**Framework Document**

**Framework Introduction:** -Here we used Data Driven Framework, using BDD -Cucumber design pattern.

This BDD Cucumber framework divided in to three main components: -

1. Feature File
2. Step Definition file
3. Runner file
4. **Feature File: -** In Feature File, we write the test cases (Scenarios), using Gherkin Language (Feature, Scenario, Scenario Outline…Examples, Given, When, Then, And, But).

Gherkin language is written in simple English language, So any Non-Technical person can understand that what we are covering in written scenario.

1. **Steps Definition file: -**It is java file, which contains Java and selenium code. It provides the definition to Feature file
2. **Runner file: -** It is Junit file, which works as an execution engine.

* In this file we link Feature file and Steps definition file using “features” and “glue” cucumber options
* In this file we define all Cucumber Options, like plugin, monochrome, strict, dryRun etc.

**Tool used: -**

1. Eclipse (IDE)
2. Maven Project
3. Chrome Browser

**Maven dependencies used( in pom.xml) -**

1. cucumber-java
2. cucumber-jvm
3. cucumber-junit
4. cucumber-core
5. cucumber-jvm-deps
6. cucumber-reporting
7. gherkin
8. junit-jupiter-api
9. selenium-java
10. cucumber-picocontainer

**How to run code (Steps): -**

1. Open Eclipse (IDE)
2. Import project in Eclipse
3. Run as Junit Test
4. Check the test result (Junit or Cucumber report)

**4.Validate the response through**[OpenWeather API](https://openweathermap.org/api" \t "_blank)**:-**

**Scenario: - Weather details calling API using “city name”**

* API Name-> URI: -https://api.openweathermap.org/data/2.5/weather?q=London,uk
* API Key: -2154b9a96178ba901814bec2b214e29d
* Request type:-GET

**Steps: -**

step1: - send request

step2: - Print the response

step3: - Display the HTTP status code & response time

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**If you had more time: -**

**1)Could make framework more robust using following feature**

**-** By creating config file, remove hard coded value like URL, browser etc.

- Could make TestUtil file, which covers more common utility. like -open selected browser (Chrome /FF), Maximize window, define wait etc.

-Could import HTML Extend report for advance reporting.