# Maintenance Knowledge Advisor

**Project Guide:**
Meet Shah

**Candidate:**
Ashwin Waghmare

Internship Tenure:
2 May - 2 August
2024

# Acknowledgements

I would like to express my sincere gratitude to Tata Motors Ltd. for granting me the opportunity to intern with your esteemed organization. I am truly honored to have been selected for this internship and am excited to contribute my skills and learn from the talented professionals at your company.

I am thankful to my supervisor, Mr Abhishek Sharma for presenting me with a unique chance for me to gain practical insights into the industry, refine my skills, and grow both personally and professionally.

I would also like to extend my appreciation to my mentor, Mr Meet Shah who guided me throughout this internship. Your guidance and expertise undoubtedly played a crucial role in shaping my understanding of the industry and enhancing my skill set.

I am grateful to Mr Mahendra Waghmare for his unwavering support throughout. Your support has made the difference, and I am truly fortunate to have your helping hand.

Lastly, I would like to extend my thanks to the HR and IT departments of Tata Motors Ltd. for ensuring a seamless onboarding process.

# Contents

## 0.1 Abstract

Retrieval augmented generation (RAG) is a natural language processing (NLP) technique that combines the strengths of both retrieval- and generative-based artificial intelligence (AI) models. RAG AI can deliver accurate results that make the most of pre-existing knowledge but can also process and consolidate that knowledge to create unique, context-aware answers, instructions, or explanations in human-like language rather than just summarizing the retrieved data. RAG AI is different from generative AI in that it is a superset of generative AI. RAG combines the strengths of both generative AI and retrieval AI. RAG is also different from cognitive AI, which mimics the way the human brain works to get its results.

RAG, short for retrieval augmented generation, works by integrating retrieval-based techniques with generative-based AI models. Retrieval-based models excel at extracting information from pre-existing online sources like newspaper articles, databases, blogs, and other knowledge repositories such as Wikipedia or even internal databases. However, such models cannot produce original or unique responses. Alternatively, generative models can generate original responses that are appropriate within the context of what is being asked, but can find it difficult to maintain strict accuracy. To overcome these relative weaknesses in existing models, RAG was developed to combine their respective strengths and minimize their drawbacks. In a RAG-based AI system, a retrieval model is used to find relevant information from existing information sources while the generative model takes the retrieved information, synthesizes all the data, and shapes it into a coherent and contextually appropriate response.

By integrating retrieval and generative artificial intelligence (AI) models, RAG delivers responses that are more accurate, relevant, and original while also sounding like they came from humans. That's because RAG models can understand the context of queries and generate fresh and unique replies by combining the best of both models. By doing this, RAG models are:

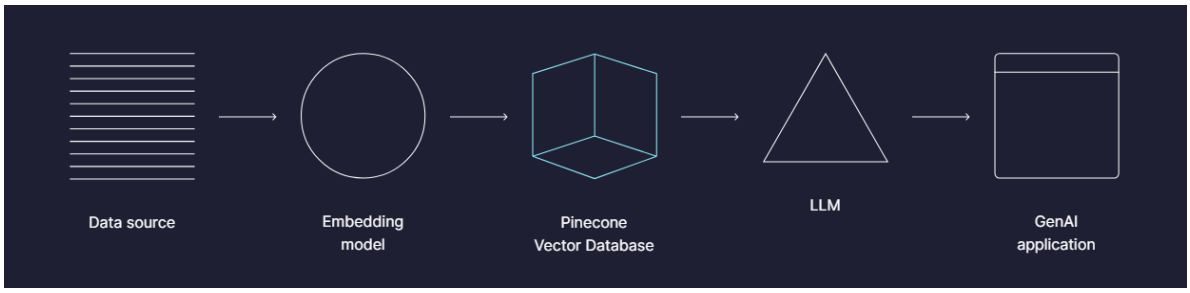
1. **More accurate** — By first using a retrieval model to identify relevant information from existing knowledge sources, the original human-like responses that are subsequently generated are based on more relevant and up-to-date information than a pure generative model.

2. Better at synthesizing information — By combining retrieval and generative models, RAG can synthesize information from numerous sources and generate fresh responses in a human-like way. This is particularly helpful for more complex queries that require integrating information from multiple sources.

## 0.2 Introduction

The Maintenance Knowledge Advisor API employs Retrieval Augmented Generation (RAG) to provide precise solutions for queries related to factory machinery. Developed with a focus on integrating with ChatGPT, it aims to directly assist factory workers by offering immediate resolutions to issues encountered with their machines.

