Policy Research Journal
ISSN (E): 3006-7030 ISSN (P) : 3006-7022 Volume 4, Issue 3, 2026

# AN ENTERPRISE MULTI-AGENT AI FRAMEWORK FOR HR DOCUMENT INTELLIGENCE USING RETRIEVAL-AUGMENTED GENERATION AND VECTOR SEARCH

**Muhammad Talha Jahangir[*1], Samina Naz[2], Muhammad Tahir[3], Muneeza Talib[4], Anas Mahmood[5], Khadija Rehman[6], Zainab Faheem[7], Humera Muhammad Ali[8]**

[*1,5,6,7,8]*Dept. of Computer Science, Muhammad Nawaz Sharif University of Engineering & Technology, Multan, Pakistan*
[2]*Faculty of Computing and Emerging Technologies, Emerson University Bosan road Multan*
[3]*Teaching Fellow, Department of Computer Science, LUMS*
[4]*Department of Information Technology, Bahauddin Zakariya University, Multan*

[1]mtalhajahangir@mnsuet.edu.pk

DOI: http://doi.org/10.5281/zenodo.21524887

**Keywords**
*Multi-Agent System, HR Automation, Large Language Models, Retrieval-Augmented Generation, FAISS, Vector Search, CrewAI, Ollama, Local-First AI*

**Article History**
*Received: 19 January 2026*
*Accepted: 04 March 2026*
*Published: 18 March 2026*

**Copyright** @Author
**Corresponding Author: ***
**Muhammad Talha Jahangir**

## Abstract

*Human Resource (HR) departments in universities and large enterprises remain constrained by manual document workflows: policy retrieval, compliance verification, and correspondence drafting are performed by administrators reading dense unstructured charters page by page. Existing cloud-based Large Language Model (LLM) solutions promise automation but introduce three unacceptable enterprise risks: factual hallucination, opaque "black-box" reasoning, and cross-border transmission of personally identifiable information. This paper proposes an Enterprise Multi-Agent HR Document System that resolves these limitations through a local-first cognitive architecture combining Multi-Agent Systems (MAS), Retrieval-Augmented Generation (RAG), and high-performance vector indexing. A Chief Orchestrator agent delegates sub-tasks to three specialized workers (a Document Finder, a Compliance Vet, and a Document Writer), each governed by a bounded system prompt and an isolated toolset. Institutional documents are ingested through a sliding-window chunking pipeline (chunk size c = 1000, overlap o = 200), encoded into 384-dimensional dense vectors using the all-MiniLM-L6-v2 Sentence-Transformer, and indexed in a FAISS store paired with a structured JSON metadata registry that preserves the source file path, section heading, page number, and version tag of every chunk. Quantized LLMs (Llama-3-8B-Instruct) are served locally via Ollama, keeping all sensitive HR data on-premise. Empirical evaluation on 150 pages of university bylaws (≈450 chunks) shows sub-second FAISS retrieval (2.1 ms per query on the optimized workstation), a Precision@3 of 0.942 on synonym-heavy natural-language queries, and an approximately 90% reduction in end-to-end document retrieval latency relative to manual administrative lookup. Every generated response is traceable through the JSON registry back to its exact source paragraph, closing the enterprise auditability gap that has historically blocked adoption of generative AI in regulated HR workflows.*

## I. INTRODUCTION

The Fourth Industrial Revolution has repositioned Artificial Intelligence (AI) from an academic curiosity to the primary engine of enterprise automation, predictive analytics, and conversational intelligence. Within this transformation, Human Resource (HR) management is a critical yet notoriously manual domain: it governs organisational talent, enforces statutory compliance, and maintains the corporate document repository, but does so through workflows that remain largely transactional. HR document automation is therefore one of the most under-automated pillars of modern enterprise operations, and it is the concern of this paper.

HR information systems have evolved through four generations. First-generation paper registries offered zero scalability. Second-generation relational databases (RDBMS) introduced digital indexing but were rigidly schema-bound. Third-generation cloud-based Human Resource Information Systems (HRIS) centralised storage and enabled web access but remained fundamentally passive: they behave as digital filing cabinets requiring explicit human orchestration and rigid keyword matching. A fourth generation of cognitive multi-agent systems is now emerging, offering autonomous reasoning, semantic understanding, and context-aware Retrieval-Augmented Generation (RAG), but it demands strict verification guardrails to be deployable inside regulated organisations.

Despite substantial enterprise investment in digital transformation, contemporary HR departments remain limited by five interconnected structural bottlenecks. Unstructured data overload: over 80% of an organisation's internal knowledge resides in unstructured formats such as PDF policy sheets, DOCX charters, and scanned bylaws, which conventional RDBMS-backed HRIS platforms cannot query directly. Semantic blindness of lexical search: standard enterprise search engines rely on exact-match algorithms (BM25, SQL LIKE); an employee asking about "paternity leave protocols" will fail to retrieve a charter section indexed under "Spousal and Family Bonding Accommodations," although the two refer to the same policy [1]. Monolithic automation bottlenecks: rule-based "if-this-then-that" pipelines are effective for deterministic tasks but collapse under multi-step cognitive objectives such as cross-referencing an employment contract with an updated policy amendment. Cognitive fatigue and interpretive variance: human administrators facing dense multi-page charters produce inconsistent policy interpretations, creating compliance risk in decentralised institutions like universities. The hallucination and trust barrier: vanilla LLMs generate grammatically fluent but factually incorrect assertions; a misstated healthcare-coverage answer or miscalculated severance benefit exposes the organisation to legal liability [2].

The purpose of this research is to design, implement, and evaluate an intelligent, secure, and production-ready cognitive architecture that automates the HR document lifecycle through multi-agent orchestration.

### *A. Contributions*

The contributions of this paper are as follows.

1. A role-decoupled four-agent architecture (Chief Orchestrator, Document Finder, Compliance Vet, Document Writer) built on CrewAI [3] that empirically avoids the "context overload" and "attention dilution" failure modes of single-agent LLM setups.
2. A dual-storage design that couples a FAISS vector index [4] with a structured JSON metadata registry, providing full audit lineage for every generated response.
3. A local-first deployment topology using Ollama-hosted quantised LLMs (Llama-3-8B-Instruct) that keeps sensitive HR data on-premise while retaining sub-second retrieval.
4. An empirical evaluation on a 150-page university-policy corpus reporting Precision@3 of 0.942 and an approximately 90% reduction in retrieval latency against manual baselines, together with a per-goal analysis of the design objectives.

The remainder of the paper is organised as follows. Section II reviews the theoretical foundations and related enterprise architecture. Section III details the hardware and software stack. Section IV presents the system methodology, data pipeline,

and multi-agent control flow. Section V reports empirical benchmarks and per-goal evaluation. Section VI concludes with limitations and a future-work roadmap.

## II. RELATED WORK

The proposed system sits at the intersection of three rapidly-evolving domains: Distributed Artificial Intelligence (Multi-Agent Systems), Knowledge Engineering (Retrieval-Augmented Generation), and Enterprise Information Systems.

### *A. From Monolithic LLMs to Multi-Agent Systems*

Early enterprise LLM deployments treated the underlying neural network as a monolithic conversational oracle: massive single prompts contained all system rules, data constraints, and execution context. Vaswani et al. [5] showed that the self-attention mechanism, while effective for global context, scales quadratically ($O(n^2)$) with sequence length $n$. Appending hundreds of pages of institutional policies into a single context window therefore degrades performance through two phenomena. The "Lost in the Middle" effect, formalised by Liu et al. [2], shows that autoregressive language models retrieve information at the beginning or end of the prompt with high fidelity but drop accuracy sharply for content buried in the middle. Context overload and attention dilution occur when a single LLM must simultaneously act as document parser, compliance auditor, drafting assistant, and security gatekeeper; its attention weights disperse across competing objectives, increasing hallucination rates.

Multi-Agent Systems resolve these failure modes by rooting themselves in Minsky's Society of Mind [6], in which intelligence emerges from the interaction of localised specialised sub-agents. Formally, each agent is a tuple

$$A = \langle S, \mathcal{A}, \tau, R, \pi \rangle \quad (1)$$

where S is the perceptible environment state set, $\mathcal{A}$ is the action set, $\tau : S \times \mathcal{A} \to S$ is the transition function, $R$ is the reward function, and $\pi : S \to \mathcal{A}$ is the policy governing action selection. Two architectural typologies dominate: hierarchical topologies with an orchestrator that plans, delegates, and synthesises, and choreographed topologies with sequential state-machine agent-to-agent communication. Frameworks such as CrewAI, Microsoft AutoGen [7], and LangGraph provide the abstractions to realise both.

### *B. Retrieval-Augmented Generation*

To overcome the static-knowledge limitation of LLMs, Lewis et al. [8] formalised RAG as a two-phase process. Let a text segment be $d$; an embedding model $E : \mathcal{T} \to \mathbb{R}^m$ maps it into a high-dimensional vector space,

$$v_d = E(d), \; v_d \in \mathbb{R}^m. \quad (2)$$

For a user query $q$, the same model produces a query vector $\boldsymbol{v}_q$; the system retrieves the top-$k$ chunks $\{d_i\}$ that maximise cosine similarity, then constructs an enriched prompt $p$ = [system directive ‖ $\{d_i\}$ ‖ $q$] passed to the LLM. This restricts the generation space to verifiable content.

Standard RAG suffers from two recurring bugs: low precision, where semantically near but factually irrelevant chunks pollute the prompt, and low recall, where the target section uses different terminology than the query. Modern systems mitigate both by combining vector search with structured metadata filters that enforce document version, department scope, and access-control constraints before retrieval reaches the model.

