T.C.
ISTANBUL UNIVERSITY-CERRAHPASA
INSTITUTE OF GRADUATE STUDIES

M.Sc. THESIS

EVALUATION AND COMPARISON OF WEB-BASED AUTOMATED TESTING TOOLS

MHD KHALED AL SAWAF

SUPERVISOR
Assoc. Prof. Dr. Zeynep ORMAN

[illegible]

Doç.Dr. Zeynep ORMAN
İ.Ü. Bilgisayar Mühendisliği
Bölümü

Department of Computer Engineering

Computer Engineering Programme

ISTANBUL- October, 2019

This study was accepted on 18.10.2019 as a M. Sc. thesis in Department of Computer Engineering, Computer Engineering Programme by the following Committee.

Examining Committee Members

[illegible]

Assoc.Prof.Dr. Zeynep ORMAN (Supervisor)
İstanbul University-Cerrahpaşa
Faculty of Engineering

[illegible]

Assoc.Prof.Dr. Rüya ŞAMLI
İstanbul University-Cerrahpaşa
Faculty of Engineering

[illegible]

Assist.Prof.Dr. Ceren ÇUBUKÇU
Maltepe University
Faculty of Engineering

FOREWORD

I want to thank God for completion in this work first, then my family and friends, in addition to the supervisor Assoc. Prof. Dr. Zeynep ORMAN, who helped me accomplish this task.

October 2019 MHD KHALED AL SAWAF

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

LIST OF SYMBOLS AND ABBREVIATIONS

Abbreviation	**Explanation**
SSL	: Secure Socket Layer
QA	: Quality Assurance
IDE	: Integrated Development Environment
XML	: Extensible Mark-up Langauge
AJAX	: Asynchronous JavaScript and XML
GB	: GigaByte
AMD	: Advanced Micro Devices
GUI	: Graphical User Interface
HTML	: Hypertext Mark-up Language
ROI	: Return on Investment
SAP	: System Applications and Products
JDBC	: Java Database Connectivity
SMTP	: Simple Mail Transfer Protocol
MAC	: Media Access Control
RAM	: Random Access Memory
IPv6	: Internet Protocol Version Six
XSS	: Cross-site Scripting
IAST	: Interactive Application Security Testing

ÖZET

YÜKSEK LİSANS TEZİ

WEB TABANLI OTOMATIK TEST ARAÇLARININ DEĞERLENDIRILMESI VE KARŞILAŞTIRILMASI

MHD KHALED AL SAWAF

İstanbul Üniversitesi-Cerrahpasa

Lisansüstü Eğitim Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman : Doç. Dr. Zeynep ORMAN

Yazılım ve uygulamaları test etmek yazılım ve uygulama geliştirme sürecinin önemli bir parçasıdır. Özellikle Web uygulamaları daha yüksek kullanıcı yüküne, kullanımına maruz kaldığı zaman bununla doğru orantılı biçimde performans gereksinimleri gibi karmaşıklık da artmaktadır.

Bu tez çalışmasında, Web uygulama testinde var olan eğilimler araştırılmaktadır. Bu çalışmadaki çıktılar; çeşitli otomatik test araçlarını inceleyerek, bunları karşılaştırarak, uygulama, kullanım, sağlanan teknik destek ve bakım kriterlerinden en iyisi seçilerek elde edilmiştir.

Ekim 2019, 69 sayfa.

Anahtar kelimeler: Yazılım test, test, web uygulaması, otomatik araçlar

SUMMARY

M.Sc. THESIS

Evaluation and Comparison of Web-Based Automated Testing Tools

MHD KHALED AL SAWAF

Istanbul University-Cerrahpasa

Institute of Graduate Studies

Department of Computer Engineering

Supervisor : Assoc. Prof. Dr. Zeynep ORMAN

Testing software and applications is an important part of the development process. Web applications in particular continue to grow in complexity, as do their performance requirements as they are exposed to higher user loads.

This thesis explores current trends in Web application testing. This is achieved by surveying various automated testing tools, comparing them, and selecting the best according to the criteria of implementation, usage, provided technical support, and maintainability.

October 2019, 69 pages.

Keywords: Software testing, testing, web application, automated tools

1. INTRODUCTION

Anything needs to be tested before it is used. Testing is everywhere, whether it is hardware or software everything needs to be tested before it is taken into use. English testing means everything which can be tried out. In computer science, testing is an act of searching faults in software. This is done by system analyst by executing parts of the software; it is the act of testing with some predefined data. The main motivation is to prove that the program or the software is working as required and producing the correct results. A simple testing definition is as shown by the diagram below. Here data is given as input which is then executed with inputted data. Testing is usually corrected done if the results are correct and when the inner state has been changed to another state.[1]

1.1 Definition of Testing

Web testing is a name given to software testing which focusses on web applications. It is the process of checking whether a web application has bugs before the web code is moved into a production environment or for any potential bugs. It is a web testing where web security is checked, the ability of the site to handle traffic, and the regular users of the site. [2]

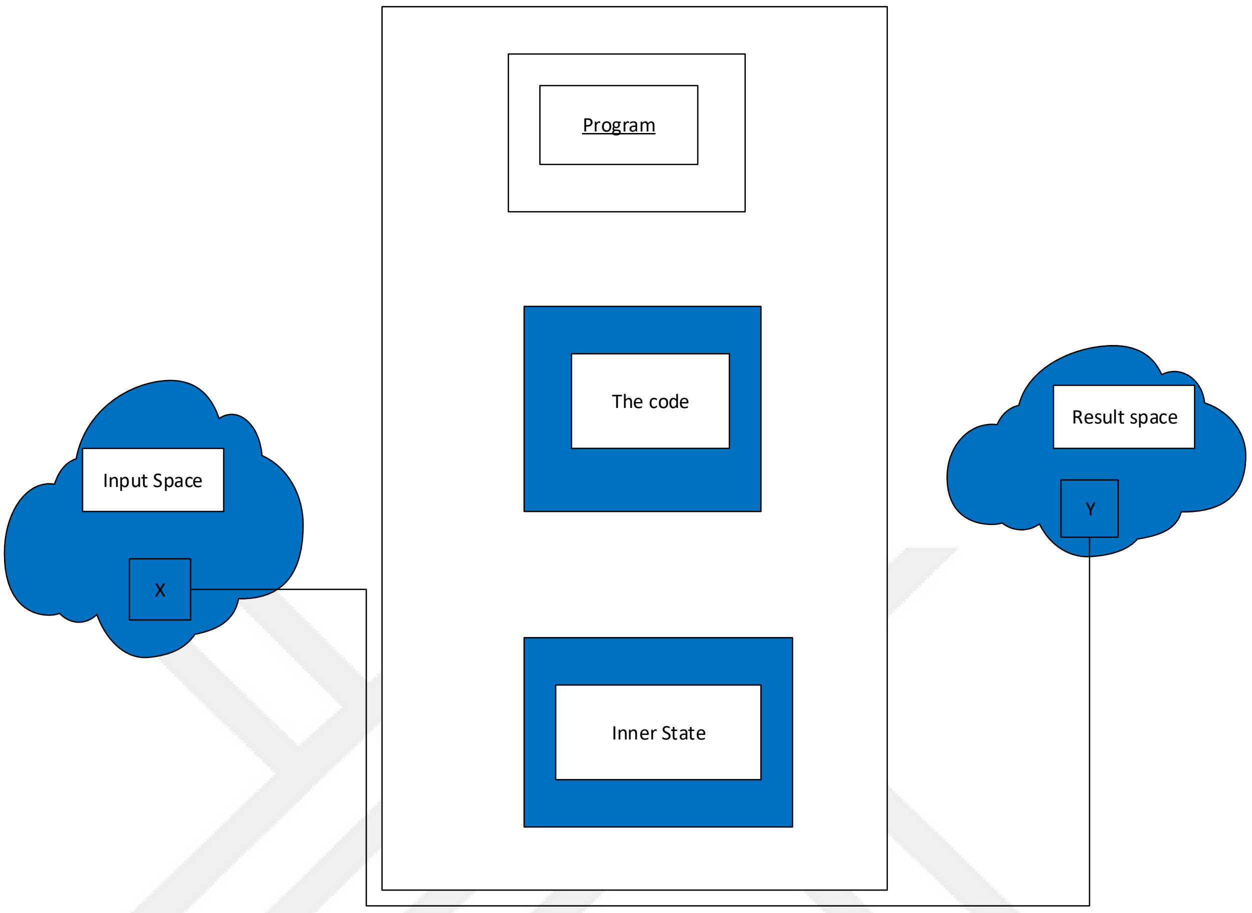


Figure 1.1: Illustration of testing

In software and applications, good testing usually includes test planning, creation of test environments, viewing the results, and test cases. These four phases usually take half of the resources reserved for application testing. When testing a new program such as a web application, it starts from the beginning. The planning phase usually specifies how testing is to be done. Testing results in mistake, bug, and language error. Error is defined as a deviation from specifications; the application is doing something that it is not supposed to do. Also, errors can result in a program or an application working very slowly; usually the user has trouble using it. Jukka defines an error as a human function that causes application defects. A fault is a reason for the failure of an application. [3]

There are usually four levels of testing during software or application development. This is as shown by figure below

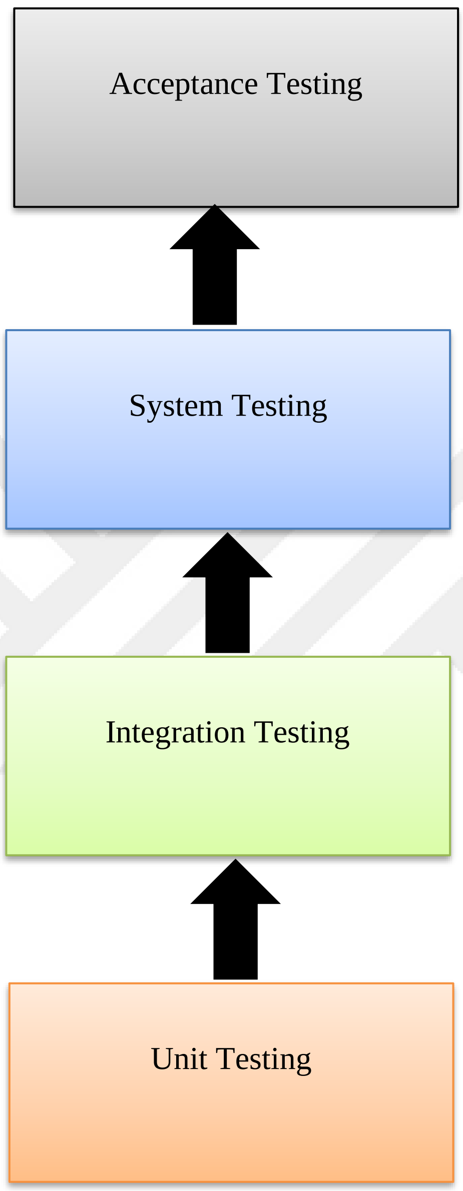


Figure 1.2: Methods of testing

As shown by figure, in-unit testing an individual component or a unit is tested. The major aim of this part is to make a single module in an application or a program to work correctly. In here unit testing is usually carried out by the unit implementer. Here testing can be done by some Personal Computer which resembles a real application or systems. A testbed is used to simulate the functionality of the module or the unit. [4], [5] When all the modules are combined together and tested it is what is referred to as integration testing. The major aim of integration testing is to find any errors or omissions between the integrated units. It is at this level that makes sure that the work is done correctly. This means that testing needs to start from a low level all the way to the upper level. [6]

System testing level usually tests the various parts of an application or the system; this usually consists of units or modules which have been integrated into a complete system. The idea behind this level is to make sure that the whole system complies with the predefined requirements. In addition, this level tests both non-functional and functional requirements. Some of the non-functional requirements tested are usability, reliability, and performance. The results of non-functional requirements are compared with functional requirements. The person who performs non-functional testing ought to be independent of the development of the system people. [7]

As shown by figure system testing is then followed by acceptance testing. This level evaluates the application against the application requirements and it assists in deciding whether the application is ready for delivery. [8]

1.2 Division of testing

According to the way, testing is performed it can be executed by humans. The first division of testing is manual testing which in this case involves manual tasks like setting up tests environment and executing tested functionality, reviewing and collecting the results and recording the found issues. The second division which has been highly exploited in this thesis is automated testing; which is the execution of tests without human intervention. Usually, automated testing includes the ability to run a subset of all tests, capturing the results, running the test cases, automatic set-up and recording of environmental variables, and analysis the processing and results in a comprehensive and clear way.

One of the benefits of manual testing is that the set-up time is shorter as compared to automatic testing. One of the major drawbacks of manual testing is that it leads to incorrect or inaccurate results. Also, the execution of test cases is slower than automated testing. One of the benefits of automated testing is that it assists in the elimination of human errors. Second automated tests are faster as compared to manual testing. Third automated testing can lead to cost reductions. [9]

A basic testing method is a box-based approach. Box approach is divided into two which are black-box testing and white box testing. These two approaches are used to describe a point of view of the test engineer. Black box testing is where a tester does not have any information about the internal procedure and working of a web application testing. Black box testing which is also known as the closed box is data-driven testing that tests the functional part of a web application. On the other hand, white box testing is structural testing or a code based type of testing. In the white-box type of testing, one has full knowledge of the internal working of the web application. It is also important to note that the black box type of testing is based on external expectation as the internal behavior of a web application is unknown. On the hand in white box testing, the internal working of a web application is fully known by the tester. Lastly, white box testing is exhaustive and very time consuming. [10]

1.3 Web application testing

The previous section highlighted extensively what is testing, the elements of testing. This part will focus on web testing. In addition, as highlighted there are several web applications that are being developed on a daily basis and with each line of code being written for web applications, potential bugs arise on a daily basis. This raises the need for web testing. Web testing is an important part of any application development yet some developers underestimate this process. The chance of bug appearing increased with every life of code. There are six basics of web application testing; the first one is functionality testing. [11] This step is specifically used to make sure that the web application is working correctly. Specifically, this faces checks if a database connection is working. This is done in the early stages of development so as to build up the whole process of app building. It usually reduces the risks towards the end of the cycle. A typical functionality testing in web application usually includes identification of functions that the web application is supposed to do, the analysis of the actual results, the execution of the test cases, and data input of the test case. Usually, the tester here is supposed to simulate the actual application and creates test conditions that are related to the user requirements. Usability testing is the second phase which goes beyond the first test. This step combines both the user experience and functionality testing. This step usually involves developing the testing strategy which ensures that all functions of the application or the program are examined and included in the content and navigation. Second, the recruits test participants either externally or internally. Third, tests are done under the experts' observation and lastly is analyzing the results so as to improve the web application accordingly [12]. The process of usability testing is done as shown by the diagram below in Figure 1.3 [13]

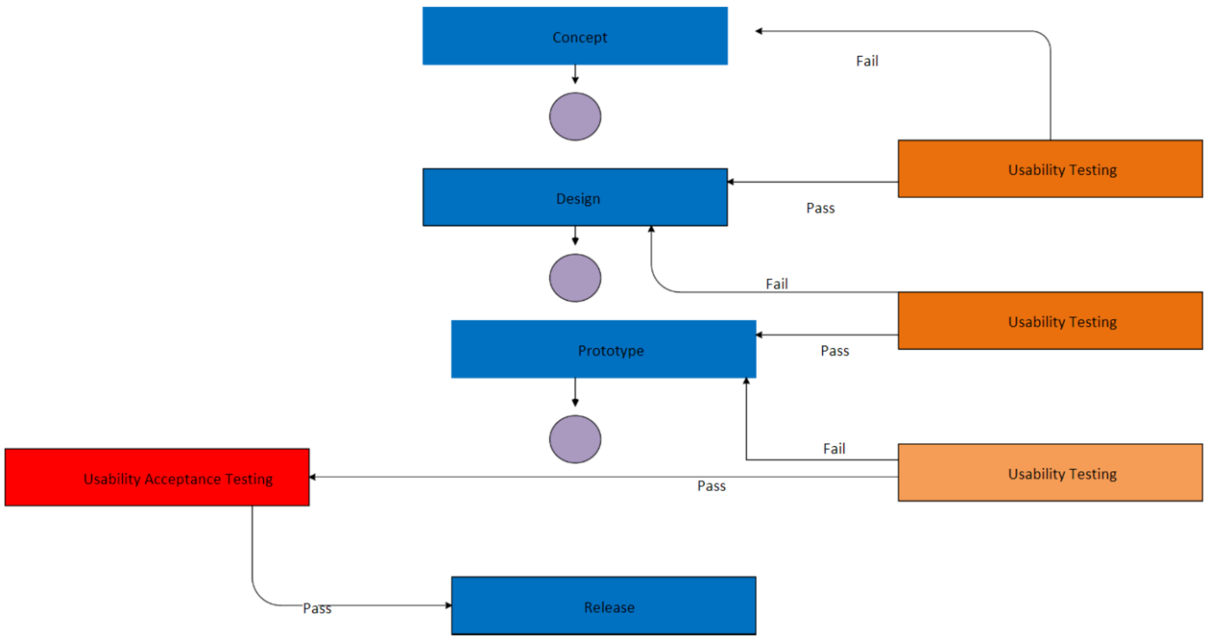


Figure 1.3: Usability testing of a web application

The next phase is interface testing; in here the web application tester checks whether or not there an interaction between the web server and the app server. In here not all the communication is tested but also the displaying of error messages. It is also by this phase that the interruptions by the user and the server are handled. The next phase is compatibility testing, which ensures that the web application is compatible with all the devices and various browsers. There are various elements of compatibility testing; the first element is browser compatibility; here the tester checks whether the application is compatible across various browsers. Here the tester checks whether Ajax, authentication requests, browser notifications, and JavaScript are working correctly. Operating system compatibility is also checked at this phase, where the tester checks of the application run smoothly on Unixes, Linux, Windows, and macOS. Mobile compatibility is also done at this phase; here the tester checks whether the application is running smoothly on various mobile devices [14].

The fifth phase is performance testing; in here the tester checks whether the application is responsive to all applications and it performs under very heavy load. This usually includes testing under a very heavy load under different internet speeds and how the application behaves under peak and normal loads. In addition, this phase tests application resiliency; how the web application performs under stress and various hardware configurations. The final testing in the web application is security testing; this makes sure that the application is protected against unauthorized access or any form of harmful actions via malicious software and other viruses. Usually, security testing involves four activities that are testing whether all the secured pages can be accessed, verifying the application SSL, makings sure that the restricted files cannot be accessed or downloaded without adequate authorization, and checking that all the open sessions are closed after an ongoing user activity. Security checklist comes hand in hand on this stage and ought to include tasks such as secure transmission, error handling, session management, an transmission, and authorization, denial of service, specific functionality tests, and cryptography. This means that security testing has five major goals which are identifying the web application security needs, preparing a test plan, automated testing on top of manual testing, carrying out the best security test cases, retesting, and fixing identifying defects [15].

From the above web testing steps, it means that web application testing is the practice of testing web applications or websites for potential bugs. It is a complete testing of designed web applications. The web applications need to be tested from end to end before the application goes live for the end-users. The web application testing checklist ought to have six components which functionality testing, usability testing, interface testing, compatibility testing, performance testing, and security testing [16].

There are four major types of web testing which are simple static testing, dynamic testing, e-commerce website testing, and mobile testing. In static testing few points are considered which are testing the GUI design, checking the page web-application links, and checking the scroll bar carefully. Dynamic web testing is where the designer checks if the user can change or update the web application regularly [17].

1.4 Benefits of web application testing

Testing is any application development is the first step to quality assurance. The major aim of testing is to not necessarily to verify the finished work with the initial specifications of the contributor but to ensure that the application is user-friendly. In addition, testing does not only ensure the finding of bugs in an application but also to control the quality of the application. Meaning the first customer of testing is the quality assurance team [18].

There are various advantages of web testing. First is that it ensures complete correctness of the web application. As highlighted previously, web application testing is carried out in layers or levels. This means that web testing ensures completeness and correctness of the web application. Second web testing increases confidence in the performance and functioning of the web application before the web application is released. Third, testing reduces future risks; the web application is tested rigorously after every sprint and iteration. Meaning that there very few chances of risks and failure in the future. Forth, testing decreases repetitive efforts. As a web application is tested thoroughly then it means that they are no looking back; it reduces the chances of a breakdown. Fifth web-testing reduces costs and time via what is known as an automation testing tool. Sixth, web application testing boosts on customer satisfaction; it is through testing that web application quality is reached; testing here ensures that customers' patience is not taxed with a defective web application. Seventh, web testing comes together with profit; testing is part of profit-making. Offering a rigorously tested and quality checked a web application [19], [20].

Other advantages of web testing are;

- ❖ Overcoming the blindness to issues when a designer or a project manager has been looking at the same time
- ❖ Ensures high-quality web application that generates better results
- ❖ Alleviates the pressure of the designing team
- ❖ By carrying out all the steps of web application testing then comprehensive testing is completed no matter how the designing team are busy
- ❖ Web application testing provides what designers refer to as the additional level of testing by allowing testing to be carried as per the web application steps [21].

1.5 Problem statement

Web application testing or software testing is a very difficult task. This is because of the peculiarities of some programs and applications. According to Giuseppe, in the last past years there have been several issues specifically in the field of web application testing which has resulted in many research work done; in web application testing techniques and methods of web-application testing have been looked at widely but this is not adequate. This thesis will look at web application testing tools. Focusing on this will assist in finding out the best web-application tools thus making web-based testing easily.

1.6 Objective of this thesis

The major aim of this thesis is to conduct an evaluation of the best tool to be used for testing web applications based on several criteria. To refer to the very best tool there is a lot of factors to be considered such as browser compatibility, technical support, and integration. The important part of this work is to compare several web application tools and suggest the best tool.

1.7 Research question

This thesis has systematically reviewed various web application tools. The leading question can be put like: Which is the best web application testing tool?

1.8 Principles of web application testing

There are seven principles of web application testing. First is exhaustive testing is not possible. Instead, designers need the optimal amount to test which has to be based on risk assessment of the web application. Second is the defect clustering principle; which states that a small number of modules can contain most defects that are detected. This principle applies the Pareto principle to web application testing. Third, is the principle of pesticide paradox; which states that repetitive test cases can be conducted to a web application to discover any new trends; this is done to improve the existing testing methods continually. Forth, is testing shows a presence of defects; hence, any time designers talk of testing the never mean absence of defects. The firth principle is the absence of error; fallacy. When testing a web application, the absence of an error is a fallacy. Early testing is the sixth principle; this states that testing ought to start as early as possible during web application development; so that defects can be discovered early during web application design. Seventh is testing is context-dependent; meaning that the way you test a web application is not the same way one tests an e-commerce site.

1.9 Literature Review

Samad Paydar (2010), have focused on the framework of web-based systems. The author has presented an agent-based framework for testing web-based applications. According to the author, the major design goal is to develop a flexible and effective system. The framework has been developed in such a way that it utilizes various sources of information about what the author referees to as automate the test processes. The resulting frameworks is a system that consists of a set of agents. Samer (2013), has focused on the comparison of the GUI automation tools for dynamic web applications. According to the author, automating the process of software testing assists the designing team in releasing a quality software or an application. It also shortens the period of application development. The author has summarized the best guidelines and practices for GUI functional tests against web applications. To find out the best GUI practices, the author has given an overview of HTML as it is what all the automation tools try to access [22].

Samer (2013) has compared how several web application tools function in some browsers. His comparison was based on Firefox, Internet Explorer, and Google Chrome. According to Samer, TestComplete seems to be less compatible with web browsers especially Internet Explorer as compared to Sahi and selenium. This is because the TestComplete tool seems to be using an engine to access the web application or the web pages that are based on accessing Internet Explorer [23].

Suguna and Rajya (2015) and others have given a review of automated testing tools of testing supported by the tools led by a survey of some interesting facts by the two authors available. [24] Sahi and selenium tools appears to be more reliable than other tools. Besides, they appear to be more time saving and effective as compared to other tools. Running tests and recording by using TestComplete appears to be weaker when dealing with dynamic web applications [25], [26].

A case study has been carried out by PhotoSnack on the best web application tool, and it is evident that Selenium is the best web application tool tester. The tool supports all the testing activities of a web application tool listed by Table 1.1 below.

Table 1.1: Web application testing activities

Testing activity	**Description**
Security testing	This an activity specifically planned to uncover imperfections in the security components for data framework.
Usability testing	This is a type of testing with a view of viewing clients' perception of the web application.
Compatibility testing	This type of testing activity where the test engineer checks for the compatibility of the web application

Performance testing	Checks the viability of the web application
Stress stressing	This type of testing activity assess the conduct of the web application

It is evident that PhotoSnack performed several steps in turn with three client levels i.e, one user, fifty clients, and one hundred clients. PhotoSnack went ahead to detail the performance testing of selenium web application software. As one can view from the testing, the web application was very efficient when only one user was accessing it and when performing several tasks on it. The average time is a minimum case of one. When the number of clients is increased, the average mean time is increased as well. It is moderate in the case of fifty users and doubled in the case of a hundred clients. Therefore one can conclude that as the number of clients is increased, then the performance of the application is reduced. But as far as the testing mechanism is concerned then selenium tool proofs to be the best tool. It is best suited for testing web applications under peal conditions. The tool can also be used to check errors in a web-application and the performance of the web application [27], [28], [29].

It is evident that there not so many tools testing non-functional attributes of web applications such as trustworthiness, fault tolerance, and reliability which are not readily available. Also, there is the death of all open source tools which are using mutation techniques that can perform automated test case execution which is based on mutation analysis while at the same time optimizing the test suite [30].

2. MATERIALS AND METHODS

2.1 MATERIALS

Web testing tools usually improve reliability, increases ROI (Return on Investment), and turnaround time. Various web testing tools assist in diverse web application testing. With web testing, issues like web functionality, usability, accessibility, performance, and compatibility are released in public when web testing tools are used. This chapter will critically analyze the various web application testing tools, list their features, pro, and cons.

To start with web application testing tools are categorized into seven categories which are load, stress, and performance testing tools, W3C Link checker, cross-browser testing tools, web functionality and regression testing tools, link manager testing tools, and web site security testing tools [31].

2.1.1 Webload

Webload is one of the load testing tools or what is known as a performance testing tool. This tool combines performance, integrity, and scalability. This tool usually simulates lots of users which makes it possible to test large loads and report any form of a bottleneck. This tool was first launched in August 1997. Since this time there have been about 20 versions of the Webload. After 1997, the first version of Web load was in June 2010, and the version was 8.5 which was side-by-side views and JAX based. The second version was in the same year version 8.6 which has a statistical correlation. The third version in 2012; version 9.0 which was a load testing tool from the cloud. The tool can probe a statistical client. The other version was the 10.0 version which could support IPv6 and had a new interface. Later version 10.1, and version 10.2 were released on May 2013 and December 2014 consecutively. The tow version has the Jenkins plug-in and web dashboard. Later version 10.3 and version the latest version 10.3.1

were released on October 2015 and February 2016 consecutively. The latest version could unfreeze and freeze a test during the integration or execution of a new New Relic [32].

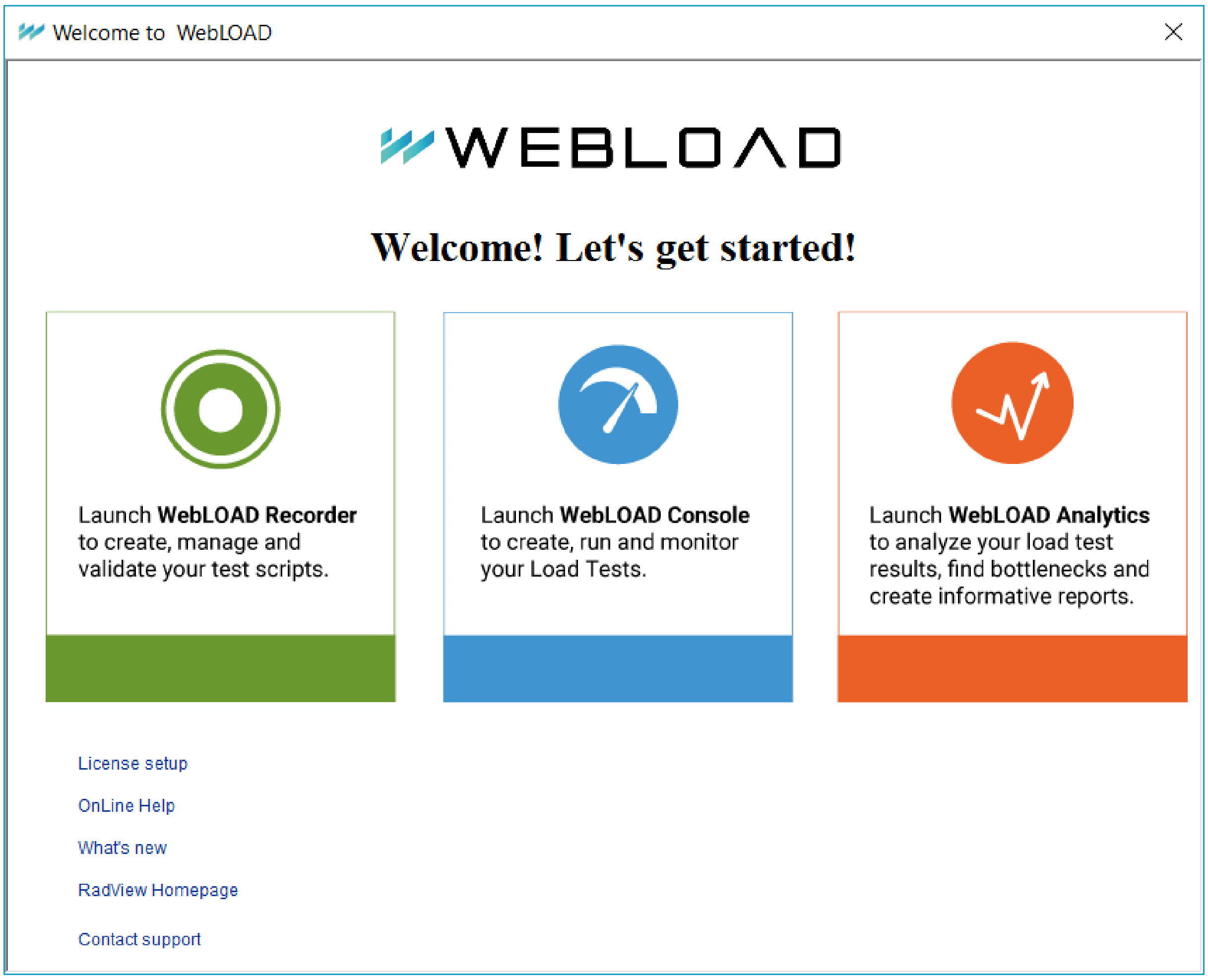
The current version, 10.3.1 has six major features which are IDE, Correlation, Load generation, analytics, PMM, and web dashboard. With a web dashboard feature, the tool can analyze performance tests results from any mobile device or browser. Second, with PMM feature, one can collect the server-side statistics during tests runs thus able to provide the users with any additional data for root-cause analysis. With the Load generation feature, the tool can generate from the cloud. IDE feature one is able to record and edit load tests scripts visually. With the analytics feature, one can set a predefined analysis report which provides performance data and assists users with identifying bottlenecks.

Other features of webload include:

- Mobile load testing
- Technologies supported
- Test creation
- JavaScript
- Test execution

One of the advantages of Webload tool is that the tester can provide a clear analysis of the web application. Second, one can pinpoint issues and the bottlenecks which may stand on the way of achieving one load response requirements. Also with Webload, a tester can work with cloud providers such as Amazon to create a dry run of massive virtual user load with load generation console on windows and Linux.

Another benefit of Webload is that it offers a robust testing platform and flexibility. Also, one can create efficient load testing. Lastly, the tool has offered integration with Jenkins, app dynamics, New Relic, Amazon Web Dynamics, Selenium, and Dynatrace. Some of the cons of the tool are; full functionalities for Webload is not offered for free; a trial version is provided but later one needs to purchase the tool to one can realize full functionality of the tool


Welcome to WebLOAD

WEBLOAD

Welcome! Let's get started!

Launch **WebLOAD Recorder** to create, manage and validate your test scripts.

Launch **WebLOAD Console** to create, run and monitor your Load Tests.

Launch **WebLOAD Analytics** to analyze your load test results, find bottlenecks and create informative reports.

License setup

OnLine Help

What's new

RadView Homepage

Contact support

Figure 2.1: WEBLOAD web application testing tool

2.1.2 TestIO

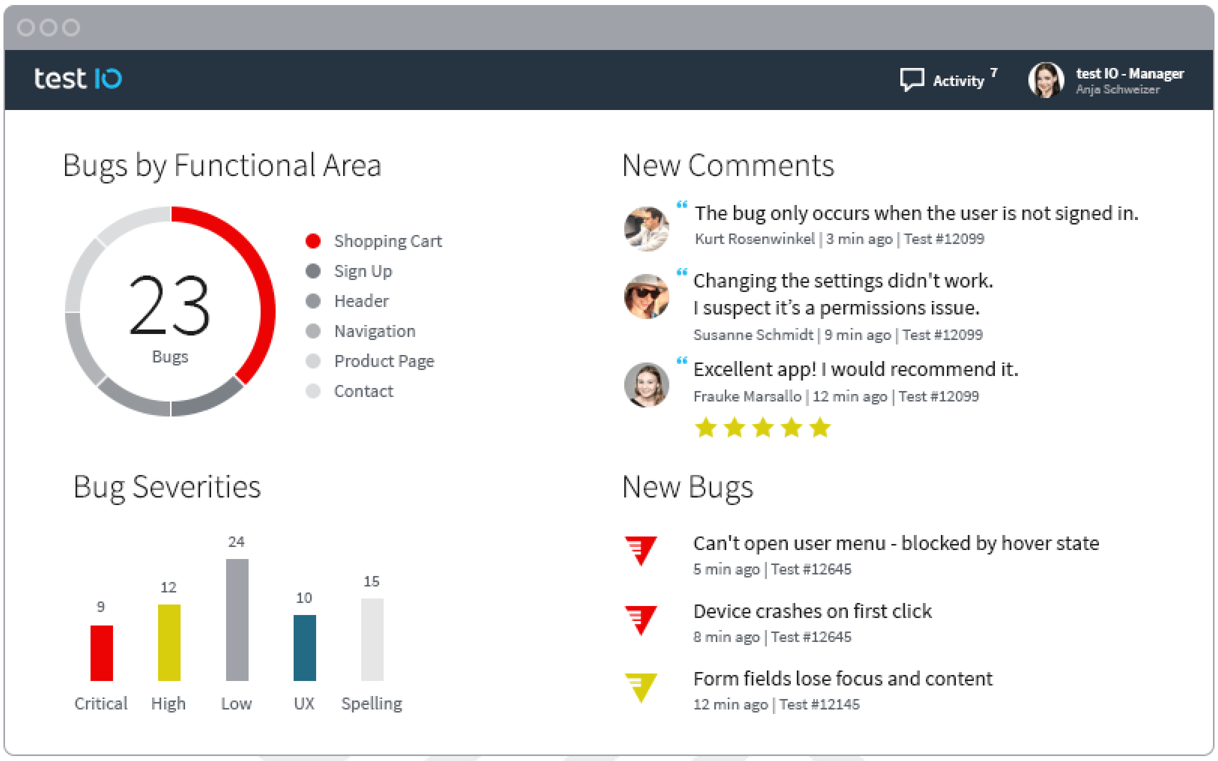
TestIO is another web testing tool, this tool makes sure that one web application works everywhere by crowd testing. With this tool, one can deliver the test. Some of the features of this tool are; able to remove QA bottlenecks with flexible testing which scales up to one needs. Test on rea; devices; this feature enables one to expand on one coverage to the hundreds of platforms and devices. The device also ensures one of the professional testers have unbiased eyes on the product.

Other features are:

- ❖ Cross-device compatibility
- ❖ Cross-browser compatibility
- ❖ Parallel execution of various tasks
- ❖ Offers cloud-based solution
- ❖ Hierarchical presentation
- ❖ QA script reviews

Some of the advantages of the tool can integrate project management with bug tracking tools. The tool also can offer a crowd-powered quality assurance testing platform thus achieving enhanced efficiency and a more extensive reach among the actual web application users. Second, with test IO tool, one can uncover any programming errors and at the same time, provide contextual data to assist the designer in creating a more efficient web application. Third, the tool can provide support to multiple browsers and devices. Besides, the tool is compatible with most operating systems and is readily available. Lastly, the tool provides a holistic platform for all various types of software, be it a web application, web sites, iOS, mobile-optimized web software, and mobile apps.

Some of the cons of the tool are; TestIO is not provided even on trial version. The starter version with limited capabilities is provided with 3000 dollars per month, and the professional version is provided at the cost of 4500 dollars per month. Also, as compared with Webload testing tool, the response time of this tool can go up to 24 hours.



"The bug only occurs when the user is not signed in.
Kurt Rosenwinkel | 3 min ago | Test #12099

"Changing the settings didn't work.
I suspect it's a permissions issue.
Susanne Schmidt | 9 min ago | Test #12099

"Excellent app! I would recommend it.
Frauke Marsallo | 12 min ago | Test #12099

Can't open user menu - blocked by hover state
5 min ago | Test #12645

Device crashes on first click
8 min ago | Test #12645

Form fields lose focus and content
12 min ago | Test #12145

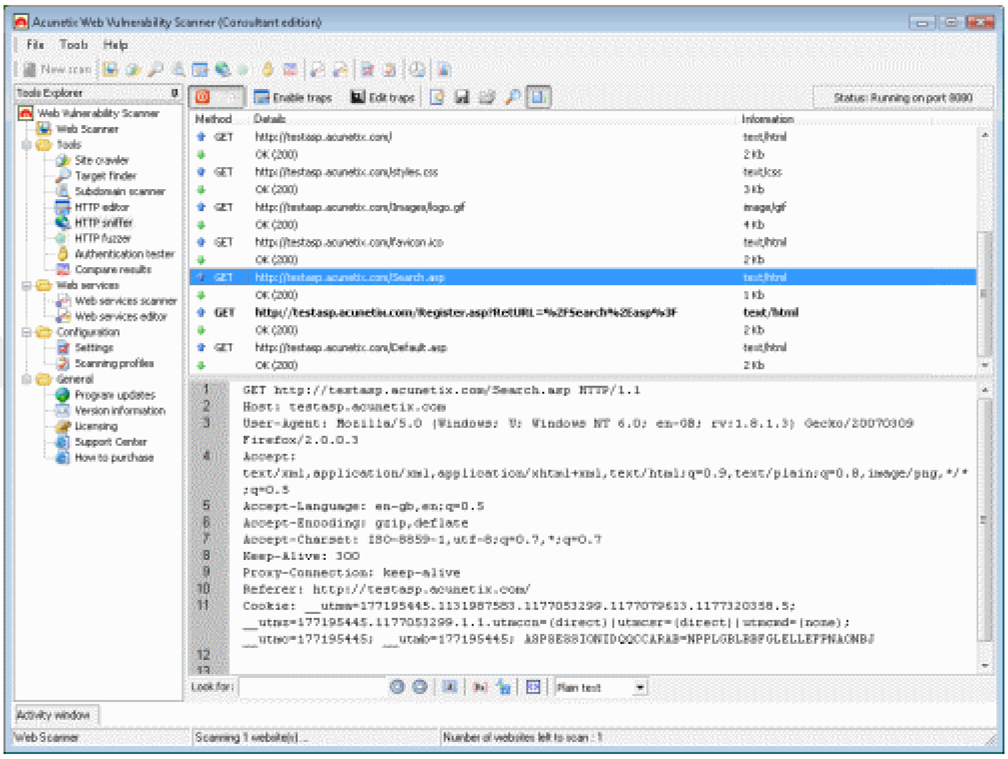
Figure 2.2: TestIO web application tool

2.1.3 Acunetix

Acunetix is an automated fully web application testing tool that can detect and report to over 4500 web application vulnerabilities which include all the variants of XSS and SQL injection. The tool fully supports JavaScript, authenticated applications, allowing of complex, and supports HTML 5 [33].

There are many features with this tool, such as deep-scan technology: This feature is an automated crawler that can crawl on complex custom HTML5 websites and web applications. This feature also includes even client-side single-page applications. Second with this tool, one has the ability of scanning websites with modern web technologies; this includes Javascript frameworks such as Vue, ember, react and angular. Back-end technologies such as Asp.net, PHP, Ruby on rails and java. Third, the tool has other features such as custom authentication schemes, multi-factor authentication, and single sign-on authentication.

One advantage of the tool is that it provides interactive application testing commonly known as IAST or what is known as gray box testing for java powered web applications. Second, it enhances regular dynamic scan via deployment



Method	Details	Information
GET	http://testasp.acunetix.com/	text/html
	OK (200)	2 Kb
GET	http://testasp.acunetix.com/styles.css	text/css
	OK (200)	3 Kb
GET	http://testasp.acunetix.com/Images/logo.gif	image/gif
	OK (200)	4 Kb
GET	http://testasp.acunetix.com/favicon.ico	text/html
	OK (200)	2 Kb
GET	http://testasp.acunetix.com/Search.asp	text/html
	OK (200)	1 Kb
GET	http://testasp.acunetix.com/Register.asp?RetURL=%2FSearch%2Easp%3F	text/html
	OK (200)	2 Kb
GET	http://testasp.acunetix.com/Default.asp	text/html
	OK (200)	2 Kb

```
GET http://testasp.acunetix.com/Search.asp HTTP/1.1
Host: testasp.acunetix.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-GB; rv:1.8.1.3) Gecko/20070309
Firefox/2.0.0.3
Accept:
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*
;q=0.5
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Referer: http://testasp.acunetix.com/
Cookie: __utma=177195445.1131987583.1177053299.1177079613.1177320358.5;
__utmz=177195445.1177053299.1.1.utmccn=(direct)|utmcsr=(direct)|utmcmd=(none);
__utmc=177195445; __utmb=177195445; ASPSESSIONIDQQCCARAB=NPPLGBLBBFGLELLEFPNAONBJ
```

Figure 2.3: Acunetix web application testing tool

2.1.4 TestingWhiz

This is one of web application testing tool developed by cygnet Infotech. The tool offers automated solutions such as web testing, database testing, automation, optimization, and API testing. Some of the features of this tool includes are; [34]

- ❖ Delivery in agile cycles
- ❖ Risk-based testing
- ❖ 300 testing commands which also includes an in-built JavaScript
- ❖ Playback test automation framework

- Object eye internal recorder: Allows the tester to archive and record web application controls. It permits one to edit the stored objects through smart editing features
- An integration with test management tools such as HP quality center
- Reusable methods
- Image comparison
- Dynamic test data support: This feature is used to reduce the maintenance of the automation scripts and the test coverage
- Roust logs and reports
- Visual recorder
- Captcha automation

TestingWhiz tool is very easy to use for both large and small automation. The tool comes with what software test engineers refer to as FAST engine which utilizes intelligent and reusable recording techniques; this includes data-driven, keyboard driven and Ms –excel programming. In addition, the tool allows one to create powerful and modular automation scripts with ease. Also the test commands applied by the tool are usable even to those users who have no coding skills to optimize testing workloads and to boost the efficiency on the automation projects. Also, the web application testing tool supports various browsers which are firefox, safari, android mobile browsers, and IE, Safari, IE and Chrome. In addition, the engine executes automation projects with inimitable flexibility, and speed since the tool has the ability to implement various file formats which includes DOS, .exe, and .bat file.

The tool is not provided for free; it is subscription based which is readily available on request. Also the tool does not have the ability to fully scan a web application like Acunetix

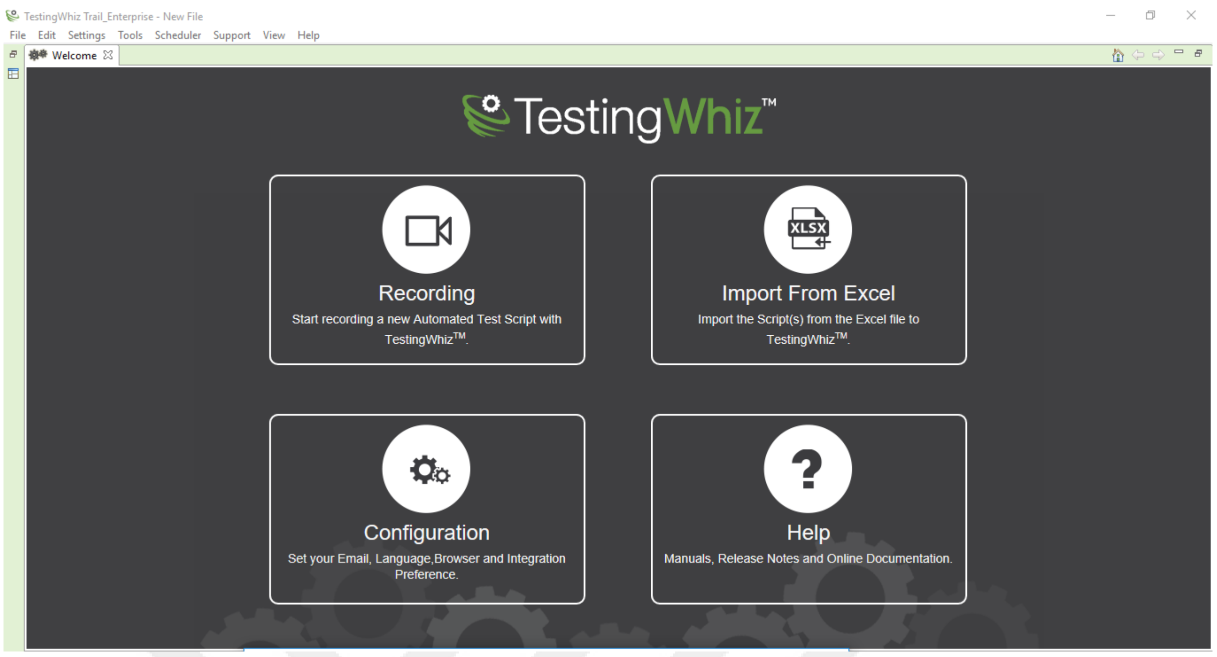

TestingWhiz Trail_Enterprise - New File
File Edit Settings Tools Scheduler Support View Help
Welcome
TestingWhiz™
Recording
Start recording a new Automated Test Script with TestingWhiz™.
Import From Excel
Import the Script(s) from the Excel file to TestingWhiz™.
Configuration
Set your Email, Language,Browser and Integration Preference.
Help
Manuals, Release Notes and Online Documentation.

Figure 2.4: TestingWhiz web application testing tool

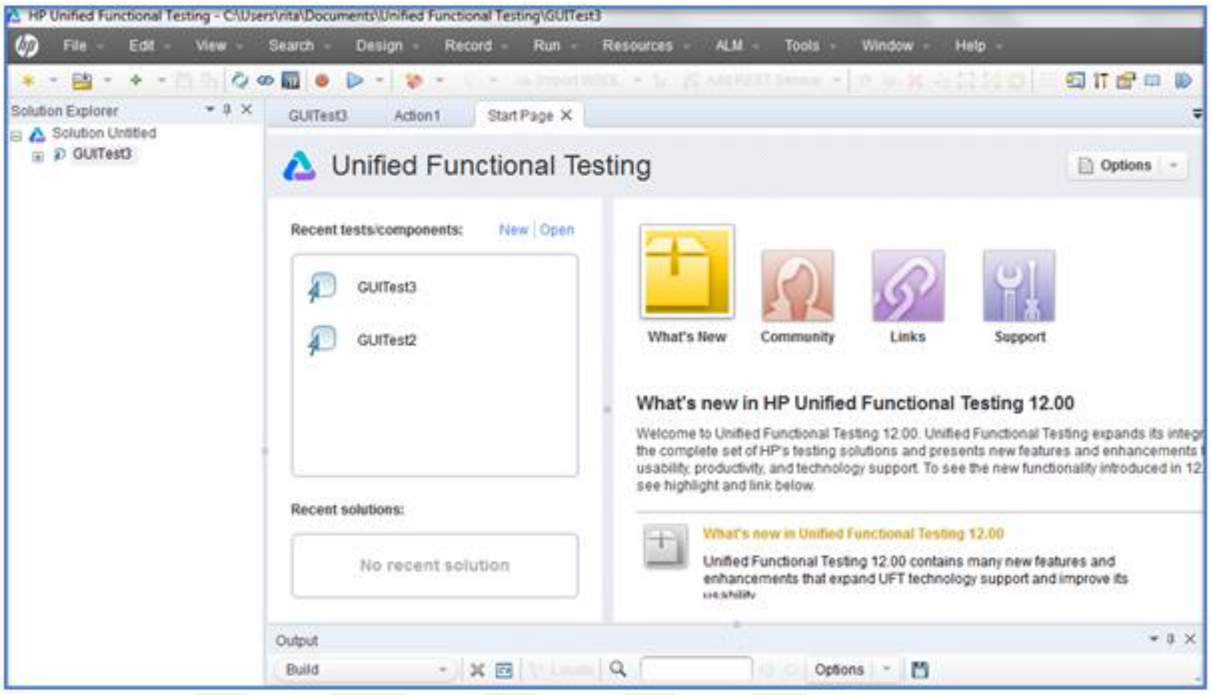
2.1.5 HPE UFT (QTP)

This is a web application tool which provides users with interactive tools which are used for executing and creating automated apps on web, mobile, and desktop platform. The tools are used by software testers to allow users to execute and create automated functionality and performance tests. Besides, the tool is specifically used to perform both regression and functional testing via a user interface like a web interface or GUI.

Some of the features are; Manages exception handling, Supports data-driven testing, extensibility, Complex UI objects, Extensibility, Automated documentation, Error handling mechanism, Unique handling mechanism, and Integration with mercury quality center and mercury business process testing. Some of the advantages of the tool support several technologies but it depends on versions which are java, .net, SAP, Oracle, Siebel, Delphi, Power Builder, and windows mobile. The tool has the ability to let software testers to edit and display test codes that use VBScript. Lastly, the tool is designed for more advanced users where they can edit all the test functions for the Global root actions.

Some of the advantages of the tool support several technologies but it depends on versions which are java, .net, SAP, Oracle, Siebel, Delphi, Power Builder, and windows mobile. The tool has the ability to let software testers to edit and display test codes that use VBScript. Lastly, the tool is designed for more advanced users where they can edit all the test functions for the Global root actions.

Some of the disadvantages are; available as on-premise software for both windows but it is not readily available for free; all the interested parties are required to contact HPE for licensing and pricing options; the free platform which available do not have all the required features. Also, the tool runs only on Windows environments and cannot test with all browsers versions and types; specifically, the tool supports only Opera. Third, there is no way one can run tests independently even though remote execution is possible. The high licensing cost of the tool indicates that it can only be used on the windows environment, but it is limited to the smaller testing team. Lastly, the supports VB Script meaning that the tool cannot use some Visual Basic keywords. Also, the VB script does not support inheritance and polymorphism.



What's new in HP Unified Functional Testing 12.00

Welcome to Unified Functional Testing 12.00. Unified Functional Testing expands its integ
the complete set of HP's testing solutions and presents new features and enhancements
usability, productivity, and technology support. To see the new functionality introduced in 12.
see highlight and link below.

What's new in Unified Functional Testing 12.00

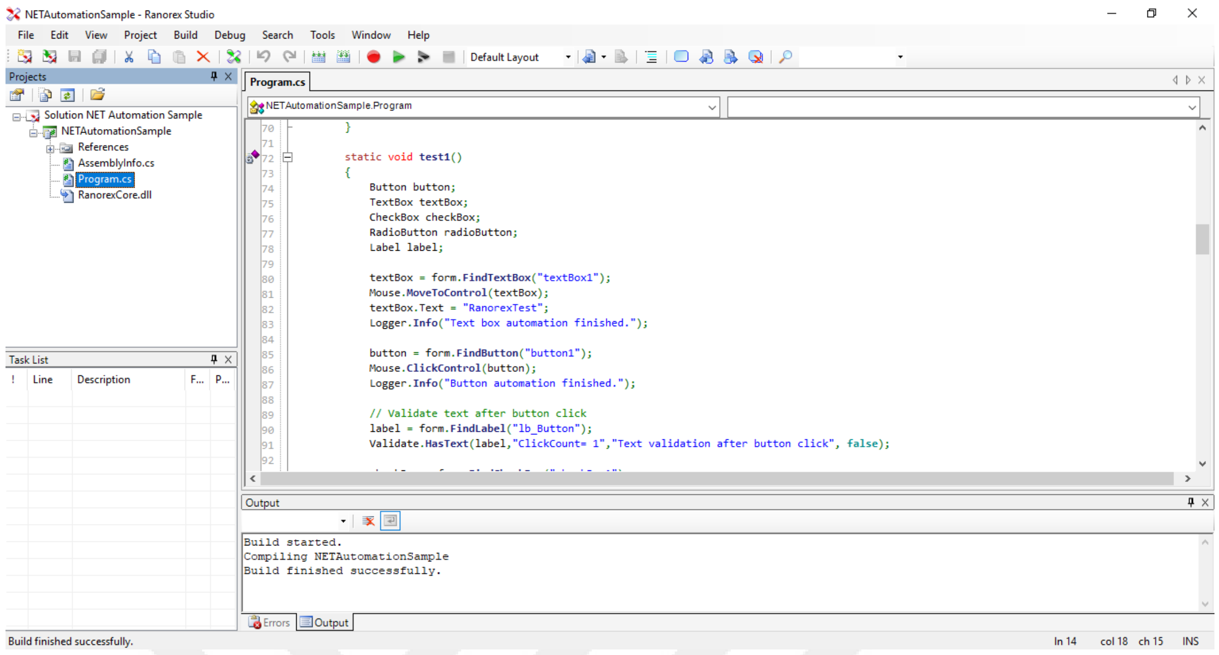
Unified Functional Testing 12.00 contains many new features and
enhancements that expand UFT technology support and improve its
usability

Figure 2.5: HPE unified web application functional testing tool

2.1.6 Ranorex

Ranorex studio is one of the web application testing automation tool which covers all the mobile, desktop and web applications. Some of the features are record and playback, GUI recognition, reusable test code, and integration with the various tools.

The tool delivers robust object recognition for any of the web technology thus making web application testing reliable and resilient. The tool also can identify UI elements with flexible and powerful RanoreXPath syntax that is capable of handling dynamic elements. Third, the tool can support web frameworks and web technologies. The test hybrid is based on open source chromium, embedded framework, and testing of JavaScript and Test Java. Forth, the tool can perform what testers refer to as web element identification, thus making it possible to analyze web application and to apply predefined rules for stable identification of elements. Besides, the tool integrates with the current solutions [35].


NETAutomationSample - Ranorex Studio
File Edit View Project Build Debug Search Tools Window Help
Default Layout
Projects
Solution NET Automation Sample
NETAutomationSample
References
AssemblyInfo.cs
Program.cs
RanorexCore.dll
Task List
! Line Description F... P...
Program.cs
NETAutomationSample.Program
Output
Errors Output
Build finished successfully.
ln 14 col 18 ch 15 INS

```
        }

        static void test1()
        {
            Button button;
            TextBox textBox;
            CheckBox checkBox;
            RadioButton radioButton;
            Label label;

            textBox = form.FindTextBox("textBox1");
            Mouse.MoveToControl(textBox);
            textBox.Text = "RanorexTest";
            Logger.Info("Text box automation finished.");

            button = form.FindButton("button1");
            Mouse.ClickControl(button);
            Logger.Info("Button automation finished.");

            // Validate text after button click
            label = form.FindLabel("lb_Button");
            Validate.HasText(label,"ClickCount= 1","Text validation after button click", false);
```

```
Build started.
Compiling NETAutomationSample
Build finished successfully.
```

Figure 2.6: Ranorex web application tool

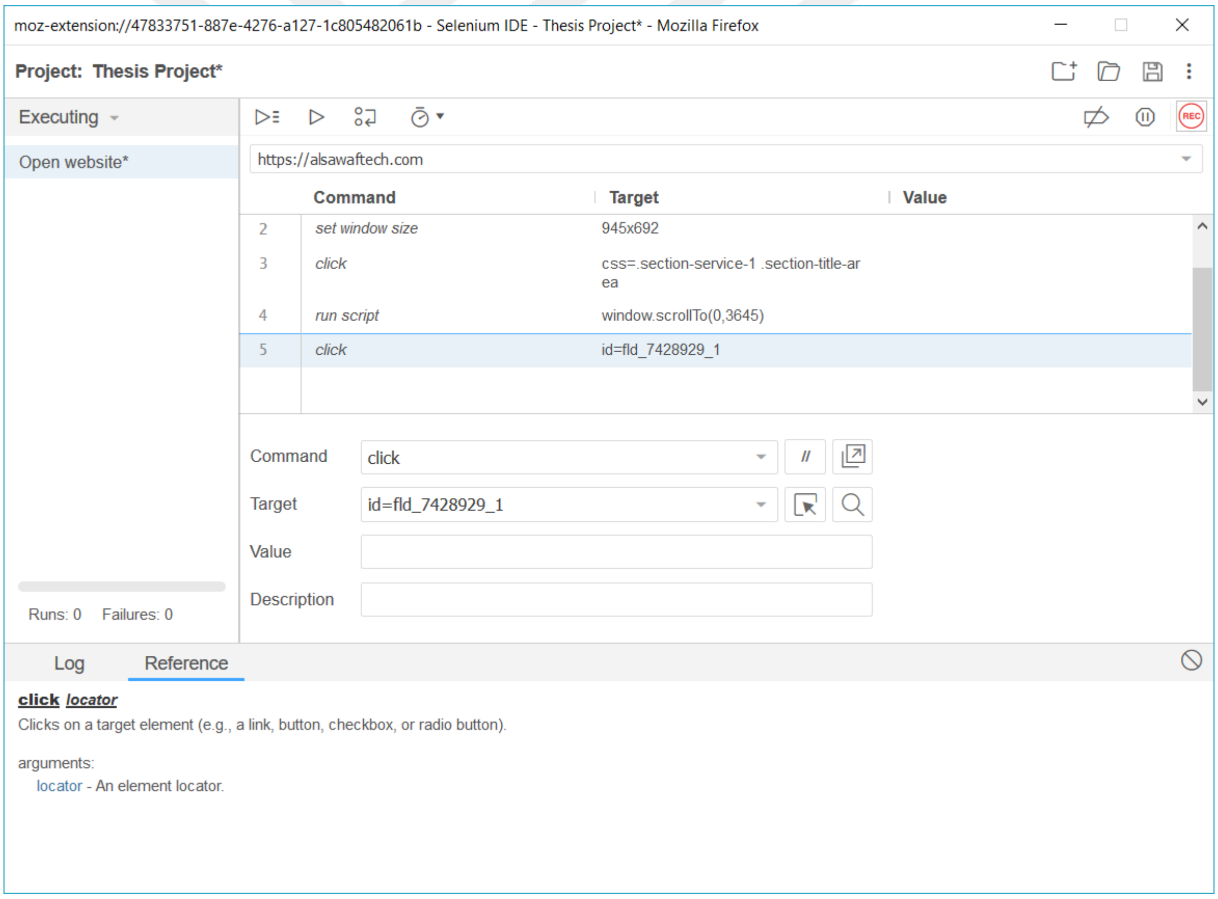
2.1.7 Selenium

Selenium is a web application tool that supports test automation. Selenium was first discovered in 2004 by Huggins as an internal application at ThoughtWorks. The tool is composed of several components which where each of the components has a specific role in helping in the development of web applications. One of the important components is the selenium IDE which is a complete IDE for selenium tests. Most of the Selenium Quality Assurance engineers focus on two or one tools that the needs of their project. Some of the features are; Supports android testing and iPhone, runs a little faster and even server is not required, it is very easy for a Web Driver to build a keyword, and Selenium server initializing is not required [36].

