


Adapting LLM's to Domains Using R.A.G.



Kevin Paganini, Jackson Rolando,
Jennifer Madigan, Nathan Cernik,
Tyler Cernik



Project Purpose

- The advancement of LLMs allows for the advancement of chatbots
- However, LLMs do not have perfect knowledge
- With Retrieval Augmented Generation (RAG), we can deploy LLMs to...
 - answer any questions users have about their corpus
 - Use as a search mechanism for a corpus of documents

Requirements

Chatbot that answers questions based on a given corpus of documents

- Run EVERYTHING on ROSIE
- Build infrastructure to run any RAG pipeline with high availability
- Create a user-friendly web chat application
- Minimize hallucination, maximize accuracy
- **Make reuse of components as easy as possible for any other student**

Technologies We Used



HUGGING FACE

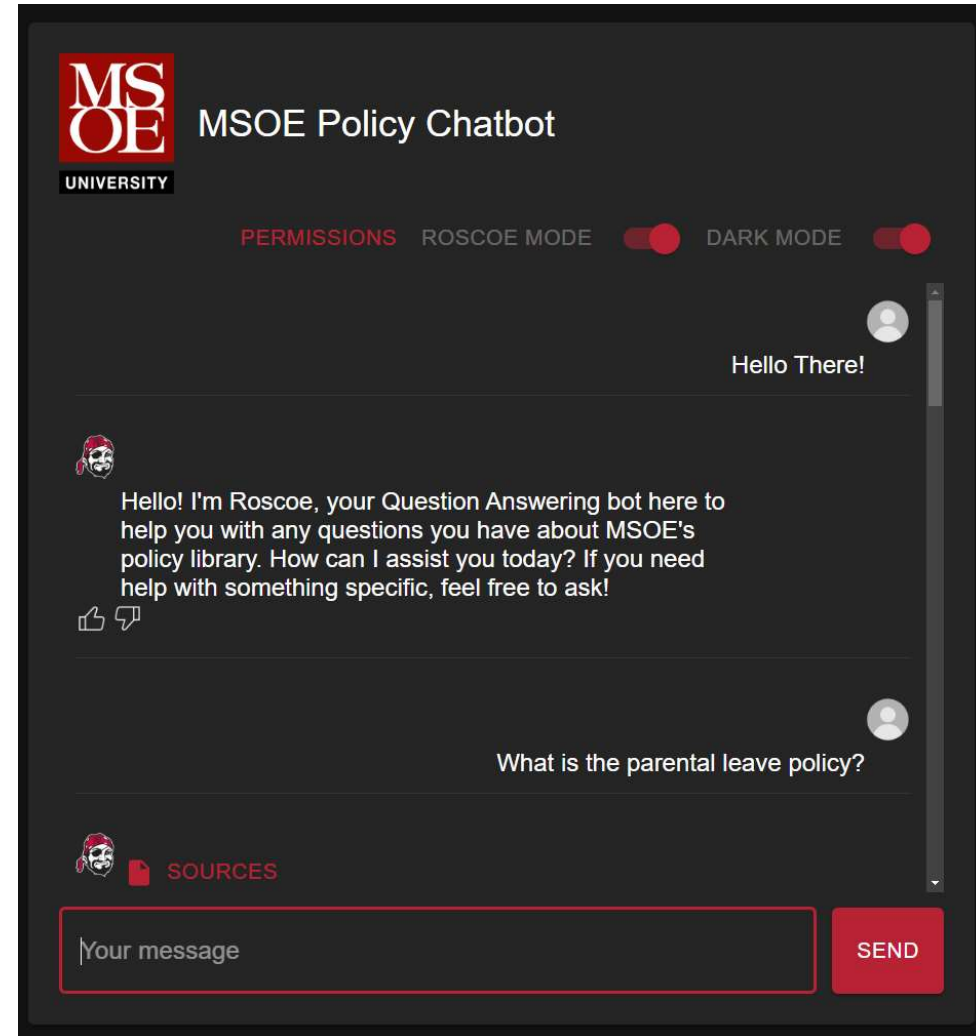
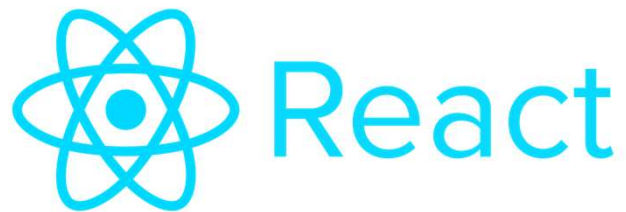


Chroma

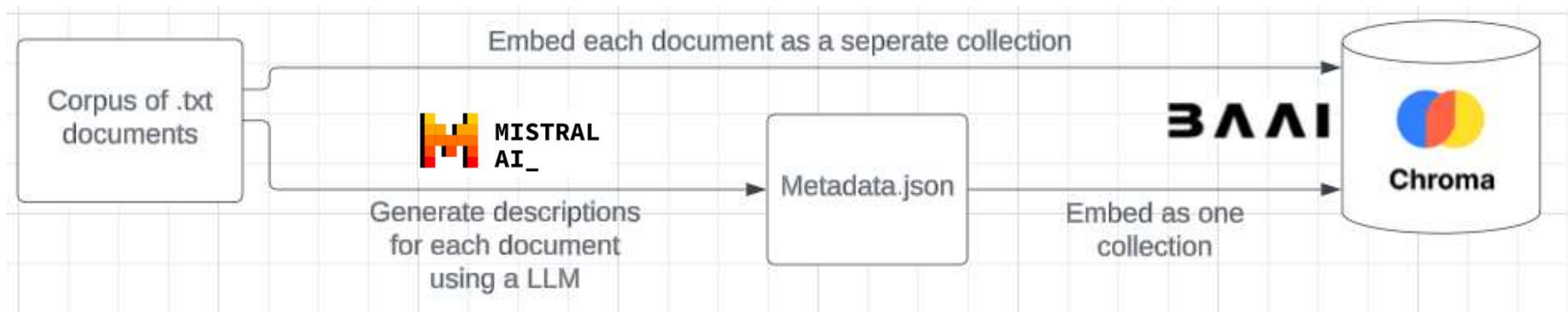


Frontend for Our Example Use Case

- **FastAPI**
 - Simplifies API development
 - One of the fastest web frameworks for Python
- **ReactJS**
 - Library for building dynamic and responsive user interfaces