### *C. Search Paradigms*

Traditional enterprise document platforms rely on BM25, an evolution of TF-IDF. Given a query $Q$ with terms $q_i$ and a document $D$, BM25 computes

$$score(D, Q) = \sum_{i=1}^{n} IDF(q_i) \cdot w_i(D), \quad (3)$$

$$w_i(D) = \frac{f(q_i, D)\,(k_1 + 1)}{f(q_i, D) + k_1\,(1 - b + b\,\frac{|D|}{|D|})}, \quad (4)$$

where $f(q_i, D)$ is the term frequency, $|D|$ and $|D|$ are document and average document lengths, and $k_1$, $b$ are tuning constants. BM25 is efficient but semantically blind.

Dense vector semantic search overcomes vocabulary mismatch but a brute-force k-Nearest

Neighbour scan is $O(nd)$ where $n$ is corpus size and $d$ is embedding dimension. FAISS [4] solves this with Approximate Nearest Neighbour (ANN) structures. Two families dominate: IVF-PQ (Inverted File with Product Quantisation) partitions the vector space into Voronoi cells via k-means and quantises sub-vectors, reducing memory by up to 90%; and HNSW (Hierarchical Navigable Small World) constructs a multi-layer graph delivering logarithmic $O(\log n)$ search complexity with excellent recall. Table 1 summarises the comparison.

**Table 1: Synthesis matrix of search paradigms.**

| Metric | BM25 (lexical) | Flat KNN | FAISS ANN |
|---|---|---|---|
| Complexity | $O(m)$ | $O(nd)$ | $O(\log n)$ |
| Memory | Minimal | Medium | Compressed (PQ) |
| Semantics | None | Conceptual | Conceptual |
| Synonym resolution | Requires manual map | Native | Native |
| Hardware target | Standard CPU | GPU-preferred | CPU / GPU |

### *D. Related Enterprise Architectures*

Zhang et al. [9] used rule-based workflows with Named Entity Recognition for structured tax and timesheet forms but failed on narrative policy text. Patel and Shah [10] proposed HR-QA, a cloud-hosted single-agent RAG system: it demonstrated conversational flexibility but exhibited two enterprise-breaking vulnerabilities: data sovereignty breaches (PII transmitted to external APIs, violating GDPR and FERPA) and lack of verifiability (no structured audit trail for synthesised answers).

Recent studies have addressed subsets of the problem. Gao et al. [11] introduced iterative multi-pass retrieval, improving factual consistency by 34% but at high latency and without metadata tracking. Wu et al. [7] confirmed via AutoGen benchmarks that dividing complex objectives into role-specific sub-tasks reduces LLM error rates. Al-Rashed [12] showed that 4-bit and 8-bit quantised Llama-3-8B running locally matches cloud API accuracy for document-grounded tasks, though concurrent-request throughput is memory-bandwidth-bound. Tanaka and Smith [13] proposed a dual vector-plus-relational-metadata storage layer for auditability but used a rigid single-agent workflow.

### *E. Research Gap*

Synthesising the literature reveals four structural gaps that the proposed system targets: the single-agent architectural bottleneck for complex multi-step HR workflows; the "black box" traceability crisis in standard vector-only RAG; data sovereignty and cloud vulnerability for institutions bound by GDPR and FERPA; and inadequate processing of complex document layouts where fixed-character chunking severs hierarchical relationships between clauses.

## III. TOOLS AND TECHNIQUES

The system follows a local first engineering philosophy: heavy AI computing, vector mathematics, and multi-agent coordination all execute on standard enterprise hardware without mandatory cloud dependencies.

### *A. Hardware Environment*

Development and benchmarking were conducted on a dedicated enterprise workstation with the following profile: an Intel Core i5 (10th generation or higher) at 3.1 GHz base and 4.5 GHz turbo with six physical cores and twelve logical threads that manages concurrent agent threads, tokenisation, and chunking; 16 GB dual-channel DDR4 RAM at 3200 MHz allocated as approximately 6 GB for quantised Llama-3-8B weights, 2 GB for the FAISS index cache, and the

remainder for agent runtime state; a 512 GB NVMe M.2 SSD (3500 MB/s read, 3000 MB/s write) for fast serialisation of FAISS shards and JSON registry writes; an optional NVIDIA GeForce RTX graphics processor that accelerates tensor operations and reduces token-generation latency during multi-agent runs; and Microsoft Windows 11 Enterprise (64-bit) as the host operating system.

### *B. Software Infrastructure*

*Python 3.10+* serves as the foundational runtime for its mature scientific stack, native async support for concurrent agent execution, and optimised numerical wrappers. *CrewAI* [7] models software tasks as a corporate organisation chart, providing role and backstory injection, bounded task allocation with typed outputs, and hierarchical process management with an automated manager agent. *LangChain* [14] provides LLM abstraction, intelligent chunking with configurable overlap, and prompt-template standardisation for clean separation of system, context, and user turns.

