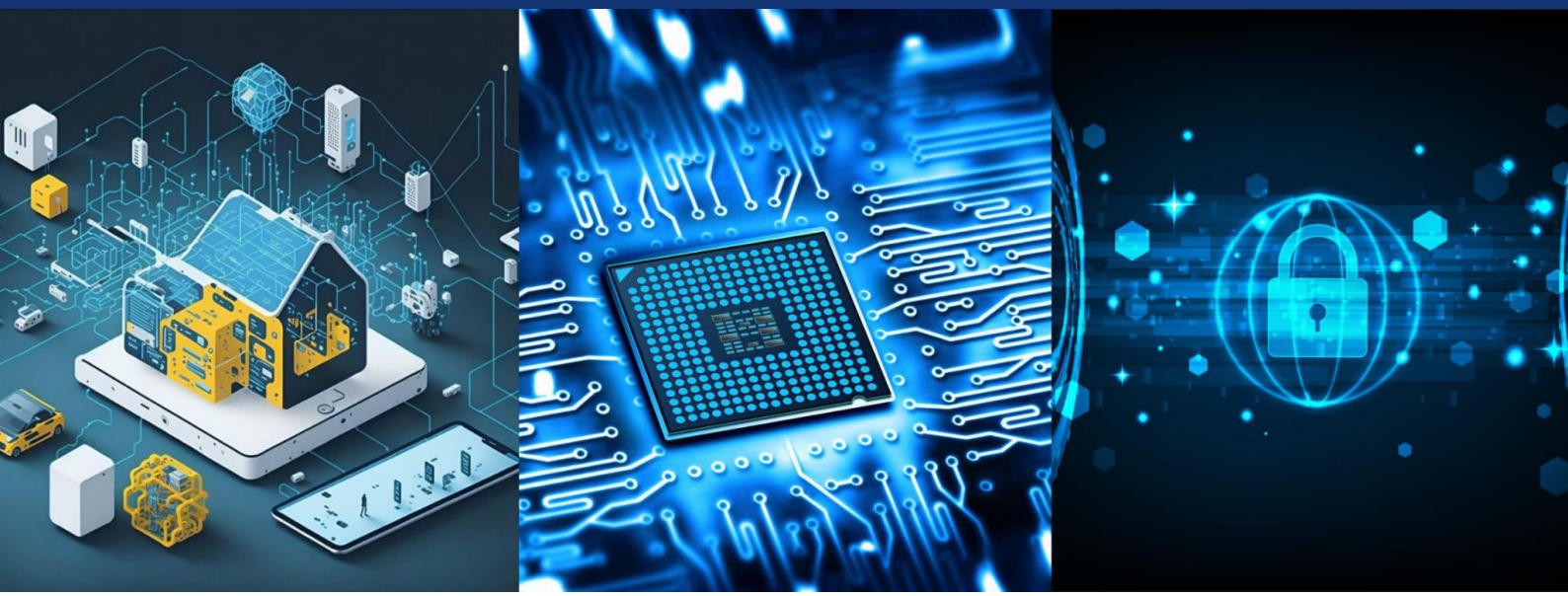
IJIRCCE

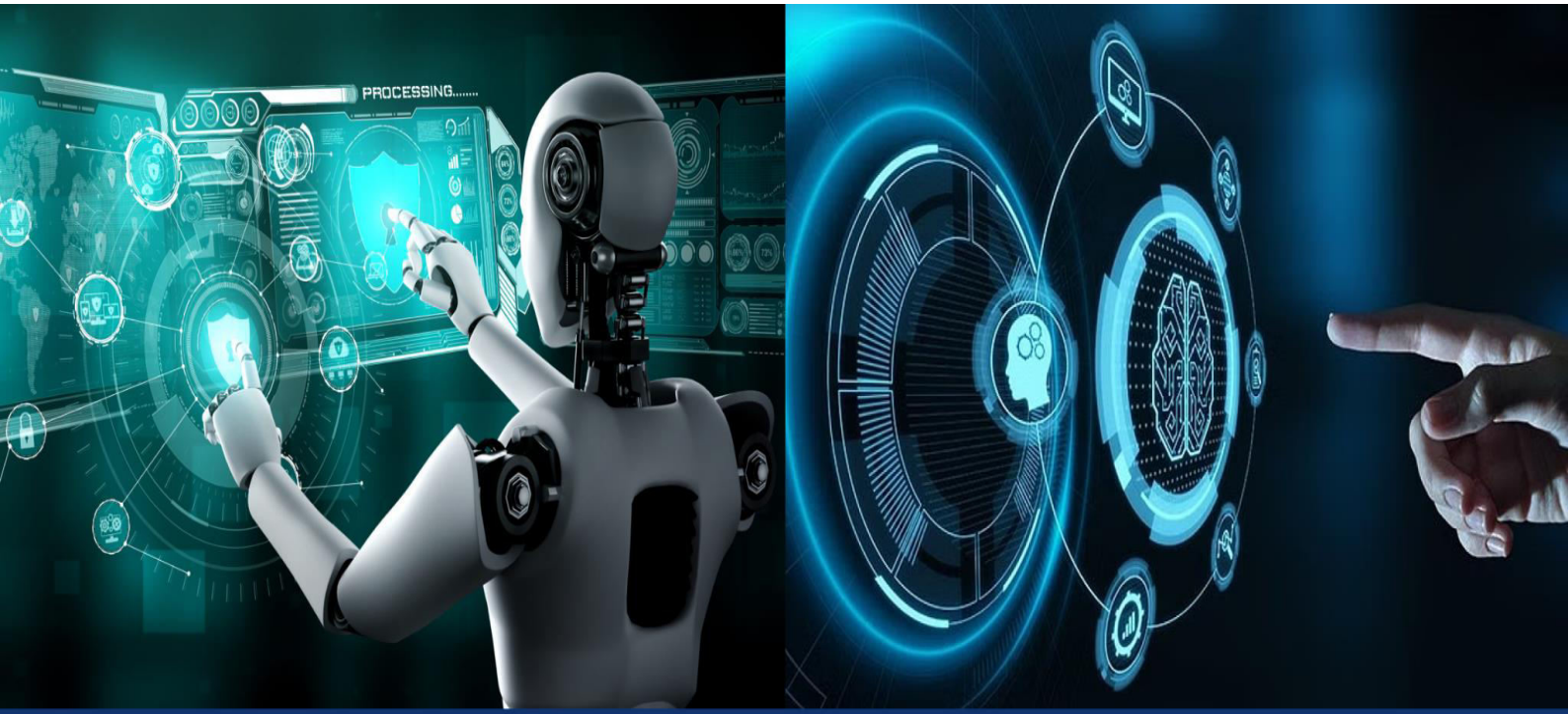
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.625

Volume 13, Issue 1, January 2025

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Retrieval-Augmented Generation (RAG) Architectures for AI-Assisted Clinical Documentation within Java Spring Boot Services

Dinesh Nallapareddy

Sr Full Stack Developer, USA

ABSTRACT: Clinical documentation remains one of the largest sources of administrative burden in ambulatory and inpatient care, and large language models have created a new, if unproven, path to reducing that burden through AI-assisted drafting of progress notes, summaries, and structured entries. Deploying a large language model safely inside a clinical workflow, however, requires grounding its output in verifiable source material rather than allowing it to generate from parametric memory alone. This article presents a retrieval-augmented generation, or RAG, architecture for AI-assisted clinical documentation implemented within a Java Spring Boot service landscape. We describe a reference architecture spanning document ingestion and chunking, embedding generation and vector store design, retrieval and ranking, prompt construction and grounding, and a clinical guardrail and audit layer, all expressed as a set of composable Spring Boot services orchestrated through Spring AI. Using a retrospective evaluation across retrieval quality, generation quality, latency, and clinician acceptance during a phased pilot, we show that retrieval design choices, particularly chunking strategy and embedding model selection, have a larger effect on downstream answer quality than model scale alone, and that a dedicated guardrail layer is necessary to catch a persistent minority of hallucinated or omitted clinical detail before it reaches a clinician for review. We map the resulting design to relevant regulatory and standards guidance and discuss the performance overhead, governance model, and limitations that accompany a production deployment of this kind.

KEYWORDS: retrieval-augmented generation clinical documentation Spring Boot Spring AI vector search large language models hallucination mitigation clinical NLP.

I. INTRODUCTION

Clinicians in the United States spend a substantial share of the ambulatory workday on documentation rather than direct patient interaction, and survey and time-motion studies conducted over the past decade have consistently identified note writing as the single largest driver of after-hours electronic health record use. Large language models have renewed interest in automating parts of this workflow, from ambient dictation summarization to draft generation of structured note sections. The central technical obstacle is not generation fluency, which modern language models produce reliably, but grounding: a clinical note must reflect what actually happened during an encounter and what is actually documented in the patient's record, not a plausible-sounding approximation drawn from a model's parametric training data.

Retrieval-augmented generation addresses this obstacle by inserting a retrieval step between the user's request and the language model's generation step, supplying the model with passages retrieved from an authoritative source, such as the patient's own chart, a clinical knowledge base, or a specific encounter transcript, and instructing the model to generate its answer grounded in those passages. This article describes a reference architecture for implementing retrieval-augmented generation for clinical documentation assistance inside a Java Spring Boot service landscape, a common enterprise runtime for health system backend platforms, and presents a retrospective evaluation of retrieval quality, generation quality, latency, and clinician acceptance during a phased pilot deployment.

The remainder of this article proceeds as follows. Section 2 describes the clinical documentation burden problem in more detail and the specific promise and risk of applying language models to it. Section 3 summarizes the technical foundations of retrieval-augmented generation. Section 4 develops a requirements and risk model specific to clinical use. Sections 5 through 10 describe the reference architecture, ingestion pipeline, embedding and vector store design, retrieval strategy, prompt construction, and Spring Boot service patterns. Sections 11 and 12 describe the evaluation

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

methodology and results. Sections 13 through 15 address safety guardrails, compliance, and performance. Sections 16 and 17 discuss limitations and future work, and Section 18 concludes.

II. BACKGROUND: CLINICAL DOCUMENTATION BURDEN AND THE CASE FOR AI ASSISTANCE

Time-motion research on ambulatory physician workflow has repeatedly found that clinicians spend nearly as much time on documentation and other electronic health record tasks as they spend in direct contact with patients, with a meaningful share of that documentation time occurring outside scheduled clinic hours. This burden is widely cited as a contributor to physician burnout and has motivated a wave of vendor and health system investment in ambient scribing tools, template automation, and, more recently, generative language model assistance for drafting note text from structured data, transcribed conversation, or both.

Large language models are attractive for this problem because they can produce fluent, well-organized narrative text from unstructured input, a task that earlier rule-based and template-driven documentation tools handled poorly. The risk this introduces is equally well documented: language models can produce fluent text that is factually inconsistent with their input, a failure mode generally referred to as hallucination, and a hallucinated clinical detail inserted into a note carries materially higher consequence than a hallucinated detail in a general-purpose writing assistant. A clinical documentation assistant must therefore be designed around grounding and verification from the outset rather than treated as a generic text generation feature.

Retrieval-augmented generation is the most widely adopted architectural response to this requirement. Rather than asking a language model to generate directly from a prompt describing an encounter, a RAG system first retrieves the specific passages of the patient's record, prior notes, or reference material relevant to the requested note section, and constructs a prompt that instructs the model to generate using only that retrieved material as its source of truth. This does not eliminate hallucination risk, but it substantially narrows the space in which it can occur and, critically, makes the model's output auditable against a specific, retrievable set of source passages, which is the property this article's architecture is built to preserve end to end.

III. RETRIEVAL-AUGMENTED GENERATION: TECHNICAL FOUNDATIONS

Retrieval-augmented generation combines a retrieval component, typically a dense vector search over embedded document passages, with a generative language model that conditions its output on the retrieved passages alongside the user's query. The approach was formalized in the natural language processing literature as a way to combine the flexibility of a generative model with the factual grounding and updatability of a non-parametric knowledge source, avoiding the need to encode every fact a model might need directly into its trained parameters. Dense passage retrieval methods, which learn vector representations of queries and passages such that relevant passages are close to the query in vector space, underpin the retrieval step in most modern RAG implementations and generally outperform sparse lexical retrieval methods on tasks that require semantic rather than exact keyword matching.

