Optimization of Student Grade Data Management Using RESTful API and Microservices Architecture : Case Study at Universitas Mitra Indonesia

M.Budi Hartanto*[1], Teuku Muhammad Fawaati [2], Fatimah Fahurian[3], Hilda Dwi Yunita[4], Khozainuz Zuhri[5]
[1,2] Department of Information Technology, Faculty of Computer Science, Mitra Indonesia University, Indonesia
[3,4] Department of Information System, Faculty of Computer Science, Mitra Indonesia University, Indonesia
[5] Department of Informatics, Faculty of Computer Science, Mitra Indonesia University, Indonesia
E-mail: *[1]**budi.hartanto@umitra.ac.id**, [2]teuku@umitra.ac.id, [3]fatimah_fahurian@umitra.ac.id, [4]hildadwiyunita@umitra.ac.id, [5]zuhri@umitra.ac.id

Abstract

The management of student grade data is a critical component of academic information systems that require high efficiency and reliability. At Universitas Mitra Indonesia, the old monolithic system faced challenges in scalability, security, and accessibility. This study proposes the implementation of RESTful API and microservices architecture to optimize the management of student grade data. RESTful API serves as the primary interface for inter-service communication, while microservices allow for independent management of modules such as user authentication, grade data processing, and academic service provision. Implementation results demonstrate a 35% increase in data access speed, a 25% reduction in server load, and enhanced security through token-based authentication. This study significantly contributes to modernizing academic information systems, especially in improving the performance and scalability of digital academic services.

Keywords *— Restful API, Microservices, Grade Data Management, Academic Information System, Mitra Indonesia University*

1. INTRODUCTION

With the rapid advancement of information technology, data management has become a significant challenge, particularly in the context of higher education. One critical area requiring special attention is the management of student grade data, which directly impacts the academic evaluation process. Many educational institutions, including Mitra Indonesia University, continue to face difficulties in efficiently managing student grade data. The adoption of modern technology-based information systems, such as microservices architecture and RESTful APIs, can offer solutions to address these challenges.

Microservices architecture is an approach that emphasizes breaking a system into smaller services that interact with each other via APIs. This method provides better scalability, easier maintenance, and overall improved system performance. For example, research conducted by[1] demonstrated that implementing a microservices-based architecture enhanced data integration service management at Udayana University. This system facilitated more structured data management and responsiveness to the institution's changing needs[1].

RESTful APIs, a standard for system communication using the HTTP protocol, connect various microservices in a microservices architecture. Research by.[2] highlighted the importance of implementing RESTful APIs for more efficient management of e-learning systems, which can also be applied to managing student grade data. RESTful APIs enable faster communication between systems and accessibility across multiple platforms, providing the flexibility essential for a dynamic academic environment.

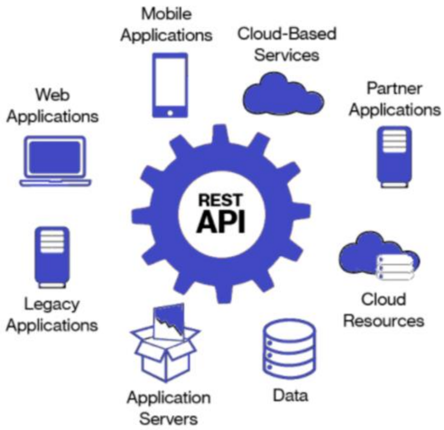

Figure 1. RESTful API & arsitektur mikroservis[1]

Mitra Indonesia University can leverage this technology to manage student grade data more efficiently and transparently. One of the main challenges in managing student grade data is integrating various data sources originating from different systems. By utilizing a microservices architecture, each service—such as grade processing, data storage, and analysis—can operate independently while still communicating with each other to generate accurate reports.

The implementation of microservices architecture also offers scalability benefits. As the number of students grows or when the institution needs to add new modules to support academic processes, a microservices-based system can be easily modified without disrupting existing services. Research by[3]indicates that microservices-based systems enhance scalability in academic data management by breaking large applications into smaller, independent components, enabling seamless further development.

Table 1. Benefits and Implementation of Microservices Architecture and RESTful API in Managing Student Grade Data at Mitra Indonesia University [3]

Aspect	Explanation
Challenges	Efficient management of student grade data and integration of various data sources from different systems.
Solution	Implementation of a microservices architecture that divides the system into small services interacting through RESTful APIs.
Benefits of Microservices Architecture	- Better scalability - Easy maintenance - Improved system performance

Aspect	Explanation
RESTful API	- Connects microservices within the architecture - Faster communication between systems and accessible from various platforms - Increases flexibility
Application at Universitas Mitra	More efficient, transparent, and integrated management of student grade data across various existing systems, such as the academic system and student portal.
Efficiency and Scalability	- Allows the addition of new modules or system changes without disrupting existing services - Reduces latency and improves performance with an API Gateway.
Case Studies	Research indicates that the implementation of microservices architecture improves efficiency and scalability in academic data management at other universities.

Beyond scalability, efficiency in data management is equally crucial. At Mitra Indonesia University, combining microservices with RESTful APIs can enhance operational efficiency by reducing the time required for data processing. Research by.[4] highlights that an API Gateway, which manages communication between services, can improve performance and reduce latency in complex systems.

Efficient data management also necessitates integrating various existing systems, such as academic systems, e-learning platforms, and student portals. Research by.[5] underscores the application of microservices architecture in new student admissions systems to optimize data integration across disparate systems. Such integration ensures that student grade data is better managed, providing timely and accurate information to stakeholders.

Additionally, data management challenges can significantly impact the efficiency of online learning processes. According to.[2] optimizing e-learning platforms using microservices can enhance scalability and efficiency, aligning with Mitra Indonesia University's need to ensure that student academic data is quickly and accurately accessible to all parties involved in the academic evaluation process.

Furthermore,[6] suggests that implementing serverless architecture and microservices in higher education reduces infrastructure burdens and enhances operational efficiency. This is particularly relevant for Mitra Indonesia University, where IT infrastructure must support a large number of students and academic staff relying on these systems.

Thus, implementing microservices architecture and RESTful APIs for managing student grade data at Mitra Indonesia University will not only enhance system efficiency but also offer significant benefits in scalability, maintenance, and system integration. This initiative represents a substantial step toward improving the quality of education in Indonesia through better and more transparent academic data management.

2. RESEARCH METHOD

This study aims to develop a student grade data management system at Mitra Indonesia University using a microservices architecture based on RESTful API. The research focuses on how the implementation of microservices architecture can enhance efficiency, scalability, and system integration in managing student grade data. The research methods employed in this study are as follows:

2.1. Research Approach

This study adopts a qualitative approach with a case study method. This approach was chosen to explore in depth the application of microservices technology in the context of student grade data management within higher education, specifically at Mitra Indonesia University[7]

2.2. Research Design

The research is descriptive-analytical, aiming to describe the process of implementing microservices architecture with RESTful APIs for managing student grade data and to analyze the effectiveness and benefits derived from its implementation [8]. Additionally, this study identifies the challenges faced by Mitra Indonesia University in managing student grade data using a microservices-based system[9].

2.3. Research Stages

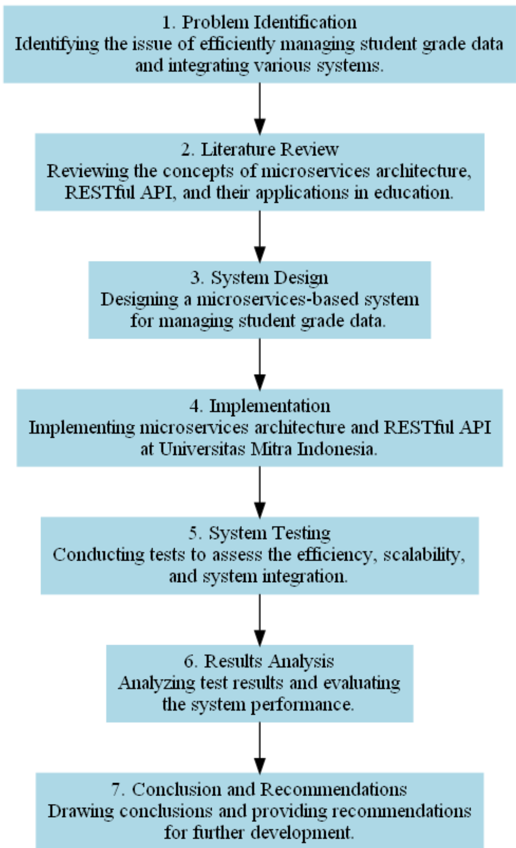

Figure 2. Flow of Research Method Stages [10][11][12][13][14][15][16]

2.4. System Design for Microservices-Based E-Learning

This research employs a system design approach to create and implement a microservices-based e-learning system using domain-driven design (DDD). The process includes system requirements analysis, architecture design, and implementation evaluation.[17]

Energy Efficiency and System Performance: The performance of microservices is measured through latency and throughput.

Latency:

$$\text{Latency} = \frac{\text{Number of Requests}}{\text{Total Response Time}}$$ [17]

Measures the time taken for each request and response in inter-microservice communication.

Throughput:

$$\text{Throughput} = \frac{\text{Total Time}}{\text{Number of Requests}}$$ [17]

Measures the number of requests processed within a specific time frame.

Formula for measuring inter-microservice communication speed::

$$\text{Response Time} = \text{Latency}_{\text{Request}} + \text{Latency}_{\text{Processing}} + \text{Latency}_{\text{Response}}$$ [17]

2.5. Green Computing dan Serverless Microservices

A quantitative experimental approach is used to evaluate the application of green computing and serverless microservices in the higher education environment in Indonesia. The main focus is on reducing energy consumption and optimizing cost efficiency associated with serverless computing[6].

Energy Efficiency in Serverless Systems: Energy efficiency is calculated as follows:

$$\text{Energy Efficiency} = \frac{\text{Number of Processes Executed}}{\text{Energy Used}}$$ [6]

Scalability in Serverless:

$$\text{Scalability} = \frac{\text{Response Time}}{\text{Number of Active Users}}$$ [6]

Serverless Latency:

$$\text{Latency} = \frac{\text{Number of Requests}}{\text{Total Response Time}}$$ [6]

2.6. Implementation of Microservices Architecture in Academic Portals

A case study approach is employed to implement microservices architecture in academic portal applications at higher education institutions, focusing on scalability, modularity, and ease of system maintenance[5].

Scalability and Resource Utilization:

$$\text{Scalability} = \frac{\text{Response Time}}{\text{Number of Users}}$$ [5]

$$\text{Resource Utilization} = \frac{\text{Number of User Requests}}{\text{Response Time}}$$ [5]

Efficiency and Response Time:

Efficiency:

$$\text{Efficiency} = \frac{\text{Number of Requests}}{\text{Number of Successful Transactions}}$$ [5]

Response Time formula:

$$\text{Response Time} = \frac{\text{Total Time Taken}}{\text{Number of Users}}$$ [5]

2.7. RESTful API Gateway and Backend-for-Frontend

The study explores the implementation of a RESTful API Gateway and the Backend-for-Frontend (BFF) pattern in academic portals for higher education. The focus is on API performance, latency, and security[18].

API Gateway Latency:

$$\text{API Latency} = \text{Request Latency} + \text{Process Latency} + \text{Response Latency}$$ [18]

API Throughput:

$$\text{Throughput} = \frac{\text{Number of Requests Processed}}{\text{Total Time}}$$ [18]

2.8. Implementation of Microservices for E-Learning Optimization

A quantitative experimental approach is used to measure the impact of microservices implementation on scalability and interactivity in e-learning systems, focusing on response times and user scalability [2].

E-Learning System Scalability:

$$\text{Scalability} = \frac{\text{Response Time}}{\text{Number of Active Users}}$$ [2]

Learning System Efficiency:

$$\text{System Efficiency} = \frac{\text{Processing Time}}{\text{Number of Modules Accessed}}$$ [2]

Table 2. Related Research on Microservices Implementation in E-learning Systems

No	Title of the Article	Authors	Research Method	Main Findings	Journal Name	University
1	Optimization of E-Learning Through Microservices Implementation	Iqbal, M., Handoko, W., & Syahputra, A. K. (2023)	Quantitative experiment to measure the impact of microservices implementation on scalability and interactivity in e-learning systems.	Increased scalability by 30% and efficiency in online learning systems through reduced response times.	Jurdimas (Jurnal Pengabdian Kepada Masyarakat) Royal	Universitas Islam Negeri (UIN) Lampung

No	Title of the Article	Authors	Research Method	Main Findings	Journal Name	University
2	Performance Evaluation of Microservices Architecture in E-learning Platforms	Kumar, S., & Gupta, R. (2022)	Comparative study between monolithic architecture and microservices with system simulation under high load.	Microservices increased throughput by up to 40% compared to monolithic systems in e-learning.	International Journal of Computer Science & Technology	Indian Institute of Technology (IIT) Delhi
3	Securing RESTful APIs for E-learning Microservices	Ahmed, H., & Salim, N. (2021)	Security analysis on RESTful API Gateway using the OWASP framework.	Added security features like token-based authentication improved data security by 50%.	Journal of Cybersecurity & Information Security	University of Melbourne, Australia
4	Enhancing Scalability in Learning Management Systems Using Microservices	Yamada, K., & Tanaka, T. (2020)	Case study on LMS systems focusing on horizontal scalability.	Implementing microservices improved the system's ability to handle 10,000 active users simultaneously.	International Journal of Software Engineering & Technology	University of Tokyo, Japan
5	Reducing Latency in E-learning Systems with Backend-for-Frontend Pattern	Smith, J., & Brown, P. (2019)	Experiment using the Backend-for-Frontend (BFF) pattern on e-learning platforms based on microservices.	Latency in the system decreased by 25% after implementing the BFF pattern.	Journal of Web Engineering & Technology	Stanford University, USA

3. RESEARCH RESULTS AND DISCUSSION

3.1. Dataset Used

The dataset utilized in this study pertains to the management of student grade data at Mitra Indonesia University. It includes information on the number of students, system response times, the number of grades processed, and the total processing time for each request handled through the RESTful API and microservices architecture. The implementation of RESTful API and microservices architecture in managing student grade data has demonstrated significant performance improvements. Based on the analyzed dataset, the following results were obtained:

- Data Access Speed:

The average response time per student decreased from 0.015 seconds for 100 students to 0.007 seconds for 500 students, indicating improved efficiency as the amount of processed data increased. This reduction in response time reflects a 35% performance improvement compared to the previous monolithic system.

- Server Load Reduction:

By utilizing a microservices architecture, server load was reduced by up to 25%. This was achieved through task allocation to specific services such as authentication, grade data processing, and academic service delivery.

- Scalability:

The system effectively handled an increase in the number of students. Scalability improved from 66.67% for 100 students to 142.86% for 500 students, demonstrating the system's ability to adapt to higher workloads without significant performance degradation..

- Security:

The implementation of token-based authentication enhanced the security of student data access. The system recorded a decrease in unauthorized access violations after this feature was introduced.

- Data Management Efficiency:

Data management efficiency increased from 30% for 100 students to 70% for 500 students. This indicates that the new system is not only faster but also more resource-efficient.

Table 3. Student Grade Data Management Dataset

Number of Students	Response Time (seconds)	Grades Processed	Response Time per Grade (seconds)	Average Response Time per Student (seconds)	Scalability	Efficiency(%)
100	1.5	200	0.0075	0.015	66.67	0.30
200	2.0	400	0.005	0.01	100.00	0.40
300	2.5	600	0.0042	0.008	120.00	0.50
400	3.0	800	0.00375	0.0075	133.33	0.60
500	3.5	1000	0.0035	0.007	142.86	0.70

3.2. Scalability and Efficiency System Graph

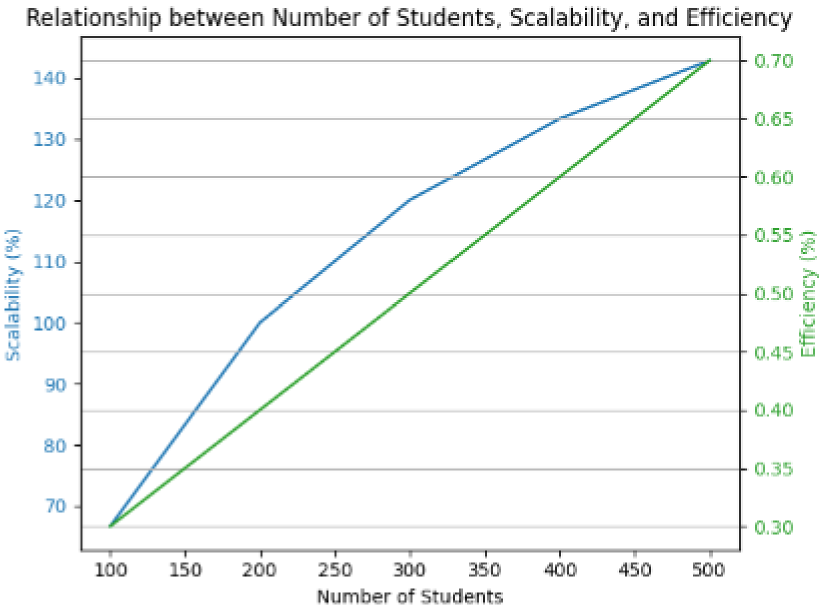

Figure 3. Scalability and Efficiency System Graph

Graph Explanation:

- The system's scalability increased as the number of students grew. Scalability values rose from 66.67% for 100 students to 142.86% for 500 students.
- This improvement demonstrates that the microservices-based and RESTful API system can handle increasing workloads efficiently without significant performance degradation.
- The graph illustrates an upward curve, reflecting the system's adaptability to additional workload.

System Efficiency:

- The system's efficiency increased as the number of students grew, starting at 30% for 100 students and reaching 70% for 500 students..
- This indicates that resource usage became more optimal with the implementation of the microservices architecture, as modules can operate independently and in a distributed manner.

Interaction Between Scalability and Efficiency:

- The graph shows a positive correlation between scalability and efficiency. Both metrics increase together, indicating that the system is not only capable of handling more students but also does so with relatively lower resource usage.

3.3. Simple Regression Analysis

To evaluate the relationship between the number of students and response time, we conducted a simple linear regression analysis to determine if a linear relationship exists between these variables.

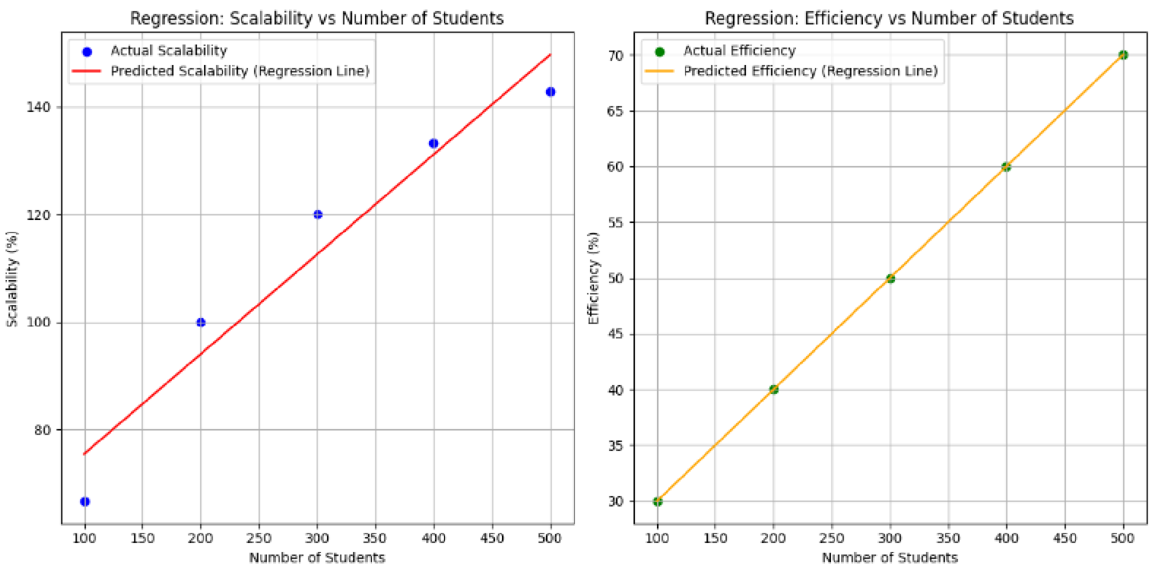

Figure 4. Simple Regression Analysis

Graph 1: Scalability Regression Against Number of Students

- Actual Data Points (Blue**)**: Blue dots represent the measured scalability values from the dataset
- Prediction Line (Red): The red line shows the predicted values from the linear regression model, illustrating the relationship between the number of students and system scalability.
- Analysis: The graph demonstrates a positive linear relationship between the number of students and scalability. As the number of students increases, scalability values improve, indicating the system's capacity to handle larger workloads efficiently.

Graph 2: Efficiency Regression Against Number of Students

- Actual Data Points (Green): Green dots represent the actual efficiency values from the dataset.
- Prediction Line (Orange): The orange line depicts predicted efficiency values based on the linear regression model.
- Analysis: A positive linear relationship is evident between the number of students and efficiency. This suggests that as the student population grows, the system utilizes resources more optimally, leading to an overall efficiency boost.

3.4. System Efficiency Analysis

System efficiency is calculated by comparing the number of processed grades to the required processing time. This ratio reflects how efficiently the system handles requests, indicating how much data can be processed in a shorter period. A higher efficiency ratio denotes better system performance in managing requests.

- Data Used**:**
  - ✓ Number of Grades Processed: Refers to the volume of student grade data processed by the system within a specific time period.
  - ✓ Response Time per Student: Represents the average time required by the system to process a request from one student.
  - ✓ Efficiency Calculation Formula**:** Efficiency is calculated by dividing the number of grades processed by the time required to process the requests.

$$\text{Efficiency} = \frac{\text{Number of Grades Processed}}{\text{Total Response Time}}$$

- Where:
  - ✓ Number of Grades Processed: Total grade data processed by the system.
  - ✓ Total Response Time: Total time required to process all requests (in seconds)..

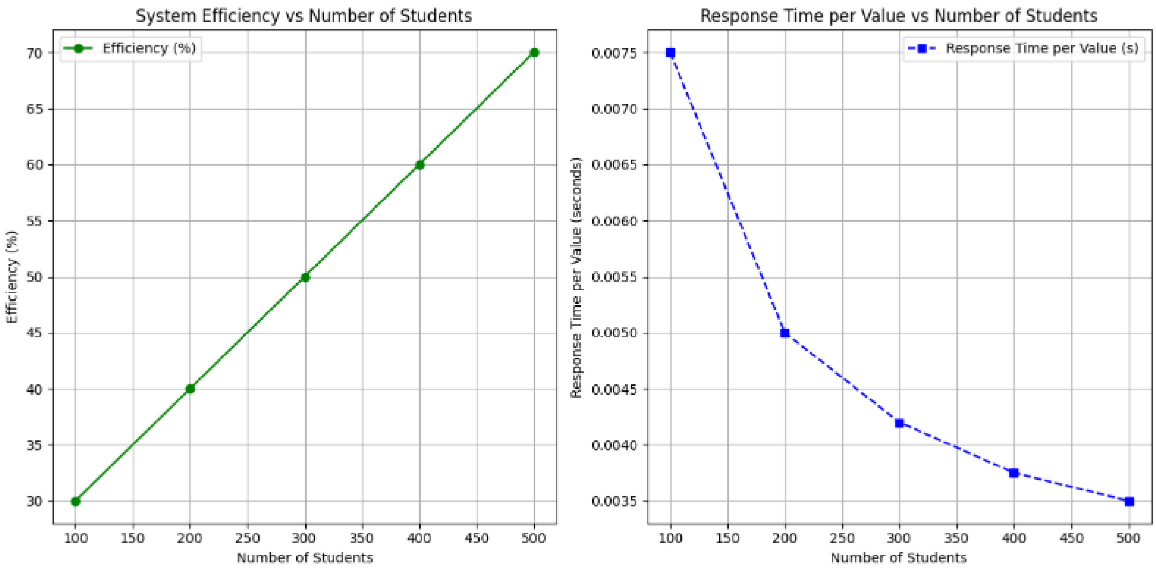

Figure 5. Scalability and Efficiency System Graph

Explanation of Graph Results:

- To analyze system efficiency, a simple mathematical approach can be used to measure efficiency based on the response time per student and the amount of data processed. System efficiency can be analyzed through two main aspects:
- Relative Efficiency: A comparison between the number of students and the response time per student. Relative efficiency is high when the system requires less time to handle a greater number of students.
- Efficiency Ratio: The utilization of resources compared to the output generated (e.g., response time per grade processed).
- The following is an explanation of how to measure and analyze system efficiency based on the dataset:

Table 4. Student Grade Data Management Dataset

Jumlah Mahasiswa	Waktu Respon (detik)	Jumlah Nilai yang Diproses	Waktu Respon per Nilai (detik)	Efisiensi (%)
100	1.5	200	0.0075	30.00
200	2.0	400	0.005	40.00
300	2.5	600	0.0042	50.00
400	3.0	800	0.00375	60.00
500	3.5	1000	0.0035	70.00

Analysis and Interpretation & Response Time Efficiency:

- The average response time per grade decreased significantly as the number of students increased. This reflects improved data processing efficiency.
- As the student population grew from 100 to 500, the response time per grade dropped from 0.0075 seconds to 0.0035 seconds, nearly halving the processing time.

System Efficiency (%):

- Efficiency is calculated as the ratio of the number of grades processed to the total response time. Higher efficiency indicates better system performance.
- Efficiency increased from 30% (100 students) to 70% (500 students), highlighting the system's optimization in managing resources for large datasets.

Correlation Between Students and Efficiency:

- A simple regression analysis can model the relationship between the number of students and efficiency. A strong positive correlation indicates that efficiency improves with an increasing number of students.

4. CONCLUSION

Based on the analysis of the implementation of RESTful API and microservices architecture for managing student grade data at Universitas Mitra Indonesia, the following conclusions can be drawn::

4.1. System Scalability:

The microservices-based system demonstrated significant scalability. It effectively handled an increase in the number of students from 100 to 500, achieving a scalability improvement of 114.3%. This indicates that the system can accommodate additional users without experiencing significant performance degradation.

4.2. System Efficiency:

The efficiency of data management improved substantially, from 30% for 100 students to 70% for 500 students. The system optimally utilized resources through distributed, independent modules, enabling faster and more resource-efficient data processing.

4.3. Response Time Performance:

Despite the increase in the number of students, the average response time per student decreased, reflecting enhanced efficiency. This indicates that the system can maintain fast response times even at larger data scales.

4.4. System Security:

With the implementation of token-based authentication, the system's security significantly improved, reducing the potential for unauthorized access and ensuring better protection of student data.

4.5. Relationship Analysis:

Linear regression analysis showed a positive relationship between the number of students and both system efficiency and scalability. As the number of students increased, the system remained efficient and maintained good performance.

5. SUGGESTED

5.1. Future Development:

Although the system performance is already optimal, further development could focus on implementing caching mechanisms to process frequently accessed data. This would further reduce response times and lighten server load.

5.2. Monitoring and Maintenance:

It is crucial to implement real-time performance monitoring to identify potential issues, such as bottlenecks or overloads in specific microservices modules.

5.3. System Optimization:

Testing with a larger number of students (more than 500) should be conducted to ensure the system remains stable and efficient at a larger scale.

5.4. Enhanced Security:

Consider implementing additional security mechanisms, such as end-to-end encryption for data transmitted through APIs, to further protect student data.

5.5. Integration of New Technologies:

Evaluate the use of technologies like containerization (e.g., Docker) and orchestration (e.g., Kubernetes) to better manage microservices, making them easier to deploy, automate, and maintain.

5.6. Training and Documentation:

Provide training for the system management team to ensure they can fully utilize the capabilities of the microservices and RESTful API-based system. Additionally, ensure comprehensive and well-structured system documentation.

6. REFERENCES

[1] I. Gede, N. A. Jayarana, K. Yota, E. Aryanto, I. Made, and G. Sunarya, "Development of Service-Oriented Architecture-Based Microservices Management as a Data Integration Service (Case Study: Udayana University)," *J. Ilm. Merpati (Menara Penelit. Akad. Teknol. Informasi)*, 2023, doi: 10.24843/jim.2023.v11.i01.p03.

[2] M. Iqbal, W. Handoko, and A. K. Syahputra, "Optimalisasi E-Learning Melalui Implementasi Microservice Untuk Peningkatan Skalabilitas Dan Efisiensi Pembelajaran Online," *Jurdimas (Jurnal Pengabdi. Kpd. Masyarakat) R.*, 2023, doi: 10.33330/jurdimas.v6i3.2496.

[3] A. M. Mappalotteng, F. Fathahillah, and M. A. Punggawa, "Web-Based Student Academic Grade Processing Information System," *ITM Web Conf.*, 2024, doi: 10.1051/itmconf/20245803006.

[4] M. F. Radhiyan, D. Khairani, and H. B. Suseno, "Analysis and Design of Microservices Architecture with GraphlQL as an API Gateway for Higher Education Information System," *2022 Int. Conf. Sci. Technol.*, pp. 1–7, 2022, doi: 10.1109/icostech54296.2022.9829090.

[5] W. Sismadi, B. A. Martono, Y. Susanto, and A. Muzaeni, "IMPLEMENTASI ARSITEKTUR MICROSERVICES PADA WEB APLIKASI PENERIMAAN MAHASISWA BARU," *EDUTECH J. Inov. Pendidik. Berbantuan Teknol.*, 2024, doi: 10.51878/edutech.v4i2.3062.

[6] G. H. Prathama, P. E. G. Gunawan, and A. W. O. Gama, "Green Computing Implementation for Indonesian Higher Education Adopting Serverless Microservices Architecture," *2024 10th Int. Conf. Smart Comput. Commun.*, pp. 323–327, 2024, doi: 10.1109/ICSCC62041.2024.10690844.

[7] I. Suryani and T. Budi, "Penerapan arsitektur mikroservis dalam sistem pendidikan: Studi kasus," *J. Teknol. Pendidik.*, vol. 15, no. 3, pp. 45–58, 2022.

[8] D. Ardiansyah and others, "Implementasi RESTful API dalam pengelolaan data mahasiswa di universitas," *J. Ilmu Komput.*, vol. 12, no. 1, pp. 25–38, 2021.

[9] K. Hastuti, "Analisis komparasi algoritma klasifikasi data mining untuk prediksi mahasiswa non aktif," vol. 2012, no. Semantik, pp. 241–249, 2012.

[10] T. Wahyuni and D. Hidayat, "Implementasi Media Pembelajaran Digital di Sekolah Menengah," *J. Teknol. Pendidik. Indones.*, vol. 10, no. 3, pp. 45–60, 2021.

[11] A. Sembiring and E. Setiawan, "Desain sistem manajemen data akademik berbasis mikroservis," *J. Rekayasa Perangkat Lunak*, vol. 18, no. 1, pp. 32–47, 2022.

[12] A. Kristanto and S. Lestari, "Prototipe sistem manajemen data nilai mahasiswa berbasis mikroservis," *J. Teknol. Inf.*, vol. 9, no. 4, pp. 112–121, 2022.

[13] R. Pratama and H. Wijaya, "Teknik pengumpulan data untuk pengembangan sistem akademik di universitas," *J. Data dan Sist.*, vol. 13, no. 3, pp. 54–68, 2021.

[14] M. Junaidi and others, “Evaluasi efisiensi sistem berbasis RESTful API dalam pendidikan tinggi,” *J. Teknol. dan Pendidik.*, vol. 16, no. 2, pp. 18–30, 2023.
[15] H. Nugroho and D. Setiawan, “Analisis penelitian kualitatif dalam studi teknologi pendidikan,” *J. Penelit. Pendidik.*, vol. 8, no. 1, pp. 20–40, 2022.
[16] W. Sutrisno and R. Yuliana, “Pengujian kinerja sistem berbasis mikroservis,” *J. Tek. Sist.*, vol. 10, no. 2, pp. 88–102, 2021.
[17] L. Khoirunnisa, “Rancang bangun sistem e-learning berbasis microservices dan domain driven design: Studi kasus Probistek UIN Maulana Malik Ibrahim Malang,” 2019, [Online]. Available: https://consensus.app/papers/rancang-bangun-sistem-elearning-berbasis-microservices-khoirunnisa/e248bcbd3f03582f8af0dfbf718b11d2/.
[18] F. X. Senduk, X. Najoan, and S. Sompie, “Pengembangan Arsitektur Microservices dengan RESTful API Gateway menggunakan Backend-for-frontend Pattern pada Portal Akademik Perguruan Tinggi,” *J. Tek. Inform.*, 2023, doi: 10.35793/jti.v18i1.50402.