WebDriverFactory.java:

```
package com.bstack.utils;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.edge.EdgeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class WebDriverFactory {
	public static WebDriver createDriver(String browser) {
		WebDriver driver;
		switch (browser.toLowerCase()) {
		case "chrome":
			driver = new ChromeDriver();
			break;
		case "firefox":
			driver = new FirefoxDriver();
			break;
		case "edge":
			driver = new EdgeDriver();
			break;
		default:
			throw new RuntimeException("Unsupported browser: " + browser);
		}
		return driver;
	}

}
```

ConfigReader.java:

```
package com.bstack.utils;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class ConfigReader {

	public static final long PAGE_LOAD_TIMEOUT = 20;
	public static final long IMPLICIT_WAIT = 20;
	public static final String TESTDATA_SHEET_PATH =
System.getProperty("user.dir") + File.separator + "src"
			+ File.separator + "main" + File.separator + "java" +
File.separator + "com" + File.separator + "bstack"
```

```
                    + File.separator + "testdata" + File.separator +
"LoginData.xlsx";

	static XSSFWorkbook book;
	static XSSFSheet sheet;
	static XSSFCell cell;
	static XSSFRow row;

	public static Object[][] getTestData(String sheetName) throws IOException {

		FileInputStream file = new FileInputStream(TESTDATA_SHEET_PATH);

		book = new XSSFWorkbook(file);

		sheet = book.getSheet(sheetName);

		Object[][] data = new
Object[sheet.getLastRowNum()][sheet.getRow(0).getLastCellNum()];
		System.out.println(sheet.getLastRowNum());

		for (int i = 0; i < sheet.getLastRowNum(); i++) {
			row = sheet.getRow(i + 1);
			if (row != null) {
				for (int k = 0; k < sheet.getRow(0).getLastCellNum(); k++)
{
					cell = row.getCell(k);
					if (cell != null) {
						data[i][k] = cell.toString();
					} else {
						data[i][k] = "";
					}
				}
			}

		}
		return data;
	}

}
```

WaitUtils.java:

```
package com.bstack.utils;

import java.io.File;
import  java.io.IOException;
import java.text.SimpleDateFormat;
import java.time.Duration;
import java.util.Date;

import org.openqa.selenium.By;
```

```
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.io.FileHandler;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

public class WaitUtils {

	public static WebElement waitForVisibility(WebDriver driver, By element, int
timeout) {
		return new WebDriverWait(driver, Duration.ofSeconds(timeout))

	.until(ExpectedConditions.visibilityOfElementLocated(element));
	}

	public static void waitForCartCount(WebDriver driver, int expectedCount) {
		WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
		By cartCount = By.cssSelector(".bag__quantity");

		wait.until(ExpectedConditions.presenceOfElementLocated(cartCount));
		wait.until(ExpectedConditions.textToBe(cartCount,
String.valueOf(expectedCount)));
	}

	public static String captureScreenshot(WebDriver driver, String testName,
String browser) {
		String timestamp = new SimpleDateFormat("yyyyMMddHHmmss").format(new
Date());

		String path = "Screenshots/" + browser + "_" + testName + "_" +
timestamp + ".png";

		File src = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);
		File dest = new File(path);
		try {
			FileHandler.copy(src, dest);
		} catch (IOException e) {
			e.printStackTrace();
		}
		return path;
	}
}
```

TestBase.java:

```
package com.bstack.base;

import java.time.Duration;
```

```java
import org.openqa.selenium.WebDriver;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Parameters;

import com.aventstack.extentreports.ExtentTest;
import com.aventstack.extentreports.MediaEntityBuilder;
import com.aventstack.extentreports.Status;
import com.bstack.utils.WaitUtils;
import com.bstack.utils.WebDriverFactory;

public class TestBase {
	public static ExtentTest logger;
	public static WebDriver driver;
	private static ThreadLocal<String> threadBrowser = new ThreadLocal<>();

	public static String getBrowser() {
		return threadBrowser.get();
	}

	@BeforeMethod
	@Parameters("browser")
	public void setUp(String browser) {
		threadBrowser.set(browser);
		driver = WebDriverFactory.createDriver(browser);
		driver.manage().window().maximize();
		driver.get("https://bstackdemo.com");
		driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
	}

	@AfterMethod
	public void tearDown() {
		if (driver != null) {
			driver.quit();
		}
		threadBrowser.remove();
	}

	protected void logWithScreenshot(String message) {

		String testName = message.replace(" ", "_");

		String browser = getBrowser();

		String path = WaitUtils.captureScreenshot(driver, testName, browser);

		if (logger != null) {

			logger.log(Status.INFO, message);

			logger.info("🖼 Click to Open Screenshot",
MediaEntityBuilder.createScreenCaptureFromPath(path).build());

		} else {
```

```
                    System.out.println("⚠ Logger is null. Screenshot saved at: " +
path);

            }
      }

}
```

