



Bharati Vidyapeeth's Institute of Management & Information Technology

C.B.D. Belapur, Navi Mumbai 400614

Vision:

Providing high quality, innovative and value-based education in information technology to build competent professionals.

Mission

M1. Technical Skills: -To provide solid technical foundation theoretically as well as practically capable of providing quality services to industry.

M2. Development: -Department caters to the needs of students through comprehensive educational programs and promotes lifelong learning in the field of computer Applications.

M3. Ethical leadership: -Department develops ethical leadership insight in the students to succeed in industry, government and academia.

CERTIFICATE

This is to certify that the journal is the work of
Mr. Sayas S. Sonawane RollNo. **51** of MCA (Sem- **III** Div: **B**) for the academic year 2022 - 2023.
Subject Code: **MCAL35**

Subject Name: **Software Testing & Quality Assurance Lab**

Subject-in-charge

Principal

Date: _____

External Examiner

Bharati Vidyapeeth's Institute of Management & Information Technology

MCA Semester III AY 2021-22

MCAL35: Software Testing & Quality Assurance Lab

INDEX

Name: Sayas Sonawane

Div: B

Roll No.: 51

Sr. No.	Practical Name	Date	Sign
1.	Take a review and write test cases for any known application (Amazon/flipkart)		
2.	Implement Web Drivers on Chrome & Firefox Browsers.		
3.	Demonstrate handling multiple frames in selenium		
4.	Implement Browser command and navigation Commands.		
5.	Implement the find element command		
6.	Demonstrate the Locator(id,css selector, path)		
7.	Demonstrate synchronization in selenium		
8.	Demonstrate different types of alerts		
9.	Demonstrate : Handling Drop Down, List Boxes		
10.	Demonstrate : Command Button,Radio buttons & text boxes.Waits command in selenium		
11.	Demonstrate action classes in Selenium		
12.	Installation of TestNg , running testNg and TestNg annotations		
13.	Demonstrate data driven Framework.		

Practical No.: 01**Aim: Write test cases for any known application Amazon.**

Test Case No	Test Scenarios	Test Steps	Test Data	Expected Results	Actual Result	Test Status
tc_01	Check weather the site is loading	Load the site.	Site="https://www.amazon.in/"	Site should be loaded.	Site is Loaded.	Pass
tc_02	To search for a particular Item in the Search bar	Enter item name in the search bar.	Item name- "Camera stand"	Related items Should displayed on the screen	Related items are displayed on the screen	Pass
tc_03	To check weather user is add able to same items to	Add item in the cart.	item name = "Item name" and site = "https://www.amazon.in/gp/cart/view.html?ref_=nav_cart"	Items should be added in the cart and user	Items is be added in the cart and user continue d navigatin g another items	Pass
tc_04	To check whether item count in the cart is incrementing and their totals are displayed in the cart	Navigate to the same item Click on Add to Cart button	site = "https://www.amazon.in/gp/cart/view.html?ref_=nav_cart" Item - "Camera stand" Qty = 4	After clicking on Add to cart button of same product the cart count should increment by 1 and price should get added to existing cart price	cart count is increment by 1 and price get added to existing cart price	Pass
				Product should show inclusive of all taxes	Show's inclusive of all taxes	

			item = "camera stand"	Related items should be removed from the cart and item count	item removes from the cart and item count	
--	--	--	-----------------------	--	---	--

				should decrement by 1	count decremented by 1	
tc_05	Verify weather tax as per location should applied	Click on any item in the cart and try to change the delivery location	-	Tax should be displayed with amount.	Amount is displayed	Pass
tc_06	To remove one of the item from the cart	Open cart and Remove one item from the cart.	Item – “Camera stand” Qty = 2	Removed item should delete from cart	Removed item is deleted from cart	Pass
tc_07	to check whether we can proceed to buy	Open cart and click on Proceed to Buy button	Link of cart = "https://www.amazon.in/gp/cart/view.html/ref=dp_atch_dss_cart?"	The site should navigate user to sign-in	after clicking on Proceed to Buy button sign in option appears	
tc_08	check different payment options don't checkout, close the site & come back later. the site should retain the items in the cart	close the site and reopen it	site = "https://www.amazon.in/"	The cart should retain the number of items	Cart symbol on the top right corner shows same number of items as previous session	

tc_09	check if allowing checkout as guest, simply finish the purchase and provide an option to register.	Click on checkout button		The cart should retain the number of items	Cart symbol on the top right corner shows same number of items as previous session	
tc_10	To check whether sign in with mobile no. or email input field appears			should allow to checkout as guest On the sign in page sign in with mobile no. or email input field should appear	Didn't allow to checkout as guest and redirecete d to sign in.	
	check whether input field is able to validate email id	Enter email id in the input field	email = "abc@gmail.com"	Email should get validated	Email gets validated	
		Enter email id in the input field without @ symbol	email = "abgmail.com"	Email should not get validated and should show error message	Email doesn't get validated and shows message "Invalid mail id"	
		Enter email id in the input field without ''	phone = 9632587415	Phone number should get validate	Phone number validated	
	check whether input field is able to validate mobile number	Enter mobile number exactly 10 digit long in input field	phone = 9632587415	Phone number should get validate	Phone number validated	

		Enter mobile number less than 10 digit in input field	phone = 9632587415	Phone number should not get validate	Phone number didn't validate	
		Enter mobile number greater than 10 digit in input field	phone = 78945623698	Phone number should not get validate	Phone number didn't validate	
	check whether continue button is working	Click on continue button				
tc_11	check signup	On homepage ,from	site = "https://www.a	Sign up page should open	Sign up page	

		accounts and lists dropdown select start here	mazon.i n/"		opens	
	Check your name field	Enter your name	name = "abc"	field should not show any error	Didn't show error	
			name="abc 123"	field should not accept numbers as input	Field accepts numbers also as name	
	check whether input field is able to validate email id	Enter email id in there input field	email = "abc077@gmail.com"	Email should get validated	Email gets validated	
		Enter email id in the input field without @ symbol	email = "abcmail.com"	Email should not get validated and should show error message	Email doesn't get validated and shows message "Invalid mail id"	
	check whether input field is able to validate mobile number	Enter mobile number in input field	phone = 9632587415	Phone number should get validate	Phone number validated	

		Enter mobile number in input field 9 numbers long	phone = 96325875	Phone number should not get validate	Phone number didn't validate	
	Enter password with 6 characters	Enter password with 6 characters	password=1234 56	field should accept the password	field accepts the password	
		Enter password greater than 6 characters	password=1234 56789	field should accept the password	field accepts the password	
		Enter password less than 6 characters	password=1234	field should not accept the password	field didnt accepts the password	
		leave password field blank	password = " "	field should show error as required field	field gives error as required field	
	check password field Password again	Enter exact password as entered in Password field		field should accept the password	field accepts the password	
		Enter password different than Password field		Enter password different than Password field	field didn't accept the password	

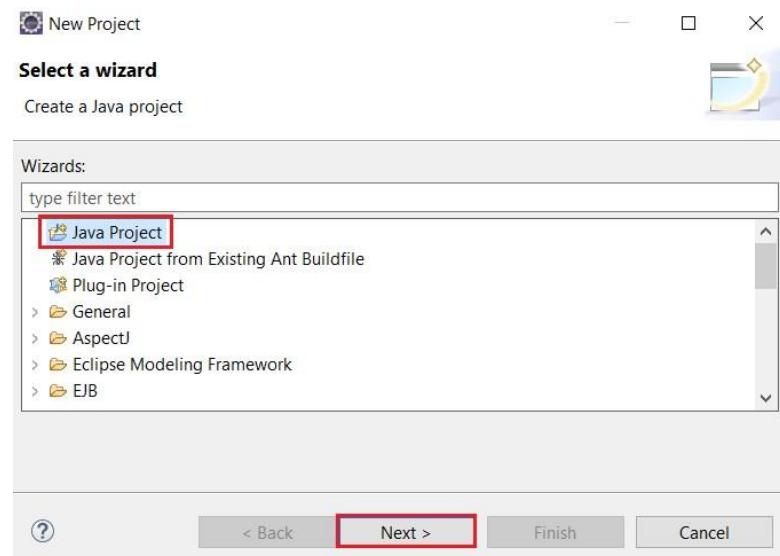
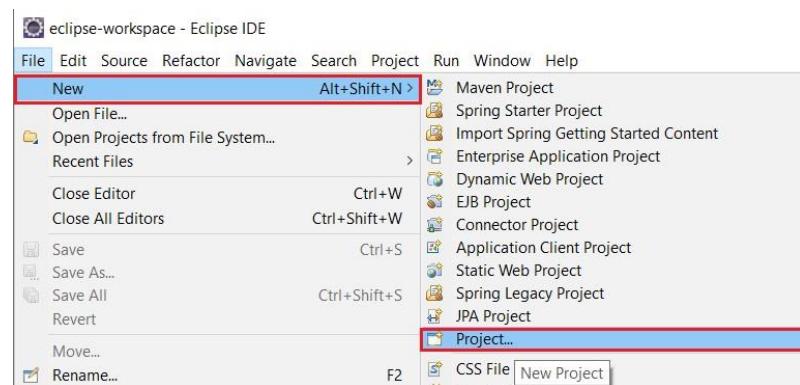
Practical No.: 02

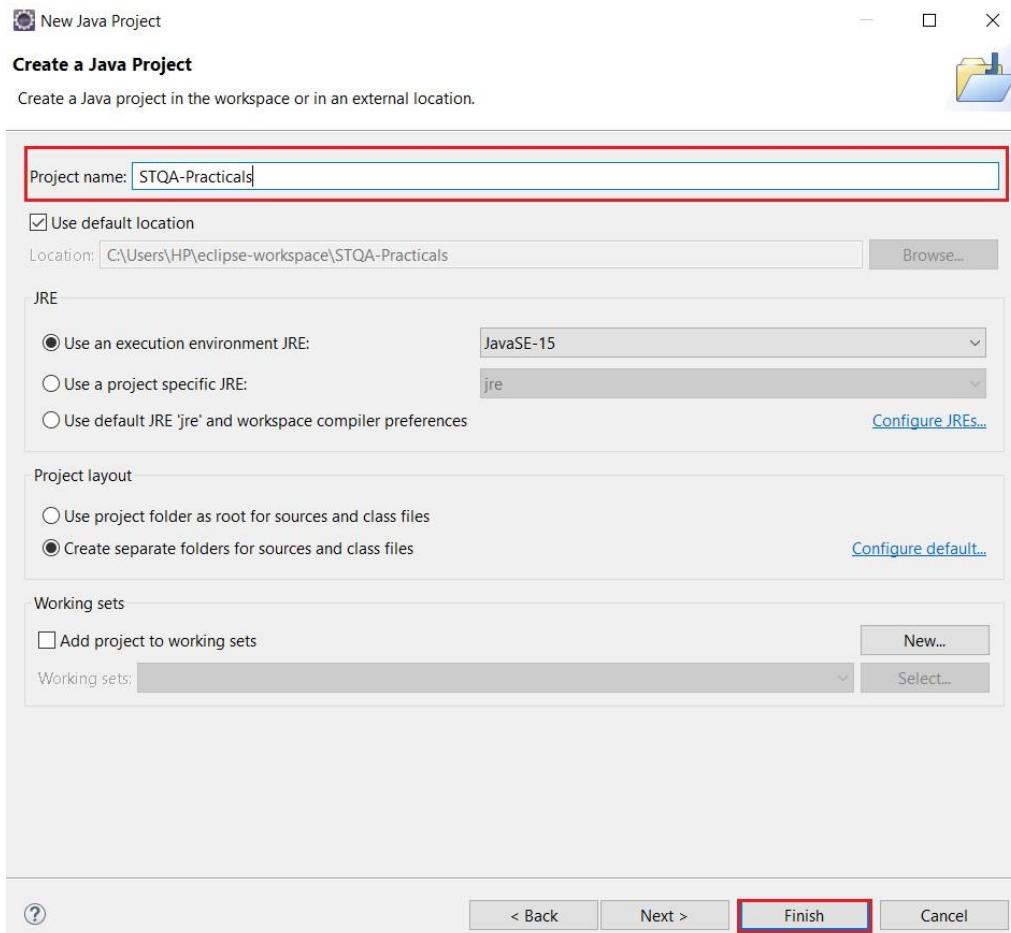
Aim: Implement Web Drivers on Chrome & Firefox Browsers.

You need to download the following drivers to work with different browsers.

1. Firefox- Mozilla GeckoDriver
2. Selenium- Webdriver jar's
3. Chrome- ChromeDriver

1. Open Eclipse > Create project > Give name
2. Create Package > Create Class (with main method)
3. Add Selenium jar files

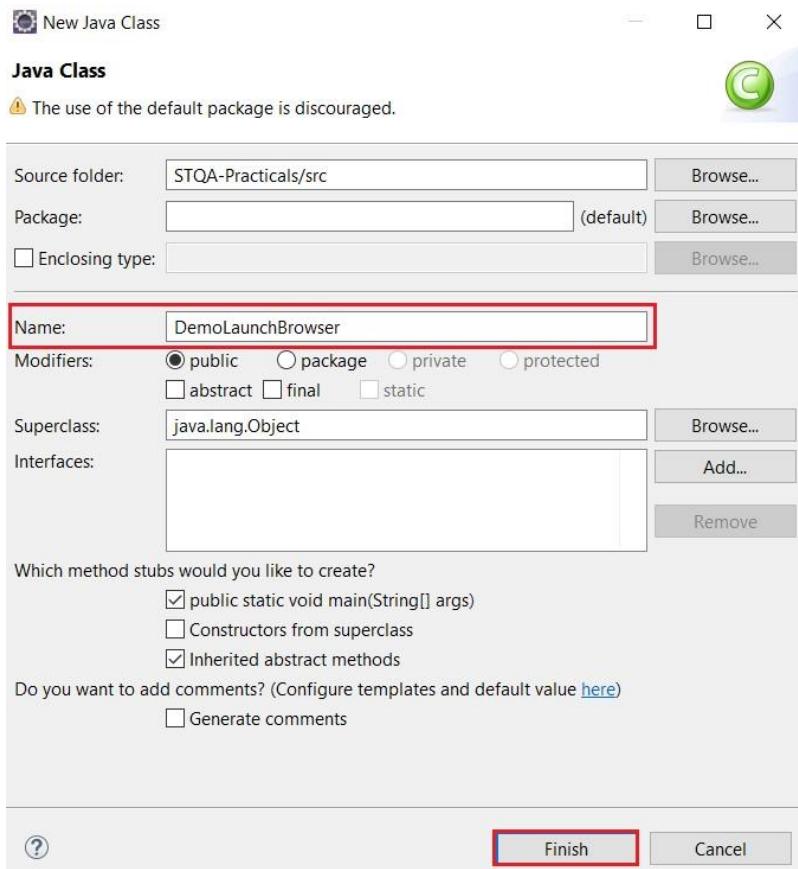




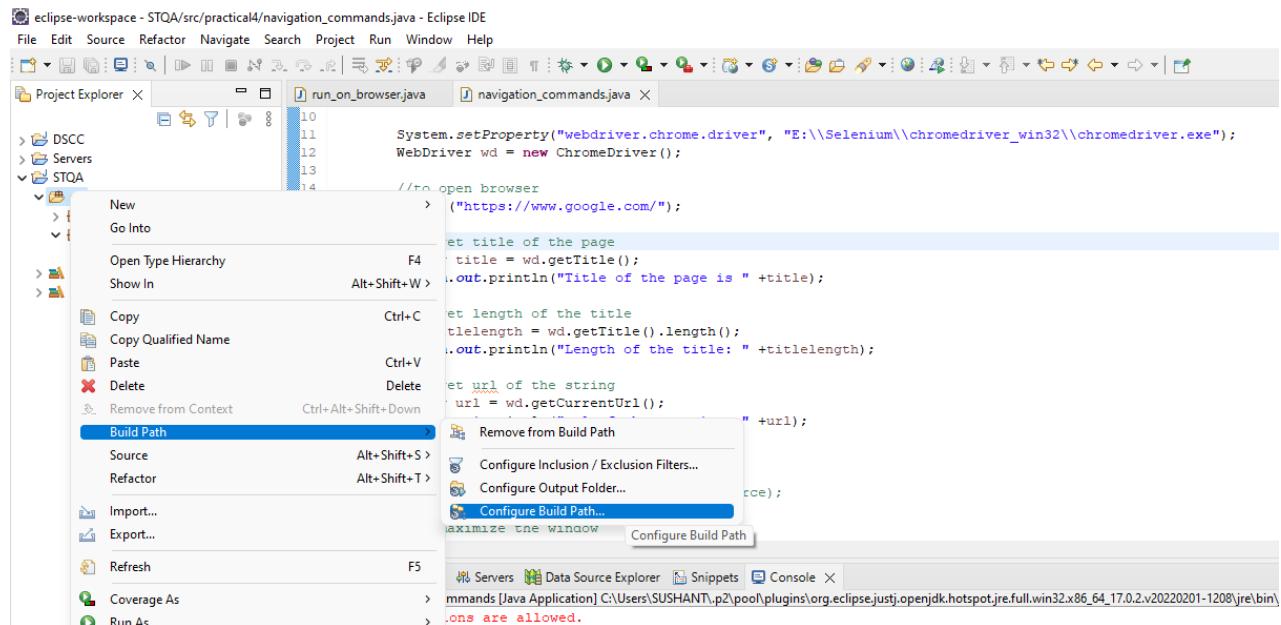
Step1. Right click on the "src" folder and create a new Class File from New > Class.

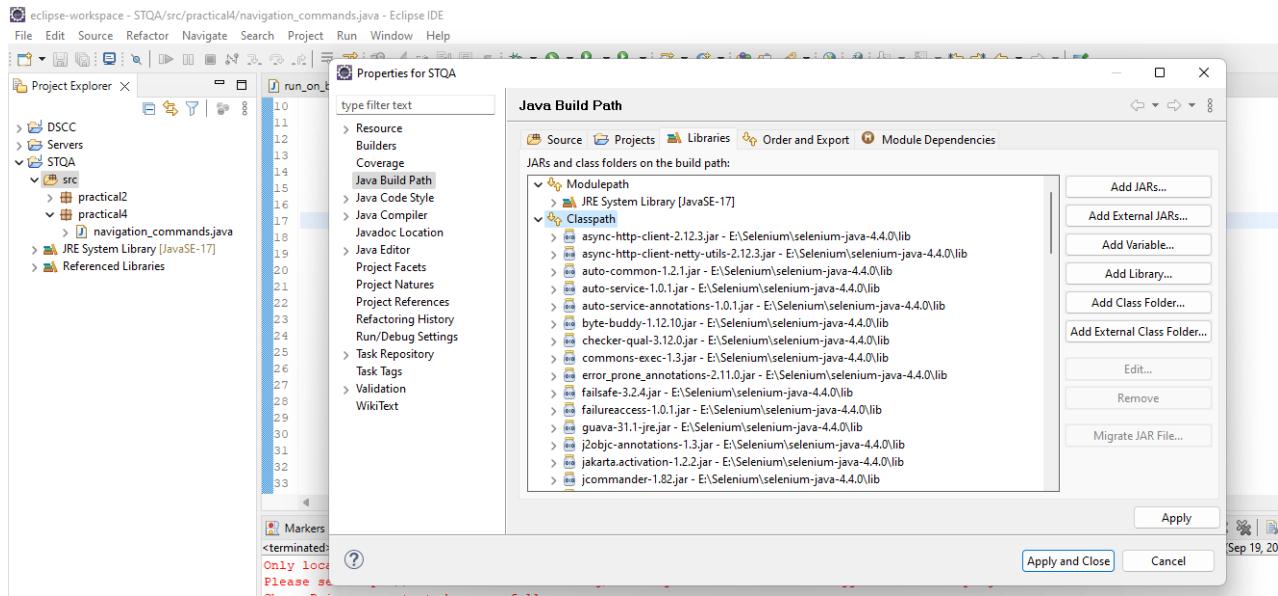


Give your Class name as "DemoLaunchBrowser" and click on "Finish" button.



Step2: Add jar files



**Program:****Run_on_browser.java**

```

package practical2;

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

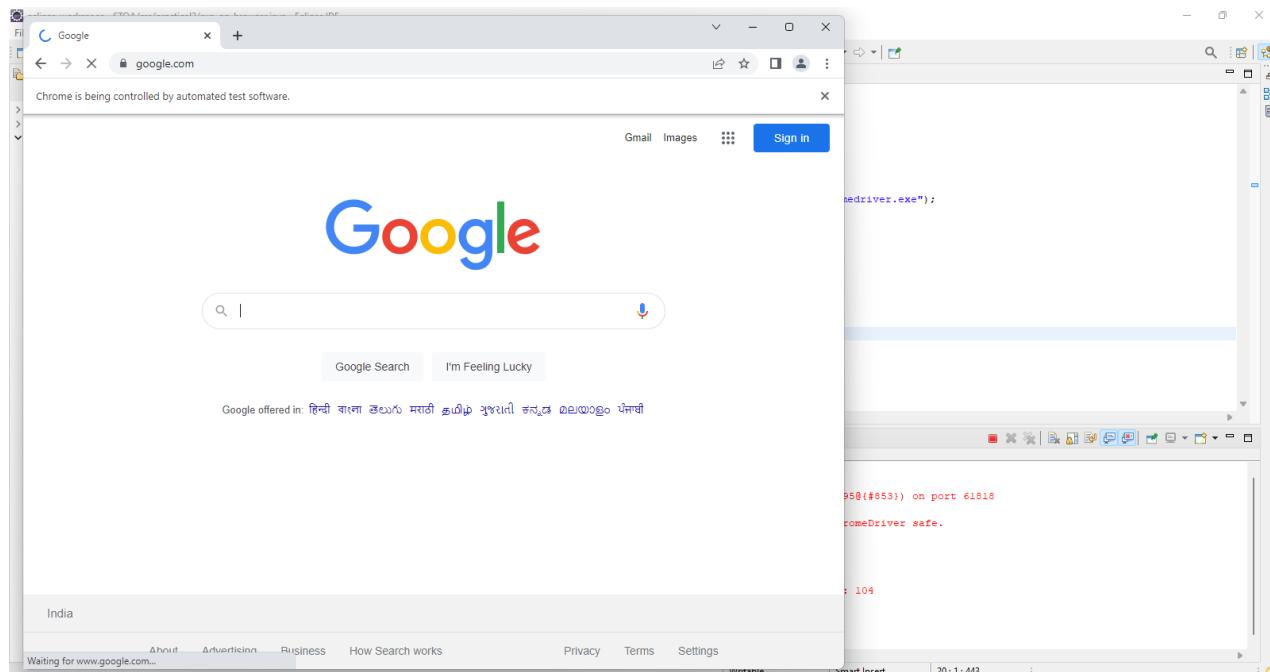
public class run_on_browser {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();
        wd.get("https://www.google.com/");
        //wd.close();

    }
}

```

Output:



Practical No.: 03

Aim: Demonstrate handling multiple frames in selenium

Program:

```
package practical2;
```

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.*;
import java.util.concurrent.TimeUnit;
```

```
public class Practical_3 {
    public static void main(String[] args)

