WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Software Testing Using Selenium with Python

Rutuja Korde[1],Bharati Wadekar[2]

Dr. D. Y. Patil Arts, Commerce & Science College, Pimpri, Pune, Maharashtra, India

ABSTRACT :

Software testing is a crucial process in software development to ensure product quality, functionality, and reliability. With the growing demand for continuous integration and delivery, automated testing has become essential. Selenium, an open-source automation testing tool, combined with Python's simplicity and power, provides a robust framework for automating web application testing. This paper explores the implementation of Selenium with Python for automated testing, emphasizing its architecture, advantages, and a practical case study of automating a login test. The study highlights how Selenium with Python enhances testing efficiency, reduces manual effort, and supports reliable test case execution.

Keywords: Software Testing, Selenium, Python, Automation Testing, Web Application Testing

1. Introduction

Software testing ensures that applications meet the desired quality standards and function correctly across multiple platforms. Manual testing, though effective for exploratory testing, is time-consuming and prone to human error. Automation testing tools such as Selenium have revolutionized the process by enabling repetitive and regression tests to be executed automatically. Python's easy syntax and extensive libraries make it an ideal choice for implementing automated testing frameworks.

2. Literature Review

Several researchers have studied automation testing frameworks and their efficiency. According to prior studies, Selenium is among the most widely adopted tools for web automation due to its cross-browser compatibility and support for multiple programming languages. The integration of Selenium with Python has gained popularity because Python provides an expressive syntax and a rich ecosystem of testing libraries such as PyTest and Unittest. Comparative analyses have shown that Selenium-based automation offers higher reusability and maintainability compared to traditional manual testing approaches.

3. Methodology

This research adopts a practical implementation-based approach to demonstrate Selenium's integration with Python for automating web testing. The methodology involves setting up the Selenium framework, configuring the Python environment, and executing automated test scripts. The process focuses on automating functional test cases, particularly login validation scenarios, which are common in most web applications.

3.1 Selenium Framework Overview

Selenium is an open-source framework for automating web browsers. It supports multiple programming languages, including Python, Java, and C#. Selenium WebDriver allows interaction with web elements like buttons, text boxes, and links. The framework consists of four main components: Selenium IDE, Selenium RC, Selenium Grid, and Selenium WebDriver, with WebDriver being the most advanced and widely used component.

3.2 Python Integration

Python simplifies the creation of automated test scripts with its rich libraries and simple syntax. Integrating Selenium with Python involves installing Selenium using pip and configuring a WebDriver (e.g., ChromeDriver). Test cases can be written using frameworks like PyTest for modularity and maintainability.

4. Case Study: Automated Login Test using Selenium with Python

To demonstrate practical implementation, a case study was conducted to automate the login functionality of a sample web application using Selenium with Python. The goal was to verify the correct behavior of the login module when valid and invalid credentials are provided.
The following Python script was used for the automation test:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get('https://example.com/login')
driver.find_element(By.ID, 'username').send_keys('testuser')
driver.find_element(By.ID, 'password').send_keys('testpass')
driver.find_element(By.ID, 'login').click()
assert 'Dashboard' in driver.title
driver.quit()
```

This script opens a web browser, navigates to the login page, enters credentials, and verifies whether the login was successful by checking the page title. Such automation eliminates repetitive manual testing and enhances reliability.

5. Results and Discussion

The automated login test was executed successfully, validating that Selenium with Python can efficiently perform browser-based automation tasks. Execution time was significantly reduced compared to manual testing. Moreover, test reusability improved as scripts could be easily modified for other web modules. The results indicate that Selenium with Python provides a scalable and maintainable solution for automated web testing.

6. Conclusion and Future Work

This research demonstrates that Selenium integrated with Python offers a powerful framework for automated web application testing. It enhances accuracy, reduces human error, and saves time in regression testing. Future work can focus on integrating Selenium with continuous integration tools like Jenkins and incorporating AI-based test case generation to further improve automation efficiency.

REFERENCES

[1] SeleniumHQ, “Selenium WebDriver Documentation,” Available: https://www.selenium.dev/documentation.
[2] Python Software Foundation, “Python Documentation,” Available: https://docs.python.org.
[3] T. Kumar, et al., “Comparative Study of Automated Testing Tools: Selenium, QTP, and LoadRunner,” International Journal of Computer Science, 2021.
[4] A. Sharma and R. Gupta, “Implementation of Web Testing using Selenium and Python,” IJERT, vol. 10, no. 5, 2022.