Capstone Project

Project Name: Automation Testing for **bstackdemo.com**

Domain: E-Commerce Application

Submitted By: Ritik Prajapat [StarAgile Student Trainee]

Submitted To: [StarAgile Consulting Pvt Ltd]

GitHub link:

Table of Contents

- Project Overview
- Objectives
- Scope of the Project
- Tools &Technologies Used
- Framework Architecture
- Test Case Scenarios
- Sample Code Structure
- Testcase Design &Execution
- Test Result Analysis & Reporting
- Conclusion

1. Project Overview

bstackdemo.com is a sample e-commerce web application developed by BrowserStack, a leading cloud-based testing platform. It serves as a demonstration tool for automated cross-browser and cross-platform testing. This application simulates a real-world online shopping experience and provides the core functionalities of a modern e-commerce site, including features such as user login, product catalogue, filtering, adding items to the cart, and order placement.

This project focuses on automating the testing of the **bstackdemo E-commerce** web application. To ensure the application's functionality, reliability, and performance across critical user journeys. The automation testing suite is developed using Java, Selenium WebDriver, and TestNG, following best practices such as the Page Object Model (POM) design pattern.

2. Objectives

- To build a modular, reusable, and scalable test automation framework using Java, Selenium WebDriver, and TestNG & Page Object Model (POM).
- Validate key functionalities like user login, product search, cart operations, checkout process, and order confirmation.
- Implement test cases for both positive and negative scenarios.
- To use Extent Reports for generating a real-time, user-friendly report.

3. Scope of the Project

Modules Covered:

- **Login:** Validate valid and invalid user login.
- **Product Listing:** Verify product details and filters
- **Add to Cart:** Verify items added to cart
- **Cart Verification:** Ensure cart shows correct quantity and price
- **Checkout:** Validate order placement functionality

4. Tools &Technologies Used

- **Automation Framework:** Selenium WebDriver
- **Programming Language:** Java
- **IDE:** IntelliJ IDEA / Eclipse
- **Build Tool**: Maven
- **Unit Testing Framework:** TestNG
- **Version Control:** Git-GitHub

5. Framework Architecture

Design Pattern: Page Object Model (POM)

Components:

- Page Classes – LoginPage, ProductPage, CartPage, CheckoutPage
- Utilities – WebDriverFactory, WaitUtils
- Test Cases – LoginTest, AddToCartTest, CheckoutTest
- TestNG.xml – Test suite configuration
- Reports – ExtentReports integration

6. TestCase Scenarios

1. Login Tests

- TC_001: Login with valid credentials (demouser, testingisfun99)
- TC_002: Login with invalid credentials
- TC_003: Login with empty fields

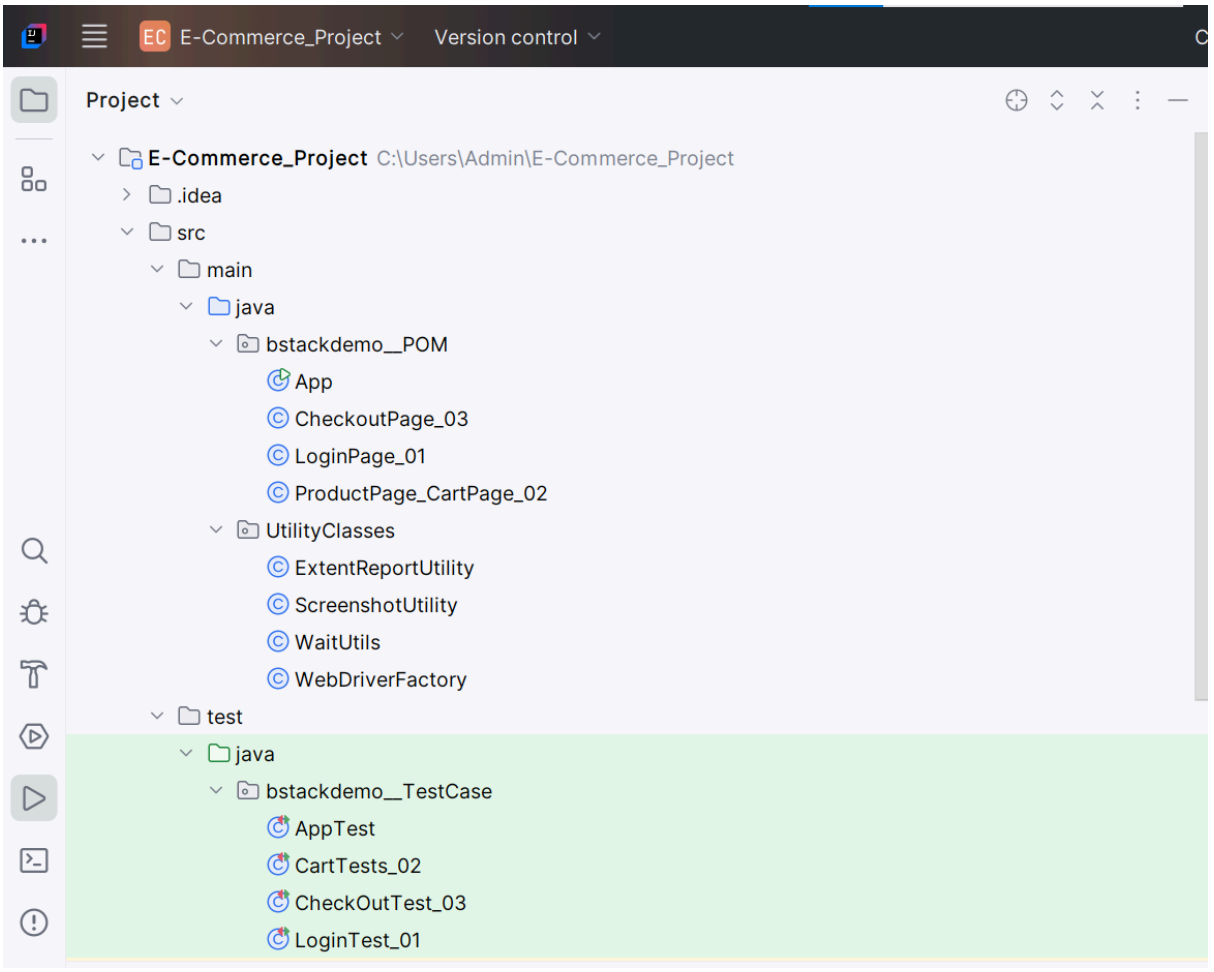
2. Cart Tests

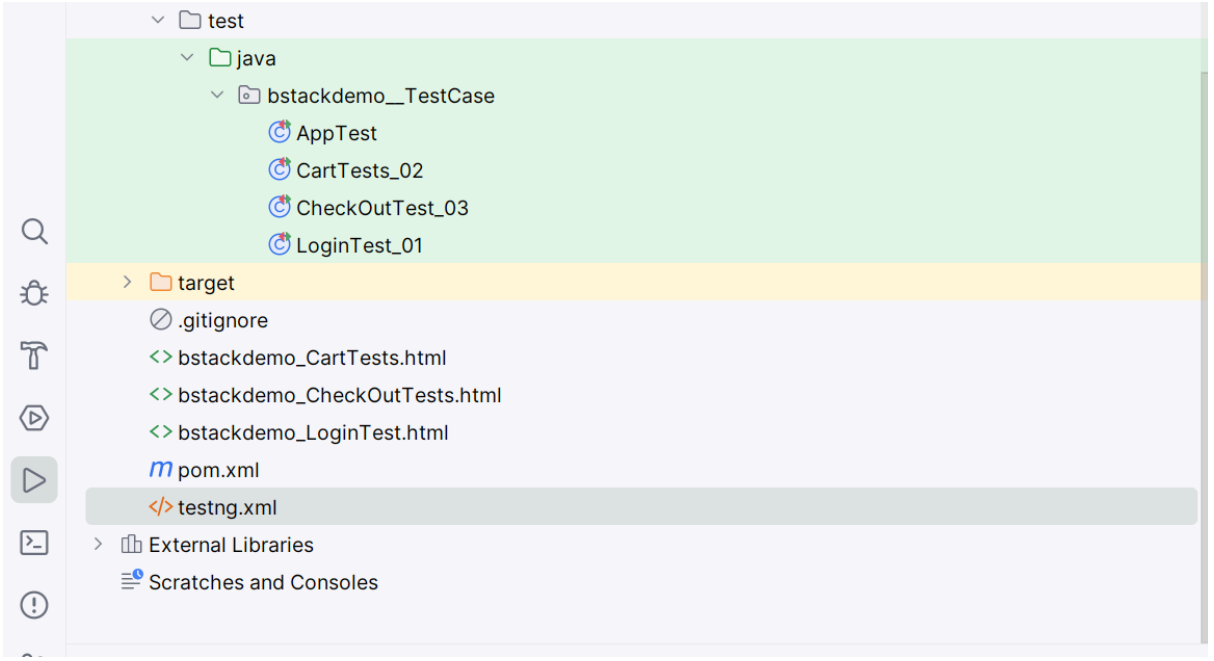
- TC_004: Add single item to cart
- TC_005: Add multiple items and verify cart count
- TC_006: Remove item from cart

3. Checkout Tests

- TC_007: Place order with valid details
- TC_008: Try checkout without adding items (negative)

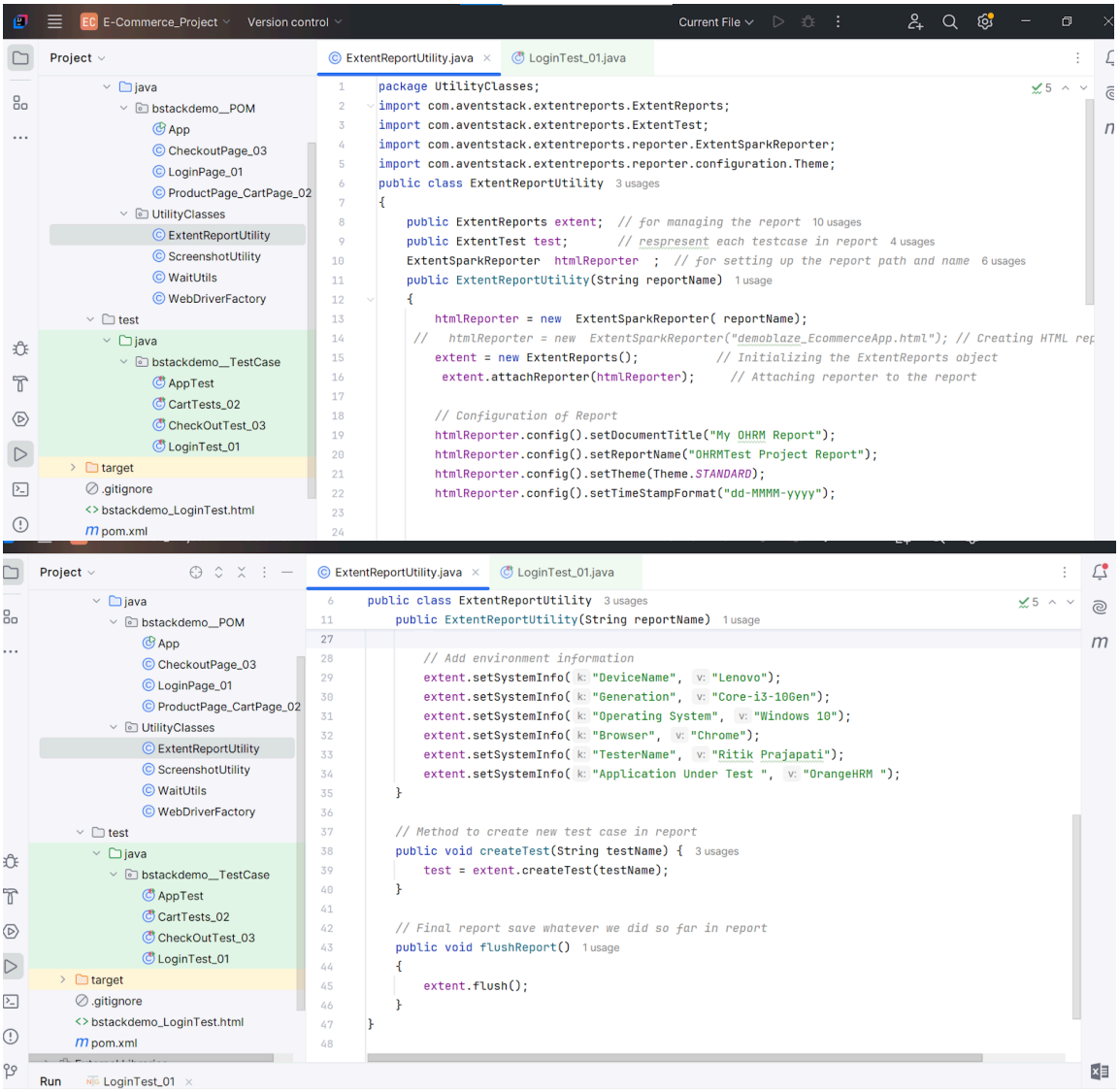
7. Sample Code Structure





8. Testcase Design & Execution

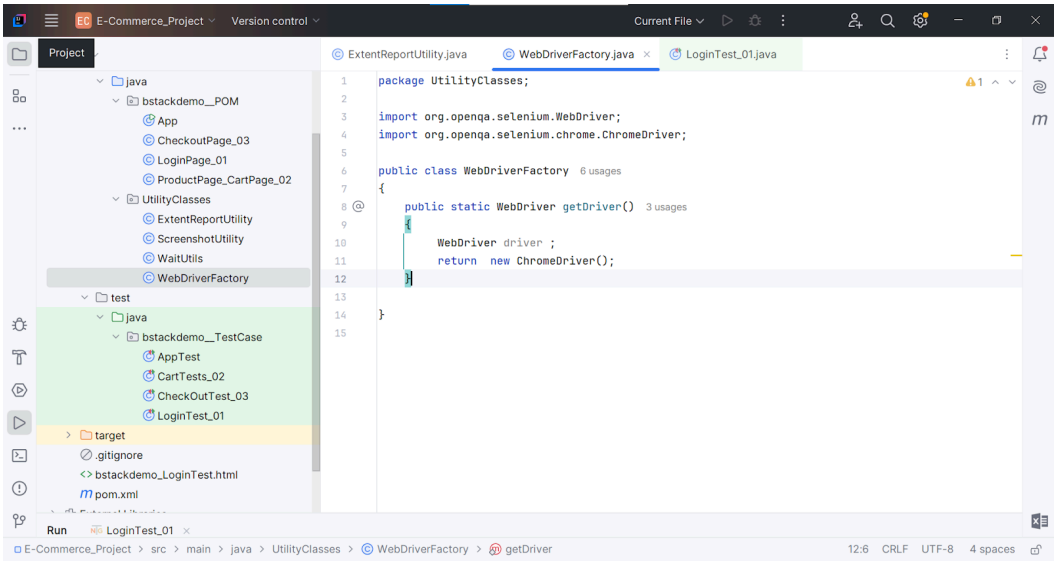
UtilityClass: ExtentReport



```java
package UtilityClasses;
import com.aventstack.extentreports.ExtentReports;
import com.aventstack.extentreports.ExtentTest;
import com.aventstack.extentreports.reporter.ExtentSparkReporter;
import com.aventstack.extentreports.reporter.configuration.Theme;
public class ExtentReportUtility  3 usages
{
    public ExtentReports extent;  // for managing the report  10 usages
    public ExtentTest test;       // respresent each testcase in report  4 usages
    ExtentSparkReporter  htmlReporter  ;  // for setting up the report path and name  6 usages
    public ExtentReportUtility(String reportName)  1 usage
    {
        htmlReporter = new  ExtentSparkReporter( reportName);
    //   htmlReporter = new  ExtentSparkReporter("demoblaze_EcommerceApp.html"); // Creating HTML rep
        extent = new ExtentReports();           // Initializing the ExtentReports object
         extent.attachReporter(htmlReporter);     // Attaching reporter to the report

        // Configuration of Report
        htmlReporter.config().setDocumentTitle("My OHRM Report");
        htmlReporter.config().setReportName("OHRMTest Project Report");
        htmlReporter.config().setTheme(Theme.STANDARD);
        htmlReporter.config().setTimeStampFormat("dd-MMMM-yyyy");
```

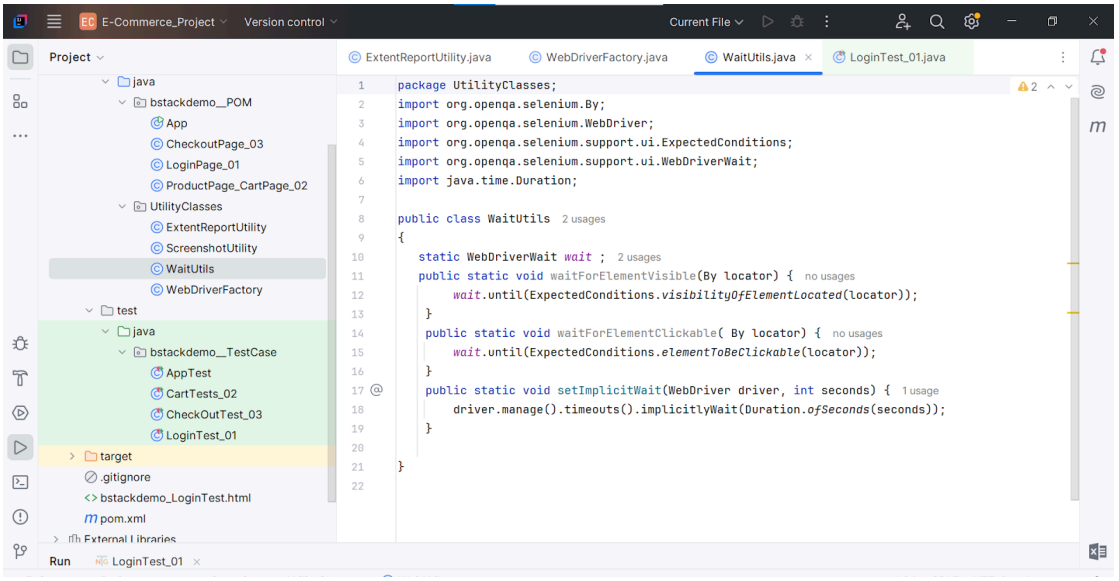
```java
public class ExtentReportUtility  3 usages
    public ExtentReportUtility(String reportName)  1 usage

        // Add environment information
        extent.setSystemInfo( k: "DeviceName", v: "Lenovo");
        extent.setSystemInfo( k: "Generation", v: "Core-i3-10Gen");
        extent.setSystemInfo( k: "Operating System", v: "Windows 10");
        extent.setSystemInfo( k: "Browser", v: "Chrome");
        extent.setSystemInfo( k: "TesterName", v: "Ritik Prajapati");
        extent.setSystemInfo( k: "Application Under Test ", v: "OrangeHRM ");
    }

    // Method to create new test case in report
    public void createTest(String testName) {  3 usages
        test = extent.createTest(testName);
    }

    // Final report save whatever we did so far in report
    public void flushReport()  1 usage
    {
        extent.flush();
    }
}
```

UtilityClass: WebDriverFactory



```java
package UtilityClasses;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class WebDriverFactory  6 usages
{
    public static WebDriver getDriver()  3 usages
    {
        WebDriver driver ;
        return  new ChromeDriver();
    }

}
```

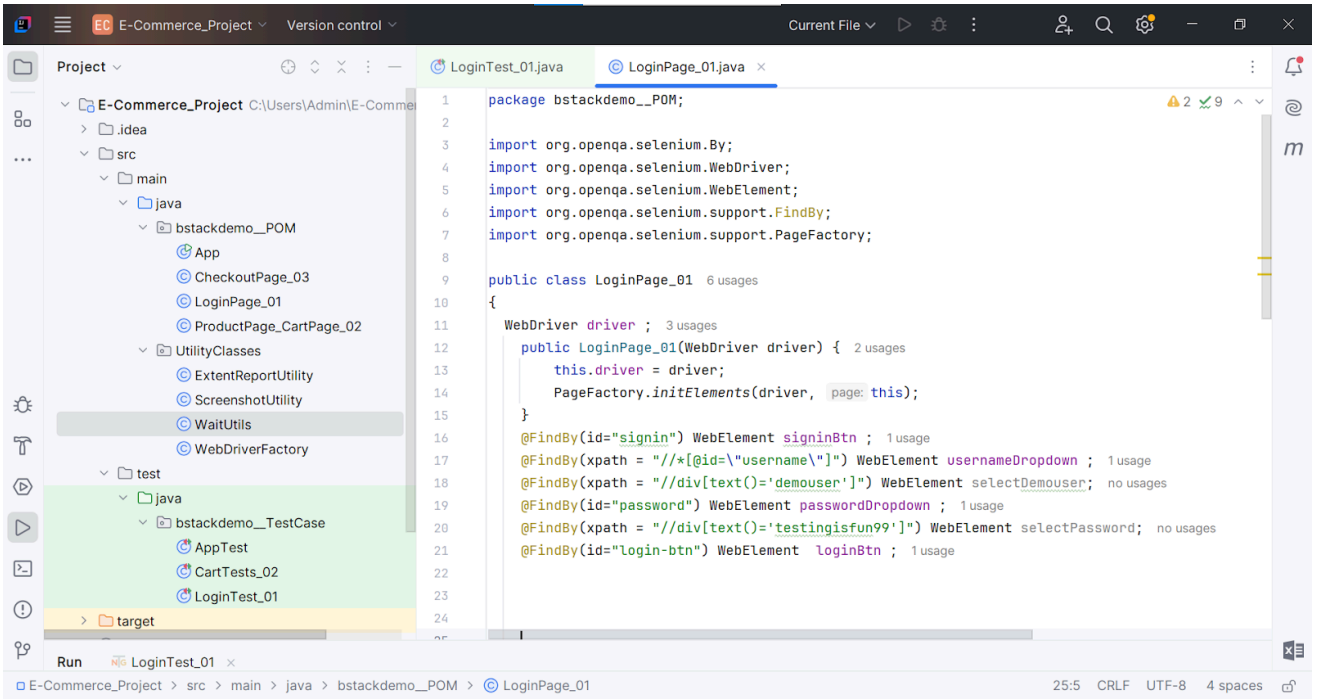
UtilityClass: WaitsUtils

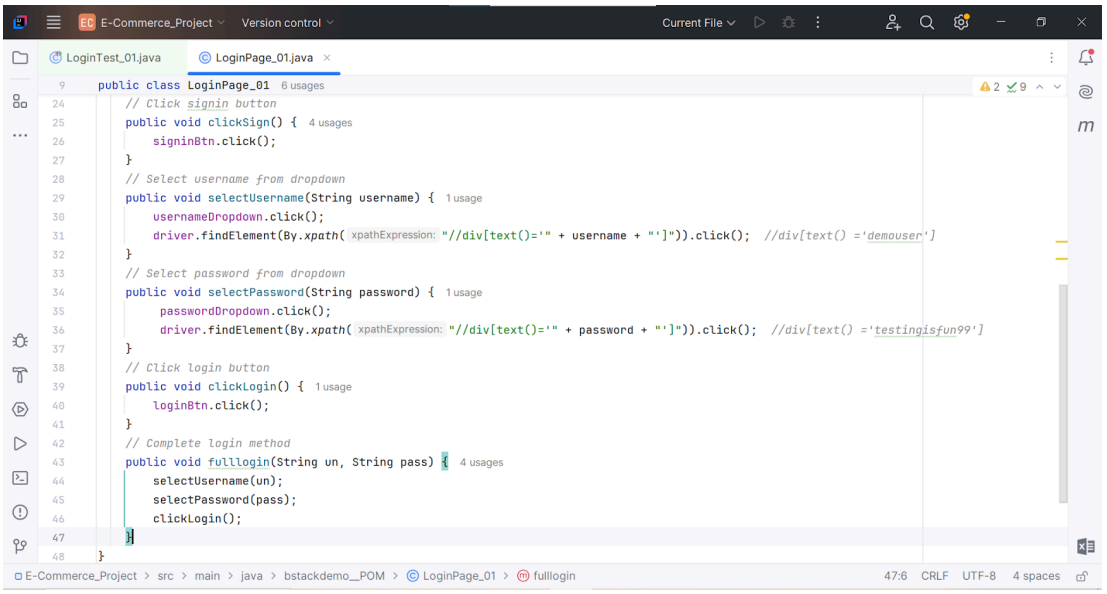


1. Login Tests

- TC_001: Login with valid credentials (demouser, testingisfun99)
- TC_002: Login with invalid credentials
- TC_003: Login with empty fields

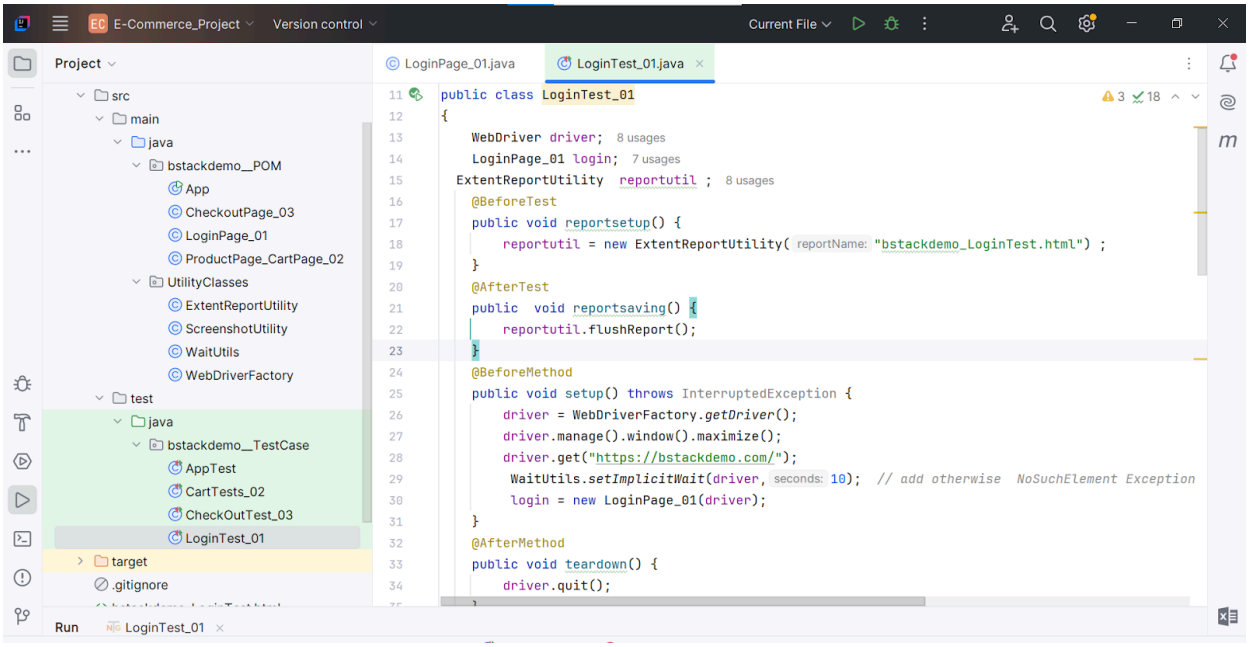
LoginPage: POM



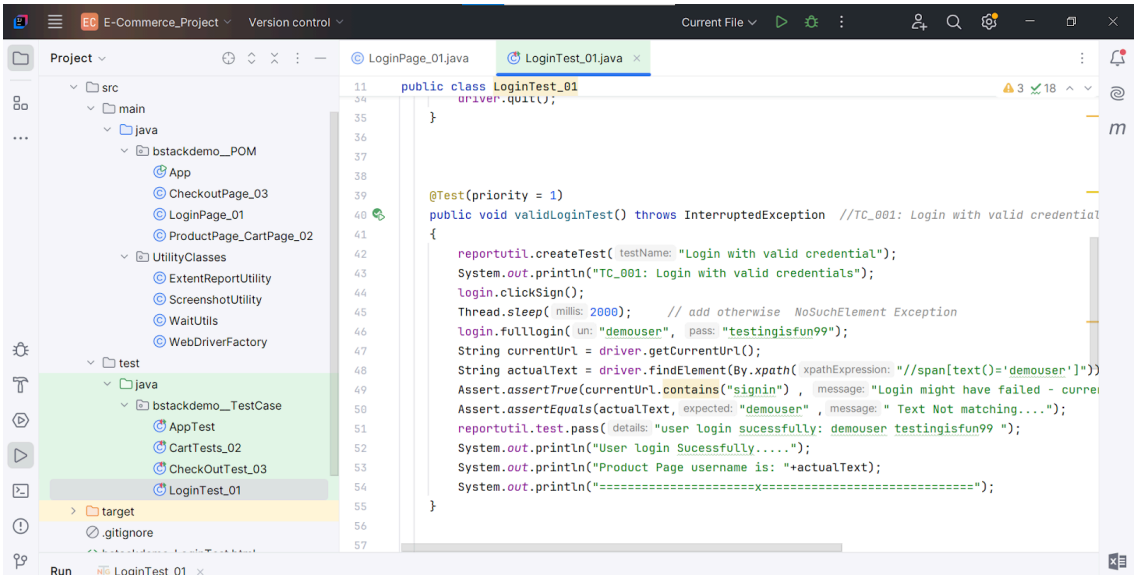


LoginTests

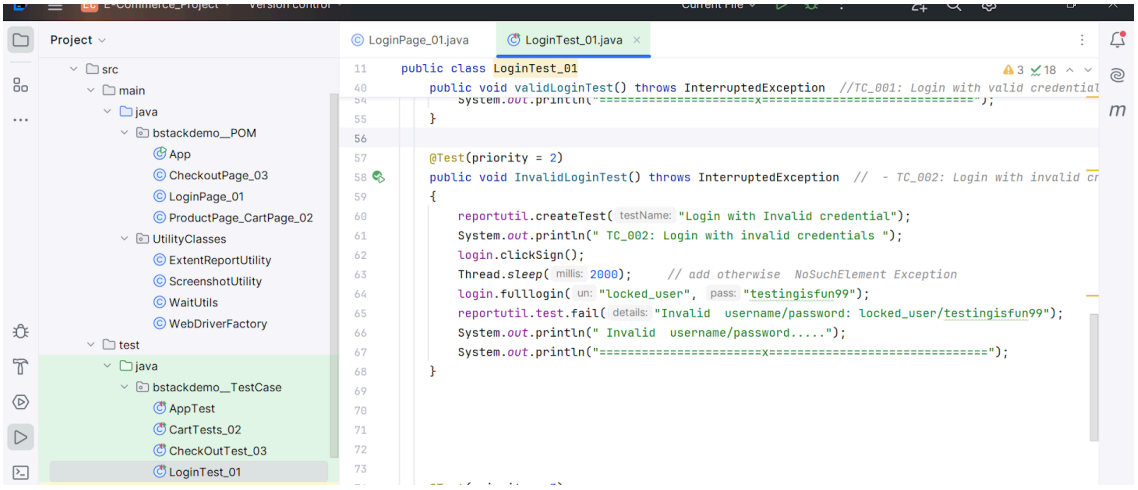
Before-AfterTest/Method



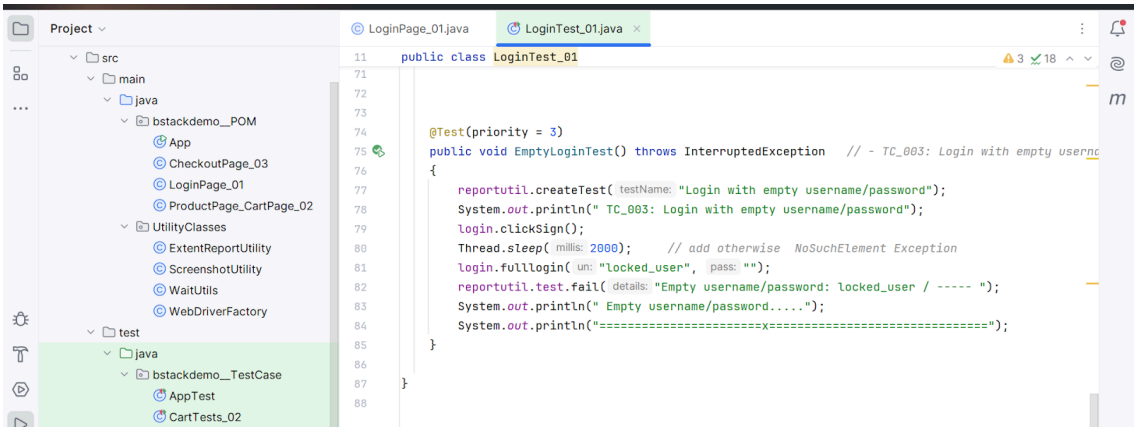
- TC_001: Login with valid credentials (demouser, testingisfun99)



- TC_002: Login with invalid credentials

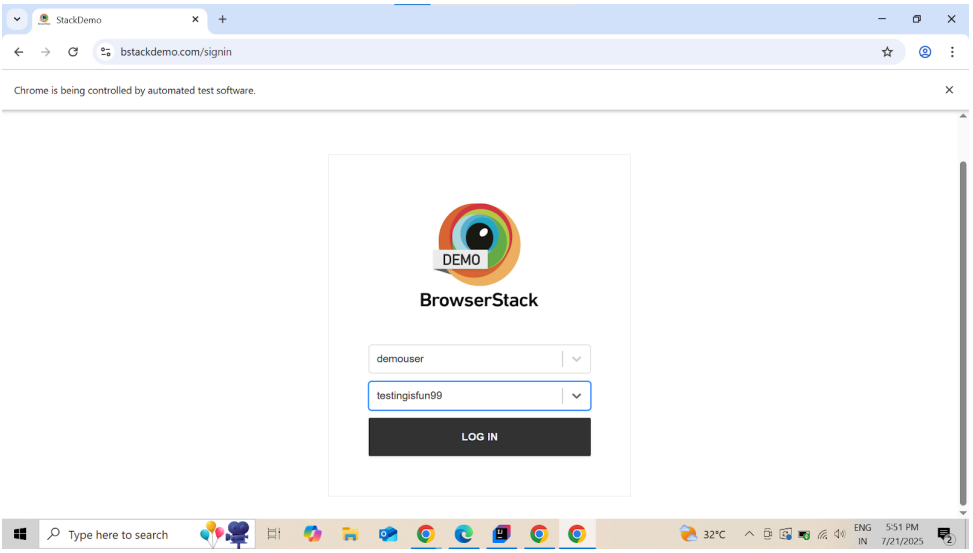


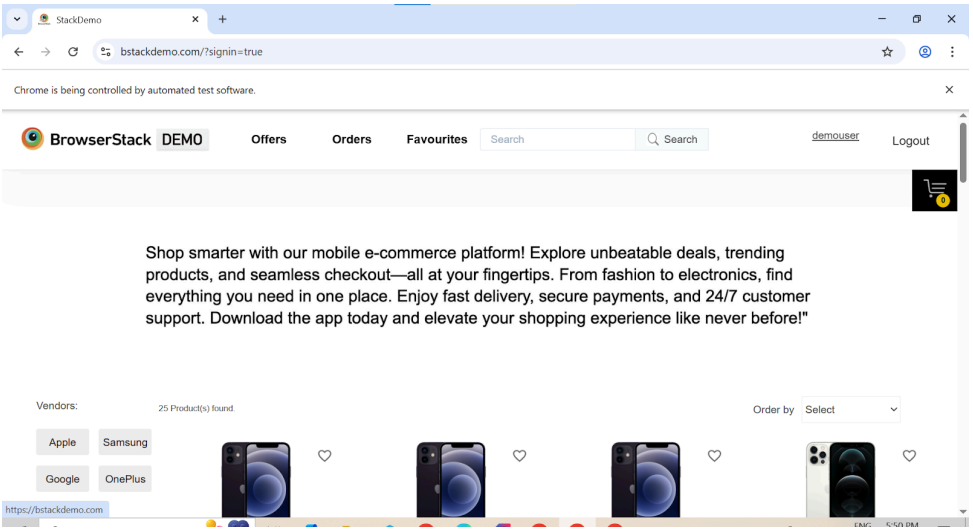
- TC_003: Login with empty fields



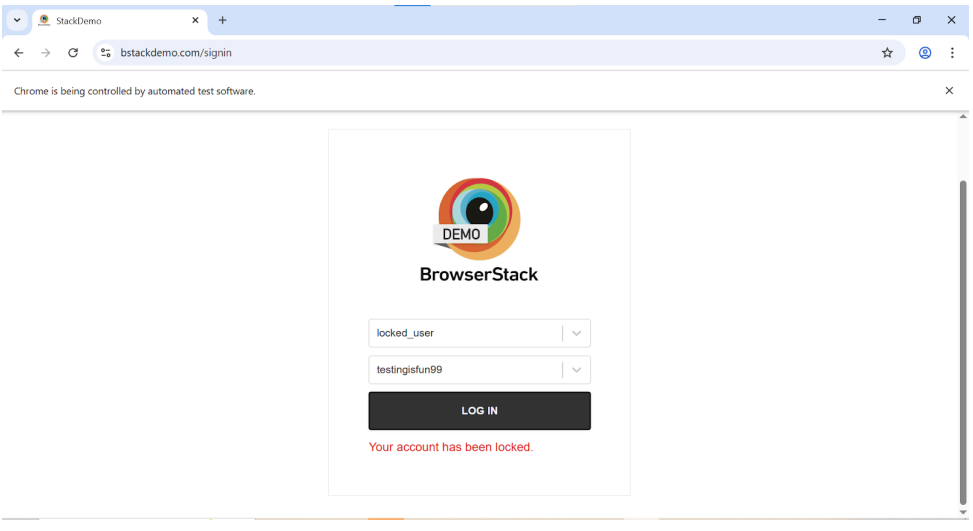
Screenshots

TC_001: Login with valid credentials (username: demouser, password: testingisfun99)

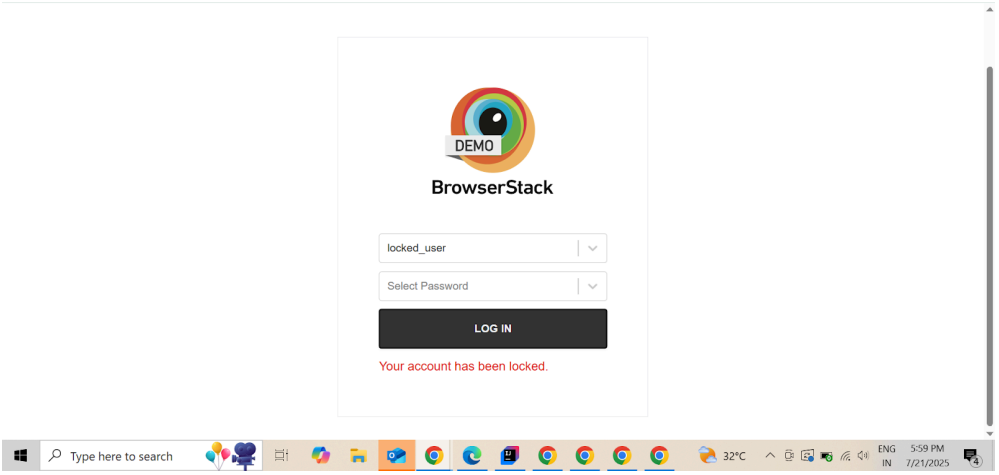




TC_002: Login with invalid credentials ("locked_user", "testingisfun99")



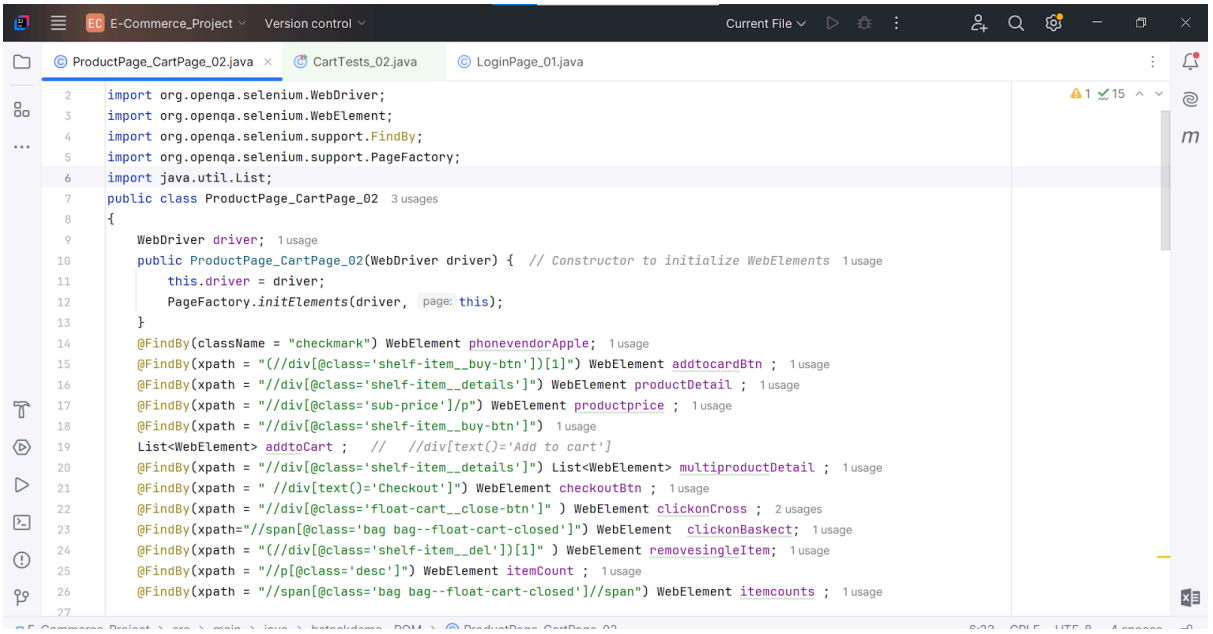
TC_003: Login with empty username/password("locked_user", "")

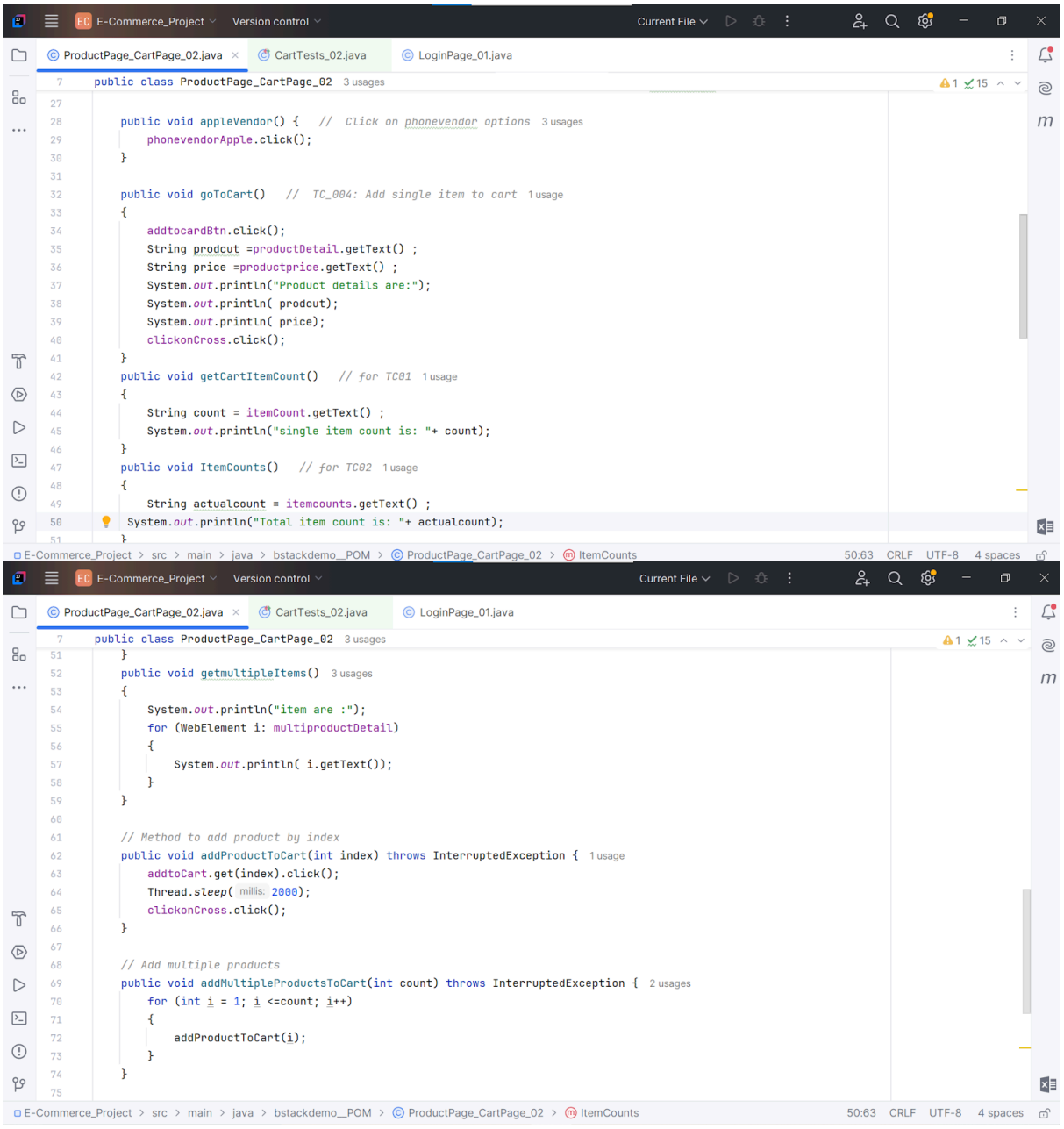


2. Cart Tests

- TC_004: Add single item to cart
- TC_005: Add multiple items and verify cart count
- TC_006: Remove item from cart

Product_CartPage : POM

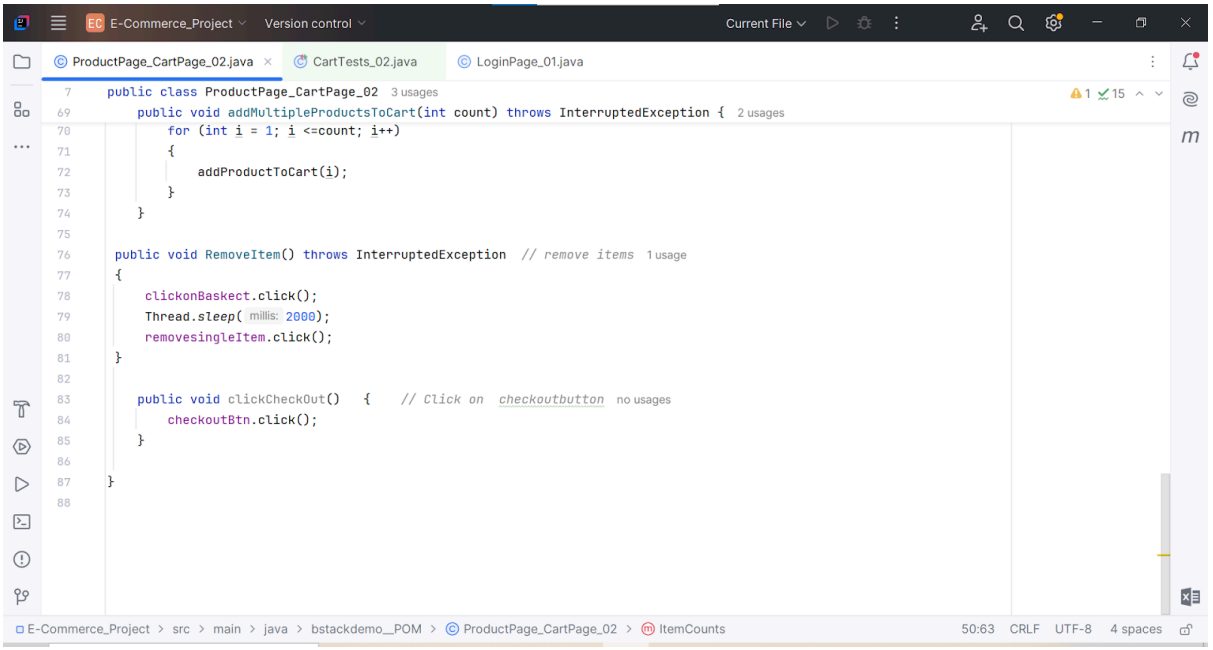




```
public class ProductPage_CartPage_02  3 usages

    public void appleVendor() {   //  Click on phonevendor options  3 usages
        phonevendorApple.click();
    }

    public void goToCart()   //  TC_004: Add single item to cart  1 usage
    {
        addtocardBtn.click();
        String prodcut =productDetail.getText() ;
        String price =productprice.getText() ;
        System.out.println("Product details are:");
        System.out.println( prodcut);
        System.out.println( price);
        clickonCross.click();
    }
    public void getCartItemCount()   // for TC01  1 usage
    {
        String count = itemCount.getText() ;
        System.out.println("single item count is: "+ count);
    }
    public void ItemCounts()   // for TC02  1 usage
    {
        String actualcount = itemcounts.getText() ;
        System.out.println("Total item count is: "+ actualcount);
    }
```



