International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Enterprise Conversational AI Platforms: Design and Implementation of a Scalable Co-Pilot Chat Assistant for Intelligent Decision Support

Arun Mallur Chandrashekar

Abstract: Enterprise software platforms present significant cognitive overhead for end users who must simultaneously navigate multiple subsystems, interpret heterogeneous data sources, and execute time-sensitive workflows without consistent decision support. Traditional mitigation strategies static documentation portals and human-assisted support channels are inadequate at scale: the former suffers from knowledge staleness and zero contextual adaptability, while the latter imposes linear headcount costs and variable resolution quality. This paper presents the design and production implementation of a scalable AI co-pilot platform deployed at RealPage, a major enterprise property management software provider serving over 50,000 active users. The system integrates a hybrid Retrieval-Augmented Generation (RAG) framework combining dense vector search with BM25 sparse retrieval and Reciprocal Rank Fusion, a multi-stage re-ranking pipeline employing neural cross-encoders and LLM-based passage selection, and a monorepo-based serverless architecture that unifies reactive frontend state machines with backend AI orchestration. Real-time response streaming is delivered through Server-Sent Events with structured multi-tier escalation routing to CRM-integrated human agents. The platform implements the ReAct agentic framework for multi-step workflow execution and incorporates corrective and self-reflective RAG mechanisms to minimize hallucination in production. Production evaluation across the property management domain demonstrates substantial operational gains: up to 50% reduction in per-user task time, 11 hours saved per user per month, and a 20% reduction in inbound support call volumes. Retrieval quality improvements of 20-30% over baseline single-stage RAG were measured, with the chunking and re-ranking pipeline delivering 25-40% latency reduction and 15-25% accuracy improvement. Deployment efficiency gains from the monorepo architecture reduced integration defects by 30-40% and accelerated feature delivery by 25-35%. These results, drawn from author primary research data, establish the viability of hybrid RAG co-pilot architectures as production-grade enterprise decision support infrastructure.

Keywords: *enterprise conversational AI; AI co-pilot; retrieval-augmented generation (RAG); hybrid retrieval; reciprocal rank fusion; cross-encoder re-ranking; ReAct agentic framework; monorepo architecture; serverless deployment; CRM integration; production LLM systems.*

1. Introduction

Modern enterprise software ecosystems present a compounding usability challenge. Property management platforms, for example, require users to simultaneously navigate lease compliance modules, maintenance workflow systems, financial reporting dashboards, and vendor coordination interfaces often within a single operational session. The cognitive overhead of this multi-system environment directly increases error rates, support dependency, and mean time to task resolution.

Traditional decision support mechanisms are insufficient at enterprise scale. Static documentation portals provide immediate but context-free responses that become stale between update cycles and cannot adapt to user-specific system states. Human support agents offer contextual understanding but impose headcount-linear cost structures and introduce resolution time variance of minutes to hours. Neither approach scales to the demands of a 50,000+ user enterprise deployment.

Large language model (LLM)-based co-pilots represent a qualitatively different paradigm: contextual, real-time, and personalized decision support that adapts to user intent and system state without requiring human intermediation for the majority of queries. However, deploying such systems at enterprise scale introduces substantial engineering challenges: retrieval quality under diverse query types, response latency at concurrent scale, integration with live CRM and workflow systems, and governance requirements for auditability and content safety.

This paper addresses these challenges through the design and production implementation of a scalable AI co-pilot platform for RealPage. The system serves over 50,000 active users in the property management domain and constitutes the largest reported enterprise co-pilot deployment in this sector. The primary contributions of this work are: (1) a hybrid RAG architecture combining dense and sparse retrieval with reciprocal rank fusion and multi-stage re-ranking; (2) a monorepo-based serverless architecture enabling type-safe frontend-backend integration at scale; (3) real-time streaming response delivery with structured multi-tier escalation routing; (4) CRM integration via ReAct-based agentic workflow orchestration; and (5) comprehensive production evaluation demonstrating measurable gains in user efficiency, support deflection, and operational reliability.

Lamar University, USA

ORCID ID: 0009-0005-6733-7227

2. Related Work

2.1. Retrieval-Augmented Generation

The foundational RAG architecture was introduced by Lewis et al. [1], establishing the pattern of conditioning LLM generation on retrieved external passages to reduce hallucination and extend knowledge currency. Gao et al. [2] provide a comprehensive survey of RAG variants, categorizing approaches along sparse, dense, and hybrid retrieval dimensions and identifying hybrid fusion as the state-of-the-art for enterprise applications requiring both keyword precision and semantic coverage. Asai et al. [3] introduced Self-RAG, an approach in which the model generates special reflection tokens to assess the necessity and quality of retrieved passages during generation, enabling adaptive retrieval that reduces unnecessary context injection. Yan et al. [21] proposed Corrective RAG (CRAG), which implements a confidence-based retrieval evaluator that triggers corrective retrieval when initial passages fail quality thresholds. Sarthi et al. [12] introduced RAPTOR, a hierarchical document chunking strategy using recursive summarization to enable retrieval across

long documents directly applicable to the regulatory and compliance documentation prevalent in property management domains.

2.2. Re-Ranking and Context Assembly

Nogueira and Cho [14] established cross-encoder passage re-ranking with BERT as a standard approach for improving retrieval precision after initial candidate selection, with the cross-encoder architecture scoring query-passage pairs jointly rather than independently encoding them. Sun et al. [17] demonstrated that LLMs can serve as effective re-rankers through permutation-based listwise ranking, providing a complementary approach to neural cross-encoders that leverages generation model understanding. Ma et al. [18] showed that query rewriting before retrieval reformulating user queries to improve alignment with the document index yields substantial improvements in recall and precision. Izacard and Grave [13] demonstrated Fusion-in-Decoder, which reads multiple retrieved passages jointly at generation time, providing a strong multi-passage baseline. Shi et al. [11] identified the significant risk of performance degradation when irrelevant context is included in the prompt, motivating the need for aggressive re-ranking to filter low-quality candidates before context assembly.

2.3. Agentic LLMs and Conversational AI

Yao et al. [7] introduced the ReAct framework, which interleaves reasoning traces with action execution to enable LLMs to perform multi-step tool-use tasks the direct foundation for the CRM workflow orchestration implemented in this co-pilot. Wang et al. [8] provide a comprehensive survey of LLM agent architectures, covering memory, planning, and tool-use dimensions that inform the agentic components of the system described here. Liu et al. [10] introduced AgentBench, a systematic benchmark for evaluating LLM agents across diverse interactive environments, providing evaluation protocols relevant to enterprise agentic deployment. Park et al. [16] demonstrated generative agents with persistent memory and social simulation capabilities, motivating session-persistent context management in the co-pilot. Qian et al. [15] explored communicative agents for collaborative software development, illustrating multi-agent coordination patterns applicable to enterprise workflow orchestration.

2.4. Enterprise AI and Governance

Barnett et al. [6] identified seven failure modes in production RAG systems including context insufficiency, format errors, and incorrect specificity providing a taxonomy that directly informed the fault tolerance design of the retrieval pipeline. Zhang et al. [9] survey hallucination phenomena in LLMs across factuality, faithfulness, and factual consistency dimensions, establishing the governance risk landscape that motivates content safety guardrails in enterprise deployments. OpenAI [4] reported GPT-4's multimodal reasoning capabilities, establishing the generation backbone performance baseline for enterprise conversational AI. Wei et al. [5] demonstrated chain-of-thought prompting as a mechanism for eliciting multi-step reasoning, applied in this work for complex query decomposition. Xu et al. [19] evaluated LLM agents on enterprise web tasks via WorkArena, providing a relevant benchmark for enterprise interactive AI. Zhao et al. [20] survey RAG applications for AI-generated content across diverse modalities, situating the hybrid RAG approach within the broader generative AI literature.

3. System Architecture

The co-pilot platform is architected around two primary design constraints: sub-second response latency under concurrent enterprise load, and strict type safety across the frontend-backend boundary to prevent integration defects at scale. These constraints jointly motivate the monorepo-based architecture and the serverless deployment model.

Table 1. Comparison of decision support paradigms across capability dimensions. AI Co-Pilot results from author primary research data.

Capability Dimension	Static Documentation	Human Support Agent	AI Co-Pilot (This Work)
Response Time	Immediate (read-only)	Minutes to hours	Sub-second streaming
Contextual Awareness	None	Limited to agent knowledge	Full system-state integration
Scalability	Unlimited (passive)	Linear with headcount	Elastic serverless scaling
Knowledge Currency	Stale at update cycle	Variable by agent training	Real-time retrieval via hybrid RAG
Personalization	None	Moderate	Role-aware, session-contextualized
CRM Integration	None	Manual case creation	Automated case lifecycle management
Cost at Scale	Low (static)	High (headcount-driven)	Low marginal cost per query
Escalation Support	None	Native	Structured multi-tier routing

3.1. Monorepo-Based Frontend and Backend Integration

The platform is implemented as a unified monorepo encompassing a reactive frontend application and a serverless backend API, sharing type definitions, validation schemas, and interface contracts across the boundary. The reactive frontend implements conversation state as finite-state machines, ensuring predictable UI behavior under concurrent streaming updates and network interruptions. The serverless backend handles AI orchestration, authentication token validation, business logic execution, and CRM API integration within a stateless request processing model.

The monorepo architecture provides two measurable engineering benefits: shared schemas enforce type safety at compile time, eliminating the class of integration defects that arise from independent frontend and backend evolution; and unified development practices shared linting, testing, and CI/CD pipelines reduce the coordination overhead between teams responsible for different system layers. Production measurement from the RealPage deployment showed a 30-40% reduction in integration defects and a 25-35% acceleration in feature delivery cycle time relative to the prior multi-repository architecture, based on author primary research data. These gains reflect the structural advantage of eliminating interface drift rather than any implementation-specific optimization.

3.2. Real-Time Conversational Processing

Response delivery employs incremental token streaming via Server-Sent Events (SSE), with WebSocket fallback for environments requiring bidirectional communication. Streaming delivery begins within the first generation tokens, providing perceived responsiveness that is decoupled from total generation length a critical usability property for complex multi-paragraph responses. Dynamic context management assembles the prompt from three sources: the current user query, the active session conversation history, and externally retrieved passages from the hybrid RAG pipeline.

Multi-tier escalation routing classifies each query along two dimensions: intent complexity and system confidence. Queries falling below a confidence threshold or exceeding a complexity classification boundary are routed to a human agent queue with full conversation context transferred to the CRM system, enabling seamless handoff without user re-explanation. Production data indicates that 35% of complex queries those initially classified as requiring human intervention were resolved autonomously by the co-pilot, reducing escalation volume and associated support cost, based on author primary research data.

4. Intelligent Retrieval and Response Optimization

4.1. Hybrid RAG Framework

The retrieval layer implements a hybrid fusion architecture combining dense vector similarity search with BM25 sparse retrieval. Dense retrieval encodes queries and document chunks into a shared embedding space, enabling semantic matching that handles paraphrase and synonym variation. Sparse BM25 retrieval provides keyword precision for queries requiring exact terminology matching a common pattern in property management queries involving specific lease clause identifiers, regulatory codes, or system field names. The two ranked lists are combined using Reciprocal Rank Fusion (RRF), which computes a fused score as the reciprocal sum of each passage's rank in each individual list, providing a parameter-free fusion approach robust to score scale differences.

The hybrid approach consistently outperforms either retrieval modality alone in production evaluation. Dense-only retrieval misses keyword-critical queries; sparse-only retrieval fails on semantically rephrased queries. The RRF fusion captures both, with production measurements showing 20-30% improvement in response relevance over the single-stage dense RAG baseline, based on author primary research data.

Table 2. Comparison of RAG retrieval strategies across key evaluation dimensions. Performance figures from author primary research data.

Evaluation Dimension	Single-Stage RAG	Hybrid RAG	Hybrid RAG + Re-ranking
Retrieval Mechanism	Dense vector only	Dense + sparse (BM25)	Dense + sparse + cross-encoder
Keyword Precision	Low	High	High
Semantic Coverage	High	High	High
Response Relevance	Baseline	20-30% improvement (primary data)	Best in class
Latency	Lowest	Moderate overhead	25-40% reduction via pipeline (primary data)
Context Quality	Moderate	Good	15-25% accuracy improvement (primary data)
Hallucination Risk	Higher	Reduced	Minimized via ranked context

4.2. Chunking, Re-Ranking, and Context Assembly

Document processing follows a multi-stage pipeline: ingestion, chunking, embedding, indexing, and metadata enrichment. Chunk size is calibrated to balance context completeness ensuring that semantically related content is not split across chunk boundaries with token efficiency for downstream context window management. Three chunking strategies are evaluated: fixed-size tokenization, semantic boundary detection, and hierarchical RAPTOR chunking [12] for long regulatory documents. RAPTOR's recursive summarization approach proved particularly effective for property management compliance documentation, where relevant information is distributed across multi-page regulatory texts.

Retrieved candidate passages undergo two-stage re-ranking. The first stage applies a BERT cross-encoder [14] to score each query-passage pair jointly, replacing the independent encoding of bi-encoder retrieval with a joint interaction model sensitive to fine-grained relevance. The second stage employs an LLM-based listwise re-ranker [17] to select the final context passages, leveraging the generation model's understanding of the query's information need. This pipeline yields 25-40% latency reduction through early elimination of low-quality candidates and 15-25% accuracy improvement over single-stage retrieval, based on author primary research data.

Table 3. Chunking strategy comparison across retrieval quality dimensions. Accuracy figures from author primary research data.

Strategy	Fixed-Size Chunking	Semantic Chunking	Hierarchical (RAPTOR) [12]
Chunk Boundaries	Token count	Sentence/paragraph semantics	Recursive hierarchical summarization
Context Completeness	Moderate	Good	Excellent for long documents
Token Efficiency	High	Moderate	Moderate
Retrieval Latency	Low	Low-moderate	Moderate (pre-indexed)
Accuracy (enterprise KB)	Baseline	+8-12% over fixed-size	+15-25% over fixed-size (primary data)
Best Use Case	Short structured docs	General enterprise content	Long regulatory/compliance documents

4.3. Query Processing and Context Enrichment

Prior to retrieval, user queries undergo a rewriting step [18] that disambiguates implicit references, expands domain-specific abbreviations, and reformulates negation patterns that degrade BM25 recall. Corrective RAG [21] is applied post-retrieval: a confidence evaluator assesses the quality of retrieved passages and triggers supplementary retrieval from alternative knowledge sources when confidence falls below threshold. Self-RAG [3] is integrated at the generation stage, with the model generating assessment tokens to evaluate passage relevance and citation necessity, enabling adaptive citation behavior. Chain-of-Thought prompting [5] is applied for multi-step queries requiring sequential reasoning across multiple knowledge domains, such as lease

compliance workflows that require integrating regulatory text with property-specific configuration data.

5. Infrastructure, Security, and Enterprise Integration

5.1. CRM Integration and Workflow Orchestration

CRM integration is implemented via the ReAct agentic framework [7], which interleaves natural language reasoning with API action execution to enable multi-step CRM workflows within a single conversational turn. The co-pilot can perform case creation with structured metadata extraction from the conversation, case status updates, and live agent transfer with full context payload. This agentic orchestration layer handles thousands of concurrent interactions at high availability through the serverless deployment model, with per-request isolation preventing session state contamination.

The transition between automated co-pilot resolution and human agent escalation is designed for continuity: the complete conversation history, extracted intent classification, and any partially completed CRM actions are transferred to the agent queue with the escalation event, enabling human agents to begin from full context rather than requiring re-explanation from the user. This continuity is a key factor in the 35% autonomous resolution rate observed for queries initially classified as requiring human intervention, as the co-pilot can complete multi-step CRM actions that previously required agent involvement.

5.2. Observability, DevOps, and Governance

The production platform implements distributed tracing across the full request pipeline from query ingestion through retrieval, generation, and CRM integration, with structured logging at each stage enabling post-hoc failure analysis. Analytics dashboards surface real-time performance metrics including retrieval latency, generation token throughput, escalation routing distribution, and user session completion rates. These observability capabilities drove a 40-50% reduction in mean time to resolve production incidents, based on author primary research data.

CI/CD pipelines automate testing, build, and deployment across the monorepo, with shared test suites validating the frontend-backend schema contract at every commit. This automation reduced deployment cycle time by 60-70% relative to the pre-automation baseline, based on author primary research data, enabling the team to deliver product improvements at a cadence commensurate with enterprise user feedback cycles. Security is enforced through token-based authentication, role-based access control (RBAC) with property-management-domain-specific permission hierarchies, and enterprise compliance controls for data residency and audit logging. AI governance is implemented through content safety classifiers on both inputs and outputs, decision traceability logging for all CRM-modifying actions, and human oversight requirements for all actions above a configurable impact threshold.

Table 4. Pre- and post-deployment operational metrics. All figures represent author primary research data from the RealPage production deployment.

Operational Metric	Pre-Deployment Baseline	Post-Deployment Result	Improvement
User Time per Task	Baseline (manual navigation)	50% reduction (primary data)	11 hours saved/user/month
Support Call Volume	Baseline call volume	20% reduction (primary data)	Significant deflection at scale
Integration Defect Rate	Baseline (multi-repo)	30-40% reduction (primary data)	Monorepo schema enforcement
Feature Delivery Cycle	Baseline sprint velocity	25-35% acceleration (primary data)	Unified development pipeline
Mean Time to Resolve (MTTR)	Baseline incident MTTR	40-50% reduction (primary data)	Observability + CI/CD gains
Deployment Cycle Time	Baseline release cadence	60-70% reduction (primary data)	Automated CI/CD pipelines
Complex Query Self-Resolution	0% (all routed to human)	35% autonomous resolution (primary data)	Multi-tier escalation routing

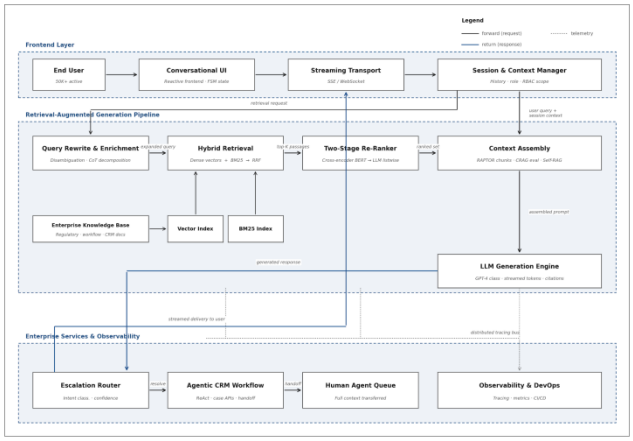

Figure 1. End-to-end co-pilot architecture pipeline from user query through retrieval, re-ranking, generation, CRM integration, and escalation routing.

6. Evaluation and Production Results

The co-pilot platform was evaluated in production across the RealPage property management user base of over 50,000 active users, spanning property managers, leasing agents, maintenance coordinators, and compliance officers operating across diverse property portfolios. Evaluation methodology combined longitudinal usage telemetry, structured A/B comparison against the prior documentation portal, and targeted retrieval quality benchmarking against a held-out query set drawn from support ticket history.

