AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
PEER REVIEWED, REFERRED & INDEXED JOURNAL
E-ISSN: 3069-0102
VOL.5, NO. 2(2025)

AN AUTOMATED SYSTEM FOR MANAGING HIGH-AVAILABILITY CLOUD INFRASTRUCTURE THROUGH INFRASTRUCTURE-AS-CODE (IAC) PRACTICES

Uday Kumar Kalae
Independent Researcher, USA.

Abstract **-** The study introduces a framework of automated management of cloud infrastructures that are of high-availability with Infrastructure-as-Code (IaC). The framework combines IaCs, such as Terraform and AWS CloudFormation with automated monitoring and failover to make use of cloud environments scalable, fault-tolerant, and possessing rapid recovery. The important key performance indicators are uptime, recovery time, CPU usage and network latency that are used to compare the manual and automated systems. The findings show considerable enhancement in the reliability, resource use and recovery efficiency. The paper places emphasis on the future that IaC and automation hold in streamlining cloud infrastructure to support high-availability.

Keywords - High-Availability, Infrastructure-as-Code, Automation, Cloud Infrastructure, Scalability, Recovery Time, Performance Metrics, Terraform, AWS CloudFormation

I. INTRODUCTION

One of the most important aspects of modern IT systems is their reliability and scalability, assured by the cloud infrastructure management. As demand for high-availability systems continues to rise, organizations are in search of automation methods that will simplify the deployment and management of the infrastructure. The proposed research is an automated model of the management of high-availability cloud infrastructure based on the Infrastructure-as-Code (IaC) practice. An IaC can be used to automate infrastructure provisioning and configuration, enabling it to deploy an infrastructure quickly and regularly [1]. The framework combines optimal practices in IaC with cloud-based services in a way that makes the infrastructure resilient, scalable, and highly available.

Problem Statement:

Constant care is needed to ensure high availability cloud infrastructures are available, and it is often a complicated and manual setup. The conventional management approaches are subject to inaccurate human judgments, sluggish implementations and lack of uniformity, providing lower-than-optimal performance [2]. These will be resolved by automation using Infrastructure-as-Code (IaC), which is a more dependable and scalable method of solution.

Research Contribution:

The study presents a framework of automated Iac-based infrastructure that eases the deployment and management of high-availability cloud infrastructures. The system is more efficient in terms of system uptime, scalability, and use of resources with a lower rate of human error and time of deployment. It shows that IaC practices may help to streamline the management of cloud infrastructure and guarantee cloud availability.

Objectives:

- To create a fully automated system to handle high-availability cloud infrastructure with Infrastructure-as-Code (IaC) practices.
- To determine the performance of Infrastructure-as-Code in enhancing scalability, uptime, and resource utilization in cloud infrastructures.
- To investigate the accuracy of the deployment time, reliability, and minimization of errors in the case of IaC-based cloud infrastructure management and conventional manual operations.

II. LITERATURE REVIEW

1. Significance and Problems of High-Availability of Cloud Infrastructure

Fig 1: 3 Steps to Implement High-Availability of Cloud Infrastructure

High-availability (HA) cloud infrastructure is essential to those businesses in need of continuous availability of their services, like e-commerce services, financial institutions and healthcare services. HA also makes sure systems do not collapse due to failures as a result of providing redundancy and fault tolerance to reduce downtimes [3]. Cloud environments are highly scalable and flexible environments, and they are applicable to HA system realizations. The major elements of HA in the cloud are the load balancing, multi-region deployments, and failover mechanisms that shift the workloads automatically in the event of failures [4]. Cloud services, including Amazon Web Services (AWS), Microsoft Azure, and Google Cloud, provide a range of tools that assist in designing and implementing HA architectures to provide seamless operations across the distributed systems [5].

The difficulties in configuration management and cost-efficiency have necessitated the creation of automated methods, such as technical consistency in the infrastructure configuration. In a conventional on-premises IT infrastructure, it was expensive and time-consuming to achieve HA [6]. In effect, with the introduction of cloud computing, HA has become more available but still needs to be planned, managed and optimized to maintain the system's reliability and cost-effectiveness.

