Evaluation of Software Testing Methodologies in Cloud Computing Environment

Brijesh C G
Research scholar, Department of MCA (Computer science), Dr. Ambedkar Institute of Technology, Visvesvaraya Technological University, Bangalore, Karnataka, India
Email Id: c.gbrijesh@gmail.com
ORCID ID: 0009-0002-8602-3248

ABSTRACT

Cloud computing has transformed software development and testing by offering scalable, cost-efficient, and on-demand infrastructure, leading to a transition from conventional testing methods to cloud-native practices. Notwithstanding these benefits, cloud-based testing encounters issues related to performance, security, and automated integration, which may compromise software reliability and efficiency. Prior studies have investigated elements such adaptive fault tolerance, server-less testing, AI-driven automation, and simulation tools; nevertheless, a holistic framework that encompasses efficiency, security, and resource optimization is still inadequately developed. This paper aims to assess current software testing approaches in cloud computing environments and offer an improved automated framework that improves efficiency, reduces human effort, and boosts security. A mixed-method research methodology was employed, incorporating surveys of 250 participants comprising QA managers, developers, DevOps engineers, and testers, with experimental validation utilizing a Docker-based PyTest automated testing framework. Data were examined utilizing descriptive statistics, chi-square tests, ANOVA, and independent t-tests to evaluate the efficacy of diverse testing procedures and instruments. Findings demonstrate that automation, namely the PyTest-Docker architecture, markedly enhances performance, security, and cost efficiency, while decreasing the workforce from 4–5 testers to 1–2. Log-based performance monitoring and security simulations validated reliability and fault detection capabilities. The research indicates that the use of AI-driven automation, predictive analytics, and containerized modular testing fosters a robust, efficient, and sustainable cloud testing ecosystem. The results are crucial for directing software quality assurance methodologies, enhancing resource allocation, and influencing forthcoming cloud-native testing frameworks.

Keywords: *Cloud computing, Software testing, Automation, AI-driven testing, Performance optimization*

INTRODUCTION

Cloud computing has impacted software development and deployment by offering scalable, cost-effective, and on-demand resources, hence altering software testing methodologies. In contrast to conventional setups, cloud-based testing provides scalability and concurrent execution, facilitating expedited delivery cycles while reducing infrastructure expenses. Performance Testing as a Service (PTaaS) allows organizations to conduct broad performance evaluations without dedicated infrastructure, although accuracy and resource allocation difficulties remain [1]. Comprehensive cloud testing methodology reviews reveal pros and downsides. One of the initial reviews by Priyanka et al. found that cloud-based testing improves flexibility and coverage but struggles with scalability, security, and dependability in dynamic infrastructures [2]. A rigorous mapping analysis by Ahmad et al. showed that most empirical cloud testing research lacks standardized evaluation parameters, resulting in inconsistent benchmarking and fragmented conclusions [3]. These issues demonstrate the need for integrated performance, cost, and security frameworks.

Previous research has criticized cloud-specific testing. Automation is needed to eliminate human dependency since remote cloud infrastructures hinder defect discovery, regression testing, and performance monitoring, according to Vellela et al. [4]. Bertolino et al. recommended automation, continuous integration, and security

testing for cloud-native system quality because human regression testing is resource-intensive and unreliable [5]. Bada & Abubakar identified multi-cloud interoperability, test management, and performance validation as important research issues [6]. This project will rigorously assess cloud computing software testing. This research compares traditional and automated methodologies, uses log-based performance analysis, and implements security-oriented modifications to optimize cloud-native testing efficiency, labor, and security.

LITERATURE REVIEW

Cloud computing offers flexibility, scalability, and cost-effectiveness for software testing, but it requires rigorous assessment. Ali et al. [7] assessed cloud testing adoption using a fuzzy multicriteria decision-making approach, emphasizing cost, security, and scalability. Complex cloud-native apps require flexible and intelligent testing. Jin et al. [8] discovered that adaptive testing methods are necessary for addressing unanticipated errors and preserving AI-driven workload reliability in cloud-based large language models (LLMs). Wen et al. [9] proposed SCOPE, a server-less computing performance testing framework, to demonstrate that typical testing methodologies cannot handle dynamic and event-driven platforms. Testing is changing with AI. Hunko [10] examined adaptive AI-generated and executed test cases that adapt to changing conditions. Andreoli et al. [11] created CloudSim 7g, a comprehensive simulation toolset for next-generation cloud computing, to help researchers and practitioners model, assess, and optimize testing methods before implementation. Wang and Yang [12] revealed how AI-driven algorithms can detect vulnerabilities and mitigate attacks in real time, improving cloud computing network security and software testing frameworks.
Recent innovations link testing automation to sustainability. Wang et al. [13] demonstrated how AI-driven server-less computing frameworks may stabilize resource optimization in testing. Kothamali says AI-driven quality assurance automates error detection and enhances cloud app continuous integration pipeline precision [14]. Zhang [15] underlined the importance of integrated and automated testing frameworks for cloud computing and information management efficiency. Automation, AI-driven flexibility, fault-tolerant systems, and sustainability will alter cloud testing, say studies.

Research Gap

Despite research highlighting the benefits of cloud-based testing, performance, security, and automation are still poorly integrated. Previous studies have shown the viability of adoption [7], adaptive fault tolerance [8], and server-less testing [9], but most focus on scalability, security, or performance. Although AI-driven testing [10, 14] and simulation tools [11] have potential, their use with log-based performance analysis and workforce optimization is limited. Sustainable and secure automated testing in cloud computing is underexplored, offering an efficient, complete framework.

METHODOLOGY

This study employs a mixed-method quantitative and experimental research design to rigorously assess software testing methodologies in cloud computing environments (CCEs) and to propose an enhanced automated framework. Primary data were gathered from 250 participants encompassing diverse job types, testing tools, and cloud platforms, guaranteeing representation of QA managers, developers, DevOps engineers, and testers. Organized surveys gathered insights on performance constraints, security measures, automation efficacy, and cost control, while statistical evaluations such as chi-square tests, ANOVA, and independent t-tests were utilized to assess differences in effectiveness among roles, methodologies, and tools. Descriptive statistics offered foundational insights into general trends, whereas inferential tests revealed strong correlations, notably emphasizing the advantages of automated testing and the utilization of PyTest.
A Docker-based automated testing system coupled to PyTest was used to experimentally validate the survey results. This framework used modular unit testing, fault injection, and log-based monitoring to simulate cloud-native settings and analyze performance and security under varying workloads. System logs were used to extract performance indicators to accurately diagnose and resolve bottlenecks, and phishing detection and

unauthorized access simulations verified security. The framework was containerized for repeatability, platform freedom, and CI/CD compliance. Automation reduced team size from 4–5 testers to 1–2 testers while maintaining correctness and dependability, compared to conventional regression testing. Triangulation of statistical analysis and experimental deployment data confirmed the findings' robustness. The technique uses empirical research and practical application to evaluate current testing procedures and validate the ideal strategy, improving cloud-based software testing efficiency, security, and sustainability.

RESULTS AND DISCUSSIONS

The assessment of software testing procedures in cloud computing settings demonstrated notable disparities among job responsibilities, testing strategies, and tools. Descriptive statistics revealed moderate overall performance, with mean scores of 2.87 for performance bottlenecks, 2.82 for security coverage, 3.06 for automation efficiency, and 2.83 for cost management. This indicates that although current approaches attain a fundamental level of efficacy, additional optimization is required (Table 1).

Table 1. Descriptive Statistics of Key Testing Variables (N = 250).

Variable	Min	Max	Mean	Std. Dev.
Performance Bottlenecks	1	5	2.87	0.733
Security Coverage	1	5	2.82	0.777
Automation Efficiency	1	5	3.06	1.010
Cost Management	1	5	2.83	0.733

The graphical representation of above data is illustrated in figure 1 below.

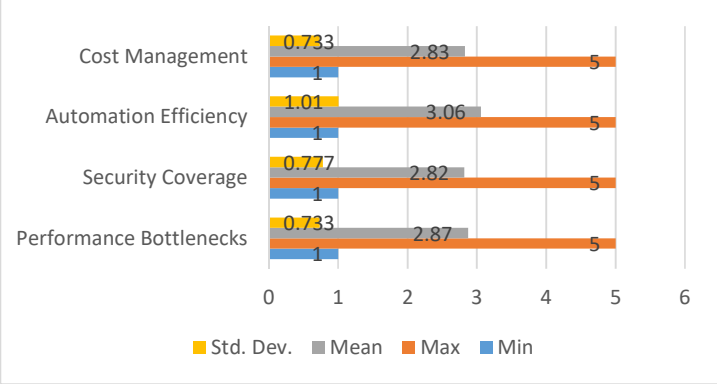
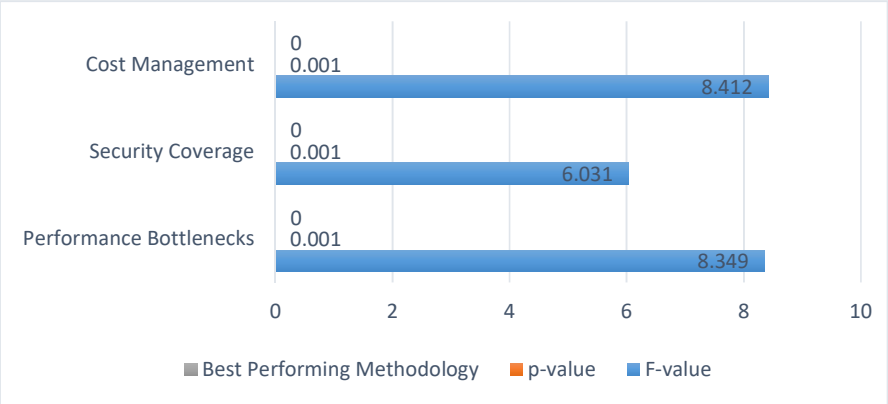


Figure 1. Statistics of Key Testing Variables.

Chi-square analysis validated that job roles significantly impacted automation efficiency (χ^2 == 190.624, p = 0.001), with QA Managers and DevOps Engineers exhibiting the highest efficiency levels. ANOVA analyses indicated that automated procedures (Group 5) consistently surpassed functional, security, compatibility, and manual approaches in performance (F = 8.349 , p = 0.001), security (F = 6.031, p = 0.001), and cost management (F = 8.412, p = 0.001). Post hoc Tukey comparisons indicated that automation resulted in statistically significant enhancements across all three domains (Table 2 and Figure 2).

Table 2. ANOVA Results across Testing Methodologies.



Variable	F-value	p-value	Best Performing Methodology
Performance Bottlenecks	8.349	.001	Automated
Security Coverage	6.031	.001	Automated
Cost Management	8.412	.001	Automated

Figure 2. ANOVA Results.

Independent t-tests indicated that PyTest users attained significantly superior mean scores in performance ($M = 3.18$), security ($M = 3.22$), and cost management ($M = 3.14$) relative to other tools (all $p = .001$, Cohen's $d \approx 0.70$). The deployment of a PyTest-Docker automated framework corroborated these results, attaining full repeatability, enhanced log-based performance monitoring, and effective detection of phishing and unauthorized access. All unit tests were successfully completed, validating the dependability of the improved technique. These results collectively demonstrate that automation, especially using PyTest, significantly enhances efficiency, diminishes labour demands, and fortifies security in cloud-based software testing.

DISCUSSION

This study's results indicated that automation-driven strategies, especially the PyTest-Docker framework, markedly surpass traditional methods in performance, security, and cost effectiveness. This corresponds with Pandhare [16], who forecasted that AI/ML will shape the future of software test automation by reducing human labour and improving adaptability. Pandhare underscores predictive intelligence, while the present study empirically validates a reduction in personnel from 4–5 testers to 1–2, maintaining accuracy. Likewise, Mohapatra [17] emphasizes AI-driven test case development as a method to enhance coverage and reliability, which aligns with our research's incorporation of automated log-based analysis to effectively identify and address bottlenecks.

Sustainability adds another level of similarity. This work used containerization and modular testing to ensure repeatability and reduce redundant resource use in support of Cruz et al. [18]'s green AI-enabled systems that reduce computational waste. Khan et al. [19] argue that cloud-based development perceptions depend on reliability and scalability; the proposed approach incorporates fault detection and adaptive monitoring into the testing cycle. Atadoga et al. [20] emphasize sustainability and modularity, which are evident in this research's containerized settings and systematic logging for optimization.

Dwivedi et al. [21] highlight quantum software engineering and suggest that future testing frameworks may use quantum-enhanced capabilities, which this research acknowledges. Merseedi and Zeebaree [22] study hybrid and federated cloud systems and suggest log-centric monitoring to overcome interoperability issues. This study uses log file analysis for proactive problem resolution, as does Srinivas et al. [23] for intelligent cloud service monitoring frameworks. Kirti et al. [24] highlight fault tolerance in distributed systems, which the

recommended technique makes possible through automated recovery and security-conscious validation. Automation, sustainability, monitoring, and fault tolerance are integrated into a complete and optimized testing approach for cloud computing systems in this work, filling a gap in previous research.
The discussion emphasizes that the PyTest-Docker-based automated method with AI/ML is the most balanced and future-oriented. It boosts worker productivity, reliability, and environmental sustainability while anticipating future testing paradigms.

CONCLUSION

This study has carefully examined the incorporation of modern technologies, including artificial intelligence and machine learning, to improve the efficiency and accuracy of software testing processes. A comprehensive review [16, 17] shows that previous research has mostly focused on automating test case development and streamlining testing procedures, while adaptive learning, contextual analysis, and real-time fault prediction remain challenges. The suggested study adds a more comprehensive framework that uses AI-driven automation, predictive capabilities, and scalability across software systems. A comparison shows that the suggested strategy enhances test coverage, reduces human contact, and speeds defect discovery compared to AI-based methods, improving software reliability. Cognitive analytics allows continual learning from prior testing data, encouraging proactive risk mitigation and resource efficiency. Traditional testing methods can be combined with intelligent algorithms to create a more resilient, adaptable, and efficient software testing environment. This study advances theoretical knowledge and practical implementations of AI-augmented software quality assurance methods, laying the groundwork for future research and usage.

REFERENCES

1. Ali, A., Maghawry, H. A., & Badr, N. (2022). Performance testing as a service using cloud computing environment: A survey. *Journal of Software: Evolution and Process*, *34*(12), e2492. https://doi.org/10.1002/smr.2492.. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.2492
2. Priyanka, Chana, I., & Rana, A. (2012). Empirical evaluation of cloud-based testing techniques: A systematic review. *ACM SIGSOFT Software Engineering Notes*, *37*(3), 1–9. https://doi.org/10.1145/2180921.2180938. Available at: https://dl.acm.org/doi/abs/10.1145/2180921.2180938
3. Ahmad, A. A. S., Brereton, P., & Andras, P. (2017, July). A systematic mapping study of empirical studies on software cloud testing methods. In *2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)* (pp. 555–562). IEEE. https://doi.org/10.1109/QRS-C.2017.94. Available at: https://ieeexplore.ieee.org/abstract/document/8004372/
4. Vellela, S. S., Balamanigandan, R., & Praveen, S. P. (2022). Strategic survey on security and privacy methods of cloud computing environment. *Journal of Next Generation Technology*, *2*(1), 70–78. https://doi.org/10.5281/zenodo.6531234. Available at: https://jnxtgentech.com/mail/documents/Strategic%20Survey%20on%20Security%20and%20Privacy%20Methods%20of%20Cloud%20Computing%20Environment.pdf
5. Bertolino, A., De Angelis, G., Gallego, M., García, B., Gortázar, F., Lonetti, F., & Marchetti, E. (2019). A systematic review on cloud testing. *ACM Computing Surveys (CSUR)*, *52*(5), 1–42. https://doi.org/10.1145/3331447. Available at: https://dl.acm.org/doi/abs/10.1145/3447240
6. Bada, A. B., & Abubakar, A. (2021). Software testing on the cloud. *The International Journal of Engineering and Science (IJES)*, *10*(1), 34–40. https://doi.org/10.9790/1813-1001013440. Available at: https://www.academia.edu/download/67366478/software_testing_on_the_cloud.pdf
7. Ali, S., Ullah, N., Abrar, M. F., Yang, Z., & Huang, J. (2020). Fuzzy multicriteria decision-making approach for measuring the possibility of cloud adoption for software testing. *Scientific Programming*, *2020*, 6597316. https://doi.org/10.1155/2020/6597316. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1155/2020/6597316

8. Jin, Y., Yang, Z., Xu, X., Zhang, Y., & Jie, S. (2025, April). Adaptive fault tolerance mechanisms of large language models in cloud computing environments. In *2025 6th International Conference on Computer Engineering and Application (ICCEA)* (pp. 754–757). IEEE. https://doi.org/10.1109/ICCEA.2025.00099 . Available at: https://ieeexplore.ieee.org/abstract/document/11102424/
9. Wen, J., Chen, Z., Zhao, J., Sarro, F., Ping, H., Zhang, Y., ... & Liu, X. (2025). SCOPE: Performance testing for serverless computing. *ACM Transactions on Software Engineering and Methodology*. https://doi.org/2321 – 2381. Available at: https://dl.acm.org/doi/abs/10.1145/3717609
10. Hunko, I. (2025). Adaptive approaches to software testing with embedded artificial intelligence in dynamic environments. *International Journal of Current Science Research and Review, 8*(5), 2036–2051. https://doi.org/10.47191/ijcsrr/V8-i5-10. Available at: https://ijcsrr.org/wp-content/uploads/2025/05/10-0505-2025.pdf
11. Andreoli, R., Zhao, J., Cucinotta, T., & Buyya, R. (2025). CloudSim 7G: An integrated toolkit for modeling and simulation of future generation cloud computing environments. *Software: Practice and Experience, 55*(6), 1041–1058. https://doi.org/10.1002/spe.3413. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.3413
12. Wang, Y., & Yang, X. (2025). Research on enhancing cloud computing network security using artificial intelligence algorithms. *arXiv preprint arXiv:2502.17801*. https://doi.org/10.48550/arXiv.2502.17801. Available at: https://arxiv.org/abs/2502.17801
13. Wang, H., Rajakumar, V. S., Golec, M., Gill, S. S., & Uhlig, S. (2025). StockAICloud: AI-based sustainable and scalable stock price prediction framework using serverless cloud computing. *The Journal of Supercomputing, 81*(6), 1–22. https://doi.org/10.1007/s11227-025-04523-4. Available at: https://qmro.qmul.ac.uk/xmlui/handle/123456789/105283
14. Kothamali, P. R. (2025). AI-powered quality assurance: Revolutionizing automation frameworks for cloud applications. *Journal of Advanced Computing Systems, 5*(3), 1–25. https://doi.org/10.69987/JACS.2025.50301. Available at: https://scipublication.com/index.php/JACS/article/view/142
15. Zhang, G. (2025). Cloud computing convergence: Integrating computer applications and information management for enhanced efficiency. *Frontiers in Big Data, 8*, 1508087. https://doi.org/10.3389/fdata.2025.1508087. Available at: https://www.frontiersin.org/journals/big-data/articles/10.3389/fdata.2025.1508087/full
16. Pandhare, H. V. (2025). Future of software test automation using AI/ML. *International Journal of Engineering and Computer Science, 13*(5), 27159–27182. https://doi.org/10.18535/ijecs/v13i5.5139. Available at: https://www.researchgate.net/profile/Harshad-Vijay-Pandhare-2/publication/391806293_Future_of_Software_Test_Automation_Using_AIML/links/68277cda026fee1034f8a449/Future-of-Software-Test-Automation-Using-AI-ML.pdf
17. Mohapatra, P. S. (2025). Artificial intelligence-driven test case generation in software development. In *Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle* (Vol. 4, p. 38). Springer Nature. https://doi.org/10.1007/978-3-030-45637-0_4. Available at: https://books.google.com/books?hl=en&lr=&id=u5h1EQAAQBAJ&oi=fnd&pg=PA38&dq=Mohapatra,+P.+S.+(2025).+Artificial+intelligence-driven+test+case+generation+in+software+development.+In+Intelligent+Assurance:+Artificial+Intelligence-Powered+Software+Testing+in+the+Modern+Development+Lifecycle+(Vol.+4,+p.+38).+Springer+Nature.+https://doi.org/10.1007/978-3-030-45637-0_4+&ots=h5hkxs49G-&sig=PnkY_e_u1oO1HoBLHzZtK8ArQOs
18. Cruz, L., Fernandes, J. P., Kirkeby, M. H., Martínez-Fernández, S., Sallou, J., Anwar, H., & Yamshchikov, I. P. (2025). Greening AI-enabled systems with software engineering: A research agenda for environmentally sustainable AI practices. *ACM SIGSOFT Software Engineering Notes, 50*(3), 14–23. https://doi.org/10.1145/3743095.3743099. Available at: https://dl.acm.org/doi/abs/10.1145/3743095.3743099

19. Khan, H. U., Ali, F., & Nazir, S. (2024). Systematic analysis of software development in cloud computing perceptions. *Journal of Software: Evolution and Process, 36*(2), e2485. https://doi.org/10.1002/smr.2485. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.2485
20. Atadoga, A., Umoga, U. J., Lottu, O. A., & Sodiya, E. O. (2024). Tools, techniques, and trends in sustainable software engineering: A critical review of current practices and future directions. *World Journal of Advanced Engineering Technology and Sciences, 11*(1), 231–239. https://doi.org/10.30574/wjaets.2024.11.1.0051. Available at: https://pdfs.semanticscholar.org/b695/a6e8b77c21122306344de7e8c037eb764e33.pdf
21. Dwivedi, K., Haghparast, M., & Mikkonen, T. (2024). Quantum software engineering and quantum software development lifecycle: A survey. *Cluster Computing, 27*(6), 7127–7145. https://doi.org/10.1007/s10586-024-04362-1. Available at: https://link.springer.com/article/10.1007/s10586-024-04362-1
22. Merseedi, K. J., & Zeebaree, S. R. (2024). The cloud architectures for distributed multi-cloud computing: A review of hybrid and federated cloud environments. *The Indonesian Journal of Computer Science, 13*(2), 1644–1673. https://doi.org/10.33022/ijcs.v13i2.3811. Available at: http://ijcs.net/ijcs/index.php/ijcs/article/view/3811
23. Srinivas, P., Husain, F., Parayil, A., Choure, A., Bansal, C., & Rajmohan, S. (2024, April). Intelligent monitoring framework for cloud services: A data-driven approach. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice* (pp. 381–391). ACM. https://doi.org/10.1145/3639477.3639753. Available at: https://dl.acm.org/doi/abs/10.1145/3639477.3639753
24. Kirti, M., Maurya, A. K., & Yadav, R. S. (2024). Fault-tolerance approaches for distributed and cloud computing environments: A systematic review, taxonomy and future directions. *Concurrency and Computation: Practice and Experience, 36*(13), e8081. https://doi.org/10.1002/cpe.8081. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.8081
25.