Archives available at journals.mriindia.com

International Journal on Advanced Computer Theory and Engineering

ISSN: 2319-2526

Volume 15 Issue 01, 2026

Automated Deployment of Full-Stack Applications Using a CI/CD Pipeline in a DevOps Environment

[1]Prince Ganga Rai, [2]Ganesh Mahesh Tiwari, [3]Atul Jaisingh Yadav, [4]Aditya Singh Chauhan, [5]Mr. Aakash Gojare

[1,2,3,4] Department of Computer EngineeringShree L.R Tiwari College of Engineering Mumbai, India

[5]Assistant Professor, Department of Computer Engineering, Shree L.R Tiwari College of Engineering, Mumbai, India

Email: [1]prince.g.rai@slrtce.in, [2]ganesh.m.tiwari@slrtce.in, [3]atul.j.yadav@slrtce.in, [4]aditya.s.chauhan@slrtce.in

Peer Review Information

Submission: 19 March 2026

Revision: 08 April 2026

Acceptance: 24 April 2026

Keywords

CI/CD Pipeline, DevOps,

Abstract

Modern software development requires rapid and reliable deployment mechanisms to maintain system stability and ensure faster delivery of new features. Traditional deployment methods often rely on manual configuration processes that are time-consuming and prone to human error. DevOps practices address these challenges by introducing automation across the software development lifecycle using Continuous Integration and Continuous Deployment (CI/CD) pipelines.

This paper presents the design and implementation of an ***automated CI/CD pipeline*** for deploying a Full-Stack application using modern DevOps tools and cloud infrastructure. The proposed framework integrates GitHub for source code management, Jenkins for automated pipeline execution, Maven for build automation, SonarQube for code quality analysis, and Trivy for vulnerability scanning. Docker is used for containerization, while Kubernetes orchestrates container deployment within a cloud infrastructure hosted on Amazon Web Services (AWS).

The automated pipeline streamlines the software delivery workflow by integrating code validation, containerization, and cloud deployment within a unified system. By reducing manual intervention and ensuring consistent deployment environments, the proposed system improves reliability, efficiency, and scalability in modern application deployment.

Introduction

Modern software systems are continuously evolving, requiring frequent updates, feature additions, and bug fixes. In traditional software development environments, deploying applications often involves several manual steps such as configuration management, environment setup, and server deployment. These manual procedures increase the possibility of configuration errors and slow down the overall software delivery process.

To overcome these challenges, organizations have adopted DevOps methodologies that encourage collaboration between development and operations teams while promoting automation across the software lifecycle. One of the most important technologies supporting DevOps practices is the Continuous Integration and Continuous Deployment (CI/CD) pipeline. CI/CD pipelines automate the process of building, testing, and deploying software applications, allowing development teams to deliver updates more efficiently and reliably. Several studies have highlighted the effectiveness of CI/CD in improving deployment frequency and reducing integration issues [8], [16].

Continuous Integration ensures that code changes from multiple developers are regularly merged into a shared repository where automated build and testing processes validate the code. Continuous Deployment extends this process by automatically deploying validated builds to production environments without manual intervention. Research by Hyun et al. [1] and Amgothu [3] demonstrates how Jenkins-based automation significantly enhances deployment efficiency in containerized environments.

In addition to automation, containerization technologies such as Docker have significantly improved application deployment practices. Docker allows applications and their dependencies to be packaged into lightweight containers that can run consistently across different computing environments. Studies on container technologies and deployment automation highlight their effectiveness in improving portability and scalability [5], [15]. Kubernetes further enhances this process by providing automated container orchestration, enabling scalable and reliable application deployment in cloud environments [11].

Security and data integrity are also critical aspects of modern deployment pipelines. Techniques for secure data transmission and encryption [2], along with pipeline integrity preservation mechanisms [7], play a vital role in ensuring safe deployments. Additionally, integrating vulnerability scanning and code quality analysis tools helps detect potential risks early in the development lifecycle.

Cloud computing platforms such as Amazon Web Services (AWS) provide scalable infrastructure for implementing CI/CD pipelines. Early work on cloud platforms [20], [21] and recent advancements in cross-cloud DevOps frameworks [38] demonstrate how cloud environments support automated deployment, scalability, and high availability.

Furthermore, advancements in machine learning, data processing, and intelligent systems [23], [25], [30], [32] have influenced modern DevOps practices by enabling predictive analysis, anomaly detection, and smarter automation in pipelines. These interdisciplinary contributions highlight the growing integration of intelligent systems within software engineering processes.

This research focuses on implementing an automated CI/CD pipeline for deploying a Full-Stack application using widely adopted DevOps tools and cloud infrastructure. The system integrates Jenkins for automation, Maven for build management, SonarQube for code quality analysis, Trivy for security scanning, Docker for containerization, and Kubernetes for orchestration. The deployment infrastructure is hosted on Amazon Web Services (AWS) using EC2 instances.

By automating the deployment workflow, the proposed system reduces manual errors, improves deployment consistency, and accelerates software delivery cycles.

Research Contributions

The primary contributions of this research are summarized as follows:

1. Design and implementation of an automated CI/CD pipeline for deploying a Full-Stack application using DevOps tools.
2. Integration of multiple DevOps technologies including Jenkins, Maven, SonarQube, Trivy, Docker, Nexus, and Kubernetes.
3. Implementation of containerized application deployment to ensure consistent execution across development and production environments.
4. Deployment of the automated pipeline within a Kubernetes cluster hosted on AWS cloud infrastructure.
5. Automation of code quality validation and security scanning within the CI/CD pipeline.

Literature Review

Recent advancements in software engineering emphasize the importance of automation in improving software delivery processes. Continuous Integration and Continuous Deployment pipelines have become fundamental components of modern DevOps practices, enabling organizations to automate the integration, testing, and deployment of software applications. A systematic review by Shahin et al. [8] highlights the widespread adoption of CI/CD practices and their impact on reducing development cycle time and improving software quality. Additionally, Donca et al. [16] proposed pipeline generation methods that further streamline CI/CD processes in agile environments.

Automation tools such as Jenkins have been widely studied for their role in enhancing deployment efficiency. Hyun et al. [1] demonstrated that Jenkins-based automation significantly improves container-based deployment performance, while Amgothu [3] presented an end-to-end CI/CD pipeline integrating Jenkins and Kubernetes for efficient software delivery. Similarly, cross-cloud DevOps frameworks [38] extend these capabilities by enabling deployment across multiple cloud platforms.

Containerization has emerged as a key technology for ensuring environment consistency in application deployment. Kaiser et al. [5] provided a comprehensive survey on container technologies, highlighting their advantages in portability and performance. Furthermore, Bermejo et al. [15] explored automated Docker image deployment across network environments, emphasizing the role of automation in large-scale systems. Kubernetes-based deployment frameworks [11] further enhance container orchestration by enabling scalability, fault tolerance, and automated recovery mechanisms.

Software architecture also plays a crucial role in deployment strategies. Blinowski et al. [12] compared monolithic and microservice architectures, demonstrating that microservices combined with containerization offer better scalability and flexibility in CI/CD environments.

Security and integrity within CI/CD pipelines are critical concerns. Muñoz et al. [7] introduced methods for preserving project integrity within CI/CD pipelines, while Kaul et al. [2] proposed secure encryption techniques for data transmission. Additional studies on authentication and security mechanisms [9], [26] further reinforce the importance of integrating security measures within automated pipelines.

Beyond DevOps-specific research, several studies in machine learning, image processing, and intelligent systems [10], [13], [23], [25], [30], [31] contribute to the broader understanding of automation, data processing, and optimization techniques that can be leveraged in CI/CD workflows. For instance, data preprocessing and augmentation techniques [23] and optimization methods [25] can improve testing and validation processes in automated pipelines.

Research in IoT and cloud-based systems [22], [24], [33], [34] also demonstrates the growing reliance on automated deployment and monitoring frameworks, which align with DevOps practices. Similarly, advancements in AI-driven systems such as brain activity mapping [6], healthcare applications [28], [29] and predictive analytics [32], [35], [36] highlight the increasing demand for scalable and automated deployment solutions.

Additionally, earlier works on tracking systems, biometric recognition, and intelligent surveillance [4], [14], [17], [18], [19] provide foundational insights into real-time system deployment and automation challenges. Emerging technologies such as nano-electronic devices with machine learning capabilities [27] and intelligent recommendation systems [37] further emphasize the importance of efficient deployment pipelines to support complex applications.

Cloud computing continues to be a backbone of modern DevOps practices. Early work on cloud platforms such as Windows Azure [20] and recent innovations in cross-cloud deployment [38] demonstrate how cloud environments enable scalable, reliable, and automated CI/CD implementations.

Overall, the literature indicates that integrating CI/CD pipelines with containerization, cloud computing, and security mechanisms significantly enhances software deployment efficiency, scalability, and reliability. The combination of these technologies forms the foundation for modern DevOps environments and supports the automated deployment of full-stack applications.

System Architecture

The proposed system architecture integrates multiple DevOps tools to automate the software deployment lifecycle. The architecture begins with developers committing code to a GitHub repository, which serves as the central source code management system.

Whenever code changes are pushed to the repository, a webhook triggers the Jenkins CI/CD pipeline. Jenkins retrieves the latest source code and initiates the automated build process using Maven. During this stage, the application code is compiled and necessary dependencies are resolved.

Following the build stage, code quality analysis is performed using SonarQube. This process evaluates the maintainability of the code and detects potential bugs or vulnerabilities. In addition, Trivy performs security scanning of container images to identify known vulnerabilities before deployment.

After successful validation, Docker packages the application into container images. These images ensure consistent application environments across development, testing, and production systems.

Finally, Kubernetes orchestrates the deployment of containerized applications within a cluster hosted on AWS EC2 infrastructure.

The architecture integrates source code management, build automation, security validation, containerization, and cloud deployment within a unified system deployment within a unified system.

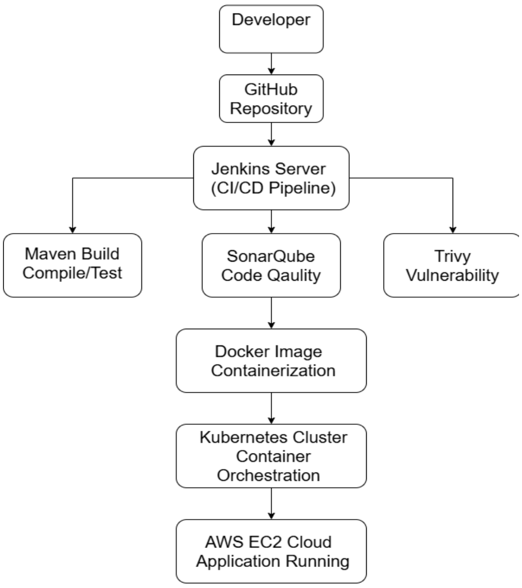

Fig 1: Architecture of the automated CI/CD pipeline integrating DevOps tools for containerized application deployment.

The architecture of the proposed automated deployment system is illustrated in Fig. 1. Source code changes pushed to the GitHub repository trigger the Jenkins CI/CD pipeline. The pipeline performs build automation using Maven, static code analysis using SonarQube, and vulnerability scanning through Trivy. After successful validation, Docker containerizes the application and Kubernetes orchestrates the deployment on AWS cloud infrastructure.

Proposed Methodology

The implementation of the CI/CD pipeline follows several sequential stages that automate the software delivery lifecycle.

1. **Source Code Management**

Developers maintain the application source code in a GitHub repository. Each code commit triggers the automated CI/CD pipeline.

