IMPLEMENTASI SISTEM MONITORING INFRASTRUKTUR SERVER BERBASIS PROMETHEUS DAN GRAFANA DENGAN INTEGRASI CHATBOT SLACK MENGGUNAKAN METODE NETWORK DEVELOPMENT LIFE CYCLE PADA PT GLOBAL TIKET NETWORK

Ridwan Dwi Susilo[1)], Nopiyanto[2)], Saga Cindai Alisanda[3)]
[1)]Program Studi Informatika, Universitas Panca Sakti Bekasi
E-mail : 1ridwanfps@gmail.com
[2)]Program Studi Manajemen Informatika, Universitas Panca Sakti Bekasi
E-mail : info.nopiyanto@gmail.com
[3)]Program Studi Manajemen Informatika, Universitas Panca Sakti Bekasi
E-mail: saga.cindai@gmail.com

ABSTRAK

Pemantauan infrastruktur server yang masih dikerjakan secara manual membuat anomali pemakaian sumber daya sering terlambat diketahui. Kondisi tersebut ditemukan pada PT Global Tiket Network, tempat tim System Engineer memeriksa kondisi server dengan mengakses setiap server melalui SSH sehingga anomali baru diketahui puluhan menit hingga lebih dari satu jam setelah terjadi. Penelitian ini mengimplementasikan sistem monitoring infrastruktur server berbasis Prometheus dan Grafana yang diintegrasikan dengan chatbot Slack berbasis Large Language Model menggunakan metode Network Development Life Cycle dengan tahapan analysis, design, simulation, implementation, monitoring, dan management. Sistem dibangun bertahap melalui Model Draft 1 untuk pengumpulan metrik dan visualisasi terpusat, Model Draft 2 untuk notifikasi otomatis ke Slack, serta Model Final yang menambahkan interaksi dua arah beserta pembatasan perintah. Pengujian dilakukan pada lingkungan simulasi yang merepresentasikan server cloud perusahaan dan mencakup aspek fungsionalitas, akurasi deteksi anomali, kecepatan notifikasi, serta efektivitas chatbot. Lima skenario fungsional berhasil dijalankan seluruhnya, alert terkirim ketika utilisasi CPU, memori, dan disk melewati ambang batas 60 persen serta tidak muncul pada kondisi normal sebagai kontrol negatif, notifikasi diterima pada kanal Slack kurang dari 60 detik, dan chatbot menjawab permintaan status sumber daya sekaligus menolak perintah di luar daftar yang diizinkan. Penelitian ini menyimpulkan bahwa penggabungan monitoring Prometheus dan Grafana dengan lapisan ChatOps berbasis Large Language Model mempercepat kesadaran tim terhadap anomali dibandingkan pemantauan manual, sekaligus menyediakan riwayat pemakaian sumber daya yang sebelumnya tidak terekam.

Kata kunci: monitoring infrastruktur server, Prometheus, Grafana, ChatOps, Network Development Life Cycle

1. PENDAHULUAN

Ketersediaan layanan digital bergantung pada kesehatan infrastruktur server yang menopangnya. Lonjakan pemakaian prosesor, memori, kapasitas penyimpanan, maupun trafik jaringan yang tidak segera ditangani dapat menurunkan mutu layanan hingga menyebabkan waktu henti. Kondisi tersebut menuntut kemampuan pemantauan yang memberi gambaran kondisi sistem secara menyeluruh dan berkelanjutan (Ramadoni dkk., 2021; Chapman & Holmes, 2023).

Penelitian ini dilakukan pada PT Global Tiket Network, perusahaan teknologi di bidang pemesanan tiket dan akomodasi daring yang mengelola sejumlah server cloud untuk menopang layanannya. Berdasarkan observasi dan wawancara dengan Lead System Engineer perusahaan tersebut, pemantauan kondisi server sebelum penelitian ini masih dikerjakan secara manual dengan mengakses tiap server melalui SSH lalu memeriksa pemakaian sumber daya menggunakan perintah dasar sistem operasi. Karena sistem monitoring terpusat dan notifikasi otomatis belum tersedia, anomali umumnya baru diketahui ketika pemeriksaan berkala dilakukan atau setelah tim lain melaporkan layanan yang melambat. Jeda antara munculnya anomali dan kesadaran tim berkisar puluhan menit hingga lebih dari satu jam, terutama di luar jam kerja. Riwayat tren pemakaian sumber daya yang tidak terekam juga menyulitkan analisis setelah gangguan terjadi.

