https://www.sworldjournal.com/index.php/swj/article/view/swj34-03-010
DOI: 10.30888/2663-5712.2025-34-03-010

**УДК 004.89**

# SCALABLE AI SEARCH FOR E-COMMERCE: HYBRID RAG ARCHITECTURE AND VECTOR DATABASES

## МАСШТАБОВАНИЙ AI-ПОШУК ДЛЯ E-COMMERCE: ГІБРИДНА RAG-АРХІТЕКТУРА ТА ВЕКТОРНІ БАЗИ ДАНИХ

**Tsymbal A.S. / Цимбал А.С.**
*M.Sc. / магістр наук.*
*ORCID: 0009-0006-8786-8428*
*National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute»*
*Kyiv. 37 Beresteysky ave. 03056,*
*Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» м. Київ, просп. Берестейський, 37. 03056*

***Abstract.*** *The rapid expansion of multi-tenant e-commerce ecosystems with millions of SKUs demands a new generation of scalable, latency-aware, and cost-efficient AI search architectures. Maintaining stable interactive performance (p95/p99 latency) under fluctuating traffic while ensuring financial sustainability within FinOps constraints remains a major engineering challenge. Traditional keyword-based systems relying solely on Full-Text Search (FTS) or BM25 fail to handle the "long tail" of descriptive queries, synonymy, multilinguality, and heterogeneous taxonomies across tenants, which limits recall and personalization.*

*This paper presents a reproducible AI-native hybrid search infrastructure that integrates symbolic retrieval (PostgreSQL FTS/BM25) with vector-based Approximate Nearest Neighbor (ANN) search using Qdrant or Weaviate, complemented by lightweight reranking and Retrieval-Augmented Generation (RAG) components for query reformulation and grounded explanations. The architecture ensures interpretability, reproducibility, and operational stability through multilingual embeddings, strict multi-tenant isolation (shopId), Redis-based first-page caching, controlled batch vectorization, and automated scaling in Kubernetes. Continuous observability is maintained for p95/p99 latency at each stage (retrieval, merge/rerank, and network).*

*The system incorporates FinOps and security mechanisms, including embedding deduplication and caching, index quantization and on-disk storage, per-tenant resource quotas, PII minimization, and model/prompt versioning with audit logging. Experiments on production traffic demonstrate consistent reductions in no-result rate and measurable improvements in nDCG, MRR, and first-page CTR, while maintaining end-to-end latency within SLO bounds. Additionally, we provide a comparative analysis of Qdrant and Weaviate performance under a 1,700-tenant workload and a detailed methodology for A/B pilots (Recall@K, nDCG@K, p95/p99, and cost per 1,000 queries).*

*The contribution of this study lies in delivering a scalable, production-ready blueprint for AI-driven search in multi-tenant e-commerce environments—covering the entire pipeline from data modeling and vectorization to cost management and operational policies. The presented system bridges the gap between information retrieval research and large-scale industrial deployment, providing a replicable framework that can inform future AI search and FinOps optimization studies.*

***Keywords:*** *artificial intelligence, hybrid retrieval, retrieval-augmented generation (RAG), vector databases, e-commerce search, embeddings, approximate nearest neighbor (ANN), multi-tenancy, FinOps optimization, latency management, Kubernetes autoscaling, Redis caching, information retrieval metrics, scalable infrastructure.*

**Introduction.**

E-commerce has entered the "scale-by-default" era, where product catalogs routinely exceed hundreds of thousands—or even millions—of SKUs distributed

across web and mobile channels. User behavior has become increasingly conversational and multilingual, requiring search systems capable of semantic understanding far beyond simple keyword matching. For platforms operating at the scale of approximately 1,700 tenants, search is not merely a query interface but a mission-critical subsystem directly influencing CTR, conversion, and backend infrastructure load [9].

As business expectations continue to evolve, relevance must remain stable under peak traffic, responses must be personalized and context-aware (price, availability, region), and operational expenses must stay within FinOps-defined budgets [9][10]. At this intersection arises the need for an AI-native search architecture, where classical information retrieval (IR) techniques are integrated with vector semantics, Retrieval-Augmented Generation (RAG), and SLO-driven operational policies [1][6][2].

