International Journal of Computations, Information and Manufacturing (IJCIM) 3(2) -2023

Contents available at the publisher website: G A F T I M . C O M

International Journal of Computations, Information and Manufacturing (IJCIM)

Journal homepage: https://journals.gaftim.com/index.php/ijcim/index

Utilizing Machine Learning for Predicting Software Faults Through Selenium Testing Tool

Ghada Alsuwailem[1], Ohoud Alharbi[1]

[1] *King Saud University, Riyadh, Saudi Arabia*

A R T I C L E I N F O

Keywords:
Machine Learning, Software Faults, Selenium Testing Tool.

Received: Nov, 19, 2023
Accepted: Nov, 30, 2023
Published: Dec, 22, 2023

A B S T R A C T

Software quality assurance, especially in the context of the testing phase, plays a pivotal role in ensuring the reliability and functionality of software systems. Automation testing is recognized as a valuable technique to enhance test coverage and accuracy. However, challenges such as diverse automation tools and unrealistic expectations can hold up its effectiveness. This research explores the integration of machine learning into the Selenium automation testing tool to predict faults based on UI and historical scenarios. The study aims to investigate the impact of machine learning on perceived task difficulty and time required for fault prediction during software testing. The literature review emphasizes the importance of software testing, automation testing, and the Selenium tool. The research methodology employs a mixed-methods approach, combining quantitative and qualitative analyses. The results show positive perceptions regarding the clarity of implementing machine learning-based Selenium but mixed opinions on the ease of implementation. The ML-based Selenium tool demonstrates increased effectiveness, reliability, and reduced testing duration. Interviews highlight the complementary roles of manual and automated testing. The discussion addresses improved test effectiveness, reliability, challenges, and future considerations, affirming the viability and advantages of incorporating machine learning into the Selenium framework for automation testing.

1. INTRODUCTION

Commencing software quality assurance represents one of the most critical facets within the software development lifecycle. Within this lifecycle, there is a testing phase which involves the process of exercising a software system using a variety of inputs with the intention of validating its behavior and discovering faults. These faults, also known as bugs or defects, can cause failures in their software systems (Chen, et al., 1998) The importance of the testing phase would verify and validate the aim of the software. To achieve high level of software quality, we should concentrate on the techniques and methods used for testing process. Automation testing is one of the useful techniques that would help to increases test coverage and accuracy. On account of the comprehensive, pluralism features and integrated of software, it is needful to conduct appropriate techniques to guarantee reaching a high level of software quality. The software engineer is exposed to challenges to achieve a high level of software quality, including:

- The list of test automation tools in use can pose a challenge for testers, as each tool necessitates a unique learning curve. For instance, the usage methodology of Selenium differs from that of Appium. Furthermore, each tool exhibits distinct

variations and applications; for example, Selenium encompasses three types—Selenium WebDriver, Selenium IDE, and Selenium Grid—each demanding dedicated learning and expertise.
- In alignment with the 7 Testing Principles (Florea, R. and Stray, V. 2019), the concept of 'Absence-of-errors' is recognized as a fallacious notion. However, organizations setting unrealistic expectations of automation by assuming that automation would solve all issues which is not true and lead to facing heightened demands and expectations from testers.

Software invariably exhibits defects due to the inherent fallibility of human programmers. The genesis of such imperfections lies in several factors, including the potential for ambiguous or erroneous software requirements, misinterpretation of these requirements, misuse of software components, human errors during the coding process, and the susceptibility of previously functional code to discrepancies arising from changes. In light of these challenges, the practice of software testing emerges as an intuitive and indispensable approach to mitigate the impact of these imperfections (Gamido, H. V. and Gamido, M. V. 2019). One of the practices technique of software testing is Software fault prediction (SFP) that encompasses the endeavor of constructing predictive models intended for deployment by software professionals during the nascent stages of the software development life cycle, with the primary aim of identifying defective elements, such as modules or classes. Historically, a spectrum of machine learning methodologies has been harnessed for the task of fault prediction (Honest, N. 2019). Machine learning can be harnessed to streamline the automation of test scripts, consequently enhancing the level of software quality and concurrently alleviating the testing team's workload.

1.1. Objective

This research paper aims to find the appropriate solution to avoid challenges and achieve a high level of software quality. This is pursued through the implementation of a machine learning-based Selenium automation testing tool, which aids in script generation based on the UI and historical scenarios to predict faults**.** To accomplish the objective the detailed goals should be on consideration:

- Conducting a thorough investigation to familiarize with the machine learning-based Selenium automation testing tool, gaining insight into both their advantages and shortcomings.
- A profound understanding of machine learning utilization techniques within the domain of software quality assurance.
- Developing a system to implement the machine learning-based Selenium technique in the testing phase. The system would be designed to bring together a community of testers, offering a variety of features, with the primary focus being the collaborative exchange of software testing expertise.

1.2. Research Questions

To verify the effectiveness of applying machine learning – based Selenium technique, we design the following research question.

RQ1: How does the utilizing of machine learning technology in a Selenium automation testing tool impact the perceived task difficulty and time required for fault prediction during software testing?

The following is the hypotheses associated with research question:

H1a: Faults prediction effort as measured by the perceived task difficulty and time to complete the task will be significantly lower for software testing tools supported with machine learning technology. In this paper, we will employ a machine learning-based Selenium automation testing tool.

2. LITERATURE REVIEW

This section provides an overview of the literature review conducted in the study, encompassing the significance of software testing in achieving software quality. It delves into a brief exploration of automation testing, focusing on the application of one prominent tool, Selenium Automation Testing Tool. Additionally, the section provides an insight into the broader context of machine learning and its application in predicting software faults.

2.1. Importance of Software Testing to Achieve Software Quality

A lot of researches have been conducted to emphasize the importance of software quality with the domain of the effectiveness of automation testing utilizing the machine learning capability in software faults prediction. A group of researchers has explained the concept of software testing. One of their explanations is that, software testing is among the umbrella activities performed at any organization to provide value and quality, ensuring the longevity of software products in the market (Kaur, M. and Kumari, R. 2011). Besides that, other researches assert the importance of the testing phase that consuming an average of 40% to 70% of software development process. Furthermore, software testing plays a crucial role in evaluating and ensuring the quality of software. It is essential in confirming that software functions as intended and does not perform unintended actions (Kufel, J. et al. 2023). Correspondingly, the optimal level of testing efficiency is characterized by its capacity to achieve the desired software quality standard while demanding a reduced level of effort (Li, Z. et al., 2018). Notably, testing accounts for a substantial portion of software development costs and emphasized the importance of software testing in ensuring software quality on account of modern software systems have grown increasingly complex, making conventional testing techniques less scalable. This complexity has driven the adoption of machine learning-based techniques in testing.

2.2. Automation Testing

On the spot of automation testing, automated testing addresses the difficulties arising from manual testing and the testers put more focus on automated tests than on manual tests (Wardhan, H. and Madan, S). The researchers mention the strength points of using the automation testing which are: Automation testing executes test cases significantly faster than manual testing, less testers are required in automation testing as a result, less investment is required in human resources, Automation testing programmable that program sophisticated tests to bring out hidden information, and the last point is that Automation testing is more reliable and less error prone than manual testing (Seralina, N. 2021). The advertisers put more focus on automated tests than on manual tests.

2.3. Selenium Automation Testing Tool

According to (Li, Z. et al., 2018) selenium is defined as one of the preeminent automation frameworks encompassing a multitude of tools and extensions designed for the purpose of conducting testing on web applications also, it is recognized for its powerful capability in performance testing and maintains a prominent presence in the realm of open-source test automation (Jaganeshwari, K.). Discussed various automation tools, including those utilizing artificial intelligence that work as a means to address challenges in testing, one of the mentioned automation tools is selenium. There are common advantages and drawbacks of selenium, the advantages are: Open source, no licensing and maintenance fees, Open for integration with other tools and frameworks, Ease of use, Flexibility, Capability to debug and set breakpoints in test cases and Allows tests written in different programming languages for advantages. On the other hand, the drawbacks are: Writing test cases with Selenium requires a certain level of programming skill and no built-in error handling capabilities, which can make it challenging to handle and report errors effectively (Marijan, D. and Gotlieb, A. 2020) . Selenium simplifies the work of automation testers, leading to improved testing efficiency and cost-effectiveness. Selenium's open-source nature, flexibility in scripting languages, compatibility across operating systems and browsers, and seamless integration with other tools contribute to its popularity. Testers can write scripts in various languages and perform testing on Windows, MacOS, and Linux, across different browsers, ensuring cross-browser compatibility (Mobaraya, F. and Ali, S. 2019). Selenium is a versatile automation testing framework, comprises three key components. Firstly, Selenium IDE facilitates the recording, editing, debugging, and replaying of functional tests. Testers can effortlessly record browser interactions and export tests in multiple programming languages for enhanced flexibility. Secondly, Selenium Grid empowers parallel automated testing across multiple machines and browsers, optimizing time and overall performance. This component allows testers to conduct tests simultaneously in various browsers and operating systems. Lastly, Selenium WebDriver stands out as the cornerstone of the Selenium Suite, providing a programming interface for the creation and execution of automation

scripts. Testers can choose their preferred programming language to identify web elements on pages and perform actions, thus tailoring the automation process to their specific needs. Selenium WebDriver's compatibility with popular browsers further contributes to an efficient and comprehensive testing experience (Nyamathulla, S. et al. 2021).

Machine learning – based selenium tool consists of eight main steps to implement which are: Collect Training Data, Feature Extraction, Labeling, Train Machine Learning Model, Monitoring and Inference, Dynamic Test Case Generation, Execute Test Cases, Feedback Loop, Iterate and Improve.

2.4. Application of Software Faults prediction utilizing Machine Learning

Machine learning is a subset of AI that involves building computer models that are capable of learning and making independent predictions or decisions, it is a field at the intersection of AI, computer science, and statistics, is used to automate and streamline software testing thus, some software testing problems can be framed as learning problems, making machine learning a suitable approach (Nyamathulla, S. et al. 2021). In the past, extensive research has focused on the application of machine learning in software testing. A Systematic Literature Review (SLR) that covers 154 studies from 1990 to June 2019, providing guidelines for software practitioners and researchers (Wardhan, H. and Madan, S. 2023). Furthermore, various studies have demonstrated the significance of software fault prediction (SFP) by conducting comprehensive reviews. One study systematically reviewed 74 research articles from 11 different journals up until 2007. This review categorized machine learning-based approaches and commonly used software metrics in the SFP field. Another study conducted a systematic literature review covering the period from 1991 to 2013. This review analyzed a range of machine learning methods, software metrics, datasets, and performance measures in the context of fault prediction (Pandey, S.K. et al. 2021; Spicer, J. and Sanborn, A.N. 2019). Moreover, a study indicated that the AI support for the quality test extracted in technical batches, which used to solve multiple problems when testing focuses on information systems for machine intelligence applied to automated software testing, with an exploration of artificial intelligence and its importance in the software development and testing process, additionally, it highlights AI's ability to generate quick and efficient tests, ultimately saving time and resources (Mobaraya, F. and Ali, S. 2019). Various machine learning techniques are categorized into eight groups: Bayesian learners, Decision Trees, Evolutionary Algorithms, Ensemble Learners, Neural Networks, Support Vector Machines, Rule-Based Learning, and Miscellaneous. Notable methods such as Bayesian learners, Decision Trees, and Miscellaneous are frequently used in SFP and some articles use a combination of these methods for optimal results (Wardhan, H. and Madan, S. 2023). In contrast, there are researches categorized machine learning algorithms into two main learning categories: supervised and unsupervised. Supervised learning involves mapping input variables to corresponding output variables for prediction or understanding. Unsupervised learning deals with input data only and focuses on clustering problems (Wardhan, H. and Madan, S, .2023). Similarly, some studies fall into a semi-supervised category when only a subset of input data has associated output data (Malhotra, R. 2015). As will, the effectiveness of various machine learning techniques, such as SVM and RF, can vary depending on the dataset and problem (Sugali, K. et al. 2021). Furthermore, a novel methodology introduced to enhance the process of GUI testing through the utilization of machine learning techniques for the recognition and categorization of GUI widgets. Machine learning can be applied to predict the effectiveness of test cases by learning from data about what constitutes an effective test case, also, machine learning algorithms can identify patterns and structures in the data to create models for making predictions about test case effectiveness. Controlled AI systems are credited with achieving high test case coverage while effectively addressing scalability and error-related challenges (Noorian, M. et al. 2018). Regarding the machine learning models, it is achieved an average AUC between 0.72 to 0.84 and an accuracy range of 75.7% to 85.01% in SFP, besides that, machine learning faces challenges that related to dealing with imbalanced datasets, overfitting, data quality, and variations in fault information across different projects. As a result of the study, it recommended the availability of benchmark datasets from

various industries as crucial for improving SFP models and encouraged industries to provide freely available datasets with more features and test cases to support deep learning (DL) applications without overfitting (Phuc Nguyen, D. and Maag, S. 2020). One of the observations that obtained that the absence of comprehensive guidelines for selecting suitable machine learning methods for software testing (Sugali, K. et al. 2021). Correspondingly, other study suggested more rigorous empirical evaluations to support the proposed solutions with reason of an increasing interest in the application of machine learning algorithms to automate software testing (Okezie, F. et al. 2019).

In conclusion, our observations have identified several knowledge gaps that motivate further exploration into the application of machine learning in software testing, particularly when coupled with automation testing tools. These gaps include the scarcity of empirical studies focused on machine learning-based automation testing and a lack of research addressing the comparison of appropriate machine learning techniques tailored to the field of software testing with manual testing.

3. RESEARCH METHODOLOGY

This section outlines the research methodology employed in the study, detailing the procedures and techniques used to collect and analyze data. The research is based on utilizing machine learning – based selenium tool on Tester Community System (TCS) to measure number of closed test cases, time duration of the test, tester satisfaction, and overall software quality of (TCS) using the tool. The research follows a mixed-methods approach to gain a comprehensive understanding of the research problem. This approach combines quantitative and qualitative analyses, allowing us, on one hand, to obtain reliable results regarding test cases and the number of closed test cases in a shorter time frame while maintaining a high level of quality (Spicer, J. and Sanborn, A.N. 2019).

3.1. Research Design

This research is an experimental study aimed at investigating the impact of software quality in the Tester Community System (TCS) by utilizing a machine learning-based selenium tool. The experimental design will employ a within-subject approach with two conditions. This design will enable a group of four experienced testers to assess the effectiveness of using the machine learning-based selenium tool on TCS, which involves a system of moderate complexity. Consequently, we will conduct two sets of experiments: one utilizing the machine learning-based selenium tool and the other employing manual testing methods.

3.2. Data Collection Methods

We collected data through interviews with testers who have experience in both automation and manual testing. To select participants, we considered their expertise and involvement in the software quality domain. Additionally, we gathered post-experimental data using a survey containing five key questions that focused on aspects such as clarity, easily, effectiveness, and reliability of the tool. These interviews were audio-recorded and later transcribed for analysis (see Appendix A for usability testing script and Appendix B for survey questions).

3.3. Sampling Strategy

The target population for this research comprised four experienced testers, each with a minimum of five years of experience. We selected testers as participants because the tool is specifically designed for testing purposes. These testers possess extensive experience in software quality control, encompassing both manual testing and automation testing, and they are familiar with the use of the Selenium tool. The decision to include four testers in the study was driven by the tool's learning curve, which requires a significant amount of time for proficiency. For the ethical considerations, we obtained informed consent from all selected participants, explaining the study's purpose, voluntary participation, and response confidentiality (see Appendix C for consent form).

3.4. Data Analysis Techniques

We analyzed the transcribed interviews to understand the testers' opinions on the quality improvement facilitated by the machine learning-based selenium tool. Additionally, we used a Paired Samples t-Test within a within-subject design to assess the significance of differences between observations when using the machine learning-based selenium tool compared to manual testing on our system.

In summary, Selenium's open-source nature, language flexibility, cross-browser compatibility, and parallel testing capabilities have made it an essential tool for software testers and developers. It has significantly contributed to streamlining the testing process, improving test coverage, and ensuring the reliability of web applications.

4. DATA ANALYSIS

In this section, we present the results of the research focused on the utilization of a Machine Learning (ML)-based Selenium tool for automation testing. The study aimed to assess the clarity, easily, shorter time, effectiveness and reliability of incorporating ML algorithms into the traditional Selenium framework. We requested from testers to start doing the excremental by preparing the necessary prerequisite for implementing both types of testing which are automation testing using Machine Learning (ML) – based Selenium and manual testing.

4.1. Implementing automation and manual testing

A group of four experienced testers had ask to implement automation testing by utilizing Machine Learning (ML)-based Selenium by integrating machine learning techniques with the Selenium automation testing framework. This integration aims to enhance Selenium's capabilities by leveraging ML algorithms to optimize test automation processes. After establishing all the necessary processes for utilizing ML-based Selenium, a team of four testers initiated the implementation phase. We asked to test one test case by crafting a Selenium script to test the login use case, as illustrated in Figure 1.

```
from selenium import webdriver
import time
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
#chrome driver
from selenium.webdriver.chrome.service import Service
#-- Chrome
service_obj = Service("/Users/galsuwailem/Desktop/chromedriver.exe")
driver = webdriver.Chrome(service=service_obj)

driver.get("                                                    ")
driver.find_element(By.XPATH,"//a[contains(text(),'Free Access to InterviewQues/ResumeAssistance/Mate')]").click()
windowsOpened = driver.window_handles
driver.switch_to.window(windowsOpened[1])
email = driver.find_element(By.XPATH, "//a[contains(@href,'mentor@rahulshettyacademy.com')]").text
print(email)
driver.close()
driver.switch_to.window(windowsOpened[0])
driver.find_element(By.XPATH, "//input[@id='username']").send_keys(email)
driver.find_element(By.XPATH, "//input[@id='password']").send_keys(1234)
driver.find_element(By.XPATH, "//input[@id='signInBtn']").click()
wait = WebDriverWait(driver,10)
wait.until(expected_conditions.visibility_of_element_located((By.CSS_SELECTOR, ".alert-danger")))
print(driver.find_element(By.CSS_SELECTOR, ".alert-danger").text)

Run: assignment-Ghada
C:\Users\galsuwailem\Python39\python.exe C:\Users\galsuwailem\Desktop\pythonTesting\pythonSelenium\assignment-Ghada.py
mentor@rahulshettyacademy.com
Incorrect username/password.

Process finished with exit code 0
```

Figure 1 Selenium Script

Regarding the prerequisites of implementing manual testing are to prepper the exapted result and test cases of each use cases and do the test manually. As illustrated in Figure 2, our team of four experienced testers had test login use case by entering different kind of data to find the defects manually.

