

# Gradle Project with Cucumber, Selenium and TestNG

---

 [qaautomation.expert/2022/10/03/gradle-project-with-cucumber-selenium-and-testng/](https://qaautomation.expert/2022/10/03/gradle-project-with-cucumber-selenium-and-testng/)

vibssingh

October 3, 2022

## HOME

The previous tutorial explained the Integration of Cucumber with Selenium and TestNG in a Maven Project. This tutorial explains the test automation framework based on **Gradle, Cucumber, Selenium, and TestNG**.

### ***Pre Requisite:***

---

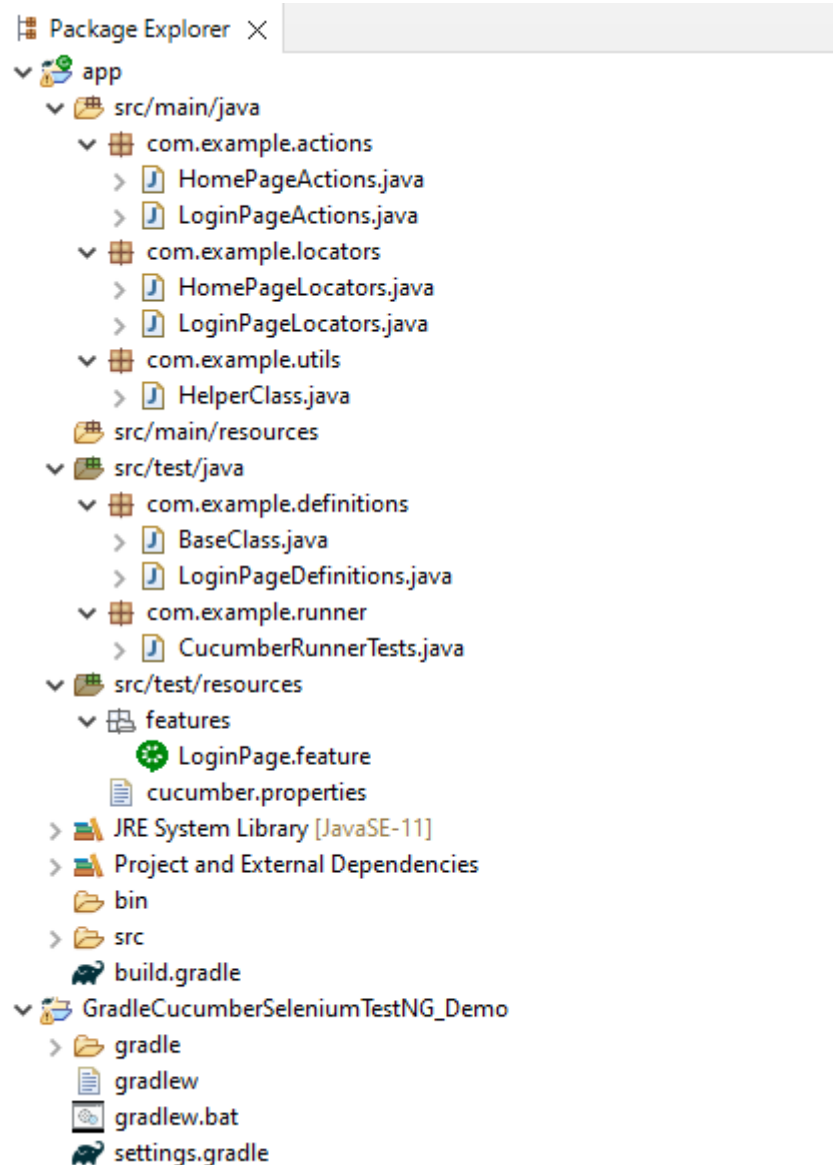
1. Java 8 or above installed
2. Eclipse or IntelliJ IDE installed
3. Gradle Installed
4. Environment variables **JAVA\_HOME** and **GRADLE\_HOME** are correctly configured

In this tutorial, I'll create a BDD Framework for the testing of web applications using **Cucumber**, and **Selenium WebDriver** with **TestNG**. This framework consists of:-

1. Cucumber Java- 7.6.0
2. Cucumber TestNG – 7.6.0
3. Java 11
4. TestNG – 7.6.0
5. Gradle – 7.5.1
6. Selenium – 4.3.0

### ***Project Structure***

---



---

### Steps to set up Cucumber Test Automation Framework with Selenium and TestNG

1. Download and Install **Java** on the system
2. Download and setup Eclipse IDE on the system
3. Install and setup Gradle
4. Install Cucumber Eclipse Plugin (For Eclipse IDE)
5. Create a new Gradle Project
6. Add **Selenium**, **TestNG**, and **Cucumber** dependencies to the **build.gradle**
7. Create a **feature** file under **src/test/resources**
8. Create the **classes for locators, actions and utilities** in **src/main/java**
9. Create the **Step Definition** class or Glue Code in **src/test/java**
10. Create a **Hook class** to contain the initialization and closing of browser in **src/test/java**
11. Create a TestNG Cucumber **Runner** class in **src/test/java**
12. Run the tests from **Command Line**
13. **Cucumber Report** Generation

---

### Implementation Steps

## Step 1- Download and Install Java

---

Cucumber and Selenium need Java to be installed on the system to run the tests. Click [here](#) to know **How to install Java.**

## Step 2 – Download and setup Eclipse IDE on the system

---

The Eclipse IDE (integrated development environment) provides strong support for Java developers. Click [here](#) to know **How to install Eclipse.**

## Step 3 – Setup Maven

---

To build a test framework, we need to add a number of dependencies to the project. Click [here](#) to know **How to install Maven.**

## Step 4 – Install Cucumber Eclipse Plugin

---

The cucumber plugin is an Eclipse plugin that allows eclipse to understand the Gherkin syntax. When we are working with cucumber we will write the feature files that contain Feature, Scenario, Given, When, Then, And, But, Tags, Scenario Outline, and Examples. By default, eclipse doesn't understand these keywords so it doesn't show any syntax highlighter. Cucumber Eclipse Plugin highlights the keywords present in Feature File. Refer to this tutorial to get more detail – **How to setup Cucumber with Eclipse.**

## Step 5 – Create a new Gradle Project

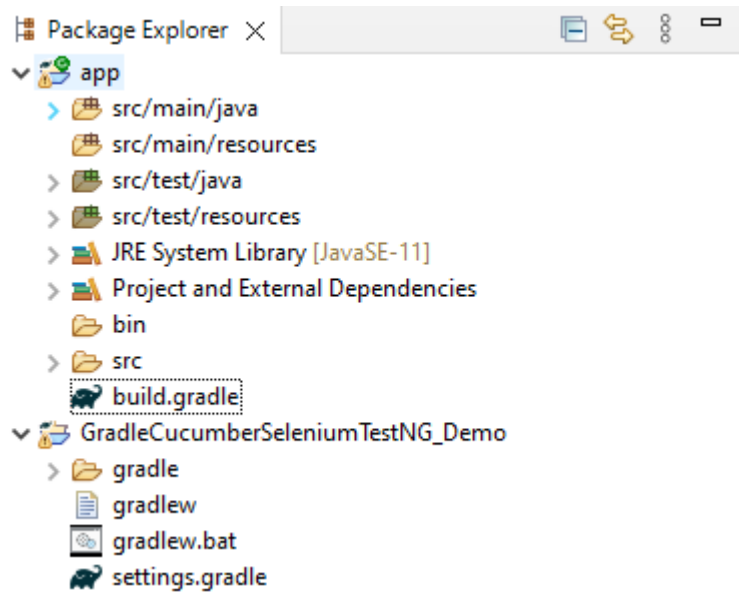
---

Below are the steps to create the Gradle project from the command line.

```
C:\eclipse-workspace_personnel\GradleCucumberSeleniumTestNG_Demo>gradle init
Select type of project to generate:
 1: basic
 2: application
 3: library
 4: Gradle plugin
Enter selection (default: basic) [1..4] 2
Select implementation language:
 1: C++
 2: Groovy
 3: Java
 4: Kotlin
 5: Scala
 6: Swift
Enter selection (default: Java) [1..6] 3
Split functionality across multiple subprojects?:
 1: no - only one application project
 2: yes - application and library projects
Enter selection (default: no - only one application project) [1..2] 1
Select build script DSL:
 1: Groovy
 2: Kotlin
Enter selection (default: Groovy) [1..2] 1
Generate build using new APIs and behavior (some features may change in the next minor release)? (default: no) [yes, no]
no
Select test framework:
 1: JUnit 4
 2: TestNG
 3: Spock
 4: JUnit Jupiter
Enter selection (default: JUnit Jupiter) [1..4] 2
Project name (default: GradleCucumberSeleniumTestNG_Demo):
Source package (default: GradleCucumberSeleniumTestNG_Demo): com.example

> Task :init
Get more help with your project: https://docs.gradle.org/7.3.3/samples/sample_building_java_applications.html
BUILD SUCCESSFUL in 1m 27s
2 actionable tasks: 2 executed
```

