



QA Engineer assignment: API testing

Solution URL on GitHub

<https://github.com/Abokhalifa/com.payconiq>

Features of the solution of Gists

1. Written in Java using Rest-Assured framework.
2. Design pattern like Page Object Model
3. Data driven tests from external data source
4. TestNg is used as a testing framework.
5. Basic CRUD functions of are included.
6. Autonomous tests that do not depend on one another.
7. Modular solution with Maven dependencies

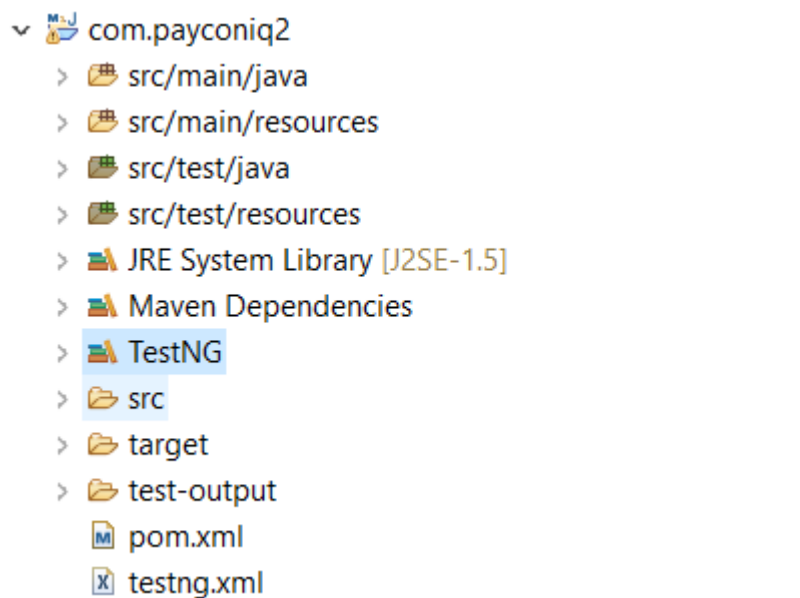
Future Plan

1. Cover more tests based on [POISED](#) test strategy.
 2. Handle exceptions.
 3. Cover Negative, Security and Performance.
 4. Improve the test report.
- Test strategy
 - [POISED](#) API test strategy is considered.
 - A mix of unit, integration, and system end-to-end tests are included.
 - Performance → To be covered in a later sprint.
 - Security → To be covered in a later sprint.
 - Negative tests → This is crucial in API testing and will be covered in a later sprint.
 - Programming skills
 - Inheritance and encapsulation are included among classes to improve reusability and extendibility.

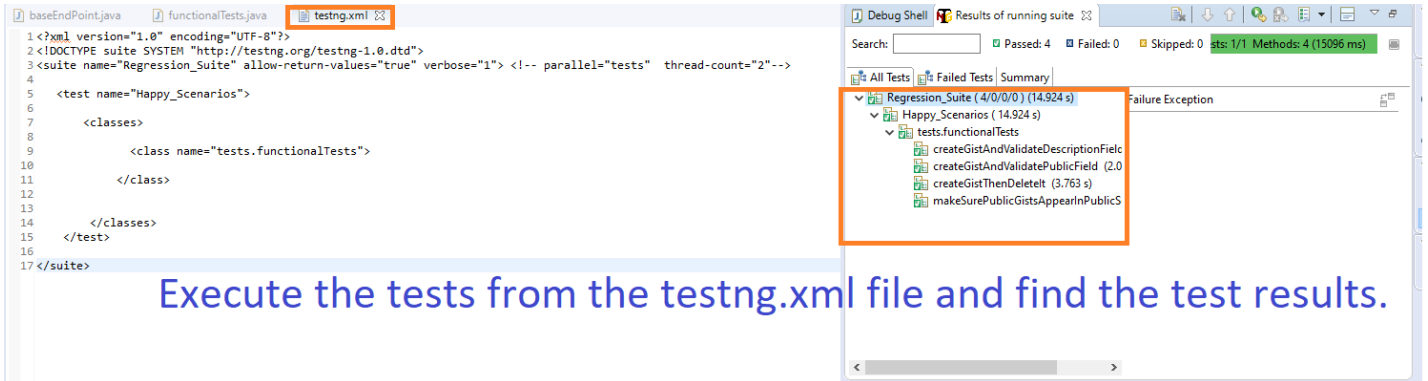
- How you test APIs
 - o Same as in Test Strategy section above
- The maintainability of the solution
 - o Separation of concerns is considered to isolate responsibilities of each class and hence more easier debugging is guaranteed.
- Show good structuring of tests for reusability /naming convention
 - o Clear and up to the point folder structures
 - o Consistent naming convention is used from the common practice of java naming convention
 - o A design pattern like the Pag Object Model is used

Instruction to run the solution

1. Make sure Maven installs the dependencies in the POM.xml file. To install the dependencies, right click the project directory → Maven → Update Project or simply **Alt + F5**.
2. Make sure to add TestNg framework.
3. Make sure the folder structure is something like the below image:



4. Set the “Token” parameter @ “baseEndPoint” class before using the app
5. Execute the tests from testing.xml and see the results as shown below.



The screenshot displays an IDE with two main panels. The left panel shows the `testng.xml` file with the following content:

```
1<?xml version="1.0" encoding="UTF-8"?>
2<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
3<suite name="Regression_Suite" allow-return-values="true" verbose="1"> <!-- parallel="tests" thread-count="2"-->
4
5  <test name="Happy_Scenarios">
6
7    <classes>
8
9      <class name="tests.functionalTests">
10
11      </class>
12
13    </classes>
14  </test>
15</suite>
```

The right panel shows the "Results of running suite" window. It indicates that 4 tests passed, 0 failed, and 0 were skipped, with a total of 15096 ms. The test results are as follows:

Test Name	Status	Time (s)
Regression_Suite	Passed	14.924
Happy_Scenarios	Passed	14.924
tests.functionalTests	Passed	
createGistAndValidateDescriptionField	Passed	2.0
createGistAndValidatePublicField	Passed	2.0
createGistThenDeleteIt	Passed	3.763
makeSurePublicGistsAppearInPublicS	Passed	

Execute the tests from the testng.xml file and find the test results.

Ahmed Abokhalifa

01-29-2020