BioGecko

Vol 12 Issue 01 2023

ISSN NO: 2230-5807

Automation Testing Using Cucumber and Selenium WebDriver

Prof. Neha Mittal[1]
Assistant Professor: - Department of BCCA,
Dr. Ambedkar Institute of Management Studies & Research,
Nagpur, Maharashtra.
Email: -mittalneha218@gmail.com

Prof. Nikhil Khandar[2]
Assistant Professor: - Department of BCCA,
Dr. Ambedkar Institute of Management Studies & Research,
Nagpur, Maharashtra.

***Abstract* –** The testing step of the SDLC is the most important. Testing is a step in the quality assurance process that is used to identify mistakes and faults that occurred during application development, ensuring that users may use the product with confidence. By using critical thinking and a logical method, the tester imagines a business situation. The automated tools can execute, report, and compare the results with earlier test runs to make a tester's job easier. The goal in this study is to produce a complete output report that includes data on the test cases that were run during the test run. With the assistance of the free and open-source Selenium web driver, Maven, and Cucumber, the researcher is testing a web application using the BDD framework.Each requirement for this article will have a feature file created by the researcher, who will then map it to the appropriate step definition class. All of the mapped methods in the step definition class will be called as we run the feature file. Detailed information about the cases that were executed according to the feature file will be included in the output that is generated.

Index Terms – Automation, BDD, Cucumber, Selenium WebDriver

INTRODUCTION

Clients will communicate all of their requirements to the business analyst during the software development process. The business analyst will then verify the collected requirements and build the programme to meet the client's primary criteria. The third phase involves the developer, who carefully considers the business analyst's requirements before writing the code. To test the software, the Quality Assurance team will create the scenarios. Software testing is done in a methodical way to detect defects and to make sure that the results we get are what we anticipate them to be. Automation testing is used instead of manual testing to boost success rates. We have two fundamental methods for testing. Black box testing and white box testing are two examples of testing.Black box testing ignores internal testing procedures and focuses primarily on an application's functionality; it is typically carried out as a validation process. Validation is carried out in order to determine whether or not the specific customer need has been met. White box testing is another type of testing that examines an application's internal process flow and produced code. Selenium, which is used for functional and regression testing, is employed in this case for automation testing. Black box testing includes regression and functional testing.

Before they are deployed, we can automate web applications with the help of the Selenium set of API and Jars. Selenium is platform independent and supports a wide range of operating systems, browsers, and computer languages. Selenium has a variety of tools with various features and abilities. These technologies include the Selenium IDE (Integrated Development Environment), Selenium Web Driver, Selenium Grid, and the discontinued Selenium RC (Remote Control). Selenium Web Driver is being used in this case.

Selenium 2.0 is another name for Selenium Web Driver. It is a tool for web application testing. It is quicker than SeleniumRC since it can communicate directly with browsers, as opposed to SeleniumRC's use of a server that injects a http proxy into browsers to comprehend them. In the selenium web driver test suite, test cases are constructed, and then element locator, object locator, and web driver methods are used to run them. Web Driver allows data-driven testing and does not have any cryptic commands. Any language test scripts can call the Selenium web driver and automate the online application, making it faster and clearer.Since the Selenium web driver is unable to produce thorough test reports, we employ BDD (feature files), which can.

Software testing tool Cucumber uses the Behavior Driven Development (BDD) method for testing applications. Tests are written in plain text using Cucumber in the style of Given, When, and Then statements, where the Given statement lists all prerequisites, the When statement lists all conditions, and the Then condition lists the intended result. In order to make it easier for a non-technical person to comprehend what is actually intended to happen, these steps are expressed in plain language, which hides the technicalities and complexities of the automation scripts bound underneath. The feature files for these steps are located in an IDE.

Here is a quick test as an illustration:
I've already accessed the Google Search homepage.
I would see some search results after entering some text and clicking the Google Search button.

By enabling direct contact between our code and the web browser using the Selenium WebDriver API, the appropriate actions can be carried out automatically. With the help of the WebDriver API, we may create a variety of implementation classes for diverse contexts, including Chrome Driver, Firefox Driver, Html Unit Driver, Safari Driver, Internet Explorer Driver, Android Driver, and others for the most widely used web browsers.
Using Locators, Selenium WebDriver locates each HTML element that is present in a web application. Eight different locator types are supported by Selenium, including id, name, class Name, xPath, link Text, partial Link Text, tag Name, and css Selector. Selenium WebDriver gives us the ability to use the find Element method to locate any sort of WebElement that is available in a web application by properly employing these locators. A WebElement can be uniquely recognised when more than one element has the same locator values by using the find Elements function.

