Online-Journals.org

iTDAF IETI Transactions on Data Analysis and Forecasting

iTDAF | eISSN: 2959-0442 | Vol. 3 No. 4 (2025) | OPEN ACCESS

https://doi.org/10.3991/itdaf.v3i4.59213

PAPER

TaskFlow: A DevOps-Oriented Task Management System for Software Development Environments

Raheel Riaz[1](✉), Fadia Bibi[1], Sidra Tahir[1], Ahthasham Sajid[2], Sabitha Banu[3]

[1]PMAS – Arid Agriculture University, Rawalpindi, Pakistan

[2]Air University, Islamabad, Pakistan

[3]PSGR Krishnammal College for Women, Coimbatore, India

raheelriazraja@gmail.com

ABSTRACT

Effective task management is critical to productivity and organizational success, especially in software development settings where an organization is required to handle a lot of tasks, deadlines, and collaborations. The existing offerings, including Trellis, GitHub, and Microsoft Teams, are fragmented solutions, lacking an easy integration with version control tools, real-time collaborative solutions, and DevOps processes. In order to address these gaps, this paper introduces TaskFlow, a task management system oriented at DevOps, which is intended to ease the creation, assignment, tracking, and deployment of tasks in a single platform. TaskFlow is coded into Dart and Flutter and relies on cross-platform support and adds automated tests (e.g., Selenium) and CI/CD pipelines to ensure quality and efficiency. The system provides end-to-end reporting, version control, and real-time collaboration through modular architecture, ensuring that a gap between the development and operations is bridged. Task Flow is found to be better than existing tools in automation, DevOps integration, and cross-platform compatibility. According to the system evaluation findings, it will lead to a 35–40 percent reduction in manual labor, a 50 percent reduction in post-deployment faults, and improved team coordination. In spite of the fact that the system is better in terms of task automation and integration, the system features added by users include in-built chat and better role management. TaskFlow is a user-friendly and scalable software solution that fits well in the current software teams, collaborating with DevOps to enhance productivity and deliver projects.

KEYWORDS

task management system, software development productivity, real-time collaboration tools

1 INTRODUCTION

Productivity and organizational success are built on good task management that has an impact on personal life as well as professional life. The skill of planning, monitoring, and undertaking tasks in the era of increased amounts and amounts of distractions adds up to the highest importance. Task management is no longer a personal to-do list but is part of a project management in business. In software

Riaz, R., Bibi, F., Tahir, S., Sajid, A., Banu, S. (2025). TaskFlow: A DevOps-Oriented Task Management System for Software Development Environments. *IETI Transactions on Data Analysis and Forecasting (iTDAF)*, 3(4), pp. 41–54. https://doi.org/10.3991/itdaf.v3i4.59213

Article submitted 2025-09-11. Revision uploaded 2025-10-23. Final acceptance 2025-10-25.

development, such as, teams have to deal with vast numbers of tasks, timelines, and partnerships to complete projects in time and within budget. Task management is also required in team-based environments where there is a need to assign tasks and monitor and deliver them in real-time. However, even with the existing plethora of task management solutions, some critical areas of serving the needs and requirements of groups, especially the ability to coordinate tasks to facilitate collaboration, organization, and integration with other significant development, toolset is full of various task management solutions, with Microsoft Teams, Trello, and GitHub being examples of solutions that fulfill various functions in varying workflows. For example, GitHub offers version control and is utilized in software development to track code changes and collaborate on projects [1]. Yet, although it is excellent at source code management, it lacks integrated features for wider task management, such as task prioritization and deadline tracking. Trello is a highly visual task organization tool, a tool that helps the project owner to use boards, lists, and cards to track the project. Advancement, however, is not good regarding live-time cooperation and assimilation. Controlling version: with version control systems [2]. On the other hand, Microsoft Teams bundles collaboration and communications tools but separately must be combined with it to keep track of tasks and is not built in with task tracking in its suite of tools [3].

The need to have a unified, one-stop task management tool that not only has functionalities on how to create tasks, assign them, and even prioritize them but also must also be compatible with other systems required to run the modern-day development processes. Current solutions are isolated, and they are likely to require users to alternate through various and different tools and interfaces that can slow down the work process and introduce inefficiencies. In a bid to address this need, this paper proposes a new Task Management Application developed using Dart and Flutter, whose purpose is to address this limitation. The application can be run on both iOS and Android platforms by exploiting the cross-platform capabilities of Flutter, so it will suit teams based on different platforms well. The DevOps practices are used to make sure that the application is tested deployed, integrated and deployed continuously, offering high quality and performance levels as well as an end-to-end user experience.

Fig. 1. Importance of task management

The paper presents the process of the creation, testing, and implementation of the proposed task management application. It solves the issue of the stand-alone tools of task management and introduces a complete solution including version control real-time collaboration systems (VCS). The second major innovation in this project will be the incorporation of the application with version control applications such as Git, so that software development teams can not only handle the activities of their projects but also see the changes in the codes directly on the task management interface. This incorporation lowers the friction of having the tasks and code changes as two separate items, which enhances the coordination within the team members. Automated reminders are also offered in the application, and these remind the users about task progress and deadlines (see Figure 1).

The research will be user-friendly in nature and will include the process of developing a user-friendly and practical task management interface. The front-end will be built on Flutter, an open-source UI framework that is highly popular and enables the creation of highly responsive applications on both Android and iOS based on a single codebase. The Dart programming language, combined with Flutter, enables the creation of the application quickly and the retention of the responsiveness of the application even when it is being scaled. The development process also includes the other DevOps practices such as continuous integration, automated testing, and automated deployment so that the app is not only stable but also maintainable. The system is to be enhanced and refined over a period, as the system keeps on collecting and feeding the previous ones back, and with the changing needs of the end users, it is modified to suit not only the relevant but also the current needs of the users in the constantly changing technological world.

This study fills the gaps left by current task management systems while also expanding the field of software development through a case implementation of contemporary developmental methodologies such as DevOps, mobile app development for multiple devices, and synchronous multi-user editing. The results of this research will illustrate the possibilities of enhancing task management systems for development teams, especially where collaboration and versioning of the source code are closely integrated.

2 LITERATURE REVIEW

Task management and collaboration tools are central in modern software development and organizational practices. Softwares, such as Trello, GitHub, and Microsoft Teams, has found common usage in recent years based on the advantages that each has in the visualization of projects, versioning, and messaging [7]. In application to large-scale development and DevOps, these tools have several weaknesses. The following evaluates these tools based on primary features, including task management, collaboration, integration, deployment, and analysis, and how the proposed TaskFlow system addresses the weaknesses.

2.1 Trello

Trello is a commonly used Kanban-based software application primarily used for task tracking and simple project management. It has basic functionalities such as task creation, task assignment, notification to users, and a feedback loop through comments and tracking card activity [8]. Its drag-and-drop functionality makes task organization simple, and thus, it is suitable for non-technical project tracking. However, Trello lacks real-time collaboration capabilities such as live editing or

real-time content synchronization. It lacks native version control support as well as automated deployment or DevOps pipeline support. Trello is also lacking in terms of consistency across platforms; while it is accessible through mobile, supported features are much fewer compared to the desktop client. Reporting and analytics support is weak and mostly requires integration with third-party Power-Ups or other reporting software. Absence of in-built communication tools also limits its usage in synchronous team settings [4]. Recent research highlights that disjointed tools increase cognitive load and reduce productivity in agile teams [9].

2.2 GitHub

GitHub is essentially a distributed version control system based on Git widely employed in software development for source code and collaborative development management. It has the capability of managing tasks with issues and project boards to develop and allocate tasks. GitHub is efficient in integrating version control, and it also makes cooperation among developers easier by means of pull requests, comments, and code review. The system is provided with notifications and user feedback mechanisms, which increase its transparency and traceability. GitHub is not necessarily a platform of real-time collaborative use, although it has an excellent base on versioning. Although GitHub Actions does allow for CI/CD, it is difficult to implement and is not inherently part of the user experience for users who are not technical users. It also lacks inherent chat functionality, and although there is some level of cross-platform functionality inherent in it, it is not on all devices. Reporting and analysis are limited to simple repository levels, with no high-level project-level visual analytics [5]. Studies show that integrating task management directly into CI/CD pipelines reduces deployment errors by 30% [10]. GitHub's limited real-time collaboration features hinder synchronous workflows, as noted in distributed team studies [11].

2.3 Microsoft Teams

Microsoft Teams is a comprehensive enterprise communication platform with chat, video calling, file sharing, and work organization through its Planner feature. It enables real-time collaboration, tasking and delegation, cross-platform support, and messaging integration, thus acting as a great organizational alignment tool. Feedback is enabled through chat, forms, and surveys, and analytic feedback through integration with Microsoft Power BI. However, Microsoft Teams does not natively provide version control and is not optimized in a direct way for DevOps pipelines or automatic deployments. Such features are typically obtained through integration with Azure DevOps or any other third-party solution, which complicates things and raises the time of setups. Although Teams is more effective in communications and real-time space, not all developers may be interested in integrated platforms and need to integrate Teams into the build, test, and deploy pipe [6]. A 2022 survey showed that 67 percent of developers favor unified platforms that combine communication, task tracking, and deployment [12].

The proposed TaskFlow will overcome these issues by offering a single platform that has built-in features of synchronous collaboration, versioning, DevOps operations, automated deployment, and platform support. This minimizes the use of supplementary tools and workflow management administration by both the non-development and development teams to develop the system and to manage the workflows in the hybrid teams. TaskFlow is based on cross-platform frameworks such as Flutter, which is reported to

accelerate development by 40% [13] when used in hybrid teams. Automated testing (e.g., Selenium) goes hand in hand with the results that CI/CD automation saves half of manual work [14]. Modular architectures tested in recent studies in the area of DevOps [15] are scaled, whereas the principles of user-centric design enhance adoption [16], [17].

3 METHODOLOGY

The TaskFlow system in question is developed to enhance the process of tasks tracking, assignment, testing, and deploying to a great extent through automated and integrated workflow. The solution assumes a well-organized and systematized pipeline that makes sure of effectiveness in all phases of the job life cycle. The system takes care of the whole life cycle of a duty, creation to deployment hence a well governed systematic procedure. The system provides an end-to-end task management solution based on task management, high-quality standards, and development and operations integration according to the DevOps principles as illustrated in Figure 2.

3.1 Task management lifecycle

The lifecycle of any task will start with the creation of a new task, which will be completed in the create task process. During this step the users will be asked to specify and write about new tasks that will meet the overall objectives of the project. These activities are designed with specifications such as the name, description, deadlines, and priorities and therefore expand the project scope vision. After creating a task, the task is transferred to the Manage Task module, where some of the task attributes are configured. The system also ensures that every task is clearly outlined and sufficiently categorized by organizing the information in a systematic way like priority and deadlines, so that the stakeholders can understand the urgency and need of the task.

The second step in the process is called the Assign Task stage. During this time, the system enables the project managers or the team leaders to assign the task to the team member (assignee) responsible, who is mandated to complete the assignment. The assignee is supposed to perform the mandated development or operation works in regard to the task specifications. The work on the task phase follows the assignment. During this stage the assignee performs the decisive measures required in order to meet the requirements of the task. This stage is quite essential because it shows when the actual development or implementation of the task is done once the assignee has taken the necessary actions, the task is assigned a task completed status. At this stage, the system checks the status of the task to establish whether it is in compliance with the necessary requirements and standards that were established initially. In case the task is considered to be incomplete or it needs more work or modifications, it is re-entered in the Work on Task stage. This is to ensure that the job is mastered and changed as necessary. Otherwise, when the work is considered finished and corresponding to the necessary requirements, it is permitted to pass to the subsequent step.

3.2 Automated testing and deployment pipeline

Once any task is assigned a completion status, it is transferred to the scheduled automated testing stage which is an important stage in the task management system. The tests are operated to authenticate the output of the task to be sure that the work produced is devoid of any flaw and meets the defined quality standards. The automated

