IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 4, May 2022
Impact Factor: 6.252

An Interactive System for Web Testing through Selenium Web Driver and Web Page Downloading

Mr. Akash Satish Ugale[1], Mr. Roshan Gorakshnath Arote[2],
Ms. Samiksha Anil Bombale[3], Prof. Shinde Pavan B.[4]
Students, Department of Information Technology[1,2,3]
Faculty, Department of Information Technology[4]
Amrutvahini Polytechnic, Sangmner, Maharashtra, India

Abstract: *The deployment of source code is usually preceded by rigorous testing that needs to be performed to ensure that the source code is working as per the design requirements. Testing is highly useful as it can unearth the various problems and redundancy in the functioning of the source code. The testing currently is being performed through the use of Selenium WebDriver tool, which is a highly useful for automation of the testing approach. The test cases in this tool can be easily written and performed faster which can be highly useful in improving the testing efficiency. The main advantage of using the Selenium WebDriver for the testing procedure, is the fact that the tester does not need to fully understand it before using it, which makes this framework extremely user friendly. The proposed approach further enhances the testing paradigm by introducing an additional functionality for different kind of testing like automation testing, load testing and unit testing. Addition to this proposed system also deploys the auto download of the web pages for the given http URL to make the process of web page development easier for the beginners.*

Keywords: Selenium WebDriver, Unit testing, Automation testing and unit testing, Site crawler

I. INTRODUCTION

Currently, there are ample software systems based on web applications that are complex in nature. It is difficult to manually test such complicated applications, so automated testing is essential in such scenarios. This automation testing framework will be implemented using Selenium WebDriver tool. The fundamental concept in this project is to test single functionality of numerous websites by usage of generic test case. This framework tester can write test cases in a simple way and efficiently in less time. As this framework is user-friendly, the tester does not need to understand the selenium webdriver tool in detail. It is very easy to maintain and repair the test suite for new release of the application using this framework.

A single web page consists of a huge set of significant functionalities that leads to the execution of web service operations. Besides, it becomes difficult to test all of these functionalities that are implemented in a web page service as they are complicated in nature. This is the reason why there are many automation tools that are available to help the testers and to facilitate the execution of testing processes.

The well-known and most adopted tools in the industry are namely Selenium and Open Script. Recent studies have shown that Selenium is very helpful for testing applications which are browser based. It runs specific tests on diverse installed browsers and return results, alerting you to failures in browsers as they crop up.

In Functional Test Automation, algorithms are defined and implemented to estimate how can you automate any software related web service product under test, then the machine can identify or predict the changes and adapt those changes to the suitable test cases . The proposed system will test websites based on the search functionality without the need to revise any test codes.

This project aims to implement a framework that automates functional testing. This mainly emphasizes on providing better system quality product by saving the time, giving faster feedback cycle and validate newly developed features. In recent times, an automated test suite has the capability to simulate hundreds of virtual users all interacting with your application and this would never be possible with manual testing.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 4, May 2022
Impact Factor: 6.252

This framework guides the developer to analyze their code in depth as it has the feature that allows to take a screen shot. The framework produces the customized test report to tester, thus it is easy to maintain and repair the test suite for new release of the application using this framework. The system can also get dynamically accustomed to the changes made in the website. Another aspect of the proposed methodology is the inclusion of the selenium for performing automated testing of the source code of the application. The Selenium web driver tool is being used to perform the testing of the code in a variety of different formats, such as performance testing, which evaluates the performance of the code if it is up to the mark, load testing to determine how the system behaves under high load conditions, stress testing to achieve evaluation of the stress which can be defined as actively trying to break functionality just by rigorous usage of the code and regression testing to understand the code behavior on modification or implementation of any new features or additions.

The Literature Survey component of this research paper examines previous work. Section 3 delves into the approach in depth, while section 4 focuses on the outcomes evaluation. Finally, Section 5 brings this report to a close and gives some hints for future research.