Advantages of selenium, the test scripts can be written in any of the programming languages like Python, c£, Perl, Java, and .net. Second selenium tests can be carried out in any of the operating system, Linux, Mac, Windows. The tests can be carried out in Opera, Safari, Mozilla

Firefox, and google chrome. Forth, the tests can be carried out using JUnit and TestNG for generating reports and managing test cases [37].

Some of the disadvantages are; Test engineers can only use selenium to test web applications only; the test engineers can’t test desktop-based applications or any other software. Second, when using Selenium, there is no support available for selenium. Tests engineers are required to leverage the available customer communities. Third, test engineers cannot test images; they need to integrate selenium with what is known as Sikuli for image-based testing. Lastly, the tool does not offer a native reporting facility [38].



moz-extension://47833751-887e-4276-a127-1c805482061b - Selenium IDE - Thesis Project* - Mozilla Firefox

Project: Thesis Project*

Executing

Open website*

https://alsawaftech.com

	Command	Target	Value
2	*set window size*	945x692	
3	*click*	css=.section-service-1 .section-title-area	
4	*run script*	window.scrollTo(0,3645)	
5	*click*	id=fld_7428929_1	

Command	click
Target	id=fld_7428929_1
Value	
Description	

Runs: 0 Failures: 0

Log Reference

click ***locator***

Clicks on a target element (e.g., a link, button, checkbox, or radio button).

arguments:

locator - An element locator.

Figure 2.7: Selenium web application tool

2.1.8 JMeter

JMeter web application testing tool for both dynamic and static resources and dynamic web applications. JMeter web application testing is used to simulate a heavy load on a server, object or network, group or to analyze overall performance under load types. Some of JMeter features are: [39]

- ❖ The ability to load and performance tests many different applications
  - ✓ Database through JDBC
  - ✓ Message-oriented middleware
  - ✓ Mail SMTP, via JMS
  - ✓ Shell scripts or native commands
- ❖ Full features test IDE which allows fast test plan
- ❖ A complete dynamic HTML report
- ❖ Offers complete compatibility with java purity

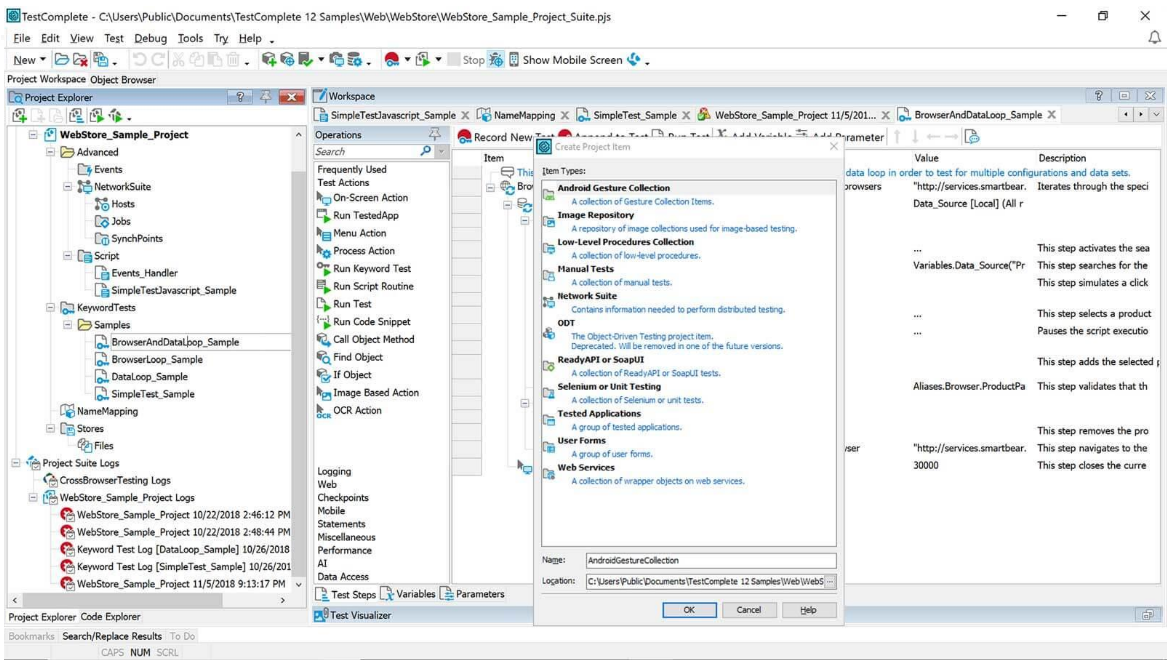
2.1.9 TestComplete

This is a web application testing tool that was developed by SmartBear Software. The tool gives the designers or the testers the ability to create automated tests for iOS, Android operating systems, and Microsoft. Usually, TestComplete contains three major modules which are mobile, Web, and Desktop. Each of the modules contains its own functionality for creating automated tests.

Some of the features are; extensions and SDK, test Visualizer: In here the tool has the ability to capture screenshots during test recording and playback, open architecture, data-driven testing, bug tracking integration, scripted testing: This feature shows that the tool has a built-in code which assists the tester in writing scripts manually, keyword testing, CPOM-based.

Some of the supported testing types by the tool are; Keyword testing, Mobile application testing, manual testing, data-driven testing, regression testing, GUI testing, distributed testing, load testing for web services, and unit testing [40].

Some of the advantages of TestComplete are; Easy to use, it is reliable, it is fast, it saves one time, it has a 24/7 supporting team, offers timely updates, trimming the cost of testing, and easy continuous integration. On the other hand, one disadvantage of the tool is that do not support MAC OS.



Item Types:

Android Gesture Collection
A collection of Gesture Collection Items.
Image Repository
A repository of image collections used for image-based testing.
Low-Level Procedures Collection
A collection of low-level procedures.
Manual Tests
A collection of manual tests.
Network Suite
Contains information needed to perform distributed testing.
ODT
The Object-Driven Testing project item.
Deprecated. Will be removed in one of the future versions.
ReadyAPI or SoapUI
A collection of ReadyAPI or SoapUI tests.
Selenium or Unit Testing
A collection of Selenium or unit tests.
Tested Applications
A group of tested applications.
User Forms
A group of user forms.
Web Services
A collection of wrapper objects on web services.

Name: AndroidGestureCollection
Location: C:\Users\Public\Documents\TestComplete 12 Samples\Web\WebS

data loop in order to test for multiple configurations and data sets.

Value	Description
"http://services.smartbear.	Iterates through the speci
Data_Source [Local] (All r	
...	This step activates the sea
Variables.Data_Source("Pr	This step searches for the
	This step simulates a click
...	This step selects a product
...	Pauses the script executio
	This step adds the selected
Aliases.Browser.ProductPa	This step validates that th
	This step removes the pro
"http://services.smartbear.	This step navigates to the
30000	This step closes the curre

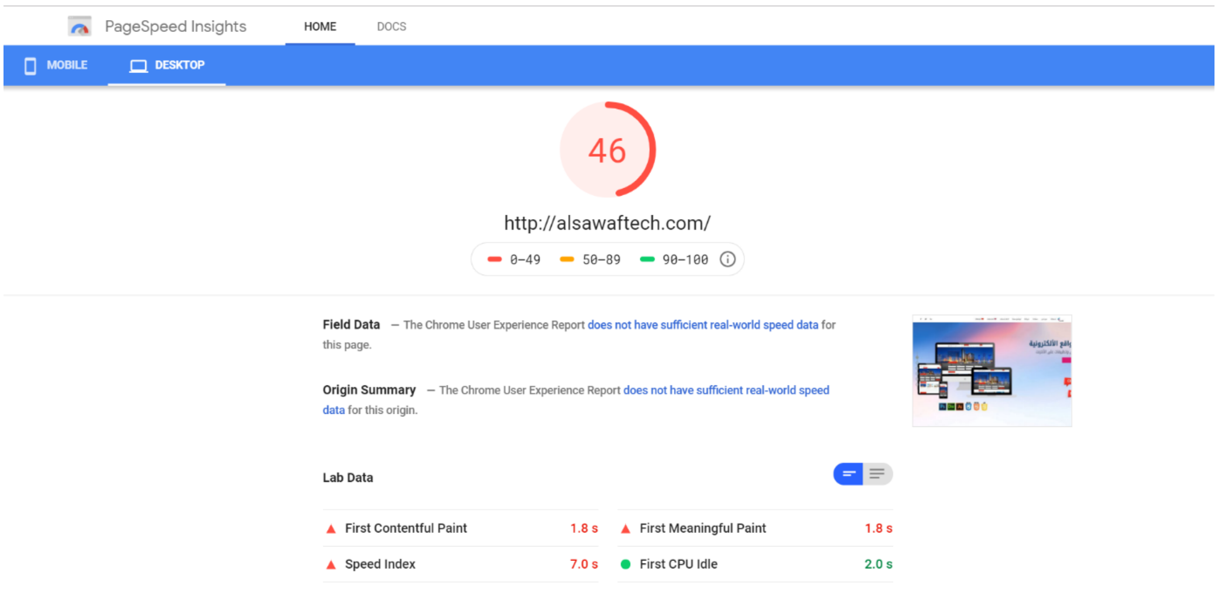
Figure 2.8: TestComplete web application tool

2.1.10 Other web application testing tools

2.1.10.1 Google Pagespeed Insight

Google Page is one of the current web application tools from Google Inc. family. The tool was designed to assist in website performance optimization. It was first introduced at a developer conference in 2010. The tool has four main components for this tool which are PageSpeed insights, PageSpeed Chrome, PageSpeed Module, and PageSpeed service. All the components are there to identify website compliance faults. It is also used to automate the adjustment process. Also, the tool can measure how a web page can be improved with performance on time to full page load.

Some of the features are; combines heads: The tool has the ability to combine several <head> tag into one tag this prevents browser workflow, removes comments: Has the ability to delete HTML comments, trims URLs: The tool has the ability to substitute absolute URLs with the relative ones, and local storage cache: These features saves inline resources. Some of the advantages of the tool offer an improved user experience, it is easy to use the tool, provides a very detailed report of its findings, includes additional languages besides English, and it is provided for free by Google. Some of the disadvantages of the tool are that it is not supposed to be used by a professional developer, and the tool rules can be very difficult to interpret [41]. With Google Pagespeed one enters the web application link on the tool to test the application. The look of the tool is as shown below in Figure 2.9



http://alsawaftech.com/

Field Data – The Chrome User Experience Report does not have sufficient real-world speed data for this page.

Origin Summary – The Chrome User Experience Report does not have sufficient real-world speed data for this origin.

▲ First Contentful Paint	1.8 s	▲ First Meaningful Paint	1.8 s
▲ Speed Index	7.0 s	● First CPU Idle	2.0 s

Figure 2.9: Google Pagespeed Insight web application tool

2.1.10.2 GTmetrix

GTmetrix is a web application tool goes into detail as it checks both Yslow and Pagespeed matrix. This means that compare with Google page insight tool, GTmetrix can assign a website grade from F to A. Also the tool offers free registration, where one can test a web application from around several different locations.

Some of the features are; the ability to carefully monitor pages and run its tests weekly, daily, or monthly, the ability to visualize performance with at least three graphs available, annotate areas of interest, and Zooms, pans, and able to set a date range to find specific performance history. Some of the advantages of the tool are; monitored analysis, multiple test options, mobile analysis, alerts and digests, and page-load analysis. One of the disadvantages of the tool is that it is not offered for free [42].

2.1.10.2 Pingdom

Pingdom is a tool that offers availability for one website, web services, and web applications. The tool uses more than seventy global polling stations to test and verify the customer's website. With the tool, one has the ability of monitoring web application performance, uptime, and user experience. The tool can analyze one website load and speed. Lastly, the tool is designed to assist a developer in making the site faster and offering an in-depth insight into a web page speed and performance expectations with email notifications or SMS [43].

Some of the features are offering uptime monitoring, offering real user monitoring, offering page speed monitoring, offering Root Cause Analysis, and offers reliable alerting. Some of the pros of the tool are; it is cost-effective, reliable, and uptime monitoring. One con for the tool is that it is not offered for free it only 14 days day free trial with limited features.

2.2 METHODOLOGY

2.2.1 Automated Software Testing tools

When application designers start researching on the best software tester tool, it is essential to first create a list of the most used software testing tools as it has been in chapter one. If we do not have a list of the software testing tools, test engineers will be wasting a lot of time downloading, evaluating, and installing testing tools that may only meet some of the requirements.

There are more advantages of automated testing over manual testing; various organizations were engaged in developing different automated testing tools. Specifically, there are two types of test tools. These are open-source test tools and commercial test tools. The open test tools are free for use, such as selenium, QTP, and TC.

2.2.2 Traditional web application methodologies

Different types of techniques have been employed in the past to test web applications. Some of these methodologies are;

- ❖ Structural testing: This is data flow analysis on web applications which applied to web applications to test if a web application is built dynamically
- ❖ Statistical testing: This is a type of web application whereby the input sequence is generated to test the interaction between a web application based on the profile use of a web application
- ❖ Mutation testing: This is a methodology of introducing faulty code which is referred to as mutants into the source code of the web application. This is done deliberately to predetermine the points and testing of the web application so as one can uncover unknown errors

- GUI interaction testing: This is a traditional web application testing whereby a web application is tested for correctness. This is done by observing the state of web application
- Hierarchical strategy: This is one of the high-level operational profile which is developed by enumerating frequency of operations
- Combinatory interaction testing: this is another type of traditional testing whereby the user uses combinations of the various techniques to first design a unique input space matrix for the web application.
- Invariant based technique: This is a type of web application methodology used by crawling web pages and formally designing the state of flow of graphs with all the possible user interaction sequence
- Cross-browser compatibility testing: This is a traditional type of testing which is done across various browsers for adherence to the expected results.
- Invariant based technique: this is a traditional type of web application testing which is done by crawling of web pages and formally designing of a state flow graph with all the possible user interaction sequences.

2.2.3 Archival research methodology

To conduct this research, the researcher adopted archival research methodology. This where data that already exists in other people's articles and to the developers of the web application tool is used to evaluate the best web application testing tool. The types of data available online are on the various web application tools, there features, advantages and disadvantages.

Most researchers prefer to use original data as they have more control over it but in this case, web application testing tools information already exists. Second, with archival data, it will be very easy, and it will take less time to evaluate and process the data. Third, archival data which already exists about web application tools have already been processed by statisticians. Forth, by adopting the archival method, the researcher will in a position to find out more than was currently gathered about web application tools. Fifth, the researcher will be able to eliminate the need to correct for issues or problems like improper sampling and observer bias. Sixth with the archival method, one can make it possible for small organizations with very limited resources to conduct thorough evaluation studies.

The organizations likely to have this information are companies such as Phonosnack who are involved in developing applications, academia who have done much research in web application testing and dissertations for advanced degrees related to software testing, funded research by web such engines like Google scholar [44].

To better understand web application testing tools. This will be done by reviewing articles; this one will get a clearer picture of software testing and help in interpreting any results which may come on the way. Second, with the archival type of research methodology, the researcher will be able to identify the best tool to use to test web applications along with a clear picture of why it is best suited for use by test engineers. Also, the researcher will be able to establish the baseline against which to measure results. Lastly, the researcher will be in a position to measure the results of the study. Lastly, the researcher will be in a position to establish a standard of comparison against which to measure the research effort; this can give a sense of how serious the issue of a web application is.

3. RESULTS

Choosing the very best web application tool for testing is not an easy task. This usually needs lots of consideration, such as whether a certain tool has relived of the integration. Besides, a testing tool has to be companionable with the execution and blueprint of the web application as well. As discussed before there are various tools which are available in the market and choose one of the tools is an intricate task.

Tables 3.1 and 3.2 below shows a summary of a comparison of the selected web-applications

Table 3.1: A summary of comparison of web-applications (Part 1)

Parameters	**Selenium**	**Webload**	**TestIO**	**Acunetix**
License	Open Source	Licensed Software	Licensed Software	Licensed Software
Cost	The tool is provided for free	Available on request	Available on request	High
Software Type	Set of APIs	Web Application	Web Application	Web Application
Ease of use & Coding experience	It requires the tester to have some programming skills to start the process of testing	Easy to use	Easy to use	Easy to use
Customer support	No professional support is provided	Dedicated	Dedicated	Dedicated
Language support	Java, C#, Ruby, Python, PHP, Perl, Javascript, R, etc.	JavaScript	-	-
Environment support	Microsoft Windows, Apple OS X, Linux	Windows, Linux	Web-based	Web-based
Browser support	All browsers	All browsers	All browsers	All browsers

Hardware requirement	The hardware required for this tool is 4x Dual-core AMD opteron	IBM-compatible PC (x86-32) with Pentium III 800 MHz (or higher) microprocessor	-	-
Hardware consumption during script execution	Low	High	-	-
Supported IDE	Eclipse, Intellij and any other IDE which supports Java	Webload IDE	-	-
Data driven framework	Yes	-	-	-
Test result generation	It won't generate any reports. TestNG will generate the report.	JUNIT, HTML, DOC, ODT, XLS, XLSX, RTF, PDF, CSV, RAW	CSV, XLS	PDF, HTML
Type of testing supported	GUI Testing Functional testing Regression testing Unit testing Keyword testing Web Testing Data-Driven Testing	Load Testing, Capacity Testing, Stress Testing, Soak Testing	Regression Testing Functional Testing Beta Testing Usability Testing Exploratory Testing Black Box Testing	Vulnerability Scanner Penetration Testing Software Web Application Security Website Security Scanner Enterprise Features External Vulnerability Scanner Network Security Scanner WordPress Vulnerability Scanner

Integration	Can be integrated with many paid or free tools.	Selenium Jenkins Dynatrace AppDynamics Amazon Web Services Perfecto Mobile New Relic	GitHub REDMINE Trello JIRA Software Pivotal Tracker Redmine	Microsoft TFS JIRA GitHub Imperva SecureSphere F5 BIG-IP Application Security Manager FortiWeb WAF Jenkins

Table 3.2: A summary of comparison of web-applications (Part 2)

Parameters	**Testing Whiz**	**QTP**	**Ranorex**	**JMeter**	**Test Complete**
License	Licensed Software	Licensed Software	Licensed Software	Open Source	Licensed Software
Cost	Available on request	High	Low	The tool is provided for free	High
Software Type	Desktop Application	Desktop Application	Desktop Application	Desktop Application	Desktop Application
Ease of use & Coding experience	It requires the tester to have some programmin g skills to start the process of testing	The tool is flexible to use and it can easily be used for regression and functional testing	Easy to use	It requires the tester to have some programmin g skills to start the process of testing	The tool is very easy to use.
Customer support	Dedicated	Dedicated	Dedicated	Free Comunities	Dedicated
Language support	-	VBS(Visual Basic Script)	No specific scripting language is used as it is written in .NET language using C hash,		JavaScript, Python, VBScript, Jscript, DelphiScript, C#, and C+

			Iron python, and VB.net		
Environment support	Microsoft Windows	Microsoft Windows	Microsoft Windows	Microsoft Windows, Apple OS X, Linux	Microsoft Windows
Browser support	All browsers	All browsers	All browsers	JMeter looks like a browser	All browsers
Hardware requirement	Intel Pentium 4 or later	1.6 Ghz or higher	2 GHz dual core	-	Intel Core i5 or Intel Core i7 (the 3rd generation)
Hardware consumption during script execution	High	High	High	Medium	High
Supported IDE	TestingWhiz IDE	UFT IDE	Ranorex Studio IDE	Eclipse, Intellij and any other IDE which supports Java	TestComplete IDE
Data driven framework	Yes	Yes	Yes	Yes	Yes
Test result generation	CSV, XLS	HTML	HTML, XML	CSV, HTML	JUnit reports, HTML, XML, PDF
Type of testing supported	Web Test Automation Mobile Testing Cross-Browser Test Automation Regression Test Automation Web Services Testing Data-Driven	GUI Testing API Testing Business Process Testing	Quality Assurance Testing Black Box Testing Jenkins Test Automation Functional Testing GUI Testing Regression Testing Keyword-Driven	Performance testing Load testing Stress testing	GUI Testing Regression testing Unit testing Keyword testing Web Testing Mobile application testing Distributed Testing Functional Testing Load testing

	Testing Database Testing Big Data Testing		Testing Data-Driven Testing		of web services Coverage Testing Data-Driven Testing Manual Testing
Integration	Jenkins Bamboo	Tasktop Sync TestComplete Connector JIRA CollabNet TeamForge Hudson Blueprint Jama Worksoft Certify VersionOne iRise Kovair Automic Calber ServiceNow	Azure DevOps/Team Foundation Server, Jenkins, Hudson, Bamboo, Team City	BlazaMeter Jenkins Meliora TestLab Maven Visual Studio Dynatrace JSUnit CloudGen	Bamboo, JIRA, Jenkins, Selenium, Team Foundation Server, QAComplete

Since the first question that comes to mind of any company that wants to buy or use any software is about the after-sales service, and technical support services, it was necessary to make some comparisons between the technical support services of the previous tools.

The Table 3.3 below shows a comparison in terms of technical support

Table 3.3: Technical Support services

Tool	Articles	Technical personnel	Bug tracking	Forums	Documentation
Webload	✓	✓		✓	✓
TestIO	✓	✓			
Acunetix	✓	✓			✓
TestingWhiz	✓	✓		✓	✓
HPE UFT	✓	✓		✓	✓
Ranorex	✓	✓		✓	✓
Selenium			✓	✓	✓
JMeter			✓		✓
TestComplete	✓	✓		✓	✓

All tools discussed in this thesis have a record feature. Usually, the tester needs to run a web application where a login page is displayed. The tester then goes ahead to enter a password and username and goes ahead to click the log in page. After a successful login page, a dashboard is displayed with grid views that show sample data. The tester then goes ahead to click on the person link in the major navigation panel that is located on the left screen. The tool the loads what tester engineers refer to as person search and the tester engineer goes ahead to search any data on the web application after clicking the search button. The page is then loaded.

A list of different techniques for test engineers to identify web elements on a web page is shown in the Table 3.4 below

Table 3.4: Overview of different Web Page Location Mechanisms Components

Tool	**Element ID**	**XPath**	**DOM**	**Group of attributes**	**Object Recognition Engine**
Selenium	✓	✓	✓	✓	
TestComplete	✓	✓		✓	✓
Ranorex	✓	✓	✓	✓	✓
JMeter	✓	✓	✓		
Testing Whiz	✓	✓	✓		✓
HPE UFT	✓	✓	✓		✓
Webload	✓	✓	✓		✓
TestIO					
Acunetix					

From all the tables provided on the comparison of the various tools, it means that there is a need for indicating the health of web application testing tools. Plotting of the number of tools vs. the type of testing supported by each tool is supported by the Figure 3.1 below.

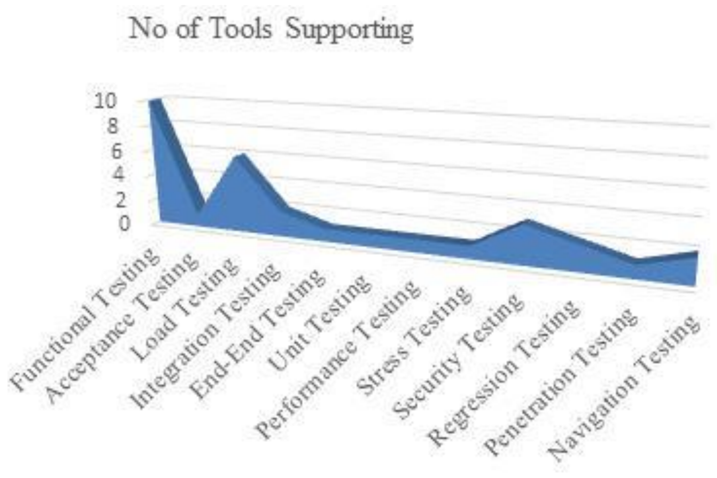


Figure 3.1: Number of tools vs the type of testing

The no. of web application tools vs. open source or licensed has been plotted by the figure shown below. The figure depicts the topmost tools which are available for web application testing and which are commercial.

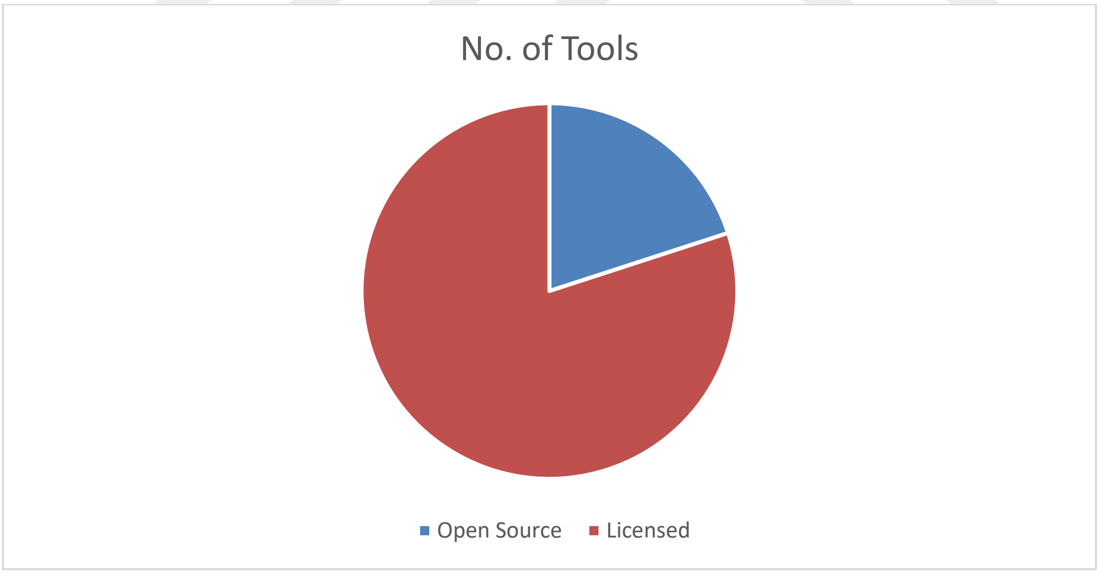


Figure 3.2: Number of open source tools vs. the licensed tool

As Selenium is the most popular tool used in automated testing processes and is the only one available free of charge for in-depth testing among the above tools, you can view the results of load testing for many cases from the Table 3.5 and Figure 3.3 below.

Table 3.5: Results for performance testing for Selenium

Test identifier	**Test case description**	**Average Mean testing Time**		
		One user	Fifty users	100 users
Test 1	Query on upload of photos	4.9	9.3	11.8
Test 2	Query rate records	11	15	13
Test 4	Query about sign-up	3.1	8.5	7.1
Test 4	Making new slide show	6	10	8
Test 5	Importing pictures from Facebook	13	24.4	31
Test 6	Importing pictures from Instagram	11.4	20	30

Test 7	Importing pictures form Flicker	3.8	5.8	7.5
Test 8	Saving slideshow	8.7	9.0	29
Test 9	Query for sharing slideshow on Flicker	5.4	8.4	10.2
Test 10	Query for sharing slideshow on Instagram	5	9.3	11.7
Test 11	Query for sharing slideshow on Facebook	3.7	8.6	9.4
Test 12	Query for downloading templates	5.3	7.6	6.7

Test 13	Query for uploading videos	14	23	32

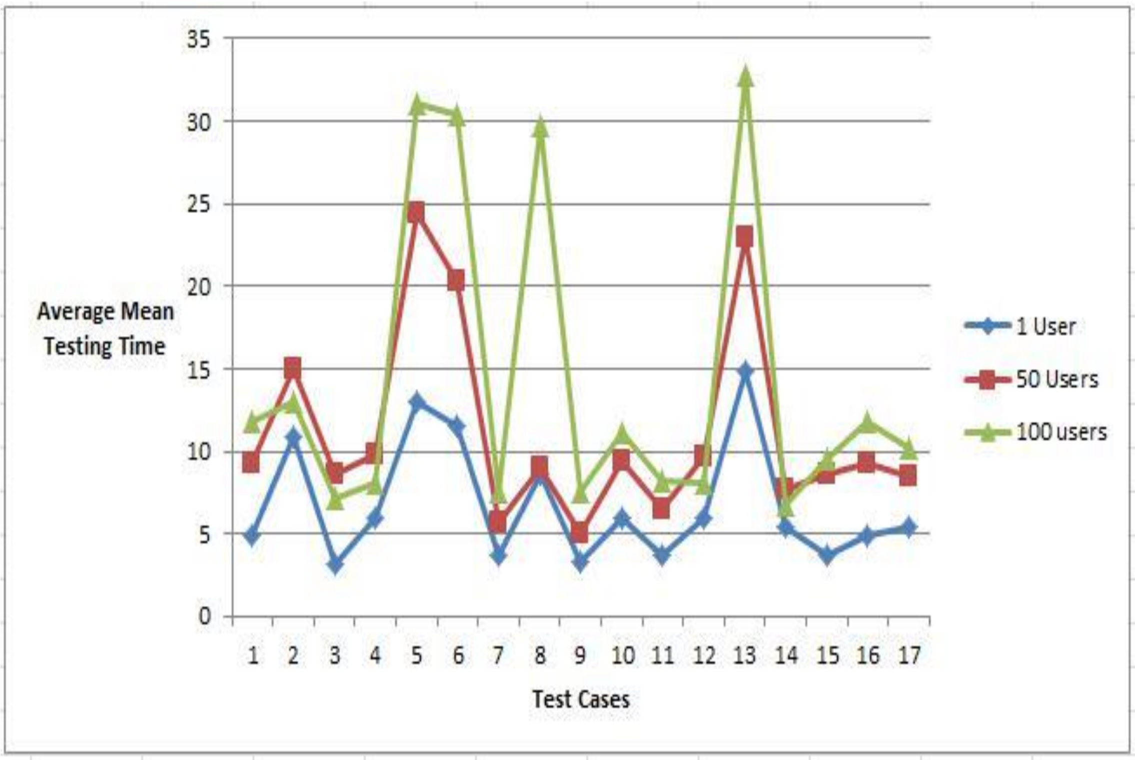


Figure 3.3: A graph showing the average mean time for one, fifty, and one hundred users

Also From the previous sections, it is evident that testing plays a very important role in both software development and web application development. Usually, web applications tend to take faster and quicker release cycle, thus making the process of web application testing very challenging. The major issues which have been put across by various web application developers are bug detection efficiency and cost-efficiency.