ExtentTestListener.java:

```
package com.bstack.extentreportlistener;

import java.text.SimpleDateFormat;
import java.util.Date;

import org.openqa.selenium.WebDriver;
import org.testng.ISuite;
import org.testng.ISuiteListener;
import org.testng.ITestListener;
import org.testng.ITestResult;

import com.aventstack.extentreports.ExtentReports;
import com.aventstack.extentreports.ExtentTest;
import com.aventstack.extentreports.MediaEntityBuilder;
import com.aventstack.extentreports.reporter.ExtentSparkReporter;
import com.bstack.base.TestBase;
import com.bstack.utils.WaitUtils;

public class ExtentTestListener implements ITestListener, ISuiteListener {

      private static ExtentReports extent;
      private static ExtentTest test;

      @Override
      public void onStart(ISuite suite) {
            ExtentSparkReporter spark = new
ExtentSparkReporter("ExtentReport.html");
            extent = new ExtentReports();
            extent.attachReporter(spark);
            extent.setSystemInfo("Suite", suite.getName());

      }

      @Override
      public void onFinish(ISuite suite) {
            extent.flush();
      }

      @Override
```

```
	public void onTestStart(ITestResult result) {
		test = extent.createTest(result.getMethod().getMethodName());
		TestBase.logger = test;

		String browser = TestBase.getBrowser();

		test.info("Browser Used: " + browser);
	}

	@Override
	public void onTestSuccess(ITestResult result) {
		test.pass("Test passed");
	}

	@Override
	public void onTestFailure(ITestResult result) {

		test.fail("Test failed: " + result.getThrowable());

		WebDriver driver = TestBase.driver;

		String timestamp = new SimpleDateFormat("yyyyMMdd_HHmmss").format(new
Date());

		String screenshotName = result.getName() + "_" + timestamp;

		String browser = TestBase.getBrowser();

		String filePath = WaitUtils.captureScreenshot(driver, screenshotName,
browser);

		test.fail("📸 Screenshot on failure",
MediaEntityBuilder.createScreenCaptureFromPath(filePath).build());

	}

	@Override
	public void onTestSkipped(ITestResult result) {
		test.skip("Test skipped");
	}

}
```

LoginPage.java:

```
package com.bstack.pages;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;

import com.bstack.utils.WaitUtils;
```

```
public class LoginPage {
	WebDriver driver;

	public By signInLink = By.id("signin");

	public By userNameDropdown = By
			.xpath("//div[@id='username']//div[contains(@class,'css-tlfecz-
indicatorContainer')]");

	public By passwordDropdown = By
			.xpath("//div[@id='password']//div[contains(@class,'css-tlfecz-
indicatorContainer')]");

	private By loginButton = By.xpath("//button[@id='login-btn']");

	public By loggedInUserName = By.xpath("//span[@class='username']");

	public By usernameFieldLocator = By.id("username");

	public LoginPage(WebDriver driver) {
		this.driver = driver;
	}

	public void login(String username, String password) {

		driver.findElement(userNameDropdown).click();

		By usernameOption = By.xpath("//div[@id='username']//div[text()='" +
username + "']");

		WaitUtils.waitForVisibility(driver, usernameOption, 10);

		driver.findElement(usernameOption).click();

		driver.findElement(passwordDropdown).click();

		By passwordOption = By.xpath("//div[@id='password']//div[text()='" +
password + "']");

		WaitUtils.waitForVisibility(driver, passwordOption, 10);

		driver.findElement(passwordOption).click();

		driver.findElement(loginButton).click();
	}

	public boolean isLoginSuccessful() {

		WaitUtils.waitForVisibility(driver, loggedInUserName, 10);

		return driver.findElement(loggedInUserName).isDisplayed();
	}
}
```

