Cucumber Terminologies:

What is Gherkin?  It is a [Business Readable, Domain Specific Language](https://martinfowler.com/bliki/BusinessReadableDSL.html)  that lets you describe software's behaviour.

Example: Pop up message is displayed when buttons are clicked and errors are gone

Cucumber is a framework which supports BDD.

Keywords Used in Cucumber: **Scenario, Feature, Feature file, Scenario outline, Step Definition, Background**

Scenarios:

In Cucumber Testcases are represented as Scenarios.

Scenarios contain Steps which are equivalent to test Steps and use the following keywords (Gherkin syntax) to denote them: Given, When, Then, But, and And (case sensitive).

* **Given**: Preconditions are mentioned in the Given keyword
* **When**: The purpose of the When Steps is to describe the user action.
* **Then**: The purpose of Then Steps is to observe the expected output. The observations should be related to the business value/benefit of your Feature description.

When we specify a business requirement, sometimes there are multiple pre-conditions, user actions, and expected outcomes

we are going to add one more Scenario and will use the And and But keywords:

* And: This is used for statements that are an addition to the previous Steps and represent positive statements.
* But: This is used for statements that are an addition to previous Steps and represent negative statements.

Feature and Feature File:

Feature represents Business requirement.

Feature File acts as a Test Suite which consists of all Scenarios.

In Cucumber, Feature files contain Scenarios. We can simply create feature file with. feature extension

Scenarios belonging to specific area of Application will be grouped into one Feature file

The text that immediately follows the Feature keyword, and is in the same line, is the Title of the Feature file

 Feature file should contain either Scenario or Scenario Outline. The naming conventions for Feature files should be lowercase with. feature extension

 The Cucumber Framework: BDD Framework for Selenium

Cucumber BDD framework mainly consists of three major parts – Feature File, Step Definitions, and the Test Runner File.

### 1. Feature File

A standalone unit or a single functionality (such as a login) for a project can be called a Feature. Each of these features will have scenarios that must be tested using Selenium integrated with Cucumber. A file that stores data about features, their descriptions, and the scenarios to be tested is called a Feature File.

Cucumber tests are written in these Feature Files that are stored with the extension – “.feature”. A Feature File can be given a description to make the documentation more legible.

Example:

The Login function on a website

Feature File Name: userLogin.feature

Description: The user shall be able to login upon entering the correct username and password in the correct fields. The user should be directed to the homepage if the username and password entered are correct.

Keywords such as GIVEN, WHEN, and THEN used to write the test in Cucumber are called Annotations.

GIVEN user navigates to login page by opening Firefox

WHEN user enters correct <username> AND <password> values

THEN user is directed to the homepage

### 2. Step Definitions

Now that the features are written in the feature files, the code for the related scenario has to be run. To know which batch of code needs to be run for a given scenario, Steps Definitions come into the picture. A Steps Definitions file stores the mapping data between each step of a scenario defined in the feature file and the code to be executed.

Step Definitions can use both Java and Selenium commands for the Java functions written to map a feature file to the code.

Example:

package StepDefinition;

import cucumber.api.java.en.Given;

import cucumber.api.java.en.Then;

import cucumber.api.java.en.When;

public class Steps

{

@Given("^user navigates to the login page by opening Firefox$")

//Code to Open Firefox Browser and launch the login page of application to define the GIVEN step of the feature

@When("^user enters correct username and password values$")

//take inputs for username and password fields using find element by xpath. Put the correct username and password values as inputs to define the WHEN step of the feature

@Then (“^user gets directed to homepage$”)

//Direct to the Homepage of the application as a result of correct username and password inputs in the WHEN step. This would define the THEN step of the feature

### 3. Test Runner File

To run the test, one needs a Test Runner File, which is a JUnit Test Runner Class containing the Step Definition location and the other primary metadata required to run the test.

The Test Runner File uses the @RunWith() Annotation from JUnit for executing tests. It also uses the @CucumberOptions Annotation to define the location of feature files, step definitions, reporting integrations, etc.

Example:

Test Runner Class in cucumberTest package, with the feature files in “src/test/Feature” location and Step Definition files in “src/main/stepDefinition” folder.

**package** cucumberOptions;

**import** org.junit.runner.RunWith;

**import** io.cucumber.junit.Cucumber;

**import** io.cucumber.junit.CucumberOptions;

@RunWith(Cucumber.**class**)

@CucumberOptions(

features = "src/test/java/features"

,glue={"stepDefinitions"}

)

**public** **class** TestRunnerJunit {

}