Beragam penelitian telah membahas pemantauan infrastruktur. Putra dkk. (2023) menyusun model utilisasi dan visualisasi sumber daya server menggunakan Prometheus dan Grafana, sedangkan Ramadoni dkk. (2021) mengevaluasi kombinasi keduanya untuk memantau basis data MongoDB. Izza dan Sholeha (2025) menerapkan monitoring server berbasis Linux dengan visualisasi Grafana pada lingkungan kampus. Pada jalur perangkat jaringan, Pradana dkk. (2022) memanfaatkan Zabbix berbasis SNMP, Yulvianda dan Ismail (2023) menggabungkan Zabbix dengan Grafana, dan Ariyadi dkk. (2024) menerapkan Zabbix dengan metode Network Development Life Cycle. Kusuma dan Oktiawati (2022) menyoroti bahwa instrumentasi metrik pada level aplikasi berpotensi menambah beban kerja sistem, sehingga pemantauan pada level infrastruktur tetap relevan.

Sebagian penelitian telah menyambungkan sistem monitoring dengan kanal komunikasi tim. Saory dkk. (2024) mengirim notifikasi otomatis ke Slack, Rasyidi dan Pratama (2024) memakai kanal Telegram, dan Sopyan dan Rovi (2025) memanfaatkan WhatsApp Gateway dengan kerangka Network Development Life Cycle. Saputra dkk. (2025) serta Defani dan Wijayanto (2025) memperkuat temuan bahwa

Copyright@2026 FST PSU Bekasi

penyatuan visualisasi pada satu dashboard mempermudah pengawasan infrastruktur yang beragam. Meskipun demikian, arah komunikasi pada penelitian tersebut masih satu arah, yaitu sistem mengirim pesan dan tim menerimanya. Ketika tim membutuhkan detail lanjutan, dashboard tetap harus dibuka atau server tetap harus diakses secara langsung.

Di sisi lain, perkembangan Large Language Model membuka peluang percakapan yang lebih luwes dibanding balasan berbasis templat tetap (Zhang dkk., 2024; Alvin dkk., 2025). Peluang tersebut belum banyak dimanfaatkan pada konteks pemantauan infrastruktur, padahal pendekatan ChatOps memungkinkan kegiatan operasional dijalankan langsung dari platform percakapan yang telah dipakai tim setiap hari. Celah itulah yang dituju penelitian ini, dengan tetap membatasi perintah yang boleh dijalankan agar akses ke server tetap terkendali (Tuyishime dkk., 2023).

Berdasarkan uraian tersebut, penelitian ini bertujuan mengimplementasikan sistem monitoring infrastruktur server berbasis Prometheus dan Grafana dengan metode Network Development Life Cycle, menyusun dashboard visualisasi terpusat untuk metrik CPU, memori, storage, dan jaringan, mengintegrasikan chatbot Slack berbasis *Large Language Model* sebagai media interaktif, serta menguji kinerja sistem pada aspek fungsionalitas, akurasi deteksi anomali, kecepatan notifikasi, dan efektivitas chatbot.

2. LANDASAN TEORI

2.1. Monitoring Infrastruktur Server

Monitoring infrastruktur server merupakan kegiatan mengumpulkan, menyimpan, dan menyajikan metrik kondisi server secara berkelanjutan agar penyimpangan dapat dikenali sedini mungkin. Metrik yang lazim dipantau meliputi utilisasi CPU, pemakaian memori, kapasitas penyimpanan, dan trafik jaringan. Chapman dan Holmes (2023) menjelaskan bahwa pemantauan yang baik tidak berhenti pada penyajian angka, tetapi menghasilkan kemampuan mengamati perilaku sistem sepanjang waktu sehingga pola kenaikan beban dapat dikenali sebelum berdampak pada layanan. Putra dkk. (2023) menunjukkan bahwa penyatuan metrik dari banyak server pada satu dashboard mempersingkat waktu yang dibutuhkan pengelola untuk menilai kondisi keseluruhan infrastruktur.

