Table of Contents

[Project Objective: 2](#_Toc92592606)

[Technologies Used: 2](#_Toc92592607)

[**Installing Cucumber Plugin from IntelliJ idea:**](#_Toc92592608)

[**IDE and Other Details:** IDE:](#_Toc92592609)

[Project Structure: 2](#_Toc92592610)

[Dependencies used in POM.xml file: 3](#_Toc92592612)

[TestNG.xml](#_Toc92592613)

[TestRunner.java 6](#_Toc92592615)

[ServiceHooks.java 7](#_Toc92592617)

[Feature File: 7](#_Toc92592619)

Steps to execute the project: 11

Summary 11

**Project Objective:** To Automate TodoMVC web-application to manage list of daily tasks

**Technologies Used:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Programming language | Testing tool | Build tool | Execution Tool | Reporting Tool | Framework (BDD) |
| Java | Selenium | Maven | TestNG | Extent Reports | Cucumber |
| Version:8 | Version : 3.141.59 | Version : 3.6.3 | Version: 6.9.8 | Version: 5.0.4 | Version:6.1.1 |

### 

### **Installing Cucumber Plugin from IntelliJ idea:** IntelliJ Idea (top left corner menu)🡪 Preferences 🡪Plugins(from left side menu) 🡪search with cucumber for java plugin in Market place tab 🡪 click on install 🡪 Once the plugin is installed, restart the IntelliJ IDE for the plugin to take effect

|  |
| --- |
|  |

### 

### **IDE and Other Details:** IDE: IntelliJ Idea Community Edition 2021.1 OS: MAC OS

**Project Structure:**

|  |
| --- |
|  |

**Dependencies used in POM.xml file:**  
  
1. Cucumber-Java:

This will indicate Maven, which java integrated Cucumber files are to be downloaded from the central repository to the local repository.

|  |
| --- |
| <dependency>  <groupId>io.cucumber</groupId>  <artifactId>cucumber-java</artifactId>  <version>6.1.1</version>  <scope>test</scope> </dependency> |

2. cucumber-testng:

This will indicate Maven, which Cucumber JUnit files are to be downloaded from the central repository to the local repository.

|  |
| --- |
| <dependency>  <groupId>io.cucumber</groupId>  <artifactId>cucumber-testng</artifactId>  <version>6.1.1</version>  <scope>test</scope> </dependency> |

3. Selenium-java:

This will indicate Maven, to download the Selenium java library into our local Maven repository.

|  |
| --- |
| <dependency>  <groupId>org.seleniumhq.selenium</groupId>  <artifactId>selenium-java</artifactId>  <version>3.141.59</version> </dependency> |

4. Testing:  
  
This will indicate Maven, to download the TestNG libraries into our local Maven repository.

|  |
| --- |
| <dependency>  <groupId>org.testng</groupId>  <artifactId>testng</artifactId>  <version>6.9.8</version>  <scope>test</scope> </dependency> |

5. extent reports:  
  
This will Indicate Maven, to download the extent reports related library to our local Maven Repository to create a pretty execution reports.

|  |
| --- |
| *<!-- https://mvnrepository.com/artifact/tech.grasshopper/extentreports-cucumber6-adapter -->* <dependency>  <groupId>tech.grasshopper</groupId>  <artifactId>extentreports-cucumber6-adapter</artifactId>  <version>2.1.0</version> </dependency> *<!-- https://mvnrepository.com/artifact/com.aventstack/extentreports -->* <dependency>  <groupId>com.aventstack</groupId>  <artifactId>extentreports</artifactId>  <version>5.0.4</version> </dependency> |

6. webdrivermanager:  
  
This will Indicate Maven, to download the webdrivermanager related library to our local Maven Repository to open custom chrome browser without the need to include driver in our project.

|  |
| --- |
| <dependency>  <groupId>io.github.bonigarcia</groupId>  <artifactId>webdrivermanager</artifactId>  <version>4.4.3</version> </dependency> |

7. maven-resources-plugin

The Resources Plugin handles the copying of project resources to the output directory. There are two different kinds of resources: main resources and test resources. The difference is that the main resources are the resources associated to the main source code while the test resources are associated to the test source code.

|  |
| --- |
| <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-resources-plugin</artifactId>  <version>2.4</version> </plugin> |

**TestNG.xml**This file is used in our project for executing the test cases. So we will run our test cases from testing.xml file.   
  
We can also use Junit instead of testing but I used testNG as it is easy for grouping of testcases and testing is large test suit friendly tool when compared to Junit

|  |
| --- |
|  |

**TestRunner.java**When testing.xml file is executed, the control first goes to TestRunner.java file as specified in Class name in testng.xml. Then it fetches below cucumber options   
1. features : Path which specifies feature file path to execute

2. glue : Path of step definitions  
3. Plugins : Path of reporting file plugin

4. Monochrome: Display the console output in readable format  
  
Other cucumber Options that can be included are:  
  
5. dryRun: If it is set as true, it means that Cucumber will only check that every Step mentioned in the Feature File has corresponding code written in Step Definition file or not. So in case any of the functions are missed in the Step Definition for any Step in Feature File. it will give us the message.

6. tags: To execute scenarios with specific tags

7. strict: If true, the execution will fail if there are unimplemented steps

|  |
| --- |
| **Graphical user interface, text, application  Description automatically generated** |

**ServiceHooks.java**This file is mainly used for providing hooks like @Before, @After from Cucumber framework. User will be allowed to actions such as Browser initialization and Closing the driver

|  |
| --- |
|  |

**Feature File:**  
  
A Feature File is an entry point to the Cucumber tests. This is a file where you will describe your tests in Descriptive language (Like English). It is an essential part of Cucumber, as it serves as an automation test script as well as live documents.

|  |
| --- |
|  |

## **Stepdefinition.java** This java file will have all the implemented steps of feature file, Where the steps in feature file will look for its matching regular expression in this file using Cucumber Gherkin language.

|  |
| --- |
|  |

## **TododList.java** This java file contains all the web elements defined in it ,along with few reusable methods

|  |
| --- |
|  |

## **Extent.properties:** This is mainly used for specifying the path for generating reports in the specified folder and also to enable the reporting. **Graphical user interface, text, application Description automatically generated**



## **Steps to execute the project: Process 1:** Right click on testing.xml file 🡪 Click on Run



|  |
| --- |
|  |

## **Process 2: Run from Cucumber Feature File** Go to feature file 🡪 Right click on the green arrow mark visible beside ” feature” to run all the scenarios at a time. 🡪 Select Run option Each scenario can also be run in the same way individually



|  |
| --- |
|  |

|  |
| --- |
|  |

## **How to see Generated Reports: test-output folder 🡪SparkReport 🡪Spark.html🡪 Right click 🡪Open in 🡪 Browser Graphical user interface, application, Teams Description automatically generated**

## **Graphical user interface, application Description automatically generated**

## **Summary:** The intention of this document is to provide a clear understanding of the approach followed for coding this web Application.