AN AUTOMATED DATA DRIVEN CONTINUOS TESTING FRAMEWORK

Sonu Lamba[1], Vinay Rishiwal[2], Arjun Rana[3]

[1] *ABV-Indian Institute of Information Technology, Gwalior, (India)*

[2,3] *MJP Rohilkhand University, Bareilly, UP, (India)*

ABSTRACT

Automation testing plays crucial role to facilitate a user with quality software. The challenge begins when end user demands the quality software with constrained resource and time. To achieve this objective, continuous integration can be used to ensure that web applications are automatically tested via scripts as opposed to manually. Here Data Driven automation can play the major role because of its ability to increases the test coverage by executing test cases iteratively unless the volume of the test cases is gigantic.

In this paper, a data driven continuous testing framework has been proposed. In this framework multiple scripts can be run efficiently. This framework uses an excel spreadsheet to execute various test cases. This framework has been implemented in conjunction with selenium. Various test cases have been used to check the efficiency of the proposed data driven framework. These cases are being run on the anvil of different parameters. Results of the paper show that framework has the capability to handle a large volume of test cases and it can produce accurate results as per the test case. This framework completely reduces the manual dependency in automation testing.

Keywords: Automation Testing, Continuous testing, data Driven Framework, Selenium

I. INTRODUCTION

Software testing is the controlling of software under defined conditions to check whether the software works accordingly and to settle the bugs, and also to make sure that we are delivering the desired software what the user demands. Testing of software is the main phase of Software development life cycle in IT industries. It should be done during the software development process.

Historically, test automation has not met with the level of success that it could. Most often this failure is the result of incorrect design, lack of flexibility for future enhancements, unchecked redundancy etc. Automated Testing is the best way to test the product with increase in efficiency and quality. Automated testing is the process or activity of examining a software by using automation tools. Selenium is a well known open source automation tool which is widely used for testing web based applications. It runs on cross platforms such as Windows, Linux and Macintosh and supports almost all modern browsers such as Safari, Firefox, chrome, IE, Opera, etc

Automation testing demands a well described approach, based on a comprehensive framework in order to reduce time and gain maximum advantages. A test automation framework is a set of hypothesis, abstract ideas, and implementations that provides support for automated software testing.

There are various kinds of automation frameworks; some of the most commonly known frameworks include the data-driven testing, Modularity-driven testing, Keyword-driven testing and Hybrid testing.

The Data driven testing is creation of test scripts to run together with their related data sets in the framework. The main advantage of the automated tests is the reusability and also the maintenance of these tests is not complex. This needs the preparation of the data sheets which is completely independent of the test automation tool.

Some of the Advantages of data driven testing are*:*

1. Large amount of data can be fetched using the data sheet for repeated use of test case execution.
2. Reusability and maintenance.
3. Only the script representing a "Business Function" needs to be modified/updated in case if the functionality of the application under test (AUT) changes.
4. Few components of the framework are highly customizable such as test record and report.

Automation Testing reduces the need of manual or human involvement, repetitive or redundant tasks with the help of automation tool. Automated software testing improves the accuracy, test coverage within short interval of time at low cost. Test automation has specific advantages for improving the long-term efficiency of a software team's testing processes at low cost and time

Although there is a significant body of existing research on software test automation, an analysis of the related work discloses there are mostly too specific to a particular application. Also none discuss the actual problems faced in framework development and efficient solutions to those.

II. RELATED WORK

Most of the published research on test automation frameworks [2] [6-12] presents case studies or feasibility studies. This section highlights literature review phase and evolved with a problem statement with the help of work that has been published till today in the area of automation framework design for Data Driven Frameworks. Artzi et al. [1], describes rudimentary test automation framework for perform feedback-directed test generation for JavaScript web applications. This paper presented the case study of a specific system (rather than general or all web applications) also it needed access to AUT's source code. Manpreet Kaur et al. [3] have presented Xml Schema Based Approach for Testing of Software Components mentions apt use of XML format for representing test data. Authors in [4] suggested DEVELOPMENT OF TEST AUTOMATION FRAMEWORK FOR TESTING AVIONICS SYSTEMS and they describes aptly some basics for implementing data driven frameworks but again it does not give a generic model or architecture for general design. Tsai, Paul, Song, and Cao [5] presented a description of an XML based framework named Coyote which was designed for testing Web services. Again, this paper was a case study and presented no conclusions.

