Prototipo de Chat Inteligente con la API de GPT para Consultas de Normativas sobre el Espectro Radioeléctrico en Colombia

Maria Camila Alfonso Roncancio y Camila Andrea Beleño Cabrales

Trabajo de Grado para Optar al Título de Ingeniería Electrónica

Director
Homero Ortega Boada
Doctor en ciencias de la ingeniería, radiocomunicaciones

Codirector
Álvaro Enrique Patiño
Representante Directo de la Empresa TesAmerica

Universidad Industrial de Santander
Facultad de Ingenierías Fisicomecánicas
Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones
Ingeniería Electrónica
Bucaramanga
2025

Dedicatoria

Dedico este proyecto a quienes fueron mi sostén, mi impulso y mi inspiración en todo este camino.

A mi mamá, Marlen Cabrales por cada mensaje de aliento, por cada oración dedicada a mí, por cada palabra en el momento justo, por ser mi refugio y mi motor incansable. Gracias por enseñarme a seguir incluso cuando todo parecía imposible, fuiste la pieza más importante en este proceso.

A mi papá, Janner Beleño por sus consejos sabios, por creer en mí incluso en los momentos más difíciles, y por enseñarme, con su ejemplo, el valor del esfuerzo.

A mis hermanos, Mayra, Sadys, Jhon y Sofia por sus palabras de ánimo, su compañía incondicional, por recordarme siempre lo orgullosos que están de mí, y por ser una fuente constante de alegría.

A mis abuelos Marcelina, Sabas y a toda mi familia, por sus palabras de aliento, y por hacerme sentir siempre acompañada y celebrada en este camino.

A mi ángel en el cielo, que desde lo alto me impulsaba cada día. Sentí tu presencia en cada logro, en cada duda superada, en cada nuevo intento. Esta meta también es tuya.

A mi amiga y compañera de este proyecto de grado, gracias por tu compromiso, tu apoyo incondicional y por enfrentar conmigo cada etapa del proceso. Lo logramos juntas, y eso lo hace aún más valioso.

Este proyecto es fruto de muchas voluntades, de manos que apoyaron y corazones que acompañaron. A cada uno de ustedes, mi más profunda y eterna gratitud.

Camila Andrea Beleño Cabrales

Agradezco profundamente a Dios por ser mi guía constante, fuente de fortaleza y claridad en los días difíciles. También extiendo mi gratitud a las personas que han sido pilares fundamentales a lo largo de este camino académico y personal.

A mi mamá Yurleisy Roncancio Mahecha y a mi papá Gustavo Alfonso Martinez, gracias por su amor incondicional, por creer en mí incluso en los momentos más difíciles y por enseñarme con su ejemplo la importancia del esfuerzo, la perseverancia y los valores. Cada logro mío es también suyo, porque han estado presentes en cada paso, animándome a seguir adelante.

A mi hermana Laura Alejandra Velásquez, por estar siempre ahí con su alegría, sus palabras de ánimo y su compañía incondicional. Gracias por ser ese apoyo silencioso pero firme que me impulsó en los momentos de mayor cansancio.

A mi abuela Marleny Mahecha Guerrero, por ser una fuente infinita de cariño, sabiduría y fortaleza. Su apoyo constante, sus oraciones, sus palabras llenas de amor, me han acompañado siempre, dándome fuerzas cuando más las necesitaba.

A toda mi familia, por su apoyo, sus palabras de aliento y su compañía, incluso a la distancia. Su presencia ha sido refugio, motivación y alegría durante todo este proceso.

A mi amiga y compañera de tesis, gracias por caminar a mi lado en este proyecto con compromiso, entrega y compañerismo. Afrontamos juntas los desafíos, las dudas y los logros, y gracias a tu apoyo, este trabajo no solo fue posible, sino también significativo.

A todos ustedes, gracias por ser parte esencial de este logro.

Con cariño y gratitud,

Maria Camila Alfonso Roncancio

Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a todas las personas que contribuyeron de manera significativa a la realización de este proyecto.

En especial, agradecemos al profesor Homero Ortega Boada, director de nuestro trabajo de grado, por su valiosa orientación, su dedicación constante y por guiarnos con compromiso y paciencia a lo largo de todo el proceso. Su acompañamiento fue clave para transformar nuestras ideas en un proyecto sólido y significativo.

De igual manera, agradecemos a los profesores de la Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones en la Universidad Industrial de Santander, quienes con sus conocimientos, comentarios y apoyo nos ayudaron a fortalecer este proyecto y a crecer profesionalmente.

Finalmente, agradecemos de corazón a nuestros amigos y familiares. Su apoyo incondicional, su fe en nosotras y sus palabras de aliento fueron esenciales para llegar hasta aquí.

A todos ustedes, gracias por ser parte de este logro.

Tabla de Contenido

Pág.

Introducción 15

1. Objetivos 19

1.1 Objetivo General 19

1.2 Objetivos Específicos 19

2. Marco Teórico 20

2.1. Procesamiento del Lenguaje Natural (NLP) y Representaciones Semánticas. 20

2.2 Alternativas de Desarrollo de Modelos: Fine-Tuning vs. Prompt Engineering 24

2.2.1 Fine-Tuning 24

2.2.3 Prompt Engineering 24

2.3 Bases vectoriales y recuperación inteligente de información 25

2.4 Modelos de Lenguaje Pre Entrenados 26

2.5 API (Interfaces de Programación de Aplicaciones) 26

2.6 API RESTful 27

2.6.1 API GPT de OpenAI 27

2.7 Inteligencia Artificial Generativa 28

2.8 Modelos de Lenguaje Extensos (LLM) 28

2.8.1 Ventajas para el Acceso Normativo 29

2.9 LangChain (Orquestación de Componentes Para Respuestas Fundamentadas) 29

2.9.1 Cadenas 30

2.9.2 Agentes 30

2.10 Flask: puente entre el usuario y la inteligencia artificial 31

2.10.1 Flask para Chatbots 31

3. Diseño de la Solución 33

3.1 Necesidades del Usuario 33

3.1.1 Acceso Rápido y Preciso a la Información 34

3.1.2 Actualización automática de la información 34

3.1.3 Interfaz de Fácil Uso para Cualquier Persona 34

3.1.4 Escalabilidad 35

3.2 Requerimientos de la Solución 35

3.2.1 Requerimientos Funcionales 35

3.2.2 Requerimientos No Funcionales 36

3.3 Visión General del Sistema 37

3.4 Visión Particular de la Solución 37

3.5 Selección de Plataformas y Herramientas 38

3.5.1 OpenAI API – modelos de embedding y procesamiento lingüístico 39

3.5.2 Proveedor de servicios en la nube 39

3.5.3 Gestión y procesamiento de documentos 40

3.5.4 Orquestador modular del procedimiento semántico 41

3.5.5 Base de Datos Vectorial. 41

3.6 Enfoque metodológico: desarrollo ágil basado en sprints 42

3.7 Funcionamiento Completo del Sistema 43

3.7.1 Gestión Documental y Método de Entrenamiento 43

3.7.2 Consultas y generación de respuesta 45

3.8 Rol de las plataformas y herramientas en la solución 48

3.8.1 PythonAnywhere 48

3.8.2 Google Drive 48

3.8.3 Google Apps Script 49

3.8.4 Script de Entrenamiento en Python 49

3.8.5 Pinecone 49

3.8.6 OpenAI GPT API 49

3.8.7 Frontend Flask 50

3.9 Garantía de actualización y calidad en la información 50

4. Validación de la Solución 52

5. Conclusiones 55

Recomendaciones 57

Referencias Bibliográficas 58

Apéndices 62

Lista de Figuras

Pág.

Figura 1. *Ejemplo de cómo se visualiza un texto tokenizado*..... 20

Figura 2. *Búsqueda de vecinos aproximados.*..... 22

Figura 3. *Métricas de similitud.* 25

Figura 4. *Servicio que se encarga del proceso completo de entrenamiento.*..... 43

Figura 5. *Servicio que se encarga de procesar consultas y generar respuestas contextualizadas.*..... 45

Figura 6. *Análisis de validación semántica.* 53

Lista de Apéndices

Los apéndices están disponibles en el Repositorio Institucional

Apéndice A. Carpeta con el código completo de la interfaz web del chatbot inteligente. 62

Apéndice B. Código del web service que detecta nuevos documentos en la carpeta seleccionada, activado cada vez que es invocado por el script de entrenamiento. 62

Apéndice C. Código completo (“Script de entrenamiento”) para el entrenamiento del sistema, responsable de mantener actualizada la base de datos vectorial. 62

Apéndice D. Cuaderno de Google Colab utilizado para validar las respuestas del chatbot mediante pruebas de similitud semántica 62

Apéndice E. Manual técnico detallado del script de entrenamiento. 62

Apéndice F. Manual técnico detallado del web service, explicando su arquitectura, funcionamiento y despliegue. 62

Apéndice G. Manual sobre la arquitectura general y el funcionamiento del chatbot inteligente. 62

Apéndice H. Manual de uso y configuración del entorno de despliegue en PythonAnywhere. 62

Apéndice I. Archivo en formato Excel con los resultados del análisis de similitud semántica entre las respuestas generadas por el chatbot y las respuestas esperadas, utilizado como parte del proceso de validación del sistema. 62

Apéndice J. Google Sheet con el listado de preguntas utilizadas para evaluar el rendimiento del chatbot, formuladas a partir del contenido de los documentos normativos del espectro radioeléctrico en Colombia. 62

Glosario

API: "Interfaz de programación de aplicaciones". En el contexto de las API, la palabra aplicación se refiere a cualquier software con una función distinta. La interfaz puede considerarse como un contrato de servicio entre dos aplicaciones. Este contrato define cómo se comunican entre sí mediante solicitudes y respuestas. (Amazon Web Services, s.f.).

API RESTful: Estas son las API más populares y flexibles que se encuentran en la web actualmente. El cliente envía las solicitudes al servidor como datos. El servidor utiliza esta entrada del cliente para iniciar funciones internas y devuelve los datos de salida al cliente. (Amazon Web Services, s.f.).

Procesamiento de Lenguaje Natural (NLP): Es una tecnología de machine learning que brinda a las computadoras la capacidad de interpretar, manipular y comprender el lenguaje humano. (Amazon Web Services, s.f.).

Tokenización: Proceso de convertir una secuencia de texto en partes más pequeñas, conocidas como tokens. La razón principal por la que este proceso es importante es que ayuda a las máquinas a comprender el lenguaje humano (DataCamp, 2023)

Embedding (incrustración): Es un vector (lista) de números de punto flotante. La distancia entre dos vectores mide su grado de relación. Las distancias pequeñas indican un alto grado de relación, mientras que las grandes indican un bajo grado de relación (OpenAI, s.f.).

Prompt engineering: Proceso de escribir instrucciones efectivas para un modelo, de modo que genere consistentemente contenido que cumpla con sus requisitos. (OpenAI, s.f.).

Inteligencia artificial generativa: Tipo de IA capaz de crear nuevos contenidos e ideas, como conversaciones. Además, es capaz de aprender lenguaje humano, lenguajes de programación o cualquier otro tema complejo. (Amazon Web Services, s.f.).

GPT: Los transformadores generativos preentrenados, comúnmente conocidos como GPT, son una familia de modelos de redes neuronales que utilizan la arquitectura de transformadores y representan un avance clave en la inteligencia artificial (IA) (Amazon Web Services, s.f.)

HTML: Lenguaje de marcado mediante el cualh se estructura la base y contenido de cualquier página web. (Lenguaje HTML, s.f.)

Javascript: Es un lenguaje de programación, o lo que es lo mismo, un mecanismo con el que podemos decirle a nuestro navegador que tareas debe realizar, en que orden y cuantas veces. (ManzDev, s.f.)

Python: Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). (Amazon Web Services, s.f.)

JSON: (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. (JSON.org, 2002)

Resumen

Título: Prototipo de Chat Inteligente con la API de GPT para Consultas de Normativas sobre el Espectro Radioeléctrico en Colombia[*]

Autor: Camila Andrea Beleño Cabrales, Maria Camila Alfonso Roncancio[**]

Palabras Clave: Chatbot, Inteligencia artificial, NLP.

Descripción: El trabajo de grado surge del interés de TESAmerica por explorar el potencial de la inteligencia artificial. Por ello, se realiza un análisis interno para identificar un problema relevante que estuviera atravesando la empresa y que pudiera solucionarse con IA. En este estudio el difícil acceso, comprensión y volumen de la normativa del espectro radioeléctrico en Colombia. Esto representa un gran obstáculo para empresas y usuarios interesados en el sector de las telecomunicaciones, debido a que el espectro radioeléctrico es un recurso natural limitado, pero, además, es esencial para las comunicaciones modernas, llevando a que una mala interpretación de su regulación pueda desencadenar en consecuencias legales y técnicas considerables. Para solucionarlo, se desarrolló un prototipo de chatbot basado en IA generativa y procesamiento de lenguaje natural, capaz de responder consultas con información precisa y actualizada. Integrado con herramientas como Google Drive, LangChain, servicios web, y una infraestructura en la nube, facilitando su escalabilidad y sostenibilidad. Es importante resaltar que esta solución tiene el potencial de generar un impacto positivo no solo en el sector regulatorio, sino también en otras áreas estratégicas del país donde se requiera consultar y aplicar documentaciones complejas.

---

[*] Trabajo de Grado de Investigación

[**] Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones. Director: Homero Ortega Boada. Doctor en ciencias de la ingeniería. Codirector: Álvaro Enrique Patiño. Representante Directo de la Empresa TesAmerica.

Abstract

Title: Prototype of an Intelligent Chat with the GPT API for Regulations Queries on the Radio Spectrum in Colombia.[*]

Author: Camila Andrea Beleño Cabrales, Maria Camila Alfonso Roncancio.[**]

Key Words: Chatbot, Artificial Intelligence, PLN.

Description: This thesis arose from TESAmerica's interest in exploring the potential of artificial intelligence. Therefore, an internal analysis was conducted to identify a relevant problem the company was facing that could be solved with AI. This study highlights the difficult access, understanding, and volume of radio spectrum regulations in Colombia. This represents a major obstacle for companies and users interested in the telecommunications sector, given that the radio spectrum is a limited natural resource, yet essential for modern communications. A misinterpretation of its regulations can lead to considerable legal and technical consequences. To address this issue, a prototype chatbot based on generative AI and natural language processing was developed, capable of responding to queries with accurate and up-to-date information. Integrated with tools such as Google Drive, LangChain, web services, and a cloud infrastructure, it facilitates its scalability and sustainability. It is important to highlight that this solution has the potential to generate a positive impact not only in the regulatory sector but also in other strategic areas of the country where complex documentation needs to be consulted and applied.

---

[*] Research degree work

[**] Faculty of Physicomechanical Engineering. School of Electrical, Electronic, and Telecommunication Engineering. Electronic Engineering. Director: Homero Ortega Boada. PhD in Engineering Sciences, Radiocommunications. Co-director: Álvaro Enrique Patiño. Direct Representative of TesAmerica

Introducción

Los ingenieros llevan años hablando sobre inteligencia artificial, pero hoy la situación es diferente: la IA ha llegado al común de las personas, y con ello está sugiriendo una transformación comparable en impacto a la que trajo la rueda en su época. Con la rueda cambió la misma mente de las personas, de realizar sus labores y dio origen a una revolución social que se conoció como la sociedad industrial. Más adelante, con los avances en las comunicaciones, surgieron otras revoluciones, como la sociedad de la información, hasta llegar a la globalización del mundo. Ahora, estamos a puertas de una nueva revolución social, impulsada por la inteligencia artificial generativa, pues esta se posiciona como el motor de una nueva transformación, que promete redefinir los entornos laborales, educativos, y cotidianos, gracias a su rápida evolución y auge a partir de herramientas como ChatGPT, Claude, Gemini, entre otras (García-Peñalvo, Llorens-Largo & Vidal, 2024). ¿Cómo llamaremos a la nueva sociedad post-IA? No lo sabemos, en todo caso, este proyecto explora cómo la IA puede ir más allá de sus aplicaciones en sectores especializados y convertirse en una herramienta poderosa y accesible para las empresas y sus clientes.

Gracias al auge de la inteligencia artificial, distintas empresas han comenzado a interesarse en sus aplicaciones. Una de ellas es TESAmerica, la cual se planteó como objetivo explorar el alcance de la misma y cómo puede potenciar sus servicios a través de su implementación. Por esto, se llevó a cabo un análisis interno, buscando esa necesidad que pudiera ser resuelta por medio de la inteligencia artificial. Llegando a un problema que se ha persistido en los últimos años: las personas interesadas en el uso del espectro deben realizar un arduo trabajo de búsqueda, interpretación y análisis de la normativa vigente. Esta normativa suele manejar un lenguaje bastante técnico y complejo, lo que puede derivar en errores costosos, sanciones e incluso la

suspensión de servicios. Frente a este panorama, la inteligencia artificial se presenta como una solución prometedora, capaz de facilitar el acceso y análisis de información normativa de manera más eficiente y precisa.

El espectro es uno de los recursos más importantes y limitados para la prestación de los servicios de comunicaciones, cuya administración adecuada es esencial para garantizar su funcionamiento eficiente, evitando interferencias y optimizando su uso. Por lo cual, existe una normativa específica que la regula, la cual suele actualizarse constantemente generando que en la web existan varios documentos con información contradictoria. Esta situación ha sido reconocida en estudios sobre el entorno digital en Colombia, donde se señala que el espectro es un habilitador fundamental para la economía digital, y que su acceso eficiente requiere no sólo infraestructura, sino también políticas públicas claras que garanticen conectividad y competitividad (Ortiz Laverde & Herrera Zapata, 2024).

Esta propuesta surge a partir de dos referentes clave. Por un lado, una solución anterior desarrollada por el grupo RadioGis, en el que se implementó inteligencia artificial para responder preguntas relacionadas con la acreditación ABET en la Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones (Rueda & Hernández, 2024); y por otro, casos de éxito en otros sectores, como el jurídico, donde sistemas basados en IA y aprendizaje automático como Ross Intelligence permiten filtrar, priorizar y personalizar el contenido legal (Zakir, Bashir, Ali & Khan, 2024).

Inspirados en estos enfoques, y con el fin de combinar lo mejor de ambos y darle solución a aquellas falencias que se identificaron, se propuso el desarrollo de un prototipo de chat inteligente basado en IA generativa para facilitar la comprensión y consulta de estas normativas. La implementación de técnicas de procesamiento de lenguaje natural (NLP) y bases vectoriales como Pinecone, orientados a mejorar la comprensión semántica de las consultas para este proyecto, es

fundamental, ya que permiten que la información a pesar de ser tratada por IA generativa (GPT) y modelo extenso de lenguaje (LLM) siga siendo precisa y basada en documentos oficiales y actualizados. Esto es clave para mitigar los riesgos de desinformación asociados al uso de IA generativa (García-Peñalvo et al., 2024). Además, el modelo de gestión documental por medio de carpetas en Google Drive facilita la actualización constante del contenido, haciéndolo escalable y apto para cualquier entidad o usuario.

El sistema se diseñó en una arquitectura modular y automatizada, lo que significa que está compuesto por partes independientes que trabajan en conjunto, pero que también pueden actualizarse o cambiarse por separado sin afectar a todo el sistema. Se combinaron herramientas como: Google Drive (repositorio dinámico), Google Apps Script (para la creación de un web service que extrajera automáticamente el contenido), OpenAI (ext-embedding-ada-002 y GPT-4), LangChain (coordinador de flujo semántico), Pinecone (donde se guarda toda la información procesada) y PythonAnywhere (entorno de ejecución en la nube). Durante el desarrollo, se evaluaron múltiples tecnologías y se seleccionaron las que ofrecieran mejor rendimiento, escalabilidad y facilidad de integración. El proceso se llevó a cabo bajo la metodología ágil Scrum, en cuatro sprints que abarcan desde la extracción automática de documentos hasta la validación con usuarios reales. Las pruebas piloto con TESAmerica y el grupo RadioGIS permitieron ajustar la precisión del sistema.

Un aspecto clave de este proyecto fue la transformación del contenido normativo en un formato que pudiera entender la inteligencia artificial. Para esto, se utilizó un proceso llamado vectorización (embeddings), mediante el cual los fragmentos de los documentos se convierten en una especie de “huella digital numérica”. Esto permite que, cuando un usuario hace una pregunta, el sistema pueda buscar de manera inteligente los fragmentos más parecidos en contenido y

significado, aunque no usen exactamente las mismas palabras. Gracias a esta técnica, el sistema puede responder preguntas formuladas de muchas formas distintas, entender lo que el usuario necesita y devolver una respuesta precisa basada en documentos reales. Esto evita respuestas inventadas y garantiza que la información sea confiable, clara y actualizada.

Como resultado, este proyecto de grado ha permitido a las empresas y usuarios que hicieron parte de las pruebas piloto que consulten la normativa del espectro de forma sencilla, reduciendo los errores por malinterpretaciones y evitando sanciones. Además, este proyecto abre la posibilidad de que entidades como la ANE y el MinTIC consideren este tipo de herramientas en sus estrategias de modernización, y también demuestra que la misma tecnología podría adaptarse para facilitar el acceso a documentación informativa en sectores como la salud, el medio ambiente o la educación, ampliando su impacto social y tecnológico (Rivero & Beltrán, 2024; Ortiz Laverde & Herrera Zapata, 2024; Zakir et al., 2024).

1. Objetivos

1.1 Objetivo General

Desarrollar un prototipo basado en la API de GPT orientado al aprendizaje automático mediante un entrenamiento autónomo y un almacenamiento eficiente en una base vectorial para consultas de normativas sobre el espectro radioeléctrico ene Colombia.

1.2 Objetivos Específicos

1. **Definir los requisitos funcionales y no funcionales del prototipo, identificando las características técnicas necesarias para la integración un sistema de chat inteligente en la nube con una base de datos vectorial.**
2. **Implementar un sistema eficiente para la recepción y procesamiento automático de documentos normativos, integrando Google Drive con servicios web y la API de GPT para la generación de embeddings (resultados de entrenamiento) y almacenamiento en una base de datos vectorial.**
3. **Desarrollar e implementar la arquitectura del prototipo, desarrollando un sistema que combine las prestaciones de la base de datos vectorial y la API de GPT con una interfaz interactiva para los usuarios finales que facilite consultas eficientes y precisas.**
4. **Validar el prototipo en el contexto de la normatividad del espectro radioeléctrico en Colombia mediante pruebas funcionales y de rendimiento, asegurando la precisión y relevancia de las respuestas generadas.**

2. Marco Teórico

En la última década, el desarrollo de soluciones inteligentes ha sido impulsado por la unión de diversas tecnologías que permiten construir sistemas capaces de procesar información compleja, interactuar con los usuarios de manera natural y escalar eficientemente en entornos digitales. Este marco teórico aborda los principales conceptos y herramientas utilizadas para el desarrollo de un prototipo de chat inteligente, orientado a la consulta de la normativa del espectro radioeléctrico en Colombia, integrando componentes de inteligencia artificial generativa, procesamiento de lenguaje natural y servicios web modernos.

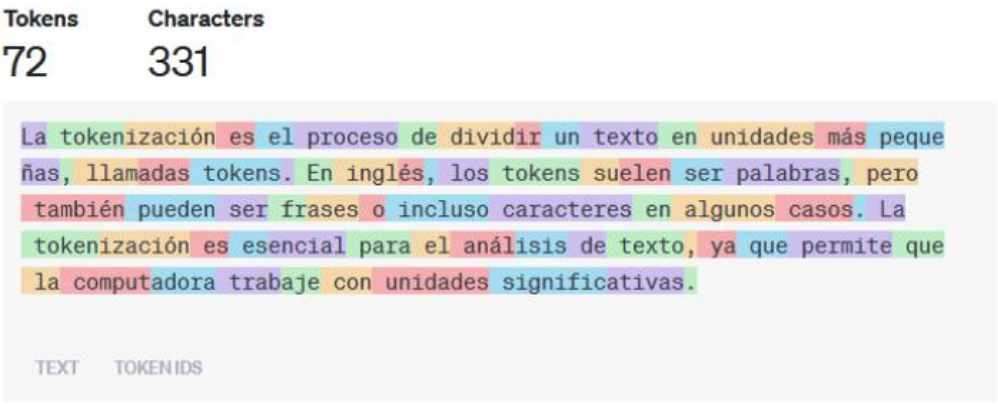
2.1. Procesamiento del Lenguaje Natural (NLP) y Representaciones Semánticas.

El procesamiento de lenguaje natural, es un campo dentro de la inteligencia artificial la cual le otorga a las computadoras la habilidad para comprender y procesar el lenguaje humano. Este proceso se basa en varias etapas fundamentales, entre las cuales destacan la tokenización y la generación de embeddings.

La tokenización es uno de los pasos iniciales y más importantes: consiste en dividir un texto en pequeñas unidades llamadas tokens, que pueden ser palabras, signos de puntuación o incluso fragmentos de palabras. Esta división convierte el texto en una forma que la IA puede entender y procesar, siendo esencial para tareas como búsqueda de información, clasificación de texto o generación de respuestas (Gamallo & García, 2012).

Figura 1.

Ejemplo de cómo se visualiza un texto tokenizado



Nota. Tomado de https://platform.openai.com/tokenizer

Tal como se muestra en la Figura 1, este proceso divide el texto en unidades más pequeñas, lo que lo convierte en una forma más manejable y computacionalmente procesable. Es similar a separar los ingredientes de una receta antes de cocinar, permitiendo comprender y trabajar con cada componente por separado.

La tokenización tiene unas ventajas específicas para el proceso normativo, entre las cuales podemos destacar:

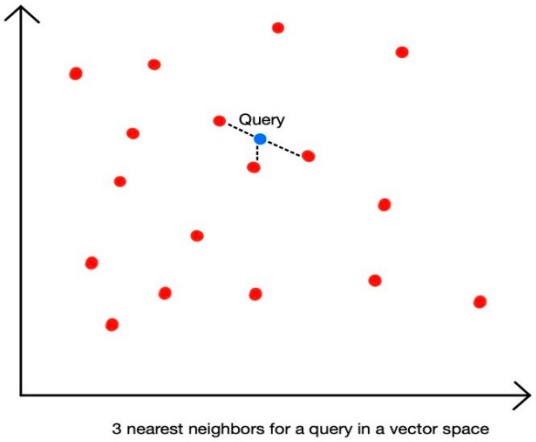
- **Fragmentación Precisa de Textos Complejos.** La tokenización permite dividir normativas extensas en fragmentos pequeños y manejables. Esto facilita que el sistema procese solo el segmento relevante, en lugar de analizar documentos completos, lo que reduce la carga computacional y mejora la velocidad de respuesta.
- **Preservación del Contexto Técnico-legal.** Un buen tokenizador adaptado al dominio legal respeta expresiones como “banda de 2.4 GHz” o “Resolución 105 de 2023” como unidades semánticas. Esto garantiza que no se rompa el significado jurídico o técnico del texto (Gamallo & García, 2012).

- **Mejora de la Búsqueda Semántica Posterior.** Al definir con precisión los límites del texto, se generan mejores embeddings, lo que permite una recuperación semántica más precisa cuando el usuario hace una consulta. Es decir, mejora la calidad del contenido que se compara con la pregunta del usuario.
- **Permite la Trazabilidad del Fragmento Normativo Citado.** Cada fragmento tokenizado puede asociarse a su fuente original y ubicación exacta (resolución, artículo, numeral), lo cual permite ofrecer respuestas verificables, que el usuario puede rastrear en el documento fuente.

Una vez el texto ha sido tokenizado, puede ser transformado en embeddings, que son representaciones numéricas que capturan el significado de palabras o frases dentro de un espacio multidimensional. Esta técnica permite que dos expresiones distintas, como “permiso ambiental” y “licencia ecológica”, puedan identificarse como equivalentes si comparten un significado similar. Esto resulta clave en entornos normativos, donde la forma de expresar un concepto puede variar, pero la intención detrás del texto se mantiene.

Figura 2.

Búsqueda de vecinos aproximados.



Nota. **Tomado de: https://www.pinecone.io/learn/what-is-similarity-search/**

Para comprender cómo funciona este proceso, podemos imaginar un plano como el que se muestra en la Figura 2. Allí, cada punto rojo representa un fragmento de texto normativo previamente procesado y almacenado, y el punto azul representa una consulta realizada por un usuario. Todos estos fragmentos, anto la pregunta como las respuestas posibles, han sido convertidos en embeddings y ubicados en un espacio vectorial, donde la distancia entre los puntos refleja qué tan similares son en significado. El sistema puede entonces identificar cuáles fragmentos están más cerca de la consulta en este espacio (conocidos como los "vecinos más cercanos") y devolverlos como respuesta.

Este enfoque permite una búsqueda semántica real, es decir, una búsqueda que entiende lo que el usuario quiere decir, no solo las palabras exactas que utiliza. Así, el sistema puede recuperar con precisión información relevante incluso cuando el lenguaje utilizado por el usuario es diferente al del documento original. Además, al almacenar estos vectores en una base de datos especializada (como Pinecone), es posible realizar búsquedas eficientes sobre miles de documentos sin necesidad de analizarlos uno por uno. Esto hace que la tecnología de embeddings no solo sea precisa, sino también rápida y adaptable a entornos con información normativa en constante actualización.

Las ventajas de utilizar esta técnica para los procesos normativos, son:

Recuperación semántica más precisa. Los embeddings permiten comparar textos por su significado, no solo por coincidencia literal de palabras. Esto significa que si el usuario formula una pregunta con términos distintos a los que usa la normativa, el sistema igual podrá encontrar los fragmentos relevantes (Cortez et al., 2009).

- **Reformulaciones y lenguaje natural.** En normativas jurídicas, el mismo concepto puede expresarse de múltiples formas. Los embeddings permiten detectar

similitudes semánticas entre consultas del usuario y textos normativos, incluso si utilizan vocabularios distintos (Gamallo & García, 2012).

- **Facilita la organización y búsqueda en grandes corpus normativos.** Cada fragmento embebido se convierte en un vector almacenado en una base vectorial, lo que permite búsquedas eficientes y escalables sobre miles de normas, resoluciones y artículos sin necesidad de estructurarlas manualmente.
- **Posibilidad de actualización automática del conocimiento.** Una vez que nuevos documentos normativos se vectorizan y se agregan a la base vectorial, el sistema puede consultarlos de inmediato sin necesidad de reentrenar el modelo, lo cual facilita su mantenimiento en entornos con regulaciones dinámicas.

2.2 Estrategias para Potenciar Modelos de Lenguaje

Al entrenar modelos de lenguaje para tareas específicas, existen dos enfoques predominantes:

2.2.1 Fine-Tuning

Consiste en reentrenar un modelo de lenguaje con datos específicos del dominio (por ejemplo, normativas legales). Aunque ofrece alto control y precisión, es costoso en recursos computacionales y requiere gestión de versiones.

2.2.3 Prompt Engineering

En lugar de modificar el modelo, se optimiza la forma en que se formula la instrucción. Un buen prompt puede guiar al modelo general (como GPT-4, Gemini, Claude, LlaMA) a comportarse como un experto, sin necesidad de reentrenar. Esta práctica involucra técnicas como el encadenamiento de pensamientos, la autocoherencia o el uso de ejemplos múltiples (Srivastava & Beri, 2024).

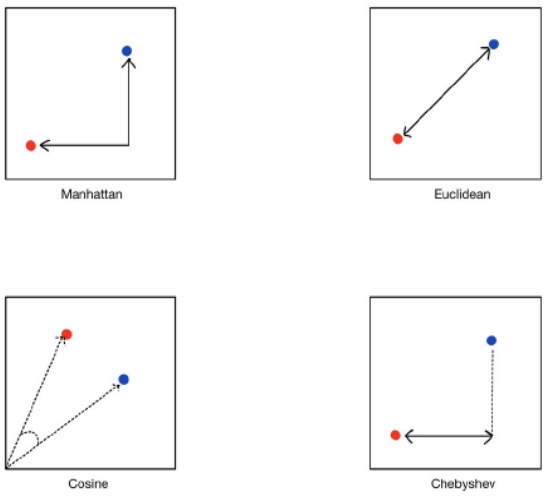
En este proyecto, se optó por el prompt engineering junto con embeddings vectoriales, lo cual ofrece mayor flexibilidad y escalabilidad, sin sacrificar calidad en las respuestas.

2.3 Bases vectoriales y recuperación inteligente de información

A diferencia de las bases de datos tradicionales que dependen de palabras claves exactas, las bases vectoriales permiten almacenar y recuperar información basada en significados. En este proyecto, cada fragmento normativo es procesado y convertido en un vector numérico (embedding) que representa su contenido semántico. Estos vectores son almacenados a una base vectorial como Pinecone, lo que permite realizar búsquedas rápidas, precisas y escalables sobre grandes volúmenes de texto.

Figura 3.

Métricas de similitud.



Nota. Tomado de: https://www.pinecone.io/learn/what-is-similarity-search/

Una vez que los fragmentos normativos y las preguntas del usuario han sido transformados en embeddings, el sistema necesita una forma de compararlos para determinar cuáles son más similares entre sí. Existen diversas métricas para medir esta relación entre vectores, como se ilustra en la Figura 3:

- Manhattan (suma de diferencias en cada dimensión)
- Euclidiana (distancia “recta” entre puntos)
- Chebyshev (máxima diferencia entre coordenadas)
- Coseno (ángulo entre vectores)

En este proyecto, se implementó la similitud coseno, una métrica ampliamente utilizada en procesamiento de lenguaje natural. A diferencia de otras, no se enfoca en la distancia exacta, sino en la dirección del significado. Es decir, dos textos pueden tener longitudes distintas, pero si apuntan hacia una idea similar en el espacio vectorial, serán considerados cercanos.

Esta estrategia permite que el sistema encuentre fragmentos normativos, aunque estén redactados de forma distinta a la consulta del usuario, fomentando una búsqueda verdaderamente semántica.

2.4 Modelos de Lenguaje Pre Entrenados

Los modelos de lenguaje pre entrenados, como GPT o BERT, han sido entrenados con enormes cantidades de texto (libros, artículos, páginas web) para aprender patrones del lenguaje humano. Es decir, antes de que les demos una tarea específica, estos modelos ya han leído y comprendido gran parte del lenguaje general.

Una vez tienen este conocimiento, pueden adaptarse a tareas concretas con muy pocos ejemplos, lo que se conoce como fine-tuning (ajuste fino). En este proyecto, se usó GPT-4 Turbo para responder preguntas normativas, sin necesidad de re entrenarlo, porque ya cuenta con una base lingüística general potente.

2.5 API (Interfaces de Programación de Aplicaciones)

Imagina que tienes un restaurante y necesitas comunicarte con la cocina para hacer un pedido. No necesitas saber cómo cocinan todo, solo debes pasar una orden clara. Así funciona una

API (Application Programming Interface): permite que dos programas “hablen” entre sí sin conocer los detalles internos del otro.

Por ejemplo, al construir un chatbot inteligente, usamos la API de OpenAI para enviarle preguntas y recibir respuestas. También usamos otras APIs (como Google Drive o Pinecone) para gestionar documentos y datos. Todo el sistema se conecta gracias a estas interfaces, de forma segura, modular y automatizada.

2.6 API RESTful

Una API RESTful (Representational State Transfer) es una forma estandarizada de permitir que diferentes sistemas se comuniquen entre sí por medio de internet, siguiendo ciertas reglas. Es como un mensajero eficiente y ordenado que sabe cómo hacer solicitudes claras, obtener respuestas específicas y no mezclar la información entre usuarios.

En términos sencillos, una API RESTful permite que dos aplicaciones se conecten sin necesidad de estar hechas con la misma tecnología (Park et al., 2024). Por ejemplo, este proyecto usa una API RESTful para conectarse con Google Drive (hecho por Google), con OpenAI (para enviar preguntas al modelo GPT), o con Pinecone (para buscar documentos similares en una base vectorial).

2.6.1 API GPT de OpenAI

Al estar GPT ya pre entrenado, la API permite afinar el modelo para tareas específicas con el entrenamiento de datos personalizados. Esta tiene características relevantes, en las cuales se destaca la capacidad de mantener conversaciones contextualizadas con el modelo, facilitando así, la creación de chatbots y asistentes virtuales más interactivos, además admite varios idiomas.

Para este proyecto de grado se usa la API GPT de OpenAI. Y para acceder a ella se realiza una suscripción que incluye un límite en tokens por mes a través de la API Key proporcionada,

esta fue llamada en el código, obteniendo múltiples beneficios, como la conversión de cada palabra en una representación matemática de vector numérico (embedding), lo cual ayuda a representar el significado de las palabras y su contexto en relación con otras palabras del texto. Además, una vez GPT está pre entrenado, se usa para la generación de texto, ya que, dado un contexto, GPT realiza la respuesta de manera natural.

2.7 Inteligencia Artificial Generativa

Los modelos generativos actuales, basados en aprendizaje profundo y procesamiento de lenguaje natural (NLP), pueden generar nuevas ideas y narrativas a partir del análisis de grandes volúmenes de datos, lo cual ha revolucionado campos como la educación, la investigación y la salud (Barroso et al., 2025). La IA generativa (IAG) permite que las máquinas no solo procesen, sino que creen contenido nuevo (texto, imágenes, etc.). A diferencia de los sistemas que solo clasifican, la IAG puede redactar, resumir o traducir.

Este proyecto, integra un mecanismo de respuestas basado en la recuperación semántica de documentos normativos reales, a través de embeddings y bases vectoriales (Pinecone). Este enfoque permite que las respuestas del chatbot estén siempre respaldadas por documentos oficiales, minimizando el riesgo de información inventada. Este tipo de integración entre generación y recuperación ha sido destacado como fundamental en entornos donde la precisión normativa es crítica (Miah et al., 2025).

2.8 Modelos de Lenguaje Extensos (LLM)

Los modelos de lenguaje extensos (LLM, por sus siglas en inglés) son redes neuronales profundas entrenadas con billones de palabras provenientes de textos diversos, lo que les permite comprender y generar lenguaje natural a un nivel avanzado. A diferencia de los modelos tradicionales que usaban técnicas estadísticas básicas, los LLM modernos están construidos sobre

arquitecturas tipo transformer, lo cual mejora su capacidad para entender el contexto y realizar tareas complejas (Alberts et al., 2023).

2.8.1 Ventajas para el Acceso Normativo

El uso de LLM en soluciones como la del chatbot normativo brinda muchos beneficios. Por un lado, permite procesar preguntas ambiguas o incompletas y ofrecer respuestas relevantes mediante técnicas como "few-shot learning", sin requerir entrenamiento adicional. Por otro lado, permite reformular o simplificar normas complejas para facilitar su comprensión a públicos no especializados. Sin embargo, el uso de los LLM debe ser supervisado, dado que también pueden presentar la alucinación (respuestas creíbles pero falsas), por ello, en este proyecto se combina el uso de LLM con bases vectoriales y embeddings que restringen las respuestas del modelo únicamente a documentos previamente cargados y validados, reduciendo así los riesgos de desinformación.

2.9 LangChain (Orquestación de Componentes Para Respuestas Fundamentadas)

La principal ventaja de LangChain radica en que permite conectar modelos de lenguaje como GPT con otras herramientas esenciales, como bases de datos vectoriales, validadores semánticos, APIs o servicios externos dentro de un flujo de trabajo coherente (Jeong et al., 2024).

Cuando usamos la API de OpenAI directamente, hay tres grandes limitaciones:

1. El modelo no recuerda documentos largos completos
2. No puede justificar de dónde saca su respuesta
3. No sabe si la información es actualizada

LangChain resuelve esos problemas. Es como un director de orquesta: organiza lo que debe hacerse en qué orden, conectando el modelo con herramientas externas.

Por ejemplo, al recibir una pregunta del usuario:

- LangChain tokeniza y transforma el texto
- Genera su embedding (significado vectorial)
- Lo compara con documentos normativos en Pinecone
- Recupera los más parecidos y se los pasa a GPT
- GPT responde solo con base en esos fragmentos

Gracias a esto, el sistema ofrece respuestas contextualizadas y con trazabilidad documental. Como demuestran Jeong et al. (2024), LangChain ha sido implementado exitosamente en dominios donde se requiere alta trazabilidad, como en revisiones científicas automatizadas, y su estructura basada en bloques reutilizables lo hace ideal para proyectos que pueden escalar a otros sectores regulatorios como salud, educación o medio ambiente.

2.9.1 Cadenas

Una cadena (chain), es una secuencia de acciones que se ejecutan en orden para lograr un objetivo específico. Aquí, las acciones están codificadas directamente en el código y se ejecutan de manera secuencial.

2.9.2 Agentes

Un agente en LangChain son componentes que permiten a un modelo de lenguaje como GPT tomar decisiones dinámicas sobre qué acciones realizar para responder a una consulta o resolver una tarea, en lugar de seguir una cadena fija de pasos. El proceso del agente empieza cuando recibe una entrada del usuario, y este, en lugar de generar directamente una respuesta, se hace preguntas sobre cómo, ¿debo buscar en documentos?, ¿es importante el uso de otra API?, ¿Qué documentación existe para esta pregunta?, para que finalmente, este agente decida usar las herramientas integradas(tools) y va construyendo una solución en varios pasos, pero guiado por el razonamiento del modelo.

2.10 Flask: puente entre el usuario y la inteligencia artificial

Flask es una herramienta ligera pero poderosa escrita en Python, que permite construir aplicaciones web de manera sencilla. Puede imaginarse como el "puente" que conecta la lógica de un programa con una página web que el usuario puede ver y usar. Gracias a su flexibilidad, Flask es ideal para proyectos como chatbots, donde se necesita responder a preguntas de manera dinámica y en tiempo real.

Una de las principales ventajas de Flask es que no impone reglas estrictas sobre cómo organizar el proyecto. En cambio, proporciona los elementos esenciales y deja que el desarrollador construya lo que necesita, lo que lo hace muy adaptable.

Esto permite que un chatbot desarrollado con Flask reciba preguntas, las procese con inteligencia artificial, y entregue respuestas en una interfaz clara, todo en pocos segundos (Xiao et al., 2023).

2.10.1 Flask para Chatbots

En el contexto de este proyecto, Flask cumple el rol de intermediario entre el usuario y el sistema inteligente. Al escribir una consulta, Flask se encarga de:

1. Recibir la pregunta del usuario desde la web
2. Llevar esa pregunta al sistema de procesamiento de lenguaje natural
3. Obtener la respuesta generada
4. Mostrar la respuesta en pantalla

Este proceso ocurre de forma automática y rápida, gracias a la estructura simple pero eficiente de Flask. Además, permite integrar fácilmente bases de datos (como MySQL o PostgreSQL) y conectarse con modelos de lenguaje como GPT, NLTK o Spacy para mejorar las respuestas del chatbot (Singh et al., 2023).

En resumen, Flask no solo facilita la creación del sitio web visible para el usuario, sino que también conecta esa interfaz con toda la inteligencia que ocurre detrás del chatbot.

3. Diseño de la Solución

Este capítulo describe de manera estructurada el proceso integral que permitió el desarrollo de la solución, desde la identificación y comprensión de las necesidades planteadas por TESAmerica y el grupo RadioGIS, hasta la construcción de un sistema que respondiera a cada uno de estos requerimientos de manera práctica y eficiente.

A continuación, se ofrece una visión amplia de la solución desarrollada, resaltando los componentes fundamentales y la manera en que estos trabajan en conjunto para atender de forma efectiva los retos identificados. Posteriormente, se documenta en detalle el proceso seguido para el diseño y construcción del sistema.

3.1 Necesidades del Usuario

En la fase inicial del proyecto, se llevaron a cabo reuniones para recolectar información y validar necesidades con el equipo técnico de TESAmerica, empresa interesada en explorar herramientas basadas en inteligencia artificial, con el fin de conocer sus capacidades y posibilidades; y con investigadores del grupo RadioGIS, quienes acompañaron la validación conceptual y técnica de la propuesta.

La principal necesidad expresada por TesAmerica era poder consultar, de manera ágil y comprensible, grandes volúmenes de información técnica y normativa, que tradicionalmente, resultan difíciles de buscar y entender. Su interés no era únicamente resolver un caso específico, sino explorar de primera mano hasta dónde podía llegar la inteligencia artificial para facilitar el acceso y manejo de conocimiento complejo; por ello el problema base que se plantea es manejar y comprender la normativa del espectro radioeléctrico en Colombia, un tema particularmente complejo y extenso.

Con este escenario de trabajo, el equipo en conjunto estableció las siguientes necesidades prioritarias, que orientan directamente el diseño de la solución:

3.1.1 Acceso Rápido y Preciso a la Información

El principal reto identificado por TESAmerica fue la dificultad de encontrar información puntual en los documentos que regulan el uso del espectro radioeléctrico en Colombia. Estos documentos suelen ser muy extensos, están escritos en lenguaje técnico y, muchas veces, para resolver una sola duda, se requiere revisar varios documentos normativos de más de cien páginas cada uno. Por ello, se consideró fundamental contar con una herramienta que permitiera buscar y obtener respuestas rápidas, claras y precisas, a partir de preguntas sencillas, evitando la exploración manual de los archivos.

3.1.2 Actualización automática de la información

TESAmerica, expresó su inquietud por el riesgo de trabajar con información que pudiera quedar desactualizada, ya que en el campo regulatorio esto puede conducir a errores tanto técnicos como legales. Por esta razón, se dio prioridad al diseño de una solución capaz de actualizar automáticamente la base de datos cada 24h, eliminando la necesidad de realizar revisiones manuales.

3.1.3 Interfaz de Fácil Uso para Cualquier Persona

La normativa del espectro radioeléctrico en Colombia es consultada por ingenieros, personal administrativo, consultores jurídicos y usuarios de distintas áreas. Por esto, la solución debía ser amigable y fácil de usar para cualquier persona, permitiendo hacer consultas desde un navegador web, sin pasos técnicos ni instalaciones complicadas.

