EasyChair Preprint

№ 13602

Towards the Application of Automated Testing in Education Systems

Atef Tayh Nour El-Din Raslan and Abdullah Mahdy

June 9, 2024

Towards applying An Automation Testing Framework for Educational Web Application

Dr. Atef Raslan[1] and Abduallah Ahmed Mahdy[2]

***Abstract*—**Automated testing has revolutionized various industries by enhancing efficiency and reliability in software development. This research explores the potential application of automated testing methodologies in the education system. By leveraging automated testing tools and techniques, educators can streamline assessment processes, ensure consistency in grading, and provide timely feedback to students. This study investigates the feasibility of implementing automated testing in educational settings, considering factors such as test design, integration with learning management systems, and the impact on teaching methodologies. Through a comprehensive review of existing literature and case studies, this research aims to identify challenges, opportunities, and best practices for incorporating automated testing into the education system. The findings of this study offer insights into how automated testing can improve assessment practices, promote student engagement, and promote personalized learning experiences in the digital age.

***Keywords*—** Automated testing, education system, feedback mechanisms, learning management systems.

I. Introduction

software applications these days are built and worked in web environments; they are also called web-based applications. A web-based application is a program that is accessed over a network connection, rather than existing within a device.

memory, it often runs inside a web browser. Their advantage is the ability to share functions and data in the system; support multiple environments working. Therefore, providing quality assurance solutions for web-based applications is becoming more and more important. Software testing is a technique used widely in software quality assurance. It is a process of executing a program or application with the intent of finding the software bugs. It can also be stated as the process of validating and verifying that a software program or application meets technical requirements that guided its design and development.

In practice, software applications can be tested using two methods: manual testing and automation testing. However, manual testing can become tedious and error-prone over time. To overcome these drawbacks and streamline the testing process while saving time and resources, automation testing is employed. This involves utilizing automation tools to execute a suite of test cases automatically.

This paper proposes the development of a Python-based automation testing framework specifically designed for web-based applications. This framework seamlessly integrates with

- POM elements in a resource file.
- Use keywords to encapsulate complex interactions.

-The Robot Framework is a generic open-source automation framework for acceptance testing, acceptance test-driven development (ATDD), and robotic process automation (RPA) known for its simplicity and extensibility. And that makes it an excellent choice for testing educational web applications.

- It uses keyword-driven testing and supports the use of external libraries.

The contributions of this paper encompass:

1. A module-driven approach to web-application testing.
2. A high-performance and secure framework capable of executing multiple test scripts simultaneously.[2]

The structure of the paper is as follows:

- Section 2 provides background information on web automation testing.
- Section 3 presents the Related Works for Robot Framework.
- Section 4 Presents Architecture of the Proposed Framework.
- Section 5 Presents the implementation OF the Framework
- Section 6 offers a case study of an Education management system to illustrate the framework's application.
- Finally, Section 6 Conclusion Of the paper and suggests future research directions.

.

[1] Department of Computer Science, Faculty of Statistical Studies and Research, Cairo University, Egypt, Dr.Atef.Raslan@gmail.com.

2 Department of Software Engineering, Faculty of Statistical Studies and Research, Cairo University, Egypt, Abdullah.mahy@yahoo.com.

II. Background

Automation testing is a method of software testing that leverages software tools to perform tests and then compares actual test results with expected results, often with minimal human intervention.[6]The process includes several crucial phases:

1. **Test Tool Selection**: This is a vital initial step in any organization before beginning automation. The choice of test automation tool heavily depends on the technology on which the Application Under Test (AUT) is built. For instance, this research provides an automated testing tool specifically for web applications.
2. **Defining the Scope of Automation**: This involves determining which areas of the AUT will be automated. Key considerations include scenarios involving large data sets, common functionalities across applications, and the complexity of test cases.
3. **Planning, Design, and Development**: This phase encompasses building the strategy and plan for automation. It includes tasks such as selecting automation tools, designing the framework, identifying in-scope and out-of-scope items, and preparing the automation testbed.
4. **Test Execution**: This involves the actual running of automated tests or integrating the automation framework with built systems. The execution can be performed directly via the automation tool or through a Test Management tool that triggers the automation tool. It also includes verifying and analyzing the results generated by automated tests and logging any detected failures.
5. **Maintenance**: Ensuring that automated tests are regularly updated to accommodate changes in the AUT, maintaining their purpose and reliability.

Web-based application testing poses unique challenges due to its execution environment's heterogeneity, multi-platform support, autonomy, cooperation, and distribution. Various approaches are proposed to address these challenges, focusing on:

- **Functionality Testing**: Verifies if the product meets the intended specifications and functional requirements, including testing all site links, formats used for user information exchange, database connections, cookies, and HTML/CSS verification.
- **Usability Testing**: Ensures the application is user-friendly.
- **Interface Testing**: Checks the interface and data flow between systems, including the application, web, and database servers.
- **Database Testing**: Critical for ensuring comprehensive database integrity and performance.
- **Compatibility Testing**: Ensures the application performs well in different contexts.
- **Performance Testing**: Assesses the site's ability to handle various loads.
- **Security Testing**: Verifies the application's security against data theft and unauthorized access.

In the realm of automation testing, two primary methods exist: manual and automated. Manual testing, while valuable, can become monotonous and error prone. Automated testing addresses these drawbacks by reducing the time and cost involved in the testing process, using tools to execute test case suites.

III. Related work

-Previous Studies on Automated Testing in Education
Research have shown the effectiveness of automated testing in educational technology.[3]

-Automated testing frameworks like Selenium and TestNG have been widely used. However, the Robot Framework, with its keyword-driven approach, offers an easier learning curve and better maintainability, which is advantageous for educational web applications. .[7]

- While Selenium WebDriver and TestNG are powerful, they require extensive coding knowledge.

Robot Framework, in contrast, allows for the creation of readable and maintainable test scripts through its keyword-driven approach, making it accessible to educators and testers with limited programming experience.

IV. Architecture of the Proposed Framework

In this section, we introduce the architecture of the proposed automation framework specifically designed for educational web applications. Following the architectural overview, we delve into its detailed design. The proposed testing approach, which is both modularity-driven and based on the Page Object Model (POM), facilitates easier development and maintenance of test cases. The main steps include:

1. Generating POM from web applications.
2. Developing test scripts.
3. Executing test scripts.
4. Logging the test results.

Following this methodology, we construct the automation framework with the following key characteristics:

- **Reusability**: Web elements are defined once and reused across all test cases.
- **Functionality-Based Test Cases**: Test cases are constructed to align with specific functionalities of the web applications.

The architecture of the proposed framework is depicted in Figure 1. Pages and objects are automatically generated from web applications using the Selenium Page Object Generator.[6]The core components of the framework involve constructing test scripts from the POM model and executing test cases using Robot Framework. The process is as follows:

1. **Development of Test Cases**: Users develop test cases using the POM model. The framework then creates test scripts that correspond to these test cases based on the POM model.
2. **Test Plan Execution**: The framework converts the user-defined test plan into an XML-based file, which is executable by Robot Framework. The framework runs the test cases on the browsers using Robot Framework and the Selenium Library.
3. **Result Logging**: Test results are captured and stored in a file for review and analysis.

The proposed framework also incorporates security policies to prevent test scripts from unintentionally accessing restricted resources. The secure model is based on a sandbox mechanism deployed within the Java platform. Users have the flexibility to modify these security policies by customizing the configuration file, which is formatted similarly to a Java policy file.

Additionally, Robot Framework and Selenium are integrated into the framework, allowing for customization via an XML-based configuration. This integration enables efficient execution and management of test cases, ensuring comprehensive and reliable testing of educational web applications.

. This separation of concerns improves the organization of test scripts, making them more maintainable and scalable.

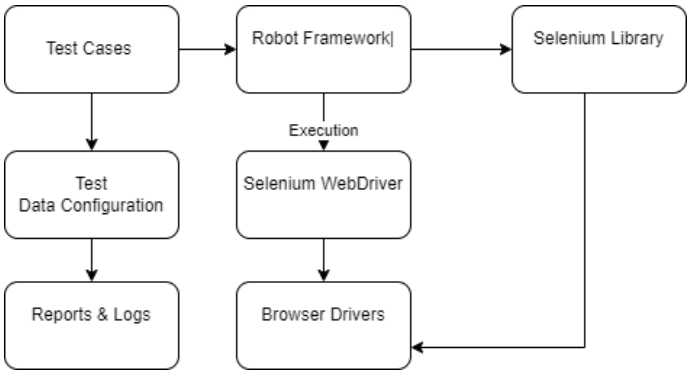


Figure 1. Architecture of robot framework [1].

Figure 1 illustrates the architecture of the proposed framework, showing the interaction between various components, including Test Cases, Robot Framework, Selenium Library, Selenium WebDriver, Browser Drivers, and others.

V. Implementation

Setting Up the Environment: [1]

- Installing Python.
- Installing Robot Framework:
  Detailed steps to install Robot Framework and its dependencies.
- Required Libraries and Tools:
  List of required libraries, including Selenium Library and Browser drivers.

Creating Test Cases:

User Registration: Test case to verify the registration functionality.
User Login: Test case to check the login functionality.
Create Courses: Test Cases to verify the courses are created.
Create Subjects: Test Cases to verify the subjects are created.
Course Enrollment: Test case to test course enrollment.
Subject Enrollment: Test case to test Subject enrollment.
Content Access: Test case to validate access to course content.
Quiz Taking: Test case to ensure quizzes can be taken and results are recorded.

To illustrate the use of the proposed framework, we define ten test cases as follow [Table1]

Table 1. Test cases of the Educational Web Application

No	TC Name	Description
1	TC01	Open College Management System
2	TC02	Login With Valid Username and Password
3	TC03	Login With Invalid Username and Password
4	TC04	Check Functionality of add course
5	TC05	Check Functionality of add Subjects
6	TC06	Check Functionality of add Staff
7	TC07	Check Functionality of add Students
8	TC08	Check Functionality of Manage Staff
9	TC09	Check Functionality of Manage Students
10	TC10	Delete an account

Following the proposed approach, Robot users need to Add Courses, subjects, Students and Staff.
objects of home, log in, and register pages. objects of home, log in, and register pages In the Robot Framework, keywords correspond to the functional features of the pages, while variables define attributes for web element identifiers.

Sample Code:
Explanation of the Sample Code: Step-by-step explanation of a sample test suite.

Listing 1. A part of Login page Test Case

```
*** Settings ***
Library  SeleniumLibrary
Library  String
*** Variables ***
${Username_Text}     //*[@name="email"]
${Password_Text}     //*[@name="password"]
${Login_btn}         //*[@type="submit"]
${Invalidmsg}       //*[@class="alert alert-danger text-center "]

*** Keywords ***
Open_Chrome
    Set Selenium Implicit Wait  5
    Open Browser    http://127.0.0.1:8000/  chrome
    Maximize Browser Window
Close_Chrome
    Close Browser
Login_Successfully
    Input Text      ${Username_Text}    User.Email@gmail.com
    Input Password  ${Password_Text}    123456
    Click Element   ${Login_btn}
```

Listing 2. A part of Add Course page Keywords.