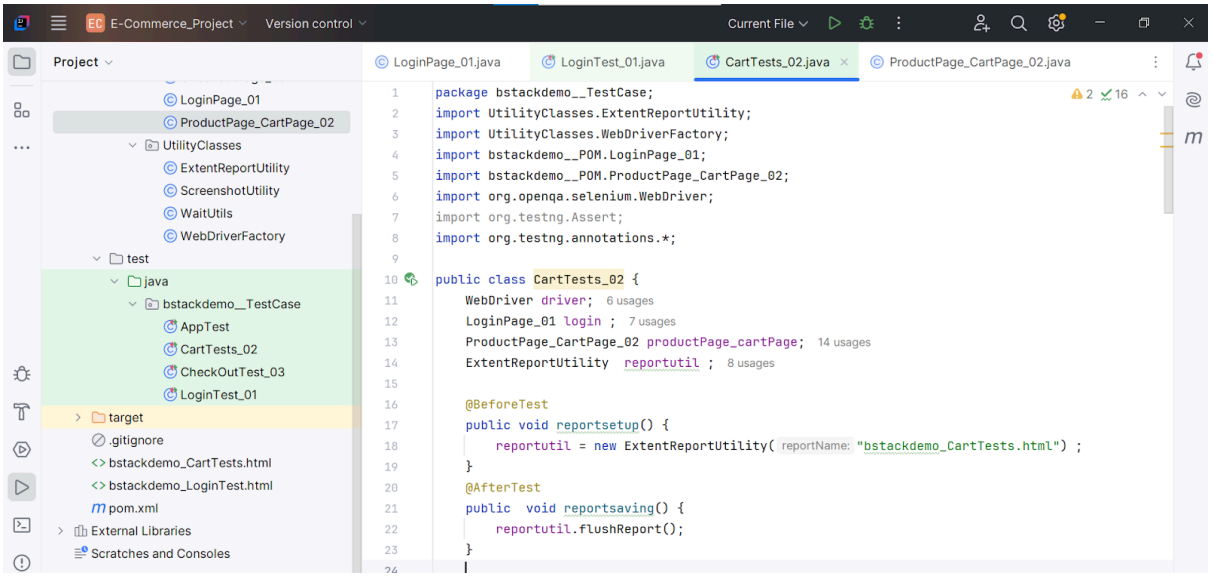
```
public class ProductPage_CartPage_02  3 usages
    }
    public void getmultipleItems()  3 usages
    {
        System.out.println("item are :");
        for (WebElement i: multiproductDetail)
        {
            System.out.println( i.getText());
        }
    }

    // Method to add product by index
    public void addProductToCart(int index) throws InterruptedException {  1 usage
        addtoCart.get(index).click();
        Thread.sleep( millis: 2000);
        clickonCross.click();
    }

    // Add multiple products
    public void addMultipleProductsToCart(int count) throws InterruptedException {  2 usages
        for (int i = 1; i <=count; i++)
        {
            addProductToCart(i);
        }
    }
```



```
public class ProductPage_CartPage_02  3 usages
    public void addMultipleProductsToCart(int count) throws InterruptedException {  2 usages
        for (int i = 1; i <=count; i++)
        {
            addProductToCart(i);
        }
    }

 public void RemoveItem() throws InterruptedException  // remove items  1 usage
 {
     clickonBaskect.click();
     Thread.sleep( millis: 2000);
     removesingleItem.click();
 }

    public void clickCheckOut()   {    // Click on  checkoutbutton  no usages
        checkoutBtn.click();
    }

}
```

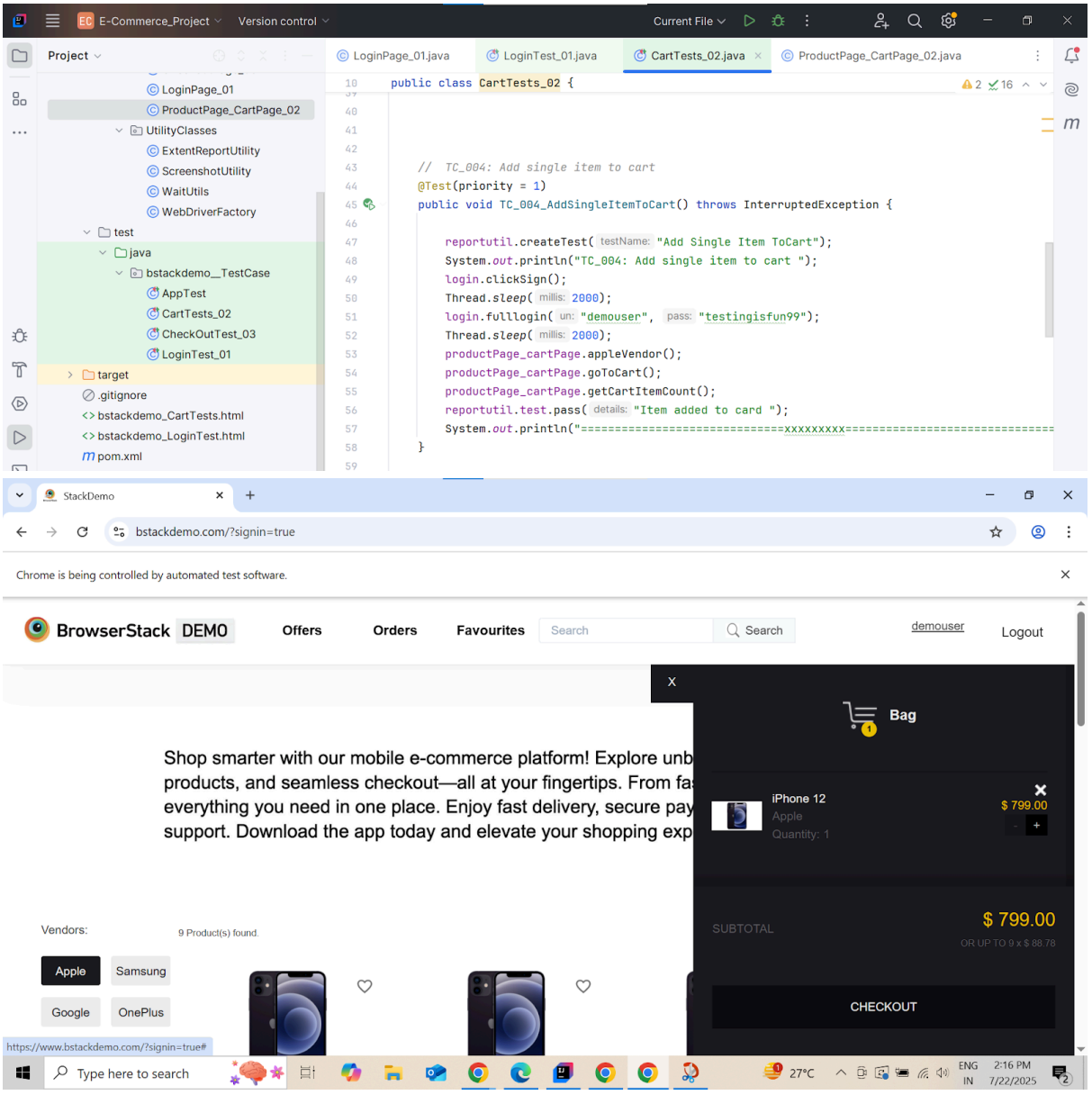
CartTestCases:

Before-AfterTest/Method:

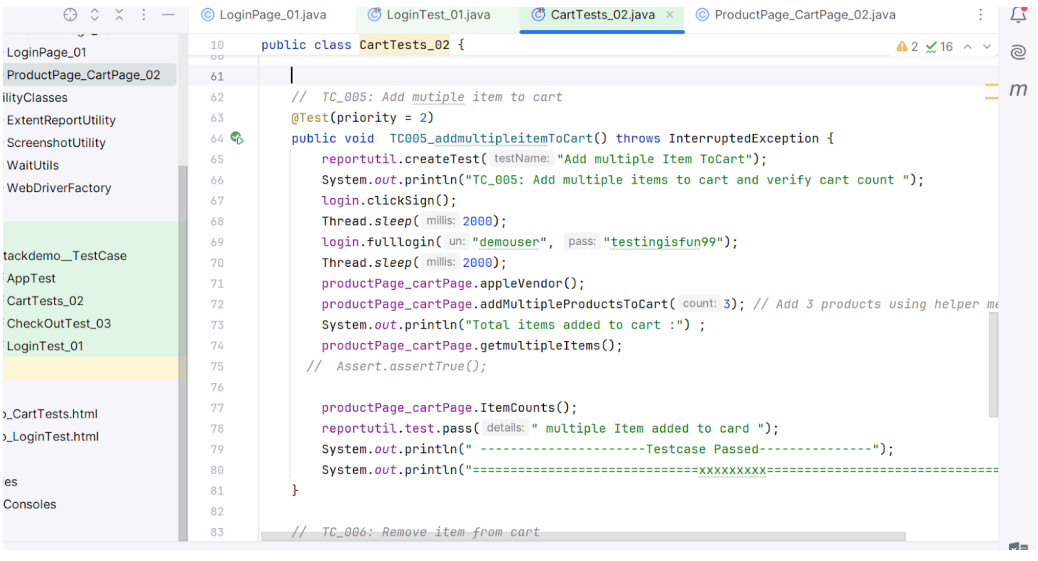


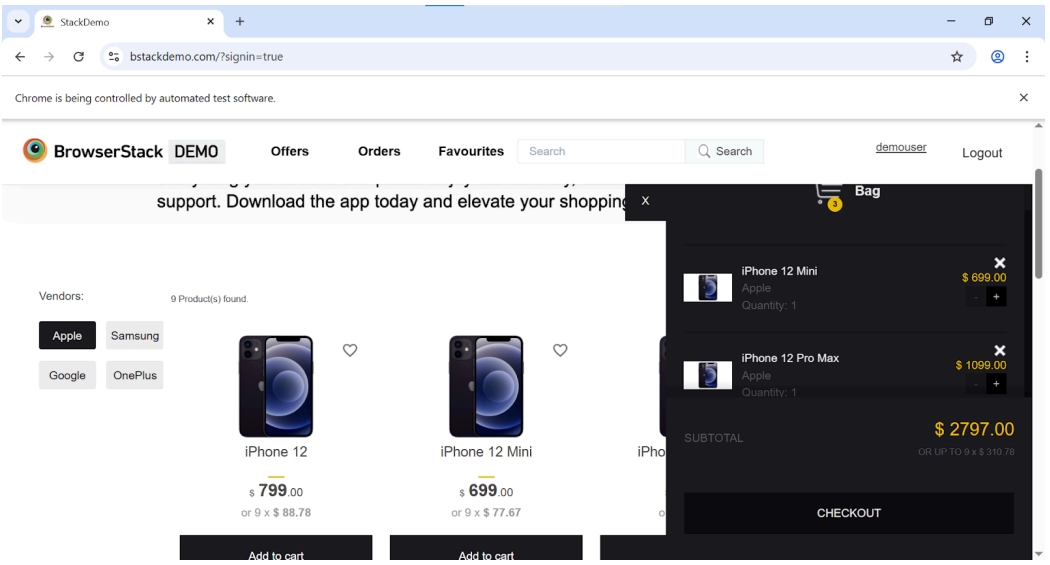

```
@BeforeMethod
public void setup() throws InterruptedException {
    driver = WebDriverFactory.getDriver();
    driver.manage().window().maximize();
    driver.get("https://www.bstackdemo.com/");
    Thread.sleep( millis: 3000);
    productPage_cartPage = new ProductPage_CartPage_02(driver);
    login = new  LoginPage_01(driver);
}
@AfterMethod
public void closeup() {
    driver.close();
}
```

- **TC_004: Add single item to cart**

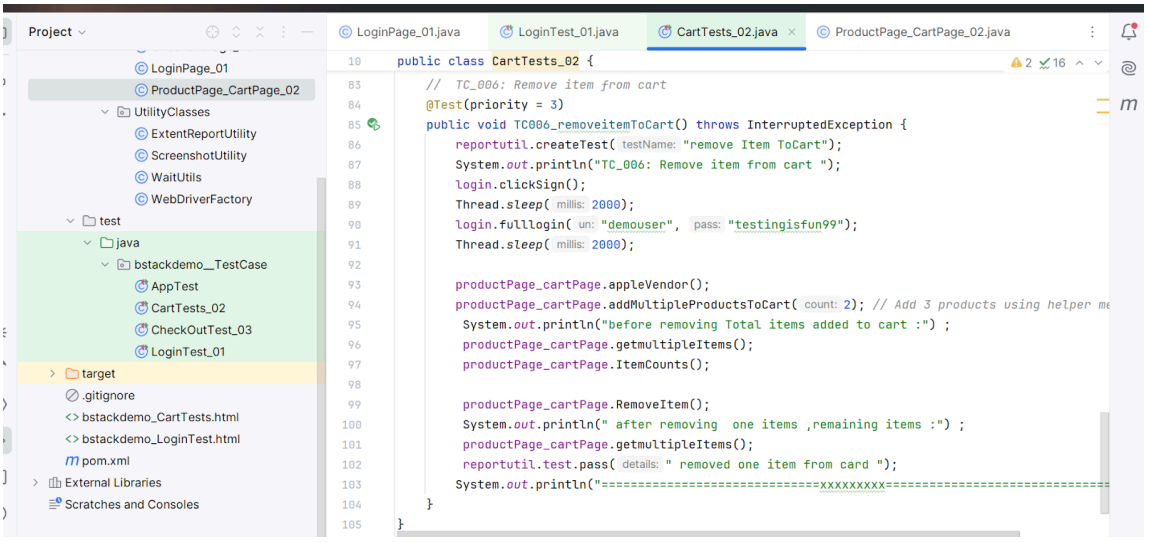


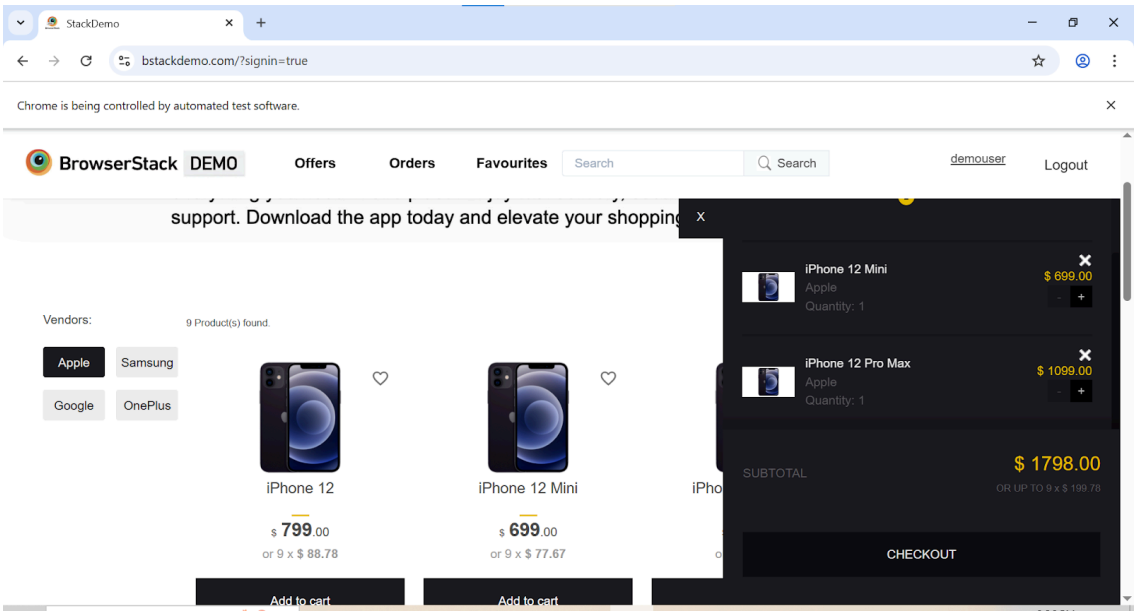
- **TC_005: Add multiple items and verify cart count**

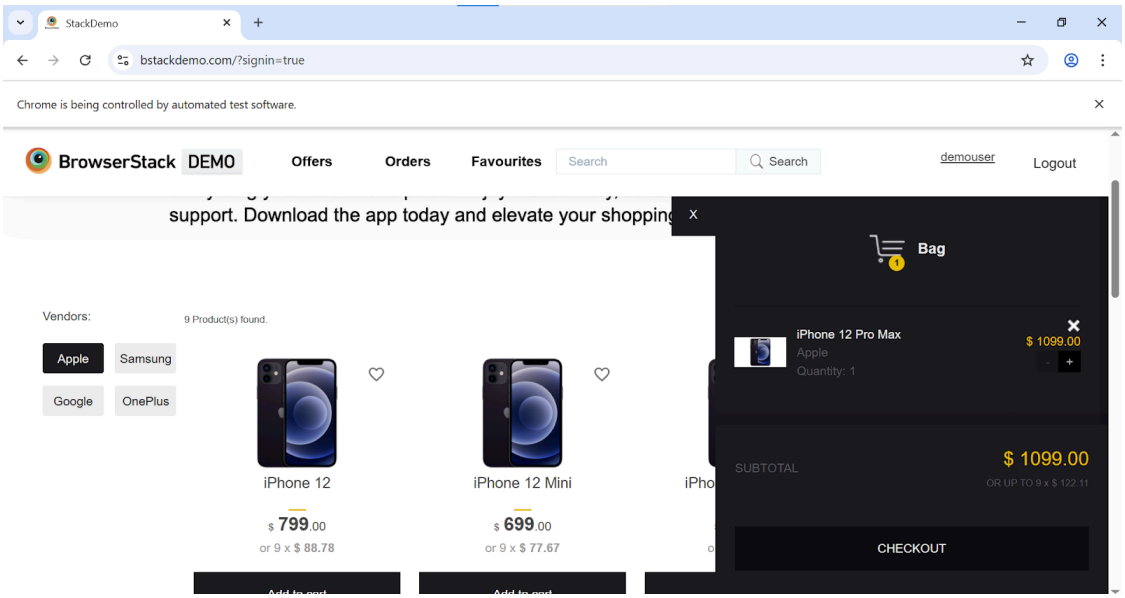




- **_TC_006: Remove item from cart**



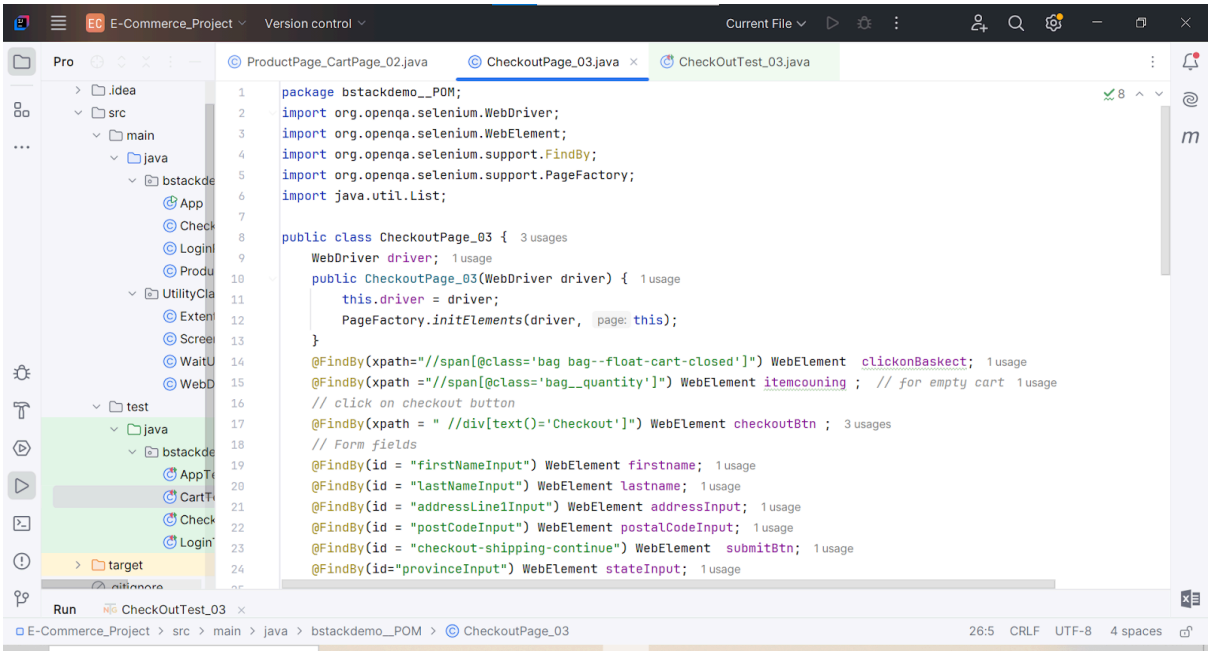


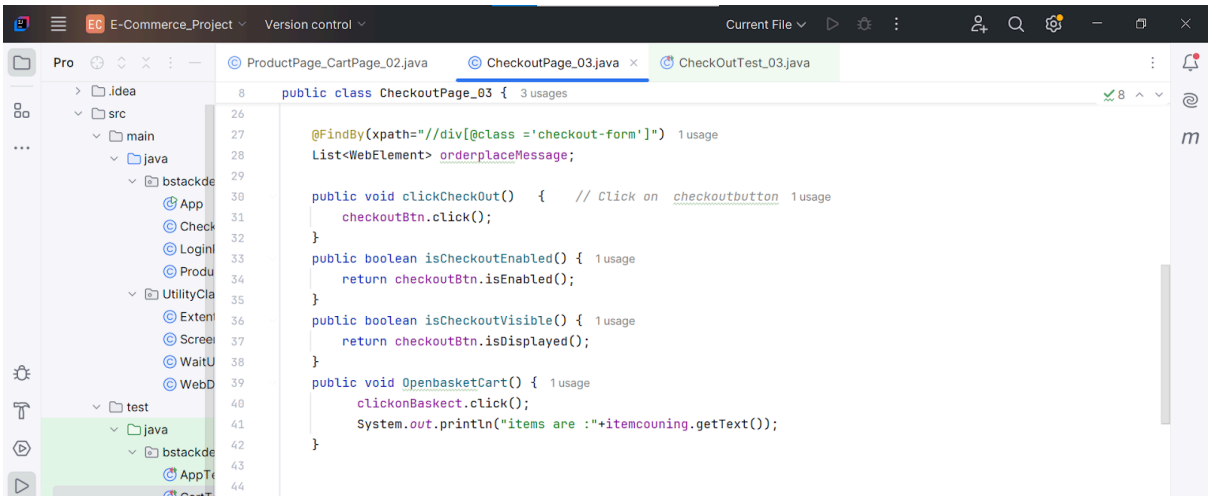


3. Checkout Tests

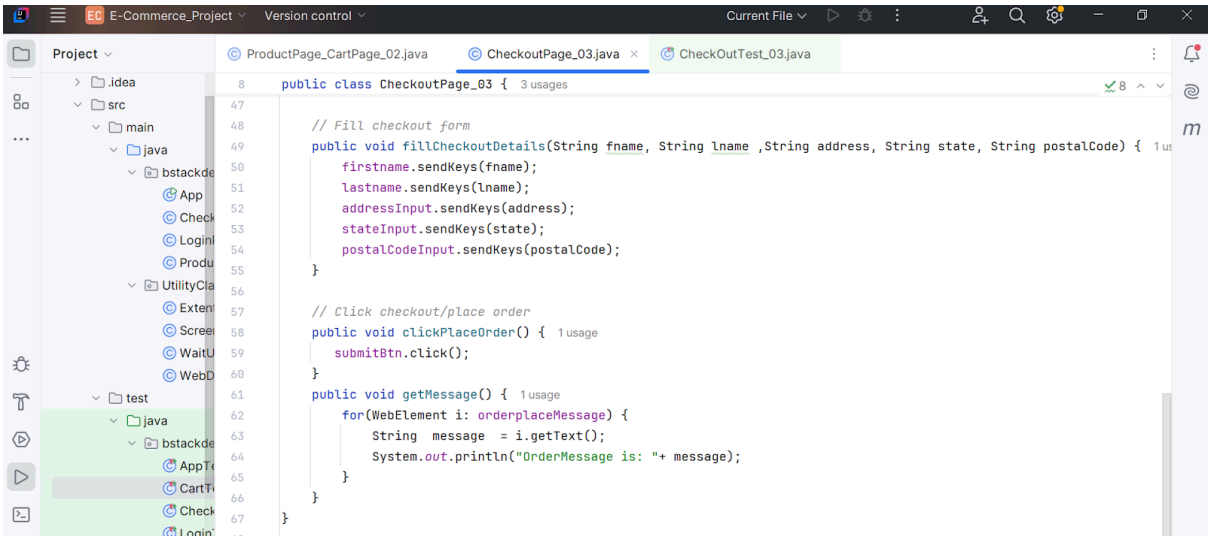
- TC_007: Place order with valid details
- TC_008: Try checkout without adding items (negative)

CheckOutPage: POM





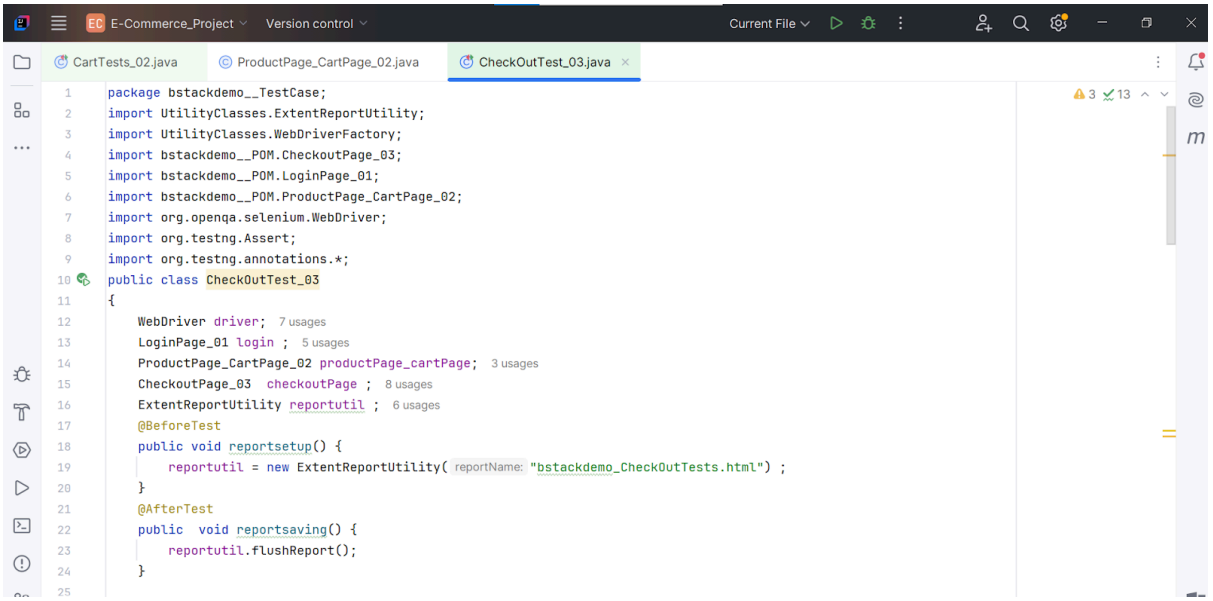
```
public class CheckoutPage_03 {  3 usages

    @FindBy(xpath="//div[@class ='checkout-form']")  1 usage
    List<WebElement> orderplaceMessage;

    public void clickCheckOut()   {    // Click on  checkoutbutton  1 usage
        checkoutBtn.click();
    }
    public boolean isCheckoutEnabled() {  1 usage
        return checkoutBtn.isEnabled();
    }
    public boolean isCheckoutVisible() {  1 usage
        return checkoutBtn.isDisplayed();
    }
    public void OpenbasketCart() {  1 usage
          clickonBaskect.click();
          System.out.println("items are :"+itemcouning.getText());
    }
```



```
public class CheckoutPage_03 {  3 usages

    // Fill checkout form
    public void fillCheckoutDetails(String fname, String lname ,String address, String state, String postalCode) {
        firstname.sendKeys(fname);
        lastname.sendKeys(lname);
        addressInput.sendKeys(address);
        stateInput.sendKeys(state);
        postalCodeInput.sendKeys(postalCode);
    }

    // Click checkout/place order
    public void clickPlaceOrder() {  1 usage
       submitBtn.click();
    }
    public void getMessage() {  1 usage
        for(WebElement i: orderplaceMessage) {
            String  message  = i.getText();
            System.out.println("OrderMessage is: "+ message);
        }
    }
}
```

CheckOutTests

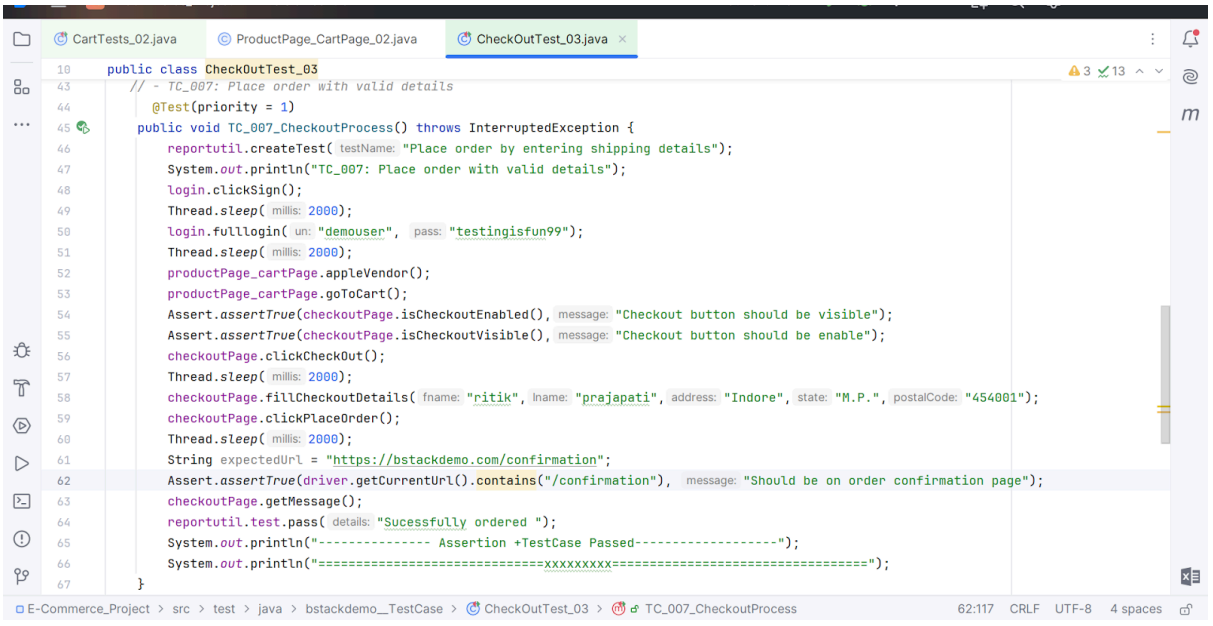
BeforeAfterTest/Method

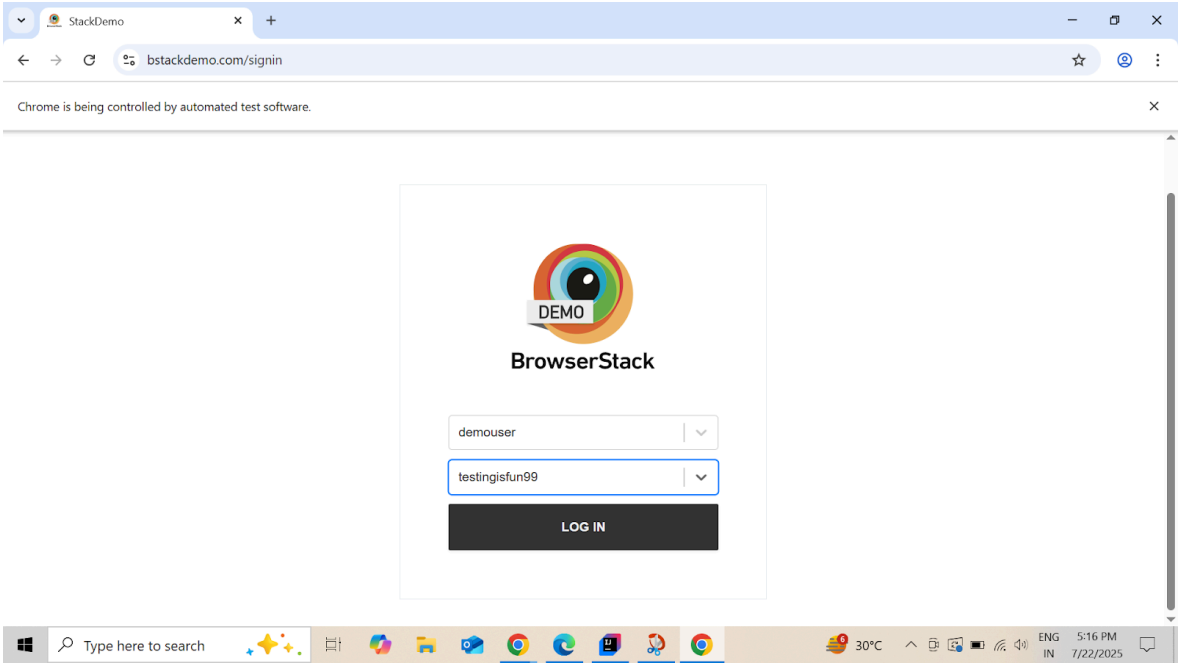


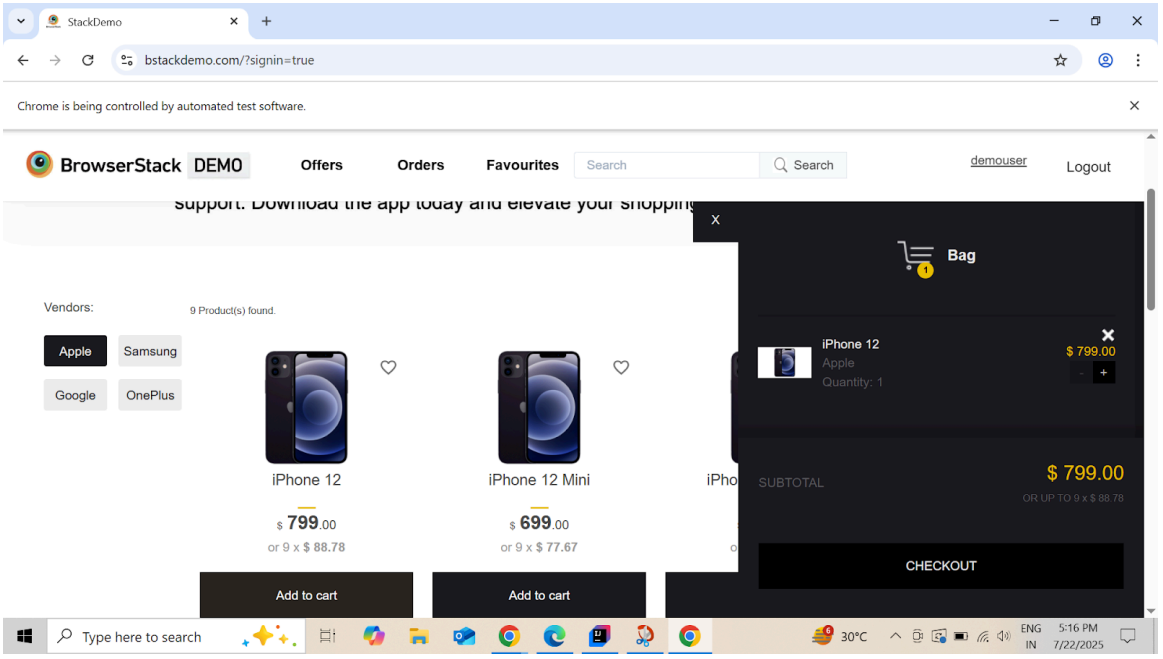
```
package bstackdemo__TestCase;
import UtilityClasses.ExtentReportUtility;
import UtilityClasses.WebDriverFactory;
import bstackdemo__POM.CheckoutPage_03;
import bstackdemo__POM.LoginPage_01;
import bstackdemo__POM.ProductPage_CartPage_02;
import org.openqa.selenium.WebDriver;
import org.testng.Assert;
import org.testng.annotations.*;
public class CheckOutTest_03
{
    WebDriver driver;  7 usages
    LoginPage_01 login ;  5 usages
    ProductPage_CartPage_02 productPage_cartPage;  3 usages
    CheckoutPage_03  checkoutPage ;  8 usages
    ExtentReportUtility reportutil ;  6 usages
    @BeforeTest
    public void reportsetup() {
        reportutil = new ExtentReportUtility( reportName: "bstackdemo_CheckOutTests.html") ;
    }
    @AfterTest
    public  void reportsaving() {
        reportutil.flushReport();
    }
```

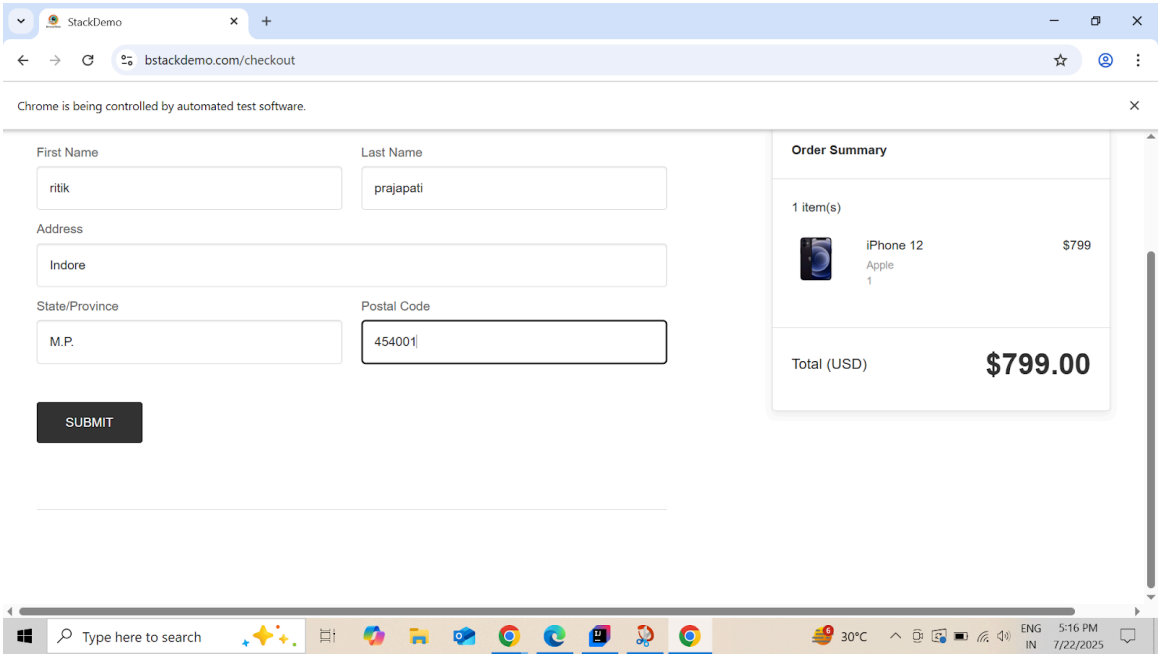
```
    }


    @BeforeMethod
    public void setup() throws InterruptedException {
        driver = WebDriverFactory.getDriver();
        driver.manage().window().maximize();
        driver.get("https://www.bstackdemo.com/");
        Thread.sleep( millis: 3000);

        productPage_cartPage = new ProductPage_CartPage_02(driver);
        login = new  LoginPage_01(driver);
        checkoutPage = new CheckoutPage_03(driver);
    }
    @AfterMethod
    public void closeup() {
      //  driver.close();
    }
```

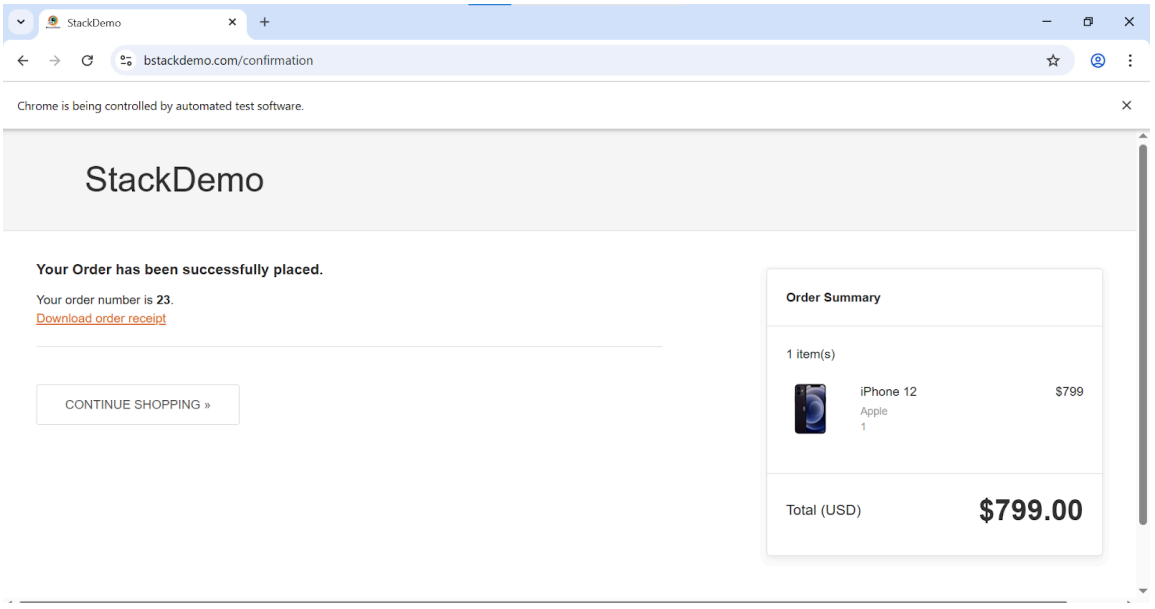
TC_007: Place order with valid details



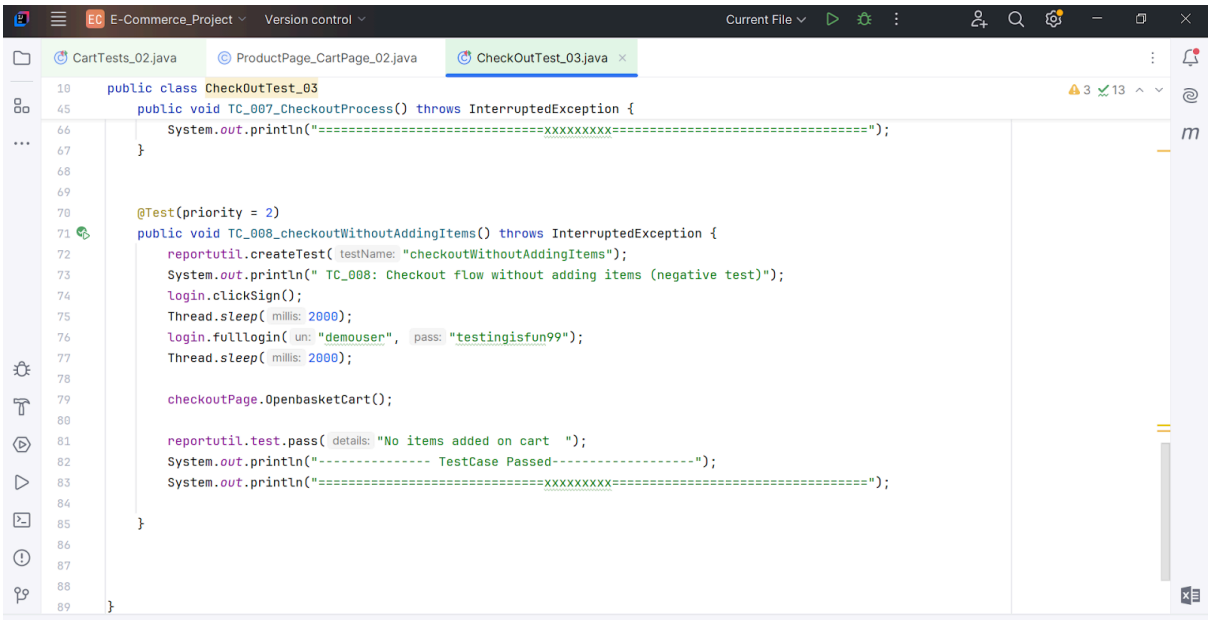



StackDemo
bstackdemo.com/?signin=true
Chrome is being controlled by automated test software.
BrowserStack DEMO
Offers
Orders
Favourites
Search
Search
demouser
Logout
Vendors:
9 Product(s) found.
Apple
Samsung
Google
OnePlus
iPhone 12
$ 799.00
or 9 x $ 88.78
Add to cart
iPhone 12 Mini
$ 699.00
or 9 x $ 77.67
Add to cart
Bag
iPhone 12
Apple
Quantity: 1
$ 799.00
SUBTOTAL
$ 799.00
OR UP TO 9 x $ 88.78
CHECKOUT
Type here to search
30°C
ENG
IN
5:16 PM
7/22/2025

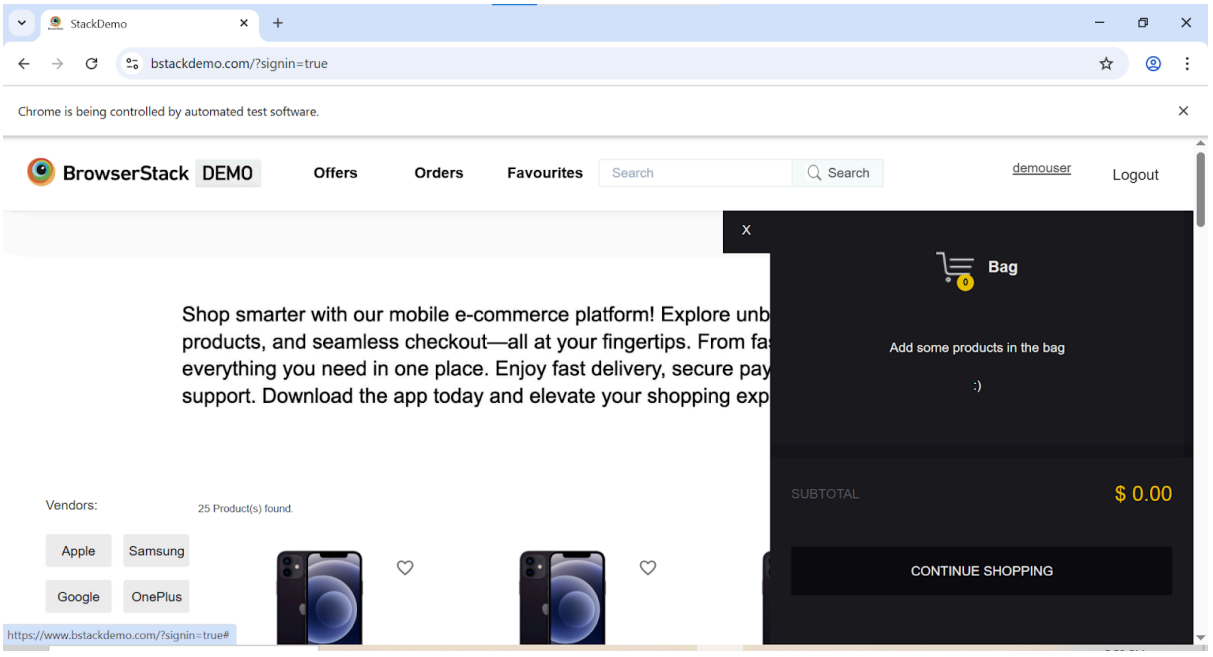

StackDemo
bstackdemo.com/checkout
Chrome is being controlled by automated test software.
First Name
ritik
Last Name
prajapati
Address
Indore
State/Province
M.P.
Postal Code
454001
SUBMIT
Order Summary
1 item(s)
iPhone 12
Apple
1
$799
Total (USD)
$799.00
Type here to search
30°C
ENG
IN
5:16 PM
7/22/2025


StackDemo
bstackdemo.com/confirmation
Chrome is being controlled by automated test software.
StackDemo
Your Order has been successfully placed.
Your order number is 23.
Download order receipt
CONTINUE SHOPPING »
Order Summary
1 item(s)
iPhone 12
Apple
1
$799
Total (USD)
$799.00

TC_008: Checkout flow without adding items (negative test)

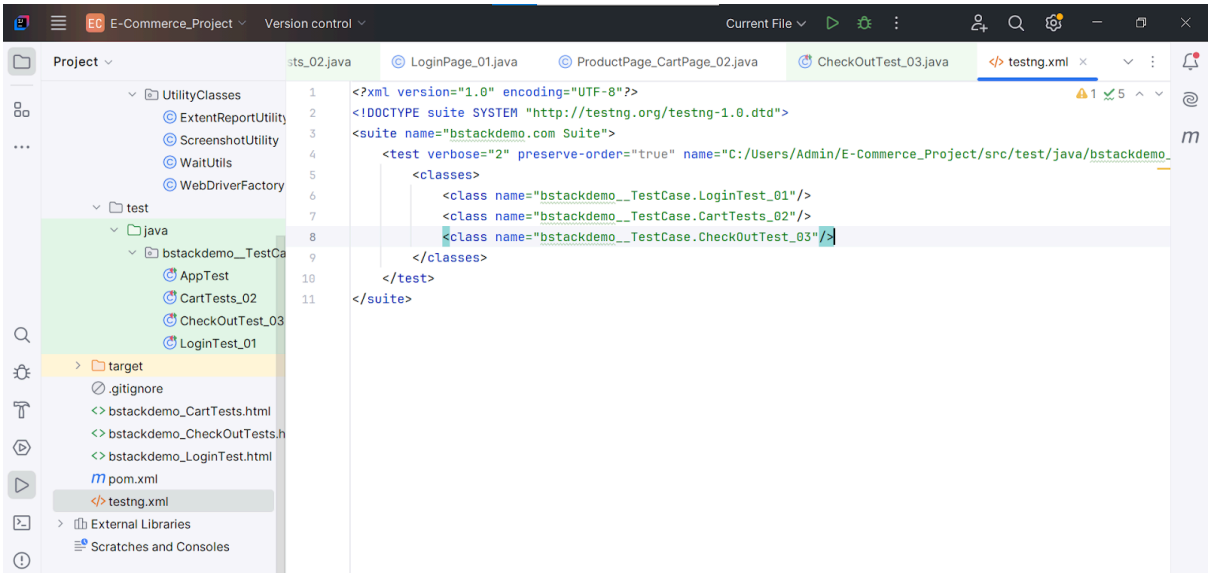


```
public class CheckOutTest_03
    public void TC_007_CheckoutProcess() throws InterruptedException {
        System.out.println("==============================xxxxxxxxx==================================");
    }


    @Test(priority = 2)
    public void TC_008_checkoutWithoutAddingItems() throws InterruptedException {
        reportutil.createTest( testName: "checkoutWithoutAddingItems");
        System.out.println(" TC_008: Checkout flow without adding items (negative test)");
        login.clickSign();
        Thread.sleep( millis: 2000);
        login.fulllogin( un: "demouser", pass: "testingisfun99");
        Thread.sleep( millis: 2000);

        checkoutPage.OpenbasketCart();

        reportutil.test.pass( details: "No items added on cart  ");
        System.out.println("--------------- TestCase Passed------------------");
        System.out.println("==============================xxxxxxxxx==================================");

    }

}
```



Testcase Execution:

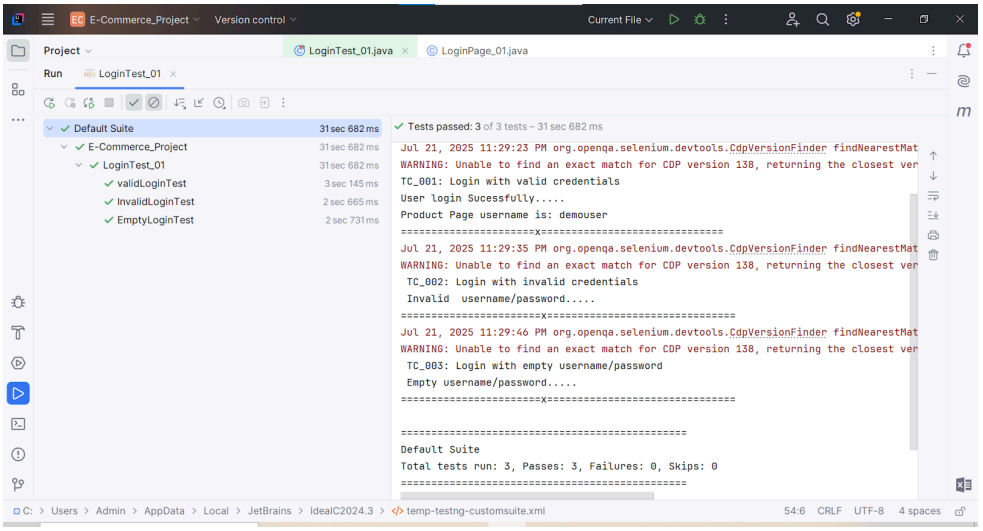
[Using TestNG XML]



9 . Test Result Analysis & Reporting

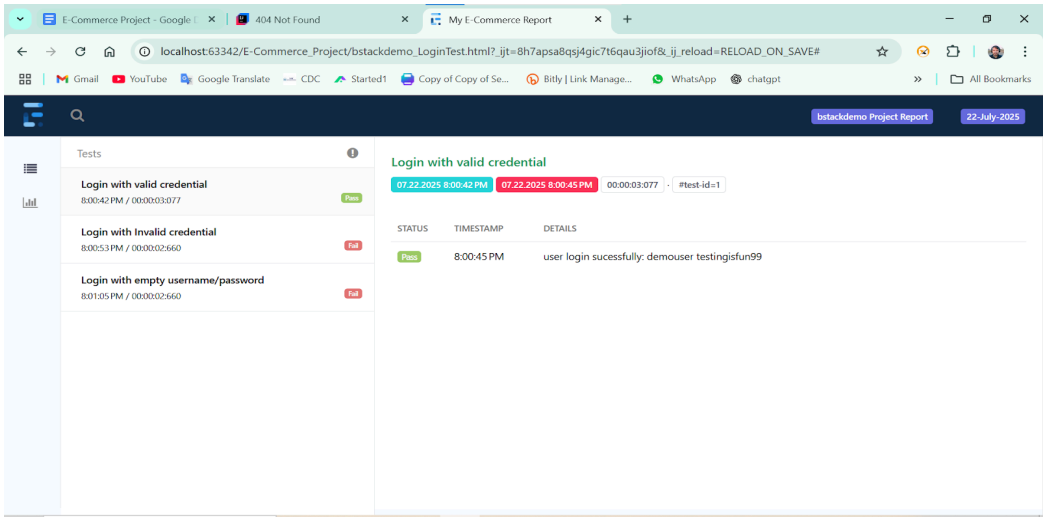
1. Login Tests:

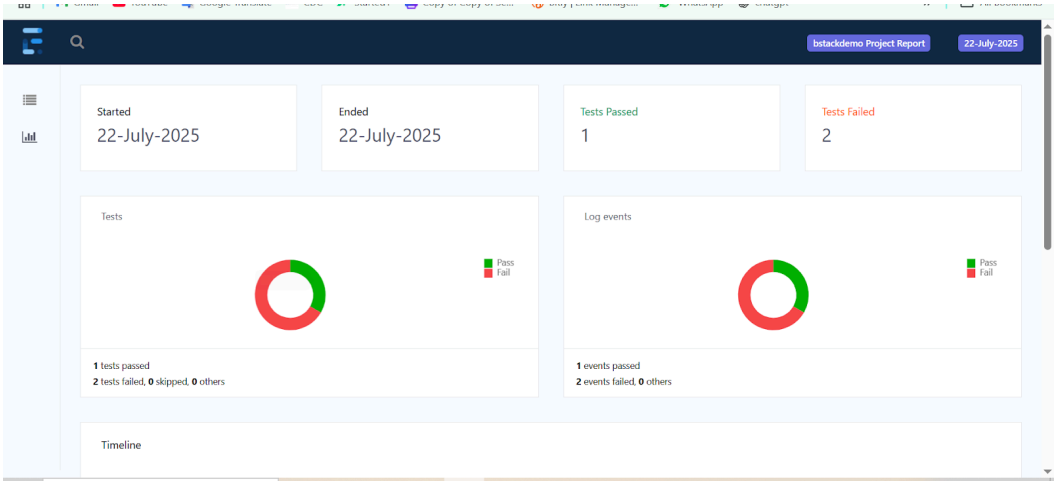
[TestNG report:]



ExtentReport:

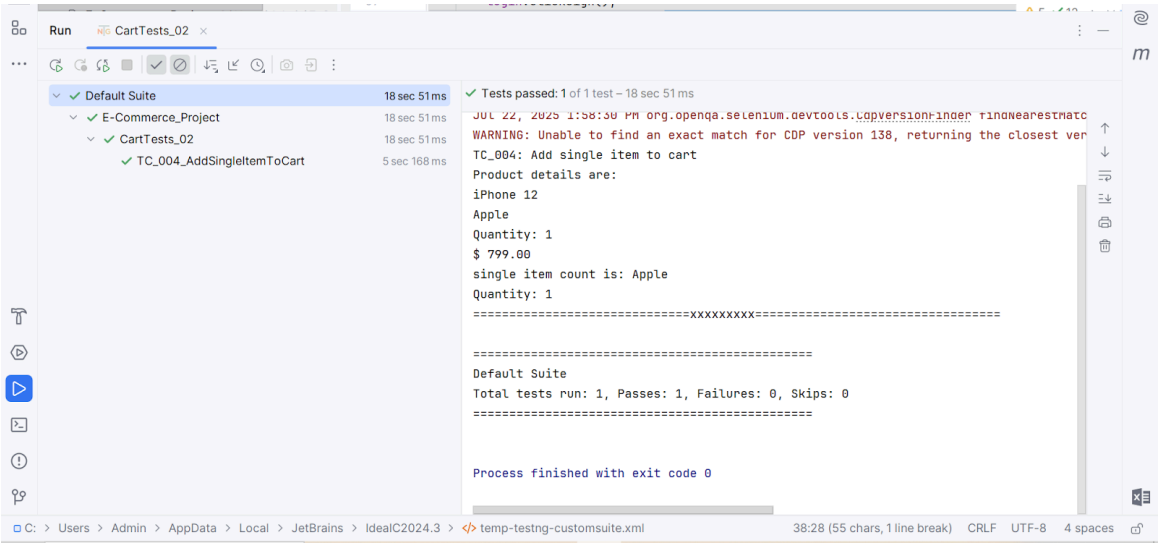
http://localhost:63342/E-Commerce_Project/bstackdemo_LoginTest.html?_ijt=8h7apsa8qsj4gic7t6qau3jiof&_ij_reload=RELOAD_ON_SAVE#

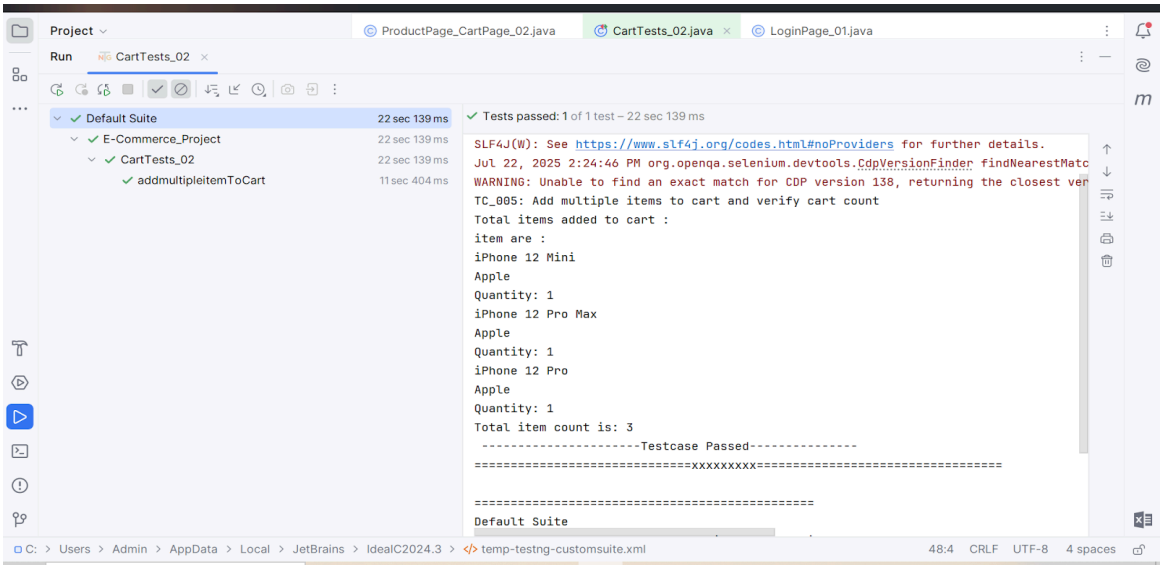


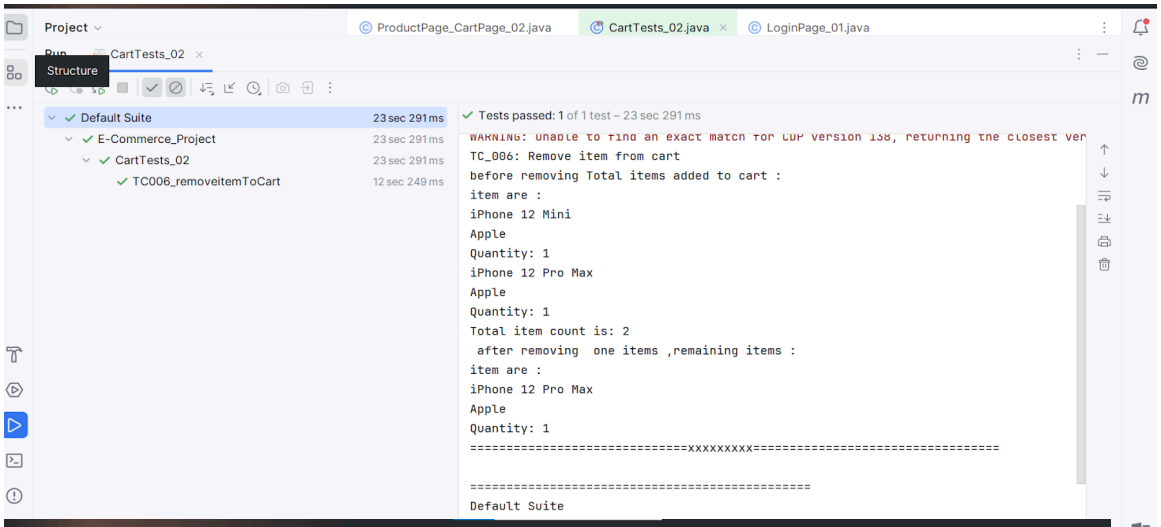


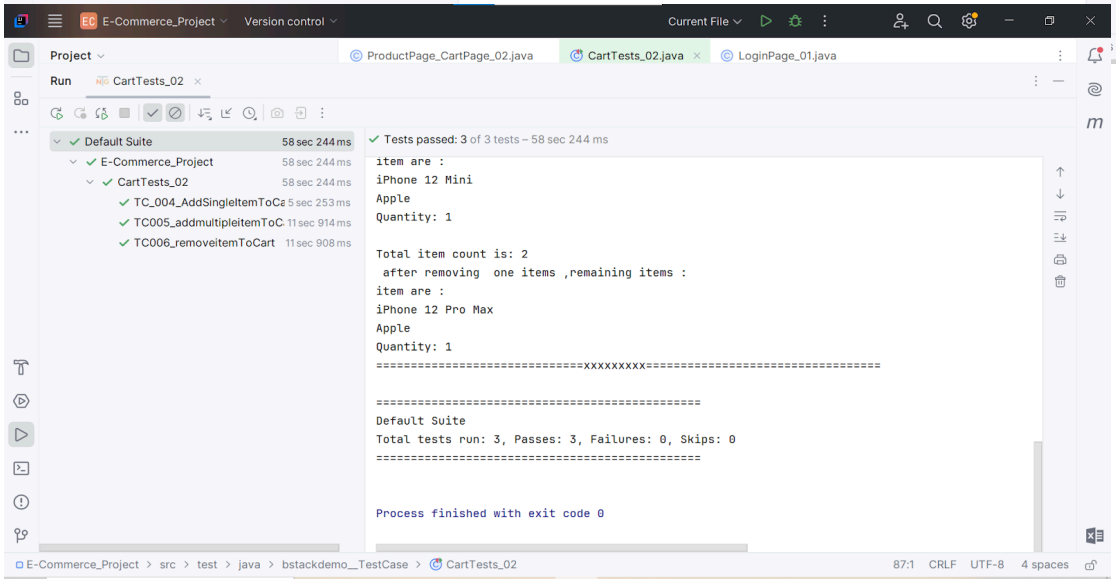
2. Cart Tests:

TestNG report:



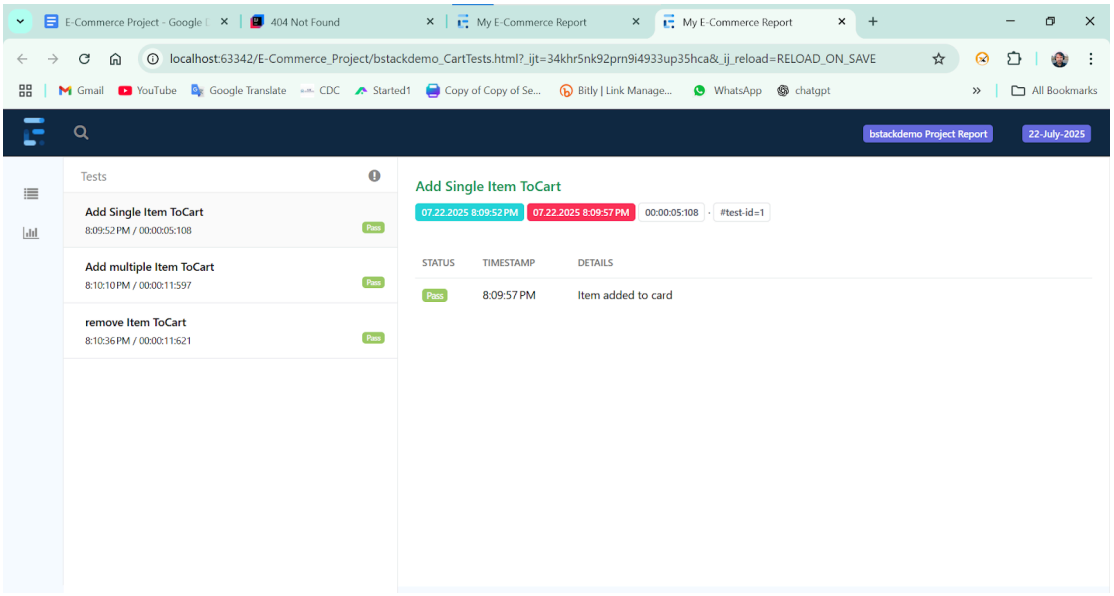

Project
ProductPage_CartPage_02.java
CartTests_02.java
LoginPage_01.java
Run
CartTests_02
Default Suite 22 sec 139 ms
E-Commerce_Project 22 sec 139 ms
CartTests_02 22 sec 139 ms
addmultipleitemToCart 11 sec 404 ms
Tests passed: 1 of 1 test – 22 sec 139 ms
SLF4J(W): See https://www.slf4j.org/codes.html#noProviders for further details.
Jul 22, 2025 2:24:46 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatc
WARNING: Unable to find an exact match for CDP version 138, returning the closest ver
TC_005: Add multiple items to cart and verify cart count
Total items added to cart :
item are :
iPhone 12 Mini
Apple
Quantity: 1
iPhone 12 Pro Max
Apple
Quantity: 1
iPhone 12 Pro
Apple
Quantity: 1
Total item count is: 3
----------Testcase Passed----------
==========xxxxxxxxx==========
==========
Default Suite
C: > Users > Admin > AppData > Local > JetBrains > IdealC2024.3 > temp-testng-customsuite.xml
48:4 CRLF UTF-8 4 spaces

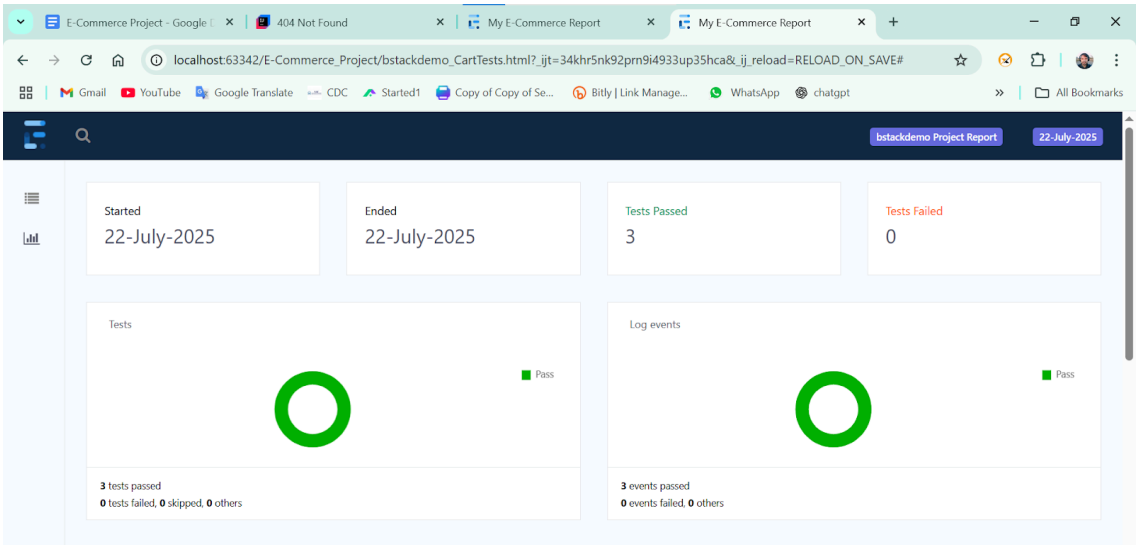

Project
Structure
ProductPage_CartPage_02.java
CartTests_02.java
LoginPage_01.java
Run
CartTests_02
Default Suite 23 sec 291 ms
E-Commerce_Project 23 sec 291 ms
CartTests_02 23 sec 291 ms
TC006_removeitemToCart 12 sec 249 ms
Tests passed: 1 of 1 test – 23 sec 291 ms
TC_006: Remove item from cart
before removing Total items added to cart :
item are :
iPhone 12 Mini
Apple
Quantity: 1
iPhone 12 Pro Max
Apple
Quantity: 1
Total item count is: 2
after removing one items ,remaining items :
item are :
iPhone 12 Pro Max
Apple
Quantity: 1
==========xxxxxxxxx==========
==========
Default Suite


E-Commerce_Project
Version control
Current File
Project
ProductPage_CartPage_02.java
CartTests_02.java
LoginPage_01.java
Run
CartTests_02
Default Suite 58 sec 244 ms
E-Commerce_Project 58 sec 244 ms
CartTests_02 58 sec 244 ms
TC_004_AddSingleItemToCa 5 sec 253 ms
TC005_addmultipleitemToC 11 sec 914 ms
TC006_removeitemToCart 11 sec 908 ms
Tests passed: 3 of 3 tests – 58 sec 244 ms
item are :
iPhone 12 Mini
Apple
Quantity: 1
Total item count is: 2
after removing one items ,remaining items :
item are :
iPhone 12 Pro Max
Apple
Quantity: 1
==========xxxxxxxxx==========
==========
Default Suite
Total tests run: 3, Passes: 3, Failures: 0, Skips: 0
==========
Process finished with exit code 0
E-Commerce_Project > src > test > java > bstackdemo__TestCase > CartTests_02
87:1 CRLF UTF-8 4 spaces

ExtentReport:

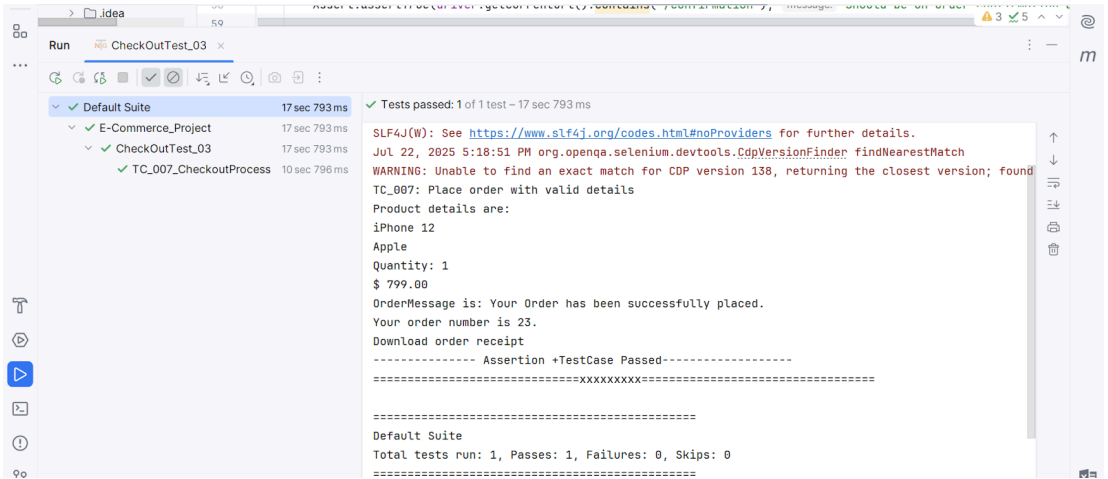
http://localhost:63342/E-Commerce_Project/bstackdemo_CartTests.html?_ijt=34khr5nk92prn9i4933up35hca&_ij_reload=RELOAD_ON_SAVE#

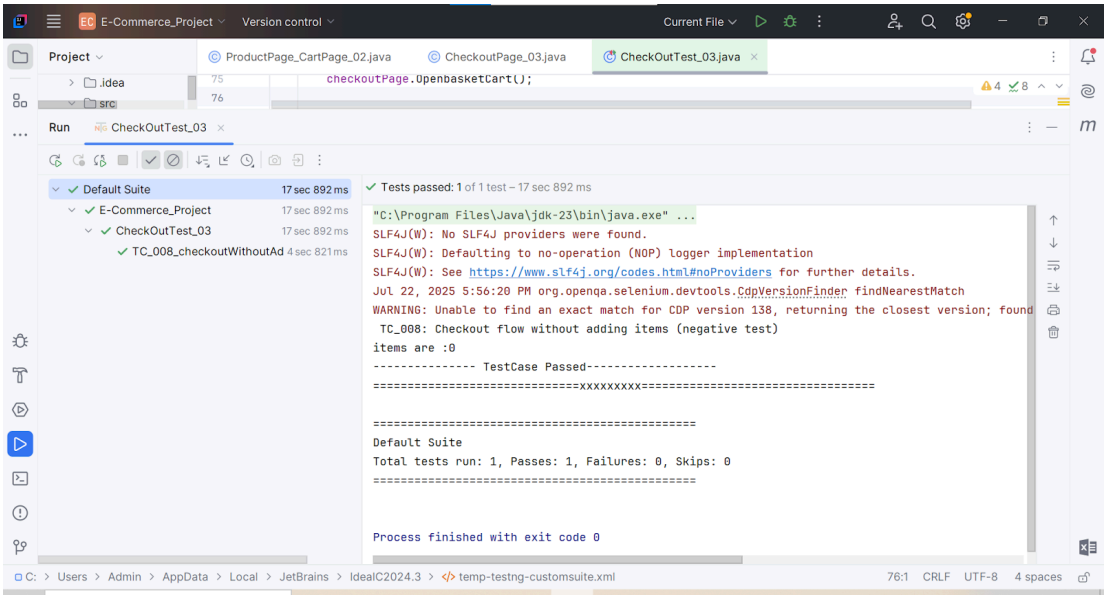




3. Checkout Tests:

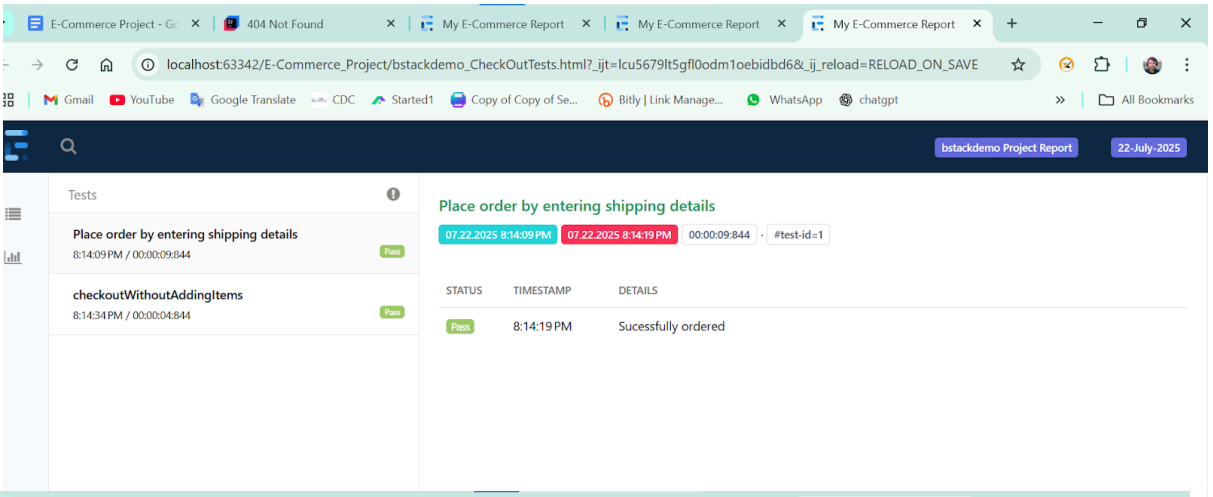
TestNG report:

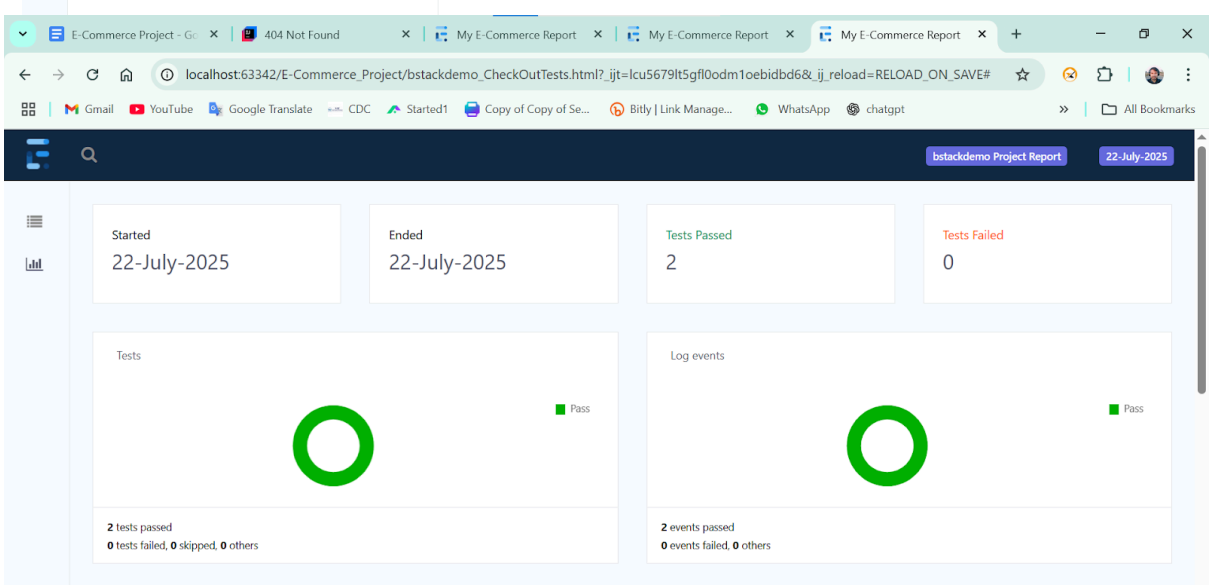


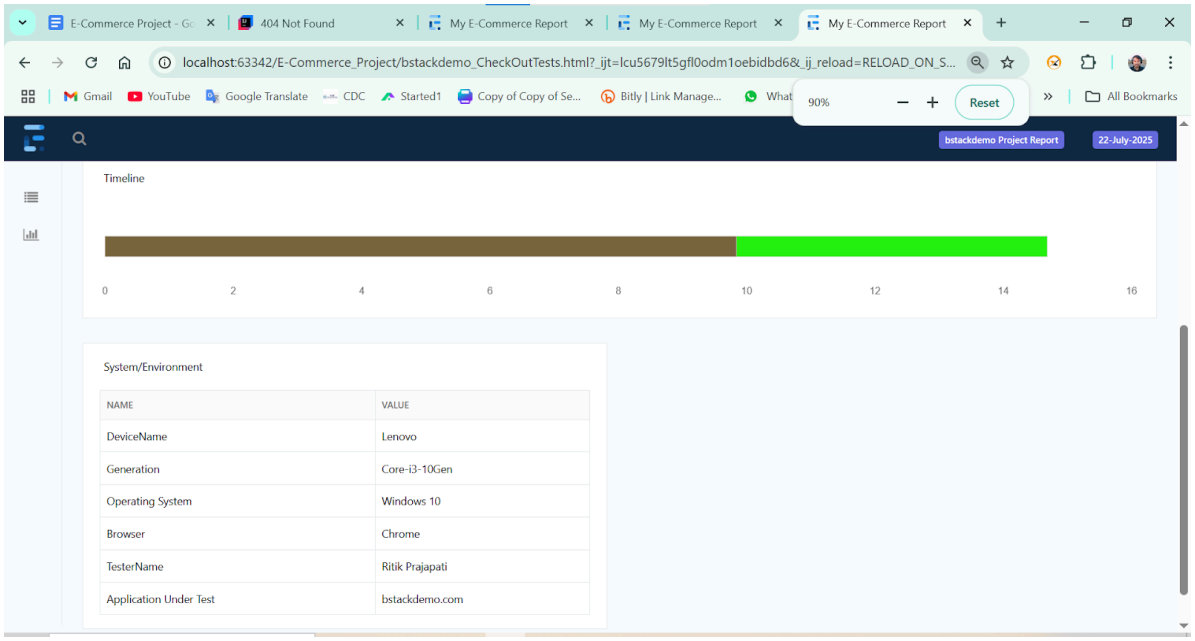


ExtentReport:

http://localhost:63342/E-Commerce_Project/bstackdemo_CheckOutTests.html?_ijt=lcu5679lt5gfl0odm1oebidbd6&_ij_reload=RELOAD_ON_SAVE#







10. Conclusion

The automation project for bstackdemo.com successfully demonstrates a robust and modular end-to-end testing framework using Java, Selenium WebDriver, TestNG, Maven, and ExtentReports. By implementing the Page Object Model (POM) design pattern, the framework ensures reusability, readability, and maintainability across all major modules — login, cart, and checkout. The test case was systematically validated using clear assertions and enriched with real-time ExtentReports, providing a detailed visual summary of the test execution results.

---------- **END** ----------