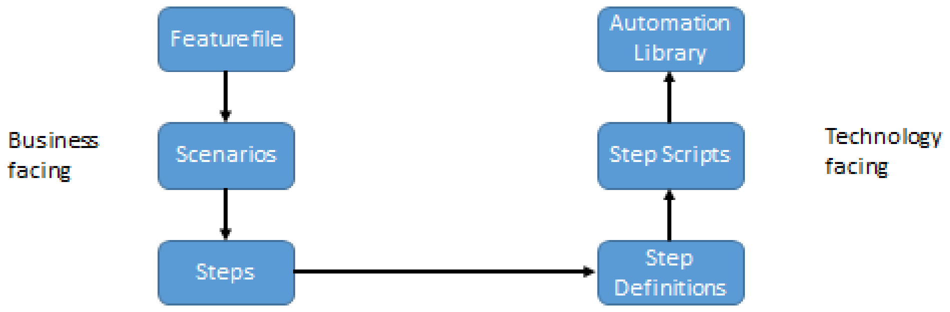


Figure 1 – Framework of automation

UNDERSTANDING CONCEPT OF BDD FRAMEWORK

Software development employs the behavioural driven development technique, in which every team member communicates to fully comprehend the requirements. Here, the product owner, tester, business analyst, and developer discuss and work together on the need based on business needs. The owner outlines the requirement and desired programme behaviour. Developers and testers are able to produce the desired outcome thanks to these interactions. They gain a clear understanding with the aid of user stories and then

compose the requirements into a feature file, also known as a gherkin, which is written in plain English. Several benefits of BDD are -
1) Effective cooperation
2) Wide exposure
3) The software design takes business value into account.
4) Common language
5) Greater assurance on the part of the developers
6) Reduced prices

A. Cucumber
Cucumber is a tool for software testing that helps business managers and software developers communicate better. It was developed to handle behavioural driven development (BDD), in which needs are implied in feature files written in a simple form of English called Gherkin language. It follows the "Given, When, Then" pattern. The Gherkin-written test scenarios are saved in a feature file. The scrum team will first understand the requirements before writing the step definitions with the quality assurance team.

B. Feature File
Feature files are used to store test cases that are written as files. We develop a unique feature file for each Cucumber feature. For a certain feature file, a scenario and scenario outline are created. using the ".feature" file extension. Feature files are made up of certain basic keywords.
1) A feature describes a particular functionality in a test case.
2) The scenario describes how the functionality operates.
3) Giving refers to what is given to us.
4) When Next Particular circumstance.
5) What happens next if the condition is met.
6) Scenario outline: we'll provide numerous inputs for a single scenario so it can test them all

C. Step Definition
The researcher invokes an intermediate step definition class to carry out our job as described in the feature file. The function's code is mapped to each step in the feature file for execution. Now, by scanning the step definition file, cucumber will execute every scenario in the feature file.
1) Annotation - Cucumber has some pre-defined content that will aid the compiler's execution.

MAVEN
Software management tools include Maven. We add particular dependencies that our project needs to Maven. Projects are built using Maven. On the project object model, Maven works. An xml file is the POM. The software project's data is contained in the POM file. Our project is configured through POM. Java libraries and plugins are downloaded by Maven from a repository. Maven assists us by downloading each and every crucial jar needed for the project.

A. Dependency - The researcher will add a specific dependent by providing their version, scope, group id, and artefact id. Additionally, it will download every jar file it needs from the web server's maven repository.

TESTNG
The researcher will employ the more potent TestNG framework for this study. Next generation is indicated here by NG. It is an open source tool that works best for integration testing.
A. Benefits of TestNG
1) Test case prioritisation
2) Testing is done in sequence,

Vol 12 Issue 01 2023
ISSN NO: 2230-5807

3) test cases are grouped.
4) Concurrent testing.
5) Log generation
6) Notes
7) HTML reports are provided by TestNG.

B. Annotation
An annotation can specify techniques. The approach has no established structure or pattern. There is an option to pass more parameters. Compilers can readily identify problems thanks to annotations.

REQUIREMENT SPECIFICATION

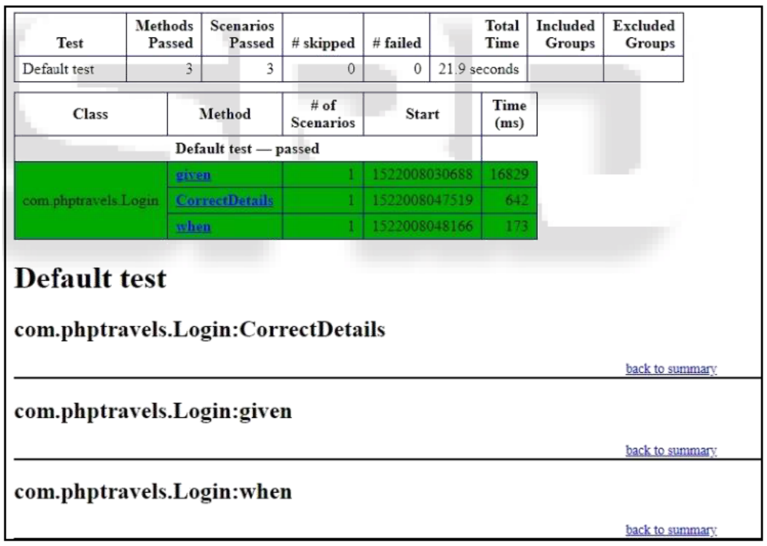
A document that describes software needs is called a software requirement specification. Stakeholders, business analysts, developers, and testers all share it. As a result, developers and testers move forward with writing code and test cases, respectively.

AIRLINE APPLICATION

The purpose of an airline application is to offer clients services like flight search, flight booking, ticket cancellation, etc. This application will assist users in researching their trip choices. As this application provides to create an account in which all the previous and upcoming tickets will be presented, the customer may manage and view his recent tickets. Due to increased market rivalry, customers are searching for affordable and dependable flight options. According to client searches, this application would offer the best flights. The following features of the airline application will be tested by the researcher:
1) Access your account (Successful and Unsuccessful)
2) Lookup flights

EXPERIMENTATION & IMPLEMENTATION

A. Issue Analysis - The researcher will talk about historical issues with analysing test results in this part. They have adopted TestNG for better reporting and business user understanding. However, the issue has not been totally fixed because the TestNG report does not give specific details about the cases that were ran.



Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
Default test	3	3	0	0	21.9 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
	Default test — passed			
com.phptravels.Login	given	1	1522008030688	16829
	CorrectDetails	1	1522008047519	642
	when	1	1522008048166	173

Default test

com.phptravels.Login:CorrectDetails

back to summary

com.phptravels.Login:given

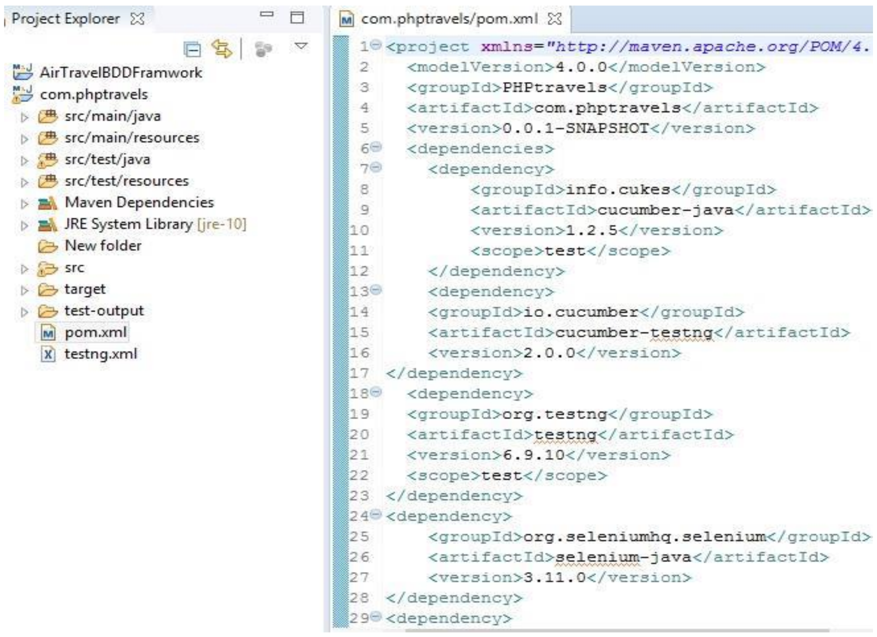
back to summary

com.phptravels.Login:when

back to summary

Fig. 1: TestNG report

However, there are some drawbacks because it does not give specific information about the scenarios that are passed or failed, and which step was failed in case of failure. Figure 1 shows information about passed scenarios and methods, and execution time is also displayed.



```
AirTravelBDDFramwork
com.phptravels
  src/main/java
  src/main/resources
  src/test/java
  src/test/resources
  Maven Dependencies
  JRE System Library [jre-10]
  New folder
  src
  target
  test-output
  pom.xml
  testng.xml
```

```xml
<project xmlns="http://maven.apache.org/POM/4.
  <modelVersion>4.0.0</modelVersion>
  <groupId>PHPtravels</groupId>
  <artifactId>com.phptravels</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <dependencies>
    <dependency>
        <groupId>info.cukes</groupId>
        <artifactId>cucumber-java</artifactId>
        <version>1.2.5</version>
        <scope>test</scope>
    </dependency>
    <dependency>
    <groupId>io.cucumber</groupId>
    <artifactId>cucumber-testng</artifactId>
    <version>2.0.0</version>
</dependency>
  <dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>6.9.10</version>
  <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.11.0</version>
</dependency>
<dependency>
```

Fig. 2: Maven Pom.Xml File

Figure 2 displays the POM.xml file. There are several dependencies in it. All the prerequisites needed to test this application can be added by the researcher. All of the jar files will be fetched by Maven from its repository.

B. Writing Test Cases

1) Login Page - The login page is broken down into three parts in figure 3. They are the text boxes for the email box, password box, and login button (submit button). We will enter the customer's registered email address in the email text field. Two scenarios call for logging in:

1) Use your registered email to login.
2) Use an unregistered email to log in.

User should log into account after entering valid and registered email address and password. Customer should not be able to login to the account if user enters erroneous or unregistered email ID and password.

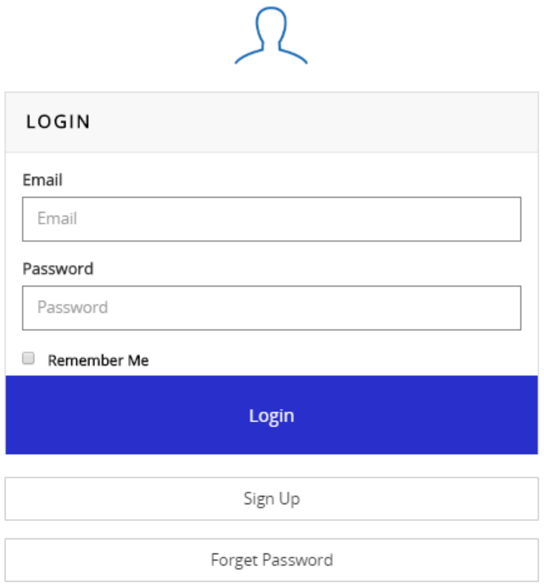
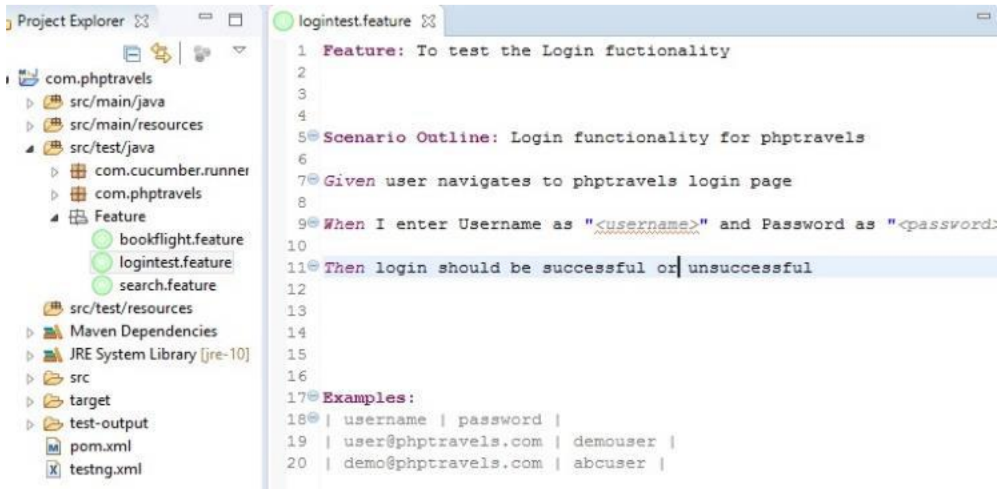


