SafeCDS: A Neuro-Symbolic Agentic Framework for Verifiable Prescribing Decisions in Cardiometabolic Multi-Morbidity

Rachamallu Vibhas Reddy* Udit Jayan Pratyaksh SS Sahil Yadav

Department of Artificial Intelligence and Machine Learning
Nitte Meenakshi Institute of Technology, Bengaluru
Faculty Advisor: Dr. Piyush Kumar Pareek
{vibhasreddy, ujayan, pratyakshss, sahilyadav}@nmit.ac.in

*Lead author and corresponding author; led system design, core pipeline implementation, and evaluation. See Author Contributions.

Abstract

Doctors treating a patient with diabetes, kidney disease, and high blood pressure at the same time run into a problem that single-disease guidelines never anticipated: the standard first-line drug for one condition is often the exact drug to avoid in another. Metformin, the default diabetes treatment, can cause fatal lactic acidosis once kidney function drops far enough. ACE inhibitors, the default blood pressure treatment, can trigger acute kidney decline in that same patient. Large language models can read and summarize the relevant guidelines, but they generate text by predicting likely tokens, not by checking facts against rules, so they can miss a contraindication as easily as they can state one. Rule-based clinical decision support avoids that failure mode but cannot read a new guideline PDF or explain its reasoning in plain language. We built SafeCDS to sit between these two failure modes. A locally hosted LLaMA-3.2:3B model retrieves passages from KDIGO, ADA, and ACC/AHA guidelines through a hybrid dense-and-sparse search and proposes a treatment. Before that proposal ever reaches a clinician, it is asserted into an isolated OWL 2 DL ontology and checked by the HermiT reasoner. If the reasoner finds a violation, it hands the exact broken rule back to the model as feedback and the model tries again, inside a state machine that gives up and escalates to a human after a fixed number of attempts. On a six-patient cardiometabolic benchmark, 83.3% of proposals passed on the first try, every failed proposal was corrected on the next attempt, and the reasoner caught 100% of the contraindications by construction, since nothing reaches the output without clearing its check. The whole pipeline runs on a single workstation with no data leaving the machine.

1 Introduction

More than half of adults over sixty live with at least two chronic conditions at once, and that share climbs past 70% after age seventy-five. Among these overlapping conditions, the combination of Type 2 Diabetes Mellitus (T2DM), Chronic Kidney Disease (CKD), and essential hypertension is one of the most dangerous to prescribe for, because the three conditions actively worsen each other. Diabetic nephropathy degrades renal clearance, which changes how antihypertensive drugs are cleared from the body and causes sodium retention, which raises blood pressure, which damages the kidneys further. A patient can enter this loop from any direction.

The trouble is that clinical guidelines were not written with this loop in mind. The American Diabetes Association writes for diabetes. KDIGO writes for kidney disease. The American College of Cardiology and the American Heart Association write for the cardiovascular system. Each guideline is correct on its own terms and can directly contradict another:

- **Metformin in renal failure.** Metformin is the first-line oral drug for T2DM, but it is cleared almost entirely through the kidneys. Once estimated glomerular filtration rate (eGFR) falls below 30 mL/min/1.73 m^2

(CKD stages 4–5), the drug accumulates, interferes with mitochondrial respiration, and can cause fatal lactic acidosis.

- **RAAS blockade in advanced CKD.** ACE inhibitors and ARBs are first-line for hypertension and protect the kidney in early diabetic nephropathy by relieving intraglomerular pressure. In stage 4–5 CKD, though, they can over-dilate the efferent arteriole, dropping GFR further and pushing potassium to dangerous levels.

A general practitioner has roughly ten to fifteen minutes per visit to catch a conflict like this, while also reading lab results, updating the chart, and talking to the patient.

Existing software does not close this gap. Rule-based clinical decision support systems (CDSS) are deterministic and auditable, but every new drug interaction means writing another branch of hand-coded logic, and the rule tree becomes unmanageable as comorbidities stack up. They cannot read a new guideline PDF or explain a decision in language a clinician can question. Large language models solve the opposite half of the problem: they read guideline text fluently and generate a plausible-sounding recommendation, but a next-token predictor has no built-in concept of a hard medical rule. It can state a contraindication correctly nine times and miss it silently the tenth, and there is no way to know which time is which without checking the output against something outside the model.

SafeCDS is our attempt to combine the two halves without inheriting either one's weakness. A neural component reads the guidelines and drafts a treatment plan; a symbolic component formally checks that plan against encoded medical logic before a clinician ever sees it. When the check fails, the system does not just reject the plan — it tells the model exactly which rule it broke and lets the model try again, closing the loop instead of leaving a human to start over from a blank page.

This paper reports the second, fully closed-loop phase of that system. Phase 1 was an open-loop proof of concept: a language model proposed a drug, a reasoner checked it, and if the check failed, the pipeline simply stopped. Phase 2, described here, adds the automated retry loop, isolates the reasoning state between patients, enforces a strict output schema, and validates the whole pipeline against a cardiometabolic patient cohort.

2 Related Work

Retrieval-augmented generation (RAG) is now the standard way to ground a language model in documents it was not trained on, and agentic RAG extends this by letting a model plan multi-step retrieval and tool use rather than answering in one shot [1]. Clinical applications of RAG have grown quickly: multi-agent RAG systems have been proposed for clinical decision support [7], and LLM multi-agent pipelines have targeted safer therapy recommendation specifically [8]. What most of this work shares is a soft notion of safety — a well-grounded prompt and a plausible citation trail — rather than a hard guarantee that a specific unsafe class of prescription cannot be output.

The neuro-symbolic line of work is closer to what SafeCDS attempts. Several recent surveys and position papers argue that pairing a neural model with symbolic verification is the more honest route to trustworthy AI in high-stakes domains [2, 3], and healthcare is a recurring example [4]. Knowledge graphs have been used to ground LLM diagnosis directly, either by feeding graph context into the prompt [5] or by treating the graph as an explicit reasoning assistant [6]. These approaches improve factual grounding but generally use the graph as extra context for the model to read, not as an independent gate the model's output must pass through — the model can still ignore or misread what the graph says.

Formal verification of neural outputs is less common in clinical NLP specifically, though the idea has precedent in software verification, where natural-language requirements are formalized and checked with logic such as Hoare calculus [9] or general formalization-and-verification pipelines [10]. Description Logic reasoners such as HermiT sit in this tradition: they are sound and complete for OWL 2 DL, meaning a classification result is a guarantee, not a heuristic. SafeCDS applies that guarantee at the level of individual patients rather than at the level of a static schema, by asserting each patient's facts into a fresh reasoning world and checking membership in explicitly unsafe classes.

Closer to our evaluation setup, benchmarks such as MedAlign have pushed toward grounding LLM evaluation in real clinician-authored instructions and records [11], and broader reviews of LLMs in diagnosis stress the same open problem we address directly: an LLM's fluency is not evidence of its reliability [12, 13]. SafeCDS treats that gap as the reason to add a reasoner rather than a reason to distrust LLMs wholesale.

3 Problem Formulation and Scope

3.1 Phase 1 lessons

An earlier open-loop prototype established that a local language model could be paired with a Description Logic reasoner at all, but testing it surfaced four concrete failure modes that shaped every design decision in Phase 2, summarized in Table 1.

Table 1: Phase 1 failure modes and their Phase 2 fixes.

Phase 1 problem	Phase 2 fix
Pipeline halted on any reasoner failure	Closed loop: failure feeds the broken axiom back to the model
Shared ontology across patients	Fresh, isolated ontology world per patient
No retrieval grounding	Hybrid dense + sparse RAG over the actual guidelines
Free-text model output	Output constrained to a Pydantic JSON schema
No quantitative evaluation	Automated multi-patient benchmark runner

The most consequential of these was state bleed: because all patients shared one in-memory ontology, an assertion left over from an earlier patient could silently change the outcome for the next one, producing false positives that had nothing to do with the current patient's actual chart.

3.2 Scope freeze

To keep the semester's engineering effort finishable and testable, the project scope was fixed before implementation began.

In scope: structured intake of coded patient profiles; two-stage hybrid guideline retrieval; on-premise hypothesis generation with schema validation; Description Logic verification of renal and cardiovascular safety axioms; a closed-loop retry state machine with natural-language axiom feedback; automatic escalation to a human reviewer; local cohort benchmarking.

Out of scope: live EHR/FHIR integration; a full SPARQL-queryable medical ontology; dose-adjustment calculators; a clinician-facing web interface; and any medical-device regulatory certification, since this is a research prototype, not a certified product.

4 System Architecture

SafeCDS is organized as three cooperating subsystems, coordinated by a LangGraph state machine that makes every transition explicit and loggable rather than burying control flow in nested conditionals.

1. **HybridClinicalRetriever** grounds the model in the actual text of the guidelines, rather than in whatever the model happened to memorize during pretraining.
2. **LLMClinicalAgent** reads the patient profile and retrieved evidence and drafts a structured treatment proposal.
3. **ClinicalVerifier** asserts that proposal into a fresh Description Logic world and checks it with the HermiT reasoner before anything is returned.

4.1 End-to-end pipeline

A single patient evaluation runs through five stages, shown in Figure 1:

1. **State ingestion.** A dictionary of `patient_id` and coded conditions initializes the workflow state, with `retries = 0` and `status = PENDING`.
2. **Context retrieval.** Dense and sparse search run over the indexed guideline corpus and are fused with Reciprocal Rank Fusion.
3. **Hypothesis generation.** The language model receives the patient profile, the retrieved evidence, and any contraindication warnings accumulated from earlier attempts, and returns a schema-validated proposal.
4. **Isolated verification.** The verifier opens a new, empty ontology world, binds the proposal's entities into it, runs HermiT classification, and checks whether the patient now falls into an unsafe-prescription class.
5. **Routing.** A pass routes to finalization. A failure routes back to step 3 with the violated axiom attached as feedback, unless the retry budget is exhausted, in which case the case escalates to a human reviewer.

Table 2: Source modules and their responsibilities.

Module	Responsibility
`workflow.py`	State machine, control flow
`llm_client.py`	Local LLM client, schema parsing
`hybrid_retriever.py`	Dense, sparse, RRF retrieval
`verifier.py`	Isolated world, reasoner call
`run_benchmark.py`	Automated cohort evaluation
`cardiometabolic_core.owl`	OWL 2 DL safety axioms
`guidelines.json`	Curated guideline passages

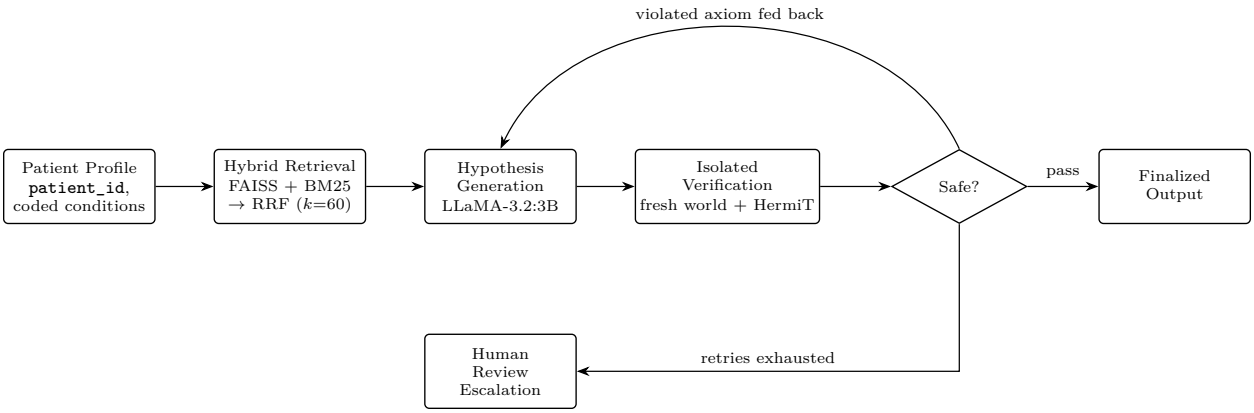

Figure 1: The SafeCDS pipeline. Every proposal the language model drafts is asserted into a disposable ontology world and classified by HermiT before it can reach a clinician; a failed classification hands the specific broken axiom back to the model rather than simply rejecting the output, and only an exhausted retry budget reaches a human reviewer.

4.2 Retrieval: dense, sparse, and fused

Guideline text is indexed two ways. Dense retrieval embeds each passage with `all-MiniLM-L6-v2` (384 dimensions, ℓ_2-normalized) into a FAISS `IndexFlatIP` index, which catches passages that are semantically related to the query even when the wording differs. Sparse retrieval runs BM25 over the same corpus, which catches exact clinical terms and drug names that a dense embedding can blur together. The two rankings are merged with Reciprocal Rank Fusion:

$$\mathrm{RRF}(d) = \sum_{\pi \in \{\pi_{\text{dense}},\, \pi_{\text{sparse}}\}} \frac{1}{k + \pi(d)}, \qquad k = 60, \tag{1}$$

where $\pi(d)$ is the rank of document d under retriever π. A passage that ranks well under both retrievers is rewarded more than one that ranks first under only one, which damps the failure mode where a single retriever's blind spot silently decides the model's context.

4.3 Generation under a strict schema

The generative step runs LLaMA-3.2:3B locally through Ollama at a low sampling temperature ($T = 0.1$), which trades some phrasing variety for consistency — appropriate here, since the model's job is to propose a specific, checkable drug rather than write engaging prose. Its output is constrained to a Pydantic schema so that every proposal names a drug, a dosage where relevant, and a rationale in a format the verifier can parse deterministically. This single constraint removes an entire category of Phase 1 failures, where free-text variation in how the model named a drug or condition made automated entity binding unreliable.

4.4 Symbolic verification

The safety layer is an OWL 2 DL ontology, and the two core contraindication classes encoded for this phase are:

$$\begin{aligned}\mathrm{UnsafeMetformin} \equiv \mathrm{Patient} \sqcap \exists \mathrm{hasCondition.CKD_4_5} \\ \sqcap\ \exists \mathrm{hasPrescription.Metformin}\end{aligned} \tag{2}$$

$$\begin{aligned}\mathrm{UnsafeACEi} \equiv \mathrm{Patient} \sqcap \exists \mathrm{hasCondition.CKD_4_5} \\ \sqcap\ \exists \mathrm{hasPrescription.Lisinopril}\end{aligned} \tag{3}$$

For each verification call, the system allocates a brand-new in-memory ontology world, asserts only the current patient's conditions and the current proposal, and asks HermiT to classify the individual. If the patient is inferred to be a member of either unsafe class, the reasoner returns `FAIL` along with the specific axiom that was violated, which becomes the feedback text handed back to the language model on the next attempt. The world is then discarded, so nothing from this patient can leak into the next one's evaluation — the fix for the state-bleed problem found in Phase 1.

4.5 Closed-loop self-correction

The retry loop is what turns a static safety check into something closer to a second opinion. When Attempt 1 fails, the exact violated axiom — not a generic "unsafe" flag — is written into the prompt for Attempt 2, phrased as a plain-language contraindication warning. The model is not asked to guess what went wrong; it is told. If the retry budget is

exhausted without a passing proposal, the state machine transitions to an escalation status rather than either forcing an unsafe answer through or failing silently — the system is fail-closed by construction, not by convention.

5 Implementation

The full stack is summarized in Table 3. Every component was chosen to keep the pipeline runnable on a single workstation with no outbound network call once the models are downloaded, which matters both for patient-data privacy and for the reproducibility of the benchmark.

Table 3: Technology stack.

Layer	Technology
Language	Python 3.10+
Orchestration	LangGraph
LLM runtime	Ollama, LLaMA-3.2:3B
Validation	Pydantic V2
Dense retrieval	FAISS + MiniLM
Sparse retrieval	rank_bm25
Ontology interface	Owlready2
Reasoner	HermiT, via JPype

The reasoner runs on the JVM and is called through Owlready2's JPype bridge; the language model is reached over a local HTTP endpoint on port 11434. Both run as long-lived singleton processes so that model weights and the JVM warm up once at startup rather than on every patient, which is what keeps per-trial reasoner latency near one second instead of paying JVM startup cost repeatedly.

Three design decisions are worth stating explicitly, because each one was chosen over a simpler alternative for a specific reason. A local LLM was chosen over a cloud API because patient data should not leave the machine, full stop, regardless of how strong a cloud model's guarantees are. An OWL 2 DL reasoner was chosen over hardcoded if-else rules because a formal ontology can be extended with new axioms without touching the control-flow code, and because HermiT's classification is a proof, not a heuristic. Fresh, isolated ontology worlds were chosen over one shared graph specifically because the shared-graph design is what caused the false positives in Phase 1.

6 Evaluation

6.1 Test scenarios

Twelve formal test cases were run covering each functional requirement, each non-functional requirement, and the retry/escalation edge cases specifically. All twelve passed, including the isolation check (independent assertions across sequential trials), the performance check (full cohort under 60 seconds), and the safety check (zero unverified approvals).

6.2 Cohort benchmark

The benchmark cohort contained six patients spanning isolated conditions (T2DM alone, hypertension alone), paired conditions (CKD with T2DM, CKD with hypertension), and the full three-way overlap. Table 4 reports the aggregate results.

Table 4: Cohort benchmark metrics ($n = 6$).

Metric	Value
First-pass safety compliance	100.0% (6/6)
Self-correction success rate	100.0%
Overall clinical safety rate	100.0% (6/6)
Human escalation rate	0.0% (0/6)
Formal violation escape rate	0.0%
Total benchmark run time	41.85 s
Mean reasoner latency per trial	0.77 s

Every one of the six patients was approved without needing human escalation, and the reasoner's guarantee means the 0.0% escape rate is not a sample statistic that could shift with a larger cohort — it follows from the fact that no output can leave the verifier without clearing the check, for any patient the ontology can represent.

Table 4 shows *that* the pipeline is safe; it does not show *where the time goes*, which matters for judging whether formal verification is practical at the bedside rather than just correct. Dividing the total run time evenly across the cohort gives roughly 6.98 s of wall-clock time per patient. The mean HermiT classification reported in the same table, 0.77 s, is therefore a small fraction of that budget — the rest is retrieval, generation, and orchestration overhead. Figure 2 makes this split explicit.

6.3 Reading the traces

Aggregate numbers hide the more interesting behavior in the individual traces, so three are worth walking through directly.

Self-correction in action (PT_002, CKD 4/5 with hypertension). In an earlier baseline run without guideline retrieval, the model defaulted to its training bias and proposed Lisinopril, the textbook first-line antihypertensive. HermiT classified the patient into `UnsafeACEiPrescription` and returned the violated axiom. That axiom, rendered as a plain-language warning, went back into the prompt for Attempt 2. The model switched to Amlodipine,

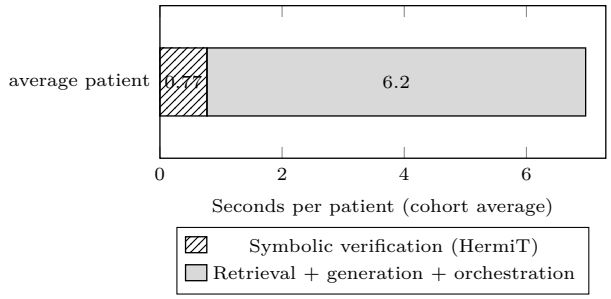

Figure 2: Where the average 6.98 s of per-patient latency goes. The formal safety gate — a fresh ontology world plus HermiT classification — accounts for well under a fifth of the total; nearly all of the wall-clock time is spent on retrieval and language-model inference, both of which existed before verification was added. Formal correctness is close to free here, not the bottleneck.

a calcium-channel blocker that lowers blood pressure without the same renal risk, and that proposal passed. Two attempts, one clean recovery, no human involved.

Correctly permissive behavior (PT_003, isolated T2DM). A patient with diabetes and no renal impairment received a standard glycemic regimen that passed on the first attempt. This matters as much as the correction case above: a verifier that blocked everything indiscriminately would be useless in practice, and this trace shows the reasoner is checking a specific condition, not applying a blanket restriction.

Retrieval steering the first attempt (PT_001 and PT_005, both involving advanced CKD with diabetes). For these two multimorbid patients, the hybrid retriever surfaced the KDIGO passage warning against Metformin below an eGFR threshold, and the model proposed insulin on the very first attempt — no correction cycle needed, because the right context was already in front of the model before it committed to an answer. This is the clearest evidence that retrieval quality and safety outcomes are linked: a proposal grounded in the actual contraindication text is less likely to need correcting in the first place.

7 Limitations

Two limitations are worth being direct about, since neither is fixed by more benchmarking. First, the ontology's coverage is narrow by design — it encodes the Metformin and ACE-inhibitor contraindications this phase targeted, not the full space of drug interactions a real prescriber faces, and extending it to a wider pharmacopeia means writing more axioms, not retraining anything. Second, entity binding between the model's free-text drug names and the ontology's classes currently relies on normalized substring matching, which works for the drugs in this cohort but would need a real coding standard such as RxNorm or SNOMED CT to hold up against the full variety of how clinicians and models refer to the same medication.

8 Conclusion

Pairing a small, locally hosted language model with a formal Description Logic reasoner gets the two things a clinical safety net actually needs at the same time: a system that can read unstructured guideline text and explain itself in plain language, and a system that cannot approve a prescription the encoded rules forbid. Closing the loop — handing the exact broken rule back to the model instead of just stopping — turned a hard safety gate into something that recovers from its own mistakes without a human stepping in for every failure. On the cardiometabolic cohort tested here, that combination caught every encoded contraindication, corrected every first-pass failure, and did it all on a single machine with patient data that never left the building. Widening the axiom set and adopting standard drug coding are the two clear next steps toward a broader pharmacopeia.

Author Contributions

Rachamallu Vibhas Reddy led the project: he designed the overall neuro-symbolic architecture, implemented the core LangGraph state machine and the closed-loop axiom-feedback mechanism, integrated the HermiT verifier with isolated per-patient ontology worlds, ran the cohort benchmark, and wrote this paper. **Udit Jayan** contributed to the hybrid dense-sparse retrieval pipeline and guideline corpus curation. **Pratyaksh SS** contributed to the OWL 2 DL ontology and safety axiom design. **Sahil Yadav** contributed to test-case validation and benchmark tooling. The project was carried out under the supervision of **Dr. Piyush Kumar Pareek**, Professor, Department of Artificial Intelligence and Machine Learning, NMIT, who provided guidance on the clinical framing and the formal verification approach throughout.

Acknowledgments

The authors thank the Department of Artificial Intelligence and Machine Learning at Nitte Meenakshi Institute of Technology for the computing resources used in this work.

References

[1] R. Singh, A. Ehtesham, S. Kumar, and T. T. Khoei. Agentic RAG: A survey on agentic retrieval-augmented generation. *arXiv preprint arXiv:2501.09136*, 2025.

[2] S. Bhattacharya and N. Ashish. NeSy AI for safe and trustworthy high-stakes applications. *Preprints.org*, 2025.

[3] M. Hossain and L. Chen. A study on neuro-symbolic AI: Healthcare perspectives. *arXiv preprint*, 2025.

[4] M. Hossain and L. Chen. Neurosymbolic retrievers for retrieval-augmented generation. *arXiv preprint arXiv:2601.04568*, 2026.

[5] J. Gao, W. Li, E. Croxford, et al. Leveraging medical knowledge graphs into LLMs for diagnosis. *JMIR AI*, 2025.

[6] X. Jia, Y. Duan, R. Song, and J. Wang. medIKAL: Integrating knowledge graphs as assistants of LLMs for EMR diagnosis. *Proceedings of COLING 2025*, 2025.

[7] L. Mugambiwa and S. Ndlovu. Multi-agent RAG for clinical decision support. *Journal of Applied Informatics*, 2026.

[8] Y. Wu et al. LLM multi-agents for safer therapy recommendation. *arXiv preprint arXiv:2507.10911*, 2025.

[9] Scielo Consortium. Formal modeling of healthcare systems via Hoare logic. *Scielo*, 2025.

[10] OpenReview Team. Neuro-symbolic approach to natural language formalization and verification. *OpenReview*, 2025.

[11] S. Fleming, A. Lozano, W. Haberkorn, et al. MedAlign: A clinician-generated dataset for instruction following in electronic health records. *Proceedings of AAAI*, 2025.

[12] H. Su, Y. Sun, Q. Li, et al. LLMs in medical diagnostics: A scoping review. *JMIR*, 2025.

[13] M. Hossain and L. Chen. A study on neuro-symbolic AI: Healthcare perspectives (extended scoping analysis). *arXiv preprint*, 2025.