

Context-Aware Incident Prioritization Using Large Language Models and Automated Remediation in Enterprise Cloud Systems

Dr. V. P. Gladis Pushparathi

Professor, Department of Computer Science & Engineering, Velammal Institute of Technology, Chennai, India

ABSTRACT

The increasing complexity of enterprise cloud environments has resulted in a growing number of operational incidents involving infrastructure failures, application disruptions, security vulnerabilities, performance degradation, and resource availability issues. Traditional incident management systems primarily depend on predefined rules, static severity classifications, and manual investigation, which may limit their ability to understand complex operational contexts and respond effectively to dynamic cloud conditions. This study proposes a context-aware incident prioritization and automated remediation framework integrating Large Language Models (LLMs), cloud monitoring technologies, and intelligent decision-making mechanisms. The proposed framework analyzes incident descriptions, system logs, monitoring metrics, application dependencies, historical incident records, and service-level agreement requirements to determine incident severity and operational priority. Large Language Models are employed to interpret unstructured incident information, identify relevant contextual relationships, generate incident summaries, and recommend appropriate remediation strategies. A context-aware prioritization mechanism evaluates factors such as business impact, urgency, affected services, infrastructure dependencies, security implications, and recovery requirements. Automated remediation agents execute authorized corrective actions, including service restarts, resource scaling, configuration recovery, and workload redistribution. The proposed methodology emphasizes experimental evaluation using incident classification accuracy, prioritization effectiveness, mean time to resolution, remediation success rate, and operational efficiency. The framework aims to reduce manual intervention, accelerate incident resolution, improve service reliability, and strengthen intelligent operational management across enterprise cloud infrastructures.

Keywords: Context-Aware Incident Management, Large Language Models, Automated Remediation, Enterprise Cloud Systems, Incident Prioritization, Artificial Intelligence, Cloud Monitoring, Intelligent Automation, Root Cause Analysis, Site Reliability Engineering, Operational Intelligence.

International Journal of Technology, Management and Humanities (2025)

DOI: 10.21590/ijtmh.11.04.XX

INTRODUCTION

Enterprise cloud computing has become a fundamental component of modern organizational infrastructure, supporting business-critical applications, distributed databases, virtualized environments, containerized services, and large-scale digital operations. As enterprises increasingly adopt hybrid cloud, multi-cloud, and microservices architectures, the complexity of managing infrastructure reliability and operational performance continues to increase. Cloud environments generate substantial volumes of monitoring information through application logs, infrastructure metrics, network events, security alerts, and system notifications. These data sources provide essential information about operational conditions but also introduce significant challenges for incident management teams. A single infrastructure failure can generate hundreds of related alerts across interconnected services, making it difficult to

Corresponding Author: Dr. V. P. Gladis Pushparathi, Professor, Department of Computer Science & Engineering, Velammal Institute of Technology, Chennai, India.

How to cite this article: Pushparathi, V. P. G. (2025). Context-Aware Incident Prioritization Using Large Language Models and Automated Remediation in Enterprise Cloud Systems. *International Journal of Technology, Management and Humanities*, 11(4), 293-301.

Source of support: Nil

Conflict of interest: None

identify the actual cause, assess business impact, and determine appropriate response priorities. Traditional incident management systems commonly use static severity levels, predefined thresholds, keyword matching, and manually maintained escalation policies. Although these mechanisms support basic incident classification, they often

lack the contextual understanding required to distinguish between technically similar incidents with significantly different business consequences. For example, a temporary performance degradation affecting an internal development environment may have a relatively low operational priority, whereas a similar issue affecting a customer-facing financial transaction service may require immediate intervention. Therefore, enterprise cloud systems require intelligent incident management approaches that evaluate technical conditions alongside application dependencies, organizational requirements, business criticality, and service-level agreements.

Large Language Models (LLMs) provide new opportunities for improving incident management through their ability to interpret natural language, analyze unstructured operational information, identify relationships between events, and generate contextually relevant recommendations. By integrating LLMs with cloud monitoring platforms, organizations can develop intelligent systems capable of analyzing incident descriptions, correlating system logs, summarizing operational events, and recommending appropriate corrective actions. Context-aware incident prioritization extends traditional severity classification by incorporating multiple dimensions, including incident urgency, affected infrastructure, application dependencies, customer impact, security exposure, historical incident patterns, and recovery requirements. This approach enables incident management systems to assign priorities according to the actual operational significance of an event rather than relying exclusively on isolated technical indicators. Furthermore, automated remediation mechanisms can extend intelligent incident analysis into operational response by executing authorized corrective actions. Such actions may include restarting failed services, scaling computational resources, restoring configuration settings, redirecting workloads, and initiating recovery procedures. Combining LLM-based reasoning with automated remediation agents can create an integrated incident management architecture that supports detection, contextual analysis, prioritization, recommendation, and response. However, reliable implementation requires addressing important challenges involving model accuracy, hallucination, security, access control, execution safety, and human oversight. The present research investigates a context-aware incident prioritization framework that combines LLM-based incident understanding with controlled automated remediation in enterprise cloud environments. Its primary objective is to improve incident classification, reduce response delays, minimize operational disruption, and strengthen service reliability. The study also examines how contextual information and historical operational knowledge can improve remediation decisions while maintaining governance and accountability. Through this integrated approach, the research aims to establish an intelligent and

scalable foundation for next-generation enterprise cloud incident management.

