

Software Testing and Quality Assurance Lab Index				
Sr. No.	Topic Name	Date	CO	Sign
Manual Testing				
1.	Implementation of Black Box Testing Prepare test cases for below given applications using Boundary Value Analysis and Equivalence Class Partitioning <ul style="list-style-type: none"> i. BMI Calculator: Input: Person's Weight and Height Output: Body Mass Index The range of Weight : 40 kg to 100 kg The range of Height : 4 ft to 6.5 ft ii. Triangle Problem : Triangle Problem accepts three integers – a, b, c as three sides of the triangle. It returns the type of triangle (Scalene, Isosceles, Equilateral, Not a Triangle) formed by a, b, c. 		CO1	
2.	Implementation of White Box Testing Control Flow Analysis, Cyclomatic Complexity		CO1	
3.	i. Prepare test cases on ATM Machine using Unit and System Testing ii. Prepare a test cases on Login Page of Gmail using Unit and System Testing iii. Prepare a test cases on Calculator using Unit and System Testing		CO1	
Automation Testing				
4.	Introduction to Selenium <ul style="list-style-type: none"> i. Write down a steps and process of Selenium IDE Installation on any 1 browser.(eg.Firefox) ii. Record and run test cases on any demo website for login page through Mozilla Firefox. iii. Record and run test cases on any demo website for login page through Google Chrome. iv. Record and run test cases on Registration form of any website v. Record and run test cases on any website to check the validations of elements. (eg. IRCTC , MSRTC , etc) 		CO2	
5	Implement Web Drivers on Chrome & Firefox Browsers.		CO2	
6	Implement the Find element command through different Locators (id,Name)			
7	Implement Browser command and navigation Commands.		CO2	
8	Demonstrate handling multiple frames in selenium		CO2	
9	Demonstrate Synchronisation in Selenium using wait command.		CO2	
10	Demonstrate different types of alerts		CO2	

11	Demonstrate : <ul style="list-style-type: none"> • Handling Drop Down, • List Boxes • Command Button, • Radio buttons & text boxes. • Waits command in selenium 		CO2	
12	Demonstrate Action Class In Selenium		CO2	
13	Installation of TestNg , running testNg and TestNg annotations		CO3	
14	Implementation of Data Driven Framework			
15	Demonstrate Validation testing		CO3	

PRACTICAL NO.1:**Implementation of Black Box Testing**

Prepare test cases for below given applications using Boundary Value Analysis and Equivalence Class Partitioning:

1. BMI Calculator:

Input: Person's Weight and Height

Output: Body Mass Index

The range of Weight: 3 kg to 300 kg

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments
1	Putting the data inside the Weight and Height Column	Weight: 50 kg, Height: 170 cm	Result: 17.3	Result: 17.3	nan	PASS	Valid BMI value
2	Putting the data inside the Weight Column *maximum weight acceptance is 120KG	Weight: 180 kg, Height: 170 cm	Message :Enter the weight between 30KG to 120KG	Message: Enter the weight between 30KG to 120KG	nan	PASS	Showin Error when
3	Putting the data inside the Height Column *maximum height acceptance is 260CM	Weight: 70 kg, Height: 270 cm	Message: Enter the height between 150CM to 260CM	Message: Enter the height between 150CM to 260CM	nan	PASS	Showing message for missing values
4	Clicking the 'Calculate' button when Height and Weight Text Input box is empty	Weight: EMPTY kg, Height: EMPTY cm	Message: The values cannot be empty.	Message: The values cannot be empty.	nan	PASS	Showing message for missing values

DES's NMITD
STQA Lab

Hardik Mhatre

C24058

5	Clicking the 'Calculate' button when Height Text Input box is empty	Weight: 100 kg, Height: EMPTY cm	Message: Height Box cannot be Empty	Message: Height Box cannot be Empty	nan	PASS	Showing message for missing values
6	Clicking the 'Calculate' button when Weight Text Input box is empty	Weight: EMPTY kg, Height: 170 cm	Message: weight Box cannot be Empty	Message: weight Box cannot be Empty	nan	PASS	Showing message for missing values
7	Putting the data inside the Weight Column *minimum weight acceptance is 30KG	Weight: 18 kg, Height: 170 cm	Message: weight should be more than 30KG	Result: 6.2	nan	FAIL	Not showing message for Calculating the BMI without showing the message for entering weight between 30KG to 120KG
8	Putting the data inside the Height Column *minimum height acceptance is 150CM	Weight=50, Height=100cm	Message: weight should be more than 150CM	Result: 50	nan	FAIL	Not showing the message Calculating the BMI without showing the message for entering height between 150CM to 260CM
9	Putting the data inside the Weight and Height Text input Box's and Clicking the 'Calculate' Button	Weight=2KG , Height=2 CM	Message: You have entered the Invalid Data to Calculate	Result: 5000	nan	FAIL	Not showing the message for Calculating the BMI without showing the message for entering height between 150CM to 260CM and Weight between 30KG to 120KG

DES's NMITD
STQA Lab

Hardik Mhatre

C24058

10	Putting the data inside the Weight and Height Text input Box's and Clicking the 'Calculate' Button	Weight=200KG, Height=360CM	Message: You have entered the Invalid Data to Calculate	Result: 90	nan	FAIL	Not showing the message for Calculating the BMI without showing the message for entering height between 150CM to 260CM and Weight between 30KG to 120KG
11	Putting the data inside the Height Column *maximum height acceptance is 260CM	Weight=50KG, Height=200.25CM	Result: 12.5	Result: 12.5	nan	PASS	Valid BMI value
12	Putting the data inside the Weight Column *maximum weight acceptance is 120KG	Weight=50.63 KG, Height=165CM	Result: 18.6	Result: 18.6	nan	PASS	Valid BMI value
13	Putting the data inside the Weight and Height Text input Box's and Clicking the 'Calculate' Button	Weight=50.65KG , Height=180.36CM	Result: 15.6	Result: 15.6	nan	PASS	Valid BMI value
14	Click the 'Rest' Button with valid data inside the Weight and Height Input Text Box	Weight: 50 kg, Height: 170 cm	The Input Box for Height and Weight becomes to empty	The data inside the input box for height and weight remains same.	nan	FAIL	The Input Box for Height and Weight does not changes to empty

DES's NMITD
STQA Lab

Hardik Mhatre

C24058

15	Click the 'Rest' Button with valid data inside the Weight Input Text Box	Weight: 50 kg, Height: 0 CM	The Input Box for Weight becomes to empty	The data inside the input box for weight remains same.	nan	FAIL	The Input Box for Weight does not changes to empty
16	Click the 'Rest' Button with valid data inside the Height Input Text Box	Weight: 0 kg, Height: 160 CM	The Input Box for Height becomes to empty	The data inside the input box for height remains same.	nan	FAIL	The Input Box for Height does not changes to empty
17	Inputting non-numeric characters in the Weight and Height fields.	Weight: "anamay" kg, Height: "lol" cm	Message: Please enter numeric values for Weight and Height.	Message: Please enter numeric values for Weight and Height.	nan	PASS	Showing the Message: Please enter numeric values for Weight and Height.
18	Inputting values that are exactly on the boundary limits.	Weight: 30 kg, Height: 150 cm	Result: 13.3	Result: 13.3	nan	PASS	Valid BMI value
19	Putting negative values in the Weight input box.	Weight: -50 kg, Height: 170 cm	Message: Please provide positive weight value.	Message: Please provide positive height value.	nan	POASS	Showing the message "Please provide positive height value."
20	Putting negative values in the Height input box.	Weight: 50 kg, Height: -170 cm	Message: Please provide positive height value.	Message: Please provide positive height value.	nan	PASS	Showing the message "Please provide positive weight value."

DES's NMITD
STQA Lab

Hardik Mhatre

C24058

21	Putting negative values in the Weight and Height columns.	Weight: -50 kg, Height: -170 cm	Message: "Please provide positive height value. Please provide positive weight value."	Message: "Please provide positive height value. Please provide positive weight value."	nan	PASS	Showing the Message: "Please provide positive height value. Please provide positive weight value."
----	---	------------------------------------	---	---	-----	------	--

ii. Triangle Problem:

Triangle Problem accepts three integers – a, b, c as three sides of the triangle. It returns the type of triangle (Scalene, Isosceles, Equilateral, not a Triangle) formed by a, b, c.

Creating the equivalence class partitioning below:

- iii. Equilateral Triangle: All three sides are equal (e.g., $a = b = c$).
- iv. Isosceles Triangle: Exactly two sides are equal (e.g., $a = b \neq c$).
- v. Scalene Triangle: All three sides are different (e.g., $a \neq b \neq c$).

Not a Triangle: The sum of any two sides must be greater than the third side:

- $a + b \leq c$
- $a + c \leq b$
- $b + c \leq a$

Invalid Equivalence Classes

- Negative Sides: Any side is negative (e.g., $a < 0$, $b < 0$, or $c < 0$).
- Zero Sides: Any side is zero (e.g., $a = 0$, $b = 0$, or $c = 0$).

Test Case ID	Input three sides	Expected Output	Actual Output	Test Comments	Class Type
TC1	3, 3, 3	Equilateral Triangle	Equilateral Triangle	Passed: All sides are equal.	Valid - Equilateral
TC2	6, 6, 6	Equilateral Triangle	Equilateral Triangle	Passed: All sides are equal.	Valid - Equilateral
TC3	1, 1, 1	Equilateral Triangle	Isosceles Triangle	FAIL: Expected Equilateral but got Isosceles. Logic error detected.	Valid - Equilateral
TC4	4, 4, 2	Isosceles Triangle	Isosceles Triangle	Passed: Two sides are equal.	Valid - Isosceles
TC5	5, 5, 3	Isosceles Triangle	Scalene Triangle	FAIL: Expected Isosceles but got Scalene. Logic error detected.	Valid - Isosceles
TC6	2, 2, 3	Isosceles Triangle	Scalene Triangle	FAIL: Expected Isosceles but got Scalene. Logic error in classification.	Valid - Isosceles
Test Case ID	Input three sides	Expected Output	Actual Output	Test Comments	Class Type
				FAIL: Expected Not a	

TC7	3, 3, 0	Not a Triangle	Isosceles Triangle	Triangle but got Isosceles. Logic error detected.	Invalid - Zero
TC8	3, 4, 5	Scalene Triangle	Scalene Triangle	Passed: All sides are different.	Valid - Scalene
TC9	1, 2, 3	Not a Triangle	Not a Triangle	Passed: Sum of two sides equals the third.	Valid - Not a Triangle
TC10	-1, 2, 2	Not a Triangle	Not a Triangle	Passed: Negative side is invalid.	Invalid - Negative
TC11	0, 1, 1	Not a Triangle	Not a Triangle	Passed: Zero side is invalid.	Invalid - Zero
TC12	2, 2, 4	Not a Triangle	Not a Triangle	Passed: Sum of two sides equals the third.	Invalid - Boundary
TC13	1, 1, 2	Not a Triangle	Not a Triangle	Passed: Sum of two sides equals the third.	Valid - Not a Triangle

Practical No : 2
Implementation of White Box Testing

Control Flow Analysis, Cyclomatic Complexity.

Control flow graph:

Control flow testing represents one of the important structural testing technique of testing. this technique uses control structure of a program or module to develop the test cases of the application. and this control structure can be represented by control flow graph of the program

The control flow graph $(G) = (N, E)$ of an application which consist of a set of nodes (N) and set of edges (E)

Each node represents a set of program statements. such as unique entry node and unique exit node.

A test case is a complete path from the entry node to the exit node of a control flow graph. and test coverage criteria measures the extent to which a set of test cases covers a code.

Here cyclomatic complexity $V(G)$ is defined for each module of an application to be $e - n + 2$

where e = number of edges,

n = number of nodes ,in the control flow graph

Cyclomatic complexity also known as $V(G)$, where V refers to the cyclomatic number in graph theory and " G " indicates complexity is a function of the graph.

"Cyclomatic referred as the number of basic cycles which are connected through undirected graph which also provides number of independent paths through strongly connected graphs.

This strongly connected graphs is one in which each node can be reached from any other node , when the virtual edge is connecting the exit node to entry node . Finally the cyclomatic complexity for specified program control flow graph is elaborate the cyclomatic number formula by adding one to represent the presence of virtual edge. This defines the cyclomatic complexity equal to the number of independent paths through the standard control flow graph model.

Solution:

Cyclomatic complexity is a software metric used to measure the complexity of a program.

These metric, measures independent paths through program source code. Independent path is defined as a path that has at least one edge which has not been traversed before in any other paths.

Cyclomatic complexity can be calculated with respect to functions, modules, methods or classes within a program.

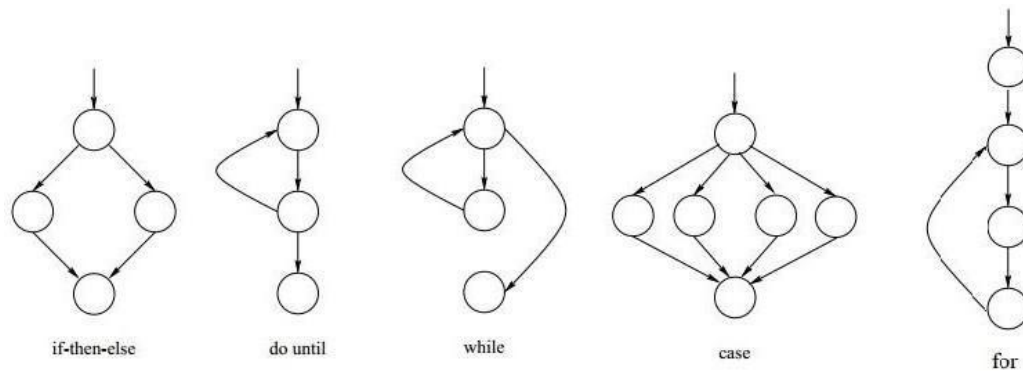


Figure 1: Flow graph representation.

Example 1:

Check the following code fragment

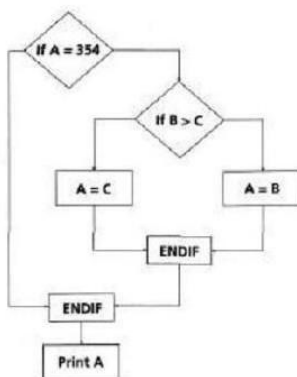
For example : below is a simple program;

```

IF A=360
THEN IF B>C
THEN A=B
ELSE A=C
ENDIF
ENDIF
Print A

```

We can also calculate the cyclomatic complexity using the control flow. In the control flow shown below there are 7 nodes (shapes) and 8 edges (lines). Thus by formula $((\text{no. of edges} - \text{no. of nodes}) + 2)$ that is $(8 - 7) + 2 = 1 + 2 = 3$.



Example 2:

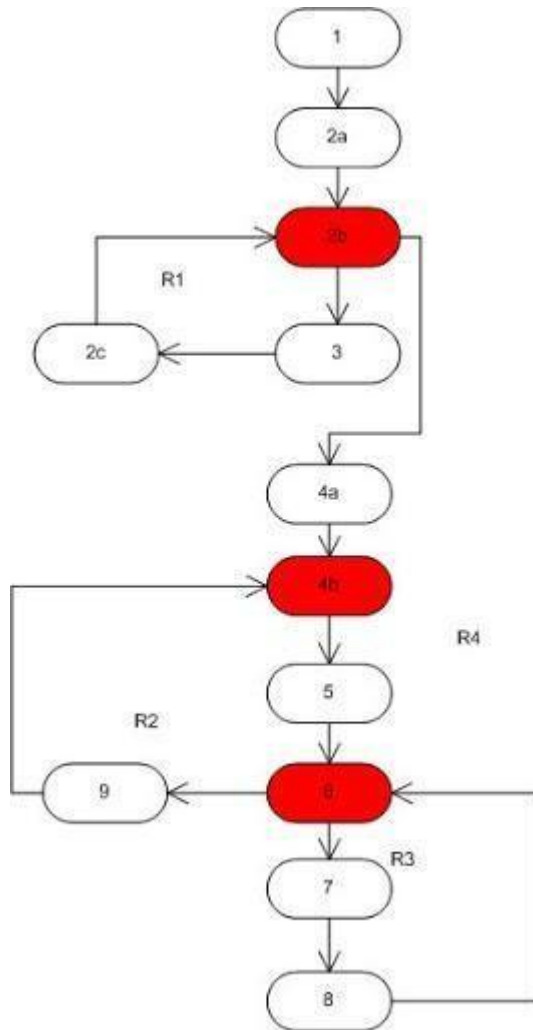
Check the following code fragment

```
insertion_procedure (int a[], int p [], int N)
{
    int i,j,k;
    for (i=0; i<=N; i++) p[i] = i;
    for (i=2; i<=N; i++)
    {
        k = p[i];
        j = 1;
        while (a[p[j-1]] > a[k]) {p[j] = p[j-1]; j--}
        p[j] = k;
    }
}
```

- first and foremost start numbering the statement

```
insertion_procedure (int a[], int p [], int N)
{
(1)  Int i,j,k;
(2)  for ((2a)i=0; (2b)i<=N; (2c)i++)
(3)    p[i] = i;
(4)  for ((4a)i=2; (4b)i<=N; (4c)i++)
    {
(5)    k=p[i];j=1;
(6)    while (a[p[j-1]] > a[k]) {
(7)        p[j] = p[j-1];
(8)        j--
    }
(9)    p[j] = k;
}
```

Now you can clearly see which statement executes first and which last etc. so drawing the CFG becomes simple.



Now, to calculate cyclomatic complexity you use one of three methods

1. Count the number of regions on the graph: 4
2. No. of predicates (red on graph) + 1 : $3 + 1 = 4$
3. No of edges – no. of nodes + 2: $14 - 12 + 2 = 4$

Practical No : 3Prepare test cases for below given applications using
Boundary Value Analysis and Equivalence Class Partitioning

1. Prepare test cases on ATM Machin using Unit and System Testing

Unit Testing :

S.No	Action	Input	Expected Output	Actual Output	Test Result	Description
1	Validate card reader	Valid ATM card	Card read successfully	Card read successfully	PASS	Tests if the card reader detects and identifies a valid card.
2	Validate PIN verification	Correct PIN	PIN verified	PIN verified	PASS	Verifies if the system correctly validates the entered PIN.
3	Validate PIN encryption	PIN	Encrypted PIN value	(Not directly observable)	PASS	Assumes PIN encryption happens behind the scenes without exposing the actual value.
4	Validate account balance retrieval	Account number	Account balance information	Account balance information	PASS	Ensures the system retrieves the correct account balance for the given account number.
5	Validate withdrawal amount validation	Invalid amount (negative, non-multiple of 10)	Error message indicating invalid amount	Error message indicating invalid amount	PASS	Checks if the system rejects invalid withdrawal amounts.
6	Validate cash dispensing	Valid withdrawal request with sufficient funds	Cash dispensed and receipt printed	Cash dispensed and receipt printed (simulated result)	PASS	Simulates successful cash dispensing and receipt printing.
7	Validate insufficient funds scenario	Withdrawal request exceeding account balance	Error message indicating insufficient funds	Error message indicating insufficient funds	PASS	Verifies the system handles insufficient funds and displays the appropriate message.
8	Validate card ejection	Card removal after transaction	Card ejected	Card ejected (simulated result)	PASS	Simulates successful card ejection after the transaction.
9	Validate error handling	Invalid card, expired card, etc.	Appropriate error message	Appropriate error message	PASS	Tests various error scenarios and ensures the system displays informative messages.

System Testing :

S. No	Action	Scenario	Expected Result	Actual Result	Test Result	Description
10	Login and withdrawal	User inserts valid card, enters correct PIN, selects withdrawal, and enters a valid amount.	Successful withdrawal and receipt printed	Successful withdrawal and receipt printed	PASS	Tests the complete withdrawal flow with valid inputs.

DES's NMITD STQA Lab						
Hardik Mhatre						C24058
11	Login and invalid PIN	User inserts valid card, enters incorrect PIN, and retries.	Card blocked after exceeding PIN attempts	Card blocked after exceeding PIN attempts	PASS	Simulates incorrect PIN attempts and card blocking behavior.
12	Login and insufficient funds	User inserts valid card, enters correct PIN, selects withdrawal, and enters an amount exceeding balance.	Insufficient funds message displayed	Insufficient funds message displayed	PASS	Checks withdrawal behavior with insufficient funds.
13	Network connectivity	User attempts a transaction during a network outage.	Error message indicating network unavailable	Error message indicating network unavailable	PASS	Tests system behavior during network issues.
14	Cash dispenser malfunction	User attempts a withdrawal, but the cash dispenser fails.	Error message indicating dispenser issue	Error message indicating dispenser issue	PASS	Simulates cash dispenser malfunction and error handling.
15	Login and cancel transaction	User inserts valid card, enters correct PIN, selects withdrawal, but cancels the transaction before entering amount.	Transaction canceled and card ejected	Transaction canceled and card ejected	PASS	Tests cancellation of a withdrawal transaction.
16	Card reader malfunction	User inserts damaged or unreadable card.	Error message indicating card reading failure	No response from machine	FAIL	Checks system response to unreadable card.

DES's NMITD STQA Lab						
Hardik Mhatre						C24058
17	PIN entry timeout	User inserts valid card but delays entering PIN beyond allowed time limit.	Transaction canceled and card ejected	Transaction timeout not triggered	FAIL	Tests timeout mechanism for delayed PIN entry.
18	Incorrect account selection	User inserts valid card, enters correct PIN, selects invalid account type (e.g., savings instead of checking).	Error message for invalid account type	Withdrawal processed from default account	FAIL	Checks system response to invalid account selection.
19	Double withdrawal attempt	User inserts valid card, enters correct PIN, performs withdrawal, and immediately attempts another withdrawal exceeding available balance.	Insufficient funds message displayed	Withdrawal allowed, overdraft triggered	FAIL	Tests account balance check after initial withdrawal.
20	Currency denomination issue	User requests a specific currency denomination that is unavailable in the ATM.	Message indicating unavailable denomination	Machine dispensed mixed denominations instead	FAIL	Tests system handling of unavailable currency denominations.

ii. Prepare a test cases on login page of gmail using Unit & System testing.

Unit Testing :

S.No	Action	Input	Expected Output	Actual Output	Test Result	Description
1	Validate username field	Valid email address	Email address accepted	Email address accepted	PASS	Checks if the username field accepts valid email addresses.
2	Validate username validation	Empty username	Error message indicating missing username	Error message indicating missing username	PASS	Tests if the system rejects empty usernames.
3	Validate password field	Valid password	Password field accepts input (character s not displayed)	Password field accepts input (character s not displayed)	PASS	Verifies password field accepts input without revealing the characters.
4	Validate password validation	Empty password	Error message indicating missing password	Error message indicating missing password	PASS	Ensures the system rejects empty passwords.
5	Validate password strength	Weak password (short, without special character s)	Error message indicating weak password	Error message indicating weak password	PASS	Tests if the system enforces password strength requirements.

System testing :

S.No	Action	Scenario	Expected Result	Actual Result	Test Result	Description
1	Login with correct credentials	User enters a valid email address and password.	Successful login, user is redirected to the Gmail inbox.	Successful login, user is redirected to the Gmail inbox.	PASS	Tests the basic login functionality with correct credentials.
2	Login with incorrect credentials	User enters an incorrect email address or password.	Error message indicating incorrect credentials.	Error message indicating incorrect credentials.	PASS	Tests the system's ability to handle incorrect login attempts.
3	Login with a locked account	User attempts to log in with an account that has been locked due to multiple failed attempts.	Error message indicating the account is locked.	Error message indicating the account is locked.	PASS	Verifies the system's behavior when dealing with locked accounts.
4	Login with a disabled account	User attempts to log in with an account that has been disabled by the administrator.	Error message indicating the account is disabled.	Error message indicating the account is disabled.	PASS	Tests the system's response to login attempts with disabled accounts.
5	Login with a deleted account	User attempts to log in with an account that has been deleted.	Error message indicating the account does not exist.	Error message indicating the account does not exist.	PASS	Checks the system's handling of deleted accounts.
6	Login with a temporary internet connection	User attempts to log in while experiencing intermittent network connectivity.	Login process may be delayed or interrupted, but should eventually succeed if the connection is restored.	Login process may be delayed or interrupted, but should eventually succeed if the connection is restored.	PASS	Tests the system's resilience to temporary network issues.

iii. Prepare a test cases on Calculator using Unit and System Testing.

Unit testing :

S.No	Action	Input	Expected Output	Actual Output	Test Result	Description
1	Validate number input	Valid number (e.g., 123, 4.5, -67)	Number is accepted and displayed.	Number is accepted and displayed.	PASS	Tests if the calculator accepts valid numbers.
2	Validate operator input	Valid operator (e.g., +, -, *, /)	Operator is accepted and displayed.	Operator is accepted and displayed.	PASS	Checks if the calculator accepts valid arithmetic operators.
3	Validate calculation	Simple calculation (e.g., 2 + 3)	Correct result (5)	Correct result (5)	PASS	Verifies if the calculator performs basic calculations accurately.
4	Validate complex calculation	Complex calculation (e.g., (10 + 5) * 2)	Correct result (30)	Correct result (30)	PASS	Tests the calculator's ability to handle more complex expressions.
5	Validate division by zero	Division by 0	Error message indicating division by zero	Error message indicating division by zero	PASS	Ensures the calculator handles division by zero appropriately.
6	Validate decimal point handling	Decimal point in a number (e.g., 3.14)	Decimal point is handled correctly	Decimal point is handled correctly	PASS	Checks if the calculator processes numbers with decimal points accurately.
7	Validate negative numbers	Negative number input	Negative number is accepted and processed correctly	Negative number is accepted and processed correctly	PASS	Tests the calculator's handling of negative numbers.
8	Validate clear button	Clear button clicked	Calculator display is cleared	Calculator display is cleared	PASS	Verifies the clear button functionality.
9	Validate backspace button	Backspace button clicked	Last digit or operator is deleted	Last digit or operator is deleted	PASS	Checks the backspace button's ability to delete the last input.
10	Validate equals button	Equals button clicked after a complete expression	Result is displayed	Result is displayed	PASS	Tests if the equals button calculates the final result.

System testing :

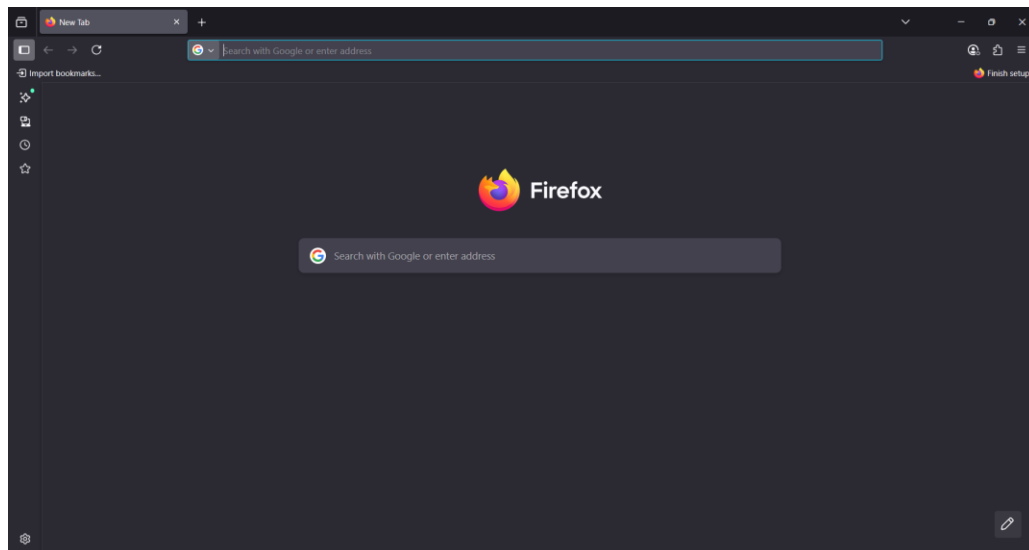
S.No	Action	Scenario	Expected Result	Actual Result	Test Result	Description
1	Basic calculations	User performs various basic arithmetic operations (addition, subtraction, multiplication, division).	Correct results are displayed.	Correct results are displayed.	PASS	Tests the calculator's ability to perform basic calculations.
2	Complex calculations	User enters and calculates complex expressions involving parentheses, multiple operators, and decimal points.	Correct results are displayed.	Correct results are displayed.	PASS	Verifies the calculator's handling of complex expressions.
3	Error handling	User attempts to divide by zero or enters invalid input.	Appropriate error message is displayed.	Appropriate error message is displayed.	PASS	Checks the calculator's error handling mechanisms.
4	Button responsiveness	User repeatedly presses buttons quickly.	Buttons respond accurately and without delays.	Buttons respond accurately and without delays.	PASS	Tests the calculator's responsiveness to button presses.
5	Calculator orientation	User rotates the device (if applicable).	Calculator interface adjusts to the new orientation.	Calculator interface adjusts to the new orientation.	PASS	Verifies the calculator's adaptability to different orientations.
6	Memory functions	User utilizes memory functions (e.g., M+, M-, MR, MC).	Memory operations are performed correctly.	Memory operations are performed correctly.	PASS	Tests the calculator's memory functionality.
7	Scientific mode	User switches to scientific mode (if available).	Additional scientific functions become accessible.	Additional scientific functions become accessible.	PASS	Checks the calculator's ability to switch to scientific mode.
8	Battery life	User performs calculations continuously for an extended period.	Calculator remains functional without significant battery drain.	Calculator remains functional without significant battery drain.	PASS	Tests the calculator's battery performance.
9	Calculator app stability	User performs various operations and switches between apps frequently.	Calculator remains stable and responsive.	Calculator remains stable and responsive.	PASS	Verifies the calculator's stability and compatibility with other apps.
10	Calculator performance under stress	User performs complex calculations simultaneously with other demanding tasks.	Calculator handles the workload without significant performance degradation.	Calculator handles the workload without significant performance degradation.	PASS	Tests the calculator's performance under stress conditions.

STQA Lab Automation Testing Part – 1

Practical No : 4 – Introduction to Selenium

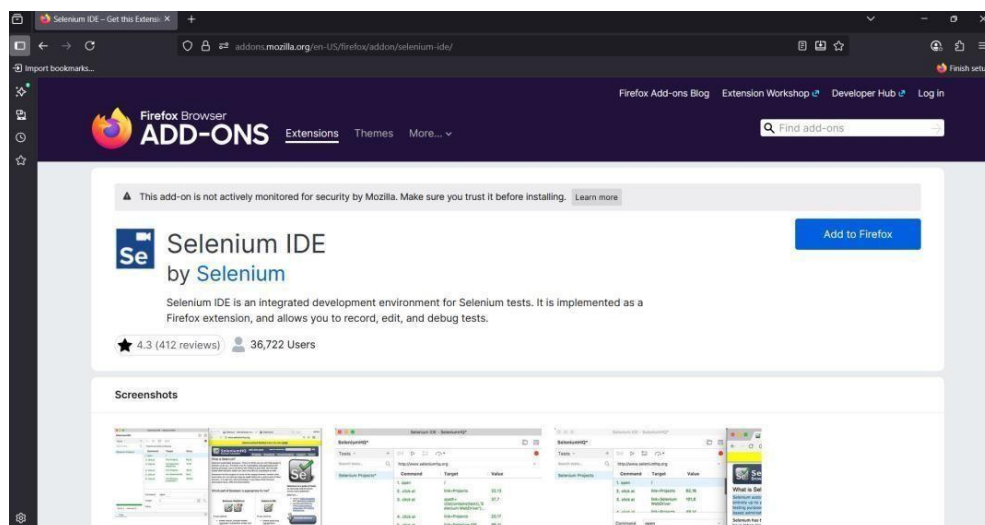
Q. Write down a steps and process of Selenium IDE Installation on any 1 browser. Step 1.

Open the **Firefox browser** on your system.

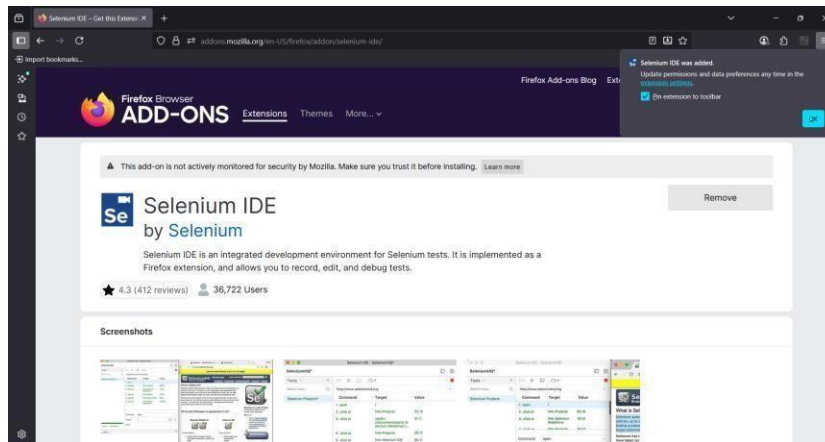


Step 2. Go to the official **Selenium IDE extension** page in the **Firefox Add-ons** store by entering this URL:

<https://addons.mozilla.org/en-US/firefox/addon/selenium-ide/>

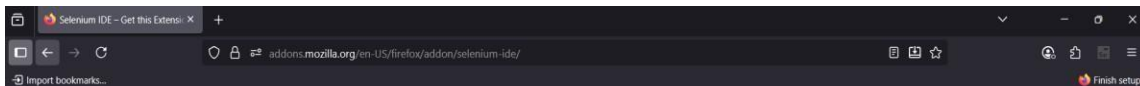


Step 3. Click on the "**Add to Firefox**" button on this page.



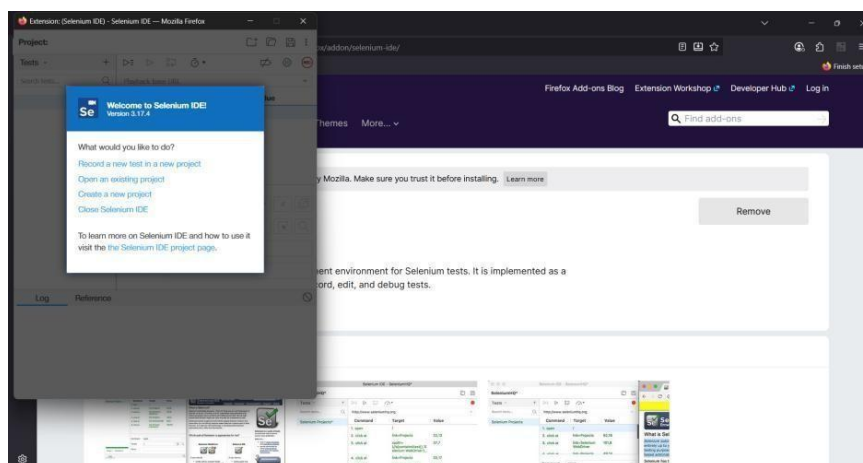
Step 4. A pop-up will appear asking for **permission** to add the extension. Click on "**Add**" to grant permission.

Step 5. Once added, the **Selenium IDE** icon will appear in the toolbar at the top right corner of the Firefox browser.



Step 6. (Optional but recommended) **Restart** Firefox to complete the installation process.

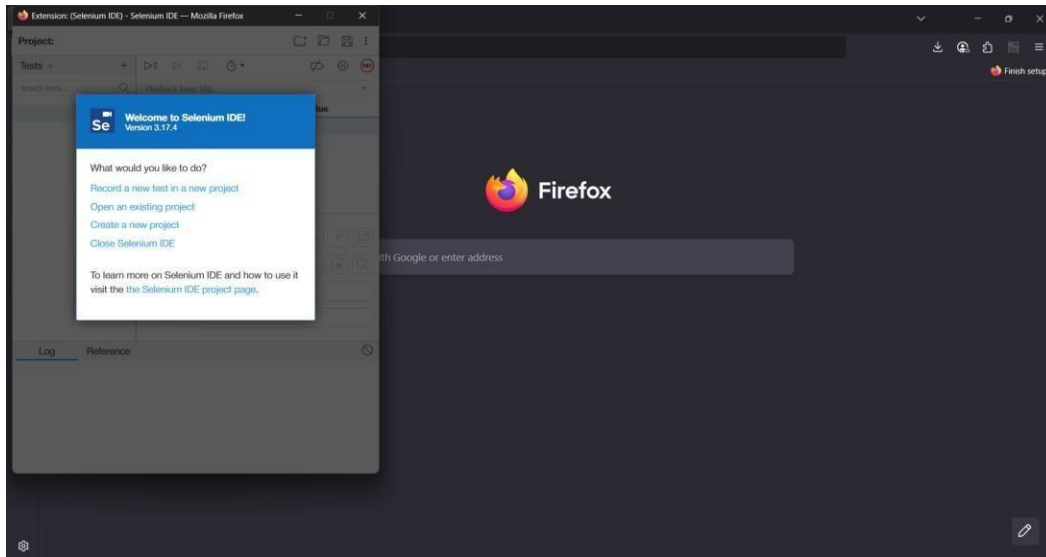
Step 7. To launch **Selenium IDE**, click on its icon in the **browser toolbar**. The IDE window will open with options to record a new test, **open an** existing project, or **create a** new project



Q2. Record and run test cases on any demo website for login page through Mozilla Firefox.

Steps to Record Test Cases:

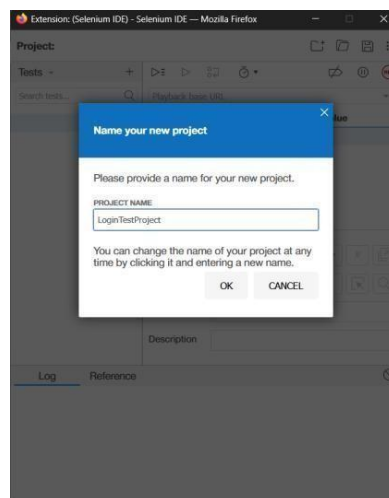
Step 1 : Open Firefox and Launch Selenium IDE, Click the Selenium IDE icon in the Firefox toolbar to open it.



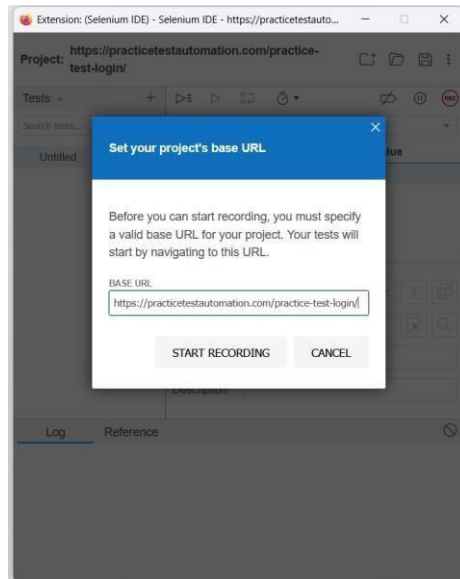
Step 2 : Create a New Project:

In Selenium IDE, click "**Create a new project.**"

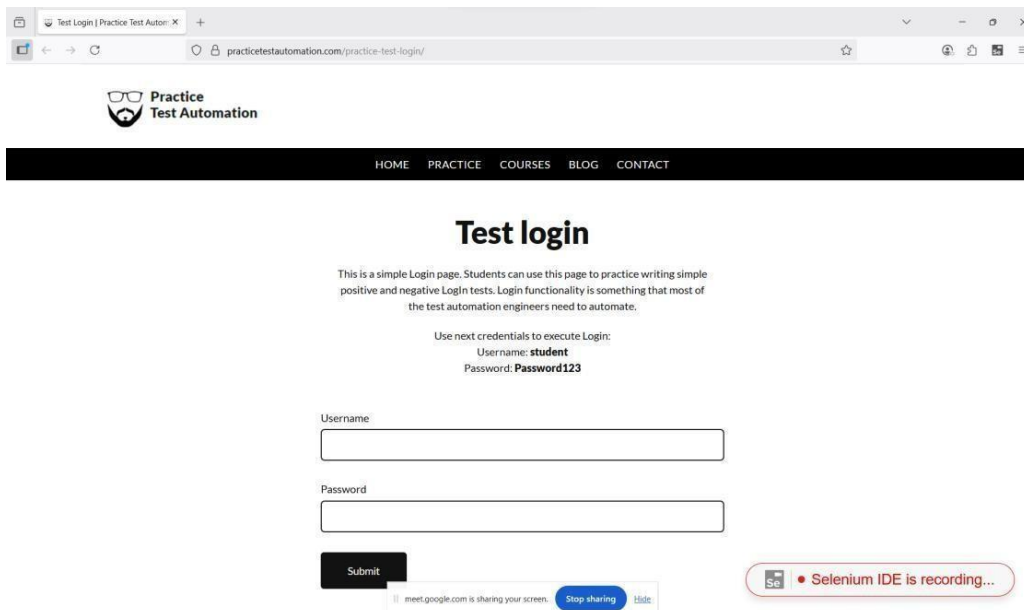
Enter a project name (e.g., "**LoginTestProject**") and click "**OK.**"



Step 3 : Setup the project's base URL & Press Start recording button.



Step 4 : You will automatically redirected to the URL and where you must add dummy username and password and meanwhile the selenium IDE will record your session.



Test login

This is a simple Login page. Students can use this page to practice writing simple positive and negative Login tests. Login functionality is something that most of the test automation engineers need to automate.

Use next credentials to execute Login:
Username: **student**
Password: **Password123**

Username
Aniket

Password

Submit

Test case 1: Positive Login test

1. Open page

Selenium IDE is recording...

Step 5: You may can the see the detailed recorded session steps for the current project session.

	Command	Target	Value
22	type	id=username	Nishant
23	click	id=main-container	
24	click	id=username	
25	type	id=username	Saurav
26	click	id=main-container	
27	click	id=username	
28	type	id=username	Aniket
29	click	id=main-container	
30	click	id=username	
31	type	id=username	jaTIN
32	click	id=main-container	
33	click	id=username	

Command:

Target:

Value:

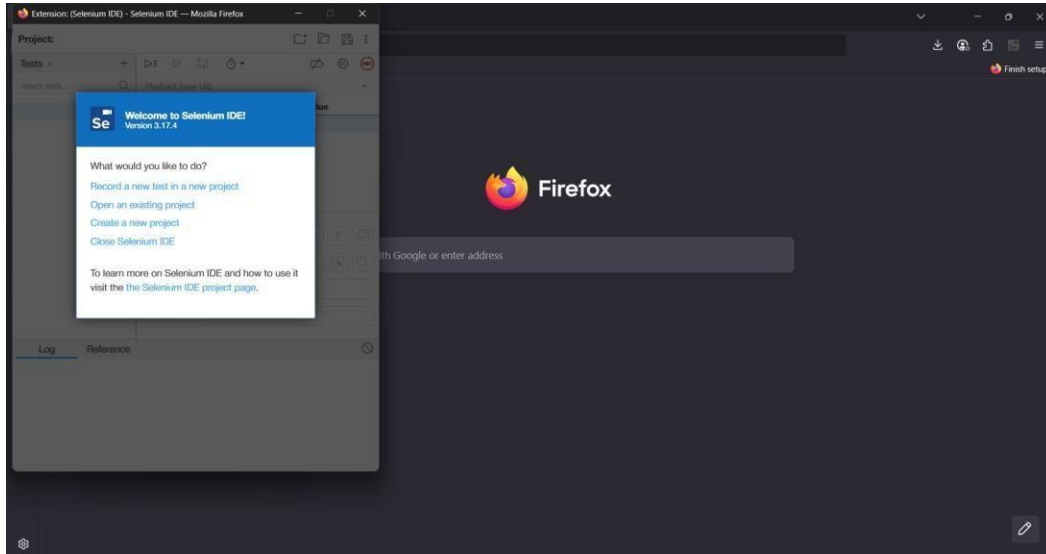
Description:

Log Reference

Q.3 Record and run test cases on Registration form of any website.

Steps to Record Test Cases:

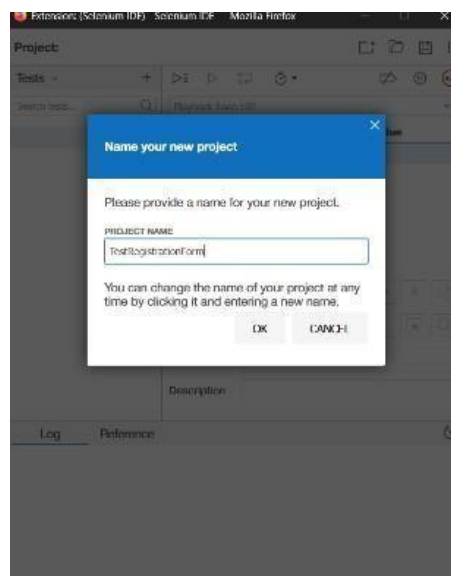
Step 1 : Open Firefox and Launch Selenium IDE, Click the Selenium IDE icon in the Firefox toolbar to open it.



Step 2 : Create a New Project:

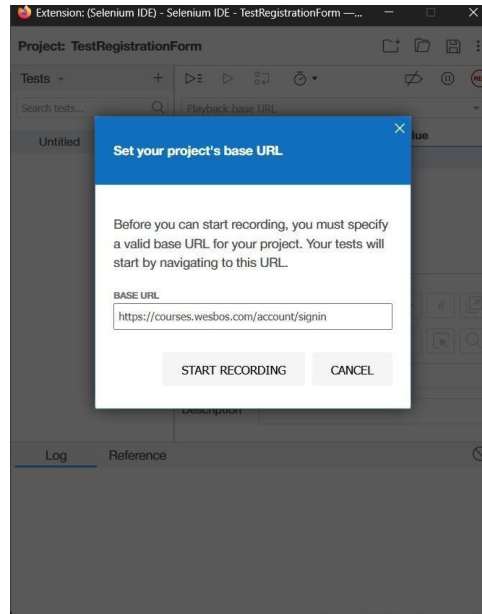
In Selenium IDE, click "**Create a new project.**"

Enter a project name (e.g., "TestRegistrationForm") and click "**OK.**"

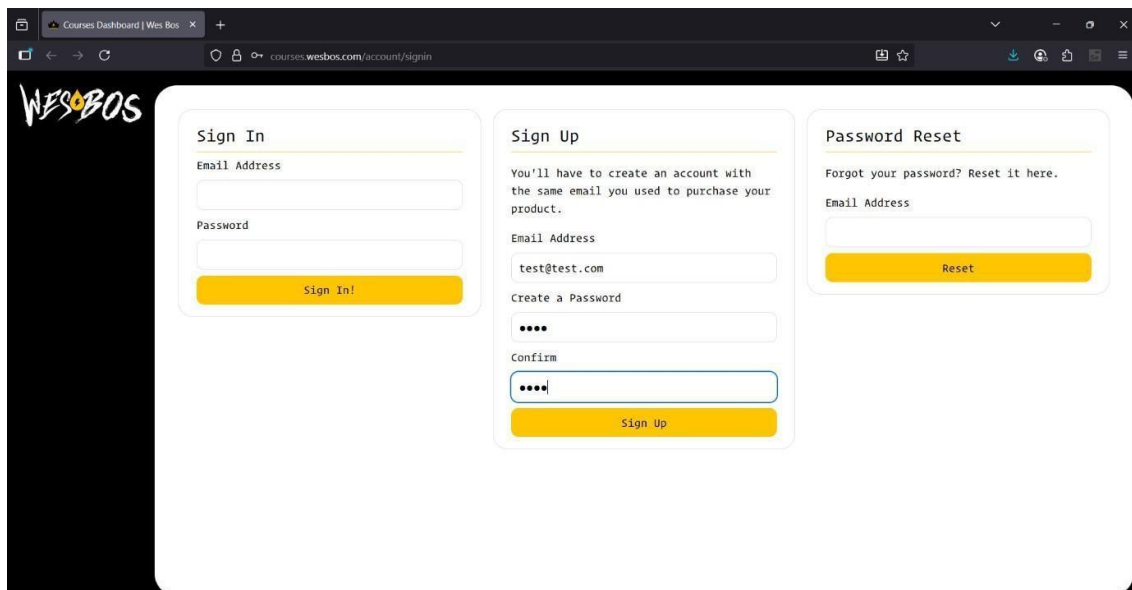


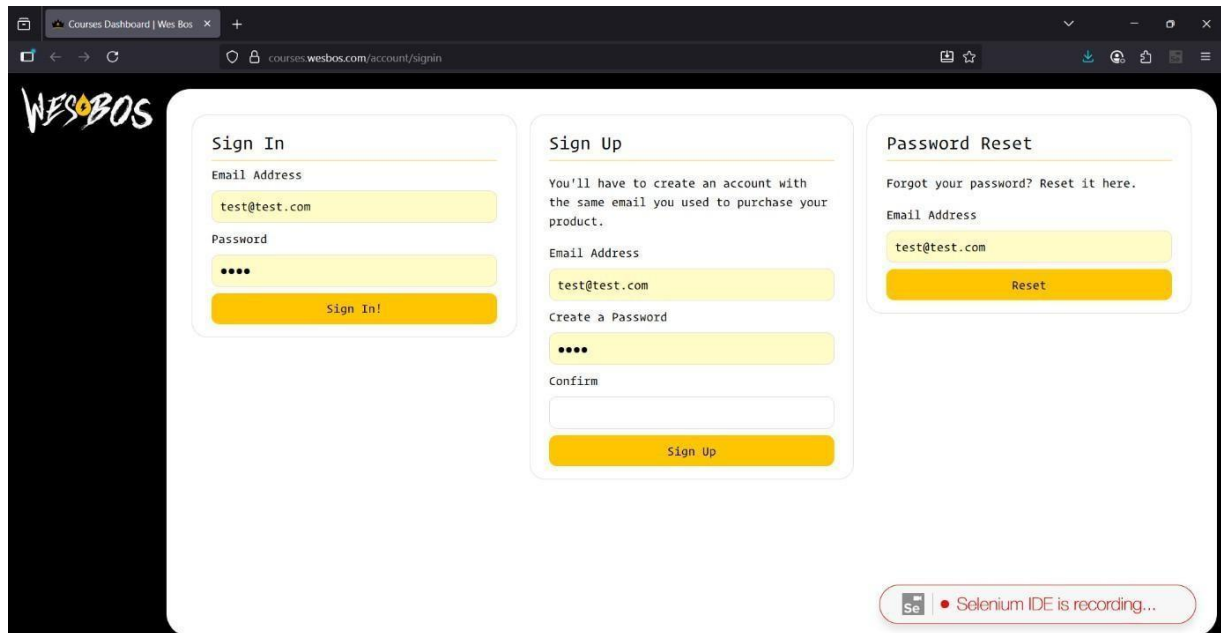
Step 3 : Add the Project's base URL:

Enter a project's base URL and click "**Start Recording.**"

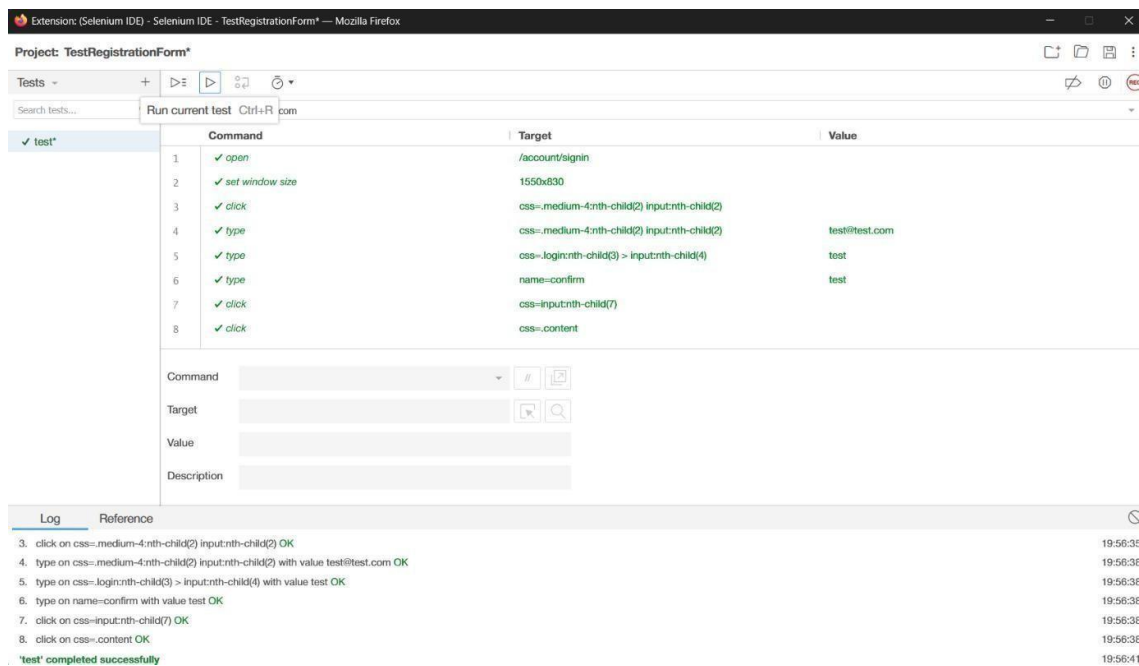


Step 4: It will start recording the steps by visiting the URL and start recording the inputs credentials for the registration form.



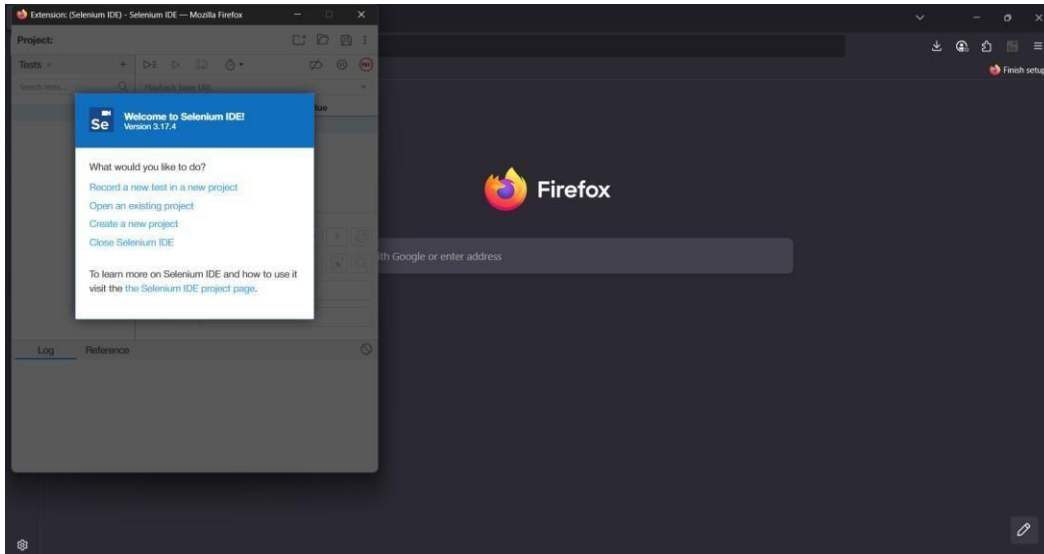


Step 5: After recording all the steps make sure to run the program with proper file name and it will start perform the testing on that site.



Q5. Record and run test cases on any website to check the validations of elements. (eg. IRCTC , MSRTC , etc)

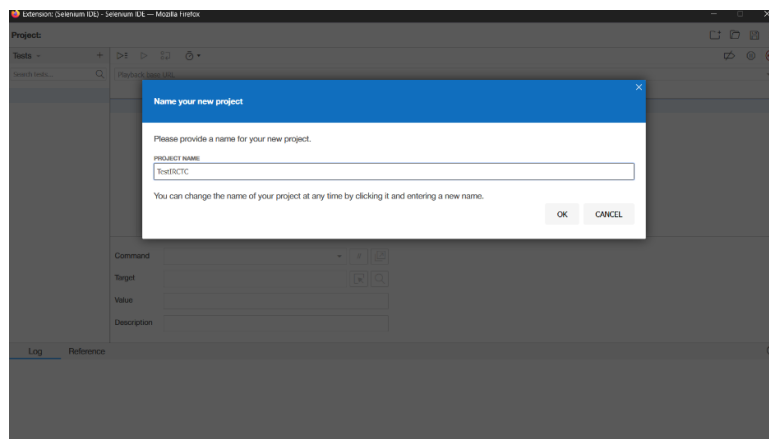
Step 1 : Open Firefox and Launch Selenium IDE, Click the Selenium IDE icon in the Firefox toolbar to open it.



Step 2 : Create a New Project:

In Selenium IDE, click "**Create a new project.**"

Enter a project name (e.g., "TestIRCTC") and click "**OK.**"



Step 3 : Add the Project's base URL:Enter a project's base URL and click "**Start Recording.**"

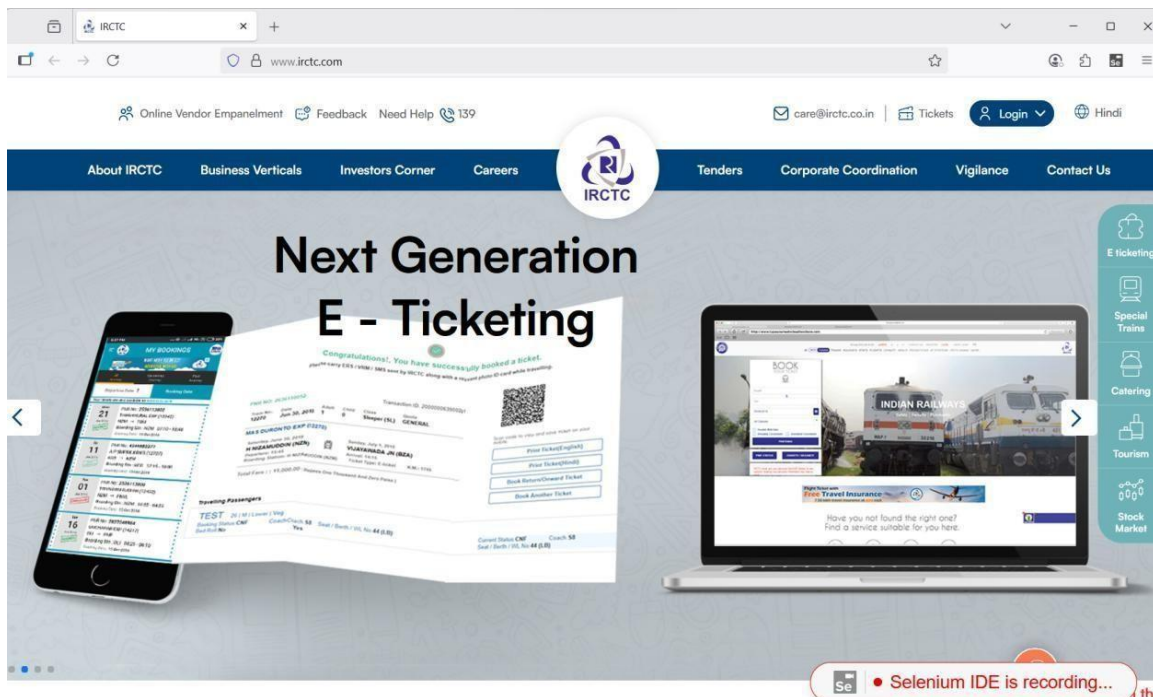
Set your project's base URL

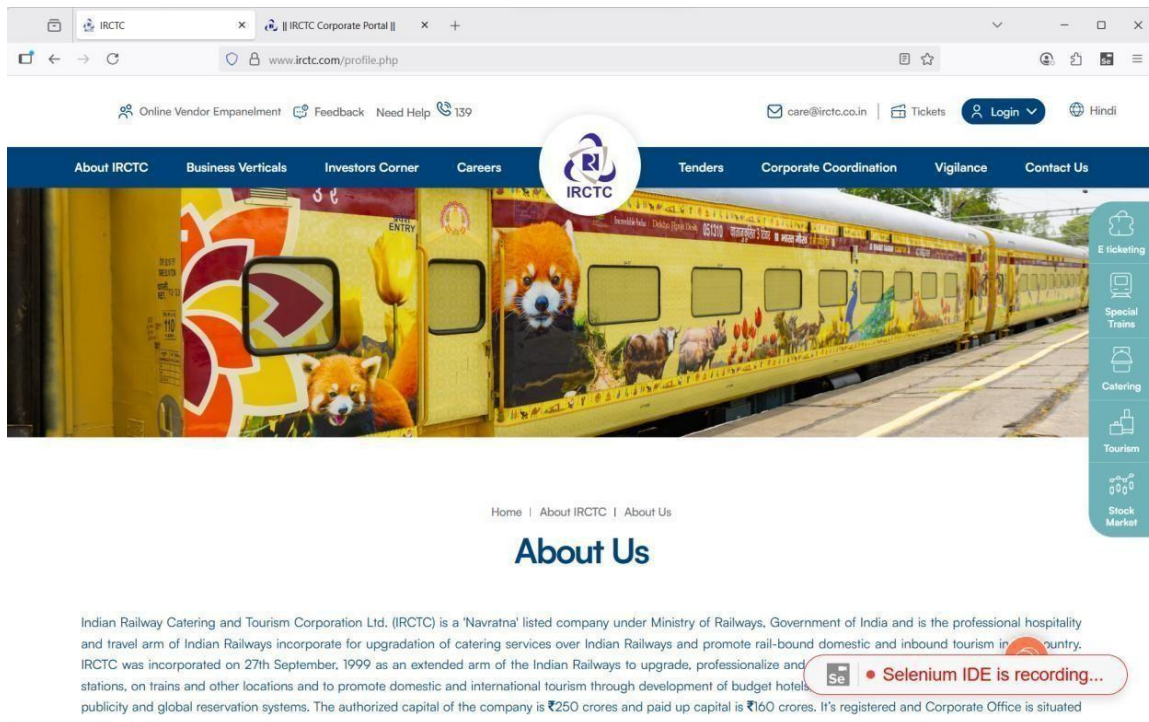
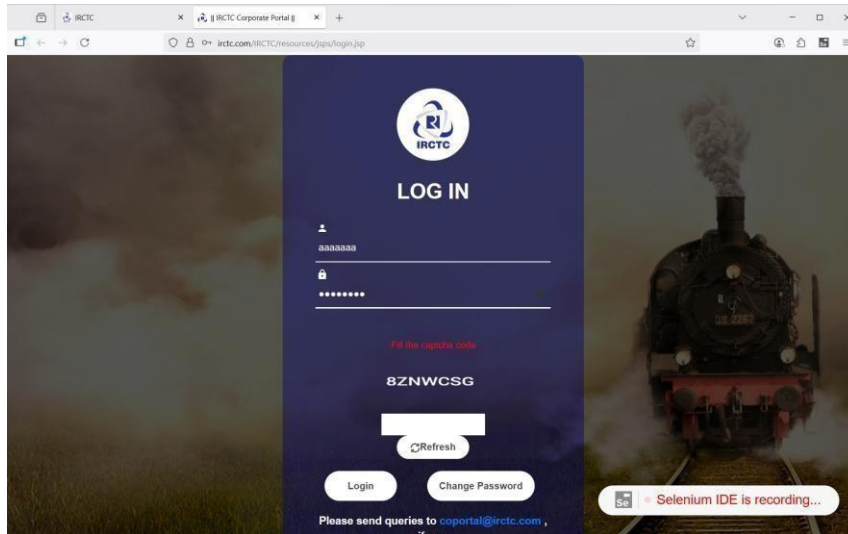
Before you can start recording, you must specify a valid base URL for your project. Your tests will start by navigating to this URL.

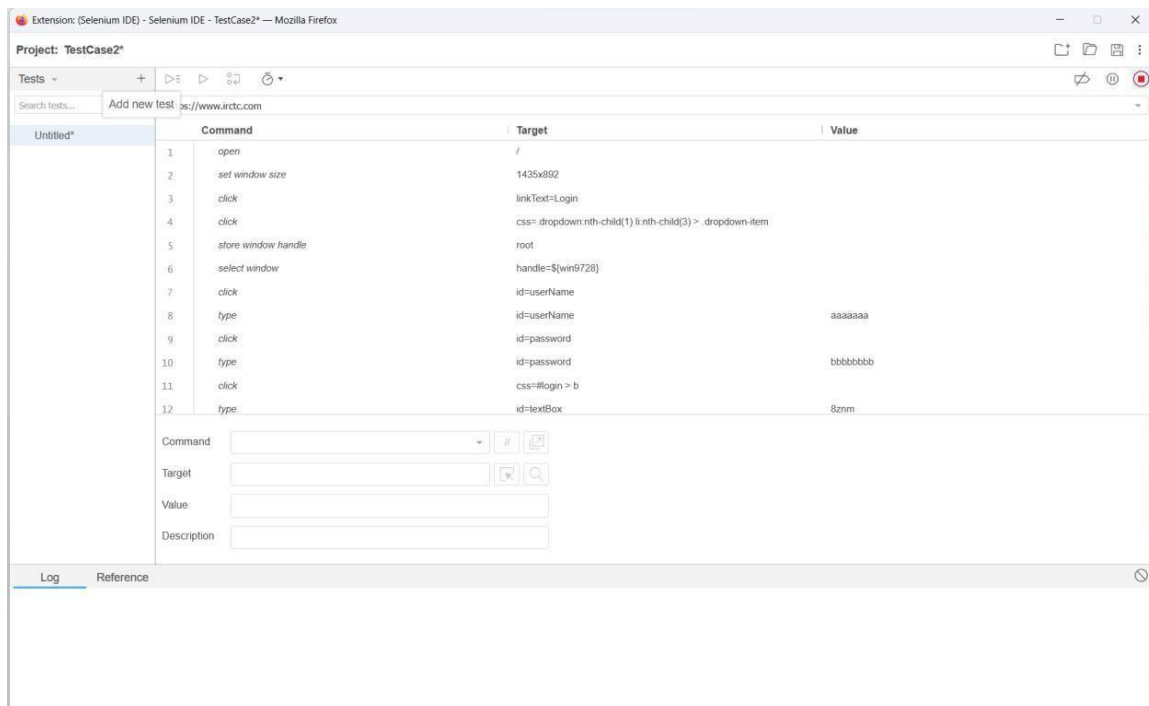
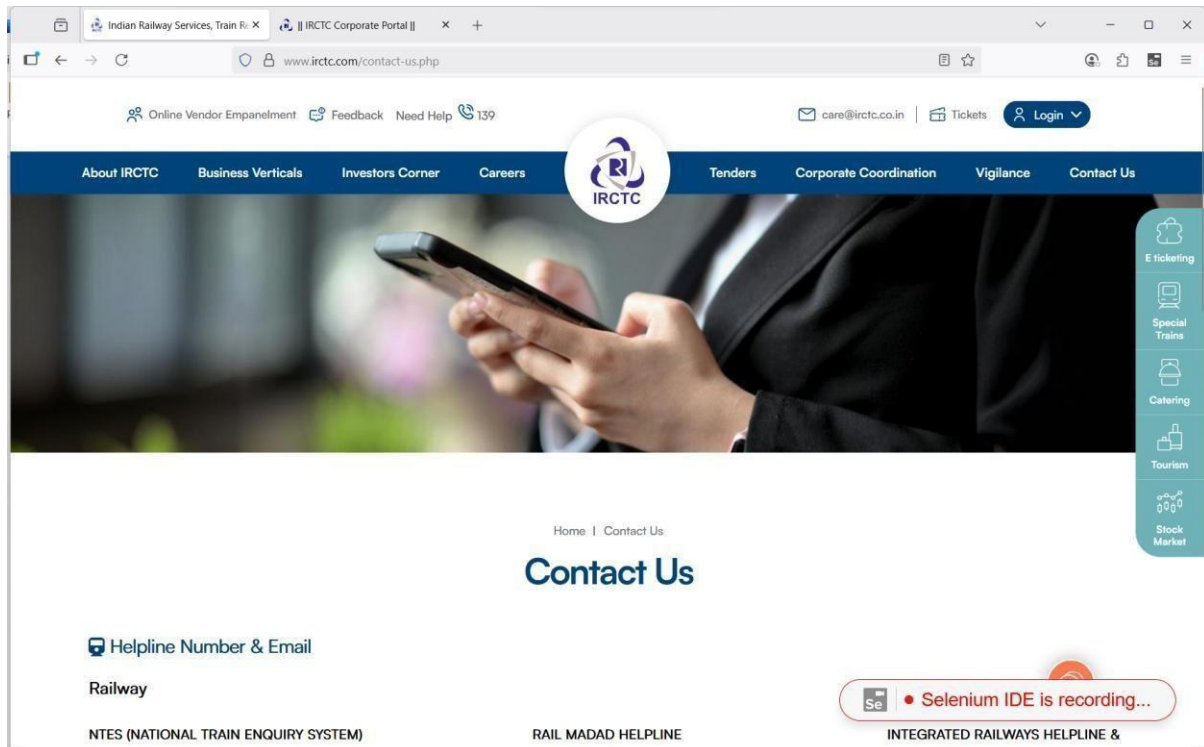
BASE URL

START RECORDING

CANCEL







Practical No : 5

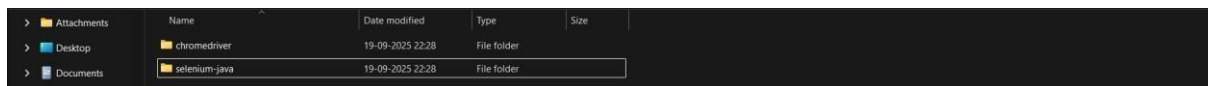
Implement **Web Drivers** on Chrome & Firefox Browsers.

Q1. Implement Selenium WebDriver on Chrome Browser

Step 1 : Download ChromeDriver

Go to the official ChromeDriver download site and download the version compatible with your Chrome browser version.

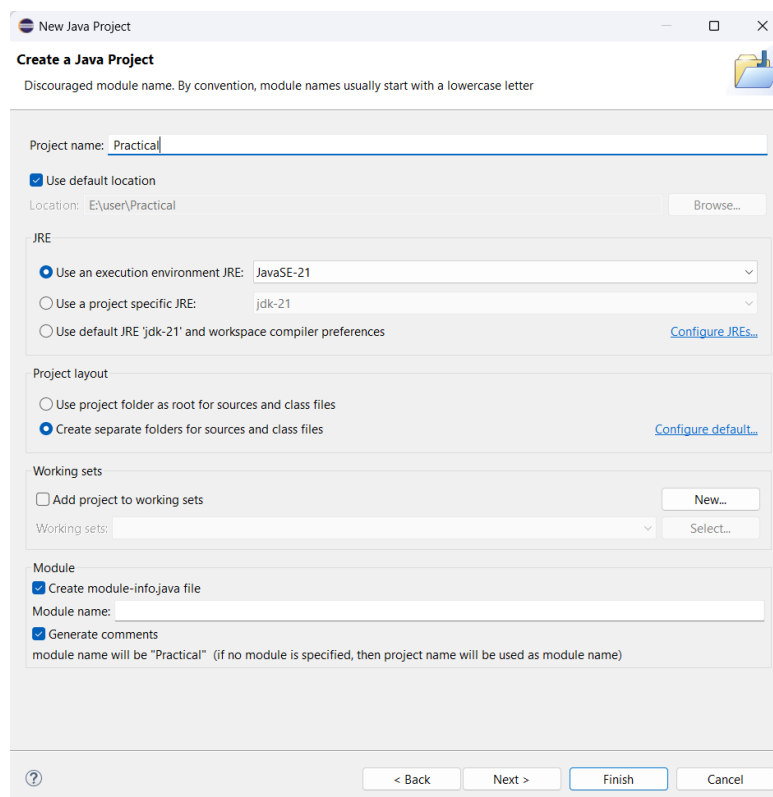
Extract the executable to a folder on your machine.

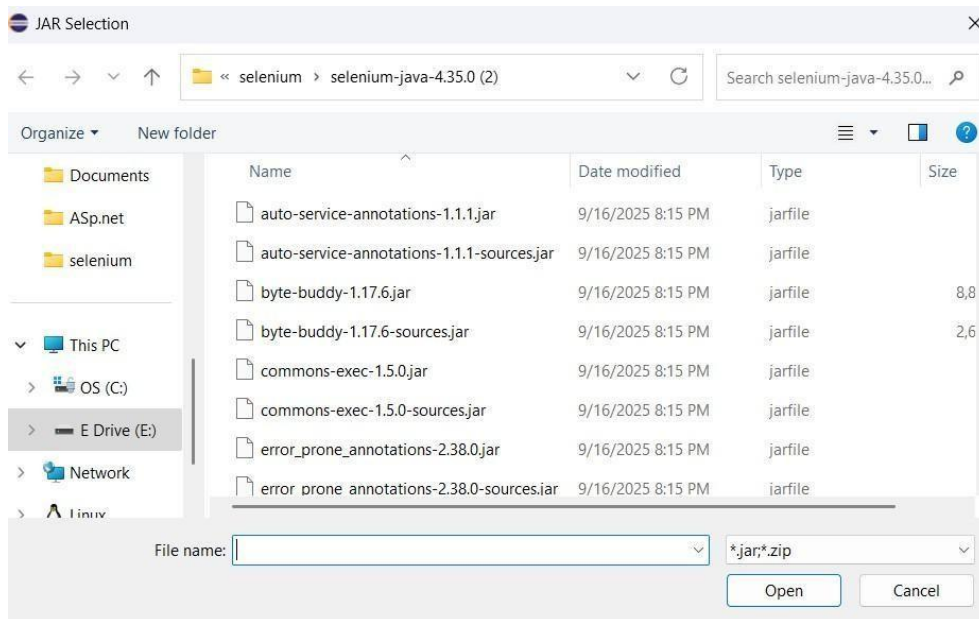


Step 2: Set up Selenium and ChromeDriver in your project: Add

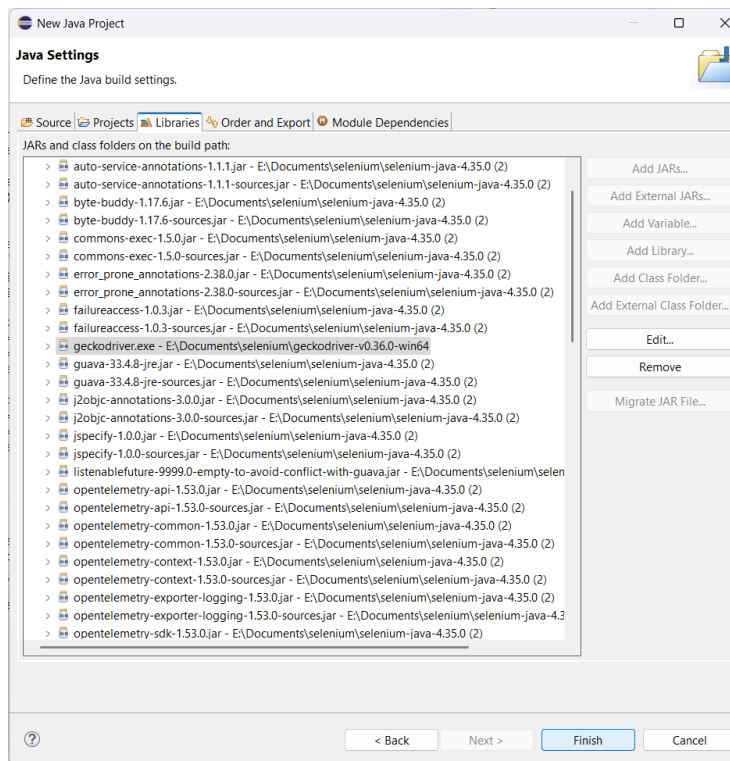
Selenium WebDriver library dependencies to your project,

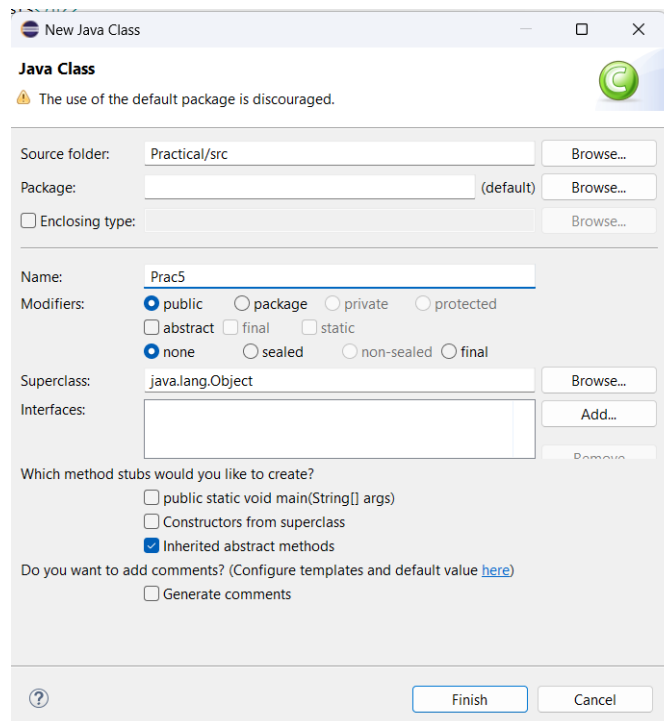
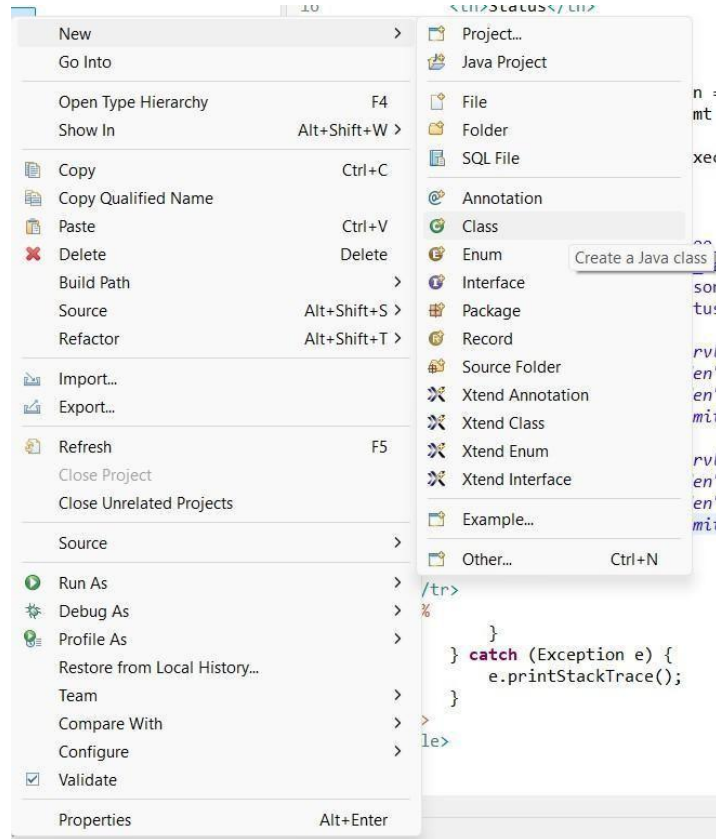
Set the system property in your code to point to the ChromeDriver executable path, inside new java project





Step 3: Add these libraries under the new project inside java settings.



Step 4: Create the new class instance

```

1 import org.openqa.selenium.WebDriver;
2 import org.openqa.selenium.chrome.ChromeDriver;
3
4 public class Prac5 {
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         System.setProperty("webdriver.gecko.driver", "E://Documents//selenium//geckodriver-v0.36.0-win64/geckodriver.exe");
8         WebDriver driver = new ChromeDriver();
9         driver.get("https://www.google.com/");
10    }
11
12 }

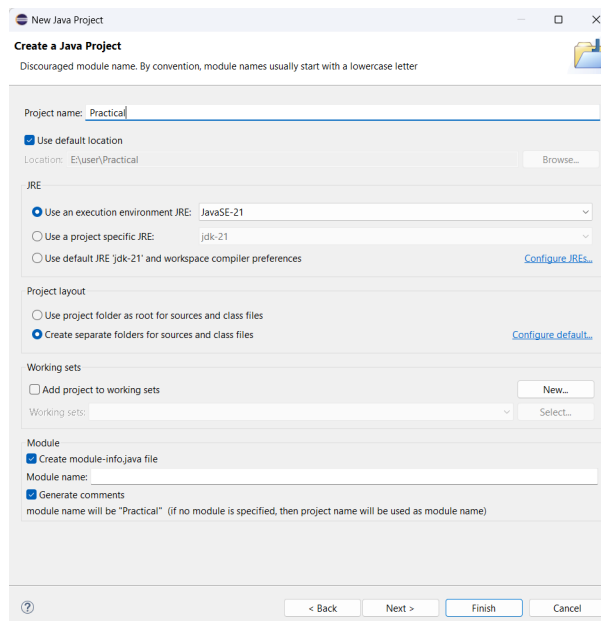
```

Q2. Implement Selenium WebDriver on Firefox Browser

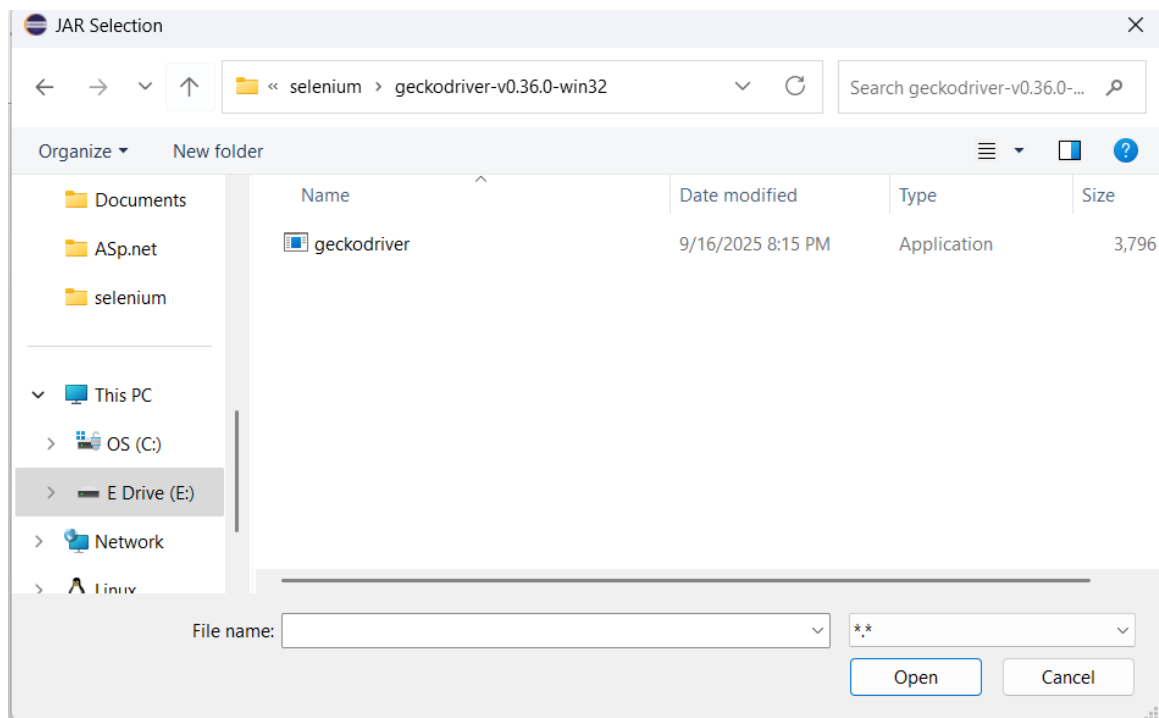
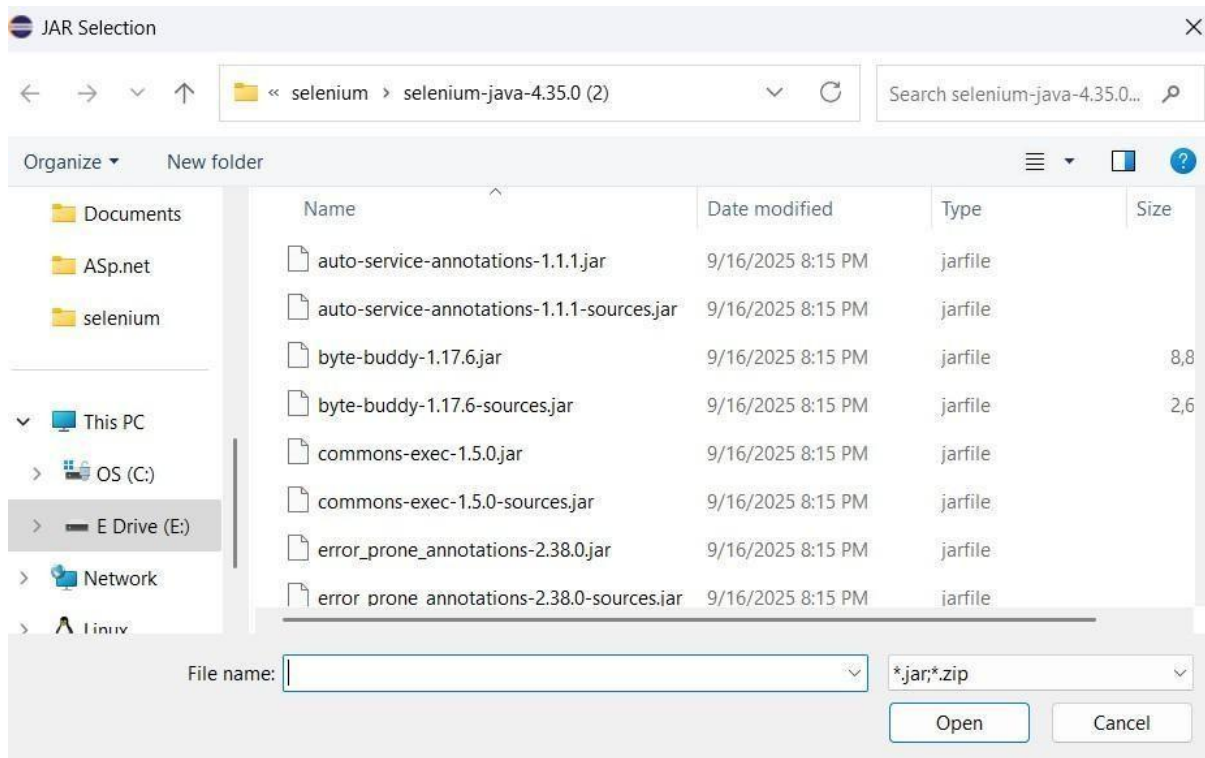
Step 1 : Download GeckoDriver:

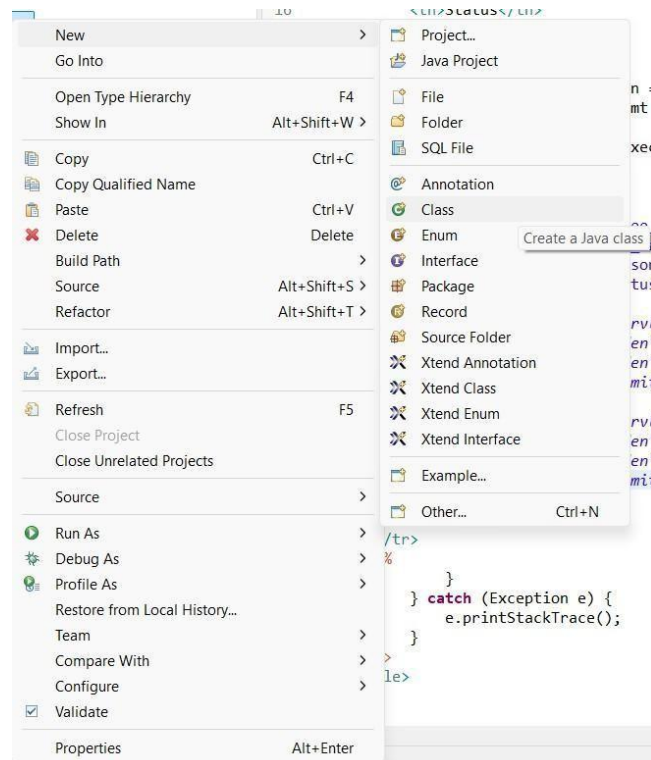
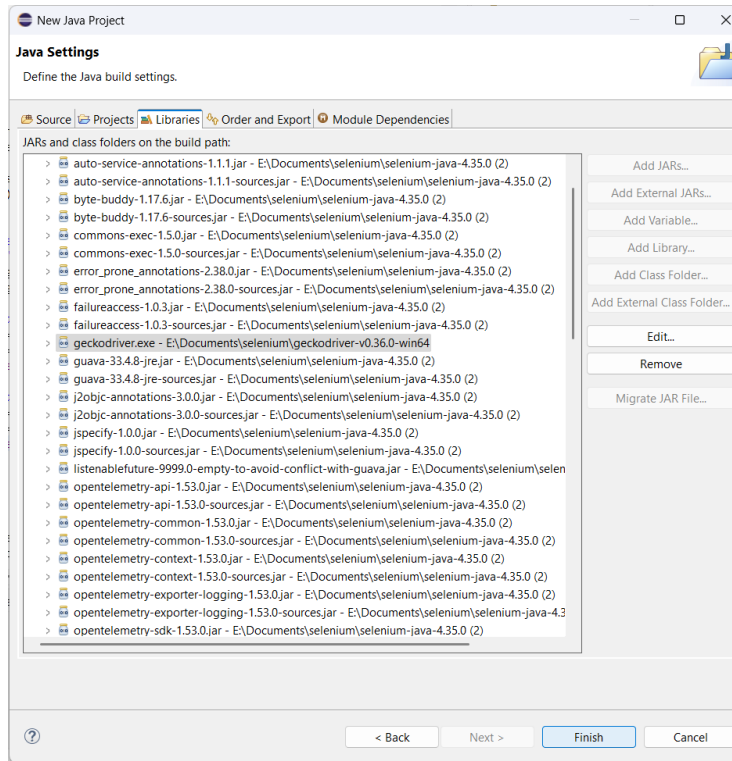
- Download GeckoDriver (FirefoxDriver) from its official GitHub release page.
- Extract and place the executable in a folder, add it to system PATH.

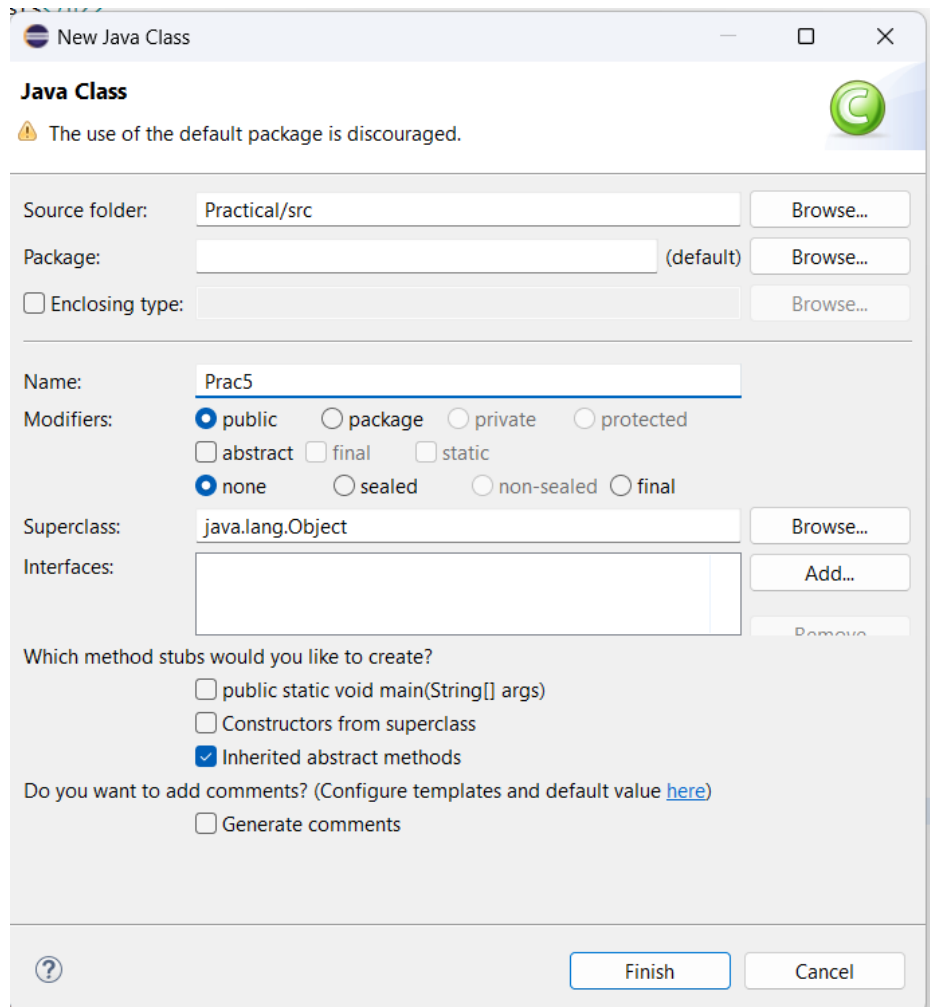
Step 2: Create a Java Project inside the Eclipse IDE.



geckodriver-v0.36.0-win32	9/16/2025 8:15 PM	File folder
selenium-java-4.35.0 (2)	9/16/2025 8:15 PM	File folder







Code parc1:

```
package nnnn;
```

```
import org.openqa.selenium.By;
```

```
import org.openqa.selenium.WebDriver;
```

```
import org.openqa.selenium.WebElement;
```

```
import org.openqa.selenium.firefox.FirefoxDriver;
```

```
public class nnrr {
```

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



```
// Set GeckoDriver path (notice lowercase "webdriver")
System.setProperty("webdriver.gecko.driver",
    "E://Documents//selenium//geckodriver-v0.36.0-win64//geckodriver.exe");

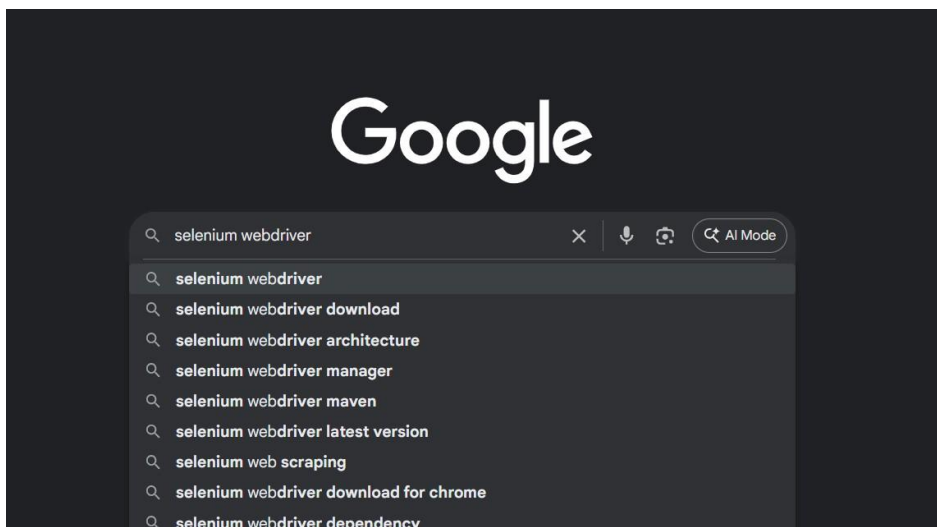
// Start Firefox
WebDriver driver = new FirefoxDriver();

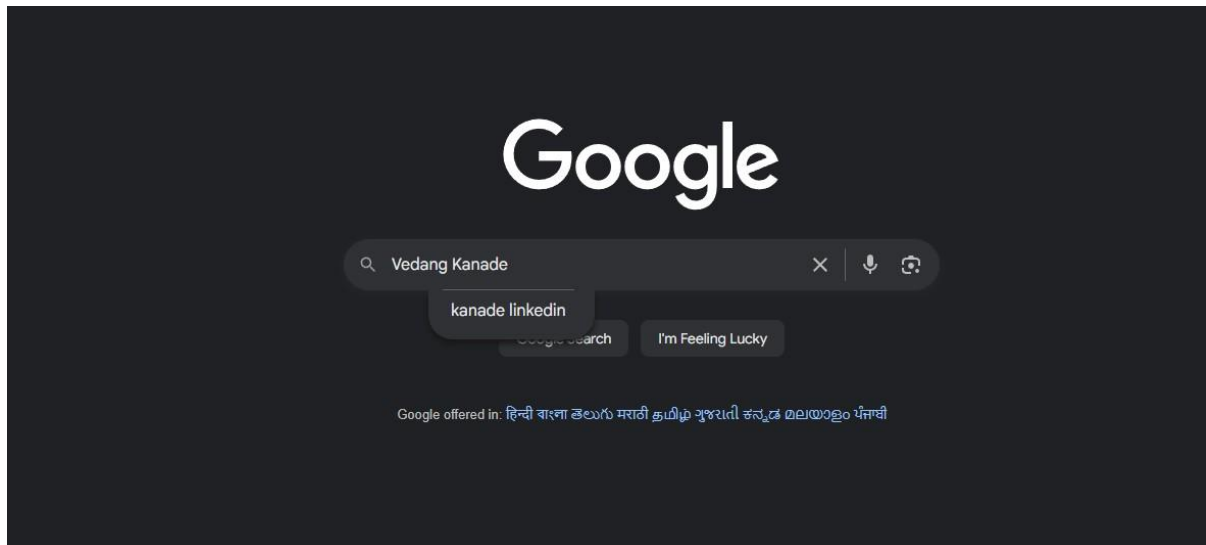
// Open Google
driver.get("https://www.google.com");

// Find the search box by 'name' attribute
WebElement searchBox = driver.findElement(By.name("q"));

// Type text into search box
searchBox.sendKeys("Vedang Kanade");
}
}
```

Output :





Practical No : 6

Q. Implement the Find element command through different Locators (id,Name)

Code :

```
package newpackage;

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

public class MyClass {

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "D:\\\\NMITD-College\\\\SEM-3\\\\STQA-
Lab\\\\Setups\\\\chromedriver\\\\chromedriver.exe");

        WebDriver driver = new ChromeDriver();

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

        // Find email input and enter text
        WebElement un1 = driver.findElement(By.id("email"));
        un1.sendKeys("swap@gmail.com");

        // Find password input and enter text
        WebElement un2 = driver.findElement(By.name("pass"));
        un2.sendKeys("123");

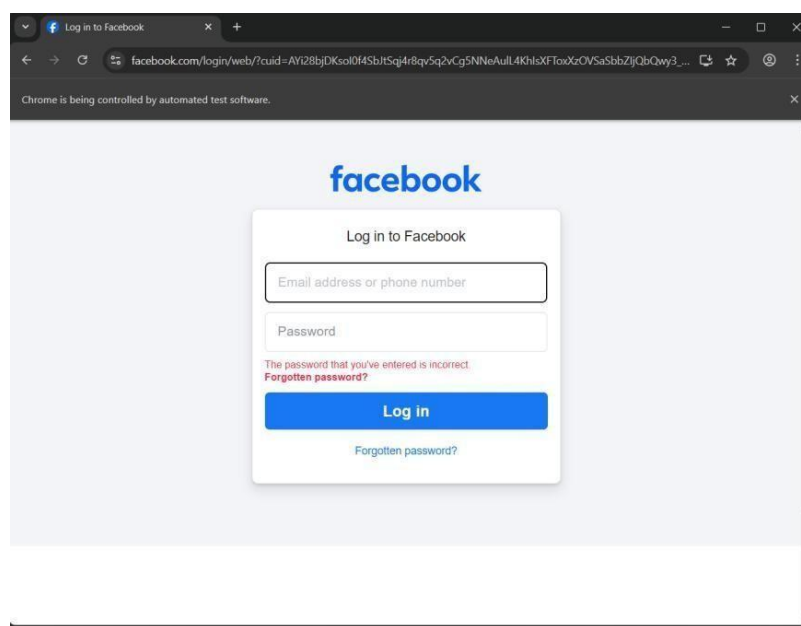
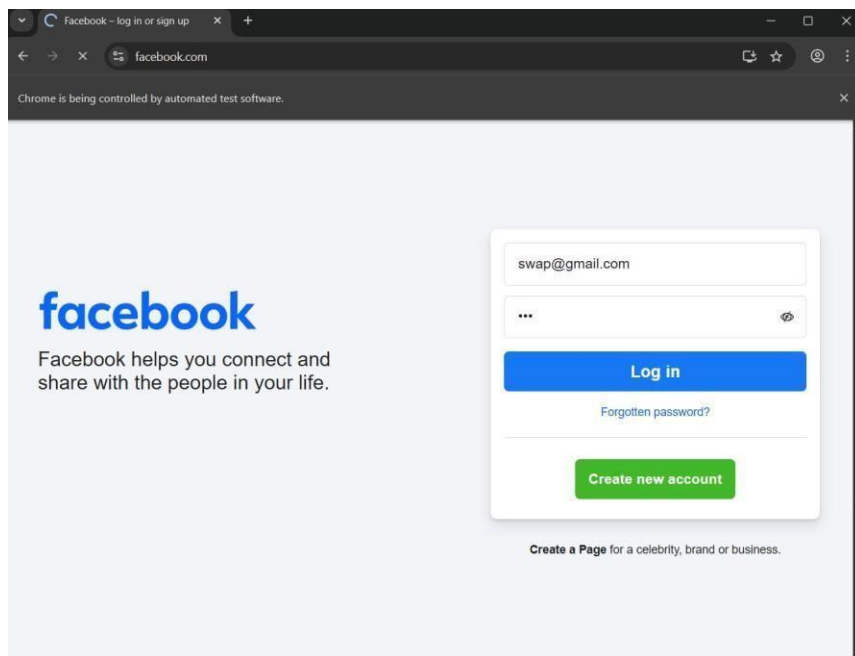
        // Find and click the login button
        WebElement button = driver.findElement(By.name("login"));
```

```
button.click();
```

```
}
```

```
}
```

Output :



Practical No : 7

Q. Implement Browser command and navigation Commands.

Code :

```
package newpackage;

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

public class BrowserNavigationDemo {
    public static void main(String[] args) {
        // Set up Chrome driver
        System.setProperty("webdriver.chrome.driver", "D:\\NMITD-
College\\SEM-3\\STQA-Lab\\Setups\\chromedriver\\chromedriver.exe");

        // Optional: set Chrome binary path if needed
        ChromeOptions options = new ChromeOptions();
        // Uncomment if Chrome is installed in a non-default location
        // options.setBinary("C:\\Path\\To\\Your\\Chrome\\chrome.exe");

        WebDriver driver = new ChromeDriver(options);

        // Maximize the browser window
        driver.manage().window().maximize();

        try {
```

```
// Open Google
driver.get("https://www.google.com");
System.out.println("Opened Google homepage");

// Navigate to another website, for example, OpenAI
driver.navigate().to("https://www.openai.com");
System.out.println("Navigated to OpenAI");

// Go back to Google
driver.navigate().back();
System.out.println("Navigated back to Google");

// Go forward to OpenAI
driver.navigate().forward();
System.out.println("Navigated forward to OpenAI");

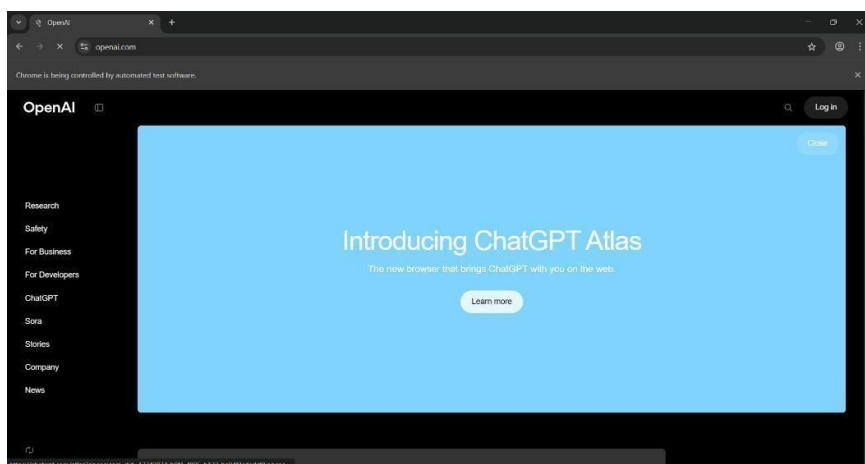
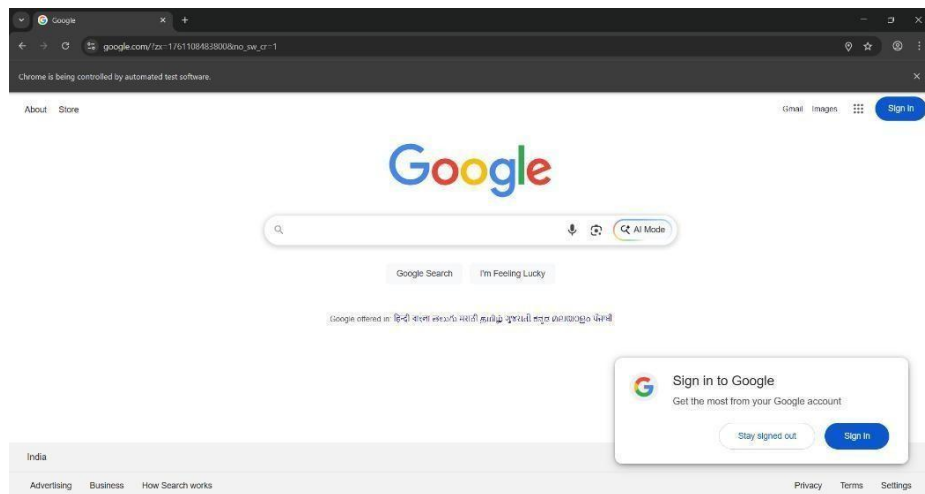
// Refresh the page
driver.navigate().refresh();
System.out.println("Page refreshed");

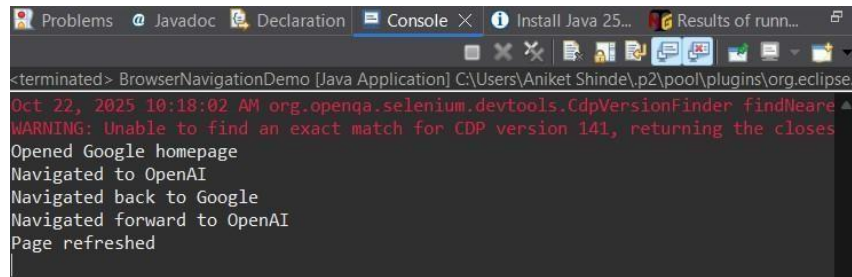
} catch (Exception e) {
    e.printStackTrace();
} finally {
    // Close the browser
    driver.quit();
}
```

}

}

Ouput :





The screenshot shows the Eclipse IDE's console window. The title bar includes tabs for 'Problems', 'Javadoc', 'Declaration', 'Console', 'Install Java 25...', and 'Results of runn...'. The console output is as follows:

```
<terminated> BrowserNavigationDemo [Java Application] C:\Users\Aniket Shinde\p2\pool\plugins\org.eclipse.  
Oct 22, 2025 10:18:02 AM org.openqa.selenium.devtools.CdpVersionFinder findNeare  
WARNING: Unable to find an exact match for CDP version 141, returning the closes  
Opened Google homepage  
Navigated to OpenAI  
Navigated back to Google  
Navigated forward to OpenAI  
Page refreshed
```


Practical No : 8

Q. Demonstrate handling multiple frames in selenium

Code :

```
package newpackage;

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.chrome.ChromeOptions;

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

        System.setProperty("webdriver.chrome.driver", "D:\\NMITD-
College\\SEM-3\\STQA-Lab\\Setups\\chromedriver\\chromedriver.exe");

        // Optional: ChromeOptions for custom configurations
        ChromeOptions options = new ChromeOptions();
        // Uncomment if needed

        WebDriver driver = new ChromeDriver(options);

        try {
// Open the specified URL

        driver.get("http://demo.guru99.com/test/guru99home/");
```

```
System.out.println("Opened target URL");

// Maximize the browser window
driver.manage().window().maximize();

// Wait for 2 seconds
Thread.sleep(2000);

driver.switchTo().frame("a077aa5e");
System.out.println("Switched to the iframe with ID 'a077aa5e'");

Thread.sleep(2000); // Wait before interacting with element

WebElement iframeElement =
driver.findElement(By.xpath("/html/body/a/img"));
iframeElement.click(); // Perform the click action
System.out.println("*We are done*");

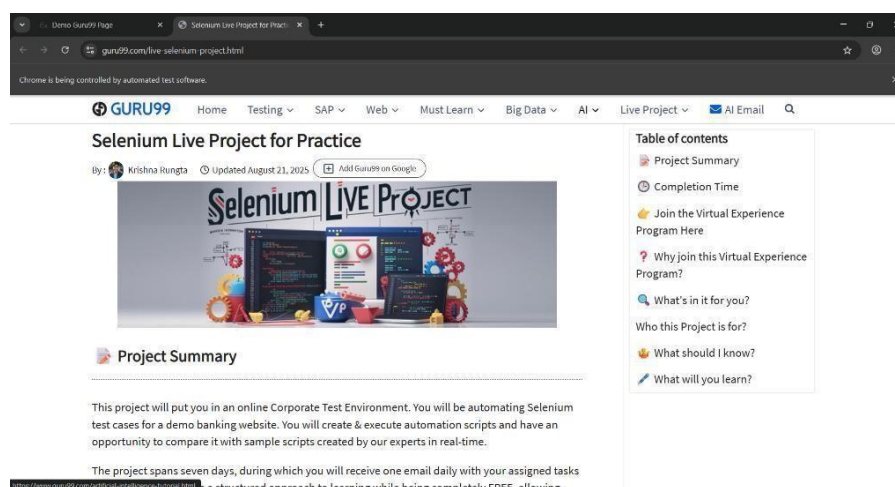
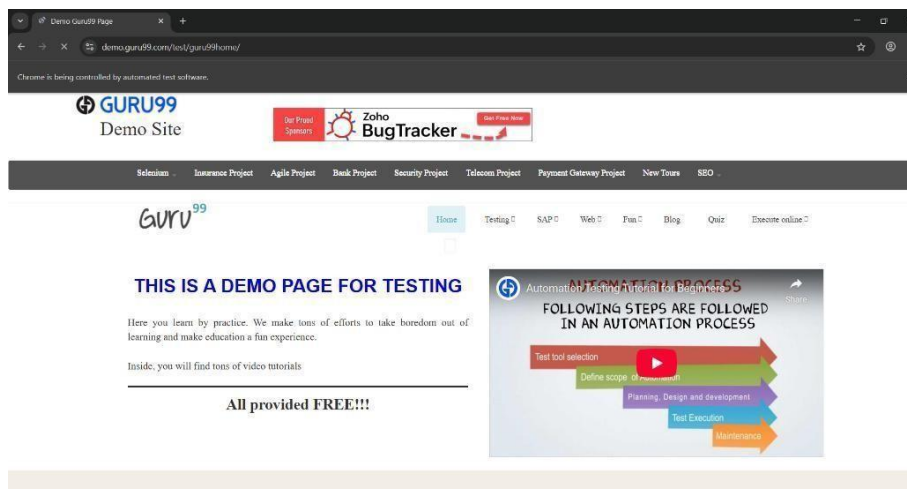
Thread.sleep(2000); // Wait to observe post-click action

// Optionally, switch back to the main page
driver.switchTo().defaultContent();

} catch (Exception e) {
    e.printStackTrace();
} finally {
    // Close the browser
```

```
driver.quit();  
  
}  
  
}}
```

Output :



```
WARNING: Unable to find an exact match for CD  
Opened target URL  
Switched to the iframe with ID 'a077aa5e'  
*We are done*
```

Practical No : 9

Q. Demonstrate Synchronisation in Selenium using wait command.

Code :

```
package newpackage;

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.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.FluentWait;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;
import java.util.NoSuchElementException;
import java.util.function.Function;

public class SynchronizationDemo {
    public static void main(String[] args) {
        // 1. Set path to ChromeDriver executable
        System.setProperty("webdriver.chrome.driver",
            "D:\\NMITD-College\\SEM-3\\STQA-Lab\\Setups\\chromedriver\\chromedriver.exe");

        // 2. Initialize ChromeDriver
        WebDriver driver = new ChromeDriver();

        try {
            // 3. Navigate to a page with dynamic content
            driver.get("https://the-internet.herokuapp.com/dynamic_loading/1");

            // 4. Implicit Wait: applies to all findElement calls
            driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

            // 5. Start the dynamic load
            driver.findElement(By.cssSelector("#start button")).click();

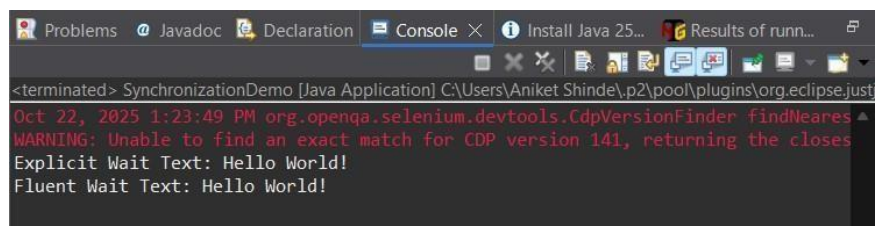
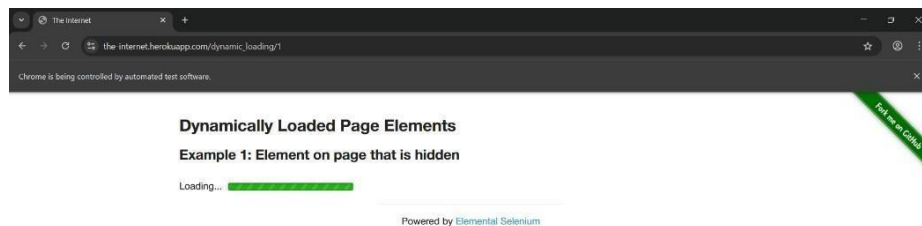
            // 6. Explicit Wait: wait until the loading spinner disappears and the element is visible
            WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(15));
            WebElement loadedText = wait.until(
                ExpectedConditions.visibilityOfElementLocated(By.cssSelector("#finish h4"))
            );
            System.out.println("Explicit Wait Text: " + loadedText.getText());
        }
    }
}
```

```
// 7. Fluent Wait: check every 2 seconds up to 20 seconds for the same element
FluentWait<WebDriver> fluentWait = new FluentWait<>(driver)
    .withTimeout(Duration.ofSeconds(20))
    .pollingEvery(Duration.ofSeconds(2))
    .ignoring(NoSuchElementException.class);

WebElement fluentElement = fluentWait.until(new Function<WebDriver,
WebElement>() {
    public WebElement apply(WebDriver drv) {
        return drv.findElement(By.cssSelector("#finish h4"));
    }
});

System.out.println("Fluent Wait Text: " + fluentElement.getText());

} finally {
    // 8. Quit the driver
    driver.quit();
}
}
```

Output:

Practical No : 10

Q. Demonstrate different types of alerts.

Code :

```
package newpackage;

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

public class AlertDemo {

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

        // Set path to ChromeDriver

        System.setProperty("webdriver.chrome.driver", "D:\\NMITD-College\\SEM-3\\STQA-Lab\\Setups\\chromedriver\\chromedriver.exe");

        // Launch Chrome browser

        WebDriver driver = new ChromeDriver();

        // Open the webpage

        driver.get("https://demo.guru99.com/test/delete_customer.php");

        // Enter Customer ID

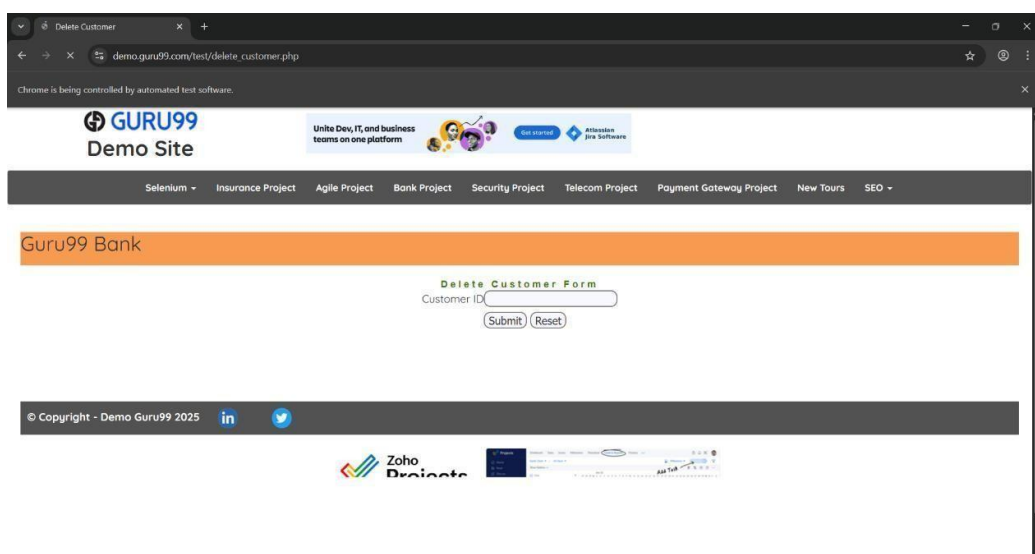
        driver.findElement(By.name("cusid")).sendKeys("30");

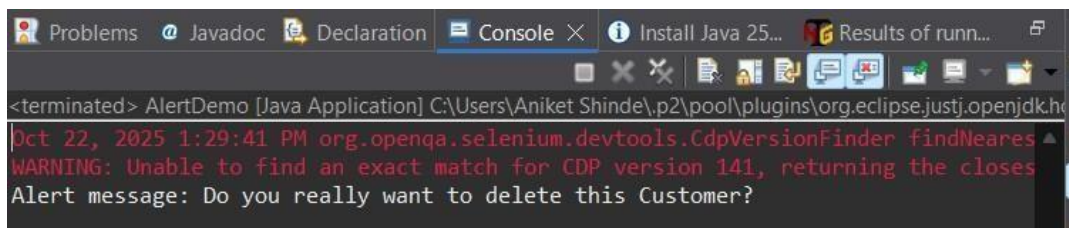
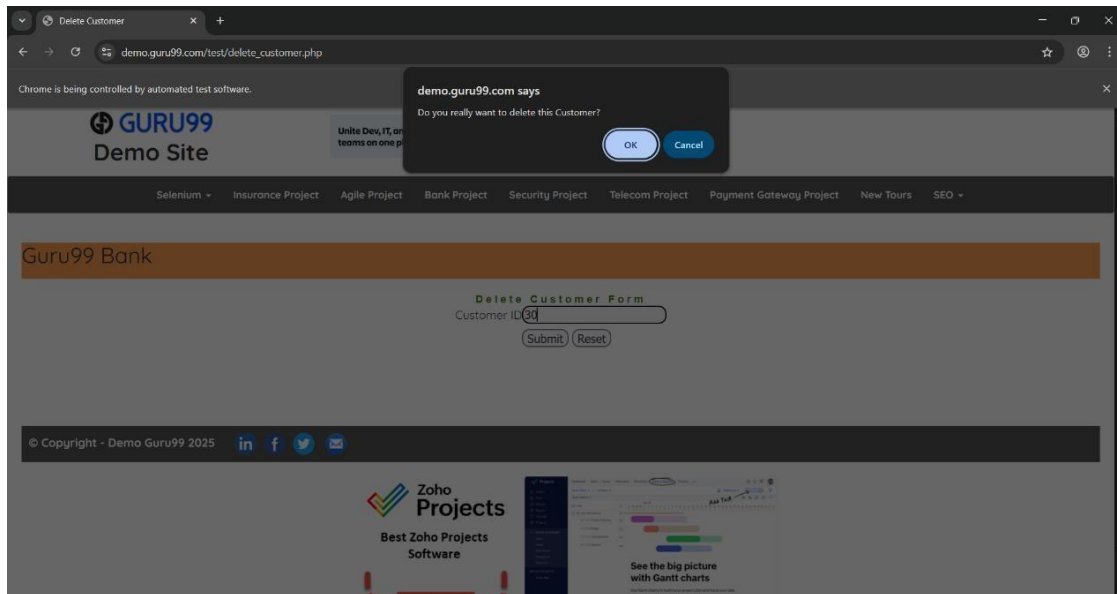
        // Click Submit

        driver.findElement(By.name("submit")).click();
```

```
// Switch to alert  
Alert alt = driver.switchTo().alert();  
  
// Capture alert message  
String alertMessage = alt.getText();  
System.out.println("Alert message: " + alertMessage);  
  
// Wait for 5 seconds  
Thread.sleep(5000);  
  
// Accept the alert (click OK)  
alt.accept();  
  
// Close the browser  
driver.quit();  
}  
}
```

Output:





Practical No : 11

Q. Demonstrate :

- Handling Drop Down,
- List Boxes
- Command Button,
- Radio buttons & text boxes.
- Waits command in selenium

Code :

```
package newpackage;
```

```
import org.openqa.selenium.*;
```

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

```
import org.openqa.selenium.support.ui.*;
```

```
import java.time.Duration;
```

```
public class Prac11Demo {
```

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

```
        System.setProperty("webdriver.chrome.driver",
```

```
            "D:\\NMITD-College\\SEM-3\\STQA-  
Lab\\Setups\\chromedriver\\chromedriver.exe");
```

```
        WebDriver driver = new ChromeDriver();
```

```
        driver.get("https://demo.guru99.com/test/newtours/register.php");
```

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

```
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

Thread.sleep(1000);

// Fill form
driver.findElement(By.name("firstName")).sendKeys("John");
Thread.sleep(500);
driver.findElement(By.name("lastName")).sendKeys("Doe");
Thread.sleep(500);
driver.findElement(By.name("phone")).sendKeys("1234567890");
Thread.sleep(500);
driver.findElement(By.name("userName")).sendKeys("john.doe@example.com");
Thread.sleep(500);

// Dropdown
WebElement countryDropdown = driver.findElement(By.name("country"));
Select selectCountry = new Select(countryDropdown);
selectCountry.selectByVisibleText("INDIA");
Thread.sleep(1000);

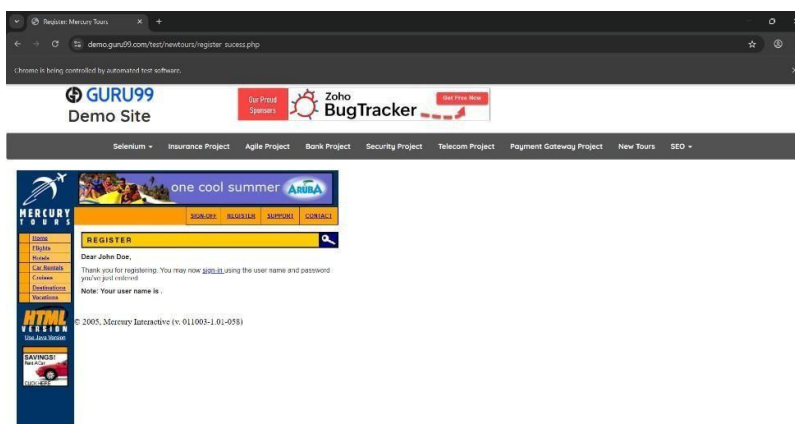
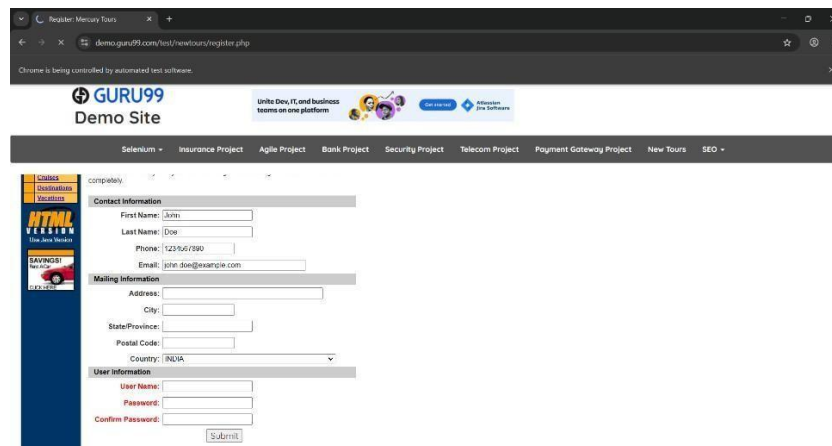
// Wait for overlay (if any) to disappear
try {
    wait.until(ExpectedConditions.invisibilityOfElementLocated(
        By.cssSelector("div.cb-box_wrapper-center_modal")));
} catch (Exception e) {
    System.out.println("No modal overlay detected. Continuing...");
}
```

```
// Click submit using JavaScript (to avoid interception)
WebElement submitButton = driver.findElement(By.name("submit"));
((JavascriptExecutor) driver).executeScript("arguments[0].click();", submitButton);

wait.until(ExpectedConditions.titleContains("Register"));

Thread.sleep(2000);

driver.quit();
}
}
```

Output:

Practical No : 12

Q. Demonstrate Action Class In Selenium

Code :

```
package newpackage;

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 ActionClassDemo {

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver",
            "D:\\NMITD-College\\SEM-3\\STQA-Lab\\Setups\\chromedriver\\chromedriver.exe");

        WebDriver driver = new ChromeDriver();

        try {

            driver.get("https://demoqa.com/buttons");

            Thread.sleep(1000);

            Actions actions = new Actions(driver);

            // Double click button

            WebElement doubleClickBtn = driver.findElement(By.id("doubleClickBtn"));

            actions.doubleClick(doubleClickBtn).perform();
```

```
System.out.println("Double click performed");

Thread.sleep(1000);


// Right click button
WebElement rightClickBtn = driver.findElement(By.id("rightClickBtn"));
actions.contextClick(rightClickBtn).perform();

System.out.println("Right click performed");

Thread.sleep(1000);


// Click button (normal click)
WebElement clickMeBtn = driver.findElement(By.xpath("//button[text()='Click Me']"));
actions.click(clickMeBtn).perform();

System.out.println("Click performed");

Thread.sleep(1000);


// Navigate to drag and drop page for drag and drop example
driver.get("https://demoqa.com/droppable");

Thread.sleep(1000);


WebElement source = driver.findElement(By.id("draggable"));
WebElement target = driver.findElement(By.id("droppable"));
actions.dragAndDrop(source, target).perform();

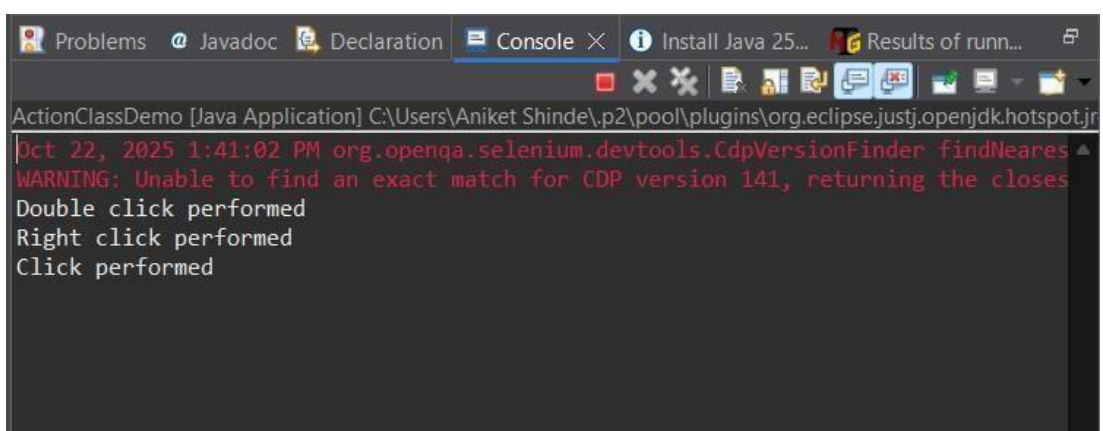
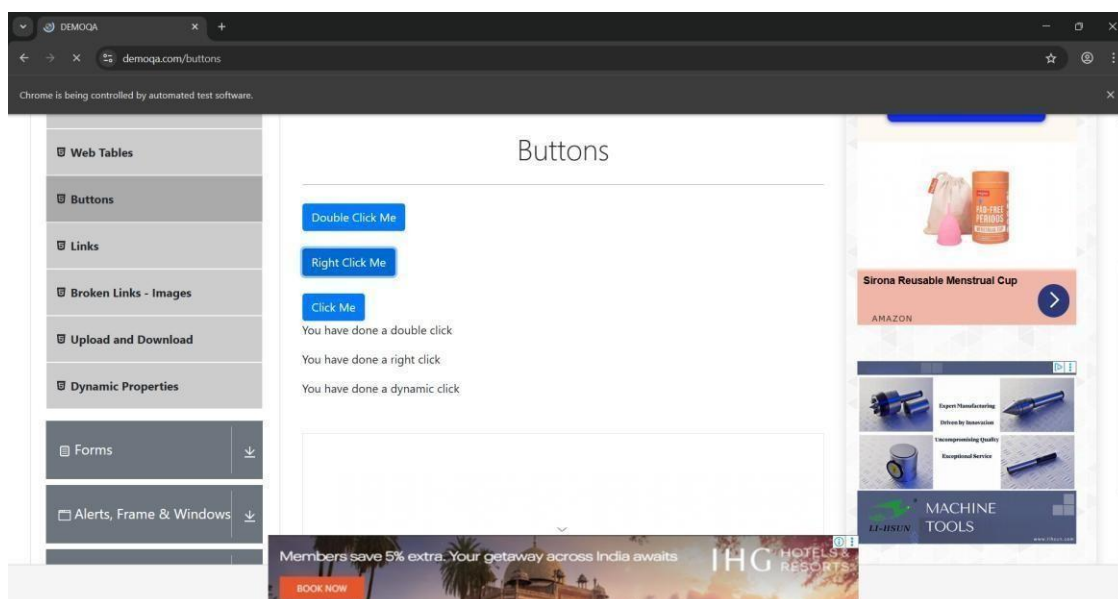
System.out.println("Drag and drop performed");

Thread.sleep(1000);


} catch (InterruptedException e) {
    e.printStackTrace();
} finally {
```

```
        driver.quit();  
    }  
}  
}
```

Output:



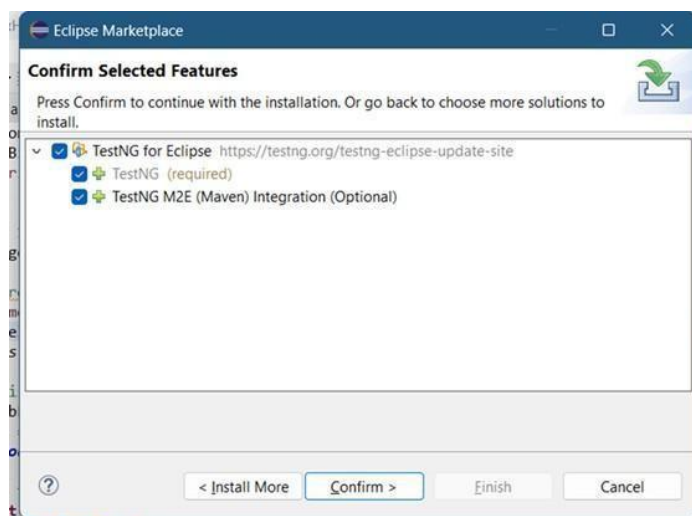
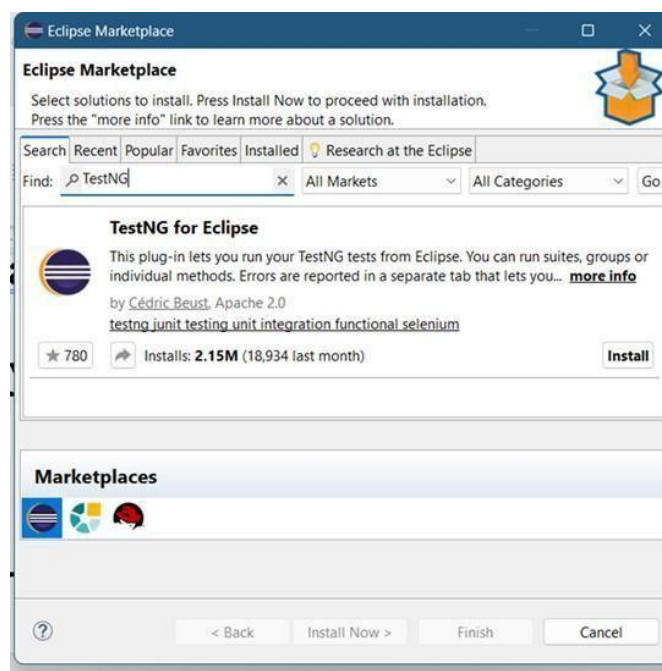
Practical No : 13

Q. Installation of TestNg , running testNg and TestNg annotations.

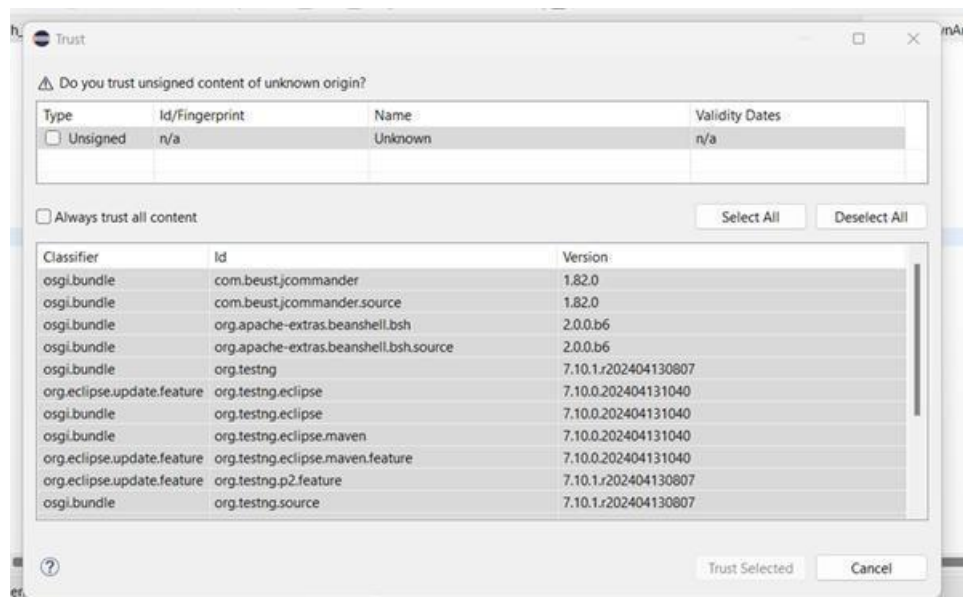
Code :

The following are the steps to install TestNG framework

Step 1: Search for TestNG in Eclipse marketplace and download it.



Step 2: If you encounter any security issue click on 'Trust all' and install it.



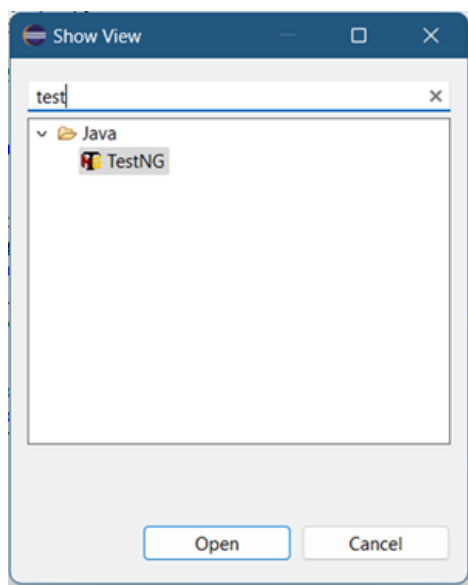
Step 3: Finish the installation and restart your system

Wait for the TestNG install in Eclipse to finish. When Eclipse prompts you for a restart, click

“Restart now.”

Verify if the installation is done properly

After the restart, verify if TestNG for Eclipse was successfully installed. Click Window > Show View > Other.



The following are the steps to Run TestNG Annotation.

Step 1: First download the two external JAR files and add them in your build path under the classpath section. The JAR files are slf4j-api.jar and slf4j-simple.jar from <http://www.java2s.com/>. Then Add those JARs in Configure Built-Path > Under the Classpath.

Step 2: Write the code for TestNG1:

Code :

```
package newpackage;

import org.junit.Test;

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

public class TestNG1 {

    public String baseUrl = "https://www.google.com/";

    String driverPath = "D:\\\\NMITD-College\\\\SEM-3\\\\STQA-
Lab\\\\Setups\\\\chromedriver\\\\chromedriver.exe";

    public WebDriver driver;

    @Test

    public void f() throws InterruptedException {

        System.out.println("Launching Chrome browser");

        // Set ChromeDriver path

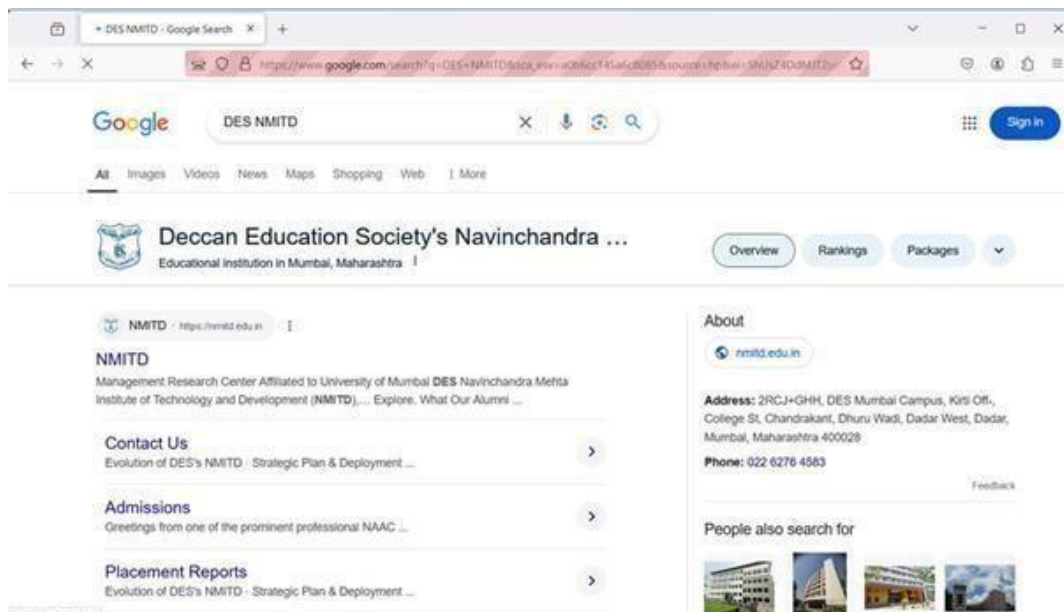
        System.setProperty("webdriver.chrome.driver", driverPath);
```

```
ChromeOptions options = new ChromeOptions();  
options.addArguments("--start-maximized"); // open full screen  
options.addArguments("--disable-notifications");  
driver = new ChromeDriver(options);  
// Open Google  
driver.get(baseUrl);  
// Perform a Google search  
driver.findElement(By.name("q")).sendKeys("DES NMITD", Keys.ENTER);  
// Wait for results to load  
Thread.sleep(2000);  
// Close browser  
driver.quit();  
}  
}
```

Output:

```
400 [main] INFO org.testng.internal.Utils - [Utils] dynamicDapHelper.createDynamicDap() took 4 ms.  
Launching Firefox browser  
PASSED: seleniumtest_firefox.TestNG1.f  
  
=====  
Default test  
Tests run: 1, Failures: 0, Skips: 0  
=====
```

```
=====  
Default suite  
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0  
=====
```



Step 3: Write the code for TestNG2 – Running TestNG:

Code :

```
package seleniumtest_chrome;

import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.testng.annotations.Test;

public class TestNG2 {

    String driverPath =
"D:\\\\NMITD-College\\\\SEM-3\\\\STQA-Lab\\\\Setups\\\\chromedriver\\\\chromedriver.exe"

    @Test

    public void TestGoogle() throws InterruptedException {
```

```
// Set ChromeDriver path
System.setProperty("webdriver.chrome.driver", driverPath);

// Chrome options
ChromeOptions options = new ChromeOptions();
options.addArguments("--start-maximized");
options.addArguments("--disable-notifications");

// Launch Chrome
WebDriver driver = new ChromeDriver(options);
driver.get("https://www.google.com/");

// Perform a search on Google
driver.findElement(By.name("q")).sendKeys("DES's NMITD", Keys.ENTER);
Thread.sleep(2000);

// Close browser
driver.quit();
}
```

@Test

```
public void TestFacebook() throws InterruptedException {
```

```
    // Set ChromeDriver path
    System.setProperty("webdriver.chrome.driver", driverPath);

    // Chrome options
    ChromeOptions options = new ChromeOptions();
    options.addArguments("--start-maximized");
    options.addArguments("--disable-notifications");
```

```
// Launch Chrome

WebDriver driver = new ChromeDriver(options);

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

// Enter text in Facebook email field

driver.findElement(By.name("email")).sendKeys("DES's NMITD", Keys.ENTER);

Thread.sleep(2000);

// Close browser

driver.quit();

}

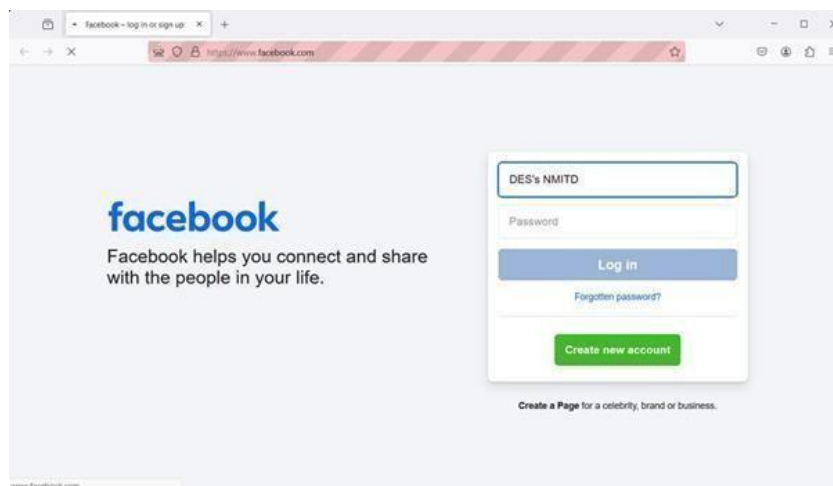
}
```

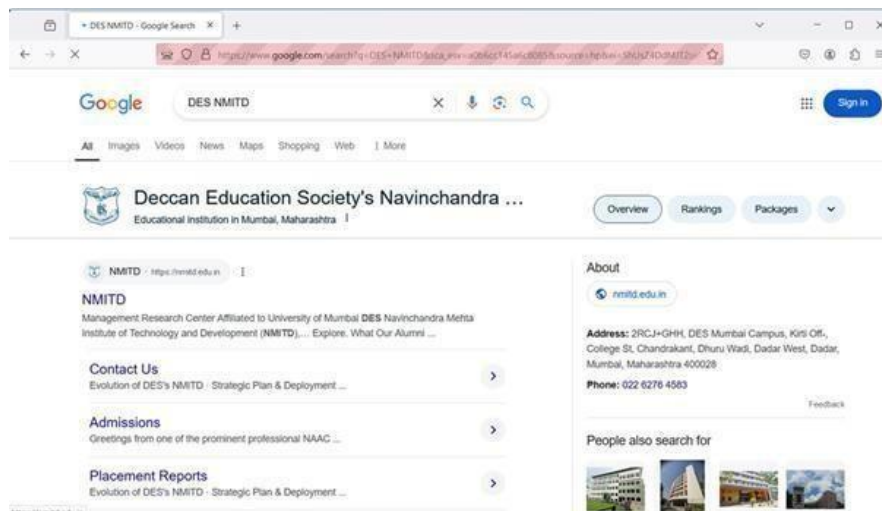
Output :

```
PASSED: seleniumtest_firefox.TestNG2.TestFacebook
PASSED: seleniumtest_firefox.TestNG2.TestGoogle

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

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





Q. Demonstration with TestNG3

Code :

```
public class TestNG3 {
```

```
    String actualTitle, expectedTitle;
```

```
    WebDriver driver;
```

```
    @BeforeTest
```

```
    public void launchBrowser() {
```

```
        System.out.println("Launching Chrome browser");
```

```
        System.setProperty("webdriver.chrome.driver",
```

```
"D:\\Anamay\\Projects\\2024\\java_selenium\\chrome_driver\\chromedriver.exe");
```

```
        // Set Chrome options
```

```
        ChromeOptions options = new ChromeOptions();
```

```
        options.addArguments("--start-maximized");
```

```
options.addArguments("--disable-notifications");

// Launch Chrome
driver = new ChromeDriver(options);

// Open test site
driver.get("https://demo.guru99.com/test/newtours/");
}

@Test
public void verifyHomepageTitle() {
    expectedTitle = "Welcome: Mercury Tours";
    actualTitle = driver.getTitle();
    Assert.assertEquals(actualTitle, expectedTitle);
}

@AfterTest
public void terminateBrowser() {
    driver.close();
}
}
```

Output :

```
Launching Firefox browser
8168 [main] INFO org.testng.internal.Utils: - [Utils] DynamicGraphHelper.createDynamicGraph() took 5 ms.
PASSED: seleniumtest_firefox.TestNG3.verifyHomepageTitle

=====
Default test
Tests run: 1, Failures: 0, Skips: 0
=====

=====
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
=====
```



Practical No : 14

Q. Implementation of Data Driven Framework.

Code :

```
package newpackage;

import org.testng.annotations.Test;
import org.testng.annotations.DataProvider;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class DataDrivenTest {

    @Test(dataProvider = "loginData")

    public void loginTest(String username, String password) {

        System.setProperty("webdriver.chrome.driver",
            "D:\\\\NMITD-College\\\\SEM-3\\\\STQA-
            Lab\\\\Setups\\\\chromedriver\\\\chromedriver.exe");

        WebDriver driver = new ChromeDriver();

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

        WebElement user = driver.findElement(By.name("userName"));
        user.sendKeys(username);

        WebElement passwrđ = driver.findElement(By.name("password"));
        passwrđ.sendKeys(password);
```

```
WebElement submit_btn = driver.findElement(By.name("submit"));
submit_btn.click();

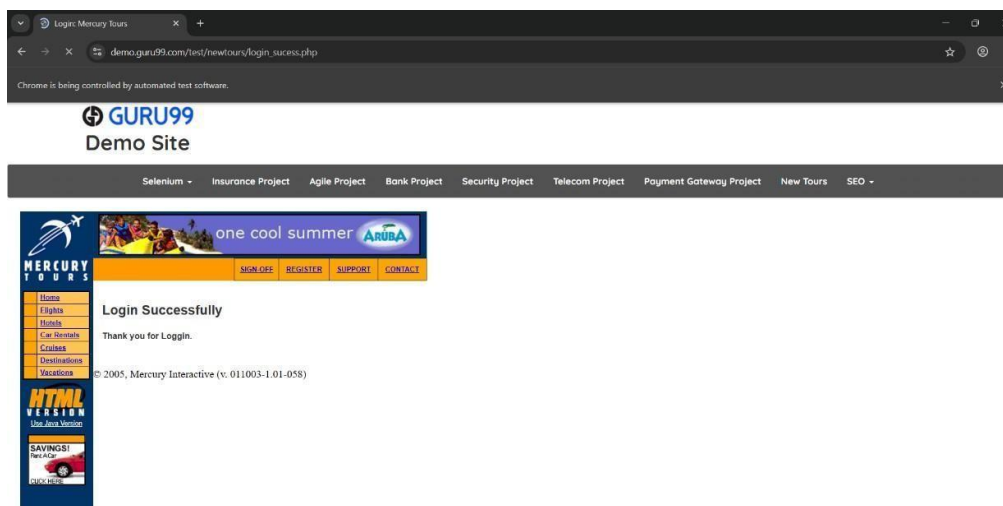
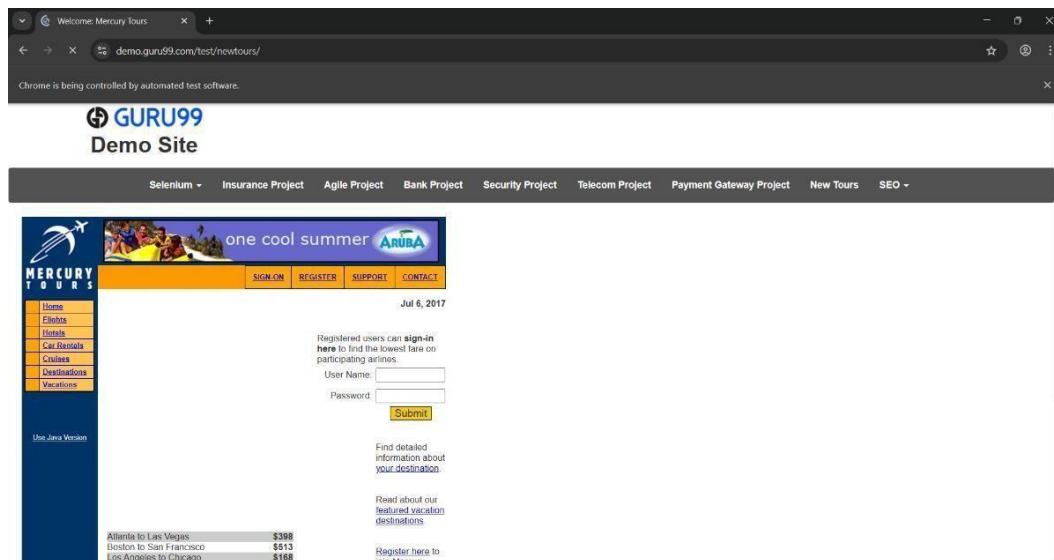
String expectedTitle = "Login: Mercury Tours";
String actualTitle = driver.getTitle();
System.out.println("Page Title is: " + actualTitle);

if (actualTitle.equals(expectedTitle)) {
    System.out.println("Login Test Passed");
} else {
    System.out.println("Login Test Failed");
}

driver.quit();
}

@DataProvider(name = "loginData")
public Object[][] getData() {
    return new Object[][] {
        {"mercury", "mercury"},
        {"tutorial", "tutorial"}
    };
}
}
```

Output :



Practical No : 15

Q. Demonstrate Validation testing.

Code :

```
package newpackage;

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

public class Validations_Test {

    public static void main(String[] args) {

        // Set ChromeDriver path

        System.setProperty("webdriver.chrome.driver",
                           "D:\\NMITD-College\\SEM-3\\STQA-
Lab\\Setups\\chromedriver\\chromedriver.exe");

        // Launch Chrome browser

        WebDriver driver = new ChromeDriver();

        // Open Google

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

        // Validate the title of the page

        String title = driver.getTitle();

        System.out.println("Title is: " + title);

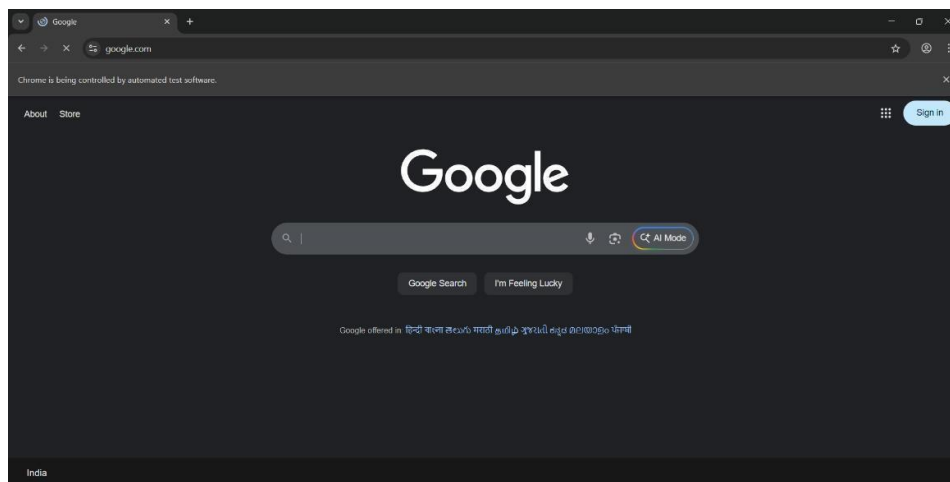
        // Validate the current URL
```

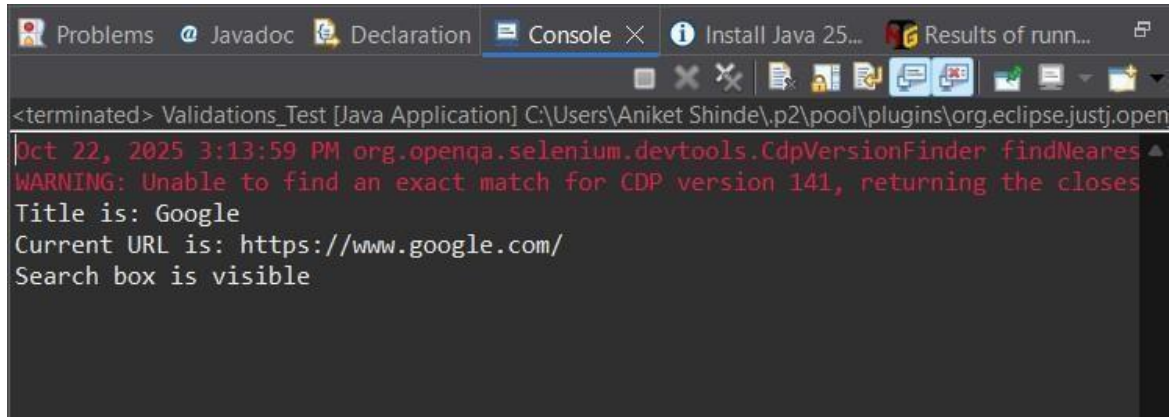
```
System.out.println("Current URL is: " + driver.getCurrentUrl());

// Check if the search box is visible using the updated XPath
boolean status = driver.findElement(By.xpath("//textarea[@name='q']")).isDisplayed();
if (status) {
    System.out.println("Search box is visible");
} else {
    System.out.println("Search box is not visible");
}

// Close the browser
driver.quit();
}
```

Output :





The screenshot shows the Eclipse IDE's Console window. The title bar includes tabs for 'Problems', 'Javadoc', 'Declaration', 'Console', 'Install Java 25...', and 'Results of runn...'. The console output shows a terminated Java application with the following log messages:

```
<terminated> Validations_Test [Java Application] C:\Users\Aniket Shinde\p2\pool\plugins\org.eclipse.justj.open  
Oct 22, 2025 3:13:59 PM org.openqa.selenium.devtools.CdpVersionFinder findNeares  
WARNING: Unable to find an exact match for CDP version 141, returning the closes  
Title is: Google  
Current URL is: https://www.google.com/  
Search box is visible
```