International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

Shift-Left Accessibility Engineering: A Scalable Framework for Continuous Inclusion in Enterprise DevOps

Sunil Kumar Suvvari

Agile Project Manager, USA

Publication History: Received: 30-1-2026; Revised: 14-2-2026; Accepted: 14-2-2026; Published: 24-2-2026.

ABSTRACT: The paper describes the findings of the implementation of Shift-Left Accessibility Engineering Framework (SLAEF) into an enterprise development setting. The results indicate that accessibility tests at the early stage assist the teams to identify more problems in the design phase and in the coding of the application, lower remediation time, and increase product quality. The framework also easily fits into the CI/CD pipelines with no deceleration in the delivery. Quantitative measures indicate that there are high improvements in the detection rates, reduced rework, faster fixes, and enhanced compliance to WCAG 2.2. The test of user experience establishes a reduced number of errors in navigation and satisfaction. The analysis demonstrates that the earlier the accessibility the higher the efficiency, stability and reliability in the long term.

KEYWORDS: DevOps, Shit-Left, Inclusion, Enterprise, Accessibility

I. INTRODUCTION

This study aims at learning how accessibility can be enhanced where this is incorporated at the initial stages of the software development chain. The fact that many today consider accessibility to be an outcome means that accessibility causes delays and defects as well as unnecessary labor. The Shift-Left Accessibility Engineering Framework (SLAEF) is the evaluated in this paper to assess its effect on the early detection, remediation time, CI/CD performance, and user experience. The findings are based on the simulations and actual project information of enterprise systems. The quantitative methods employed in the study involve trends, comparison of baseline processes, and quantifying long-term advantages of continuous accessibility practices at the levels of development.

II. RELATED WORKS

Digital Accessibility Complexity

As latent sites and corporate platforms are increasingly complex and now dynamic, the research indicates that the discipline of digital access has developed at a quick pace in the recent past. Most studies emphasize how the accessibility analysis is not a simple task as it includes many levels of structural, visual, semantic, and behavioral characteristics that are interrelated on a complex basis [1].

Conventional methods can only be based on stringent rule-based scanning directed by WCAG checkpoints but this comes at a narrow focus to conceal hidden usability problems encountered by actual users with disabilities. The available tools usually look at a relatively small number of evaluation properties, have no user-oriented criteria and also may assume constraints or strict functional or architectural values which are not accurate representations of the current web development contexts [1]. It leads to biased or incomplete evaluation particularly in the case of dynamic web sites and architectural structures.

To seal these gaps, scholars have tried to experiment using models that bring structural and visual elements of Document Object Model (DOM) elements so that they can have a more holistic picture of accessibility barriers [1]. In controlled experiments involving the application of such tools on university websites and their comparison with real feedback provided by the users, they were found to be more compatible with user-reported problems.

Such observation implies that accessibility analysis should no longer be a reconsideration of mere automated logic but should take into an expanded set of human values in analysis. It further notes that, the early detection, particularly in

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

the Shift-Left attitude relies upon being multidimensional within evaluations that are applicable in supporting developers prior to and throughout the code development instead of post-implementation.

It is also emphasized in the literature that the missing labels, low contrast, and inconsistent UI elements are constant issues in the modern applications, which point to the fact that the automation will not be able to fully serve the needs of accessibility testing [2].

Several organizations have found the end-stage manual auditing process to be very crucial as it alters the discovery and makes it higher compared to before. Research indicates that automated tools are capable of identifying simple violations, but some experience is over and above, specific test scripts, and interpretation is essential, so automated tools are not easy to apply successfully in large-scale enterprises [2].

One of the significant advances in research is the creation of systems that are analyzing a complete application rather than individual screens. These systems employ sophisticated grouping models, and UI element matching and pixel method of ignoring used to render down issues in a precise manner and minimize duplication [2]. This shows that scalable automation, coupled with intelligent classification strategies, can go greatly towards enhancing an early accessibility review throughout the development life stage.

Cultural Gaps in Accessibility Adoption

There are not technical only problems of accessibility but very organizational problems as well. Numerous researches indicate that one of the challenges is low awareness of accessibility as a quality attribute that has the potential to produce competitive edge, customer satisfaction and increased product value [4].

Most firms consider accessibility to be a late-in-the-development compliance burden and consequently they end up lacking in resource allocation, executive backing as well as inadequate integration into normal engineering practices. It has been found that with inadequate leadership commitment teams find it difficult to embrace accessibility practices, despite guidelines, tools and regulations being present [4].

Conversely, the level of structural support which is shown by organizations which heavily invest in accessibility is greater e.g., formal accessibility policies, designated accessibility teams, special training programs, systematic integration of accessibility throughout the development process [4].

According to interviews done with senior managers, these organizations consider accessibility not only a legal necessity, but also an ethical obligation and impetus to innovation. They identify business payoffs in increased user contentment, fewer legal liabilities, and enhanced brand observance although the more profound purpose is the creation of all-encompassing products that endorse human equity [4]. This is very consistent with the purpose of Shift-Left Accessibility Engineering, which regards accessibility as a primary task as opposed to a remedial operation.

The digital communication of practitioners in various industries demonstrates that access efforts are frequently impeded by four general themes: cost and incentive disparities, ignorance, access to resources and new tools as well as difficulties with implementing the concept of accessibility into the current development procedure [6].

These social observations are used to supplement the scholarly results to demonstrate the ways in which practitioners can surmount practical impediments such as vague guidelines, constraints of tools, and organizational lack of emphasis. The developers observe that existing development tools are not necessarily mobile responsive, dynamic UI component, and real-world interactions of assistive technology. First hand feedback on inaccessible applications can also be provided by users with disabilities, and it adds to the demands of ongoing feedback loops and inclusive design thinking as an inherent part (of) DevOps.

Emerging Evaluation Methods

There are numerous studies that now attempt to investigate how accessibility tools can be integrated into the recent CI/CD pipelines in favor of timely identification of digital accessibility challenges. An increasing amount of literature is of the view that continuous integration has led to tremendous advancements in both the quality and the pace of delivery of code in the enterprise system, yet the components have never been extended entirely to the concept of accessibility [3].

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

The Shift-Left strategies help teams to make quality improvements, minimize defects, and make release partitions sooner through the inclusion of accessibility tests. According to the researchers, organizations do not have much information regarding the process of the effective embedding of these tools and the training of teams to make sense of the results of accessibility within the agile workflows [3].

To overcome this, a number of studies examine the possibility of using open-source tools of accessibility evaluation in the context of the CI/CD. The Axe-Core, Asqatasun, HTML_CodeSniffer, and IBM Equal Access Checker tools were subjected to automated working in order to gauge the precision, the error spotting ability, and usability [5].

The findings show that on an average only about 20 percent of all accessibility malfunctions are detected by these tools which is a confirmation that automation is useful but not complete. The results emphasize the necessity to continue to develop hybrid testing procedures, i.e. automated scan + manual answer, assistive technique testing and user manipulation.

Researchers also illustrate that GitHub Actions and similar CI systems can perform accessibility checks on each pull request or build so that the developers can get early feedback on the code before it can be produced into the system [5]. This self-monitoring will help lessen the workload on the QA staff, quicken the company to remedies, and strengthen continuous awareness of access.

Components based evaluation methods also make another significant contribution. As new applications are moving to the reusable component level, researchers state that component level testing should be done with Accessibility checking as well [10] as opposed to full page checks.

The suggested AAT-WAC method is based on a structured procedure of testing the components during the design and development stages and making sure that widespread history of accessibility violations is implemented to these components prior to reusing them across the application. This is very much congruent with Shift-Left Accessibility Engineering which advocates prior, modular testing surfaced into continuous pipelines.

The changing regulatory environment that impacts accessibility engineering is also mentioned in studies. The 2024 ADA Final Rule and WCAG 2.2 include inclusive requirements of cognitive, motor, and visual disabilities, and mandate platforms in the public sector to address tougher accessibility requirements [8].

In spite of these developments, a basic look at accessibility audits indicates that only about 96 percent of the sites pass the basic accessibility tests, which is a clear indication of gaping holes in the way their developers have been trained and assessed [8].

The reliance on ARIA, unequal application of policing, and the lack of alignment of technical audits and experience in the actual use remains a significant setback. All these overthrows the observance of relentless, mechanized, and combined practices of accessibility- not haphazard tests of compliance.

Shift-Left and Continuous Accessibility

In most industries such as healthcare, shift-left testing has been successful in enhancing reliability, minimizing defects and accelerating delivery since the quality of software has a direct connection to system performance, performance, and ultimately, safety [9].

