

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**](https://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

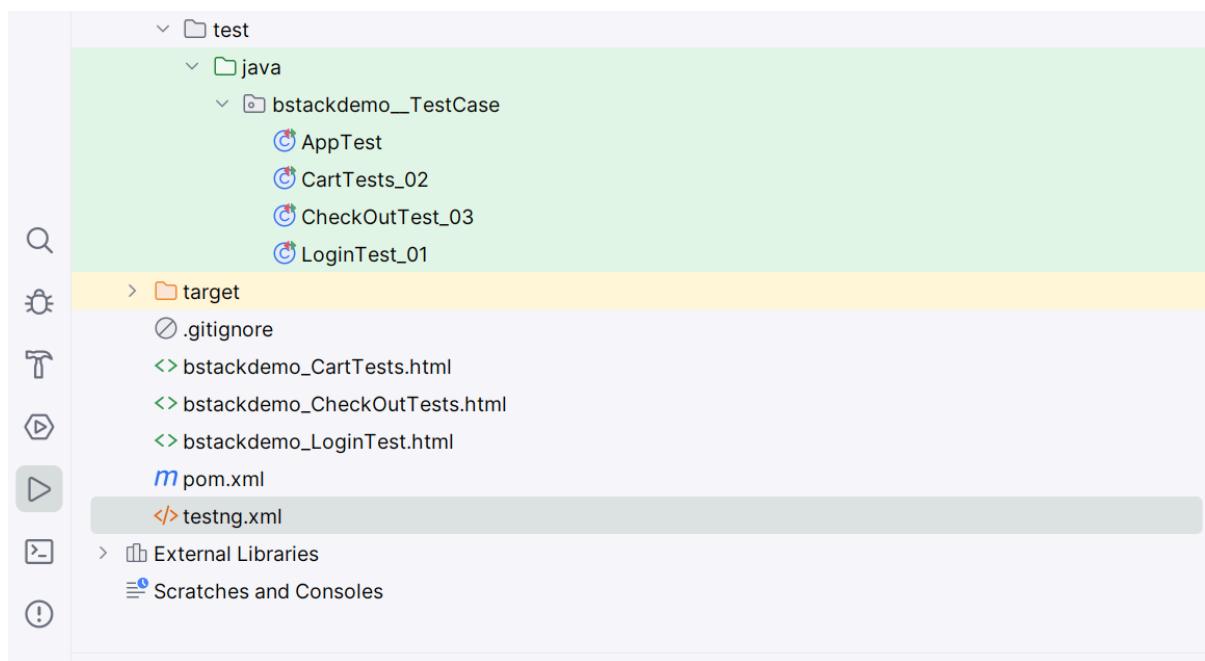
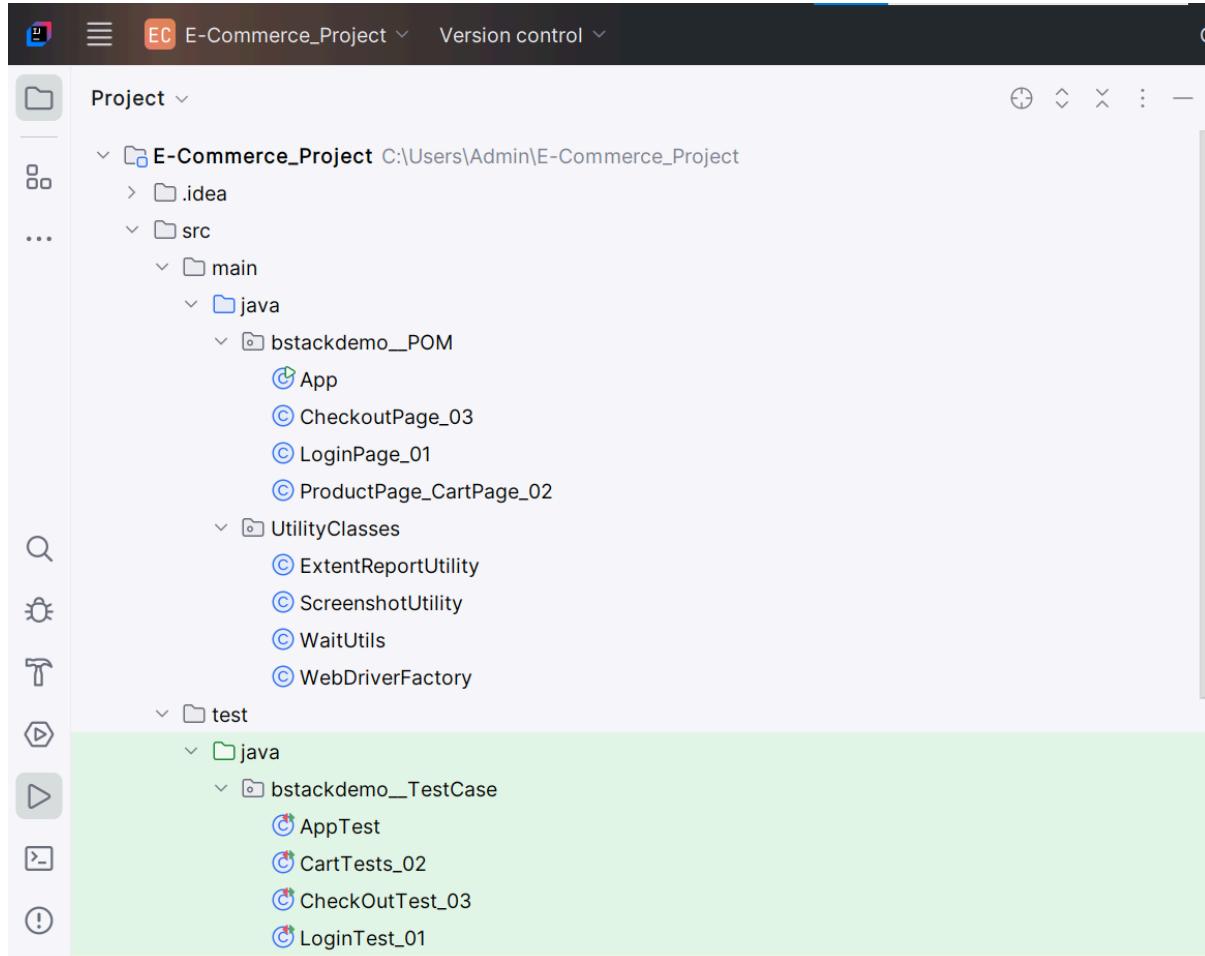
2. Cart Tests

- 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

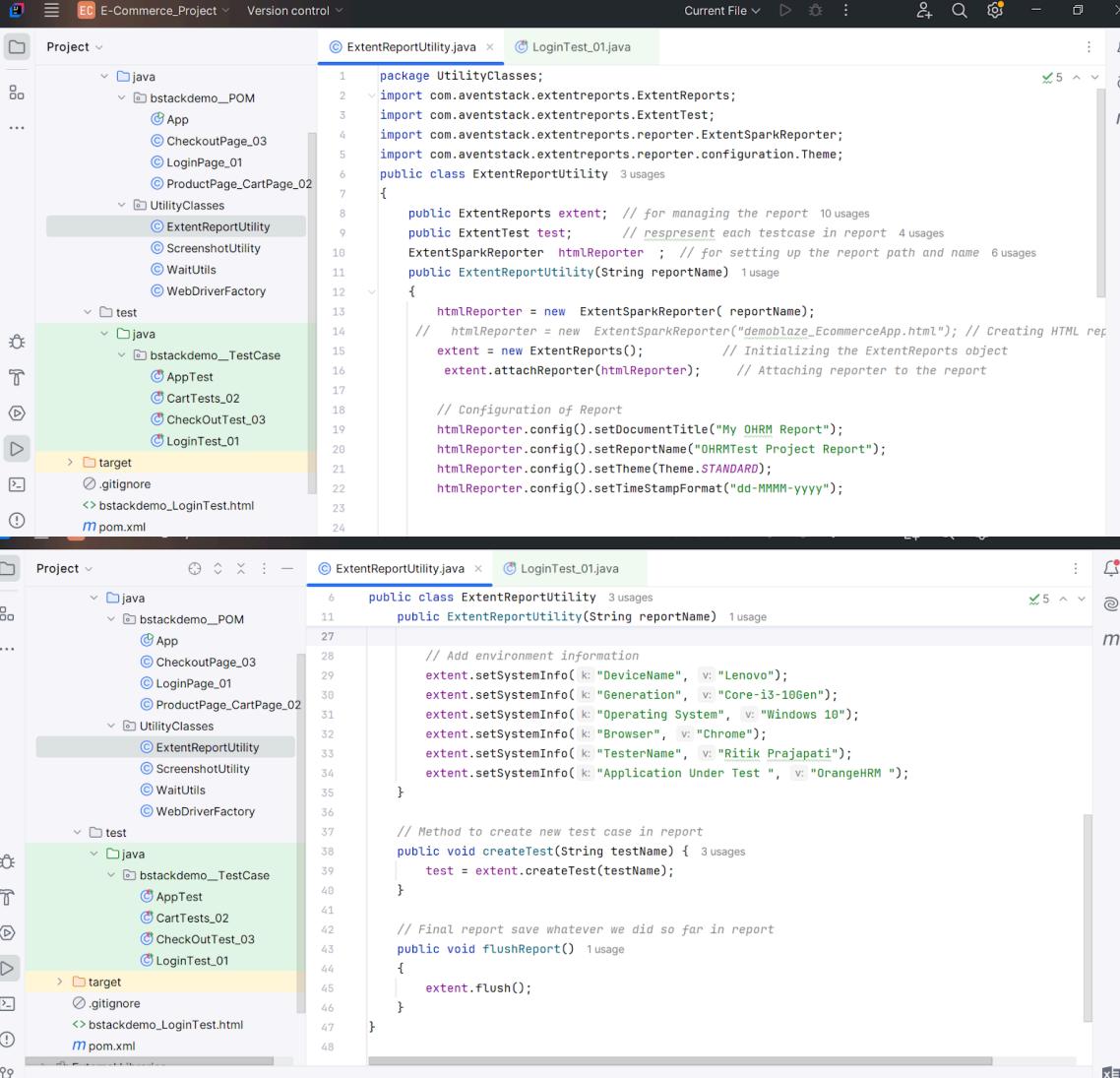
- TC_007: Place order with valid details
- TC_008: Try checkout without adding items (negative)

