Java Selenium Automation for bstackdemo.com

1. Capstone Project Title

Automation Testing Framework for bstackdemo.com using Java and Selenium

2. Domain

E-Commerce Web Application

3. Application Under Test (AUT)

Website: https://bstackdemo.com/

Description: bstackdemo.com is a sample e-commerce application provided by BrowserStack to demonstrate cross-browser testing. It includes features such as user login, product catalog, filtering, adding items to the cart, and order placement.

4. Project Objective

To design and implement a modular and scalable automation testing framework using Java, Selenium WebDriver, and TestNG for testing the core functionalities of the bstackdemo.com e-commerce platform.

5. Tools and Technologies Used

- Java: Programming Language

- Selenium WebDriver: Browser Automation

- TestNG: Test Execution and Reporting

- Maven: Project Build and Dependency Management

- ExtentReports: Customized Test Reporting

- Git/GitHub: Version Control

6. Framework Architecture

Design Pattern: Page Object Model (POM)

Components:

  - BaseTest.java – Initializes WebDriver and loads configs

  - Page Classes – LoginPage, ProductPage, CartPage, CheckoutPage

  - Utilities – ConfigReader, WebDriverFactory, WaitUtils

  - Test Cases – LoginTest, AddToCartTest, CheckoutTest

  - TestNG.xml – Test suite configuration

  - Reports – ExtentReports integration

7. 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
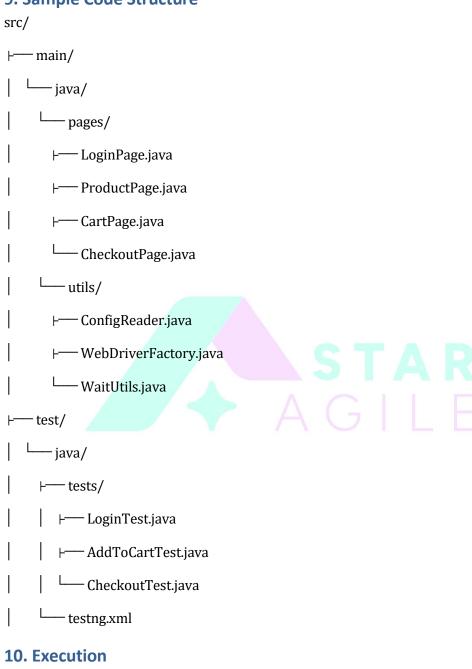
8. Automate Test Case Scenarios

🟩 Login Tests

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

  - TC_002: Login with invalid credentials

  - TC_003: Login with empty username/password

🛒 Cart Tests

  - TC_004: Add single item to cart

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

  - TC_006: Remove item from cart

🧾 Checkout Tests

- TC_007: Place order with valid details

- TC_008: Checkout flow without adding items (negative test)

9. Sample Code Structure

```
src/
├── main/
│   └── java/
│       └── pages/
│           ├── LoginPage.java
│           ├── ProductPage.java
│           ├── CartPage.java
│           └── CheckoutPage.java
│       └── utils/
│           ├── ConfigReader.java
│           ├── WebDriverFactory.java
│           └── WaitUtils.java
├── test/
│   └── java/
│       ├── tests/
│       │   ├── LoginTest.java
│       │   ├── AddToCartTest.java
│       │   └── CheckoutTest.java
│       └── testng.xml
```

10. Execution

Command Line (Maven):

mvn clean test

Using TestNG XML:

Run testng.xml from your IDE or via Maven Surefire plugin.

11. Reporting

ExtentReports used for HTML-based graphical reports.

TestNG default reports generated under test-output/ directory.