Institute of Systems Science, National University

MASTER OF TECHNOLOGY PROJECT REPORT

ArchiveFlow

TEAM MEMBERS
SUN YANSHU A0296705U
WANG WENJIE A0296855H
WU ZHENGXI A0296199E

Table of Contents

1 Project Overview

In the digital age, organizations and institutions face a rapidly growing amount of information, which leads to significant challenges in information retrieval and decision-making. Whether in corporations, educational institutions, or research facilities, stakeholders often struggle with slow information access, inefficient decision-making, and underutilized knowledge bases. This challenge extends to various scenarios, from students navigating extensive academic resources to healthcare providers accessing patient histories, and researchers synthesizing vast amounts of scientific literature.

To address these challenges, we propose ArchiveFlow, an intelligent knowledge management and decision automation system powered by advanced artificial intelligence technology. By leveraging retrieval augmented generation (RAG) technology, ArchiveFlow enhances information processing capabilities and optimizes users' decision-making processes. Our core goal is to build a robust RAG system that enables quick access to relevant information through intelligent retrieval and in-depth analysis, facilitating more informed and efficient decision-making.

Unlike conventional language models such as ChatGPT, ArchiveFlow specializes in domain-specific knowledge management and decision support. The system stands out through its ability to integrate real-time data sources, maintain strict data security protocols, and adapt to diverse organizational needs. Its architecture ensures both flexibility in deployment and reliability in operation, making it suitable for various institutional settings.

Through these innovations, ArchiveFlow serves as a powerful assistant in the age of information explosion, helping organizations and individuals navigate complex knowledge landscapes. The system improves decision-making efficiency, optimizes resource allocation, and enhances operational effectiveness, representing a significant advancement in knowledge management technology.

2 Business Case and Market Research

2.1 Business Case

The impact of inefficient knowledge management is particularly evident in various sectors. In healthcare, medical professionals struggle to quickly access and utilize the latest treatment protocols and patient records, potentially affecting patient care quality. Educational institutions face challenges in managing their vast academic resources, making it difficult for students and faculty to efficiently access research materials and course content. Research organizations grapple with organizing and retrieving scientific data from multiple studies and experiments. For enterprises, this challenge manifests in extended employee onboarding periods, inconsistent policy implementation, and delayed decision-making processes, directly impacting operational efficiency and competitiveness. These issues underscore the critical need for an advanced knowledge management system that can serve diverse organizational needs.

ArchiveFlow addresses these challenges by providing an integrated platform that transforms how organizations manage and utilize their knowledge assets. Through retrieval-augmented generation (RAG) technology, the system enables healthcare providers to instantly access relevant medical protocols, helps educators and students efficiently navigate academic resources, and assists researchers in synthesizing data across multiple studies. For enterprise users, ArchiveFlow streamlines employee training, ensures consistent policy implementation, and accelerates decision-making processes.

The system's impact extends beyond immediate operational improvements to long-term organizational success. By facilitating faster, more accurate access to critical information, ArchiveFlow helps organizations reduce operational costs and make more informed decisions. The platform's intelligent analysis capabilities provide valuable insights that drive strategic planning and innovation. As organizations across various sectors increasingly recognize the importance of efficient knowledge management, ArchiveFlow's comprehensive solution addresses a growing market need while delivering tangible value to its users.

2.2 Market Research

In the field of knowledge management and information retrieval, AI technology has entered a stage of rapid development. Through extensive market research, we identified that while many existing systems excel at dialogue generation, they often fall short in retrieval performance and practical utility. ArchiveFlow differentiates itself through enhanced information retrieval capabilities combined with advanced generation technology, leading the market in several key aspects:

1. Enhanced Retrieval and Generation Integration

   While traditional language models like GPT or BERT offer sophisticated generation capabilities, they rely heavily on pre-trained data without real-time retrieval functionality. ArchiveFlow's RAG system uniquely combines generative models with dynamic knowledge bases, enabling real-time information retrieval from specialized databases. This integration ensures both accuracy and contextual relevance in information delivery, significantly outperforming systems that rely solely on generation.