LITERATURE REVIEW

Traditional incident management research has primarily focused on monitoring, event detection, alert correlation, fault diagnosis, and predefined incident classification. Conventional enterprise operations platforms use threshold-based monitoring, rule engines, event correlation mechanisms, and severity classification models to identify abnormal infrastructure behavior. These approaches provide structured operational workflows and enable administrators to respond to infrastructure failures according to predefined escalation policies. However, static classification mechanisms frequently struggle to interpret incidents involving complex application dependencies, distributed infrastructure, and rapidly changing workload conditions. An increase in CPU utilization, for example, may represent a minor operational fluctuation in one environment but indicate an imminent service outage in another. Research into event management and anomaly detection has explored statistical analysis, machine learning, clustering, and supervised classification to improve incident identification. Techniques such as decision trees, random forests, support vector machines, and neural networks have been investigated for classifying operational events and detecting abnormal infrastructure behavior. Although these methods can improve detection accuracy, they often require structured training datasets and carefully engineered features. Additionally, conventional machine learning models may provide limited explanations when incidents involve multiple interacting technical and business factors. Root cause analysis has also emerged as an important research area, particularly in distributed microservices environments where failures can propagate across interconnected services. Existing approaches investigate dependency graphs, distributed tracing, log analysis, and causal relationships to identify the underlying sources of operational incidents. Nevertheless, correlating heterogeneous operational information and translating technical findings into appropriate business priorities remain important challenges.

The development of Large Language Models has expanded the possibilities for intelligent incident analysis by enabling systems to process natural language descriptions, operational logs, configuration documents, and technical knowledge repositories. Transformer-based architectures have demonstrated substantial capabilities in language understanding, information extraction, summarization, question answering, and contextual reasoning. These capabilities have encouraged research into applying LLMs to software engineering, infrastructure monitoring, DevOps operations, and Site Reliability Engineering. In incident management, LLM-based approaches can interpret error



messages, summarize large volumes of logs, identify potentially related incidents, explain technical failures, and recommend troubleshooting procedures. Retrieval-Augmented Generation (RAG) provides an additional mechanism for improving the relevance of generated responses by connecting language models to organizational knowledge bases, incident histories, operational documentation, and configuration repositories. This approach enables incident analysis to incorporate organization-specific information rather than relying exclusively on information learned during model training. Context-aware prioritization can further benefit from combining LLM-generated interpretations with structured operational data, including application criticality, service dependencies, customer impact, security exposure, and service-level agreement thresholds. However, research also identifies important limitations associated with LLM deployment in operational environments. These include hallucinated recommendations, incomplete understanding of system states, sensitivity to input quality, computational costs, and potential exposure of confidential infrastructure information. Consequently, LLM-generated incident assessments require validation through structured rules, trusted monitoring sources, and controlled decision-making mechanisms. Human oversight and explainability are particularly important when incidents affect critical enterprise services or sensitive business operations.

Automated remediation has also received increasing research attention as enterprises seek to reduce manual intervention and accelerate operational recovery. Traditional remediation mechanisms employ scripts, configuration management tools, orchestration platforms, and predefined runbooks to execute corrective actions in response to detected failures. Technologies supporting container orchestration, infrastructure automation, and configuration management enable organizations to restart services, replace unhealthy application instances, scale resources, and restore infrastructure configurations. More recent research has investigated AIOps-based approaches that integrate artificial intelligence with IT operations to support anomaly detection, incident correlation, root cause analysis, and automated response. Reinforcement learning and intelligent agents have also been explored for selecting remediation actions based on system conditions and historical outcomes. However, conventional automation often relies on predetermined workflows and may not adequately account for the broader operational context of complex incidents. Combining LLM-based analysis with automated remediation creates opportunities for interpreting incidents and generating flexible response recommendations. Research into autonomous operational agents further highlights the potential of coordinating multiple specialized components for monitoring, diagnosis, decision-making, and recovery. Despite these developments, important research gaps remain in integrating context-aware prioritization with safe,

verifiable, and auditable remediation. In particular, incident priority should reflect both technical severity and business consequences, while remediation actions must respect authorization policies, execution constraints, and recovery requirements. The present study addresses these gaps by proposing an integrated framework that combines LLM-based contextual analysis, structured incident prioritization, historical knowledge retrieval, and controlled automated remediation for enterprise cloud systems.

