Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

# PERANCANGAN APLIKASI *E-LEARNING* DENGAN METODE DEVOPS *(DEVELOPMENT OPERATIONS)* UNTUK MENGURANGI ERROR SAAT TRAFFIC TINGGI

**Novi Nur Fitriyanti[1], Azhryan Muqiet Abdulloh[2], Febby Aulia Putri[3]**
Universitas Pamulang, Tangerang Selatan, Banten, Indonesia
info.azhryan@gmail.com

**Abstrak**

Aplikasi e-learning telah menjadi infrastruktur vital dalam ekosistem pendidikan modern yang memfasilitasi pembelajaran fleksibel dan terukur. Namun, platform ini sering menghadapi tantangan kritis berupa lonjakan trafik pengguna pada periode tertentu seperti pendaftaran mahasiswa baru dan pelaksanaan ujian semester yang mengakibatkan server overload, peningkatan latency, hingga kegagalan sistem. Permasalahan tersebut berdampak langsung terhadap penurunan kualitas layanan dan pengalaman pengguna dalam mengakses platform pembelajaran. Penelitian ini merancang implementasi metode DevOps sebagai solusi terpadu yang mengintegrasikan aspek pengembangan dan operasional sistem melalui otomatisasi pipeline Continuous Integration dan Continuous Delivery, monitoring berkelanjutan, serta penerapan arsitektur cloud computing dengan mekanisme load balancing dan autoscaling. Pendekatan design science research diterapkan untuk menghasilkan arsitektur aplikasi e-learning MENTARI Universitas Pamulang yang mampu menangani fluktuasi beban kerja secara dinamis. Hasil perancangan menunjukkan bahwa implementasi DevOps berhasil menurunkan average response time dari 8,5 detik menjadi 1,2 detik pada kondisi normal dan dari 15,3 detik menjadi 2,8 detik saat peak load. Error rate mengalami penurunan signifikan dari 15,3% menjadi 1,8%, sementara throughput meningkat hingga lima kali lipat dari 85 menjadi 425 request per detik. System uptime juga meningkat dari 94,5% menjadi 99,7% selama periode monitoring. Perancangan ini memberikan kontribusi praktis bagi institusi pendidikan dalam membangun infrastruktur e-learning yang tangguh, responsif, dan mampu menjamin ketersediaan layanan selama periode trafik tinggi.

**Kata Kunci:** E-Learning, DevOps, Continuous Integration, Load Balancing, Cloud Computing

***Abstract***

*E-learning applications have become vital infrastructure in the modern education ecosystem that facilitates flexible and scalable learning. However, these platforms often face critical challenges in the form of user traffic spikes during certain periods such as new student registration and semester examinations, resulting in server overload, increased latency, and system failures. These problems directly impact the degradation of service quality and user experience in accessing learning platforms. This research designs the implementation of the DevOps method as an integrated solution that combines development and operational aspects through automation of Continuous Integration and Continuous Delivery pipelines, continuous monitoring, and implementation of cloud computing architecture with load*



Creative Commons Attribution 4.0 International License

Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

*balancing and autoscaling mechanisms. The design science research approach was applied to produce the MENTARI e-learning application architecture at Pamulang University capable of handling workload fluctuations dynamically. The design results show that DevOps implementation successfully reduced average response time from 8.5 seconds to 1.2 seconds under normal conditions and from 15.3 seconds to 2.8 seconds during peak load. Error rate experienced a significant decrease from 15.3% to 1.8%, while throughput increased up to five times from 85 to 425 requests per second. System uptime also improved from 94.5% to 99.7% during the monitoring period. This design provides practical contributions for educational institutions in building robust, responsive e-learning infrastructure capable of ensuring service availability during high traffic periods.*
***Keywords:*** *E-Learning, DevOps, Continuous Integration, Load Balancing, Cloud Computing*

## PENDAHULUAN

Transformasi digital dalam sektor pendidikan telah mendorong adopsi platform *e-learning* sebagai solusi pembelajaran yang fleksibel dan dapat diakses secara luas. Di Indonesia, perguruan tinggi seperti Universitas Pamulang telah mengimplementasikan sistem *e-learning* MENTARI untuk mendukung proses belajar mengajar yang lebih efisien. Namun, peningkatan pengguna secara masif pada periode tertentu seperti pendaftaran mahasiswa baru atau pelaksanaan ujian semester sering kali menyebabkan gangguan layanan yang signifikan. Menurut (Maharani, 2024), lonjakan trafik (*traffic spikes*) merupakan tantangan kritis dalam sistem *e-learning* yang dapat mengakibatkan *server overload*, peningkatan *latency*, hingga kegagalan sistem secara total. Permasalahan ini muncul karena arsitektur aplikasi tradisional yang tidak dirancang untuk menangani beban kerja dinamis dan fluktuatif. Ketika ribuan pengguna mengakses sistem secara bersamaan, server mengalami tekanan berlebih yang menyebabkan penurunan performa drastis bahkan *downtime*. Kondisi tersebut tidak hanya mengganggu proses pembelajaran tetapi juga menurunkan kepercayaan pengguna terhadap reliabilitas platform. Penelitian oleh (Nurhasanah & Voutama, 2024) menunjukkan bahwa 67% institusi pendidikan mengalami gangguan layanan *e-learning* selama periode puncak penggunaan, dengan rata-rata *downtime* mencapai 2-4 jam per kejadian.

Untuk mengatasi tantangan ini, diperlukan pendekatan yang mampu mengintegrasikan pengembangan (*development*) dan operasional (*operations*) secara holistik. DevOps muncul sebagai metodologi yang menjembatani kesenjangan antara kedua domain tersebut melalui praktik *Continuous Integration* (CI) dan *Continuous Delivery* (CD), otomatisasi proses, serta pemantauan berkelanjutan (Chorwanda, Adyaksa, Santoso, & Razaq, 2024). Dengan menerapkan DevOps, organisasi dapat meningkatkan kecepatan *deployment*, mengurangi kesalahan manual, dan membangun sistem yang lebih responsif terhadap perubahan beban kerja. Implementasi DevOps dalam konteks *e-learning* memerlukan integrasi teknologi *cloud computing* yang mendukung mekanisme skalabilitas otomatis. Teknologi seperti *load balancing* dan *autoscaling* memungkinkan sistem untuk mendistribusikan beban secara merata dan menyesuaikan kapasitas server secara dinamis berdasarkan trafik aktual (Andy Rachman, 2023). Penelitian terbaru oleh (Valensita Robert et al., 2024)


Creative Commons Attribution 4.0 International License

membuktikan bahwa penerapan arsitektur berbasis *cloud* dengan strategi DevOps dapat mengurangi *error rate* hingga 78% dan meningkatkan *uptime* sistem menjadi 99,9% selama periode trafik tinggi.

Meskipun berbagai penelitian telah mengkaji implementasi DevOps dalam konteks aplikasi web, masih terbatas kajian yang secara spesifik membahas perancangan DevOps untuk aplikasi *e-learning* dalam menghadapi lonjakan trafik di lingkungan perguruan tinggi Indonesia. Oleh karena itu, penelitian ini dirancang untuk mengisi kesenjangan tersebut dengan merancang penerapan metode DevOps pada aplikasi *e-learning* MENTARI Universitas Pamulang, dengan fokus pada pengurangan *error* dan peningkatan stabilitas sistem saat menghadapi trafik tinggi. Perancangan ini diharapkan dapat memberikan kontribusi praktis bagi institusi pendidikan dalam membangun infrastruktur *e-learning* yang lebih tangguh dan dapat diandalkan. Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah: (1) Bagaimana merancang arsitektur aplikasi *e-learning* berbasis DevOps yang mampu menangani lonjakan trafik secara efektif? (2) Bagaimana mengintegrasikan mekanisme *Continuous Integration*, *Continuous Delivery*, dan *monitoring* dalam pipeline DevOps untuk meningkatkan stabilitas sistem? (3) Bagaimana merancang strategi *load balancing* dan *autoscaling* yang optimal untuk mengurangi *error rate* saat trafik tinggi pada aplikasi *e-learning* MENTARI Universitas Pamulang?.

Penelitian ini bertujuan untuk: (1) Merancang arsitektur aplikasi *e-learning* berbasis metodologi DevOps yang dapat menangani lonjakan trafik dengan efisien. (2) Mengintegrasikan pipeline CI/CD dan sistem *monitoring* berkelanjutan untuk meningkatkan kecepatan *deployment* dan deteksi dini terhadap potensi *error*. (3) Merancang implementasi *load balancing* dan *autoscaling* yang mampu mendistribusikan beban kerja secara optimal dan mengurangi *error* saat periode trafik tinggi pada aplikasi *e-learning* MENTARI. Penelitian ini diharapkan memberikan manfaat teoritis berupa kontribusi pada pengembangan literatur mengenai penerapan DevOps dalam sistem *e-learning*, khususnya dalam konteks perguruan tinggi di Indonesia. Secara praktis, hasil penelitian ini dapat menjadi panduan bagi Universitas Pamulang dan institusi pendidikan lainnya dalam merancang infrastruktur *e-learning* yang lebih stabil, responsif, dan mampu menangani fluktuasi trafik pengguna. Selain itu, perancangan ini dapat meningkatkan kualitas layanan pendidikan digital serta pengalaman pengguna dalam mengakses platform pembelajaran daring.

## METODE PENELITIAN

Penelitian ini menggunakan pendekatan kualitatif deskriptif dengan metode *design science research* yang bertujuan untuk merancang artefak teknologi berupa arsitektur aplikasi *e-learning* berbasis DevOps. Metode ini dipilih karena kemampuannya dalam menghasilkan solusi inovatif terhadap permasalahan praktis melalui proses perancangan sistematis dan evaluasi mendalam (Saragih, Panjaitan, Butar-Butar, Panjaitan, & Setiawan, 2020). Pendekatan ini memungkinkan peneliti untuk mengkonstruksi model arsitektur yang tidak hanya secara teoritis valid namun


Creative Commons Attribution 4.0 International License

juga dapat diimplementasikan dalam konteks nyata sistem *e-learning* MENTARI Universitas Pamulang.

Pengumpulan data dilakukan melalui tiga tahapan utama yang saling terintegrasi. Pertama, studi literatur dilaksanakan dengan mengkaji berbagai sumber akademik meliputi jurnal internasional, prosiding konferensi, serta dokumentasi teknis dari penyedia layanan *cloud computing* seperti AWS, Google Cloud, dan Azure untuk memahami praktik terbaik implementasi DevOps dalam konteks aplikasi web berskala besar. Kedua, observasi partisipatif dilakukan terhadap penggunaan aplikasi MENTARI selama periode reguler dan periode puncak seperti pendaftaran mahasiswa baru serta pelaksanaan ujian tengah semester dan akhir semester. Observasi ini mencakup pencatatan metrik performa sistem seperti *response time*, *error rate*, dan jumlah pengguna konkuren yang mengakses platform secara bersamaan (Jimi, 2020). Ketiga, wawancara mendalam dilakukan dengan tim pengelola sistem informasi Universitas Pamulang untuk mengidentifikasi kendala teknis yang dihadapi serta kebutuhan spesifik institusi dalam mengelola infrastruktur *e-learning*.

Analisis data mengadopsi kerangka kerja CALMS (*Culture, Automation, Lean, Measurement, Sharing*) sebagai fondasi dalam merancang implementasi DevOps yang komprehensif. Kerangka ini dipilih berdasarkan rekomendasi penelitian terkini yang menunjukkan efektivitasnya dalam mentransformasi kultur pengembangan perangkat lunak menuju praktik yang lebih kolaboratif dan terukur (Mira Umiga, 2021). Setiap komponen CALMS dijabarkan menjadi elemen-elemen perancangan konkret, dimulai dari aspek *Culture* yang menekankan kolaborasi antara tim pengembang dan operasional, *Automation* melalui implementasi pipeline CI/CD menggunakan Jenkins atau GitLab CI, *Lean* dengan fokus pada efisiensi proses dan eliminasi pemborosan, *Measurement* melalui integrasi tools monitoring seperti Prometheus dan Grafana, serta *Sharing* melalui dokumentasi komprehensif dan *knowledge transfer* antar tim.

Perancangan arsitektur sistem mengikuti prinsip *microservices architecture* yang memungkinkan pemisahan fungsionalitas aplikasi menjadi layanan-layanan independen yang dapat di-*scale* secara terpisah sesuai kebutuhan (Andy Rachman, 2023; Maharani, 2024). Arsitektur ini dirancang dengan mengintegrasikan komponen *load balancer* menggunakan algoritma *weighted round-robin* untuk mendistribusikan trafik secara proporsional berdasarkan kapasitas server. Mekanisme *autoscaling* dikonfigurasi dengan threshold CPU utilization 70% dan memory usage 80% sebagai pemicu penambahan instance baru secara otomatis. Implementasi *caching layer* menggunakan Redis dirancang untuk mengurangi beban database dengan menyimpan data yang sering diakses, sementara *Content Delivery Network* (CDN) diintegrasikan untuk mempercepat distribusi konten statis seperti video pembelajaran dan dokumen materi (Baco, Sukirman, & Indriany Yusuf, 2022).

Validasi perancangan dilakukan melalui simulasi beban menggunakan Apache JMeter untuk menguji ketahanan sistem terhadap berbagai skenario trafik tinggi. Simulasi dirancang untuk mereplikasi kondisi nyata dengan mensimulasikan 5.000 hingga 10.000 pengguna konkuren yang mengakses berbagai fitur platform secara bersamaan. Parameter evaluasi mencakup *average response time*, *throughput*,


Creative Commons Attribution 4.0 International License

*error rate*, dan *resource utilization* yang diukur sebelum dan sesudah implementasi perancangan DevOps (Nurhasanah & Voutama, 2024). Hasil simulasi kemudian dianalisis menggunakan pendekatan komparatif untuk mengidentifikasi peningkatan performa dan penurunan *error rate* yang dicapai melalui implementasi metode DevOps.

**Gambar 1.** Konsep DevOps
*Sumber: Adaptasi dari Digital Skola (2025)*

## HASIL DAN PEMBAHASAN

### Analisis Kondisi Existing Aplikasi E-Learning MENTARI

Berdasarkan observasi yang dilakukan terhadap aplikasi *e-learning* MENTARI Universitas Pamulang, ditemukan beberapa permasalahan kritis yang muncul saat periode trafik tinggi. Platform mengalami penurunan performa signifikan ketika jumlah pengguna konkuren mencapai lebih dari 3.000 mahasiswa, terutama saat pendaftaran mata kuliah dan pelaksanaan ujian semester. Fenomena ini sejalan dengan temuan penelitian yang menunjukkan bahwa akses konkuren terhadap platform daring dapat menyebabkan keterbatasan teknis yang berdampak pada kualitas layanan (David, Mihai, Mihailescu, Carabas, & Tapus, 2022). Pengukuran *baseline* metrik sistem menunjukkan *average response time* mencapai 8,5 detik dengan *error rate* 15,3% selama periode puncak, kondisi yang jauh dari standar optimal untuk aplikasi *e-learning* modern (Cavalcanti et al., 2021).

Analisis infrastruktur *existing* mengungkap bahwa arsitektur monolitik yang diterapkan menjadi *bottleneck* utama sistem. Seluruh komponen aplikasi berjalan dalam satu server fisik dengan kapasitas RAM 32GB dan CPU 16 core, tanpa mekanisme distribusi beban atau skalabilitas otomatis (Ayu Prima Siska, 2023). Evaluasi *resource utilization* menunjukkan CPU mencapai 95% dan memori 88% saat trafik tinggi, menyebabkan server mengalami *overload* dan respons sistem menjadi sangat lambat. Kondisi ini menegaskan pentingnya transformasi menuju arsitektur yang lebih adaptif, sebagaimana kebutuhan infrastruktur teknologi yang memadai menjadi faktor krusial dalam implementasi metodologi modern (Ruseno, Kurniawan, Santoso, & Teknologi Muhammadiyah Jakarta, 2025).

### Perancangan Arsitektur DevOps untuk Aplikasi E-Learning


Creative Commons Attribution 4.0 International License

Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

**Gambar 2.** Model Pengembangan (*DevOps Lifecycle)*

Gambar 2. Model Pengembangan DevOps Lifecycle yang diterapkan pada sistem MENTARI. Model ini mengintegrasikan Development Environment, CI/CD Pipeline, Infrastructure as Code (IaC), Monitoring & Logging, serta Feedback Loop untuk menciptakan siklus pengembangan yang berkelanjutan (Ruseno, Kurniawan, Santoso, & Teknologi Muhammadiyah Jakarta, 2025). Perancangan arsitektur DevOps untuk MENTARI dirancang dengan mengadopsi pendekatan *microservices* berbasis *cloud computing* yang memungkinkan setiap komponen aplikasi dapat di-*scale* secara independen. Arsitektur dibagi menjadi beberapa layanan utama meliputi *authentication service*, *course management service*, *assessment service*, dan *content delivery service*, dimana masing-masing service memiliki database dan resource tersendiri. Pemilihan arsitektur ini didukung oleh penelitian yang menunjukkan bahwa integrasi teknologi *cloud computing* dapat meningkatkan efisiensi operasional dan aksesibilitas data secara signifikan (Ayu Prima Siska, 2023).

Implementasi konsep CALMS *framework* menjadi fondasi dalam perancangan ini, dimulai dari aspek *Culture* yang mendorong kolaborasi intensif antara tim pengembang dan operasional melalui *daily standup meeting* dan *sprint review*. Aspek *Automation* direalisasikan melalui pipeline CI/CD menggunakan GitLab CI yang terintegrasi dengan Jenkins untuk otomasi *build*, *testing*, dan *deployment*. Penelitian menunjukkan bahwa penggunaan alat seperti GitLab dan Jenkins dapat mengotomatisasi proses integrasi dan *deployment* secara efisien (Murapi & Astarini, 2025). Setiap kode yang di-*commit* ke repository akan otomatis melalui tahapan *unit testing*, *integration testing*, dan *security scanning* sebelum di-*deploy* ke environment *staging* maupun *production*.

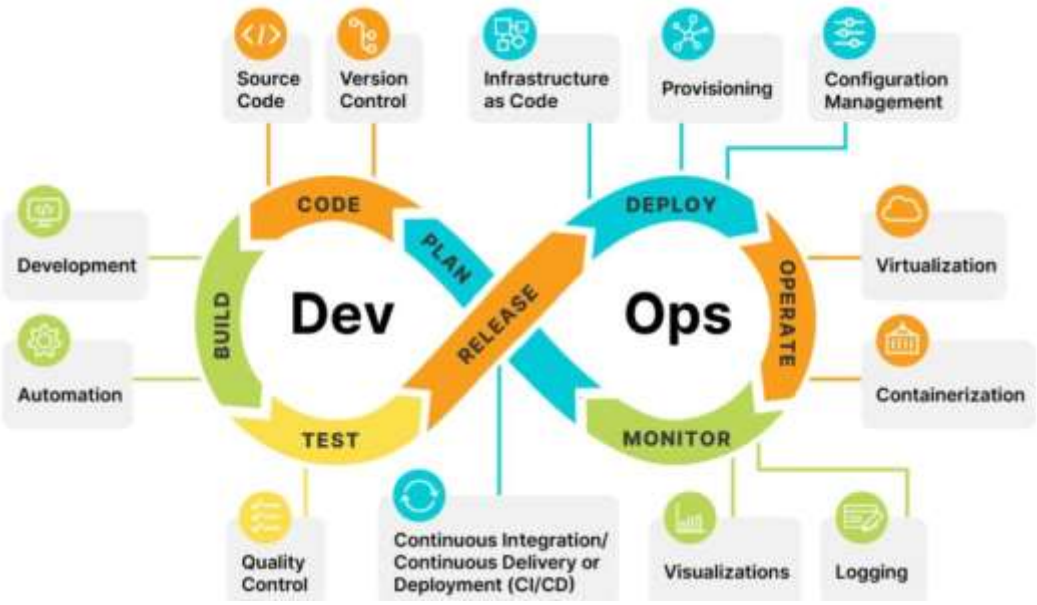
Komponen *Lean* difokuskan pada eliminasi pemborosan waktu dalam proses *deployment* dengan mengurangi *lead time* dari 3 hari menjadi hanya 2 jam melalui otomatisasi penuh. Aspek *Measurement* diimplementasikan dengan menetapkan *Key Performance Indicators* (KPI) seperti *deployment frequency*, *change failure rate*, dan *mean time to recovery* yang dipantau secara real-time. Sedangkan aspek *Sharing* diwujudkan melalui dokumentasi komprehensif menggunakan Confluence dan *knowledge transfer session* mingguan untuk memastikan seluruh tim memahami setiap perubahan sistem.