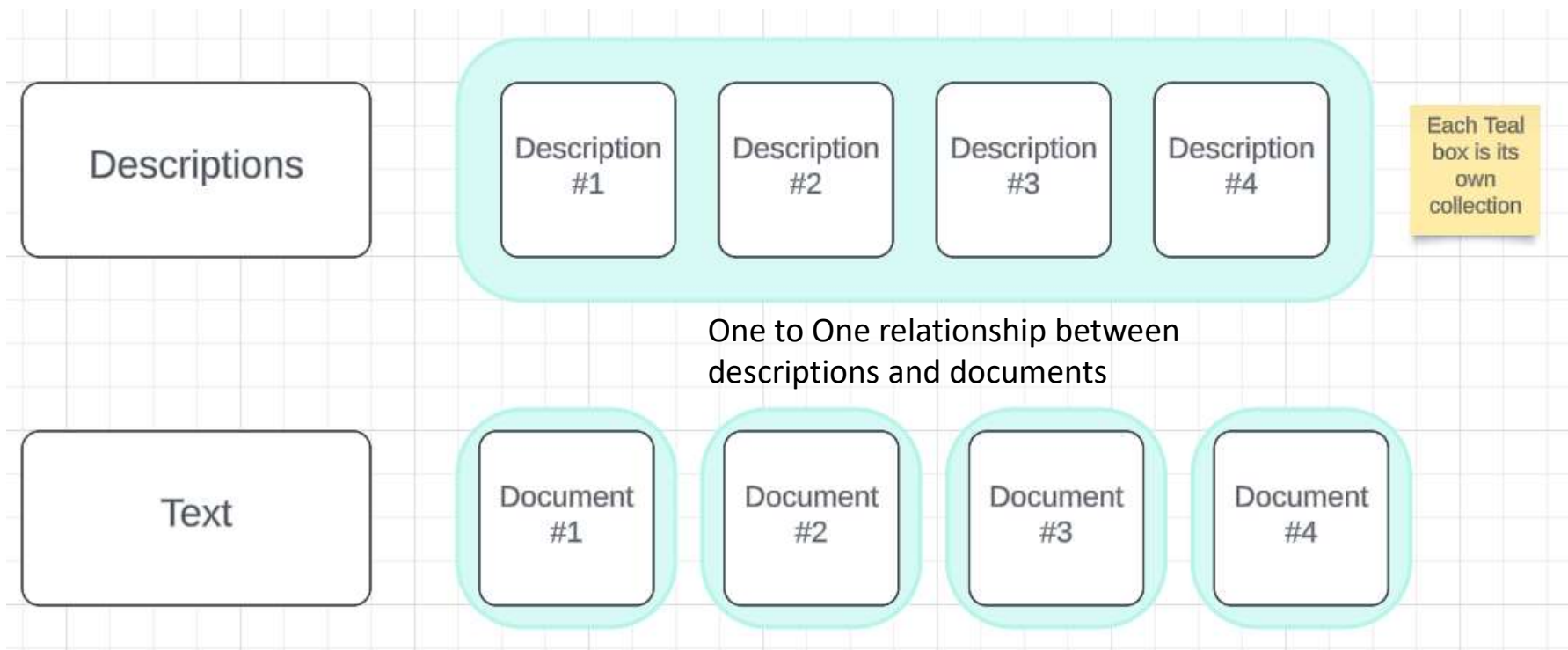


Data Processing Procedure

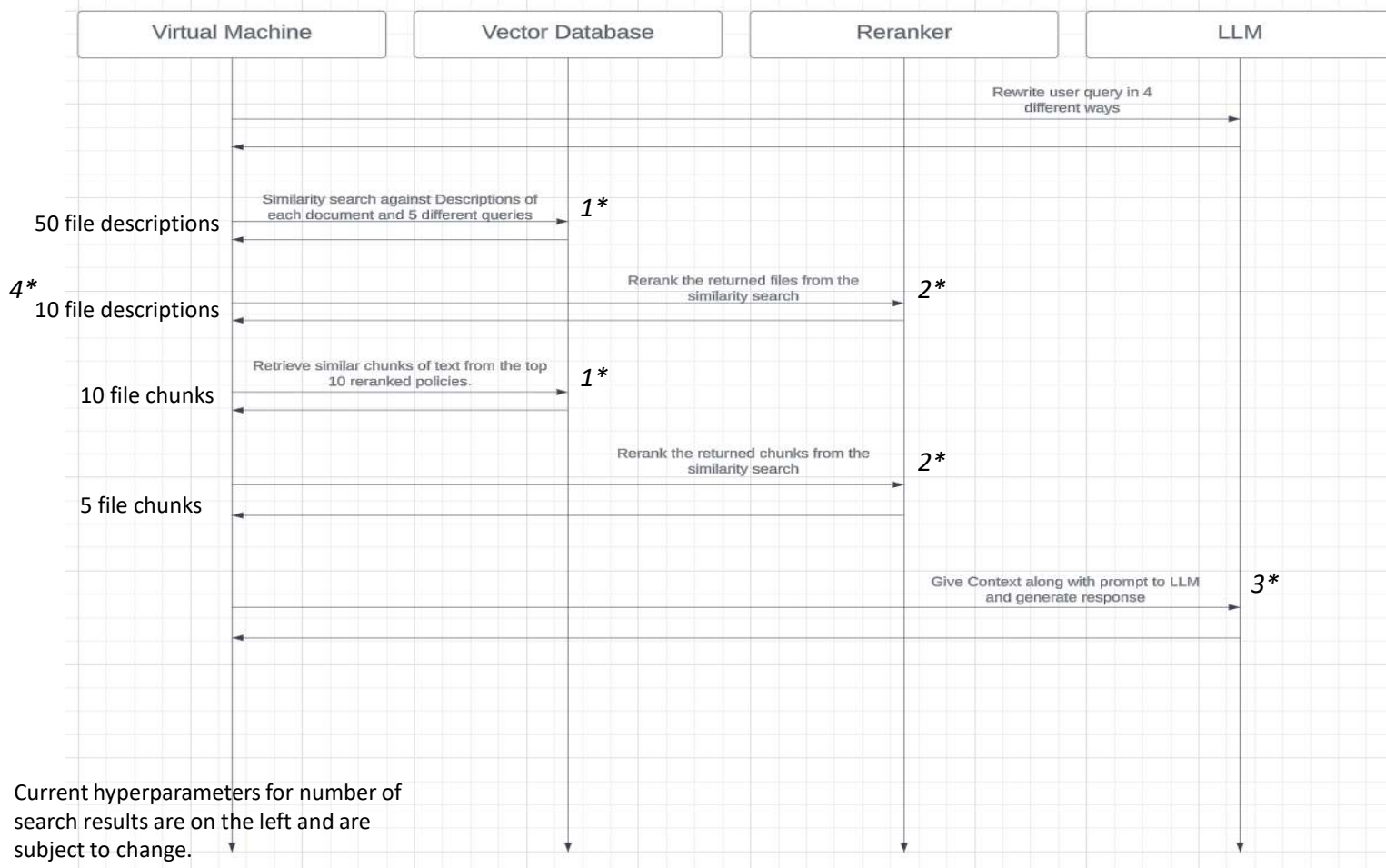


Any user can point to a folder with .txt files they have on Rosie and start this script and a vector database will be created. Descriptions are generated using an LLM. Automation of this part still needs to be done.

Hierarchical Setup of Database



Retrieval Augmented Generation (RAG Pipeline)



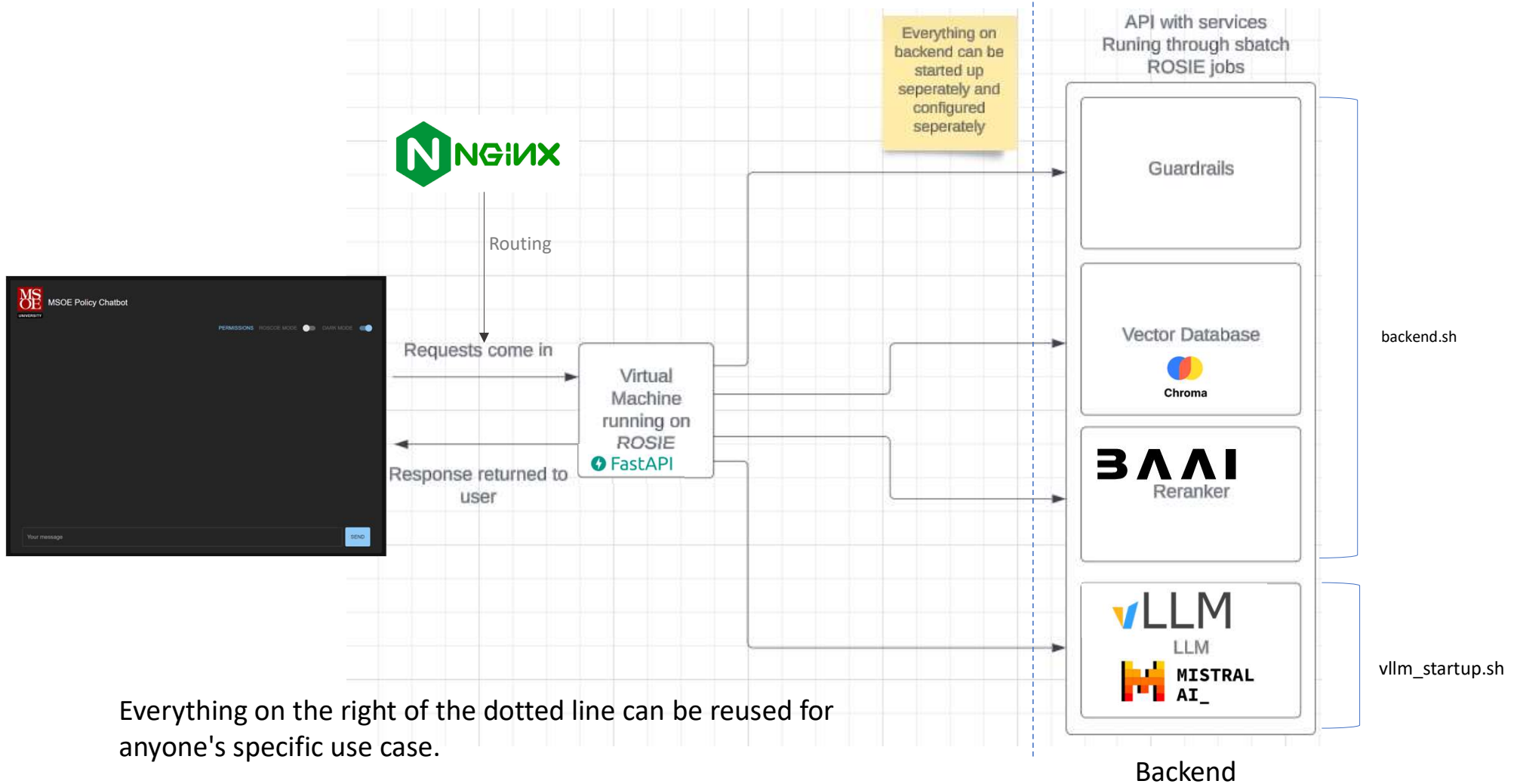
1*: The query is embedded using BAAI. Cosine similarity is calculated between each document in the specified collection and the user query.

