entorno de producción o pre-producción.
- No contemplará manual de usuario y funcional. Solo abordará aspectos técnicos.

Resultados

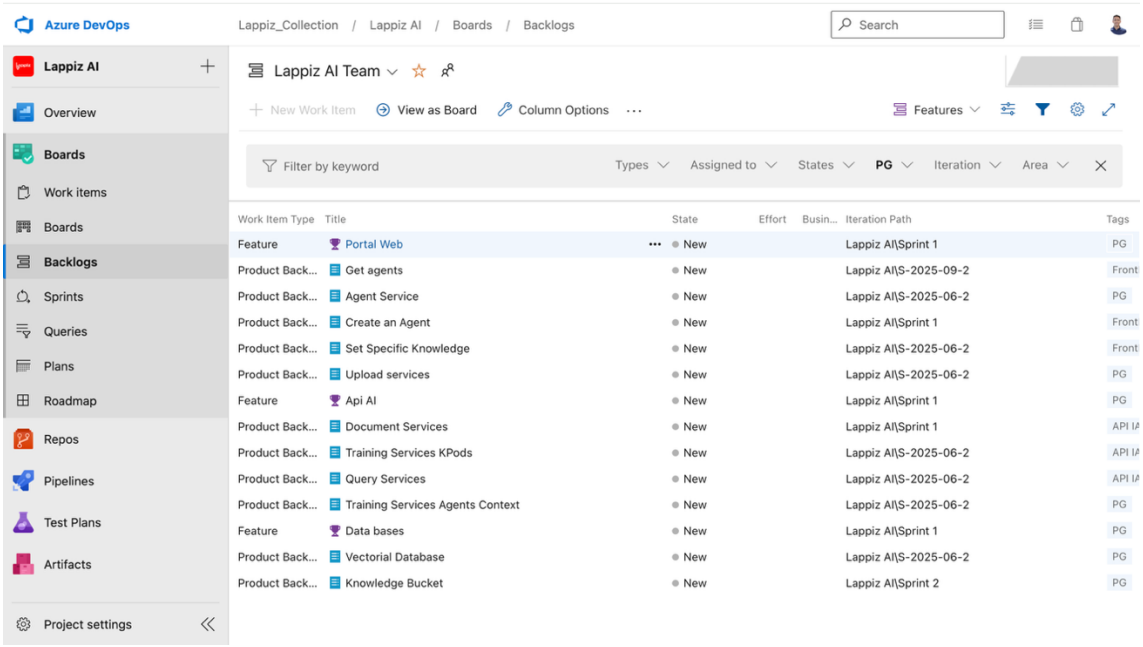
Categorizar las Fuentes de Información Documental

El objetivo de este proceso alineado con la fase de levantamiento de requerimientos hace uso de Azure DevOps como herramienta para la custodia de las historias de usuario básicas en la definición del alcance de la plataforma. Este proceso de levantamiento de requerimientos genera criterios de aceptación importantes para el acotamiento del alcance y futura escalabilidad.

Dentro de los requerimientos se consolidan características para la carga documental en el ecosistema RAG, para el cual Fintic proveerá un 30% de documentos en el caso de estudio aplicado. De este porcentaje se aglomeran temas relacionados con propuestas comerciales elaboradas por el equipo, propuestas confidenciales de participación en licitaciones, presentaciones acerca de la empresa y documentación técnica de proyectos y plataforma. La decisión de tomar estos tipos de documentos se fundamente en el ánimo de orientar el sistema de información y ecosistema RAG a la creación inicial de un agente que permita correlacionar información técnica, propuestas pasadas y licitaciones actuales sirviendo de apoyo al equipo comercial en la gestión de nuevas oportunidades y construcción del contenido comercial y económico apoyado en conocimiento previo y transversal a este proceso del área. Todo el roadmap del proyecto se crea y gestiona en Azure Devops como se muestra en la siguiente figura.

Figura 1

Backlog Planeado

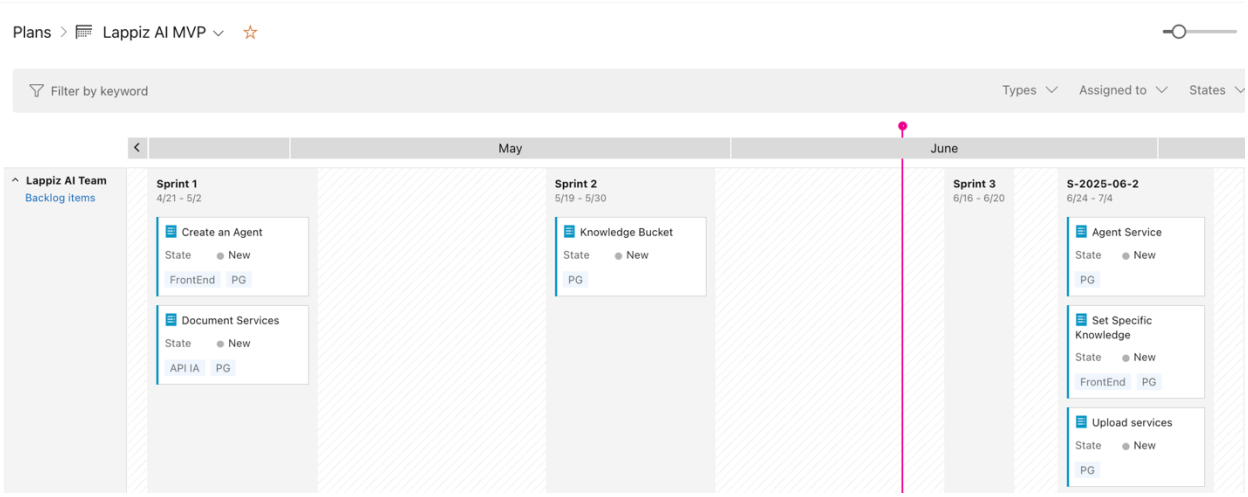


Épica de Portal Web

En la figura 2, se muestra el backlog definido para la épica de portal web, compuesta por 5 historias de usuario. Dichas historias recopilan los requerimientos para crear un agente, listarlos y chatear con estos. Así mismo, se define el proceso para cargar conocimiento a los agentes por medio de texto o cargando archivos planos en formato txt u otras extensiones como PDF o docx que contienen las instrucciones o información relevante para el agente de inteligencia artificial (para el alcance inicial de la plataforma no se admitirá otro formato diferente a txt, pdf, docx).

Figura 2

Historias de Usuario, Portal Web

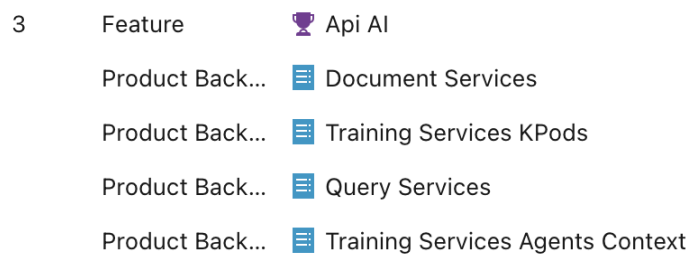


Épica de API AI

Compuesta por 4 historias de usuario como se explica en la figura 3. Estás comprenden las funcionalidades y mecanismos para cargar los archivos a la nube de AWS S3 donde tendrán su custodia documental y que, posteriormente, serán usados para el proceso de generación de *embeddings* y creación de la base de datos vectorial y optimizada para el contexto del agente de inteligencia artificial. Esta API usará como LLM principal Gemini.

Figura 3

Backlog API RAG

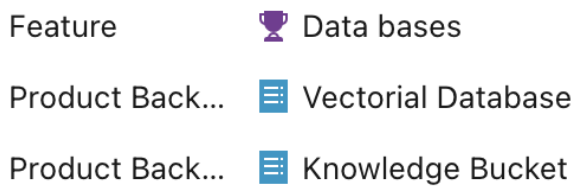


Épica de Data Bases

Recopila dos historias de usuario que se basan en las consideraciones para centralizar los documentos corporativos en un único storage (AWS S3) y la base de datos en SQL Server que relaciona estos documentos con los datos de la información de los agentes, así mismo, la creación de bases de datos vectoriales con la herramienta Pinecone como se enseña en la figura 4. Estas dos fuentes de datos son primordiales para guardar los documentos que serán dispuestos para el RAG de los agentes de preguntas y respuestas.

Figura 4

Backlog Bases de Datos



Diseño de la Arquitectura Técnica del Sistema RAG, Portal Web y API

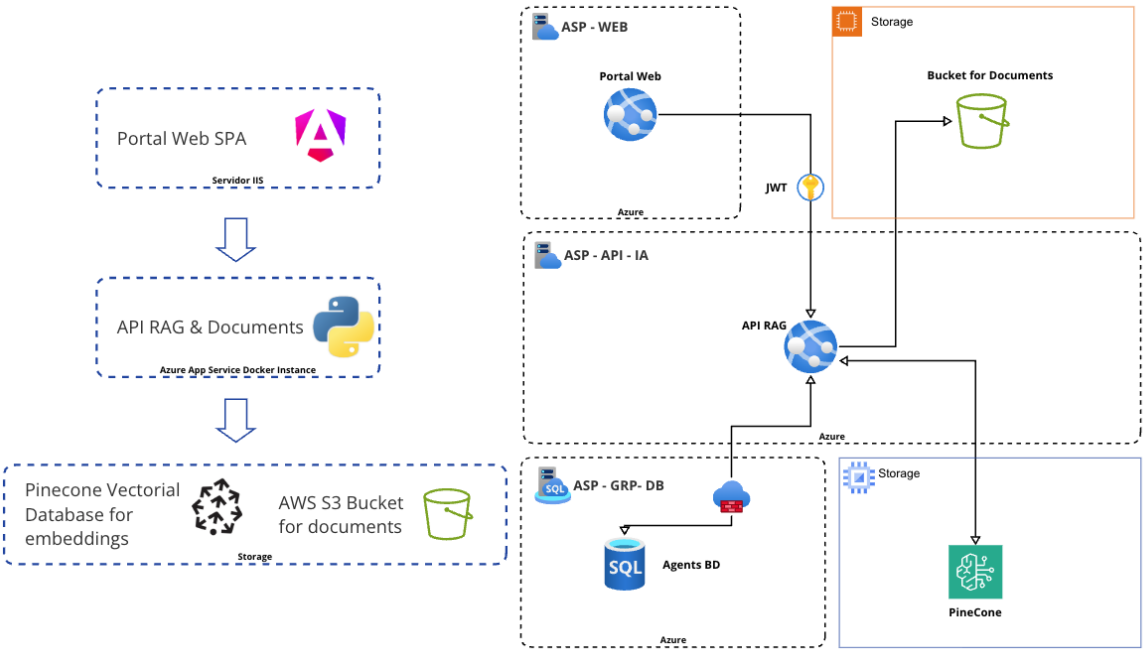
La arquitectura de referencia definida para el proyecto consiste en una arquitectura orientada a servicios. Esta tiene 3 componentes fundamentales, portal web, portal api y almacenamiento. Para cada uno de estos pilares se manejan tecnologías y frameworks que corren en un modelo de nube híbrido y de base gratuita o pay as you go económico.

Diagrama de Arquitectura

Con la finalidad de servir de guía para la representación de componentes del sistema de información, se crea el diagrama de arquitectura como se muestra en la figura 5, el cual abarca la tecnología usada, seguridad básica de red y la comunicación entre los diferentes artefactos de código y servicios de nube dispuestos para el proyecto aplicado.

Figura 5

Diagrama Arquitectura Alto Nivel



El modelo de nube es híbrido donde se usa Azure para correr los servicios de Portal web (SPA Angular), API RAG y documentos (Docker con Python y Fast API) y una base de datos relacional con el servicio de Azure SQL, la nube de AWS es usada para el aprovisionamiento del servicio de AWS S3. Finalmente, los servicios de compute engine de Google Cloud Platform (o AWS como alternativa) son usados para las instancias de bases de datos vectoriales.

Selección de Tecnología

Para darle cumplimiento al alcance del proyecto se define el siguiente stack de tecnologías para los diferentes frentes de la plataforma:

- Portal Web. Aplicación SPA con Angular 19 TypeScript. Este servicio es la interfaz gráfica con la que el usuario final interactúa para acceder a las funcionalidades de los agentes.

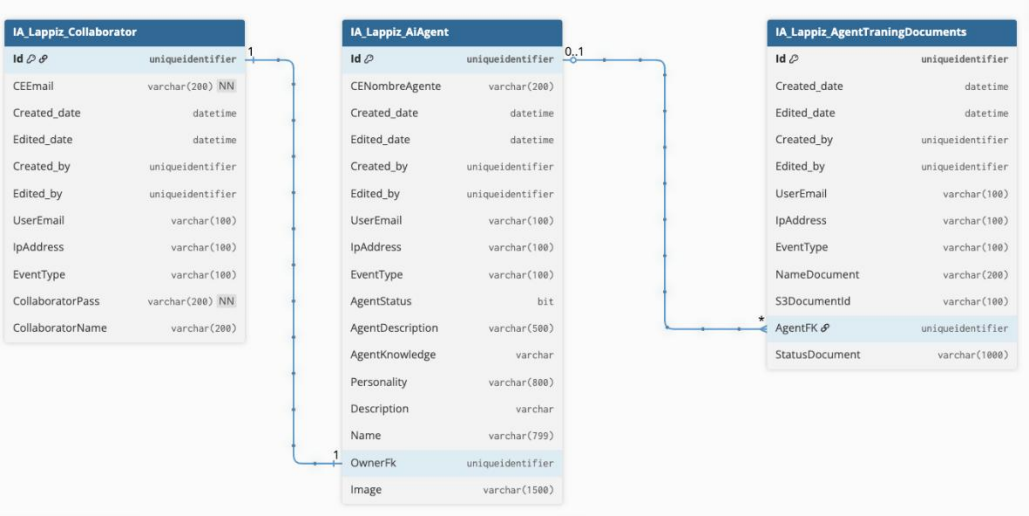
- Api RAG. Interfaz construida con Python y FastAPI. Servicio con endpoints para gestionar las diferentes reglas de negocio como la administración de archivos (documentos), generación de Embeddings, conexión con la base de datos vectorial y el LLM.
- Almacenamiento de datos vectorial. Se define el uso de Pinecone como base de datos de indexación de los documentos a consultar por el LLM.
- Almacenamiento documental. Se define AWS S3 como gestor de almacenamiento de archivos por su costo ligero y escalabilidad futura del sistema con herramientas como Amazon Bedrock.

Diseño de Base de Datos Relacional

El siguiente modelo relacional mostrado en la figura 6, permite verificar las diferentes tablas y columnas que custodiarán la información de los agentes, su configuración y documentos relacionados al conocimiento.

Figura 6

Modelo Relacional



La entidad de IA_Lappiz_Collaborator, es la tabla que guarda la información del colaborador para hacer login dentro del sistema y que después será relacionado con los agentes que crea.

La entidad IA_Lappiz_AiAgent, hace referencia a la tabla que guarda la configuración del agente que crea el colaborador otorgando características como nombre al agente, una descripción o propósito, un conocimiento general (por medio de texto) y que, posteriormente se le puede asociar documentos o información con la que será gestionado su RAG y conocimiento con conexión al LLM.

IA_Lappiz_AgentTrainingDocuments, es la entidad que se encarga de relacionar los documentos con la ruta e identificación con la que son guardadas en el bucket de S3. Esta tabla es consultada para obtener la ruta del documento en el almacenamiento externo que son objetos o artefactos por procesar para generar los embeddings y alimentar la base de datos vectorial.

De forma general, para todas las tablas se guarda la información relevante a logs de quién efectúa alguna transacción en la plataforma, como es el caso de inserción o edición de registros.

Implementación de Base de Datos Vectorial

Diseño de Base de Datos Vectorial

Por medio de un código optimizado en Python, se consultan los documentos guardados en el bucket de S3 correspondiente al agente de IA y se genera los embeddings que serán guardados en una base de datos indexada en el motor de Pinecone.

```
def generate_embeddings(docs: list[Document], main_folder: str):
    print(f"")

    """
    Genera y actualiza embeddings para todos los documentos en S3.
    Sobrescribe el contenido existente en el índice Pinecone.
    """
    if not docs:
```

```
    return {
        "embedResult": "error",
        "message": "No hay documentos para procesar"
    }

# Inicializar embeddings y obtener dimensión
embeddings_model = GoogleGenerativeAIEmbeddings(model="models/embedding-001")
embedding_dim = len(embeddings_model.embed_query("test"))

# Crear índice si no existe - agregar nombre del index
index_name = normalize_name(main_folder) if main_folder else index_name
if index_name not in pinecone.list_indexes().names():
    pinecone.create_index(
        name=index_name,
        dimension=embedding_dim,
        metric="cosine",
        spec=ServerlessSpec(cloud=PINECONE_CLOUD, region=PINECONE_REGION)
    )

index = pinecone.Index(index_name)

# Generar embeddings y preparar datos
upsert_data = []
for i, doc in enumerate(docs):
    vector = embeddings_model.embed_query(doc.page_content)
    unique_id = str(uuid.uuid4())
    upsert_data.append((
        unique_id,
        vector,
        {
            "source": doc.metadata["source"],
            "content": doc.page_content
        }
    ))


# Subir por batches para rendimiento
BATCH_SIZE = 100
for i in range(0, len(upsert_data), BATCH_SIZE):
    batch = upsert_data[i:i + BATCH_SIZE]
    index.upsert(vectors=batch)

return {
    "embedResult": "success",
    "message": f"Índice {index_name} actualizado con {len(docs)} fragmentos."
}
```

En el procesamiento de documentos es susceptible a generación de índices y vectores ineficientes (ya que variables presentes como extensión, peso y caracteres usados inciden en esta lógica), al implementar el modelo embedding-001 de Google, garantiza que el contenido textual se transforme en representaciones vectoriales de alta dimensionalidad para una búsqueda por significado. Por otro lado, el método no solo sube los documentos, sino que también separa un índice por agente (para que el conocimiento no sea transversal sino indexado de acuerdo con el grupo de documentos que se le cargan). Finalmente, el uso de un esquema de carga por lotes (Batching) demuestra una arquitectura pensada para el rendimiento, evitando el agotamiento de memoria y optimizando las llamadas de red.

De otra forma, se detectan oportunidades de mejoras como incorporar una estrategia de Chunking o fragmentación, ya que documentos muy largos pueden generar que el embedding pierda precisión. Del mismo modo, agregar un método de reintento ya que las APIs de Google y Pinecone pueden generar una excepción. No menos relevante si existe un incremento exponencial en el uso de la herramienta y carga de documentos por diferentes usuarios del sistema, se recomienda implementar paralelismo, es decir, generar los embeddings en un proceso simultáneo reduciendo el tiempo total de ejecución.

Desarrollo de un Ecosistema Digital Integral

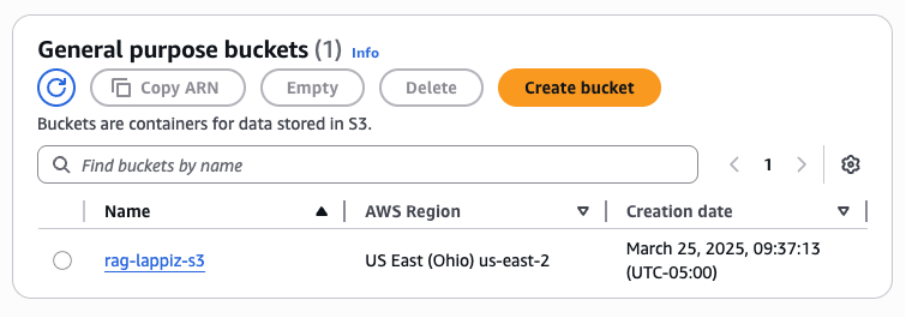
Una vez definido el alcance del proyecto aplicado y la tecnología a usar, se ejecuta el cronograma de desarrollo con la metodología de trabajo Kanban en la herramienta de Azure DevOps. Todo el repositorio de código generado como resultado del producto se encuentra en el repositorio privado del siguiente enlace.

Configuración de AWS S3

Se aprovisiona en la región east-2 un bucket S3 y se genera una API Key para la conexión desde el API que se encarga de subir, eliminar y consultar los documentos de interés de la compañía, como se muestra en la figura 7 el bucket se nombra como rag-lappiz-s3.

