Hybrid Testing Frameworks: Benefits and Challenges in Automation

Mohnish Neelapu

Category / Domain : QA Automation

Abstract

The Hybrid Automation Framework is structured around the core principles of test automation, which include test independence, idempotency, and clarity—essential elements for developing a robust testing suite. This document details the development of Hybrid Test Automation Framework also provides an overview of its architecture and design. The framework is user-friendly and integrates a high-level programming language that supports object-oriented design, facilitates low-level access to user interface components, and allows for rapid recovery from alterations to the user interface of the application. The framework's architecture features a multi-layered design that separates browser UI components from test scripts. It incorporates various hybrid methodologies, including Linear Scripting, Module-Based Testing, Keyword-Driven Testing, as well as Data-Driven Testing. Additionally, the paper addresses both the advantages and challenges associated with the implementation of hybrid testing automation frameworks.

Keywords: Automation, benefits and challenges, hybrid testing, hybrid architecture.

I. Introduction

In the ever-evolving realm of software development, the assurance of software application quality and reliability is of utmost importance [1]. The dynamic nature of software projects, characterized by frequent updates and enhancements, necessitates a robust testing methodology capable of keeping pace with these changes [2]. The web-based applications were carrying the increasing and increasing complicated business logic and increasing and increasing massive information platform construction, along with short release cycle nature and the rapid regeneration [3]. At the same time, the software development cycle is growing shorter and shorter; this also makes web application testing, especially functional regression testing, more difficult [4]. Conventional methods of testing are no longer capable of addressing the demands of software development, and therefore automation testing is the only option [5]. Test automation has risen as an essential tool in this scenario, providing the promise of quicker, more efficient, and more uniform testing processes [6]. In this setting, test automation has become essential, offering the ability to carry out laborious and repetitive testing activities with little assistance from humans. The choice of the right test automation framework is at the heart of effective test automation [7]. They set up the organization and the rules that regulate the test procedure, Making cooperative testing by the testers and the developers an easy possibility [8, 9] and supporting the release of high-quality software. Yet deciding on the appropriate test automation framework is no single solution situation. It involves an intuitive understanding of the types of frameworks available, their very nature and the consequent trade-offs [10]. The testing automation conducted with the assistance of testing tools or some form of programming languages to manage the testing class [11]. The primary concept behind automation testing is toward naturally accomplishing the tests are conducted without the involvement of humans; it minimizes the cost and also renders testing more effective and reliable. Numerous tools are created for this testing need [12]. An automated tool called Selenium facilitates the various way or methods to test application. The whole suite of tools has a rich set of testing functions particularly for the requirements of testing of web applications of any type [13]. These processes are incredibly versatile, offering a wide range of choices for identifying user interface components and contrasting anticipated test outcomes with real program behavior. One of Selenium's attributes to aid in running one's tests on several browser platforms [14].

A. Need of Test Automation

Companies of software must not only ensure that their software is tested adequately but also that it is tested thoroughly and efficiently. To achieve this objective, automation testing is essential. Manual testing tends to be time-consuming and susceptible to errors; however, automation addresses these issues by enabling frequent test executions, thereby enhancing confidence in the application. Additionally, executing test cases aids users in comprehending the software's functionality. A collection of automation tools can be utilized to create a regression test suite. Furthermore, automation facilitates the early detection of issues, allowing for timely resolutions.

B. Importance of Test Automation

Test automation offers numerous benefits, including expedited and more precise testing processes, along with the ability to execute tests frequently. However, as applications expand in size and complexity, the challenge of conducting tests across various contexts and configurations increases. This is where tools like Selenium Grid and other comparable technologies prove to be invaluable. The artifact that was published in Journal of Systems as well as Software describes a new method for testing graphical user interfaces (GUIs) that makes use of distributed state model inference. The objectives of this study are to overcome the difficulties that are connected with traditional graphical user interface testing, including the high costs of maintenance and the difficulty of scripting test cases. As a means of facilitating script less testing and so minimizing the reliance on scripts that are created manually, the authors offer an approach that makes use of state model inference. Their strategy entails dispersing the process of inferring the state model in order to cope with the dynamic and interactive character of modern graphical user interfaces in an effective manner [15].

C. Automation Testing Principles

Test cases are written beforehand and not vague.

Means that the test cases define clearly the precise functionality of test. And test cases never provided multiple meanings, before initiation of writing automation scripts we should keep in mind test case. If test case is not written then we should write test cases prior to automation start as well as the test cases should be crystal clear and should not have multiple meanings.

