International Journal of Current Engineering and Technology
©2023 INPRESSCO®, All Rights Reserved

E-ISSN 2277 – 4106, P-ISSN 2347 – 5161
Available at http://inpressco.com/category/ijcet

*Research Article*

# Future-Proofing QA: Key Trends and Directions in Automation Testing

**Krishna Gandhi[1*] and Pankaj Verma[2]**

[1,2]Independent researcher, India

Received 26 Nov 2023, Accepted 05 Dec 2023, Available online 15 Dec 2023, **Vol.13, No.6 (Nov/Dec 2023)**

**_Abstract_**

*The creation and security of software systems in the internet domain have relied heavily on IT software testing, even if quality assurance technologies and professionals have proliferated. Due to the complexity and difficulty of testing, professionals have turned to test automation technologies, which entail several steps and ultimately require more oversight and administration. This paper discusses the current developments in QA automation, looking at its significance, issues with the process, and prospects. Quality assurance and software testing are defined to clarify why these two concepts are intertwined but separate. The progress made in the last few years, namely the AI in testing, cloud-based solutions, and no-code/low-code automation, are reviewed to determine how it affects testing efficiency in identifying defects. Also, the paper describes the significance of QA, the software quality, the advantages of automation tools, and the limitations that need to be considered for the accomplishment of objectives. Future directions indicate the use of AI/ML together with robotic process automation to improve predictive analysis, test scalability, and effectiveness. The goal of this research is to offer practical recommendations and best practices to companies willing to improve their QA performance and meet the changing needs of the software industry.*

***Keywords:*** *Software Testing, Quality Assurance (QA), QA Automation, Code/No-Code Platforms, DevOps, Testing Frameworks, Software Quality.*

## Introduction

Softwares are today part of and involved in nearly everything, driving industries, business and even day to day activities. There has never been a greater need for reliable calculations that are accurate and efficient and software to support those calculations. Due to the constantly developing technologies in different societies, it becomes important to maintain the quality and performance of these systems. Software testing as a process that is critical in the software development life cycle helps assert confidence that the software is as hoped, functional, and user-friendly. But when software complexity grew up and it size increases then traditional testing approach faces the problem and it becomes essential to adopt new testing approaches.

However, manually performing tests has disadvantages, like being very slow, repetitive and affected by human error and cannot meet today's requirements of practices such as Agile and DevOps. In order to eliminate these drawbacks, QA automation has become the new way of performing software testing [1]. Thus, apart from increasing the efficacy of the operation by eliminating monotonous and time-consuming testing exercises, QA automation minimizes errors. Automated test scripts make test cases easy to repeat or modify, and in doing so, the recurrent and reliable consistency of test coverage across multiple development cycles makes integrated continuous delivery an effective tool for teams facing strict deadlines.

In addition to the technical advantages, QA automation promotes synergies of development teams to include business analysts, developers, and DevOps engineers [2]. This guarantees its use in every phase of software production that defines the quality standards which are pivotal to the development of efficient, reliable and scalable software solutions. The fact that there are many different testing tools for different languages and platforms presents teams with flexibility and lets them focus on the challenges that may have for their particular projects [3].

QA methods should change through the years as a result of technology advancement, in order to be more effective. As previously established, improving quality assurance for the future means incorporating new trendy concepts such as AI and ML for testing [4]. These advancements make the generation of better tests possible, the usage of new approaches to testing, and the identification of defects to make certain that QA approaches are commensurate with the continual evolution and complexity of today's application systems. Through such approaches, organizations will

*Corresponding author's ORCID ID: 0000-0000-0000-0000
DOI: https://doi.org/10.14741/ijcet/v.13.6.8

foster synthesis of solid software systems that are capable of performing existing and future tasks.

