Retrieval Augmented Generation (RAG) and Vector Databases

Introduction

In this lesson we will cover the following:

- An introduction to RAG, what it is and why it is used in AI (artificial intelligence).
- Understanding what vector databases are and creating one for our application.
- A practical example on how to integrate RAG into an application.

Learning Goals

After completing this lesson, you will be able to:

- Explain the significance of RAG in data retrieval and processing.
- Setup RAG application and ground your data to an LLM
- Effective integration of RAG and Vector Databases in LLM Applications.

Our Scenario: enhancing our LLMs with our own data

For this lesson, we want to add our own notes into the education startup, which allows the chatbot to get more information on the different subjects. Using the notes that we have, learners will be able to study better and understand the different topics, making it easier to revise for their examinations. To create our scenario, we will use:

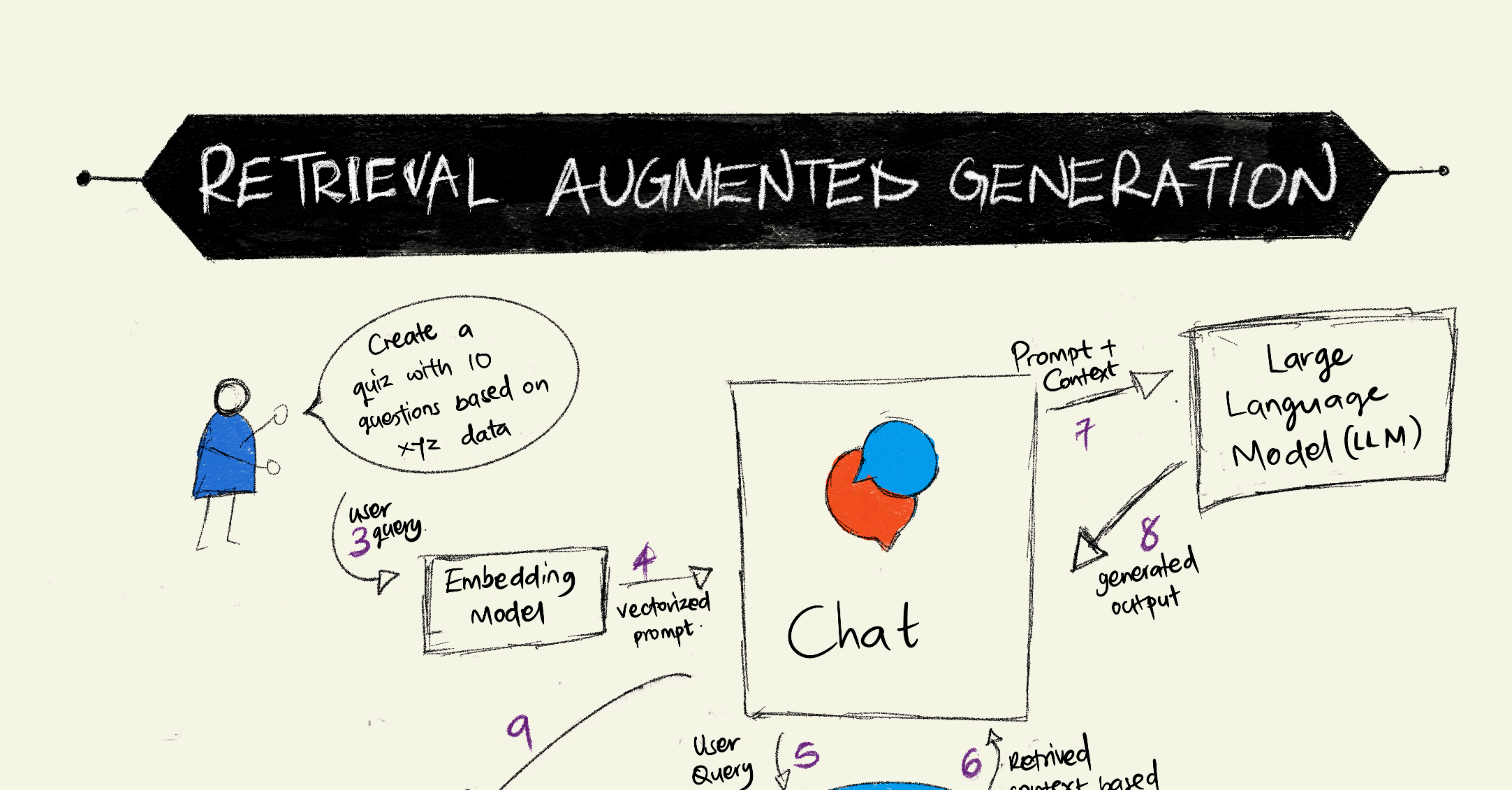
- `Azure OpenAI:` the LLM we will use to create our chatbot
- `AI for beginners' lesson on Neural Networks`: this will be the data we ground our LLM on
- `Azure AI Search` and `Azure Cosmos DB:` vector database to store our data and create a search index

Users will be able to create practice quizzes from their notes, revision flash cards and summarize it to concise overviews. To get started, let us look at what is RAG and how works:

Retrieval Augmented Generation (RAG)

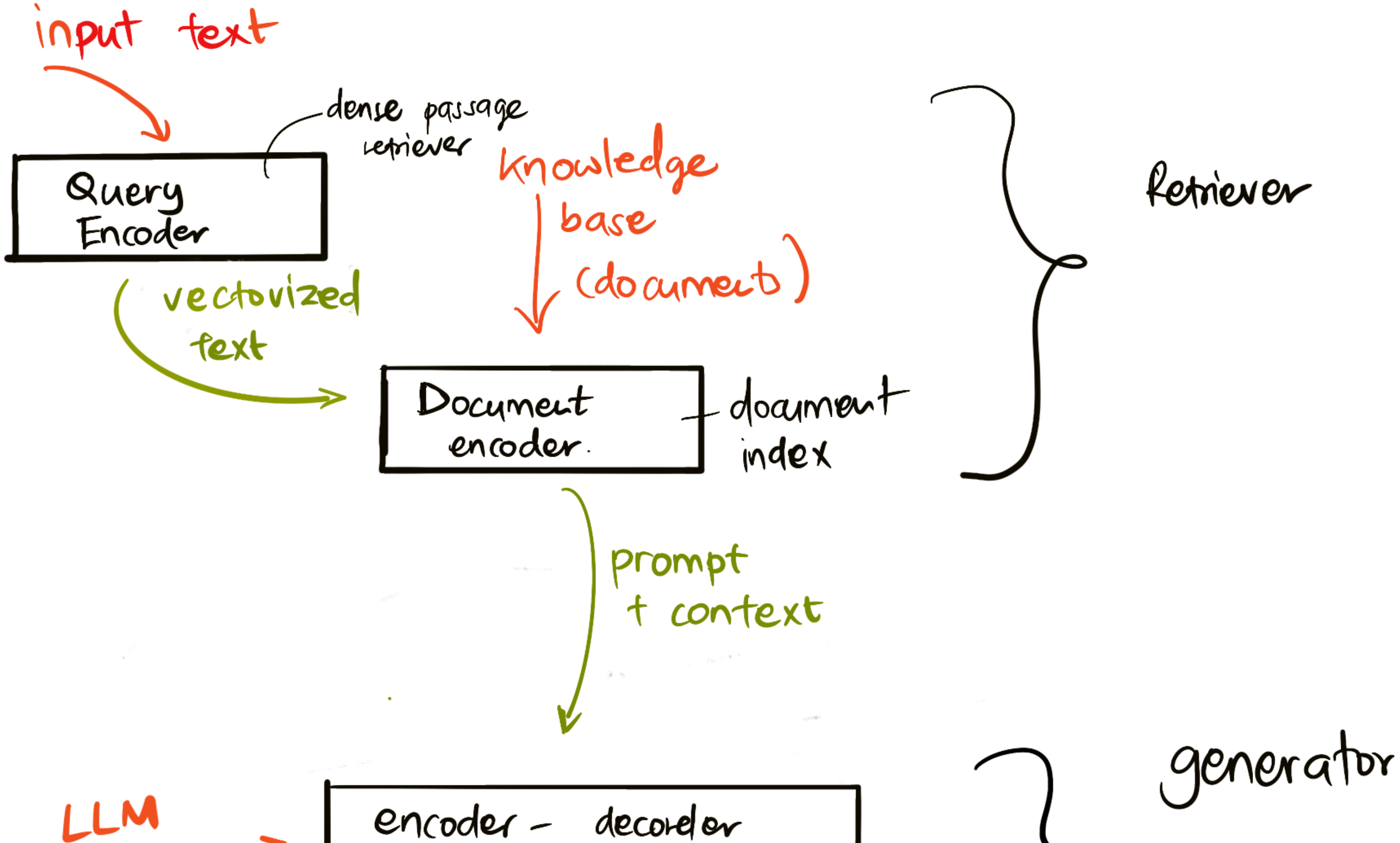
An LLM powered chatbot processes user prompts to generate responses. It is designed to be interactive and engages with users on a wide array of topics. However, its responses are limited to the context provided and its foundational training data. For instance, GPT-4 knowledge cutoff is September 2021, meaning, it lacks knowledge of events that have occurred after this period. In addition, the data used to train LLMs excludes confidential information such as personal notes or a company's product manual.

How RAGs (Retrieval Augmented Generation) work



RAGs operate as follows:

- **Knowledge base:** Before retrieval, these documents need to be ingested and preprocessed, typically breaking down large documents into smaller chunks, transforming them to text embedding and storing them in a database.
- **User Query:** the user asks a question
- **Retrieval:** When a user asks a question, the embedding model retrieves relevant information from our knowledge base to provide more context that will be incorporated into the prompt.
- **Augmented Generation:** the LLM enhances its response based on the data retrieved. It allows the response generated to be not only based on pre-trained data but also relevant information from the added context. The retrieved data is used to augment the LLM's responses. The LLM then returns an answer to the user's question.


input text
dense passage retriever
Query Encoder
vectorized text
knowledge base (documents)
Document encoder.
document index
Retriever
prompt + context
generator
LLM
encoder - decoder

Two approaches when implementing RAG according to the proposed paper: Retrieval-Augmented Generation for Knowledge intensive NLP (natural language processing software) Tasks are:

- ***RAG-Sequence*** using retrieved documents to predict the best possible answer to a user query
- **RAG-Token** using documents to generate the next token, then retrieve them to answer the user's query

Why would you use RAGs?

- **Information richness:** ensures text responses are up to date and current. It, therefore, enhances performance on domain specific tasks by accessing the internal knowledge base.
- Reduces fabrication by utilizing **verifiable data** in the knowledge base to provide context to the user queries.
- It is **cost effective** as they are more economical compared to fine-tuning an LLM

Use Cases for using RAG (Retervival Augmented Generation) and vector databases

There are many different use cases where function calls can improve your app like:

- Question and Answering: grounding your company data to a chat that can be used by employees to ask questions.
- Recommendation Systems: where you can create a system that matches the most similar values e.g. movies, restaurants and many more.
- Chatbot services: you can store chat history and personalize the conversation based on the user data.
- Image search based on vector embeddings, useful when doing image recognition and anomaly detection.