Project- Text Analytics

Knowledge Base Design and Implementation

**Team: TalkTech Tinkerers**

**Group Members:**

**Rodgers Okeyo Ochieng**

**Jihee Wang**

**Divyanshi Singh**

**Chandan Patel**



ISM 6564: Text Analytics

Dr: Tim Smith

Date: October 15th 2023

Contents

[1) Executive Summary 3](#_Toc148301493)

[ Objective: 3](#_Toc148301494)

[ Findings: 3](#_Toc148301495)

[ Recommendations 3](#_Toc148301496)

[2) Business Process Analysis 4](#_Toc148301497)

[a) Detailing the existing MS BAIS business process landscape 4](#_Toc148301498)

[b) Identifying Bottlenecks 4](#_Toc148301499)

[c) Exploring areas where a ChatGPT based Application like Q&A could add value 5](#_Toc148301500)

[3. Phase 1: Knowledge Database Design Document 5](#_Toc148301501)

[a) Data Source Identification 5](#_Toc148301502)

[b) Data Schema Development: 7](#_Toc148301503)

[c) Vectorization Techniques 7](#_Toc148301504)

[d) Documentation 9](#_Toc148301505)

# Executive Summary

* + Objective: Our project's objective is to create a ChatGPT-based interface tailored to enhance MS BAIS business operations at the University of South Florida (USF). The focus is on assisting prospective and new students, streamlining their initial engagement with USF, from choosing a program to the application and onboarding processes.

## Findings:

* + 1. **System in Use:** The primary tool for handling, tracking, and resolving student inquiries within the MS BAIS program at USF is the Jira system.
    2. **Assistance Approach:** USF's MS BAIS program assists students via its information request page on the graduate course website. While a comprehensive amount of data is available on the online portal, students often reach out directly for clarifications. In response, the USF IT department leverages the JIRA system to systematically address and manage these inquiries by raising tickets.
    3. **Admission Process:** The enrollment process at USF is methodical, involving an initial scrutiny of documents, data gathering, and a concluding decision based on the information and applicant's qualifications.
    4. **Student Interaction Tools:** Notably, the MS BAIS program website currently lacks any interactive interface or chatbot system, relying entirely on conventional methods to resolve student queries.
    5. **Primary Information Source::** The program’s FAQ page has been identified as a pivotal resource, capturing, and addressing the most recurrent student questions and concerns
  + Recommendations**:**
    1. **Streamlining Data Collection:** For bolstered digital support, it's essential to gather specific data concerning frequently asked MS BAIS program questions. These data points can guide automated responses once the user provides initial context. Historical data, including past JIRA tickets related to MS BAIS programs and previous email interactions, can be pivotal in creating a comprehensive corpus for any automated tool. Additionally, web scraping can be utilized to extract generic program information from the official website. The program's detailed FAQ section can also serve as an invaluable resource to train the Q&A application, ensuring it remains aligned with students' needs.
    2. **Q&A Application to be integrated with MS BAIS website:** Given the existing reliance on manual processes and the availability of the above-crafted corpus/data, introducing a Q&A application within an MS BAIS website can significantly optimize student interactions. Once operational, this application can address FAQs, guide prospective students through the application process, and reduce the workload on the administrative teams.

# Business Process Analysis

1. Detailing the existing MS BAIS business process landscape:
   * 1. **System Utilization**: At its core, USF employs the Jira system, a renowned tool for handling, tracking, and addressing student inquiries related to the MS BAIS program.
     2. **Assistance Dynamics**: While there is a robust presence of the MS BAIS program on the USF graduate course website, which provides extensive information, students often find the need to approach directly for specific clarifications. This is typically handled by the IT department of USF, which, in turn, makes use of the JIRA system to manage these queries.
     3. **Enrollment Paradigm**: The admission procedure is well-defined. It starts with a preliminary review of application documents, followed by a systematic gathering of necessary information, culminating in a final decision based on the comprehensive data and the applicant's credentials.
2. Identifying Bottlenecks**:**
   * 1. **Reliance on Manual Channels**: Despite the availability of comprehensive information online, students find themselves reaching out directly, which indicates potential clarity gaps on the portal.
     2. **Absence of Automated Support**: There's a noticeable lack of instant digital assistance tools, like chatbots, which can significantly delay response times and increase the manual workload.
     3. **Over-reliance on JIRA**: While JIRA is efficient, using it as a primary tool for all student queries may lead to potential backlogs, especially during peak admission seasons.
3. Exploring areas where a ChatGPT based Application like Q&A could add value**:**
   * 1. **Immediate Assistance**: A ChatGPT based Q&A application can provide real-time assistance to students, helping answer frequent queries instantly and reducing the time taken to provide responses.
     2. **Reducing Workload**: Automated responses to common queries can significantly decrease the manual workload on the administrative and IT teams.
     3. **Optimizing Admission Funnel**: By guiding prospective students through the application process interactively, the Q&A application can enhance user experience, potentially increasing successful applications.
     4. **Bridging Information Gaps**: A dynamic chat interface could adapt to user questions, filling the information voids that the current static web pages might not address explicitly.

# 3. Phase 1: Knowledge Database Design Document

## Data Source Identification

In this crucial phase, we meticulously identified and gathered data from diverse sources to create a robust knowledge base for the MS BAIS Program. Our approach was comprehensive, covering both web-based resources and valuable conversation records:

A group of different types of papers

Description automatically generated with medium confidence

*Source: Google Image search*

* **Data Extraction from MS BAIS Graduate Webpage and Catalog: Web Scraping and Manual Collection**

Our first data source involved extracting information directly from the official MS BAIS graduate webpage and catalog. This process was twofold.

Web Scraping: We utilized web scraping techniques to automatically collect data from web pages. This included details such as program information, admission procedures, course costs, contact details, and frequently asked questions.

Manual Data Extraction: In parallel with web scraping, we manually retrieved data from the website, ensuring that we captured all relevant documents, including admission processes, application guides, curriculum requirements, and more.

By combining web scraping and manual extraction, we created a comprehensive dataset comprising essential documents in PDF format:

doc1\_ overall admission process of MS BAIS.pdf

doc2\_how to apply for MS BAIS.pdf

doc3\_MS BAIS Admission.pdf

doc4\_cost of MS BAIS course.pdf

doc5\_contact freshman admissions.pdf

doc6\_MS BAIS application.pdf

doc7\_explore graduate programs.pdf

doc8\_what is I20.pdf

doc9\_most frequently asked questions.pdf

doc10\_additional information.pdf

doc11\_graduate assistantships.pdf

doc12\_MS BAIS Curriculum.pdf

doc13\_MS BAIS curriculum requirements.pdf

doc14\_MS BAIS prerequisites courses for incoming students.pdf

doc15\_global executive program (full time, 100 per online.pdf

doc16\_getting started with MS BAIS\_new student checklist.pdf

* **Conversation Records between Students and College Management via Jira Ticketing:**

Our second data source was the invaluable conversation records between students and the college management. These records were obtained through the Jira Ticketing system in JSON format. However, to make this data suitable for integration into our knowledge base, we undertook the following essential steps:

* Cleaning PII Data: Ensuring privacy and security, we meticulously employed regular expressions (regex) to methodically identify and expunge personally identifiable information (PII) and unnecessary special characters from the conversation datasets.
* Data Labeling: To enhance readability and maintain an organized structure, we performed key renaming. This involved changing keys like "Query 1" and "Response 1" to more intuitive labels, such as "Student Conversation 1," closely aligned with conversation numbers.

**File Format Conversion**: To ensure compatibility with our knowledge base, we converted the cleaned JSON records into the PDF file format, facilitating easier integration through our document loader.

The resulting dataset represented as “doc17\_Conversation Records between Students and College Management.pdf” became an invaluable resource for our knowledge base.

## Data Schema Development:

A crucial aspect of our knowledge base project involved defining a well-structured data schema. To achieve this, we employed Pinecone DB as our vector database and the Langchain Text Splitter known as RecursiveCharacter Text Splitter for data segmentation:

* Pinecone DB for Vector Database: We recognized the power and efficiency of Pinecone DB for managing our vector data. Pinecone DB operates as a 'key-value' store, where the 'key' corresponds to the embedding vector, and the 'value' represents the original text data.
* RecursiveCharacter Text Splitter: For effective data segmentation and structuring, we adopted the RecursiveCharacter Text Splitter. This tool enabled us to break down our data into manageable 'chunks,' ensuring optimal organization within the knowledge base.

The careful selection of these tools laid the groundwork for our data schema, setting the stage for efficient vectorization and retrieval processes.

## Vectorization Techniques

Embedding translates words into numerical values, enabling computers to interpret them with ease. In the context of data processing, "chunks" refer to smaller, more manageable portions of a larger dataset or information set. By segmenting large datasets into these chunks, we can simplify and organize the data more effectively. Meanwhile, vectorization turns elements, such as words or data points, into number sequences, effectively converting data into a format that's easily processed by computer systems.

The vectorization strategy we adopted was instrumental in the success of our project. After thorough research and deliberation, we settled on the OpenAI Ada document embedding API as our primary method. Nonetheless, we also recognize the significance of other vectorization techniques like TF-IDF, One-Hot Encoding, and Word2Vec, each bringing distinct advantages to the table:

• **TF-IDF (Term Frequency-Inverse Document Frequency)**: An established staple in NLP, TF-IDF gauges a word's relevance in a document compared to a larger set of documents. It accounts for both the term's frequency within a single document and its rarity across the entire collection. While it's adept at document retrieval, its capacity to capture deeper semantic relationships between words is sometimes limited.

• **One-Hot Encoding**: This approach encodes words as binary vectors. Each word is uniquely represented, marked by a '1' in its specific spot and '0s' elsewhere. While it shines with categorical data, it may falter in representing vast vocabularies and in grasping intricate semantic meanings.

• **Word2Vec**: This neural network-driven technique provides dense vector representations for words, capturing their semantic relationships and the context surrounding them. Its adaptability makes it potent for a range of NLP challenges.

A cartoon character and a question mark

Description automatically generated with medium confidence

*Source: Google Image Search*

We considered various vectorization techniques, including TF-IDF, One-Hot Encoding, and Word2Vec.

However, we selected the OpenAI Ada document embedding API for several reasons:

* Deep Learning Advantage: Ada employs advanced deep learning models, enhancing its understanding of complex language structures.
* Semantic Understanding: Ada's embeddings capture semantic relationships between words and phrases, unlike TF-IDF and One-Hot.
* Reduced Dimensionality: Ada efficiently reduces dimensionality, improving search and retrieval performance.
* Contextual Relevance: Ada excels in understanding context, critical for accurate responses in natural language understanding.
* End-to-end Solution: Ada provides a comprehensive solution for document embedding, eliminating the need for manual preprocessing.

In summary, while TF-IDF, One-Hot Encoding, and Word2Vec are valuable techniques, Ada's document embedding API offers superior performance for our Q&A application. Its deep learning capabilities, semantic understanding, dimensionality reduction, contextual relevance, and end-to-end convenience make it the ideal choice for structuring our knowledge base, ensuring context-aware and relevant responses to user queries.

This comprehensive knowledge database design document serves as the cornerstone for the subsequent phases of our project, demonstrating our commitment to creating an efficient and responsive Q&A application tailored to the MS BAIS Program at the University of South Florida (USF).

Below is the image pinecone vector database index = "langchain-demo-index" created for the project. This is our knowledge base.

A screenshot of a computer

Description automatically generated

## Documentation

In the rapidly evolving realm of data management and retrieval, the marriage of traditional databases with vector-based similarity checks offers a promising avenue for efficient and intuitive information extraction. This report delves into a two-fold process that not only ingests and processes a variety of document types like PDFs, TXTs, and URLs but also harnesses the power of vector embeddings and machine learning models to facilitate natural language querying. By leveraging these advanced techniques, the system epitomized in our study promises a seamless integration of data storage and contextually relevant data retrieval, catering to the demands of modern-day information processing.

A screenshot of a computer

Description automatically generated

The above diagram is a representation of a two-step process related to creating a database and querying a document using a vectorized approach. Below is an overview of steps 1 and step 2 which we are doing in this project. In this report, we are more focused on step 1.

**Step 1: Create Data Base (This is the main part of the project)**

* Split Type:
* Various types of files, such as PDFs, TXTs, and URLs, are ingested.
* Preprocessing or splitting phase to extract relevant data or sections from these sources.
* Embedding Type:
* The extracted data or sentences from the source files are then embedded or transformed into a numerical vector form.
* This might be using word embeddings or other Natural Language Processing techniques.
* Model & VectorStore:
* The created vectors are then stored in a "VectorStore". This is like a database but specifically designed to handle vectors. We are using the Pinecone database in this project.
* In the case of search the model calculates the "distance" between vectors. This would be useful for understanding the similarity between different pieces of data or to retrieve relevant information.
* As you can see the vector storage system is visually represented as a container holding various vectorized data.

**Step 2: Ask for the document (Here we use retrieval and LLM – OpenAI API Calls)**

* Question:
* For example, a user or system can pose a question: "What is …". Through retrieval we can get the relevant files or chunks for vector search based on the semantic search on this query.
* LLM (OpenAI- “gpt-3.5-turbo”):
* The question plus the relevant doc retrieved will be passed to an LLM.
* The LLM understands and processes the question in the context of the data passed to it and its own knowledge parameter.
* Relevance and Answers:
* The LLM identifies the most relevant 'splits' from the data to formulate the best possible answer to the question posed.
* It then provides an answer in natural language: Ex. "It is ...".

In conclusion, the convergence of traditional database methods with advanced vector embeddings and natural language processing heralds a new era in information management and retrieval. The system we've outlined not only holds the potential to streamline data integration but also ensures more contextually relevant data extraction based on natural language queries. As the demands of information processing continue to grow, it is imperative that our systems evolve in tandem, offering solutions that are both efficient and intuitive. This study has laid the groundwork for what promises to be an exciting journey ahead in the realm of data storage and retrieval.