Green University of Bangladesh
Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering
Semester: (Fall, Year:2023), B.Sc. in CSE (Day)

Lab Report NO: 05
Course Title: Software Testing & Quality Assurance Lab.
Course Code: CSE 454 Section: D-8

Lab Experiment Name: Software Testing Tools (Setting Up Selenium WebDriver)

Student Details

	Name	ID
1.	Md. Mostak Mohosin	201002195

Lab Date : 14.12.23
Submission Date : 21.12.23
Course Teacher's Name : Mr. Montaser Abdul Quader

Lab Report Status
Marks: ………………………………… Signature:…………………
Comments:……………………………………… Date:………………………..

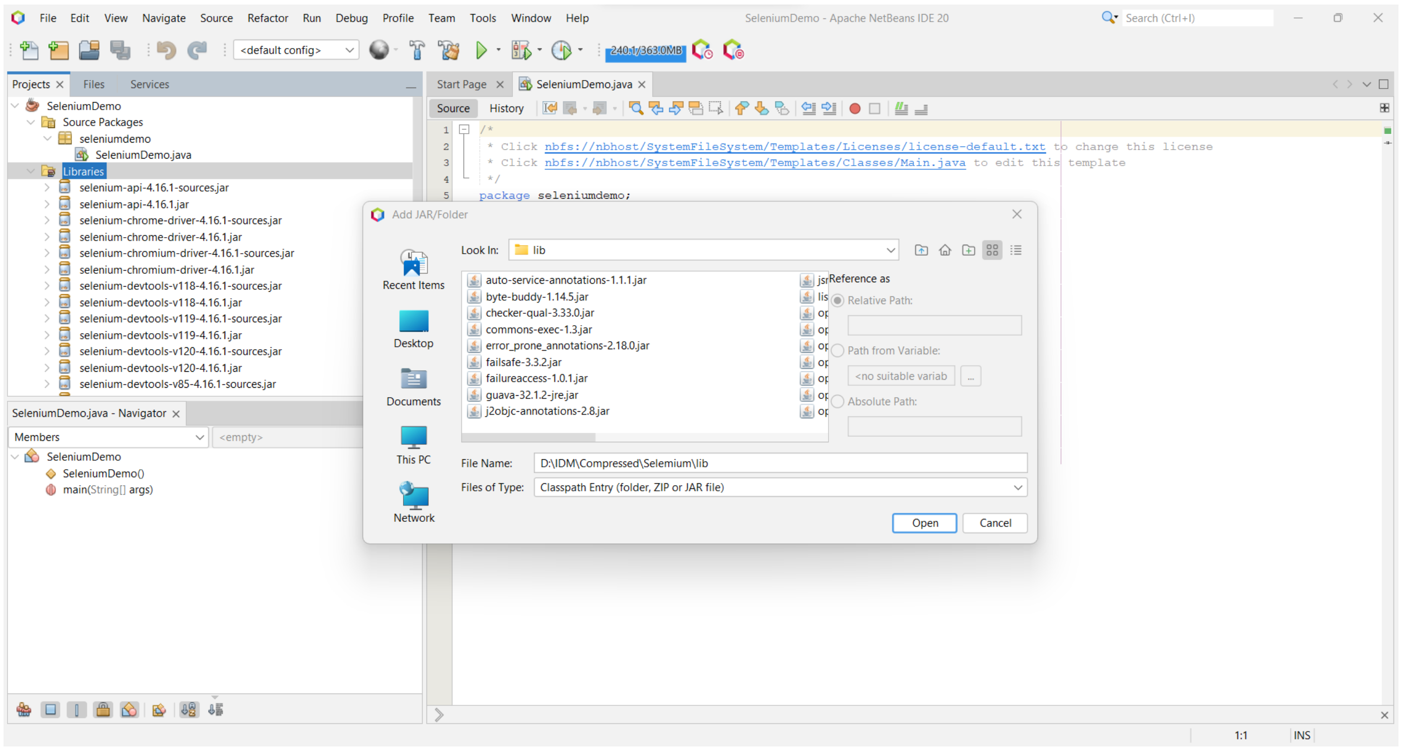
1. **TITLE OF THE LAB REPORT EXPERIMENT: Software Testing Tools (Setting Up Selenium WebDriver).**

2. **OBJECTIVES/AIM:**
   - To be familiar with web testing tools.
   - To gain practical knowledge on Selenium Web-driver.
   - To setup Selenium Web-driver for running automated testing.

