

Design and Implementation of a Real-Time Web Infrastructure for Student Monitoring: A Kafka-Based Plugin for Moodle

Rima Kilany Chamoun¹, Wadad Wazen² and Mario Gharib²

¹*Saint-Joseph University, Faculty of Engineering, Beirut, Lebanon*

²*Saint-Joseph University, CINIA, Beirut, Lebanon*

Keywords: Learning Management Systems, Moodle, Apache Kafka, Real-Time Streaming, Event-Driven Architecture, Educational Analytics, Web Infrastructure.

Abstract: Modern Learning Management Systems (LMS) require increasingly responsive and scalable infrastructures to support real-time learning analytics. This paper presents the design and implementation of robust technical architecture that integrates Moodle, an open-source LMS, with Apache Kafka, a distributed streaming platform, to enable real-time student performance monitoring. The proposed solution captures high-velocity event data from Moodle (e.g., assignment submissions, quiz attempts, forum activity) and routes it through dynamically generated Kafka topics into a scalable pipeline, where it is processed in real time and stored in MongoDB for downstream analysis. This infrastructure supports immediate visualization of engagement data, threshold-triggered alerts, and seamless extensibility toward predictive analytics using Kafka Streams and machine learning models. The system demonstrates how architectural innovations in event-driven web applications can be applied to education, enabling data-driven interventions and advancing the capabilities of LMS platforms beyond traditional batch-based reporting.

1 INTRODUCTION

1.1 Background and Motivation

The evolution of Learning Management Systems (LMS) has significantly transformed digital education, but many platforms still rely on batch-based data processing models. Such models delay insight generation and hinder timely responses to students' evolving needs. The growing demand for responsive, web-scale educational systems highlights the need for infrastructure capable of supporting real-time data collection, processing, and visualization. Event-driven architectures and stream-processing pipelines—standard in modern web applications—are underutilized in educational platforms.

1.2 Objective and Scope of the Study

This paper presents a technical infrastructure that integrates Moodle (Moodle, n.d.) with Apache Kafka to enable real-time educational data streaming. Designed to address performance, scalability, and

extensibility challenges common in LMS environments, the system applies architectural patterns from modern web infrastructure—such as decoupled messaging, real-time stream processing, and scalable storage—to enhance the responsiveness and effectiveness of educational platforms. The goal is to transform Moodle from a traditional, batch-oriented LMS into a dynamic platform for real-time analytics that empowers educators with timely, actionable insights.

At the core of the system, Moodle serves as the primary source of educational data, generating events based on student activities such as logins, quiz submissions, and assignment completions. These events are captured and streamed into a real-time data pipeline, where they are categorized for further processing and analysis. This design enables timely monitoring of student engagement and supports flexible analytic strategies, including both real-time and batch-based insights.

To ensure optimal throughput and fault tolerance, Kafka topic configurations—such as the number of partitions and replication factors—are tuned based on

 <https://orcid.org/0000-0002-5710-6901>

the nature and frequency of events. General guidelines suggest assigning more partitions to high-frequency events (e.g., attendance or quiz submissions) to enable parallel consumption and faster processing, while lower-frequency or high-impact events (e.g., late assignments or failed quizzes) can benefit from increased replication for reliability.

After processing, the data are stored in MongoDB and visualized using external analytics tools. Although both Grafana and Kibana offer robust visualization capabilities, Grafana was selected due to its native support for MongoDB via official and community plugins, its flexibility in building real-time educational dashboards, and its ease of integration without requiring data migration to Elasticsearch. This setup enables dynamic dashboards and in-depth analysis of student activity. The system also implements role-based access controls, providing tailored analytics views for instructors and managers, as detailed in Section 4.4. This integration facilitates timely tracking of engagement and performance metrics, supports threshold-based alerts for proactive intervention, and lays the foundation for future integration with predictive machine learning models.

This study explores the end-to-end communication architecture, demonstrating how real-time data can be captured, processed, stored, and visualized in a seamless, modular pipeline. By combining Kafka's distributed streaming capabilities with Moodle's LMS environment, the infrastructure enables timely educational interventions and contributes to the development of more personalized and engaging learning experiences. Ultimately, this work aims to provide a replicable model for real-time learning analytics that supports improved student outcomes across diverse educational contexts.

2 LITERATURE REVIEW

Extensive research has been conducted on the integration of Learning Management Systems (LMS) like Moodle with various data processing technologies to enhance the monitoring and assessment of student performance. Recent studies emphasize the benefits of integrating Moodle LMS with real-time data streaming platforms like Apache Kafka to enhance student engagement. One proposed architecture uses Apache Flume to capture student activity logs in real time, which are then transmitted via Kafka and processed using Apache Spark for immediate analysis and responsive feedback

mechanisms (Chaffai, 2023). This setup enables institutions to monitor engagement patterns and intervene when students show signs of disengagement. In addition to technical integration, Moodle's native collaborative features—such as discussion forums and group assignments—have been shown to increase student participation and satisfaction, particularly in STEM education contexts (Gamage et al., 2022), (Martin et al., 2021). Furthermore, the use of Moodle's learning analytics tools allows educators to track learner progress, detect at-risk students, and tailor interventions accordingly (Martin et al., 2021). Recent developments also explore the incorporation of artificial intelligence within Moodle to personalize learning. For instance, an AI-powered system has been developed that uses Natural Language Processing (NLP) to send personalized feedback and support messages based on student behavior and performance data (Aammou et al., 2024). Collectively, these approaches highlight how the synergy of real-time analytics, collaborative tools, and AI-driven personalization within Moodle ecosystems can significantly enhance learner engagement and educational outcomes. On the other hand, traditional educational methods, which often rely on periodic assessments, do not sufficiently capture the continuous progress or struggles of students. By incorporating real-time data processing systems such as Apache Kafka, these systems can provide more dynamic and responsive educational experiences. This approach allows for the ongoing assessment of student engagement and understanding, facilitating immediate academic support where necessary. The literature reveals a significant trend towards leveraging technology to transform educational environments, making them more adaptable and attuned to the needs of both students and educators.

2.1 What are the Key Benefits of Integrating Apache Kafka with Moodle?

Integrating Apache Kafka with a Learning Management System (LMS) like Moodle can significantly enhance real-time monitoring and data analysis in educational institutions. Kafka's robust event streaming capabilities allow for reliable, high-volume data exchange, which is crucial for processing educational logs and analytics in real time (Theofanis, Raptis, Cicconetti, & Passarella, 2024). By optimizing Kafka's configuration, such as the number of partitions and brokers, institutions can

tailor the system to meet specific application requirements, improving performance and resource utilization (Theofanis, Raptis, Cicconetti, & Passarella, 2024)(Han, Shang, & Wolter, 2020). Moreover, local analytics tools, such as the JavaScript library developed for Moodle, enable institutions to analyze educational logs without transferring sensitive data to third parties, thus addressing privacy concerns (Amo, Cea, Jimenez, Gomez, & Fonseca, 2021). This local-first approach not only mitigates risks associated with data exposure but also facilitates the use of advanced analytics techniques, such as clustering and visualization, to monitor student engagement and identify at-risk students effectively (Sáiz-Manzanares, Rodríguez-Díez, Díez-Pastor, Rodríguez-Arribas, Marticorena-Sánchez, & Ji, 2021). Overall, the integration of Kafka with Moodle fosters a more responsive and secure educational environment, enhancing both data-driven decision-making and student support.

2.2 How Does Apache Kafka Enhance Real-Time Monitoring

Real-time analytics are pivotal in modern educational systems, providing a framework through which data can be utilized immediately as it is generated. The role of these analytics is to enable educators to make informed decisions swiftly, which is crucial for intervening effectively in students' academic paths. This process involves the continuous monitoring of data points like assignment submissions, quiz performances, and overall engagement levels within an LMS. The integration of real-time analytics into educational practices not only supports personalized learning paths but also fosters an educational environment that can adapt to the pace and style of each student. By facilitating a more granular view of student performance, real-time analytics help educational institutions to maximize their resources and enhance educational outcomes. Apache Kafka offers several benefits for real-time monitoring and data analysis in education-based Learning Management Systems (LMS) like Moodle. Firstly, Kafka's distributed and fault-tolerant architecture ensures high reliability and scalability, which is crucial for handling the large volumes of data generated by user interactions in an LMS (Theofanis, Raptis, Cicconetti, & Passarella, 2024)(Santos, 2023). This capability allows for efficient data streaming and processing, enabling timely insights into student engagement and performance. Moreover, Kafka facilitates seamless integration with various applications, enhancing the ability to analyze data in

real time and make informed decisions quickly (Sanjana, Sinchana, Prabhu, & Sandhya, 2023). Its support for high-throughput message processing ensures that data can be collected and analyzed without significant delays, which is essential for maintaining an effective learning environment (Zhou, Zhou, & Chen, 2024). Additionally, Kafka's ability to trigger alerts based on specific conditions can help educators respond promptly to student needs, thereby improving overall educational outcomes (Santos, 2023). Overall, the implementation of Apache Kafka in an LMS like Moodle can significantly enhance operational efficiency and decision-making processes.

2.3 How Does Apache Kafka Enhance Scalability in Moodle LMS?

Implementing Apache Kafka's event-driven architecture (EDA) can significantly enhance the scalability and reliability of a Moodle-based Learning Management System (LMS) for real-time monitoring and data analysis in educational institutions. Research indicates that EDA improves performance and scalability in microservices by facilitating asynchronous communication, which can lead to a 19.18% increase in response time and a 34.40% reduction in error rates compared to traditional API-driven architectures (Alam, Nugraha, Rohmat, & Darmawan, 2022). Furthermore, Kafka's ability to handle high-volume data streams with low latency makes it suitable for real-time applications, as demonstrated in a cloud-based architecture that processed up to 8000 messages per second reliably (Khriji, Benbelgacem, Cheour, El Houssaini, & Kanoun, 2021). Additionally, optimizing database performance through distributed systems has shown substantial improvements in Moodle's efficiency, with performance gains of up to 384% in course read operations (Johan, Prakasa, Hanani, Rohman, & Utama, 2024). Thus, integrating Kafka with Moodle can address performance bottlenecks and enhance the system's capacity to manage concurrent users effectively, ensuring a robust educational environment.

2.4 What Role Does Event-Driven Architecture Play in Modern LMS Infrastructures?

Event-driven architecture (EDA) has become a pivotal model for designing scalable, responsive Learning Management Systems (LMS) by enabling asynchronous, decoupled processing of student-

generated events such as quiz submissions, logins, or forum activity. In contrast to monolithic or polling-based designs, EDA allows system components to operate independently, processing high-throughput data streams in real time without introducing performance bottlenecks (Kiran, 2021).

In LMS environments like Moodle, EDA improves scalability during peak usage (e.g., exams or course registration) and supports responsiveness through asynchronous event handling (Kommera, 2020). Its modular design also facilitates integration with modern technologies—including real-time data pipelines and machine learning—without disrupting existing services (Malviya, Mandala, Lekkala, & Reddy, 2025). Frameworks based on platforms such as Apache Kafka and Flink add fault tolerance and allow real-time analytics and alerts (Kiran, 2021), (Kommera, 2020).

While EDA introduces complexity in event orchestration and consistency management, these challenges are outweighed by its adaptability, fault resilience, and support for personalized interventions. Our proposed architecture builds on these principles, implementing a Kafka-based event streaming system tailored to the evolving needs of scalable, analytics-driven LMS infrastructures.

3 PROPOSED FRAMEWORK

3.1 Overview of the Proposed System

The proposed system is designed to harness the robust capabilities of Apache Kafka integrated with Moodle to facilitate a real-time student performance monitoring framework. This system transforms the traditional reactive educational model into a proactive, data-driven approach. By leveraging Kafka's efficient data ingestion and processing capabilities alongside Moodle's widespread educational platform, the system can provide immediate insights into student performance and engagement. This integration allows for the continuous evaluation of students' activities and academic progress, enabling educators to intervene promptly and effectively.

3.2 Real-Time Data Pipeline: From Kafka Ingestion to MongoDB Storage

In our SMART plugin architecture, Apache Kafka

serves as the backbone for ingesting real-time educational events from Moodle, such as quiz submissions and forum posts, using the PHP-rdkafka client (GitHub, n.d.). These events are categorized into course-specific Kafka topics for modular processing and fault-tolerant distribution. Once captured, the events are stored in MongoDB, chosen for its document-oriented structure that aligns with Moodle's hierarchical data. MongoDB enables fine-grained querying and longitudinal tracking by storing metadata like timestamps, user IDs, and activity types. This integration supports both real-time responsiveness and historical analysis, empowering educators with timely insights and facilitating role-based access to dashboards via tools like Grafana.

3.3 The Potential Integration of Kafka Streams in the System

The current system uses Apache Kafka to collect real-time student data, which is stored in MongoDB. Integrating Kafka Streams would enable advanced, real-time analytics directly on the data streams before storage. This includes operations like windowed aggregations, stream joins, and real-time scoring. Such enhancements would allow the system to generate immediate insights—including detailed engagement trends, participation levels, and early warning signs for at-risk students (see Figure 1). The enriched data would then be stored in MongoDB for deeper analysis, supporting timely and informed educational interventions.

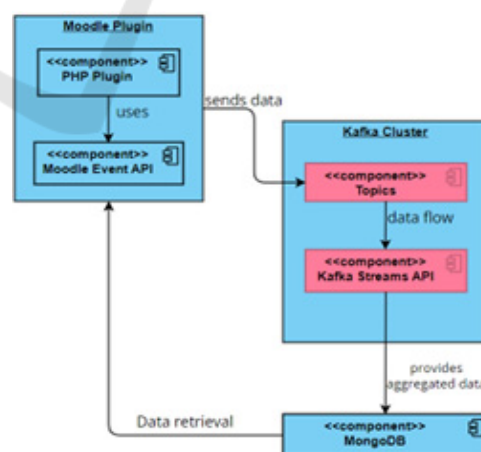


Figure 1: UML Component diagram of our SMART Plugin integrating Moodle with Apache Kafka and MongoDB for real-time data processing.

4 IMPLEMENTATION

In this section we will detail the implementation of the proposed architecture, as shown in the UML deployment diagram in Figure 2. The resources used were as follows:

- Moodle: Our smart plugin is being developed with the latest stable version of Moodle (4.3.2+).
- Apache Kafka: Kafka cluster that includes the Kafka Streams API (Kafka version 3.6.1).
- MongoDB: The latest stable release of MongoDB for data storage (7.0).
- MongoDB Java Driver: Library for interacting with MongoDB from the Kafka Streams application
- MongoDB PHP Library for reading data from MongoDB.
- Java Standard Library, which includes the `java.util.regex.Pattern` package. This package allows the application to subscribe to topics based on a certain pattern, including those corresponding to courses that have not yet been created with their specific topics.
- Rdkafka library: php library for producing/consuming messages.
- Security Protocols: Software to manage secure communications, like SSL/TLS for HTTPS connections.
- Programming Languages and Frameworks: PHP (version 8.1) for Moodle plugin development, Java for Kafka Streams, and any other languages necessary for custom development.

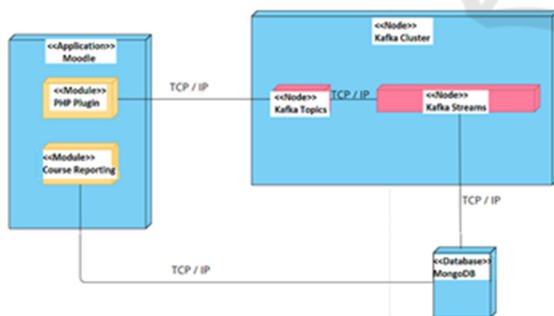


Figure 2: UML Deployment Diagram of the Moodle-Kafka Architecture.

4.1 Data Ingestion from Moodle

Data ingestion is handled by custom event handlers within the SMART plugin, defined in `observer.php` and registered via Moodle's Events API. These handlers respond to key student activities—such as quiz submissions, forum posts, attendance logging,

and assignment uploads—by extracting relevant metadata (e.g., user ID, course ID, timestamps) using `$event->get_data()`, parsing the payload via `parse_event_data()`, and serializing the result into JSON using `json_encode()`. The structured message is then streamed to Apache Kafka using the `php-rdkafka` client (GitHub, n.d.). Each event is routed to a designated Kafka Topic based on its type (e.g., `quiz_submissions`, `forum_posts`, `assignments_submitted`). When a new course is created, the plugin initializes corresponding topics (e.g., `quizzes`, `absences`) to prepare for incoming events. For time-sensitive activities like quizzes and assignments, Moodle's task manager schedules ad hoc checks for non-submissions post-deadline.

Performance tuning identified that using three Kafka partitions with a replication factor of two ensured optimal throughput and low latency for high-volume topics like quiz failures and attendance, while single-partition topics sufficed for lower-frequency events like late assignment tracking. These settings were validated through simulated testing with up to 200 concurrent Moodle users during peak activity.

To ensure compliance with educational data regulations, the system incorporates anonymization protocols and adheres to GDPR and FERPA standards (European Union, 2016), safeguarding student privacy while supporting real-time monitoring.

4.2 Data Storage with MongoDB

After real-time processing (or directly from the Kafka producer in the current implementation), the data is ingested into MongoDB, which serves as the central repository for storing all processed student data. Each document represents an event or processed data point, including fields like student ID, activity type, timestamp, and relevant metrics. Collections are organized by activity type (e.g., `quiz_results`, `forum_interactions`) to align with Kafka topics. Indexes on key fields optimize query performance for generating reports and alerts. MongoDB is deployed as a replica set with configurations prioritizing availability and partition tolerance—using `writeConcern: {w: "majority", j: true, wtimeout: 5000}`, `readPreference: "secondaryPreferred"`, and `readConcern: "available"`. This setup ensures responsive dashboards while accepting eventual consistency for non-critical analytics. For critical operations, stronger consistency is enforced using `readConcern: "majority"` and `readPreference: "primary"`, balancing performance with data integrity (MongoDB Inc., 2024).

4.3 Implementation of Event Handlers and a Kafka Producer Within the Moodle Plugin

The core of the real-time data ingestion process lies in the custom Moodle plugin, which integrates event handlers and a Kafka producer using PHP-rdkafka. Each event handler is designed to listen to specific Moodle events, using the Moodle Events API (Moodle Docs, n.d.), such as when a student submits an assignment, completes a quiz, or posts in a forum. Upon capturing an event, the handler extracts relevant information (e.g., student ID, activity details, timestamps) and formats it into a JSON message.

This message is then passed to the Kafka producer, which is configured to connect to the Kafka cluster. The producer serializes the JSON message using the librdkafka library, ensuring that the data is compact and efficiently transmitted over the network. The message is then published to the appropriate Kafka topic, where it becomes available for real-time processing or direct storage in MongoDB.

The plugin's architecture is designed to handle high throughput, ensuring that even during peak usage times, all student interactions are captured and processed without delay. This is achieved through careful optimization of the event handlers and the Kafka producer's configuration, including tuning parameters such as batch sizes, linger time, and compression settings to balance performance and resource utilization.

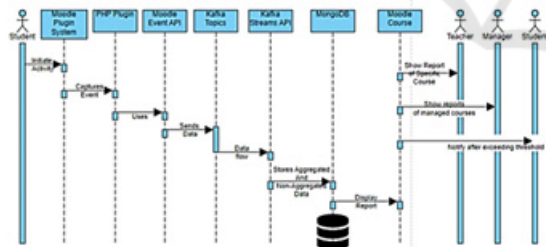


Figure 3: UML Sequence Diagram of the Moodle-Kafka Architecture.

4.4 User Interaction

The system provides educators with role-based, user-friendly interfaces for real-time access to student performance data. Instructors interact with customized Moodle dashboards enhanced by the SMART plugin, which retrieves course-specific data—such as attendance, quiz scores, and participation—from MongoDB. These dashboards support configurable alerts, allowing educators to

define threshold-based triggers for significant events, such as missed assignments or low engagement. Figure 4 shows the SMART plugin interface used to set monitoring parameters and alert conditions tailored to specific course contexts.

At the institutional level, managers access broader analytics via Grafana dashboards, which visualize aggregated data across multiple courses and departments. Data flow from Moodle's event system through Apache Kafka to MongoDB is illustrated in Figure 3, showcasing the real-time pipeline architecture. The reporting interface combines data from Moodle (student names, identifiers) and MongoDB (engagement metrics), presenting it in sortable HTML tables for intuitive exploration. This dual-interface design ensures that both instructors and administrators have timely access to actionable insights, aligned with their respective roles and responsibilities.

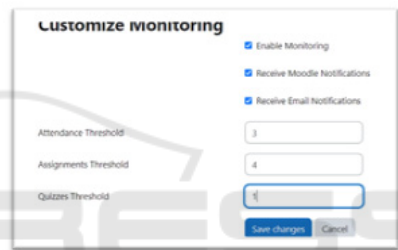


Figure 4: SMART Plugin Configuration panel.

5 RESULTS AND DISCUSSION

The proposed system demonstrates robust real-time monitoring capabilities by integrating Apache Kafka and MongoDB to capture, process, and visualize student engagement data as it occurs. In a simulated deployment reflecting a university with 1,200 students and approximately 3,600 hourly interactions, the SMART plugin streamed events via php-rdkafka to Kafka and stored them in MongoDB. The Kafka cluster—configured with 3 partitions and a replication factor of 2—maintained stable throughput with average end-to-end latency below 80 milliseconds and no data loss, validating its suitability for small-to-medium-scale institutions and its scalability in line with Kafka's capacity to handle tens of thousands of events per second (Theofanis, Raptis, Cicconetti, & Passarella, 2024)(Han, Shang, & Wolter, 2020). The system facilitates timely pedagogical interventions by alerting instructors when students fall behind, and its real-time dashboards—illustrated in Table 1—enable

personalized learning by presenting metrics such as missed assignments, absences, and quiz failures. These analytics empower educators to provide tailored support based on student-specific activity patterns, thereby enhancing engagement and improving learning outcomes. Moreover, the infrastructure supports institutional decision-making through the continuous analysis of performance trends, informing curriculum adjustments, resource allocation, and the design of targeted academic programs. Technical integration challenges—such as configuring Kafka for reliable message streaming, designing scalable MongoDB schemas, and resolving compatibility between Moodle (PHP) and Kafka (Java)—were mitigated through Docker-based containerization and fine-tuned Kafka connector settings. While the system is fully functional in controlled test environments, future work will involve benchmarking its performance with live student cohorts at institutional scale.

Table 1: Sample Monitoring Output Report from Moodle.

ID	Name	Total Assignments Missed	Total Quizzes Missed	Total Absences	Total Quizzes Fails
10	John Doe	8	2	8	0
11	Jane Roe	1	0	1	0

6 CONCLUSIONS AND FUTURE WORK

The SMART plugin shows how integrating Apache Kafka with Moodle enables real-time monitoring and improved student performance assessment within LMS environments. This scalable and responsive infrastructure facilitates timely educational interventions and supports personalized learning. Additionally, the modular framework is designed for extensibility, paving the way for future integration of tools like Kafka Streams and predictive machine learning models.

The following table compares our smart Moodle-Kafka plugin to native Moodle reporting tools.

Table 2: Moodle Kafka Plugin features vs Native Moodle Reporting.

Feature	Moodle Native Reports	SMART Kafka Plugin
Real-Time Monitoring	✗	✓
Dynamic Event Topic Creation	✗	✓
Stream Processing Pipeline	✗	✓ (Kafka-compatible)
Custom Threshold-Based Alerts	✗	✓

Extensible for Future ML Models	✗	✓
Scalable with High User Volume	Limited	✓ (Kafka partitioned topics)
Immediate Intervention Support	✗	✓

6.1 Implications and Contributions

This work tackles the broader challenge of integrating real-time analytics into Learning Management Systems (LMS) to support timely, data-driven educational interventions. It demonstrates how technologies like Apache Kafka and MongoDB can be adapted to educational contexts through modular, event-driven architectures. Key contributions include:

A scalable infrastructure connecting Moodle with Kafka for real-time ingestion, dynamic topic management, and low-latency streaming, with future support for Kafka Streams and machine learning.

The SMART plugin, which extends Moodle with real-time dashboards and alerting features, enabling educators to respond proactively to live student data.

A roadmap for predictive modeling, using historical and streaming data to identify academic risks and support personalized learning.

A bridge between educational practice and scalable web infrastructure, showing how modern data engineering can be embedded into open-source LMS platforms with minimal disruption.

Beyond the technical proof-of-concept, the architecture is generalizable to other platforms, promoting broader adoption of responsive, analytics-driven instruction. Future empirical studies will assess the impact of real-time alerts and predictive models on learning outcomes across diverse academic settings.

6.2 Future Work

Future work will include empirical evaluation of the system’s usability and its effect on educational outcomes, focusing on how real-time feedback influences both student engagement and instructor decision-making. The next development phase involves integrating Kafka Streams and machine learning to transition from monitoring to predictive analytics. Historical data from MongoDB will be used for training models, with features engineered to capture student behavior patterns. Algorithms such as decision trees and neural networks will be applied to predict dropout risk and academic performance. These models will be embedded into the Kafka Streams pipeline for real-time inference, generating alerts for at-risk students. The system will support

continuous learning through periodic model retraining. Predictions and alerts will be stored in MongoDB and visualized through real-time dashboards in Moodle.

ACKNOWLEDGEMENTS

The authors would like to thank Karim Issa and Omar Metlej for their major contribution to the project. The authors also acknowledge the use of OpenAI ChatGPT, for assistance with summarizing content and refining the language during the preparation of this paper.

REFERENCES

- Aammou et al., "Smart agent-based educational system using NLP for personalized feedback in Moodle LMS," **Br. J. Educ. Technol. Syst.**, vol. 12, no. 1, 2024. [Online]. Available: <https://www.brajets.com/index.php/brajets/article/view/1723>
- Alam, R., F. Nugraha, G. Rohmat, and I. Darmawan, "Event-driven architecture to improve performance and scalability in microservices-based systems," in **Proc. IEEE ICADeIS**, 2022, doi: 10.1109/ICADeIS56544.2022.10037390.
- Amo, D., S. Cea, N. M. Jimenez, P. Gomez, and D. Fonseca, "A privacy-oriented local web learning analytics JavaScript library with a configurable schema to analyze any edtech log: Moodle's case study," *Sustainability*, vol.13, no.9, 2021, doi:10.3390/su13095085.
- Chaffai, A., "Real-time analysis of students' activities on an e-learning platform based on Apache Spark," **Academia.edu**, 2023. [Online]. Available: <https://www.academia.edu/109325203>
- European Union, "Regulation (EU) 2016/679 (General Data Protection Regulation)," **Off. J. Eur. Union**, 2016. [Online]. Available: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32016R0679>
- Gamage et al., "The impact of Moodle LMS integration on group discussions to support collaborative learning," **ResearchGate**, 2022. [Online]. Available: <https://www.researchgate.net/publication/382848020>
- GitHub, "php-rdkafka." [Online]. Available: <https://github.com/arnaud-lb/php-rdkafka>
- Han, W., Z. Shang, and K. Wolter, "Learning to reliably deliver streaming data with Apache Kafka," in **Proc. IEEE/IFIP DSN**, 2020, doi:10.1109/DSN48063.2020.00068.
- Johan, E., W. Prakasa, A. Hanani, F. Rohman, and S. N. Utama, "Improving Moodle performance using HAProxy and MariaDB Galera Cluster," **Appl. Inf. Syst. Manag**, vol. 7, no. 1, 2024, doi: 10.15408/aism.v7i1.34871.
- Khriji, S., Y. Benbelgacem, R. Cheour, D. El Houssaini, and O. Kanoun, "Design and implementation of a cloud-based event-driven architecture for real-time data processing in wireless sensor networks," **J. Super comput**, 2021, doi: 10.1007/s11227-021-03955-6.
- Kiran, V. N. S., "Event-Driven architecture: Building responsive and scalable systems," **Int. J. Sci. Res.**, vol. 10, no. 7, 2021, doi: 10.21275/sr24716231109.
- Kommerer, A. R., "The power of event-driven architecture: Enabling real-time systems and scalable solutions," **Türk. J. Comput. Math. Educ.**, vol. 11, no. 1, 2020, doi: 10.61841/turcomat.v11i1.14928.
- Malviya, R. K., V. Mandala, S. Lekkala, and M. S. Reddy, "Event-driven integration in multi-cloud and hybrid architectures: Ensuring data consistency and performance," **SSRN**, 2025, doi:10.2139/ssrn.5080809.
- Martin et al., T. K. "Learning analytics in STEM education: The role of Moodle tools for monitoring engagement," **Int. J. STEM Educ.**, vol. 8, no. 12, 2021. [Online]. Available: <https://link.springer.com/article/10.1186/s40594-021-00323-x>
- Moodle, "Moodle Learning Platform." [Online]. Available: <https://docs.moodle.org/>
- Moodle Docs, "Events API." [Online]. Available: https://docs.moodle.org/dev/Events_API
- MongoDB Inc., "Read and Write Concerns," *MongoDB Manual*, 2024. [Online]. Available: <https://www.mongodb.com/docs/manual/core/replica-set-read-write-concerns>
- Sáiz-Manzanares, M. C., J. J. Rodríguez-Díez, J. F. Díez-Pastor, S. Rodríguez-Arribas, R. Marticorena-Sánchez, and Y. P. Ji, "Monitoring of student learning in learning management systems: An application of educational data mining techniques," **Appl. Sci.**, 2021, doi: 10.3390/app11062677.
- Sanjana, N., R. Sinchana, V. Prabhu, and S. Sandhya, "Real-time event streaming for financial enterprise systems with Kafka," in **Proc. IEEE AsianCon**, 2023, doi: 10.1109/ASIANCON58793.2023.10270532.
- Santos, J. L., "Streamlining enterprise data processing, reporting and real-time alerting using Apache Kafka," in **Proc. IEEE ISDFS**, 2023, doi: 10.1109/ISDFS58141.2023.10131800.
- Theofanis, P., C. Raptis, C. Cicconetti, and A. Passarella, "Efficient topic partitioning of Apache Kafka for high-reliability real-time data streaming applications," **Future Gener. Comput. Syst.**, 2024, doi: 10.1016/j.future.2023.12.028.
- Zhou, Z., L. Zhou, and Z. Chen, "A distributed real-time monitoring scheme for air pressure stream data based on Kafka," **Appl. Sci.**, vol. 14, no. 12, 2024, doi: 10.3390/app14124967.