EPH- International Journal of Science and Engineering
ISSN (Online): 2454-2016
Volume 07 Issue 02 June 2021

DOI:10.53555/ephijse.v7i2.278

AI-AUGMENTED TEST AUTOMATION: LEVERAGING SELENIUM, CUCUMBER, AND CYPRESS FOR SCALABLE TESTING

Varun Varma Sangaraju*

**Senior QA Engineer at Cognizant*

Corresponding Author

Abstract

Thanks to improved efficiency, adaptability, and scalability, artificial intelligence is revolutionizing test automation in modern software development. Reliable test scripts, high maintenance costs, and inadequate adaptability to fit frequent user interface changes are among the challenges traditional test automation meets. Overcoming these limitations is made possible in part by self-repairing scripts, automated test case generation, and anomaly detection mixed with AI-driven solutions. Important instruments for commercial software testing, such as Selenium, Cucumber, and Cypress, offer special benefits as well. Selenium's cross-browser compatibility, Cucumber's behavior-driven development (BDD) approach, and Cypress's quick execution skills define basic elements of effective test automation systems. Human maintenance of these systems can thus be demanding and prone to mistakes. Artificial intelligence enhances these tools via forecasts of test failures, autonomous modification of test scripts, and reduction of duplicate test cases, increasing efficiency and dependability. Artificial intelligence makes major contributions to test automation in the following areas: AI-driven test scripts independently change to fit changes in the user interface, hence lowering the test failures resulting from small changes.Artificial intelligence looks over past data to create best test cases, hence improving test coverage and efficiency. AI-driven analytics for anomaly detection expose unusual behavior in application performance, therefore enabling early identification of likely development cycle difficulties. AI-driven test automation offers improved accuracy and faster delivery in fields including pharmaceutical benefit management (PBM) and healthcare, where regulatory conformity and software reliability are vital. Combining artificial intelligence with Selenium, Cucumber, and Cypress helps companies to streamline deployment schedules, maximize test dependability, and simplify testing processes. AI is improving rather than replacing existing testing technologies to create more intelligent, self-sustaining test automation systems. AI-enhanced testing becomes a transforming solution for attaining scalable, strong, and intelligent test automation as companies strive for improved software quality and accelerated releases.

Keywords: *AI-Augmented Testing, Selenium, Cucumber, Cypress, Test Automation, Machine Learning in Testing, Self-Healing Scripts, Predictive Test Analytics, Healthcare Software Testing, Scalable Testing, Continuous Testing, Agile, DevOps, PBM Testing, Intelligent Test Orchestration.*

1. INTRODUCTION

Ensuring application quality by effective testing becomes more important in the present fast software development environment. Previously sufficient, conventional manual testing has become difficult to fit the rapid progress of modern initiatives. Test automation systems like Selenium, Cucumber, and Cypress have so been extensively applied. Still, even these powerful tools struggle to control the rising complexity and scope of modern uses. Artificial intelligence (AI) inclusion into test automation is transforming and increasing scalability, accuracy, and efficiency. This introduction looks at the development of test automation, the need of artificial intelligence in testing, and the ways in which AI enhances generally used frameworks such as Selenium, Cucumber, and Cypress. Ensuring application quality by efficient testing becomes ever more critical in the current quick software development scenario. When companies aim to accelerate product introductions, constraints of conventional manual testing methods become more obvious. Once enough to ensure software quality, manual testing is insufficient today considering the fast development of software programs, increasing complexity, and demand for speedy releases. Still, traditional models struggle to handle the expanding complexity and scope of modern applications even with their advantages. Large-scale business software, cross-platform compatibility requirements, and dynamic user interfaces all demand ever more complex and flexible testing approaches. By increasing scalability, accuracy, and efficiency artificial intelligence (AI) including test automation is revolutionizing software testing.

1.1 Evolution of Automaton Testing

As Agile and DevOps emerged as software development methods, it became clear that faster and more consistent testing tools were needed. This necessity led to the development of test automation systems including:
● One often used open-source tool that enables automated web application testing on several browsers and platforms is Selenium.
● By means of simple-language test scenarios, a behavior-driven development (BDD) tool for fostering cooperation among developers, testers, and business stakeholders helps to Designed for end-to--end testing of modern web applications, Cypress is a JavaScript framework with real-time debugging tools and fast execution capability.
● These systems have greatly improved the testing process, but they still need constant script maintenance; they are prone to faulty tests; and they may find it difficult to properly respond to dynamic application changes.
● The requirement of fast and consistent testing solutions became obvious as Agile and DevOps became the top software development approaches.
● Since they were excessively sluggish and prone to errors, conventional manual testing techniques were inadequate for the rapid iterations needed by contemporary software development.
● This demand resulted in the creation and extensive use of test automation tools, which have significantly improved scalability, accuracy, and output of software testing.

1.1.1 Main Tools for Automaton Testing

● Many automation testing techniques have fundamentally transformed the scene of software testing.
● Selenium, a highly used open-source automation technology, enables web application automatic testing across various browsers and platforms. Because it supports numerous programming languages, including Java, Python, and C#, it is adaptable for testers and developers.
● Designed especially for end-to--end testing of contemporary web applications, Cypress is a JavaScript-based testing framework with fast execution powers and real-time debugging tools. Cypress runs straight inside the browser gives faster testing speed and better control than Selenium.
● Usually used for unit and integration testing, JUnit and TestNG are Java-based testing systems with seamless interaction using build tools like Maven and Gradle with extensive reporting functionality.
Common for mobile apps, Appium provides a basis for automation supporting both iOS and Android systems. It allows testers to develop scripts in several programming languages executed on emulators and actual devices. A Behavior-Driven Development (BDD) tool facilitates developers, testers, and business stakeholders in more seamless cooperation by letting the construction of test scenarios in simple English. This helps to connect team members—technical and nontechnical both.

1.1.2 Automotive Testing Challenges

Though it offers numerous advantages, automation testing has several issues that must be addressed for best utilization. As systems expand, automation scripts must be constantly changed to fit changes in the user interface and capability.
● Lack of frequent maintenance causes test cases to become outdated, thereby producing erroneous test results. Depending on network slowness, synchronization problems, or changeable content, automated tests can reveal variances.
● Flaky test findings can lead to issues all through development. Among the dynamically changing components Web apps sometimes employ are pop-ups, AJAX-generated data, and variable IDs.
● Automation systems demand sophisticated techniques including implicit waits, explicit waits, and strong locators if we are to properly control such things. Automated tests included into Continuous Integration/Continuous Deployment (CI/CD) pipelines will help to optimize the advantages of test automation.
● Establishing and preserving test environments in CI/CD systems calls for both expertise and complexity. Good control of test data is prerequisite for effective application of automated testing.

●Effective test data configuration and remedial action plans determine both exact test execution and data discrepancies prevention.

1.1.2 The Development of Automation Testing

The field of test automation is continually changing because of creative technologies and methodology integration. Under development are artificial intelligence and machine learning in testing to autonomously fix test scripts, find trends in test failures, and enhance test runs. Test automation driven by artificial intelligence reduces maintenance requirements and raises automated test dependability. Increasingly widespread and let testers with no programming experience build and run automated test cases using visual interfaces are no-code and low-code automation solutions. Cloud-based testing solutions progressively allow the simultaneous running of automated tests over numerous environments, hence greatly lowering the test execution time.

●Early testing during the development cycle is given first importance in the shift-left approach to identify and correct problems immediately.

●The development process calls for adding automated testing methods to apply this strategy. Automation testing has transformed software quality assurance by letting quick, scalable, uniform testing procedures.

●As new technologies emerge, the future of automation testing will present rising test efficiency, the minimization of maintenance work, and top priority enhancement of software product quality.

1.2 The Argument for Artificial Intelligence in Automated Testing

1.2.1 The growing complexity of modern applications

Modern programs show great dynamism, defined by many integrations, complex processes, and regular user interface improvements. Sometimes conventional automation systems require major maintenance in reaction to UI changes, dynamic content, and complex user interactions. Increased testing efforts and resource allocation follow from this. Modern software programs exhibit unparalleled complexity defined by smart integration, multiple procedures, and consistent user interface (UI) improvements.

●The expansion of digital channels calls for comprehensive and adaptable testing tools. Many times, traditional automated testing systems cannot adapt with these fast developments.

●Typical test scripts can be greatly changed by user interface, dynamic content, and interactive components; so, regular changes and hand corrections are rather necessary.

●Maintaining traditional automation systems becomes resource-demanding as applications change since it wastes valuable time and effort more suitable for innovation and feature enhancement.

●Artificial intelligence systems notice and adjust to dynamic aspects, hence they promise consistent test execution independent of content changes over test runs.

To help test case development, artificial intelligence promises total test coverage, predictions of likely failure sites, and user behavior analysis. Fast and effective testing solutions are now rather required with adoption of agile approaches and pipelines for continuous integration/continuous deployment (CI/CD). AI-driven automation conveniently suits contemporary development methods, thereby supporting:

●Independent adjustment to meet changes in application behavior led by artificial intelligence helps to lower maintenance loads by itself.

●Based on historical test performance data, artificial intelligence ranks most among the tests; so, it helps to increase execution time efficiency.

●Basic testing across different devices, operating systems, and browsers guarantees consistent user experiences by use of AI-driven solutions.

● Artificial intelligence enhances reporting and analytics by producing abundance of data on defect patterns, test performance, and system stability.

1.3 The Need of Scalability and Efficiency in Testing

Test automation aims to reduce the time of testing while maintaining accuracy. Still, especially for large-scale projects, the hand authoring and maintenance of scripts remains a labor-intensive task. Companies seek a smart system that can automatically test and also adjust with little human participation.

1.3.1 Limitations of Conventional Automaton Systems

Notwithstanding their effectiveness, traditional test automation systems have certain natural limits including:
● Increased maintenance costs: Program changes call for consistent changes to test scripts.
● Self-heal test scripts by dynamically responding to UI changes, hence reducing script maintenance needs.
● How AI Improves Cucumbers, Cypress, and Selenium?
● AI enhances present test automation systems by including features like:
● AI-driven technologies could find UI changes and dynamically update locators, hence reducing test failures in Selenium and Cypress.
● AI-driven analytics help to identify high-risk test instances, hence supporting more strategic test prioritizing.
● AI can propose new test cases in Cucumber by evaluating prior test results, user interactions, and application logs.

1.4 Pragmatic Applications in the Business Context

Currently having a significant impact in companies by: AI-enhanced test automation is:
AI guarantees constant and dependable test execution throughout several application versions, hence improving regression testing. AI-driven automation speeds test execution and helps to enable faster release cycles, hence improving CI/CD pipelines. Self-healing scripts save time and money by relieving the need for constant script changes. Including artificial intelligence into test automation helps companies to improve test dependability, increase productivity, and speed software delivery. The next parts will look closely at the unique AI-driven features of Selenium, Cucumber, and Cypress, thereby providing ideas on how businesses may apply these developments for intelligent and scalable test automation.

As software applications become more sophisticated, the demand for intelligent, scalable, and efficient test automation has reached hitherto unheard-of heights. Although conventional automation systems—Selenium, Cucumber, Cypress—run afoul of dynamic user interface changes, are prone to erroneous tests, and need major maintenance—they have transformed software testing. Use artificial intelligence-driven test automation to offer faster, more intelligent, more powerful testing alternatives. This part examines how artificial intelligence is strengthening, optimizing, and enabling Selenium, Cucumber, and Cypress.

2. AI with Selenium: Improving Web Testing

2.1 Enhancing Selenium-Based Automation via AI

Online automated testing has traditionally preferred Selenium as the framework. As online applications progress, the issues with test script administration and execution performance also shift. AI-driven improvements in Selenium's efficiency are reducing test flakiness, raising element detection, and sharpening test execution techniques. With Selenium automation, managing constantly changing web components is a main challenge. AI-driven test automation solutions enable self-healing scripts that let AI autonomously identify UI changes and update element locators in real time.

This helps to totally eliminate the requirement for human script maintenance and significantly lowers test failures resulting from UI changes. Artificial intelligence systems using risk assessment could review past test data and rank test execution. By spotting test cases with increased probability of failure, artificial intelligence helps teams prioritize important tests and thereby promotes faster feedback cycles. Furthermore, led by artificial intelligence, test parallelization dynamically distributes test execution among several contexts, hence minimizing resource use and lowering test execution time. Analyzing their web-based patient management system, a reputable hospital applied Selenium automation driven by artificial intelligence. Their 60% test maintenance reduction came from self-healing scripts. By cutting the 6-hour duration of regression testing to 2-hour length led by artificial intelligence, faster product releases and better patient care were made possible.

2.2 AI powered by cucumbers: cognitive behavior-driven testing

2.2.1 How might artificial intelligence enhance BDD test cases?

● Cucumbers enable developers, testers, and business analysts to cooperate by means of behavior-driven development (BDD).
● AI-driven advances are bringing BDD automation by means of better test scenarios and simplified test case maintenance.
● Natural language processing (NLP) lets artificial intelligence automatically generate test cases from user stories or application requirements.

● Artificial intelligence analyzes prior test cases, application logs, and historical faults to propose new BDD scenarios, hence lowering the manual effort needed for test development. AI-driven analytics can evaluate past test performance data and rank scenarios with increased probability of revealing problems first.
● By means of dynamic modification of test coverage, artificial intelligence guarantees efficient test execution, preserving great software quality.

2.3 Cypress AI with Enhanced End-to-End Testing

2.3.1 Cypress AI-augmented visual regression testing

Cypress's rapid execution and real-time debugging capabilities render it the optimal tool for testing modern web applications. Artificial intelligence enhances Cypress through visual regression testing. AI-driven photo recognition guarantees pixel-perfect accuracy and may thus identify even the slightest UI changes undetected in conventional regression tests.

2.3.1 Predictive Failure Detection and Causal Analysis

In artificial intelligence, predictive analytics looks at test execution trends to identify flawed tests and predict errors before they even occur. Artificial intelligence finds most likely underlying reasons by analyzing past test failures, hence boosting test reliability and accelerating developer debugging.

2.3.2 Artificial intelligence based test analytics for mobile and online applications

Deep knowledge of test execution trends given by artificial intelligence-driven test analytics enables teams to enhance their testing strategy. By means of data consolidation from numerous test runs, artificial intelligence (AI) may identify recurrent failure sites, propose enhancements, and generate real-time dashboards for improved decision-making. A healthcare online company assessed Cypress automation improved by artificial intelligence against its appointment booking system. Artificial intelligence visual regression testing found UI differences across devices, so guaranteeing a flawless patient experience. Predictive failure analysis helped 80% of expected mistakes to be found and corrected before manufacturing, therefore enhancing system stability and user happiness.

3. Case Studies: AI in PBM and Healthcare Testing

Artificial intelligence (AI) is transforming many disciplines, most notably healthcare and pharmaceutical benefit management (PBM), software testing. These sectors' complicated procedures, rigorous compliance rules, and high transaction volumes help them to fit for AI-enhanced test automation. This section will review practical applications in PBM and healthcare testing in addition to required insights and best practices for artificial intelligence integration in corporate testing strategies.

3.1 Artificial Intelligence Applied in Pharmacy Benefit Management Testing

3.1.1 Problems with PBM Program Assessment

Pharmaceutical benefit management systems handle millions of daily pharmaceutical transactions and also provide regulatory compliance and control of complicated claim procedures. Conventional testing approaches find it difficult to fit:
Increased transaction volumes calling for quick validation.
Regulatory compliance evaluations that have to fit evolving policies (like HIPAA, CMS rules).
Working with several stakeholders—including healthcare providers, insurance companies, and pharmacies—allows one to.
Spotting and fixing differences in claim processing before client effect.

3.1.2 Improvement of Compliance Testing and Claims Processing Made Possible by AI

AI-driven test automation helps to minimize these challenges by:
● AI-generated test scripts recreate real-world pharmaceutical transactions to guarantee accuracy and compliance before deployment, therefore validating claims.
● AI constantly watches legislative changes and modifies test cases to ensure PBM systems remain compliant.
● AI assesses past test results to reduce non-essential test failures, hence improving test dependability.

3.2.3 Intelligent Test Orchestration for Large Volume Transactions

Dynamic test orchestration driven by artificial intelligence changes testing based on risk assessment and real-time system performance. This stands for:
● High-risk transactions get more thorough test coverage.
● Automated root cause analysis quickly finds problems in claims processing.
● AI stresses key test cases to ensure flawless operations under highest transaction volumes.
● Using artificial intelligence in PBM testing helps companies to achieve faster release cycles, improved compliance, and more accuracy in claim processing.

3.2 Use Case: Examining AI-Augmented Healthcare Applications
The Value of Automated Testing Within Electronic Health Records (EHR)
EHR systems protect and manage private patient data, so security and dependability become quite important. From patient history documentation to prescriptions and billing, healthcare providers rely on these platforms for a range. Traditional evaluation of EHR systems runs against challenges including:
● Different approaches used in different healthcare facilities.
● Regular software updates required by changes in regulations.
● Growing need for security and data accuracy.

3.2.1 AI enhances EHR testing using:
Automating regression testing ensures that newly implemented changes do not compromise current procedures. Finding anomalies in patient data: AI models spot differences that might point to data integrity issues. Improving test coverage: AI generates test scenarios created from real-world user interactions, therefore reducing the possibility of unnoticed mistakes.
Artificial intelligence automaton of workflow and patient data validation
AI-driven test automation is indispensable in:
● Through cross-referencing for correctness and guaranteeing data consistency across linked healthcare systems, artificial intelligence checks patient records.
● AI replicas interactions among EHR systems, pharmacies, and insurance companies to verify perfect data exchange.
● AI automates repetitive tasks such as data entry validation, therefore freeing testers to concentrate on more complex test cases by reducing manual work.

3.2.2 Healthcare Software as a Service Applications Enhanced by Artificial Intelligence
Performance testing is becoming critical as healthcare software moves toward cloud-based models. Artificial intelligence raises performance testing standards by:
● AI generates traffic patterns based on past use data, therefore simulating real user loads. Forecasting system failures.
● AI algorithms find trends in performance decline before their escalation into major issues. Improving resource allocation.
● AI instantly changes test settings to match expected workloads, therefore reducing the testing costs.
● AI-enhanced testing ensures that healthcare applications satisfy performance criteria even in the busiest times for patients and providers.

3.3 Realizations and Main Findings
Test Automation Driven by AI: Return on Investment
● Companies including artificial intelligence into test automation find measurable improvements such:
● A 50% drop in test execution time brought on the self-healing tests and automated script generation. 40% less defect leaking using predictive analytics driven by artificial intelligence. Significant savings from faster software delivery cycles and less manual testing operations.

3.3.1 Best Methods for Including AI into Corporate Testing Plans
Organizations which want effective AI integration into test automation have to:
Start with pilot projects: Evaluate artificial intelligence's impact in a limited implementation before organizational-wide growth. AI models rely on historical data; so, dependable test automation depends on keeping exact and diversified datasets. Integrate artificial intelligence with human expertise; testers should focus on complex edge situations and risk assessments since AI enhances testing but does not replace human discernment. Continually update AI models by means of constant feedback systems, hence optimizing AI algorithms for higher testing accuracy and efficiency.

3.3.2 Improvements in Practical Performance and Correcting of Errors
Case studies from notable PBM and healthcare companies show that test automation driven by artificial intelligence produces:
AI improves test execution efficiency, hence reducing time-to--market for necessary software changes. Self-repairing properties of artificial intelligence help to reduce defective tests, thereby ensuring constant test outcomes. AI-driven predictive testing finds early problems in manufacturing, therefore preventing costly post-release faults.

4. The Future Scope of AI in Automated Testing
As artificial intelligence gradually changes industries, its purpose in test automation is changing from simple enhancement. Autonomous automation that not only runs tests but also generates, improves, and adjusts them with minimum human interaction defines the direction of AI-driven testing. The upcoming phase will change team approaches on software quality, improving testing efficiency, flexibility, and integration inside DevOps approaches.

4.1 The Future Frontier: AI-Centered Automaton Test Development
From AI-Augmented to AI-First Testing

●AI has so far mostly served as a supporting tool for test automation, helping with pattern identification, test case development, and script optimization.
●The next step is AI-first testing systems, in which artificial intelligence independently creates and runs tests free from pre-written scripts.
●These systems will automatically create tests, instantly adapt to code changes, and absorb knowledge from past runs to improve performance over time.

4.1.1 Generative artificial intelligence for development and debugging of test scripts
A major step forward is the use of generative artificial intelligence into test script creation. AI-powered tools will:
●Create test scripts automatically using logs and application usage statistics.
●Analyze error trends to identify failing tests and provide fixes.
●Reducing reliance on human testers for script maintenance will help test automation to be more durable against UI and API changes.

