Automation Testing VS Manual Testing

Pradeep Kumar Medam

ITMD-536 / A20528923

10.11.2023

Nazneen Hashmi

Abstract

With the introduction of automated technologies in recent years, the landscape of software testing has undergone tremendous change. This study compares automation testing with manual testing critically, analysing the advantages, difficulties, and overall effects on the quality of the programme. Within the framework of both testing approaches, the study looks at things like test case coverage, time and cost efficiency, correctness, error detection, user experience, and employee satisfaction. This study clarifies the intricacies and trade-offs related to each strategy by providing insights into when and how automated and manual testing are most appropriate through a thorough investigation. Enhancing software development processes and end-user happiness, the findings have significance for practitioners and organisations looking to optimise their software testing procedures.

The introduction:

It is impossible to exaggerate the significance of software testing in the dynamic software business. High standards of quality, performance, and security are demanded of software programmes, which are ever more sophisticated. Two core software testing methodologies, automated testing, and manual testing have emerged as the leading competitors to satisfy these expectations.

The foundation for guaranteeing software quality is software testing. Making the decision between automated and manual testing isn't always simple, though. In manual testing, the judgement and intuition of human testers are used to comprehensively examine software applications; in automation testing, scripts and specialised tools are used to run tests effectively and methodically.

In order to offer insightful information on the advantages, disadvantages, and areas of application of each testing method, this research study conducts a thorough examination of both manual and automated testing. Through this analysis of these two testing approaches, readers will acquire a more profound comprehension of the variables impacting their selection and how to utilise them efficiently in various software development efforts.

Research Objective:

This research paper's main goal is to provide an extensive comparison of software development approaches for manual and automated testing. This study intends to offer useful insights that assist software development teams and organisations in making informed decisions when choosing the most appropriate testing methodology for their projects through a thorough analysis of each approach's benefits, drawbacks, and applicability in a variety of project scenarios. The study will combine qualitative and quantitative data, including case studies and industry surveys, to provide a comprehensive view of the benefits and drawbacks of automation and manual testing, ultimately contributing to the ongoing discussion about these critical aspects of software quality assurance.

Significance of Study:

In the software development business, where producing high-quality software is still of utmost importance, the research "Automation Testing vs. Manual Testing" is significant because it can address important issues and offer practical insights.

Above all, this study work provides a timely solution to the growing complexity of software systems. With the development of technology, software projects grow and complexity, necessitating extensive testing to find flaws and guarantee the end product's functioning, dependability, and security. The right decision can result in more project expenses, poor resource allocation, and a larger chance of problems remaining undetected, so knowing when and how to use automated versus manual testing is essential.

The results of this study might also have a big influence on organisations and software development teams. The study gives decision-makers the information they need to maximise their testing strategies by illuminating the advantages and disadvantages of both testing approaches. Project management, resource allocation, and the general effectiveness of software development processes are all directly impacted by this.

Also, by offering a research-based framework for thoughtful conversations and debates, the study deepens the current discourse in the software business. This article advances the collective knowledge of the software development community by thoroughly examining both automated and manual testing. This helps to improve the overall quality of software products given to end users and paves the path for better informed decision-making.

1. Automation Testing & Manual Testing

1.1 Automation Testing:

To conduct test cases with less manual intervention, automation testing uses automated scripts and specialised technologies. It is a useful approach in software quality assurance since it improves testing efficiency, accuracy, and repeatability.

Take an e-commerce company that refreshes its product catalogue on a regular basis, for instance. After each change, manually testing the entire website can be laborious and prone to mistakes. In this case, automation testing saves the day. It is possible to write test scripts that will automatically browse the website, add items to the cart, and finish the checkout procedure. Then, these scripts may run on many systems and browsers. The automated tool raises problems for evaluation if they occur. This procedure makes that the website continues to work properly following upgrades, saving time and effort compared to manual testing.

Below picture shows the user interface of an automation tool for an example automation testing script. A series of automated operations, including item selection, cart addition, discount application, card data verification, and expected result verification, are contained in the script. Reusing this script allows for a more effective and uniform testing strategy by evaluating the website in a variety of circumstances.

Automation testing is a crucial technique in contemporary software development since it greatly expedites the testing process, lowers the possibility of human mistake, and permits frequent regression testing.

1.2 Manual Testing:

A crucial software testing technique is manual testing, in which human testers assess programmes by hand to find errors, confirm features, and make sure they live up to user expectations. These testers should have subject expertise as well as a critical eye. Manual testing depends on human judgement, flexibility, and intuition as opposed to automated testing.

Manual testers follow user scenarios, run through a range of test cases, and investigate the programme from the viewpoint of the end user. This method works especially well in situations when the application's functionality is ill-defined, arbitrary, or subjective. In early stages of software development, when it may not be fully built yet, it is crucial for exploratory and usability testing. Testers can also see small faults that automated scripts might miss and swiftly adjust to changing needs by using manual testing. It also plays a crucial part in testing mobile apps and graphical user interfaces (GUIs), where usability, user experience, and visual components are vital.

Manual testing is still essential for software quality assurance even if it might be resource intensive. In order to provide thorough and user-centric assessments and guarantee that software products are dependable, easy to use, and defect-free, it supplements automated testing and is frequently included into the testing process.

1.3 Differences between the two:

1.3.1 By way of definition:

Automation Testing: Automated scripts and specialised tools are used to run test cases in automation testing. In order to automate the testing process, these scripts are designed to resemble human interactions with the programme.

Manual Testing: This method uses human testers to engage with the programme like an end-user would, carrying out test cases by hand. The application is explored and validated by testers using their intuition and subject expertise.

1.3.2 For Testing Purposes:

Automation Testing: High-volume, repetitious, and time-consuming testing jobs are the ideal candidates for automation testing. It works well in circumstances when repeating the same test cases is necessary, such as regression testing and load testing.

Manual Testing: In situations where human judgement, inventiveness, and flexibility are needed, such as exploratory testing and usability testing, manual testing is very useful. In the early phases of testing, when the software is not completely created and the test cases are not properly specified, it is frequently employed.

1.3.3 Detection Methods:

Automation Testing: Automated testing is based on pre-written test scripts that go through a sequence of operations and compare the outcomes to what is expected. By contrasting real results with the predetermined criteria, it finds flaws. Functional and performance problems can be methodically found using automation methods.

Manual Testing: This type of testing is done by human testers who look for errors in the application by directly seeing, investigating, and assessing its features, usability, and other elements. Testers employ their expertise to find problems—like UI glitches, unexpected actions, and subjective parts of the user experience—that automated scripts could miss.

1.4 Advantages and disadvantages of Automation Testing:

1.4.1 Advantages:

1) Efficiency: Automation testing enables the execution of several test cases in a comparatively short amount of time and is faster and more efficient for large-scale and repetitive testing jobs.

2) Repeatability: Regression testing requires that the same test scenarios be carried out precisely, and automated tests may be run consistently and frequently to ensure this.

3) Accuracy: Test results with fewer false positives and negatives are produced by automation systems, which execute tests precisely and without human mistake.

4) Parallel execution: It is appropriate for testing across numerous configurations and devices at the same time since it can run several test cases at the same time.

5) Comprehensive Testing: While load, performance, and stress testing might be difficult to carry out manually, automation enables comprehensive testing spanning a broad range of scenarios.

6) Logging and Reporting: Automation testing facilitates the identification of problems and the tracking of the execution of tests by offering comprehensive logs and reports.

Automation Testing Example:

Let's consider a simple scenario where we are automating a login functionality of a web application using a popular automation testing framework called Selenium (in Python). This script navigates to the login page, enters a username and password, and clicks the login button. It then verifies whether the user is successfully logged in.

```
from selenium import webdriver


# Initialize the Selenium WebDriver
driver = webdriver.Chrome()


# Navigate to the login page
driver.get("https://example.com/login")


# Find username and password fields, and the login button
username_field = driver.find_element_by_id("username")
password_field = driver.find_element_by_id("password")
```

```
login_button = driver.find_element_by_id("login-button")

# Enter login credentials
username_field.send_keys("your_username")
password_field.send_keys("your_password")

# Click the login button
login_button.click()

# Verify if login is successful by checking for a welcome message
welcome_message = driver.find_element_by_id("welcome-message")
assert "Welcome, User!" in welcome_message.text

# Close the browser
driver.quit()
```

1.4.2 Shortcomings of Automation Testing:

1) Initial Setup: The process of setting up automation tests involves a substantial upfront commitment of time and work. This includes writing test scripts and installing testing environments.

2) Cost: The cost of automation testing instruments can be high, and additional costs for maintenance and training are incurred.

3) Script Maintenance: It might take a lot of time and resources to maintain and update test scripts as software changes.

4) Low Adaptability: Because automation depends on pre-defined test cases, it is less flexible when it comes to exploratory or unscripted testing.

5) Complex programmes: Automating the testing of extremely complex programmes, particularly those with dynamic user interfaces, can be difficult.

6) Continuous Change: Continuous change might make automation scripts less useful in Agile development settings when software is updated often.

1.5 Advantages and disadvantages of Manual Testing:

1.5.1 Advantages:

1) Adaptability: Manual testing is quite flexible when it comes to growing software and changing needs. Real-time method adjustments allow testers to explore novel settings and form arbitrary conclusions.

2) Exploratory testing: This is the best method for exploratory testing since it enables testers to find unforeseen problems, assess user experience, and confirm software behaviour in unplanned scenarios.

3) Usability Testing: When evaluating an application's graphical user interface (GUI) and user experience (UX), manual testing performs exceptionally well.

4) Subjective Analysis: Although automated technologies could find it difficult to analyse, testers are able to analyse subjective aspects of the programme, such as visual design, content, and general user happiness.

5) Early Software Development: Since software is often changing and test cases may not be fully specified, manual testing is helpful in these early stages of software development.

6) Human Judgement: To assess complicated software behaviours, manual testing depends on human judgement, inventiveness, and intuition.

1.5.2 Shortcomings of Automation Testing:

1) Resource-intensive: Manual testing is less effective for repeated or high-volume testing activities since it might take a lot of time and labour.

2) Inconsistent Execution: It might be difficult to get consistent and repeatable findings when using human testers since they can bring unpredictability into the test process.

3) Restricted to Test Depth: Automation is more efficient in deep and comprehensive testing scenarios, such load or performance testing, where manual testing may not be appropriate.

4) Increased Human Error Risk: Human error can occur during manual testing, such as forgetting to check for flaws or committing mistakes when carrying out test cases.

5) Expensive: Manual testing can be expensive in terms of staff, training, and testing materials since it needs experienced human testers.

6) Slower Execution: Manual testing may not proceed at the necessary pace in situations when speed and repetitious execution are crucial.

Manual Testing Example:

In a manual testing scenario, a human tester follows the test cases and interacts with the application without automation. For example, in a manual login test:

1. The tester would open a web browser, navigate to the login page, and visually identify the username and password fields.
2. The tester would manually enter the username and password into the fields.
3. The tester would click the login button by physically interacting with the mouse or keyboard.
4. The tester would then visually inspect the page for a welcome message or other indications of a successful login.
5. If any issues are encountered, the tester would document them in a test report, including steps to reproduce and the observed behaviour.

2 Research Design

2.1 Automation Testing:

To evaluate the efficacy, efficiency, and application of automated testing in software, a systematic approach combining qualitative and quantitative approaches is used in the research design of Automation Testing.

Data Collection: A variety of sources, including industry case studies, software testing professional surveys, and actual instances of automated testing implementations, will be consulted throughout the course of this research. The benefits, difficulties, and best practises related to automated testing will be shown by this data.

Experimental Analysis: To assess the effectiveness of automated testing, controlled experiments may be carried out whereby automated test suites are contrasted with manual testing endeavours about duration, resource usage, and errors found rate. Metrics like execution durations, fault detection, and test coverage may be used in the study design.

Qualitative Data: To get comprehensive insights on the applicability of automation testing in different project settings, open-ended surveys and interviews with seasoned automation testers and project managers will be undertaken. These interviews will concentrate on the variables that impact the choice to employ automated testing as well as the difficulties encountered in putting automation into practise.

Case Studies: To investigate the practical effects of automated testing on software development processes, such as cost savings and enhanced software quality, the study design may incorporate in-depth case studies of companies that have successfully used the technology.

Data analysis: To make inferences and spot trends, quantitative data will be examined statistically, while qualitative data will be examined thematically.

2.2 Manual Testing:

The manual testing study design will place a strong emphasis on the adaptability and flexibility of manual testing techniques.

Case Studies: To comprehend how manual testing is used in complex, changing contexts, in-depth case studies of software development projects that significantly rely on manual testing will be carried out.

Expert Opinions: To get information on the decision-making procedures, difficulties, and best practises related to manual testing, surveys and interviews with seasoned manual testers will be undertaken.

Comparative Analysis: To pinpoint the situations in which manual testing really shines, a comparison of manual and automated testing will be conducted. This comparison will place special emphasis on the human judgement, flexibility, and exploratory skills of manual testers.

Qualitative Data: The subjectivity, flexibility, and inventiveness inherent in manual testing will be examined through open-ended survey questions and interviews.

Data Analysis: To produce findings and insights into the advantages and disadvantages of manual testing in various circumstances, data gathered from case studies and expert comments will be subjected to a thematic analysis.

2.3 Factors influencing choice of Automation Testing:

Repetitive Test Cases: Automation is the best option for test cases that must be run frequently, such in performance, load, and regression testing.

Big Test Suites: Automation may greatly expedite the testing process in projects that have large test suites covering a variety of situations.

Continuous Integration (CI): Automation smoothly blends into the development process in CI/CD systems, giving quick feedback on changes to the code.

Stability of Test Cases: Automation can produce consistent results over time if test cases stay stable and do not change frequently.

Resource Efficiency: Because automation eliminates the need for labour-intensive manual testing, it can result in long-term cost reductions for projects with limited funding.

Regular Software Updates: Automating code change validation helps minimise the need for human retesting in projects where software is updated on a regular basis.

Efficiency and Speed: Automation is far quicker and more effective than human testing, which makes it perfect for situations when quick test execution is required.

Complex Applications: Automation is more successful at managing the considerable testing needs of projects with complex, data-intensive, or large-scale applications.

2.4 Factors influencing choice of Manual Testing:

Changing needs: Manual testing enables testers to swiftly adjust to changing circumstances in projects where needs change regularly and fast.

Flexibility and adaptability: Because manual testing is so flexible and adaptable, it may be used in situations where there is a need for exploration or where test cases are not well specified.

Usability and Subjectivity: Human testing is superior at evaluating subjective aspects that automated testing could struggle to analyse, such as usability, user interfaces, and overall user experience.

Early-Stage Testing: Manual testing is useful for project development if requirements are not yet completely defined. It helps to validate changes rapidly and evaluate early software versions.

Exploratory Testing: Without manual testing, exploratory testing would not be possible. It gives testers the freedom to use their judgement, imagination, and intuition to find bugs and assess how software behaves in unplanned circumstances.

Human Judgement: Manual testing is favoured in situations that call for human judgement, such as evaluating the software's subjective features or formulating difficult choices based on the program's actions.

Cost-Effective for Small Projects: Because manual testing doesn't involve significant expenditures in automation tools, it could be a more economical option for smaller projects with tighter resources or budgets.

High Variability of Frequent Changes: Manual testing is less resource-intensive and more flexible than continuously updating automation scripts in projects where test data and software functionality change often.

Conclusion

In conclusion, selecting between manual and automated testing in software development is a strategic option that must consider the unique requirements and circumstances of a project rather than being a one-size-fits-all approach. A balanced viewpoint on automated vs. human testing is essential to ensuring the effective delivery of high-quality software, as both methodologies have clear benefits and drawbacks.

Automation testing is a great option for projects requiring continuous integration, huge test suites, and robust test cases because of its well-known efficiency, repeatability, and affordability. It is excellent in finding regressions quickly, maintaining consistency, and increasing the effectiveness of testing. Automation is made even more crucial in agile and DevOps contexts by its connection with CI/CD pipelines. a considerable cost savings and Automation is strongly supported by the possibility of thorough testing.

However, the strengths of manual testing include its flexibility, exploratory nature, and capacity to assess usability. It works best in projects with dynamic requirements, those in the early stages of development, those requiring subjective evaluation, and situations involving human judgement. Human testers can detect subtle flaws, evaluate the user experience, and quickly adjust to changing conditions. However, repeatability is a problem and manual testing can be resource intensive.

The best course of action turns out to be a well-rounded technique that makes use of both approaches' advantages to satisfy the many testing's needs of contemporary software development. Companies understand how important it is to combine the effectiveness of automation with the flexibility and human insights of manual testing. Ultimately, a careful analysis of project-specific variables, such as project needs, financial restrictions, application complexity, and release cycles, should inform the decision between automated and human testing. The harmony between automated and manual testing is evidence of the dedication to adaptation, pragmatism, and producing high-quality software in the fast-paced field of software development.

References

Automation Testing:

1) Automation Testing Made Easy: Automated Testing for Developers by John Scarpino (2013).

2) Selenium Testing Tools Cookbook by Unmesh Gundecha (2012).

3) Test Automation using Selenium WebDriver with Java: Step by Step Guide by Mr. Navneesh Garg (2013).

4) Mastering Selenium WebDriver by Mark Collin (2016).

5) Appium Essentials by Manoj Hans (2015).

Manual Testing:

1) Software Testing: A Craftsman's Approach by Paul C. Jorgensen (2002).

2) A Practitioner's Guide to Software Test Design by Lee Copeland (2004).

3) Exploratory Software Testing by James A. Whittaker (2009).

4) Testing Computer Software by Cem Kaner, Jack Falk, and Hung Q. Nguyen (1999).

5) User Interface Testing: A Software Engineering Perspective by Brian R. Marick (1997).

6) Agile Testing: A Practical Guide for Testers and Agile Teams by Lisa Crispin and Janet Gregory (2009).

7) Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation by Jez Humble and David Farley (2010).