If I divided the tools I selected in this article by type; we will find the classification as follows:

- Web Application Testing: Selenium, TestComplete, QTP, Ranorex and TestingWhiz
- Load Testing: JMeter and WebLoad
- Security Testing: Acunetix

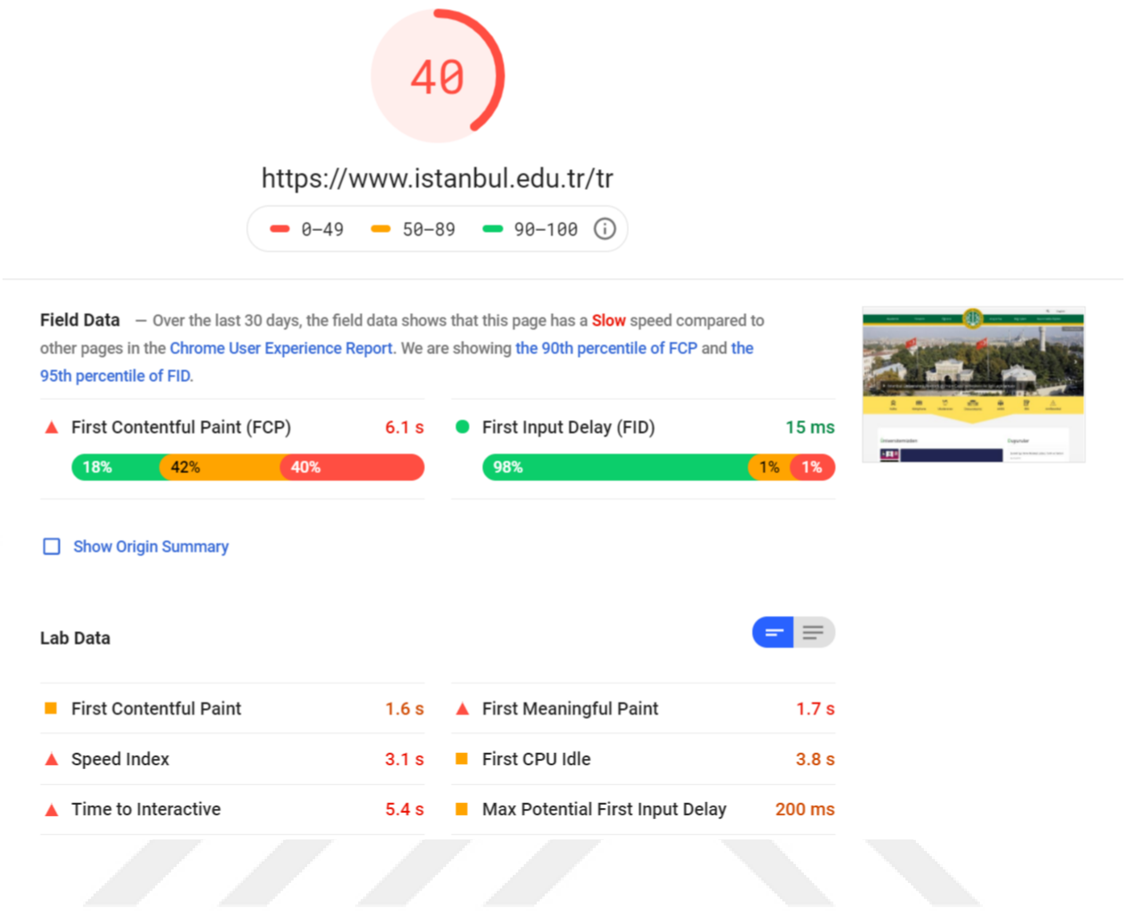
- Testing as a Service: TestIO
- Automated Usability Testing Tools: Google Insight, GTmetrix and Pingdom

Usability is a critical variable in web application performance measurement. Table 3.6 below shows a summary of a comparison of the selected Automated Usability Testing Tools

Table 3.6: Automated Usability Testing Methods Comparative Analysis

Feature	GTmetrix	PageSpeed	Pingdom
Test Locations	7 test regions	Test Region Unknown	7 test regions
Scores and Recommendations	Based on PageSpeed and YSlow 27 PageSpeed recommendations 18 YSlow recommendations	Separated into Opportunities, Diagnostics, and Passed Audits 20 “Audits/Opportunities”	Approx 11 recommendations Likely based on PageSpeed
Time to Stop Test	Fully Loaded Time (default) Onload time (optional)	First Contentful Paint and DOM Content Loaded	Onload time (only option)
Real Browsers vs Headless/Emulated Browser	Real Browsers Firefox (default) Chrome Chrome (Android)	Emulated browser	Real browser
Connection throttling options	Unthrottled by default	-	No connection throttling options
HTTP/2 support	Yes	No	No
Test resolutions	Multiple	Unknown exact dimensions	1024×878
Hardware provision	Consistent for every test location	Unknown	Unknown
Historical tracking	Yes	No	No
Page Size	Yes	No	Yes

Accordingly, Istanbul university website is tested using the above tools and the results were different due to the different methods used in calculating the evaluation of the site as shown in the previous table, the results were as follows in Figures 3.4, 3.5, 3.6 below


40

https://www.istanbul.edu.tr/tr

0–49 50–89 90–100

Field Data — Over the last 30 days, the field data shows that this page has a **Slow** speed compared to other pages in the Chrome User Experience Report. We are showing the 90th percentile of FCP and the 95th percentile of FID.

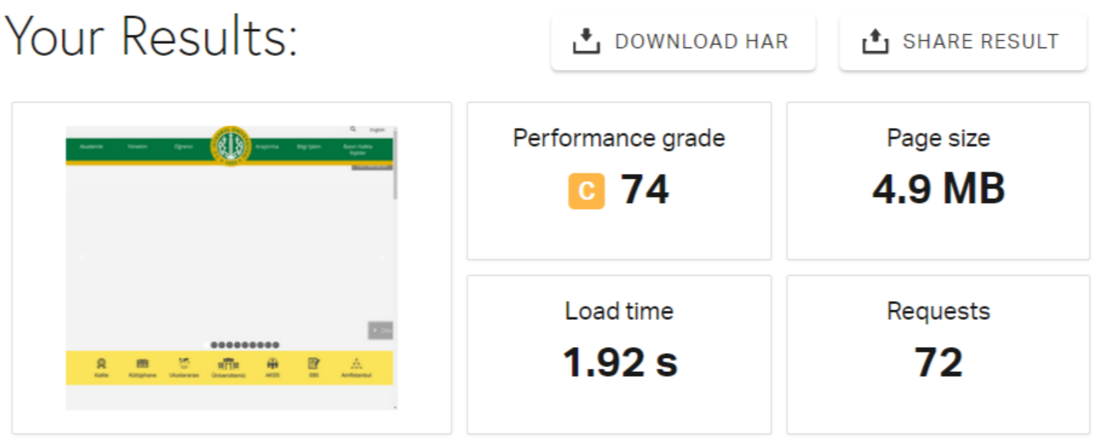
First Contentful Paint (FCP)	6.1 s	First Input Delay (FID)	15 ms
18% 42% 40%		98% 1% 1%	

Show Origin Summary

Lab Data

First Contentful Paint	1.6 s	First Meaningful Paint	1.7 s
Speed Index	3.1 s	First CPU Idle	3.8 s
Time to Interactive	5.4 s	Max Potential First Input Delay	200 ms

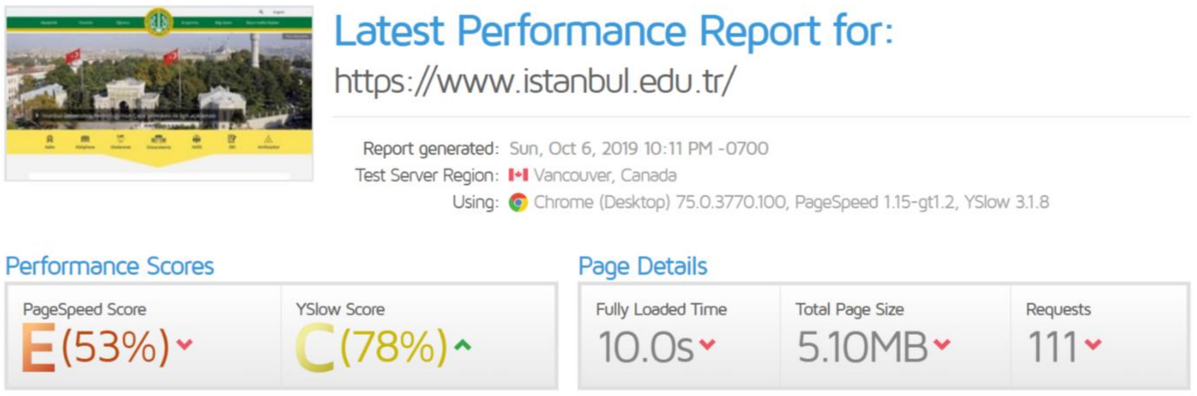
Figure 3.4: University website test results with Google PageSpeed Insight


Your Results:

DOWNLOAD HAR SHARE RESULT

Performance grade C 74	Page size 4.9 MB
Load time 1.92 s	Requests 72

Figure 3.5: University website test results with Pingdom


Latest Performance Report for:
https://www.istanbul.edu.tr/

Report generated: Sun, Oct 6, 2019 10:11 PM -0700
Test Server Region: Vancouver, Canada
Using: Chrome (Desktop) 75.0.3770.100, PageSpeed 1.15-gt1.2, YSlow 3.1.8

Performance Scores

PageSpeed Score	YSlow Score
E (53%)	C (78%)

Page Details

Fully Loaded Time	Total Page Size	Requests
10.0s	5.10MB	111

Figure 3.6: University website test results with GTmetrix

Based on the previous results, we note that the university site has received a weak assessment in both Google and GTmetrix, but on the other hand, received a good evaluation from Pingdom due to different policies in the assessment of priorities among these tools, for example the biggest difference between these tools in the evaluation for size of the images compared to the display mechanism in the space allocated to them, was evaluated by Pingdom good while each of the other two tools were evaluated very poorly.

However, running those tests takes a while based on the algorithms they use on each website, and on their servers response, therefore I made a small comparison between these tools test time in seconds:

Table 3.7: Automated Usability Testing Tools Run Speed

	Google Pagespeed	**GTmetrix**	**Pingdom**
1st run	24.16	30.52	23.19
2nd run	16.16	35.83	18.11
3rd run	23.26	34.91	13.06

Based on the result above in Table 3.7, it seems like Pingdom did use caching while testing the websites, or they have more available servers and internet lines to improve their results.

Back to the web application testing tools, while installing them, it's noticed that there's a huge difference between their application size for installation files, program data files and RAM usage in Table 3.8:

Table 3.8: Web application testing tools resource usage

	Installation File	**Program Files**	**RAM Usage**
Selenium	10.2 MB	15.8 MB	60 MB
Ranorex	280 MB	548 MB	192 MB
TestComplete	623 MB	1.17 GB	279 MB
Testing Whize	344 MB	476 MB	278 MB

Moreover, when comparing the code generated between the Selenium, TestComplete and Ranorex programs, the difference between them is obvious. Selenium generates a full source code that we can run with many available IDE's, while the code generated by TestComplete is only functions because this code will only run through the TestComplete IDE. This is a simple code generated by TestComplete for browsing a website:

```
function Test1()
{
  //Clicks the 'BrowserWindow2' object.
  Aliases.browser.BrowserWindow2.Click(399, 74);
  //Opens the specified URL in a running instance of the specified browser.
  Browsers.Item(btChrome).Navigate("https://www.istanbul.edu.tr/tr/_");
  //Navigates to the ''https://www.istanbul.edu.tr/tr'' address.
  Aliases.browser.ToUrl("https://www.istanbul.edu.tr/tr");
  //Clicks the 'Akademik
  Aliases.browser.pageStanbulNiversitesiTarihtenGe.navHeader.ClickItem("Akademik\n");
  //Clicks an item of the 'link' drop-down control.
  Aliases.browser.pageStanbulNiversitesiTarihtenGe.link.ClickItem("Fakülteler");
  //Clicks the 'link' link.
  Aliases.browser.pageStanbulNiversitesiTarihtenGe2.link.Click();
  //Waits until the browser loads the page and is ready to accept user input.
  Aliases.browser.pageTr2.Wait();
}
```

On the other hand, this is the code generated by Selenium while executing the same test:

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
public class BrowsewebsiteTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  public String waitForWindow(int timeout) {
    try {
      Thread.sleep(timeout);
    } catch (InterruptedException e) {
      e.printStackTrace();
    }
    Set<String> whNow = driver.getWindowHandles();
    Set<String> whThen = (Set<String>) vars.get("window_handles");
    if (whNow.size() > whThen.size()) {
      whNow.removeAll(whThen);
    }
    return whNow.iterator().next();
  }
  @Test
  public void browsewebsite() {
    driver.get("https://www.istanbul.edu.tr/tr/_");
    driver.manage().window().setSize(new Dimension(1536, 822));
    driver.findElement(By.linkText("Akademik")).click();
    driver.findElement(By.linkText("Fakülteler")).click();
    vars.put("window_handles", driver.getWindowHandles());
    driver.findElement(By.cssSelector(".tab-pane:nth-child(1) tr:nth-child(2) >
td:nth-child(2) > a")).click();
    vars.put("win5163", waitForWindow(2000));
    driver.switchTo().window(vars.get("win5163").toString());
```

However, the code generated by Ranorex can be viewed like below, while it’s not editable, they do provide two files, one for the recording methods and another one to call these methods if needed:

```
void ITestModule.Run()
        {
            Mouse.DefaultMoveTime = 300;
            Keyboard.DefaultKeyPressTime = 20;
            Delay.SpeedFactor = 1.00;

            Init();

            Report.Log(ReportLevel.Info, "Mouse", "Mouse Left Click item 
'WebAutomationAutomatedWebsiteWeb.Pane' at 358;85.", 
repo.WebAutomationAutomatedWebsiteWeb.PaneInfo, new RecordItemIndex(0));
            repo.WebAutomationAutomatedWebsiteWeb.Pane.Click("358;85");
            Delay.Milliseconds(0);

            Report.Log(ReportLevel.Info, "Keyboard", "Key sequence 
'istanbul.edu.tr{Return}' with focus on 'WebAutomationAutomatedWebsiteWeb1'.", 
repo.WebAutomationAutomatedWebsiteWeb1.SelfInfo, new RecordItemIndex(1));
            repo.WebAutomationAutomatedWebsiteWeb1.Self.EnsureVisible();
            Keyboard.Press("istanbul.edu.tr{Return}");
            Delay.Milliseconds(0);

            Report.Log(ReportLevel.Info, "Mouse", "Mouse Left Click item 
'ApplicationUnderTest.Akademik' at 148;58.", repo.ApplicationUnderTest.AkademikInfo, new 
RecordItemIndex(3));
            repo.ApplicationUnderTest.Akademik.Click("148;58");
            Delay.Milliseconds(0);

            Report.Log(ReportLevel.Info, "Mouse", "Mouse Left Click item 
'ApplicationUnderTest.Fakuelteler' at 100;23.", 
repo.ApplicationUnderTest.FakuelteleriInfo, new RecordItemIndex(4));
            repo.ApplicationUnderTest.Fakuelteler.Click("100;23");
            Delay.Milliseconds(0);

            Report.Log(ReportLevel.Info, "Mouse", "Mouse Left Click item 
'NewNotification.DismissTextBlock' at 6;8.", repo.NewNotification.DismissTextBlockInfo, 
new RecordItemIndex(5));
            repo.NewNotification.DismissTextBlock.Click("6;8");
            Delay.Milliseconds(0);

        }
```

It is also important to note that the TestComplete program needs less code to search for a particular element because of the technique of automatic identification of elements in it using artificial intelligence. In contrast, Selenium program is still dependent on determining the elements by the element id or XPath.

4. DISCUSSION

From chapter two, it is evident that the automation tools have been there for several years. Besides, web application testing tools work against web pages in an HTML format. Using the web application tools, the test engineers can record a functional test and playback if needed. There are various factors to consider when selecting a tool; first, a web application tool needs to be reliable to have a playback feature especially when testing dynamic web applications. The web application testing tool ought to execute the test during the test without any errors. Second web application testing tools need to have the ability to export the resulting tests as programmable scripts so that the test engineers can understand, refactor, and maintain the modules. Third, a web application testing tool needs to be diverse; the diversity provided by the tool helps in finding the elements on a web application especially the dynamic applications. The fourth factor to consider when selecting a web application is the ability of the tool to support cross-browser compatibility. Most of the web applications have a priority feature supported by various browsers like Mozilla Firefox and Chrome. Forth, a web application needs to be effective especially when it comes to waiting for the web application to load completely before they start manipulating and accessing the web application elements. Lastly, test engineers need to consider the technical support services of the web application testing tool. Most of the web application has technical documentation that has been provided by the respective vendors but without the technical support when a need arises. This results in the technical team running into problems or issues while using the automation tool yet the major reason for using a certain web application testing tool is saving time and speed, and the test engineers need not waste much time in developing a solution which someone else already knows. The technical support services besides technical documentation which need to be there are bug tracker systems, technical support services, and forums.

5. CONCLUSION AND RECOMMENDATIONS

From this thesis, it is evident that the web has a very significant impact on all features of our society. As organizations and companies rely more on the web, web-application has become increasingly more important. On the other hand, sophisticated attacks have grown, which has raised the need of coming up with web-applications that are secure and not prone to attacks. This can only be achieved by using the best web-application testing tools to test the applications before using the application.

Since we have gone through many differences between testing tools ' pros & cons, I hope we've been successful in clearing the image of exactly what you expected. Selenium and JMeter, the open-source tools, have several industry rivals, yet it is considered one of the best in the business. However, if we go beyond the financial costs, we will find that TestComplete and Webload tools will be better in terms of features in addition to technical support services, with the possibility of multi-platform testing at the same time more easily without the need for specialized developers. Moreover, in terms of vulnerability tests, it is necessary to use the Acunetix, and the tests for the usability should be with the use of GTmetrix more comprehensive among the previous tools.

Some of the limitations of this study are; first, most of these tools are not provided for free. Therefore the study could not offer a clear analysis of these tools from the demo which was provided from the various sources. Second, the process of testing web application tools requires time, and also needs to create a web application or the webpage and test it using the various tools to ascertain the listed features to get more specific results.

To do further research about software testing, I think we have to make a plan based on two parts. First, is to expand and deepen more in the content of the current research, and the second is to invent new technologies to remove the existing complexities in the automated testing. As for the expansion and deepening of the current research content, the research categories discussed from the performance test to usability tests, security tests, and functional tests. On the other hand, only web-testing technologies have reviewed, while nowadays most of the companies do create projects with cross-platform applications, therefore it is necessary to expand in these areas, and prepare a research for each type of test listing all the tools available and accesses its precise details to reach an accurate scientific result to choose the best tool among these tools, and propose the optimal options to reach the best performance in each area. As for inventing new technologies in the area of automated testing, I think it is time to enter the artificial intelligence into this field to make it more creative. At the moment, testers or programmers have to write or record test cases manually, even in modification, while changing the names of options, windows or adding new windows or options, the user will have to write new test cases or modify many previous test cases, which is stressful for large companies, it must be replaced by artificial intelligence techniques for these systems to create automatic test cases based on additions or modification according to certain standards, thus, companies will shorten long hours of work and retain fewer employees, saving large amounts of costs on companies.

REFERENCES

[1] S. Gojare, R. Joshi, and D. Gaigaware, “Analysis And Design Of Selenium Webdriver Automation Testing Framework,” *Procedia Comput. Sci.*, vol. 50, pp. 341–346, 2015.

[2] A. Petukhov and D. Kozlov, “Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing,” *Appl. Secur. Conf.*, 2008.

[3] A. Bertolino and E. Marchetti, “A Brief Essay on Software Testing,” *Area*, pp. 1–14, 2003.

[4] A. Bertolino, “Software testing research: Achievements, challenges, dreams,” *FoSE 2007 Futur. Softw. Eng.*, no. September, pp. 85–103, 2007.

[5] S. Kundu, “Web Testing: Tool, Challenges and Methods,” *Int. J. Comput. Sci. Issues*, vol. 9, no. 2, pp. 481–486, 2012.

[6] A. Hedayati, M. Ebrahimzadeh, and A. A. Sori, “Investigating into Automated Test Patterns in Erratic Tests by Considering Complex Objects,” *Int. J. Inf. Technol. Comput. Sci.*, vol. 7, no. 3, pp. 54–59, 2015.

[7] B. Falah, M. Akour, and N. El Marchoum, “Testing patterns in action: Designing a test-pattern-based suite,” *Int. Rev. Comput. Softw.*, vol. 10, no. 5, pp. 489–494, 2015.

[8] S. Elbaum, S. Karre, and G. Rothermel, “Improving web application testing with user session data,” *Proc. - Int. Conf. Softw. Eng.*, pp. 49–59, 2003.

[9] S. Gupta, “Review Paper on Comparison of Automation Testing Tools Selenium and QTP,” vol. 5, no. 2, pp. 55–57, 2015.

[10] M. Monier and M. M. El-mahdy, “Evaluation of automated web testing tools,” *Int. J.*

Comput. Appl. Technol. Res., vol. 4, no. 5, pp. 405–408, 2015.

[11] V. Rufus, *AngularJS Web Application Development Blueprints*. 2014.

[12] R. Angmo and M. Sharma, “Performance evaluation of web based automation testing tools,” *Proc. 5th Int. Conf. Conflu. 2014 Next Gener. Inf. Technol. Summit*, pp. 731–735, 2014.

[13] T. Parviainen, *Real-time Web Application Development using Vert.x 2.0*. 2013.

[14] Juraj Húska, “Automated Testing of the Component-based Web Application User Interfaces,” 2012.

[15] G. A. Di Lucca and A. R. Fasolino, “Testing Web-based applications: The state of the art and future trends,” *Inf. Softw. Technol.*, vol. 48, no. 12, pp. 1172–1186, 2006.

[16] N. Antunes and M. Vieira, “Penetration testing for web services,” *Computer (Long. Beach. Calif).*, vol. 47, no. 2, pp. 30–36, 2014.

[17] E. V. Sandin, N. M. Yassin, and R. Mohamad, “Comparative Evaluation of Automated Unit Testing Tool for PHP,” *Int. J. Softw. Eng. Technol.*, vol. 03, no. 2, pp. 7–11, 2016.

[18] X. Yuan, M. B. Cohen, and A. M. Memon, “GUI interaction testing: Incorporating event context,” *IEEE Trans. Softw. Eng.*, vol. 37, no. 4, pp. 559–574, 2011.

[19] N. Gogna, “Study of Browser Based Automated Test Tools WATIR and Selenium,” *Int. J. Inf. Educ. Technol.*, vol. 4, no. 4, pp. 336–339, 2014.

[20] M. Sharma and R. Angmo, “Web based Automation Testing and Tools,” *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 1, pp. 908–912, 2014.

[21] Y. F. Li, P. K. Das, and D. L. Dowe, “Two decades of Web application testing - A

survey of recent advances," *Inf. Syst.*, vol. 43, pp. 20–54, 2014.

[22] S. Paydar and M. Kahani, "An Agent-Based Framework for Automated Testing of Web-Based Systems," *J. Softw. Eng. Appl.*, vol. 04, no. 02, pp. 86–94, 2011.

[23] S. Al-Zain, D. Eleyan, and Y. Hassouneh, "Comparing GUI Automation Testing Tools for Dynamic Web Applications," *Asian J. Comput. Inf. Syst.*, vol. 01, no. 02, pp. 2321–5658, 2013.

[24] D. R. Lakshmi and S. S. Mallika, "A Review on Web Application Testing and its Current Research Directions," *Int. J. Electr. Comput. Eng.*, vol. 7, no. 4, pp. 2132–2141, 2017.

[25] J. Križanić, A. Grgurić, M. Mošmondor, and P. Lazarevski, "Load testing and performance monitoring tools in use with AJAX based web applications," in *MIPRO 2010 - 33rd International Convention on Information and Communication Technology, Electronics and Microelectronics, Proceedings*, 2010.

[26] L. Dukes, X. Yuan, and F. Akowuah, "A case study on web application security testing with tools and manual testing," in *Conference Proceedings - IEEE SOUTHEASTCON*, 2013.

[27] H. Mahmood and M. Sirshar, "A Case Study of Web Based Application by Analyzing Performance of a Testing Tool," *Int. J. Educ. Manag. Eng.*, vol. 7, no. 4, pp. 51–58, 2017.

[28] A. M. F. V De Castro, G. A. Macedo, E. F. Collins, and A. C. Dias-Neto, "Extension of Selenium RC tool to perform automated testing with databases in web applications,"

2013 8th Int. Work. Autom. Softw. Test, AST 2013 - Proc., pp. 125–131, 2013.

[29] I. Altaf, J. A. Dar, F. U. Rashid, and M. Rafiq, "Survey on selenium tool in software testing," in *Proceedings of the 2015 International Conference on Green Computing and Internet of Things, ICGCIoT 2015*, 2016.

[30] J. Offutt, "Quality attributes of Web software applications," *IEEE Softw.*, 2002.

[31] D. Chmurciak, "Automation of regression testing of web applications," 2013.

[32] L. Xu, B. Xu, Z. Chen, J. Jiang, and H. Chen, "Regression Testing for Web Applications Based on Slicing," in *Proceedings - IEEE Computer Society's International Computer Software and Applications Conference*, 2003.

[33] Acutenix, "What is SQL Injection (SQLi) and How to Fix It," *Acutenix.com*, 2015. .

[34] S. Miller and G. Pupedis, "Spatial interface design for the web - A question of usability," *Cartography*, 2002.

[35] S. Gunasekaran and V. Bargavi, "Survey on Automation Testing Tools for Mobile Applications," *Int. J. Adv. Eng. Res. Sci.*, 2015.

[36] V. Garousi, A. Mesbah, A. Betin-Can, and S. Mirshokraie, "A Systematic Mapping Study Of Web Application Testing," *Inf. Softw. Technol.*, vol. 55, no. 8, pp. 1374–1396, 2013.

[37] F. Wang and W. Du, "A test automation framework based on WEB," in *Proceedings - 2012 IEEE/ACIS 11th International Conference on Computer and Information Science, ICIS 2012*, 2012.

[38] Da Zhang, "End to end testing using integrated tools," 2012.

[39] Sergey Uspenskiy, “A Survey And Classification Of Software Testing Tools,” 2010.

[40] N. Dubey and S. Shiwani, “Studying and Comparing Automated Testing Tools; Ranorex and TestComplete,” *Int. J. Eng. Comput. Sci.*, 2014.

[41] S. Kaur, K. Kaur, and P. Kaur, “An Empirical Performance Evaluation of Universities Website,” *Int. J. Comput. Appl.*, vol. 146, no. 15, pp. 10–16, 2016.

[42] N. Kumar, A. Kalia, and R. Kumar, “Evaluation of WebPages performance W.R.T UI/UX developed using different frameworks,” *Int. J. Eng. Adv. Technol.*, vol. 8, no. 6, pp. 575–579, 2019.

[43] S. Gopinath, V. Senthooran, N. Lojenaa, and T. Kartheeswaran, “Usability and accessibility analysis of selected government websites in Sri Lanka,” in *Proceedings - 2016 IEEE Region 10 Symposium, TENSYMP 2016*, 2016.

[44] J. S. R, “A Feature Model For Web Testing Tools,” 2015.

CURRICULUM VITAE

Personal Information	
Name Surname	MHD KHALED AL SAWAF
Place of Birth	DAMASCUS
Date of Birth	13.08 1989
Nationality	❒ T.C. [✓ Other:] Syria
Phone Number	05379132254
Email	Khaled.sawwaf@gmail.com
Web Page	

Educational Information	
B. Sc.	
University	Near East University
Faculty	Engineering
Department	Information System Engineering
Graduation Year	09.07.2014

M. Sc.	
University	Istanbul University-Cerrahpasa
Institute	Graduate Studies in Science and Engineering
Department	Computer Engineering
Programme	Computer Engineering