4.1.2 AI- Improved Dynamic Test Execution
Whereas an AI-centric approach will prioritize and dynamically change tests in response to application changes, conventional test automation uses predefined test cases. Executing adaptive tests will:
●Identify areas of high risk within the application and direct attention on concentration testing.
●Examine past mistakes and instantly modify how tests are conducted.
●Improve test performance to minimize unnecessary runs by means of speed and efficacy.

4.2 How artificial intelligence affects DevOps and ongoing testing
Integration of Continuous Integration/Continuous Deployment Pipeline Improved by AI
Fast and consistent feedback systems define Continuous Integration and Continuous Deployment (CI/CD). Artificial intelligence is likely to change Continuous Integration/Continuous Deployment by:
●Expecting likely mistakes before they even happened.
●Automatically grading test cases based on most recent code changes.
●By use of historical test data analysis, minimising false positives and instability in test suites.

4.2.1 AI-driven automated risk-based testing
Artificial intelligence will enable risk-based testing, therefore emphasizing the most important parts of a system. This approach spans:
●Dynamic risk evaluation depending on code complexity, changes, and past mistakes.
●Running just the most relevant test cases to speed through cycles of testing.
●Hence saving time and money by avoiding unnecessary test runs.

4.2.2 Artificial intelligence driven strategies for quality engineering
Beyond basic test automation, quality engineering is pushing forward. Artificial intelligence will drive plans of the future by:
●Enabling shift-left testing, whereby artificial intelligence assesses initial requirements and suggests test paths before start of work.
●Using exploratory testing powered by artificial intelligence to find edge scenarios that traditional automation would miss.
●Artificial intelligence independently fixes malfunctioning test scripts in reaction to application upgrades, hence improving self-healing test automation.

4.3 Ethical Conundrums and Problems
Depending more and more in artificial intelligence for the field of mandatory tests
While artificial intelligence offers major production advantages, depending too much on it could lead to:
●Absence of human supervision increases the possibility of unnoticed mistakes.
●Artificial intelligence created test scripts lack the contextual validation given by human testers.
●Difficulties in identifying abnormal failure occurrence in AI-generated test cases

4.3.1 Privacy Issues Regarding AI-Driven Evaluation
Extensive datasets for maximum efficiency in AI-driven test automation beg questions about:
Test data could include private user records calling for strict data security measures.
Laws such GDPR and CCPA necessitate strong management of test data, which artificial intelligence-driven systems must follow.AI models created from biased data could produce false or unfair testing results.

4.3.2 Ensuring the clarity and openness of test automation powered by artificial intelligence
The complexity and opacity of artificial intelligence algorithms make understanding of test prioritizing and the underlying causes of particular failures difficult. Companies striving openness and confidence have to:

● Create understandable artificial intelligence models with easily available justifications for choices on test automation.
● Make sure human oversight helps to control artificial intelligence-driven automation. Verify regularly AI-generated test cases to guarantee fit with business goals.

5. Conclusion

Conventional test automation has repeatedly concentrated on minimising human labour in testing, uniformity, and efficiency. Still, its effectiveness has been limited by things like faulty tests, ongoing script maintenance, and inability to support dynamic UI changes. AI ranks test cases in order of risk assessment, past flaws, and real-time application performance, therefore improving efficiency. Machine learning techniques find trends in test failures and project possible flaws before they ever materialize. By including these features into current systems, artificial intelligence enhances Selenium's browser-based testing, Cucumber's BDD partnership, and Cypress's fast, developer-centric testing approach, hence increasing their intelligence and adaptability.

5.1 Techniques for Companies Using AI-Augmented Testing Frameworks

Integration of artificial intelligence into test automation does not call for a whole overhaul of present systems. Companies can increasingly use artificial intelligence to enhance their current test automation systems. Here is a manual for companies starting their operations:
Analyze current test automation maturity. Evaluate present systems of test automation and frameworks. Point up problems including inconsistent tests, too demanding maintenance, or extended running times.
Use intelligent automation technologies like Mabl, Applitools for visual testing, Testim for self-healing tests, or Mabl driven solutions. Improve tests using real-time artificial intelligence results.

5.2 Key Realizations for Improving Software Quality via AI-Driven Automation

AI is a strategic tool for ensuring software quality at scale, not only a test-automaton upgrade. Essential insights consist in:
Self-healing automation driven by artificial intelligence greatly reduces the need for script maintenance. AI improves behavior-driven development (BDD) by closely examining test cases and suggesting improvements.The future of artificial intelligence in test automation is bright, marked by several major trends just on their horizon:
AI will enable even non-technical individuals to create and maintain automated test cases with less effort, hence enhancing the use of low-code/no-code test automation. Improved integration of artificial intelligence into CI/CD pipelines: AI will enable the testing optimization in DevOps environments, therefore guaranteeing faster and more consistent software releases. Predictive analytics driven by artificial intelligence for proactive testing: machine learning will foresee issues before they arise, therefore improving the quality of software. Natural language processing (NLP) in test automation will enable AI to help create and understand tests in simple English, hence improving the usability of automation. AI-enhanced test automation is not a pipe dream; rather, it is changing the tools companies use for software testing. Using AI-driven testing approaches helps companies achieve higher software dependability, lower costs, and more productivity. Companies which welcome AI-driven automation today will be more strategically positioned to innovate and stay competitive going forward as AI develops. Companies should now look at test automation solutions driven by artificial intelligence and apply them into their software development process. Doing this would help companies reach improved scalability, efficiency, and quality—so heralding a new era of intelligent, automated testing.

6. References

1. Battina, Dhaya Sindhu. "AI-Augmented Automation for DevOps, a Model-Based Framework for Continuous Development in Cyber-Physical Systems." International Journal of Creative Research Thoughts (IJCRT), ISSN (2016): 2320-2882.
2. Marino, Daniel L., et al. "AI augmentation for trustworthy AI: Augmented robot teleoperation." 2020 13th International Conference on Human System Interaction (HSI). IEEE, 2020.
3. Defize, D. R. Developing a Maturity Model for AI-Augmented Data Management. MS thesis. University of Twente, 2020.
4. Hunt, Xin J., et al. "An ai-augmented lesion detection framework for liver metastases with model interpretability." arXiv preprint arXiv:1907.07713 (2019).
5. Xue, Yujia, et al. "Illumination coding meets uncertainty learning: toward reliable AI-augmented phase imaging." arXiv (2019).
6. Pavlou, Paul A. "Internet of things–will humans be replaced or augmented?." NIM Marketing Intelligence Review 10.2 (2018): 42-47.
7. Toulkeridou, Varvara. "Steps towards AI augmented parametric modeling systems for supporting design exploration." Blucher Design Proceedings 37 (2019): 81-92.
8. Carlile, Morgan, et al. "Deployment of artificial intelligence for radiographic diagnosis of COVID-19 pneumonia in the emergency department." JACEP Open 1.6 (2020): 1459-1464.
9. Shariff, Shahnaz Mohammedi. Investigating Selenium Usage Challenges and Reducing the Performance Overhead of Selenium-Based Load Tests. MS thesis. Queen's University (Canada), 2019.

10. Sadaj, Aleksander, et al. "Maintainability of Automatic Acceptance Tests for Web Applications—A Case Study Comparing Two Approaches to Organizing Code of Test Cases." SOFSEM 2020: Theory and Practice of Computer Science: 46th International Conference on Current Trends in Theory and Practice of Informatics, SOFSEM 2020, Limassol, Cyprus, January 20–24, 2020, Proceedings 46. Springer International Publishing, 2020.
11. Starov, O., and S. Vilkomir. "Cloud services and tools for mobile testing." Радіоелектронні і комп'ютерні системи 5 (2013): 362-371.
12. Starov, Oleksii, et al. "Testing-as-a-service for mobile applications: state-of-the-art survey." Dependability Problems of Complex Information Systems. Springer International Publishing, 2015.
13. Starov, Oleksii. Cloud platform for research crowdsourcing in mobile testing. East Carolina University, 2013.
14. Currie, Geoff, K. Elizabeth Hawk, and Eric M. Rohren. "Ethical principles for the application of artificial intelligence (AI) in nuclear medicine." European Journal of Nuclear Medicine and Molecular Imaging 47 (2020): 748-752.
15. Rouse, William B., and James C. Spohrer. "Automating versus augmenting intelligence." Journal of Enterprise Transformation 8.1-2 (2018): 1-21.