Jurnal Esensi Infokom Vol 10 No. 1 Mei 2026
e-ISSN: 2828-6707
https://esensijournal.com/index.php/infokom
DOI: 10.55886/infokom.v10i1.459

DevOps Berbasis Infrastructure as Code (IaC): Tinjauan Komprehensif terhadap Implementasi, Tantangan, dan Arah Masa Depan dalam Pipeline CI/CD Modern

Yogi Priya Agsena[1]*, Ariawan Aryapranata[2], Yuliansyah Al Rasyid[3], Yusuf Hidayat[4]
[1,2,3,4]*Program Studi Bisnis Digital, Institut Pariwisata Trisakti*
Jakarta, Indonesia
[1*]yogi.agsena@iptrisakti.ac.id (penulis korespondensi)
[2] ariawan.aryapranata@iptrisakti.ac.id
[3]yuliansyah@iptrisakti.ac.id
[4]yusufhidayat554@gmail.com

Intisari— **Transformasi digital mendorong organisasi untuk mengadopsi pendekatan pengembangan perangkat lunak yang mampu memberikan kecepatan, reliabilitas, dan skalabilitas tinggi. DevOps muncul sebagai paradigma integratif yang menyatukan pengembangan dan operasi melalui otomatisasi serta kolaborasi lintas fungsi. Infrastructure as Code (IaC) berperan sebagai enabler utama DevOps dengan memungkinkan provisioning dan konfigurasi infrastruktur dilakukan secara otomatis, terstandarisasi, dan dapat direplikasi. Penelitian ini bertujuan memberikan tinjauan komprehensif mengenai integrasi DevOps dan IaC dalam pipeline Continuous Integration dan Continuous Delivery/Deployment (CI/CD), termasuk praktik implementasi, ekosistem alat, tantangan kualitas dan keamanan, serta arah penelitian masa depan. Metode yang digunakan adalah studi literatur sistematis terhadap publikasi ilmiah terkait DevOps dan IaC. Hasil kajian menunjukkan bahwa integrasi IaC meningkatkan konsistensi deployment, efisiensi operasional, serta ketahanan sistem, namun juga memunculkan risiko baru berupa cacat konfigurasi dan kerentanan keamanan. Perkembangan terbaru menunjukkan potensi pemanfaatan kecerdasan buatan untuk otomatisasi DevOps yang lebih otonom.**
Kata kunci— DevOps, Infrastructure as Code, CI/CD, Cloud Computing, DevSecOps

***Abstract*— Digital transformation compels organizations to adopt software development approaches that provide high speed, reliability, and scalability. DevOps has emerged as an integrative paradigm that unifies development and operations through automation and cross-functional collaboration. Infrastructure as Code (IaC) serves as a key enabler of DevOps by enabling automated, standardized, and reproducible infrastructure provisioning. This study provides a comprehensive review of DevOps–IaC integration within modern CI/CD pipelines, including implementation practices, tooling ecosystems, quality and security challenges, and future research directions. A systematic literature review method is employed to analyze relevant scientific publications. The findings indicate that IaC integration improves deployment consistency and operational efficiency but introduces new risks such as configuration defects and security vulnerabilities. Recent advances highlight the potential of artificial intelligence to enable more autonomous DevOps processes**
Keywords— DevOps, Infrastructure as Code, CI/CD, Cloud Computing, DevSecOps
.

I. PENDAHULUAN

Perkembangan teknologi informasi yang pesat menuntut organisasi untuk mengembangkan perangkat lunak secara cepat, adaptif, dan andal. Metodologi tradisional yang memisahkan tim pengembangan dan operasional seringkali menyebabkan keterlambatan rilis serta inkonsistensi lingkungan sistem. DevOps muncul sebagai pendekatan yang mengintegrasikan kedua fungsi tersebut melalui otomatisasi dan kolaborasi lintas tim untuk mempercepat siklus pengembangan perangkat lunak [1].

Salah satu komponen utama yang memperkuat implementasi DevOps adalah Infrastructure as Code (IaC). IaC memungkinkan konfigurasi dan penyediaan infrastruktur dilakukan menggunakan kode yang dapat diuji, dikelola versinya, dan direplikasi pada berbagai lingkungan [2]. Pendekatan ini sangat penting dalam arsitektur cloud-native dan mikroservis yang membutuhkan skalabilitas tinggi serta konsistensi konfigurasi [3].

Integrasi DevOps dan IaC menghasilkan pipeline Continuous Integration dan Continuous Delivery/Deployment (CI/CD) yang mampu mengotomatisasi seluruh siklus hidup perangkat lunak, mulai dari pembangunan kode hingga deployment dan monitoring [4]. Namun demikian, penggunaan IaC juga membawa tantangan baru seperti potensi kesalahan konfigurasi, risiko keamanan, serta kompleksitas manajemen perubahan [5].

II. LATAR BELAKANG

DevOps merupakan pendekatan pengembangan modern yang menekankan kolaborasi antara tim pengembang dan operasional untuk meningkatkan kualitas dan kecepatan delivery aplikasi. Pendekatan ini mengandalkan otomatisasi,

Jurnal Esensi Infokom Vol 10 No. 1 Mei 2026
e-ISSN: 2828-6707
https://esensijournal.com/index.php/infokom
DOI: 10.55886/infokom.v10i1.459

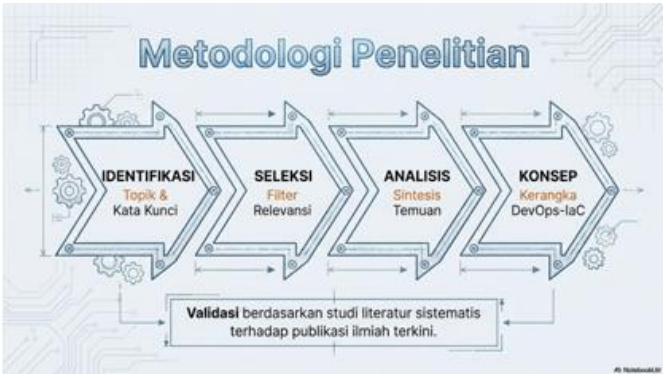
monitoring berkelanjutan, serta feedback cepat untuk mendukung continuous delivery [1], [6].

Infrastructure as Code menjadi komponen penting dalam ekosistem DevOps karena memungkinkan provisioning infrastruktur dilakukan secara otomatis dan konsisten. Dengan IaC, konfigurasi server, jaringan, dan sumber daya cloud dapat didefinisikan dalam bentuk kode sehingga dapat dilacak perubahan versinya dan direproduksi kapan saja [2].

Dalam arsitektur cloud modern, penggunaan mikroservis dan container meningkatkan kompleksitas sistem sehingga memerlukan orkestrasi infrastruktur yang efisien. IaC memungkinkan provisioning komponen terdistribusi secara otomatis dan mendukung skalabilitas sistem [3]. Integrasi dengan teknologi container seperti Docker dan orkestrasi Kubernetes semakin memperkuat implementasi DevOps berbasis cloud [7].

III. METODOLOGI PENELITIAN

Penelitian ini menggunakan metode studi literatur sistematis terhadap publikasi ilmiah yang relevan dengan DevOps dan Infrastructure as Code. Literatur dipilih berdasarkan kualitas sumber, relevansi terhadap topik, serta kontribusi terhadap pemahaman konsep dan implementasi DevOps berbasis IaC.



Gambar 1. Metodologi Penelitian

Tahapan penelitian meliputi:

1. Identifikasi kata kunci dan topik penelitian
2. Seleksi literatur berdasarkan relevansi
3. Analisis isi dan sintesis temuan
4. Penyusunan kerangka konseptual

Pendekatan ini memungkinkan peneliti memperoleh gambaran komprehensif mengenai praktik implementasi DevOps berbasis IaC serta tantangan yang dihadapi [5].

IV. HASIL DAN PEMBAHASAN

Integrasi IaC dalam pipeline CI/CD memungkinkan otomatisasi provisioning infrastruktur dan deployment aplikasi secara end-to-end. Setiap perubahan kode dapat memicu proses build, testing, deployment, dan monitoring secara otomatis [4].

Strategi deployment modern seperti *blue-green deployment* dan *canary release* memungkinkan rilis aplikasi dilakukan secara bertahap dengan risiko minimal terhadap sistem produksi [8]. Implementasi DevOps berbasis IaC memanfaatkan berbagai alat seperti Terraform dan Ansible untuk provisioning, Jenkins atau GitLab CI untuk pipeline otomatis, Docker untuk containerization, serta Kubernetes untuk orkestrasi [7].

