# SOFTWARE QUALITY ASSURANCE PROFESSIONAL PROGRAM

**B.A.A.W.S PERERA**

**1st of July 2024**

# 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 |
|---|---|---|
| B.A.A.W.S Perera | IT/SQAP/23/28/0004 | |

# Acknowledgment

In order to properly complete this assignment, I would like to thank Mr. Chandana Wijesuriya for all of his help, advice, and insightful criticism. This report's content and quality were greatly influenced by his knowledge and support.

Thank you

# Table of Contents

# List of Figures

# List of Tables

# 1. Introduction

A methodical procedure known as "Software Quality Assurance" (SQA) makes sure software products adhere to standards and specifications. Many tasks, such as planning, monitoring, testing, and process improvement, must be put into practice in order to produce software that satisfies high standards. Through the use of best practices and standards, software quality assurance, or SQA, seeks to prevent problems throughout the software development process.

People who are well-experienced in automation testing, enhance the accuracy, efficiency, and coverage of the testing process. Automation testing employs tools and scripts to automatically execute repetitive and complex test cases, minimizing human error and saving time. Assuring that fresh code modifications do not introduce new defects, it permits regular and comprehensive regression testing. Testers may accelerate test execution, support continuous integration and delivery (CI/CD), and eventually help produce software that is more dependable and robust by mastering automation testing.

In this assignment, We are able to improve our knowledge regarding automation testing. Swag Labs is one of an open-source web that we are able to use for automation testing practices. This assignment applying the knowledge that we learn from the Software quality assurance professional program. This project can be considered as a real-world scenario.

## 1.1 Test environment

| Resources | Description |
|---|---|
| Laptop | Vivobook_ASUSLaptop X1504VA_X1504VA |
| Web Browser | Google Chrome - Version 126.0.6478.63 (Official Build) (64-bit) |
| Operation System | Microsoft Windows 11 Home Single Language |
| Internet | MediaTek Wi-Fi 6E MT7902 Wireless LAN Card |

Table 1: Testing environment

### 1.2 Test tools

| Artifact | Tool |
|---|---|
| Test case design | Xmind - Mind maps |
| Test Automation | Java, TestNG, Selenium |
| Diagram drawing | MockFolw |
| Integrated development environment | IntelliJ IDEA 2023.3.4 (Community Edition) |

Table 2: Test tools

## 2. Objectives

Getting involved in an automation testing program provides an excellent opportunity to learn about modern approaches to current software development. This assignment mainly focuses on how to implement a Java basic automation testing code base for a website.

- First, we need to implement a code base for login for the Swag Labs website by using the provided username and password.
- Verify that the user navigates to the correct page and starts adding items to the cart.
- Then we need to add items to the cart as well as remove items from the cart.
- After that, View the cart and verify that the current user is on the correct page, and continue the testing process for the view cart process.
- Then, proceed to the checkout page and add order details.
- Next, validate the order summary and purchase the order.
- At the end, the user is allowed to see the successfully placed order alert and be directed back to the home page.

# 3. Methodology

## 3.1 System Overview

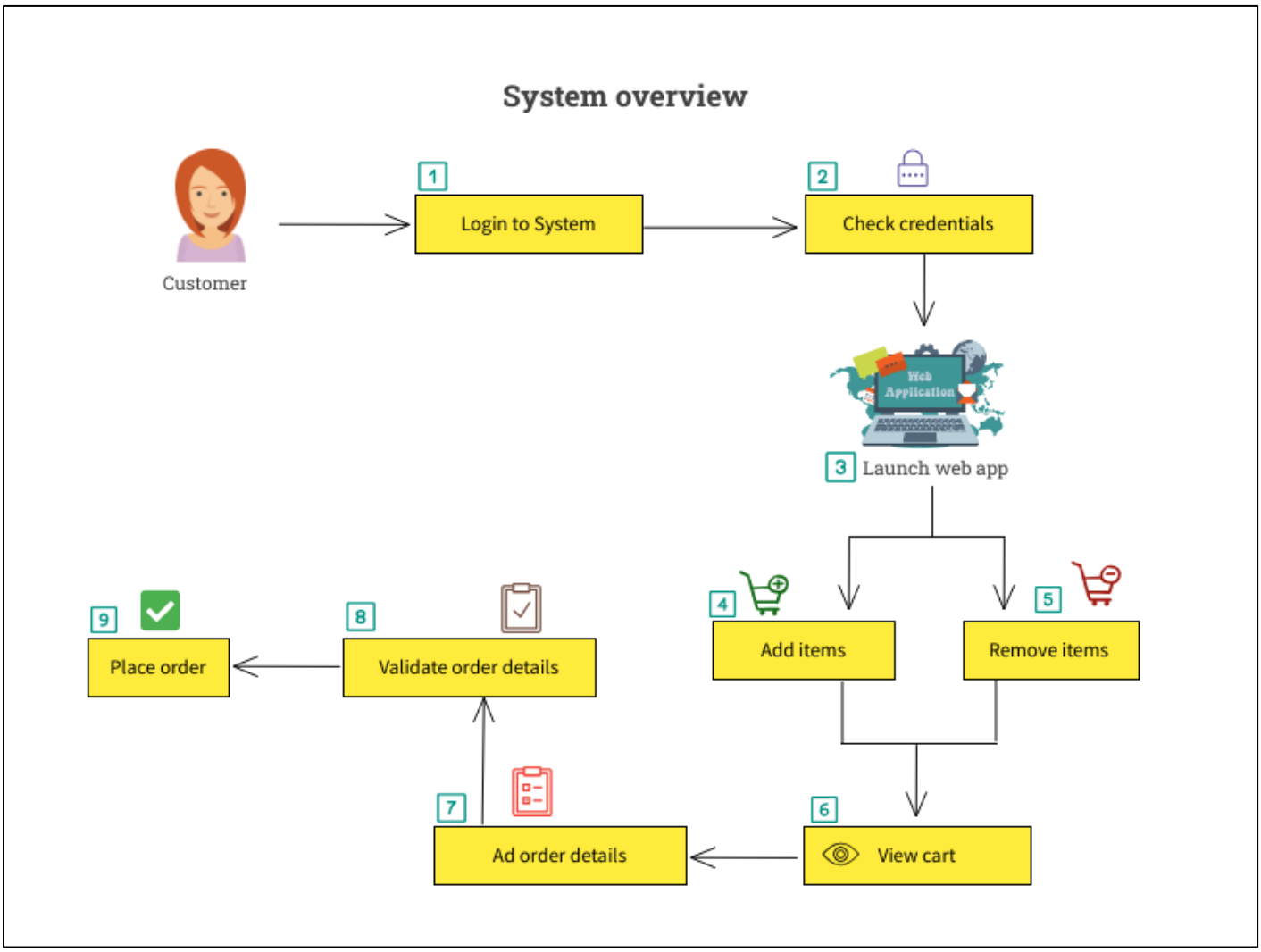



Figure 1: System Overview

## 3.2 System Overview in Swim lane diagram

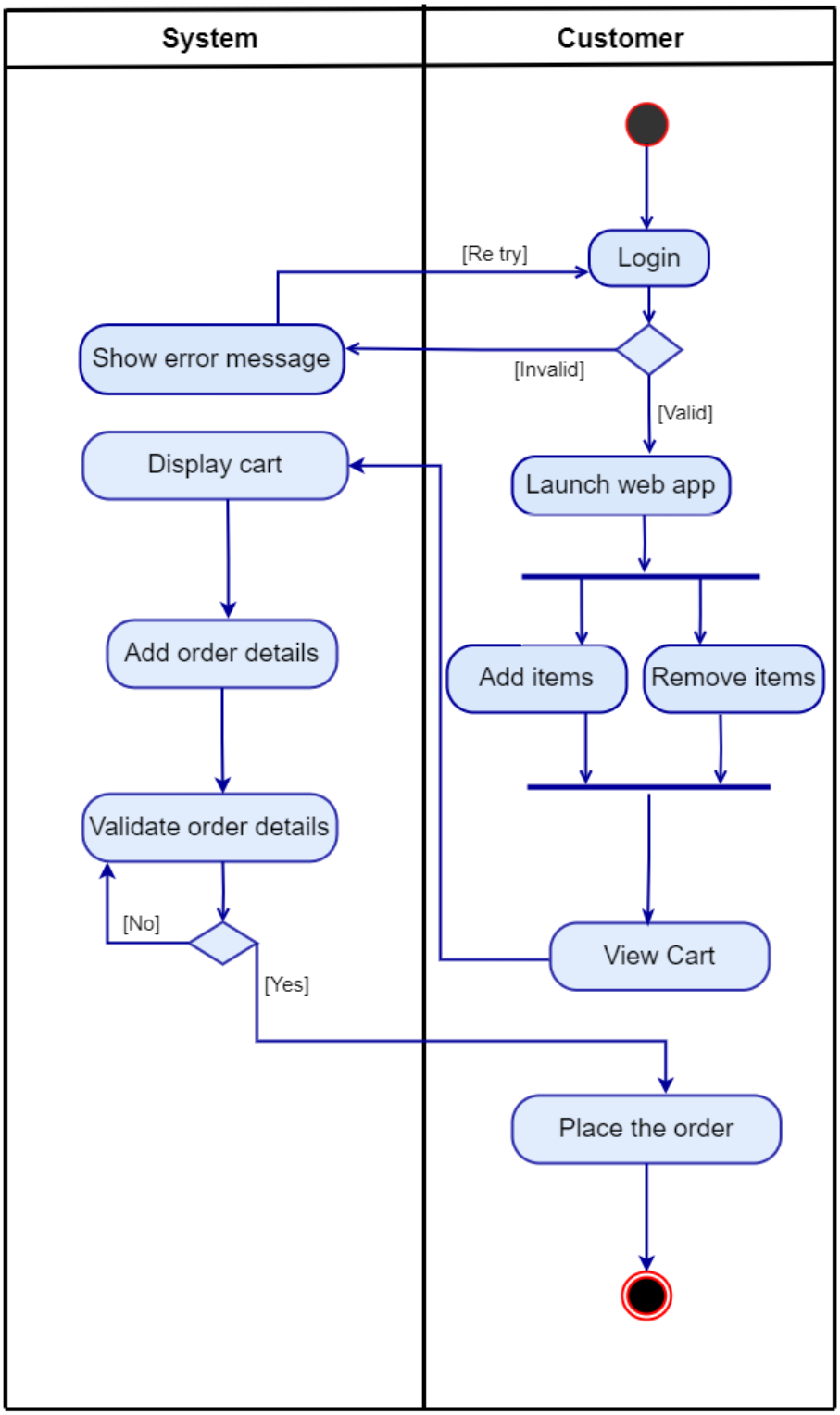



Figure 2: System Overview in Swim Lane diagram

## 3.3 Flow Charts

### 3.2.1 Login to Swag Labs

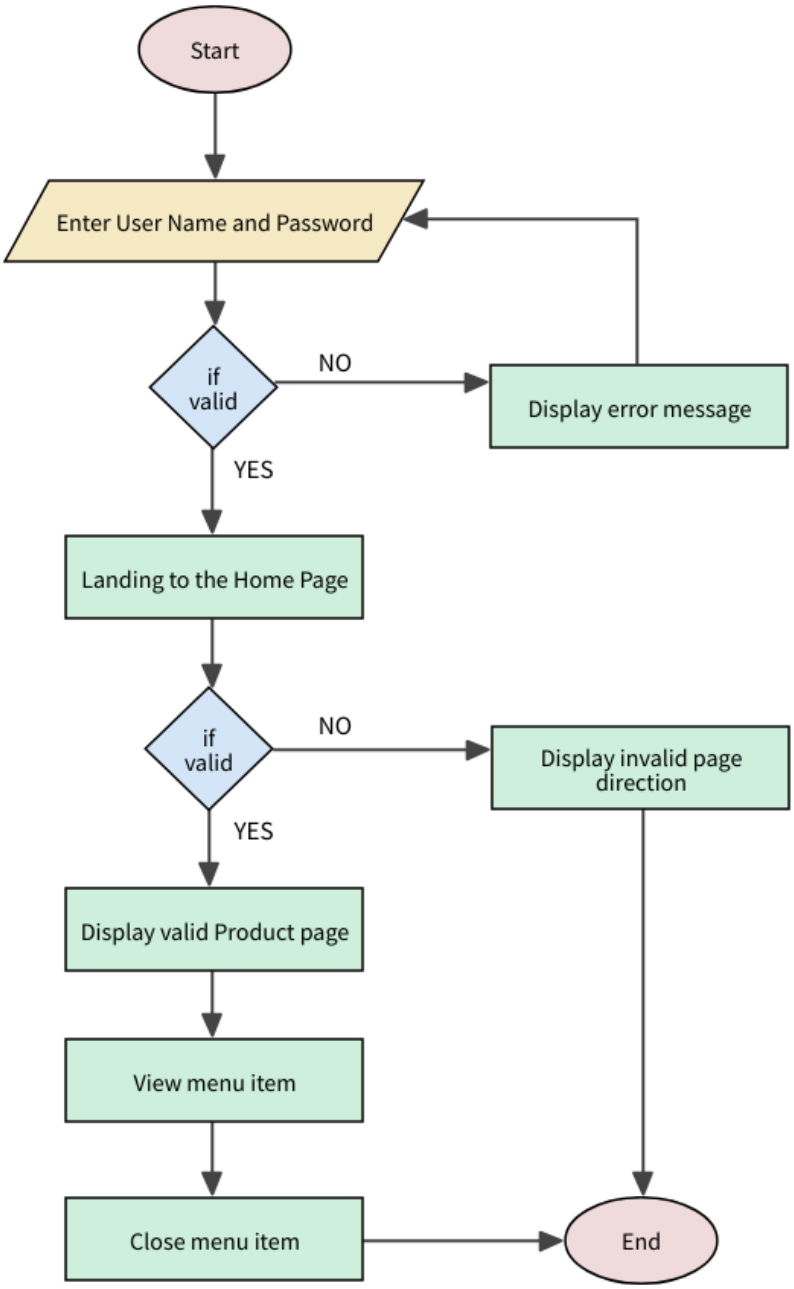



Figure 3: Login to Swag labs

### 3.2.2 Add items and manage shopping cart

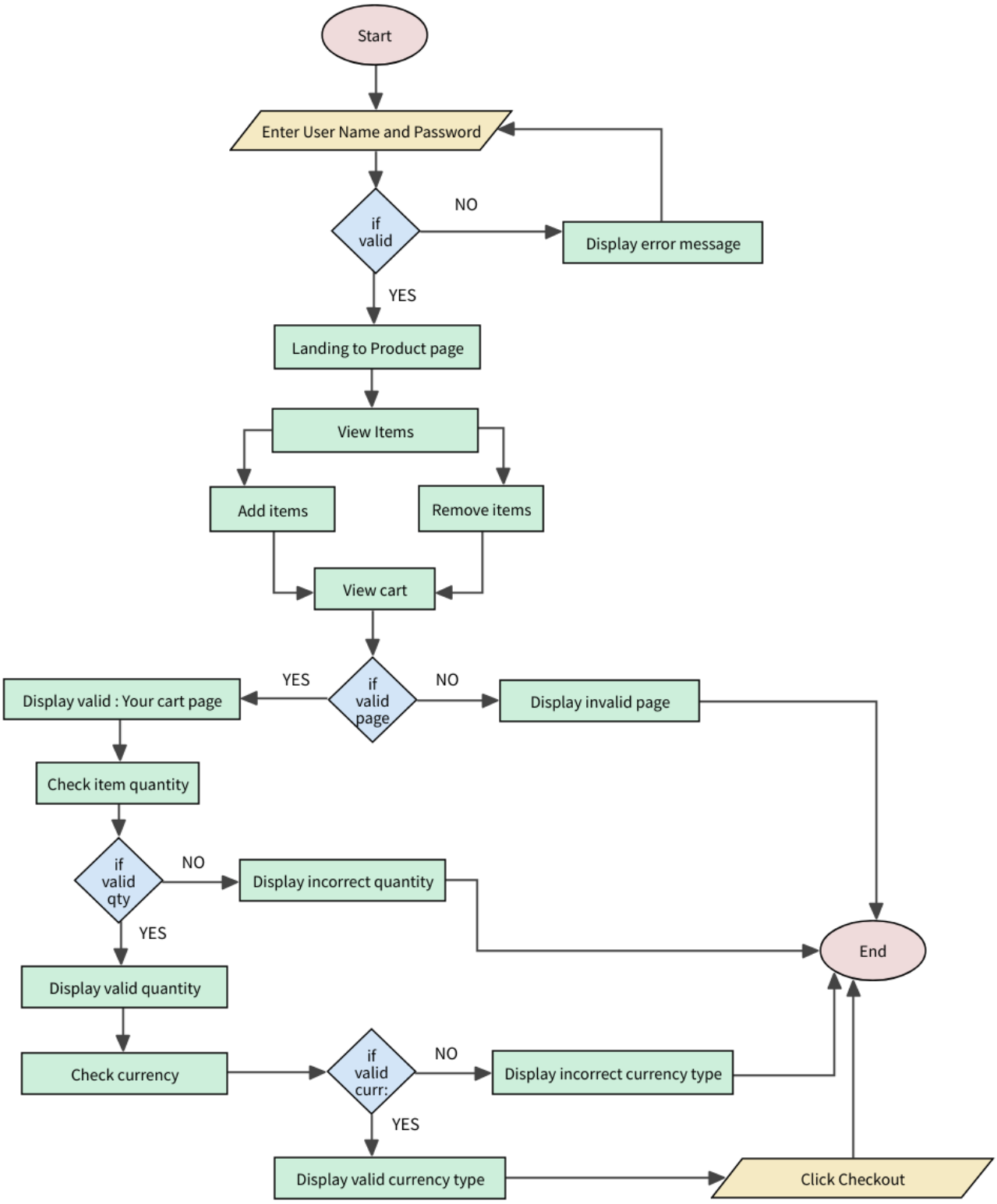



Figure 4: Add items and manage shopping cart

### 3.2.3 Add user details to place the order

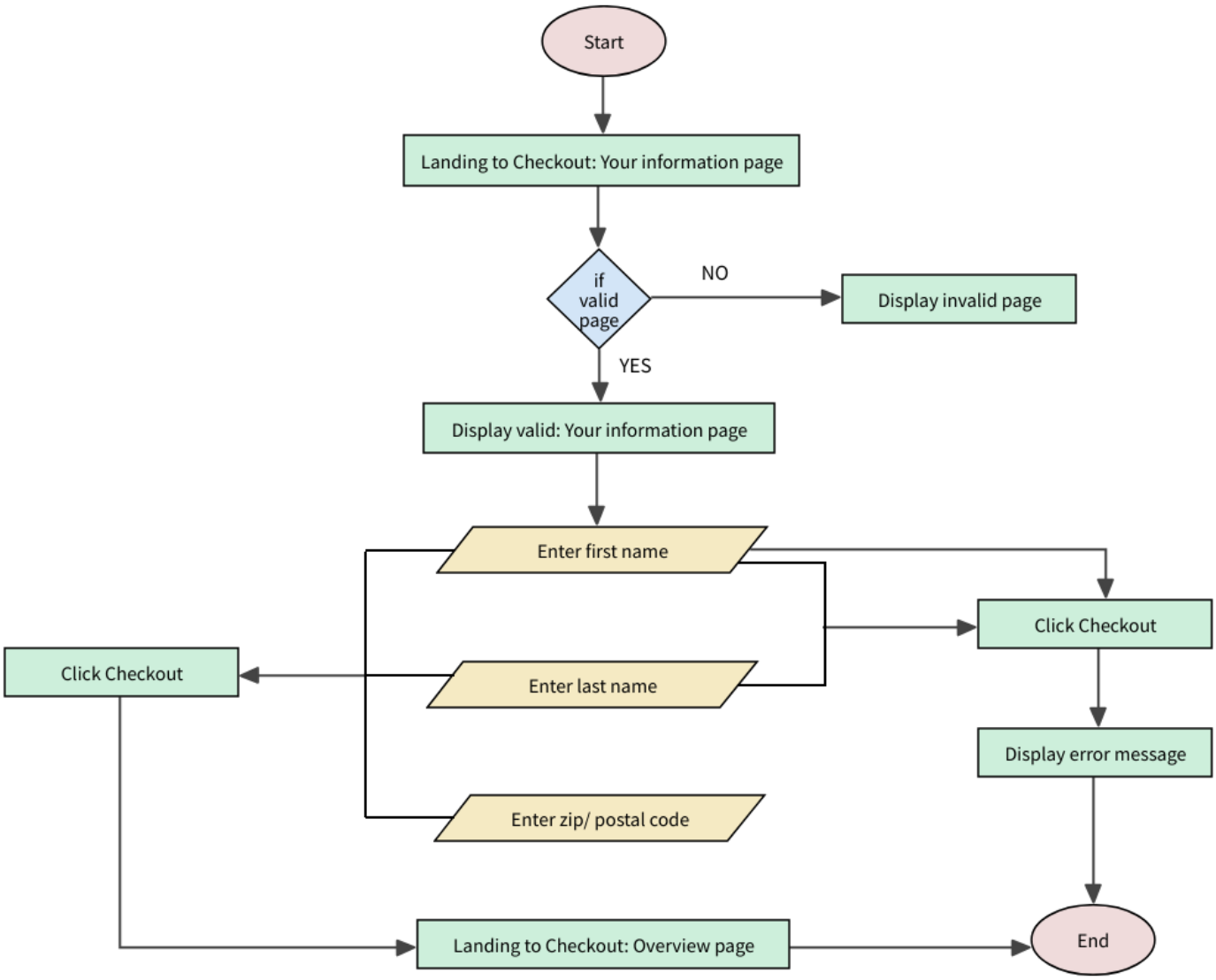



Figure 5: Add user details to place the order

### 3.2.4 Checkout and place the order

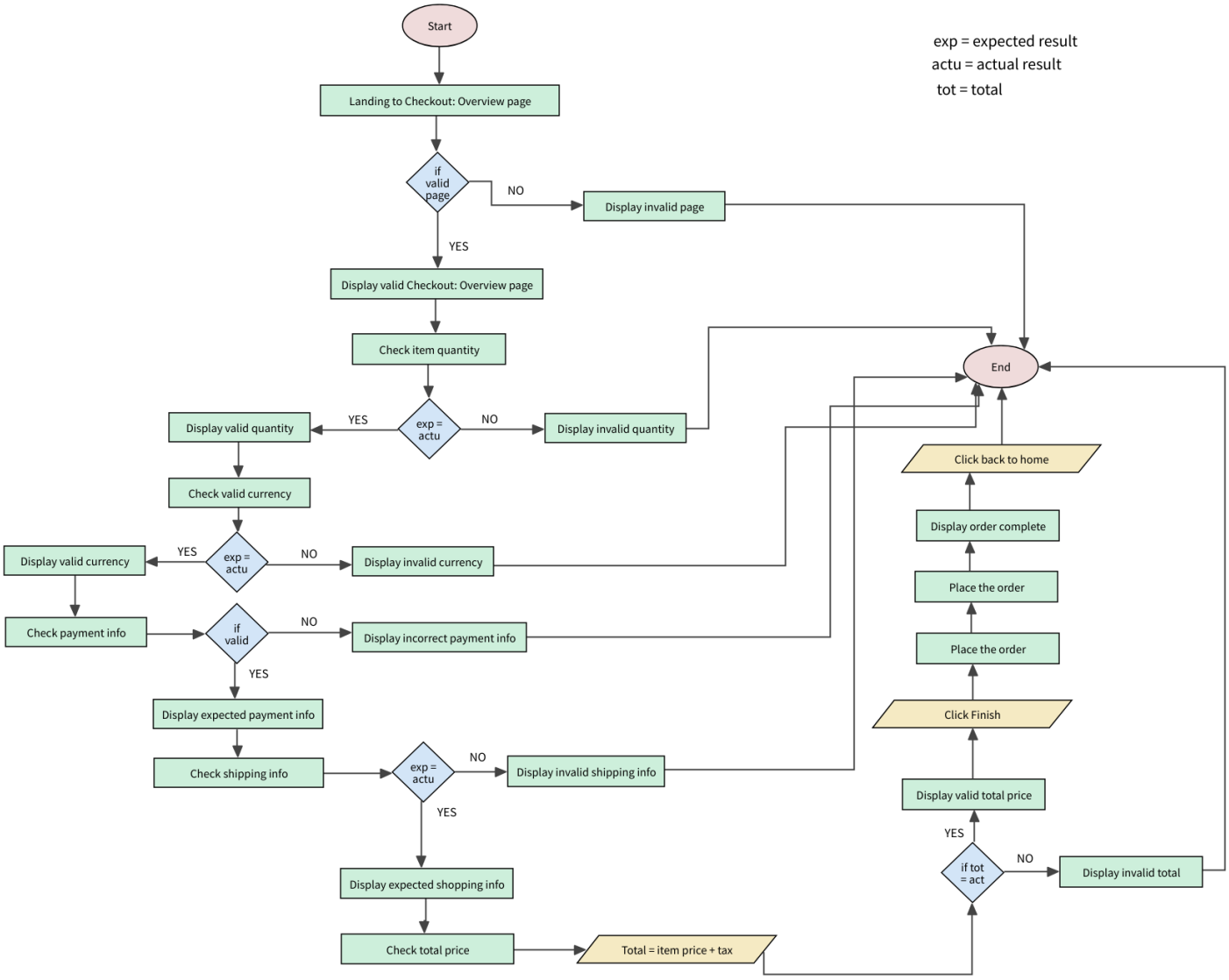



Figure 6: Checkout and place the order

# 4. Test Cases

## 4.1 Login to the system

| **Module:** Login to the system | |
|---|---|
| **Test Case No:** | TC001 |
| **Priority:** | High |
| **Test Case Description:** | Ensure that the customers are able to log in with valid credentials and that invalid credentials are handled appropriately. |
| **Sub-test cases:** | • Check user can login with a valid username and password<br>• Check user can login with an invalid username and password |
| **Pre-Condition:** | The application is accessible and the login page should be loaded. |
| **Test Steps:** | 1. Load the Swag Labs login page<br>2. Enter the user name and Password<br>3. Click on the login button |
| **Test Data:** | Valid user name: standard_user<br>Valid password: secret_sauce |
| **Expected Output:** | The user is able to login to the system successfully |
| **Actual Output:** | Successfully worked as expected |
| **Status (Pass / Fail) :** | Pass |
| **Test evidence:** | |

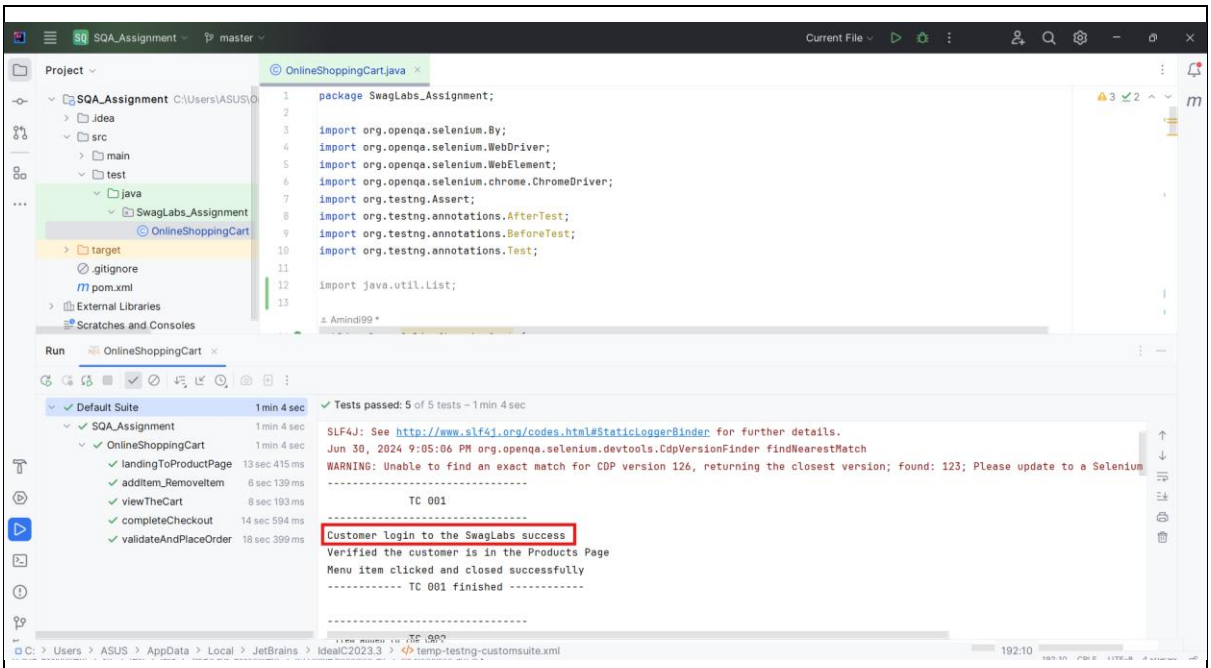



Figure 7: Login to the system test evidence

Table 3: Login to the system

## 4.2 Landing to the product page and open/ close the side navigation menu

| **Module:** Landing to the product page and open/ close the side navigation menu | |
|---|---|
| **Test Case No:** | TC002 |
| **Priority:** | Medium |
| **Test Case Description:** | Ensure that after login, the behavior of the landing page (Product page) |
| **Sub-test cases:** | • Check after login, the user is directed to the relevant page (product page)<br>• Check the side navigation menu opening<br>• Check the side navigation menu closing |
| **Pre-Condition:** | The user is able to login to the Swag Labs online store. |
| **Test Steps:** | 1. Landing on the Product page<br>2. Click on the side navigation menu<br>3. Click on the close icon in the side navigation menu |
| **Test Data:** | - |
| **Expected Output:** | Landing on the product page, able to open and close the side navigation menu by simply clicking. |
| **Actual output:** | Successfully worked as expected |
| **Status (Pass / Fail) :** | Pass |
| **Test evidence:** | |

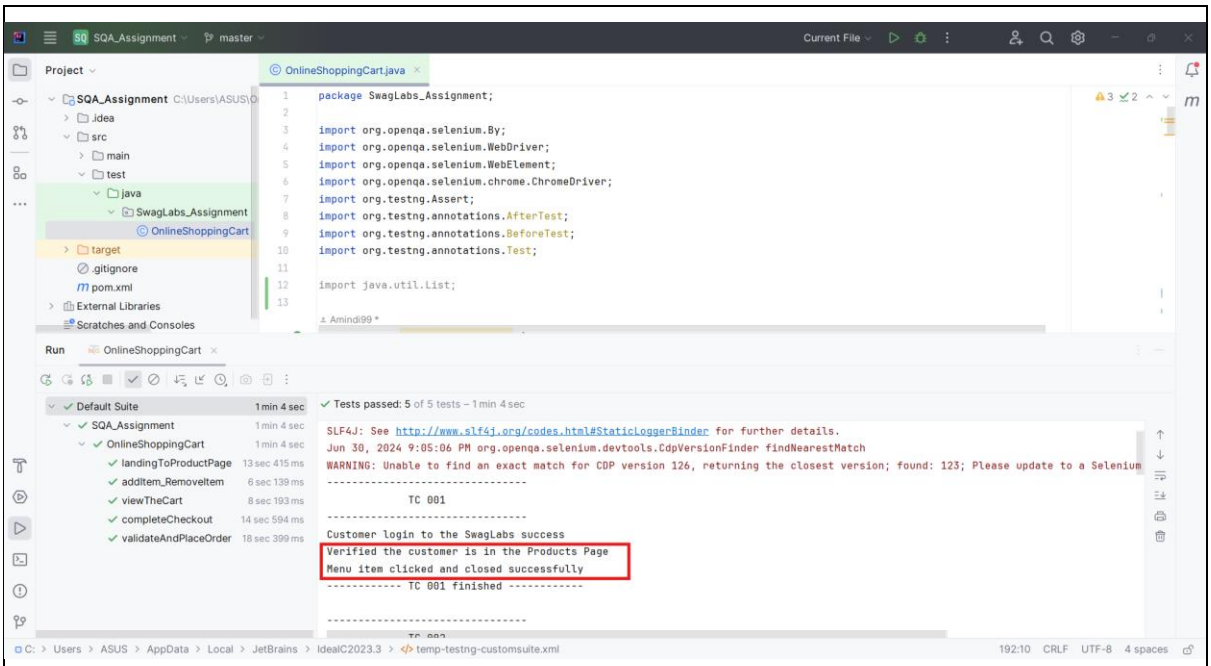



Figure 8: Landing to the product page and side navigation bar behavior test evidence

Table 4: Landing on the product page and open/ close the side navigation menu

## 4.3 Add/ remove items and manage the shopping cart

| **Module:** Add/ remove items and manage the shopping cart | |
|---|---|
| **Test Case No:** | TC003 |
| **Priority:** | High |
| **Test Case Description:** | Ensure that, customers are able to add items to the cart as well as remove items from the cart. |
| **Sub-test cases:** | • Check directing to the "Your cart" page<br>• Check users are able to add items to the cart<br>• Check users are able to remove items from the cart<br>• Check the view cart<br>• Check the added item count (Qty)<br>• Check the currency type<br>• Check the "Checkout" button clickable |
| **Pre-Condition:** | Navigating to the products page |
| **Test Steps:** | 1. Click on "Add to cart" button<br>2. Click on the "Remove" button on the same product<br>3. Again, click on the "Add to cart" button<br>4. Click on the cart icon in the right corner → Directs to Your cart page<br>5. View the item quantity<br>6. View the currency type<br>7. Click on the "Checkout" button |
| **Test Data:** | - |
| **Expected Output:** | Able to add and remove items from the cart and view the cart. And display the valid item quantity with the correct currency type. Finally, the user is directed to the checkout page. |
| **Actual output:** | Successfully worked as expected |
| **Status (Pass / Fail) :** | Pass |
| **Test evidence:** | |

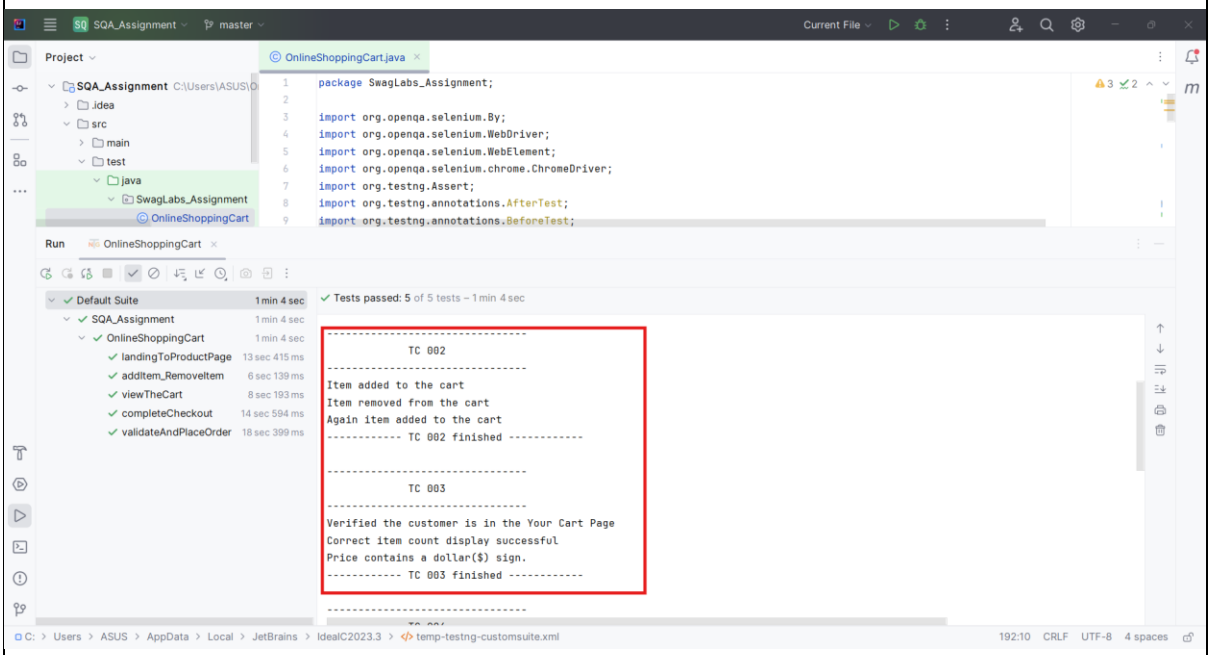



Figure 9: Add/ remove items and manage the shopping cart test evidence

Table 5: Add/ remove items and manage the shopping cart

## 4.4 Add user details to place the order

| **Module:** Add user details to place the order | |
|---|---|
| **Test Case No:** | TC004 |
| **Priority:** | High |
| **Test Case Description:** | Validate that, the customer able to add user details to place the order. |
| **Sub-test cases:** | • Check the first name field validation<br>• Check the last name field validation<br>• Check the zip/ postal code field validation<br>• Check the continue button validation and button clickable. |
| **Pre-Condition:** | Directing to the “Checkout: your information” page |
| **Test Steps:** | • Enter the first name<br>• Click on the “Continue” button<br>• Enter the last name<br>• Click on the “Continue” button<br>• Enter the zip/ postal code<br>• Click on the “Continue” button |
| **Test Data:** | First name: Amindi<br>Last name: Perera<br>Zip/ postal code: 11010 |
| **Expected Output:** | Validate the all fields and direct to the checkout page successfully. |
| **Actual output:** | Successfully worked as expected |
| **Status (Pass / Fail) :** | Pass |
| **Test evidence:** | |

