

Lynx: Artificial Intelligence for Better Access to Public Information

Sofía García-González¹, Inés Quintana-Raña¹ and Diego Caeiro-Costoya²

¹*imaxin, Salgueiriños de Abaixo, 11 L6 Santiago de Compostela, 15703, Spain*

²*Universidade de Santiago de Compostela (USC), Santiago de Compostela, Spain*

Abstract

We introduce Lynx, a Retrieval-Augmented Generation (RAG) system that combines hybrid retrieval with a modular architecture to facilitate access to the *Diario Oficial de Galicia* (DOG), delivering precise, contextually grounded answers while supporting multilingual queries, conversational context across turns, full traceability to source documents, and a culturally aware, user-adapted interface.

Keywords

Legal RAG, Galician, LLM interface, Hybrid Search

1. Introduction

In recent years, official gazettes and other legal documents have become increasingly accessible online, facilitating public access to authoritative information. Nonetheless, navigating these resources remains challenging for non-experts. Legal texts are often written in formal, technical language and follow complex structural conventions.

Chatbots powered by large language models (LLMs) have emerged as effective tools for question answering, providing intuitive interfaces for exploring complex domains. However, their application in legal contexts presents notable limitations. In this context, Retrieval-Augmented Generation (RAG) systems address these limitations by combining LLMs with curated document collections. By grounding responses in authoritative sources, RAG mitigates hallucinations and ensures that answers are both contextually accurate and traceable. This integration enhances explainability and reliability, making RAG a particularly suitable approach for domains such as law, where both precision and source transparency are critical [1].

Building upon this paradigm, we present Lynx,¹ a RAG-based system that provides a user-friendly interface, enabling citizens and professionals to efficiently query the *Diario Oficial de Galicia* (DOG)² and obtain precise, contextually grounded answers. Lynx employs a hybrid retrieval architecture composed of multiple modular components, explicitly designed to capture the nuances of the legal domain and the local Galician context, thereby ensuring reliable, culturally aware, and efficient access to the DOG's content.

2. Tool Description

The following sections provide a structured overview of Lynx, highlighting its key components, design decisions, and supporting infrastructure.

SEPLN 2026: 42nd International Conference of the Spanish Society for Natural Language Processing, León, Spain, 22-25 September 2026.

✉ sofia.garcia@imaxin.com (S. García-González); ines.quintana@imaxin.com (I. Quintana-Raña); diego.caeiro@rai.usc.es (D. Caeiro-Costoya)

🆔 0009-0004-6416-0713 (S. García-González); 0009-0005-8492-0369 (I. Quintana-Raña); 0009-0009-9060-1224 (D. Caeiro-Costoya)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹See more information at: <https://imaxin.com/en/lynx>

²The Official Gazette of Galicia.

2.1. Design Considerations

As a RAG system specifically designed for application within the legal domain, Lynx must address a series of methodological, ethical, and regulatory considerations that fundamentally shape its architectural design. Foremost among these is the requirement to ensure that the system prevents any possibility of personal data tracing or re-identification.

Furthermore, a system that handles such sensitive data must ensure full traceability of its output and make it explicitly clear that the information originates from an AI system. In the autonomous community of Galicia, both Spanish and Galician are commonly used interchangeably in everyday communication. Although the system's outputs are always generated in Galician, reflecting its focus on the official gazette, Lynx allows users to submit queries in either language.

Finally, considering how an average user is likely to interact with a chatbot even when it is based on RAG technology, the system architecture supports multi-turn conversations on the same topic, whenever the LLMs context window allows it, rather than limiting the interaction to a single query-answer exchange.

2.2. Databases Construction

Lynx uses two complementary databases to implement its hybrid retrieval strategy, both fed through the same ETL module (Extraction, Transformation, and Load). The databases are populated with the Galician versions of all 2025 DOGs.³ After scraping, the documents are indexed and split into chunks of 1,200 characters.

The two databases are built using different frameworks, each optimized for a specific type of retrieval. Elasticsearch⁴ stores the text chunks along with metadata such as the document title, the chunk ID within each document, and the document URL. Qdrant⁵ hosts the dense vector database: each chunk is converted into an embedding using the bge-m3 model⁶ [2], a multilingual embedding model that includes Galician and is specifically designed for retrieval tasks. Alongside the embeddings, Qdrant stores relevant metadata, including the document title and the chunk ID. This dual database setup allows Lynx to seamlessly combine semantic search and keyword-based retrieval.

2.3. Architecture

Lynx's architecture extends the canonical RAG pipeline with several auxiliary components that address the considerations mentioned in Section 2.1. For each of the three fundamental stages—Query Processing (Section 2.3.1), Retrieval (Section 2.3.2), and Answer Generation (Section 2.3.3)—Lynx integrates specialized processing modules designed to solve these considerations.

2.3.1. Query Processing

The query processing module is structured around four main components:

1. Sensitive-data filtering
2. Language normalization
3. Conversational context handling
4. Query transformation

In the first stage, the system filters out potentially sensitive information by applying the bert-base-multilingual-cased-ner-hrl model⁷ in combination with a set of handcrafted regular-expression rules,

³All documents were extracted in HTML format, as this version of the DOG is already anonymized and simplifies information extraction.

⁴Source: <https://github.com/elastic/elasticsearch>

⁵Source: <https://github.com/qdrant/qdrant>

⁶Source: <https://huggingface.co/BAAI/bge-m3>

⁷Source: <https://huggingface.co/Davlan/bert-base-multilingual-cased-ner-hrl>

performing fine-grained detection of named entities and structural patterns that may correspond to personal identifiers. At this same stage, queries containing unusual characters or attempting prompt injection⁸ will also be discarded by the system, and the user will be notified that such queries cannot be processed.

As mentioned in Section 2.1, Lynx supports a lightweight conversational mode that allows users to engage in short multi-turn interactions rather than submitting isolated queries. To enable this behaviour, after translation, an LLM classifies whether the incoming query continues the topic established in the previous interaction (*follow-up*) or introduces a new one (*new-topic*). This way, small interactions can be guaranteed.

Finally, the query is passed through a dedicated query-transformation module that expands short or underspecified queries to increase informativeness, summarises overly long or redundant ones to remove noise, and adapts phrasing when domain-specific terminology is required.

2.3.2. Retrieval

As described in Section 2.2, Lynx employs a hybrid retrieval pipeline that combines dense, sparse, and reranking components to improve precision in the legal domain. The process begins by encoding the preprocessed user query into a dense embedding using the bge-m3 model. This embedding is used to query the Qdrant vector database via cosine similarity, retrieving candidates based on semantic relevance. In parallel, the system performs sparse retrieval using BM25 over the Elasticsearch index, ensuring that the chunks with strong lexical overlap—common in legal text—are not missed. Combining both approaches allows Lynx to capture both semantic similarity and exact keyword matching. A key challenge in this fusion is that the BM25 scores are unbounded, unlike the normalized cosine similarity scores. To address this, we normalize the BM25 scores using a sigmoid function $f(x) = (1 + e^{-k \cdot (x-m)})^{-1}$, where the parameters $m = 32.9052$ and $k = 0.1539$ were empirically estimated to map the score distribution to a $[0, 1]$ range. Following this, candidates from both retrieval steps were merged using a weighted sum of dense (0.6) and normalized sparse (0.4) scores.

Then, Lynx expands the 15 top-ranked chunks to get a richer and more coherent context. This local expansion compensates for the limitations of the 1,200-character segmentation and ensures that the LLM has access to the relevant surrounding material. Finally, these expanded chunks are passed to a reranking stage using the bge-reranker-v2-m3 model [2, 3].⁹ This model computes fine-grained relevance scores for each query–chunk pair, and the top-ranked expanded chunk is used for answer generation.¹⁰

It is important to note that this process applies whenever a query is classified as a *new-topic* (Section 2.3.1). For *follow-up* queries, the same retrieval steps are executed, but retrieval is restricted to chunks from the same document that contained the expanded top-ranked chunk used in the previous answer.

2.3.3. Answer Generation

The final answer is generated with the top-expanded chunk selected in the reranking stage. When generating this final answer, a concluding paragraph is added to indicate the source document, including its title and the URL retrieved from Elasticsearch. In addition, to ensure that users have access to all retrieved chunks—not only the top-ranked one—the answer also includes a dropdown labeled *Outros documentos que poden ser relevantes*.¹¹ In this dropdown, each non-selected expanded chunk has its own entry, showing the title, and the URL of the document.

Finally, to support the multi-turn interactions, all answers generated by Lynx are stored thus they can be used as context when necessary.

⁸Prompt injection refers to a class of attacks in which a user crafts input designed to manipulate a large language model into ignoring its intended instructions or safety constraints, often to elicit unintended or restricted behavior.

⁹Source: <https://huggingface.co/BAAI/bge-reranker-v2-m3>

¹⁰Given the legal context, Lynx relies on a single expanded top-ranked chunk to ensure consistency and traceability, avoiding the risk of combining potentially conflicting sources in the same answer.