testing stage is a quality gate where only fully operational and bug-free tasks are permitted to get through the gate, and the failed ones are immediately identified as such and sent to the work on task stage, where they are debugged or altered accordingly. This feedback mechanism ensures the systemic integrity of the workflow by ensuring that incomplete or incorrect tasks cannot be processed in the workflow. The task, on the other hand, is transferred to the CI/CD pipeline, which becomes the next significant step of the working process. When the task is successfully tested with all the test case requirements and the distributed code integration, the build procedures and the deployment processes operate. Here, the task that has been tested and made ready for production is added to the main repository, followed by running a series of build processes. The CI/CD pipeline makes sure the code is updated and the newly developed features or updates are added to the project in the right manner. Once the pipeline runs successfully, the task is marked to task deployed, i.e., the feature or update has been deployed successfully in the production environment. The well-structured and integrated methodology ensures traceability throughout all stages, making it simple for stakeholders to monitor the status and progress of work. Besides, it establishes a clear responsibility line by giving each team member a specific job to perform, at the earliest occasion of the task to be implemented. In addition, the automated testing and deployment process incorporated into the task management system along with the fact that only thoroughly tested and proven tasks will proceed to the deployment level will guarantee high quality levels. With the integrative combination of task management, automated testing, and deployment, the system can not only promote efficient software development but also promote the culture of DevOps, wherein a single and automated continuous flow of either the development or the operations teams can be achieved. The end-to-end process significantly increases the general response of productivity and quality of the software development process and timely and productively delivers the functionality and updates.

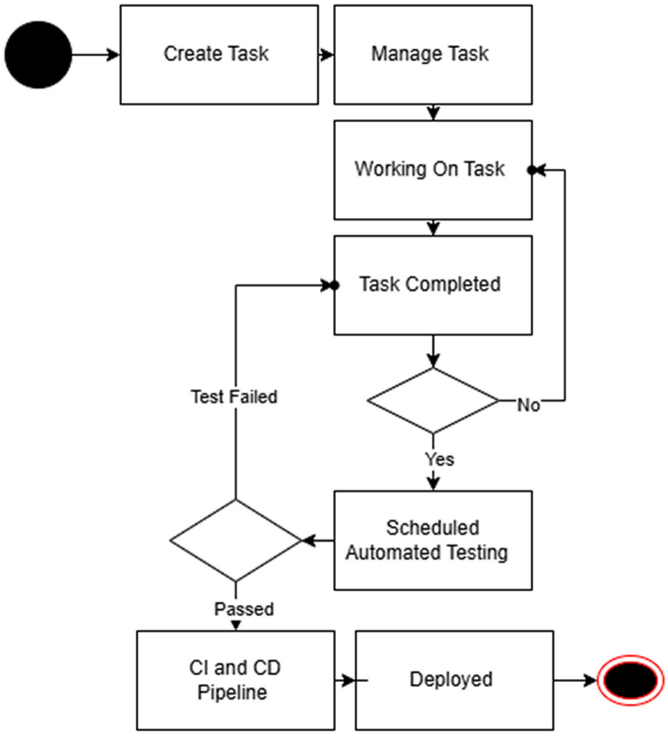

Fig. 2. Proposed methodology

3.3 System implementation

The implementation of the presented TaskFlow framework is systematically oriented at the integration of the distinctive aspects that relate to the task management, automated testing, and deployment into an integrated flow see in Figure 3. This stage solidifies the significant parameters on the design of the system, architecture and the technological structures to execute the task management lifecycle and the automated pipeline.

3.4 System architecture

The TaskFlow architecture uses a modular structure in which responsibilities are acknowledged between different layers. The architecture is supposed to support the automated functions of creating tasks, managing tasks, assigning tasks, testing, and deploying tasks, and each of the tasks is supposed to be facilitated in a single module. The architectural components comprise the following significant components:

- This module manages the entire life cycle of a task from the time when it is initiated to the time when it is finished. It also has submodules for creating tasks, managing tasks, and assigning tasks, and therefore tasks are set up in order, monitored, and delegated to the particular team members.
- In system, an automation framework is used to determine the quality of the tasks. An automation framework utilizes Selenium as a tool to run predefined test cases for verifying the outcome of the task against predefined requirements.
- The continuous integration and continuous deployment pipeline is utilized to automate build, integration, and deployment processes. It enables smooth activity transition from the development phase to the production phase, thereby ensuring activities are continually integrated and deployed effectively.

3.5 Task management implementation

The task management capability is the system's main capability, which allows users to create, manage, assign, and monitor tasks from start to completion. The task management system architecture consists of a user interface that facilitates project managers.

Task creation: Users can create new tasks, along with all the pertinent information, such as the task name, description, deadlines, and priority levels see Figure 3. This information is saved in a centralized task repository, which can be accessed by other modules in future phases.

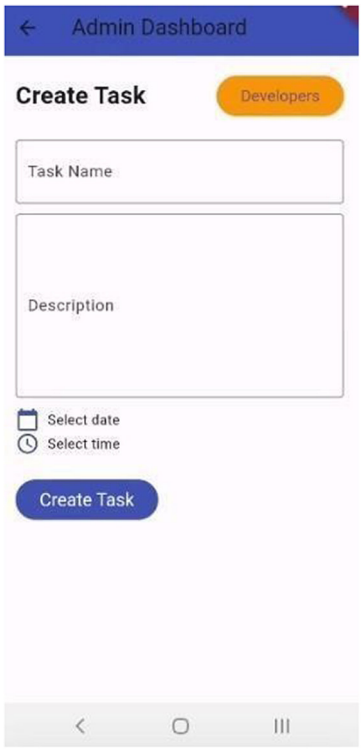

Fig. 3. Create task

Manage tasks: After task has been created, it can be further edited by project administrators or managers. It is accomplished by altering task attributes, i.e., updating deadlines or assigning priorities, and monitoring task progress (see Figures 4 and 5).

Fig. 4. Manage task

Fig. 5. All tasks

Assign tasks: Project managers can assign tasks to certain team members (assignees) see in Figure 6. Assigning is made easy using an easy-to-use interface with which one can easily select team members based on their availability and skills.

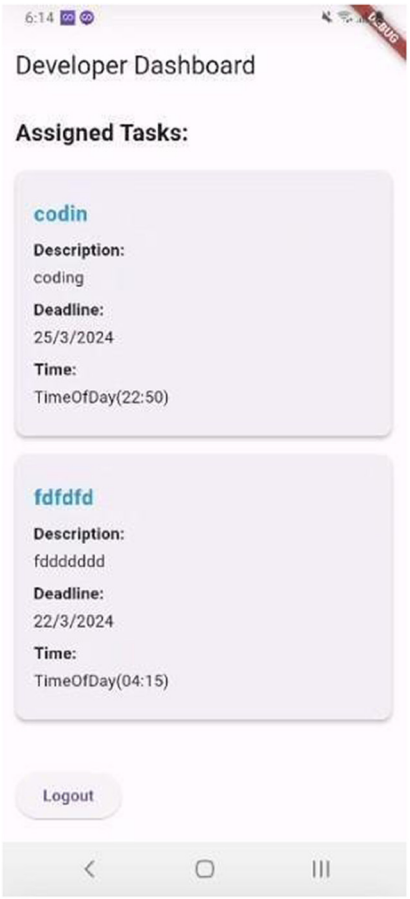

Fig. 6. Assign task

Task tracking: The task feature provides real-time updates on the status of tasks, allowing users to monitor progress and identify any delays or issues early on (see Figure 7).

Fig. 7. Confirm task

4 COMPARATIVE ANALYSIS

Table 1 gives an overview of the features offered by the three tools and the proposed TaskFlow system. It can be seen that although Trello, GitHub, and Microsoft Teams are individually better in their respective functionalities, none of the platforms provide an end-to-end solution encompassing all the major features required for contemporary DevOps and collaborative task management procedures.

Table 1. Comparison table

Features	TaskFlow	GitHub	Trello	Microsoft Team
Task Creation	✓	✓	✓	✓
Task Assignment	✓	✓	✓	✓
Real-time Collaboration	✓	×	×	✓
Version Control Integration	✓	✓	×	×
Notifications	✓	✓	✓	✓

(Continued)

Table 1. Comparison table *(Continued)*

Features	TaskFlow	GitHub	Trello	Microsoft Team
Cross-Platform	✓	×	×	✓
Automated Deployment	✓	×	×	×
User Feedback Loop	✓	✓	✓	✓
Integrated Chat	×	×	×	✓
DevOps Integration	✓	×	×	×
Reporting/Analytics	✓	×	×	✓

The TaskFlow offers large-scale task creation, task allocation, and notice management functions such as those delivered by GitHub, Trello and Microsoft Teams. It is unlike though since it has some additional reporting and analytical features to offer better monitoring of the progress and performance.

4.1 Enhanced real-time collaboration

Even though Trello and Microsoft Teams possess such an option as real-time collaboration, this option is absent in GitHub. The suggested system offers a seamless form of collaboration at every level of carrying out an activity, as such, eliminating the distance of communication between the developer and the manager and the developer and the tester.