If you want to create the Gradle project from Eclipse IDE, click here to know [\*\*How to create a Gradle Java project. Below is the structure of the\*\*](#) Gradle project.



#### **Step 6 – Add Selenium, TestNG, and Cucumber dependencies to the build.gradle**

Add below mentioned Selenium, TestNG, and Cucumber dependencies to the project.

I have added WebDriverManager dependency to the POM.xml to download the driver binaries automatically. To know more about this, please [\*\*refer to this tutorial – How to manage driver executables using WebDriverManager.\*\*](#)

```

/*
 * This file was generated by the Gradle 'init' task.
 *
 * This generated file contains a sample Java application project to get you
started.
 */

plugins {
    // Apply the application plugin to add support for building a CLI application
in Java.
    id 'application'
    id 'io.qameta.allure' version '2.11.0'
}

repositories {
    // Use Maven Central for resolving dependencies.
    mavenCentral()
}

java {
    sourceCompatibility = 11
    targetCompatibility = 11
}

dependencies {

    // Use TestNG framework, also requires calling test.useTestNG() below
    testImplementation 'io.cucumber:cucumber-java:7.6.0'
    testImplementation 'io.cucumber:cucumber-testng:7.6.0'

    //TestNG
    testImplementation 'org.testng:testng:7.6.0'

    //Others
    implementation 'com.google.guava:guava:31.0.1-jre'
    implementation 'org.seleniumhq.selenium:selenium-java:4.4.0'
    implementation 'io.github.bonigarcia:webdrivermanager:5.3.0'
}

application {
    // Define the main class for the application.
    mainClass = 'com.example.App'
}

tasks.named('test') {
    // Use TestNG for unit tests.
    useTestNG()
}

configurations {
    cucumberRuntime {
        extendsFrom testImplementation
    }
}

task cucumber() {

```

```

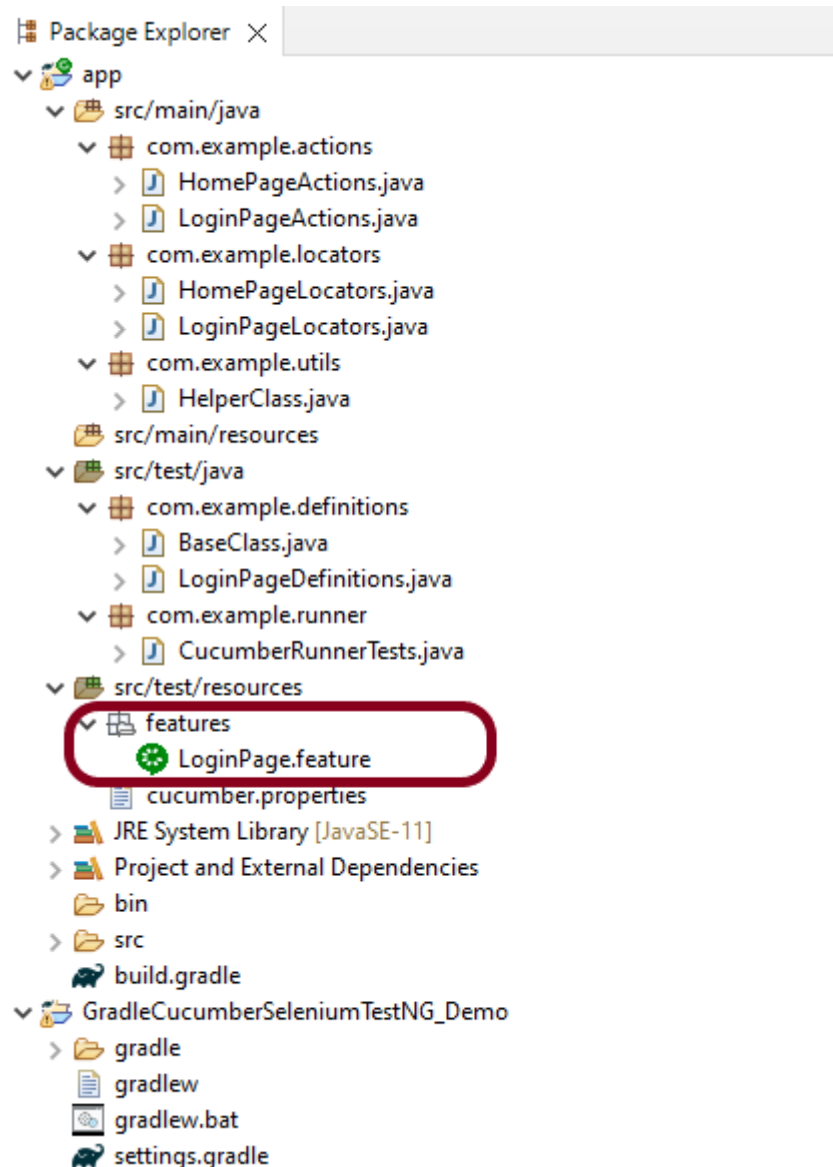
dependsOn assemble, testClasses
doLast {
    javaexec {

        main = "io.cucumber.core.cli.Main"
        classpath = configurations.cucumberRuntime + sourceSets.main.output +
sourceSets.test.output
        args = ['--plugin', 'pretty',
            '--glue', 'com.example.definitions', 'src/test/resources']
    }
}
}

```

### Step 7 – Create a feature file in the src/test/resources directory

Create a folder with name **features**. Now, create the feature file in this folder. The feature file should be saved with the extension **.feature**. This feature file contains the test scenarios created to test the application. The Test Scenarios are written in Gherkins language in the format of Given, When, Then, And, But.



Below is an example of the Feature File.

Feature: Login to HRM Application

Background:

Given User is on HRMLLogin page "https://opensource-demo.orangehrmlive.com/"

@ValidCredentials

Scenario: Login with valid credentials

When User enters username as "Admin" and password as "admin123"

Then User should be able to login successfully and new page open

@InvalidCredentials

Scenario Outline: Login with invalid credentials

When User enters username as "<username>" and password as "<password>"

Then User should be able to see error message "<errorMessage>"

Examples:

username	password	errorMessage	
Admin	admin12\$\$	Invalid credentials	
admin\$\$	admin123	Invalid credentials	
abc123	xyz\$\$	Invalid credentials	

@MissingUsername

Scenario: Verify error message when username is missing

When User enters username as "" and password as "admin123"

Then User should be able to see error message "Empty Username"

## ***Step 8 – Create the classes for locators, actions and utilities in src/main/java***

---

Below is the sample code of the **LoginPageLocators**.

```

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;

public class LoginPageLocators {

    @FindBy(name = "username")
    public WebElement userName;

    @FindBy(name = "password")
    public WebElement password;

    @FindBy(xpath = "//*
[id='app']/div[1]/div/div[1]/div/div[2]/div[2]/form/div[3]/button")
    public WebElement login;

    @FindBy(xpath = "//*
[id='app']/div[1]/div/div[1]/div/div[2]/div[2]/div/div[1]/div[1]/p")
    public WebElement errorMessage;

    @FindBy(xpath = "//*
[id='app']/div[1]/div/div[1]/div/div[2]/div[2]/form/div[1]/div/span")
    public WebElement missingUsernameErrorMessage;

}

```

Below is the sample code for the **HomePageLocators**.

```

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;

public class HomePageLocators {

    @FindBy(xpath = "//*
[id='app']/div[1]/div[2]/div[2]/div/div[1]/div[1]/div[1]/h5")
    public WebElement homePageUserName;

}

```

Create the **action classes** for each web page. These action classes contain all the methods needed by the step definitions. In this case, I have created 2 action classes – **LoginPageActions** and **HomePageActions** .

