Alqalam Journal of Medical and Applied Sciences. 2026;9(4):1053-1066
https://doi.org/10.54361/ajmas.269439

Original article

Investigating Automated Software Testing in Terms of Efficiency: A Comparative Study of Selenium and Playwright

Saad Almabruk[1], Samia Abdalhamid[2]*, Tahani Almabruk[2]

[1]*Department of Computer Science, School of Science, The Libyan Academy, Elbeida, Libya.*
[2]*Department of Computer Science, Faculty of Science, University of Omar Al-Mukhtar, Elbeida, Libya.*
Corresponding Email. *samia.abdalhamid@omu.edu.ly*

Abstract
This study compares the performance of Selenium and Playwright, two popular automated software testing tools, based on performance metrics such as response time, load time, and throughput. The tests measure their performance based on varying operating conditions, including different system loads and hardware configurations. Selenium's response times were slower on average compared to Playwright, particularly during low-traffic conditions, but it demonstrated consistent stability with no errors or timeouts across all test scenarios. Playwright, conversely, exhibited faster response times during low-traffic periods but encountered timeouts and errors under heavy traffic conditions, revealing a trade-off between speed and reliability. Further, the study identifies the involvement of hardware configurations, with Dell laptops outperforming HP laptops in all the parameters measured, thereby underlining the significance of properly optimized hardware in improving testing tool performance. The findings present a trade-off between the popularity of Selenium and the improved efficiency of Playwright, guiding testers seeking to maximize stability, speed, and system compatibility in automated testing pipelines.
Keywords. Software Testing, Selenium, Playwright, Efficiency.

Introduction

Automated software testing has become a necessity for companies looking to accelerate release cycles without sacrificing quality as a part of modern software development. The automated methods are different from manual testing and utilize tools and frameworks to run pre-written test cases rapidly, consistently, and with minimal human input [1]. As the demand for continuous integration and deployment (CI/CD) rises and the complexity of software applications increases, automated testing frameworks have become essential to increasing efficiency in the software development life cycle [2,3].

Automated software testing is essential in modern web development because it simplifies the process of creating, running, and analyzing tests. It enhances software quality and accelerates deployment [4]. Among the various tools available, Selenium and Playwright are two of the leading frameworks for automating web application testing. Selenium has long been regarded as an industry standard, valued for its robustness and flexibility [5]. Playwright, a modern alternative, aims to address several of Selenium's limitations by offering enhanced features and capabilities [6]. However, companies often lack sufficient guidance when selecting the appropriate framework, as there are few comprehensive, empirical comparisons of different tools under various operational conditions. This study conducts a comparative analysis of Selenium and Playwright to bridge this gap.

This research primarily aims to conduct a comprehensive analysis of the performance of Selenium and Playwright concerning key parameters of software testing, with a particular emphasis on efficiency. Specifically, this study examines factors such as homepage load times, navigation response times, image loading times, and throughput. By systematically evaluating the advantages and limitations of each framework, this research provides valuable insights that enable businesses to make informed decisions regarding their software testing strategies.

Furthermore, this study evaluates the performance of Selenium and Playwright in controlled environments by analyzing their behavior under various conditions, such as periods of high and low traffic. To examine the impact of hardware on test execution, the comparison includes two hardware platforms: HP and Dell laptops. While the report provides a comprehensive assessment of web application testing, it does not extend to other domains, such as API or mobile application testing. Additionally, although result variability due to differences in testing scripts, network conditions, and application architectures is acknowledged, it is not explicitly included within the scope of this study.

Efficiency is a fundamental metric for web application testing tools since it directly affects resource usage, execution time, and software quality. Selenium, as one of the most popular tools, has been instrumental in advancing testing efficiency through the use of automation. With its capability to automate repetitive tasks and run in parallel on different browsers and operating systems, it has become an indispensable component in Web Application Testing (WAT). Selenium Grid, for example, significantly reduces execution time by allowing concurrent test execution and thus maximizing resource usage in dynamic test environments [7]. However, Selenium's efficiency is to some extent undermined by its reliance on efficient locator strategies, which can make maintenance more complicated because of the high volatility of the user interface [8]. Despite these limitations, artificial intelligence framework integrations, such as Q-

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

learning, have been able to provide improvement in test coverage and execution path optimization, hence further cementing Selenium's applicability in contemporary testing procedures [9].

A different study highlights the efficacy of Selenium WebDriver in automating tasks like form filling and login activities, thereby demonstrating the feasibility of conducting simple and complex interactions. Reducing disturbance by using proper locator strategies, Selenium enables rapid fault identification and rigorous functionality testing, thereby facilitating high-quality software development [10]. The integration of Selenium with other tools such as Sikuli in hybrid test scripts enhances efficiency by automating visually oriented element identification processes, saving time and human effort in test execution [11].

Comparative analyses have identified emerging alternatives, such as Playwright, that demonstrate superior performance over Selenium in specific scenarios. Notably, Playwright's ability to execute tests more efficiently is reflected in a significant reduction in execution time—from 8.3 minutes with Selenium to 4.9 minutes for 12 test cases. This improvement underscores Playwright's effectiveness in managing parallel executions and handling complex user interactions [12,13]. Moreover, Playwright offers enhanced resource management by enabling test execution across multiple browser contexts without the need to launch separate instances, thereby minimizing overhead and improving scalability [13]. However, despite these advantages, Selenium remains a widely trusted solution due to its extensive community support and broad compatibility with various testing frameworks, making it a reliable choice for diverse testing environments [14].

Hybrid frameworks and tools highlight the scope for improving efficiency in automated testing. A review of the literature examining the use of Selenium with Sikuli demonstrates that structuring test cases in automated test suites decreases not just the necessity for manual intervention but also ensures more consistency in outcomes [11]. Similarly, integrating Playwright with CI/CD pipelines facilitates smooth test approvals and thereby maximizes efficiency in DevOps workflows [15]. In conclusion, although tools like Playwright demonstrate progress in terms of performance, Selenium's overall ecosystem and flexibility guarantee its long-term dominance in the field of Web Application Testing (WAT). This study aims to provide valuable insights into the comparative performance of Selenium and Playwright.

Methods

A comparative performance evaluation of Selenium and Playwright was conducted to respond to the questions of the study mentioned above. As may be observed from (Fig 1), several procedures were employed to reach the necessary outcomes.

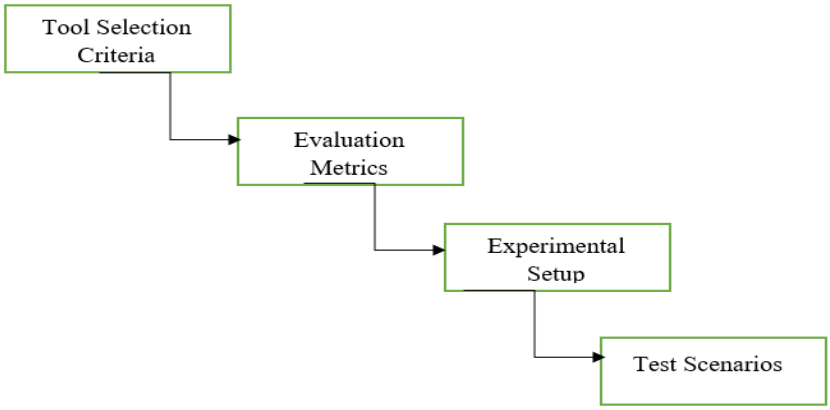

Figure 1. The procedure for comparing the performance of Playwright and Selenium

Tool Selection Criteria

Playwright and Selenium were selected for the purpose of this research because of their unique features and popularity in software testing. Battle-tested Selenium is a gold standard in automated testing because of its high compatibility with browsers and strong legacy system support. A new utility named Playwright possesses even more sophisticated features, including automatic browser context management and compatibility with newer web technologies, along with seamless interoperability with various browsers, like Chromium, WebKit, and Firefox. Due to these varying features, effectiveness in real testing scenarios can be exhaustively studied.

Evaluation Metrics

The study employed measures that are specific to assess efficiency, in terms of execution time (i.e., homepage loading time, navigation link response, and image loading) and throughput (actions per second).

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

Experimental Setup

Two laptops of varying hardware configurations were used for the test to compare the efficiency of Selenium and Playwright. The approach enabled a critical examination of the extent to which hardware variation affects the working effectiveness of the two frameworks.

Hardware Configuration

i. *HP Laptop*

- Processor: Intel Core i7-8550U CPU
- RAM: 12 GB
- Graphics: NVIDIA GeForce MX 130
- Storage: 930 GB HDD
- Operating System: Windows 10 Pro

ii. *Dell Latitude 3400*

- Processor: Intel Core i7-8565U CPU
- RAM: 12 GB
- Graphics: NVIDIA GeForce MX 130
- Storage: 256 GB SSD
- Operating System: Windows 10 Pro

The purpose of the selected hardware configurations was to assess the impact of system architecture and storage type (SSD and HDD) on Selenium and Playwright performance. It was hypothesized that the Dell Latitude 3400 D SSD would offer faster data access rates compared to the HDD of the HP laptop, thus potentially leading to a faster test run and better system responsiveness.

Software Environment

In order to provide a consistent testing environment, both laptops were configured to run Windows 10 Pro. To avoid performance variations that may arise as a result of disparities in software, the same version of browsers was installed on all the test machines. Specifically, the stable version of Google Chrome, version 130.0.6735.44, was installed to provide consistent browser behavior and rendering during the experiments. Moreover, automatic browser updates were disabled during the period of testing to ensure no potential inconsistencies due to version changes. As far as the development environment is concerned, both Playwright and Selenium scripts were executed with Python 3.x, hence giving both frameworks a level playing field. This was done to ensure a fair and credible performance benchmark of Playwright and Selenium under similar conditions.

Test Implementation

The methodology employed in this comparative study was designed to establish a common background for evaluating the performance of Playwright and Selenium frameworks in web application testing. Two scripts, one written in Python for Playwright and the other for Selenium, were composed to execute nearly identical action chains. These scripts were programmed to perform fundamental tasks frequently encountered in testing workflows, including measuring homepage load time, recording navigation link response times, and assessing image loading performance. Such tasks were deliberately selected to capture the frameworks' capacity to manage both static and dynamic web resources, while also simulating typical user interactions.

To ensure reliability and reproducibility, the scripts were executed repeatedly in batches over a continuous 24-hour period. Comprehensive logs were maintained throughout the testing process, documenting parameters such as execution stability, error occurrence, and response times. This systematic approach provided a robust dataset for comparing the efficiency, consistency, and resilience of the two frameworks under similar conditions, thereby enabling meaningful conclusions about their suitability for real-world web application testing.

Test Scenarios

In the present study, the test cases were carefully designed to evaluate the comparative performance of Selenium and Playwright in handling both static and dynamic elements on the designated webpage, *https://nclc.ly/*. The cases focused on three core interactions: loading the homepage, navigating through multiple links, and processing image content. To capture potential variations in resource usage and tool behavior, the tests were executed at three distinct times of day—8:00 AM, 4:00 PM, and 12:00 AM—representing peak, off-peak, and nighttime conditions. Performance assessment centered on quantifying the tools' efficiency in serving web content. Measurements included execution time, such as homepage load duration, navigation link response, and image loading speed, as well as throughput expressed in actions per second. These metrics were systematically recorded for subsequent analysis and visualization, enabling a detailed comparison of the two frameworks.

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

To ensure fairness and eliminate hardware bias, the tests were staggered across two machines (HP and Dell), alternating the execution of Selenium and Playwright at each time interval. For example, at 8:00 AM Selenium was executed on HP while Playwright ran on Dell, followed by a reversal at 8:15 AM. This alternating pattern was repeated at 4:00 PM and 12:00 AM, thereby producing a balanced dataset across different times and hardware configurations. The staggered approach guaranteed a comprehensive review of system performance under varying conditions, offering insights into stability, error frequency, and responsiveness of both frameworks.

Results and Discussion

This study is an in-depth examination of Selenium and Playwright in controlled test environments with the utilization of performance metrics essential for gauging their performance, including homepage loading time, navigation link response time, image loading time, and throughput. Data was collected through real-time observation of test executions on two hardware setups: an HP laptop with a hard disk drive (HDD) and a Dell laptop with a solid-state drive (SSD). This approach facilitated an examination of the extent to which hardware differences impact the performance of the two frameworks. Furthermore, the results are scrutinized meticulously, with a specific focus on variations in homepage load times, navigation link response times, image loading times, and throughput between Selenium and Playwright. Furthermore, the influence of hardware architecture, i.e., storage type (HDD or SSD), is also examined to determine its effect on tool performance. Through the objective comparison of both platforms, this study provides beneficial information on their advantages and drawbacks, thus presenting practical suggestions to testers and developers in choosing the optimal tool to utilize in automated testing activities.

Homepage Load Time

(Table 1) presents performance data of the Selenium and Playwright tools, based on the various experimental durations and laptop models (HP and Dell). The efficiency, stability, and responsiveness of the tools are indicated through the homepage load times recorded.

Table 1. presents the homepage load time performance of Selenium vs Playwright.

Experiment Time	Laptop	Tool	Homepage Load Time (s)
8:00 am	HP	Selenium	47.17
8:00 am	Dell	Playwright	51.89
8:15 am	HP	Playwright	46.99
8:15 am	Dell	Selenium	58.54
4:00 pm	HP	Selenium	174.11
4:00 pm	Dell	Playwright	Timeout
4:15 pm	HP	Playwright	Error
4:15 pm	Dell	Selenium	67.74
12:00 am	HP	Selenium	9.14
12:00 am	Dell	Playwright	8.65
12:15 am	HP	Playwright	9.12
12:15 am	Dell	Selenium	10.23

To allow for a more comprehensive analysis, the data is discussed in greater detail in (Table 2, 3, and 4), offering greater insight into the frameworks' performances. This sequence allows for a more granular exploration of how system architecture, including differences in hardware setups, tooling, and time slots, influences the efficiency and responsiveness of Selenium and Playwright under controlled test settings.
To allow for a more comprehensive analysis, the data is discussed in greater detail in (Table 2, 3, and 4), offering greater insight into the frameworks' performances. This sequence allows for a more granular exploration of how system architecture, including differences in hardware setups, tooling, and time slots, influences the efficiency and responsiveness of Selenium and Playwright under controlled test settings.

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

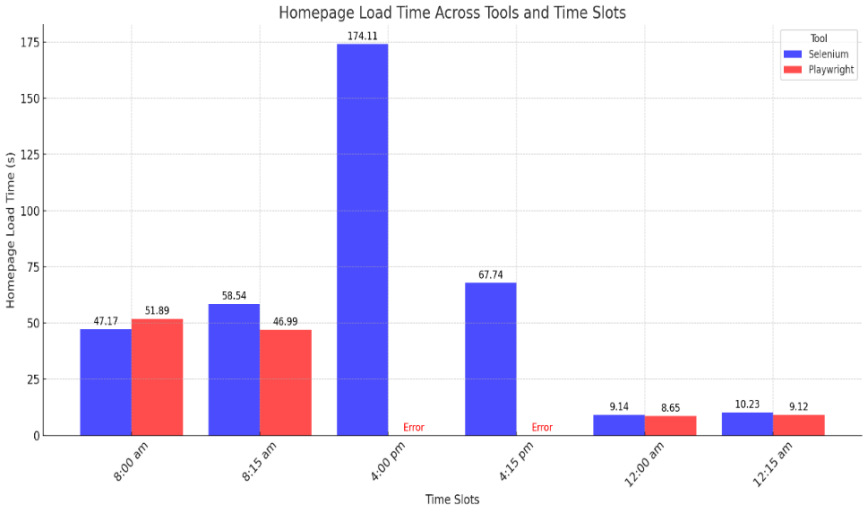

Figure 1. The bar chart represents the homepage load times for Selenium and Playwright across various time slots

A Performance by Tool

In terms of tool performance, Selenium is the more stable and reliable tool, while Playwright is faster but has a higher propensity for errors at peak traffic hours, as reflected in (Table 2). Specifically, Selenium records a mean load time of 61.32 seconds, hence guaranteeing that it performs similarly under any given circumstances. Yet, its load time is noticeably longer than Playwright's. On the other hand, Playwright reveals a mean load time of 28.53 seconds (excluding errors), which is considerably lower and performs exceptionally well even at low-traffic times. Once more, this does not reflect the failed scenarios, which may have an impact on its overall reliability. About errors and timeouts, Selenium reported no errors or timeouts, thereby demonstrating its stability and reliability across varying conditions.

Table 2. Comparison of homepage load times between Selenium and Playwright regarding performance by tool

Tool	Average Load Time (s)	Error/Timeout Frequency	Observations
Selenium	61.32	0	Consistent performance across all time slots, with no errors or timeouts.
Playwright	28.53 (excluding errors)	2 (1 Timeout, 1 Error)	Faster during low-traffic periods but unreliable under high-load conditions.

Performance by Laptop

On the HP laptop, Selenium recorded a 74.04-second average load time with no errors or timeouts, thereby exhibiting stable and consistent behavior across varying time periods. Playwright, on the other hand, recorded a considerably lesser average load time of 18.37 seconds (excluding errors) but incurred a single error at 4:15 PM, probably because of peak load conditions. On the Dell laptop, Selenium was reliable with a mean load time of 67.74 seconds and zero failures, and performed slightly better than on the HP laptop at peak times. Similarly, Playwright on the Dell laptop was efficient with a mean load time of 20.67 seconds (excluding a timeout), but failed at 4:00 PM via a timeout, thereby reaffirming potential instability under heavy loads, as indicated in (Table 3). In comparison, Selenium was stable and reliable on both laptops and was the ideal tool for any project that required consistency; the Dell laptop even performed slightly better than the HP laptop during some peak session times. Playwright had faster load times than Selenium on both devices--especially during periods of low load--but it was also prone to errors on the HP laptop, and towards the end of high-use sessions, it timed out on the Dell laptop; this suggests that it was slightly less stable than Selenium during periods of high use.

Table 3. Comparison of homepage load times between Selenium and Playwright concerning performance on a laptop

Laptop	Tool	Average Load Time (s)	Error/Timeout Frequency	Observations
HP	Selenium	74.04	0	Stable performance with no failures.
HP	Playwright	18.37 (excluding errors)	1 (Error)	Fast during low-traffic periods but failed at 4:15 PM.
Dell	Selenium	67.74	0	Stable performance, slightly faster than HP in peak hours.
Dell	Playwright	20.67 (excluding timeout)	1 (Timeout)	Efficient during low traffic, but timed out at 4:00 PM.

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

Performance by Time Slot

The results presented in (Table 4) highlight the influence of traffic load on the comparative performance of Selenium and Playwright across three distinct time periods. During off-peak hours at midnight, both tools provided a satisfactory user experience with negligible differences in load times, although Playwright demonstrated marginally faster performance. In the morning traffic period, the outcomes were more variable: Selenium performed slightly better at 8:00 AM, while Playwright showed a noticeable advantage at 8:15 AM, suggesting sensitivity to subtle fluctuations in system or network conditions. At high-traffic times in the afternoon, both frameworks encountered challenges. Selenium exhibited slower load times but remained functional and usable, whereas Playwright experienced failures due to timeouts and errors, indicating limitations in reliability under heavy demand. Taken together, these findings suggest that Selenium maintained consistent usability across all time slots, including peak periods, albeit with reduced speed. Playwright, by contrast, excelled in low-traffic conditions with superior load times but demonstrated instability during high-traffic periods. This contrast underscores the importance of considering both performance efficiency and reliability when selecting a framework for web application testing under varying traffic conditions.

Table 4. Comparison of homepage load times between Selenium and Playwright in regard to performance by time slot

Time Slot	Selenium Load Time (s)	Playwright Load Time (s)	Key Observations
8:00 AM	47.17	51.89	Comparable performance; Selenium is slightly faster.
8:15 AM	58.54	46.99	Playwright outperforms Selenium by approximately 11.55s.
4:00 PM	174.11	Timeout	Selenium slows significantly; Playwright fails entirely.
4:15 PM	67.74	Error	Selenium is stable; Playwright encounters an error.
12:00 AM	9.14	8.65	Both tools are highly efficient during off-peak hours.
12:15 AM	10.23	9.12	Similar results, with Playwright slightly faster.

From the results, we conclude that Selenium demonstrated consistent reliability, with no recorded failures throughout the testing period. In contrast, Playwright exhibited reliability challenges, particularly during peak hours, where timeouts and errors were more frequent. Performance analysis revealed that Playwright outperformed Selenium in low-traffic periods, such as at 12:00 AM and 12:15 AM. However, Selenium demonstrated competitive speed during moderate-traffic conditions but experienced noticeable slowdowns during peak traffic hours. Both Playwright and Selenium functioned efficiently across HP and Dell laptops, with minimal variations in performance. Nevertheless, Dell laptops exhibited a slight performance advantage over HP devices under peak traffic conditions.

Navigation Link Response Time

(Table 5 and Figure 3) Show average navigation link response times for Selenium and Playwright under varying experiment times and HP and Dell laptops. The comparison evaluates the performance, stability, and flexibility of tools based on recorded response times.

Table 5. presents the average navigation link response times performance of Selenium vs Playwright

Experiment Time	Laptop	Tool	Average Navigation Link Response Time (s)
8:00 am	HP	Selenium	9.21
8:00 am	Dell	Playwright	9.06
8:15 am	HP	Playwright	8.82
8:15 am	Dell	Selenium	10.96
4:00 pm	HP	Selenium	91.92
4:00 pm	Dell	Playwright	Error
4:15 pm	HP	Playwright	Error
4:15 pm	Dell	Selenium	62.88
12:00 am	HP	Selenium	2.49
12:00 am	Dell	Playwright	9.70
12:15 am	HP	Playwright	1.62
12:15 am	Dell	Selenium	10.18

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

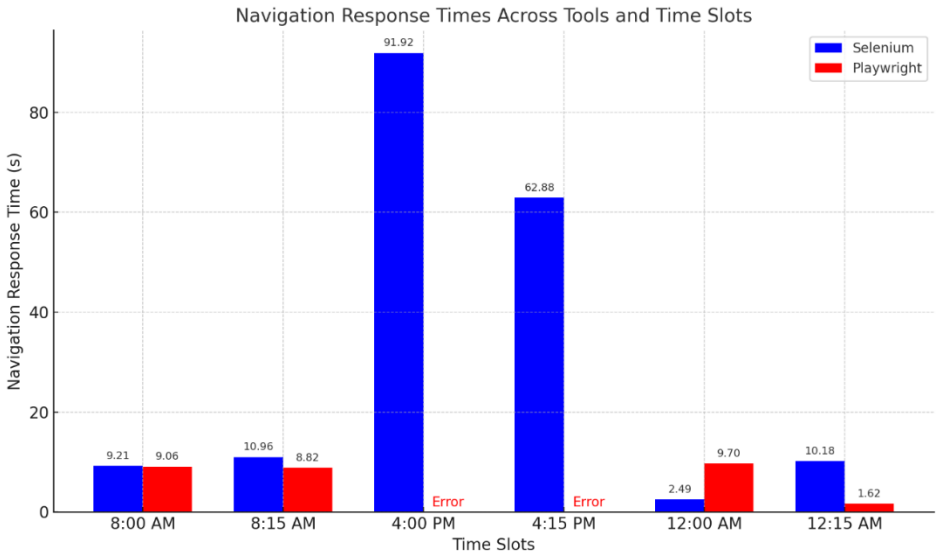

Figure 3. Bar chart represents the average navigation link response times for Selenium and Playwright t across various time slots

Performance by Tool

The results presented in (Table 6) offer a comparison of Selenium and Playwright navigation link response times with their respective performance measures. Selenium records a mean response time of 31.77 seconds, which is far above Playwright's 7.80 seconds (excluding errors). Although this degraded performance can be detrimental to user experience, where performance under timing constraints becomes paramount, Selenium records stable and consistent performance against changing loads and hence presents itself as a dependable option. On the contrary, Playwright is performance-tuned, hence providing significantly shorter response times in normal conditions. Its performance is, however, less predictable when under high load, indicating probable limitations regarding scalability or stability. On error frequency, Selenium executed without errors, proving to be robust and reliable under different situations, but at a slower pace. Although in Playwright 2 errors were witnessed, the occurrence of errors under high-load conditions suggests Playwright might not be excellent at scalability or accepting extreme conditions. That would be a weakness for such applications that require being highly reliable. It was discovered that Selenium gives good and reliable performance, whereas Playwright operates more quickly during periods of low traffic but is error-prone during high traffic.

Table 6. Comparison of navigation link response times between Selenium and Playwright regarding performance by tool

Tool	Average Response Time (s)	Error Frequency	Observations
Selenium	31.77	0	Consistent performance, though significantly slower during peak periods.
Playwright	7.80 (excluding errors)	2	Fast response times during low-traffic periods, but failed under high-load conditions.

Performance by Laptop

The side-by-side comparison of response times for navigation links by Playwright and Selenium on HP and Dell laptops reveals significant performance differences, as seen in (Table 7). Selenium executed consistently on the HP laptop, taking an average of 26.02 seconds, and it reported no errors. It was, however, significantly slower under heavy traffic. On the other hand, Playwright delivered a much faster mean response time of 5.22 seconds (excluding failure), roughly five times better than Selenium. Despite this gain in speed, Playwright registered a single failure at 4:15 PM, evidencing occasional instability under some conditions.

On the Dell laptop, Selenium tended to be slower with a mean response time of 37.52 seconds, but responded consistently even during the peak hours, as expected from its stability on the HP laptop. Playwright, although faster than Selenium on the Dell with a mean response time of 9.39 seconds (excluding errors), also failed once at 4:00 PM, as expected from the HP results. The slower performance on the Dell laptop for both tools suggests that environmental or hardware issues may influence response times. Overall, Playwright outperformed Selenium in speed on both laptops but was subject to random errors, which can impact reliability. Selenium, while being slower, is more stable across a range of circumstances. These results suggest that Playwright is better suited to circumstances that prioritize speed, particularly at times of low traffic; however, Selenium might be a more reliable choice for apps that

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

require the consistency of performance across a range of circumstances. The choice between the tools must consider hardware resources, traffic fluctuations, and the ability of the application to tolerate sporadic failure.

Table 7. Comparison of navigation link response times between Selenium and Playwright in regard to performance on a laptop

Laptop	Tool	Average Response Time (s)	Error Frequency	Observations
HP	Selenium	26.02	0	Stable performance, though slower during peak hours.
HP	Playwright	5.22 (excluding errors)	1	Exceptionally fast in low-traffic periods, but failed at 4:15 PM.
Dell	Selenium	37.52	0	Slightly slower overall but more consistent during peak traffic.
Dell	Playwright	9.39 (excluding errors)	1	Efficient during low traffic, but failed at 4:00 PM.

Performance by Time Slot

The findings in (Table 8) highlight notable differences in the performance of Playwright and Selenium in terms of navigation link response time at varying time intervals. In the morning session at 8:00 AM, both tools performed equally well, with Playwright having a slightly faster response time of 9.06 seconds compared to Selenium's 9.21 seconds. By 8:15 AM, however, Playwright significantly outperformed Selenium, performing about 2.14 seconds faster. During the afternoon, at 4:00 PM and 4:15 PM, Selenium had drastic slowdowns with response times of 91.92 seconds and 62.88 seconds, respectively. In the meantime, Playwright had errors and wasn't able to finish the tasks, suggesting stability problems. At midnight, the tools had a role reversal. At 12:00 AM, Selenium performed really well with a very quick response time of 2.49 seconds, significantly beating Playwright's 9.70 seconds. That said, by 12:15 AM, Playwright did better, finishing in only 1.62 seconds as opposed to Selenium's 10.18 seconds. What these findings indicate is that Playwright tends to perform better normally; however, it has stability issues when under load, while Selenium is less efficient but more stable when under heavy demand.

Table 8. Comparison of navigation link response times between Selenium and Playwright in regard to performance by time slot

Time Slot	Selenium Response Time (s)	Playwright Response Time (s)	Key Observations
8:00 AM	9.21	9.06	Comparable performance, with Playwright slightly faster.
8:15 AM	10.96	8.82	Playwright outperforms Selenium by approximately 2.14s.
4:00 PM	91.92	Error	Selenium slows significantly; Playwright fails entirely.
4:15 PM	62.88	Error	Selenium is slower but operational; Playwright fails.
12:00 AM	2.49	9.70	Selenium is exceptionally fast, significantly outperforming Playwright.
12:15 AM	10.18	1.62	Playwright is faster by a wide margin.

The examination of navigation link response times presents some significant findings. Selenium showed consistent reliability across all scenarios, and no errors were noted. Playwright, on the other hand, showed reliability issues, specifically failing at peak hours at 4:00 PM and 4:15 PM. In terms of speed, Playwright recorded significantly faster response times than Selenium under low-traffic conditions. However, while Selenium was comparatively slower, it presented stable performance in every scenario. Both Dell and HP laptops executed Selenium with skill, with minimal difference in performance. Nevertheless, while Playwright demonstrated a speed advantage on both laptop types, its excessive error rate nullified the advantage.

Image Loading Time

Table 9 provides image loading time for Selenium and Playwright under different experiment durations and laptops (HP and Dell). They are compared based on their effectiveness and consistency in relation to the recorded performance.

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

Table 9. presents the average navigation link response times performance of Selenium vs Playwright

Experiment Time	Laptop	Tool	Image Loading Performance
8:00 am	HP	Selenium	0.05
8:00 am	Dell	Playwright	0.02
8:15 am	HP	Playwright	0.05
8:15 am	Dell	Selenium	0.02
4:00 pm	HP	Selenium	0.09
4:00 pm	Dell	Playwright	0.05
4:15 pm	HP	Playwright	0.06
4:15 pm	Dell	Selenium	0.03
12:00 am	HP	Selenium	0.04
12:00 am	Dell	Playwright	0.03
12:15 am	HP	Playwright	0.02
12:15 am	Dell	Selenium	0.02

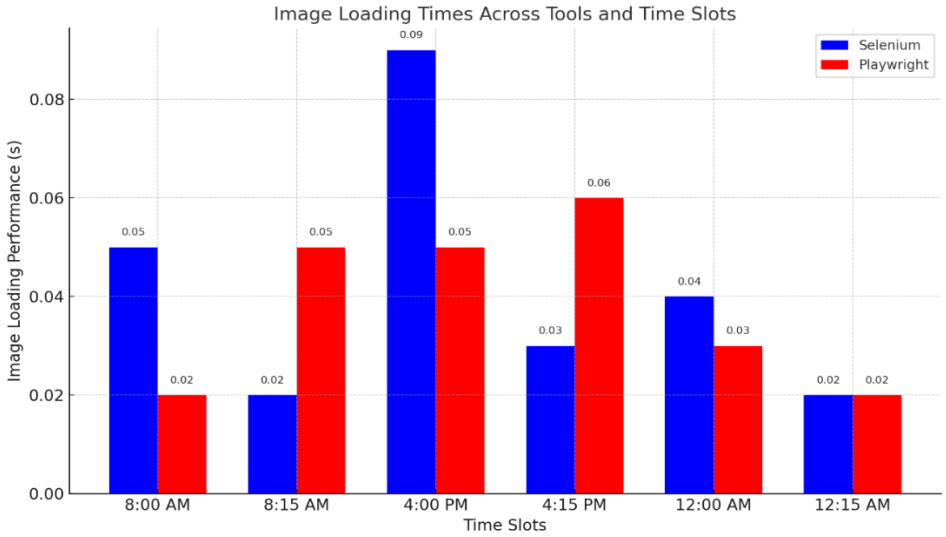

Figure 4. Bar chart represents the image loading times for Selenium and Playwright t across various time slots

Performance by Tool

The performance comparison of Selenium and Playwright based on image load times indicates that, on average, Playwright is a bit faster, with an average load time of 0.037 seconds against Selenium's 0.045 seconds. Although Playwright performs better in terms of speed, it also has slight variations in response times, indicating intermittent performance inconsistencies. In contrast, Selenium provides more consistent and stable results with minimal delays at 4:00 PM. Although Playwright performs better in terms of speed, Selenium's consistency in ensuring consistent speeds with minimal fluctuations, as presented in (Table 10) positions it as a better option in terms of stability. In conclusion, while Playwright is faster in terms of raw speed, Selenium is more consistent in ensuring consistent image loading time.

Table 10. Comparison of image loading times between Selenium and Playwright in regard to tool responsiveness

Tool	Average Image Loading Time (s)	Observations
Selenium	0.045	Stable and consistent performance, with slight delays at 4:00 PM.
Playwright	0.037	Slightly faster overall, but with marginal variation in response times.

Performance by Laptop

The image loading time comparison between Selenium and Playwright confirms that Playwright performs better in all cases, as demonstrated in (Table 11). In the HP laptop environment, Playwright records 0.043 seconds of average image load time, whereas Selenium records 0.055 seconds, thereby confirming that Playwright provides comparatively faster and more stable performance. On the Dell laptop, Playwright

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

consistently displays superior performance with 0.031 seconds loading time, which is slightly lower than Selenium's 0.032 seconds. This is an observation that indicates that Playwright is optimally tuned for quick image loading. Secondly, the Dell laptop outperforms the HP laptop in all test cases with quicker loading times, irrespective of the tool. The HP laptop does experience slight lag at peak hours, namely 4:00 PM, while using Selenium, which can be due to resource overload. Playwright, however, handles these peak hours more consistently. Playwright shows better performance in terms of faster and more stable image loading, while Dell outperforms HP in all the scenarios tested.

Table 11. Comparison of image loading times between Selenium and Playwright in regard to performance by tool

Laptop	Tool	Average Image Loading Time (s)	Observations
HP	Selenium	0.055	Slight delays during peak hours (4:00 PM).
HP	Playwright	0.043	Stable performance with slightly faster response times.
Dell	Selenium	0.032	Faster response times compared to HP across all conditions.
Dell	Playwright	0.031	Slightly faster than Dell Selenium, maintaining consistency.

Performance by Time Slot

The image load times of Playwright and Selenium are compared in (Table 12), which presents changing performance across varying time periods. At 8:00 AM, Playwright performs much better than Selenium, as it has a loading time of 0.02 seconds, whereas that of Selenium is 0.05 seconds. At 8:15 AM, Selenium performs better with a quicker loading time of 0.02 seconds, while the loading time of Playwright rises to 0.05 seconds. In the afternoon, at 4:00 PM, both tools slow down but still maintain an edge, where Playwright loads in 0.05 seconds while Selenium is at 0.09 seconds. Selenium records a marginal improvement in speed at 4:15 PM, where it loads in 0.03 seconds against Playwright's 0.06 seconds. At midnight, both tools have efficient performance, but Playwright has a marginally quicker response time of 0.03 seconds, while Selenium has a time of 0.04 seconds. Finally, at 12:15 AM, both tools performed the same, with each having a loading time of 0.02 seconds. In general, Playwright performs better in the majority of scenarios; however, Selenium performs better in some time intervals, which shows the impact of external factors such as server load, network traffic, or optimization of the tools at a specific moment.

Table 12. Comparison of image loading times between Selenium and Playwright regarding performance by time slot

Time Slot	Selenium Image Loading Time (s)	Playwright Image Loading Time (s)	Key Observations
8:00 AM	0.05	0.02	Playwright is significantly faster than Selenium.
8:15 AM	0.02	0.05	Selenium outperforms Playwright in this slot.
4:00 PM	0.09	0.05	Both tools slow down, but Playwright maintains an edge.
4:15 PM	0.03	0.06	Selenium is slightly faster.
12:00 AM	0.04	0.03	Both tools are efficient, with Playwright slightly ahead.
12:15 AM	0.02	0.02	Both tools perform equally well.

From these results, we conclude that both Selenium and Playwright demonstrated reliable performance, with no recorded errors or timeouts. Additionally, Dell laptops consistently exhibited faster response times for both tools, suggesting that hardware plays a role in performance variations. In terms of speed, Playwright outperformed Selenium on average, particularly in low-traffic conditions. However, while Selenium was slightly slower, it maintained stable operational performance across different scenarios. Regarding hardware influence, both Selenium and Playwright performed better on Dell laptops, with minimal performance differences between the two tools. This indicates that while hardware contributes to performance optimization, both tools remain effective across different systems.

Throughput (Actions per Second)

In this study, image loading time was used as the measurement unit to compare Selenium and Playwright's throughput capacities. (Table 13 and Figure 5) Provide a close analysis of throughput

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

performance according to varying time intervals and hardware setups, presenting variations in execution speed and efficiency under varying traffic conditions.

Table 13. presents the throughput performance of Selenium vs Playwright

Experiment Time	Laptop	Tool	Throughput (Actions/s)
8:00 am	HP	Selenium	1.41
8:00 am	Dell	Playwright	1.16
8:15 am	HP	Playwright	1.33
8:15 am	Dell	Selenium	1.31
4:00 pm	HP	Selenium	0.11
4:00 pm	Dell	Playwright	Timeout
4:15 pm	HP	Playwright	0.24
4:15 pm	Dell	Selenium	0.25
12:00 am	HP	Selenium	1.01
12:00 am	Dell	Playwright	0.87
12:15 am	HP	Playwright	0.73
12:15 am	Dell	Selenium	0.95

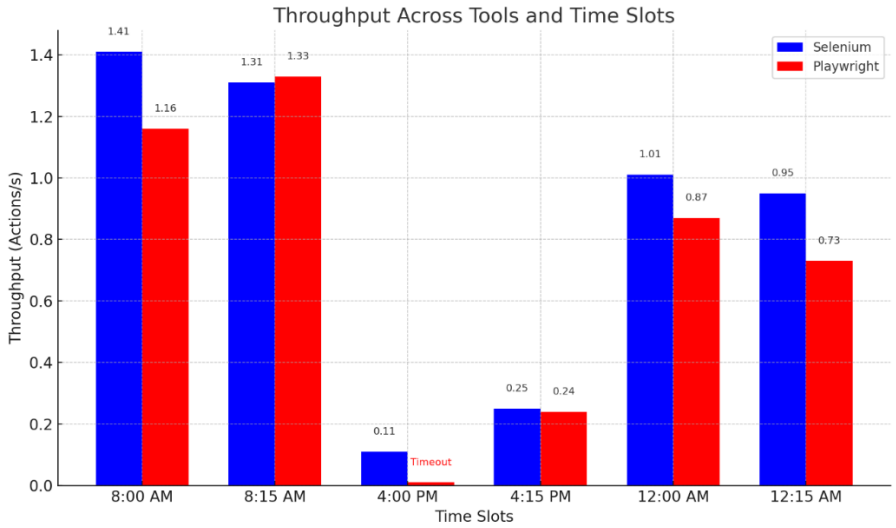

Figure 5. Bar chart represents the throughput for Selenium and Playwright t across various time slots

Performance by Tool

The comparison in (Table 14) illustrates the performance distinction between Selenium and Playwright regarding image loading time, considering both throughput and consistency. Playwright performs somewhat better, with an average image loading time of 0.037 seconds, about 18% quicker than Selenium's 0.045 seconds. The observation indicates that Playwright is able to incorporate optimizations to manage image loading operations more effectively. In terms of stability, Selenium is said to be stable and consistent but experiences small delays at certain points, e.g., around 4:00 PM, which could be due to excessive server loads or heightened client-side activity. Playwright, however, generally records faster performance but registers small variations in response time, perhaps due to how it manages asynchronous network activity or environment variables. Playwright's shorter mean loading time renders it a more efficient tool, particularly for use cases that demand high speed and rapid interaction. However, its negligible performance differences mean that it can trade off some consistency because of its resource management strategies at the expense of speed. Selenium can be better suited for use cases where stability and reliability are more vital, even if there is some compromise on speed.

Table 14. Comparison of throughput between Selenium and Playwright in regard to tool performance

Tool	Average Image Loading Time (s)	Observations
Selenium	0.045	Stable and consistent performance, with slight delays at 4:00 PM.
Playwright	0.037	Slightly faster overall, but with marginal variation in response times.

Received: *22-02-2026* - ***Accepted***: *19-06-2026* - ***Published***: *26-04-2026*

Performance by Laptop

(Table 15) compares the performance of Selenium and Playwright based on image load time on two laptops, i.e., HP and Dell. Selenium records an average image loading time of 0.055 seconds on the HP laptop but experiences slight delays at peak hours (4:00 PM), suggesting likely drops in performance under a scenario of increased load. Conversely, Playwright averages a faster loading time of 0.043 seconds and has more consistent performance, hence suggesting spikes are handled better. Moreover, Playwright runs faster on the HP laptop by approximately 22% and exhibits more stable behavior during the peak usage period. Compared to the HP laptop, the Dell laptop provides better results with Selenium, with an average image loading time of 0.032 seconds, which is an improvement. This could be due to better hardware or a more compatible operating system. Playwright on Dell is somewhat faster with an average loading time of 0.031 seconds, but it was consistent throughout all conditions. The difference in performance between Playwright and Selenium on a Dell laptop was negligible, with Playwright slightly faster at 3%. Overall, both tools achieved faster average loading times on the Dell laptop versus the HP laptop, again confirming that hardware matters with performance. To sum up, Playwright consistently delivered faster image load times when compared to Selenium, with a larger margin of error on the HP laptop. Playwright also delivered a more consistent performance under hard-stop conditions, such as peak traffic. If stability and slightly less-than-ideal faster response times are necessary, one should switch to Playwright. Both tools appropriately benefit from higher-performing hardware, which reiterates the importance of more hardware-aware results.

Table 15. Comparison of throughput between Selenium and Playwright regarding performance by tool

Laptop	Tool	Average Image Loading Time (s)	Observations
HP	Selenium	0.055	Slight delays during peak hours (4:00 PM).
HP	Playwright	0.043	Stable performance with slightly faster response times.
Dell	Selenium	0.032	Faster response times compared to HP across all conditions.
Dell	Playwright	0.031	Slightly faster than Dell Selenium, maintaining consistency.

Performance by Time Slot

An analysis of image load times in Selenium and Playwright, per every time slot, results in significant performance differences, which are illustrated in Table 16. Both tools demonstrate variable performance, which seems to be affected by external factors like network congestion. Playwright is consistently faster or on par with Selenium load times, particularly at the 8:00 AM and 4:00 PM timeslots. On the other hand, Selenium was faster than Playwright in both the 8:15 AM and the 4:15 PM timeslots, with only minor differences in performance. In the 12:15 PM timeslot, however, both tools performed similarly in load time, illustrating equal performance under certain parameters. Throughout the evaluation, the advantages of both tools are observed, as Playwright leads more often than not, but Selenium still produces reasonable image load times when compared to its alternative.

Table 16. Comparison of throughput between Selenium and Playwright in regard to performance by time slot

Time Slot	Selenium Image Loading Time (s)	Playwright Image Loading Time (s)	Key Observations
8:00 AM	0.05	0.02	Playwright is significantly faster than Selenium.
8:15 AM	0.02	0.05	Selenium outperforms Playwright in this slot.
4:00 PM	0.09	0.05	Both tools slow down, but Playwright maintains an edge.
4:15 PM	0.03	0.06	Selenium is slightly faster.
12:00 AM	0.04	0.03	Both tools are efficient, with Playwright slightly ahead.
12:15 AM	0.02	0.02	Both tools perform equally well.

Therefore, the answers to the study questions mentioned in Section 1 are as follows:

Research Question 1: How do Selenium and Playwright compare in terms of performance, specifically regarding load times and key efficiency metrics?

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

Answer: The comparison research showed that Playwright outperforms Selenium in performance metrics such as homepage load times, the responsiveness of navigation links, and image loading times, particularly in low traffic conditions. Playwright had faster execution and responsiveness and would be an excellent choice where low latency is a critical factor. However, this performance gain came at the expense of reduced stability under high-load conditions, where Playwright frequently encountered timeouts and failures in execution. In comparison, Selenium's performance was consistent and stable in all scenarios, albeit with slightly delayed response times. Its stability, especially in throughput at peak loads, testifies to Selenium's conduciveness to test environments in need of operational reliability and minimal error rates.

Research Question 2: How does the reliability of test execution differ between the two tools under varying traffic conditions?
Answer: Under different traffic loads, the tools were quite different in reliability. Selenium maintained a consistent pattern of execution under all traffic levels, with only marginal slowdowns in speed during heavy traffic, but without a major disruption to functionality. This stability is indicative of its robustness for use in dynamic, real-world test environments. Playwright, however, executed best under low traffic conditions, recording faster response times and successful execution. However, it progressively grew to be dependent on high traffic, leading to execution failure and increased downtime. These statements suggest that Selenium offers better reliability under variance in traffic, whereas Playwright can only stand under controlled, low-load environments.

Research Question 3: What impact does hardware configuration (e.g., HP vs. Dell laptops) have on the performance of Selenium and Playwright in a controlled testing environment?
Answer: The study evidenced that hardware configuration significantly affects the performance of Selenium and Playwright. Dell laptops outperformed HP laptops in all the parameters studied, including average response time and throughput, for both tools. This suggests that the computational efficiency and hardware optimization in the Dell systems enabled greater testing tool performance. These results point out that hardware capabilities cannot be overlooked when it comes to automated testing environments, as they can have a determining impact on tool behavior and performance. High-performance hardware investment can therefore be a critical consideration in order to gain optimum efficacy from automated testing tools.

Conclusion

This research offers a thorough analysis of Selenium and Playwright, two premier automated software testing tools, evaluating their performance under a variety of operating environments. The results reveal important performance evaluations measured by a number of metrics, such as load times, response times, and throughput, that will allow the authors to recommend data-driven choices for developers to use. Selenium is remarkably dependable and stable, which are great features for consistent testing environments. That said, though it is slower on average than Playwright, especially during low-traffic times, it can be optimized for speed-sensitive situations. Playwright is much faster and more efficient than Selenium during low-traffic times, but may not be as reliable during peak traffic, where you observed more errors and timeouts. In terms of hardware attributes, Dell laptops were consistently more responsive and offered greater throughput than HP laptops for both tools. This emphasizes the importance of solid hardware configuration to capture optimal performance benefits of automated testing tools. In summary, the study aids developers and testers in a better understanding of the trade-offs between efficiency and reliability in automated testing tools. If organizations align the selection of tools and investment in hardware with their intended testing needs, they can streamline their testing workflows and improve their quality assurance of software. Future studies need to explore integrating such tools with state-of-the-art technologies, such as AI-based test orchestration and cloud environments, to extend their scalability and flexibility further. In addition, overcoming limitations discovered, such as Selenium's variance in response time and Playwright's inadequate error handling, will enable these tools to keep pace with the evolving demands of today's software testing in a complete manner. This research offers a sound foundation for advancing the development and usage of automated test tools and contributes substantially to software quality assurance.

Conflicts of Interest

Declare conflicts of interest or state “The authors declare no conflicts of interest.

References

1. Rafi DM, Moses KRK, Petersen K, Mäntylä MV. Benefits and limitations of automated software testing: systematic literature review and practitioner survey. In: 2012 7th International Workshop on Automation of Software Test (AST). IEEE; 2012.
2. García B, Gallego M, Gortázar F, Munoz-Organero M. Automated driver management for Selenium WebDriver. Empir Softw Eng. 2021;26:1-51.
3. Mathew S. An overview on testing using Selenium. 2024.

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*

4. Wang Y, Källén M, Holmström Olsson H, Bosch J. Test automation maturity improves product quality—quantitative study of open source projects using continuous integration. J Syst Softw. 2022;188:111259.
5. Shariff SM. Investigating Selenium usage challenges and reducing the performance overhead of Selenium-based load tests [thesis]. Kingston (ON): Queen's University (Canada); 2019.
6. Bansal M, Dar MA, Bhat MM. Data ingestion and processing using Playwright. Authorea Preprints. 2023.
7. Aydos M, Çalık Y, Baykal B, Tüzün E. Security testing of web applications: a systematic mapping of the literature. J King Saud Univ Comput Inf Sci. 2022;34(9):6775-92.
8. Kiranagi VH, Shyam GK. Feature driven hybrid test automation framework (FDHTAF) for web based or cloud based application testing. In: 2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon). IEEE; 2017.
9. Fan Y, Chen Y, Sun Y, Liu X, Wang Q. A comprehensive evaluation of Q-learning based automatic web GUI testing. In: 2023 10th International Conference on Dependable Systems and Their Applications (DSA). IEEE; 2023.
10. Rusdiansyah R, Fauzi A, Kurniawan D, Hidayat T. Web program testing using Selenium Python: best practices and effective approaches. Sinkron J Penelit Tek Inform. 2024;8(2):994-1000.
11. Lathwal A. A literature review on automation testing using Selenium + Sikuli. Int J Distrib Artif Intell. 2019;11(2):35-40.
12. Melyawati NLP, Arsa DMST, Putra IMW, Paramartha IMA. Comparison of automation testing on card printer project using Playwright and Selenium tools. J Comput Netw Archit High Perform Comput. 2024;6(3):1309-20.
13. Vadia K, Patel H, Shah P, Mehta S. Bug testing automation with Playwright and a backend API. In: 2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). IEEE; 2024.
14. García B, Gortázar F, Munoz-Organero M. A survey of the Selenium ecosystem. Electronics. 2020;9(7):1067.
15. dos Santos Marques PT. Optimization of approval time in web UI tests [thesis]. 2023.

Received*: 22-02-2026 -* ***Accepted****: 19-06-2026 -* ***Published****: 26-04-2026*