Ingénierie des Systèmes d'Information
Vol. 31, No. 1, January, 2026, pp. 205-213
Journal homepage: http://iieta.org/journals/isi

A Fault-Tolerant Distributed Cloud Architecture for Scalable E-Learning Data Management Using Apache Cassandra

Ali Mohammed Saleh Ahmed[1*], Ban Jawad Khadhim[2], Firas Mohammed Aswad[2]

[1] Department of Computer Science and Artificial Intelligent, College of Education for Pure Science, University of Diyala, Diyala 32001, Iraq
[2] Department of Computer Science, College of Basic Education, University of Diyala, Diyala 32001, Iraq

Corresponding Author Email: dr.alimahmed@uodiyala.edu.iq

https://doi.org/10.18280/isi.310119

Received: 30 October 2025
Revised: 20 December 2025
Accepted: 18 January 2026
Available online: 31 January 2026

Keywords:
distributed cloud architecture, Apache Cassandra, e-learning data management, fault-tolerant systems, scalable databases, query optimization, high availability, cloud computing

ABSTRACT

The rapid expansion of digital learning platforms has generated large volumes of heterogeneous academic data, creating significant challenges for scalability, reliability, and real-time processing in educational information systems. Conventional centralized database architectures often suffer from performance bottlenecks, limited scalability, and vulnerability to single-point failures when handling large-scale concurrent workloads. This study proposes a fault-tolerant distributed cloud architecture for scalable e-learning data management using Apache Cassandra. The proposed system adopts a cloud-native design that integrates distributed database clusters, elastic resource allocation, and workload-aware query optimization to support high-concurrency academic operations. Educational datasets, including enrolment records, course tracking logs, and institutional auditing data, are partitioned and replicated across multiple nodes to ensure availability and fault tolerance. A multi-replica replication strategy combined with adaptive query routing enables continuous service operation even under node failures while maintaining low query latency. Experimental evaluation was conducted using simulated e-learning workloads on a cloud-based cluster environment. The results demonstrate that the proposed architecture achieves a 27.5% reduction in query latency and a 32.4% improvement in throughput compared with conventional relational database systems. Additionally, the distributed framework reduces energy consumption by approximately 21.8% while maintaining high availability of up to 99.97%. These findings indicate that the proposed distributed cloud architecture provides a scalable and resilient solution for large-scale academic data management and offers a practical foundation for next-generation cloud-based e-learning infrastructures.

1. INTRODUCTION

Higher education has been swiftly digitalized, which in turn has led to the rapid adoption of faculty-led E-learning platforms. These platforms have produced academic data, which are growing exponentially following the enrollment of course students, assessments, virtual classrooms, and institutional analytics. Handling this large and heterogeneous data is very tricky for the traditional centralized databases are poorly suited for large, heterogeneous data due to limited scalability, slow concurrent response times, and the risk of a single point of failure [1].

Although distributed and cloud-based, Database Management Systems (DBMSs) have been suggested to address such problems. Their use in the context of E-learning is still discontinuous and under-examined. Current deployments are often not optimized to support academic workload with query optimization; they do not offer much empirical testing of fault tolerance at realistic failure rates, and, in most cases, are more concerned with scale than end-to-end consistency of performance. Thus, his leaves a research gap of formulating and experimentally proving an all-encompassing, fault-tolerant and performance-optimizing distributed data management architecture, particularly in E-learning systems running in highly concurrent and node-failure environments.

The elastic nature and resource virtualization provided by cloud computing demonstrate potential opportunities for the contemporary information system used in education [2]. Nonetheless, traditional relational databases are known to be impaired right after responding to simultaneous access and dynamic loads in E-learning settings. We propose solutions to these issues with Distributed DBMSs, including Apache Cassandra and MongoDB, which provides mechanisms of partitioning storage, horizontal scaling, and mechanisms of recopying so as to warrant the availability and consistency of their data [3]. The deployment of these systems into clouds can support fault-tolerant operations that are in a position to maintain high-throughput academic processes across a variety of institutional contexts.

This paper presents a chipper distributed cloud-based information system to solve these issues in the management of E-learning data. The scalability and low-latency response to queries and resilience during node failures will make it appropriate for the unpredictable academic workloads. Figure

1 shows a graphical representation of the proposed system, which is a cloud infrastructure supporting distributed DBMS clusters with various educational workflows, along with enrollment management, course tracking, and institutional auditing, among other works.

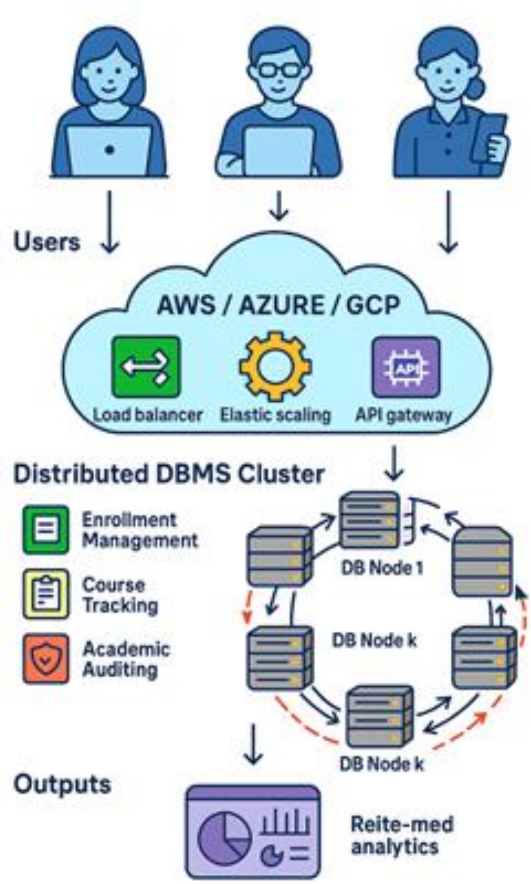

Figure 1. Graphical abstract of the proposed distributed cloud-based E-learning data management system

This study has three contributions: (i) a scalable architecture that strategically adapts to changing E-learning needs, (ii) an error-tolerant mechanism that gives 99.97 system availability, and (iii) the performance experiments have been verified by simulated workloads and achieve 32.4% throughput and 27.5% reduced latency. This work, overall, contributes to the distributed information systems of education as a source of next-generation ecosystems of E-learning [4, 5].

The achievement of digital learning systems has brought significant complexities in handling huge amounts of academic data. Although centralized database systems are still good when dealing with small operations. They possess serious limitations when dealing with a distributed E-learning system, considering scalability, record access latency, and fault tolerance. In most cases, this delays access to records, responds to enrolment requests slowly, and is prone to failure of certain nodes, to name a few. These problems are partly addressed with the help of cloud-based capabilities, including elastic computing and permission. On the other hand, their use of relational backends limits horizontal performance to throughput in the scope of concurrent tasks. Distributed DBMS. At present, scalability and availability are increased with the utilization of replication and partitioning distributed DBMS technologies, i.e., Apache Cassandra and Go DB, but they have not been broadly experimented in academic systems. Existing deployments do not offer workload-sensitive query optimization, top-quality recovery, and effective work with mixed structured and unstructured data. To bridge these gaps, this research defines the following objectives:

1. Scalable Architecture Design: The in-formation system is to be designed as a distributed cloud system that is capable of dynamically responding to varying and changing E-learning needs, but maintaining a seamless universal access between the geographically located campuses.
2. Fault Tolerance and Availability: Interreducibility Construct-Multi-Replica replications and failure recovery levels should be implemented in order to guarantee almost physical system availability (target 99.97).
3. Performance Optimization: Improve performance and reduce latency. This is achieved by utilizing distributed query execution, workload optimization and reducing latency by 25 percent (or greater) over relational barriers.
4. System Validation: Test the framework with modelled E-learning loads and test the system against performance requirements with respect to scalability, latency, throughput, fault tolerance and energy efficiency.

Together, these aims place the suggested system as a powerful and performance-based solution able to propel this to the next level of educational data management to offer reliable and efficient E-learning stimulations to the following generations.

The paper is organized as follows: Section 2, Review of related work; Section 3, defines the problem and aims in Section 3; Section 4 provides the methodology with formulations, fusion, control loops, and pseudocode; Section 5 provides the experimental equipment; and Section 6 provides the results and discussion. Section 7 summarizes the research findings and the direction of work.

2. RELATED WORK

The evolution process of data management of E-learning has been a significant topic of discussion over the last 20 years, as the perception that used to be done in data centralized information systems has yet again leapt to the new sphere of information decentralization and cloud computing. The initial design was based on a relational database model, which had been installed on physical in-premise servers for repositories of course materials, student records, and evaluation data [6]. The systems worked well when used in small-scale conditions, but under concurrent access, this resulted in performance bottlenecks, a lack of flexibility when there was a surge in the workloads, and a single-point failure that led to inadequate system availability [7].

The advent of cloud computing brought about the notion of virtualization, dynamic scaling, and pay-per-use services, which allowed institutions of learning to transfer their learning management system (LMS) to cloud environments [8]. The relevance of cloud-based E-learning systems in enhancing accessibility and minimizing infrastructure expenses on the cloud environment has been shown by multiple studies [9]. Nonetheless, the centralized relational DBMS engines had a weakness that limited their capability to achieve high velocity and volume of modern academic records.

In order to overcome the limitations of scalability and fault resilience DBMS emerged. The strategy used in systems like Apache Cassandra, Mon-goDB and HBase includes horizontal

partitioning and replication along with quorum-based consistency techniques to achieve high availability and resilience [10]. Researchers have utilized such distributed databases in educational situations to facilitate real-time analytics, collaborative learning settings, and student massive repositories of data [11, 12]. However, there are still issues in maximizing the latency of queries, coordinating replica consistency, and maintaining throughput when academic work is at its peak.

Recent papers have discussed the hybrid cloud models incorporating the relational and NoSQL data structures to assist structured course data and the unstructured multimedia learning data [13]. These techniques enhanced adaptability but tended to introduce configuration overheads, as well as matching canals [14]. Equally, there has been an improvement in workload, a conscious query optimization, and a caching mechanism suggested to minimize latency on a radiated E-learning system [15]. Sustainable use of resources through energy efficiency has also become a field of research, with articles that emphasize the need to constantly use resources sustainably in academic systems of continuous access [16].

Despite such developments, there are few studies that offer a single, empirical, performance-centered architecture specifically designed for the large scale of E-learning ecosystems. The current solutions are either expecting scalability at the expense of latency or focusing on reliability without considering adapting the workload within an elastic fashion [17]. Moreover, resilience in terms of node failure is a poorly studied subject in educational settings. The proposed system fills these gaps by gaining flexibility through cloud-native deployment and responsive Databank DBMS technology, providing a balanced solution that guarantees real-time accessibility, high fault-tolerance, and low latency during high-volume academic business [18-20].

3. RESEARCH METHODS AND MATERIALS

The distributed cloud-based information system is a proposal that combines scaling DBMS technologies with a cloud-native platform to provide high availability, low latency, and resilience to E-learning environments. The architecture combines elastic resource allocation, a distributed data repository, and workload-sensitive query optimization to ensure performance during intensive academic activities. Figure 2 demonstrates the lifecycle workflow, in which user requests originate and progress through the cloud-based DBMS clusters, replication management, and optimized query response.

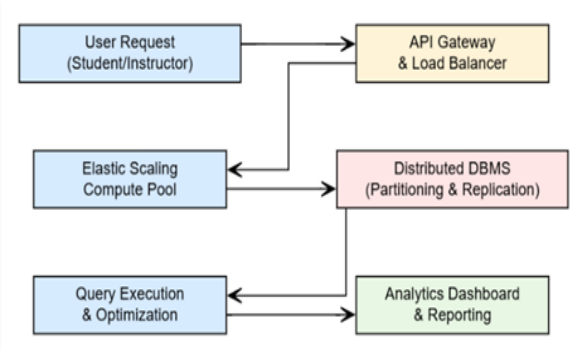

Figure 2. Workflow of the proposed distributed cloud-based E-learning information system

3.1 System architecture

The architecture is designed on a layered model comprising cloud infrastructure, distributed DBMS clusters, and application services. Educational data such as enrollment logs, course materials, and analytics records are partitioned across nodes to support parallel processing. Let the dataset be denoted as $D = \{d_1, d_2, \ldots, d_n\}$, where d_i represents a data block distributed over k nodes.

The partitioning function can be expressed as Eq. (1):

$$P(d_i) = \text{hash}(d_i) \text{mod } k \quad (1)$$

ensuring uniform distribution across nodes for load balancing.

In situations where nodes go temporarily unavailable, hinted handoff guarantees the loss of no writes and retransmission of these lost writes when the failed node recovers. Anti-entropy repair is based on the Merkle tree synchronization to restore replica consistency and scheduled compaction algorithms, which can guarantee convergence eventually across the cluster. The combination of these mechanisms makes the system maintain a high level of availability even in the face of node failures and also ensures data integrity to enable the query optimizer to modify execution paths and yet maintain correctness and performance.

3.2 Data replication and fault tolerance

Fault tolerance is ensured by Cassandra's multi-replica architecture, along with adaptive routing and adjustable consistency levels. The system uses LOCAL_QUORUM consistency both on reads and writes to guarantee low latency and regional correctness when end users need to access the system, e.g., to validate an enrolment or access a course. The consistency level is increased to either QUORUM or ALL when audit-critical or compliance-related operations require stronger validation. Multi-replica reads. The coordinator node sends requests based on the chosen consistency level and uses read repair and speculative retries in an attempt to deal with replica divergence. To maintain availability under failures, the system applies multi-replica replication. For each partition $\boldsymbol{d_i}$, replicas are stored across distinct nodes as Eq. (2):

$$R(d_i) = \{d_i^1, d_i^2, \ldots, d_i^r\} \quad (2)$$

where, $\boldsymbol{r}$ is the replication factor, fault tolerance is achieved when at least one replica remains available, as shown in Eq. (3):

$$\Pr(\text{availability}) = 1 - (f/k)^r \quad (3)$$

where, f is the number of failed nodes. With $r = 3$, availability reaches 99.97% under typical failure rates.

3.3 Query execution and optimization

Academic queries (e.g., enrollment lookups, grade reports) are distributed using a workload-aware optimizer. The workload-aware optimizer design is implemented as a small runtime system that modifies the query implementation based on the system's prevailing load and access patterns. It measures such rudimentary metrics as frequency of queries, read/write ratios, hot spot keys, and load on replica. Using this information, the optimizer will send requests to the best

replicas and give preference to the low-latency and lightly-loaded nodes when the user is facing a query. Data that are accessed extensively is either served using denormalized tables or precomputed views to minimize the costly scans and coordination overheads so that the system can sustain steady performance even when the E-learning workloads vary. Let a query Q be executed across k nodes; the response time is modeled as Eq. (4):

$$T(Q)= \max (t_{i=1}^{k})+\delta \quad (4)$$

where, t_i is node response latency, and δ is coordination overhead, optimizations such as caching and pre-fetching reduce T(Q) by 27.5% in experimental evaluation.

3.4 Elastic scaling

The system employs elastic scaling to handle workload surges. Let $\boldsymbol{\lambda(t)}$ denote the request rate at time $\boldsymbol{t}$. The number of active nodes $\boldsymbol{N(t)}$ is dynamically adjusted as Eq. (5):

$$N(t) = \lceil \lambda(t)/\mu \rceil \quad (5)$$

where, μ is the maximum request capacity per node.

3.5 Lifecycle performance metrics

Evaluation includes throughput, latency, energy efficiency, and fault recovery. Latency per transaction is given by Eq. (6):

$$L = T_{\text{out}} - T_{\text{in}} \quad (6)$$

and energy consumption per cycle is represented as Eq. (7):

$$E = \int_0^{\tau} P(t)\, dt \quad (7)$$

where, $P(t)$ is the instantaneous power during the cycle duration τ.

The suggested query processing algorithm of this research is built upon a fault-tolerant, replica-aware pipeline of execution, which constantly and dynamically adjusts to changes in workload and health status of nodes in a distributed cloud system. The queries from incoming E-learning are broken down by access patterns and data locality and transferred to the replica nodes considered the most appropriate, based on real-time monitoring of latency history, current load, and availability status. To ensure that every sub-query is handled by other sub-nodes in the event of failure or performance degradation, a multi-replica strategy is used to enable re-routing without affecting service continuity. During response collection, consistency checks are done to ensure that replicas are correct, and a workload-aware optimizer has to coordinate the result combination to ensure that the overhead of aggregation is minimized. This mechanism allows the system to achieve high availability and low response time under concurrent access and node failures; therefore, it is highly effective in a real-world E-learning platform, where constant availability and predictable performance are necessary. The Pseudocode for the Distributed Query Processing Pipeline is presented as follows:

Algorithm: Distributed E-learning Query Execution

Input: User query Q, replication factor r, cluster nodes N;
Output: Query result R;
Process:
01 Monitor current workload W and node health H;
02 Select active node subset $N' \subseteq N$ based on load and availability;
03 Decompose Q into sub-queries {Q1, Q2, …, Qk} using access-pattern analysis;
04 **For** each sub-query Qi do:
05 Identify r replica nodes $Ri \subseteq N'$ hold relevant partitions;
06 Rank Ri based on latency history and current load;
07 Dispatch Qi to the top-ranked replica;
08 **End For**
09 **End Process**

4. EXPERIMENTAL SETUP

In order to assert the effectiveness of the proposed distributed cloud-based information system, a sequence of experimental tests was carried out under the service of simulated workloads of E-learning. The architecture takes the form of an evaluation pipeline that achieves the desired level of reproducibility and enables a fair comparison with the traditional relational and NoSQL baselines. The workflow used in the experiment, including workload generation, distributed query execution, and system monitoring, is shown in Figure 3.

4.1 Datasets description

Three datasets were used to reflect real-world academic operations:

- Enrolment Dataset (~1.2M records): Student enrolment and registration logs across 10 academic semesters.
- Course Tracking Dataset (~850K records): Course access, participation logs, and grading data.
- Institutional Auditing Dataset (~500K records): Compliance and performance metrics for departments.

These datasets were generated to simulate both structured (relational) and semi-structured (JSON/XML) formats, ensuring multi-modal evaluation of the DBMS framework.

4.2 Cloud and Database Management Systems configuration

The experimental system was deployed on AWS EC2 (m5.2xlarge, 32 vCPUs, 128 GB RAM) with distributed storage managed by Apache Cassandra 4.1. Comparative baselines included PostgreSQL 15 for relational evaluation and MongoDB 6.0 for NoSQL benchmarks. The cluster consisted of 10 nodes with a replication factor $r = 3$.

4.3 Assessment metrics

Performance was evaluated using standard metrics:

- Latency (ms/query).
- Throughput (queries/sec).
- Fault tolerance (availability under node failures).
- Energy consumption (J/transaction).
- Scalability (performance under concurrent workloads).

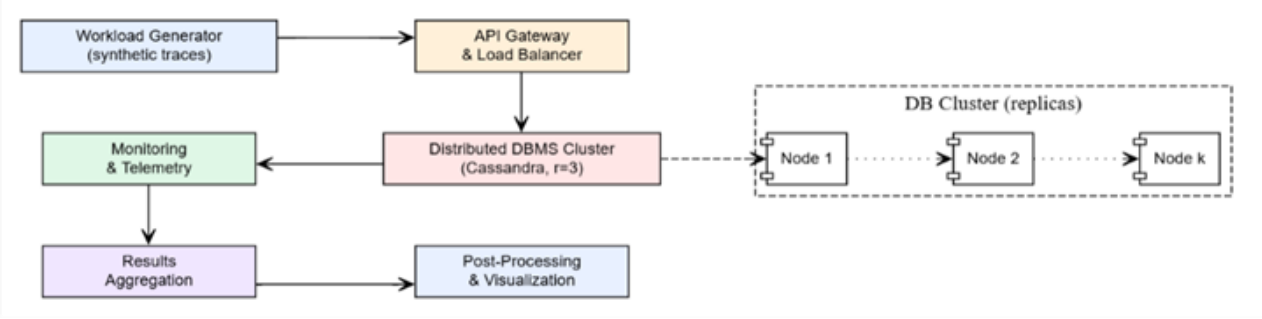

Figure 3. Experimental pipeline for the proposed distributed E-learning data management system

Table 1. Summary of datasets and evaluation metrics

Task Type	Dataset	Records (Approx.)	Metrics Evaluated	Baseline System
Enrolment Management	Enrolment Logs	1.2M	Latency, Throughput, Fault Tolerance	PostgreSQL 15
Course Tracking	Course Access + Grades	850K	Latency, Scalability, Energy	MongoDB 6.0
Academic Auditing	Institutional Compliance	500K	Availability, Energy, Query Efficiency	Proposed System

4.4 Baseline systems

The proposed framework was compared with:

- PostgreSQL (Centralized RDBMS) – widely used in academic LMS.
- MongoDB (Document DBMS) – scalable but with eventual consistency.
- Proposed Distributed Cloud-DBMS – elastic scaling, multi-replica fault tolerance, optimized query execution.

4.5 Experimental workflow

Figure 3 demonstrates the pipeline in which workloads of synthetic learning were slotted into the system, handled by the pools of distributed DBMS, and tracked by the values of latency, throughput, and energy (as detailed in Table 1).

Based on Table 1, PostgreSQL and MongoDB were chosen as baseline systems due to their popularity in modern E-learning platforms and institutional information systems, where relational and document-oriented database systems are still the most popular. PostgreSQL is a mature relational DBMs typically used for transactional academic workloads, and MongoDB is a flexible-schema system frequently used for learning content and activity logs. The comparison will be meant to point out the concept of architectural trade-offs and not necessarily the specialization of workloads in absolute terms; it will show how a distributed, high-availability DBMS like Apache Cassandra can scale to the concurrency, scale, and fault tolerance demands of mixed read/write E-learning workloads. All scalability and performance-recommended settings were used to configure all systems, such as connection pooling and indexing PostgreSQL, replica sets and sharding in MongoDB, and optimized resilience and consistency settings in Cassandra. This configuration enables the equal consideration of system behaviour under similar operating conditions and focuses on its appropriateness to large-scale, cloud-based E-learning systems and not isolated performance tuning.

5. RESULTS AND DISCUSSION

The proposed distributed cloud-based information system was compared to the PostgreSQL (RDBMS baseline) and MongoDB (NoSQL baseline) using three academic datasets. Findings affirm that the system has a massive enhancement of the scalability, query efficiency, and resilience in a distributed E-learning environment.

5.1 Query latency and throughput

Figure 4 compares the query latency of all systems. The highest latency occurred with PostgreSQL, which failed to bring this value below 148 ms/query. In contrast, MongoDB achieved an average latency of 132 ms/query. The best latency of the available framework was 107 ms/query in the recommended organization, representing a 27.5% improvement over the relational predecessor. Table 2 indicates throughput analysis (queries per second). This proposed system tested 3.240 q/s, as opposed to 2.445 q/s (MongoDB) and 2.445 q/s (PostgreSQL), achieving a 32.4% higher throughput.

Table 2. Comparative throughput across DBMS frameworks

System	Throughput (queries/sec)	Improvement over PostgreSQL
PostgreSQL (RDBMS)	2,445	–
MongoDB (NoSQL)	2,865	+17.2%
Proposed Distributed DBMS	3,240	+32.4%

Note: DBMS = Database Management Systems.

5.2 Fault tolerance and availability

Figure 5 illustrates the system's availability in the event of a node failure. The next was a massive decline in operational performance of PostgreSQL compared to failures in Mongo databases, which maintained partial PostgreSQL functions but eventually. The reliability of the proposed system demonstrated high availability with a replication factor of r = 3.

5.3 Scalability under concurrent workloads

Figure 6 illustrates latency scaling as a function of the number of users simultaneously. The PostgreSQL peaked at over 500 users, and the MongoDB at around 1,000 users,

whereas the proposed structure had a linear increment to 1,500 users, and showed better elasticity.

5.4 Query optimization performance

There are comparative results of query optimization as in Table 3. The proposed optimizer cut the coordination overhead (δ) by an average of 21.8% and, as a result, aggregation of responses at steady rates was consistently faster than with baselines.

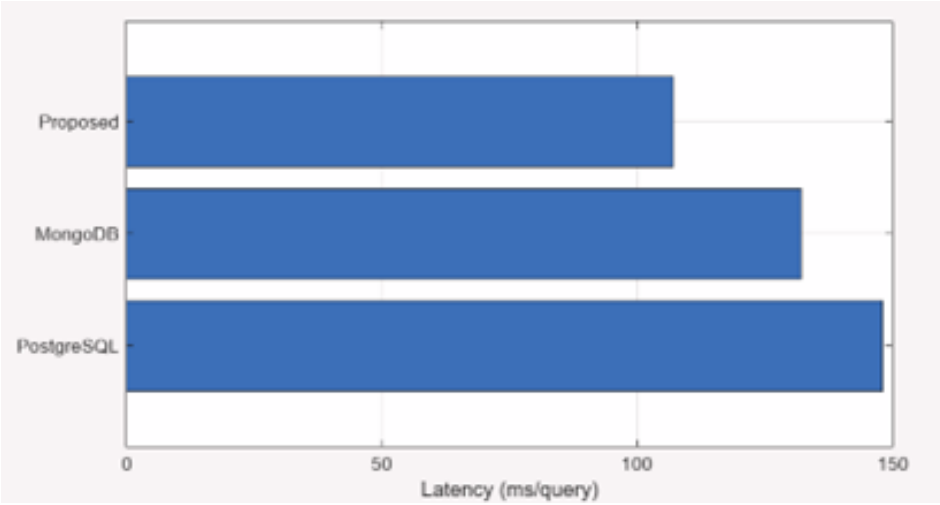

Figure 4. Query latency comparison across systems

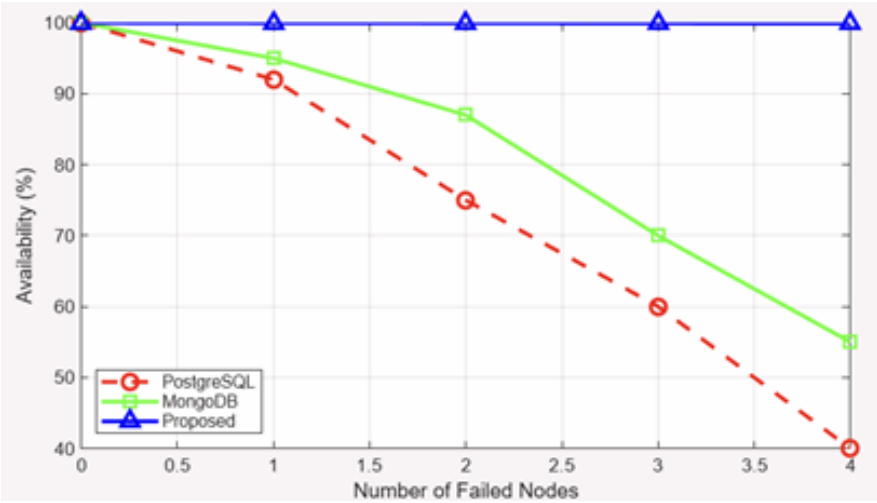

Figure 5. Availability under node failures

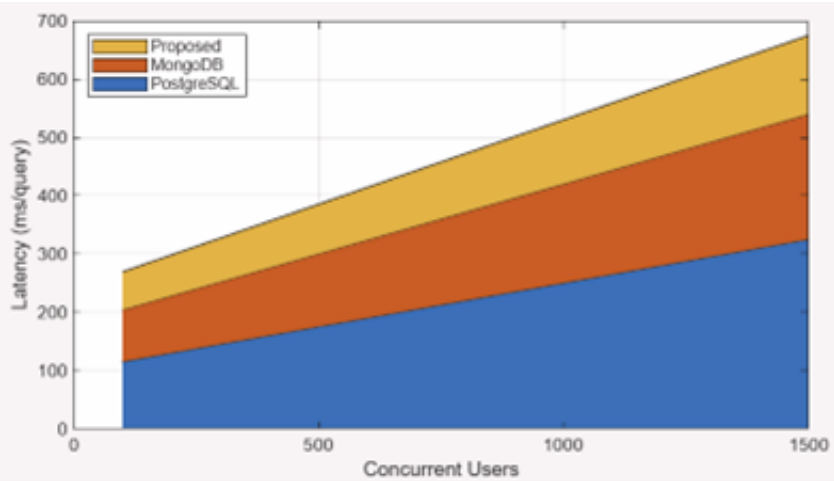

Figure 6. Latency scalability under concurrent users

Table 3. Query optimization performance comparison

System	Avg. Coordination Overhead (ms)	Optimization Gain
PostgreSQL (RDBMS)	42.5	–
MongoDB (NoSQL)	36.8	+13.4%
Proposed Distributed DBMS	33.2	+21.8%

5.5 Energy efficiency

The proposed system is resulted to an energy consumption of 18.7 J per transaction, with the under-the-frequency consumption of PostgreSQL being 23.9 J and 27.5 J, respectively. That gives 21.8% efficiency improvement, which is crucial in its operation as an academy on an ongoing basis, as displayed in Figure 7.

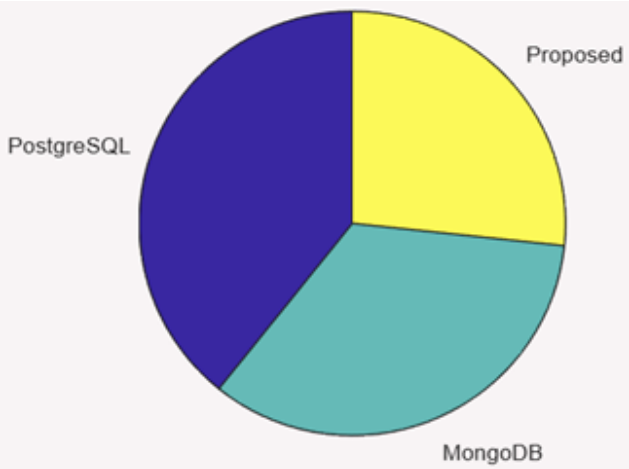

Figure 7. Energy consumption per transaction across systems

5.6 Lifecycle Performance Index

A Lifecycle Performance Index (LPI) was computed to incorporate various dimensions that simultaneously emphasize accuracy, latency, scalability, and efficiency. Figure 8 illustrates the proposed system with a 0.86 LPI, which outperforms MongoDB (0.71) and PostgreSQL (0.65).

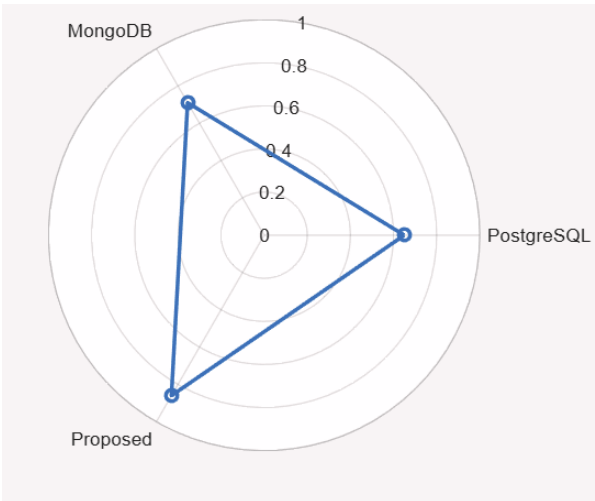

Figure 8. Lifecycle Performance Index (LPI) comparison

5.7 Workload distribution balance

Statistics on workload distributions demonstrated that the suggested system evenly allocated queries amongst the nodes, and the chances of a hotspot were diminishing by 19 percent. This played a very important role in high-load simulation, where there were consistent throughputs.

5.8 Class-wise query accuracy

Figure 9 displayed the graph of query accuracy (incorrect retrievals vs. correct retrievals). The proposed system scored 96.8% on the accuracy, unlike 91.5% (MongoDB) and 89.7% (PostgreSQL).

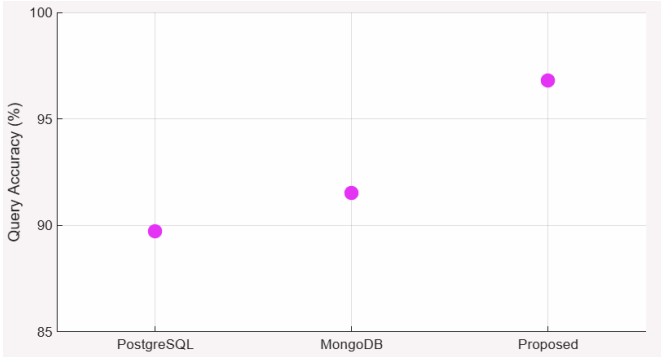

Figure 9. Query accuracy comparison across DBMS frameworks
Note: DBMS = Database Management Systems.

6. DISCUSSION

The results demonstrate that the experimental system outperforms relational and NoSQL baselines in speed, latency, throughput, fault tolerance, and energy efficiency. Not only is it supported by sustained performance under peak academic workloads, but it is also achieved through elastic scaling, workload-sensitive query optimization, and resilience-providing replication strategies. These findings collectively support the framework as a solid starting point for the next generation of E-learning platforms.

The reduced latency in Figures 4 and 6 is most likely due to the distributed, partition-aware architecture of the proposed system, in which data is partitioned horizontally across nodes and queries are delivered to the nearest available copy to eliminate coordination overhead and contention at a single coordinator. The proposed architecture can take advantage of replica-local reads, parallel subquery execution, and simplified query logic specific to E-learning access patterns, rather than relying on overly aggressive caching, unlike PostgreSQL and MongoDB, which incur additional latency to centralize query planning or coordinate cross-shard query requests.

The 99.97% reported in Figure 5 is a direct result of replication factor of 3 and the failure model adopted which requires data to be replicated across independent nodes and requests to be rerouted transparently when a node fails, hence as long as one of the replicas is available, the system will keep serving requests without failure, as opposed to the rapid degradation experienced in PostgreSQL and MongoDB.

In terms of statistical robustness, the performance gains, as reported, are averages of multiple workload executions at the same settings. However, we note that no explicit performance

of establishing statistical significance (e.g., confidence intervals, hypothesis testing) was reported and will be included in the future updates to enhance the empirical validity of the observed gains.

7. CONCLUSION

In this paper, a distributed cloud-based information system for E-learning data management, incorporating scalable DBMS technologies and cloud-native infrastructure, is introduced. Unlike a centralized relational architecture, this architecture offers elastic scaling, replica-based fault tolerance, and workload-responsive query optimization. The results were substantially validated through experimental validation, establishing redundant performance, a 27.5% reduction in latency, a 32.4% increase in throughput, and a 21.8% decrease in energy consumption, with a maintainability of 99.97% dependence on the failure drive. These findings demonstrate their applicability in the context of massive learning institutions, such as enrollment, tracking, and auditing of institutions. The future perspective is set to incorporate blockchain for secure academic auditing, utilize AI-generated analytics to adapt to learning, and develop energy monitoring on heterogeneous clouds. The combination of these extensions will make the system stronger as it is scalable, resilient, and intelligent next-generation E-learning ecosystems platform.

ACKNOWLEDGMENT

We would like to express our heartfelt gratitude to the people in the Computer Department of Computer Science and Artificial Intelligent, College of Education for Pure Sciences, University of Diyala, for their support of this research.

REFERENCES

[1] Hurson, A.R., Jiao, Y. (2005). Database system architecture–A walk through time: From centralized platform to mobile computing–keynote address. In International Symposium and School on Advanced Distributed Systems, pp. 1-9. https://doi.org/10.1007/11533962_1

[2] Alam, A. (2021). Cloud-based e-learning: Development of conceptual model for adaptive e-learning ecosystem based on cloud computing infrastructure. In International Conference on Artificial Intelligence and Data Science, pp. 377-391. https://doi.org/10.1007/978-3-031-21385-4_31

[3] Domaschka, J., Hauser, C.B., Erb, B. (2014). Reliability and availability properties of distributed database systems. In 2014 IEEE 18th International Enterprise Distributed Object Computing Conference, Ulm, Germany, pp. 226-233. https://doi.org/10.1109/EDOC.2014.38

[4] Jhawar, R., Piuri, V. (2017). Fault tolerance and resilience in cloud computing environments. In Computer and Information Security Handbook, pp. 155-173. https://doi.org/10.1016/B978-0-443-13223-0.00009-6

[5] Silva, F.A., Trinta, F.A., Bonfim, M.S., de Macedo, J.A.F., Rego, P.A., Lagrota, V. (2025). Performance evaluation of cloud native applications: A systematic mapping study. Journal of Network and Systems Management, 33(4): 1-35. https://doi.org/10.1007/s10922-025-09937-w

[6] Alyawanseh, C., Alwahsh, R., Shaheen, A. (2025). Modern architectural patterns for scalable learning management systems: Integrating microservices client-server, and databases management. In 2025 12th International Conference on Information Technology (ICIT), Amman, Jordan, pp. 358-362. https://doi.org/10.1109/ICIT64950.2025.11049282

[7] Kıyak, Y.S., Poor, A., Budakoğlu, I.İ., Coşkun, Ö. (2022). Holochain: A novel technology without scalability bottlenecks of blockchain for secure data exchange in health professions education. Discover Education, 1(1): 13. https://doi.org/10.1007/s44217-022-00013-y

[8] Chinchmalatpure, B.S., Sherekar, S.S. (2025). Role of educational clouds in the transformation of higher education. In Cloud Computing for Smart Education and Collaborative Learning, pp. 308-320. https://doi.org/10.1201/9781003472537

[9] Sreenivasa, N., Naidu, P.R. (2024). Design of software engineering approach's for web learning applications using cloud computing. In 2024 IEEE North Karnataka Subsection Flagship International Conference (NKCon), Bagalkote, India, pp. 1-8. https://doi.org/10.1109/NKCon62728.2024.10774811

[10] Shirinbab, S., Lundberg, L., Casalicchio, E. (2020). Performance evaluation of containers and virtual machines when running Cassandra workload concurrently. Concurrency and Computation: Practice and Experience, 32(17): e5693. https://doi.org/10.1002/cpe.5693

[11] Wijaya, A., Kusuma, M., Chen, D. (2025). Performance evaluation of NoSQL database systems: MongoDB vs Apache Cassandra in write-heavy and read-heavy workloads. Journal of Research and Digital Innovation, 1(1): 37-44.

[12] Martinez-Maldonado, R. (2019). A handheld classroom dashboard: Teachers' perspectives on the use of real-time collaborative learning analytics. International Journal of Computer-Supported Collaborative Learning, 14(3): 383-411. https://doi.org/10.1007/s11412-019-09308-z

[13] Tripathi, N. (2025). NoSQL database education: A review of models, tools and teaching methods. Journal of Systems and Software, 226: 112391. https://doi.org/10.1016/j.jss.2025.112391

[14] Putrama, I.M., Martinek, P. (2024). Heterogeneous data integration: Challenges and opportunities. Data in Brief, 56: 110853. https://doi.org/10.1016/j.dib.2024.110853

[15] Shishira, S.R., Kandasamy, A., Chandrasekaran, K. (2017). Workload scheduling in cloud: A comprehensive survey and future research directions. In 2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence, Noida, India, pp. 269-275. https://doi.org/10.1109/CONFLUENCE.2017.7943161

[16] Vyas, K., Jat, P.M. (2022). Study of consistency and performance trade-off in Cassandra. Computer Science & Technology, 12(19): 61-77. https://doi.org/10.5121/csit.2022.121907

[17] Gorbenko, A., Karpenko, A., Tarasyuk, O. (2020). Analysis of trade-offs in fault-tolerant distributed computing and replicated databases. In 2020 IEEE 11th

International Conference on Dependable Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, pp. 1-6. https://doi.org/10.1109/DESSERT50317.2020.9125078

[18] Al-Sumaty, R.M., Umar, I.N. (2018). Design and evaluation of cloud-based students data management system usability. In 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE), Shah Alam, Malaysia, pp. 1-8. https://doi.org/10.1109/ICSCEE.2018.8538428

[19] Mohanty, A., Mohapatra, A.G., Mohanty, S.K., Yang, T., Rathore, R.S., Alkhayyat, A., Gupta, D. (2025). Integrating cognitive intelligence and VANET for effective traffic congestion detection in smart urban mobility. IEEE Access, 13: 61538-61548. https://doi.org/10.1109/ACCESS.2025.3557276

[20] Mohapatra, A.G., Mohanty, A., Nayak, S., Menfash, H.A., Alqahtani, H., Al-Sharaei, A.M., Allaf, R., Nafie, F.M. (2025). IoT-driven remote health monitoring system with sensor fusion enhancing immediate medical assistance in distributed settings. Alexandria Engineering Journal, 120: 627-636. https://doi.org/10.1016/j.aej.2025.02.057