Kassahun Fuchuro

Development of Summary Exercises for a Robot Framework Software Automation Course

Metropolia University of Applied Sciences
Bachelor of Engineering
Information and Communications Technology
Bachelor's Thesis
23 May 2024

Abstract

Author:	Kassahun Fuchuro
Title:	Development of Summary Exercises for a Robot Framework Software Automation Course
Number of Pages:	27 pages
Date:	23 May 2024
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communication Technology
Professional Major:	Software Engineering
Supervisors:	Janne Salonen, Head of Department ICT

With an emphasis on the use of the Python and Selenium frameworks, this thesis investigates the conception and execution of summary exercises for a Robot Framework Software Automation Course. The need to bridge the knowledge gap between theory and practice becomes critical as the demand for skilled software automation skills rises.

The study highlights the value of practical experiences in enhancing students' comprehension of the Robot Framework and was conducted as part of the Information and Communication Technology programme. This effort attempts to offer a thorough learning experience by exploring the nuances of Python and Selenium within the framework of Robot Framework.

The tasks have been carefully designed to support theoretical ideas, encourage the use of problem-solving techniques, and foster a practical grasp of software automation. The thesis evaluates the impact of these exercises on students' learning outcomes, ensuring alignment with industry demands and educational objectives.

Keywords: Robot Framework, Software Automation, Python, Selenium

Contents

Figures

List of Abbreviations

API	Application Programming Interface
ATDD	Acceptance Test Driven Development
CI/CD	Continuous Integration/Continuous Delivery
FTP	File Transfer Protocol
IBM	International Business Machines
n.d.	No date
QA	Quality Assurance
RF	Robot Framework
RIDE	Integrated Development Environment for RF
ROI	Return on Investment
RPA	Robotics Progress Automation
SOAP	Simple Object Access Protocol
TA	Test Automation
UI	User Interface

1 Introduction

These days, conducting test automation (TA) is a prerequisite for most applications and developers use different TA tools to perform this task. Even though there are many TA software, Robot Framework (RF) is by far the most used one. TA is incredibly versatile and user-friendly because it operates using keywords. Tools like Robot Framework, which help improve our procedures, are like superheroes in the ever-changing world of software development. It is constructed using Python! Thus, software developers may create and execute excellent automated tests with its user-friendly interface, particularly for complex online applications. (Bisht.S, n.d.)

This thesis is driven by the realization that to effectively understand RF, developers must have real-world, practical experience. Though theoretical understanding is fundamental, the inclusion of thoughtfully crafted summary tasks aids in teaching students how to distil the most important concepts from a work, enhances the ability to think critically and solve problems of the real-world scenarios, and increases confidence for students.

By creating challenges that push students to apply their knowledge in real-world situations while still reinforcing theoretical principles, this research aims to close the gap that exists between theoretical understanding and practical ability. Enhancing the robot framework Software Automation Course students' educational experience and giving them the skills and self-assurance, they need to succeed in the software development business is the ultimate goal.

To sum up, Software Automation Course Using the RF to investigate test automation, this Python based program improves students learning and reveals a flexible and easy-to-use tool, giving confidence and useful skills for success in software development. This thesis acknowledges the necessity for practical knowledge and presents carefully designed assignments to close the knowledge gap between theory and practice, equipping students with useful skills for the workplace.

1.1 Motivation

Ever wondered why software testing is such a big deal? Picture this you are all hyped up about a new app or software, ready to explore its cool features. But, what if it's glitchy, crashes, or just does not work as anticipated? That is where the motivation for diving into the world of software testing kicks in. We want our software to be tremendous, dependable, and commodity users can count on. This disquisition is each about understanding why testing is pivotal to ensure our software shines and meets the high norms that users anticipate.

1.2 Overview

Now, let's take a touring through the basics of software testing. Imagine it as a developer that safeguards our software from bugs. It's this methodical process where we check every bug and error of our software to make sure it does what it's supposed to do. We will dig into why testing is not just a one-time thing but a nonstop circle that is integrated into the whole software creation process. From the early stages of figuring out what the software should do to the final release, testing is there, making sure everything works seamlessly. (IBM, n.d.)

1.3 Why This is Done

So, why do we go through all the hassle of testing? Simple. We want top- notch software that people can calculate on. Testing is not just about fixing bugs, it's about creating software that is flexible and meets the prospects of end users. It's like having a secure friend who twice- checks everything before you make your grand entrance. By doing this, we ensure our software stands the test of time, conforming to new challenges and keeping up with the ever- changing tech geography. In a nutshell, this disquisition is each about understanding the why behind testing and how it makes our software shine. (IBM, n.d.)

2 Background

2.1 Test Automation

Test automation is a technique that enables to conduct software testing using different automatic software testing mechanism to perform test cases. Testing is essential for any exertion to succeed; this applies to software as well. It not only covers verification but also the evidence of the software. Generally testing practices include unit and functional testing that validates the demanded functionality. Manual Testing is done carefully by software developers, until this day by executing steps by steps in every tests. (Hamilton. T, 2024)

In addition, manual efforts by testers or stakeholders are demanded to verify that the software being tested factory as asked, as part of the software acceptance process. (Sambamurthy.M, n.d.)

The system can also upload programs under its tests. It can also compare forecasted and real test outcomes and produce full report for the end-user. A large amount of resources and costs are needed for software automation testing. (Saeed.M, 2024)

In the test automation development cycle, similar test suits have to be executed multiple times. In addition, using the same automation mechanism, the developer can record and replay the outcome as needed as possible. After the tests are automated there is no need of any human interaction. Doing this way enhance the test automation return of investment (ROI). Finally, the main purpose of automated test is to minimize the amount of manually executed test cases without entirely eliminating human intervention. (Arpitha, 2023)

2.2 Automated Testing and It's merits

The most crucial step in application development cycle is software testing. So far there are two important methods of testing. Both are: automated and manual testing. In manual testing the process of the testing is performed entirely by human. On the other hand, automated testing is done by high performing machines and relies on

test scripts which are created by software testers and automated systems. Besides this, the goal of test automation is not to eliminate the ordinary testing method. Different companies struggle to locate the comprehensive and decisive testing solution that fits their needs and all of their requirements for the development. To address and get rid of this issue, a hybrid solution involving multiple testing systems is necessary. (Automation A, 2024)

Furthermore, here are some of the benefits of software automation testing:

- Save resources
- Quick response for every actions
- Address many sites concurrently
- Scripts can be reusable
- The testing covers a broad range
- The tests can be executed in flexible way
- Conducting the testing manually Increase quality
- Simple testing driven by data
- Improves the accuracy of the work
- Useful for the continuity of the tests
- Enhance the working principles of the group
- Makes reporting and tracking simple
- Capable of executing extended test scenarios
  (Kanai S, 2022)

2.3 Limitations of Test Automation

There is never enough test automation, which is an ongoing challenge for test engineers. Test engineers are always under time limitations to complete manual or automated tests and ship the finished feature. Just with manual testing, test engineers encounter a range of issues on a regular basis. Several of them are listed below (Ghodke, P. 2024):

- Insufficient test automation is a common challenge.
- Lack of preparation can lead to repetition and tiredness during the automation process.

- A lack of collaboration between developers and test engineers while planning and developing automated tests may result in low-quality scripts or complex frameworks. This affects the build pipeline's quality.
- The lack of experienced test engineers has a negative impact on quality.
- Failure to synchronize testing and development processes may result in delivery delays.
- fail to comprehend the requirements and assume the expected behavior rather than confirming it with the product.
- The test automation framework is not scalable or portable (ExecutiveAutomats, n.d.)

3 Robot Framework

Robot Framework (RF) is a general-purpose free software that can be used to automate different kinds of testing like Robotics Progress Automation (RPA), acceptance test driven development (ATDD), and acceptance testing. In addition, RF is a framework used for acceptance testing, determining whether an application meets end user approval standards. This offers a test data format which is simple and tabular, and employs a test approach that is keyword-driven. Test libraries written in Python or Java can be used to improve its testing capabilities. (Kumar P, 2023)

Robot framework was created in 2005 as mentioned in the master thesis of Pekka Klärck. His thesis was about developing a framework to conduct test automation that is keyword and data-driven. Hence, the idea of RF was first comes from the affirmation person. Nevertheless, RF as an application was created as a first version in 2005 as mentioned in Nokia Network's research. Even though, version 1.0 was released in 2005, RF was not recognized as an open-source software until June 24, 2008. After the release of version 2.0 in 2008, subsequent versions like 3.0.2 was released as an open-source in a year 2017. The most recent and stable version of RF was released on 11 January 2024. Therefore, currently the RF has a large community of developers and used in most known programming languages like Python. Hence, we can find Python from www.robotframework.org, which is the official website. (Robot Framework, n.d.)

The RF is a framework that improves the readability of an organization test suits. Key word driven automation serves as a programming methods or functions, allowing users to easily develop their own key-words using accessible libraries. In addition, this key-words includes ready-made, logging and reporting mechanism. The test report will show whether the test cases executed correctly or has an error. The test report mirrors the test case and application, and is used for troubleshooting. (Wikipedia contributors, 2024)

In addition, RF has many test libraries and other tools that you can use. The Selenium is probably the most used external test library, but the robot framework also can test File Transfer Protocol (FTP), MongoDB android and more. It has a lot of APIs to help make it as extensible as possible.

There are lots of editor options when creating robot tests. The best known is probably RIDE which is a standalone robot framework editor, designed solely for editing robot framework test data. Also, there are plugins for PyCharm, Eclipse, and for many programmers there are editors like Visual Studio Code, Sublime Text and so on.

When the RF starts, it processes the test data. The framework interacts using test libraries with the planned application, eliminating the requirement for prior knowledge of it. Drivers for the libraries are either UI or different testing tools.

Generally, RF is the most important software to automate testing flexibly so that to provide end users, simple test mechanism. The RF’s architecture is modular with ready-made libraries to enable software developers to test different system operations.

The following figure shows how the real framework depends on the produced output and subsequently uses it through of several libraries. They can use different systems and interfaces to instruct the application being tested to do various methods. (Robot Framework User Guide, n.d.)

