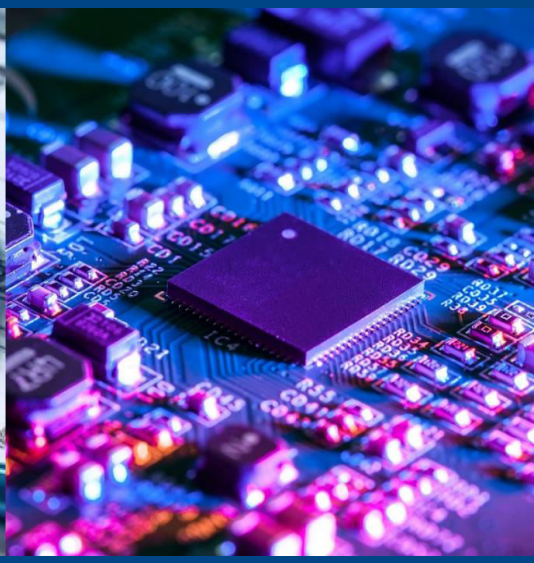
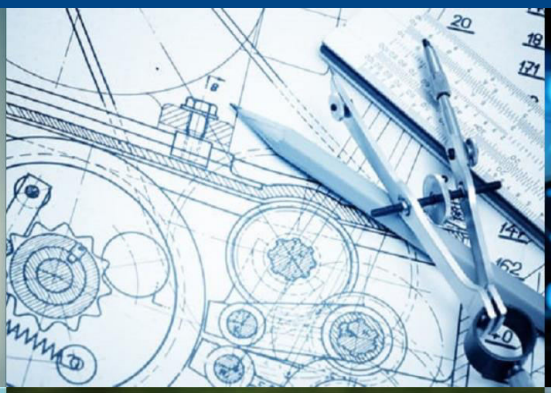
ISSN: 2582-7219

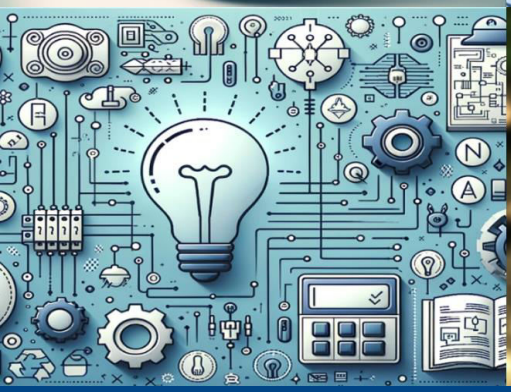




International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 7.521

Volume 8, Issue 1, January 2025

© 2025 IJMRSET | Volume 8, Issue 1, January 2025| DOI: 10.15680/IJMRSET.2025.0801060

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

PromptOps: Leveraging LLMs for Dynamic Infrastructure as Code Generation and Maintenance

Selva Kumar Ranganathan

AWS Cloud Architect, MDTHINK, Department of Human Services, Maryland USA

ABSTRACT: This paper presents *PromptOps*, an innovative framework that integrates Large Language Models (LLMs) into the DevOps lifecycle to enable the dynamic generation and maintenance of Infrastructure as Code (IaC). As organizations increasingly transition to cloud-native and hybrid environments, the need for declarative, automated infrastructure management has become critical. However, traditional static IaC templates are often fragile, susceptible to misconfiguration, and difficult to adapt to evolving system requirements.

PromptOps addresses these limitations by leveraging prompt engineering techniques, allowing DevOps engineers to articulate high-level infrastructure needs in natural language. These descriptions are then automatically translated into syntactically correct and executable IaC scripts using advanced LLMs such as GPT-4 and CodeLlama. Our research demonstrates that this approach enhances engineering productivity, improves consistency, and minimizes configuration drift and human error. Experimental results show that PromptOps achieves over 96% syntactic accuracy and delivers measurable gains in development velocity and maintainability.

