FRAMEWORK DOCUMENT

Definition: Framework is a set of rules and guidelines which need to be followed while converting manual test cases to automation test.

What Is use of an Automation Framework?

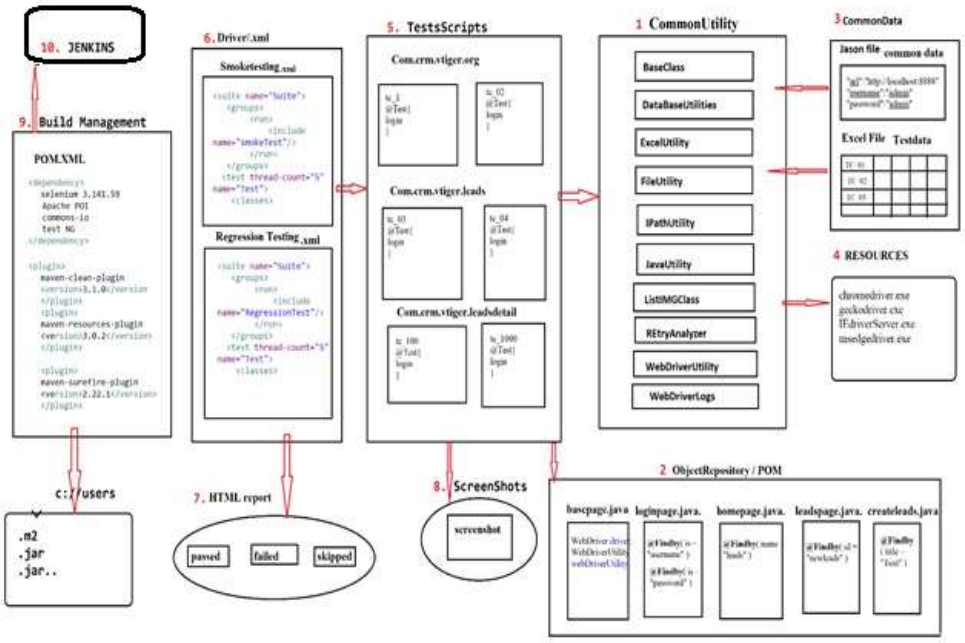
An automation framework is primarily designed to do the following:

- Enhance efficiency during the design and development of automated test scripts by enabling the reuse of components or code
- Provide a structured development methodology to ensure uniformity of design across multiple test scripts to reduce dependency on individual test-case developers
- Provide reliable issue detection and efficient root-cause analysis with minimum human intervention for the system under test
- Reduce dependence on subject matter experts by automatically selecting the test to execute according to test scenarios and dynamically refining the test scope according to changes in the test strategy or conditions of the system under test
- Improve the utilization of various resources in each cycle to enable maximum returns on effort and also ensure an uninterrupted automated testing process with minimum human intervention

Framework has the following basic components:

1. Common Utils
2. POM/object Repository
3. Test data
4. Resources
5. Test scripts
6. Driver/Xml file
7. HTML report
8. Screenshots
9. Maven
10. Jenkins

Framework architecture is depicted in the below figure



1. **Common Utils:** it is one of the common components in my Framework which can be used to any project. It contains several reusable classes like.
   a. Base class: it contains common test in JEE configuration and notations which is required for all the test scripts. As per the automation rule every test script should extend base class to use those annotations.
      - **@Before suite:** it is used to configure database connectivity and extend report, it will execute before <test> tag in XML file.
      - @**Before test**: it is used to launch the browser in case of parallel execution before executing each <test> tag in XML file.
      - @**Before Class**: it is used to launch the browser in sequential execution, it will get executed before executing each class.
      - @**before method** it is used to login into application which is executed before executing every test method.
      - @**After method** it is used to log out into a application which is executed after executing every test method.
      - @**After class** it is used to close the browser in sequential execution it will get executed after executing each class
      - @**After suite** it is used to close the database connectivity.

b. Java utils it contains Java specific methods which can be used for all the test scripts like get random data and take screenshot etc.

c. WebDriver utils: it contains WebDriver action which is common for all the test script like select switch to window switch to frame in real Framework we use WebDriver utility to perform action on the browser so that the test script level is easy and no need to remember the syntax.

d. Excel utility: it is developed using Apache POI which is used to read and write the data in Excel sheets, in this package we create methods by giving the data like test ID and leave name email and many other requirements by which we can fetch the data from Excel sheets here we also right methods to input the error messages inside the Excel sheets.

e. File utility: it is developed using commons IO package which is used to create the files whenever we encounter any error it will take the screenshot and help us to save that screenshot in our directory permanently as proof of testing.

f. ListIMGclass: it is developed by implementing ITestListener it is a type of listener which will look after the test if any test got fail, we use this class to take the screenshot and to store inside the system.

g. Retry listeners: it is an implementation class of test engineer retryanalyzer which is used to execute same script multiple times when your test script is failed.

h. WebDriver logs: it is an implementation class of test engineer WebDriver event listener in which we have all the overridden methods which help us to log any error occur during the execution.

2. **POM/Object Repository**: it is a collection of reusable web elements and business library which can be used to specific business or project. in order to develop we have used POM design pattern as per rule automation element locator should not be hardcoded in test script because maintenance and modification is TDS job whenever requirements are changed. In real -time most of the organizations follow agile process. Where frequent requirement changes should be handled, in such cases from repository helps us to maintain and modification in very much easy way.

3. **Test data.** It is one of the components in the frame or it contains the data which required to run the Framework there are two types of data.

   a. **Common data**
      i. URL used to run test scripts in different environment.
      ii. Username password which is used to run all the data in different credentials.

iii. Browser used to run all the test scripts in different browsers.

b. **Test script:** data which is required for all the test script. Every test script is dedicated with one role within excel file. As per the rule of automation data should be extracted from external sources because modification maintenance is easier.

4. Resource: It is one of the components in the Framework which contains resources to run the frameworks like chromedriver server, Firefox driver, server user guide, documents internet explorer server etc.

5. Test scripts: it contains collection of TestNG test scripts which is automated using @Test. During test script development make sure generic libraries, object repository is being used. In real time whenever manual test cases allocated to us, we should create a separate package for whole module and developer test script by taking utility.

6. Driver / XML file: it is a test NG component used to execute all the test script in batch parallel, group. In real time whenever we get a new build, we should create testing.xml and trigger the execution.

7. HTML report: whenever test execution is completed it automatically generate HTML report which helps us to know the status of application. In real time reporting component is very much important because it provides the quality of application and same report is being submitted to customer. In real- time report helps the engineer to know the root cause of the issue whenever test script is getting failed.

8. Screenshots: whenever any test script is failed during execution, we will have screenshot in screenshot folder, same screenshot can be used for debugging and rise defect in JIRA.

ADVANTAGES OF FRAMEWORK:

1. Test script development is faster and easier because of reusability.
2. Modification and maintenance of data is easy because data is stored in external resource.
3. Modification and maintenance of element is easy because we have used form design pattern to maintain elements in well-organized way.
4. POM design pattern is a perfect fit for Agile process.
5. Framework provide automatic screenshot for failed script.
6. Framework provides flexibility to achieve cross-browser testing, distributed environment testing, smoke testing, regression testing, regional regression testing.
7. Framework provide accurate execution report for every execution
8. Framework provides generic reusable utility for all actions like Excel utility base class database utility.
9. Test script can be used for every new build.
10. Test script is optimized.

DISADVANTAGES OF FRAMEWORK:

1. Should be good in programming.
2. Very hard to find errors.

PHASES OF FRAMEWORK: There are 3 phases in framework

1. **Framework design Phase:** in this phase the Framework developer will design the Framework which contain generic libraries Pom classes with partial elements test data template this phase is executed in print one or released one.

2. **Framework implementation phase:** this starts with ring sprint 2 to all the automation engineer will start writing the test scripts utilizing the framework

3. **Framework execution phase:** this phase is usually handled by JENKINS tool; this is when we get a new build to the testing environment JENKINS will automatically start the execution and it will send the email notification for the concerned engineer about the status of execution.