

API Automation Framework

QA



June 5, 2020

Hissah

Table of Contents

[Project structure 2](#_Toc41925242)

[1.1 Project Structure 2](#_Toc41925243)

[1.2 Project Structure Description 2](#_Toc41925244)

[1.2.1 Common 2](#_Toc41925245)

[1.2.2 Main 3](#_Toc41925246)

[1.2.3 Services 3](#_Toc41925247)

# Project structure

## 1.1 Project Structure

## 1.2 Project Structure Description

As shown in the above figure we have 3 main folders (Common, main, Services) under main>java, below the detailed for each folder:

### 1.2.1 Common

This folder contains the main common methods that will be accessed across all the project.

**API Configuration**

At this folder a class that contains API request builder setting (Endpoints) like URI, ports, and we will extend the attributes from this class to use it to another class on service folder and will update the URIs and ports on Excel sheet using data provide for example:

APIConfigClass:

RequestSpecBuilder builder;

builder = new RequestSpecBuilder();

RequestSpecification requestSpec = builder.build();

builder.setBaseUri(http://000.000.00.000:8080); >>>>> the URI read from Excel sheet

builder.setContentType("application/json");

builder.addHeader("id", String.valueOf(121)); >>>>> this value read from Excel sheet

builder.addHeader("token-value","X-3scale-proxy-secret-token"); >>>>> this value read from Excel sheet

**Data Provider**

At this folder 2 classes that responsible to call test data and read data as the above figure.

**DB Connection**

At this folder a class that responsible to login and check data base connection. Note: we will use this class as object on (src < main < java < Services < Service name < Test cases < test case name)

**Logs Reports**

At this folder a class that responsible to generate a report using Extent report framework with the following features:

1. Test cases status (pass, fail, info)
2. Test case name
3. Test steps
4. Send report by email

Note: we will use this class on (src > main > java > Services > Service name > Test cases > test case name) to write Test case name, Test steps, etc.

### 1.2.2 Main

This is the main folder for running all the project. It has a class (running) which include the testing that run all the test cases.

### 1.2.3 Services

From the attached .rar file, by following this path:

src < main < java < Services < Service name < Test cases < test case name

We need the following points in the (test case name.class):

1. **Merge one more than data providers into one in TestNG (one method) for example:**

* We have single test method to send API request to get car information, so we need data from more than one sheet from one single workbook excel file called (DataConfigration.xlsx):
* API End point information sheet.
* DB Connection sheet
* Test Cases Date sheet

**Example:**

@TestNg(DataProveders = "dbconnection\_sheet1" , "APIEndpoint\_sheet2" , "testdata\_sheet3")

public void test (declare variables) {

statements …

}

1. **Read data row-by-row from (Excel sheet) for example:**

* I have a data file excel sheet having 100 entries for PersonID. I want to read a random value from the rows using random function.

1. **Send extent report file by email:**

* After Execution testing the result should be send by email (Each time the test suite is run it will send the result by email with suit information (DateTime , suit reference number, test final result).

1. **Stored the final test result in Database:**

* Create new database called (QA\_FRAMEWORK)
* Create table (TBL\_QA\_Test\_Cases\_Details) to store all cases details with below fields:

1. TestReference[PK] (Each time the test suite is run it will have a unique reference number (.
2. TestSuitName (this name will take from the testing.xml)
3. TestMethodName (this name will take from the method inside the class (under @TestNG)).
4. TestClassName (this will take from testing.xml).
5. TestStatus (this will take from Extent report logger (under @TestNG))).
6. ExecutionDateTime.

* Create table to store the final test suit result:
* (The final result of the test suit will be stored according to the results of the test status recorded in the previous table (TBL\_QA\_Test\_Cases\_Details) for example:

If all test cases on the previous table (TBL\_QA\_Test\_Cases\_Details ) with status (pass) so the final result of the test suit will be stored (pass) . But if there is any failure in one of the cases previous table (TBL\_QA\_Test\_Cases\_Details), the final result of the test suit will be stored (fail).

* Table (TBL\_QA\_FINAL\_TEST\_RESULT) fields:

1. TestRefernce[FK]
2. TestSuitName
3. FianlRsult
4. ExecutionDateTime.

**Example:**

* Stord the test cases details for single test suit but multiple execution with different time and reference.

TBL\_QA\_Test\_Cases\_Details:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TestRefernce | TestSuitName | TestMethodName | TestClassName | TestStatus | ExecutionDateTime |
| 1 | suit1 | ValidRequest | Class1 | Fail | 02/05/2020 14:00 PM |
| 1 | suit1 | InValidRequest | Class1 | Pass | 02/05/2020 14:00 PM |
| 1 | suit1 | IDvalid | Class2 | Pass | 02/05/2020 14:00 PM |
| 2 | suit1 | ValidRequest | Class1 | Pass | 02/05/2020 18:00 PM |
| 2 | suit1 | InValidRequest | Class1 | Pass | 02/05/2020 18:00 PM |
| 2 | suit1 | IDvalid | Class2 | Pass | 02/05/2020 18:00 PM |

* Stord test final result for single test suit but multiple execution with different time and reference

TBL\_QA\_FINAL\_TEST\_RESULT:

|  |  |  |  |
| --- | --- | --- | --- |
| **TestReference** | TestSuitName | **FinalRsult** | **ExecutionDateTime** |
| 1 | suit1 | Fail | 02/05/2020 14:00 PM |
| 2 | suit1 | Pass | 02/05/2020 18:00 PM |

**5- Additional service (API to call the database to read final test resualt):**

* Create new table called (TBL\_QA\_API\_TRANSECTIONS\_LOG) under QA\_FRAMEWORK database with Following fields :
* LOG\_ID
* CALL\_TIME
* TestRefrence
* TestSuitName
* ExecutionDateTime
* FinalRsult
* STATUS\_CODE (Http status code)
* URI
* Rest service to read result from table TBL\_QA\_FINAL\_TEST\_RESULT
* Return response JSON file with latest TestReference & TestSuiName & ExecutionDateTime and loged in table (TBL\_QA\_API\_TRANSECTIONS\_LOG) for example:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **LOG\_ID** | **CALL\_TIME** | **TestReference** | **TestSuitName** | **ExecutionDateTime** | **FinalResult** | **STATUS\_CODE** | **URI** |
| 1 | 2020-03-03 16:09:46 | 2 | suit1 | 02/05/2020 18:00 PM | Pass | 200 | /api/v1/qa-framework/getFinalResult |

**API Testing Example**

**Rest API details:**

**General note:**

Method type, endpoint, params, header, body > should be read from excel sheet.

All test case details should be export to the same excel that code read from it the test data and DB and extent report.

**Module name: Employee data**

**Test cases:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **TC#** | **Test case Title** | **Method** | **Endpoint** | **Params** | **Status Code** | **Response body** | **Test case result** |
| **1** | **Get all employee data** | **GET** | **http://dummy.restapiexample.com/api/v1/employees** |  |  |  |  |
| **2** | **Get a single employee data** | **GET** | **http://dummy.restapiexample.com/api/v1/employee/2** |  |  |  |  |
| **3** | **Save transaction on DB** |  | **http://dummy.restapiexample.com/api/v1/employee/2** |  |  |  |  |

**Notes:**

1. **All test case details above should be exported on extent report**
2. **In Status Code + Response body, it should be created and fill by the code**
3. **Test data should be read from excel sheet (End point, params, body in case of post request).**
4. **Test case result, it should be created and fill by the code if the status code 200 OK (Pass) otherwise (Fail)**
5. **All the details of test cases of response and request should be saved on DB.**
6. **All the cells above in orange color it will be filled by code.**

**API request types that should be automated:**

1. **Get request**
2. **Get request with params**
3. **Get request with authorization header**
4. **Get request with header**
5. **Post request with body (raw) json**
6. **Post request with body (form-data) text**
7. **Post request with body (form-data) file**
8. **Validate all above requests**

**SOAP API details:**

**General note:**

Test data > should be read from excel sheet.

All test case details should be export to the same excel that code read from it the test data and DB and extent report.

**WSDL:** <http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL>

**Service name:**

CapitalCity

**Test case:**

**(Excel sheet format)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **TC#** | **Test case Title** | **Scenario** | **Test data** | **XML** | **Expected result** | **Actual result** | **Test case result** |
| **1** | **Request the capital city** | **Send request to get the capital city name** | **SA** |  | **<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://www.oorsprong.org/websamples.countryinfo">**  **<soapenv:Header/>**  **<soapenv:Body>**  **<web:CapitalCity>**  **<web:sCountryISOCode>SA</web:sCountryISOCode>**  **</web:CapitalCity>**  **</soapenv:Body>**  **</soapenv:Envelope>** |  |  |
| **2** | **Receive capital city name** | **Receive response of capital city name** | **SA** |  | **<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">**  **<soap:Body>**  **<m:CapitalCityResponse xmlns:m="http://www.oorsprong.org/websamples.countryinfo">**  **<m:CapitalCityResult>Riyadh</m:CapitalCityResult>**  **</m:CapitalCityResponse>**  **</soap:Body>**  **</soap:Envelope>** |  |  |
| **3** | **Save transaction on DB** | **Request and response saved on DB** | **NA** | **NA** | **saved** |  |  |

**Notes:**

1. **All test case details above should be exported on extent report**
2. **In XML, it should be created and fill by the code**
3. **Test data should be read from excel sheet.**
4. **Actual result, it should be created and fill by the code**
5. **Test case result, it should be created and fill by the code if the expected match the actual result (Pass) otherwise fail.**
6. **All the details of test cases of response and request should be saved on DB.**
7. **All the cells above in orange color it will be filled by code.**

**API request types that should be automated:**

1. **XML request**
2. **Validate XML response**