ASM Sc. J., 20(1), 2025
https://doi.org/10.32802/asmscj.2025.2006

Identifying Research Gaps in AI-Driven Software Testing: A Review of Automation Tools and Challenges in SMEs

S.S. Raju and W.Y. Leong*

INTI International University, Persiaran Perdana BBN, 71800 Nilai, Malaysia

AI-driven (Artificial Intelligence) software testing frameworks have become crucial in today's fast-paced digital environment to guarantee the timely delivery of high-quality systems. Software testing is undergoing a revolution because of the incorporation of AI-driven frameworks, which automate intricate test scenarios, improve accuracy, and drastically cut down on time-to-market. However, a lack of defined frameworks, cost concerns, and ambiguity over performance hinder the adoption of advanced AI-powered testing solutions by many SMEs (Small and Medium-Sized Enterprises). Researchers and industry professionals will use the review's conclusions as a starting point to close the gap between innovation and real-world application, guaranteeing the smooth incorporation of AI-driven testing frameworks into contemporary software engineering procedures. Initially, 3998 research papers were extracted and at the third filtering, 20 research papers were chosen for the final review. This study offers a comprehensive analysis of the commonly used automation tools in various stages of software testing. The findings of this literature review study suggest there are no experience-based testing approaches for SMEs. There is a need to conduct surveys with practitioners to identify the benefits and limitations they are experiencing from using these automation tools for testing and ultimately to provide a comprehensive framework for automation software testing.

Keywords: Artificial Intelligence (AI); Small and Medium-Sized Enterprise (SMEs); Software Testing; Innovation; Testing Framework

I. INTRODUCTION

Software testing, which methodically assesses the functionality, performance, and dependability of software, is essential in an age of rapid technological innovation. Maintaining reliable and error-free applications has become more important than ever as companies and consumers depend more and more on software-driven solutions. However, as technology advances, consumer expectations are always changing, requiring software systems to be updated and improved on a regular basis. This flexibility, sometimes referred to as modifiability, is necessary to keep software relevant and satisfy changing user needs. The intricacy of software systems and the increasing demand for quick deployment draw attention to the shortcomings of manual testing techniques. In order to overcome these obstacles, artificial intelligence (AI) has become a game-changer in software testing, providing creative answers that boost automation, optimise testing methods, and boost productivity. Organisations may facilitate testing, guarantee quicker release cycles, and preserve software quality with less human involvement by utilising AI.

This research study aims to find the current trends and standard automation tools available and their features. This study looks at the different approaches, methods and resources used in this field and assesses their effectiveness. AI can optimise testing tactics and automate the testing process, which eventually improves software testing's effectiveness, accessibility, and efficiency. Furthermore, AI is a workable way to address the lack of qualified software testers. This research study will look into the following research questions.

RQ1: What obstacles prevent businesses from implementing AI in software testing?

*Corresponding author's e-mail: waiyie@gmail.com

RQ2: What are the common types of test automation frameworks which are available?

RQ3: Which software testing automation tools are commonly used among small and medium-sized enterprises (SMEs)?

RQ4: What are the factors to be considered in choosing the right automation tools?

There are 3998 publications or research studies from various research libraries that were evaluated for this investigation. After a few rounds of reading and filtering the research papers, 20 research studies have been identified for final review. The contributions of this study are mentioned below.

• To identify AI tools and their features available for automating software testing.

• To identify research that has been conducted in automation software testing.

II. RELATED WORKS

According to (Verma, Choudhary & Tiwari, 2023), the software industry is extremely dynamic, unpredictable, and volatile; new trends in software have been emerging since the 19th century to satisfy the expanding demands of consumers. However, software quality and dependability have always been a top priority during this time. Software reliability is becoming a more significant factor as society becomes increasingly reliant on the daily use of software. Software testing is the primary determinant of software reliability. More dependable software could result from improved software testing. Since testing entails a number of tasks, it has been found to take up over 50% of software development time. The following tasks are included in testing activities: creating test cases, creating test suites, running test suites, evaluating the execution's outcomes, and creating testing activity reports (Raju, 2025a; 2025b).

As stated in the World Quality Report (Europe Italy and Eastern Europe, 2024), the adoption of AI, particularly Gen AI, is advancing in the world of Quality Engineer(QE). A recent survey shows that 61% of Chief Information Officers (CIOs) have more confidence in the IT systems due to the improved testing coverage. According to 60% of the senior management team, the delivery of more features is possible due to higher levels of automation. As per the survey, 59% of CIOs mentioned that the system can go live faster due to the faster testing through automation testing, which allows for an increase in customer base.

According to (Ateşoğulları & Mishra, 2020; Ateşoğulları & Mishra, 2020), the International Software Testing Qualifications Board's annual test reports, which are released globally, show that test automation rose from 58.5% in 2015–2016 to 64.4% in 2017–2018. This suggests that automation testing technologies will be important in the future, which encourages more research, analysis, and comparison of this kind of information.

(Leotta *et al.*, 2024) assert that online applications require excellent quality control and a rapid time to market. It's acknowledged that releasing web apps that are not up to par, that consumers dislike, or even worse, that have flaws, can result in a loss of business. A web application problem can result in indirect costs from a drop in consumer loyalty and brand reputation, in addition to direct losses in company income. As a result, one of the most important factors is the quality assurance of the generated applications.

Software testing is the primary method of confirming that software satisfies the established criteria. It takes over half of the time and money to develop software, and it is suggested that system infrastructure improvements could reduce expenses by up to one-third, as stated (Lima, Da Cruz & Ribeiro, 2020).

In the increasingly complicated and rapidly evolving digital world, integrating AI into testing automation is not just a trend but a necessary development that has the potential to revolutionise the way software quality is determined and eventually result in more dependable and trustworthy applications, according to (Prathyusha Nama, 2024).

Agile regression testing is growing in popularity these days. After every sprint or delivery window, test cases are run as part of agile testing according to (Mukherjee & Patnaik, 2021). Therefore, an efficient and reliable testing methodology is vital to produce quality and bug-free software.

According to (Dhaya Sindhu Battina, 2019), with test automation, "maintenance" is the biggest problem. To keep up with the growing complexity of software, more tests must be written. We have overburdened with testing and maintenance as a result. Debugging and fixing failed tests require a lot of time and work. According to recent studies, maintaining test accounts for about 40% of the testers' time.

The benefits of automation testing, as stated by (George Murazvu & Simon Parkinson, 2024), include test inclusion, reusability, and repeatability as well as the reduction of effort required for test executions, which might take days if done by hand. Engineers are looking for automated techniques to detect software flaws, which would increase accuracy and trust because even simple software systems can have complicated flaws.

