Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Engineering Enterprise-Grade Generative AI: Architectures, Agentic Frameworks, and Protocols for Production-Ready LLM Systems

Dipans Verma
PhD Scholar, Department of Mathematics Amity University Maharashtra
Mumbai, India

Ganesh Randave
Research Scholar, Dept. of Applied Science and Humanities, MIT-ADT University Pune, India

Sandeep Kulkarni*
Assistant Professor
Ajeenkya DY Patil University Pune, India

Anshul Tiwari
Student
Lokmanya Tilak College of Engineering Mumbai University, India

Saksham Koli
B.Tech Student
Amity University Maharashtra
Mumbai, India

Ansh Agrawal
B.Tech Student Ajeenkya DY Patil University
Pune, India

Vishal Gosavi
Assistant Professor Ajeenkya DY Patil University
Pune, India

***Abstract*—Databricks has harnessed the power of Generative AI for business AI in their full-stack platform, driven by Large Language Models (LLMs), to revolutionize enterprise software ecosystems through advancements in reasoning, automation, and intelligent content generation. Leveraging the true potential of LLM-based systems in production environments, however, requires far more than evaluating models in isolation. Deployment of modern, cloud-native applications at enterprise grade requires comprehensive architectural design, scalable infrastructure, orchestrated multi-agent coordination, governance frameworks, and observability modules with complete lifecycle management to ensure reliability, security, compliance, and sustainable performance. This paper addresses these challenges by presenting a comprehensive engineering framework for developing production-ready GenAI systems that are modular, interoperable, scalable, and operationally resilient. The proposed framework integrates best practices spanning three domains: system design (SD), machine learning operations (MLOps), and agentic architectures, in order to be appropriate for real-world enterprise adoption.**

***Index Terms*—Generative Artificial Intelligence, Large Language Models, Retrieval-Augmented Generation, Multi-Agent Systems, MLOps, Enterprise AI Systems, Agentic Architectures**

I. INTRODUCTION

Generative AI—driven by large language models like GPT-3, LLaMA, LaMDA, and GPT-4—has completely transformed how intelligent software works. These models understand language, reason through problems, summarize content, assist with translations, write code, and offer automated decision support. Their flexibility as foundation models makes LLMs useful across a wide variety of fields, placing GenAI at the heart of the next wave of enterprise technology.

Making these models work reliably in real business settings, however, is far from straightforward. Systems must respond fast, scale on demand, comply with governance requirements, prevent hallucinations, maintain data security, control spending, ensure reproducibility, and sustain performance throughout their lifetime. Research into aligning models with human feedback demonstrates how crucial it is to make them behave reliably and remain controllable [17]. There is a distinct layer of concern around AI safety and risk, requiring strong safeguards and governance at the forefront [7].

Simply plugging in a model and running inference is not enough. Enterprises need systems that are robust, auditable, observable, and effective across distributed networks. The challenges are not purely technical—they are also software engineering challenges. Integrating AI into production pipelines is inherently complex, and frameworks like MLOps underscore how much expertise and coordination that demands [10].

This paper presents a practical engineering blueprint built on several key components: Retrieval-Augmented Generation (RAG) to keep outputs grounded in facts, combining sparse and dense retrieval methods with large-scale similarity search engines such as FAISS [8]; multi-agent orchestration drawing on the latest agent-based systems [15], [23]; structured Agent-to-Agent (A2A) communication protocols; the Model Context Protocol (MCP) for packaging context and maintaining traceability [16]; and observability-focused MLOps for monitoring,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

evaluation, and lifecycle management [2], [10]. Combining these components into a single, well-structured framework yields LLM systems that scale reliably and remain compliant in real enterprise environments.

II. RELATED WORK

Retrieval-Augmented Generation (RAG) has become a standard approach for improving the reliability of large language models. By combining external knowledge retrieval with generative modeling—pulling in relevant information at inference time to shape the model's output—these systems reduce hallucinations and produce more accurate answers for fact-intensive tasks [11].

Model alignment techniques, particularly Reinforcement Learning from Human Feedback (RLHF), have pushed generative models to behave more safely and in closer accordance with human expectations [17]. Supervised fine-tuning and direct preference optimization have further improved model predictability and output quality [18].

Multi-agent LLM systems represent a more recent direction, distributing difficult reasoning tasks among specialized agents [22], [23]. This modular approach enables parallel processing, structured tool use, and substantially greater scalability.

Despite these advances, most research addresses only individual components in isolation. Few efforts bring together retrieval, agent orchestration, standardized communication, and enterprise observability into a single engineering framework. As enterprise adoption grows, there is a clear need for a unified blueprint that fuses accurate grounding, modular reasoning, seamless collaboration, and reliable monitoring within a production-ready design.

III. Design Principles for Enterprise GenAI

The enterprise GenAI architecture rests on five core design principles that keep the system scalable, reliable, and aligned with organizational governance while ensuring long-term sustainability.

1) **Modularity:** The system decomposes into distinct components—retrieval layers, agent orchestration modules, protocol interfaces, and monitoring subsystems. Because each component is independently operable, individual parts can be developed, tested, scaled, or replaced without disrupting the rest of the architecture.
2) **Observability:** Continuous insight into system behavior is essential in production. Observability encompasses performance monitoring, failure detection, hallucination tracking, and cost analytics, enabling early issue identification and strong governance.
3) **Safety and Alignment:** The architecture embeds alignment strategies, guardrails, and policy enforcement so that outputs adhere to organizational standards, regulatory requirements, and ethical AI guidelines [6], [7].
4) **Reproducibility:** Every model input, retrieved context, agent decision, and piece of system metadata is logged and versioned, enabling full traceability for scenario replay, auditing, and cross-environment experimentation.
5) **Interoperability:** Standardized communication protocols and structured data interfaces allow the system to connect with different models, tools, cloud platforms, and enterprise services, reducing vendor lock-in and simplifying integration.

To formalize operational reliability, observability is defined as a composite function:

$$O = f(t_{\text{latency}},\ S_{\text{rate}},\ H_{\text{ratio}},\ C_{\text{ops}}) \tag{1}$$

where t_{latency} tracks average response time; S_{rate} measures the percentage of requests completed without failures; H_{ratio} captures the frequency of factually incorrect or unsupported outputs; and C_{ops} aggregates all operational costs including compute, API fees, and infrastructure overhead. Together, these metrics provide a comprehensive snapshot of real-world system performance across speed, stability, accuracy, and cost-effectiveness.

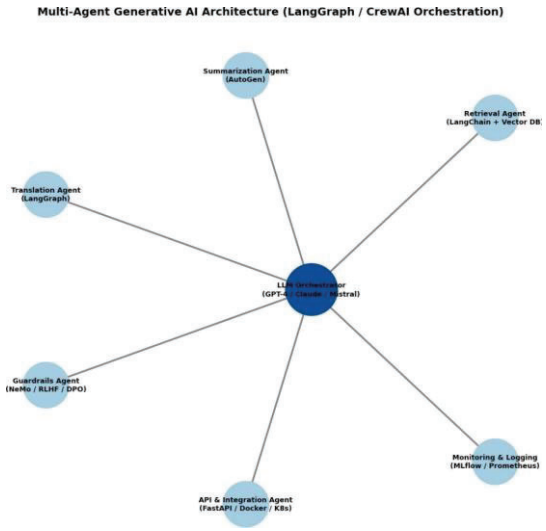

Fig. 1. Layered enterprise GenAI architecture showing data inges-tion, hybrid retrieval, multi-agent orchestration, observability, and application interface layers.

IV. SYSTEM ARCHITECTURE

The enterprise GenAI system uses a five-layer architecture, with each layer serving a distinct role while integrating smoothly with the others to support fault isolation, independent scaling, and clean operation in a distributed environment.