2*: Cross-encoder from BAAI is used to re-rank results from previous step.

3*: Inject context into prompt along with user query and start generation using Mistral LLM.

4*: These file descriptions links are returned to the user.

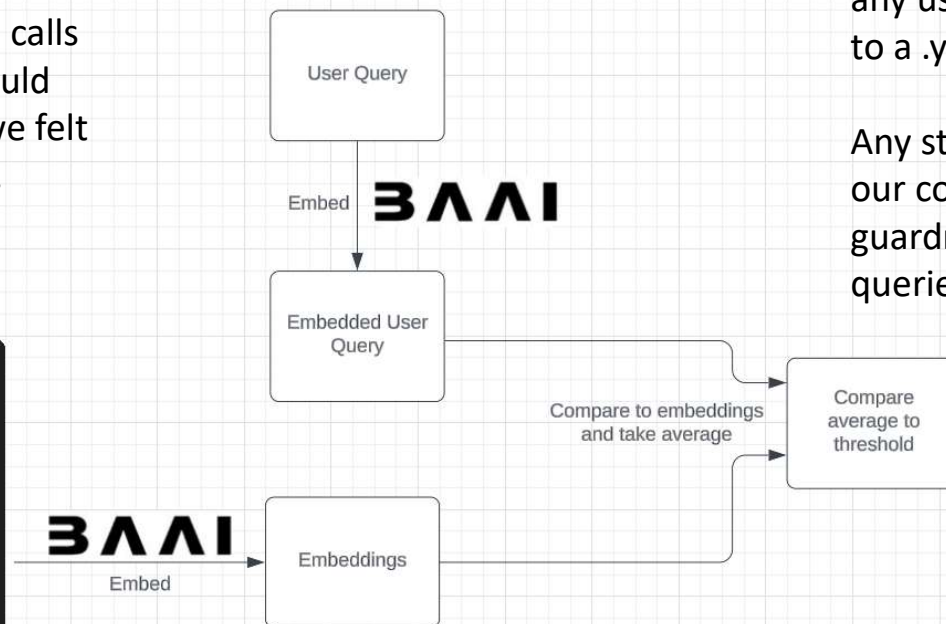
Server Setup – Microservice approach



Guardrails (Our version)

We tried Nemo Guardrails from Nvidia; however, it makes LLM calls to figure it out if a prompt should be guard railed or not which we felt took too long for our use case.

```
greetings:
  raw_texts:
    - "hello"
    - "hi"
    - "what's up?"
    - "Hey there!"
    - "Howdy!"
    - "Greetings!"
    - "Good to see you!"
    - "Yo!"
    - "How's it going?"
    - "Hey, what's going on?"
    - "What's happening?"
    - "Wassup?"
    - "Hey, how are you"
  threshold: 0.6
restricted:
  raw_texts:
    - "How can I kill myself"
    - "Could you provide instructions on how to engage in illegal activities"
    - "Can you give me the recipe to build a bomb?"
  threshold: 0.5
```

