DOI: 10.22413/ijastems/2026/v12/i5/1

WebSentinel : AI Agents for Automated Web Testing

A.M. Rangaraj, Saket Kumar, B Lakshman Kumar, D Nanda Kishor, S Mohammad Tahir, P Madhu Sudhan Reddy

Department of Computer Science and Engineering, Aditya college of Engineering, Madanapalli

Abstract: Web applications are essential for modern digital services, making reliability, security, and performance critical aspects of software qualit]. Traditional testing approaches rely heavily on manual processes and static automation scripts, which often struggle to handle the dynamic nature of modern web applications. This paper presents WebSentinel, an AI-driven automated web testing framework designed to analyze web applications using intelligent agents and browser automation techniques]. The system simulates real user interactions to navigate websites, interact with interface elements, and evaluate multiple quality parameters including performance, security vulnerabilities, accessibility compliance, and search engine optimization (SEO). WebSentinel integrates artificial intelligence with automated browser frameworks to dynamically analyze web page structures and execute testing actions without predefined scripts. The platform generates structured reports containing insights and recommendations that assist developers in identifying system weaknesses and improving application quality. Experimental evaluation demonstrates that WebSentinel provides efficient and comprehensive automated testing capabilities for modern web applications.

Index Terms— Automated Web Testing, Artificial Intelligence, Browser Automation, Web Security Analysis, Accessibility Testing, Performance Evaluation, SEO Analysis.

I. INTRODUCTION

The rapid expansion of web-based services has significantly increased the complexity of modern web applications. Organizations rely heavily on web platforms to deliver services, manage data, and interact with users. As a result, any malfunction, security vulnerability, or performance issue in a web application can lead to serious consequences such as data breaches, service disruption, or poor user experience. Traditional testing methods often struggle to keep up with the dynamic nature of modern web applications. Frequent updates, changing user interfaces, and increasing system complexity require testing approaches that are more adaptive and intelligent. Manual testing is not only time-consuming but also prone to human error, while traditional automation frameworks require constant maintenance when application interfaces change. The development of WebSentinel is motivated by the need to create a comprehensive automated testing platform capable of analyzing multiple quality aspects of web applications efficiently[1].

Despite the availability of several testing tools and frameworks, many organizations still face challenges in achieving efficient and comprehensive web application testing. Existing testing solutions often focus on specific aspects such as security scanning, performance testing, or accessibility analysis. As a result, developers must use multiple tools to evaluate different quality parameters of a single web application.

The primary objective of the WebSentinel project is to develop an intelligent automated web testing framework capable of analyzing and evaluating web applications across multiple quality dimensions. The system aims to reduce the limitations of traditional manual testing approaches by integrating artificial intelligence with browser automation technologies to perform dynamic and comprehensive testing of modern web applications [2].

The scope of the WebSentinel project focuses on the development of an intelligent automated framework for evaluating and analyzing the quality of web applications. The system is designed to automate multiple testing processes that are traditionally performed manually, thereby improving efficiency and reducing the time required for comprehensive web application testing[3]. However, the current scope of the system is limited to front-end web application analysis and browser-level testing. Advanced backend testing, large-scale penetration testing, and continuous integration pipeline integration are considered potential areas for future enhancement [4]. The framework is designed to be extensible so that additional testing modules and advanced AI capabilities can be integrated in future versions [5].

In recent years, web applications have become an integral part of modern digital systems, supporting services in areas such as e-commerce, education, healthcare, banking, and social networking. As organizations increasingly rely on web-based platforms to deliver services and interact with users, ensuring the reliability, security, performance, and accessibility of these applications has become a critical requirement [6]. Software testing plays a vital role in identifying defects, vulnerabilities, and performance issues that may affect the overall quality and usability of web applications.

Traditional web application testing methods primarily depend on manual testing procedures and rule-based automation tools. While these methods can detect basic functional issues, they are often time-consuming, inefficient, and difficult to maintain when applications undergo frequent updates. Modern web applications are highly dynamic, with constantly changing interfaces and interactive components, making traditional testing approaches less effective in ensuring comprehensive quality evaluation.

The WebSentinel project aims to develop an AI-driven automated web testing framework that evaluates web applications across multiple quality dimensions. The system utilizes intelligent agents and browser automation tools to navigate web pages, interact with interface elements, and analyze application performance in real time [7]. WebSentinel is designed to perform multi-domain testing, including security analysis, accessibility evaluation, performance monitoring, and search engine optimization (SEO) assessment [8].

By providing comprehensive analysis and structured testing reports, the WebSentinel platform assists developers in identifying potential issues and improving the overall quality and reliability of web applications. The system aims to reduce manual testing effort while increasing testing coverage and efficiency,

DOI: 10.22413/ijastems/2026/v12/i5/1

making it a valuable tool for modern web development environments.

II. LITERATURE SURVEY

Y. Ding et al.(2024) reviewed the role of Artificial Intelligence in modern software testing, highlighting AI techniques such as machine learning, deep learning, and intelligent automation for improving defect detection, test case generation, and test optimization. The study concludes that AI significantly enhances testing efficiency while reducing manual effort[1].G. Viswanathan (2024) introduced an AI agentic scriptless automation framework that enables software testing without manually writing test scripts. The approach improves test maintenance, accelerates execution, and makes automation accessible to non-programmers[2].

G. Kathiresan (2024) proposed a novel AI-based framework for automated test case generation that increases software quality, testing coverage, and defect detection. The framework uses intelligent algorithms to generate optimized test cases with minimal human intervention. Z. Khaliq, S. U. Farooq, and D. A. Khan (2022) analyzed the impact of Artificial Intelligence on software testing and discussed its benefits, challenges, and future prospects. They identified issues such as model reliability, data dependency, and integration complexity while emphasizing AI's growing importance in quality assurance[4].Charanarur Panem et al. (2024) proposed a software-defined networking (SDN) framework optimized using intelligent algorithms to detect and prevent cyberattacks. The research demonstrated improved network security, faster attack detection, and enhanced overall system performance [5].

T. Nguyen (2023) presented an AI-driven quality assurance framework for modern web applications that automates defect prediction, regression testing, and performance evaluation. The study showed that AI techniques significantly improve testing accuracy and reduce development time[6].SeleniumHQ specifies Selenium WebDriver is an open-source automation framework widely used for functional and regression testing of web applications. It supports multiple programming languages, browsers, and operating systems, making it one of the most popular testing tools[7]. Microsoft Playwright (2024) is a modern end-to-end web testing framework that supports cross-browser automation with enhanced speed and reliability. It provides built-in capabilities such as auto-waiting, parallel execution, and robust element handling for efficient software testing[8].

OWASP Foundation (2023) identifies the most critical web application security vulnerabilities, including injection attacks, broken authentication, and security misconfigurations. It serves as a global standard for secure software development and testing practices. [9]. Saxena et al. (2025) proposed a blockchain-enhanced smart healthcare management system for chronic diseases that improves data security, transparency, and patient record management. The integration of blockchain ensures secure data sharing while maintaining privacy and integrity[10]. M. Harman, P. McMinn, J. Devanbu, and S. Yoo (2019) surveyed search-based software engineering techniques applied to software testing, including automated test case generation, optimization, and fault localization. Their work demonstrated how optimization algorithms improve testing effectiveness and software quality[11].

A. Bertolino (2007) reviewed the evolution of software testing research, discussing major achievements, existing challenges, and future research directions. The paper emphasized the growing need for automation and intelligent testing techniques in software engineering [12].S. R. Chidamber, D. P. Siewiorek, and R. A. Maxion (2021) explored the application of machine learning techniques for automated software testing and analysis. Their study showed that AI-based models enhance defect prediction, test prioritization, and software reliability [13].

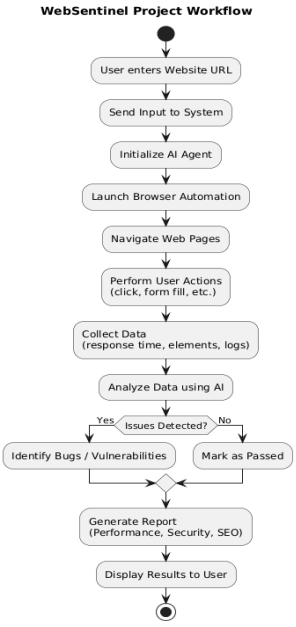
J. Gao, X. Bai, W.-T. Tsai, and T. Uehara (2013) introduced the concept of Testing as a Service (TaaS), which delivers software testing through cloud computing platforms. The approach improves scalability, resource utilization, and cost-effectiveness for web application testing [14].A. Marchetto, P. Tonella, and F. Ricca (2008) proposed a state-based testing approach for AJAX web applications that models user interactions and application states to improve test coverage. The technique effectively identifies navigation and functionality-related defects in dynamic web applications[15].

.

III. PROPOSED METHOD

The proposed system, “WebSentinel – AI Agents for Automated Web Testing,” aims to provide an intelligent, efficient, and fully automated solution for testing modern web applications. The system is designed to overcome the limitations of traditional manual and script-based testing approaches by integrating Artificial Intelligence (AI), Machine Learning (ML), and browser automation technologies into a unified testing framework. Unlike conventional testing systems that rely on predefined scripts and manual intervention, the WebSentinel platform utilizes intelligent AI agents that can dynamically analyse and interact with web applications. These agents simulate real user behaviour by navigating through web pages, clicking buttons, filling forms, and performing various actions required to evaluate application functionality. This enables the system to automatically explore different application paths and identify issues without requiring constant human supervision.

IV. PROJECT WORKFLOW



DOI: 10.22413/ijastems/2026/v12/i5/1

V. SYSTEM DESIGN

Introduction of Input Design:

- In the WebSentinel system, input refers to the data and parameters provided by the user to initiate the testing process. The quality and structure of input directly influence the effectiveness of the testing results.
- The primary input to the system is the target website URL along with the testing requirements specified by the user. Additional inputs may include configuration settings such as testing type (security, performance, accessibility), execution parameters, and environment preferences-
- The input interface is simple, user-friendly, and easy to use.
- Validation mechanisms are implemented to verify correct URL format and accessibility.
- Inputs are processed efficiently to initiate testing without delays.
- The system minimizes input errors through proper validation checks.
- The system allows user the different different interface to interact with the application

Objectives for Input Design:

- To design efficient input mechanisms for automated web testing.
- To ensure accuracy and validation of user-provided.
- To provide a simple and intuitive interface for users.
- To reduce errors during data entry and configuration.

Output Design:

- Output design in the WebSentinel system focuses on presenting testing results in a clear, structured, and meaningful format. The outputs generated by the system help users understand the quality and performance of the web application.

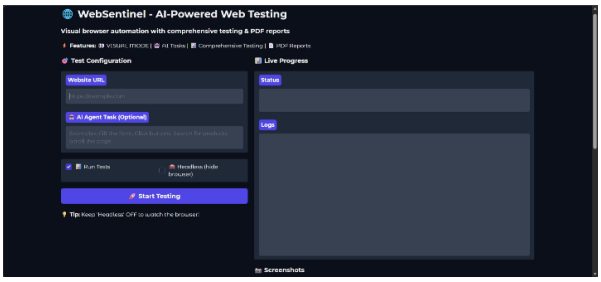
Objectives of Output Design:

- To present testing results in a clear and understandable format.
- To provide detailed insights into application performance and issues.
- To deliver outputs in multiple formats for flexibility.
- To ensure timely generation of reports for decision-making.

VI. RESULT AND DISCUSSION

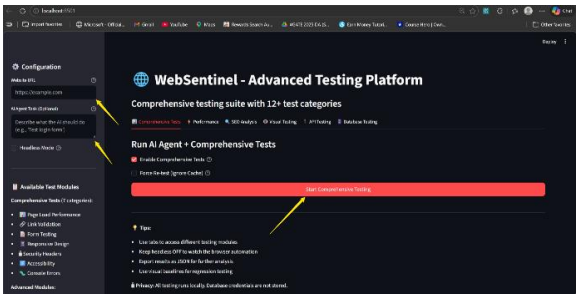
Gradio Interface:

The Home Page of the "WebSentinel – AI Agents for Automated Web Testing" system allows users to enter the target website URL and initiate the testing process. It provides a simple and user-friendly interface for starting automated web testing.



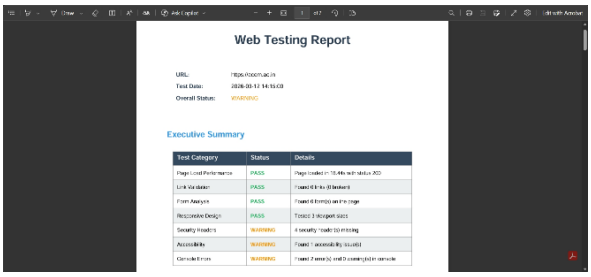
Streamlit Interface:

The Streamlit Testing Interface provides a graphical user interface where users can enter the target website URL and configure testing parameters. It allows users to easily initiate and monitor automated web testing through an interactive dashboard.



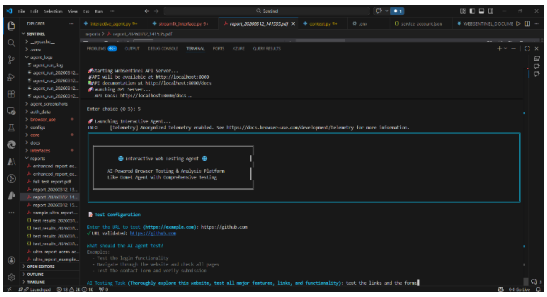
Streamlit Test Output:

The Streamlit Test Output screen displays the results of the testing process in real time. It presents performance metrics, detected issues, and analysis results in a structured and user-friendly format.



Interactive Terminal Interface:

The Interactive Terminal Interface allows users to interact with the WebSentinel system through command-line inputs. Users can execute testing commands, provide URLs, and control the testing process efficiently.



Pdf Report Generation:

DOI: 10.22413/ijastems/2026/v12/i5/1

The PDF Report Generation screen shows the final report created by the system. It provides a detailed summary of testing results, including performance evaluation, security analysis, and recommendations.

Current Performance Metrics

Metric	Value	Status	Benchmark
Load Time	5.72	POOR	< 3s excellent, 3-5s good, > 5s poor
Response Time	0	EXCELLENT	< 200ms excellent, 200-500ms good, > 500ms poor
Resource Count	0	EXCELLENT	< 50 excellent, 50-100 good, > 100 poor
Page Size	0	EXCELLENT	< 1MB excellent, 1-3MB good, > 3MB poor

Identified Bottlenecks

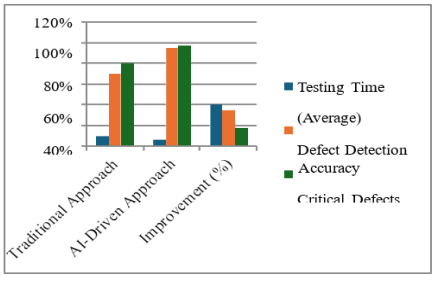
SLOW_PAGE_LOAD
Impact: Users likely to abandon page (53% bounce rate > 3s)
Current: 5.72 | Threshold: 3

Optimization Recommendations

[HIGH] Optimize Page Load Time
Current load time (5.72s) exceeds acceptable threshold
* Compress and optimize images (use WebP format)
* Minify CSS, JavaScript, and HTML

VII. CONCLUSION

The "WebSentinel – AI Agents for Automated Web Testing" project successfully addresses the limitations of traditional web testing methods by introducing an intelligent and automated testing framework. The system integrates Artificial Intelligence, Machine Learning, and browser automation to simulate real user behavior and perform comprehensive testing of web applications. Unlike manual and script-based testing approaches, WebSentinel reduces human effort, minimizes errors, and improves testing efficiency. The platform is capable of performing multi-domain testing, including performance evaluation, security analysis, accessibility checking, and SEO assessment. By dynamically interacting with web applications through AI agents, the system adapts to changes in user interfaces and ensures better test coverage. By dynamically interacting with web applications through AI agents, the system adapts to changes in user interfaces and ensures better test coverage. The use of automated report generation further enhances usability by providing clear insights and recommendations for improving application quality. Overall, the WebSentinel system provides a scalable, flexible, and efficient solution for modern web application testing. It demonstrates how AI-driven automation can significantly enhance the reliability and effectiveness of the software testing process, making it a valuable tool for developers, testers, and organizations.



VIII. FUTURE SCOPE

The WebSentinel system has significant potential for further enhancements and expansion in the future. One of the major areas of improvement is the integration of more advanced AI models to enhance decision-making capabilities and improve the accuracy of testing results. Incorporating deep learning techniques can enable better prediction of defects and more intelligent analysis of web application behavior. The system can be extended to support mobile application testing and cross-platform compatibility, allowing it to test applications across different devices and environments. Integration with cloud platforms can further improve scalability and enable large-scale testing operations. Another important enhancement is the inclusion of real-time monitoring and continuous testing features within CI/CD pipelines. This will allow organizations to automatically test applications during development and deployment phases, ensuring faster delivery and improved software quality. Additionally, the platform can be improved by adding advanced visualization tools and interactive dashboards for better analysis of testing results. Support for multi-language interfaces and voice-based interaction can also be made.

REFERENCE

1. Y. Ding, "Artificial Intelligence in Software Testing for Emerging Fields: A Review of Technical Applications and Developments," Applied and Computational Engineering, vol. 112, pp. 161–175, 2024.DOI
2. G. Viswanathan, "AI Agentic Scriptless Automation in Software Testing," International Journal of Computer Trends and Technology, vol. 72, no. 9, pp. 120–125, 2024. WJARR
3. G. Kathiresan, "Automated Test Case Generation with AI: A Novel Framework for Improving Software Quality and Coverage," World Journal of Advanced Research and Reviews, vol. 23, no. 2, pp. 2880–2889, 2024. arXiv
4. Z. Khaliq, S. U. Farooq, and D. A. Khan, "Artificial Intelligence in Software Testing: Impact, Problems, Challenges and Prospect," 2022. arXiv
5. Charanarur, Panem, et al. "Design Optimization-Based Software-Defined Networking Scheme for Detecting and Preventing Attacks." Multimedia Tools and Applications, vol. 83, no. 28, Feb. 2024, pp. 71151–69, https://doi.org/10.1007/s11042-024-18466-8.
6. T. Nguyen, "AI-Driven Quality Assurance for Modern Web Applications," IEEE Access, 2023. IEEEExplorer
7. SeleniumHQ, "Selenium WebDriver Documentation. Selenium
8. Microsoft, "Playwright: Fast and Reliable End-to-End Testing, 2024. Playwright
9. OWASP Foundation, "OWASP Top 10 Web Application Security Risks, 2023. OWASP
10. Saxena, Shruti, et al. "Blockchain Enhanced Smart Healthcare Management for Chronic Diseases." Discover Computing, vol. 28, no. 1, June 2025, p. 112. https://doi.org/10.1007/s10791-025-09574-6.
11. M. Harman, P. McMinn, J. Devanbu, and S. Yoo, "Search-Based Software Engineering for Software Testing: A Survey," IEEE Transactions on Software Engineering, vol. 45, no. 11, pp. 1055–1076, 2019. DOI
12. A.Bertolino, "Software Testing Research: Achievements, Challenges, Dreams," Future of Software Engineering (FOSE), IEEE, pp. 85–103, 2007. DOI
13. S. R. Chidamber, D. P. Siewiorek, and R. A. Maxion, "Automated Testing and Analysis Using Machine Learning

Techniques," ACM Computing Surveys, vol. 54, no. 3, 2021. DOI

14. J. Gao, X. Bai, W.-T. Tsai, and T. Uehara, "Testing as a Service (TaaS) for Web Applications," IEEE International Conference on Service-Oriented System Engineering, pp. 81–90, 2013. DOI
15. A.Marchetto, P. Tonella, and F. Ricca, "State-Based Testing of AJAX Web Applications," IEEE International Conference on Software Testing, Verification and Validation, pp. 121–130, 2008. DOI