Cucumber (BDD) Test Automation Framework

**Scope of the project:**

BDD (Behaviour-Driven Development) Behaviour-driven development is a testing practice that follows the idea of specification by example (e.g., Test Driven Development - TDD)

The focus of the Cucumber (BDD) is to explain how the application behaves in simple user/business-focused language (with the help of feature files)

**Framework Goals and Objectives:**

* Ability to maintain test cases in human readable format (any developer, tester, business people or managers can understand)
* Ease of use – requires 1 day of training
* Portability - It can be used for any other applications with little or no modifications
* All the automated test scripts are independent and no impact on other test scripts, if any failures
* Minimize required Developer resources to basic framework maintenance activities.
* All the test cases are written in a simple plain English language called **Gherkin**
* Followed Page Object Model while designing the framework to increase the readability of the framework

**Technologies Used:**

* Java (JDK11.0) with JUNIT
* Selenium (3.14)
* Gherkin for Cucumber BDD

**Automation Framework Components:**

* Feature Files
* Resources
* Application Pages
* Step Definition files
* Test Runner File
* Test Reports (of different formats)

**Feature Files:**

* Feature files are the test suites which contain related test scenarios and makes use of **.feature** extension. We make use of **Gherkin** language to write the scenarios in plain English language. Please check example below

**Resources:**

* Resources is the package which contain any tool specific files, application specific test data files or any other miscellaneous files which are being used while running the scripts

**Application Pages:**

* Here the application is divided into independent modules such as login page, home page and profile page. And then respective class files are created which cover the functionalities of respective module

**Step Definition files:**

* Based on the feature files we write, the matching glue code is generated and kept inside this package. These are the main building blocks which embed the selenium code inside this. And all the methods written inside the Step Definition files are unique in nature and can be called infinite times depending on the feature file requirements

**Test Runner File:**

* This is the entry criteria while the tester runs the cucumber scenarios. Here we setup which scenarios to be run and which need to be ignored. And also we can specify the reporting mechanism once the scenarios are executed

**Test Reports (of different formats)**

* This is the folder which contain all the test reports in various formats such as **.html**, **.json** and **.xml**. We can refresh the project and get the latest test reports as soon as the scenarios are executed

**Repo Details:**

// Mention repo URL here

**How to run sample scenario?**

1. Make sure you have setup the matching browser and its respective driver file
2. Go to respective feature file and update the scenario tag (example @Regression)
3. Go to TestRunner.Java and update the tags attribute with above tag
4. Right click on the TestRunner.Java and run as JUNIT
5. Find that script is loaded and executed
6. Refresh the project and navigate to target folder
7. Make sure test reports are generated for all the scenario mentioned in Step 2