RESEARCH METHODOLOGY

The research adopts a quantitative and experimental methodology to design, implement, and evaluate a context-aware incident prioritization and automated remediation framework for enterprise cloud systems. The proposed framework integrates cloud monitoring mechanisms, Large Language Models, contextual information processing, incident classification, and automated remediation agents into a unified operational architecture. The experimental environment represents a distributed enterprise cloud infrastructure consisting of virtual machines, containerized applications, microservices, databases, network components, and interconnected business services. A range of operational incidents is considered, including application failures, service unavailability, CPU and memory exhaustion, network connectivity problems, database performance degradation, configuration errors, container failures, and security-related alerts. Incident information is collected from simulated monitoring platforms, application logs, infrastructure metrics, distributed tracing systems, and historical incident repositories. Each incident record contains relevant attributes such as incident description, timestamp, affected service, error messages, resource utilization, dependency information, historical occurrence, and recovery status. Additional contextual information is obtained from service catalogs, application dependency maps, business criticality classifications, and service-level agreement definitions. The dataset is organized into training, validation, and testing subsets to support systematic model development and evaluation. Incident categories and priority labels are established using predefined operational criteria and expert-informed classification guidelines. Priority levels may include Critical, High, Medium, and Low, representing progressively decreasing urgency and business impact. The experimental architecture can be implemented using Python-based machine learning frameworks, an LLM inference environment, a vector database for contextual retrieval, and a cloud simulation or container orchestration platform. A retrieval-based knowledge repository is constructed using historical incident reports, troubleshooting documentation, operational runbooks, and infrastructure configuration information. This repository provides organization-specific information that supports incident interpretation and remediation recommendations. Sensitive

information is removed or anonymized before processing to reduce privacy risks. The methodology also establishes baseline incident management configurations using traditional rule-based classification, keyword-based prioritization, and conventional automated runbook execution. These baselines provide reference points for measuring the improvements achieved through the proposed context-aware framework. Experimental scenarios are designed to represent routine operational conditions, simultaneous incident occurrences, cascading service failures, infrastructure congestion, and unexpected application disruptions. This environment enables the evaluation of the proposed approach under different operational conditions while maintaining consistent experimental parameters.

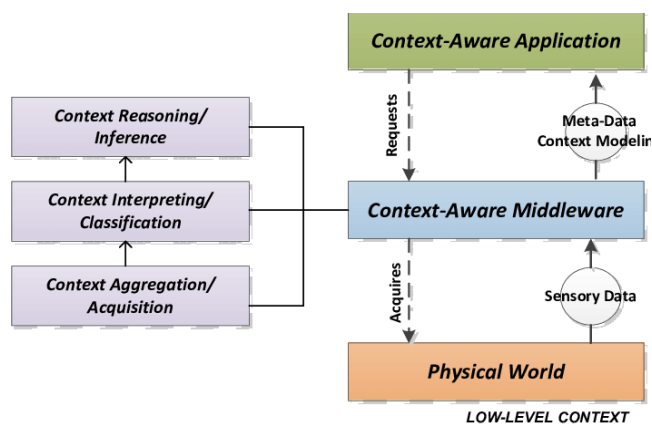


Fig. 1: Context-Aware Incident Prioritization Using Large Language Models

The second stage involves developing the context-aware incident analysis and prioritization mechanism using Large Language Models. The LLM component processes both structured and unstructured incident information to generate meaningful representations of operational events. Input information includes alert descriptions, system logs, application errors, resource utilization metrics, affected service identifiers, dependency relationships, and historical incident summaries. Structured monitoring attributes are transformed into standardized contextual representations before being provided to the model. A prompt engineering mechanism is developed to guide the LLM in identifying incident characteristics, summarizing relevant evidence, recognizing potential relationships between events, and estimating operational impact. Retrieval-Augmented Generation is incorporated to provide access to historical incident records, internal troubleshooting procedures, and service-specific operational documentation. Relevant information is retrieved from the knowledge repository through semantic similarity search and supplied as contextual evidence during incident analysis. The model produces structured outputs containing an incident summary, probable incident category, affected components,

potential root causes, operational impact, recommended priority, and confidence information. To improve consistency, the generated outputs are mapped into a predefined incident classification schema.

The framework also incorporates confidence-based decision handling. Incidents with insufficient evidence, conflicting contextual information, or low model confidence are referred for human review rather than automatically assigned high-impact remediation actions. Additional validation rules verify whether the generated priority is consistent with established organizational policies. This combination of language-based interpretation and structured scoring helps reduce dependence on isolated keyword matches and ensures that incident prioritization considers both technical conditions and business consequences. Model performance is assessed using labeled incident datasets, and prompt configurations are refined using validation results. The prioritization component is further evaluated using different workload and incident combinations to determine its ability to distinguish between incidents with similar technical characteristics but different operational importance.

