**Metodologia**

A metodologia deste trabalho foi desenhada para transformar documentos públicos do Sistema Eletrônico de Execução Unificada (SEEU) em uma base de dados estruturada, com objetivo de integrá-los a um modelo de linguagem natural por meio da técnica de Retrieval-Augmented Generation (RAG), permitindo consultas inteligentes, contextuais e juridicamente relevantes.

**1. Coleta e organização dos dados**

O primeiro passo consistiu em coletar documentos oficiais e públicos do SEEU, como resoluções, notas técnicas, manuais e relatórios emitidos pelo Conselho Nacional de Justiça (CNJ). Esses arquivos foram organizados em uma estrutura de diretórios por tipo de documento, data e origem, a fim de facilitar o seu processamento posterior, seguindo práticas recomendadas de engenharia de dados (Kleppmann, 2017).

**2. Pré-processamento dos dados**

Os documentos foram convertidos para texto limpo e normalizado. Esse processo envolveu:

* Remoção de caracteres especiais e espaços redundantes;
* Padronização de acentuação e formatação;
* Correção de erros de OCR (Reconhecimento Óptico de Caracteres), nos casos de arquivos digitalizados (Smith, 2007).

O conteúdo textual foi segmentado em parágrafos ou seções, permitindo que cada fragmento possa ser vetorizado individualmente, garantindo granularidade na recuperação posterior.

**3. Vetorização do conteúdo**

A vetorização dos textos foi realizada utilizando modelos de embeddings do tipo all-MiniLM-L6-v2 (Wang et al., 2020) e BAAI/bge-base, ambos compatíveis com a biblioteca sentence-transformers, que produzem representações vetoriais semânticas de alta qualidade. Isso possibilita que buscas futuras sejam baseadas em significado, não apenas em palavras-chave.

Os vetores gerados foram armazenados em um banco de dados vetorial para uso na pipeline de recuperação.

**4. Escolha do banco de dados vetorial**

Foram avaliadas três principais alternativas para armazenamento e recuperação vetorial:

* **Elasticsearch**: já consolidado em sistemas de busca, com suporte a vetores desde a versão 7.3.
* **FAISS**: desenvolvido pelo Facebook AI Research, ideal para buscas extremamente rápidas, mas com menor flexibilidade de integração (Johnson et al., 2019).
* **PostgreSQL com extensão pgvector**: solução robusta e gratuita, com excelente compatibilidade com Python e outras bibliotecas open-source.

A opção adotada foi o uso do **PostgreSQL com a extensão pgvector**, por ser leve, eficiente, gratuito e de fácil implantação em contextos acadêmicos e prototipagem rápida (Zhang et al., 2022).

**5. Integração com LLM via RAG**

A integração com um modelo de linguagem natural será feita utilizando a técnica **RAG (Retrieval-Augmented Generation)** (Lewis et al., 2020). O processo é dividido em etapas:

1. O usuário envia uma pergunta em linguagem natural;
2. A pergunta é convertida em um vetor usando o mesmo modelo de embeddings da base;
3. Os vetores mais similares são recuperados do banco;
4. Os textos associados aos vetores são enviados como contexto para o modelo;
5. O LLM gera uma resposta embasada nesse conteúdo.

Esse fluxo permite que o modelo gere respostas precisas com base no conteúdo jurídico do SEEU, sem precisar armazenar todo o conhecimento internamente.

**6. Ferramentas e tecnologias utilizadas**

As ferramentas utilizadas neste projeto foram selecionadas com foco em acessibilidade, reprodutibilidade e custo zero:

* **Python 3.11+** – Linguagem principal para manipulação de dados e orquestração;
* **Transformers e Sentence-Transformers (HuggingFace)** – Embeddings e modelos de linguagem;
* **LangChain** – Integração e gerenciamento da pipeline RAG;
* **PostgreSQL + pgvector** – Armazenamento vetorial;
* **VS Code + GitHub** – Versionamento e controle de código-fonte;
* **Jupyter Notebook** – Prototipagem e testes.

**Referências**

* Johnson, J., Douze, M., & Jégou, H. (2019). *Billion-scale similarity search with GPUs*. arXiv preprint arXiv:1702.08734.
* Kleppmann, M. (2017). *Designing Data-Intensive Applications*. O’Reilly Media.
* Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Riedel, S. (2020). *Retrieval-augmented generation for knowledge-intensive NLP tasks*. Advances in Neural Information Processing Systems.
* Smith, R. (2007). *An overview of the Tesseract OCR engine*. In Ninth International Conference on Document Analysis and Recognition (Vol. 2, pp. 629-633). IEEE.
* Wang, S., Reimers, N., & Gurevych, I. (2020). *Making monolingual sentence embeddings multilingual using knowledge distillation*. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).
* Zhang, H., Wang, Y., & Lin, C. (2022). *pgvector: Bringing vector search to PostgreSQL*. PostgreSQL Conference Proceedings.