Experiment 1: Develop the Test Plan for Testing an E-commerce Web/Mobile Application(www.amazon.in)

Aim:

The aim of this experiment is to develop a comprehensive test plan for testing the functionalityand usability of the e-commerce web/mobile application www.amazon.in.

Algorithm:

1. Identify the Scope: Determine the scope of testing, including the features and functionalitiesthat need to be tested.

2. Define Test Objectives: Specify the primary objectives of testing, such as functional testing,usability testing, performance testing, security testing, etc.

3. Identify Test Environment: Define the platforms, browsers, devices, and operating systems onwhich the application will be tested.

4. Determine Test Deliverables: Decide on the documents and artifacts that will be generatedduring the testing process, such as test cases, test reports, and defect logs.

5. Create Test Strategy: Develop an overall approach for testing, including the testing techniques,entry and exit criteria, and the roles and responsibilities of the testing team.

6. Define Test Scope and Schedule: Specify the timeline for each testing phase and the scope oftesting for each phase.

7. Risk Analysis: Identify potential risks and their impact on the testing process, and devise riskmitigation strategies.

8. Resource Planning: Allocate the necessary resources, including the testing team, hardware, andsoftware required for testing.

9. Test Case Design: Prepare detailed test cases based on the requirements and functionalities ofthe e-commerce application.

10. Test Data Setup: Arrange test data required for executing the test cases effectively.

11. Test Execution: Execute the test cases and record the test results.

12. Defect Reporting: Document any defects encountered during testing and track theirresolution.

Test Plan:

The test plan should cover the following sections:

1. Introduction: Briefly describe the purpose of the test plan and provide an overview of the e-commerce application to be tested.

2. Test Objectives: List the primary objectives of testing the application.

3. Test Scope: Specify the features and functionalities to be tested and any limitations on testing.

4. Test Environment: Describe the hardware, software, browsers, and devices to be used fortesting.

5. Test Strategy: Explain the overall approach to be followed during testing.

6. Test Schedule: Provide a detailed timeline for each testing phase.

7. Risk Analysis: Identify potential risks and the strategies to mitigate them.

8. Resource Planning: Specify the resources required for testing.

9. Test Case Design: Include a summary of the test cases developed for the application.

10. Test Data Setup: Describe the process of arranging test data for testing.

11. Defect Reporting: Explain the procedure for reporting and tracking defects.

Test Case Table:

Process	**No.**	**Test Case**	**Steps**	**Description**	**Status**	**Expected Result**	**Actual Result**	**Comment**
Test Plan	TC001	Scope of Testing	1. Review the test plan document.	Verify thescope of testing.	Done	The test plan includes all features.		
	TC002	Test Objective s	1. Review the test plan document.	Verify the test objectives .	Done	The test objectives are well-defined.		
	TC003	Test Environmen t	1. Review the test plan document.	Check the specified environments .	Done	Test environments are mentioned.		
	TC004	Test Deliverable s	1. Review the test plan document.	Ensure all deliverable sare listed.	Done	The test plan includes all deliverables.		
	TC005	Test Strateg y	1. Review the test plan document.	Verify the overall approach.	Done	The test strategy is clearly stated.		
	TC006	Test Scopeand Schedule	1. Review the test plan document.	Check the schedule andscope.	Done	The schedule and scope are defined.		

	TC007	Risk Analysis	1. Review the test plan document.	Ensure potential risksare identified.	Done	Risks and mitigation strategies are mentioned.		
	TC008	Resource Planning	1. Review the test plan document.	Check the required resources.	Done	Resources needed for testing arelisted.		
	TC009	Test Case Design	1. Review and executethe test cases.	Validate the prepared testcases.	Done	Test cases are accurate and functional.		
	TC010	Test Data Setup	1. Review the test datasetup process.	Verify the availability oftest data.	Done	Test data is available fortesting.		
	TC011	Test Execution	1. Run thetest cases and document the outcomes.	Execute thetest cases.	In Progress	Test results are recorded and documented.		
	TC012	Defect Reporting	1. Log defects with detailed information.	Ensure defects are reported correctly.	Not Started	Defects are reported with sufficient details.		
	TC013	Defect Tracking	1. Monitor defect statusand updates.	Verify the tracking ofdefects.	Not Started	Defects are tracked until resolution.		

Explanation:

The test plan is a crucial document that outlines the entire testing process. It ensures that all aspects of the e-commerce application are thoroughly tested, and the results are systematicallydocumented.

Result:

Upon completion of the experiment, you will have a well-structured test plan that provides a clearroadmap for testing the e-commerce web/mobile application www.amazon.in.

Experiment 2: Design the Test Cases for Testing the E-commerce Application

Aim:

The aim of this experiment is to design a set of comprehensive and effective test cases for testingthe e-commerce application www.amazon.in.

Algorithm:

1. Understand Requirements: Familiarize yourself with the functional and non-functionalrequirements of the e-commerce application.