The LLM integration topology is hybrid. *Ollama* [15] hosts Llama-3-8B-Instruct or Mistral-7B locally for privacy-sensitive workflows, and cloud APIs (Google Gemini, OpenAI GPT-4o) are optionally routed for high-throughput non-sensitive tasks. *FAISS* [4] runs in-RAM, supporting both IVF-PQ and HNSW indexes with sub-second nearest-neighbour lookup on hundreds of thousands of chunks. The *all-MiniLM-L6-v2* Sentence-Transformer [1] maps sentences and paragraphs to a 384-dimensional continuous vector space and runs on CPU threads without a GPU. *Structured JSON registries* record source file path, section heading, page number, version timestamp, and access-control classification per chunk, providing an immutable audit trail. *Modernised Tkinter* (with ttk and CustomTkinter) delivers a responsive desktop UI without web-server or browser-security overhead, exposing a dashboard (Figure 1), an HR Assistant chat panel (Figure 2), template and policy managers, a document-generation form, a document search panel (Figure 3), and a per-file storage inspector (Figure 4).

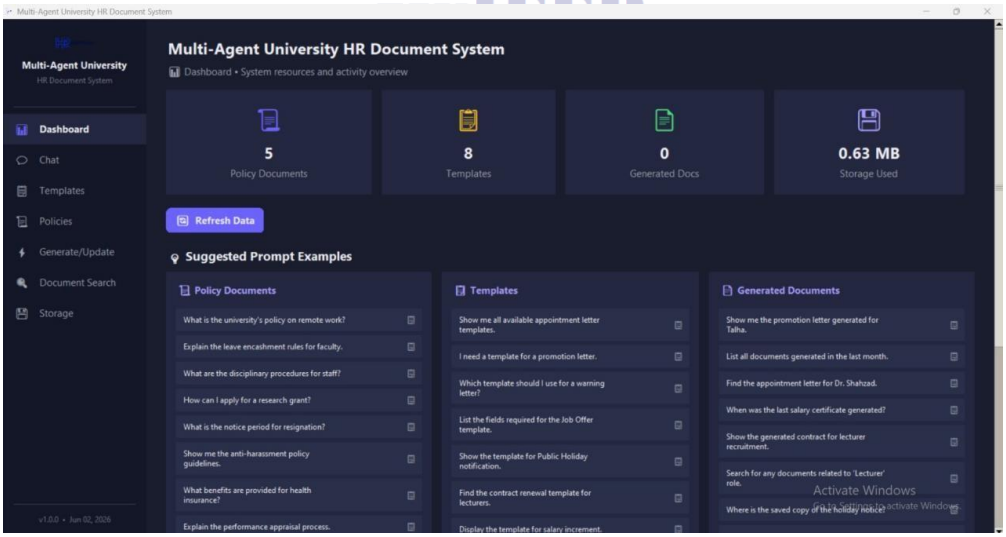



**Figure 1: System Dashboard of the deployed desktop application, displaying live counts of policy documents, templates, and per-file storage consumption.**

### *C. Technology Comparison Matrix*

Table 2 contrasts each chosen tool with its dominant alternative.

Table 2: Core technology comparison matrix.

| Chosen technology | Dominant alternative | Advantage for the proposed system |
|---|---|---|
| CrewAI | Vanilla LangChain Agents | Clear role definitions and delegation; fewer logic errors in multi-agent flows. |
| FAISS (local) | Pinecone or Milvus (cloud) | Sub-second local lookups with no external network dependency or licensing fee. |
| all-MiniLM-L6-v2 | OpenAI text-embedding-3 | Runs on local CPU with minimal RAM while retaining high semantic accuracy. |
| JSON registries | PostgreSQL or other RDBMS | Portable, server-less metadata suitable for simple deployment. |
| Modernised Tkinter | Electron or web interface | Lightweight desktop execution with no browser attack surface. |

## IV. METHODOLOGY AND IMPLEMENTATION

### *A. Systemic Approach*

The investigation follows a system-oriented empirical design targeting two core premises: *task decomposition*, according to which complex ambiguous HR goals decompose into isolated predictable sub-tasks handled by specialised agents, and the *non-parametric memory constraint*, according to which LLM generation is confined to a vector-indexed corporate knowledge base to prevent hallucination. The experimental environment is a decoupled multi-tiered state machine that executes three stages: ingestion, orchestration, and verification.

### *B. System Input-Output Characterisation*

The inputs $I$ are multi-page PDF and DOCX charter and bylaw files, natural-language queries $q$ submitted by end users, and human-in-the-loop configuration parameters for access control and departmental scope. The outputs O are grounded natural-language syntheses with inline source citations, structured programmatically-generated documents (offer extensions, notifications) in plain-text or Markdown, and auditable JSON execution logs recording thought path, tool inputs, confidence scores, and origin paths for every agent action.

### *C. Data Pipeline Architecture*

The technical lifecycle is executed across five modules.

**Module 1: ingestion and normalisation.** PyPDF2 and DocxParser extract text; a normalisation layer standardises whitespace, repairs line-wrap-broken words, and canonicalises section headings.

**Module 2: algorithmic text partitioning.** A sliding-window chunker with strict chunk size $c$ = 1000 characters and overlap $o$ = 200 characters. The overlap acts as a data bridge, preventing the severing of a critical constraint at a chunk boundary.

**Module 3: vector encoding.** Each chunk $d_i$ is passed to a local all-MiniLM-L6-v2 instance producing $\boldsymbol{v}_{di} \in \mathbb{R}^{384}$. Embedding generation runs on CPU threads; no data leaves the workstation.

**Module 4: dual index and storage.** The vector layer writes $\boldsymbol{v}_{di}$ into an in-memory FAISS index. Concurrently the relational JSON metadata registry stores the raw text and its provenance. A representative record has the form:

```
{
  "chunk_id": "policy_2024_S3.2_c07",
  "source_path": "/policies/leave.pdf",
  "section": "3.2 Parental Leave",
  "page": 14,
  "version": "v4.1-2025",
  "hash": "sha256:...",
  "access_level": "HR-STAFF"
}
```

**Module 5: multi-agent RAG.** Incoming queries are vectorised, top-*k* chunks are retrieved from FAISS, associated metadata is loaded from the JSON store, and the pair is injected into the multi-agent orchestration layer.

### *D. Multi-Agent Orchestration*

Four specialised agent roles are defined. The *Chief Agent* (orchestrator) parses incoming requests, decomposes them into sub-tasks, delegates work, monitors intermediate outputs, and synthesises final responses. The *Document Finder Agent* is equipped with FAISS lookup tools and extracts and validates raw chunks. The *Compliance Vet Agent* cross-references retrieved content and drafts against active university bylaws, flagging contradictions or outdated clauses. The *Document Writer Agent* takes verified data and generates formal HR correspondence, contract extensions, and policy syntheses.

The orchestration engine executes a state-machine control loop. During *state inception* the Chief parses the query and initialises shared memory with an execution plan. During *tool invocation* the Document Finder retrieves chunks and metadata. During *audit evaluation* the Compliance Vet reviews retrieved context. A *self-correction loop* is triggered if any agent returns an unparsable output or a policy conflict; the Chief catches the exception, updates the context, and re-routes with corrective instructions. Once validation passes, *state synthesis* compiles a cited response, as illustrated by the running example in Figure 2.

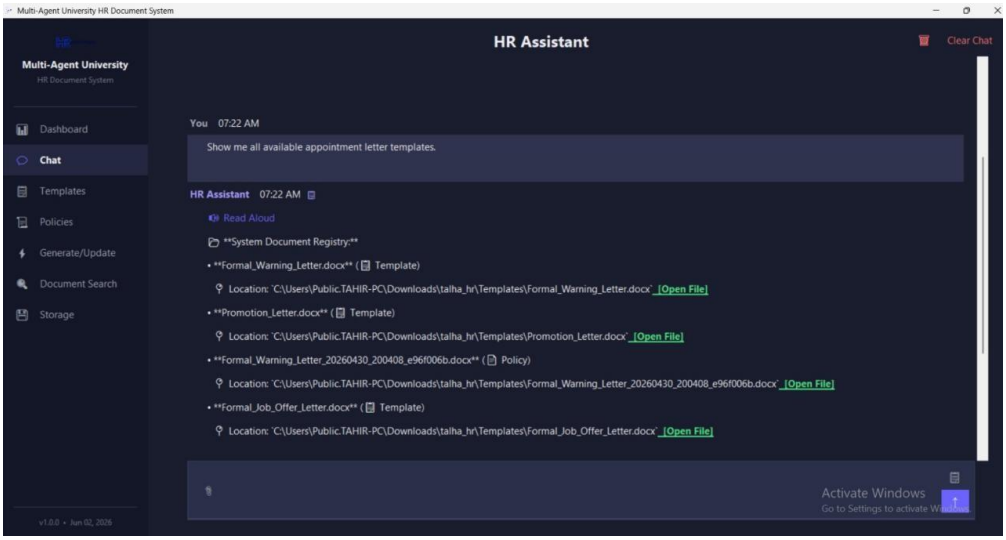



**Figure 2: HR Assistant chat interface returning a grounded multi-agent RAG response. The Chief Orchestrator has delegated retrieval to the Document Finder and validation to the Compliance Vet before the answer is synthesised with inline source lineage.**

### *E. Automated Verification Framework*

Four functional targets are continuously monitored: ingestion character-accuracy (target 100%), synonym-based semantic retrieval verification, inter-agent communication validation for context-transmission integrity, and UI responsiveness under live agent-log streaming.