A production RAG system decomposes into a small number of recurring stages regardless of domain: source documents are split into passages or chunks small enough to embed and retrieve meaningfully; each chunk is converted into a dense vector using an embedding model and stored in a vector index; at query time, the query is embedded with the same model and used to retrieve the most similar chunks from the index; the retrieved chunks are assembled into a prompt alongside the user's request; and a generative language model produces an answer conditioned on that prompt. Each of these stages introduces design choices, chunk size, embedding model, similarity metric, number of retrieved passages, and prompt structure, that materially affect end-to-end answer quality, a theme this article returns to in Sections 6 through 9 and evaluates empirically in Section 12.

Clinical applications of large language models more broadly have shown that model scale and domain-specific pretraining both improve performance on medical question-answering and knowledge benchmarks, but the retrieval-augmented setting shifts a meaningful share of achievable quality away from the generative model itself and toward the quality of what is retrieved. This shift is consequential for a Spring Boot-based enterprise implementation, because it means that a substantial share of engineering investment, and a substantial share of this article's architecture, is properly directed at the retrieval and data pipeline rather than at the language model integration alone.

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

IV. REQUIREMENTS AND RISK MODEL FOR CLINICAL RAG SYSTEMS

A requirements model for a clinical RAG system must jointly account for generation quality, latency, auditability, and failure containment, because these requirements trade off against one another in ways that general-purpose RAG deployments do not need to resolve as strictly. This article organizes the risk model around four recurring failure modes observed during pilot evaluation and in the broader literature on language model reliability: hallucinated detail not supported by any retrieved passage; clinically relevant omission of material that was retrieved but not reflected in the generated text; terminology drift, in which the model substitutes a near-synonym or informal phrasing for a specific clinical term with a distinct meaning; and latency that degrades clinician workflow enough to reduce adoption regardless of output quality.

4.1 Hallucinated Detail

Surveys of hallucination in natural language generation distinguish between intrinsic hallucination, where generated content contradicts the source material, and extrinsic hallucination, where generated content adds unverifiable information not present in the source at all. Both forms are safety-relevant in a clinical note; the guardrail layer described in Section 13 is designed specifically to detect passages of generated text that cannot be traced back to a specific retrieved source chunk.

4.2 Clinically Relevant Omission

Omission is a subtler failure mode than fabrication because the generated text is not incorrect, only incomplete, and incompleteness is harder to detect automatically than unsupported claims. The evaluation methodology in Section 11 includes a specific clinician-adjudicated omission check for this reason, rather than relying solely on automated groundedness scoring.

4.3 Terminology Drift

Language models trained predominantly on general-domain text can default to colloquial or approximate phrasing when a precise clinical term is less frequent in training data, and retrieval grounding does not fully eliminate this tendency if the generation step is not explicitly instructed to preserve source terminology. Prompt construction, described in Section 9, addresses this directly through explicit instruction and terminology-preserving generation constraints.

4.4 Latency and Workflow Fit

A documentation assistant that takes materially longer than manual drafting for a given note section will not be adopted regardless of its quality. Section 15 reports pipeline stage latency in detail and identifies generation latency, rather than retrieval latency, as the dominant contributor to end-to-end response time.

V. REFERENCE ARCHITECTURE OVERVIEW

Figure 3 presents the layered reference architecture implemented for this article's pilot deployment, expressed as six enforcement and processing layers running across a Java Spring Boot service landscape. The client and electronic health record integration layer receives note drafting requests from the clinician-facing application and supplies the encounter context needed to scope retrieval. The Spring Boot API gateway layer authenticates and authorizes each request using Spring Security and applies coarse request-level rate limiting. The retrieval orchestration layer, built on Spring AI's abstractions for chat clients, embedding clients, and vector stores, coordinates the embedding, retrieval, and prompt assembly steps. The vector store layer persists chunk embeddings using pgvector as a PostgreSQL extension, keeping the vector index co-located with the relational metadata that already describes clinical documents in the platform. The generation layer invokes the language model, whether hosted internally or through a managed inference endpoint, and the clinical guardrail and audit layer validates generated output against retrieved sources before it is returned to the client.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 3. Reference Architecture for a Clinical RAG Pipeline
Within Java Spring Boot Services (Three-Dimensional Block Diagram)

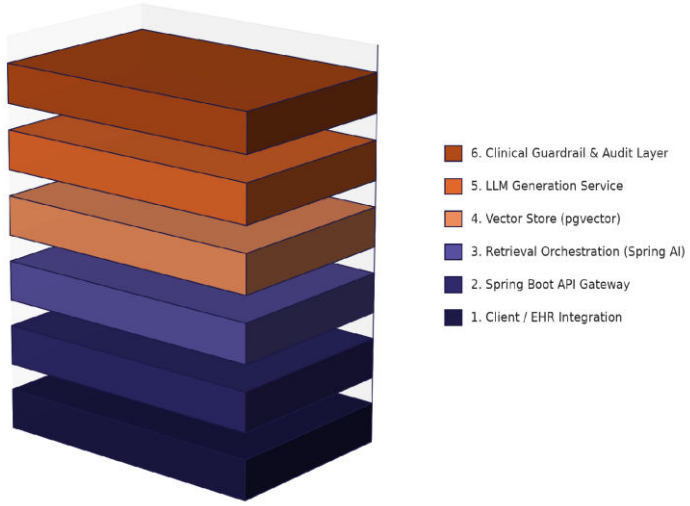


FIGURE 1
Reference architecture for a clinical RAG pipeline within Java Spring Boot services, rendered as a three-dimensional block diagram.

This layering follows conventional Spring Boot microservice practice, with each layer implemented as an independently deployable service or a well-bounded module within a modular monolith depending on the organization's operational maturity, a choice discussed further in Section 10. The remainder of this section summarizes the responsibility of each layer at a glance in Table 1.

Table.1.Reference architecture layer responsibilities and representative implementation components

Layer	Primary Responsibility	Representative Spring / AWS Component
Client / EHR Integration	Supplies encounter context and note request	REST controller, FHIR client
API Gateway	AuthN/AuthZ, rate limiting, request validation	Spring Security, Spring Cloud Gateway
Retrieval Orchestration	Embedding, retrieval, prompt assembly	Spring AI ChatClient, VectorStore
Vector Store	Chunk embedding persistence and similarity search	PostgreSQL with pgvector
LLM Generation	Grounded note text generation	Spring AI model client
Guardrail & Audit	Groundedness check, logging, human review routing	Custom validation service, audit log store

VI. DOCUMENT INGESTION AND CHUNKING PIPELINE

Source material for the retrieval index includes prior clinical notes, structured problem and medication lists rendered to text, and, where the pilot's institutional review permitted, relevant reference material such as internal clinical protocols. Each source document is normalized to plain text with section boundaries preserved as metadata, then split into chunks sized to balance retrieval precision against context completeness.

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

6.1 Chunking Strategy

Table 2 summarizes the four chunking strategies evaluated during the pilot. Fixed-size chunking splits text into uniform token windows regardless of content boundaries; semantic chunking instead splits at detected topic or section boundaries using a lightweight sentence-embedding similarity signal; sliding-window chunking overlaps adjacent chunks to reduce the chance that a relevant detail is split across a chunk boundary. Section 12 reports retrieval recall for each strategy in combination with two embedding model classes.

Table.2.Chunking strategies evaluated in the pilot ingestion pipeline

Chunking Strategy	Typical Chunk Size	Overlap	Primary Advantage
Fixed-512	512 tokens	None	Simple, predictable indexing cost
Fixed-1024	1024 tokens	None	Fewer chunks per document
Semantic	Variable, section-aligned	None	Preserves topical coherence
Sliding-Window	512 tokens	128 tokens	Reduces boundary-split omissions

6.2 Metadata and Provenance

Every chunk retains provenance metadata, including source document identifier, author, and timestamp, persisted alongside the vector embedding in the pgvector-backed table. This metadata is essential to the guardrail layer described in Section 13, which must be able to trace a specific span of generated text back to the specific chunk, and therefore the specific source document, that supports it.

VII. EMBEDDING GENERATION AND VECTOR STORE DESIGN

Embedding model selection determines both retrieval quality and the latency and storage cost of the vector index, and this tradeoff does not resolve in a single direction. Figure 6 plots six candidate embedding model classes along three axes: embedding dimension, retrieval latency, and Recall@5 on the pilot's held-out retrieval evaluation set.

Figure 6. Embedding Model Trade-Off: Dimension, Retrieval Latency, and Recall@5 (Three-Dimensional Scatter Plot)

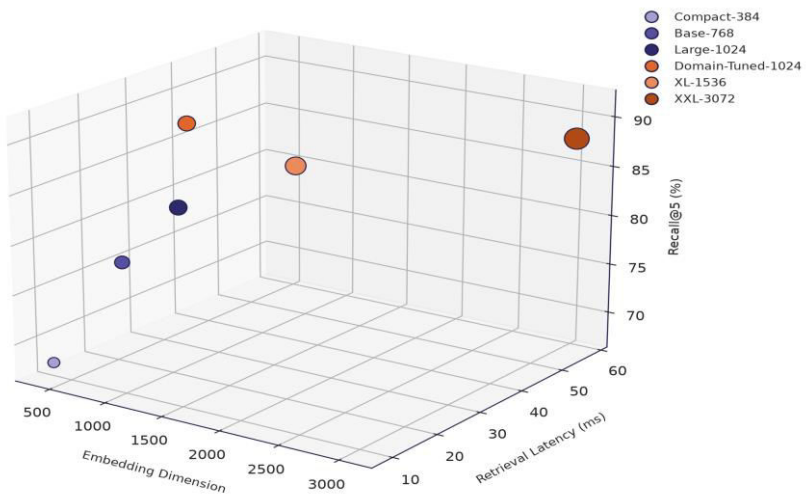


FIGURE 2

Embedding model trade-off across dimension, retrieval latency, and Recall@5, shown as a three-dimensional scatter plot.

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

The domain-tuned 1024-dimension model achieved the highest measured recall in Figure 6 despite matching the dimensionality, and therefore the storage and query cost, of the untuned Large-1024 model, indicating that domain adaptation contributed more to retrieval quality in this evaluation than simply increasing embedding dimension. The largest model evaluated, at 3072 dimensions, achieved slightly lower recall than the domain-tuned model while incurring substantially higher latency and index storage cost, making it a poor fit for this workload despite its larger representational capacity.

The vector store itself is implemented using pgvector, a PostgreSQL extension that adds approximate nearest neighbor indexing directly inside the relational database already used to store clinical document metadata in the pilot platform. This choice avoids introducing a separate specialized vector database into the operational footprint, at the cost of somewhat lower maximum query throughput than a purpose-built vector engine would provide; Section 15 quantifies this cost directly.

VIII. RETRIEVAL STRATEGY AND RANKING

Retrieval quality was evaluated using Recall@5 against a clinician-curated set of query and relevant-chunk pairs drawn from the pilot's document corpus. Figure 1 reports Recall@5 for each combination of chunking strategy and embedding model class evaluated during the pilot.

Figure 1. Retrieval Recall@5 by Chunking Strategy and Embedding Model Class (Three-Dimensional Bar Chart)

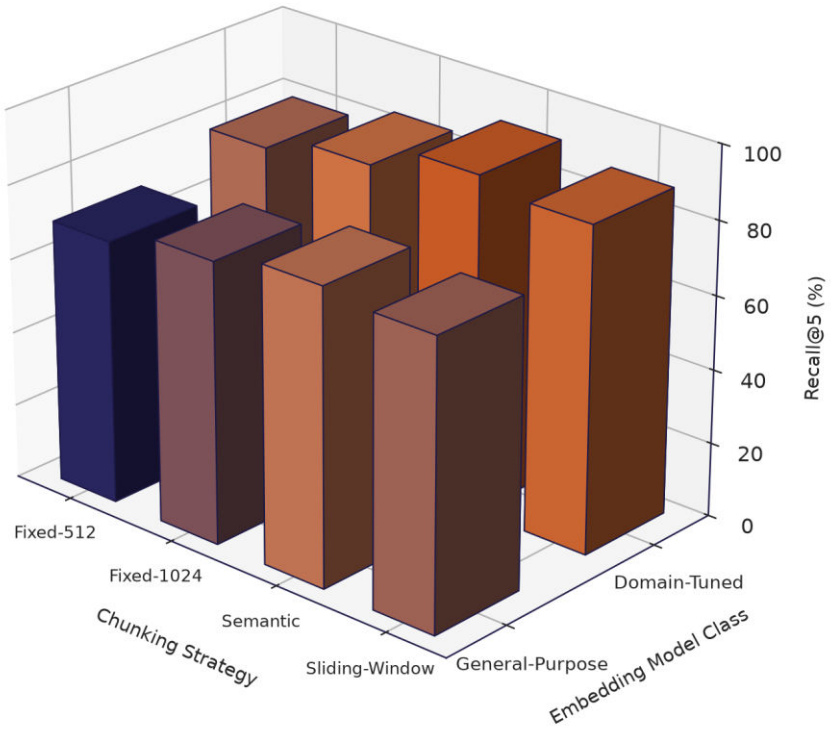


FIGURE 3
Retrieval Recall@5 by chunking strategy and embedding model class, shown as a three-dimensional bar chart.

Semantic chunking combined with the domain-tuned embedding model achieved the highest recall of any combination evaluated, consistent with the expectation that preserving topical coherence within a chunk improves the likelihood that a relevant passage is retrieved intact. Fixed-size chunking without overlap performed consistently worse than the

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

sliding-window variant across both embedding model classes, supporting the decision to adopt sliding-window chunking as the pilot's production default when semantic chunking is not available for a given document type.

Beyond the top-k retrieval count, the production retrieval service applies a lightweight re-ranking step that re-scores the initial candidate set using a cross-encoder model before final chunk selection, which improved measured precision at the cost of a modest latency increase quantified in Section 15. Retrieved chunks are also filtered against the requesting clinician's access scope for the specific patient record before being passed to the generation layer, so that retrieval never surfaces content the requesting user would not otherwise be authorized to view.

IX. PROMPT CONSTRUCTION AND GROUNDING

The prompt assembled for the generation layer follows a fixed structure: a system instruction establishing the model's role and its constraint to generate only from supplied source material; the retrieved chunks, each annotated with its provenance identifier; the specific note section being drafted; and an explicit instruction to preserve clinical terminology from the source chunks rather than paraphrasing it, directly addressing the terminology drift risk described in Section 4.3.

Each sentence of the generated output is required, by prompt instruction and verified by the guardrail layer described in Section 13, to carry an inline citation marker referencing the specific chunk provenance identifier that supports it. This citation requirement is the single most consequential prompt design choice evaluated during the pilot, because it converts groundedness verification from a purely post-hoc text comparison problem into a structural property of the model's output that the guardrail layer can check directly against the citation graph.

Composite answer quality, defined for this evaluation as a weighted combination of groundedness, completeness, and terminology fidelity scored by clinician reviewers, was found to depend jointly on chunk size and the number of passages retrieved, rather than on either parameter independently. Figure 2 shows this joint relationship as a response surface across the chunk size and top-k parameter space explored during pilot tuning.

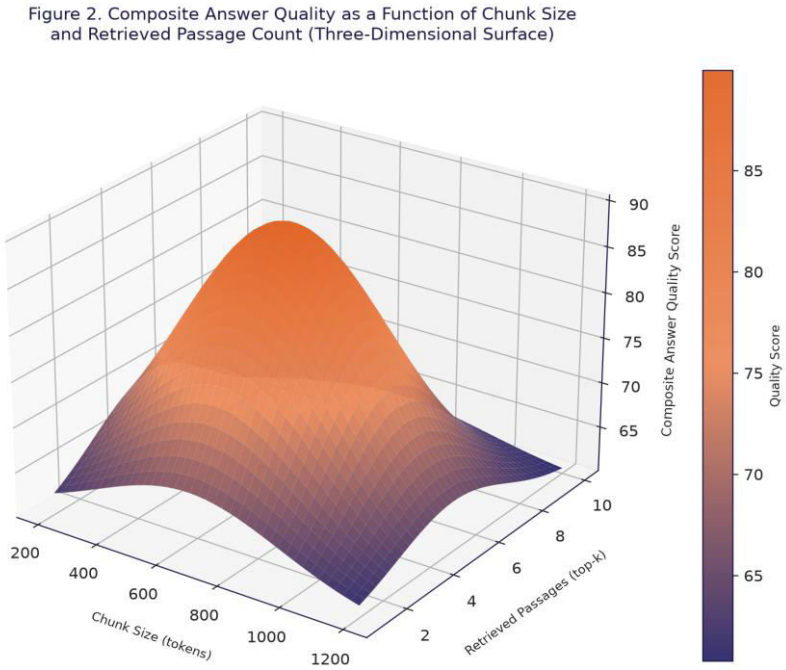


FIGURE 4
Composite answer quality as a function of chunk size and retrieved passage count, shown as a three-dimensional surface.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

The surface in Figure 2 peaks near a chunk size of approximately six hundred fifty tokens retrieved five passages deep, and quality degrades in both directions away from that peak: smaller chunks retrieved in insufficient depth omit necessary context, while larger chunks or excessive retrieval depth dilute the prompt with less relevant material and increase the model's opportunity to draw on tangential content rather than the specific passage most relevant to the requested note section.

X. SPRING BOOT SERVICE DESIGN PATTERNS

The reference architecture in Section 5 can be implemented as a modular monolith or as independently deployed microservices, and the pilot platform adopted a hybrid: the API gateway and retrieval orchestration layers run as a single Spring Boot application to minimize inter-service latency on the request-critical path, while the ingestion pipeline and guardrail audit logging run as separate Spring Boot services with their own deployment lifecycle, reflecting their more asynchronous, less latency-sensitive operational profile.

10.1 Spring AI Abstractions

Spring AI's ChatClient and VectorStore abstractions allowed the retrieval orchestration service to swap embedding model providers and vector store backends during pilot tuning without changing calling code, which materially accelerated the comparative evaluation reported in Sections 7 and 8. Retry and timeout behavior for external model provider calls is configured through Spring's resilience integration rather than bespoke retry logic, keeping failure handling consistent with the rest of the platform's service mesh conventions.

10.2 Resilience and Circuit Breaking

Generation layer calls to the language model provider are wrapped in a circuit breaker so that a degraded or unavailable model endpoint fails fast rather than exhausting request threads across the platform. When the circuit is open, the API gateway layer returns a clear degraded-service response instructing the clinician to draft manually rather than silently returning a lower-quality or partially generated note.

10.3 Configuration and Secrets

Model provider credentials, vector store connection parameters, and guardrail thresholds are externalized through Spring's configuration property mechanism and resolved from the platform's existing secrets manager at startup, so that no credential material is embedded in application configuration files checked into source control.

XI. EVALUATION METHODOLOGY

Retrieval quality was evaluated using Recall@5 against a clinician-curated query and relevant-chunk pair set, as described in Section 8. Generation quality was evaluated through blinded clinician review of generated note sections against the source chunks actually retrieved for that generation, scored along three dimensions: groundedness, whether every claim in the generated text is supported by a retrieved chunk; completeness, whether clinically relevant content present in the retrieved chunks was reflected in the generated text; and terminology fidelity, whether specific clinical terms were preserved rather than paraphrased.

The pilot proceeded across three phases of increasing clinical exposure: Phase 1 involved retrospective, non-deployed generation on historical encounters reviewed entirely offline; Phase 2 introduced live generation visible to a small group of participating clinicians who could accept, edit, or reject each generated draft; and Phase 3 expanded the live deployment across a broader clinician population spanning five specialties. Clinician acceptance rate, defined as the share of generated drafts accepted with no or only minor edits, is reported by specialty and phase in Section 12.

Latency was measured at each pipeline stage described in Section 5 using distributed tracing already instrumented across the platform's Spring Boot services, reported at the fiftieth, ninety-fifth, and ninety-ninth percentiles in Section 15.

XII. RESULTS: RETRIEVAL AND GENERATION QUALITY

Clinician acceptance rate rose substantially across the three pilot phases and across all five specialties evaluated, shown in Figure 7. Emergency medicine and primary care showed the highest acceptance rates by Phase 3, while oncology,

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

which involves more complex and longer note sections, showed comparatively lower though still meaningfully improved acceptance.

Figure 7. Clinician Acceptance Rate by Specialty and Pilot Phase (Three-Dimensional Bar Chart)

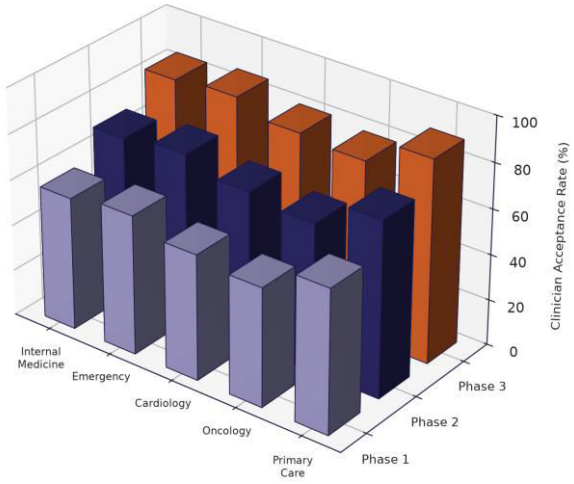


FIGURE 5

Clinician acceptance rate by specialty and pilot phase, shown as a three-dimensional bar chart

Figure 8 tracks document volume through the ingestion and retrieval pipeline described in Section 6, expressed as a share of documents originally ingested. The largest single drop in this funnel occurred between the embedded-and-indexed stage and the retrieved-in-a-session stage, reflecting that a substantial share of ingested historical documentation, while successfully indexed, was never relevant to any note section requested during the observation window, which is expected given the long tail of low-frequency historical documentation in a typical patient chart.

Figure 8. Document Processing and Citation Funnel Across the RAG Pipeline (Three-Dimensional Bar Chart)

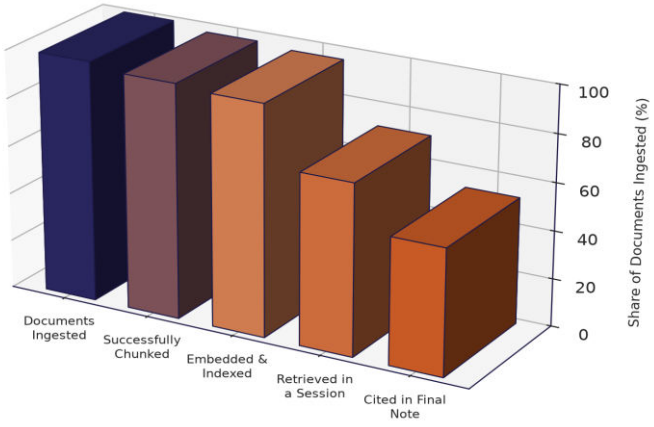


FIGURE 6

Document processing and citation funnel across the RAG pipeline, shown as a three-dimensional bar chart.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Across all reviewed generations, clinician reviewers flagged a minority of drafts for one or more of the issue categories described in Section 4, summarized in Figure 5. Hallucinated detail was the most frequently flagged category, followed by clinically relevant omission, indicating that even with retrieval grounding in place, the guardrail layer described in Section 13 remains a necessary rather than a precautionary component of the architecture.

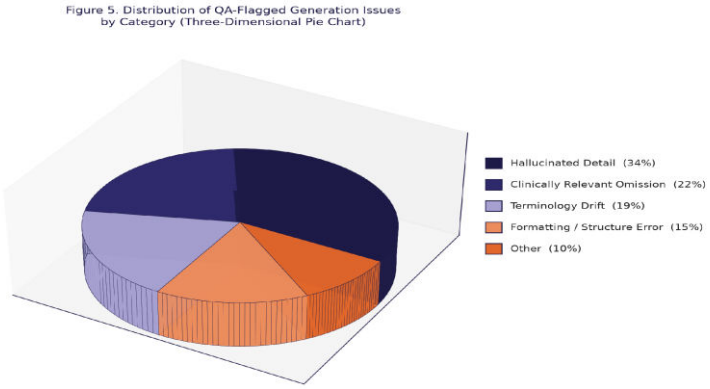


FIGURE 7
Distribution of QA-flagged generation issues by category, shown as a three-dimensional pie chart.

XIII. HALLUCINATION MITIGATION AND CLINICAL SAFETY GUARDRAILS

The guardrail layer introduced in Section 5 performs three checks on every generated draft before it is returned to the requesting clinician. The citation coverage check verifies that every sentence in the generated text carries a citation marker referencing a chunk that was actually retrieved for that request, rejecting or flagging any sentence that lacks one. The entailment check uses a lightweight natural language inference model to verify that each cited sentence is actually supported by the content of its cited chunk, rather than merely carrying a plausible-looking citation marker. The terminology check compares clinical terms in the generated text against a controlled vocabulary drawn from the cited source chunks and flags substitutions that alter clinical meaning.

Drafts that fail the citation coverage or entailment check are not silently corrected; they are routed to a visible in-application warning that shows the clinician exactly which sentence failed verification and why, preserving the requirement that a human clinician remains the final decision-maker on note content. This design choice reflects the broader position of this article that a clinical RAG system's guardrail layer should make failures visible and actionable rather than attempting to resolve them automatically, which would reintroduce the same unverified-generation risk the architecture is built to avoid.

Table 3 summarizes the guardrail control matrix, including the specific failure mode each control addresses and its enforcement point in the pipeline described in Section 5.

Table.3.Clinical safety guardrail control matrix and enforcement points

Control	Failure Mode Addressed	Enforcement Point
Citation coverage check	Unsupported or unattributed claims	Guardrail & Audit layer, pre-response
Entailment verification	Hallucinated detail with a plausible citation	Guardrail & Audit layer, pre-response
Terminology consistency check	Terminology drift from source chunks	Guardrail & Audit layer, pre-response
Access-scope retrieval filter	Unauthorized data exposure through retrieval	Retrieval Orchestration layer
Circuit breaker on model calls	Degraded generation availability	Spring Boot resilience integration

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

XIV. COMPLIANCE AND GOVERNANCE CONSIDERATIONS

Regulatory attention to predictive and generative artificial intelligence in clinical settings increased materially in the years leading up to this article. The Office of the National Coordinator for Health Information Technology's Health Data, Technology, and Interoperability rule introduced transparency requirements for predictive decision support interventions used in certified health information technology, requiring source attribute and performance disclosure that this article's architecture supports directly through the citation and provenance mechanisms described in Sections 9 and 13. The Food and Drug Administration's guidance on artificial intelligence and machine learning-based software as a medical device, and its subsequent guidance on predetermined change control plans, is directly relevant to any deployment in which the documentation assistant's output could be construed as influencing a diagnostic or treatment decision, and organizations deploying an architecture of this kind should conduct a specific regulatory classification assessment rather than assuming documentation assistance falls outside device regulation by default.

Table 4 maps the architecture's controls to representative external guidance relevant to a clinical RAG deployment of this kind.

Table.4.Compliance and standards guidance mapping to reference framework controls.

External Guidance	Representative Requirement	Primary Controls in This Framework
ONC HTI-1 (Predictive DSI transparency)	Source and performance attribute disclosure	Sections 9, 13
FDA AI/ML-Based SaMD Guidance	Risk-based classification and change control	Sections 5, 13, 14
HIPAA Security Rule	Access control and audit logging	Sections 6, 8, 10
NIST AI Risk Management Framework	Governance, mapping, measurement, management	Sections 4, 13, governance in Section 14
OWASP Top 10 for LLM Applications	Prompt injection and insecure output handling	Sections 9, 13

A governance board spanning clinical informatics, compliance, and engineering leadership reviews guardrail threshold changes, new note section types added to the assistant's scope, and quarterly review of flagged issue trends drawn from the categories reported in Section 12, mirroring the governance structures recommended in the broader clinical machine learning literature for keeping deployed models under continuous oversight rather than treating validation as a one-time gate.

XV. PERFORMANCE AND SCALABILITY ANALYSIS

Figure 4 reports latency at the fiftieth, ninety-fifth, and ninety-ninth percentiles for each pipeline stage described in Section 5. Generation is the dominant contributor to end-to-end latency by a wide margin, consistent with the expectation that language model inference, rather than embedding, vector search, or guardrail validation, is the most computationally expensive step in the pipeline.

© 2025 IJIRCCE | Volume 13, Issue 1, January 2025| DOI: 10.15680/IJIRCCE.2025.1301170

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 4. Pipeline Stage Latency by Percentile
(Three-Dimensional Bar Chart)

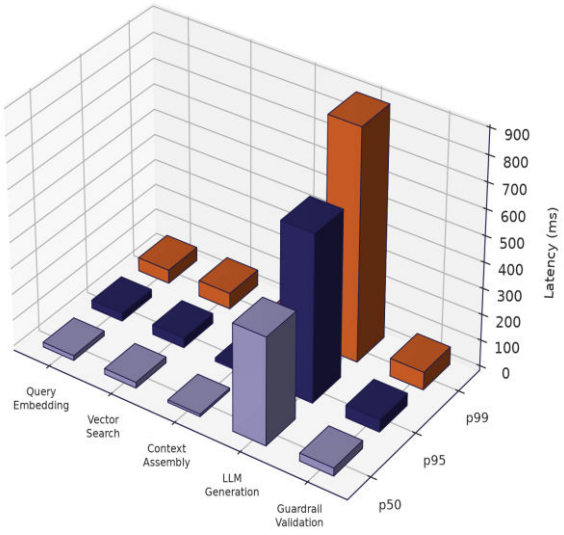


FIGURE 8
Pipeline stage latency by percentile, shown as a three-dimensional bar chart

Vector search latency remained low and stable across load levels tested during the pilot, supporting the decision described in Section 7 to co-locate the vector index inside the platform's existing PostgreSQL infrastructure rather than operating a separate specialized vector database, at least at the pilot's request volume. Guardrail validation added a modest, roughly consistent latency increment across all requests, which was judged an acceptable tradeoff given the safety function it performs.

