1. Definition of cucumber?
   - Cucumber is a popular behavior-driven development (BDD) tool used in software testing.
   - It enables teams to write executable specifications in plain text, promoting collaboration between non-technical stakeholders and developers.
   - QA professionals use Cucumber to create and automate test scenarios, ensuring that software functions as described in these specifications.

2. Definition of Gherkin?
   Gherkin is a simple, human-readable language used in behavior-driven development (BDD) to write structured descriptions of software features and test cases. It serves as the foundation for creating executable test scenarios in tools like Cucumber.

3. Definations
   - ➢ Feature - The feature file starts with the keyword Feature. Under feature, you can mention the feature name which is to be tested
   - ➢ Scenario - Each scenario starts with a keyword Scenario There can be one or more scenarios in a feature file
     Given - It is used to define the pre-condition in our test
   - ➢ When - It is used to define an action And - It is used to connect two statements and provides the logical
   - ➢ AND condition between any two statements
   - ➢ Then - It is used to define the final outcome or for validation
   - ➢ But - It defines the negative assertion

4. What are tags of feature
   - ❖ features: Specifies the path to the directory or file containing Cucumber feature files that define the behavior to be tested.
   - ❖ dryRun: A flag that, when set to "false," indicates that Cucumber should execute the tests; when set to "true," it checks the feature file's syntax and steps without actually running the tests.
   - ❖ snippets: Specifies the code style for automatically generating missing step definition snippets when running Cucumber tests. In this case, it's set to "CAMELCASE," which means step snippets will be generated using camel case naming conventions.
   - ❖ monochrome: When set to "true," this option makes the console output of Cucumber tests more readable by removing unnecessary characters and applying color formatting.
   - ❖ glue: Specifies the package or path where step definitions are located. Cucumber uses this information to link the steps in feature files with the corresponding Java code. In your example, step definitions are expected to be in the "steps" package.

```
@CucumberOptions(
      features = {"src/test/java/features/login.feature"},
      dryRun = false,
      snippets = CucumberOptions.SnippetType.CAMELCASE,
      monochrome = true,
      glue = "steps"
)
```

```
//AbstractTestNGCucumberTests - use for testng to run as TestNG
public class Runner extends AbstractTestNGCucumberTests {

}
```

First scenario

Feature file : login.feature

```
Feature: Bookcart application tests

  Scenario: Login should be success

    Given User navigate to Bookcart application
    And User clicks on login button
    And User enter the username as ortoni
    And User enter the password as Pass1234
    When User click on login button
    Then Login should be success
```

Class: LoginSteps.java

```
package steps;

import io.cucumber.java.en.And;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
import io.github.bonigarcia.wdm.WebDriverManager;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

import java.time.Duration;

public class LoginSteps {

    WebDriver driver;


    @Given("User navigate to Bookcart application")
    public void user_navigate_to_bookcart_application() {
        WebDriverManager.chromedriver().setup();
        driver= new ChromeDriver();
        driver.get("https://bookcart.azurewebsites.net/");
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));
        driver.manage().window().maximize();
        System.out.println("Title: "+driver.getTitle());
    }

    @And("User clicks on login button")
    public void user_clicks_on_login_button() {
        driver.findElement(By.xpath("//span[normalize-space()='Login']")).click();
    }

    @And("User enter the username as ortoni")
    public void user_enter_the_username_as_ortoni() {
        driver.findElement(By.xpath("//input[@id='mat-input-0']")).sendKeys("ortoni");
    }

    @And("User enter the password as Pass1234")
    public void user_enter_the_password_as_pass1234() {
        driver.findElement(By.xpath("//input[@id='mat-input-1']")).sendKeys("Pass1234");
    }

    @When("User click on login button")
    public void user_click_on_login_button() {
        driver.findElement(By.xpath("(//span[@class='mat-button-wrapper' and
text()='Login'])[2]")).click();
    }

    @Then("Login should be success")
    public void login_should_be_success() {
        WebElement userEle = driver.findElement(By.
                 xpath("//button[contains(@class,'mat-focus-indicator mat-menu-trigger')]//span[1]"
                ));
        driver.quit();
    }

}
```

Runner class

```
package testRunner;

import io.cucumber.testng.AbstractTestNGCucumberTests;
import io.cucumber.testng.CucumberOptions;

@CucumberOptions(
        features = {"src/test/java/features/login.feature"},
        dryRun = false,
        snippets = CucumberOptions.SnippetType.CAMELCASE,
        monochrome = true,
        glue = "steps"
)
```

```
//AbstractTestNGCucumberTests - use for testng to run as TestNG
public class Runner extends AbstractTestNGCucumberTests {

}
```

Some more

- tags: You can use tags to selectively run specific scenarios or features. For example, you can tag scenarios with "@smoke" or "@regression" and then run only scenarios with those tags.
- format: This option allows you to specify the format for the output reports generated by Cucumber, such as "pretty," "html," "json," "junit," etc.
- strict: When set to "true," this option causes Cucumber to fail the execution if any step definitions are undefined or pending.
- plugin: Similar to "format," you can use this option to specify output plugins for various report formats.
- name: You can filter scenarios to run based on their names by using the "name" option. For example, you can run scenarios that match a specific regular expression pattern.
- dryRunSnippet and strictCucumber: These options are used for generating step definition snippets and enforcing strict Cucumber compliance, respectively.
- tagsExpression: An alternative way to specify which scenarios to run based on tag expressions.
- order: Specifies the order in which scenarios are executed, such as "defined," "random," or "random:<SEED>."
- gluePath: Specifies additional package paths where Cucumber should look for step definitions.
- background: Defines a background section in feature files to specify common steps that should be executed before each scenario.

==================================

Naveen Automation

Repo :
https://github.com/naveenanimation20/Cucumber6LatestFeatures/tree/master/src/test/resources/AppFeatures

Basic Sample feature file

Pom.xml

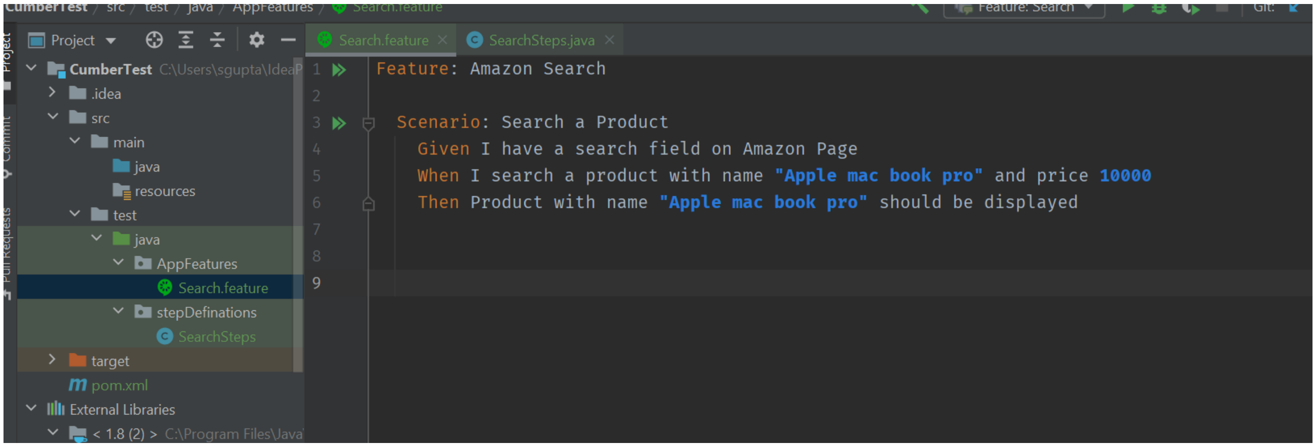
```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
```

```
<modelVersion>4.0.0</modelVersion>

<groupId>cucumber1</groupId>
<artifactId>CumberTest</artifactId>
<version>1.0-SNAPSHOT</version>

<properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
    <cucumber.version>7.0.0</cucumber.version>
</properties>

<dependencies>
    <!-- Cucumber dependencies -->
    <dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-java</artifactId>
        <version>${cucumber.version}</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-junit</artifactId>
        <version>${cucumber.version}</version>
        <scope>test</scope>
    </dependency>

    <!-- JUnit dependency -->
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>3.2.2</version>
            <configuration>
                <includes>
                    <include>**/*Test.java</include>
                </includes>
            </configuration>
        </plugin>
```

```
            <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.8.1</version>
            </plugin>
        </plugins>
    </build>
</project>
```

Basic one

Create a feature package and create a feature file in it



```
Feature: Amazon Search

  Scenario: Search a Product
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed
```

Search.feature

```
Feature: Amazon Search

 Scenario: Search a Product
   Given I have a search field on Amazon Page
   When I search a product with name "Apple mac book pro" and price 10000
   Then Product with name "Apple mac book pro" should be displayed
```

Create a stepDefinations package and provide definaitions

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class SearchSteps {

  @Given("I have a search field on Amazon Page")
  public void i_have_a_search_field_on_amazon_page() {
      System.out.println("Step 1: I have a search field on Amazon Page");
  }
  @When("I search a product with name {string} and price {int}")
  public void i_search_a_product_with_name_and_price(String productName, Integer price) {
        System.out.println("Step 2: I search a product with name : "+productName+" and price : 
"+price);

    }
    @Then("Product with name {string} should be displayed")
    public void product_with_name_should_be_displayed(String productName) {
        // Write code here that turns the phrase above into concrete actions
        System.out.println("Step 3: I have a search field on Amazon Page: "+productName);

    }

}
```

On run as feature file

```
C:/Users/sgupta/IdeaProjects/CumberTest/src/test/java/AppFeatures/Search.feature
Testing started at 11:21 ...
Step 1: I have a search field on Amazon Page
Step 2: I search a product with name : Apple mac book pro and price : 10000
Step 3: I have a search field on Amazon Page: Apple mac book pro

1 Scenarios (1 passed)
3 Steps (3 passed)
0m1.611s
=========
```

Now providing a real time scenario - We have to search the product “ Apple mac book pro” in the product list.

For thar we are creating the product class and search class
Create a package in src/main/java >> amazonImplementations >> Product.java

Product.java

```
package amazonImplementations;

import java.util.ArrayList;
import java.util.List;

public class Product {

    private String productName;     //encapsulation
    private int price;

    public Product(String productName, int price) {
        this.productName = productName;
        this.price = price;
    }
}               public String getProductName() {
                    return productName;
                }

                public void setProductName(String productName) {
                    this.productName = productName;
                }

                public int getPrice() {
                    return price;
                }

                public void setPrice(int price) {
                    this.price = price;
                }


                public List<String> getProductList()
                {
                    List<String> productList = new ArrayList<>();
                    productList.add("Apple mac book pro");
                    productList.add("Apple mac book air");
                    productList.add("Apple mac iphone12");
                    return productList;
                }


            }
```

Search.class

```
            package amazonImplementations;

public class Search {

    //method to return and search product in thr product list and matching it with
    //search steps product constructor

    public String displayProduct(Product tempProduct)
    {
        if (tempProduct.getProductList().contains(tempProduct.getProductName()))
        {
            return tempProduct.getProductName();
        }
        return null;
    }

}
```

SearchSteps.java

```
package stepDefinations;

import amazonImplementations.Product;
import amazonImplementations.Search;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
import org.junit.Assert;

public class SearchSteps {

    Product product;
    Search search;

    @Given("I have a search field on Amazon Page")
    public void i_have_a_search_field_on_amazon_page() {
        System.out.println("Step 1: I have a search field on Amazon Page");
    }

    @When("I search a product with name {string} and price {int}")
    public void i_search_a_product_with_name_and_price(String productName, Integer price) {
        System.out.println("Step 2: I search a product with name : " + productName + " and price : " + price);

        //Calling constructor of product class
        product = new Product(productName, price);

    }

    @Then("Product with name {string} should be displayed")
    public void product_with_name_should_be_displayed(String productName) {
        // Write code here that turns the phrase above into concrete actions
        System.out.println("Step 3: I have a search field on Amazon Page: " + productName);

        // Searching the product in the Product list of Passing object of Product class in Search class
        search = new Search();

        //Search and display
        String name_Product = search.displayProduct(product);
        System.out.println("searched product is: " + name_Product);

        //Assertion
        Assert.assertEquals(product.getProductName(), name_Product);
    }

}
```

Output

```
Testing started at 12:28 ...
Step 1: I have a search field on Amazon Page
Step 2: I search a product with name : Apple mac book pro and price : 10000
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro

1 Scenarios (1 passed)
3 Steps (3 passed)
```

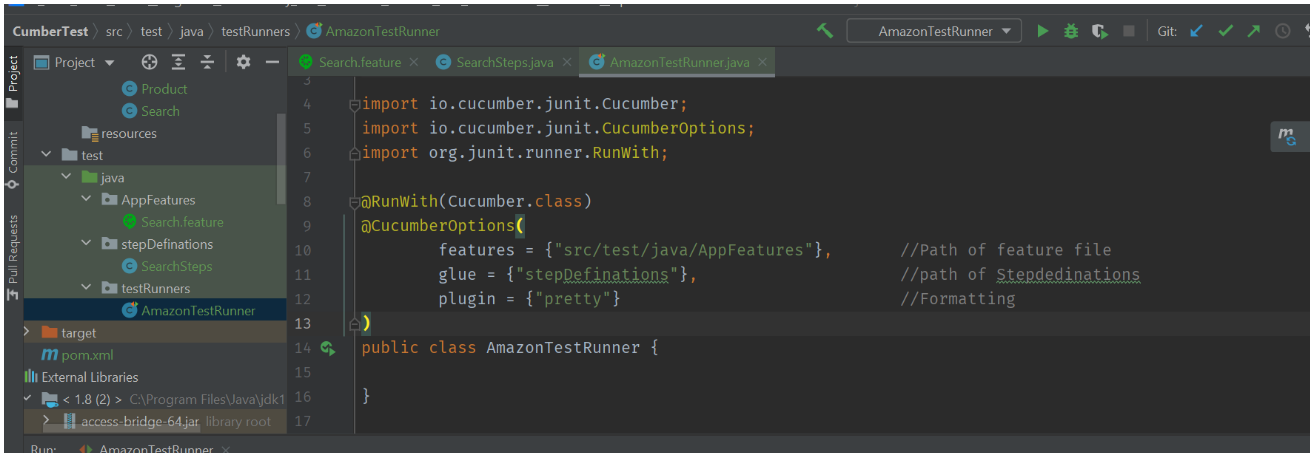
—---**********************---

Notes/QA

Notes/QA

Runner class: It is created to avoid multiple feature file run in one go.

Create a package testRunners>> AmazonTestRunner



```
import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/AppFeatures"},       //Path of feature file
        glue = {"stepDefinations"},                     //path of Stepdedinations
        plugin = {"pretty"}                             //Formatting
)
public class AmazonTestRunner {

}
```

AmazonTestRunner.java

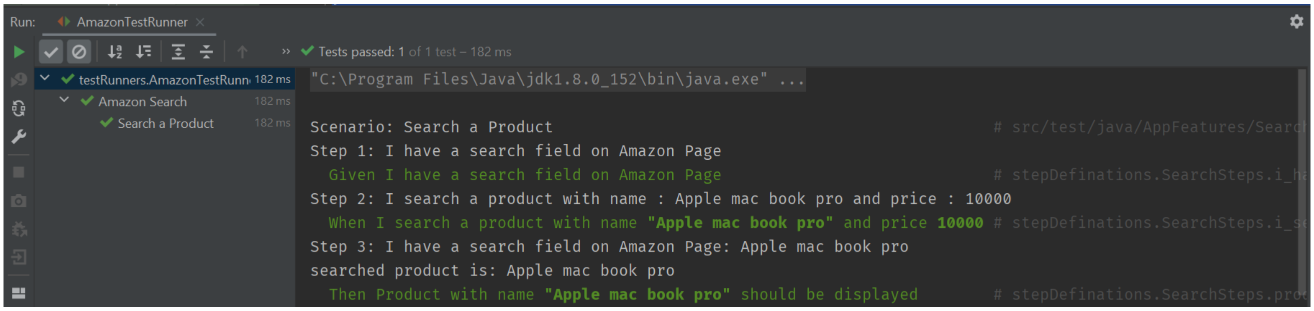
```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/AppFeatures"},       //Path of feature file
        glue = {"stepDefinations"},                     //path of Stepdedinations
        plugin = {"pretty"}                             //Formatting
)
public class AmazonTestRunner {

}
```

- tags: You can use tags to selectively run specific scenarios or features. For example, you can tag scenarios with "@smoke" or "@regression" and then run only scenarios with those tags.
- format: This option allows you to specify the format for the output reports generated by Cucumber, such as "pretty," "html," "json," "junit," etc.
- strict: When set to "true," this option causes Cucumber to fail the execution if any step definitions are undefined or pending.

- plugin: Similar to "format," you can use this option to specify output plugins for various report formats.
- name: You can filter scenarios to run based on their names by using the "name" option. For example, you can run scenarios that match a specific regular expression pattern.
- dryRunSnippet and strictCucumber: These options are used for generating step definition snippets and enforcing strict Cucumber compliance, respectively.
- tagsExpression: An alternative way to specify which scenarios to run based on tag expressions.
- order: Specifies the order in which scenarios are executed, such as "defined," "random," or "random:<SEED>."
- gluePath: Specifies additional package paths where Cucumber should look for step definitions.
- background: Defines a background section in feature files to specify common steps that should be executed before each scenario.

Output on running AmazonTestRunner.java

```
Scenario: Search a Product                                  #
src/test/java/AppFeatures/Search.feature:3
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                         #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang
.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
```



Run:
AmazonTestRunner
Tests passed: 1 of 1 test – 182 ms
testRunners.AmazonTestRunn 182 ms
Amazon Search 182 ms
Search a Product 182 ms

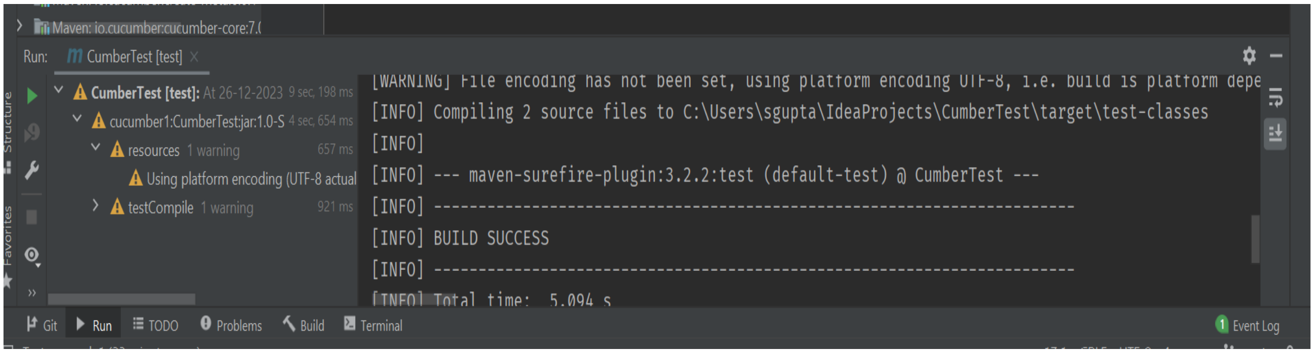
```
"C:\Program Files\Java\jdk1.8.0_152\bin\java.exe" ...