ProductPage.java:

```
package com.bstack.pages;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

import java.util.List;

public class ProductPage {
	WebDriver driver;

	private By addToCartButtons = By.cssSelector(".shelf-item__buy-btn");

	public ProductPage(WebDriver driver) {
		this.driver = driver;
	}

	public void addFirstProductToCart() {
		List<WebElement> buttons = driver.findElements(addToCartButtons);
		if (!buttons.isEmpty()) {
			buttons.get(0).click();
		}
	}

	public void addMultipleProductsToCart(int count) {
		List<WebElement> buttons = driver.findElements(addToCartButtons);
		System.out.println("Number of add-to-cart buttons: " + buttons.size());
		for (int i = 0; i < count && i < buttons.size(); i++) {
			buttons.get(i).click();
		}
	}
}
```

CartPage.java:

```
package com.bstack.pages;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

public class CartPage {
	WebDriver driver;

	private By cartCount = By.cssSelector(".bag__quantity");
	private By cartItems = By.cssSelector(".float-cart__shelf-container > div");
	private By removeButtons = By.cssSelector(".shelf-item__del");
```

```
	public CartPage(WebDriver driver) {
		this.driver = driver;
	}

	public String getCartItemCount() {
		return driver.findElement(cartCount).getText();
	}

	public int getTotalItemsInCart() {
		return driver.findElements(cartItems).size();
	}

	public void removeFirstItemFromCart() {
		List<WebElement> buttons = driver.findElements(removeButtons);
		if (!buttons.isEmpty()) {
			buttons.get(0).click();
		}
	}
}
```

CheckoutPage.java:

```
package com.bstack.pages;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

public class CheckoutPage {
    WebDriver driver;

    public By checkoutButton = By.cssSelector(".buy-btn");
    private By firstName = By.id("firstNameInput");
    private By lastName = By.id("lastNameInput");
    private By address = By.id("addressLine1Input");
    private By state = By.id("provinceInput");
    private By postalCode = By.id("postCodeInput");
    private By submitButton = By.id("checkout-shipping-continue");
    private By orderConfirmation = By.id("confirmation-message");
    public static By cartButton = By.xpath("//span[@class='bag bag--float-cart-
closed']");
    public By continueShoppingButton = By.xpath("//div[@class='buy-btn']");

    public CheckoutPage(WebDriver driver) {
        this.driver = driver;
    }

    public void clickCheckout() {
        driver.findElement(checkoutButton).click();
    }
```

```
    public void fillShippingFormAndSubmit(String fname, String lname, String addr,
String st, String pin) {
        driver.findElement(firstName).sendKeys(fname);
        driver.findElement(lastName).sendKeys(lname);
        driver.findElement(address).sendKeys(addr);
        driver.findElement(state).sendKeys(st);
        driver.findElement(postalCode).sendKeys(pin);
        driver.findElement(submitButton).click();
    }

    public boolean isOrderConfirmed() {
        WebElement confirmation = driver.findElement(orderConfirmation);
        return confirmation.isDisplayed();
    }
}
```

LoginTest.java:

```
package com.bstack.tests;

import org.testng.Assert;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;

import com.bstack.base.TestBase;
import com.bstack.pages.LoginPage;
import com.bstack.utils.ConfigReader;


public class LoginTest extends TestBase {
	String sheetName = "Logindata";

	@Test(dataProvider = "LoginData")

	public void loginTestCases(String username, String password) throws
InterruptedException {
		LoginPage loginPage = new LoginPage(driver);

		driver.findElement(loginPage.signInLink).click();

		loginPage.login(username, password);

		Assert.assertTrue(loginPage.isLoginSuccessful(),
				"Login failed with username: '" + username + "' and
password: '" + password + "'");

		Thread.sleep(1000);

		logWithScreenshot("Signin Successful with valid credentials.");

	}

	@DataProvider(name = "LoginData")
```

```
	public Object[][] getData() throws Exception {
		Object data[][] = ConfigReader.getTestData(sheetName);

		return data;
	}

}
```