3.1.4 Escalabilidad

Aunque el enfoque inicial fue la normativa del espectro radioeléctrico en Colombia, tanto TESAmerica como el grupo de investigación RadioGIS coincidieron en la importancia de desarrollar una solución con proyección de crecimiento y adaptable a otros dominios temáticos. Por ello, se prioriza en un enfoque, capaz de evolucionar sin rediseños estructurales, en donde cada componente puede escalar de forma independiente.

3.2 Requerimientos de la Solución

Durante la fase de diseño, se definieron los requerimientos funcionales y no funcionales que debe cumplir el prototipo, con el fin de responder a las necesidades identificas junto a TESAmerica y el grupo de investigación RadioGIS.

3.2.1 Requerimientos Funcionales

Estos requerimientos describen las funciones principales que el sistema debe cumplir:

- **RF1. Consulta en Lenguaje Natural:** El sistema debe permitir a cualquier persona realizar preguntas utilizando palabras sencillas y cotidianas, sin la obligación de conocer términos técnicos o especializados sobre normativa. Por ejemplo, un usuario puede preguntar "¿Cuál es el trámite para solicitar una frecuencia?" y recibir una respuesta clara y adaptada a su consulta.
- **RF2. Recuperación automática de documentos:** El sistema debe buscar, identificar y extraer de manera automática los documentos normativos que estén almacenados en una carpeta organizada en Google Drive.
- **RF3. Relación inteligente entre preguntas y documentos:** La solución debe analizar el contenido de los documentos y organizarlo de forma que, al recibir una pregunta, sea capaz de identificar los fragmentos más relevantes y relacionados con

la consulta realizada. Así, cada respuesta se fundamenta en la información más apropiada del archivo original.

- **RF4. Generación de respuestas contextualizadas:** Ante cada pregunta, el sistema debe entregar una respuesta precisa, sencilla de entender y adaptada al contexto de la consulta, asegurando que la información dada sea útil y verificable.
- **RF5. Actualización automática de la base de conocimiento**: El sistema debe actualizar la información disponible cada día de manera automática, de modo que siempre consulte la versión más reciente de los documentos disponibles en la carpeta de Google Drive, sin depender de revisiones manuales.
- **RF6. Acceso a través de una interfaz web sencilla:** la solución debe contar con una página web donde cualquier persona, incluso aquellas sin conocimientos técnicos, pueda acceder al servicio desde su navegador, sin necesidad de instalar programas adicionales.

3.2.2 Requerimientos No Funcionales

Estos requerimientos definen las condiciones de calidad, rendimiento y sostenibilidad que debe garantizar la solución a mediano y largo plazo:

- **RNF1. Disponibilidad permanente:** El sistema debe estar siempre disponible para los usuarios, funcionando de forma continua en la web para que pueda consultarse en cualquier momento del día.
- **RNF2. Confianza en la actualización:** La actualización automática de la base de documentos, debe funcionar de manera confiable cada 24 horas, evitando que se mezclen versiones antiguas y nuevas.

- **RNF3. Tiempo de Respuesta Competitiva:** Al recibir una consulta, el sistema debe entregar la respuesta en menos de quince segundos, incluyendo tanto la búsqueda como la generación del texto.
- **RNF4. Consumo eficiente de recursos:** El sistema debe funcionar de manera óptima, sin requerir grandes recursos técnicos o económicos, permitiendo su ejecución en plataformas buenas y asequibles.
- **RNF5. Integración con otros servicios:** El sistema debe ser capaz de conectarse e intercambiar información con otros servicios externos mediante interfaces de programación (APIs), facilitando así futuras expansiones o integraciones, garantizando flexibilidad y compatibilidad.

3.3 Visión General del Sistema

En un entorno donde la información regulatoria es abundante y cambia constantemente, se vuelve clave contar con herramientas que faciliten su acceso, interpretación y uso. Este proyecto propone una solución innovadora que aprovecha las capacidades de la inteligencia artificial para mejorar la forma en que se gestionan y consultan documentos normativos, especialmente en contextos técnicos y especializados como el del espectro radioeléctrico.

La propuesta está orientada a reducir las barreras de comprensión y acceso a la información, promoviendo procesos más ágiles, eficientes y sostenibles en entornos organizacionales y regulatorios.

3.4 Visión Particular de la Solución

La solución busca principalmente facilitar el manejo y la consulta de información extensa e importante, como la normativa del espectro radioeléctrico en Colombia, evitando que los usuarios tengan que lidiar con procesos complejos o realizar tareas manuales. Este sistema ayuda

a combinar procesos de automatización e integraciones con la inteligencia artificial, garantizando que la información tratada siempre se encuentre actualizada y que las personas puedan acceder a ella de manera sencilla, formulando preguntas en lenguaje común y asimismo obtener respuestas claras y contextualizadas.

Para lograrlo, la solución brinda dos servicios fundamentales que trabajan de manera complementaria:

- El primer servicio automatiza la incorporación y actualización de la normativa en la base de datos vectorial, asegurando que el sistema siempre trabaje con la información más reciente.
- El segundo servicio permite que cualquier usuario realice consultas y reciba respuestas precisas en pocos segundos, aprovechando las capacidades de la inteligencia artificial para comprender y buscar información relevante.

Esta estructura modular no solo resuelve el reto de la gestión normativa en TESAmerica, sino que también permite que la misma tecnología se aplique fácilmente a otros contextos y tipos de información en el futuro.

3.5 Selección de Plataformas y Herramientas

El desarrollo de una solución de inteligencia artificial para la gestión y consulta de cualquier tipo de información, en este caso información normativa, requiere mucho más que la simple utilización de una API de procesamiento de lenguaje. Aunque el acceso a modelos avanzados facilita tareas como la comprensión o generación de texto, una sistema realmente útil y escalable debe integrar una arquitectura compuesta por diferentes plataformas y herramientas tecnológicas, cada una cumpliendo funciones específicas y complementarias dentro del flujo operativo general. La selección de plataformas y herramientas tecnológicas constituye un pilar

fundamental en este desarrollo, pues determina la eficiencia, escalabilidad y facilidad de mantenimiento del sistema. Para este prototipo, la selección respondió tanto a criterios técnicos como a la disponibilidad de recursos y la proyección de crecimiento del sistema.

3.5.1 OpenAI API – modelos de embedding y procesamiento lingüístico

Se emplea la API para adquirir dos servicios, uno para la generación de vectores semánticos mediante el modelo text-embedding-ada-002, destacado por su velocidad, bajo costo y alta combatividad con flujos de recuperación semántica (OpenAI, 2024a); y el otro para la generación de respuestas, en la cual se emplea el modelo gpt-4-1106-preview, perteneciente a la familia GPT-4 Turbo, optimizando especialmente para tareas que requieren razonamiento y manejo de contextos extensos. Ambos modelos se integran mediante la API oficial de OpenAI, garantizando estabilidad, soporte y escalabilidad del sistema conversacional (OpenAI, 2024b).

La elección de estos modelos se justifica frente a alternativas como GPT-3.5 debido a que GPT-4 Turbo ofrece mayor precisión y comprensión de normativas extensas, y text-embedding-ada-002 presenta el mejor balance entre costo y eficiencia para indexación semántica a diferencia de otras opciones como text-embedding-ada-001. El uso de otros modelos libres habría implicado mayores costos, menor velocidad de interferencia, inconsistencia en respuestas, entre otros factores.

3.5.2 Proveedor de servicios en la nube

La elección del entorno de despliegue se centró en equilibrar facilidad de configuración, costos y compatibilidad con los componentes de software utilizados (Python, Flask, LangChain, etc.). Se optó por PythonAnywhere como plataforma inicial de despliegue debido a su enfoque específico en aplicaciones Python, integración nativa con Flask y su entorno de ejecución

simplificado que permite la creación y gestión de scripts periódicos sin necesidad de configurar servidores complejos (PythonAnywhere, 2023).

Esta plataforma resulta especialmente conveniente en etapas de desarrollo y validación de prototipos, al permitir iteraciones rápidas y bajo costo operativo. Sin embargo, se reconoce que presenta limitaciones en términos de escalabilidad horizontal, por lo que se considera migrable a entornos más robustos como AWS Lambda, Google Cloud Run o Microsoft Azure Functions en caso de aumento significativo en la carga de usuarios o requerimientos de procesamiento.

Proveedores como AWS o Azure ofrecen una gama mucho más amplia de servicios, infraestructura distribuida y escalabilidad prácticamente ilimitada (Amazon Web Services, 2023; Microsoft Azure, 2023). No obstante, implican una mayor complejidad de configuración y un modelo de precios menos predecible, lo cual puede ser desfavorable en etapas iniciales del proyecto.

3.5.3 Gestión y procesamiento de documentos

El sistema utiliza Google Drive como repositorio documental central para almacenar los decretos, leyes y resoluciones de interés. Su uso responde a la necesidad de contar con una plataforma confiable, accesible para todo tipo de personas, y que a su vez permita el control de versiones y colaboración distribuida. Además, dado que el contenido fuente se restringió a que solo en formato PDF, se implementó un microservicio mediante Google Apps Script, el cual actúa como intermediario entre Drive y el sistema, permitiendo la extracción automática del texto a través de peticiones HTTP autenticadas (Google Developers, 2023).

Esta arquitectura ofrece una ventaja significativa al evitar el uso de herramientas locales para OCR o extracción documental, centralizando todo el acceso y control desde la nube de Google.

3.5.4 Orquestador modular del procedimiento semántico

El sistema requiere un marco especializado dedicado en coordinar los procesos de gestion documental, generación de embeddings y recuperación semántica. Este framework permite componer cadenas reutilizables que encapsulan funciones complejas, como la segmentación contextual de textos, la integración con Pinecone, y la generación de prompts personalizados para la API de OpenAI (Chase et al, 2023). Frente a otras alternativas como Haystack o LlamaIndex, LangChain fue seleccionado por su robusta comunidad, documentación clara, amplia información en la web y soporte para múltiples entornos de despliegue (LangChain, 2023; Gowda et al., 2023).

Además, su uso facilita el mantenimiento del sistema, permite escalar a nuevos dominios documentales, y agiliza la iteración sobre componentes individuales sin afectar la arquitectura general.

3.5.5 Base de Datos Vectorial.

La gestión y búsqueda de fragmentos documentales en forma vectorial requiere una base de datos especializada en alta dimensionalidad y rendimiento en tiempo real. Por ello se evaluaron varias opciones destacadas en el mercado profesional y open-source, considerando criterios técnicos como rendimiento, facilidad de integración con LangChain y costos operativos.

De todas las opciones, Pinecone fue seleccionado por ofrecer una base completamente gestionada con búsqueda de similitud de baja latencia, escalabilidad automática y alta disponibilidad, todo sin exigir mantenimiento de infraestructura, lo que lo posiciona como una opción óptima para prototipos y sistemas en producción (DataCamp, 2025; Pinecone, 2023). Weaviate y Qdrant también fueron contemplados por su naturaleza open source: Weaviate destaca por su gestión distribuida de vectores y uso de GraphQL, mientras que Qdrant sobresale por su eficiencia en consultas de baja latencia y soporte para búsquedas híbridas, lo que lo hace ideal en

entornos con múltiples tipos de datos (Murf AI, 2025; Ozkaya, 2025; Reddit, 2024). Milvus, con su arquitectura distribuida y soporte para GPU (Unidad de Procesamiento Gráfico), puede escalar horizontalmente en contextos de gran volumen de vectores, aunque requiere mayor complejidad administrativa (Murf AI, 2025). En entornos más limitados, Chroma DB propone una alternativa ligera y simple, ideal para prototipos; sin embargo, su enfoque en modo único lo limita en escenarios de gran escala (DataCamp, 2025). Por último, pgvector como extensión de PostgreSQL permite consultar vectores dentro de una base relacional existente, lo que resulta conveniente cuando se combinan datos tabulares y vectores, aunque su desempeño es menor frente a motores especializados (CloudRaft, 2025).

Finalmente, en nuestra solución, se prioriza Pinecone por su rendimiento inmediato, escalabilidad cloud y facilidad de integración con LangChain. Las alternativas open source especialmente Qdrant y Weaviate representan opciones viables para fases autogestionadas o despliegues en infraestructuras propias.

3.6 Enfoque metodológico: desarrollo ágil basado en sprints

Para asegurar que la solución respondiera realmente a las necesidades de TESAmerica y del grupo RadioGIS, el desarrollo del sistema se organizó utilizando la metodología ágil Scrum. Este enfoque se basa en una construcción paso a paso del prototipo, a través de ciclos cortos de trabajo llamados sprints. En cada sprint se planificaron, desarrollaron y revisaron funcionalidades, permitiendo una entrega continua de avances y la obtención constante de sugerencias por parte del equipo.

Esta forma de trabajo facilitó adaptarse a los cambios, encontrar y corregir errores temprano y mantener el rumbo según lo que los usuarios esperaban. Así, el sistema fue mejorando gradualmente, validando cada etapa antes de continuar con la siguiente.

En total, se realizaron cuatro sprints, con los siguientes enfoques principales:

Sprint 1. Implementación del mecanismo para extraer automáticamente el contenido de los documentos normativos desde Google Drive, utilizando Google Apps Script para asegurar la recuperación eficiente sin intervención manual.

Sprint 2. Diseño del proceso para dividir los documentos en fragmentos, convertirlos en vectores (embeddings) y almacenarlos en Pinecone, usando LangChain y la API de OpenAI. Esto permitió búsquedas semánticas más precisas.

Sprint 3. Desarrollo de una interfaz web sencilla para que cualquier persona pudiera hacer preguntas y recibir respuestas directamente en su navegador.

Sprint 4. Se planteó una etapa de validación y ajuste, en la cual se realizaron pruebas con el chatbot para evaluar cómo respondía a diferentes preguntas. A partir de los resultados, se hicieron los cambios necesarios para mejorar la precisión y la utilidad de las respuestas.

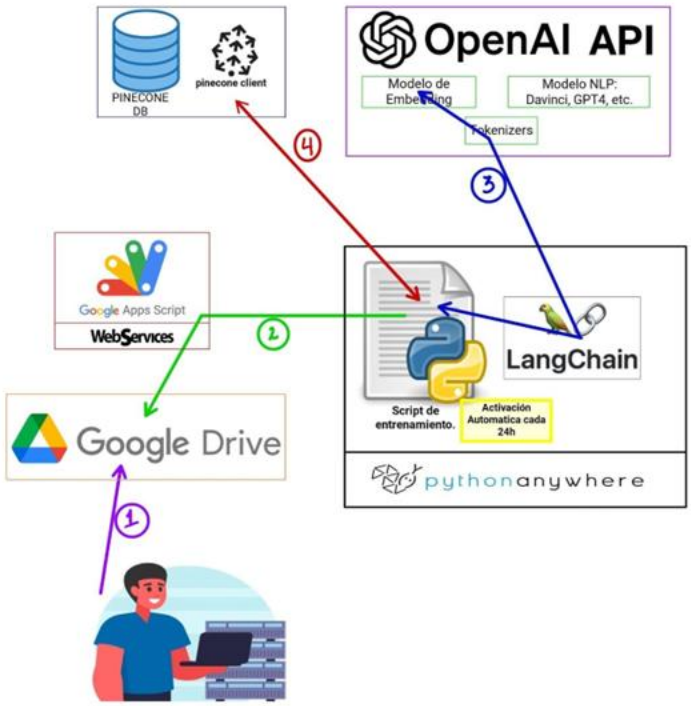
3.7 Funcionamiento Completo del Sistema

Esta solución responde a dos necesidades principales: por un lado, mantener la documentación normativa siempre actualizada, y por otro, facilitar que cualquier persona pueda consultarla de forma sencilla, rápida y precisa, sin importar su experiencia en el tema del espectro radioeléctrico en Colombia. Para lograrlo, el sistema cuenta con dos servicios fundamentales que trabajan juntos, permitiendo que toda la información esté disponible y sea fácil de encontrar cuando se necesite.

3.7.1 Gestión Documental y Método de Entrenamiento

Figura 4.

Servicio que se encarga del proceso completo de entrenamiento.



Nota. Autoría Propia

En la Figura 4 se muestra el flujo completo del sistema, que describe cómo se incorporan y preparan los documentos para que siempre estén disponibles al momento de ser consultados. El script de entrenamiento actúa como el componente central del proceso, coordinando todo gracias a la integración con LangChain. Este flujo operativo asegura que los documentos sean procesados de manera eficiente y estén listos para proporcionar respuestas precisas cuando el usuario realice una consulta.

El proceso comienza en el paso 1, donde cualquier persona autorizada, como una secretaria o un responsable del área, puede agregar, modificar o eliminar documentos normativos en una carpeta central de Google Drive. Esto asegura que la información esté siempre actualizada y disponible para el sistema.

En el paso 2, un servidor web creado en la plataforma de Google Apps Script detecta automáticamente los documentos presentes en la carpeta específica de Google Drive donde se

almacena la normativa. Posteriormente, realiza la extracción textual de cada archivo y devuelve la información al script de entrenamiento en formato JSON, incluyendo el nombre del documento y su texto correspondiente. Este proceso se ejecuta cada vez que el servidor es llamado por un script en Python, el cual se activa automáticamente cada 24 horas.

En el paso 3, el script de entrenamiento procesa el texto recibido para adaptarlo y asegurar su compatibilidad con los modelos de inteligencia artificial de OpenAI. Usando la biblioteca LangChain, el sistema fragmenta los textos en partes más pequeñas y los convierte en vectores numéricos (embeddings) mediante los servicios de OpenAI. Estos vectores regresan al script de entrenamiento para su siguiente procesamiento.

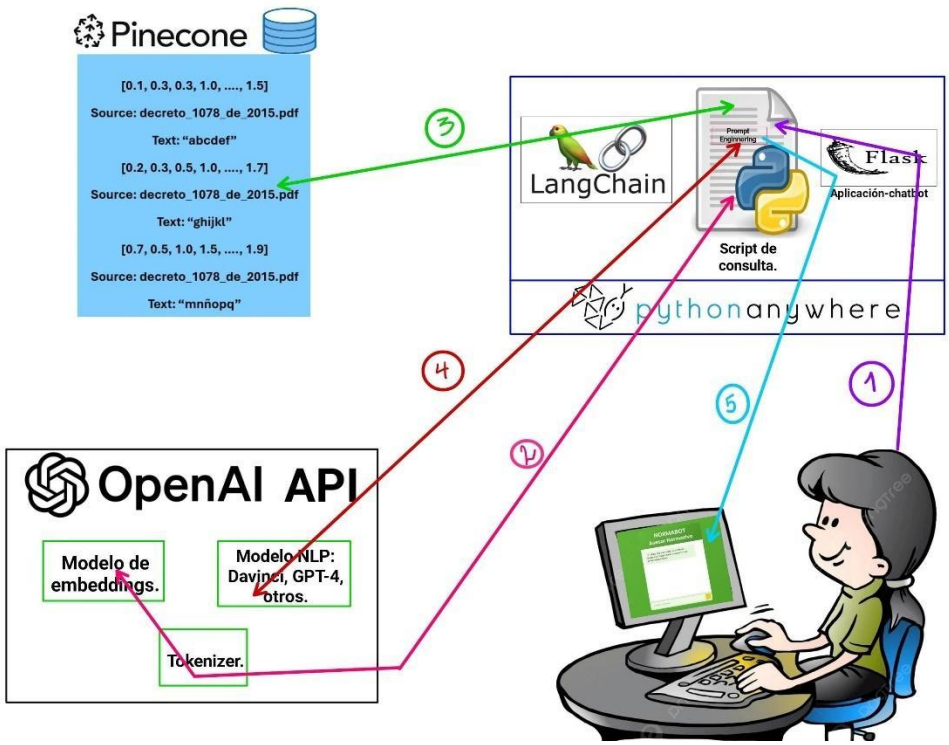
Luego, en el paso 4, estos vectores generados por OpenAI se almacenan automáticamente en la base de datos Pinecone, facilitando las búsquedas rápidas y precisas basadas en el significado de las consultas y no solo en palabras exactas.

Todo este proceso es completamente automático: cada 24 horas el sistema revisa y actualiza la base de datos en Pinecone con cualquier novedad encontrada en la carpeta de Google Drive, asegurando que siempre se cuente con la información normativa más reciente y lista para ser consultada por el chatbot..

3.7.2 Consultas y generación de respuesta

Figura 5.

Servicio que se encarga de procesar consultas y generar respuestas contextualizadas.



Nota. Autoría propia.

El script de consulta, representado en la Figura 5, es el componente encargado de coordinar todo el proceso de generación de respuestas en el sistema. Gracias a la integración con LangChain, este script orquesta el paso a paso necesario para obtener respuestas precisas a las consultas de los usuarios. La interacción con la aplicación Flask se realiza de manera fluida, ya que Flask actúa como el intermediario entre el usuario y el backend. Flask recibe las solicitudes de los usuarios, las envía al script de consulta, y este, a su vez, ejecuta las acciones necesarias para generar la respuesta que se devuelve al usuario en tiempo real.

Todo comienza en el paso 1, cuando el usuario escribe su pregunta en la plataforma web del chatbot. Esta consulta llega automáticamente al núcleo del sistema, una aplicación desarrollada en Python usando Flask y alojada en el servidor como pythonanywhere. Esta aplicación es la responsable de coordinar todo el proceso de respuesta.

El paso 2 inicia cuando el sistema toma la pregunta del usuario y la envía a herramientas de inteligencia artificial, en este caso servicios de OpenAI, los cuales transforman la pregunta en una representación numérica especial llamada embedding o vector. Esta representación numérica especial permite que la pregunta pueda ser comparada por su significado, y no solo por las palabras que contiene. El resultado de esta transformación se envía de vuelta al script de consulta.

En el paso 3, el vector de la pregunta se envía a la base de datos Pinecone, quien almacena las representaciones vectoriales de fragmentos de los documentos normativos. Pinecone se encarga de buscar similitudes vectoriales entre la pregunta y los vectores almacenados, devolviendo la información más cercana y relevante a la pregunta planteada por el usuario. Toda la información recopilada en pinecone es enviada devuelta al script de consulta.

A continuación, en el paso 4, el sistema aplica una técnica fundamental llamada prompt engineering. Aquí se toma la pregunta original del usuario junto con la información recopilada de Pinecone se organizan cuidadosamente en un mensaje estructurado, conocido como prompt, el cual es clave en este proceso, ya que permite a la inteligencia artificial comprender tanto la consulta como su contexto, e incorporar instrucciones específicas para brindar una respuesta coherente, relevante y alineada con las necesidades del usuario. Este mensaje es enviado al modelo avanzado de OpenAI, quien interpreta el contexto, comprende la intención de la consulta y genera una respuesta precisa y fácil de entender.

Por último, en el paso 5, la respuesta generada por el modelo de inteligencia artificial se muestra directamente al usuario en la plataforma web del chatbot. Gracias a este flujo, el sistema responde consultas complejas de forma automática y eficiente, asegurando información actualizada, contextualizada y de calidad.

3.8 Rol de las plataformas y herramientas en la solución

El sistema está compuesto por diferentes herramientas que trabajan de manera coordinada para que la información normativa esté siempre disponible, actualizada y sea fácil de consultar. Cada tecnología cumple un papel específico dentro de la solución. Gracias a esta integración, desde que se actualiza un documento hasta que el usuario recibe una respuesta.

