Studi Kasus Penerapan API Gateway sebagai Titik Masuk Terpusat pada Sistem Microservices

Dika Setiyawan[1*], Besarajanu Ariyanto[2], Maulana Bintang Hayuka[3], Muhammad Faza Wildani[4]

[1]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[1*]240103129@mhs.udb.ac.id

[2]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[2]240103127@mhs.udb.ac.id

[3]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[3]240103136@mhs.udb.ac.id

[4]Teknik Informatika/Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[4]240103140@mhs.udb.ac.id

Abstrak **Arsitektur microservices telah menjadi pendekatan yang semakin populer dalam pengembangan sistem perangkat lunak modern, terutama untuk aplikasi berskala besar yang membutuhkan skalabilitas dan fleksibilitas tinggi. Salah satu komponen krusial dalam ekosistem microservices adalah API Gateway, yang berfungsi sebagai titik masuk terpusat bagi seluruh permintaan klien ke berbagai layanan. Artikel ini menyajikan studi kasus penerapan API Gateway pada delapan domain sistem, meliputi sistem informasi penjualan online, aplikasi Point of Sale (POS), portal akademik perguruan tinggi, sistem integrasi layanan asuransi, aplikasi payment gateway pada layanan pembayaran digital, sistem manajemen rumah sakit, platform IoT smart city, dan platform edukasi online. Melalui analisis komparatif dari berbagai implementasi tersebut, artikel ini mengidentifikasi peran, manfaat, tantangan, serta pola terbaik dalam penerapan API Gateway pada arsitektur microservices. Hasil kajian menunjukkan bahwa API Gateway secara konsisten berkontribusi pada peningkatan keamanan, efisiensi komunikasi antar layanan, dan kemudahan pengelolaan sistem secara keseluruhan, dengan adaptasi fungsi yang disesuaikan terhadap karakteristik beban kerja masing-masing domain.**
Kata kunci API Gateway, Arsitektur Microservices, REST API, Skalabilitas, Integrasi Sistem

Abstract **Microservices architecture has become an increasingly popular approach in modern software development, particularly for large-scale applications that require high scalability and flexibility. One of the crucial components in the microservices ecosystem is the API Gateway, which serves as a centralized entry point for all client requests to various services. This article presents case studies on the implementation of API Gateways across eight system domains, including online sales information systems, Point of Sale (POS) applications, university academic portals, insurance service integration systems, payment gateway applications for digital payment services, hospital management systems, smart city IoT platforms, and online education platforms. Through a comparative analysis of these implementations, this article identifies the roles, benefits, challenges, and best practices in deploying API Gateways within microservices architectures. The results indicate that the API Gateway consistently contributes to improved security, more efficient inter-service communication, and easier overall system management, with functional adaptations tailored to the workload characteristics of each domain.**
Keywords **API Gateway, Microservices Architecture, REST API, Scalability, System Integration, IoT, Healthcare Information System, E-Learning**

I. PENDAHULUAN

Perkembangan teknologi informasi yang pesat telah mendorong transformasi besar dalam cara sistem perangkat lunak dirancang dan dikembangkan. Dalam konteks bisnis digital, terutama pada sektor e-commerce, sistem yang digunakan harus mampu mengelola transaksi dalam jumlah besar dengan efisiensi tinggi dan keandalan yang terjaga [1]. Selama bertahun-tahun, pengembangan aplikasi didominasi oleh arsitektur monolitik, di mana tim pengembang membuat satu aplikasi besar yang melakukan semua yang diperlukan untuk kebutuhan bisnis. Kelemahan dari sistem ini mulai muncul ketika aplikasi monolitik diperbarui dan dikembangkan, meningkatkan ukuran dan jumlah fitur [2]. Arsitektur monolitik yang menggabungkan seluruh aplikasi dalam satu kesatuan cenderung menghambat pengembangan, pembaruan, dan pemeliharaan sistem secara efektif, serta mengonsumsi banyak sumber daya server [1].

Pengembangan software dengan arsitektur monolith sangat sulit dikembangkan ketika skala aplikasi menjadi besar karena dibutuhkan waktu bagi developer untuk memahami kode dan melakukan kompilasi aplikasi [3]. Sebagai respons atas keterbatasan tersebut, arsitektur microservices mulai dipertimbangkan sebagai solusi utama [1] Dalam pendekatan ini, aplikasi dipecah menjadi layanan-layanan kecil yang independen dan saling berkomunikasi melalui REST API [1]. Arsitektur microservices memungkinkan pengembangan aplikasi yang lebih modular, di mana setiap bagian dari sistem berfungsi secara independen dan lebih mudah untuk dikembangkan serta dipelihara [4]. Namun, adopsi microservices juga membawa kompleksitas baru dalam pengelolaan komunikasi antar layanan. Ketika konsep microservices mulai populer, API Gateway menjadi komponen standar untuk integrasi di lapisan aplikasi atas [2]. API Gateway berperan sebagai titik masuk tunggal yang menerima seluruh permintaan dari klien eksternal, menangani autentikasi, otorisasi, dan mendistribusikannya ke layanan internal yang sesuai [1]

II. METODOLOGI PENELITIAN

Penelitian ini menggunakan pendekatan studi kasus komparatif (comparative case study) untuk mengidentifikasi peran, manfaat, tantangan, dan pola penerapan API Gateway pada arsitektur microservices di berbagai domain sistem. Pemilihan pendekatan ini didasarkan pada kebutuhan untuk memahami bagaimana satu konsep arsitektural yang sama diterapkan secara berbeda sesuai dengan karakteristik dan kebutuhan masing-masing domain bisnis.Tahapan penelitian dilakukan melalui empat langkah utama sebagai berikut. Studi Literatur. Tahap ini dilakukan untuk membangun kerangka konseptual mengenai arsitektur microservices dan peran API Gateway di dalamnya, mencakup definisi, fungsi inti (request routing, autentikasi dan otorisasi, rate limiting, agregasi respons, dan translasi protokol), serta posisi API Gateway dibandingkan dengan reverse proxy biasa maupun service mesh [5]. Pengumpulan Studi Kasus. Penelitian ini mengumpulkan lima studi kasus penerapan API Gateway pada domain yang berbeda, yaitu: (a) sistem informasi penjualan online (e-commerce) [1], (b) portal akademik perguruan tinggi dengan pola Backend-for-Frontend (BFF) [2], (c) aplikasi Point of Sale (POS) [3], (d) sistem integrasi layanan asuransi [4], dan (e) aplikasi payment gateway pada layanan pembayaran digital [6]. Kelima studi kasus dipilih karena merepresentasikan karakteristik beban kerja dan kebutuhan keamanan yang berbeda, mulai dari sistem transaksional volume tinggi (e-commerce dan POS), sistem dengan multi-klien heterogen (portal akademik), sistem dengan kebutuhan integrasi lintas-organisasi yang ketat terhadap keandalan data (asuransi), hingga sistem yang menuntut akurasi routing dan keamanan transaksi finansial lintas bank (payment gateway). Analisis Komparatif. Setiap studi kasus dianalisis berdasarkan empat dimensi yang konsisten, yaitu: (i) struktur arsitektur dan pola penempatan API Gateway, (ii) fungsi yang diimplementasikan (routing, autentikasi, rate limiting, agregasi, atau transformasi protokol), (iii) manfaat yang dilaporkan terhadap skalabilitas, keamanan, dan efisiensi komunikasi antar layanan, dan (iv) tantangan implementasi yang dihadapi, termasuk potensi single point of failure, kompleksitas operasional, dan masalah versioning [5]. Sintesis dan Identifikasi Pola Terbaik. Hasil analisis dari kelima studi kasus kemudian disintesis untuk mengidentifikasi pola-pola umum (common patterns) dan praktik terbaik (best practices) yang dapat digeneralisasi dalam penerapan API Gateway pada arsitektur microservices, terlepas dari domain spesifiknya. Sumber data dalam penelitian ini berasal dari dokumentasi teknis, hasil implementasi, dan laporan penelitian terkait pada masing-masing studi kasus, yang kemudian dilengkapi dengan kajian literatur sekunder dari sumber primer berupa artikel jurnal dan tinjauan sistematis mengenai pola-pola API Gateway pada arsitektur microservices secara umum.

III. HASIL DAN PEMBAHASAN

A. Perbandingan Penerapan API Gateway pada Lima Domain

Berdasarkan hasil studi kasus pada kelima domain, API Gateway secara konsisten ditempatkan sebagai titik masuk tunggal (single entry point) yang berada di antara klien eksternal dan kumpulan

layanan internal, sejalan dengan pola arsitektural yang dijelaskan secara umum dalam literatur [5]. Namun demikian, terdapat perbedaan penekanan fungsi pada setiap domain sebagaimana dirangkum pada Tabel 2. Pada sistem informasi penjualan online, API Gateway berperan utama dalam menangani volume permintaan yang tinggi serta memastikan distribusi beban (load balancing) yang efisien ke layanan-layanan seperti katalog produk, keranjang belanja, dan pembayaran [1]. Penerapan ini menitikberatkan pada peningkatan skalabilitas horizontal dan pengurangan beban komputasi pada sisi klien, mengingat karakteristik transaksi e-commerce yang fluktuatif dan dapat melonjak tajam pada periode tertentu. Pada portal akademik perguruan tinggi, API Gateway diimplementasikan bersama pola Backend-for-Frontend (BFF), di mana setiap jenis klien (web, mobile) memiliki lapisan perantara yang disesuaikan dengan kebutuhan data dan performanya masing-masing [2]. Pola ini sejalan dengan konsep BFF yang memisahkan kepentingan lintas-layanan (cross-cutting concerns) seperti keamanan dan kontrol lalu lintas pada API Gateway, dari kepentingan spesifik klien yang ditangani oleh lapisan BFF [5]. Pendekatan ini memungkinkan portal akademik melayani kebutuhan data yang berbeda antara aplikasi mahasiswa, dosen, dan administrasi tanpa duplikasi logika pada lapisan layanan inti. Pada aplikasi Point of Sale, API Gateway lebih menitikberatkan pada efisiensi komunikasi waktu nyata (real-time) antara perangkat kasir dan layanan backend, termasuk manajemen transaksi dan sinkronisasi data inventori [3]. Karakteristik POS yang membutuhkan respons cepat dan keandalan tinggi pada saat transaksi berlangsung menjadikan fungsi routing dan aggregation pada API Gateway sebagai elemen kritis untuk meminimalkan latensi pada sisi pengguna. Pada sistem integrasi layanan asuransi, API Gateway berfungsi sebagai lapisan integrasi yang menjembatani berbagai layanan internal dengan mitra eksternal, dengan penekanan lebih besar pada keamanan, validasi data, dan konsistensi proses bisnis lintas sistem [4]. Hal ini konsisten dengan karakteristik industri asuransi yang memiliki regulasi ketat terhadap keamanan data dan integritas transaksi, sehingga fungsi autentikasi, otorisasi, dan audit menjadi prioritas utama dibandingkan domain lainnya. Pada aplikasi payment gateway, API Gateway berperan sebagai komponen routing yang menentukan rute pengiriman data transaksi pembayaran ke bank tujuan berdasarkan kode bank penerima (beneficiary bank code), dan akan menjalankan mekanisme query berbasis prioritas apabila rute langsung ke bank tujuan tidak ditemukan [6]. Karakteristik domain ini menjadikan akurasi routing dan keandalan komunikasi lintas-bank sebagai prioritas utama, mengingat kesalahan rute dapat berdampak langsung pada keberhasilan transaksi finansial nasabah. Hal ini menempatkan payment gateway pada posisi yang relatif unik dibandingkan keempat domain lainnya, karena API Gateway tidak hanya berfungsi sebagai titik masuk dan keluar layanan, tetapi juga mengambil peran pengambilan keputusan routing yang bersifat kondisional dan bergantung pada data transaksi itu sendiri.

Tabel 2. Perbandingan Penekanan Fungsi API Gateway pada Delapan Domain

Domain	Fungsi Utama Yang Ditekankan	Tantangan Utama
E-commerce	Load balancing, routing volume tinggi	Skalabilitas saat lonjakan trafik
Portal Akademik	Backend-for-Frontend, multi-klien	Kompleksitas pemeliharaan BFF
POS	Routing real-time, agregasi transaksi	Latensi dan keandalan transaksi
Asuransi	Keamanan, validasi, integrasi lintas-sistem	Konsistensi data lintas-organisasi
Payment Gateway	Routing transaksi berbasis kode bank, validasi rute	Akurasi dan keandalan routing lintas-bank
Manajemen Rumah Sakit	Autentikasi berlapis, integrasi HL7/FHIR, audit trail	Kepatuhan regulasi data kesehatan dan keamanan

		akses rekam medis
IoT Smart City	Agregasi data sensor, translasi protokol MQTT ke REST, rate limiting	Volume data tinggi dari ribuan sensor dan heterogenitas protokol perangkat
Platform Edukasi Online	Caching konten, routing adaptif berdasarkan peran pengguna, skalabilitas video streaming	Lonjakan pengguna serentak saat jadwal kelas dan manajemen sesi belajar lintas perangkat

B. Manfaat yang Konsisten Ditemukan

Meskipun penekanan fungsinya berbeda, terdapat tiga manfaat yang secara konsisten muncul pada kelima studi kasus. Pertama, peningkatan keamanan melalui sentralisasi mekanisme autentikasi dan otorisasi pada satu titik kontrol, yang menghilangkan kebutuhan setiap layanan untuk mengimplementasikan logika keamanan secara terpisah [1][4]. Hal ini sejalan dengan temuan bahwa sentralisasi kebijakan keamanan pada API Gateway mengurangi duplikasi logika dan memastikan konsistensi penerapan kontrol akses di seluruh layanan [5]. Kedua, efisiensi komunikasi antar layanan, yang dicapai melalui fungsi agregasi respons (response aggregation) dan transformasi protokol, sehingga klien tidak perlu melakukan banyak panggilan langsung ke berbagai layanan backend [2], [3]. Ketiga, kemudahan pengelolaan sistem secara keseluruhan, karena perubahan pada layanan internalbaik penambahan, pembaruan, maupun penghapusandapat dilakukan tanpa memengaruhi kontrak antarmuka yang diekspos ke klien, selama API Gateway tetap menjaga konsistensi routing dan versioning [4], [5]. Konsistensi serupa juga teramati pada studi kasus payment gateway, di mana sentralisasi logika routing pada API Gateway memungkinkan penambahan rute bank baru dilakukan tanpa mengubah kontrak antarmuka yang digunakan oleh aplikasi klien [6].

C. Tantangan dalam Penerapan

Di sisi lain, kelima studi kasus juga menunjukkan tantangan yang relatif serupa. Tantangan pertama adalah risiko single point of failure, mengingat seluruh permintaan klien melewati satu komponen terpusat. Risiko ini umumnya dimitigasi melalui penerapan multiple instance API Gateway yang dikombinasikan dengan load balancer [5], sebagaimana juga teridentifikasi pada implementasi e-commerce dan POS yang menangani volume transaksi tinggi [1], [3]. Tantangan kedua adalah kompleksitas operasional yang meningkat seiring bertambahnya jumlah layanan dan kebijakan routing yang harus dikelola. Pada studi kasus portal akademik, kompleksitas ini terlihat pada kebutuhan pemeliharaan beberapa varian BFF untuk klien yang berbeda [2]. Tantangan ketiga, khususnya pada domain asuransi, adalah kebutuhan untuk menjaga konsistensi dan validitas data pada saat terjadi integrasi dengan sistem eksternal pihak ketiga, yang menuntut mekanisme validasi dan audit yang lebih ketat dibandingkan domain lainnya [4]. Tantangan keempat ditemukan pada domain payment gateway, yaitu kebutuhan untuk menjaga akurasi mekanisme routing berbasis kode bank, di mana kegagalan menemukan rute yang tepat dapat berdampak langsung pada keberhasilan transaksi finansial sehingga diperlukan mekanisme fallback berbasis prioritas yang andal [6].

D. Pola Terbaik (Best Practices)

Berdasarkan sintesis terhadap kelima studi kasus dan kajian literatur pendukung, dapat diidentifikasi beberapa pola terbaik dalam penerapan API Gateway pada arsitektur microservices:

a. Penempatan API Gateway sebaiknya dilakukan pada batas (boundary) antara klien eksternal dan layanan internal, sehingga kompleksitas topologi layanan internal dapat diabstraksi dari klien [5].
b. Untuk sistem dengan banyak jenis klien yang memiliki kebutuhan data berbeda (seperti portal akademik), kombinasi API Gateway dengan pola Backend-for-Frontend dapat meningkatkan fleksibilitas tanpa mengorbankan sentralisasi kebijakan keamanan [2], [5].
c. Untuk sistem dengan volume transaksi tinggi dan sensitif terhadap latensi (seperti e-commerce dan POS), penerapan strategi load

balancing dan caching pada level gateway terbukti efektif dalam menjaga performa sistem [1], [3], [5].

d. Untuk sistem yang menuntut kepatuhan regulasi dan keamanan data yang tinggi (seperti asuransi), API Gateway perlu diperkuat dengan mekanisme autentikasi berlapis serta audit trail yang terintegrasi dengan sistem pemantauan [4], [5].
e. Untuk sistem yang bergantung pada keputusan routing kondisional berbasis data transaksi (seperti payment gateway), API Gateway perlu dilengkapi dengan mekanisme fallback dan penentuan rute berbasis prioritas, sehingga kegagalan menemukan rute utama tidak menyebabkan kegagalan transaksi secara keseluruhan [6].
f. Pada seluruh domain, pemisahan tanggung jawab antara API Gateway (yang menangani trafik north-south dari klien ke sistem) dan service mesh (yang menangani trafik east-west antar layanan internal) dapat dipertimbangkan untuk sistem berskala besar guna mencapai kontrol yang lebih menyeluruh terhadap komunikasi microservices [5].

E. Studi Kasus Tambahan: Sistem Manajemen Rumah Sakit, IoT Smart City, dan Platform Edukasi Online

Untuk memperluas cakupan analisis, tiga domain tambahan dikaji guna mengeksplorasi penerapan API Gateway pada konteks yang berbeda dari kelima studi kasus sebelumnya. Ketiga domain ini dipilih karena mewakili tantangan arsitektural yang belum tercakup, yaitu: kepatuhan regulasi data sensitif di sektor kesehatan, heterogenitas protokol komunikasi pada sistem IoT berskala kota, dan skalabilitas distribusi konten digital pada platform edukasi daring.

Sistem Manajemen Rumah Sakit. Pada sistem informasi rumah sakit berbasis microservices, API Gateway berfungsi sebagai lapisan kontrol akses tunggal yang mengatur lalu lintas antara aplikasi klien (dokter, perawat, pasien, dan administrator) dengan layanan-layanan backend seperti rekam medis elektronik, jadwal dokter, manajemen obat, dan billing. Karakteristik domain kesehatan yang sangat sensitif terhadap kerahasiaan data mengharuskan API Gateway untuk mengimplementasikan autentikasi berlapis dengan mekanisme OAuth 2.0 dan OpenID Connect, sekaligus memastikan setiap akses ke data rekam medis dicatat dalam audit trail yang terstruktur [7]. Selain itu, integrasi dengan standar pertukaran data kesehatan seperti HL7 dan FHIR (Fast Healthcare Interoperability Resources) menempatkan API Gateway sebagai komponen translasi protokol yang mengonversi format pesan antar sistem internal dengan sistem rujukan eksternal, termasuk laboratorium dan apotek rekanan. Tantangan utama yang dihadapi pada domain ini adalah menjaga konsistensi kebijakan akses berbasis peran (Role-Based Access Control) di seluruh layanan tanpa menciptakan bottleneck performa, terutama pada kondisi beban puncak seperti jam visite dokter atau saat sistem diintegrasikan dengan perangkat medis IoT secara real-time.

Sistem IoT Smart City. Platform smart city mengintegrasikan ribuan sensor dan perangkat IoT yang tersebar di berbagai titik kota, mencakup sensor kualitas udara, kamera pengawas lalu lintas, meteran air dan listrik pintar, serta sistem pengelolaan sampah. API Gateway pada konteks ini menghadapi tantangan yang secara fundamental berbeda dari domain transaksional konvensional: volume data masuk dari sensor bisa mencapai jutaan pesan per menit dengan pola traffic yang tidak dapat diprediksi. Fungsi translasi protokol menjadi peran yang paling kritis, karena perangkat IoT umumnya berkomunikasi menggunakan protokol ringan seperti MQTT atau CoAP, sedangkan layanan backend menggunakan REST atau gRPC [8]. API Gateway bertindak sebagai jembatan protokol yang mengonversi pesan MQTT dari sensor ke format REST sebelum diteruskan ke layanan analitik, penyimpanan data, atau sistem peringatan dini. Kemampuan rate limiting pada gateway juga berperan vital untuk mencegah lonjakan data dari satu cluster sensor yang mengalami malfungsi membanjiri seluruh sistem. Tantangan utama pada domain ini adalah latency yang harus dijaga serendah mungkin untuk aplikasi kritis seperti sistem peringatan banjir atau kemacetan, di mana keterlambatan pemrosesan data sensor dalam hitungan detik dapat berdampak nyata pada respons operasional kota.

Platform Edukasi Online. Pada platform edukasi daring berskala besar yang melayani ratusan ribu pelajar secara bersamaan, API Gateway mengambil peran sentral dalam mengelola distribusi konten dan skalabilitas layanan. Platform ini terdiri dari sejumlah layanan independen, termasuk layanan manajemen kursus, streaming video materi, forum diskusi, sistem kuis dan penilaian, serta layanan notifikasi. API Gateway mengimplementasikan strategi caching respons untuk mengurangi beban pada layanan konten yang statis, serta routing adaptif yang mengarahkan permintaan berdasarkan peran pengguna, yaitu pelajar, instruktur, atau administrator, ke instansi layanan yang sesuai [9]. Lonjakan traffic yang terjadi serentak saat jadwal kelas dimulai atau saat batas pengumpulan tugas menuntut kapasitas auto-scaling yang dikendalikan oleh sinyal dari gateway. Tantangan spesifik pada domain ini adalah menjaga pengalaman streaming video yang mulus bagi ribuan pengguna bersamaan, di mana gateway harus mampu mengoptimalkan routing ke server CDN terdekat berdasarkan lokasi geografis pengguna tanpa menambah latensi yang berarti. Dari ketiga studi kasus tambahan ini, terlihat bahwa meskipun prinsip dasar penerapan API Gateway tetap konsisten, skala dan karakteristik beban kerja yang berbeda mendorong adaptasi fungsi gateway yang signifikan, terutama pada aspek translasi protokol, manajemen sesi, dan strategi caching.

IV. KESIMPULAN

Penelitian ini telah mengkaji penerapan API Gateway sebagai titik masuk terpusat pada arsitektur microservices melalui studi kasus komparatif pada delapan domain sistem yang berbeda, yaitu sistem informasi penjualan online, portal akademik perguruan tinggi, aplikasi Point of Sale, sistem integrasi layanan asuransi, aplikasi payment gateway, sistem manajemen rumah sakit, platform IoT smart city, dan platform edukasi online. Hasil analisis menunjukkan bahwa meskipun kedelapan domain memiliki karakteristik beban kerja dan kebutuhan bisnis yang berbeda, API Gateway secara konsisten berperan sebagai komponen sentral yang menyederhanakan interaksi klien dengan layanan internal, sekaligus menjadi lapisan penegakan kebijakan lintas-layanan seperti autentikasi, otorisasi, dan pengaturan lalu lintas. Tiga manfaat utama yang secara konsisten ditemukan pada seluruh studi kasus adalah peningkatan keamanan melalui sentralisasi kontrol akses, efisiensi komunikasi antar layanan melalui agregasi dan transformasi protokol, serta kemudahan pengelolaan sistem akibat berkurangnya kebutuhan koordinasi langsung antara klien dan setiap layanan backend. Di sisi lain, tantangan yang muncul juga relatif serupa pada kedelapan domain, terutama terkait risiko single point of failure dan meningkatnya kompleksitas operasional seiring bertambahnya jumlah layanan yang dikelola. Studi kasus tambahan pada sistem manajemen rumah sakit memperkuat temuan bahwa domain dengan regulasi ketat membutuhkan mekanisme audit dan autentikasi berlapis pada gateway. Domain IoT smart city menunjukkan peran translasi protokol sebagai fungsi kritis yang membedakannya dari domain transaksional konvensional. Sementara itu, platform edukasi online menegaskan pentingnya strategi caching dan routing adaptif berbasis peran untuk mengelola lonjakan pengguna serentak. Perbedaan penekanan fungsi pada masing-masing domain menunjukkan bahwa penerapan API Gateway bersifat adaptif terhadap karakteristik domain, tanpa mengubah prinsip dasarnya sebagai titik masuk terpusat. Temuan ini menegaskan bahwa keberhasilan penerapan API Gateway tidak hanya bergantung pada pemilihan teknologi, tetapi juga pada penyesuaian pola arsitektural terhadap kebutuhan spesifik organisasi. Penelitian selanjutnya dapat diarahkan pada evaluasi kuantitatif terhadap dampak performa API Gateway pada masing-masing domain, serta eksplorasi lebih lanjut mengenai integrasi API Gateway dengan service mesh pada sistem berskala lebih besar.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Universitas Duta Bangsa Surakarta atas dukungan fasilitas dan ekosistem akademik yang telah memungkinkan terlaksananya penelitian ini. Terima kasih juga kepada dosen pengampu mata kuliah yang telah membimbing dan memberikan masukan konstruktif selama proses penulisan. Kepada seluruh