    {

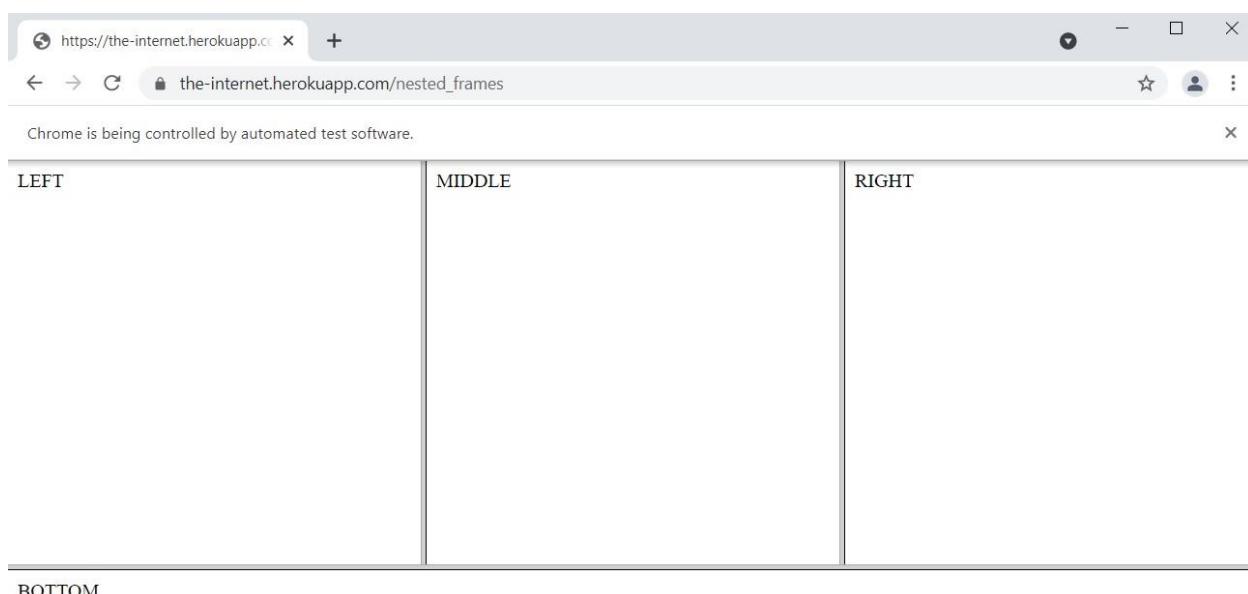
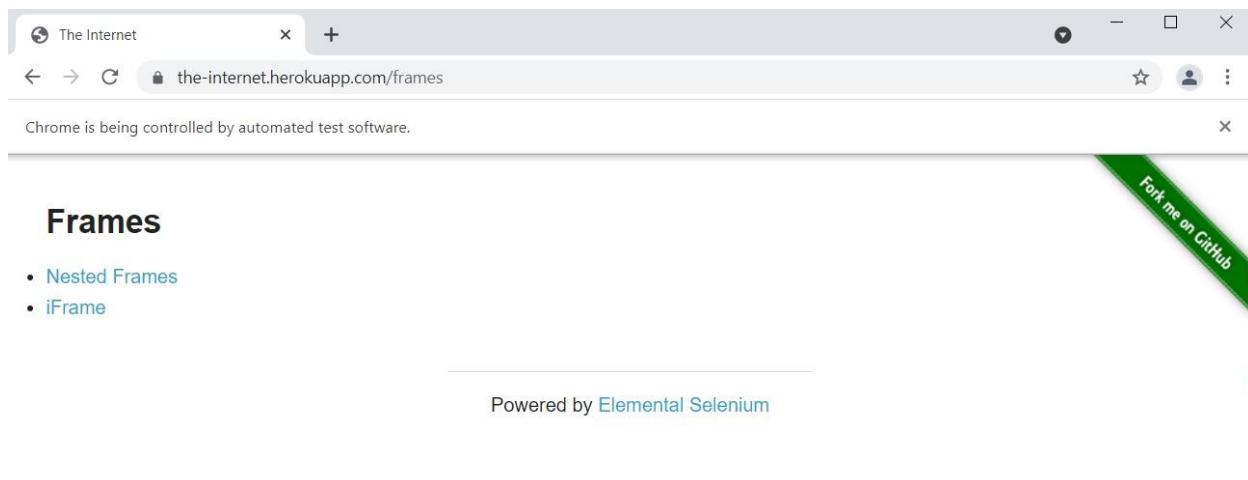
        System.setProperty("webdriver.chrome.driver",
"E:\Selenium\chromedriver_win32\chromedriver.exe");
        WebDriver driver = new ChromeDriver();

        String url = "https://the-internet.herokuapp.com/frames";
        driver.get(url);
        driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);
        // identify element
        driver.findElement(By.linkText("Nested Frames")).click();           // switch to frame with frame
name and identify inside element
        driver.switchTo().frame("frame-bottom");
        WebElement l = driver.findElement(By.cssSelector("body"));
        System.out.println("Bottom frame text: " +l.getText());
        // switch to main page
        driver.switchTo().defaultContent();
        // driver.quit();

    }

}
```

Output:



```
Markers Properties Servers Data Source Explorer Snippets Coverage Console Call Hierarchy
<terminated> MultipleFrameHandling [Java Application] C:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.
Starting ChromeDriver 96.0.4664.45 (76e4c1bb2ab4671b8beba3444e61c0f17584b2fc-refs/branch-heads/4664@{#947}) on port 10520
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Dec 02, 2021 10:35:16 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Bottom frame text: BOTTOM
```

Practical No.: 04

Aim: Implement Browser command and navigation Commands.

1. Open Eclipse > Create project > Give name
2. Create Package > Create Class (with main method)
3. Add Selenium jar files

Program:

```
package practical4;

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

public class navigation_commands {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();

        //to open browser
        wd.get("https://www.google.com/");

        //to get title of the page
        String title = wd.getTitle();
        System.out.println("Title of the page is " +title);

        //to get length of the title
        int titlelength = wd.getTitle().length();
        System.out.println("Length of the title: " +titlelength);

        //to get url of the string
        String url = wd.getCurrentUrl();
        System.out.println("url of the page is : " +url);

        //to get source code of the page
        //String source = wd.getPageSource();
        //System.out.println("Source code: " +source);

        //to maximize the window
        wd.manage().window().maximize();

        //navigate directly to another url
        wd.navigate().to("https://www.youtube.com/");

        //navigate back
        wd.navigate().back();

        //navigate forward
        wd.navigate().forward();
```

```

//refresh page
wd.navigate().refresh();

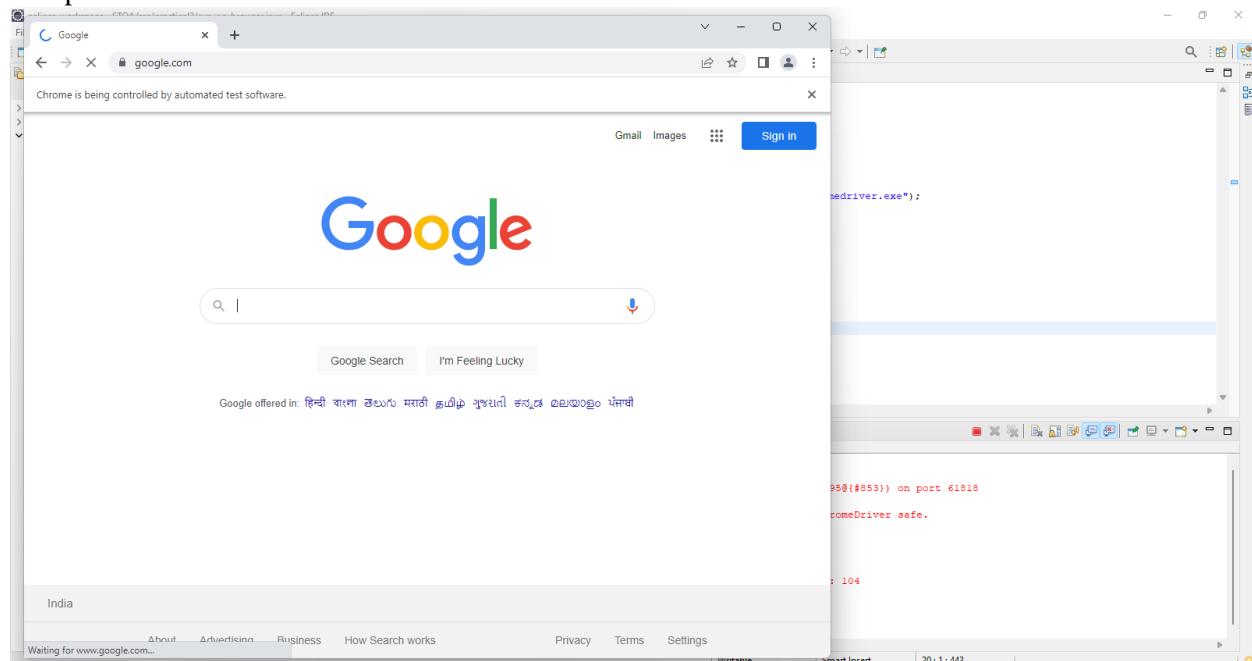
//to close the browser
wd.close();
}

}

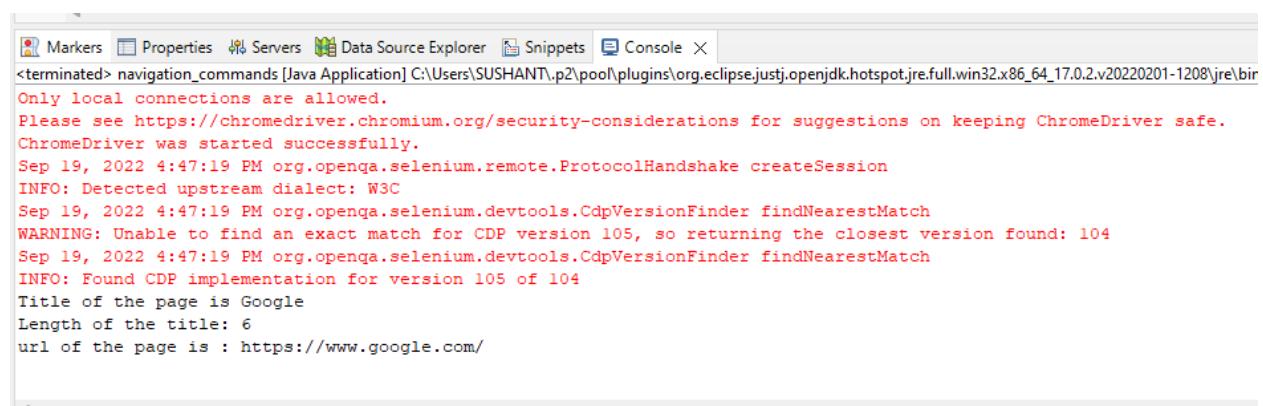
```

Output:

To open browser

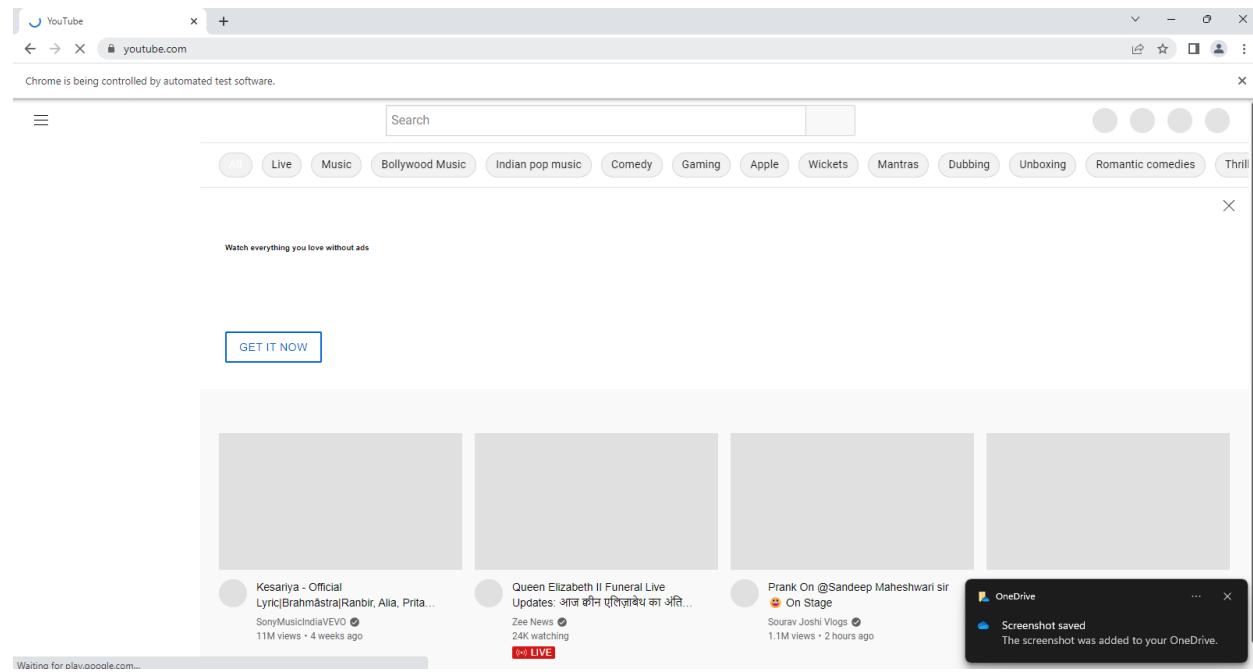


Title Length and URL of the page:

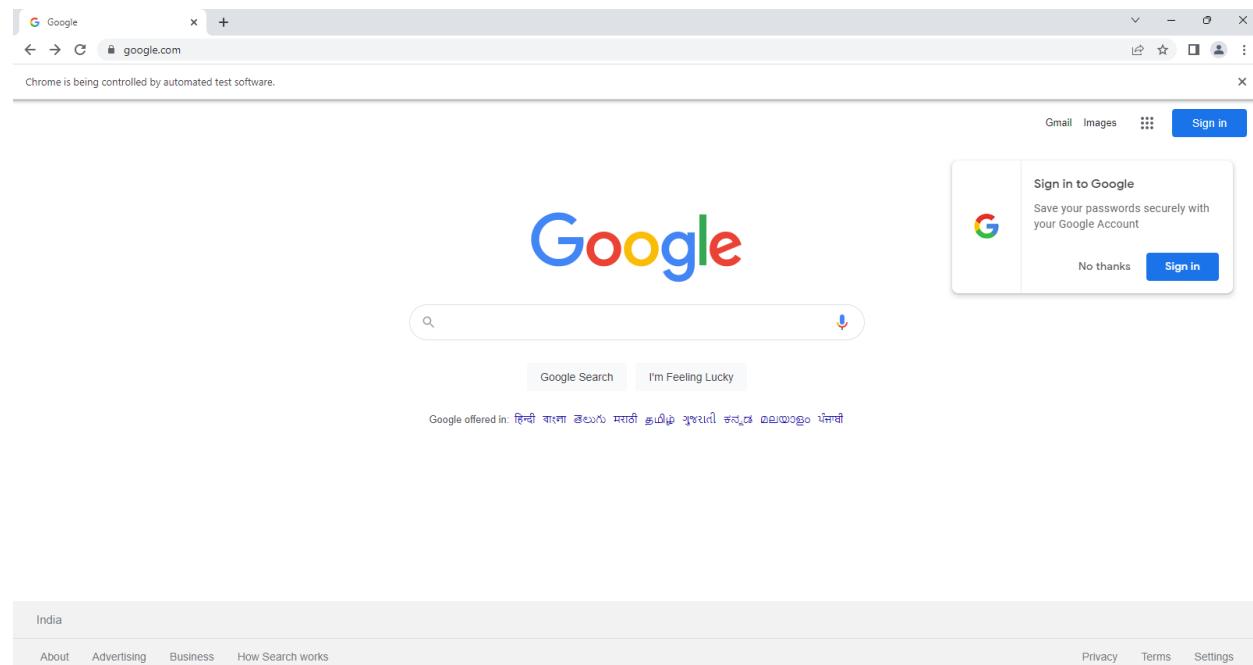


Maximize the window:

Navigate to other url:



Navigate back:



Source code:



The screenshot shows a Java console window titled "Console X" with the path "terminated> GoogleDrive [Java Application] C:\Users\admin\Desktop\kajal\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (Sep 12, 2022, 3:10:07 PM)". The window contains a large amount of obfuscated JavaScript code, which appears to be a closure library implementation. The code includes various functions like ne, oe, catch, try, and catch, along with comments about the Closure Library Authors and Apache 2.0 license. It also references objects like A, B, C, D, E, F, G, H, J, K, L, M, and N, and properties like passive, click, and various event listeners.

Close Browser using wd.close();

Browser automatically closed.

Practical No.: 05

Aim: Implement the find element command

Program:

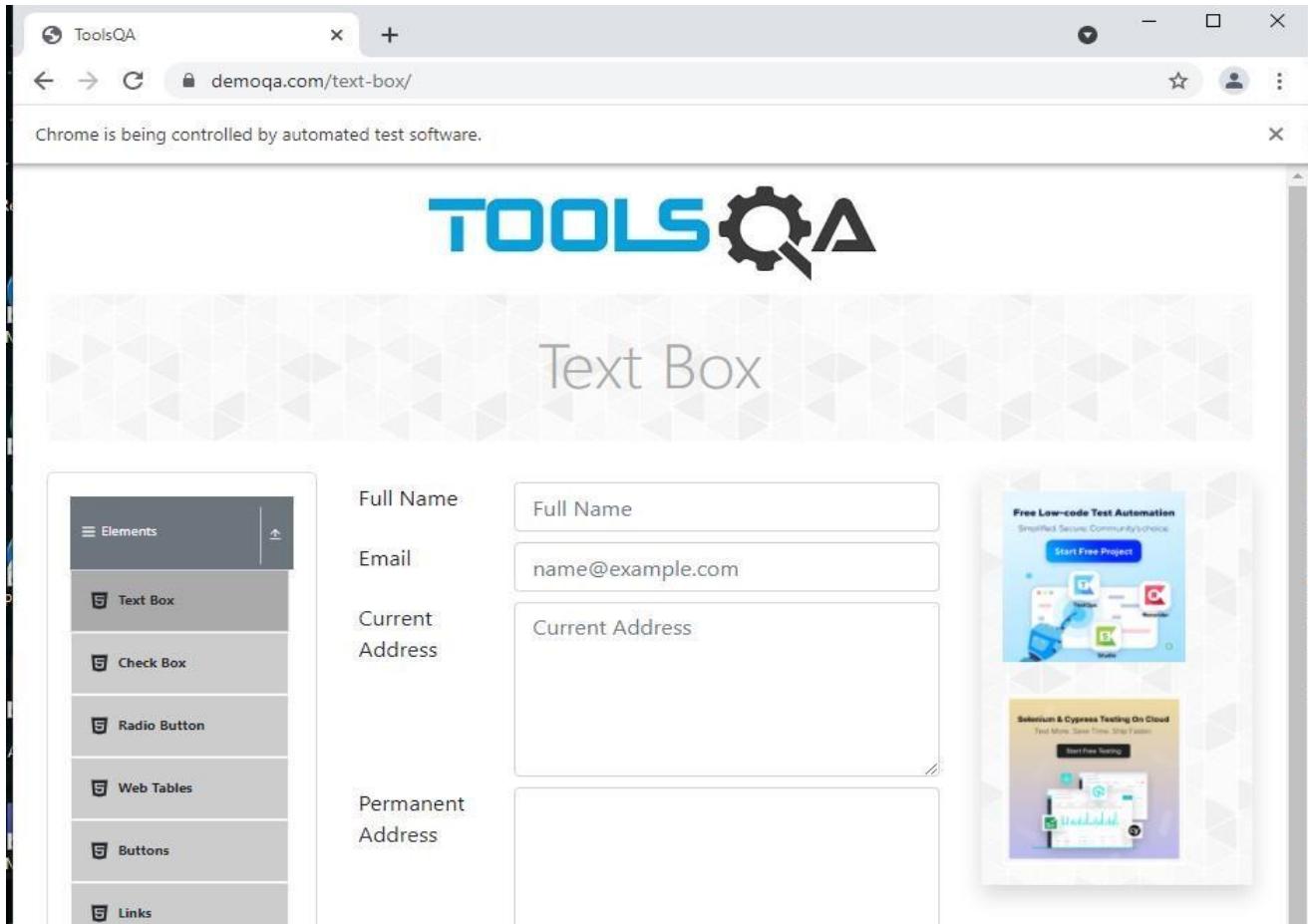
```
package stqa;
import java.util.List;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class FindElement {

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "C:\\\\Eclipse\\\\Drivers\\\\chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://demoqa.com/text-box/");           // Find elements
        using tag name
        List<WebElement> allInputElements =
        driver.findElements(By.tagName("input"));

        if(allInputElements.size() != 0)
        {
            System.out.println(allInputElements.size() + " Elements found by TagName as input
\\n");
            for(WebElement inputElement : allInputElements)
            {
                System.out.println(inputElement.getAttribute("placeholder"));
            }
        }
    }
}
```

Output:

```
<terminated>.FindElement [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver was started successfully.
Dec 07, 2021 7:45:38 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
2 Elements found by TagName as input

Full Name
name@example.com
```

Practical No.: 06

Aim: demonstrate the Locator (ID, CSS selector, Path)

Program:

Locator:-

```
package practical6;

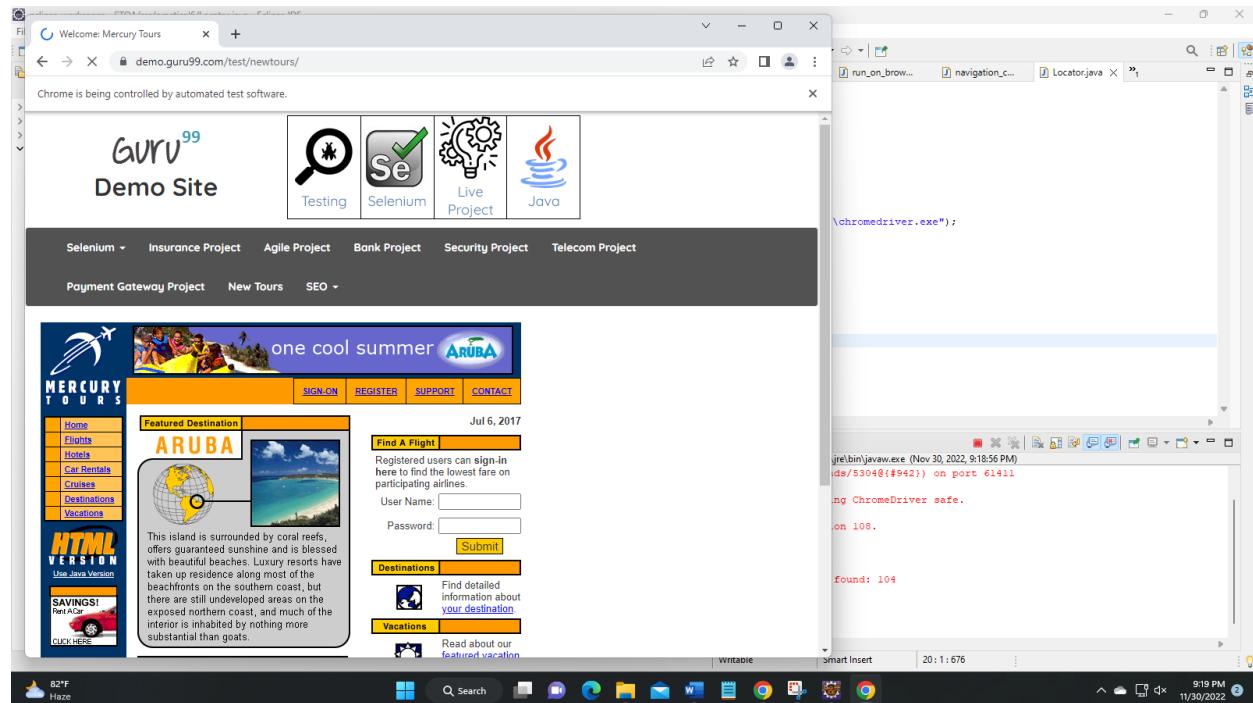
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.By;