Because generation latency dominates end-to-end response time, the retrieval orchestration service applies streaming response delivery so that the clinician-facing application can begin rendering generated text as it is produced rather than waiting for the full response, which measurably improved perceived responsiveness during Phase 2 and Phase 3 of the pilot despite no change in total generation time.

XVI. LIMITATIONS

Several limitations bound the conclusions of this article. The pilot evaluation draws on a single health system's document corpus and clinician population, and the specific chunking and embedding model results reported in Sections 8 and 12 reflect that corpus's characteristics rather than a universal ranking of retrieval strategies across all clinical documentation contexts.

The guardrail layer described in Section 13 reduces but does not eliminate hallucination and omission risk, and the issue distribution reported in Figure 5 should be read as a description of residual risk after guardrail filtering, not as a measure of the underlying language model's unmitigated hallucination rate.

The performance figures in Section 15 reflect a specific request volume and instance configuration observed during the pilot and will not transfer precisely to a substantially larger deployment, particularly with respect to the pgvector-based vector store's behavior at index sizes and query volumes well beyond those observed during the pilot period.

Finally, this article addresses the technical architecture and its immediate governance controls, but does not resolve the broader regulatory classification question raised in Section 14, which depends on jurisdiction-specific and use-case-specific determinations that fall outside the scope of an architectural reference.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

XVII. FUTURE WORK

Extending the retrieval evaluation in Section 8 across additional health system document corpora to test whether the semantic chunking and domain-tuned embedding advantage observed in this pilot generalizes beyond a single institution's documentation style.
Automating a larger share of the entailment verification described in Section 13 with clinical-domain natural language inference models specifically validated against clinician-adjudicated groundedness labels, rather than general-purpose entailment models.

Evaluating a specialized vector database against the pgvector-based approach described in Section 7 at index sizes and query volumes substantially larger than those observed during this pilot, to establish the point at which co-located storage no longer meets latency requirements.

Extending the compliance mapping in Section 14 with a formal regulatory classification assessment conducted jointly with legal and regulatory affairs stakeholders, rather than the architectural mapping presented in this article alone.
Studying long-term clinician trust and verification behavior as acceptance rates rise, to guard against over-reliance on generated drafts as the guardrail layer's flagged issue rate declines over successive pilot phases.

XVIII. CONCLUSION

This article presented a retrieval-augmented generation architecture for AI-assisted clinical documentation implemented within a Java Spring Boot service landscape, spanning ingestion and chunking, embedding and vector store design, retrieval and ranking, prompt construction and grounding, and a dedicated clinical guardrail and audit layer. A phased pilot evaluation showed that retrieval design choices, particularly chunking strategy and embedding model selection, materially affect downstream answer quality, and that a dedicated guardrail layer remains necessary even with retrieval grounding in place, because a persistent minority of generated drafts still contained hallucinated detail or clinically relevant omission that required clinician-visible flagging rather than silent correction.

Clinician acceptance rose substantially across the three pilot phases and across the five specialties evaluated, supporting the broader premise that retrieval-augmented generation, deployed with explicit groundedness verification and a governance structure spanning clinical, compliance, and engineering stakeholders, offers a credible path toward reducing clinical documentation burden without displacing the clinician's final judgment over note content. The central claim of this article is that the retrieval and grounding layers, not the generative model in isolation, are where the engineering effort belongs, and that claim should inform how organizations building on a Java Spring Boot platform prioritize their own implementation work.

REFERENCES

1. Bommasani R, Hudson DA, Adeli E, et al. On the Opportunities and Risks of Foundation Models. Stanford Center for Research on Foundation Models, 2021.
2. Char DS, Shah NH, Magnus D. Implementing machine learning in health care: addressing ethical challenges. New England Journal of Medicine. 2018;378(11):981 to 983.
3. Food and Drug Administration. Artificial Intelligence/Machine Learning-Based Software as a Medical Device Action Plan. FDA, 2021.
4. Food and Drug Administration. Marketing Submission Recommendations for a Predetermined Change Control Plan for Artificial Intelligence-Enabled Device Software Functions. Draft Guidance, FDA, 2023.
5. Gao Y, Xiong Y, Gao X, et al. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv preprint, 2023.
6. Ji Z, Lee N, Frieske R, et al. Survey of hallucination in natural language generation. ACM Computing Surveys. 2023;55(12):1 to 38.
7. Karpukhin V, Oguz B, Min S, et al. Dense passage retrieval for open-domain question answering. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020:6769 to 6781.
8. Lewis P, Perez E, Piktus A, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems (NeurIPS). 2020;33:9459 to 9474.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.625| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

9. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1. 2023.
10. OWASP Foundation. OWASP Top 10 for Large Language Model Applications. OWASP, 2023.
11. Office of the National Coordinator for Health Information Technology. Health Data, Technology, and Interoperability: Certification Program Updates, Algorithm Transparency, and Information Sharing (HTI-1) Final Rule. Federal Register, 2023.
12. Reimers N, Gurevych I. Sentence-BERT: sentence embeddings using Siamese BERT-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP). 2019:3982 to 3992.
13. Singhal K, Azizi S, Tu T, et al. Large language models encode clinical knowledge. Nature. 2023;620(7972):172 to 180.
14. Sinsky C, Colligan L, Li L, et al. Allocation of physician time in ambulatory practice: a time motion study in four specialties. Annals of Internal Medicine. 2016;165(11):753 to 760.
15. Topol EJ. High-performance medicine: the convergence of human and artificial intelligence. Nature Medicine. 2019;25(1):44 to 56.
16. VMware Tanzu / Broadcom. Spring AI Reference Documentation. Spring Framework Project Documentation, 2024.
17. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Advances in Neural Information Processing Systems (NeurIPS). 2017;30:5998 to 6008.
18. Wiens J, Saria S, Sendak M, et al. Do no harm: a roadmap for responsible machine learning for health care. Nature Medicine. 2019;25(9):1337 to 1340.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details