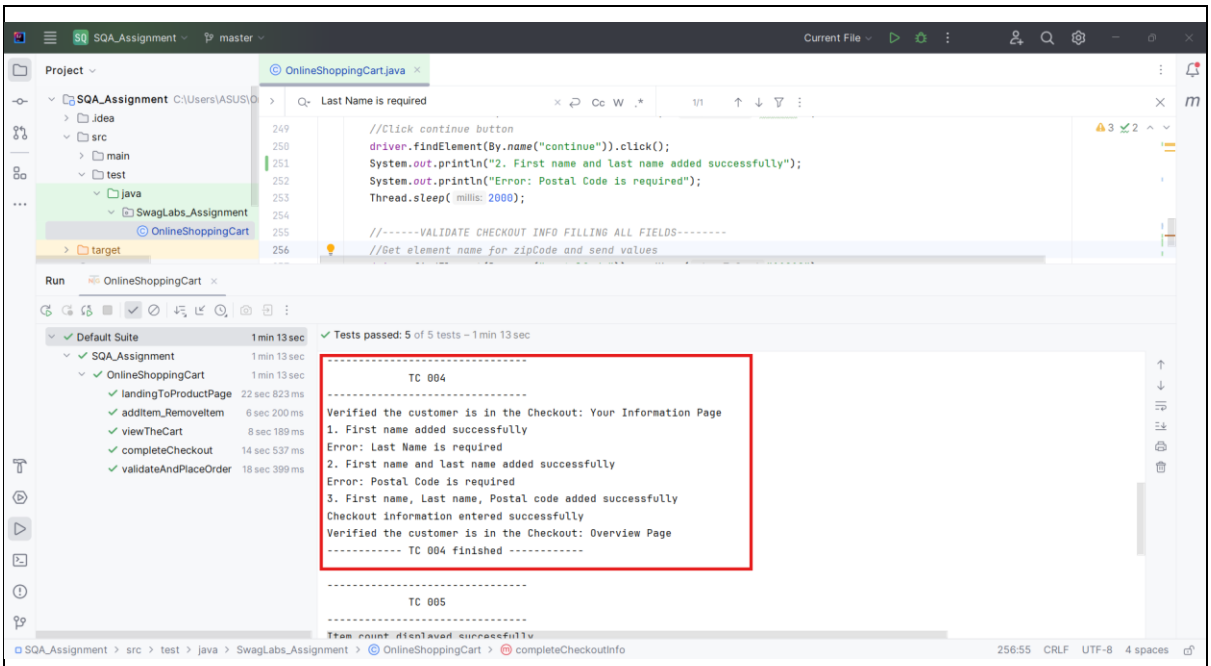



Figure 10: Add user details to place the order test evidence

Table 6: Add user details to place the order

## 4.4 Checkout and place the order

| **Module:** Checkout and place the order | |
|---|---|
| **Test Case No:** | TC005 |
| **Priority:** | High |
| **Test Case Description:** | Ensure that the user able to place an order |
| **Sub-test cases:** | • Verify that the user is on the Checkout: Overview page<br>• Check the no of items(Qty) in the cart<br>• Validate the currency type<br>• Check the payment info is correct<br>• Check the Shipping info is correct<br>• Check the total price = item total + tax<br>• Check the “Finish” button is clickable and directed to the payment success page<br>• Check the “Back home” button is clickable and navigate again to home page |
| **Pre-Condition:** | Directing to the “Checkout: Overview” page |
| **Test Steps:** | 1. Landing on “Checkout: Overview” page<br>2. Click the “Finish” button<br>3. Landing on the “Checkout: Complete!” page<br>4. Click on the “Back home” button |
| **Test Data:** | - |
| **Expected Output:** | The user able to successfully purchase the order and display “Thank you for your order!” |
| **Actual output:** | Successfully displayed as expected |
| **Status (Pass / Fail) :** | Pass |
| **Test evidence:** | |

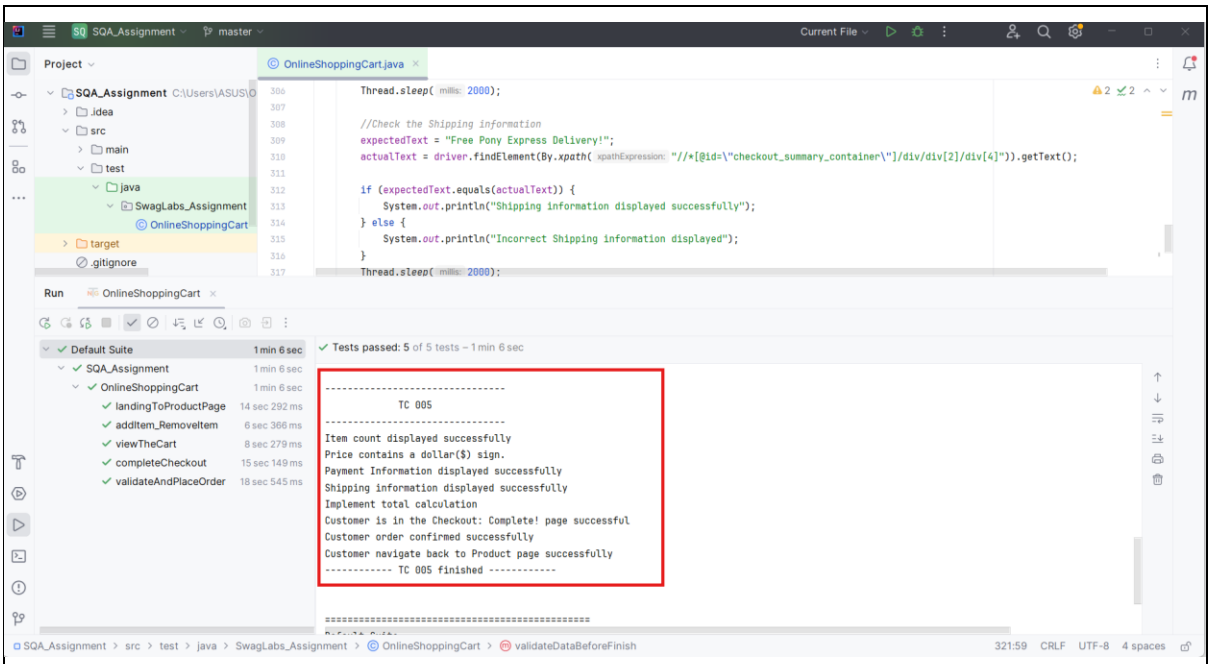



Figure 11: Checkout and place the order test evidence

Table 7: Checkout and place the order

# 5. Issues

| Issues | Mitigation |
|---|---|
| Identify the scenario properly | Refer to the lecture recording that describes the assignment. |
| Diagram drawing | Use an online tool (Neat and easy to identify) |
| Time management | Schedule the free time separately for codebase implementation and report creation. |
| Gettings errors while continuing the implementation | Refer to YouTube videos and lecture recordings |

Table 8: Issues

# 6. References


1. [1]Q. on Cloud and QAonCloud, "Test Automation ROI: How Do We Calculate It?," QAonCloud, Jun. 28, 2023. https://www.qaoncloud.com/blog/test-automation-roi-how-do-we-calculate-it (accessed Jun. 22, 2024).

2. [2]"Building Reliable Web Element Locators for Test Automation," Telerik Blogs, Jun. 15, 2022. https://www.telerik.com/blogs/building-reliable-web-element-locators-test-automation (accessed Jun. 22, 2024).