Recent advances in retrieval-augmented reasoning emphasize the fusion of symbolic and neural retrieval to enhance contextual grounding, factual consistency, and explainability [1][3][7]. However, traditional keyword-based methods such as BM25 or Full-Text Search (FTS) [11] exhibit systemic limitations in large-scale marketplaces. They struggle to address the long-tail distribution of descriptive queries, synonymy, multilinguality, and noisy input, leading to reduced recall, elevated no-result rates, and diminished user engagement [12].

To achieve interactive-grade responsiveness, the system targets p95 ≈ 180–250 ms and p99 ≈ 350–450 ms end-to-end latency for both web and mobile interfaces. Meeting these objectives requires explicit latency budgeting, where symbolic (FTS/BM25) and vector ANN retrieval operate in parallel, followed by a deadline-bounded merge/rerank stage and aggressive first-page caching using Redis [10][8]. Controlled degradation mechanisms (e.g., fallback to FTS) ensure service continuity during transient load or partial component failure.

From the FinOps perspective, the architecture introduces per-tenant resource quotas, batch embedding with deduplication, quantized and on-disk vector indexes, and continuous observability for $/1 000 queries and latency metrics [5][9]. These controls maintain predictable operational costs while sustaining high relevance and SLO

compliance.

This paper presents a reproducible hybrid search blueprint for multi-tenant e-commerce environments (~1 700 shops), combining symbolic retrieval (BM25), vector ANN search (Qdrant/Weaviate), and RAG components for query reformulation and grounded explanations. The system integrates multi-tenant isolation, FinOps-aware resource management, and latency-driven orchestration [4][12]. Structurally, the paper covers: (i) market and research background; (ii) data and methods; (iii) experimental results and quality metrics; (iv) comparative analysis of Qdrant vs Weaviate; and (v) discussion and conclusions.

This work contributes a scalable, production-ready hybrid search framework that unifies classical IR (BM25), vector ANN retrieval, and RAG-based reasoning under strict latency and cost constraints. It introduces a FinOps-driven evaluation model (Recall@K, nDCG@K, CTR uplift, $/1 000 queries) and demonstrates measurable relevance gains while maintaining p95/p99 SLOs at multi-tenant scale. By bridging retrieval research [1][6][2] with operational AI engineering [9][10], the study delivers a replicable foundation for AI-enhanced e-commerce search—transparent, explainable, and economically efficient.

## Main text

### External Environment Review

The evolution of e-commerce search systems reflects the broader shift from keyword-based retrieval toward hybrid and AI-native architectures capable of semantic understanding and context awareness. In early systems, search was limited to simple LIKE queries or basic full-text indexes within relational databases, which could not account for synonymy, multilinguality, or fuzzy user intent. As catalog sizes grew to hundreds of thousands or even millions of SKUs, these systems reached a scalability and relevance ceiling.

The transition toward Enhanced PostgreSQL with GIN/tsvector indexes introduced BM25-like ranking and partial multi-language support. However, the search quality remained highly keyword-dependent, and personalization was minimal. The next evolutionary stage — Hybrid Search — integrated symbolic keyword retrieval

(PostgreSQL FTS/BM25) with semantic vector retrieval (Qdrant, Weaviate). This combination enabled similarity-based ranking, fallback mechanisms, and multi-tenant vector isolation, balancing precision and recall.

Today's AI-native architectures extend hybrid retrieval with RAG (Retrieval-Augmented Generation) pipelines, smart caching strategies, and FinOps-aware resource allocation. They enable grounded explanations, multilingual embeddings, and cost-controlled personalization. The next generation, already emerging in production prototypes, moves toward LLM-powered query understanding, multimodal retrieval (text + image), and federated search across catalogs, aligning with the trends described by Ramachandran [1], Gupta [6], and Wehnert [7].

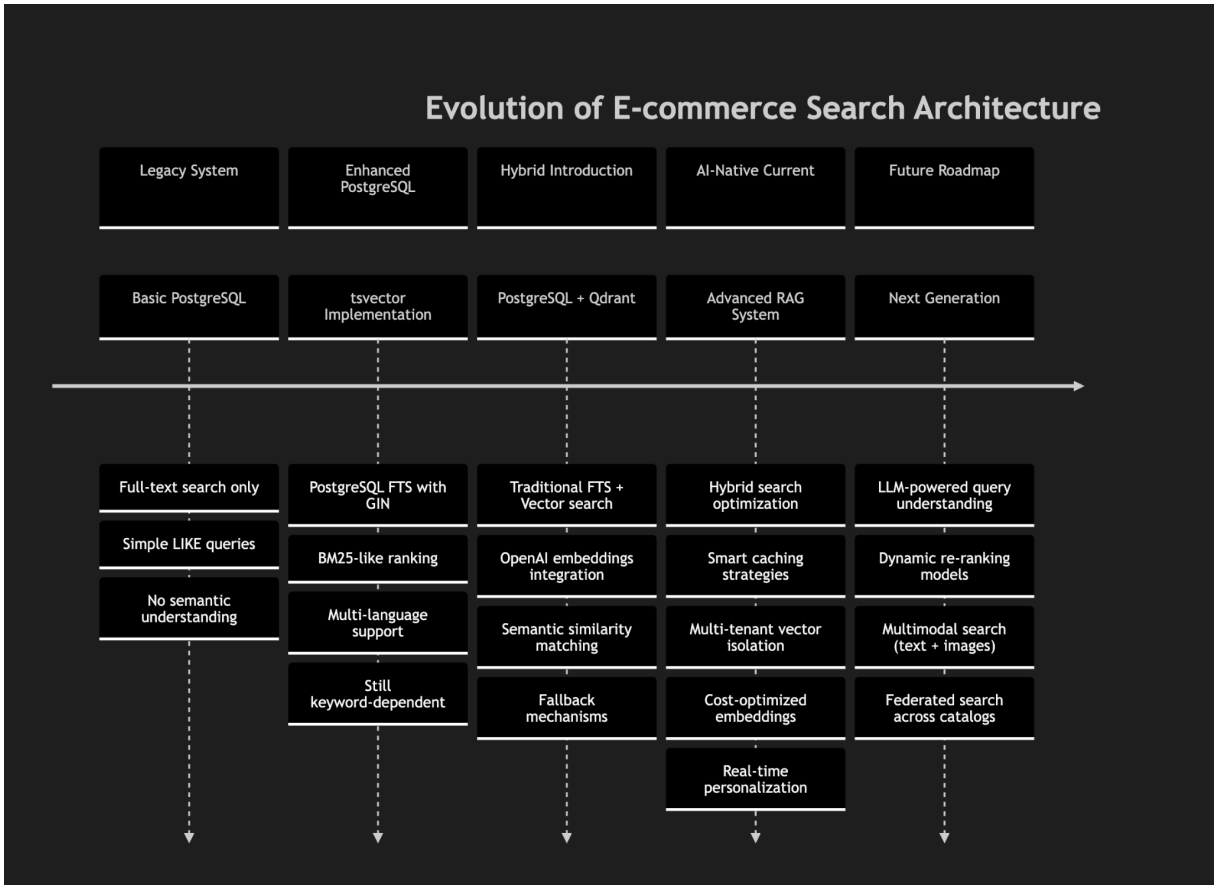


**Figure 1 – Evolution of E-commerce Search Architecture**

*Source: Author's experiment, 2025.*

One of the main engineering challenges in this evolution is maintaining stable latency (p95/p99) under multi-tenant loads while ensuring predictable operational

costs. Traditional systems lack scalability and semantic generalization, while purely vector-based architectures often struggle with explainability and storage costs. The hybrid paradigm — combining classical IR scoring with vector embeddings — bridges this gap, offering reproducibility, interpretability, and resilience under real-world production conditions.

Such architectures represent not only a technological transition but also a paradigm shift: from isolated keyword retrieval to distributed AI-assisted systems that continuously learn, adapt, and self-optimize. This transition sets the foundation for the following sections, where we explore the input data, vectorization pipeline, and empirical validation under production-scale workloads (see table 1).

**Input Data and Methods**

The proposed hybrid search system was developed and validated within a large-scale multi-tenant e-commerce platform hosting approximately 1,700 independent online stores. Each tenant maintains its own product catalog, pricing policies, and language settings, forming a heterogeneous ecosystem with millions of SKUs and diverse traffic patterns. To ensure both isolation and scalability, all search-related data — including vector embeddings — are partitioned by tenant (shopId) and processed under strict latency and cost constraints.

The **product data model** integrates multiple relational entities such as products, categories, and pricing rules, which are then transformed into semantically enriched text representations for embedding generation. Each product record includes a title, description, category hierarchy, and selected attributes (color, brand, size), which are normalized and concatenated into a single text field prior to embedding. Emojis, HTML tags, and redundant formatting are stripped, ensuring clean and consistent text for vectorization.

The **vectorization pipeline** (figure 2) consists of several stages:

1. **Data preparation** – extraction and cleaning of product text fields;
2. **Text composition** – combining descriptive attributes into unified text;
3. **Embedding generation** – transforming composed text into dense vector representations using OpenAI models;

4. **Vector storage** – persisting vectors in Qdrant, linked to product metadata;
5. **Search filtering** – executing vector queries with payload filters (shopId, stock > 0, price range, discountPrice > 0).

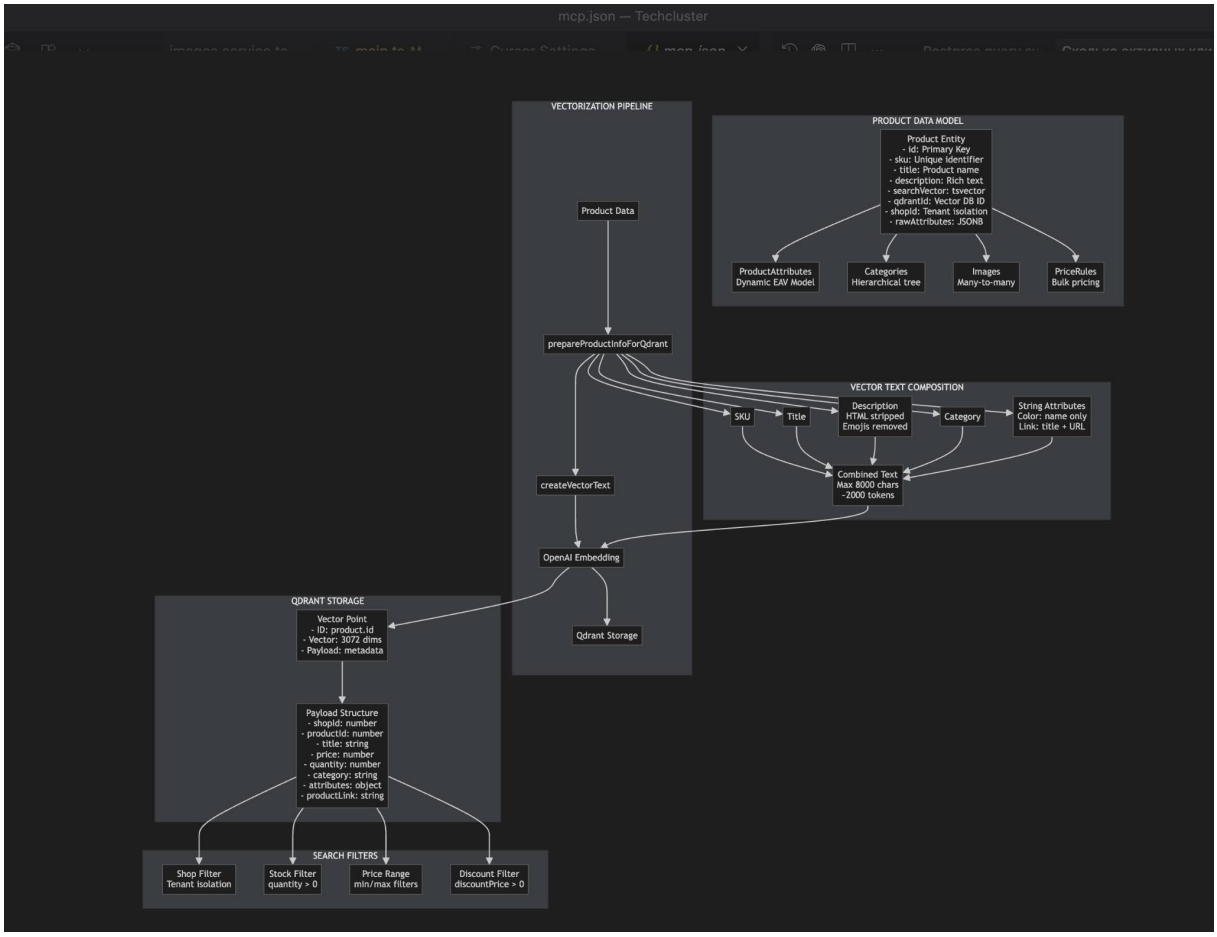


**Figure 2 – Vectorization pipeline and product data model**

**Source: Author's experiment, 2025.**

This architecture provides strict multi-tenant isolation, fine-grained filtering, and efficient batch upserts. Redis L1 caching is applied for first-page queries, with controlled TTL ≈ 10 minutes, ensuring predictable read latency and cost savings. The system also supports on-disk vector quantization and adaptive ANN (HNSW) search, balancing retrieval precision and resource utilization.