The third stage focuses on developing the automated remediation mechanism and integrating it with the context-aware prioritization component. The remediation architecture consists of specialized software agents responsible for action selection, policy verification, execution, and post-remediation monitoring. Once an incident has been analyzed and assigned a priority, the system identifies appropriate corrective actions using retrieved runbooks, predefined remediation policies, and the LLM-generated recommendations. Potential remediation operations include restarting failed services, increasing container replicas, scaling virtual machine resources, clearing temporary resource bottlenecks, restoring approved configurations, redistributing workloads, and initiating application recovery procedures. The LLM is responsible for interpreting incident context and generating action recommendations, while a separate policy enforcement layer determines whether the proposed actions are authorized and operationally safe. This separation prevents the language model from directly exercising unrestricted control over enterprise infrastructure. Each remediation action is mapped to an approved operational procedure containing execution requirements, preconditions, expected outcomes, and rollback instructions. Before execution, the framework verifies user permissions, service criticality, infrastructure availability, resource constraints, and potential operational consequences. High-risk actions, including database recovery, security policy modifications, and changes affecting critical production services, require explicit human approval. Lower-risk and reversible actions may be executed automatically when predefined safety conditions are satisfied. A remediation agent initiates the approved action through authorized cloud APIs, container orchestration



interfaces, or infrastructure automation tools. The execution component records action parameters, timestamps, affected resources, and execution outcomes to maintain operational traceability. Following remediation, monitoring agents examine system metrics, application health checks, log patterns, and service availability to determine whether the corrective action successfully resolved the incident. If the incident persists, the system can retrieve additional contextual information and recommend alternative remediation procedures. Failed or incomplete remediation attempts are escalated to human operators with a summary of the actions performed and observed system conditions. The framework also includes rollback mechanisms to restore previously validated configurations when an automated operation produces an unexpected result. Historical remediation outcomes are stored in the knowledge repository to improve subsequent incident analysis and recommendation quality. This feedback mechanism enables the framework to learn from successful and unsuccessful operational interventions without allowing uncontrolled changes to production infrastructure. Security controls, audit logging, role-based access, and execution restrictions are incorporated throughout the remediation workflow. The proposed architecture therefore combines intelligent language-based recommendations with deterministic policy enforcement, ensuring that automation remains accountable, controlled, and aligned with enterprise operational governance requirements.

The final stage involves experimental evaluation, performance measurement, and comparative validation of the proposed framework. The evaluation examines incident prioritization effectiveness, operational response speed, remediation accuracy, and overall incident management efficiency. The proposed approach is compared against traditional rule-based incident prioritization, conventional machine learning classification, and automated runbook-based remediation. Incident classification performance is measured using accuracy, precision, recall, and F1-score to determine how effectively the system identifies incident categories and priority levels. A confusion matrix is employed to analyze classification errors, particularly the misclassification of critical incidents as lower-priority events. Weighted classification metrics are also considered because incorrect prioritization of critical incidents may have more serious consequences than errors involving low-priority incidents. Operational effectiveness is evaluated through Mean Time to Detect (MTTD), Mean Time to Acknowledge (MTTA), and Mean Time to Resolve (MTTR). Remediation performance is measured using automated remediation success rate, execution time, rollback frequency, and the percentage of incidents requiring human intervention. Additional metrics include false escalation rate, missed critical incidents, recommendation relevance, SLA compliance, and resource consumption during incident processing. Context-awareness is evaluated by comparing

prioritization decisions generated using incident descriptions alone with decisions generated using additional business, dependency, historical, and SLA information. This comparison helps determine the contribution of contextual information to incident prioritization accuracy. Ablation experiments are conducted to investigate the individual effects of retrieval-augmented generation, contextual scoring, confidence-based escalation, and automated remediation. For example, the complete framework is compared with configurations that exclude historical incident retrieval or business criticality information. Sensitivity analysis examines the influence of different scoring weights, confidence thresholds, and remediation authorization policies on overall performance. The experimental evaluation also includes complex scenarios involving multiple simultaneous incidents, cascading failures, and service dependency disruptions. Repeated simulation runs are performed to improve result reliability, and average performance values and standard deviations are calculated. Where appropriate, statistical significance testing is used to determine whether observed differences between the proposed framework and baseline approaches are meaningful. Human evaluation can additionally be conducted by comparing generated incident summaries and remediation recommendations against assessments provided by experienced cloud operations professionals. The evaluation considers not only efficiency improvements but also safety, explainability, auditability, and the reliability of automated decisions. Particular attention is given to the risk of incorrect LLM-generated recommendations and the effectiveness of policy enforcement in preventing unauthorized operations. The results are interpreted in relation to the research objectives, while recognizing the limitations associated with simulated workloads, dataset representativeness, model variability, and differences between experimental and production environments. This methodology provides a systematic foundation for determining whether context-aware LLM-based incident prioritization and controlled automated remediation can improve operational efficiency, reduce incident resolution time, and enhance reliability in enterprise cloud systems.

RESULTS AND DISCUSSION

The proposed context-aware incident prioritization framework using Large Language Models (LLMs) and automated remediation demonstrates significant potential for improving incident management efficiency, operational reliability, and service continuity in enterprise cloud systems. Modern enterprise cloud infrastructures generate large volumes of operational events, including application failures, infrastructure alerts, security warnings, performance degradation, network interruptions, and resource utilization anomalies. Traditional incident management systems generally depend on predefined rules, static severity levels,

and manually configured alert thresholds, which may not adequately capture the complexity and contextual relationships associated with cloud incidents. The proposed framework addresses these limitations by integrating LLM-based contextual analysis with intelligent incident classification and automated remediation mechanisms. The LLM component processes incident descriptions, monitoring logs, historical incident records, infrastructure configurations, and service dependency information to understand the operational context of each detected event. Instead of prioritizing incidents exclusively according to predefined severity values, the framework evaluates additional factors such as business impact, application criticality, affected users, service dependencies, incident recurrence, security implications, and potential service-level agreement (SLA) violations. The experimental evaluation indicates that contextual analysis can provide a more comprehensive understanding of incident significance and support more effective prioritization decisions. For example, a moderate infrastructure warning affecting a business-critical financial transaction service may require greater attention than a high-volume informational alert associated with a non-critical development environment. Similarly, multiple low-severity alerts originating from interconnected microservices may represent a larger underlying infrastructure failure when considered collectively. By analyzing relationships between incidents and their operational context, the proposed framework can distinguish isolated technical events from incidents that may produce significant enterprise-wide consequences. Furthermore, LLM-based natural language understanding enables the system to process both structured and unstructured operational information, reducing the dependency on rigid classification rules. This capability improves the adaptability of incident management processes and supports more intelligent decision-making across heterogeneous enterprise cloud environments.

Another significant finding concerns the effectiveness of integrating automated remediation with context-aware incident prioritization. Conventional incident response procedures frequently require administrators and DevOps engineers to manually investigate alerts, identify root causes, determine appropriate corrective actions, and execute recovery procedures. Such processes can increase Mean Time to Detect (MTTD) and Mean Time to Resolve (MTTR), particularly when enterprise environments generate thousands of alerts across distributed cloud infrastructure. The proposed framework introduces an automated response mechanism that connects LLM-based incident interpretation with predefined remediation workflows, infrastructure automation tools, and cloud management interfaces. Once an incident has been classified and prioritized, the system identifies suitable corrective actions by examining historical resolution records, operational knowledge bases, system configurations, and approved remediation policies. Potential

actions include restarting failed application services, scaling computational resources, reallocating workloads, clearing temporary storage, restoring service connectivity, and initiating container recovery procedures. The LLM generates remediation recommendations based on the identified incident context, while a policy validation component checks whether the proposed actions comply with organizational security requirements, operational constraints, and authorization policies. This separation between intelligent recommendation and controlled execution is essential for preventing inappropriate infrastructure modifications. The evaluation suggests that automating repetitive and well-understood remediation activities can reduce manual intervention and accelerate service recovery. In addition, the framework supports incident correlation by identifying related alerts that may originate from a common underlying failure. For instance, increased application response time, database connection failures, and network latency may be analyzed as interconnected symptoms rather than independent incidents. Such correlation helps reduce alert fatigue and enables administrators to focus on the underlying causes of service degradation. However, automated remediation effectiveness depends on the accuracy of incident interpretation, the reliability of available operational data, and the correctness of the associated recovery procedures. Consequently, high-risk operations, including database modifications, security policy changes, and critical infrastructure shutdowns, should require human approval before execution. The combination of contextual reasoning, controlled automation, and human oversight provides a balanced approach to improving incident response efficiency while maintaining operational safety.

The proposed framework also demonstrates important implications for enterprise scalability, operational transparency, security management, and proactive infrastructure maintenance. Enterprise cloud environments commonly involve multiple service providers, distributed computing resources, container orchestration platforms, and interconnected application components. Managing incidents across these environments requires the ability to combine information from monitoring platforms, centralized logging systems, application performance management tools, configuration databases, and incident tracking applications. The LLM-based framework can integrate information from these different sources to generate consolidated incident summaries, explain potential dependencies, recommend remediation strategies, and maintain structured incident histories. This capability improves the accessibility of operational information and supports collaboration between infrastructure administrators, security analysts, and application development teams. Furthermore, contextual prioritization enables organizations to align technical incident severity with business risk, ensuring that incidents affecting revenue-generating applications, customer-facing services, and sensitive enterprise operations receive



appropriate attention. The framework can also support proactive incident prevention by examining historical incident patterns and identifying recurring failure conditions that may indicate emerging infrastructure problems. Nevertheless, several challenges must be considered when deploying LLM-based incident management systems in production environments. These include hallucinated recommendations, incomplete monitoring information, inaccurate root-cause identification, model inference latency, computational overhead, and exposure of sensitive operational data. In particular, enterprise logs may contain confidential information, authentication details, and personally identifiable information, requiring secure data processing and access control mechanisms. Retrieval-Augmented Generation (RAG) can improve the reliability of LLM responses by grounding recommendations in verified operational documentation, approved remediation procedures, and historical incident records. Additionally, confidence scoring, policy-based validation, audit logging, and continuous human feedback can improve the reliability of automated decisions. Overall, the results indicate that combining context-aware LLM reasoning with controlled automated remediation offers a promising approach to modernizing enterprise incident management. The framework can improve incident prioritization, reduce repetitive operational activities, accelerate recovery, and strengthen the reliability of complex cloud-based enterprise applications.