Figura 7

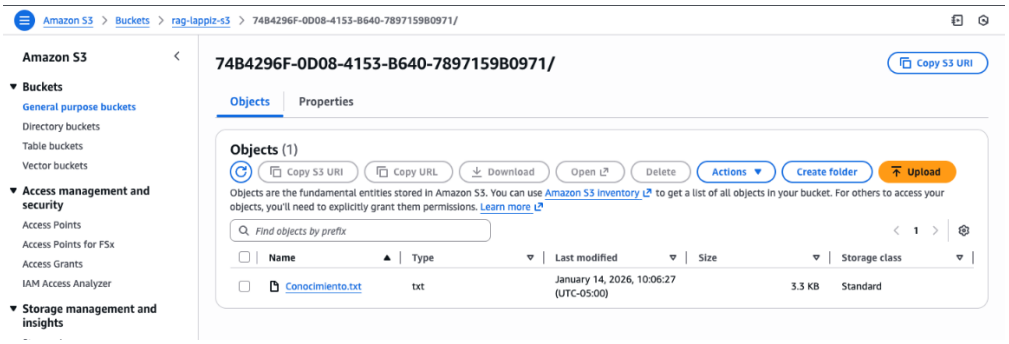
Bucket AWS S3



Para soportar una gestión documental ordenada, se propone segmentar las carpetas o rutas donde se guardan los documentos con una carpeta principal nombrada con el id del agente. En la figura 8 se nota que la carpeta principal tiene un UUID de nombre, dentro de esta, son almacenados los documentos que se relacionan al conocimiento que tendrá dicho agente.

Figura 8

Documentos Agrupados por Id de Agente

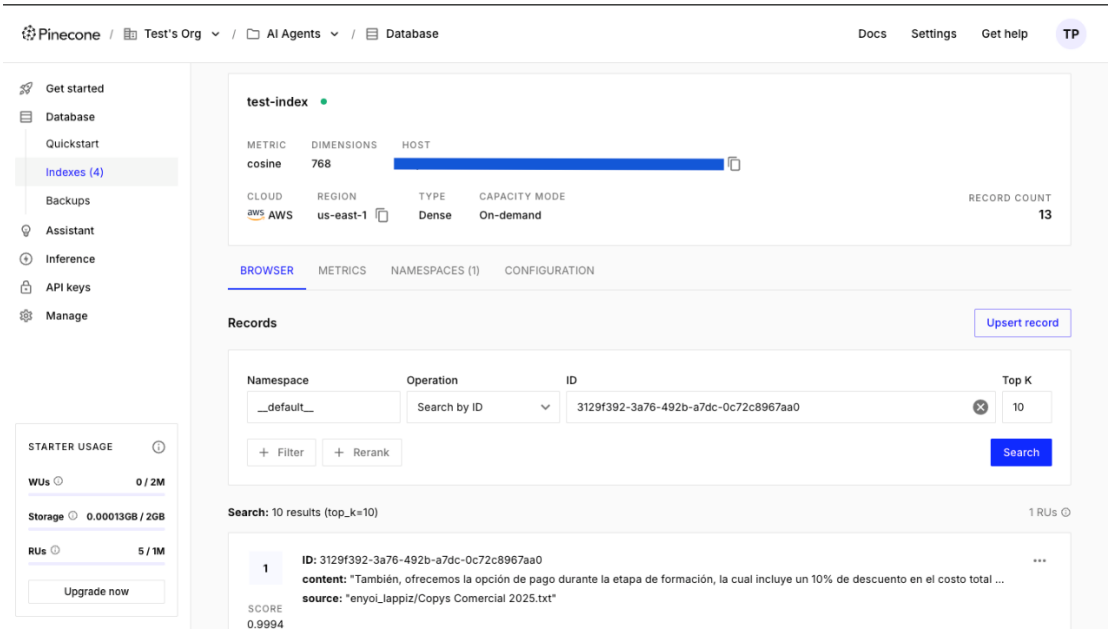


Configuración y Conexión con Pinecone

Para disponibilizar el servicio de Pinecone se crea un servidor en la capa gratuita con la configuración ligera y optimizada con la finalidad de aprovechar toda la capacidad del tier como se muestra en la siguiente figura 9. Se debe tener en cuenta que Pinecone ofrece hasta 5 índices gratuitos.

Figura 9

Servidor y Base de Datos Vectorial Indexada



En este tier, se crea una base de datos indexada con el contenido generado de embeddings explicado en el numeral “diseño de bases vectoriales” adicionando el id del agente como nombre principal de índice.

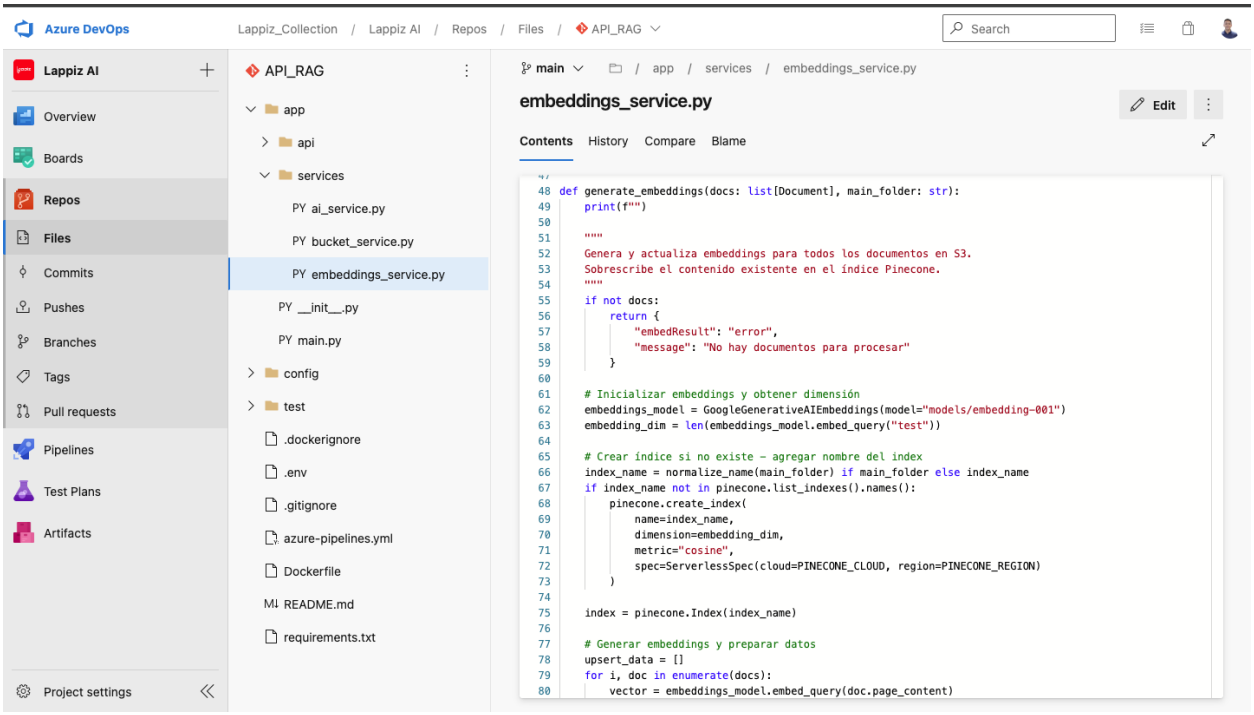
Creación de API con Python

Se crea un proyecto Python con el framework FastAPI para la habilitación de un servicio API para la gestión de integraciones y lógica necesaria para las funcionalidades de carga de

archivo y creación de la BD Vectorial. La biblioteca principal para la lógica del RAG y su funcionamiento se basa en las funciones expuestas por langchain. El código por seguridad de la organización se aloja en Azure DevOps Repos como se muestra en la figura a continuación.

Figura 10

Repositorio Código API RAG

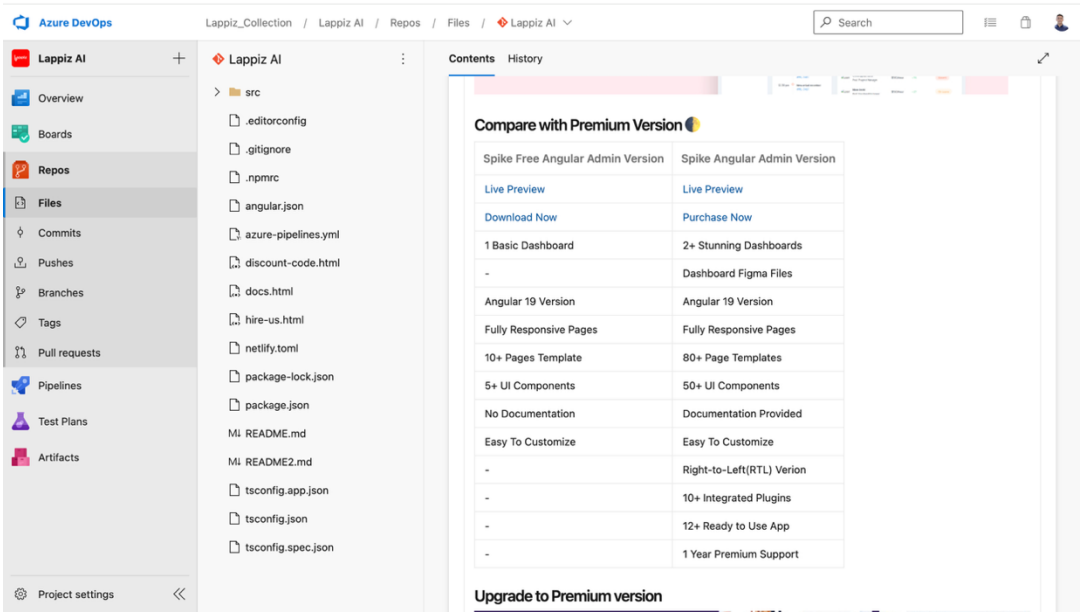


Creación de SPA con Angular

Como se muestra en la figura 11, se crea un proyecto SPA con Angular y una estructura basada en Screaming Arquitecture.

Figura 11

Repositorio de Código SPA Angular



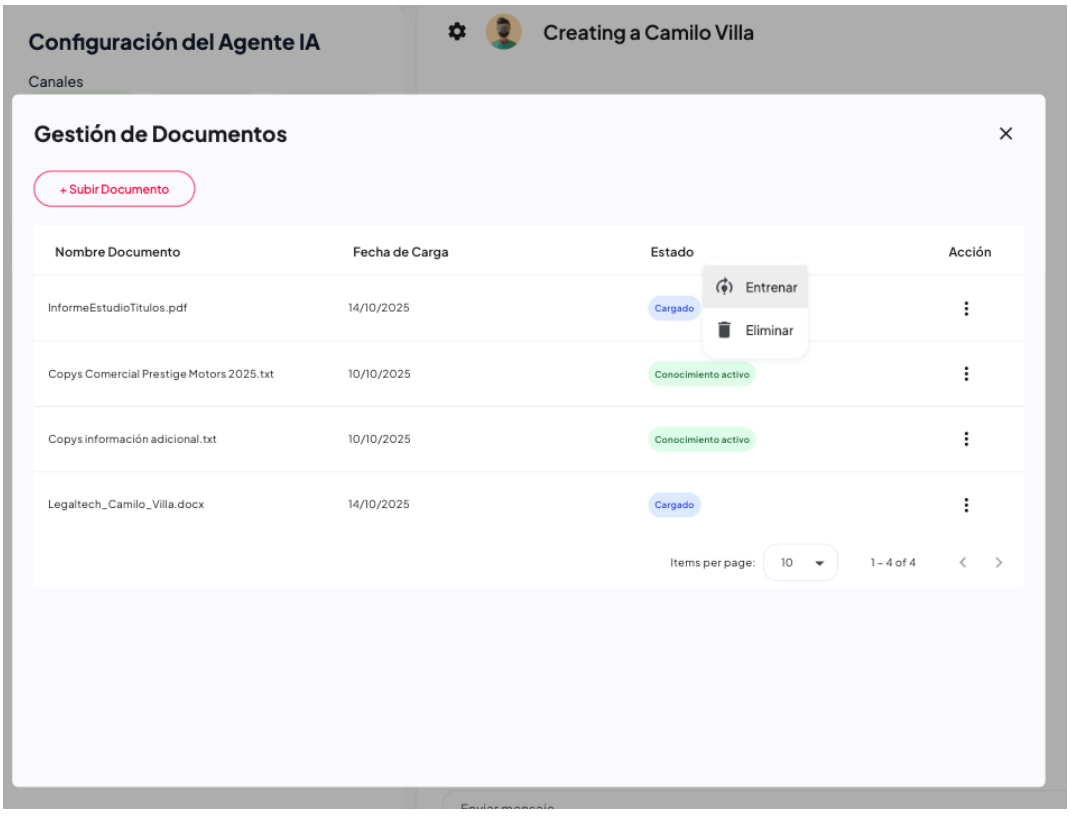
Pruebas y Evaluación

Métrica de Generación de Conocimiento

En este módulo se carga el archivo txt, pdf o docx con la información que el colaborador desea optimizar y guardar en la memoria del agente. Una vez cargada se puede generar los embeddings de conocimiento o removerlos si ya se encuentra ese conocimiento en uso (pero se volvió obsoleto para la organización). En la figura 12 se puede observar que para la generación de conocimiento se ingresa por el apartado de “Gestión de documentos” en la configuración del agente desde la interfaz gráfica diseñada para que los usuarios puedan interactuar de forma fluida.

Figura 12

Interfaz Gráfica para Carga de Documentos y Solicitud de Entrenamiento



La generación de embeddings por documento está tomando un promedio de tiempo de 500 ms a 1 s para archivos con un peso de 1 kb. Esto quiere decir que la creación de bases indexadas para los agentes con una cantidad de 70 documentos que consulta recurrentemente la empresa por área (con información comercial, de contratos, fichas técnicas, propuestas comerciales u otros) y con un peso promedio de 10 kb tienen el siguiente comportamiento promedio:

- 1 KB aproximadamente 500 ms - 1000 ms.
- Al pasar de 1 KB a 10 KB, el modelo debe procesar 10 veces más tokens, estimando así entonces entre 5 y 10 segundos por documento.
- Total para 70 documentos por área:
  - Mínimo: $70 x 5 s = 350 segundos (5.8 minutos).
  - Máximo: $70 x 10 s = 700 segundos (11.6 minutos).