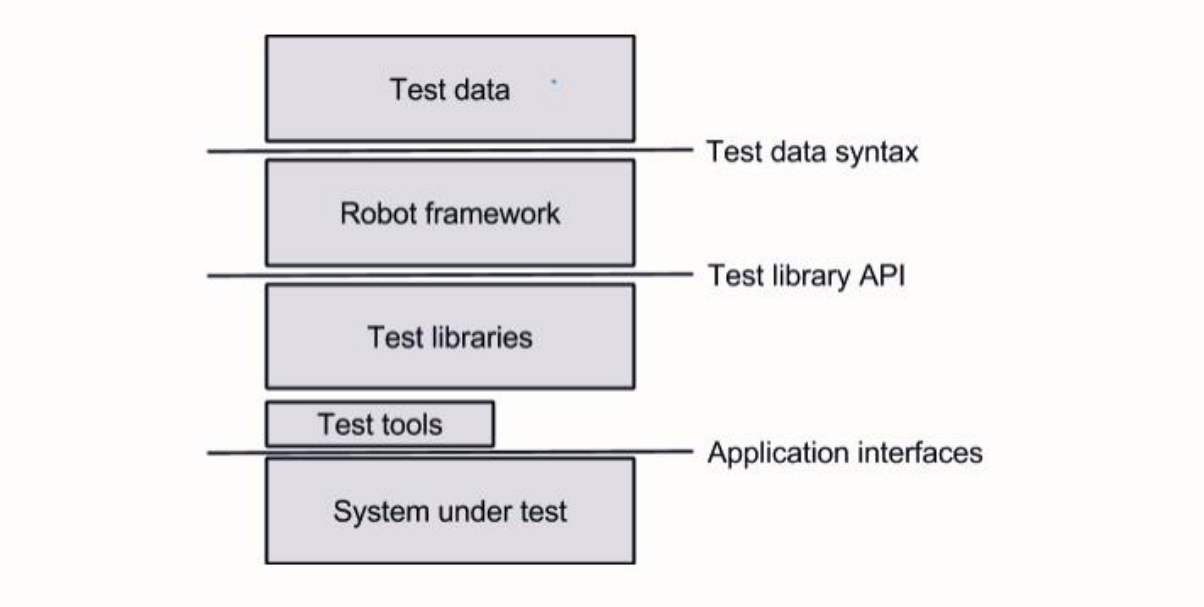

Figure 1. Robot framework high level architecture.

3.1 Relevance of Robot framework in Software Automation

In the evolving field of software automation, the RF is a versatile and essential tool. Its usefulness stems from its capacity to speed and simplify the automation process, as well as providing a user-friendly, readable syntax that transcends traditional programming boundaries. RF's keyword-driven testing strategy improves test case readability and maintainability, making it accessible to individuals with diverse situations requiring specialized determination. The frame's support for both test automation and acceptance testing reinforce its rigidity across scripts. In substance, the RF's importance in software automation stems from its ability to empower users and stimulate collaboration and make the development and testing cycles in a rapidly evolving technological field. (ChatGPT, n.d.)

3.2 Tests Supported by Robot Framework

RF is used in software development to automate tests. In most projects which demand nonstop integration as well as fast distribution time in order to help a variety of test methods. By doing so, RF can be connected with git and Jenkins, which are the popular version control tools.

Here are some of the most common cases for RF (Kumar P, 2023):

- Functional Testing**:** We can use programming languages like Python to conduct this testing to assure whether a software system is functional. Functional testing can be used to test a system as individual or as whole.
- Regression Testing: RF is one of the most known software to conduct regression testing due to it's modular nature as well as capability to fluently exercises different tests. Moreover, regression testing is used to check whether any changes or updates done on a software does not create unpredicted bugs.
- Acceptance Testing: RF automation is frequently can be used for different kinds of testing, like acceptance testing to make sure whether certain criteria of a software is met. Since, software development lifecycle involves different stages from prototyping to the final release, acceptance testing can also be done in any of these stages. (Hamilton T, 2024)
- Integration Testing**:** software developers can use integration testing to check how application system works as one unit. Devos can use RF for the automation of integration testing which in turn allows them simply to check the relations between different factors.
- Continuous Integration/ Delivery (CI/CD): The integration of RF with CI/CD channels test automation in any processes of application software development. Moreover, CI/CD permits to handle any kind of software related issues in advance to protect the occurrence of new bugs on the application development lifecycle.
- API Testing: since RF has many libraries, doing API testing is possible. Furthermore, this built-in library makes RF developer's the best choice for conducting web services testing like Simple Object Access Protocol (SOAP) and Rest API's.

All the above mentioned RF cases focuses on the quality of different parts of the application development. Even though, all of them exists in RF, their importance varies among different projects. However, developers ought to be consider all of these when they plan to test any application system. To sum-up, the Quality Assurance (QA) team must use the above-mentioned test.

3.3 Installing and configuring Robot Framework with Selenium

Because the RF lacks the tools required to create and perform automation tests for web applications, testers rely on the Selenium2library. This Selenium-based library enables the RF to run Selenium and carry out web-based operations inside. It can be used to simulate a wide range of user actions, from browsing a website to executing all UI functions.

3.4 Writing a Test Case using the Robot Framework - Selenium2library

If Python is already installed, installing the RF is simple. Simply execute the following command.

- pip install robotframework

The following command install Selenium and the Selenium2library:

- pip install selenium robotframework-selenium2library webdrivermanager

To check if it is installed successfully, run the command below:

- robot –version

If the installation was successful, one will see the framework version, as seen in the figure below.

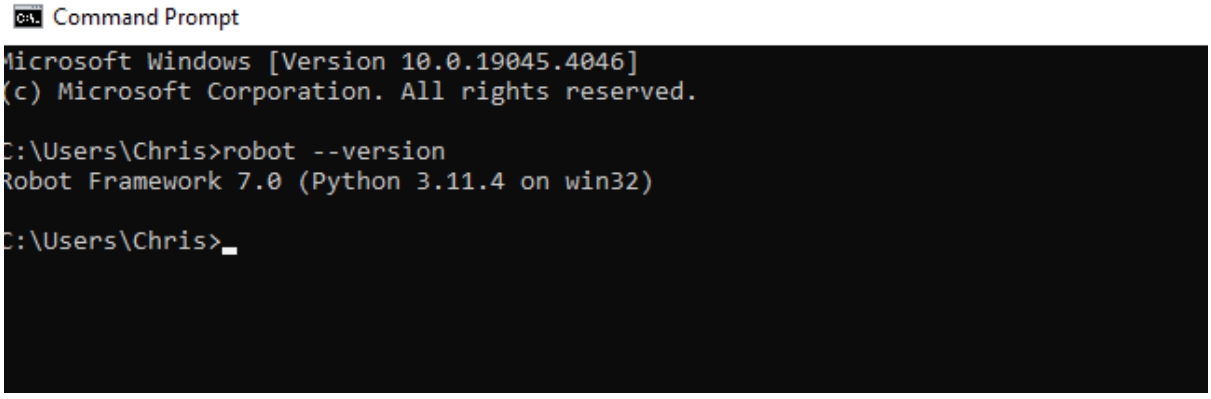

Figure 2. Robot Framework version.

4 Summary Exercise

4.1 Summary Exercise Development

A regular strategy was used to ensure efficacy and applicability when developing summary exercises acclimated for the RF as part of the software automation course. The mechanism includes a series of procedures to induce structured and practical exercises:

4.1.1 Identification of learning objectives:

- By easily expressing the exact learning objectives that each activity sought to achieve.

4.1.2 Selection of Key Topics:

- Precisely named essential principles that are closely associated with the RF and easily integrated into the whole course program.

4.1.3 Design of the Exercise Structure**:**

- Creating an organized approach for each exercise that includes an easily defined issue statement, step-by-step tasks, and anticipated outcomes.

4.1.4 Development of Supporting Materials**:**

- Preparing thorough supporting resources, like code snippets, guidelines, and references, to insure a smooth learning experience.

4.1.5 Incorporation of real-world scenarios**:**

- Each activity is introduced with real-world events to encourage practical application and develop learners' problem-solving capacities.

A set of criteria was decisively used during the concept selection process:

4.1.6 Applicability to Learning objectives:

- The concepts were chosen for their direct contribution to meeting the learning objectives.

4.1.7 Alignment with Course Curriculum:

- Concepts are thoroughly linked into the overall software automation course, providing a harmonious learning experience.

4.1.8 Progressive Complexity:

- The concepts were chosen step-by-step to increase its complexity, by enhancing participants' different capability situations.

4.1.9 Feedback Integration:

- This enabled the flawless cooperation of participant feedback, resulting in an iterative and responsive learning environment.

The following conditions are done to demonstrate how these ideas can be applied in the environment of the RF:

4.1.10 Designing Test Cases Using Robot Framework:

- The thing was to help testers understand how to use RF syntax to develop effective test cases.

4.1.11 Integration with Selenium for Web Testing:

- To widen developers' skill sets, I demonstrated how to link RF with Selenium, a prominent automated web testing tool.

4.1.12 Data-driven Testing Management:

- Developers were able to put data-driven testing methodologies into reality, answering a demand that's regularly encountered in real automation enterprises.

This regular and criteria-driven methodology guarantees that the generated exercises not only align with the RF, but also effectively contribute to the overall points of the software automation course, resulting in a well-rounded and practical learning experience.

4.2 Importance of the Summary exercises

The importance of the summary exercises in software testing automation using robot framework courses is deep, as they serve as the foundation for effective learning and skill development in the field. These activities serve an important role in molding students' educational experiences in programming-related areas. Several significant features emphasize the need of such exercises are beneficial because (Kanai, S. 2022):

- provide students with hands-on experience.
- reinforce and refine their coding skills.
- enhance logical thinking and problem-solving skills.
- increase students debugging skill in the aspect of software development.
- Enhance students deeper understanding of programming concepts
- Build collaborative skills

5 Implementation

The integration of the set summary exercises into the software automation course was a well planned and carried out process. To begin, each exercise was strategically placed across the program to correspond with crucial theoretical motifs. This assured that developers could quickly apply their recently learned information in a practical environment. The conditioning was strictly drafted to fit with specific literacy objects and cover a wide range of RF capabilities, including test case design and keyword-driven testing.

The advisor of this thesis played the main role in easing the integration. He guided and supported the students during the completion of the exercises, ensuring that testers could navigate challenges effectively. The hands-on nature of the exercises not only reinforced theoretical concepts of the learning process but also allowed students to gain practical knowledge pivotal for real-world software automation course. Regular feedback sessions were incorporated into the integration process, creating a dynamic circle for improvement based on the participants experience and difficulties encountered.

The end result was a smoothly integrated learning experience that included theory and practice. We created an environment that encouraged ongoing participation and skill development by incorporating summary tasks throughout the course. This integrated approach not only improved the overall efficacy of the software automation course, but it also provided students with a comprehensive understanding of the RF.