There are various frameworks which give assistance in automation testing such as Junit, Jersey Test Framework, Software Testing Automation Framework (STAF), Selendroid, Spock, Quick Test Professional (HP Unified Functional Testing Software), Telerik TestStudio, Framework for Integrated Test (FIT) StoryTestIQ, Ranorex, Test Automation FX, Concordion and Selenium, ranging from open source to commercial. After studied these framework we concluded each framework has its own specialties and different features

Yalla and Shanbhag [13] has describes that reusable automated testing Framework combined with open source automation tool is the solution to reduce the testing application timing and cost. Here the focus is on creating a script less automated testing Framework by using Selenium RC. The Framework provides easy methods for generating the script file and also allows the remote execution. They create an effective but easy to use automation Framework called Selenium Automation Framework (SAF) based on Selenium. MindTree Selenium Automation Framework has strengths of SeleniumRC, TestNG, JAVA, ReportNG and ANT. This combination ensures that the Framework developed meets the automation objectives. However, it has some limitation such as not using the WebDriver API which has more functionality than SeleniumRC

III. PROPOSED WORK: IMPLEMENTATION OF CONTINUOUS TESTING FRAMEWORK WITH SELENIUM

STEP 1 : Creating Project Structure For Framework
STEP 2 : Add jar Files In Project's Build Path.
STEP 3 : Creating Required Class Files
STEP 4 : Add Required .xls Files Of Data
STEP 5 : Add .xls File Reading And Writing Utility In Framework
STEP 6 : Creating Sample Data Reading Test In Framework
STEP 7 : Implementing Data Reading Test In Both Test Suites
STEP 9 : Adding testng.xml file To Run Suites From One Place
STEP 10 : Reporting Test Suite Execution Status
STEP 11 : Add Test Case Skip Function In automation framework
STEP 12 : Add Data Skip Function In Framework For Selenium
STEP 13 : Reporting Test Failure In TestNG Reports .
STEP14 : Capturing Screenshot On Failure Or Pass
STEP15 : Implement Logging Using Log4j
STEP16 : ANT - Generate XSLT Reports
STEP17 : Run WebDriver Test From Batch(.bat) File
STEP18**:** Store the test data in an Excel File.
STEP19: Store the Environment related Information in a property File.
STEP20: Store various objects in the application on which user action needs to be taken in object repository file.
STEP21: Test suite contains the logic to verify acceptance criteria mentioned in the requirement.
STEP22: Execute the script on various browsers as per need.
STEP23:Generate the reports capturing screenshots and pass/fail results. To achieve advance results TestNG is used.

We used data driven continues testing in Selenium to automate the following testing scenarios for Paytm.com as under:-

(1) User should only be able to logging its account, when we entered correct id and password otherwise not.
(2) If the login credentials are valid, then user clicked on one of the Product categories.
(3) After selecting a Product Category, user clicked on the subcategories and picked an available Product.
(4) Add the selected Product into the Cart.

(5) Now, user provides Payment details from the Excel file (TestData.xls).

(6) If the credentials are valid, transaction proceeds and this procedure is repeated with different Product name and Category.

(7) If the credentials are not valid, then the test case fails and this event is reported in form of HTML in test reports folder.

Proposed framework can be used directly across any application without spending any extra time on it. In order to make all the components of the system work in sync, it is important to define the components and its functionalities, as well as the binding relationship between them. Different packages which are used in proposed work are shown and discussed below:

- Config :- Keeps all the configuration files such as property files
- App Modules :- contains all the modules of the Application
  (1) Check_Out_Action.java
  (2) Confirmation_Action.java
  (3) Payment_Details_Action.java
  (4) Product_Select_Action.java
  (5) Sign_in_Action.java
  (6) Verification_Action.java
- Page Objects :- contains all the Page Objects of the Application
  (1) BaseClass.java
  (2) CheckOut_Page.java
  (3) Confirmation_Page.java
  (4) Home_Page.java
  (5) Login_Page.java
  (6) Product_List_Page.java
  (7) Product_Selection_Page.java
- Screenshots :- contains screenshots of various test cases
- Test Cases: - Contains the test cases for the Application testing.
- Test Data: - Contains an Excel Sheet from which data is to be fetched.
- Utility:- Contains various utility functions like:-
  (1) Excel_Utils.java
  (2) Logs.java
  (3) Constants.java
  (4) LogUtils.java

IV. TESTING FRAMEWORK, RESULTS AND DISCUSSION

Various test cases which are being used to check the proper functionality of data driven continuous testing framework are namely login including user name and password, retrieval of data, validation of payment details. The visual representations of the considered test cases are shown in Figure 2 through Figure 5 in the form of snap shots taken during the execution of test cases.

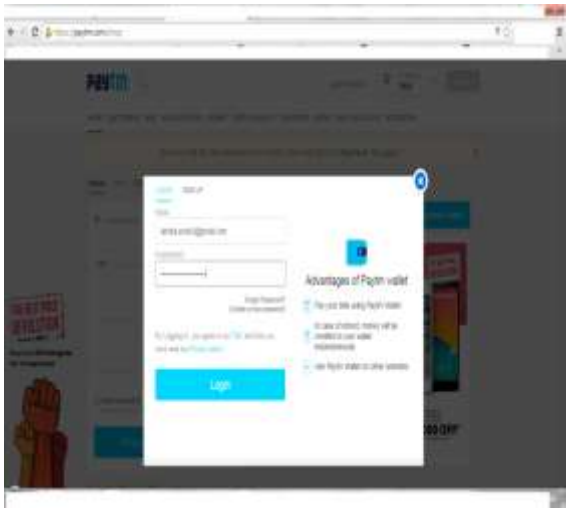

Figure 2: Snapshot of 1st Test Case (Testing the Login Functionality)

4.1 Modules Used for this Test Case

We have a Constant.java class where we declared all the constant values used in the testing of the App. This class looks like this:-

```
Public class Constant {
        public static final String URL = "https://paytm.com/shop";
        public static final String Username = "testuser_1";
        public static final String Password ="Test@123";
            public static final String Path_TestData =
"F://SetUp2//EclipsePortable//Data//workspace//Day1//src//testData//";
            public static final String File_TestData = "TestData.xlsx";
                        //Test Data Sheet Columns
            public static final int Col_TestCaseName = 0;
            public static final int Col_UserName =1 ;
            public static final int Col_Password = 2;
            public static final int Col_Browser = 3;
            public static final int Col_ProductType = 4;
            public static final int Col_ProductNumber = 5;
            public static final int Col_FirstName = 6;
            public static final int Col_LastName = 7;
            public static final int Col_Address = 8;
            public static final int Col_City = 9;
            public static final int Col_Country = 10;
            public static final int Col_Phone = 11;
            public static final int Col_Email = 12;
            public static final int Col_Result = 13;
            public static final String Path_ScreenShot =
"F://SetUp2//EclipsePortable//Data//workspace//Day1//src//Screenshots//";
        }
```

We have a LogIn.java class where the methods are defined.

```
public static WebElement txtbx_UserName() throws Exception{
        try{
                element = driver.findElement(By.id("log"));
                Log.info("Username text box is found on the Login Page");
        }catch (Exception e){
                 Log.error("UserName text box is not found on the Login Page");
                 throw(e);
                 }
        return element;
      }
        public static WebElement txtbx_Password() throws Exception{
        try{
                 element = driver.findElement(By.id("pwd"));
                Log.info("Password text box is found on the Login page");
        }catch (Exception e){
                 Log.error("Password text box is not found on the Login Page");
                 throw(e);
                 }
        return element;
    }
        public static WebElement btn_LogIn() throws Exception{
        try{
                 element = driver.findElement(By.id("login"));
                Log.info("Submit button is found on the Login page");
        }catch (Exception e){
                 Log.error("Submit button is not found on the Login Page");
                 throw(e);
                 }
        return element;
    }
```

Figure 3: Snapshot for testing the other test scenario's for Paytm.com

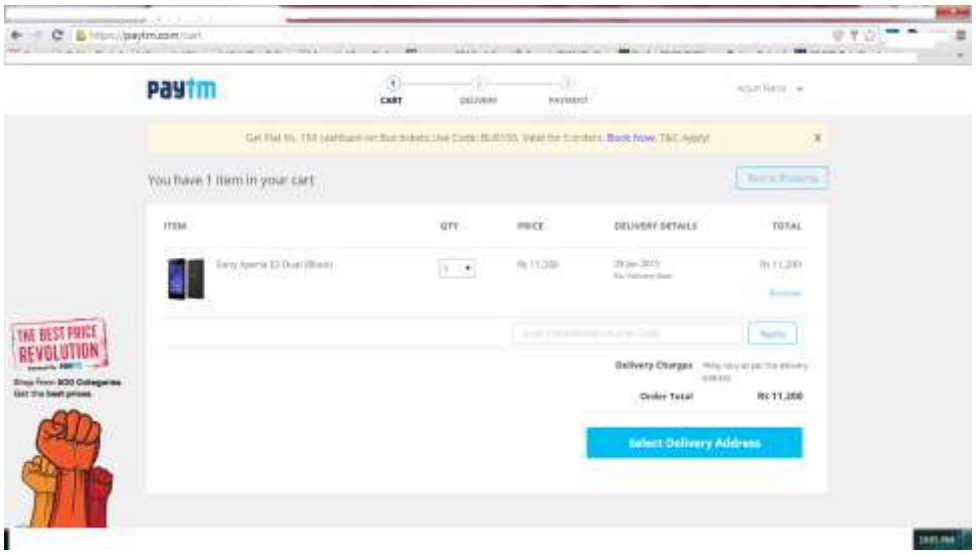

Figure 4: Retrieving Data From Excel Sheet:-

We have ExcelUtils.java class which contains all the methods needed for retrieving the data from ExcelSheet. This class contains methods:-

// This method is used to set the file path.

public static void setExcelFile(String Path,String SheetName)

//This method is to read the test data from the Excel cell, in this we are passing parameters as Row num and Col num

public static String getCellData(**int** RowNum, **int** ColNum)

//This method is to write in the Excel cell, Row num and Col num are the parameters

public static void setCellData(String Result, **int** RowNum, **int** ColNum)

public static int getRowContains(String sTestCaseName, **int** colNum)

public static int getRowUsed()

Logs:

Initialize the logger for getting logs with: static Logger log = Logger.getLogger(Test.class.getName())

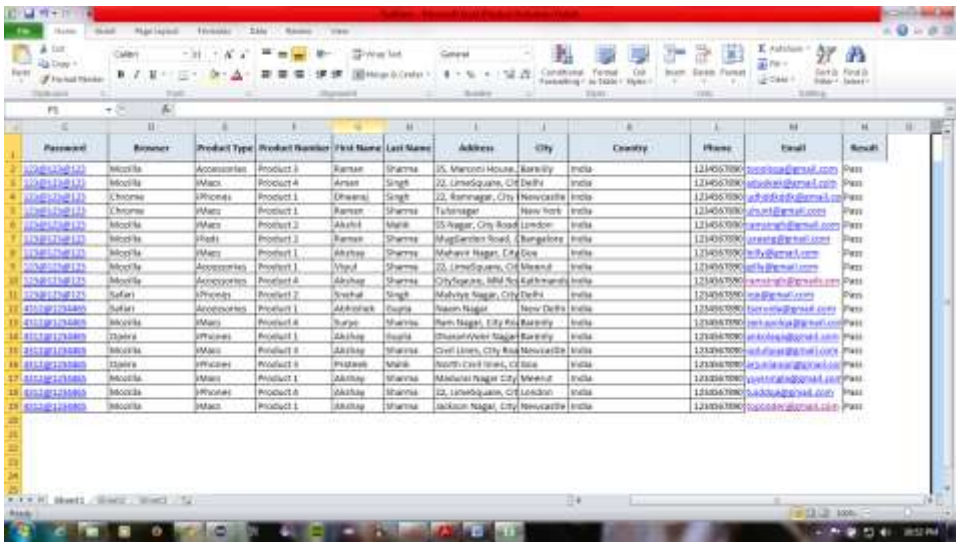

Figure 5: Snapshot of Excel Sheet used in the Data Driven Automaton Framework.