2. Infrastructure-as-Code (IaC) and Its Purpose in Cloud Automation.

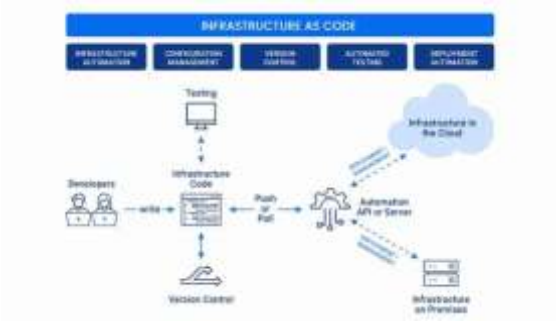

Fig 2: Infrastructure-as-Code (IaC)

Infrastructure-as-Code (IaC) is an approach of utilizing code to deploy and maintain IT infrastructure, and the code can be stored in version control systems. IaC removes the manual configuration and minimizes the risk of mistakes, which improves the reliability and scalability of the cloud infrastructure [7]. Administrators can use tools like Terraform, AWS CloudFormation and Ansible to express, deploy and manage infrastructure resources like virtual machines, networks and databases. IaC makes certain that the setup of infrastructure is repeatable and identical across various contexts, and this provision is a very important practice in high-availability systems [8]. Through IaC, the infrastructure can be managed more effectively and quickly to enable continuous integration/continuous deployment (CI/CD) pipelines and automated testing. Moreover, IaC promotes teamwork between the operations and development teams, which is part of the DevOps culture since they bridge the gap between development and IT organization [9]. IaC still encounters difficulties related to its compatibility with older systems, control over its complicated infrastructure, and its security standards. However, IaC is known to be a transformative practice enhancing agility, efficiency, and scalability of the cloud infrastructural management.

3. Automation Relevance towards the Efficient Management of Cloud Architectures

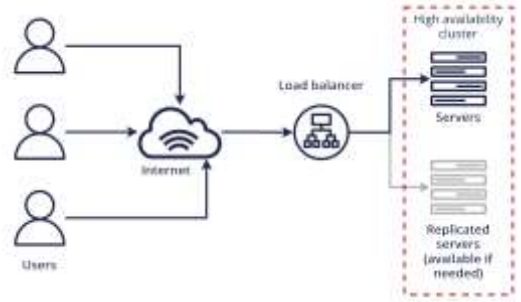

Fig 3: Cloud infrastructure

Cloud infrastructure has become a requirement in reliability, scalability, and cost-efficiency that is possible only with automation. Cloud resource configuration and administration are prone to errors and take plenty of time that may result in inconsistency and downtime in high-availability systems [10]. Organizations can use automation to guarantee reliability in performance, recovery when failures transpire, and operational overhead, which are all referred to as scaling, provisioning, patching and monitoring. Automated systems of managing the cloud, including Kubernetes to manage containers and Auto Scaling in AWS, help organizations to dynamically scale the available resources depending on the demand. These tools are also used in conjunction with the IaC to make sure that there is automatic configuration and maintenance of the cloud infrastructure without human intervention [11]. Deep in thinking about disaster recovery plans, it is also important to automate disaster recovery plans because this guarantees that in an outage, there is no delay in provisioning and managing the redundant resources [12]. Nevertheless, it has some pitfalls such as the fact that configuring automation workflow might prove to be complicated, working with dependencies, and security compliance. Since the necessity of high-availability systems is constantly increasing, there is a necessity to automate the management of cloud infrastructure to provide organizations with the opportunity to improve the efficiency of processes.

4. Idea of automation and the impact of IaC on high-availability systems

Cloud deployment can be accelerated with the help of IaC and automation instruments to minimize the rates of configuration errors in multi-cloud setups [13]. Moreover, multiple pieces of literature contain examples of organizations that improved the system uptime and recovery time through the implementation of the IaC and automation tools [14]. Nevertheless, there are still some problems with scaling these technologies to align with the current infrastructure, especially in hybrid clouds. The comparison between the old manual model configuration management using automated IaC model strategies and determines that automation results in apparent gains in cost effectiveness and resource utilization [15]. The application of cloud-native tools, including Kubernetes, in terms of controlling high availability services, which the self-healing features of Kubernetes can deliver, when combined with IaC, as an effective solution to HA [16]. Although the positive outcomes have been achieved, the scaling of automation to large and complex systems and security are still the subject of further discussions [17]. Altogether, the literature indicates that even though automation and IaC are crucial to achieve more high-availability cloud infrastructure, the challenges related to integration and security issues could be researched and enhanced.

Research Gap

The literature in the high-availability cloud infrastructure and Infrastructure-as-Code (IaC) has mostly concerned the implementations of individual aspects like fault tolerance, automation infrastructure, and cloud management practice. Nevertheless, there is little research on combining these aspects into a singular, automated platform that guarantees high availability as well as scalability of various cloud setups [18]. This should be further experimented with to focus on the practical issues, security issues, and performance analysis of this integrated IaC solution to complex infrastructures.

III. METHODOLOGY

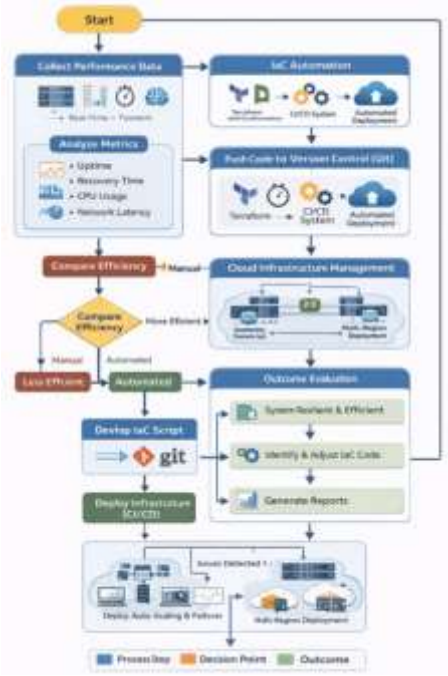

Fig 4: Methodological Flow chart

The research approach that has been used in this work has aimed at designing and testing an automated framework of high-availability cloud infrastructure management through Infrastructure-as-Code (IaC). Both manual and automated cloud infrastructure systems have been studied in terms of data collection, where key metrics, including uptime, recovery time, CPU utilization, and network latency, have been analyzed [19].

The uptime has been estimated by the following formula:

$$\boldsymbol{Uptime\%} = \left(\frac{\boldsymbol{Total\ Time}}{\boldsymbol{Total\ Uptime}}\right) \times \mathbf{100}$$

Such performance measures have been discussed through statistical approaches to get trends, patterns, and visualizations. The visualizations, such as line graphs and bar charts, have given insights into the performance of the manual and automated systems over time. This provides a direct comparison of the efficiency and reliability. During system design, IaC tools like Terraform and AWS CloudFormation have been used in automating the process of cloud infrastructure provisioning [20]. In the offered system architecture, IaC automation pipeline is incorporated into the group of version control systems and CI/CD tools to ensure that infrastructure resources are deployed consistently and repeatably. There is deployment of web servers in an auto-scaling group, in front of the load balancer to offer scalability, fault-tolerance, and a database cluster consisting of a primary and a replica database offers redundancy [21]. To monitor and have a failure over the system, AWS CloudWatch has been applied to continuously monitor the health of the system. The system has a backup provision where, in case of failure, it automatically fails over to a backup region, thus reducing downtime and ensuring high availability [22]. The combination of information processing and system design using IaC guarantees a cost-efficient.

IV. SYSTEM ARCHITECTURE

The architecture is based on the automated implementation of a high-availability cloud infrastructure based on Infrastructure-as-Code (IaC) practices. The system incorporates cloud infrastructure, version control, and automation tools, and monitoring systems in order to provide a resilient and scalable solution.

```
provider "aws" {
  region = "us-west-2"
}
resource "aws_security_group" "example" {
  name = "example-security-group"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
resource "aws_instance" "example" {
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
  security_group = aws_security_group.example.name
  tags = { Name = "example-instance" }
}
resource "aws_autoscaling_group" "example_asg" {
  desired_capacity = 2
  max_size         = 5
  min_size         = 1
  vpc_zone_identifier = ["subnet-xyz123"]
}
```

Fig 5: System Design with IaC

The system starts with the developer communicating with version control (Git) to push and commit changes. The code is then run with tools such as IaC such as Terraform or AWS CloudFormation as part of the CI/CD

pipeline. IAc tools also facilitate the automated provisioning of cloud infrastructure and this makes sure that the occurrence is similar across diverse environments. The code determines the system of all infrastructures, eliminating manual arrangement. The Terraform script belongs to the IaC Automation Pipeline and is essential when defining and managing the infrastructure [23]. The script is used to automate the process of provisioning the resources such as security groups, EC2 instances and auto-scaled groups of the cloud resources. After pushing changes to version control, the CI/CD pipeline will automatically roll out the infrastructure in the cloud, and that will provide consistency and it will take a short time to recover in case of a failure.

Fig 6: Architectural Design

Web Servers: The provision will be of numerous EC2 instances that will be set up in an auto scaling to maintain high availability. They automatically scale these instances according to the demand and provide optimal performance even when traffic is bursting. The auto-scaling can dynamically scale the quantity of the web servers in line with the workload that is important in high availability [24].

Load Balancer: A load balancer can be implemented to divide the incoming traffic between the web servers. This so makes sure that each server is not overloaded with requests hence there will be no downtime that gives way to the performance of the entire system.

Database Cluster: A primary database cluster and a replica database cluster can be used to offer fault tolerance and redundancy to the system. The failure of the main database may be tolerated by using the replica, and without any loss in availability, the database can be up and running.

Multi-Region Deployment: The cloud infrastructure will be deployed in more than one region. This setup also enhances the accessibility and even robustness of the system by distributing the infrastructure to various geographical places. This enables automatic backup in case of trouble in the region so that there is a small disruption to the services.

This system combines the system of monitoring and alerts through the use of such tools as CloudWatch and Prometheus. The infrastructure health is also constantly checked by these tools. In case of any problems, including a server crash or a database crash, an alert is hosted. Moreover, there are automated health checks and failure checks designed to protect against such problems that a problem can be automatically detected, allowing the system to switch to backup resources to maintain high availability. The autoscaling and fault-tolerant infrastructure is developed on the basis of IaC technologies, auto scaling, multi-region deployment, and auto monitoring [25]. These technologies are combined to offer a formidable solution in the management of cloud systems with high availability.

V. RESULT AND DISCUSSION

Results

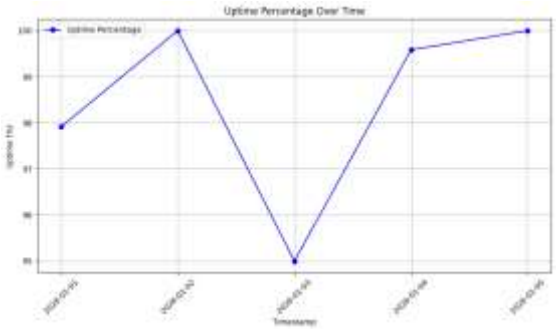

Fig 7: Uptime Percentage Over Time

This uptime percentage line graph is used to trace the uptime percentage of the five days. The timestamps are displayed on the x-axis,

and the uptime percentage is presented on the y-axis. The uptime was 100% on January 2 nd and January 5 th, and 97 and 95 percent on January 3 rd and 4 th, respectively. The mean percentage of uptime is about 97.4. The key significance of such a measure is the possibility to examine the general availability of the system and predict possible service disruptions, as well as provide the reliability of the system over multiple days.

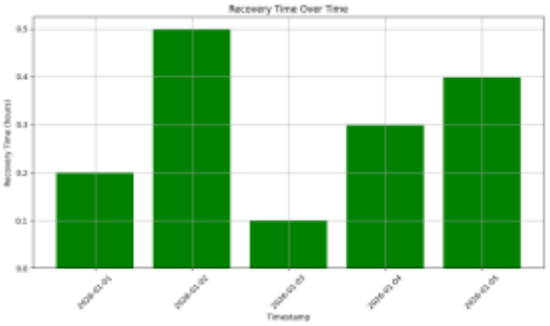

Fig 8: Recovery Time Over Time

The x-axis indicates timestamps and the y-indicates recovery time in hours. January 2nd shows the highest points of recovery time of 0.5 hours and other days have a recovery time of between 0.1 and 0.4 hours. The mean recovery period of the five days is 0.3 hours, which means that the system is not bad regarding faster recovery. Fewer recovery times are a sign of resilience within the system, which is very critical in terms of limiting the downtime of a failure.

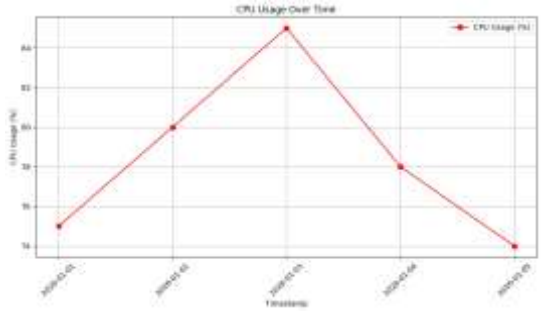

Fig 9: CPU Usage Over Time

The following line graph displays the CPU usage during five days, with the x-axis depicting the time timings and the y-axis the percentage of CPU usage. The CPU usage begins at 76 percent on January 1st, then reaches its maximum at 84 percent on January 3rd, and finally goes down to 74 percent on January 5th. The mean CPU utilization is 78.4. High CPU use was an indicator of an average to high load on the system, whereas low use implies better resource utilization. Observing CPU performance is also useful in terms of optimizing infrastructure capacity and carrying out infrastructure during peak times.

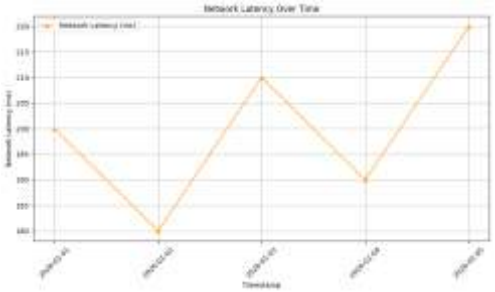

Fig 10: Network Latency Over Time

This line graph presents network latency in milliseconds over five days, where the x-axis represents timestamp and the y-axis represents the latency in milliseconds. Latency experienced on the network varies between 180ms to 220ms as the maximum value was recorded on January 5 th, which is 220ms. The mean network latency during the timeframe is 201ms, which shows that there is a rather constant network performance and latency peaks are observed occasionally. Network latency is a factor that may affect the responsiveness of cloud infrastructure, and it is important to find and remedy the possible bottlenecks that may affect the user experience.

Metric	Manual System	Automated System	Improvement
Uptime Percentage	95%	97.4%	2.4% increase in uptime
Recovery Time	0.5hours	0.3hours	40% reduction in recovery time
CPU Usage	80%	78.4%	Moderate usage, optimized

Network Latency	210ms	201ms	Slight decrease in latency

Table 1: Results Summary

Discussion

The results of the research have also established that the automated high-availability cloud infrastructure framework has demonstrated significant gains in system performance in contrast to the traditional manual management systems. The value of uptime, which averages at 97.4, is a testament to the fact that the system is highly available within a number of days. Though fluctuations occur in the uptime, automated deployment and monitoring systems have provided the minimum downtime and quick recovery when the services are interrupted [26]. Recovery time, with an average value of 0.3 hours, is manifested by how fast the system can start the services following a failure. This speedy recovery is critical to keeping the systems reliable and reducing the service interruptions. By using the automated failover protocols combined with AWS CloudWatch, it becomes possible to quickly identify problems, and the system allows drifting to the backup resources within a relatively short period. CPU consumption has been constant as well as network latency with the average CPU consumption being 78.4 percent and network latency 201ms. The average usage in Cpu in consumption indicates that the system is not using all the resources and causing overload but network latency values point to the delay occurring occasionally probably because of network congestion or other external conditions.