3. **PROCEDURE / ANALYSIS:**
   1. **Environment Setup:**
      - Install Selenium WebDriver: Download and set up the Selenium WebDriver in the preferred programming language (Java, C#, Python, etc.).
      - Choose an Integrated Development Environment (IDE) like Eclipse or Visual Studio for coding.
   2. **Project Initialization:**
      - Create a new project for the Selenium test scripts.
      - Set up the project structure and dependencies.
   3. **WebDriver Configuration:**
      - Initialize the WebDriver based on the chosen browser (Chrome, Firefox, etc.).
      - Configure browser options and settings as needed.
   4. **Write Test Scripts:**
      - Use the selected programming language to write Selenium test scripts.
      - Implement actions like opening websites, interacting with elements, filling forms, and navigating through pages.
      - Include assertions to validate expected outcomes.
   5. **Handling Synchronization and Waits:**
      - Implement appropriate waits and synchronization mechanisms to handle dynamic web elements and ensure the stability of test scripts.
   6. **Test Case Design:**
      - Design test cases based on the application's functional requirements.
      - Organize test cases into test suites for better manageability.
   7. **Execution:**
      - Execute the Selenium test scripts on different browsers and platforms.
      - Monitor the test execution and identify any failures or errors.
   8. **Report Generation:**
      - Integrate a reporting mechanism to automatically generate comprehensive test reports.
      - Include details such as test pass/fail status, execution time, and error messages.

9. **Analysis:**
   - Analyze the test results to identify issues or defects in the application.
   - Trace and log any encountered errors for further investigation.

4. <u>IMPLEMENTATION OF TEST CASES</u>:



SeleniumDemo - Apache NetBeans IDE 20

Projects: SeleniumDemo › Source Packages › seleniumdemo › SeleniumDemo.java; Libraries:

```
selenium-api-4.16.1-sources.jar
selenium-api-4.16.1.jar
selenium-chrome-driver-4.16.1-sources.jar
selenium-chrome-driver-4.16.1.jar
selenium-chromium-driver-4.16.1-sources.jar
selenium-chromium-driver-4.16.1.jar
selenium-devtools-v118-4.16.1-sources.jar
selenium-devtools-v118-4.16.1.jar
selenium-devtools-v119-4.16.1-sources.jar
selenium-devtools-v119-4.16.1.jar
selenium-devtools-v120-4.16.1-sources.jar
selenium-devtools-v120-4.16.1.jar
selenium-devtools-v85-4.16.1-sources.jar
```

