

# Centre for Distance and Online Education

Vidya Nagari, Kalina, Santacruz East – 400098.

CERTIFICATE

This is to certify that **Ms. Omkar Balu Auti** of (MCA) Semester III has completed the specified term work in the subject of **Software Testing and Quality Assurance Lab** satisfactorily within this institute as laid down by University of Mumbai during the academic year 2023 to 2024.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Subject In-charge | External Examiner | Coordinator – M.C.A |

INDEX

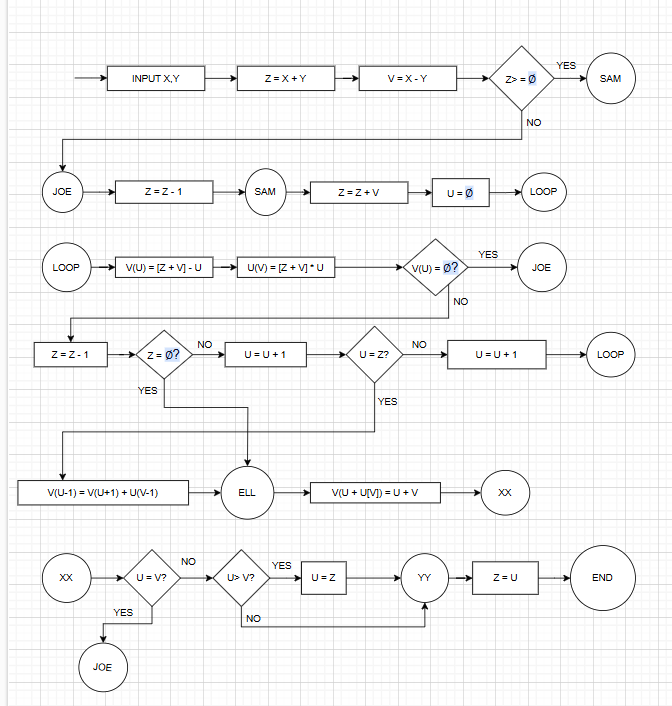
# Subject: Software Testing and Quality Assurance Lab

|  |  |  |  |
| --- | --- | --- | --- |
| Sr.  No. | Experiment Name | Sign. | Remarks |
| 1 | **Practical 1**  Flow graphs and Path Testing using Some elements |  |  |
| 2 | **Practical 2**  Installation of Selenium IDE & case test using Selenium IDE |  |  |
| 3 | **Practical 3**  Implementing handling multiple frames |  |  |
| 4 | **Practical 4**  Implementing Selenium WebDriver- find element command, Locator(id,css selector, Xpath), Input Box, Buttons, Submit Buttons. |  |  |
| 5 | **Practical 5**  Implementing methods in TestNG file |  |  |
| 6 | **Practical 6**  Select Value from DropDown using Selenium Webdriver. |  |  |
| 7 | **Practical 7**  Demonstrate CheckBox and Radio Button in Selenium WebDriver and Testng. |  |  |
| 8 | **Practical 8**  Implementing Selenium WebDriver – find element command,  Locator(Class name, css selector, Xpath) |  |  |
| 9 | **Practical 9**  Demonstrate login form in Selenium WebDriver and Testng. |  |  |
| 10 | **Practical 10**  Preparing a Report on test case for given scenario |  |  |

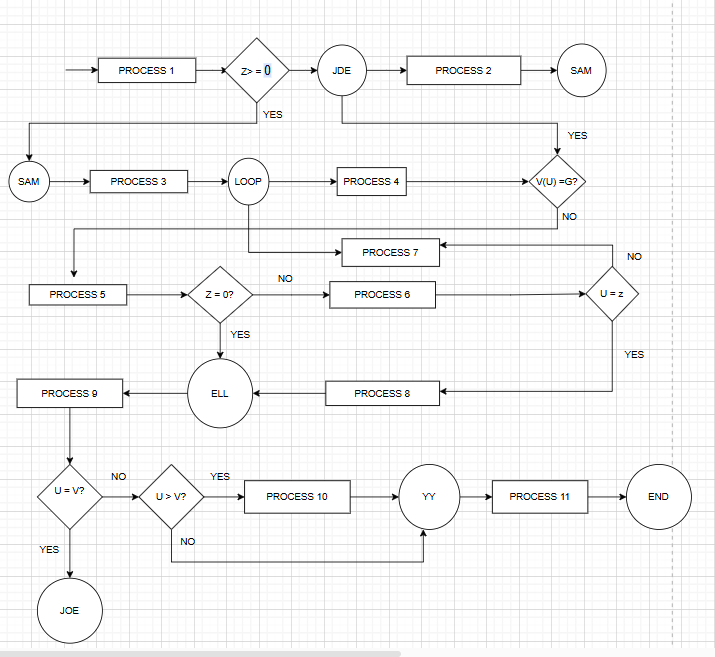
**PRACTICAL 1**

# OUTPUT:

one-to-one flowchart :



Control Flowgraph :



# PRACTICAL 2

**CODE:**

Testdemo.java

**package** com.testselenium;

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

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

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

**public class** Testdemo { @Test

**public void** myTest()

{

System.***out***.println("hello World");

System.*setProperty*("webdriver.chrome.driver","./drivers/chromedriver.exe"); WebDriver driver = **new** ChromeDriver(); driver.get("https://mu.ac.in/distance-open-learning");

}

}

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>testing</groupId>

<artifactId>test</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>3.141.5</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.testng/testng -->

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.14.3</version>

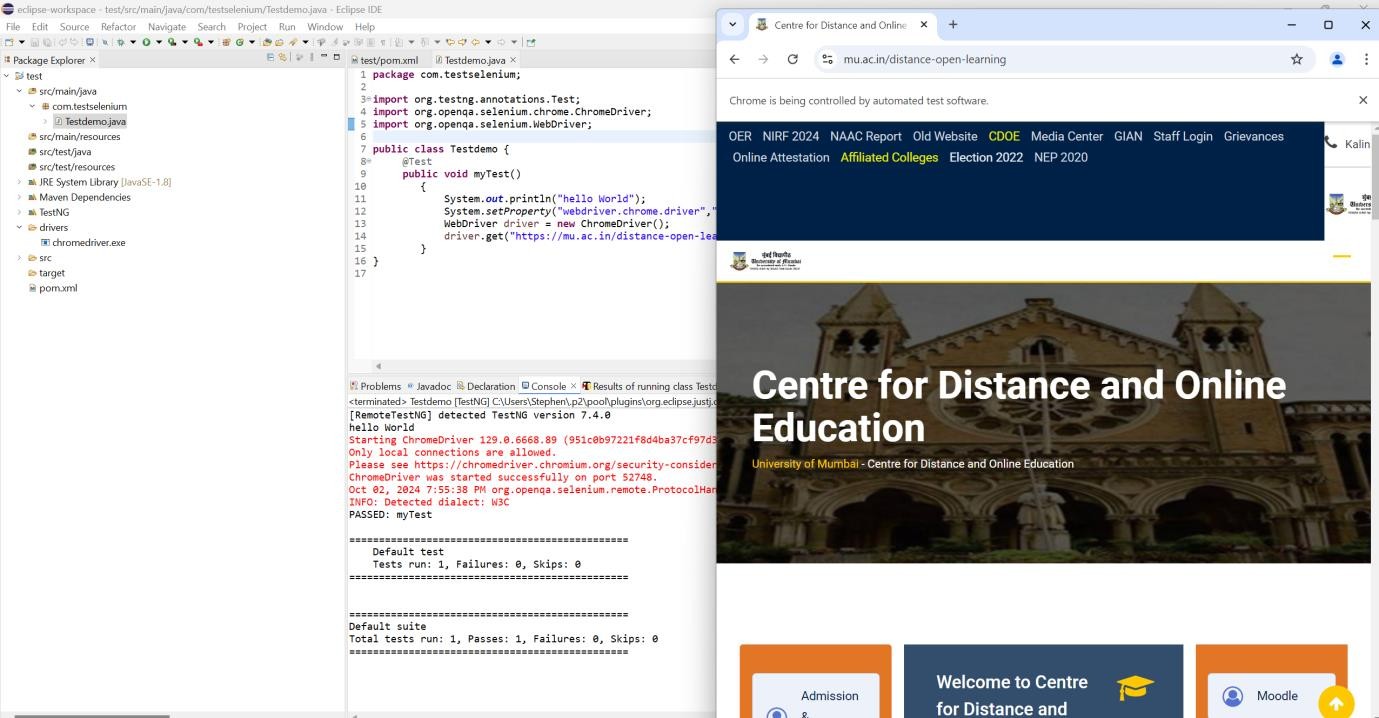
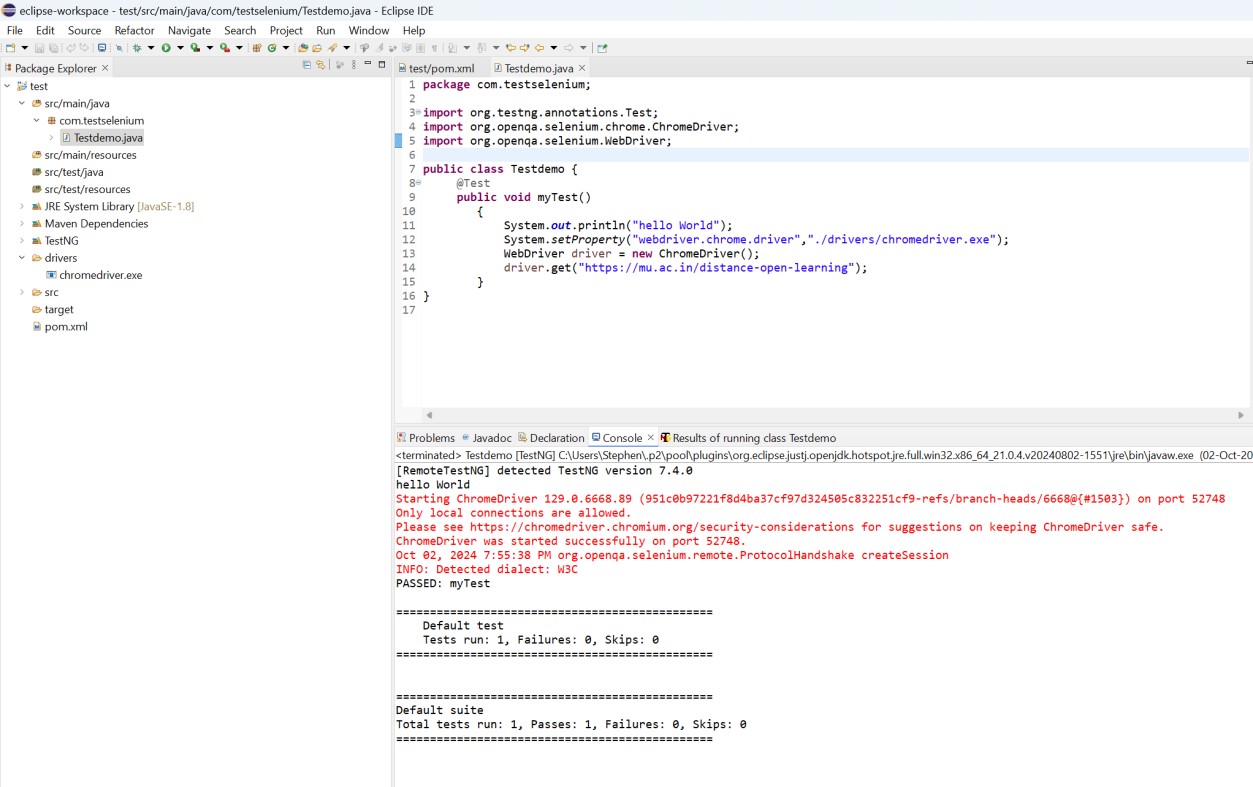
<scope>test</scope>

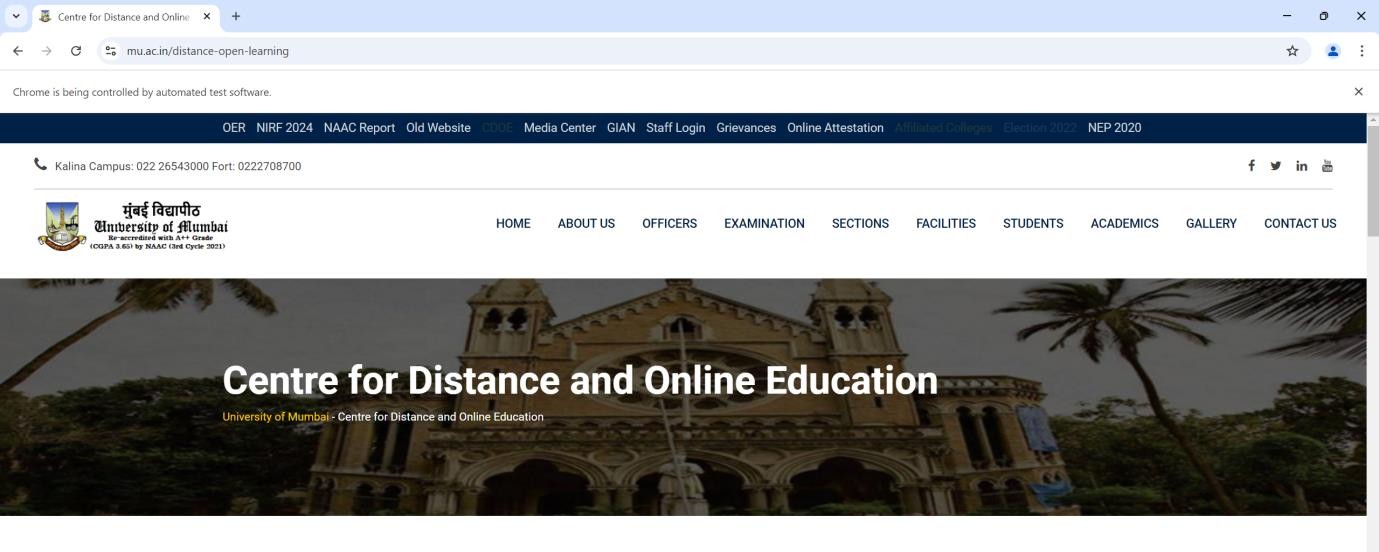
</dependency>

</dependencies>

</project>

# OUTPUT:





**PRACTICAL 3**

# CODE:

Testdemo2.java

**package** com.testselenium;

**import** java.util.ArrayList; **import** java.util.List; **import** java.util.Set;

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

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

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

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

**public class** Testdemo2 {

@Test

**public void** myTest()

{

//System.out.println("hello World"); System.*setProperty*("webdriver.chrome.driver","./drivers/chromedriver.exe"); WebDriver driver = **new** ChromeDriver(); driver.get("https:kjsit.somaiya.edu.in/en/"); System.***out***.println("Parent:"+driver.getTitle());

String parentwindow = driver.getWindowHandle(); System.***out***.println(parentwindow); driver.findElement(By.*linkText*("B.Tech.(IT)")).click();

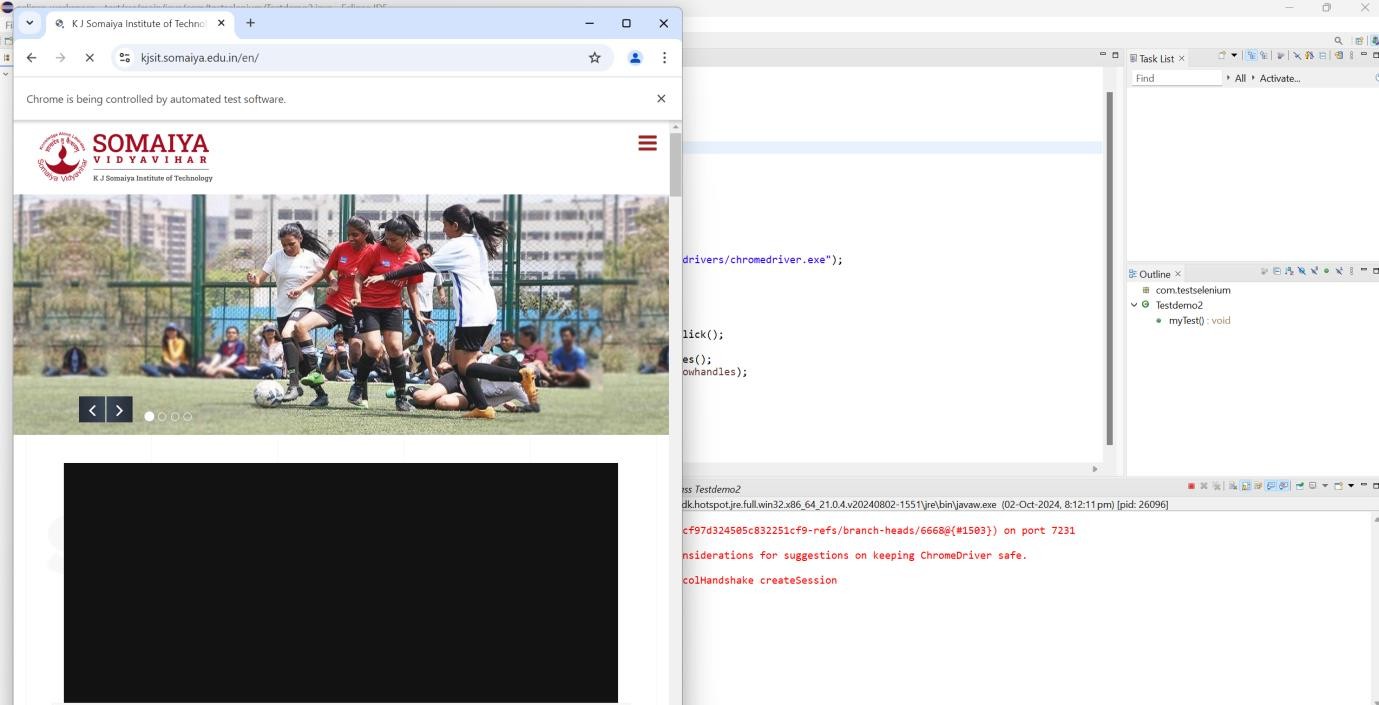
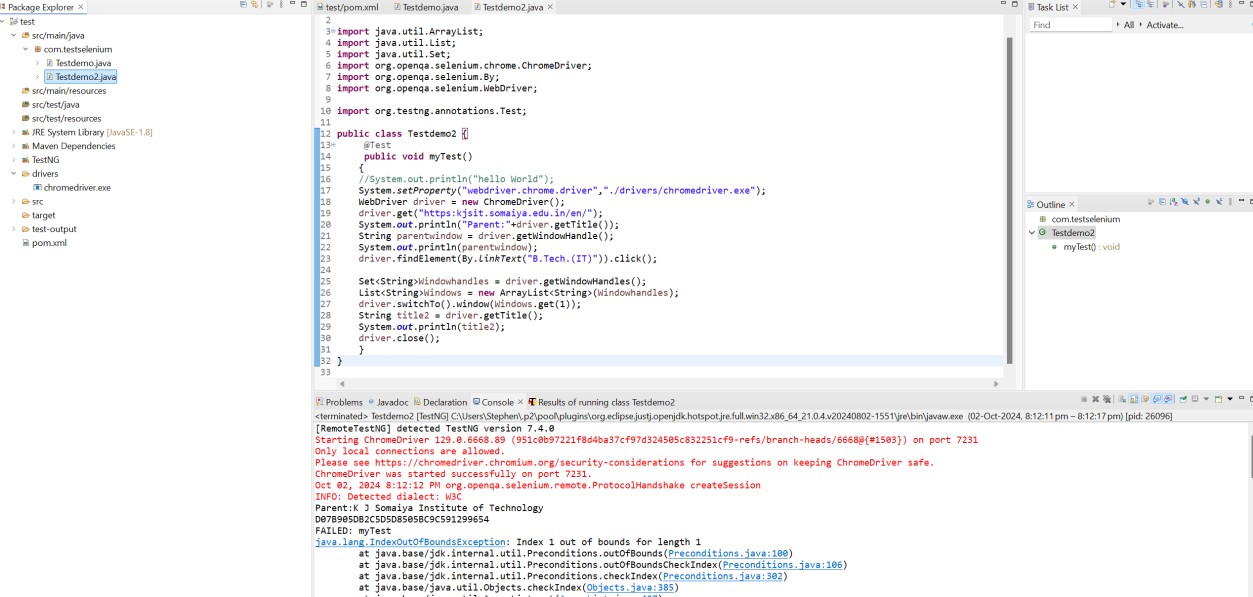
Set<String>Windowhandles = driver.getWindowHandles(); List<String>Windows = **new** ArrayList<String>(Windowhandles); driver.switchTo().window(Windows.get(1));

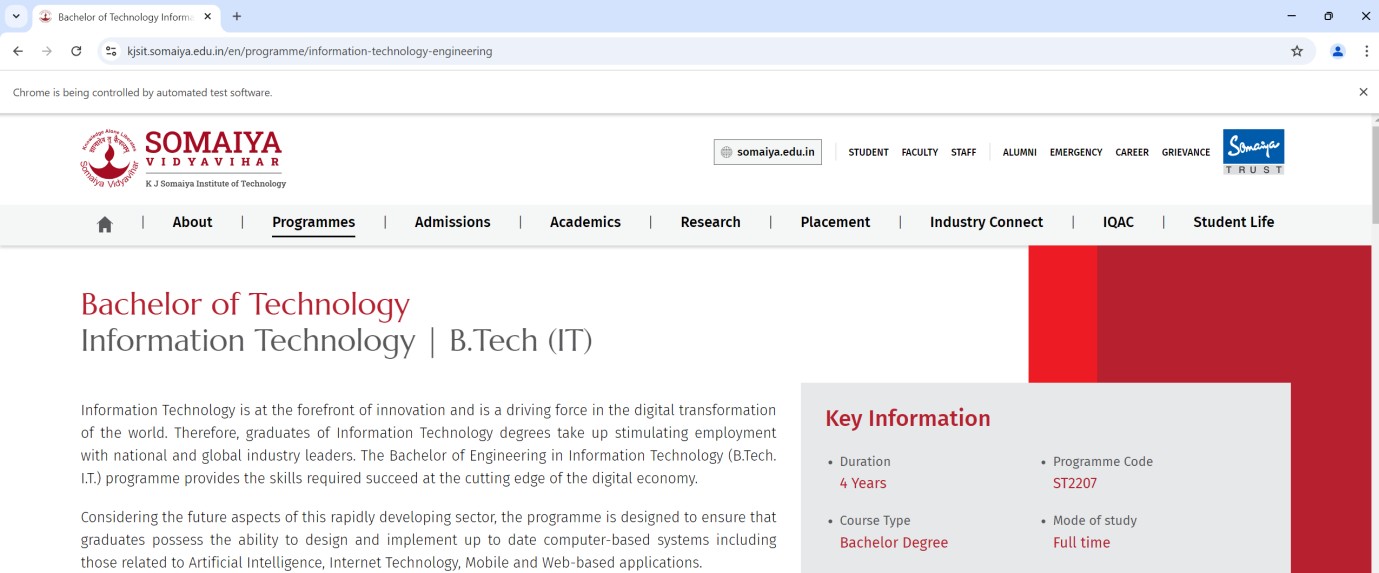
String title2 = driver.getTitle(); System.***out***.println(title2); driver.close();

}

}

# OUTPUT:





**PRACTICAL 4**

# CODE:

**package** com.testselenium;

**import** org.openqa.selenium.\*;

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

**public class** Testdemo3 {

**public static void** main(String args[]) **throws** InterruptedException { System.*setProperty*("webdriver.chrome.driver", "./drivers/chromedriver.exe"); ChromeDriver driver = **new** ChromeDriver(); driver.get("https://letcode.in/frame");

//frame

//driver.switchTo().frame("firstFr");

//driver.switchTo().frame(0);

WebElement frameEle = driver.findElementByXPath("//\*[@id=\"firstFr\"]"); driver.switchTo().frame(frameEle); driver.findElement(By.*name*("fname")).sendKeys("Idol"); driver.findElement(By.*name*("lname")).sendKeys("Mca");

Thread.*sleep*(2000);

//nested frame

WebElement innerFrames = driver.findElement(By.*cssSelector*("iframe.has- background-white"));

driver.switchTo().frame(innerFrames); driver.findElement(By.*name*("email")).sendKeys("[idol.mca@somaiya.edu"](mailto:idol.mca@somaiya.edu));

Thread.*sleep*(2000);

//parent Frame driver.switchTo().parentFrame();

driver.findElement(By.*name*("fname")).sendKeys(" Sy ");

Thread.*sleep*(2000);

//default frame driver.switchTo().defaultContent(); driver.findElement(By.*linkText*("Log in")).click();

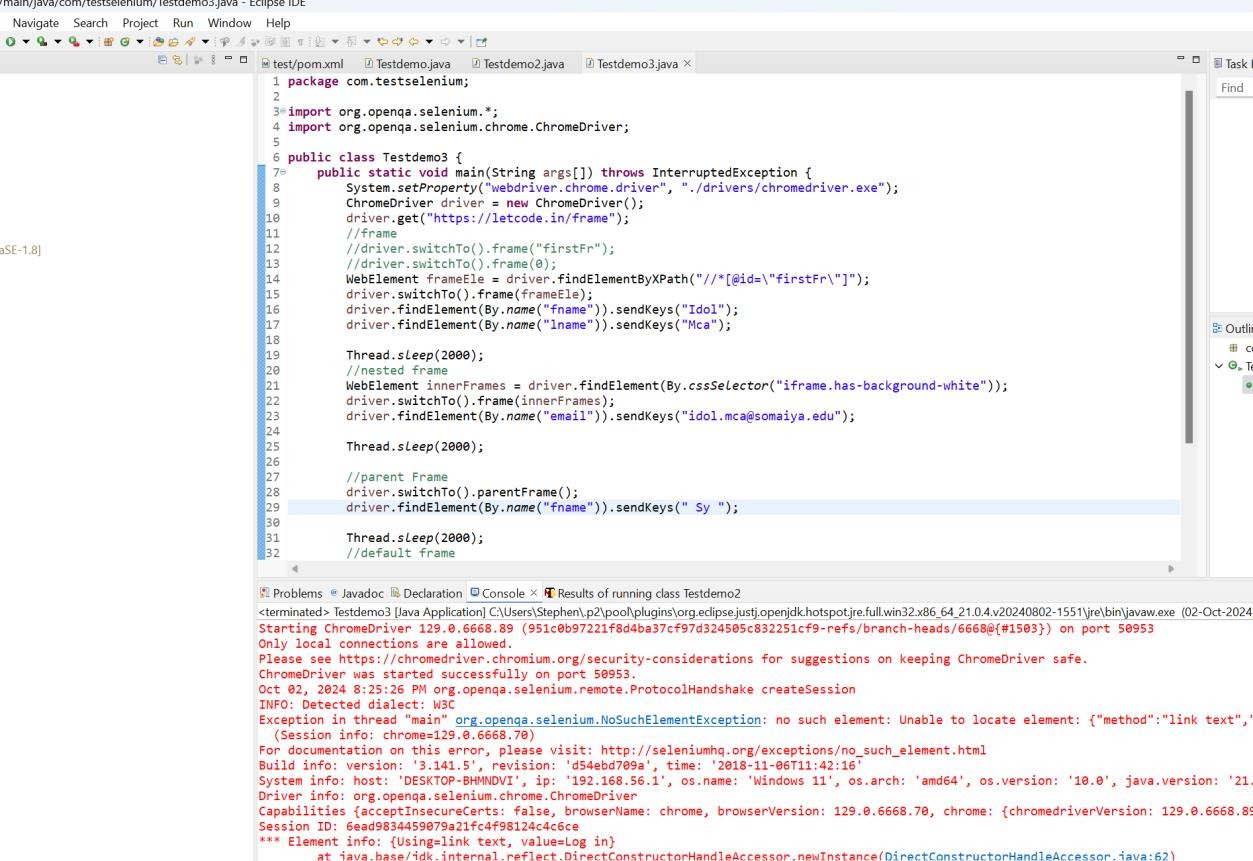
Thread.*sleep*(2000);

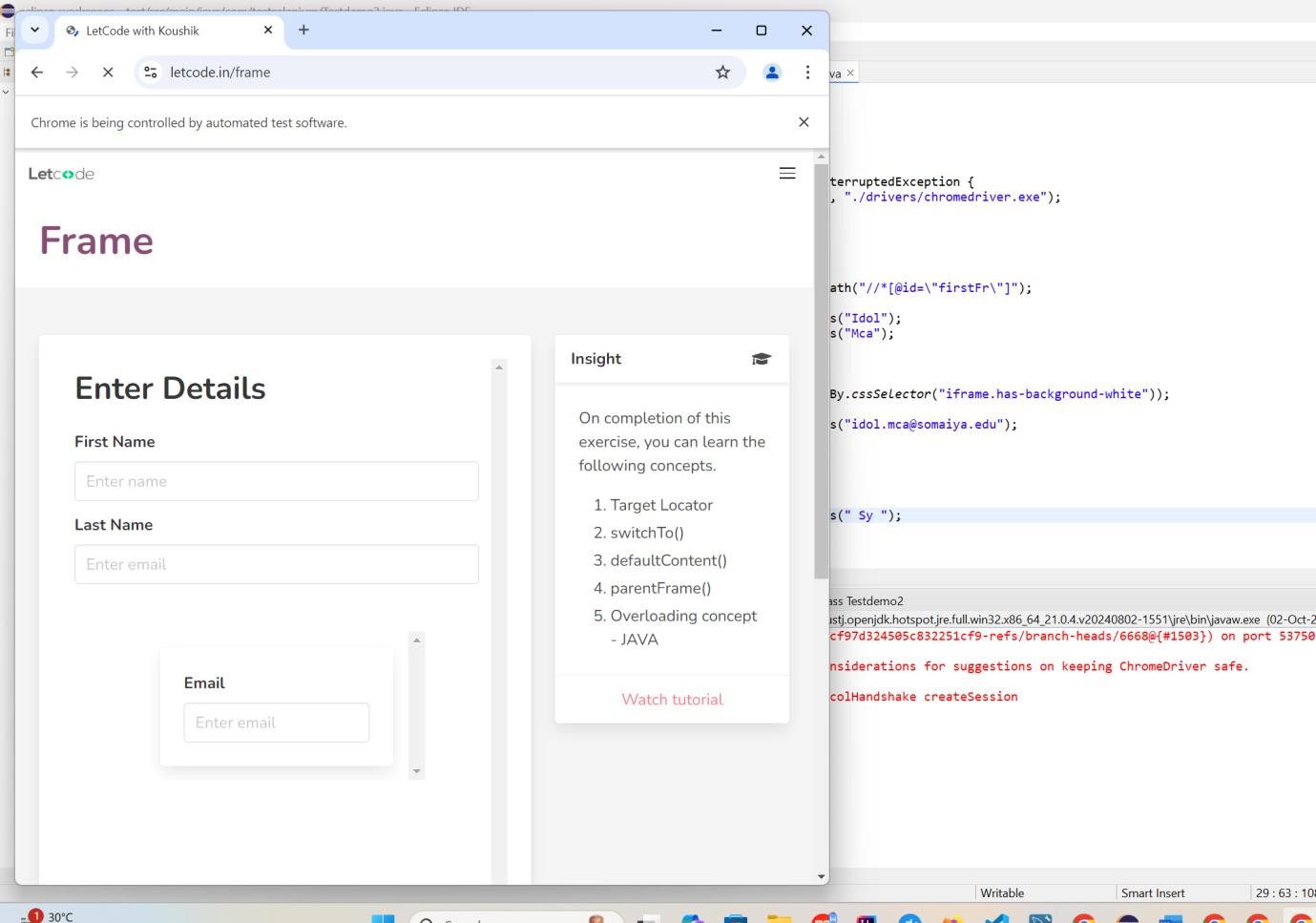
// driver.quit();

