JURNAL LOCUS: Penelitian & Pengabdian
Volume 2 No. 10 Oktober 2023
E-ISSN 2829-7334| P-ISSN 2829-5439
Hompage: https://locus.rivierapublishing.id/index.php/jl

REENGINEERING ARSITEKTUR MONOLITHIC KE MICROSERVICES PADA WEBSITE MANAGEMENT CONTENT MQTV

Daffa Putra Permana[1], Herdi Ashaury[2], Puspita Nurul Sabrina[3]
Universitas Jenderal Achmad Yani
dafaputra876@gmail.com

Abstrak:

Perkembangan teknologi memberikan banyak manfaat. Salah satu teknologi yang sudah banyak digunakan dan membantu pekerjaan masyarakat adalah internet. MQTV adalah perusahaan stasiun TV yang menyediakan konten mereka dalam bentuk digital. MQTV memiliki website CMS dalam pengelolaan konten. Namun, seiring berjalannya waktu, website MQTV dapat mengalami masalah seperti batasan skalabilitas terbatas. Penggunaan arsitektur microservices dalam pengembangan aplikasi dapat memecahkan masalah arsitektur monolithic. Tujuan penelitian melakukan reengineering arsitektur website menjadi microservice, melakukan tahapan atau fase reengineering dan membandingkan hasil load testing dari website monolith dan microservice. Metode penelitian yang dilakukan yaitu project feasibility analisis, analisi and planning, reengineering implementation dan testing and transition. Analisis kelayakan dilakukan pada website MQTV dengan hasil yaitu akan dilakukan reengineering arsitektur menjadi microservice. Analisi dan perancangan dilakukan untuk memahami struktur microservice yang akan dibuat. Reengineering implementasi dilakukan dengan membuat service-service dengan menggunakan REST api dan mencantumkan dokumentasi-dokumentasi dari hasil implementasi. Hasil dari implementasi website microservice dilakukan pegujian dengan menggunakan metode load testing dengan hasil menunjukan laju kinerja website arsitektur microservice lebih unggul dari arsitektur monolithic. metode penelitian yang di gunakan adalah Project Feasibility Analysis Mengevaluasi kebutuhan dan tujuan organisasi yang dicapai oleh sistem yang ada, produk perangkat lunak yang digunakan saat ini harus dianalisis dalam hal spesifikasi masalah termasuk tujuan, motivasi, batasan, dan aturan bisnis. hasil dari penelitian ini agar mengetahuai hasil percobaan pengujian yang dapat disimpulkan pada website CMS MQTV dengan arsitektur microservice lebih unggul dibandingkan dengan monolithic.

Kata Kunci: Pengujian Beban; MQTV; Monolitik; Layanan Mikro; Rekayasa Ulang; REST api.

Abstract:

Technological developments provide many benefits. One technology that is widely used and helps people's work is the internet. MQTV is a TV station company that provides their content in digital form. MQTV has a CMS website for content management. However, as time goes by, the MQTV website can experience problems such as limited scalability limitations. The use of microservices architecture in application development can solve the problem of monolithic architecture. The aim of the research is to reengineer the website architecture into a microservice, carry out reengineering stages or phases and compare the results of load testing from monolith and microservice websites. The research methods used were project feasibility analysis, analysis and planning, reengineering implementation and testing and transition. A feasibility analysis was carried out on the MQTV website with the result that the architecture would be reengineered into a microservice. Analysis and design are carried out to understand the structure of the microservice that will be created. Implementation reengineering is carried out by creating services using REST api and including documentation from the implementation results. The results of the implementation of the microservice website were tested using the load testing method with the results showing that the performance rate of the microservice architecture website was superior to monolithic architecture. The research method used is Project Feasibility Analysis. Evaluating organizational needs and goals

achieved by existing systems, software products currently used must be analyzed in terms of problem specifications including goals, motivations, constraints and business rules. The results of this research are to find out the results of testing experiments which can be concluded on the MQTV CMS website with microservice architecture which is superior to monolithic.

***Keywords**: Load Testing; MQTV; Monolithic; Microservices; Reengineering; REST api.*

Corresponding: Daffa Putra Permana
E-mail: dafaputra876@gmail.com

PENDAHULUAN

Perkembangan teknologi memberikan banyak manfaat. Salah satu teknologi yang sudah banyak digunakan dan membantu pekerjaan masyarakat adalah internet (Setiawan, 2018). Dari berbagai aplikasi dan fitur yang disediakan menjadikan pengguna internet terus bertambah secara signifikan, penetrasi pengguna internet di Indonesia kurang lebih mencapai 73.7 % dari total populasi, angka ini merupakan hasil survey yang dilakukan oleh APJII (Asosiasi Penyelenggara Jasa Internet Indonesia) perioda 2019-2020 (Q2).

Meningkatnya jumlah pengguna aplikasi dapat berdampak pada proses pemeliharaan aplikasi, performa aplikasi, dan kompleksitas proses pembaruan aplikasi yang dibangun dengan arsitektur monolithic (FATHONY, 2022). Salah satu kompleksitas yang sering terjadi pada arsitektur *monolithic* yaitu *resilient challenges*, dimana jika terjadi kesalahan pada saat update aplikasi atau penambahan fitur baru, semua fungsi aplikasi akan mengalami error system (Khoirunnisa, 2019). MQTV adalah perusahaan stasiun TV yang menyediakan konten mereka dalam bentuk digital (Al Ansori & Vinianto, 2020).

Namun, seiring berjalannya waktu, *website* MQTV dapat mengalami masalah seperti batasan skalabilitas terbatas. Arsitektur *microservice* menawarkan paradigma baru dalam pembangunan dan penerapan aplikasi yang memungkinkan pemisahan fungsionalitas menjadi unit yang lebih kecil dan terdefinisi dengan jelas.

Arsitektur *microservices* merupakan pegembangkan aplikasi menjadi layanan kecil, yang prosesnya berjalan sendiri dan berkomunikasi dengan mekanisme ringan seperti HTTP API (Irawan, 2021). Bisnis proses pada *microservices* disediakan melalui API, konfigurasi tedefinisi dengan baik dan juga fungsi yang disediakan untuk bisnis proses (Amiruddin et al., n.d.). Memecahkan layanan berfungsi untuk menciptakan web *services* yang *scalable*, *resilence*, dan *high-availability*

Reengineering merupakan proses analisis, perubahan dan pembuatan ulang sistem *software* yang sudah ada, dengan tujuan untuk meningkatkan fungsi, performa atau implementasinya (Vitariani, 2019). Tujuan dari proses ini adalah untuk memahami *software* yang sudah ada dan kemudian mengimplementasikannya kembali serta mempersiapkan untuk ditambahkan fungsi di masa yang akan datang (Yanuarsari, Asmadi, Muchtar, & Sulastini, 2021).

Penelitian terdahulu menjelaskan bahwa perubahan model arsitektur *monolithic* ke *microservices* mendapatkan hasil yaitu performa *website* meningkat dari sebelumnya, hasil yang ditunjukan adalah hasil perbandingan dari pengujian *load testing* pada arsitektur *monolithic* dan *microservice* (FATHONY, 2022). Penelitian terdahulu menjelaskan dengan hasil aspek kualitas resiliensi dapat dicapai pada arstiketur *microservices*, penelitian tersebut berhasil menunjukkan implementasi *microservices* untuk studi kasus sistem manajemen asosiasi/keanggotaan yang digunakan untuk membuktikan adopsi model *Microservices Migration Patterns,* lalu mengevaluasi model tersebut pada implementasi di Asosiasi Sistem Informasi Indonesia (AISINDO) (Khoirunnisa, 2019). Penelitian terdahulu menjelaskan bahwa dilakukan implementasi *micorservice* dimana layanan-layanan berdiri sendiri sehingga membuat kinerja aplikasi akan lebih cepat dan mudah dikembangkan menjadi lebih kompleks.

Berdasarkan penelitian sebelumnya kualitas *software* dapat meningkat dengan memanfaatkan arsitektur *microservice* (Putra, 2020). Penelitian ini bertujuan untuk melakukan *reengineering website* MQTV untuk meningkatan kualitas serta mengikuti panduan yang ada dalam proses *reengineering* dari *monolithic* ke *microservice*.