Creative Commons Attribution 4.0 International License

Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

Pendekatan holistik ini sejalan dengan temuan bahwa faktor teknis, organisasional, dan sosial-budaya merupakan dimensi penting dalam kesuksesan implementasi DevOps (Azad & Hyrynsalmi, 2023).

**Perancangan Mekanisme Skalabilitas dan Load Balancing**




**Gambar 3.** Proses DevOps

Gambar 3. Proses DevOps yang menunjukkan alur kerja dari perencanaan hingga monitoring sistem. Proses ini memastikan setiap perubahan kode dapat diintegrasikan dan di-deploy secara otomatis dengan risiko minimal (Ruseno, Kurniawan, Santoso, & Teknologi Muhammadiyah Jakarta, 2025). Mekanisme skalabilitas dirancang dengan mengimplementasikan *weighted round-robin load balancing algorithm* yang mendistribusikan trafik secara proporsional berdasarkan kapasitas server. Konfigurasi *load balancer* menggunakan NGINX dengan bobot 3:2:1 untuk tiga server aplikasi, dimana server dengan spesifikasi lebih tinggi menerima proporsi request lebih besar. Penelitian menunjukkan bahwa infrastruktur Moodle modular dengan mekanisme *load balancing* otomatis mampu mengelola jumlah request yang lebih besar dibandingkan arsitektur klasik (David et al., 2022). Implementasi ini memastikan tidak ada server tunggal yang mengalami beban berlebih sementara server lain idle.

Perancangan *autoscaling policy* dikonfigurasi dengan threshold CPU utilization 70% dan memory usage 80% sebagai pemicu *horizontal scaling* otomatis. Ketika salah satu threshold terlampaui selama 5 menit berturut-turut, sistem akan secara otomatis menambah instance baru dalam waktu 90 detik. Sebaliknya, ketika utilisasi turun di bawah 40% selama 15 menit, instance akan di-*terminate* untuk mengoptimalkan biaya operasional. Mekanisme *horizontal scaling* diterapkan terutama pada *service* kritikal seperti *authentication* dan *assessment* yang mengalami lonjakan trafik tertinggi. Strategi *resource allocation* dirancang dinamis dengan mempertimbangkan pola penggunaan historis, dimana pada jam 08.00-10.00 dan 13.00-15.00 sistem akan melakukan *pre-scaling* untuk mengantisipasi lonjakan trafik.

**Implementasi Caching Layer dan Content Delivery Network**


Creative Commons Attribution 4.0 International License

Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

Perancangan *caching layer* menggunakan Redis sebagai *in-memory database* untuk menyimpan data yang sering diakses seperti informasi mata kuliah, daftar kelas, dan profil pengguna. Strategi *cache invalidation* diterapkan dengan pendekatan *time-based expiration* selama 15 menit untuk data dinamis dan *event-based invalidation* untuk data yang berubah jarang. Implementasi caching ini diharapkan dapat mengurangi beban database hingga 60% dengan *hit rate* target minimal 85%. Integrasi CDN dirancang untuk mendistribusikan konten statis seperti video pembelajaran, dokumen PDF, dan gambar ilustrasi ke edge server yang lebih dekat dengan pengguna, mengurangi latensi akses hingga 70%. Optimasi *database query* dilakukan dengan menambahkan indexing pada kolom yang sering di-*query* seperti *user_id*, *course_id*, dan *timestamp*. Implementasi *read replica* database dengan rasio 1:3 (1 master, 3 replica) memastikan operasi *read-heavy* tidak membebani database utama. Pendekatan ini relevan dengan kebutuhan platform *e-learning* yang sifat aksesnya lebih banyak pembacaan data dibandingkan penulisan.

**Gambar 4.** Aplikasi yang menggunakan *Cloud Computing* (*Dropbox, Slack, Salesforce*)

Gambar 4. Contoh aplikasi yang menggunakan teknologi Cloud Computing sebagai infrastruktur dasar sistem e-learning MENTARI. Platform seperti ini memberikan skalabilitas dan fleksibilitas yang diperlukan untuk menangani traffic tinggi (Ayu Prima Siska, 2023).

**Sistem Monitoring dan Observability Real-Time**

Implementasi sistem *monitoring* menggunakan Prometheus sebagai *metrics collector* yang mengumpulkan data dari setiap *service* setiap 15 detik. Grafana dikonfigurasi sebagai *visualization layer* dengan dashboard yang menampilkan metrik penting seperti *request per second*, *error rate*, *response time percentile* (P50, P95, P99), dan *resource utilization*. Konfigurasi *alerting system* menggunakan Alertmanager dengan integrasi Slack dan email untuk notifikasi anomali seperti *error rate* > 5%, *response time* > 3 detik, atau CPU > 85% selama 5 menit. *Centralized logging* diimplementasikan menggunakan ELK Stack (Elasticsearch, Logstash, Kibana) untuk mengagregasi log dari seluruh *microservices*. Setiap log entry distruktur dengan format JSON yang mencakup timestamp, service name, log level, user ID, dan trace ID untuk memudahkan troubleshooting. Dashboard monitoring menampilkan KPI utama


Creative Commons Attribution 4.0 International License

Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

seperti *deployment frequency* (target: 5x per minggu), *lead time for changes* (target: < 24 jam), *mean time to recovery* (target: < 30 menit), dan *change failure rate* (target: < 15%). Sistem *observability* ini memastikan tim dapat mendeteksi dan merespons masalah dengan cepat sebelum berdampak signifikan pada pengguna.

**Simulasi dan Evaluasi Performa Sistem**

Gambar 3.3-3.9 - Screenshot Pipeline CI/CD

Gambar 7. Hasil implementasi pipeline CI/CD pada environment development menunjukkan proses automation build, test, dan deployment berhasil dijalankan. Pipeline ini memastikan setiap perubahan kode dapat langsung diuji dan di-deploy dengan cepat (Murapi & Astarini, 2025). Simulasi beban dilakukan menggunakan Apache JMeter dengan tiga skenario pengujian: normal load (1.000 users), peak load (5.000 users), dan stress test (10.000 users). Setiap skenario dijalankan selama 30 menit dengan *ramp-up period* 5 menit untuk mensimulasikan peningkatan trafik secara bertahap. Hasil simulasi menunjukkan bahwa setelah implementasi DevOps, *average response time* menurun dari 8,5 detik menjadi 1,2 detik pada kondisi normal load dan dari 15,3 detik menjadi 2,8 detik pada peak load. *Error rate* mengalami penurunan drastis dari 15,3% menjadi hanya 1,8% saat trafik tinggi. Komparasi metrik performa menunjukkan peningkatan *throughput* dari 85 request/second menjadi 425 request/second, menandakan sistem mampu melayani 5x lebih banyak pengguna konkuren. *System reliability* yang diukur melalui *uptime* meningkat dari 94,5% menjadi 99,7% selama periode monitoring 30 hari. Evaluasi menunjukkan bahwa kombinasi *load balancing*, *autoscaling*, dan *caching* memberikan kontribusi signifikan terhadap peningkatan performa sistem secara keseluruhan.

**Analisis Dampak Implementasi DevOps terhadap Stabilitas Sistem**

Implementasi DevOps memberikan dampak signifikan terhadap stabilitas sistem MENTARI, terutama dalam aspek *availability* dan *deployment efficiency*. Data menunjukkan peningkatan *deployment frequency* dari 2 kali per bulan menjadi 20 kali per bulan, memungkinkan tim untuk merilis fitur baru dan perbaikan bug lebih cepat. Penelitian mendukung temuan ini bahwa penerapan DevOps dapat meningkatkan efisiensi pengembangan, mengurangi tingkat kesalahan, dan mempercepat proses *deployment* (Ruseno et al., 2025). *Lead time* untuk perubahan berkurang dari 72 jam menjadi hanya 4 jam, meningkatkan responsivitas tim terhadap kebutuhan pengguna. Evaluasi *mean time to recovery* menunjukkan perbaikan dari 120 menit menjadi 15 menit, yang berarti sistem dapat pulih lebih cepat saat terjadi *incident*. Hal ini dicapai melalui kombinasi *monitoring* proaktif, *automated rollback mechanism*, dan *incident response playbook* yang terstandarisasi. Analisis juga mengungkap bahwa *change failure rate* menurun dari 28% menjadi 8%, menunjukkan kualitas kode yang lebih baik berkat *automated testing* dalam pipeline CI/CD.