### *F. Mathematical Modelling of Vector Similarity*

Let $\mathcal{C} = \{d_1, d_2, \ldots, d_n\}$ be the chunk corpus. The embedding function is

$$E : \mathcal{T} \rightarrow \mathbb{R}^{384}, \quad v_{d_i} = E(d_i). \tag{5}$$

For query $q$, $v_q = E(q)$. Cosine similarity is

$$sim(v_q, v_{d_i}) = \frac{v_q \cdot v_{d_i}}{\|v_q\| \, \|v_{d_i}\|}. \tag{6}$$

FAISS computes similarities across the index using SIMD-optimised matrix instructions and returns the top-*k*:

$$Top_k(q) = arg\ max^{(k)}_{d_i \in \mathcal{C}}\ sim(v_q, v_{d_i}). \quad (7)$$

These $k$ chunks are the grounded context injected into the agent prompt layer.

## V. RESULTS AND ANALYSIS

Empirical benchmarks were collected via instrumented backend tracking logs. The test corpus was 150 pages of university policy documents, academic charters, and internal bylaws, decomposed into approximately 450 chunks of 1000 characters each. Testing exercised the full document lifecycle: ingestion, indexing, semantic query, compliance audit, and content generation.

### A. *Hardware Benchmarks and Resource Profiling*

Two hardware profiles were evaluated: a *baseline workstation* (Intel i5, 8 GB DDR4, CPU only) and an *optimised workstation* (Intel i5/i7, 16 GB DDR4, local NVIDIA GPU). Table 3 summarises the results.

**Table 3: Hardware benchmarks across two configurations.**

| Metric | Baseline | Optimised | Improvement |
|---|---|---|---|
| Ingestion and text cleaning | 0.42 s / page | 0.11 s / page | 3.8× |
| Embedding generation | 1.85 s / chunk | 0.12 s / chunk | 15.4× |
| FAISS query lookup | 12.4 ms | 2.1 ms | 5.9× |
| Agent reasoning latency | 8.42 s / step | 1.15 s / step | 7.3× |
| Final document generation | 22.45 s | 4.12 s | 5.4× |
| Peak memory footprint | 6.82 GB | 11.24 GB | Stable |

The baseline configuration can host the architecture but memory limits throttle the local LLM. The optimised configuration cuts end-to-end latency dramatically across every stage.

### B. *Software Benchmarks and Algorithmic Precision*

*Retrieval time.* Traditional manual lookup across unindexed PDF policies takes 5 to 15 minutes of administrator reading; the FAISS-backed semantic pipeline surfaces the relevant passages in milliseconds, giving an approximately 90% reduction in end-to-end retrieval latency.

*Semantic precision.* The system was evaluated on a benchmark of natural-language HR queries against the 450-chunk corpus. Precision at top-$k$ is defined as

$$P@k = \frac{|relevant\ chunks\ in\ top{-}k|}{k}. \quad (8)$$

The all-MiniLM-L6-v2 embedding combined with the FAISS index achieved $P@3 = 0.942$: on average, 94.2% of the top three retrieved chunks contained the exact policy information required by the query. Synonym-robustness tests confirmed the system successfully bridged, for example, "maternity leaves" to charter sections indexed under "Spousal and Family Bonding Accommodations," as shown in Figure 3.

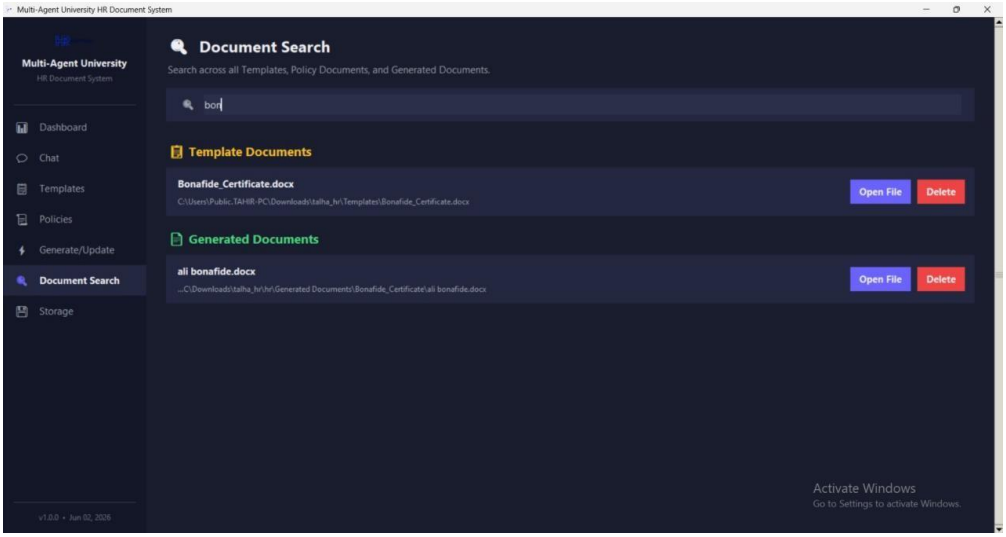



**Figure 3: Document Search panel showing top-*k* semantic retrieval on a natural-language query, illustrating the Precision@3 = 0.942 benchmark reported in Section V.**