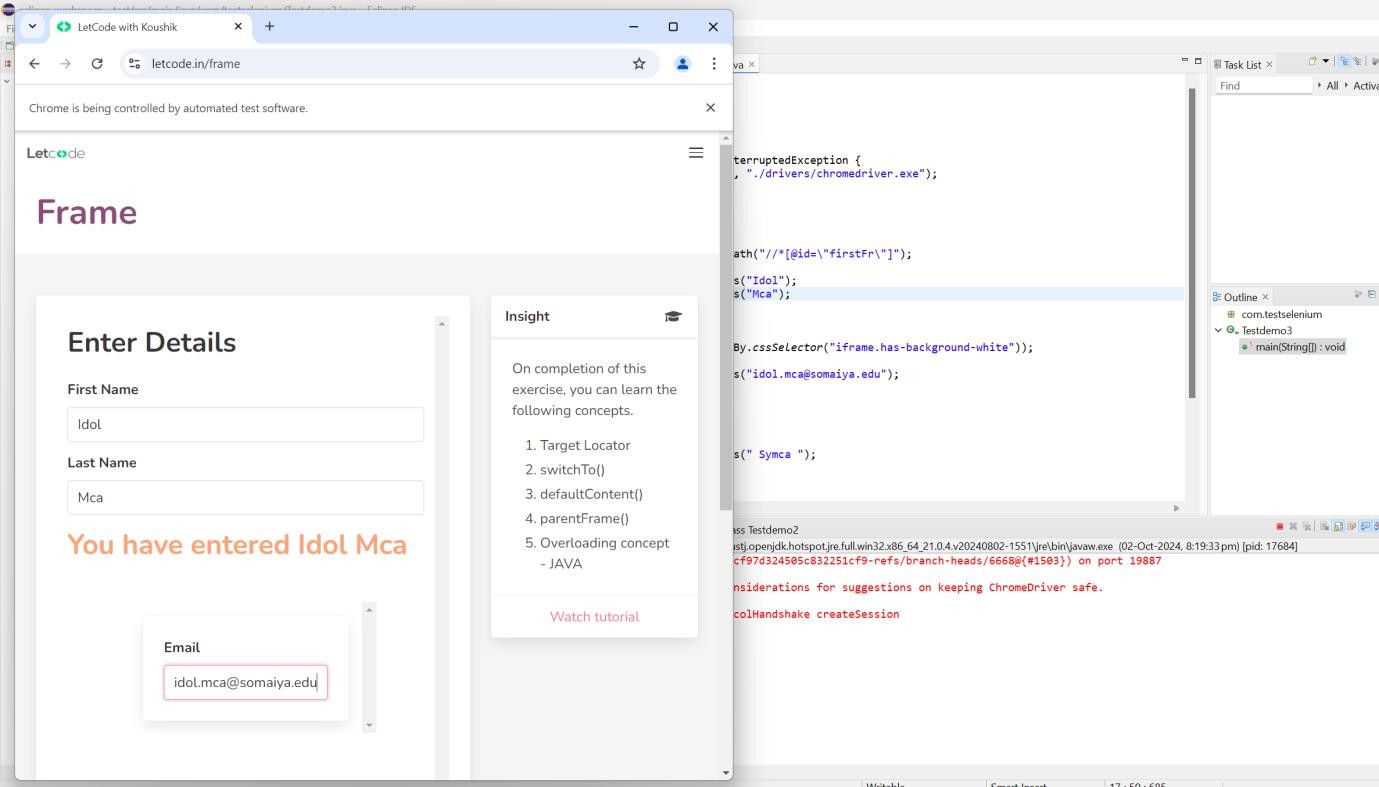
}

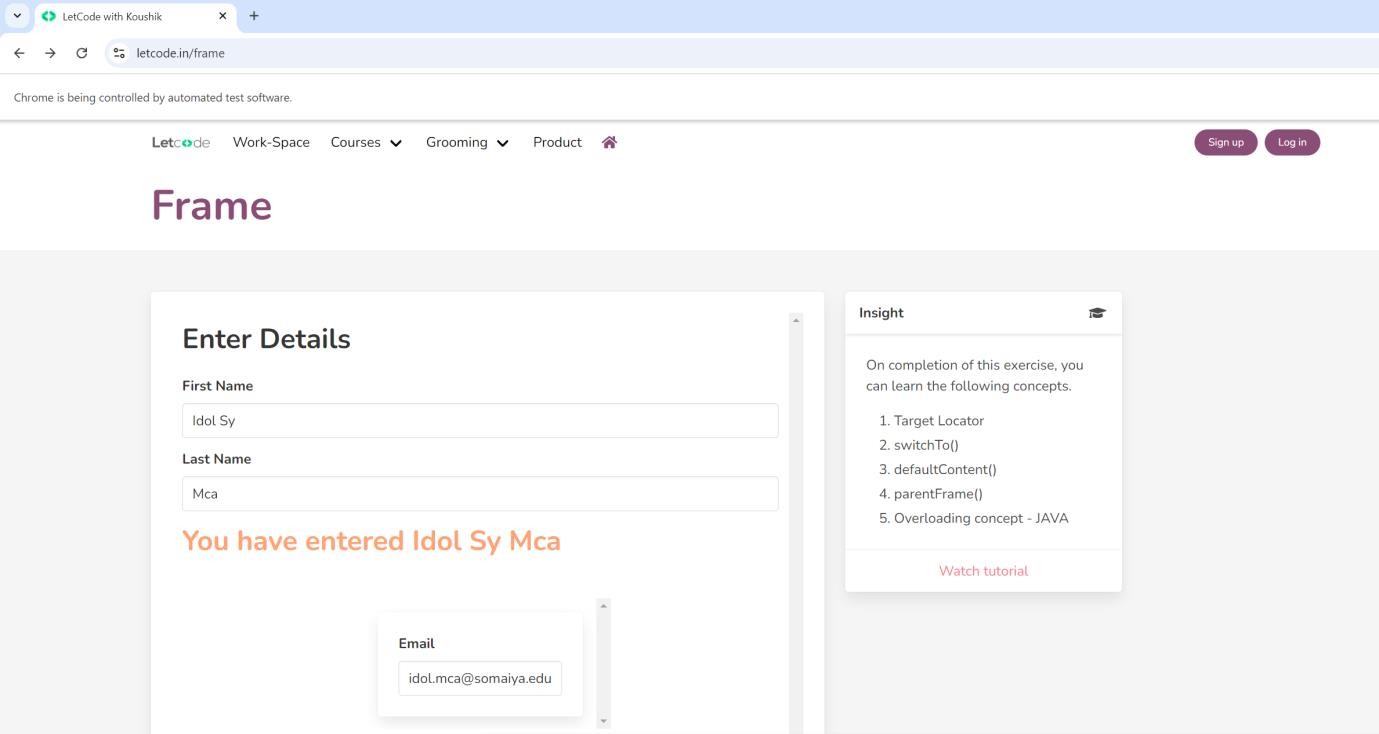
}

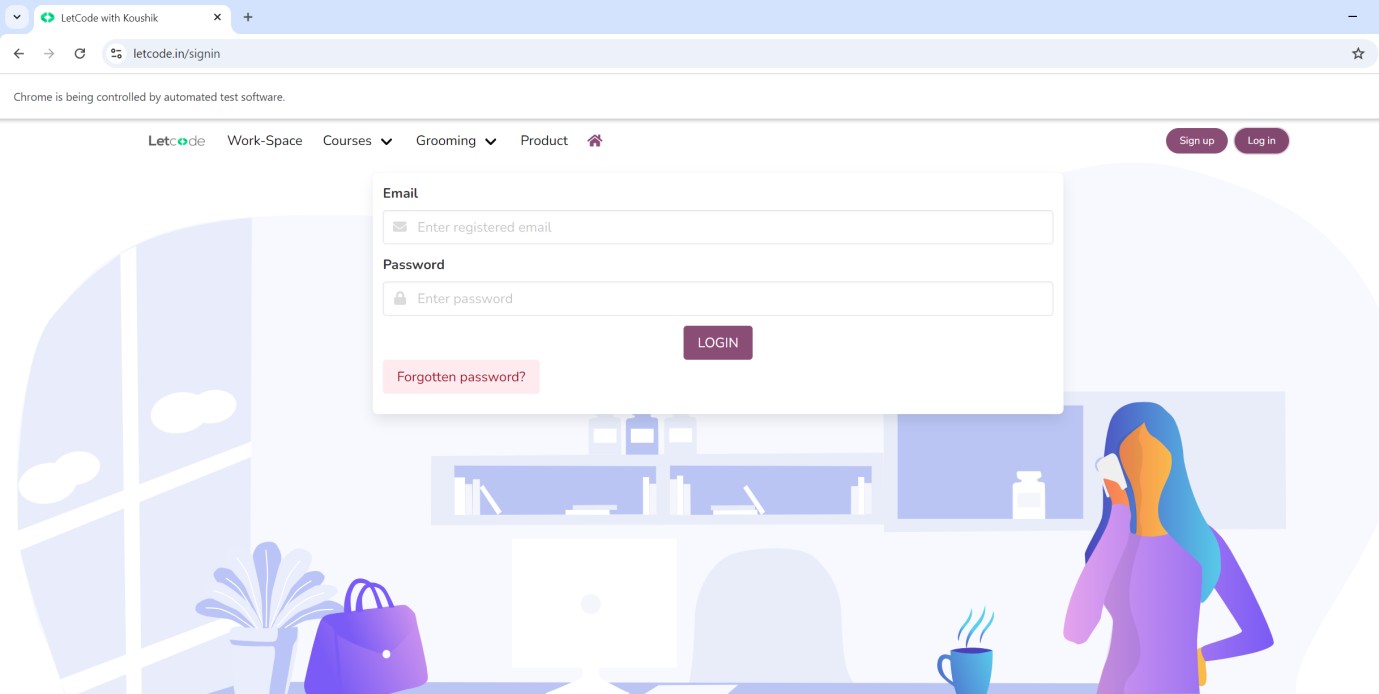
# OUTPUT:











**PRACTICAL 5**

# CODE:

**package** com.testselenium;

**import** org.testng.annotations.AfterClass; **import** org.testng.annotations.AfterMethod; **import** org.testng.annotations.AfterSuite; **import** org.testng.annotations.AfterTest; **import** org.testng.annotations.BeforeClass; **import** org.testng.annotations.BeforeMethod; **import** org.testng.annotations.BeforeSuite; **import** org.testng.annotations.BeforeTest; **import** org.testng.annotations.Test;

**public class** Testdemo4 { @Test

**public void** f() {

System.***out***.println("First Test");

}

@BeforeMethod

**public void** beforemethod() { System.***out***.println("Before Method");

}

@AfterMethod

**public void** aftermethod() { System.***out***.println("After Method");

}

@BeforeClass

**public void** beforeclass() { System.***out***.println("Before Class");

}

@AfterClass

**public void** afterclass() { System.***out***.println("After Class");

}

@BeforeTest

**public void** beforeTest()

{

System.***out***.println("Before Test");

}

@AfterTest

**public void** afterTest()

{

System.***out***.println("After Test");

}

@BeforeSuite

**public void** beforeSuite()

{

System.***out***.println("Before Suite");

}

@AfterSuite

**public void** afterSuite()

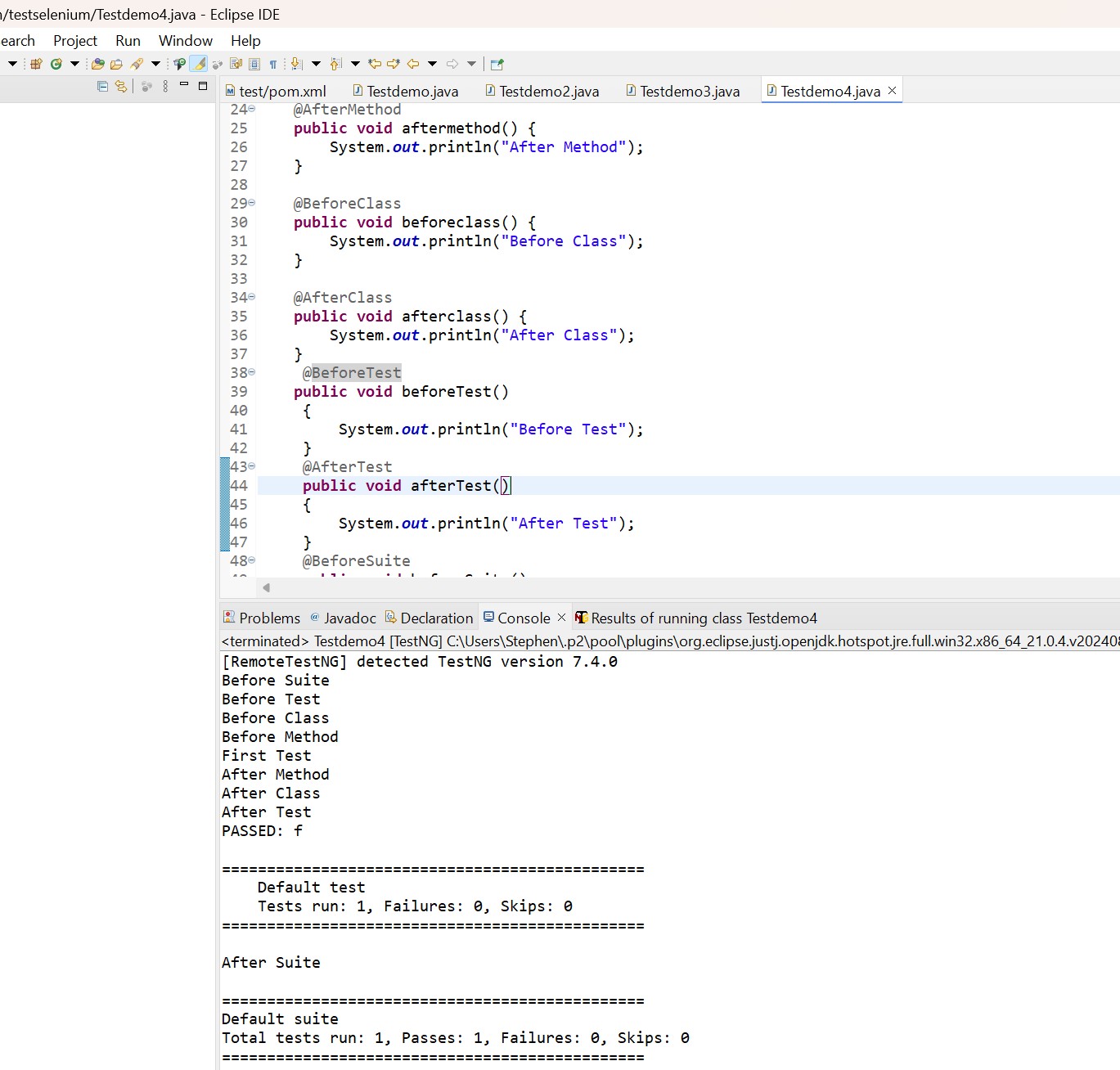
{

System.***out***.println("After Suite");

}

}

# OUTPUT:



**PRACTICAL 6**

# CODE:

**package** com.testselenium;

**import** org.openqa.selenium.WebDriver.Options; **import** org.openqa.selenium.WebDriver.Window; **import** org.openqa.selenium.chrome.ChromeDriver; **import** org.openqa.selenium.support.ui.Select;

**public class** Testdemo5 {

**public static void** main(String args[]) **throws** InterruptedException { System.*setProperty*("webdriver.chrome.driver", "./drivers/chromedriver.exe"); ChromeDriver driver = **new** ChromeDriver();

Options manage = driver.manage(); Window window = manage.window(); window.maximize(); driver.get("https://blazedemo.com/"); Thread.*sleep*(1000);

Select departure = **new** Select(driver.findElementByName("fromPort")); Select destination = **new** Select(driver.findElementByName("toPort")); departure.selectByVisibleText("Boston"); destination.selectByVisibleText("London");

Thread.*sleep*(2000);

driver.findElementByCssSelector("body > div.container > form > div >

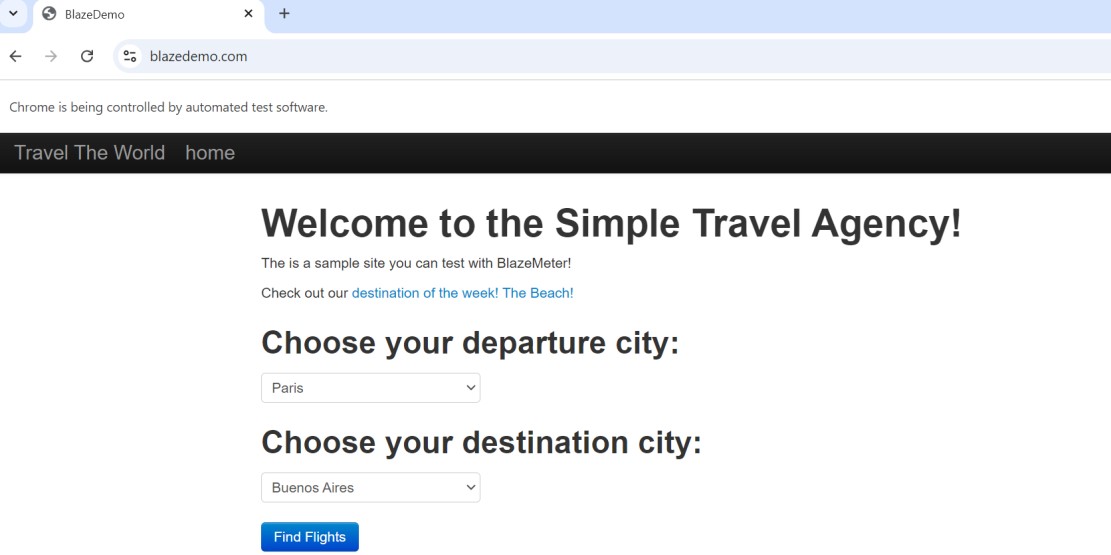
input").click();

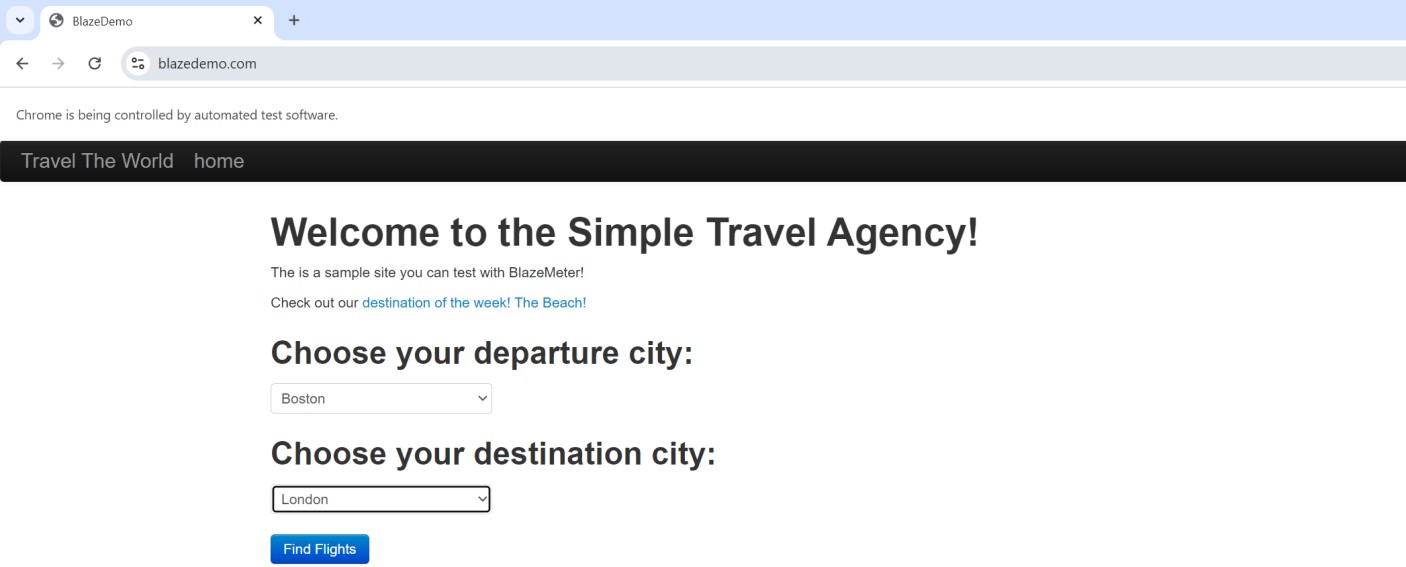
}

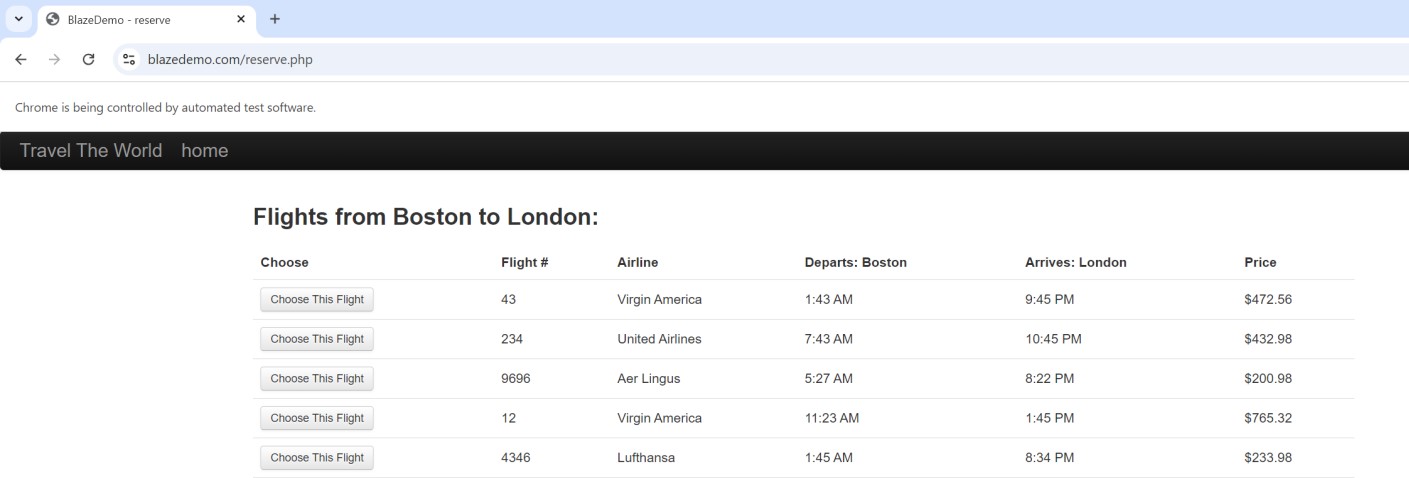
Thread.*sleep*(2000); driver.close();

}

# OUTPUT:







**PRACTICAL 7**

# CODE:

**package** com.testselenium;

**import** org.openqa.selenium.WebDriver.Options; **import** org.openqa.selenium.WebDriver.Window; **import** org.openqa.selenium.WebElement;

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

**public class** Testdemo6 {

**public static void** main(String args[]) **throws** InterruptedException { System.*setProperty*("webdriver.chrome.driver", "./drivers/chromedriver.exe"); ChromeDriver driver = **new** ChromeDriver();

Options manage = driver.manage(); Window window = manage.window(); window.maximize();

driver.get("https:/[/www](http://www.htmldog.com/examples/inputcheckboxes.html).[htmldog.com/examples/inputcheckboxes.html"](http://www.htmldog.com/examples/inputcheckboxes.html)); WebElement drama = driver.findElementByName("drama"); WebElement action = driver.findElementByName("action"); WebElement comedy = driver.findElementByName("comedy"); WebElement horror = driver.findElementByName("horror"); WebElement scifi = driver.findElementByName("scifi");

comedy.click(); scifi.click();

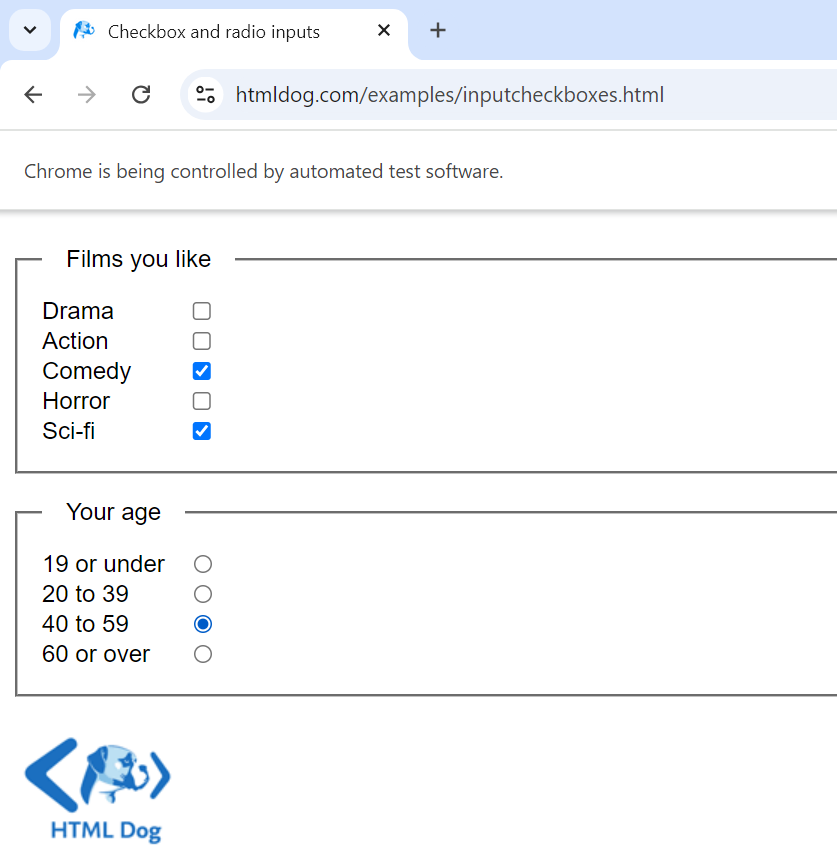
WebElement lt20 = driver.findElementById("lt20"); WebElement gt20tolt40 = driver.findElementById("20to39"); WebElement gt40tolt59 = driver.findElementById("40to59"); WebElement gt59 = driver.findElementById("gt59");

gt40tolt59.click(); Thread.*sleep*(3000); driver.close();

}

}

# OUTPUT:



**PRACTICAL 8**

# CODE:

**package** com.testselenium;

**import** org.openqa.selenium.WebDriver.Options; **import** org.openqa.selenium.WebDriver.Window; **import** org.openqa.selenium.chrome.ChromeDriver;

**public class** Testdemo7 {

**public static void** main(String args[]) **throws** InterruptedException { System.*setProperty*("webdriver.chrome.driver", "./drivers/chromedriver.exe"); ChromeDriver driver = **new** ChromeDriver();

Options manage = driver.manage(); Window window = manage.window(); window.maximize(); driver.get("https:/[/www](http://www.google.co.in/).[google.co.in"](http://www.google.co.in/));

//driver.findElementByClassName("gLFyf").sendKeys("Mumbai University");

//driver.findElementByCssSelector("#APjFqb").sendKeys("Mumbai

University"); Mumbai");

driver.findElementByXPath("//\*[@id=\"APjFqb\"]").sendKeys("University of driver.findElementByCssSelector("body > div.L3eUgb >

div.o3j99.ikrT4e.om7nvf > form > div:nth-child(1) > div.A8SBwf > div.FPdoLc.lJ9FBc > center > input.gNO89b").click();

driver.findElementByXPath("//\*[@id=\"rso\"]/div[1]/div/div/div/div/div/div/div/div[1

]/div/span/a").click();

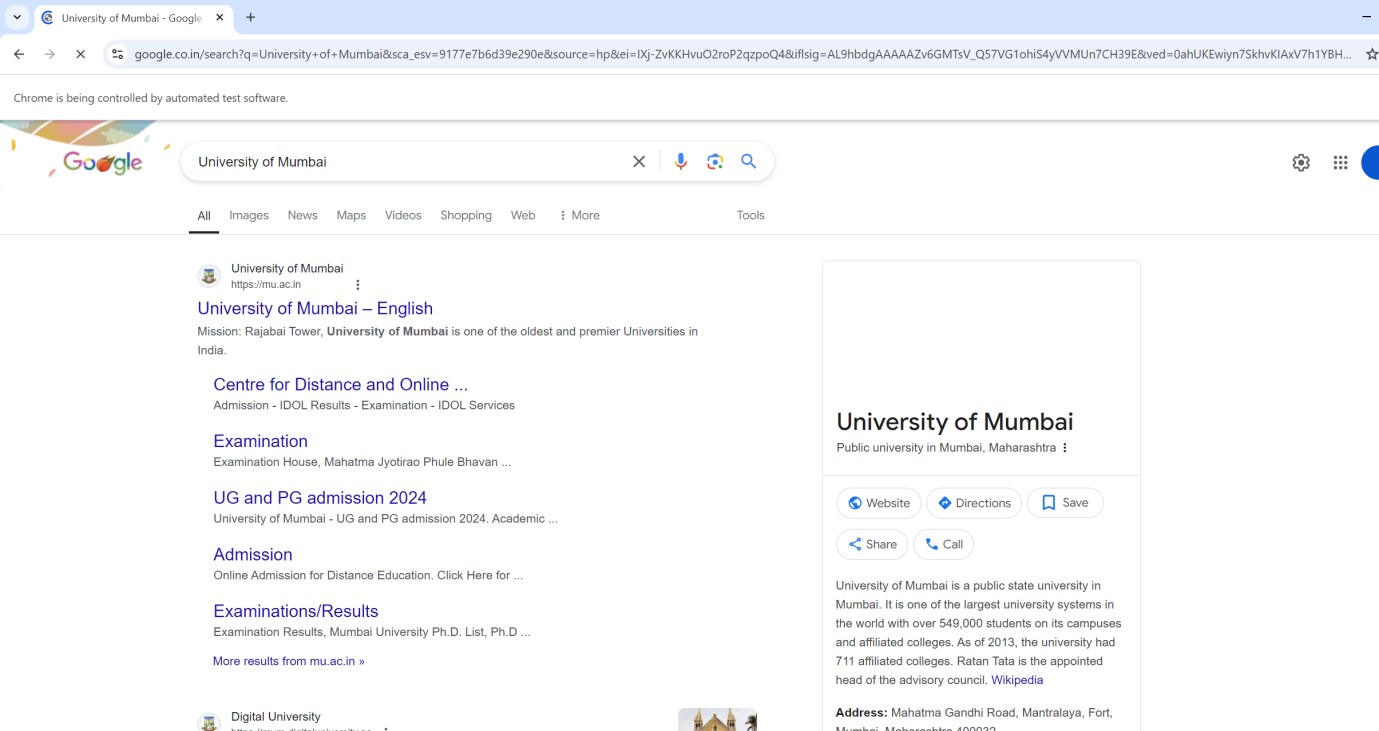
Thread.*sleep*(2000); driver.close();

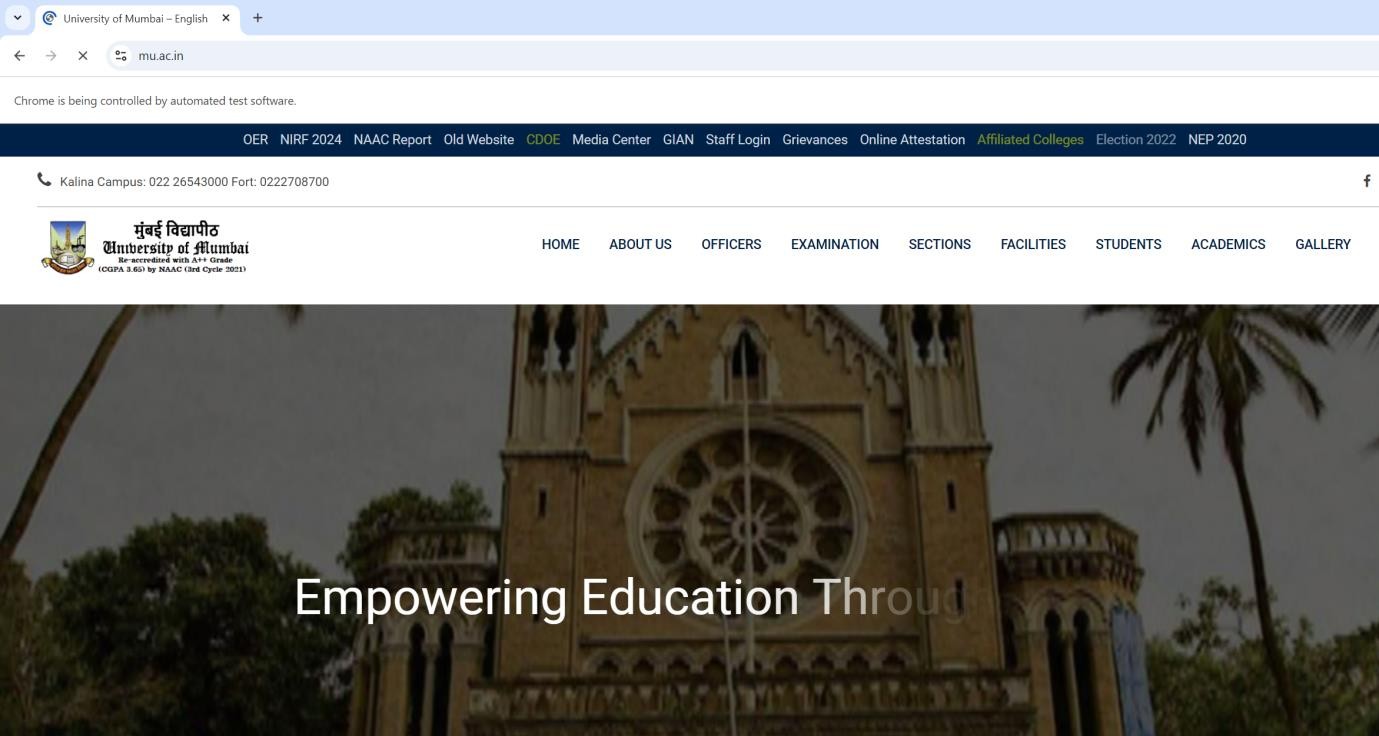
}

}

# OUTPUT:







**PRACTICAL 9**

# CODE:

**package** com.testselenium;

**import** org.openqa.selenium.WebDriver.Options; **import** org.openqa.selenium.WebDriver.Window; **import** org.openqa.selenium.chrome.ChromeDriver;

**public class** Testdemo8 {

**public static void** main(String args[]) **throws** InterruptedException { System.*setProperty*("webdriver.chrome.driver", "./drivers/chromedriver.exe"); ChromeDriver driver = **new** ChromeDriver();

Options manage = driver.manage(); Window window = manage.window(); window.maximize();

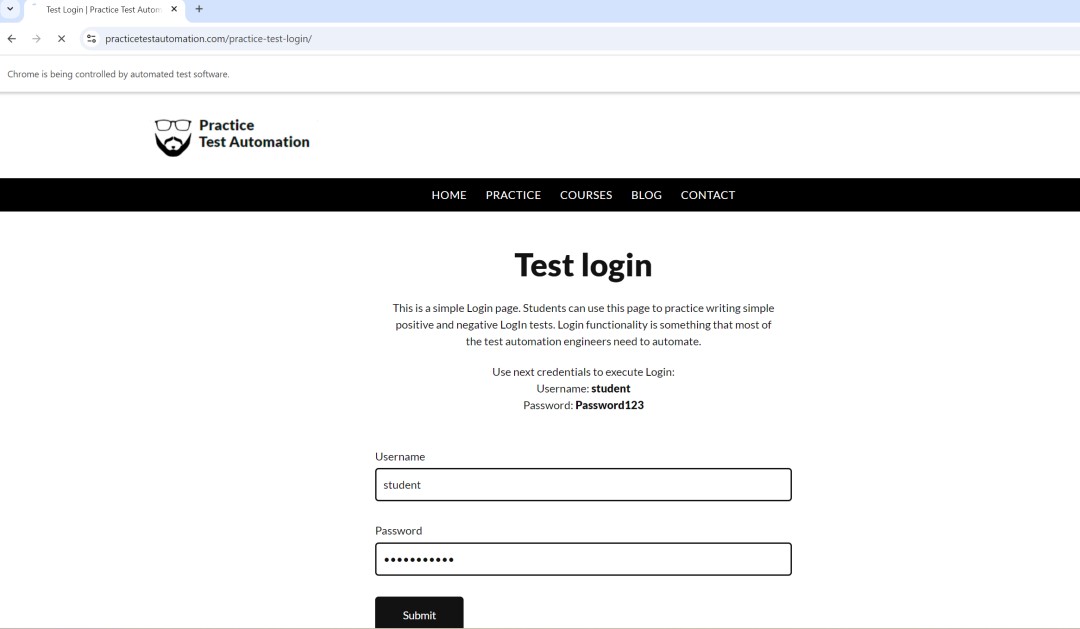
driver.get("https://practicetestautomation.com/practice-test-login/"); driver.findElementByName("username").sendKeys("student"); driver.findElementByName("password").sendKeys("Password123"); driver.findElementById("submit").click();

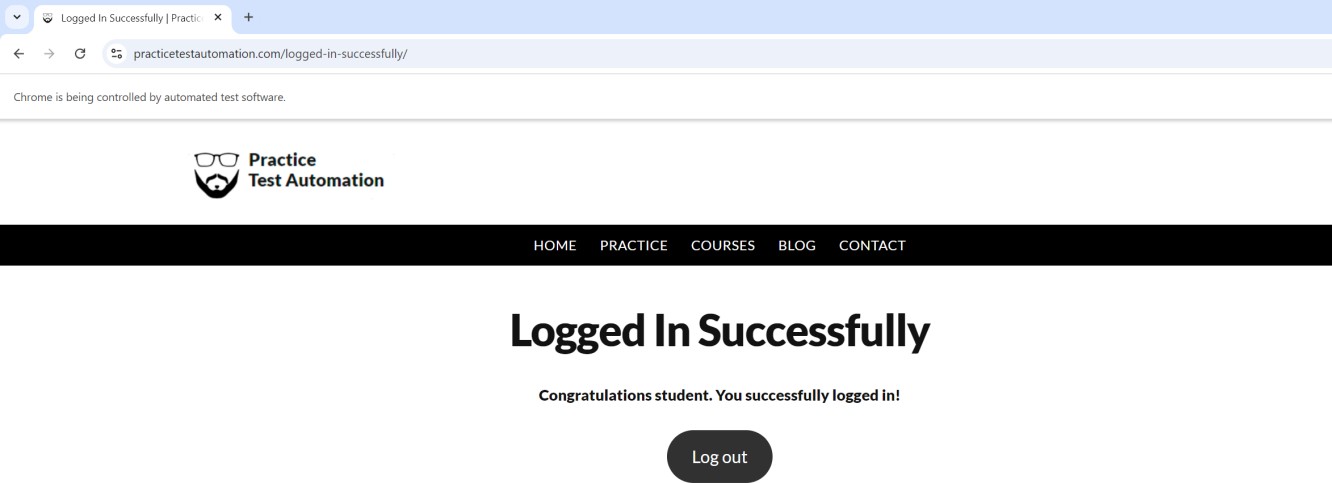
Thread.*sleep*(2000); driver.close();

}

}

# OUTPUT:





**PRACTICAL 10**

1. Aim: Functional Testing using Boundary Value Analysis

Scenario: A program takes the value of age from 21 to 65. Design test cases using boundary value analysis.

|  |  |  |
| --- | --- | --- |
| **BOUNDARY VALUE TEST CASE** | | |
| INVALID TEST CASE  (Min Value - 1) | VALID TEST CASE  (Min, +Min, Max, -Max) | INVALID TEST CASE  (Max Value + 1) |
| 20 | 21, 22, 65, 64 | 66 |

# Test Case Scenario

* 1. Input: - Enter the value of age as 20(21 - 1) Output: - Invalid
  2. Input: - Enter the value of age as 21 Output: - Valid
  3. Input: - Enter the value of age as 22(21 + 1) Output: - Valid
  4. Input: - Enter the value of age as 65 Output: - Valid
  5. Input: - Enter the value of age as 64(65 - 1) Output: - Valid
  6. Input: - Enter the value of age as 66(65 + 1) Output: - Invalid

1. Aim: Functional Testing using Equivalence Partitioning

Scenario: 3% rate of interest is given if the balance in the account is in the range of $0 to

$100, 5% rate of interest is given if the balance in the account is in the range of $100 to

$1000, and 7% rate of interest is given if the balance in the account is $1000 and above, we would initially identify three valid equivalence partitions and one invalid partition as shown below.

**Partition 1:** balance 0-100

Valid Inputs - 0-100- >=0 And <=100 Invalid Input-<0, $,#,@ A-Z **Partition 2**: balance 100-1000

Valid Input - 100-1000 (>100 And <=1000)

Invalid Input- $,#,@ A-Z

**Partition 3**: balance >1000

Valid Input- >1000 Invalid Input- $,#,@ A-Z

Invalid Valid partitions Valid Partition

<0 0-100 100-1000 >1000

3% Interest 5% Interest 7% Interest

# Test Case

**Test case id Test input Expected**

Account Balance Output B001 -90 Invalid input B002 50 3% Interest

B003 900 5% Interest

B004 2500 7% Interest B005 A Invalid input B006 $ Invalid input

1. Aim: Functional Testing using Decision Table

Consider test cases based on decision table for a ‗Login‘ Page Functionality. Business Rules:

1. On entering correct combination of ID & Password, user should be able to login successfully.
2. User is not allowed to login when any or both of the ID & Password are incorrect /blank. In such cases, it should show ‗Invalid

Credentials‘ message.

# We created the following combinations of Conditions, Actions and the respective rules in the decision table.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Rule 1 | Rule 2 | Rule 3 | Rule 4 | Rule 5 | Rule 6 | Rule 7 | Rule 8 | Rule 9 |
| Conditions | TC01 | TC02 | TC03 | TC04 | TC05 | TC06 | TC07 | TC08 | TC09 |
| UserID | Blank | Valid | Invalid | Blank | Blank | Invalid | Valid | Invalid | Valid |
| Password | Blank | Blank | Blank | Valid | Invalid | Valid | Invalid | Invalid | Valid |
| Actions |  | | | | | | | | |
| Login Successfully |  |  |  |  |  |  |  |  | Execute |
| Error Showing 'Invalid Credentials' | Execute | Execute | Execute | Execute | Execute | Execute | Execute | Execute |  |

In the above table, there are

1. conditions – UserID, Password
2. Actions – Login Successfully, Error showing ‗Invalid Credentials‘ and
3. Options — Blank, Valid, Invalid.

So, the total number of test cases are as follows:

# Options conditions i.e 32 = 9 Test cases

All test cases are not valid and significant some we need to optimise the test cases

Rules 1, 2, 3, 4, and 5 cover the same action Item ―Invalid Credentials‖ with options Blank and Invalid. Hence, we can consider any one of these test cases TC01 OR TC02 OR TC03 OR TC04 OR TC05

1. Rules 6,7, and 8 cover the same action Item ―Invalid Credentials‖ with options Valid and Invalid. Hence, we can consider any of these test cases TC06 OR TC07 OR TC08
2. Rule 9 covers the action item ―Login Successfully‖ with all valid options. Hence, we should consider the test case TC09.

# Condensed Decision Table as shown below:

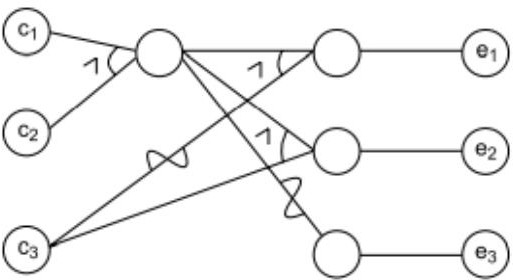
|  |  |  |  |
| --- | --- | --- | --- |
|  | Rule 2 Rule 8 Rule 9 | | |
| Conditions | TC02 TC08 TC09 | | |
| UserID | Valid | Invalid | Valid |
| Password | Blank | Invalid | Valid |
| Actions |  | | |
| Login Successfully |  |  | Execute |
| Error Showing 'Invalid Credentials' | Execute | Execute |  |

1. Aim: Functional Testing using Cause Effect Graph

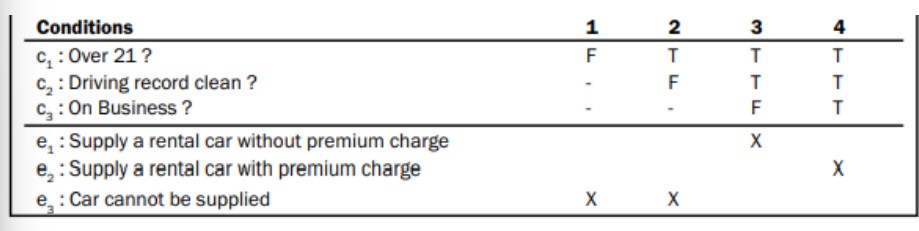
A tourist of age greater than 21 years and having a clean driving record is supplied a rental car. A premium amount is also charged if the tourist is on business, otherwise it is not charged. If the tourist is less than 21 year old, or does not have a clean driving record, the system will display the following message: ―Car cannot be supplied‖ Draw the cause-effect graph and generate test cases

Solution: The causes are c1 : Age is over 21 c2 : Driving record is clean c3 : Tourist is on business and effects are e1 : Supply a rental car without premium charge. e2 : Supply a rental car with premium charge e3 : Car cannot be supplied .The Cause Effect Graph and test cases based on that are as follows :

Cause Effect Graph:



Decision Table on Rental Car Problem:



Test Cases:

