

CUCUMBER:

- It's a Behavior Driven Development test tool (BDD).
- Cucumber is a software tool used by computer programmers used for testing the developed software's.
- It runs automated acceptance test written in BDD.

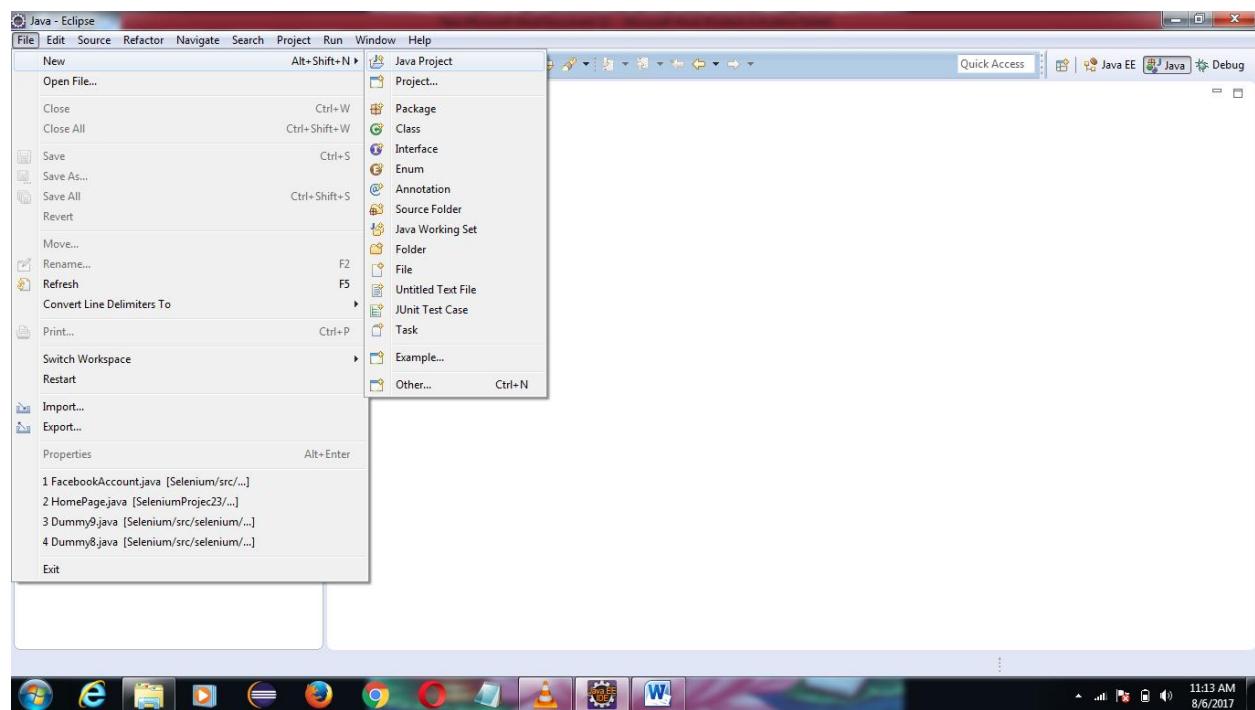
Things needed for Cucumber:

1. Feature file → It consists of scenarios.
2. Step definition file → It consists of coding part (java +selenium).
3. Test runner class → It consists of Feature files.

Simple program using cucumber:

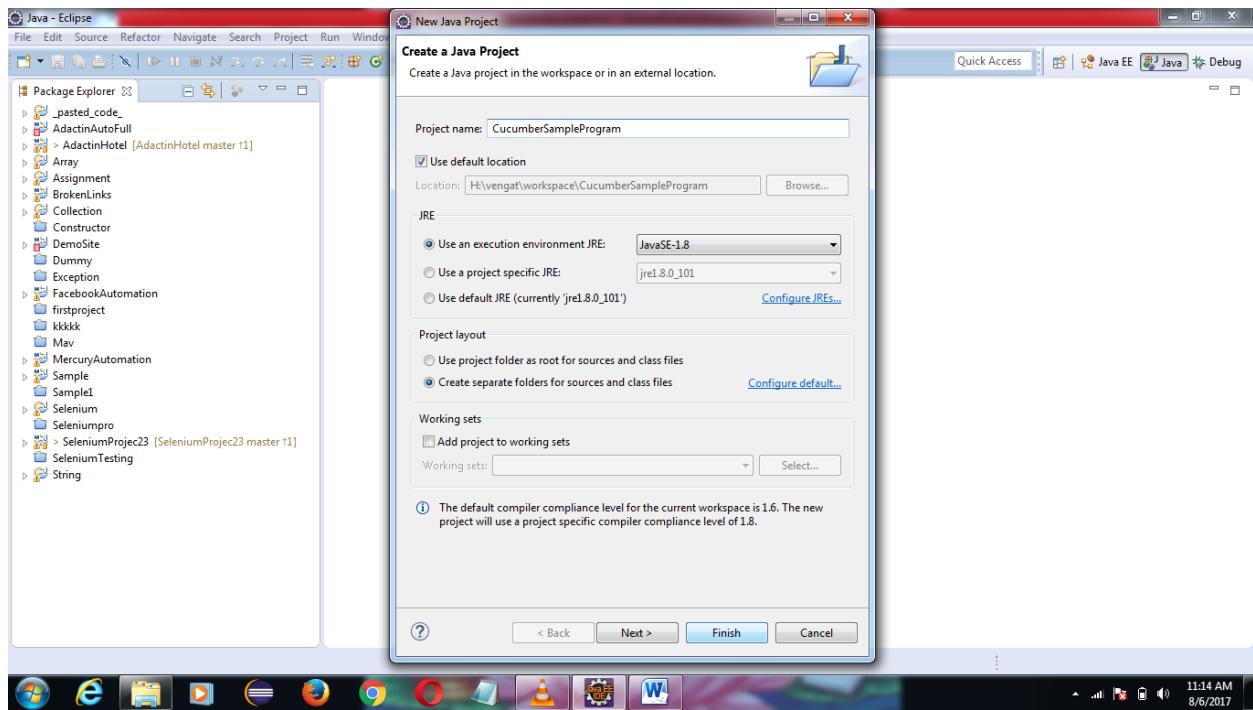
Step 1:

Create new java project.



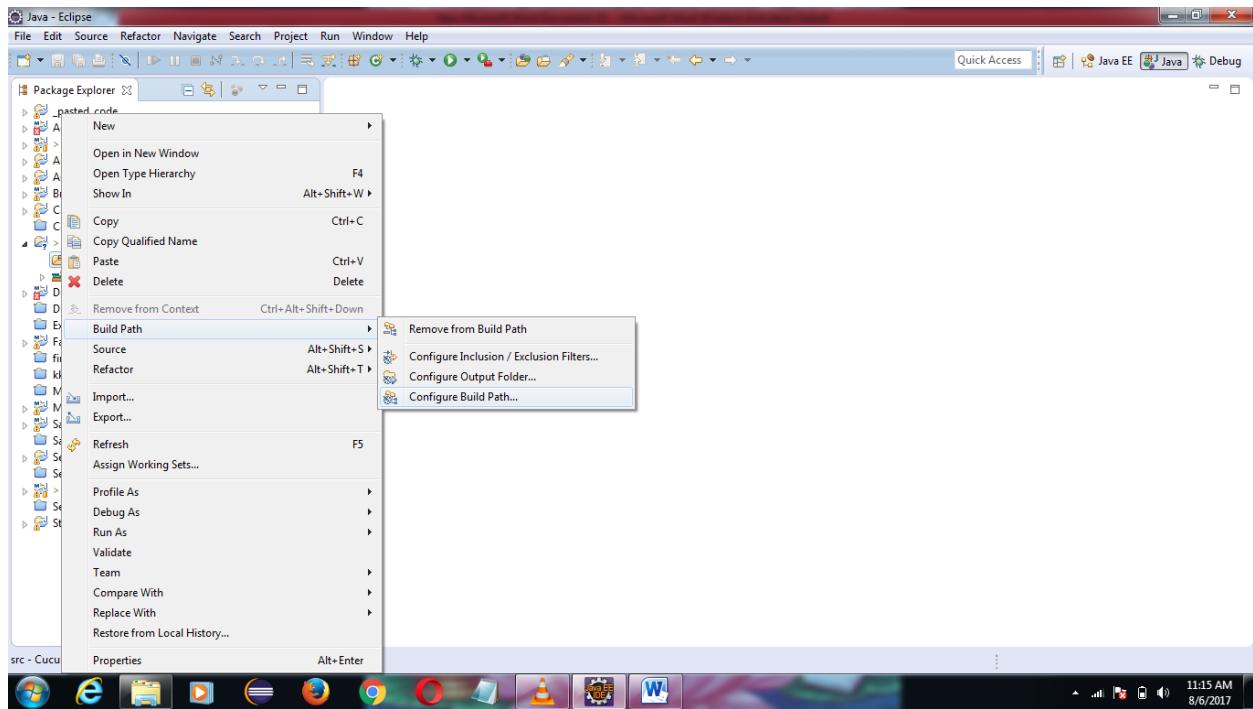
Step 2:

Enter the project name and then click finish.



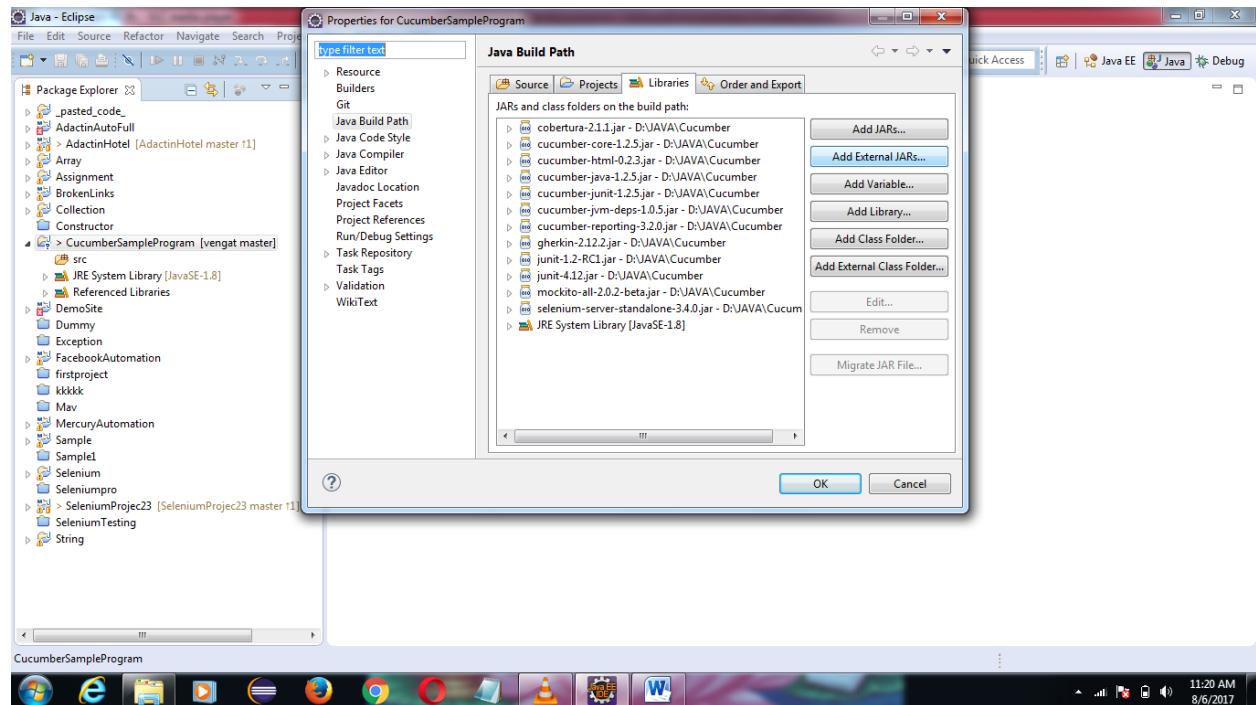
Step 3:

We need to configure jar files.

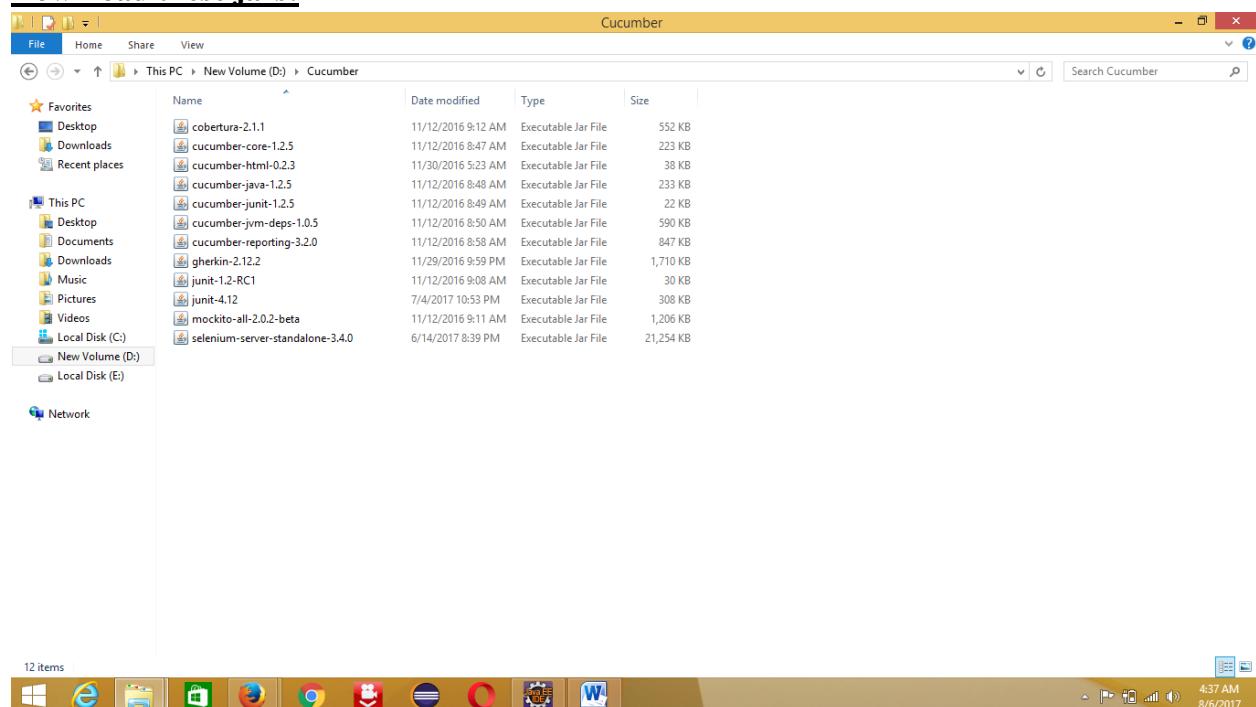


Step 4:

We need to configure our jar files by click Add External jars and then choose our location of the jar, then select the jars we want to add and click ok.

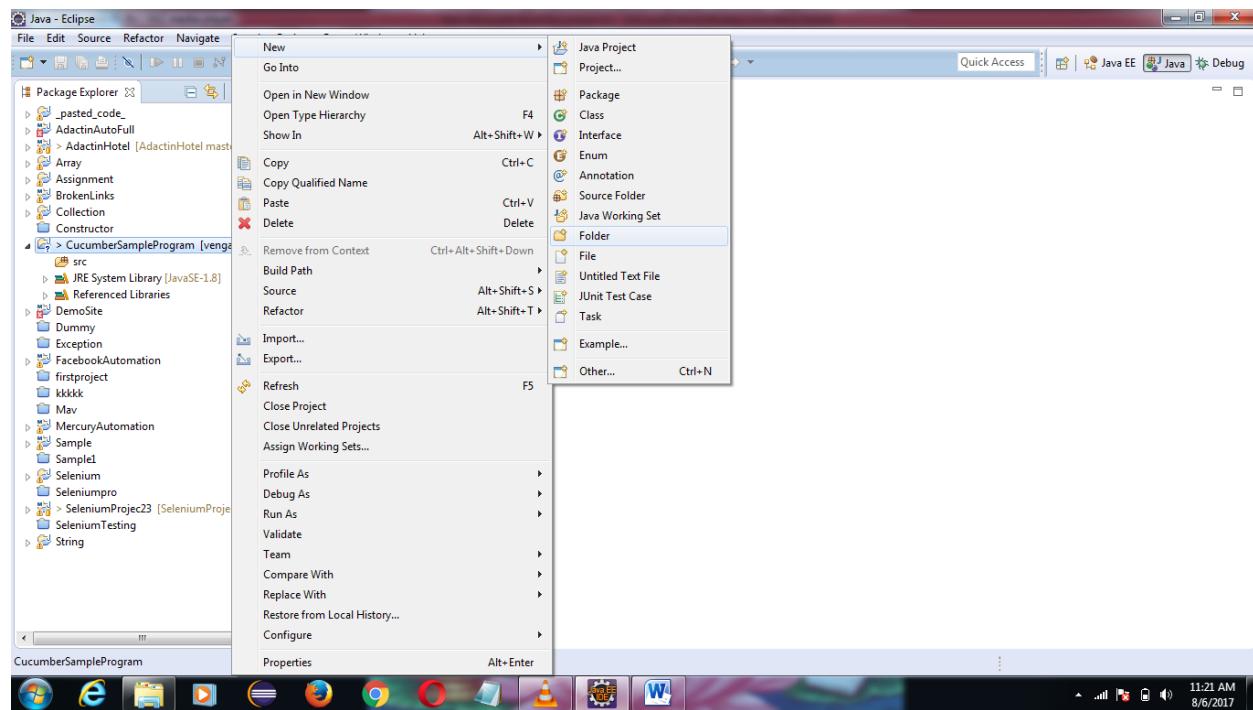


Download these jars:

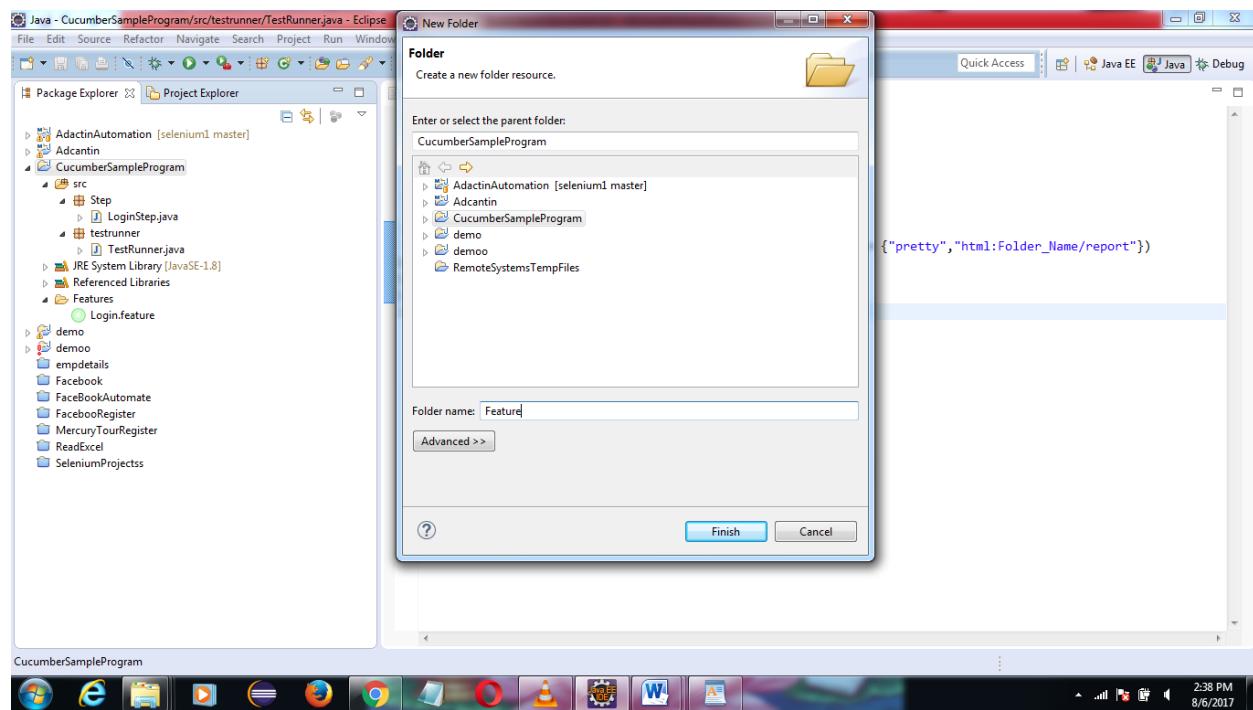


Step 5:

Create a new folder

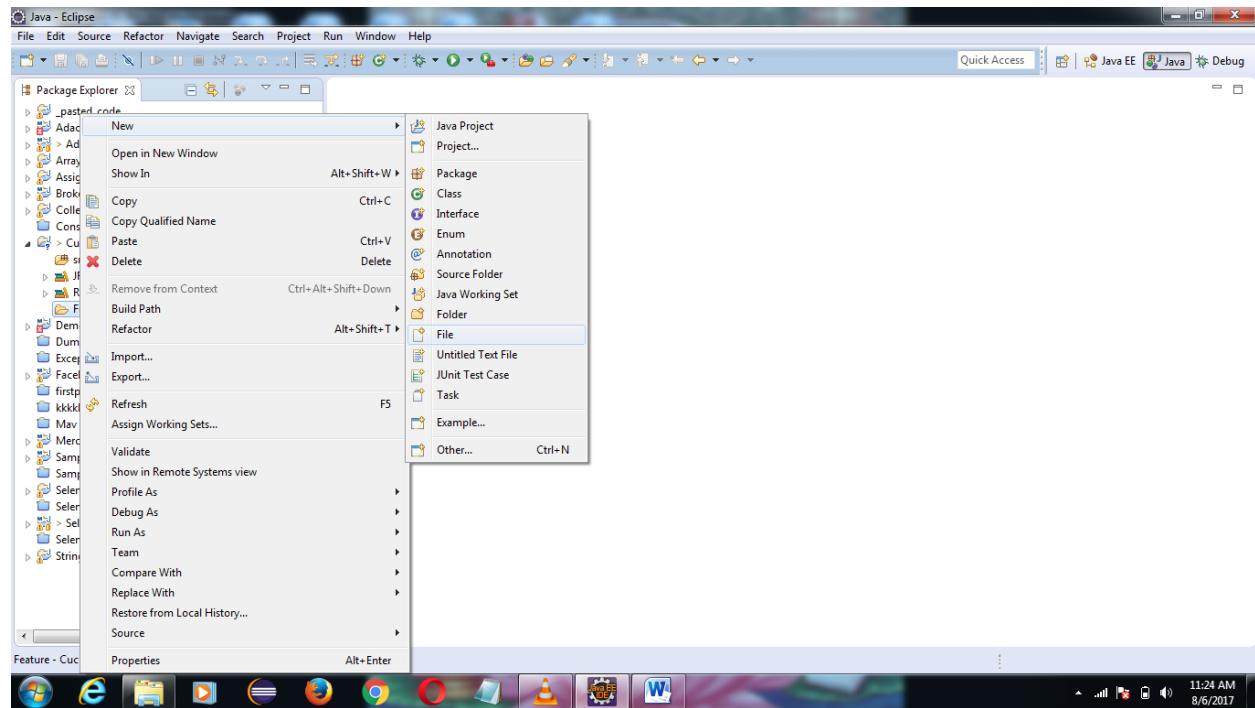


And enter the name as Feature and then click finish.

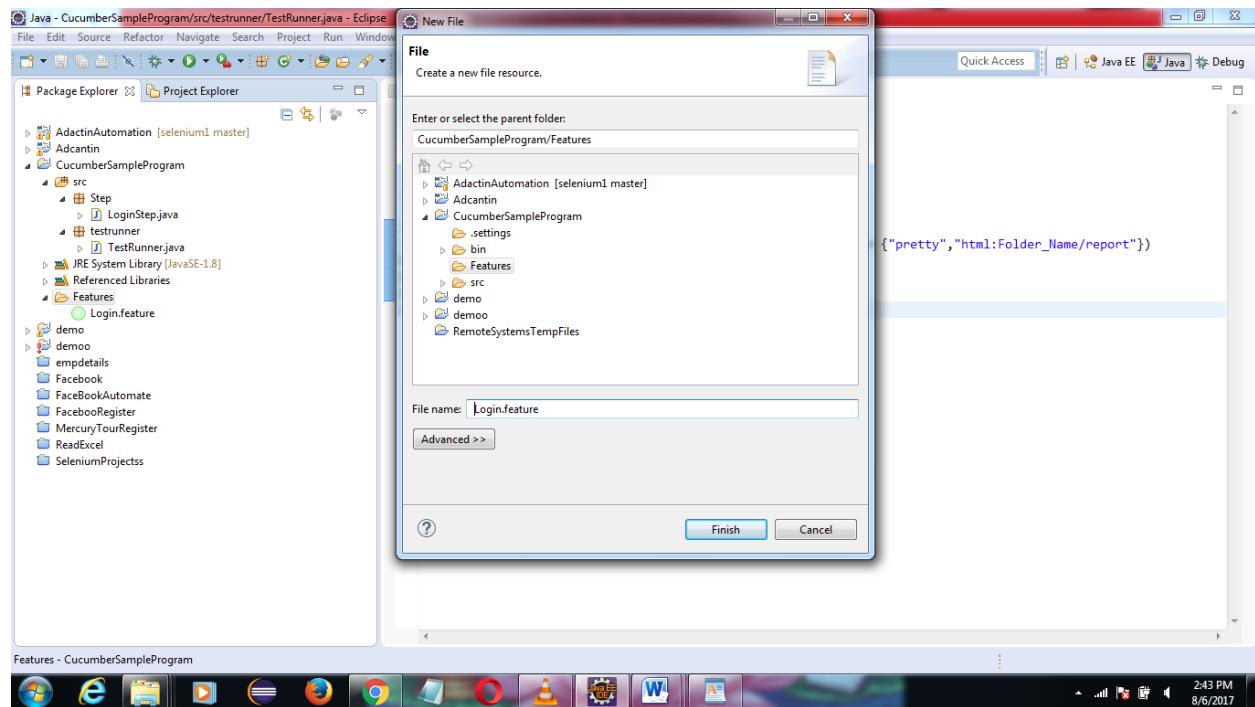


Step 6:

On the Feature folder we need to add File.

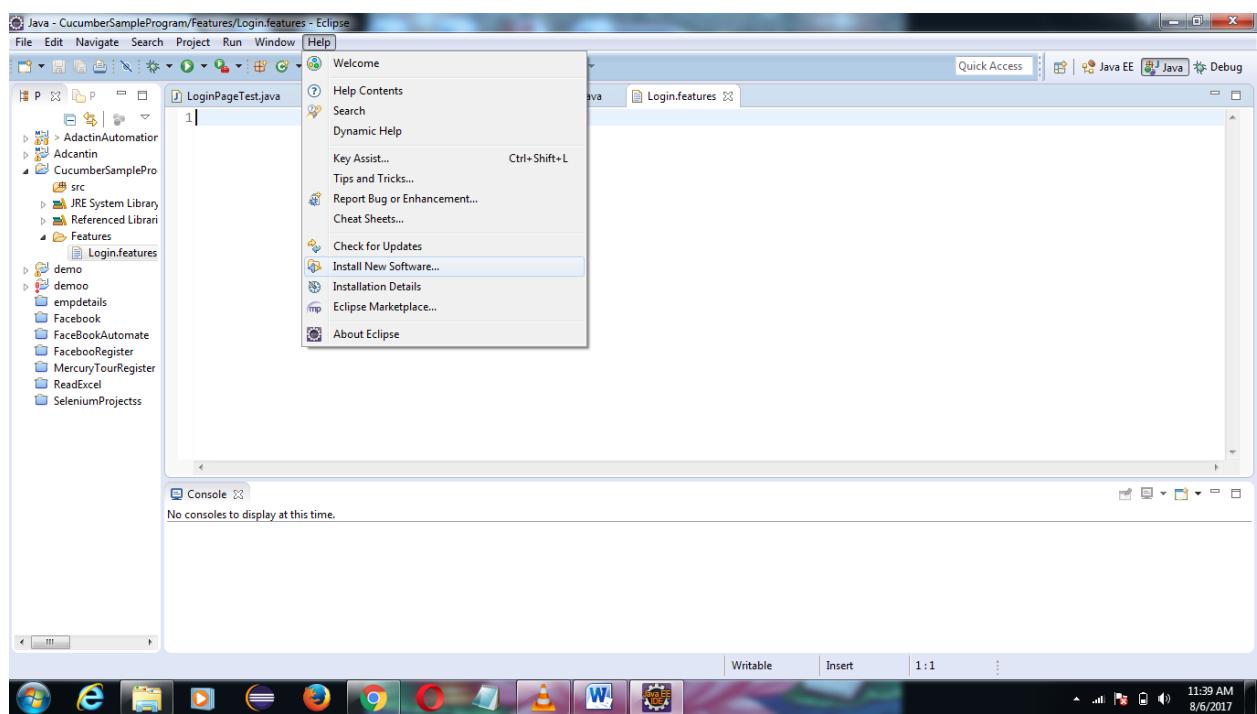
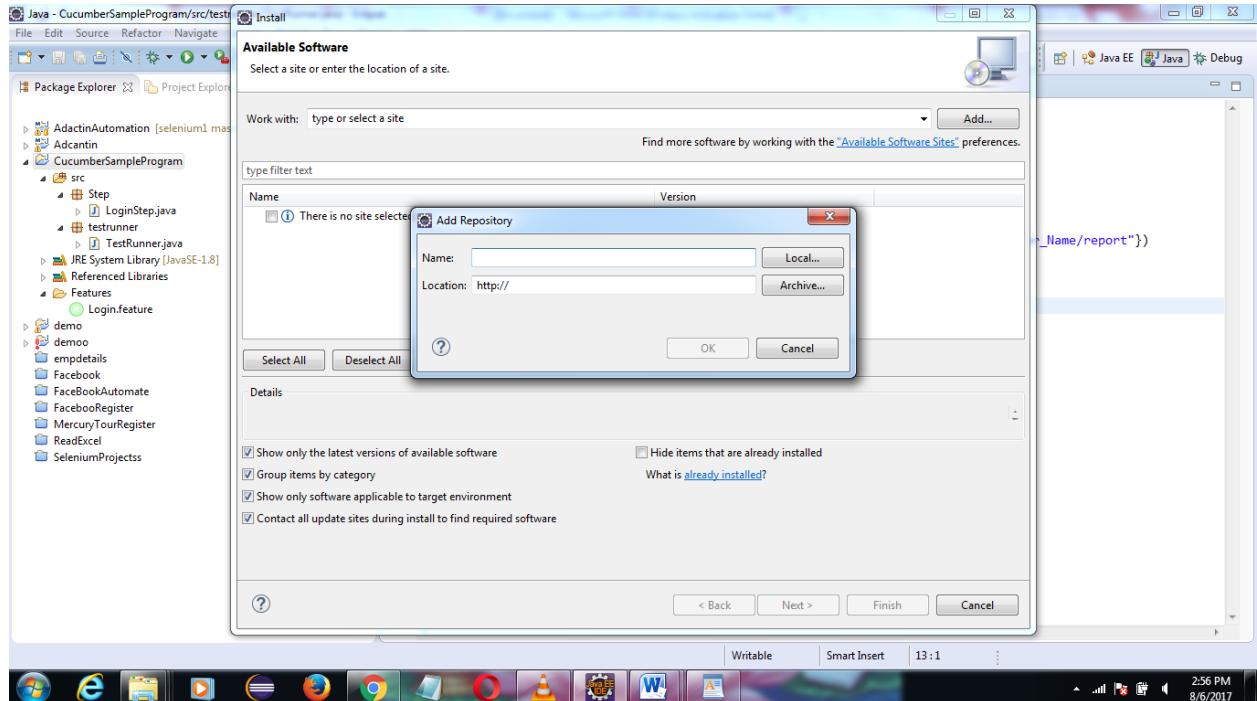


And the enter the file name it must ends with ".feature" and Click Finish.



Step 7: Configure in two ways:

1. Now , we need configure Gherkin plugin in eclipse through, Go to Market place →Install New Software

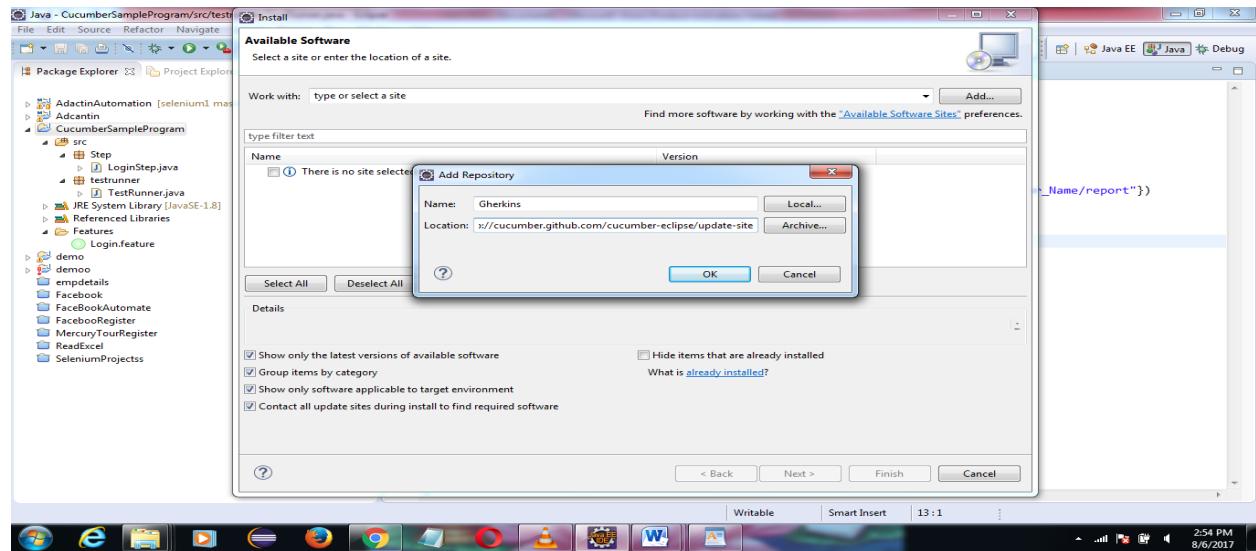


On that window click → add .

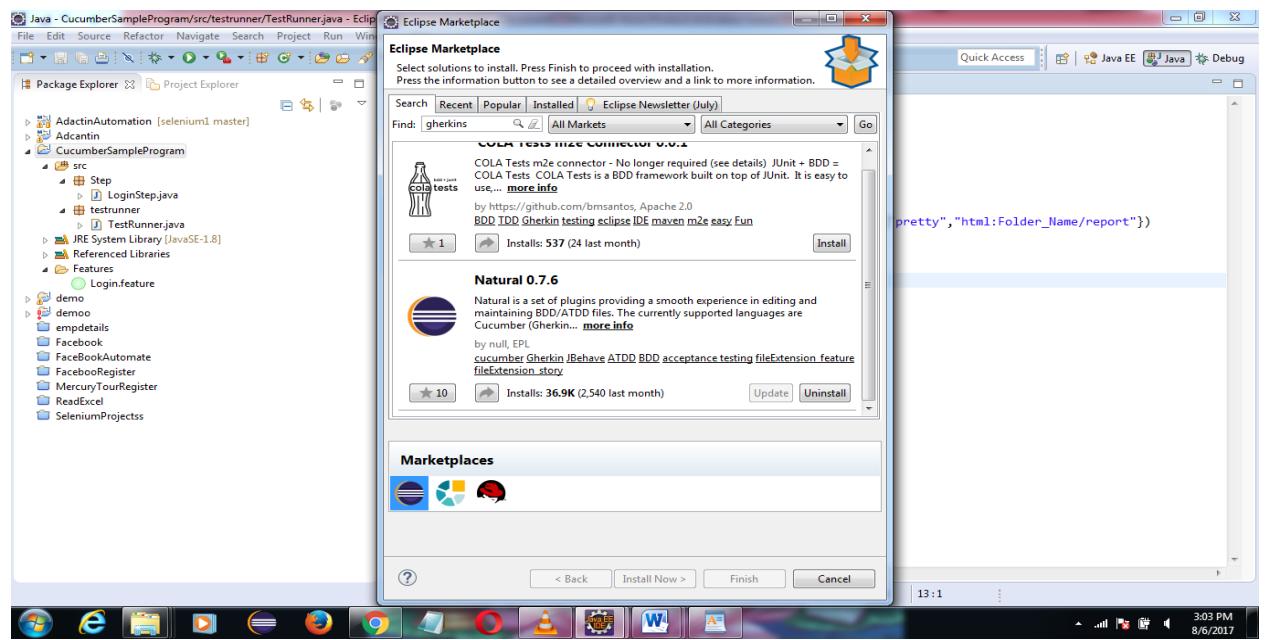
In that name: fill any name.

Address: use this address "<http://cucumber.github.com/cucumber-eclipse/update-site>"

And click ok → Accept user agreement → Download plugins



2. In help → Eclipse market place → search Gherkins → install Natural 0.7.6.



By this way you download Gherkins plugin for cucumber.

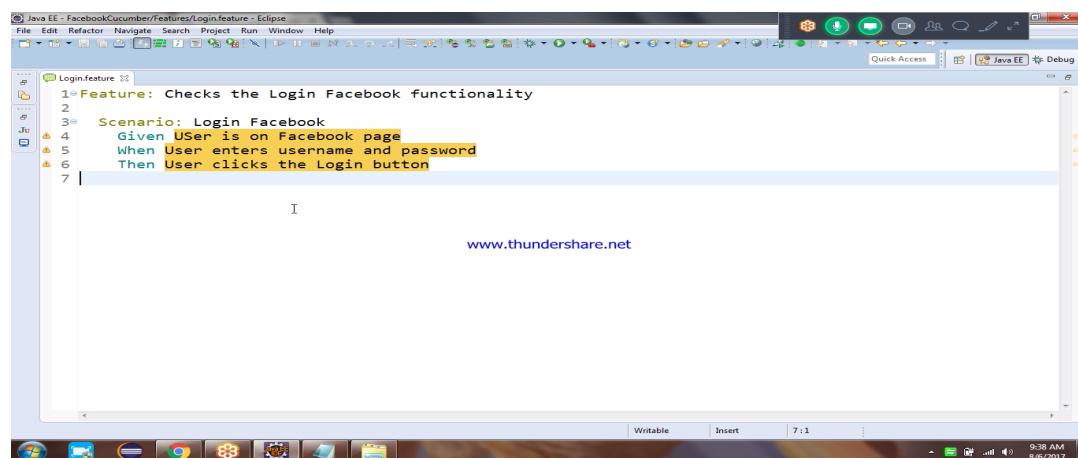
Step 8:

In our project ,we need to create three packages under src.

1. Testrunner → In this we add our test cases .
2. Steps → in this we create our methods .
3. pages → Here we add our Locators of web element and provide getters and setters.

Add future class on the file created on the Feature folder.

Create feature file like below



```
Java EE - FacebookCucumber/Features/Login.feature - Eclipse
File Edit Refactor Navigate Search Project Run Window Help
File Edit View Project Run Window Help
Login.feature:23
1 Feature: Checks the Login Facebook functionality
2
3 Scenario: Login Facebook
4 Given User is on Facebook page
5 When User enters username and password
6 Then User clicks the Login button
7 |
```

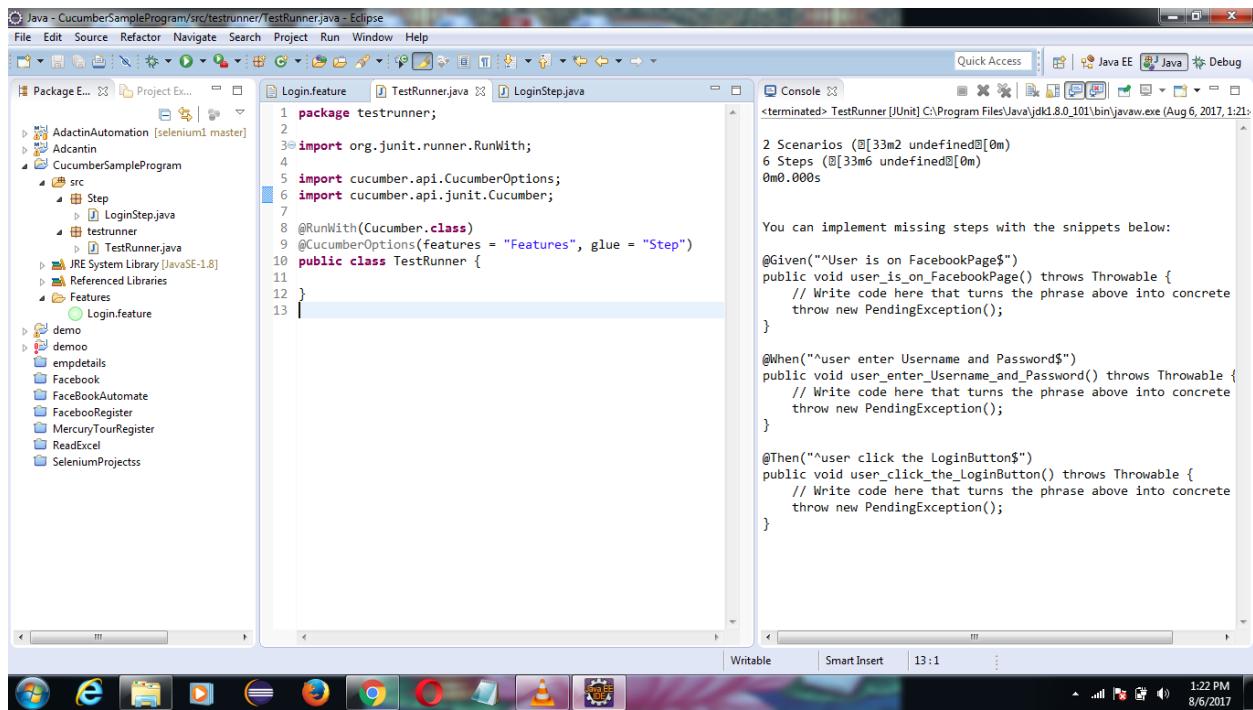
Package: testrunner

features:It contains Feature files.

Glue: It consists of steps of test runner class which obtained on console.

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step")
public class TestRunner {

}
```



After, we run the test case, we get console output use that steps for step package.

We create LoginSteps under the steps package and use that console output codes

Program of LoginStep class:

```
public class LoginStep {

    @Given("^User is on FacebookPage$")
    public void user_is_on_FacebookPage() throws Throwable {
        System.out.println("User is on FacebookPage");
    }

    @When("^user enter Username and Password$")
    public void user_enter_Username_and_Password() {
        System.out.println("user enter Username and Password");
    }

    @Then("^user click the LoginButton$")
    public void user_click_the_LoginButton() {
        System.out.println("user click the LoginButton");
    }

    @When("^user enter Details$")
    public void user_enter_Details() {
```

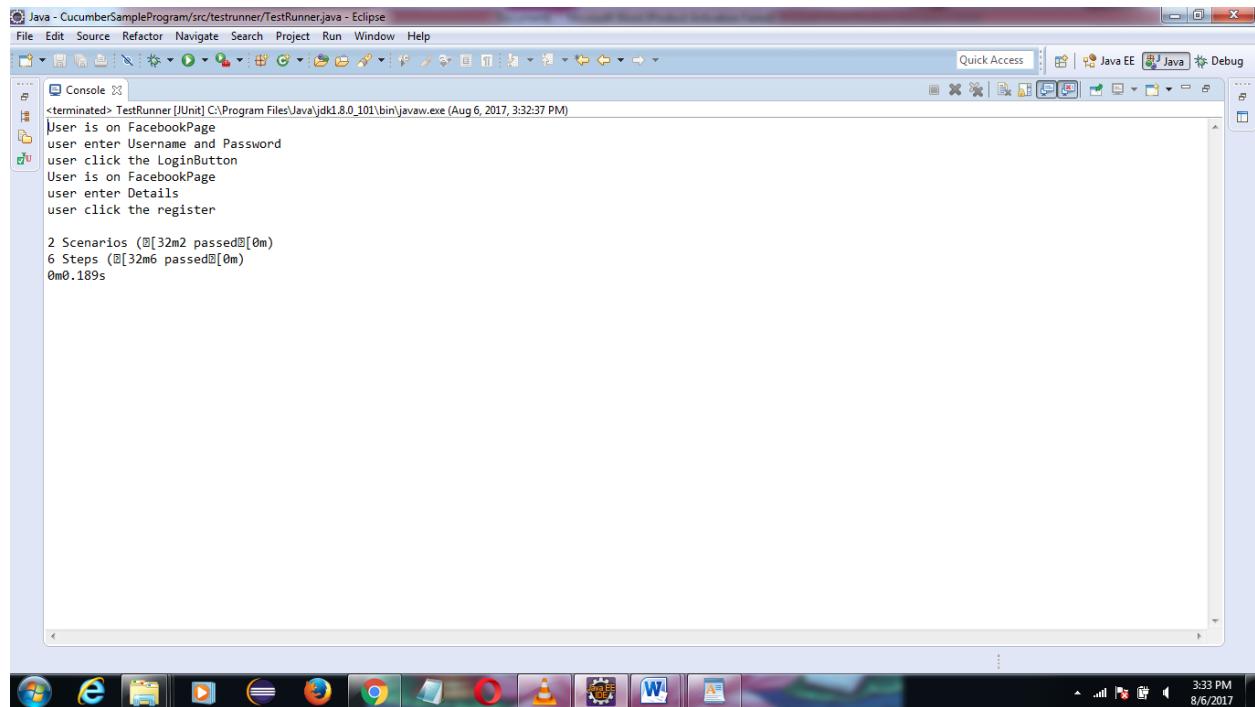
```

        System.out.println("user enter Details");
    }

@Then("^user click the register$")
public void user_click_the_register() {
    System.out.println("user click the register");
}

```

Output:



The screenshot shows the Eclipse IDE interface with the title bar "Java - CucumberSampleProgram/src/testrunner/TestRunner.java - Eclipse". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The status bar at the bottom right shows the date and time as "3:33 PM 8/6/2017". The main window displays the Eclipse Java perspective with the "Console" view selected. The console output shows the execution of a Cucumber test:

```

<terminated> TestRunner [JUnit] C:\Program Files\Java\jdk1.8.0_101\bin\javaw.exe (Aug 6, 2017, 3:32:37 PM)
User is on FacebookPage
user enter Username and Password
user click the LoginButton
User is on FacebookPage
User enter Details
user click the register

2 Scenarios (0m32s2 passed0m0s)
6 Steps (0m32s6 passed0m0s)
0m0.189s

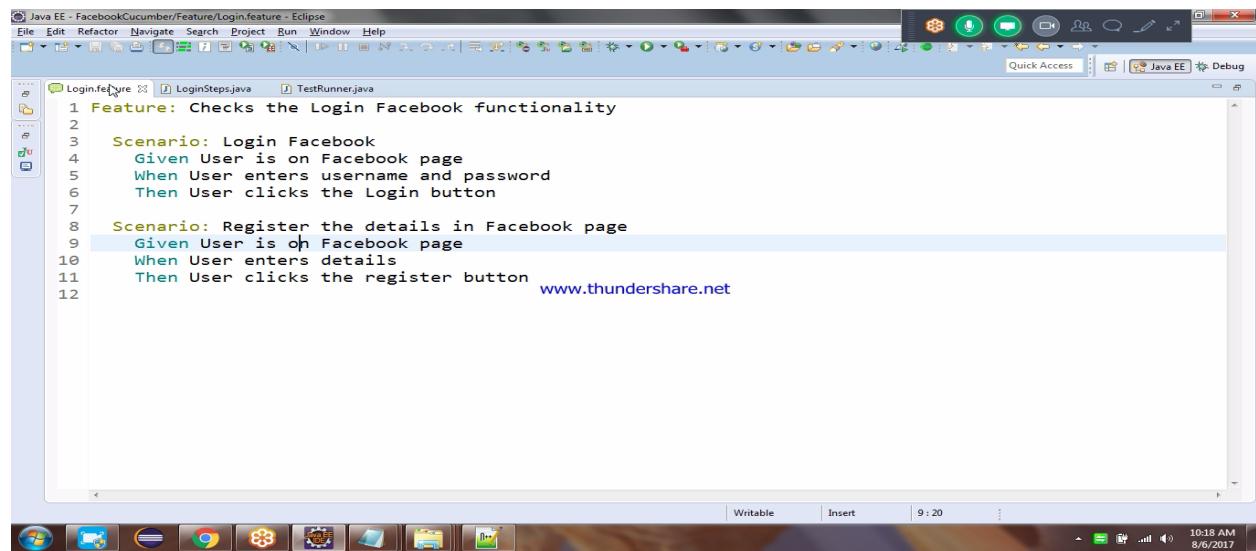
```

Formats of Report Using cucumber:

Cucumber supports various report formats. They are

1. HTML REPORT FORMAT:

Feature file:



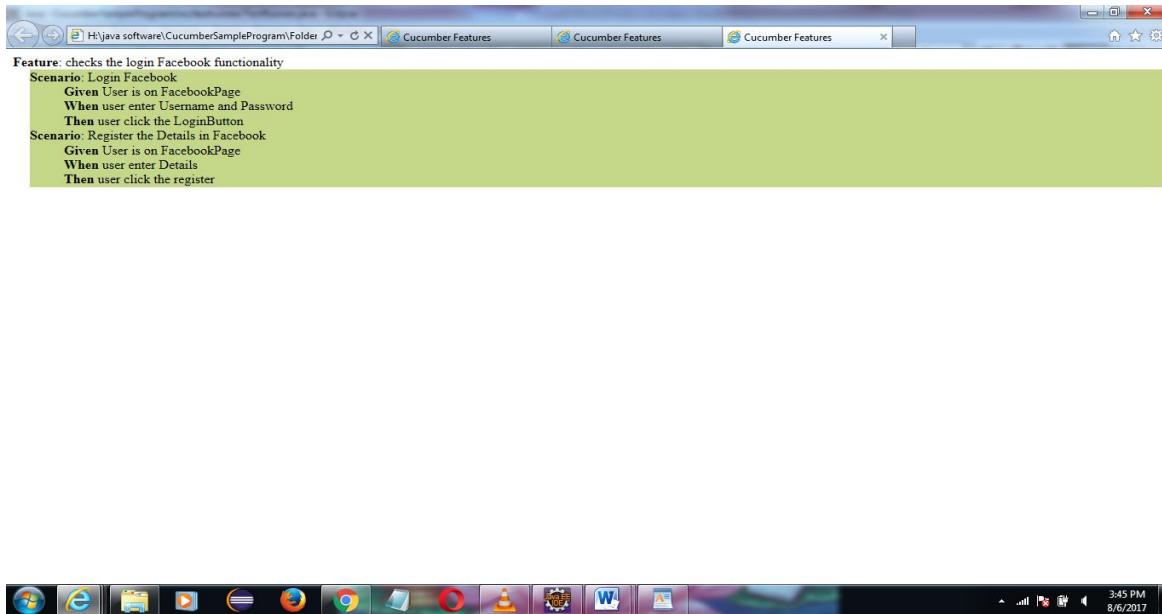
```
Java EE - FacebookCucumber/Feature/Login.feature - Eclipse
File Edit Refactor Navigate Search Project Run Window Help
Quick Access Java EE Debug
Login.feature [1] LoginSteps.java TestRunner.java
1 Feature: Checks the Login Facebook functionality
2
3 Scenario: Login Facebook
4   Given User is on Facebook page
5   When User enters username and password
6   Then User clicks the Login button
7
8 Scenario: Register the details in Facebook page
9   Given User is on Facebook page
10  When User enters details
11  Then User clicks the register button www.thundershare.net
12
```

Package: Test runner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step", plugin= {"pretty","html:Folder
name/report/cucumber"})
public class TestRunner {
```

```
}
```

Output:



2. JUNIT REPORT FORMAT:

Feature file:

```
Java EE - FacebookCucumber/Feature/Login.feature - Eclipse
File Edit Refactor Navigate Search Project Run Window Help
File Edit View Insert Run Project Properties Java EE Debug
Login.feature  LoginSteps.java  TestRunner.java
1 Feature: Checks the Login Facebook functionality
2
3 Scenario: Login Facebook
4   Given User is on Facebook page
5   When User enters username and password
6   Then User clicks the Login button
7
8 Scenario: Register the details in Facebook page
9   Given User is on Facebook page
10  When User enters details
11  Then User clicks the register button  www.thundershare.net
12
```

A screenshot of the Eclipse IDE interface. The title bar says "Java EE - FacebookCucumber/Feature/Login.feature - Eclipse". The main area shows a feature file named "Login.feature" with its contents. The code is color-coded: "Feature" and "Scenario" are in blue, "Given", "When", and "Then" are in red, and URLs are in purple. The Eclipse toolbar and menu bar are visible at the top. The status bar at the bottom shows the date and time.

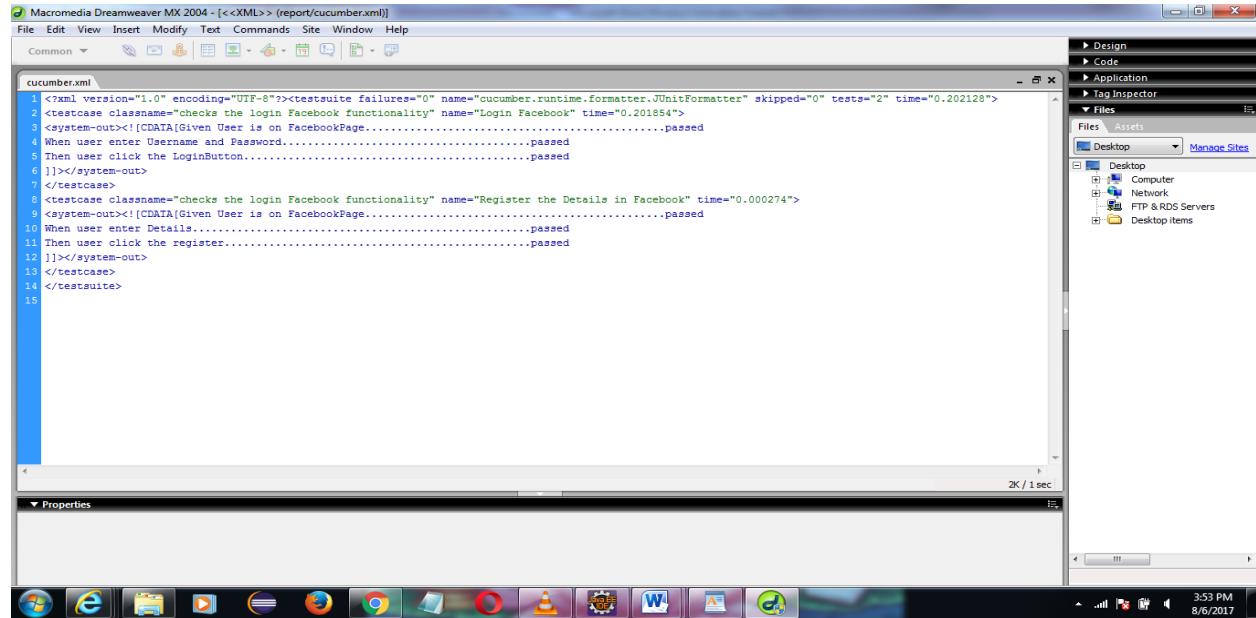
Package: Test runner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step", plugin= {"pretty","junit:Folder
name/report/cucumber.xml"})
```

```
public class TestRunner {
```

```
}
```

Output:

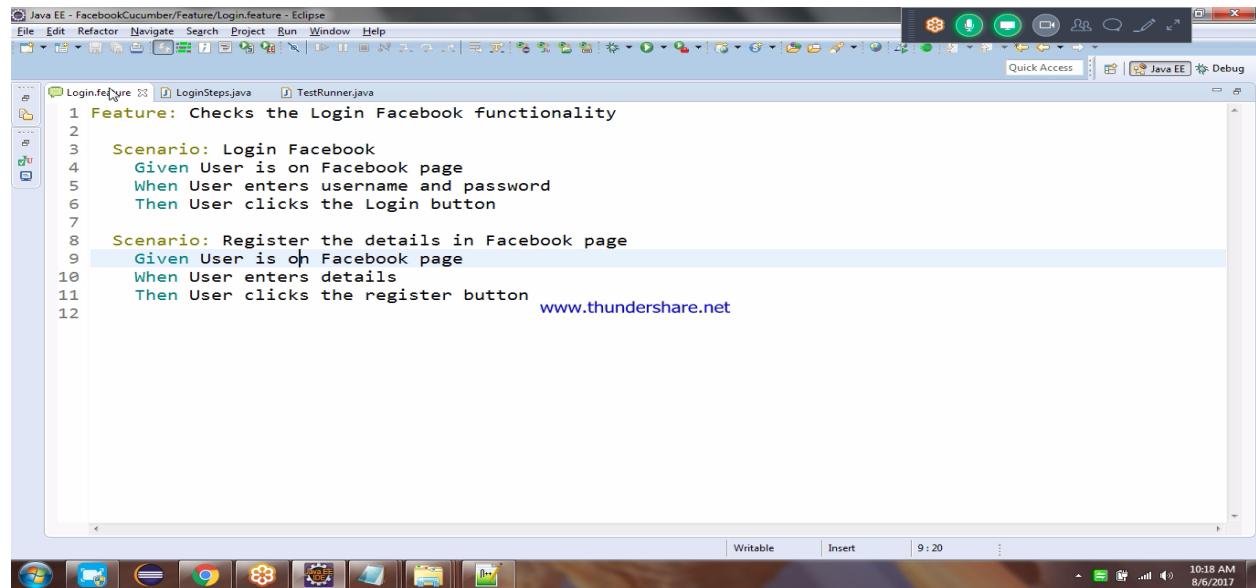


The screenshot shows the Macromedia Dreamweaver MX 2004 interface with the file "cucumber.xml" open. The XML content details test cases for Facebook login and registration functionality. The interface includes a menu bar, toolbars, and various panels like Design, Code, Application, Tag Inspector, and Files.

```
<?xml version="1.0" encoding="UTF-8"?><testsuite failures="0" name="cucumber.runtime.formatter.JUnitFormatter" skipped="0" tests="2" time="0.202128">
  < testcase classname="checks the login Facebook functionality" name="Login Facebook" time="0.201854">
    <system-out><![CDATA[Given User is on FacebookPage.....passed
When user enter Username and Password.....passed
Then user click the LoginButton.....passed
]]></system-out>
  </ testcase >
  < testcase classname="checks the login Facebook functionality" name="Register the Details in Facebook" time="0.000274">
    <system-out><![CDATA[Given User is on FacebookPage.....passed
When user enter Details.....passed
Then user click the register.....passed
]]></system-out>
  </ testcase >
</ testsuite >
```

3. JSON TYPE REPORT:

Feature file:



The screenshot shows the Eclipse IDE interface with the file "Login.feature" open. It contains a feature definition for "Checks the Login Facebook functionality" with scenarios for logging in and registering details. The Eclipse toolbar and various icons are visible at the bottom.

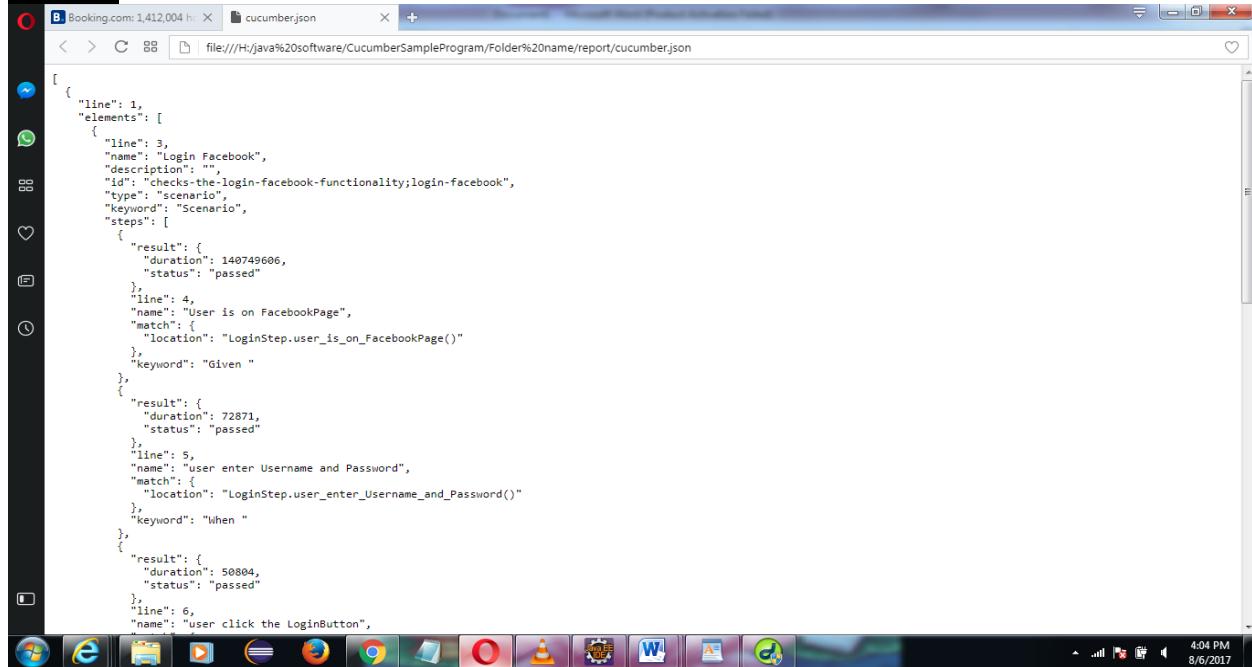
```
1 Feature: Checks the Login Facebook functionality
2
3   Scenario: Login Facebook
4     Given User is on Facebook page
5     When User enters username and password
6     Then User clicks the Login button
7
8   Scenario: Register the details in Facebook page
9     Given User is on Facebook page
10    When User enters details
11    Then User clicks the register button      www.thundershare.net
12
```

Package: Test runner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step", plugin= {"pretty","json:Folder
name/report/cucumber.json"})
public class TestRunner {
```

}

Output:



```
[{"line": 1, "elements": [{"line": 3, "name": "login Facebook", "description": "", "id": "checks-the-login-facebook-functionality;login-facebook", "type": "scenario", "keyword": "Scenario", "steps": [{"result": {"duration": 140749606, "status": "passed"}, "line": 4, "name": "User is on FacebookPage", "match": {"location": "LoginStep.user_is_on_FacebookPage()"}, "keyword": "Given "}, {"result": {"duration": 72871, "status": "passed"}, "line": 5, "name": "user enter Username and Password", "match": {"location": "LoginStep.user_enter_Username_and_Password()"}, "keyword": "When "}, {"result": {"duration": 50804, "status": "passed"}, "line": 6, "name": "user click the LoginButton", "match": {"location": "LoginStep.user_click_LoginButton()"}, "keyword": "Then "}]}, {"line": 7, "elements": [{"line": 8, "name": "Scenario: login Facebook", "description": "checks the login facebook functionality", "id": "checks-the-login-facebook-functionality;login-facebook", "type": "scenario", "keyword": "Scenario", "steps": [{"line": 9, "name": "Given User is on FacebookPage", "description": "", "id": "User is on FacebookPage", "type": "step", "keyword": "Given", "result": {"duration": 140749606, "status": "passed"}, "match": {"location": "LoginStep.user_is_on_FacebookPage()"}, "locationName": "LoginStep.user_is_on_FacebookPage()", "locationType": "StepDefinition"}, {"line": 10, "name": "When user enter Username and Password", "description": "", "id": "user enter Username and Password", "type": "step", "keyword": "When", "result": {"duration": 72871, "status": "passed"}, "match": {"location": "LoginStep.user_enter_Username_and_Password()"}, "locationName": "LoginStep.user_enter_Username_and_Password()", "locationType": "StepDefinition"}, {"line": 11, "name": "Then user click the LoginButton", "description": "", "id": "user click the LoginButton", "type": "step", "keyword": "Then", "result": {"duration": 50804, "status": "passed"}, "match": {"location": "LoginStep.user_click_LoginButton()"}, "locationName": "LoginStep.user_click_LoginButton()", "locationType": "StepDefinition"}]}], "line": 12, "elements": [{"line": 13, "name": "Feature: login Facebook", "description": "checks the login facebook functionality", "id": "checks-the-login-facebook-functionality", "type": "feature", "keyword": "Feature", "steps": [{"line": 14, "name": "Scenario: login Facebook", "description": "checks the login facebook functionality", "id": "checks-the-login-facebook-functionality;login-facebook", "type": "scenario", "keyword": "Scenario", "steps": [{"line": 15, "name": "Given User is on FacebookPage", "description": "", "id": "User is on FacebookPage", "type": "step", "keyword": "Given", "result": {"duration": 140749606, "status": "passed"}, "match": {"location": "LoginStep.user_is_on_FacebookPage()"}, "locationName": "LoginStep.user_is_on_FacebookPage()", "locationType": "StepDefinition"}, {"line": 16, "name": "When user enter Username and Password", "description": "", "id": "user enter Username and Password", "type": "step", "keyword": "When", "result": {"duration": 72871, "status": "passed"}, "match": {"location": "LoginStep.user_enter_Username_and_Password()"}, "locationName": "LoginStep.user_enter_Username_and_Password()", "locationType": "StepDefinition"}, {"line": 17, "name": "Then user click the LoginButton", "description": "", "id": "user click the LoginButton", "type": "step", "keyword": "Then", "result": {"duration": 50804, "status": "passed"}, "match": {"location": "LoginStep.user_click_LoginButton()"}, "locationName": "LoginStep.user_click_LoginButton()", "locationType": "StepDefinition"}]}]}]
```

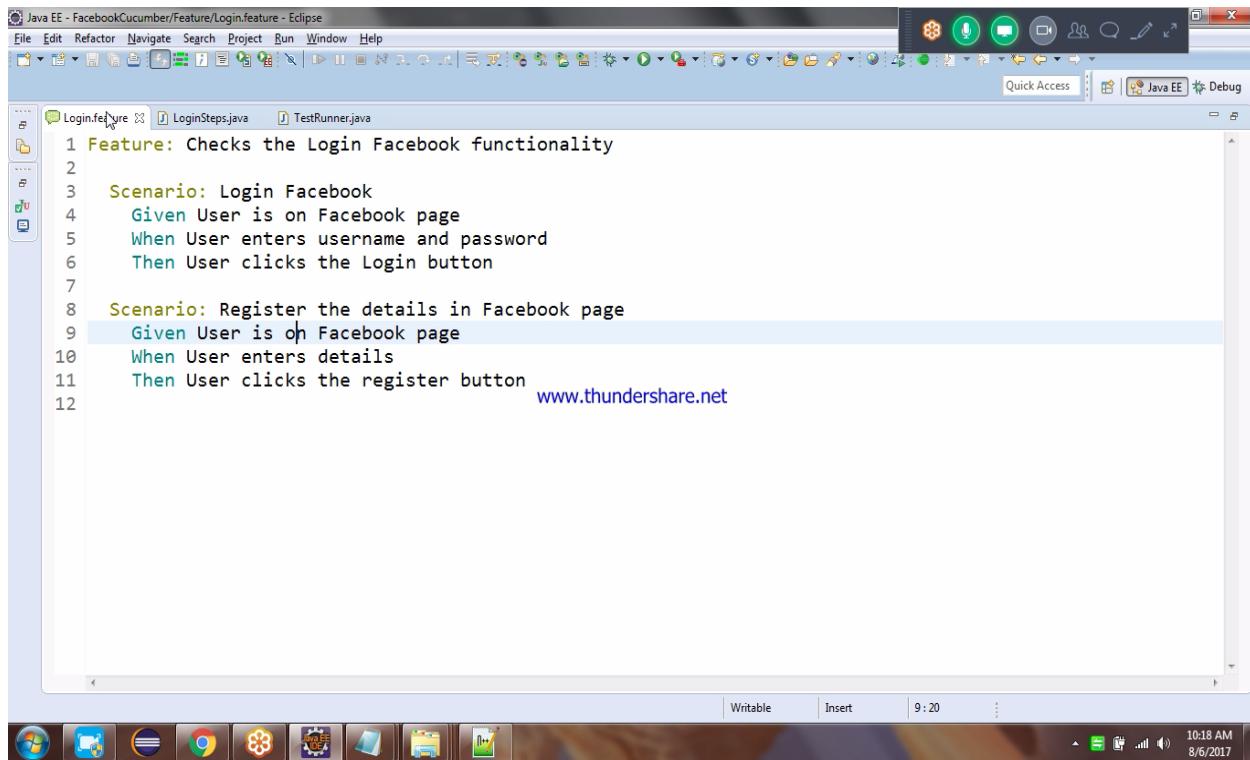
DRY RUN:

- This is used for check our formats on the Featured file.

dryRun=true → Validate our Featured file.

dryRun=false → Not Validate our featured file

Feature file:



The screenshot shows the Eclipse IDE interface with the title bar "Java EE - FacebookCucumber/Feature/Login.feature - Eclipse". The main editor area displays the content of the "Login.feature" file. The file contains the following Cucumber steps:

```
1 Feature: Checks the Login Facebook functionality
2
3 Scenario: Login Facebook
4   Given User is on Facebook page
5   When User enters username and password
6   Then User clicks the Login button
7
8 Scenario: Register the details in Facebook page
9   Given User is on Facebook page
10  When User enters details
11  Then User clicks the register button
```

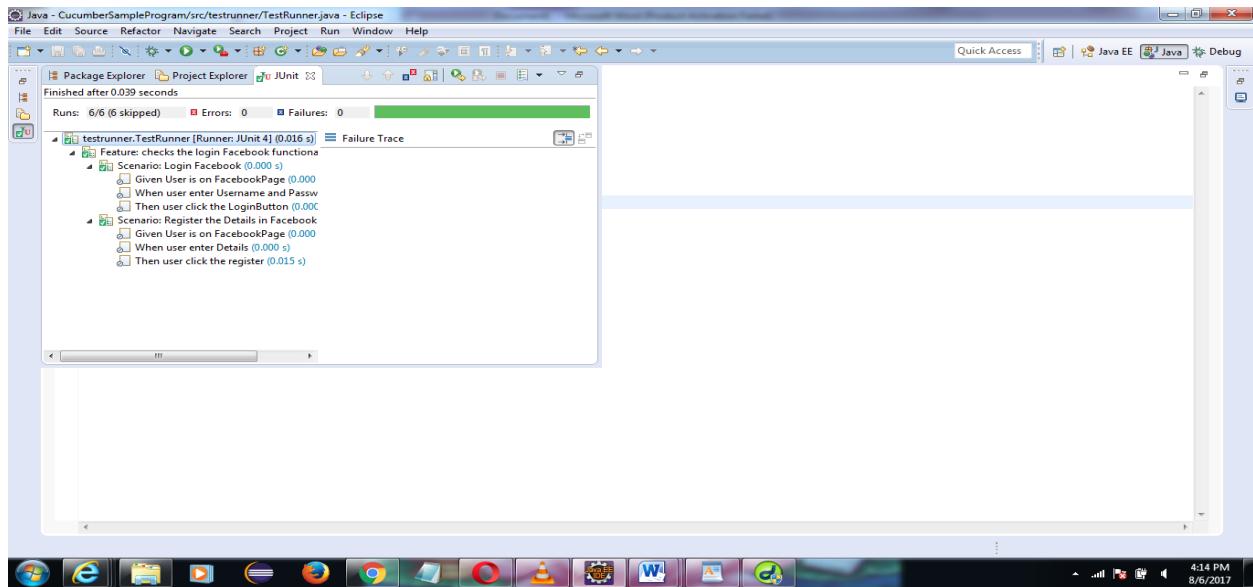
The step "Given User is on Facebook page" at line 9 is highlighted with a blue selection bar. The status bar at the bottom right shows the date and time as "8/6/2017 10:18 AM".

Package: Test runner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step", dryRun=True)
public class TestRunner {

}
```

Output:



MONOCHROME:

- It gives report on console.
- monochrome = true → It a report.
- monochrome = false → It not display a report on console.

Feature file:

The screenshot shows the Eclipse IDE interface with the title bar "Java EE - FacebookCucumber/Feature/Login.feature - Eclipse". The "Text Editor" view displays the content of the "Login.feature" file. The file contains the following Cucumber steps:

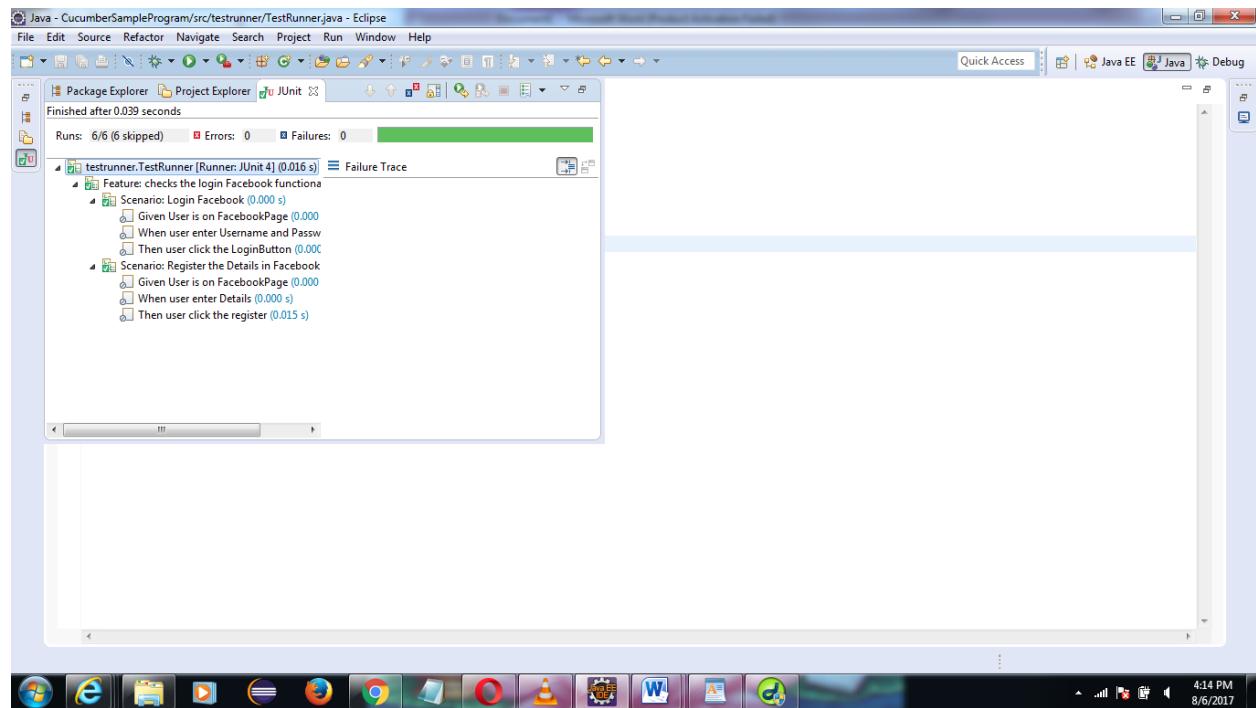
```
1 Feature: Checks the Login Facebook functionality
2
3 Scenario: Login Facebook
4   Given User is on Facebook page
5   When User enters username and password
6   Then User clicks the Login button
7
8 Scenario: Register the details in Facebook page
9   Given User is on Facebook page
10  When User enters details
11  Then User clicks the register button www.thundershare.net
```

Package: Test runner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step", dryRun=True)
public class TestRunner {

}
```

Output:

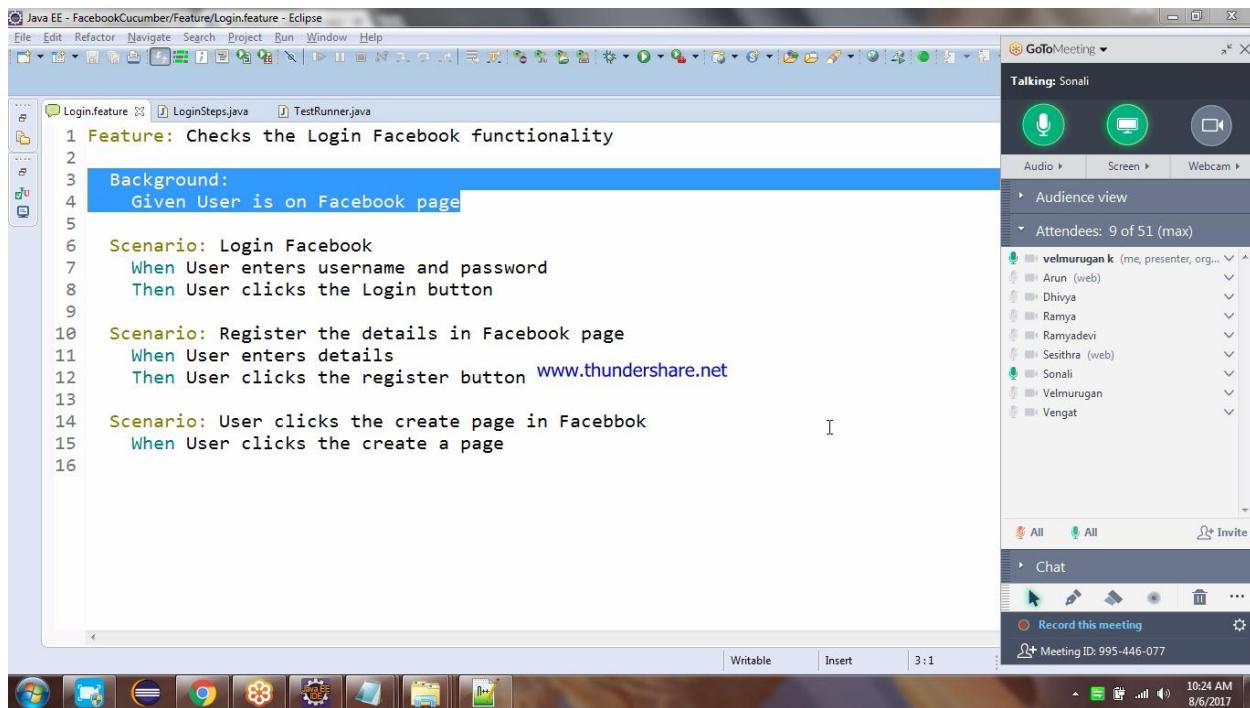


BACKGROUND:

- It's a keyword used in featured class for declare the common step as once.

Output:

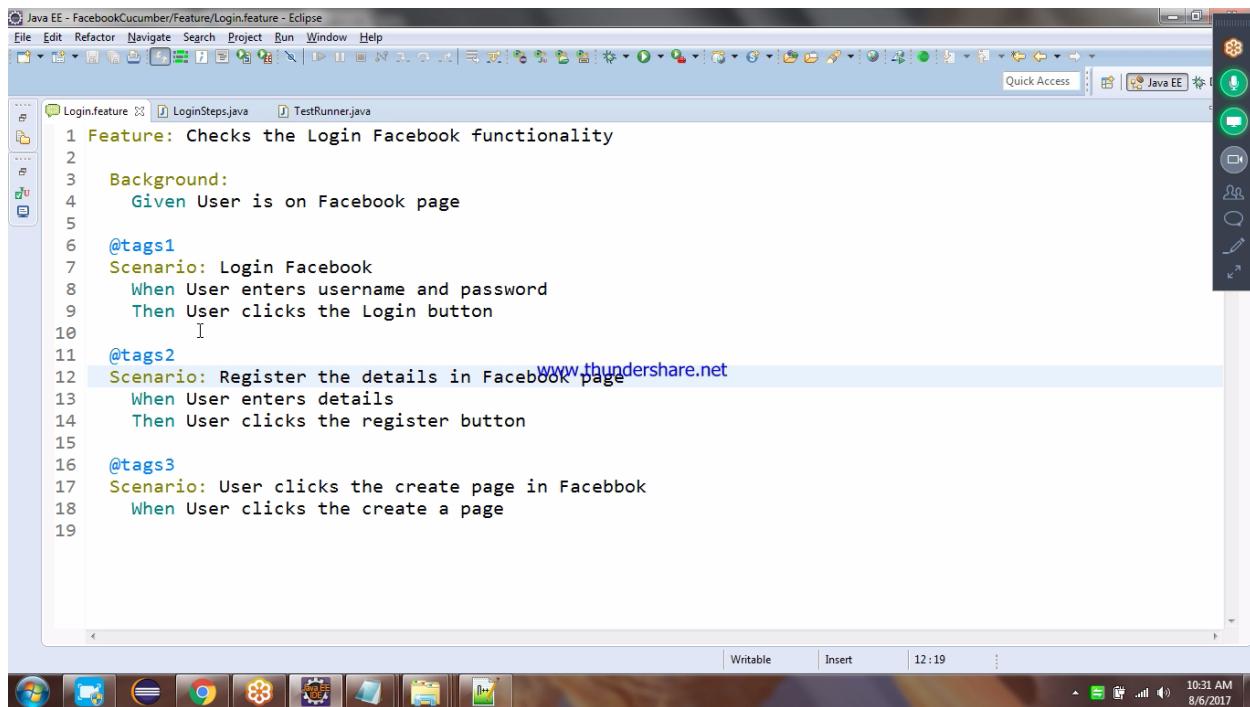
Here “Given” is common for all scenarios ...so we declare under the background keyword.



TAGS:

- It's a command which is used for execute a particular scenario.

Feature file:

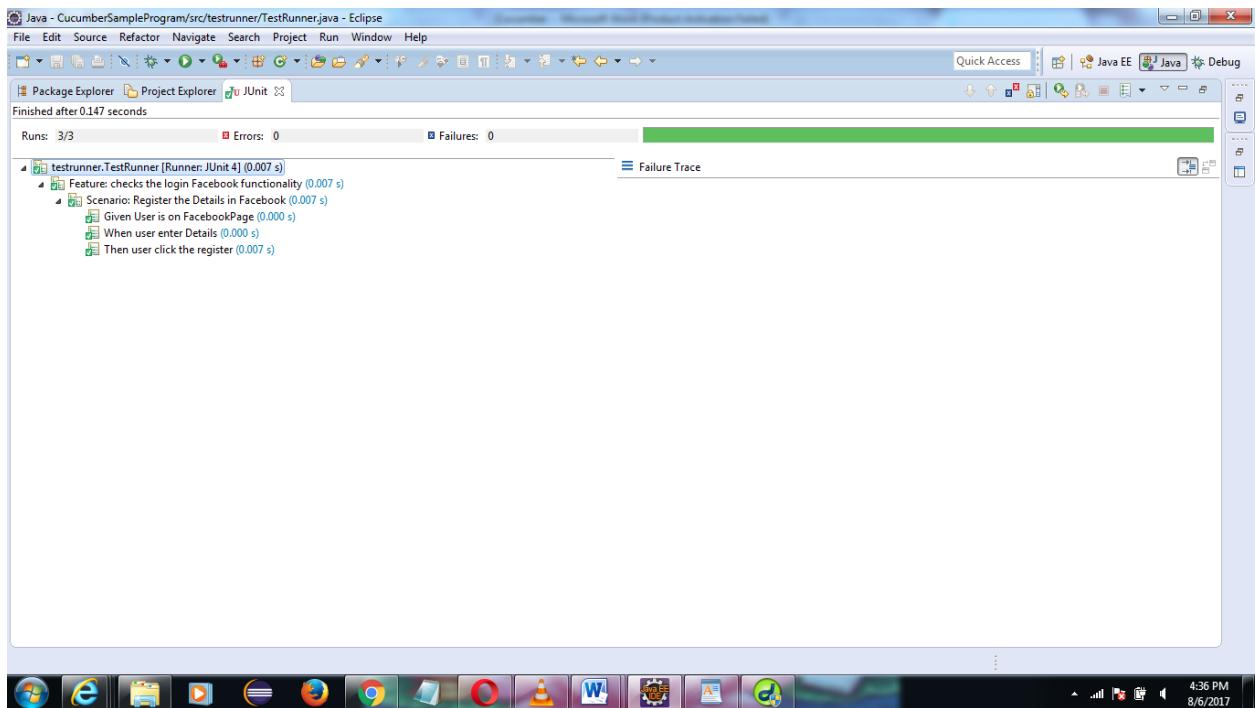


Package: Test runner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "Step", dryRun=True,
tags={"@tags1"})
public class TestRunner {

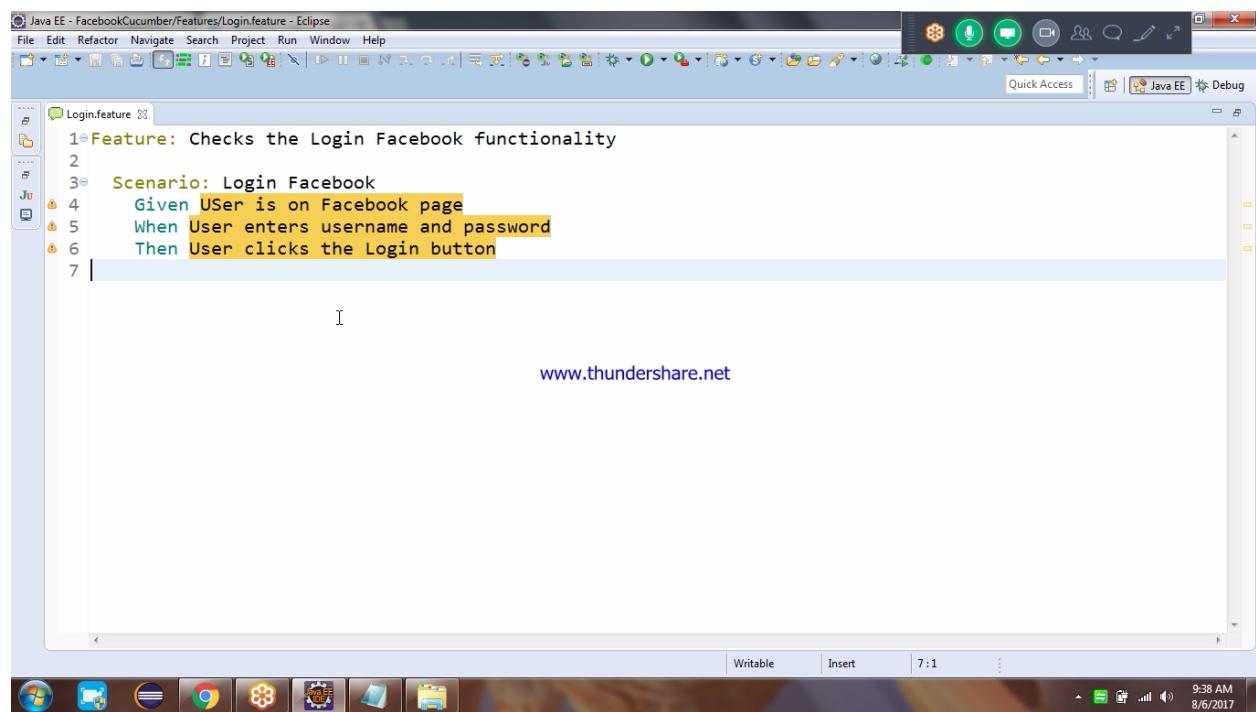
}
```

Output:



FB LOGIN USING CUCUMBER:

Feature file:



```
Java EE - FacebookCucumber/Features/Login.feature - Eclipse
File Edit Refactor Navigate Search Project Run Window Help
Quick Access Java EE Debug

Login.feature
1 Feature: Checks the Login Facebook functionality
2
3 Scenario: Login Facebook
4   Given User is on Facebook page
5   When User enters username and password
6   Then User clicks the Login button
7

www.thundershare.net
```

Package: testrunner

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Feature", glue = "steps")
public class TestRunner {
```

```
}
```

Package: steps

```
public class LoginSteps {
    WebDriver driver;
    @Given("^User is on Facebook page$")
    public void user_is_on_Facebook_page() {
        System.setProperty("webdriver.gecko.driver", "C:/Users/mohan
pc/Desktop/CucumberProgram	driver/geckodriver.exe");
        driver = new FirefoxDriver();
```

```

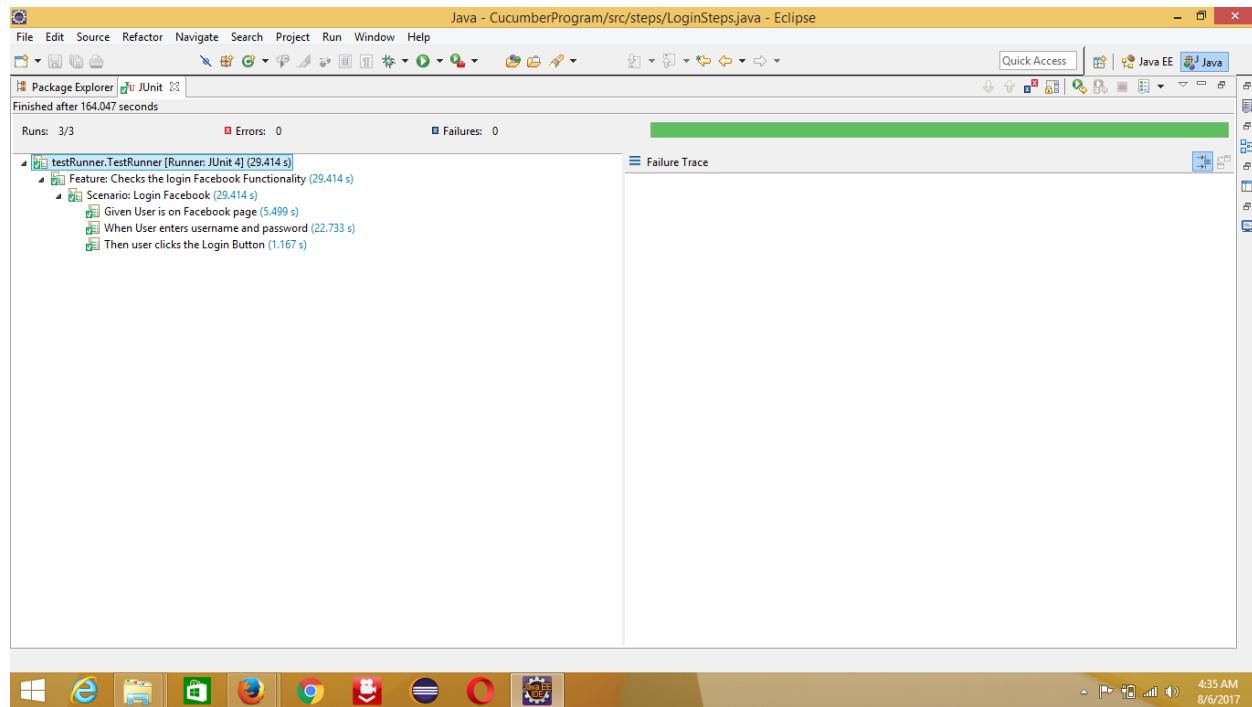
        driver.get("https://www.facebook.com/");
    }

    @When("^User enters username and password$")
    public void user_enters_username_and_password() {
        driver.findElement(By.id("email")).sendKeys("mohanrajeee6@gmail.com");
        driver.findElement(By.id("pass")).sendKeys("mohanrajeee6@gmail.com");
    }

    @Then("^user clicks the Login Button$")
    public void user_clicks_the_Login_Button() {
        driver.findElement(By.id("u_0_r")).click();
    }
}

```

Output:



ENTERING VALUE IN FEATURE FILE:

Feature file:

Feature: check the login adactin functionality

Scenario: Login Adactin

Given User is an adactin page

When User enter "vengat16" and "Karthick" and click login button

Then Message displayed Login Successfully

Step definition:

```
public class LoginTest {  
    WebDriver driver;  
  
    @Given("^User is an adactin page$")  
    public void user_is_an_adactin_page() {  
        System.setProperty("webdriver.gecko.driver",  
                           "H:\\java software\\CucumberSample1\\driver\\geckodriver.exe");  
        driver = new FirefoxDriver();  
        driver.get("http://www.adactin.com/HotelApp/");  
    }  
  
    @When("^User enter \"([^\"]*)\" and \"([^\"]*)\" and click login button$")  
    public void user_enter_UserName_and_Password(String Username,  
                                                String Password) {  
        driver.findElement(By.id("username")).sendKeys(Username);  
        ;  
        driver.findElement(By.id("password")).sendKeys>Password);  
        ;  
        driver.findElement(By.id("login")).click();  
    }  
  
    @Then("^Message displayed Login Successfully$")  
    public void message_displayed_Login_Successfully() {  
        driver.quit();  
    }  
}
```

Test runner class:

```
@RunWith(Cucumber.class)  
@CucumberOptions(features = "Feature", glue =  
"step", plugin = { "pretty", "html:Report/cucumber" })  
public class TestRunner {  
}
```

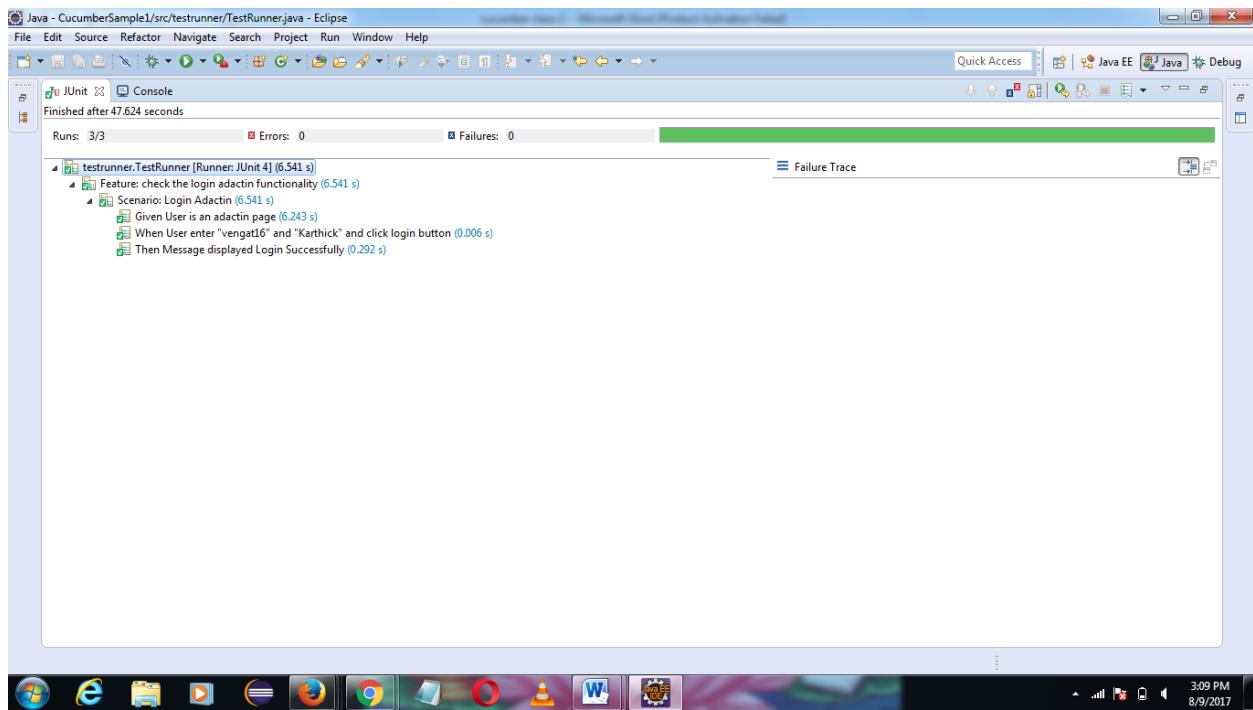
Output:

The screenshot shows the Eclipse IDE interface. The left sidebar displays the 'Package Explorer' with a project named 'AdactinAutomation [selenium1 master]'. Inside this project, there are several packages and files: 'src' (containing 'step', 'LoginTest.java', 'testrunner', 'TestRunner.java'), 'JRE System Library [JavaSE-1.8]', 'Referenced Libraries', 'driver', 'Feature' (containing 'Login.feature'), 'Report', 'CucumberSampleProgram', 'demo', 'demos', 'empdetails', 'Facebook', 'FaceBookAutomate', 'FacebooRegister', 'MercuryTourRegister', 'ReadExcel', and 'SeleniumProjectss'. The main editor window shows the content of 'TestRunner.java':

```
1 package testrunner;
2
3 import org.junit.runner.RunWith;
4 import cucumber.api.CucumberOptions;
5 import cucumber.api.junit.Cucumber;
6
7 @RunWith(Cucumber.class)
8 @CucumberOptions(features = "Feature", glue = "step", plugin={"pretty", "html:Report/cucumber"})
9 public class TestRunner {
10
11 }
```

Below the editor is a 'JUnit' view showing the execution of 'TestRunner' with the message: 'Feature: check the login adactin functionality'. The console output shows log entries from 'geckodriver' and 'marionette' starting the browser.

The screenshot shows a web browser displaying the Adactin.com website. The URL is 'www.adactin.com/Hotel/App/SearchHotel.php'. The page header features the Adactin logo and the tagline 'Innovation in Testing'. The main content area is titled 'Search Hotel' and contains fields for location, hotel, room type, number of rooms, check-in date, check-out date, adults per room, and children per room. Below these fields are 'Search' and 'Reset' buttons. To the right of the form, there are sections for 'Sample TestCases' (with a link to download sample test cases), 'HotelApp Web Services' (with a link to learn about web services testing), 'Known Defects' (with a link to download a list of known defects), and 'Book on Test Automation' (with a link to a guide on test automation using Microsoft Coded UI). The status bar at the bottom of the browser window shows the URL 'www.adactin.com/our-products/bookspublications/test-automation-using-microsoft-coded-ui-with-c/' and the time '3:08 PM 8/9/2017'.



To check multiple values:

- Using this method we can check the login function with multiple values without using iteration
- This is the easy way to check multiple values
In this method we use “Examples” one keyword in feature file.
- If we use “Examples” , we must give “Scenario Outline” instead of “Scenario”

Feature file:

Feature: check the login adactin functionality

Scenario Outline: Login Adactin

Given User is an adactin page

When User enter "<UserName>" and "<Password>" and click login button

Then Message displayed Login Successfully

Examples:

UserName	Password
vengat16	Karthick
ezhilarasan	12345

Step definition :

```
public class LoginTest {
    WebDriver driver;

    @Given("^User is an adactin page$")
    public void user_is_an_adactin_page() {
        System.setProperty("webdriver.gecko.driver",
                           "H:\\\\java
software\\\\CucumberSample1\\\\driver\\\\geckodriver.exe");
        driver = new FirefoxDriver();
        driver.get("http://www.adactin.com/HotelApp/");
    }

    @When("^User enter \"([^\"]*)\" and \"([^\"]*)\" and click login
button$")
    public void user_enter_UserName_and_Password(String Username,
                                                String Password) {
        driver.findElement(By.id("username")).sendKeys(Username);

        driver.findElement(By.id("password")).sendKeys(Password);

        driver.findElement(By.id("login")).click();
    }

    @Then("^Message displayed Login Successfully$")
    public void message_displayed_Login_Successfully() {
        driver.quit();
    }
}
```

Output:

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Displays the project structure for "CucumberSample1".
- Code Editor:** Shows the content of `TestRunner.java` with the following code:

```
1 package testrunner;
2
3* import org.junit.runner.RunWith;
4
5 @RunWith(Cucumber.class)
6 @CucumberOptions(features = "Feature", glue = "step",plugin={"pretty","html:Report/cucumber"})
7 public class TestRunner {
8
9 }
10
11 }
```

- Console Output:** Displays the execution results of the Cucumber test.

```
TestRunner (I) JUnit C:\Program Files\Java\jdk1.8.0_101\bin\javaw.exe (Aug 9, 2017, 3:29:06 PM)
Feature: check the login adactin functionality
  Scenario Outline: Login Adactin
    Given User is an adactin page
    When User enter "<UserName>" and "<Password>" and click login button
    Then Message displayed Login Successfully

Examples:
1502272750892  geckodriver      INFO    Listening on 127.0.0.1:6001
1502272751151  geckodriver::marionette INFO    Starting browser C:\Program Files\Mozilla Firefox\firefox.exe with ar
```

- Taskbar:** Shows various application icons including Windows, Internet Explorer, File Explorer, and several browser icons.