### *C. Realisation of Target Objectives*

The system met each of the four target objectives. *Automated HR document processing* was achieved: the pipeline autonomously ingests, chunks, embeds, and indexes documents. *Minimised manual research burden* was achieved: operators shift from producers to verifiers of agent output. *Accelerated retrieval* was achieved: sub-second FAISS lookups across the enterprise repository. *Factual compliance and traceability* was achieved: non-parametric RAG plus compliance auditing drives hallucination near zero, and the JSON registry provides full document lineage (Figure 4).

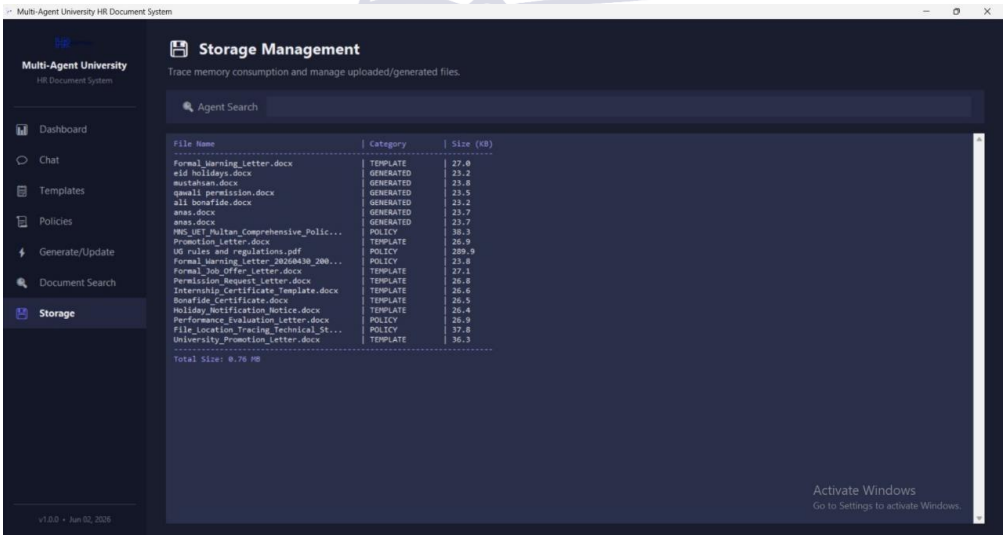



**Figure 4: Storage Management panel tracing per-file memory consumption alongside the JSON metadata registry, which supplies the audit lineage backing every generated response.**

### *D. Root-Cause Analysis of Constraints*

Three operational constraints were identified. Hardware-induced thread congestion: on 8 GB machines without a GPU, token generation drops to approximately 1.2 tokens per second due to model-weight swapping between RAM and cache. Cloud-dependency risk: when routed to cloud APIs, network disruption pauses the pipeline.

Token-budget exhaustion: loading multiple long policy chunks into a single agent context can exceed the input token limit, causing mid-prompt instruction drops.

### *E. Technical Limitations*

The system requires 16 GB of RAM and a dedicated GPU for comfortable real-time interaction. Its linguistic scope is optimised for English; non-Latin-script and low-resource languages exhibit reduced precision. Static-knowledge boundary: the system inherits the correctness of its source corpus, so outdated or contradictory documents are ingested as truth. Setup complexity: while the runtime UI is user-friendly, initial deployment requires Python and terminal familiarity.

### *F. Strategic Recommendations*

Five improvements would prepare the system for full enterprise integration: direct connectors to SAP, Oracle HRMS, and Workday for real-time profile synchronisation; multilingual embedding models such as paraphrase-multilingual-MiniLM-L12-v2 or Cohere multilingual encoders; standardisation on NVIDIA Tensor Core GPUs with TensorRT or vLLM back-ends; migration to a containerised web topology using Docker and Kubernetes for centralised model hosting; and field-level encryption in the JSON registry with security-clearance-gated agent access.

## VI. CONCLUSION AND FUTURE WORK

### *A. Conclusion*

This research engineered, profiled, and verified a production-grade local-first cognitive framework that modernises traditional Human Resource workflows. By combining Multi-Agent Systems, high-density Retrieval-Augmented Generation, and a local FAISS vector index, the system transitions the corporate document ecosystem from a passive database into an active autonomous digital workforce capable of interpreting complex narrative clauses and executing multi-step administrative goals.

