

Features of Selenium.**NAME:R.HIRANMAI****Description**

This section will guide you to understand:

- The features of Selenium

This guide has mainly one subsection, namely:

Features of Selenium

Steps : Features of Selenium

- **Open-Source:**

Selenium is a freeware and a portable tool. It has no upfront direct costs involved. The tool can be freely downloaded and the support for it is freely available, as it is community-based.

- **Supports languages:**

Selenium supports a range of languages, including Java, Perl, Python, C#, Ruby, Groovy, Java Script, etc. It has its own script, but it doesn't limit it to that language. It can work with various languages and whatever the developers/testers are comfortable with.

- **Supports Operating Systems:**

Selenium operates across and supports multiple Operating Systems, (OS) like Windows, Mac, Linux, UNIX, etc. With Selenium Suite of solutions, a tailored testing suite can be created over any platform and then executed on another one. For instance, you can create test cases using Windows OS and run it with ease on a Linux-based system.

- **Supports multiple browsers:**

Selenium provides support across multiple browsers, namely, Internet Explorer, Chrome, Firefox, Opera, Safari, etc. This becomes highly resourceful while executing tests and testing it across various browsers simultaneously.

The browsers supported by the Selenium packages are:

- Selenium IDE can be used with Firefox as a plug-in.
- Selenium RC and Webdriver supports diverse browsers, such as Internet Explorer.
- Supports programming languages and frameworks
Selenium integrates with programming languages and various frameworks. For instance, it can integrate with ANT or Maven type of framework for source code compilation. Furthermore, it can integrate with the TestNG testing framework for testing applications and reporting purposes. It can integrate with Jenkins or Hudson for Continuous Integration (CI) and can even integrate with other Open-Source tools to support other features.
- **Tests across devices**
Selenium Test Automation can be implemented for mobile web application automation on Android, iPhone, and Blackberry. This can help in generating necessary results and addresses issues on a continuous basis.
- **Constant updates**
Selenium support is community-based and active community support enables constant updates and upgrades. These upgrades are readily available and do not require specific training. This makes Selenium resourceful and cost-effective as well.
- **Loaded Selenium Suites**
Selenium is not just a singular tool or utility, it is a loaded package of various testing tools and so is referred to as a Suite. Each tool is designed to cater to different testing needs and requirements of test environments.

Additionally, Selenium comes with capabilities to support Selenium IDE, Selenium Grid, and Selenium Remote Control (RC).

- **Ease of implementation**
Selenium offers a user-friendly interface that helps create and execute tests easily and effectively. Its open-source features help users to script their own extensions which makes it easy to develop customized actions and even manipulate at an advanced level. Tests run directly across

browsers and users can watch while the tests are being executed. Additionally, Selenium's reporting capabilities are one of the reasons for being chosen, as it allows testers to extract results and take follow-up actions.

- **Reusability and Add-ons**

Selenium Test Automation Framework uses scripts that can be tested directly across multiple browsers. Concurrently, it is possible to execute multiple tests with Selenium, as it covers almost all aspects of functional testing by implementing add-on tools that broaden the scope of testing.