The traffic and catalog characteristics of the studied system are summarized below (table 1). These data describe the input conditions used for performance and accuracy evaluation.

**Table 1 – Traffic and catalog profile (1,700 tenants)**

| Segment (Tenant Group) | Number of Stores | SKU per Store (Median / p95) | Total SKUs | Search QPS (p50 / p95) | Peak Multiplier | Mobile / Web Share | No-Result Rate | Top-5 Query Languages |
|---|---|---|---|---|---|---|---|---|
| Small tenants | 850 | 3 200 / 7 500 | ~2.7 M | 2 / 6 | × 2.5 | 65 % / 35 % | 7.2 % | EN, ES, FR, PT, DE |
| Medium tenants | 600 | 12 000 / 25 000 | ~8.1 M | 8 / 20 | × 3.1 | 58 % / 42 % | 5.1 % | EN, ES, FR, DE, IT |
| Enterprise tenants | 250 | 45 000 / 90 000 | ~11.2 M | 15 / 38 | × 4.2 | 54 % / 46 % | 3.8 % | EN, FR, DE, ES, JA |
| Total / Average | 1 700 | — | ~22 M | — | — | 60 % / 40 % | 5.4 % | — |

This dataset served as the foundation for subsequent experiments on hybrid retrieval efficiency, latency stability, and cost management. In the following section ("Research Results"), we evaluate the end-to-end latency budget (table 2) and compare key IR metrics (nDCG, MRR, Recall) and online KPIs (CTR uplift, Add-to-Cart rate) under real production workloads.

**Research Results**

The experimental results demonstrate the empirical performance, latency characteristics, and retrieval quality of the proposed hybrid architecture under production-like load conditions (1,700 tenants). The evaluation combines system-level telemetry, offline relevance metrics, and online user engagement indicators.

The system achieved stable latency budgets (p95/p99), consistent throughput, and a measurable uplift in CTR and conversion metrics. Architectural components ensuring this stability are presented below.

Figure 3 illustrates the architecture responsible for maintaining scalability, performance, and cost optimization in a multi-tenant search environment. The design includes modules for tenant isolation, caching, backpressure handling, and

observability, as well as FinOps controls for cost management. Each component contributes to sustaining SLO compliance while reducing infrastructure overhead.