AddToCartTest.java:

```
package com.bstack.tests;

import org.testng.Assert;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

import com.bstack.base.TestBase;
import com.bstack.pages.CartPage;
import com.bstack.pages.LoginPage;
import com.bstack.pages.ProductPage;
import com.bstack.utils.WaitUtils;


public class AddToCartTest extends TestBase {
	LoginPage loginPage;
	ProductPage productPage;
	CartPage cartPage;

	@BeforeMethod
	public void initPagesAndLogin() throws InterruptedException {

		loginPage = new LoginPage(driver);
		productPage = new ProductPage(driver);
		cartPage = new CartPage(driver);

		driver.findElement(loginPage.signInLink).click();

		loginPage.login("demouser", "testingisfun99");

		WaitUtils.waitForVisibility(driver, loginPage.usernameFieldLocator, 10);

	}

	@Test(priority = 1)

	public void addSingleItemToCartTest() throws InterruptedException {
		productPage.addFirstProductToCart();
		WaitUtils.waitForCartCount(driver, 1);
	    Assert.assertEquals(Integer.parseInt(cartPage.getCartItemCount()), 1,
"Cart count mismatch after adding one item.");
		logWithScreenshot("Single item added to cart");

	}
```

```
	@Test(priority = 2)

	public void addMultipleItemsToCartTest() throws InterruptedException {
		productPage.addMultipleProductsToCart(3);
		WaitUtils.waitForCartCount(driver, 3);
	    Assert.assertEquals(Integer.parseInt(cartPage.getCartItemCount()), 3,
"Cart count mismatch after adding one item.");
		logWithScreenshot("Multiple items added to cart");

	}

	@Test(priority = 3)

	public void removeItemFromCartTest() throws InterruptedException {
		productPage.addMultipleProductsToCart(2);
		WaitUtils.waitForCartCount(driver, 2);
		int before = cartPage.getTotalItemsInCart();
		logWithScreenshot("2 items added to cart");
		cartPage.removeFirstItemFromCart();
		WaitUtils.waitForCartCount(driver, before - 1);
		logWithScreenshot("1 item removed from the cart");
		int after = cartPage.getTotalItemsInCart();
		Assert.assertEquals(after, before - 1);
		logWithScreenshot("1 item left in the cart");

	}
}
```

CheckOutTest.java:

```
package com.bstack.tests;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.testng.Assert;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

import com.bstack.base.TestBase;
import com.bstack.pages.CartPage;
import com.bstack.pages.CheckoutPage;
import com.bstack.pages.LoginPage;
import com.bstack.pages.ProductPage;
import com.bstack.utils.WaitUtils;


public class CheckOutTest extends TestBase {
	LoginPage loginPage;
	ProductPage productPage;
	CartPage cartPage;
	CheckoutPage checkoutPage;
```

```
	@BeforeMethod
	public void initPagesAndLogin() throws InterruptedException {
		loginPage = new LoginPage(driver);
		productPage = new ProductPage(driver);
		cartPage = new CartPage(driver);
		checkoutPage = new CheckoutPage(driver);

		driver.findElement(loginPage.signInLink).click();

		loginPage.login("demouser", "testingisfun99");

		WaitUtils.waitForVisibility(driver, loginPage.usernameFieldLocator, 10);

	}

	@Test(priority = 1)

	public void placeOrderWithValidDetails() throws InterruptedException {

		productPage.addFirstProductToCart();
		WaitUtils.waitForCartCount(driver, 1);
		checkoutPage.clickCheckout();
		checkoutPage.fillShippingFormAndSubmit("Chandu", "Kola", "MVP", "AP",
"530017");
		Thread.sleep(2000);
		Assert.assertTrue(checkoutPage.isOrderConfirmed(), "Order confirmation
not displayed!");
		logWithScreenshot("Order Confirmed");

	}

	@Test(priority = 2)

	public void checkoutWithoutItemsNegativeTest() throws InterruptedException {

		driver.findElement(checkoutPage.cartButton).click();

		Thread.sleep(3000);

		String emptyCartMessage =
driver.findElement(By.xpath("//p[@class='shelf-empty']")).getText();

		List<WebElement> productsInCart = driver
				.findElements(By.xpath("//div[@class='float-cart__shelf-
container']//div[@class='shelf-item']"));

		if (productsInCart.isEmpty() && emptyCartMessage.contains("Add some
products in the bag")) {

			Assert.assertTrue(emptyCartMessage.contains("Add some products in
the bag"),
					"Empty cart message not displayed correctly!");

			logWithScreenshot("Cart is empty and the message displaying as -
Add some products in the bag.");
```

```
			}

			else {
				Assert.fail("Either the cart is not empty or the expected empty
cart message was not found.");
			}
		}
}
```