Estos indicadores demuestran que la configuración inicial de los agentes con la información promedio actual por área podría tardar un máximo de 12 minutos, permitiendo a la empresa disminuir la cantidad de correos a cruzar para compartir información entre colaboradores. Por otro lado, se reduce al 80% la exposición de los documentos de la empresa a modelos LLM públicos ya que al adoptar esta política dentro de la organización la información queda almacenada en instancias privadas y seguras de acuerdo con la arquitectura de referencia. El 20% de los documentos restantes son de carácter público como políticas de tratamiento de datos, ANS de atención entre otras.

Métrica de Generación de Respuesta

Una vez se encuentran generadas las bases de datos de conocimiento, se genera un método en el API RAG que permite la consulta hacia la base de datos vectorial con intermediación del LLM para la recuperación de datos.

Para esto se habilitó en el portal web un chat flotante el cual permite hacer consultas en lenguaje natural, esta se suministra como argumento al endpoint del API que usa la técnica RAG para la recuperación de información específica y actualizada de los documentos.

En pruebas funcionales, se sometió al agente a escenarios controlados de información donde se le proporcionó como conocimiento un txt, pdf o docx con una serie de instrucciones y pasos que ejecuta el equipo de TIC de Fintic para reestablecer el MFA de la empresa a los colaboradores, entre otras más instrucciones. El input solicitado fue “Dime cual es el protocolo para configurar el authenticator en mi dispositivo”, obteniendo una respuesta que como se muestra en la tabla a continuación, presenta las siguientes características de coherencia y cohesión:

Tabla 1

Características Cualitativas de las Respuestas del Sistema LLM y RAG

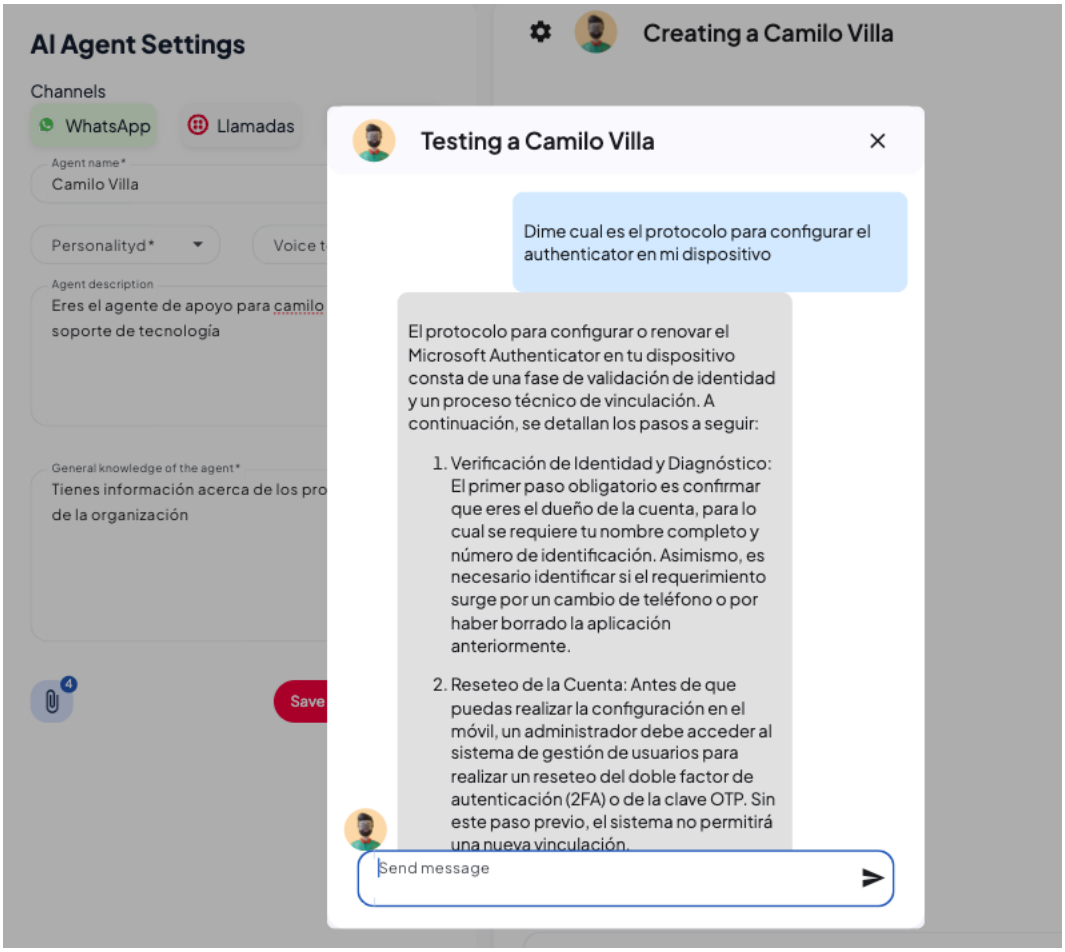
Aspecto	Nivel de Error	Observación
Contenido	0%	No se inventaron pasos adicionales ni se omitió información crítica del protocolo original.
Terminología	0%	Utiliza los términos correctos como "OTP", "2FA" y "Microsoft Authenticator" tal como aparecen en la guía,
Contexto	Bajo	La IA redactó la respuesta como un procedimiento general. El único matiz es que el texto original parece ser una guía para el personal de soporte (ej. "Pregúntale al usuario"), mientras que la respuesta de la IA fue adaptada para responder directamente al usuario final.

Nota. Esta tabla muestra los resultados arrojados al medir las métricas cualitativas durante las pruebas manuales del sistema RAG.

La figura 13 muestra que en el portal web se ha desarrollado un chat para el colaborador. Este chat sirve para una interacción con el ecosistema RAG ya que al ser un sistema apificado debe tener una interfaz intuitiva para su uso.

Figura 13

Interfaz de Web Modo Chat para Interactuar con LLM y RAG



.

Las respuestas del flujo de conexión entre el LLM, la base de datos vectorial y la técnica RAG programada para el endpoint Get AI Response documentado en el Swagger oscila entre 7 a 15 segundos desde la recepción de la pregunta, búsqueda en la información indexada y preparación de la repuesta en lenguaje natural. La figura 14 muestra este análisis de red.

Figura 14

Performance y Tiempos de Respuesta

Bajo este escenario, los colaboradores tardan en promedio 2 minutos en consultar el documento instructivo y generar una respuesta al tiquet del usuario por los canales de comunicación. Esta métrica arroja un acierto importante dentro del proyecto aplicado, ya que al tener un promedio de máximo 15 segundos en la generación de una respuesta coherente y congruente con los datos, ayuda en un 87% a reducir el tiempo que emplean los colaboradores.

Teniendo presente que el 40% del tiempo promedio de los colaboradores del área administrativa se invierte en esta clase de procesos operativos se obtiene las siguientes métricas bajo una jornada de 8 horas (480 minutos) como se explica en la tabla a continuación:

Tabla 2

Comparación Eficiencia de Ecosistema RAG

Métrica	Sin RAG	Con RAG	Impacto Real
Tiempo por consulta	~115 segundos	15 segundos	-87% de espera
Carga diaria (40% jornada)	192 minutos	25 minutos	167 min ahorrados

Nota. La tabla muestra los resultados de las métricas de eficiencia operativa del sistema RAG antes y después de su implementación.

La implementación de la solución permite que el 24% del tiempo total de la jornada laboral (antes "desperdiciado" en búsqueda manual) se convierta en tiempo disponible de 2 horas y 47 minutos para tareas estratégicas o de soporte presencial de alto valor.

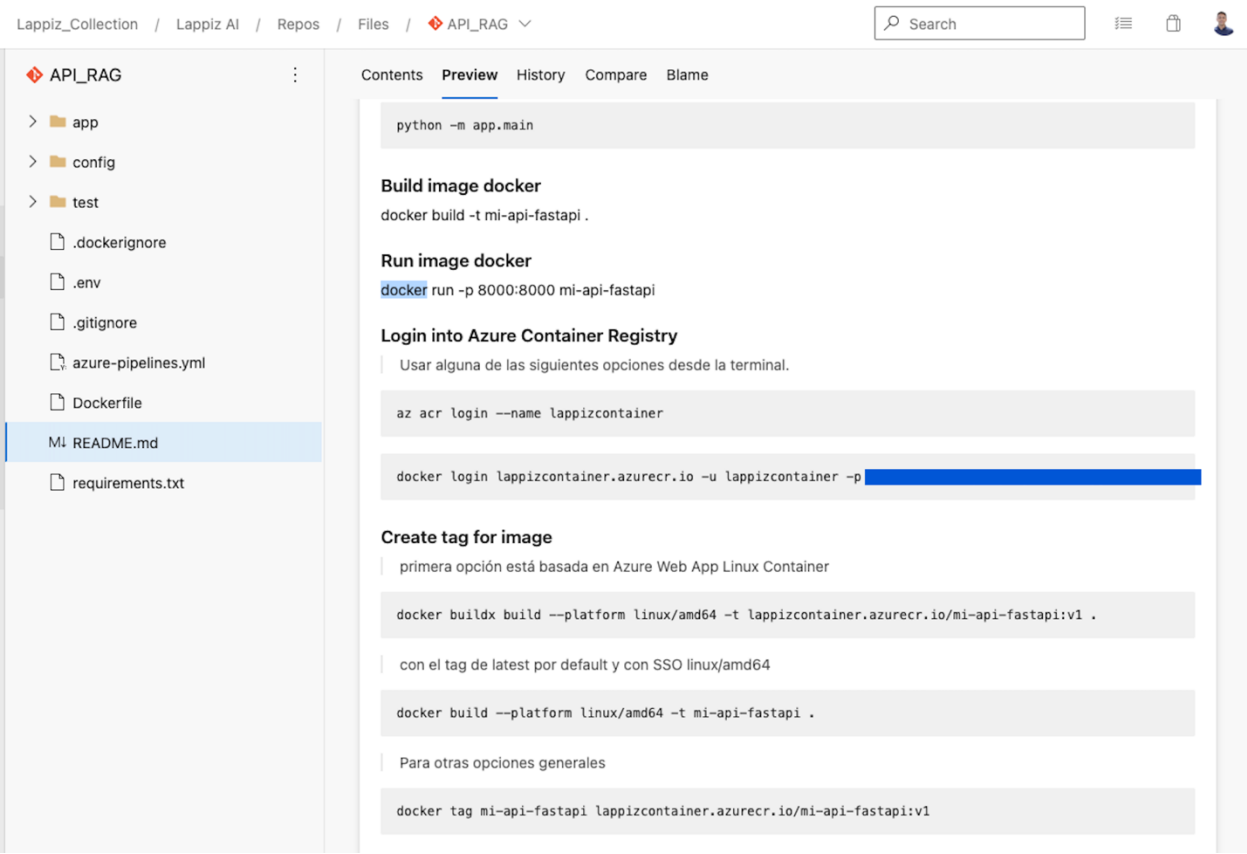
Documentación y Despliegue

Transferencia de Conocimiento

Para esta fase, la documentación del proyecto se entrega inmersa en los archivos readme.md de cada repositorio de código donde se explica la forma de ejecución y despliegue de cada componente bajo formato markdown como se muestra en la figura a continuación.

Figura 15

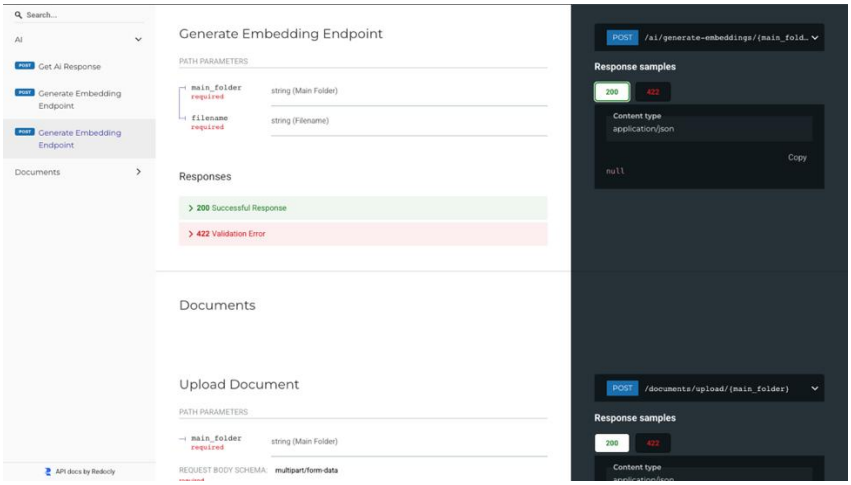
Documentación Técnica para Ejecución y Despliegue de API RAG



Además, el API se documenta por medio de Swagger como se ve en la figura 16. Este sirve como complemento del framework de FastAPI bajo la convención de OpenAPI.

Figura 16

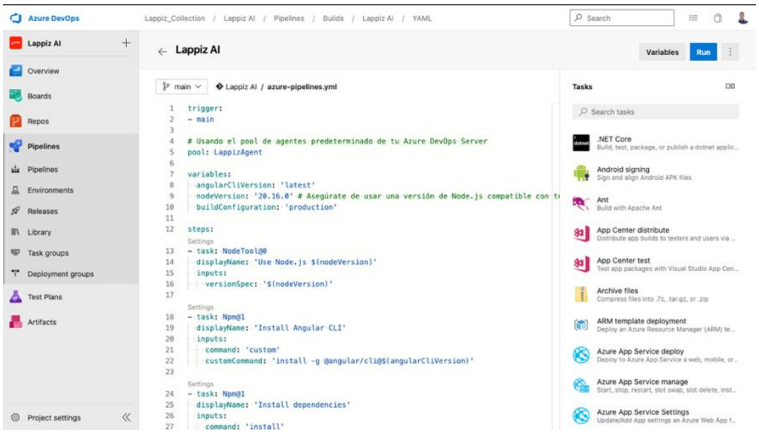
Documentación Bajo Convención OpenAPI con Swagger y Redocs



Finalmente, se opta por incorporar mecanismos de divulgación al equipo técnico y stakeholders del proyecto mediada por herramientas organizacionales como Microsoft Teams y se adiciona como se ve en la figura 17, la configuración de un Azure Pipeline para los procesos de CI (Continuos integration) y CD (Continuos delivery) del proyecto SPA en formato YAML.

