API Automation (E2E Tests)

Approach

1. I have used a hybrid framework which includes Cucumber + JAVA + Rest Assured + JUnit.

2. Benefits of this is it is very easy to read Scenario flow and easy to maintain.

Library used to automate REST API Calls

1. I have chosen Rest Assured as a Java based library to automate REST endpoints.

2. RestAssured provides a large set of built in functions for REST API's.

3. RestAssured freely available library to automate REST endpoints.

Programming language

1. I have chosen JAVA as a programming language.

2. Java is the most widely used programming language and there are a lot of libraries available to read/write JSON, excel etc.

3. We can use the same programming language for **UI as well as API** Automation.

Unit testing framework

1. I have chosen **JUnit** as a unit-testing framework because it is most compatible with cucumber.

Build tool

I have chosen **Maven** as a build management tool and using this I can manage all dependencies required to run the project.

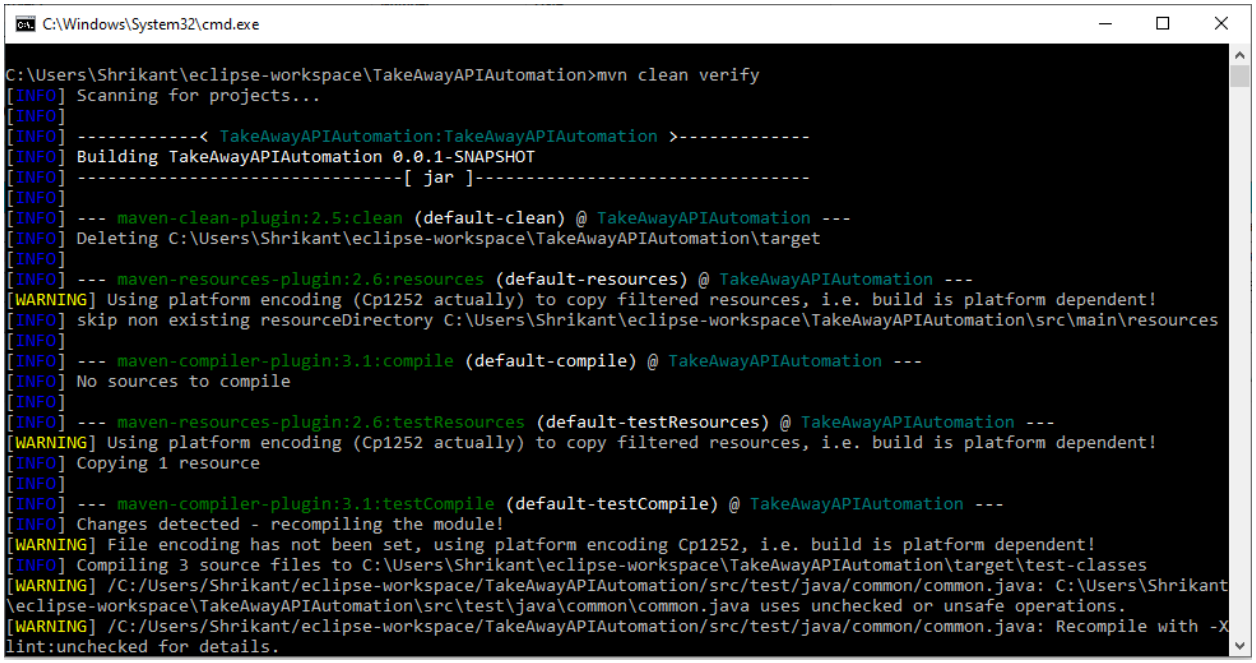
Logging Mechanism -

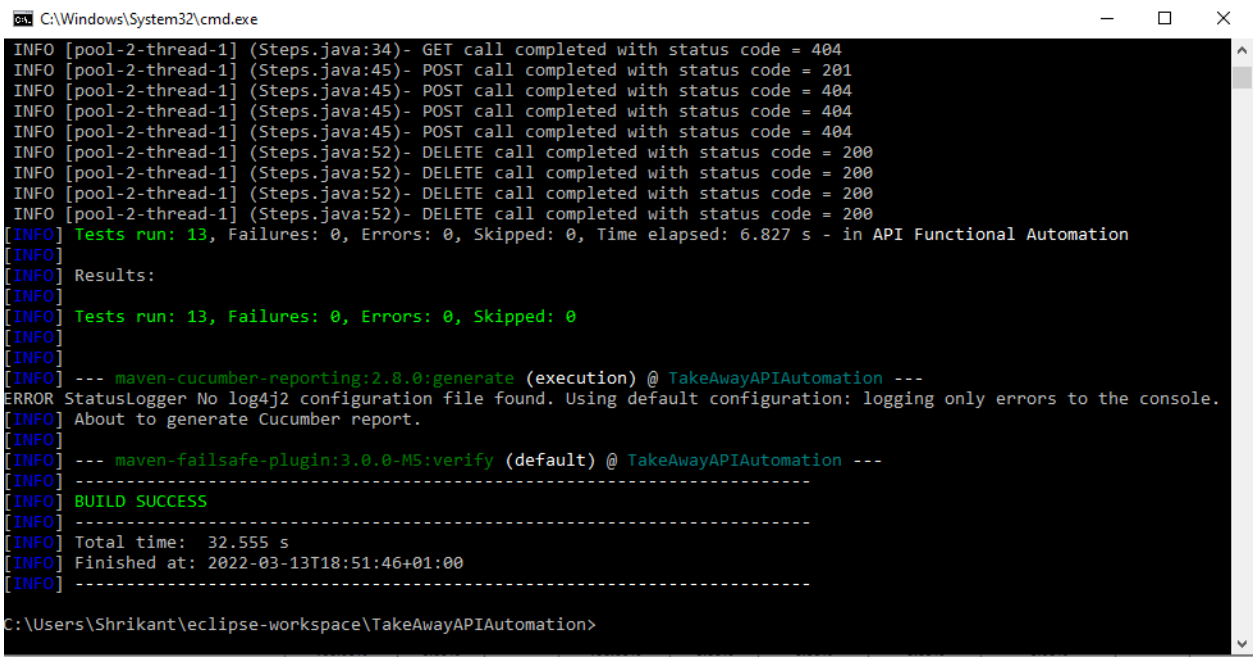
1. Use **Log4J** Java based library to generate automation logs.

How to run

1. Clone the entire project on a local machine from GitHub.

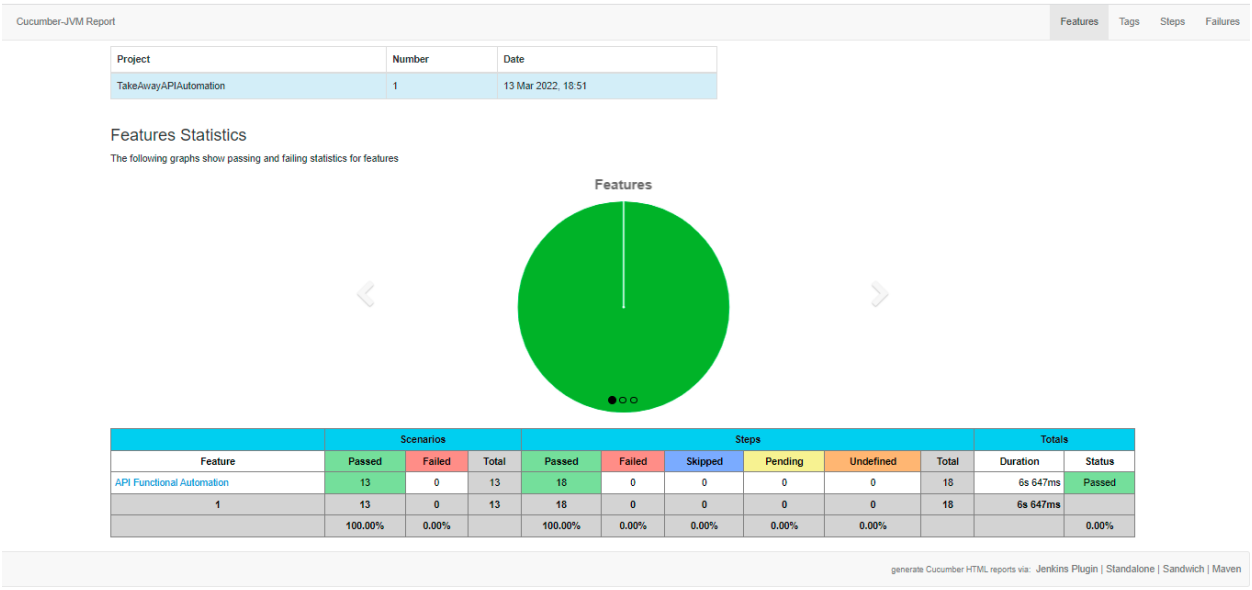
2. Execute maven command from command-line as per below screen-shot.


C:\Windows\System32\cmd.exe
C:\Users\Shrikant\eclipse-workspace\TakeAwayAPIAutomation>mvn clean verify
[INFO] Scanning for projects...
[INFO]
[INFO] ------------< TakeAwayAPIAutomation:TakeAwayAPIAutomation >-------------
[INFO] Building TakeAwayAPIAutomation 0.0.1-SNAPSHOT
[INFO] --------------------------------[jar]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ TakeAwayAPIAutomation ---
[INFO] Deleting C:\Users\Shrikant\eclipse-workspace\TakeAwayAPIAutomation\target
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ TakeAwayAPIAutomation ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory C:\Users\Shrikant\eclipse-workspace\TakeAwayAPIAutomation\src\main\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ TakeAwayAPIAutomation ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ TakeAwayAPIAutomation ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ TakeAwayAPIAutomation ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding Cp1252, i.e. build is platform dependent!
[INFO] Compiling 3 source files to C:\Users\Shrikant\eclipse-workspace\TakeAwayAPIAutomation\target\test-classes
[WARNING] /C:/Users/Shrikant/eclipse-workspace/TakeAwayAPIAutomation/src/test/java/common/common.java: C:\Users\Shrikant
\eclipse-workspace\TakeAwayAPIAutomation\src\test\java\common\common.java uses unchecked or unsafe operations.
[WARNING] /C:/Users/Shrikant/eclipse-workspace/TakeAwayAPIAutomation/src/test/java/common/common.java: Recompile with -X
lint:unchecked for details.


C:\Windows\System32\cmd.exe
INFO [pool-2-thread-1] (Steps.java:34)- GET call completed with status code = 404
INFO [pool-2-thread-1] (Steps.java:45)- POST call completed with status code = 201
INFO [pool-2-thread-1] (Steps.java:45)- POST call completed with status code = 404
INFO [pool-2-thread-1] (Steps.java:45)- POST call completed with status code = 404
INFO [pool-2-thread-1] (Steps.java:45)- POST call completed with status code = 404
INFO [pool-2-thread-1] (Steps.java:52)- DELETE call completed with status code = 200
INFO [pool-2-thread-1] (Steps.java:52)- DELETE call completed with status code = 200
INFO [pool-2-thread-1] (Steps.java:52)- DELETE call completed with status code = 200
INFO [pool-2-thread-1] (Steps.java:52)- DELETE call completed with status code = 200
[INFO] Tests run: 13, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 6.827 s - in API Functional Automation
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 13, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-cucumber-reporting:2.8.0:generate (execution) @ TakeAwayAPIAutomation ---
ERROR StatusLogger No log4j2 configuration file found. Using default configuration: logging only errors to the console.
[INFO] About to generate Cucumber report.
[INFO]
[INFO] --- maven-failsafe-plugin:3.0.0-M5:verify (default) @ TakeAwayAPIAutomation ---
[INFO] --
[INFO] BUILD SUCCESS
[INFO] --
[INFO] Total time: 32.555 s
[INFO] Finished at: 2022-03-13T18:51:46+01:00
[INFO] --
C:\Users\Shrikant\eclipse-workspace\TakeAwayAPIAutomation>

Reporting

2. HTML Reporting



<u>Current Automation Features -</u>

1. Following the **BDD approach** using cucumber and gherkin language any non-technical user can maintain/execute tests.

2. **Data and test-cases** are separate; we use a Data driven approach easy to maintain.

3. Most of the things are **configurable** ex - URL etc.

4. **HTML reporting** anyone can understand automation reports.

5. Logging mechanism using **Log4J** is easy to debug.

6. We can run tests from the **IDE + Command line** as well.

7. Execute the same tests for **multiple API paths** with the same code base.

8. All dependencies imported at **runtime** using maven as a build tool.

9. Passes expected **status codes** from feature file only and it's easily configurable.

10. Created **Reusable** methods ex - GET/POST/DELETE.

What features we can add to framework in next phase

1. Implement **parallel execution** to reduce overall execution time.

2. Integrate Automation with **Jenkins** for continuous integration.

3. Pull code at runtime from **GitHub** by Jenkins, execute, and send emailable reports to relevant audiences.

4. Create Automation dashboard to **monitor 24*7 Jenkins jobs** status.

5. Store automation reports on cloud ex – S3 bucket in AWS.

6. Manage most of Automation configuration as command-line arguments.

7. Separate Object Repository/Test Data on Cloud or any other Third Party tool for less maintenance.

8. Execute same scripts on different environments ex – Integration/Staging/Preview/Production etc.

9. Integrate Automation with different Third party tools ex – Slack for notification purpose / Integrate with JIRA to update automation results.