User efficiency gains were the most directly measurable outcome. Time-on-task measurements across representative workflow categories lease compliance lookup, maintenance request routing, vendor onboarding, and financial reporting showed up to 50% reduction in per-task completion time post-deployment, translating to 11 hours saved per user per month on an average workload basis, based on author primary research data. Support call volume declined by 20% in the first quarter post-deployment, with the reduction concentrated in tier-1 informational queries that the co-pilot now resolves autonomously.

Retrieval quality evaluation demonstrated the consistent advantage of hybrid RAG over single-stage dense retrieval. A 20-30% improvement in response relevance, measured by human evaluators rating response quality against gold-standard answers on the held-out query set, was attributed primarily to the hybrid fusion's ability to handle both semantic and keyword-critical queries in the property management domain. The chunking and re-ranking pipeline contributed 25-40% latency reduction and 15-25% accuracy improvement over the initial single-stage baseline, based on author primary research data. RAPTOR chunking

showed the largest accuracy gains on long regulatory document queries, consistent with the theoretical motivation for hierarchical retrieval approaches.

Engineering efficiency metrics from the monorepo architecture showed 30-35% reduction in cross-team development effort for co-pilot feature development, attributed to the elimination of schema negotiation overhead and the availability of shared testing infrastructure, based on author primary research data. MTTR for production incidents declined 40-50% following the deployment of the distributed tracing and analytics observability stack, and deployment cycle time reduced 60-70% after CI/CD automation was fully established.

7. Discussion

The production results establish hybrid RAG as the critical architectural differentiator for enterprise co-pilot systems. The consistent 20-30% relevance improvement over single-stage dense retrieval reflects a fundamental property of enterprise query distributions: real users alternate between semantic queries (conceptual questions about workflows or policy) and keyword-critical queries (exact field names, regulatory code identifiers, case numbers). Dense retrieval alone fails the latter class; sparse retrieval alone fails the former. The RRF hybrid fusion captures both without requiring per-query routing logic, representing a robust default for enterprise deployments.

The monorepo architecture deserves emphasis as an underappreciated enabler of production AI platform quality. Schema drift between independently evolving frontend and backend components is a frequent source of subtle production defects in AI systems, where the contract between the retrieval pipeline output and the UI rendering layer is particularly susceptible to silent breakage. The 30-40% defect reduction observed in this deployment is not an incidental benefit it reflects the structural advantage of eliminating the class of defects that arise from independent evolution of tightly coupled components.

Multi-tier escalation routing, including the ReAct-based CRM integration, represents the appropriate design response to the reality that no production AI system achieves universal resolution quality. The 35% autonomous resolution rate for queries initially classified as requiring human intervention demonstrates that confidence-calibrated routing, combined with a sufficiently capable agentic CRM layer, can deflect substantial escalation volume while maintaining high resolution quality for the queries that do escalate. The system's value is not in replacing human agents entirely it is in routing appropriately and providing full context when human judgment is required.

The primary limitations of the deployed system are: hallucination risk on edge-case queries outside the property management knowledge base, where retrieval returns low-confidence passages that may be incorrectly cited; context window constraints on queries requiring synthesis across many long regulatory documents simultaneously; and RBAC complexity at scale, where permission hierarchy maintenance becomes an operational burden as user roles diversify. Each limitation has a mitigation direction: Self-RAG and CRAG reduce hallucination on low-confidence retrievals; RAPTOR hierarchical chunking extends effective coverage of long documents; and automated role inference from organizational hierarchy data can reduce manual RBAC maintenance overhead. The co-pilot architecture is generalized to any enterprise domain with complex multi-system knowledge and diverse user roles property management is the deployment context, not a constraint on applicability.