Masalah yang ingin dikaji lebih dalam yakni tentang bagaimana proses *reengineering* arsitektur *monolitich* ke *microservice* dan bagaimana perbandingan kinerja dari *website monolithic* dan *microservice*. Ada banyak manfaat dari penelitian yang dilakukan diantaranya yakni bagi peningkatan kinerja, skalabilitas, pemeliharaan yang lebih mudah, pengembangan yang lebih cepat dan ketersediaan yang lebih tinggi.

METODE PENELITIAN

1. Project Feasibility Analysis

Tahap pertama yang dilakukan adalah analisis kelayakan. Analisis kelayakan akan dilakukan 3 tahap yaitu:

a. Identifikasi tujuan

Tujuan dilakukannya *reengineering* pada *webiste* MQTV ini untuk meningkatkan kualitas *software*, *reengineering* yang dilakukan mengubah arsitektur *webiste* dari *monolithic* ke *microservice*.

b. Penilaian teknis

Evaluasi yang dilakukan dimulai dari *code program*, code program pada sistem *monolithic* belum memiliki API. Performa *webiste microservice* lebih unggul karna webiste terpisah-pisah menjadi kecil (Risnanto, n.d.).

c. Keputusan

Webiste dilakukan *reengineering* sesuai dengan hasil analisis kelayakan.

2. Analysis and Planning

Tahap pertama dalam *Analysis* and *planning* adalah menemukan semua kode termasuk spesifikasi kebutuhan. Setelah itu akan membahas tentang *rerequirement*, *re-design*, *re-implementation* dan *testing*.

a. Analisa Arsitektur Tekonologi

Analisa arsitektur teknologi dilakukan dengan mengidentifikasikan teknologi-teknologi yang digunakan pada sistem. Analisa teknologi dilakukan pada bagian perangkat keras dan perangkat lunak yang digunakan pada system (Permana & Dewantara, 2018). Tabel 1 dan tabel 2 merupakan informasi mengenai teknologi perangkat keras dan perangkat lunak yang digunakan pada sistem.

Tabel 1 Tekonologi Perangkat Lunak

Nama Perangkat Lunak	Spesifikasi
Bahasa Pemrograman	PHP
Backend Framework	Lumen
Database	MySQL
Frontend framework	Laravel
UI Framework	Bootstrap

Tabel 2 Teknologi Perangkat Keras

Nama Perangkat Keras	Spesifikasi
Prosesor	Intel core i5 8^{th} gen
Ram	8 GB
Disk	256 GB

b. Analisa Format Pertukaran Data

Analisa format pertukaran data akan membahas tentang format pertukara data yang akan dipakai pada website MQTV microservice. Berdasarkan hasil perbanding dari (S. Saryanto, S. Sumarsono, and N. D. Retnowati), dua format data antara JSON dan XML didapatkan hasil yaitu:

1. JSON lebih cepat pada proses serialisasi dan parsing data dibandingkan XML.
2. Ukuran data JSON lebih kecil dari pada Ukuran XML.
3. Pertukaran data dengan menggunakan format XML memiliki batas limit size sampai 31456.31 KB sedangkan JSON melebihi batas limit size XML.

Berdasarkan hasil diatas JSON lebih unggul dari XML. Oleh karena itu, pada penelitian ini pertukaran data (response data) akan menggunakan JSON

c. Pemecahan Service

Berdasarkan hasil studi literature didapatkan 3 cara untuk memecah *webiste* menjadi beberapa *service* yaitu berdasarkan kepemilikan data, berdasarkan bisnis proses dan berdasarkan fungsionalitas (Juniati, 2014). *Webiste* MQTV akan dipecah dengan cara pemecahan berdasarkan fungsionalitas karena dalam system baru fungsi-fungsi webiste dapat bertanggung jawab pada fungsinya sendiri.

Fungsi-fungsi atau fitur-fitur dari *webiste* dapat dipecah menjadi *microservice* yang terpisah. Setiap *microservice* akan bertanggung jawab atas fungsionalitas yang spesifik (Arifiansyah, 2023). Berikut adalah hasil pemecahan *service* berdasarkan fungsionalitasnya yang dapat dilihat pada Tabel 3 Pemecahan *Service*:

Tabel 3 Pemecahan *Service*

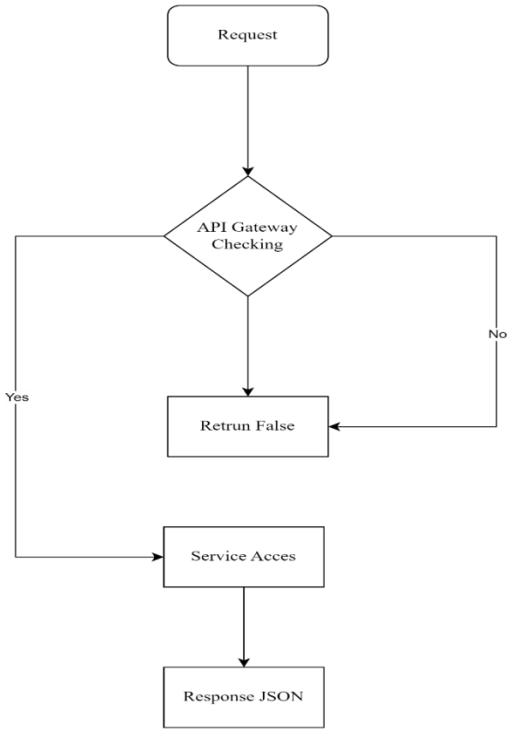
No	Nama Service	Modul
1	User	User
2	Tentang Kami	Tentang kami dan logo
3	Artikel	Artikel
4	Program Unggulan	Program Unggulan
5	Jadwal Tayang	Hari, Jam tayang dan jadwal tayang

3. Reengineering Implementation

Setelah merencanakan perancangan *microservice* pada bab ini akan berfokus pada *reengineering* implementasi. Perangkat lunak yang dirancang akan menggunakan laravel sebagai bagian *frontend* yang akan berinteraksi dengan user, dan bahasa pemrograman PHP sebagai *backend* untuk software yang dibuat. Penggunaan bahasa pemrograman yang akan digunakan *software* ini juga didukung dengan framework Lumen sebagai kerangka kerja agar pengerjaan *software* menjadi lebih terstruktur (Setyawan & Pratiwi, 2020). Dengan diintegrasikan dengan *software* XAMPP MySQL sebagai DBMS (Database Management System) yang akan digunakan sebagai Web Server penyimpanan *database*(Sentosa, 2018). Untuk penulisan *source code software* dibuat menggunakan Visual Studio Code.

a. Alur kerja microservice

Alur kerja *microservice* dimulai dari permintaan HTTP client kemudian dilakukan cek *end point* dan *method* jika hasilnya true maka *service* akan bisa diakses dan menghasilkan *return response* JSON. Untuk alur dapat dilihat pada gambar 1 dibawah.



Gambar 1 Alur kerja *microservice*

4. Testing and transition

Pada tahap *testing* and *transition* ini setiap *service* diuji dengan metode pengujian *load testing*. Tahap pertama yaitu membuat scenario *load testing*. Setelah selesai menentukan scenario maka dilanjutkan dengan tahap selanjutnya yaitu melakukan *load testing* (Riyanti & Lukma, 2021). *Load testing* disini bertujuan untuk mengukur laju kinerja *microservice* dan *monolith*ic, untuk *tool* yang digunakan disini akan memakai jmeter. Parameter yang akan digunakan dalam *load testing* ini adalah *throughput* (laju kinerja).

a. Scenario Load Testing

Jumlah maksimum pengguna yang ditetapkan dalam skenario adalah 200. Skenario ini didasarkan pada perkiraan jumlah pengguna. Sementara itu, ramp-up adalah waktu yang dibutuhkan untuk menjalankan total jumlah pengguna target. *Load testing* akan dilakukan untuk kedua *webiste* sebagai perbandingan.

Tabel 4 Scenario *Load Testing*

No	Jumlah User	Ramp-up Period
1	50	100 sec
2	100	100 sec
3	200	100 sec

HASIL DAN PEMBAHASAN

a. Identifikasi API Gateway

Setelah membuat dokumentasi API selanjut adalah mengidentifikasi API *gateway*. Tabel 5 API *Gateway* adalah hasil identifikasi API *gateway*.

Tabel 5 API *Gateway*

No	Daftar API	Operasi HTTP
1	api/hari	GET
2	api/jam_tayang	GET
3	api/jadwal_tayang	GET, POST, PUT, DELETE
4	api/program	GET, POST, PUT, DELETE

No	Daftar API	Operasi HTTP
5	api/article	GET, POST, PUT, DELETE
6	api/program_unggulan	GET, POST, PUT, DELETE
7	api/logo	GET, POST, DELETE
8	api/tentang_kami	GET, PUT
9	api/user	GET, POST, PUT, DELETE

b. Load Testing

Pada pengujian ini akan menggunakan metode *load testing* dengan bantuan *tool* jmeter. Pengujian akan dilakukan dengan 3 percobaan sesuai dengan scenario yang sudah dibuat. Pengujian akan dilakukan untuk arsitektur *monolithic* dan *microservice* sebagai perbandingan.

Percobaan Ke 1:

Tabel 6 Hasil Percobaan ke-1

Arsitektur	User	Ramp-up period	Laju Kinerja	Error
Microservice	50	100 detik	3.8/detik	0%
Monolith	50	100 detik	4.6/detik	0%

Pada percobaan ke 1 didapatkan hasil laju kinerja dari arsitektur *microservice* 3.8/detik dan status error 0%. Pada arsitektur *monolithic* didapatkan hasil laju kinerja 4.6/detik dan status error 0%.

Percobaan ke-2:

Tabel 7 Hasil Percobaan ke-2

Arsitektur	User	Ramp-up period	Laju Kinerja	Error
Microservice	100	100 detik	1.3/detik	0%
Monolith	100	100 detik	9.0/detik	0%

Pada percobaan ke 2 didapatkan hasil laju kinerja dari arsitektur *microservice* 1.3/detik dan status error 0%. Pada arsitektur *monolithic* didapatkan hasil laju kinerja 9.0/detik dan status error 0%.

Percobaan ke-3:

Tabel 8 Hasil Percobaan ke-3

Arsitektur	User	Ramp-up period	Laju Kinerja	Error
Microservice	200	100 detik	15.1/detik	0%
Monolith	200	100 detik	18.0/detik	0%

Pada percobaan ke 3 didapatkan hasil laju kinerja dari arsitektur *microservice* 15.1/detik dan status error 0%. Pada arsitektur *monolithic* didapatkan hasil laju kinerja 18.0/detik dan status error 0%.

KESIMPULAN

Berdasarkan hasil penelitian dapat disimpulkan bahwa melakukan *reengineering* arsitektur *monolithic* menjadi arsitektur *microservice* pada *webiste CMS* MQTV menunjukan hasil kualitas meningkat dari pengujian dengan menggunakan *load test* pada arsitektur *microservice* dan *monolithic*, arsitektur *microservice* yang telah dibangun dapat menangani beban yang telah ditentukan. Dari hasil yang didapatkan pada percobaan ke 1 pengujian webiste *microservice* dan *monolithic* didapatkan hasil pada 50 user *microservice* mendapatkan hasil laju kinerja 3.8/detik dan pada *monolithic* 4.6/detik, pada percobaan ke 2 pada 100 user didapatkan hasil laju kinerja *microservice* 1.3/detik dan *monolithic* 9.0/detik, pada percobaan ke 3 pada 200 user didapatkan hasil laju kinerja *microservice* 15.1/detik dan *monolithic* 18.0/detik. Sehingga dari hasil percobaan

pengujian dapat disimpulkan pada *website CMS* MQTV dengan arsitektur *microservice* lebih unggul dibandingkan dengan *monolithic*.

DAFTAR PUSTAKA

Al Ansori, Ade Nasihudin, & Vinianto, Andika. (2020). Daya Tahan Parahyangan TV Sumedang Sebagai TV Lokal Di Tengah Perkembangan Digital. Jurnal Kajian Jurnalisme, 3(2), 180–195.

Amiruddin, S., Yazid, M. T. I. Ir Setiadi, Bayu Anggorojati, S. T., Setiawan, Hermawan, TI, M., Priambodo, Dimas Febriyan, Ismoyo, Dimas Dwiki, TP, S. Tr, & cipta dilindungi Undang-undang, Hak. (n.d.). Penulis.

Arifiansyah, Muhammad. (2023). Modul Term Graph dalam Perancangan Aplikasi Quality Assessment Tools untuk Tinjauan Pustaka Sistematis.

Fathony, Ikhwan Alfath Nurul. (2022). Kontainerisasi Shibboleth Idp Sebagai Akses Manajemen Single Sign On Pada Arsitektur Microservice Sistem Enterprise Dengan Metode Load Balancing.

Irawan, Aji Karuniadi. (2021). Implementasi Arsitektur Microservice Untuk Input Nilai Praktikum Mahasiswa STMIK Akakom Yogyakarta Menggunakan Restful Api. STMIK Akakom Yogyakarta.

Juniati, Desak Putu. (2014). Prototype Layanan Izin Pemanfaatan Ruang Menggunakan Service Oriented Enterprise Architecture Framework. Jurnal Nasional Teknik Elektro Dan Teknologi Informasi, 3(2), 116–122.

Khoirunnisa, Lia. (2019). Rancang bangun sistem e-learning berbasis microservices dan domain driven design: Studi kasus Probistek UIN Maulana Malik Ibrahim Malang. Universitas Islam Negeri Maulana Malik Ibrahim.

Permana, Diyan Agus, & Dewantara, Rizki Yudhi. (2018). Analisis Dan Perancangan Sistem Informasi Perekrutan Karyawan Berbasis Web (Studi pada PT Sumber Abadi Bersama, Gondanglegi, Kabupaten Malang). Jurnal Administrasi Bisnis (JAB)| Vol, 56(1).

Putra, Yuri Chandra Tri. (2020). Implementasi Arsitektur Microservice pada Aplikasi Web Pengajaran Agama Islam Home Pesantren. Universitas Atma Jaya Yogyakarta.

Risnanto, Heri. (n.d.). Rancang Bangun Sistem Informasi Layanan Mandiri Perpustakaan Berbasis Arsitektur Microservice (studi kasus: Pusat Perpustakaan UIN Syarif Hidayatullah Jakarta). Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah ….

Riyanti, Kurnia Paranita Kartika, & Lukma, Hazairin Nikmatul. (2021). Introduction of Automatic Wooden Bridge Load Testing Detector For Pedestrians Using Load Cell Sensor. Procedia of Engineering and Life Science, 2.

Rozaq, Abdul. (2020). Konsep Perancangan Sistem Informasi Bisnis Digital. Poliban Press.

Sentosa, Rio Bayu. (2018). Membangun Web Konten Manajemen Sistem Secara Dinamis Dengan Bahasa Pemrograman Php Framework Codeigniter Dengan Database Mariadb. INTECOMS: Journal of Information Technology and Computer Science, 1(2), 212–223.

Setiawan, Daryanto. (2018). Dampak perkembangan teknologi informasi dan komunikasi terhadap budaya. JURNAL SIMBOLIKA Research and Learning in Communication Study, 4(1), 62–72.

Setyawan, Muhammad Yusril Helmi, & Pratiwi, Dinda Ayu. (2020). Membuat sistem informasi gadai online menggunakan codeigniter serta kelola proses pemberitahuannya. Kreatif Industri Nusantara.

Vitariani, Ria. (2019). Business Process Reengineering Pada Sistem Layanan Prima Perbendaharaan (Studi Kasus Sub Bagian Keuangan Badan Pemeriksa Keuangan Perwakilan Provinsi Jawa Barat). Program Sistem Informasi S1 Fakultas Teknik Universitas Widyatama.

Yanuarsari, Revita, Asmadi, Iwan, Muchtar, Hendi Suhendraya, & Sulastini, Rita. (2021). Peran Program Merdeka Belajar Kampus Merdeka dalam Meningkatkan Kemandirian Desa. Jurnal Basicedu, 5(6), 6307–6317.