2. **Continuous Integration**

Jenkins executes the pipeline script to automatically retrieve code updates and initiate the build process using Maven.

3. **Code Quality Analysis**

SonarQube analyzes the application source code to detect bugs, security vulnerabilities, and maintainability issues.

4. **Security Scanning**

Trivy scans container images to identify potential vulnerabilities before deployment.

5. **Containerization**

Docker packages the application and its dependencies into container images.

6. **Deployment**

Kubernetes containerized deploys the application within a cluster hosted on AWS EC2

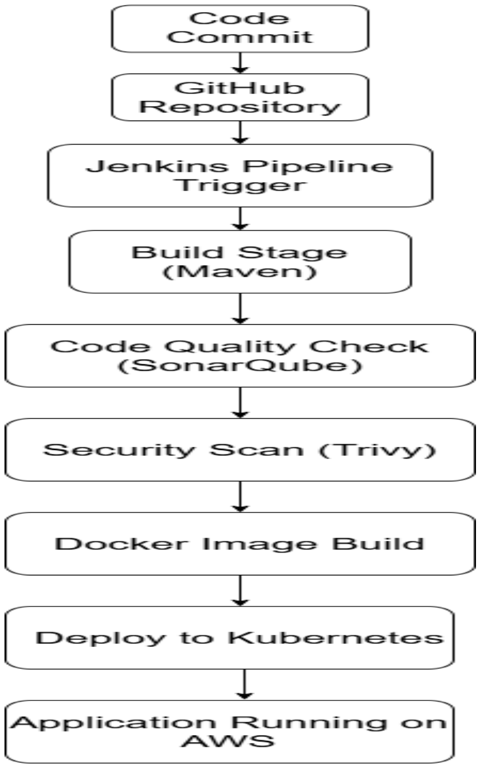

Fig 2: CI/CD pipeline workflow showing automated build, testing, and deployment stages.

The pipeline automates build validation, container creation, and cloud deployment processes, ensuring reliable application delivery.

Implementation

The implementation of the automated CI/CD pipeline integrates multiple DevOps tools to create a fully functional deployment workflow.

When developers push code changes to the GitHub repository, Jenkins automatically detects the update and triggers the pipeline. Jenkins retrieves the source code and initiates the build process using Maven. After successful compilation, SonarQube performs static code analysis to evaluate code quality and detect potential issues.

Following code validation, Docker creates a container image of the application. The container image includes all required dependencies, ensuring consistent execution across different environments. Kubernetes manages the deployment of containerized applications within a cluster environment. The platform automatically schedules containers, monitors their health, and scales application instances when necessary.

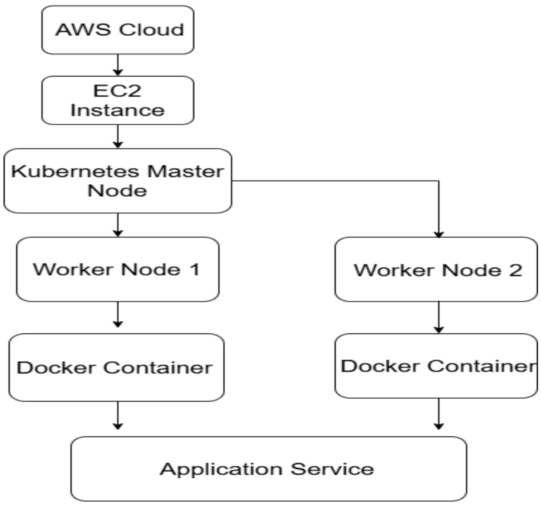

Fig 3: Kubernetes-based deployment of containerized applications on AWS infrastructure.

Hosting the deployment environment on AWS EC2 provides scalable cloud resources for running the Kubernetes cluster.

Results and Discussion

The implementation of the automated CI/CD pipeline demonstrates the effectiveness of DevOps automation in simplifying application deployment processes. By integrating build automation, security scanning, containerization, and cloud deployment within a unified workflow, the system eliminates many manual tasks associated with traditional deployment methods.