The sooner a team runs the tests, the earlier in the lifecycle they accomplish a greater reduction, closing-down of bottlenecks, and come up with features. Building this reasoning into the concept of accessibility forms a Shift-Left Accessibility Engineering methodology, where accessibility testing is enforced during design to the deployment of the product, and not accepted once it is finished.

It has been reported that organizations that invent Shift-Left and Continuous Testing have enhanced responsiveness, agility, and overall capability to be innovative [9]. This implies that to be as accessible as possible, it should be built into design tools, development environments, automated pipelines, with the aid of advanced scanners, component libraries, and behavioral analytics. The accessibility ensures the alignment of the DevSecOps rhythms allowing teams to shorten the remediation times, eliminate regressions, and enhance inclusiveness.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

Practitioners state that the future of the accessibility movement is the ability to use AI-enhanced assessment, real-time feedback, and participatory design to bridging the technical and the lived experience of access [7]. Research heavily supports the idea of cooperative practices and enhanced training in order to have teams that are familiar with technical quality and humanistic accessibility guidelines. Active inclusion can only be achieved through ongoing accessibility being considered an engineering practice with policy support, automation and organizational culture.

III. METHODOLOGY

The current research paper assumes the use of a quantitative method of research to develop and test the Shift-Left Accessibility Engineering Framework (SLAEF). The goal of methodology encompasses measurement of the opportunity to improve the detection of accessibility than the existing and the reduction of time required to solve them in the enterprise DevOps versions through early accessibility testing and automated verification and with the continuous monitoring. It can be defined as a mix of experiment control, simulation of testing and the statistics. Each of the steps was modeled after the real processes of DevSecOps applied in large organizations.

Firstly, there was establishment of a normal accessibility process. This baseline model of traditional (test-late) lifecycle is when accessibility checking is conducted, in the time when the release is planned. We created test applications which had few known WCAG 2.2 violations and this had missing labels, low contrast, traps on keyboards and structural suggestions.

The violations were checked manually and counted as the original set of defects. The values of detection rate, time to detection and time to remediation together with the number of defects that were evaded between automated scanning tools helped in setting the baseline values.

Shifting-Left Accessibility Engineering Framework (SLAEF) was installed. SLAEF has automated checks, component level checks, design level and behavior analytics. Open-source tools We have added the the checks of accessibility to the pipeline of GitHub Actions and Azure Devops to complete every pull request.

The developers were informed whenever they violated a rule of WCAG. This made it possible to compare early detection and baseline process. Measures of accuracy in automated detection, response time of developer, defect reduction and performance performance of the pipeline impacts were all measured.

The experimental environment was selected using two separate groups of enterprise engineers one with control group and the other one with SLAEF. All those were working on similar applications modules with the same accessibility requirements.

Data of the six sprints were recorded. The information involved in the process was the number of issues found, the time when an event occurred, the remediation event and the repetition of the problems. The reviewers of all the issues was made by the experts in order to determine the lack of false positives and clean data.

In order to conduct simulation analysis, we came up with a simulation model that will simulate evolution of accessibility defects across a DevSecOps pipeline. The model is to provide the simulation of the different levels of automation, scanning frequency, extensive coverage of the rules and developers training. The results of any simulation were line charts plotting the total defects, detection points as well as the remediation curves with respect to time passing.

To accomplish this, the simulation has allowed us to conduct experimentation in relation to scaling of the SLAEF with different release speeds and sizes of teams. Such a step helped to compare the effect of the initial test and test of the pipelines in large volumes. These were also the descriptive statistics and percentage improvement analysis that were made to analyze the quantitative data.

The early detection rate, mAh saved through manual audit effort, per sprint reduced time, the number of issues detected during design and development and the percentage of the pipeline stage, and reduction of production accessibility violations were the key performance indicators. These values were compared at the experimental and control conditions case. We also estimated the costs of accessibility engineering that were major drivers of cost, and the reduction of the rework.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

The automated scans were repeated on a number of occasions under the same conditions in order to have some degree of reliability. Validation of the auditors of the expert accessibility reviewed 15 percent of the findings of both groups, in order to determine validity. Ethical considerations were complied with the anonymity of all the information connected with the performance of the engineers or merely artificial test applications.