KeywordsAddCourse.robot

```
Resource    Common.robot

*** Variables ***
${CourseLink}       //a[@class='nav-link' and @href='#' and .//i[contains(@class, 'fa-bookmark')]]
${AddCourseLink}    //*[@href="/course/add"]
${CourseName}       //*[@id="id_name"]
${AddButton}        //*[@type="submit"]
*** Keywords ***
Click on Open Coursses
    Click Link    ${CourseLink}
Click On AddCourses
    Click Link    ${AddCourseLink}
Input The Course Name
    Input Text    ${CourseName}    Python
Click On Add Button
    Click Button    ${AddButton}
```

Listing 3. A part of Add Course page Test Case

AddCourse.robot

```
*** Settings ***

Library     SeleniumLibrary
Resource    ../Resources/Common.robot
Resource    ../Resources/KeywordsAddCourse.robot
Suite Setup    Run Keywords    Open_Chrome    Login_Successfully



*** Test cases ***
Check Functionality of add course
     Click on Open Coursses
     Click On AddCourses
     Input The Course Name
     Click On Add Button


    Close Browser
```

In the Robot Framework, you can create a test plan by defining a suite that specifies which test cases to run and how many times to run each test. You can use the ***Repeat Keyword*** feature to run specific test cases multiple times. Here is how you can modify the description to fit the Robot Framework:

Listing 4. A part of Running Specific Test Cases

```
*** Settings ***
Resource    ../Resources/Common.robot


*** Variables ***
${RUN_COUNT}    2

*** Test Cases ***
Run Test Plan
    Run Test Case    TC01
    Run Test Case    TC02
    Run Test Case    TC02
    Run Test Case    TC03
    Run Test Case    TC09

*** Keywords ***
Run Test Case
    [Arguments]    ${test_case_name}
    Run Keyword    ${test_case_name}
```

Or You can Run All test suits Using This Command
Ex: robot -d results/ tests/

Figure 2. Result After of Running Specific Test Cases

```
Tests                                                                 | FAIL |
5 tests, 3 passed, 2 failed
==============================================================================
Output:  C:\Users\abdullah.ahmed\IdeaProjects\collageMangmentSystem\results\output.xml
Log:     C:\Users\abdullah.ahmed\IdeaProjects\collageMangmentSystem\results\log.html
Report:  C:\Users\abdullah.ahmed\IdeaProjects\collageMangmentSystem\results\report.html
Created TensorFlow Lite XNNPACK delegate for CPU.
```

Figure 3. Tests Report After of Running Specific Test Cases

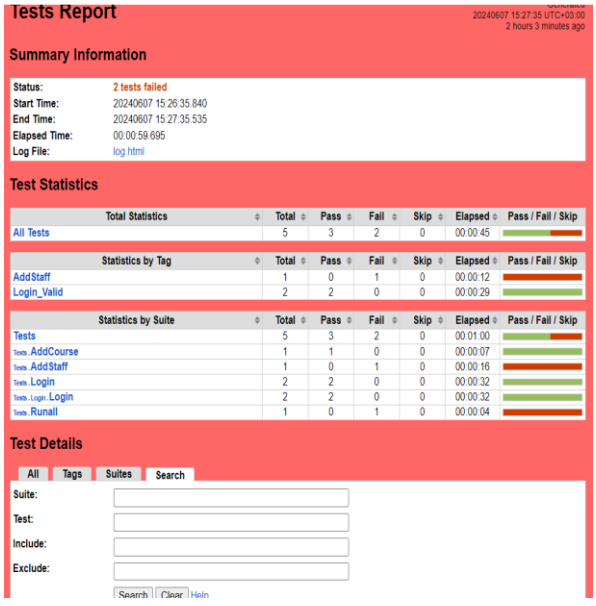
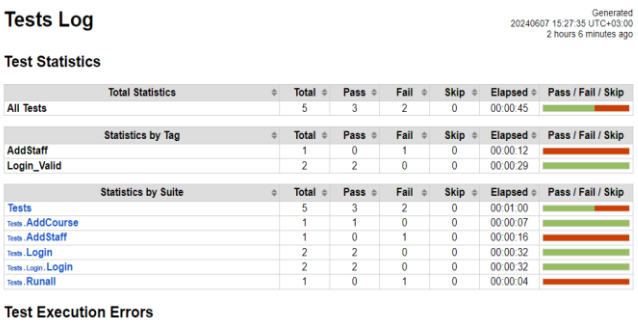


Figure 4. Tests Log After of Running Specific Test Cases



Tests Log

Generated
20240607 15:27:35 UTC+03:00
2 hours 6 minutes ago

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	5	3	2	0	00:00:45	

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
AddStaff	1	0	1	0	00:00:12	
Login_Valid	2	2	0	0	00:00:29	

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Tests	5	3	2	0	00:01:00	
Tests.AddCourse	1	1	0	0	00:00:07	
Tests.AddStaff	1	0	1	0	00:00:16	
Tests.Login	2	2	0	0	00:00:32	
Tests.Login.Login	2	2	0	0	00:00:32	
Tests.RunAll	1	0	1	0	00:00:04	

Test Execution Errors

To run all test cases, the test plan just needs to define Test suite—All—1.

Figure 4 shows the result of the execution of test plan five. It graphically reports.

that 03 test cases are successful, and 02 test case is failed.

VI. Discussion

Advantages of Using Robot Framework:

The Robot Framework offers several advantages, including ease of use, readability, and

maintainability. Its keyword-driven approach makes it accessible to non-developers, while its

integration with Selenium Library provides robust web automation capabilities.

Challenges and Limitations

While the Robot Framework is powerful, it does have some limitations. The initial setup can be.

complex and integrating with other tools may require additional effort. However, these challenges

can be mitigated with proper documentation and support.

Comparison with Other Testing Approaches

Compared to other testing frameworks, the Robot Framework's keyword-driven approach offers a higher level of abstraction, making it more accessible to a broader audience. Its flexibility and

extensibility further enhances its appeal for educational web application testing.

VII. Conclusion and Future Work

Summary of Findings

The study demonstrated that the Robot Framework is an effective tool for automated testing of

educational web applications. Its ease of use and flexibility make it an asset for ensuring the

quality and reliability of these applications.

Future Enhancements

Future work could focus on integrating the Robot Framework with continuous integration (CI) tools.

and exploring additional libraries and plugins to enhance its capabilities further.

VIII. References

[1] Robot Framework Documentation. Retrieved from https://robotframework.org/

[2] Selenium Library Documentation. Retrieved from https://robotframework.org/SeleniumLibrary/.

[3] Educational Web Application Testing: Challenges and Best Practices. Journal of Educational

Technology, 2023

[4] Dave, R., & Patel, R. (2019). A Web Page for Automation Test Framework. International Journal of Computer Applications, 178(14), 42–47. https://doi.org/10.5120/ijca2019918911

[5] Tarhini, A., Fouchal, H., & Mansour, N. (n.d.). A Simple Approach for Testing Web Service Based Applications.

[6] Nguyen, H. P., Le, H. A., & Truong, N. T. (2019). jFAT: An automation framework for web application testing. Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST, 266, 48–57. https://doi.org/10.1007/978-3-030-06152-4_5

[7] Nikfard, P., Suhaimi bin Ibrahim, A., & Hossein Abolghasem Zadeh, M. (n.d.). *A Comparative Evaluation of approaches for Web Application Testing*.

[8] Nikfard, P., Suhaimi bin Ibrahim, A., & Hossein Abolghasem Zadeh, M. (n.d.). *A Comparative Evaluation of approaches for Web Application Testing*.

Prasad, P. (n.d.). *AUTOMATION TECHNOLOGY AND TOOLS*. www.iejrd.com

[9] Nikfard, P., Suhaimi bin Ibrahim, A., & Hossein Abolghasem Zadeh, M. (n.d.). *A Comparative Evaluation of approaches for Web Application Testing*.

[10] St Henderi, 1. (2020). *Software Testing as Quality Assurance on Web Application*. http://domain_name.tld/index.php/member/C_monitor