International Journal of Engineering Research and Development
e- ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 22, Issue 8 (August 2026), PP 61-65

Privacy-Preserving Edge-RAG for Intelligent Web Applications: A Client-Side Context Governance Framework

Venkata Krishna Teja Kolli
Independent Researcher, United States
Corresponding Author: Venkata Krishna Teja Kolli

ABSTRACT
Intelligent web applications increasingly use retrieval-augmented generation to personalize answers, summarize user data and guide actions inside browser-based workflows. This design improves usefulness but creates privacy and governance concerns because session state, user preferences, documents and interaction history can be copied into remote model contexts without adequate minimization. This paper proposes a Privacy-Preserving Edge-RAG framework for intelligent web applications. The framework places a context classifier, local vector cache, policy-governed context budget and answer/action gate between the user interface and any model call. Private context is processed locally where feasible, while cloud retrieval and remote inference receive only redacted, summarized or consent-approved context. The paper contributes an architecture, an operational workflow and an evaluation protocol for measuring answer quality together with privacy exposure, latency, consent burden and auditability. The proposed approach shows how intelligent web applications can deliver adaptive behavior without treating all user context as freely reusable model input.

Date of Submission: 05-08-2026 Date of acceptance: 17-08-2026

I. INTRODUCTION

Modern web applications are becoming intelligent interfaces that learn from user behavior, retrieve task-specific information and generate personalized responses. Examples include AI help centers, learning portals, dashboards, developer tools, shopping assistants and enterprise intranets. These systems often depend on retrieval-augmented generation because application knowledge changes rapidly and because the user expects answers grounded in current documents, account data and page context.

The same context that makes an intelligent web application useful can also make it risky. Browser sessions contain search history, form values, account metadata, uploaded files, location signals, preferences and behavioral traces. When all of this context is sent to a remote model or retrieval service, the application may violate data minimization principles even if the final answer appears correct. Long context windows also create engineering problems: irrelevant context can distract the model, increase latency and raise cost without improving answer quality [3].

This paper addresses the following research question: how can intelligent web applications use retrieval and personalization while minimizing unnecessary exposure of user context? The proposed answer is a Privacy-Preserving Edge-RAG framework. Edge-RAG is defined here as a retrieval and generation architecture in which sensitive context is classified, filtered and processed at the browser or edge layer before any remote inference or cloud retrieval is performed.

The contribution of this paper is threefold. First, it proposes a client-side context governance architecture for intelligent web applications. Second, it defines an operational workflow for routing context among local inference, local retrieval, public retrieval and consent-approved cloud escalation. Third, it presents an evaluation protocol that measures not only answer quality but also privacy exposure, consent burden, latency and auditability. The paper is written as a design-science contribution and does not claim experimental deployment results.

II. RELATED WORK

Retrieval-augmented generation combines language models with external evidence and has become an important pattern for knowledge-intensive applications [1]. Recent surveys describe RAG as a flexible architecture involving indexing, retrieval, augmentation and generation [2]. For web applications, RAG is attractive because product documentation, user records and public information can be updated independently of model weights. However, retrieval can also pull untrusted or private data into the model context, which turns context selection into a governance problem.

Long-context research further shows that simply providing more text is not always beneficial. Liu et al. reported that models can struggle to use relevant information when it appears in the middle of long contexts [3]. This finding matters for web applications because page state, chat history and retrieved passages are often concatenated into large prompts. A context governance layer can improve both privacy and accuracy by limiting the prompt to information that is necessary, trusted and relevant.

Security literature has shown that LLM-integrated applications can be compromised through indirect prompt injection when malicious instructions are embedded in retrieved documents or web pages [4]. OWASP identifies prompt injection, sensitive information disclosure, excessive agency and overreliance as major LLM application risks [10]. NIST AI RMF and the Generative AI Profile also frame AI risk management as an ongoing lifecycle activity that includes mapping, measuring and managing system-specific risks [11], [12].

Edge and client-side inference are becoming more practical because browsers can use WebGPU and WebAssembly-backed execution paths [5], [6], [14], [15]. These technologies do not remove the need for governance, but they create an architectural option: sensitive context can be summarized, embedded, classified or redacted locally before remote services are used. Federated learning, secure aggregation and differential privacy provide additional background for privacy-preserving computation over distributed data [7], [8], [9].

III. PROPOSED EDGE-RAG FRAMEWORK

The proposed framework separates personalization from unrestricted context sharing. The web interface collects user intent and page state, but the raw state is not automatically forwarded to a model. Instead, a context classifier assigns each context item a sensitivity label, trust label, freshness label and relevance score. The labels determine whether the item may be used locally, summarized, redacted, sent to a cloud retrieval service or excluded entirely.

A. Architectural Components

The first component is the user session layer, which observes the user's request, current page, selected text and relevant application state. The second component is the context classifier. It identifies personal data, credentials, financial data, health-related text, organizational secrets and other sensitive categories. It also distinguishes first-party application data from third-party or untrusted web content.

The third component is the local index and cache. It stores approved embeddings, recent task memory and user preferences on the client or near-edge environment. This allows the application to personalize low-risk responses without repeatedly exposing private data to remote services. The fourth component is the policy-governed context budget. This budget limits the amount and type of context that can enter a prompt or retrieval request. The fifth component is the public or cloud retrieval layer, which is used for non-sensitive knowledge or for user-approved escalation. The final component is the answer and action gate, which checks citations, redaction, consent status and audit metadata before the result is displayed or an action is executed.

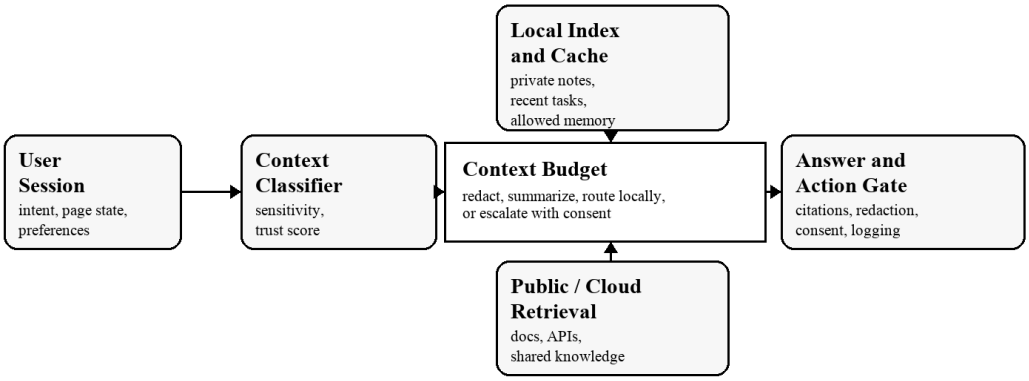

Figure 1: Edge-RAG context governance flow for intelligent web applications

B. Context Labels

Each context item is represented as a tuple containing content, source, sensitivity, trust, relevance, freshness and permitted destination. Sensitivity describes the privacy risk of exposing the item. Trust describes whether the item came from the user, the application, an approved document or an untrusted external source. Relevance estimates whether the item is needed for the task. Permitted destination determines whether the item may stay local, be summarized, be redacted, be sent remotely with consent or be blocked.

Table 1: Context classes and permitted handling rules

Context Class	Examples	Default Handling
Public knowledge	Documentation, public web pages, published policies.	May be retrieved remotely; cite source and scan for prompt injection.
First-party app data	Ticket status, product settings, non-sensitive account metadata.	Use if relevant; summarize before remote use when possible.
Private user content	Uploaded files, notes, drafts, messages and form entries.	Prefer local processing; require consent for remote inference.
High-risk sensitive data	Credentials, payment data, health data and regulated identifiers.	Block by default; redact or use specialized secure workflow.
Untrusted instructions	Hidden page text, external comments and retrieved adversarial content.	Treat as quoted data, not as model instruction.

