# GRAFT: Generic & Reusable Automation Framework for Agile Testing

Lipika Bose
Department of Computer Science,
Amity University,
Noida, India.
lipika.bose@gmail.com

Sanjeev Thakur
Department of Computer Science,
Amity University,
Noida, India.
sthakur3@amity.edu

***Abstract*— Automation testing framework enables the complete automation of the entire testing process in a software development life cycle. This research paper will mainly focus on a new hybrid automation testing framework called GRAFT – Generic & Reusable Automation Framework which supports cross browser testing.**

***Keywords*—Cross Browser testing, Data Driven, Keyword Driven, Hybrid.**

## I. INTRODUCTION

Software testing has profound place in software industry due to its crucial role in quality software production. With the increase in the business requirements the need to develop a higher number of quality product in a short interval of time and with limited resources becomes a constraint[1]. This scenario leads to the identification of such a tool which is best suited for the application to be developed and automation of the testing process with a cost effective approach[1].

Designing process of an automated test framework is not a science. The essential element in automation history is to assure a thought out approach on the basis of common and successfully implemented software industrial practices.

Software testing include special software in test automation process. This software controls the execution of test and the comparison of actual and predicted outcomes. Test automation regulates some repetitive but important task through structured testing process that would be tedious for manual execution.
[2].

## II. PROBLEM STATEMENT

Cross browser testing[3] helps to ensure uniform web application behavior across various web browsers. Lack of any well-defined automation framework to test various application functional/non-functional scenarios lead to huge amount of manual effort of testing the same functional scenarios in multiple browsers. In agile development where sprint durations are just of 4-6 weeks, running regression suit manually across multiple browsers require huge manual effort.
This paper describes a Generic & Reusable Automation Framework for cross browser testing by using selenium. The selenium helps in implementing the reporting features and in cross browser testing.

## III. BACKGROUND

The approach to design an automation framework includes the following procedure:

- Determination of a Framework Type
- Identifying Framework Components
- Identifying Framework Directory Structure
- Developing Implementation Standards
- Develop Automated Tests

### *A. Different Ways to Design A Framework*

1. Test Script Driven Modularity Framework :
    - This framework builds a layer of abstraction at the fore of a component to cloud it from the remaining software application.
    - This is done by writing small unconventional scripts.
    - Each script represents the different modules and various methods of the AUT.
    - Small scripts can be combined to from larger tests
    - Results in high degree of modularization and maintainability.

2. Data-Driven Automation Framework[5] :
    - Input and the output values of a test scenario are read from data files.
    - These values loaded into corresponding variables in captured scripts.
    - Test flow navigation coded into test script.
    - Thus script can be defined as "driver," or consignment operation, for the data.

3. Keyword Driven Automation Framework[1] :
    - This framework includes the development of keywords and the data depending on the application to be tested.
    - The keywords and the data sets are independent of the automation testing tool.

- In this framework a manual test flow is represented with a sequence of keywords to be implemented.
- The functionality of the application to be tested is logged in a tabular notation along with the specification of each test.

4. Hybrid Test Automation Framework[7]:
    - It is the amalgamation of keyword driven, test script and data driven automation frameworks. This framework focuses to evacuate the supremacy of the above mentioned frameworks.
    - Hybrid framework can be designed as per your or the stakeholders requirement.
    - This framework uses scripts to perform certain might be difficult to implement in a pure keyword driven approach.
    - This framework should be designed in a way that it is reusable, generic and easily scalable.

## IV. RESEARCH METHODOLOGY

This framework builds automated test scripts with underlying Automation tool Selenium. Selenium is portable software testing tool for Web applications[2]. It supports cross browser testing.

In GRAFT test cases are broken down into business keywords. Keywords are strung together in an excel sheet to form automated test case based on business flow. Therefore simplifying the process of creating automation script.

Centralized test data in excel is maintained to improve the reusability of common data in multiple test case. Therefore test data can be customized easily. Feature of error handling with automatic screen captures of failures. Thereby enables uninterrupted batch execution of the scripts .

Customized reports on excel/html can be generated .Thereby helping in analysis .

### A. *Following are the modules that make up the GRAFT:*

- Business Keywords
- Business Components
- Component Groups
- Business Flow
- Test Data
- Support Libraries
- Test Scripts
- Allocator & Run Manager
- Customized Test Results
- HTML &Excel

### B. *Architecture:*

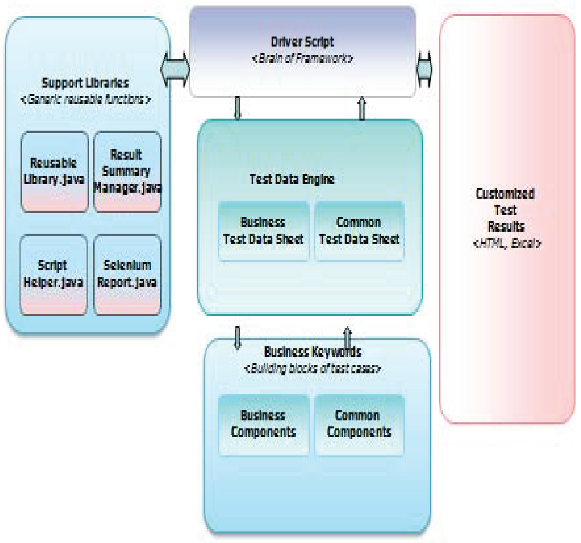


Fig1. Architecture of GRAFT

***Business Keywords:*** All the logic of the keyword components will be implemented in the business keyword. Whenever new application needs to be tested new modules specific to the application will be added in the Business component.

