**Solution:**

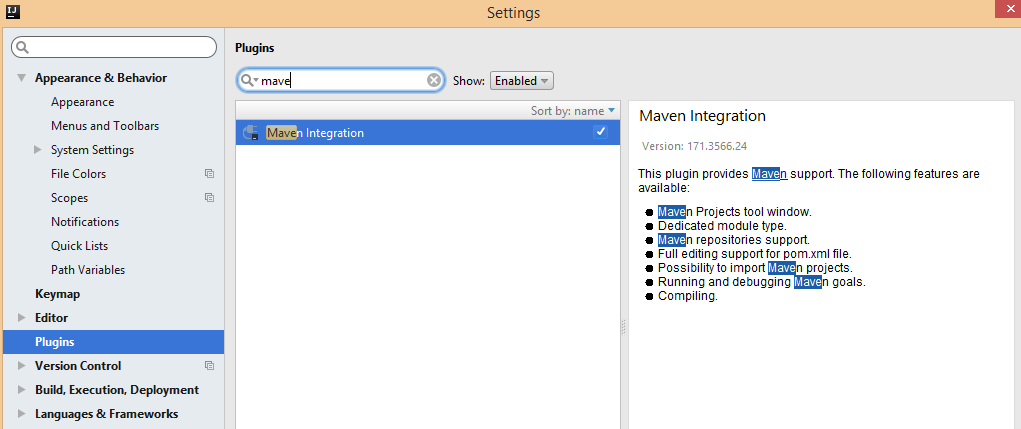
I have implemented the facebook page signup with random testdata solution, in testng- cucumber framework. The project is of type maven and the programming language used in java.

**Steps to install the Project:**

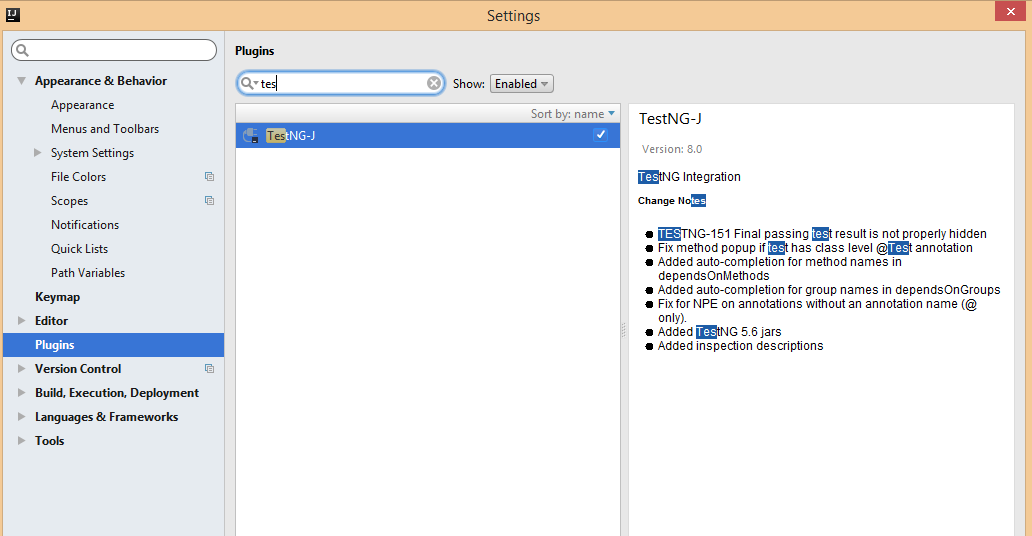
* Step-1: Copy the repository (https://github.com/chandru680/FacebookSignupScenarios.git)from GITHUB to your local machine
* Step-2: Browser & version required : Chrome - 73.0.3683(as this test will run in chrome browser)
* Step-3: Import the project to eclipse/Intellij. The plugins that are required are maven, testng and cucumber for java. The essential maven dependencies are mentioned in the pom.xml file.

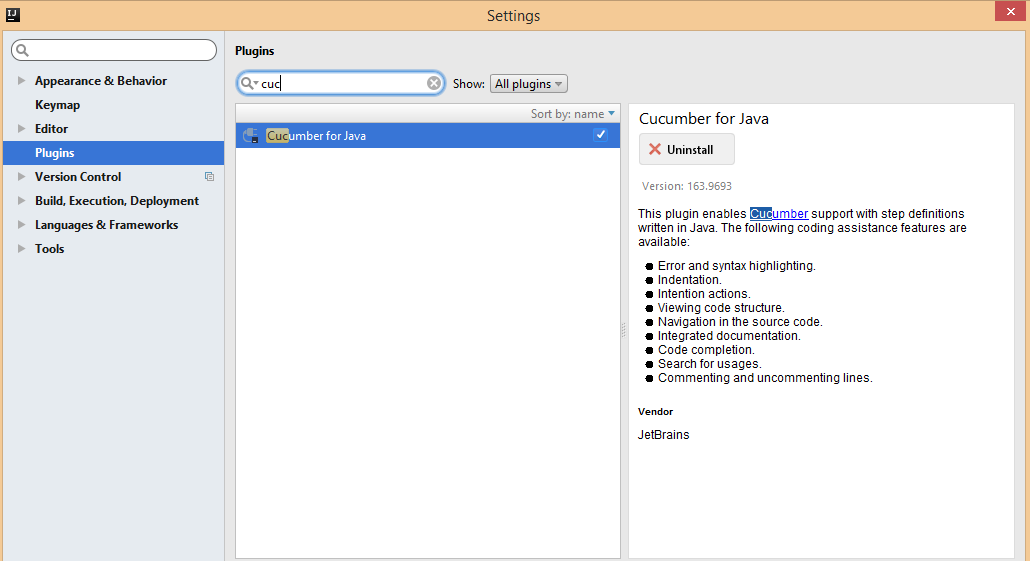
Screenshots of Plugins used:

1. Maven



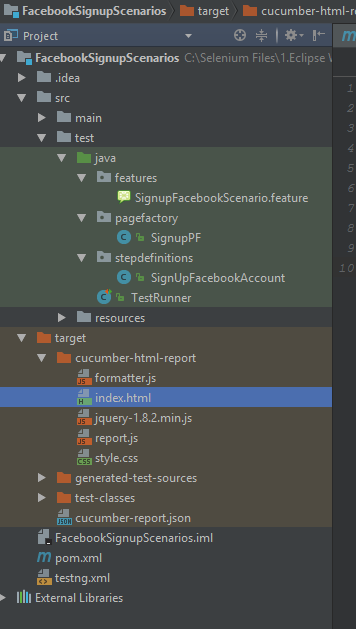
1. TestNG



1. Cucumber

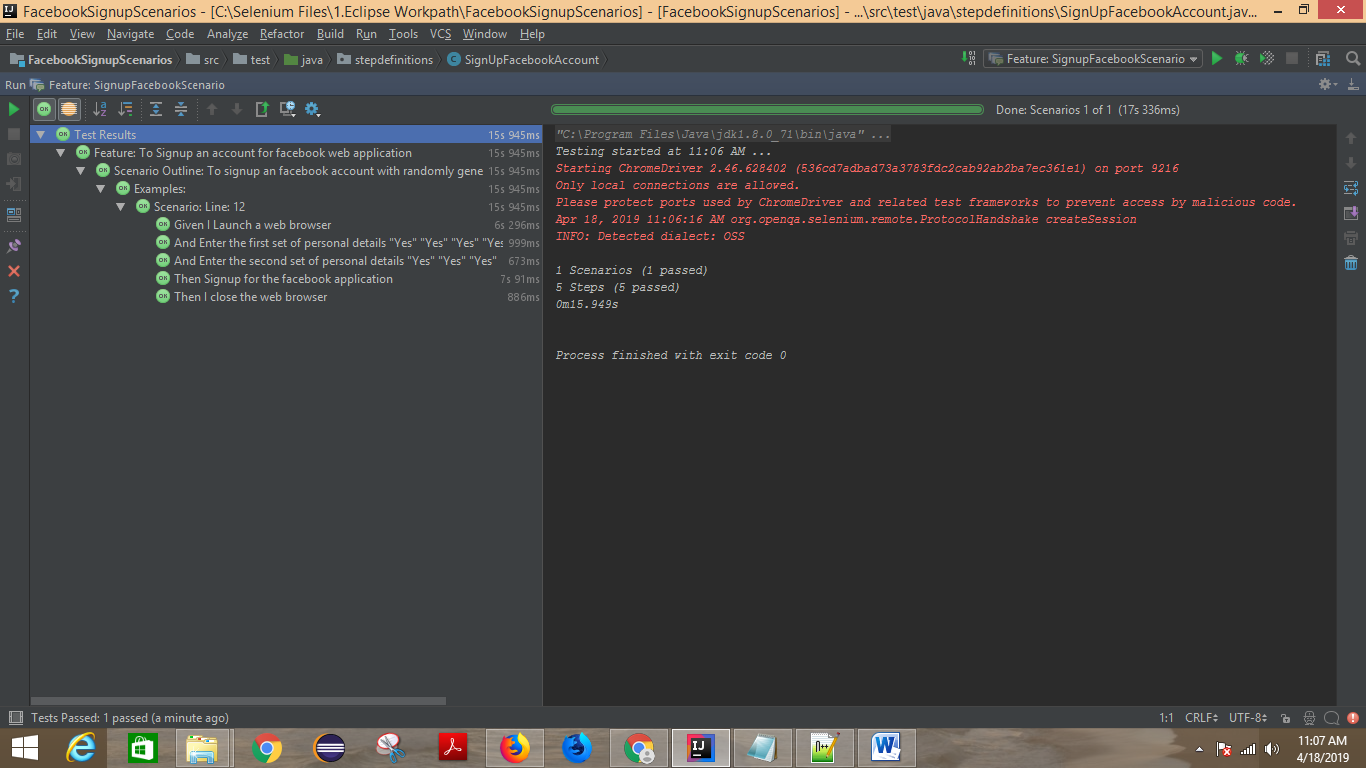
**Steps to run the Project:**

* Under the features folder we have the feature file stored which contains the test scenarios.
* The page factory folder contains all the Web elements in a structured format which we are using in our test.
* Step definitions containing the test steps matching to the scenarios from the feature file.
* The test runner is heart of the project. We can either run the test by running the TestRunner or running the testing.xml file



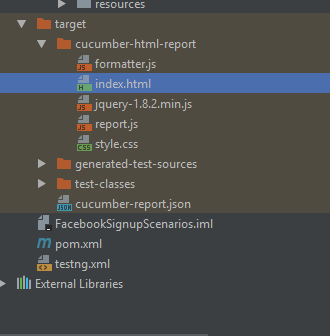
Once the test is run, the status of the run should be displayed like this below. For our test cases, all of the scenarios should pass.

Screenshot of the test run:



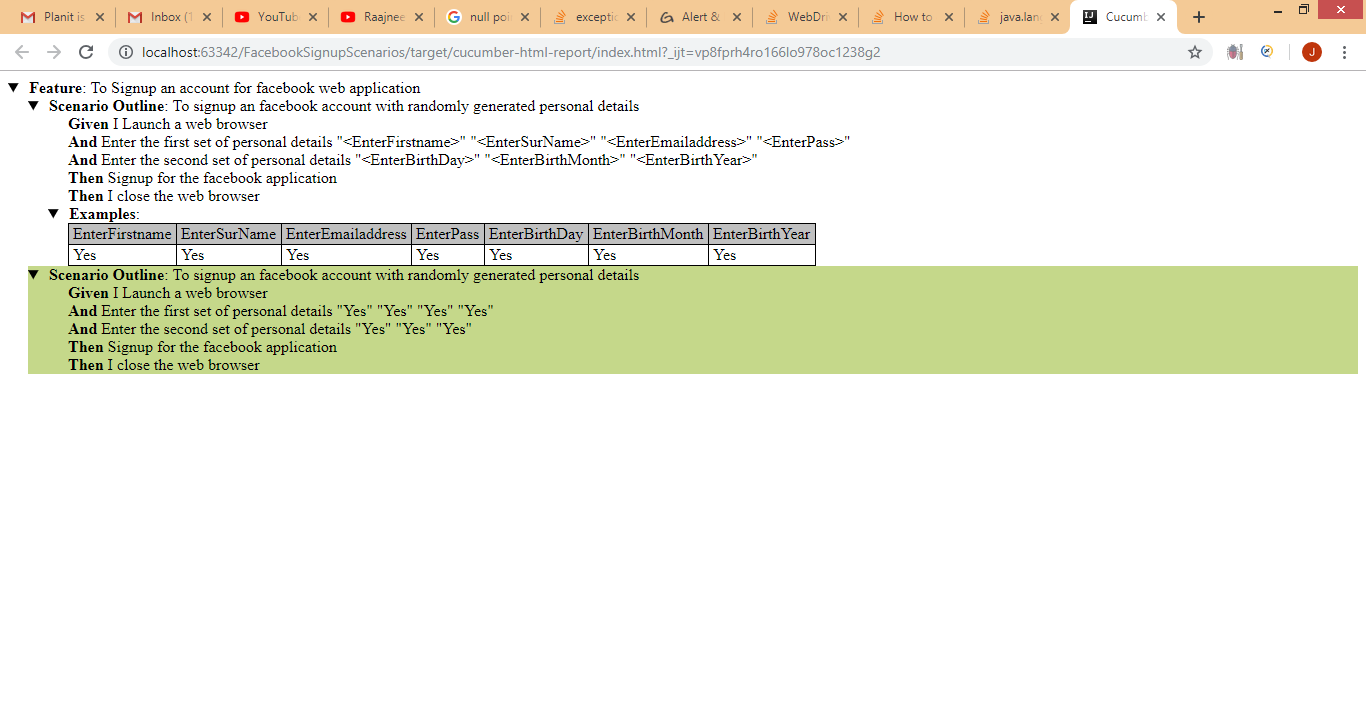
To view the reports, In your project structure, there is a folder called *target*. Inside *target* folderour cucumber html report will be generated.

Screenshot of the target folder structure:

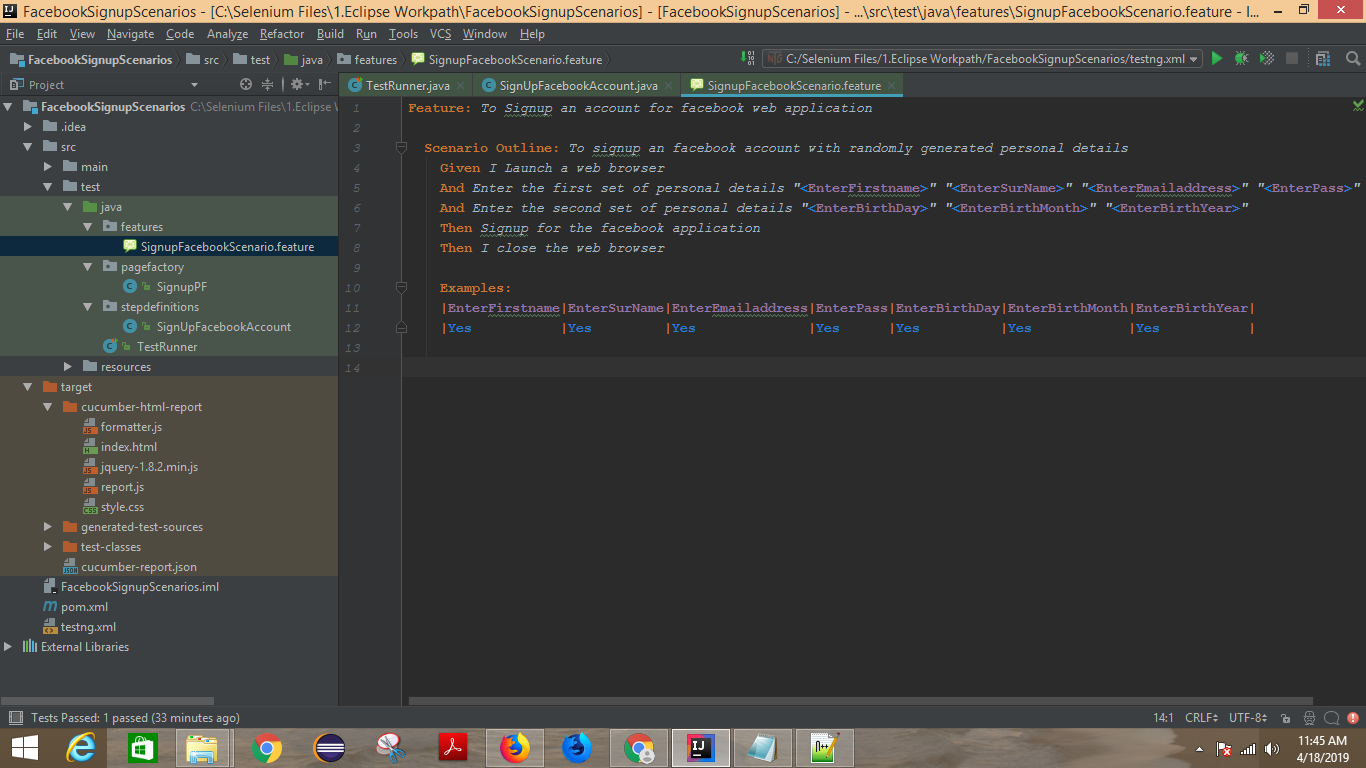


On opening the index.html in a browser, the test results will be shown by scenarios wise, how it has fared.

Screenshot of the test results:



I have put the validation in the scenario of whether we are entering the data to the respective field or not.So the code block will only execute, if the field has the value of “Yes” else it will not enter the data for that field in the signup page.In this way we test the negative test case also by capturing the alerts when we do not enter data and click signup.



Answers to the other questions which were like asked in the email:

1. What are some common causes of flakiness in automation tests and what can you do to help mitigate these causes?

Most of the cases when the automation fails is because

* The page has not loaded properly so we are not able perform action on the webelement
* It is taking too much time to load the webelements in the webpage.
* webPage has crashed.(reload the page and continue the test)
* when the locator values of the webelements have changed.(in this case update the locator values )
* We can solve this by using the explicit wait function which would wait for the webelement to load and do the activity.

1. Consider this scenario: You have estimated that a project needs 100 test cases automated, and it will take you 2 months to complete. However, you have now been told that you only have 1 month to complete the project. What options can you consider to solve this problem?

I will focus on the main business cases and smoke testing cases that can realistically be completed rather than handling all possible scenarios and test data’s.

1. Choose 3 software quality characteristics that relate to automation code and explain why you think they are important or not important to observe.

* Modular approach
* Bringing in a automation strategy which align with the continuous changes in the application
* Understanding of the application well.
* Responsiveness of the web page and actions.

1. (OPTIONAL) You are working at a bank, you are leading a team of testers, and you have been assigned to a project to introduce a new self-service tool where customers can create a new bank account by filling out an online form. Briefly describe how you would create a test strategy for this work, what factors would you need to consider, what extra information would you need to gather, what are the risks, etc.

On a high level , the thing that I would consider are

**Test Strategy**

* Resource planning
* Schedule for testing phase (includes automation and manual effort)
* Automation analysis and strategy,
* tools to be used for automation, defect management and project management

**Test development**

* Test plan
* Test scripts
* Types of test data to be used

**Test execution and defect management**

* Execute the test
* Report the defects
* Submit metrics
* Defect verification and retesting

**Delivery**

* Perform UAT and BVT
* Installation testing
* Deliver it to production after stakeholders walkthrough