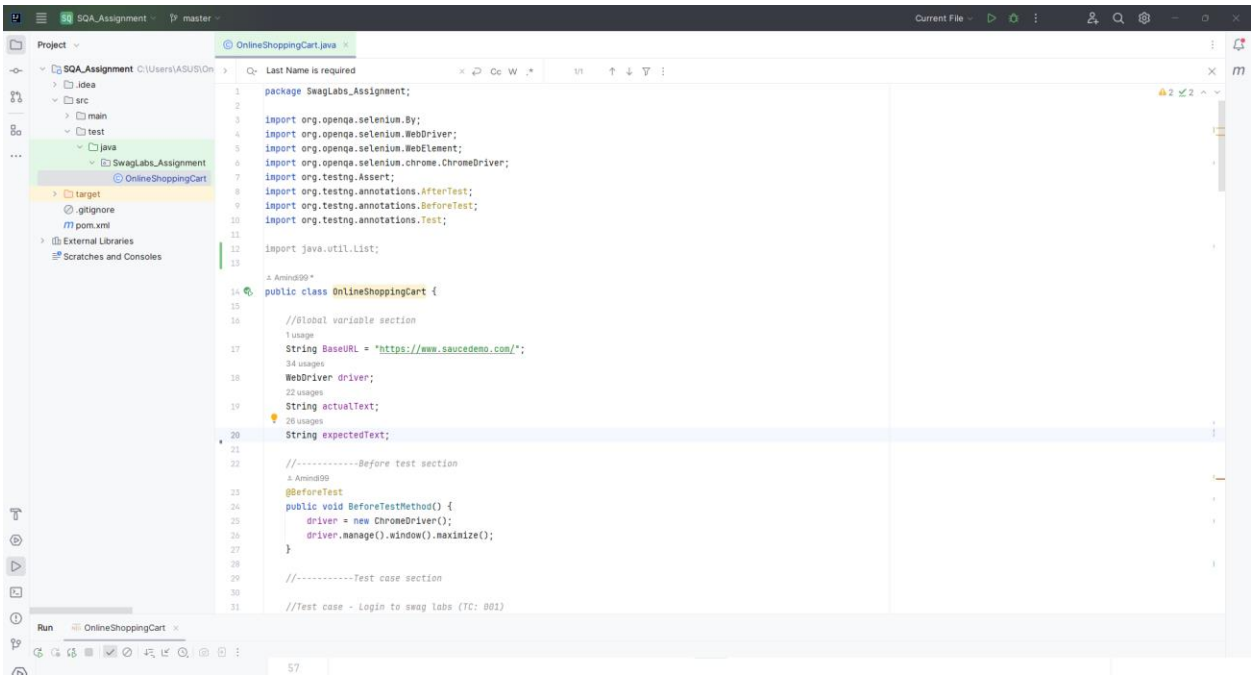
3. "Automation Testing Tutorial: Getting Started," BrowserStack. https://www.browserstack.com/guide/automation-testing-tutorial

4. "The Selenium Browser Automation Project," Selenium. https://www.selenium.dev/documentation/

5. [1]"Selenium Tutorial - javatpoint," www.javatpoint.com. https://www.javatpoint.com/selenium-tutorial

# 7. Appendix

## 7.1 Source code



```java
package SwagLabs_Assignment;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Test;

import java.util.List;

public class OnlineShoppingCart {

    //Global variable section
    String BaseURL = "https://www.saucedemo.com/";
    WebDriver driver;
    String actualText;
    String expectedText;

    //------------Before test section
    @BeforeTest
    public void BeforeTestMethod() {
        driver = new ChromeDriver();
        driver.manage().window().maximize();
    }

    //----------Test case section

    //Test case - Login to swag labs (TC: 001)
```

Figure 12: Global variables and before test section

### 7.1.1 Test case - Login to swag labs (TC: 001)

```java
//Test case - Login to swag labs (TC: 001)
Amindi99
@Test(priority = 1)
public void landingToProductPage() throws Exception {
    System.out.println("--------------------------------");
    System.out.println("             TC 001             ");
    System.out.println("--------------------------------");

    //Login to the system
    customerLogin();
    System.out.println("Customer login to the SwagLabs success");

    //Verify the landing page is product page
    expectedText = "Products";
    actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"header_container\"]/div[2]/span")).getText();

    if (expectedText.equals(actualText)) {
        System.out.println("Verified the customer is in the " + expectedText + " Page");
    } else {
        System.out.println("Unsuccessful landing page" + '\n');
    }

    //Click menu item
    menuItemClick();
    System.out.println("Menu item clicked and closed successfully");
    System.out.println("------------ TC 001 finished ------------" + '\n');
}
```

```java
//----------After test section
Amindi99
@AfterTest
public void AfterTestMethod() {}

//Supportive method section
1 usage  Amindi99
public void customerLogin() throws InterruptedException {

    //Calling the SwagLabs login page
    driver.get(BaseURL);
    Thread.sleep( millis: 2000);

    //Get element name for userName and send values
    driver.findElement(By.name("user-name")).sendKeys( ...keysToSend: "standard_user");
    Thread.sleep( millis: 2000);
    //Get element name for password and send values
    driver.findElement(By.name("password")).sendKeys( ...keysToSend: "secret_sauce");
    Thread.sleep( millis: 2000);
    //Get element name for submitButton and send values
    driver.findElement(By.name("login-button")).click();
    Thread.sleep( millis: 2000);
}

1 usage  Amindi99
```

### 7.1.2 Test case - Add item to cart and remove the added item (TC: 002)

```java
//Test case - Login to swag labs (TC: 001)
Amindi99
@Test(priority = 1)
public void landingToProductPage() throws Exception {...}

//Test case - Add item to cart and remove the added item (TC: 002)
Amindi99
@Test(priority = 2)
public void addItem_RemoveItem() throws Exception {
    System.out.println("--------------------------------");
    System.out.println("             TC 002             ");
    System.out.println("--------------------------------");

    //Add item to cart and remove the added item
    addToCart_RemoveClick();
    System.out.println("------------ TC 002 finished ------------" + '\n');

}

//Test case - View shopping cart (TC: 003)
Amindi99
@Test(priority = 3)
public void viewTheCart() throws Exception {...}

//Test case - Complete checkout process (TC: 004)
Amindi99
@Test(priority = 4)
```

```java
    //Close the menu item
    driver.findElement(By.xpath( xpathExpression: "//*[@id=\"react-burger-cross-btn\"]")).click();
    Thread.sleep( millis: 2000);
}

1 usage  Amindi99
public void addToCart_RemoveClick() throws InterruptedException {

    // Add item to cart
    driver.findElement(By.xpath( xpathExpression: "//*[@id=\"add-to-cart-sauce-labs-backpack\"]")).click();
    Thread.sleep( millis: 2000);
    System.out.println("Item added to the cart");

    // Remove item from the cart
    driver.findElement(By.xpath( xpathExpression: "//*[@id=\"remove-sauce-labs-backpack\"]")).click();
    Thread.sleep( millis: 2000);
    System.out.println("Item removed from the cart");

    // Again add item to cart
    driver.findElement(By.xpath( xpathExpression: "//*[@id=\"add-to-cart-sauce-labs-backpack\"]")).click();
    Thread.sleep( millis: 2000);
    System.out.println("Again item added to the cart");
}

1 usage  Amindi99 *
public void shoppingCartClick() throws InterruptedException {
```

### 7.1.3 Test case - View shopping cart (TC: 003)

```java
@Test(priority = 2)
public void addItem_RemoveItem() throws Exception {...}

//Test case - View shopping cart (TC: 003)
@Test(priority = 3)
public void viewTheCart() throws Exception {
    System.out.println("--------------------------------");
    System.out.println("             TC 003             ");
    System.out.println("--------------------------------");

    //View the shopping cart
    shoppingCartClick();
    System.out.println("------------ TC 003 finished ------------" + '\n');

}

//Test case - Complete checkout process (TC: 004)
@Test(priority = 4)
public void completeCheckout() throws Exception {...}

//Test case - Validate order details (TC: 005)
@Test(priority = 5)
public void validateAndPlaceOrder() throws Exception {...}
```

```java
public void shoppingCartClick() throws InterruptedException {

    //Click on shopping cart icon
    driver.findElement(By.xpath("//*[@id=\"shopping_cart_container\"]/a")).click();
    Thread.sleep(2000);

    //Verify that navigate to correct Your cart page
    expectedText = "Your Cart";
    actualText = driver.findElement(By.xpath("//*[@id=\"header_container\"]/div[2]/span")).getText();

    if (expectedText.equals(actualText)) {
        System.out.println("Verified the customer is in the " + expectedText + " Page");
    } else {
        System.out.println("Unsuccessful page navigation");
    }

    //Check the selected item count
    expectedText = "1";
    actualText = driver.findElement(By.xpath("//*[@id=\"cart_contents_container\"]/div/div[1]/div[3]/div[1]")).getText();

    if (expectedText.equals(actualText)) {
        System.out.println("Correct item count display successful");
    } else {
        System.out.println("Item count display unsuccessful");
    }
    Thread.sleep(3000);

    //Check the currency element
    // Locate the price element
    WebElement priceElement = driver.findElement(By.xpath("//*[@id=\"cart_contents_container\"]/div/div[1]/div[3]/div[2]/div[2]/div"));

    String priceText = priceElement.getText();

    // Check the price contains $ mark
    Assert.assertTrue(priceText.contains("$"), "Price does not contain a dollar sign: " + priceText);

    System.out.println("Price contains a dollar($) sign.");
    Thread.sleep(3000);
}
```

### 7.1.4 Test case - Complete checkout process (TC: 004)

```
//Test case - Complete checkout process (TC: 004)
Amindi99
@Test(priority = 4)
public void completeCheckout() throws Exception {
    System.out.println("--------------------------------");
    System.out.println("             TC 004             ");
    System.out.println("--------------------------------");

    //Checkout
    checkoutButtonClick();
    Thread.sleep( millis: 2000);

    //Fill checkout information
    completeCheckoutInfo();
    Thread.sleep( millis: 2000);
    System.out.println("------------ TC 004 finished ------------" + '\n');

}
```

```
public void completeCheckoutInfo() throws InterruptedException {

    //------VALIDATE CHECKOUT INFO ONLY FILLING FIRST NAME----------
    //Get element name for FirstName and send values
    driver.findElement(By.name("firstName")).sendKeys( ...keysToSend: "Amindi");
    //Click continue button
    driver.findElement(By.name("continue")).click();
    System.out.println("1. First name added successfully");
    System.out.println("Error: Last Name is required");
    Thread.sleep( millis: 2000);

    //------VALIDATE CHECKOUT INFO ONLY FILLING FIRST NAME, LAST NAME--------
    //Get element name for LastName and send values
    driver.findElement(By.name("lastName")).sendKeys( ...keysToSend: "Perera");
    //Click continue button
    driver.findElement(By.name("continue")).click();
    System.out.println("2. First name and last name added successfully");
    System.out.println("Error: Postal Code is required");
    Thread.sleep( millis: 2000);

    //------VALIDATE CHECKOUT INFO FILLING ALL FIELDS--------
    //Get element name for zipCode and send values
    driver.findElement(By.name("postalCode")).sendKeys( ...keysToSend: "11010");
    //Click continue button
    driver.findElement(By.name("continue")).click();
    System.out.println("3. First name, Last name, Postal code added successfully");
    System.out.println("Checkout information entered successfully");
    Thread.sleep( millis: 2000);

    //Verify that navigate to Checkout: Overview page
```

```
        System.out.println("Checkout information entered successfully");
        Thread.sleep( millis: 2000);

        //Verify that navigate to Checkout: Overview page
        expectedText = "Checkout: Overview";
        actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"header_container\"]/div[2]/span")).getText();

        if (expectedText.equals(actualText)) {
            System.out.println("Verified the customer is in the " + expectedText + " Page");
        } else {
            System.out.println("Unsuccessful page navigation");
        }
        Thread.sleep( millis: 2000);
    }

    1 usage  Amindi99
    public void validateDataBeforeFinish() throws InterruptedException {...}
        //----------------------------------------------------------------------------------------------------

    1 usage  Amindi99
    public void finishButtonClick() throws InterruptedException {...}

    1 usage  Amindi99
    public void backToHomeButtonClick() throws InterruptedException {...}
}
```

### 7.1.5 Test case - Validate order details (TC: 005)

```
    Amindi99
    @Test(priority = 4)
    public void completeCheckout() throws Exception {...}

    //Test case - Validate order details (TC: 005)
    Amindi99 *
    @Test(priority = 5)
    public void validateAndPlaceOrder() throws Exception {
        System.out.println("--------------------------------");
        System.out.println("             TC 005             ");
        System.out.println("--------------------------------");

        //Validate details
        validateDataBeforeFinish();
        Thread.sleep( millis: 2000);

        //Finish data validation
        finishButtonClick();
        Thread.sleep( millis: 2000);

        //Back to home page
        backToHomeButtonClick();
        Thread.sleep( millis: 2000);
        System.out.println("------------ TC 005 finished ------------" + '\n');
    }

    //----------After test section
    Amindi99
    @AfterTest
    public void AfterTestMethod() {}
```

```java
public void validateDataBeforeFinish() throws InterruptedException {

    //Check the selected item count
    expectedText = "1";
    actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"checkout_summary_container\"]/div/div[1]/div[3]/div[1]")).getText();

    if (expectedText.equals(actualText)) {
        System.out.println("Item count displayed successfully");
    } else {
        System.out.println("Incorrect item count displayed");
    }
    Thread.sleep( millis: 2000);

    //Check the currency element
    WebElement priceElement = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"checkout_summary_container\"]/div/div[1]/div[3]/div[2]/div[2]/div
    // Get the text of the price element
    String priceText = priceElement.getText();
    // Verify that the price contains a dollar sign
    Assert.assertTrue(priceText.contains("$"), message: "Price does not contain a dollar sign: " + priceText);
    System.out.println("Price contains a dollar($) sign.");
    Thread.sleep( millis: 2000);

    //Check the payment information
    expectedText = "SauceCard #31337";
    actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"checkout_summary_container\"]/div/div[2]/div[2]")).getText();

    if (expectedText.equals(actualText)) {
        System.out.println("Payment Information displayed successfully");
    } else {
        System.out.println("Incorrect Payment Information displayed");
    }
```

```java
        System.out.println("Payment Information displayed successfully");
    } else {
        System.out.println("Incorrect Payment Information displayed");
    }
    Thread.sleep( millis: 2000);

    //Check the Shipping information
    expectedText = "Free Pony Express Delivery!";
    actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"checkout_summary_container\"]/div/div[2]/div[4]")).getText();

    if (expectedText.equals(actualText)) {
        System.out.println("Shipping information displayed successfully");
    } else {
        System.out.println("Incorrect Shipping information displayed");
    }
    Thread.sleep( millis: 2000);

    //-----------------------------------------------------------------------------------------
    //Verify total amount calculation
    System.out.println("Implement total calculation");
}
    //-----------------------------------------------------------------------------------------
```

```
Build Output
SQA_Assignment: build finished UP-TO-DATE at 7 2 sec, 127 ms
Executing pre-compile tasks…
Executing post-compile tasks…
```

```java
        //-----------------------------------------------------------------------------------------------
    1 usage   Amindi99
    public void finishButtonClick() throws InterruptedException {

        //click the finish button
        driver.findElement(By.xpath( xpathExpression: "//*[@id=\"finish\"]")).click();
        Thread.sleep( millis: 2000);

        //Verify that navigate to Checkout: Complete! page
        expectedText = "Checkout: Complete!";
        actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"header_container\"]/div[2]/span")).getText();

        if (expectedText.equals(actualText)) {
            System.out.println("Customer is in the Checkout: Complete! page successful");
        } else {
            System.out.println("Unsuccessful page navigation");
        }

        //Verify that navigate to Checkout: Complete! page
        expectedText = "Thank you for your order!";
        actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"checkout_complete_container\"]/h2")).getText();

        if (expectedText.equals(actualText)) {
            System.out.println("Customer order confirmed successfully");
        } else {
            System.out.println("Customer order is unsuccessful");
        }
    }
```

```java
    }

    1 usage   Amindi99
    public void backToHomeButtonClick() throws InterruptedException {
        //click the back to home button
        driver.findElement(By.xpath( xpathExpression: " //*[@id=\"back-to-products\"]")).click();

        //Verify direct back to product page
        expectedText = "Products";
        actualText = driver.findElement(By.xpath( xpathExpression: "//*[@id=\"header_container\"]/div[2]/span")).getText();

        if (expectedText.equals(actualText)) {
            System.out.println("Customer navigate back to Product page successfully");
        } else {
            System.out.println("Unsuccessful navigate to Product page");
        }
        Thread.sleep( millis: 2000);
    }
}
```