## Problem Statement

QA plays a critical role in delivering dependable applications in contemporary software development process. On the other hand, in the agile and DevOps environment, ongoing testing is mandatory and usually, traditional QA approaches face challenges such as time limits and increased system complexity. QA activities with QA staff that are heavily template-based or partially automated tend to lag behind development and slow down the development process or allow QA issues to surface in production instead of during development. This is aggravated by the absence of scalability, flexibility, and timely addressing of emerging problems. This puts significant pressure on improving the efficiency of the quality assurance systems and for the new strategies that would apply such technologies in advancing QA techniques like artificial intelligence automation, probabilistic analytics, and self-healing functions. Solving it brings about enhanced software quality and cost reduction coupled with the desired usability in a very competitive and changing environment.

## Structure of the paper

The structure of this paper is as follows: Section II discusses the quality assurance in software development with its importance, recent trends in automation testing and quality assurance provided in Section III. Section IV present the Future directions for QA automation, previous research are provide in Section V. Conclusion with potential future study are provide in Section VI.

## Quality Assurance and Software Testing

Maintaining a focus on product quality throughout development has become more important in the cutthroat software industry. Ensuring that software satisfies the expectations of stakeholders and users is the responsibility of QA. In Malaysia, where the software industry is thriving, optimizing processes and implementing effective QA strategies have become essential for businesses to stay competitive [5]. By adhering to best practices and industry standards, companies can deliver top-notch software solutions that meet customer expectations and stand out in the market. To guarantee that the software goods or services offered to clients are of high quality, a process called software testing (or QA testing) is carried out. In order to guarantee satisfied customers, it's necessary to test software at many levels of quality:

**Functional testing:** Does the software program meet all of the functional needs and requirements? They will shortly get into the topic of testing automation, but there is also the option of doing tests manually.

**Non-functional testing:** Does the software program live up to consumers' expectations in terms of non-functional characteristics, such as dependability, efficiency, and usability? Manual testing of this would be challenging.

**Statistical testing:** Discover the program's dependability and the impact of flawed programs on operational circumstances using statistical methodologies.

In every stage of the software development lifecycle, quality assurance is vital. In order to be competitive in the highly competitive market of today, businesses must prevent the harm to their brand that comes from releasing substandard software.

## Difference between Quality Assurance V/S Testing

There is a common misconception that quality assurance and software testing mean the same thing. Software testing involves looking for mistakes and faults in previously produced code, while quality assurance aims to avoid such problems from happening in the first place. Testing is a product-oriented activity aimed at detecting and fixing defects in the software by executing it under various conditions to validate functionality and reliability.

**Table I** present the differences. Difference between QA and Software Testing

| Aspect | Quality Assurance (QA) | Testing |
|---|---|---|
| Definition | Ensures software meets standards and customer expectations through systematic processes. | Identifies bugs, defects, or errors by executing the system or application. |
| Primary Focus | Preventing defects. | Identifying defects. |
| Scope | Covers the entire software development process. | Limited to testing phases post-development. |
| Key Activities | Establishing standards, managing environments, and continuous process improvement. | Designing test cases, executing tests, and identifying software defects. |
| Objective | Prevent defects through improved processes and system checks before testing begins. | Detect and resolve defects after the product is developed. |
| Approach | Proactive – focuses on eliminating errors before it occur. | Reactive – focuses on finding and addressing errors introduced into the product. |
| Team Involvement | Involves all team members across various roles. | Primarily the testing team's responsibility, with occasional involvement from others. |
| Output | Ensures quality throughout the software lifecycle. | Produces a product that meets quality standards as per testing outcomes. |
| Tools Used | Process monitoring, | Test case management, |

| | | |
|---|---|---|
| | audit management, and compliance management tools. | automation, defect tracking, and performance testing tools. |
| Lifecycle Phase | Spans the entire software development lifecycle from requirements to post-deployment. | Conducted after development is completed, during the testing phase. |

## Importance of Software Quality Assurance

Maintaining the software's ease of maintenance and modification by present and future developers is crucial and facilitated by SQA.

**Customer Satisfaction:** SQA is a key component in making sure software meets user expectations and demands. Organizations may enhance customer satisfaction, trust, and loyalty via the regular delivery of high-quality software.

**Compliance and Standards:** SQA checks that your program follows all rules and laws as well as quality standards. A dedication to quality is shown when the program follows certain requirements. Concurrently, it streamlines the certification process and is useful for businesses in certain sectors.

**Scalability and Maintainability:** Software systems benefit from SQA procedures when they are scalable and easy to maintain. Coding standards, code reviews, and adequate documentation are ways in which organizations may build software. As needs change in the commercial world, this software may be quickly updated, modified, and expanded. This has the potential to improve risk mitigation via more efficient use of time and resources, less costly code, and more teamwork and communication.

## Recent Trends in Automation Testing and Quality Assurance

Automation testing and quality assurance (QA) have undergone significant evolution in recent years due to the rapid advancements in technology and the growing demand for faster, more reliable software delivery [6].

**Continuous Integration Testing:** This process involves testing in tiny increments in an environment that mimics production, with code integration occurring often. Early issue detection, change effectiveness measurement, and end-user expectation satisfaction may all be achieved with this form of testing.

**Higher Automation Levels:** Testing teams are attempting to use automation wherever they can because of agile testing teams, an increasing number of TCoEs, and the intense demand to shorten time-to-market. This applies not just to regression testing but also to load and unit testing.

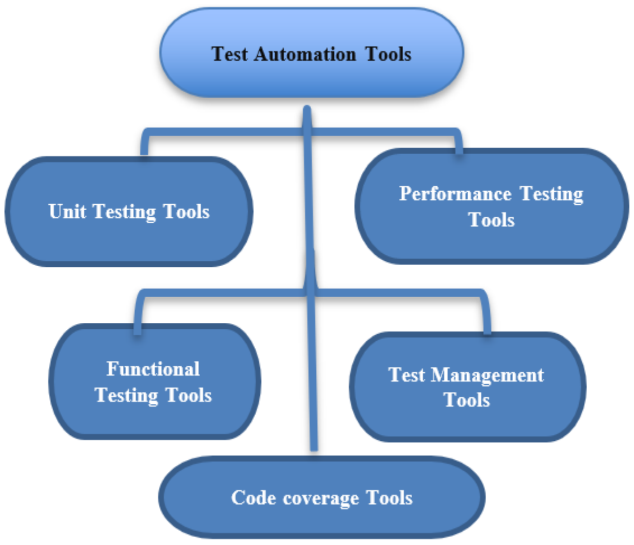
**Cloud-Based Testing:** The usage of cloud-based testing has been on the rise, according to IT experts and professionals, as cloud computing becomes more commonplace in the IT industry. In 2015, over 26% of software applications were hosted on the cloud. For the obvious reason that, in contrast to other test environments, cloud infrastructure provides a hassle-free and cost-effective option [7]. Businesses may use a dynamic testing approach that involves scaling up or down with cloud testing because of its flexibility.

**AI and Machine Learning in Test Automation:** AI-driven test creation leverages advanced algorithms to automatically generate test cases by analyzing user behavior patterns and historical defect data, ensuring comprehensive test coverage with minimal manual intervention. Self-healing scripts, powered by ML, dynamically adapt to changes in the application's user interface or functionality, thereby reducing the maintenance overhead traditionally associated with automated test scripts.

**Testing in Agile Development:** Many businesses are making strides towards a more comprehensive testing strategy that can integrate with agile development and make use of the most appropriate testing resources [8]. Continuous testing will become more common as businesses strive to enter the delivery phase as quickly as possible. Agile methodologies are now widely used. It provide both obstacles and possibilities to testers. Agile refers to a set of closely connected concepts rather than a single concept.

## Automation Tools Categories

There are a number of ways to classify software testing automation tools: Performance Testing Tools, Code Coverage Tools, Unit Testing Tools, and Test Management Tools [9].



Categories of Test Automation Tools

## Unit Testing Tools

Unit testing is a technique that helps test smaller, more fundamental pieces of code. One tool that is used for this purpose is the Unit Testing tool. When it comes to automating tests, unit testing tools are by far the most popular and convenient option. It also happens to be readily integrated as a framework inside development environments like NetBeans. Unit testing tools are used to verify the proper operation of individual units (or methods), examine code structure, and guarantee proper adherence to programming best practices.