4.2 Results

For the proposed work, results are generated in the form of report using testNG which is a well accepted report generation framework used for the Java programming language. The TestNG framework is introduced to overcome the limitations of JUnit framework. The manual analysis of the generated results changed to automatic report generation with the Inclusion of TestNG in selenium web driver. A report which is generated using TestNG is shown in Figure 6. This report is clearly showing all the execution steps performed over the considered test scenarios.

Figure 6: Report for Considered Test Cases

V. CONCLUSIONS AND FUTURE WORK

In this paper, a data driven continuous testing framework has been proposed and tested over different test cases. This framework can be used directly across any application without incurring much time. This framework can also be run efficiently on any web browser. This framework is robust enough to handle a large size of test cases. The limitation of data driven framework is that non technical users can not use it well and sometimes it cannot explore the reusability of library functions at their best. To remove/improve these limitations a hybrid

framework is required which can facilitate the non technical user with keyword driven approach as well as which can make use of reusability feature as much as possible. Another important issue which may arise in data driven approach is that when the test cases becomes gigantic then an efficient procedure/algorithm will be required to run the test cases on a faster rate as compare to traditional test case algorithms.

REFERENCES

[1] Shay Artzi, Julian Dolby, Simon Holm Jensen, Anders Møller, and Frank Tip. A framework for automated testing of javascript web applications. In Proceeding of the 33rd international conference on Software engineering, ICSE '11, pages 571–580, New York, NY, USA, 2011. ACM.

[2] James M. Slack "System Testing of Desktop and Web Applications" Information Systems Education Journal (ISEDJ). Volume 9, No. 3 August 2011.

[3] 2010 paper by Manpreet Kaur et all "Xml Schema Based Approach for Testing of Software Components" International Journal of Computer Applications, Volume 6– No.11, September 2010

[4] 2010 paper by Ashutosh Jha "DEVELOPMENT OF TEST AUTOMATION FRAMEWORK FOR TESTING AVIONICS SYSTEMS" Digital Avionics Systems Conference (DASC), 2010 ieee.

[5] W.T. Tsai, R. Paul, S. Weiwei, and C. Zhibin, "Coyote: an XML-based Framework for Web Services Testing", Proceedings of the 7th IEEE International Symposium on High Assurance Systems Engineering, 2002, pp. 173-174.

[6] C. Merchant, M. Tellez, and J. Venkatesan, "A Browser yu6Agnostic Web Application UI Test Framework:

[7] Motivation, Architecture, and Design", Proceedings of the 6th International Conference on Information Technology New Generations, April 2009, pp. 748-751.

[8] "Writing GUI Applications in Perl/Tk" article in Visual Developer Magazine http://www.perl.com/pub/2001/03/gui.html

[9] GoogleCharts: http://code.google.com/apis/chart/image/docs/chart_playground.html and Perl module: http://search.cpan.org/~dmaki/Google-Chart-0.05014/lib/Google/Chart.pm

[10] Selenium for Web Automation: http://seleniumhq.org/

[11] D.R. Kuhn, D.R. Wallace, and A. Gallo, "Software Fault Interactions and Implications for Software Testing," IEEE Trans. Software Eng., vol. 30, no. 6, 2004, pp. 418–421.

[12] http://www.satisfice.com/tools.shtml James Bach, AllPairs Perl Script.

[13] M. Yalla and M. Shanbhag, "Building automation framework around open source technologies," in proc. of Software Testing Conference, pp. 6-9, Banglore, India, November, 2009