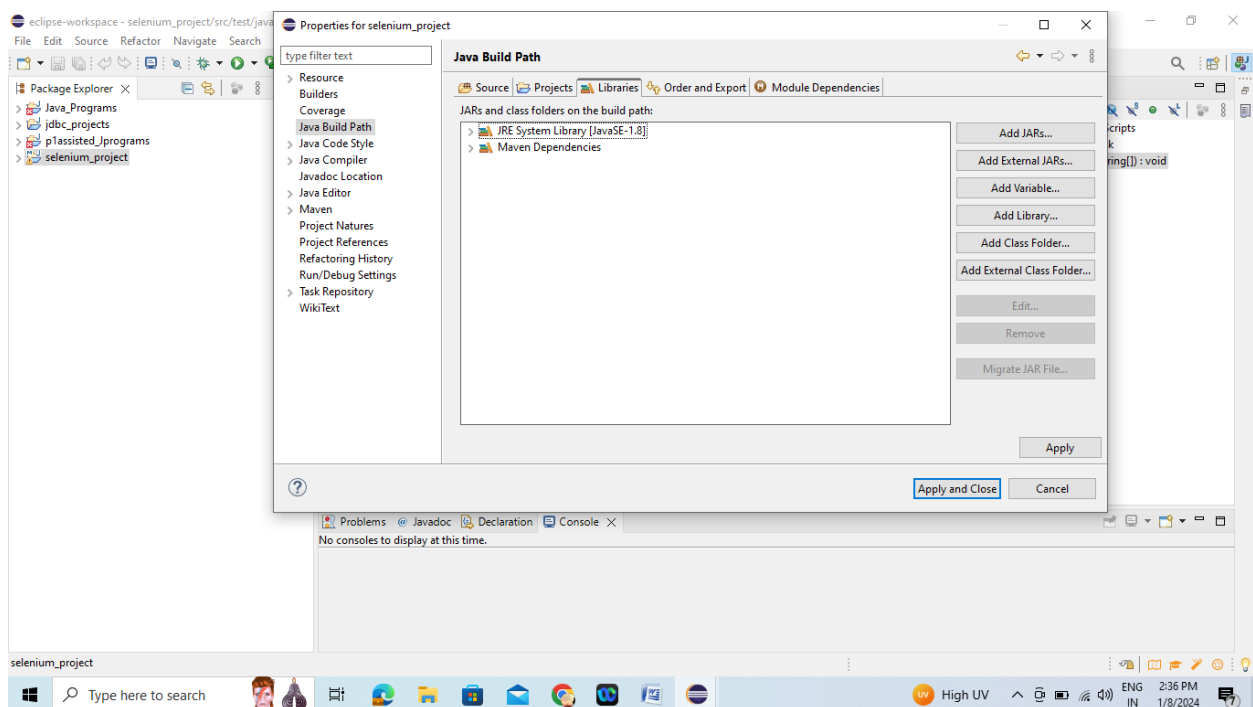
**WEBDRIVER INSTALLATION AND
INTEGRATION IN ECLIPSE**

Step 1: Launching Eclipse and creating a Java project

- Launch the Eclipse and create a Workspace.
- Create Project:
Click on File -> New -> Java Project.

Step 2: Configuring WebDriver with Eclipse

- Add selenium standalone server jars.
- Right-click on Project -> select Properties -> Select Java Build Path.
- Navigate to the Libraries tab and click on the Add External Jars button.
- Add selenium standalone server Jar files.
- Click on the Apply and Close button.
- In Eclipse, it looks like the screenshot below:

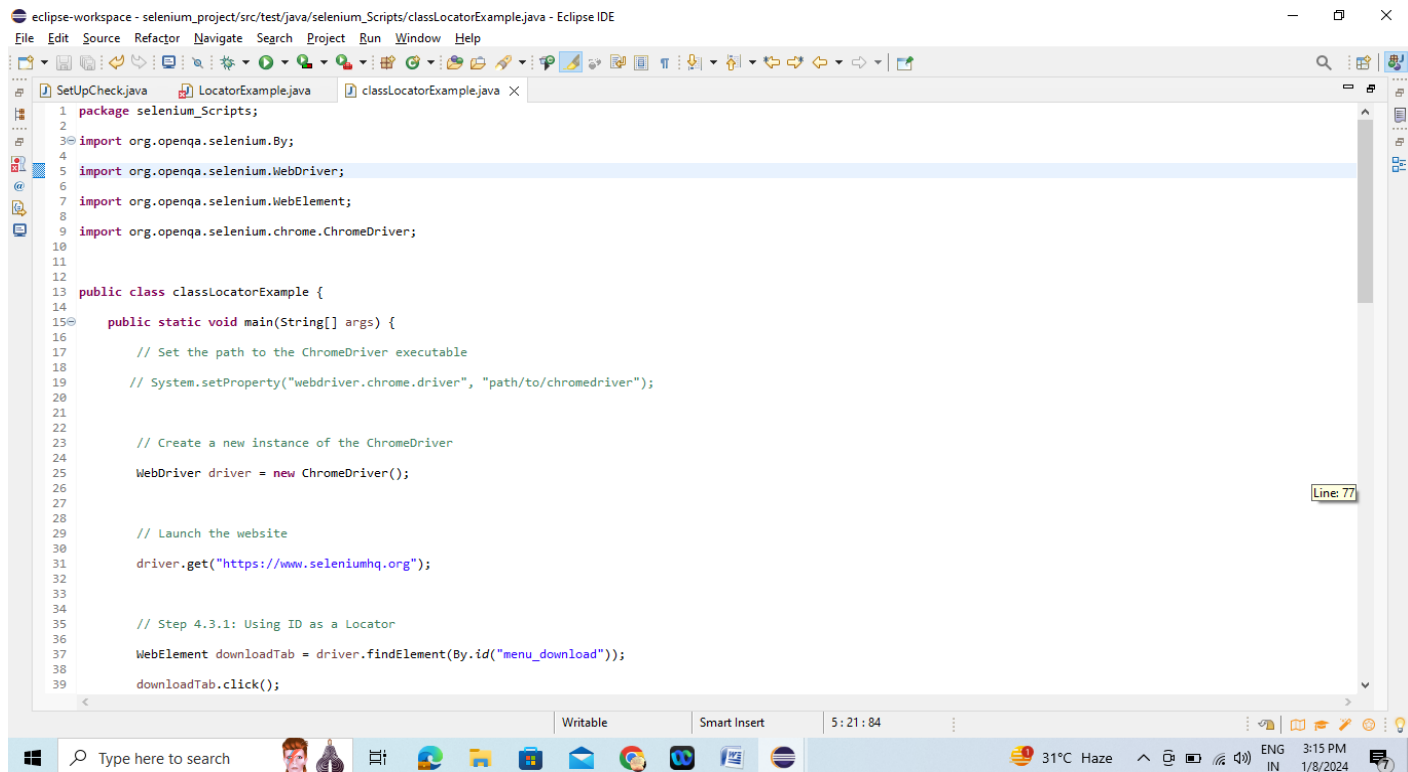


ASSISTED PRACTICE -3

EMP ID: 2587325

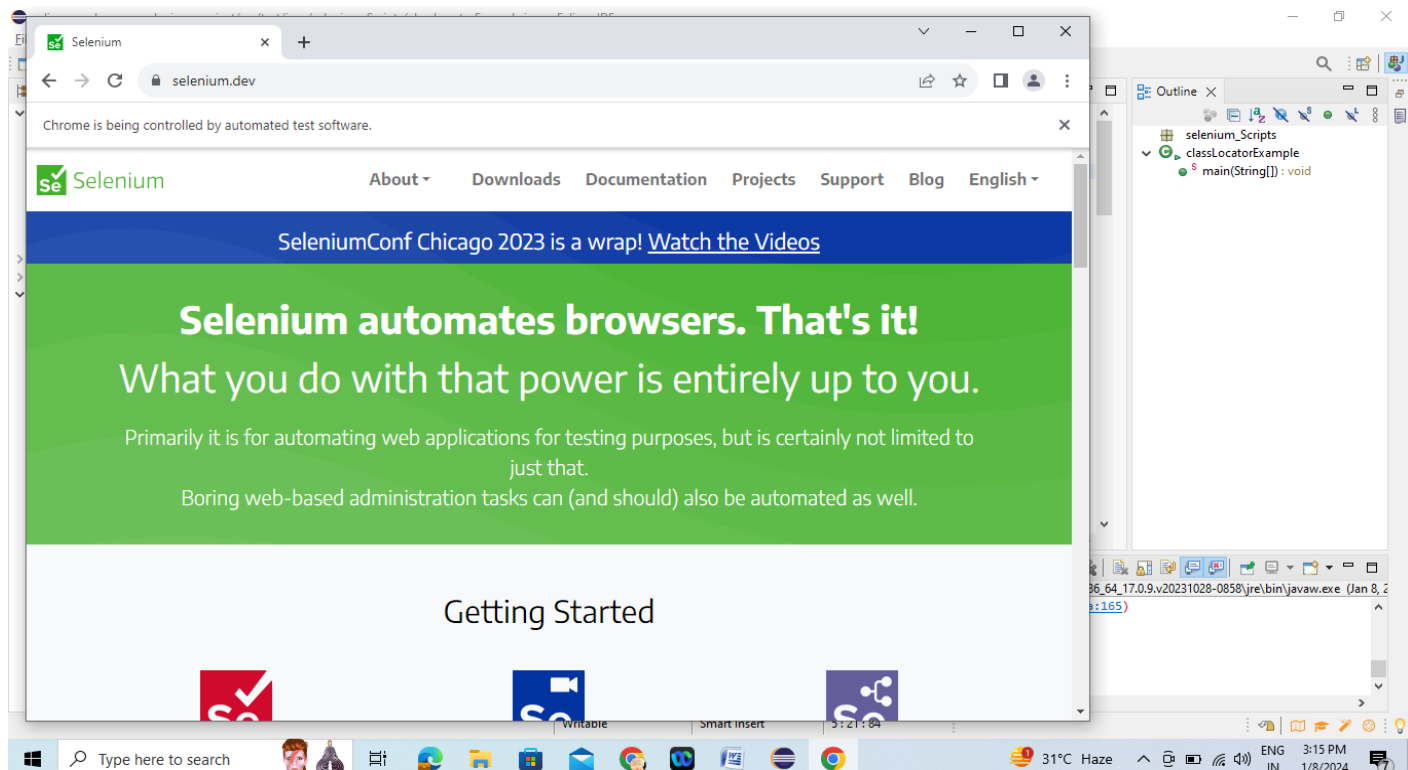
Multiple Ways to Locate Elements

NAME: HIRANMAI



The screenshot shows the Eclipse IDE with the file `classLocatorExample.java` open. The code is as follows:

```
1 package selenium_Scripts;
2
3 import org.openqa.selenium.By;
4
5 import org.openqa.selenium.WebDriver;
6
7 import org.openqa.selenium.WebElement;
8
9 import org.openqa.selenium.chrome.ChromeDriver;
10
11
12 public class classLocatorExample {
13
14     public static void main(String[] args) {
15
16         // Set the path to the ChromeDriver executable
17         // System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");
18
19
20         // Create a new instance of the ChromeDriver
21
22
23         WebDriver driver = new ChromeDriver();
24
25
26         // Launch the website
27
28         driver.get("https://www.seleniumhq.org");
29
30
31         // Step 4.3.1: Using ID as a Locator
32         WebElement downloadTab = driver.findElement(By.id("menu_download"));
33
34         downloadTab.click();
35     }
36 }
```

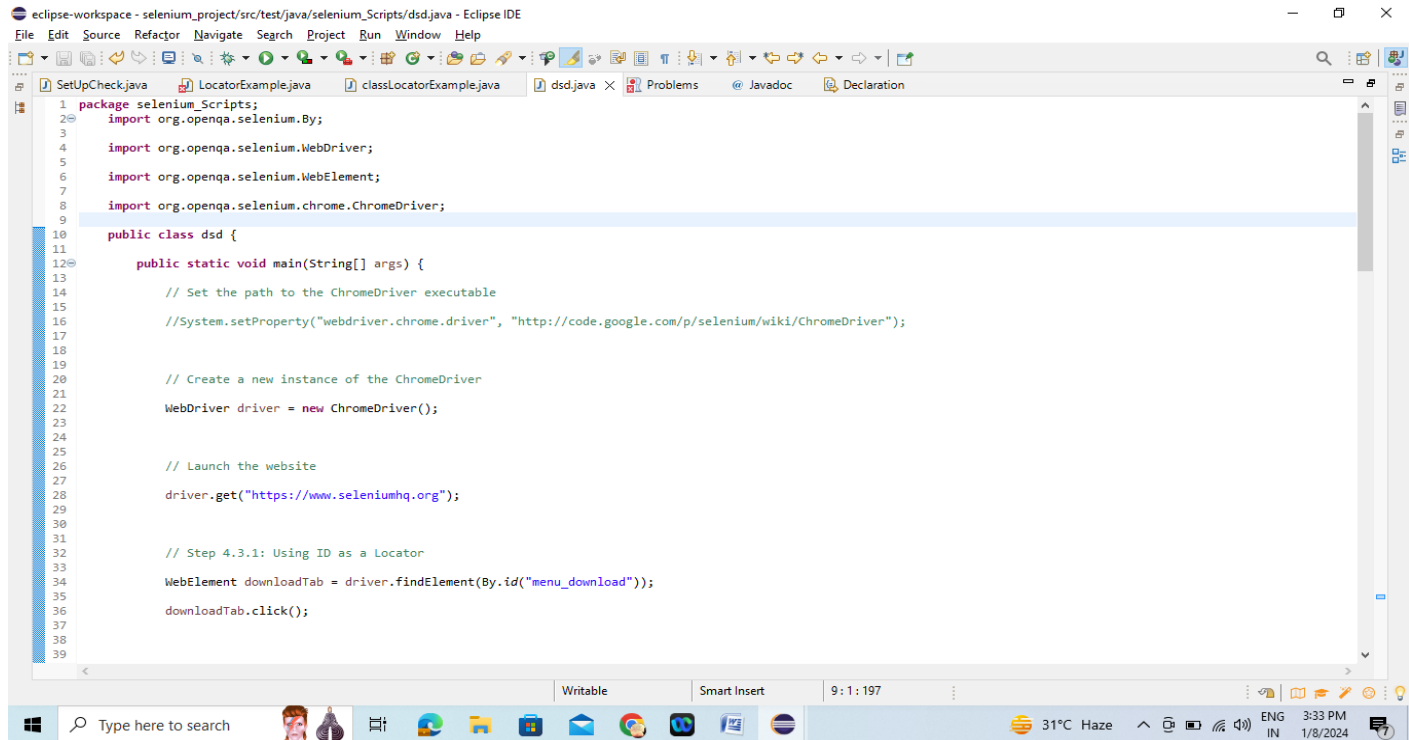


ASSISTED PRACTICE – 4

EMP ID-2587325

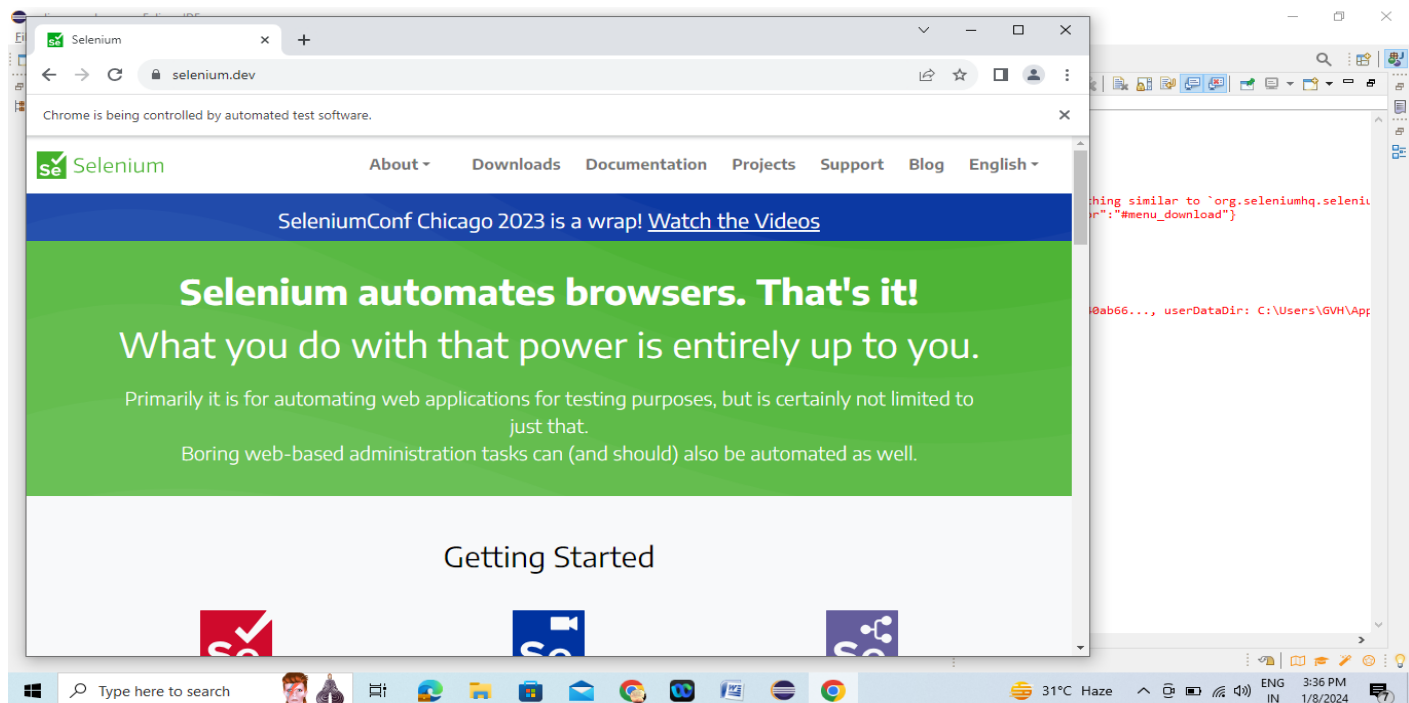
Locating Elements through CSS and XPath.