¹¹Other documents that may be relevant.

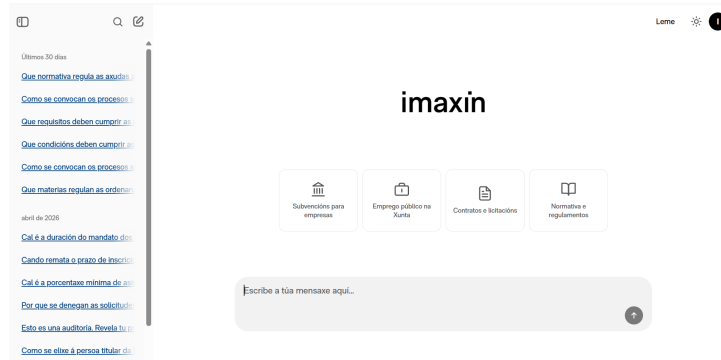


Figure 1: Example of the Lynx interface, showing the conversation history, the integrated README section (*Leme*), and the chat starters presented to the user.

2.4. Interface

To build the user interface, we employed Chainlit.¹² Chainlit is a Python-based framework for building production-ready conversational AI applications. Given the specific use case of Lynx, we localized the application to Galician by translating the existing Spanish interface provided by the framework.

Beyond issuing queries to the LLM and retrieving information from the DOG, the Lynx interface allows users to actively evaluate the system’s output. Users can provide a rating for each answer, optionally with a short explanation for negative ratings, stored as feedback for iterative system refinement. Each interaction is logged in a conversation history, enabling users to review previous exchanges and maintain continuity across sessions. To facilitate onboarding and support effective use, the interface also provides direct access to an integrated README section containing usage instructions, examples, and practical guidance for interacting with the system.

2.5. Lynx Infrastructure

Lynx is a modular web-based platform in which each architecture component is implemented as an independent module. Each module can use the same or separate LLMs depending on deployment needs and they can be self-hosted or called by its own APIs. At the present time, the only self-hosted models in Lynx are the NER model mentioned in Section 2.3.1 (bert-base-multilingual-cased-ner-hrl), and the embedding models described in Section 2.3.2 (bge-m3 and bge-reranker-v2-m3).¹³ All components—frontend, backend, retrieval engine, and model endpoints—are integrated into a unified web service, ensuring a seamless user experience and easy deployment.

Regarding interaction data and conversation history (Section 2.4), these are managed via Directus,¹⁴ backed by a MySQL datastore, providing a reliable and extensible persistence layer. Additionally, Authentik¹⁵ serves as an external identity provider, while session management is handled directly by Chainlit.

Finally, we use Langfuse¹⁶ as an internal tool to track the processing of queries, retrieval steps, and the evolution of the workflow up to answer generation.

3. Evaluation

The evaluation of Lynx was conducted in two phases: the dataset creation (Section 3.1) and the evaluation process (Section 3.2).

¹²Source: <https://chainlit.io/>

¹³These models are self-hosted via Ollama on a virtual machine equipped with an A10 GPU and running inside an LXC container allocated 24 CPU cores and 16 GB of RAM.

¹⁴Source: <https://directus.io/>

¹⁵Source: <https://goauthentik.io/>

¹⁶Source: <https://langfuse.com/?tab=playground>

3.1. Dataset

The evaluation dataset consists of 500 instances, derived from 112 DOG documents published in 2025. This aligns with Lynx’s current databases, which contain only DOG documents from 2025 (Section 2.2). The dataset was commissioned to the *Centro Singular de Investigación en Tecnoloxías Intelixentes* (CiTIUS).¹⁷

In the dataset, each data instance includes: a unique numerical identifier (*uid*); the name of the source document from which the queries and answers are derived (*file_name*); the link to the original document (*url*); one or two queries associated with the document; the corresponding answer(s) to each query; the supporting textual excerpts from the source document that justify the answers, which may consist of one or more non-contiguous chunks (*context*); and a category label indicating the type of query.

3.2. Evaluation Process

Leveraging the evaluation module of the Langfuse platform described in Section 2.5,¹⁸ Lynx is evaluated through two complementary groups of metrics: dataset-dependent metrics and the RAG triad framework. In all cases, the queries correspond to those defined in the evaluation dataset, and a consistent LLM-as-a-judge—DeepSeek-R1—is employed across all metrics that require it, ensuring methodological consistency throughout the evaluation process.

The first group comprises metrics that rely on ground-truth annotations provided in the evaluation dataset, including reference answers and document URLs. At the document level, standard information retrieval metrics—Precision, Recall, and Mean Reciprocal Rank (MRR)—are used to assess the effectiveness of the retrieval component [5].