## Code Coverage Tools

Code coverage tools are a kind of testing instrument that may be used to ascertain whether sections of code are encompassed by automated testing. An important indicator for understanding the quality of QA activities is code coverage testing. As test cases are being produced, code coverage tools point out parts of the code that may not have been tested enough and need further testing. It reveals how much of the software code isn't covered by automated tests and is, therefore, susceptible to bugs. Code coverage tools such as Cobertura, Code Cover, EMMA, PI Test, Atlassian Clover, and more will be available shortly.

## Test Management Tools

Many different types of test management solutions exist, each with its own unique set of capabilities and approaches to test administration. However, it often provides the chance to simplify the testing process, provide quick access to data analysis, and provide easy communication across multiple project groups [10]. Tools for managing tests include Test Manager, Test Link, TET ware, Test Environment Toolkit (TET), QA Complete, and soon.

## Unified Functional Testing

In addition to other automated test tools, there is Unified Functional Testing. More importantly, this tool's AI-based object detection is its most vital functionality. Furthermore, Continuous Integration is also available for integration. Programming in VBScript underpins this Windows-based application.

## Performance Testing Tools

The purpose of performance testing tools is to facilitate performance testing. The purpose of performance testing is to find out how stable and responsive the program is under different situations and loads. The purpose of performance testing tools is to measure how well a program or component works in relation to predefined performance criteria, such as resource consumption, throughput, and stimulus-response time. Several tools are available for performance testing, including JMeter, Rational Performance Tester, HP LoadRunner, Silk Performer, and many more.

## Benefits and Limitations

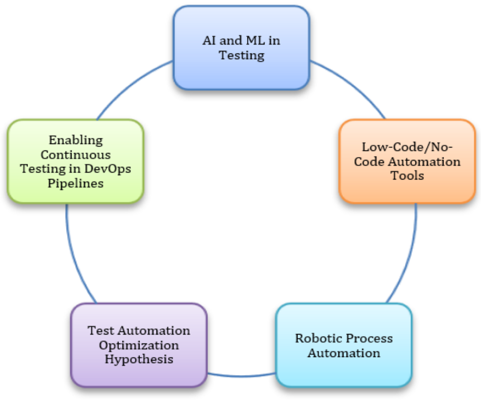
They find that the technique yields advantages when these factors, which impact software testing automation, are balanced. Some benefits of automation include greater product quality, test coverage, decreased testing time, dependability, increased confidence, test reusability, less human work, lower long-term costs, better ROI, and increased fault detection. When testing anything that requires deep subject expertise, nothing beats manual testing. Some potential drawbacks of automated software testing include difficulties in maintenance, a lack of skilled individuals, and enough time to reach maturity.

**Table 2** Advantages and Disadvantages of Automated Software Testing

| Advantages | Disadvantages |
|---|---|
| The results are more precise, and the errors are found faster than with manual testing. | Devotion, time, and a strategy for change are necessary for tool selection. |
| There is a repeatable script for the automation test. | Ability to build scripts for automation tests |
| Streamlines testing, saving time, and reduces effort | Proficiency with the testing instrument is expected. |
| Use of several testing tools simultaneously enables concurrent testing of various test scenarios, which increases test coverage. | A small fortune goes into purchasing the testing gear and, for playback approaches, test upkeep. |

## Future Directions for Qa Automation

Future Directions for QA Automation refers to the evolving trends and advancements in the field of quality assurance automation. This involves the exploration of new tools, methodologies, and technologies that can improve the efficiency, accuracy, and scalability of automated testing processes. This includes improving data quality and availability, developing more transparent AI models, and creating scalable frameworks that can be adapted to diverse software environments. Additionally, as AI technologies evolve, integrating them with agile and DevOps methodologies will be crucial for maintaining the effectiveness of QA automation in fast-paced development cycles. Efforts to streamline the implementation and reduce costs associated with AI-driven testing solutions will also play a critical role in their broader adoption. Here are some key future directions for QA automation that can help shape the next generation of software testing:


AI and ML in Testing
Low-Code/No-Code Automation Tools
Robotic Process Automation
Test Automation Optimization Hypothesis
Enabling Continuous Testing in DevOps Pipelines

Future directions for QA

**AI and ML in Testing:** The integration of AI and ML is revolutionizing QA automation by enabling predictive analysis, intelligent test generation, and self-healing test scripts [11]. Smarter testing techniques were made possible by these technologies, which analyzed massive volumes of data to find trends and forecast possible faults.

**Low-Code/No-Code Automation Tools:** Low-code programming allows the developer to focus less on code syntax and more on the visual design and functioning of the app, which in turn reduces the developer's time spent fixing bugs and creating new features [12]. The development of user-friendly tools that enable non-technical users to create automated tests, democratizing QA automation.

**Robotic Process Automation:** The term "robots" used to describe software programs used in RPA refers to the automation of routine, rule-based processes that were once handled by people. These jobs require a lot of time, have a lot of potential for human mistake, and have very specific inputs and outputs [13]. RPA is the practice of using specialized software to automate business operations by simulating human interaction with an information system (IS).

**Test Automation Optimization Hypothesis:** Utilizing AI in QA automation will optimize test execution by dynamically prioritizing test cases based on risk factors and previous test outcomes, resulting in faster testing cycles and more efficient resource utilization.

**Enabling Continuous Testing in DevOps Pipelines:** Integrating AI with QA automation supports the goal of achieving continuous testing in DevOps environments. Automated testing may happen at any point in the software development lifecycle with the help of AI-powered solutions that can interact smoothly with CI/CD pipelines.

## Literature Review

The work recommends the superiority of test automation when multiple regression testing rounds are required, as well as the benefits of test software, including reusability, repeatability, and the reduction of effort in test executions. This research is the result of a systematic literature review and practitioner survey.

This study, Sivaji et al. (2020) evaluate and contrast the efficacy of open-source Smart Manufacturing toolkits in testing web applications. With respect to web application testing, scripting, verification, and maintenance, Katalon Studio 7.0 proved to be a more intelligent and efficient tool than Robot Framework 3.0. The null hypothesis of " statistical analysis revealed no discernible variations in the productivity rate of SMEs when it came to automating software tests using RF and KS " may be rejected with a p-value of less than 0.05, as shown by the results of a Wilcoxon Matched Pairs Signed-Rank Test. Similar research might be replicated in the future to help choose appropriate mobile manufacturing toolkits and increase efficiency [14].

This study, Chatterjee et al. (2018) provides a new approach to collaborative automation testing by directing the execution of various test scenarios towards a number of distant devices. The execution of test cases is only one of several advantages this framework has over manual testing. No human operators are needed to run each UE using the Centralised Test Automation Framework (CTAF) since all that is required is access to a web client. They use laboratory experiments to illustrate the performance of Their system. It is difficult to come up with workable testing methods to concurrently monitor and operate these devices. It is necessary to appoint a human tester to run each user's equipment in order to manually test numerous devices. [15].

This study, Contan, Dehelean, and Miclea (2018) reviews five software projects that adhere to the Agile methodology, which emphasises the need of test automation in ensuring high quality standards. Software engineers working on test automation should apply more critical thinking to the models advocated in literature, according to the paper's claimed results. The study also noted the need for other models that are more in line with current standards in the field and have stronger correlations with data collected from actual projects [16].

This study, Wang (2018) to investigate the elements that contribute to an advanced test automation process and the criteria for evaluating its degree of maturity. Software development processes are increasingly reliant on test automation. Despite its widespread use, many are not shocked to learn that a robust test automation process is still a ways off. Organizations must understand the elements that contribute to mature test automation and how to evaluate the present state of test automation maturity in order to determine improvement actions if they want to obtain or maintain test automation benefits and accomplish continuous improvement [17].

This study, Sneha and Malle (2018) intends to gain knowledge about software testing methods, tools, and methodologies, as well as to compare and contrast manual testing with automated testing. There has been an effect on testing from automation testing. Automation technologies have replaced human testers for the majority of software testing nowadays, cutting down on both the number of testers needed and the likelihood of human mistakes. The use of test cases in automation testing simplifies the process of capturing and storing various situations [10].

This study, Straub (2016) explores the topic of automated testing of additive manufacturing parts and quality assurance in this context. The steps involved in finding errors, assessing their severity, and maybe fixing them are covered. These goals are addressed by the algorithms that are offered. They take a look at several evaluation examples. This article covers the topic of fixing both accidental and purposeful mistakes [18].

Table III provides an overview of studies focusing on test automation across various domains, highlighting objectives, application areas, and future research directions. The studies emphasize enhancing automation maturity, addressing device testing challenges, and advancing defect correction in manufacturing systems. Future work explores broader adoption of tools, industry-aligned models, and innovative techniques to improve software quality and productivity.

**Table 3** Background Study on Test Automation Objectives, Application Domains, and Future Directions

| Study | Objective | Application Domain | Future Scope |
|---|---|---|---|
| Sivaji et al. (2020) | Comparison of productivity between Katalon Studio 7.0 and Robot Framework 3.0 for web application testing. | Web application testing. | Future studies could replicate findings for mobile manufacturing toolkits to enhance productivity. |
| Chatterjee et al. (2018) | Propose a collaborative automation testing framework for testing multiple remote devices simultaneously. | Collaborative and remote device testing. | Develop viable testing techniques to manage and monitor devices in real-time during automation. |
| Contan, Dehelean, and Miclea (2018) | Analysis of test automation in Agile software projects. | Agile software development and test automation. | Develop alternative models consistent with industry best practices and real-world applications. |
| Wang (2018) | Examine factors leading to mature test automation processes and assess maturity levels. | Test automation maturity in software development. | Organizations should implement steps to assess and improve their test automation maturity levels. |
| Sneha and Malle (2018) | Compare manual testing with automation testing. | Comparison of manual and automation testing in modern software development. | Explore advancements in automation tools for broader adoption in complex software testing processes. |
| Straub (2016) | Discuss automation of QA and defect correction in additive manufacturing systems. | Quality assurance and defect correction in additive manufacturing. | Extend research on automated defect correction and its integration with advanced manufacturing techniques. |

## Conclusion And Future Work

Automating software tests improves software quality and lowers software development expenses over time. This is because automation leads to a higher return on investment, more mature automated test creation, and experience-based debugging. With a program statement and certain parameters as input, a good testing tool should be able to execute as many program statements as feasible. Quality Assurance (QA) automation has become a cornerstone in modern software testing, addressing the inefficiencies of manual testing by delivering improved defect detection, scalability, and reduced testing timelines. It enhances software quality and ensures faster time-to-market, making it indispensable in today's dynamic development environments. Emerging technologies, such as AI-driven tools and low-code/no-code platforms, have further expanded the capabilities of automation, making it more accessible and efficient. However, challenges such as high initial setup costs, tool compatibility, and the need for skilled personnel persist, requiring organizations to adopt strategic approaches for successful implementation. Looking ahead, the integration of AI and ML into QA automation holds the potential to enable predictive analytics, autonomous testing, and self-healing scripts. These advancements promise to redefine testing practices, fostering greater adaptability and efficiency. QA automation, thus, stands as a strategic enabler, driving innovation and ensuring sustained competitiveness in software development.

## References

[1] D. Kumar and K. K. Mishra, "The Impacts of Test Automation on Software's Cost, Quality and Time to Market," in *Procedia Computer Science*, 2016. doi: 10.1016/j.procs.2016.03.003.

[2] R. Publication, "The Future of QA Automation: Trends and Predictions," *SSRN Electron. J.*, vol. 10, pp. 873–880, 2021.