Meskipun memberikan manfaat signifikan, IaC juga menghadirkan tantangan berupa cacat konfigurasi, kerentanan keamanan, serta kompleksitas manajemen perubahan. Analisis statis dan pengujian otomatis diperlukan untuk memastikan keandalan dan keamanan infrastruktur [5].

TABEL 1
DEVOPS BERBASIS INFRASTRUCTURE AS A CODE (IAC)

Topik Utama	Komponen Teknologi	Manfaaat Implementasi	Tantangan dan Risiko	Startegi Deploymen	Alat yang digunakan	Arah Masa Depan (Infrred)
DevOps Berbasis Infrastructure as Code (IaC)	Infrastruktur Cloud-native, Mikroservis Container, Orkestrasi, Pipeline CI/CD, Monitoring	Otomatisasi provisioning, konsistensi deployment efisiensi operasional, reliabilitas skalabilitas tinggi, ketahanan sistem dan percepatan siklus pengembangan	Cacat konfigurasi, kerentanan keamanan (security vulnerabilities). kompleksitas manajamen perubahan dan risiko inkonsistensi jika tidak terstandararisasi	Blue-green deployment, Canary release, Provisioning otomatis, Rollback startegies	Terraform, Ansible, Jenkins, Gitlab CI, Docker, Kubernetes, CircleCI	Integrasi AL dan Large Language Models (LLM) untuk pembuatan kode infrastruktur, deteksi anomaly konfigurasi otomatis dan transisi menuju Autinomous DevOps

Perkembangan terbaru menunjukkan integrasi kecerdasan buatan dalam proses DevOps, termasuk penggunaan model bahasa besar untuk menghasilkan kode infrastruktur dan mendeteksi anomali konfigurasi [9]. Konsep autonomous DevOps berpotensi meningkatkan efisiensi operasional, namun memerlukan tata kelola yang ketat untuk memastikan keandalan dan kepatuhan sistem [10].

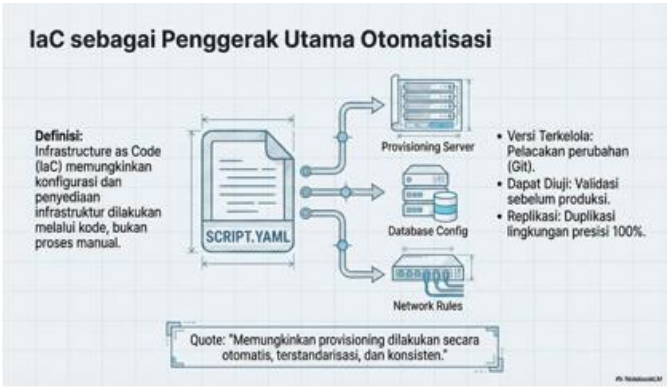
Kerangka Tata Kelola Infrastructure as Code (IaC): Strategi Mitigasi Risiko Konfigurasi dan Keamanan dalam Ekosistem DevSecOps

1. Arsitektur Strategis: Integrasi DevOps dan Infrastructure as Code (IaC)

Dalam lanskap transformasi digital kontemporer, Infrastructure as Code (IaC) bukan sekadar pilihan teknis, melainkan mandat strategis bagi organisasi yang mengejar skalabilitas dan reliabilitas tinggi. Berdasarkan perspektif arsitektur sistem tingkat lanjut, dapat ditegaskan bahwa pergeseran dari manajemen infrastruktur tradisional yang manual menuju

Jurnal Esensi Infokom Vol 10 No. 1 Mei 2026
e-ISSN: 2828-6707
https://esensijournal.com/index.php/infokom
DOI: 10.55886/infokom.v10i1.459

otomatisasi berbasis kode adalah fondasi utama untuk menghilangkan limitasi operasional. IaC mengubah penyediaan sumber daya menjadi proses yang dapat diprediksi, terukur, dan sepenuhnya dapat diaudit.



Gambar 2. IaC Sebagai Penggerak Utama Otomatisasi

Integrasi antara metodologi DevOps dan kapabilitas IaC menciptakan sinergi operasional yang krusial melalui prinsip-prinsip berikut:

1. **Konsistensi Deployment dan Environment Parity:** IaC menjamin bahwa lingkungan pengembangan, pengujian, dan produksi tetap identik. Hal ini merupakan prasyarat mutlak bagi *compliant change management* untuk meminimalkan kegagalan rilis akibat perbedaan konfigurasi.

2. **Efisiensi Operasional Terakselerasi:** Otomatisasi menghilangkan intervensi manual yang rentan kesalahan, mempercepat *time-to-market* tanpa mengorbankan standar kualitas.

3. **Replikasi dan Resiliensi Sistem:** Kemampuan untuk mereproduksi seluruh tatanan infrastruktur secara instan dari repositori kode memungkinkan pemulihan bencana (*disaster recovery*) yang lebih cepat dan akurat.

Meskipun efisiensi ini menawarkan keunggulan kompetitif, adopsi otomatisasi skala besar tanpa kendali yang tepat akan memperluas permukaan serangan, sehingga memerlukan pengawasan tata kelola yang jauh lebih ketat.

2. Taksonomi Risiko: Tantangan Keamanan dan Cacat Konfigurasi

Otomatisasi infrastruktur membawa risiko sistemik yang signifikan; satu kesalahan kecil dalam kode IaC dapat mereplikasi kerentanan ke seluruh pipeline produksi secara instan. Sebagai konsultan tata kelola, saya mengidentifikasi "sisi gelap" otomatisasi ini sebagai ancaman yang harus dimitigasi melalui kontrol preventif, bukan sekadar deteksi reaktif.

Tiga kategori risiko utama yang berdampak langsung pada kelangsungan bisnis meliputi:

1. **Cacat Konfigurasi (*Configuration Defects*):** Kesalahan logika dalam definisi kode yang menyebabkan instabilitas sistem. **Dampak Bisnis:** Lonjakan *Total Cost of Downtime* dan degradasi kepercayaan pengguna akibat layanan yang tidak tersedia.

2. **Kerentanan Keamanan (*Security Vulnerabilities*):** Pengaturan hak akses yang berlebihan (*excessive privileges*) atau paparan data sensitif. **Dampak Bisnis:** Risiko *Regulatory Non-Compliance* terhadap standar global seperti GDPR atau ISO 27001, yang berujung pada denda finansial dan kerusakan reputasi permanen.

3. **Manajemen Perubahan dan *Configuration Drift*:** Ketidakteraturan dalam melacak perubahan menyebabkan kondisi di mana status riil cloud tidak lagi sesuai dengan kode yang diaudit. **Dampak Bisnis:** *Drift* adalah "kerentanan keamanan senyap" (*silent security vulnerability*) yang mempersulit audit kepatuhan dan melemahkan postur keamanan organisasi secara keseluruhan.

Oleh karena itu, diperlukan kerangka kerja proaktif yang mampu mengunci konfigurasi pada standar tertinggi sejak tahap inisiasi.

3. Standarisasi Pipeline CI/CD dan Strategi Deployment Modern

Pipeline CI/CD yang terstandarisasi berfungsi sebagai instrumen kontrol kualitas otomatis dan gerbang utama penegakan kebijakan (*policy enforcement*). Tanpa pipeline yang kokoh, IaC hanya akan menjadi alat untuk mempercepat distribusi kegagalan.

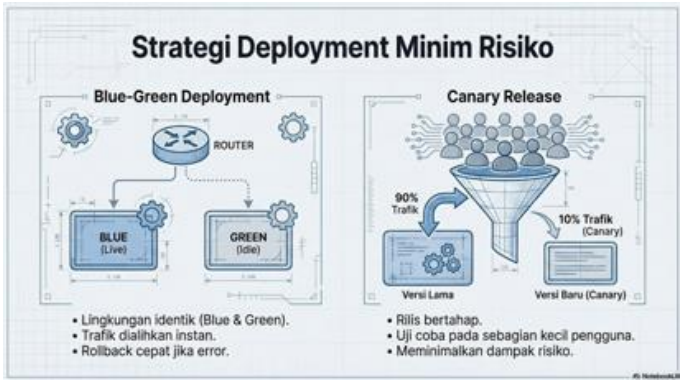
Berikut adalah pemetaan ekosistem alat dan kontribusinya terhadap kepatuhan tata kelola:

TABEL 2.
PEMETAAN EKOSISTEM ALAT DAN KONTRIBUSINYA TERHADAP KEPATUHAN TATA KELOLA

Kategori	Teknologi	Fungsi Utama	Konstribusi Terhadap Keputusan (Compliance)
Provisioning	Terraform, Ansible	Definisi infrastruktur dan konfigurasi server	*Auditability* melalui file status dan Riwayat versi yang transparan
Orchestration	Kubernetes, Docker	Pengelolaan container dan komponen terdistribusi.	Penegakan kebijakan otomatis melalui *Network Policies* dan *Runtime Security*
Automation	Jenkins, GitLab CI	Otomatisasi alur kerja *build* hingga rilis	Penyediaan log audit yang tidak dapat diubah (*immutable*) untuk setiao deployment.

Untuk memitigasi risiko saat pembaruan, organisasi wajib menerapkan strategi deployment modern:

- **Blue-Green Deployment:** Meminimalkan risiko produksi dengan menyediakan lingkungan paralel untuk *rollback* instan.
- **Canary Release:** Validasi fitur pada segmen pengguna terbatas untuk memastikan stabilitas sistem sebelum rilis penuh.



Gambar 3. Strategi Deployment

Mekanisme ini harus diintegrasikan dengan lapisan validasi otomatis yang lebih mendalam untuk mencapai standar DevSecOps yang paripurna.

4. Integrasi DevSecOps: Audit Otomatis dan Policy as Code (PaC)

Transisi menuju DevSecOps menuntut penerapan prinsip *Shift-Left Security*, di mana keamanan bukan lagi tahap akhir, melainkan komponen integral dari siklus pengembangan. Dalam kerangka tata kelola ini, keamanan harus dikelola sebagai kode (*Policy as Code*).

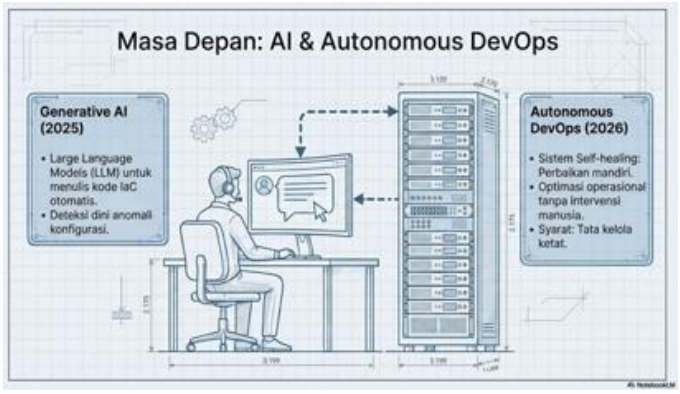
Metode mitigasi risiko yang wajib diterapkan meliputi:

1. **Analisis Statis (*Static Analysis*):** Bukan sekadar pemeriksaan teknis, tetapi penegakan otomatis kebijakan organisasi. Dengan memindai kode IaC sebelum eksekusi, kita dapat mendeteksi anomali seperti grup keamanan yang terlalu terbuka, sehingga secara drastis mengurangi *human-error surface area*.
2. **Pengujian Otomatis:** Validasi fungsional untuk memastikan infrastruktur berperilaku sesuai parameter keamanan yang ditetapkan sebelum mencapai lingkungan cloud.

Sinergi kedua metode ini memastikan bahwa setiap baris kode yang masuk ke produksi telah melalui filter keamanan berlapis. Namun, seiring meningkatnya kompleksitas arsitektur *cloud-native*, tata kelola masa depan akan menuntut adaptabilitas yang lebih cerdas.

5. Visi Masa Depan: Kecerdasan Buatan dan Autonomous DevOps

Masa depan tata kelola infrastruktur akan didominasi oleh kecerdasan buatan (AI) untuk mengelola kompleksitas lingkungan cloud yang kian masif. Tren *Autonomous DevOps* menjanjikan sistem yang mampu melakukan penyembuhan mandiri (*self-healing*) dan deteksi anomali secara otonom.



Gambar 4. Autonomous DevOps

Dua tren transformatif yang perlu diperhatikan:

1. **Generasi Kode berbasis LLM:** Penggunaan Model Bahasa Besar untuk mempercepat penulisan IaC. Namun, terdapat risiko kritis berupa **"Hallucinated Vulnerabilities"**, di mana AI mungkin menyarankan parameter keamanan yang usang atau tidak aman.
2. **Autonomous DevOps:** Sistem yang mampu mendeteksi dan mengoreksi penyimpangan konfigurasi secara otomatis.

