Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 4, 2026

GROUNDING LEGAL GUIDANCE IN STATUTE: A RETRIEVAL-AUGMENTED GENERATION SYSTEM FOR ACCESS TO JUSTICE IN PAKISTAN

Omama Mahrukh Usmani[1], Kaml Mirza[1,] Abdullah Iqbal[1,] Dr. Taha Shabbir[2]

[1]Department of Computing, Faculty of Engineering Sciences & Technology, Hamdard University, Main Campus, Karachi, Pakistan.
[2]Associate Professor Hamdard University, Karachi.
Corresponding author: tahashabbir51@gmail.com

DOI: https://doi.org/10.5281/zenodo.21019197

Keywords
access to justice; retrieval-augmented generation; legal NLP; large language models; case classification; pgvector; semantic search; responsible AI; Pakistan

Article History
Received on 20 March 2026
Accepted on 15 April 2026
Published on 28 April 2026

Corresponding Author: *
Omama Mahrukh Usmani*

Abstract

Pakistan's justice system carries an estimated 2.26 million pending cases, and the World Justice Project ranks the country near the bottom of its civil-justice and delay indices. For ordinary citizens, the binding obstacle is rarely the law itself but the cost and opacity of obtaining reliable first-line legal guidance. General-purpose large language models appear to fill this gap, yet the legal-hallucination literature shows they fabricate citations and misstate doctrine at alarming rates, which makes their unguarded use in a legal setting unsafe. We present the AI Justice Assistant, a retrieval-augmented generation (RAG) platform that answers lay legal questions by grounding a large language model in a curated corpus of Pakistani statutes rather than in its parametric memory. The system classifies a described problem into Civil, Criminal, or Family law, retrieves the most relevant statutory passages through pgvector similarity search, generates a grounded answer with source attribution, and recommends jurisdiction-appropriate courts and specialization-matched lawyers. It is built as a three-tier serverless application using React, Supabase edge functions, and PostgreSQL with the pgvector extension, with Google Gemini accessed through OpenRouter for embedding and generation. Acceptance testing exercised the full functional surface registration, authentication, grounded chat, classification, court and lawyer recommendation, history, and logout with all eight defined test cases passing. Non-functional measurements met or exceeded every target: a 3.2-second mean response time against a four-second budget, 420-millisecond recommendation latency, 92% case-classification accuracy, and 99.6% availability. We are explicit that these are system-acceptance results rather than a benchmarked study of answer quality, and we specify a reference-free RAGAS protocol faithfulness, answer relevancy, context precision, and context recall as the appropriate next-stage measurement. The work contributes a fully localized, end-to-end RAG architecture for Pakistani legal guidance; an integration of classification, retrieval-grounded consultation, and recommendation in a single platform; and a candid evaluation and responsible-AI discussion that situates the system against current legal-AI reliability evidence and marks the boundary between legal information and legal advice.

INTRODUCTION

1.1 The access-to-justice gap in Pakistan

Access to justice is a constitutional guarantee, yet for most Pakistani citizens it remains practically out of reach. The Law and Justice Commission of Pakistan (2024) records roughly 2.26 million cases pending across the courts, with the great majority lodged in the district and subordinate judiciary and the remainder distributed across the High Courts and the Supreme Court (Figure 1). The World Justice Project (2023) places Pakistan's civil-justice performance near the bottom of its global index and lower still on the specific dimension of unreasonable delay. Behind these aggregates lies a simple lived reality: a litigant in an ordinary civil dispute may attend court dozens of times before disposition, and the chronic shortage of judges leaves a single judicial officer responsible for tens of thousands of citizens.

The first obstacle most citizens meet, however, is not the trial itself but the difficulty of obtaining reliable preliminary guidance. Understanding one's rights under the Pakistan Penal Code, identifying the court with jurisdiction over a dispute, and locating a suitably qualified advocate all require either prior legal literacy or paid consultation. For low-income and rural communities, these prerequisites are frequently prohibitive, and the result is delayed, uninformed, or abandoned claims. The AI Justice Assistant is designed to lower precisely this first barrier.

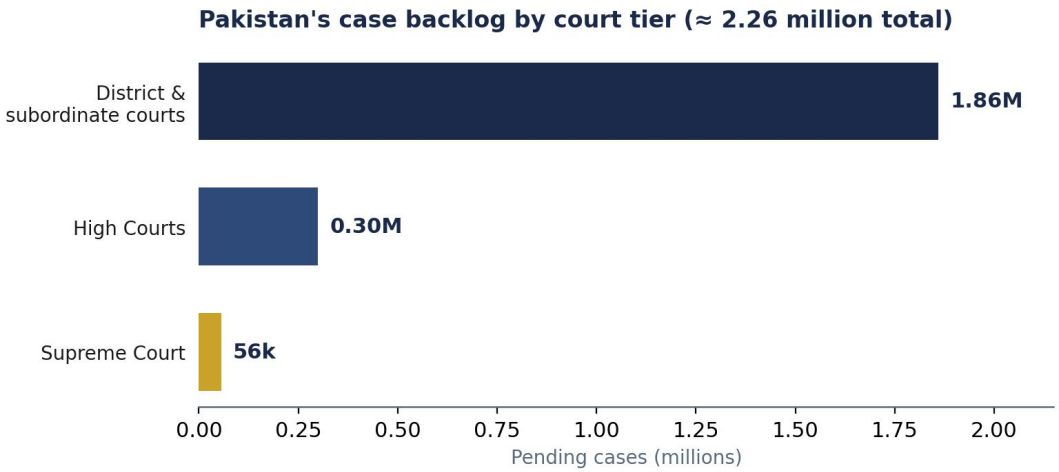

Figure 1. Distribution of pending cases across Pakistan's court tiers (Law and Justice Commission of Pakistan, 2024). The district and subordinate judiciary carries the overwhelming share of the backlog, where first-line guidance has the greatest marginal value.

1.2 Why a general-purpose model is not enough

Conversational large language models can produce fluent answers to legal questions, and their accessibility makes them an obvious candidate for closing the guidance gap. The reliability evidence, however, is sobering. In a study of more than 800,000 verifiable legal questions, Dahl et al. (2024) found that leading 2023-era models hallucinated between 58% and 88% of the time, fabricating citations, misattributing holdings, and failing to correct a user's false legal premises. A follow-up evaluation of commercial legal-research tools reported that even retrieval-enabled products still produced incorrect or misgrounded answers at a material rate (Magesh et al., 2025). The general survey literature characterizes hallucination as an intrinsic property of free-form generation rather than a defect that scale alone removes (Huang et al., 2025).

These findings carry a direct design implication. A model that answers from its trained weights cannot reliably distinguish a real provision of Pakistani law from a plausible fabrication, and in a legal setting a confident fabrication is more dangerous than a refusal. The system must therefore be constructed so that its answers are constrained by, and attributable to, an authoritative source.

A retrieval-grounded response

Retrieval-augmented generation (RAG) supplies exactly this constraint. Rather than relying on parametric memory, a RAG system retrieves relevant passages from an external corpus at query time and conditions generation on that retrieved text (Lewis et al., 2020). For legal guidance the corpus is a curated body of statutes, so the model's latitude to invent is sharply reduced and every answer can carry a citation to the provision that supports it. Crucially for a resource-constrained deployment, Li et al. (2024) show that retrieval from a small, vetted set of legal sources can match or exceed retrieval from the open internet, which means a focused national corpus is a viable foundation rather than a compromise. The AI Justice Assistant adopts this stance: it grounds

every consultation in Pakistani statutory text and surfaces the supporting sources to the user.

1.4 Contributions

This paper makes four contributions:

A localized, end-to-end RAG architecture. A complete three-tier serverless design that grounds legal consultation in Pakistani statutes, documented from data ingestion through retrieval, generation, and recommendation.

Integration of three functions in one platform. Case classification, retrieval-grounded consultation, and jurisdiction-aware court and lawyer recommendation are unified in a single citizen-facing workflow, which prior local efforts treat separately.

A transparent, appropriately scoped evaluation. Functional and non-functional acceptance results are reported in full, with an explicit statement of what they do and do not establish, and a reference-free RAGAS protocol is specified for rigorous next-stage measurement of answer quality.

A responsible-AI treatment. The system is positioned against the current legal-AI reliability literature, and its scope limits, failure modes, and the boundary between legal information and legal advice are stated plainly.

1.5 Organization

Section 2 reviews access-to-justice technology, retrieval-augmented generation, and the legal-AI reliability literature, and positions the system against existing assistants. Section 3 presents the system design and methodology, including the architecture, data model, and RAG pipeline. Section 4 describes the implementation. Section 5 reports the evaluation and specifies the proposed RAGAS protocol. Section 6 discusses strengths, limitations, threats to validity, and ethical considerations. Section 7 concludes and sets out future work.

Background and Related Work

This section establishes the theoretical and empirical foundations of the system across four strands legal technology for access to justice, retrieval-augmented generation, large language models in law and their reliability, and existing legal assistants and closes by positioning the AI Justice Assistant against the most directly comparable systems.

Legal technology and access to justice

The use of information technology to widen access to justice has progressed from

early rule-based expert systems toward modern language-model assistants. The recurring promise is to help self-represented litigants and laypeople understand rights and procedures without the cost of a consultation. Recent work frames AI-powered chatbots and retrieval-augmented self-help as a credible route to this goal (Ajmi, 2024), while emphasizing that the data on which such systems draw must be authoritative and current. Within Pakistan specifically, commentators have argued that ICT and AI-based case-management tools can reduce trial delay and administrative friction in the courts, and a growing body of work documents both the opportunity and the ethical hazards of digitizing justice. The throughline across this literature is that accessibility gains are contingent on the quality and provenance of the underlying legal content.

Retrieval-augmented generation

Retrieval-augmented generation couples a parametric generator with a non-parametric retriever, inserting passages drawn from an external corpus directly into the model's context (Lewis et al., 2020). The retrieval step typically relies on dense vector representations: encoders such as Sentence-BERT (Reimers & Gurevych, 2019) and dense passage retrieval (Karpukhin et al., 2020) map both query and documents into a shared embedding space in which semantic similarity is computed by inner product or cosine distance. The approach reduces hallucination, allows a system's knowledge to be updated by editing the corpus rather than retraining the model, and makes answers attributable to specific sources properties that recent surveys identify as the central reasons for RAG's rapid adoption in knowledge-intensive applications (Fan et al., 2024).

In the legal domain these properties are not conveniences but requirements. Wiratunga et al. (2024) combine case-based reasoning with RAG for legal question answering, retrieving precedent to structure generation; Beauchemin et al. (2024) apply RAG to Quebec automobile-insurance questions, a specialized legal-document setting; and Li et al. (2024) build a human-centric legal-QA pipeline and show that retrieval from a curated expert corpus can match internet-wide retrieval despite using orders of magnitude less data. This last result is the empirical warrant for grounding a national legal

assistant in a focused statutory corpus rather than attempting open-web coverage.

Large language models in law and the reliability problem

A vigorous research programme now benchmarks language models on legal reasoning. LegalBench assembles a collaborative suite of legal-reasoning tasks (Guha et al., 2023), LawBench probes legal knowledge across difficulty tiers (Fei et al., 2024), and domain models such as LawLLM target jurisdiction-specific applications (Shu et al., 2024); a recent survey catalogues the broader landscape of legal NLP tasks, datasets, and open challenges (Ariai et al., 2024). The consistent finding is that performance is uneven and highly sensitive to task complexity, and that fluency is a poor proxy for correctness.

The reliability evidence is the part of this literature most consequential for system design. Dahl et al. (2024) document legal hallucination rates of 58–88% on verifiable questions and show that models frequently fail to flag their own errors. Magesh et al. (2025) extend the concern to commercial, retrieval-enabled legal-research tools, finding that grounding reduces but does not eliminate misgrounded answers. Together these studies justify two design commitments adopted here: retrieval grounding to constrain the generator, and source attribution so that a user or a reviewing professional can verify any answer against the cited provision.

Existing legal assistants and the Pakistan gap

Globally, consumer legal assistants such as DoNotPay and various LawBot-style chatbots demonstrated demand for conversational legal help but were neither built on authoritative Pakistani sources nor designed for the country's court structure. Research prototypes have begun to close the localization gap elsewhere in South Asia: LawPal applies RAG to improve legal accessibility in India (Panchal et al., 2025), and several hackathon and open-source efforts target bilingual Urdu–English legal chat. What remains absent in the Pakistani setting is a single platform that combines retrieval-grounded consultation over national statutes, automated case classification, and jurisdiction-aware court and lawyer recommendation. The AI Justice Assistant is built to occupy that gap. **Table 1 positions it against the most directly comparable systems.**

Table 1. Positioning of the AI Justice Assistant against representative legal-AI systems.

System	Jurisdiction	Retrieval grounding	Integrated recommendation	Source attribution
DoNotPay / LawBot-style	US / UK	Limited	No	No
LawLLM (Shu et al., 2024)	United States	Model-centric	No	Partial
CBR-RAG (Wiratunga et al., 2024)	General legal QA	Yes (case-based)	No	Yes
LawPal (Panchal et al., 2025)	India	Yes (RAG)	No	Yes
AI Justice Assistant (this work)	Pakistan	Yes (RAG over statutes)	Yes (court + lawyer)	Yes

The comparison is qualitative and is intended to locate the contribution rather than to rank answer quality, which Section 5 addresses separately and more cautiously.

System Design and Methodology

This section documents the engineering process, the three-tier architecture, the data model, and the retrieval-augmented consultation pipeline that together constitute the AI Justice Assistant. The presentation is deliberately concrete: each component is described in terms of its responsibility, its inputs and outputs, and the design rationale that connects it to the grounding requirement established in Section 1.

Engineering process

Development followed an Agile process organized around object-oriented design, chosen so that the system could be built incrementally with regular supervisor feedback and adapted as the legal corpus and requirements were refined (Sommerville, 2015). Work proceeded in five phases across the two project semesters: requirements capture in a Software Requirements Specification; design expressed as entity-relationship, sequence, and architecture diagrams in a Software Design Specification; implementation of the core modules; testing against a defined plan; and deployment with final documentation. Each sprint targeted a single module authentication, grounded chat, classification, or

recommendation and closed with a review. Figure 2 sketches the phase timeline and Table 2 records the phase deliverables.

