LLM Maintenance Chatbot

Sankhya Ganesh, Ptolemy Henson, Ashish Kashyap, Yassir Khalaf,
Neelay Ranjan, Anuj Shah, Shreyas Sreenath, Fetume Teklu

PURDUE UNIVERSITY | The Data Mine

Background

Our project built an LLM-based chatbot to **reliably answer maintenance questions** related to the project dataset and a **user-focused application** for demonstration.

Our analogous dataset of Pan-AM manuals are a comprehensive set of scanned documents spanning approximately 13,000 images.

Integration

Objective: Integrating Application with LLM enhances the User experience and functions as a fundamental aspect of the project.

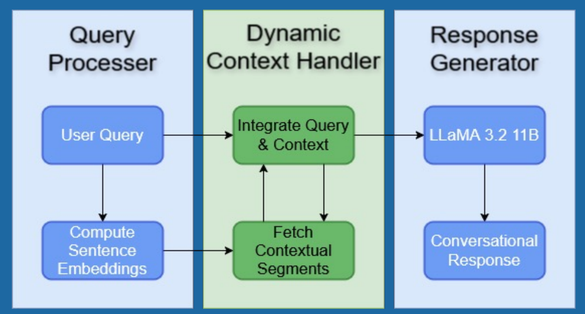
Methodology: For portability and ease modifications to the codebase, the entire project is containerized in a Docker. Running a containerized Docker allows for the project to run, regardless of operating system or underlying hardware.

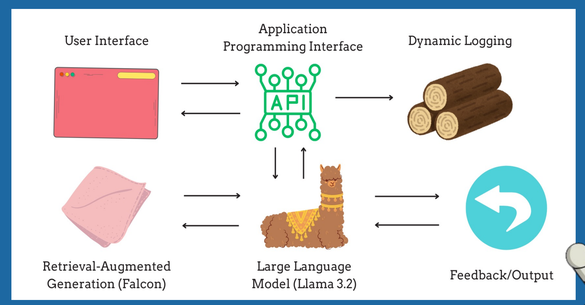
Toolbox

Incorporated Tools:

- Google Cloud
- Mongo DB
- React
- NLTK
- Transformers
- Anvil
- PyTorch
- *Llama 3.2* - 3B param, Text only model
- *Llama 3.2* - 11B param, Vision enabled model

Methodology





Feature	RAG	Fine-Tuning
Adaptability to Dynamic Information	Adapts well with access to latest information	May require updates to stay relevant
Customization and Linguistic Style	Limited customization based on retrieved data	High degree of personalization possible
Data Efficiency and Requirements	Leverages external datasets, less labeled data needed	Requires substantial, task-specific training data
Efficiency and Scalability	Cost-effective and scalable with external data	Higher initial resource investment required
Domain-Specific Performance	Broad topical coverage, versatile	Deep, precise domain expertise

Application Features

Objective: Provide user-focused application to interface with LLM that emphasizes security, flexibility, and user-feedback capabilities.

Main Features:

- Multi-Format Communication
- Modifiable Conversation History
- Temporary Chats
- User Feedback Functionality

Methodology: Frontend and backend implementation was done using ReactJS and Express.JS. The conversation history utilizes a MongoDB database and Google Cloud Storage for saved media files.

RAG System

Project uses **Retrieval-Augmented Generation (RAG)** system, which segments documents by sentence and uses embedding to match questions with correct sources.

- **Combats hallucinations.**
- **Reduces** need for **model retraining** on new data.

Barnett, Scott, et al. "Fine-tuning or fine-failing? debunking performance myths in large language models." (2024).

Future Goals

Our vision remains centered on enhancing **performance**, **transparency**, and **accessibility** to build a more seamless and reliable system.

- Implement **automated fact-checking** to ensure response accuracy and credibility.
- Integrate **multi-language support** for broader accessibility and inclusivity.
- Optimize real-time processing to **reduce** latency in high-demand scenarios.
- Return excerpts from the manual to give user option to **learn** more **from the source**

Conclusion

The LLM Maintenance Chatbot integrates a large language model into a user-focused application to **accurately respond to maintenance queries**.

Our team improved efficiency and scalability by implementing a **RAG system**, an **API-driven approach**, and a **unified frontend-backend structure**.

We would like to thank **Reva Pettigrew (TA), Doug D (Corporate Mentor), Jennifer, Michelle,** and **Dustin** for their constant support. We would also like to thank the **Data Mine (Program) and V2X** for making this possible.