Containerization using Docker ensures consistent application environments across development and production stages. Kubernetes orchestration further enhances deployment reliability by managing container scaling and fault tolerance.

Additionally, integrating security and quality analysis tools within the pipeline allows developers to detect potential issues early in the development lifecycle. This proactive approach improves overall software quality and reduces deployment risks.

The results indicate that automated CI/CD pipelines can significantly improve deployment efficiency and maintain consistent software delivery processes.

Conclusion

This research presented the design and implementation of an automated CI/CD pipeline for deploying a Full-Stack application using DevOps tools and cloud infrastructure. The proposed system integrates Jenkins, Maven, SonarQube, Trivy, Docker, Kubernetes, and AWS to automate the entire software deployment lifecycle. By eliminating manual deployment steps and integrating quality and security validation within the pipeline, the system improves reliability, scalability, and efficiency in application delivery.

Future work may include integrating monitoring tools, implementing automated rollback mechanisms, and exploring AI-based optimization techniques to further enhance CI/CD pipeline performa

References

G. Hyun, J. Kim, and S. Lee, "The Impact of an Automation System Built with Jenkins on the Efficiency of Container-Based Deployment," *Sensors*, vol. 24, no. 18, 2024.

V. Kaul, B. Nemade, and V. Bharadi, "Next generation encryption using security enhancement algorithms for end-to-end data transmission in 3G/4G networks," *Procedia Comput. Sci.*, vol. 79, pp. 1051–1059, 2016, doi: 10.1016/j.procs.2016.03.133.

S. Amgothu, "An End-to-End CI/CD Pipeline Solution Using Jenkins and Kubernetes," *International Journal of Science and Research*, vol. 13, no. 8, pp. 1576–1578, 2024.

B. Nemade, "Automatic traffic surveillance using video tracking," *Procedia Comput. Sci.*, vol. 79, pp. 402–409, 2016, doi: 10.1016/j.procs.2016.03.052.

K. Kaiser, M. S. Haq, A. S. Tosun, and T. Korkmaz, "Container Technologies for ARM Architecture: A Comprehensive Survey of the State-of-the-Art," *IEEE Access*, vol. 10, pp. 84853–84881, 2022.

B. Nemade *et al.*, "Adaptive transformer-based framework for brain activity mapping," *Front. Hum. Neurosci.*, vol. 19, p. 1551168, 2025.

A. Muñoz, A. Ojdowska, and A. Przybyłek, "P2ISE: Preserving Project Integrity in CI/CD Pipelines," *Information*, vol. 12, no. 9, p. 357, 2021.

M. Shahin, M. A. Babar, and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review," *IEEE Access*, vol. 5, pp. 3909–3943, 2017.

K. Patra *et al.*, "Cued-click point graphical password using circular tolerance to increase password space," *Procedia Comput. Sci.*, vol. 79, pp. 561–568, 2016, doi: 10.1016/j.procs.2016.03.071.

N. S. T. Sai *et al.*, "Truncated DCT and decomposed DWT-SVD features for image retrieval," *Procedia Comput. Sci.*, vol. 79, pp. 579–588, 2016, doi: 10.1016/j.procs.2016.03.073.

K. Abhishek, D. R. Rao, and K. Subrahmanyam, "Framework to Deploy Containers Using Kubernetes and CI/CD Pipeline," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 4, 2022.

G. Blinowski, A. Ojdowska, and A. Przybyłek, "Monolithic vs. Microservice Architecture: A Performance and Scalability Evaluation," *IEEE Access*, vol. 10, pp. 20357–20374, 2022.

H. B. Kekre *et al.*, "Performance comparison of DCT, FFT, WHT, Kekre's transform and Gabor filter based feature vectors for online signature recognition," *Int. J. Comput. Appl.*, 2011.

B. Nemade and V. A. Bharadi, "Adaptive automatic tracking, learning and detection of real-time objects in video streams," in *Proc. Int. Conf. Confluence*, Noida, India, 2014, pp. 569–575.

G. Bermejo *et al.*, "Automating Docker Image Deployment Across Network Environments," *SoftwareX*, 2025.

I.-C. Donca, O. P. Stan, M. Misăroș, D. Gota, and L. Miclea, "Method for Continuous Integration and Deployment Using a Pipeline Generator for Agile Software Projects," *Sensors*, vol. 22, no. 12, p. 4637, Jun. 2022.

V. A. Bharadi, B. Pandya, and B. Nemade, "Multimodal biometric recognition using iris and fingerprint by texture feature extraction," in *Proc. Int. Conf. Confluence*, Noida, India, 2014, pp. 697–702.

V. A. Bharadi, V. I. Singh, and B. Nemade, "Hybrid wavelets-based feature vector generation for online handwritten signature recognition," in *Proc. Int. Conf. Confluence*, Noida, India, 2014, pp. 561–568.

M. Shirodkar *et al.*, "Automated attendance management system using face recognition," in *Proc. Int. Conf. Emerging Trends Technol.*, 2015.

B. Nemade, S. Moorthy, and O. Kadam, "Cloud computing: Windows Azure platform," in *Proc. Int. Conf. Emerging Trends Technol.*, 2011.

K. Talapatra *et al.*, "Extramedullary haematopoiesis causing spinal cord compression," *Ann. Indian Acad. Neurol.*, vol. 10, no. 2, pp. 115–117, 2007.

B. Nemade and D. Shah, "IoT-based water parameter testing in linear topology," in *Proc. Int. Conf. Confluence*, 2020, pp. 546–551.

K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proc.*, vol. 3, no. 1, pp. 91–99, 2022.

B. Nemade and D. Shah, "An IoT-based efficient water quality prediction system for aquaponics farming," in *Computational Intelligence*, Springer, 2023.

B. Nemade *et al.*, "A comprehensive review: SMOTE-based oversampling methods," *Int. J. Intell. Syst. Appl. Eng.*, 2023.

B. Nemade *et al.*, "Enhancing information security using moth-flame optimization," *ICTACT J. Commun. Technol.*, 2023.

B. C. Surve, B. Nemade, and V. Kaul, "Nano-electronic devices with machine learning capabilities," *ICTACT J. Microelectron.*, 2023.

S. S. Alegavi *et al.*, "Revolutionizing healthcare using wearable biomedical devices," *Int. J. Recent Innov. Trends Comput. Commun.*, 2023.

B. Nemade *et al.*, "Amphibious trash collector system," 2023.

R. Mishra *et al.*, "Improved inductive learning approach-5 (IILA-5)," *Int. J. Intell. Syst. Appl. Eng.*, 2023.

V. A. Bharadi *et al.*, "Using CNN-LSTMs and transformer RNNs for COVID-19 prediction," 2023.

B. Nemade *et al.*, "Efficient GDP growth forecasting using modified LSTM," 2024.
B. Marakarkandy *et al.*, "Enhancing multi-channel consumer behavior analysis using Apriori algorithm," 2024.

B. Nemade *et al.*, "Enhancing connectivity using embedded IoT devices," 2024.

G. Khandelwal *et al.*, "Enhancing water quality prediction accuracy for aquaponics farming," 2024.

V. Kulkarni *et al.*, "ADHD detection using convolutional neural networks," *Front. Psychiatry*, vol. 15, 2024.

B. Nemade *et al.*, "Graph attention dialog network-based drug recommendation model," *IEEE Trans. Consum. Electron.*, vol. 71, no. 2, 2025.

M. D. G. Manohara and K. S. Jyothi, "Cross-Cloud DevOps Framework for Automated CI/CD Deployment," *Iconic Research and Engineering Journals*, vol. 9, no. 3, pp. 1374–1378, 2025.