Pom.xml:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<groupId>com.bstackdemo</groupId>
	<artifactId>BrowserStackEcommerce_Automation</artifactId>
	<version>0.0.1-SNAPSHOT</version>

	<build>
		<plugins>
			<plugin>
				<groupId>org.apache.maven.plugins</groupId>
				<artifactId>maven-compiler-plugin</artifactId>
				<version>3.11.0</version>
				<configuration>
					<release>21</release>
				</configuration>
			</plugin>

			<plugin>
				<groupId>org.apache.maven.plugins</groupId>
				<artifactId>maven-surefire-plugin</artifactId>
				<version>3.1.2</version>
				<configuration>
					<suiteXmlFiles>
						<suiteXmlFile>TestNG.xml</suiteXmlFile>
					</suiteXmlFiles>
				</configuration>
			</plugin>

		</plugins>
	</build>
	<dependencies>
		<!--
		https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java
-->
		<dependency>
			<groupId>org.seleniumhq.selenium</groupId>
			<artifactId>selenium-java</artifactId>
			<version>4.33.0</version>
```

```
		</dependency>

		<dependency>
			<groupId>org.testng</groupId>
			<artifactId>testng</artifactId>
			<version>7.11.0</version>
		</dependency>


		<dependency>
			<groupId>org.apache.poi</groupId>
			<artifactId>poi</artifactId>
			<version>5.4.0</version>
		</dependency>

		<dependency>
			<groupId>org.apache.poi</groupId>
			<artifactId>poi-ooxml</artifactId>
			<version>5.4.0</version>
		</dependency>

		<dependency>
			<groupId>com.aventstack</groupId>
			<artifactId>extentreports</artifactId>
			<version>5.1.2</version>
		</dependency>
	</dependencies>

</project>
```

TestNG.xml:

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="BrowserStackDemo E-Commerce Test Suite" parallel="false">

	<listeners>
		<listener
			class-name="com.bstack.extentreportlistener.ExtentTestListener"
/>
	</listeners>

	<test name="Chrome Tests">
		<parameter name="browser" value="chrome" />
		<classes>
			<class name="com.bstack.tests.LoginTest" />
			<class name="com.bstack.tests.AddToCartTest" />
			<class name="com.bstack.tests.CheckOutTest" />
		</classes>
	</test>

	<test name="Firefox Tests">
		<parameter name="browser" value="firefox" />
		<classes>
```

```
                <class name="com.bstack.tests.LoginTest" />
                <class name="com.bstack.tests.AddToCartTest" />
                <class name="com.bstack.tests.CheckOutTest" />
            </classes>
        </test>
</suite>
```

GitHub repo: https://github.com/Dilip2012/BrowserStack_E-commerce

📁 Project Structure

```
│
├── src/main/java/
│   ├── com.bstack.base/ # TestBase setup
│   ├── com.bstack.extentreportlistener/ # ExtentReport listener
│   ├── com.bstack.pages/ # Page classes (POM)
│   ├── com.bstack.testdata/ # Excel test data
│   └── com.bstack.utils/ # Utility classes (Waits, Config, WebDriverFactory)
│
├── src/test/java/
│   └── com.bstack.tests/ # Test classes (Login, Cart, Checkout)
│
├── Screenshots/ # Captured screenshots
├── test-output/ # TestNG reports
├── ExtentReport.html # HTML report generated by ExtentReports
├── TestNG.xml # TestNG suite configuration
├── pom.xml # Maven dependencies
└── README.md # Project documentation
```

Output:

📸 Sample Output

- Screenshots are saved in `/Screenshots` folder
- Extent Report is generated as `ExtentReport.html` in the project root directory