AdactIn.com - Hotel Reservation

www.adactin.com/HotelApp/

Welcome to AdactIn Group of Hotels

Existing User Login - Build 1

Username: vengat16
Password: *****

[Forgot Password?](#)

[New User Register Here](#)

Important Note:
Hotel Application has 2 builds:

- **Build 1** – Has been developed with known defects. Thus, functional test cases and automation scripts will fail on this build.
- **Build 2** – Known defects have been fixed. Thus, functional test cases and automation test scripts should pass when executed on this build.
[Go to Build 2](#)

AdactIn.com - Hotel Reservation

www.adactin.com/HotelApp/

Welcome to AdactIn Group of Hotels

Existing User Login - Build 1

Username: ezhilarasan
Password: *****

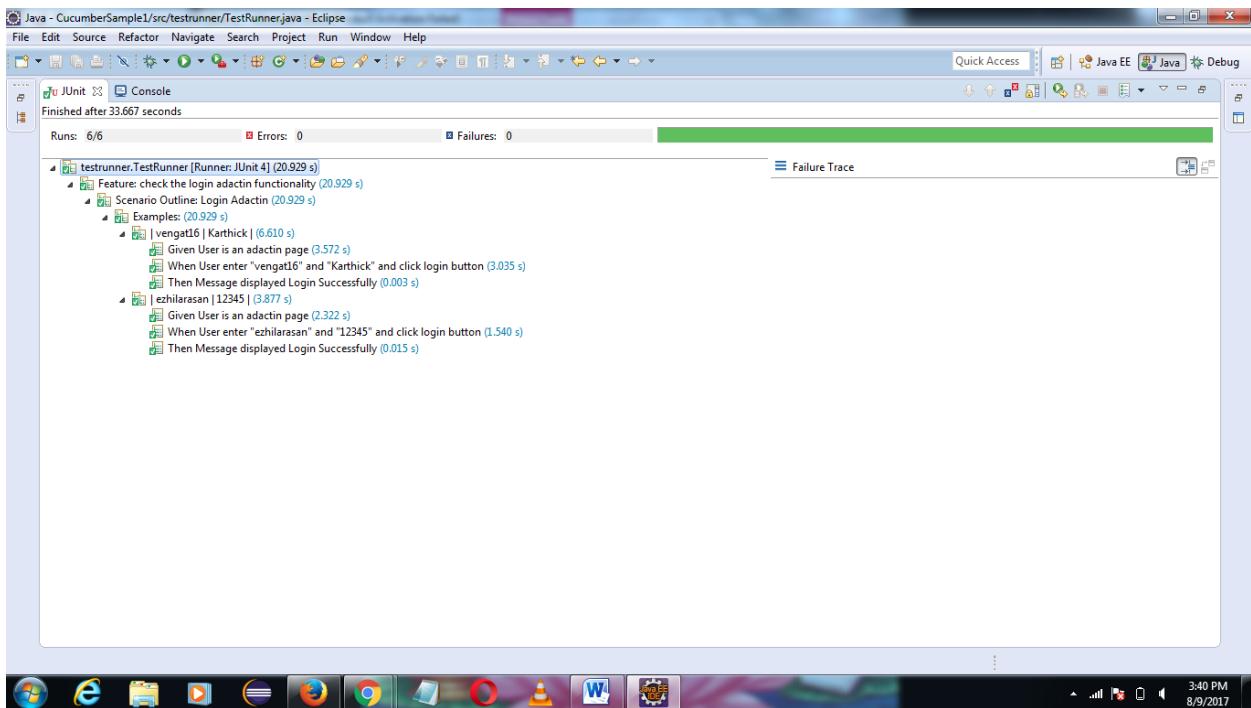
[Forgot Password?](#)

[New User Register Here](#)

Important Note:
Hotel Application has 2 builds:

- **Build 1** – Has been developed with known defects. Thus, functional test cases and automation scripts will fail on this build.
- **Build 2** – Known defects have been fixed. Thus, functional test cases and automation test scripts should pass when executed on this build.
[Go to Build 2](#)

Transferring data from www.adactin.com...



- If we enter more than one value it will automatically execute one by one.

Entering value in future file (another method):

By using List:

Feature file:

```

Feature: check the login adactin functionality
Scenario: Login Adactin
  Given User is an adactin page
  When User enters credentials and click login button
    |vengat16|Karthick|
    |ezhilarasan|12345|
  Then Message displayed Login Successfully

```

Step definition:

```

public class LoginTest {
  WebDriver driver;

  @Given("^User is an adactin page$")
  public void user_is_an_adactin_page() {
    System.setProperty("webdriver.gecko.driver",

```

```

        "H:\\java
software\\CucumberSample1\\driver\\geckodriver.exe");
        driver = new FirefoxDriver();
        driver.get("http://www.adactin.com/HotelApp/");
    }
@When("^User enters credentials and click login button$")
public void user_enters_credentials_and_click_login_button(DataTable
data) {
    List<List<String>> dataList = data.raw();
driver.findElement(By.id("username")).sendKeys(dataList.get(1).get(0));

    driver.findElement(By.id("password")).sendKeys(dataList.get(1).get(1));
        driver.findElement(By.id("login")).click();
}

@Then("^Message displayed Login Successfully$")
public void message_displayed_Login_Successfully() {
    driver.quit();
}
}

```

- This is the data table concept
- Here we input the data based on index

Test runner class:

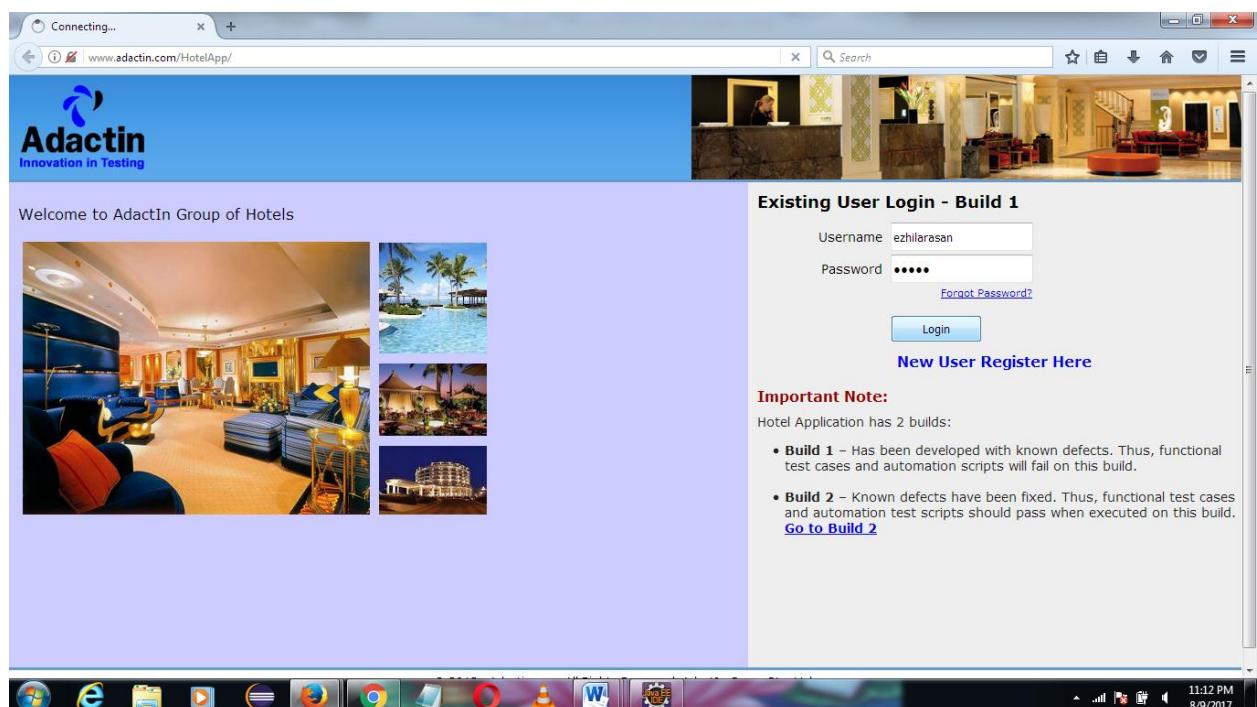
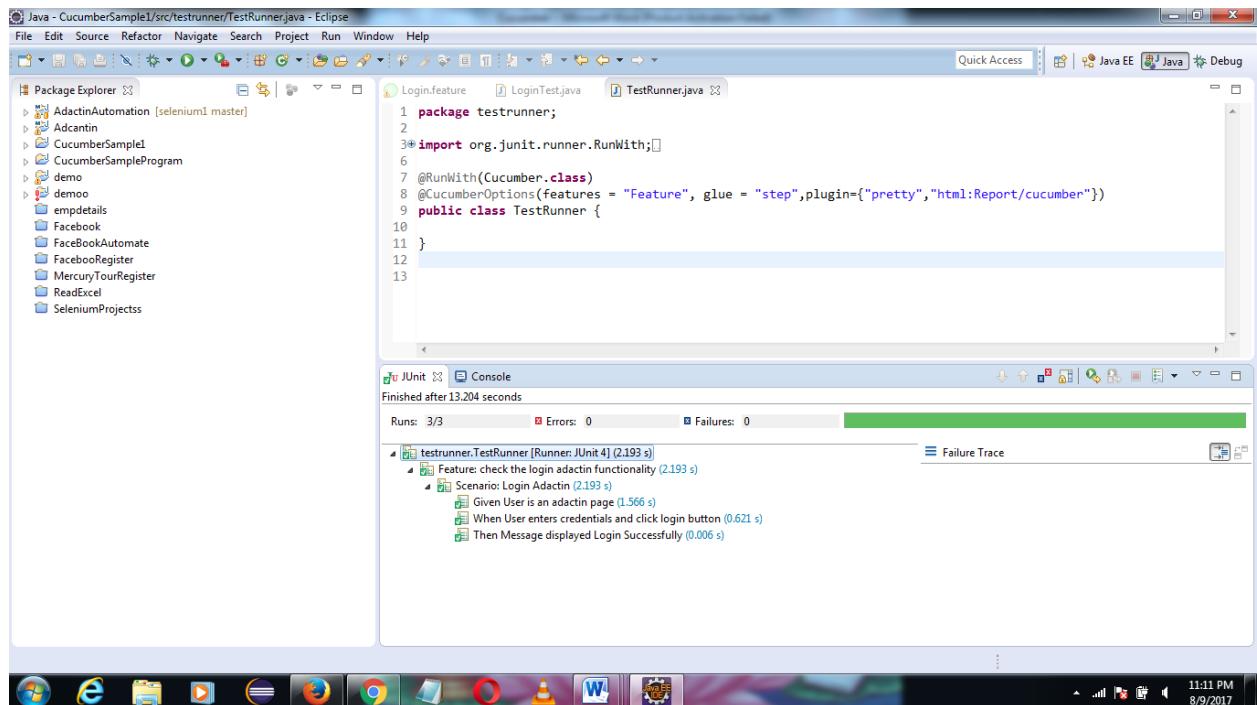
```

@RunWith(Cucumber.class)
@CucumberOptions(features = "Feature", glue =
"step",plugin={"pretty","html:Report/cucumber"})
public class TestRunner {

}

```

Output:



- Here we given first index, so it takes first index value.
- If we give 0th index, it will take 0th value

By using map:

Feature file:

Feature: check the login adactin functionality

Scenario: Login Adactin

```
Given User is an adactin page
When User enters credentials and click login button
|userName|Password|
|vengat16|Karthick|
|ezhilarasan|12345|
Then Message displayed Login Successfully
```

Step definition :

```
public class LoginTest {
    WebDriver driver;

    @Given("^User is an adactin page$")
    public void user_is_an_adactin_page() {
        System.setProperty("webdriver.gecko.driver",
                           "H:\\java
software\\CucumberSample1\\driver\\geckodriver.exe");
        driver = new FirefoxDriver();
        driver.get("http://www.adactin.com/HotelApp/");
    }
    @When("^User enters credentials and click login button$")
    public void user_enters_credentials_and_click_login_button(DataTable
data) {
        List<Map<String, String>> dataMap = data.asMaps(String.class,
String.class);
        driver.findElement(By.id("username")).sendKeys(dataMap.get(0).get("userName"));
    }

        driver.findElement(By.id("password")).sendKeys(dataMap.get(0).get("Pass
word"));
        driver.findElement(By.id("login")).click();
    }

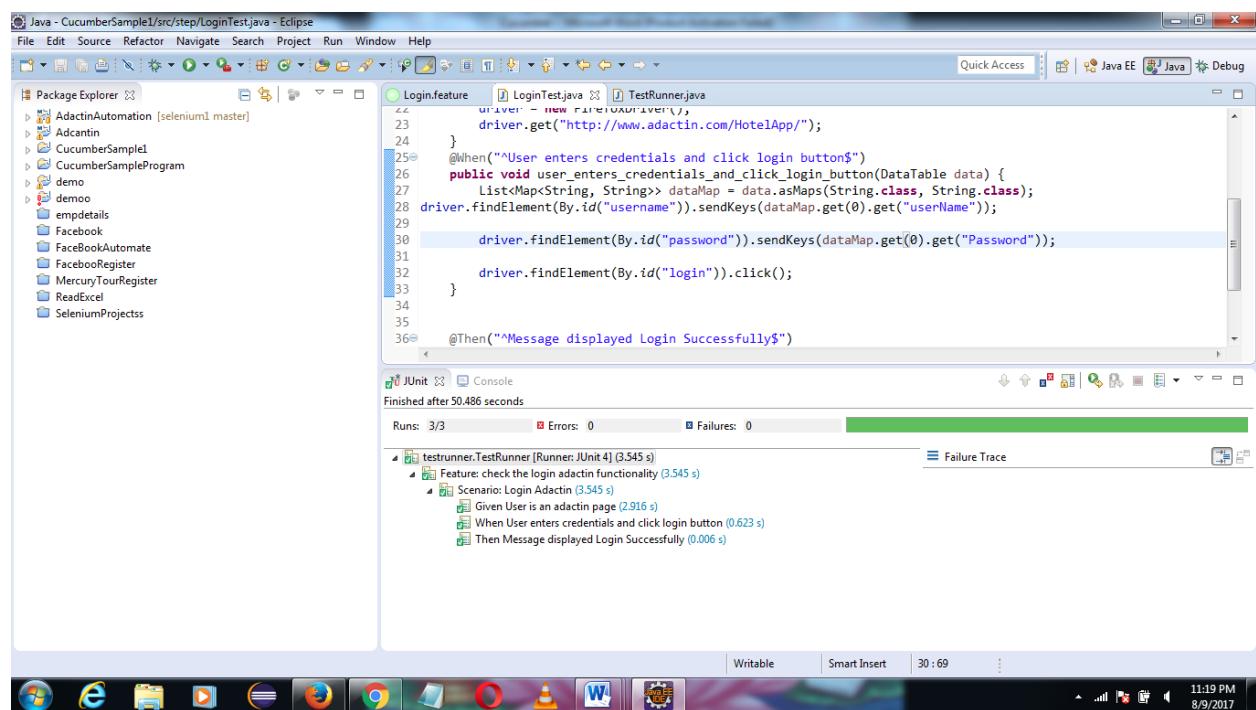
    @Then("^Message displayed Login Successfully$")
    public void message_displayed_Login_Successfully() {
        driver.quit();
    }
}
```

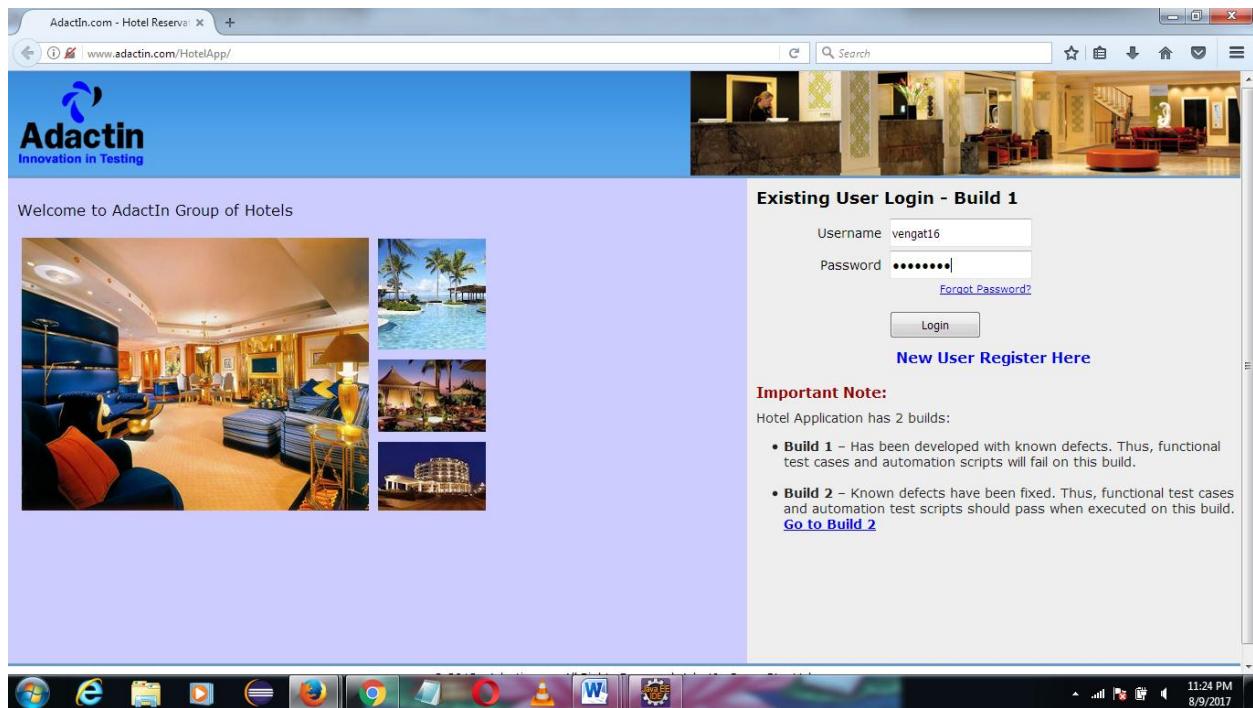
Test runner class:

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Feature", glue =
"step",plugin={"pretty","html:Report/cucumber"})
public class TestRunner {

}
```

Output :





- Here we using map, so we pass the key and it will enter relevant values of the particular index

To check random input values

- Here for example , one unique username is there, using this user name we need to check multiple times. That time we get error(i.e) “user name already registered”, these kind of error will show.
- To avoid these, we use **RandomStringUtils** class and append the value in the user name
- So we get the random input
- Syntax :
 - To append alphabet value
`String s = RandomStringUtils.randomAlphabetic(4);`
 Here, 4 means 4 alphabet letters will add randomly
 - To append alphabet value
`String s = RandomStringUtils.randomAlphanumeric (4);`
 Here, 4 means 4 alphanumeric letters will add randomly

- To append alphabet value

```
String s = RandomStringUtils.randomNumeric(4);
Here, 4 means 4 numeric values will add randomly
```

Feature file:

Feature: check the login adactin functionality

Scenario: Login Adactin

```
Given User is an adactin page
When User enters credentials and click login button
|userName|Password|
|vengat16|Karthick|
|ezhilarasan|12345|
Then Message displayed Login Successfully
```

Step definition :

```
public class LoginTest {
    WebDriver driver;

    @Given("^User is an adactin page$")
    public void user_is_an_adactin_page() {
        System.setProperty("webdriver.gecko.driver",
                           "H:\\\\java
software\\\\CucumberSample1\\\\driver\\\\geckodriver.exe");
        driver = new FirefoxDriver();
        driver.get("http://www.adactin.com/HotelApp/");
    }
    @When("^User enters credentials and click login button$")
    public void user_enters_credentials_and_click_login_button(DataTable
data) {
        String s = RandomStringUtils.randomAlphanumeric(4);
        List<Map<String, String>> dataMap = data.asMaps(String.class,
String.class);
        driver.findElement(By.id("username")).sendKeys(dataMap.get(0).get("userName")
+s);

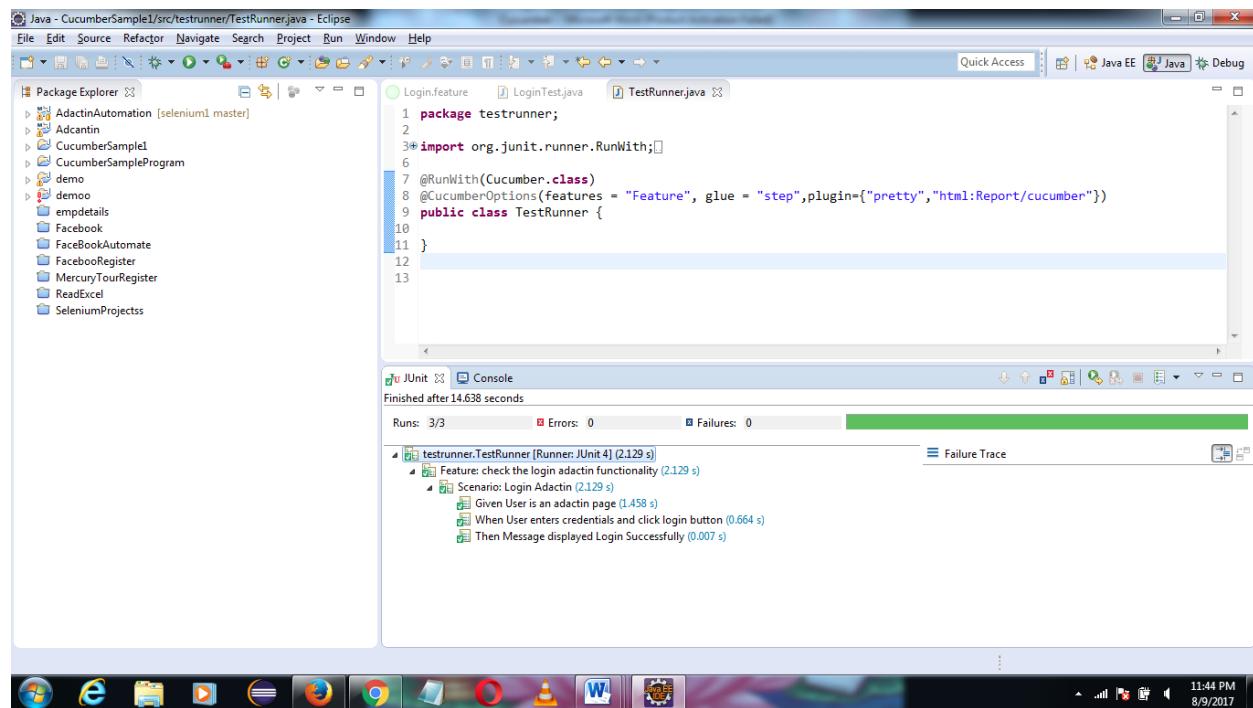
        driver.findElement(By.id("password")).sendKeys(dataMap.get(0).get("Pass
word"));
        driver.findElement(By.id("login")).click();
    }
    @Then("^Message displayed Login Successfully$")
    public void message_displayed_Login_Successfully() {
        driver.quit();
    }
}
```

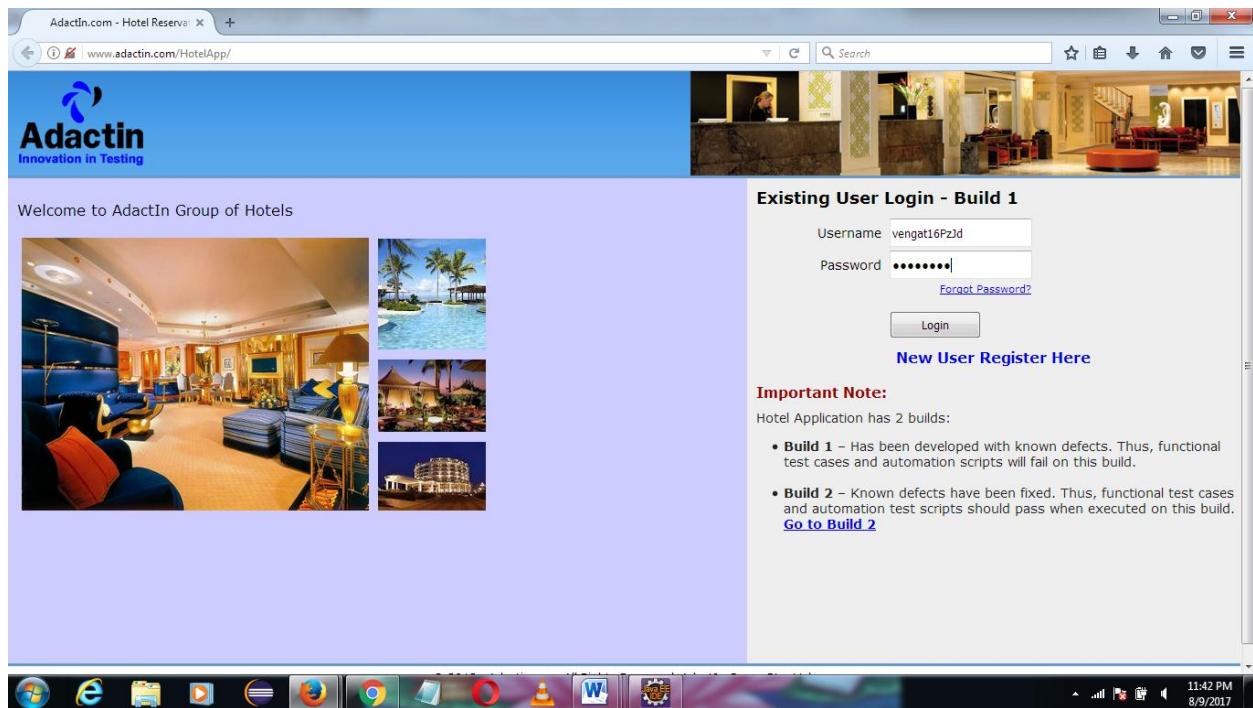
Test runner class:

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Feature", glue =
"step",plugin={"pretty","html:Report/cucumber"})
public class TestRunner {

}
```

Output :



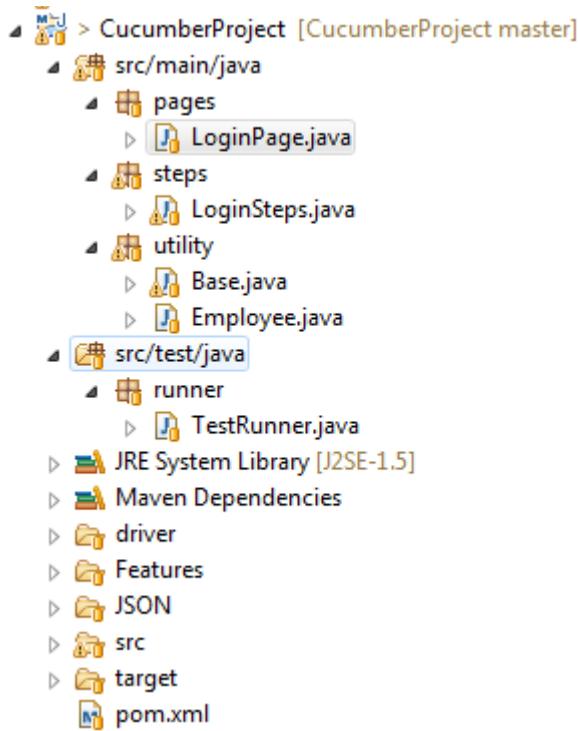


- Here 4 alphanumeric letters added in the user name
- It will randomly change for the next iteration

Cucumber with maven and POM:

- Here we use maven, POM and cucumber combination

Framework structure:



POM class:

```
public class LoginPage extends Base {

    @FindBy(id = "email")
    private WebElement txtUserName;

    @FindBy(id = "pass")
    private WebElement txtPassword;

    @FindBy(xpath = "//*[text()='Log In']")
    private WebElement btnLogin;

    @FindBy(xpath = "//a[@title='Go to Facebook home']")
    private WebElement imgFbLogo;

    public WebElement getImgFbLogo() {
        return imgFbLogo;
    }
}
```

```

public void setImgFbLogo(WebElement imgFbLogo) {
    this.imgFbLogo = imgFbLogo;
}

public void setTxtUserName(WebElement txtUserName) {
    this.txtUserName = txtUserName;
}

public void setTxtPassword(WebElement txtPassword) {
    this.txtPassword = txtPassword;
}

public LoginPage() {
    PageFactory.initElements(driver, this);
}

public WebElement getTxtUserName() {
    return txtUserName;
}

public WebElement getTxtPassword() {
    return txtPassword;
}

public void setTxtPassword(String txtPassword) {
    this.txtPassword.sendKeys(txtPassword);
}

public WebElement getBtnLogin() {
    return btnLogin;
}

public void setBtnLogin(WebElement btnLogin) {
    this.btnLogin = btnLogin;
}

}

```

Step definition:

```

public class LoginSteps extends Base {
    WebDriver driver;
    LoginPage loginPage;
}

```

```

@Given("^User is on facebook Page$")
public void user_is_on_Home_Page() {
    driver = getDriver();

}

@When("^User enters \"([^\"]*)\", \"([^\"]*)\" and click the login button$")
public void user_enters_and_click_the_login_button(String userName,
    String Password) {
    LoginPage = new LoginPage();

    setText(LoginPage.getTxtUserName(), userName);
    setText(LoginPage.getTxtPassword(), Password);
    clickBtn(LoginPage.getBtnLogin());

}

@Then("^Message displayed Login Successfully$")
public void message_displayed_Login_Successfully() {

    driver.quit();

}

```

Feature file:

Feature: check the login adactin functionality

Scenario: Login Adactin

Given User is an adactin page

When User enter "vengat16" and "Karthick" and click login button

Then Message displayed Login Successfully

Utility package:

Base:

```

public class Base {
    public static WebDriver driver;
    static WebDriverWait wait;
    static File f1 = new File("./JSON/Configuration.json");

    public static WebDriver getDriver() {
        JSONObject jsonObject = JSONReadFromFile();
        String browser = (String) jsonObject.get("browser");

```

```

        File f = new File("./driver");
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver",
f.getAbsolutePath()
                    + "/chromedriver.exe");
            driver = new ChromeDriver();
        } else if (browser.equals("firefox")) {
            System.setProperty("webdriver.gecko.driver",
f.getAbsolutePath()
                    + "/geckodriver.exe");
            driver = new FirefoxDriver();
        } else if (browser.equals("ie")) {
            System.setProperty("webdriver.ie.driver",
f.getAbsolutePath()
                    + "/IEDriverServer.exe");
            driver = new InternetExplorerDriver();
        }
        driver.manage().window().maximize();
        driver.get((String) jsonObject.get("url"));
        return driver;
    }

    public static boolean elementToBeVisible(WebDriver driver, int time,
                                              WebElement element) {
        boolean flag = false;
        try {
            wait = new WebDriverWait(driver, time);
            wait.until(ExpectedConditions.visibilityOf(element));
            flag = true;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return flag;
    }

    public static boolean alertIsPresent(WebDriver driver, int time) {
        boolean flag = false;
        try {
            wait = new WebDriverWait(driver, time);
            wait.until(ExpectedConditions.alertIsPresent());
            flag = true;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return flag;
    }
}

```

```
public static boolean elementToBeClickable(WebDriver driver, int time,
                                         WebElement element) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);

        wait.until(ExpectedConditions.elementToBeClickable(element));
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}

public static boolean elementFound(WebDriver driver, int time,
                                   WebElement element) {
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time,
TimeUnit.SECONDS);
    try {

        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();

    }
    return res;
}

public static boolean elementFound(WebElement element) {
    boolean res = false;
    try {
        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();

    }
    return res;
}

public static void setText(WebElement element, String name) {
    if (name != null && elementFound(element)) {
        element.clear();
        element.sendKeys(name);
    }
}
```

```

public static String getText(WebElement element) {
    String name = null;
    if (elementFound(element)) {
        name = element.getAttribute("value");
    }
    return name;
}

public static void clickBtn(WebElement element) {
if (elementFound(element)) {
    element.click();
}
}

public static JSONObject JSONReadFromFile() {
    JSONParser parser = new JSONParser();
    JSONObject jsonObject = null;
    try {

        Object obj = parser.parse(new
FileReader(f1.getAbsolutePath()));

        jsonObject = (JSONObject) obj;

    } catch (Exception e) {
        e.printStackTrace();
    }
    return jsonObject;
}

public static void getScreenShot(String screenShotFileName) {
    File screenShotLocation = new File("./screenshot/" +
screenShotFileName
            + ".png");
    TakesScreenshot screenshot = (TakesScreenshot) driver;
    File file = screenshot.getScreenshotAs(OutputType.FILE);
    try {
        FileUtils.copyFile(file, screenShotLocation);
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public static void uploadFiles(File path) {
try {
    Robot robot = new Robot();
    robot.setAutoDelay(3000);
    StringSelection selection = new StringSelection(
        path.getAbsolutePath());
}

```

```

        Toolkit.getDefaultToolkit().getSystemClipboard()
            .setContents(selection, null);
        // press control+v
        robot.keyPress(KeyEvent.VK_CONTROL);
        robot.keyPress(KeyEvent.VK_V);
        robot.setAutoDelay(3000);
        // release control+v
        robot.keyRelease(KeyEvent.VK_CONTROL);
        robot.keyRelease(KeyEvent.VK_V);
        // press enter
        robot.setAutoDelay(3000);
        robot.keyPress(KeyEvent.VK_ENTER);
        robot.keyRelease(KeyEvent.VK_ENTER);

    } catch (AWTException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public static List<HashMap<String, String>> readValueFromExcelSheet() {
    List<HashMap<String, String>> mapDatasList = new ArrayList();
    try {
        File excelLocaltion = new File("./Excel/Facebook.xlsx");

        String sheetName = "Datas";

        FileInputStream f = new FileInputStream(
            excelLocaltion.getAbsolutePath());
        Workbook w = new XSSFWorkbook(f);
        Sheet sheet = w.getSheet(sheetName);
        Row headerRow = sheet.getRow(0);
        for (int i = 0; i < sheet.getPhysicalNumberOfRows(); i++) {
            Row currentRow = sheet.getRow(i);
            HashMap<String, String> mapDatas = new
HashMap<String, String>();
            for (int j = 0; j <
headerRow.getPhysicalNumberOfCells(); j++) {
                Cell currentCell = currentRow.getCell(j);

                switch (currentCell.getCellType()) {
                    case Cell.CELL_TYPE_STRING:
mapDatas.put(headerRow.getCell(j).getStringCellValue(),
currentCell.getStringCellValue());
                    break;
                    case Cell.CELL_TYPE_NUMERIC:

```

```

mapDatas.put(headerRow.getCell(j).getStringCellValue(),
String.valueOf(currentCell
.getNumericCellValue())));
break;
}
}
mapDatasList.add(mapDatas);
}

} catch (Throwable e) {
e.printStackTrace();
}
return mapDatasList;
}

}

public static List<Employee> retrieveValueFromDataBase() {
ResultSet rs = null;
List<Employee> emp = new ArrayList<Employee>();
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection(
"jdbc:mysql://127.0.0.1:3306/selenium_schema",
"root", "");
PreparedStatement ps = con
.prepareStatement("SELECT * FROM
selenium_schema.emp_table where roll=3;");

rs = ps.executeQuery();

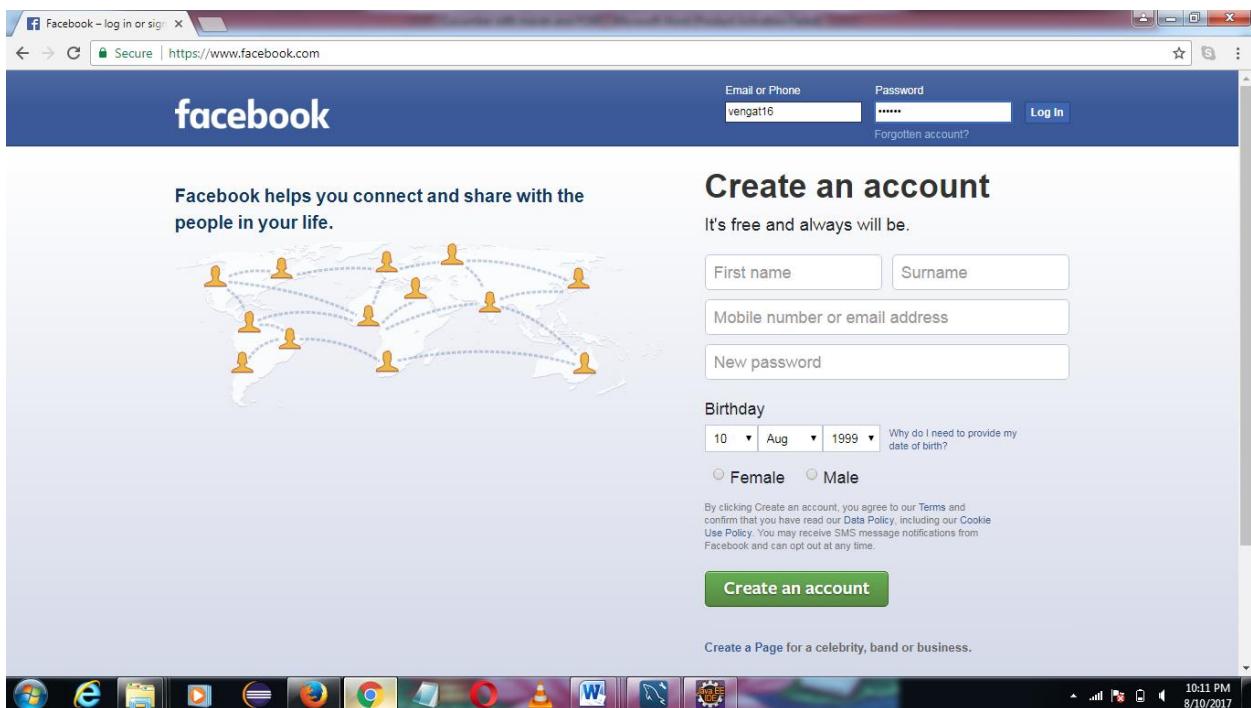
while (rs.next()) {
Employee e = new Employee();
e.setName(rs.getString("name"));
e.setPassword(rs.getString("password"));
emp.add(e);
}
} catch (ClassNotFoundException e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (SQLException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}
return emp;}}
```

Test runner class:

```
@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "steps")
public class TestRunner {

}
```

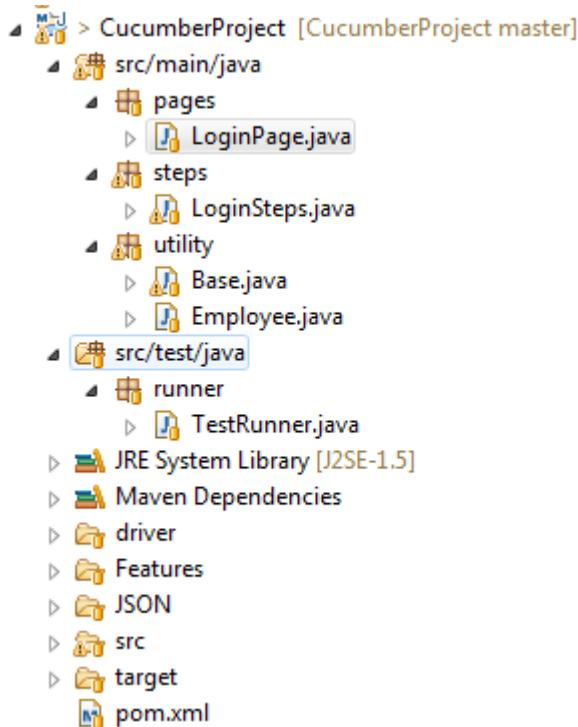
Output:



Cucumber with POM and data base:

- In this method, we taken all the input data's from data base

Framework structure:



Step definition:

```
public class LoginSteps extends Base {  
    WebDriver driver;  
    LoginPage loginPage;  
  
    @Given("^User is on facebook Page$")  
    public void user_is_on_Home_Page() {  
        driver = getDriver();  
    }  
  
    @When("^User enters \"([^\"]*)\", \"([^\"]*)\" and click the login  
button$")  
    public void user_enters_and_click_the_login_button(String userName,  
                                                    String Password) throws SQLException {  
        loginPage = new LoginPage();  
        List<Employee> emp = retrieveValueFromDataBase();  
  
        setText(loginPage.getTxtUserName(), emp.get(0).getName());  
        setText(loginPage.getTxtPassword(), emp.get(0).getPassword());  
        clickBtn(loginPage.getBtnLogin());
```

```

}

@Then("^Message displayed Login Successfully$")
public void message_displayed_Login_Successfully() {

    driver.quit();

}

}

```

POM class:

```

public class LoginPage extends Base {

    @FindBy(id = "email")
    private WebElement txtUserName;

    @FindBy(id = "pass")
    private WebElement txtPassword;

    @FindBy(xpath = "//*[text()='Log In']")
    private WebElement btnLogin;

    @FindBy(xpath = "//a[@title='Go to Facebook home']")
    private WebElement imgFbLogo;

    public WebElement getImgFbLogo() {
        return imgFbLogo;
    }

    public void setImgFbLogo(WebElement imgFbLogo) {
        this.imgFbLogo = imgFbLogo;
    }

    public void setTxtUserName(WebElement txtUserName) {
        this.txtUserName = txtUserName;
    }

    public void setTxtPassword(WebElement txtPassword) {
        this.txtPassword = txtPassword;
    }

    public LoginPage() {
        PageFactory.initElements(driver, this);
    }
}

```

```

public WebElement getTxtUserName() {
    return txtUserName;
}

public WebElement getTxtPassword() {
    return txtPassword;
}

public void setTxtPassword(String txtPassword) {
    this.txtPassword.sendKeys(txtPassword);
}

public WebElement getBtnLogin() {
    return btnLogin;
}

public void setBtnLogin(WebElement btnLogin) {
    this.btnLogin = btnLogin;
}
}

```

Utility package:

Base:

```

public class Base {
    public static WebDriver driver;
    static WebDriverWait wait;
    static File f1 = new File("./JSON/Configuration.json");

    public static WebDriver getDriver() {
        JSONObject jsonObject = JSONReadFromFile();
        String browser = (String) jsonObject.get("browser");

        File f = new File("./driver");
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver",
f.getAbsolutePath()
                + "/chromedriver.exe");
            driver = new ChromeDriver();

        } else if (browser.equals("firefox")) {
            System.setProperty("webdriver.gecko.driver",
f.getAbsolutePath()
                + "/geckodriver.exe");
            driver = new FirefoxDriver();

        } else if (browser.equals("ie")) {
    }
}

```

```

        System.setProperty("webdriver.ie.driver",
f.getAbsolutePath()
                    + "/IEDriverServer.exe");
driver = new InternetExplorerDriver();

}

driver.manage().window().maximize();
driver.get((String) jsonObject.get("url"));
return driver;
}

public static boolean elementToBeVisible(WebDriver driver, int time,
    WebElement element) {
boolean flag = false;
try {
    wait = new WebDriverWait(driver, time);
    wait.until(ExpectedConditions.visibilityOf(element));
    flag = true;
} catch (Exception e) {
    e.printStackTrace();
}
return flag;
}

public static boolean alertIsPresent(WebDriver driver, int time) {
boolean flag = false;
try {
    wait = new WebDriverWait(driver, time);
    wait.until(ExpectedConditions.alertIsPresent());
    flag = true;
} catch (Exception e) {
    e.printStackTrace();
}
return flag;
}

public static boolean elementToBeClickable(WebDriver driver, int time,
    WebElement element) {
boolean flag = false;
try {
    wait = new WebDriverWait(driver, time);

wait.until(ExpectedConditions.elementToBeClickable(element));
    flag = true;
} catch (Exception e) {
    e.printStackTrace();
}
return flag;
}

```

```
public static boolean elementFound(WebDriver driver, int time,
    WebElement element) {
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time,
TimeUnit.SECONDS);
    try {

        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();

    }
    return res;
}

public static boolean elementFound(WebElement element) {
    boolean res = false;
    try {
        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();

    }
    return res;
}

public static void setText(WebElement element, String name) {
    if (name != null && elementFound(element)) {
        element.clear();
        element.sendKeys(name);
    }
}

public static String getText(WebElement element) {
    String name = null;
    if (elementFound(element)) {
        name = element.getAttribute("value");
    }
    return name;
}

public static void clickBtn(WebElement element) {
if (elementFound(element)) {
        element.click();
    }
}
```

```
public static JSONObject JSONReadFromFile() {
    JSONParser parser = new JSONParser();
    JSONObject jsonObject = null;
    try {

        Object obj = parser.parse(new
FileReader(f1.getAbsolutePath())));
        jsonObject = (JSONObject) obj;

    } catch (Exception e) {
        e.printStackTrace();
    }
    return jsonObject;
}

public static void getScreenShot(String screenShotFileName) {
    File screenShotLocation = new File("./screenshot/" +
screenShotFileName
            + ".png");
    TakesScreenshot screenshot = (TakesScreenshot) driver;
    File file = screenshot.getScreenshotAs(OutputType.FILE);
    try {
        FileUtils.copyFile(file, screenShotLocation);
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public static void uploadFiles(File path) {
    try {
        Robot robot = new Robot();
        robot.setAutoDelay(3000);
        StringSelection selection = new StringSelection(
                path.getAbsolutePath());
        Toolkit.getDefaultToolkit().getSystemClipboard()
                .setContents(selection, null);
        // press control+v
        robot.keyPress(KeyEvent.VK_CONTROL);
        robot.keyPress(KeyEvent.VK_V);
        robot.setAutoDelay(3000);
        // release control+v
        robot.keyRelease(KeyEvent.VK_CONTROL);
        robot.keyRelease(KeyEvent.VK_V);
        // press enter
        robot.setAutoDelay(3000);
        robot.keyPress(KeyEvent.VK_ENTER);
        robot.keyRelease(KeyEvent.VK_ENTER);
    }
}
```

```

        } catch (AWTException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    public static List<HashMap<String, String>> readValueFromExcelSheet() {
        List<HashMap<String, String>> mapDatasList = new ArrayList();
        try {
            File excelLocaltion = new File("./Excel/Facebook.xlsx");

            String sheetName = "Datas";

            FileInputStream f = new FileInputStream(
                excelLocaltion.getAbsolutePath());
            Workbook w = new XSSFWorkbook(f);
            Sheet sheet = w.getSheet(sheetName);
            Row headerRow = sheet.getRow(0);
            for (int i = 0; i < sheet.getPhysicalNumberOfRows(); i++) {
                Row currentRow = sheet.getRow(i);
                HashMap<String, String> mapDatas = new
                    HashMap<String, String>();
                for (int j = 0; j <
                    headerRow.getPhysicalNumberOfCells(); j++) {
                    Cell currentCell = currentRow.getCell(j);

                    switch (currentCell.getCellType()) {
                        case Cell.CELL_TYPE_STRING:
                            mapDatas.put(headerRow.getCell(j).getStringCellValue(),
                                currentCell.getStringCellValue());
                            break;
                        case Cell.CELL_TYPE_NUMERIC:
                            mapDatas.put(headerRow.getCell(j).getStringCellValue(),
                                String.valueOf(currentCell
                                    .getNumericCellValue())));
                            break;
                    }
                }
                mapDatasList.add(mapDatas);
            }
        } catch (Throwable e) {
    }
}

```

```

        e.printStackTrace();
    }
    return mapDatasList;
}

public static List<Employee> retriveValueFromDataBase() {
    ResultSet rs = null;
    List<Employee> emp = new ArrayList<Employee>();
    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:3306/selenium_schema",
                "root", "");
        PreparedStatement ps = con
                .prepareStatement("SELECT * FROM
selenium_schema.emp_table where roll=3;");

        rs = ps.executeQuery();

        while (rs.next()) {
            Employee e = new Employee();
            e.setName(rs.getString("name"));
            e.setPassword(rs.getString("password"));
            emp.add(e);

        }
    } catch (ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return emp;
}
}

```

Employee:

```

public class Employee {
    private String name;
    private String password;

    public String getName() {
        return name;
    }
}

```

```

public void setName(String name) {
    this.name = name;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

}

```

Test runner class:

```

@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "steps")
public class TestRunner {

}

```

Feature file:

Scenario Outline: Successful Login with Valid Credentials

Given User is on facebook Page

When User enters "<userName>","<password>" and click the login button

Then Message displayed Login Successfully

Examples:

	userNmae	password	
	ganesh	Java65	

Input data:

➤ This is a input data from data base

The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays various database management sections like MANAGEMENT, INSTANCE, PERFORMANCE, SCHEMAS, and TABLES. The central area shows a query editor with the SQL command: `SELECT * FROM selenium_schema.employee_table;`. The Result Grid shows one row of data: id (1), name (vengat16), and password (Karthick). Below the result grid is an Action History window titled 'employee_table 1' which lists several database operations with their times, actions, messages, and durations.

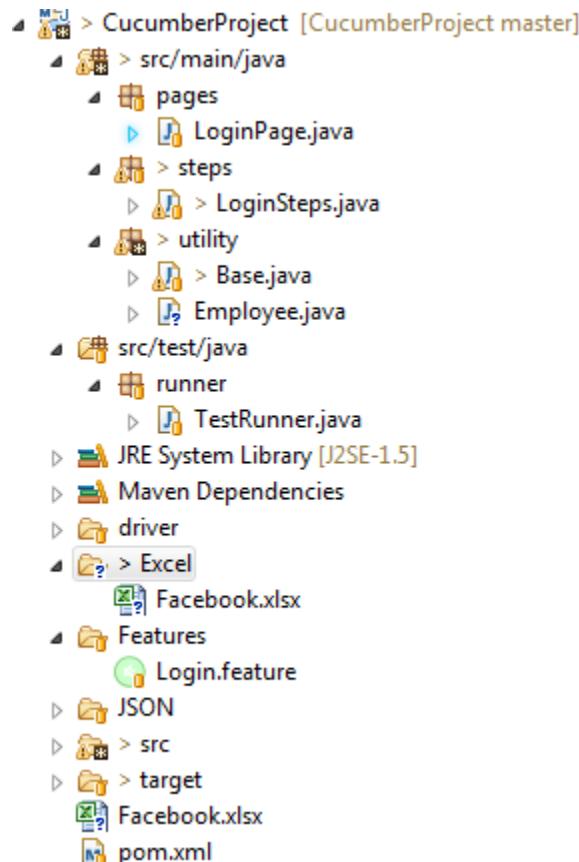
Output :

The screenshot shows a web browser displaying the Facebook login page at <https://www.facebook.com>. The page features the classic blue header with the word 'facebook'. It includes fields for 'Email or Phone' (containing 'vengat16') and 'Password'. Below these are links for 'Log In' and 'Forgotten account?'. To the right, there's a large 'Create an account' button. The main content area features a map of the world with orange icons representing users, and text encouraging users to connect and share. At the bottom, there's a note about account creation terms and conditions, and a 'Create an account' button.

Cucumber with POM and Data driven:

- Here we use data driven to read the input data's from excel sheet

Framework structure:



- Here one excel folder will added and in this folder have one excel sheet contains all the input datas

POM class:

```
public class LoginPage extends Base {

    @FindBy(id = "email")
    private WebElement txtUserName;

    @FindBy(id = "pass")
    private WebElement txtPassword;

    @FindBy(xpath = "//*[text()='Log In']")
    private WebElement btnLogin;
```

```
@FindBy(xpath = "//a[@title='Go to Facebook home']")
private WebElement imgFbLogo;

public WebElement getImgFbLogo() {
    return imgFbLogo;
}

public void setImgFbLogo(WebElement imgFbLogo) {
    this.imgFbLogo = imgFbLogo;
}

public void setTxtUserName(WebElement txtUserName) {
    this.txtUserName = txtUserName;
}

public void setTxtPassword(WebElement txtPassword) {
    this.txtPassword = txtPassword;
}

public LoginPage() {
    PageFactory.initElements(driver, this);
}

public WebElement getTxtUserName() {
    return txtUserName;
}

public WebElement getTxtPassword() {
    return txtPassword;
}

public void setTxtPassword(String txtPassword) {
    this.txtPassword.sendKeys(txtPassword);
}

public WebElement getBtnLogin() {
    return btnLogin;
}

public void setBtnLogin(WebElement btnLogin) {
    this.btnLogin = btnLogin;
}
}
```

Utility package:

Base class:

```
public class Base {
    public static WebDriver driver;
    static WebDriverWait wait;
    static File f1 = new File("./JSON/Configuration.json");

    public static WebDriver getDriver() {
        JSONObject jsonObject = JSONReadFromFile();
        String browser = (String) jsonObject.get("browser");

        File f = new File("./driver");
        if (browser.equals("chrome")) {
            System.setProperty("webdriver.chrome.driver",
f.getAbsolutePath()
                    + "/chromedriver.exe");
            driver = new ChromeDriver();
        } else if (browser.equals("firefox")) {
            System.setProperty("webdriver.gecko.driver",
f.getAbsolutePath()
                    + "/geckodriver.exe");
            driver = new FirefoxDriver();
        } else if (browser.equals("ie")) {
            System.setProperty("webdriver.ie.driver",
f.getAbsolutePath()
                    + "/IEDriverServer.exe");
            driver = new InternetExplorerDriver();
        }
        driver.manage().window().maximize();
        driver.get((String) jsonObject.get("url"));
        return driver;
    }

    public static boolean elementToBeVisible(WebDriver driver, int time,
        WebElement element) {
        boolean flag = false;
        try {
            wait = new WebDriverWait(driver, time);
            wait.until(ExpectedConditions.visibilityOf(element));
            flag = true;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return flag;
    }
}
```

```
}

public static boolean alertIsPresent(WebDriver driver, int time) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);
        wait.until(ExpectedConditions.alertIsPresent());
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}

public static boolean elementToBeClickable(WebDriver driver, int time,
    WebElement element) {
    boolean flag = false;
    try {
        wait = new WebDriverWait(driver, time);

        wait.until(ExpectedConditions.elementToBeClickable(element));
        flag = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return flag;
}

public static boolean elementFound(WebDriver driver, int time,
    WebElement element) {
    boolean res = false;
    driver.manage().timeouts().implicitlyWait(time,
    TimeUnit.SECONDS);
    try {

        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();

    }
    return res;
}

public static boolean elementFound(WebElement element) {
    boolean res = false;
    try {
        res = element.isDisplayed();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

```
        }
    return res;
}

public static void setText(WebElement element, String name) {
    if (name != null && elementFound(element)) {
        element.clear();
        element.sendKeys(name);
    }
}

public static String getText(WebElement element) {
    String name = null;
    if (elementFound(element)) {
        name = element.getAttribute("value");
    }
    return name;
}

public static void clickBtn(WebElement element) {
if (elementFound(element)) {
    element.click();
}
}

public static JSONObject JSONReadFromFile() {
    JSONParser parser = new JSONParser();
    JSONObject jsonObject = null;
    try {

        Object obj = parser.parse(new
FileReader(f1.getAbsoluteFile()));

        jsonObject = (JSONObject) obj;

    } catch (Exception e) {
        e.printStackTrace();
    }
    return jsonObject;
}

public static void getScreenShot(String screenShotFileName) {
    File screenShotLocation = new File("./screenshot/" +
screenShotFileName
            + ".png");
    TakesScreenshot screenshot = (TakesScreenshot) driver;
    File file = screenshot.getScreenshotAs(OutputType.FILE);
    try {
```

```

        FileUtils.copyFile(file, screenShotLocation);
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public static void uploadFiles(File path) {
    try {
        Robot robot = new Robot();
        robot.setAutoDelay(3000);
        StringSelection selection = new StringSelection(
            path.getAbsolutePath());
        Toolkit.getDefaultToolkit().getSystemClipboard()
            .setContents(selection, null);
        // press control+v
        robot.keyPress(KeyEvent.VK_CONTROL);
        robot.keyPress(KeyEvent.VK_V);
        robot.setAutoDelay(3000);
        // release control+v
        robot.keyRelease(KeyEvent.VK_CONTROL);
        robot.keyRelease(KeyEvent.VK_V);
        // press enter
        robot.setAutoDelay(3000);
        robot.keyPress(KeyEvent.VK_ENTER);
        robot.keyRelease(KeyEvent.VK_ENTER);
    } catch (AWTException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public static List<HashMap<String, String>> readValueFromExcelSheet() {
    List<HashMap<String, String>> mapDatasList = new ArrayList();
    try {
        File excelLocaltion = new File("./Excel/Facebook.xlsx");

        String sheetName = "Face";

        FileInputStream f = new FileInputStream(
            excelLocaltion.getAbsolutePath());
        Workbook w = new XSSFWorkbook(f);
        Sheet sheet = w.getSheet(sheetName);
        Row headerRow = sheet.getRow(0);
        for (int i = 0; i < sheet.getPhysicalNumberOfRows(); i++) {
            Row currentRow = sheet.getRow(i);
            HashMap<String, String> mapDatas = new
HashMap<String, String>();

```

```

                for (int j = 0; j <
headerRow.getPhysicalNumberOfCells(); j++) {
                    Cell currentCell = currentRow.getCell(j);

                    switch (currentCell.getCellType()) {
                        case Cell.CELL_TYPE_STRING:

mapDatas.put(headerRow.getCell(j).getStringCellValue(),
currentCell.getStringCellValue());
                                break;
                        case Cell.CELL_TYPE_NUMERIC:

mapDatas.put(headerRow.getCell(j).getStringCellValue(),
String.valueOf(currentCell
.getNumericCellValue()));

                                break;
                    }
                }

                mapDatasList.add(mapDatas);
            }

        } catch (Throwable e) {
            e.printStackTrace();
        }
        return mapDatasList;
    }

public static List<Employee> retrieveValueFromDataBase() {
    ResultSet rs = null;
    List<Employee> emp = new ArrayList<Employee>();
    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:3306/selenium_schema",
"root", "");
        PreparedStatement ps = con
                .prepareStatement("SELECT * FROM
selenium_schema.emp_table where roll=3;");

        rs = ps.executeQuery();

        while (rs.next()) {
            Employee e = new Employee();
            e.setName(rs.getString("name"));
    
```

```

        e.setPassword(rs.getString("password"));
        emp.add(e);

    }
} catch (ClassNotFoundException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
return emp;
}

}

```

Test runner class:

```

@RunWith(Cucumber.class)
@CucumberOptions(features = "Features", glue = "steps")
public class TestRunner {

}

```

Feature file:

Scenario Outline: Successful Login with Valid Credentials

Given User is on facebook Page

When User enters "<userName>","<password>" and click the login button

Then Message displayed Login Successfully

Examples:

	userNmae		password	
	ganesh		Java65	

Step definition file:

```

public class LoginSteps extends Base {
    WebDriver driver;
    LoginPage loginPage;

    @Given("^User is on facebook Page$")
    public void user_is_on_Home_Page() {
        driver = getDriver();

    }

```

```
@When("User enters \"([^\"]*)\", \"([^\"]*)\" and click the login button$")
public void user_enters_and_click_the_login_button(String userName,
                                                 String Password) {
    LoginPage = new LoginPage();

    setText(loginPage.getTxtUserName(),readValueFromExcelSheet().get(1)
        .get("Username"));

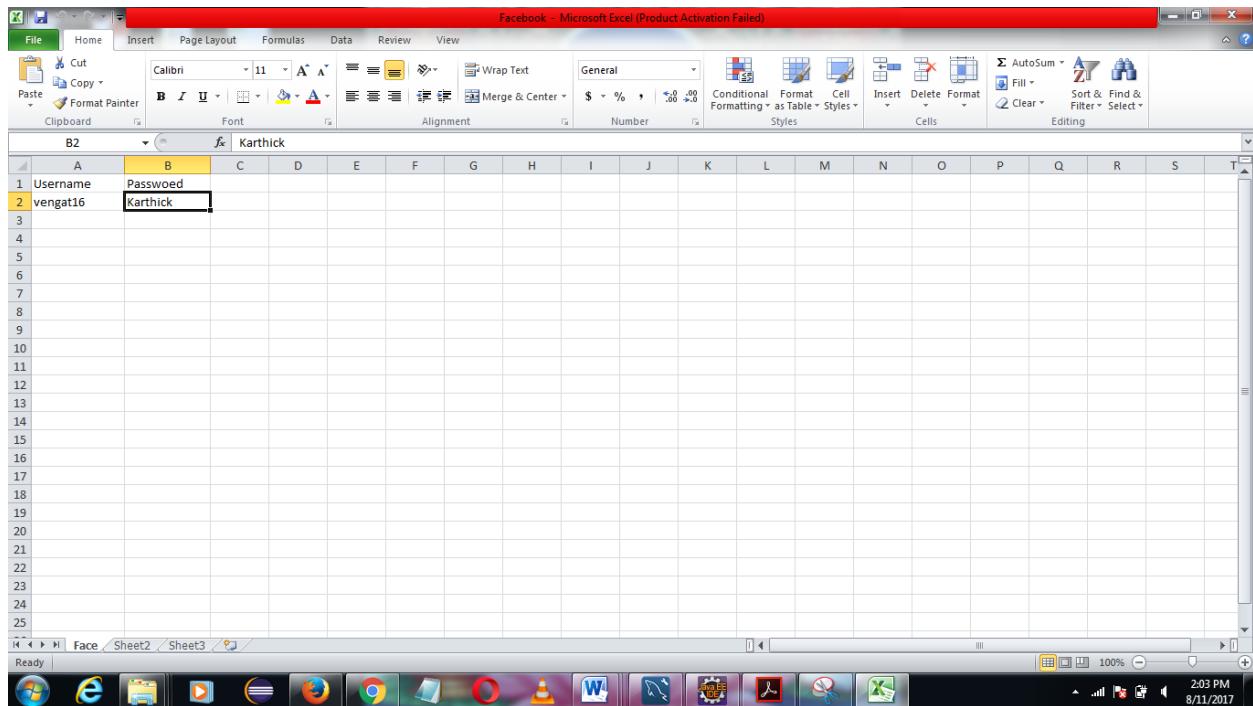
    setText(loginPage.getTxtPassword(),readValueFromExcelSheet().get(1)
        .get("Password"));

    clickBtn(loginPage.getBtnLogin());
}

@Then("Message displayed Login Successfully$")
public void message_displayed_Login_Successfully() {

    driver.quit();
}
```

Excel input:



Output :