***Driver Script***: It is the core of the framework. It is point from where the framework starts. It can be called the “entry point”.It contains the core logic of the framework. It executes the given test case. All the operations like startup(), Initializing of test iterations, Initialization of web drivers, report initialization, Execution of test iteration.

***Customized Test Result:*** Test Results are generated in the form of EXCEL and HTML.

***Test Data Engine:*** Business Test Data Sheet: It contains the list of test case ids along with their associated keywords. It also contains the number of iterations and sub iterations associated with each of the test case id. Left to right in excel

***Common Test data sheet:*** It contains the common test data that can be repeatedly used in the application.

***Support Libraries***: These contain generic reusable functions which are required for testing the application. This component will also contain the logic for report generation. Few of the sample classes in this component are as follows:

- *ReusableLibrary.java :* It is the base class which contains the reusable libraries created by the user. Eg. Selenium, data table.

- *ResultSummaryManager.java*: class that manages the result summary creation during a batch execution.
- *ScriptHelper.java* : common framework objects, to be used across the entire test case and dependent libraries.
- *SeleniumReport.java:* Class to extend the reporting features of the framework.
- *Browser.java:* represent the browser to be used for execution.
- *SeleniumTestParameters.java:* Class to encapsulate various input parameters required for each test script. Browser, version, platform.
- *WebDriverFactory.java*: to get the drivers/ systems proxy settings.

***Allocator and Run Manager***: This component contains the implementation to manage the batch execution of test scripts within the framework. It also implements the parallel execution of the test scripts.

***Run Manager.xls:*** This file will contain the sequence and all the other related information like test case id, description, browser type, platform for the batch execution of the test case.

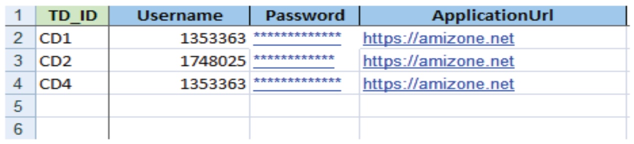
## V. Result & Screenshots

The framework was tested on an application of college portal of Amity University: **"AMIZONE ".** To prove its generic & reusability the same framework was tested for gmail and yahoo mail applications.

A complete test suite was written to cover all the scenarios. Result of simple login/ logout functionality and generation of fees receipt are illustrated below. All the test cases were executed in the following ways:

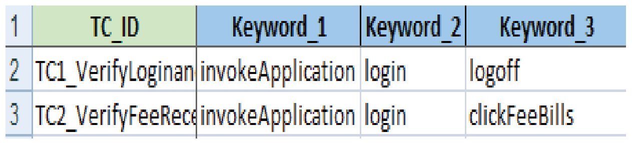
1. Three different browsers chrome, Firefox and Internet Explorer(It can be tested in other browsers also).
2. Execution of individual test scenarios with single data set.
3. Multiple execution of the same test case with different data sets.
4. Batch execution of the test cases.

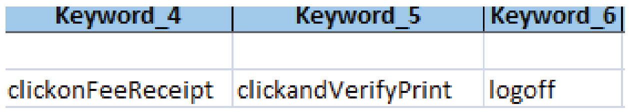
***Common Data Se for Amizone Application:***



| 1 | TD_ID | Username | Password | ApplicationUrl |
|---|---|---|---|---|
| 2 | CD1 | 1353363 | ************* | https://amizone.net |
| 3 | CD2 | 1748025 | ************ | https://amizone.net |
| 4 | CD4 | 1353363 | ************* | https://amizone.net |
| 5 | | | | |
| 6 | | | | |

***Business Flow for LoginLogout functionality and generation of fees receipt:***

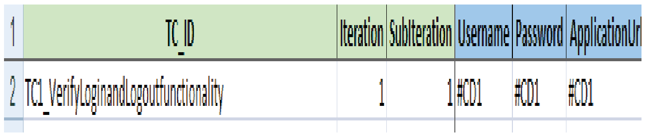




| 1 | TC_ID | Keyword_1 | Keyword_2 | Keyword_3 | Keyword_4 | Keyword_5 | Keyword_6 |
|---|---|---|---|---|---|---|---|
| 2 | TC1_VerifyLoginan | invokeApplication | login | logoff | | | |
| 3 | TC2_VerifyFeeRece | invokeApplication | login | clickFeeBills | clickonFeeReceipt | clickandVerifyPrint | logoff |

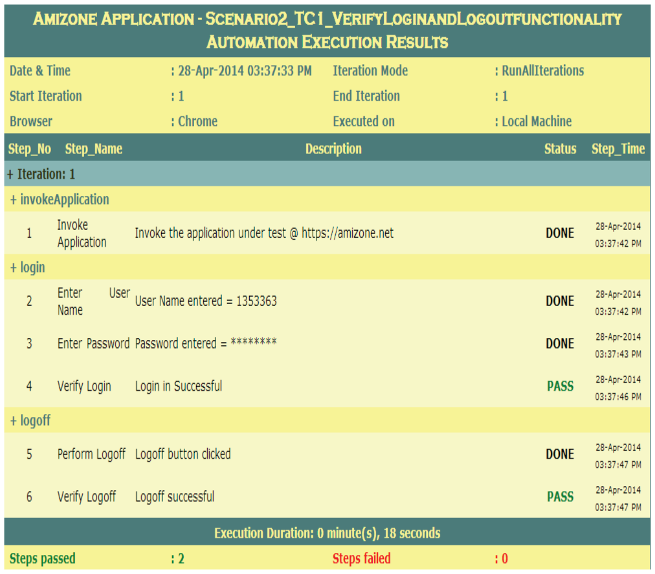
**Scenario 1 - Verify Login and Logout Functionality**

***General Data Set :***



| 1 | TC ID | Iteration | SubIteration | Username | Password | ApplicationUrl |
|---|---|---|---|---|---|---|
| 2 | TC1_VerifyLoginandLogoutfunctionality | 1 | 1 | #CD1 | #CD1 | #CD1 |

**a. In Crome:**



AMIZONE APPLICATION - SCENARIO2_TC1_VERIFYLOGINANDLOGOUTFUNCTIONALITY
AUTOMATION EXECUTION RESULTS

| | | | |
|---|---|---|---|
| Date & Time | : 28-Apr-2014 03:37:33 PM | Iteration Mode | : RunAllIterations |
| Start Iteration | : 1 | End Iteration | : 1 |
| Browser | : Chrome | Executed on | : Local Machine |

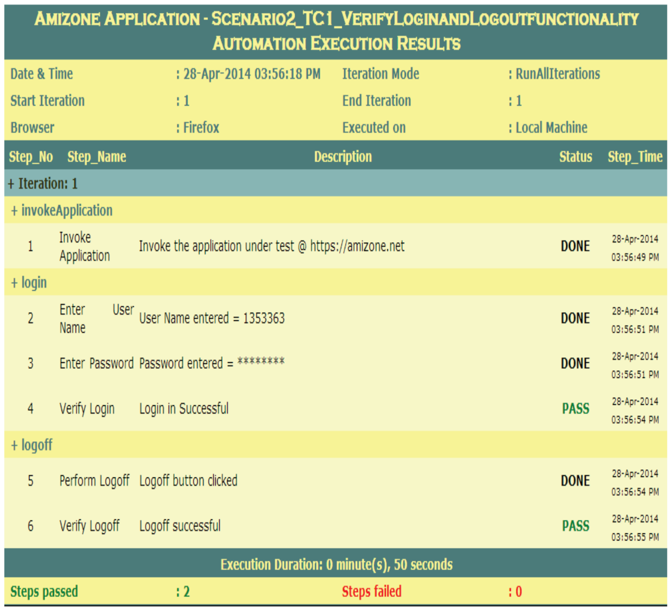
| Step_No | Step_Name | Description | Status | Step_Time |
|---|---|---|---|---|
| + Iteration: 1 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://amizone.net | DONE | 28-Apr-2014 03:37:42 PM |
| + login | | | | |
| 2 | Enter User Name | User Name entered = 1353363 | DONE | 28-Apr-2014 03:37:42 PM |
| 3 | Enter Password | Password entered = ******** | DONE | 28-Apr-2014 03:37:43 PM |
| 4 | Verify Login | Login in Successful | PASS | 28-Apr-2014 03:37:46 PM |
| + logoff | | | | |
| 5 | Perform Logoff | Logoff button clicked | DONE | 28-Apr-2014 03:37:47 PM |
| 6 | Verify Logoff | Logoff successful | PASS | 28-Apr-2014 03:37:47 PM |

Execution Duration: 0 minute(s), 18 seconds

Steps passed : 2 Steps failed : 0

Fig 2. Screenshot of the test result generated on testing the login/logout functionality in chrome.

**b. Firefox:**



AMIZONE APPLICATION - SCENARIO2_TC1_VERIFYLOGINANDLOGOUTFUNCTIONALITY
AUTOMATION EXECUTION RESULTS

| | | | |
|---|---|---|---|
| Date & Time | : 28-Apr-2014 03:56:18 PM | Iteration Mode | : RunAllIterations |
| Start Iteration | : 1 | End Iteration | : 1 |
| Browser | : Firefox | Executed on | : Local Machine |

| Step_No | Step_Name | Description | Status | Step_Time |
|---|---|---|---|---|
| + Iteration: 1 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://amizone.net | DONE | 28-Apr-2014 03:56:49 PM |
| + login | | | | |
| 2 | Enter User Name | User Name entered = 1353363 | DONE | 28-Apr-2014 03:56:51 PM |
| 3 | Enter Password | Password entered = ******** | DONE | 28-Apr-2014 03:56:51 PM |
| 4 | Verify Login | Login in Successful | PASS | 28-Apr-2014 03:56:54 PM |
| + logoff | | | | |
| 5 | Perform Logoff | Logoff button clicked | DONE | 28-Apr-2014 03:56:54 PM |
| 6 | Verify Logoff | Logoff successful | PASS | 28-Apr-2014 03:56:55 PM |

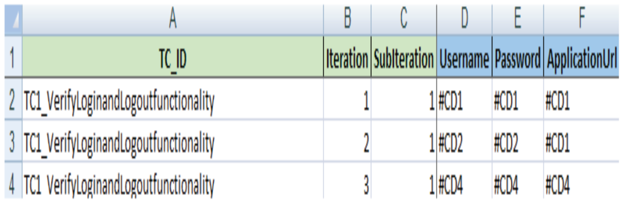
Execution Duration: 0 minute(s), 50 seconds

