CATOLICA DE SANTA CATARINA DE JARAGUÁ DO SUL

INTELIGÊNCIA ARTIFICIAL E GENERATIVA

Documento RFC

# OraculumKB: Chatbot com RAG Sobre Bases de Conhecimento

Sandro Cesar Camara

Jaraguá do Sul

Setembro de

2024

## Resumo

Este documento descreve o desenvolvimento de um chatbot com Retrieval-Augmented Generation (RAG), projetado para responder a perguntas dos usuários sobre informações contidas em documentos PDF. A base de conhecimento será criada a partir de embeddings extraídos desses documentos e armazenada em um banco de dados vetorial utilizando Qdrant. O chatbot utilizará a plataforma Ollama para utilizar modelos de llm localmente que irá gerar respostas com base nas informações recuperadas. A interface será construída em Streamlit, permitindo uma interação amigável com os usuários.

# 1. Introdução

## 1.1. Contexto

Nos últimos anos, as tecnologias de aprendizado de máquina e processamento de linguagem natural evoluíram para permitir que sistemas realizem buscas mais inteligentes e gerem respostas contextuais com base em grandes volumes de dados. O uso de técnicas de RAG se destaca por combinar a recuperação de informações armazenadas com a geração de respostas em linguagem natural, oferecendo resultados precisos e contextualizados.

## 1.2. Justificativa

Organizações que lidam com grandes volumes de documentos, como PDFs, enfrentam dificuldades em extrair e consultar informações de forma eficiente. Este projeto de chatbot visa resolver esse problema ao combinar técnicas de recuperação vetorial com modelos de linguagem natural, permitindo que os usuários obtenham respostas rápidas e precisas a partir de documentos previamente carregados.

## 1.3. Objetivos

- Objetivo Principal:
  - Desenvolver um chatbot que permita ao usuário fazer perguntas sobre documentos PDF, com respostas geradas a partir de um sistema RAG baseado em um banco de dados vetorial.

- Objetivos Secundários:
  - Implementar um pipeline de processamento de PDFs para extração de embeddings.
  - Armazenar esses embeddings em Qdrant para consultas vetoriais rápidas.
  - Utilizar LLM localmente na plataforma Ollama para gerar respostas baseadas nas informações recuperadas.

# 2. Descrição do Projeto

## 2.1. Tema do Projeto

O projeto visa criar um chatbot baseado em RAG, que utiliza embeddings gerados a partir de PDFs e os armazena em um banco vetorial. O chatbot será capaz de processar perguntas dos usuários e gerar respostas contextuais, utilizando a plataforma Ollama para linguagem natural.

## 2.2. Problemas a Resolver

Busca Ineficiente em PDFs: Sistemas de busca tradicionais são limitados em identificar contextos complexos em grandes volumes de dados não estruturados.

Interpretação de Dados: O chatbot deve interpretar o conteúdo dos PDFs e gerar respostas coerentes a partir de perguntas em linguagem natural

# 3. Especificação Técnica

A proposta detalha os requisitos funcionais e não funcionais bem como a arquitetura utilizada e as bibliotecas que compõem o projeto.

## 3.1. Requisitos

- Requisitos Funcionais

| Código | Descrição |
|---|---|
| RF001 | O sistema deve ser capaz de processar documentos PDF, extraindo seus embeddings. |
| RF002 | O sistema deve armazenar os embeddings em um banco vetorial utilizando Qdrant. |
| RF003 | O chatbot deve ser capaz de receber perguntas e gerar respostas utilizando LLM localmente através da plataforma Ollama com base nas informações recuperadas. |
| RF004 | A interface gráfica deve ser construída em Streamlit, permitindo fácil interação. |

- Requisitos não funcionais

| Código | Descrição |
|---|---|
| RNF001 | O sistema deve ser capaz de lidar com grandes volumes de documentos PDF. |
| RNF002 | A resposta gerada pelo chatbot deve ser entregue em tempo real (inferior a 2 segundos para respostas simples). |
| RNF003 | O banco vetorial deve ser otimizado para consultas rápidas e eficientes. |
| RNF004 | O sistema deve ser modular para permitir futuras expansões. |

## 3.2. Arquitetura

A infraestrutura para a implementação deste projeto será baseada em uma arquitetura modular, com cada componente executado de forma eficiente e escalável. O sistema será desenvolvido em Python, que servirá como a base para a implementação da lógica de processamento de dados e integração com o banco vetorial e os modelos de linguagem. O armazenamento vetorial será gerenciado pelo Qdrant, rodando em um ambiente de containerização via Docker, permitindo a escalabilidade e facilidade de manutenção do banco de dados. Para a geração de respostas em linguagem natural, será utilizado o Ollama Desktop, configurado com modelos de última geração como Llama 3.1 e o modelo de embeddings Nomic-embed, ambos otimizados para gerar respostas contextuais precisas e realizar consultas vetoriais. Essa combinação proporciona uma arquitetura robusta e flexível para manipulação de grandes volumes de dados em tempo real.

## 3.3 Componentes

- Pipeline de Processamento de PDFs: O conteúdo dos PDFs será extraído utilizando pdfplumber e transformado em embeddings.
- Banco Vetorial (Qdrant): Os embeddings gerados serão armazenados em Qdrant para consultas vetoriais rápidas.
- Chatbot (Ollama): O chatbot processará as perguntas dos usuários e utilizará a plataforma Ollama para gerar respostas.
- Interface de Usuário (Streamlit): Uma interface em Streamlit permitirá aos usuários interagir com o chatbot e visualizar as respostas.

## 3.4 Bibliotecas Python Utilizadas

Conforme indicado no arquivo requirements.txt, as principais bibliotecas utilizadas são:

- LangChain para o gerenciamento da integração com os modelos de linguagem.
- Ollama para geração de respostas em linguagem natural.
- Qdrant para armazenamento e recuperação de embeddings.
- pdfplumber para extração de conteúdo dos PDFs.
- Streamlit para a criação da interface do usuário.

## 4. Fluxo de Dados

- O usuário carrega um ou mais documentos PDF no sistema.
- Os documentos são processados e transformados em embeddings.
- Os embeddings são armazenados no banco vetorial Qdrant.
- O usuário faz uma pergunta em linguagem natural via a interface Streamlit.
- O chatbot consulta o banco vetorial para recuperar as informações relevantes.
- O modelo Ollama gera uma resposta baseada nas informações recuperadas e a apresenta ao usuário.

## 5. Avaliação dos professores Considerações

Professor/a: