Automated Testing Tools and Frameworks: Benefits, Pitfalls and Future Works

Amrita Ghosh[1], Aishwarya Gautam[2], Gauri Anil Godghase[3], Omkar Gholap[4], Pradnya S Kulkarni[5]
[1,2,3,4] *UG Students, School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India*
[5]*Assistant Professor, School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India*

***Abstract*—Software testing has long been an important element of the software development life cycle, assisting in analyzing limitations of new softwares. The emergence of automated testing can largely be attributed to the low efficiency and high human errors in manual testing. Recently, automated testing has made several strides in the field, through the means of various automated testing frameworks. The main objective of this paper is to provide a comprehensive analysis of these frameworks. It summarizes the different types of automation testing and highlights popular tools used in each type. Further, the various tools are compared based on essential parameters such as applicability, technical support and platform support etc. Finally, the paper also discusses the future scope of automated testing and what new possibilities can be explored to further optimize it.**

***Index Terms*— Wireframe Testing Tools, UI Testing Tools, Unit Testing Tools, Load Testing Tools, Integration Testing Tools, Automation Testing Tools, Automated Frameworks**

I. INTRODUCTION

Microsoft Windows, or as commonly known as Windows have been dominating the majority of the world's market share since its introduction from 1985. As time passed by, Microsoft made several improvements in the graphical user interface (GUI) of this operating system to launch the most recent version which is Windows 11.Increasing growth of the Windows operating system also led to the simultaneous development of many Windows applications. Windows applications use the graphical user interface through the controls like radio button, CheckBox, list item, text box, button etc. These controls are in turn given by Windows Forms. Typical Windows applications come in handy in every sphere of our lives from managing the workload through calendars, mails, meeting applications to entertainment purposes like shopping and media. As applications were being used by so many people worldwide, it became a necessity to ensure their quality. Thus, there was a need for proper testing of the applications to ensure that the customers have the best experience while using them. Testing is also required to guarantee that the applications are meeting their specified requirements and are bug-free.

Software testing is the practice of reviewing and validating that a software application accomplishes what it is designed to do. It helps us in preventing defects, lowering development costs and achieving productivity.

Software testing can be manual as well as automated. As the name suggests in manual testing, test scenarios are written and executed by the developer to see if the application is performing as mentioned in the requirement document or not. Whereas using automated software testing tools reduces the need for manual intervention in the testing process. This leads to a faster process with a more comprehensive test coverage. Automated testing can provide us with a higher application quality as well as performance. And finally, it can also make the process of quality assurance more cost-effective.

There are several steps involved when developing an application like generating an idea, designing, wireframing, coding, testing. And consequently, we need testing in each of the steps to detect errors as early as possible in the development cycle. We have several wireframe testing tools, UI/UX testing tools, unit testing tools, integration testing tools and performance testing tools available in the market today.

This paper is a review or comparative analysis among the most popular tools in each of the testing domains so that a user can select the best tool that he needs according to his purpose. We have done the comparison of the tools in accordance with the most prominent features like robustness, performance, scalability, testability, correctness and usability.

Table 1: Comparison of Wireframe Testing Tools

Features	Adobe XD	Sketch	Figma	Justinmind
Licensing cost	Trialware; Payment of $9.99 is required after a free trial period of 7 days.	Proprietary; Monthly payment of $9 per editor.	Proprietary; Free version only lets one create 3 Figma files and monthly payment is available for $12 per editor.	Proprietary; Available with a free version also but paid version requires a monthly payment of $9 per user.
Applicability	Can be used for web design, app design, brand design and game design.	Can be used for designing the UI and UX of mobile apps and web.	Allows for collaborating and designing digital products as well UI/UX design of apps and websites.	Can be used to create and design interactive prototypes and wireframes for web and mobile apps.
Platform support	Windows MAC	MAC only	Windows MAC UNIX LINUX	Windows MAC
User-friendliness	Very	Very	Very	Very
Drag and Drop components	Present	Present	Present	Present
Online Collaboration	Present	Present	Best	Present
Live Preview	Present on iOS only	Present	Present	N/A
Technical support	Best	Good	Good	Not much
Browser support	Can only be used through an app. No online version.	Can be used as a desktop application for apple computers only.	Can be used in-browser.	No web version available.

II. WIREFRAME TESTING TOOLS

The preliminary representation or outline design of the application under development is referred to as wireframe. It is usually represented using simple blocks, buttons and images. The idea behind wireframe testing is to facilitate incorporating missing requirements in the application early in the development cycle and to make the application more user-friendly. Table 1 provides comparison of various wireframe testing tools based on parameters such as licensing cost, applicability, platform support, browser support etc.

III. UI TESTING TOOLS

UI testing primarily entails testing the application's UI components and interface to ensure that they meet specifications and provide a smooth user experience. Testing the UI prototype manually could be tedious and time-consuming, and this is where UI testing tools come into the picture.

Table 2: Comparison of UI Testing Tools

Features	Selenium	Katalon Studio	TestComplete	Squish
Licensing cost	Open-Source	Katalon Studio: Free Katalon Studio Enterprise (for businesses and large-sized projects): $1,899/year	TestComplete Base: Starting at $6,519. Free trial available. TestComplete Pro: Starting at $10,150. Free trial available.	Squish Floating Tester Subscription: 499,17 € Squish Floating Execution Subscription: 82,50 € Free trial available.
Applicability	Used for various types of testing and is a very popular choice for GUI testing. It is easy to use, offers multiple features and a good community support.	Useful in UI testing for people with or without coding experience. It uses the page object model (POM) and has self-healing capabilities.	Easy to use UI testing tool by SmartBear that requires little or no programming knowledge. It provides the latest features such as object recognition.	GUI and regression testing tool that supports various features such as image-based testing, OCR, behavior-driven development, record and playback.
Platform support	Windows Linux macOS Solaris	Windows 7 or later macOS 10.11 or later Linux	Windows 7 or later	Windows 7 or later macOS 10.10 or later Linux
Supported languages	C# Java Python PHP Ruby	Groovy Java	JavaScript Python VBScript JScript DelphiScript	Python Perl JavaScript Tcl Ruby
Browser support	Google Chrome Mozilla Firefox Internet Explorer Microsoft Edge Opera Safari	Google Chrome Firefox Microsoft Edge Internet Explorer Safari	Google Chrome Firefox Microsoft Edge Internet Explorer	Google Chrome Firefox Microsoft Edge Internet Explorer Safari
Technical support	No official technical support	Only available at enterprise level	Good technical support	May need improvement

In today's competitive market, user experience is critical to a project's success, and so the demand for UI testing tools is rapidly increasing. There are abundant tools available, and this section compares them based on various factors. Table 2 provides comparison of various UI testing tools based on parameters such as licensing cost, applicability, platform support, browser support etc.

IV. UNIT TESTING TOOLS

Unit testing is a software development approach in which the smallest testable elements of a program, or units, are checked separately and independently for proper operation. This testing technique is used by software developers and, on occasion, QA staff during the development process. The main purpose of unit testing is to isolate written code in order to test and verify that it functions as expected.

Table 3: Comparison of Unit Testing Tools

Features	Devmate	Diffblue	Ponicode
Licensing cost	Single user license: €9/month/per user Corporate users: €49/month	For an individual developer: $175 For teams of 10: $186 For teams of 25: $175	Basic: Free Premium: 29 Euro Enterprise: 15000 Euro
Applicability	Used to automatically generate unit tests for a given set of class files	Used to quickly write tests for large legacy codebases, and identify untestable code that should be refactored	Used to Create, modify and visualize unit tests in seconds while boosting code coverage with our AI-based test generation.
Programming Language Support	C# Java	Java only	Javascript Typescript, Python.
Development Environment	VS Code Eclipse	IntelliJ	IntelliJ VS Code NPM Package
Community Support	Present	Present	Present
Platform support	Windows 8 or newer	Windows 10 Enterprise Ubuntu 18.04 RHEL 7.7 macOS 10.15	Windows Mac Os
Supported Test Frameworks	nUnit xUnit msTest JUnit	JUnit	N/A
Technical support	Good Technical Support	Good Technical Support	Good Technical Support

Unit testing is an important part of the development process because, if done correctly, it may help detect early bugs in code that would otherwise be difficult to find in later stages. Table 3 compares several Unit testing tools based on factors like license costs, applicability, platform compatibility, browser support, and so on.

V. LOAD TESTING TOOLS

Load testing is a sort of non-functional software testing in which the performance of a software application is assessed under a certain load. It regulates how the software program functions when several individuals use it at the same time.

Table 4: Comparison of Load Testing Tools

Features	LoadRunner	Boomq.io	Gatling	K6
Licensing cost	$1.40 per virtual user day	Standard $50 Pro $400	Scout: €69 Scale: €276 Corporate: €1380	Developer: $59 Team: $339 Pro: $1199
Applicability	More suitable for medium- to large-sized organizations.	used to accelerate testing cycles by a magnitude.	Used to write tests in SCALA	used in testing high-performance code, as well as businesses that are unable, for the moment, to organize independent testing.
Protocols	N/A	HTTP REST SOAP	HTTP(s)/1 JMS SOAP MQTT JDBC WebSockets	HTTP/1.1 HTTP/2 WebSockets gRPC
Plugins	N/A	Jmeter	Kafka, RabbitMQ, JDBC	Kafka, Datadog, InfluxDB, JSON and StatsD
Platform support	Windows, Linux macOS	N/A	Windows, MAC OS	Linux, MacOS Windows
Supported Programming Languages	ANSI C, Java .Net	Not Required	Java, Kotlin and Scala	Javascript
Technical support	Best	Not Much	Not Much	Good
Browser support	Internet Explorer Chrome, FireFox	Chrome Firefox	Firefox Chrome	Chrome Firefox

Prior to deployment, load testing is done to detect performance bottlenecks and ensure that software programs are stable and run smoothly. Table 4 compares several load testing tools based on factors like license costs, applicability, platform compatibility, browser support, and so on.

VI. INTEGRATION TESTING TOOLS

During the integration testing phase of software testing, individual software modules are connected and analyzed as a group. Integration testing is performed to determine whether a system or component meets specified functional criteria. This occurs between unit and system testing.

Table 5: Comparison of Integration Testing Tools

Features	LambdaTest	Endtest	Protractor	LDRA
Licensing cost	Proprietary; Monthly payment of $15 per user.	Proprietary; Monthly payment of $79 for 10 users.	Open-Source	Open-Source
Applicability	LambdaTest platform helps you to ensure your web app elements (such as JavaScript, CSS, HTML5, etc.) render seamlessly across every desktop and mobile web browser with support of manual, visual, and automated testing.	Can be used to create and run your automated tests in our cloud infrastructure and mobile device lab.	It is developed on top of WebDriverJS, which interacts with the application using native browser, specialized drivers.	LDRA is an open platform that allows integration tests to be built using the LDRA tool suite, as well as static and dynamic analysis across several platforms.
Platform support	Windows MAC IOS Android	Windows MAC LINUX UNIX	Windows MAC LINUX	Windows MAC LINUX UNIX
User-friendliness	Very Good	Very Good	Good	Good
Drag and Drop components	Present	Present	Present	Present
Technical support	Not much	Very Good	Good	Good
Browser support	Can be used in-browser	Can be used in-browser	Can be used in-browser.	Can be used in-browser

Integration testing takes unit-tested modules as input, groups them into larger aggregates, runs tests against those aggregates according to an integration test plan, and generates an integrated system suited for system testing as an output.

Table 5 compares several Integration testing solutions based on factors like license costs, applicability, platform compatibility, browser support, and so on.

VII. DISCUSSION ON AUTOMATION TESTING TOOLS

Selenium is one of the most popular web application testing tools. Selenium was developed in 2004 at ThoughtWorks in Chicago, with Jason Huggins for an internal Time and Expenses Application [1]. Since that time selenium has taken over other testing tools and become a popular choice of many. Surely the success of selenium could be attributed to many features that the tool provides, but there are also

some pitfalls to using this tool. The starting of this section will cover some of the advantages and disadvantages of using selenium.

Selenium is supported by a variety of web browsers such as Google Chrome, Mozilla Firefox, Internet Explorer, Microsoft Edge, Opera, Safari. Selenium WebDriver also supports cross-browser testing. Testers can see live automated tests being performed on their computer screens. This makes tracking the limitations of the software easy. Selenium supports a variety of languages and frameworks. Usually, for a developer language could be a hindrance as every developer has a preferred language, and this is where selenium works like a charm. Selenium supports languages such as C#, Java, Python, PHP, Ruby, Perl, JavaScript, and each of these languages has dedicated frameworks. A major advantage of using Selenium is its open-source availability. It is available for free to everyone, be it an enterprise or an individual. Selenium offers some advantageous features that contribute a lot to the popularity of the tool. Some of these features are regrouping and refactoring of test cases, parallel test execution with Selenium Grid, record and playback with Selenium IDE, ease of integration with other frameworks, less hardware requirement and many more.

A major drawback of selenium is that it only supports web application testing and does not work on testing windows applications or mobile application testing. Selenium relies on external tools for generating reports after test execution. Reports are a critical part of automation testing as it helps to analyze what went wrong and which part of the application needs improvement. But selenium does not have an in-built report generation feature. Selenium is an open-source tool with no official technical support available in case of any technical problems. However, as selenium has become quite popular it has a large community to back users up. Questions or queries can be posted on online forums, but no official customer support is available for selenium. Some other limitations of selenium are that it does not have a built-in object repository, recovery scenario is absent and image testing cannot be done using the tool.

Selenium is preferred by many, but that does not restrict one to stick to selenium, there are various other tools available in the market for test automation. Some of these tools are widely used and may provide a viable alternative to selenium. One such tool is Watir which was primarily developed by Bret Pettichord and Paul Rogers. Watir (Web Application Testing in Ruby, pronounced water), is an open-source (BSD) family of Ruby libraries for automating web browsers [2]. But every tool has its pros and cons, this section will further cover the advantages and disadvantages of Watir.

Watir is an open-source tool that is easy to use, and like many open-source tools available today, Watir also has a community to help people with technical challenges. Watir is supported by a wide range of browsers such as Google Chrome, Internet Explorer, Firefox, Safari, and Edge. It also supports cross-browser automation testing which is a bonus. Watir is essentially developed in Ruby language. Like other programming languages, Ruby gives the power to connect to databases, read data files and spreadsheets, export XML, and structure the code as reusable libraries[2].

In Spite of being quite popular as an easy-to-use tool, WATIR has a few disadvantages also. WATIR is compatible with the Ruby test framework only and testing of mobile browsers is not supported properly by it. Another major limitation of WATIR is that it can only be used for web-based applications and there is no feature available for recording so scripts can only be written manually.

An exhaustive comparative study of Selenium and WATIR tools can be found in the work of Gogna and Nisha[2]. One of the key differences pointed out between these two tools in this paper is the need to learn Ruby language for using WATIR whereas there is no such language barrier for Selenium. WATIR uses a separate library for each web browser. Though there is no recording feature available in WATIR as in Selenium, Frames and popups can be used easily with the help of API in WATIR while one may face trouble recording Frames and popups in Selenium.

VIII. AUTOMATED TESTING FRAMEWORKS

Various testing frameworks are designed to standardize the process of automated testing. One of them is Robot Framework [3]. Robot framework is prevailing, keyword driven and easily extendible. In this framework simple APIs are provided for building custom libraries in python and java. Robot

framework has high level architecture with simple and tabular syntax. It contains Generic as well as remote test libraries. Plugins for Jenkins and Maven are also available. High quality reports are generated containing detailed logs. However, no provision for debugging is present within the framework. Moreover, support of external libraries and third-party extensions is limited and often unstable. In addition, the Robot framework has several integrated development environment difficulties.

TestNG is another framework inspired from JUnit and NUnit with some additional functionalities. The comparison of JUnit and TestNG is provided in the work of [4]. TestNG and JUnit both support test annotations and test suite initialization. Both have an integrated development environment (IDE). However, only TestNG provides support for parallel testing and generation of test reports. In addition, JUnit does not support test groups and only has limited support for Parameterized Tests.

IX. CONCLUSION AND FUTURE WORKS

In the current scenario, manual testing is giving way to automation testing, and as time goes by, the dependence on automated work will grow. In this paper, we tried to condense and compare the various tools based on different types of testing. In the end, it could be concluded that the number of tools/frameworks available for web applications testing is far greater than that of windows applications testing. Therefore, for the future, we propose a framework using the latest technologies like PyCharm, Python, and Pywinauto for testing windows applications. The proposed tool will have a user-friendly interface and provide a detailed report after every test scenario execution. With our framework, we are trying to achieve the goal of creating a robust tool for testing windows applications for hassle-free user experience.

REFERENCES

[1] Brown, C. T., Gheorghiu, G., and Huggins, J., An introduction to testing web applications with twill and selenium., O'Reilly Media, Inc., 2007.

[2] Nisha Gogna, "Study of Browser Based Automated Test Tools WATIR and Selenium," International Journal of Information and Education Technology, vol. 4, pp. 336-339, August 2014.

[3] Neha S Batni and Jyoti Shetty, "A Comprehensive Study on Automation using Robot Framework," International Journal of Science and Research (IJSR), vol. 9, Issue 7, pp. 1033–36, July 2020.

[4] Manasi Patil and Mona Deshmukh, "Comparative Analysis of JUnit and TestNG framework," International Research Journal of Engineering and Technology (IRJET), vol. 05, May 2018.

[5] Dheeraj Kakaraparthy, "Overview and Analysis of Automated Testing Tools: Ranorex, Test Complete, Selenium," International Research Journal of Engineering and Technology (IRJET), vol. 04, Issue 10, Oct 2017.

[6] Milad Hanna, Amal Elsayed Aboutabl, and Mostafa-Sami M. Mostafa, "Automated Software Testing Framework for Web Applications," International Journal of Applied Engineering Research, vol. 13, pp. 9758-9767, 2018.

[7] Sarathy Venkatakrishnan and Chethana G., "Automated Testing - A Literature Review, " International Journal of Emerging Technologies and Innovative Research, vol. 07, Issue 6, pp. 670-681, June 2020.

[8] Devi, Jyoti and Bhatia, Kirti and Sharma, Rohini, "A Study on Functioning of Selenium Automation Testing Structure," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 07, Issue 05, pp. 855-862, May 2017.