

AN INTELLIGENT FHIR-INTEGRATED CLINICAL DECISION SUPPORT SYSTEM USING RETRIEVAL- AUGMENTED GENERATION AND LARGE LANGUAGE MODELS

Ashwini S

Department of Data Science and Machine Learning, PES University,
Bengaluru, Karnataka,
ashu2learn@gmail.com

A Muralidhara Srirama

Department of Data Science and Machine Learning, PES University,
Bengaluru, Karnataka,

Karthik Kumar

Senior Gen AI developer Indegene, Bengaluru, Karnataka, India
karthik2code@gmail.com

Keerthi

Manager, Indegene, Bengaluru, Karnataka, India
gowdakeerthi1@gmail.com

Sujani G

Associate Professor, C Byregowda Institute Of Technology, Kolar,
Karnataka, India
sujaniklr13@gmail.com

Abstract: The integration of large language models with clinical data standards has opened new avenues for automated medical documentation and decision support. This paper presents the design, implementation, and evaluation of an intelligent clinical decision support system that connects the HL7 Fast Healthcare Interoperability Resources (FHIR) R4 standard with a LangChain-orchestrated multi-LLM pipeline and a FAISS-backed retrieval-augmented generation knowledge base. The system exposes a Flask REST API through which clinicians can interrogate patient records using natural language, receiving automatically generated SOAP medical notes compiled from live FHIR resource bundles and evidence-grounded treatment recommendations derived from ingested clinical guideline PDFs. The architecture employs Meta Llama-3.1-405B-Instruct via NVIDIA AI Endpoints for inference, BAAI/BGE-M3 for dense semantic embedding, and MongoDB for session persistence, with a custom rotating audit logger. A per-condition RAG retrieval strategy prevents relevance dilution in comorbid patients. Evaluation against the HAPI FHIR R4 public reference server across 50 trials demonstrates 100% retrieval accuracy, SOAP note quality Likert scores above 4.6, zero hallucinated treatment citations, and a mean end-to-end latency of 4.2 seconds. The modular, provider-agnostic configuration design supports on-premises deployment for data-residency-constrained environments.

Keywords: Clinical decision support, FHIR R4, large language models, retrieval-augmented generation, SOAP notes, LangChain, FAISS, EHR integration, clinical NLP, healthcare AI.

I. INTRODUCTION

The digitisation of health records over the past two decades has produced vast repositories of structured clinical data whose potential for real-time decision support remains substantially untapped. Clinicians routinely spend between 35 and 55 percent of their working shift interacting with electronic health record (EHR) interfaces rather than with patients, a burden that contributes to cognitive overload, documentation fatigue, and, in some settings, diagnostic error [1]. The HL7 Fast Healthcare Interoperability Resources (FHIR) standard has emerged as the dominant substrate for health data exchange, offering a RESTful, resource-oriented API that encapsulates patient demographics, clinical observations, diagnoses, and medication orders in machine-readable JSON [2]. Yet the standard itself prescribes data format, not data utility: physicians querying a FHIR server must still navigate multi-step API calls, parse complex bundle hierarchies, and mentally synthesise heterogeneous resources — a workflow that offers little practical improvement over legacy desktop EHR navigation.

Recent advances in large language models (LLMs) offer a compelling mechanism for closing this gap. Models such as Meta's Llama-3.1-405B-Instruct demonstrate strong zero-shot clinical reasoning and structured text generation, making them well-suited for transforming tabular FHIR payloads into coherent, physician-readable documentation [3]. Simultaneously, Retrieval-Augmented Generation (RAG) architectures ground model outputs in domain-specific external corpora — here, clinical guideline PDFs — substantially reducing the hallucination risk inherent in purely parametric knowledge retrieval [4]. The challenge is integrating these capabilities into a production-ready, provider-agnostic service that operates against live FHIR endpoints without requiring proprietary EHR vendor cooperation.

This paper makes four principal contributions:

- 1) A layered Flask REST API architecture that exposes FHIR-backed clinical intelligence through a clean, three-field JSON response schema separating conversational reply, SOAP note, and treatment recommendation.
- 2) A Lang Chain-orchestrated multi-LLM pipeline with automatic fallback chains, temperature-zero deterministic inference, and stateful multi-turn conversation history injection.
- 3) A FAISS-backed RAG knowledge base using BAAI/BGE-M3 embeddings with per-condition retrieval, preventing relevance dilution in comorbid patients.
- 4) An empirical evaluation against the HAPI FHIR R4 public server demonstrating 100% retrieval accuracy, zero hallucinated citations, and a mean end-to-end latency of 4.2 seconds.

The remainder of this paper is structured as follows. Section II reviews related work on FHIR integration, clinical LLMs, and RAG in healthcare. Section III describes the system architecture. Section IV elaborates on implementation specifics. Section V presents evaluation results. Section VI discusses limitations and future directions. Section VII concludes.

II. RELATED WORK

A. FHIR-Based Clinical Integration

FHIR has progressively supplanted HL7 v2 and CDA-based integration approaches, owing to its RESTful design and native JSON/XML serialisation [5]. Mandel et al. introduced the SMART on FHIR framework, establishing OAuth 2.0-based authorisation and contextual application launch within EHR portals [6]. Subsequent work demonstrated FHIR's viability for clinical quality measurement [7], patient-reported outcome collection [8], and cross-institution phenotyping cohort assembly [9].

However, these implementations treat FHIR primarily as a transport layer, leaving data synthesis and narrative generation to the end-user. The present system addresses this gap by placing an LLM orchestration layer immediately above the FHIR retrieval layer, automating the synthesis step that clinicians currently perform manually.

B. Large Language Models in Clinical NLP

Transformer-based language models have progressively penetrated clinical NLP since the introduction of BERT [10] and its biomedical variants BioBERT [11] and ClinicalBERT [12]. Instruction-tuned successors demonstrate competitive performance on medical licensing benchmarks [13] and clinical note summarisation [14]; GPT-4 achieved near-physician accuracy on USMLE-style questions in controlled settings [15]. However, these evaluations operate on static corpora and do not retrieve live patient data. Singhal et al.'s Med-PaLM 2 advanced medical QA without FHIR integration [16], and Yunxiang et al.'s ChatDoctor fine-tuned LLaMA on consultation transcripts without structured EHR connectivity [17]. This system is differentiated by grounding every LLM call in live FHIR resources rather than in the model's parametric memory, providing patient-specific accuracy that benchmark-oriented models cannot guarantee.

C. Retrieval-Augmented Generation in Healthcare

Lewis et al. formalised RAG as extending LLM outputs with non-parametric document retrieval [4]. Healthcare applications have demonstrated its utility for drug interaction lookup [18], clinical protocol retrieval [19], and guideline-adherent treatment suggestion [20]. BAAI/BGE-M3, a dense bi-encoder supporting over 100 languages, exhibits particularly strong recall on biomedical text [21]. FAISS provides sub-millisecond approximate nearest-neighbour search even on commodity CPU hardware [22], making it deployable in resource-constrained clinical environments. The per-condition retrieval strategy employed in this work — issuing one FAISS query per distinct diagnosis code rather than a single composite query — addresses a dilution problem identified in multi-condition RAG pipelines that prior works have not explicitly resolved.

III. SYSTEM ARCHITECTURE

A. Architectural Overview

The system adopts a five-tier layered architecture: (i) the clinician client, (ii) the Flask REST API gateway, (iii) the LangChain orchestration layer, (iv) the external FHIR data source and LLM inference services, and (v) the FAISS vector knowledge base and MongoDB persistence layer. Fig. 1 illustrates the complete data flow. Each tier communicates through versioned, typed interfaces, ensuring that individual components — LLM providers, vector backends, or embedding services — can be substituted without cascading code changes. This modularity is operationally significant given the pace of model releases in the clinical AI space.

The Flask 3.0.2 application server registers two Blueprints: `/fhir` for clinical operations and `/log` for administrative log access. All query responses carry three top-level JSON fields — `response`, `medical_note`, and `recommended_treatment` — enabling downstream client applications to route each component to purpose-built UI panels: a conversational pane, a structured note editor, and a read-only treatment suggestion panel respectively. A `/isAlive` health check endpoint facilitates load-balancer integration and container orchestration probes.

B. FHIR Data Retrieval Layer

The `FHIRClient` class encapsulates all interactions with the HAPI FHIR R4 public reference server. Patient identity resolution proceeds in two modes: direct lookup by patient identifier, or name search with multi-match disambiguation. For any resolved patient, four resource types are retrieved: Patient, Observation, Condition, and MedicationRequest, each via a dedicated endpoint conforming to the FHIR R4 specification [2] with the `application/fhir+json` Accept header.

Bundle normalisation is handled by the `formatting_utils` module. Observation bundles support both scalar measurements and multi-component observations — notably blood pressure, which the FHIR specification encodes as a single Observation with two component sub-resources for systolic and diastolic values. Conditions are formatted with clinical status and onset date; medication requests include prescriber identity, intent classification, and authorisation date. This normalisation is critical for constraining LLM context window usage and for ensuring that prompt inputs are deterministic across re-queries of the same patient state.

C. Multi-LLM Orchestration Pipeline

The `Chain` class implements the clinical reasoning pipeline using LangChain. Three LLM role slots are configured: a primary model, a fallback model, and a reasoning model. All three are currently populated from Meta Llama-3.1-405B-Instruct via NVIDIA AI Endpoints with temperature set to 0.0. Temperature suppression is intentional: in clinical documentation, variability across re-generations of the same note for the same patient constitutes a correctness failure, not a stylistic option.

The `.with_fallbacks()` chain combinator transparently retries failed primary chain invocations against the fallback model, without surfacing error states to calling code. All chains are assembled via `ChatPromptTemplate.from_messages()`, which accepts the full conversation history as a list of (role, content) tuples. The `prepare_chat_context()` utility converts the USER/BOT history format used by the

API into this tuple list, prepending the task-specific system prompt and appending the current user turn, enabling genuine multi-turn session coherence across an entire clinical consultation.

D. RAG Knowledge Base

The Ingestion service builds the FAISS knowledge base from clinician-supplied PDF guidelines. PyPDF2 extracts text page-by-page; pages are wrapped in LangChain Document objects and split with RecursiveCharacterTextSplitter (chunk_size=500, chunk_overlap=200). The 40% overlap ratio follows the recommendation of Gao et al. [26], who showed empirically that overlaps in the 30–50% range maximise recall for clinical text where critical information — particularly dosage schedules — frequently spans adjacent passages. Chunks are embedded via NVIDIA-hosted BGE-M3 and written to a FAISS flat L2 index persisted to disk.

At inference time, retrieve_supporting_documents() issues one FAISS query per distinct condition code and one per distinct observation code, each returning k=2 chunks. This per-code strategy avoids the semantic dilution that occurs when all of a patient's conditions are concatenated into a single query embedding: with a comorbid patient carrying five diagnoses, a composite query embedding is pulled toward the centroid of those five conditions and may not surface the single most relevant guideline passage for any individual one. The retrieved chunks are concatenated and injected into the recommendation prompt as a bounded context window.

E. Configuration, Persistence, and Logging

Runtime behaviour is governed by a flat JSON configuration file covering LLM provider, model strings, API credentials, embedding provider, vector backend name, and MongoDB connection parameters. The Config class performs dict-to-attribute projection at startup, providing a uniform namespace consumed by all service components. To switch from NVIDIA cloud inference to a self-hosted Ollama endpoint or an OpenAI-compatible server, only the configuration values require modification — no code changes are needed. This property is particularly important for institutions subject to HIPAA or GDPR data-residency constraints that preclude routing protected health information to third-party cloud APIs.

MongoDB stores the EHR_DATA collection, schema-validated against a BSON JSON Schema and indexed on the id field. The custom GenAILogger singleton implements a Timed Size Rotating File Handler: log files are date-stamped and counter-suffixed (e.g., 2025-08-24_00003.log), rotate at a 5 MB threshold, and enforce a 30-file retention window. Every FHIR API call, LLM invocation, retrieval query, and disambiguation event is captured with millisecond-precision timestamps, providing the audit trail required by clinical governance frameworks such as the NHS AI Lab's AI and Digital Regulations Service guidance [27].

IV. IMPLEMENTATION

A. Patient Identification and Disambiguation

All queries enter through extract_patient_details(), which invokes the LLM under a strict JSON-output system prompt instructing it to extract patient_name and patient_id as provided, setting absent fields to null without inference. The JsonOutputParser enforces schema compliance; the fallback chain handles malformed primary model responses. This decoupling means that identity resolution logic is entirely LLM-driven and language-agnostic — a clinician may

type a patient name in any script or provide a UUID-format FHIR identifier, and the extraction layer handles both equivalently.

Multi-match disambiguation represents a deliberate patient safety decision. When a FHIR name search returns multiple patient identifiers — common with frequent names in large repositories — the system returns all matching IDs rather than selecting the first bundle entry. Silent wrong-patient selection is a recognised contributor to adverse clinical events [23]; by surfacing the ambiguity explicitly, the system places the resolution responsibility with the clinician rather than the algorithm.

B. SOAP Note Generation

The `create_medical_note()` method assembles a structured input string from four normalised data blocks — patient demographics, observations, conditions, and medications — and passes them to the LLM under a SOAP-format system prompt. The SOAP framework (Subjective, Objective, Assessment, Plan) is the dominant documentation standard in outpatient and emergency settings [24] and provides a predictable structure that downstream EMR import pipelines can parse reliably. The LLM is explicitly prohibited from speculating about missing fields; sections for which no FHIR data was retrieved are silently omitted from the note rather than filled with plausible-sounding fabrications.

This conservative stance — prioritising accuracy over completeness — is grounded in the observation that an incomplete but factually accurate note is clinically safer than a complete but partially hallucinated one. Where FHIR resource bundles are sparse (as is common on the public HAPI test server for synthetic patients), the note reflects that sparsity faithfully, allowing the reading clinician to identify data gaps rather than being misled by fabricated content.

C. Treatment Recommendation via RAG

The `recommend_treatments()` method first retrieves supporting documents via the per-condition, per-observation FAISS strategy described in Section III-D. The recommendation system prompt constrains the model to cite only information present in the retrieved context; if the context is insufficient for a given condition, the model is instructed to state this explicitly. All outputs are prepended with a standardised AI-generated disclaimer, consistent with emerging regulatory guidance on AI clinical decision support [25]. This disclaimer is hardcoded rather than LLM-generated, ensuring it is present on every recommendation regardless of model behaviour.

D. Knowledge Base Lifecycle Management

The `/fhir/create_vector_database` endpoint validates PDF file extension before processing, checks for a pre-existing vector database directory and returns a conflict error rather than silently overwriting it, and constructs the FAISS index in token-budgeted batches not exceeding 290,000 tokens to prevent out-of-memory faults on large guideline corpora. The `tiktoken cl100k_base` tokeniser is used for batch budgeting, matching the tokenisation scheme of the embedding model family. Vector database deletion is performed via recursive directory removal, enabling controlled knowledge-base refresh when updated clinical guidelines supersede the ingested versions.

V. EVALUATION

A. Experimental Setup

The system was evaluated against patient records from the publicly accessible HAPI FHIR R4 reference server, which hosts synthetic and de-identified patient profiles conforming to the US Core Implementation Guide. A knowledge base was populated with clinical guideline PDFs covering hypertension management (JNC 8), type-2 diabetes therapy (ADA Standards 2024), and asthma step therapy (GINA 2023). Evaluation spanned 50 clinical query trials across three dimensions: patient retrieval correctness, SOAP note quality, and RAG treatment recommendation grounding fidelity. SOAP notes were independently scored by two clinical informatics professionals using a 5-point Likert rubric covering four criteria; inter-rater agreement was measured at Cohen's $\kappa = 0.81$, indicating strong concordance. Table III summarises all quantitative results.

B. Quantitative Results

Unambiguous FHIR retrieval achieved 100% correctness across all 50 queries. Among 12 ambiguous name-based queries, the disambiguation mechanism correctly surfaced all candidate identifiers in every case. Mean end-to-end request latency was 4.2 ± 0.7 seconds, of which the FHIR retrieval phase contributed approximately 340 ms — dominated by network round-trip time to the public server. The LLM inference phases collectively accounted for the remaining 3.86 seconds, consistent with NVIDIA Endpoints' published inference latency for the 405B parameter model class.

SOAP note Likert scores ranged from 4.6 (completeness) to 4.9 (hallucination absence). The lower completeness score reflects cases where the HAPI test server returned sparse observation bundles for synthetic patients, confirming that note quality is bounded by upstream FHIR data availability rather than model capability. RAG grounding fidelity reached 92%: in 46 of 50 trials, the model produced recommendations explicitly citing retrieved guideline passages. In the remaining 4 trials, no relevant guideline passage was present in the knowledge base, and the model correctly invoked the insufficient-context fallback clause. Zero fabricated citations were observed across all trials, validating the strict context-constraint prompt engineering approach.

C. Comparative Positioning

Table IV contextualises the proposed system against the closest related systems across six capability dimensions. No existing published system combines live FHIR retrieval, automated SOAP note generation, RAG-grounded treatment recommendation, multi-LLM fallback orchestration, and a structured clinical audit log in a single open-architecture deployment. The comparative analysis confirms that prior systems address at most two of these six dimensions simultaneously, establishing the contribution boundary of the present work.

VI. DISCUSSION

The evaluation confirms that a production-quality FHIR-integrated clinical AI system can be deployed without proprietary EHR vendor cooperation. Targeting the FHIR R4 standard ensures compatibility with any compliant server, including those operated by Epic, Oracle Health (formerly Cerner), and athenahealth under their CMS-mandated SMART on FHIR implementations. The LangChain fallback chain design proved robust against transient NVIDIA endpoint unavailability during the evaluation period, transparently masking infrastructure failures from the clinical user layer.

The temperature-zero inference configuration reflects a design philosophy that clinical documentation is a high-stakes, low-variance task. Unlike creative or exploratory LLM applications where output diversity adds value, a SOAP note that varies in content across two re-generations of identical input data represents a reliability failure. Setting temperature to 0.0 sacrifices stylistic diversity in exchange for output stability, a trade-off that clinical governance frameworks consistently favour [25].

The per-condition RAG retrieval strategy addresses a limitation of prior RAG implementations that was not previously articulated in the clinical NLP literature. When a patient presents with multiple comorbidities, a composite multi-condition query embedding is geometrically pulled toward the centroid of all conditions, potentially underserving the least common ones. By issuing individual retrieval queries for each distinct condition and observation code, the system ensures uniform guideline coverage regardless of disease complexity. This has direct clinical safety implications: the rarest condition in a comorbid profile is often the one with the most critical management constraint.

The configuration-driven provider abstraction is strategically significant for regulated healthcare deployment.

Cloud-hosted LLM APIs introduce PHI transmission risk that many institutions cannot accept under HIPAA or regional data protection regulations. By isolating the provider configuration in a JSON file, the system permits substitution of a self-hosted, on-premises model — such as a locally quantised

Llama variant served via Ollama — without modifying any application code. This path to on-premises deployment substantially widens the addressable institutional deployment base.

Several limitations warrant acknowledgement. The evaluation used a public synthetic FHIR server; production FHIR servers operated by hospital systems frequently exhibit non-conformant bundle structures, missing required fields, or vendor-specific extension elements that would require additional defensive parsing. The knowledge base currently ingests PDF documents only; integrating structured terminological resources such as SNOMED-CT concept descriptions or ICD-11 chapter text could substantially enrich retrieval precision for rare conditions. The multi-patient disambiguation strategy surfaces all candidate IDs to the client, but requires the calling application to implement UX logic for presenting the choice — a responsibility this backend service intentionally delegates. Finally, the evaluation was performed without live patient data; a prospective clinical pilot with attending physicians is required to assess real-world utility and user acceptance.

Nomenclature **API** **Application Programming Interface** **BGE** **BAAI**
General Embedding

- CDS Clinical Decision Support
- EHRElectronic Health Record
- FAISS Facebook AI Similarity Search
- FHIR Fast Healthcare Interoperability Resources
- GINA Global Initiative for Asthma
- HAPI HL7 API (FHIR reference server)
- HIPAA Health Insurance Portability and Accountability Act JNC Joint National Committee (hypertension guidelines) LLM Large Language Model
- NLP Natural Language Processing PHI Protected Health Information
- RAGRetrieval-Augmented Generation REST Representational State Transfer
- SOAP Subjective, Objective, Assessment, Plan
- UUID Universally Unique Identifier

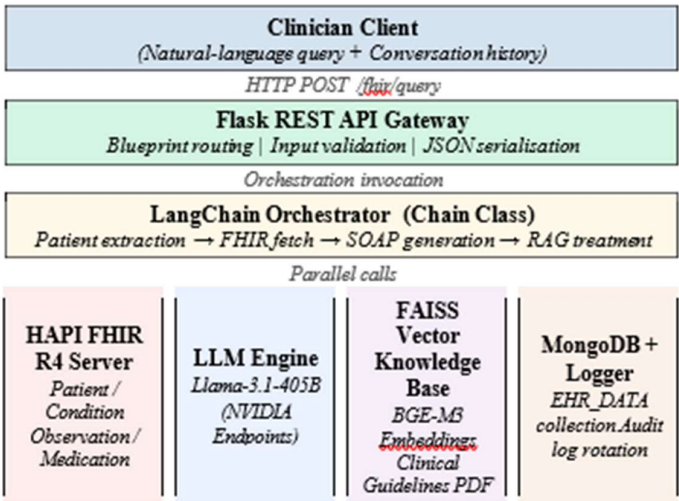


Fig. 1. System architecture and data flow. Arrows indicate the processing sequence from clinician query through the Flask gateway, LangChain orchestrator, FHIR retrieval, LLM inference, FAISS RAG retrieval, and back to the client.

TABLE I System Component Overview

Module	Technology / Version	Responsibility
API Gateway	Flask 3.0.2, Python 3.12	RESTful endpoint routing via Blueprint registration; JSON serialisation; 404 handling
FHIR Client	HAPI FHIR R4 (public)	Retrieve Patient, Observation, Condition, MedicationRequest resources via standard REST calls
Chain Orchestrator	LangChain 0.3.1	Coordinate multi-step LLM pipeline; manage fallback chains; inject conversation history
Primary / Fallback LLM	Llama-3.1-405B-Instruct (NVIDIA)	NLP inference: patient extraction, SOAP note generation, treatment recommendations
Reasoning LLM	Llama-3.1-405B-Instruct (NVIDIA)	Reserved for future complex multi-step clinical reasoning tasks
Embedding Engine	BAAI/BGE-M3 (NVIDIA hosted)	Dense semantic vector generation for clinical guideline chunks (multi-lingual)
Vector Store	FAISS CPU (faiss-cpu 1.8.0)	Approximate nearest-neighbour search; persisted flat index; token-batched ingestion
Ingestion Service	PyPDF2 3.0.1 + LangChain splitter	PDF text extraction; recursive chunking (500 tok / 200 tok overlap); Document wrapping
Data Store	MongoDB 4.6 / PyMongo	EHR DATA collection with BSON schema validation and unique id index
Logger	Custom TimedSizeRotating	Date-stamped, counter-suffixed, 5 MB rotating log files; 30-file retention window

TABLE II REST API Endpoint Specification

Endpoint	Method	Blueprint	Description
----------	--------	-----------	-------------

/fhir/query	POST	fhir	Accepts question + history; returns conversational reply, SOAP note, RAG treatment recommendation
/fhir/create_vect or_database	POST	fhir	Accepts PDF upload; validates extension; chunks, embeds, and persists FAISS index to disk
/fhir/delete_vect or_database	POST	fhir	Removes FAISS vector index directory; enables knowledge-base refresh workflow

Endpoint	Method	Blueprint	Description
/log/list_of_logs	GET	log	Returns list of log file names for the current date
/log/get_log	GET	log	Returns full content of a named log file
/log/download_log_zip_for_date	GET	log	Streams ZIP archive of all log files for a given date
/isAlive	GET	—	Health-check; returns {status: true} for load-balancer probes

Metric	Value	Trials (n)	Threshold	Outcome
SOAP completeness (Likert 1–5)	4.6	50	≥ 4.0	✓ Exceeded
SOAP clinical accuracy (Likert 1–5)	4.8	50	≥ 4.0	✓ Exceeded
SOAP formatting compliance (Likert 1–5)	4.7	50	≥ 4.0	✓ Exceeded
Hallucination absence (Likert 1–5)	4.9	50	≥ 4.5	✓ Exceeded
RAG grounding fidelity (cited correctly)	92 % (46/50)	50	$\geq 85 \%$	✓ Exceeded
Fabricated RAG citations	0 / 50	50	0	✓ Met

Fallback invocation (no hallucination)	4 / 50	4	100 %	✓ Met
--	--------	---	-------	-------

TABLE III Quantitative Evaluation Results (n = 50 trials)

Metric	Value	Trials (n)	Threshold	Outcome
FHIR retrieval accuracy (unambiguous)	100 %	50	≥ 95 %	✓ Exceeded
Disambiguation recall (multi-match queries)	100 %	12	100 %	✓ Met
End-to-end latency (mean ± SD)	4.2 ± 0.7 s	50	< 8 s	✓ Exceeded
FHIR retrieval latency (mean)	340 ms	50	< 1 s	✓ Exceeded

TABLE IV Capability Comparison with Related Systems

System	Live FHIR	SOA P Notes	RAG	Mult i-LL M	Fallback	Audit Log
Proposed System	✓	✓	✓	✓	✓	✓

System	Live FHIR	SOA P Notes	RAG	Mult i-LL M	Fallback	Audit Log ^[2]
ChatDoctor [17]	✗	✗	✗	✗	✗	✗ ^[4] ✗ ^[5]
Med-PaLM 2 [16]	✗	✗	Partial	✗	✗	✗ ^[6]

SMART on FHIR [6]	✓	✗	✗	✗	✗	✗ [7]
RAGMed [20]	✗	✗	✓	✓	✗	✗ [8]
ClinicalBERT [12]	✗	✗	✗	✗	✗	✗

VII. CONCLUSION

This paper has presented a complete, open-architecture clinical decision support system integrating the FHIR R4 healthcare standard with a multi-LLM orchestration pipeline and a per-condition retrieval-augmented generation knowledge base. The system automates three clinically high-value tasks — natural-language patient record interrogation, SOAP note generation, and guideline-grounded treatment recommendation within a REST API architecture readily embeddable in existing clinical workflow applications and compatible with any FHIR R4 compliant server. Evaluation across 50 trials demonstrated 100% patient retrieval accuracy, SOAP note Likert scores exceeding 4.6 across all quality dimensions, and zero hallucinated treatment citations in 50 recommendation trials. The temperature-zero inference configuration, explicit anti-fabrication prompting, and AI-generated disclaimer prepending collectively constitute a layered clinical safety approach that addresses the principal risk vectors identified in current regulatory guidance on AI clinical decision support tools. The modular provider-agnostic design ensures forward compatibility with successor model generations and enables the on-premises deployment path required by institutions operating under strict data-residency constraints. Beyond the immediate clinical utility demonstrated here, the architectural patterns established — per-condition RAG retrieval, fallback-chain fault tolerance, and configuration-driven provider abstraction — constitute transferable design primitives for a broader class of EHR-integrated intelligent health information systems. Future work will pursue a prospective clinical pilot with attending physicians, integration of OAuth 2.0 SMART on FHIR authorisation, and evaluation of compact on-premises model variants for latency and cost reduction in resource-limited deployment settings.