Test cases are independent

Means that every test case should not rely on the other test cases. Each test case can be executed independently.

Every test case can execute independently.

Each test case is independent means that prerequisites should also include in all the test cases and test case reverse all the changes done during testing the object or application.

Test cases cannot cover end to end scenarios.

Means that the test case document should cover all end to end scenerios. Nothing should remain after completion of test cases document. It also provides the meaning of covering all scenario by opening the first door first and closing the first door at last.

Keep test tests brief

Test cases must be written precisely and uniformly. Each test case is unique and cover more details in concise terms,

Each test case will cover one functionality

Means that each test case will cover only one functionalities. We might have happy scenarios in covering more than one functionalities but in writing test cases we must accept this rule, if we have additional functionalities to be covered then we must redesign the test cases or we can include additional test cases.

Test cases exhibit idempotent

This indicating that they can be executed multiple times as needed. Whether invoking a single test case or a multitude, such as 1000 iterations, the testing framework must possess the flexibility to accommodate this requirement.

To minimize incident test coverage

It is essential to adopt a proactive approach to incident prevention and response, preparing for the worst while hoping for the best. To mitigate the impact of external incidents, a dedicated server environment should be established, equipped solely with the necessary software to limit exposure to external events.

The selection of an automation testing tool should be conducted through thorough research.

Various companies utilize different automation tools; however, it is crucial to first assess the specific requirements of your organization or product. A well-structured proof of concept (POC) is essential for tool selection and automation implementation. As a general guideline, prior to choosing a tool, evaluate the available coverage and support resources online, including training materials, before proceeding.

Establishing coding standards for script writing

It is imperative for effective analysis and optimal code performance. These standards should be clearly defined and tailored to the programming language chosen for script development.

II. Literature review

Hyperautomation and its necessity in the current situation are briefly discussed by Haleem, A. et al. in 2021 [16]. The important responsibilities that sensors play in enhancing hyperautomation are then explained. A variety of adaptable technologies are also covered diagrammatically, including specialized workflow procedures and particular realms of solicitations related to hyperautomation. The capabilities of hyper-automation for industries are then further identified and discussed in this paper. Hyperautomation is being used to significantly improve automated activities' human element and efficiency. To address the demand for automated testing of complicated dynamic Web pages, Chen, G. et al. [17] designed and implemented a Selenium-based Web application automation testing framework in 2021. The suggested framework is assessed in the experiment from a number of angles, including comparison with other frameworks, test efficiency, and stability. The outcomes demonstrate the great test efficiency and rapid and stable support for automated testing of Web applications provided by this framework. In 2020 B. Kamini (Simi[18] we present test automation framework named TAFPro that serves as a test harness for IBM test automation tools, increasing productivity while decreasing application and life cycle cost. As companies are shifting our goal with open source tools is to make the TAFPro tool self-sufficient for test tools based on Java. We finished a CASE analysis using Selenium for TDM and met our objective. Python, a well-liked backend programming language, is used to show web page scraping in 2022 by Gudavalli, A. et al., [19]. It is also supported by test automation frameworks like Selenium. Static web sites can also have information scraped from them. Therefore, the main benefits and difficulties of using web scraping to create web apps are examined in this study. Both inexperienced and seasoned users can utilize the fully featured framework to automate website testing. Ray, V. (2022) [20] Compared to automated testing, manual testing most likely takes longer. An analyst and QA engineers carry out the procedure of manual testing. Analysts and QA engineers must install the software application before they can begin automated testing, though. Thus, the execution is done by the software. Among the many advantages of automated testing is its robust feedback loop and reduced time consumption.

A. Hybrid testing framework

In contrast, it operates based on the idea that integrating components from various structures toward develop a versatile design of test cases. The greatest prevalent implementation outline merges all the techniques previously mentioned, leveraging their strengths to compensate for any weaknesses. Fig. 1 explains the hybrid test automation model.

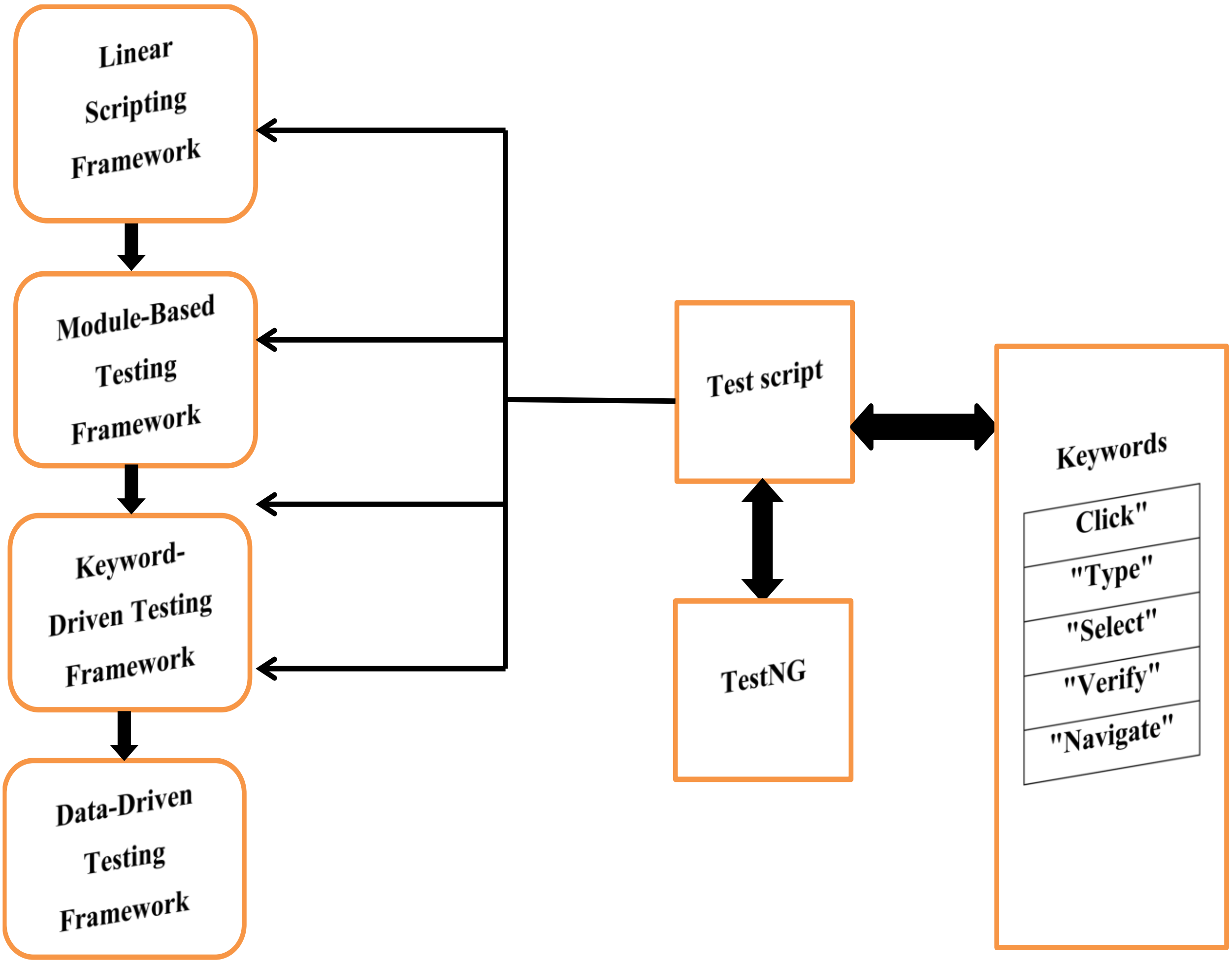
Linear Scripting Framework
Module-Based Testing Framework
Keyword-Driven Testing Framework
Data-Driven Testing Framework
Test script
TestNG
Keywords
Click"
"Type"
"Select"
"Verify"
"Navigate"

Fig 1. Proposed model for hybrid test automation

Linear Scripting Framework

This specific structure exemplifies the simplest type of automation framework, in which test cases are developed in a linear sequence. Every test case operates as a screenplay that carries out a specific achieement otherwise a sequence of actions.

Module-Based Testing Framework

In this structure, test cases are divided into sections or operations that are reusable, with every module designated to test a particular component or functionality.

Keyword-Driven Testing Framework

The framework of Keyword-Driven Testing is based on concept that test cases are formulated use advanced keywords that represent specific test instructions found within methods or functionalities. This approach promotes a clear separation of concerns, facilitating the creation of simpler and more maintainable tests. Additionally, it allows professionals from various disciplines and backgrounds to collaborate effectively at different levels of abstraction.

Data-Driven Testing Framework

The foundation of this framework lies in the distinction between test case logic and test data. Test cases are structured toward be parameterized, while the test data is maintained external.

Keywords

Click"	To click on a specific element
"Type"	To enter text into an input field
"Select"	To select an option from a dropdown list
"Verify"	To check if an element is present or has a specific value
"Navigate"	To go to a specific URL

B. Factors for Framework Selection

Choosing the appropriate test automation framework is a pivotal determination for any software testing endeavor.

Numerous elements ought to be taken into account when selecting a test automation framework in order to guarantee its harmonization with the specific demands and objectives of your project. Presented below are a few pivotal considerations for the selection of a test automation framework

Selecting right test automation framework is a critical decision for any software evaluation project. There are many factors that should be considered while choosing an automation framework for testing in order to ensure its alignment by particular needs as well as goals of your project. Given underneath are some key factors to take into account for choosing a test automation framework.

The parameters like Project Objectives and Requirements, Skill set also Expertise, Application under Test (AUT) Characteristics, Test Environment and Tools, Cost as well as Licensing.

Let's assume that you possess defined a list of N criteria (C1, C2, ., CN) to determine the effectiveness of test automation frameworks. You have also assigned weights (W1, W2, and WN) to every criterion according to their relevance toward your project.

Aimed at each test automation framework F, a comparison is made against every principle, leading to the assignment of a score (S1, S2, SN) that indicates its degree of compliance with said principle. These scores are usually quantified on a scale from 1 to 10, where 1 represents a poor performance and 10 represents an outstanding performance.

The equation used toward calculates Total Score (TS) for a particular framework F is as follows: $TS(F) = (W1 * S1) + (W2 * S2) + ... + (WN * SN)$.

Essentially, every criterion score is multiplied by its corresponding weight, and the final goods are then added up toward get the entire score of framework.

C. Benefits of Hybrid testing framework

Our hybrid automation framework offers numerous advantages for organizations, irrespective of their technology stack:

Minimized Manual Labor: Automation greatly diminishes the necessity for manual testing, allowing valuable resources to be redirected towards strategic initiatives.

Thorough Test Coverage: We ensure complete test coverage across all platforms and functionalities, independent of the underlying technology, thereby mitigating the risk of undetected bugs and issues.

Improved User Experience: A smooth and error-free experience across all platforms enhances user satisfaction and loyalty, fostering business growth, regardless of the selected technology stack.

Instant Performance Insights: Automation delivers critical insights into real-time user interactions and application performance, facilitating further optimization and enhancement, regardless of the technology employed.

Elevated Application Quality: Our framework plays a crucial role in producing high-quality applications, resulting in increased customer engagement and business expansion, irrespective of the development technology utilized.

D. Challenges

The challenges occurred in proposed hybrid test automation framework

Linear Scripting Framework

- The absence of reusability becomes evident when modifications are made to the user interface.
- Additionally, it does not support the execution of multiple data sets.
- Furthermore, maintenance becomes challenging and costly when updates to the user interface and feature improvements are implemented.

Module-Based Testing Framework

- Establishing the framework necessitates a certain level of technical expertise.
- The process of analyzing and pinpointing reusable flows demands a considerable investment of time.
- Additionally, when modifications are made to the code or script, the presence of embedded test data within the test script can pose challenges.
- Furthermore, due to the hard-coded nature of the data in the script, executing the same test case with different data values requires alterations to the data after each execution.

Keyword driving testing framework

- Extended development duration is necessary.
- Establishing and stabilizing the framework, as well as developing the function libraries, requires significant time investment.
- The inherent flexibility leads to longer and more intricate test cases. A substantial level of automation expertise is essential.
- The initial investment for framework setup is considerable, with the advantages becoming apparent primarily in large applications where test scripts will be sustained over several years.

Data-driven testing

- The initial configuration period for both Test Scripts and Test Data is considerable.

- Identifying and resolving bugs may pose challenges due to the tester's limited familiarity with programming languages.
- It is essential for testers to possess strong technical expertise.
- Additionally, the involvement of seasoned automation testers is crucial for the effective design of the framework.

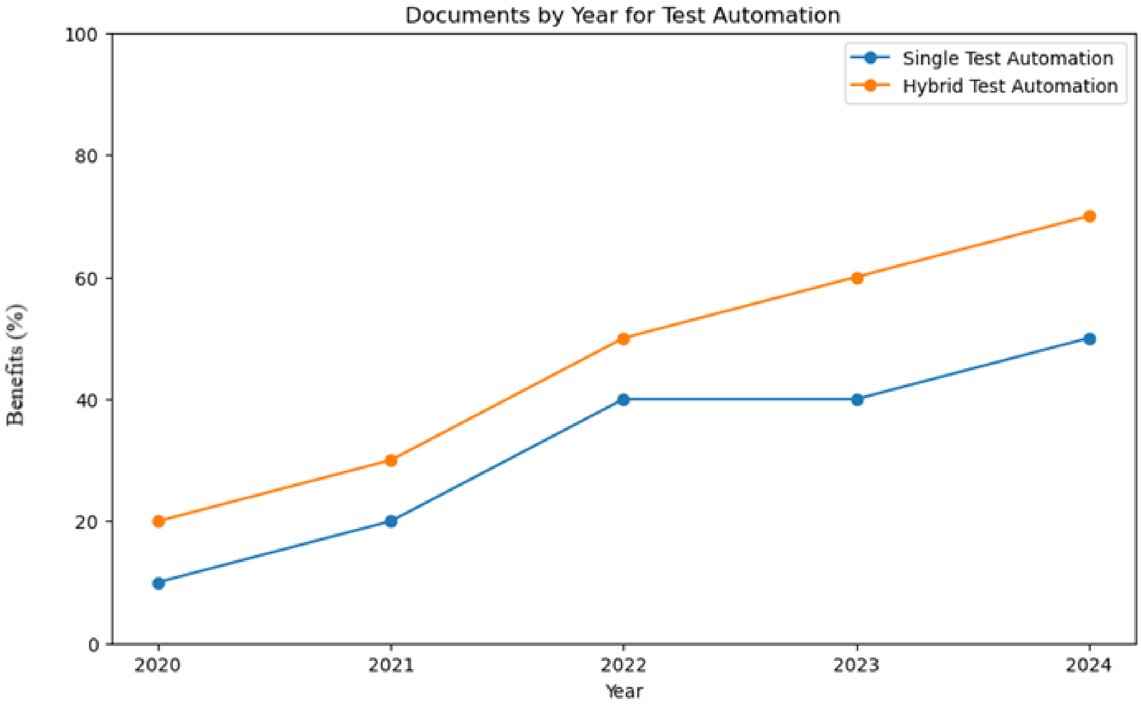

Fig. 2. Benefits analysis for single and test automation.

III. FUTURE DIRECTIONS

Historically, Testing frameworks for automation have been the backbone of software quality control, ensuring the efficacy and accuracy of verification of software functionality. However, the current trend is poised to transform the automation testing landscape, introducing new methodologies and capabilities that can potentially change the way software is assured. Figure 2 explains the benefits % of using single and hybrid automation. Here, the hybrid test automation have more benefits.

Integration of AI Vs ML: AI as well as machine learning are being more and more incorporated into frameworks for test automation to facilitate more intelligent creating test cases, using predictive analytics to rank tests, and self-governing test execution.

Test Automation of AI/ML Applications - While companies move forward with creating applications based on artificial intelligence and machine learning, there is an increased need for test automation frameworks to adapt their process so that these applications can be tested and addressed effectively as these technologies bring their own set of unique challenges.

Cloud based Testing - Test automation frameworks are expected toward make more use of cloud resources in a bid toward attain scalability also facilitate similar test execution. This will allow teams to carry out larger-scale testing without having to make huge savings in substructure.

Codeless Test Automation - While codeless test automation has existed in market aimed at a while now, the future will see more adoption of automation that is low-code and no-code. This new methodology enables developers and testers to run automated tests without the need for requirement of coding expertise. Not only does it provide businesses with a solid platform to create suites aimed at automation, but it also makes upkeep easier and decreases the price of automation overall.

IV. Conclusion

The Hybrid Framework is robust, straightforward to implement, user-friendly, easily expandable, and simple to maintain. It operates independently of both technology and platform, and it is distinct from Test Design. Once established, it enhances the speed and ease of maintenance of automated test cases by utilizing reusable libraries, thereby reducing costs. This paper examines the implementation of the Hybrid Test Automation Framework, highlighting the various advantages of four frameworks, the challenges encountered, and how it can be effectively utilized in automation. Additionally, we have addressed the framework's folder structure and execution flow, emphasizing its ease of extension and maintenance. Future work will involve researching a broader range of applications to support more complex test cases, including an exploration of mobile implementation. Concurrently, we will further investigate and enhance exception handling in testing for greater effectiveness.

Reference

1. D. S. Patil, and P. B. Pawar, "Comparative Study of Test Automation Framework's for Web Application Testing," vkykspu, pp. 31.
2. D. Andrade, "Challenges of automated software testing with robotic process automation RPA-A comparative analysis of UiPath and automation anywhere," International Journal of Intelligent Computing Research (IJICR), vol. 11, no. 1, pp. 1066-1072, 2020.
3. A. Byri, A. Joshi, F. Antara, S. P. Singh, O. Goel, and O. Tharan, "The Role of Automated Testing in Silicon Design for Data Centers," 2020.
4. A. M. Imbulpitiya, and S. Jayalal, "Test Automation Frameworks for Handling Requirement Changes in Web Applications".
5. G. Bathla, K. Bhadane, R. K. Singh, R. Kumar, R. Aluvalu, R. Krishnamurthi, A. Kumar, R. N. Thakur, and S. Basheer, "Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities," Mobile Information Systems, vol. 2022, no. 1, pp. 7632892, 2022.
6. S. Pochu, S. R. Kathram, and S. D. Engineer, "Synergizing Automation and Human Insight: A Comprehensive Approach to Software Testing for Quality Assurance," Journal of Multidisciplinary Research, vol. 8, no. 01, pp. 51-62, 2022.
7. Hoss, M., Scholtes, M. and Eckstein, L., 2022. A review of testing object-based environment perception for safe automated driving. Automotive Innovation, 5(3), pp.223-250.
8. S. Pradhan, and V. Nanniyur, "Large scale quality transformation in hybrid development organizations–A case study," Journal of Systems and Software, vol. 171, pp. 110836, 2021.
9. Kumar, G. and Chopra, V., 2022. A NOVEL APPROACH FOR TEST DATA GENERATION. ICTACT Journal on Soft Computing, 12(4).
10. Xu, R., Guo, Y., Han, X., Xia, X., Xiang, H. and Ma, J., 2021, September. Opencda: an open cooperative driving automation framework integrated with co-simulation. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC) (pp. 1155-1162). IEEE.
11. Nass, M., Alégroth, E. and Feldt, R., 2021. Why many challenges with GUI test automation (will) remain. Information and Software Technology, 138, p.106625.

12. Sun, J., Zhang, H., Zhou, H., Yu, R. and Tian, Y., 2021. Scenario-based test automation for highly automated vehicles: A review and paving the way for systematic safety assurance. IEEE transactions on intelligent transportation systems, 23(9), pp.14088-14103.
13. García, B., Munoz-Organero, M., Alario-Hoyos, C. and Kloos, C.D., 2021. Automated driver management for Selenium WebDriver. Empirical Software Engineering, 26, pp.1-51.
14. G. Chen, G. Chen, D. Wu, Q. Liu, L. Zhang, and X. Fan, "A selenium-based web application automation test framework," In 2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA). IEEE, vol. 2, pp. 257-261, 2021, December.
15. Bathla, G., Bhadane, K., Singh, R.K., Kumar, R., Aluvalu, R., Krishnamurthi, R., Kumar, A., Thakur, R.N. and Basheer, S., 2022. Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities. Mobile Information Systems, 2022(1), p.7632892.
16. Haleem, A., Javaid, M., Singh, R.P., Rab, S. and Suman, R., 2021. Hyperautomation for the enhancement of automation in industries. Sensors International, 2, p.100124.
17. Chen, G., Chen, G., Wu, D., Liu, Q., Zhang, L. and Fan, X., 2021, December. A selenium-based web application automation test framework. In 2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA) (Vol. 2, pp. 257-261). IEEE.
18. B. Kamini (Simi), "Hybrid Test Automation Framework for managing Test Data," International Journal of Pure and Applied Mathematics, vol. 118, no. 9, pp. 265-277, 2020.
19. Gudavalli, A. and JayaLakshmi, G., 2022. Implementation of Test Automation with Selenium Webdriver. Grenze International Journal of Engineering & Technology (GIJET), 8(1).
20. Ray, V., 2022. Inside Clinical Arena Automated Evaluation Using Selenium testing framework. YMER Digital, 21.