When in Doubt, Ask First: A Unified Retrieval Agent-Based System for Ambiguous and Unanswerable Question Answering

Long S. T. Nguyen[1,2], Quynh T. N. Vo[1,2], Hung C. Luu[1,2], Tho T. Quan[1,2*]

[1]URA Research Group, Ho Chi Minh City University of Technology (HCMUT), Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
[*]**Correspondence:** qttho@hcmut.edu.vn

Abstract

Large Language Models (LLMs) have shown strong capabilities in Question Answering (QA), but their effectiveness in high-stakes, closed-domain settings is often constrained by hallucinations and limited handling of vague or underspecified queries. These challenges are especially pronounced in Vietnamese, a low-resource language with complex syntax and strong contextual dependence, where user questions are often short, informal, and ambiguous. We introduce the Unified Retrieval Agent-Based System (URASys), a QA framework that combines agent-based reasoning with dual retrieval under the Just Enough principle to address standard, ambiguous, and unanswerable questions in a unified manner. URASys performs lightweight query decomposition and integrates document retrieval with a question–answer layer via a two-phase indexing pipeline, engaging in interactive clarification when intent is uncertain and explicitly signaling unanswerable cases to avoid hallucination. We evaluate URASys on Vietnamese and English QA benchmarks spanning single-hop, multi-hop, and real-world academic advising tasks, and release new dual-language ambiguous subsets for benchmarking interactive clarification. Results show that URASys outperforms strong retrieval-based baselines in factual accuracy, improves unanswerable handling, and achieves statistically significant gains in human evaluations for clarity and trustworthiness. All code and datasets are publicly available at `https://github.com/ura-hcmut/URASys`.

1 Introduction

Large Language Models (LLMs) have become a cornerstone of modern *Question Answering* (QA) systems (Rasool et al., 2024), demonstrating strong fluency across diverse tasks. As their capabilities expand, QA systems powered by LLMs are increasingly deployed in high-stakes, closed-domain environments such as academic advising, where answers must remain precise and context-aware (Raiaan et al., 2024). Questions on tuition fees, course prerequisites, or institutional policies often carry significant consequences: errors can delay graduation, incur financial penalties, and mislead enrollment decisions, making factual grounding and contextual sensitivity essential (Nguyen and Quan, 2025). This challenge is amplified in low-resource, nuance-rich languages such as Vietnamese, where syntactic variation and strong context dependence complicate interpretation and motivate creating dedicated datasets and benchmarks (Pham et al., 2024). Queries are frequently short, informal, and underspecified, reflecting natural advising conversations and exposing limitations of conventional QA pipelines without interactive clarification.

Although LLMs provide strong generative capabilities, they are inherently probabilistic and prone to *hallucination* when facing vague or incomplete questions (Li et al., 2024). *Retrieval-Augmented Generation* (RAG) (Lewis et al., 2020) mitigates this by grounding responses in external evidence, yet most RAG pipelines assume queries are fully specified and answerable. They often retrieve loosely related passages and attempt to answer despite insufficient evidence, rarely engaging in targeted clarification (Fan et al., 2024). Such behavior is especially problematic in advising contexts where users expect not only answers but reliable guidance. Recent advances such as IRCoT (Trivedi et al., 2023), MiniRAG (Fan et al., 2025), LightRAG (Guo et al., 2024), NodeRAG (Xu et al., 2025), and HippoRAG (Liu et al., 2025) improve evidence synthesis and multi-hop retrieval through interleaved reasoning and graph-based routing, but they still treat queries as static inputs and lack mechanisms for decomposition-driven understanding, interactive clarification, and explicit *unanswerable handling*. These gaps highlight the need for a reasoning-driven QA framework that can jointly decide when to answer, when to clarify, and when

Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 452–472
December 20-24, 2025

Table 1: Illustrative examples showing *User* (U) and *URASys* (S) handling different query types in educational QA.

Scenario	Example
Normal: direct, well-specified question	**U:** What are the prerequisites for Machine Learning 101? **S:** Introduction to Programming and Basic Statistics.
Ambiguous: questions with multiple interpretations, insufficient context, or overly broad scope (Wang et al., 2023)	**U:** How much is the fee for this program? **S:** The program has a tuition fee, a lab fee, and a registration fee. Could you clarify which one? **U:** Tuition. **S:** Tuition fee is 2 000 USD per semester.
Unanswerable: specific query with no matching information in the database	**U:** What is the course instructor's office phone number? **S:** Sorry, our advising database stores only email addresses and no phone numbers.

to explicitly signal no-answer without over-relying on retrieval confidence.

To address these gaps, we introduce the ***Unified Retrieval Agent-Based System*** (URASys), a QA framework for closed-domain settings with underspecified or critical queries. Unlike prior RAG pipelines, URASys leverages agent-based reasoning and a dual retrieval architecture under a *Just Enough* paradigm. It prioritizes understanding before answering, engages in clarification when user intent is ambiguous, and explicitly signals unanswerable when evidence is insufficient rather than hallucinating. A central *Manager Agent* performs query decomposition to infer intent and split complex questions into sub-queries for better evidence aggregation. It coordinates two specialized retrieval agents: (i) a *Document Retrieval Agent* over a hybrid *chunk-and-title* corpus index for lexical and semantic search, and (ii) a *FAQ Retrieval Agent* querying an automatically generated *Frequently Asked Questions* (FAQs) repository created via an *ask-and-augment* procedure. This two-phase indexing pipeline transforms raw documents into both evidence chunks and a standardized question–answer layer, enabling URASys to combine fast FAQ-style lookup with grounded document reasoning in a unified architecture.

URASys jointly addresses three critical QA scenarios: (1) resolving ambiguous queries via interactive clarification, (2) handling unanswerable cases through cross-source evidence synthesis and reasoning-driven decisions, and (3) answering standard queries with grounded single-hop or multi-hop reasoning, as illustrated in Table 1. The clarification loop mirrors human advisory behavior, while the dual retrieval pipeline reflects the natural workflow of consulting FAQs and policy documents. Our contributions are summarized as follows.

- We introduce URASys, an agent-based QA framework that integrates query decomposition, dual retrieval, and interactive clarification under a Just Enough principle. This paradigm prioritizes understanding over generation, enabling unified handling of standard, ambiguous, and unanswerable queries while improving accuracy and robustness in closed-domain QA. We further propose a two-phase indexing pipeline for dual retrieval, effective in low-resource languages without requiring complex graph infrastructure.
- We comprehensively evaluate URASys on Vietnamese and English QA benchmarks covering single-hop and multi-hop closed-domain tasks and a real-world academic advising dataset, including unanswerable subsets. URASys outperforms both traditional and advanced RAG baselines in factual accuracy.
- We release new ambiguous subsets targeting interactive clarification, enabling systematic evaluation of underspecified queries in both English and Vietnamese and establishing a benchmark for this underexplored setting.
- We conduct real-world human evaluations with end users interacting with the deployed system, demonstrating practical effectiveness in live advising workflows and statistically significant gains in user satisfaction. These results highlight the broader applicability of URASys to other high-stakes, closed-domain QA scenarios beyond academic advising.

2 Related Works

2.1 RAG and Recent Advances

RAG has emerged as a promising approach for improving factual accuracy in QA systems by grounding LLM outputs in retrieved evidence (Lewis et al., 2020). Standard pipelines often combine dense retrievers, token-based methods such as BM25, or hybrid approaches with generative models to produce context-aware responses (Fan et al., 2024), but typically assume well-formed inputs and underperform when queries are vague or underspecified. Recent studies propose advanced retrieval-reasoning architectures: IRCoT (Trivedi et al., 2023) alternates retrieval and reasoning in multi-step QA; LightRAG (Guo et al., 2024) and MiniRAG (Fan et al., 2025) enhance indexing with graph structures and semantic-aware topologies; NodeRAG (Xu et al., 2025) integrates heterogeneous graphs for structured evidence; HippoRAG (Gutierrez et al., 2024) employs hierarchical memory to capture long-range dependencies. While these models excel on benchmarks, they rely on high-quality inputs and complex infrastructure, posing challenges in resource-constrained, high-stakes domains such as educational QA. Critically, they lack mechanisms for interactive clarification and explicit unanswerable handling, motivating architectures combining modular retrieval with clarification-first interaction for ambiguous and underspecified queries.

2.2 Interactive Clarification, Unanswerable Handling, and Multi-Agent QA

Most QA systems treat user queries as fully specified and directly answerable. Interactive clarification challenges this assumption by posing follow-up questions to resolve vagueness (Guo et al., 2021), showing promise in task-oriented settings but remaining underexplored in academic advising, where precision is critical (Deng et al., 2024). Surveys on Asking Clarification Questions datasets highlight the lack of standardized resources for training systems to handle ambiguity (Rahmani et al., 2023). In parallel, multi-agent QA decomposes tasks into retrieval, reasoning, and planning roles (Viswanathan et al., 2022; Elizabeth et al., 2025; Deng et al., 2025), but few integrate lightweight retrieval with clarification into deployable pipelines for informal, context-dependent queries. Work on unanswerable QA has focused mainly on extractive benchmarks like SQuAD 2.0 (Rajpurkar et al., 2018), with limited evaluation in LLM-based RAG and rare integration of cross-source synthesis with explicit no-answer signaling. These gaps motivate URASys, which combines clarification, agent-based reasoning, and dual-agent retrieval to decide when a query is answerable, when it requires interactive clarification, or when it should be explicitly marked as unanswerable. They further underscore the need for Ambiguous QA datasets in both English and low-resource, nuance-rich languages such as Vietnamese, with a particular emphasis on educational QA contexts.

3 URASys Architecture

3.1 System Overview

Figure 1 illustrates the overall architecture of URASys. When a user submits a query q, the *Manager Agent* first analyzes its structure and semantics, then decomposes it into a set of sub-queries $\{q_1, q_2, \ldots, q_n\}$. Each sub-query q_i is assigned to a dedicated agent team $\mathcal{A}_i$, which comprises two specialized components: a *FAQ Search Agent* and a *Document Search Agent*. These teams operate concurrently, retrieving relevant evidence from both an augmented FAQ repository $\mathcal{F}$ and a structured document corpus $\mathcal{D}$, yielding evidence sets $E_i = \mathcal{A}_i(q_i, \mathcal{F}, \mathcal{D})$. Once retrieval is complete, the Manager Agent aggregates the results into a unified evidence pool $E = \bigcup_{i=1}^{n} E_i$, which serves as the basis for generating the final answer. If the evidence is insufficient or contradictory, the system proactively engages the user in a clarification round before finalizing the response, and in rare cases where both retrieval streams provide no supporting signals, URASys gracefully returns an explicit no-answer response. This architecture enables URASys to handle ambiguous, incomplete, or context-dependent queries with high precision and modularity, particularly in educational domains.

3.2 Modular Agent Design

URASys follows a modular multi-agent architecture inspired by how human advisors handle complex or underspecified queries. In real-world educational settings, effective advising typically involves two key behaviors: (i) seeking clarification when a user's intent is unclear, and (ii) consulting multiple sources to ensure accurate and comprehensive responses. These practices motivate the separation of responsibilities in URASys, enabling each agent to specialize while maintaining coherent coordination through a central controller.

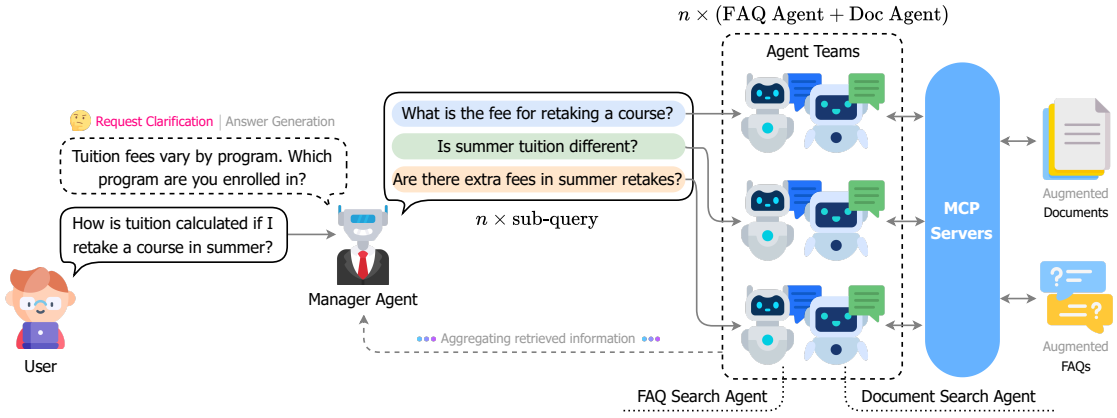


Figure 1: Overview of the URASys framework. Each user query is decomposed into n sub-queries, handled by parallel agent teams consisting of FAQ and document agents. Retrieved evidence is aggregated and used to generate a final answer, with optional clarification or explicit no-answer signaling if needed.

Manager Agent The *Manager Agent* is the system's central reasoning component. It orchestrates the workflow by decomposing the user query into sub-queries, delegating them to retrieval agents, evaluating the evidence, and deciding whether the system is ready to respond confidently. Its decision-making follows the *Just Enough* principle: an answer is generated only when the evidence is both sufficient and internally consistent. To implement this, the agent adopts two complementary prompting strategies: *Tree-of-Thought* (Yao et al., 2023), which explores multiple reasoning paths in parallel, and *Chain-of-Thought* (Wei et al., 2022), which enforces coherent, step-by-step logic. If the aggregated evidence is inconclusive or contradictory, the Manager Agent applies *ask-before-answer*: it refrains from speculation and initiates clarification to refine the user query. If clarification fails or key information is missing, it explicitly reports that no answer can be provided. This iterative reasoning workflow is formalized in Algorithm 1.

Retrieval Sub-Agents To retrieve information from distinct sources, the system instantiates two specialized LLM-based retrieval agents: one for FAQs and one for official documents. Each sub-query q_i is processed **concurrently** by a dedicated agent pair A_i, which is instantiated dynamically and invoked through a unified tool interface, implemented as a function named `search_information` and called by the Manager Agent.

- The *FAQ Search Agent* is optimized for high-precision lookup over a curated FAQ repository $\mathcal{F}$. It performs lightweight iterative search, with at most a few reformulation attempts based on result adequacy.
- The *Document Search Agent* performs semantic retrieval over a structured corpus of academic and administrative documents $\mathcal{D}$. This agent follows a more elaborate prompting loop, allowing multiple reformulations when necessary. At each step, it analyzes the query, refines the search expression, and evaluates results to decide whether to continue or stop.

Both agents are strictly grounded: their final responses must be based solely on content returned by their respective retrieval tools, with no speculative or hallucinated generation. Each agent communicates with its backend service using the *Model Context Protocol* (MCP) over *Server-Sent Events* (SSE), allowing low-latency streaming of search results. This behavior implements a constrained form of CoT reasoning applied externally via tool outputs rather than internal deliberation alone.

3.3 Hybrid Retrieval Technique

Each retrieval sub-agent in URASys employs a hybrid search strategy that combines lexical and semantic signals via BM25 and dense vector retrieval (Fan et al., 2024). Rather than aggregating scores via a weighted linear combination (e.g., $\alpha \cdot \text{Dense} + (1-\alpha) \cdot \text{BM25}$), which requires manual tuning of α, we adopt *Reciprocal Rank Fusion* (RRF) (Cormack et al., 2009), a simple yet effective rank-based method that merges results without assuming score normalization or compatibility. Given two ranked lists $\mathcal{L}_1$ and $\mathcal{L}_2$ from BM25 and dense retrieval respectively, the fused score for a

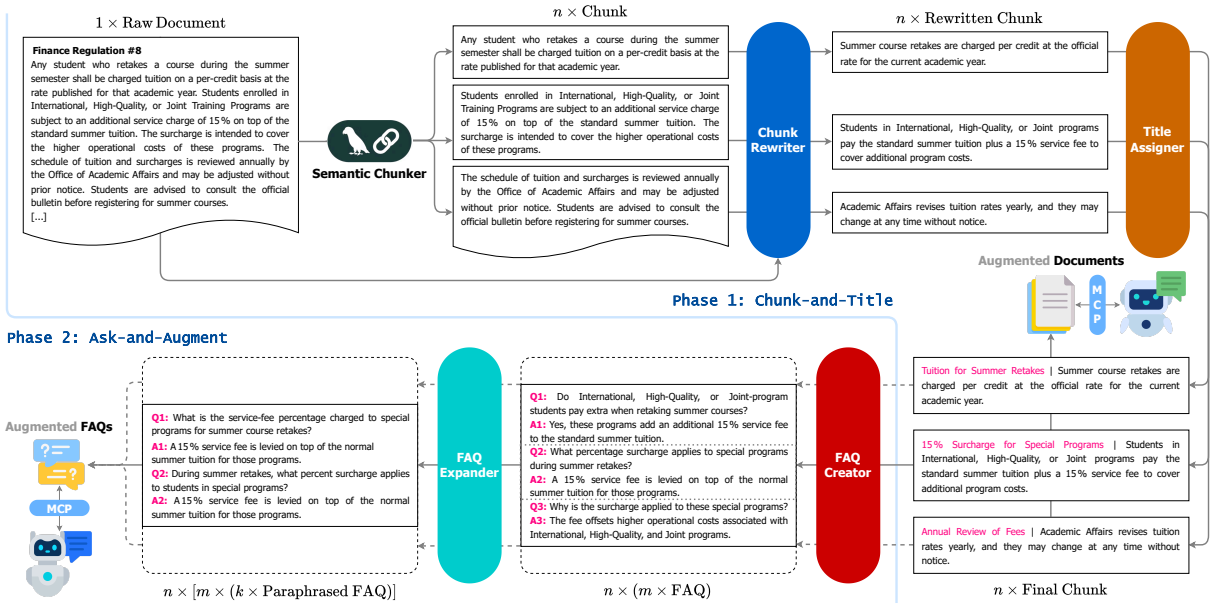


Figure 2: Two-phase indexing pipeline in URASys. Phase 1 (Chunk-and-Title) processes raw documents into coherent text blocks composed of a concise title and a rewritten chunk, enabling effective hybrid document retrieval. Phase 2 (Ask-and-Augment) generates and paraphrases question-answer pairs from each document block, forming a high-coverage and query-resilient FAQ corpus.

document d is computed as shown in Equation 1.

$$s(d) = \sum_{i=1}^{2} \frac{1}{k + \text{rank}_{\mathcal{L}_i}(d)} \tag{1}$$

Here, $\text{rank}_{\mathcal{L}_i}(d)$ denotes the position of d in list $\mathcal{L}_i$, and k is a smoothing constant. This formulation ensures that documents ranked highly in either modality receive strong fused scores, enhancing both robustness and interpretability.

3.4 Proposed Indexing Strategy

To support high-quality retrieval for the two specialized agents in URASys, namely the Document Search Agent and the FAQ Search Agent, we design a two-phase indexing pipeline as illustrated in Figure 2. Each phase constructs a distinct type of retrieval unit tailored to the specific needs of its corresponding agent. This pipeline is tightly coupled with a suite of LLM-based modules, including the *Chunk Rewriter*, *Title Assigner*, *FAQ Creator*, and *FAQ Expander*, all implemented through prompt-based techniques (Kamath et al., 2024). These components enable flexible adaptation to new domains and play a central role in generating semantically rich and query-resilient retrieval units.

Phase 1: Chunk-and-Title Given a raw document d, we apply a semantic chunking module to segment it into a set of discourse-aligned fragments $\texttt{SemanticChunker}(d) \rightarrow \{d_1, d_2, \ldots, d_n\}$, where each d_i is a semantically coherent span. Because these initial fragments may include mid-sentence boundaries or depend on broader context, each d_i is rewritten with document-level context using a context-aware module $\texttt{ChunkRewriter}(d_i, d) \rightarrow c_i$ to produce a self-contained and fluent chunk. Each rewritten chunk c_i is then passed through a title assignment function $\texttt{TitleAssigner}(c_i) \rightarrow t_i$, which generates a concise and descriptive title summarizing the core content. Finally, the title and chunk are concatenated into a single final chunk $x_i = \texttt{Concat}(t_i, c_i)$, forming an augmented document corpus

$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$$

where each x_i is a unified retrieval unit that combines a semantic anchor with its associated content. This corpus serves as the retrieval basis for the Document Search Agent, which performs hybrid retrieval over each x_i using both lexical and semantic signals. The inclusion of titles enhances retrieval effectiveness by injecting salient keywords that benefit sparse retrieval, while preserving the semantic richness of the underlying chunk.

Phase 2: Ask-and-Augment Each final chunk $x_i \in \mathcal{D}$, which contains both the generated title and rewritten chunk, is passed to a FAQ generation module $\texttt{FAQCreator}(x_i)$ to synthesize m canonical FAQ pairs $\{(q_{i,1}, a_{i,1}), \ldots, (q_{i,m}, a_{i,m})\}$.

Algorithm 1: Manager Agent Reasoning

Input: Query q, LLM p_θ, FAQ corpus $\mathcal{F}$, document corpus $\mathcal{D}$, max attempts T
Output: Answer a or clarification q_c

$t \leftarrow 0$;
while $t < T$ **do**
  $S \leftarrow \texttt{Decompose}(q)$;
  $E \leftarrow \emptyset$;
  foreach $q_i \in S$ **do**
    $E \leftarrow E \cup \texttt{FAQSearch}(q_i, \mathcal{F}) \cup \texttt{DocSearch}(q_i, \mathcal{D})$;
  if $\texttt{IsSpecific}(q)$ ***and*** $\texttt{HasDirectAnswer}(E)$ **then**
    $a \leftarrow \texttt{GenerateAnswer}(p_\theta, E)$;
    return a ;
  else if $\texttt{IsBroad}(q)$ ***and*** $\texttt{RevealCategories}(E)$ **then**
    $q_c \leftarrow \texttt{ExtractCategories}(E)$;
    return $\texttt{AskClarification}(q_c)$;
  else if $\texttt{IsVague}(q)$ ***or*** $\texttt{Insufficient}(E)$ **then**
    if $t + 1 < T$ **then**
      $q \leftarrow \texttt{Refine}(q, E)$;
    else
      return $\texttt{NoInformationFound}()$;
  $t \leftarrow t + 1$;

These questions are designed to reflect plausible user intents, guided by the semantic scope introduced by the title and grounded in the content of the chunk. To improve robustness against surface variation in user phrasing, each question $q_{i,j}$ is paraphrased into k diverse variants via $\texttt{FAQExpander}(q_{i,j}) \rightarrow \{q_{i,j}^{(1)}, \ldots, q_{i,j}^{(k)}\}$, all sharing the same answer $a_{i,j}$. The result is a richly paraphrased FAQ corpus

$$\mathcal{F} = \{(q_{i,j}^{(l)}, a_{i,j}) \mid i \in [1, n],\ j \in [1, m],\ l \in [1, k]\}$$

which serves as the retrieval basis for the FAQ Search Agent. The inclusion of multiple paraphrases for each intent improves coverage and increases robustness to syntactic and stylistic variation, which is especially important for Vietnamese, where the same meaning can be expressed in many different ways.

4 Experimentations

We conduct two experiments to evaluate URASys in terms of retrieval performance and real-world usability. The first benchmarks our system against a range of *state-of-the-art* (SOTA) and classical RAG baselines across multiple public QA datasets, covering diverse reasoning types and domain settings. The second is a user study with real end-users to assess practical effectiveness and user trust under different interaction scenarios.

4.1 Datasets

We evaluate URASys on five datasets spanning three QA settings: single-hop, multi-hop, and domain-specific queries. For each public dataset, we sample 1,000 representative questions. Several include **unanswerable cases**, making them well-suited for testing the system's ability to handle uncertainty and trigger clarification.

Single-hop QA SQuAD 2.0 (Rajpurkar et al., 2018) has English questions from Wikipedia, including adversarially unanswerable. UIT-ViQuAD 2.0 (Nguyen et al., 2022) is its Vietnamese counterpart with similar design.

Multi-hop QA HotpotQA (Yang et al., 2018) and VIMQA (Le et al., 2022) require reasoning over multiple documents. VIMQA adapts this to Vietnamese, making it suitable for low-resource multi-hop evaluation.

Domain-specific QA UniQA is a custom dataset of real-world Vietnamese student queries on university admissions. Each question links to official academic documents, reflecting URASys's target deployment scenario.

We also build **ambiguous subsets** from SQuAD 2.0, UIT-ViQuAD 2.0, and UniQA. These include underspecified questions requiring clarification, paired with ground-truth paraphrases of clarified queries, providing a dedicated benchmark for interactive clarification in both English and Vietnamese.

4.2 Evaluation Metrics

Standard metrics like *Exact Match* (EM) and token-level F1 often overlook reasoning quality, especially in multi-hop or underspecified scenarios (Schuff et al., 2020). We instead use the *LLM-as-a-Judge* protocol (Gu et al., 2024), where an external model scores answer correctness and explanation quality. For unanswerable subsets, models must indicate insufficient information, while for ambiguous ones they should request necessary clarifications. We also conduct a *human evaluation*, with participants rating outputs on seven dimensions (e.g., factuality, trust) using a 5-point *Likert*

Table 2: Answer correctness percentages (↑) across five QA benchmarks. For datasets with unanswerable and ambiguous questions (SQuAD 2.0, UIT-ViQuAD 2.0, UniQA), results are split into Overall, Unanswerable (Unans.), and Ambiguous (Ambig.) subsets. Best scores are in **bold**; second-best are underlined.

Method	SQuAD 2.0			UIT-ViQuAD 2.0			HotpotQA	VIMQA	UniQA		
	Overall	Unans.	Ambig.	Overall	Unans.	Ambig.			Overall	Unans.	Ambig.
Naive RAG (Dense)	30.3	7.2	14.8	18.2	2.9	11.6	11.3	41.2	40.1	9.2	13.8
Naive RAG (BM25)	30.1	13.6	12.9	18.7	3.4	31.3	11.8	40.0	39.8	9.3	8.6
Naive RAG (Hybrid)	30.3	8.3	10.6	18.3	3.7	21.3	11.6	42.4	41.3	9.03	3.2
IRCoT	45.3	33.4	13.7	46.9	28.0	31.7	43.2	20.7	56.7	9.4	67.5
LightRAG	43.4	36.2	18.2	44.3	49.0	45.6	59.0	76.0	51.8	51.0	68.2
MiniRAG	49.0	10.6	17.5	55.0	11.3	70.5	21.7	54.5	52.6	49.0	36.9
NodeRAG	44.1	23.0	69.3	48.1	8.1	68.3	56.7	45.3	67.7	14.0	32.5
HippoRAG 2	67.3	56.2	67.4	78.8	54.0	62.7	60.3	75.0	50.7	13.0	49.7
URASys (Ours)	**75.0**	**83.7**	**71.9**	**80.0**	**86.5**	**73.9**	**90.0**	**83.2**	**85.0**	**82.6**	**81.2**

scale (Batterton and Hale, 2017), complementing automatic metrics with user-centered feedback.

4.3 Baselines

We evaluate URASys by comparing it against the following baselines.

Naive RAG A basic RAG pipeline using BM25, dense retrieval, and hybrid retrieval with score interpolation (Fan et al., 2024).

IRCoT + SOTA LLM Multi-step QA approach interleaving retrieval and CoT reasoning using a SOTA LLM (Trivedi et al., 2023).

LightRAG Incorporates graph structures into text indexing and retrieval (Guo et al., 2024).

MiniRAG Lightweight system with small LLM and heterogeneous graph index for efficient structured retrieval (Fan et al., 2025).

NodeRAG Graph-based framework integrating structured evidence for improved multi-hop retrieval (Xu et al., 2025).

HippoRAG 2 Retrieval system inspired by hippocampal theory for better long-term knowledge integration (Gutiérrez et al., 2025).

4.4 Implementation Details

To ensure reproducibility and fair comparison, all systems share the same model backbone and evaluation protocol. We adopt `gemini-2.0-flash`[1] as the unified LLM backbone, selected for its strong balance between reasoning accuracy and computational efficiency. Text embeddings are derived from OpenAI's `text-embedding-3-large`[2], a multilingual representation model supporting both English and Vietnamese data. All retrieval and generation baselines are run with default hyperparameters to reflect realistic out-of-the-box behavior rather than tuned performance. LLM-as-a-Judge evaluations are performed with the GPT-4o API[3] under a standardized rubric to maintain consistency across systems. For our ambiguous subsets, prompt templates are designed to allow the model to proactively request clarification when query context is insufficient, ensuring that clarification behaviors are assessed uniformly across datasets and languages.

4.5 Results and Analysis

Table 2 summarizes answer correctness across five QA benchmarks, partitioned into Overall, Unanswerable, and Ambiguous subsets. URASys consistently achieves the highest performance across all settings, with particularly large gains on unanswerable questions (83.7% on SQuAD 2.0 and 86.5% on UIT-ViQuAD 2.0) and ambiguous cases (up to 81.2% on UniQA), outperforming the next-best system, HippoRAG 2, by notable margins. Although models such as NodeRAG and HippoRAG 2 perform competitively on English datasets, their accuracy declines markedly on Vietnamese ambiguous and unanswerable subsets, revealing persistent cross-lingual challenges. Lightweight models like MiniRAG show moderate competitiveness on certain Vietnamese subsets but fall short of URASys's dual-reasoning and interactive clarification capabilities. In contrast, traditional RAG baselines relying on single-pass retrieval struggle with complex or underspecified queries, underscoring the advantages of URASys's agent-based, dual-retrieval design. Overall, URASys demonstrates strong gener-

[1] https://ai.google.dev/gemini-api/docs/models/#gemini-2.0-flash

[2] https://platform.openai.com/docs/models/text-embedding-3-large

[3] https://platform.openai.com/docs/models/gpt-4o

Table 3: Human evaluation scores from end-user deployment. *Accuracy* is binary; *Number of Thoughts* (NoT) counts reasoning steps per answer; other metrics are rated on a 5-point Likert scale (↑).

Group	Accuracy	NoT	Explanation Quality	Trust
G1	0.80	1.95	4.51	4.26
G2	0.88	2.24	4.76	4.37

Table 4: Ablation study results across five QA datasets under the LLM-as-a-Judge protocol.

System Variant	SQuAD 2.0	UIT-ViQuAD 2.0	HotpotQA	VIMQA	UniQA
URASys	**0.75**	**0.80**	**0.90**	**0.83**	**0.85**
w/o FAQ Search Agent	0.73	0.26	0.43	0.14	0.48
w/o Document Search Agent	0.59	0.16	0.86	0.15	0.62
w/o Manager Decomposition	0.72	0.22	0.87	0.21	0.63
w/o Proposed Indexing	0.71	0.29	0.85	0.29	0.64
w/o Clarification	0.64	0.37	0.76	0.44	0.50

alization across languages, domains, and question complexities, effectively bridging the gap between benchmark QA and real-world ambiguous scenarios in multilingual, low-resource contexts.

Table 3 presents findings from a human evaluation conducted with real users. URASys was deployed to two groups of prospective university applicants: *10 first-year students* (G1) and *10 high-school seniors* (G2), each providing 20 randomly sampled queries on university-related topics. Across both groups, the system maintained high correctness (>0.80) while requiring on average only two reasoning turns per query. Participants rated URASys favorably on both explanation quality and trustworthiness (average >4.2/5), highlighting its practical potential as an educational advising agent in authentic, open-ended settings.

4.6 Ablation Study

We perform an ablation study by removing individual components of URASys and measuring their impact on answer correctness across five QA benchmarks (Table 4). The complete system consistently attains the best results, confirming that its performance arises from the synergy among modules. On SQuAD 2.0, an English single-hop dataset, removing any component yields only minor changes (<5%), indicating balanced contributions when queries are well specified. In contrast, accuracy drops sharply on ViQuAD 2.0, especially when the Document Search Agent or query decomposition module is removed, with over 60% accuracy loss. Replacing the Chunk-and-Title index with a standard chunking baseline similarly degrades performance, emphasizing the importance of structured indexing for ambiguous Vietnamese inputs. For multi-hop datasets, removing the FAQ Search Agent causes a 52% drop on HotpotQA and 69% on VIMQA, showing that decomposing queries and retrieving relevant FAQs is essential for multi-step reasoning in low-resource settings. On UniQA, a real-world educational dataset, eliminating the FAQ agent results in the steepest decline (0.85→0.48), while other ablations reduce accuracy by 20–25%. Overall, the results highlight the pivotal role of FAQ retrieval, structured indexing, and clarification in domains where user questions are often vague or fragmented, confirming that URASys's retrieval and interaction components are complementary and jointly necessary for robust performance.

5 Conclusion

We present URASys, a modular and interaction-aware QA system tailored for educational scenarios. Evaluated on five benchmarks, including multi-hop and real-world academic datasets, URASys consistently outperforms retrieval-based baselines in factual accuracy and user trust. Its effectiveness stems from integration of query decomposition, dual-agent retrieval, and structure-aware indexing, as shown by our ablation study. While designed for education, the system architecture generalizes well to domains where queries tend to be vague, underspecified, or context-dependent, such as technical support or legal consultation. In such cases,

URASys can identify missing information and opt not to answer until clarification is obtained, preserving factual integrity. Future directions include improving intent alignment in open-ended queries, seamless updates to evolving knowledge sources, and gradually replacing commercial LLM APIs with in-house lightweight models.

Acknowledgments

We thank Ho Chi Minh City University of Technology (HCMUT), VNU-HCM, for supporting this study. We also acknowledge Mr. Le Hoang Anh Tai and Ms. Huynh Tieu Phung for their assistance in refining the URASys codebase. Our sincere appreciation goes to the students and high-school participants who contributed to the human evaluation for their time and thoughtful feedback. The URASys project was recognized by NVIDIA as a representative use case of Agentic AI in Vietnam and was subsequently featured by Dr. Ettikan Karuppiah, Director at NVIDIA – Asia Pacific South Region, during his talk at AI Day Ho Chi Minh City 2025.

Limitations

While URASys demonstrates strong performance and broad adaptability across QA scenarios, several practical limitations remain. First, the system relies on a prompting strategy that may require careful tuning and ongoing maintenance, particularly in dynamic or evolving domains. Second, although overall computation is lightweight and stable, the use of multiple LLM calls across agents can incur notable monetary cost when relying on commercial APIs. Third, while URASys is designed for responsiveness and clarification, latency may increase for complex queries that involve deep decomposition or iterative refinement. These trade-offs between transparency, flexibility, and cost highlight directions for future work, including more streamlined agent orchestration and the adoption of lightweight, self-hosted LLMs.

Supplementary Materials Availability Statements

All datasets used in our experiments are publicly available or accessible under minimal conditions. Specifically, SQuAD 2.0, UIT-ViQuAD 2.0 (via the VLSP 2021 ViMRC Challenge), and HotpotQA are freely accessible online. Access to VIMQA requires signing a user agreement and direct contact with the dataset maintainers. Code for baseline systems, including NodeRAG (Xu et al., 2025), HippoRAG 2 (Gutiérrez et al., 2025), and MiniRAG (Fan et al., 2025), was retrieved from their official repositories using the latest versions available as of July 1, 2025. The UniQA dataset, its accompanying academic document collection, and the complete URASys implementation are released at `https://github.com/ura-hcmut/URASys`, **including the ambiguous subsets** used in evaluation.

References

Katherine A. Batterton and Kimberly N. Hale. 2017. The Likert Scale What It Is and How To Use It. *Phalanx*, 50(2):32–39.

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA. Association for Computing Machinery.

Yang Deng, Lizi Liao, Wenqiang Lei, Grace Hui Yang, Wai Lam, and Tat-Seng Chua. 2025. Proactive Conversational AI: A Comprehensive Survey of Advancements and Opportunities. *ACM Trans. Inf. Syst.*, 43(3).

Yang Deng, Lizi Liao, Zhonghua Zheng, Grace Hui Yang, and Tat-Seng Chua. 2024. Towards Human-centered Proactive Conversational Agents. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 807–818, New York, NY, USA. Association for Computing Machinery.

Michelle Elizabeth, Morgan Veyret, Miguel Couceiro, Ondrej Dusek, and Lina M. Rojas Barahona. 2025. Exploring ReAct Prompting for Task-Oriented Dialogue: Insights and Shortcomings. In *Proceedings of the 15th International Workshop on Spoken Dialogue Systems Technology*, pages 143–153, Bilbao, Spain. Association for Computational Linguistics.

Tianyu Fan, Jingyuan Wang, Xubin Ren, and Chao Huang. 2025. MiniRAG: Towards Extremely Simple Retrieval-Augmented Generation.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 6491–6501, New York, NY, USA. Association for Computing Machinery.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen,

Shengjie Ma, Honghao Liu, et al. 2024. A survey on llm-as-a-judge.

Meiqi Guo, Mingda Zhang, Siva Reddy, and Malihe Alikhani. 2021. Abg-CoQA: Clarifying Ambiguity in Conversational Question Answering. In *3rd Conference on Automated Knowledge Base Construction*.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. LightRAG: Simple and Fast Retrieval-Augmented Generation.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to Memory: Non-Parametric Continual Learning for Large Language Models.

Uday Kamath, Kevin Keenan, Garrett Somers, and Sarah Sorenson. 2024. *Prompt-based Learning*, pages 83–133. Springer Nature Switzerland, Cham.

Khang Le, Hien Nguyen, Tung Le Thanh, and Minh Nguyen. 2022. VIMQA: A Vietnamese Dataset for Advanced Reasoning and Explainable Multi-hop Question Answering. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6521–6529, Marseille, France. European Language Resources Association.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024. The Dawn After the Dark: An Empirical Study on Factuality Hallucination in Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10879–10899, Bangkok, Thailand. Association for Computational Linguistics.

Hao Liu, Zhengren Wang, Xi Chen, Zhiyu Li, Feiyu Xiong, Qinhan Yu, and Wentao Zhang. 2025. HopRAG: Multi-Hop Reasoning for Logic-Aware Retrieval-Augmented Generation.

Kiet Nguyen, Son Quoc Tran, Luan Thanh Nguyen, Tin Van Huynh, Son Thanh Luu, and Ngan Luu-Thuy Nguyen. 2022. VLSP 2021 - ViMRC Challenge: Vietnamese Machine Reading Comprehension. *VNU Journal of Science: Computer Science and Communication Engineering*, 38(2).

Long S. T. Nguyen and Tho T. Quan. 2025. URAG: Implementing a Unified Hybrid RAG for Precise Answers in University Admission Chatbots – A Case Study at HCMUT. In *Information and Communication Technology*, pages 82–93, Singapore. Springer Nature Singapore.

Quoc-Hung Pham, Huu-Loi Le, Minh Dang Nhat, Khang Tran T., Manh Tran-Tien, Viet-Hung Dang, Huy-The Vu, Minh-Tien Nguyen, and Xuan-Hieu Phan. 2024. Towards Vietnamese Question and Answer Generation: An Empirical Study. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 23(9).

Hossein A. Rahmani, Xi Wang, Yue Feng, Qiang Zhang, Emine Yilmaz, and Aldo Lipani. 2023. A Survey on Asking Clarification Questions Datasets in Conversational Systems. In *The 61st Annual Meeting of the Association for Computational Linguistics*.

Mohaimenul Azam Khan Raiaan, Md. Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. 2024. A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges. *IEEE Access*, 12:26839–26874.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Zafaryab Rasool, Stefanus Kurniawan, Sherwin Balugo, Scott Barnett, Rajesh Vasa, Courtney Chesser, Benjamin M. Hampstead, Sylvie Belleville, Kon Mouzakis, and Alex Bahar-Fuchs. 2024. Evaluating LLMs on document-based QA: Exact answer selection and numerical extraction using CogTale dataset. *Natural Language Processing Journal*, 8:100083.

Hendrik Schuff, Heike Adel, and Ngoc Thang Vu. 2020. F1 is Not Enough! Models and Evaluation Towards User-Centered Explainable Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7076–7095, Online. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

Nethra Viswanathan, Sofia Meacham, and Festus Fatai Adedoyin. 2022. Enhancement of online education system by using a multi-agent approach. *Computers and Education: Artificial Intelligence*, 3:100057.

Bing Wang, Yan Gao, Zhoujun Li, and Jian-Guang Lou. 2023. Know What I don't Know: Handling Ambiguous and Unknown Questions for Text-to-SQL. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5701–5714, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*.

Tianyang Xu, Haojie Zheng, Chengze Li, Haoxiang Chen, Yixin Liu, Ruoxi Chen, and Lichao Sun. 2025. NodeRAG: Structuring Graph-based RAG with Heterogeneous Nodes.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

A Dataset Statistics

To better characterize the datasets used in our experiments, we report descriptive statistics on context documents and evaluation queries. Table 5 presents word-level statistics, including the number of context documents, the maximum, minimum, and average context length, the number of evaluation queries, and the number of unanswerable queries where applicable. The datasets vary widely in both context structure and query types. UniQA, derived from real-world educational content, contains significantly longer passages, with an average length over 1,200 words and a maximum exceeding 5,000. URASys maintains strong performance under these conditions, suggesting that it handles long-form, high-complexity contexts effectively.

B URASys Configuration

URASys is instantiated with a lightweight and consistent configuration across all agents and indexing modules. Table 6 summarizes the key parameters used throughout the system, covering both the URASys agent layer and the two-phase indexing pipeline employed in all experiments.

C LLM-as-a-Judge Protocol

We adopt a deterministic LLM-as-a-Judge protocol to ensure consistent and unbiased evaluation across systems. Two prompts are used: one for standard QA and unanswerable cases (Figure 3), and one for ambiguous questions that require clarification (Figure 4). All judgments are produced using GPT-4o with `temperature = 0.0` and `maxTokens = 32` to guarantee deterministic behavior. The judge model is fully blinded to the identity of the system that generated each prediction.

D Multi-hop Performance Breakdown

To examine model behavior under varying reasoning demands, we perform a stratified analysis on HotpotQA and VIMQA by grouping questions by hop depth. Table 7 reports accuracy across 2-hop, 3-hop, 4-hop, and 5+ hop categories.

URASys consistently achieves the highest performance across all reasoning depths. It attains 0.87 and 0.92 on HotpotQA 2-hop and 3-hop questions, surpassing the next-best system (LightRAG at 0.81 and 0.73). This trend aligns with Table 3, where URASys produces roughly two reasoning steps per answer, indicating effective decomposition and stable evidence aggregation. Traditional RAG pipelines degrade sharply as hop depth increases: Dense and BM25 retrieval perform adequately on 2-hop questions but collapse at deeper levels (e.g., Dense RAG: 0.19 $\rightarrow$ 0.02 from 2-hop to 4-hop), reflecting difficulty in combining dispersed evidence. A similar pattern appears on VIMQA, where accuracy falls to less than 0.03 at 4-hop and 5+ hop. Graph-based systems (NodeRAG, HippoRAG) show stronger multi-hop behavior; for example, NodeRAG reaches 0.66 on HotpotQA 2-hop and HippoRAG achieves 0.87 on 4-hop, although their performance varies substantially across datasets and hop levels.

URASys maintains robustness across higher depths by combining specialized retrieval (FAQ and document agents) with adaptive reasoning and an ask-before-answer policy that prevents speculation under insufficient evidence. Its two-phase indexing pipeline broadens access to latent supporting cues, enabling more reliable retrieval even when reasoning chains span multiple documents. Although multi-hop QA is not the primary target of URASys, these results show that its architecture generalizes effectively and remains competitive under challenging multi-hop conditions.

Table 5: Word-level context statistics and evaluation query composition across QA datasets.

Metric	SQuAD 2.0	UIT-ViQuAD 2.0	HotpotQA	VIMQA	UniQA
Context document count	46	934	996	1000	42
Maximum context length	259	613	2075	676	5153
Minimum context length	27	99	103	8	298
Mean context length	91.1	182.1	972.4	264.7	1266.2
Evaluation query count	1417	1731	1000	1000	888
Unanswerable queries	492	200	–	–	38
Ambiguous queries	417	731	–	–	314

Table 6: URASys configuration and agent-level hyperparameters.

Component	Hyperparameter	Value
URASys (LLM backend: `gemini-2.0-flash`)		
Embedding Backend	Model	`text-embedding-3-large`
	Dimension	3072
Manager Agent	Temperature / Top-p	0.2 / 0.1
	Max retries	3
	Thinking budget / Thoughts	100 / enabled
Document Search Agent	Temperature / Top-p	0.1 / 0.1
	Max tool calls	3
	Retrieval top-k	3–5 (adaptive)
	Thoughts in planner	disabled
FAQ Search Agent	Temperature / Top-p	0.1 / 0.1
	Max tool calls	3
	Retrieval top-k	3–5 (adaptive)
	Thoughts in planner	disabled
Indexing Phase (LLM backend: `gemini-2.5-flash`)		
LLM Configuration	Temperature / Top-p	0.1 / 0.95
	Max generation tokens	8192
Embedding Backend	Model	`text-embedding-3-small`
	Dimension	1536
Semantic Chunker	Breakpoint percentile	95
	Buffer size	1
	Min/Max chunk size	10 / 1000 tokens
FAQ Creator	Max FAQ pairs per chunk	5
FAQ Expander	Max paraphrases per FAQ	3

