End-Term Report for Agentic Retrieval-Augmented Generation (RAG) System

Team 57

1. INTRODUCTION

The project focuses on developing an advanced Agentic Retrieval-Augmented Generation (RAG) system that seamlessly integrates retrieval and generative AI capabilities. The primary objective is to create a dynamic system capable of fetching relevant information from external sources in real-time and combining it with generative AI to produce precise, contextually relevant responses. The system begins by evaluating input queries for safety and formal structure compliance through robust input guardrails. Queries are then classified as formal or informal. Formal queries trigger a specialized retrieval pipeline, fetching top-k documents from Pathway's VectorStore or initiating a web search if necessary. Informal queries, on the other hand, are processed directly through conversational AI mechanisms for swift and relevant responses. By combining retrieval-driven accuracy with the creative flexibility of generative AI, this framework ensures responses are both factually robust and contextually tailored. This modular and adaptive architecture is designed to address diverse use cases, offering improved decision-making and user experience through real-time indexing, seamless query classification, and continuous learning from user feedback.

2. UNIQUENESS

Our Agentic RAG system distinguishes itself through innovative features that redefine query handling and response generation. Unlike traditional single-agent systems, the Leader-Analyst framework enables dynamic task division and collaboration, ensuring precision and scalability. The integration of Pathway's VectorStore with real-time indexing and external web searches supports adaptive data retrieval, providing accurate and up-to-date information from both structured and unstructured sources. Additionally, the system incorporates a Human-in-the-Loop mechanism to iteratively refine outputs based on user feedback, making it highly user-centric. One unique aspect of our approach is that we exclusively utilized Pathway without LangGraph in our final submission, demonstrating a streamlined yet powerful architecture. With advanced safety protocols, including Guardrails.ai, and the ability to distinguish and optimize processing for formal and informal queries, our solution sets a new standard for precision, adaptability, and ethical AI interactions in the RAG ecosystem.

3. USE CASE SELECTION AND NOVELTY

3.1 Selected Use Case: Multi-Agent Collaboration System for Financial Investment Research

Our Agentic Retrieval-Augmented Generation (RAG) system is designed to process both formal and informal text-based queries efficiently, offering accurate, context-aware responses through a multi-agent framework. The solution integrates advanced retrieval, dynamic task management, and user feedback for continuous improvement.

3.2 Uniqueness of the Use Case

- **Dual Query Processing:** The system distinguishes between formal and informal queries. Informal queries are handled directly by the LLM for quick responses, while formal queries undergo structured processing using Pathway's VectorStore and external web searches, ensuring comprehensive and relevant answers.

- **Hierarchical Agent Framework:** A Leader-Analyst model orchestrates the query resolution process. The Leader splits complex tasks and synthesizes responses from Analysts, ensuring quality and refining outputs dynamically, surpassing traditional single-agent approaches.

- **Adaptive Data Retrieval:** By integrating structured data from Pathway's VectorStore and unstructured, realtime data from web searches, the system ensures accurate and up-to-date information retrieval, enhancing the depth of responses.
- **Human-in-the-Loop Feedback:** User feedback is incorporated to refine responses iteratively. This feedback-driven loop allows for additional context gathering and reprocessing, ensuring improved accuracy and relevance over time.
- **Safety and Quality Assurance:** Input Guardrails and final output checks ensure ethical query handling and grammatical accuracy. The system prevents harmful content and delivers polished, coherent answers tailored to user needs.

4. SOLUTION OVERVIEW

This document outlines the design of the Agentic RAG (Retrieval-Augmented Generation) system, focusing on its core concepts, assumptions, and approach to delivering accurate, efficient, and ethical query processing.

4.1 Assumptions

- **Text-Only Queries:**The system processes only text-based inputs. Non-textual data must be converted into text before submission.
- **No Hyperlinks in Input:**Queries will not contain hyperlinks or URLs, simplifying query processing and eliminating the need for web scraping.

4.2 Key Components

- **Input Guardrail:**Filters harmful content from the query before processing, using Guardrails.ai and custom techniques to ensure safety.
- **Source Planner:**Determines the query type and retrieves relevant documents from Pathway's VectorStore or triggers a web search if necessary.
- **Leader:**Orchestrates the resolution process, splitting complex queries into subtasks and synthesizing responses from analysts, refining them if needed.
- **Analyst A and Analyst B:**Retrieve relevant documents from Pathway's VectorStore and generate responses based on semantic similarity.
- **Web Search:**If VectorStore fails, a web search is triggered to gather real-time data, which is processed and refined by the Leader.
- **Human In The Loop(HIL):**Collects user feedback to refine responses, reprocessing the query for greater accuracy if needed.
- **Final Output Generation:**The Leader ensures the final response meets quality standards, reassigning tasks as necessary to improve the result.
- **Pathway Adaptive RAG Question Answering System:** Adaptive RAG dynamically adjusts the number of supporting documents in prompts based on question difficulty and model feedback. Starting with a minimal context, it expands geometrically (e.g., doubling documents) if the model cannot confidently answer, ensuring cost-efficiency and low latency. Overlapping prompts retain previously used relevant documents, maintaining accuracy while progressively refining context for harder questions.

4.3 Approach

The system integrates modular components, cutting-edge technologies, and human oversight to efficiently and ethically process queries.

- **Safety and Ethics:** Guardrails.ai ensures that both input and output are screened for harmful content, preventing biased or inappropriate responses.
- **Efficient Data Retrieval:** The system uses Pathway's VectorStore for relevant document retrieval, and if needed, a web search is triggered to supplement data for real-time answers.
- **Leader Agent Framework:** Complex queries are split into subtasks, which are assigned to analysts. The Leader synthesizes responses and refines them for accuracy.
- **User Feedback Loop:** Feedback is gathered to reprocess and improve the response, ensuring higher accuracy.
- **Pathway Integration:** Pathway's VectorClient and parsing algorithms enhance document retrieval and data structuring, improving response quality.

This approach combines efficient retrieval, continuous refinement, and ethical safeguards to ensure accurate and high-quality query resolution.

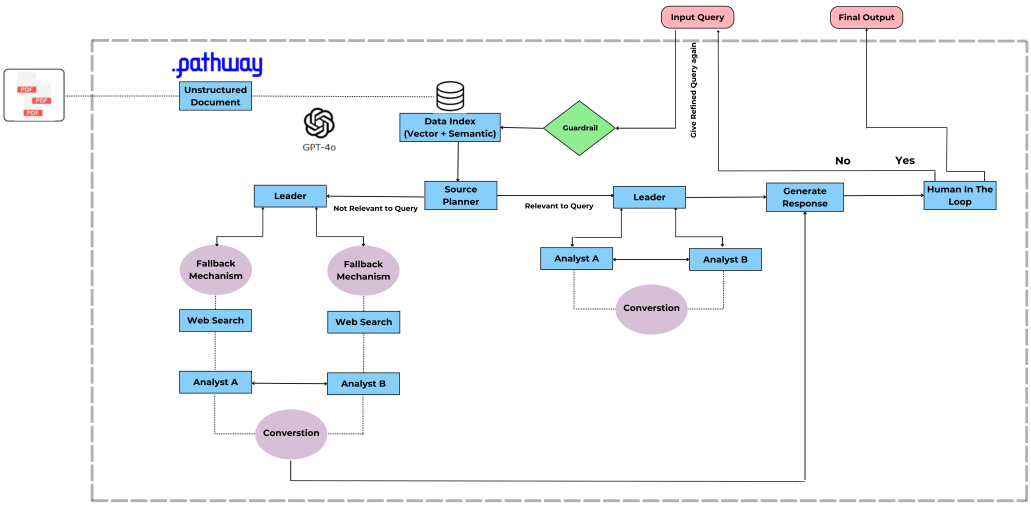

Figure 1: Workflow Diagram for Intent Detection and Processing

5. SYSTEM ARCHITECTURE

5.1 Input Guardrail

- **Data Flow:**The query undergoes a safety check to ensure it's free from harmful or inappropriate content. If the query is deemed safe, it moves to the next step; otherwise, it is blocked with an informative response.
- **Technical Implementation:** The system uses Guardrails.ai and custom prompting techniques to ensure the query complies with safety standards before processing further.

5.2 Source Planner

- **Data Flow:**The Source Planner evaluates the query type to decide the next course of action.If the query is document-related, it retrieves relevant documents from the Pathway VectorStore. If the query requires real-time information, it flags the need for a web search. The query then moves to the Leader, which executes the planned action and delegates tasks to Analysts.
- **Technical Implementation:** TThe Source Planner uses Pathway's VectorClient to fetch top-k documents, ensuring relevance. If insufficient results are found, it triggers a web search workflow to fetch real-time data.

5.3 Leader

- **Data Flow:** Planner: For document-based queries: It processes the documents and divides the task into subtasks (one or two, depending on complexity). Analysts then handle these subtasks to generate responses. For web search-based queries: The Leader executes the web search using external APIs (e.g., SERP API) and processes the results. Subtasks are created and delegated to Analysts for further refinement. Once Analysts provide their responses, the Leader synthesizes them into a coherent final output. If the response is insufficient, the task is re-divided and re-assigned.
- **Technical Implementation:** Implemented with custom classes and functions to delegate tasks, evaluate responses, and manage iterative refinements for improved accuracy.

5.4 Analyst A and Analyst B

- **Data Flow:** After receiving subtasks from the Leader, each analyst retrieves top-k documents and generates responses based on the retrieved information.
- **Technical Implementation:** Analyst A and Analyst B use Pathway's VectorStore to ensure relevant and accurate responses for their respective subtasks.

5.5 Web Search

- **Data Flow:** If the VectorStore fails to provide relevant documents, a web search is triggered using an external API (e.g., SERP API). The results are processed into contexta and contextb, which are passed to the Leader for further refinement.

- **Technical Implementation:** The system integrates with a web search API to gather real-time data, which is then processed and passed to the Leader for refinement.

5.6 Human-in-the-Loop (HIL)

- **Data Flow:** After generating an initial response, the system asks for user feedback. If the user is dissatisfied, additional context is requested, and the query is reprocessed for enhanced accuracy. If necessary, the process repeats with further refinements.
- **Technical Implementation:** User feedback triggers a new round of processing, where the updated query is re-routed through input guardrails and formality checks for refined output generation.

5.7 Output Generation

- **Data Flow:** Once the Leader has processed the query and context, the final response is generated. If the user is satisfied, the process concludes. Otherwise, further iterations may be carried out based on feedback.
- **Technical Implementation:** The Leader evaluates the output quality, reassigning tasks if needed, to ensure high-quality, relevant responses.

5.8 Pathway Integration

- **Pathway Vector Client:** Enables top-k document retrieval from the Pathway VectorStore, ensuring the system fetches relevant data based on semantic similarity for both question answering and parsing.
- **Question-Answering System:** Combines RAG (Retrieval-Augmented Generation) with adaptive capabilities, leveraging Pathway's VectorClient to ensure accurate data retrieval and improved answers.
- **Parser:** Uses Pathway's LLM-driven parsing algorithms to process unstructured data, efficiently extracting structured insights from complex datasets. This enhances data handling and overall response quality.

6. RESULTS AND METRICS

To evaluate the system's performance, three distinct approaches were implemented and tested, leveraging Pathway, LangGraph, and AutoGen frameworks. These implementations were assessed for hallucination detection and answer relevance, metrics critical for ensuring reliability and usability. Each approach brought unique features and trade-offs, providing valuable insights into their comparative performance.

6.1 Approaches and Features:

- The first approach, utilizing only Pathway components, showcased the power of Pathway's dynamic processing and vector store for efficient data handling. This streamlined design offered fast query responses and robust data management, making it an ideal choice for scenarios prioritizing speed and simplicity. This approach consistently delivered reliable and contextually relevant answers, demonstrating its effectiveness as a standalone solution.
- second approach integrated Pathway's vector store with LangGraph, enabling leader-analyst agents implemented in LangGraph. This added modularity and enhanced query-specific accuracy through hierarchical workflows, striking a balance between efficiency and precision.
- The third approach extended the second by implementing leader-analyst agents in AutoGen and integrating them with LangGraph. This design introduced advanced agent logic for better contextual understanding but required greater computational resources.

6.2 Findings and Metrics:

Performance metrics were evaluated using the Opik library, with an LLM-based judge assessing hallucination and answer relevance on 30 queries of the provided Alphabet Inc. dataset. The opik library leverages Large Language Models as judge evaluation metrics to assess the quality of generated text. It specifically focuses on evaluating two key aspects: hallucination, ensuring the generated answers are factually accurate, and answer relevance, ensuring the responses are contextually appropriate. By using LLMs to automate and scale the evaluation of these factors, opik enables consistent and effective monitoring of LLM performance in tasks like question answering and dialogue systems. Key findings include:

- **Hallucination Detection:** All three approaches achieved a hallucination score of 0, demonstrating full reliance on validated data.

Metric	Pathway	Langraph	Autogen
Average Answer Relevance Score	0.84	0.83	0.775

Figure 2: Average Relevance Scores

- **Answer Relevance:** The **Pathway only** approach achieved an **average answer relevance score of 0.84**, reflecting precise but less context-aware responses.**The Pathway + LangGraph approach** achieved **a score of 0.83.** The **Pathway + LangGraph + AutoGen** implementation **scored 0.775.**

These results underscore the high performance and reliability of all three implementations, with the Pathway-only solution excelling in scenarios where simplicity and speed are paramount. All approaches maintained zero hallucinations and achieved average relevance scores above 0.80, aligning with the intended use case. The results are stored in Excel files which can be found in the GitHub repository.

6.3 Efficiency

To explore efficiency improvements, an experimental approach utilizing "Not Diamond" was tested. "Not Diamond" dynamically selects the most efficient model for a given query based on complexity, optimizing token usage and reducing computational overhead. While it demonstrated potential for significant efficiency gains, further calibration was required for seamless integration into the system.

- **Findings:** "Not Diamond" optimized resource allocation by tailoring model selection to query needs, showing promise for reducing costs and improving processing times.

- **System Efficiency:** Both deployed systems demonstrated consistent query processing times and maintained resource efficiency through dynamic API fallback mechanisms.

6.4 Results and Insights:

- **Performance:** Pathway-LangChain CRAG showed superior handling of complex queries and better adaptation to user feedback.

- **Error Handling:** Implementation of comprehensive error detection and correction mechanisms improved response quality for ambiguous queries.

- **System Flexibility:** The hybrid approach demonstrated better adaptability to various query types while maintaining accuracy.

7. CHALLENGES FACED AND SOLUTIONS

7.1 Overcoming Summarization Challenges with GoogleSerperAPIWrapper:

can be tricky because web search results often include unnecessary content like HTML tags, image URLs, and links to menus or home pages. These extra details create noise and make it harder to produce clear and concise summaries. Removing this irrelevant information can take a lot of time and effort and sometimes the important informations also get removed from the summary. To fix this, we use the GoogleSerperAPIWrapper, a built-in search tool that helps filter out unnecessary content. It focuses on getting clean and relevant data, making the summarization process easier and more accurate.

7.2 Tackling Ambiguous Document Issues with isFormalcheck Agent:

Detecting whether a document is truly ambiguous can be challenging when using CRAG. This happens because the process relies heavily on hyperparameters, making it inconsistent and harder to identify ambiguous information accurately. This can sometimes lead to incorrect results. To solve this problem, we created the isFormalcheck agent, which checks whether a query is related to finance or not. This helps in managing ambiguity effectively and improves the accuracy of the results.

7.3 Addressing the Guardrail System's Limitations:

The Guardrail system sometimes has difficulty detecting harmful content within PDF documents. In some cases, harmful data may already be present in the PDF, and Guardrail may not catch it, which could lead to unsafe outputs being generated. To solve this issue, we used pretrained Guardrail models that help check whether a query is appropriate or not, ensuring better content moderation.

Query	Pathway		Langgraph		Autogen	
	Hallucination_Score	Relevance_Score	Hallucination_Score	Relevance_Score	Hallucination_Score	Relevance_Score
How has Alphab	0	0.95	0	0.85	0	0.8
How do Google'	0	0.95	0	0.85	0	0.85

Figure 3: Top two query's result

8. RESILIENCE TO ERROR HANDLING:

In financial research, accuracy and reliability are paramount. To ensure resilient and reliable operation even in the face of service failures, the system employs advanced strategies to dynamically switch between multiple APIs. By leveraging an exponential backoff approach, it minimizes downtime and ensures consistent performance under varying conditions.

8.1 System Design and Implementation:

- The core mechanism of our Web Search Tool revolves around a sophisticated fallback system that seamlessly transitions between multiple search APIs, primarily Google Custom Search and SERP API. This approach addresses the critical challenge of maintaining consistent data access in scenarios where individual APIs might experience temporary failures or performance issues.

- The implementation features a sophisticated error handling strategy centered on exponential backoff with jitter. This technique introduces progressive delay intervals between retry attempts, with added randomization to prevent network congestion and synchronized request overloads. By implementing a maximum retry limit and gradually increasing delay times, the tool ensures efficient resource utilization while maintaining the potential to retrieve critical financial research data.

- The tool's design emphasizes flexibility and resilience. When initiating a search, it systematically attempts to retrieve results through different APIs, logging each attempt and actively tracking the active API. This approach provides comprehensive traceability and enables detailed performance monitoring. If one API fails, the system automatically switches to an alternative, ensuring uninterrupted access to research information.

8.2 Flexibility and Extensibility:

The system's architecture is designed to be modular and extensible, making it easy to integrate additional APIs as needed. For example, incorporating a service like Tavily Search would involve adding a new search function and including it in the list of fallback options. This modularity ensures the system can adapt to evolving requirements or leverage newer, more efficient APIs as they become available, enhancing its longevity and applicability.

8.3 Conclusion:

The system exemplifies robust error handling through its exponential backoff strategy and dynamic service switching. These features not only enhance reliability but also support the system's future scalability and adaptability. By incorporating modular design principles, the system ensures resilience against service failures, making it a dependable and forward-looking solution for financial investment research tasks.

9. USER INTERFACE OVERVIEW

Our user interface is designed to deliver a simple, intuitive, and transparent experience, enabling users to interact effortlessly while gaining meaningful insights. Key features include:

9.1 Landing Page: Easy & Efficient

- **PDF Upload Made Simple:** The landing page includes a straightforward upload feature, allowing users to quickly add PDFs without unnecessary complexity. Just select your file, and you're ready to go!

- **Customizable Model Responses:** A temperature adjustment slider empowers users to fine-tune the tone and creativity of the language model (LLM) responses. Whether you prefer concise, fact-based outputs or more imaginative, creative answers, you have full control.

9.2 Enhanced Transparency

- **Query Processing Time Display:** The system provides a clear processing time display for each query, ensuring users are always informed about system performance with real-time feedback.

- **"Not Diamond" Feature:** Token usage is displayed, offering users transparency regarding resource consumption and helping them manage system usage effectively.

9.3 Engaging Visualization & Customization

The user interface supports dynamic visualizations, enhancing user interaction and making the system's outputs more accessible and engaging.

10. RESPONSIBLE AI PRACTICES

10.1 Implemented Guardrails

To ensure Responsible AI practices and maintain high ethical standards, accuracy, and safety in our system, we have integrated comprehensive guardrails powered by Guardrails AI and advanced contextual checks. These mechanisms are crucial for maintaining the integrity of our system's operations and ensuring safe interactions at every stage.

- **Input Guardrails:** The system utilizes Guardrails AI to inspect user queries before they enter the processing pipeline. Multiple validation techniques are employed to detect harmful, inappropriate, or nonsensical content, effectively preventing malicious inputs, data leakage, and potential misuse. When a query fails to meet safety standards, it is blocked, and the user receives a clear explanation of the issue. This proactive approach ensures that only safe and relevant data enters the system.

- **Output Guardrails:** After generating responses, Guardrails AI further refines the content through a series of context-sensitive checks to ensure the generated output aligns with ethical guidelines and company policies. These checks address potential risks such as biased language, offensive remarks, or misleading information. By incorporating these safeguards, we ensure that the responses are not only accurate but also fair, transparent, and aligned with responsible AI principles.

11. LESSONS LEARNED: INSIGHTS AND IMPROVEMENTS

11.1 Key Insights and Challenges:

- **Multi-Agent RAG Architecture:** Implementing a multi-agent framework (Source Planner and Leader-Analyst) significantly improved task distribution and query handling. The Leader-Analyst coordination allowed for precise, context-driven responses, enhancing accuracy and efficiency.

- **Pathway Library Utilization:** Leveraging Pathway's capabilities optimized data retrieval and parsing, especially in handling semantic search and complex workflows. The experience underscored the importance of selecting specialized libraries for processing both structured and unstructured data.

- **Dynamic Data Handling with Real-Time Indexing:** The integration of real-time indexing through Pathway ensured up-to-date retrieval of relevant documents, enabling seamless access to both static and dynamic data. Combining this with web search expanded the system's ability to provide timely and contextually rich responses

11.2 Improvements Made:

- **Enhanced Data Retrieval:** Incorporating web scraping tools improved real-time data access, broadening information availability for better decision-making and more comprehensive responses.

- **Code and Workflow Optimization:** Refinements in the code reduced latency and improved overall system responsiveness, making the RAG pipeline more efficient and scalable.

- **Strengthened Input Validation:** Enhanced guardrails and input checks reduced risks associated with inappropriate queries, ensuring safe, ethical interactions.

11.3 Potential Enhancements or Extensions:

- **Conversational Memory:** Introducing conversational memory would allow for context continuity across sessions, enhancing personalization and user engagement.

- **Web-Interface:** In our web interface, we can add the "Not Diamond" feature to display token usage, providing users with transparency about resource consumption. We can also integrate a chain-of-thought visualization, allowing users to see the AI's reasoning process in real-time, which enhances user engagement and understanding. Additionally, we plan to offer advanced customization options, such as the ability to fine-tune settings for specific types of queries, giving users more control and flexibility to personalize their experience and improve system performance.

- **Advanced Pathway Features:** Further leveraging Pathway's advanced tools, including distributed processing and handling larger datasets, could unlock performance improvements in high-demand scenarios.

- **Adaptive Query Handling:** Implementing a more flexible query classification system could optimize how formal and informal queries are routed, improving system efficiency and accuracy.

12. CONCLUSION

- Our Agentic Retrieval-Augmented Generation (RAG) system provides a robust and scalable solution for handling formal and informal queries. By leveraging a multi-agent framework, it ensures dynamic task management through the Leader-Analyst model, enabling efficient query resolution with high accuracy. The system's ability to integrate structured data from Pathway's VectorStore and real-time data through web searches allows for comprehensive and timely responses.

- Key safeguards, such as input guardrails and user feedback loops, ensure ethical query processing and continuous improvement. This modular, feedback-driven approach not only enhances response quality but also adapts to evolving user needs, making the system versatile and reliable for diverse use cases.

- The thoughtful integration of advanced retrieval techniques, human oversight, and safety protocols positions this system as a cutting-edge tool in text-based query processing, ensuring efficiency, accuracy, and ethical standards.

12.1 Sources that are crucial:

- AutoGEN

- GuardRails AI

- CRAG

13. APPENDIX

13.1 Expanded Approach with LangGraph Integration:

The Agentic RAG system is primarily designed to leverage the full capabilities of Pathway, ensuring efficient and accurate retrieval-augmented generation. In addition to the main implementation, we have developed an alternative approach that integrates LangGraph to enhance modularity and adaptability. This hybrid setup demonstrates the system's flexibility in interfacing with other frameworks, expanding its capabilities for specific use cases.

13.2 Pathway and Langgraph Coherence:

In the LangGraph-integrated approach, Pathway and LangGraph collaborate seamlessly to optimize task execution:

- **Pathway's Role:** Remains the core component for efficient top-k document retrieval, semantic matching, and contextual data enrichment. Ensures the foundational retrieval pipeline functions at high efficiency, enabling accurate and relevant data access.

- **Langgraph's Role:** Introduces a structured framework for task delegation, splitting, managing, and refining sub tasks using its agent-based architecture. Enhances workflow flexibility by encapsulating individual components (e.g., Input Guardrail, Source Planner) as nodes, allowing dynamic interaction and scalability.

14. SOLUTION OVERVIEW

This document outlines the design of the Agentic RAG (Retrieval-Augmented Generation) system, focusing on its assumptions, key components, and approach to handling user queries with efficiency, accuracy, and safety.

14.1 Assumptions

1. **Text-Only Queries:** The system processes only text-based inputs. Non-textual elements, like images and tables, must be converted into text before submission.

2. **No Hyperlinks in Input:** Queries will not contain hyperlinks or URLs, simplifying query handling by removing the need for web scraping or link resolution.

14.2 Key Components

1. **Input Guardrail:** Ensures query safety by filtering harmful content using Guardrails.ai and prompting techniques, blocking unsafe queries and allowing safe ones to proceed.

2. **Formality Check:** Classifies queries as formal or informal. Informal queries are handled directly by the LLM, while formal ones are routed to the Source Planner for in-depth processing.

3. **Source Planner:** Retrieves top-k relevant documents from Pathway's VectorStore. If documents are insufficient, a fallback web search is initiated to gather additional information.

4. **Leader:** Manages the query by splitting it into subtasks for analysts. It synthesizes their responses, ensuring coherence and accuracy. Tasks are reassigned if further refinement is needed.

5. **Analyst A & Analyst B:** Analysts handle subtasks by retrieving documents from Pathway's VectorStore and generating accurate, relevant responses based on semantic similarity.

6. **Generation:** The system validates the final response for accuracy and grammatical correctness, ensuring the output meets quality standards before delivering it to the user.

7. **Web Search (Fallback):** If the VectorStore fails, a web search is triggered using external APIs to gather real-time data, which is then processed and passed to the Leader for refinement.

8. **Human-in-the-Loop (HIL):** User feedback is collected after the initial response. If unsatisfied, users can provide additional context, prompting a reprocessing of the query for a refined output.

14.3 Approach

The Agentic RAG system leverages a modular design, integrating advanced technologies and human oversight to ensure ethical, efficient, and accurate query handling.

1. **Safety and Ethics:** Guardrails.ai ensures all inputs are screened for harmful content, maintaining ethical standards in both query handling and response generation.

2. **Efficient Retrieval:** The system prioritizes data retrieval from Pathway's VectorStore. If necessary, a web search supplements the data to ensure up-to-date answers.

3. **Leader-Agent Framework:** The Leader divides complex tasks among analysts, synthesizes their responses, and refines the output for accuracy and coherence.

4. **User Feedback Loop:** User feedback helps refine responses through iterative reprocessing, enhancing accuracy and relevance.

5. **Pathway Integration:** Pathway's VectorClient ensures efficient document retrieval, while its parsing algorithms extract insights from unstructured data, improving response quality.

This approach optimizes query resolution through efficient information retrieval, continuous refinement, and ethical safeguards, ensuring high-quality results.

15. SYSTEM ARCHITECTURE

15.1 Input Guardrail

Data Flow: The query undergoes a content safety check. If it is safe, it proceeds to the next step; otherwise, processing is halted, and the user receives an appropriate response.
Technical Implementation: We use Guardrails.ai, a pre-trained content safety library, alongside a prompting guardrail technique for added verification.

15.2 Formality Check

Data Flow: The query is classified as formal or informal. Informal queries are handled directly by the LLM, while formal queries are passed to the Source Planner for detailed processing.
Technical Implementation: A custom-built formality-checking module routes formal queries to the retrieval pipeline and informal ones to the LLM for quick response generation.

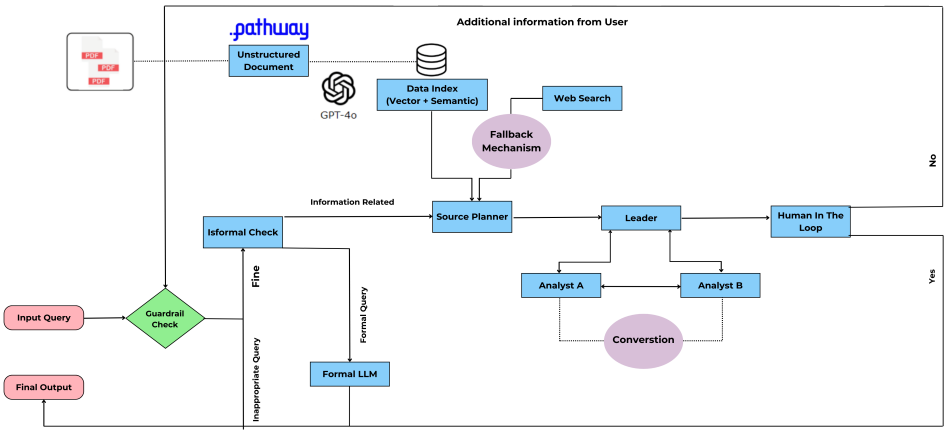

Figure 4: Workflow Diagram for Intent Detection and Processing

15.3 Source Planner

Data Flow: The Source Planner retrieves the top-k relevant documents from Pathway's VectorStore based on the query. If no relevant documents are found, it triggers a web search. Pathway's VectorClient ensures that the retrieval is efficient and relevant.
Technical Implementation: Pathway's VectorStore and Client are used for top-k document retrieval. If the documents don't meet the relevance criteria, the system initiates a web search to gather additional information.

15.4 Leader

Data Flow: After retrieving relevant documents, the Leader splits the query into subtasks (one for simple queries, two for complex queries). It synthesizes the responses from Analyst A and Analyst B, ensuring the final answer is coherent and accurate. If needed, the task is divided further for refinement.
Technical Implementation: The Leader uses a custom class and functions to manage task delegation, document evaluation, and response synthesis. It ensures high-quality output by re-assigning tasks when necessary.

15.5 Analyst A & Analyst B

Data Flow: Analysts retrieve top-k documents from Pathway's VectorStore for their respective subtasks. They generate detailed responses based on the documents.
Technical Implementation: Analysts are agents in LangGraph, leveraging Pathway's VectorStore for document retrieval to ensure the generated responses are relevant to the subtasks.

15.6 Generation

Data Flow: The system generates a final response and performs a validation check for grammatical accuracy and relevance before delivering it to the user.
Technical Implementation: Output guardrails are integrated to perform final checks on the response, ensuring the content is free of errors and meets the required standards.

15.7 Web Search (Fallback)

Data Flow: If the Source Planner fails to find sufficient documents, a web search is initiated using an external API. The retrieved content is processed into context and passed to the Leader for response generation.
Technical Implementation: A backup API, such as Google Custom Search or Bing, is used to perform a real-time web search, ensuring up-to-date and relevant content is retrieved when necessary.

15.8 Human in the Loop (HIL)

Data Flow: After generating the initial response, the system asks the user for feedback. If the user is unsatisfied, they can provide additional context, and the query is reprocessed for enhanced accuracy.
Technical Implementation: The system interacts with the user interface to collect feedback and reroutes the re-

fined query through the system's guardrails and formality checks for further processing.

16. CONCLUSION

While our primary focus has been to build a fully Pathway-powered Agentic RAG system, we have also expanded the framework by integrating Pathway with LangGraph, showcasing the versatility of our approach.

- **Pathway plays a pivotal role in creating a better Agentic RAG by offering:**

  Efficient and accurate document retrieval, ensuring relevance through semantic understanding. Scalability and robustness, enabling seamless handling of large datasets and complex queries. Advanced data parsing and contextual enrichment, improving the precision and depth of generated responses. In our main pipeline, we have solely leveraged Pathway to deliver a high-performance, streamlined RAG system. However, our additional implementation integrating Pathway with LangGraph demonstrates how modularity and adaptability can enhance system capabilities. This dual-pipeline approach ensures we are not only building an efficient solution for today's challenges but also creating a framework that is prepared for future advancements in agentic RAG systems.