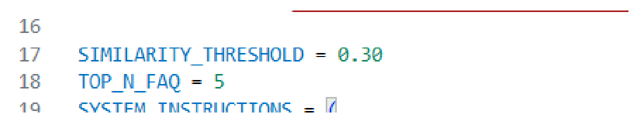
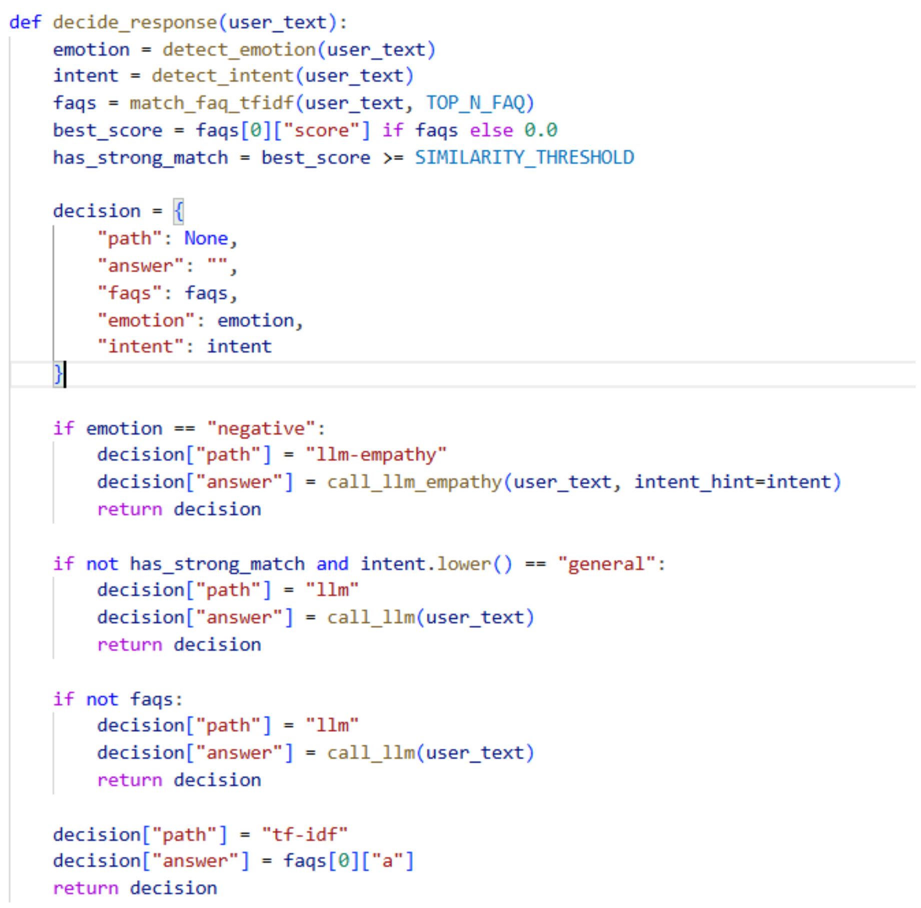
Mandeep Singh Bhar – 8989367

The presentation Question Response of rag code and Llama trigger - Team 5



```

SIMILARITY_THRESHOLD = 0.30
TOP_N_FAQ = 5
SYSTEM_INSTRUCTIONS = (
```

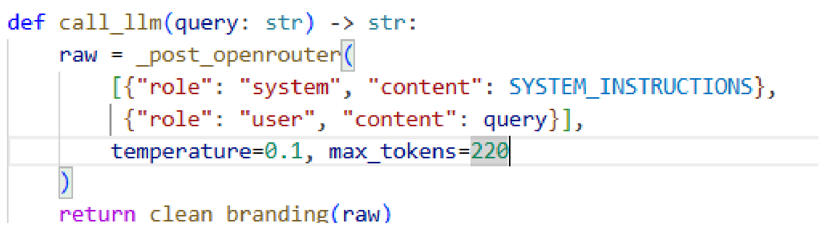
This is the numeric cutoff we compare the top similarity score against. Scores **≥ 0.30** are treated as strong; scores **< 0.30** are treated as medium/low and will route to the LLM for general queries.



```
def decide_response(user_text):
    emotion = detect_emotion(user_text)
    intent = detect_intent(user_text)
    faqs = match_faq_tfidf(user_text, TOP_N_FAQ)
    best_score = faqs[0]["score"] if faqs else 0.0
    has_strong_match = best_score >= SIMILARITY_THRESHOLD

    decision = {
        "path": None,
        "answer": "",
        "faqs": faqs,
        "emotion": emotion,
        "intent": intent
    }

    if emotion == "negative":
        decision["path"] = "llm-empathy"
        decision["answer"] = call_llm_empathy(user_text, intent_hint=intent)
        return decision

    if not has_strong_match and intent.lower() == "general":
        decision["path"] = "llm"
        decision["answer"] = call_llm(user_text)
        return decision

    if not faqs:
        decision["path"] = "llm"
        decision["answer"] = call_llm(user_text)
        return decision

    decision["path"] = "tf-idf"
    decision["answer"] = faqs[0]["a"]
    return decision
```

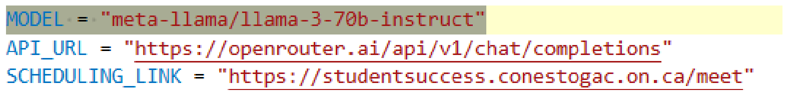
We compute a **similarity score** for the user's query against our Faq.

If the top score is **below 0.30** (our "medium/low" band) **and** the intent is *general*, we **trigger the LLM** via call_llm(user_text). If the score is high, we return the answer directly from our knowledge store.



```
def call_llm(query: str) -> str:
    raw = _post_openrouter(
        [{"role": "system", "content": SYSTEM_INSTRUCTIONS},
         {"role": "user", "content": query}],
        temperature=0.1, max_tokens=220
    )
    return clean_branding(raw)
```

This is the generation step. When the similarity is in the medium/low band, we call our LLM with the user's query to produce the answer.



```
MODEL = "meta-llama/llama-3-70b-instruct"
API_URL = "https://openrouter.ai/api/v1/chat/completions"
SCHEDULING_LINK = "https://studentsuccess.conestogac.on.ca/meet"
```

We use **meta-llama/llama-3-70b-instruct** through the OpenRouter API. This is what actually runs when the decision gate routes to the LLM on medium/low similarity.

Concluion

1. We calculate a similarity score for the query in the retrieval step of our RAG pipeline.
2. The score is compared against a fixed cutoff: SIMILARITY_THRESHOLD = 0.30.
3. If the score is **below 0.30** and the intent is "general," we the **generation** step of RAG by sending the query to the LLM.
4. The LLM used is **meta-llama/llama-3-70b-instruct**, invoked via call_llm(user_text) → _post_openrouter(...).
5. If the score is **0.30 or higher**, the answer comes directly from the retrieved content, skipping LLM generation.