IV. OPERATIONAL WORKFLOW

The workflow begins when the user issues a request. The application decomposes the request into intent, required evidence and potential actions. The context classifier then scans available session context and assigns labels. Items that are irrelevant or too sensitive are excluded immediately. Items that are useful but private are routed to local summarization or local embedding. Items that are public or low risk may be passed to a remote retrieval service.

The policy-governed context budget is applied before generation. A budget can be expressed in tokens, sensitivity points or allowed data classes. For example, a shopping assistant may use public product details and user-selected filters but should not include unrelated browsing history. An enterprise support assistant may use a ticket summary and approved knowledge-base passages but should avoid copying entire customer records into a prompt. The budget therefore functions as a privacy and relevance constraint.

If the system cannot answer with local or low-risk context, it asks for consent to escalate. Consent should be specific, explaining which data category will be sent and why. After remote inference or retrieval, the answer and action gate validates citations, checks that blocked context was not included and records an audit event. For action-taking tasks, the gate requires explicit approval before submitting forms, sending messages or changing application state.

V. EVALUATION PROTOCOL

A useful evaluation protocol must measure more than final answer correctness. The proposed protocol evaluates five dimensions: answer quality, privacy exposure, latency, consent burden and auditability. Answer quality measures correctness, completeness and citation coverage. Privacy exposure measures how much sensitive context leaves the client or trusted edge boundary. Latency measures the time cost of local classification, retrieval and generation. Consent burden measures the number and clarity of user approvals required. Auditability measures whether a reviewer can reconstruct what context was used and why.

The evaluation can be conducted using scripted web tasks across representative domains such as help centers, personal dashboards, document portals and shopping assistants. Each task should have a gold-standard answer, a list of context items that are necessary and a list of context items that are sensitive but unnecessary. The framework can then be compared against a baseline that sends all available context to a remote model and another baseline that uses retrieval without sensitivity labels.

Table 2: Evaluation dimensions for privacy-preserving Edge-RAG

Dimension	Metric	Evaluation Question
Answer quality	Correctness, completeness and citation coverage.	Did the system answer the task with adequate evidence?
Privacy exposure	Sensitive tokens or fields sent outside the local boundary.	Did the system minimize remote exposure of private context?
Latency	End-to-end response time and local processing overhead.	Is the privacy-preserving path usable in a web interface?
Consent burden	Number of prompts and user comprehension of data use.	Does the user understand what is being shared and why?
Auditability	Completeness of context labels, routing decisions and model-call logs.	Can the application explain which context influenced the answer?

VI. ILLUSTRATIVE APPLICATION SCENARIOS

The first scenario is an intelligent learning portal. A student asks for help understanding a concept while the page contains assignment text, previous quiz mistakes and private notes. The framework allows the system to use the selected assignment prompt and relevant local notes, but it prevents unrelated learning history from being sent to a cloud model. The answer can be personalized while still preserving data minimization.

The second scenario is a customer support dashboard. An agent asks the assistant to draft a reply using a ticket and product documentation. The system summarizes the ticket locally, removes payment details and retrieves public troubleshooting steps. The generated reply includes citations to approved documents and does not expose the complete customer record. If the agent wants the system to send the message, the action gate requests confirmation.

The third scenario is a personal finance web application. A user asks why expenses increased this month. Raw transactions are sensitive, so local aggregation computes category totals before any model receives context. The remote model may receive only category summaries, not merchant-level details, unless the user explicitly approves a deeper analysis. This example shows how local computation can preserve utility while reducing sensitive disclosure.

The fourth scenario is a research assistant that reads public pages and private drafts. Public web pages can be retrieved remotely, but private drafts remain local. The system quotes retrieved external content as data and ignores hidden instructions inside pages. The answer separates public evidence from private notes and records which sources were used. This reduces the risk of indirect prompt injection and improves reviewability.

VII. DISCUSSION

The proposed Edge-RAG framework makes privacy a first-class architectural property. Rather than asking whether a model provider is trusted in general, the framework asks which specific context item is needed for the present task and where it is allowed to travel. This aligns with data minimization and privacy-by-design principles while preserving the benefits of retrieval and personalization.

There are trade-offs. Local inference and local indexing may increase client resource use. Some devices may not support efficient browser-side models, requiring fallback to cloud services. Frequent consent prompts may interrupt user flow if they are poorly designed. Strict context budgets may reduce answer completeness for complex tasks. These trade-offs should be measured rather than hidden inside implementation choices.

The framework is also complementary to the trust-aware orchestration approach used in agentic web applications. Trust-aware orchestration focuses on tool use, evidence and action control. Edge-RAG focuses on context governance before retrieval and generation. Together, they suggest that future intelligent web applications should separate four concerns: what the user wants, what context is necessary, what tools may be used and what evidence supports the final answer.

The limitation of this paper is that it proposes a reference framework and evaluation method rather than reporting deployment results. Future work should implement the framework in a real web application and compare it against common RAG baselines. A useful study would measure answer quality, privacy exposure and latency across different device classes and model sizes. Additional work is also needed on standard formats for context labels and portable consent receipts.

VIII. CONCLUSION

Intelligent web applications need user context to be helpful, but unrestricted context sharing creates privacy, security and reliability risks. This paper proposed a Privacy-Preserving Edge-RAG framework that classifies context, applies a policy-governed context budget, prefers local processing for sensitive information and escalates to remote services only when necessary or consented. The framework gives developers a practical structure for building AI-enabled web applications that personalize without treating all user data as model input.

The central conclusion is that privacy-preserving intelligence is an architectural design problem. Retrieval, generation, local inference and consent must be composed deliberately. By evaluating answer quality alongside privacy exposure, latency, consent burden and auditability, future web applications can become more adaptive while remaining accountable to the users whose context makes that intelligence possible.

REFERENCES

[1] P. Lewis et al., Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, Advances in Neural Information Processing Systems, 2020.
[2] Y. Gao et al., Retrieval-Augmented Generation for Large Language Models: A Survey, arXiv:2312.10997, 2023.
[3] N. F. Liu et al., Lost in the Middle: How Language Models Use Long Contexts, Transactions of the Association for Computational Linguistics, 2024.
[4] K. Greshake et al., Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection, arXiv:2302.12173, 2023.
[5] C. F. Ruan et al., WebLLM: A High-Performance In-Browser LLM Inference Engine, arXiv:2412.15803, 2024.
[6] R. Levine et al., Llamas on the Web: Memory-Efficient, Performance-Portable, and Multi-Precision LLM Inference with WebGPU, arXiv:2605.20706, 2026.
[7] K. Bonawitz et al., Practical Secure Aggregation for Privacy-Preserving Machine Learning, ACM CCS, 2017.
[8] H. B. McMahan et al., Communication-Efficient Learning of Deep Networks from Decentralized Data, AISTATS, 2017.
[9] C. Dwork and A. Roth, The Algorithmic Foundations of Differential Privacy, Foundations and Trends in Theoretical Computer Science, 2014.
[10] OWASP Foundation, OWASP Top 10 for Large Language Model Applications, OWASP GenAI Security Project, 2025.

[11] National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, 2023.
[12] National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile, NIST AI 600-1, 2024.
[13] National Institute of Standards and Technology, NIST Privacy Framework: A Tool for Improving Privacy Through Enterprise Risk Management, Version 1.0, 2020.
[14] World Wide Web Consortium, WebGPU Specification, W3C Technical Report, 2025.
[15] WebAssembly Community Group, WebAssembly Core Specification, 2026.