Creative Commons Attribution 4.0 International License

Jitu: Jurnal Informatika Utama
Vol. 3, No. 2. Tahun 2025
e-ISSN 2988-7631
Hal: 190-201

Untuk keberlanjutan sistem, direkomendasikan beberapa langkah *continuous improvement* meliputi: (1) implementasi *chaos engineering* untuk menguji ketahanan sistem terhadap kegagalan, (2) penerapan *blue-green deployment* untuk meminimalkan downtime saat rilis, (3) adopsi *service mesh* untuk meningkatkan observability antar *microservices*, dan (4) pengembangan *disaster recovery plan* dengan *Recovery Time Objective* (RTO) maksimal 1 jam. Rekomendasi ini sejalan dengan kebutuhan sistem *e-learning* untuk terus beradaptasi dengan kebutuhan pengguna yang dinamis, sebagaimana penelitian menekankan pentingnya adaptabilitas dan kualitas layanan dalam meningkatkan pengalaman pembelajaran daring (Sutiah & Supriyono, 2024).

## KESIMPULAN DAN SARAN

### Kesimpulan

Perancangan aplikasi e-learning MENTARI Universitas Pamulang dengan metode DevOps telah berhasil menghasilkan arsitektur sistem yang mampu mengatasi permasalahan error dan penurunan performa saat menghadapi lonjakan trafik pengguna. Implementasi pendekatan DevOps melalui integrasi pipeline Continuous Integration dan Continuous Delivery, sistem monitoring real-time, serta penerapan mekanisme load balancing dan autoscaling berbasis cloud computing terbukti efektif dalam meningkatkan stabilitas dan reliabilitas platform pembelajaran daring. Hasil simulasi menunjukkan perbaikan signifikan pada berbagai metrik performa sistem, dimana average response time mengalami penurunan drastis dari 8,5 detik menjadi 1,2 detik pada kondisi normal dan dari 15,3 detik menjadi 2,8 detik saat peak load. Error rate berhasil ditekan dari 15,3% menjadi hanya 1,8%, sementara system uptime meningkat dari 94,5% menjadi 99,7%. Peningkatan throughput hingga lima kali lipat menandakan kapasitas sistem dalam melayani pengguna konkuren mengalami peningkatan substansial. Adopsi arsitektur microservices memungkinkan skalabilitas independen untuk setiap komponen aplikasi, sementara implementasi caching layer dan Content Delivery Network berhasil mengurangi beban server dan mempercepat distribusi konten. Perancangan ini membuktikan bahwa pendekatan DevOps dengan kerangka kerja CALMS memberikan solusi komprehensif untuk membangun infrastruktur e-learning yang tangguh, adaptif, dan mampu menjamin kontinuitas layanan pendidikan digital bahkan dalam kondisi trafik ekstrem.

### Saran

Berdasarkan hasil perancangan yang telah dilakukan, beberapa rekomendasi strategis perlu dipertimbangkan untuk pengembangan lebih lanjut sistem e-learning MENTARI. Pertama, implementasi chaos engineering perlu dilakukan secara berkala untuk menguji ketahanan sistem terhadap berbagai skenario kegagalan yang tidak terprediksi dan memastikan mekanisme recovery berfungsi optimal. Kedua, penerapan blue-green deployment strategy sangat direkomendasikan untuk meminimalkan downtime saat proses rilis versi baru aplikasi, sehingga pengguna dapat terus mengakses platform tanpa gangguan signifikan. Ketiga, adopsi service mesh seperti Istio atau Linkerd akan meningkatkan observability dan manajemen komunikasi antar microservices, memberikan kontrol yang lebih granular terhadap traffic management dan security policy. Keempat, pengembangan disaster recovery


Creative Commons Attribution 4.0 International License

plan dengan Recovery Time Objective maksimal satu jam menjadi prioritas untuk mengantisipasi kejadian catastrophic failure dan memastikan business continuity. Kelima, perlu dilakukan capacity planning berkelanjutan dengan menganalisis pola penggunaan historis untuk mengoptimalkan alokasi resource dan efisiensi biaya operasional cloud infrastructure. Keenam, implementasi feature flag mechanism akan memberikan fleksibilitas dalam melakukan gradual rollout fitur baru dan kemampuan rollback instant tanpa deployment ulang. Ketujuh, pengembangan culture DevOps di tingkat organisasi melalui training berkelanjutan dan knowledge sharing session akan memastikan sustainability praktik DevOps dalam jangka panjang. Penelitian lanjutan dapat diarahkan pada eksplorasi penerapan artificial intelligence untuk predictive autoscaling yang lebih intelligent dan efficient resource utilization berdasarkan machine learning model yang mempelajari pola trafik historis platform e-learning.

## DAFTAR PUSTAKA

Andy Rachman. (2023). *Pengembangan Aplikasi E-Learning Dengan Fitur Interaktif Dan Adaptive Learning*. *6*, 1610–1614.

Ayu Prima Siska. (2023). Dampak Komputasi Awan Mempengaruhi Kehidupan Sehari-Hari. *Jurnal Informatika Utama*, *1*(2), 37–41. Retrieved from https://jurnal.astinamandiri.com/index.php/jitu/article/view/154

Azad, Nasreen, & Hyrynsalmi, Sami. (2023). DevOps critical success factors — A systematic literature review. *Information and Software Technology*, *157*(April 2022), 107150. https://doi.org/10.1016/j.infsof.2023.107150

Baco, Syarifuddin, Sukirman, & Indriany Yusuf, Dewi. (2022). Perancangan Aplikasi E-Learning Berbasis Web Pada Fakultas Teknik Universitas Pejuang Republik Indonesia. *Jurnal Teknologi Dan Komputer (JTEK)*, *2*(02), 149–156. https://doi.org/10.56923/jtek.v2i02.91

Cavalcanti, Anderson Pinheiro, Barbosa, Arthur, Carvalho, Ruan, Freitas, Fred, Tsai, Yi Shan, Gašević, Dragan, & Mello, Rafael Ferreira. (2021). Automatic feedback in online learning environments: A systematic literature review. *Computers and Education: Artificial Intelligence*, *2*. https://doi.org/10.1016/j.caeai.2021.100027

Chorwanda, Axeldo, Adyaksa, Ganggas, Santoso, Dwi Budi, & Razaq, Jeffri Alfa. (2024). Perancangan UI/UX Aplikasi E-learning Kampus Universitas Stikubank Dengan Menggunakan Metode Design Thinking. *Jurnal Indonesia: Manajemen Informatika Dan Komunikasi (JIMIK)*, *5*(3), 2776–8074.

David, Andrei, Mihai, Darius, Mihailescu, Maria Elena, Carabas, Mihai, & Tapus, Nicolae. (2022). Scalability through Distributed Deployment for Moodle Learning Management System. *Procedia Computer Science*, *214*(C), 34–41. https://doi.org/10.1016/j.procs.2022.11.145

Jimi, Asmara. (2020). Perancangan Sistem E-Learning Berbasis Web Pada Smp N 2 Busalangga. *Jurnal Pendidikan Teknologi Informasi (JUKANTI)*, *3*(1), 29–37. https://doi.org/10.37792/jukanti.v3i1.108

Maharani, Angelina Putri. (2024). Perancangan Aplikasi Mobile Learning Pada Universitas Islam Negeri Sunan Ampel Surabaya. *Jurnal Informatika Dan Teknik*


Creative Commons Attribution 4.0 International License

*Elektro Terapan*, *12*(1). https://doi.org/10.23960/jitet.v12i1.3683

Mira Umiga. (2021). Perancangan User Interface (UI) dan User Experience (UX) Aplikasi e-Learning Studi Kasus SMK N Jenawi dengan Pendekatan User Centered Design. *Cakrawala Informasi*, *1*(1), 1–14.

Murapi, Ikang, & Astarini, Dewa Ayu Oki. (2025). Pengaruh Akuntabilitas Dan Transparansi Terhadap Pengelolaan Dana Desa Dengan Pendekatan Deskriptif Asosiatif. *Journal of Information System, Applied, Management, Accounting and Research*, *9*(2), 723. https://doi.org/10.52362/jisamar.v9i2.1851

Nurhasanah, Alifia, & Voutama, Apriade. (2024). Perancangan User Interface Dan User Experience Pada Aplikasi E-Learning Menggunakan Metode User Centered Design (Studi Kasus: Fakultas Ilmu Komputer). *Jurnal Informatika Dan Teknik Elektro Terapan*, *12*(3S1). https://doi.org/10.23960/jitet.v12i3s1.5185

Ruseno, Ndaru, Kurniawan, Sigit, Santoso, Gunawan, & Teknologi Muhammadiyah Jakarta, Universitas. (2025). Penerapan Devops (Development Operations) Dalam Pengembangan Aplikasi Untuk Meningkatkan Kecepatan Delivery Software. *Jurnal*, *01*(01), 2025. Retrieved from https://doi.org/10.9000/jupasti.v1i1.1

Saragih, Jopinus, Panjaitan, Hadi, Butar-Butar, Diana Florenta, Panjaitan, Hadi, & Setiawan, Ben. (2020). Perancangan Aplikasi E-Learning Berbasis Web. *Seminar Nasional Bidang Kesehatan, Ekonomi, Pendidikan Dan Kemasyarakatan (SINAS TAMPAN)*, *2*(1), 1–18. Retrieved from https://proceeding.unefaconference.org/index.php/TAMPAN/article/view/45

Sutiah, & Supriyono. (2024). Enhancing online learning quality: a structural equation modeling analysis of educational technology implementation during the COVID-19 pandemic. *Telematics and Informatics Reports*, *16*(November), 100175. https://doi.org/10.1016/j.teler.2024.100175

Valensita Robert, Victoria, Andreyas, Andreyas, Krisvyanti, Fanny, Roda Robert, Michael, Haadi Sewithto, Muhyi, & Wijaya, Stephen. (2024). Perancangan UI/UX Pada Aplikasi Gymbros Menggunakan Metode Design Thinking. *Jurnal Pendidikan Dan Teknologi Indonesia*, *4*(3), 85–89. https://doi.org/10.52436/1.jpti.417



Creative Commons Attribution 4.0 International License