[3] R. Bishukarma, "The Role of AI in Automated Testing and Monitoring in SaaS Environments," *Int. J. Res. Anal. Rev.*, vol. 8, no. 2, pp. 846–852, 2021.

[4] H. V. Gamido and M. V. Gamido, "Comparative review of the features of automated software testing tools," *Int. J. Electr. Comput. Eng.*, 2019, doi: 10.11591/ijece.v9i5.pp4473-4478.

[5] J. Reynolds, J. Kizito, N. Ezumah, P. Mangesho, E. Allen, and C. Chandler, "Quality assurance of qualitative research: A review of the discourse," *Heal. Res. Policy Syst.*, 2011, doi: 10.1186/1478-4505-9-43.

[6] R. Gopalakrishnan, N. Rajasekaran, and G. Eswaramoorthi, "A Study of Emerging Trends in Software Testing and Quality Assurance," *Int. J. Eng. Manag. Res. Page Number*, no. 1, pp. 376–380, 2021.

[7] S. Poulding and H. Waeselynck, "Adding contextual guidance to the automated search for probabilistic test profiles," in *Proceedings - IEEE 7th International Conference on Software Testing, Verification and Validation, ICST 2014*, 2014. doi: 10.1109/ICST.2014.42.

[8] M. Katara and A. Kervinen, "Making model-based testing more agile: A use case driven approach," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2007. doi: 10.1007/978-3-540-70889-6_17.

[9] M. A. Umar and C. Zhanfang, "A Study of Automated Software Testing : Automation Tools and Frameworks," *Int. J. Comput. Sci. Eng.*, 2019.

[10] K. Sneha and G. M. Malle, "Research on software testing techniques and software automation testing tools," in

*2017 International Conference on Energy, Communication, Data Analytics and Soft Computing, ICECDS 2017*, 2018. doi: 10.1109/ICECDS.2017.8389562.

[11] V. S. Thokala, "Integrating Machine Learning into Web Applications for Personalized Content Delivery using Python," *Int. J. Curr. Eng. Technol.*, vol. 11, no. 06, 2021, doi: https://doi.org/10.14741/ijcet/v.11.6.9.

[12] R. Waszkowski, "Low-code platform for automating business processes in manufacturing," *IFAC-PapersOnLine*, vol. 52, no. 10, pp. 376–381, 2019, doi: 10.1016/j.ifacol.2019.10.060.

[13] Y. Ketkar and S. Gawade, "Effectiveness of Robotic Process Automation for data mining using UiPath," in *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*, 2021. doi: 10.1109/ICAIS50930.2021.9396024.

[14] A. Sivaji *et al.*, "Software Testing Automation: A Comparative Study on Productivity Rate of Open Source Automated Software Testing Tools for Smart Manufacturing," in *2020 IEEE Conference on Open Systems, ICOS 2020*, 2020. doi: 10.1109/ICOS50156.2020.9293650.

[15] A. Chatterjee, A. Bala, M. Shah, and A. H. Nagappa, "CTAF: Centralized Test Automation Framework for multiple remote devices using XMPP," in *INDICON 2018 - 15th IEEE India Council International Conference*, 2018. doi: 10.1109/INDICON45594.2018.8987182.

[16] A. Contan, C. Dehelean, and L. Miclea, "Test automation pyramid from theory to practice," in *2018 IEEE International Conference on Automation, Quality and Testing, Robotics, AQTR 2018 - THETA 21st Edition, Proceedings*, 2018. doi: 10.1109/AQTR.2018.8402699.

[17] Y. Wang, "Test Automation Maturity Assessment," in *Proceedings - 2018 IEEE 11th International Conference on Software Testing, Verification and Validation, ICST 2018*, 2018. doi: 10.1109/ICST.2018.00052.

[18] J. Straub, "Automated testing and quality assurance of 3D printing/3D printed hardware: Assessment for quality assurance and cybersecurity purposes," in *AUTOTESTCON (Proceedings)*, 2016. doi: 10.1109/AUTEST.2016.7589613.