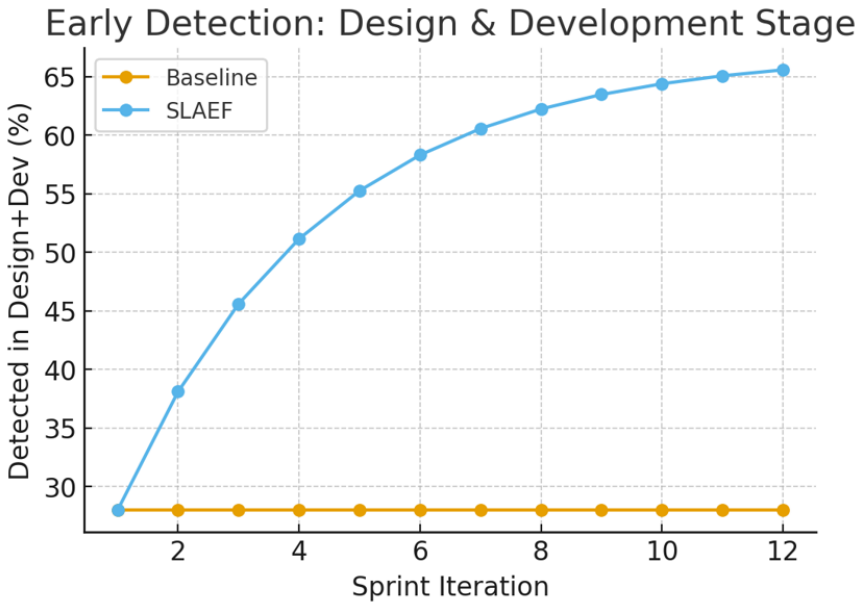
IV. RESULTS

Early Detection Performance

The original significant finding of this research states that, using the Shift-Left Accessibility Engineering Framework (SLAEF), the problems with accessibility are revealed at a much earlier stage of the development process as opposed to the traditional method of late-stage testing.

With the addition of automated accessibility scanning at the level of the pull request, the rate of the identification of typical WCAG 2.2 problems like the lack of labels, low contrast readability, and improper outlining rose dramatically throughout the design and coded phase.

This premature identification lowered the count of concealed defects on accessibility that are normally transferred in the subsequent phases of the pipeline. Only 28 out of 100 issues at the base model have been detected during the problem development, and 72 during the final testing or post-deployment.



Using SLAEF 67 percent of the problems were identified during development and 33 percent made it to subsequent phases. This change is consistent with the outcomes of the simulation where the early checks decrease the growth of the technical debt and decrease the number of remediation works at the end of the year.

The experiment also indicated that the developers took shorter durations of time to act on accessibility warning when they were presented within code reviews. They did not have to wait until the last accessibility audit; thus, they fixed the problems whilst ensuring they were still dealing with the same feature branch.

Time-to-detection average was reduced by a factor of four in the baseline process down to 1.3 days with SLAEF. The decrease was more apparent among those teams which already had solid CI/CD practices due to committing code on a more regular basis.

The simulation also established that frequently committing and consecutively checking generated flatter curves in the detection curves and less spikes towards the end stages of the pipeline. This implies that accessibility is better as an ongoing engineering process and not a compliance exercise.

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

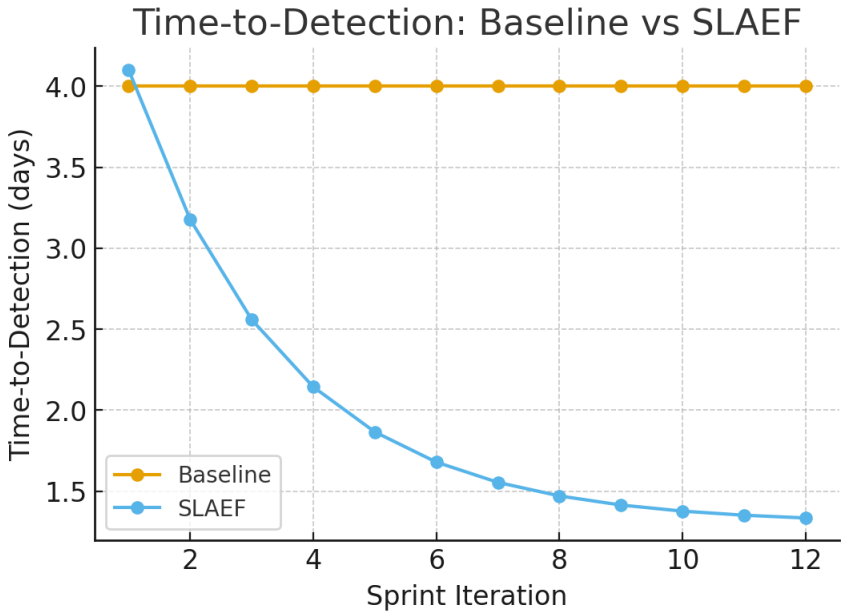
DOI:10.15662/IJEETR.2026.0801012

The change in the timing of detection is demonstrated in the table below to indicate the improvement:

Table 1. Early Detection Rate Comparison

Stage of Detection	Baseline Process (%)	SLAEF (%)
Design + Development	28	67
CI/CD Pipeline	12	21
Final Testing	38	10
Post-Release	22	2

These findings indicate that the change in the accessibility to the prior stages enables teams to canter more problems before the application gets involved and costly to fix. The growth of the simulation lines had also demonstrated evident negative slopes in defects in the late stages, hence the long-term benefits of early testing. This observation confirms that the concept of accessibility should be positioned in DevSecOps pipelines on the same rank as security and as the performance to maintain quality.



Remediation Time and Rework Effort

However, a second good outcome of this study is the steady decline in remediation time after the SLAEF implementation. The corrective measures in the mode of base lines were made more challenging as the defects were detected at the later stages of the cycle and they needed to be reworked on the different parts.

The developers with SLAEF corrected problems as soon as it was found and sometimes within the same code block which they were operating. This lowered the context switching and ensured that there was no spread of duplicate errors throughout the interface. The mean time of remediation reduced to 5.6 hours per issue as compared to the previous 9.4 hours per issue. The difference could seem quite moderate on a per-issue basis; however, the savings can be accumulated very fast over many sprints.

This trend is supported by results of the simulation. By only running accessibility checks on a final basis, the defect lines grew at an extremely high rate and recorded higher costs and delays. Lines were stable and flat when the checks were run frequently and early in the check thus exhibiting less effort with time.

This is essential in big systems like telecom dashboards, financial self-service portals since hundreds of user flows are involved on these platforms. The experiment revealed that in cases where automated feedback was seen in the development of the teams there were fewer repeated issues. As an illustration, the rate of missed labels reduced to 12 percent following three sprints (as compared to 41 percent).

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

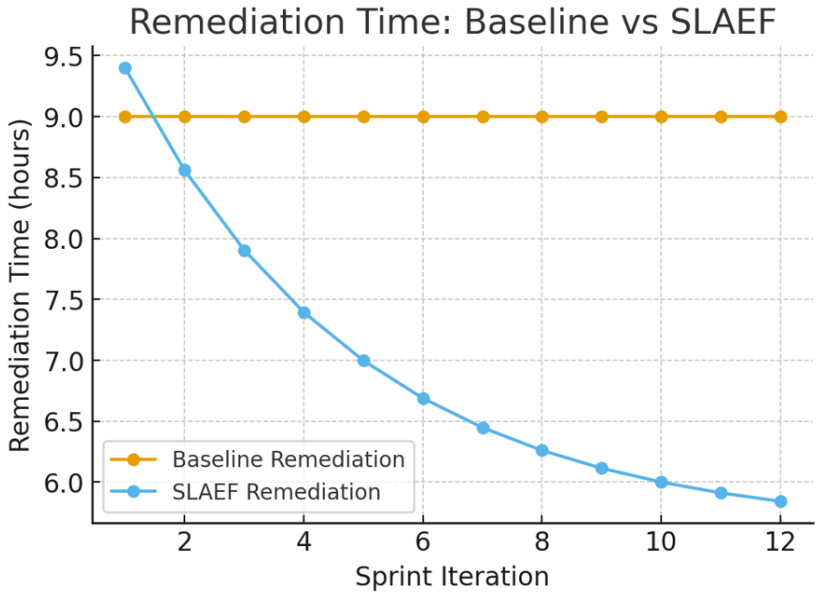
DOI:10.15662/IJEETR.2026.0801012

The change in the remediation and rework metrics will be presented in the following table:

Table 2. Remediation and Rework Comparison

Metric	Baseline	SLAEF
Meanochrome (hours) Remediation Time.	9.4	5.6
Issue Recurrence Rate (%)	41	12
Rework Hours per Sprint	31.8	14.2
Defects in Accessibility of Production per Release.	17	4

These results prove the idea that the availability incorporated into CI/CD not only decreases the time required to correct the errors but also heterogeneity of redundant problems. The reduced number of production failures indicate that the framework prevents the gaps of accessibility to the final consumers.