7. Sample Code Structure



8. Testcase Design & Execution

UtilityClass: ExtentReport



```

1 package UtilityClasses;
2 import com.aventstack.extentreports.ExtentReports;
3 import com.aventstack.extentreports.ExtentTest;
4 import com.aventstack.extentreports.reporter.ExtentSparkReporter;
5 import com.aventstack.extentreports.reporter.configuration.Theme;
6 public class ExtentReportUtility 3 usages
{
7     public ExtentReports extent; // for managing the report 10 usages
8     public ExtentTest test; // represent each testcase in report 4 usages
9     ExtentSparkReporter htmlReporter; // for setting up the report path and name 6 usages
10    public ExtentReportUtility(String reportName) 1 usage
11    {
12        htmlReporter = new ExtentSparkReporter( reportName );
13        // htmlReporter = new ExtentSparkReporter("demoblaze_EcommerceApp.html"); // Creating HTML report
14        extent = new ExtentReports(); // Initializing the ExtentReports object
15        extent.attachReporter(htmlReporter); // Attaching reporter to the report
16
17        // Configuration of Report
18        htmlReporter.config().setDocumentTitle("My OHRM Report");
19        htmlReporter.config().setReportName("OHRMtest Project Report");
20        htmlReporter.config().setTheme(Theme.STANDARD);
21        htmlReporter.config().setTimeStampFormat("dd-MMM-yyyy");
22    }
23
24}

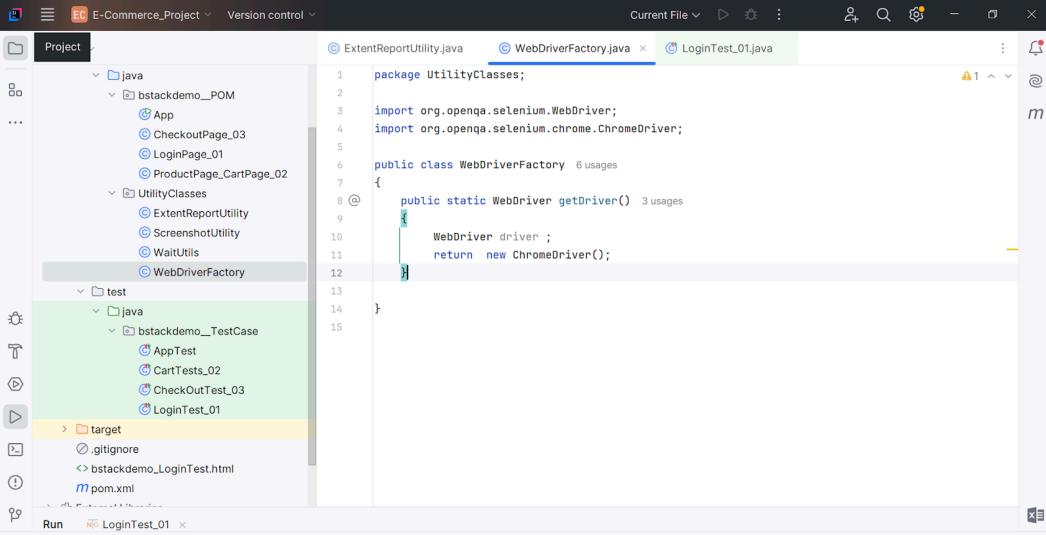
```

```

6 public class ExtentReportUtility 3 usages
11    public ExtentReportUtility(String reportName) 1 usage
27
28    // Add environment information
29    extent.setSystemInfo( k: "DeviceName", v: "Lenovo" );
30    extent.setSystemInfo( k: "Generation", v: "Core-i3-10Gen" );
31    extent.setSystemInfo( k: "Operating System", v: "Windows 10" );
32    extent.setSystemInfo( k: "Browser", v: "Chrome" );
33    extent.setSystemInfo( k: "TesterName", v: "Ritik Prajapati" );
34    extent.setSystemInfo( k: "Application Under Test ", v: "OrangeHRM " );
35
36
37    // Method to create new test case in report
38    public void createTest(String testName) { 3 usages
39        test = extent.createTest(testName);
40    }
41
42    // Final report save whatever we did so far in report
43    public void flushReport() 1 usage
44    {
45        extent.flush();
46    }
47
48}

```

UtilityClass: WebDriverFactory



```

1 package UtilityClasses;
2
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.chrome.ChromeDriver;
5
6 public class WebDriverFactory 6 usages
7 {
8     public static WebDriver getDriver() 3 usages
9     {
10        WebDriver driver ;
11        return new ChromeDriver();
12    }
13
14}

```

UtilityClass: WaitsUtils

```

package UtilityClasses;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;

public class WaitUtils {
    static WebDriverWait wait ; 2 usages
    public static void waitForElementVisible( By locator) { no usages
        wait.until(ExpectedConditions.visibilityOfElementLocated(locator));
    }
    public static void waitForElementClickable( By locator) { no usages
        wait.until(ExpectedConditions.elementToBeClickable(locator));
    }
    public static void setImplicitWait(WebDriver driver, int seconds) { 1 usage
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(seconds));
    }
}

```

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

```

package bstackdemo__POM;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class LoginPage_01 {
    WebDriver driver ; 3 usages
    public LoginPage_01(WebDriver driver) { 2 usages
        this.driver = driver;
        PageFactory.initElements(driver, page: this);
    }
    @FindBy(id="signin") WebElement signinBtn ; 1 usage
    @FindBy(xpath = "//*[@id='username']") WebElement usernameDropdown ; 1 usage
    @FindBy(xpath = "//div[text()='demouser']") WebElement selectDemouser; no usages
    @FindBy(id="password") WebElement passwordDropdown ; 1 usage
    @FindBy(xpath = "//div[text()='testingisfun99']") WebElement selectPassword; no usages
    @FindBy(id="login-btn") WebElement loginBtn ; 1 usage
}

```

```

public class LoginPage_01 {
    // Click signin button
    public void clickSignin() { 4 usages
        signinBtn.click();
    }

    // Select username from dropdown
    public void setUsername(String username) { 1 usage
        usernameDropdown.click();
        driver.findElement(By.xpath( xpathExpression: "//div[text()='" + username + "']").click(); //div[text() ='demouser']
    }

    // Select password from dropdown
    public void setPassword(String password) { 1 usage
        passwordDropdown.click();
        driver.findElement(By.xpath( xpathExpression: "//div[text()='" + password + "']").click(); //div[text() ='testingisfun99']
    }

    // Click login button
    public void clickLogin() { 1 usage
        loginBtn.click();
    }

    // Complete login method
    public void fulloLogin(String un, String pass) { 4 usages
        setUsername(un);
        setPassword(pass);
        clickLogin();
    }
}