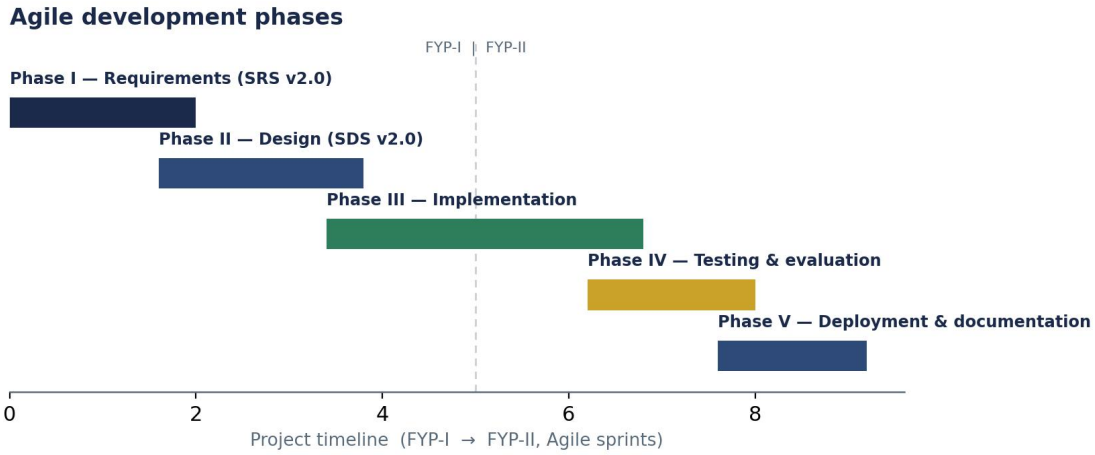

Figure 2. Agile development phases across the two project semesters. Implementation overlaps design and testing, reflecting the iterative, sprint-based process rather than a strict waterfall.

Table 2. Project phases and deliverables.

Phase	Focus	Primary deliverable
I	Requirements	Software Requirements Specification (SRS v2.0)
II	Design	Software Design Specification (SDS v2.0): architecture, ER, sequence diagrams
III	Implementation	Core modules: authentication, grounded chat, classification, recommendation
IV	Testing	Functional and non-functional acceptance against the test plan
V	Deployment	Frontend on Vercel, backend on Supabase edge functions; final documentation

Three-tier serverless architecture

The system adopts a three-tier serverless architecture (Figure 3) that separates presentation, application logic, and data, with external model services drawn in only where generation and embedding are required. A single-page React application renders the interface and routing. Application logic lives in Supabase edge functions small, independently

deployable handlers for case classification, retrieval-grounded chat, court recommendation, lawyer recommendation, and authentication. The data tier is a PostgreSQL database extended with pgvector for similarity search. Embedding and generation are delegated to Google Gemini, reached through an OpenRouter gateway. The serverless arrangement keeps each concern modular and lets the deployment scale horizontally without managing servers.

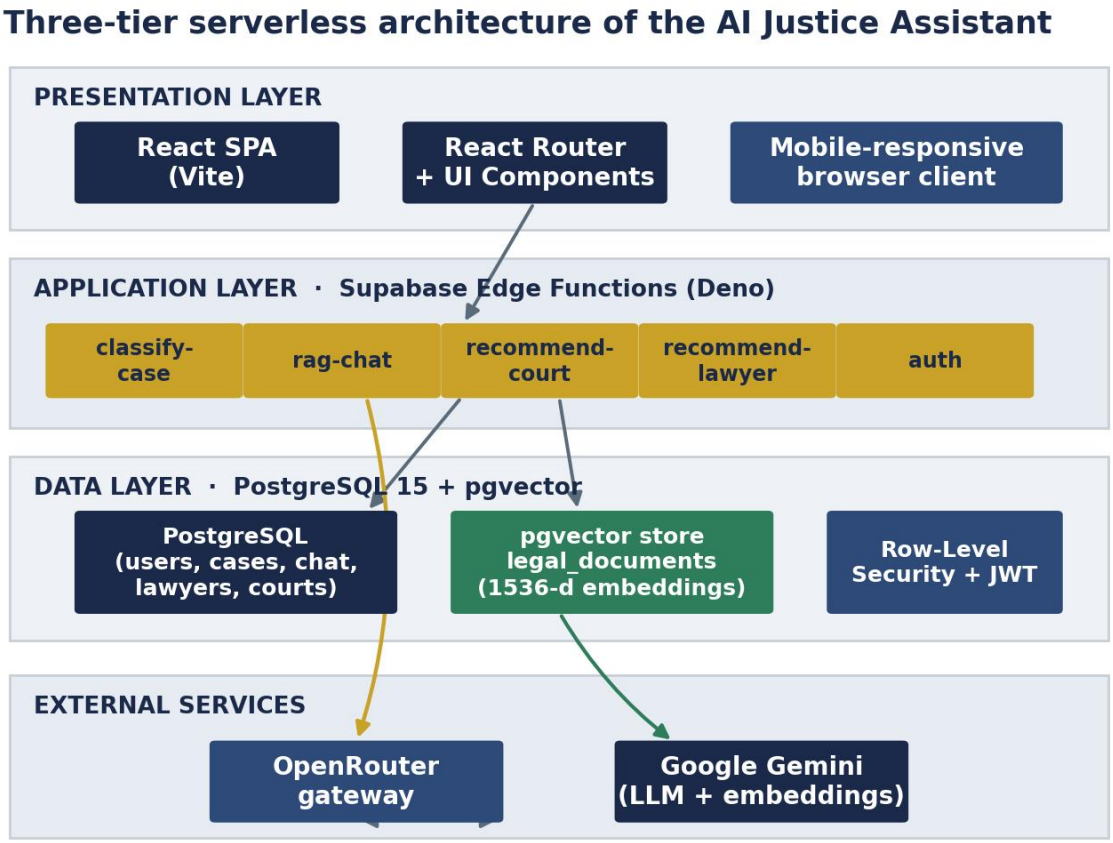

Figure 3. Three-tier serverless architecture. The presentation tier (React) calls stateless edge functions, which read and write PostgreSQL and the pgvector store and call Gemini through OpenRouter for embedding and generation.

Table 3. Technology stack and the role of each component.

Tier	Technology	Role
Presentation	React + Vite	Single-page interface, routing, conversational UI

Application	Supabase edge functions (Deno)	Classification, RAG chat, recommendation, auth logic
Data	PostgreSQL 15	Relational store for users, sessions, cases, courts, lawyers
Data	pgvector	Vector similarity search over embedded statute chunks
External	Google Gemini (via OpenRouter)	Query/document embedding and grounded text generation
Hosting	Vercel + Supabase	Frontend hosting and serverless backend execution

Data model

The relational schema (Figure 4) captures users and their consultation history, the classified cases that consultations produce, and the reference data courts and lawyers from which recommendations are drawn. A dedicated legal_documents table stores the statutory corpus together with a vector embedding column, which pgvector indexes for retrieval. Separating the embedded corpus from transactional data keeps the retrieval path simple and lets the corpus be re-embedded independently when statutes are updated. Table 4 summarizes the principal entities.

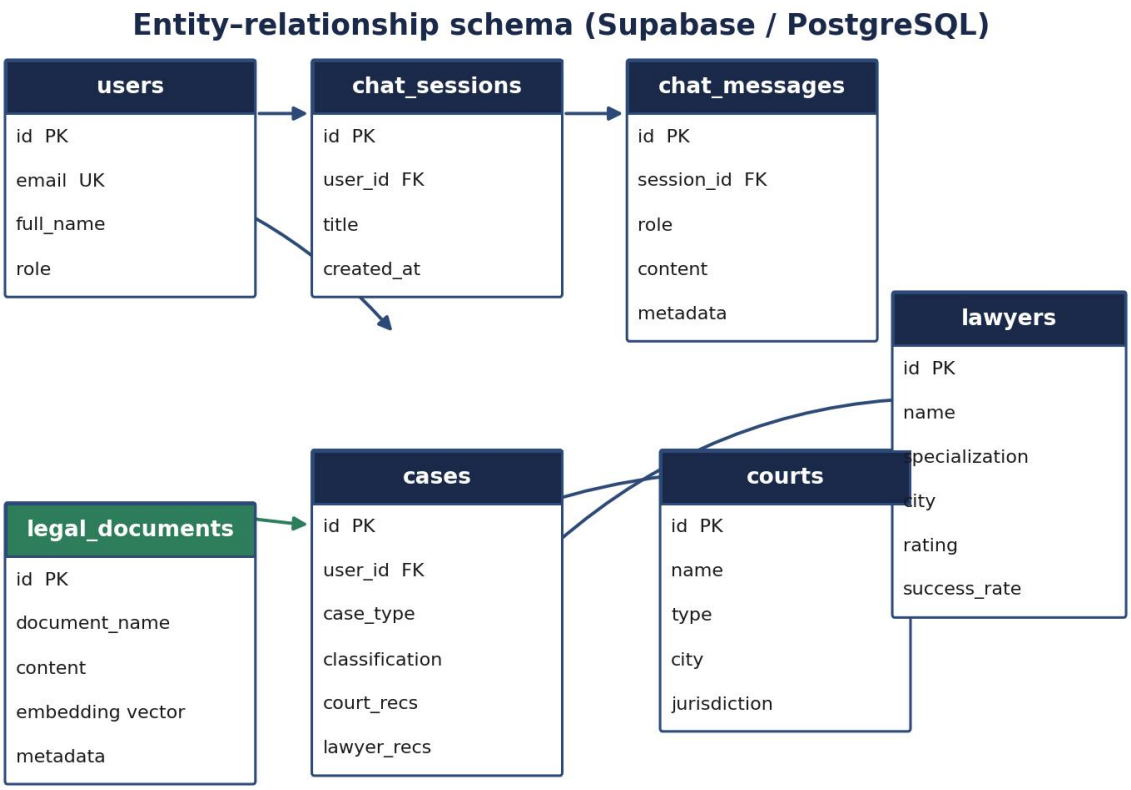

Figure 4. Entity–relationship schema. Users own chat sessions and cases; cases carry classification results and recommendation payloads; the legal_documents table holds the embedded statutory corpus that drives retrieval.

Table 4. Principal database entities.

Entity	Key fields	Purpose
users	id, email, role	Account identity and access role (citizen / lawyer)
chat_sessions	id, user_id, title	Persistent consultation sessions
chat_messages	id, session_id, role, content	Turn-level message history with metadata
cases	id, user_id, case_type, classification	Classified problem with court and lawyer recommendations
legal_documents	id, content, embedding	Statutory corpus with vector embeddings for retrieval

courts	id, name, type, jurisdiction	Reference data for jurisdiction-aware recommendation
lawyers	id, specialization, city, rating	Professional profiles for ranked recommendation

The retrieval-augmented consultation pipeline

The core of the system is the read path that turns a citizen's natural-language question into a grounded answer (Figure 5). The query is first embedded with Gemini's embedding model. The resulting vector is matched against the embedded statutory corpus in pgvector by cosine similarity, returning the top-k most relevant passages. These passages are assembled into a grounded context and passed, together with the original question, to Gemini for generation. Because the model is instructed to answer from the supplied context and to cite the provisions it uses, its scope to invent is sharply curtailed and the answer carries verifiable sources. The statutory corpus is chunked and embedded once, offline; only the lightweight query-time path runs per request.

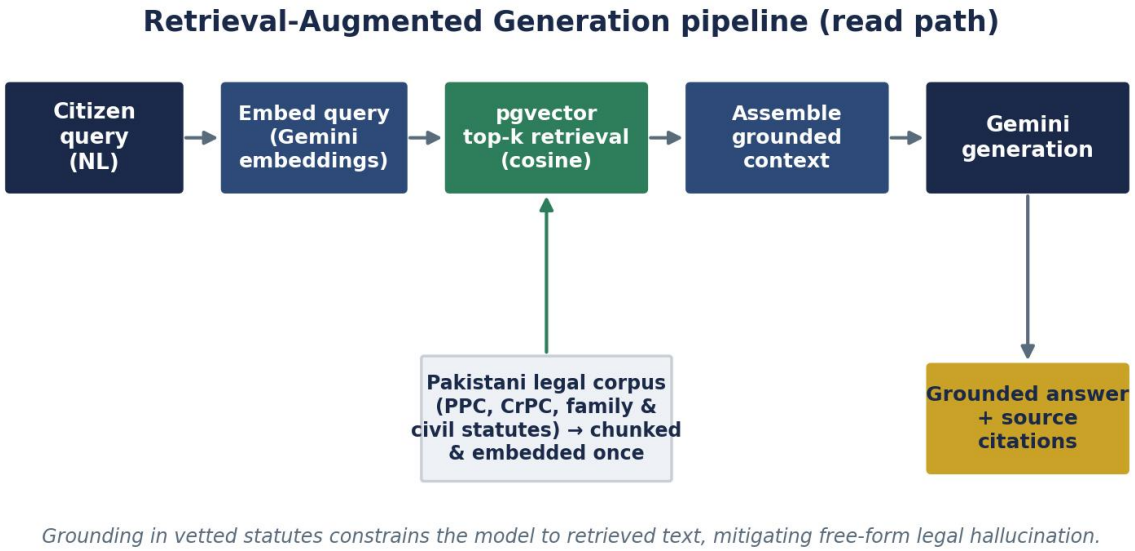

Figure 5. Retrieval-augmented generation read path. Grounding the generator in retrieved statutory text rather than parametric memory is the system's primary defence against legal hallucination (Lewis et al., 2020; Dahl et al., 2024).

Figure 6 traces the same interaction as a sequence across the system's components,

from classification through retrieval to grounded generation and the return of an answer with citations.

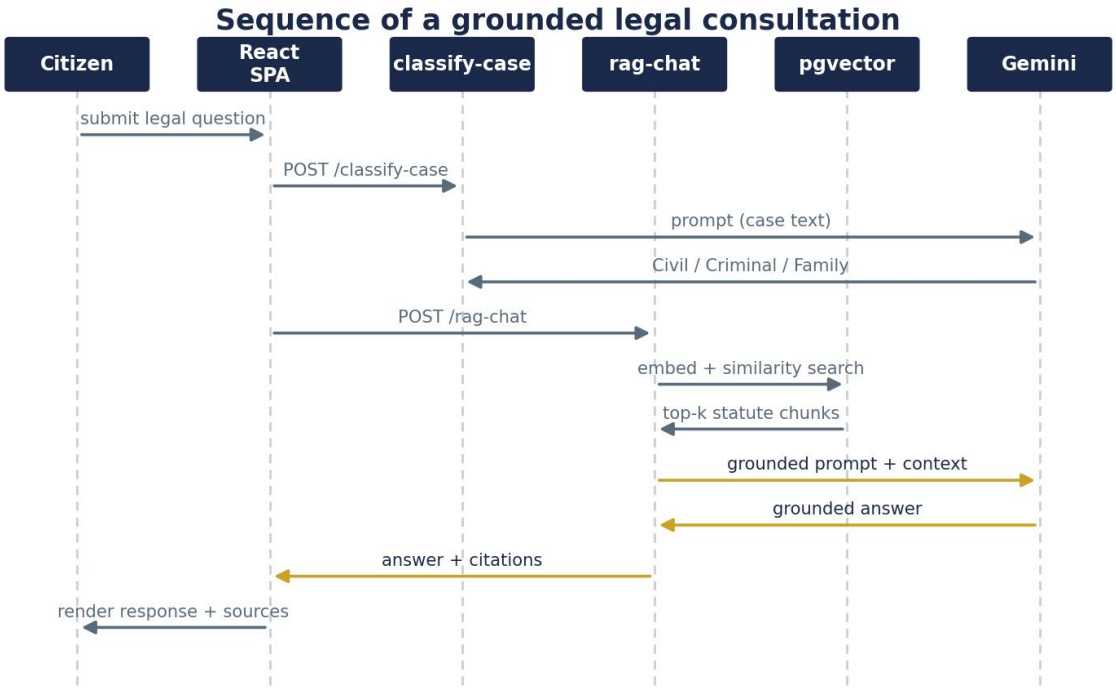

Figure 6. Sequence of a grounded consultation. Classification routes the problem; the rag-chat function performs retrieval over pgvector and conditions Gemini on the retrieved statutes before returning a cited answer.

Case classification

Before retrieval, the system classifies the described problem into one of three in-scope domains Civil, Criminal, or Family law (Figure 7) which focuses retrieval on the relevant statutes and drives downstream court and lawyer recommendation. Classification is performed by the classify-case edge function, which prompts the language model with the problem description and a constrained label set. Restricting the taxonomy to three well-understood domains keeps classification tractable and interpretable for a first-line guidance tool; broader coverage is identified in Section 7 as future work.

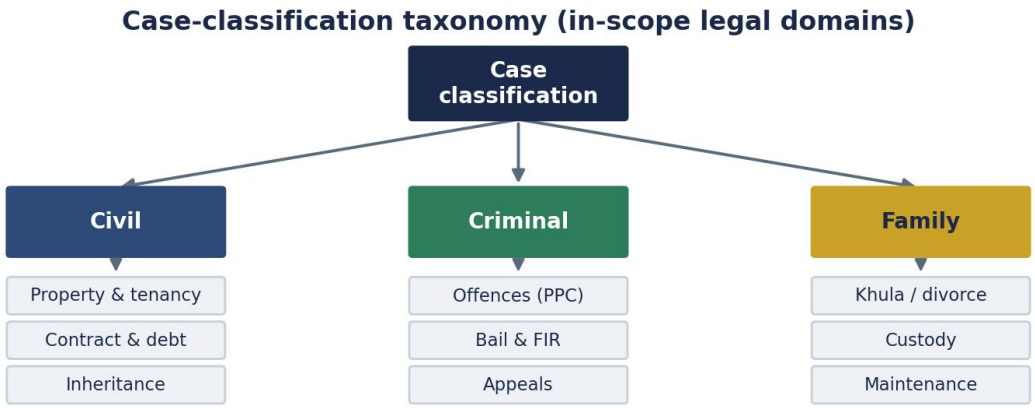

Figure 7. Case-classification taxonomy. The three in-scope domains and representative sub-issues that the classifier routes to the appropriate statutory context and recommendation logic.

Court and lawyer recommendation

Once a case is classified, the recommend-court function selects courts whose jurisdiction matches the case type and whose location is in or near the user's city, returning each with its name, address, and jurisdictional scope. The recommend-lawyer function selects advocates whose specialization matches the case type and who practice in the user's city, ranking candidates by a composite of experience, rating, and historical case volume drawn from the lawyers table. Both functions return structured results that the interface renders as actionable next steps, closing the loop between an abstract legal answer and the concrete avenues for pursuing it.

Security and data protection

Because consultations may contain sensitive personal circumstances, the design treats data protection as a first-class requirement. Authentication uses JSON Web Tokens; transport is encrypted with TLS; and PostgreSQL row-level security restricts every user to their own sessions, messages, and cases. Sessions and case records persist so that a citizen can return to a consultation, while the row-level policies ensure that persistence never becomes exposure. These controls are revisited in Section 6 as part of the responsible-AI discussion.

The guided consultation workflow

From the citizen's perspective, a consultation is a four-stage guided flow (Figure 8): describe the case, receive court

recommendations, receive lawyer recommendations, and obtain a structured summary with next steps. The staged design lowers the cognitive load on users with limited legal or digital literacy by asking for one thing at a time and building toward a complete, actionable overview rather than a single undifferentiated answer.

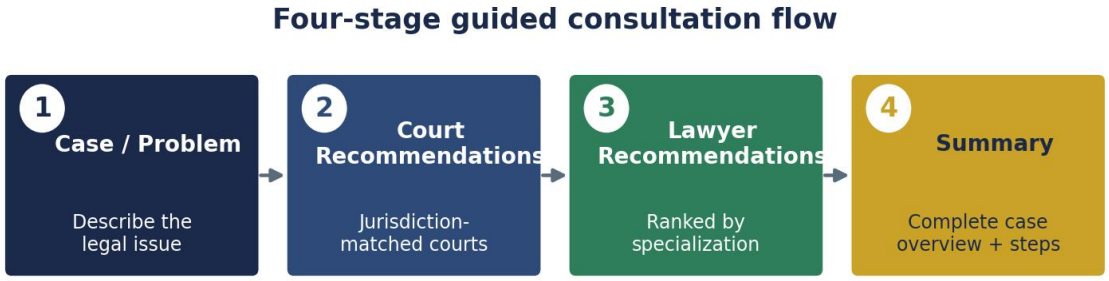

Figure 8. The four-stage guided consultation flow presented to the citizen, mirroring the staged progress indicator in the live interface.

Implementation

This section records how the design was realized, with emphasis on the choices that affect grounding and reliability.

Corpus ingestion and embedding

The statutory corpus is prepared offline. Source statutes are segmented into passage-level chunks, each chunk is embedded with Gemini's embedding model, and the resulting vectors are written to the legal_documents table alongside the source text and metadata. pgvector indexes the embedding column so that query-time similarity search runs efficiently. Because ingestion is decoupled from serving, the corpus can be expanded or updated for example, when a statute is amended by re-running ingestion without touching the application code, a property that directly addresses the temporal-staleness failure mode documented for static legal models.

Edge functions

Application logic is implemented as Supabase edge functions running on Deno. The classify-case function maps a problem description to a domain label; the rag-chat function performs query embedding, pgvector retrieval, context assembly, and grounded generation; and the recommend-court and recommend-lawyer functions execute the jurisdiction- and specialization-matching logic described in Section 3.6. Keeping each function small and single-purpose makes the system easy to test and to reason about, and isolates the external model calls behind well-defined boundaries.

Prompt design and grounding

Grounding is enforced at the prompt level. The generation prompt instructs the model to answer using only the retrieved statutory context, to cite the provisions it relies on, and to indicate when the supplied context is insufficient rather than to supplement it from memory. This instruction is the operational expression of the paper's central commitment: the retrieved corpus, not the model's parameters, is the authority. The same discipline underlies the decision to surface source citations to the user, so that any answer can be checked against the provision it claims to rest on.

Frontend and deployment

The interface is a React single-page application built with Vite and styled for mobile-responsive use, since many citizens will reach the service from a phone. Routing and the staged consultation progress indicator are implemented as UI components that call the edge functions

over authenticated requests. The frontend is deployed on Vercel and the backend on Supabase, an arrangement that requires no server administration and scales with demand.

Evaluation

Evaluation proceeded in two registers: functional acceptance testing of the system's features and non-functional measurement against pre-set targets. We then state precisely what these results establish, and specify the answer-quality protocol that should follow.

Testbed

The system was exercised on its deployed cloud infrastructure: a React frontend on Vercel, Supabase edge functions and PostgreSQL 15 with pgvector, and Gemini reached through OpenRouter. Client testing spanned current versions of Chrome, Firefox, Edge, and Safari and devices from desktop to smartphone form factors. Functional behaviour was validated with scripted test cases; non-functional behaviour was measured with browser developer tools and Supabase query logs.

Functional testing

The functional suite covers the system's full feature surface. All eight defined test cases passed (Table 5, Figure 9), confirming that registration, authentication, session management, grounded chat, classification, recommendation, and logout behave as specified.

Table 5. Functional test results.

Test	Verifies	Expected result	Status
TC_001	New-user signup with email verification	Account created, JWT issued	Pass
TC_002	Lawyer signup with specialization details	Record added to lawyers table	Pass
TC_003	Citizen login and access to chat	Authenticated, redirected to chat	Pass
TC_004	Lawyer login and profile access	Authenticated, profile shown	Pass
TC_005	New chat-session creation	Session created and persisted	Pass
TC_006	RAG endpoint processes query	Grounded response with	Pass

		metadata	
TC_007	Chat-session deletion	Session removed from UI and flagged	Pass
TC_008	Logout and session termination	Session cleared, redirected to login	Pass

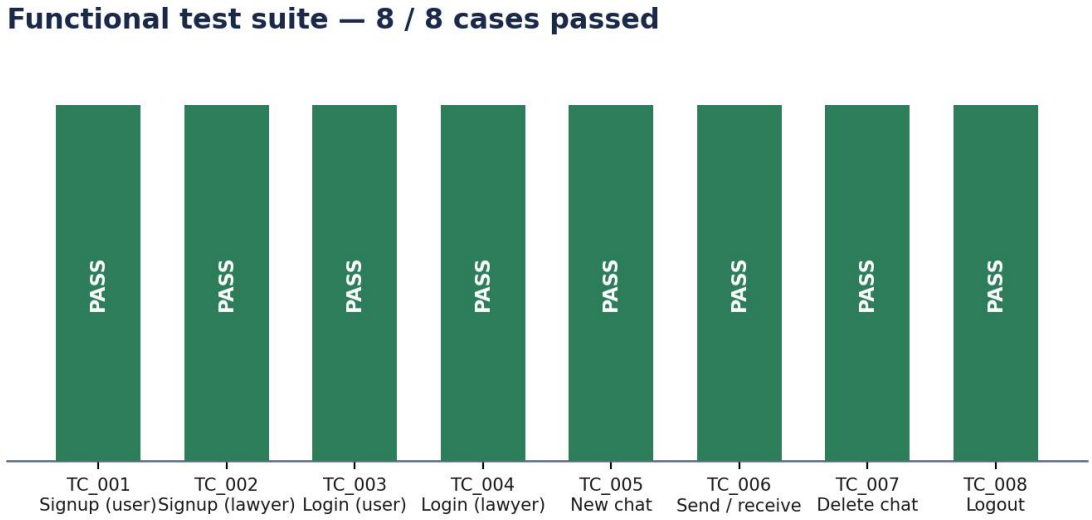

Figure 9. Functional acceptance summary: all eight test cases passed, covering authentication, session management, grounded chat, and recommendation.

Non-functional results

Non-functional measurements met or exceeded every acceptance target (Table 6, Figure 10). Mean response time for a grounded answer was 3.2 seconds against a four-second budget; court and lawyer recommendations returned in 420 milliseconds against a 500-millisecond target; case-classification accuracy on the project test set was 92% against a 90% threshold; measured availability was 99.6%; and the system sustained 55 concurrent users against a target of 50.

Table 6. Non-functional results against acceptance targets.

Metric	Target	Observed	Outcome
AI response time	≤ 4 s	3.2 s	Met
Court/lawyer recommendation latency	≤ 500 ms	420 ms	Met

Case-classification accuracy	≥ 90%	92%	Met
System availability	99.5%	99.6%	Met
Concurrent users	50	55	Met
Browser compatibility	Latest 2 versions	Fully compatible	Met

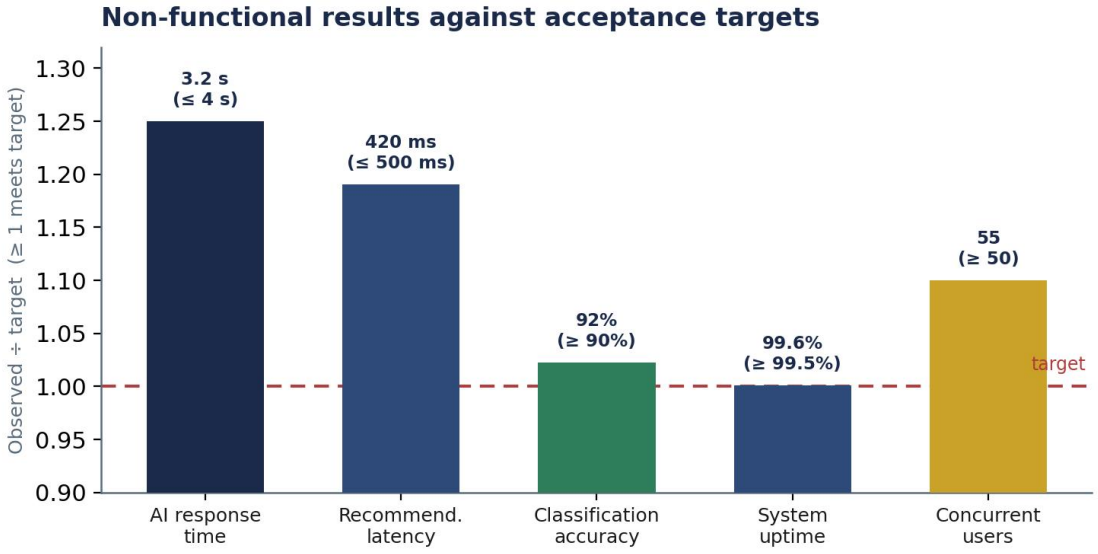

Figure 10. Observed non-functional results expressed as a ratio to their acceptance targets; the dashed line marks the target. Every metric meets or exceeds its threshold.

What these results establish and what they do not

It is important to state the scope of these findings precisely. The results above are acceptance results: they establish that the system functions as specified and that it is fast, available, and scalable enough for its intended use. The 92% figure refers to case-classification accuracy on the project's own test set and should not be read as a measure of the legal correctness of generated answers. Acceptance testing of this kind does not, by itself, establish answer faithfulness, retrieval quality, or robustness against adversarial or out-of-distribution questions. We regard the rigorous measurement of answer quality as the necessary next stage rather than a settled outcome, and Section 5.5 specifies how it should be conducted.

Proposed answer-quality protocol (RAGAS)

Because the system is a RAG pipeline, its answer quality is best assessed with metrics designed for that architecture. We propose adopting RAGAS, a reference-free

evaluation framework that scores a RAG system along four dimensions without requiring large hand-labelled datasets (Es et al., 2024). Two dimensions assess retrieval context precision (the proportion of retrieved passages that are relevant) and context recall (whether all necessary statutes were retrieved) and two assess generation faithfulness (whether the answer is supported by the retrieved context) and answer relevancy (whether it addresses the question). Their mean forms a single RAGAS score. Figure 11 maps the four metrics onto the retrieval and generation stages, and Table 7 specifies the protocol.

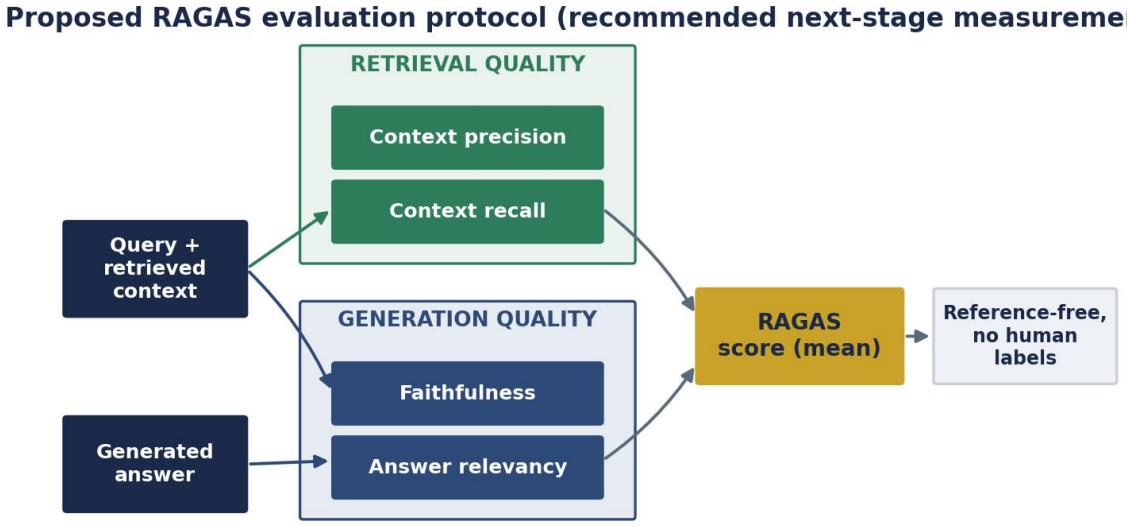

Figure 11. Proposed RAGAS evaluation protocol. Context precision and recall measure retrieval; faithfulness and answer relevancy measure generation. The framework is reference-free, which suits a setting without a large annotated Pakistani legal-QA corpus.

Table 7. Proposed RAGAS protocol for next-stage answer-quality evaluation.

Metric	Stage	What it measures
Context precision	Retrieval	Share of retrieved statute passages relevant to the question
Context recall	Retrieval	Whether all statutes needed to answer were retrieved
Faithfulness	Generation	Whether each claim in the answer is supported by retrieved text

Answer relevancy	Generation	Whether the answer addresses the citizen's actual question

Operationalizing this protocol requires a modest evaluation set of Pakistani legal questions with expert-verified answers and citations itself a worthwhile community resource. Until that measurement is performed, claims about the legal accuracy of generated answers should remain provisional.

6 Discussion

Strengths

The system's principal strength is architectural honesty about the reliability problem. By grounding generation in retrieved statutes and attributing answers to sources, it addresses head-on the failure mode that the legal-AI literature identifies as disqualifying for unguarded models (Dahl et al., 2024; Magesh et al., 2025). Its second strength is integration: classification, grounded consultation, and recommendation are unified in a single citizen-facing flow, which is what distinguishes practical access-to-justice tooling from a bare chatbot. Third, the serverless design makes the system inexpensive to operate and straightforward to scale, and the decoupled corpus makes legal updates a data operation rather than a code change.

6.2 Limitations

Several limitations bound the current contribution, stated plainly so they are not mistaken for settled strengths:

- **Answer quality is not yet benchmarked.** As Section 5.4 makes clear, the evaluation is acceptance-level; the RAGAS protocol of Section 5.5 has not yet been run, so the legal faithfulness of generated answers remains unmeasured.
- **Geographic and linguistic scope.** Court and lawyer coverage is presently limited to Sindh, and the interface is English-only, which excludes many of the citizens who most need first-line guidance.
- **Domain scope.** Only Civil, Criminal, and Family law are in scope; other domains are not yet covered.
- **External dependency.** Reliance on an external model accessed through OpenRouter introduces latency and availability risk, and ties answer behaviour to a third-party model the team does not control.

6.3 Threats to validity

The evaluation faces the threats to validity normal for a system of this kind, summarized with their mitigations in Table 8. The most consequential is construct validity: acceptance metrics measure that the system works, not that its legal answers are correct, and conflating the two would overstate the contribution. The corpus's completeness and currency is a second threat, since retrieval can only ground answers in statutes that are present and up to date. The classification accuracy figure depends on the composition of the project test set, whose representativeness is not independently established.

Table 8. Threats to validity and mitigations.

Threat	Risk	Mitigation
Construct validity	Acceptance metrics misread as answer-quality evidence	Explicit scoping (§5.4); RAGAS protocol proposed (§5.5)
Corpus completeness	Missing or outdated statutes degrade grounding	Decoupled ingestion enables routine corpus updates
Test-set representativeness	92% accuracy may not generalize	Stated as project-set result; broader evaluation set proposed
Model dependency	Third-party model changes affect behaviour	Grounding + citations allow human verification of outputs

Ethical and responsible-AI considerations

A tool that speaks about the law to vulnerable users carries obligations beyond functioning correctly. Four considerations are central. First, the boundary between legal information and legal advice must be explicit: the system is designed to improve legal literacy and orientation, not to replace a qualified advocate, and its outputs should be framed and received as first-line guidance. Second, reliability must be communicated, not assumed; grounding and source citation exist precisely so that users and reviewing professionals can verify answers, and the system should fail toward caution by signalling uncertainty rather than fabricating when retrieval is thin (Dahl et al., 2024). Third, fairness and representativeness of the corpus matter, since gaps or skew in the statutes and reference data propagate directly into guidance. Fourth, data protection is an

ethical as well as a technical requirement: consultations can be sensitive, and the JWT authentication, TLS transport, and row-level security of Section 3.7 are the concrete safeguards. These commitments situate the system within the responsible-AI governance frame that should accompany any deployment of AI in the justice domain.