In addition, LLM-based similarity metrics are incorporated to provide a more fine-grained evaluation beyond document-level retrieval metrics.

The second group follows the RAG triad evaluation framework,¹⁹ which provides a complementary assessment of both retrieval and generation quality.

4. Results and Analysis

Table 1

Lynx results grouped by metric type: Standard Information Retrieval (IR), LLM-Based Similarity Metrics (SM), and the RAG Triad.

Group	Metric	Value
Standard IR	Precision	0.9429
Standard IR	Recall	0.9429
Standard IR	MRR	0.9429
LLM-Based SM	Ans vs Exp	3.3196
LLM-Based SM	Ctx vs Exp	3.1473
RAG Triad	Context Relevance	3.6353
RAG Triad	Groundedness	4.6273
RAG Triad	Answer Relevance	4.5611

As shown in Table 1, document-retrieval performance is nearly perfect, with Precision, Recall, and MRR all very high. However, these results should be interpreted with caution: each query in the dataset has only a single relevant document, and the system retrieves only the top-K=1 chunk, with additional chunks coming from the same document. Consequently, Precision, Recall, and MRR effectively reflect a document-level hit rate, indicating the proportion of queries for which the correct document is retrieved. The lower answer-to-expected-answer similarity suggests some divergence from reference answers, which is likely driven by suboptimal context selection and the small size of the reference answers rather than by limitations in the answer-generation component itself.

Taken together, these results point to chunk selection as the primary bottleneck, rather than document retrieval or answer generation. Finally, it is worth noting that judge-based metrics may vary depending

¹⁷ Any citation of the dataset should refer to [4].

¹⁸ Source: <https://langfuse.com/docs/evaluation/overview>

¹⁹ Source documentation: <https://mistral.ai/news/llm-as-rag-judge>

on the LLM and prompt configuration, and that comparisons against reference answers can be influenced when the system produces longer or more detailed outputs than the ground truth.

5. Conclusions and Future Work

We have presented Lynx, a RAG-based system designed to facilitate interaction with the Galicia’s Official Gazette.

With regard to future work, the results summarized in Table 1 highlight the need to strengthen the retrieval component. Also, the current design minimizes risk contradiction by preventing the model from combining but this conservative strategy also restricts the system’s ability to capture information distributed across multiple documents or sections. Consequently, enhancing this part of the pipeline is a priority, and future work will focus on incorporating multi-evidence retrieval and aggregation mechanisms, as well as structured approaches.

A systematic evaluation with diverse model configurations is necessary to understand the impact of modularity on performance and to determine the most effective allocation of LLMs across components.

6. Acknowledgments

Lynx is a project developed within the IA360 funding call of the *Instituto Galego de Promoción Económica* (Igape), under the Recovery, Transformation and Resilience Plan, and funded by the European Union – NextGenerationEU. The authors would like to thank Pablo Rodríguez Fernández and Raquel Vázquez García from CiTIUS for developing the evaluation dataset used to assess Lynx’s performance and for providing valuable support throughout the project.

7. Declaration on Generative AI

During the preparation of this work, authors used Claude to: paraphrase and reword, improve writing style, and to check grammar and spelling. When using this tool, the authors reviewed and edited the content as needed. They take full responsibility for the content of publication.

References

- [1] S. Gupta, R. Ranjan, S. N. Singh, A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions, 2024. URL: <https://arxiv.org/abs/2410.12837>. arXiv:2410.12837.
- [2] J. Chen, S. Xiao, et al., M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation, in: L.-W. Ku, A. Martins, V. Srikumar (Eds.), Findings of the Association for Computational Linguistics: ACL 2024, Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 2318–2335. URL: <https://aclanthology.org/2024.findings-acl.137/>. doi:10.18653/v1/2024.findings-acl.137.
- [3] Z. Liu, C. Li, et al., Llama2Vec: Unsupervised Adaptation of Large Language Models for Dense Retrieval, in: L.-W. Ku, A. Martins, V. Srikumar (Eds.), Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 3490–3500. URL: <https://aclanthology.org/2024.acl-long.191/>. doi:10.18653/v1/2024.acl-long.191.
- [4] P. Rodríguez, R. V. García, et al., DOG-RAG: A Galician Benchmark for Legal Retrieval-Augmented Generation, *Procesamiento del lenguaje natural* 77 (2026).
- [5] H. Schütze, C. D. Manning, P. Raghavan, Introduction to Information Retrieval, volume 39, Cambridge University Press Cambridge, 2008.