**Automate an E-Commerce Web Application**

**DESCRIPTION:**

To automate a real-world web application

**Description:**

Flipkart is an e-commerce platform, and they have launched a new feature to search for a product in a particular category. Once the product is searched, Flipkart displays it as a list of product items. To enhance the performance of the application, Flipkart has implemented lazy loading. It displays only a few products that can come on the screen.

To display or load more products, the user must scroll down.

As a Test Engineer, you are expected to test this feature end-to-end.

**Detailed Scenario:**

* Navigate to the Flipkart homepage (<https://www.flipkart.com/>)
* Determine a page load time with a performance test
* Search for a product, say, “iPhone 13” under the “Mobile” category
* Check if the images are loaded and visible till the screen height only
* Check if the page has a scroll feature
* Check the frequency at which the content will be refreshed while scrolling
* Verify that the image is downloaded just before the user scrolls to its position and gets displayed in time
* Verify that it navigates to the bottom of the page
* Check whether different browsers and screen resolutions render it the same way

**Tools Required**:

1. Selenium Library
2. Eclipse IDE
3. TestNG Library
4. Maven

**Required Steps :**

1.It would launch the Chrome browser and maximize the window.

**2.**Navigate to the Flipkart website and perform Login operation.

***Note:****You can change the username and password values or leave them as is. Since it’s just a demo, so you don’t need the credentials.*

**3.** Now, it’ll search for the book keyword which we choose as <*Selenium*>. Change it if you want to.

**4.** In this step, the code will click to view the search results.

**5.** We’ll now fetch the list of books displayed and select the last one using the XPath locator given below. You change the XPath to choose a different book.

**CODE:**

package flipkartfinalproject;

import java.util.ArrayList;

import java.util.concurrent.TimeUnit;

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.testng.Assert;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.Test;

public class search {

public static String *str*;

public static WebDriver *driver*;

public static WebElement *element*;

*@BeforeSuite*

public void initialization()

{

System.*setProperty*("webdriver.chrome.driver", "D:\\seleniumtutorial\\chromedriver\_win32\\chromedriver.exe");

*driver* = new ChromeDriver();

}

*@Test*(priority=1)

public static void search()

{

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

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

//Long loadtime = (Long)((JavascriptExecutor)driver).executeScript(

//"return performance.timing.loadEventEnd - performance.timing.navigationStart;");

*driver*.manage().timeouts().pageLoadTimeout(5, *TimeUnit*.***SECONDS***);

*driver*.manage().timeouts().implicitlyWait(6, *TimeUnit*.***SECONDS***);

*element* = *driver*.findElement(By.*xpath*("//input[@placeholder='Search for products, brands and more']"));

*element*.sendKeys("iPhone 13");

*element*.findElement(By.*xpath*("//button[@type='submit']//\*[name()='svg']")).click();

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

System.***out***.println("successfully searched the product..!!");

}

}

Testing.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>**

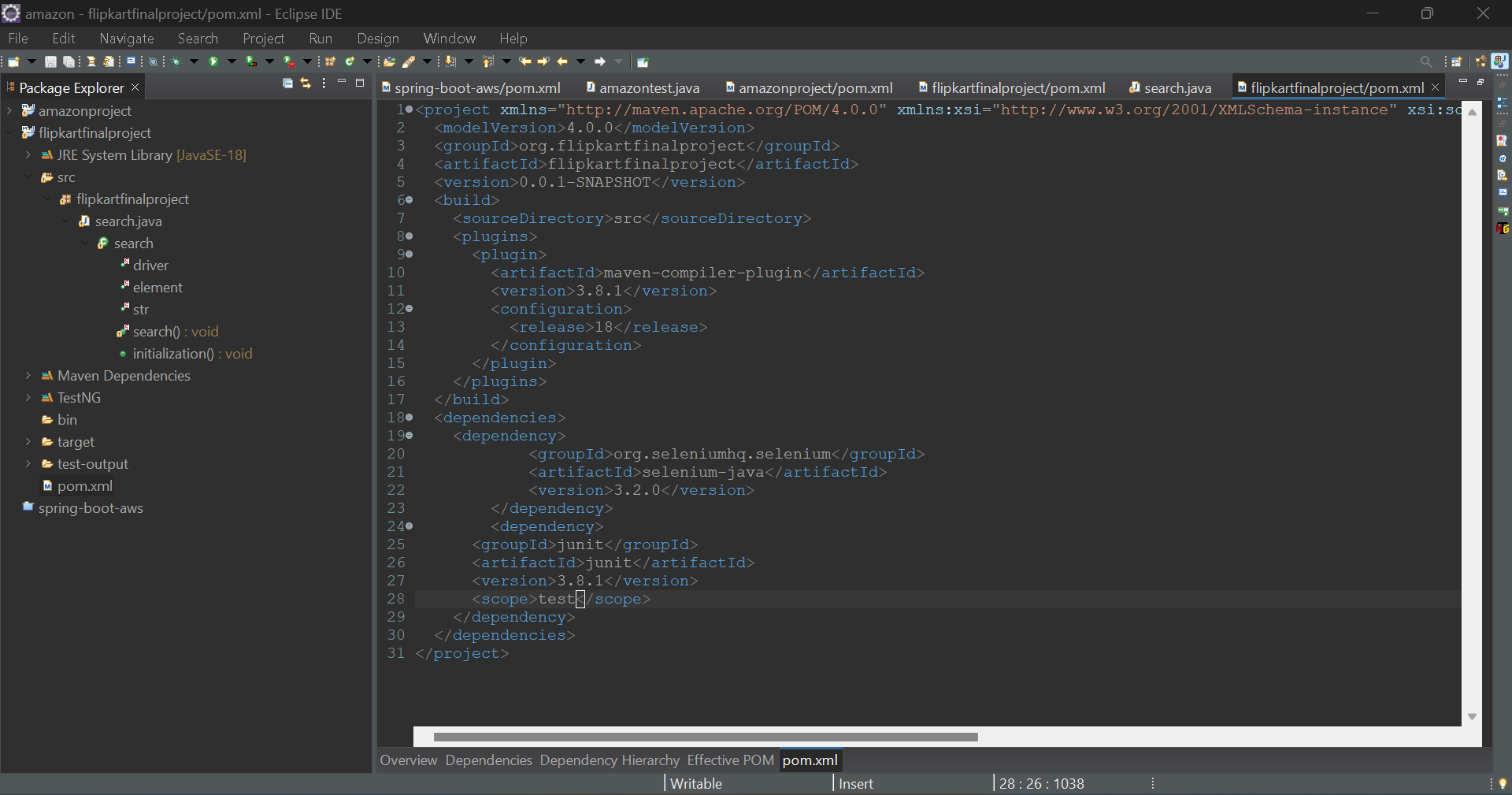
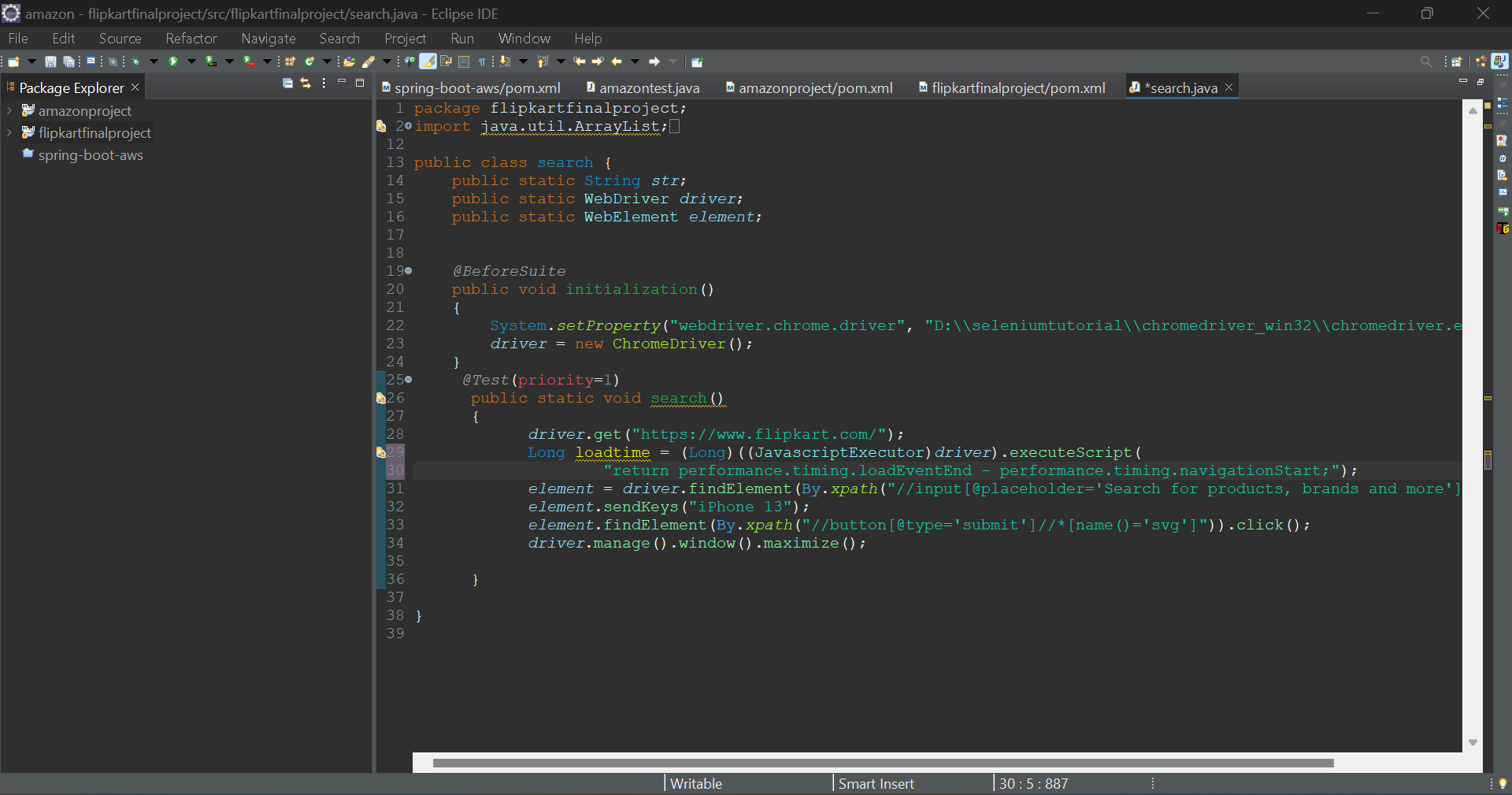
**<class name=*"flipkartfinalproject.Flipkartsearch"*/>**

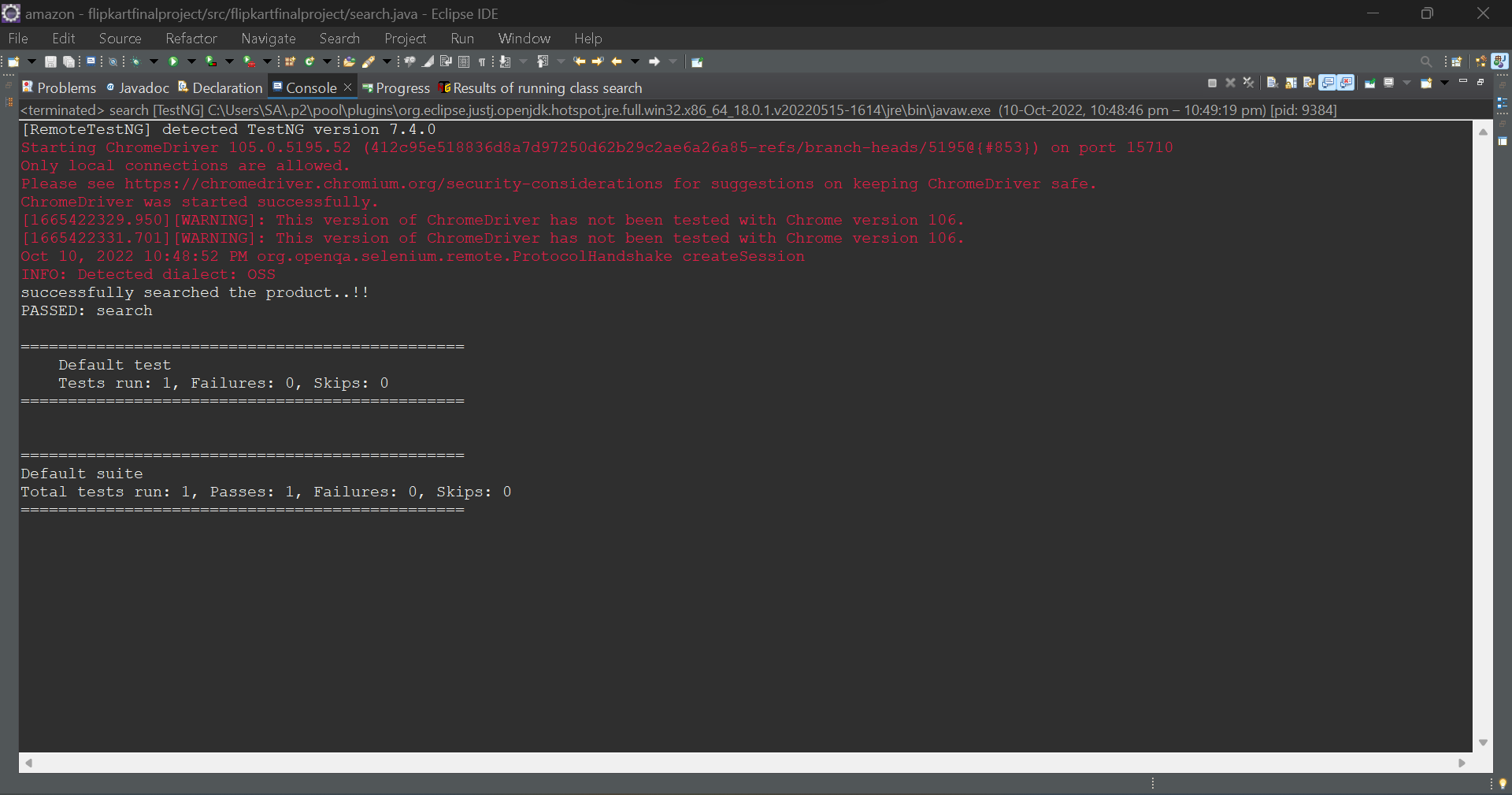
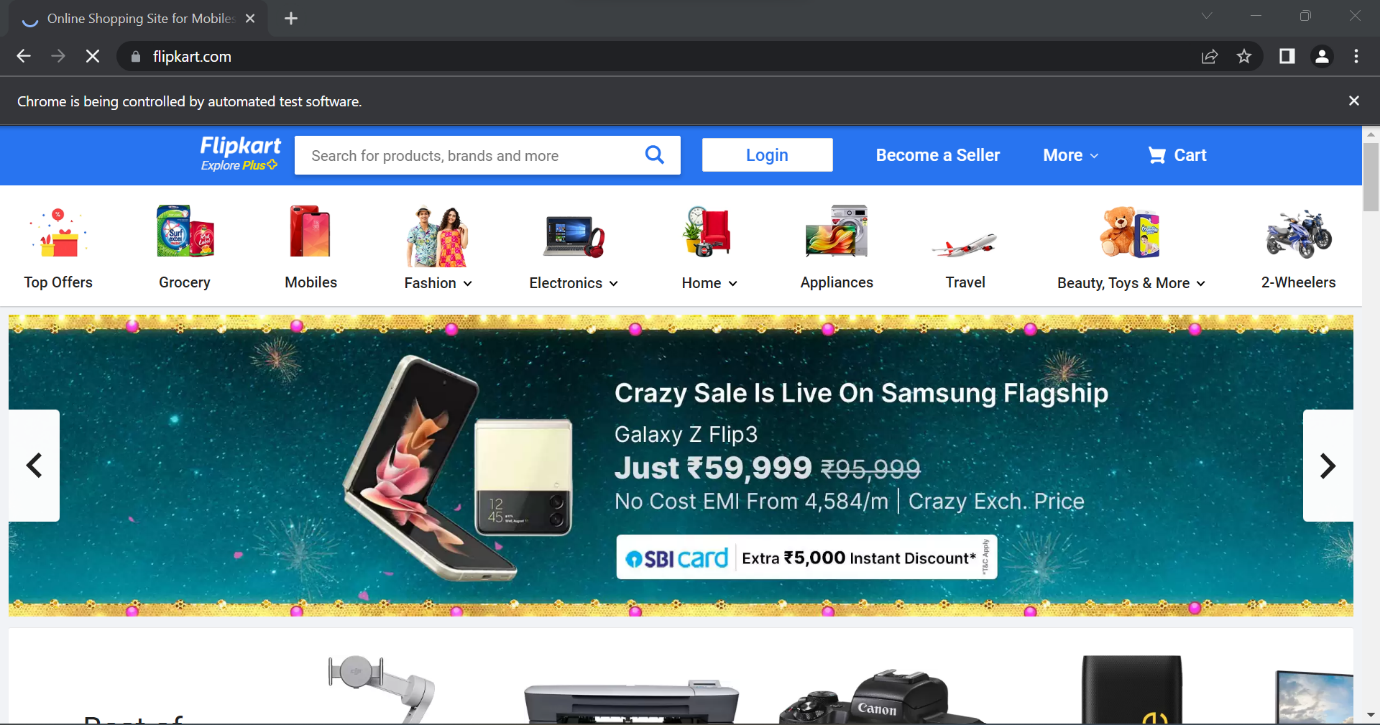
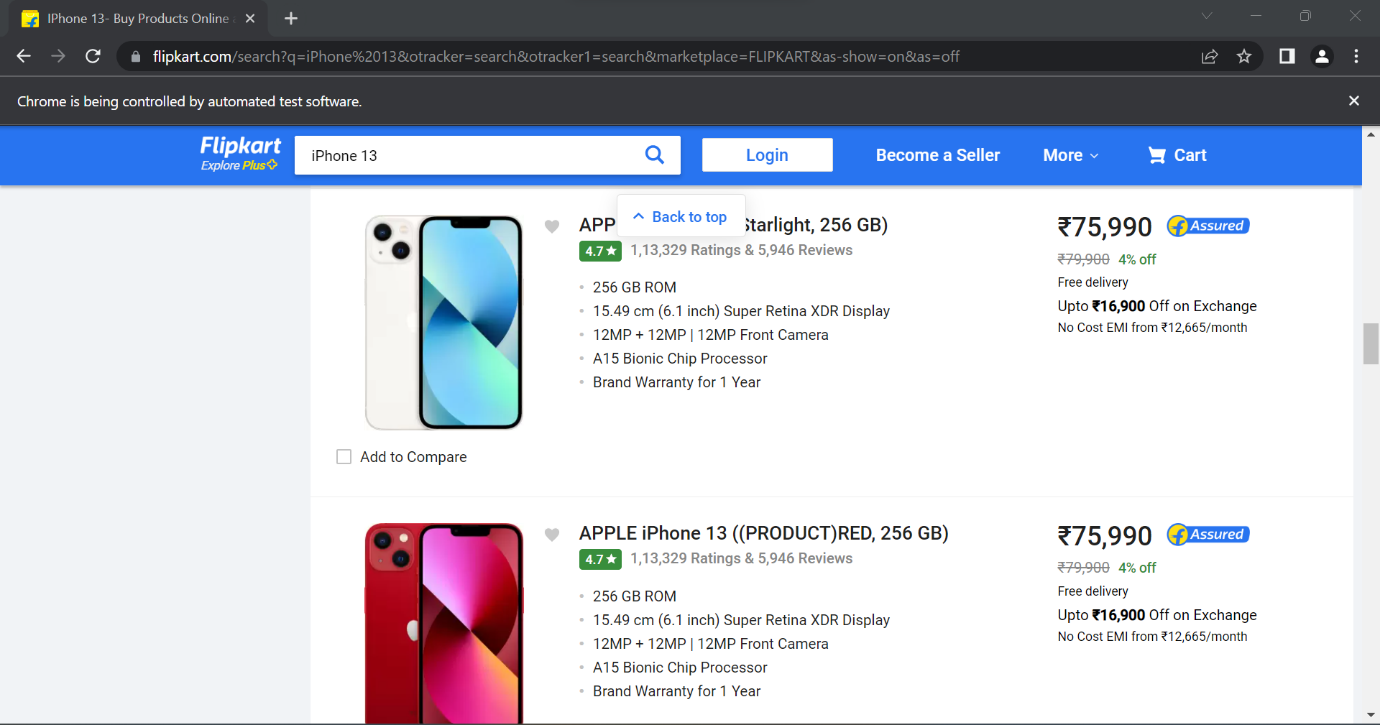
**</classes>**

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

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

**Output:**

**Graphical user interface, text

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, text, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated**