Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

Designing and Implementing Microservices for Patient Appointment Queuing Systems

Widi Santoso[1], Setyawan Widyarto[2]
Magister Ilmu Komputer, Universitas Budi Luhur, Jakarta[1], Universiti Selangor, Malaysia2
swidyarto@unisel.edu.my[2]

***Abstract*— The aim of this research is to build and implement a microservices architecture for a patient appointment queuing system. The main focus of the research is how to use microservices architecture to increase the efficiency and flexibility of patient appointment queuing systems. The scope of research includes system requirements analysis, microservices architecture design, implementation, and system performance evaluation. The approach used in this research is a microservices-based system design and development approach, which includes system requirements analysis, microservices architecture design, and system performance evaluation. The research results show that using a microservices architecture can increase the efficiency, flexibility, and resilience of the patient appointment queuing system. Additionally, they can handle more work and ensure services remain available. The main conclusion is that microservices architecture can improve the performance of patient appointment queuing systems. Microservices, architecture, patient appointment queues, efficiency and flexibility are the key words.**

Keywords: microservices, architecture, patient appointment queues, efficiency, flexibility.

I. INTRODUCTION

Patient appointment queuing systems are becoming increasingly important for hospitals and clinics in the modern era to improve patient experience and operational efficiency. This is because it allows patients to make online appointments with doctors or health facilities, thereby reducing waiting times and increasing patient satisfaction (Endra et al. 2019).

However, many existing patient appointment queuing systems still face several problems, including limited flexibility, limited scalability, and service availability issues. Often, these systems have a monolithic architecture, which makes it difficult to adapt to changing user needs and desires (Haddara and Staaby 2022).

Previous studies have looked at the use of new technologies to improve the performance of patient appointment queuing systems. For example, (Bassey Ele, John Adinya Odey, N. E. Frank 2020) suggested a web-based patient appointment queuing system integrated with an SMS alert notification and explored the Internet of Things technology for patient appointment queuing systems. However, these methods still have limitations in terms of flexibility and ability to adapt to changing needs.

Microservices architecture is a promising technology to overcome these limitations. Microservices make systems more flexible, modular, and independently scalable (Newman n.d.). Several studies have shown that the application of microservices to complex systems such as supply chain management systems and e-commerce.

However, there has been no research that specifically looks at how to use microservices architecture for patient appointment queuing systems. Therefore, this research aims to design and implement this microservices architecture, as well as evaluate how the resulting system functions. This method is expected to increase the flexibility, efficiency and availability of the patient appointment queue system.

II. LITERATURE REVIEW

Microservices, which is a software architecture, has become an important part of modern system design, such as patient appointment queuing systems. (M Rizki, A N Fajar 2021) investigated the use of custom microservices in patient queuing systems. Their study shows that combining tasks such as registration, scheduling, and notifications into different services can improve efficiency and reduce latency. This increases the responsiveness and efficiency of the queuing system to meet user needs.

One of the main challenges in implementing microservices is integration and security, according to (Iqbal, Naqvi, and Arif 2020). To overcome this problem, they suggest the use of API gateways and strong authentication techniques. Patient appointment queuing systems require special attention to data security and integration between services to ensure successful implementation.

Next, according to (Söylemez and Tekinerdogan 2022) they discuss the advantages and disadvantages of microservices. They state that although microservices offer better maintenance and problem handling, they have disadvantages, such as higher management complexity and higher implementation costs. When deciding whether microservices are the right choice for a

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

patient appointment queuing system, these should be taken into consideration.

(Ramu 2023) provide a practical example of implementing microservices in a large hospital. This study shows that microservices successfully improve the efficiency and integration of queuing systems, proving that this method has real benefits in the real world.

(Panagiotis 2023) emphasize that microservices enable better scalability and focused management of every aspect of the queuing system. They also emphasize how important modular design is for overcoming emerging problems and optimizing overall system performance.

According to this literature, building and implementing microservices inpatient appointment queuing systems has many benefits, such as increasing efficiency, flexibility, and scalability. Microservices have been proven to improve system performance and manageability, although they face issues such as integration, security, and cost. By showing evidence of the success of queuing systems, the studies reviewed support the implementation of microservices. It also provides a comprehensive overview of the advantages and disadvantages of this architecture.

III. METHODS

To ensure the system runs efficiently, safely, and reliably, many quality factors are important to consider when designing and implementing microservices for patient appointment queuing systems. To achieve this goal, here are some non-functional requirements (NFRs) or quality attributes that must be considered:

A. Scalability

Scalability is a system's ability to use more resources to handle more workloads. Scalability means that each service, such as registration, scheduling, and notifications, can be scaled independently within the context of the design and implementation of microservices for patient appointment queuing systems. This allows the system to handle the increasing number of users and requests without disrupting the performance of other services. Each microservice can be easily extended to meet increasing needs by using technologies such as containerization and orchestration (for example, Docker and Kubernetes).

Additionally, microservices architecture allows for the addition of resources according to demand. For example, during peak operating hours, appointment scheduling services can be added to handle spikes in demand, while other less used services remain at standard capacity. This improves resource usage efficiency and ensures that the system remains responsive and reliable.

B. Maintability

The concept of maintainability refers to how easily system maintenance and repairs can be performed. Each service in a microservices architecture is considered a separate module that has a specific function. This means that fixes or changes to one service do not affect other services, so the risk of errors spreading throughout the system is minimized. This is especially important for patient appointment queuing systems because it ensures that bugs or feature updates can be fixed quickly without disrupting other services.

Microservices support continuous deployment and continuous development. Development teams can work on multiple services at once and release updates independently. This reduces downtime and enables rapid response to user needs or problems, increasing reliability and quality of service.

C. Security

Due to the large number of entry points that need to be protected, the security of microservices systems becomes more complex. Each microservice must have strong authorization and authentication mechanisms to ensure that only authorized people can access certain data or services. This is especially important in patient appointment queuing systems to protect patient personal information.

API gateways can improve security by serving as a primary gateway that verifies each request before it is forwarded to the appropriate microservices. Additionally, implementing security protocols such as OAuth and SSL/TLS, as well as data encryption during transit and storage, can help keep communications between services secure. To quickly detect and respond to security threats, good monitoring and logging are also important.

D. Performance

In the context of microservices, performance refers to a system's ability to respond to user requests quickly and efficiently. Each service can be optimized separately and the workload can be distributed more evenly by dividing functionality into smaller services. This means that each service, such as registration and scheduling, can be processed simultaneously in the patient appointment queuing system, reducing user waiting times.

Microservices architecture allows the use of more efficient load balancing and caching techniques to improve performance. The system can avoid bottlenecks and keep response speed high by distributing requests among the various available service instances. Additionally, continuous monitoring and performance analysis enables rapid discovery of bottlenecks, allowing immediate improvements to be made.

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

E. Interoperability

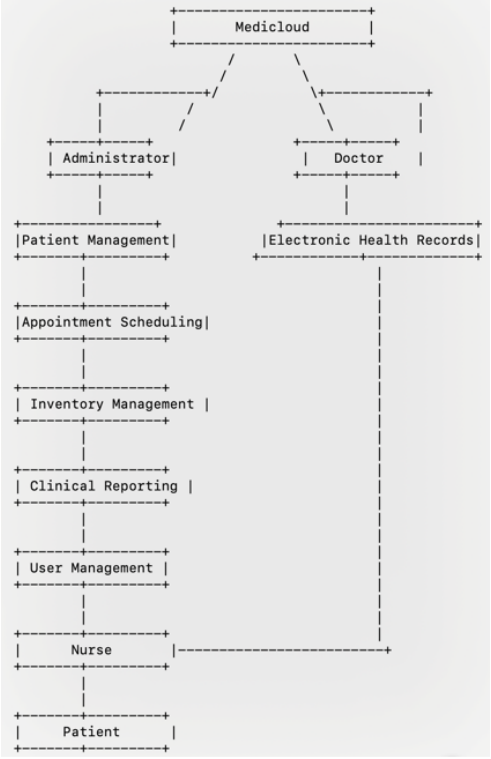
The ability of a system to communicate and function well with other systems is called interoperability. Different services communicate through well-defined APIs in a microservices architecture. This allows easy integration with external systems such as hospital management systems and electronic health records (EHR). To ensure that patient data can be easily accessed and updated across multiple platforms, this is essential for a patient appointment queuing system.

Microservices can ensure that data can be transferred safely and efficiently between systems by using standard protocols such as RESTful APIs or gRPC. Additionally, interoperability allows replacing or upgrading specific services without disrupting the entire system, providing greater flexibility and adaptability to meet changing business and technology needs.

3.1 Functional Modules

Regulatory compliance is critical when building and implementing microservices for patient appointment queuing systems. To protect the security and privacy of patient data, patient appointment queuing systems must comply with Health Insurance Portability and Accountability Act (HIPAA) standards, which involve implementing security measures such as data encryption, strict access controls, and audit logs to track user activity. By complying with HIPAA, patient appointment queuing systems can ensure that sensitive patient data remains safe on mobile.

Successful implementation of microservices relies heavily on choosing the right technology stack. The use of powerful and flexible frameworks, such as Node.js for server-side, React.js for front-end, PostgreSQL for databases, and Docker for containerization, enables the development of responsive and scalable systems. The system must also be deployed in the cloud and supervised using Kubernetes, which offers automated orchestration for container management. Patient appointment queuing systems can leverage the benefits of cloud computing such as dynamic scalability, efficient resource management, and increased availability and reliability by using this technology.



3.2 Architecture System

3.2.1 Overview

The Medicloud Clinic application uses a microservices-based architecture consisting of several independent services that are managed and developed separately. Each microservice communicates via RESTful APIs and, in some cases, uses gRPC for efficient communication.

3.2.2 Components

A. Backend

• Node.js with Express.js: The ultimate framework for building RESTful APIs.

• Prisma: ORM for interaction with PostgreSQL databases.

• gRPC: For communication between services that require low latency.

• Docker: Containerization of services for ease of deployment and scalability.

• Kubernetes: Container orchestration for large-scale management and deploy automation.

B. Frontend

• React.js with Next.js: To build responsive user interfaces and support server-side rendering.

• React: UI components for consistency and ease of development.

3.2.3 Databases

• PostgreSQL: Relational database for structured data storage.

• MinIO: Object storage for files such as medical reports and test results.

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

3.2.4 Security

• JSON Web Tokens (JWT): For authentication and authorization.

• OAuth 2.0: For third-party login integration.

• Encryption: All data is encrypted during storage and transmission.

3.2.5 External Integrations

• Laboratory API: For retrieving laboratory test results.

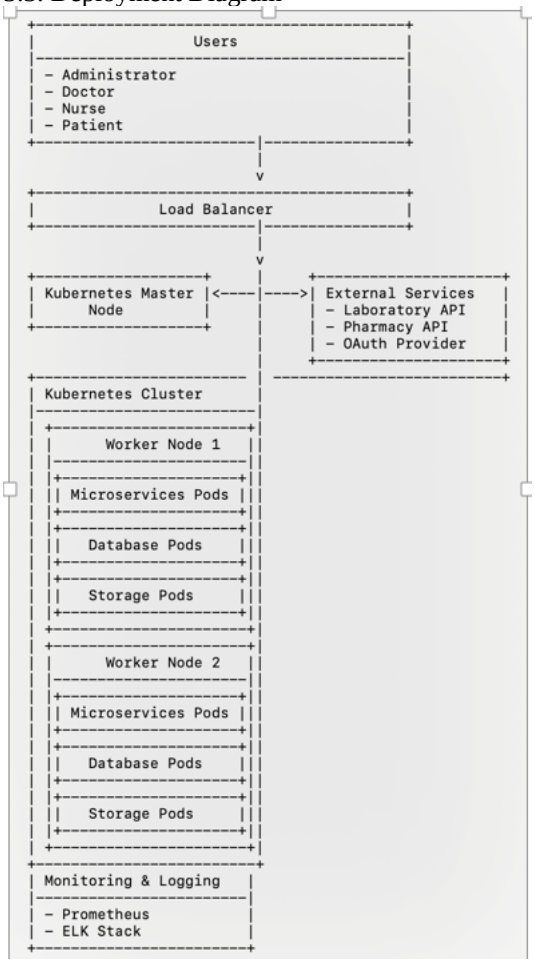
•Pharmacy API: For medication prescription management.

• FHIR Compliance: Ensures interoperability with other health systems.

• BPJS Health PCare API

• One Healthy API

3.3. Deployment Diagram



3.4. Design Interface

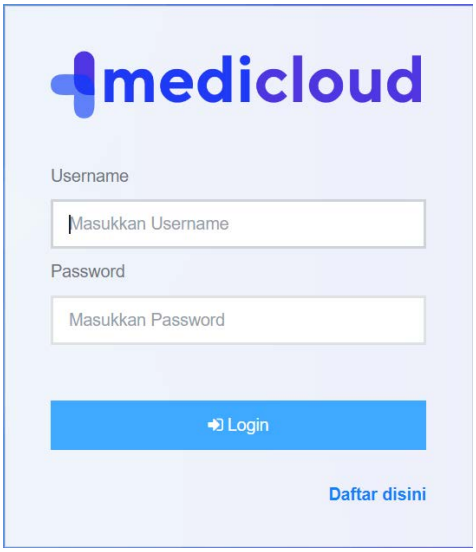


Figure 1 Login Medicloud

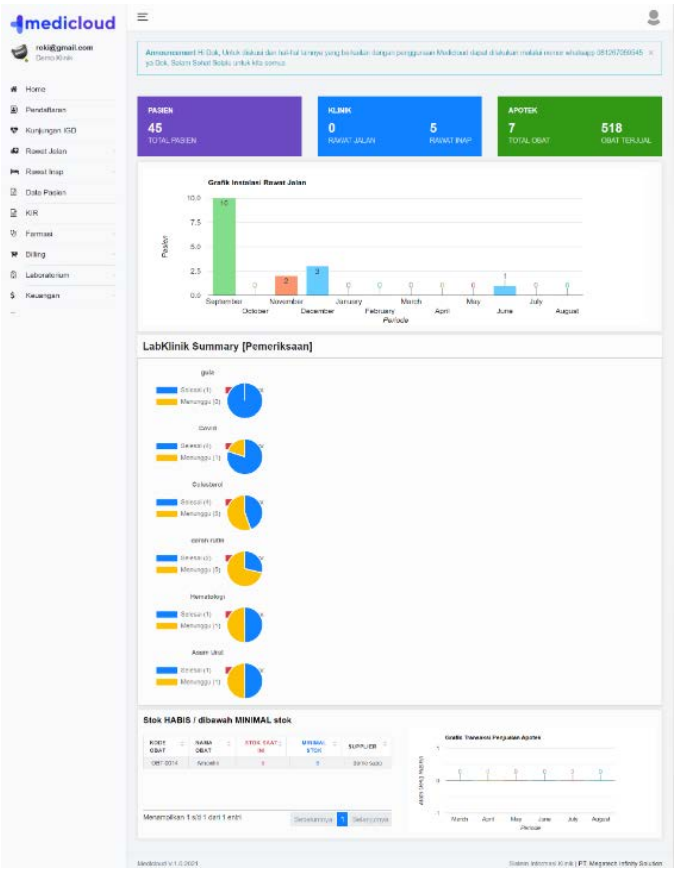


Figure 2 Dashboard Medicloud

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

IV. CONCLUSION

This document summarizes the proposed software architecture for the Medicloud Clinic application. With a modular, microservices-based approach, the use of modern technologies, and a focus on security and performance, this architecture is designed to ensure flexible, scalable, and secure applications for clinic needs.

REFERENCES

Bassey Ele, John Adinya Odey, N. E. Frank, Ideba Mboto Ekinya. 2020. “A Web-Based Medical Appointment Scheduling with SMS Alert Notification System.” 8(6). doi: 10.14738/tmlai.86.9098.

Endra, Febri, Budi Setyawan, Stefanus Supriyanto, and Feny Tunjungsari. 2019. “Medical Staff Services Quality to Patients Satisfaction Based on SERVQUAL Dimensions.” 8(1):51–57. doi: 10.11591/ijphs.v8i1.17066.

Haddara, Moutaz, and Anne Staaby. 2022. “RFID Applications for Patient Safety in the Healthcare Sector.” (April). doi: 10.4018/978-1-7998-9198-7.ch009.

Iqbal, Muhammad Waseem, Muhammad Raza Naqvi, and Khawaja Sarmad Arif. 2020. “IoT Based Remote Patient Monitoring System.” (November). doi: 10.1109/DASA51403.2020.9317213.

M Rizki, A N Fajar, A. Retnowardhani. 2021. “Designing Online Healthcare Using DDD in Microservices Architecture Designing Online Healthcare Using DDD in Microservices Architecture.” doi: 10.1088/1742-6596/1898/1/012010.

Newman, Sam. n.d. *Building Microservices*. O’Reilly Media.

Panagiotis, D. 2023. “ClinApp : A Microservices-Based Platform for Efficient Medical Visit Scheduling.” 0:1–2. doi: 10.3233/SHTI230499.

Ramu, Vivek Basavegowda. 2023. “Performance Impact of Microservices Architecture.” 3(6).

Söylemez, Mehmet, and Bedir Tekinerdogan. 2022. “Challenges and Solution Directions of Microservice Architectures: A Systematic Literature Review.”

First A. Author Widi Santoso is a forward-thinking and inventive professional, currently serving as IT Management at a prominent private company. With a keen eye for technological advancement, Widi has been pivotal in steering his organization towards embracing modern IT solutions that enhance operational efficiency and drive growth.

Second. Author Setyawan Widyarto, M.Sc., Ph.D. is a distinguished academic and researcher in the field of Information Technology. With a robust educational background and a profound dedication to advancing knowledge, Dr. Widyarto has made significant contributions to both academia and industry.