1) **Data Ingestion and Indexing Layer:** This foundational layer pulls data from diverse sources—databases, document repositories, APIs, and real-time streams—then scrubs and transforms it, generating vector embeddings for semantic search while maintaining keyword indexes for precise lexical lookup.
2) **Hybrid Retrieval Layer:** This layer combines dense semantic search (using embeddings to capture meaning and context) with traditional keyword search (for precision and exact matching). The combination produces

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

higher-quality results and reduces missed or irrelevant answers.

3) **Multi-Agent Orchestration Layer:** Acting as the conductor of the system, this layer manages teams of specialized agents—planners, data retrievers, fact-checkers, and response generators. Dividing work this way makes complex tasks manageable and the overall system easier to scale [22], [23].
4) **Observability and MLOps Layer:** This layer monitors system behavior continuously, tracking latency, data drift, hallucination rates, and costs. It supports rolling updates, model versioning, and rapid fault remediation [2], [10], [24].
5) **Application Interface Layer:** The topmost layer is where users and administrators interact with the system through APIs and dashboards. It enforces authentication, authorization, and compliance policies to keep the system secure and enterprise-ready.

V. Hybrid Retrieval

To formally define retrieval fusion, the hybrid scoring function is expressed as:

$$R_{\text{hybrid}} = \alpha R_{\text{dense}} + (1 - \alpha) R_{\text{sparse}} \tag{2}$$

where R_{dense} denotes the semantic similarity score derived from embedding-based vector search, and R_{sparse} represents the lexical relevance score computed through traditional term-based ranking methods such as BM25 [19]. The weighting parameter $\alpha \in [0, 1]$ governs the balance between contextual semantic alignment and exact keyword matching.

Dense retrieval leverages embedding representations trained on large neural models to understand queries semantically beyond exact keyword matches. Sparse retrieval contributes precision for domain-specific terminology and structured queries. The weighted aggregation allows the system to adjust its reliance on each mode based on domain requirements, query complexity, and performance targets. Blending semantic abstraction with exact term matching, hybrid retrieval maintains contextual accuracy, handles heterogeneous enterprise data more robustly, and delivers reliable answers at scale.

VI. Agentic Framework and Protocols

To scale reasoning in enterprise GenAI systems while maintaining modularity and governance, the architecture standardizes communication and context management through two core protocols.

A. Model Context Protocol (MCP)

The Model Context Protocol (MCP) [16] provides a standardized method for packaging context before it reaches a model or agent. Rather than relying on unstructured prompts, MCP organizes context in a structured, auditable format aligned with enterprise governance requirements:

$$C_{\text{MCP}} = \langle \text{Metadata}, \text{Payload}, \text{Trace}, \text{Security} \rangle \tag{3}$$

The components are defined as follows:

- **Metadata:** Source provenance, timestamp, version, and domain classification.
- **Payload:** Structured data intended for the model—retrieved documents, intermediate reasoning outputs, or tool results.
- **Trace:** An immutable record of actions taken, supporting reproducibility and audit trails.
- **Security:** Access permissions, data sensitivity classifications, and applicable compliance rules.

By adhering to this structure, MCP enables precise traceability and controlled management of the information presented to models, satisfying enterprise requirements for governance and lifecycle accountability.

B. Agent-to-Agent (A2A) Protocol

The Agent-to-Agent (A2A) protocol organizes communication and cooperation among specialized agents, replacing informal prompt chaining with standardized message formats. This supports task decomposition, modular reasoning, and orderly management of complex multi-step workflows. Formally, an A2A message is defined as:

$$M_{\text{A2A}} = \langle \text{Goal}, \text{Context}, \text{Constraints}, \text{TraceID} \rangle \tag{4}$$

The fields are interpreted as follows:

- **Goal:** The objective or problem the receiving agent is tasked with solving.
- **Context:** All inputs, knowledge, or intermediate results required to complete the task.
- **Constraints:** Rules, limits, or validation requirements governing the agent's response.
- **TraceID:** A unique identifier linking the interaction to the broader workflow for transparency and debugging.

With this structure, agents reason in predictable, sequential steps, collaborate without conflicts, and every step remains auditable. Together, MCP and A2A provide the foundation for agentic AI systems that are scalable, interoperable, and governance-aware.

VII. Experimental Evaluation

The proposed enterprise GenAI framework is evaluated using quantitative and qualitative performance metrics capturing accuracy, responsiveness, and operational efficiency across real-world enterprise use cases.

System accuracy is defined as:

$$\text{Accuracy} = \frac{\text{Correct Responses}}{\text{Total Queries}} \tag{5}$$

where *Correct Responses* denotes outputs satisfying factual correctness, policy compliance, and task completion criteria [3].

Average latency is computed as:

$$\text{Latency}_{\text{avg}} = \frac{\sum_{i=1}^{N} t_i}{N} \tag{6}$$

where t_i is the response time for query i and N is the total number of queries. This end-to-end measure encompasses retrieval, generation, and all background processing steps.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

A. Case Study 1: Multilingual Customer Interaction

The system was deployed in a customer service platform handling queries across multiple languages, aiming to automate resolution and escalation workflows. Key outcomes:

- Manual handoffs to human representatives decreased by **40%**.
- First-response accuracy improved by **35%**.
- End-to-end average latency was **750 ms**, meeting real-time conversation requirements.

This demonstrates that combining hybrid retrieval with multi-agent coordination effectively handles language diversity while maintaining responsiveness.

B. Case Study 2: Predictive Marketing Platform

The system was integrated into a marketing platform for campaign optimization and automated reporting. Key outcomes:

- Campaign efficiency improved by **25%** through better targeting and reduced budget waste.
- User engagement increased by **30%** due to improved personalization and contextual relevance.
- Reporting time was reduced by **40%** through automation and agent-driven orchestration.

Together, these case studies confirm the framework's ability to scale and deliver measurable business value across diverse enterprise environments.

VIII. DISCUSSION

Combining semantic vector search with lexical keyword retrieval substantially improves factual accuracy, reduces hallucinations, and strengthens robustness against heterogeneous enterprise data. Decomposing the system into specialized agents enables more parallel processing, easier maintenance, and smoother upgrades, allowing the workflow to adapt without requiring architectural overhaul.

Standardizing protocols through MCP and A2A communication ensures interoperability with different tools, services, and models, avoiding vendor lock-in and simplifying third-party integration. Continuous observability—monitoring model outputs, latency, data drift, and system health—reduces surprises, accelerates fault resolution, and keeps the system aligned with evolving business requirements.

IX. LIMITATIONS AND FUTURE WORK

The current framework has several limitations that motivate future research:

- **External Dependencies:** Reliance on third-party models, APIs, and services introduces latency variability, pricing uncertainty, and provider availability risks.
- **Long-Context Handling:** Although retrieval mitigates some context limitations, managing very long data streams remains demanding. Token limits, memory requirements, and inference costs compound at scale.
- **Multi-Agent Alignment:** Complex agent interactions can produce unexpected emergent behaviors. Maintaining consistent alignment and safety across all agents is non-trivial.
- **Operational Complexity:** Additional orchestration, monitoring, and compliance layers increase infrastructure and maintenance costs if not carefully managed.

Future work should focus on:

- Self-verifying pipelines that perform cross-checking and built-in factual validation before delivering responses.
- Smarter context compression and summarization techniques to maximize information density within tight token budgets.
- Multimodal agents capable of processing text, vision, speech, and structured business data within unified workflows.
- Compliance auditing systems built directly into interaction pipelines for automatic policy enforcement and regulatory reporting.

Advancing in these directions will make generative AI more scalable, trustworthy, and ready for serious enterprise deployment.

X. CONCLUSION

This paper presents a comprehensive framework for deploying enterprise-scale generative AI. By integrating retrieval-augmented generation, hybrid search strategies, coordinated multi-agent orchestration, and standardized protocols (MCP and A2A), together with robust MLOps practices, the architecture is modular, scalable, and operationally ready for real-world business environments. The combination of advanced retrieval, agent workflows, and standardized communication makes the system reliable, traceable, and governable. Continuous monitoring and lifecycle management sustain high performance and model accountability over time. Validation through real-world case studies demonstrates significant gains in efficiency, scalability, and practical readiness. This blueprint gives organizations a path to build secure, maintainable, and compliance-friendly generative AI systems capable of supporting complex, long-running enterprise workflows.

ACKNOWLEDGMENT

The authors thank Ajeenkya DY Patil University and Amity University Maharashtra for providing the collaborative environments and computational resources that supported this research.

REFERENCES

[1] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *Proc. 41st Int. Conf. Software Engineering*, 2019, pp. 291–300.

[2] T. Ash, A. Brown, M. Garcia, and J. Sim, "MLOps: Continuous delivery and automation pipelines in machine learning," *Journal of Systems and Software*, vol. 198, p. 111571, 2023.

[3] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora et al., "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

[5] Google Cloud, “Google Cloud Vertex AI Documentation,” 2023. [Online]. Available: https://cloud.google.com/vertex-ai
[6] D. Hendrycks, C. Burns, S. Basart et al., “Unsolved problems in ML safety,” *arXiv preprint arXiv:2109.13916*, 2021.
[7] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, and J. Steinhardt, “Ethics and safety in large language models: A survey,” *arXiv preprint arXiv:2302.06476*, 2023.
[8] J. Johnson, M. Douze, and H. Je´gou, “Billion-scale similarity search with GPUs,” *IEEE Transactions on Big Data*, 2019.
[9] J. Kreps, “Kafka: A distributed messaging system,” 2011.
[10] D. Kreuzberger, L. Ku¨hne et al., “Machine learning operations (MLOps): Overview, definition, and architecture,” *IEEE Access*, 2023.
[11] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, M. Kuc˘erova´, S. Min, W.-t. Yih, T. Rockta¨schel et al., “Retrieval-augmented generation for knowledge-intensive NLP tasks,” in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 9459–9474.
[12] S. Melnik et al., “Dremel: Interactive analysis of web-scale datasets,” in *VLDB*, 2010.
[13] S. Newman, *Building Microservices*. Sebastopol, CA: O’Reilly, 2015.
[14] OpenAI, “GPT-4 technical report,” 2023. [Online]. Available: https://cdn.openai.com/papers/gpt-4.pdf
[15] OpenAI, “OpenAI agent framework: Multi-agent coordination and tool use,” 2023. [Online]. Available: https://openai.com/research
[16] OpenAI, “Model Context Protocol (MCP),” 2024. [Online]. Available: https://github.com/modelcontextprotocol
[17] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray et al., “Training language models to follow instructions with human feedback,” *arXiv preprint arXiv:2203.02155*, 2022.
[18] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, and C. Finn, “Direct preference optimization: Your language model is secretly a reward model,” *arXiv preprint arXiv:2305.18290*, 2023.
[19] S. Robertson and H. Zaragoza, “The probabilistic relevance framework: BM25 and beyond,” *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.
[20] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du et al., “LaMDA: Language models for dialog applications,” *arXiv preprint arXiv:2201.08239*, 2022.
[21] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozie`re, N. Goyal, E. Hambro, F. Azhar et al., “LLaMA: Open and efficient foundation language models,” *arXiv preprint arXiv:2302.13971*, 2023.
[22] M. Wooldridge, *An Introduction to MultiAgent Systems*. Hoboken, NJ: Wiley, 2009.
[23] X. Wu, K. Li, X. Qiu, Y. Chen, Y. Zhou, Q. Liu et al., “AutoGen: Enabling next-gen LLM applications via multi-agent conversation framework,” 2023. [Online]. Available: https://github.com/microsoft/autogen
[24] M. Zaharia et al., “MLflow: Platform for managing the ML lifecycle,” 2019. [Online]. Available: https://mlflow.org