IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 6, June 2024

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.423

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

Automated Testing Framework for Capital Project Management Software

Harsh Bhan Giri, Dr. J.N. Hemalatha

B.E, Department of EEE, RV College of Engineering, Bengaluru, India

Associate Professor, Department of EEE, RV College of Engineering, Bengaluru, India

ABSTRACT: Modern software development approaches have come to rely heavily on automated testing, which provides many advantages over manual testing in terms of efficiency, dependability, and scalability. This paper provides a thorough analysis of the developments in automated testing methods, emphasizing the most important approaches, instruments, and industry best practices. The paper examines several automated testing methodologies, including functional testing, regression testing, and performance testing, after providing an outline of the basics of automated testing and outlining its advantages and disadvantages. The paper also explores popular automation technologies, including Selenium, and talks about their features, capabilities, and suitability for various testing scenarios. Furthermore, the paper examines emerging trends and future directions in automated testing, including the integration of artificial intelligence and machine learning techniques. Through this review, the paper aims to provide researchers and practitioners with valuable insights into the evolving landscape of automated testing, enabling them to make informed decisions and advancements in software quality assurance.

KEYWORDS: Automated Testing, Software Quality Assurance, Test Automation Tools, Selenium, Functional Testing, Regression Testing, Performance Testing.

I. INTRODUCTION

The use of automated testing has revolutionized quality assurance and is now a fundamental component of contemporary software development. Software tools use this technique to test software applications before they are released, comparing the results to what is predicted and what is really achieved. By using this approach, you can make sure that the program always fulfills requirements and works as intended, even after multiple updates and iterations.

Several tools and methods are used in automated testing to mimic user activities with the software, including typing text, pressing buttons, and navigating between displays. Selenium is a popular web automation tool that lets testers check expected results and automate browser actions. In order to interact with the website or application being tested, test scripts are usually written in programming languages like Java, which Selenium supports. Systematic and repeatable testing is made possible by these scripts, which specify a set of tasks to be completed and the anticipated outcomes.

Web browser automation replicates the actions a human tester would take during manual testing by inserting JavaScript code into the browser driver. After that, the automation programs compare the results to what was anticipated, highlighting any inconsistencies or mistakes that need more research. Essentially, when compared to manual testing methods, automated testing improves efficiency, dependability, and accuracy by streamlining the testing process. Software engineers may guarantee the same level of quality across updates and releases by using technologies like Selenium and programming languages like Java.

II. LITERATURE REVIEW

In [1], For load test execution, the asset utilization of Selenium-based load tests in different designs was read. The two most important findings are that headless browsers use far fewer resources than other types of browser implementations. Similarly, a load driver's limit can be increased. by minimum 20% by sharing instances among client cases.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

In [2], It is discovered that in a software development life cycle (SDLC), software testing is thought to be the most important progress. The main goal of testing methods is to compare the results that are obtained with those that the end-user is aware of. The task of a tester is made easier by employing specialized software to automate the component that involves execution. This focuses on using Selenium's webdriver to test an application and demonstrate how to use it in conjunction with other tools like TestNG, Maven, and so on, enabling progressively easier methods of enhancing the quality of testing.

In [3], Consideration is given to the benefits and drawbacks of utilizing Git. A vast array of commands in Git make it easier for developers to deal with and utilize the repository. The developer can work more productively as a result. Git enables developers to branch more frequently without affecting the master code. The developer is free to act whichever they choose on the private branch. Git has the drawback that, if the merging process is not carefully supervised, it may destroy or contaminate knowledge regarding modifications made to the code history, which is crucial for ongoing data development.

In [4], It is a given that software testing is the process of running a program with the intention of identifying flaws in order to produce software that has no defects. As a result, several testing techniques have been applied over time. This area of research has seen significant advancements. In the future, this field will become increasingly significant. There are several approaches to test case creation. The author discussed the various viewpoints on software testing, including the significance of testing standards and programming techniques. Furthermore, a pertinent information regarding the many kinds of testing standards is also given.

In [5], It is evident that software testing can be done in two ways: manually or by using automation technologies to assess the product's quality, find flaws, and build user confidence. Automation tools assist in creating and running test scripts, saving money and effort compared to manual testing. The primary focus now accessible to support design and execution activity is automation tools.

III. BASICS OF SELENIUM AND TESTNG

Several software automation technologies, including Selenium IDE, Selenium RC (selenium 1.0), and Selenium webdriver (selenium 2.0), make up Selenium [7]. The integrated development environment (IDE) used to create test scripts is called Selenium. You may record, edit, and debug Selenium test cases with this Firefox plug-in [6]. It creates the test scripts and logs every action the user takes. For a long time, the primary focus of the selenium project was remote control (RC). Because Selenium RC employs the Java Script application known as Selenium Core, it is slower than Selenium Webdriver [9]. Before running the test scripts, Selenium RC demands that the server be started. It is incompatible with Ajax apps. By combining webdriver and selenium, selenium webdriver was created in order to get beyond the drawbacks of selenium RC. Another name for Selenium webdriver is Selenium 2.0 [9]. Because Selenium Webdriver communicates directly with the browser, it operates more quickly than Selenium RC. Ajax applications and a variety of web browsers are supported by the Selenium webdriver. The selenium webdriver's primary objective is to provide better support for contemporary web application testing issues. Multiple languages are supported by Selenium Webdriver for writing test scripts. The API for Selenium Webdriver is less complex than that of Selenium RC [5]. Selenium online Driver does have several limits when it comes to testing online applications, nevertheless, despite all of its benefits. Screenshot generation for failed test cases is not a feature of the Selenium webdriver. Selenium webdriver does not have built in capability to generate the test results. It depends on third party tools to generate the test reports. This limitation can be avoided by using TestNG framework.

A testing framework called TestNG was created to get around the drawbacks of the JUnit testing framework [10]. TestNG is more capable than JUnit since it offers certain additional features. All test types, including unit, functional, and integration testing, are covered by TestNG. We have combined TestNG with Eclipse in the suggested framework to produce a test report and run several test cases concurrently. All of the test cases that passed and failed are included in this TestNG report. The TestNG report needs to be modified because it is difficult to interpret. Regarding the test report, every organization has various criteria. According to the needs of the organization, we have customized the TestNG report in the suggested structure. Thus, the company is free to receive the test report in any format they choose. The failed test case URL is also included in this report. By using this functionality, developer can easily find out defects in web application.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

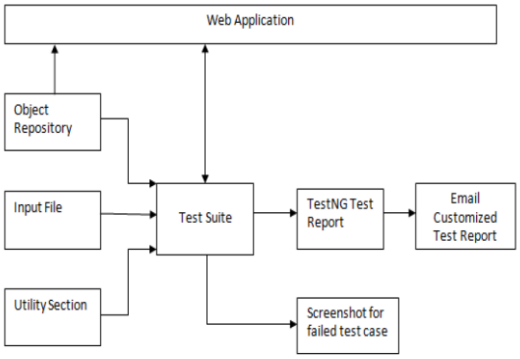

Figure 1: Framework with Selenium and TestNG

IV. TESTING

The method of conducting each and every test by hand is known as manual testing. When a module is put through manual testing, testers ensure that its functionality follows a logical flow. If a series of steps are not followed, the module is considered to have faults and is reported as a bug. A test case workflow is shown in Fig. 1.

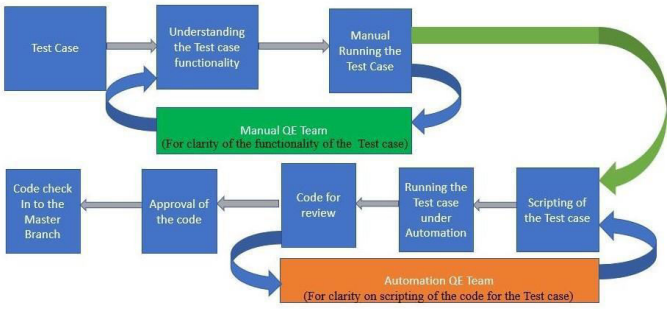

Figure 2: Testing Methodology

4.1 Manual Testing

For the initial layer of verification, manual testers are tasked with these test cases. If a module failure is discovered, it will be reported as a bug, and the development team will strive to resolve it before sending it again for testing. The manual testing team will forward the test case to automation if the module functions as intended.

If there is a reported bug. The designated developer must fix the problem and then submit it back to the original manual tester for confirmation that the problem has been fixed. The module functionality will be once again examined and verified by the manual testers before being forwarded.

4.2 Automation Testing

The next step in the testing process is automation testing. Automation is crucial since testing the same test case repeatedly for each version update of a website or browser would waste human labor. The automated approach is used to overcome this issue. Automation involves coding the test case step-by-step and validating against the expected and actual results. Because of the way it's written, you may run it again to make sure the feature works with any upgrade.

Writing a test case script is the first stage in the automated testing process. Coding is the process of writing scripts. Here, Java has been programmed using the IntelliJ IDEA platform for Selenium. TestNG and the Maven repository are the extra tools that are employed. Specifics and details of the tools used has been provided in respective chapter.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

Coding the test cases is the initial step in the automation process. The allocated automation tester will receive the steps and desired outcome from the manual testing team. Every step will be coded by the automation tester, who will then compare the outcome to what was anticipated.

The automation run takes place after the scripting is finished. Automation is used to run the test; if a scripting problem causes a failure, the code is rewritten and tested again until the issue is resolved. The code is added to the Git Repository once the script has ran repeatedly and successfully without any errors.

Every automation tester submits their code for review and completion to the automation team's own Git repository. Senior members of the automation team have permission access to the repository. If there are no code standard breaches discovered, these senior members will inspect the pushed code and approve it. Senior members can provide comments to enhance the code standard if the script does not meet the anticipated standards. The code is subsequently integrated with the source code of the automation team as a whole after approval. When this step is finished, the test case is said to have finished the testing phase.

Maintenance of test cases is another branch of the work. The test cases that have completed the testing stage and are run on regular basis still have chances of failing because of browser or website update. These updates have to be reflected back in the test cases.

4.3 Test Case Segregation

The test cases are segregated into three groups – Smoke, Sanity and Regression. This classification is based on the importance of the test cases. The importance of the test case also determines the frequency of testing of that case.

4.3.1. Smoke

Smoke test cases are the most important test cases. These test cases revolve around the fundamental functioning of the module under test. If smoke test case fails, the bug is raised with the top severity and are expected to be fixed as soon as possible. As only the important test cases are grouped in this, smoke has the lowest count of test cases in it.

4.3.2. Sanity

The test cases covered by the sanity group are significant yet do not seriously impair the functionality of the module. This group also hosts test cases that have little effect on how users interact with the module.

4.3.3. Regression

The largest number of test cases are found in the regression suite. Test cases containing validations and verifications are the focus of regression. To ensure that the bug repair or new changes haven't altered the behavior of the module, this test suite is run.

There are various testing phases involved in the work's implementation. The initial section consists of manual testing and bug correction. The automation team receives the test cases from the manual testing team once the test steps have been created. Thus, the second phase of the testing process is automation. After the tests are automated, measures are made to guarantee the reliability of the test cases in several builds. Brief segregation of the test cases is based on importance that decides the selection of test cases to be automated on priority.

V. RESULTS

1.Risk Register Navigation: Two strings are initialized: risk Name and logged In User Name. The code then retrieves the Risk Register Page class using the get Page method and calls the navigate To Risk Register method to presumably navigate to the risk register page on the application.

2.Risk Register Validation: An assertion is made using the assert Helper class to verify if the Ribbon Menu class is present on the page using the get Page method. The Ribbon Menu class likely represents a menu element on the application. This might be a way to validate that the navigation to the risk register page was successful.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

```
// Navigate to Risk Register
String riskName = TestDataUtil.getRandomName();
String loggedInUserName = currentUser.username;
getPage(RiskRegisterPage.class).navigateToRiskRegis

// Validate if Audit log is present
assertHelper.assertTrue(getPage(RibbonMenu.class).
```

Figure 3: Test Case

1.Method Definition: The code defines a method named navigate To Risk Register Page. This method likely belongs to a larger class that interacts with the web application.

2.Navigation: A line calls navigation. Navigate To Form In Left Pane Tree("Risk Analysis"). This suggests the code interacts with an object named navigation which has a method named navigate To Form In Left Pane Tree. This method likely performs the following actions: Expands the left pane tree within the web application. Selects a specific form named "Risk Analysis" from the expanded tree.

3.Frame Switching: The code then calls navigation. Switch Frame To Content(). This indicates that the application utilizes frames, and the code needs to switch to the content frame after selecting the form.

4.Verification: The final line calls wait Helper. Wait For Page To Load("Ribbon Icons. New"). This suggests the code waits for an element named "Ribbon Icons. New" to appear on the page. This element likely serves as a verification point, confirming successful navigation to the risk register.

```
public void navigateToRiskRegisterPage() {

    navigation.navigateToFormInLeftPaneTree(RiskAn
    navigation.switchFrameToContent();
    waitHelper.waitForPageToLoad(RibbonIcons.New);
}
```

Figure 4: Method

1.Try Block: The code utilizes a try block to encapsulate attempts at frame switching.

2.Frame Switching Logic: Within the try block, conditional statements implement different methods for switching frames: The first attempts to retrieve the current frame using Java Script Util. get Current Frame(). Success likely results in the function returning true. The second leverages wait Helper. wait For Frame Available And Switch, suggesting the code waits for a specific frame to be available before switching. The third involves Exception Handler. login And Continue Web Driver, indicating potential login-related exception handling during frame switching.

3.Catch Block: If any frame switching attempt in the try block encounters an exception, the catch block executes. The exception is logged using Exception Handler. Log And Continue Web Driver. The function then returns false. The switch Frame To Content function demonstrates a robust approach to switching frames in a web application. It attempts multiple methods and incorporates exception handling to ensure successful frame switching in various scenarios.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

```
public boolean switchFrameToContent() {
    try {
        if (JavaScriptUtil.getCurrentFrame(driv
            return true;
        }
        waitHelper.waitForFrameAvailableAndSwit
        return true;
    } catch (Exception e) {
        ExceptionHandler.logAndContinueWebDrive
        return false;
    }
}
```

Figure 5: switch Frame To Content

Email reports can be automatically generated by the process when automated test cases are run with the Selenium WebDriver. By promptly updating stakeholders on test execution outcomes, this feature improves test management efficiency. Teams can guarantee easy communication and monitoring of testing operations by utilizing Selenium WebDriver's features in conjunction with email integration. This will promote more efficient teamwork and prompt resolution of any issues that are discovered throughout the testing process.

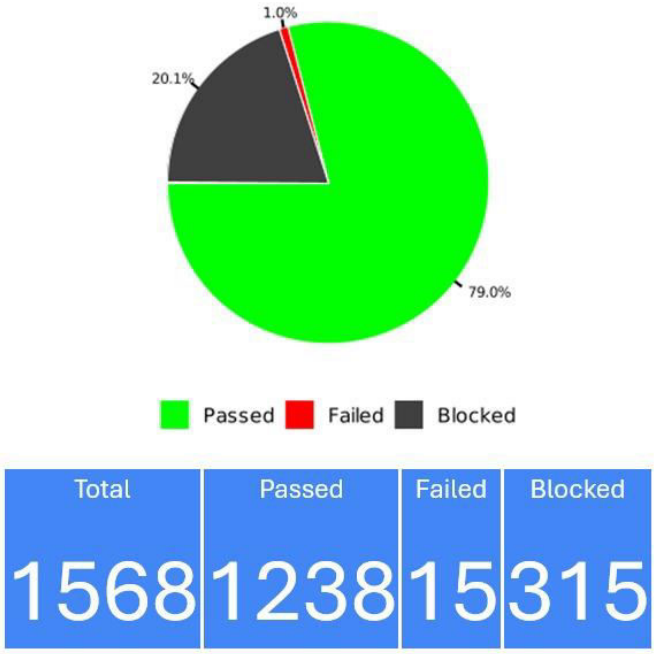

Figure 6: Report

VI. CONCLUSION & FUTURE SCOPE

Developers must meet the tight development timelines of today's software industry, automation testing becomes essential. In the past, builds happened on predetermined dates at intervals of days or even months. Continuous software builds are necessary for novel development techniques, notably those connected to object-oriented programming and languages like Java. New features are introduced to the system under test, and as with subsequent cycles, automation scripts must be developed, approved, and maintained for every release. This process

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.423

|www.ijirset.com | A Monthly Peer Reviewed & Referred Journal | e-ISSN: 2319-8753; p-ISSN: 2347-6710|

|Volume 13, Issue 6, June 2024|

|DOI: 10.15680/IJIRSET.2024.1306117|

creates a cycle. Maintenance is the key to making automation scripts more successful. As a result, the automated process finds it difficult to complete its tasks and keep up with the developer team's modifications and new requirements. Keeping the test automation scripts' overall pass rate high as well. It is now possible to draw the conclusion that, because automation is based on code and script, it is superior to manual testing of the identical situations in every software. Because automation is aided by a computer language, it is also faster than manual testing. Even though automated testing is thought to be quicker and more effective than manual testing, ad hoc testing is prohibited. On the other hand, ad hoc testing can be done manually. The set of scenarios in a given software can be tested across multiple browsers at once thanks to the automation tools' compatibility with nearly all of the current browsers. This is not possible with manual testing because more resources (manual testers + device) are needed to manually test the software on multiple browsers at once. As automation is not yet 100% reliable, expanding its scope and making it the best option for software testing will be the next big step towards the automation process's success.

Future Scope

The automation team's primary objective is to fully cover the product; after this is accomplished, increasing efficiency will come first. The degree to which automation is included into the testing process determines the breadth of automated testing. Automation makes load testing, security, and API testing easier. When a product is released, regression testing is done to look for errors. Long-term, robotics will be the way of the future. Another item that has to be tested in test coverage is artificial intelligence (AI) and machine learning (ML). Tests that are automatically generated will eventually be possible thanks to AI and machine learning. Automation is approaching a tipping point due to the quick advancements in robotics and artificial intelligence. These days, robots are capable of carrying out a wide range of tasks with little assistance from humans. Automated technologies are greatly enhancing the capabilities of the workforce in addition to carrying out iterative duties. The majority of the world's manual labor is predicted to be replaced by automated procedures in the near future. Automation is being used by a variety of industries, including banking and manufacturing, to improve quality, productivity, safety, and profitability.

REFERENCES

[1] S. M. Shariff, H. Li, C.-P. Bezemer, A. E. Hassan, T. H. D. Nguyen, and P. Flora, “Improving the testing efficiency of selenium-based load tests,” in Proceedings of the 14th International Workshop on Automation of Software Test, ser. AST ’19, Montreal, Quebec, Canada: IEEE Press, 2019, pp. 14–20. DOI: 10 . 1109 / AST . 2019.00008. [Online]. Available: https://doi.org/10.1109/AST.2019.00008.

[2] P. Ramya, V. Sindhura, and P. V. Sagar, “Testing using selenium web driver,” 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1–7, 2017.

[3] S. Just, K. Herzig, J. Czerwonka, and B. Murphy, “Switching to git: The good, the bad, and the ugly,” in 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), 2016, pp. 400–411.

[4] J. Gaur, A. Goyal, T. Choudhury, and S. Sabitha, “A walk through of software testing techniques,” in 2016 International Conference System Modeling Advancement in Research Trends (SMART), 2016, pp. 103–108.

[5] C. Jain and R. Kaluri, “Design of automation scripts execution application for selenium webdriver and testng framework,” vol. 10, pp. 2440–2445, Jan. 2015.

[6] Sherry Singla, HarpreetKaur. Selenium Keyword Driven Automation testing Framework, International Journal of Advance Research in Computer Science and software Engineering, VOL. 4,Issue 6,2014

[7] RigzinAngmo,Monika Sharma. Selenium Tool:A web based Automation testing Framework. International Journal of Emerging Technologies in Computational and Applied Science,2014.

[8] Z. Wanadan,J. Ninkang,Z. Xubo. Design And Implementation Of A Web Application Automation Testing Framework; Ninth International Conference On Hybrid Intelligent Systems, 2009.

[9] SeleniumDocumentation.[Online].(http://www.seleniumhq.org). (Accessed 15 DEC. 2014).

[10] TestNGDocumentation.[Online].(http://www.testng.org). (Accessed 25 DEC. 2014).

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details