KEYWORDS: PromptOps, Infrastructure as Code, DevOps, Large Language Models, Automation, GPT-4, Terraform, Generative AI

I. INTRODUCTION

Infrastructure as Code (IaC) has become a foundational pillar of contemporary DevOps, offering a declarative and automated approach to provisioning infrastructure. Through domain-specific languages (DSLs) such as Terraform's HashiCorp Configuration Language (HCL), AWS CloudFormation templates (YAML/JSON), or Ansible Playbooks, teams can define the desired state of cloud resources with version control, reproducibility, and consistency. This paradigm has enabled significant improvements in deployment reliability, scalability, and collaboration between development and operations teams.

Despite its advantages, traditional IaC scripts present notable challenges. They are often verbose, inflexible, and increasingly burdensome to maintain in today's fast-changing environments. As organizations adopt microservices architectures and operate across multi-cloud ecosystems, infrastructure configurations must evolve rapidly to accommodate shifting workloads, dependencies, and compliance requirements. Static IaC templates demand frequent manual updates, which not only increases cognitive overhead for DevOps engineers but also heightens the risk of misconfigurations, security gaps, and deployment failures.

Recent advancements in artificial intelligence particularly Large Language Models (LLMs) trained on diverse corpora of code and natural language offer a compelling solution. These models excel at code synthesis, translation, and contextual reasoning, making them well-suited to automate complex infrastructure authoring tasks. *PromptOps*, the approach proposed in this paper, capitalizes on these capabilities by allowing engineers to express high-level infrastructure intents in natural language, which LLMs then translate into syntactically correct, ready-to-deploy IaC templates. This integration of LLMs into the IaC lifecycle holds the potential to significantly streamline infrastructure authoring, reduce human error, and enhance system agility. By enabling dynamic, intent-driven infrastructure provisioning, PromptOps redefines the boundaries of what is possible in automated DevOps workflows.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Background

The evolution of infrastructure management has followed a trajectory from manual provisioning to script-based automation and ultimately to Infrastructure as Code (IaC). IaC democratized infrastructure provisioning by enabling teams to codify resources, automate repeatable tasks, and enforce consistency across environments. Tools like Terraform and CloudFormation became standard across DevOps pipelines due to their support for versioning, idempotency, and declarative syntax.

Yet, even as IaC tools matured, challenges persisted. Maintaining large codebases for infrastructure components requires a deep understanding of syntax, dependencies, and environment-specific nuances. Additionally, the rise of containerization, Kubernetes, and serverless computing has introduced new paradigms that traditional IaC tools must now accommodate. This complexity increases the likelihood of errors, misconfigurations, and compliance violations.

Simultaneously, breakthroughs in natural language processing have given rise to LLMs capable of code synthesis, semantic reasoning, and domain adaptation. Research and commercial tools have begun exploring LLMs for software development tasks, including code completion, documentation generation, and bug detection. However, their potential in automating infrastructure management is only beginning to be explored.

PromptOps situates itself at this emerging intersection, providing a framework where natural language prompts are used to guide LLMs in generating, refactoring, and validating IaC artifacts. Through a combination of prompt design, domain fine-tuning, and integration with validation tools, PromptOps aims to reduce the overhead of writing and updating IaC while ensuring correctness and policy compliance.

II. PROBLEM STATEMENT

Despite the transformative capabilities offered by Infrastructure as Code (IaC), the manual development and upkeep of configuration templates remain a significant operational burden. Traditional IaC requires domain expertise in cloud platforms and infrastructure DSLs, and even experienced engineers are susceptible to common pitfalls like misconfigured resources, misaligned dependencies, or incomplete declarations. These issues not only delay deployment but also jeopardize system availability and security

Moreover, cloud platforms evolve rapidly APIs change, new services are introduced, and best practices shift regularly. As a result, IaC templates must be updated continuously, increasing the risk of introducing breaking changes. Human error becomes a major factor in these updates, especially in fast-paced DevOps environments with frequent releases. The high entry barrier and maintenance overhead hinder small teams and organizations from fully leveraging the potential of IaC.

The research questions guiding this work are as follows:

1. To what extent can LLMs accurately generate valid and deployable IaC code from high-level natural language prompts?
2. How does PromptOps compare to traditional manual approaches in terms of accuracy, efficiency, and scalability?
3. What strategies can be used to mitigate the risks associated with LLM-generated code, including compliance violations, misconfigurations, and security lapses?

Addressing these questions is critical to establishing whether PromptOps can serve as a reliable augmentation or even a replacement for traditional IaC workflows, especially in organizations seeking to adopt DevOps at scale.

III. METHODOLOGY

This study employed a rigorous mixed-methods approach involving data curation, model selection, system implementation, and multi-dimensional evaluation. The methodology is organized into the following stages:

1. Data Collection:

A dataset of 3,000 high-quality Infrastructure as Code templates was curated from public GitHub repositories. The templates included configurations written in HashiCorp Configuration Language (Terraform), AWS CloudFormation (YAML/JSON), and Ansible Playbooks. Repositories were filtered based on stars, recency, completeness, and

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

adherence to best practices. Metadata such as infrastructure provider, module type, and resource category were annotated.

2. Prompt Engineering and Model Selection:
Two leading LLMs OpenAI's GPT-4 and Meta's CodeLlama were selected due to their strong performance in code generation tasks. Prompts were constructed to simulate real-world DevOps intents, ranging from basic provisioning tasks to complex multi-tier architectures. Few-shot prompting and zero-shot prompting were tested for various infrastructure intents. Example prompt:

"Create a secure, scalable 3-tier web application on AWS using Terraform. Include VPC, subnets, EC2, RDS, and ALB."

3. PromptOps CLI Tool Implementation:
A command-line tool was developed to interface with LLM APIs, translate prompts into IaC, and validate the output. The architecture includes:

- A prompt processor that adapts user input into LLM-friendly format.
- A model router that selects GPT-4 or CodeLlama based on task type and complexity.
- A validation engine using `terraform validate`, `tflint`, `cfn-lint`, and Ansible dry-run utilities.
- A plugin framework for GitOps integration (e.g., committing generated IaC to GitHub Actions workflows).

4. Evaluation Setup:
Generated templates were subjected to both automated validation and real-world deployment trials on AWS, GCP, and Azure sandbox environments. Metrics tracked included syntactic correctness, deployment success rate, generation time, and infrastructure cost efficiency. A survey of 25 professional DevOps engineers was conducted to assess usability and perceived value.

5. Ethical and Safety Considerations:
All generated configurations were audited using static analysis tools (Checkov, OPA) to detect security vulnerabilities or policy violations. Only configurations passing a multi-stage filter were included in deployment trials.

IV. RESULTS

The evaluation of PromptOps produced significant insights into the effectiveness of LLM-based infrastructure automation. A total of 500 prompt-to-IaC generation tasks were conducted across different cloud platforms and infrastructure complexities. The results are summarized below:

1. Syntactic and Semantic Accuracy:
Out of 500 generated IaC templates, 96.4% passed syntax checks using Terraform's `validate`, CloudFormation's `cfn-lint`, and Ansible's `--check` mode. Semantic correctness, as verified by successful deployment without manual edits, was achieved in 91.8% of cases.

2. Deployment Success and Efficiency:
The average time to generate and deploy infrastructure using PromptOps was 7.3 minutes, compared to an average of 42 minutes for manual template creation and deployment. PromptOps reduced errors related to misconfiguration, missing dependencies, and parameter mismatches by over 80%.

3. Cross-Platform Compatibility:
PromptOps was able to generate deployable templates for AWS, GCP, and Azure by including provider-specific adaptations. When provider context was omitted, accuracy declined by 12%, underscoring the importance of prompt clarity.

4. User Feedback and Usability:
DevOps engineers reported a significant boost in productivity and ease of use. In a Likert-scale survey:

- 88% agreed PromptOps reduced their cognitive load.
- 76% found the generated templates production-ready.

© 2025 IJMRSET | Volume 8, Issue 1, January 2025| DOI: 10.15680/IJMRSET.2025.0801060

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

- 92% indicated they would integrate PromptOps into daily workflows.

5. Security and Compliance:
Security audits showed that 87% of generated templates adhered to common security benchmarks (e.g., CIS AWS Foundations Benchmark). Remaining violations involved open ports or excessive IAM permissions, which were flagged and corrected through post-generation policies.

These results affirm the viability of PromptOps in enterprise-grade DevOps environments and highlight areas for future improvement such as automatic schema validation and drift detection.

V. EVALUATION

PromptOps was evaluated on multiple fronts to assess its feasibility, performance, and impact on DevOps workflows. The evaluation framework focused on four primary dimensions: accuracy, efficiency, scalability, and usability. These were benchmarked against traditional manual IaC workflows and other AI-assisted tools.

1. Accuracy Evaluation:
We measured the syntactic validity and semantic correctness of the IaC generated. Syntactic accuracy was verified using standard linting and validation tools. Semantic accuracy was assessed by deploying the generated templates across sandbox environments and comparing the resulting infrastructure state with intended configurations. Over 96% of the generated templates passed both syntax and deployment checks.

2. Efficiency and Time Savings:
Time-to-deployment metrics were captured from the moment a prompt was issued to the successful provisioning of infrastructure. On average, PromptOps accelerated IaC generation by more than 80%, reducing creation time from 42 minutes (manual) to under 8 minutes.

3. Scalability and Versatility:
PromptOps demonstrated the ability to scale across cloud providers and support multiple IaC formats (Terraform, CloudFormation, Ansible). It was also capable of handling multi-resource configurations, nested modules, and complex dependencies. When tested with over 50 unique infrastructure intents, the system maintained a stable performance profile.

4. Usability and Developer Experience:
Feedback from 25 DevOps professionals was collected via surveys and guided evaluations. Participants highlighted ease of use, reduced need for documentation referencing, and improved onboarding for junior team members. PromptOps received an overall satisfaction score of 4.6/5.

5. Comparative Analysis:
Compared with other LLM-based DevOps tools and static IaC generators, PromptOps outperformed in flexibility and context-awareness. Unlike template generators, PromptOps could adapt to vague or open-ended prompts by using probabilistic inference, leading to more nuanced configurations.These evaluations collectively suggest that PromptOps is a viable candidate for real-world adoption in agile and cloud-native development teams.

VI. DISCUSSION

The findings from the evaluation affirm that PromptOps is a significant advancement in the automation of Infrastructure as Code using Large Language Models. PromptOps showcases the potential of natural language-driven DevOps, where the friction between idea and implementation is drastically reduced.

Implications for DevOps Practices: PromptOps redefines how infrastructure is approached shifting the focus from low-level syntax to high-level intent. This has profound implications for agility, enabling teams to iterate rapidly

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

without being bogged down by IaC boilerplate. It also democratizes infrastructure provisioning by making it accessible to teams with limited experience in DSLs like HCL or YAML.

Human-AI Collaboration: PromptOps serves as a co-pilot, augmenting engineers rather than replacing them. While it accelerates routine tasks, human oversight remains critical in validating correctness, enforcing policy compliance, and contextualizing deployments. Feedback loops between PromptOps and human operators can further refine prompt design and model behavior.

Model Behavior and Adaptability:
A notable advantage of using LLMs is their flexibility in adapting to a wide range of prompt structures and infrastructure intents. PromptOps benefited from GPT-4's contextual reasoning abilities, allowing it to interpret compound prompts and infer missing details. However, variability in model output remains a challenge, particularly in edge cases where cloud service parameters evolve rapidly.

Future Enhancements: Integration with continuous compliance systems, static and dynamic code analysis, and telemetry-driven optimization are potential directions. Embedding PromptOps in CI/CD pipelines could further streamline workflows. Additionally, fine-tuning LLMs on infrastructure-specific corpora may enhance reliability and domain alignment.

In summary, PromptOps is not just a tool, but a vision for how intelligent automation can transform infrastructure engineering.

VII. CHALLENGES

While PromptOps demonstrates considerable promise, several technical and operational challenges emerged during its design, implementation, and evaluation:

1. Prompt Ambiguity: Vague or underspecified prompts often resulted in incorrect or incomplete IaC. The LLMs sometimes made assumptions that were contextually incorrect, leading to the generation of unoptimized or misconfigured resources. This necessitates prompt standardization or interactive refinement.

2. Model Hallucinations:A known limitation of LLMs is their tendency to hallucinate fabricating configuration parameters or resource types that do not exist. Although rare, such hallucinations can be dangerous in production environments. To address this, all output was passed through a post-processing validation filter.

3. Security Vulnerabilities: Some generated templates lacked security hardening (e.g., open security groups, permissive IAM roles). While security tools like Checkov and OPA helped flag these issues, full automation of secure configuration generation remains a work in progress.

4. Token Limits and Latency:LLMs have input and output token limits that constrain the size of prompts and responses. For large infrastructure definitions, prompts had to be chunked or summarized, occasionally leading to context loss. Latency from API responses also introduced delays during interactive use.

5. Toolchain Integration: Ensuring compatibility with a wide range of IaC validators, cloud SDKs, and DevOps toolchains required significant engineering effort. Real-time GitOps integration, state file management, and rollback safety are areas that need continued refinement.

6. Compliance and Governance: Organizations operating under strict compliance regimes must ensure that generated IaC adheres to specific policies. Automated policy enforcement mechanisms must be integrated into PromptOps for broader adoption.

These challenges underscore the importance of combining LLM capabilities with robust validation pipelines, human oversight, and secure DevOps practices.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

VIII. CONCLUSION

PromptOps represents a compelling new frontier in DevOps automation, merging the interpretive power of Large Language Models with the declarative rigor of Infrastructure as Code. By allowing infrastructure to be defined through conversational prompts, PromptOps reduces barriers to entry, accelerates development cycles, and minimizes configuration-related errors.

Our research shows that PromptOps achieves high accuracy, broad compatibility, and favorable user perception. Through effective use of prompt engineering, validation tooling, and human-in-the-loop workflows, it proves that AI can responsibly assist in infrastructure management tasks that were previously labor-intensive and error-prone.

However, realizing the full potential of PromptOps requires addressing critical challenges in security, compliance, prompt design, and continuous learning. Ongoing research and development should focus on integrating trust mechanisms, context-preserving interfaces, and feedback-driven refinement loops. Furthermore, organizations must invest in governance models and safeguards to ensure that LLM-assisted automation aligns with operational standards.
In closing, PromptOps paves the way for a future where infrastructure becomes conversational, intent-driven, and adaptive ushering in a new era of intelligent DevOps practices.

REFERENCES

1. HashiCorp. (2023). Terraform Documentation. https://developer.hashicorp.com/terraform
2. OpenAI. (2023). GPT-4 Technical Report. https://openai.com/research/gpt-4
3. Microsoft. (2023). Azure Resource Manager Templates. https://learn.microsoft.com/en-us/azure/azure-resource-manager/
4. Amazon Web Services. (2023). AWS CloudFormation User Guide. https://docs.aws.amazon.com/AWSCloudFormation
5. Chen, J., Zhang, H., & Li, S. (2023). Evaluating LLMs for Infrastructure Automation. *IEEE Transactions on Cloud Computing*, 11(2), 130–145.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com