### ***LoginPageActions***



```

import org.openqa.selenium.support.PageFactory;
import com.example.locators.LoginPageLocators;
import com.example.utils.HelperClass;

public class LoginPageActions {

    LoginPageLocators loginPageLocators = null;

    public LoginPageActions() {

        this.loginPageLocators = new LoginPageLocators();

        PageFactory.initElements(HelperClass.getDriver(), loginPageLocators);
    }

    public void login(String strUserName, String strPassword) {

        loginPageLocators.userName.sendKeys(strUserName);
        loginPageLocators.password.sendKeys(strPassword);
        loginPageLocators.login.click();

    }

    // Get the error message when invalid credentials are provided
    public String getErrorMessage() {
        return loginPageLocators.errorMessage.getText();
    }

    // Get the error message when username is blank
    public String getMissingUsernameText() {
        return loginPageLocators.missingUsernameErrorMessage.getText();
    }
}

```

## ***HomePageActions***

```

import org.openqa.selenium.support.PageFactory;

import com.example.locators.HomePageLocators;
import com.example.utils.HelperClass;

public class HomePageActions {

    HomePageLocators homePageLocators = null;

    public HomePageActions() {

        this.homePageLocators = new HomePageLocators();

        PageFactory.initElements(HelperClass.getDriver(),homePageLocators);
    }

    // Get the User name from Home Page
    public String getHomePageText() {
        return homePageLocators.homePageUserName.getText();
    }

}

```

Create a **Helper class** where we are initializing the web driver, initializing the web driver wait, defining the timeouts, and creating a private constructor of the class, it will declare the web driver, so whenever we create an object of this class, a new web browser is invoked.

```

import java.time.Duration;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.github.bonigarcia.wdm.WebDriverManager;

public class HelperClass {

    private static HelperClass helperClass;
    private static WebDriver driver;
    public final static int TIMEOUT = 10;

    private HelperClass() {

        WebDriverManager.chromedriver().setup();
        driver = new ChromeDriver();

        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(TIMEOUT));
        driver.manage().window().maximize();

    }

    public static void openPage(String url) {
        driver.get(url);
    }

    public static WebDriver getDriver() {
        return driver;
    }

    public static void setUpDriver() {

        if (helperClass==null) {

            helperClass = new HelperClass();
        }

    }

    public static void tearDown() {

        if(driver!=null) {
            driver.close();
            driver.quit();
        }

        helperClass = null;

    }

}

```

### ***Step 9 – Create the Step Definition class or Glue Code in src/test/java***

---

Now, we need to create the Step Definition of the Feature File –  
***LoginPageDefinitions.java.***

```

import org.testng.Assert;
import org.testng.SkipException;
import com.example.actions.HomePageActions;
import com.example.actions.LoginPageActions;
import com.example.utils.HelperClass;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class LoginPageDefinitions {

    LoginPageActions objLogin = new LoginPageActions();
    HomePageActions objHomePage = new HomePageActions();

    @Given("User is on HRMLLogin page {string}")
    public void loginTest(String url) {

        HelperClass.openPage(url);

    }

    @When("User enters username as {string} and password as {string}")
    public void goToHomePage(String userName, String passWord) {

        objLogin.login(userName, passWord);

    }

    @Then("User should be able to login sucessfully and new page open")
    public void verifyLogin() {

        Assert.assertTrue(objHomePage.getHomePageText().contains("Employee
Information"));

    }

    @Then("User should be able to see error message {string}")
    public void verifyErrorMessageForInvalidCredentials(String
expectedErrorMessage) {

        Assert.assertEquals(objLogin.getErrorMessage(), expectedErrorMessage);

    }

    @Then("User should be able to see error message for empty username as
{string}")
    public void verifyErrorMessageForEmptyUsername(String expectedErrorMessage) {

        Assert.assertEquals(objLogin.getMissingUsernameText(), expectedErrorMessage);

    }

}

```

## Step 10 – Create a Hook class to contain the initialization and closing of the browser in src/test/java

---

Below is the example of the **Hook** class where we initialize the browser as well as close the browser at the end of the execution.

```
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import com.example.utils.HelperClass;
import io.cucumber.java.After;
import io.cucumber.java.Before;
import io.cucumber.java.Scenario;

public class BaseClass {

    @Before
    public static void setUp() {

        HelperClass.setUpDriver();
    }

    @After
    public static void tearDown(Scenario scenario) {

        //validate if scenario has failed
        if(scenario.isFailed()) {
            final byte[] screenshot = ((TakesScreenshot)
HelperClass.getDriver()).getScreenshotAs(OutputType.BYTES);
            scenario.attach(screenshot, "image/png",
scenario.getName());
        }

        HelperClass.tearDown();
    }
}
```

## Step 11 – Create a TestNG Cucumber Runner class in src/test/java

---

Cucumber needs a **TestRunner** class to run the feature files. It is suggested to create a folder with the name of the **runner** in the **src/test/java** directory and create the Cucumber TestRunner class in this folder. Below is the code of the Cucumber TestRunner class.

