

A Generative AI and Retrieval-Augmented Generation Framework for Intelligent, Adaptive, and Automated Enterprise Workflow Architecture

Olakunle Saheed Soyeye
Bay Cove Human Services, Boston, MA, USA

Abstract

Enterprises are adding generative artificial intelligence (AI) to their workflows to classify documents, draft responses, recommend decisions, and handle exceptions. Most discussion focuses on what these capabilities can do. Much less attention goes to the architectural qualities that decide whether they succeed in production: throughput under high volume, adaptability to change, safety of AI outputs, auditability for regulators, reliability when components fail, and modifiability as models and rules evolve. This article takes a software architecture perspective on generative AI and retrieval-augmented generation (RAG) in enterprise workflows. It identifies six quality attributes as the architecture's main drivers and expresses each as a measurable scenario. It then proposes a reference architecture that combines an event streaming backbone, a configurable workflow engine, RAG-based AI services, a guardrail gateway, and an audit-ready compliance ledger. The article maps architectural tactics to each quality attribute and records the key design decisions with their alternatives and rationale. Using a design-oriented approach grounded in ten studies published between 2017 and 2024, it maps the evidence base and traces an event through the architecture. It illustrates the architecture with a payments company handling merchant chargeback disputes at high volume. It argues that intelligent, adaptive, and automated workflows are achieved by designing explicitly for quality attributes, not by adding AI capabilities alone.

Keywords: enterprise architecture, workflow automation, generative AI, retrieval-augmented generation, quality attributes, architectural tactics, event streaming, guardrails, auditability, responsible AI, architecture decisions

1. Introduction

Generative AI is being built into enterprise workflows at a rapid pace. Language models read incoming documents, draft replies, summarize cases, and recommend decisions. RAG grounds

these outputs in organizational knowledge (Ram et al., 2023). The promise is workflows that are more intelligent, able to handle unstructured work, more adaptive to new situations, and more automated.

Many such initiatives nonetheless struggle when they move from pilot to production. The reasons are often architectural rather than algorithmic. A prototype that works for a hundred requests fails at a hundred thousand. An assistant that performs well in testing produces an unsafe output in production. A decision that seemed reasonable cannot be explained to an auditor. A model update breaks a downstream integration. Software architecture research has long held that such outcomes depend on quality attributes, meaning properties such as performance, availability, security, and modifiability, and that architectures must be designed deliberately to achieve them (Bass et al., 2021; Richards & Ford, 2020). Research on architecture for machine learning systems has shown that AI components add new concerns of their own, such as uncertainty, data dependence, and continuous change (Lewis et al., 2021).

Several studies address parts of the problem. Event streaming architectures support high-volume transaction processing (*Event Streaming Architectures for High-Volume Transaction Processing*, 2022; Kleppmann, 2017). Configurable workflow architecture supports adaptable, governed processes (Basireddy, 2022). Guardrails constrain large language model (LLM) behavior (Sannidhanam, 2024). Audit-ready compliance architectures make LLM use traceable (Basireddy, 2023). Responsible AI patterns offer practices for governing AI systems (Lu et al., 2022). This article brings them together from an architecture perspective.

It asks: *Which quality attributes should drive the architecture of generative AI-enabled enterprise workflows, and which architectural tactics and decisions achieve them?* It contributes a mapping of the evidence base (Figures 1 and 2), a set of quality attribute scenarios, a reference architecture (Figure 3), a mapping of tactics to quality attributes, a record of key architecture decisions, an event path through the architecture (Figure 4), and an illustrative application.

2. Related Work

2.1 Quality attributes and architectural tactics

Bass et al. (2021) define software architecture as the set of structures needed to reason about a system. They argue that architecture is shaped mainly by quality attributes, which should be specified as concrete scenarios describing a stimulus, the environment, and a measurable response. Each quality attribute is achieved through architectural tactics. Examples include introducing concurrency for performance, redundancy for availability, and encapsulation for modifiability. Richards and Ford (2020) describe how architects identify the architectural

characteristics that matter most for a system, choose styles such as event-driven or service-based architectures that support them, and record decisions and their trade-offs explicitly.

2.2 Architecture for machine learning systems

Lewis et al. (2021) identify software architecture challenges specific to systems with ML components. These include the uncertainty of model outputs, the need to monitor model behavior in production, the co-evolution of models and data, and the difficulty of specifying requirements for components whose behavior is learned. These challenges apply with even greater force to generative models.

2.3 Retrieval augmentation

Ram et al. (2023) showed that simply placing retrieved documents in the input of an unchanged language model, without retraining it, substantially improves language modeling quality and grounding. This in-context approach makes RAG practical as an architectural component. A retrieval service can supply context to any model through its input, and models can be swapped without retraining the retrieval integration.

2.4 Event streaming and data systems

Kleppmann (2017) describes the design of data-intensive applications and treats the event log as a central architectural element. A durable, ordered, replayable log of events decouples producers from consumers, supports stream processing, and allows derived data to be rebuilt by replaying history. The study on *Event Streaming Architectures for High-Volume Transaction Processing* (2022) examines event streaming for high-volume transactions, in which events are captured durably and processed by many consumers in parallel, giving high throughput and loose coupling between systems.

2.5 Workflow, guardrails, and compliance

Basireddy (2022) proposes **configurable enterprise workflow architecture** for digitizing risk management, with steps, approvals, escalations, and remediation defined as configurable workflows with audit trails. Sannidhanam (2024) examines **guardrails and safety mechanisms for LLM-powered enterprise applications**, including input filtering, output validation, action restrictions, data protection, and escalation. Basireddy (2023) proposes an **audit-ready compliance platform architecture for enterprises that use LLMs in regulated settings**, with traceable records of model inputs, outputs, versions, and decisions.

2.6 Responsible AI patterns

Lu et al. (2022) compiled a catalogue of responsible AI patterns covering governance, process, and product design. Examples include establishing AI risk committees, keeping records of AI system behavior, providing human oversight mechanisms, and designing products for

explainability and safety. Their catalogue connects high-level responsible AI principles to concrete practices that architecture can support.

2.7 Research gap

These studies provide architectural foundations, AI-specific architecture challenges, retrieval approaches, event streaming, workflow, guardrail, compliance, and responsible AI patterns. What is missing is an architecture-centered treatment that starts from the quality attributes of generative AI-enabled workflows and systematically derives tactics and decisions to achieve them. This article provides that treatment.

3. Research Methodology

The study follows a design-oriented research approach grounded in attribute-driven architecture design (Bass et al., 2021). The main quality attributes were identified from the reviewed studies and expressed as scenarios. Tactics and decisions were then selected to satisfy each scenario and combined into a reference architecture. The architecture was demonstrated through an illustrative payments case. It was not evaluated empirically, which is left to future work, for example through scenario-based architecture evaluation.

To make the evidence base transparent, each study was assigned to the theme it primarily supports. Figure 1 shows the distribution across themes, and Figure 2 shows the studies by publication year. These figures describe the literature used in this article. They are not performance measurements.

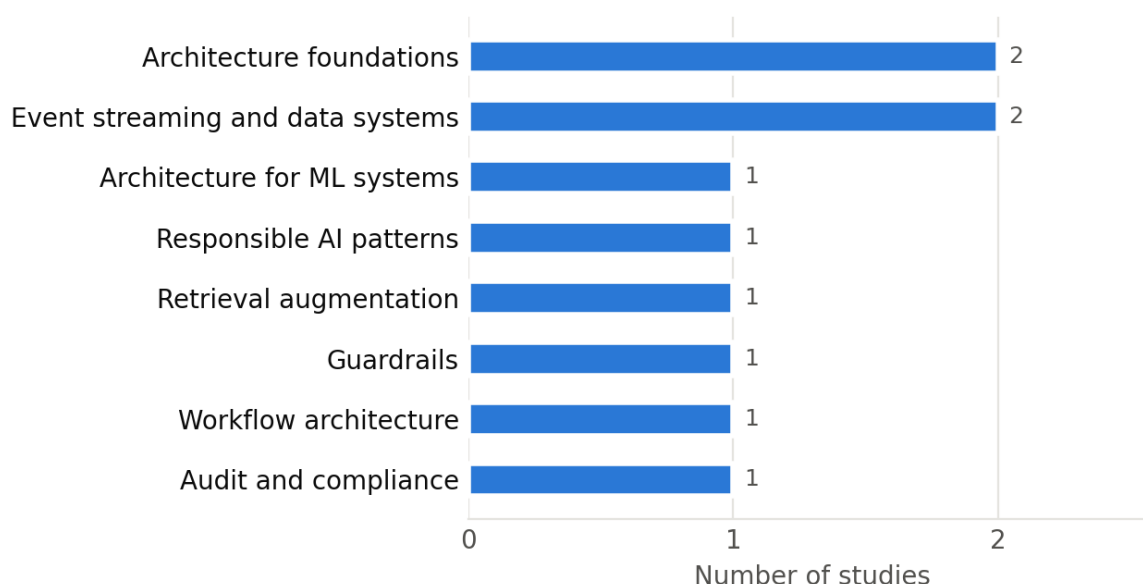


Figure 1. Number of reviewed studies primarily supporting each theme (n = 10)

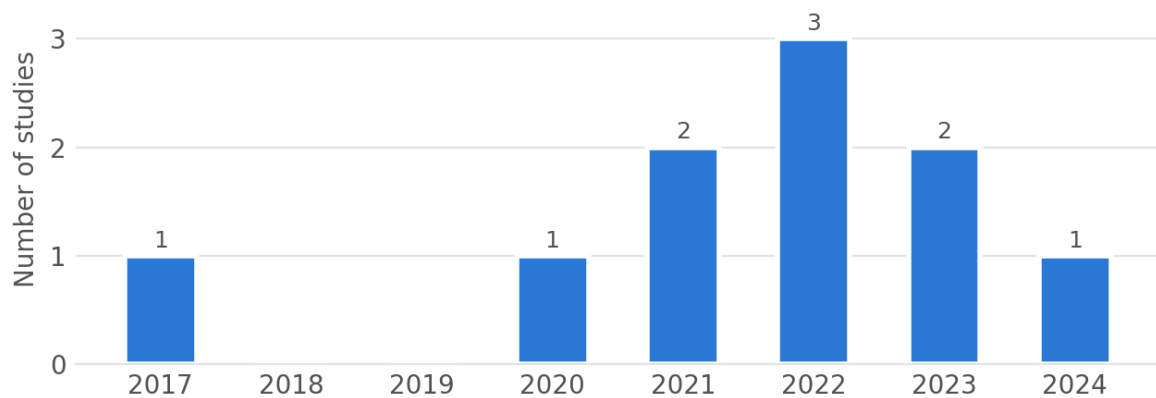


Figure 2. Reviewed studies by publication year (n = 10)

Software architecture foundations and event streaming each have two studies. The AI-specific themes, covering architecture for ML, responsible AI, retrieval, guardrails, workflow, and compliance, each rest on one study. Most studies date from 2021 onward, which reflects how recently architecture research has turned to generative AI.

4. Architectural Drivers

Following Bass et al. (2021), the architecture is driven by six quality attributes, each expressed as a scenario (Table 1).

Table 1. Quality attribute scenarios for generative AI-enabled workflows

Quality attribute	Scenario	Why it matters	Supporting studies
Throughput	During a volume peak of several times normal load, all events are processed without loss and AI steps complete within their service level	High-volume workflows cannot drop or delay cases	<i>Event Streaming Architectures</i> (2022); Kleppmann (2017)
Adaptability	A new rule, approval step, or model version is introduced without changing application code, and takes effect after governed approval	Workflows must change as quickly as policy and models do	Basireddy (2022); Richards & Ford (2020)
Safety	An AI output that is unsupported, contains restricted	Generative outputs are uncertain and can cause harm	Sannidhanam (2024); Lewis et al. (2021)

Quality attribute	Scenario	Why it matters	Supporting studies
	data, or proposes a disallowed action is blocked or escalated before it affects a case		
Auditability	For any AI-influenced decision, an auditor can retrieve the inputs, retrieved sources, model and prompt versions, output, and human decision	Regulators require traceable decisions	Basireddy (2023); Lu et al. (2022)
Reliability	If the AI service is unavailable, cases continue through a defined fallback path without loss	AI components fail and change more often than core systems	Lewis et al. (2021); Bass et al. (2021)
Modifiability	A language model or retrieval index is replaced with a new one without changes to workflows or integrations	Models and knowledge evolve continuously	Ram et al. (2023); Bass et al. (2021)

5. Reference Architecture

The reference architecture has five main elements (Figure 3).

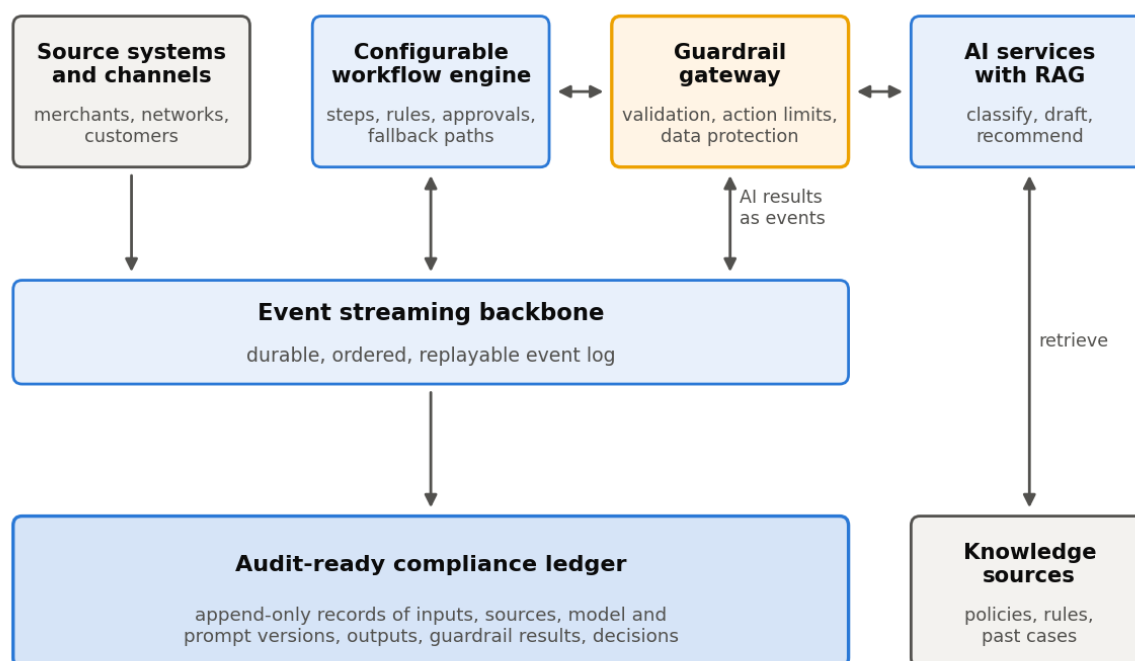


Figure 3. Reference architecture for generative AI-enabled enterprise workflows

Event streaming backbone. All business events flow through a durable, ordered, replayable event log (Kleppmann, 2017; *Event Streaming Architectures*, 2022). Examples include cases opened, documents received, AI results produced, decisions made, and actions completed. The workflow engine, AI services, and compliance ledger all consume and produce events, which decouples them from one another.

Configurable workflow engine. The engine runs workflows defined as configuration, including steps, rules, approvals, and escalations. It determines when an AI step is invoked and what happens with its result (Basireddy, 2022).

AI services. AI capabilities such as classification, summarization, drafting, and recommendation run as independent services. Each uses a RAG component that retrieves relevant policies, procedures, and past cases and supplies them in the model's input (Ram et al., 2023). Models are accessed through a stable interface, so they can be replaced without affecting callers.

Guardrail gateway. All AI requests and responses pass through a gateway that applies input filtering, output validation against case data and retrieved sources, restrictions on proposed actions, and sensitive data protection. Outputs that fail are blocked or escalated (Sannidhanam, 2024).

Audit-ready compliance ledger. Every AI interaction and decision is written to an append-only ledger. Each record holds the inputs, retrieved sources, model and prompt versions,

output, guardrail results, and human decision, so that it can be traced end to end (Basireddy, 2023; Lu et al., 2022).

5.1 Tactics mapped to quality attributes

Table 2 maps architectural tactics to the quality attributes they support.

Table 2. Architectural tactics for each quality attribute

Quality attribute	Tactics	Architectural element
Throughput	Asynchronous events, parallel consumers, horizontal scaling of AI services, back-pressure	Event backbone, AI services
Adaptability	Configuration over code, externalized rules, versioned workflow definitions	Workflow engine
Safety	Input and output validation, action allow-lists, confidence thresholds, human escalation	Guardrail gateway, workflow engine
Auditability	Append-only records, version stamping of models and prompts, source citation	Compliance ledger, AI services
Reliability	Timeouts, circuit breakers, fallback to rule-based or human path, event replay	Workflow engine, event backbone
Modifiability	Stable model interface, independent AI services, retrieval separated from generation	AI services

5.2 Key architecture decisions

Following Richards and Ford (2020), the main decisions are recorded with their alternatives and rationale (Table 3).

Table 3. Key architecture decisions

Decision	Alternatives considered	Choice	Rationale
Integration style	Synchronous calls between services; event streaming	Event streaming	Supports high throughput, loose coupling, and replay for audit and recovery

Decision	Alternatives considered	Choice	Rationale
Where AI sits in the process	AI controls the process; AI as a step within a workflow	AI as a bounded workflow step	Keeps process state, approvals, and accountability in the workflow engine
Knowledge grounding	Fine-tune models on company data; retrieval at request time	Retrieval at request time	Knowledge can be updated without retraining; sources can be cited and models swapped
Safety enforcement	Checks inside each AI service; a shared guardrail gateway	Shared gateway	Applies policies consistently and makes them auditable in one place
Audit storage	Application logs; dedicated append-only ledger	Dedicated ledger	Gives complete, tamper-evident, queryable records for regulators
Failure handling	Retry until the AI succeeds; fallback path	Fallback path	Cases keep moving when AI services fail or return low-confidence results

5.3 Event path through the architecture

Figure 4 traces a single case event through the architecture.

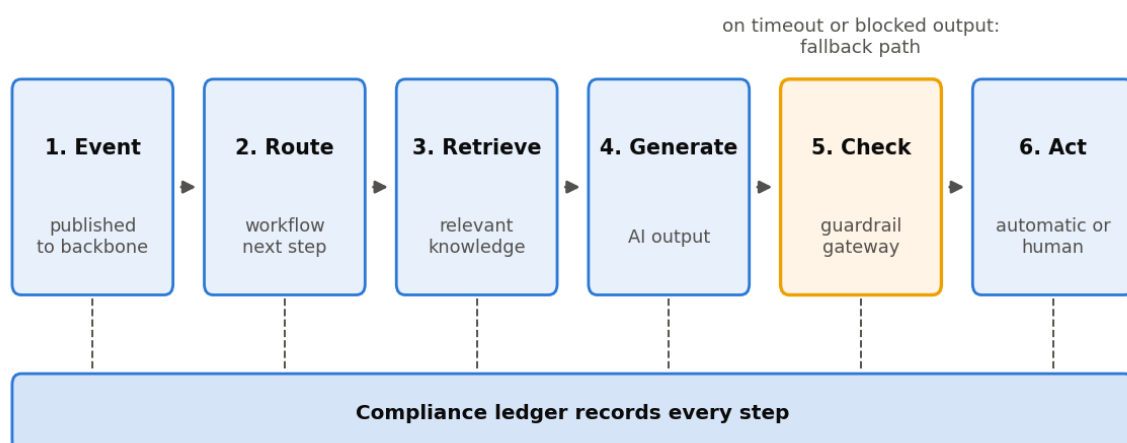


Figure 4. Path of a case event through the architecture

An incoming business **event** is published to the backbone. The workflow engine **routes** the case to its next step. When the step is an AI step, the AI service **retrieves** relevant knowledge

and **generates** an output. The guardrail gateway **checks** it, and the engine then **acts** on it automatically or through a person. Each step is **recorded** in the compliance ledger. If the AI service times out or the guardrails block the output, the engine switches the case to its fallback path. The case continues, and the fallback is recorded.

6. Illustrative Case: Chargeback Disputes at a Payments Company

Consider a payments company that processes transactions for thousands of merchants and handles tens of thousands of chargeback disputes each month. For each dispute, it must gather evidence from the merchant, assess it against card network rules, and submit a response by a strict deadline. Volumes spike after major shopping periods.

Throughput. Each dispute notification from card networks is published to the event backbone (*Event Streaming Architectures*, 2022). During the post-holiday spike, dispute events rise to several times normal volume. Consumers and AI services scale horizontally, and the event log absorbs the backlog without loss (Kleppmann, 2017).

Intelligence. When merchant evidence arrives, an AI service classifies the documents and drafts a dispute response. It retrieves the relevant card network rule, the merchant's past successful responses for similar disputes, and the company's response templates (Ram et al., 2023).

Safety. The guardrail gateway checks that the draft cites evidence actually present in the merchant's submission and contains no full card numbers. It also checks that the recommended action is one the workflow allows for this dispute type (Sannidhanam, 2024). A draft that claims a signed delivery receipt the merchant did not provide is blocked and routed to an analyst.

