A PRIVACY FOCUSED DISTRIBUTED RAG ARCHITECTURE USING SMALL LANGUAGE MODELS FOR HIGHER EDUCATION

Žarko Bogićević[1]*,
[0000-0002-0831-1300]

Marjan Milošević[2]
[0000-0003-4730-1292]

[1]x1F Serbia,
Beograd, Serbia

[2]University of Kragujevac,
Faculty of Technical Sciences Čačak,
Čačak, Serbia

Abstract:

This paper presents a privacy-focused teaching assistant system based on locally deployed Small Language Models (SLMs) combined with Retrieval Augmented Generation (RAG) for higher education environments. Existing AI-powered educational tools predominantly rely on cloud-based large language models, introducing challenges related to data privacy, regulatory compliance, and infrastructure cost. These limitations are particularly significant in university settings, where sensitive institutional data must remain protected.

The proposed system is designed as a distributed, hardware-aware architecture operating entirely within institutional infrastructure. It supports deployment across diverse environments, including CPU-only systems, integrated GPUs, and discrete GPUs, enabling institutions to leverage existing hardware without requiring specialized AI infrastructure. The architecture integrates modular services for request routing, authentication, orchestration, and dynamic worker node management, alongside a vector database for semantic retrieval of educational content.

Experimental observations demonstrate that retrieval mechanisms are essential for accessing institution specific knowledge, while also highlighting the importance of careful system design when integrating retrieval with small-scale models.

The proposed approach provides a scalable, cost-efficient, and privacy-focused solution for deploying AI assistants in higher education.

Keywords:

Small Language Models, Retrieval Augmented Generation, Distributed Systems, Educational AI, Privacy.

INTRODUCTION

The significant advancement of AI has transformed how knowledge is processed and utilized across different domains. In education, AI-driven systems offer opportunities for personalized learning, automated feedback, and intelligent tutoring [1], [2]. However, most solutions rely on cloud-based large language models, introducing challenges that are related to data privacy, infrastructure cost, and overall institutional control.

Higher education institutions usually operate within strict regulatory frameworks, such as the EU AI Act [3], which have constraints on data processing and storage. Cloud-based systems often require transferring sensitive institutional data to external providers, raising concerns regarding compliance and data sovereignty.

Correspondence:
Žarko Bogićević

e-mail:
zarko1993@hotmail.com

CC BY

DOI: 10.15308/Sinteza-2026-254-259

At the same time, universities need to manage large volumes of diverse educational content, where traditional search methods are often insufficient for retrieving relevant information from these datasets.

To address these challenges, this paper proposes a privacy-focused teaching assistant architecture based on locally deployed Small Language Models (SLMs) combined with Retrieval Augmented Generation (RAG) [4], [5]. The system is designed to operate within institutional infrastructure, enabling secure and efficient access to educational content.

The main contributions of this paper are:

- A distributed, privacy-focused RAG architecture for higher education.
- A hardware-aware deployment model supporting heterogeneous environments.
- A practical implementation using open-source technologies.
- Empirical insights into retrieval-based augmentation with small-scale models.

2. RELATED WORK

Retrieval Augmented Generation (RAG) has become a foundational approach for improving factual consistency in language model outputs by grounding responses in external knowledge sources [4], [5].

In education, AI systems have demonstrated a strong potential in supporting learning and assessment processes [1], [2]. However, most implementations rely on cloud-based infrastructure, raising concerns regarding privacy and accessibility.

Recent studies confirm the growing adoption of RAG-based systems in educational environments. Surveys by Li et al. [6] and Swacha and Gracel [7] identify retrieval augmentation as a dominant architecture for educational chatbots. Several deployed systems, such as Prof. Leodar [8] and OwlMentor [9], demonstrate the effectiveness of retrieval in accessing domain-specific knowledge. Additional studies show that students frequently rely on such systems to obtain information not explicitly covered in lectures [10].

From a retrieval perspective, Sarmiento and Lauría [11] highlight trade-offs between retrieval quality and latency, while Pasquarelli et al. [12] compare RAG systems with traditional keyword-based approaches.

Local deployment of small language models remains relatively underexplored. Shanthakumar et al. [13] demonstrate the feasibility of running quantized models locally, supporting the motivation for privacy-focused architectures.

3. SYSTEM ARCHITECTURE

The proposed system is designed as a distributed architecture operating entirely within a local institutional network.

The high-level system architecture is illustrated in Figure 1.

The architecture consists of the following components:

- Gateway Service – an API Gateway that provides dynamic routing, load balancing, and security features

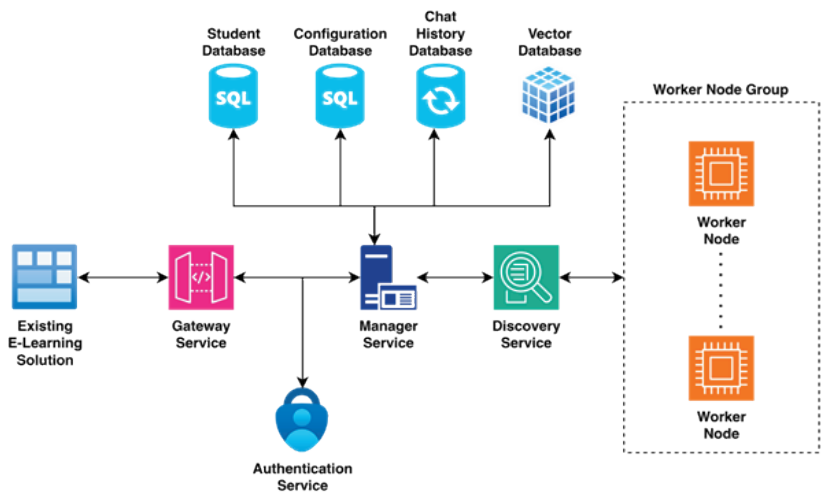

Figure 1. High level System Architecture Diagram

- Authentication Service – used for identity and access management, providing JWT, OAuth2, and Single Sign-On (SSO).
- Manager Service – acts as a central orchestration component of the system, responsible for request routing and resource allocation, it dynamically routes requests based on hardware tier, current load and model availability. In addition to that, it maintains configuration related data, routing policies and constraints. It interacts with the discovery service to identify active worker nodes and adapt routing decisions.
- Discovery Service – utilized to significantly simplify the registration and discovery processes within the microservice architecture
- Worker Nodes – the local hosting of SLMs is managed here using an efficient and lightweight environment for running language models locally across diverse hardware and operating systems. A custom wrapper is integrated to handle system specific tasks, such as load reporting, request prioritization, and dynamic registration with the Discovery Service.
- Databases (configuration, chat history, vector storage, student information) – the system relies on multiple databases for different aspects of the operation. The configuration metadata is stored in the relational databases that contain system settings, model metadata and other policies. Chat history is maintained to support conversational continuity and enable future personalization features. A vector database is used to store embeddings of educational materials, enabling semantic retrieval within the RAG pipeline. Additionally, student-related information can be stored to support a personalized learning experience while adhering to strict privacy and access control requirements.

Worker nodes dynamically register with the system and process requests based on their hardware capabilities. This enables efficient resource utilization across diverse environments.

A central component is the vector database, which stores embeddings of educational materials and enables semantic retrieval.

The modular design of the system enables flexible deployment strategies depending on institutional requirements. Components can be scaled independently, allowing, for example, the Worker Nodes to be replicated to support increased load without affecting the control plane services. This separation also facilitates easier maintenance and upgrades, as individual services can be updated or replaced without disrupting the entire system. Furthermore, the architecture supports incremental adoption, where institutions can initially deploy a subset of components and expand functionality over time.

3.1. IMPLEMENTATION OVERVIEW

The system leverages widely adopted open-source technologies to ensure robustness and ease of deployment.

The Gateway Service is implemented using Zuul [14], while service discovery is handled through Eureka [15]. PostgreSQL [16] is used as the primary relational database, extended with pgvector [17] for embedding storage. For scalable similarity search, vector databases such as Milvus [18], Chroma [19], and FAISS [20] are supported. Authentication and authorization are managed through Keycloak [21], while local model execution is handled through the Ollama framework [22], enabling efficient deployment of quantized SLMs.

The use of open-source technologies ensures transparency, extensibility, and vendor independence. This is particularly important in educational environments, where long-term sustainability and cost efficiency are critical factors. Additionally, leveraging widely adopted tools reduces the barrier to adoption, as institutional IT teams are often already familiar with these technologies.

4. EXPERIMENT RESULTS

To assess deployment feasibility across diverse institutional environments, the system was evaluated on three hardware configurations using the same physical machine (AMD Ryzen 7 260, 32GB DDR5), varying only the inference accelerator:

- Tier 1 (CPU only): All CPU threads, GPU disabled
- Tier 2 (iGPU): AMD Radeon 780M via ROCm
- Tier 3 (dGPU): NVIDIA RTX 5060 8GB via CUDA

Using the same machine across all tiers eliminates variables such as differences in CPU architecture, memory speed, or cache hierarchy, providing a controlled comparison where only the inference accelerator changes.

End-to-end response latency was measured across 60 queries per model in RAG mode. Results, as seen in Table 1, represent the median latency across the full query set.

These results highlight the importance of aligning model size with available hardware resources. While smaller models provide acceptable latency on CPU only systems, larger models require hardware acceleration to achieve practical response times. This trade-off is particularly relevant in educational settings, where infrastructure constraints may vary significantly across institutions.

The dGPU tier delivers 4.5–13.8× lower latency than CPU-only depending on the model. The iGPU tier consistently sits between the two, achieving 1.5–3.1× improvement over CPU without requiring dedicated GPU hardware. Notably, the speedup from CPU to dGPU is largest for llama3.2 (13.8×), which benefits most from GPU memory bandwidth due to its architecture, while phi4-mini shows a more modest gain, likely due to architectural characteristics that limit GPU parallelism for shorter responses.

Server-side timing instrumentation breaks each request into its constituent phases. Table 2. reports median retrieval and generation times for RAG mode across tiers.

Retrieval latency (vector similarity search over pgvector) is consistently 28 - 39 ms across all three tiers and is effectively hardware-independent; it is bounded by database I/O, not the inference accelerator, as shown in Table 2. Generation accounts for over 99% of total response time in every configuration. This confirms that the vector retrieval component introduces negligible overhead and that hardware investment directly translates into improved user-facing response time through faster generation.

To evaluate behaviour under concurrent load, requests were sent at 1, 5, and 10 simultaneous users in RAG mode. This test was conducted on the CPU (Tier 1) and dGPU (Tier 3) configurations. The iGPU tier was not included in the concurrency test.

The CPU tier degrades significantly under concurrent load. Even at 5 simultaneous users, error rates exceed 23% for the smallest model and 87% for gemma3:4b, which means that requests time out before generation completes. At 10 concurrent users, the 4B model fails 90% of the time, making CPU only deployment effectively unsuitable for multi-user scenarios with larger models.

Table 1. Median RAG response latency by hardware tier (seconds)

Model	CPU (Tier 1)	iGPU (Tier 2)	dGPU (Tier 3)	dGPU speedup vs CPU
gemma3:1b	17.0 s	11.1 s	3.8 s	4.5×
llama3.2	45.6 s	14.6 s	3.3 s	13.8×
phi4-mini	54.5 s	20.4 s	10.2 s	5.3×
gemma3:4b	51.4 s	25.1 s	6.1 s	8.4×

Table 2. Retrieval vs. generation latency breakdown (median, ms)

Model	CPU (Tier 1)	iGPU (Tier 2)	dGPU (Tier 3)	dGPU speedup vs CPU
CPU	gemma3:1b	37 ms	16,796 ms	99.8%
CPU	llama3.2	36 ms	45,442 ms	99.9%
CPU	gemma3:4b	36 ms	51,202 ms	99.9%
iGPU	gemma3:1b	28 ms	11,084 ms	99.7%
iGPU	llama3.2	29 ms	14,599 ms	99.8%
iGPU	gemma3:4b	39 ms	25,002 ms	99.8%
dGPU	gemma3:1b	32 ms	3,701 ms	99.1%
dGPU	llama3.2	31 ms	3,154 ms	99.0%
dGPU	gemma3:4b	32 ms	6,030 ms	99.5%

Table 3. RAG concurrency: p50 latency and error rate

Model	Concurrency	CPU p50	CPU error rate	dGPU p50	dGPU error rate
gemma3:1b	1	29.8 s	0%	6.6 s	0%
gemma3:1b	5	105.0 s	23%	20.6 s	0%
gemma3:1b	10	86.9 s	37%	29.1 s	0%
gemma3:4b	1	109.3 s	20%	15.0 s	0%
gemma3:4b	5	61.6 s	87%	39.4 s	0%
gemma3:4b	10	92.6 s	90%	54.1 s	0%

The dGPU tier remains fully stable across all concurrency levels tested, with 0% error rates for both models at all concurrency levels. Latency scales sub linearly from c = 1 to c = 10 (e.g. gemma3:1b goes from 6.6 s to 29.1 s over 10 times increase in concurrency), indicating that the GPU scheduler queues requests efficiently rather than failing them.

These findings emphasize that system scalability is primarily constrained by the inference component rather than the retrieval pipeline. As concurrency increases, efficient request scheduling and hardware acceleration become critical factors in maintaining system responsiveness. This further supports the need for distributed architectures that can dynamically allocate workloads across available resources.

5. CONCLUSION

This paper presents a privacy-focused distributed RAG architecture based on locally deployed small language models for higher education.

The proposed system enables secure, scalable, and cost-efficient deployment of AI assistants using existing institutional infrastructure. The findings indicate implications for institutional deployment scenarios, as the choice of hardware tier is not merely a performance trade-off, it determines whether multi-user deployment is viable at all. CPU only is suitable for single user traffic deployments but breaks under concurrent load with larger models. The iGPU tier provides a meaningful improvement in a single user latency without dedicated GPU infrastructure, making it a practical baseline for institution-wide deployment. The dGPU is necessary for reliable multi-user serving, a single modest GPU (RTX 5060 with 8GB VRAM) handles 10 concurrent RAG requests with zero errors and remains stable at 5 concurrent users for 4B model.

Critically, the vector retrieval layer adds no meaningful latency overhead regardless of tier, confirming that the RAG architecture itself is not the performance bottleneck.

The presented results demonstrate that practical deployment of AI assistants in higher education is feasible without reliance on cloud-based infrastructure. This opens opportunities for institutions to adopt AI technologies while maintaining full control over their data and systems.

Future work will focus on improving retrieval strategies, model routing, and personalization mechanisms.

6. ACKNOWLEDGEMENTS

This study was partly supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia, and these results are parts of the Grant No. 451-03-34/2026-03/ 200132, with University of Kragujevac - Faculty of Technical Sciences Čačak.

REFERENCES

[1] E. Micheni, J. Machii and J. Murumba, "The role of artificial intelligence in education," *Open Journal for Information Technology*, vol. 7, no. 1, pp. 43-54, 2024, doi: 10.32591/coas.ojit.0701.04043m.

[2] P. Lameras and S. Arnab, "Power to the Teachers: An Exploratory Review on Artificial Intelligence in Education," *Information*, vol. 13, no. 1, p. 14, 2021, doi: 10.3390/info13010014.

[3] European Union, "Artificial Intelligence Act," 2024. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng.

[4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, T. Rocktäschel, S. Riedel and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver BC Canada, 2020, pp. 9459-9474.

[5] D. Vujić, A. Njeguš and N. Bačanin Džakula, "Enhancing retrieval-augmented generation with graph-based retrieval and generative modeling," in *Proceedings of Sinteza 2025,* Belgrade, 2025, pp. 3-9, doi: 10.15308/Sinteza-2025-3-9.

[6] Z. Li, Z. Wang, W. Wang, K. Hung, H. Xie and F. L. Wang, "Retrieval-augmented generation for educational application: A systematic survey," *Computers and Education: Artificial Intelligence*, vol. 8, p. 100417, 2025, doi: 10.1016/j.caeai.2025.100417.

[7] J. Swacha and M. Gracel, "Retrieval-augmented generation (RAG) chatbots for education: A survey of applications," *Applied Sciences*, vol. 15, p. 4234, 2025, doi: 10.3390/app15084234.

[8] M. Thway, J. Recatala-Gomez, F. S. Lim, K. Hippalgaonkar and L. W. T. Ng, "Battling Botpoop Using GenAI for Higher Education: A Study of a Retrieval-Augmented Generation Chatbot's Impact on Learning," *arXiv preprint arXiv*:2406.07796, 2024.

[9] D. Hüs, S. Malone and R. Brünken, "Exploring generative AI in higher education: A RAG system to enhance student engagement with scientific literature," *Frontiers in Psychology*, vol. 15, p. 1474892, 2024, doi: 10.3389/fpsyg.2024.1474892.

[10] G. Lang and T. Gürpinar, "Use of a retrieval-augmented generation (RAG) chatbot in an online R programming course," in *Proceedings of the ISCAP Conference*, 2024.

[11] C. Sarmiento and E. J. M. Lauría, "Investigating flavors of RAG for applications in college chatbots," in *Proceedings of the 17th International Conference on Computer Supported Education*, 2025, pp. 421-428, doi: 10.5220/0013468200003932.

[12] L. Pasquarelli, "Evaluating the use of retrieval-augmented generation for enhancing online courses," Aalto University, Espoo, 2025.

[13] A. Shanthakumar, F. Fassihi, A. Lotfi and J. Bird, "Retrieval-augmented large language model chatbots in higher education: A study on university open days," in *Advances in Computational Intelligence Systems*, 2025, pp. 32-44, doi: 10.1007/978-3-031-78857-4_3.

[14] Netflix, "Zuul API Gateway," [Online]. Available: https://github.com/Netflix/zuul.

[15] Netflix, "Eureka Service Discovery," [Online]. Available: https://github.com/Netflix/Eureka.

[16] PostgreSQL Global Development Group, "PostgreSQL," [Online]. Available: https://www.postgresql.org/.

[17] pgvector, "Vector Extension for PostgreSQL," [Online]. Available: https://github.com/pgvector/pgvector.

[18] Milvus, "Milvus: Vector Database," [Online]. Available: https://milvus.io/.

[19] Chroma, "Chroma: Embedding Database," [Online]. Available: https://www.trychroma.com/.

[20] Facebook AI Research, "FAISS: A Library for Efficient Similarity Search," [Online]. Available: https://github.com/facebookresearch/faiss.

[21] Keycloak, "Keycloak: Identity and Access Management," [Online]. Available: https://www.keycloak.org/.

[22] Ollama, "Ollama: Local Model Serving Framework," [Online]. Available: https://ollama.com/.