**Test Automation Framework - UI & API**

Sample end 2 end test automation framework for UI and API testing.

This sample Software Test Automation Framework is built as a Maven project using Behavior Driven Development principles with Cucumber and Junit in Java. And also utilizing Page Object Module structure to maximize framework re-usability and maintainability. The scenario is written in Gherkin language, which is a plain English, that could benefit connecting the team.

And the framework is also structured to support API testing using RestAssured with TestNG, and Database integration capabilities with JDBC.

**Project structure**

The project structure is mainly divided into 2 sections, src/test/java and src/test/resource.

**src/test/java**

* Api tests - (API tests written using RestAssured library in TestNG tests structure).
* pages - (Page objects are created within this folder)
* runners - (One or more Cucumber runner classes are within this folder)
* step definitions - (Test steps implementation for the scenario steps defined in feature files)
* utils - (Any utility and support classes created within this folder)

**src/test/resources**

* features - (Cucumber feature files are created within this folder to define scenarios for the feature)
* testdata - (Test data / properties files are stored within this folder)
* testfiles - (Test Json or any dummy files are stored here)

reports - (cucumber generated reports are stored in this location)

test-output - (testng generated reports are stored in this location)

pom.xml - (Project configurations, and content/library/dependencies management and build/run configuration)

testng.xml files - (TestNG test suite management and executions control)

**Tools used**

**UI - tools used for UI test automation:**

* Maven - (Project configurations, and content/library/dependencies management and build/run configuration)
* Cucumber - (Used to define feature scenarios in gherkin, and also to create test suites and execution flow with tagging, and reporting)
* Selenium - (Used to automate the web application by implementing the step definitions, managing the page objects)
* Junit - (Used to run the cucumber scenarios with Cucumber options, and Assertions)
* JavaFaker ( to create dummy data for testing activities).
* JDBC - (Used to connect to application database for testing and verification activities)
* MySQL driver - (Used to allow connection to MySQL database that application uses)

**API - tools used to API test automation:**

* TestNG - (To create API tests and manage test suites, test execution flow, assertion and reporting)
* RestAssured - (Used to define our API tests)
* JavaFaker ( to create dummy data for testing activities).
* JDBC - (Used to connect to database for testing activities)
* MySQL driver - (Used to allow connection to MySQL database that application uses)

**Other tools used for end 2 end testing activities.**

* Git - (Source code management / version control)
* GitHub - (Remote source code management / version control platform)
* Jenkins - (CI/CD tool which we have used to run our test suites on demand or periodically)
* Eclipse - (IDE for project development)

**## Setup**

This project requires Java 11 or higher, and Java supported IDEs like Eclipse or IntelliJ.

* Clone the repository to local machine
* Open the repository on an IDE (we're using Eclipse)
* Install dependencies - (Maven install - right click on pom.xml and run as - Maven install)

**Creating tests**

Creating UI tests: [Cucumber docs](https://cucumber.io/docs/cucumber/api/?lang=java#running-cucumber)

1. Create a feature file in features folder under src/test/resources folder with extension. Feature.
2. Define scenario with cucumber key words Given, When and Then structure
3. Generate step definition snippets (use dryrun)
4. Create a steps class under step\_definitions folder and put those generated steps snippets.
5. Create page classes under pages folder and create the page objects (elements)
6. Implement the step definitions based on the scenario behavior.

**Running UI tests**

1. Create a runner class under runners folder.
2. Define the RunWith cucumber options with necessary options.
3. Pass a scenario tag to execute tests locally using cucumber tags, and runs as Junit test
4. To run the tests remotely via a CI/CD tool like Jenkins, create a build in pom.xml and point to the runner class in build configuration. (refer to existing pom.xml for examples)
5. On Jenkins job under build workflow, select top level maven target and pass the following command:

clean test -Dcucumber.filter.tags="@smoketest"

**Creating API tests:** [**RestAssured Docs**](https://rest-assured.io/)

1. Create a class in Api tests folder under src/test/java folder.
2. Create a TestNG test and define the API test flow with RestAssured.

**Running API tests**

1. Run the particular API test class as a TestNg test locally.
2. To run the API tests remotely via a CI/CD tool like Jenkins, create a TestNG xml file, and define a TestNG suite that includes certain API test classes to execute.
3. Create a profile in pom.xml that includes the Testng xml as a test suite - (refer to existing pom.xml for examples).,
4. On Jenkins job under build workflow, select top level maven target and pass the following command: