BDD Automation using Cucumber-JVM, Selenium and Kotlin

- Before creating the project, make sure you have Java 8 installed i.e. JDK 1.8.0 because, as of now, Cucumber Kotlin will not work with any Java version above 8
- Use IntelliJ Community Edition for this project
- Install Cucumber for Java IntelliJ plugin
- Install Gherkin IntelliJ plugin
- Install Kotlin IntelliJ plugin
- In IntelliJ, create a Maven project. Go to File --> New Project, select Maven and give it a filename and select location.
- If the default JDK that you are using is not 1.8.0, make the project use JDK 1.8.0. Go to File → Project Structure → Project SDK
- Add kotlin-maven-plugin to pom.xml
- Add Kotlin Stdlib, Cucumber, Cucumber-Junit, JUnit, Selenium, WebDriver Manager dependencies to the project by going to https://mvnrepository.com/, search for the dependency, select the recent version, copy the Maven dependency, and paste it in your pom.xml
- Reload Maven by clicking on the Load Maven Changes button that appears on the top right, this will download all the added dependencies to the project
- If the default JDK is not 1.8 and to run the tests from Maven, we must point the installed JDK to 1.8. This is temporary, meaning, when you close the IDE, the pointing will revert to the default. To point it, run the below 2 commands in the Terminal from the IDE

set JAVA_HOME=C:\Program Files (x86)\Java\jdk-1.8 [Whatever is your JDK 1.8 path]

set PATH=%JAVA_HOME%\bin;%PATH%

- Once dependencies are downloaded, run **mvn clean test** in the terminal, you will see a BUILD SUCCESS message
- Create a folder, **reports**, under **src/**, this will hold the generated reports
- Create a folder, **resources**, under **src/test**, this will hold the feature files. Right click this folder and select the option Mark directory as --> Test Resources Root.
- Create a package, **kotlincucumberselenium**, under **src/test/java**, this will hold the step definitions, class files
- Create a class file, **RunCucumberTest**, under **src/test/java**, this will be the Runner class
- Create a **cucumber.properties** file under **src/test/resources**
- Create feature files under **src/test/resources**
- Create step definition files as class files under **src/test/java/kotlincucumberselenium**

Running commands:

- As we have installed the 2 IntelliJ plugins at the start of creating the project, features/ scenarios can be run individually by clicking against the run button in the IDE
- As noted in the section above, we need to point the JDK temporarily to 1.8 if you are running another JDK by default.

- mvn test -Dcucumber.filter.tags="@ProductCheckout" -Dcucumber.plugin=html:./src/reports/Report.html – This will run the scenarios with the @ProductCheckout tag and store the report in the reports folder
- mvn test -Dcucumber.feature=./src/test/resources/ -Dcucumber.plugin=html:./src/reports/Report.html – This will run all the feature files and store the report in the reports folder

Folder structure: