# **Scenarios covered:**

Use BDD approach to test "petstore" [https://petstore.swagger.io/](http://s.bl-1.com/h/c26GhrWj?url=https://eur01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fpetstore.swagger.io%2F&data=02%7C01%7C%7C26f0e368f761450c190708d68abb65bd%7C84df9e7fe9f640afb435aaaaaaaaaaaa%7C1%7C0%7C636848934798639184&sdata=WTFz1HXi2g%2Fla%2B2fuxMeEnboEBvgd7T2HXOUv7IYhgA%3D&reserved=0) RESTful Web Service.

Your task is to test POST /pet service:

* Use any BDD framework of your choice, e.g. Cucumber, JBehave
* Identify and model Steps needed to test the feature. Please use some pseudo code to show your approach of implementing the steps.
* Implement the test scenarios using the Steps

# **Framework description:**

## **Overview**:

PetStore webservice automation is done on Cucumber and TestNg driven framework using maven for build, execution & dependency management. Framework is developed using Selenium Webdriver which supports cross browser testing (Chrome, Firefox, IE). Page Object with functional/structural implementation is done to maintain object repository. Page class for every page in our application and a page test class to maintain test for that pages. Also separate package for page and test classes. For ordering of tests and do parallel testng framework is used. Framework is flexible enough so we can do more customization in it as per requirement with breaking existing functionality.

## **Features**:

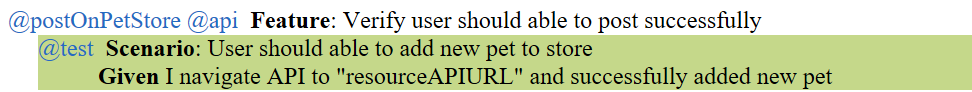
* Supports all cross browsers
* Supports parallel execution
* Takes Screenshot on failure
* Data are parametrized

## **Reporting**:

Cucumber report has been used for failure it also captures screenshots.

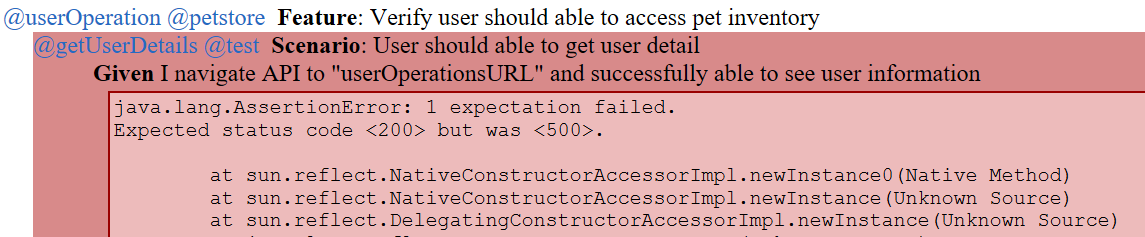
**Sample failure report structure:**

**Passed report: Can be customized more to get enhanced UI. In addition to this can be integrated with Jenkins to get different kind of reports and can be emailed automatically to stakeholders.**



**Failed reports:**

**Framework has capability to take screenshot on failure.**

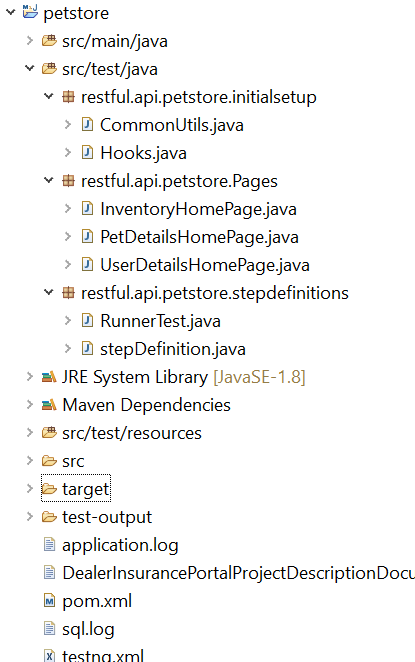


## **Tools Used:**

* Selenium Webdriver: 3.14.0
* Bdd framework Cucumber
* Chrome version:  73.0.3683.86
* Firefox: 66.0.1
* Internet explorer: 11.379.17763.0
* Testng version: 6.14.3
* Java version: Java 8
* Apache-Maven-3.3.9

# **Framework Structure:**

**Framework is following maven structure, so it has main and test folders:**



## **Src/test/java:**

**It contains three packages:**

* For initial set up, pages and stepdefinitions:
  + - Pages: Each functionality has one page
    - Initial setup: It contains setup classes like Hooks, Testbase and utilities

## **Configs:**

Kept test data/config properties files in this folder

## **Jars:**

Folder to place required ‘exe’ for Firefox and IE.

## **Target/Reporting:**

Folder for execution reports

## **Execution:**

It can be done in various ways.

### **RunnerTest.java:**

We can simply execute from RunnerTest.java file.

### **Testng.xml:**

To execute test cases and this also supports browser parameterization and parallel execution.