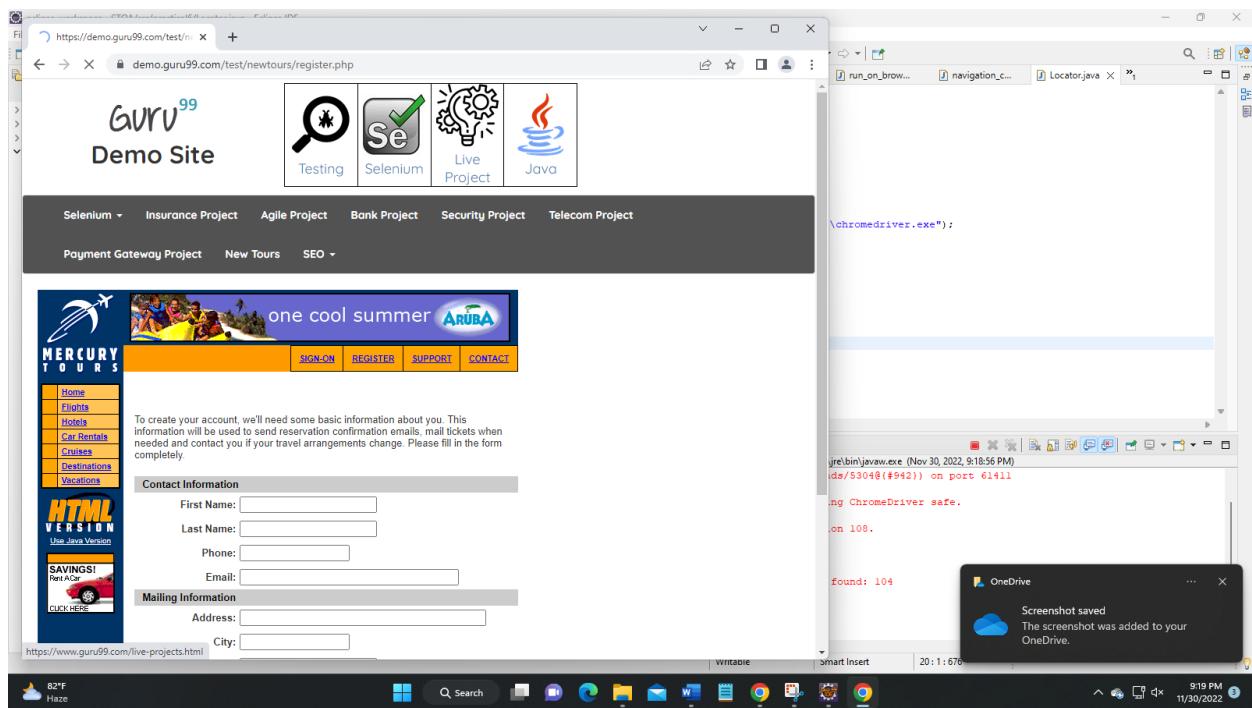
public class Locator {
    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();

        wd.get("https://demo.guru99.com/test/newtours/");
        wd.findElement(By.name("userName")).sendKeys("Admin");//locator name
        wd.findElement(By.name("password")).sendKeys("admin123");
        wd.findElement(By.name("submit")).click();
        wd.findElement(By.linkText("REGISTER")).click();//locator by linktext
    }
}
```

Output:





Practical No: 07

Aim: Demonstrate synchronization in selenium

Program:

Implicit wait

```
package practical9;  
import java.time.Duration;  
import org.openqa.selenium.By;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.chrome.ChromeDriver;  
  
public class Synchronization {  
    //Implicit wait  
    public static void main(String[] args) {  
        System.setProperty("webdriver.chrome.driver",  
        "E:\\Selenium\\chromedriver_win32\\chromedriver.exe");  
        WebDriver wd = new ChromeDriver();  
        wd.get("https://demo.guru99.com/test/newtours/");  
        wd.manage().timeouts().implicitlyWait(Duration.ofSeconds(60));  
        wd.findElement(By.name("userName")).sendKeys("admin");//locator id  
        wd.findElement(By.name("password")).sendKeys("admin");//locator name  
        wd.manage().timeouts().implicitlyWait(Duration.ofSeconds(60));  
        wd.findElement(By.name("submit")).click(); //locator className  
        //wd.close();  
    }  
}
```

Output:

The screenshot illustrates a Selenium test setup. It features two separate browser windows side-by-side. The top browser window shows the Guru99 Demo Site, specifically a flight search page for Aruba. The bottom browser window shows a successful login message. To the right of the browsers, there are two terminal windows. The top terminal window shows Java command-line output related to chromedriver.exe. The bottom terminal window shows a OneDrive notification about a screenshot being saved.

ExplicitWait:

```
package practical9;

import java.time.Duration;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.openqa.selenium.support.ui.ExpectedCondition;

import org.openqa.selenium.support.ui.ExpectedConditions;

public class ExplicitWait {

//ExplicitWait

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");

        WebDriver wd = new ChromeDriver();

        wd.get("https://demo.guru99.com/test/newtours/");

        WebDriverWait wt = new WebDriverWait(wd, Duration.ofSeconds(20));

        wd.findElement(By.name("userName")).sendKeys("admin");//locator id

        wd.findElement(By.name("password")).sendKeys("admin");//locator name

        wd.findElement(By.name("submit")).click(); //locator className

        wd.findElement(By.partialLinkText("REGISTER")).click(); //locator partial link

text

        wt.until(ExpectedConditions.visibilityOfElementLocated(By.linkText("CONTACT")));

        wd.findElement(By.linkText("CONTACT")).click();//locator link text

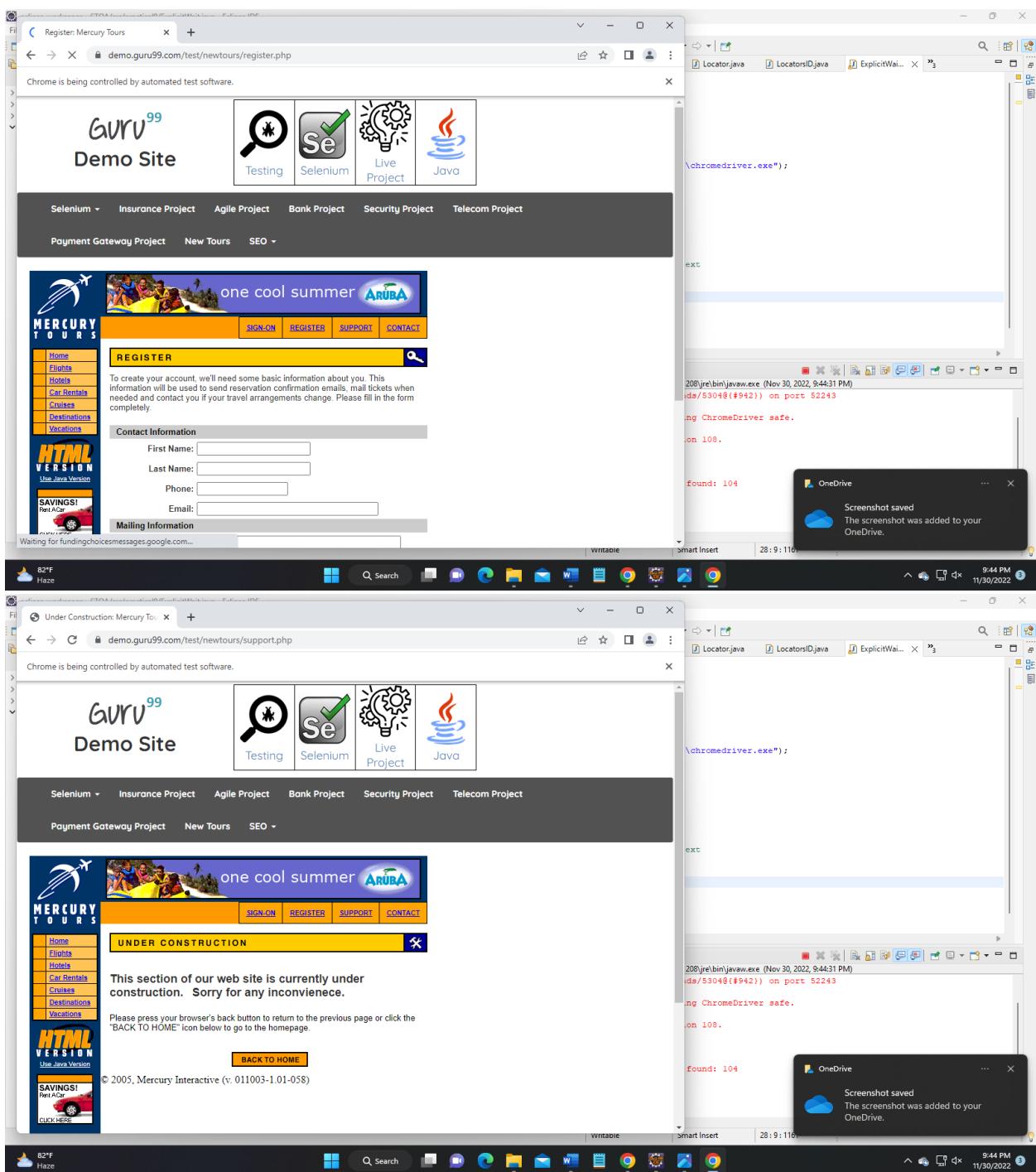
    }

}
```

Output:

The screenshot shows a desktop environment with two browser windows open. Both windows are controlled by a Java application, as indicated by the status bar message "Chrome is being controlled by automated test software".

- Top Window:** Displays the Guru99 Demo Site. The URL is `demo.guru99.com/test/newtours/`. The page shows a travel booking form for Aruba, featuring a globe icon, a beach image, and a "Find A Flight" button. The date "Jul 6, 2017" is displayed above the flight search field.
- Bottom Window:** Displays a successful login message: "Login Successfully". The URL is `demo.guru99.com/test/newtours/login_success.php`. The message says "Thank you for Loggin." and includes the copyright notice "© 2005, Mercury Interactive (v. 011003-1.01-05\$)".
- Background:** Shows the Java application's interface with code snippets and a terminal window displaying Java command-line output.



Practical No: 8

Aim: Demonstrate different types of alerts

Program:

```
package practical10;

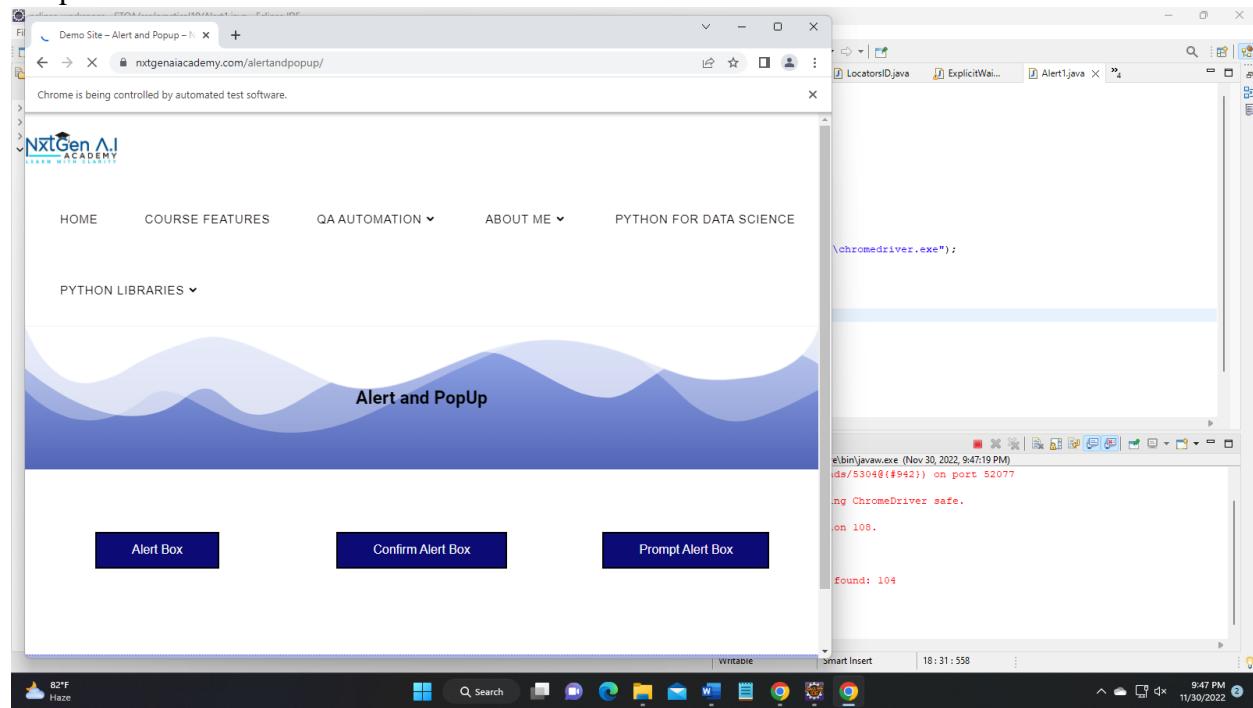
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;

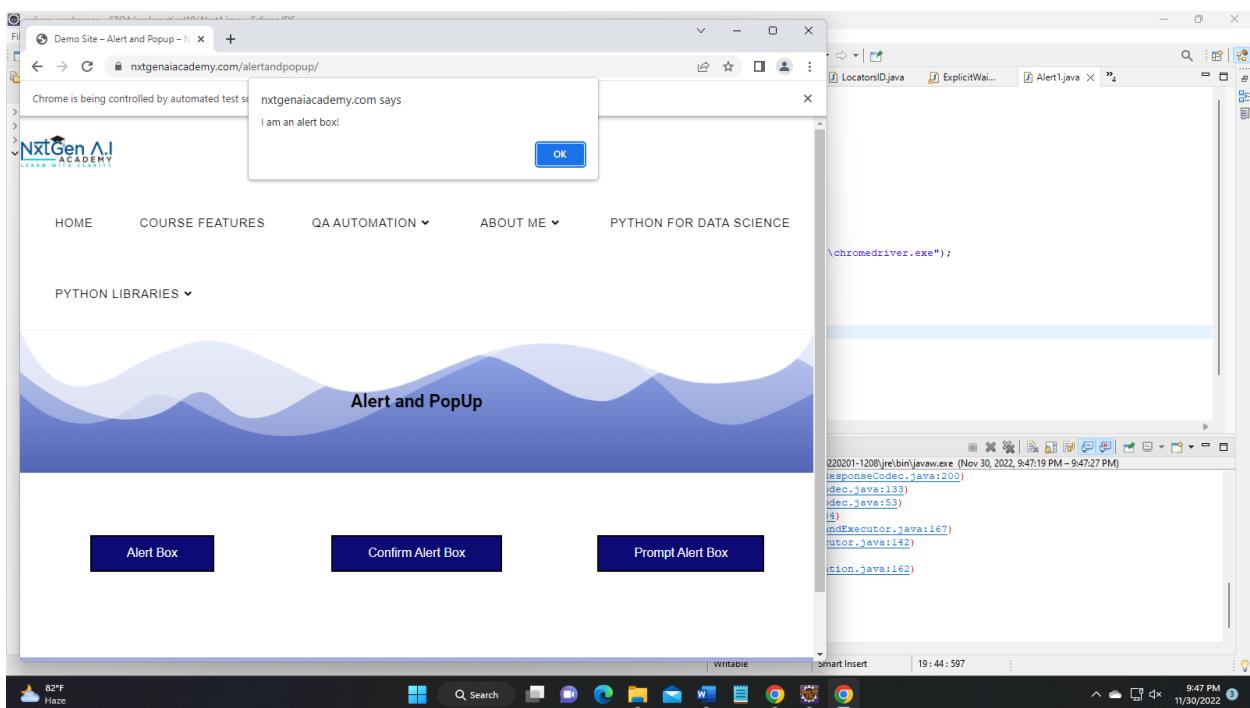
public class Alert1 {

    public static void main(String[] args) throws InterruptedException {

        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();
        wd.get("https://nxtgenaiacademy.com/alertandpopup/");
        wd.findElement(By.name("alertbox")).click();
        //Thread.sleep(2000);
        Alert alt = wd.switchTo().alert();
        System.out.println("Text of Alert : " + alt.getText());//capture text
        alt.accept();//click on OK button
        //alt.dismiss(); click on cancel button
    }
}
```

Output:





```
Markers Properties Servers Data Source Explorer Snippets Console <terminated> Alert1 [Java Application] C:\Users\SUSHANT\.p2\pool\plugins\org.eclipse.jdt.core\org.eclipse.jdt.core_3.20.0.v20200201-1208\jre\bin\javaw.exe (Nov 30, 2022, 9:53:35 PM) Starting ChromeDriver 107.0.5304.62 (1eec40d3a5764881c92085aaee66d25075c159aa-refs/branch-heads/5304@(#942)) on port 55834 Only local connections are allowed. Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe. ChromeDriver was started successfully. [1669825415.192] [WARNING]: This version of ChromeDriver has not been tested with Chrome version 108. Nov 30, 2022 9:53:35 PM org.openqa.selenium.remote.ProtocolHandshake createSession INFO: Detected upstream dialect: W3C Nov 30, 2022 9:53:35 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch WARNING: Unable to find an exact match for CDP version 108, so returning the closest version found: 104 Nov 30, 2022 9:53:35 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch INFO: Found CDP implementation for version 108 of 104 Text of Alert : I am an alert box!
```

Practical No: 09

Aim: Demonstrate Handling Drop Down, List Boxes

Dropdown Program:

```
package practical11;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;

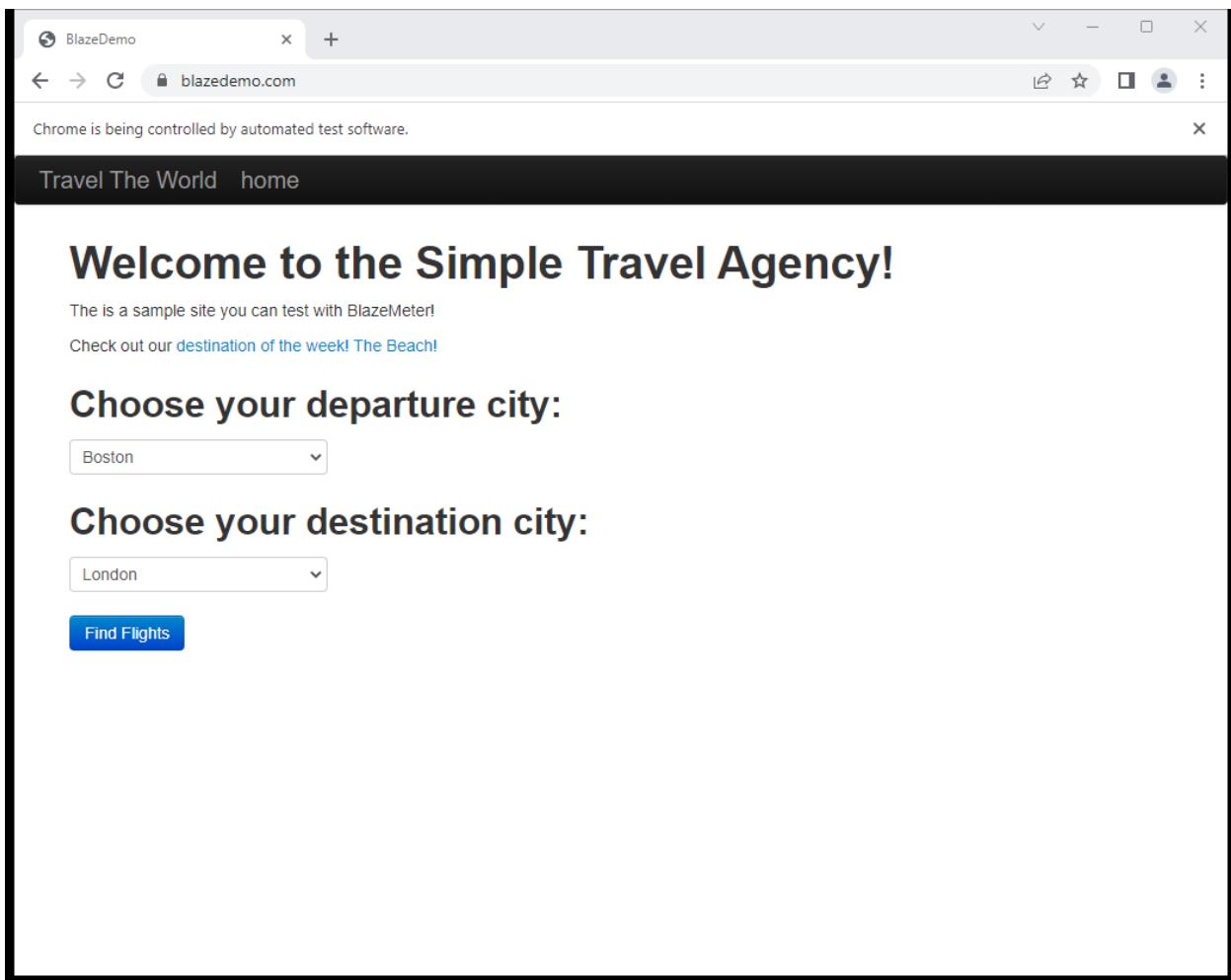
public class DropDown {

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver",
"E:\\\\Selenium\\\\chromedriver_win32\\\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();

        wd.get("https://blazemeter.com/");
        Select s = new Select(wd.findElement(By.name("fromPort")));
        Select s1 = new Select(wd.findElement(By.name("toPort")));
        //s.selectByIndex(1);
        //s.selectByValue("Boston");
        s.selectByVisibleText("Boston");
        s1.selectByVisibleText("London");

    }
}
```

Output:**List Boxes:**

```
import org.openqa.selenium.Alert; import
org.openqa.selenium.By; import
org.openqa.selenium.WebDriver; import
org.openqa.selenium.chrome.ChromeDriver; import
org.openqa.selenium.support.ui.Select; public class
ListBoxes {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        System.setProperty("webdriver.chrome.driver","C:\\Selenium\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();
    }
}
```

```
wd.get("file:///D:/SQLT/index.html");

wd.findElement(By.name("custname")).sendKeys("Shubham");

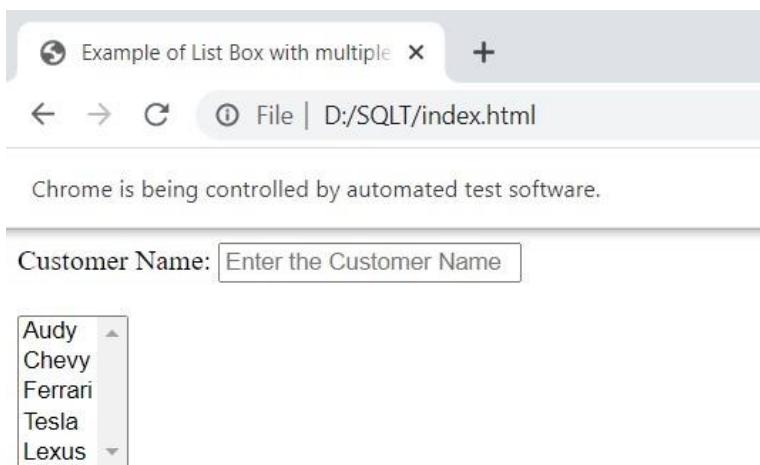
//Selecting the second checkbox using Xpath
//wd.findElement(By.xpath("/html/body/select/option[2]")).click();

Select s = new Select(wd.findElement(By.name("Cars")));
s.selectByIndex(1);

s.selectByVisibleText("Tesla");

