-Lines starting with the keyword Feature

-it just gives you a convenient place to put some summary documentation about the group of tests

Scenario:

-A feature usually contains a list of scenarios.

In valid Gherkin, a Feature must be followed by one of the following:

• Scenario

• Background

• Scenario Outline

-Every scenario consists of a list of [steps](http://docs.behat.org/en/v2.5/guides/1.gherkin.html#steps), which must start with one of the keywordsGiven, When, Then, But or And.

-In addition to basic [scenarios](http://docs.behat.org/en/v2.5/guides/1.gherkin.html#scenarios), feature may contain [scenario outlines](http://docs.behat.org/en/v2.5/guides/1.gherkin.html#scenario-outlines) and [backgrounds](http://docs.behat.org/en/v2.5/guides/1.gherkin.html#backgrounds).

-Scenario is one of the core Gherkin structures.

-Every scenario starts with the Scenario: keyword , followed by an optional scenario title.

When Cucumber runs a scenario, if the system behaves as described in the

scenario, then the scenario will pass; if not, it will fail.

Background:

A *background* section in a feature file allows you to specify a set of steps that

are common to every scenario in the file.

Scenario outline:

We can use a scenario outline to specify the steps once

and then play multiple sets of values through them.

We indicate *placeholders* within the scenario outline using angle brackets

(<..>) where we want real values to be substituted.

The scenario outline itself

is useless without a table of Examples, which lists rows of values to be substituted

for each placeholder.

You can have any number of Scenario Outline elements in a feature and any

number of Examples tables under each scenario outline. Behind the scenes,

behat converts each row in the Examples table into a scenario before executing

it. You can prove this to yourself by using the –expand

grouping:

You can use [tags](http://docs.behat.org/en/v2.5/guides/1.gherkin.html#tags) to group features and scenarios together, independent of your file and directory structure.

Step definitions:

When you’re executing the feature, the trailing portion of each step is matched to a regular expression, which executes a PHP callback function. That we call as step definitions.

About mink:

Browser is the window through which web users interact with web applications and other users. Users are always talking with web applications through browsers.

So, in order to test that our web application behaves correctly, we need a way to simulate this interaction between the browser and the web application in our tests. We need a **Mink**.