Fig. 3: Login Page

Figure 4 shows the login feature file that the researcher generated for the airline application's login functionality. Every line in the feature file is mapped to the step definition class' equivalent code. The test case was done by the researcher using Scenario Outline and two different sets of data.
A feature file will execute each method in the step specification class in turn when launched as a cucumber feature (given, when, then). For every set of data in the example scenario outline in the feature file, the same flow will be executed.



```
Feature: To test the Login fuctionality



Scenario Outline: Login functionality for phptravels

Given user navigates to phptravels login page

When I enter Username as "<username>" and Password as "<passvord:

Then login should be successful or unsuccessful






Examples:
| username | password |
| user@phptravels.com | demouser |
| demo@phptravels.com | abcuser |
```

Fig. 4: Feature File for Login Functionality

2) Search Page - Figure 5 displays the search page. The user will enter the origin and destination, departure date, and passenger count. When a user clicks the search button, a list of flights that match their search parameters ought to appear.

Vol 12 Issue 01 2023
ISSN NO: 2230-5807

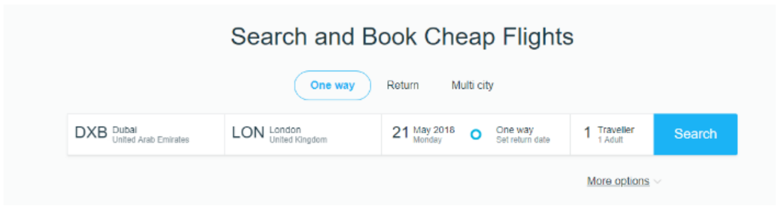
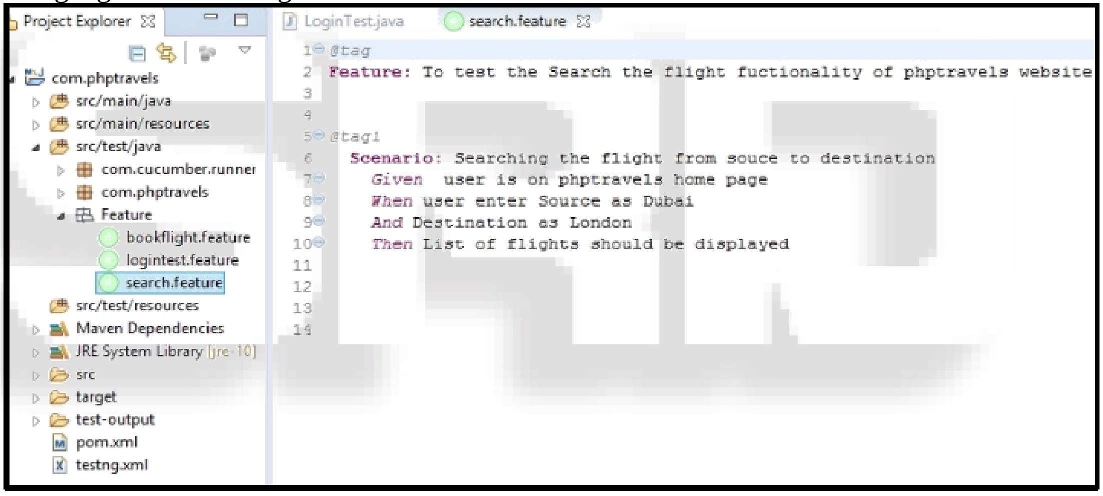


Figure 5: Search Page

With the help of the "provided, when, then" keywords, we will describe our scenario in a straightforward gherkin language as seen in Figure 6.



```
@tag
Feature: To test the Search the flight fuctionality of phptravels website


@tag1
  Scenario: Searching the flight from souce to destination
    Given  user is on phptravels home page
    When user enter Source as Dubai
    And Destination as London
    Then List of flights should be displayed
```

Fig. 6: The Search Functionality Feature File

Figure 7 displays the search page's results; here, the researcher can see all of the flights that are available based on choices.

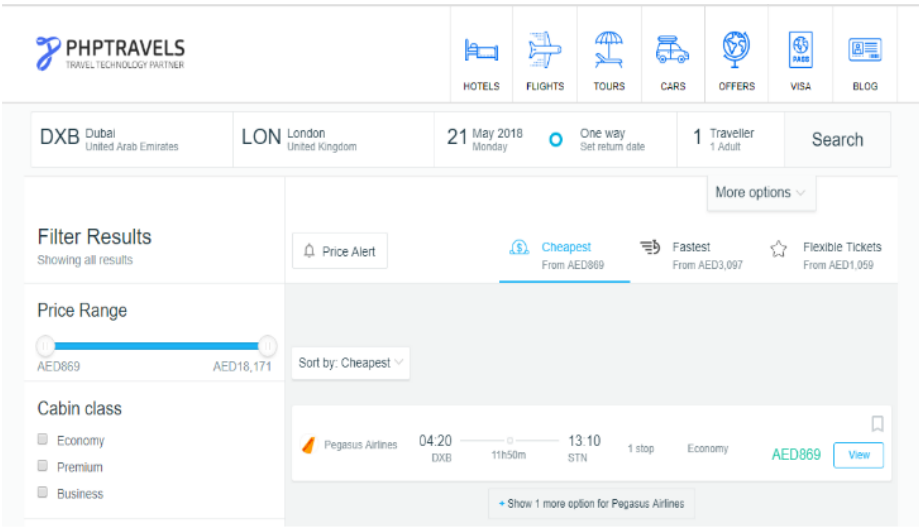
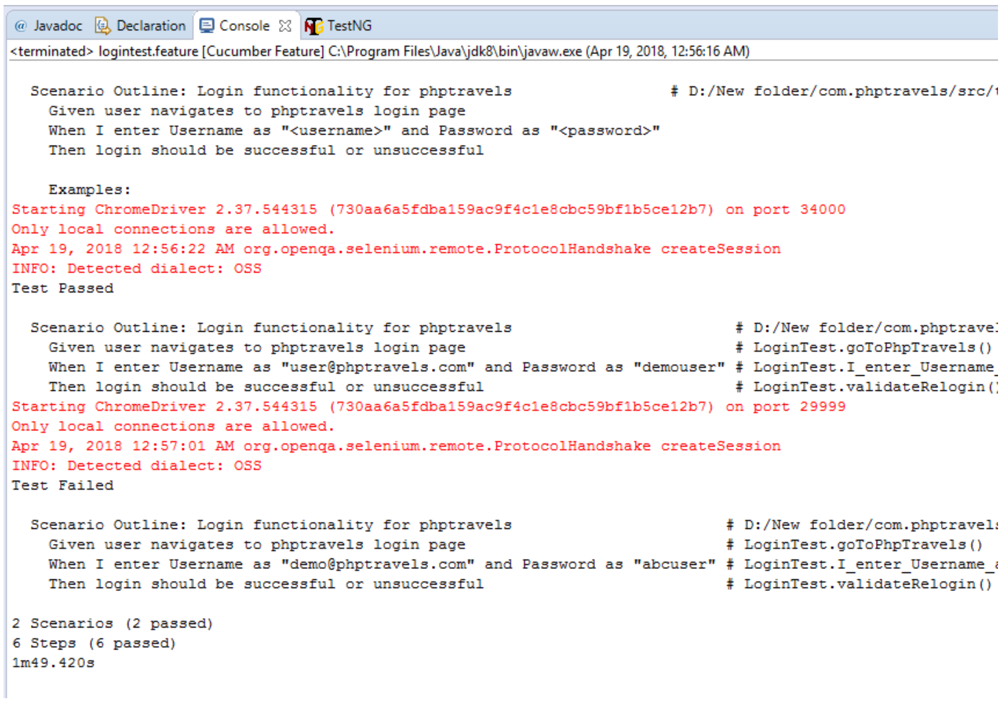


Fig. 7: Search File Result

TEST RESULT

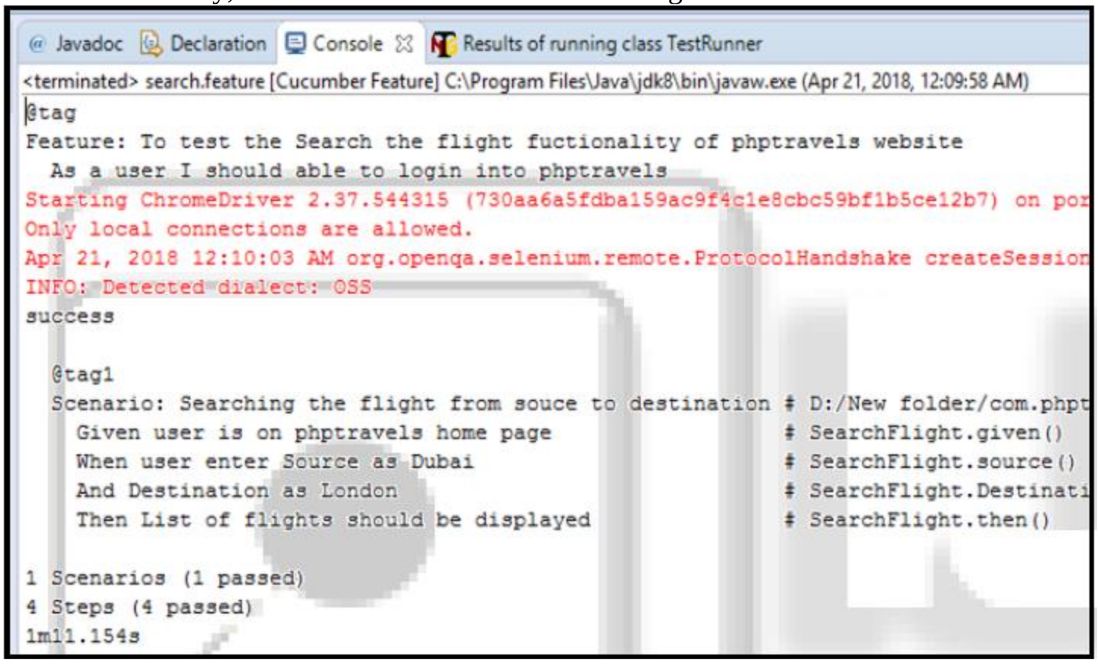
Figure 8 displays cucumber runner's output. The results of the login functionality are displayed in this figure. a situation where one scenario description passes while another fails.



```
@ Javadoc  Declaration  Console ⊠  TestNG
<terminated> logintest.feature [Cucumber Feature] C:\Program Files\Java\jdk8\bin\javaw.exe (Apr 19, 2018, 12:56:16 AM)

  Scenario Outline: Login functionality for phptravels              # D:/New folder/com.phptravels/src/
    Given user navigates to phptravels login page
    When I enter Username as "<username>" and Password as "<password>"
    Then login should be successful or unsuccessful

    Examples:
Starting ChromeDriver 2.37.544315 (730aa6a5fdba159ac9f4c1e8cbc59bf1b5ce12b7) on port 34000
Only local connections are allowed.
Apr 19, 2018 12:56:22 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS
Test Passed

  Scenario Outline: Login functionality for phptravels                     # D:/New folder/com.phptrave
    Given user navigates to phptravels login page                          # LoginTest.goToPhpTravels()
    When I enter Username as "user@phptravels.com" and Password as "demouser" # LoginTest.I_enter_Username_
    Then login should be successful or unsuccessful                        # LoginTest.validateRelogin(
Starting ChromeDriver 2.37.544315 (730aa6a5fdba159ac9f4c1e8cbc59bf1b5ce12b7) on port 29999
Only local connections are allowed.
Apr 19, 2018 12:57:01 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS
Test Failed

  Scenario Outline: Login functionality for phptravels                    # D:/New folder/com.phptravel
    Given user navigates to phptravels login page                         # LoginTest.goToPhpTravels()
    When I enter Username as "demo@phptravels.com" and Password as "abcuser" # LoginTest.I_enter_Username_
    Then login should be successful or unsuccessful                       # LoginTest.validateRelogin()

2 Scenarios (2 passed)
6 Steps (6 passed)
1m49.420s
```

Fig. 8: Cucumber Output

Here, the researcher may see in plain sight which tests were successful and which ones were not. These output numbers demonstrate how it would unmistakably fix the TestNG issue. Figure 9 shows the results of the search functionality, which show which scenarios and stages have finished.



```
@ Javadoc  Declaration  Console ⊠  Results of running class TestRunner
<terminated> search.feature [Cucumber Feature] C:\Program Files\Java\jdk8\bin\javaw.exe (Apr 21, 2018, 12:09:58 AM)
@tag
Feature: To test the Search the flight fuctionality of phptravels website
  As a user I should able to login into phptravels
Starting ChromeDriver 2.37.544315 (730aa6a5fdba159ac9f4c1e8cbc59bf1b5ce12b7) on por
Only local connections are allowed.
Apr 21, 2018 12:10:03 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS
success

  @tag1
  Scenario: Searching the flight from souce to destination # D:/New folder/com.phpt
    Given user is on phptravels home page                  # SearchFlight.given()
    When user enter Source as Dubai                        # SearchFlight.source()
    And Destination as London                              # SearchFlight.Destinati
    Then List of flights should be displayed               # SearchFlight.then()

1 Scenarios (1 passed)
4 Steps (4 passed)
1m11.154s
```

Fig. 9: Output of Cucumber for Searching a Flight

CONCLUSION

The researcher used the BDD Framework and the Selenium Web Driver in this study. in order to enhance communication and comprehension between the people involved. It also gives all participants a lot of visibility because it employs general language that everyone can understand. This allows developers to produce high-quality products that perfectly fulfil user needs and lower the cost of maintaining their code.

REFERENCES

- P. V. Sagar, R. Paruchuri and S. Vemuri, "Testing using selenium web driver," IEEE, 23 November 2017.
- D. Gaur and D. R. S. Chhilla, "Implementation of Selenium with JUNIT and Test-Ng," IJCSMS International Journal of Computer Science and Management Studies, vol. 12, no. 3, pp. 226-232, 2012.
- K. a. D. W. Andreas Bruns, "Web Application Tests," IEEE, pp. 88-91, 2009.
- S. Sivanandan and Y. C. B, "Agile Development Cycle: Approach to Design an Effective Model Based Testing".
- P. Yalla, D. L. S. S. Reddy, M.Srinivas and S. M. Rao, "Framework for Testing Web Applications using Selenium Testing tool with respect to Integration Testing," IJCST International Journal of Computer Science and technology, vol. 2, no. 3, pp. 165-170, September 2011.
- V. K. Shah, "Dynamic Heuristic Approach of An Acceptance Test Data Generation with Behavior Driven For Selenium Test," International Journal of Scientific & Engineering Research, vol. 3, no. 6, June 2012.
- L. Nagowah and K. Doorgah, "Improving Test Data Management in Record and," IEEE, pp. 931-937, 2012.
- Jagannatha, Niranjanamurthy, Manushree and Chaitra, "Comparative Study on Automation Testing using Selenium Testing Framework and QTP," International Journal of Computer Science and Mobile Computing, vol. 3, no. 10, pp. 258-267, October 2014.
- S. Singla and H. Kaur, "Selenium Keyword Driven Automation Testing Framework," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 6, pp. 125-129, June 2014.
- P. Bindal, "Test Automation Selenium WebDriver using TestNG," Journal of Engineering Computers & Applied Sciences(JECAS), vol. 3, no. 9, September 2014.
- S. Gojarea, R. Joshib and D. Gaigaware, "Analysis and Design of Selenium WebDriver Automation Testing," science direct, pp. 341-346, 2015.
- Chandraprabha, A. Kumar and S. Saxena, "Data Driven Testing Framework using Selenium," International Journal of Computer Applications, vol. 118, no. 18, pp. 18-23, May 2015.
- R. J. C and R. Kaluri, "DESIGN OF AUTOMATION SCRIPTS EXECUTION APPLICATION FOR SELENIUM WEBDRIVER AND TestNG FRAMEWORK," ARPN Journal of Engineering and Applied Sciences, vol. 10, no. 6, pp. 2440-2445, 2015.
- P. A. Motwani, A. Agrawal, D. P. N. Singh and P. A. Shrivastava, "Novel Framework for Browser Compatibility Testing of a Web Application using Selenium," (IJCSIT) International Journal of Computer Science and Information Technologies, vol. 6, pp. 5196-5162, 2015.
- Okolnhchyi and k. fogen, "A study of tools for behavior driven development," RWTH aachen university, Germany, 2016.
- Keus and a. dyck, "How much does testing cost," in SWC seminar RWTH achen university, Germany, 2016.
- "Guru99," [Online]. Available: https://www.guru99.com/selenium-tutorial.html.
- "Seleniumhq," [Online]. Available: https://www.seleniumhq.org/.