```

LoginTests

Before-AfterTest/Method

```

public class LoginTest_01 {
    WebDriver driver; 8 usages
    LoginPage_01 login; 7 usages
    ExtentReportUtility reportutil ; 8 usages

    @BeforeTest
    public void reportsetup() {
        reportutil = new ExtentReportUtility( reportName: "bstackdemo_LoginTest.html" );
    }

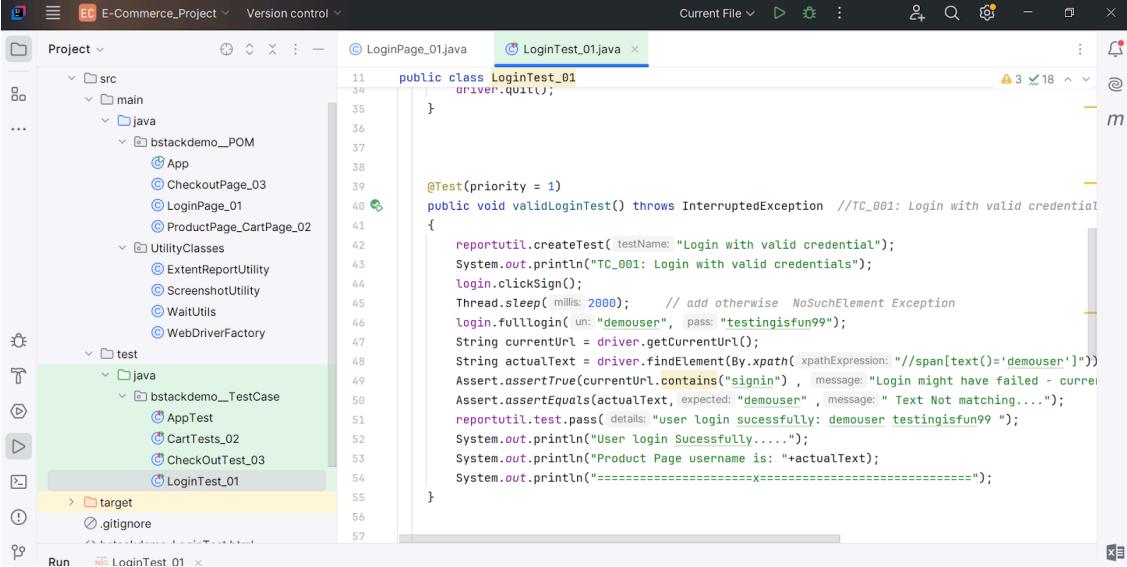
    @AfterTest
    public void reportsaving() {
        reportutil.flushReport();
    }

    @BeforeMethod
    public void setup() throws InterruptedException {
        driver = WebDriverFactory.getDriver();
        driver.manage().window().maximize();
        driver.get("https://bstackdemo.com/");
        WaitUtils.setImplicitWait(driver, seconds: 10); // add otherwise NoSuchElementException
        login = new LoginPage_01(driver);
    }

    @AfterMethod
    public void teardown() {
        driver.quit();
    }
}

```

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



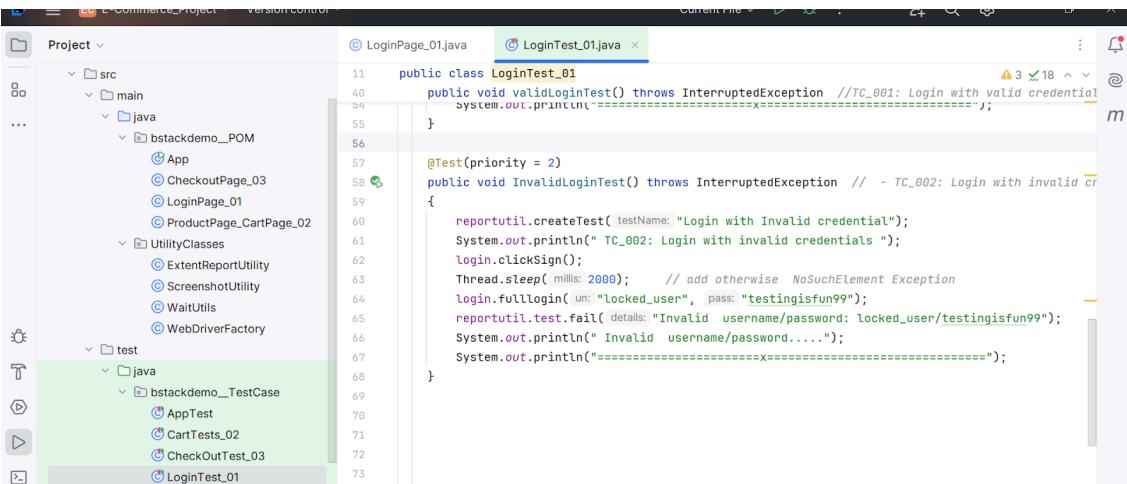
The screenshot shows the IntelliJ IDEA interface with the project 'E-Commerce_Project' open. The current file is 'LoginTest_01.java'. The code defines a test class 'LoginTest_01' with a single test method 'validLoginTest'. The test uses a driver to log in with the credentials 'demouser' and 'testingisfun99'. It asserts that the current URL contains 'signin' and that the actual text on the page matches the expected value 'demouser'. The code also includes reporting utility calls and prints to the console.

```

public class LoginTest_01 {
    @Test(priority = 1)
    public void validLoginTest() throws InterruptedException //TC_001: Login with valid credential
    {
        reportutil.createTest( testName: "Login with valid credential");
        System.out.println("TC_001: Login with valid credentials");
        login.clickSignIn();
        Thread.sleep( millis: 2000); // add otherwise NoSuchElementException
        login.fullLogin( uni: "demouser", pass: "testingisfun99");
        String currentUrl = driver.getCurrentUrl();
        String actualText = driver.findElement(By.xpath( xpathExpression: "//span[text()='demouser']"));
        Assert.assertTrue( currentUrl.contains("signin") , message: "Login might have failed - current url is $currentUrl");
        Assert.assertEquals(actualText, expected: "demouser" , message: " Text Not matching....");
        reportutil.test.pass( details: "user login sucessfully: demouser testingisfun99 ");
        System.out.println("User login Successfully....");
        System.out.println("Product Page username is: "+actualText);
        System.out.println("=====x=====");
    }
}

```

- TC_002: Login with invalid credentials



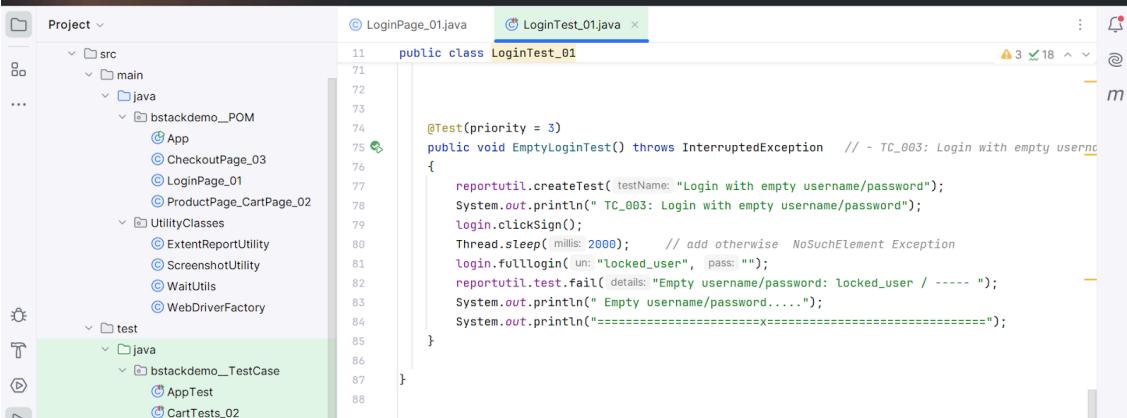
The screenshot shows the IntelliJ IDEA interface with the project 'E-Commerce_Project' open. The current file is 'LoginTest_01.java'. The code defines a test class 'LoginTest_01' with a new test method 'invalidLoginTest'. This test logs in with an invalid user 'locked_user' and an invalid password 'testingisfun99'. It fails the test and prints an error message to the console.

```

public class LoginTest_01 {
    @Test(priority = 2)
    public void InvalidLoginTest() throws InterruptedException // - TC_002: Login with invalid credentials
    {
        reportutil.createTest( testName: "Login with Invalid credential");
        System.out.println(" TC_002: Login with invalid credentials ");
        login.clickSignIn();
        Thread.sleep( millis: 2000); // add otherwise NoSuchElementException
        login.fullLogin( uni: "locked_user", pass: "testingisfun99");
        reportutil.test.fail( details: "Invalid username/password: locked_user/testingisfun99");
        System.out.println(" Invalid username/password.....");
        System.out.println("=====x=====");
    }
}

```

- TC_003: Login with empty fields



The screenshot shows the IntelliJ IDEA interface with the project 'E-Commerce_Project' open. The current file is 'LoginTest_01.java'. The code defines a test class 'LoginTest_01' with a new test method 'emptyLoginTest'. This test logs in with an empty username and an empty password. It fails the test and prints an error message to the console.

```

public class LoginTest_01 {
    @Test(priority = 3)
    public void EmptyLoginTest() throws InterruptedException // - TC_003: Login with empty username/password
    {
        reportutil.createTest( testName: "Login with empty username/password");
        System.out.println(" TC_003: Login with empty username/password");
        login.clickSignIn();
        Thread.sleep( millis: 2000); // add otherwise NoSuchElementException
        login.fullLogin( uni: "locked_user", pass: "");
        reportutil.test.fail( details: "Empty username/password: locked_user / ----- ");
        System.out.println(" Empty username/password.....");
        System.out.println("=====x=====");
    }
}

```

Screenshots

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

StackDemo

bstackdemo.com/signin

Chrome is being controlled by automated test software.

DEMO

BrowserStack

demouser

testingisfun99

LOG IN

StackDemo

bstackdemo.com/?signin=true

Chrome is being controlled by automated test software.

Type here to search

32°C

ENG IN

7/21/2025

Offers Orders Favourites Search demouser Logout

Shop smarter with our mobile e-commerce platform! Explore unbeatable deals, trending products, and seamless checkout—all at your fingertips. From fashion to electronics, find everything you need in one place. Enjoy fast delivery, secure payments, and 24/7 customer support. Download the app today and elevate your shopping experience like never before!"

Vendors: Apple Samsung Google OnePlus

25 Product(s) found.

Order by Select

https://bstackdemo.com

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

StackDemo

bstackdemo.com/signin

Chrome is being controlled by automated test software.

DEMO

BrowserStack

locked_user

testingisfun99

LOG IN

Your account has been locked.

StackDemo

bstackdemo.com/?signin=true

Chrome is being controlled by automated test software.

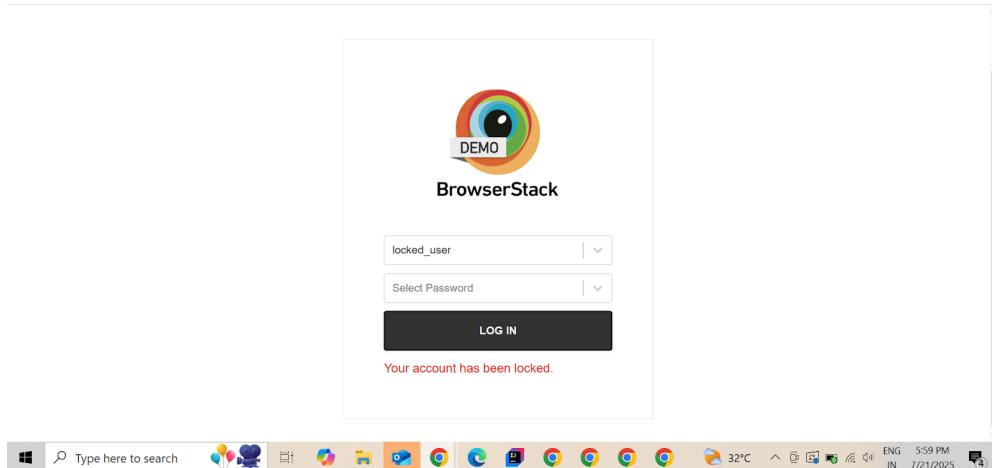
Type here to search

32°C

ENG IN

7/21/2025

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

```

1 import org.openqa.selenium.WebDriver;
2 import org.openqa.selenium.WebElement;
3 import org.openqa.selenium.support.FindBy;
4 import org.openqa.selenium.support.PageFactory;
5 import java.util.List;
6
7 public class ProductPage_CartPage_02 {
8
9     WebDriver driver; //usage
10    public ProductPage_CartPage_02(WebDriver driver) { // Constructor to initialize WebElements
11        this.driver = driver;
12        PageFactory.initElements(driver, this);
13    }
14
15    @FindBy(className = "checkboxmark") WebElement phonevendorApple; //usage
16    @FindBy(xpath = "//div[@class='shelf-item__buy-btn'][1]") WebElement addtocardBtn; //usage
17    @FindBy(xpath = "//div[@class='shelf-item__details']") WebElement productDetail; //usage
18    @FindBy(xpath = "//div[@class='sub-price']/p") WebElement productprice; //usage
19    @FindBy(xpath = "//div[@class='shelf-item__buy-btn']") WebElement addtoCart; //usage
20    @FindBy(xpath = "//div[@class='shelf-item__details']") List multiproductDetail; //usage
21    @FindBy(xpath = " //div[text()='Checkout']") WebElement checkoutBtn; //usage
22    @FindBy(xpath = "//div[@class='float-cart__close-btn']") WebElement clickonCross; // 2 usages
23    @FindBy(xpath = "//span[@class='bag bag--float-cart-closed']") WebElement clickonBasket; //usage
24    @FindBy(xpath = "(//div[@class='shelf-item__del'][1])") WebElement removesingleItem; //usage
25    @FindBy(xpath = "//p[@class='desc']") WebElement itemCount; //usage
26    @FindBy(xpath = "//span[@class='bag bag--float-cart-closed']") WebElement itemcounts; //usage
27
}

```

Screenshot 1: IDE View of ProductPage_CartPage_02.java

```

public class ProductPage_CartPage_02 {
    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 product = productDetail.getText();
        String price = productPrice.getText();
        System.out.println("Product details are:");
        System.out.println(product);
        System.out.println(price);
        clickonCross.click();
    }

    public void getItemCount() // 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);
    }
}

```

E-Commerce_Project > src > main > java > bstackdemo__POM > ProductPage_CartPage_02 > ItemCounts

Screenshot 2: IDE View of ProductPage_CartPage_02.java

```

public class ProductPage_CartPage_02 {
    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();
    }

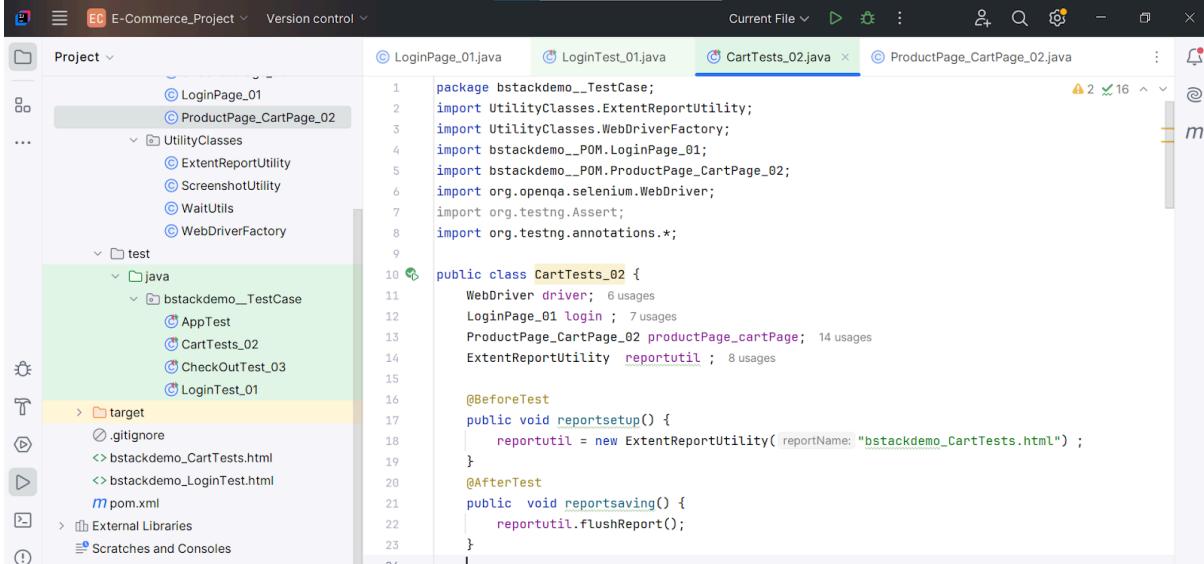
    // Add multiple products
    public void addMultipleProductsToCart(int count) throws InterruptedException { 2 usages
        for (int i = 1; i <=count; i++)
        {
            addProductToCart(i);
        }
    }
}

```

E-Commerce_Project > src > main > java > bstackdemo__POM > ProductPage_CartPage_02 > ItemCounts

CartTestCases:

Before-AfterTest/Method:



The screenshot shows the Eclipse IDE interface with the following details:

- Project View:** Shows the project structure with files like LoginPage_01, ProductPage_CartPage_02, UtilityClasses, and test.
- Current File:** CartTests_02.java is the active file.
- Code Editor:** The code for CartTests_02.java is displayed, including imports for ExtentReportUtility, WebDriverFactory, POM classes, and TestNG annotations.
- Code Snippet (Bottom):** A larger portion of the code is shown, starting with the setup() method and ending with the cleanup() method.

```

1 package bstackdemo__TestCase;
2 import UtilityClasses.ExtentReportUtility;
3 import UtilityClasses.WebDriverFactory;
4 import bstackdemo__POM.LoginPage_01;
5 import bstackdemo__POM.ProductPage_CartPage_02;
6 import org.openqa.selenium.WebDriver;
7 import org.testng.Assert;
8 import org.testng.annotations.*;
9
10 public class CartTests_02 {
11     WebDriver driver; 6 usages
12     LoginPage_01 login ; 7 usages
13     ProductPage_CartPage_02 productPage_cartPage; 14 usages
14     ExtentReportUtility reportutil ; 8 usages
15
16     @BeforeTest
17     public void reportsetup() {
18         reportutil = new ExtentReportUtility( reportName: "bstackdemo_CartTests.html" );
19     }
20     @AfterTest
21     public void reportsaving() {
22         reportutil.flushReport();
23     }
24
25     @BeforeMethod
26     public void setup() throws InterruptedException {
27         driver = WebDriverFactory.getDriver();
28         driver.manage().window().maximize();
29         driver.get("https://www.bstackdemo.com/");
30         Thread.sleep( millis: 3000 );
31         productPage_cartPage = new ProductPage_CartPage_02(driver);
32         login = new LoginPage_01(driver);
33     }
34     @AfterMethod
35     public void cleanup() {
36         driver.close();
37     }
38
39

```

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

The screenshot shows a Java IDE interface with several files open:

- LoginPage_01.java
- LoginTest_01.java
- CartTests_02.java (highlighted)
- ProductPage_CartPage_02.java

The CartTests_02.java file contains the following code:

```

public class CartTests_02 {
    // TC_004: Add single item to cart
    @Test(priority = 1)
    public void TC_004_AddSingleItemToCart() throws InterruptedException {
        reportutil.createTest( testName: "Add Single Item ToCart");
        System.out.println("TC_004: Add single item to cart ");
        login.clickSignIn();
        Thread.sleep( millis: 2000);
        login.fullLogin( un: "demouser", pass: "testingisfun99");
        Thread.sleep( millis: 2000);
        productPage_cartPage.appleVendor();
        productPage_cartPage.goToCart();
        productPage_cartPage.getCartItemCount();
        reportutil.test.pass( details: "Item added to card ");
        System.out.println("=====XXXXXX=====");
    }
}

```

Below the IDE is a screenshot of a web browser window titled "StackDemo". The URL is "bstackdemo.com/?signin=true". The page displays an e-commerce platform with a search bar and navigation links. A shopping cart icon is visible in the top right corner. The main content area shows a promotional message and a product listing for an iPhone 12.

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

The screenshot shows a Java IDE interface with several files open:

- LoginPage_01.java
- ProductPage_CartPage_02.java
- UtilityClasses
- ExtentReportUtility
- ScreenshotUtility
- WaitUtils
- WebDriverFactory
- bstackdemo_TestCase
- AppTest
- CartTests_02
- CheckOutTest_03
- LoginTest_01
- CartTests.html
- LoginTest.html
- Consoles

The CartTests_02.java file contains the following code:

```

public class CartTests_02 {
    // TC_005: Add mutiple item to cart
    @Test(priority = 2)
    public void TC005_AddmultipleitemToCart() throws InterruptedException {
        reportutil.createTest( testName: "Add multiple Item ToCart");
        System.out.println("TC_005: Add multiple items to cart and verify cart count ");
        login.clickSignIn();
        Thread.sleep( millis: 2000);
        login.fullLogin( un: "demouser", pass: "testingisfun99");
        Thread.sleep( millis: 2000);
        productPage_cartPage.appleVendor();
        productPage_cartPage.addMultipleProductsToCart( count: 3); // Add 3 products using helper method
        System.out.println("Total items added to cart :");
        productPage_cartPage.getmultipleItems();
        // Assert.assertTrue();

        productPage_cartPage.ItemCounts();
        reportutil.test.pass( details: " multiple Item added to card ");
        System.out.println("-----Testcase Passed-----");
        System.out.println("=====XXXXXX=====");
    }
}

// TC_006: Remove item from cart

```

The screenshot shows a web browser window with the URL bstackdemo.com/?signin=true. The page is titled "BrowserStack DEMO". A search bar and navigation links for "Offers", "Orders", and "Favourites" are visible. The main content is a shopping cart interface. It displays a list of products with their names, brands, quantities, and prices. The cart contains:

- iPhone 12 Mini (Apple) - Quantity: 1 - \$699.00
- iPhone 12 Pro Max (Apple) - Quantity: 1 - \$1099.00

The total subtotal is \$2797.00. A "CHECKOUT" button is present at the bottom right.

- **TC_006: Remove item from cart**

```

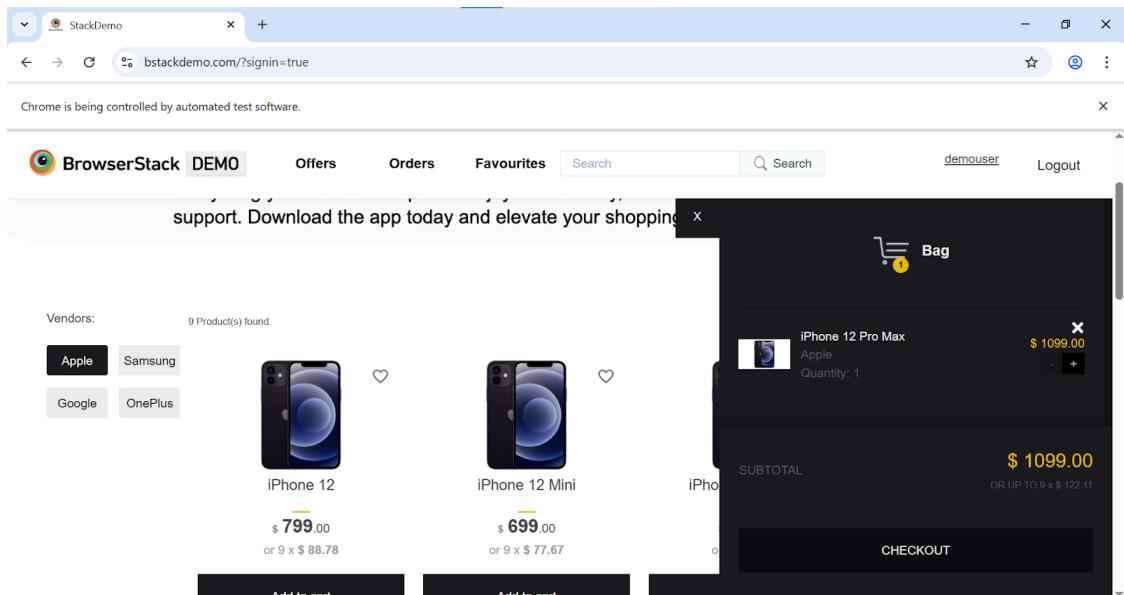
public class CartTests_02 {
    // TC_006: Remove item from cart
    @Test(priority = 3)
    public void TC006_removeitemToCart() throws InterruptedException {
        reportutil.createTest( testName: "remove Item ToCart");
        System.out.println("TC_006: Remove item from cart ");
        login.clickSignIn();
        Thread.sleep( millis: 2000 );
        login.fullLogin( uni: "demouser", pass: "testingisfun99");
        Thread.sleep( millis: 2000 );

        productPage_cartPage.appleVendor();
        productPage_cartPage.addMultipleProductsToCart( count: 2); // Add 3 products using helper method
        System.out.println("before removing Total items added to cart :");
        productPage_cartPage.getmultipleItems();
        productPage_cartPage.itemCounts();

        productPage_cartPage.removeItem();
        System.out.println(" after removing one items ,remaining items :");
        productPage_cartPage.getmultipleItems();
        reportutil.test.pass( details: " removed one item from card ");
        System.out.println("=====XXXXXXXX=====");
    }
}

```

The screenshot shows a web browser window with the URL bstackdemo.com/?signin=true. The page is titled "BrowserStack DEMO". The shopping cart now contains only the iPhone 12 Pro Max (Apple) at \$1099.00. The subtotal is \$1798.00. The "CHECKOUT" button is visible at the bottom right.



3. Checkout Tests

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

CheckOutPage: POM

```

package bstackdemo__POM;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import java.util.List;

public class CheckOutPage_03 {
    WebDriver driver; // 1 usage
    public CheckOutPage_03(WebDriver driver) { // 1 usage
        this.driver = driver;
        PageFactory.initElements(driver, this);
    }
    @FindBy(xpath="//span[@class='bag bag--float-cart-closed']") WebElement clickonBasket; // 1 usage
    @FindBy(xpath ="//span[@class='bag__quantity']") WebElement itemcounting; // for empty cart 1 usage
    // click on checkout button
    @FindBy(xpath = " //div[text()='Checkout']") WebElement checkoutBtn; // 3 usages
    // Form fields
    @FindBy(id = "firstNameInput") WebElement firstname; // 1 usage
    @FindBy(id = "lastNameInput") WebElement lastname; // 1 usage
    @FindBy(id = "addressLine1Input") WebElement addressInput; // 1 usage
    @FindBy(id = "postCodeInput") WebElement postalCodeInput; // 1 usage
    @FindBy(id = "checkout-shipping-continue") WebElement submitBtn; // 1 usage
    @FindBy(id="provinceInput") WebElement stateInput; // 1 usage
}

```

The image shows two side-by-side code editors from a Java IDE. Both editors have the same title bar: "E-Commerce_Project" and "Version control". The top editor displays the file "CheckoutPage_03.java" with the following code:

```

public class CheckoutPage_03 {
    @FindBy(xpath="//div[@class ='checkout-form']")
    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
        clickonBasket.click();
        System.out.println("items are :" + itemcouning.getText());
    }
}

```

The bottom editor displays the file "CheckoutPage_03.java" with the following code:

```

public class CheckoutPage_03 {
    // Fill checkout form
    public void fillCheckoutDetails(String fname, String lname ,String address, String state, String postalCode) { 1 us
        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](#)

The image shows a code editor with the title bar "E-Commerce_Project" and "Version control". The current file is "CheckOutTest_03.java". The code is as follows:

```

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();
    }
}

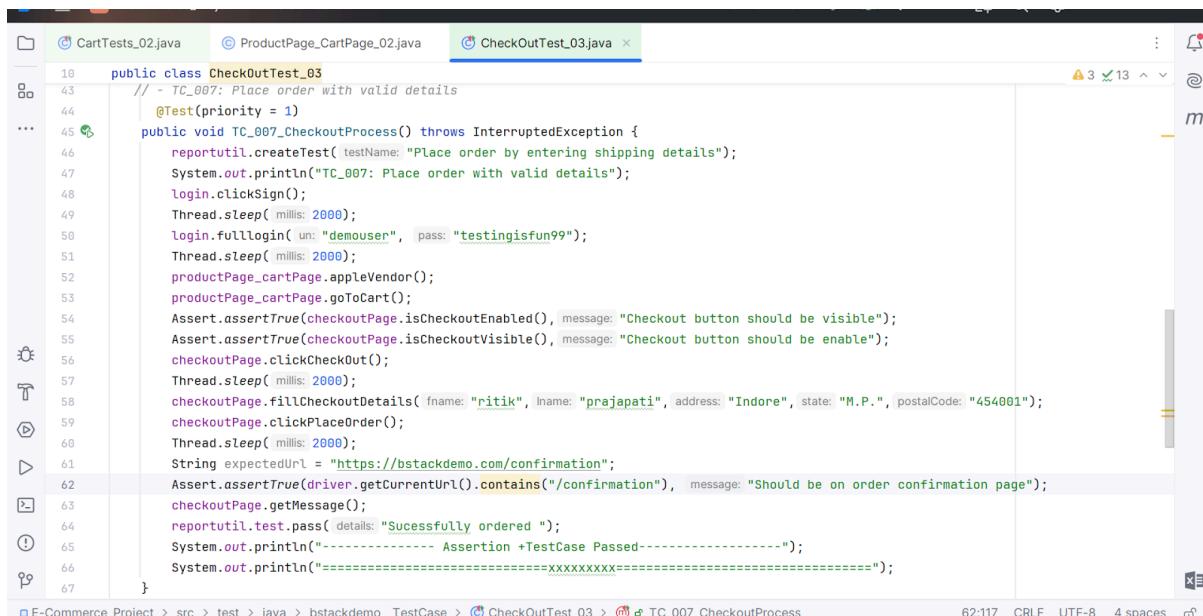
```

```

24
25
26
27
28     @BeforeMethod
29     public void setup() throws InterruptedException {
30         driver = WebDriverFactory.getDriver();
31         driver.manage().window().maximize();
32         driver.get("https://www.bstackdemo.com/");
33         Thread.sleep( millis: 3000 );
34
35         productPage_cartPage = new ProductPage_CartPage_02(driver);
36         login = new LoginPage_01(driver);
37         checkoutPage = new CheckoutPage_03(driver);
38     }
39
40     @AfterMethod
41     public void closeup() {
42         // driver.close();
43     }

```

TC_007: Place order with valid details

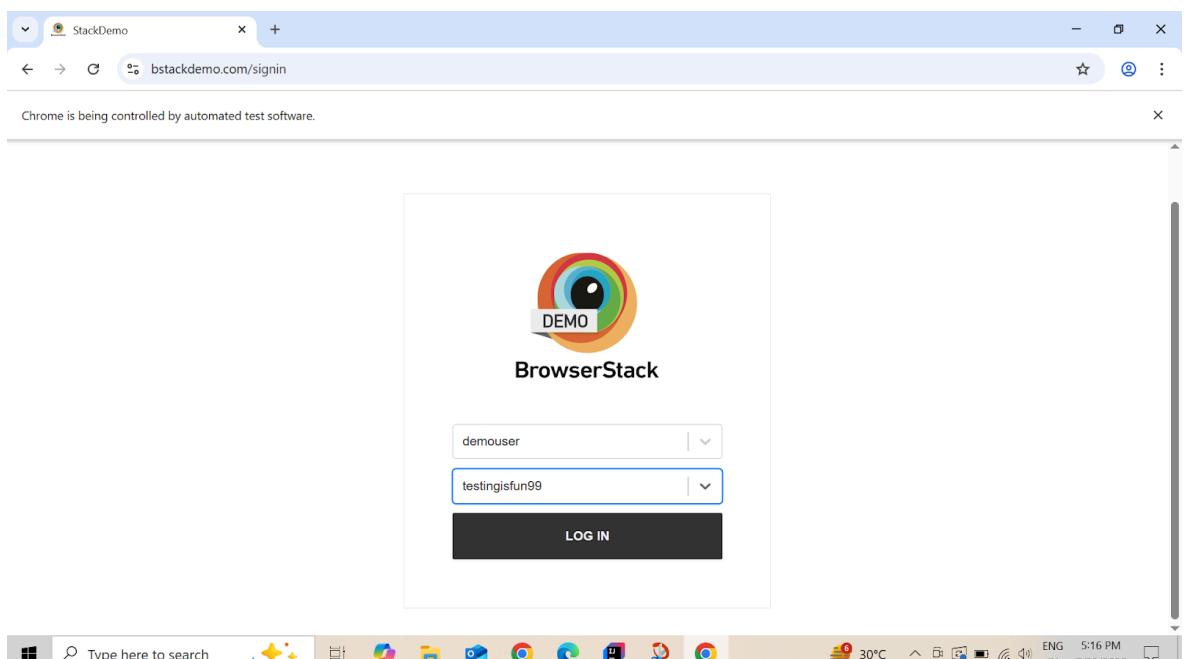


```

10  public class CheckOutTest_03
11      // - TC_007: Place order with valid details
12
13      @Test(priority = 1)
14      public void TC_007_CheckoutProcess() throws InterruptedException {
15          reportutil.createTest( testName: "Place order by entering shipping details");
16          System.out.println("TC_007: Place order with valid details");
17          login.clickSignIn();
18          Thread.sleep( millis: 2000 );
19          login.fullLogin( uni: "demouser", pass: "testingisfun99");
20          Thread.sleep( millis: 2000 );
21          productPage_cartPage.appleVendor();
22          productPage_cartPage.goToCart();
23          Assert.assertTrue(checkoutPage.isCheckoutEnabled(), message: "Checkout button should be visible");
24          Assert.assertTrue(checkoutPage.isCheckoutVisible(), message: "Checkout button should be enable");
25          checkoutPage.clickCheckOut();
26          Thread.sleep( millis: 2000 );
27          checkoutPage.fillCheckoutDetails( fname: "ritik", lname: "prajepati", address: "Indore", state: "M.P.", postalCode: "454001");
28          checkoutPage.clickPlaceOrder();
29          Thread.sleep( millis: 2000 );
30          String expectedUrl = "https://bstackdemo.com/confirmation";
31          Assert.assertTrue(driver.getCurrentUrl().contains("/confirmation"), message: "Should be on order confirmation page");
32          checkoutPage.getMessage();
33          reportutil.test.pass( details: "Sucessfully ordered ");
34          System.out.println("----- Assertion +TestCase Passed-----");
35          System.out.println("=====XXXXXXXXXX=====");
36      }

```

E-Commerce_Project > src > test > java > bstackdemo_TestCase > CheckOutTest_03 > TC_007_CheckoutProcess



StackDemo

bstackdemo.com/?signin=true

Chrome is being controlled by automated test software.

BrowserStack DEMO Offers Orders Favourites Search demouser Logout

Support. Download the app today and elevate your shopping!

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 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

Order Summary

1 item(s)

iPhone 12 Apple 1 \$799

Total (USD) **\$799.00**

SUBMIT

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

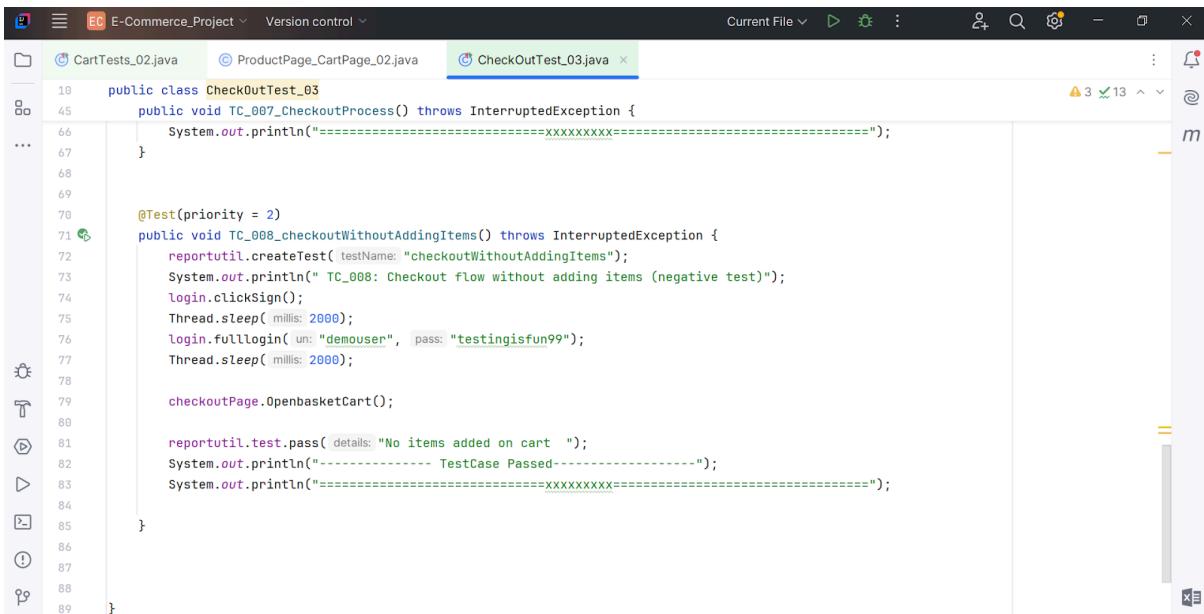
1 item(s)

iPhone 12 Apple 1 \$799

Total (USD) **\$799.00**

Type here to search 30°C ENG IN 5:16 PM 7/22/2025

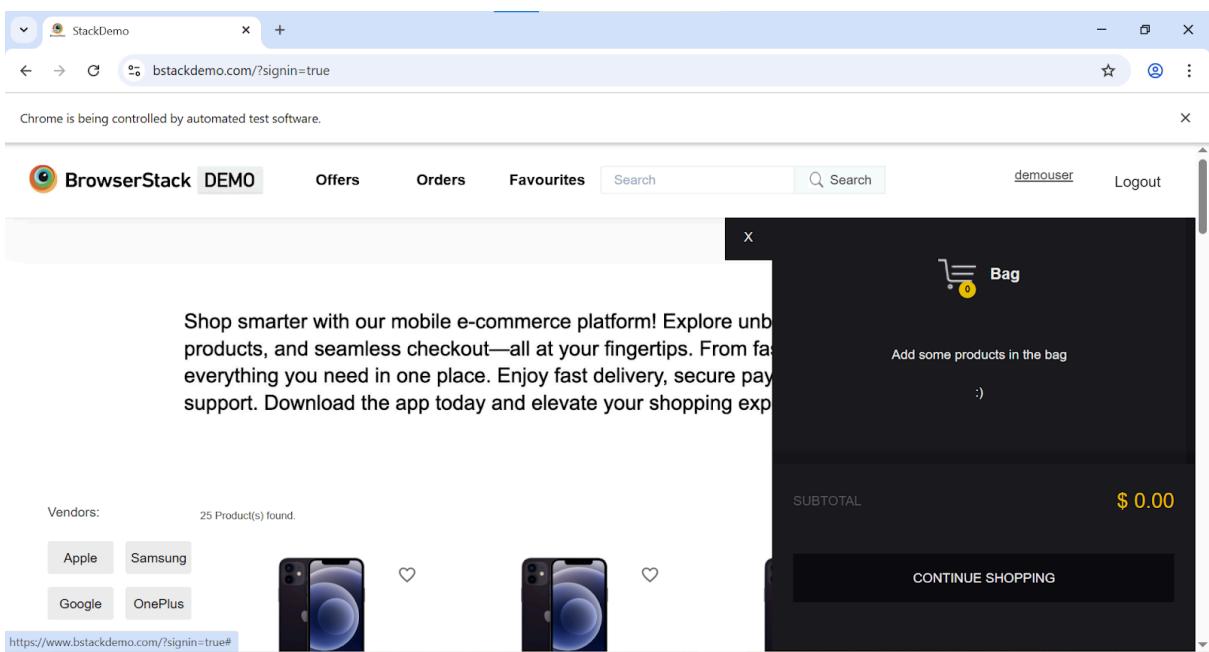
TC_008: Checkout flow without adding items (negative test)



```

10  public class CheckOutTest_03 {
11      public void TC_007_CheckoutProcess() throws InterruptedException {
12          System.out.println("=====XXXXXXXXX=====");
13
14          @Test(priority = 2)
15          public void TC_008_checkoutWithoutAddingItems() throws InterruptedException {
16              reportutil.createTest( testName: "checkoutWithoutAddingItems");
17              System.out.println(" TC_008: Checkout flow without adding items (negative test)");
18              login.clickSignin();
19              Thread.sleep( millis: 2000);
20              login.fulllogin( uni: "demouser", pass: "testingisfun99");
21              Thread.sleep( millis: 2000);
22
23              checkoutPage.OpenbasketCart();
24
25              reportutil.test.pass( details: "No items added on cart  ");
26              System.out.println("----- TestCase Passed-----");
27              System.out.println("=====XXXXXXXXX=====");
28
29          }
30      }
31  }

```



StackDemo

bstackdemo.com/?signin=true

Chrome is being controlled by automated test software.

BrowserStack DEMO Offers Orders Favourites Search demouser Logout

Shop smarter with our mobile e-commerce platform! Explore unb products, and seamless checkout—all at your fingertips. From fa everything you need in one place. Enjoy fast delivery, secure pay support. Download the app today and elevate your shopping exp

Vendors: 25 Product(s) found.

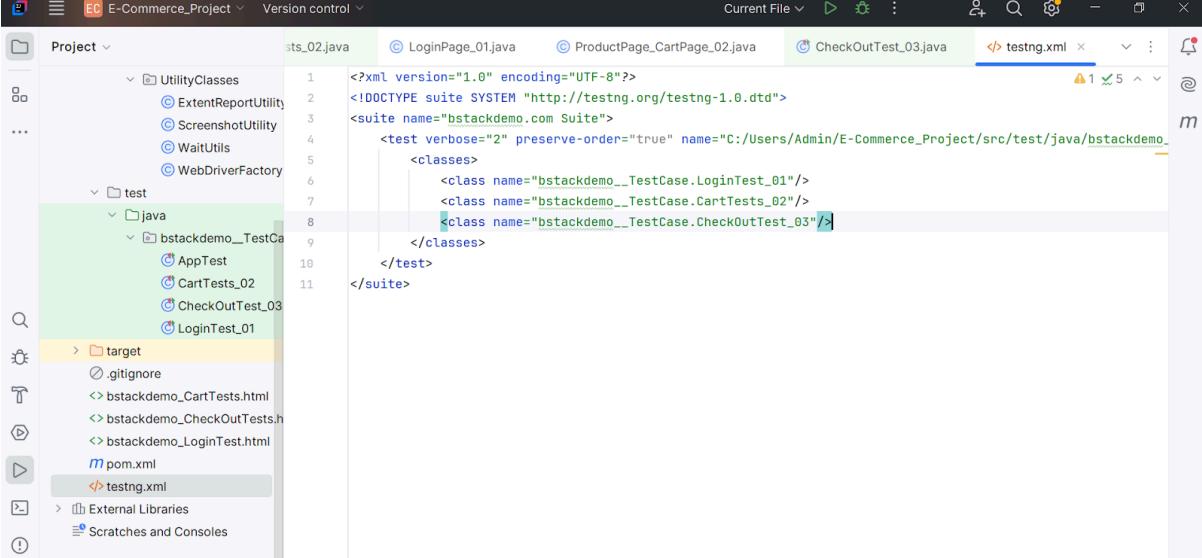
Apple Samsung
Google OnePlus

CONTINUE SHOPPING

https://www.bstackdemo.com/?signin=true#

Testcase Execution:

Using TestNG XML



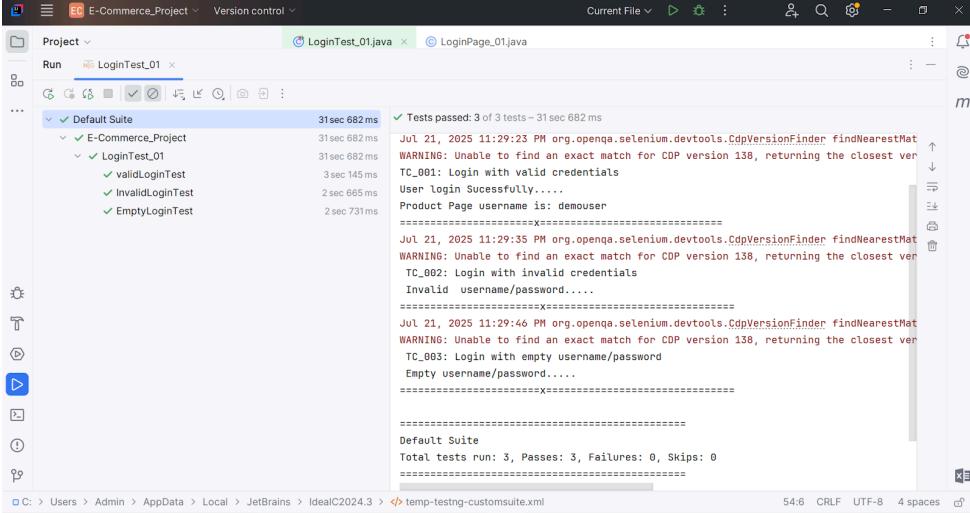
The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The left sidebar shows the project structure under "E-Commerce_Project". It includes a "UtilityClasses" package with several utility classes like ExtentReportUtility, ScreenshotUtility, WaitUtils, and WebDriverFactory. Below it is a "test" package containing a "java" folder with "AppTest", "CartTests_02", "CheckOutTest_03", and "LoginTest_01" files.
- Current File:** The main editor window displays the content of "testng.xml". The XML code defines a suite named "bstackdemo.com Suite" with three tests: "LoginTest_01", "CartTests_02", and "CheckOutTest_03".
- Toolbars and Status Bar:** The top bar includes standard IntelliJ icons for file operations. The status bar at the bottom shows the file path "C:/Users/Admin/E-Commerce_Project/src/test/java/bstackdemo/com/testng.xml" and file statistics: 54:6 CRLF, UTF-8, 4 spaces.

9 . Test Result Analysis & Reporting

1. Login Tests:

TestNG report:



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The left sidebar shows the project structure under "E-Commerce_Project". A "Run" configuration for "LoginTest_01" is selected.
- Current File:** The main editor window displays the execution results for "LoginTest_01". The results show 3 tests passed in 31 sec 682 ms. The log output includes test cases like "TC_001: Login with valid credentials", "TC_002: Login with invalid credentials", and "TC_003: Login with empty username/password".
- Toolbars and Status Bar:** The top bar includes standard IntelliJ icons. The status bar at the bottom shows the file path "C:/Users/Admin/AppData/Local/JetBrains/IdeaIC2024.3/testng.xml" and file statistics: 54:6 CRLF, UTF-8, 4 spaces.

ExtentReport:

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

2. Cart Tests:

TestNG report:

```

Run: CartTests_02
...
Default Suite
  ✓ E-Commerce_Project
    ✓ CartTests_02
      ✓ TC_004_AddSingleItemToCart
        Tests passed: 1 of 1 test - 18 sec 51 ms
        JUL 22, 2025 1:58:30 PM org.openqa.selenium.devtools.LogVersionFinder findNearestMatch
        WARNING: Unable to find an exact match for CDP version 138, returning the closest ver
        TC_004: Add single item to cart
        Product details are:
        iPhone 12
        Apple
        Quantity: 1
        $ 799.00
        single item count is: Apple
        Quantity: 1
        =====XXXXXX=====
        =====
        Default Suite
        Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
        =====
        Process finished with exit code 0
  
```

C: > Users > Admin > AppData > Local > JetBrains > IdeaIC2024.3 > temp-testing-customsuite.xml 38:28 (55 chars, 1 line break) CRLF UTF-8 4 spaces

Screenshot 1: Run Configuration - CartTests_02

```

    ✓ 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 findNearestMatch
    WARNING: Unable to find an exact match for CDP version 138, returning the closest version
    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-----
    =====XXXXXXXX=====
    =====
    Default Suite
  
```

Screenshot 2: Structure View - CartTests_02

```

    ✓ Tests passed: 1 of 1 test – 23 sec 291 ms
    WARNING: UNABLE TO FIND AN EXACT MATCH FOR CDP VERSION 138, RETURNING THE CLOSEST VERSION
    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
    =====XXXXXXXX=====
    =====
    Default Suite
  
```

Screenshot 3: Run Configuration - CartTests_02

```

    ✓ 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
    =====XXXXXXXX=====

    =====
    Default Suite
    Total tests run: 3, Passes: 3, Failures: 0, Skips: 0
    =====

    Process finished with exit code 0
  
```

ExtentReport:

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

The ExtentReport interface displays test results and a summary dashboard.

Test Log (Top Screenshot):

- Test: Add Single Item ToCart
- Status: Pass
- Timestamp: 08:09:52 PM / 00:00:05:108
- Details: Item added to card

Summary Dashboard (Bottom Screenshot):

- Started: 22-July-2025
- Ended: 22-July-2025
- Tests Passed: 3
- Tests Failed: 0

3. Checkout Tests:

TestNG report:

The TestNG report in IntelliJ IDEA shows the execution of the 'CheckOutTest_03' test case in the 'Default Suite'.

Test Execution Details:

- Run: CheckOutTest_03
- Suite: Default Suite
- Test: E-Commerce_Project > CheckOutTest_03 > TC_007_CheckoutProcess
- Time: 17 sec 793 ms
- Status: Tests passed: 1 of 1 test – 17 sec 793 ms

Test Log Output:

```

SLF4J(W): See https://www.slf4j.org/codes.html#noProviders for further details.
Jul 22, 2025 5:18:51 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 138, returning the closest version; found
TC_007: Place order with valid details
Product details are:
iPhone 12
Apple
Quantity: 1
$ 799.00
OrderMessage is: Your Order has been successfully placed.
Your order number is 23.
Download order receipt
----- Assertion +TestCase Passed-----
=====XXXXXXXXX=====
Default Suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
=====
```

```

Current File > CheckOutTest_03.java
Run > CheckOutTest_03

Default Suite
  ✓ E-Commerce_Project 17 sec 892 ms
    ✓ CheckOutTest_03 17 sec 892 ms
      ✓ TC_008_checkoutWithoutAd 4 sec 821ms

Tests passed: 1 of 1 test – 17 sec 892 ms

"C:\Program Files\Java\jdk-23\bin\java.exe" ...
SLF4J(W): No SLF4J providers were found.
SLF4J(W): Defaulting to no-operation (NOP) logger implementation
SLF4J(W): See https://www.slf4j.org/codes.html#noProviders for further details.
Jul 22, 2025 5:56:20 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 138, returning the closest version; found
TC_008: Checkout flow without adding items (negative test)
items are :0
----- TestCase Passed -----
=====XXXXXXXXX=====

Default Suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
=====

Process finished with exit code 0

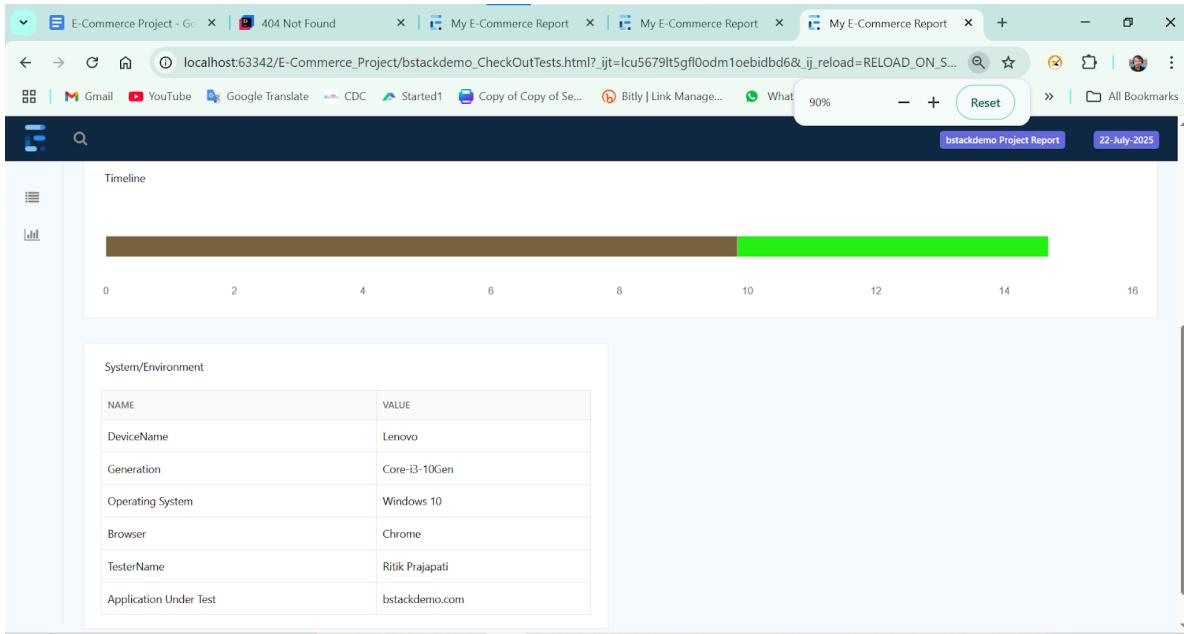
```

ExtentReport:

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

| Test Step | Status | Timestamp | Details |
|--|--------|-----------------------|---|
| Place order by entering shipping details | Pass | 07.22.2025 8:14:09 PM | 07.22.2025 8:14:19 PM 00:00:09:844 #test-id=1 |
| checkoutWithoutAddingItems | Pass | 8:14:34PM | 00:00:04:844 |

| Category | Value |
|--------------|--------------|
| Started | 22-July-2025 |
| Ended | 22-July-2025 |
| Tests Passed | 2 |
| Tests Failed | 0 |



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 -----

