International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4165 - 4170
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0538

Deployio – AI-Powered Devops Automation Platform

Dr R Guru[1], Dr Anusuya M A[2], Chaithanya S[3], Vasudev D M[4], Varsha M Vandaganoor[5], Pavan Gowda N[6]
[1,2]Professor, Dept. of CSE, JSS Science and Technology University, Mysuru, Karnataka, India
[3,4,5,6]UG Student, Dept. of CSE, JSS Science and Technology University, Mysuru, Karnataka, India
Emails: *guruirg@sjce.ac.in[1], anusuya_ma@sjce.ac.in[2], chaithanya9ig@gmail.com[3], vasudeepu2815@gmail.com[4], varsham4440@gmail.com[5], pavangowda16102004@gmail.com[6]*

Abstract
In the era of cloud-native applications and continuous delivery, DevOps automation has become a crucial part of the development process. However, the process of establishing deployment workflows, which encompasses environment setup, dependency management, CI/CD pipeline setup, and cloud provisioning, can be complex and prone to errors, especially for students and early-stage developers. In this paper, we introduce Deployio, a DevOps automation AI-powered system that aims to simplify the deployment of applications by providing an integrated and transparent approach. It automatically scans GitHub repositories to detect the technology stack, creates CI/CD pipelines, builds container images, and deploys applications to the cloud with minimal configuration. Our initial evaluation suggests that Deployio reduces configuration complexity, standardizes deployment workflows, and detects configuration problems early. The findings indicate that Deployio offers a good balance between automation and transparency, making it suitable for both deployment and educational purposes.

Keywords— *AI-assisted systems, CI/CD, cloud deployment, DevOps automation, Docker.*

1. Introduction
DevOps has become a critical aspect of modern software development, allowing for more efficient, reliable, and scalable deployment of applications (Gartner, 2017; Chen et al., 2023). Practices like Continuous Integration and Continuous Deployment (CI/CD), containerization, and cloud hosting are widely used to ensure that applications are production-ready and to streamline the software development lifecycle (Kumar & Patel, 2023). However, the adoption of these practices adds complexity to the development process, especially for early - stage developers who may lack experience with deployment and cloud technologies. In many educational and small-scale development settings, applications are typically only run locally due to the complexities involved in setting up Docker environments, configuring CI/CD workflows, deploying to cloud-based resources, and monitoring applications. Current deployment solutions either fully abstract these processes – diminishing visibility and learning opportunities - or are highly advanced, making them difficult to adopt and manage effectively (Harris, 2024). This suggests automation of DevOps processes with transparency and user control. To overcome these limitations, this paper proposes Deployio, a DevOps automation system powered by AI that makes the software deployment process faster and seamless with an integrated and transparent workflow. The system enables a single click to deploy from GitHub, with scanning the code, setting up CI/CD, building container images, and deploying to the cloud. The key novelty of this work is the design of a framework that offers automation and visibility to allow users to monitor and have complete control over the deployment, without complexity and manual work interventions. The system aims to democratize DevOps without changing the traditional deployment practices. This paper is structured as follows. Section II discusses related work. Section III presents the problem and our approach. Section IV discusses the experimentation results.

2. Related Work
Artificial intelligence is a key element in recent studies on DevOps automation, in addressing the growing complexity of software systems. Artificial Intelligence for IT Operations (AIOps) has been developed to solve problems in large systems via automated analysis, decision-making, and operational intelligence (Gartner, 2017). Current

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4165 - 4170
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0538

AIOps techniques are primarily concerned with post-deployment activities such as log monitoring, anomaly detection, and event correlation. These approaches improve system stability but are reactive and offer little help in the early stages of software deployment. The paper AIOps: Real-World Challenges and Research Innovations discusses the need for proactive and comprehensive automation across the software development life cycle, and the issues of scalability, explainability, and process integration (Gartner, 2017). Specifically, the absence of smart automation for deployment configuration and CI/CD pipeline generation is a key challenge. Deployio overcomes this challenge by integrating AI-based repository and pipeline creation in the pre-deployment phase to prevent configuration issues when the deployment occurs. The use of large language models (LLMs) has also influenced research in automated software engineering. Large Language Models for a Survey demonstrates that LLMs can understand the structure of software repositories, write configuration files, and automate engineering tasks (Chen et al., 2023). It also recognises CI/CD scripts and infrastructure configurations as candidates for AI-assisted automation due to their structured format. However, existing LLM-based solutions workflows. The Deployio platform extends this research by integrating AI-assisted analysis and synthesis of configurations. Instead of producing configuration snippets, we synthesise contextual, automatically executed, and monitored CI/CD pipelines and container configurations are primarily AI assistants, and not part of full deployment. This enables the configurations to be verified during deployment, addressing problems found in earlier work (Chen et al., 2023). Existing CI/CD solutions like Jenkins and GitHub Actions provide automation, but require users to explicitly manage pipelines, which can be difficult and error-prone for beginners. Alternatively, platform-as-a-service (PaaS) providers such as Vercel, Netlify, Render, and Railway simplify deployment by hiding infrastructure and processes. This makes things easier, but less transparent and less educational. Deployio aims to solve this problem by offering automation and transparency, by exposing pipeline configurations, deployment logs, and application behaviour, as well as a few manual steps. In short, current solutions focus on either automation or transparency - but not both. This suggests a need for a solution that combines automation and transparency to enable fast deployment without sacrificing user knowledge. Deployio addresses this by combining AI automation and DevOps to enable an easier deployment with transparency.

3. Problem and Proposed Solution

3.1. Problem Description

Although DevOps has become mainstream, setting up processes for deployment still remains a tedious and error-prone task, particularly for students. Managing the deployment environment, dependencies, CI/CD pipelines, and cloud infrastructure involves complex workflows requiring expert skills. This affects the deployment of many applications in small projects in the local environment [1-5]. Today's solutions either provide automation and customisation, but at the cost of configuration, or they simplify the deployment process by hiding the complexity, which reduces the transparency and learning opportunities. Therefore, there is a need for an automated system for deployment operations that is flexible.

3.2. System Architecture

Deployio uses a microservices architecture to achieve scalability, flexibility, and maintainability. The system is structured into three layers: the client layer, the backend layer, and the DevOps layer. The client layer, built with React, provides a web interface for authentication, repository submission, and deployment status. It receives build and deployment logs in real-time through WebSocket. The server layer, implemented using Node.js and Express, acts as the orchestrator, handling OAuth authentication, repository access, data storage (MongoDB), and communication with other components. It also includes an AI analysis service, developed with Python and FastAPI, to analyse the repository structure, configuration files, and dependencies to identify the technology stack of the application and to determine the deployment requirements. The DevOps layer is responsible for execution and deployment, including CI/CD pipeline generation, containerization (Docker), and deployment on AWS EC2. This allows an automated

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4165 - 4170
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0538

and traceable deployment process, as shown in Figure 1.

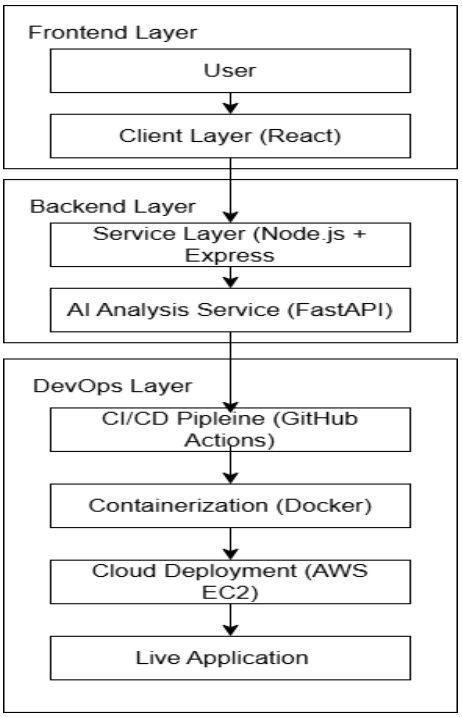

Figure 1 System Architecture of Deployio platform

3.3. Methodology

The Deployio process is fully automated and begins with access to the repository and concludes with application deployment into the production environment, see Figure 2.

3.3.1. Repository Validation

First, the user logs in using GitHub OAuth. If the user is logged in, they should provide a URL to the repository. The system validates that the repository is accessible, has the right permissions, and is built in a way that is safe for automated deployment. If it passes, the repository is cloned into a sandboxed environment so that it does not interfere with the deployment, and the integrity of the project is preserved.

3.3.2. AI-Based Technology Analysis

The AI-based analysis service scans the repository, using configuration files, dependency files, and folder structures. It detects the tech stack, runtime environment, and build environment. This auto-detection avoids manual configuration and configuration errors.

3.3.3. CI/CD Pipeline Generation

The Deployio solution ensures consistency by creating Docker configuration and creating images with the application code, dependencies, and environment [6-10]. The deployment manager subsequently spins up EC2 instances, sets up networking, ports, and services. It then gives an endpoint to the application.

3.3.4. Containerization and Deployment

The Deployio solution guarantees consistency by generating Docker configuration and building images containing the application code, dependencies, and environment. The deployment manager then runs containers on EC2 instances, configuring networking, ports, and activating services. It then provides an endpoint for the application.

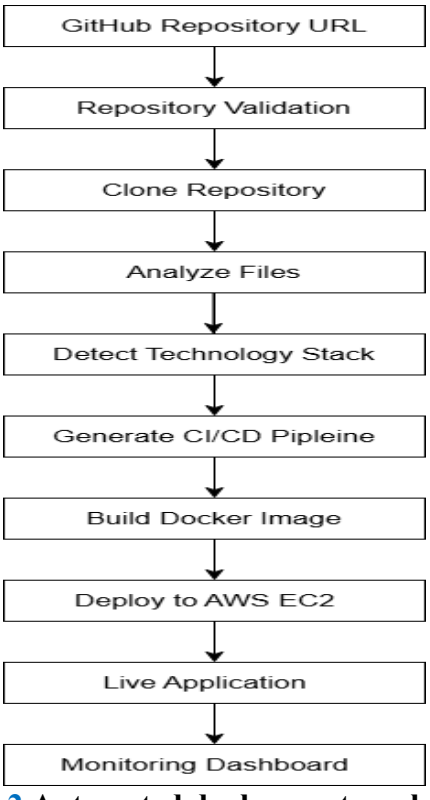

Figure 2 Automated deployment workflow of Deployio

3.3.5. Monitoring and Feedback

Deployio uses the client dashboard to monitor deployment. This contains build, deployment, and even runtime logs through a WebSocket connection.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4165 - 4170
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0538

This gives prompt feedback and enhances transparency, enabling problems with deployments to be identified and troubleshooted early.

3.4. Implementation Details

The Deployio implementation uses a combination of technologies to automate, scale, and offer reliability. The front-end, which is built on React, offers an interactive web interface to build, deploy, manage, and monitor workflows. WebSocket is employed to send real-time messages to keep users informed about deployments without the need to refresh the page. The server is a Node.js and Express-based server that offers RESTful APIs to authenticate, analyze the repository, and manage the workflow. Authentication is done using OAuth, and data is stored using MongoDB. The AI analysis service is a FastAPI-based service that creates deployment templates based on the repository and dependencies. This removes configuration and consistency problems. Docker containerization is employed to provide consistency. CI/CD is automated using GitHub Actions, and cloud hosting is done using AWS EC2.

4. Experimental Results and Discussion

4.1. Experimental Setup

The performance of Deployio was evaluated in a qualitative comparative study with conventional DevOps deployment practices and popular cloud deployment services. The evaluation included the assessment of factors like the complexity of deployment, the effectiveness of automation, visibility, and reliability, which are significant in the research of DevOps and AIOps (Gartner, 2017). The tests involved the deployment of a set of sample web applications on GitHub with Deployio and the comparison of this with the manual deployment procedures and platform-as-a-service (PaaS) platforms. The evaluation is based on workflow management, usability, and reliability as opposed to numbers and values.

4.2. Performance and Results

Typical deployment processes require developers to set up CI/CD pipelines, create Dockerfiles, runtime environments, and deployment scripts. This adds to the complexity of the system and opens the door to configuration mistakes, especially for users without extensive DevOps knowledge. Deployio simplifies this by automating critical steps, including repository scanning, CI/CD pipeline generation, and containerization. Reducing manual work, the system prevents configuration errors and speeds up deployment. The streamlined process means applications can be deployed in a repeatable and predictable way for various projects. A key insight is the early identification of configuration problems. Deployio detects conflicts and errors related to dependencies and configuration during analysis and pipeline stages, not at runtime. This early detection enhances the reliability and reduces the debugging burden, in line with the AIOps approach (Gartner, 2017).

4.3. Comparative Analysis

A comparison of deployment strategies proves the success of Deployio: Manual Deployment: It requires a lot of configuration, including the environment setup, pipeline construction, and the deployment scripts. It is inaccurate and time-consuming. Platform-as-a-Service (Vercel, Netlify, Render, Railway): Higher-level infrastructure and processes to deploy more easily. However, they do not have much transparency in their internal processes, which restricts learning opportunities. Deployio: It is an automation tool that offers visibility by automatically creating CI/CD pipelines, containerizing applications, and deploying them and exposes CI/CD pipeline configurations, logs, and stages. This guarantees effective deployment and transparency of DevOps processes. This example underscores the fact that Deployio balances between user-friendliness and observability, which many solutions lack.

4.4. Discussion

Our evaluation shows that Deployio enhances deployment productivity by reducing manual effort and automating involved processes. Its AI-powered analysis of the repository and automatic creation of deployment pipelines ensure consistent and optimal deployments. An important aspect of Deployio is its transparency. It provides visibility into how the deployments are configured, executed and run. This position makes it ideal for use in education, where learning to deploy is an important element of the learning process. Also, it demonstrates the use of AI-driven automation in DevOps. Deployio addresses some of the shortcomings of previous research on

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4165 - 4170
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0538

stand-alone AI-powered tools (Chen et al., 2023) by performing configuration validations.

4.5. Limitations and Future Work

Deployio has limitations. The current system has a limited variety of technology stacks and templates of CI/CD pipelines and containerization. This increases uniformity but may restrict flexibility to more complicated or customized applications. It is also restricted to one cloud platform, which restricts interoperability with other cloud providers. It also does not have important AIOps capabilities like predictive failure, anomaly detection, and recovery. Future efforts will focus on supporting a wider range of technology stacks, enhanced support of AI-based configuration generation using sophisticated models, and multi-cloud deployment [11-15]. Lastly, the addition of smart monitoring, anomaly detection, and self-healing will also increase reliability and bring Deployio to the state-of-the-art AIOps systems (Gartner, 2017; Chen et al., 2023) shown in Table 1.

Table 1 Comparison with other existing platforms

Feature	Deployio	Vercel	Netlify	Render	Railway
Full-stack support	MERN and multi-stack support	Frontend-focused	Frontend-focused	Full-stack support	Full-stack support
CI/CD automation	Automatically generated pipelines	Limited automation	Limited automation	GitHub Actions integration	Basic automation
Configuration effort	Low (automated)	Very low	Very low	Medium	Medium
Transparency	High (logs and pipeline visibility)	Low	Low	Medium	Medium
Educational value	High (learning-oriented platform)	Low	Low	Low	Low

Conclusion

In this paper, Deployio, an AI-based DevOps automation platform that simplifies the process of deploying applications with an integrated and transparent process, was introduced [16-19]. The platform combines automated code analysis, CI/CD pipeline generation, containerization, and cloud deployment to simplify the DevOps processes. This simplifies deployment practices to students, early-stage developers, and small teams. Unlike existing tools, which are either too configurable or too abstract, Deployio provides a balance between automation and visibility. The tool enables early detection of configuration problems and visibility of the deployment process through AI analysis during the pre-deployment stage of the deployment process. The results indicate that Deployio enhances the speed of deployment, repeatability, and learning of DevOps practices. Overall, the system illustrates how AI-based automation can enhance the viability, effectiveness, and stability of software implementation in the current development processes.

References

[1]. Chen, L., Wang, H., & Zhang, K. (2023). Machine learning-driven incident management in cloud-native environments. IEEE Transactions on Network and Service Management, 20(3), 1127–1140.

[2]. Chen, M., Tworek, J., Jun, H., et al. (2021). Evaluating large language models trained on

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4165 - 4170
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0538

code. arXiv. https://arxiv.org/abs/2107.03374

[3]. Cloud Native Computing Foundation. (2023). Cloud native landscape: Annual report 2023. CNCF Technical Report.

[4]. Gartner, Inc. (2017). Market guide for AIOps platforms. Gartner Research.

[5]. Harris, L. (2024). The evolution of DevOps: Automation and AI integration in modern software delivery. Asian Journal of Computer Science, 3(2), 112–128.

[6]. Kästner, C., Vasilescu, B., & Herbsleb, J. (2023). Teaching DevOps in higher education: Challenges and opportunities. IEEE Software, 40(3), 67–75.

[7]. Kumar, A., & Patel, S. (2023). Predictive maintenance in containerized infrastructure: An AIOps approach. Journal of Cloud Computing, 12(1), 45–62.

[8]. Luz, W. P., Alencar, G., & Nunes, M. A. S. N. (2022). Automated assessment in DevOps education: A systematic literature review. Computers & Education, 186.

[9]. Nijkamp, E., Pang, B., Hayashi, H., et al. (2023). CodeGen: An open large language model for code with multi-turn program synthesis. In International Conference on Learning Representations (ICLR).

[10]. OpenAI. (2023). GPT-4 technical report. arXiv. https://arxiv.org/abs/2303.08774

[11]. Pahl, C., & Jamshidi, P. (2023). Microservices architecture patterns: A comprehensive survey. ACM Computing Surveys, 55(8), 1–35.

[12]. Petersen, K., & Wohlin, C. (2023). Authentic learning in software engineering education: Industry collaboration and real-world projects. ACM Transactions on Computing Education, 23(2), 1–24.

[13]. Zhang, H., & Liu, Q. (2023). Multi-model reliability strategies for production AI systems. ACM Transactions on Intelligent Systems and Technology, 14(3), 1–22.

[14]. Zhang, Y., Liu, J., & Brown, M. (2023). Predictive analytics for software deployment reliability. ACM Transactions on Software Engineering and Methodology, 32(4), 1–28.

[15]. Bernstein, D. (2022). Container security and optimization: Best practices for production deployments. IEEE Software, 39(6), 34–42.

[16]. Ahmad, A., Waseem, M., Liang, P., et al. (2023). Towards human-bot collaborative software development: A systematic literature review. IEEE Transactions on Software Engineering, 49(4), 2291–2307.

[17]. Alenezi, M. (2022). Can artificial intelligence transform DevOps? arXiv. https://arxiv.org/abs/2206.00225

[18]. Vercel Inc. (2023). The frontend cloud: Performance and developer experience at scale. Technical Whitepaper.

[19]. Netlify Inc. (2023). JAMstack architecture: Modern web development best practices. Technical Documentation.