2. Scalable Performance

   As knowledge bases grow exponentially in size and complexity, many existing systems struggle with performance at scale. ArchiveFlow addresses this challenge through optimized vector embedding technology, enabling efficient processing of large-scale data while maintaining high retrieval accuracy. The system excels at handling both structured and unstructured data, delivering consistent performance even with complex, extensive knowledge bases.

3. Domain Adaptation

   The system's advanced retrieval capabilities enable fine-tuning of models to specific domains and use cases, providing deeper understanding of specialized terminology and context. This adaptability allows ArchiveFlow to excel in domains where general-purpose language models might fall short, while maintaining the ability to handle general knowledge management tasks.

4. Enterprise-Grade Security

   In an era where data security is paramount, ArchiveFlow implements robust security measures throughout its architecture. Unlike open models, the system enables organizations to maintain complete control over their knowledge bases, ensuring sensitive information remains protected during both retrieval and generation processes. This closed-loop approach to knowledge management makes ArchiveFlow particularly suitable for environments with stringent data protection requirements.

These technical advantages position ArchiveFlow as a market leader in intelligent knowledge management solutions. The system's combination of advanced retrieval capabilities, scalable architecture, and enterprise-grade security addresses the core challenges organizations face in managing and utilizing their knowledge assets effectively.

3 System Design

ArchiveFlow System Architecture is designed with modularity and scalability principles to efficiently address enterprise knowledge management and retrieval needs. The system consists of three layers: data layer, application layer, and presentation layer. Each layer is decoupled to facilitate easier maintenance, updates, and future enhancements.

3.1 Data Layer

This foundational layer is responsible for storing and managing the enterprise knowledge base. Users can upload various data formats (such as PDF, Word, CSV, URL etc.) through the system's intuitive interface. Once uploaded, the data is parsed into smaller text chunks, vectorized into embeddings, and stored in a high-performance vector database, Milvus.

Milvus efficiently retrieves similar vectors using algorithms like Euclidean distance (L2). In conjunction with Milvus, a MySQL database manages metadata, user sessions, and system configuration. Together, these databases ensure efficient data storage and enable fast, relevant retrieval. The system also optimizes storage and retrieval by merging similar vectors, thereby reducing redundancy and enhancing performance.

3.2 Server Layer

Serving as the core functional layer, this component implements a sophisticated retrieval and generation pipeline. The system processes user queries through a dual-stage retrieval process before generating contextual responses.

1. Query Processing: The initial stage handles user input through a dedicated LLM, ensuring accurate understanding of query intent and context. The system then performs embedding-based retrieval to identify relevant information chunks from the vector database.

2. Enhanced Retrieval: A second retrieval stage further refines the search results through rerank, significantly improving response accuracy. This two-stage approach ensures both comprehensive coverage and precise relevance in information retrieval.

3. Response Generation: The final stage utilizes a Reader LLM to synthesize information from the retrieved chunks and generate coherent, contextually appropriate responses. This process ensures that answers are both accurate and relevant to the user's query.

3.3 Presentation layer

This layer provides a comprehensive and user-friendly web interface for interacting with the system. Users can easily upload data, initiate queries, and view answering results in real-time.

1. Customizable LLM Interface: A key feature of this layer is the highly customizable LLM interface. Users can input their own API key and base URL to tailor the language model to their specific needs. This allows users to fully customize the LLM interface according to their preferences, enabling more personalized and effective interactions.

2. Multi-turn Conversation Mode: In addition to basic query-answering functionalities, users can toggle a multi-turn conversation mode. This feature allows them to maintain context across multiple interactions, facilitating deeper and more continuous conversations with the system.

3. Retrieval Mode: The interface also includes a retrieval option that allows users to refine search results for specific question to obtain better accuracy. And if users select the retrieval function during a query, the retrieved content will be retained until they initiate a new chat, thereby improving the accuracy and relevance of subsequent interactions.