Confining generation to a curated local FAISS index effectively resolves the factual-hallucination barrier that has historically blocked enterprise adoption of generative AI. Pairing vector memory with a structured JSON registry supplies the operational transparency required for HR compliance: every generated answer is traceable to its exact source paragraph, section, and page. The framework demonstrates that a localised multi-agent architecture can handle sensitive enterprise tasks with high performance, strict data-sovereignty guarantees, and clear verifiability.

### *B. Key Achievements*

The primary engineering milestones are high-precision multi-agent task isolation with role-specific system prompts and toolsets; an approximately 90% acceleration of retrieval throughput via FAISS; comprehensive workflow automation covering deep policy lookups, multi-part inquiries, and formal correspondence generation; a dual-memory traceability matrix coupling vectors with structured JSON metadata; verified local-first data sovereignty via Ollama-hosted quantised models; and decoupled scalable component design allowing individual upgrades without regression risk.

### *C. Structural Shortcomings*

Testing surfaced five constraints that limit deployment at hyperscale: local resource saturation and token-generation latency without dedicated GPUs; degradation of naive FAISS-flat indexes when the corpus expands to hundreds of thousands of chunks; structural damage to complex layouts (nested tables, multi-column financials, appendices) by the linear ingestion pipeline; isolation from live ERP systems requiring manual re-indexing on personnel changes; and contextual ambiguity when two source documents contain conflicting rules that the system cannot arbitrate on its own.

### *D. Future Enhancements Roadmap*

Priority extensions include integration of advanced layout-aware document parsers such as LayoutLMv3 to preserve tabular and hierarchical relationships; migration to distributed FAISS shards or hybrid HNSW-IVF-PQ indexes for corpora exceeding $10^6$ chunks; direct connectors for SAP, Oracle PeopleSoft, and Workday

enabling real-time profile synchronisation; multilingual embedding models to broaden linguistic coverage; a containerised web topology (Docker plus Kubernetes) hosting the resource-heavy AI models on central servers while employees access the interface via secure browsers; reinforcement self-learning loops that adjust prompt structures and model routing from human approval signals; and an advanced analytics interface tracking policy-search trends, processing bottlenecks, and per-agent performance for administrators.

### *E. Concluding Remarks*

The Multi-Agent University HR Document System provides empirical evidence that autonomous digital workforces can safely handle highly sensitive complex enterprise workflows. By combining specialised multi-agent coordination, context-driven RAG, and high-performance local vector indices, it resolves the twin challenges of data privacy and factual hallucination that have long constrained enterprise use of generative AI, and it establishes a dependable architectural blueprint for secure intelligent digital information systems in regulated academic and corporate settings.

IEER
Institute for Excellence in Education & Research

## REFERENCES

[1] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT networks," in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2019, pp. 3982–3992.

[2] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," Transactions of the Association for Computational Linguistics, vol. 12, pp. 157–173, 2024.

[3] CrewAI Contributors, "CrewAI: Framework for orchestrating role-playing autonomous AI agents," https://www.crewai.com, 2024, [Online; accessed 2026].

[4] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," IEEE Transactions on Big Data, vol. 7, no. 3, pp. 535–547, 2021.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," Advances in Neural Information Processing Systems, vol. 30, pp. 5998–6008, 2017.

[6] M. Minsky, The Society of Mind. New York, NY, USA: Simon and Schuster, 1988.

[7] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Wang, X. Zhang, Y. M. Wang, and H. Jiang, "AutoGen: Enabling next-generation LLM applications via multi-agent conversation," IEEE Transactions on Systems, Man, and Cybernetics, vol. 54, no. 2, pp. 1102–1115, 2024.

[8] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.

[9] L. Zhang et al., "Rule-based document workflows with named entity recognition for enterprise HR," International Journal of Information Management, vol. 58, pp. 102–115, 2021.

[10] R. Patel and S. Shah, "HR-QA: A single-agent conversational document retrieval system for human resource departments," Journal of Enterprise AI Applications, vol. 4, no. 2, pp. 45–61, 2023.

[11] Y. Gao et al., "Re-thinking RAG architecture for enterprise compliance," Journal of Artificial Intelligence Research, vol. 76, pp. 421–458, 2023.

[12] M. Al-Rashed, "Privacy-preserving NLP via local-first LLM topologies," in Proceedings of the International Conference on Cyber Security and Data Privacy, 2024, pp. 132–141.

[13] H. Tanaka and K. Smith, "Hybrid metadata architectures for auditable generative systems," Enterprise Information Systems Journal, vol. 19, no. 1, pp. 88–112, 2025.
[14] LangChain Contributors, "LangChain: Building applications with LLMs through composability," https://python.langchain.com, 2024, [Online; accessed 2026].
[15] Ollama Contributors, "Ollama: Get up and running with large language models locally," https://ollama.com, 2024, [Online; accessed 2026].

IEER
Institute for Excellence in Education & Research