Add JAR/Folder

Look In: lib

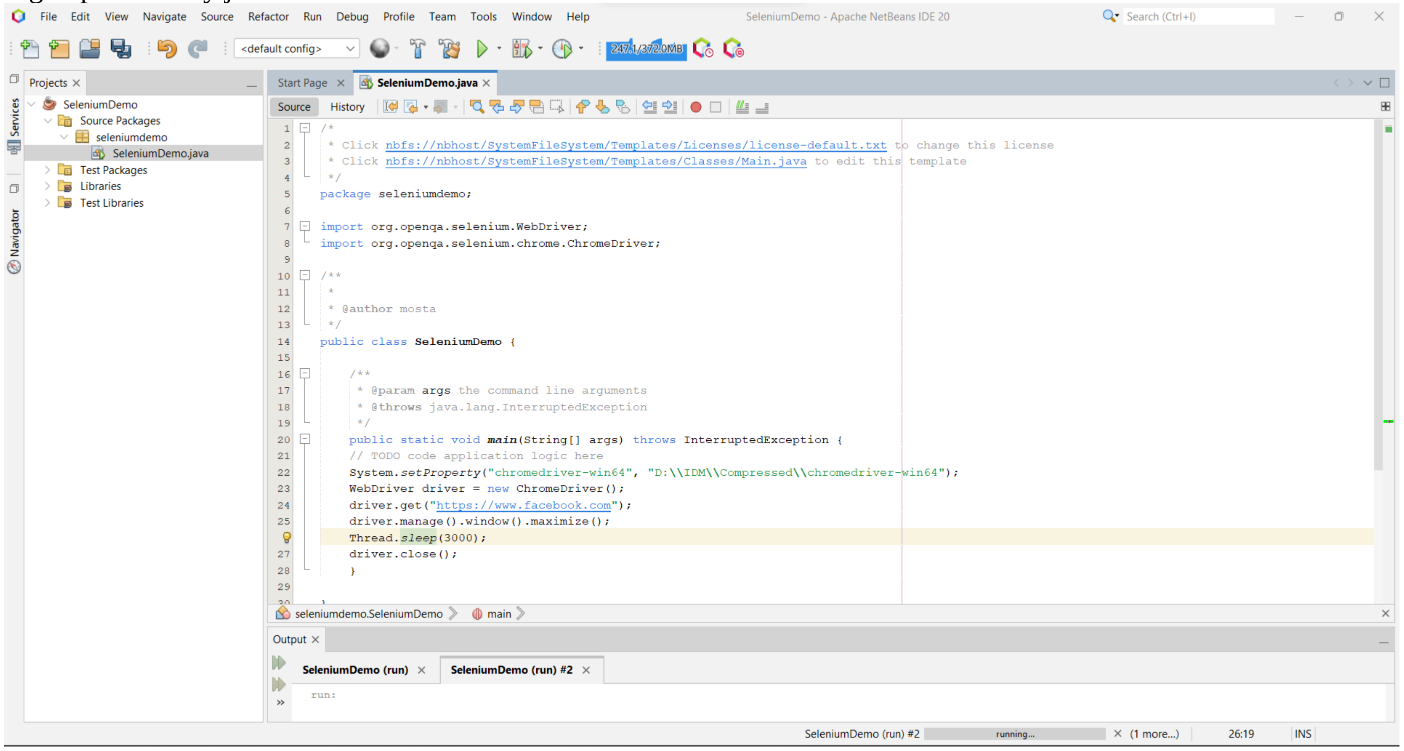
```
auto-service-annotations-1.1.1.jar
byte-buddy-1.14.5.jar
checker-qual-3.33.0.jar
commons-exec-1.3.jar
error_prone_annotations-2.18.0.jar
failsafe-3.3.2.jar
failureaccess-1.0.1.jar
guava-32.1.2-jre.jar
j2objc-annotations-2.8.jar
```

Reference as: [✓ Relative Path] / Path from Variable: <no suitable variab / Absolute Path

File Name: D:\IDM\Compressed\Selemium\lib

Files of Type: Classpath Entry (folder, ZIP or JAR file)

Fig: Upload every jar file from selenium driver



```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
 */
package seleniumdemo;

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

/**
 *
 * @author mosta
 */
public class SeleniumDemo {

    /**
     * @param args the command line arguments
     * @throws java.lang.InterruptedException
     */
    public static void main(String[] args) throws InterruptedException {
    // TODO code application logic here
    System.setProperty("chromedriver-win64", "D:\\IDM\\Compressed\\chromedriver-win64");
    WebDriver driver = new ChromeDriver();
    driver.get("https://www.facebook.com");
    driver.manage().window().maximize();
    Thread.sleep(3000);
    driver.close();
    }

}
```