II. LITERATURE SURVEY

M. Badri et al investigates the idea of clone refactoring and its influence on the length of test code in object-oriented languages. To build an approach suitable of estimating the effect, the researchers employed linear regression in combination with different machine learning algorithms such as K-Nearest Neighbors, Nave Bayes Theorem, and Random Forest [1]. The information was acquired exhaustively using an open source Java-based software that has undergone some form of clone-refactoring. The approach has been measured in comparison to conventional procedures and yields much greater efficiency.

J. Zhao et al describes the multi-core computation revolution, which has put large multi-core CPUs with in disposal of the general public at an unbelievable reasonable price. It introduced the notion of parallelization and, more crucially, the OpenMP protocol. Because of the fast expansion of information in this era of the internet, memory and computing capabilities are restricted [2]. As a result, the academics used a MapReduce component in combination with the OpenMP packages to operate parallelly in a cloud infrastructure, decreasing the burden on resources, mainly computational and memory.

G. Kaur et al indicates that one of the essential skills required of computer programmers and software designers is code organization. As a result, the code is refactored in order to reduce cost of operation. Because the complexity of the code grows mostly with number of functionalities introduced, it must be mitigated by code refactoring [3]. The practice of code restructuring improves the manageability of the code and reduces its complexity, which is helpful to the programmer. The suggested method works as expected and is far more effective than existing methodologies.

B. Lin et al presents the notion of code simplicity, since much study has been performed in this field to establish that source code, just like many the other human languages, is consistent and largely repetitious. This validates the hypothesis that unconventional code is assumed to be buggy. According to the experts, this approach may be used to accomplish advanced code restructuring [4]. Because problematic code has been declared abnormal, code restructuring may be readily accomplished while retaining the code's quality. The approach is one-of-a-kind and outperforms commonly used code refactoring technologies.

M. Saca et al builds on the ideas of Code Refactoring by describing the many stages necessary to accomplish optimal refactoring. The researchers have meticulously discussed the different principles and the fundamental operation of the code refactoring operation in inadequately constructed computers [5]. Improperly conceived programs have productivity problems, and code rewriting can significantly improve efficiency while lowering maintenance fees.

J. Zhao et al argues that developments in the field of Big Data have created a significantly greater aggressive environment in which conventional storage solutions are now being replaced in favor of Big Data since they could indeed maintain pace with the platform's reaction speed and reliability. The majority of organizations are now migrating their processes from local machines to the cloud in order to profit from MapReduce advantages in their processes. As a result, the researchers designed a code refactoring approach for sequential code to a Mapreduce framework for migration of data to the cloud. [6]

U. Devi et al in this research the ubiquity of code duplication as a key problem in the field of object-oriented languages is examined. Although it has been an issue for other methods, including the SPL (Software Product Line), since one of the

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 4, May 2022
Impact Factor: 6.252

key problems is maintainability [7]. SPL has indeed been heavily affected mostly by code cloning process, and the methodologies for evaluation have been erratic in their application. As a result, the researchers evaluated a large number of studies in this field in order to accomplish maintainability with high levels of e□ciency.

J. Vedurada et al investigates the notion of code refactoring, since it are among the most crucial characteristics for software maintenance. Code is often refactored to preserve the application and enhance readability while not interfering with its primary purpose. This is a highly intricate and challenging task, since among the most essential phases in the procedure is identifying refactoring possibilities in the code [8]. The researchers accomplished this using "Replace Type Code with State" and "Replace Type Code with Subclass" in real-time Java programs, as these are two of the foremost effective Java-based refactoring mechanisms.

J. Kanwal et al analyses the incidence of code duplication and the consequences for applications when programmers rewrite various portions of code when reworking the code, they typically use a completely distinct method for dealing with clones [9]. The researchers conducted a comprehensive research in order to understand how software engineers operate on code refactoring. According to the report's results, copies of the very same class's code are systematically refactored, but only a tiny set of code is refactored in a specified timeframe G. Szoke et al indicates that in order to preserve the integrity of the software and minimize degradation, the data must be refactored on a regular basis. Refactoring is a demanding and involved procedure since developers must continuously discover potential areas throughout the code which require modification. However, constantly rewriting the code seems to have its own drawbacks because the updated code must be checked for inconsistency and errors [10]. As a result, the academics suggested an approach for refactoring that makes use of code smells. The proposed solution has been extensively tested and evaluated to function effectively for the purpose of refactoring.

III. Proposed Methodology

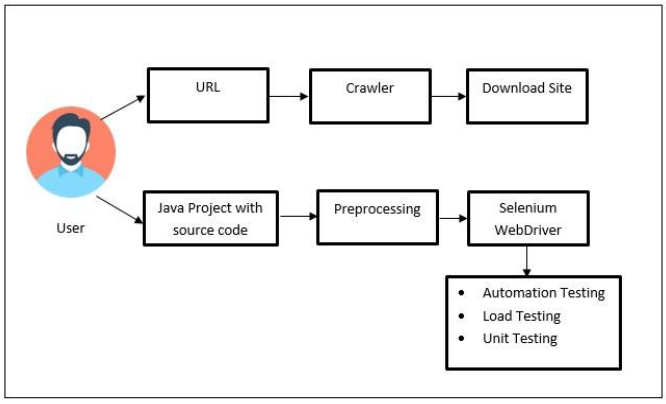

Figure 1: Proposed Work System Overview

In this section the testing has been performed on the webpages, using the Selenium WebDriver API for the same input project that has been fed to code refactoring. This web testing is performed on 3 different types of testing using Selenium WebDriver as explained below.

3.1 Automation Testing

The automation testing is being performed to determine the transition of the webpages from the source page to the destination page. The testing procedure provides the system with the respective credentials for the login page (username and password) or the source page and provides the destination page for the same. Using the Selenium WebDriver API the automation testing logs into the webpage using these credentials. After successful login if the webpage navigates to the correct destination page, the test is deemed as passed, otherwise it is termed as fail.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 4, May 2022
Impact Factor: 6.252

3.2 Load Testing

The load testing has also been performed to determine the level of load that can be handled by the webpage that is being deployed on the Glassfish server. The Load Testing is being performed by initiating the webpage for N instances. If the webpage loads perfectly for the N instances, then the test is deemed as a pass and if the webpage fails to load for any of the N instances, then it is considered as a fail. The load testing determines the performance of the webpage under stress and high load scenarios and evaluates its performance accordingly.

3.3 Unit Testing

The Unit Testing is performed to ascertain is there is the presence of an element or a string on the webpage. This is crucial as all the units expected on the webpage must be present to provide the user with a uniform and consistent experience. The unit testing approach is performed on all the webpages effectively wherein the string is searched on the webpage and if the string is encountered then the testing is considered as passed. On the other hand if the string or the unit is not found then the test is considered as fail.

IV. Results And Discussions

The proposed designed system for Selenium web testing through the web driver API is developed using Java programming language. This is done by using in the IDE NetBeans 8.2, MySQL 5.5 as Database server. Web pages of the input projects are tested using the Glass fish web Server integrated into the IDE.

The proposed module is designed in such a way that it takes both Web application and standalone application project source codes developed in Java programming as input. For the web applications three main testing is done, namely performance testing, load testing and unit testing using selenium web driver API integrated into our source code. As this part of the project uses the selenium web driver API.

The evaluation for web testing is conducted by RMSE or Root Mean Square Approach. The Root Mean Square Error or the RMSE approach can be effectively utilized as a performance metric, due to the fact that it is being used for the analysis of the error achieved between two corresponding and continuous entities. These two entities in the proposed approach are the expected project testing and the achieved project testing. The equation for the calculation of the RMSE values has been depicted in equation 1 given below.

$$\mathrm{RMSE}_{fo} = \left[\sum_{i=1}^{N} (z_{fi} - z_{oi})^2 / N\right]^{1/2}$$

Where, $\sum$ - Summation

$(Zfi - Zoi)^2$ - Differences Squared for the expected and achieved project testing

N - Number of conducted Experiments.

The RMSE values are computed for a number of iterations of Selenium WebDriver Testing performed through this proposed approach. These values of RMSE are rigorously calculated with the outcomes stipulated in the table 1 and the figure 2 given below

Test Name	No. of Projects for Testing	No. of proper Testing performed	MSE
Automation Testing	3	3	0
Load Testing	3	3	0
Unit Testing	3	2	1
		MSE	0.334

Table 1: Recorded values for MSE.