Dalam ekosistem ini, peran manusia bergeser dari sekadar penulis kode menjadi **auditor kebijakan cerdas**. Tata kelola manusia yang ketat tetap menjadi jangkar utama untuk

memastikan bahwa keputusan otonom AI tetap selaras dengan regulasi dan tujuan strategis perusahaan.

6. Sintesis Strategis

Keberhasilan implementasi IaC bukan hanya tentang kecanggihan alat, melainkan tentang integrasi antara keunggulan teknis dan ketegasan struktur tata kelola. Tanpa kendali yang kuat, otomatisasi adalah risiko; dengan tata kelola yang tepat, IaC adalah katalisator utama resiliensi bisnis.

Rekomendasi Strategis untuk Eksekutif dan Arsitek Sistem:

TABEL 3
STRATEGIS BAGI EKSEKUTIF DAN ARSITEK

No.	Aspek Strategis	Deskripsi Implementasi
1	Standarisasi *Tooling*	Organisasi perlu mewajibkan penggunaan alat provisi dan orkestrasi yang seragam di seluruh unit bisnis guna menjamin konsistensi operasional serta meningkatkan transparansi audit infrastruktur.
2	Implementasi *Policy as Code* (PaC)	Penerapan *Policy as Code* dilakukan melalui integrasi analisis statis pada setiap proses *commit* untuk memastikan standar keamanan dan kepatuhan diterapkan secara otomatis.
3	Eliminasi *Configuration Drift*	Diperlukan mekanisme pemantauan berkelanjutan untuk mendeteksi serta memperbaiki deviasi konfigurasi secara *real-time* guna menjaga konsistensi sistem infrastruktur.
4	*Deployment* Berisiko Rendah	Strategi *Blue-Green Deployment* atau *Canary Deployment* perlu diterapkan untuk meminimalkan dampak operasional apabila terjadi kegagalan sistem selama proses implementasi.
5	Audit *AI-Generated Code*	Organisasi perlu menetapkan protokol peninjauan oleh tenaga ahli terhadap seluruh kode infrastruktur yang dihasilkan oleh kecerdasan buatan untuk mengurangi potensi kerentanan akibat halusinasi sistem AI.

Mengadopsi pendekatan komprehensif ini adalah satu-satunya cara untuk memastikan bahwa infrastruktur organisasi tetap tangguh, aman, dan siap menghadapi tantangan di era cloud modern.

V. KESIMPULAN

Integrasi DevOps dan Infrastructure as Code merupakan evolusi penting dalam pengembangan perangkat lunak modern. IaC menyediakan mekanisme otomatisasi dan konsistensi yang diperlukan untuk pipeline CI/CD yang cepat dan andal, sementara DevOps menyediakan kerangka budaya dan organisasi yang mendukung implementasi tersebut.

Meskipun memberikan manfaat signifikan dalam efisiensi operasional dan skalabilitas, penerapan IaC juga memunculkan risiko baru terkait keamanan dan kualitas konfigurasi. Oleh karena itu, pendekatan komprehensif yang mencakup aspek teknis, organisasi, dan tata kelola diperlukan untuk memastikan keberhasilan implementasi DevOps berbasis IaC di lingkungan cloud modern.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Institut Pariwisata Trisakti dan pihak yang telah memberikan dukungan dalam penyusunan penelitian ini.

REFERENSI.

[1] D. Sokolowski, “Deployment coordination for cross-functional DevOps teams,” 2021.
[2] S. Ranjan, “Empowering DevOps with Infrastructure as Code: Trends, Tools and Techniques,” 2024
[3] P. Ganore, “Microservices and DevOps: Achieving Scalability and Agility in Cloud Architectures,” 2021.
[4] A. Saxena et al., “DevOps Automation Pipeline Deployment with IaC,” 2024.
[5] A. Rahman et al., “A systematic mapping study of infrastructure as code research,” Information and Software Technology, vol. 108, 2019.
[6] E. Romero et al., “Integration of DevOps Practices on a Noise Monitor System with CircleCI and Terraform,” ACM Trans. MIS, 2022.
[7] N. Koneru, “Automating CI/CD Pipelines Using Terraform and GitLab,” 2021.
[8] Y. Avuthu, “Change management and rollback strategies using IaC in CI/CD pipelines,” 2021.
[9] S. Joshi, “Generative AI and DevOps Pipelines,” 2025.
[10] C. Pahl, “Artificial Intelligence for Infrastructure-as-Code,” Electronics, 2026.