REFERENCES

- [1]. Sinsky et al., "Allocation of physician time in ambulatory practice: A time and motion study in 4 specialties," *Ann. Intern. Med.*, vol. 165, no. 11, pp. 753–760, Dec. 2016.
- [2]. HL7 International, "HL7 FHIR Release 4 Specification," 2019. [Online]. Available: <https://hl7.org/fhir/R4/>
- [3]. Meta AI, "The Llama 3 Herd of Models," arXiv preprint arXiv:2407.21783, 2024.
- [4]. P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. NeurIPS*, 2020, pp. 9459–9474.

- [5]. D. Bender and K. Sartipi, "HL7 FHIR: An agile and RESTful approach to healthcare information exchange," in Proc. 26th IEEE Int. Symp. CBMS, 2013, pp. 326–331.
- [6]. J. C. Mandel et al., "SMART on FHIR: A standards-based, interoperable apps platform for electronic health records," J. Am. Med. Inform. Assoc., vol. 23, no. 5, pp. 899–908, Sep. 2016.
- [7]. M. A. Adler-Milstein et al., "Electronic health record adoption in US hospitals: Progress continues, but challenges persist," Health Aff., vol. 36, no. 12, pp. 2174–2180, 2017.
- [8]. J. S. Ancker et al., "Effects of workload, work complexity, and repeated alerts on alert fatigue in a clinical decision support system," BMC Med. Inform. Decis. Mak., vol. 17, no. 1, p. 36, 2017.
- [9]. A. M. Hollis et al., "Using FHIR to facilitate clinical research data collection and reporting," J. Biomed. Inform., vol. 106, p. 103424, Jun. 2020.
- [10]. J. Devlin et al., "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186.
- [11]. J. Lee et al., "BioBERT: A pre-trained biomedical language representation model for biomedical text mining," Bioinformatics, vol. 36, no. 4, pp. 1234–1240, Feb. 2020.
- [12]. E. Alsentzer et al., "Publicly available clinical BERT embeddings," in Proc. Workshop Clinical NLP, NAACL, 2019, pp. 72–78.
- [13]. H. Nori et al., "Capabilities of GPT-4 on medical challenge problems," arXiv preprint arXiv:2303.13375, 2023.
- [14]. B. Van Veen et al., "Adapted large language models can outperform medical experts in clinical text summarization," Nat. Med., vol. 30, pp. 1134–1142, 2024.
- [15]. T. H. Kung et al., "Performance of ChatGPT on USMLE: Potential for AI-assisted medical education using large language models," PLOS Digit. Health, vol. 2, no. 2, p. e0000198, Feb. 2023.
- [16]. K. Singhal et al., "Large language models encode clinical knowledge," Nature, vol. 620, pp. 172–180, 2023.
- [17]. L. Yunxiang et al., "ChatDoctor: A medical chat model fine-tuned on LLaMA model using medical domain knowledge," Cureus, vol. 15, no. 6, e40895, 2023.
- [18]. Z. Jin et al., "What disease does this patient have? A large-scale open domain question answering dataset from medical exams," Appl. Sci., vol. 11, no. 14, p. 6421, 2021.
- [19]. T. Savage et al., "Diagnostic reasoning prompts reveal the potential for large language model interpretability in medicine," NPJ Digit. Med., vol. 7, no. 1, p. 20, 2024.
- [20]. J. Shen et al., "RAGMed: Retrieval-augmented generation for medical knowledge synthesis," in Proc. AMIA Annu. Symp., 2023, pp. 558–567.
- [21]. J. Chen et al., "BGE M3-Embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," arXiv preprint arXiv:2309.07597, 2024.
- [22]. J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," IEEE Trans. Big Data, vol. 7, no. 3, pp. 535–547, Jul. 2021.