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

@CucumberOptions(tags = "", features =
{"src/test/resources/features/LoginPage.feature"}, glue =
{"com.example.definitions"})

public class CucumberRunnerTests extends AbstractTestNGCucumberTests {

}
```

## Step 12 – Run the tests from Command Line

---

Run the below command in the command prompt to run the tests and to get the test execution report.

```
gradle cucumber
```

The output of the above program is

```
INFO: Found CDP implementation for version 106 of 104
Given User is on HRMLLogin page "https://opensource-demo.orangehrmlive.com/" # com.example.definitions.LoginPageDefinitions.LoginTest(ja
When User enters username as "" and password as "admin123" # com.example.definitions.LoginPageDefinitions.goToHomePage
Then User should be able to see error message for empty username as "Empty Username" # com.example.definitions.LoginPageDefinitions.verifyError
  java.lang.AssertionError: expected [Empty Username] but found [Required]
    at org.testng.Assert.fail(Assert.java:110)
    at org.testng.Assert.failNotEquals(Assert.java:1413)
    at org.testng.Assert.assertEqualsImpl(Assert.java:149)
    at org.testng.Assert.assertEquals(Assert.java:131)
    at org.testng.Assert.assertEquals(Assert.java:655)
    at org.testng.Assert.assertEquals(Assert.java:665)
    at com.example.definitions.LoginPageDefinitions.verifyErrorMessageForEmptyUsername(LoginPageDefinitions.java:50)
    at ??.User should be able to see error message for empty username as "Empty Username"(file:///C:/eclipse-workspace_personnel/
Embedding Verify error message when username is missing [image/png 74972 bytes]
[1664802576.353][WARNING]: Timed out connecting to Chrome, retrying...
[1664802584.432][WARNING]: Timed out connecting to Chrome, retrying...
Failed scenarios:
file:///C:/Users/Singvi04/eclipse-workspace_personnel/GradleCucumberSeleniumTestNG_Demo/app/src/test/resources/features/LoginPage.feature:26 # Ver
5 Scenarios (1 failed, 4 passed)
15 Steps (1 failed, 14 passed)
1m31.491s

java.lang.AssertionError: expected [Empty Username] but found [Required]
  at org.testng.Assert.fail(Assert.java:110)
  at org.testng.Assert.failNotEquals(Assert.java:1413)
  at org.testng.Assert.assertEqualsImpl(Assert.java:149)
  at org.testng.Assert.assertEquals(Assert.java:131)
  at org.testng.Assert.assertEquals(Assert.java:655)
  at org.testng.Assert.assertEquals(Assert.java:665)
  at com.example.definitions.LoginPageDefinitions.verifyErrorMessageForEmptyUsername(LoginPageDefinitions.java:50)
  at ??.User should be able to see error message for empty username as "Empty Username"(file:///C:/eclipse-workspace_personnel/
????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
? View your Cucumber Report at: ?
? https://reports.cucumber.io/reports/a7d17bb5-c1f8-4632-96d3-39dad99a7d96 ?
? ?
? This report will self-destruct in 24h. ?
? Keep reports forever: https://reports.cucumber.io/profile ?
????????????????????????????????????????????????????????????????????????????????????????????????????????????????????????
```

## Step 13 – Cucumber Report Generation

To get Cucumber Test Reports, add **cucumber.properties** under **src/test/resources** and add the below instruction in the file.

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

Below is the image of the Cucumber Report generated using the Cucumber Service.

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

4 PASSED

1 FAILED

**80% passed**

5 executed

**an hour ago**

Last Run

**44.24 seconds**

Duration



Windows 10



OpenJDK 64-Bit Server VM 11.0.10+9-LTS



cucumber-jvm 7.6.0

Delete Report

✖ file:///C:/Users/SingVi04/eclipse-workspace\_personnel/GradleCucumberSeleniumTestNG\_Demo/app/src/test/resources/features/LoginPage.feature

Report Edit

Feature: Login to HRM Application

Background:

✓ Given User is on HRMLogin page "https://opensource-demo.orangehrmlive.com/"

@ValidCredentials

Scenario: Login with valid credentials

✓ When User enters username as "Admin" and password as "admin123"

✓ Then User should be able to login successfully and new page open

@InvalidCredentials

Scenario Outline: Login with invalid credentials

✓ When User enters username as "<username>" and password as "<password>"

✓ Then User should be able to see error message "<errorMessage>"

Examples:

	username	password	errorMessage
✓	Admin	admin123	Invalid credentials

Examples:

	username	password	errorMessage
✓	Admin	admin123	Invalid credentials
✓	admin123	admin123	Invalid credentials
✓	abc123	xyz123	Invalid credentials

@MissingUsername

Scenario: Verify error message when username is missing

✓ When User enters username as "" and password as "admin123"

✖ Then User should be able to see error message for empty username as "Empty Username"

```
java.lang.AssertionError: expected [Empty Username] but found [Required]
    at org.testng.Assert.fail(Assert.java:110)
    at org.testng.Assert.failNotEquals(Assert.java:1413)
    at org.testng.Assert.assertEqualsImpl(Assert.java:149)
    at org.testng.Assert.assertEquals(Assert.java:131)
    at org.testng.Assert.assertEquals(Assert.java:655)
    at org.testng.Assert.assertEquals(Assert.java:665)
    at com.example.definitions.LoginPageDefinitions.verifyErrorMessageForEmptyUsername(LoginPageDefinitions.java:50)
    at ?..User should be able to see error message for empty username as "Empty Username"(file:///C:/Users/SingVi04/eclipse-workspace_personnel/GradleCucumberSeleniumTestNG_Demo/app/src/test/resources/features/LoginPage.feature:29)
```

► Attached Image

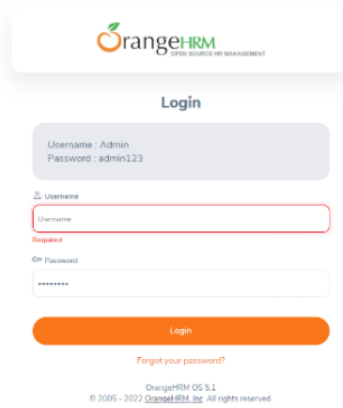
In the above example, as we can see, one of the tests has failed. So, when a test fails, we have written the code to take a screenshot of the failed step. The highlighted box above shows the image of the failed test. You can click on that to see the screenshot.

✓ **When** User enters username as "" and password as "admin123"

✗ **Then** User should be able to see error message for empty username as "Empty Username"

```
java.lang.AssertionError: expected [Empty Username] but found [Required]
    at org.testng.Assert.fail(Assert.java:110)
    at org.testng.Assert.failNotEquals(Assert.java:1413)
    at org.testng.Assert.assertEqualsImpl(Assert.java:149)
    at org.testng.Assert.assertEquals(Assert.java:131)
    at org.testng.Assert.assertEquals(Assert.java:655)
    at org.testng.Assert.assertEquals(Assert.java:665)
    at com.example.definitions.LoginPageDefinitions.verifyErrorMessageForEmptyUsername(LoginPageDefinitions.java:50)
    at ?..User should be able to see error message for empty username as "Empty Username"(file:///C:/Users/SingVi04/eclipse-workspace_personnel/GradleCucumberSeleniumTestNG_Demo/app/src/test/resources/features/LoginPage.feature:29)
```

▼ Attached Image



The screenshot shows the OrangeHRM login interface. At the top, the OrangeHRM logo is displayed. Below it, the word "Login" is centered. A light blue box contains the text "Username : Admin" and "Password : admin123". Below this, there are two input fields: "Username" and "Password". The "Username" field has a red border and a red error message "Required" below it. The "Password" field has a red border and a red error message "Required" below it. Below the input fields is an orange "Login" button. At the bottom, there is a link "Forgot your password?". The footer text reads "OrangeHRM OS 5.1" and "© 2006 - 2022 OrangeHRM, Inc. All rights reserved".



Congratulations on making it through this tutorial and hope you found it useful! Happy Learning!! Cheers!!