A continuación, se explica de forma sencilla la función de cada plataforma y cómo aporta al funcionamiento del sistema. La Figura 4 y la Figura 5 muestran cómo se relacionan estos componentes en los dos servicios principales del sistema.

3.8.1 PythonAnywhere

Orquestador central y entorno de ejecución backend. Es el núcleo donde se ejecutan los programas principales que controlan el sistema. Desde aquí se programan y gestionan las tareas automáticas que mantienen la información actualizada y permiten que el chatbot esté disponible en la web. Además, PythonAnywhere integra la librería LangChain, que organiza y coordina el procesamiento de los documentos para que la información siempre esté lista y bien estructurada para las consultas.

Esta integración permite que el sistema sea fácil de mantener, pueda adaptarse a nuevas necesidades y siga funcionando correctamente cuando se agregan nuevos tipos de documentos o fuentes de información.

3.8.2 Google Drive

Repositorio normativo dinámico. Funciona como el repositorio principal de documentos normativos, en donde cualquier persona autorizada puede agregar, modificar o eliminar archivos, asegurando que siempre se trabaje con la información más reciente. Este repositorio es la base para todo el proceso automatizado que sigue el sistema.

3.8.3 Google Apps Script

Expone un servicio web personalizado que actúa como detector y extractor de información en los documentos PDF almacenados en la carpeta especifica Google Drive. Su funcionamiento es totalmente automático, ya que cada día es consumido por el script de entrenamiento en Python, el cual le solicita extraer el contenido de todos los documentos presentes en la carpeta.

3.8.4 Script de Entrenamiento en Python

Procesamiento y actualización del índice vectorial. Procesa el contenido extraído de los documentos, realiza la tokenización del contenido y genera embeddings semánticos utilizando la API de OpenAI. Estos vectores son enviados a Pinecone, donde quedan almacenados e indexados, actualizando inmediatamente la información anterior. Este módulo es crucial para asegurar que el sistema pueda recuperar información relevante con rapidez y precisión.

3.8.5 Pinecone

Base de datos vectorial para recuperación semántica. Actúa como el índice semántico del sistema. A través de su API, permite almacenar, actualizar y consultar vectores de alta dimensionalidad. Cuando se formula una pregunta, el sistema genera su correspondiente representación y busca en la base de datos vectorial los fragmentos de información más cercanos en el espacio semántico, facilitando así una recuperación precisa y contextualizada de los datos relevantes.

3.8.6 OpenAI GPT API

Motor de generación de respuestas. Los fragmentos normativos relevantes, junto con la pregunta original del usuario, se utilizan para construir un prompt que se envía a la API de GPT. Este modelo de lenguaje genera una respuesta redactada en lenguaje natural, precisa y contextualizada según la información normativa almacenada.

3.8.7 Frontend Flask

Interfaz del chatbot. Proporciona una interfaz web que permite a los usuarios interactuar con el sistema mediante un navegador. Desde aquí se ingresan las preguntas, se visualizan las respuestas y se controla la experiencia general del usuario. El frontend comunica con el backend Flask a través de solicitudes HTTP.

3.9 Garantía de actualización y calidad en la información

El sistema mantiene su base de conocimiento siempre al día, incorporando automáticamente las versiones más recientes de los documentos normativos, sin depender de revisiones manuales. Cada vez que se añade o modifica un documento en la carpeta central, el sistema lo identifica y actualiza su contenido en la base de conocimiento, asegurando que las respuestas correspondan siempre al estado más vigente de la normativa.

Para garantizar un buen manejo de documentos y que la información sea siempre útil y confiable, se recomienda a las organizaciones implementar las siguientes pautas estratégicas:

- **Calidad y claridad en la información:** Antes de procesar e incorporar cualquier documento a Google Drive, se realiza una revisión básica para garantizar que el texto sea claro, bien estructurado y fiel al contenido original. Los documentos deben tener una organización estructural para facilitar tanto la navegación del usuario como el análisis por parte del sistema.
- **Estructura lógica y jerárquica:** Definir una estructura clara y lógica para los documentos, empleando títulos, subtítulos y secciones bien organizadas. Esto mejora la accesibilidad y comprensión, tanto para las personas como para el modelo de IA.

- **Lenguaje coherente y uniforme:** Se recomienda el uso de un vocabulario consistente, evitando ambigüedades o términos con múltiples significados, para asegurar que las interpretaciones sean uniformes.
- **Contextualización:** Asegurar que cada parte del documento cuente con su contexto definido, lo que ayuda a relacionar temas y facilita la comprensión tanto para el usuario como para el sistema.
- **Eliminación de ambigüedades:** Revisar el contenido para eliminar frases o palabras con doble sentido. Cuando es necesario, se debe aclarar el significado y el contexto, evitando interpretaciones erróneas.
- **Validación humana:** La revisión manual por parte del equipo permite detectar y corregir detalles que podrían pasar desapercibidos, asegurando la calidad y coherencia final del contenido.
- **Aclaración de elementos visuales:** Cuando se incluyen imágenes, tablas o gráficos, es importante acompañarlos de descripciones claras. Esto garantiza que la información sea comprensible incluso en formato solo texto.

Estas medidas aseguran que el prototipo no solo responda rápidamente a las consultas, sino que también entregue información clara y de calidad, brindando confianza a todos los usuarios.

4. Validación de la Solución

La validación para este sistema conversacional fue un proceso fundamental para garantizar la calidad de las respuestas generadas por el chatbot, considerando que la normativa consultada se encuentra en permanente actualización y los modelos de lenguaje evolucionan constantemente. Se diseñó un procedimiento el cual permitiera medir la capacidad del sistema para responder de manera precisa y coherente a las consultas realizadas por los usuarios sobre la normativa sobre el espectro radioeléctrico en Colombia.

El proceso inició con la elaboración de un banco de preguntas representativas, abarcando diferentes niveles de complejidad y aspectos relevantes de la normativa vigente. Cada una de estas preguntas fue ingresada en el chatbot y se registró la respuesta generada por el mismo. Paralelamente, se extrajo manualmente la respuesta considerada correcta directamente de los documentos normativos oficiales, de modo que sirviera como referencia objetiva para la comparación.

Con el fin de obtener un análisis objetivo y evitar el sesgo humano, se implementó un mecanismo de validación automatizada basado en técnicas de similitud semántica. Mediante un script ejecutado en Google Colab y empleando el modelo de embeddings SentenceTransformer (all-MiniLM-L6-v2), donde se calculó la similitud semántica entre cada respuesta proporcionada por el chatbot y la respuesta extraída directamente de los documentos normativos documentada en la hoja de cálculo. Este proceso permitió asignar a cada respuesta una calificación de "Correcta", "Parcial" o "Incorrecta", según el grado de similitud alcanzado. Para facilitar el análisis, los resultados se almacenaron en un documento .xlsx permitiendo visualizar para cada pregunta el valor de similitud obtenido y la evaluación respectiva.

Figura 6.

Análisis de validación semántica.

PREGUNTA	RESPUESTA DEL BOT	LO QUE DEBERIA RESPONDER	Evaluación	Similitud
1. ¿Cuales son los requis	Los requisitos únicos para el d	DECRETO 1370 ARTÍCULO 2.2.2.5.1	Correcta	0.858859003
2. ¿Cómo deberán actua	Por supuesto, puedo ayudarte	LEY 1341 ARTíCULO 8.- LAS TELECO	Correcta	0.862484455
3. ¿Cuales son los plazos	El permiso para el uso del espe	LEY 1341 ARTíCULO 12.- PLAZO Y RI	Correcta	0.801922143
4. ¿Quién realiza la regu	La regulación de precios de los	LEY 1341 ARTíCULO 23.- REGULACIÓ	Parcial	0.737846136
5. Cuales son los criterio	Los criterios para la definición	LEY 1341 ARTíCULO 66.- CRITERIOS	Parcial	0.691473484
6. ¿Qué deben hacer las	Las personas o empresas resp	Resolución 000773 ARTÍCULO 6. OB	Correcta	0.787688494
7. Para el procedimiento	La etapa previa para el proced	DECRETO 4392 Articulo 1°. ETAPA	Correcta	0.882597387
8. Que deben contener l	Las solicitudes para el uso del	DECRETO 4392 Artículo 3°. CONTEI	Correcta	0.775629699
9. Soy una persona natu	Para realizar el registro TIC cor	DECRETO 1078 ARTíCULO 2.2.1.2.1	Parcial	0.731672347
10. Cuales son las causa	Las causas de retiro del registr	DECRETO 1078 ARTiCULO 2.2.1.5.1	Correcta	0.767172694
11. Cómo esta asignado	El uso del espectro en bandas	DECRETO 1078 ARTICULO 2.2.2.1.2	Correcta	0.774173021
12. Cuales son las funcio	Las funciones de la Agencia Na	ARTICULO 2, DECRETO 093 DE 2010	Correcta	0.943642497
13. Como está estructur	La Agencia Nacional del Espect	ARTICULO 3, DECRETO 093 DE 2010	Parcial	0.62854594
14. Cuales son las funcio	Las funciones de la Subdirecció	ARTICULO 7, DECRETO 093 DE 2010	Correcta	0.935431659
15. Cuales son las sancio	Las sanciones en materia de Te	ARTÍCULO 65, LEY 1753 DE 2015----	Correcta	0.866969228
16. En que consiste el fo	El Fondo de Adaptación es una	ARTÍCULO 155, LEY 1753 DE 2015---	Correcta	0.774719894
17. ¿Qué es un estudio d	Un estudio de impacto ambien	ARTÍCULO 57, LEY 1753 DE 2015----	Correcta	0.945848107
18. ¿Cuál es el propósito	El propósito de los sistemas de	ARTÍCULO 233, LEY 1753 DE 2015---	Correcta	0.932921529
19. Cuáles son las infrac	Las infracciones específicas a l	ARTíCULO 64.- INFRACCIONES. Sin	Correcta	0.905972362
20. Cuales son las reglas	Las reglas para los procesos de	ARTICULO 72.- REGLAS PARA LOS PI	Correcta	0.842477918
		PROMEDIO DE PRESICION DEL SISTEMA:		**0.8224024**

Nota. Autoría propia.

La validación arrojó que la gran mayoría de las respuestas generadas por el sistema fueron clasificadas como "Correcta", alcanzando un promedio de similitud de 0.822 entre las respuestas del bot y las respuestas extraidas de la normativa. Las respuestas calificadas como "Parcial" correspondieron, en su mayoría, a situaciones en las que la pregunta formulada era ambigua, incompleta o requería un nivel de detalle adicional no incluido de manera explícita en la base documental del sistema. No se detectaron respuestas clasificadas como "Incorrecta", lo que sugiere un desempeño optimo del sistema en términos de comprensión, contextualización y recuperación de la información normativa. También se identificaron algunas limitaciones técnicas, puesto que el resultado depende en gran medida de la calidad y precisión de las respuestas de referencia extraídas manualmente de los documentos normativos; si estas no son lo suficientemente claras o completas, pueden generarse falsos positivos o negativos en la evaluación automática.

En resumen, este enfoque de validación no solo permitió comprobar el funcionamiento actual del prototipo, sino que también ofrece una metodología replicable y escalable que puede aplicarse en futuras actualizaciones tanto de los documentos normativos como de los modelos de lenguaje empleados. La integración del flujo de validación con herramientas como Google Colab y hojas de cálculo en formato .xlsx facilita la ejecución de este tipo de procesos, intentando mantener la calidad y confiabilidad del sistema conversacional a lo largo del tiempo.

5. Conclusiones

Durante la ejecución de este proyecto se logró cumplir satisfactoriamente con los objetivos planteados, consolidando avances significativos en cada una de las etapas. La implementación del prototipo de chatbot inteligente basado en la API de GPT no solo fue técnicamente exitosa, sino que también respondió de manera efectiva a los requerimientos funcionales y no funcionales establecidos por los actores involucrados, TESAmerica y el grupo de investigación RadioGIS.

En primer lugar, la definición y cobertura de requisitos técnicos permitió garantizar una integración robusta del sistema en la nube, apoyada en tecnologías serverless, el uso de Google Drive como fuente dinámica de entrenamiento y Pinecone como base de datos vectorial. Esta arquitectura facilita la automatización del proceso de ingesta, procesamiento, vectorización y consulta de documentos normativos, asegurando respuestas contextualmente pertinentes y actualizadas.

Uno de los principales logros del sistema desarrollado es su capacidad de actualización continua sin intervención manual, al realizar la detección automática diariamente de nuevos documentos normativos almacenados en una carpeta compartida de Google Drive. Este enfoque permite mantener vigente la base de conocimiento del chatbot, alineándose con la necesidad crítica de precisión y actualidad en el dominio de la regulación del espectro radioeléctrico en Colombia.

La integración de servidores web mediante plataformas como Google Apps Script ha optimizado la supervisión de la carpeta en el repositorio central, evitando la necesidad de permisos de acceso para Google Drive y permitiendo la activación eficiente de los flujos de entrenamiento y consulta. Esta solución no solo mejora los procesos existentes, sino que también establece una base sólida para futuras optimizaciones. Esto incluye la capacidad de escalar el sistema, adaptarlo a otros marcos informativos y asociarlo con nuevos modelos y fuentes de información.

Finalmente, el despliegue del chatbot en una interfaz web funcional permitió comprobar su utilidad práctica, facilidad de uso y en un ambiente simulado de consultas reales. Esta implementación no solo evidenció el cumplimiento de los objetivos técnicos del proyecto, sino que también demostró la capacidad de la solución para transformar la manera en que se accede a normativas, facilitando su consulta de forma eficiente y contextualizada.

En resumen, este trabajo de grado no solo presenta un prototipo funcional, sino que también propone una metodología estructurada y replicable para el desarrollo de soluciones inteligentes basadas en procesamiento de lenguaje natural, aplicable a diversas temáticas más allá del ámbito normativo. Este aporte abre nuevas oportunidades para la creación de herramientas innovadoras en la industria de las telecomunicaciones, así como en otros sectores que demandan acceso eficiente a información técnica, abundante y especializada.

6. Recomendaciones

A partir de la experiencia adquirida en el desarrollo y validación del prototipo conversacional, se identifican diversas oportunidades y retos que pueden orientar futuros trabajos en este campo. Una de las principales recomendaciones consiste en fortalecer la actualización continua de la información documental, implementando mecanismos automáticos de monitoreo y alerta que permitan identificar de manera inmediata cambios en la normativa como la adición, edición o eliminación de archivos. Esto permitiría mantener actualizada la base de datos vectorial de forma eficiente, precisa y sin ejecutar tareas innecesarias.

Adicionalmente, se recomienda explorar la posibilidad de utilizar herramientas de automatización disponibles en el mercado, como Make, n8n o Zapier, las cuales pueden reducir significativamente los procesos manuales y optimizar los tiempos en tareas repetitivas, como por ejemplo los procesos de validación del sistema. La integración de estos recursos no solo simplificar el mantenimiento, sino que también incrementa la eficiencia operativa y facilita la escalabilidad de la solución de manera general.

En lo relacionado con la arquitectura, es recomendable evolucionar hacia una estructura basada en microservicios desacoplados, lo que permitiría desplegar componentes independientes, escalar módulos específicos según la demanda y facilitar el mantenimiento individualizado sin afectar el conjunto del sistema.

Finalmente, se recomienda realizar revisiones periódicas de los modelos de lenguaje, las herramientas utilizadas y las nuevas tecnologías que surjan en el campo de la inteligencia artificial. Esto permitirá asegurar que el sistema conversacional se mantenga actualizado frente a los avances tecnológicos, incorporando oportunamente mejoras en precisión, eficiencia y adaptabilidad.

Referencias Bibliográficas

Alberts, I. L., Mercolli, L., Pyka, T., Prenosil, G., Shi, K., Rominger, A., & Afshar Oromieh, A. (2023). Large language models (LLM) and ChatGPT: what will the impact on nuclear medicine be? European Journal of Nuclear Medicine and Molecular Imaging, 50, 1549–1552. https://link.springer.com/article/10.1007/s00259-023-06172-w

Amazon Web Services. (2023). AWS Lambda documentation. https://docs.aws.amazon.com/lambda/

Rueda Rodríguez, M., & Hernández Prince, C. A. (2024). Diseño e implementación de un bot de charla basado en GPT para la acreditación internacional de los programas de pregrado de la E3T (Trabajo de grado, Universidad Industrial de Santander). Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones, Facultad de Ingenierías Fisicomecánicas.

Chase, H., et al. (2023). LangChain: Building Applications with LLMs through Composition [White paper]. LangChain.

Chen, B., Zhang, Z., Langrené, N., & Zhu, S. (2023). Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review. arXiv. https://doi.org/10.48550/arXiv.2310.14735

CloudRaft. (2025). Top 5 vector databases in 2025. https://www.cloudraft.io/blog/top-5-vector-databases

Cortez Vásquez, A., Vega Huerta, H., & Pariona Quispe, J. (2009). Procesamiento de lenguaje natural. Revista de Ingeniería de Sistemas e Informática, 6(2), 45–49. https://d1wqtxts1xzle7.cloudfront.net/77493941/5121-libre.pdfDataCamp. (2025). The top vector databases in 2025. https://www.datacamp.com

Gamallo Otero, P., & García González, M. (2012). Técnicas de Procesamiento del Lenguaje Natural en la Recuperación de Información. Novática, (217), 33–40. https://minerva.usc.es/entities/publication/e7e52a05-29e6-401b-abf1-52655d3456b0

García-Peñalvo, F. J., Llorens-Largo, F., & Vidal, J. (2024). La nueva realidad de la educación ante los avances de la inteligencia artificial generativa. RIED-Revista Iberoamericana de Educación a Distancia, 27(1). https://www.redalyc.org/journal/3314/331475280001/331475280001.pdf

Google Developers. (2023). Google Apps Script Overview. https://developers.google.com/apps-script

Gowda, S., Bhattacharya, D., & Shah, P. (2023). Evaluating Open-Source Vector Databases for LLM-Based Retrieval. arXiv preprint arXiv:2307.12476. https://arxiv.org/abs/2307.12476

Jeong, J., Gil, D., Kim, D., & Jeong, J. (2024). Current Research and Future Directions for Off-Site Construction through LangChain with a Large Language Model. Buildings, 14(8), 2374. https://www.mdpi.com/2075-5309/14/8/2374

Miah, A. S. M., et al. (2025). ChatGPT in Research and Education: A SWOT Analysis of Its Academic Impact. Computer Modeling in Engineering & Sciences, 143(3), 2574–2582. https://www-sciencedirect-com.bibliotecavirtual.uis.edu.co/org/science/article/pii/S1526149225001614

Microsoft Azure. (2023). Azure Functions documentation. https://learn.microsoft.com/en-us/azure/azure-functions/

Murf AI. (2025, julio). 7 best vector databases in 2025. https://murf.ai/blog/best-vector-databases

OpenAI. (2024a). Text embeddings guide. https://platform.openai.com/docs/guides/embeddings

OpenAI. (2024b). GPT-4 Technical Report. https://openai.com/research/gpt-4

Ortiz Laverde, S. M., & Herrera Zapata, L. M. (2024). Elementos necesarios para la economía digital: el espectro radioeléctrico, infraestructura y redes. Prolegómenos, 27(53), 87–106. https://revistas.umng.edu.co/index.php/dere/article/view/6984/5839

Ozkaya, M. (2025, enero). Exploring vector databases. Medium. https://mehmetozkaya.medium.com

Park, Y., et al. (2024). Design of REST API Client for Conversational Agent using Large Language Model with Open API System. https://ieeexplore-ieee-org.bibliotecavirtual.uis.edu.co/stamp/stamp.jsp?tp=&arnumber=10685639

Pinecone. (2023). Vector database for semantic search. https://www.pinecone.io

PythonAnywhere. (2023). Web app hosting. https://www.pythonanywhere.com/

Reddit user bornforspace. (2024). "Qdrant does..." r/vectordatabase. https://reddit.com

Rivero Panaqué, C., & Beltrán Castañón, C. (2024). La inteligencia artificial en la educación del siglo XXI: avances, desafíos y oportunidades. Educación, 33(64), 5–7. http://www.scielo.org.pe/pdf/educ/v33n64/2304-4322-educ-33-64-5.pdf

Salyuk, D., Zhornitsky, A., & Gritsenko, D. (2022). Búsqueda de respuestas utilizando redes neuronales. Journal of Artificial Intelligence Research, 58(3), 215-234. https://riunet.upv.es/server/api/core/bitstreams/a375bf4e-ff81-4a96-b7e5-43752d8aca4e/content

Shakudo. (2025, junio). Top 9 vector databases as of June 2025. https://shakudo.io/blog/top-9-vector-databases

Singh, V., Rohith, Y., Prakash, B., & Kumari, U. (2023). ChatBot using Python Flask. IEEE ICICCS-2023, 1182-1185. https://ieeexplore-ieee-org.bibliotecavirtual.uis.edu.co/stamp/stamp.jsp?tp=&arnumber=10142484

Srivastava, V., & Beri, G. (2024). Advanced Techniques in Prompt Engineering for Large Language Models: A Comprehensive Study. https://ieeexplore-ieee-org.bibliotecavirtual.uis.edu.co/stamp/stamp.jsp?tp=&arnumber=10911672

Xiao, J., Wang, L., Cheng, Y., Zhang, J., Hu, J., Tan, S., Su, Y., & Zhou, H. (2023). Web Front-end Development based on Flask Architecture for Image Recognition. IEEE ITOEC, 979-984. https://ieeexplore-ieee-org.bibliotecavirtual.uis.edu.co/stamp/stamp.jsp?tp=&arnumber=10291828

Zakir, M. H., Bashir, S., Ali, R. N., & Khan, S. H. (2024). Artificial Intelligence and Machine Learning in Legal Research: A Comprehensive Analysis. Qlantic Journal of Social Sciences, 5(1), 307–317. https://qjss.com.pk/index.php/qjss/article/view/252/236

Apéndices

Apéndice A. Carpeta con el código completo de la interfaz web del chatbot inteligente.

Apéndice B. Código del web service que detecta nuevos documentos en la carpeta seleccionada, activado cada vez que es invocado por el script de entrenamiento.

Apéndice C. Código completo ("Script de entrenamiento") para el entrenamiento del sistema, responsable de mantener actualizada la base de datos vectorial.

Apéndice D. Cuaderno de Google Colab utilizado para validar las respuestas del chatbot mediante pruebas de similitud semántica

Apéndice E. Manual técnico detallado del script de entrenamiento.

Apéndice F. Manual técnico detallado del web service, explicando su arquitectura, funcionamiento y despliegue.

Apéndice G. Manual sobre la arquitectura general y el funcionamiento del chatbot inteligente.

Apéndice H. Manual de uso y configuración del entorno de despliegue en PythonAnywhere.

Apéndice I. Archivo en formato Excel con los resultados del análisis de similitud semántica entre las respuestas generadas por el chatbot y las respuestas esperadas, utilizado como parte del proceso de validación del sistema.

Apéndice J. Google Sheet con el listado de preguntas utilizadas para evaluar el rendimiento del chatbot, formuladas a partir del contenido de los documentos normativos del espectro radioeléctrico en Colombia.