Steps passed : 2 Steps failed : 0

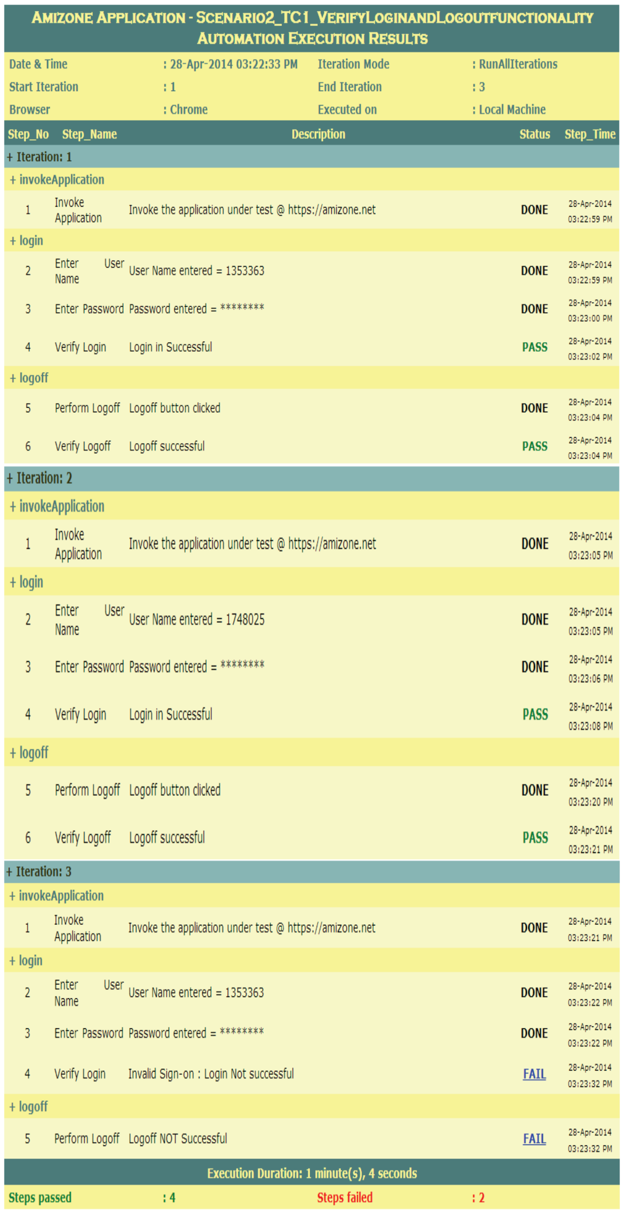
Fig 3. Screenshot of the test result generated on testing the login/logout functionality in Firefox.

**Scenario 2 – Multiple times verification of Login and Logout Functionality with different data sets.**

***General Data Set :***



| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | TC_ID | Iteration | SubIteration | Username | Password | ApplicationUrl |
| 2 | TC1_VerifyLoginandLogoutfunctionality | 1 | 1 | #CD1 | #CD1 | #CD1 |
| 3 | TC1_VerifyLoginandLogoutfunctionality | 2 | 1 | #CD2 | #CD2 | #CD1 |
| 4 | TC1 VerifyLoginandLogoutfunctionality | 3 | 1 | #CD4 | #CD4 | #CD4 |



AMIZONE APPLICATION - SCENARIO2_TC1_VERIFYLOGINANDLOGOUTFUNCTIONALITY
AUTOMATION EXECUTION RESULTS

| | | | |
|---|---|---|---|
| Date & Time | : 28-Apr-2014 03:22:33 PM | Iteration Mode | : RunAllIterations |
| Start Iteration | : 1 | End Iteration | : 3 |
| Browser | : Chrome | Executed on | : Local Machine |

| Step_No | Step_Name | Description | Status | Step_Time |
|---|---|---|---|---|
| + Iteration: 1 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://amizone.net | DONE | 28-Apr-2014 03:22:59 PM |
| + login | | | | |
| 2 | Enter User Name | User Name entered = 1353363 | DONE | 28-Apr-2014 03:22:59 PM |
| 3 | Enter Password | Password entered = ******** | DONE | 28-Apr-2014 03:23:00 PM |
| 4 | Verify Login | Login in Successful | PASS | 28-Apr-2014 03:23:02 PM |
| + logoff | | | | |
| 5 | Perform Logoff | Logoff button clicked | DONE | 28-Apr-2014 03:23:04 PM |
| 6 | Verify Logoff | Logoff successful | PASS | 28-Apr-2014 03:23:04 PM |
| + Iteration: 2 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://amizone.net | DONE | 28-Apr-2014 03:23:05 PM |
| + login | | | | |
| 2 | Enter User Name | User Name entered = 1748025 | DONE | 28-Apr-2014 03:23:05 PM |
| 3 | Enter Password | Password entered = ******** | DONE | 28-Apr-2014 03:23:06 PM |
| 4 | Verify Login | Login in Successful | PASS | 28-Apr-2014 03:23:08 PM |
| + logoff | | | | |
| 5 | Perform Logoff | Logoff button clicked | DONE | 28-Apr-2014 03:23:20 PM |
| 6 | Verify Logoff | Logoff successful | PASS | 28-Apr-2014 03:23:21 PM |
| + Iteration: 3 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://amizone.net | DONE | 28-Apr-2014 03:23:21 PM |
| + login | | | | |
| 2 | Enter User Name | User Name entered = 1353363 | DONE | 28-Apr-2014 03:23:22 PM |
| 3 | Enter Password | Password entered = ******** | DONE | 28-Apr-2014 03:23:22 PM |
| 4 | Verify Login | Invalid Sign-on : Login Not successful | FAIL | 28-Apr-2014 03:23:32 PM |
| + logoff | | | | |
| 5 | Perform Logoff | Logoff NOT Successful | FAIL | 28-Apr-2014 03:23:32 PM |

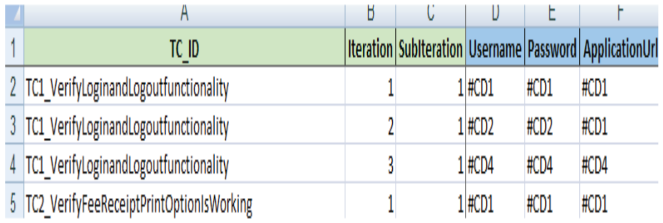
Execution Duration: 1 minute(s), 4 seconds

| | | | |
|---|---|---|---|
| Steps passed | : 4 | Steps failed | : 2 |

Fig 4. Screenshot of the test result generated on testing the login/logout functionality multiple times with different data sets in chrome.

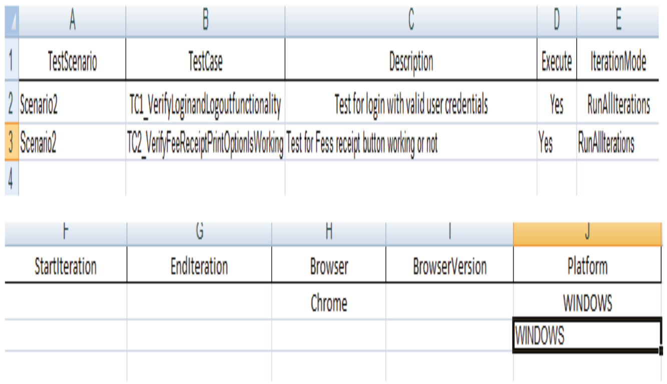
**Scenario 3 – 3.Batch execution of multiple test cases:**

***General Data Set :***



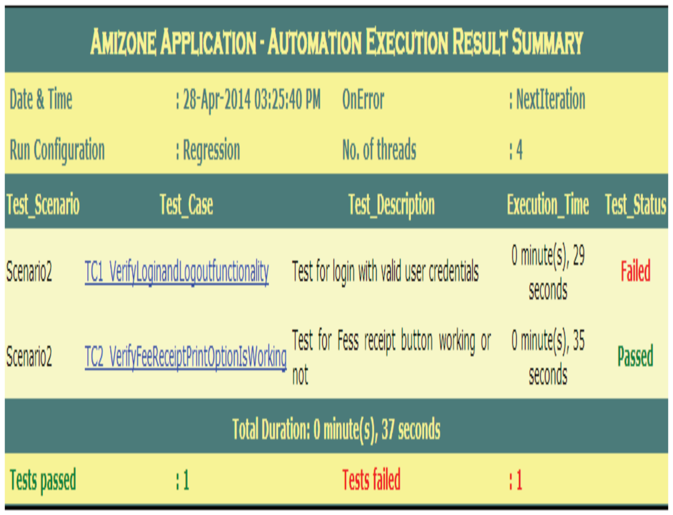
| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | TC_ID | Iteration | SubIteration | Username | Password | ApplicationUrl |
| 2 | TC1_VerifyLoginandLogoutfunctionality | 1 | 1 | #CD1 | #CD1 | #CD1 |
| 3 | TC1_VerifyLoginandLogoutfunctionality | 2 | 1 | #CD2 | #CD2 | #CD1 |
| 4 | TC1_VerifyLoginandLogoutfunctionality | 3 | 1 | #CD4 | #CD4 | #CD4 |
| 5 | TC2_VerifyFeeReceiptPrintOptionIsWorking | 1 | 1 | #CD1 | #CD1 | #CD1 |

***RunManager.xls:***



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | TestScenario | TestCase | Description | Execute | IterationMode |
| 2 | Scenario2 | TC1_VerifyLoginandLogoutfunctionality | Test for login with valid user credentials | Yes | RunAllIterations |
| 3 | Scenario2 | TC2_VerifyFeeReceiptPrintOptionIsWorking | Test for Fess receipt button working or not | Yes | RunAllIterations |
| 4 | | | | | |

| F | G | H | I | J |
|---|---|---|---|---|
| StartIteration | EndIteration | Browser | BrowserVersion | Platform |
| | | Chrome | | WINDOWS |
| | | | | WINDOWS |



AMIZONE APPLICATION - AUTOMATION EXECUTION RESULT SUMMARY

| | | | |
|---|---|---|---|
| Date & Time | : 28-Apr-2014 03:25:40 PM | OnError | : NextIteration |
| Run Configuration | : Regression | No. of threads | : 4 |

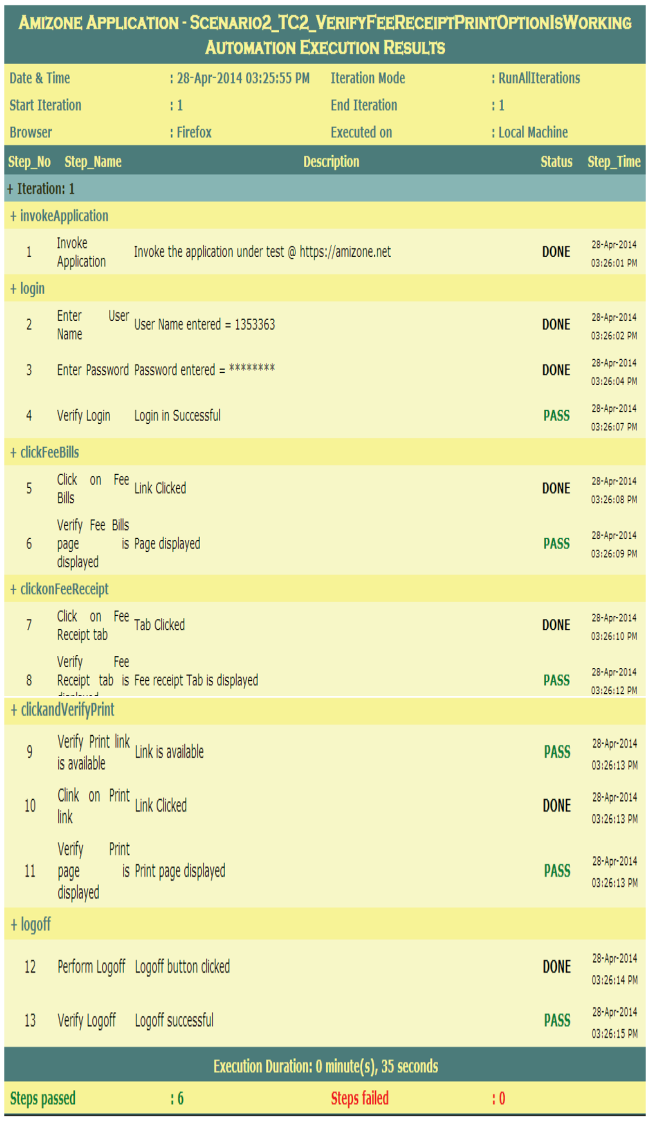
| Test_Scenario | Test_Case | Test_Description | Execution_Time | Test_Status |
|---|---|---|---|---|
| Scenario2 | TC1_VerifyLoginandLogoutfunctionality | Test for login with valid user credentials | 0 minute(s), 29 seconds | Failed |
| Scenario2 | TC2_VerifyFeeReceiptPrintOptionIsWorking | Test for Fess receipt button working or not | 0 minute(s), 35 seconds | Passed |

Total Duration: 0 minute(s), 37 seconds

| | | | |
|---|---|---|---|
| Tests passed | : 1 | Tests failed | : 1 |

Fig 5. Screenshot of the test result generated on testing the login/logout functionality and fee receipt verification in a batch.

Screenshot of TC1_VerifyLoginLogoutfunctionality is in Fig3 and the screenshot of TC2 VerifyFeeReceiptPrintOptionWorkingFine is in Fig5.



**AMIZONE APPLICATION - SCENARIO2_TC2_VERIFYFEERECEIPTPRINTOPTIONISWORKING AUTOMATION EXECUTION RESULTS**

| | | | |
|---|---|---|---|
| Date & Time | : 28-Apr-2014 03:25:55 PM | Iteration Mode | : RunAllIterations |
| Start Iteration | : 1 | End Iteration | : 1 |
| Browser | : Firefox | Executed on | : Local Machine |

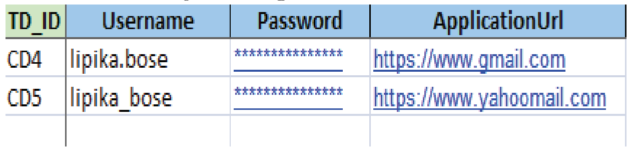
| Step_No | Step_Name | Description | Status | Step_Time |
|---|---|---|---|---|
| + Iteration: 1 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://amizone.net | DONE | 28-Apr-2014 03:26:01 PM |
| + login | | | | |
| 2 | Enter User Name | User Name entered = 1353363 | DONE | 28-Apr-2014 03:26:02 PM |
| 3 | Enter Password | Password entered = ******** | DONE | 28-Apr-2014 03:26:04 PM |
| 4 | Verify Login | Login in Successful | PASS | 28-Apr-2014 03:26:07 PM |
| + clickFeeBills | | | | |
| 5 | Click on Fee Bills | Link Clicked | DONE | 28-Apr-2014 03:26:08 PM |
| 6 | Verify Fee Bills page is displayed | Page displayed | PASS | 28-Apr-2014 03:26:09 PM |
| + clickonFeeReceipt | | | | |
| 7 | Click on Fee Receipt tab | Tab Clicked | DONE | 28-Apr-2014 03:26:10 PM |
| 8 | Verify Fee Receipt tab is displayed | Fee receipt Tab is displayed | PASS | 28-Apr-2014 03:26:12 PM |
| + clickandVerifyPrint | | | | |
| 9 | Verify Print link is available | Link is available | PASS | 28-Apr-2014 03:26:13 PM |
| 10 | Clink on Print link | Link Clicked | DONE | 28-Apr-2014 03:26:13 PM |
| 11 | Verify Print page is displayed | Print page displayed | PASS | 28-Apr-2014 03:26:13 PM |
| + logoff | | | | |
| 12 | Perform Logoff | Logoff button clicked | DONE | 28-Apr-2014 03:26:14 PM |
| 13 | Verify Logoff | Logoff successful | PASS | 28-Apr-2014 03:26:15 PM |

Execution Duration: 0 minute(s), 35 seconds

Steps passed : 6 Steps failed : 0

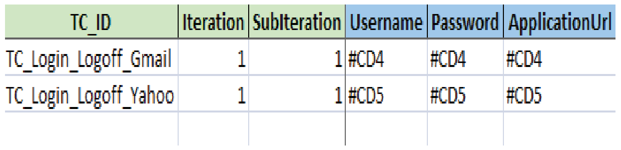
Fig 6. Screenshot of the test result generated on testing the Verification of Fee Receipt Print Option.