7 Conclusion and Future Work

The AI Justice Assistant is a localized, retrieval-augmented system that grounds first-line legal guidance for Pakistani citizens in vetted statutory text, integrating case classification, source-attributed consultation, and jurisdiction-aware court and lawyer recommendation in a single serverless platform. Its design responds directly to the central finding of the legal-AI literature that unguarded language models hallucinate at rates incompatible with legal use by constraining generation to retrieved sources and making every answer verifiable. Acceptance testing confirms that the system functions as specified and meets its performance, availability, and scalability targets, and the paper is explicit that rigorous measurement of answer quality, via the proposed RAGAS protocol, is the necessary next step. Future work follows the limitations of Section 6. The immediate priority is to construct an expert-verified Pakistani legal-QA evaluation set and to run the RAGAS protocol, turning provisional claims about answer quality into measured ones. Beyond that, the roadmap is to extend court and lawyer coverage from Sindh toward nationwide reach; to add Urdu and other regional-language support so the tool serves the citizens who most need it; to widen the domain taxonomy beyond Civil, Criminal, and Family law; and to strengthen retrieval with re-ranking and hybrid sparse–dense search. Each of these moves the system further toward its purpose: making reliable legal orientation an ordinary, accessible good rather than a privilege.

References

.jmi, A. (2024). *Revolutionizing access to justice: The role of AI-powered chatbots and retrieval-augmented generation in legal self-help.* Working paper.

.riai, F., Mackenzie, J., & Demartini, G. (2024). *Natural language processing for the legal domain: A survey of tasks, datasets, models,*

and challenges (arXiv:2410.21306). arXiv. https://doi.org/10.48550/arXiv.2410.21306

Beauchemin, D., Gagnon, Z., & Khoury, R. (2024). *Quebec automobile insurance question-answering with retrieval-augmented generation* (arXiv:2410.09623). arXiv. https://doi.org/10.48550/arXiv.2410.09623

Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., & Androutsopoulos, I. (2020). LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (pp. 2898–2904). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.261

Dahl, M., Magesh, V., Suzgun, M., & Ho, D. E. (2024). Large legal fictions: Profiling legal hallucinations in large language models. *Journal of Legal Analysis, 16*(1), 64–93. https://doi.org/10.1093/jla/laae003

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019* (pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

s, S., James, J., Espinosa-Anke, L., & Schockaert, S. (2024). RAGAS: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations* (pp. 150–158). Association for Computational Linguistics. https://aclanthology.org/2024.eacl-demo.16/

an, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., & Li, Q. (2024). A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 6491–6501). ACM.

ei, Z., Shen, X., Zhu, D., Zhou, F., Han, Z., Zhang, S., Chen, K., Shen, Z., & Ge, J. (2024). LawBench: Benchmarking legal knowledge of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Gemini Team, Google. (2024). *Gemini: A family of highly capable multimodal models* (arXiv:2312.11805). arXiv. https://doi.org/10.48550/arXiv.2312.11805

Guha, N., Nyarko, J., Ho, D. E., Ré, C., Chilton, A., Narayana, A., Chohlas-Wood, A., Peters, A., Waldon, B., Rockmore, D., Zambrano, D., Talisman, D., Hoque, E., Surani, F., Fagan, F., … Li, Z. (2023). LegalBench: A collaboratively built benchmark for measuring legal reasoning in large language models. In *Advances in Neural Information Processing Systems 36* (Datasets and Benchmarks Track).

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., & Liu, T. (2025). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems, 43*(2), 1–55.

Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W. (2020). Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (pp. 6769–6781). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.550

Law and Justice Commission of Pakistan. (2024). *Bi-annual report of judicial statistics (July–December 2023).* Government of Pakistan.

ewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33* (pp. 9459–9474).

i, J., Bhambhoria, R. V., Dahan, S., & Zhu, X. (2024). *Experimenting with legal AI solutions: The case of question-answering for access to justice* (arXiv:2409.07713). arXiv. https://doi.org/10.48550/arXiv.2409.07713

lagesh, V., Surani, F., Dahl, M., Suzgun, M., Manning, C. D., & Ho, D. E. (2025). Hallucination-free? Assessing the reliability of leading AI legal research tools. *Journal of Empirical Legal Studies, 22*(2), 216–242.

anchal, D., Gole, A., Narute, V., & Joshi, R. (2025). *LawPal: A retrieval augmented generation based system for enhanced legal accessibility in India* (arXiv:2502.16573). arXiv. https://doi.org/10.48550/arXiv.2502.16573

eimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using

Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* (pp. 3982–3992). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1410

Shu, D., Zhao, H., Liu, X., Demeter, D., Du, M., & Zhang, Y. (2024). LawLLM: Law large language model for the US legal system. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management* (pp. 4882–4889). ACM.

Sommerville, I. (2015). *Software engineering* (10th ed.). Pearson.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30* (pp. 5998–6008).

Wiratunga, N., Abeyratne, R., Jayawardena, L., Martin, K., Massie, S., Nkisi-Orji, I., Weerasinghe, R., Liret, A., & Fleisch, B. (2024). CBR-RAG: Case-based reasoning for retrieval augmented generation in LLMs for legal question answering. In *Case-Based Reasoning Research and Development (ICCBR 2024)* (pp. 445–460). Springer.

World Justice Project. (2023). *Rule of law index 2023*. World Justice Project. https://worldjusticeproject.org/rule-of-law-index

Appendix A. Figures and Reproduction Notes

This appendix lists the figures used in the manuscript and records how each was produced, so that the visual material can be regenerated and audited. The figures fall into two kinds, and the distinction is deliberate.

A.1 System schematics

The architecture, retrieval pipeline, entity-relationship schema, use-case, sequence, consultation-workflow, classification-taxonomy, and proposed-evaluation diagrams (Figures 3–8, 11, and the taxonomy) are schematic drawings of the implemented system. They describe components, data flows, and relationships; they contain no measured data and assert no empirical results.

- Figure 3 Three-tier serverless architecture.
- Figure 4 Entity-relationship schema (users, sessions, messages, cases, legal_documents, courts, lawyers).
- Figure 5 Retrieval-augmented generation read path.

- Figure 6 Sequence of a grounded consultation.
- Figure 7 Case-classification taxonomy.
- Figure 8 Four-stage guided consultation flow.
- Figure 11 Proposed RAGAS evaluation protocol (recommended, not yet executed).

A.2 Charts of reported values

Three figures chart values reported by the project. Each plots only the numbers stated in the corresponding table; none is simulated or inferred.

- Figure 1 Pending-case distribution by court tier, from the Law and Justice Commission of Pakistan (2024): ≈ 1.86 M district, ≈ 0.30 M high courts, ≈ 0.056 M Supreme Court.
- Figure 9 Functional acceptance summary: the eight test cases of Table 5, all passing.
- Figure 10 Non-functional results of Table 6, plotted as the ratio of each observed value to its acceptance target.

A.3 Reproduction and integrity notes

The schematics were generated programmatically with a fixed navy-and-gold palette and no gridlines; they can be regenerated from the figure script that accompanies the project. The three charts are driven directly by the values in Tables 5 and 6 and by the cited Law and Justice Commission figures, so they update automatically if those inputs change. No confusion matrices, latency distributions, retrieval scores, or per-class metrics are presented, because the project did not measure them; the proposed RAGAS protocol of Section 5.5 is the route by which those quantities should be obtained. This separation schematics and reported values only is intended to keep the manuscript's visual claims fully traceable to either the system design or the project's own measurements.

A.4 Availability

The system was deployed with the React frontend on Vercel and the backend on Supabase edge functions with PostgreSQL and pgvector. Project artefacts the requirements and design specifications, the test plan, and interface captures are maintained in the project repository and supplementary dossier referenced in the original Final Year Project documentation.