Figura 17

Azure CI/CD Pipeline Formato YAML



Conclusiones

La implementación del sistema de Recuperación Aumentada por Generación (RAG) logró transformar la gestión documental de Fintic S.A.S., pasando de sistemas de búsqueda tradicionales ineficientes a una solución inteligente que permite el acceso rápido y seguro a la información. Al anclar las respuestas de los modelos de lenguaje (LLMs) exclusivamente a la base de conocimiento interna, se resolvió el problema de las "alucinaciones" y se garantizó la veracidad de la información corporativa.

El proyecto cumplió con la meta de optimizar los procesos de búsqueda, logrando una reducción del 87% en el tiempo de respuesta (bajando de un promedio de 115 segundos a solo 15 segundos por consulta). Esto se traduce en un ahorro diario de 167 minutos por colaborador en el área administrativa, permitiendo que el 24% del tiempo de la jornada laboral se redirija a tareas estratégicas de alto valor.

La arquitectura propuesta bajo un modelo de nube híbrida logró reducir en un 80% la exposición de documentos confidenciales a modelos de IA públicos. Al utilizar instancias privadas y seguras para el almacenamiento y procesamiento, la empresa ahora puede aprovechar el potencial de la IA generativa sin comprometer la integridad de su propiedad intelectual o datos sensibles.

El diseño basado en una Arquitectura Orientada a Servicios (SOA), utilizando Python (FastAPI), Langchain, Angular y bases de datos vectoriales (Pinecone), demostró ser una solución técnica robusta y escalable. La integración exitosa de estos componentes permitió no solo realizar búsquedas semánticas precisas, sino también establecer una base tecnológica que soporta el crecimiento futuro del volumen documental de la organización.

Recomendaciones

El proyecto en su elaboración tuvo diferentes oportunidades de mejora debido a su alcance acotado para dar cumplimiento al proyecto aplicado.

Ampliar los formatos de documentos soportados es una primera recomendación, dado que la fase inicial del proyecto se acotó principalmente a archivos de texto plano (.txt), pdf o docx para la vectorización, se recomienda incorporar el soporte para otros formatos comunes como pptx, documentos de audio y hojas de cálculo. Esto permitirá que el sistema RAG tenga una visión más integral de la base de conocimiento de la empresa.

Optimizar el monitoreo de costos de la nube para asegurar la sostenibilidad financiera del proyecto, es vital monitorear mensualmente los consumos de servicios como AWS S3 y Pinecone, especialmente si el volumen de datos crece significativamente. Se sugiere mantener la estrategia de utilizar planes gratuitos (tier free) o de bajo costo (pay-as-you-go) mientras la escala del sistema lo permita.

Implementar ciclos de re-entrenamiento y evaluación continua de forma periódica para la generación de nuevos embeddings a medida que la documentación operativa y comercial de Fintic se actualice. Además, es pertinente aplicar de forma regular las métricas de calidad definidas (precisión, coherencia y relevancia contextual) para garantizar que el sistema mantenga su alto nivel de acierto frente a nuevas consultas.

Desarrollar capacitaciones y controles de cambios, aunque el sistema cuenta con una interfaz intuitiva, se recomienda realizar jornadas de transferencia de conocimiento no solo técnicas, sino también funcionales para los usuarios finales de todas las áreas (dirección, gerencia y operaciones). Esto asegurará una adopción plena de la herramienta y maximizará los beneficios de la inteligencia colectiva de la organización.

Migrar la lógica de almacenamiento de documentos de AWS S3 a Cloud Storage de GCP debido a que se puede optimizar mejor con los documentos guardados en la base de datos vectorial que promueve conexión directa con librerías como ADK (Agent Development Kit) en reemplazo de PineCone.

Referencias Bibliográficas

Aarathisree, B., Sarkar, S., Dammala, B., Panday, M., & Sharma, N. (2024). *Revolutionizing the Future of Automated Subjective Answer Sheet Evaluation System with Machine Learning and LLMs.* IEEE Conference. Obtenido de https://openurl-ebsco-com.bibliotecavirtual.unad.edu.co/c/qcagk4/openurl?sid=ebsco:plink&id=ebsco:edseee:edseee.10895748

Arslan, M., & Cruz, C. (2024). *Business-RAG: Information Extraction for Business Insights.* Proceedings of the 21st International Conference on Smart Business Technologies (ICSBT). Obtenido de https://hal.science/hal-04862172/

Collado Alonso, M. A. (2024). *Implementación de técnicas de RAG (Retrieval Augmented Generation) sobre LLM (Large Language Models) para la extracción y generación de documentos en las entidades públicas*. Obtenido de https://research-ebsco-com.bibliotecavirtual.unad.edu.co/linkprocessor/plink?id=c0e7e813-40ce-352c-916d-4fdba25163e1

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., & Wang, H. (2023). *Retrieval-augmented generation for large language models: A survey.* arXiv preprint arXiv. Obtenido de https://simg.baai.ac.cn/paperfile/25a43194-c74c-4cd3-b60f-0a1f27f8b8af.pdf

Kang, Z., Yayun, H., Botao, Z., Xiaoyang, Q., Junqing, P., Xiao, J., & Wang, J. (2024). *Retrieval-Augmented Audio Deepfake Detection.* Proceedings of the 2024 International Conference on Multimedia Retrieval (ICMR '24). Obtenido de https://dl.acm.org/doi/abs/10.1145/3652583.3658086

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., & Kiela, D. (2020). *Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural*

Information Processing Systems. NeurIPS. Obtenido de https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

Solano Cohen, K. (2023). *Recuperación Aumentada Generativa (RAG) para la creación de chatbot de dominio específico.* Bogotá: Universidad de los Andes. Obtenido de https://repositorio.uniandes.edu.co/entities/publication/c12760d0-735b-46f2-b89b-ef045df2e548

Teixeira de Oliveira, D., Nascimento Silva, P., & Mafra Pereira, F. C. (2024). *Inteligência artificial para recuperação de dados abertos: reflexões e proposições para a experiência do usuário no Portal Brasileiro de Dados Abertos.* Brazilian Journal of Information Science. Obtenido de https://dialnet.unirioja.es/servlet/articulo?codigo=9545629

Varma, S., Shivam, S., Natajaran, S., Banerjee, A., & Roy, S. (2025). *From Data to Decisions: Enterprise-Level Domain-Specific Graph Retrieval-Augmented Generation Systems for Advanced Question Answering*. Obtenido de https://ieeexplore.ieee.org/abstract/document/10936909

Wang, X., Wang, Z., Gao, X., Zhang, F., Wu, Y., Xu, Z., . . . Huang, X. (2024). *Searching for Best Practices in Retrieval-Augmented Generation.* China: Cornell University. Obtenido de https://arxiv.org/pdf/2407.01219