Scenario: Search a Product                                        # src/test/java/AppFeatures/Searc
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                      # stepDefinations.SearchSteps.i_ha
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 # stepDefinations.SearchSteps.i_se
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed  # stepDefinations.SearchSteps.pro
```

=============

Notes/QA

Running test with maven

Right click and run as test

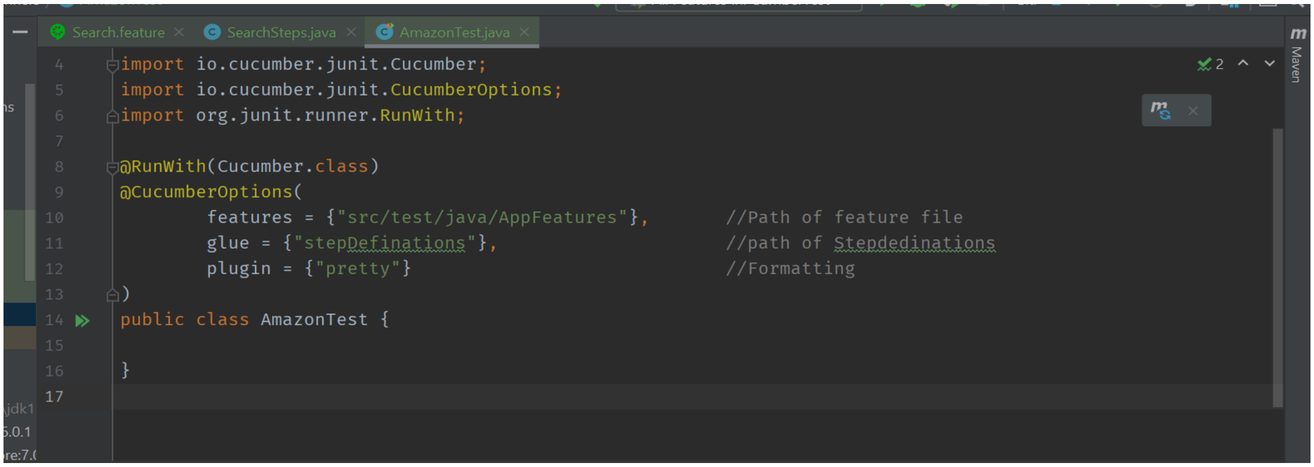


```
CumberTest [test]: At 26-12-2023 9 sec, 198 ms
  cucumber1:CumberTest:jar:1.0-S 4 sec, 654 ms
    resources 1 warning 657 ms
      Using platform encoding (UTF-8 actual
    testCompile 1 warning 921 ms
```

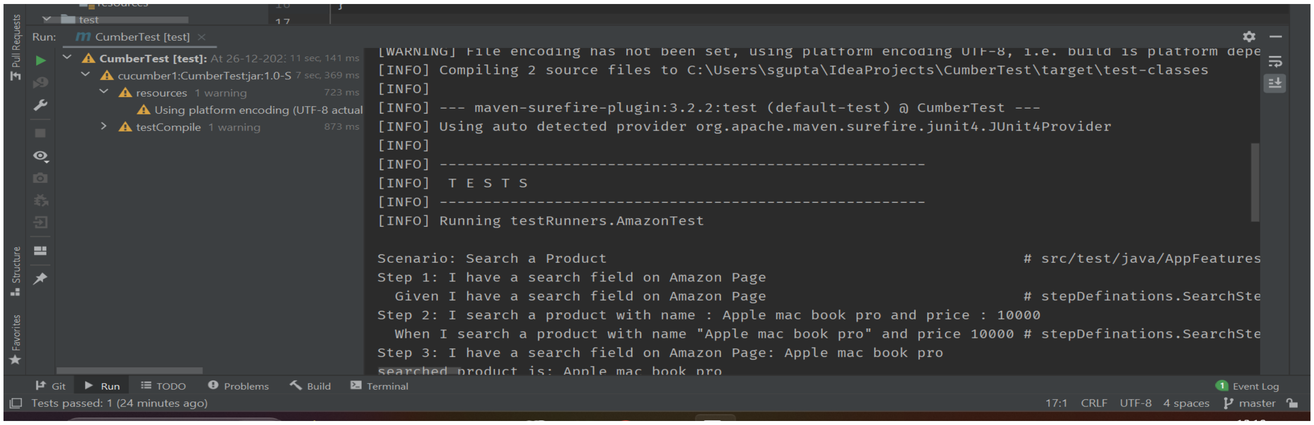
```
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform depe
[INFO] Compiling 2 source files to C:\Users\sgupta\IdeaProjects\CumberTest\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:3.2.2:test (default-test) @ CumberTest ---
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  5.094 s
```

But it is not showed test output instead test runs, the reason is the name of runner class contains "Runner". So that we need to replace it from "AmazonTestRunner" to "AmazonTest"

Now run again



```java
import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/AppFeatures"},       //Path of feature file
        glue = {"stepDefinations"},                     //path of Stepdedinations
        plugin = {"pretty"}                             //Formatting
)
public class AmazonTest {

}
```



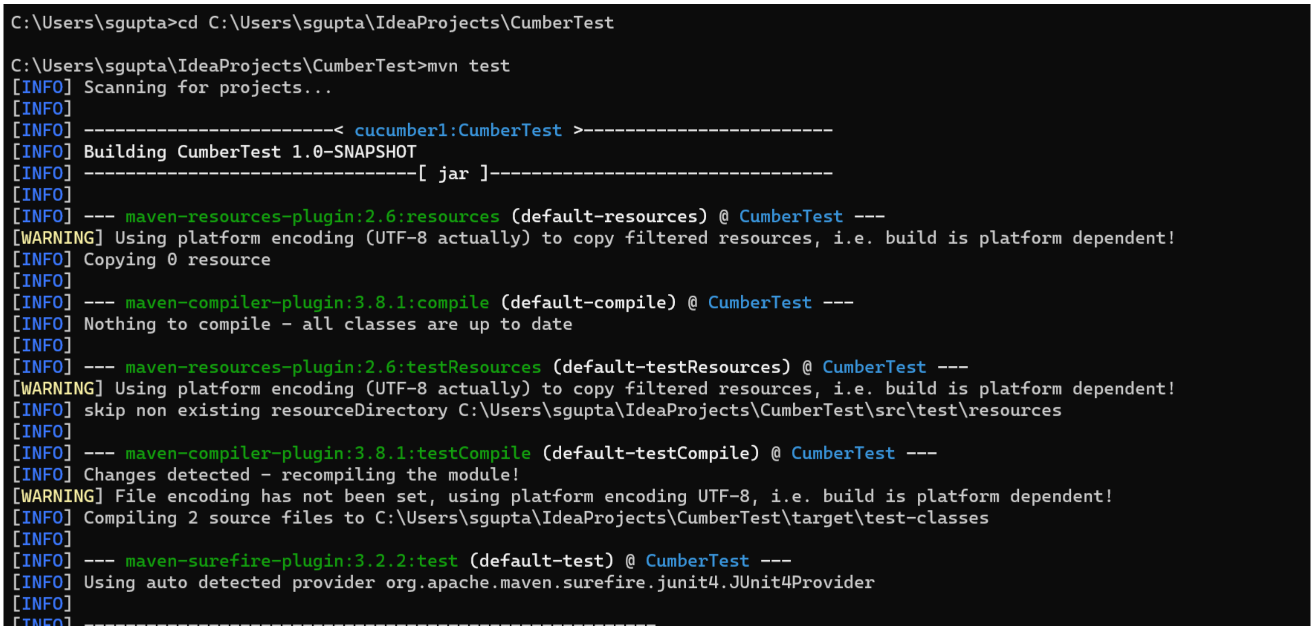
```
CumberTest [test]: At 26-12-202: 11 sec, 141 ms
  cucumber1:CumberTest:jar:1.0-S 7 sec, 369 ms
    resources 1 warning 723 ms
      Using platform encoding (UTF-8 actual
    testCompile 1 warning 873 ms
```

```
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform depe
[INFO] Compiling 2 source files to C:\Users\sgupta\IdeaProjects\CumberTest\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:3.2.2:test (default-test) @ CumberTest ---
[INFO] Using auto detected provider org.apache.maven.surefire.junit4.JUnit4Provider
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running testRunners.AmazonTest

Scenario: Search a Product                                             # src/test/java/AppFeatures
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                           # stepDefinations.SearchSte
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 # stepDefinations.SearchSte
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
```

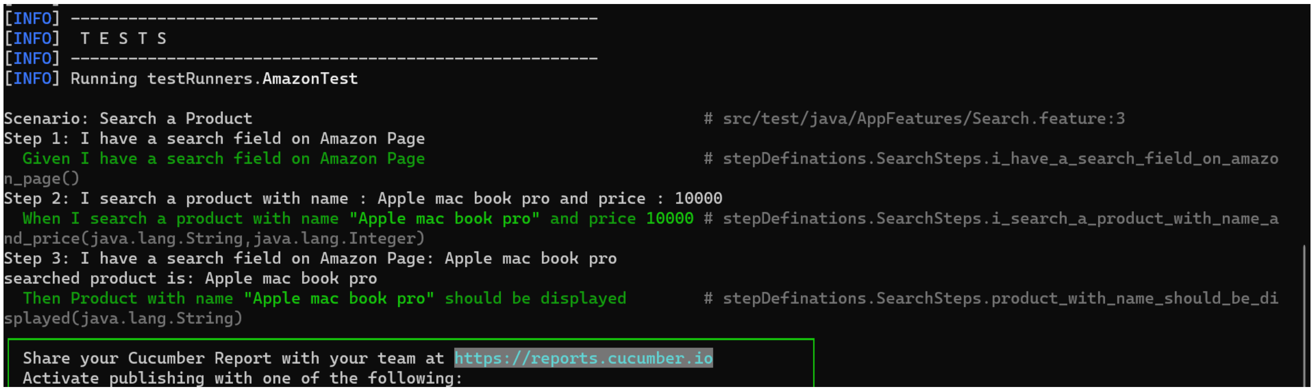
Tests passed: 1 (24 minutes ago)

Using command line

```
C:\Users\sgupta>cd C:\Users\sgupta\IdeaProjects\CumberTest
C:\Users\sgupta\IdeaProjects\CumberTest>mvn test
```



```
C:\Users\sgupta>cd C:\Users\sgupta\IdeaProjects\CumberTest

C:\Users\sgupta\IdeaProjects\CumberTest>mvn test
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------< cucumber1:CumberTest >------------------------
[INFO] Building CumberTest 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ CumberTest ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ CumberTest ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ CumberTest ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory C:\Users\sgupta\IdeaProjects\CumberTest\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ CumberTest ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 2 source files to C:\Users\sgupta\IdeaProjects\CumberTest\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:3.2.2:test (default-test) @ CumberTest ---
[INFO] Using auto detected provider org.apache.maven.surefire.junit4.JUnit4Provider
[INFO]
[INFO] -------------------------------------------------------
```



```
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running testRunners.AmazonTest

Scenario: Search a Product                                            # src/test/java/AppFeatures/Search.feature:3
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                          # stepDefinations.SearchSteps.i_have_a_search_field_on_amazo
n_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 # stepDefinations.SearchSteps.i_search_a_product_with_name_a
nd_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed       # stepDefinations.SearchSteps.product_with_name_should_be_di
splayed(java.lang.String)

 Share your Cucumber Report with your team at https://reports.cucumber.io
 Activate publishing with one of the following:
```

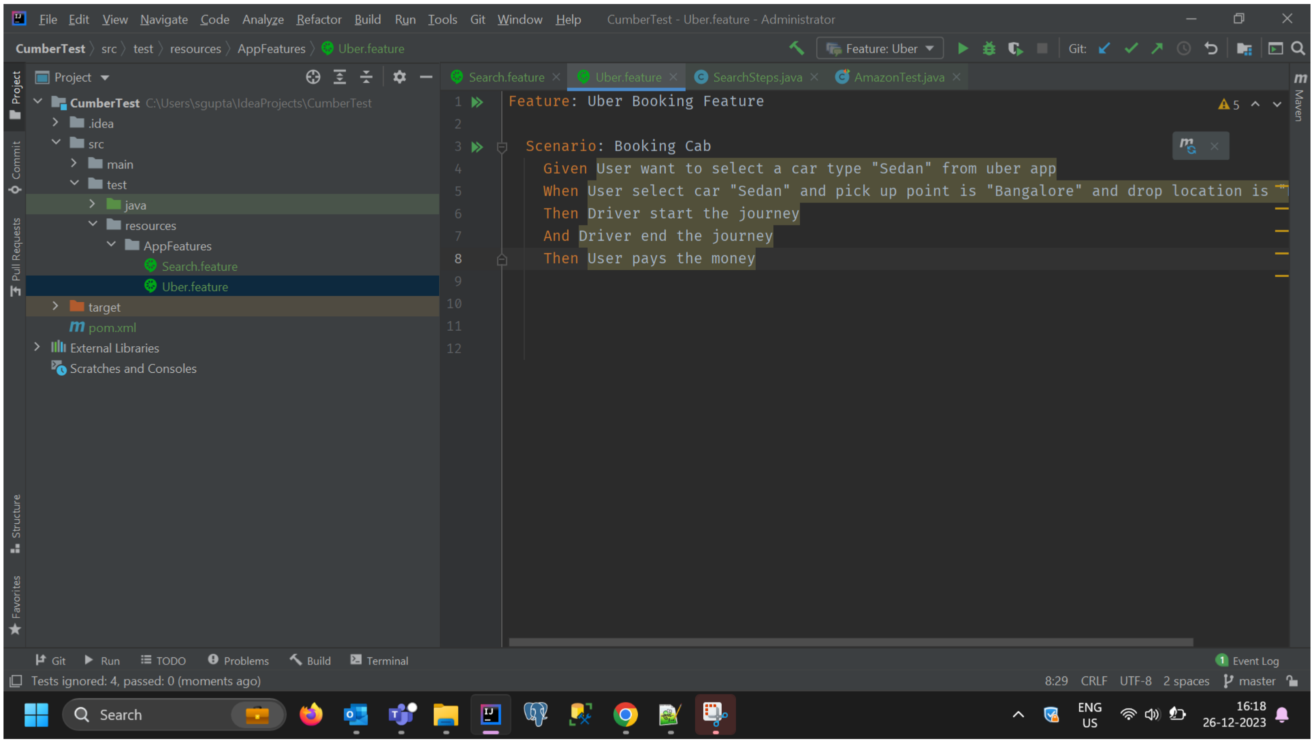
=============

Notes/QA

Notes/QA

Step Defination Warning

If any step not denied then it shows warning - use cucumber for java plugin in intellij



```
Feature: Uber Booking Feature

  Scenario: Booking Cab
    Given User want to select a car type "Sedan" from uber app
    When User select car "Sedan" and pick up point is "Bangalore" and drop location is
    Then Driver start the journey
    And Driver end the journey
    Then User pays the money
```

Tests ignored: 4, passed: 0 (moments ago)

==

Uber.feature

```
Feature: Uber Booking Feature

  Scenario: Booking Cab
    Given User want to select a car type "Sedan" from uber app
    When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune"
    Then Driver start the journey
    And Driver end the journey
    Then User pays 1000 money
```

UberBookingSteps.java

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class UberBookingSteps {

    @Given("User want to select a car type {string} from uber app")
    public void user_want_to_select_a_car_type_from_uber_app(String carType) {
        System.out.println("Step 1: User want to select a car type: " +carType+" from uber app");
    }
    @When("User select car {string} and pick up point is {string} and drop location is {string}")
    public void user_select_car_and_pick_up_point_is_and_drop_location_is(String carType, String
pickLocation, String dropLocation) {
        System.out.println("Step 2: User select car: "+carType+" and pick up point is: "+pickLocation+"
and drop location is "+dropLocation);
    }
    @Then("Driver start the journey")
    public void driver_start_the_journey() {
        System.out.println("Driver start the journey");
    }
    @Then("Driver end the journey")
    public void driver_end_the_journey() {
        System.out.println("Driver end the journey");
    }
    @Then("User pays {int} money")
    public void user_pays_the_money(int amount) {
        System.out.println("Uber pays the money: "+amount);
    }

}
```

===

Notes/QA

Regular Expressions
These are following rules are defined in cucumber for Regular Expressions:
Two types of Reg Expressions in Cucumber:

1. Regular Expression: [0-9]+, (\\d+)
2. Cucumber Expression (introduced in 2017)

Rules:
1. Step def file will be generating cucumber expression only by default
2. But you can use regular expression also in step def file
3. You can mix both regular and cucumber expression in step definition file
4. But you can not mix both expressions in the same step definition method

Quantifiers in Reg Expression:
Define, how many times a character needs to be occurred
https://www.oreilly.com/library/view/...

Regular expressions
[0-9] : [ ] is called capture group

Quantifiers: means how many times a character need to be occur these are +, * , ? , {n}

([0-9]+) : 0 to 9 digits appear (once or more) : I want these 0 to digit will appear one or more than once in a number
([0-9]{4}) : 0000 , 9999, 1212, 3456, 1234, 8888 : {4} genrate 4 digits between 0-9
([0-9]*) : zero or more : generate zero or more
([0-9]?) : zero or once : generate zero or max once

Short hand character :
\d: digits
(\d+) : digits

"([^"]+)" : ^"] dont include the “ from the string regular expression

```
@When("^I search a product with name \"([^\"]+)\" and price (\\d+)$")
```

^ and $ in the start and end should be used to make step legal

Search.feature

```
Feature: Amazon Search

  Scenario: Search a Product
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed
```

SearchSteps.Java

```
package stepDefinations;

import amazonImplementations.Product;
import amazonImplementations.Search;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
import org.junit.Assert;

public class SearchSteps {

   Product product;
   Search search;

   @Given("I have a search field on Amazon Page")
   public void i_have_a_search_field_on_amazon_page() {
      System.out.println("Step 1: I have a search field on Amazon Page");
   }

//Regular expression : Either user regular expression or cucumber expression in single step both
are not allowed, but you can use regular expression and cucumber expression in different steps but
individlally
   @When("^I search a product with name \"([^\"]+)\" and price (\\d+)$")
   public void i_search_a_product_with_name_and_price(String productName, Integer price) {
      System.out.println("Step 2: I search a product with name : " + productName + " and price : " + price);

      //Calling constructor of product class
      product = new Product(productName, price);

   }

   @Then("Product with name {string} should be displayed")
   public void product_with_name_should_be_displayed(String productName) {
      // Write code here that turns the phrase above into concrete actions
      System.out.println("Step 3: I have a search field on Amazon Page: " + productName);

      // Searching the product in the Product list of Passing object of Product class in Search class
      search = new Search();

      //Search and display
      String name_Product = search.displayProduct(product);
      System.out.println("searched product is: " + name_Product);

      //Assertion
      Assert.assertEquals(product.getProductName(), name_Product);
   }
}
```

Output

```
Testing started at 18:14 ...
Step 1: I have a search field on Amazon Page
Step 2: I search a product with name : Apple mac book pro and price : 10000
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro

1 Scenarios (1 passed)
3 Steps (3 passed)
0m1.177s
```

=========================*****************=======================================

Notes/QA

Notes/QA

Tags in the cucumber

Tags can be apply feature level, scenario and scenario outline, we can not apply tags on given when then. We can add 2 or more tags on same scenario also

Execute the single tag execution - **tags = "@Sanity",**   **//→** execute the sanity one only

Uber.feature

```
Feature: Uber Booking Feature

 @Smoke
 Scenario: Booking Cab Sedan
   Given User want to select a car type "Sedan" from uber app
   When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune"
   Then Driver start the journey
   And Driver end the journey
   Then User pays 1000 money

  @Regression
 Scenario: Booking Cab SUV
   Given User want to select a car type "SUV" from uber app
   When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune"
   Then Driver start the journey
   And Driver end the journey
   Then User pays 1300 money

   @Sanity
 Scenario: Booking Cab Mini
   Given User want to select a car type "Mini" from uber app
   When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune"
   Then Driver start the journey
   And Driver end the journey
   Then User pays 3300 money
```

```
package testRunners;



import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Uber.feature"},      //Path of feature file
        glue = {"stepDefinations"},                   //path of Step dedinations
        tags = "@Sanity",                              //→ execute the sanity one only
```

```
    plugin = {"pretty"}                    //Formatting
)
public class UberTest {

}
```

UberBookingSteps.java

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;


public class UberBookingSteps {

  @Given("User want to select a car type {string} from uber app")
  public void user_want_to_select_a_car_type_from_uber_app(String carType) {
    System.out.println("Step 1: User want to select a car type: " +carType+" from uber app");

  }
  @When("User select car {string} and pick up point is {string} and drop location is {string}")
  public void user_select_car_and_pick_up_point_is_and_drop_location_is(String carType, String
pickLocation, String dropLocation) {
    System.out.println("Step 2: User select car: "+carType+" and pick up point is: "+pickLocation+" and
drop location is "+dropLocation);
  }
  @Then("Driver start the journey")
  public void driver_start_the_journey() {
    System.out.println("Driver start the journey");
  }
  @Then("Driver end the journey")
  public void driver_end_the_journey() {
    System.out.println("Driver end the journey");
  }
  @Then("User pays {int} money")
  public void user_pays_the_money(int amount) {
    System.out.println("Uber pays the money: "+amount);
  }

}
```

Output

```
@Sanity
```

```
Scenario: Booking Cab Mini                                                   # src/test/resources/AppFeatures/Uber.feature:20
Step 1: User want to select a car type: Mini from uber app
  Given User want to select a car type "Mini" from uber app                          # stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Mini and pick up point is: Ooty and drop location is Pune
  When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune" # stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                              # stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                                 # stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 3300
  Then User pays 3300 money                                                      # stepDefinations.UberBookingSteps.user_pays_the_money(int)
—-------------------------------------------------------------------------------------------------------------------------------------------------
```

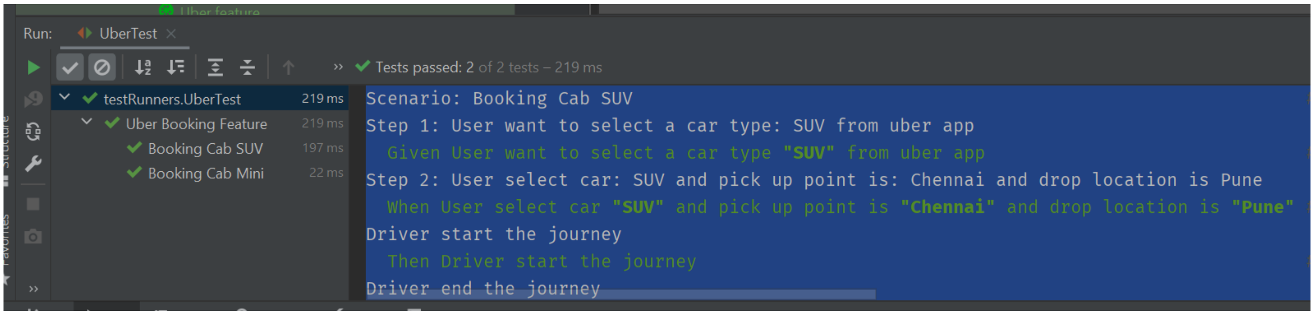
Execute scenarios with different tags - **tags = "@Sanity or @Regression" - execute test cases which have either of these Sanity and Regression tag**

Same class as above

UberTest.java

```
@RunWith(Cucumber.class)
@CucumberOptions(
	features = {"src/test/resources/AppFeatures/Uber.feature"},	//Path of feature file
	glue = {"stepDefinations"},			//path of Step dedinations
	tags = "@Sanity or @Regression",
	plugin = {"pretty"}				//Formatting
)
public class UberTest {

}
```

Output



Uber feature
Run:
UberTest
Tests passed: 2 of 2 tests – 219 ms
✓ testRunners.UberTest 219 ms
✓ Uber Booking Feature 219 ms
✓ Booking Cab SUV 197 ms
✓ Booking Cab Mini 22 ms
Structure
Favorites

```
Scenario: Booking Cab SUV
Step 1: User want to select a car type: SUV from uber app
  Given User want to select a car type "SUV" from uber app
Step 2: User select car: SUV and pick up point is: Chennai and drop location is Pune
  When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune"
Driver start the journey
  Then Driver start the journey
Driver end the journey
```

```
@Regression
Scenario: Booking Cab SUV                                              #
src/test/resources/AppFeatures/Uber.feature:12
Step 1: User want to select a car type: SUV from uber app
  Given User want to select a car type "SUV" from uber app                          #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: SUV and pick up point is: Chennai and drop location is Pune
  When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                          #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                             #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1300
  Then User pays 1300 money                                                #
stepDefinations.UberBookingSteps.user_pays_the_money(int)

@Sanity
Scenario: Booking Cab Mini                                             #
src/test/resources/AppFeatures/Uber.feature:20
Step 1: User want to select a car type: Mini from uber app
  Given User want to select a car type "Mini" from uber app                       #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Mini and pick up point is: Ooty and drop location is Pune
  When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                         #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                            #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 3300
  Then User pays 3300 money                                               #
stepDefinations.UberBookingSteps.user_pays_the_money(int)
```

—---

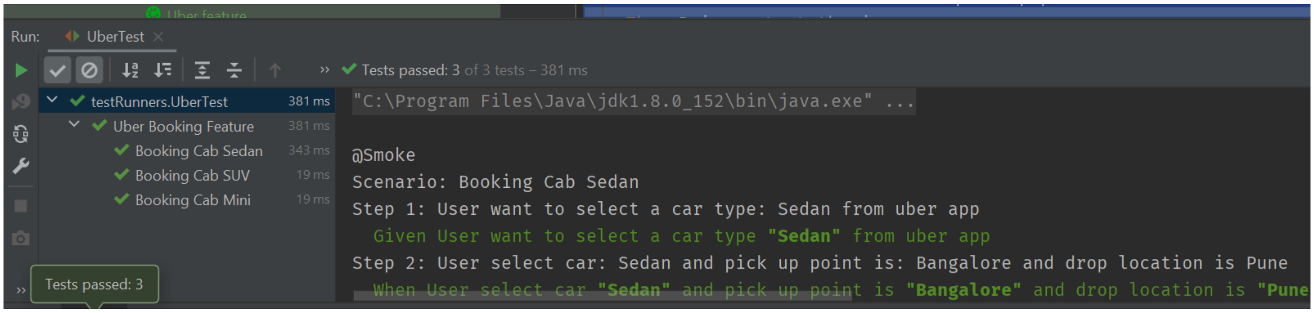
Execute scenarios with different tags - **tags = "@Sanity or @Regression or @Smoke" - execute test cases which have either of these Sanity, Smoke and Regression tag**

Same class as above

UberTest.java

```
@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Uber.feature"},       //Path of feature file
        glue = {"stepDefinations"},                      //path of Step dedinations
        tags = "@Sanity or @Regression or @Smoke",
        plugin = {"pretty"}                              //Formatting
)
public class UberTest {

}
```

Output



```
Tests passed: 3 of 3 tests – 381 ms

testRunners.UberTest        381 ms
  Uber Booking Feature      381 ms
    Booking Cab Sedan       343 ms
    Booking Cab SUV          19 ms
    Booking Cab Mini         19 ms
```

```
"C:\Program Files\Java\jdk1.8.0_152\bin\java.exe" ...

@Smoke
Scenario: Booking Cab Sedan
Step 1: User want to select a car type: Sedan from uber app
  Given User want to select a car type "Sedan" from uber app
Step 2: User select car: Sedan and pick up point is: Bangalore and drop location is Pune
  When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune
```

```
@Smoke
Scenario: Booking Cab Sedan                                                  #
src/test/resources/AppFeatures/Uber.feature:4
Step 1: User want to select a car type: Sedan from uber app
  Given User want to select a car type "Sedan" from uber app                            #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Sedan and pick up point is: Bangalore and drop location is Pune
  When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                              #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                                 #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1000
  Then User pays 1000 money                                                  #
stepDefinations.UberBookingSteps.user_pays_the_money(int)

@Regression
Scenario: Booking Cab SUV                                                    #
src/test/resources/AppFeatures/Uber.feature:12
Step 1: User want to select a car type: SUV from uber app
  Given User want to select a car type "SUV" from uber app                             #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
```

```
Step 2: User select car: SUV and pick up point is: Chennai and drop location is Pune
  When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                    #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                       #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1300
  Then User pays 1300 money                                          #
stepDefinations.UberBookingSteps.user_pays_the_money(int)

@Sanity
Scenario: Booking Cab Mini                                         #
src/test/resources/AppFeatures/Uber.feature:20
Step 1: User want to select a car type: Mini from uber app
  Given User want to select a car type "Mini" from uber app                #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Mini and pick up point is: Ooty and drop location is Pune
  When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                   #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                      #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 3300
  Then User pays 3300 money                                         #
```

—---

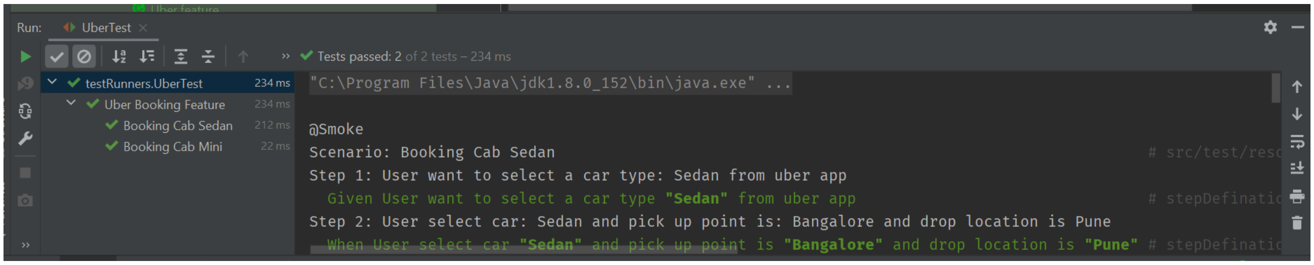
Execute scenarios which not contains the tags - **tags = "not @Regression" - execute test cases which have not Regression tag**

Same class as above

UberTest.java

```
@RunWith(Cucumber.class)
@CucumberOptions(
      features = {"src/test/resources/AppFeatures/Uber.feature"},      //Path of feature file
      glue = {"stepDefinations"},                //path of Step dedinations
      tags = "not  @Regression",
      plugin = {"pretty"}                        //Formatting
)
public class UberTest {

}
```

Output


Run:
UberTest
Tests passed: 2 of 2 tests – 234 ms
testRunners.UberTest 234 ms
Uber Booking Feature 234 ms
Booking Cab Sedan 212 ms
Booking Cab Mini 22 ms

```
"C:\Program Files\Java\jdk1.8.0_152\bin\java.exe" ...

@Smoke
Scenario: Booking Cab Sedan                                                    # src/test/reso
Step 1: User want to select a car type: Sedan from uber app
  Given User want to select a car type "Sedan" from uber app                   # stepDefinatio
Step 2: User select car: Sedan and pick up point is: Bangalore and drop location is Pune
  When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune" # stepDefinatio
```

```
@Smoke
Scenario: Booking Cab Sedan                                         #
src/test/resources/AppFeatures/Uber.feature:4
Step 1: User want to select a car type: Sedan from uber app
  Given User want to select a car type "Sedan" from uber app                   #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Sedan and pick up point is: Bangalore and drop location is Pune
  When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                     #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                        #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1000
  Then User pays 1000 money                                         #
stepDefinations.UberBookingSteps.user_pays_the_money(int)

@Sanity
Scenario: Booking Cab Mini                                          #
src/test/resources/AppFeatures/Uber.feature:20
Step 1: User want to select a car type: Mini from uber app
  Given User want to select a car type "Mini" from uber app                    #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Mini and pick up point is: Ooty and drop location is Pune
  When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                     #
stepDefinations.UberBookingSteps.driver_start_the_journey()
```

```
Driver end the journey
  And Driver end the journey                                   #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 3300
  Then User pays 3300 money                                    #
```

—--

Execute scenarios which contains the multiple tags using “And”- **tags = "@Smoke and @Regression" - execute test cases which contains both Regression as well as Smoke tag**

Same class as above

UberTest.java

```
@RunWith(Cucumber.class)
@CucumberOptions(
	features = {"src/test/resources/AppFeatures/Uber.feature"},	//Path of feature file
	glue = {"stepDefinations"},			//path of Step dedinations
	tags = "@Smoke and @Regression",
	plugin = {"pretty"}				//Formatting
)
public class UberTest {

}
```

Uber.feature

```
Feature: Uber Booking Feature

  @Smoke
  Scenario: Booking Cab Sedan
    Given User want to select a car type "Sedan" from uber app
    When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune"
    Then Driver start the journey
    And Driver end the journey
    Then User pays 1000 money

  @Smoke @Regression
  Scenario: Booking Cab SUV
    Given User want to select a car type "SUV" from uber app
    When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune"
    Then Driver start the journey
    And Driver end the journey
    Then User pays 1300 money

    @Sanity
  Scenario: Booking Cab Mini
    Given User want to select a car type "Mini" from uber app
    When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune"
    Then Driver start the journey
    And Driver end the journey
    Then User pays 3300 money
```

Output

```
@Smoke @Regression
Scenario: Booking Cab SUV                                        #
src/test/resources/AppFeatures/Uber.feature:12
Step 1: User want to select a car type: SUV from uber app
  Given User want to select a car type "SUV" from uber app                  #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: SUV and pick up point is: Chennai and drop location is Pune
  When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                   #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                      #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1300
  Then User pays 1300 money                                         #
stepDefinations.UberBookingSteps.user_pays_the_money(int)
```

===

Execute scenarios which contains All scenarios “All””- **tags = "@All" - execute all the test cases, on using this there is no meaning of all other tags**

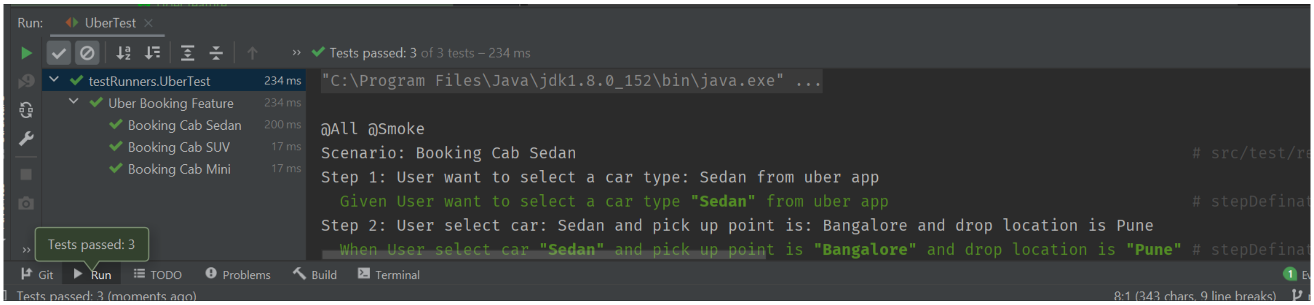
Uber.feature

```
@All                        // inherit all below tags smoke sanity and regression and execute all
Feature: Uber Booking Feature

  @Smoke
  Scenario: Booking Cab Sedan
    Given User want to select a car type "Sedan" from uber app
    When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune"
    Then Driver start the journey
    And Driver end the journey
    Then User pays 1000 money
```

```
 @Smoke @Regression
 Scenario: Booking Cab SUV
   Given User want to select a car type "SUV" from uber app
   When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune"
   Then Driver start the journey
   And Driver end the journey
   Then User pays 1300 money

   @Sanity
 Scenario: Booking Cab Mini
   Given User want to select a car type "Mini" from uber app
   When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune"
   Then Driver start the journey
   And Driver end the journey
   Then User pays 3300 money
```

UberTest.java

```
@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Uber.feature"},        //Path of feature file
        glue = {"stepDefinations"},                        //path of Step dedinations
        tags = "@All",
        plugin = {"pretty"}                          //Formatting
)
public class UberTest {

}
```



```
Tests passed: 3 of 3 tests – 234 ms
testRunners.UberTest      234 ms
  Uber Booking Feature    234 ms
    Booking Cab Sedan     200 ms
    Booking Cab SUV        17 ms
    Booking Cab Mini       17 ms
Tests passed: 3
```

```
"C:\Program Files\Java\jdk1.8.0_152\bin\java.exe" ...

@All @Smoke
Scenario: Booking Cab Sedan                                                    # src/test/re
Step 1: User want to select a car type: Sedan from uber app
  Given User want to select a car type "Sedan" from uber app                   # stepDefinat
Step 2: User select car: Sedan and pick up point is: Bangalore and drop location is Pune
  When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune" # stepDefinat
```

```
Tests passed: 3 (moments ago)                                   8:1 (343 chars, 9 line breaks)
```

```
@All @Smoke
Scenario: Booking Cab Sedan                                        #
src/test/resources/AppFeatures/Uber.feature:5
Step 1: User want to select a car type: Sedan from uber app
  Given User want to select a car type "Sedan" from uber app                       #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Sedan and pick up point is: Bangalore and drop location is Pune
  When User select car "Sedan" and pick up point is "Bangalore" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
```

```
Driver start the journey
  Then Driver start the journey                                                    #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                                       #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1000
  Then User pays 1000 money                                                            #
stepDefinations.UberBookingSteps.user_pays_the_money(int)

@All @Smoke @Regression
Scenario: Booking Cab SUV                                                          #
src/test/resources/AppFeatures/Uber.feature:13
Step 1: User want to select a car type: SUV from uber app
  Given User want to select a car type "SUV" from uber app                         #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: SUV and pick up point is: Chennai and drop location is Pune
  When User select car "SUV" and pick up point is "Chennai" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                                 #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                                    #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 1300
  Then User pays 1300 money                                                         #
stepDefinations.UberBookingSteps.user_pays_the_money(int)

@All @Sanity
Scenario: Booking Cab Mini                                                      #
src/test/resources/AppFeatures/Uber.feature:21
Step 1: User want to select a car type: Mini from uber app
  Given User want to select a car type "Mini" from uber app                     #
stepDefinations.UberBookingSteps.user_want_to_select_a_car_type_from_uber_app(java.lang.String)
Step 2: User select car: Mini and pick up point is: Ooty and drop location is Pune
  When User select car "Mini" and pick up point is "Ooty" and drop location is "Pune" #
stepDefinations.UberBookingSteps.user_select_car_and_pick_up_point_is_and_drop_location_is(java.lang
.String,java.lang.String,java.lang.String)
Driver start the journey
  Then Driver start the journey                                               #
stepDefinations.UberBookingSteps.driver_start_the_journey()
Driver end the journey
  And Driver end the journey                                                  #
stepDefinations.UberBookingSteps.driver_end_the_journey()
Uber pays the money: 3300
```

```
  Then User pays 3300 money                                        #
stepDefinations.UberBookingSteps.user_pays_the_money(int)
====================================================
```

Tags using maven and override

We can override the tags from the maven and run as per our need - Command prompt

```
C:\Users\sgupta\IdeaProjects\CumberTest
C:\Users\sgupta\IdeaProjects\CumberTest>mvn test -Dcucumber.filter.tags="@Smoke or @Sanity"
```

Execute only either of smoke and sanity test cases

```
C:\Users\sgupta\IdeaProjects\CumberTest
C:\Users\sgupta\IdeaProjects\CumberTest>mvn test -Dcucumber.filter.tags="@Smoke or @Sanity"
```

===

Notes/QA

Notes/QA

Notes/QA

Background:
As per cucumber official documentation:
Occasionally you’ll find yourself repeating the same Given steps in all of the scenarios in a Feature.

Since it is repeated in every scenario, this is an indication that those steps are not essential to describe the scenarios; they are incidental details. You can literally move such Given steps to the background, by grouping them under a Background section.

A Background allows you to add some context to the scenarios that follow it. It can contain one or more Given steps, which are run before each scenario, but after any Before hooks.

A Background is placed before the first Scenario/Example, at the same level of indentation.

My understanding : Background is like to consoldiate prerequisite and repeated steps of different scenarios so that user can avoid to write multiple steps definition repeatadely and help user to make feature file more readable.

Order.feature

```
Feature: Amazon Order
  In order to check all the pages od order like previous, cancelled and open

  Background:                    //here this steps or used in each scenario but we can define them once
    Given a registered user exists
    Given user is navigate to login page
    When user enters user name
    And user enters password
    And user clicks on password
    Then  user navigate to order page

  Scenario: Check Previous order details
    When user is on Previous Orders page Link
    Then user click on Previous Orders details

  Scenario: Check Open order details
    When user is on Open Orders page Link
    Then user click on Open Orders details

  Scenario: Check Cancelled order details
    When user is on Cancelled Orders page Link
    Then user click on Cancelled Orders details
```

OrderTest.Java

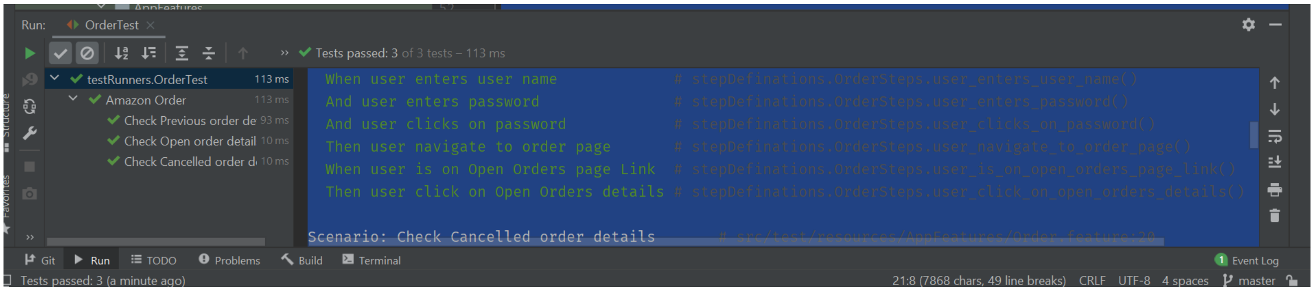
```
package testRunners;

import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
      features = {"src/test/resources/AppFeatures/Order.feature"},      //Path of feature file
      glue = {"stepDefinations"},                    //path of Step dedinations
      plugin = {"pretty"}                          //Formatting
)
public class OrderTest {

}
```

OrderSteps.java

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class OrderSteps {
   @Given("a registered user exists")
   public void a_registered_user_exists() {

   }
   @Given("user is navigate to login page")
   public void user_is_navigate_to_login_page() {

   }
   @When("user enters user name")
   public void user_enters_user_name() {

   }
   @When("user enters password")
   public void user_enters_password() {

   }
   @When("user clicks on password")
   public void user_clicks_on_password() {

   }
   @Then("user navigate to order page")
   public void user_navigate_to_order_page() {

   }
   @When("user is on Previous Orders page Link")
```

```
    public void user_is_on_previous_orders_page_link() {

    }
    @Then("user click on Previous Orders details")
    public void user_click_on_previous_orders_details() {

    }

    @When("user is on Open Orders page Link")
    public void user_is_on_open_orders_page_link() {

    }
    @Then("user click on Open Orders details")
    public void user_click_on_open_orders_details() {

    }

    @When("user is on Cancelled Orders page Link")
    public void user_is_on_cancelled_orders_page_link() {

    }
    @Then("user click on Cancelled Orders details")
    public void user_click_on_cancelled_orders_details() {

    }

}
```

Output



Run: OrderTest
Tests passed: 3 of 3 tests – 113 ms
testRunners.OrderTest 113 ms
Amazon Order 113 ms
Check Previous order de 93 ms
Check Open order detail 10 ms
Check Cancelled order d 10 ms
Git
Run
TODO
Problems
Build
Terminal
Event Log
Tests passed: 3 (a minute ago)
21:8 (7868 chars, 49 line breaks) CRLF UTF-8 4 spaces master

```
  When user enters user name               # stepDefinations.OrderSteps.user_enters_user_name()
  And user enters password                 # stepDefinations.OrderSteps.user_enters_password()
  And user clicks on password              # stepDefinations.OrderSteps.user_clicks_on_password()
  Then user navigate to order page         # stepDefinations.OrderSteps.user_navigate_to_order_page()
  When user is on Open Orders page Link    # stepDefinations.OrderSteps.user_is_on_open_orders_page_link()
  Then user click on Open Orders details   # stepDefinations.OrderSteps.user_click_on_open_orders_details()

Scenario: Check Cancelled order details        # src/test/resources/AppFeatures/Order.feature:20
```

```
Scenario: Check Previous order details       # src/test/resources/AppFeatures/Order.feature:12
  Given a registered user exists            # stepDefinations.OrderSteps.a_registered_user_exists()
  Given user is navigate to login page       # stepDefinations.OrderSteps.user_is_navigate_to_login_page()
  When user enters user name                 # stepDefinations.OrderSteps.user_enters_user_name()
  And user enters password                  # stepDefinations.OrderSteps.user_enters_password()
  And user clicks on password                # stepDefinations.OrderSteps.user_clicks_on_password()
  Then user navigate to order page           # stepDefinations.OrderSteps.user_navigate_to_order_page()
  When user is on Previous Orders page Link  #
stepDefinations.OrderSteps.user_is_on_previous_orders_page_link()
```

```
  Then user click on Previous Orders details #
stepDefinations.OrderSteps.user_click_on_previous_orders_details()

Scenario: Check Open order details       # src/test/resources/AppFeatures/Order.feature:16
  Given a registered user exists         # stepDefinations.OrderSteps.a_registered_user_exists()
  Given user is navigate to login page   # stepDefinations.OrderSteps.user_is_navigate_to_login_page()
  When user enters user name             # stepDefinations.OrderSteps.user_enters_user_name()
  And user enters password               # stepDefinations.OrderSteps.user_enters_password()
  And user clicks on password            # stepDefinations.OrderSteps.user_clicks_on_password()
  Then user navigate to order page       # stepDefinations.OrderSteps.user_navigate_to_order_page()
  When user is on Open Orders page Link  #
stepDefinations.OrderSteps.user_is_on_open_orders_page_link()
  Then user click on Open Orders details #
stepDefinations.OrderSteps.user_click_on_open_orders_details()

Scenario: Check Cancelled order details       # src/test/resources/AppFeatures/Order.feature:20
  Given a registered user exists              # stepDefinations.OrderSteps.a_registered_user_exists()
  Given user is navigate to login page        # stepDefinations.OrderSteps.user_is_navigate_to_login_page()
  When user enters user name                  # stepDefinations.OrderSteps.user_enters_user_name()
  And user enters password                    # stepDefinations.OrderSteps.user_enters_password()
  And user clicks on password                 # stepDefinations.OrderSteps.user_clicks_on_password()
  Then user navigate to order page            # stepDefinations.OrderSteps.user_navigate_to_order_page()
  When user is on Cancelled Orders page Link  #
stepDefinations.OrderSteps.user_is_on_cancelled_orders_page_link()
  Then user click on Cancelled Orders details #
stepDefinations.OrderSteps.user_click_on_cancelled_orders_details()
================================================================================
```

Notes/QA

Notes/QA

Notes/QA

Hooks

- we can not write hooks in feature files, but “background” should be part of feature files
- Hooks can be write in the step definition class or separate configuration class
- Hooks are annotated with @ symbol

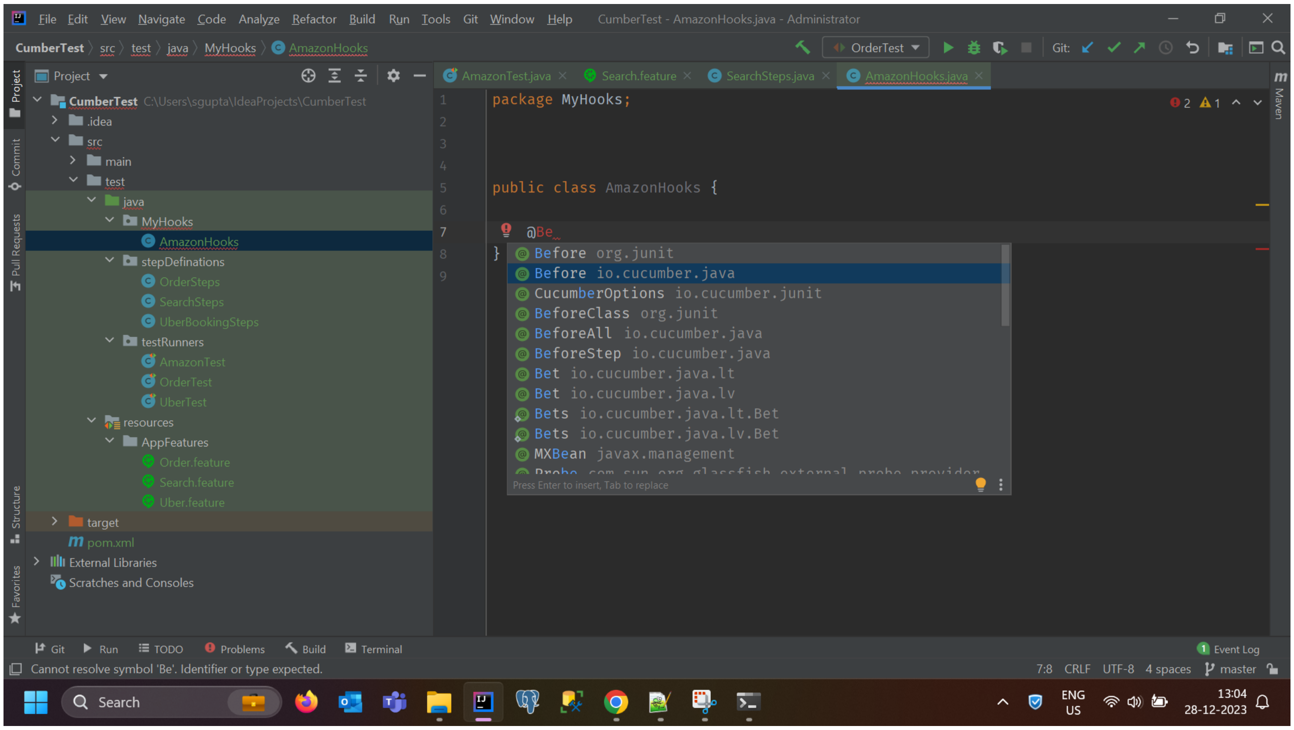
Setup and tear down
@Before - will executed before each scenario
@After - will executed aftereach scenario

@BeforeStep - will executed before each step of the scenario
@AfterStep - will executed aftereach step of the scenario

We can have multiple @before and @after

@Before
m1(order=1)
@Before
m1(order=2)
@After
m1(order=2)
@After
m1(order=1)

Always import the hooks for cucumber not from Testng or Junit



```
package MyHooks;


public class AmazonHooks {

    @Be
}
```

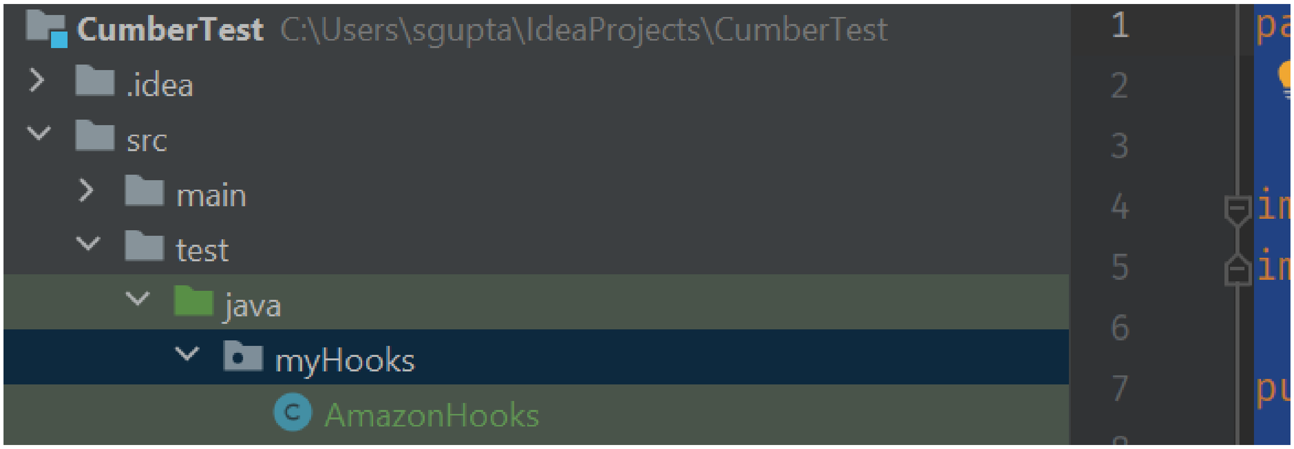
```
Before org.junit
Before io.cucumber.java
CucumberOptions io.cucumber.junit
BeforeClass org.junit
BeforeAll io.cucumber.java
BeforeStep io.cucumber.java
Bet io.cucumber.java.lt
Bet io.cucumber.java.lv
Bets io.cucumber.java.lt.Bet
Bets io.cucumber.java.lv.Bet
MXBean javax.management
Probe com.sun.org.glassfish.external.probe.provider
Press Enter to insert, Tab to replace
```

Cannot resolve symbol 'Be'. Identifier or type expected.

==

Simple Single hooks @Before and @After

Create a package in the src/test/java>>myHooks and then create a class for thee same as AmazonHooks



AmazonHooks .java

package myHooks;

import io.cucumber.java.After;
import io.cucumber.java.Before;

public class AmazonHooks {

    @Before
    public void setup_Browser()
    {

```
        System.out.println("Launch Chrome browser");
    }

    @After
    public void close_Browser()
    {
        System.out.println("Close Chrome browser");
    }

}
```

AmazonTest .java   //Runner class

Now glue it on the runner class

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Search.feature"},     //Path of feature file
        glue = {"stepDefinations","myHooks"},           //path of Stepdefinations and hooks
        plugin = {"pretty"}                           //Formatting
)
public class AmazonTest {

}
```

Add one extra scenario to understand more

Search.feature

```
Feature: Amazon Search

  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

OrderSteps.java

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class OrderSteps {
    @Given("a registered user exists")
    public void a_registered_user_exists() {

    }
    @Given("user is navigate to login page")
    public void user_is_navigate_to_login_page() {

    }
    @When("user enters user name")
    public void user_enters_user_name() {

    }
    @When("user enters password")
    public void user_enters_password() {

    }
    @When("user clicks on password")
    public void user_clicks_on_password() {

    }
    @Then("user navigate to order page")
    public void user_navigate_to_order_page() {

    }
    @When("user is on Previous Orders page Link")
    public void user_is_on_previous_orders_page_link() {

    }
    @Then("user click on Previous Orders details")
    public void user_click_on_previous_orders_details() {

    }

    @When("user is on Open Orders page Link")
    public void user_is_on_open_orders_page_link() {

    }
    @Then("user click on Open Orders details")
    public void user_click_on_open_orders_details() {

    }

    @When("user is on Cancelled Orders page Link")
    public void user_is_on_cancelled_orders_page_link() {

    }
    @Then("user click on Cancelled Orders details")
    public void user_click_on_cancelled_orders_details() {

    }

}
```

Output : Before and after execute before and after each scenario of feature file

```
Scenario: Search a Product mac book                          #
src/test/resources/AppFeatures/Search.feature:3
Launch Chrome browser
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                       #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser

Scenario: Search a Product iPhone                    # src/test/resources/AppFeatures/Search.feature:8
Launch Chrome browser
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page              #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser
=================================================================
```

Multiple @Before and @After with order

AmazonHooks .java

```
package myHooks;


import io.cucumber.java.After;
import io.cucumber.java.Before;

public class AmazonHooks {

   @Before(order = 1)
   public void setup_Browser()
   {
      System.out.println("Launch Chrome browser");
   }

   @Before(order = 2)
   public void setup_Application()
   {
      System.out.println("Launch Application Url and Amazon app");
   }

   @After(order = 1)
   public void close_Application()
   {
      System.out.println("Close Application Url and Amazon app");
   }

   @After(order = 2)
   public void close_Browser()
   {
      System.out.println("Close Chrome browser");
   }

}
```

AmazonTest .java   //Runner class

Now glue it on the runner class

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Search.feature"},       //Path of feature file
        glue = {"stepDefinations","myHooks"},           //path of Stepdefinations and hooks
        plugin = {"pretty"}                             //Formatting
)
public class AmazonTest {

}
```

Add one extra scenario to understand more

Search.feature

```
Feature: Amazon Search

  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

OrderSteps.java

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class OrderSteps {
    @Given("a registered user exists")
    public void a_registered_user_exists() {

    }
    @Given("user is navigate to login page")
    public void user_is_navigate_to_login_page() {

    }
    @When("user enters user name")
    public void user_enters_user_name() {
    }
    @When("user enters password")
    public void user_enters_password() {

    }
    @When("user clicks on password")
    public void user_clicks_on_password() {

    }
    @Then("user navigate to order page")
    public void user_navigate_to_order_page() {

    }
    @When("user is on Previous Orders page Link")
    public void user_is_on_previous_orders_page_link() {

    }
    @Then("user click on Previous Orders details")
    public void user_click_on_previous_orders_details() {

    }

    @When("user is on Open Orders page Link")
    public void user_is_on_open_orders_page_link() {

    }
    @Then("user click on Open Orders details")
    public void user_click_on_open_orders_details() {

    }

    @When("user is on Cancelled Orders page Link")
    public void user_is_on_cancelled_orders_page_link() {

    }
    @Then("user click on Cancelled Orders details")
    public void user_click_on_cancelled_orders_details() {

    }

}
```

Output: now there are 2 before and after condition as per order property

```
Scenario: Search a Product mac book                              #
src/test/resources/AppFeatures/Search.feature:3
Launch Chrome browser
Launch Application Url and Amazon app
```

```
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                              #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser
Close Application Url and Amazon app

Scenario: Search a Product iPhone                        # src/test/resources/AppFeatures/Search.feature:8
Launch Chrome browser
Launch Application Url and Amazon app
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser
Close Application Url and Amazon app
```

==

Scenario class: we can use the Scenario class methods for printing status of scenario, name and etc.

AmazonHooks .java

```
package myHooks;


import io.cucumber.java.After;
import io.cucumber.java.Before;

public class AmazonHooks {


   @Before(order = 1)
   public void setup_Browser(Scenario scenario)
   {
      System.out.println("Launch Chrome browser");
```

```
    System.out.println(scenario.getName());

  }

  @Before(order = 2)
  public void setup_Application(Scenario scenario)
  {
    System.out.println("Launch Application Url and Amazon app");
    System.out.println(scenario.getStatus());
  }


  @After(order = 1)
  public void close_Application()
  {
    System.out.println("Close Application Url and Amazon app");
  }

  @After(order = 2)
  public void close_Browser()
  {
    System.out.println("Close Chrome browser");
  }

}
```

AmazonTest .java   //Runner class

Now glue it on the runner class

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
    features = {"src/test/resources/AppFeatures/Search.feature"},     //Path of feature file
    glue = {"stepDefinations","myHooks"},         //path of Stepdefinations and hooks
    plugin = {"pretty"}                         //Formatting
)
public class AmazonTest {

}
```

Add one extra scenario to understand more

Search.feature

```
Feature: Amazon Search

  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

Output: now there are 2 before and after condition as per order property and scenario class methods

```
Scenario: Search a Product mac book                                #
src/test/resources/AppFeatures/Search.feature:3
Launch Chrome browser                //Before order 1
Search a Product mac book //Scenario name using getname()
Launch Application Url and Amazon app //Before order 2
PASSED                                                             //status of Scenario
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                       #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser                                    //After order =1
Close Application Url and Amazon app     //After order =2

Scenario: Search a Product iPhone                      # src/test/resources/AppFeatures/Search.feature:8
Launch Chrome browser     //Before order 1
Search a Product iPhone //Scenario name using getname()
Launch Application Url and Amazon app  //Before order 2
PASSED                              //status of Scenario
Step 1: I have a search field on Amazon Page
```

```
  Given I have a search field on Amazon Page                #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser                                               //After order =1
Close Application Url and Amazon app       //After order =2
=============================================================
```

Hooks @Before and @After with @BeforeStep and @AfterStep

AmazonHooks .java

```
package myHooks;


import io.cucumber.java.*;


public class AmazonHooks {

   @Before
   public void setup_Browser()
   {
      System.out.println("Launch Chrome browser");

   }

   @BeforeStep
   public void takesSnap()
   {
      System.out.println("Takes Screenshot on each step");
   }

   @AfterStep
   public void refresh()
   {
      System.out.println("Refresh on each step");
   }

   @After
   public void close_Browser()
   {
      System.out.println("Close Chrome browser");
   }

}
```

AmazonTest .java   //Runner class

Now glue it on the runner class

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Search.feature"},       //Path of feature file
        glue = {"stepDefinations","myHooks"},           //path of Stepdefinations and hooks
        plugin = {"pretty"}                             //Formatting
)
public class AmazonTest {

}
```

Add one extra scenario to understand more

Search.feature

```
Feature: Amazon Search

  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

OrderSteps.java

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class OrderSteps {
   @Given("a registered user exists")
   public void a_registered_user_exists() {

   }
   @Given("user is navigate to login page")
   public void user_is_navigate_to_login_page() {

   }
   @When("user enters user name")
   public void user_enters_user_name() {

   }
   @When("user enters password")
   public void user_enters_password() {

   }
   @When("user clicks on password")
   public void user_clicks_on_password() {

   }
   @Then("user navigate to order page")
   public void user_navigate_to_order_page() {

   }
   @When("user is on Previous Orders page Link")
   public void user_is_on_previous_orders_page_link() {

   }
   @Then("user click on Previous Orders details")
   public void user_click_on_previous_orders_details() {

   }

   @When("user is on Open Orders page Link")
   public void user_is_on_open_orders_page_link() {

   }
   @Then("user click on Open Orders details")
   public void user_click_on_open_orders_details() {

   }

   @When("user is on Cancelled Orders page Link")
   public void user_is_on_cancelled_orders_page_link() {
   }
   @Then("user click on Cancelled Orders details")
   public void user_click_on_cancelled_orders_details() {

   }

}
```

Output

```
Scenario: Search a Product mac book                              #
src/test/resources/AppFeatures/Search.feature:3
Launch Chrome browser               //Before
Takes Screenshot on each step       //BeforeStep
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                      #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step                //AfterStep
Takes Screenshot on each step       //BeforeStep
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step                //AfterStep
Takes Screenshot on each step              //BeforeStep
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step                //AfterStep
Close Chrome browser                //After

Scenario: Search a Product iPhone                    # src/test/resources/AppFeatures/Search.feature:8
Launch Chrome browser                      //Before
Takes Screenshot on each step                     //BeforeStep
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page              #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step                       //AfterStep
Takes Screenshot on each step                     //BeforeStep
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step           //AfterStep
Takes Screenshot on each step              //beforeStep
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
```

```
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step            //AfterStep
Close Chrome browser                    //After
=========================================================================
```

Hooks with Tags: we can use the tags with hooks like this
@before("@smoke")

Search.feature

```
Feature: Amazon Search

  @Smoke
  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  @Regression
  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

AmazonTest.java   **//Runner class**

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
      features = {"src/test/resources/AppFeatures/Search.feature"},       //Path of feature file
      glue = {"stepDefinations","myHooks"},            //path of Stepdefinations and hooks
      plugin = {"pretty"}  ,                           //Formatting
      tags="@Smoke or @Regression"
)
public class AmazonTest {

}
```

AmazonHooks.java

```
package myHooks;


import io.cucumber.java.*;
import java.util.Scanner;

public class AmazonHooks {

   @Before("@Smoke")
   public void setup_Browser()
   {
      System.out.println("Launch Chrome browser");

   }

   @BeforeStep
   public void takesSnap()
   {
      System.out.println("Takes Screenshot on each step");
   }

   @AfterStep
   public void refresh()
   {
      System.out.println("Refresh on each step");
   }

   @After("@Smoke")
   public void close_Browser()
   {
      System.out.println("Close Chrome browser");
   }

}
```

Output: only smoke marked tags executed as we use only @Before("@Smoke") and @After("@Smoke") for smoke tags so that regression one is not in the scope, so for regression mark scenario not executed and take @Before("@Smoke") and @After("@Smoke")

```
@Smoke
Scenario: Search a Product mac book                              #
src/test/resources/AppFeatures/Search.feature:4
Launch Chrome browser
Takes Screenshot on each step
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                     #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step
Takes Screenshot on each step
Step 2: I search a product with name : Apple mac book pro and price : 10000
```

```
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step
Close Chrome browser

@Regression
Scenario: Search a Product iPhone                          # src/test/resources/AppFeatures/Search.feature:10
Takes Screenshot on each step
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page               #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step
Takes Screenshot on each step
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step
```

===

Hooks with Tags: we can use the tags with hooks like this
@before(“@smoke”) and @before(“@Regression”)

Search.feature

```
Feature: Amazon Search

  @Smoke
  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  @Regression
  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
```

```
   Then Product with name "iPhone" should be displayed
```

AmazonTest.java   //Runner class

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
      features = {"src/test/resources/AppFeatures/Search.feature"},      //Path of feature file
      glue = {"stepDefinations","myHooks"},            //path of Stepdefinations and hooks
      plugin = {"pretty"}  ,                            //Formatting
      tags="@Smoke or @Regression"
)
public class AmazonTest {

}
```

AmazonHooks.java

```
package myHooks;


import io.cucumber.java.*;

import java.util.Scanner;

public class AmazonHooks {

   @Before("@Smoke")
   public void setup_Browser()
   {
      System.out.println("Launch Chrome browser");

   }

   @BeforeStep("@Regression")
   public void takesSnap()
   {
      System.out.println("Takes Screenshot on each step");
   }

   @AfterStep("@Regression")
   public void refresh()
   {
      System.out.println("Refresh on each step");
```

```
    }

    @After("@Smoke")
    public void close_Browser()
    {
        System.out.println("Close Chrome browser");
    }

}
```

Output: here in this case first scenario executed with Smoke tag i.e it have Before and After but no BeforeStep and AfterStep as it is tagged as Regression

While Second scenario executed with Regression i.e BeforeStep and AfterStep as it is tagged as Regression but no Before and After as it is marked Smoke tag

```
@Smoke
Scenario: Search a Product mac book                              #
src/test/resources/AppFeatures/Search.feature:4
Launch Chrome browser                                   //Before
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                       #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser                              //After

@Regression
Scenario: Search a Product iPhone                       # src/test/resources/AppFeatures/Search.feature:10
Takes Screenshot on each step                     //BeforeStep
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page               #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step                                                        //AfterStep
Takes Screenshot on each step                           //BeforeStep
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step                                                        //AfterStep
Takes Screenshot on each step                                 //BeforeStep
Step 3: I have a search field on Amazon Page: iPhone
```

```
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
```

Refresh on each step **//AfterStep**

===

Notes/QA

Notes/QA

Notes/QA

Notes/QA

Notes/QA

Generate JSON, Junit Reports

Search.feature

```
Feature: Amazon Search

  @Smoke
  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  @Regression
  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

AmazonTest.java   //Runner class

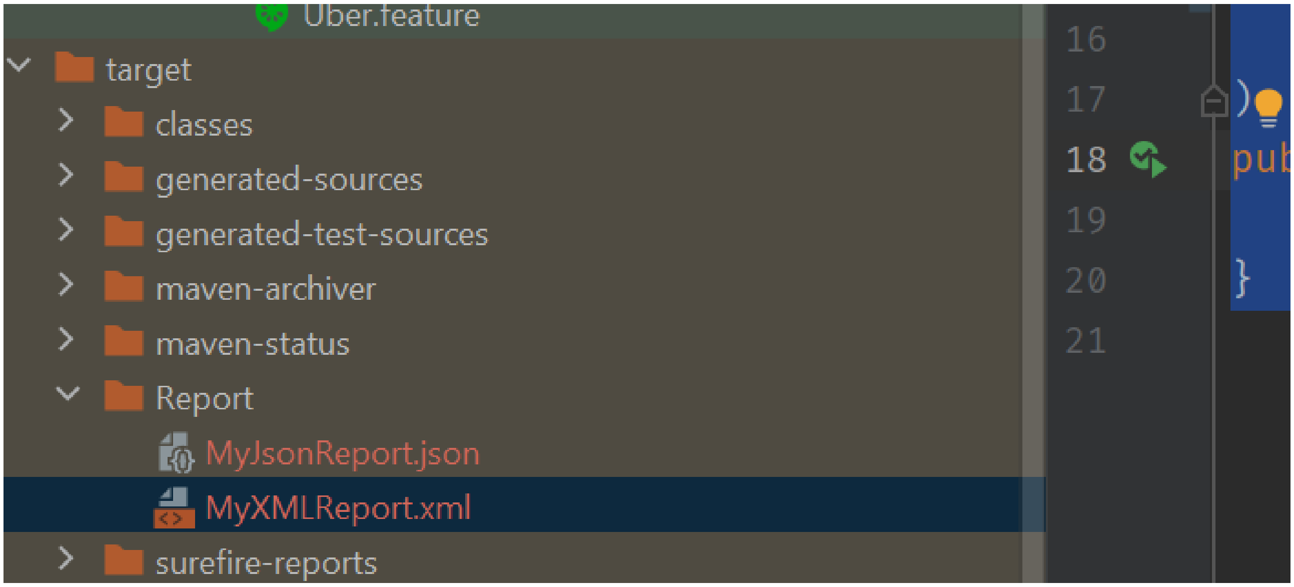
```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Search.feature"},       //Path of feature file
        glue = {"stepDefinations","myHooks"},            //path of Stepdefinations and hooks
        plugin = {"pretty",            //Formatting
          "json:target/Report/MyJsonReport.json",   //generate json report in target>>Report folder
          "junit:target/Report/MyXMLReport.xml"}  , //generate Junit report in target>>Report folder
        tags="@Smoke or @Regression"
)
public class AmazonTest {

}
```

AmazonHooks.java

```
package myHooks;


import io.cucumber.java.*;

import java.util.Scanner;
```

```
public class AmazonHooks {

    @Before("@Smoke")
    public void setup_Browser()
    {
        System.out.println("Launch Chrome browser");

    }

    @BeforeStep("@Regression")
    public void takesSnap()
    {
        System.out.println("Takes Screenshot on each step");
    }

    @AfterStep("@Regression")
    public void refresh()
    {
        System.out.println("Refresh on each step");
    }

    @After("@Smoke")
    public void close_Browser()
    {
        System.out.println("Close Chrome browser");
    }

}
```

Output



```
@Smoke
Scenario: Search a Product mac book                        #
src/test/resources/AppFeatures/Search.feature:4
Launch Chrome browser
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                        #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser

@Regression
Scenario: Search a Product iPhone                    # src/test/resources/AppFeatures/Search.feature:10
Takes Screenshot on each step
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page              #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step
Takes Screenshot on each step
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step
┌──────────────────────────────────────────────────────────────────────────────
──────────────────┐
│ Share your Cucumber Report with your team at https://reports.cucumber.io          │
│ Activate publishing with one of the following:                                  │
│                                                                                  │
│ src/test/resources/cucumber.properties:          cucumber.publish.enabled=true    │
│ src/test/resources/junit-platform.properties:    cucumber.publish.enabled=true    │
│ Environment variable:                            CUCUMBER_PUBLISH_ENABLED=true     │
│ JUnit:                                           @CucumberOptions(publish = true) │
│                                                                                  │
│ More information at https://cucumber.io/docs/cucumber/environment-variables/      │
│                                                                                  │
│ Disable this message with one of the following:                                 │
│                                                             │
│ src/test/resources/cucumber.properties:          cucumber.publish.quiet=true       │
│ src/test/resources/junit-platform.properties:    cucumber.publish.quiet=true       │
└──────────────────────────────────────────────────────────────────────────────
──────────────────────────┘
```

===

Generate Cucumber Web Report (new reporting in cucumber 6.9.0) What is publish=true flag

On running test cases you will see this options

```
src/test/resources/cucumber.properties:          cucumber.publish.enabled=true   │          //Create a
properties file
│ src/test/resources/junit-platform.properties:    cucumber.publish.enabled=true   │
│ Environment variable:                            CUCUMBER_PUBLISH_ENABLED=true   │
│ JUnit:                          @CucumberOptions(publish = true) -         //Genrerate web report
```

Using (publish = true)

Search.feature

```
Feature: Amazon Search

  @Smoke
  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed

  @Regression
  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

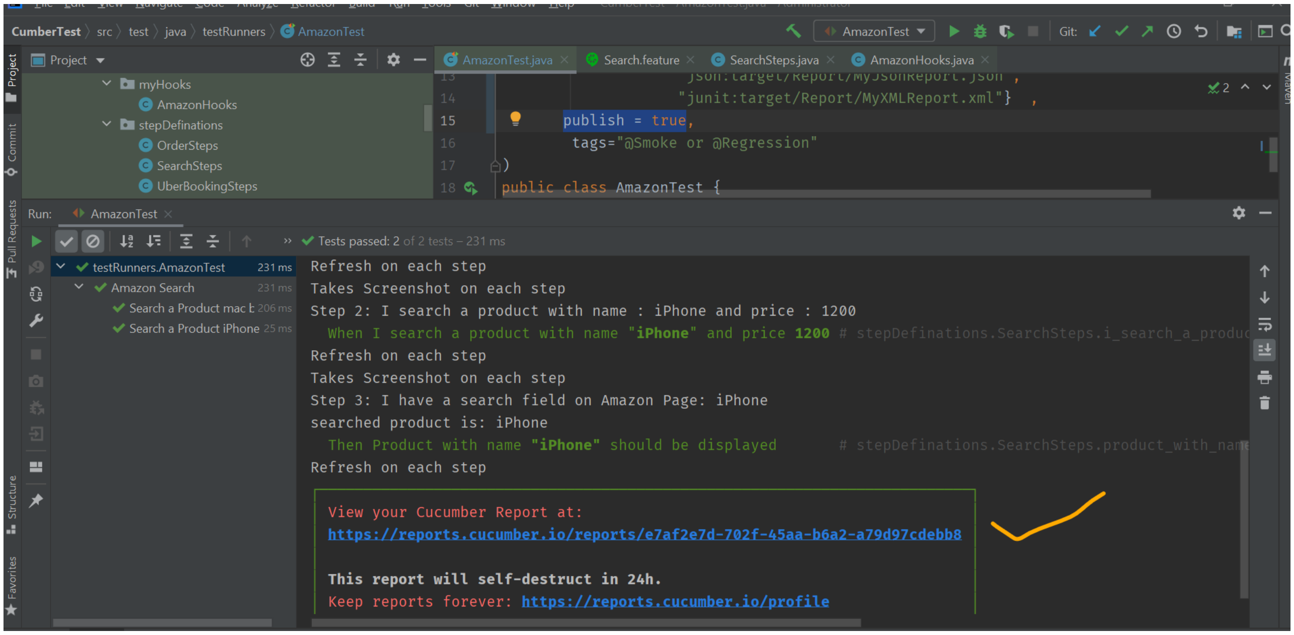
AmazonTest.java **//Runner class**

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
    features = {"src/test/resources/AppFeatures/Search.feature"},      //Path of feature file
    glue = {"stepDefinations","myHooks"},          //path of Stepdefinations and hooks
    plugin = {"pretty",          //Formatting
      "json:target/Report/MyJsonReport.json",   //generate json report in target>>Report folder
      "junit:target/Report/MyXMLReport.xml"}  , //generate Junit report in target>>Report folder
    tags="@Smoke or @Regression",
publish = true                              //generate web report

)
public class AmazonTest {

}
```

AmazonHooks.java

```
package myHooks;


import io.cucumber.java.*;

import java.util.Scanner;

public class AmazonHooks {

  @Before("@Smoke")
  public void setup_Browser()
  {
    System.out.println("Launch Chrome browser");

  }

  @BeforeStep("@Regression")
  public void takesSnap()
  {
    System.out.println("Takes Screenshot on each step");
  }

  @AfterStep("@Regression")
  public void refresh()
  {
    System.out.println("Refresh on each step");
  }

  @After("@Smoke")
  public void close_Browser()
  {
    System.out.println("Close Chrome browser");
    }

}
```

Output



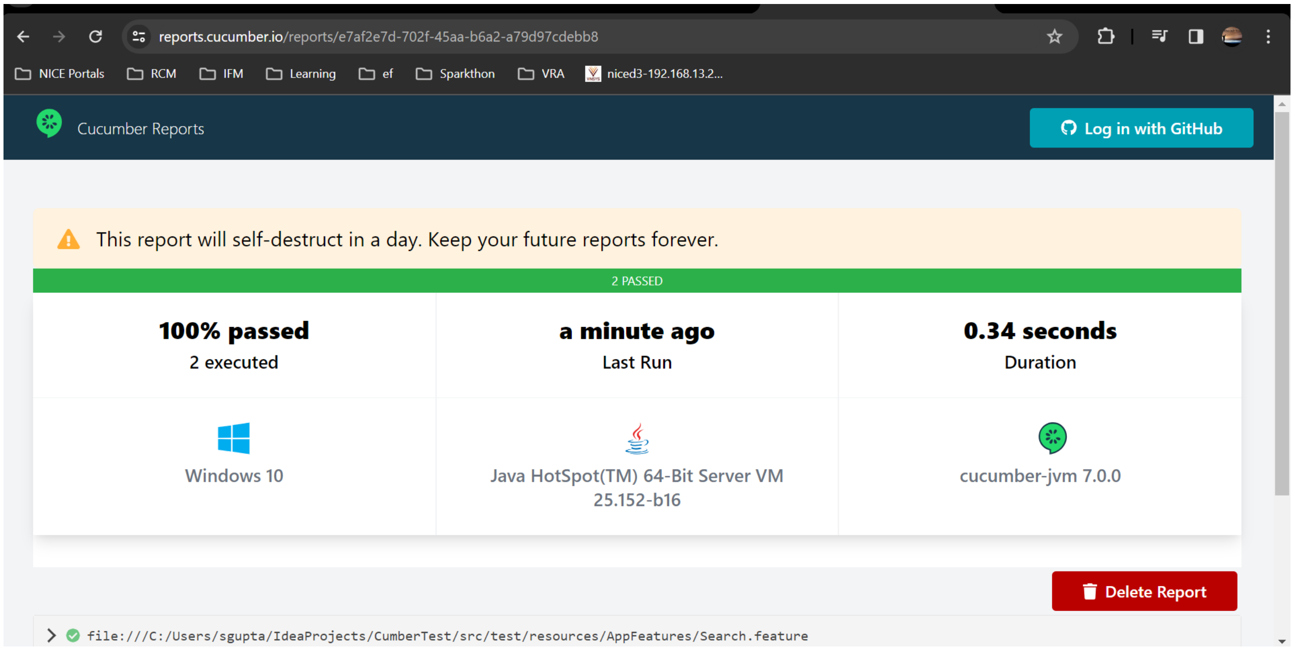
```
                          json:target/Report/MyJsonReport.json" ,
                        "junit:target/Report/MyXMLReport.xml"}  ,
        publish = true,
         tags="@Smoke or @Regression"
)
public class AmazonTest {
```

✓ Tests passed: 2 of 2 tests – 231 ms

```
Refresh on each step
Takes Screenshot on each step
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 # stepDefinations.SearchSteps.i_search_a_produc
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed      # stepDefinations.SearchSteps.product_with_name
Refresh on each step

 View your Cucumber Report at:
 https://reports.cucumber.io/reports/e7af2e7d-702f-45aa-b6a2-a79d97cdebb8

 This report will self-destruct in 24h.
 Keep reports forever: https://reports.cucumber.io/profile
```

```
@Smoke
Scenario: Search a Product mac book                                  #
src/test/resources/AppFeatures/Search.feature:4
Launch Chrome browser
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                         #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser

@Regression
Scenario: Search a Product iPhone                        # src/test/resources/AppFeatures/Search.feature:10
Takes Screenshot on each step
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page               #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step
Takes Screenshot on each step
```

```
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed      #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step
┌──────────────────────────────────────────────────────────────────────────
──────────┐
│ View your Cucumber Report at:                                 │
│ https://reports.cucumber.io/reports/e7af2e7d-702f-45aa-b6a2-a79d97cdebb8 │
│                                                   │
│ This report will self-destruct in 24h.                        │
│ Keep reports forever: https://reports.cucumber.io/profile             │

Process finished with exit code 0
└──────────────────────────────────────────────────────────────────────────
──────────┘
```



reports.cucumber.io/reports/e7af2e7d-702f-45aa-b6a2-a79d97cdebb8

This report will self-destruct in a day. Keep your future reports forever.

2 PASSED

100% passed	a minute ago	0.34 seconds
2 executed	Last Run	Duration
Windows 10	Java HotSpot(TM) 64-Bit Server VM 25.152-b16	cucumber-jvm 7.0.0

file:///C:/Users/sgupta/IdeaProjects/CumberTest/src/test/resources/AppFeatures/Search.feature

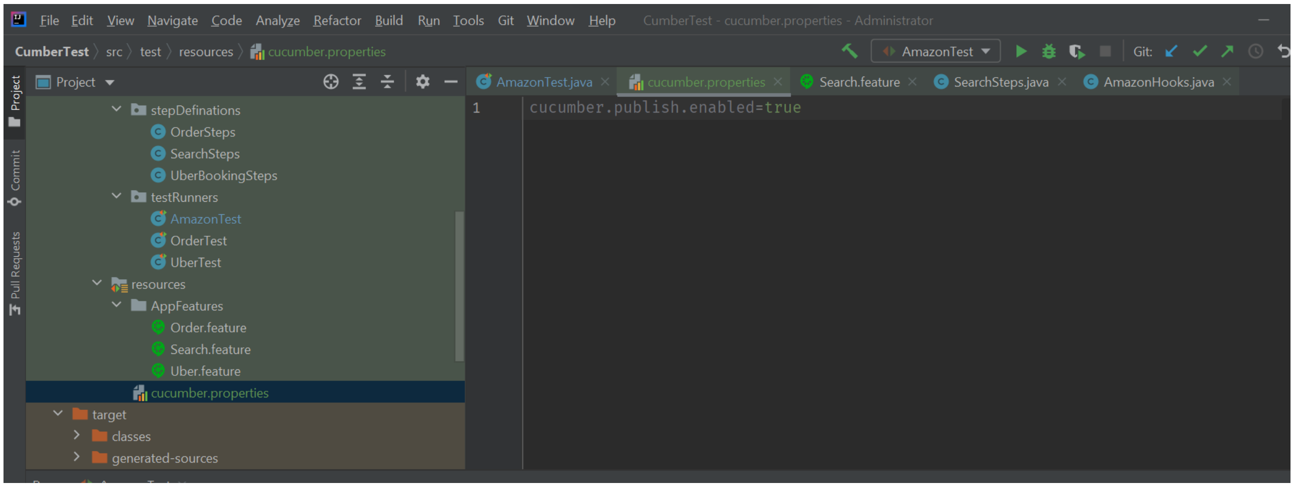
==

2nd option/alternative of web report using cucumber.properties

src/test/resources/ : cucumber.publish.enabled=true | **//Create a properties file**

Create a file under: src/test/resources/cucumber.properties

In this case disabled the publish=true



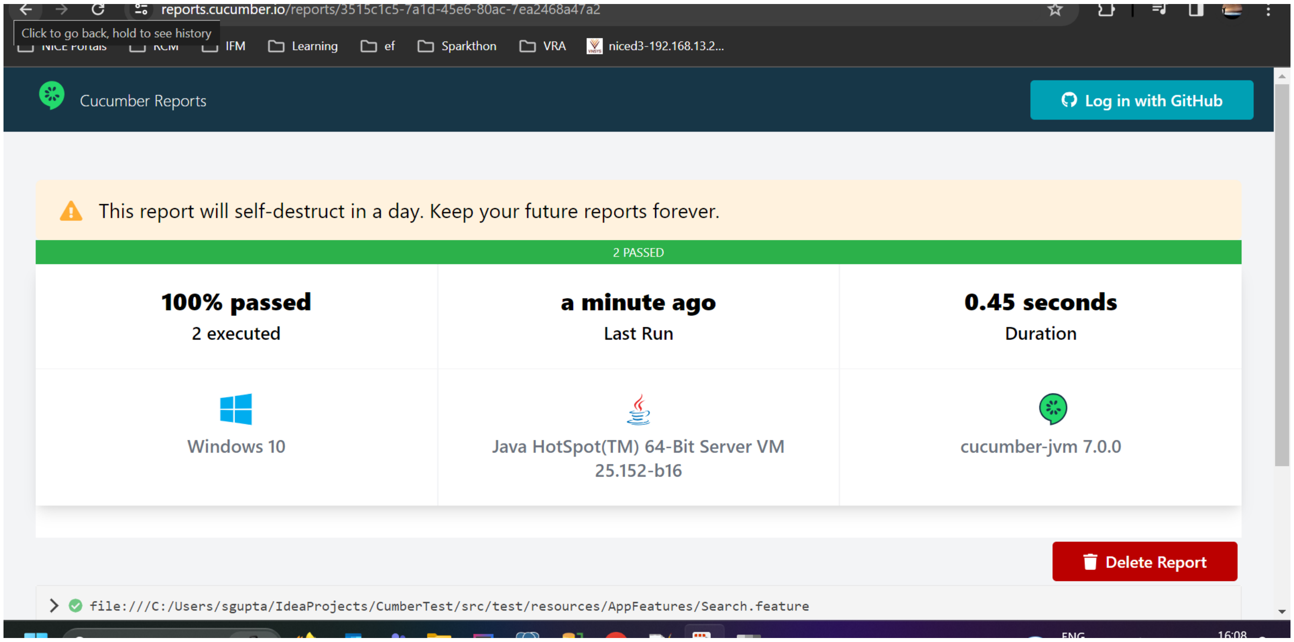
```
cucumber.publish.enabled=true
```

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
      features = {"src/test/resources/AppFeatures/Search.feature"},       //Path of feature file
      glue = {"stepDefinations","myHooks"},           //path of Stepdefinations and hooks
      plugin = {"pretty",           //Formatting
                 "json:target/Report/MyJsonReport.json",
                 "junit:target/Report/MyXMLReport.xml"}  ,
     //publish = true,                              //In this case disabled the publish=true
      tags="@Smoke or @Regression"
)
public class AmazonTest {

}
```

And run

```
@Smoke
Scenario: Search a Product mac book                          #
src/test/resources/AppFeatures/Search.feature:4
Launch Chrome browser
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                      #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
```

```
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser

@Regression
Scenario: Search a Product iPhone                         # src/test/resources/AppFeatures/Search.feature:10
Takes Screenshot on each step
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step
Takes Screenshot on each step
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step
┌──────────────────────────────────────────────────────────────────────────────
──────────┐
│ View your Cucumber Report at:                                   │
│ https://reports.cucumber.io/reports/3515c1c5-7a1d-45e6-80ac-7ea2468a47a2 │
│                                              │
│ This report will self-destruct in 24h.                          │
│ Keep reports forever: https://reports.cucumber.io/profile             │

Process finished with exit code 0
└──────────────────────────────────────────────────────────────────────────────
──────────┘
```


reports.cucumber.io/reports/3515c1c5-7a1d-45e6-80ac-7ea2468a47a2
Click to go back, hold to see history
IFM
Learning
ef
Sparkthon
VRA
niced3-192.168.13.2...
Cucumber Reports
Log in with GitHub
Delete Report

This report will self-destruct in a day. Keep your future reports forever.

2 PASSED

100% passed 2 executed	**a minute ago** Last Run	**0.45 seconds** Duration
Windows 10	Java HotSpot(TM) 64-Bit Server VM 25.152-b16	cucumber-jvm 7.0.0

file:///C:/Users/sgupta/IdeaProjects/CumberTest/src/test/resources/AppFeatures/Search.feature

==

What is monochrome true in Cucumber?

Specify different cucumber options in Java using Eclipse

Monochrome. This option can either set as true or false. If it is set as true, it means that the console output for the Cucumber test are much more readable. And if it is set as false, then the console output is not as readable as it should be

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
	features = {"src/test/resources/AppFeatures/Search.feature"},	//Path of feature file
	glue = {"stepDefinations","myHooks"},		//path of Stepdefinations and hooks
	plugin = {"pretty",		//Formatting
			"json:target/Report/MyJsonReport.json",
			"junit:target/Report/MyXMLReport.xml"}  ,
       //publish = true,
	tags="@Smoke or @Regression",
	monochrome=true
)
public class AmazonTest {

}
```

```
Output
@Smoke
Scenario: Search a Product mac book                              #
src/test/resources/AppFeatures/Search.feature:4
Launch Chrome browser
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page                      #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Step 2: I search a product with name : Apple mac book pro and price : 10000
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Step 3: I have a search field on Amazon Page: Apple mac book pro
searched product is: Apple mac book pro
  Then Product with name "Apple mac book pro" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Close Chrome browser

@Regression
Scenario: Search a Product iPhone                         # src/test/resources/AppFeatures/Search.feature:10
Takes Screenshot on each step
Step 1: I have a search field on Amazon Page
  Given I have a search field on Amazon Page              #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
Refresh on each step
Takes Screenshot on each step
Step 2: I search a product with name : iPhone and price : 1200
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
Refresh on each step
Takes Screenshot on each step
Step 3: I have a search field on Amazon Page: iPhone
searched product is: iPhone
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
Refresh on each step
┌──────────────────────────────────────────────────────────────────────────────
──────────┐
│ View your Cucumber Report at:                                   │
│ https://reports.cucumber.io/reports/c13d0fd7-e114-4d18-af97-f6ec402637f6 │
│                                                   │
│ This report will self-destruct in 24h.                          │
│ Keep reports forever: https://reports.cucumber.io/profile               │

Process finished with exit code 0
└──────────────────────────────────────────────────────────────────────────────
──────────┘
```

===

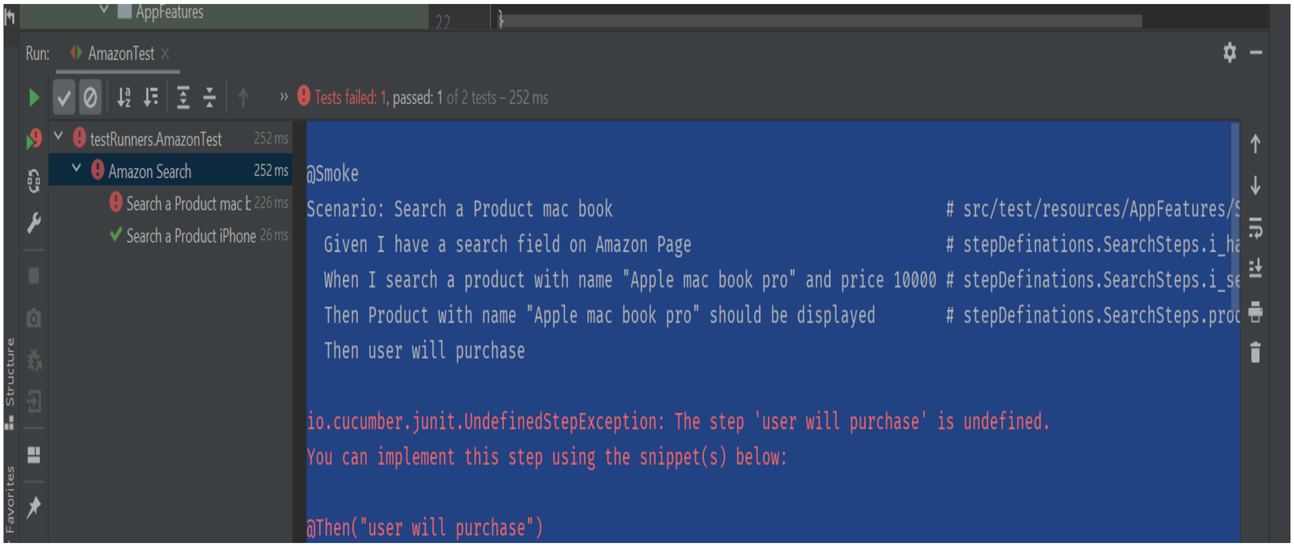
Dry run: It is not executing but it will tell their are some steps which are not implemented

We may quickly validate the steps in your feature using Dry Run, without having to run the code inside the appropriate step definitions, to ensure that every Step has its corresponding Step Definition present in the Step Definition file. A cucumber dry run is used to confirm the compilation faults and compile the Step Definition and Feature files.

The Dry Run option can either be set as true or false. By default, it is false.

```
Feature: Amazon Search

  @Smoke
  Scenario: Search a Product mac book
    Given I have a search field on Amazon Page
    When I search a product with name "Apple mac book pro" and price 10000
    Then Product with name "Apple mac book pro" should be displayed
    Then user will purchase                                    //not defined

  @Regression
  Scenario: Search a Product iPhone
    Given I have a search field on Amazon Page
    When I search a product with name "iPhone" and price 1200
    Then Product with name "iPhone" should be displayed
```

Output



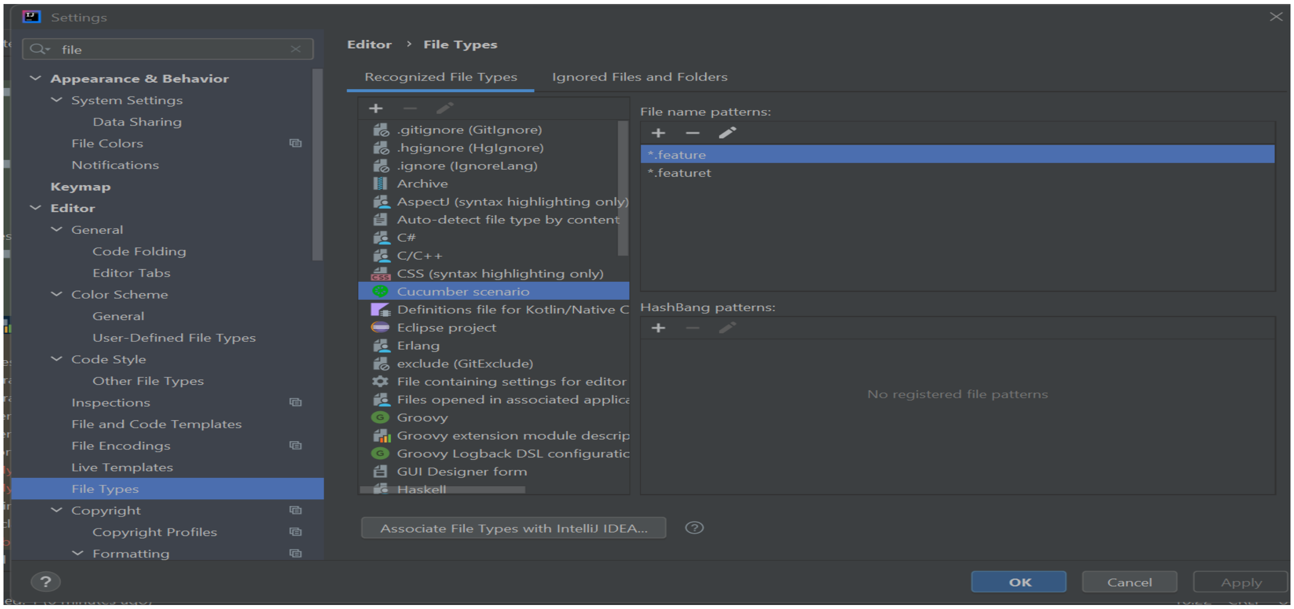
```
Tests failed: 1, passed: 1 of 2 tests – 252 ms

testRunners.AmazonTest 252 ms
  Amazon Search 252 ms
    Search a Product mac b 226 ms
    Search a Product iPhone 26 ms
```

```
@Smoke
Scenario: Search a Product mac book                                 # src/test/resources/AppFeatures/S
  Given I have a search field on Amazon Page                        # stepDefinations.SearchSteps.i_ha
  When I search a product with name "Apple mac book pro" and price 10000 # stepDefinations.SearchSteps.i_se
  Then Product with name "Apple mac book pro" should be displayed   # stepDefinations.SearchSteps.proc
  Then user will purchase

io.cucumber.junit.UndefinedStepException: The step 'user will purchase' is undefined.
You can implement this step using the snippet(s) below:

@Then("user will purchase")
```

```
@Smoke
Scenario: Search a Product mac book                              #
src/test/resources/AppFeatures/Search.feature:4
  Given I have a search field on Amazon Page                            #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
  When I search a product with name "Apple mac book pro" and price 10000 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
  Then Product with name "Apple mac book pro" should be displayed        #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
  Then user will purchase

io.cucumber.junit.UndefinedStepException: The step 'user will purchase' is undefined.
You can implement this step using the snippet(s) below:

@Then("user will purchase")
public void user_will_purchase() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}



@Regression
Scenario: Search a Product iPhone                      # src/test/resources/AppFeatures/Search.feature:11
  Given I have a search field on Amazon Page              #
stepDefinations.SearchSteps.i_have_a_search_field_on_amazon_page()
  When I search a product with name "iPhone" and price 1200 #
stepDefinations.SearchSteps.i_search_a_product_with_name_and_price(java.lang.String,java.lang.Integer)
  Then Product with name "iPhone" should be displayed       #
stepDefinations.SearchSteps.product_with_name_should_be_displayed(java.lang.String)
```

===================================

For cucumber feature file



Editor › File Types

Recognized File Types: Cucumber scenario

File name patterns:
*.feature
*.featuret

HashBang patterns:
No registered file patterns

Notes/QA

Notes/QA

Notes/QA

Notes/QA

==

How to create a scenario with data table values in feature file
How to implement data table in step definition file
What is DataTable Object
DataTable (asLists) in Cucumber BDD

We need to convert in one of the following

```
 // Write code here that turns the phrase above into concrete actions
        // For automatic transformation, change DataTable to one of
        // E, List<E>, List<List<E>>, List<Map<K,V>>, Map<K,V> or
        // Map<K, List<V>>. E,K,V must be a String, Integer, Float,
        // Double, Byte, Short, Long, BigInteger or BigDecimal.
        //
        // For other transformations you can register a DataTableType.
```

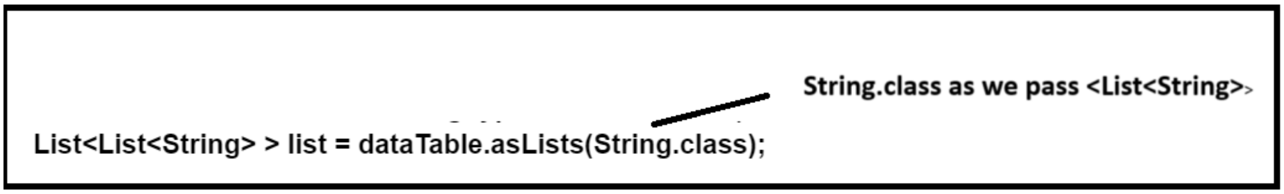
Class : need to use **DataTable class**

Registration.feature

```
Feature: User Registration

Scenario: user Registration with different data
  Given User is on Registration page
  When User enters following user details
      |Sankalp|Automation|sankalp@gmail.com|92222|Pune|
      |Harish|QA|hb@gmail.com|33222|Gonda|
      |Vikalp|Dev|vk@gmail.com|92222|Bangalore|
  Then User Registration should be successful
```

UserRegistrationSteps.java

```
package stepDefinations;

import io.cucumber.datatable.DataTable;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

import java.util.List;

public class UserRegistrationSteps {
    @Given("User is on Registration page")
    public void user_is_on_registration_page() {
        System.out.println("User is on Registration page");
    }
    @When("User enters following user details")
    public void user_enters_following_user_details(DataTable dataTable) {

        //List<List<E>> implemented
        //Here asLists return list of String type list as we pass (String.class)
        List<List<String> > list = dataTable.asLists(String.class);
        for (List<String> tempList : list)
        {
            System.out.println(tempList);
        }
    }
    @Then("User Registration should be successful")
    public void user_registration_should_be_successful() {
        System.out.println("User Registration should be successful");
    }

}
```



String.class as we pass <List<String>>

```
List<List<String> > list = dataTable.asLists(String.class);
```

Runner class: RegistrationTest.java

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Registration.feature"},       //Path of feature file
        glue = {"stepDefinations"},             //path of Step defination and hooks
        plugin = {"pretty",             //Formatting
                "json:target/Report/MyJsonReport.json",
                "junit:target/Report/MyXMLReport.xml"}  ,
      //publish = true,
        monochrome=false,
        dryRun = false
)
public class RegistrationTest {

}
```

output

```
Scenario: user Registration with different data # src/test/resources/AppFeatures/Registration.feature:3
User is on Registration page
  Given User is on Registration page            #
stepDefinations.UserRegistrationSteps.user_is_on_registration_page()
[Sankalp, Automation, sankalp@gmail.com, 92222, Pune]
[Harish, QA, hb@gmail.com, 33222, Gonda]
[Vikalp, Dev, vk@gmail.com, 92222, Bangalore]
  When User enters following user details       #
stepDefinations.UserRegistrationSteps.user_enters_following_user_details(io.cucumber.datatable.DataTable)
    | Sankalp | Automation | sankalp@gmail.com | 92222 | Pune      |
    | Harish  | QA         | hb@gmail.com      | 33222 | Gonda     |
    | Vikalp  | Dev        | vk@gmail.com      | 92222 | Bangalore |
User Registration should be successful
  Then User Registration should be successful   #
stepDefinations.UserRegistrationSteps.user_registration_should_be_successful()
```

==

Notes/QA

Notes/QA

Notes/QA

Notes/QA

Notes/QA

Notes/QA

Notes/QA

Notes/QA

How to create a scenario with data table values in feature file with columns

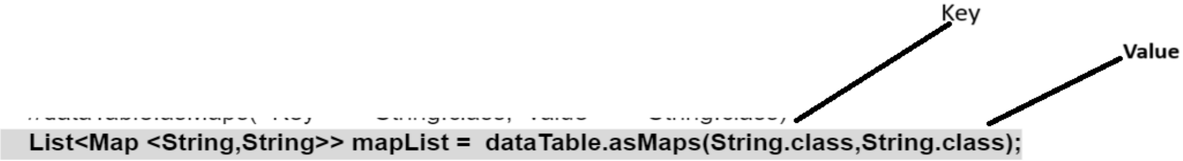
How to implement data table in step definition file
What is DataTable Object
How to get List of Maps from DataTable

Registration.feature

```
Feature: User Registration

Scenario: user Registration with different data
  Given User is on Registration page
  When User enters following user details
       |Sankalp|Automation|sankalp@gmail.com|92222|Pune|
       |Harish|QA|hb@gmail.com|33222|Gonda|
       |Vikalp|Dev|vk@gmail.com|92222|Bangalore|
  Then User Registration should be successful

Scenario: user Registration with different data with column
  Given User is on Registration page
  When User enters following user details with column
     |name|profile|email|mobile|city|
     |Sankalp|Automation|sankalp@gmail.com|92222|Pune|
     |Harish|QA|hb@gmail.com|33222|Gonda|
     |Vikalp|Dev|vk@gmail.com|92222|Bangalore|
  Then User Registration should be successful
```

UserRegistrationSteps .java

```
package stepDefinations;

import io.cucumber.datatable.DataTable;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

import java.util.List;
import java.util.Map;

public class UserRegistrationSteps {

    @Given("User is on Registration page")
    public void user_is_on_registration_page() {
        System.out.println("User is on Registration page");
    }
    @When("User enters following user details")
    public void user_enters_following_user_details(DataTable dataTable) {
        //List<List<E>> implemented
        //Here asLists return list of String type list as we pass (String.class)
        List<List<String> > list = dataTable.asLists(String.class);
        for (List<String> tempList : list)
        {
            System.out.println(tempList);
        }
    }

    @When("User enters following user details with column")
    public void user_enters_following_user_details_with_column(DataTable dataTable) {
       //List<Map<K,V>>
        //dataTable.asMaps(<Key> == String.class,<value>===String.class)
        List<Map <String,String>> mapList =  dataTable.asMaps(String.class,String.class);

        //Simple way
        //output [{name=Sankalp, profile=Automation, email=sankalp@gmail.com, mobile=92222, city=Pune},
        // {name=Harish, profile=QA, email=hb@gmail.com, mobile=33222, city=Gonda},
        // {name=Vikalp, profile=Dev, email=vk@gmail.com, mobile=92222, city=Bangalore}]
        //System.out.println(mapList);
        for (Map tMap : mapList)
        {
            System.out.println(tMap.get("name"));
            System.out.println(tMap.get("profile"));
            System.out.println(tMap.get("email"));
            System.out.println(tMap.get("mobile"));
            System.out.println(tMap.get("city"));
        }
    }

    @Then("User Registration should be successful")
    public void user_registration_should_be_successful() {
        System.out.println("User Registration should be successful");
    }
}
```



```
    List<Map <String,String>> mapList =  dataTable.asMaps(String.class,String.class);
```

Runner class = RegistrationTest.java

package testRunners;

```
import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/resources/AppFeatures/Registration.feature"},       //Path of feature file
        glue = {"stepDefinations"},           //path of Step defination and hooks
        plugin = {"pretty",            //Formatting
                "json:target/Report/MyJsonReport.json",
                "junit:target/Report/MyXMLReport.xml"}  ,
       //publish = true,
        monochrome=false,
        dryRun = false
)
public class RegistrationTest {

}
```

```
Scenario: user Registration with different data # src/test/resources/AppFeatures/Registration.feature:3
User is on Registration page
  Given User is on Registration page           #
stepDefinations.UserRegistrationSteps.user_is_on_registration_page()
[Sankalp, Automation, sankalp@gmail.com, 92222, Pune]
[Harish, QA, hb@gmail.com, 33222, Gonda]
[Vikalp, Dev, vk@gmail.com, 92222, Bangalore]
  When User enters following user details      #
stepDefinations.UserRegistrationSteps.user_enters_following_user_details(io.cucumber.datatable.DataTable)
    | Sankalp | Automation | sankalp@gmail.com | 92222 | Pune      |
    | Harish  | QA         | hb@gmail.com      | 33222 | Gonda     |
    | Vikalp  | Dev        | vk@gmail.com      | 92222 | Bangalore |
User Registration should be successful
  Then User Registration should be successful   #
stepDefinations.UserRegistrationSteps.user_registration_should_be_successful()

Scenario: user Registration with different data with column #
src/test/resources/AppFeatures/Registration.feature:11
User is on Registration page
  Given User is on Registration page                     #
stepDefinations.UserRegistrationSteps.user_is_on_registration_page()
Sankalp
Automation
sankalp@gmail.com
```

```
92222
Pune
Harish
QA
hb@gmail.com
33222
Gonda
Vikalp
Dev
vk@gmail.com
92222
Bangalore
  When User enters following user details with column       #
stepDefinations.UserRegistrationSteps.user_enters_following_user_details_with_column(io.cucumber.datat
able.DataTable)
    | name    | profile    | email             | mobile | city      |
    | Sankalp | Automation | sankalp@gmail.com | 92222  | Pune      |
    | Harish  | QA         | hb@gmail.com      | 33222  | Gonda     |
    | Vikalp  | Dev        | vk@gmail.com      | 92222  | Bangalore |
User Registration should be successful
  Then User Registration should be successful              #
stepDefinations.UserRegistrationSteps.user_registration_should_be_successful()
=================================================================
```

Scenario Outline

The Scenario Outline keyword can be used to run the same Scenario multiple times, with different combinations of values.

Feature: Login Feature

```
  Scenario Outline: Login Fail - Possible Combinations
    Given user is on application landing page
    When user click on sign in button
    Then user name is displayed on login screen
    When user enters "<username>" in username field
    And user enters "<password>" in password field
    And user click on sign in button
    Then user gets login failed error message

    Examples:
    |username|password|
    |incorrectUser1|password111|
    |incorrectUser2|password112|
    |incorrectUser2|password32|
```

```
package stepDefinations;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
```

LoginSteps.java

```
public class LoginSteps {


   @Given("user is on application landing page")
   public void user_is_on_application_landing_page() {

   }
   @When("user click on sign in button")
   public void user_click_on_sign_in_button() {

   }
   @Then("user name is displayed on login screen")
   public void user_name_is_displayed_on_login_screen() {

   }
   @When("user enters {string} in username field")
   public void user_enters_in_username_field(String string) {

   }
   @When("user enters {string} in password field")
   public void user_enters_in_password_field(String string) {

   }
   @Then("user gets login failed error message")
   public void user_gets_login_failed_error_message() {

   }


}
```

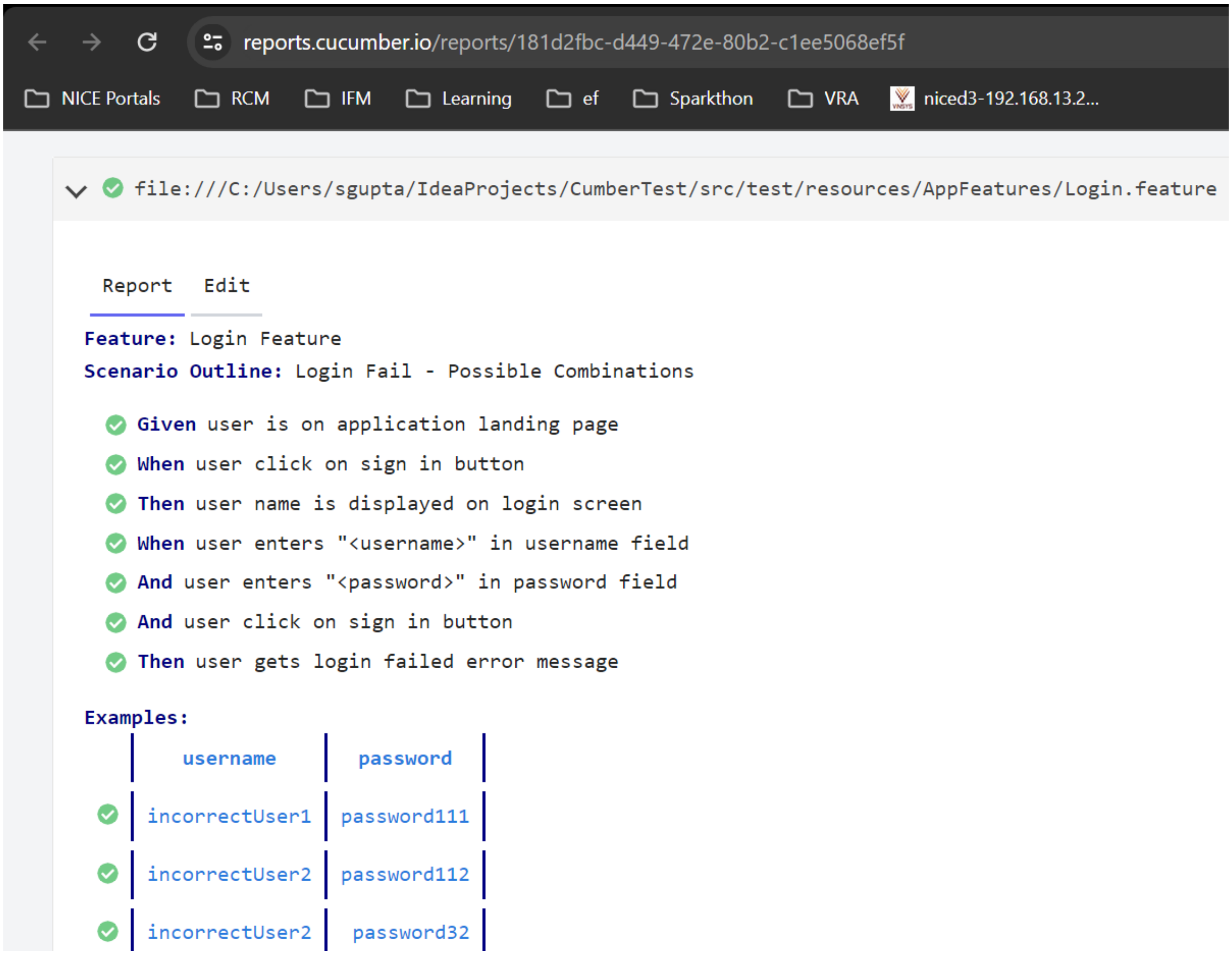
Runner class : LoginTest

```
package testRunners;


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
      features = {"src/test/resources/AppFeatures/Login.feature"},      //Path of feature file
      glue = {"stepDefinations"},                    //path of Step dedinations
      plugin = {"pretty"}    //Formatting
)
public class LoginTest {

}
```

Output

```
Scenario Outline: Login Fail - Possible Combinations  # src/test/resources/AppFeatures/Login.feature:14
  Given user is on application landing page          #
stepDefinations.LoginSteps.user_is_on_application_landing_page()
  When user click on sign in button                  #
stepDefinations.LoginSteps.user_click_on_sign_in_button()
  Then user name is displayed on login screen        #
stepDefinations.LoginSteps.user_name_is_displayed_on_login_screen()
  When user enters "incorrectUser1" in username field #
stepDefinations.LoginSteps.user_enters_in_username_field(java.lang.String)
  And user enters "password111" in password field     #
stepDefinations.LoginSteps.user_enters_in_password_field(java.lang.String)
  And user click on sign in button                   # stepDefinations.LoginSteps.user_click_on_sign_in_button()
  Then user gets login failed error message          #
stepDefinations.LoginSteps.user_gets_login_failed_error_message()

Scenario Outline: Login Fail - Possible Combinations  # src/test/resources/AppFeatures/Login.feature:15
  Given user is on application landing page          #
stepDefinations.LoginSteps.user_is_on_application_landing_page()
  When user click on sign in button                  #
stepDefinations.LoginSteps.user_click_on_sign_in_button()
  Then user name is displayed on login screen        #
stepDefinations.LoginSteps.user_name_is_displayed_on_login_screen()
  When user enters "incorrectUser2" in username field #
stepDefinations.LoginSteps.user_enters_in_username_field(java.lang.String)
  And user enters "password112" in password field     #
stepDefinations.LoginSteps.user_enters_in_password_field(java.lang.String)
  And user click on sign in button                   # stepDefinations.LoginSteps.user_click_on_sign_in_button()
  Then user gets login failed error message          #
stepDefinations.LoginSteps.user_gets_login_failed_error_message()

Scenario Outline: Login Fail - Possible Combinations  # src/test/resources/AppFeatures/Login.feature:16
  Given user is on application landing page          #
stepDefinations.LoginSteps.user_is_on_application_landing_page()
  When user click on sign in button                  #
stepDefinations.LoginSteps.user_click_on_sign_in_button()
  Then user name is displayed on login screen        #
stepDefinations.LoginSteps.user_name_is_displayed_on_login_screen()
  When user enters "incorrectUser2" in username field #
stepDefinations.LoginSteps.user_enters_in_username_field(java.lang.String)
  And user enters "password32" in password field      #
stepDefinations.LoginSteps.user_enters_in_password_field(java.lang.String)
```

```
 And user click on sign in button                 # stepDefinations.LoginSteps.user_click_on_sign_in_button()
 Then user gets login failed error message          #
stepDefinations.LoginSteps.user_gets_login_failed_error_message()
┌──────────────────────────────────────────────────────────────────────
──────────┐
│ View your Cucumber Report at:                                  │
│ https://reports.cucumber.io/reports/181d2fbc-d449-472e-80b2-c1ee5068ef5f │
│                                                        │
│ This report will self-destruct in 24h.                        │
│ Keep reports forever: https://reports.cucumber.io/profile             │
```



```
file:///C:/Users/sgupta/IdeaProjects/CumberTest/src/test/resources/AppFeatures/Login.feature

Feature: Login Feature
Scenario Outline: Login Fail - Possible Combinations

  ✓ Given user is on application landing page
  ✓ When user click on sign in button
  ✓ Then user name is displayed on login screen
  ✓ When user enters "<username>" in username field
  ✓ And user enters "<password>" in password field
  ✓ And user click on sign in button
  ✓ Then user gets login failed error message

Examples:
```

	username	password
✓	incorrectUser1	password111
✓	incorrectUser2	password112
✓	incorrectUser2	password32

Difference between Scenario Outline vs data table
Scenario Outline:

Purpose: Scenario Outline is used when you have a scenario that follows a similar structure but with different sets of data.

Usage: It is typically used when you want to test the same scenario with multiple inputs or combinations of inputs.

Example:

```
Scenario Outline: Add two numbers
  Given I have two numbers <num1> and <num2>
  When I add them together
  Then the result should be <result>

Examples:
  | num1 | num2 | result |
  | 2    | 3    | 5      |
  | 5    | 7    | 12     |
```

Explanation: In this example, the same scenario is outlined with placeholders <num1>, <num2>, and <result>. The Examples table provides different sets of values for these placeholders, allowing the scenario to be executed multiple times with different data.

DataTable:

Purpose: DataTable is used when you want to pass a tabular structure of data directly within a scenario.

Usage: It is often used when dealing with a variable number of inputs or when the data structure is more complex.

Example:

```
Scenario: Add multiple numbers
  Given I have the following numbers to add
    | Number |
    | 2      |
    | 3      |
    | 5      |
  When I add them together
  Then the result should be 10
```

Explanation: In this example, the DataTable directly provides a set of numbers to be added. The step definition for "Given I have the following numbers to add" would handle the data in the DataTable.

In summary, Scenario Outline is used when you want to run the same scenario with different sets of data, and it involves replacing placeholders in the scenario outline with values from an Examples table. DataTable, on the other hand, is used when you want to pass a table of data directly within a scenario, and the step definitions are responsible for processing and using the data.

Difference between Scenario Outline vs Scenario
Scenario Outline:

Usage: Utilized for running the same scenario with different sets of data.

Example:

```
Scenario Outline: Add two numbers
  Given I have two numbers <num1> and <num2>
  When I add them together
  Then the result should be <result>

Examples:
  | num1 | num2 | result |
  | 2    | 3    | 5      |
  | 5    | 7    | 12     |
```

Scenario (One-Liner):

Usage: Suitable for expressing simple scenarios without the need for parameterization.

Example:

```
Given I have two numbers 2 and 3
When I add them together
Then the result should be 5
```

These two styles represent different approaches based on the complexity and variability of your scenarios.

Notes/QA

Notes/QA

Notes/QA

Notes/QA