**Phase end Project**

1) Create Maven Project

2) Add all the dependencies for Cucumber and Selenium

3) Get the drivers

4) Create the feature files for this application https://www.saucedemo.com/

i) Login.feature

a) Positive Login using scenario outline

b) Negative Login using scenario outline

ii) Products.feature

a) Write the scenario to validate the prices of the product

5) Run your features using regression and sanity tag from runner class

6) Execute your scenarios using cmd and jenkins job

7) Publish the report on the dashboard.

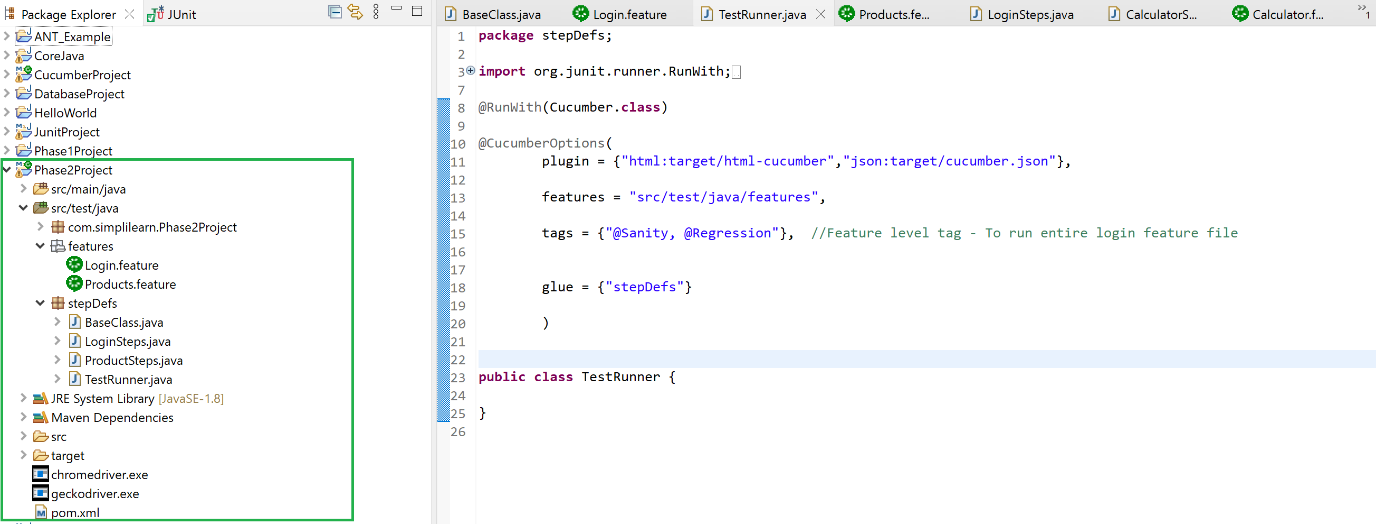
**Git Repository:** [**https://github.com/BhumikaDureja/Phase2Project.git**](https://github.com/BhumikaDureja/Phase2Project.git)

**Jenkins Report:**

**http://localhost:8080/job/Phase2Project/1/cucumber-html-reports/overview-features.html**

**Screenshot from Eclipse:**

**Screenshot of the project folder:**

****

**Screenshot of Login & Product feature files:**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**Screenshot of Login Steps & Product steps:**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Screenshot of BaseClass:**

Graphical user interface, text, application, email

Description automatically generated

**Screenshot of TestRunner with execution results on the left:**

Graphical user interface, application

Description automatically generated

**Screenshot of test run through Command Line:**

Text

Description automatically generated

Text

Description automatically generated

**Screenshot of Jenkins run:**

Graphical user interface, text, application

Description automatically generated

**Console output:**

Text

Description automatically generated with medium confidence

**Jenkins Report:**

**http://localhost:8080/job/Phase2Project/1/cucumber-html-reports/overview-features.html**

Graphical user interface, application

Description automatically generated

**Source Code:**

**Login feature file:**

@F\_Login

Feature: This feauture would be used to design the Login page of the SwagLab application

Background: Open the SwagLab application

Given I have opened the application in browser

@Sanity

Scenario Outline: Validate the successful Login

When I enter username "<UserName>"

And I enter password "<Password>"

And I click on the login button

Then I should have landed on the Product page

Examples:

| UserName | Password |

| standard\_user | secret\_sauce |

| locked\_out\_user | secret\_sauce |

@Regression

Scenario Outline: Validate the Negative Login

When I enter username "<UserName>"

And I enter password "<Password>"

And I click on the login button

Then I should get error message as "Epic sadface: Username and password do not match any user in this service"

Examples:

| UserName | Password |

| abc@xyz.com | pqr@1234 |

------------------------------------------------------------------------------------------------------------------------

**Product Feature file:**

Feature: This feauture would be used to validate the prices of the product on the SwagLab application

Background: Open the SwagLab application

Given I have opened the application in browser

When I enter username "standard\_user"

And I enter password "secret\_sauce"

And I click on the login button

Then I should have landed on the Product page

@Sanity

Scenario: Validate the price of the product

And Product price should be as per below table

| Sauce Labs Backpack | $29.99 |

| Sauce Labs Bolt T-Shirt | $49.99 |

---------------------------------------------------------------------------------------------------------------------------

**Login Steps Class:**

package stepDefs;

import org.junit.Assert;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

public class LoginSteps {

WebDriver driver = BaseClass.driver;

@Given("I have opened the application in browser")

public void i\_have\_opened\_the\_application\_in\_browser() throws InterruptedException {

// Write code here that turns the phrase above into concrete actions

driver.get("https://www.saucedemo.com/");

Thread.sleep(5000);

}

@When("I enter username {string}")

public void i\_enter\_username(String UserNameVal) {

// Write code here that turns the phrase above into concrete actions

WebElement UserName = driver.findElement(By.id("user-name"));

UserName.sendKeys(UserNameVal);

}

@When("I enter password {string}")

public void i\_enter\_password(String PasswordVal) {

// Write code here that turns the phrase above into concrete actions

WebElement Password = driver.findElement(By.id("password"));

Password.sendKeys(PasswordVal);

}

@When("I click on the login button")

public void i\_click\_on\_the\_login\_button() {

// Write code here that turns the phrase above into concrete actions

WebElement LoginBtn = driver.findElement(By.id("login-button"));

LoginBtn.click();

}

@Then("I should have landed on the Product page")

public void i\_should\_be\_landed\_on\_the\_home\_page() {

// Write code here that turns the phrase above into concrete actions

System.out.println("On the Product Page");

}

@Then("I should get error message as {string}")

public void i\_should\_get\_error\_message\_as(String ExpMsg) {

// Write code here that turns the phrase above into concrete actions

WebElement ErrorMsg = driver.findElement(By.xpath("//\*[starts-with(text(),'Epic sadface')]"));

String ActualMsg = ErrorMsg.getText();

Assert.assertEquals(ExpMsg, ActualMsg);

}

}

**Product Steps Class:**

package stepDefs;

import org.junit.Assert;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import io.cucumber.java.en.Then;

public class ProductSteps {

WebDriver driver = BaseClass.driver;

@Then("Product price should be as per below table")

public void product\_price\_should\_be\_as\_per\_below\_table(io.cucumber.datatable.DataTable dataTable) {

int rows = dataTable.height();

for(int i = 0;i<rows;i++) {

String product = dataTable.cell(i, 0);

String price = dataTable.cell(i, 1);

String actual\_price = driver.findElement(By.xpath("//div[text()='"+product +"']/following::div[@class='inventory\_item\_price']["+(i+1)+"]")).getText();

Assert.assertEquals(price, actual\_price);

//xpath for price - //div[text()='Sauce Labs Backpack']/following::div[@class='inventory\_item\_price']

// to make the xpath dynamic and to iterate through the data table we have given (i+1) so now thee price will iterate //div[text()='Sauce Labs Backpack']/following::div[@class='inventory\_item\_price'][1] and //div[text()='Sauce Labs Backpack']/following::div[@class='inventory\_item\_price'][2]

}

}

}

**Base Class:**

package stepDefs;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import io.cucumber.java.After;

import io.cucumber.java.Before;

public class BaseClass {

public static WebDriver driver;

@Before

public void SetUp() {

System.setProperty("webdriver.chrome.driver", "chromedriver.exe");

System.setProperty("webdriver.gecko.driver", "geckodriver.exe");

driver = new ChromeDriver();

driver.manage().window().maximize();

driver.manage().timeouts().implicitlyWait(5000, TimeUnit.MILLISECONDS);

}

@After()

public void TearDown() {

driver.close();

}

}

-------------------------------------------------------------------------------------------------------------------------

**Test Runner Class:**

package stepDefs;

import org.junit.runner.RunWith;

import io.cucumber.junit.Cucumber;

import io.cucumber.junit.CucumberOptions;

@RunWith(Cucumber.class)

@CucumberOptions(

plugin = {"html:target/html-cucumber","json:target/cucumber.json"},

features = "src/test/java/features",

tags = {"@Sanity, @Regression"},

glue = {"stepDefs"}

)

public class TestRunner {

}