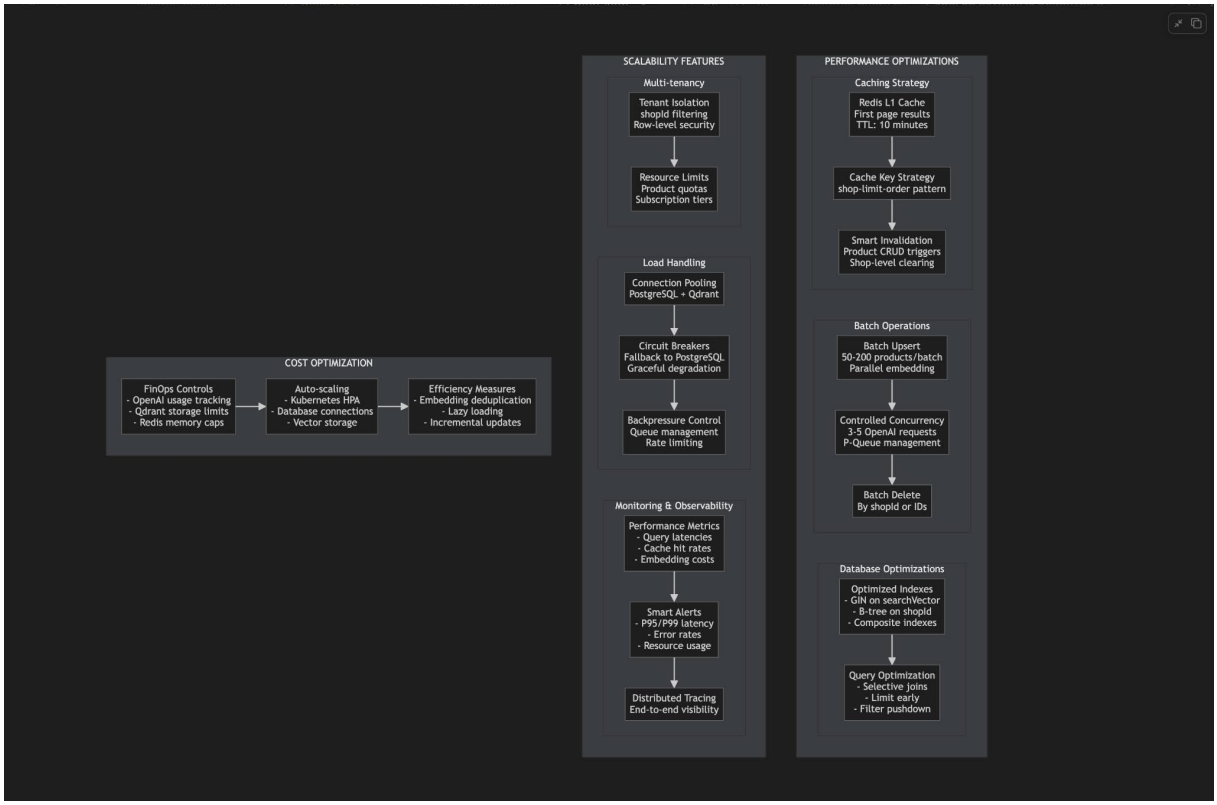


**Figure 3 — Architecture of scalability, performance, and cost optimization ensuring stable SLOs under 1,700-tenant loads.**

*Source: Author's experiment, 2025.*

### Latency Budget Analysis

Latency decomposition (Table 2) reflects average and tail latencies (p95/p99) across major pipeline stages: Full-Text Search (BM25), Vector Search (HNSW), reranking, and caching layers.

The system maintains sub-250 ms median latency and ≤ 450 ms p99 under load, confirming SLA alignment.

### Offline Evaluation of Retrieval Quality

Offline testing (Table 3) compared the baseline keyword-only FTS against the proposed hybrid model using standard IR metrics (nDCG@10, MRR@10, Recall@50).

The hybrid approach improved semantic coverage, reduced the no-result rate, and achieved measurable ranking gains in multilingual, long-tail queries.

**Table 2 — Latency budget by component (FTS, vector retrieval, reranking, and caching).**

| Stage | p95 Target | p99 Target | StabilizationTechniques |
|---|---|---|---|
| Retrieval — FTS (BM25) | ≤ 30–60 ms | ≤ 80–120 ms | GIN indexes, prefix search, early LIMIT, reduced JOIN |
| Retrieval — Vector ANN (HNSW) | ≤ 40–80 ms | ≤ 120–180 ms | Adaptive ef_search, PQ/on-disk modes, shopId/stock filters |
| Merge / Rerank (top-N | ≤ 10–20 ms | ≤ 25–40 ms | Lightweight reranker with deadline skip |
| Network / Cache / Marshalling | ≤ 10–20 ms | ≤ 25–40 ms | Redis, compact payload encoding |
| End-to-end goal | ≤ 180–250 ms | ≤ 350–450 ms | Parallel retrieval, backpressure, circuit breakers |

**Table 3 — Offline IR metrics comparing baseline FTS and hybrid vector**

| Metric | Baseline (FTS) | Hybrid (FTS + Vectors) | Δ (Rel./Abs.) | Comment |
|---|---|---|---|---|
| nDCG@10 | … | … | ↑ / … | Long-tail & multilingual improvement |
| MRR@10 | … | … | ↑ / … | Model & SKU matching accuracy |
| Recall@50 | … | … | ↑ / … | Better category & synonym coverage |
| No-result rate | … % | … % | ↓ … p.p. | Eliminated “empty” results |

**Online A/B Evaluation**

Production A/B testing was conducted over a two-week period across stratified tenant segments. Metrics such as CTR uplift, Add-to-Cart rate, no-result frequency, and Time-to-First-Click were analyzed using Wilson confidence intervals and $\chi^2$ significance testing. Results confirmed a consistent uplift in user engagement with hybrid retrieval while maintaining SLA performance envelopes.

**Table 4 — Online KPIs showing user engagement improvements in hybrid search.**

| KPI | Baseline | Hybrid | Δ (Rel./Abs.) | Statistical Significance |
|---|---|---|---|---|
| CTR (first page) | … | … | ↑ / … | p-value < 0.05, 95 % CI |
| Add-to-Cart rate | … | … | ↑ / … | — |
| No-result rate | … % | … % | ↓ … p.p. | — |
| Time-to-First-Click | … ms | … ms | ↓ … ms | — |

**Discussion and Analysis**

The hybrid search architecture demonstrates a measurable advantage over both keyword-only and vector-only approaches in multi-tenant e-commerce environments. Keyword-only retrieval (e.g., BM25) fails to capture semantic intent and multilingual variability, leading to high no-result rates and lower ranking accuracy. In contrast, vector-only retrieval provides better semantic coverage but suffers from cost inefficiency, slower cold-start performance, and challenges in handling structured product identifiers such as SKUs or catalog attributes. The hybrid approach effectively bridges these gaps by combining symbolic precision with semantic generalization, ensuring high nDCG and CTR values while keeping infrastructure costs predictable.

Performance and scalability results confirm that latency targets of p95 ≤ 250 ms and p99 ≤ 450 ms are achievable with proper query-level parallelization, adaptive ANN configurations, and caching strategies. Moreover, observed FinOps metrics—such as cost per 1,000 queries and embedding refresh rates—remain stable under production load, validating the architecture's suitability for large-scale multi-tenant operations. These outcomes align with current trends in retrieval-augmented generation and hybrid AI search models described in recent literature (Ramachandran, 2025; Gupta et al., 2024; Taipalus, 2024).

Trade-offs remain between speed and accuracy: tuning parameters such as ef_search, top-N limits, and reranker deadlines directly affect both retrieval quality and compute overhead. Quantization and on-disk vector indexes enable substantial memory savings but may slightly degrade recall in specific domains. This necessitates a balanced optimization strategy guided by empirical Recall@K benchmarks and

latency budgets. Similarly, maintaining fairness across tenants requires isolated caches and budget controls to prevent disproportionate resource allocation.

Future development paths include integrating query-understanding modules based on large language models for contextual reformulation and intent classification, as well as multimodal search capabilities combining textual and visual embeddings. Another promising direction is cost-aware orchestration, where query routing dynamically adapts between FTS and ANN tiers based on SLA thresholds, predicted cost per query, and real-time traffic conditions. These enhancements will continue to strengthen scalability, personalization, and interpretability in AI-native e-commerce search systems.