Output – SeleniumDemo (run) #2

```
run:
```

Fig: Initial Code

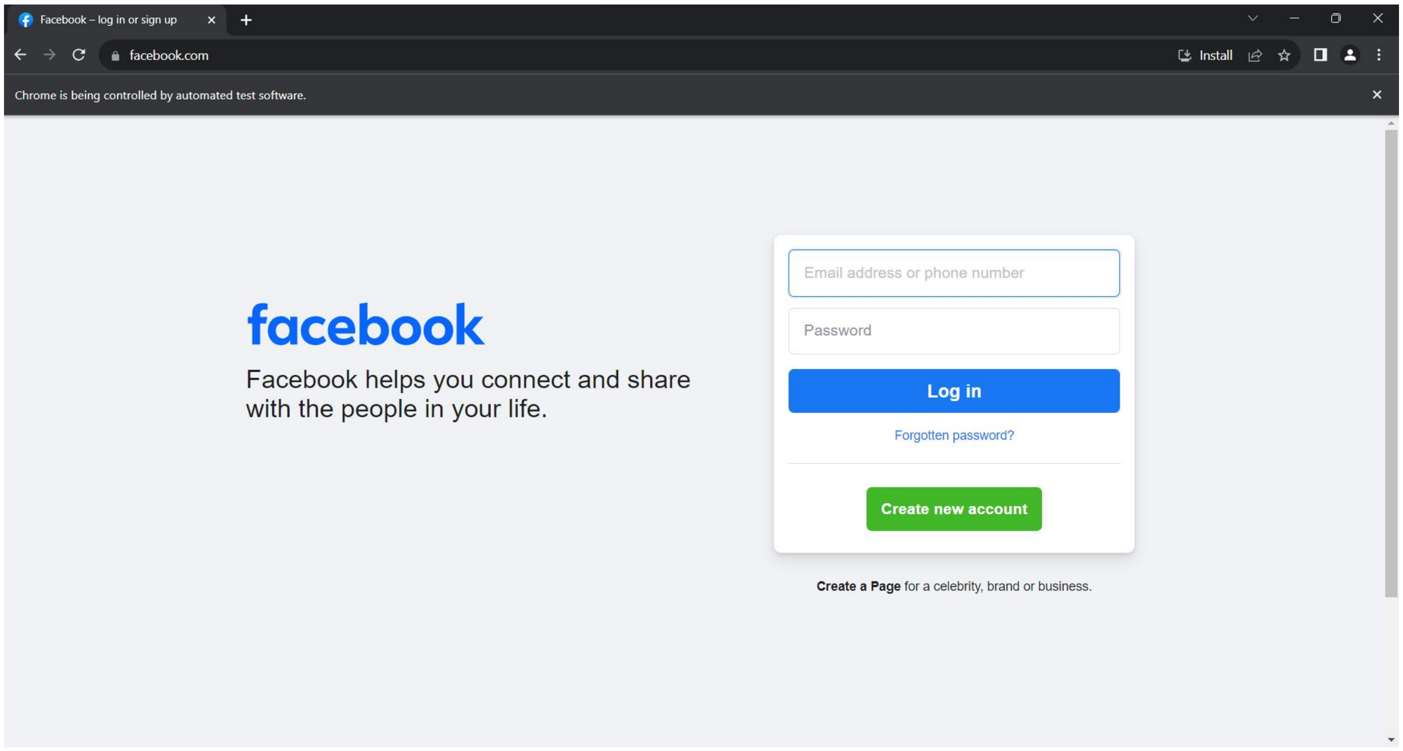

Facebook – log in or sign up
facebook.com
Install
Chrome is being controlled by automated test software.
facebook
Facebook helps you connect and share with the people in your life.
Email address or phone number
Password
Log in
Forgotten password?
Create new account
Create a Page for a celebrity, brand or business.

Fig: Final Output

5. <u>ANALYSIS AND DISCUSSION</u>:

Selenium WebDriver is a robust open-source automated testing framework offering significant efficiency gains through automated test execution. It ensures cross-browser compatibility, allowing applications to function reliably across various platforms. Resource optimization is achieved by reducing manual testing efforts and enabling parallel test execution. The tool's reliability is enhanced by robust mechanisms for handling dynamic web elements and synchronization issues. Comprehensive reporting features simplify result analysis, facilitating quick issue identification.

However, Selenium does pose challenges. Certain application types and frequent changes can be testing hurdles. Skill requirements include knowledge of programming languages and web technologies, demanding continuous skill development. Integration challenges may arise when using Selenium alongside other testing tools in a diverse testing ecosystem. While Selenium is primarily focused on functional testing, additional tools may be required for comprehensive security testing. Despite these challenges, Selenium benefits from a strong open-source community that provides support, updates, and ongoing development. Careful consideration of these factors ensures effective utilization of Selenium in automated testing processes

6. **SUMMARY:** Selenium WebDriver is a powerful open-source automated testing framework that offers efficiency gains through automated test execution, ensuring cross-browser compatibility and resource optimization. Its reliability is strengthened by robust mechanisms for handling dynamic web elements. Despite challenges such as application-specific issues and skill requirements, Selenium integrates well into diverse testing ecosystems. While primarily focused on functional testing, additional tools may be needed for comprehensive security testing. The framework benefits from strong community support, providing ongoing development and updates. Careful consideration of these factors enables effective utilization of Selenium in automated testing processes.