Capstone Project

Capstone Project: E-Commerce Application Domain Project Requirements

Project Overview Project Name: Automation Testing for Nixonex E-commerce

Domain: E-Commerce Application

Project Description: This project aims to automate testing of the https://www.demoblaze.com/ website using Java programming language and Selenium WebDriver.

The automation will cover functional testing of key user flows to ensure the application behaves as expected across different browsers and platforms.

Objectives:

• Automate testing user workflows on https://www.demoblaze.com/

• Verify application functionality across multiple browsers (Chrome, Firefox, and Edge).

• Validate user interface elements for responsiveness.

• Implement test cases for both positive and negative scenarios.

Scope:

In Scope:

• Register Functionality

• Login functionality.

• Navigation through different pages and sections.

• Search functionality.

• Add products to the Shopping Cart.

• Basic UI responsiveness across devices and browsers.

Out of Scope:

• Performance testing.

• Security testing beyond basic input validation.

• Load testing.

Tools and Technologies:

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

Test Environment:

- **Target Application**: https://www.demoblaze.com/
- **Supported Browsers**: Chrome, Firefox, Edge (Any One)
- **Operating Systems**: Windows, macOS, Linux (Any One)

Flow Of the Project:

Home Page:

- User lands on the home page.
- View featured products or promotions.

Product Browsing:

- User navigates to different product categories (e.g., Electronics, Clothing, Home Appliances).

• Filters products based on criteria (e.g., price range, brand).

Product Detail Page:

• User clicks on a product to view details.

• View product images, descriptions, specifications, and customer reviews.

• Adds the product to the cart or wish list.

Shopping Cart:

• User adds products to the shopping cart.

• Adjust quantities or remove items.

• Proceeds to checkout.

Checkout Process:

• User enters shipping address and selects shipping method.

• Enters payment information (credit card, PayPal, etc.).

• Reviews order summary and confirms purchase.

Order Confirmation:

• User receives confirmation of the order.

• Download PDF for the same.

Description of Automation Testing:

1. Create a page Object model for this application and write an automation script for a complete process where the user can register to log in and then search for the product add it to the cart and complete the checkout process.

2. Add an Extent report for the project.

3. Add Screenshots for all modules.

4. Push the project script on GitHub and share the link along with a PDF for the complete Project Work.

Project:- E-Commerce Application Domain Project Requirements

Src/main/java

Package : com.pages

Register page:-

package pages;

```
import org.openqa.selenium.*;
import org.openqa.selenium.support.ui.*;

public class RegisterPage {
    WebDriver driver;

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

    public void register(String username, String password)
{
        driver.findElement(By.id("signin2")).click();
        new WebDriverWait(driver, Duration.ofSeconds(5))

.until(ExpectedConditions.visibilityOfElementLocated(By.i
d("sign-username")));
        driver.findElement(By.id("sign-
username")).sendKeys(username);
```

```
        driver.findElement(By.id("sign-
password")).sendKeys(password);
        driver.findElement(By.xpath("//button[text()='Sign
up']")).click();
    }
}
```

Login Page:-

```
package pages;

import org.openqa.selenium.*;
import org.openqa.selenium.support.ui.*;

public class LoginPage {
    WebDriver driver;

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

```
    public void login(String username, String password) {
        driver.findElement(By.id("login2")).click();
        new WebDriverWait(driver, Duration.ofSeconds(5))

.until(ExpectedConditions.visibilityOfElementLocated(By.id("loginusername")));

driver.findElement(By.id("loginusername")).sendKeys(username);

driver.findElement(By.id("loginpassword")).sendKeys(password);
        driver.findElement(By.xpath("//button[text()='Log in']")).click();
    }
}
```

Product Page:-

```
package pages;

import org.openqa.selenium.*;

public class ProductPage {
    WebDriver driver;

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

    public void selectProduct(String productName) {

driver.findElement(By.linkText(productName)).click();
    }

    public void addToCart() {
        driver.findElement(By.linkText("Add to cart")).click();
        driver.switchTo().alert().accept();
```

```
    }
}
```

Cart Page :-

```
package pages;

import org.openqa.selenium.*;

public class CartPage {
    WebDriver driver;

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

    public void placeOrder(String name, String card) {
        driver.findElement(By.id("cartur")).click();
        driver.findElement(By.xpath("//button[text()='Place
Order']")).click();
```

```
        driver.findElement(By.id("name")).sendKeys(name);
        driver.findElement(By.id("card")).sendKeys(card);

driver.findElement(By.xpath("//button[text()='Purchase']
")).click();
    }
}
```

Package :- com.Utilities

Extent Manager:-

```
package utilities;

import com.aventstack.extentreports.*;
import
com.aventstack.extentreports.reporter.ExtentSparkRepo
rter;

public class ExtentManager {
    private static ExtentReports extent;
```

```
    public static ExtentReports createInstance(String
fileName) {
        ExtentSparkReporter reporter = new
ExtentSparkReporter(fileName);
        extent = new ExtentReports();
        extent.attachReporter(reporter);
        return extent;
    }
}
```

ScreenShotUtil:-

```
package utilities;

import org.openqa.selenium.*;
import java.io.File;
import java.text.SimpleDateFormat;
import java.util.Date;
```

```
public class ScreenshotUtil {
    public static String captureScreenshot(WebDriver
driver, String name) {
        try {
            File src = ((TakesScreenshot)
driver).getScreenshotAs(OutputType.FILE);
            String path = "screenshots/" + name + "_" +
                    new
SimpleDateFormat("yyyyMMddHHmmss").format(new
Date()) + ".png";
            File dest = new File(path);
            dest.getParentFile().mkdirs();
            org.openqa.selenium.io.FileHandler.copy(src,
dest);
            return path;
        } catch (Exception e) {
            return null;
        }
    }
```

}

src/test/java

Package : com.Test

Base Test:-

```
package tests;

import com.aventstack.extentreports.*;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.ITestResult;
import org.testng.annotations.*;
import utilities.*;

import java.lang.reflect.Method;

public class BaseTest {
    protected WebDriver driver;
    protected static ExtentReports extent;
```

```
    protected static ExtentTest test;

    @BeforeSuite
    public void setupReport() {
        extent = ExtentManager.createInstance("test-
output/ExtentReport.html");
    }

    @BeforeMethod
    public void setUp(Method method) {
        driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.demoblaze.com/");
        test = extent.createTest(method.getName());
    }

    @AfterMethod
    public void tearDown(ITestResult result) {
        if (result.getStatus() == ITestResult.FAILURE) {
```

```
            String screenshotPath =
ScreenshotUtil.captureScreenshot(driver,
result.getName());
            test.fail(result.getThrowable());
            test.addScreenCaptureFromPath(screenshotPath);
        } else if (result.getStatus() == ITestResult.SUCCESS) {
            test.pass("Test passed");
        }
        driver.quit();
    }

    @AfterSuite
    public void tearDownReport() {
        extent.flush();
    }
}
```

RegisterTest:-

```
package tests;
```

```
import org.testng.annotations.Test;
import pages.RegisterPage;

public class RegisterTest extends BaseTest {
    @Test
    public void testRegister() {
        RegisterPage registerPage = new
RegisterPage(driver);
        registerPage.register("testuser123", "pass123");
    }
}
```

LoginTest:-

```
package tests;

import org.testng.annotations.Test;
import pages.LoginPage;

public class LoginTest extends BaseTest {
```

```
    @Test
    public void testLogin() {
        LoginPage loginPage = new LoginPage(driver);
        loginPage.login("testuser", "testpass");
    }
}
```

AddToCartTest:-

```
package tests;

import org.testng.annotations.Test;
import pages.ProductPage;

public class AddToCartTest extends BaseTest {
    @Test
    public void testAddToCart() {
        ProductPage productPage = new
ProductPage(driver);
        productPage.selectProduct("Samsung galaxy s6");
```

```
        productPage.addToCart();
    }
}
```

CheckoutTest :-

```
package tests;

import org.testng.annotations.Test;
import pages.CartPage;

public class CheckoutTest extends BaseTest {
    @Test
    public void testCheckout() {
        CartPage cartPage = new CartPage(driver);
        cartPage.placeOrder("John Doe",
"1234567812345678");
    }
}
```