2. Identify Test Scenarios: Based on the requirements, identify different test scenarios that coverall aspects of the application.

3. Write Test Cases: Develop test cases for each identified scenario, including preconditions,steps to be executed, and expected outcomes.

4. Cover Edge Cases: Ensure that the test cases cover edge cases and boundary conditions toverify the robustness of the application.

5. Prioritize Test Cases: Prioritize the test cases based on their criticality and relevance to theapplication.

6. Review Test Cases: Conduct a peer review of the test cases to ensure their accuracy andcompleteness.

7. Optimize Test Cases: Optimize the test cases for reusability and maintainability.

Test Case Design:

The test case design should include the following components for each test case:

1. Test Case ID: A unique identifier for each test case.

2. Test Scenario: Description of the scenario being tested.

3. Test Case Description: Detailed steps to execute the test.

4. Precondition: The necessary conditions that must be satisfied before executing the test case.

5. Test Steps: The sequence of actions to be performed during the test.

6. Expected Result: The outcome that is expected from the test.

Test Case Table:

Process	No.	Test Case	Steps	Description	Status	Expecte dResult	Actual Result	Comment
Test Case Design	TC001	User Registration	1. Navigateto the registrationpage.	Verify user registrationprocess.	Done	User can successfullyregister.		
	TC002	User Login	1. Navigateto the login page.	Verify userlogin process.	Done	User can successfullylog in.		
	TC003	Search Functionality	1. Enter a keyword inthe search bar.	Verify search functionality.	Done	Search results relevant tothe keyword.		
	TC004	Add to Cart	1. Browse the product catalog.	Verify adding products tothe cart.	Done	Product is added to the shopping cart.		
	TC005	Shopping Cart Validation	1. Click onthe shopping cart icon.	Verify the shopping cart contents.	Done	Items in the shopping cart are displayed.		

	TC006	CheckoutProcess	1. Click on the "Checkout "button.	Verify the checkout process.	Not Started	Checkout process proceeds as expected.		

Explanation:

Test cases are designed to validate the functionality and behaviour of the e-commerce application. They ensure that the application performs as intended and meets the specified requirements.

Result:

Upon completion of the experiment, you will have a set of well-defined test cases ready fortesting the e-commerce application www.amazon.in.

Experiment 3: Test the E-commerce Application and Report the Defects in It

Aim:

The aim of this experiment is to execute the designed test cases and identify defects or issues inthe e-commerce application www.amazon.in.

Algorithm:

1. Test Environment Setup: Set up the testing environment with the required hardware, software,and test data.

2. Test Case Execution: Execute the test cases designed in Experiment 2, following the specifiedsteps.

3. Defect Identification: During test execution, record any discrepancies or issues encountered.

4. Defect Reporting: Log the identified defects with detailed information, including steps toreproduce, severity, and priority.

5. Defect Tracking: Track the progress of defect resolution and verify fixes as they areimplemented.

6. Retesting: After defect fixes, retest the affected areas to ensure the issues are resolved.

7. Regression Testing: Conduct regression testing to ensure new changes do not introduce newdefects.

Test Case Table:

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
Test Case Design	TC001	User Registration	1. Navigateto the registratio npage.	Verify user registratio nprocess.	Done	User can successfull yregister.		
	TC002	User Login	1. Navigateto the login page.	Verify userlogin process.	Done	User can successfull ylog in.		
	TC003	Search Functionality	1. Enter a keyword inthe search bar.	Verify search functionality .	Done	Search results relevant tothe keyword.		
	TC004	Add to Cart	1. Browse the product catalog.	Verify adding products tothe cart.	Done	Product is added to the shopping cart.		
	TC005	Shopping Cart Validation	1. Click onthe shopping cart icon.	Verify the shopping cart contents.	Done	Items in the shopping cart are displayed.		
	TC006	Checkou tProcess	1. Click on the "Checkout "button.	Verify the checkout process.	Not Started	Checkout process proceeds as expected.		

Explanation:

Testing the e-commerce application aims to validate its functionality and usability. By identifyingand reporting defects, you ensure the application's quality and reliability.

Result:

Upon completion of the experiment, you will have a list of identified defects and their status afterresolution.

Experiment 4: Develop the Test Plan and Design the Test Cases for an Inventory ControlSystem

Aim:

The aim of this experiment is to create a comprehensive test plan and design test cases for an Inventory Control System.

Algorithm:

Follow the same algorithm as described in Experiment 1 for developing the test plan for aninventory control system.

Follow the same algorithm as described in Experiment 2 for designing test cases for an inventorycontrol system.

Test Plan:

Process	**No.**	**Test Case**	**Steps**	**Description**	**Status**	**Expected Result**	**Actual Result**	**Comment**
Test Plan	TC001	Scope of Testing	1. Review the requirements and project documentation.	Verify thescope of testing.	Done	The test plan includes allessential features.		
			2. Identify the modules to betested.					
			3. Determine the out-of-scope items.					

Process	**No.**	**Test Case**	**Steps**	**Description**	**Status**	**Expected Result**	**Actual Result**	**Comment**
	TC002	Test Objective s	1. Review the requirements and project documentation .	Verify the test objectives .	Done	The test objectives are clearly defined.		
			2. Discuss with stakeholders tounderstand expectations.					
	TC003	Test Environmen t	1. Identify thehardware andsoftware requirements .	Verify the required environments .	Not Started	The test environmen tis defined.		
			2. Set up the required hardware andsoftware.					
	TC004	Test Deliverable s	1. Determine the documents and artifacts tobe produced.	Verify the required deliverables .	Not Started	All necessary document sare listed.		
			2. Create templates fortest reports, defect logs, etc.					
	TC005	Test Strateg y	1. Decide on the testing approach and techniques.	Verify the overall approach fortesting.	Not Started	The test strategy isdefined.		

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
			2. Determine the entry and exit criteria.					
	TC006	Test Scopeand Schedule	1. Define thetimeline for each testing phase.	Verify the schedule fortesting.	Not Started	The schedule is established .		
			2. Determine the scope of testing for eachphase.					
	TC007	Risk Analysi s	1. Identify potential risksin the testing process.	Verify risk analysis and mitigation strategies.	Not Started	Potential risks are identified with mitigatio nplans.		
			2. Discuss riskmitigation strategies withthe team.					
	TC008	Resourc e Planning	1. Allocate therequired resources for testing.	Verify the availability ofresources.	Not Started	Resources needed for testing are allocated.		
			2. Determine the roles and responsibilitie sof the team.					

Test Case Design:

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
Test Case Design	TC001	Module A - Functionalit yTest	1. Review the requirements related to Module A.	Verify the functionalit yof Module A.	Not Started	All functionalitie sof Module A are tested.		
			2. Identify testscenarios for Module A.					
			3. Develop detailed testcases for Module A.					
	TC002	Module B - Integration Test	1. Review the requirements related to Module B.	Verify the integration of Module Bwith others.	Not Started	Module B is successfull yintegrated.		
			2. Identify integration points with other modules.					
			3. Design testcases for testing integration scenarios.					

Process	No.	Test Case	Steps	Description	Status	Expecte dResult	Actual Result	Comment
	TC003	Module C - Performanc eTest	1. Review the performance requirements for Module C.	Verify the performanc eof Module C.	Not Started	Module C performs optimally under load.		
			2. Determine performanc e metrics to bemeasured.					
			3. Develop performanc etest cases forModule C.					
	TC004	Module D -Usability Test	1. Review the usability requirements for Module D.	Verify the usability ofModule D.	Not Started	Module D is user- friendly and intuitive.		
			2. Identify usability aspects to betested.					
			3. Create testcases for evaluating Module D's usability.					
	TC005	Module E - Security Test	1. Review thesecurity requirements for Module E.	Verify the security of Module E.	Not Started	Module E isprotected against security threats.		
			2. Identify potential					

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
			security vulnerabilities.					
			3. Design security test cases to assessModule E.					

Explanation:

An inventory control system is critical for managing stock and supplies. Proper testing ensuresthe system functions accurately and efficiently.

Result:

Upon completion of the experiment, you will have a well-structured test plan and a set of testcases ready for testing the Inventory Control System.

Experiment 5: Execute the Test Cases against a Client-Server or Desktop Application andIdentify the Defects

Aim:

The aim of this experiment is to execute the test cases against a client-server or desktopapplication and identify defects.

Algorithm:

1. Test Environment Setup: Set up the testing environment, including the client-server or desktopapplication, required hardware, and test data.

2. Test Case Execution: Execute the test cases designed in Experiment 2 against the application.

3. Defect Identification: During test execution, record any discrepancies or issues encountered.

4. Defect Reporting: Log the identified defects with detailed information, including steps toreproduce, severity, and priority.

5. Defect Tracking: Track the progress of defect resolution and verify fixes as they areimplemented.

6. Retesting: After defect fixes, retest the affected areas to ensure the issues are resolved.

7. Regression Testing: Conduct regression testing to ensure new changes do not introduce newdefects.

Test Case Table:

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
Test Case Executio n	TC001	User Login	1. Launch the application.	Verify user login process.	Not Started	User can successfull ylog in.		
			2. Enter validlogin credentials.					
			3. Click on the "Login" button.					
	TC002	Data Validation	1. Access adata input form.	Verify data validation onthe form.	Not Started	Invalid datashows appropriate error messages.		
			2. Enter invaliddata in the form fields.					
			3. Submit theform.					
	TC003	File Upload	1. Access thefile upload feature.	Verify file upload functionality .	Not Started	File is uploaded successfully .		
			2. Select a filefrom the system.					
			3. Click on the"Upload" button.					

	TC004	Network Connectivit y	1. Disconnect the network.	Verify the application' sresponse.	Not Started	Application gracefully handles disconnection .		
			2. Attempt toperform an action requiring network access.					
	TC005	Concurren tUsers	1. Simulate concurrent usersessions.	Verify application performance .	Not Started	Application performs wellunder load.		
			2. Perform actions simultaneously .					
	TC006	Compatibility	1. Test the application ondifferent platforms.	Verify cross-platform compatibility .	Not Started	Application works on allspecified platforms.		
			2. Execute testson various browsers.					
	TC007	Client-Server Communicatio n	1. Monitor network trafficbetween clientand server.	Verify communicatio nintegrity.	Not Started	Data is correctly transmitted and received.		

Explanation:

Testing a client-server or desktop application ensures its functionality across different platformsand environmen

Result:

Upon completion of the experiment, you will have a list of identified defects and their status afterresolution for the client-server or desktop application

Experiment 6: Test the Performance of the E-commerce Application

Aim:

The aim of this experiment is to test the performance of the e-commerce applicationwww.amazon.in.

Algorithm:

1. Identify Performance Metrics: Determine the performance metrics to be measured, such asresponse time, throughput, and resource utilization.

2. Define Test Scenarios: Create test scenarios that simulate various user interactions and loadson the application.

3. Performance Test Setup: Set up the performance testing environment with appropriatehardware and software.

4. Execute Performance Tests: Run the performance tests using the defined scenarios and collectperformance data.

5. Analyze Performance Data: Analyze the collected data to identify any performance bottlenecksor issues.

6. Performance Tuning: Implement necessary optimizations to improve the application'sperformance.

Performance Table:

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
Performance Testing	TC001	Response Time for Home Page	1. Access the home page of the e-commerce application.	Measure the response time.	Not Started	The home page loads within the specified response time threshold.		

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
			2. Use a performance testing tool to record the time.					
			3. Analyze therecorded data to determine response time.					
	TC002	Throughput during Peak Hours	1. Simulate peak-hour traffic on the application.	Measure the throughput.	Not Started	The application can handle peak-hour traffic without significant delays.		
			2. Execute performancetests during peak					

			hours.					
			3. Analyze thedata to determine the throughput.					
	TC003	Resource Utilization	1. Monitor CPU, memory, and network usage during testing.	Measure resource utilization.	Not Started	Resource utilizationremains within acceptablelimits.		
			2. Execute performancetests while monitoring resources.					

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
			3. Analyze thedata to assess resource utilization.					
	TC004	ConcurrentUsers	1. Simulate multiple concurrent users accessingthe app.	Measure app performance under load.	Not Started	The applicationremains stable and responsiveunder load.		
			2. Increase thenumber of concurrent users gradually.					
			3. Record the application's behavior with increased load.					
	TC005	Stress Testing	1. Apply maximum loadto test the system's breaking point.	Measure system behaviorunder extreme load.	Not Started	The system recovers gracefully after stress is removed.		
			2. Apply the maximum userload the application canhandle.					
			3. Observe the application's response understress.					

Process	No.	Test Case	Steps	Description	Status	Expected Result	Actual Result	Comment
	TC006	PerformanceTuning	1. Identify performance bottlenecks and areas of improvement.	Improve application performance.	Not Started	Performance bottlenecks are addressed and application performs better.		
			2. Analyze the performance test results.					
			3. Implement necessary optimizations.					

Explanation:

Performance testing helps to identify bottlenecks in the e-commerce application, ensuring it canhandle real-world user loads effectively.

Result:

Upon completion of the experiment, you will have performance test results and any optimizationsmade to improve the application's performance.

Experiment 7: Automate the testing of e-commerce applications using Selenium.

Aim:

The aim of this task is to automate the testing of an e-commerce web application (www.amazon.in) using Selenium WebDriver, which will help improve testing efficiency andreliability.

Algorithm:

1. Set up the environment:

   - Install Java Development Kit (JDK) and configure the Java environment variables.

   - Install an Integrated Development Environment (IDE) like Eclipse or IntelliJ.

   - Download Selenium WebDriver and the required web drivers for the browsers you intend totest (e.g., ChromeDriver, GeckoDriver for Firefox).

2. Create a new Java project in the IDE:

   - Set up a new Java project in the IDE and include the Selenium WebDriver library.

3. Develop test cases:

   - Identify the key functionalities and scenarios to test in the e-commerce application.

   - Design test cases covering various aspects like login, search, product details, add to cart,checkout, etc.

4. Implement Selenium automation scripts:

   - Write Java code using Selenium WebDriver to automate the identified test cases.

   - Utilize different Selenium commands to interact with the web elements, navigate throughpages, and perform various actions.

5. Execute the automated test cases:

   - Run the automated test scripts against the e-commerce application.

   - Observe the test execution and identify any failures or defects.

6. Analyze the test results:

- Review the test execution results to identify any failed test cases.
- Debug and fix any issues with the automation scripts if necessary.

7. Report defects:

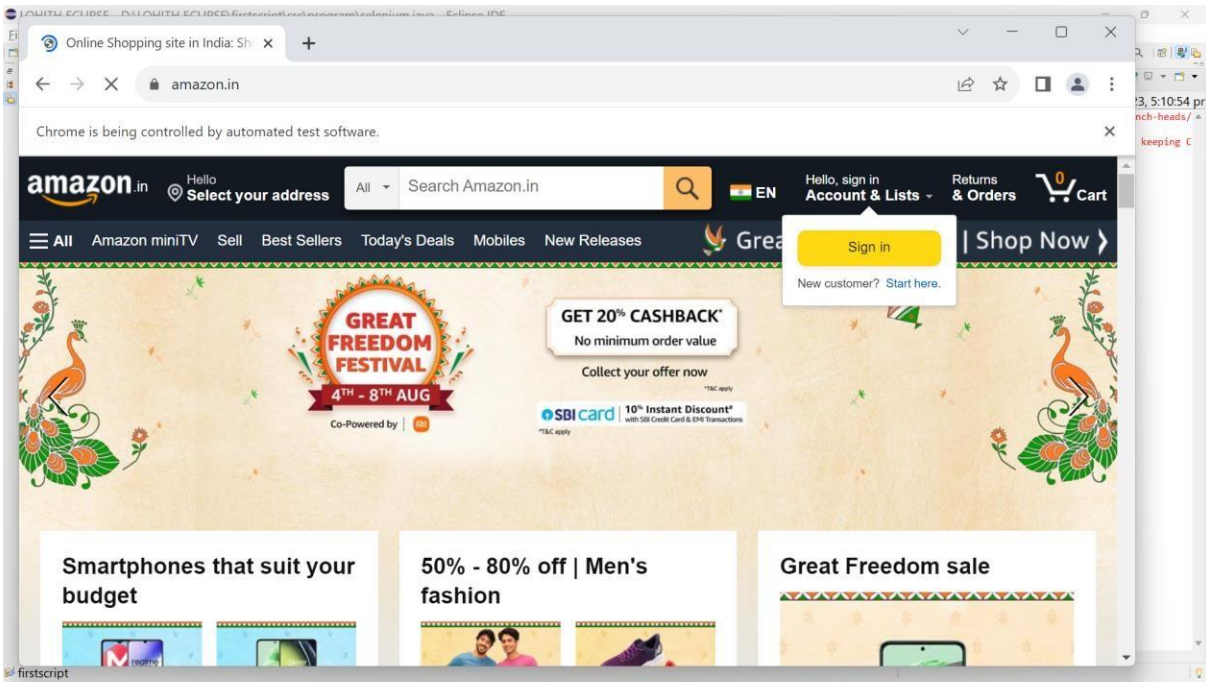
- Document any defects found during the automated testing process.
- Provide detailed information about each defect, including steps to reproduce and expectedresults.

Program:

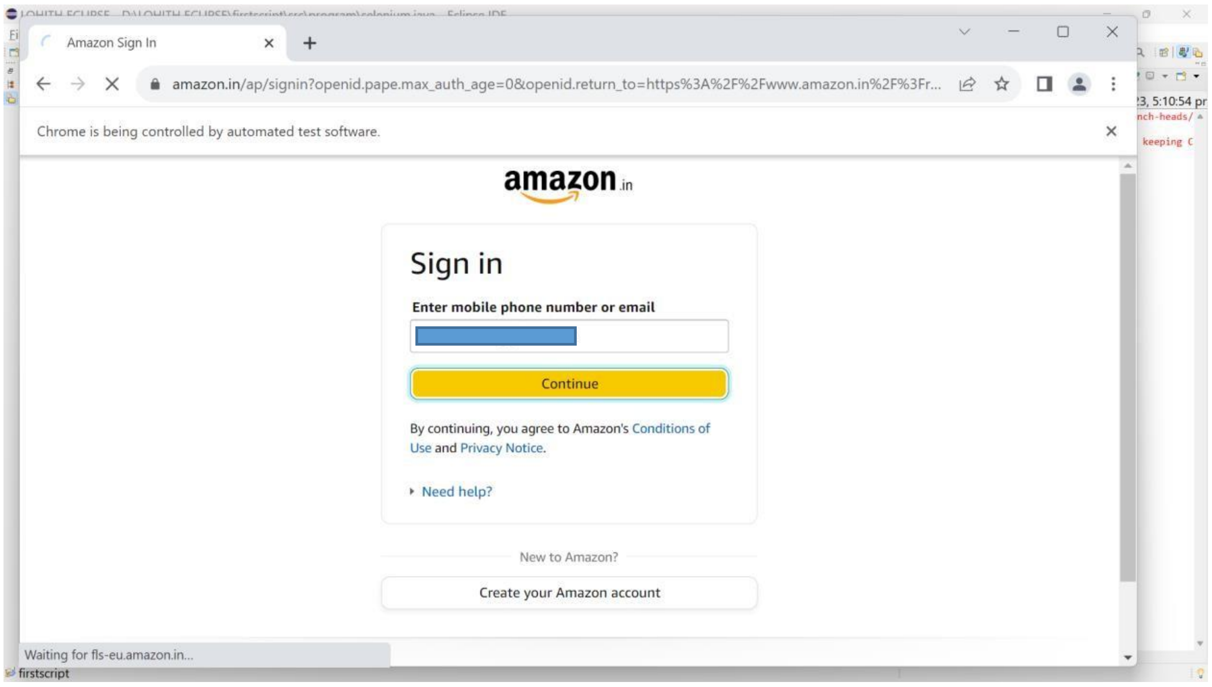
```
package program;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import
org.openqa.selenium.chrome.ChromeDriver;
public class selenium {
public static void main(String[] args)
{
System.setProperty("webdriver.chrome.driver","C:\\Users\\Admin\\Downloads\\chromedri
ver- win64\\chromedriver-win64\\chromedriver.exe");
WebDriver                    d=new
 ChromeDriver();
 d.get("https://www.amazon.in");
 d.findElement(By.xpath("//*[@id=\"nav-link-accountList\"]/span/span")).click();
d.findElement(By.id("ap_email")).sendKeys("youremail@gmail.com");
d.findElement(By.xpath("//*[@id=\"continue\"]")).click();
d.findElement(By.id("ap_password")).sendKeys("your                          password");
d.findElement(By.xpath("//*[@id=\"signInSubmit\"]")).click();
String                                                            u=d.getCurrentUrl();
if(u.equals("https://www.amazon.in/?ref_=nav_ya_signin"))
{
```

```
            System.out.println("Test Case Passed");
        }


else
        {
            System.out.println("Test Case Failed");
         }

        d.close();
        }
        }
```

Automation Process:



Chrome is being controlled by automated test software.



Amazon Sign In

amazon.in/ap/signin?openid.pape.max_auth_age=0&openid.return_to=https%3A%2F%2Fwww.amazon.in%2F%3Fr...

Chrome is being controlled by automated test software.

Sign in

Enter mobile phone number or email

Continue

By continuing, you agree to Amazon's Conditions of Use and Privacy Notice.

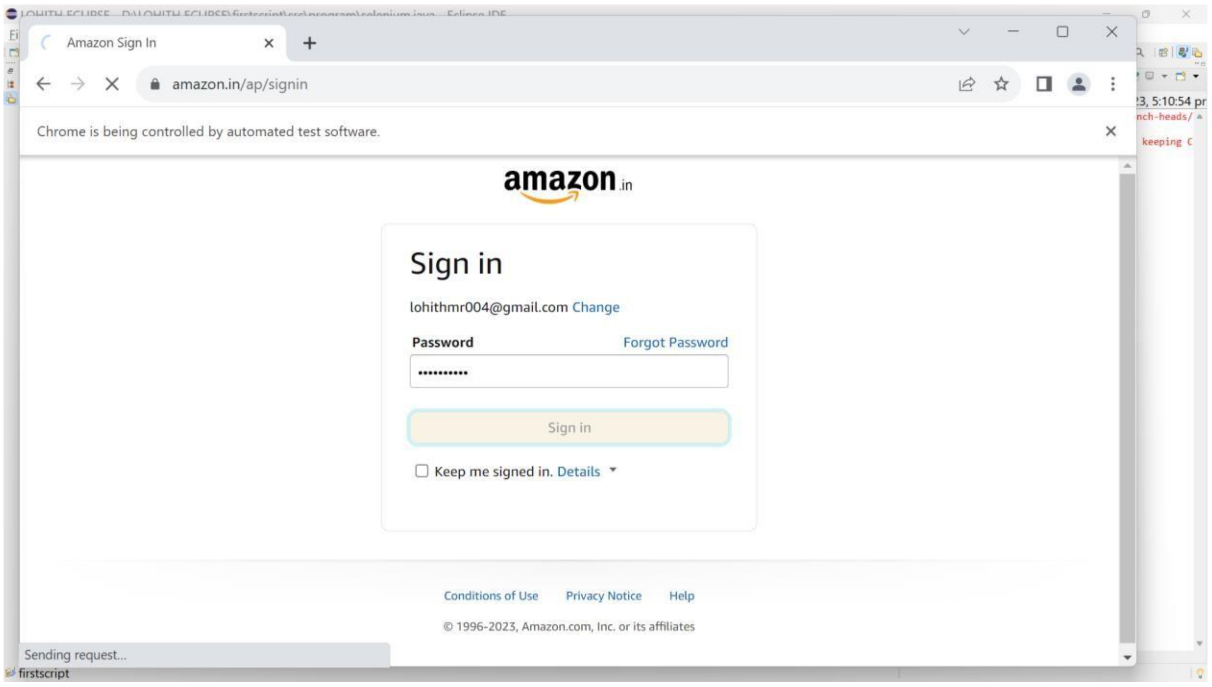
Need help?

New to Amazon?

Create your Amazon account

Waiting for fls-eu.amazon.in...

firstscript



Amazon Sign In

amazon.in/ap/signin

Chrome is being controlled by automated test software.

Sign in

lohithmr004@gmail.com Change

Password Forgot Password

••••••••••

Sign in

Keep me signed in. Details

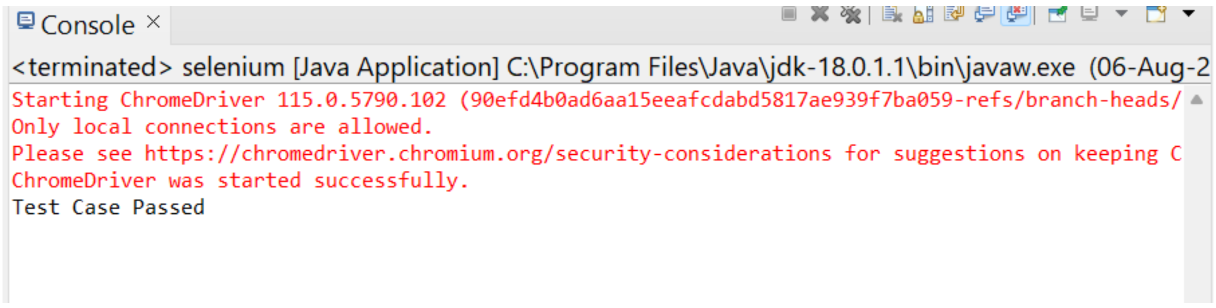
Conditions of Use Privacy Notice Help

© 1996-2023, Amazon.com, Inc. or its affiliates

Sending request...

firstscript

Console output:



```
<terminated> selenium [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe  (06-Aug-2
Starting ChromeDriver 115.0.5790.102 (90efd4b0ad6aa15eeafcdabd5817ae939f7ba059-refs/branch-heads/
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping C
ChromeDriver was started successfully.
Test Case Passed
```

Result:

The successful completion of this task will yield:

- Automated test scripts for the e-commerce application using Selenium WebDriver.
- Identification of defects, if any, in the application.

Experiment 8: Integrate TestNG with the above test automation.

Aim:

The aim of this task is to integrate TestNG with the existing Selenium automation scripts for thee-commerce application, enhancing test management, parallel execution, and reporting capabilities.

Algorithm:

1. Set up TestNG in the project:

  - Add TestNG library to the existing Java project.

2. Organize test cases using TestNG annotations:

  - Add TestNG annotations (@Test, @BeforeTest, @AfterTest, etc.) to the existing test cases.

  - Group similar test cases using TestNG's grouping mechanism.

3. Implement data-driven testing (optional):

  - Utilize TestNG's data providers to implement data-driven testing if required.

4. Configure TestNG test suite:

  - Create an XML configuration file for TestNG to define test suites, test groups, and otherconfigurations.

5. Execute the automated test cases using TestNG:

  - Run the automated test suite using TestNG.

  - Observe the test execution and identify any failures or defects.

6. Analyze the test results:

  - Review the TestNG-generated test reports to identify any failed test cases.

  - Utilize TestNG's reporting capabilities to understand the test execution status.

7. Report defects (if any):

  - Document any defects found during the automated testing process.

- Provide detailed information about each defect, including steps to reproduce and expectedresults.

Program Code (Program1.java) :

```
package mytest;

import java.time.Duration;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.Assert;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class Program1 {

        WebDriver driver;



        @BeforeMethod

        public void setUp()

        {

        System.setProperty("webdriver.chrome.driver","C:\\selenium\\chromedriver_win32\\c
        hro

         medriver.exe");

        driver=new ChromeDriver();

        driver.get("https://amazon.in");
```

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

            driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

            }


        @Test

        public void verifyTitle()

        {

        String actualTitle=driver.getTitle();
```

Program Code (testng.xml) :

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

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

<suite name="Suite">

  <test name="Test">

  <classes>

  <class name="mytest.Program1"></class>

  </classes>

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

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

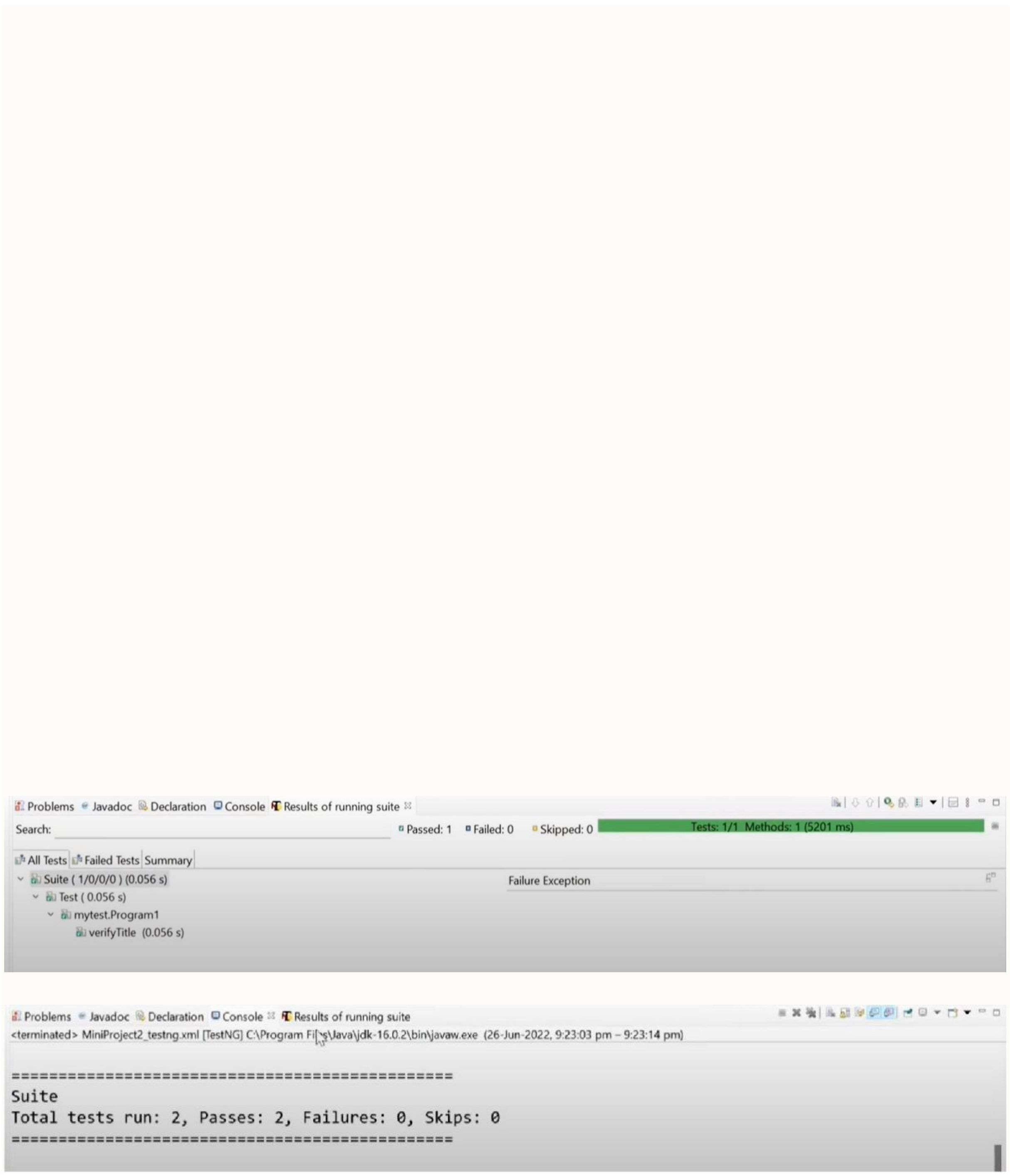
```
        String expectedTitle="Online Shopping site in India: Shop Online for Mobiles, Books,
        Watches, Shoes and More - Amazon.in";

        Assert.assertEquals(actualTitle, expectedTitle);

        }

        @Test

        public void verifyLogo()

        {

        boolean flag=driver.findElement(By.xpath("//a[@id='nav-logo-sprites']")).isDisplayed();

        Assert.assertTrue(flag);

        }

        @AfterMethod

        public void tearDown()

        {

        driver.quit();

        }

}
```

Program Code (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>MiniProject2</groupId>
```

```
 <artifactId>MiniProject2</artifactId>

 <version>0.0.1-SNAPSHOT</version>

 <dependencies>

 <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->

<dependency>

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

   <artifactId>selenium-java</artifactId>

   <version>4.3.0</version>

</dependency>

 </dependencies>

 <build>

   <sourceDirectory>src</sourceDirectory>

   <plugins>

     <plugin>

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

       <version>3.8.1</version>

       <configuration>

         <release>16</release>

       </configuration>

     </plugin>

   </plugins>

</build> </project>
```

Output:



```
Suite ( 1/0/0/0 ) (0.056 s)
  Test ( 0.056 s)
    mytest.Program1
      verifyTitle (0.056 s)
```

```
<terminated> MiniProject2_testng.xml [TestNG] C:\Program Files\Java\jdk-16.0.2\bin\javaw.exe (26-Jun-2022, 9:23:03 pm – 9:23:14 pm)

===============================================
Suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

Result:

The successful completion of this task will yield:

- Integration of TestNG with the existing Selenium automation scripts.
- Enhanced test management and reporting capabilities.
- Identification of defects, if any, in the application and improved efficiency in handling testscenarios.