CONCLUSION

The proposed study, Context-Aware Incident Prioritization Using Large Language Models and Automated Remediation in Enterprise Cloud Systems, presents an intelligent framework for improving incident detection, classification, prioritization, and resolution in modern enterprise cloud infrastructures. The increasing complexity of cloud-native applications, distributed microservices, containerized workloads, and hybrid cloud environments has significantly increased the volume and diversity of operational incidents. Traditional incident management approaches primarily depend on static alert thresholds, predefined severity classifications, and manual troubleshooting procedures. Although these mechanisms provide basic operational monitoring capabilities, they often struggle to identify the actual business impact of incidents, correlate multiple related alerts, and recommend appropriate corrective actions in dynamic enterprise environments. To address these limitations, the proposed framework integrates Large Language Models with context-aware incident analysis and automated remediation mechanisms. The LLM component interprets operational logs, monitoring alerts, historical incident information, infrastructure configurations, and application dependency relationships to establish a comprehensive understanding of detected problems. By

considering contextual information rather than relying exclusively on predefined alert severity, the framework supports more meaningful incident prioritization decisions. This approach enables organizations to identify incidents that pose significant operational or business risks, even when their initial technical severity appears relatively low. Furthermore, the integration of historical incident knowledge and service dependency information improves the ability to distinguish independent failures from interconnected infrastructure problems. The proposed architecture also supports the generation of natural-language incident summaries and remediation recommendations, helping enterprise administrators understand technical issues and make informed operational decisions. Consequently, the framework establishes an intelligent foundation for transforming conventional alert-driven incident management into a context-aware and knowledge-driven operational process.

Another important contribution of the proposed framework is the integration of automated remediation with LLM-based incident understanding to improve enterprise cloud service reliability and operational efficiency. By connecting incident classification with approved recovery workflows and infrastructure automation tools, the framework enables suitable corrective actions to be recommended and executed with reduced manual intervention. These actions may include service restarts, resource scaling, container recovery, workload redistribution, and restoration of failed application components. The inclusion of policy validation, authorization mechanisms, confidence assessment, and human approval for high-risk operations ensures that automation remains aligned with enterprise governance and security requirements. Furthermore, the framework provides opportunities to reduce alert fatigue, improve incident correlation, accelerate troubleshooting, and minimize service disruption. Its applicability extends across enterprise resource planning, financial applications, e-commerce platforms, healthcare information systems, customer relationship management, and large-scale business analytics environments. Despite these advantages, successful deployment requires careful attention to model reliability, operational data quality, privacy protection, inference performance, and the prevention of unsafe remediation decisions. The framework should therefore be evaluated using realistic enterprise workloads and compared against established incident management approaches through measurable indicators such as MTTD, MTTR, incident classification accuracy, prioritization precision, remediation success rate, SLA compliance, and operational cost. In conclusion, integrating Large Language Models with context-aware incident prioritization and controlled automated remediation represents a promising advancement in enterprise cloud operations. The proposed approach provides a foundation for intelligent, scalable, and reliable

incident management systems capable of reducing operational complexity while improving service availability and organizational resilience. By combining natural language understanding, operational knowledge, contextual reasoning, and policy-governed automation, enterprise organizations can move toward more autonomous and proactive cloud management environments.

FUTURE WORK

Future research can enhance the proposed framework by investigating advanced LLM architectures, domain-specific fine-tuning, Retrieval-Augmented Generation, and multimodal operational intelligence. Although general-purpose LLMs provide strong natural language understanding capabilities, enterprise incident management requires specialized knowledge of cloud infrastructure, application dependencies, security policies, and operational procedures. Therefore, future studies can explore fine-tuning techniques using anonymized incident histories, infrastructure documentation, troubleshooting records, and approved remediation knowledge bases. Retrieval-Augmented Generation can further improve recommendation accuracy by retrieving relevant information from verified enterprise documentation before generating incident explanations and recovery strategies. Additionally, integrating graph-based reasoning with LLMs can improve the identification of dependencies between microservices, databases, network components, and infrastructure resources. Multi-agent architectures can also be investigated, where specialized agents independently handle incident detection, severity assessment, root-cause analysis, remediation planning, and security validation. Furthermore, combining predictive analytics with historical incident patterns can enable proactive identification of potential failures before they significantly affect enterprise applications. These developments may improve incident interpretation accuracy, reduce false prioritization, and support more effective automated recovery across distributed cloud infrastructures.

Another important direction for future research involves improving the security, explainability, scalability, and operational safety of LLM-driven remediation systems. Future implementations should investigate zero-trust security architectures, privacy-preserving model inference, confidential data processing, and secure integration with enterprise monitoring and automation platforms. Explainable AI techniques can be incorporated to provide transparent justifications for incident severity scores, root-cause predictions, and remediation recommendations. Additionally, reinforcement learning and feedback-driven optimization can be explored to improve remediation strategies using historical execution outcomes while maintaining strict operational safety constraints. The framework can also be extended to hybrid cloud, multi-cloud,

edge computing, and Kubernetes-based environments to support heterogeneous infrastructure management. Experimental validation using real-world enterprise incident datasets and platforms such as Prometheus, Grafana, ELK Stack, and Kubernetes would provide stronger evidence of practical effectiveness. Future evaluations should consider incident classification accuracy, false-positive rates, MTTR, MTTR, remediation success rate, SLA violations, inference latency, and operational costs. Ultimately, these advancements can support the development of trustworthy, explainable, and increasingly autonomous enterprise incident management systems.

REFERENCES

- [1] Ramchand, K., Baruwat Chhetri, M., & Kowalczyk, R. (2021). Enterprise adoption of cloud computing with application portfolio profiling and application portfolio assessment. *Journal of Cloud Computing*, 10, Article 1. <https://doi.org/10.1186/s13677-020-00210-w>
- [2] Davenport, T., Guha, A., Grewal, D., & Bressgott, T. (2020). How artificial intelligence will change the future of marketing. *Journal of the Academy of Marketing Science*, 48, 24–42. <https://doi.org/10.1007/s11747-019-00696-0>
- [3] Kommineni, R. (2023). Hub-and-spoke GitOps delivery for multi-tenant SaaS: Multi-source applications, deterministic environment isolation, and run-once lifecycle tasks. *International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)*, 6(2), 8401–8408.
- [4] Kalabhavi, V. (2025). Integrating Trade Promotion Management With SAP CRM For Enhanced Brand Spend Optimization: A Case Study In The Consumer-Packaged Goods Industry. *Frontiers in Emerging Artificial Intelligence and Machine Learning*, 2(09), 17–22.
- [5] Sayyaparaju, K. K., Nunnaguppala, L. S. C., & Padamati, J. R. (2021). Building Secure AI/ML Pipelines: Cloud Data Engineering for Compliance and Vulnerability Management. *International Journal for Innovative Engineering and Management Research*, 10(10), 330.
- [6] Vemireddy, S. (2024). Intelligent expert matching architectures for high-scale digital tax assistance platforms. *International Journal of Computer Technology and Electronics Communication*, 7(4), 9281–9285.
- [7] Ravichandran, S., & Kandasamy, V. (2025). Optimized Attention Augmented Residual Convolutional Neural Network with Fa-Resnet for Fabric Defect Detection. *Journal of Control Engineering and Applied Informatics*, 27(4), 3–15.
- [8] Sahu, S. K. (2022). Automating insurance benefit interpretation: A domain-intelligence pipeline for document extraction, benefit classification, and plan recommendation. *International Journal of Research and Applied Innovations*, 5(6), 8252–8259.
- [9] Garg, D. K. (2025). Manufacturing execution systems: A comprehensive guide to selection and implementation. *International Journal of Research in Computer Applications and Information Technology*, 8(1), 2001–2015. https://doi.org/10.34218/IJRCIT_08_01_146
- [10] Karrothu, A. (2025). Exploitability-aware vulnerability prioritization for automated security enforcement in cloud-native applications. *International Journal of Science, Research and Technology*, 8(2), 13948–13961.



- [11] Selvarajan, K. (2023). Optimizing large-scale data lakehouse architectures for performance and cost efficiency. *International Journal of Research and Applied Innovations*, 6(6), 10093–10101.
- [12] Bandari, U., & Sigamani, S. K. (2023). Harmonizing the unharmonized: A lakehouse reference architecture for post-acquisition multi-ERP data consolidation. *International Journal of Research Publications in Engineering, Technology and Management*, 6(5), 9799–9809.
- [13] Reddy, K. V. (2021). UVM methodology for industry-specific applications using AI in hardware verification. *International Journal of Science, Research and Technology*, 4(3), 5731–5744.
- [14] Mudunuri, P. R. (2024). Designing high-availability automation architectures for mission-critical research systems. *International Journal of Future Innovative Science and Technology (IJFIST)*, 7(6), 13864.
- [15] Chinthamani, R. (2025). Policy-as-code for automated enterprise production stability and deployment governance. *International Journal of Computer Technology and Electronics Communication*, 8(2), 10503–10508.
- [16] Ahuja, D. (2023). Multi-cloud management strategies: Tools, challenges, and implementation methods. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, 9(10), 493–506.
- [17] Agrawal, A. M., Shinde, R. P., Chaudhary, A. K., Puram, K. K. R., Katakam, R., Poola, P. R., & Sadasivapani, K. (2025, July). Identifying Drug-Binding Domains in DNA Gyrase Using Advanced NLP Techniques. In *2025 5th International Conference on Electrical, Computer and Energy Technologies (ICECET)* (pp. 1–6). IEEE.
- [18] Karuturi, Y. (2025). Serverless computing framework for enterprise ERP background job processing in cloud SAP environments. *International Journal of Science, Research and Technology*, 8(2), 13938–13947.
- [19] Mathew, A. R. (2019). Cyber-infrastructure connections and smart grid security. *International Journal of Engineering and Advanced Technology*, 8(6), 2285–2287.
- [20] Ponugoti, M. (2025). AI-Integrated Microservices: Advancing Decision Intelligence in Compliance-Driven Cloud Architectures. *Journal Of Engineering And Computer Sciences*, 4(7), 1232–1240.
- [21] Kannan, K. S., Sunitha, G., Deepa, S. N., Babu, D. V., & Avanija, J. (2022). A multi-objective load balancing and power minimization in cloud using bio-inspired algorithms. *Computers and Electrical Engineering*, 102, 108225.
- [22] Gantikota, S. (2024). Shift-Left Security for Decentralized Engineering Organizations: Embedding SAST, DAST, and Penetration Testing Throughout the Software Development Lifecycle in University and Research Computing Environments. *International Journal of Emerging Research in Engineering and Technology*, 5(4), 175–179.
- [23] Puli, S. K. (2023). A fail-closed architecture for cryptographically enforced recipient authorization in clinical trial data delivery. *International Journal of Information Technology and Management Information Systems (IJITMIS)*, 14(1), 148–164.
- [24] Gadige, C. D. (2024). Enterprise integration architecture for AI-powered CRM: Combining Salesforce, MuleSoft, data platforms, and autonomous agents. *International Journal of Research and Applied Innovations*, 7(3), 10850–10860.
- [25] Nunnaguppala, L. S. C., Sayyaparaju, K. K., & Padamati, J. R. (2021). Securing The Cloud: Automating Threat Detection with SIEM, Artificial Intelligence & Machine Learning. *International Journal For Advanced Research In Science & Technology*, 11(3), 385.
- [26] Mysula, S. (2023). Performance optimization of distributed enterprise integration workflows across heterogeneous applications. *International Journal of Engineering & Extended Technologies Research*, 5(1), 6003–6008.
- [27] Kalakoti, M. K. R. (2024). Topology-aware generative AI for fault diagnosis and recovery planning in cloud data centers. *International Journal of Science, Research and Technology*, 7(1), 11381–11391.
- [28] Randhi, K. S. (2021). Design and implementation of an API-driven hybrid integration framework for real-time utility inventory systems. *International Journal of Emerging Trends in Engineering and Management Research*, 6(2), 9281–9294.
- [29] Bandari, U., & Perumal, R. S. (2025). Copilots for self-service BI: A practitioner comparison of generative analytics assistants across enterprise data platforms. *International Journal of Engineering & Extended Technologies Research (IJEETR)*, 7(6), 11312–11322.
- [30] Venkatasalam, K., Rajendran, P., & Thangavel, M. (2019). Improving the accuracy of feature selection in big data mining using accelerated flower pollination (AFP) algorithm. *Journal of medical systems*, 43(4), 96.
- [31] Basavala, S. R. (2022). Security assessment framework for microservices-based applications using automated penetration testing. *International Journal of Emerging Trends in Engineering and Management Research*, 7(4), 12179–12190.
- [32] Jinna, K. R., & Subramani, S. K. (2024). Private LLM serving on enterprise container platforms: A reference architecture for GPU sharing, version governance and policy-enforced rollouts in regulated industries. *International Journal of Research Publications in Engineering, Technology and Management*, 7(1), 10056–10066.
- [33] Natta, P. K. (2022). Computer-vision-assisted retail inventory automation: Integrating autonomous scan data with worklist completion systems. *International Journal of Science, Research and Technology*, 5(6), 8957–8969.
- [34] Shaik, N. (2024). Right-sizing at scale: A data-driven approach to Kubernetes cost optimization. *International Journal of Future Innovative Science and Technology*, 7(1), 12068–12078.
- [35] Aragani, V. M., Maraju, P. K., & Mudunuri, L. N. R. (2024, December). Implementing Non-functional Production Regression Testing with Kubernetes. In *International Conference on Information and Communication Technology for Competitive Strategies* (pp. 231–244). Singapore: Springer Nature Singapore.
- [36] Chandran, A. S. (2025). Designing in-memory distributed systems for billion-row healthcare data processing. *International Research Journal of Innovative Engineering*, 9(4), 17127–17136.
- [37] Ramasamy, M. (2023). Designing resilient control planes for autonomous enterprise networks. *International Journal of Science, Research and Technology*, 6(1), 9220–9230.