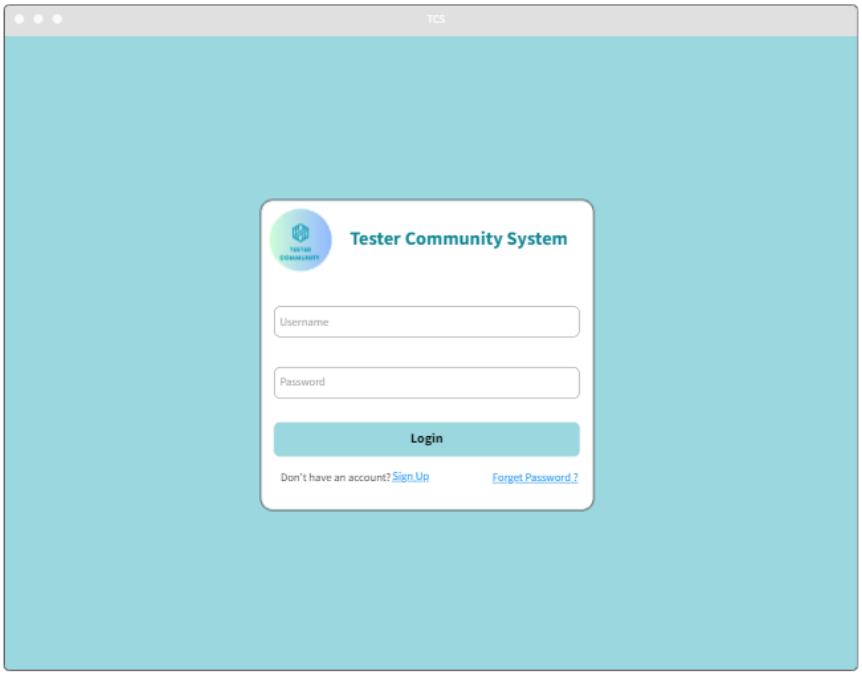

Figure 2 Selenium Script

About the assessment of the clarity of implementing ML-based Selenium integration steps, the responses overwhelmingly indicate a positive consensus among participants. As illustrated in Figure 2, a substantial 75% of respondents expressed agreement, with an additional 25% strongly agreeing that the steps for integration are clear. Notably, there were no dissenting voices in the form of disagreement, suggesting a unanimous view on the clarity of the implementation process.

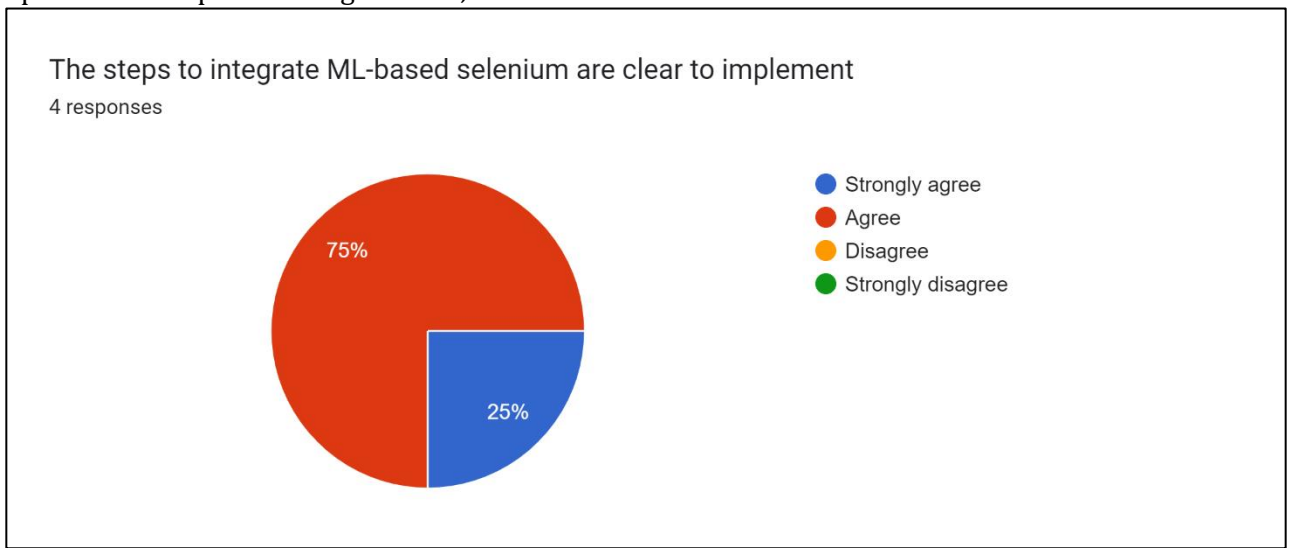

Figure 3 Clarity of Implementation

The responses regarding the perceived ease of implementing ML-based Selenium integration steps as illustrated in Figure 3, reflect a more diverse range of opinions among participants. While 25% strongly agree and an additional 25% agree that the steps are easy to implement, a notable 50% expressed disagreement. Importantly, none of the participants strongly disagreed with the statement. This distribution of responses suggests a mixed perspective on the ease of integrating ML-based Selenium, with a significant portion finding the process challenging.

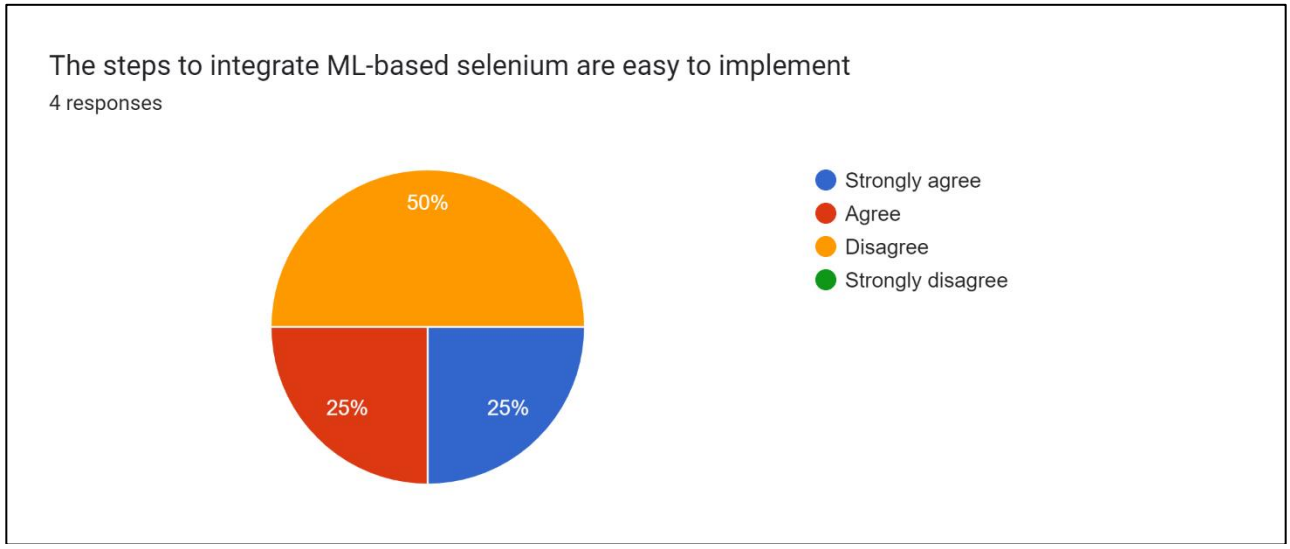

Figure 4 Ease of Implementation

4.2. Result after implementing Machine Learning (ML)- based Selenium

After the implementation of automation testing type by utilizing Machine Learning (ML)- based Selenium and testing one test case (login test case), Our findings reveal a notable point in the accuracy of automated tests when leveraging the ML-based Selenium tool, where the ML-based Selenium tool exhibits increased reliable and effective test results as illustrated in Figure 4,5.

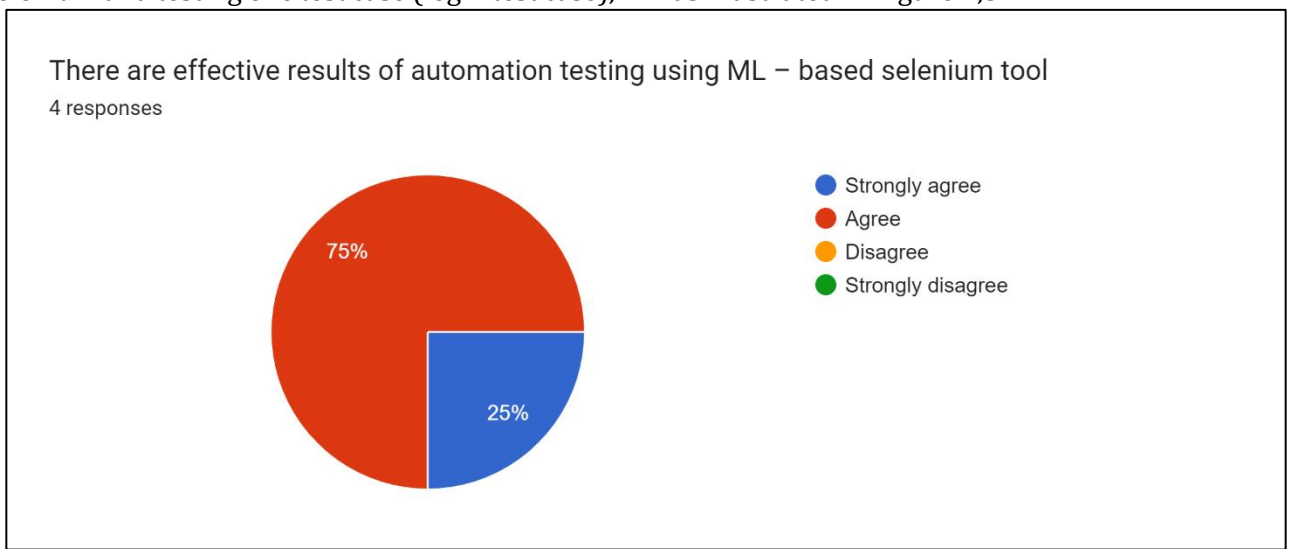

Figure 5 Effectiveness Results

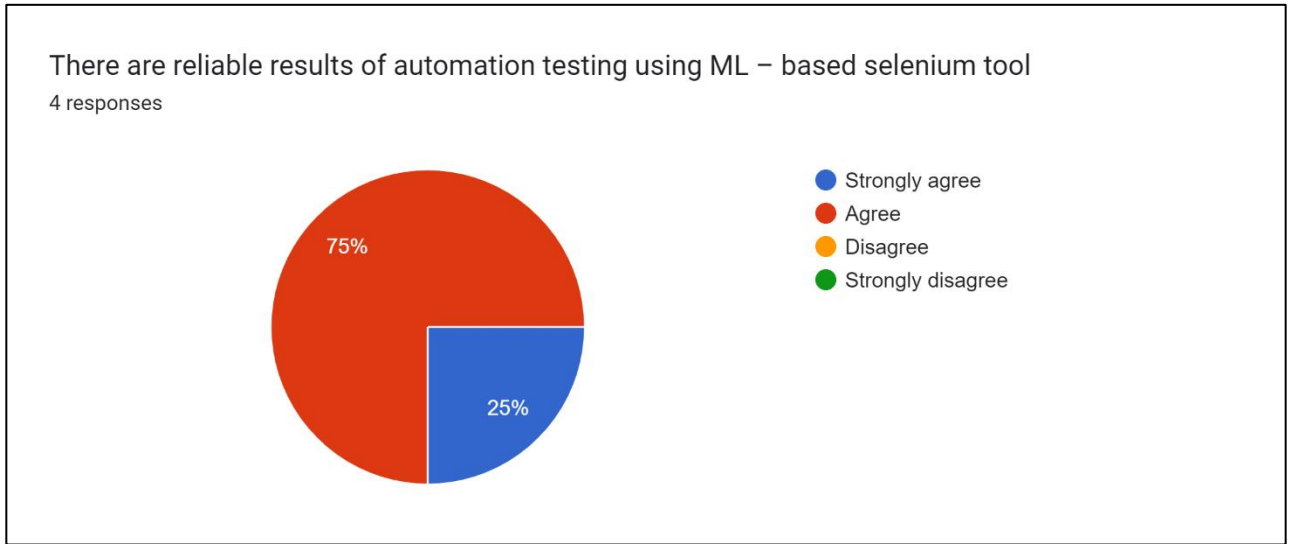

Figure 6 Reliable Result

While in Figure 6 a comparative analysis of test execution times indicates a notable decrease in the overall testing duration when utilizing an ML-based selenium tool.

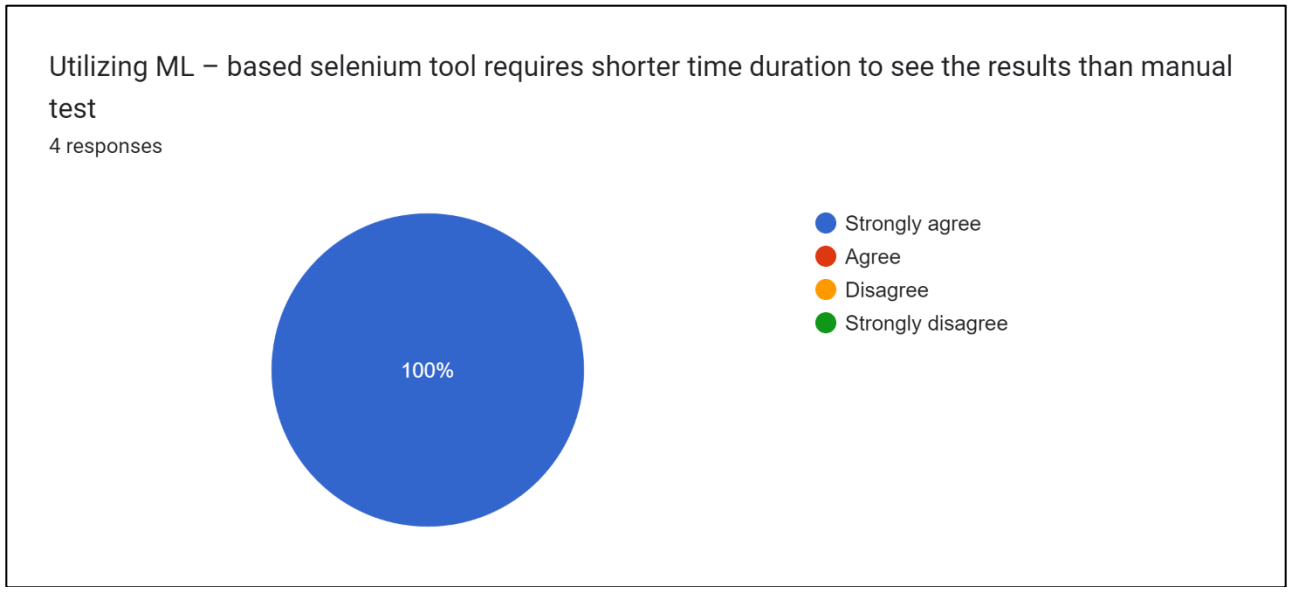

Figure 7 Time Duration Result

4.3. Interviews

According to (Business Development | Advisory Board Member) said that in mission-critical industries, the necessity of end-to-end testing is emphasized, with the understanding that while nothing is replaceable, the level of testing can be optimized as the automated test suite matures. Examples such as capacity testing, boundary testing, and robustness testing are identified as candidates for automation, yet certain aspects like UX/UI testing may still require manual intervention. Furthermore (QA & Test Automation Specialist) said that this perspective aligns with the notion that automation testing serves to enhance the manual testing process by automating specific regression test cases, emphasizing that automation is not a complete replacement for manual testing. However, (QA Automation Engineer) said that a nuanced stance is presented, suggesting that while automation can potentially cover every test, it is not a mandate to do so. The importance of manual testing in checking new features during a sprint is highlighted, asserting that automation should not be viewed as a complete substitute. The integration of both manual and automated testing is advocated, with an emphasis on a tester's understanding of when to apply each approach.

5. DISCISSION

The positive consensus of the results on the effectiveness and reliability of ML-based Selenium echoes the literature's (Malhotra, R. 2015) emphasis on the critical role of software testing in ensuring quality. The discussion on the benefits of automation testing, such as faster execution, reduced human resource requirements, and increased reliability, finds validation in the study's results. The results demonstrate a noteworthy improvement in the effectiveness and reliability of automated tests when leveraging the ML-based Selenium tool. This finding underscores the potential of machine learning to optimize fault prediction based on UI and historical scenarios, thereby enhancing the overall software quality assurance process. The significant decrease in overall testing duration with the ML-based Selenium tool is a compelling outcome. This reduction in time aligns with the goals of efficiency and streamlining the testing process. The study's positive perceptions regarding the clarity of implementing ML-based Selenium reinforce the Selenium tool's strengths outlined in the literature (Marijan, D. and Gotlieb, A. 2020). The flexibility, ease of use, and compatibility across browsers and operating systems contribute to the positive reception of the ML-based Selenium integration steps. The literature review (Pandey, S.K. et al. 2021) categorizes machine learning techniques into various groups, including Bayesian learners, Decision Trees, and Ensemble Learners. The study's implementation of ML-based Selenium doesn't delve into specific techniques but aligns with the broader idea of applying machine learning to enhance the automation testing process.

6. CONCLUSION

In conclusion, this research delves into the intersection of machine learning and automation testing, particularly within the Selenium framework, to enhance the quality assurance process in software development. The integration of machine learning introduces a promising avenue for addressing challenges in fault prediction during software testing. The study focused on understanding the impact of machine learning on task difficulty, time requirements, and overall software quality within the Tester Community System (TCS). The literature review underscores the critical role of software testing in ensuring quality, with a specific emphasis on the effectiveness of automation testing, particularly using the Selenium tool. Automation testing, though powerful, has its challenges, and this research proposes the incorporation of machine learning as a solution to improve fault prediction. The research methodology employs a well-structured mixed-methods approach, combining quantitative analyses for measuring the impact on test cases and qualitative insights gathered through interviews. The results showcase positive perceptions regarding the clarity of implementing the machine learning-based Selenium tool, with a notable consensus on its effectiveness, reliability, and reduction in testing duration. Despite the positive outcomes, challenges such as the learning curve and model training overhead are acknowledged. The findings also highlight the importance of maintaining a balance between manual and automated testing, recognizing the unique strengths of each approach. The study concludes by affirming the viability and advantages of incorporating machine learning into the Selenium framework for automation testing. It demonstrates the potential for improved accuracy, reduced testing time, and increased efficiency in the software testing process. The identified challenges pave the way for future research avenues, encouraging further refinement and exploration of the capabilities of machine learning-based Selenium tools in automated testing. Ultimately, this research contributes to the ongoing discourse on advancing software quality assurance methodologies, emphasizing the synergy between machine learning and automation testing for more robust and reliable software systems.

REFERENCES

Chen, T.Y. et al. 1998. Metamorphic Testing: A New Approach for Generating Next Test Cases.

Florea, R. and Stray, V. 2019. The skills that employers look for in software testers. Software Quality Journal. 27, 4 (Dec. 2019), 1449–1479. DOI:https://doi.org/10.1007/s11219-019-09462-5.

Florea, R. and Stray, V. 2019. The skills that employers look for in software testers. Software Quality Journal. 27, 4 (Dec. 2019), 1449–1479. DOI:https://doi.org/10.1007/s11219-019-09462-5.

Gamido, H. V. and Gamido, M. V. 2019. Comparative review of the features of automated software testing tools. International Journal of Electrical and Computer

Engineering. 9, 5 (Oct. 2019), 4473–4478. DOI:https://doi.org/10.11591/ijece.v9i5.pp4473-4478.

Honest, N. 2019. Role of Testing in Software Development Life Cycle. International Journal of Computer Sciences and Engineering. 7, 5 (May 2019), 886–889. DOI:https://doi.org/10.26438/ijcse/v7i5.886889.

Jaganeshwari, K. and Djodilatchoumy, D.S. 2022. AN AUTOMATED TESTING TOOL BASED ON GRAPHICAL USER INTERFACE WITH EXPLORATORY BEHAVIOURAL ANALYSIS. Journal of Theoretical and Applied Information Technology. 100, (2022), 22.

Kaur, M. and Kumari, R. 2011. Comparative Study of Automated Testing Tools: TestComplete and QuickTest Pro.

Kufel, J. et al. 2023. What Is Machine Learning, Artificial Neural Networks and Deep Learning?—Examples of Practical Applications in Medicine. Diagnostics. Multidisciplinary Digital Publishing Institute (MDPI).

Li, Z. et al. 2018. Progress on approaches to software defect prediction. IET Software. Institution of Engineering and Technology.

Malhotra, R. 2015. A systematic review of machine learning techniques for software fault prediction. Applied Soft Computing. (2015).

Marijan, D. and Gotlieb, A. 2020. Software Testing for Machine Learning. Proceedings of the AAAI Conference on Artificial Intelligence. 34, 09 (Apr. 2020), 13576–13582. DOI:https://doi.org/10.1609/aaai.v34i09.7084.

Mobaraya, F. and Ali, S. 2019. Technical Analysis of Selenium and Cypress as Functional Automation Framework for Modern Web Application Testing. (Dec. 2019), 27–46.

Noorian, M. et al. 2018. Machine Learning-based Software Testing: Towards a Classification Framework.

Nyamathulla, S. et al. 2021. A Review on Selenium Web Driver with Python.

Okezie, F. et al. 2019. A Critical Analysis of Software Testing Tools. Journal of Physics: Conference Series (Dec. 2019).

Pandey, S.K. et al. 2021. Machine learning based methods for software fault prediction: A survey. Expert Systems with Applications. Elsevier Ltd.

Phuc Nguyen, D. and Maag, S. 2020. Codeless web testing using Selenium and machine learning. (2020), 10. DOI:https://doi.org/10.5220/0009885400510060ï.

R. M. Sharma 2014. Quantitative Analysis of Automation and Manual Testing. (2014).

Rafael Lenz, A. et al. 2013. Linking software testing results with a machine learning approach. Engineering Applications of Artificial Intelligence. 26, 5–6 (May 2013), 1631–1640. DOI:https://doi.org/10.1016/j.engappai.2013.01.008.

Seralina, N. 2021. INFORMATION SYSTEMS FOR MACHINE INTELLIGENCE TO AUTOMATED SOFTWARE TESTING. Herald of Kazakh-British technical university. 18, 1 (Mar. 2021), 157–161. DOI:https://doi.org/10.55452/1998-6688-2021-18-1-157-161.

Spicer, J. and Sanborn, A.N. 2019. What does the mind learn? A comparison of human and machine learning representations.

Sugali, K. et al. 2021. Software Testing: Issues and Challenges of Artificial Intelligence & Machine Learning. International Journal of Artificial Intelligence & Applications. 12, 1 (Jan. 2021), 101–112. DOI:https://doi.org/10.5121/ijaia.2021.12107.

Wardhan, H. and Madan, S. 2376. STUDY ON FUNCTIONING OF SELENIUM TESTING TOOL. 2023. Certified Tester Foundation Level Syllabus v4.0 International Software Testing Qualifications Board.

Appendix A: Usability Testing Script

Introduction

Hello Participant, how are you? I really appreciate you taking time out of your day to participate in this test. I am Ghada and I am a researcher at KSU.

So, let me outline how this will go. I would like to start by asking you some questions about who you are, your background, and your relevant experience. I will then ask you to perform some tasks on our Tester Community System (TCS) utilizing ML-based selenium tool. Once the tasks have been completed, I would like to get some feedback from you about your experience with the tool.

We're doing this usability test to see how testers interact with ML-based selenium tool, and to hear their thoughts on it. We're trying to make this the best it can be, so your honest thoughts are really important to us.

I saw that you signed our consent form. Do you have any questions about this?

Is there anything you'd like to ask before we get going?

Finally, I'd like to make sure you're comfortable with us recording today's session. Is this okay with you?

Fantastic, so I'll now start recording the audio and dive into some background questions.

Background questions

So, Participant, could you tell us what your current job title is, and a brief overview of what your job entails?

How long of work experience as software tester do you have?

Could you describe your experience with automation testing tool?

Tasks

Thank you for your answers. We're now ready to start the test. Before we begin, I'd like to remind you of a few things.

First off, remember that we aren't testing you today, we're testing our system and tool. So, if something isn't working, don't worry, it's a problem with our software and not something you've done wrong. In fact, there are no wrong answers here.

When using the system and the tool, try to act as naturally as possible. I get that it's hard to do that with us watching your screen. But, please try to act as if you were using the system and the tool on your own, without anyone watching.

Ensure that you install selenium tool with type webdriver, then integrated using the URL of (TCS) to begin the test. Please start write the test cases script, then you can see that the machine learning-based selenium will predicate the potential faults.

Please think aloud as you're using our system and tool. We really want to hear your thoughts, like where you're navigating on the page, why you're clicking somewhere, what you expect to happen when you do click, that sort of thing. If you have questions, feel free to ask me, and I'll answer all of them I can.

Finally, we'd like you to be as honest as possible. If something doesn't make sense on the page, or it's not working right, then feel free to tell us. You're not going to hurt our feelings, so don't worry about that.

Great, so let's begin. I'll now start recording your screen.