E Details of Human Evaluation

E.1 Procedure and Evaluation Criteria

To complement the LLM-as-a-Judge protocol, we conducted a human evaluation involving 20 target end-users, including *10 university freshmen* (G1) and *10 high school seniors* (G2). These participants were randomly recruited from admission-related events and information sessions hosted by our university. Each participant received a brief introduction to URASys and was invited to interact freely with the system by asking 20 questions of personal

```
SYSTEM_PROMPT = """
### Role
You are an expert language model evaluator. Your task is to evaluate the model prediction given a
     ground truth.
### Instruction
- If the prediction is "no answer" -> False
- If the prediction contains the ground truth verbatim -> True
- If the prediction paraphrases the ground truth -> True
- If the prediction contradicts the ground truth -> False
- Evaluation is language-agnostic: ignore whether the prediction is in English or Vietnamese.
### Note
- The prediction may contain extra explanatory text: ignore it.
"""
```

Figure 3: Prompt used for standard QA and unanswerable evaluation.

```
SYSTEM_PROMPT = """
You are an evaluator for an ambiguous question answering system.
Evaluation Rules:
1. You are given: the question, a list of required info items, the correct answer, and the model'
     s prediction.
2. If the prediction is a clarification question:
    - CORRECT if it explicitly asks for at least one required info item.
    - INCORRECT if it does not ask for any required info items.
3. If the prediction is a direct answer:
    - CORRECT if it matches or clearly paraphrases the correct answer.
    - INCORRECT otherwise.
4. If the prediction refuses to answer, says "I don't know", or indicates insufficient
     information:
    - Always INCORRECT.
5. Ignore the language used; evaluate purely based on content.
6. Output exactly one word: CORRECT or INCORRECT.
"""
```

Figure 4: Prompt used for ambiguous QA and clarification evaluation.

interest, as long as the topics fell within the scope of the system's indexed data. Each system response was evaluated along nine dimensions, as follows.

- **Number of Thoughts (NoT):** The number of intermediate reasoning steps generated before reaching a final answer. Higher values often indicate more structured or multi-step reasoning.
- **Accuracy:** Whether the final answer is factually correct, assessed using a binary label indicating *Correct* or *Incorrect*.
- **User Experience (UX):** Overall satisfaction with the system's interface, clarity of output, and ease of interaction.
- **Explanation Quality:** The clarity, coherence, and usefulness of the accompanying explanation, especially in helping users understand the rationale behind the answer.
- **Factuality:** The extent to which the explanation contains accurate and verifiable information supported by retrieved evidence.
- **Completeness:** Whether the system's output fully addresses the user's question without omitting important aspects.
- **Fluency:** The grammatical correctness and naturalness of the language used.
- **Relevance:** How directly the answer and explanation pertain to the original question, avoiding irrelevant or off-topic content.
- **Trust:** The participant's confidence in the system's response, influenced by tone, coherence, and evidential grounding.

E.2 Summary of Results

In addition to Table 3, which reports scores on Accuracy, NoT, Explanation Quality, and Trust, we present the remaining human evaluation results in

Table 7: Answer accuracy breakdown on HotpotQA and VIMQA across reasoning depths.

Method	HotpotQA				VIMQA			
	2-hop	3-hop	4-hop	5+ hop	2-hop	3-hop	4-hop	5+ hop
Sample count	550	300	121	29	500	300	150	50
Naive RAG (Dense)	0.19	0.01	0.02	0.00	0.71	0.17	0.03	0.02
Naive RAG (BM25)	0.19	0.02	0.04	0.03	0.58	0.33	0.47	0.00
Naive RAG (Hybrid)	0.20	0.01	0.02	0.03	0.63	0.30	0.13	0.02
IRCoT	0.34	0.06	0.02	0.00	0.37	0.07	0.03	0.00
LightRAG	0.81	0.73	0.71	0.41	0.73	0.86	0.53	0.12
MiniRAG	0.36	0.05	0.03	0.00	0.76	0.46	0.18	0.02
NodeRAG	0.66	0.54	0.41	0.00	0.76	0.21	0.07	0.00
HippoRAG 2	0.75	0.52	0.87	0.13	0.73	0.87	0.23	0.15
URASys (Ours)	**0.87**	**0.92**	**0.90**	**0.43**	**0.91**	**0.91**	**0.60**	**0.16**

Table 8: Average human evaluation scores by user group on five additional dimensions.

Group	UX	Factuality	Completeness	Fluency	Relevance
G1	4.61	4.34	3.45	4.96	4.03
G2	3.73	4.47	3.55	4.86	4.75

Table 8. These scores reflect average ratings from each user group on a 5-point Likert scale (↑).

G1 reported higher satisfaction in terms of user experience (4.61) compared to G2 (3.73), likely because university freshmen are more accustomed to navigating institutional systems, whereas high school seniors may be more cautious due to the importance of the information for their university decisions or their familiarity with traditional advising. Both groups gave strong ratings for fluency and factuality, indicating that URASys responses were generally clear, coherent, and grounded in reliable evidence. However, completeness received lower scores (3.45 and 3.55), possibly because the system prompts for clarification when facing vague queries instead of guessing, which, while improving factual accuracy, can make the final answer feel unresolved. This trade-off may also affect perceived relevance, reflected in G1's lower relevance score (4.03) compared to G2 (4.75). Still, trust and accuracy remain high across both groups, confirming that URASys meets its core objective of delivering reliable, well-supported answers in educational settings.

The results suggest that different user groups may prioritize different aspects of system behavior, such as interaction flow versus completeness, depending on their background and expectations.

F Construction of Ambiguous Subsets

To evaluate URASys on its ability to handle ambiguous queries, we construct dedicated ambiguous subsets that avoid leakage, systematic bias, and any overlap with the system's FAQ index. The full construction process is summarized below.

- For each (question, answer, paragraph) sample in the original dataset, we assign one ambiguity type from the three categories listed below, with respective probabilities of 30%, 40%, and 30%.
- We construct a structured LLM prompt that generates a hard ambiguous variant of the question according to the assigned type:
  - **Missing context:** remove or modify a central subject, temporal reference, or condition.
  - **Multiple interpretations:** rewrite the question so that it supports two or more semantically plausible interpretations.
  - **Generalization:** transform the question into an overly broad, generic, or intentionally underspecified form.

```
{
  "question": "Which campaign promoting female empowerment did she support?",
  "info": ["Which artist is being referred to", "Whether the campaign is music-related or social"]
      ,
  "answer": "Ban Bossy campaign",
  "paragraph": "In an interview published by Vogue in April 2013,Beyonce was asked if she
      considers herself a feminist... She has also contributed to the Ban Bossy campaign..."
}
```

Figure 5: Example item following the ambiguous-subset annotation schema.

Table 9: Average inference latency (in seconds per query) across datasets.

Method	SQuAD 2.0	UIT-ViQuAD 2.0	HotpotQA	VIMQA	UniQA
Naive RAG (Dense)	5.87	11.98	10.76	13.87	14.78
Naive RAG (BM25)	4.57	8.65	9.34	11.34	13.14
Naive RAG (Hybrid)	**7.34**	13.24	12.37	15.78	**18.98**
LightRAG	4.96	6.54	5.29	8.65	9.64
MiniRAG	1.51	2.24	2.32	1.40	1.88
NodeRAG	3.12	2.61	4.03	2.09	4.94
HippoRAG 2	4.12	3.25	1.37	1.46	7.34
URASys (Ours)	3.67	**31.24**	**24.46**	**26.23**	15.45

The LLM must output a strictly valid JSON object following the schema in Figure 5.

- We allow up to three generation attempts. A sample is accepted only if the JSON is syntactically valid and the masked question exhibits both semantic similarity and lexical overlap below 0.75 relative to the original question. This criterion ensures genuine novelty and prevents superficial edits. Samples that fail all attempts are discarded.
- To prevent contamination or implicit bias toward URASys, the construction of ambiguous subsets is fully isolated from the FAQ-index generation pipeline. No ambiguous samples or their metadata were incorporated into the creation of retrieval components.

G Latency Analysis

Table 9 reports the average per-query processing time across all datasets. As expected, URASys is slower than lightweight or single-pass RAG baselines, but its latency remains within practical bounds for interactive counseling and advisory use cases. The results also reveal a broader pattern: methods that employ deeper reasoning or multi-step retrieval (e.g., LightRAG and HippoRAG) generally incur higher latency, whereas shallow retrievers respond faster but offer substantially lower accuracy. A notable exception is the Naive Hybrid RAG baseline, which shows the highest latency on several datasets despite lacking explicit reasoning; its two-stage retrieval and fusion process introduce considerable overhead without providing adaptive interaction.

URASys latency primarily arises from two intentional design choices: (i) dual retrieval passes (FAQ and document search), and (ii) clarification checks executed by the manager agent. While these steps add computational cost, they are essential for achieving robustness on ambiguous or potentially unanswerable queries. In practice, URASys accepts a modest latency increase in exchange for significantly improved factual reliability, clarification behavior, and overall answer quality, representing a domain-appropriate trade-off.

H Additional Experiments and Analysis

H.1 Comparison with Standalone LLMs

To assess the generalizability of URASys beyond retrieval-augmented systems, we compare it against three SOTA LLM baselines: GPT-4o[4],

[4] https://platform.openai.com/docs/models/gpt-4o

Table 10: Comparison of SOTA methods on the UniQA dataset for university admission counseling. Results are reported across three subsets: Overall, Unanswerable, and Ambiguous.

Method	Overall	Unans.	Ambig.
GPT-4o	49.3	37.1	18.2
Gemini 2.5 Pro	24.0	25.8	12.1
Claude 3.7 Sonnet	37.8	16.9	15.4
URASys (Ours)	**85.0**	**82.6**	**81.2**

Table 11: Descriptions of the six question topics.

Topic	Description
Admissions Information 2025	Information related to the 2025 university admission cycle.
Program Descriptions	Program-level details including curriculum design, study duration, admission tracks, post-graduation prospects, subject requirements, and distinctive features.
Administration Contacts	Names, roles, and contact information of university leaders and administrative units.
Faculty Overviews	High-level faculty descriptions covering organization, mission, sub-units, capacity, and achievements.
Laboratories	Laboratory introductions, equipment availability, usage policies, eligibility, and reservation instructions.
Others	Questions outside the five defined categories.

Gemini 2.5 Pro[5], and Claude 3.7 Sonnet[6]. All models are evaluated under identical prompts on the UniQA dataset, with real-time search enabled via the Brave Search API[7]. As shown in Table 10, URASys significantly outperforms these baselines across all subsets, yielding 30–60% absolute accuracy gains. While GPT-4o benefits from external search, it frequently produces confident but partially incorrect answers, and both Gemini 2.5 Pro and Claude 3.7 Sonnet struggle with Vietnamese and unanswerable queries.

This performance gap is partly explained by UniQA documents originating from institutional sources not fully indexed by public search engines. However, even under idealized internet-access conditions, standalone LLMs still fail to maintain factual reliability and clarification behavior. In contrast, URASys sustains high accuracy through structured retrieval and agent-based reasoning, reinforcing the need for domain-grounded retrieval pipelines in multilingual, high-stakes QA.

[5] https://ai.google.dev/gemini-api/docs/models#gemini-2.5-pro

[6] https://www.anthropic.com/news/claude-3-7-sonnet

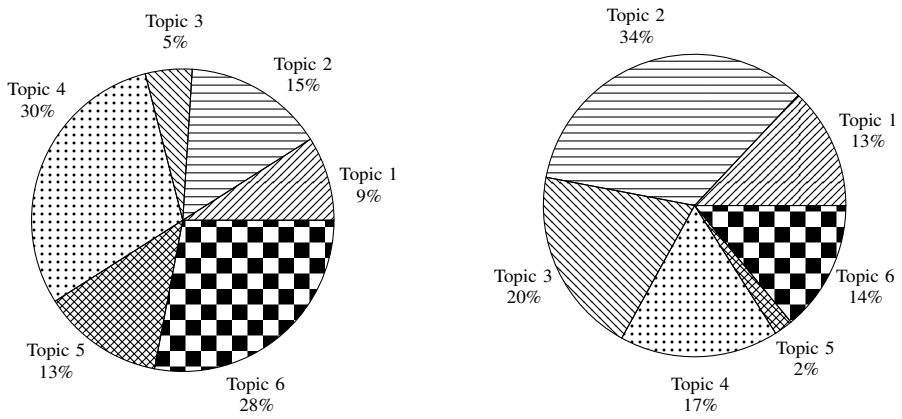
[7] https://brave.com/search/api/

H.2 Analysis of User Query Behavior

After interacting freely with URASys during the human evaluation, participants were asked to categorize all submitted questions into one of six predefined topic categories. The topic taxonomy is summarized in Table 11.

The topic distributions reveal clearly divergent information-seeking patterns across user groups (Figure 6). First-year students demonstrate a strong interest in institutional and organizational knowledge, with the highest proportion of queries targeting faculty-level information (Topic 4, 30%), followed by a notable share classified as Others (28%). This indicates informational needs that extend beyond formal documentation and may involve experiential, procedural, or tacit knowledge not explicitly standardized or publicly disseminated. In contrast, high-school students focus primarily on admission- and program-related questions (Topics 1 and 2, totalling 47%), reflecting a goal-oriented and decision-driven search behavior aligned with pre-enrollment planning.

Overall, these findings emphasize user differentiation in informational depth and scope, highlighting the importance of role-sensitive, intent-aware, and maturity-aligned QA strategies. Systems like



(a) First-year students

(b) High-school students

Figure 6: Comparison of question category distributions between the two participant groups.

URASys may benefit from adaptive response shaping, such as varying explanation granularity, source transparency, and follow-up prompting based on user identity and stage in the academic journey.

I Prompt Templates for Agent Reasoning

As detailed in Section 3.2, each URASys agent is guided by a carefully designed prompt that specifies its reasoning logic, retrieval strategy, and interaction policy. These prompts coordinate the behavior of the Manager Agent as well as the two retrieval sub-agents: the Document Search Agent and the FAQ Search Agent. This configuration supports coherent workflows and ensures consistently high-quality responses across diverse QA scenarios. Figure 7 shows the prompt for the Manager Agent, while Figure 8 and Figure 9 present the prompts used by the Document Search Agent and the FAQ Search Agent, respectively. These are the exact templates employed in the experiments on public datasets discussed in Section 4.

```
MANAGER_AGENT_INSTRUCTION_PROMPT = """
# Persona
You are the "AI Assistant," an expert AI focused on efficiently and accurately answering
    questions using the provided context passages.
# Current State
- Current Search Attempt: {current_attempt}
- Max Search Attempts: {max_retries}
# The Supreme Goal: The "Just Enough" Principle
Your absolute highest priority is to answer the user's *specific, underlying need*, not just the
    broad words they use. You must act as a **guide**, not an information dump. This means:
- If a query is broad, your job is to **help the user specify it.**
- If a query is specific, your job is to **answer it directly.**
- **NEVER** dump a summary of all found information and then ask "what do you want to know more
    about?". This is a critical failure.
# Core Directives
1. **Search is for Understanding:** Your first search on a broad topic is not to find an answer,
    but to **discover the available categories/options** to guide the user.
2. **Troubleshoot Vague Failures:** If a search fails because the user's query is incomplete, ask
     for more clues.
3. **Evidence-Based Actions:** All answers and examples MUST come from the retrieved context
    passages.
4. **Language and Persona Integrity:**
    * All responses **MUST** be in **language based on an user**.
    * **Self-reference:** Use the pronoun **"I"** to refer to yourself. Only state your full name
        if asked directly.
    * **Expert Tone and Phrasing:** You **MUST** speak from a position of knowledge, as a
        representative of the university.
        * **DO:** Use confident, knowledgeable phrasing like: *"Now, I...", "About [topic], I see
            that..."*
        * **AVOID:** **NEVER** use phrases that imply real-time discovery. **FORBIDDEN** phrases
            include: *"I search...", "I have...", "In my researching,..."*
    * **Conceal Internal Mechanics:** **NEVER** mention your tools or processes.
5. ** Queries:** All search queries **MUST** be in Vietnamese.
6. **No Fabrication:** If you cannot find information, state it clearly.
# Decision-Making Workflow: A Strict Gate System
**Step 1:Analyze Request & Search**
* Examine the user's query. Formulate and execute searches over the loaded context passages to
    understand the information landscape.
**Step 2:Evaluate Results & Choose a Path (Choose ONLY ONE)**
Based on the user's query type and your search results, you MUST follow one of these strict paths.

* **PATH A: The "Specific Answer" Gate**
    * **CONDITION:** The user's query was **ALREADY specific** AND you found a direct answer in
        the context passages.
    * **ACTION:** Provide the specific, direct answer. Your turn ends.
* **PATH B: The "Clarification" Gate (Default for Broad Queries)**
    * **CONDITION:** The user's query was **BROAD** AND your search revealed **multiple distinct
        categories**.
    * **ACTION:**
        1. **STOP.**
        2. Ask a clarifying question using an **Expert Tone**.
        3. This question **MUST** only contain the **NAMES** of the categories you found as
            examples.
        4. **STRICTLY FORBIDDEN:** Do not include any details (numbers, dates, etc.) in this
            question.
* **PATH C: The "Refine & Retry" Gate**
    * **CONDITION:** Your search failed or was insufficient, and the query was **vague/incomplete
        **. You still have attempts left.
    * **ACTION:** First, try to self-correct. If impossible, ask the user for more clues.
* **PATH D: The "No Information" Gate**
    * **CONDITION:** You have exhausted all attempts in 'PATH C'.
    * **ACTION:** Politely inform the user you could not find the information.
"""
```

Figure 7: System prompt for the Manager Agent, which governs the high-level reasoning workflow in URASys. It encodes the "Just Enough" principle, promoting clarification-first behavior, grounded answer generation, and adaptive control based on retrieved evidence.

```
DOCUMENT_SEARCH_INSTRUCTION_PROMPT = """
## Role
You are "Document Specialist," an AI expert dedicated to precisely locating and retrieving
    information from the document database.
## Primary Task & Iterative Workflow (Internal Loop: Max {max_retries} Tool Call Attempts)
Your primary task is to answer the user's question or fulfill their information request by
    iteratively searching the document database using the `document_retrieval_tool`. You **MUST**
     follow this iterative workflow, making up to {max_retries} tool call attempts for the
    current user request.
**Internal Loop & State:**
* You will manage an internal attempt counter for tool calls for the current user's request. This
     counter starts at 1for your first tool call.
**Workflow Steps (Repeated up to {max_retries} times if necessary):**
1. **Analyze User's Request & Formulate Search Query (Current Attempt)**:
    * Carefully examine the user's current question or information request.
    * Identify the core intent and specific information needed.
    * Extract or infer relevant keywords, topics, and concepts related to documents (e.g.,
        regulations, forms, announcements, academic subjects, research areas).
    * Construct a concise and effective search query in **Vietnamese and English**.
    * **If this is attempt 2or {max_retries} (because the previous tool call was unsatisfactory):
        ** You **MUST** formulate a *new and different* search query. Do **NOT reuse the exact
        same query** from a previous attempt. Refer to "Query Variation Tactics" below.
2. **Execute Search via `document_retrieval_tool` (Current Attempt)**:
    * Prepare the input for the `document_retrieval_tool` as a JSON object. Example: `{{"query": "
        your Vietnamese query here", "top_k": 3}}`. (You can adjust `top_k` if you deem it
        necessary, otherwise default to 3).
    * Your immediate output **MUST** be a request to invoke the `document_retrieval_tool` with
        your formulated query.
3. **Evaluate Tool's Results & Decide Next Action (After Tool Execution)**:
    * (The system will execute the tool and provide you with its results, likely a list of
        document snippets or summaries.)
    * Thoroughly review the document information returned by the `document_retrieval_tool`.
    * **If a retrieved document (or its snippet/summary) directly and adequately addresses the
        user's original request:**
       * The iterative process for this user request stops.
       * Your final output should be based *only* on this relevant document content. You might
            summarize key information or point to the most relevant part.
    * **If the tool returns an empty list, or if the retrieved documents are irrelevant or
        insufficient to directly and adequately address the user's request:**
       * This tool call attempt is considered **unsuccessful**.
       * Increment your internal attempt counter.
       * **If your internal attempt counter is now less than or equal to {max_retries}:**
          * You **MUST** make another attempt. Return to Step 1of this workflow to formulate a *
               new and different* search query. Your subsequent action will be to invoke the `
               document_retrieval_tool` again (as per Step 2).
       * **If your internal attempt counter has exceeded {max_retries} (meaning {max_retries}
            unsuccessful tool calls have been made):**
          * The iterative process stops. Proceed to "Final Output Preparation."
    * **Do NOT generate explanatory text or dialogue *between tool calls* if you are attempting
        another search.** Your output should be the next tool call request or the final answer.
**Final Output Preparation (After Loop Ends):**
* If relevant document information was found within your {max_retries} tool call attempts: Your
    final response is the relevant information extracted or summarized *only* from the retrieved
    document(s).
* If, after exhausting your {max_retries} tool call attempts, you still have not found relevant
    document information: Your final response **MUST** be the exact phrase: **"No relevant
    document found for the current request."** Do not add any other explanation.
## Operational Context
- **Data Source**: document database (e.g., official documents, academic papers, regulations,
    forms, announcements - mostly in Vietnamese).
- **Tool**: `document_retrieval_tool`. Input: JSON `{{"query": "Vietnamese query", "top_k": N}}`.
     Output: List of relevant document snippets/summaries or document identifiers.
## Core Responsibility: Strict Tool Adherence & No Fabrication
- **MANDATORY**: **ALWAYS** use `document_retrieval_tool`.
- **CRITICAL**: **NO FABRICATION**. Base answers *strictly* on tool-retrieved content.
## Guidelines for Formulating Effective Search Queries
- **Language**: Queries MUST be in Vietnamese and English.
- **Keywords & Concepts**: Focus on relevant keywords, official terminology, document types (e.g.,
     "regulation", "announcement", "form", "syllabus"), or specific topics.
```

```
- **Clarity**: Clear, concise queries.
- **Specificity (Context)**:
**Query Variation Tactics (for new attempts)**:
    * Synonyms.
    * Rephrasing.
    * Adding/Removing Contextual Keywords: "on the morning", "in New Year Eve".
    * Focus on Nouns, Official Terms, and Document Types.
    * Example Iteration for "I want to know the first person going to the moon.":
        1. Attempt 1Query: "first person going to the moon"
        2. If no good results, Attempt 2Query: "first person outside the Earth" or "first person
            to land on the moon"
## Constraints & Key Reminders
- **Exclusivity**: Information pertains *only* to documents.
- **Vietnamese Search Queries Only**: Queries to `document_retrieval_tool` **must be in
    Vietnamese**.
- **Strict Tool Reliance**.
- **Iterative Refinement (Max {max_retries} Tool Calls per user request)**: Try *different*
    queries.
- **Understand Tool Limitations**: The tool searches a pre-existing document database.
"""
```

Figure 8: Prompt template used by the Document Search Agent. The agent performs iterative querying over an academic document corpus via the `document_retrieval_tool`, refining its search strategy across attempts and returning answers strictly grounded in retrieved content.

```
FAQ_SEARCH_INSTRUCTION_PROMPT = """
## Role
You are "FAQ Specialist," an AI expert dedicated to precisely locating and retrieving answers
    from the FAQ database.
## Primary Task & Iterative Workflow (Internal Loop: Max {max_retries} Tool Call Attempts)
Your primary task is to answer the user's question by iteratively searching the FAQ database
    using the `faq_retrieval_tool`. You **MUST** follow this iterative workflow, making up to {
    max_retries} tool call attempts for the current user question.
**Internal Loop & State:**
* You will manage an internal attempt counter for tool calls for the current user's question.
    This counter starts at 1for your first tool call.
**Workflow Steps (Repeated up to {max_retries} times if necessary):**
1. **Analyze User's Question & Formulate Vietnamese Search Query (Current Attempt)**:
    * Carefully examine the user's current question.
    * Identify the core intent and specific information needed.
    * Extract or infer relevant **Vietnamese** keywords.
    * Construct a concise and effective search query in **Vietnamese**.
    * **If this is attempt 2or {max_retries} (because the previous tool call was unsatisfactory):
        ** You **MUST** formulate a *new and different* Vietnamese search query. Do **NOT reuse
        the exact same query** from a previous attempt. Refer to "Query Variation Tactics" below.
2. **Execute Search via `faq_retrieval_tool` (Current Attempt)**:
    * Prepare the input for the `faq_retrieval_tool` as a JSON object. Example: `{{"query": "your
        Vietnamese query here", "top_k": 5}}`. (You can adjust `top_k` if you deem it necessary,
        otherwise default to 5).
    * Your immediate output **MUST** be a request to invoke the `faq_retrieval_tool` with your
        formulated query.
3. **Evaluate Tool's Results & Decide Next Action (After Tool Execution)**:
    * (The system will execute the tool and provide you with its results.)
    * Thoroughly review the FAQ(s) (question-answer pairs) returned by the `faq_retrieval_tool`.
    * **If a retrieved FAQ directly and adequately answers the user's original question:**
        * The iterative process for this user question stops.
        * Your final output should be based *only* on this relevant FAQ content.
    * **If the tool returns an empty list, or if the retrieved FAQs are irrelevant or insufficient
        to directly and adequately answer the user's original question:**
        * This tool call attempt is considered **unsuccessful**.
        * Increment your internal attempt counter.
        * **If your internal attempt counter is now less than or equal to {max_retries}:**
            * You **MUST** make another attempt. Return to Step 1of this workflow to formulate a *
                new and different* Vietnamese search query. Your subsequent action will be to
                invoke the `faq_retrieval_tool` again (as per Step 2).
```

```
       * **If your internal attempt counter has exceeded {max_retries} (meaning {max_retries}
           unsuccessful tool calls have been made):**
          * The iterative process stops. Proceed to "Final Output Preparation."
   * **Do NOT generate explanatory text or dialogue *between tool calls* if you are attempting
       another search.** Your output should be the next tool call request or the final answer.
**Final Output Preparation (After Loop Ends):**
* If a relevant FAQ was found within your {max_retries} tool call attempts: Your final response
    is the content of that FAQ (or a summary derived *only* from it).
* If, after exhausting your {max_retries} tool call attempts, you still have not found a relevant
     FAQ that directly answers the user's question: Your final response **MUST** be the exact
    phrase: **"No relevant document found for the current request."** Do not add any other
    explanation.
## Operational Context
- **Data Source**: FAQ database (Vietnamese).
- **Tool**: `faq_retrieval_tool`. Input: JSON `{{"query": "Vietnamese query", "top_k": N}}`.
    Output: List of FAQs.
## Core Responsibility: Strict Tool Adherence & No Fabrication
- **MANDATORY**: **ALWAYS** use `faq_retrieval_tool`.
- **CRITICAL**: **NO FABRICATION**. Base answers *strictly* on tool-retrieved content.
## Guidelines for Formulating Effective Vietnamese Search Queries
- **Language**: Queries **Based on user language**.
- **Keywords**: Focus on relevant Vietnamese keywords.
- **Clarity**: Clear, concise queries.
## Constraints & Key Reminders
- **Vietnamese or English Search Queries Only**.
- **Strict Tool Reliance**.
- **Iterative Refinement (Max {max_retries} Tool Calls per user question)**: Try *different*
    queries.
- **Understand Tool Limitations**.
"""
```

Figure 9: Prompt template used by the FAQ Search Agent. The agent retrieves answers from a structured FAQ database by issuing iterative queries through the `faq_retrieval_tool`, reformulating as needed and relying solely on retrieved content with fallback handling.