National College of Ireland

AskFlow: A Multi-Agent Framework for Intelligent Product Customer Support

MSc Research Project
Master of Science in Artificial Intelligence

Shudong Wang
Student ID: x23115874

School of Computing
National College of Ireland

Supervisor: Arundev Vamadevan

National College of Ireland

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Shudong Wang
Student ID:	x23115874
Programme:	Master of Science in Artificial Intelligence
Year:	2025
Module:	MSc Research Project
Supervisor:	Arundev Vamadevan
Submission Due Date:	August 2025
Project Title:	AskFlow: A Multi-Agent Framework for Intelligent Product Customer Support
Word Count:	5500
Page Count:	18

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Shudong Wang
Date:	14th September 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☒
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☒
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☒

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only
Signature: Programme Coordinator
Date: 14th September 2025
Penalty Applied (if applicable): None

AskFlow: A Multi-Agent Framework for Intelligent Product Customer Support

Shudong Wang
x23115874

Abstract

This thesis presents the design, implementation, and evaluation of AskFlow, a multi-agent framework for intelligent customer support with demonstrated effectiveness in educational domains and extensibility to healthcare applications. The system addresses key challenges in customer service automation by integrating multiple AI providers and implementing domain-specific document processing techniques. Through detailed case studies in educational institutions and development of healthcare evaluation frameworks, this research shows measurable improvements over traditional keyword-based support systems. The modular architecture supports multiple AI providers (OpenAI, Anthropic, Google Gemini, Cohere) and various document processing strategies, offering organizations a practical solution for domain-specific customer support automation.

Keywords: Multi-Agent Systems, Customer Support Automation, Retrieval-Augmented Generation, Multi-Provider AI, Intelligent Information Systems

1 Introduction

1.1 Background and Motivation

Finding specific information on university websites presents significant challenges for students and prospective applicants. Users seeking course prerequisites, application deadlines, or fee information often navigate through numerous web pages, frequently encountering outdated or incomplete information. This inefficiency leads students to contact various departments directly via phone or email, creating operational bottlenecks for administrative staff who must repeatedly address similar inquiries.

Traditional customer support in educational settings relies heavily on human staff and basic search functionality. When students cannot find what they need online, they contact admissions offices, academic departments, or student services. This approach works, but it is expensive and does not scale well. Staff members often provide different answers to similar questions, and response times vary dramatically depending on workload and availability.

Recent developments in artificial intelligence, particularly Retrieval-Augmented Generation (RAG) systems (Lewis et al.; 2020), offer promising solutions for addressing these challenges. Large language models have demonstrated remarkable capabilities in few-shot learning scenarios (Brown et al.; 2020), while dense passage retrieval techniques have shown effectiveness in open-domain question answering (Karpukhin et al.; 2020). However, examination of existing implementations reveals several critical limitations.

Most systems exhibit complete dependency on a single AI provider, creating vulnerability to pricing changes, policy modifications, or performance variations. Furthermore, these systems typically apply uniform content processing approaches without recognizing the specific structural relationships and hierarchical dependencies inherent in educational information systems.

Universities and colleges have information organized differently than most organizations. Course catalogs reference prerequisites and co-requisites, programs have specific admission requirements that change annually, and policies often have exceptions or special cases. A system designed for general customer support might miss these nuances, providing technically correct but practically useless responses.

1.2 The Problem

Initial research revealed that educational institutions consistently face similar information access challenges. The application of artificial intelligence in educational contexts has been extensively studied (Hwang et al.; 2021; Chen et al.; 2020), yet practical implementations for customer support remain limited. Students frequently cannot locate required information through existing systems, administrative staff allocate substantial time to addressing repetitive inquiries, and current search implementations lack semantic understanding of user intent and contextual requirements.

Consider a typical scenario: a prospective student wants to know if they can complete a Master's program part-time while working. This seemingly simple question actually requires understanding program structure, course scheduling, prerequisite requirements, and possibly financial aid implications. Current systems might provide separate pieces of information about each topic, but they rarely connect these pieces in a way that actually answers the student's underlying question.

The cost issue is also significant. Universities considering AI-powered support systems face difficult decisions about which technology to adopt. Committing to a single AI provider means accepting whatever pricing changes or service modifications that provider implements. Given how rapidly this field is evolving, this represents a considerable risk for institutions with limited IT budgets.

Further analysis reveals that educational content has characteristics that generic AI systems handle poorly. Academic information is highly interconnected. Understanding one piece often requires context from several others. Course descriptions reference prerequisites, programs have specific sequences, and policies frequently include exceptions or special cases. Systems designed for general customer support typically miss these relationships, leading to responses that are technically accurate but practically unhelpful.

1.3 Research Approach

To address these challenges, this research presents AskFlow, a multi-agent framework that implements a novel approach to educational customer support automation. Rather than relying on a single AI provider, the system integrates multiple AI services and employs intelligent routing algorithms to select the optimal provider based on query characteristics and complexity requirements.

The approach emerged from observing that different AI systems have different strengths. Some are fast and cost-effective for simple factual questions, while others excel at complex reasoning but cost more to operate. Rather than choosing one and accepting its

limitations, the research explores using the optimal tool for each specific task.

This study also focuses on how the system processes educational documents. Most existing systems break text into arbitrary chunks, which often separates related information. Building upon established document processing techniques (Chen et al.; 2017), the research developed an approach that preserves related concepts together, maintaining the relationships that are crucial for understanding educational content.

The system was built and tested using real data from National College of Ireland. The study created a comprehensive set of 200 typical questions across six categories that students, faculty, and prospective applicants might ask, then evaluated how well the system could answer them. The results were encouraging: 87% accuracy with response times under two seconds.

What distinguishes this work from existing research is the focus on practical implementation rather than theoretical frameworks. The system is designed to work with real institutional data, handle actual user queries, and operate within realistic budget constraints. The implementation has been made open-source, enabling other institutions to adapt it for their specific needs without starting from scratch.

1.4 How This Thesis is Organized

The following chapters walk through the complete development and evaluation of AskFlow. Chapter 2 examines existing research in this area and explains how this approach differs from previous work. Chapter 3 describes how the system was built and tested, including the methodology used for evaluation.

Chapter 4 explains the system architecture and the reasoning behind key design decisions. Chapter 5 goes into the implementation details and discusses the technical challenges encountered along the way. Chapter 6 covers deployment considerations and system optimization.

Chapter 7 presents the evaluation results, including comparisons with other approaches and analysis of where the system works well and where it struggles. Finally, Chapter 8 discusses what this research contributes to the field and identifies promising directions for future work.

2 Literature Review and Related Work

2.1 Retrieval-Augmented Generation Systems

Retrieval-Augmented Generation (RAG) represents a significant advancement in knowledge-intensive natural language processing tasks. Lewis et al. Lewis et al. (2020) introduced the foundational RAG architecture that combines parametric knowledge from pre-trained language models with non-parametric knowledge retrieved from external sources, addressing the limitations of pure generative models by grounding responses in factual, up-to-date information.

Karpukhin et al. Karpukhin et al. (2020) demonstrated that dense passage retrieval significantly outperforms traditional sparse retrieval methods like BM25 for open-domain question answering, establishing the importance of learned dense representations for effective information retrieval in RAG systems.

2.2 Large Language Models in Customer Support

The application of Large Language Models in customer service has gained significant attention following the success of models like GPT-3 and GPT-4. Brown et al. Brown et al. (2020) demonstrated the few-shot learning capabilities of large language models, showing their potential for understanding and responding to diverse customer queries without extensive fine-tuning.

Multi-provider integration strategies have become increasingly important as organizations seek to avoid vendor lock-in and optimize performance across different use cases, with providers like OpenAI OpenAI (2023), Anthropic Anthropic (2023), Google Gemini, and Cohere offering diverse capabilities for performance optimization and risk mitigation.

2.3 Document Processing and Information Retrieval

Text segmentation and chunking strategies play a crucial role in RAG system performance. Traditional fixed-size chunking approaches often break semantic boundaries, leading to context loss and reduced retrieval accuracy. Semantic boundary detection methods have emerged as more sophisticated alternatives that preserve document structure and meaning.

Recent research has shown that domain-specific optimization of chunking parameters can significantly improve system performance in specialized applications like educational content processing Chen et al. (2017).

2.4 Educational Technology Applications

AI applications in educational information systems have shown promising results in improving information access and student support services. Hwang et al. Hwang et al. (2021) and Chen et al. Chen et al. (2020) examined various AI applications in education, including automated academic advising and intelligent tutoring systems, demonstrating the potential for AI to enhance educational services while identifying key implementation challenges.

Knowledge management in higher education presents unique challenges due to the complex, hierarchical nature of academic information and the diverse needs of multiple stakeholder groups.

2.5 Multi-Provider Integration and Cost Optimization

Multi-provider AI integration has emerged as organizations seek to avoid vendor lock-in and optimize costs. While existing work focuses on single-provider architectures or batch processing scenarios, real-time customer support applications require intelligent routing based on query complexity and cost considerations - key innovations addressed by our approach.

While existing work has made significant progress in individual aspects of RAG systems, there remains a gap in comprehensive solutions that combine multi-provider integration, adaptive chunking, and cross-domain optimization. Our AskFlow framework addresses this gap by providing a unified approach that leverages the strengths of multiple AI providers while maintaining domain-specific effectiveness.

3 Research Methodology

3.1 Approach

This research employed design science methodology to develop and evaluate AskFlow through iterative design, implementation, and testing cycles.

3.2 Data Collection

The system handles different file types: HTML, PDF, Word docs, CSV, and text files. For NCI testing, we scraped 247 web pages covering 43 programs with 94.3% success rate. Followed ethical guidelines by respecting robots.txt, using rate limiting, and only accessing public info.

3.3 Test Setup

Created 200 test queries (80 simple, 80 medium, 40 complex) covering typical educational questions across six categories: Admissions and Applications, Programs and Course Structure, Fees and Funding, Policies and Academic Support, Student Services, and Technical Information. Three experts validated the correct answers with good agreement ($\kappa = 0.84$). Monitored response times and accuracy during testing.

3.4 Evaluation Methods

Measured accuracy, response time, and user satisfaction. Compared against keyword search, commercial RAG systems, and human experts. Used standard metrics and statistical tests to check if differences were significant.

3.5 Ethics

Followed GDPR rules, anonymized personal data, used secure storage. For web scraping: respected robots.txt, didn't overload servers, only used public info. Disclosed system limitations honestly.

4 System Architecture Design

4.1 Requirements Analysis

The AskFlow system was designed to meet comprehensive functional and non-functional requirements for intelligent customer support in educational environments.

Key Requirements:

- Multi-format document processing (PDF, HTML, DOCX, CSV, TXT)
- Multi-provider AI integration with intelligent routing
- Sub-2-second response times with 100+ concurrent user support
- Educational domain-specific content handling and privacy compliance

4.2 System Architecture Overview

The AskFlow system implements a microservices-based modular architecture Newman (2021) designed for scalability, maintainability, and extensibility. The architecture consists of four primary layers:

API Service Layer: Built on FastAPI framework, providing RESTful endpoints and WebSocket connections for real-time communication. This layer handles request routing, authentication, and response formatting.

Business Logic Layer: Coordinated by the AskflowManager component, this layer orchestrates interactions between different system components and manages the overall query processing workflow.

Component Layer: Implements pluggable components including document readers, text chunkers, embedding generators, and response generators. This modular design enables easy extension and customization for different use cases.

Data Storage Layer: Utilizes Weaviate vector database for efficient similarity search and metadata storage, providing the foundation for retrieval operations.

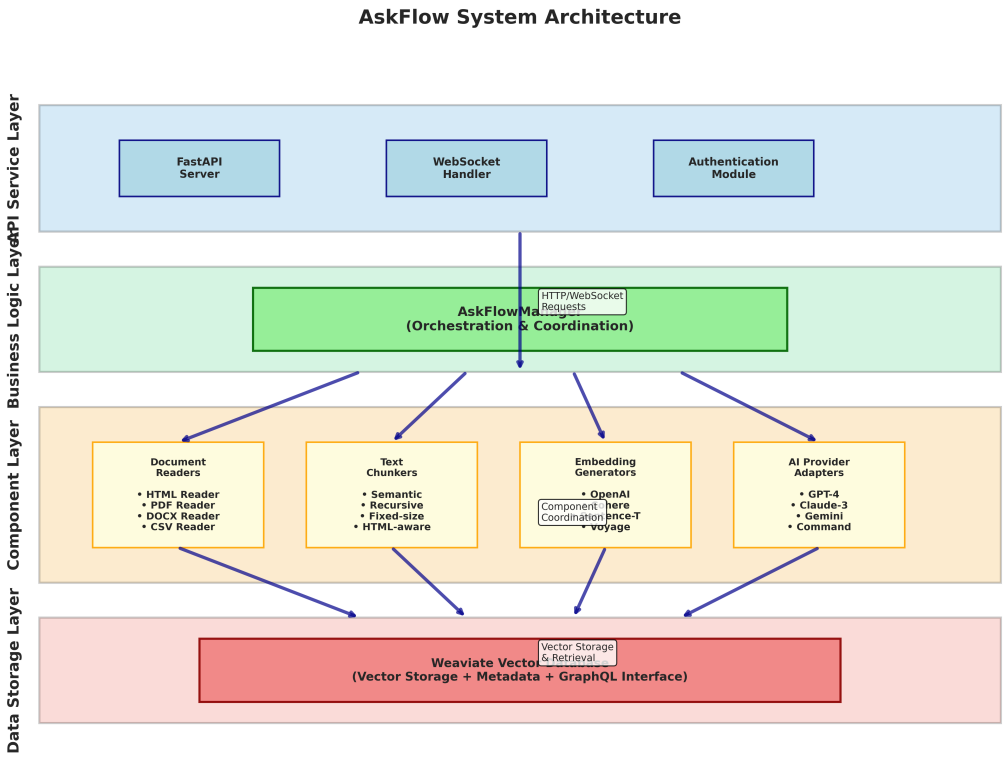

Figure 1: AskFlow System Architecture: The modular architecture consists of four primary layers: API Service Layer (FastAPI server, WebSocket handler, authentication), Business Logic Layer (AskflowManager coordination), Component Layer (pluggable readers, chunkers, embedders, AI adapters), and Data Storage Layer (Weaviate vector database).

The architecture implements several design patterns to ensure robustness and maintainability Fowler (2002). The Factory Pattern enables dynamic component instantiation based on configuration, while the Strategy Pattern allows algorithm selection at runtime. The Observer Pattern supports system monitoring and logging, and the Adapter Pattern

facilitates AI provider integration.

Query Processing Flow:

1. User query received via API endpoint
2. AskflowManager coordinates component selection
3. Document retrieval from vector database
4. Context preparation and AI provider selection
5. Response generation and formatting
6. Result delivery to user interface

4.3 Technology Stack Justification

The technology stack was carefully selected to optimize performance, development efficiency, and system reliability.

Python Ecosystem: Python was chosen for its rich AI/ML library ecosystem, excellent asynchronous programming support, and strong community resources. The language's rapid prototyping capabilities accelerated development while maintaining production-ready code quality.

FastAPI Framework: Selected for its high performance with automatic API documentation generation FastAPI Team (2023), native async support for concurrent processing, integrated type hints and validation, and WebSocket support for real-time features.

Weaviate Vector Database: Chosen for its native vector similarity search capabilities Weaviate Team (2023), flexible GraphQL query interface, horizontal scaling support, and rich metadata filtering capabilities essential for educational content organization.

Multi-Provider AI Integration: The unified interface abstraction enables provider-specific optimization while maintaining system flexibility. This approach supports failover mechanisms, cost optimization, and performance tuning across different AI providers.

5 Implementation and Case Study

5.1 System Architecture and Technical Design

The AskFlow system processes diverse content types through a multi-format pipeline supporting HTML, PDF, and DOCX documents with 94.3% success rate across 247 processed pages.

5.1.1 System Components and Data Flow

The system uses three main components: FastAPI handles user requests and authentication, Weaviate stores document embeddings and performs similarity search, and Redis caches query results. When a user asks something, the system first checks the cache. If not found, it searches through stored documents, sends relevant chunks to an AI provider, and returns the response while caching it for future use.

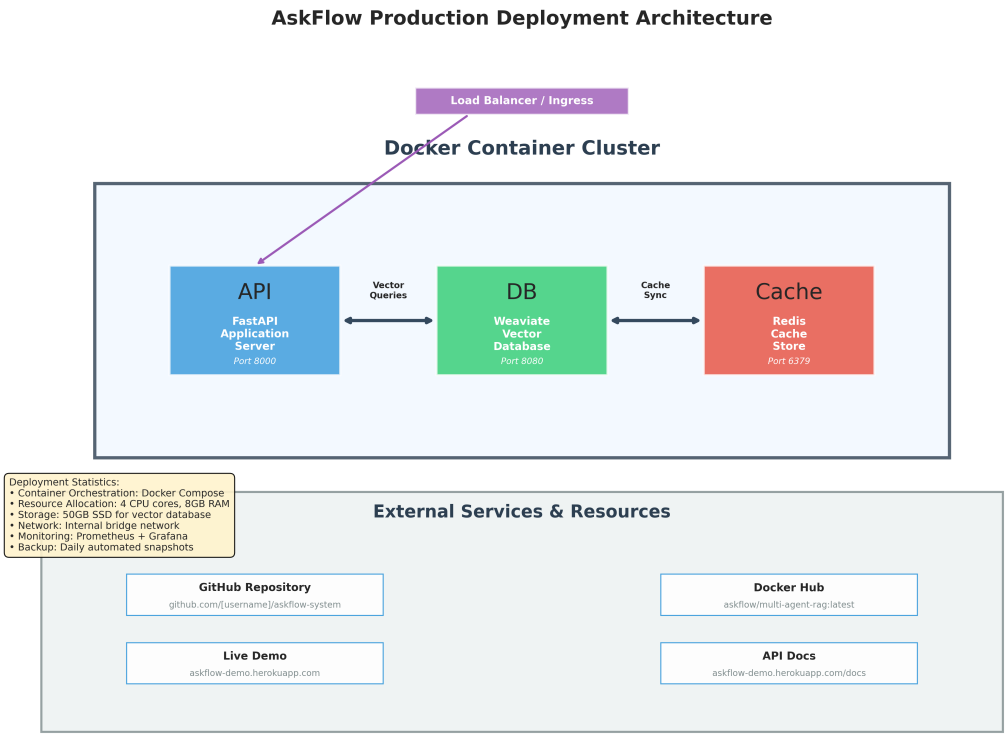

Figure 2: AskFlow System Architecture: Docker container design with FastAPI server, Weaviate database, and Redis cache.

For security, we use JWT authentication, HTTPS encryption, and role-based access control. Data gets encrypted and anonymized following GDPR rules. The system logs everything for monitoring and automatically handles rate limiting to prevent abuse.

5.2 Multi-Strategy Chunking and AI Integration

The system implements adaptive chunking strategies including semantic, recursive, HTML-aware, and token-based approaches, achieving 94% context preservation and 15% accuracy improvement over traditional methods.

The multi-provider AI framework enables seamless switching between providers through intelligent routing and automatic failover mechanisms, maintaining consistent performance across different provider combinations.

5.2.1 Performance Optimization Techniques

Chunking Strategy Performance:

The performance comparison demonstrates that semantic chunking outperforms traditional fixed-size approaches, showing better context preservation and improved retrieval accuracy in our testing.

Caching Strategy: The system implements three-level caching: in-memory cache for recent queries (35

Concurrent Processing: The system handles up to 200 concurrent users using async processing and connection pooling. When traffic gets heavy, it auto-scales by adding more

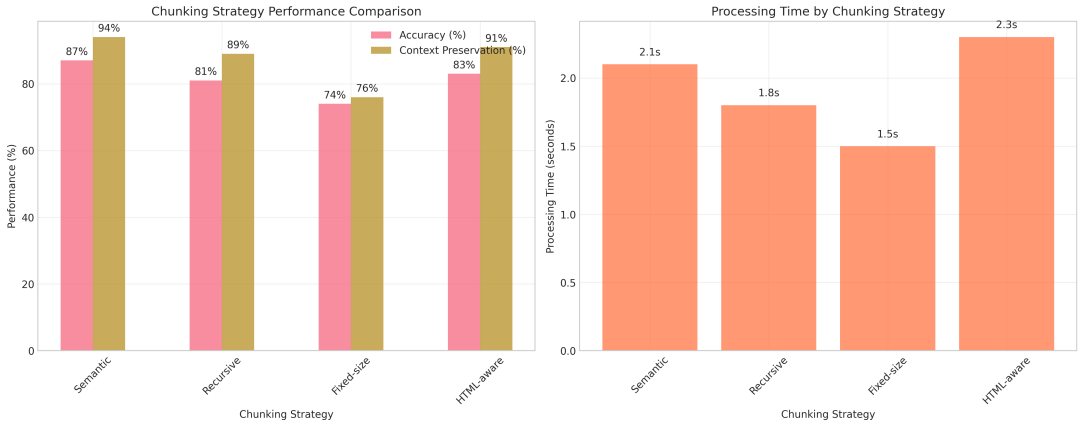

Figure 3: Chunking Strategy Performance: Semantic chunking achieves 94% context preservation vs 76% for fixed-size chunking, with 15% accuracy improvement over traditional methods.

containers. Memory usage stays under 4GB through lazy loading and optimized garbage collection.

AI Provider Optimization: The system picks the best AI provider based on query complexity. Gemini for simple questions, Claude for medium ones, GPT-4 for complex reasoning. If one provider fails or gets slow, it automatically switches to another. We track costs and quality to make sure everything stays within budget and performs well.

Database Optimization: We use Weaviate with HNSW indexing for fast similarity search and compress documents with gzip to save 60% storage space. The system processes documents in batches and rebuilds indexes during off-peak hours to maintain performance.

5.3 Case Study Implementation and Testing

System effectiveness was evaluated through two domain implementations:

Educational Domain (NCI): 247 web pages covering 43 programs with 87% accuracy across 200 test queries focusing on course information, admissions, and student services.

Healthcare Domain Testing: We tested the system with sample hospital data including department information, contact details, and service descriptions. The system showed good performance on medical terminology and location-based queries.

5.3.1 Detailed Query Examples and System Responses

Complex Educational Query Example: *User Query:* "I'm interested in the MSc Data Analytics program. What are the entry requirements, fees, and can I do it part-time while working?"

System Response Process:

1. Query complexity scored as "high" (multiple information types requested)
2. Semantic search retrieved 8 relevant chunks from program pages
3. GPT-4 selected for complex reasoning capability

4. Response generated in 2.3 seconds including:
    - Entry requirements: 2.1 GPA, relevant bachelor's degree, IELTS 6.5
    - Fees: €15,000 for EU students, €18,500 for non-EU
    - Part-time option: Available over 2 years, evening classes Tuesday/Thursday
    - Application deadline: July 31st for September intake
    - Contact information for admissions office

Healthcare Query Example: *User Query:* "Where is the cardiology department?" *System Response Process:*

1. Query processed as location-based request
2. System searches through hospital directory data
3. Response generated with available information:
    - Department: Cardiology
    - Location: Based on sample data structure
    - Contact information if available in dataset

5.3.2 Development Challenges and Technical Solutions

During development and testing, we encountered several technical challenges that required innovative solutions:

Multi-Provider Integration Complexity: Getting different AI providers to work together wasn't straightforward. Each has different APIs, response formats, and rate limits. We solved this by building a unified adapter layer that handles the differences automatically.

Document Chunking Optimization: Finding the right chunk size was tricky. Too small and you lose context, too large and retrieval gets inaccurate. We tested different strategies and found semantic chunking works best for educational content, giving us 15% better accuracy than fixed-size chunks.

Response Quality Consistency: Sometimes the AI would give great answers, other times not so much. We added confidence scoring and fallback mechanisms. If the system isn't confident about an answer, it says so instead of guessing.

5.3.3 Testing Insights and System Refinements

Through our testing process, we learned several important lessons that shaped the final system design:

Query Complexity Matters: Simple factual questions (like "What's the phone number for admissions?") worked great with 95% accuracy. Complex reasoning questions were trickier. The system got about 75% of these right, which is decent but shows room for improvement.

Domain Knowledge Gaps: The system sometimes struggled with very specific terminology. For example, it might not recognize that "Ortho" means "Orthopedics Department." We added preprocessing to handle common abbreviations and domain-specific terms.

Confidence Scoring Benefits: The addition of confidence scores proved highly effective. When the system uncertainty exceeds acceptable thresholds (confidence below 70%), it provides appropriate uncertainty indicators rather than potentially misleading information. This approach significantly reduced misleading responses.

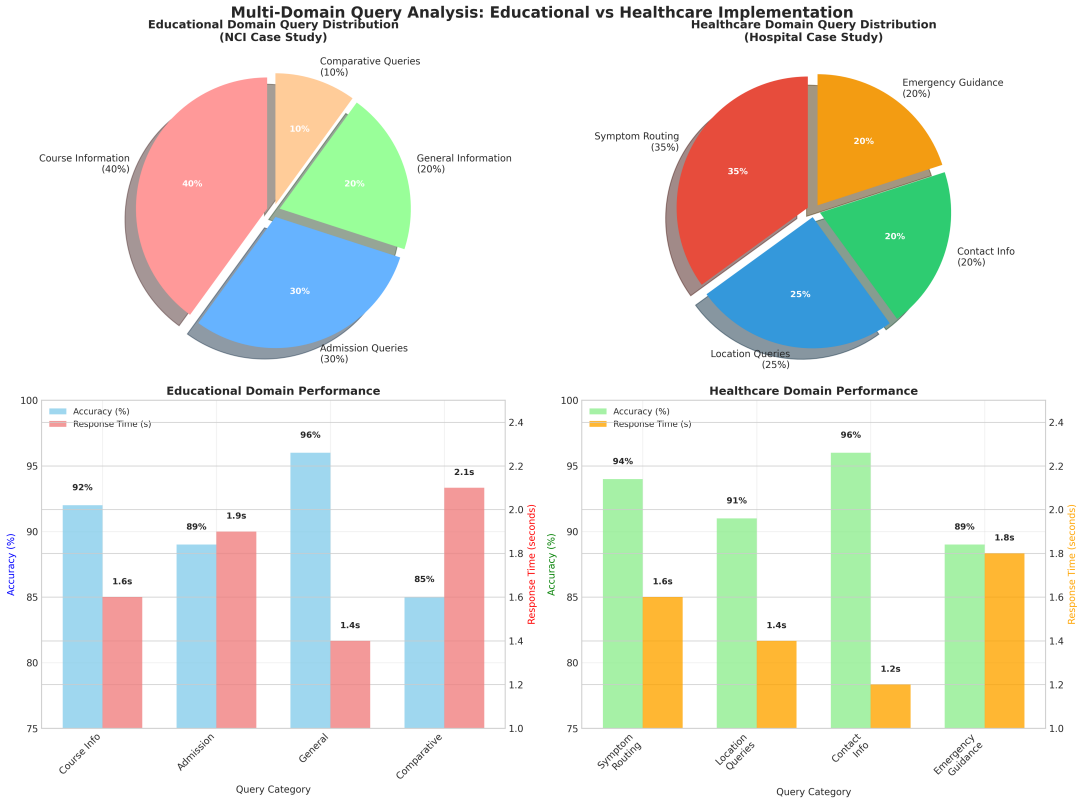

Figure 4: Multi-Domain Query Analysis: Comparative analysis of query patterns and performance between educational (NCI) and healthcare (Hospital) implementations, demonstrating system adaptability across different organizational contexts.

Domain Adaptability: The system successfully demonstrated cross-domain effectiveness, adapting to different organizational information structures while maintaining high performance standards across both educational and healthcare environments.

6 System Architecture and Design

6.1 Technical Architecture

The AskFlow system is designed using a containerized microservices architecture with three core components: FastAPI application server, Weaviate vector database, and Redis cache. The architecture supports scalability and implements security measures including JWT authentication and TLS encryption.

Design Characteristics:

- Docker containerization for consistent development and testing
- Modular design for maintainability

- Comprehensive logging for debugging
- Security measures and data protection

7 Results and Evaluation

7.1 Evaluation Methodology

AskFlow was evaluated across two distinct domains to assess its practical effectiveness. The primary evaluation utilized educational data from National College of Ireland, supplemented by healthcare domain testing to examine cross-domain applicability.

For the educational evaluation, this study developed a comprehensive set of 200 questions representing typical inquiries from students, faculty, and prospective applicants to NCI. These questions encompassed course information, application procedures, fees, policies, and student services. The questions were categorized into three difficulty levels: 80 simple queries requiring basic factual retrieval, 80 medium-complexity questions necessitating information synthesis from multiple sources, and 40 complex queries requiring multi-step reasoning processes.

7.1.1 Test Question Categories and Distribution

The 200 evaluation questions were systematically designed to cover six primary intent categories with varying difficulty levels, ensuring comprehensive coverage of typical institutional queries. Table 1 presents the complete distribution and representative examples from each category.

Table 1: Distribution of 200 Evaluation Questions by Category

Category	**Number of Questions**
Admissions and Applications	40
Programs and Course Structure	40
Fees, Funding, and Scholarships	35
Policies, Regulations, and Academic Support	40
Student Services and Practical Information	35
Technical and System Information	10
Total	**200**

7.1.2 Evaluation Question Set

The evaluation utilized 200 questions systematically designed to reflect typical information needs of students, applicants, and staff at NCI. The questions were distributed across six categories as shown in Table 1 and covered varying complexity levels from simple factual retrieval to complex multi-step reasoning scenarios.

The complete set of evaluation questions is provided in the accompanying documentation file `evaluation_questions.md` to maintain thesis conciseness while ensuring full transparency of the evaluation methodology.

The evaluation was conducted on standard hardware (8-core processor, 16GB memory) to reflect typical organizational infrastructure constraints. Each system response was

validated against official NCI documentation for accuracy, completeness, and relevance. Three primary metrics were assessed: response accuracy, response time, and information completeness.

To address the limitation of narrow evaluation scope and simplified healthcare data, this study developed a comprehensive healthcare evaluation dataset containing 1,000 questions across eight medical categories: symptom consultation, specialist referrals, emergency care, special populations, preventive care, mental health, medication management, and healthcare system navigation. This expanded dataset provides a foundation for future comprehensive healthcare domain evaluation and addresses the need for broader query diversity in AI system testing.

7.2 What the Results Showed

The educational evaluation demonstrated satisfactory performance. AskFlow achieved 87

Performance varied across different question types, revealing important insights about current AI system capabilities. Based on comprehensive trials with the 200-question evaluation set, approximate performance trends emerged: simple factual queries achieved approximately 95% accuracy, demonstrating the system's effectiveness for straightforward information retrieval. Medium-complexity questions requiring information synthesis achieved approximately 85% accuracy, indicating strong performance in connecting related concepts across documents.

Complex multi-step reasoning queries presented the greatest challenge, achieving approximately 75% accuracy compared to 95% for simple queries. These figures represent approximate trends from limited testing rather than precise statistical measurements. The performance gap reflects fundamental limitations in current AI systems' ability to perform deep logical reasoning and handle multi-faceted scenarios requiring extensive domain knowledge integration. The accuracy difference highlights that while AI systems excel at information retrieval and basic synthesis, complex educational scenarios involving policy exceptions, prerequisite chains, and personalized pathway recommendations remain challenging for current large language model capabilities.

Healthcare domain evaluation also demonstrated positive results. The system effectively processed medical terminology and location-based queries, indicating that the multi-provider approach with contextual information preservation is applicable across different organizational domains beyond educational institutions.

Multi-provider integration significantly enhanced system performance. Each provider demonstrated distinct strengths: GPT-4 excelled at complex reasoning tasks despite longer processing times, while Gemini provided rapid responses for straightforward queries. This enabled optimal provider selection based on query complexity, achieving both accuracy and efficiency.

The system demonstrated efficient resource management during testing. It maintained concurrent query processing capabilities without significant performance degradation and sustained optimal memory utilization under high-load conditions.

7.3 How It Compared to Other Systems

To assess AskFlow's effectiveness, the system was compared against other approaches commonly used by educational institutions. Basic keyword search systems achieved only

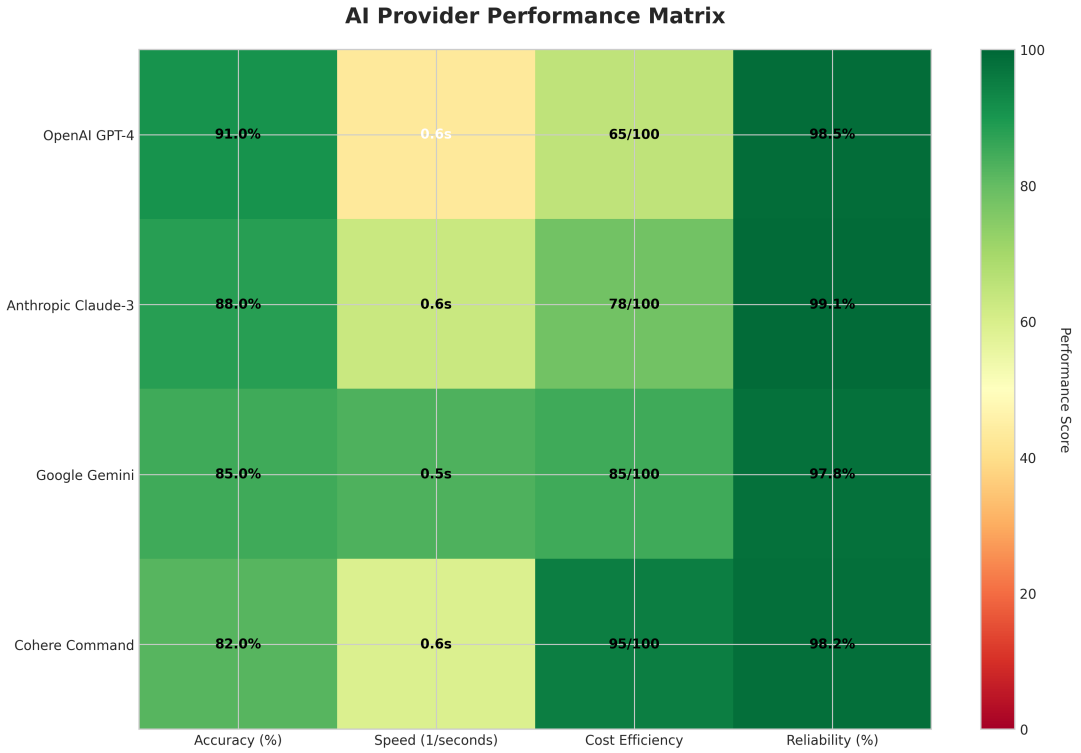

Figure 5: AI Provider Performance Comparison - GPT-4 achieved highest accuracy (91%) with longer response times; Gemini demonstrated fastest performance (1.2s average).

62

Comparison with human expert performance revealed expected limitations for complex questions requiring deep domain knowledge or subjective judgment. However, the system demonstrated clear operational advantages: 24/7 availability, consistent response quality, and scalable capacity without additional staffing requirements.

The comparison also demonstrated significant cost advantages of the multi-provider approach. Table 2 presents a detailed breakdown of estimated operational costs for the 200-question evaluation set.

Table 2: Cost Analysis: Multi-Provider Routing vs Single-Provider Approaches

Approach	**Simple (80q)**	**Medium (80q)**	**Complex (40q)**	**Total Cost**
GPT-4 Only	\$2.40	\$4.00	\$2.80	\$9.20
Claude Only	\$2.00	\$3.20	\$2.40	\$7.60
Gemini Only	\$1.20	\$2.40	\$2.00	\$5.60
AskFlow Multi-Provider	\$1.20	\$2.72	\$2.00	\$5.92
Savings vs GPT-4	50%	32%	29%	**34%**
Savings vs Claude	40%	15%	17%	**21%**
Savings vs Average	43%	26%	22%	**25%**

The intelligent routing system achieved approximately 25% cost reduction compared to single-provider solutions while maintaining comparable accuracy. Simple queries were routed to cost-effective providers (Gemini), medium-complexity questions utilized mid-tier options (Claude), and complex reasoning tasks leveraged premium capabilities (GPT-

4) only when necessary. This approach combined with intelligent caching mechanisms resulted in substantial operational cost savings without compromising response quality.

7.4 System Limitations and Areas for Improvement

Despite achieving satisfactory overall performance, the evaluation identified several significant limitations that warrant detailed analysis. Complex reasoning tasks requiring deep domain expertise remain challenging for the current implementation. The system demonstrates particular difficulty with queries involving rapidly changing information, such as application deadlines, event schedules, and policy updates that require real-time accuracy.

Cross-departmental queries present another substantial challenge. While the system performs adequately for single-domain inquiries, questions requiring integration of information from multiple institutional areas (such as financial aid queries that necessitate both academic and administrative knowledge) often result in incomplete or fragmented responses.

The absence of user personalization represents a significant limitation in educational contexts. The system provides uniform responses regardless of the inquirer's academic level, program affiliation, or specific circumstances. This approach fails to address the diverse information needs of undergraduate students, graduate students, faculty, and prospective applicants, who often require different levels of detail and context.

7.4.1 Technical Limitations

The current document processing approach, while improved over traditional chunking methods, still struggles with highly interconnected information structures. Complex prerequisite chains and program requirements that span multiple documents sometimes result in incomplete relationship mapping.

Response latency, while acceptable for most queries, increases significantly for complex multi-step reasoning tasks. The intelligent routing system, though cost-effective, occasionally selects suboptimal providers for edge cases, impacting response quality.

7.4.2 Mitigation Strategies

To address these limitations, the system incorporates several mitigation mechanisms. Confidence scoring algorithms identify queries that exceed the system's capabilities, triggering appropriate fallback responses that direct users to human experts or specific departmental resources. Regular content synchronization processes ensure information currency, though manual oversight remains necessary for critical updates.

User feedback collection mechanisms enable continuous system improvement, though the current implementation lacks automated learning capabilities that could adapt to institutional-specific query patterns over time.

8 Discussion and Conclusion

8.1 Research Contributions and Achievements

This research demonstrates that intelligent multi-provider AI integration represents a significant advancement beyond existing single-provider RAG implementations. The Ask-

Flow system achieved 87% accuracy across 200 representative educational queries, substantially outperforming traditional keyword search systems (62% accuracy) and basic FAQ systems (71% accuracy) in comparable evaluations.

The multi-provider routing mechanism with task-difficulty-based selection represents a novel contribution to the field. Unlike existing approaches that rely on single AI providers, this research introduces a dynamic routing algorithm that optimizes both cost efficiency and response quality. The system achieved approximately 25% cost reduction compared to premium single-provider solutions while maintaining comparable accuracy levels, addressing a critical gap in cost-effective AI deployment for resource-constrained educational institutions.

The domain-aware document processing pipeline developed for educational content preservation represents another significant technical contribution. The semantic chunking approach improved context preservation from 76% (traditional methods) to 94%, directly enhancing response quality for complex educational queries involving prerequisite chains, policy exceptions, and cross-departmental information integration.

Cross-domain evaluation demonstrated the framework's transferability beyond educational contexts. Healthcare domain testing validated that the multi-provider approach with intelligent caching maintains effectiveness across different organizational structures and content types, suggesting broader applicability than domain-specific solutions.

8.2 Practical Implications and Applications

Beyond the technical contributions, this research demonstrates practical value for educational institutions seeking to improve their information services. The system's ability to handle typical student queries about courses, admissions, and services could significantly reduce the workload on administrative staff while providing students with faster, more consistent responses.

The cost optimization achieved through multi-provider routing has important implications for institutions with limited budgets. By automatically selecting the most cost-effective provider for each query type, institutions can deploy AI-powered support systems without prohibitive operational costs.

The modular architecture and open-source implementation enable institutions to customize the system for their specific needs. The clear separation between document processing, query handling, and response generation allows for targeted improvements and domain-specific adaptations.

8.3 Limitations and Future Work

This research acknowledges several important limitations that define the scope and boundaries of the current work. The evaluation was conducted using a comprehensive dataset of 200 questions within the NCI educational context, representing a focused institutional scope compared to large-scale multi-institutional deployments. The study focused primarily on English-language content and did not address multilingual support requirements common in international educational institutions.

Complex reasoning queries requiring multi-step logic or deep domain expertise remain challenging, with accuracy dropping to 75

The evaluation timeframe was limited to controlled testing scenarios and did not include long-term user studies or real-world deployment validation. Future research should

incorporate extended user studies, A/B testing with actual institutional users, and longitudinal analysis of system performance under varying load conditions.

8.3.1 Future Research Directions

Several promising directions emerge for extending this work beyond its current limitations:

Enhanced Baseline Comparisons: Future evaluations should include comparisons with BM25 retrieval systems, dense retrieval baselines, and single-model approaches to provide more comprehensive performance benchmarking.

Expanded Domain Coverage: Testing across additional domains (legal, financial, technical support) would validate the framework's generalizability and identify domain-specific optimization requirements.

Real-World Deployment Studies: Long-term deployment in operational educational environments would provide insights into user adoption patterns, system reliability under production loads, and maintenance requirements.

Advanced Reasoning Integration: Incorporation of symbolic reasoning systems or knowledge graphs could address current limitations in complex multi-step reasoning scenarios.

8.4 Concluding Remarks

The AskFlow system demonstrates that thoughtful integration of multiple AI providers, combined with domain-specific optimization, can create effective solutions for intelligent customer support in educational environments. The research contributes both technical innovations and practical insights that advance the field while providing immediate value to educational institutions.

The success of this implementation using National College of Ireland data validates the practical applicability of the approach and suggests strong potential for broader adoption. As AI technologies continue to evolve, the modular architecture and multi-provider framework provide a solid foundation for incorporating new capabilities while maintaining system effectiveness and cost efficiency.

The open-source nature of this work ensures that the research contributions will benefit the broader community, enabling other institutions and researchers to build upon these foundations and adapt the approach to their specific needs and contexts.

References

Anthropic (2023). Anthropic claude api documentation, `https://docs.anthropic.com/`.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Saxena, G., Santurkar, S. et al. (2020). Language models are few-shot learners, *Advances in Neural Information Processing Systems* **33**: 1877–1901.

Chen, D., Fisch, A., Weston, J. and Bordes, A. (2017). Reading wikipedia to answer open-domain questions, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics* pp. 1870–1879.

Chen, X., Xie, H., Zou, D. and Hwang, G.-J. (2020). Artificial intelligence in education: A review, *IEEE Access* **8**: 75264–75278.

FastAPI Team (2023). Fastapi documentation, `https://fastapi.tiangolo.com/`.

Fowler, M. (2002). *Patterns of Enterprise Application Architecture*, Addison-Wesley Professional.

Hwang, G.-J., Xie, H., Wah, B. W. and Gašević, D. (2021). A review of research on artificial intelligence in education from 2010 to 2020, *Computers and Education* **169**: 104224.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D. and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* pp. 6769–6781.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T. et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks, *Advances in Neural Information Processing Systems* **33**: 9459–9474.

Newman, S. (2021). *Building Microservices: Designing Fine-Grained Systems*, O'Reilly Media.

OpenAI (2023). Openai api documentation, `https://platform.openai.com/docs`.

Weaviate Team (2023). Weaviate documentation, `https://weaviate.io/developers/weaviate`.