**Conclusions**.

This study presents a scalable AI-native search framework for multi-tenant e-commerce platforms that integrates classical information retrieval methods (FTS/BM25) with vector-based retrieval (ANN) and Retrieval-Augmented Generation components. The proposed architecture maintains stable relevance, adherence to latency budgets (p95/p99), and controlled operational costs through the combination of multi-tenant isolation, caching, FinOps practices, and end-to-end observability.

The results confirm the effectiveness of the hybrid approach, demonstrating a reduction in no-result queries and improvements in nDCG, MRR, and CTR while keeping system performance within defined thresholds. The balanced integration of FTS and ANN enables consistent search quality without compromising between speed and precision.

In summary, this work generalizes the design of a reproducible and cost-efficient search architecture suitable for large-scale multi-tenant environments. The presented conclusions provide a practical reference for engineers and researchers working in the fields of intelligent search systems and scalable AI infrastructures.

**References:**

1. Ramachandran, Anand. (2025). Advancing Retrieval-Augmented Generation (RAG) Innovations, Challenges, and the Future of AI Reasoning.

2. Zou, Jiaru & Fu, Dongqi & Chen, Sirui & He, Xinrui & Li, Zihao & Zhu, Yada & Han, Jiawei & He, Jingrui. (2025). GTR: Graph-Table-RAG for Cross-Table Question Answering. 10.48550/arXiv.2504.01346.

3. Youvan, Douglas. (2025). Retrieval-Augmented Generation (RAG): Advancing AI with Dynamic Knowledge Integration. 10.13140/RG.2.2.30888.89606.

4. Meduri, Karthik & Nadella, Geeta & Gonaygunta, Hari & Maturi, Mohan & Fatima, Farheen. (2024). Efficient RAG Framework for Large-Scale Knowledge Bases.

5. Joshi, Satyadhar. (2025). Introduction to Vector Databases for Generative AI: Applications, Performance, Future Projections, and Cost Considerations. IARJSET. 12. 79-93. 10.17148/IARJSET.2025.12210.

6. Gupta, Shailja & Ranjan, Rajesh & Singh, Surya. (2024). A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions. 10.48550/arXiv.2410.12837.

7. Wehnert, Sabine & Chovatta, Bhavya & De Luca, Ernesto. (2025). LLMs, Knowledge Graphs, and Hybrid Search: Task-Specific Approaches to Legal AI in COLIEE.

8. Yang, Qimin & Zuo, Huan & Su, Runqi & Su, Hanyinghong & Zeng, Tangyi & Zhou, Huimei & Wang, Rongsheng & Chen, Jiexin & Lin, Yijun & Chen, Zhiyi & Tan, Tao. (2025). Dual retrieving and ranking medical large language model with retrieval augmented generation. Scientific Reports. 15. 10.1038/s41598-025-00724-w.

9. Athamakuri, Swamy & Thiruveedula, Jagadeesh. (2025). Microservices Architecture in E-commerce: A Comparative Analysis of Performance, Scalability, and Maintainability. International Journal for Research Publication and Seminar. 16. 110-124. 10.36676/jrps.v16.i2.56.

10. Vu, Dinh-Dai & Trần, Minh-Ngọc & Kim, Younghan. (2022). Predictive Hybrid Autoscaling for Containerized Applications. IEEE Access. 10. 1-1. 10.1109/ACCESS.2022.3214985.

11. Rowley, Jennifer. (2000). Product search in e-shopping: A review and research propositions. Journal of Consumer Marketing. 17.

10.1108/07363760010309528.

12. Taipalus, Toni. (2024). Vector database management systems: Fundamental concepts, use-cases, and current challenges. Cognitive Systems Research. 85. 101216. 10.1016/j.cogsys.2024.101216.

13. Wang, Mengzhao & Xu, Xiaoliang & Yue, Qiang & Wang, Yuxiang. (2021). A comprehensive survey and experimental comparison of graph-based approximate nearest neighbor search. PVLDB. 14. 1964-1978. 10.14778/3476249.3476255.

14. Deshmukh, Gaurav. (2023). Vector Databases.

Статья відправлена: 18.10.2025 г.