SOFTWARE QUALITY ASSURANCE PROFESSIONAL PROGRAM

ASSIGNMENT SUBMISSION

DECLARATION

I declare that this is my own work, and this assignment does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text

Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Name	Student ID	Signature
D.H. SAMPATH	SQAP/24/31/0007	

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to SLIIT (SRILANKA INSITITUTE OF INFORMATION TECHNOLOGY) for providing me with the opportunity to work on this assignment.

I extend my heartfelt thanks to SIR CHANDANA for their invaluable guidance, support, and encouragement throughout the process. Their insights and feedback have been instrumental in the successful completion of this work.

Finally, I appreciate the resources and facilities provided by the university, which greatly contributed to the accomplishment of this assignment.

[HASHAN]
[SQA PROFESSIONAL PROGRAMME]
[28/01/2025]

TABLE OF CONTENT

Contents

1 INTRODUCTION

Software testing is a method of testing whether the actual software product is as expected requirements and ensuring that the software product is error-free. There are two types of tests: Manual testing and Automation testing. Manual software testing is the human quality assurance engineers test the quality of newly developed software without using anything automatic tools. Automated testing is the method of testing software products with specific tools and frameworks to minimize human intervention and maximize quality. Purpose both types of testing are to identify defects or defects and ensure that the product is defect-free. As software quality assurance engineers need to learn about test automation and manual testing. In this assignment, we can gain knowledge about test automation and manual testing. For that we use the " Swag Labs " open-source online business management web portal.

We can visit “Swag Labs”. The purpose of this assignment is to apply the knowledge that we gain from the Software Quality Assurance Professional program. We can think this is an example of a real-world scenario.

1.1 Test Environments

Resources	Description
Laptop	acer
Web Browser	chrome
Operation System	Windows 11
Internet	SLT Fiber

Table 1. 1 Test Environment

1.2 Test Tools

Artifact	Tool
Test case design	Mind maps
Test Automation	Java, TestNG, Selenium
Diagram drawing	Draw.io
Integrated development environment	IntelliJ IDE for Enterprise Java and Web Developers - 2024-12

Table 1. 2 Test Tool

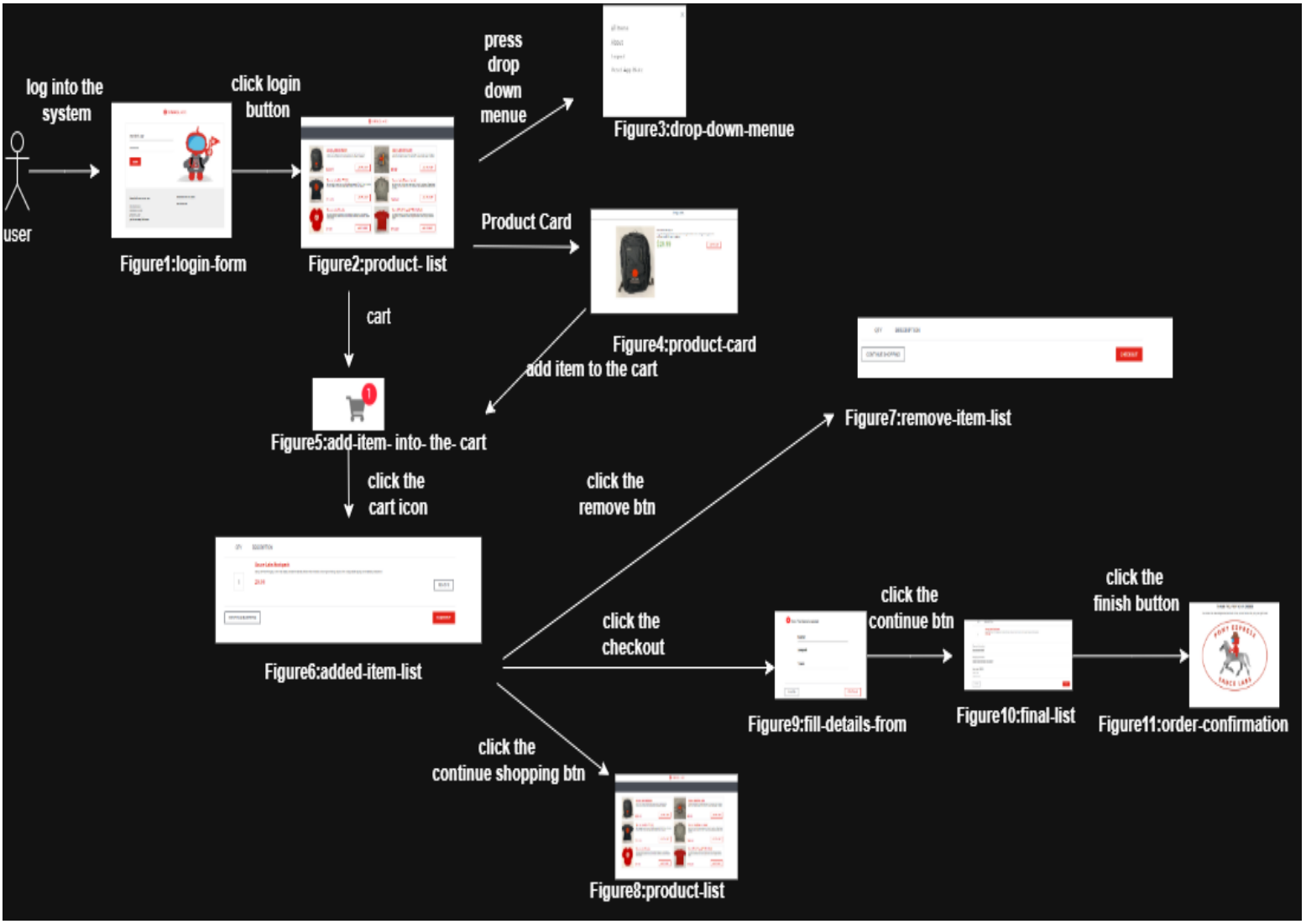
2 OBJECTIVES

In this assignment we basically focused about how to develop java basis automation testing code to test a website.

- ✓ Analysis the website for identify the testable objects and conditions
- ✓ Design the testcases and identify the suitable test techniques for testing
- ✓ Convert testcases into test scripts and prioritize them and identify test data for test the scripts
- ✓ Run the test scripts and log the test execution performs
- ✓ Generate the test completion report

3 METHODOLOGY

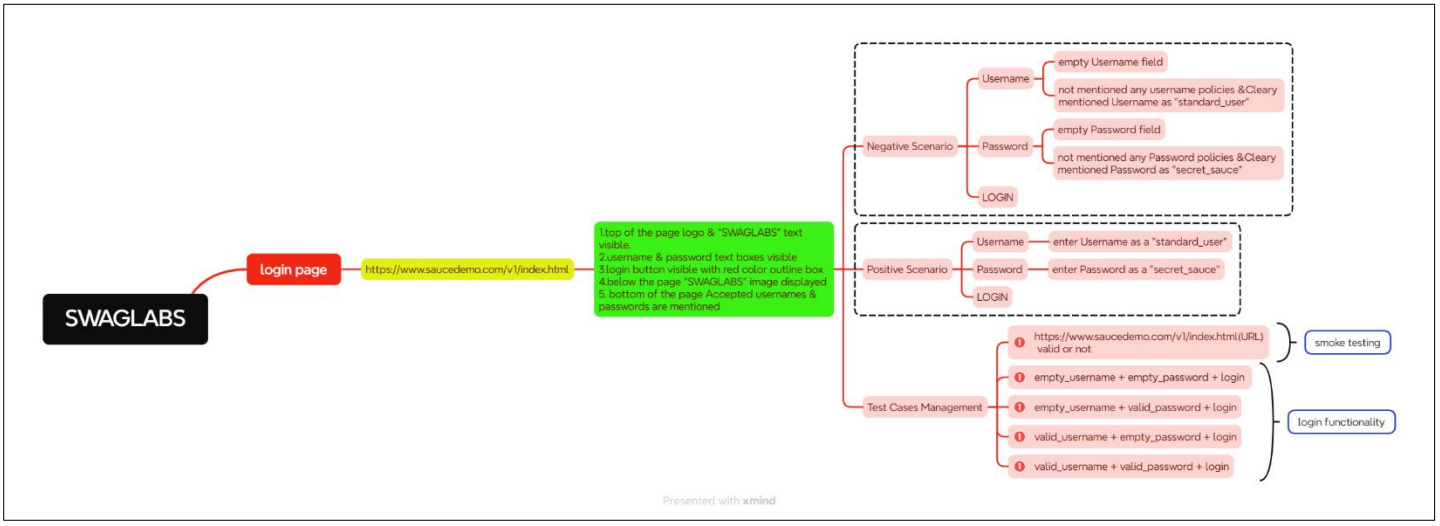
3.1 System overview



https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for clear view images)

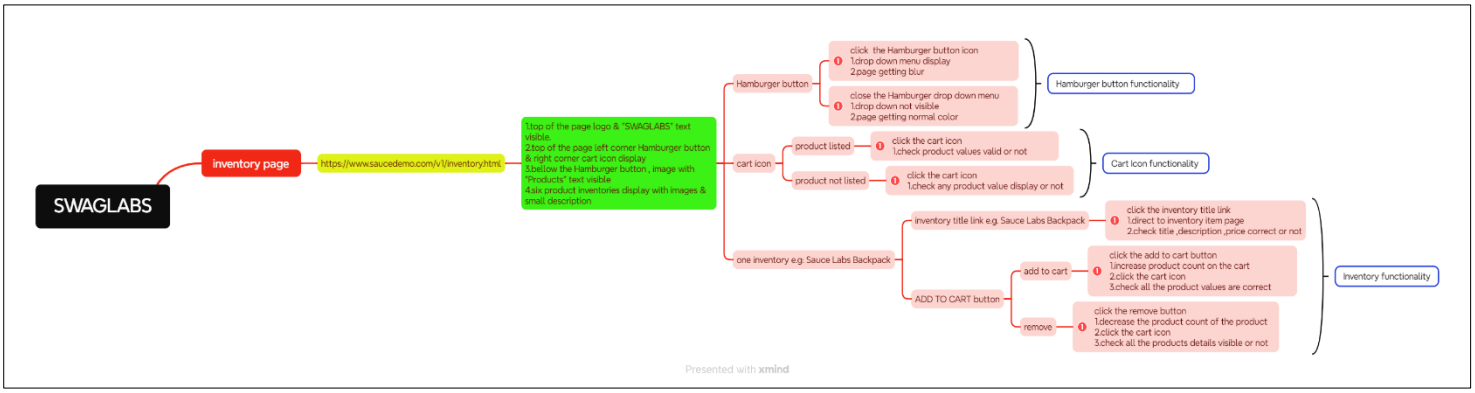
3.2 Mind Maps

3.2.1 Login Page



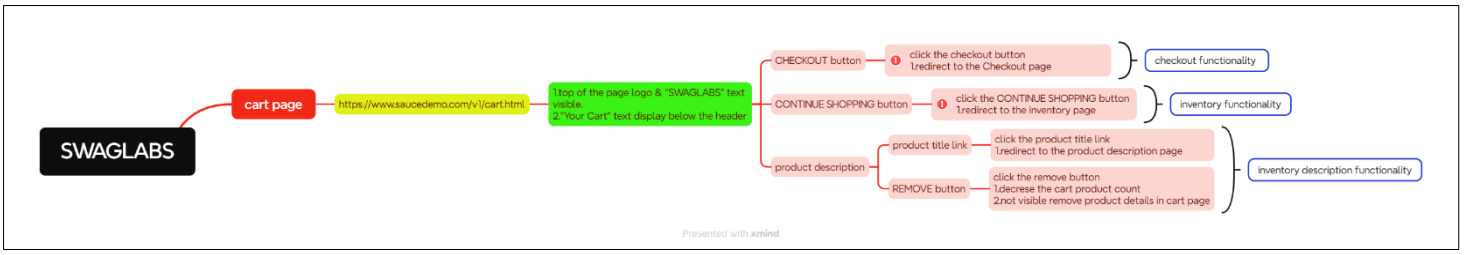
https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for clear view images)

3.2.2 Inventory Page



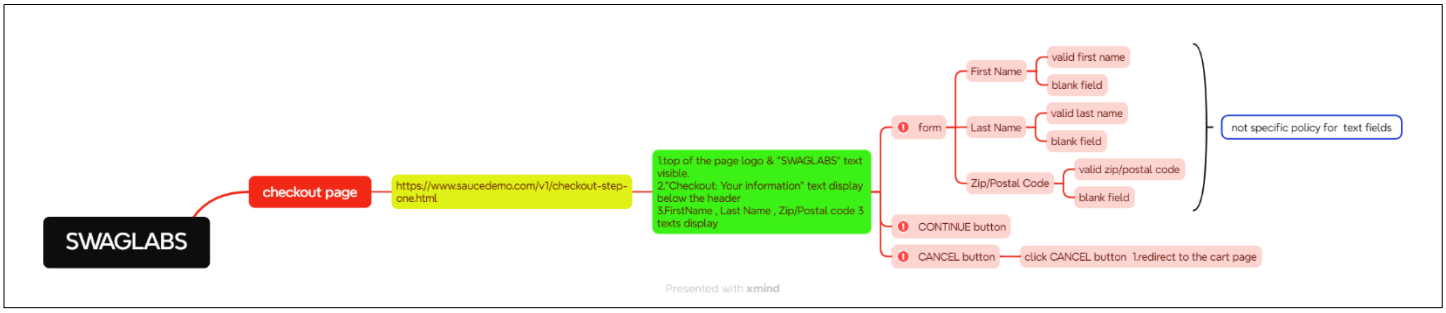
https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for clear view images)

3.2.3 Cart Page



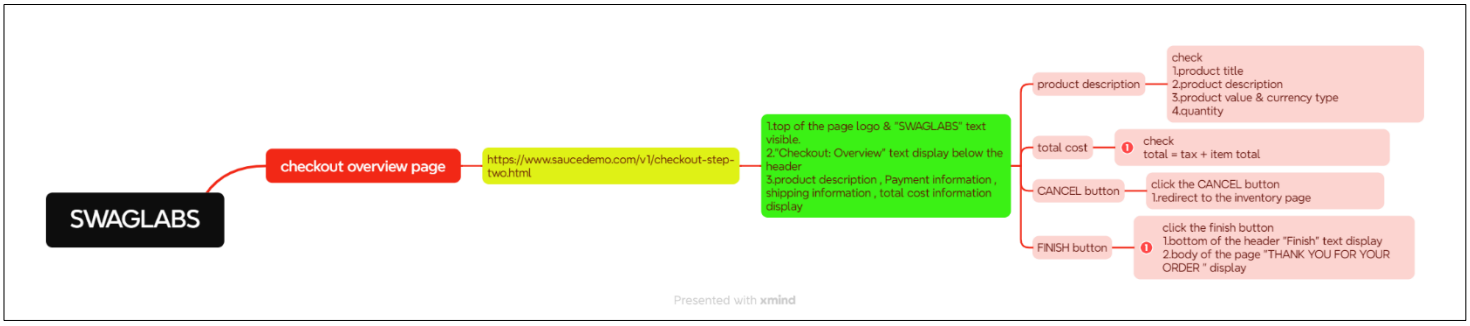
https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for clear view images)

3.2.4 Checkout Page



https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for clear view images)

3.2.5 Checkout Overview Page



https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for clear view images)

4. Test Cases

4.1 Login Functionality

Test Scenario: Login Functionality [TS_001]	
Testcase No	**Testcase Title**
TC_LF_001	Validate the given URL is valid URL (smoke testing)
TC_LF_002	Validate the Login function by providing empty username & empty password
TC_LF_003	Validate the Login function by providing empty username & valid password
TC_LF_004	Validate the Login function by providing valid username & valid password

https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for view full testcases in excel sheet)

Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
SWAGLABS	4	4	0	0	24.1 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
SWAGLABS — passed				
tests.LoginPageTest	Validate_the_Login_function_by_providing_empty_username_empty_password (login-functionality-regression) ("Validate the Login function by providing empty username & empty password")	1	1739766317674	210
	Validate_the_Login_function_by_providing_empty_username_valid_password (login-functionality-regression) ("Validate the Login function by providing empty username & valid password.....")	1	1739766317885	4310
	Validate_the_given_URL_is_valid_URL (smoke) ("Validate the given URL is valid URL")	1	1739766317659	10
	Validate_the_Login_function_by_providing_valid_username_valid_password (login-functionality-regression) ("Validate the Login function by providing valid username & valid password")	1	1739766322196	16369

SWAGLABS

tests.LoginPageTest:Validate_the_Login_function_by_providing_empty_username_empty_password

back to summary

tests.LoginPageTest:Validate_the_Login_function_by_providing_empty_username_valid_password

back to summary

tests.LoginPageTest:Validate_the_given_URL_is_valid_URL

back to summary

tests.LoginPageTest:Validate_the_Login_function_by_providing_valid_username_valid_password

back to summary

Figure1-[login-functionality-testcases]

4.2 Hamburger button Functionality

Test Scenario: Login Functionality [TS_002]	
Testcase No	**Testcase Title**
TC_HBF_001	Validate Hamburger button Functionality
TC_HBF_002	Validate Dropdown Menu Close Button Functionality

https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for view full testcases in excel sheet)

Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
SWAGLABS	2	2	0	0	35.9 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
SWAGLABS — passed				
tests.HamburgerPageTest	Validate_Hamburger_button_Functionality (HamburgerBtn-functionality-regression) ("Validate Hamburger button Functionality")	1	1739784739890	30358
	Validate_Dropdown_Menu_Close_Button_Functionality (HamburgerBtn-functionality-regression) ("Validate Hamburger button Functionality")	1	1739784770257	3201

SWAGLABS

tests.HamburgerPageTest:Validate_Hamburger_button_Functionality

back to summary

tests.HamburgerPageTest:Validate_Dropdown_Menu_Close_Button_Functionality

back to summary

Figure2-[HamburgerBtn-functionality-testcases]

4.3 Cart Functionality

Test Scenario: Cart Functionality [TS_003]	
Testcase No	**Testcase Title**
TC_CF_001	Validate "Add to Cart" Button Functionality
TC_CF_002	Validate "Remove" Button Functionality
TC_CF_003	Validate "Continue Shopping" Button Functionality

https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for view full testcases in excel sheet)

Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
SWAGLABS	3	3	0	0	51.3 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
SWAGLABS — passed				
tests.CartPageTest	Validate_Continue_ShoppingButton_Functionality (cart-functionality-regression) ("Validate Continue ShoppingButton Functionality")	1	1739771438500	174
	Validate_Add_to_Cart_Button_Functionality (cart-functionality-regression) ("Validate Add to Cart Button Functionality")	1	1739771390649	17563
	Validate_Remove_Button_Functionality (cart-functionality-regression) ("Validate Remove Button Functionality")	1	1739771408221	30275

