# RETRIEVAL AUGMENTED GENERATION BASED, LARGE LANGUAGE MODEL POWERED DOCUMENT QA CHATBOT – SAP AUSTRIA

Business Analytics MSc Capstone Project Summary

*Marcell Magda*

*June 2024*

## Table of Contents

Problem Introduction .......... 2
Client Requirements .......... 2
System Overview .......... 2
Evaluation Method .......... 3
Project Overview / Handover .......... 4
Learning Outcomes .......... 4

## Problem Description

As data volumes are ever increasing, many businesses manage a burgeoning catalogue of documents, with those responsible being bogged down by the information overload. These documents are also stored in a large number of file formats, such as PDFs, DOCX files, and scanned images. This was the exact issue faced by the client, SAP, and so it was decided that the best solution to this problem would be a centralized, AI-powered document management WebApp that could intelligently handle, store, and retrieve documents using natural language queries. This approach would aim to drastically reduce the time spent on manual document handling and improve organizational efficiency.

## Client Requirements

In more detail, as of now, the querying of a document, or collection by a user involves many clicks. These are, looking through internal systems to find the one that most likely contains the file, combing through sub-folders to locate the file and question, and only then can the process of reading through potentially hundreds of pages of content relevant to user begin. This process, while wildly inefficient, is yet the current standard. The document management WebApp aims to handle the first part by providing a central location the client to upload relevant documents, and the second, and more challenging issue, by using Large Language Models in combination with a Vector Store, based on the Retrieval Augmented Generation Framework. This way, an employee can locate and synthesize relevant information in seconds rather than hours.

## System Overview

The system created consists of three distinct parts: backend, a frontend, and a database. The backend is written in python and handles all the interaction with the database(s) and the LLM(s). Here there was an architectural choice made to create a system that is distilled to consist of four distinct "building blocks". These are: Chunking, Embedding, Retrieval and

Prompt. All these different blocks have sub-blocks behind them that are interchangeable, making the system highly robust. For example, at the chunking stage there is a choice of 3 different types (character, sentence, semantic), so it is up to the user to choose exactly what kind of chunking method to pair with what embedding model at document ingestion, and then to select a method of retrieval search and prompt template for querying this given ingested document. For the front-end, it is written in JavaScript using a framework called React, and a UI toolkit named “Material UI”. This front end allows the user to do two main interactions, there is one page for uploading documents, and there is one page for querying documents. The latter page is also where the uploaded documents are accessible, and when a given query is inputted to the system, an optional visualization, built using “Plotly” can be turned on. This displays a graphical representation of the document’s embeddings in a three-dimensional space along with the query and the pieces of text used to formulate the answer. The application was packaged up for deployment into and interconnected network of three containers, the frontend, the backend and the backup database (Chroma DB).

In a firm specific aspect, there was a focus on using SAP’s own capabilities to build the application. This involved two main tech platforms. Firstly, the SAP Business Technology Platform was used to provide both the Large Language Models and the Embedding Models. Here one can access OpenAI Models hosted in a secure Azure cloud environment. The second is the SAP HANA Cloud database, which was used for storing the documents and the embeddings through one of their premier offerings, the SAP HANA Vector datatype.

## Evaluation Method

The method used to evaluate this WebApp, namely the block-based architecture of the backend involved creating an evaluator framework. This was perhaps the hardest part of the project, as evaluation proved a very abstract concept in terms of a chatbot. There was an intense search for a possible solution to this problem, a way to quantify the effectiveness and efficiency of the system. The solution was a testing script, that can evaluate the system's efficiency in an automated manner, by cycling through all the possible block combinations, currently: $3 chunks \times 3 embeddings \times 2 search \times 3 prompt = 54 unique$. There are sample queries define together with "optimal" answers, and the system answers for these combinations are evaluated against an "optimal" answer using the BLEU (Bilingual Evaluation Understudy) scoring method. There is now a standardized method of testing out and evaluating new sub-blocks were they to be introduced, and a way to optimize the system on a document level, if the client wishes to do so.

## Project Overview / Handover

The project documentation, along with the codes for both the backend and the front end, as well as the packaged application were handed over to the client, SAP at the end of the project. The development can be considered successful as the created application fulfils the requirements set forth and performs to standard. Through careful testing and continuous consultation with the client, edge cases that would cause the system to go awry have been identified and corrected prior to the handover. Where there to every be any future issues, the source code is now fully available, and so theoretically any prospective client-side developer can engage with the codebase as necessary.

## Learning Outcomes

Throughout the project, I significantly enhanced my programming skills, honing my existing ones in Python and SQL, while mastering a new one, JavaScript. I addition to this, I also learned frameworks such as React, and gained in-depth understanding of SAP technologies. Simultaneously, I developed a profound understanding of how artificial intelligence and machine learning can be pragmatically applied to real-world business challenges, particularly in natural language processing and document management systems.