The system adopts a front-end and back-end separation design, ensuring high scalability. The decoupling of the front-end user interface from the core back-end services and database allows multiple front-ends to collaborate with multiple back-ends simultaneously. This design enhances the system's flexibility and maintainability, supporting future functional expansions and technology updates. Additionally, the system's modular design integrates seamlessly with existing systems through APIs, allowing users to customize it to their needs.

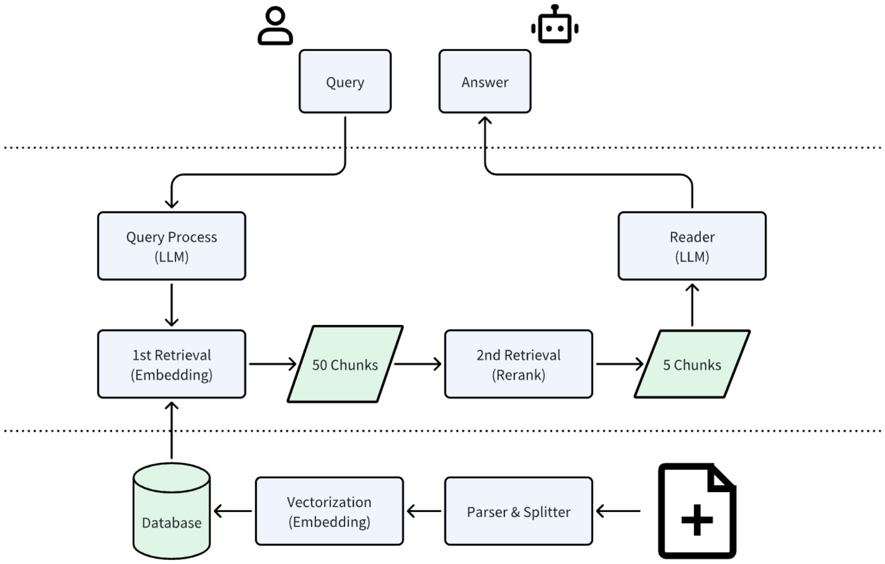

Figure 1: The system design graph.

4 System Development and Implementation

This section will detail the tools, technologies, development stages, and major challenges encountered during the project development process. ArchiveFlow combines cutting-edge AI technology with the needs of enterprise-level systems, focusing on building an efficient retrieval system and a scalable knowledge management platform.

4.1 Development Tools and Technologies

ArchiveFlow is built with a modern technology stack that seamlessly integrates AI capabilities with robust enterprise tools, enabling efficient file management, accurate search functionality, and dynamic interactions.

4.1.1 Core Technologies

ArchiveFlow leverages Python's Sanic framework on the server side, utilizing its asynchronous capabilities to handle multiple requests concurrently. This supports essential functions like file uploads, search queries, and API interactions, creating a responsive and scalable platform. For the user interface, HTML, CSS, and JavaScript work together to deliver a responsive, visually appealing experience, supporting file uploads, model interactions, and Q&A functionality.

4.1.2 Server Architecture

ArchiveFlow's server architecture is designed around four core modules to ensure efficient collaboration of file processing, model management, data retrieval, and user interaction:

1. File Parsing: Using LangChain and custom-developed tools, ArchiveFlow processes documents by parsing them into manageable segments based on file type. These parsed chunks then become the basis for vector embedding.

2. File System Management: The self-developed model manager embeds these document chunks into vector representations and inserts them into the database, tracking file status regularly for updates. This custom-built module also orchestrates the vectorization of document fragments and user queries.

3. Model Manager: This module loads and manages all models involved in the process, handling the vectorization for each document fragment and user query. The model manager ensures relevant embeddings are generated, enabling effective retrieval and analysis.

4. LLM Chat Interface: This component integrates API interfaces for LLM-based conversations. It facilitates the exchange between users and LLMs according to OpenAI standards, ensuring that LLM interactions can leverage any retrieved data.