III. AUTOMATED SOFTWARE TESTING TOOLS AND SOFTWARE TESTING MODELS

Automation testing is the process of running tests automatically without human intervention by using software tools. The chosen testing methodology has a big influence on the software testing process's efficacy and efficiency. The industry offers various automation testing tools with special features and capabilities. The particular needs of the software project and the testing team's preferences will determine which tool is used, as stated by (Khin Shn Thant & Hlaing Htake Khaung Tin, 2023).

There are three categories into which automated software tools can be divided, which are based on the tools' function, development phase, and mode of operation. There are several types of software testing, such as load testing, functional testing, and unit testing. Every test type has a specific tool, and certain tools apply to both test types, as explained by (Alferidah & Ahmed, 2020)

As stated by (Izzat & Saleem, 2023), operating systems, licensing options, supported programming languages, supported browsers, prices, and other features of automated software testing solutions can all be compared. Software testing tools can verify dependability, correctness, performance, and security. The software and technology stack that will be utilised, the specific testing needs, the skill sets that are already present inside the organisation, and the tool's license cost all have an impact on the tool selection process.

Table 1 shows the commonly used automation tools for software testing. Even though there are many open-source automation solutions on the market, they have significant drawbacks, such as one automation tool can't handle different types of testing, programming languages, and platforms. As a result, choosing the right automation tools and combining several automation tools could be required to develop a thorough and successful software testing plan.

Table 1. Automation Tools for Software Testing

Tools	Language Support	Platform Support	Types of Testing	Ease of Use	Other
Selenium (Sani & Jan, 2024)	Supports multiple programming languages Python, C#, JAVA, Perl, Ruby, and Groovy	Supports web automation multiple cross-browser Chrome, Fire-fox, Safari, etc	Functional testing	Moderate. Requires some programming skills	Open Source
Katalon Studio (Sani & Jan, 2024)	Groovy, JAVA, etc.	Support multiple platforms such as web, mobile, API and desktop applications.	Functional testing and API. Support record-and-playback and scripting for tests. Integrates with CI/CD tools like Jenkins and Git.	Easy, user-friendly and minimal coding required	Open Source
Ranorex	No specific scripting language (it is written in .NET using C hash, Iron python, and VB.net) (Samli, 2023)	Support multiple platforms such as web, mobile, API and desktop applications. (Izzat & Saleem, 2023)	Compatibility testing and Graphical User Interface testing. A comprehensive test summary report is provided after every test runs. Supports common programming languages that make it easier to edit recordings or create custom tests. (Izzat & Saleem, 2023)	Easy (Samli, 2023)	License
Junit (Sani & Jan, 2024)	JAVA	JAVA application	Simple unit testing in Java. Facilities CI by integrating with CI tools like Jenkins.	Requires some programming skills	Open Source
TestNG (Sani & Jan, 2024)	JAVA	JAVA application	Unit, functional, end-to-end, and integration testing. Parallel test execution for faster testing. Built-in support for data-driven testing.	Moderate	Open Source
Jenkins (Sani & Jan, 2024)	Multiple (via plugins)	Cross-platforms	Functional testing, open-source automation server primarily used for CI/CD.	Easy	Open Source
Apache JMeter (Samli, 2023)		Mobile testing and Web testing	Load, Performances, Stress	Requires some programming skills	Open Source
TestComplete (Sani & Jan, 2024)n	Support multiple scripting languages (JavaScript, Python, VBScript and etc).	Desktop, web, and mobile applications. It can automate GUI testing	Functional testing, broad platform support for GUI testing. Can integrate with other CI/CD	Moderate	License

		across different platforms.	tools for seamless workflow		
Parasoft Selenic (Garousi, Joy & Keleş, 2024)	Supports multiple programming languages JAVA, C#, Python and etc.	Supports web automation multiple cross-browser Chrome, Fire-fox, Safari, etc	Self-Healing test. Parasoft Selenic is a powerful tool that enhances Selenium testing by providing AI-powered features like self-healing tests, intelligent recommendations, and seamless integration with existing frameworks and CI/CD pipelines.	Requires some programming skills	Open Source
SmartBear Visual Test (Garousi, Joy & Keleş, 2024)	Support multiple scripting languages JavaScript, Python, VBScript and etc	Supports web automation multiple cross-browser Chrome, Fire-fox, Safari, etc	Visual testing using AI removes redundant components from the system, making it possible to find flaws more quickly, which lets testing proceed more quickly.	Moderate	Open Source
Rational Functional Tester (RFT) (Izzat & Saleem, 2023)	.NET and Web-based JAVA applications	Supports web automation multiple cross-browser Chrome, Fire-fox, Safari, etc	GUI, functional, data-driven, and regression functional testing. Ensuring that test plans and cases are maintained and executed for RFT defects area	Moderate	Open Source

Software testing models are testing activities that are planned, carried out, and managed using an organised strategy and methodology throughout the software development lifecycle. To ensure software quality, defect identification, and risk mitigation, these models determine when and how testing is carried out throughout the system development lifecycle. Table 2 shows the commonly used software testing models.

Table 2. Software Testing Models

Model	Testing Approach	Strengths	Weaknesses	AI-Driven Enhancements
Waterfall	Linear and Sequential Testing	Clear structure, good for requirements well defined	Late defect detection due to testing done towards the end of the development cycle, costly fixes	AI automates static code analysis, defect prediction, and test case optimisation for early defect detection.
V-Model	Parallel Verification and Validation	Early testing, defect prevention	Rigid, not suitable for frequent changes	AI automates static code analysis, defect prediction, and test case optimisation for early defect detection.
Agile	Iterative and flexible testing within sprints	Early defect detection, adaptable	Requires strong collaboration and automation challenges	AI-powered test case generation, self-healing automation, and intelligent bug triaging improve efficiency in Agile sprints.
DevOps	Integrate testing in CI/CD pipelines	Rapid releases, automation-driven	Requires robust infrastructure, continuous monitoring	AI enhances continuous testing by identifying high-risk areas, optimising test execution, and improving test coverage in CI/CD pipelines.

IV. METHOD

This review paper is a methodical investigation that has examined the research questions, existing work, empirical studies and gaps. In this review, 20 research papers have been studied from the year 2019 till 20 January 2025. These studies were collected from 5 databases such as Science Direct, Springer Nature Link, Emerald Insight, Wiley and Google Scholar. Table 3, shows the selected number of publications in the initial search string, as stated below.