The RAG (Retrieval-Augmented Generation) model integrates several components to enhance the capabilities of natural language processing systems. It begins with a robust data source providing the foundation for learning. An embedding model transforms text inputs into numerical vectors, enabling efficient storage and retrieval in a vector database. The Language Model (LLM) refines understanding and generates coherent responses. Ultimately, these components synergize in a search application, empowering advanced querying and information retrieval tasks with improved contextual relevance and accuracy.




## 0.3 Data source

The data source comprises of two components:

1. **PDFs**: These manuals are provided by the original equipment manufacturer (OEM) and contain comprehensive explanations of potential issues, faults, alarms, and detailed machine operations.

2. **PSR**: These monthly surveillance records are stored as Excel sheets. They document various issues encountered across all machines, including issue descriptions, incident and problem resolution codes (ICR and PCR), as well as additional details such as the involved personnel and dates of resolution.

Given the predominantly static nature of this data, our approach involves locally storing embeddings as vector indices. This method includes creating distinct vector indices for each machine, enabling more personalized and precise responses tailored to specific machine references.

## 0.4 Embedding model

Embedding models function as mathematical representations that capture the essence of words or phrases in a continuous vector space. This transformation enables machines to interpret language nuances and relationships more effectively. By mapping words to

vectors, these models can grasp semantic similarities and differences, enhancing the depth of understanding within search queries.

In RAG, search embeddings are specifically designed to efficiently retrieve the most relevant pieces of text from a wide range of data based on user queries. These embeddings are optimized to find the best possible match between user queries and available documents. Their primary application is in information retrieval systems, such as search engines and recommendation systems, where they are trained with techniques that focus on query-document relevance rather than solely on semantic closeness.

The details of the model used are as follows:

- Model name : "text-embedding-ada-002"
- Model version : "2024-02-15-preview"
- Model dimensions : 1536
- Max tokens : 8191

The text-embedding-ada-002 model represents a significant leap in OpenAI's embedding technology. It outperforms all the old embedding models on text search, code search, and sentence similarity tasks and achieves comparable performance on text classification. The new model's performance score of 53.3 surpasses the older models, ranging from 49.0 to 52.8.

One of the standout features of text-embedding-ada-002 is the unification of capabilities. It replaces five separate models that simplify the interface and perform better across diverse benchmarks. The context length has been increased by a factor of four, from 2048 to 8192, and the new embeddings have only 1536 dimensions, one-eighth the size of davinci-001 embeddings.

Moreover, the price of the new embedding models has been reduced by 90%, achieving better or similar performance at a 99.8% lower price. However, it's worth noting that the new model does not outperform text-similarity-davinci-001 on certain benchmarks, so for specific tasks, a comparison with this older model might be necessary.

## 0.5 Vector database

Vector storage serves as a fundamental component in handling high-dimensional data, particularly embeddings from deep learning models. This technology enables efficient storage and retrieval of complex data structures essential for various applications.

Vector databases play a pivotal role in storing and querying data points efficiently, allowing for quick identification of similarities within extensive datasets. They excel in supporting real-time applications and are crucial for tasks like recommendation engines.

In this project, the Simple Vector Store from LLama Index played a crucial role in managing embeddings effectively. This tool was chosen for its capability to store embeddings efficiently and its ability to handle operations swiftly, ensuring that our system could handle large volumes of data without compromising on performance. Importantly, the Simple Vector Store was selected not only for its technical capabilities but also for its cost-effectiveness, as it provided these functionalities at no cost. This strategic choice allowed us to leverage advanced vector indexing capabilities while staying within budgetary constraints, making it a practical and efficient solution for our project's needs.

## 0.6 LLM

In this project the OpenAi's GPT3.5 is used as the llm of the RAG model. Its details are as follows:

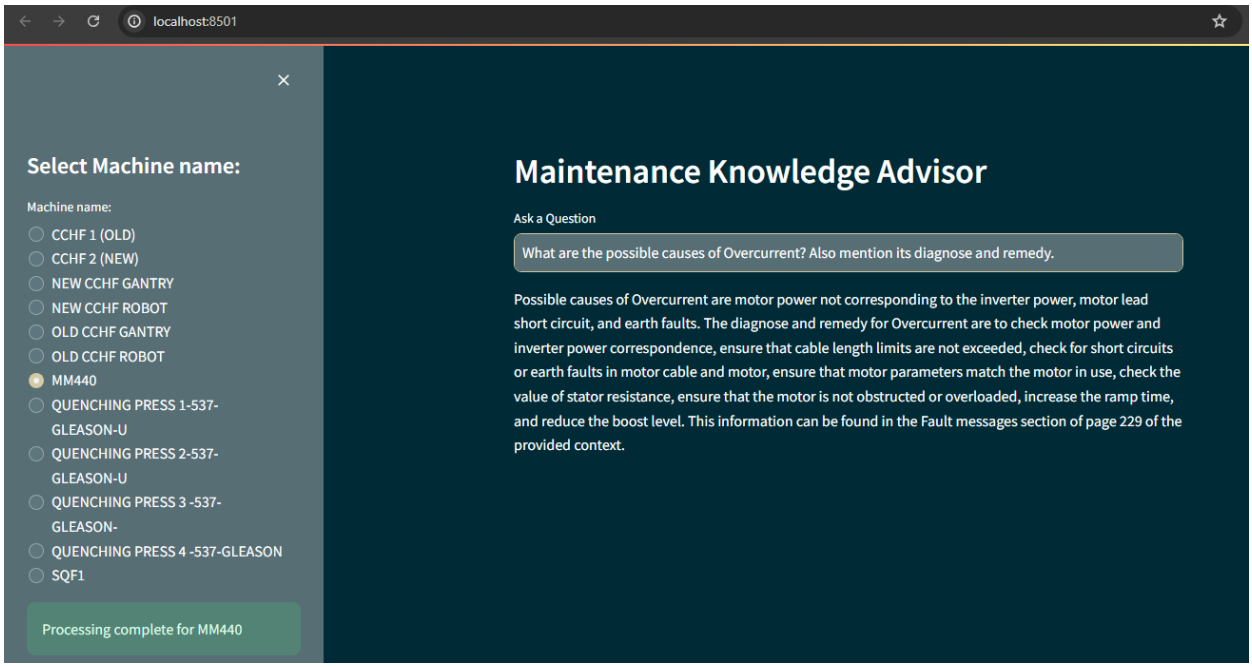
- Model name : "gpt-35-turbo-16k"
- Model version : "2023-07-01-preview"
- Model architecture : Transformer-based

The architecture of GPT-3.5-Turbo Instruct is based on the transformer architecture, which is a type of neural network architecture that was first introduced in a 2017 paper by Google researchers. The transformer architecture is composed of multiple layers, each of which includes a self-attention mechanism and a feed-forward neural network. The self-attention mechanism allows the model to weigh the importance of different parts of the input when making predictions, while the feed-forward neural network is used to make the final predictions.

Additionally, the chat includes a text_qa template designed for customized responses. It also provides prompts that outline the structure of CSV files, aiding the large language model (LLM) in better understanding the content for retrieval purposes.

## 0.7 Search application

The final search application is deployed on Streamlit. Streamlit is an open-source framework used for building web applications focused on data science and machine learning. It simplifies the process of creating and sharing interactive web apps for data analysis and machine learning projects.



## 0.8 Comparison of available models

### 0.8.1 Comparative study of vector databases

By weighing factors like speed, efficiency, and cost-effectiveness, users can make an informed decision based on their specific requirements. Whether prioritizing performance in similarity searches or seeking seamless integration with LLM applications , understanding these key differences is crucial in selecting the ideal vector storage solution.

When evaluating FAISS and Chroma for your vector storage needs, it's essential to consider their distinct characteristics. Chroma stands out as a versatile vector store and embeddings database tailored for AI applications, textbfasizing support for various data types. On the other hand, FAISS offers a cutting-edge GPU implementation focused on indexing methods, optimizing memory usage and retrieval speed for similarity searches.

1. FAISS stands out for its rapid retrieval of nearest neighbors, leveraging GPU acceleration for enhanced performance. It's a robust library optimized specifically for dense vector similarity search and clustering. FAISS prioritizes a balanced approach between memory usage and accuracy, incorporating advanced techniques such as quantization and partitioning. The inverted file index, a key feature, efficiently organizes large datasets by dividing them into partitions based on centroids. Each vector is assigned to the partition corresponding to its closest centroid, with encoded vectors stored within inverted lists. During searches, FAISS sequentially scans elements and returns the smallest distances encountered so far, making it highly efficient for retrieval tasks.

2. Chroma DB provides a self-hosted server option for storing vector embeddings and supports semantic searches on textual data. Renowned for its lightweight design and intuitive interface, Chroma DB optimizes the performance of Large Language Models (LLMs) by efficiently storing and querying vector embeddings. Similar to other vector databases, Chroma DB transforms queries into embeddings using an embedding model, conducts similarity searches, and retrieves pertinent data based on similarity metrics such as Cosine similarity or Euclidean distance.

3. Pinecone Vector DB is highly regarded as the leading vector database due to its exceptional speed and ability to swiftly create indexes. Pinecone employs Approximate Nearest-Neighbor (ANN) methods, utilizing metrics like cosine distance for efficient internal search and retrieval operations. It is also recognized for its enterprise-grade security measures and compliance with SOC 2 and HIPAA regulations, making it particularly suitable for secure storage of data and embeddings in sectors such as banking and finance.

4. As an AI application developer, grasping the varying trade-offs between accuracy and performance among these vector databases is crucial. Our evaluation, focusing on recall using a small subset of the testing dataset, reveals that PgVector exhibits the lowest recall among all databases studied. This suggests PgVector prioritizes faster search times over accuracy. Conversely, Qdrant (local mode) emerges with the highest recall among the evaluated vector databases.

### 0.8.2 Comparative study of embedding models

When evaluating the best embedding models for semantic search, it's essential to consider specific criteria that can impact performance. The key factors for comparison typically revolve around accuracy, speed, and versatility.

- **Accuracy**: This crucial metric assesses how precisely an embedding model captures the semantic relationships between words or phrases. Higher accuracy implies a better understanding of language nuances, leading to more relevant search results.
- **Speed**: The speed of an embedding model determines how quickly it can process text into vector representations. Faster models can enhance the user experience by enabling search systems to operate more swiftly, delivering quick and accurate search outcomes.

There are so many embedding models available in the market right now, but here are some hand-picked some of the leading models.

1. **Cohere Embed v3** is a cutting-edge embedding model designed for enhancing semantic search and generative AI. This model has shown very good results in various benchmarks like the Massive Text Embedding Benchmark (MTEB) and BEIR, proving it as an high performance embedding model across different tasks and domains. Some of it's key features are:
    - Compression-Aware Training: This approach optimizes efficiency without sacrificing quality and allows the model to handle billions of embeddings without significant infrastructure costs.
    - Multilingual Support: It supports over 100 languages, making it highly versatile for cross-language searches.
    - High Performance: Particularly effective in noisy real-world data scenarios, ranking high-quality documents by evaluating content quality and relevance
2. **OpenAi's embedding models**: OpenAI has recently introduced their new advanced embedding models, including text-embedding-3-small and text-embedding-3-large. These models offer better performance and are more cost-efficient.
    - Performance: The text-embedding-3-large model supports embeddings with up to 3072 dimensions. This allows for detailed and nuanced text representation. It has also outperformed previous models on benchmarks like MIRACL and MTEB.
    - Cost-Effectiveness: The previous models of OpenAI like text-embedding-ada-002 had some pricing issues because it was a bit on the expensive side. But the newer model text-embedding-3-small is almost five times more cost-effective compared to its predecessor, text-embedding-ada-002.
3. **Mistral**: The Mistral family includes some of the high-performing, open-source Large Language Models including an embedding model, E5-mistral-7b-instruct. This model is initialized from Mistral-7B-v0.1 and fine-tuned on a mixture of multilingual datasets. As a result, it has some multilingual capability.

- Instruction Following: Specifically designed to perform better in tasks that require understanding and following complex instructions, making it ideal for applications in education and interactive AI systems.
- Large-Scale Training: Pre-trained on extensive web-scale data, fine-tuned for a variety of NLP tasks to ensure robust and reliable performance.
- High Efficiency: Optimized for efficient processing, capable of handling large datasets and delivering high-quality embeddings across diverse use cases.

Selecting the best embedding model for semantic search optimization involves evaluating each model's strengths against specific task requirements and objectives. Each model offers unique capabilities that suit different use cases within semantic search applications.