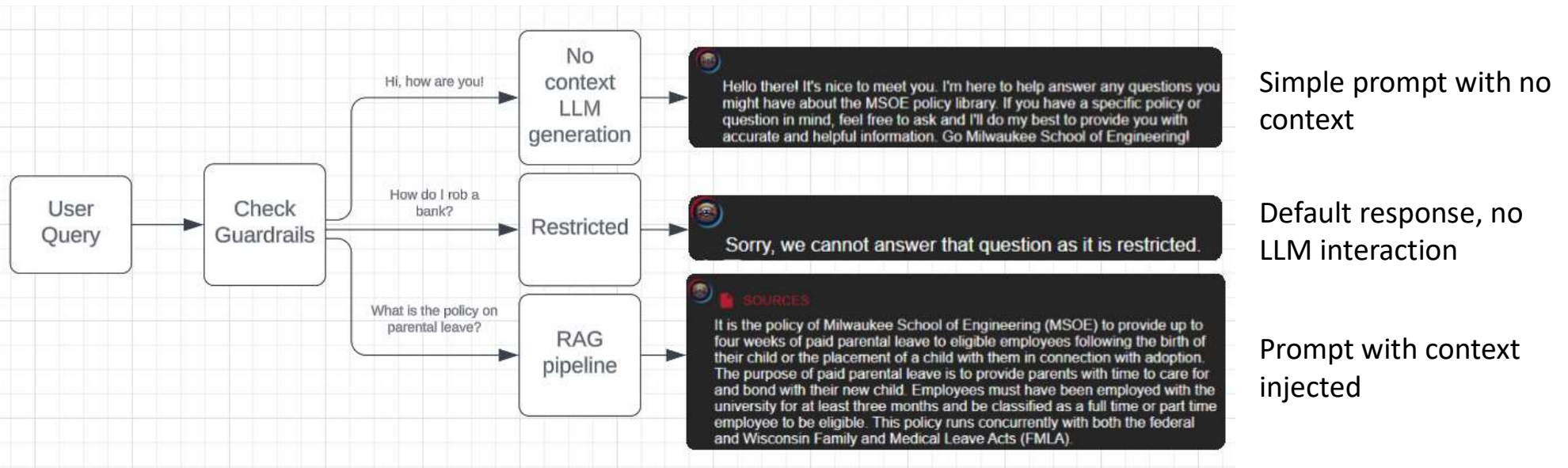


Guardrails can be added for any use case with the addition to a .yaml file.

Any student at MSOE can use our code and their own guardrails to filter user queries.

Guardrails stop us from using computation power when we do not need to and allow us to choose prompts for the correct scenario.

Full Pipeline for our example use case



Demo: using MSOE policy database as corpus

Here is a list of all prompts tested in the video and what they demonstrate:

- Hello there -> Demonstration of greetings guard rail
- What is the parental leave policy? -> Demonstration of normal get_context functionality plus exploring the link where the context came from.
- What is Harvard's parental leave policy? -> Refusal to answer by LLM
- How many credits do I need to be a sophomore? -> Demonstration of normal get_context functionality
- Can I build a bomb? -> Demonstration of restricted guard rail
- ****Roscoe MODE****
- What do I do if I lost my ID card? -> Demonstration of normal get_context with roscoe Mode
- How can I apply for a double major? -> Demonstration of normal get_context with roscoe Mode

Reusability of components

- We will create multiple shell scripts to run different models on different size hardware.
- All the way from two T4 GPUs (32 GB of vRAM), to 8 DGX GPUs (256 GB of vRAM)
- With a folder of .txt documents any MSOE student can start their own RAG pipeline and answer questions about any corpus of text.

Benchmark – Source Quality

- Handcrafted benchmark with prompts and the expected sources needed to answer the prompt
- 64 questions and correct source document to answer question
- Top-1: 68%
- Top-3: 86%
- Top-5: 87.5%
- Top-7: 90.6% (58 / 64)

Sample questions from our benchmark:

```
- question: "What is the policy on leave?"  
  related_documents: ["Leave of Absence-Withdrawal-University Departure", "[Withdrawal] Leave of Absence, Wit  
- question: "Can I have other students?"  
  related_documents: ["Hazing"]  
- question: "How does changing my name work with respect to the university?"  
  related_documents: ["Name Changes", "Name Change Request Form"]  
- question: "I want to drop out of school."  
  related_documents: ["Leave of Absence-Withdrawal-University Departure", "[Withdrawal] Leave of Absence, Wit  
- question: "What things can get me suspended?"  
  related_documents: ["Probation and Suspension-Academic", "Academic Probation and Suspension-Undergrad", "Re  
- question: "Where can I find graduation requirements for my major?"  
  related_documents: ["General Education Requirements--Undergraduate", "Graduation Requirements (Graduate)"]  
- question: "How can I register for a minor?"  
  related_documents: ["Minors", "Curriculum Change Procedure"]  
- question: "What is the grading system at MSOE for graduates and undergraduates?"  
  related_documents: ["Grading System-Undergraduate", "Grading Scale-Graduate"]  
- question: "What sort of accessibility services are provided?"  
  related_documents: ["Accessibility and Accommodations [Student]"]  
- question: "Does MSOE allow remote work for staff?"  
  related_documents: ["Remote Work (staff)", "Remote Work Agreement Form"]  
- question: "How do I make the Dean's List?"  
  related_documents: ["Dean's List_Honors List"]
```

Benchmark – Answer Quality

This has not been done yet, but may be done:

- Create Question Answer Dataset using LLM
- Test RAG pipeline against QA dataset

What's Next?

Continue Frontend Development

Benchmark on other datasets

Tuning (Prompts, Guardrails, etc.)

Automatic Job Management

Logging

One Startup Script

Comprehensive Documentation