A. Search String

In this systematic literature review, the search string for selecting the articles was ("Software Testing" AND "Automation Tool*"). There is further filtering done as well for Emerald Insight to filter by ‘Open Access Content’ and ‘Journals’. In Wiley database, there are further filters done by only selecting the subject ‘Computer Science’, ‘Open Access Content’ and ‘Journals’.

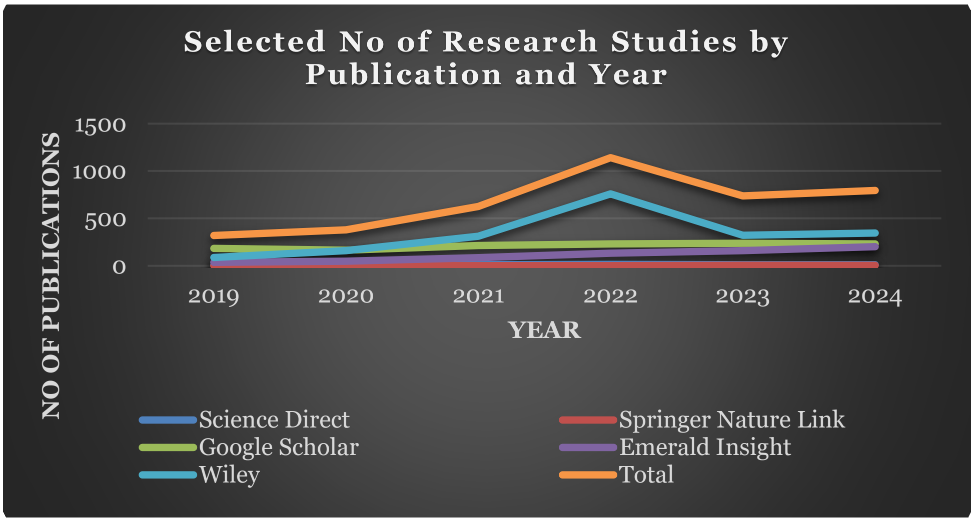
Initially, 3988 research papers were identified based on the initial selection criteria, of which 6 of the research papers were duplicates and were discarded from the selection. In the first screening, 3641 research papers were discarded due to the titles of the searched papers not being relevant to this study and then at the second screening, 290 research papers was discarded after reading the abstracts, methods and conclusion. Finally, the third screening, 41 research papers, was discarded since they did not fit the primary goals of the study and were outside its purview.

The final selection of 20 research papers was guided by the main objective of this study, which was to identify the current trends in software testing and commonly used automation tools in software testing and their features. These articles cover different types of software testing, AI-driven testing frameworks, and comprehensive analyses of automation tools. Additionally, the chosen studies have been crucial in identifying gaps in software automation testing and offering insightful information for further research.

Refer to Graph 1, since 2019 there has been an increase in research on automation in software testing. In 2022, there is a sudden peak in this area of research, however, there is a drop in 2023. n 2024, research in this area is once again on the rise.

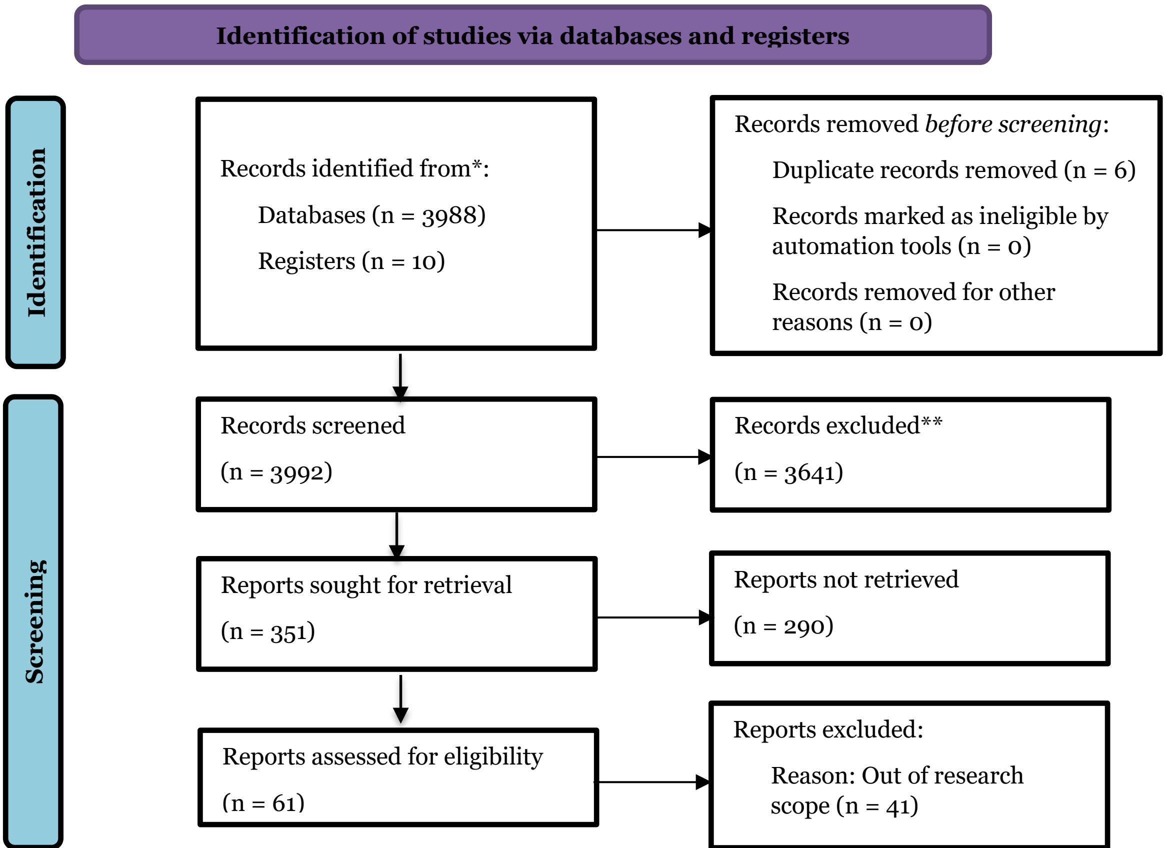
Table 3. Selected Number of Research Studies by Year and Publication

Publications	**2019**	**2020**	**2021**	**2022**	**2023**	**2024**	**2025**
Science Direct	9	4	11	15	13	11	3
Springer Nature Link	2	1	3	3	6	4	0
Google Scholar	184	165	214	232	238	232	11
Emerald Insight	37	50	85	132	159	202	26
Wiley	88	161	312	759	321	345	29
Total	**320**	**381**	**625**	**1141**	**737**	**794**	**69**



Graph 1. Selected No of Research Studies by Publication and Year

Data Screening and Extraction



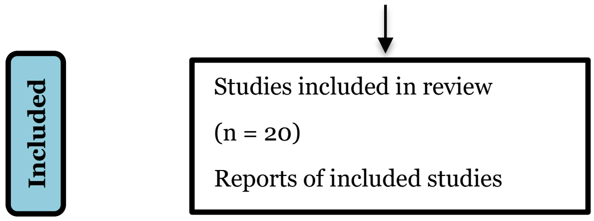


Figure 1. PRISMA flow diagram used in this study for literature review

V. RESULTS

This section provides a detailed study of the current literature review of 20 publications. As stated in Table 4, the objective, methodology, conclusion, and gaps of this study have been identified. The researcher also aims to investigate the answers to these research questions from the relevant publications.

RQ1: What obstacles prevent businesses from implementing AI in software testing?

According to (Khankhoje, 2024), there are several challenges faced by organisations to capitalise AI on full potential in their testing procedures. These challenges are data quality, algorithmic biases, complexity of tools, integration challenges, and lack of explainability. According to (Haeggström, 2024), the challenges identified are related to data privacy and confidentiality in sharing the code with OpenAI. There are many difficulties associated with software testing, according to (Salahirad, Gay & Mohammadi, 2023) ranging from test creation to process, automation, execution, and assessment.

RQ2: What are the common types of test automation frameworks which are available?

As stated by (Albarka Umar & Zhanfang, 2019), several types of test automation frameworks have their own key features and architecture, such as reusability and ease of maintenance. It is crucial to select the right framework for automating testing to ensure successful testing. The commonly used automation frameworks are linear automation framework, modular based framework, data-driven framework, keyword-driven framework, hybrid testing framework and library architecture framework.

RQ3: Which software testing automation tools are commonly used among small and medium-sized enterprises (SMEs)?

There are many automation tools, but not all tools are affordable, especially to SMEs. In this review paper based on the research done, these are the common tools available which are affordable for SMEs. Refer to Table 1, the automation tools available for software testing. The table details the automation tools with their features, such as language supported, platform supported, types of testing and ease of use.

RQ4: What are the factors to be considered in choosing the right automation tools?

According to (Albarka Umar & Zhanfang, 2019), there are various factors to be considered in order to choose the right automation tools, such as technology stack and its software, detail testing requirements which need to be tested, license cost and skill sets.

As per the literature review analysis in Table 4, there is a clear research gap in the discussion of the use of AI in software automation testing. It is identified that there is no experience-based testing approach in the literature review, as stated by (Izzat & Saleem, 2023). There is a need to conduct surveys with practitioners to identify the benefits and limitations they are experiencing from using these automation tools for testing. Empirical assessment could be done on other AI features, such as script creation and AI-based test cases, based on the Multivocal Literature Review conducted by (Garousi, Joy & Keleş, 2024). According to (Abdulwareth & Al-Shargabi, 2021), it is advised that developers and scholars widely validate the taxonomy. Consequently, an experimental investigation is advised since it can give a precise picture of the suggested taxonomy.

There is a gap that needs to be explored in the future by integrating different artificial intelligence approaches, utilising their advantages to tackle fuzzing issues, making constant advancements, and adopting fuzzing technologies to optimise test cases according to (Padmanabhan, 2024). To provide a deeper knowledge, a more thorough investigation into particular subtopics is required. Additionally, future studies must improve the availability and calibre of data available, as per stated by (Syafiq Rahman

& Farah Nadia, 2024). According to (Vo Thi Lan, 2024), with advancements in AI and machine learning, the function of automated software testing is anticipated to continue to grow. Therefore, the latest research must be ongoing to find the latest trends and innovations in automation testing. The same thought by (Bajaj & Samal, 2023), since there are ongoing improvements in AI technologies that will create robust and reliable systems, ongoing research in automation testing must be done. The researcher (Islam *et al.*, 2023), looked at a small number of studies, which is a drawback in and of itself. To obtain deeper insights, it would be beneficial to study a greater number of papers.

Table 4. Literature Review Analysis

Journal and Publication	Objectives	Methodology	Conclusion/Critique/ Comments	Gap
Empirical Anal. of Widely Used Website Auto. Testing Tools, EAI Endors. Trans. on AI and Robotics. (Sani & Jan, 2024)	This research has four goals: assess the efficacy of automated testing tools, identify the most effective tool for assessing website quality, evaluate tools based on performance, security and reliability and finally, identify best tools for optimising time, cost and resources.	Literature Review and empirical analysis of 10 automation tools.	This study explores automated testing tools that are commonly used for various types of testing in an industry and compares them based on their functionalities, platform support and ease of use. The analysis reveals that no single tool is perfect, as each tool offers its own features. Limited time and budget make it difficult to test websites using multiple tools.	There are a few gaps identified, we need to focus on machine learning and artificial intelligence in test automation to improve prediction and prioritise test cases. Explore cloud integrations that are more robust and cost-effective for better scalability, especially for large projects. Explore automating test result analysis and feedback loops.
Soft. Testing Tech. and Tools: A Review. Jnl. of Educ. and Sci. (Izzat & Saleem, 2023)	The objective of this study is to explain different testing types that are commonly used in various techniques.	Literature Review and examining methodology	It is concluded that using AI/ML will allow tests to be done faster and defects to be identified earlier. It will be able to determine the probability that a build will fail in the event that the application code is altered. Areas that will have problems and require more attention can be predicted more accurately.	The gap identified is that there are no experience-based testing approaches in the literature review
A Comprehensive Overview of Web-Based Auto. Testing Tools. Jnl. of Advanced Eng. Studies and Techn. (Samli, 2023)	The objective of this study is to assist the software testers in finding out the best web-based automated testing tools, which are efficient and effective.	Literature Review and analysis of 14 automation tools	This study has a wide-ranging analysis feature that has more tools and criteria than previous tool comparison papers. 20 different criteria are compared with 14 web-based automated tools and the outcome is presented.	Due to limited customer support available for some of the tools, there is insufficient information about certain testing tools.
A Study of Auto. Soft. Testing: Auto.Tools and Frameworks. Int. Jnl. of Computer Sci. Eng. (Albarka Umar & Zhanfang, 2019)	The goal of this study is to come up with a comprehensive study of test automation tools and frameworks. The functionalities and factors that need to be considered when selecting the right tools. Lastly, it highlights	Literature Review and comparative analysis of automated software testing methods, frameworks, and tools.	This report presents a detailed explanation about various test automation tools and frameworks as well as provides insights into some of the important factors to consider when selecting an automation	This journal was written in 2019; therefore, there is a need to review the latest tools and new features.

	the functions of unit testing and other testing techniques to guarantee the software's accuracy and dependability.		tool and framework. The paper stated that effective test automation requires the right tools and framework in order to have quality software and testing processes.	
AI-powered test auto. tools: A systematic review and empirical evaluation. (Garousi, Joy & Keleş, 2024)	There are three objectives in this study; a list of AI-based test automation tools, its features, how this feature will be helpful for effectiveness/ efficiency during testing and its limitations.	Multivocal Literature Review (MLR), Out of 55 tools, two tools were in the empirical assessment as part of the MLR.	This study provides a comparison of AI-based testing tools, and an Empirical Assessment was conducted on two AI-based testing tools, which are Parasoft Selenic and Smartbear VisualTest. Data collected as part of empirical analysis shows the effectiveness, efficiency and limitations of the features in these tools.	There are gaps identified, and there is a need to conduct surveys with practitioners to identify the advantages they are experiencing and what drawbacks they encounter while using such tools. Other AI characteristics, such AI-based test cases and script creation based on the MLR, can also be empirically evaluated.
Security testing of web appl.: A systematic mapping of the literature. Jnl of King Saud University - Computer and Information Sci. (Aydos *et al.*, 2022)	By offering the most comprehensive research in the area of web application security testing, this study hopes to assist researchers and practitioners.	80 papers were included in a systematic mapping of the literature with an emphasis on types of testing tools, vulnerability, and violation types. The researcher makes use of systematic literature mapping (SLM) and systematic literature review (SLR) principles.	The researcher recommends that companies who want to enhance their web security should review the insights and information gathered from this comprehensive review, which will assist them in the implementation of effective security testing practices. The majority of online application security testing studies concentrate on certain vulnerabilities like SQL injection and Cross-Site Scripting (XSS), which are serious security issues for web applications.	The researcher stated that future works need to be conducted between industry and academia with these results identified. There is also a need to evaluate and refine the evidence on security testing of web applications and empirical evaluation of efficiency and effectiveness. The researcher also strongly emphasises there is a need to have thorough systematic literature reviews that will assess and compare the strengths and limitations of the found vulnerabilities especially XXS and SQL injection.
A Systematic Review of AI-Based Soft. Test Case Optimization.	This study aims to evaluate the efficacy of maximising software test cases as a vulnerability	Comprehensive literature review which gathers and analyses existing information in AI-based	The study provided a thorough analysis of the most recent advancements and patterns in	The researcher identified a gap that needs to be explored in the future by combining

Int. Research Jnl. of Multidisciplinary Scope. (Padmanabhan, 2024)	detection tool in real-time systems.	software for maximising test cases.	AI-based software testing for real-time systems. The researcher emphasises the necessity for more research in this area as well as the urgency of AI in enhancing software testing performance.	various artificial intelligences, exploring their strengths in addressing fuzzing challenges, continuous progress and embracing fuzzing technology in order to maximise test cases.
Reviewing Soft. Testing Models and Optimization Tech.: An Analy. of Efficiency and Adva. Needs. Jnl of Computers, Mech. and Mgmt.. (Kumar, 2023)	The objective of this research is to provide a review of several optimisation techniques and software testing models. The researcher highlights the advantages and disadvantages and, at the same time, suggests enhancements to improve their efficiency and effectiveness.	Detailed literature review in the area of software testing models and optimisation techniques	The research shows that there is no one-size-fits-all method for software testing; instead, the model and procedures are dependent on the needs, resources, and limitations of the project.	There were no gaps identified in this paper
A review paper: optimal test cases for regression testing using AI techniques. Int. Jnl. of Elect. and Comp. Eng. (Khaleel & Anan, 2023)	The objective of this research is to identify various techniques used in the selection of test cases and determining the precedence of test cases. Secondly, identify the metrics used to evaluate their efficiency by using different techniques of artificial intelligence.	Regression testing and the methods for choosing the best test cases were the subjects of the survey. A comprehensive literature review was conducted to compare all the techniques.	Execution time and the number of error coverage are used as metrics to evaluate the test cases, and the APFD metric is frequently used to determine the priority for test cases. The primary method for deciding the order of test cases is the number of error coverage and time, which is followed by the quantity of code coverage.	There were no gaps identified in this paper.
Generative AI in Soft. Eng.: Revolutionizing Test Case Generation and Validat. Tech. Iconic Research and Eng. Jnl. (Thakur *et al.*, 2023)	The goal of this research is to determine AI benefits in different industries.	The survey was done in various industries such as banking, e-commerce, automobile, healthcare, and telecommunications in terms of AI benefits.	AI enables organisations to cope with software development expectations by automating regression testing, increasing test coverage, and giving the expected prediction. To eliminate flaws in the testing results, quality data must be used in developing AI models.	There were no gaps identified in this paper.

Comparative review of the features of auto. soft. testing tools. Int. Jnl. of Elect. and Comp. Eng. (Gamido & Gamido, 2019)	The aim of this study is to analyse many aspects of automated testing tools such as TestComplete, Selenium, Ranorex, QTP/UFT, Sahi, Watir, and SoapUI.	A systematic review of the literature and a survey of practitioners are the methods employed. A comparative analysis of the features of automated software testing tools was conducted in light of the findings.	Open-source software, like Selenium, is a preferable choice if the project has a tight budget. If they need support, simplicity of learning, and report generating, license tools like QTP/UFT are a better choice.	A study on other automated tools, including their response time need to be conducted
Pioneering Testing Tech. Advancing Soft. Quality Through Innovative Method. and Frameworks. Jnl. of AI and ML in Mgmt. (Syafiq Rahman & Farah Nadia, 2024)	The aim of this study is to analyse the current testing technologies and how they impact the software quality.	Thorough assessment of the literature and a comparison of both conventional and innovative testing techniques.	This researcher has given a comprehensive analysis of research, illustrating valuable knowledge into its historical background, ongoing developments, and future implications.	To provide a deeper knowledge, more thorough investigation into particular subtopics is required. Future studies must increase the quantity and calibre of data available.
Key factors & features influencing selection of open-source functional test automation tools. Int. Jnl. of Recent Tech and Eng. (Saravanan & Balakrishnan, 2019)	Determining the percentage of manual versus automated testing is the aim of this study. The second step is to enumerate the main elements impacting the use of open source for functional testing with automation tools. Lastly, to use automation technologies to determine the main factors driving the choice of Open Source for functional testing.	The characteristics and elements influencing the choice of open source, useful test automation tools were empirically evaluated. Indian IT companies were given the questionnaire.	The majority of IT companies are implementing both automated and manual testing. The most popular open-source functional test automation tool is called Selenium. Coding proficiency, setup simplicity, and tutorial accessibility are the top three criteria affecting the choice of open source, useful test automation technologies.	There were no gaps identified in this paper.
Toward a Multi-Criteria Framework for Selecting Soft. Test. Tools. IEEE Access. (Abdulwareth & Al-Shargabi, 2021)	The goal of this research is to create a thorough taxonomy of testing tools that addresses a variety of testing tool requirements. To choose the appropriate instruments, a taxonomy is introduced using a scientific technique. Decision-making is done using multiple	The weighted importance of various criteria is measured using a literature review and the Analytic Hierarchy Process (AHP). All relevant tools are then ranked using TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) to identify the best. A	The taxonomy of the testing instruments and the selection process have been thoroughly examined in the suggested framework. According to the specialists, the taxonomy is thorough, significant, correct, and practical. It assists developers in selecting the most	It is possible to simplify taxonomy. It is advised that developers and scholars widely validate the taxonomy. It is advised to do an experimental investigation since it can accurately depict the suggested taxonomy.

	criteria. This approach can then be developed as an engine to create a web service that helps adopt certain software or the appropriate testing tools.	team of scholars from several Saudi Arabian and Yemeni universities validates the suggested taxonomy approach.	effective software testing tools. But according to the majority of specialists, the taxonomy is a little confusing and challenging to use.	
Technological Innov. in Auto. Test.: A Detailed Exam. of their Influence on Soft. Dev. Efficiency, Quality Assurance, and the CI/CD Pipeline. Applied Research in Artificial Intelligence and Cloud Computing. (Vo Thi Lan, 2024)	This paper's goal is to conduct a thorough analysis of automation testing technological advancements, with a particular emphasis on the importance of these advancements for software development efficiency, quality assurance, and the Continuous Integration/Continuous Deployment (CI/CD) pipeline.	Literature Review	Software development, testing and maintenance are becoming efficient due to the improvement in scripting techniques, integration of AI, machine learning, regression testing, cloud-based solutions and API testing.	There are no gaps identified by the researcher, but the researcher mentioned that the automation testing will further improve as there are improvements in machine learning and AI. Therefore, the latest research must be ongoing to find the latest trends and innovations in automation testing.
Auto. Test. with ML Frameworks: A Critical Analy. Eng. Proceedings MDPI. (Fatima *et al.*, 2022)	This study aims to examine software automation-related machine learning (ML) frameworks.	Literature Review and case study on TechM's analytics tool.	In conclusion, using AI in software testing can assist in faster project delivery, and programmers can quickly discover errors.	All of the existing work in the fields of software testing and machine learning needs to be examined and categorised.
Trend Appl. of ML in Test Case Prioritization: A Review on Tech. IEEE Access. (Khatibsyarbini *et al.*, 2021)	This study's goal is to thoroughly examine machine learning (ML) methods using the research questions listed below. 1. What are the taxonomies of ML techniques in TCP (test case prioritisation)? 2. What are the differences in terms of approaches for each ML technique in TCP? 3. What are the processes involved in the ML technique in TCP?	The methodology used is a combination of research questions and literature review. There are 110 primary studies that were analysed, which consist of 58 journals, 50 conference papers and 2 other articles.	Despite the popularity of machine learning techniques in recent years, there are still certain aspects that require development, such the duration of the learning process. Due to its high APFD (average percentage fault detected) and coverage efficacy, the classification strategy in machine learning was the most often used method. This was because it profited from the availability of historical data. Lastly, a more systematic procedure was needed for the	Future work suggested by researchers are as below: 1) An approach that facilitates an agile development environment, both supervised and unsupervised. 2) More research on TCP is required to determine whether it is required for projects of a given scale. 3) For objective outcome findings, the clustering technique in machine learning needs to be improved.

	4. What is the state-of-the-art evaluation method used for ML techniques in TCP?		deployment of the reinforcement learning technique.	
Accelerating Soft. Quality: Unleashing the Power of Gen. AI for Auto. Test-Case Gen. and Bug Identification. Int. Jnl. for Research in Applied Sci. and Eng.Tech. (Bajaj & Samal, 2023)	The goal of this study is to determine the advantages and difficulties of applying generative AI to real-world test-case creation and bug detection applications.	Case study on real-world application	This study aims to identify the benefits and challenges of using generative AI in practical test-case development and bug detection applications.	There are no gaps identified by the researcher, however, the researcher highlighted there are ongoing improvements in AI technologies that will create robust and reliable systems.
Comparison of Soft. Test Auto. Tools-Selenium and UFT. American International Jnl. of Research in Formal, Applied & Natural Sci. AIJRFANS. (Karthik & Vk, 2019)	Comparing Selenium and UFT using various metrics is the aim of this study.	Comparative study between two automation test tools: Selenium and UFT (Unified Functional Testing).	Since the Selenium tool is open source, it is less expensive to install, but it involves more work to write scripts than UFT, which is better for data security but costs money and requires an annual license. .	There are no gaps identified by the researcher
AI in Soft.Test.: A Systematic Review. IEEE Region 10 Annual Int. Conference, Proceedings/TENCON. (Islam *et al.*, 2023)	The goals of this research: initially, to identify whether the manual testing has limitations. Second, can the shortcomings of manual testing be addressed by incorporating AI into software testing? Thirdly, what kinds of software testing tasks might AI automate? Lastly, how do researchers evaluate AI strategies in software testing?	To identify the existing work, research problems, improvement scope, and existing empirical studies, a systematic study was carried out. Ninety articles in all were taken from different research libraries. For the final review, filter to 20 publications based on PRISMA recommendations. To investigate the application of AI in software testing, about 50 articles were examined.	According to the study, a number of testing tasks, including those involving Machine Learning (ML) and Deep Learning (DL), including Test Case Generation, Defect Prediction, Test Case Prioritisation, Metamorphic Testing, Android Testing, Test Case Validation, and White Box Testing, can be effectively automated with AI.	One drawback is that the researcher only looked at a small number of trials. To obtain deeper insights, it would be beneficial to study a greater number of papers.

VI. FUTURE DIRECTION AND PRACTICAL RECOMMENDATION

This study will go beyond the literature review by conducting a case study with a real-world SME application to close the gap between innovation and real-world implementation. To assess how well the suggested AI-driven testing framework improves testing accuracy, efficiency, and cost-effectiveness, it will be implemented and evaluated against a real SME software system. This real-world application will address the actual software testing difficulties SMEs face and offer empirical insights into the framework's viability. Adoption of AI-driven testing remains challenging, particularly for SMEs, due to challenges such as high implementation costs, steep learning curves, and a lack of AI expertise within the team. To solve these challenges, it is necessary to deliberately integrate AI-driven testing frameworks that are lightweight, inexpensive, and simple to integrate into existing software development processes.

In addition, the case study will compare the AI-driven testing framework with conventional manual testing techniques, evaluating important performance metrics, including resource usage, test execution time, and defect detection rate. The outcomes will provide concrete proof of the useful benefits and possible drawbacks of AI-driven testing in a small and medium-sized business setting. This project intends to give SMEs a scalable and affordable testing solution by validating the framework in an actual environment, ultimately assisting small and medium-sized businesses in implementing AI-driven software testing.

From a practical standpoint, SMEs should be urged to embrace progressive AI integration strategies, which involve introducing AI-driven testing frameworks gradually as opposed to entirely overhauling their systems. Before implementing AI widely, this method lowers risk and enables companies to evaluate how it may affect their development and testing processes. Additionally, by guaranteeing dependability, security, and compliance in AI-driven testing solutions, the creation of standardised AI testing benchmarks and regulatory frameworks would benefit SMEs even more. Several AI-driven testing frameworks use AI technologies to increase the accuracy and efficiency of testing. Among these frameworks are:

1. AI-Based Test Case Generation Frameworks – These frameworks improve test coverage by using machine learning algorithms to create optimised test cases.
2. Self-Healing Test Automation Frameworks – These frameworks use AI to automatically modify test scripts in response to UI changes, minimising maintenance effort.
3. Reinforcement Learning-Based Test Optimization – This method dynamically ranks test cases according to risk analysis and past defect trends.
4. AI-Powered Visual Testing Frameworks – These frameworks use computer vision to identify platform-specific UI inconsistencies.
5. NLP-Driven Test Script Generation – This allows non-technical testers to participate in automation initiatives by converting human-readable test scenarios into automated test scripts.
6. Predictive Defect Analysis Frameworks – To help teams concentrate on high-risk areas, use AI-based data to forecast possible software flaws.

The incorporation of AI-driven framework into software testing presents significant challenges, particularly for SMEs, due to factors such as high implementation costs, steep learning curves, and a lack of AI knowledge within the teams. To bridge this gap, it is essential to adopt AI-driven testing frameworks that are affordable, lightweight, and simple to incorporate into current software development processes. These frameworks improve test automation, defect detection, and predictive analytics by utilising AI techniques like reinforcement learning, natural language processing (NLP), machine learning, and computer vision.

Successful integration of AI-driven frameworks into real-world testing environments requires consideration of several important strategies:

1. Standardized Integration with DevOps and CI/CD Pipelines – Incorporating AI-driven testing within DevOps processes ensures seamless regression testing, defect tracking, and test execution. Real-time quality assurance is made possible, and productivity is increased by incorporating AI-based testing tools into Continuous Integration/Continuous Deployment (CI/CD) pipelines.
2. Addressing Data Quality and AI Model Training – High-quality datasets for model training are essential

to the efficacy of AI-driven testing. The accuracy and dependability of AI models in identifying software flaws are improved by ensuring that test data is representative, diversified, and properly labelled.

3. Enhancing Explainability and Trust in AI Testing – To boost developer and tester confidence, AI-based testing frameworks need to produce results that are clear and easy to understand. Debugging test failures and comprehending AI-driven test decisions are made easier with Explainable AI (XAI) approaches.
4. Scalability and Customization for Different Software Architectures – To provide adaptation to diverse development contexts, AI-driven testing frameworks should support a range of software architectures, such as cloud-based apps, embedded systems, and microservices.
5. Industry-Specific Best Practices and Compliance Alignment – To guarantee compliance and security in software development, AI-based testing must abide by industry standards (such as ISO/IEC 29119 for software testing and GDPR for data protection).
6. Upskilling and Change Management for Engineering Teams – Organisations must offer upskilling initiatives and training programs to software developers and testers to assist them comprehend AI-driven testing approaches in order to promote AI adoption.
7. Continuous Monitoring and Feedback Mechanisms – Continuous improvement in software quality and testing procedures is ensured by implementing AI-powered analytics for real-time test monitoring and feedback.

VII. CONCLUSION

In today's rapidly evolving software landscape, integrating AI-driven automation technologies into software testing is a necessity. In this review paper, the most popular automation tools have been examined, 20 academic journals have been examined, and significant research gaps in AI-driven software testing have been discovered. Even though prior studies have shown that AI-powered testing is accurate, efficient, and cost-effective, issues like tool adoption, standardisation, and integration into SMEs still need to be investigated further in order to determine the best testing framework for organisational requirements. To effectively utilise AI's promise to improve software quality and speed up development cycles, these gaps must be filled. Future studies should concentrate on creating a comprehensive framework for AI-driven software testing that strikes a compromise between scalability, affordability, and accessibility. By bridging the gap between theoretical advancements and real-world application, AI-driven software testing has the potential to revolutionise the sector.

VIII. REFERENCES

Abdulwareth, AJ & Al-Shargabi, AA 2021, 'Toward a Multi-Criteria Framework for Selecting Software Testing Tools', IEEE Access, 9, pp. 158872–158891. doi: 10.1109/ACCESS.2021.3128071.

Albarka Umar, M & Zhanfang, C 2019, 'A Study of Automated Software Testing: Automation Tools and Frameworks', International Journal of Computer Science Engineering [Preprint].

Alferidah, SK & Ahmed, S 2020, 'Automated Software Testing Tools', 2020 International Conference on Computing and Information Technology, ICCIT 2020, pp. 183–186. doi: 10.1109/ICCIT-144147971.2020.9213735.

Ateşoğulları, D & Mishra, A 2020, 'Automation Testing Tools: a Comparative View', International Journal on Information Technologies & Security, vol. 12, pp. 63–76, retrieve from < https://www.researchgate.net/publication/346109409>.

Aydos, M et al. 2022, 'Security testing of web applications: A systematic mapping of the literature', Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 9, pp. 6775–6792. doi: 10.1016/j.jksuci.2021.09.018.

Bajaj, Y & Samal, MK 2023, 'Accelerating Software Quality: Unleashing the Power of Generative AI for Automated Test-Case Generation and Bug Identification', International Journal for Research in Applied Science and Engineering Technology, vol. 11, no. 7, pp. 345–350. doi: 10.22214/ijraset.2023.54628.

Dhaya Sindhu Battina 2019, 'Artificial Intelligence in

Software Test Automation: A Systematic Literature Review', Journal of Emerging Technologies and Innovative Research [Preprint].

Europe Italy, S & Eastern Europe, P 2024, World Quality Report, retrieve from <www.worldqualityreport.com>.

