# Chatbot Unicamp 2024

Arthur de Andrade Almeida

November 7, 2023

## 1 Introduction

The RAG system was implemented in two stages - Indexing and Querying (See Figure 1).

- **Indexing**: In this stage the document is loaded from https://www.pg.unicamp.br/norma/31594/0. The document are then chunked into smaller units called nodes. Each node is then converted to a value (embedding) so that a distance-measuring computation can be done to find the most relevant chunk given a user query.
- **Querying**: In this stage, the user query is embedded, and a few chunks (context) are retrieved using the same embedding model. This query and the retrieved chunks are sent to LLM(s) to generate a response. The query and the context are either concatenated into a single string block (prompt).

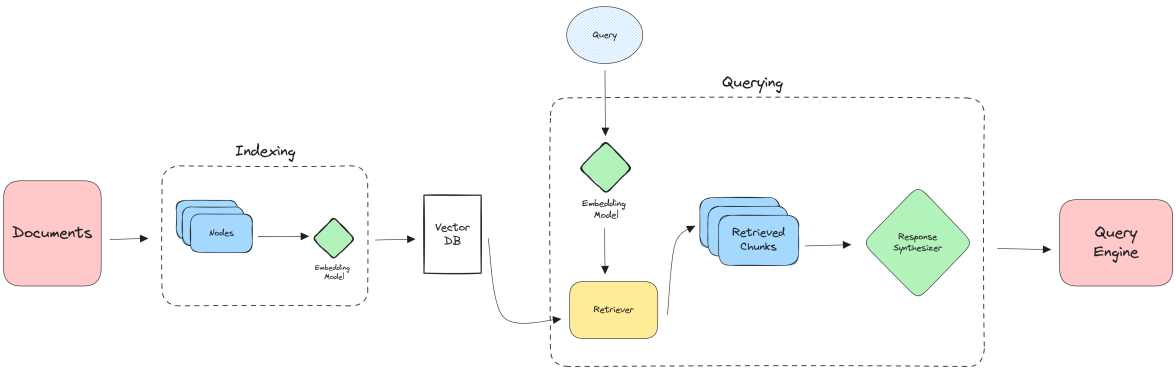


Figure 1: RAG Sytem Architecture

## 2 Project Structure

The project is organized as follows:

- **experiments**: This directory contains notebooks for conducting experiments.
- **data**: This directory contains the evaluation dataset.
- **app.py**: The main application module that serves as the entry point to the project.
- **querying.py**: It contains the implementation of the Querying stage.
- **indexing.py**: It contains the implementation of the Indexing stage.
- **utils.py**: A module that contains utility functions used by the application.
- **query_engine.py**: A module that defines a custom query engine for the project.
- **prompts.py**: A module that stores prompts used by the query engine.

# 3 Implementation Details

## 3.1 Indexing

It is implemented a class that handles the indexing process using the following steps:

1. **Load Existing Index**: If an index already exists in storage, it is loaded for further use. If the index is not found, the class follows the creation process.
2. **Node Parser Setup**: It configures a node parser that is capable of splitting text into smaller chunks.
3. **LLM and Embedding Initialization**: It initializes a LLM and an embedding model. In this case, the LLM is the *gpt-3.5-turbo* and the embedding model is OpenAIEmbedding(), both from the OpenAI API.
4. **Service Context Configuration**: The *ServiceContext* is a bundle of commonly used resources used during the indexing and querying stage in a LlamaIndex pipeline/application. A service context is created with the configured LLM, embedding model, and node parser.
5. **Document Retrieval**: It retrieves the document from a specified URL, in this case, `https://www.pg.unicamp.br/norma/31594/0`.
6. **Index Creation**: The VectorStoreIndex is created from the retrieved documents, utilizing the service context.
7. **Persist Index**: Once the index is created, it is persisted in storage for future use.

## 3.2 Querying

The querying module includes methods to build a retriever and a response syntheizer, using the existing index.

It is implemented the `build_retriever` method, which configures the retriever as the `VectorIndexRetriever` class and is set to retrieve the top 4 similar items based on the index and user queries.

Moreover, it is also created a method called `build_synthesizer` to configure and returne a response synthesizer, which is responsible to generates a response from the LLM, using a user query and a given set of retrieved text chunks, as shown in Figure 1. The synthesizer is created with a specified response mode, in this case, `compact`. The compact mode concatenates retrieved chunks that can fit within the context window.

# 4 Evaluation

Before deploying the RAG system as an application, a question-and-answer dataset was created for evaluation base on the document. The `gpt-4` model was used to create a more human-like dataset. Due to cost, only 24 question-answer pairs were generated.

Some examples of the generated pairs:

- **Question**: Quais são os critérios de desempate para a última colocação de algum curso, conforme mencionado no Art. 33 do documento?
- **Answer**: O critério de desempate para a última colocação de algum curso é a NP das provas das disciplinas prioritárias do curso, na ordem em que são apresentadas no Anexo III para cada curso.
  Se o empate persistir, prevalecerão as notas padronizadas das provas na ordem em que são apresentadas no § 3º do art. 23.
- **Question**: Quais são as habilidades gerais que a Unicamp avalia nos candidatos durante as provas de primeira e segunda fase do vestibular?

- **Answer**: As habilidades gerais que a Unicamp avalia nos candidatos durante as provas de primeira e segunda fase do vestibular incluem: identificar e reconhecer; inferir; analisar criticamente; comparar; formular hipóteses a partir de evidências; aplicar; resolver problemas propostos; relatar procedimentos utilizados; estabelecer e explicitar relações entre informações, conceitos, fenômenos, contextos, problemas e pontos de vista a partir de evidências logicamente construídas; explicar conceitos, fenômenos, contextos e problemas; indicar as evidências nas quais se baseou para produzir as respostas solicitadas; produzir textos curtos a partir de recursos como paráfrases e sínteses para responder o que indica o enunciado; argumentar em defesa de uma tese ou de um ponto de vista, de forma consistente com os insumos apresentados na prova e com os conhecimentos de cada área.

To measure the quality of the end-to-end pipeline, it was used two evaluation modules from the LlamaIndex framework:

1. **Faithfulness**: Measures if the response was hallucinated.
2. **Relevancy**: Measures if the query was actually answered by the predicted response.

The validation process occurs in batches to speed up the process. Finally, the values found for the two metrics were:

1. **Faithfulness**: 91.67% (22 out of 24 questions correctly)
2. **Relevancy**: 87.5% (21 out of 24 questions correctly)

# 5 Final Application

The final application was built using the Chainlit framework. To test the Chatbot, follow the instructions in the GitHub repository.