**// Automate an E-Commerce Web Application.**

**//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>NgTestingWebApp</groupId>

<artifactId>NgTestingWebApp</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>4.10.0</version>

</dependency>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-chrome-driver</artifactId>

<version>4.10.0</version>

</dependency>

<!--

https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-firefox-driver -->

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-firefox-driver</artifactId>

<version>4.10.0</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.8.0</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>AutomateWebAppTestng</groupId>

<artifactId>AutomateWebAppTestng</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</build>

</project>

**//testng.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name=*"Suite"*>

<test thread-count=*"5"* name=*"Test"*>

<classes>

<class name=*"NgTesting.My\_Flipkart\_Product"*/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

**//My\_Flipkart\_Product.java**

**package NgTesting;**

**import java.io.IOException;**

**import java.util.List;**

**import java.util.concurrent.TimeUnit;**

**import org.openqa.selenium.By;**

**import org.openqa.selenium.JavascriptExecutor;**

**import org.openqa.selenium.Keys;**

**import org.openqa.selenium.WebDriver;**

**import org.openqa.selenium.WebDriverException;**

**import org.openqa.selenium.WebElement;**

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

**import org.testng.Assert;**

**import org.testng.AssertJUnit;**

**import org.testng.annotations.AfterMethod;**

**import org.testng.annotations.AfterSuite;**

**import org.testng.annotations.BeforeMethod;**

**import org.testng.annotations.BeforeSuite;**

**import org.testng.annotations.Parameters;**

**import org.testng.annotations.Test;**

**import org.testng.asserts.SoftAssert;**

**public class My\_Flipkart\_Product {**

**private WebDriver driver;**

**@Test**

**public void flipkart() {**

**System.out.println("Welcome to Flipkart explore plus()");**

**}**

**@Test(groups = "flipKart")**

**public void measurePageLoadTimeTest() {**

**long startTime;**

**long endTime;**

**long pageLoadTime;**

**startTime = System.currentTimeMillis();**

**System.out.println("Start time ="+startTime);**

**// Wait for the page to load completely**

**driver.manage().timeouts().pageLoadTimeout(30, TimeUnit.SECONDS);**

**endTime = System.currentTimeMillis();**

**System.out.println("end time ="+endTime);**

**System.out.println("load time ="+ (endTime-startTime));**

**}**

**@Test(groups = "flipKart")**

**public void afterMethod() {**

**WebElement x = driver.findElement(By.cssSelector("body > div.\_2Sn47c > div > div > button"));**

**x.click();**

**WebElement mobile = driver.findElement(By.cssSelector("#container > div > div.\_331-kn.\_2tvxW > div > div > div:nth-child(2) > a > div.xtXmba"));**

**mobile.click();**

**WebElement SearchForMobile = driver.findElement(By.name("q"));**

**SearchForMobile.sendKeys("iPhone 13"+Keys.ENTER);**

**System.out.println("Searched for iphone 13");**

**}**

**@Test(groups = "flipKart")**

**public void checkImageVisibilityTest() {**

**List<WebElement> images = driver.findElements(By.tagName("img"));**

**int WebHeight = driver.manage().window().getSize().getHeight();**

**System.out.println("\n===================================================\nImages\n\n");**

**for(WebElement img:images) {**

**int imageLocation = img.getLocation().getY();**

**if(imageLocation < WebHeight && imageLocation>=0) {**

**if(img.isDisplayed()) {**

**System.out.println("Image is loaded and displayed = "+img.getAttribute("src"));**

**}**

**else {**

**System.out.println("Image is not displayed ="+img.getAttribute("src"));**

**}**

**}**

**else {**

**System.out.println("Image is out of screen height = "+img.getAttribute("src"));**

**}**

**}**

**System.out.println("\n===================================================");**

**}**

**@Test(groups = "flipKart")**

**public void scrollFeature() throws InterruptedException, WebDriverException, IOException {**

**System.out.println("\n===================================================");**

**WebElement body = driver.findElement(By.tagName("body"));**

**System.out.println(body.getLocation());**

**int tabHeight=driver.manage().window().getSize().getHeight();**

**int contentHeight=body.getSize().height;**

**System.out.println("windows tab height ="+ tabHeight);**

**System.out.println("height of dody content ="+ contentHeight);**

**int different = contentHeight-tabHeight;**

**SoftAssert softAssert = new SoftAssert();**

**softAssert.assertTrue(different>0);**

**System.out.println("This page has scroll features");**

**}**

**@Test(groups = "flipKart")**

**public void scrollToEnd() throws WebDriverException, IOException {**

**System.out.println("\n===================================================");**

**WebElement body = driver.findElement(By.tagName("body"));**

**body.sendKeys(Keys.END);**

**}**

**@Test(groups = "flipKart")**

**public void checkContentRefreshFrequencyTest() {**

**// Navigate to the Flipkart home page**

**// Scroll down multiple times to trigger content refresh**

**for (int i = 0; i < 5; i++) {**

**JavascriptExecutor jsExecutor = (JavascriptExecutor) driver;**

**jsExecutor.executeScript("window.scrollTo(0, document.body.scrollHeight);");**

**// Wait for a moment to let the content refresh**

**try {**

**Thread.sleep(2000);**

**} catch (InterruptedException e) {**

**e.printStackTrace();**

**}**

**}**

**// Perform calculations to determine the frequency of content refresh**

**int refreshFrequency = 5; // Number of times scrolled**

**long totalTimeTaken = 10000; // 10 seconds (total wait time for content to refresh)**

**// Calculate the frequency at which the content is refreshed**

**int contentRefreshFrequency = (int) (refreshFrequency / (totalTimeTaken / 1000.0));**

**System.out.println("contentRefreshFrequency:"+contentRefreshFrequency);**

**// Perform assertions on the content refresh frequency**

**// Example: Assert.assertEquals(contentRefreshFrequency, 2, "Content is not being refreshed as expected.");**

**}**

**/\* @Test(groups = "flipKart")**

**public void verifyImageDownloadAndDisplayTimingTest() {**

**// Navigate to the Flipkart home page**

**// Get the coordinates of the image element**

**WebElement imageElement = driver.findElement(By.xpath("//\*[@id=\"container\"]/div/div[3]/div[1]/div[1]/div[1]/div/div[1]/div[2]/div[1]/div[2]/img"));**

**int imageElementY = imageElement.getLocation().getY();**

**// Scroll to the image position**

**JavascriptExecutor jsExecutor = (JavascriptExecutor) driver;**

**jsExecutor.executeScript("window.scrollTo(0, arguments[0]);", imageElementY);**

**// Wait for the image to be downloaded and displayed**

**try {**

**Thread.sleep(2000); // Adjust the wait time as needed**

**} catch (InterruptedException e) {**

**e.printStackTrace();**

**}**

**// Check if the image is displayed**

**AssertJUnit.assertTrue(imageElement.isDisplayed());**

**// Check if the image is downloaded in time just before scrolling to its position**

**AssertJUnit.assertTrue(imageElement.getAttribute("naturalWidth") != "0");**

**// Optionally, you can perform additional checks on the image, such as size, resolution, etc.**

**} \*/**

**@Test(groups = "flipKart")**

**public void verifyScrollToBottomTest() {**

**// Scroll to the bottom of the page**

**JavascriptExecutor jsExecutor = (JavascriptExecutor) driver;**

**jsExecutor.executeScript("window.scrollTo(0, document.body.scrollHeight);");**

**// Wait for a moment to let the page load after scrolling**

**try {**

**Thread.sleep(2000); // You can adjust the wait time as needed**

**} catch (InterruptedException e) {**

**e.printStackTrace();**

**}**

**// Verify that the page has been scrolled to the bottom**

**long totalPageHeight = (Long) jsExecutor.executeScript("return Math.max( document.body.scrollHeight, document.body.offsetHeight, document.documentElement.clientHeight, document.documentElement.scrollHeight, document.documentElement.offsetHeight );");**

**long windowHeight = (Long) jsExecutor.executeScript("return window.innerHeight;");**

**long scrollPosition = (Long) jsExecutor.executeScript("return window.scrollY;");**

**// Assert that the scroll position is near the bottom of the page**

**long buffer = 50; // You can adjust the buffer value as needed**

**long expectedScrollPosition = totalPageHeight - windowHeight - buffer;**

**assert Math.abs(expectedScrollPosition - scrollPosition) <= buffer :**

**"The page is not scrolled to the bottom.";**

**System.out.println("expectedScrollPosition:"+ expectedScrollPosition);**

**System.out.println("----------------------------------------------------");**

**System.out.println("----------------------------------------------------");**

**// Optionally, you can perform additional checks or assertions based on the test requirements.**

**}**

**@BeforeSuite**

**public void beforeSuite() {**

**driver = new ChromeDriver();**

**driver.get("https://www.flipkart.com/");**

**driver.manage().window().maximize();**

**}**

**@AfterSuite**

**public void afterSuite() {**

**// driver.quit();**

**}**

**}**