4.1.3 Data Retrieval Process

When a user submits a query, the system follows a two-stage retrieval process:

1. Initial Retrieval: The system retrieves up to 50 relevant chunks from the vector database using vector embeddings.

2. Reranking: It then reranks these chunks based on their similarity to the original query using models like bcereranker-base_v1, retaining the top 5 chunks for further analysis. This multi-stage retrieval ensures that only the most relevant and concise information is processed, enhancing both retrieval accuracy and efficiency.

4.2 Implementation Phase

During the implementation phase, the development of ArchiveFlow followed a structured and iterative process:

1. System Design and Integration: Clarified system goals to create an efficient retrieval system and integrated the back-end with the database using the Sanic API for efficient file uploads and search queries.

2. File Processing and Vectorization: The FileSystem module managed document reception and processing, embedding uploaded files with the bce-embedding-base_v1 model and storing them in Milvus.
3. Search Optimization : Implemented a two-stage retrieval process for faster results, and tuning the reranker model to achieve optimal results.
4. Deployment and Testing: Deployed the system to a public cloud environment and conducted testing for high-traffic scenarios to ensure performance and availability.
5. System Optimization: Enhanced system concurrency and stability through architecture optimizations and improved front-end response time, providing a smoother user experience.

4.3 Challenges Encountered

ArchiveFlow encountered several challenges during development. Managing large-scale data efficiently required optimization in embedding processes and database queries to ensure Milvus could handle vector retrieval with minimal latency. One major issue was database instability caused by invalid data not conforming to the schema, which led to Milvus crashes. To address this, we implemented file format restrictions on the front end and dual validation on the back end to prevent unsupported uploads. Additionally, Sanic's asynchronous nature presented initial difficulties, especially in ensuring non-blocking file uploads and maintaining system responsiveness under load. High-load performance tuning, including load balancing and API response time optimization, was also crucial to sustaining system performance under heavy demand.

Moreover, we undertook a semantic correction of user questions, although it was ultimately not adopted. The error correction function addressed Chinese typos, incorrect order, missing characters, and extra characters, as well as English typos, incorrect order, missing characters, extra characters, and grammatical errors. The testing metric used was the average cosine similarity with the original sentence before and after correction. The results indicated that after 50 rounds of LoRa fine-tuning, the performance of the 0.5B model could approach that of the 3B model. However, after discussions, it was determined that the improvement brought by fine-tuning was not significant enough, leading to the decision that it was a waste of response time and thus ultimately not adopted.

Table 1: Similarity Test Result for Correction

Model	Before Correction	After Correction
qwen2.5-0.5B-Instruction	0.916	0.880
qwen2.5-1.5B-Instruction	0.916	0.923
qwen2.5-3B-Instruction	0.916	0.958
qwen2.5-0.5B-Instruction(SFT)	0.916	0.952

5 Findings and Discussion

ArchiveFlow demonstrates efficient document retrieval capabilities and a superior user experience. By leveraging advanced machine learning and vectorization techniques, the system can quickly locate relevant information within vast amounts of documents.

5.1 Key Findings

Table 2: Similarity Test Result for Correction

Metric	Details
Average Retrieval Time	Typically 1 to 2 seconds, varying based on device performance and network conditions.
File Processing Speed	A 10MB PDF file can be parsed in under one minute, while plain text files are processed in approximately 30 seconds.

The testing environment is listed below, and actual performance may vary depending on specific configurations and conditions. Therefore the testing results are for reference only. It's worth noting that retrieval speed improves significantly when both the database and server are on the same local area network (LAN). Conversely, network latency may affect overall performance when the servers are located in different regions, which can be a critical factor for users operating in geographically dispersed environments.

Table 3: Database Server

Component	Details
CPU	E5-2650v2
Memory	24GB
GPU Support	No
Location	Meichuan, Sichuan, China

By outlining the testing environment specifications, users gain a clearer understanding of the context behind the results. This transparency aids in setting realistic performance expectations. Variations in future testing environments

Table 4: BackEnd Server

Component	Details
vCPUs	8
Memory	30GB
GPU Support	NVIDIA A10
Location	Chengdu, Sichuan, China

may lead to different outcomes, highlighting the need to adapt ArchiveFlow to specific operational requirements.

5.2 Limitation

ArchiveFlow's performance is significantly influenced by its reliance on user-provided large language model interfaces. While the system excels in document retrieval and management, this dependence limits its universality and accessibility. Users are required to configure and maintain the necessary language model interfaces, which may create additional challenges, particularly for small businesses or those without technical support.

Moreover, the variability in user-provided interfaces can lead to inconsistent system performance. Different model configurations may impact the overall efficiency, especially during query processing and response generation, potentially resulting in delays or inaccuracies. Although ArchiveFlow is designed for flexibility, this same flexibility can complicate technical implementation for users.

In summary, despite ArchiveFlow's advantages in document management and retrieval, there is significant scope for enhancing user experience and reducing technical dependencies. Future versions could explore broader language model options or integrated solutions to lower the technical barriers and improve the system's applicability.

6 Conclusion and Future Work

6.1 Conclusion

In conclusion, ArchiveFlow effectively meets the complex needs of knowledge management through its innovative design and advanced retrieval capabilities. By integrating machine learning and vectorization techniques, the system ensures efficient document retrieval, enabling users to manage their files more effectively by quickly and accurately accessing relevant information. The modular architecture allows for scalable and flexible integration with existing systems, making ArchiveFlow a robust solution looking to streamline their archiving processes.

Moreover, integrating real-time analytics and reporting features could provide users with deeper insights into their document management practices. Conducting further tests to validate the system's performance across diverse environments will be crucial in ensuring its reliability and effectiveness in practical applications.

6.2 Future Work

Several avenues for future work can enhance ArchiveFlow's functionality. One potential direction is to refine the AI algorithms used for document classification and retrieval, incorporating more sophisticated natural language processing techniques to improve user query understanding. Additionally, expanding the system's support for various data formats and enhancing the user interface could further improve usability and accessibility.

Through these improvements, ArchiveFlow aims to solidify its position as a leading solution in knowledge management and retrieval, continuously adapting to meet the evolving needs of its users while enabling better file management.

7 Appendices

This section analyzes the system's functionalities in relation to the knowledge, techniques, and skills from the modular courses: Machine Reasoning (MR), Reasoning Systems (RS), and Cognitive Systems (CGS). This mapping highlights how course content informs system design.

System Functionalities and Course Module Mapping Table

System Functionality	**Module**	**Knowledge/Techniques/Skills**
File Management	MR	Data storage and retrieval methods, knowledge reasoning and management.
User Interaction	CGS	Cognitive model design, user behavior analysis.
Multi-turn Dialogue	RS	Natural language processing techniques, machine learning model training.
Retrieval Functionality	MR	Information retrieval techniques.
LLM Customization	CGS	Machine learning API integration, model deployment and optimization.
User Management	RS	Access control strategies, user role management.

Mapping Explanation

System Functionality	**Mapping Explanation**
File Management	This functionality relates to the MR course, focusing on data storage and retrieval methods, emphasizing the use of reasoning techniques for efficient knowledge base management.
User Interaction	Combined with the CGS course, the system enhances user experience through cognitive models and analyzes user behavior to optimize interaction methods.
Multi-turn Dialogue	This feature stems from the RS course's natural language processing and machine learning model training, providing users with the ability for continuous dialogue.
Retrieval Functionality	Closely related to the MR course, this functionality utilizes information retrieval techniques, which are essential for implementing reasoning methods that efficiently extract and organize information from the knowledge base.
LLM Customization	Supports the CGS course's machine learning API integration, allowing users to customize model interfaces based on their needs.
User Management	This functionality is based on RS course content, implementing effective access control strategies to ensure the security and privacy of user data.

Mapping the system functionalities to the MR, RS, and CGS courses clarifies how acquired knowledge shapes the system. This analysis not only enhances understanding but also supports future improvements.