Adaptability. Midway through the quarter, a card network changes its evidence requirements for one dispute category. The company adds the new rule to the retrieval knowledge base and updates the workflow configuration to add an evidence check step. Both changes are approved through the change workflow and take effect without code changes (Basireddy, 2022).

Reliability. During a brief outage of the language model provider, AI steps time out. The circuit breaker switches affected disputes to a template-based fallback path with analyst review, so deadlines are still met.

Auditability and modifiability. Later, a card network audits the company's dispute handling. For each sampled dispute, the compliance ledger shows the evidence, retrieved rules, model and prompt versions, draft, guardrail results, and analyst decision (Basireddy, 2023; Lu et al., 2022). When the company switches to a newer language model, it changes only the AI service configuration. Workflows, guardrails, and integrations are unaffected.

The case shows each quality attribute scenario in Table 1 being met by the tactics and decisions in Tables 2 and 3.

7. Discussion

7.1 Implications

For research, the article shows how attribute-driven architecture design can be applied to generative AI-enabled workflows. It connects foundational architecture thinking (Bass et al., 2021; Richards & Ford, 2020) with AI-specific concerns (Lewis et al., 2021; Lu et al., 2022) and with specific mechanisms for events, workflows, guardrails, and compliance. The evidence mapping suggests that AI-specific architecture research is still thin. For practitioners, the article offers a practical method. Specify quality attributes as measurable scenarios before choosing technologies. Select tactics that achieve each scenario. Record decisions and their trade-offs. Make sure the architecture can meet its throughput, safety, audit, and reliability requirements when AI components misbehave, not only when they work.

7.2 Limitations and future work

The architecture was demonstrated through an illustrative case and draws on a focused set of ten studies, and the evidence mapping reflects only these studies. Future work should evaluate the architecture using scenario-based methods and in real deployments, measuring throughput under load, guardrail effectiveness, audit retrieval time, recovery from AI failures, and the effort needed to replace models. Further research is needed on quality attribute scenarios specific to generative AI, on tactics for managing output uncertainty, on architecture patterns for multi-agent workflows, and on standard formats for AI audit records.

8. Conclusion

Intelligent, adaptive, and automated enterprise workflows depend on more than generative AI capabilities. They depend on an architecture designed for the qualities that production demands. This article identified six quality attributes as the architecture's drivers: throughput, adaptability, safety, auditability, reliability, and modifiability. It proposed a reference architecture combining an event streaming backbone, a configurable workflow engine, RAG-based AI services, a guardrail gateway, and an audit-ready compliance ledger. It mapped architectural tactics to each attribute and recorded the key design decisions with their rationale. The chargeback dispute case showed the architecture handling volume spikes, blocking unsafe outputs, adapting to rule changes, surviving an AI outage, and satisfying an audit. The central conclusion is that generative AI delivers durable value in enterprise workflows when the architecture is designed explicitly for the qualities that production requires.

References

1. Bass, L., Clements, P., & Kazman, R. (2021). *Software architecture in practice* (4th ed.). Addison-Wesley.
2. Basireddy, S. R. (2022). Digitizing risk management through configurable enterprise workflow architecture. *SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology*, 14(4), 231–239.
3. Basireddy, S. (2023). Audit-ready compliance platform architecture for large language model regulated enterprises. *SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology*, 15(01), 226–232. <https://doi.org/10.18090/samriddhi.v15i01.40>
4. Event streaming architectures for high-volume transaction processing. (2022). *International Journal of Advance Industrial Engineering*, 10(01), 1–8.
5. Kleppmann, M. (2017). *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. O'Reilly Media.
6. Lewis, G. A., Ozkaya, I., & Xu, X. (2021). Software architecture challenges for ML systems. In *Proceedings of the 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (pp. 634–638).
7. Lu, Q., Zhu, L., Xu, X., Whittle, J., Zowghi, D., & Jacquet, A. (2022). *Responsible AI pattern catalogue: A collection of best practices for AI governance and engineering* (arXiv:2203.00905). arXiv.
8. Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., & Shoham, Y. (2023). In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11, 1316–1331.
9. Richards, M., & Ford, N. (2020). *Fundamentals of software architecture: An engineering approach*. O'Reilly Media.
10. Sannidhanam, A. H. (2024). Guardrails and safety mechanisms for LLM-powered enterprise applications. *International Journal of Engineering Science & Humanities*, 14(3), 273–285. <https://www.ijesh.com/j/article/view/933>