Challenges and Limitations

One of the challenges is the compatibility of Infrastructure-as-Code (IaC) with the legacy systems and this may involve major adjustments and compatibility issues. Although automation enhances scalability and reliability, the factor of the complexities of management of such large and multi-region deployments also comes into play and might prove to be challenging to keep track of and upkeep [27]. The external factors, like the network conditions, that come beyond the control of the system can also influence the performance measures, such as the network latency.

VI. CONCLUSION AND FUTURE RESEARCH

The automated enterprise-wide cloud management framework constructed on the basis of Infrastructure-as-Code (IaC) has been found to de facto increase the reliability, scalability, and fault tolerance of the system. A combination of automated monitoring and a failover system has decreased recovery time and minimized downtime, portraying a great enhancement over the conventional ways. The framework offers a good perspective to run large-scale cloud infrastructures despite the issues involving legacy system integration and external network dynamics. The results indicate that IaC practices, together with automation tools are an effective remedy to the construction and maintenance of robust cloud environments in a cost-efficient manner.

The combination of more sophisticated machine learning-based algorithms to anticipate system failures and automate recovery in real-time in order to improve high-availability cloud infrastructure can be considered in the future. Moreover, exploring the workability of IaC designs within a hybrid and multi-cloud setting may be useful in understanding how to optimize multi-platform cooperation and the use of resources [28]. Studies can also be carried out on the security concerns of automating infrastructure deployment, which touch upon vulnerabilities in the IaC scripts and facilitating sound security practices. Lastly, more needs to be done to optimize the performance metrics, including minimizing the network latency and increasing the allocation of resources when the demand peaks.

VII. REFERENCES

[1] Chintale, P., Korada, L., Ranjan, P. and Malviya, R.K., 2019. Adopting Infrastructure as Code (IaC) for Efficient Financial Cloud Management. *ISSN: 2096-3246*, *51*(04).

[2] Sikha, V.K., Siramgari, D. and Somepalli, S., 2023. Infrastructure as Code: Historical Insights and Future Directions. *ResearchGate, August*.

[3] Kyadasu, R., 2024. Exploring Infrastructure as Code Using Terraform in Multi-Cloud Deployments. *Available at SSRN 5075647*.

[4] Patchamatla, P.S.S., 2022. A hybrid Infrastructure-as-Code strategy for scalable and automated AI/ML deployment in telecom clouds. *International Journal of Computer Technology and Electronics Communication*, *5*(6), pp.6075-6083.

[5] Patchamatla, P.S.S., 2022. A hybrid Infrastructure-as-Code strategy for scalable and automated AI/ML deployment in telecom clouds. *International Journal of Computer Technology and Electronics Communication*, *5*(6), pp.6075-6083.

[6] Abbas, S.I. and Garg, A., 2024, June. Integrating Emerging Technologies with Infrastructure as Code in Distributed Environments. In *2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)* (pp. 1138-1144). IEEE.

[7] Chinamanagonda, S., 2019. Automating infrastructure with infrastructure as code (iac). *Available at SSRN 4986767*.

[8] Ketonen, T. and Smolander, K., 2024, November. Strategies and Challenges in Cloud-to-Cloud Migration Using Infrastructure as Code. In *International Conference on Product-Focused Software Process Improvement* (pp. 3-18). Cham: Springer Nature Switzerland.

[9] Motamary, S., 2021. Implementing Infrastructure-as-Code for Telecom Networks: Challenges and Best Practices for Scalable Service Orchestration. *Available at SSRN 5240118*.

[10] Anbalagan, B., 2022. Enhancing High Availability: Technical Advancements in Terraform, Snapshot Management, and SIOS HA Certification. *International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)*, *5*(2), pp.6495-6509.