Observations and open-ended questions

While you interact with the system and tool, I'll be observing your actions. I might ask questions if I notice something interesting or unusual.

Wrap-up

And that's the final task finished! I've stopped recording your screen.

Before we finish, I'd like to ask you a few quick questions.

Firstly, what did you think of the tool?

Thank you for that. Is there anything you'd like to add before we finish up?

Fantastic. Well, thank you again so much for taking the time out of your day to take part in this study with us. Your input today will be extremely useful for us. Take care, I hope to speak to you soon. Goodbye!

Appendix B: Survey Questions

Q1. The steps to integrate ML-based selenium are clear to implement
- Strongly agree
- Agree
- Disagree
- Strongly disagree

Q2. The steps to integrate ML-based selenium are easy to implement
- Strongly agree
- Agree
- Disagree
- Strongly disagree

Q3. Utilizing ML – based selenium tool requires shorter time duration to see the results than manual test
- Strongly agree
- Agree
- Disagree
- Strongly disagree

Q4. There are effective results of automation testing using ML – based selenium tool
- Strongly agree
- Agree
- Disagree
- Strongly disagree

Q5. There are reliable results of automation testing using ML – based selenium tool
- Strongly agree
- Agree
- Disagree
- Strongly disagree

Appendix C: Consent Form

1. Informed Consent Form (Qualitative Study)

- I voluntarily agree to participate in this research study.

- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any question without any consequences of any kind.

- I understand that I can withdraw permission to use data from my interview within two weeks after the interview, in which case the material will be deleted.

- I have had the purpose and nature of the study explained to me in writing and I have had the opportunity to ask questions about the study.

- I understand that participation involves testing (TCS) by using ML- based selenium testing tool.

- I understand that I will not benefit directly from participating in this research.

- I agree to my interview being audio-recorded.

- I understand that all information I provide for this study will be treated confidentially.

- I understand that in any report on the results of this research my identity will remain anonymous. This will be done by changing my name and disguising any details of my interview which may reveal my identity or the identity of people I speak about.

- I understand that disguised extracts from my interview may be quoted in the research result.

- I understand that signed consent forms and original audio recordings will be retained.

- I understand that a transcript of my interview in which all identifying information has been removed will be retained.

- I understand that under freedom of information legalisation I am entitled to access the information I have provided at any time while it is in storage as specified above.

- I understand that I am free to contact any of the people involved in the research to seek further clarification and information.

Ghada Alsuwailem, master's degree student, KSU, 445920822@student.ksu.edu.sa (supervisor: Dr. Ohoud Alharbi).

Signature of research participant

Signature of participant Date

Signature of researcher
I believe the participant is giving informed consent to participate in this study

Signature of researcher Date

2. Informed Consent Form (Quantitative study)
Project title
Utilizing Machine Learning for Predicting Software Faults Through the Selenium Testing Tools.

Study researcher
Researcher: Ghada Alsuwailem, Software Engineer King Saud University (KSU).
Phone: +966-599-100012; email: 445920822@stuednt.ksu.edu.sa

Invitation to participate
You are being invited to participate in a research study on utilizing machine learning for predicting software faults through the selenium testing tools. Choosing whether or not to participate is entirely your choice. If you decide not to participate, there will be no negative impacts on your relationship with the researcher. The information provided in this form tells you about what is involved in the research, what you will be asked to do, and any potential risks or benefits. Please read this form carefully, take all the time you need, and ask any questions you may have.

Consent is an ongoing process. During the research study, we will tell you about any significant finding that could affect your willingness to continue to participate in this study.

Purpose of the research study
On account of the comprehensive, pluralism features and integrated of software, it is needful to conduct appropriate techniques to guarantee reaching a high level of software quality. This study will assess the incorporation of machine learning-based Selenium automation testing tool technology during the software testing phase augment the software's quality.

What you will be asked to do
If you decide to take part in this research, you will be asked to use selenium tool to test one test case of the system measuring the effort and time. This trail will take approximately one (1) hour in total.
Who can take part in the research study?
Participants must have at least 5 years work experience in software quality control.

Possible risks and benefits
Risks: Some sections of the questionnaire will focus on personal topics that may make you uncomfortable. You are free to refuse to answer any questions. is it good. Unfamiliarity with the tool can lead to prolonged testing procedures, which will take more time, however, the risks associated with this procedure are low.

Benefits: There is no guarantee that you will benefit directly from participating in this study. However, your participation will advance knowledge about the utilizing machine learning through the selenium testing tools.

Privacy and confidentiality
The personal information collected in this research project is handled in accordance with the Privacy Act.

Data retention
We would like to keep your data indefinitely for future research purposes. These data will be used to measure things like duration of testing, number of closed test cases and clarity of the testing. If you do not want us to keep your data, please indicate so on the signature page below.

Reporting of results
We will only report group results, therefore, you will not be identified in any way in our reports. If

you wish to be informed of the results of the research, please indicate this on the signature page below.

Withdrawing from the study

Your participation is completely voluntary. You are under no obligation to participate and are free to withdraw at any time without consequence. Your decision to withdraw will not influence your relationship with the researcher in any way. If we have begun reporting results, we will not be able to remove your data.

Questions and contact information

If you have any questions about the study or your rights as a research participant, please contact:

Ghada Alsuwailem
+966-599-100012
445920822@stuednt.ksu.edu.sa

Signature Page

Project title: Utilizing Machine Learning for Predicting Software Faults Through the Selenium Testing Tools.

Researcher: Ghada Alsuwailem

Statement of consent

By signing this form, I agree that:

- The study has been explained to me
- All my questions have been answered
- Possible harm and discomforts and possible benefits (if any) of this study have been explained to me
- I have been told that my personal information will be kept confidential

In addition, I understand that:

- I have the right not to participate and the right to stop at any time
- I may refuse to participate without consequence
- I have a choice of not answering specific questions
- I am free now, and in the future, to ask any questions about the study
- No information that would identify me will be released or printed without asking me first
- I will receive a signed copy of this consent form

You can still participate in the research if you select no:

I consent to being contacted in the future for participation in research studies ☐Yes ☐No

Name Signature Date

Please provide an email address below if you would like to be sent a summary of the study results.

Email address:

Signature of the person obtaining consent

By signing this form, I attest that:

- I have explained the study to the prospective participant
- I answered all of their questions
- I provided a copy of this consent form to the participant
- The participant seemed to understand the consent form and agreed to participate

Name Signature Date