4.2 Improved automation and DevOps integration

The most incredible thing about the proposed TaskFlow is that it can both deploy automatically and fully integrate with DevOps, which is lacking in all three tools of the comparison. This functionality promotes easy CI/CD operations, and it is completely compatible with the contemporary software delivery methods because TaskFlow and Microsoft Teams have strong cross-platform utilization on different devices and platforms in comparison to GitHub and Trello, which show limited cross-platform utilization.

4.3 Version control integration

Both TaskFlow and GitHub have close version control features, and this is highly significant in software development projects. It enables one to track the history and traceability of tasks which are linked to code.

4.4 Communication features

Microsoft is characterized by its native chat feature, which the suggested TaskFlow lacks. However, the third-party messaging apps' support may be able to compensate for this lacuna if need be.

5 RESULT AND DISCUSSION

The TaskFlow that was envisioned was created to improve task monitoring, collaboration, testing, and deployment by consolidating automation and DevOps practices into one platform. The section below outlines the major results that were achieved in implementing and testing the system and the effectiveness, strengths, and weaknesses of the system.

5.1 System evaluation

To measure the performance and efficiency of the TaskFlow system, the different modules were compared under a model project environment. The following results were obtained:

The system lowered the amount of manual effort necessary to handle tasks and deployments by a large degree. Automated testing and deployment alone saved 35–40% of operation time versus standard workflows (see Table 2).

Table 2. Feature of application

Module	Expected Outcome	Achieved Outcome	Status
Task Creation	Tasks created with title, description, priority	Successfully created and stored in DB	Success
Task Assignment	Assign tasks to specific users	Tasks assigned with user ID and updated status	Success
Real-Time Task Status	Reflect current state (e.g., In Progress, completed)	Status updates reflected in real-time	Success
Automated Testing	Run test cases on task completion	Selenium tests triggered and results recorded	Success
Deployment Pipeline	Deploy completed tasks automatically via CI/CD	Deployment completed with confirmation message	Success
Notifications	Notify users on updates or failed tests	Email-based notifications triggered successfully	Success
Reporting	Generate reports on task history and performance	PDF/Graph-based reports generated	Success

5.2 Error reduction

Manual deployment and testing are likely to be error-prone. Automation of testing (integration with Selenium) assisted in catching and correcting bugs earlier and decreased post-deployment bugs by about 50%.

Enhanced coordination:

Team coordination was facilitated by real-time updates and status tracking of the tasks. The system does not have its own chat, but the centralized dashboard was effective in terms of communication through status visibility.

5.3 DevOps alignment

The introduction of the CI/CD pipeline in the task flow led to the proximity of development and operations. It was typical DevOps practice since the work was developed and deployed with minimal human intervention.

Scalability and extensibility:

The modularity gives space to expand in the future with the integration of such tools as Slack, Microsoft Teams, or GitHub to enhance the concurrent and synchronous presence of the codes. The system is also scaled, and hence, it can support various projects and users simultaneously.

5.4 User feedback

The system was tested by a beta testing group who consisted of developers and project managers. Most of the severe criticism entailed, clean and intuitive UI, Automation of routine processes seamlessly, High-value analytical capability and reporting: There is no in-built chat system but instead third-party tools or email. It would be helpful to have more specific role management.

6 CONCLUSION

The paper shows that the TaskFlow system suits its objective of task management integration and automated testing and deployment. Its structure does not only help to improve the productivity of the team but also to ensure quality and constant delivery. Even though some features, like the features of communication and responsiveness of the UI, could still be improved, the system offers a good foundation for the modern development processes, especially for small- and medium-sized teams that are interested in practicing DevOps and do not rely on several third-party platforms.

7 REFERENCES

[1] M. Shahin, M. A. Babar, and L. Zhu, “Continuous integration, delivery and deployment: A systematic review on approaches, tools, challenges and practices,” *IEEE Access*, vol. 5, pp. 3909–3943, 2017. https://doi.org/10.1109/ACCESS.2017.2685629

[2] X. Wang, T. Räty, and A. Mäkilä, “Test automation process improvement in DevOps: A case study,” *arXiv, no. 2012.04321*, 2020.

[3] E. Alégroth, R. Feldt, and N. Kolstrup, “Maintenance of automated GUI tests: An empirical analysis of industrial cases,” *J. Syst. Softw.*, vol. 122, pp. 1–15, 2016.

[4] R. Mohan and L. Ben Othmane, “Improving CI/CD pipelines using containerized test environments,” *Int. J. Artif. Intell. Mach. Learn. Eng.,* vol. 9, no. 2, pp. 87–95, 2020.

[5] J. Mudadi and H. Lotriet, “Challenges in DevOps adoption: A South African case study,” *S. Afr. J. Ind. Eng.*, vol. 34, no. 1, pp. 130–142, 2023.

[6] L. Pando and R. Dávila, “A systematic mapping study on software testing in DevOps,” *Proc. Inst. Syst. Program. RAS*, vol. 35, no. 2, pp. 112–124, 2023.

[7] M. Keshavarz and A. M. Rahmani, “An AI-based architecture for intelligent DevOps pipeline automation,” *Future Gener. Comput. Syst.,* vol. 148, pp. 453–466, 2024.

[8] N. Hasani, "TestOps: Integrating testing lifecycle management into DevOps pipelines," *Softw. Qual. Prof.*, vol. 26, no. 2, pp. 55–67, 2024.
[9] N. Azad, R. Eramo *et al.*, "DevOps critical success factors: A systematic literature review," *Inf. Softw. Technol.,* vol. 162, p. 107180, 2023.
[10] R. Eramo, J. Cabot, and M. Wimmer, "An architecture for model-based and intelligent automation," *J. Syst. Softw.,* vol. 210, p. 111728, 2024.
[11] T. Väänänen, "Flutter in cross-platform development: Tools, performance, and optimization," Univ. of Oulu Repository, 2025.
[12] A. Farmanesh, "Developing a cross-platform mobile application for health data integration and capture from multi devices using flutter," Universidad Politécnica de Madrid, 2024.
[13] C. Zhou, "Challenges and solutions in cross-platform mobile development: A qualitative study of flutter and react native," Aalto Univ., 2024.
[14] K. Pfaffen, "Development and validation of automated tests for a multiplatform flutter application," HES-SO Univ. of Applied Sciences, Switzerland, 2024.
[15] A. Jain, "Scalable frameworks for cross-platform mobile app development," *J. Emerg. Technol. Innov. Res. (JETIR),* vol. 12, no. 6, pp. i676–i697, 2025.
[16] E. Afrihyia, A. U. Umana, and M. Appoh, "Enhancing software reliability through automated testing strategies and frameworks in cross-platform digital application environments," *Front. Multidiscip. Res.,* vol. 2, no. 2, 2022.
[17] A. M. Sattar, P. Soni, M. K. Ranjan, and A. Kumar, "Accelerating cross-platform development with Flutter framework," *Journal of Open Source Developments*, vol. 10, no. 2, 2023. https://doi.org/10.37591/joosd.v10i2.580

8 AUTHORS

Raheel Riaz is a researcher specializing in Internet of Things (IoT), DevOps, and sensor-based automation systems. He completed his Bachelor of Science in Software Engineering from Pir Mehr Ali Shah Arid Agriculture University, Rawalpindi, and is currently an MS Scholar in Computer Science. He has professional experience as a Software Engineer, working on automation solutions, database systems, and system integration. His research focuses on developing smart, data-driven monitoring and control systems by integrating IoT technologies with DevOps practices to enhance reliability, scalability, and automation. He is currently working on smart insect farming using IoT and DevOps. His broader research interests include IoT architectures, smart environments, and modern software engineering (E-mail: raheelriazraja@gmail.com).

Fadia Bibi is with the PMAS – Arid Agriculture University, University Institute of Information Technology, Rawalpindi, Pakistan.

Sidra Tahir is with the PMAS – Arid Agriculture University, University Institute of Information Technology, Rawalpindi, Pakistan.

Ahthasham Sajid is with the Department of Computer Science, Fazaia Bilquis College of Education for Women's PAF Nur Khan Base, Air University, Islamabad. Pakistan.

Sabitha Banu is with the Department of Computer Science & Cyber, PSGR Krishnammal College for Women, Coimbatore, Tamil Nadu, India.