# FlipkartLazyLoadingTest.java

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor; import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver; import org.openqa.selenium.edge.EdgeDriver;

import org.openqa.selenium.support.ui.ExpectedConditions; import org.openqa.selenium.support.ui.WebDriverWait; import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeTest; import org.testng.annotations.Test;

import io.github.bonigarcia.wdm.WebDriverManager;

public class FlipkartLazyLoadingTest {

WebDriver driver;

@BeforeTest

public void setup() { WebDriverManager.chromedriver().setup(); driver = new ChromeDriver(); driver.manage().window().maximize();

// WebDriverManager.edgedriver().setup();

// driver = new EdgeDriver();

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

}

@Test

public void testFlipkartSearch() {

// Navigate to Flipkart homepage driver.get("https:[//w](http://www.flipkart.com/)ww[.flipkart.com/"](http://www.flipkart.com/));

// Determine page load time

long startTime = System.currentTimeMillis();

// Your page load time logic here

long endTime = System.currentTimeMillis(); long pageLoadTime = endTime - startTime;

System.out.println("Page Load Time: " + pageLoadTime + " milliseconds");

// Search for "iPhone 13" under the "Mobile" category driver.findElement(By.name("q")).sendKeys("iPhone 13"); driver.findElement(By.cssSelector("button[type='submit']")).click();

// Check images are loaded and visible till the screen height

WebElement firstProductImage = driver.findElement(By.partialLinkText("APPLE iPhone 13 (Pink, 128 GB)"));

if (firstProductImage.isDisplayed()) { System.out.println("First product image is visible.");

} else {

System.out.println("First product image is not visible.");

}

// Check the page has a scroll feature JavascriptExecutor js = (JavascriptExecutor) driver;

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

// Check the frequency at which the content will be refreshed while scrolling long refreshStartTime = System.currentTimeMillis();

// Your logic to identify the frequency of content refresh

long refreshEndTime = System.currentTimeMillis(); long refreshTime = refreshEndTime - refreshStartTime;

System.out.println("Content Refresh Frequency: " + refreshTime + " milliseconds");

// Verify that the image is downloaded just before the user scrolls to its position and gets displayed in time

WebElement lazyLoadedImage = driver.findElement(By.partialLinkText("APPLE iPhone 13 (Midnight, 128 GB)"));

scrollIntoView(lazyLoadedImage);

// Wait for the image to become visible WebDriverWait wait = new WebDriverWait(driver, 10);

wait.until(ExpectedConditions.visibilityOf(lazyLoadedImage));

System.out.println("Lazy-loaded image is visible after scrolling.");

// Verify it navigates to the bottom of the page

WebElement we = driver.findElement(By.partialLinkText("APPLE iPhone 13 (Midnight, 128 GB)"));

scrollIntoView(we); we.click();

// Check whether different browsers and screen resolutions render it the same way System.out.println("Browser: " + ((ChromeDriver) driver).getCapabilities().getBrowserName()); System.out.println("Screen Resolution: " + driver.manage().window().getSize());

// System.out.println("Browser: " + ((EdgeDriver) driver).getCapabilities().getBrowserName());

// System.out.println("Screen Resolution: " + driver.manage().window().getSize());

}

private void scrollIntoView(WebElement element) { JavascriptExecutor js = (JavascriptExecutor) driver; js.executeScript("arguments[0].scrollIntoView();", element);

}

@AfterTest

public void tearDown() { if (driver != null) {

driver.quit();

}

}

}

# 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> <http://maven.apache.org/xsd/maven-4.0.0.xsd>">

<modelVersion>4.0.0</modelVersion>

<groupId>Phase5</groupId>

<artifactId>AutomatedWeb</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>AutomatedWeb</name>

<url>[http://maven.apache.org](http://maven.apache.org/)</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<!-- <https://mvnrepository.com/artifact/junit/junit>-->

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.4.0</version>

</dependency>

<!--

<https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager>--

>

<dependency>

<groupId>io.github.bonigarcia</groupId>

<artifactId>webdrivermanager</artifactId>

<version>5.6.2</version>

</dependency>

<!-- [https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-](https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java) [java](https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java) -->

<dependency>

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

<artifactId>selenium-java</artifactId>

<version>3.141.59</version>

</dependency>

</dependencies>

</project>