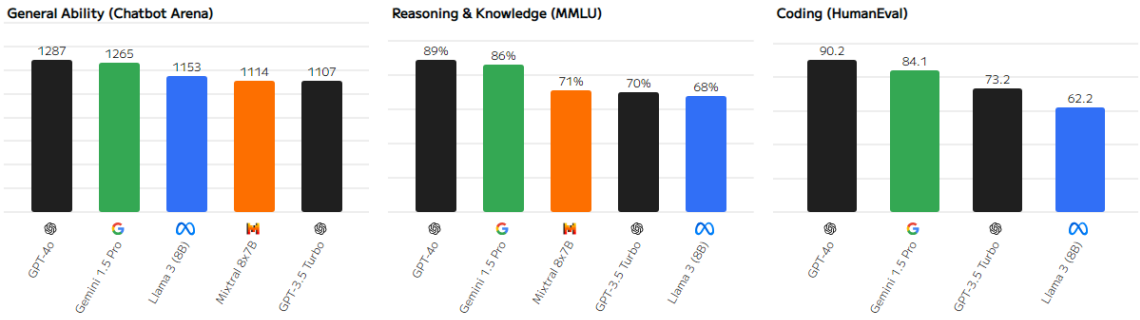
In terms of speed optimization, OpenAI's Embedding Models lead with their efficient embedding capabilities. Models like text-embedding-3-small provides quick processing speeds without compromising result quality. These models' high-dimensional embeddings and cost-effectiveness make them ideal for scenarios that require fast and affordable search outcomes.

### 0.8.3 Comparative study of llms

In artificial intelligence, three standout models are making waves: Meta's LLaMa 3, OpenAi's Gpt 4 and Mistral 7B. Key features of these models are compared below:

1. **OpenAi GPT 3.5 Turbo**
    - GPT-3.5 Turbo models can understand and generate natural language or code and have been optimized for chat using the Chat Completions API but work well for non-chat tasks as well.
    - The latest GPT-3.5 Turbo model with higher accuracy at responding in requested formats and a fix for a bug which caused a text encoding issue for non-English language function calls. Returns a maximum of 4,096 output tokens.
2. **Meta's LLaMa 3**
    - Extensive Training Data: LLaMa 3 was trained with over 15 trillion high-quality tokens from public sources, which is seven times more than its predecessor, LLaMa 2. This massive amount of data helps the model understand and generate better responses.
    - Rigorous Data Filtering: The data used to train LLaMa 3 went through extensive checks to remove inappropriate content and duplicates, ensuring the model learns from the best quality data.
3. **Mistral 7b**
    - Efficient and Powerful: Mistral 7B, with 7 billion parameters, is designed to be fast and effective, making it suitable for tasks that need quick responses.
    - High Performance: Despite its smaller size, Mistral 7B performs exceptionally well in tasks like math, coding, and logical thinking, even outdoing larger models.

- Accessible and Open Source: Mistral 7B is available under a license that allows anyone to use it freely and can be accessed on popular AI platforms.




### 0.8.4 Conclusion

The retrieval module is the most important component of the RAG chatbot. Good-quality document matches allow for meaningful responses from the instructions tuned LLM. Building a robust and accurate retrieval engine consists of designing a prompt, choosing and deploying the embedding module together with the vector store, and finally adding the re-ranking module. Each step should be evaluated, and the best-performing setup should be chosen.

## 0.9 Future works

1. Certain portions of our data repository comprise scanned copies of user manuals. Due to their scanned nature, these documents require optical character recognition (OCR) technology to extract textual content effectively. Once extracted, this text undergoes further processing to generate embeddings, enabling us to incorporate it into our analytical and retrieval systems. This approach ensures that even non-digital sources of information can be seamlessly integrated into our data-driven workflows, enhancing our ability to derive insights and support decision-making processes effectively.

2. Implement a feature to store chat history, enabling the LLM to directly reference and respond to previous queries. This capability enhances the system's ability to provide personalized and contextualized answers based on past interactions.

## 0.10 References


- Artificial Intelligence a Modern Approach by Stuart Russell and Peter
- https://www.geeksforgeeks.org/supervised-unsupervised-learning/
- https://myscale.com/blog/faiss-vs-chroma-vector-storage-battle/
- https://community.hpe.com/t5/insight-remote-support/comparing-pinecone-chroma-db-and-faiss-ex
  td-p/7210879
- https://myscale.com/blog/best-embedding-models-semantic-search-comparison/
- https://softwaremill.com/embedding-models-comparison/
- https://artificialanalysis.ai/models
- https://www.infoq.com/news/2024/01/mistral-ai-mixtral/
- https://medium.com/@kagglepro.llc/llama-3-vs-mistral-7b-a-head-to-head-ai-showdown-14c35cad0