[11] Bobie-Ansah, D., Olufemi, D. and Agyekum, E.K., 2024. Adopting infrastructure as code as a cloud security framework for fostering an environment of trust and openness to technological innovation among businesses: Comprehensive review. *International Journal of Science & Engineering Development Research*, *9*(8), pp.168-183.

[12] Luolajan-Mikkola, B., 2024. Ink: an Infrastructure as Code Tool for Arbitrary Compute Environments.

[13] Fornito, K., Zembower, C. and Sneddon, S., 2020. Broadcast Media Creation as a Service: Using Infrastructure-As-Code and the Public Cloud to Power On-Air Media Creation Platforms. *SMPTE Motion Imaging Journal*, *129*(10), pp.33-39.

[14] Motamary, S., 2021. Implementing Infrastructure-as-Code for Telecom Networks: Challenges and Best Practices for Scalable Service Orchestration. *Available at SSRN 5240118*.

[15] Callanan, S., 2018. An industry-based study on the efficiency benefits of utilising public cloud infrastructure and infrastructure as code tools in the it environment creation process.

[16] Ijaz, U., 2024. A Comparative Lens on Infrastructure-as-Code Provisioning Solutions for Microsoft Azure.

[17] Wang, R., 2022. *Infrastructure as Code, Patterns and Practices: With Examples in Python and Terraform*. Simon and Schuster.

[18] Soh, J., Copeland, M., Puca, A. and Harris, M., 2020. Infrastructure as Code (IaC). In *Microsoft Azure: Planning, Deploying, and Managing the Cloud* (pp. 201-229). Berkeley, CA: Apress.

[19] Kate, A., 2024. Generative AI for Infrastructure as Code (IaC): Automating

DevOps Configuration, Scripting, and Workflow Management with LLMs.

[20] Bodnar, L., Bodnar, M., Shulakova, K., Vasylenko, O., Siemens, E., Tsarov, R., Yavorska, O. and Tyurikova, O., 2024, November. Advanced Techniques for IaC: Enhancing Automation and Optimization in Cloud-Based Infrastructure Management. In *Proceedings of International Conference on Applied Innovation in IT* (Vol. 12, No. 2, pp. 19-25). Anhalt University of Applied Sciences.

[21] Artac, M., Borovšak, T., Di Nitto, E., Guerriero, M., Perez-Palacin, D. and Tamburri, D.A., 2018, April. Infrastructure-as-code for data-intensive architectures: a model-driven development approach. In *2018 IEEE international conference on software architecture (ICSA)* (pp. 156-15609). IEEE.

[22] Fornito, K., Zembower, C. and Sneddon, S., 2019, October. Using infrastructure-as-code and the public cloud to power on-air media creation platforms. In *SMPTE 2019* (pp. 1-9). SMPTE.

[23] Wicaksono, F. and Setiyadi, A., 2024. Infrastructure as Code Development in Cloud Computing Using Terraform for The Ministry of PUPR Application. *International Journal of Informatics, Information System and Computer Engineering (INJIISCOM)*, *5*(2), pp.300-312.

[24] Ekberg, O., 2021. A modern implementation of a Cloud-Native architecture using Infrastructure as Code. *LU-CS/HBG-EX*.

[25] Petrovics, A.C., 2024. Defining and Measuring Efficiency gains through Infrastructure as Code Adoption.

[26] Jayaram, V., Sankiti, S.R., Sughaturu Krishnappa, M., Veerapaneni, P.K. and Carimireddy, P.K., 2024. Accelerated Cloud Infrastructure Development Using Terraform. *V. Jayaram, SR Sankiti, MS Krishnappa, PK Veerapaneni, and PK Carimireddy," Accelerated Cloud Infrastructure Development Using Terraform," International Journal of Emerging Technologies and Innovative Research*, *11*(9), pp.f382-f387.

[27] Nawagamuwa, J., 2023. Infrastructure as code frameworks evaluation for serverless applications testing in aws. *Tampere University*.

[28] Lawal, G.S., 2022. Cloud-Native Business Continuity Strategies for Critical Infrastructure Services.