***Common Data Set for testing Gmail & Yahoo Mail:***



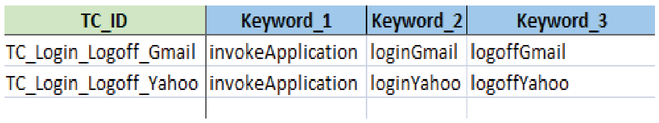
| TD_ID | Username | Password | ApplicationUrl |
|---|---|---|---|
| CD4 | lipika.bose | *************** | https://www.gmail.com |
| CD5 | lipika_bose | *************** | https://www.yahoomail.com |

***Business Flow & General Data Set for Login Logout functionality of Gmail & Yahoo Mail:***
***General Data Set:***



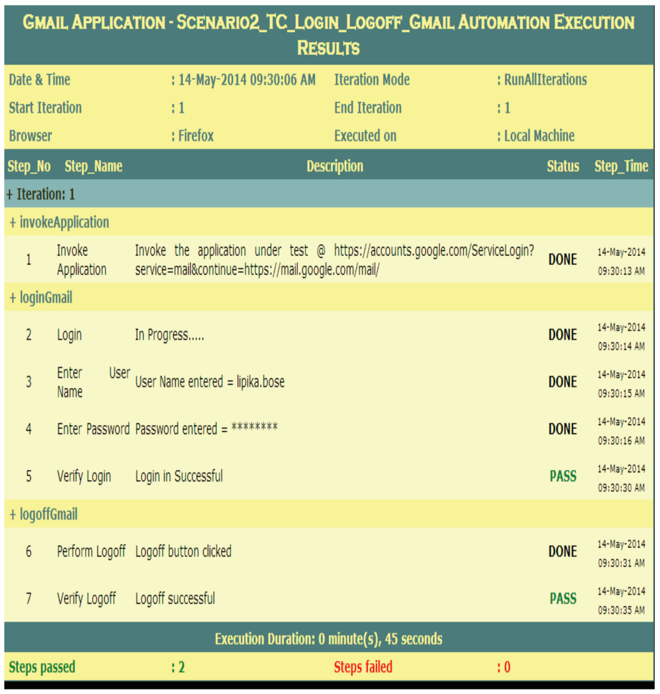
| TC_ID | Iteration | SubIteration | Username | Password | ApplicationUrl |
|---|---|---|---|---|---|
| TC_Login_Logoff_Gmail | 1 | 1 | #CD4 | #CD4 | #CD4 |
| TC_Login_Logoff_Yahoo | 1 | 1 | #CD5 | #CD5 | #CD5 |

***Business Flow:***



| TC_ID | Keyword_1 | Keyword_2 | Keyword_3 |
|---|---|---|---|
| TC_Login_Logoff_Gmail | invokeApplication | loginGmail | logoffGmail |
| TC_Login_Logoff_Yahoo | invokeApplication | loginYahoo | logoffYahoo |

**Scenario 4: Verify Login and Logout Functionality Of Gmail in Firefox:**



**GMAIL APPLICATION - SCENARIO2_TC_LOGIN_LOGOFF_GMAIL AUTOMATION EXECUTION RESULTS**

| | | | |
|---|---|---|---|
| Date & Time | : 14-May-2014 09:30:06 AM | Iteration Mode | : RunAllIterations |
| Start Iteration | : 1 | End Iteration | : 1 |
| Browser | : Firefox | Executed on | : Local Machine |

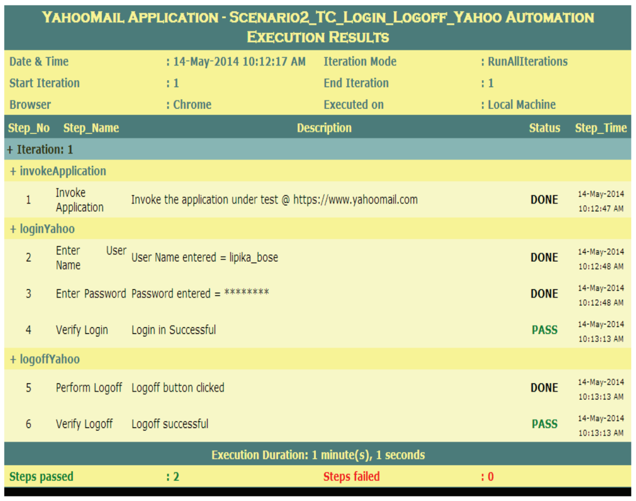
| Step_No | Step_Name | Description | Status | Step_Time |
|---|---|---|---|---|
| + Iteration: 1 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://accounts.google.com/ServiceLogin?service=mail&continue=https://mail.google.com/mail/ | DONE | 14-May-2014 09:30:13 AM |
| + loginGmail | | | | |
| 2 | Login | In Progress..... | DONE | 14-May-2014 09:30:14 AM |
| 3 | Enter User Name | User Name entered = lipika.bose | DONE | 14-May-2014 09:30:15 AM |
| 4 | Enter Password | Password entered = ******** | DONE | 14-May-2014 09:30:16 AM |
| 5 | Verify Login | Login in Successful | PASS | 14-May-2014 09:30:30 AM |
| + logoffGmail | | | | |
| 6 | Perform Logoff | Logoff button clicked | DONE | 14-May-2014 09:30:31 AM |
| 7 | Verify Logoff | Logoff successful | PASS | 14-May-2014 09:30:35 AM |

Execution Duration: 0 minute(s), 45 seconds

Steps passed : 2 Steps failed : 0

Fig 7. Screenshot of the test result generated on testing the login/logout functionality of Gmail in Firefox.