SWAGLABS

tests.CartPageTest:Validate_Continue_ShoppingButton_Functionality

back to summary

tests.CartPageTest:Validate_Add_to_Cart_Button_Functionality

back to summary

tests.CartPageTest:Validate_Remove_Button_Functionality

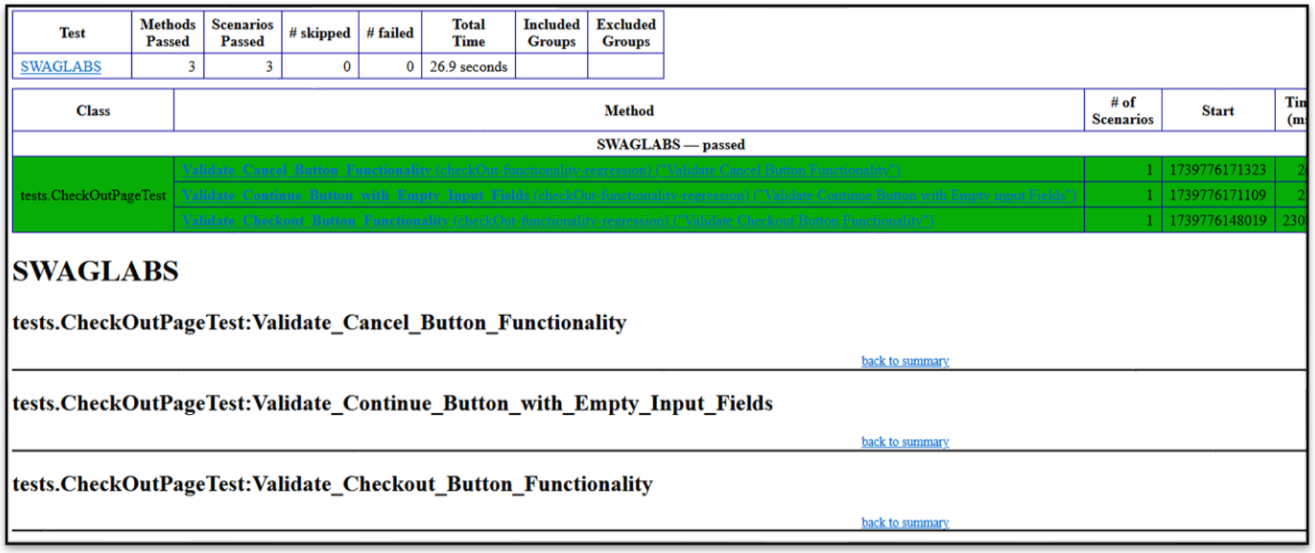
back to summary

Figure3-[cart-functionality-testcases]

4.4 Checkout Functionality

Test Scenario: Cart Functionality [TS_004]	
Testcase No	**Testcase Title**
TC_COF_001	Validate "Checkout" Button Functionality
TC_COF_002	Validate "Continue" Button with Empty Field in form
TC_COF_003	Validate "Cancel" Button Functionality

https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for view full testcases in excel sheet)



Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
SWAGLABS	3	3	0	0	26.9 seconds		

Class	Method	# of Scenarios	Start	Tin (m
SWAGLABS — passed				
tests.CheckOutPageTest	Validate_Cancel_Button_Functionality (checkOut-functionality-regression) ("Validate Cancel Button Functionality")	1	1739776171323	2
	Validate_Continue_Button_with_Empty_Input_Fields (checkOut-functionality-regression) ("Validate Continue Button with Empty input Fields")	1	1739776171109	2
	Validate_Checkout_Button_Functionality (checkOut-functionality-regression) ("Validate Checkout Button Functionality")	1	1739776148019	230

SWAGLABS

tests.CheckOutPageTest:Validate_Cancel_Button_Functionality

back to summary

tests.CheckOutPageTest:Validate_Continue_Button_with_Empty_Input_Fields

back to summary

tests.CheckOutPageTest:Validate_Checkout_Button_Functionality

back to summary

Figure4-[checkOut-functionality-testcases]

4.5 Checkout Overview Functionality

Test Scenario: Checkout Overview Functionality [TS_005]	
Testcase No	**Testcase Title**
TC_COVF_001	Validate Successful Processing of Total Product Calculation
TC_COVF_002	Validate Successful Functionality of the FINISH Button

https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for view full testcases in excel sheet)

Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
SWAGLABS	2	2	0	0	27.1 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
SWAGLABS — passed				
tests.CheckOutOverviewPageTest	Validate_Successful_Functionality_of_the_FINISH_Button(checkOutOverview-functionality-regression)("Validate Successful Functionality of the FINISH Button")	1	1739778448776	571
	Validate_Successful_Processing_of_Total_Product_Calculation(checkOutOverview-functionality-regression)("Validate Successful Processing of Total Product Calculation")	1	1739778425642	23128

SWAGLABS

tests.CheckOutOverviewPageTest:Validate_Successful_Functionality_of_the_FINISH_Button

back to summary

tests.CheckOutOverviewPageTest:Validate_Successful_Processing_of_Total_Product_Calculation

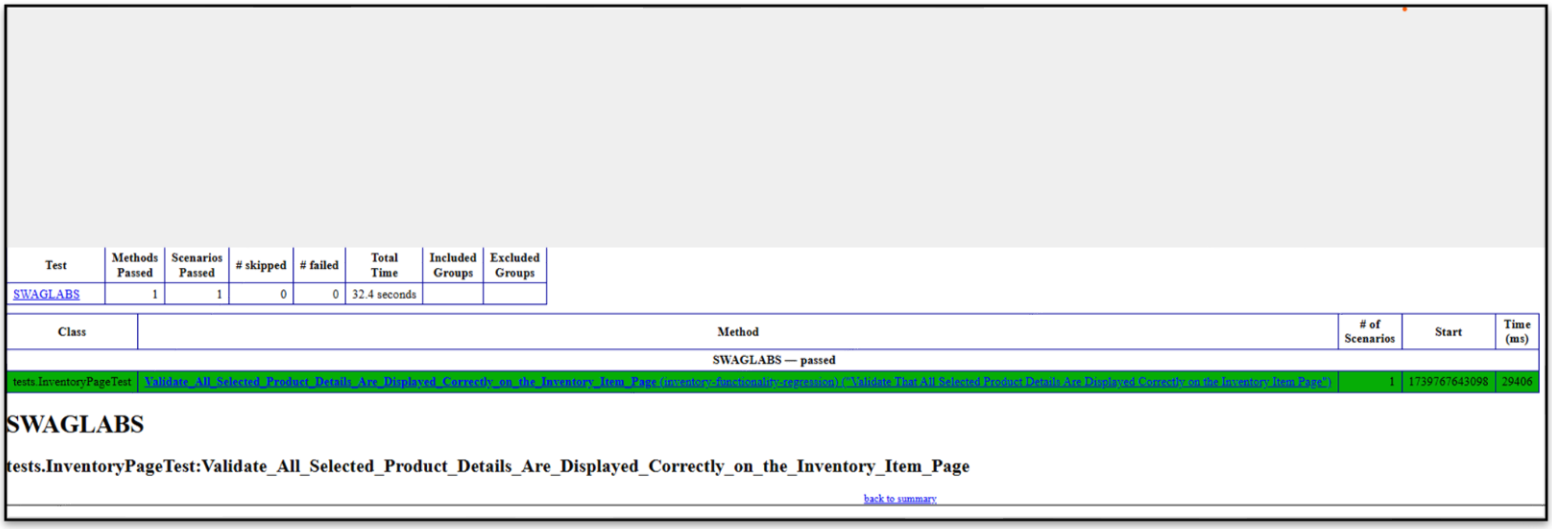
back to summary

Figure6-[checkOutOverview-functionality-testcases]

4.6 Inventory Functionality

Test Scenario: Checkout Overview Functionality [TS_006]	
Testcase No	**Testcase Title**
TC_IF_001	Validate That All Selected Product Details Are Displayed Correctly on the Inventory Item Page

https://drive.google.com/drive/folders/1bLH6gYiy3eGjxs5EBybzjFTYuKN8CaIw?usp=sharing
(Please visit Google drive link for view full testcases in excel sheet)



Test	Methods Passed	Scenarios Passed	# skipped	# failed	Total Time	Included Groups	Excluded Groups
SWAGLABS	1	1	0	0	32.4 seconds		

Class	Method	# of Scenarios	Start	Time (ms)
SWAGLABS — passed				
tests.InventoryPageTest	Validate_All_Selected_Product_Details_Are_Displayed_Correctly_on_the_Inventory_Item_Page (inventory-functionality-regression\1("Validate That All Selected Product Details Are Displayed Correctly on the Inventory Item Page")	1	1739767643098	29406

SWAGLABS

tests.InventoryPageTest:Validate_All_Selected_Product_Details_Are_Displayed_Correctly_on_the_Inventory_Item_Page

back to summary

Figure6-[checkOutOverview-functionality-testcases]

5. ISSUES

Issues	Mitigation
Time management	Worked according to the planned schedule.
Requirement identification	Refer assessment few times and refer lecture recordings.
Lack of knowledge with automation script coding	Refer lecture materials, tutorial points, and YouTube videos.

Table 5. 1 Issues

6 REFERENCES

1. Lecture recording.
2. Lecture tutorials.
3. "https://www.javatpoint.com/testng-tutorial," [Online].

7APPENDICES

7.1Source Code

7.1.1 login function

```
public class LoginPageTest extends Base   {
    @Test(
            groups = "login-functionality-regression",
//            dependsOnGroups = "smoke",
            priority = 2,
            description = "Validate the Login function by providing empty username  & empty password")
    public void Validate_the_Login_function_by_providing_empty_username_empty_password() throws InterruptedException {
        System.out.println();
        System.out.println();
        System.out.println("Validate the Login function by providing empty username  & empty password is running...............");
        loginPage.usernameInputField().sendKeys( ...keysToSend: "");
        loginPage.passwordInputField().sendKeys( ...keysToSend: "");
        loginPage.loginButton().click();
        String expectedErrorMessage = loginPage.popup_error_message().getText();
        System.out.println("expected error message :::" + expectedErrorMessage);
        String actualErrorMessage = "Epic sadface: Username is required";
        Assert.assertEquals(actualErrorMessage, expectedErrorMessage,  message: "Error message mismatch");
        System.out.println("finish the execution Validate the Login function by providing empty username  & empty password...............");
    }


    @Test(
            groups = "login-functionality-regression",
//            dependsOnGroups = "smoke",
            priority = 3,
            description = "Validate the Login function by providing empty username  & valid password.......")
    public void Validate_the_Login_function_by_providing_empty_username_valid_password() throws InterruptedException {
        System.out.println();
        System.out.println();
        System.out.println("Validate the Login function by providing empty username  & valid password is running");
        loginPage.usernameInputField().sendKeys( ...keysToSend: "");
```

```
    @Test(groups = "smoke", priority = 1, description = "Validate the given URL is valid URL")
    public void Validate_the_given_URL_is_valid_URL() throws InterruptedException {...}


    @Test(
            groups = "login-functionality-regression",
//            dependsOnGroups = "smoke",
            priority = 2,
            description = "Validate the Login function by providing empty username  & empty password")
    public void Validate_the_Login_function_by_providing_empty_username_empty_password() throws InterruptedException {
        System.out.println();
        System.out.println();
        System.out.println("Validate the Login function by providing empty username  & empty password is running...............");
        loginPage.usernameInputField().sendKeys( ...keysToSend: "");
        loginPage.passwordInputField().sendKeys( ...keysToSend: "");
        loginPage.loginButton().click();
        String expectedErrorMessage = loginPage.popup_error_message().getText();
        System.out.println("expected error message :::" + expectedErrorMessage);
        String actualErrorMessage = "Epic sadface: Username is required";
        Assert.assertEquals(actualErrorMessage, expectedErrorMessage,  message: "Error message mismatch");
        System.out.println("finish the execution Validate the Login function by providing empty username  & empty password...............");
    }
```

7.1.2 inventory function

```java
public class InventoryPageTest extends Base {

    @Test(priority = 1, groups = "inventory-functionality-regression", description = "Validate That All Selected Product Details Are Displayed Correctly on the Invent
    public void Validate_All_Selected_Product_Details_Are_Displayed_Correctly_on_the_Inventory_Item_Page() throws InterruptedException {
        userLogin(driverInventory);
        Thread.sleep( millis: 1000);
        System.out.println("navigate to the inventory page after successful login");

        System.out.println();
        System.out.println();

        //get the inventory page window handler id
        String invnetoryPageWindowHandler = driverInventory.getWindowHandle();
        System.out.println("inventory page window handle :::" + invnetoryPageWindowHandler);

        //get the title the Sauce-labs-backpack red link
        String sauceLabsBackpackClickableLinkTitle = inventoryPage.inventoryPage_sauceLabsBackpackClickableLink().getText();
        System.out.println("sauce Labs Backpack ClickableLink Title:::" + sauceLabsBackpackClickableLinkTitle);

        //get the description of the Sauce-labs-backpack red link
        String sauceLabsBackpackDescription = inventoryPage.inventoryPage_sauceLabsBackpackDescription().getText();
        System.out.println("sauce Labs Backpack Description:::" + sauceLabsBackpackDescription);

        //get the price of the Sauce-labs-backpack red link
        String sauceLabsBackpackPrice = inventoryPage.inventoryPage_sauceLabsBackpackPrice().getText();
        System.out.println("sauce Labs Backpack Price::;" + sauceLabsBackpackPrice);

        inventoryPage.inventoryPage_sauceLabsBackpackClickableLink().click();
        Thread.sleep( millis: 1000);
        System.out.println("navigate to the inventor-item page");
```

```java
Set<String> windowHandles = driverInventory.getWindowHandles();
for (String handle : windowHandles) {
    if (!handle.equals(invnetoryPageWindowHandler)) {
        driverInventory.switchTo().window(handle);
        break;
    }
}

System.out.println("inventoryItemPage url" + driverInventory.getCurrentUrl());

//initialize the driverInventory to inventItem page
inventoryItemPage = new InventoryItemPage(driverInventory);

//get title , description , price for sauce labs backpack in inventory-item page
String expect_sauceLabsBackpackClickableLinkTitle = inventoryItemPage.inventoryItemPage_sauceLabsBackpackTitle().getText();
System.out.println("expect_sauceLabsBackpackClickableLinkTitle::: " + expect_sauceLabsBackpackClickableLinkTitle);
String expect_sauceLabsBackpackDescription = inventoryItemPage.inventoryItemPage_sauceLabsBackpackDescription().getText();
System.out.println("expect_sauceLabsBackpackDescription :::" + expect_sauceLabsBackpackDescription);
String expect_sauceLabsBackpackPrice = inventoryItemPage.inventoryItemPage_sauceLabsBackpackPrice().getText();
System.out.println("expect_sauceLabsBackpackPrice :::" + expect_sauceLabsBackpackPrice);

//compare with inventory page sauce-labs-backpack title with inventory-item page  sauce-labs-backpack title
Assert.assertEquals(sauceLabsBackpackClickableLinkTitle, expect_sauceLabsBackpackClickableLinkTitle, message: "sauce labs backpack title mismatch");
//compare with inventory page sauce-labs-backpack description with inventory-item page  sauce-labs-backpack description
Assert.assertEquals(sauceLabsBackpackDescription, expect_sauceLabsBackpackDescription, message: "sauce labs backpack description mismatch");
//compare with inventory page sauce-labs-backpack price with inventory-item page  sauce-labs-backpack price
Assert.assertEquals(sauceLabsBackpackPrice, expect_sauceLabsBackpackPrice, message: "sauce labs backpack description mismatch");
```

7.1.3. cart function

```java
public class CartPageTest extends Base {
    WebDriver driverCart;  18 usages
    public LoginPage loginPage;  1 usage
    public LoginPageTest loginPageTest;  2 usages
    public InventoryPage inventoryPage;  7 usages
    public CartPage cartPage;  10 usages
    public CheckOutPage checkOutPage;  no usages

    // Pass the shared WebDriver instance to LoginPageTest
    public CartPageTest(WebDriver driver) {  1 usage
        this.driverCart = driver;
        this.cartPage =  new CartPage(driverCart);
        this.inventoryPage = new InventoryPage(driverCart);
//        this.checkOutPage = new CheckOutPage(driverCart);
    }

    @BeforeClass
    public void setUpApplication() throws IOException, InterruptedException {
        System.out.println("setup application.......");
        driverCart = initializeDriver();
        driverCart.get(prop.getProperty("url"));
        System.out.println("navigate to the url" + prop.getProperty("url"));
        loginPage = new LoginPage(driverCart);
//        loginPageTest = new LoginPageTest(driverCart);
        inventoryPage = new InventoryPage(driverCart);
        cartPage = new CartPage(driverCart);
        System.out.println("create the page object.......");
    }
```

```java
public class CartPageTest extends Base {
    @Test(priority = 1, groups = "cart-functionality-regression", description = "Validate Add to Cart Button Functionality")
    public void Validate_Add_to_Cart_Button_Functionality() throws InterruptedException {

        userLogin(driverCart);
        System.out.println("navigate to the inventory page after successful login +++++++");

        System.out.println();
        System.out.println();

        //get the current page window id for validate whether land on correct web page
        String inventoryPageWindowHandle = driverCart.getWindowHandle();
        System.out.println("inventory page window handle" + inventoryPageWindowHandle);

        inventoryPage.addToCartButton().click();
        System.out.println("click the ADD TO CART button in the inventory page relevant to the sauce backpack card");
        inventoryPage.cartIcon().click();
        System.out.println("click the CART ICON on the right corner of the inventory page");

        //navigate to the  cart web page
        Set<String> pageHandles = driverCart.getWindowHandles();
        for (String handler : pageHandles) {
            if (!handler.equals(inventoryPageWindowHandle)) {
                driverCart.switchTo().window(handler);
                System.out.println("cart web page window id:::" + driverCart.getWindowHandle());
                System.out.println(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>");
                break;
            }
        }
```

7.1.4. checkout function

```java
public class CartPageTest extends Base {
    public void setUpApplication() throws IOException, InterruptedException {
    }



    @Test(priority = 1, groups = "cart-functionality-regression", description = "Validate Add to Cart Button Functionality")
    public void Validate_Add_to_Cart_Button_Functionality() throws InterruptedException {

        userLogin(driverCart);
        System.out.println("navigate to the inventory page after successful login +++++++");

        System.out.println();
        System.out.println();

        //get the current page window id for validate whether land on correct web page
        String inventoryPageWindowHandle = driverCart.getWindowHandle();
        System.out.println("inventory page window handle" + inventoryPageWindowHandle);

        inventoryPage.addToCartButton().click();
        System.out.println("click the ADD TO CART button in the inventory page relevant to the sauce backpack card");
        inventoryPage.cartIcon().click();
        System.out.println("click the CART ICON on the right corner of the inventory page");

        //navigate to the  cart web page
        Set<String> pageHandles = driverCart.getWindowHandles();
        for (String handler : pageHandles) {
            if (!handler.equals(inventoryPageWindowHandle)) {
                driverCart.switchTo().window(handler);
                System.out.println("cart web page window id:::" + driverCart.getWindowHandle());
                System.out.println("\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\");
```

7.1.5. Hamburger button function

```
ublic class HamburgerPageTest extends Base {
    public void Validate_Hamburger_button_Functionality() throws InterruptedException {
        Actions mouseAction = new Actions(driverHamburger);
        mouseAction.moveToElement(hamburgerPage.hamburgerButton()).click().perform();
        Thread.sleep( millis: 5000);
        System.out.println("click the Hamburger button");

        //validate Hamburger Button is working
        Assert.assertTrue(isDisplayedMenueContainer(),  message: "menu close icon  button did not close the menu");

    }

    @Test( groups = "HamburgerBtn-functionality-regression", description = "Validate  Hamburger button Functionality",dependsOnMethods ="Validate_Hamburger_button_Fun
    public void Validate_Dropdown_Menu_Close_Button_Functionality() throws InterruptedException {
/       userLogin(driverHamburger);
        System.out.println("navigate to the inventory page");


        Actions action = new Actions(driverHamburger);
        action.moveToElement(hamburgerPage.sideMenueCloseButton()).click().perform();
        System.out.println("click the close icon");
        Thread.sleep( millis: 3000);
        //validate Hamburger Button is working
        Assert.assertFalse(isDisplayedMenueContainer(),  message: "Menu did not close properly");


    }
```