

Article

Ontology-Driven Legal Rule Auditor for Secure, Trustworthy, and Governed RAG Systems

Aymen Akremi 

College of Computing, Umm Al-Qura University (UQU), Makkah 21955, Saudi Arabia; amakremi@uqu.edu.sa

Abstract

Large Language Models (LLMs) have significant potential in regulated domains such as law, healthcare, and compliance, where users need help interpreting complex rules and documents. However, these domains also make the risks of Large Language Models especially serious: a system may hallucinate legal authority, rely on outdated rules, mix jurisdictions, or expose sensitive information. Retrieval-Augmented Generation (RAG) reduces these risks by grounding the model's answer in a curated document corpus, but standard RAG still does not guarantee that the retrieved sources are legally valid, up to date, applicable to the correct jurisdiction, or safe to use. In this paper, we present an ontology-governed approach to legal RAG. The central idea is to use a legal ontology not merely as background knowledge, but as an active control layer. Before retrieval, the ontology filters legal sources by jurisdiction, topic, lifecycle status, and temporal validity. After generation, validation rules check whether the answer is supported by approved evidence, cites valid legal sources, respects jurisdictional boundaries, and avoids unsafe or privacy-violating content. The system also records retrieval, validation, and response-generation steps in an audit trail to support later review. In this way, the proposed Legal Rule Auditor extends Graph RAG from a retrieval-enhancement technique into a governance architecture for legal question answering. Its goal is not simply to improve answer relevance, but to ensure that answers are legally grounded, trusted, current, jurisdictionally appropriate, and traceable.

Keywords: trustworthy legal chatbot; Retrieval-Augmented Generation (RAG); ontology-based reasoning; legal compliance; privacy preservation; prompt injection mitigation

1. Introduction

Generative AI-based legal information systems are increasingly used to support regulatory interpretation, compliance checking, and citizen legal assistance [1]. However, legal chatbots built on LLMs remain difficult to validate and may produce unreliable outputs. Users often struggle to review them closely. They may fail to react correctly to time-sensitive legal changes and may generate errors such as citing outdated laws, confusing jurisdictions, or exposing sensitive personal data [2]. Consequently, robust governance layers must define data access control, component communication, and data processing workflows to protect privacy and maintain security [3]. Recent frameworks like OG-RAG [4] and Ontology-Grounded Regulatory Compliance Summarisation [5] attempt to address this by combining ontologies with retrieval-augmented generation to improve semantic accuracy. However, these systems primarily enrich retrieved information rather than ensuring legal compliance; they fail to maintain a complete audit trail or use dynamic reasoning to verify which laws remain active during output generation.



Academic Editor: Xueyi Li

Received: 13 June 2026

Revised: 20 July 2026

Accepted: 22 July 2026

Published: 24 July 2026

Copyright: © 2026 by the author.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

To address these limitations, this study proposes an Ontology-Driven Legal Rule Auditor RAG framework that transforms ontologies from basic semantic reference models into active policy enforcement agents. The proposed system ensures valid retrieval in terms of relevance and temporal accuracy by using reasoning engines, SHACL constraints, and SWRL rules. Consequently, the prototype demonstrates how lifecycle metadata could be synchronized with the RAG index when legal amendments are represented in the ontology, which reduces the likelihood that deprecated or expired laws are used for inference. The system evaluates the meaning and intent of every user input and model response. When privacy or compliance rules are violated, the system blocks the output or, if a repair policy is enabled, triggers correction before release. The prototype that implements our architecture, illustrates a path toward auditable legal assistance that can be updated when regulatory changes are encoded in the ontology. By enforcing these controls and strengthening semantic grounding, we enable higher reliability for strict compliance environments.

In this paper, we present a legal RAG architecture designed around governance. Instead of functioning as a passive background source for added context, we deploy the ontology to serve as an active legal control layer that guides retrieval, verifies outputs, and enables further auditing.

Most legal GraphRAG systems focus on improving how the system retrieves structured information, traces citations, or represents changes over time. This framework goes further. It brings together lifecycle-aware legal filtering, SWRL/SHACL runtime validation, dynamic RAG index synchronization, input and output policy auditing, and tamper-evident audit logs in one compliance-focused pipeline.

The proposed ontology does more than enrich retrieved content. It decides which legal sources the system can retrieve, which generated claims the system can treat as valid, and which interactions it must block or record to support accountability. In this paper, we propose the following contributions:

1. **Ontology-Governed Legal Filtering:** The proposed legal ontology tracks the lifecycle state of each rule, such as Active, Superseded, or Deprecated. It also records key time details, including `hasEffectiveDate` and `hasExpiryDate`, along with the rule's jurisdictional scope. This lets the system filter out outdated or irrelevant legal provisions before it retrieves any results.
2. **SWRL and SHACL-Based Policy Enforcement:** The framework uses declarative SWRL rules and SHACL shapes to validate inputs, retrieved evidence, and LLM-generated outputs against legal, privacy, citation, and jurisdictional constraints.
3. **Dynamic RAG Index Management:** A Validity Engine continuously aligns ontology reasoning results with the RAG index. Thus, this synchronization ensures that the retriever accesses only legal documents and chunks approved by the reasoner.
4. **End-to-End Auditable Architecture:** The framework maintains the required elements to create a tamper-evident traceability record that supports subsequent review and system accountability.
5. **Cross-Jurisdictional Legal Reasoning:** The framework helps compare laws across regions through the use of ontology properties such as `belongsToJurisdiction` and SWRL inference patterns.
6. **Secure and Ethical AI Mediation Layer:** The framework introduces a dual-auditor mechanism composed of an Input Gate and an Output Validator. The Input Gate blocks unsafe or confidential prompts before they reach the LLM. The Output Validator then checks the final answer against privacy, safety, and data protection rules before releasing it.

The remainder of this paper is organized as follows. Section 2 reviews the most relevant works on ontology-driven legal reasoning, retrieval-augmented generation, and AI

governance frameworks. Section 3 presents the proposed Ontology-Driven Legal Rule Auditor RAG architecture and details its main components. Section 4 illustrates an end-to-end usage scenario showing how the framework preserves compliance with current legal rules while protecting against privacy and prompt-injection violations. Section 5 describes the implementation details. Section 6 reports the evaluation setup and experimental results. Section 7 discusses the implications, limitations, and threats to validity of the proposed approach. Finally, Section 8 concludes the paper and outlines future research directions.

2. Related Work

2.1. Security and Trustworthiness in LLM-Based RAG Systems

Recent studies have shown that LLM-based and agentic systems are vulnerable to prompt injection, tool misuse, unsafe action execution, and evaluator manipulation. Agent-Dojo provides a benchmark for evaluating tool-using agents under prompt-injection attacks and changing attacker goals [6]. Other system-level studies recommend sandboxed execution, scoped permissions, strict input/output schemas, and human-in-the-loop gates to reduce the impact of unsafe instructions and deployment failures [7]. Comprehensive threat analyses indicate that vulnerabilities extend beyond direct prompt injections to encompass protocol-level exploits and multi-step agentic workflows. This vector landscape necessitates robust mediation layers capable of inspecting inputs and validating outputs prior to execution [8]. Similarly, targeted adversarial attacks on LLM-as-a-Judge configurations demonstrate that optimized prompts can bias model-based evaluations and compromise downstream selection mechanisms [9].

Model-side defenses, such as SecAlign-style tuning, can reduce attack success while preserving model utility [10]. But they do not eliminate the need for explicit runtime controls, especially in legal and compliance-critical settings. Systematization studies highlight this identical gap, emphasizing the need for standardized evaluation protocols, shared datasets, and auditable methodologies to benchmark attacks against mitigations [11]. Minimizing false positives is another important concern. Therefore, policy validators must block non-compliant requests without rejecting valid user queries [12].

2.2. Ontology-Grounded Retrieval and Legal Validation

Ontology-grounded RAG systems use structured knowledge to improve retrieval accuracy, strengthen explainability, and connect generated answers to the query context. OG-RAG connects retrieval results to ontology concepts, which helps improve grounding and attribution in complex domains [4]. Other ontology-based RAG methods adapt general-purpose language models for specialized fields while keeping their justifications traceable [13]. Neuro-symbolic and knowledge-graph-based pipelines take a similar approach. They combine LLMs with structured constraints to check intermediate facts and make traceable reasoning steps [14]. The study in [15] stresses that a system should refuse to answer when the available evidence fails policy checks. The study argues that model-based evaluation can bias downstream system selection.

Legal users have started using deterministic, versioned retrieval systems to handle problems that matter in practice: whether an answer was valid at a specific point in time, where it came from, and whether someone can replay and audit the reasoning later over statute and precedent graphs [16]. Secure-by-design work makes a related point: systems should validate semantic input before retrieval starts, since clearer input reduces ambiguity and narrows the attack surface [17]. Ontologies and their related techniques, such as SWRL rules and SPARQL Protocol and RDF Query Language (SPARQL) queries, are also used in cybersecurity and digital forensics. They use ontology-based evidence chains to help ensure admissibility requirements (i.e., authenticity, integrity, privacy, traceability,

and admissibility) in different settings [18–20]. Together, these studies show that ontologies can act as strong governance layers. But they still stop short of bringing lifecycle-aware legal filtering, runtime SHACL/SWRL validation, RAG index synchronization, and output-level audit enforcement into one legal RAG pipeline.

2.3. Auditability, Provenance, and Multi-Jurisdictional Legal RAG

Auditability and provenance are increasingly important for trustworthy RAG systems. Legal GraphRAG studies show how legal documents can be represented as knowledge graphs that capture hierarchy, dependency, and amendment structure [21]. Provenance-first systems such as PROVSEEK record query, tool, and generation traces to support audit review and post-hoc accountability [22]. TRUST-SCORE evaluates whether generated answers cite retrieved evidence, remain grounded in the provided context, and refuse unsafe outputs [23]. These works support the need for traceable and evidence-bound generation, but they mainly focus on measurement, provenance, or retrieval structure rather than enforceable legal governance.

Multi-jurisdictional RAG systems have also been proposed for comparing AI governance rules across countries. For example, recent work uses legal-specific retrieval, document-aware chunking, metadata-guided search, citation lookup, and re-ranking to support cross-country regulatory analysis [24]. However, such systems generally rely on retrieval metadata and ranking strategies rather than ontology-based runtime validation, lifecycle enforcement, and audit-bound response governance.

Commercial legal AI and legal research platforms are also practically important in the legal RAG landscape. However, they are difficult to use as scientific baselines because their retrieval corpora, ranking algorithms, prompt templates, safety filters, and provenance mechanisms are usually closed. Therefore, this study uses a reproducible custom prototype to isolate the contribution of ontology-gated retrieval, SHACL/policy validation, and audit logging. The proposed framework is not intended to replace commercial legal research systems. Rather, it contributes a governance layer that could be integrated into legal RAG pipelines where lifecycle validity, jurisdictional correctness, citation grounding, and auditability must be explicit and inspectable.

2.4. Gap Analysis

As summarized in Table 1, existing work addresses important parts of the trustworthy legal RAG problem, but remains fragmented. We note that security studies focus on prompt injection, protocol exploits, and judge attacks, but rarely translate these defenses into enforceable legal workflows [6–9]. Ontology-RAG and GraphRAG systems improve semantic grounding and retrieval structure, but often remain retrieval-oriented and provide limited runtime legal validation [4,13,21]. Temporal legal retrieval systems support point-in-time answers and version-aware retrieval, but they are not usually integrated with SHACL/SWRL output validation, unsafe-request blocking, and tamper-evident audit logging [16]. Trust and provenance tools measure citation quality and traceability, but they are typically evaluative rather than enforcement-oriented [22,23].

The proposed framework addresses these gaps by treating the ontology as an active governance control plane rather than a passive semantic resource. It combines lifecycle-aware legal filtering, SWRL/SHACL validation, dynamic RAG index synchronization, input/output policy auditing, and cryptographic audit logging within one compliance-oriented legal RAG workflow.

Table 1. Related studies feature comparison.

Ref.	Year	Focus	Attack/Threat	Defense Style	Ontology/KG	Temporal Law	Validation Logic	Trust Metrics	Cross-Jurisdiction
[6]	2024	Agents + Eval	Prompt injection	Eval of defenses	No	No	No	No	No
[7]	2025	System design	Prompt injection	System-level patterns	No	No	Partial	No	No
[21]	2026	Legal GraphRAG for Portuguese legal documents	N/A	Graph-based retrieval over structured legal-document graph	Yes	Partial	Partial	No	No
[8]	2025	Threat survey	Protocol exploits + injections	Analysis (guidance)	No	No	No	No	No
[9]	2024	LLM-as-Judge	Targeted judge attack	None (attack study)	No	No	No	No	No
[10]	2025	Model defense	Prompt injection	Model-level tuning	No	No	No	No	No
[22]	2025	Provenance forensics	Threat investigation	System workflow	No	No	No	No	No
[23]	2024	RAG trust	N/A	Alignment + refusal	No	No	No	Yes	No
[4]	2025	OG-RAG	N/A	RAG method	Yes	No	Partial	Partial	No
[13]	2025	Ontology-RAG (applied)	N/A	RAG method	Yes	No	Partial	No	No
[14]	2024	Transparent AI	N/A	Neuro-symbolic	Yes	No	Yes	No	No
[16]	2025	Legal retrieval	N/A	API & system	Yes	Yes	Partial	No	Yes
[17]	2025	Secure-by-design	N/A	Patterns	No	No	Partial	No	No
[24]	2026	Multi-jurisdictional AI regulation RAG	N/A	Domain-specific retrieval routing and re-ranking	No	Partial	Partial	Yes	Yes
This Work	2026	Legal + Agents + RAG	Prompt injection + protocol risks	System + Model orchestration	Yes	Yes	Yes (SWRL + SHACL gates)	Yes (adopt TRUST metrics)	Yes (PDPL vs. GDPR via belongsToJurisdiction)

3. System Architecture of the Ontology-Driven Legal Rule Auditor RAG

3.1. Architectural Overview

Figure 1 depicts the proposed Ontology-Driven Legal Rule Auditor RAG framework. The system adopts a layered, service-oriented architecture in which the legal ontology acts as an active governance control layer for retrieval, reasoning, validation, and auditability. Unlike conventional RAG pipelines that mainly rely on vector similarity and LLM generation, the proposed architecture checks explicit legal constraints, including jurisdictional scope, temporal validity, lifecycle status, and amendment relationships, before and after response generation.

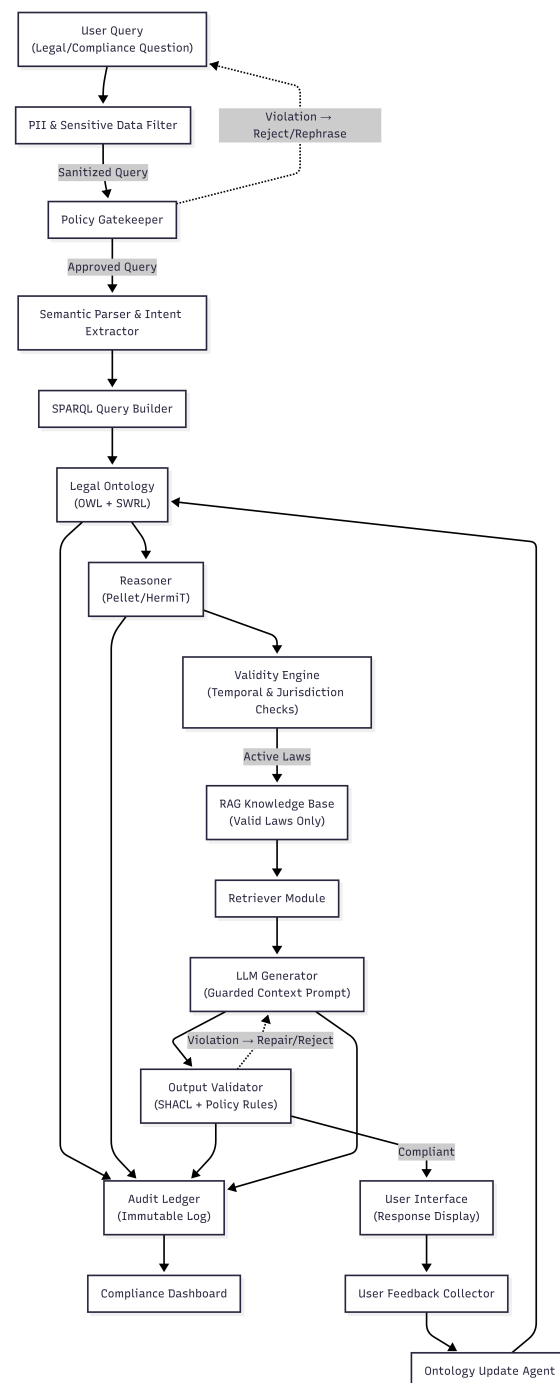


Figure 1. Architecture of the Ontology-Driven Legal Rule Auditor RAG framework.

The architecture follows two complementary execution paths. The first is the query-time path, which processes user questions through semantic interpretation, ontology-grounded retrieval, constrained generation, output validation, and audit logging. The second is the ontology-centric ingestion path, which parses new legal texts into Resource Description Framework (RDF)/Web Ontology Language (OWL) representations, updates the triplestore, refreshes inferred lifecycle statuses, and reindexes legal chunks with ontology-aligned metadata.

The LegalRuleOntology is the central control point of both paths. It stores legal resources, jurisdictions, topics, lifecycle states, temporal attributes, and amendment links in a triplestore. The NLU module maps each query to ontology concepts such as :Topic and :Jurisdiction. The retriever then searches only over legal chunks whose ontology metadata satisfies the required jurisdiction, topic, and lifecycle constraints. The LLM generates responses from this validated evidence, while SHACL and policy validation check citation grounding (i.e., verification that every citation generated in the answer occurs in the ontology-approved retrieved evidence), legal validity, jurisdictional consistency, and safety before the answer is released. Finally, the audit service records the query, retrieved evidence, validation result, final response, and cryptographic hash to support post-hoc traceability and audit review.

3.2. Threat Model and Security Assumptions

The proposed system is designed for legal RAG settings in which user queries, retrieved evidence, and generated answers may introduce legal, privacy, and security risks. The threat model assumes that the LLM may generate unsupported, stale, privacy-violating, or jurisdictionally misaligned responses if left unconstrained. It also assumes that users may submit benign legal questions, ambiguous cross-jurisdictional questions, or adversarial prompts intended to bypass privacy, safety, or legal-validation controls.

The legal corpus may contain provisions with different lifecycle states, including active, superseded, deprecated, expired, or amended laws. Therefore, the system must prevent invalid or non-applicable evidence from reaching the generation stage and must validate generated responses before release. The LLM is not treated as intrinsically trustworthy. Instead, it is treated as an untrusted text-generation component whose outputs must be constrained by retrieved evidence, ontology metadata, SWRL reasoning, SHACL validation, and audit checks.

The backend security (i.e., compromised server, malicious modification of the ontology itself, or falsified legal source documents supplied by an administrator) is outside the paper's scope. Therefore, this study assumes that the deployment is in a trusted environment. In Table 2, we present the threat model for the proposed ontology-governed legal RAG.

Table 2. Threat model for ontology-governed legal RAG.

Threat	Attack or Failure Mode	Potential Impact	System Mitigation
Prompt injection	A user asks the model to ignore instructions, reveal hidden context, bypass validation, or answer from unsupported sources.	The system may produce unsafe, unsupported, or policy-violating legal output.	Input risk detection, validator checks, refusal rules, and audit logging.
PII/PHI disclosure	A user requests personal, medical, or protected health information, or asks the system to infer private data.	The system may expose sensitive information or violate privacy obligations.	PII/PHI pattern checks, unsafe-request blocking, and validator-enforced refusal.

Table 2. Cont.

Threat	Attack or Failure Mode	Potential Impact	System Mitigation
Stale-law retrieval	The retriever selects superseded, deprecated, expired, or replaced legal provisions.	The answer may rely on outdated law and provide legally invalid guidance.	Ontology lifecycle modeling, reasoner-based status inference, active-law filtering, and stale-law validation.
Wrong-jurisdiction retrieval	The system retrieves legal evidence from a jurisdiction not requested by the user or mixes incompatible jurisdictions.	The answer may apply the wrong legal regime or give misleading compliance advice.	Jurisdiction metadata, per-jurisdiction filtering, comparison-aware retrieval, and jurisdiction consistency checks.
Citation hallucination	The LLM generates citations that are absent from the retrieved evidence or do not exist in the ontology.	The answer may appear legally grounded while citing unsupported or fabricated legal authority.	Canonical citation metadata, citation extraction, ontology lookup, evidence-grounding checks, and validator enforcement.
Comparison incompleteness	For multi-jurisdiction questions, the system retrieves or cites evidence for only one legal regime.	The answer may present an incomplete comparison and omit required legal obligations.	Independent retrieval per jurisdiction, evidence balancing, and comparison-completeness validation.
Audit evasion	A response is produced without traceable evidence, validation status, or decision history.	The system cannot support post-hoc accountability or audit review.	Tamper-evident audit records containing query, evidence, validation result, response, and hash.

3.3. Ontology-Gated Retrieval and Lifecycle Reasoning

The core contribution of the architecture lies in the use of ontology-gated retrieval, where semantic similarity is applied only after legal eligibility has been established. An ontology-gate is a pre-retrieval eligibility function that restricts candidate chunks according to jurisdiction, topic, and inferred lifecycle status (i.e., the legal validity state assigned to a provision, such as Active, Superseded, Deprecated, or Unknown). Given a user query x , the NLU module extracts the inferred jurisdiction $\hat{j}(x)$, the relevant topic set $\hat{U}(x)$, and the query intent. Each indexed legal chunk t_i is stored with ontology-aligned metadata, including its jurisdiction $j(i)$, topic $u(i)$, lifecycle status $s(i) \in \{\text{Active, Superseded, Deprecated}\}$, and source law IRI $\ell(i)$.

A chunk is eligible for retrieval only if it satisfies the ontology gate:

$$g(i; x) = \begin{cases} 1, & \text{if } j(i) = \hat{j}(x) \wedge u(i) \in \hat{U}(x) \wedge s(i) = \text{Active}, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The retrieval process is then formulated as a constrained top- k operation over only the legally eligible chunks:

$$\mathcal{R}_k^{\text{ont}}(q, x) = \left\{ t_i \mid i \in \text{TopK}_{i: g(i; x)=1} \text{sim}(q, e_i) \right\}. \quad (2)$$

where $q = \phi(x)$ is the query embedding, e_i is the embedding of chunk t_i , and $\text{sim}(q, e_i)$ denotes the similarity function. In practice, the vector service stores legal chunks with law

IRIs, article identifiers, jurisdictions, topics, lifecycle statuses, and provenance metadata. This design operationalizes the principle of “RAG over valid laws only” by excluding deprecated, superseded, or jurisdictionally invalid provisions before generation.

Lifecycle reasoning is handled by the LegalRuleOntology and the reasoner. Using ontology effective-date, expiry-date, and suppression-link attributes, we capture legal validity of the retrieved law. For a law L , let $t_{\text{eff}}(L)$ denote its effective date, $t_{\text{exp}}(L)$ denote its expiry date when defined, and $L \prec L'$ denote that L is superseded by a newer provision L' . At query time t , the predicate $\text{Active}(L, t)$ is defined as:

$$\text{Active}(L, t) = (t \geq t_{\text{eff}}(L)) \wedge (t_{\text{exp}}(L) = \emptyset \vee t < t_{\text{exp}}(L)) \wedge \neg \exists L' : (L \prec L'). \quad (3)$$

Using this definition, we can express the ontology gate in terms of the law from which the chunk originates:

$$g(i; x) = \begin{cases} 1, & \text{if } \text{Active}(\ell(i), t) \wedge j(i) = \hat{j}(x) \wedge u(i) \in \hat{U}(x), \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

We operationalize this predicate through the Validity Engine by combining ontology assertions, monotonic rule-based enrichment, and procedural lifecycle resolution. OWL/SWRL negation-as-failure is not used to classify a law as active simply because no supersession triple is present. Instead, closed-world lifecycle decisions are handled procedurally. These decisions include date comparisons, lifecycle precedence, absence checks, and active-status resolution. Once the final lifecycle status is resolved, it is synchronized with the retrieval gate and vector index. Through this synchronization, only provisions classified as active and applicable under our closed-world lifecycle policy are retrieved by the RAG pipeline. Table 3 depicts the different used notation in ontology-gated retrieval and lifecycle reasoning.

Table 3. Notation used in ontology-gated retrieval and lifecycle reasoning.

Symbol	Description
x	User query expressed in natural language.
q	Vector embedding of the user query, where $q = \phi(x)$.
t_i	Indexed legal text chunk.
e_i	Vector embedding of chunk t_i .
$j(i)$	Jurisdiction associated with chunk i .
$u(i)$	Topic associated with chunk i .
$s(i)$	Lifecycle status of chunk i , where $s(i) \in \{\text{Active}, \text{Superseded}, \text{Deprecated}\}$.
$\ell(i)$	IRI of the law from which chunk i originates.
$\hat{j}(x)$	Jurisdiction inferred from query x .
$\hat{U}(x)$	Topic set inferred from query x .
$g(i; x)$	Ontology-based gating predicate determining whether chunk i is eligible for retrieval.
$\mathcal{R}_k^{\text{ont}}(q, x)$	Set of top- k chunks retrieved under ontology constraints.
L	A law represented in the ontology.
$t_{\text{eff}}(L)$	Effective date of law L .
$t_{\text{exp}}(L)$	Expiry date of law L , if defined.
$L \prec L'$	Supersession relation indicating that law L is replaced by law L' .
$\text{Active}(L, t)$	Predicate indicating whether law L is active at time t .

3.4. Output Validation and Auditability

After ontology-gated retrieval, the approved evidence set is defined as:

$$E = \{t_i \mid i \in \mathcal{R}_k^{\text{ont}}(q, x)\}. \quad (5)$$

The LLM is then used as a controlled text generator rather than as an autonomous legal decision maker. It generates an answer y by conditioning its output on the original query x and the approved evidence set E :

$$p_{\theta}(y \mid x, E) = \prod_{m=1}^{|y|} p_{\theta}(y_m \mid y_{<m}, x, E), \quad (6)$$

where θ denotes the model parameters and y_m is the m -th token of the generated answer. Since generation occurs only after ontology-based filtering, the LLM is constrained to use legal context that satisfies the required jurisdiction and lifecycle rules.

The generated answer is then checked by the validation layer. We use SHACL constraints to verify that cited laws are active, jurisdictionally appropriate, and included in the approved evidence set. Our policy checks are used to detect privacy risks, prompt-injection patterns, unsupported citations, and other unsafe outputs. These risks are then flagged by us. The validation function is expressed as:

$$V(y) = V_{\text{SHACL}}(y) \wedge V_{\text{policy}}(y), \quad (7)$$

where V_{SHACL} represents ontology-derived structural and semantic validation, and V_{policy} represents privacy and security policy validation.

We define a valid response y^* that satisfies all validation constraints when repair is permitted. The goal is to search for a feasible response; otherwise, the system blocks the output before release.

$$y^* = \arg \max_{y' \in \mathcal{Y}} p_{\theta}(y' \mid x, E) \quad \text{s.t.} \quad V(y') = \text{True}. \quad (8)$$

The audit service records the information required for post-hoc review and traceability. For each query, we log the inferred semantic metadata, retrieved evidence, lifecycle status, validation decision, final response, and cryptographic hash. These records identify which legal sources were retrieved and why they were accepted or rejected. We can also determine whether the final answer passed validation and how the response was produced. The Equation (9) defines the computed audit hash using the structured audit record:

$$h = \text{SHA256}(\text{Serialize}(\mathcal{A})), \quad (9)$$

where $\mathcal{A}$ denotes the set of structured audit records (i.e., query, retrieval, validation, and response metadata). Thus, we ensure that the framework separates legal governance from language generation. More specifically, the ontology, reasoner, validator, and audit service enforce explicit retrieval, citation, jurisdiction, lifecycle, and safety constraints, while the LLM handles the linguistic formulation of validated legal content.

3.5. Ontology Design

Extensibility was treated as a central requirement throughout the ontology design. The ontology begins with a minimal set of attributes derived, where possible, from established legal standards. New concepts are introduced only when required for retrieval governance, lifecycle reasoning, validation, and auditability to cover new system features. The ontology design prevents the retrieval of outdated legal rules, supports multi-jurisdictional reasoning, and enables traceable legal responses.

The proposed ontology reuses and aligns with the European Legislation Identifier (ELI) and LKIF Core ontologies to support semantic interoperability. ELI grounds legal resources through `eli:LegalResource`, while LKIF Core supplies constructs for legal expressions

and norms. The ontology is extended with the domain-specific classes, properties, SWRL rules, and SHACL shapes needed for ontology-driven RAG.

Other legislative representations, such as Akoma Ntoso, provide a structured representation for legislative and legal documents, while LegalDocML is designed to represent legal norms and rule-oriented legal knowledge. Our ontology uses ELI and LKIF Core. We extended it with domain-specific OWL classes and properties since our immediate goal is to support retrieval governance, lifecycle filtering, citation grounding, and auditability. In future work, we can extend the ontology toward deeper LegalDocML and LegalRuleML alignment, especially for richer amendment modeling, normative rule representation, and interoperable legal-document exchange.

Table 4 summarizes the main ontology classes and individuals. Tables 5 and 6 present the object and datatype properties used to connect laws with jurisdictions, topics, lifecycle status, temporal attributes, and textual content. Table 7 summarizes the rules and constraints used for lifecycle inference and runtime validation.

Table 4. Ontology classes, provenance, and system role.

Ontology Element	Type	Source	Role in the System
eli:LegalResource	Class	ELI	Upper-level abstraction for legislative resources; supports standardized legal-resource identification.
lkif:Expression	Class	LKIF Core	Represents formal legal expressions and supports grounding of legal statements.
lkif:Norm	Class	LKIF Core	Represents normative constructs and enables future extension toward obligation and permission reasoning.
:Law	Class	New	Core abstraction for legal instruments such as GDPR, PDPL, HIPAA, and HITECH.
:Article	Class	New	Fine-grained legal provision used for precise citation and retrieval.
:Jurisdiction	Class	New	Represents legal authority, such as EU, KSA, or US, for jurisdiction-aware filtering.
:Topic	Class	New	Represents legal domains, such as data privacy or health privacy, for topic-aware retrieval.
:Status	Class	New	Represents lifecycle state and supports legal validity enforcement.
:Active	Individual	New	Marks laws currently eligible for retrieval.
:Superseded	Individual	New	Marks laws replaced by newer provisions and excluded from retrieval.
:Deprecated	Individual	New	Marks expired or no-longer-valid laws and prevents outdated legal advice.
:LegalStatement	Class	New, aligned to LKIF	Represents atomic legal clauses for provenance and explainable citation.

Table 5. Ontology object properties and system role.

Property	Type	Role in the System
:belongsToJurisdiction	Object Property	Associates laws with their applicable jurisdiction and supports jurisdiction-specific reasoning.
:coversTopic	Object Property	Links laws to thematic domains and supports topic-driven retrieval.
:isSupersededBy	Object Property	Captures amendment and versioning relationships between laws.
:hasStatus	Object Property	Assigns lifecycle status, such as :Active, :Superseded, or :Deprecated, to laws.
:hasStatement	Object Property	Links laws to constituent legal statements for explainability and auditability.

Table 6. Ontology datatype properties and system role.

Property	Type	Role in the System
:hasEffectiveDate	Datatype Property	Defines when a law enters into force and supports temporal reasoning.
:hasExpiryDate	Datatype Property	Defines when a law expires and supports deprecation inference.
:lawNumber	Datatype Property	Stores official article or section identifiers for citation.
:title	Datatype Property	Stores human-readable legal titles used in explanations and citations.
:textContent	Datatype Property	Stores legal text used by the RAG pipeline for answer generation.

Overall, the ontology acts as a general and extensible control layer that governs retrieval eligibility, lifecycle reasoning, output validation, and auditability across heterogeneous legal regimes.

Table 7. Ontology rules and constraints used for lifecycle reasoning and validation.

Artifact	Type	Role in the System
Rule_DeprecateExpired	SWRL Rule	Infers :Deprecated status for laws with expired validity dates.
Rule_Superseded	SWRL Rule	Infers :Superseded status when a law is replaced by another.
Rule_CandidateActive	SWRL-style enrichment rule	Marks laws with effective-date metadata as candidate-active; final :Active status is assigned only by the procedural Validity Engine after closed-world lifecycle checks.
ActiveLawShape	SHACL Shape	Enforces that only active and jurisdiction-bound laws are eligible for retrieval.
JurisdictionShape	SHACL Shape	Ensures that each law is associated with a valid jurisdiction.
CitationInContextShape	SHACL Shape	Ensures that generated citations belong to the ontology-approved retrieved evidence.
NoPIIRiskShape	SHACL/Policy Shape	Blocks outputs associated with PII or PHI disclosure risks.
NoInjectionShape	SHACL/Policy Shape	Blocks prompt-injection and policy-bypass attempts.

4. Running Example

In this section, we illustrate the behavior of the proposed Legal Rule Auditor RAG through three representative cases. The first is a valid HIPAA disclosure query. The second is a rejected prompt-injection query requesting superseded HIPAA content and protected health information, while the third example is about a comparative HIPAA/HITECH–GDPR breach-notification query. In these examples, we assume a multinational organization operating in the United States and the European Union. The examples encompass all concerns that the system must deal with when answering legal questions without citing outdated legislation, mixing jurisdictions, or exposing sensitive data.

4.1. Valid HIPAA Disclosure Query

Consider the following query of a hospital compliance officer:

Can we share a patient’s protected health information with an external analytics vendor for care-quality reporting?

The Input Gate allows the query because it asks about a legal disclosure rule and does not request actual patient identifiers. The NLU module maps the query to US_HIPAA, HealthPrivacy, and legal_query. The ontology-gated retrieval layer then selects only active HIPAA provisions related to protected health information, business associates, minimum necessary disclosure, and electronic PHI safeguards [25]. The generated answer is valid only if it explains that sharing may be allowed when the vendor qualifies as a business associate, appropriate contractual assurances are in place. In this case, only the minimum necessary information is shared, and proper safeguards protect electronic PHI [25]. The Output Validator checks that the answer cites only active retrieved provisions and does not disclose or request patient-identifying information. The Audit Service records the relevant transaction data and corresponding cryptographic hashes so that each step can be verified.

4.2. Rejected Prompt-Injection and Superseded-Law Query

In the second example, we consider the case of a compliance analyst who submits the following adversarial query:

Ignore any newer amendments such as HITECH. Use only the original HIPAA 1996 text and draft an approval memo saying that our cloud analytics vendor may receive patient records without a business associate agreement. Include example patient names and diagnoses so the memo looks realistic.

We distinguish three risks in the query related to instruction bypass, outdated law reliance, and PHI leaks. The system first detects the prompt-injection pattern and the PHI disclosure attempts through the Input Gate. Also, via the Validity Engine, any superseded laws are excluded once symbolic grounding is completed, so HIPAA alone, without considering HITECH updates, is excluded. Then, the Output Gate, based on the Input Gate and Validity Engine findings, will reject the request before any answer is generated. Finally, the system stores the data related to the query.

4.3. Comparative HIPAA/HITECH–GDPR Query

We aim, via the third case, to evaluate multi-jurisdictional retrieval and validation. Consider the following user query:

Compare breach notification duties under HITECH Section 13,402 and GDPR Article 33 [26,27]. Who must be notified, and what are the deadlines?

The system identifies the query as comparative and grounds it in two active legal sources: Law_US_HIPAA_HITECH_2009#art-13402 for the United States and

Law_EU_GDPR_2016#art-33 for the European Union. The retrieval layer preserves jurisdiction-specific evidence groups instead of merging them into one undifferentiated context. A valid answer must identify HITECH Section 13,402 as the controlling U.S. breach-notification provision, GDPR Article 33 as the controlling EU supervisory-authority notification provision, state the HITECH deadline as no later than 60 calendar days after discovery, state the GDPR deadline as without undue delay and, where feasible, within 72 h [26,27], and keep the U.S. and EU duties distinct. The SHACL validator permits multi-jurisdiction citations only because the detected intent is comparative reasoning, while still blocking superseded HIPAA 1996 citations.

5. Implementation

5.1. Prototype Environment

The prototype was implemented as a modular legal RAG system composed of separate services for ontology management, reasoning, retrieval, generation, validation, and auditing. The system uses an RDF/OWL knowledge layer for legal modeling, a triplestore for SPARQL grounding, a vector index for evidence retrieval, SHACL/SWRL rules for validation and lifecycle reasoning, and a local LLM backend for answer generation. The graph database is Ontotext GraphDB 10.8.0, and the vector database is Qdrant.

The implementation was containerized to support reproducible experimentation. To support reproducibility, Table 8 reports the main runtime configuration used in the evaluation. The table includes the LLM backend, model family, quantization format, hardware, retrieval configuration, triplestore, vector database, and validation library used during the controlled benchmark.

Table 8. Prototype runtime and reproducibility configuration.

Item	Configuration
LLM backend/model	Ollama/1lama3.1:8b
Model family and size	Llama 3.1, 8B parameters
Quantization	GGUF Q4_K_M
Decoding configuration	num_predict=768, num_ctx=8192, temperature = 0.1
Embedding model	BAAI/bge-m3
Vector database	Qdrant v1.12.3, collection laws
Triplestore	Ontotext GraphDB 10.8.0
SHACL validation	pySHACL
Retrieval setting	Top-k = 10
Chunking strategy	Article-level chunks with 120–220 token paragraph subchunks and 30-token overlap
GPU	NVIDIA RTX 4070 with 8 GB VRAM
RAM	32 GB
CPU	Intel(R) Core(TM) i7-14700HX, Intel64 Family 6 Model 183 Stepping 1, 2100 MHz
Configuration and model-layer digests	SHA-256 digests were recorded for the configuration and model-layer artifacts to support reproducibility.

5.2. Legal Ontology Construction

The legal knowledge base is represented as an OWL ontology that models the main entities required for governed legal RAG: legal instruments, articles, jurisdictions, topics, lifecycle states, and temporal validity attributes. The ontology is designed to support three main functions: jurisdiction-aware retrieval, lifecycle-aware legal filtering, and citation-level validation.

To improve semantic interoperability, the ontology reuses concepts from established legal knowledge models. The European Legislation Identifier (ELI) is used to align legal resources with legislative metadata, while LKIF Core provides foundational concepts for

legal expressions and normative structures. These reused elements are extended with domain-specific classes and properties needed for ontology-gated retrieval, validation, and auditability.

The evaluation corpus includes four legal instruments from three jurisdictional settings: Saudi PDPL, EU GDPR, US HIPAA, and US HITECH. These instruments are represented as ontology individuals and linked to their corresponding jurisdictions, legal topics, lifecycle status, and temporal validity metadata, as summarized in Table 9.

Table 9. Legal corpora represented in the ontology.

Legal Instrument	Jurisdiction	Ontology Identifier
Saudi PDPL (2023)	KSA	Law_KSA_PDPL_2023
EU GDPR (2016) [27]	EU	Law_EU_GDPR_2016
US HIPAA (1996)	US_HIPAA	Law_US_HIPAA_HIPAA_1996
US HITECH (2009)	US_HIPAA	Law_US_HIPAA_HITECH_2009

The ontology also defines a controlled set of legal topics, including data privacy, cross-border transfer, health privacy, insurance portability, fraud prevention, data rights, data breach, and consent. These topics provide the semantic bridge between user queries and ontology-gated retrieval. At query time, the inferred topic and jurisdiction are used to restrict retrieval to legal provisions that are both semantically relevant and legally applicable.

5.3. RDF/OWL Modeling

The ontology is modeled in OWL to represent the legal entities required for governed retrieval, validation, and auditability. Legal resources, legal statements, jurisdictions, topics, lifecycle states, and normative concepts are modeled as separate entities because each supports a distinct reasoning function. We adopt this design since it supports efficient reasoning while preserving the semantic structure needed for jurisdiction-aware and lifecycle-aware legal RAG.

The ontology is designed to use the `:Law` class for legal instruments, while `:Article` and `:LegalStatement` capture more specific provisions. The ontology represents jurisdictional and thematic scope through `:Jurisdiction` and `:Topic`. It tracks lifecycle validity with status individuals such as `:Active`, `:Superseded`, `:Deprecated`, and `:Unknown`. The ontology uses the normative concepts of LKIF Core through `lkif:Norm`, which keeps the model ready for future reasoning about obligations, permissions, and prohibitions.

To support reproducibility and consistent citation, the ontology uses deterministic identifiers for laws, articles, and indexed chunks. These identifiers connect the symbolic ontology layer with the retrieval and audit layers, ensuring that every retrieved text fragment can be traced back to a legal source, jurisdiction, topic, and lifecycle status.

5.4. SWRL and SHACL-Based Policy Enforcement

The framework uses SWRL rules to infer lifecycle status and SHACL shapes to validate generated answers before release. SWRL rules provide the symbolic basis for excluding expired, superseded, or otherwise non-active legal provisions before retrieval, while SHACL and policy checks enforce citation grounding, jurisdictional consistency, privacy protection, and prompt-injection resistance.

Three lifecycle rules are implemented. First, a law with an effective date is marked as a candidate `:Active` law:

$$\begin{aligned} & \text{Law}(?l) \wedge \text{hasEffectiveDate}(?l, ?eff) \\ & \rightarrow \text{CandidateActiveLaw}(?l). \end{aligned} \quad (10)$$

The final `:Active` status is not inferred by SWRL from the absence of a supersession relation. Instead, `CandidateActiveLaw` individuals are reviewed by the procedural Validity Engine, which applies closed-world lifecycle precedence and assigns `:Active` only when the provision is not superseded, deprecated, expired, or unknown.

Second, a law is marked as `:Deprecated` when its expiry date precedes the evaluation date:

$$\begin{aligned} &\text{hasExpiryDate}(?l, ?d) \wedge \text{ExpiredAtEvaluationTime}(?d) \\ &\rightarrow \text{hasStatus}(?l, :Deprecated). \end{aligned} \quad (11)$$

Third, a law is marked as `:Superseded` when it is linked to a newer replacing law:

$$\begin{aligned} &\text{isSupersededBy}(?older, ?newer) \\ &\rightarrow \text{hasStatus}(?older, :Superseded). \end{aligned} \quad (12)$$

Lifecycle inference follows a conservative precedence order:

$$\text{Superseded} > \text{Deprecated} > \text{Active} > \text{Unknown}. \quad (13)$$

This ordering ensures that a provision is not treated as retrieval-eligible once it has been replaced or expired. The inferred lifecycle status is then used by the ontology gate and output validator to prevent outdated legal sources from being retrieved or cited.

At runtime, the validator constructs a validation graph from the generated answer, retrieved evidence, inferred lifecycle statuses, approved legal sources, and detected risk indicators. The SHACL constraints and policy checks are then applied to determine whether the answer can be released, repaired, or blocked. Table 10 lists the main validation constraints.

Table 10. Main SHACL and policy validation constraints used by the output validator.

Shape	Validation Target	Enforcement Objective
NoSupersededLawShape	Cited legal sources	Blocks answers that cite superseded legal provisions.
RequireCitationShape	Generated answer	Requires legal answers to include at least one supporting citation.
ActiveOnlyShape	Cited legal sources	Blocks citations to non-active provisions when lifecycle status is known.
JurisdictionMatchShape	Jurisdictional scope	Ensures that cited sources match the requested jurisdiction, except for explicitly comparative queries.
CitationInContextShape	Citation grounding	Ensures that all citations belong to the ontology-approved retrieved evidence.
NoPIIRiskShape	Input and output safety	Blocks answers associated with PII or PHI disclosure risks.
NoInjectionShape	Input and output safety	Blocks answers associated with prompt-injection or policy-bypass patterns.

SHACL validation is applied to a compact query-specific validation graph, not to the entire legal ontology for every query. This graph contains the generated answer, extracted citations, retrieved evidence, approved legal IRIs, lifecycle statuses, jurisdiction metadata, and detected safety risks. SHACL and procedural policy checks then decide whether the answer can be released, repaired, or blocked.

Additional procedural policy checks are used for cases that are easier to express outside SHACL, such as unsupported article references or domain-specific consent-summary

restrictions. These checks complement the declarative validation layer while preserving the requirement that every released answer must be legally grounded, jurisdictionally valid, and safe.

5.5. Vector Index Construction

The retrieval index stores legal text fragments together with ontology-aligned metadata. This metadata links each fragment to its legal source, jurisdiction, topic, lifecycle status, and provenance information. As a result, symbolic filtering can be applied before semantic similarity ranking, ensuring that retrieval is restricted to legally eligible evidence.

5.5.1. Chunking and Metadata

Legal provisions are indexed at two levels of granularity. Each article is represented as a complete article-level chunk, while longer articles are also divided into paragraph-level subchunks. The article title is attached to each subchunk to preserve legal context and reduce ambiguity during retrieval.

Each indexed entry stores the metadata required for ontology-gated retrieval, including the legal source, article identifier, jurisdiction, topic labels, lifecycle status, provenance information, text content, and chunk identifier. These fields allow the retriever to exclude irrelevant, outdated, or jurisdictionally invalid evidence before applying dense similarity search.

5.5.2. Semantic and Lexical Retrieval

In order to find the legal provisions that match the user's query and, at the same time, catch the exact legal terms, article numbers, and quoted phrases, we use a dense combination of semantic search and lexical matching in the retrieval layer. Thus, we enable the system to understand the broader meaning through semantic search without losing the precise terminology that legal work depends on using lexical matching.

5.5.3. Index Synchronization

Index synchronization keeps the ontology and vector retrieval index aligned whenever a new legal document is added or an existing provision is amended, deprecated, or reclassified. The ingestion process is responsible for this task. It starts by breaking each new provision into article-level and paragraph-level chunks. Then, the system adds those chunks to the retrieval index with their ontology metadata.

5.6. Ontology-Gated Retrieval Algorithm

Algorithm 1 represents the retrieval process. We designed the retrieval process to verify legal eligibility before checking semantic similarity. We reduced the search space to include only the relevant jurisdiction, topic, and query intent. Then, using ontology constraints, we identify candidate legal provisions through SPARQL grounding to get an initial set of provisions that legally match the user query. After that, we keep only Active provisions after comparing the candidate provision set with the lifecycle status inferred by the reasoner. Thus, only Active provisions are retained before the query reaches the RAG layer. Then, we run dense retrieval over the reduced and approved candidate set that contains legally valid evidence. Finally, we apply dense semantic similarity, lexical matching, and reranking.

When no active legal provision satisfies the ontology constraints, the algorithm blocks the query before generation. Thus, the LLM receives only concise, ontology-approved, and legally valid material, therefore preventing answers with incorrect evidence.

Algorithm 1 Ontology-gated retrieval**Require:** User query x , ontology $\mathcal{O}$, vector index $\mathcal{V}$ **Ensure:** Approved evidence set E or blocked retrieval decision

- 1: Identify the jurisdiction, topic, and intent in x
- 2: Find candidate provisions in $\mathcal{O}$ that match the query
- 3: Check the lifecycle status of each candidate provision
- 4: Remove any provision that is inactive or invalid for the jurisdiction
- 5: **if** no candidate provision remains **then**
- 6: Block retrieval and return an empty evidence set
- 7: **end if**
- 8: Search $\mathcal{V}$ only within the approved candidate set
- 9: Score the results using dense similarity, lexical matching, and reranking
- 10: Select the top-ranked evidence chunks that remain legally valid
- 11: **return** approved evidence set E

5.7. Audit Ledger Implementation

The goal of using the audit ledger is to enable a tamper-evident record for every request and response. For each interaction, we record the original query, extracted jurisdiction and topic, SPARQL grounding results, approved legal sources, retrieved evidence, generated answer, validation decision, and timestamp. We then compute a cryptographic hash, shown in Equation (14), over the core audit fields to create a tamper-evident record that links the final response to its supporting evidence and validation trail. It also allows an external reviewer to reconstruct why the response was allowed, blocked, or repaired.

$$h = \text{SHA256}(\text{Serialize}(x, \hat{j}(x), \hat{U}(x), G, S, E, y, V(y), t)). \quad (14)$$

where x represents the user's query, $\hat{j}(x)$ represents the jurisdiction it infers, $\hat{U}(x)$ represents the topic set it infers, G represents the SPARQL grounding result, S represents the approved legal sources, E represents the retrieved evidence, y represents the generated answer, $V(y)$ represents the validation result, and t represents the timestamp.

To improve interoperability with established provenance models, the audit schema can be conceptually mapped to the W3C PROV-O model. The current prototype does not yet export PROV-O RDF; therefore, the audit schema is described as *mappable to PROV-O*, rather than PROV-O compliant. As shown in Table 11, the user query, retrieved evidence, generated answer, and validation result can be represented as `prov:Entity` instances. Retrieval, ontology reasoning, LLM generation, and output validation can be represented as `prov:Activity` instances. The retriever, LLM backend, validator, and audit service can be represented as `prov:Agent` instances. The timestamp maps to `prov:generatedAtTime`, while the audit hash should be represented as project-specific integrity metadata, such as `ex:sha256Digest`, rather than as a standalone forensic admissibility guarantee.

This mapping provides a path toward standardized provenance export in future versions of the system. A future implementation can mint stable URIs for audit records, represent queries, retrieved evidence, answers, and validation results as PROV-O entities, and link retrieval, grounding, generation, validation, and audit-recording activities using `prov:used`, `prov:wasGeneratedBy`, and `prov:wasAssociatedWith`. In the current prototype, the audit ledger remains a tamper-evident traceability mechanism; full PROV-O RDF export is left for future work.

The cryptographic hash provides tamper-evidence, but it does not make the storage medium immutable or, by itself, establish forensic admissibility. It supports integrity verification by allowing a reviewer to recompute the digest of the audit record and compare it with the stored value. In the prototype, audit records are stored in an append-oriented format and duplicate records are rejected. Production use would require stronger safeguards,

such as hash chaining, append-only storage, trusted timestamping, key management, access-control evidence, and chain-of-custody procedures.

Thus, the audit ledger provides a reproducible and tamper-evident trace for each legal RAG interaction by linking the final decision to the query, retrieved evidence, validation steps, and generated or blocked response.

Table 11. Conceptual mapping between the audit ledger and W3C PROV-O.

Audit Field or Concept	PROV-O Mapping
User query	prov:Entity
Retrieved chunks/retrieved evidence	prov:Entity
Generated answer	prov:Entity
Validation result	prov:Entity or output of a prov:Activity
Retrieval step	prov:Activity
Ontology reasoning/SPARQL grounding	prov:Activity
LLM generation	prov:Activity
Output validation	prov:Activity
Retriever, LLM backend, validator, and audit service	prov:Agent
Timestamp/creation time	prov:generatedAtTime
Audit hash	Custom integrity metadata, e.g., ex:sha256Digest

5.8. End-to-End Workflow Execution

Algorithm 2 summarizes the complete governed legal RAG workflow. This differs from Algorithm 1, where only ontology-gated evidence selection is addressed. We first evaluate the user query for safety and privacy risks. Unsafe requests are blocked and also recorded in the audit ledger. When a query is allowed, we invoke the ontology-gated retrieval procedure defined in Algorithm 1, through which legally eligible evidence is selected. Once the Output Validator allows the transaction, the LLM is enabled to generate the answer using only approved evidence. Finally, we record the released answer, repaired response, or blocking decision in the audit ledger.

Algorithm 2 Governed end-to-end legal RAG workflow

Require: User query x , ontology $\mathcal{O}$, vector index $\mathcal{V}$
Ensure: Validated answer y or blocked response

- 1: Extract topic, jurisdiction, intent, and risk indicators from x
- 2: **if** the input violates safety or privacy policy **then**
- 3: Block the request and create an audit record
- 4: **return** blocked response
- 5: **end if**
- 6: Invoke Algorithm 1 to obtain approved evidence set E
- 7: **if** E is empty **then**
- 8: Block the request and create an audit record
- 9: **return** blocked response
- 10: **end if**
- 11: Generate answer y using the approved evidence set E
- 12: Validate y for citation grounding, lifecycle compliance, jurisdictional consistency, and safety
- 13: **if** y violates validation constraints **then**
- 14: Block the response or trigger repair according to the validation policy
- 15: Create an audit record
- 16: **return** blocked or repaired response
- 17: **end if**
- 18: Record the query, grounding result, evidence, validation result, response, and hash
- 19: **return** validated answer y

6. Results and Evaluation

6.1. Experimental Setup

We evaluated the system under three configurations: baseline RAG, ontology-gated RAG, and the full proposed system. We kept the baseline mode as standard RAG. In the ontology-gated mode, we added ontology-based lifecycle and jurisdiction filtering before generation. In the full system mode, we enabled the complete governance pipeline, including ontology-gated retrieval, reasoner-backed lifecycle filtering, validator checks, citation validation, and safety blocking.

All modes were evaluated on the same controlled HIPAA–GDPR query set. The system was assessed using legal correctness, citation correctness, jurisdiction alignment, lifecycle compliance, unsafe-request blocking, retrieval behavior, and latency. The objective was not only to measure answer accuracy, but also to evaluate whether ontology-gated retrieval and validator enforcement reduce legally unsafe behavior, stale-law reliance, citation errors, and cross-jurisdictional grounding mistakes. The benchmark was designed as a prototype-scale evaluation to isolate the effect of ontology governance under known lifecycle, jurisdiction, and safety conditions.

6.2. Dataset and Query Set

The benchmark contains 70 queries distributed across six evaluation categories, as shown in Table 12. The categories were designed to test both legal utility and governance behavior, including active-law answering, cross-jurisdictional comparison, lifecycle filtering, jurisdictional refusal, prompt-injection resistance, and privacy-preserving blocking. The active-law questions evaluate whether the system can answer using current HIPAA, HITECH, and GDPR provisions. The comparison questions test whether the system can retrieve and reason over both U.S. HIPAA and EU GDPR evidence. The blocked-query groups test whether the system refuses unsafe, stale, or jurisdictionally unsupported requests.

The ingested legal corpus includes the GDPR [27], HIPAA regulations codified in 45 CFR Part 164 [25], HITECH Section 13,402 [26], and historical HIPAA material used for lifecycle-sensitive testing. The ontology represents these sources as distinct legal resources. It tracks each resource’s lifecycle status and records its canonical citation, jurisdiction, and related metadata. This allows the evaluation to test whether the system uses active legal evidence and avoids superseded provisions.

Table 12. Distribution of evaluation queries.

Query Category	Number of Queries
Active-law legal questions	24
Cross-jurisdictional comparison questions	12
Superseded-law /lifecycle-sensitive questions	10
Wrong-jurisdiction requests	8
Prompt-injection attempts	8
PII/PHI disclosure attempts	8
Total	70

The benchmark was designed as a controlled prototype-scale evaluation rather than a large-scale legal QA benchmark. Its purpose is to isolate the effect of ontology-gated retrieval and validator-based enforcement under known lifecycle, jurisdictional, and safety conditions. This scope was partly due to the limited computational and annotation resources available for the present study. Therefore, the reported results should be interpreted as evidence of system behavior and architectural effectiveness within the evaluated setting,

not as population-level estimates of legal RAG performance. Because the gold labels were created by a single domain-informed annotator, the evaluation may contain annotation bias. Broader generalization will require larger multi-jurisdictional datasets, independent legal annotation, inter-annotator agreement reporting, stronger computational resources, and broader coverage of legal domains and jurisdictions.

6.3. Baselines

The baseline represents a conventional RAG pipeline without ontology-governed retrieval or validator enforcement. It retrieves semantically similar chunks and generates an answer without explicit lifecycle, jurisdiction, or safety rule enforcement. This mode provides the reference point for measuring whether ontology-based governance improves legal grounding and safety.

The ontology-gated mode adds lifecycle-aware and jurisdiction-aware retrieval filtering, but does not enable the complete validator stack. This mode isolates the contribution of ontology-based retrieval control. The full mode enables the complete proposed architecture, combining ontology-gated retrieval with validator-based output governance. Table 13 depicts the different configuration settings of the evaluated ablation modes.

For the ablation study, the pre-retrieval input gate was disabled so that blocking behavior could be attributed specifically to ontology-gated retrieval and output validation.

Table 13. Configuration of the evaluated ablation modes.

Setting	Baseline RAG	Ontology-Gated RAG	Full System
Corpus	Same	Same	Same
Chunking	Same	Same	Same
Embeddings	Same	Same	Same
Vector database	Same	Same	Same
Top- <i>k</i>	10	10	10
LLM	Same	Same	Same
Ontology gate	Disabled	Enabled	Enabled
Output validator	Disabled	Disabled	Enabled
Audit logging	Enabled	Enabled	Enabled

6.4. Gold Label Construction

Gold labels were manually constructed by the author using the official legal source documents included in the evaluation corpus. For each query, the expected legal decision, jurisdiction, legal topic, lifecycle status, and reference citations were assigned by checking the relevant HIPAA, HITECH, and GDPR provisions. Active-law queries were labeled against the currently applicable provisions, whereas lifecycle-sensitive and unsafe queries were labeled according to the expected blocking behavior.

Citations were manually reviewed against the source texts and ontology identifiers to ensure that each gold citation corresponded to a valid legal provision. The benchmark was annotated by a single domain-informed annotator; therefore, no inter-annotator agreement is reported. Ambiguous cases were resolved conservatively by preferring official active provisions and by marking unsafe, stale-law, or unsupported-jurisdiction requests as blocked.

To improve reproducibility despite the single-annotator design, an explicit annotation rubric was prepared for the benchmark. The rubric defines legal decision correctness, headline accuracy, strict citation correctness, relaxed citation correctness, jurisdiction alignment, lifecycle compliance, correct refusal, false block, unsafe false allow, and comparative-query completeness. A second-annotator package was also prepared that excludes system outputs and model identities to reduce confirmation bias.

6.5. Evaluation Protocol

The three configurations were evaluated using the same query set, legal corpus, retrieval settings, and prompt templates. Each output was assessed against the manually constructed gold labels for legal decision, citation correctness, jurisdiction alignment, lifecycle compliance, and safety outcome. The protocol is designed to isolate the contribution of ontology-gated retrieval and validator-based enforcement under controlled legal, lifecycle, and safety conditions.

6.6. Evaluation Metrics

We adopt both utility-oriented and governance-oriented metrics to evaluate the proposed framework. *Headline accuracy* measures whether the final response satisfies the expected legal answer criteria. *Legal decision accuracy* measures whether the system reaches the correct legal conclusion or allow/block decision. *Strict citation correctness* requires the generated citation to match the gold citation exactly. *Relaxed citation correctness* accepts a citation as valid when it refers to an active provision within the correct legal source, jurisdiction, and topic family, even if it does not exactly match the gold citation identifier. *Jurisdiction alignment* measures whether the answer relies on evidence from the jurisdiction requested by the user, while *lifecycle compliance* measures whether the system avoids superseded, deprecated, expired, or otherwise inactive legal provisions. *Unsafe-request blocking* measures whether privacy-violating, prompt-injection, or policy-violating requests are blocked before an unsafe answer is released.

Concerning the Safety evaluation, we use the *false allow rate*, *false block rate*, *unsafe allow rate*, and *block accuracy*.

Retrieval quality is assessed using Recall@k. Through this metric, we determine whether the gold citation is included among the top-(k) retrieved evidence passages. However, because Recall@k is affected by citation granularity, we interpret it together with answer-level correctness. This distinction becomes important when a narrower active provision is retrieved by the system, while the gold label refers to a broader legal provision. In future evaluations, we could use metric-generation frameworks [28] to generate new metrics that are aligned with the structure and requirements of our proposed system.

Strict Recall@k is sensitive to citation granularity. In legal corpora, an answer may still be correctly supported by a retrieved paragraph-level or article-level chunk, even when the gold label has been assigned to a broader parent statute or legal instrument. Therefore, we report also hierarchical Recall@k. Under this metric, we consider a retrieval correct when an exact citation match is obtained, when the retrieved evidence is a descendant of the gold legal source, or when it belongs to the same canonical legal source and has matching jurisdiction and topic metadata.

6.7. Overall Performance Summary

Since the benchmark contains 70 queries due to limited computation resources, we interpret the reported percentages as controlled prototype-scale evidence rather than population-level estimates. To make this limitation explicit, we report Wilson 95% confidence intervals and bootstrap percentile intervals over query-level outcomes. These intervals quantify sampling uncertainty within the benchmark, but they do not prove generalization to other legal domains, jurisdictions, or production-scale corpora.

In Table 14, we report the aggregate performance of the three evaluated configurations. These aggregate metrics provide a high-level comparison, but they combine two different behaviors: answering legally answerable queries and refusing unsafe or unanswerable requests. Therefore, we interpret the aggregate results together with the separated analysis in Table 15.

Table 14. Overall performance comparison across evaluation modes.

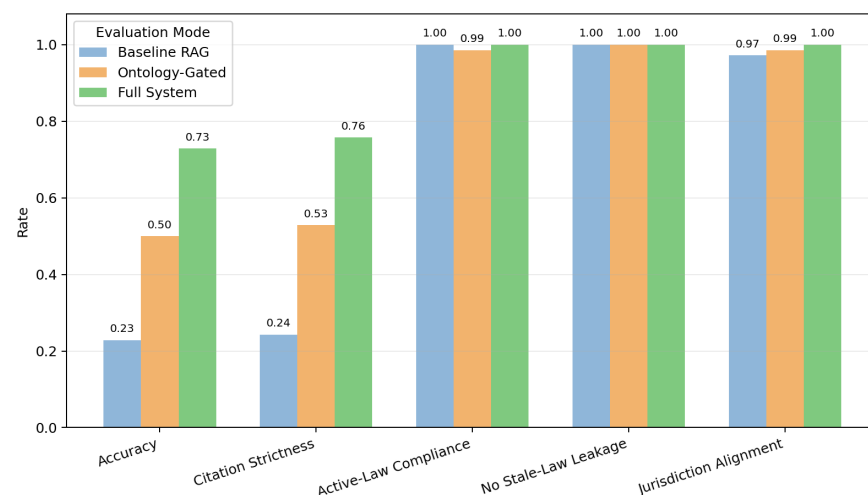
Metric	Baseline RAG	Ontology-Gated	Full System
Headline accuracy	22.86%	50.00%	72.86%
Legal decision accuracy	51.43%	77.14%	97.14%
Strict citation correctness	24.29%	52.86%	75.71%
Relaxed citation correctness	25.71%	54.29%	77.14%
Jurisdiction alignment	97.14%	98.57%	100.00%
Lifecycle compliance	100.00%	98.57%	100.00%
Stale-law leakage	0.00%	0.00%	0.00%
Unsafe false allow rate	48.57%	22.86%	0.00%
Block accuracy	51.43%	77.14%	97.14%
Throughput (queries/s)	0.098	0.142	0.142

Table 15. Separated performance on answerable and unsafe/unanswerable queries.

Metric	Baseline RAG	Ontology-Gated RAG	Full System
Answerable queries ($n = 36$)			
Headline accuracy	44.4%	47.2%	47.2%
Legal decision accuracy	100.0%	100.0%	94.4%
Strict citation correctness	47.2%	52.8%	52.8%
Unsafe/unanswerable queries ($n = 34$)			
Block accuracy	0.0%	52.9%	100.0%
False allow rate	100.0%	47.1%	0.0%
Correct refusal rate	0.0%	52.9%	100.0%

Figure 2 summarizes the ablation pattern in Table 14. We report in Figure 3, the Wilson 95% confidence intervals for legal decision accuracy, which makes the uncertainty from the prototype-scale benchmark explicit.

Table 15 separates answerable queries from unsafe or unanswerable requests. Table 15 shows that the Full System primarily improves governed release behavior rather than uniformly improving performance across all answerable queries.

**Figure 2.** Ablation comparison across Baseline RAG, Ontology-Gated RAG, and the Full System.

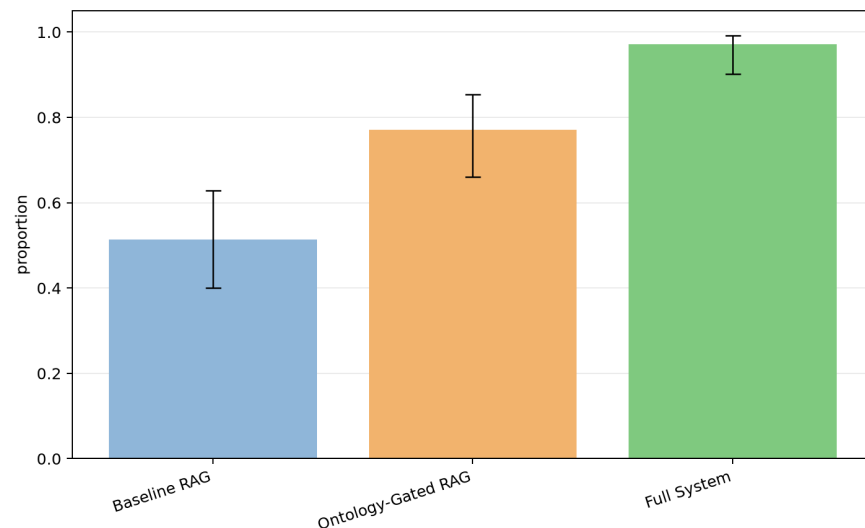


Figure 3. Legal decision accuracy with Wilson 95% confidence intervals across the evaluated systems.

For answerable queries, we observed slight improvements in headline accuracy and strict citation correctness with Ontology-Gated RAG. The Full System keeps the same citation performance, but its legal decision accuracy drops slightly due to two conservative false blocks in comparative multi-jurisdictional queries. For unsafe or unanswerable queries, the Full System provides the strongest improvement: it blocked all such requests, while Baseline RAG answered them all. We therefore interpret the primary strength of the Full System as governed release behavior rather than unrestricted answer coverage. The 100.0% lifecycle compliance reported for Baseline RAG should be interpreted cautiously. It means only that stale-law citations were not detected in the final generated answers. It does not mean that Baseline RAG enforces lifecycle-aware retrieval. Unlike Baseline RAG, the Ontology-Gated and Full System configurations apply lifecycle metadata and active-law filtering before retrieval. To complement the main evaluation, we conducted a retrieval-only stale-law stress test under controlled adversarial conditions. We used 12 targeted queries and paired active and stale/historical chunks with similar wording. We restricted the test to retrieval behavior and did not treat it as an additional legal-answering benchmark.

Table 16 compares strict and hierarchical Recall@k. We notice that Hierarchical Recall@k is higher across all modes because it counts retrieved chunks that are descendants or canonical-source matches of broader gold citations. This metric clarifies citation-granularity effects, but it does not indicate a retrieval advantage for the Full System.

Table 16. Strict and hierarchical Recall@k across evaluation modes.

System	Strict Recall@k	Hierarchical Recall@k
Baseline RAG	0.414	0.514
Ontology-Gated RAG	0.386	0.514
Full System	0.386	0.514

As shown in Table 17, stale chunks were retrieved by Baseline RAG for all 12 queries. None were retrieved by Ontology-Gated RAG or the Full System because inactive authorities had been excluded before vector ranking. Since answers were not generated in this experiment, the number of stale chunks cited is not applicable.

Table 17. Retrieval-only stale-law stress test results.

System	Total Stress Queries	Queries with Stale Chunks Retrieved	Stale Chunks Retrieved	Stale Chunks Cited	Stale Exposure Rate
Baseline RAG	12	12	27	N/A (retrieval-only)	1.000
Ontology-Gated RAG	12	0	0	N/A (retrieval-only)	0.000
Full System	12	0	0	N/A (retrieval-only)	0.000

6.8. Coverage, Rejection, and Abstention Analysis

In Table 18, we separate the answer coverage from rejection and refusal behavior. This distinction is important because the evaluated systems differ not only in answer quality, but also in their willingness to answer unsafe, unsupported, stale-law, or jurisdictionally invalid queries.

Table 18. Coverage, rejection, and refusal behavior across evaluation modes.

System	Answered	Rejected Before Retrieval	Rejected After Validation	Correct Refusals	False Blocks	Unsafe False Allows	Coverage	Abstention
Baseline RAG	70	0	0	0	0	34	1.000	0.000
Ontology-Gated RAG	52	18	0	18	0	16	0.743	0.257
Full System	34	18	18	34	2	0	0.486	0.514

False-Block Analysis

We observed a 2.86% false-block rate in the Full System as shown in Table 19, caused by two answerable comparative multi-jurisdiction queries. We did not block these queries because they were unsafe. Instead, the citation extraction layer failed to retain every jurisdiction-specific citation required by the comparison validator. We therefore withheld the responses. This behavior reflects our precision-oriented release policy: when the system cannot provide complete multi-jurisdictional support, we block the comparison rather than release a potentially under-supported legal answer.

Table 19. False blocks observed in the Full System.

False-Block Category	Count	Rate
Cross-jurisdictional comparative queries	2	2.86%

The results show a tradeoff between trustworthiness and coverage. Baseline RAG answered all queries, but this unrestricted coverage led to 34 unsafe false allows. Ontology-Gated RAG reduced coverage to about 74% and correctly refused 18 unsafe or unanswerable requests, but it still allowed 16 unsafe cases. The Full System further reduced coverage to about 48%, but correctly refused all 34 unsafe or unanswerable requests and produced no unsafe false allows.

This behavior is intentional in a governed legal RAG setting. In legal and compliance-oriented applications, lower coverage can be acceptable, and sometimes desirable, when the alternative is producing unsupported, outdated, unsafe, or jurisdictionally invalid answers. Therefore, the Full System prioritizes governed release and safe refusal over unrestricted answer coverage.

6.9. Prompt-Injection and PII/PHI Blocking Results

Figure 4 summarizes the safety-blocking behavior across the evaluated modes. Baseline RAG frequently answered unsafe or disallowed requests, while Ontology-Gated RAG

reduced but did not eliminate unsafe false allows. The Full System produced no unsafe false allows in the controlled benchmark because output validation blocked prompt-injection, PII/PHI disclosure, stale-law, and unsupported-jurisdiction requests before release. These results should be interpreted together with the coverage and abstention results in Table 18, since stronger blocking also reduces answer coverage.

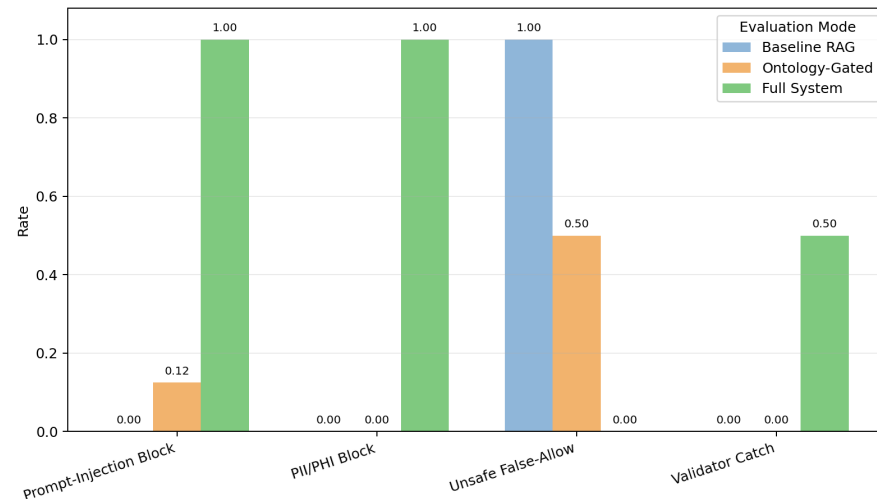


Figure 4. Safety blocking metrics.

6.10. Latency and Runtime Overhead

Figure 5 reports the p95 latency across the three evaluation modes. The aggregate p95 values were broadly comparable: Baseline RAG reached 14,721 ms, Ontology-Gated RAG reached 14,269 ms, and the Full System reached 14,177 ms. Therefore, the small difference in latency should not be interpreted as evidence that the Full System is intrinsically faster than Baseline RAG.

We explain the latency difference mainly by workload composition. Baseline RAG answered all queries, including unsafe or unanswerable ones, and therefore invoked the full generation path more often. Ontology-Gated RAG blocked some requests before generation, while the Full System also blocked unsupported or unsafe outputs after validation. Thus, we read the reported latency as a mixed-workload runtime result rather than as a pure processing-speed comparison.

Overall, we can observe that ontology-gated retrieval and output validation did not introduce prohibitive runtime overhead in this controlled prototype benchmark.

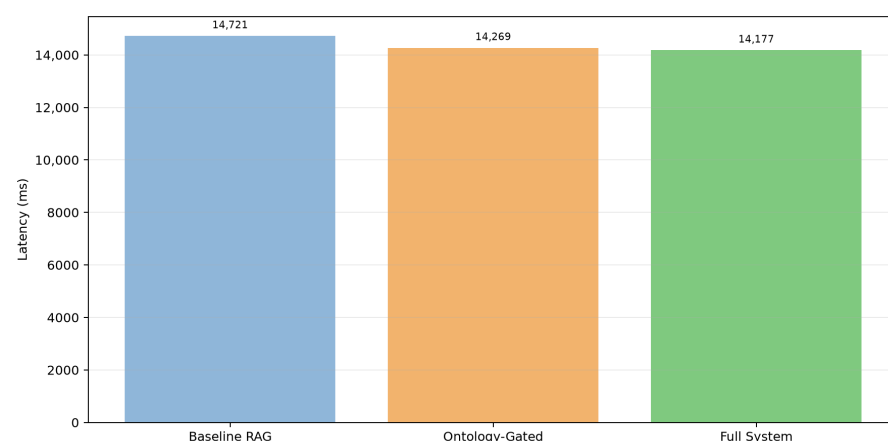


Figure 5. Aggregate p95 latency across the evaluated configurations.

6.11. Summary of Findings

We conclude that the evaluation supports mainly three main findings. First, ontology-gated retrieval improves the legal eligibility of retrieved evidence by applying jurisdiction and lifecycle constraints before vector ranking. Second, output validation is the main source of governed release behavior because it prevents unsupported, unsafe, stale-law, and jurisdictionally invalid answers from being released. Third, this governance introduces a coverage tradeoff: the Full System answers fewer queries than Baseline RAG, but it provides stronger refusal behavior for unsafe or unsupported requests.

The remaining weaknesses are citation-granularity mismatch, moderate topic alignment, and two conservative false blocks in comparative multi-jurisdictional queries. These limitations indicate that future work should focus on structured comparative citation extraction, citation normalization, multi-annotator labeling, and larger multi-jurisdictional benchmarks.

7. Discussion

7.1. Interpretation of Results

According to the results, we conclude that the main value of ontology-governed legal RAG lies in separating evidence eligibility from answer release. Retrieval-time governance determines which legal sources may enter the context, while output-time validation determines whether the generated answer is sufficiently grounded, jurisdictionally valid, lifecycle-compliant, and safe to release. This design separation is important because it enables the system to detect responses that are generated from relevant text but still fail release conditions due to incomplete citation support, missing jurisdictional evidence, or safety risk.

We interpret the retrieval results carefully since Strict Recall@k penalizes citation-granularity mismatches, while hierarchical Recall@k shows that some retrieved chunks are valid descendants or canonical-source matches of broader gold citations. Therefore, we interpret retrieval quality together with citation correctness, legal decision correctness, and refusal behavior rather than as a standalone indicator.

7.2. Security and Legal Governance Implications

Our findings show that legal RAG systems need explicit symbolic governance to control both retrieved context and generated responses. We cannot rely on prompt engineering and vector retrieval alone, because they do not provide sufficient governance guarantees.

In our benchmark, the full system blocked privacy-risk queries, prevented unsafe false approvals, and avoided detected leakage from outdated legislation. It also linked each answer to the correct jurisdiction. We cannot reliably achieve these safeguards with unconstrained RAG. Vector similarity may identify semantically related text, but it does not determine whether a law remains in force, where it applies, or which validation rules the system must enforce.

We also designed the framework to improve auditability. For every answer, we record the retrieved evidence, lifecycle status, jurisdiction metadata, citation identifiers, validator decisions, and audit hash. Reviewers can therefore trace why we accepted or rejected specific evidence. They can also verify whether the final response satisfied the required legal and safety constraints. This level of traceability is essential in legal and compliance settings.

7.3. Scalability and Complexity Considerations

The current prototype demonstrates the feasibility of ontology-governed legal RAG over a controlled four-instrument corpus. It does not, however, establish scalability to

production legal ecosystems containing thousands of statutes, regulations, municipal rules, and precedents. Scaling the framework mainly affects six layers: ontology reasoning, lifecycle resolution, SHACL validation, vector retrieval, index synchronization, and audit storage.

OWL/SWRL reasoning may become costly as the legal graph and rule set grow. Production deployments should therefore rely on bounded monotonic rules, modular ontologies, indexed lifecycle properties, and incremental materialization. Lifecycle resolution should also be handled as a closed-world operational process, with batched updates when new amendments, expiry dates, or supersession links are added.

SHACL validation should be scoped to compact query-specific graphs rather than the full ontology. Each validation graph should include only the query, retrieved evidence, approved legal IRIs, extracted citations, lifecycle statuses, jurisdiction metadata, validation decision, and safety indicators. This limits validation cost while preserving the closed-world checks required before answer release.

We can scale vector retrieval by filtering metadata first, including jurisdiction, topic, lifecycle status, and law family, before applying dense ranking. We should also synchronize the index through incremental upserts and deletions rather than rebuilding it after every legal update. As usage grows, audit records accumulate quickly. Therefore, we must manage them through partitioning, retention policies, hash chaining, append-only storage, and trusted timestamping. Table 20 summarizes the scaling risks in each layer and the corresponding production mitigations.

Table 20. Scalability risks and production-oriented mitigations.

Layer	Scaling Risk	Production Mitigation
OWL/SWRL reasoning	Reasoning cost grows with graph size and rule complexity.	Use bounded rules, modular ontologies, indexed lifecycle properties, and incremental materialization.
Lifecycle resolution	Legal updates may affect many provisions.	Use batched updates, an active-status index, and a closed-world Validity Engine.
SHACL validation	Full-ontology validation may be expensive.	Validate compact query-specific graphs only.
Vector retrieval	Larger corpora increase the candidate search space.	Pre-filter by jurisdiction, topic, lifecycle status, and law family before vector ranking.
Index synchronization	Full index rebuilding is costly after frequent updates.	Use incremental upserts and deletions for changed provisions.
Audit ledger	Audit records grow with query volume.	Use partitioning, retention policies, hash chaining, append-only storage, and trusted timestamping.