**Scenario 5:Verify login logout functionality of yahoo mail in crome:**



**YAHOOMAIL APPLICATION - SCENARIO2_TC_LOGIN_LOGOFF_YAHOO AUTOMATION EXECUTION RESULTS**

| | | | |
|---|---|---|---|
| Date & Time | : 14-May-2014 10:12:17 AM | Iteration Mode | : RunAllIterations |
| Start Iteration | : 1 | End Iteration | : 1 |
| Browser | : Chrome | Executed on | : Local Machine |

| Step_No | Step_Name | Description | Status | Step_Time |
|---|---|---|---|---|
| + Iteration: 1 | | | | |
| + invokeApplication | | | | |
| 1 | Invoke Application | Invoke the application under test @ https://www.yahoomail.com | DONE | 14-May-2014 10:12:47 AM |
| + loginYahoo | | | | |
| 2 | Enter User Name | User Name entered = lipika_bose | DONE | 14-May-2014 10:12:48 AM |
| 3 | Enter Password | Password entered = ******** | DONE | 14-May-2014 10:12:48 AM |
| 4 | Verify Login | Login in Successful | PASS | 14-May-2014 10:13:13 AM |
| + logoffYahoo | | | | |
| 5 | Perform Logoff | Logoff button clicked | DONE | 14-May-2014 10:13:13 AM |
| 6 | Verify Logoff | Logoff successful | PASS | 14-May-2014 10:13:13 AM |

Execution Duration: 1 minute(s), 1 seconds

Steps passed : 2 Steps failed : 0

Fig 8. Screenshot of the test result generated on testing the login/logout functionality of yahoo mail in chrome.

## VI. Conclusion

GRAFT is an automation testing framework which will enable cross browser testing.

Lack of any well-defined automation framework to test various application functional/non-functional scenarios lead to huge amount of manual effort of testing the same functional scenarios in multiple browsers. In agile development where sprint durations are just of 4-6 weeks, running regression suit manually across multiple browsers require huge manual effort.

Therefore this framework is beneficial for Agile Based Testing and can be used in the industries for cross browser testing. Reusability is another the most striking feature of this framework. Whenever new application needs to be tested changes are required in the Test Data engine level and in Business flow. New keywords specific to the application will be updated in the general data set and in the common data set . The corresponding implementation of those keywords will be added in the business component. No other changes are required. All the features of batch execution and reporting remains constant thereby are making this framework generic in nature.

Being a hybrid in its behavior it considers both the data sets and the keywords defined for execution of the test scenarios.

## VII. References

[1]Pajunen, T.; Takala, T.; Katara, M. "Model-Based Testing with a General Purpose Keyword-Driven Test Automation Framework" in Software Testing, Verification and Validation Workshops (ICSTW) on page no. 242-251, 2011.

[2]Selenium http://en.wikipedia.org/wiki/Selenium

[3]T. Matthews G. Leshed, E. M. Haber and T. Lau. Coscripter: Automating and sharing how-to knowledge in the enterprise. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 1719–1728, 2009

[4]Eun Ha Kim; Jong Chae Na; Seok Moon Ryoo, "Implementing an Effective Test Automation Framework" in Proceedings of COMPSAC '09. 33rd Annual IEEE International on volume 2 pages no 534-538, 2009.

[5]G. Little and R. C. Miller. Translating keyword commands into executable code. In Proceedings of the 19th ACM Symposium on User Interface Software and Technology, pages 135–144, 2006.

[6] Laukkanen, Pekka, “Data-Driven and Keyword-Driven Test Automation Frameworks”, Master’s Thesis, Software Business and Engineering Institute, Department of Computer Science and Engineering, Helsinki University of Technology, 2006.

[7]J. Bach. Agile test automation, 2003. URL http://www.satisfice.com/agileauto-paper.pdf. April 11, 2005.

[8] P. Hamill. Unit Test Frameworks. O’Reilly, 2004.

[9] E. M. Maximilien and L. Williams. Assessing test-driven development at IBM. In Proceedings of the 25th International Conference on Software Engineering, pages 564–569. IEEE Computer Society, 2003.

[10] Burnstein, Ilene, Practical Software Testing: a process-oriented approach. 709, Springer, New York, 2003.

[11] Michael Kelly, “Choose a test automation framework”, QA, Liberty Mutual, Web URL: http://www.ibm.com/developerworks/rational/library/591.html , 20 Nov 2003

[12] D. Mosley and B. Posey. Just Enough Software Test Automation. Prentice Hall PTR, 2002.

[13]C. Bird and A. Sermon. An XML-based approach to automated software testing.ACM SIGSOFT Software Engineering Notes, 26(2):64–65, March 2001.

[14] M. Fewster and D. Graham. Software Test Automation. Addison-Wesley, 1999.

[15] A. M. Memon, M. E. Pollack, and M. L. Soffa. Using a goal-driven approach to generate
test cases for GUIs. In ICSE ’99: Proceedings of the 21st international conference on Software engineering, pages 257–266. IEEE Computer Society Press, 1999.