Fatima, S et al. 2022, 'Automated Testing with Machine Learning Frameworks: A Critical Analysis', Engineering Proceedings MDPI. doi: 10.3390/engproc2022020012.

Gamido, HV & Gamido, MV 2019, 'Comparative review of the features of automated software testing tools', International Journal of Electrical and Computer Engineering, vol. 9, no. 5, pp. 4473–4478. doi: 10.11591/ijece.v9i5.pp4473-4478.

Garousi, V, Joy, N & Keleş, AB 2024, 'AI-powered test automation tools: A systematic review and empirical evaluation', retrieve from <http://arxiv.org/abs/2409.00411>.

George Murazvu, Simon Parkinson, SK 2024, 'A Survey on Factors Preventing the Adoption of Automated Software Testing : A Principal COmponent Analysis Approach', MDPI [Preprint].

Haeggström, M 2024, Hands-on Use and Adaptation of AI in Developing and Testing Software Applications.

Islam, M et al. 2023, 'Artificial Intelligence in Software Testing: A Systematic Review', IEEE Region 10 Annual International Conference, Proceedings/TENCON, December, pp. 524–529. doi: 10.1109/TENCON58879.2023.10322349.

Izzat, S & Saleem, NN 2023, 'Software Testing Techniques and Tools: A Review', Journal of Education and Science, vol. 32, no. 2, pp. 31–40. doi: 10.33899/edusj.2023.137480.1305.

Karthik, M & Vk, J 2019, 'Comparison of Software Test Automation Tools-Selenium and UFT', American International Journal of Research in Formal, Applied & Natural Sciences AIJRFANS, pp. 19–212, retrieve from <http://www.iasir.net>.

Khaleel, SI & Anan, R 2023, 'A review paper: optimal test cases for regression testing using artificial intelligent techniques', International Journal of Electrical and Computer Engineering, vol. 13, no. 2, pp. 1803–1816. doi: 10.11591/ijece.v13i2.pp1803-1816.

Khankhoje, R 2024, 'AI in Test Automation: Overcoming Challenges, Embracing Imperatives', International Journal on Soft Computing, Artificial Intelligence and Applications, vol. 13, no. 1, pp. 01–10. doi: 10.5121/ijscai.2024.13101.

Khatibsyarbini, M et al. 2021, 'Trend Application of Machine Learning in Test Case Prioritization: A Review on Techniques', IEEE Access, 9, pp. 166262–166282. doi: 10.1109/ACCESS.2021.3135508.

Khin Shn Thant & Hlaing Htake Khaung Tin 2023, 'The Impact Of Manual And Automatic Testing On Software Testing Efficiency And Effectiveness', Indian Journal of Science and Research [Preprint].

Kumar, S 2023, 'Reviewing Software Testing Models and Optimization Techniques: An Analysis of Efficiency and Advancement Needs', Journal of Computers, Mechanical and Management, pp. 32–46. doi: 10.57159/gadl.jcmm.2.1.23041.

Leotta, M et al. 2024, 'An empirical study to compare three web test automation approaches: NLP-based, programmable, and capture&replay', Journal of Software: Evolution and Process, vol. 36, no. 5, pp. 1–24. doi: 10.1002/smr.2606.

Lima, R, Da Cruz, AMR & Ribeiro, J 2020, 'Artificial Intelligence Applied to Software Testing: A Literature Review', Iberian Conference on Information Systems and Technologies, CISTI, 2020-June(June), pp. 24–27. doi: 10.23919/CISTI49556.2020.9141124.

Mukherjee, R & Patnaik, KS 2021, 'A survey on different approaches for software test case prioritization', Journal of King Saud University - Computer and Information Sciences, vol. 33, no. 9, pp. 1041–1054. doi: 10.1016/j.jksuci.2018.09.005.

Padmanabhan, M 2024, 'A Systematic Review of AI Based Software Test Case Optimization', International Research Journal of Multidisciplinary Scope, vol. 5, no. 4, pp. 847–859. doi: 10.47857/irjms.2024.v05i04.01451.

Prathyusha Nama 2024, 'Integrating AI in testing automation: Enhancing test coverage and predictive analysis for improved software quality', World Journal of Advanced Engineering Technology and Sciences, vol. 13, no. 1, pp. 769–782. doi: 10.30574/wjaets.2024.13.1.0486.

Raju, SS & Leong, WY 2025a, 'AI-Driven Automation in Software Testing: Enabling SME Adoption', INTI Journal, vol. 2025, no. 1.

Raju, SS & Leong, WY 2025b, 'AI-Modernizing Testing: A Comparative Review of Test Automation Frameworks and AI Tools', INTI JOURNAL, vol. 2025, no. 10, pp. 1-9.

Salahirad, A, Gay, G & Mohammadi, E 2023, 'Mapping the structure and evolution of software testing research over the past three decades', Journal of Systems and Software, vol. 195, p. 111518. doi: 10.1016/j.jss.2022.111518.

Samli, R 2023, 'A Comprehensive Overview of Web-Based

Automated Testing Tools', Journal of Advanced Engineering Studies and Technologies, vol. 4, no. 1, pp. 13–28.

Sani, B & Jan, S 2024, 'Empirical Analysis of Widely Used Website Automated Testing Tools', EAI Endorsed Transactions on AI and Robotics, vol. 3, pp. 1–11. doi: 10.4108/airo.7285.

Saravanan, K & Balakrishnan, SK 2019, 'Key factors & features influencing selection of open source functional test automation tools', International Journal of Recent Technology and Engineering, vol. 8(2 Special Issue 10), pp. 840–843. doi: 10.35940/ijrte.B1153.0982S1019.

Syafiq Rahman & Farah Nadia 2024, 'Pioneering Testing Technologies Advancing Software Quality Through Innovative Methodologies and Frameworks', Journal of Artificial Intelligence and Machine Learning in Management [Preprint].

Thakur, D et al. 2023, 'Generative AI in Software Engineering: Revolutionizing Test Case Generation and Validation Techniques', Iconic Research and Engineering Journals [Preprint].

Verma, AS, Choudhary, A & Tiwari, S 2023, 'Software Test Case Generation Tools and Techniques: A Review', International Journal of Mathematical, Engineering and Management Sciences, vol. 8, no. 2, pp. 293–315. doi: 10.33889/IJMEMS.2023.8.1.018.

Vo Thi Lan 2024, 'Technological Innovations in Automation Testing : A Detailed Examination of their Influence on Software Development Efficiency, Quality Assurance, and The Continuous Integration/Continuous Deployment (CI/CD) Pipeline', Applied Research in Artificial Intelligence and Cloud Computing [Preprint].