}

}
```

Output:

Example of List Box with multiple X +

← → ⌂ ⚡ File | D:/SQLT/index.html

The screenshot shows a browser window with the title "Example of List Box with multiple". The address bar shows the file path "D:/SQLT/index.html". Below the title bar, there are navigation buttons (back, forward, search, etc.). A message "Chrome is being controlled by automated test software." is displayed. On the left, there is a form field labeled "Customer Name:" with an input box containing "Enter the Customer Name". To the right of the input box is a list box containing five items: "Audy", "Chevy", "Ferrari", "Tesla", and "Lexus". The item "Chevy" is highlighted with a blue selection bar.

Customer Name:



Example of List Box with multiple X +

← → ⌂ ⚡ File | D:/SQLT/index.html

The screenshot shows a browser window with the title "Example of List Box with multiple". The address bar shows the file path "D:/SQLT/index.html". Below the title bar, there are navigation buttons (back, forward, search, etc.). A message "Chrome is being controlled by automated test software." is displayed. On the left, there is a form field labeled "Customer Name:" with an input box containing "Enter the Customer Name". To the right of the input box is a list box containing five items: "Audy", "Chevy", "Ferrari", "Tesla", and "Lexus". The items "Chevy" and "Tesla" are highlighted with a blue selection bar.

Customer Name:



Practical No: 10

Aim: Demonstrate Command Button, Radio buttons & text boxes. Waits command in selenium

Button Clicks:

```
package practical13;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.*;
import org.openqa.selenium.WebElement;

public class Buttonclicks {

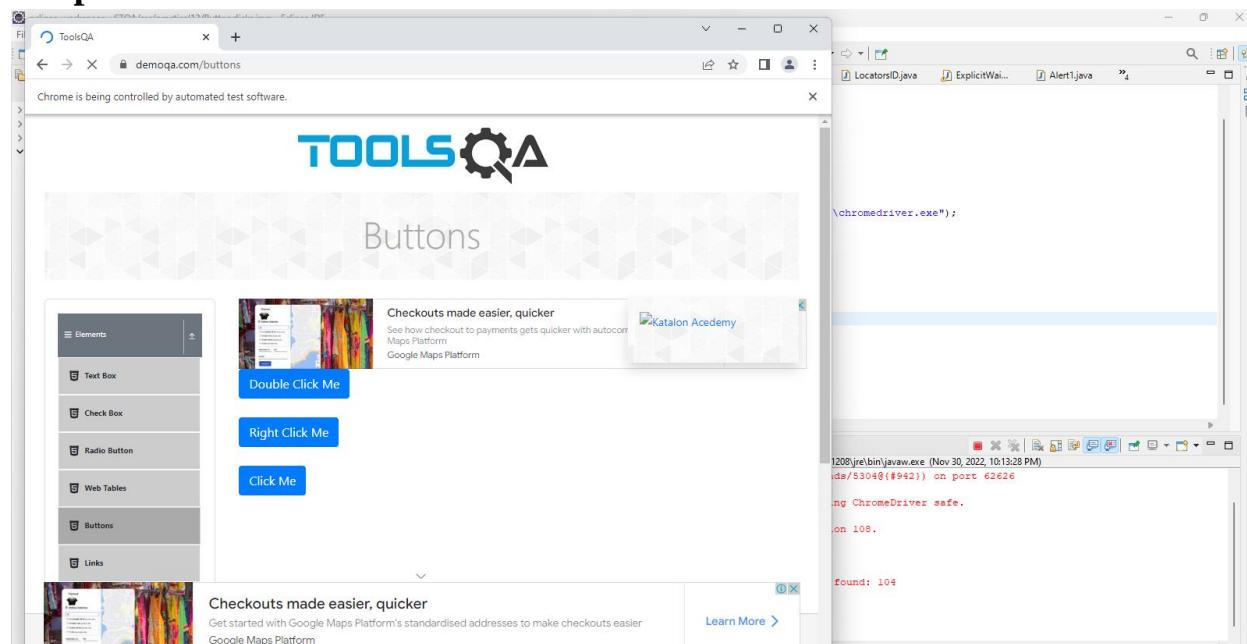
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();

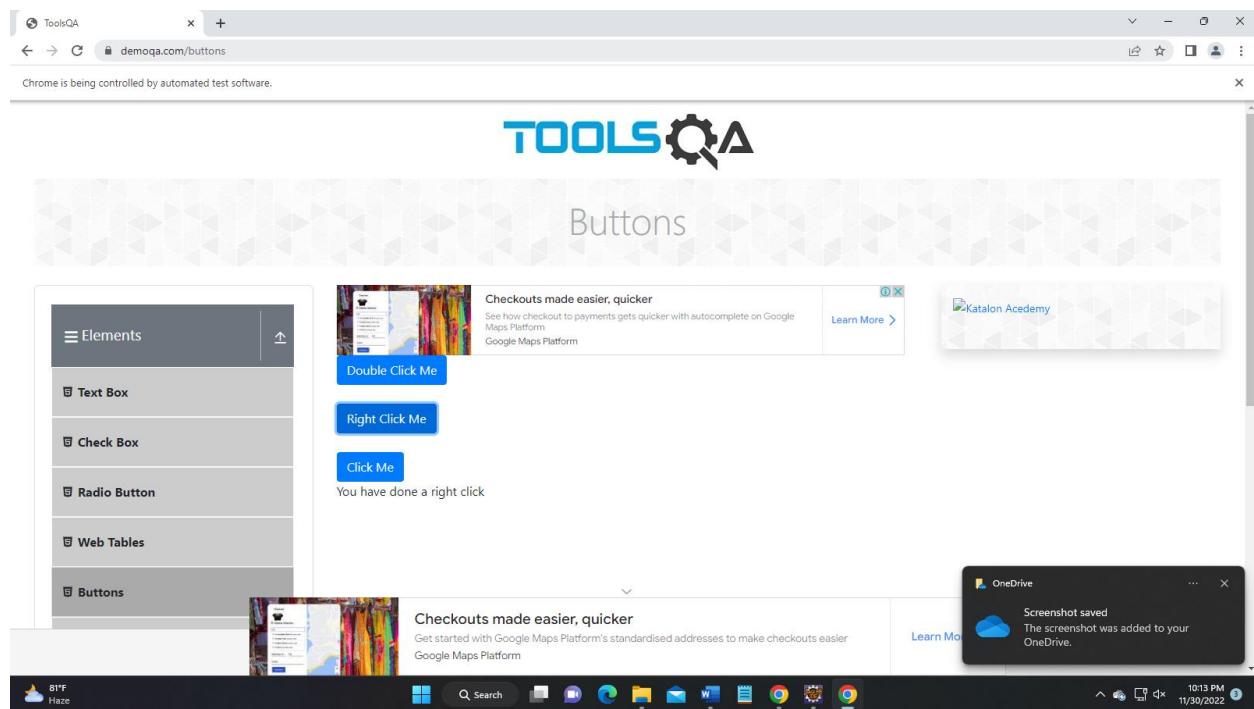
        wd.get("https://demoqa.com/buttons");
        wd.manage().window().maximize();

        Actions actions = new Actions(wd);

        WebElement btnElement = wd.findElement(By.id("rightClickBtn"));
        actions.contextClick(btnElement).perform();
        System.out.println("button is right Clicked");
    }
}
```

Output:





```
Markers Properties Servers Data Source Explorer Snippets Console X
<terminated> Buttonclicks [Java Application] C:\Users\SUSHANT\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
[1669826611.041][WARNING]: This version of ChromeDriver has not been tested with Chrome version 108.
Nov 30, 2022 10:13:31 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected upstream dialect: W3C
Nov 30, 2022 10:13:31 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 108, so returning the closest version found: 104
Nov 30, 2022 10:13:31 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found CDP implementation for version 108 of 104
button is right Clicked
```

Radio Button:

```
package practical13;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class RadiobButtons {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();

        wd.get("http://test.rubywatir.com/radios.php");

        wd.findElement(By.xpath("//*[@id='content']/div/div/div[2]/p/input[1]")).click();
        wd.findElement(By.className("radioclass")).click();

        wd.findElement(By.cssSelector("input[id='radioId']")).click();

        wd.findElement(By.linkText("Checkboxes")).click();
        wd.findElement(By.xpath("//*[@id='content']/div/div/div[2]/p/input[3]")).click();

    }
}
```

The screenshot shows a web browser window with the title "Home Page" and the URL "test.rubywatir.com/checkboxes.php". A message at the top states "Chrome is being controlled by automated test software." The main content is titled "RubyWatir Test" and "Multiple checkboxes page". It includes a note: "Note: Some checkboxes are checked by default". Below this, there is a form with the following options:

Soccer:	<input checked="" type="checkbox"/>
Football:	<input type="checkbox"/>
Baseball:	<input type="checkbox"/>
Basketball:	<input checked="" type="checkbox"/>
snooker:	<input type="checkbox"/>
Rugby:	<input type="checkbox"/>
Golf:	<input checked="" type="checkbox"/>
Netball:	<input checked="" type="checkbox"/>

On the left sidebar, under "LEVEL 1", there are links: Links, Checkboxes, Radio buttons, Tables, Maths, and Form. Under "EXERCISES", there are links: if else statements, loops, string manipulation, reg form, and a "USEFUL LINKS" section with a "Links" link.

Practical No: 11

Aim: Demonstrate action classes in Selenium

```

package actionclass;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

public class Actions_classss {

    public static void main(String[] args) throws Exception {

        System.setProperty("webdriver.chrome.driver",
"E:\\Selenium\\chromedriver_win32\\chromedriver.exe");
        WebDriver wd = new ChromeDriver();

        wd.get("https://opensource-demo.orangehrmlive.com/");
        Thread.sleep(500);

        wd.findElement(By.name("username")).sendKeys("Admin");
        wd.findElement(By.name("password")).sendKeys("admin123");

        wd.findElement(By.xpath("//*[@id='app']/div[1]/div/div[1]/div/div[2]/div[2]/form/div[3]/button")).click
();

        Actions act = new Actions(wd);

        //act.moveToElement (wd.findElement(By.className("oxd-text oxd-text--span oxd-main-menu-
item-name"))). perform();
        //act.moveToElement(wd.findElement(By.linkText("Recruitrent"))).perform();

        List<WebElement> menu =wd.findElements(By.className("oxd-main-menu-
item"));
        for (int i=0;i<=menu.size()-1;i++)
        {
            System.out.println(menu.get(i).getText());//print text of all the element
on console
            act.moveToElement(menu.get(i)).perform();

        }

        wd.findElement(By.xpath("//*[@id='app']/div[1]/div[1]/header/div[1]/div[2]/ul/li/span/i")).click();
        //locator partiallinkText
        wd.findElement(By.linkText("Logout")).click(); //locator linkText
        wd.close();
    }
}

```

```

    }
}

```

Output:

The screenshot displays two windows of the OrangeHRM application running on a Windows 10 desktop. The top window shows the 'Login' screen with the URL opensource-demo.orangehrmlive.com/web/index.php/auth/login. The login credentials entered are 'Username : Admin' and 'Password : admin123'. The bottom window shows the 'Dashboard' screen with the URL opensource-demo.orangehrmlive.com/web/index.php/dashboard/index. The dashboard includes sections for 'Time at Work' (Punched In Today at 07:43 PM (GMT 3)), 'My Actions' (Leave Requests to Approve, Timesheets to Approve, Pending Self Review, Candidate to Interview), and 'Quick Launch' (Assign Leave, Leave List, Timesheets). The desktop taskbar at the bottom shows various open applications like File Explorer, Task Manager, and a Java command-line interface window.

Practical No: 12**Aim: Installation of TestNg , running testNg and TestNg annotations****Program:**

```
package testNG;

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

public class Annotations {
    @Test(priority = 1) public
    void firstTest()
    {
        System.out.println("Test1");
    }

    @Test(priority = 0)
    public void secondTest()
    {
        System.out.println("Test2");
    }
    @BeforeMethod
    public void beforeMethod()
    {
        System.out.println("BeforeMethod");
    }

    @AfterMethod
    public void afterMethod()
```

```
{  
    System.out.println("AfterMethod");  
}  
  
@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:

```
<terminated> Annotations [TestNG] C:\Eclipse\eclipse\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v202
[RemoteTestNG] detected TestNG version 7.4.0
Before Suite
Before Test
Before Class
BeforeMethod
Test2
AfterMethod
BeforeMethod
Test1
AfterMethod
After Class
After Test
PASSED: secondTest
PASSED: firstTest

=====
Default test
Tests run: 2, Failures: 0, Skips: 0
=====

After Suite

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

Add Second test and run

```
public class DemoAnnotations {
    @Test
    public void firstTest()
    {
        System.out.println("Test1");
    }

    @Test
    public void secondTest()
    {
        System.out.println("Test2");
    }

    @BeforeMethod
    public void beforeMethod()
    {
        System.out.println("BeforeMethod");
    }
}

public class DemoAnnotations {
    @Test(priority = 1)
    public void firstTest()
    {
        System.out.println("Test1");
    }

    @Test(priority = 0)
    public void secondTest()
    {
        System.out.println("Test2");
    }
}
```

```
<terminated> DemoAnnotations [TestNG] C:\eclipse\plugins\org.eclipse.jst.j2ee\...
[RemoteTestNG] detected TestNG version 7.4.0
Before Suite
Before Test
Before Class
BeforeMethod
Test1
AfterMethod
BeforeMethod
Test2
AfterMethod
After Class
After Test
PASSED: secondTest
PASSED: firstTest
=====
Default test
Tests run: 2, Failures: 0, Skips: 0
=====
After Suite
=====
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
=====
```

```
<terminated> DemoAnnotations [TestNG] C:\eclipse\plugins\org.eclipse.jst.j2ee\...
[RemoteTestNG] detected TestNG version 7.4.0
Before Suite
Before Test
Before Class
BeforeMethod
Test2
AfterMethod
BeforeMethod
Test1
AfterMethod
After Class
After Test
PASSED: secondTest
PASSED: firstTest
=====
Default test
Tests run: 2, Failures: 0, Skips: 0
=====
After Suite
=====
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
=====
```

Practical No: 13

Aim: Demonstrate data driven Framework.

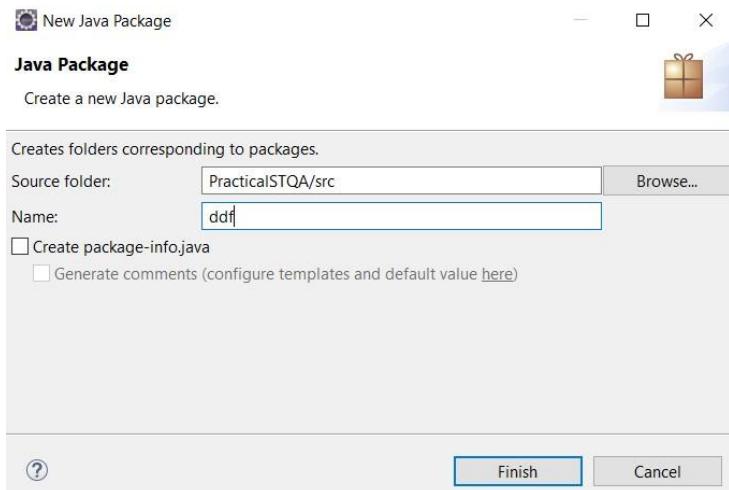
STEPS:

1.Right click on src folder

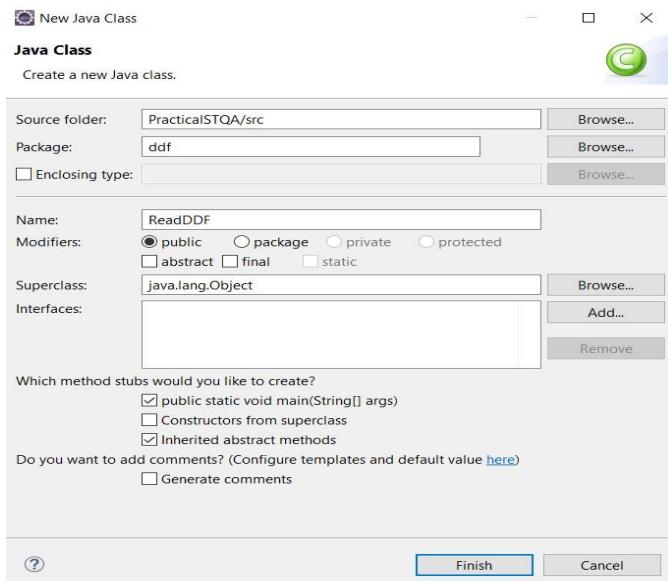
new

package(provide name ddf)

finish.



2. Now right click on package → new → class (provide the name ReadDDF) → select the main method → finish.



3.Go to the apache.org website and download the POI jar zip file. And then configured it with the project.

The screenshot shows the Apache Software Foundation homepage at https://apache.org. The main navigation bar includes links for News, About, Make a Donation, The Apache Way, Join Us, Downloads, and a search bar. The Apache logo is prominently displayed in the center. Below the logo, a grid lists various Apache projects, each with a small icon and a letter indicating its category (e.g., C, G, K, P, L). The project 'POI' is highlighted with a red box around its category letter 'P'. Other visible projects include Brooklyn, Flink, Flume, FreeMarker, Geode, Giraph, Gobblin, Gora, Griffin, jUDDI, jcclouds, Kafka, Karaf, Kibble, Knox, Kudu, Kylin, Libcloud, Logging Services, OpenWebBeans, OpenWhisk, Ozone, PDFBox, PLC4X, Parquet, Perl, Petri, Phoenix, Pig, Shiro, SkyWalking, Sling, Solr, SpamAssassin, Spark, Steve, Storm, Streams, Struts, Submarine, Subversion, Superset, Wicket, XML Graphics, Xalan, Xerces, Yetus, Zeppelin, and ZooKeeper.

The screenshot shows the Apache POI project page at https://poi.apache.org. The top navigation bar includes Home, Help, Component APIs, and Getting Involved. The left sidebar has a navigation tree with Overview (selected), Home (selected), Download, Changelog, Javadocs, Text Extraction, Encryption support, Case Studies, Related projects, Legal, and Apache Wide. The main content area features the Apache logo and a graphic of a hand holding a coffee cup. The title 'Apache POI - the Java API for Microsoft Documents' is displayed. A 'Project News' section highlights '1 November 2021 - POI 5.1.0 available'. The news text states: 'The Apache POI team is pleased to announce the release of 5.1.0. Several dependencies were updated. A summary of changes is available in the [Release Notes](#). A full list of changes is available in the [Change Log](#). See the [downloads](#) page for more details.' It also notes that 'POI requires Java 8 or newer since version 4.0.1.'

<https://poi.apache.org/download.html#POI-5.1.0>




Home Help Component APIs Getting Involved

Overview

- Home
- **Download**
- Changelog
- Javadocs
- Text Extraction
- Encryption support
- Case Studies
- Related projects
- Legal

▫ **Apache Wide**

Apache POI - Download Release Artifacts

Available Downloads

This page provides instructions on how to download and verify the Apache POI release artifacts.

- [The latest stable release is Apache POI 5.1.0](#)
- [Archives of all prior releases](#)

Apache POI releases are available under the [Apache License, Version 2.0](#). See the NOTICE file contained in each release.

Apache POI - Download Release Artifacts

Available Downloads

This page provides instructions on how to download and verify the Apache POI release artifacts. There are different ways to download the artifacts:

- [The latest stable release is Apache POI 5.1.0](#)
- [Archives of all prior releases](#)

Apache POI releases are available under the [Apache License, Version 2.0](#). See the NOTICE file contained in each release.

To ensure that you have downloaded the true release you should [verify the integrity](#) of the files using the signatures provided.

1 November 2021 - POI 5.1.0 available

The Apache POI team is pleased to announce the release of 5.1.0. Featured are a handful of new areas of functionality and improvements across the entire product line.

A summary of changes is available in the [Release Notes](#). A full list of changes is available in the [change log](#). People interested in the source code can clone the repository from [GitHub](#).

The POI source release as well as the pre-built binary deployment packages are listed below. Pre-built versions of all "org.apache.poi" and Version "5.1.0".

Binary Distribution

- [poi-bin-5.1.0-20211024.tgz](#) (57 MB, [signature \(.asc\)](#), checksum: [SHA-256](#), [SHA-512](#))
- [poi-bin-5.1.0-20211024.zip](#) (57 MB, [signature \(.asc\)](#), checksum: [SHA-256](#), [SHA-512](#))



COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

Projects ▾ People ▾ Community ▾ License ▾ Sponsors ▾

We suggest the following site for your download:

<https://dlcdn.apache.org/poi/release/bin/poi-bin-5.1.0-20211024.zip>

Alternate download locations are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature ([.asc](#) file) or a hash ([.md5](#) or [.sha*](#) file).

4. For data driven framework we need to have data file here we use Excel file as data. So create one excel file with some data.

B2	A	B	C	D	E
1	username	password			
2	admin	admin123			
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

Program:

```
package ddf;

import java.io.FileInputStream;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class ReadDDF

{
    public static void main(String[] args) throws Exception
    {
        System.setProperty("webdriver.chrome.driver","C:\\Selenium\\chromedriver.exe");
    }
}
```

```
WebDriver wd= new ChromeDriver();
wd.get("https://opensource-demo.orangehrmlive.com/");

FileInputStream fis = new FileInputStream("D:\\Selenium\\ExcelData.xlsx");
XSSFWorkbook wk = new XSSFWorkbook(fis);
XSSFSheet sh = wk.getSheet("ReadData");
for(int i=1; i<=sh.getLastRowNum(); i++)
{
    XSSFRow rw = sh.getRow(i);
    XSSFCell un = rw.getCell(0);
    XSSFCell pw = rw.getCell(1);

    System.out.println(un + " " + pw);

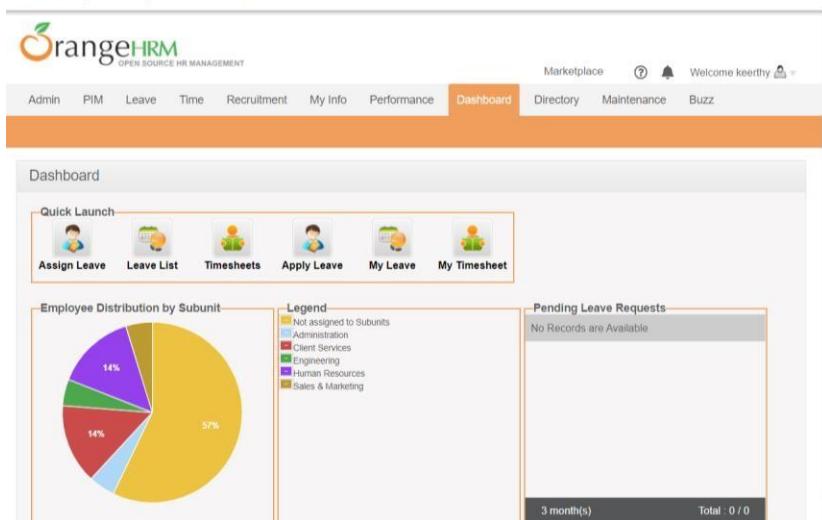
    wd.findElement(By.id("txtUsername")).sendKeys(un.toString());
    wd.findElement(By.name("txtPassword")).sendKeys(pw.toString());
    wd.findElement(By.className("button")).click();
}
}
```

Output:





(Username : Admin | Password : admin123)



```

Markers Properties Servers Data Source Explorer Snippets Coverage Console × Call Hierarchy Results of running class HRM

<terminated> ReadDDF [Java Application] C:\eclipse\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe
Starting ChromeDriver 96.0.4664.45 (76e4c1bb2ab4671b8beba3444e61c0f17584b2fc-refs/branch-heads/4664@{#947}) on port
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Dec 08, 2021 4:04:22 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
ERROR StatusLogger Log4j2 could not find a logging implementation. Please add log4j-core to the classpath. Using Sim
admin admin123

```