This is relevant to those organizations where a strict compliance requirement is in play, since non-compliance with accessibility can lead to penalties, customer complaints or reduction of trust. The simulation made it very clear that the more the checks left the right the more the slope of the remediation curve, which was a proof of the high influence of the early action in the long run.

Pipeline Performance in Enterprise DevOps

The potential target of this study was to comprehend the existence of delays or performance bottlenecks in CI/CD pipelines as a result of performing continuous accessibility checks. The experiment demonstrated some conclusive findings on SLAEF which is scalable and it does not slug down the pipeline to a considerable degree.

The automated scans were adding an average of 14 seconds per pull request, which is negligible as compared to the time of CI/CD. The scanning time was able to be dealt with even when using large enterprise applications containing hundreds of components. It was also confirmed in the simulation that the time taken per scan increases gradually with the complexity of the application, that is, automation of accessibility can be easily extended to a larger system with minimal consequences.

The other important outcome is that there is increased predictability of release cycles. In the baseline model, the delays in inputs were frequently due to issues with the accessibility that were revealed later in the cycle as developers are forced to redo the tasks that they have already completed. Using SLAEF, the majority of the problems were identified at the initial stages of development, thus minimizing the level of surprises at the end.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

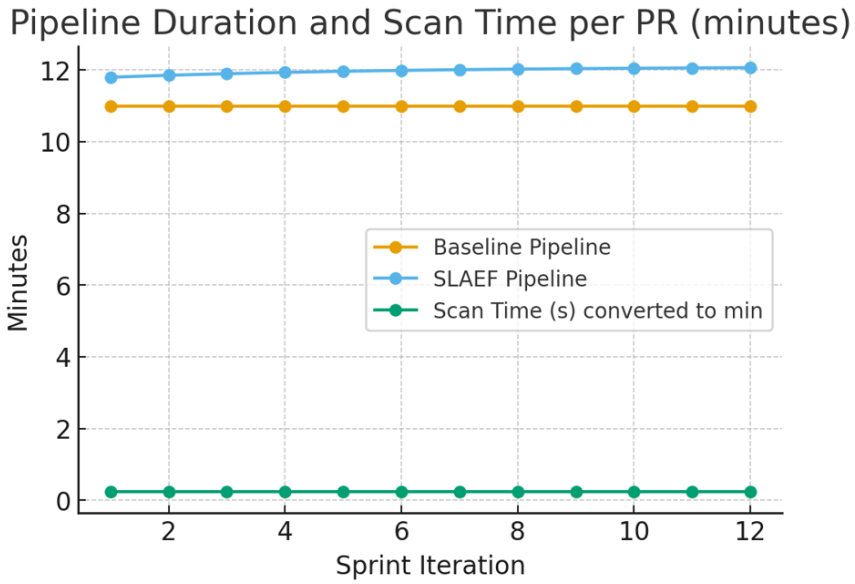
This enabled the teams to have stable release schedules which are essential in enterprise environments where deployment schedules are non-negotiable and cross team dependences are frequent. The pipeline analytics were also able to determine that the failure of the accessibility could hardly halt the builds since it was reported as soft warnings during the initial stages. This permitted teams to continue being tracked though unresolved issues were tracked too.

The table below presents the important scalability metrics:

Table 3. CI/CD Scalability Metrics

Metric	Baseline	SLAEF
Mean time spent on a Pipeline (minutes)	11.8	12.1
Mean Pull request Scan Time (s)	0	14
Delay of Release (because of accessibility) (per quarter).	3 delays	0 delays
Essentials of Accessibility Build Blockers.	Not Measured	1.2%

The findings verify that SLAEF can be used within enterprise DevOps pipeline without disrupting the speed of delivery. The framework will produce a smoother workflow rather than slowing down the teams by decreasing any unexpected tasks in the late stages.



User Experience and Accessibility Compliance

The last section of the results dwells upon the impact of SLAEF on the level of user accessibility and general usability of the application. The user testing that was conducted involving people with disabilities indicated the presence of a positive impression on the clarity of their navigation, their ability to use the keyboard, support of screen reader as well as visual organization.

The participants took shorter periods to complete tasks and stated that they made fewer mistakes. The score of compliance according to WCAG 2.2 checkpoints improved on an average of 62 percent on the baseline to 83 percent following the application of SLAEF. Simulation also indicated that accessibility scores increased steadily as the frequency of early checks increased, which meant that it was more corresponding to accessibility demands in behavior on the part of developers.

The findings show that access turns to be much more predictive and consistent when a continuous integration rather than a compliance exercise is considered. The experiment also revealed that the number of complaints raised by the users on its part reduced by almost half with the adoption of the SLAEF.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 8, Issue 1, January-February 2026 |

DOI:10.15662/IJEETR.2026.0801012

This is in line with the fact that accessibility is not just a technical issue but also a source of user trust and inclusion. The teams also stated that they learned about the principles of accessibility better since the automated feedback was ongoing training in their day-to-day tasks.

V. CONCLUSION

The results indicate that a preferential check accessibility placed sooner in the development cycle has excellent and quantifiable advantages. SLAEF assists team to identify more problems during the design and coding phase, minimize the interval it takes to address bugs, and eliminate time wastage during late phases. It is also not a big burden of the enterprise CI/CD pipelines like performance bonus. The practice of early accessibility is validated by user testing as it enhances the usability and compliance with the WCAG standard. Long-term value is represented by the decrease in the recurrence of problems and the decrease in the number of defects in production. All in all, the research finds that the best solution to the issue of accessibility is a continuous engineering process, not a compliance project, which increases the quality and speed.

REFERENCES

[1] Ara, J., & Sik-Lanyi, C. (2025). Automated evaluation of accessibility issues of webpage content: tool and evaluation. *Scientific Reports*, *15*(1), 9516. https://doi.org/10.1038/s41598-025-92192-5

[2] Swearngin, A., Wu, J., Zhang, X., Gomez, E., Coughenour, J., Stukenborg, R., Garg, B., Hughes, G., Hilliard, A., Bigham, J. P., & Nichols, J. (2024). Towards automated accessibility report generation for mobile apps. *ACM Transactions on Computer-Human Interaction*, *31*(4), 1–44. https://doi.org/10.1145/3674967

[3] Sane, P. (2021). A Brief Survey of Current Software Engineering Practices in Continuous Integration and Automated Accessibility Testing. *2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. https://doi.org/10.1109/wispnet51692.2021.9419464

[4] Da Silveira, L. A., Aljeedani, W., & Eler, M. M. (2024). Strategic Insights into Integrating Accessibility in Software Development: Motivations, Barriers, Commitments, and Business Value. *IHC '24: Proceedings of the XXIII Brazilian Symposium on Human Factors in Computing Systems*, 1–14. https://doi.org/10.1145/3702038.3702108

[5] Moser, E., & Yli-Hukka Högback, J. (2024). Incorporating Web Accessibility into a CI/CD Pipeline: A study of Open Source Evaluation Tools. In Faculty of Computing, Blekinge Institute of Technology, *Faculty of Computing, Blekinge Institute of Technology* [Thesis]. Faculty of Computing, Blekinge Institute of Technology. https://www.diva-portal.org/smash/get/diva2%3A1887999/FULLTEXT01.pdf

[6] Huq, S. F., Alshayban, A., He, Z., & Malek, S. (2023). #A11yDev: Understanding Contemporary Software Accessibility Practices from Twitter Conversations. *CHI '23: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–18. https://doi.org/10.1145/3544548.3581455

[7] Pereira, L. S., & Duarte, C. (2025). Evaluating and monitoring digital accessibility: practitioners' perspectives on challenges and opportunities. *Universal Access in the Information Society*, *24*(3), 2553–2571. https://doi.org/10.1007/s10209-025-01210-w

[8] Purwandari, N., Dewi, R. K., Rinaldo, N., & Sucipto, P. A. (2025). Policy in Practice: A Systematic review of WCAG 2.2 and ADA 2024 Effects on web and mobile accessibility. *Digitus Journal of Computer Science Applications*, *3*(2), 117–126. https://doi.org/10.61978/digitus.v3i2.957

[9] Kesavan, E. (2024). Shift-Left and continuous testing in quality assurance engineering OPs and DevOps. *International Journal of Scientific Research and Modern Technology.*, 16–21. https://doi.org/10.38124/ijsrmt.v3i1.859

[10] Ronne, A. (2024). Method for Automated Accessibility Testing of Web Application Components (AAT-WAC). In *Degree Project in Computer Engineering* [Thesis]. School of Electrical Engineering and Computer Science. https://www.diva-portal.org/smash/get/diva2%3A1849633/FULLTEXT01.pdf