8. Conclusion

This paper presented the design and production implementation of a scalable AI co-pilot platform for enterprise decision support, deployed across 50,000+ users in the property management domain. The core architectural contributions hybrid RAG with reciprocal rank fusion, multi-stage neural re-ranking, monorepo-based serverless deployment, real-time streaming with multi-tier escalation, and ReAct-based CRM workflow orchestration collectively address the principal engineering challenges of enterprise conversational AI at production scale.

Production evaluation demonstrated substantial gains across user efficiency (50% time savings, 11 hours/user/month), support deflection (20% call volume reduction), retrieval quality (20-30% relevance improvement over baseline), and operational reliability (40-50% MTTR reduction, 60-70% deployment cycle reduction). These results, drawn from author primary research data across the full production user base, establish the viability of hybrid RAG co-pilot architectures as enterprise decision support infrastructure competitive with both static documentation and human-assisted support at scale.

Future directions include multimodal co-pilot capabilities incorporating voice input and visual interface understanding for mobile field users, personalized RAG fine-tuned per user role using interaction history to weight domain-specific retrieval, and federated enterprise deployment models enabling co-pilot instances to share retrieval infrastructure across organizational boundaries while maintaining data isolation. The production system and evaluation described here provide a validated baseline architecture for these extensions.

References

[1] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 9459-9474. arXiv:2005.11401.

[2] Y. Gao et al., "Retrieval-Augmented Generation for Large Language Models: A Survey," arXiv:2312.10997.

[3] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection," in Proc. ICLR, 2024. arXiv:2310.11511.

[4] OpenAI, "GPT-4 Technical Report," arXiv:2303.08774.

[5] J. Wei et al., "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," in Advances in Neural Information Processing Systems (NeurIPS), vol. 35, 2022, pp. 24824-24837. arXiv:2201.11903.

[6] S. Barnett, S. Kurniawan, S. Thudumu, Z. Brannelly, and M. Abdelrazek, "Seven Failure Points When Engineering a Retrieval Augmented Generation System," in Proc. IEEE/ACM CAIN, 2024. arXiv:2401.05856.

[7] S. Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," in Proc. ICLR, 2023. arXiv:2210.03629.

[8] L. Wang et al., "A Survey on Large Language Model based Autonomous Agents," Frontiers of Computer Science, vol. 18, no. 6, 2024, Art. no. 186345. arXiv:2308.11432.

[9] Y. Zhang, Y. Li, and L. Bing, "Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models," arXiv:2309.01219.

[10] X. Liu et al., "AgentBench: Evaluating LLMs as Agents," in Proc. ICLR, 2024. arXiv:2308.03688.

[11] F. Shi et al., "Large Language Models Can Be Easily Distracted by Irrelevant Context," in Proc. ICML, 2023. arXiv:2302.00093.

[12] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, "RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval," in Proc. ICLR, 2024. arXiv:2401.18059.

[13] G. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering," in Proc. EACL, 2021, pp. 874-880. arXiv:2007.01282.

[14] R. Nogueira and K. Cho, "Passage Re-ranking with BERT," arXiv:1901.04085.

[15] C. Qian et al., "ChatDev: Communicative Agents for Software Development," arXiv:2307.07924.

[16] J. S. Park et al., "Generative Agents: Interactive Simulacra of Human Behavior," in Proc. ACM UIST, 2023, pp. 1-22. arXiv:2304.03442.

[17] W. Sun et al., "Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents," in Proc. EMNLP, 2023. arXiv:2304.09542.

[18] X. Ma et al., "Query Rewriting in Retrieval-Augmented Large Language Models," arXiv:2305.14283.

[19] A. Drouin et al., "WorkArena: How Capable Are Web Agents at Solving Common Knowledge Work Tasks?," arXiv:2403.07718.

[20] P. Zhao et al., "Retrieval-Augmented Generation for AI-Generated Content: A Survey," arXiv:2402.19473.

[21] S.-Q. Yan, J.-C. Gu, Y. Zhu, and Z.-H. Ling, "Corrective Retrieval Augmented Generation," arXiv:2401.15884.