6 Evaluation

In assessing the effectiveness of the summary exercises drafted for the RF software testing automation course in this thesis, the author established specific criteria to gauge their impact on testers' learning experience. These criteria were designed to assess how well the exercises aligned with the intended learning objectives, their capability to encourage practical application of RF ideas, and the nonstop integration of feedback from both developers and advisor. Also considered the adaptability of the exercises to accommodate individualities with different skill situations and backgrounds, aiming to produce a safe learning environment. Through a thorough analysis of results against these criteria, gained precious perceptivity into the strengths of the exercises and linked areas for enhancement, eventually contributing to the ongoing refinement of the RF training approach.

6.1 Test cases and Test results

The author explains how to perform test cases in the robot framework and read the results. The RF provides numerous techniques for testing case execution.

The following figures show the results of login tests performed with the RF. The tests were divided into different files these are: 'Gherkin Login', 'Invalid Login', and 'Valid Login'. In the 'Gherkin Login' file, a valid login test was successfully completed. The other one is an ' Invalid Login' file including cases similar as invalid username/password and empty fields, which all the test cases passed accordingly. Also, the' valid Login' file tested for a valid login and passed. In total, there are eight tests were run, and all of them passed.

The output, log and report files were created and these files provided full information about the execution of these login tests within the RF project. This comprehensive set of tests included a lot of different scenarios to ensure the login functionality. All the findings from the test cases show that the tests were successful and reliable. This confirms that the RF is very useful in validating login functionality.

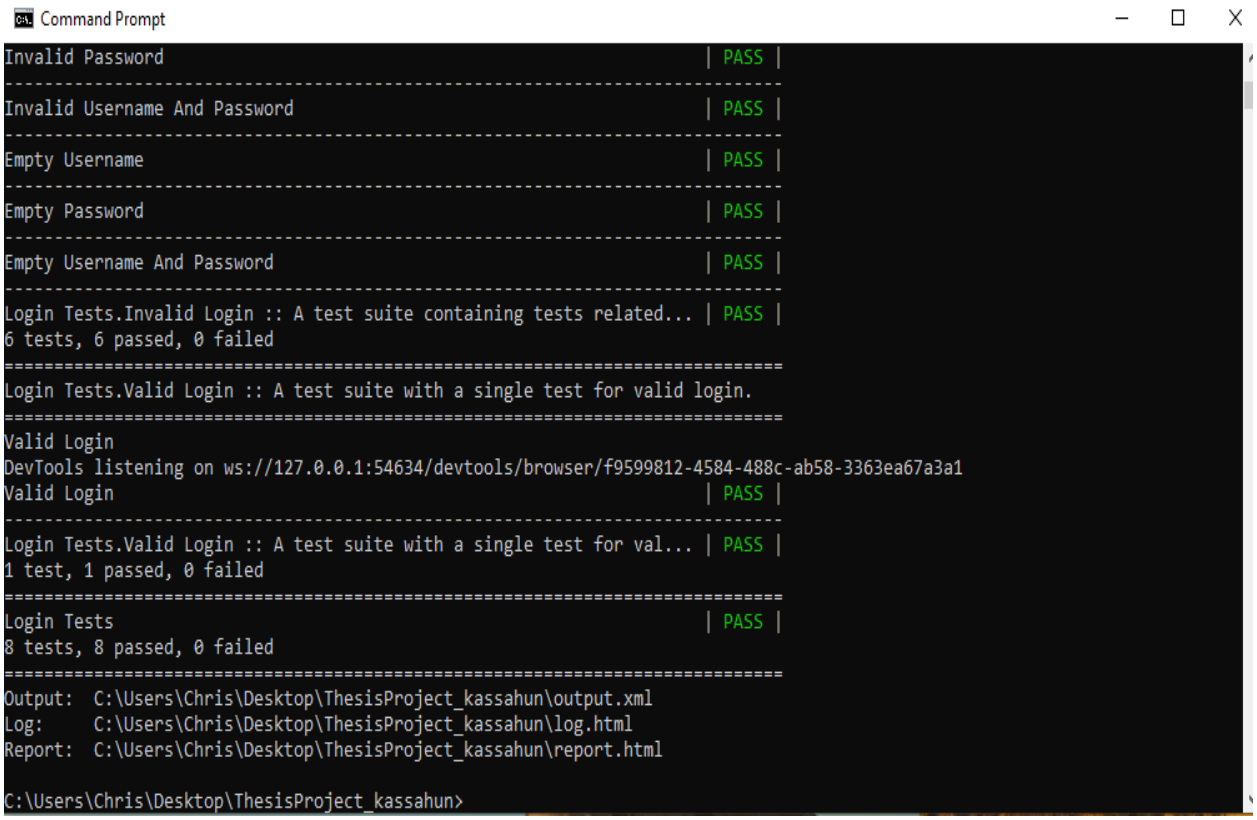

Figure 3. Result of test case execution.

The next ‘report.html file screenshot shows a detailed overview of the login tests executed using the RF. It provides the summary of the passed test and failed tests, the execution time taken, and the logs of each test case.

Figure 4 shows summary result of test case execution. The test cases shown in the figure below explains details about test statistics in a summarized way. Furthermore, the summarized result of the above information depicts like Suits, search and Tags. As shown in the affirmation figure the tab options are used to analyse the research. The tab options contain test cases like names, tags, status and documentations. Moreover, it can be used to examine, message elapsed, and details about the start and end of the test’s cases. It is the duty of the tester to use all other tab options based on the test requirements. The tab name can be found in the top right corner of the figure. The tester can use the tab name to update the reports page as per their needs. Hence, the purpose of the report is to track the operation of the test cases, so as to find the test case results. Failure of the test case due to some reason the developer receives a report, and use that to fix the code. By doing so the updated

version of the code can be forwarded to the testers. This process continues repeatedly.

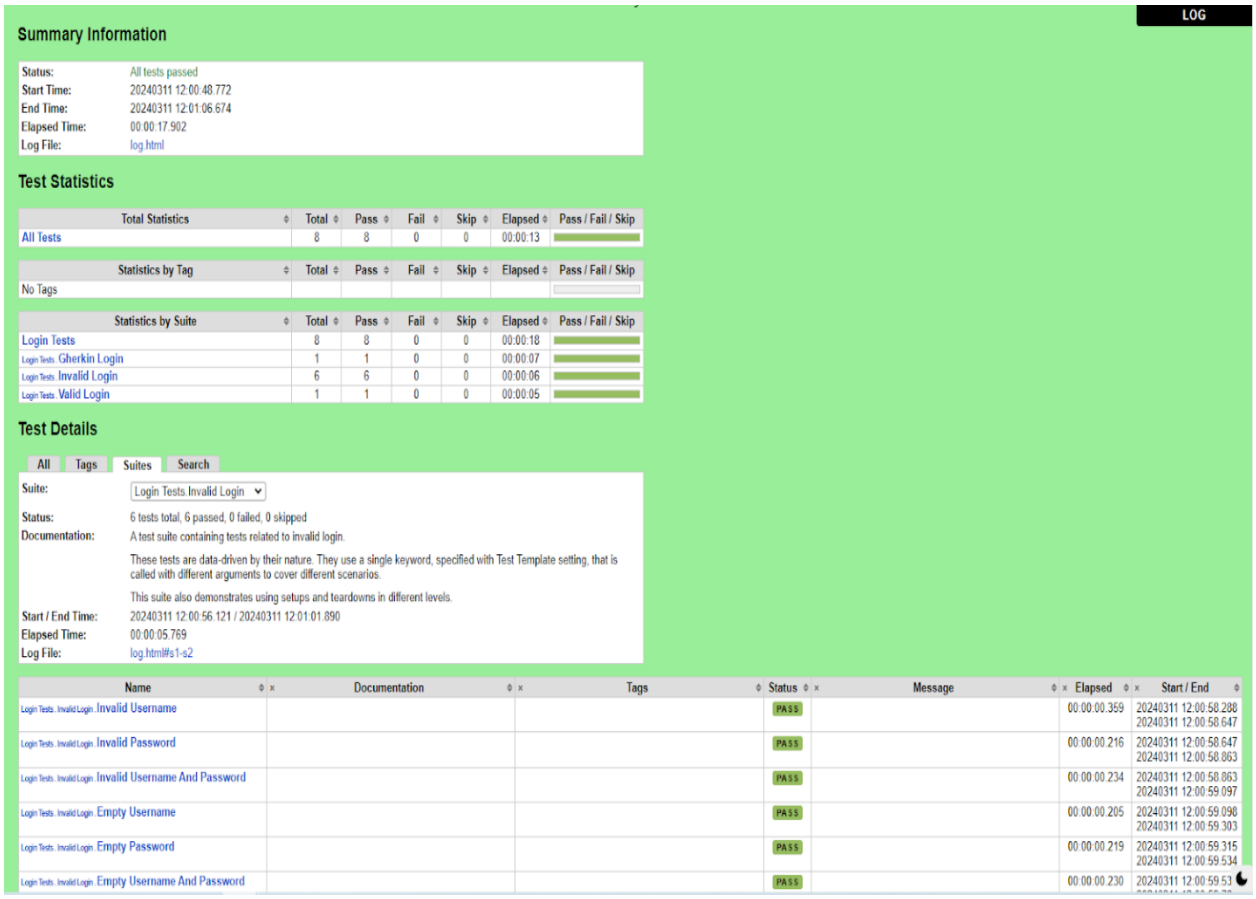

Figure 4. Analysing test results with external browser.

The following screenshot clearly shows that the Preview of "log.html" file which opens in a different tab, including the entire test case report. This .html report file can be expanded according to the tester's needs where one wants to see. Users can view all run test cases, including the overall number of passed and failed test cases. The "+" option in each test case report allows testers to view the entire procedure, even if the test fails. This report includes the test case's full name, start/end/elapsed, and status, followed by keywords with documentation and start/end/elapsed information.

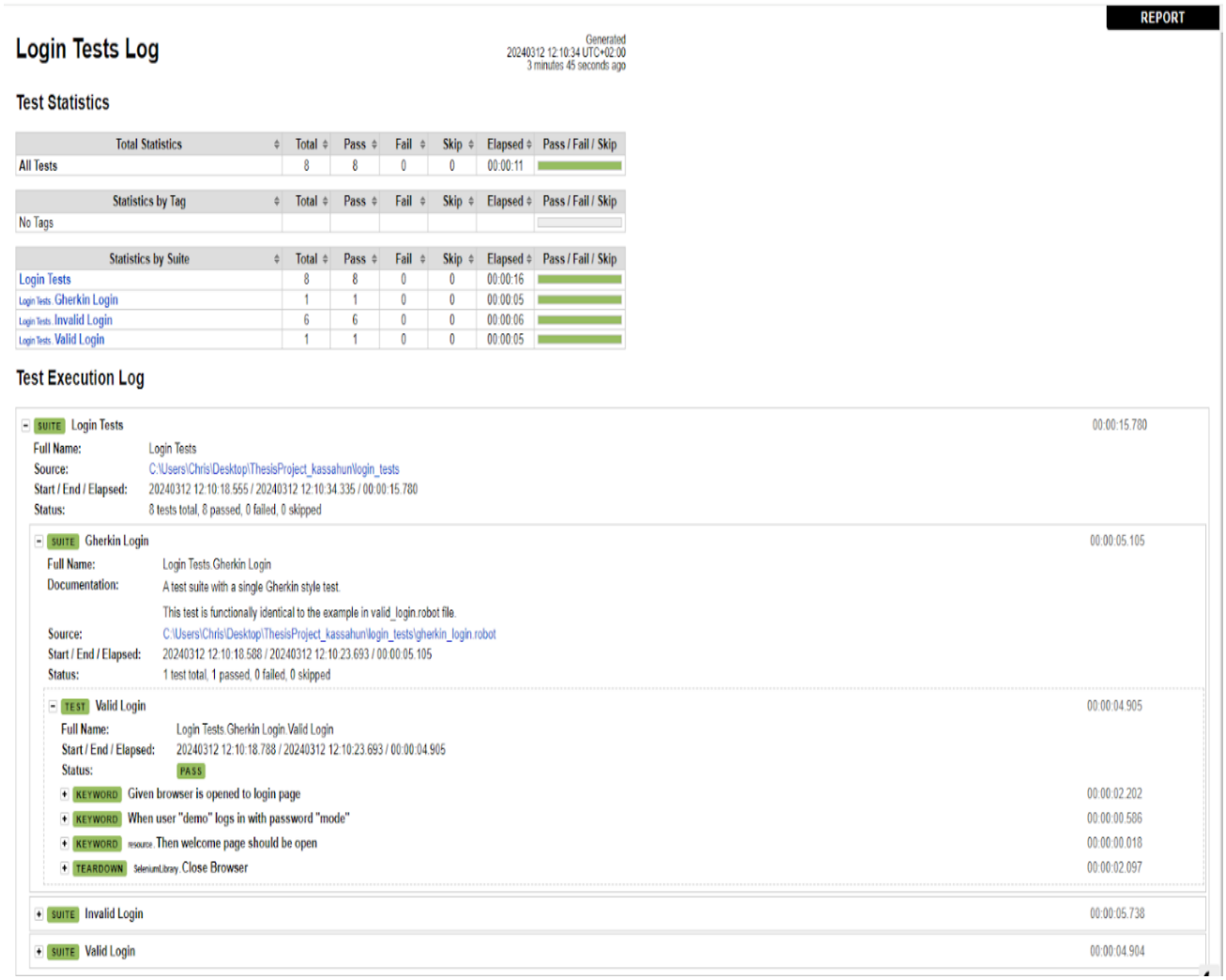

Figure 5. Analysing test results with built-in preview option.

resource - Notepad

File Edit Format View Help

```
*** Settings ***
Documentation     A resource file with reusable keywords and variables.
...
...               The system specific keywords created here form our own
...               domain specific language. They utilize keywords provided
...               by the imported SeleniumLibrary.
Library           SeleniumLibrary

*** Variables ***
${SERVER}         localhost:7272
${BROWSER}        Chrome
${DELAY}          0
${VALID USER}     demo
${VALID PASSWORD}    mode
${LOGIN URL}      http://${SERVER}/
${WELCOME URL}    http://${SERVER}/welcome.html
${ERROR URL}      http://${SERVER}/error.html

*** Keywords ***
Open Browser To Login Page
    Open Browser    ${LOGIN URL}    ${BROWSER}
    Maximize Browser Window
    Set Selenium Speed    ${DELAY}
    Login Page Should Be Open

Login Page Should Be Open
    Title Should Be    Login Page

Go To Login Page
    Go To    ${LOGIN URL}
    Login Page Should Be Open

Input Username
    [Arguments]    ${username}
    Input Text    username_field    ${username}

Input Password
    [Arguments]    ${password}
    Input Text    password_field    ${password}

Submit Credentials
    Click Button    login_button

Welcome Page Should Be Open
    Location Should Be    ${WELCOME URL}
    Title Should Be    Welcome Page
```

Figure 6. Test Workflow with Reusable Keywords and Variables.

The above screenshot shows a test workflow using a resource file. This resource file contains reusable keywords like Open Browser and variables like ${LOGIN URL} and ${BROWSER}. They make use of the keywords offered by the imported SeleniumLibrary.

6.2 Discussion

6.2.1 Findings Interpretation:

Findings All eight login tests in the RF passed successfully, indicating significant success. The regular split of tests into' Gherkin Login,'' Invalid Login,' and' Valid Login' suites enabled a thorough review of multitudinous scenarios. especially, the addition of real-world situations like invalid username/password and empty fields demonstrated the RF's capacity to efficiently handle a wide range of testing scenarios. The getting of detailed output, log, and report files gave a clear perspective of the test execution process, allowing for a complete evaluation of the findings.

6.2.2 Implications for Software Testing Automation Course:

As it is indicated on the test, the positive login test results show how effective RF is in software automation testing. In addition, this approach for developing summary exercises for software testing automation using RF course has proved that using RF provides practical experience in handling authentication scenarios. The results suggest that incorporating the summary exercises into the course can enhance software testers' skills in automated testing, this emphasizes the importance of practical and real-world applications in software testing automation courses.

6.2.3 Comparison with Existing Methods:

When we compare our approach with existing methods in software Testing automation courses, our approach differs itself by focusing on real-world scenarios and the systematic progression in exercise complexity stand out as distinctive features. Furthermore, unlike some previously used techniques that may lack practical application, on this approach using RF for summary exercises provides a hands-on learning experience. The ability of RF to seamlessly integrate with different standard tools further positions it as a versatile choice for software testing automation courses. Our technique has several advantages, this includes its adaptation to different skill levels and the continuous feedback loop incorporated into the exercise development process.

7 Conclusion

In summary, this thesis has demonstrated the effectiveness of RF for software testing automation course through the development of structured summary exercises. The evaluation of login tests within the RF yielded positive results, with all the eight tests passed successfully on the given scenarios. This strategic preparation of the tests into suits, and with the generation of the required output, makes the execution process valuable.

The contribution of this thesis for the software testing automation course with RF is immense. Firstly, it emphasizes the use of RF in practical learning environments, that focus on real-world scenarios to help testers improve their skills. Furthermore, this structured approach to the exercise's development provides progression in the complexity by giving developers the required skills. In addition, the use of continual feedback methods ensures iterative improvement to the course.

In conclusion, the author emphasizes the importance of RF-based exercises in software testing automation course as indicating a practical and effective approach for improving software testers automated testing skills. Furthermore, this has been done by encouraging hands-on experience, integrating with the industry norms, and by embracing continuous improvements. To do all these things RF is the most valuable tool for software testing automation environments.

Reference

Arpitha. (2023, July 3). Types of automation Testing - Arpitha - medium. *Medium*. https://arkaaq.medium.com/types-of-automation-testing-fc92917d2667

Automation, A. (2024, February 8). 10 Benefits of test automation. Avo Automation. Retrieved February 10, 2024, from https://avoautomation.ai/10-benefits-of-test-automation/

Bisht, S. (n.d.). Robot Framework Test automation. O'Reilly Online Learning. Retrieved February 17, 2024, from https://learning.oreilly.com/library/view/robot-frameworktest/9781783283033/

ChatGPT. (n.d.). Retrieved February 24, 2024, from https://chat.openai.com/c/c23ffefc-54eb-40a9-ae37-6d577643249e

ExecutiveAutomats. (n.d.). 10 Challenges in automation testing. Retrieved March 9, 2024, from https://www.executiveautomats.com/resources/articles/10-challenges-in-automation-testing

Ghodke, P. (2024, January 9). Automation Testing Limitations: Understanding the boundaries in quality assurance. Retrieved March 19, 2024, from https://www.linkedin.com/pulse/automation-testing-limitations-understanding-quality-assurance-tq3ff/

GfG. (2024, March 26). Automation testing software testing. GeeksforGeeks. Retrieved April 6, 2024, from https://www.geeksforgeeks.org/automation-testing-software-testing/

Hamilton, T. (2024, March 9). Functional vs Non-Functional testing – difference between them. Guru99. Retrieved March 22, 2024, from https://www.guru99.com/functional-testing-vs-non-functional-testing.html

IBM. (n.d.). What is software testing? Retrieved March 4, 2024, from https://www.ibm.com/topics/software-testing

IBM. (n.d.). Why software testing is important. Retrieved April 3, 2024, from https://www.ibm.com/topics/software-testing

Irfan, K. (2022, October 10). Robot Framework — Test Automation the smart Way!- Emumba. Medium. Retrieved March 25, 2024, from

https://blog.emumba.com/robot-framework-test-automation-the-smart-way-4c81bfdd0a9

Kumar, P. (2023, June 9). Introduction to robot Framework | BrowserStack. BrowserStack. Retrieved February 28, 2024, from https://www.browserstack.com/guide/robot-framework-guide

Kanai, S. (2022, August 16). 15 Benefits of Automated testing in app development. Retrieved March 15, 2024, from https://www.headspin.io/blog/15-benefits-of-automated-testing-in-app-development

Saeed, M. (2024, April 17). *Automation testing: pros and cons*. Kualitatem. https://www.kualitatem.com/blog/automation-testing/pros-cons/

Sambamurthy, M. (n.d.). Test Automation Engineering Handbook. O'Reilly Online Learning. Retrieved March 7, 2024, from https://learning.oreilly.com/library/view/test-automation-engineering/9781804615492/

Robot Framework. (n.d.). Introduction. Retrieved February 11, 2024, from https://robotframework.org/

Robot Framework User Guide. (n.d.). Retrieved February 24, 2024, from https://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html

Wikipedia contributors. (2024, February 22). Robot Framework. Wikipedia. Retrieved March 27, 2024, from https://en.wikipedia.org/wiki/Robot_Framework

Writing and executing test cases. (n.d.). Retrieved April 4, 2024, from https://www.tutorialspoint.com/robot_framework/robot_framework_writing_and_executing_test_cases.htm