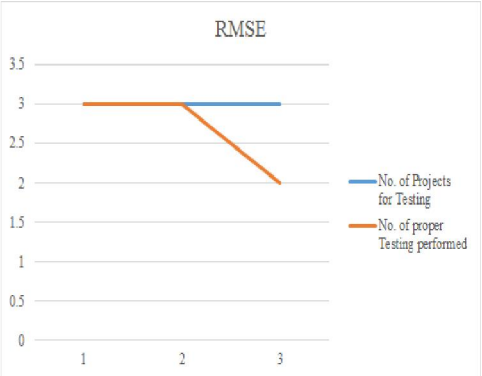

Figure 4: Line graph for MSE values

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 4, May 2022
Impact Factor: 6.252

The outcomes achieved for the performance are utilized for the purpose of plotting the line graph given in the figure 4 above. The graph and the table given above demonstrates the extremely low error rate achieved by the proposed approach for the Selenium WebDriver Testing. The improved accuracy of the testing can be attributed to the fact that the proposed approach deploys Selenium WebDriver that improves the testing accuracy exponentially. The RMSE of 0.574 is extremely satisfactory outcome for the Selenium WebDriver testing approach.

V. Conclusion And Future Scope

The testing usually is being performed through the use of manual methods and various test cases are generated and repeatedly tested to ensure the output is consistent with the expectations. The manual testing is quite tedious and highly repetitive task that can be effectively automated through the use of Selenium web driver. The selenium WebDriver has been utilized to automate and perform the Automation Testing, Load Testing and Unit Testing. This has been performed with precision and achieved the desired results. This approach also introduces the paradigm of code refactoring which suggests the removal of unused member functions and data members to clean the code effectively. This combination can be useful as a one stop solution for improving the quality of the finished software.

For the purpose of future research directions this Project can be extends by creating API for easier integration and evaluation and it Can be develop as the web service

References

[1]. M. Badri, L Badri et al, "Exploring the Impact of Clone Refactoring on TestCode Size in Object-Oriented Software", 16th IEEE International Conference on Machine Learning and Applications, 2017.

[2]. J. Zhao, M. Zhang, "Refactoring OpenMPCodeBased on MapReduce Model", IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing Communications, BigData& Cloud Computing, Social Computing & Networking, Sustainable Computing Communications, 2018.

[3]. G. Kaur and B. Singh, "Improving the Quality of Software by Refactoring", International Conference on Intelligent Computing and Control Systems, ICICCS 2017.

[4]. Bin Lin, Csaba Nagy, Gabriele Bavota, and Michele Lanza, "On the Impact of Refactoring Operationson Code Naturalness", IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2019.

[5]. Mauricio Saca, "Refactoring Improving the Design of Existing Code", IEEE 37th Central America and Panama Convention (CONCAPAN XXXVII), 2017.

[6]. J. Zhao, W. Wang, and H. Yang, "Code Refactoring Based on MapReduce in Cloud Migration", IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing Communications, 2018.

[7]. U. Devi, A. Sharma and N. Keswani, "A Review on Quality Models to Analyse the Impact of Refactored Code on Maintainability with reference to Software Product Line", International Conference on Computing for Sustainable Global Development (INDIACom), 2016.

[8]. J. Vedurada and V. Nandivada, "Refactoring Opportunities for Replacing Type Codewith State and Subclass", IEEE/ACM 39th IEEE International Conference on Software Engineering Companion, 2017.

[9]. J. Kanwal, K. Inoue and O. Maqbool, "Refactoring Patterns Study in Code Clones during Software Evolution", IEEE 11th International Workshop on Software Clones (IWSC), 2017.

[10]. GáborSz˝oke, Csaba Nagy, Lajos Jen˝oFülöp, Rudolf Ferenc, and Tibor Gyimóthy, "FaultBuster: An Automatic Code SmellRefactoring Toolset", IEEE 15th International Working Conference on Source Code Analysis and Manipulation (SCAM), 2015.5