rekan Kelompok 8 yang telah berkontribusi penuh dalam setiap tahap penelitian ini, mulai dari pengumpulan literatur, analisis studi kasus, hingga penyusunan naskah akhir, penulis menyampaikan apresiasi yang setulus-tulusnya. Semoga hasil penelitian ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan di bidang arsitektur perangkat lunak, khususnya dalam penerapan API Gateway pada sistem microservices.

REFERENSI

[1] R. Fadillah Nurrokhim *et al.*, “Penerapan Arsitektur Microservices pada Sistem Informasi Penjualan Online untuk Meningkatkan Skalabilitas dan Kinerja Sistem,” vol. 4, no. 3, pp. 37–47, 2025, doi:10.55606/jtmei.v4i3.5825.

[2] F. X. Senduk, X. B. N. Najoan, and S. R. U. A. Sompie, “Pengembangan Arsitektur Microservices dengan RESTful API Gateway menggunakan Backend- for-frontend Pattern pada Portal Akademik Perguruan Tinggi.” [Online]. Available: https://ejournal.unsrat.ac.id/index.php/informatika

[3] S. Saidah, Latipaturachmaniah, and R. Syaban, “IMPLEMENTASI ARSITEKTUR MICROSERVICES PADA APLIKASI POINT OF SALE TOKO FLYOVER STIKER,” Sep. 2023. [Online]. Available: https://flyoverstiker.online/,

[4] A. Hidayat, H. Prasetyo Utomo, and Y. Prihadi, “Arsitektur MicroServices dalam Integrasi Sistem untuk Peningkatan Layanan Asuransi Jasa Raharja,” Nov. 2025.

[5] A. Neelan, "The Evolving Role of API Gateways in Scalable Microservices Architecture," International Journal of Emerging Trends in Computer Science and Information Technology, vol. 6, no. 4, pp. 4–15, Oct. 2025, doi: 10.63282/3050-9246.IJETCSIT-V6I4P102.

[6] E. Julio and M. A. I. Pakereng, “Implementasi API Payment Gateway Menggunakan Arsitektur Microservice,” Jurnal Informatika, vol. 8, no. 2, pp. 123–130, Aug. 2021, doi: 10.31294/ji.v8i2.10590.

[7] P. Jayathissa and R. Hewapathrana, “HAPI-FHIR Server Implementation to Enhancing Interoperability among Primary Care Health Information Systems in Sri Lanka: Review of the Technical Use Case,” *European Modern Studies Journal*, vol. 7, no. 6, pp. 225–241, Feb. 2024, doi: 10.59573/emsj.7(6).2023.23.

[8] D. Z. Fawwaz, S. H. Chung, C. W. Ahn, and W. S. Kim, “Optimal Distributed MQTT Broker and Services Placement for SDN-Edge Based Smart City Architecture,” *Sensors*, vol. 22, no. 9, May 2022, doi: 10.3390/s22093431.

[9] D. Ely Kurniawan and A. Dzikri, “Development of a Prototype for Microservices Architecture and API Gateway Integration in an eLearning Platform,” European Alliance for Innovation n.o., Feb. 2024. doi: 10.4108/eai.7-11-2023.2342930.