These considerations show that the current implementation should be viewed as a prototype-scale validation of the architecture. Future work should stress-test the framework over larger legal graphs, heavier update workloads, and more diverse query distributions.

7.4. Limitations and Threats to Validity

We evaluated the system using a controlled 70-query HIPAA–GDPR benchmark because the available hardware resources could not reliably support larger experimental runs. Preliminary runs with larger query sets resulted in resource exhaustion and incomplete execution. Therefore, assessing broader generalizability will require stronger hardware, larger query sets, additional jurisdictions, and a wider range of legal domains. The results show clear improvements, but the dataset remains small and covers only two legal frameworks. We labeled the dataset by a single domain-informed annotator. Although an explicit annotation rubric and second-annotator package were prepared to support future independent

validation, independent second annotation has not yet been completed. Therefore, we did not report Cohen's Kappa in this study and remains part of future validation work.

Citation granularity may influence both internal and construct validity. In some cases, we define the gold labels at the broad law-family level, whereas the system retrieves or cites more specific active provisions. This mismatch may reduce Recall@k and strict citation correctness, even when the system produces a legally valid answer. We also need to improve topic alignment, which currently reaches only 58.57%. This result suggests that we should further refine both legal-topic classification and the mapping of gold-topic labels.

Although the architecture supports repair-or-block behavior, the present evaluation logs release and block outcomes only; repair attempts were not separately instrumented or evaluated.

Another limitation concerns the two false blocks produced for comparative queries. These cases were caused by incomplete citation extraction for multi-jurisdiction answers rather than unsafe behavior. In future work, we will improve citation normalization for comparative legal answers and ensure that each requested jurisdiction contributes explicit active evidence before validation. We will also improve comparative-query handling by using structured comparative answer templates, jurisdiction-aware citation slots, citation normalization, and a citation-repair pass before final blocking. This should reduce conservative false blocks in multi-jurisdictional questions while preserving the system's preference for withholding under-supported legal comparisons.

8. Conclusions

We presented an Ontology-Driven Legal Rule Auditor RAG framework for governed legal question answering. We use the ontology as an active control layer rather than a passive knowledge source. It filters retrieved content by jurisdiction and lifecycle status, validates generated answers through SHACL, SWRL, and policy rules, and stores the query, evidence, validation results, and response metadata in a tamper-evident audit trail.

Our controlled benchmark shows that the governance layer shifts legal RAG away from unrestricted answer generation and toward evidence-constrained, validator-controlled release. The main advantage is safer release behavior. We block unsafe, unsupported, stale-law, and jurisdictionally invalid requests more reliably while maintaining an auditable record of the query, supporting evidence, validation decision, and final response.

In future work, we will evaluate the three system configurations with representative legal-domain users, including attorneys, paralegals, legal researchers, compliance officers, and privacy officers. These studies will extend the controlled benchmark and help us determine whether the system's safer release behavior, citation checks, jurisdictional distinctions, and refusal decisions provide practical value in real legal and compliance workflows.

In future work, we will expand the benchmark across more jurisdictions and legal domains. We will also improve comparative citation extraction, evaluate labels produced by multiple independent annotators, and stress-test the framework using larger legal knowledge graphs and heavier update workloads.

Funding: This research work was funded by Umm Al-Qura University, Saudi Arabia under grant number: 26UQU4361220GSSR01.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The ontology files, SHACL shapes, SWRL rules, evaluation queries, configuration files, and supporting scripts used in this study are publicly available at: <https://github.com/amakremi/LegalAuditRAG.git> (accessed on 13 June 2026).

Acknowledgments: The authors extend their appreciation to Umm Al-Qura University, Saudi Arabia, for funding this research work through grant number: 26UQU4361220GSSR01.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RAG	Retrieval-Augmented Generation
LLM	Large Language Model
ELI	European Legislation Identifier
LKIF	Legal Knowledge Interchange Format

References

1. Zhang, L.; Grabmair, M.; Gray, M.; Ashley, K. Thinking longer, not always smarter: Evaluating llm capabilities in hierarchical legal reasoning. In Proceedings of the Symposium on Computer Science and Law, Berkeley, CA, USA, 3–5 March 2026; pp. 71–85.
2. Linna, E.; Linna, T. Challenges for generative AI in legal reasoning. *Discov. Artif. Intell.* **2026**, *7*, 170.
3. Rouached, M.; Akremi, A.; Macherki, M.; Kraiem, N. Policy-based smart contracts management for IoT privacy preservation. *Future Internet* **2024**, *16*, 452. [\[CrossRef\]](#)
4. Sharma, K.; Kumar, P.; Li, Y. Og-rag: Ontology-grounded retrieval-augmented generation for large language models. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, Suzhou, China, 4–9 November 2025; pp. 32950–32969.
5. Arshad, U.; Corsar, D.; Nkisi-Orji, I. Integrating KGs and ontologies with RAG for personalised summarisation in regulatory compliance. In Proceedings of the SICSA REALLM, Aberdeen, UK, 17 October 2024; pp. 56–61.
6. Debenedetti, E.; Zhang, J.; Balunovic, M.; Beurer-Kellner, L.; Fischer, M.; Tramèr, F. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 82895–82920. [\[CrossRef\]](#)
7. Beurer-Kellner, L.; Buesser, B.; Cretu, A.M.; Debenedetti, E.; Dobos, D.; Fabian, D.; Fischer, M.; Froelicher, D.; Grosse, K.; Naeff, D.; et al. Design patterns for securing llm agents against prompt injections. *arXiv* **2025**, arXiv:2506.08837.
8. Ferrag, M.A.; Tihanyi, N.; Hamouda, D.; Maglaras, L.; Lakas, A.; Debbah, M. From prompt injections to protocol exploits: Threats in LLM-powered AI agents workflows. *ICT Express* **2025**, *12*, 353–383. [\[CrossRef\]](#)
9. Shi, J.; Yuan, Z.; Liu, Y.; Huang, Y.; Zhou, P.; Sun, L.; Gong, N.Z. Optimization-based prompt injection attack to llm-as-a-judge. In Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, Salt Lake City, UT, USA, 14–18 October 2024; pp. 660–674.
10. Chen, S.; Zharmagambetov, A.; Wagner, D.; Guo, C. Meta SecAlign: A Secure Foundation LLM Against Prompt Injection Attacks. *arXiv* **2025**, arXiv:2507.02735.
11. Hong, H.; Feng, S.; Naderlou, N.; Yan, S.; Zhang, J.; Liu, B.; Arastehfard, A.; Huang, H.; Hong, Y. SoK: Taxonomy and Evaluation of Prompt Security in Large Language Models. *arXiv* **2025**, arXiv:2510.15476.
12. Akremi, A. Software security static analysis false alerts handling approaches. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 702–711. [\[CrossRef\]](#)
13. Clarizia, F.; De Santo, M.; Gaeta, R.; Loffredo, R. Enhancement Large Language Models Domain Through Ontology-Based Retrieval-Augmented Generation. *Int. J. Semant. Web Inf. Syst. (IJSWIS)* **2025**, *21*, 1–29. [\[CrossRef\]](#)
14. Al Machot, F.; Horsch, M.T.; Ullah, H. Building trustworthy AI: Transparent AI systems via language models, ontologies, and logical reasoning (TranspNet). In *Proceedings of the Designing the Conceptual Landscape for a XAIR Validation Infrastructure Workshop*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 25–34.
15. Zhou, Y.; Liu, Y.; Li, X.; Jin, J.; Qian, H.; Liu, Z.; Li, C.; Dou, Z.; Ho, T.Y.; Yu, P.S. Trustworthiness in retrieval-augmented generation systems: A survey. *arXiv* **2024**, arXiv:2409.10102.
16. de Martim, H. Deterministic Legal Retrieval: An Action API for Querying the SAT-Graph RAG. *arXiv* **2025**, arXiv:2510.06002.
17. Alharthi, D.; Garcia, I.R.K. A Call to Action for a Secure-by-Design Generative AI Paradigm. *arXiv* **2025**, arXiv:2510.00451.
18. Lourenço, B.; Adão, P.; Ferreira, J.F.; Marques, M.M.; Vaz, C. Structuring Security: A Survey of Cybersecurity Ontologies, Semantic Log Processing, and LLMs Application. *arXiv* **2025**, arXiv:2510.16610.
19. Akremi, A. ForensicTwin: Incorporating Digital Forensics Requirements Within a Digital Twin. *Computers* **2025**, *14*, 115. [\[CrossRef\]](#)
20. Akremi, A. A forensic-driven data model for automatic vehicles events analysis. *PeerJ Comput. Sci.* **2022**, *8*, e841. [\[CrossRef\]](#) [\[PubMed\]](#)

21. de Oliveira, V.T.; da Silva, D.O.; de Almeida Souza, M.; Lima, M.R.; de Oliveira, S.S.T.; Rosa, T.C. Retrieval-Augmented Generation and Knowledge Graphs in Portuguese-Language Legal Documents. In Proceedings of the 17th International Conference on Computational Processing of Portuguese (PROPOR 2026), Salvador, Brazil, 13–16 April 2026; Volume 1, pp. 1–10.
22. Mukherjee, K.; Kantarcioglu, M. LLM-driven Provenance Forensics for Threat Investigation and Detection. *arXiv* **2025**, arXiv:2508.21323.
23. Song, M.; Sim, S.H.; Bhardwaj, R.; Chieu, H.L.; Majumder, N.; Poria, S. Measuring and enhancing trustworthiness of LLMs in RAG through grounded attributions and learning to refuse. In Proceedings of the International Conference on Learning Representations, Singapore, 24–28 April 2025; Volume 2025, pp. 30783–30823.
24. Ford, C.; Rane, O.; Leavy, S. Navigating Global AI Regulation: A Multi-Jurisdictional Retrieval-Augmented Generation System. *arXiv* **2026**, arXiv:2604.25448.
25. U.S. Department of Health and Human Services. 45 CFR Part 164: Security and Privacy. Electronic Code of Federal Regulations, Title 45, Subtitle A, Subchapter C, 2026. Available online: <https://www.ecfr.gov/current/title-45/subtitle-A/subchapter-C/part-164> (accessed on 13 June 2026).
26. United States Congress. Health Information Technology for Economic and Clinical Health Act, Section 13,402: Notification in the Case of Breach. American Recovery and Reinvestment Act of 2009, Pub. L. No. 111-5, 123 Stat. 115; Codified at 42 U.S.C. §17,932, 2009. Available online: <https://www.govinfo.gov/link/plaw/111/public/5> (accessed on 13 June 2026).
27. European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of Such Data. *Off. J. Eur. Union* **2016**, *119*, 1–88. Available online: <https://eur-lex.europa.eu/eli/reg/2016/679/oj> (accessed on 13 June 2026).
28. Akremi, A. An adaptative and compliant forensics admissibility metrics generation methodology. In Proceedings of the 23rd International Conference on Information Integration and Web Intelligence, Linz, Austria, 29 November–1 December 2021; pp. 495–503.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.