2.2. Prometheus dan Node Exporter

Prometheus merupakan perangkat lunak pemantauan sumber terbuka yang menyimpan metrik sebagai data berbasis waktu dan mengambil data dengan mekanisme penarikan berkala terhadap target yang dikonfigurasikan. Hegedus (2024) menjelaskan bahwa pendekatan penarikan tersebut menyederhanakan pengelolaan target karena konfigurasi terpusat pada sisi peladen pemantau. Metrik tingkat sistem operasi disediakan oleh Node Exporter, yaitu agen yang dijalankan pada setiap server dan menyajikan metrik melalui protokol HTTP. Pemisahan peran antara agen pengumpul metrik dan peladen penyimpan metrik memudahkan penambahan server baru tanpa mengubah rancangan sistem (Ramadoni dkk., 2021).

2.3. Grafana dan Alertmanager

Grafana adalah perangkat visualisasi yang membaca sumber data pemantauan lalu menampilkannya sebagai panel dan grafik tren. Grafana Labs (2023) menjelaskan bahwa Grafana juga menyediakan mekanisme alerting berupa aturan yang mengevaluasi kondisi metrik pada selang tertentu, serta jalur pengiriman pemberitahuan ke kanal komunikasi. Pada arsitektur Prometheus, pengiriman pemberitahuan dijalankan oleh Alertmanager yang menerima kondisi alert, mengelompokkan pesan sejenis, dan meneruskannya ke penerima yang ditetapkan. Saory dkk. (2024) memanfaatkan jalur tersebut untuk mengirim notifikasi ke Slack, sedangkan Rasyidi dan Pratama (2024) menggunakannya untuk kanal Telegram.

2.4. ChatOps dan Chatbot Berbasis LLM

ChatOps merupakan pendekatan kerja yang menempatkan platform percakapan tim sebagai antarmuka kegiatan operasional, sehingga informasi dan tindakan berada pada satu tempat yang sama dengan diskusi tim. Slack Technologies (2023) menyediakan antarmuka pemrograman yang memungkinkan aplikasi menerima peristiwa percakapan dan mengirim balasan pada kanal. Chatbot yang dibangun di atas kanal tersebut dapat memakai pendekatan berbasis aturan maupun model bahasa. Matic dkk. (2021) menjelaskan bahwa arsitektur chatbot umumnya memisahkan pemahaman maksud pengguna dari eksekusi tindakan, sedangkan Zhang dkk. (2024) mencatat bahwa Large Language Model memperluas ragam kalimat yang dapat dipahami tanpa penambahan aturan secara manual. Alvin dkk. (2025) menunjukkan bahwa penggabungan model bahasa dengan aturan tetap diperlukan agar jawaban tetap berada pada cakupan yang diizinkan.

2.5. Network Development Life Cycle

Network Development Life Cycle merupakan metode pengembangan yang digunakan untuk merancang, membangun, mengimplementasikan, dan mengevaluasi sistem jaringan maupun infrastruktur secara terstruktur melalui tahapan analysis, design, simulation, implementation, monitoring, dan management (Ariyadi dkk., 2024). Sifat siklus pada metode ini memungkinkan hasil pemantauan dijadikan masukan bagi perbaikan rancangan pada iterasi berikutnya. Sopyan dan Rovi (2025) menjelaskan bahwa penerapan metode tersebut membuat pelaksanaan implementasi monitoring menjadi lebih terarah karena setiap tahapan dikerjakan secara berurutan mulai dari analisis kebutuhan hingga pemantauan hasil implementasi.

3. RANCANGAN SISTEM DAN APLIKASI

3.1. Tahapan Penelitian

Penelitian ini merupakan penelitian terapan dengan orientasi pengembangan model. Kerangka teknis yang dipakai adalah Network Development Life Cycle karena penelitian berfokus pada infrastruktur server dan kerangka

tersebut telah terbukti menjaga keteraturan pelaksanaan pada penelitian sejenis (Ariyadi dkk., 2024; Sopyan & Rovi, 2025; Saputra dkk., 2025). Pemetaan tahapan metode terhadap langkah pengembangan model pada penelitian ini disajikan pada Tabel 3.1.

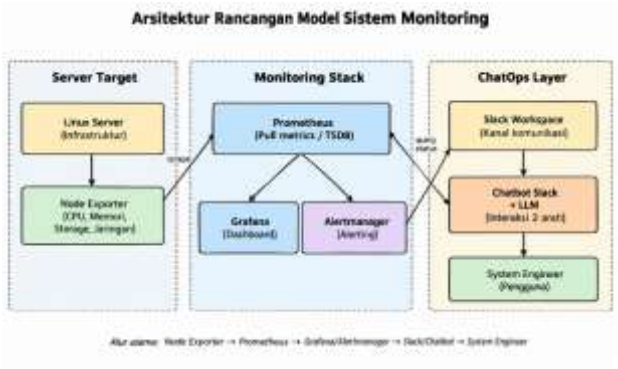
Tabel 3.1 Pemetaan Tahapan NDLC ke Langkah Penelitian

No	Langkah penelitian	Tahapan NDLC
1	Penelitian pendahuluan dan analisis kebutuhan	Analysis
2	Penyusunan rancangan model dan uji terbatas	Design, Simulation
3	Pemasangan dan konfigurasi komponen	Implementation
4	Pengamatan kestabilan sistem dan pengujian	Monitoring
5	Dokumentasi konfigurasi dan pemeliharaan	Management

Data dikumpulkan melalui observasi jalannya komponen sistem, wawancara terbatas dengan *Lead System Engineer* untuk mengonfirmasi kebutuhan operasional dan menilai perubahan setelah sistem berjalan, serta studi dokumentasi acuan teknis masing-masing komponen.

3.2. Rancangan Arsitektur Sistem

Rancangan arsitektur sistem monitoring ditunjukkan pada Gambar 3.1. Node Exporter dipasang pada setiap server sasaran sehingga metrik sistem operasi tersedia melalui protokol HTTP. Prometheus mengambil metrik dari seluruh target pada selang tetap dan menyimpannya sebagai data berbasis waktu. Grafana membaca data tersebut untuk menyajikan dashboard utilisasi CPU, memori, storage, dan jaringan beserta tren historisnya. Alertmanager menerima kondisi alert dari aturan yang ditetapkan lalu meneruskan pemberitahuan ke kanal Slack, sedangkan chatbot berbasis Large Language Model berada pada kanal yang sama untuk menjawab pertanyaan tim mengenai kondisi server.



Gambar 3.1
Rancangan Arsitektur Sistem Monitoring

3.3. Rancangan Fungsi dan Hak Akses

Sistem dirancang untuk melayani tiga peran pengguna dengan kewenangan yang berbeda. System Engineer memantau dashboard, menerima notifikasi, dan menanyakan status sumber daya melalui chatbot. Lead System Engineer memiliki kewenangan tambahan berupa penyesuaian ambang batas alert dan pengelolaan daftar perintah yang diizinkan. Support Team hanya menerima notifikasi pada kanal Slack sebagai dasar eskalasi. Pembatasan perintah diterapkan sebagai kendali keamanan sehingga permintaan di luar daftar yang diizinkan ditolak sistem, sejalan dengan prinsip pemantauan yang tetap menjaga kendali akses (Tuyishime dkk., 2023).

3.4. Lingkungan Implementasi dan Skenario Pengujian

Implementasi dijalankan pada lingkungan simulasi bersistem operasi Linux yang merepresentasikan server cloud perusahaan, sehingga pengujian dapat dilakukan tanpa mengganggu layanan yang sedang berjalan. Ambang batas alert ditetapkan pada 60 persen untuk utilisasi CPU, memori, dan disk. Rincian lingkungan implementasi disajikan pada Tabel 3.2.

Tabel 3.2 Lingkungan Implementasi Sistem Monitoring

Aspek	Keterangan
Lingkungan uji	Simulasi server Linux yang mewakili server cloud perusahaan
Metrik dipantau	CPU, memori, storage, dan jaringan
Komponen monitoring	Node Exporter, Prometheus, Grafana, Alertmanager
Media interaksi	Slack, berupa notifikasi otomatis dan chatbot berbasis LLM
Ambang batas alert	60 persen untuk CPU, memori, dan disk
Kendali keamanan	Pembatasan perintah operasional
Di luar cakupan	Pemantauan performa aplikasi, firewall, dan IDS

Pengujian dirancang pada empat aspek. Pengujian fungsionalitas memeriksa jalannya setiap fitur utama sesuai skenario penggunaan. Pengujian akurasi deteksi anomali dilakukan dengan menaikkan beban sumber daya server hingga melewati ambang batas, sekaligus memastikan alert tidak muncul pada kondisi normal sebagai kontrol negatif. Pengujian kecepatan notifikasi mengukur selisih waktu antara terlampauinya ambang batas dan diterimanya pesan pada kanal Slack dengan target tidak lebih dari 60 detik. Pengujian efektivitas chatbot menilai kesesuaian jawaban terhadap permintaan status sumber daya, termasuk perlakuan terhadap perintah yang tidak terdaftar.

Copyright@2026 FST PSU Bekasi

4. HASIL DAN PEMBAHASAN

4.1. Iterasi Pengembangan Model

Sistem dibangun bertahap agar setiap penambahan kemampuan dapat dievaluasi sebelum melangkah ke tahap berikutnya. Model Draft 1 menyiapkan fondasi pemantauan berupa pengumpulan metrik oleh Prometheus dari Node Exporter beserta visualisasinya pada Grafana. Evaluasi terbatas menunjukkan fondasi tersebut berjalan, namun tim belum memperoleh sinyal proaktif ketika sumber daya mendekati batas aman karena dashboard harus dibuka lebih dahulu. Model Draft 2 menjawab kekurangan itu dengan menambahkan Alertmanager beserta aturan alert dan meneruskan notifikasi ke kanal Slack. Kesadaran tim terhadap anomali meningkat, tetapi komunikasi masih berjalan satu arah sehingga detail kondisi server tetap ditelusuri secara manual. Model Final melengkapi arsitektur dengan chatbot Slack berbasis Large Language Model sehingga tim dapat bertanya dan memperoleh jawaban pada platform yang sama. Perbandingan kemampuan ketiga model dirangkum pada Tabel 4.1.

Tabel 4.1 Perbandingan Kemampuan Antarmodel

Kemampuan sistem	Draft 1	Draft 2	Final
Pengumpulan metrik	Ada	Ada	Ada
Dashboard terpusat	Ada	Ada	Ada
Aturan alert	Belum	Ada	Ada
Notifikasi Slack	Belum	Ada	Ada
Chatbot berbasis LLM	Belum	Belum	Ada
Pembatasan perintah	Belum	Belum	Ada

4.2. Dashboard Terpusat

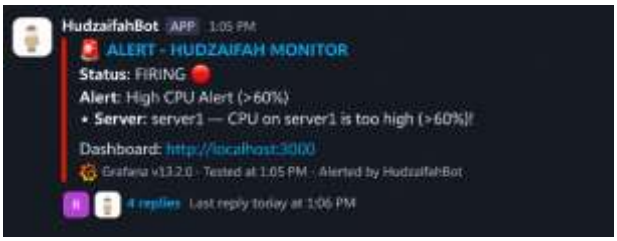
Dashboard Grafana pada Model Final menyajikan utilisasi CPU, memori, storage, dan jaringan dalam satu tampilan beserta tren historisnya seperti terlihat pada Gambar 4.1. Penyajian terpusat ini menggantikan kebiasaan mengakses tiap server satu per satu dan sekaligus menyediakan riwayat pemakaian sumber daya yang sebelumnya tidak terekam, sehingga analisis setelah gangguan dapat dilakukan berdasarkan data.



Gambar 4.1
Dashboard Grafana pada Model Final

4.3. Notifikasi Otomatis dan Lapisan ChatOps

Aturan alert dievaluasi Prometheus dan diteruskan Alertmanager ke kanal Slack ketika utilisasi sumber daya melewati ambang batas 60 persen. Contoh notifikasi yang diterima tim ditunjukkan pada Gambar 4.2. Pesan tersebut memuat nama server, jenis metrik, dan nilai yang terpantau sehingga tim dapat menilai tingkat kegentingan tanpa membuka dashboard terlebih dahulu.



Gambar 4.2
Notifikasi Alert Otomatis pada Kanal Slack

Lapisan ChatOps pada Model Final ditunjukkan pada Gambar 4.3. Tim dapat menanyakan kondisi sumber daya server langsung dari kanal Slack dan menerima jawabannya pada percakapan yang sama, tanpa berpindah ke dashboard maupun melakukan remote ke server. Perintah yang tidak termasuk dalam daftar yang diizinkan ditolak sistem, sehingga keleluasaan percakapan yang dibuka model bahasa tetap dibatasi kendali keamanan.



Gambar 4.3
Interaksi Chatbot Slack pada Model Final

4.4. Hasil Pengujian Fungsionalitas

Pengujian fungsionalitas mencakup lima skenario yang mewakili alur kerja sistem, mulai dari pengambilan metrik sampai penolakan perintah yang tidak diizinkan. Seluruh skenario berhasil dijalankan sebagaimana dirangkum pada Tabel 4.2.

Copyright@2026 FST PSU Bekasi

Tabel 4.2 Hasil Pengujian Fungsionalitas

No	Skenario uji	Status
1	Scraping metrik Node Exporter	Berhasil
2	Tampilan dashboard Grafana	Berhasil
3	Pemicuan alert CPU dan memori	Berhasil
4	Permintaan status melalui chatbot	Berhasil
5	Perintah di luar daftar izin	Berhasil

Target Prometheus berstatus aktif dan keempat kelompok metrik berhasil dikumpulkan, panel dashboard tampil sesuai data yang tersimpan, alert terkirim setelah beban melewati ambang batas, chatbot menjawab permintaan status server, dan perintah di luar daftar izin ditolak beserta pesan penolakan.

4.5. Akurasi Deteksi dan Kecepatan Notifikasi

Akurasi deteksi diuji dengan menaikkan beban sumber daya sampai melewati ambang batas, lalu mengamati kemunculan alert beserta waktu tibanya pada kanal Slack. Hasil pengujian disajikan pada Tabel 4.3.

Tabel 4.3 Hasil Pengujian Deteksi Anomali dan Kecepatan Notifikasi

Jenis anomali	Ambang	Alert	Latensi
Utilisasi CPU tinggi	60%	Ya	< 60 detik
Utilisasi memori tinggi	60%	Ya	< 60 detik
Kapasitas disk menipis	60%	Ya	< 60 detik
Kondisi normal	-	Tidak	-

Ketiga jenis anomali terdeteksi dan seluruh notifikasi memenuhi target waktu, sedangkan pada kondisi normal tidak ada alert yang terkirim sehingga sistem tidak menghasilkan peringatan palsu. Capaian tersebut sejalan dengan keterangan Lead System Engineer bahwa alert diterima kurang dari satu menit sejak anomali terjadi, jauh lebih singkat dibanding jeda puluhan menit hingga lebih dari satu jam pada praktik pemantauan manual sebelumnya.

4.6. Efektivitas Chatbot

Efektivitas chatbot dinilai dari kesesuaian jawaban terhadap permintaan status sumber daya beserta perlakuan terhadap perintah yang tidak diizinkan. Seluruh perintah uji memperoleh respons yang sesuai sebagaimana dirangkum pada Tabel 4.4.

Tabel 4.4 Hasil Pengujian Efektivitas Chatbot

Perintah uji	Respons sistem	Kesesuaian
Cek status CPU	Utilisasi CPU server ditampilkan	Sesuai
Cek status memori	Utilisasi memori ditampilkan	Sesuai
Cek status disk	Pemakaian disk ditampilkan	Sesuai
Perintah di luar daftar izin	Perintah ditolak beserta pesan peringatan	Sesuai

Pembatasan perintah menjadi penting karena chatbot berbasis Large Language Model menerima masukan berbahasa alami yang bentuknya sangat beragam. Daftar perintah yang diizinkan memastikan model bahasa hanya menjadi lapisan penafsir maksud, sedangkan tindakan terhadap server tetap dibatasi pada operasi yang telah ditetapkan (Alvin dkk., 2025).

4.7. Pembahasan

Hasil pengujian menunjukkan bahwa pemantauan infrastruktur server dapat ditingkatkan melalui tiga lapisan yang saling melengkapi. Lapisan pertama adalah pengamatan terpusat melalui dashboard, yang menggantikan pemeriksaan server satu per satu. Lapisan kedua adalah kesadaran dini melalui notifikasi otomatis, yang memindahkan beban pemantauan dari inisiatif manusia ke mekanisme sistem. Lapisan ketiga adalah kemudahan penelusuran melalui percakapan, yang memangkas jarak antara munculnya pertanyaan dan tersedianya jawaban.

Dua lapisan pertama sejalan dengan temuan Saory dkk. (2024) yang memakai kanal Slack serta Rasyidi dan Pratama (2024) yang memakai kanal Telegram, dan konsisten dengan manfaat penyatuan visualisasi yang dilaporkan Saputra dkk. (2025) maupun Defani dan Wijayanto (2025). Perbedaan penelitian ini terletak pada lapisan ketiga. Pada penelitian tersebut kanal komunikasi berperan sebagai penerima pesan, sedangkan pada penelitian ini kanal yang sama berperan pula sebagai antarmuka bertanya. Perubahan arah komunikasi itu memungkinkan tim memperoleh detail kondisi server tanpa berpindah alat, sesuai peluang percakapan luwes yang dibuka Large Language Model (Zhang dkk., 2024).

Pola perbaikan bertahap dari Model Draft 1 hingga Model Final juga memperlihatkan kesesuaian Network Development Life Cycle untuk penelitian implementasi. Setiap iterasi lahir dari kekurangan yang teridentifikasi pada iterasi sebelumnya, sehingga penambahan komponen selalu memiliki dasar kebutuhan yang jelas. Temuan ini memperkuat laporan Ariyadi dkk. (2024) serta Sopyan dan Rovi (2025) mengenai keteraturan pelaksanaan implementasi ketika metode tersebut dipakai sebagai kerangka kerja.

Keterbatasan penelitian perlu dicatat. Efektivitas sistem bergantung pada ketepatan ambang batas alert, mutu konteks yang diberikan kepada model bahasa, serta

kedisiplinan pengelolaan daftar perintah yang diizinkan. Sistem juga bergantung pada ketersediaan layanan model bahasa pihak ketiga, sehingga gangguan pada layanan tersebut berdampak pada lapisan percakapan meskipun lapisan monitoring tetap berjalan. Selain itu, cakupan pemantauan masih terbatas pada metrik infrastruktur dan pengujian dilakukan pada rentang waktu yang relatif singkat, sehingga stabilitas jangka panjang dan tingkat peringatan palsu belum terukur.

5. KESIMPULAN DAN SARAN

Sistem monitoring infrastruktur server berbasis Prometheus dan Grafana yang diintegrasikan dengan chatbot Slack berbasis Large Language Model berhasil diimplementasikan menggunakan metode Network Development Life Cycle melalui tiga iterasi model. Model Draft 1 membangun fondasi pengumpulan dan visualisasi metrik, Model Draft 2 menambahkan notifikasi otomatis, dan Model Final melengkapinya dengan interaksi dua arah beserta pembatasan perintah.

Pengujian menunjukkan lima skenario fungsional berhasil dijalankan seluruhnya, anomali CPU, memori, dan disk terdeteksi pada ambang batas 60 persen tanpa peringatan palsu pada kondisi normal, notifikasi diterima pada kanal Slack kurang dari 60 detik, dan chatbot menjawab permintaan status sumber daya sekaligus menolak perintah yang tidak terdaftar. Dengan demikian sistem memberikan pengamatan terpusat, kesadaran dini terhadap anomali, dan akses informasi operasional yang lebih cepat dibanding pemantauan manual.

Penelitian selanjutnya disarankan memperluas cakupan pemantauan sampai tingkat performa aplikasi, menguji sistem pada jumlah server dan rentang waktu yang lebih besar untuk mengukur stabilitas serta tingkat peringatan palsu, dan menambahkan pengukuran kuantitatif terhadap waktu penanganan insiden sebelum dan sesudah sistem digunakan.

DAFTAR PUSTAKA

Alvin, A., Robet, R., & Tarigan, F. A. (2025). Implementasi chatbot otomatis akademik berbasis web menggunakan LLM dan rule-based system. JIKO (Jurnal Informatika dan Komputer), 9(3), 651-665.

Ariyadi, T., Fikri, M., Irwansyah, & Yudiastuti, H. (2024). Penerapan monitoring jaringan dengan Zabbix pada PT. PLN (Persero) UIP Bagian Sumbagsel. JUITA (Jurnal Informatika Universitas Tulang Bawang), 8(2).

Chapman, R., & Holmes, P. (2023). Observability with Grafana: Monitor, control, and visualize. Packt Publishing.

Defani, R., & Wijayanto, D. (2025). Integrasi sistem monitoring aplikasi Uptime Kuma dan router MikroTik menggunakan Prometheus dan Grafana. Sistemasi: Jurnal Sistem Informasi, 14(5), 2395-2409.

Grafana Labs. (2023). Grafana documentation: Alerting. Grafana Labs.

Hegedus, W. (2024). Mastering Prometheus: Gain expert tips to monitoring your infrastructure, applications, and services. Packt Publishing.

Izza, K. W. B., & Sholeha, E. W. (2025). Implementasi sistem monitoring server berbasis Linux pada ICT Politeknik Negeri Tanah Laut. Information Technology Trends, 2(2).

Kusuma, G. Y., & Oktiawati, U. Y. (2022). Perancangan sistem monitoring performa aplikasi menggunakan OpenTelemetry dan Grafana Stack. Journal of Internet and Software Engineering (JISE), 3(1).

Matic, R., Kabiljo, M., Zivkovic, M., & Cabarkapa, M. (2021). Extensible chatbot architecture using metamodels of natural language understanding. Electronics, 10(2), 2300.

Pradana, A., Widiasari, I. R., & Efendi, R. (2022). Implementasi sistem monitoring jaringan menggunakan Zabbix berbasis SNMP. AITI: Jurnal Teknologi Informasi, 19(2), 248-262.

Putra, K. A. P., Sukadarmika, G., & Wiharta, D. M. (2023). Model utilisasi dan visualisasi resource menggunakan Prometheus dan Grafana untuk pengelolaan server di Universitas Udayana. Majalah Ilmiah Teknologi Elektro, 22(2).

Ramadoni, Amirudin, M. Z., Fahmi, R., Utami, E., & Mustafa, M. S. (2021). Evaluasi penggunaan Prometheus dan Grafana untuk monitoring database MongoDB. JIP (Jurnal Informatika Polinema).

Rasyidi, B., & Pratama, F. (2024). Server monitoring system at PT. XYZ Media Indonesia based on Grafana and Prometheus. MALCOM: Indonesian Journal of Machine Learning and Computer Science, 4(4), 1456-1465.

Saory, F., Jaman, J. H., & Carudin. (2024). Sistem monitoring server menggunakan Prometheus dan Grafana dengan integrasi Slack di PT. Atlas Lintas Indonesia. JITET (Jurnal Informatika dan Teknik Elektro Terapan), 13(3S1).

Saputra, M. D., Husain, & Marzuki, K. (2025). Desain dan implementasi monitoring infrastruktur perangkat jaringan dan server menggunakan Grafana dan Prometheus. JATISI (Jurnal Teknik Informatika dan Sistem Informasi), 12(3), 29-47.

Slack Technologies. (2023). Slack API documentation. Slack.

Sopyan, Y., & Rovi, M. (2025). Implementasi metode NDLC pada infrastruktur jaringan Universitas Bhakti Husada Indonesia dengan memanfaatkan WhatsApp Gateway sebagai monitoring. INFOTECH Journal, 11(2), 174-183.

Tuyishime, E., Balan, T. C., Cotfas, P. A., Cotfas, D. T., & Rekeraho, A. (2023). Enhancing cloud security: Proactive threat monitoring and detection using a

SIEM-based approach. Applied Sciences, 13(22), 12359.

Yulvianda, R., & Ismail, M. (2023). Desain dan implementasi sistem monitoring sumber daya server menggunakan Zabbix dan Grafana. Jurnal Informatika dan Rekayasa Komputer (JAKAKOM), 3(1).

Zhang, L., Li, Y., Wu, C., & Li, C. (2024). Business chatbots in the era of large language models: A literature review. Artificial Intelligence Review, 57(9).

Copyright@2026 FST PSU Bekasi