NAME: HIRANMAI



The screenshot shows the Eclipse IDE with a Java file named `dsd.java` open. The code is a Selenium script that sets up a ChromeDriver, launches a browser, and finds a web element by ID. The code is as follows:

```
1 package selenium_Scripts;
2 import org.openqa.selenium.By;
3
4 import org.openqa.selenium.WebDriver;
5
6 import org.openqa.selenium.WebElement;
7
8 import org.openqa.selenium.chrome.ChromeDriver;
9
10 public class dsd {
11
12     public static void main(String[] args) {
13
14         // Set the path to the ChromeDriver executable
15         //System.setProperty("webdriver.chrome.driver", "http://code.google.com/p/selenium/wiki/ChromeDriver");
16
17
18         // Create a new instance of the ChromeDriver
19         WebDriver driver = new ChromeDriver();
20
21         // Launch the website
22         driver.get("https://www.seleniumhq.org");
23
24
25         // Step 4.3.1: Using ID as a Locator
26         WebElement downloadTab = driver.findElement(By.id("menu_download"));
27         downloadTab.click();
28     }
29 }
30
31
32
33
34
35
36
37
38
39
```

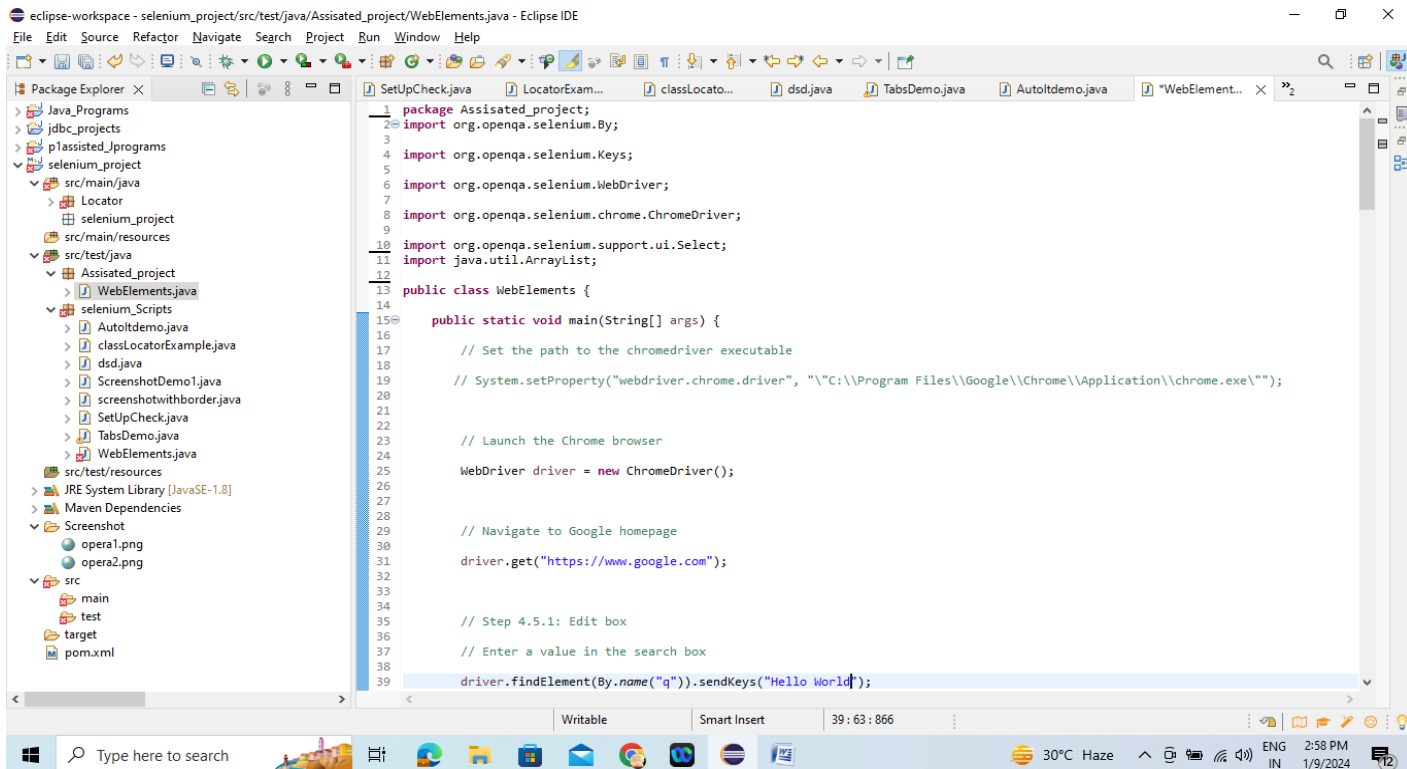


ASSISTED PRACTICE – 5

EMP ID- 2587325

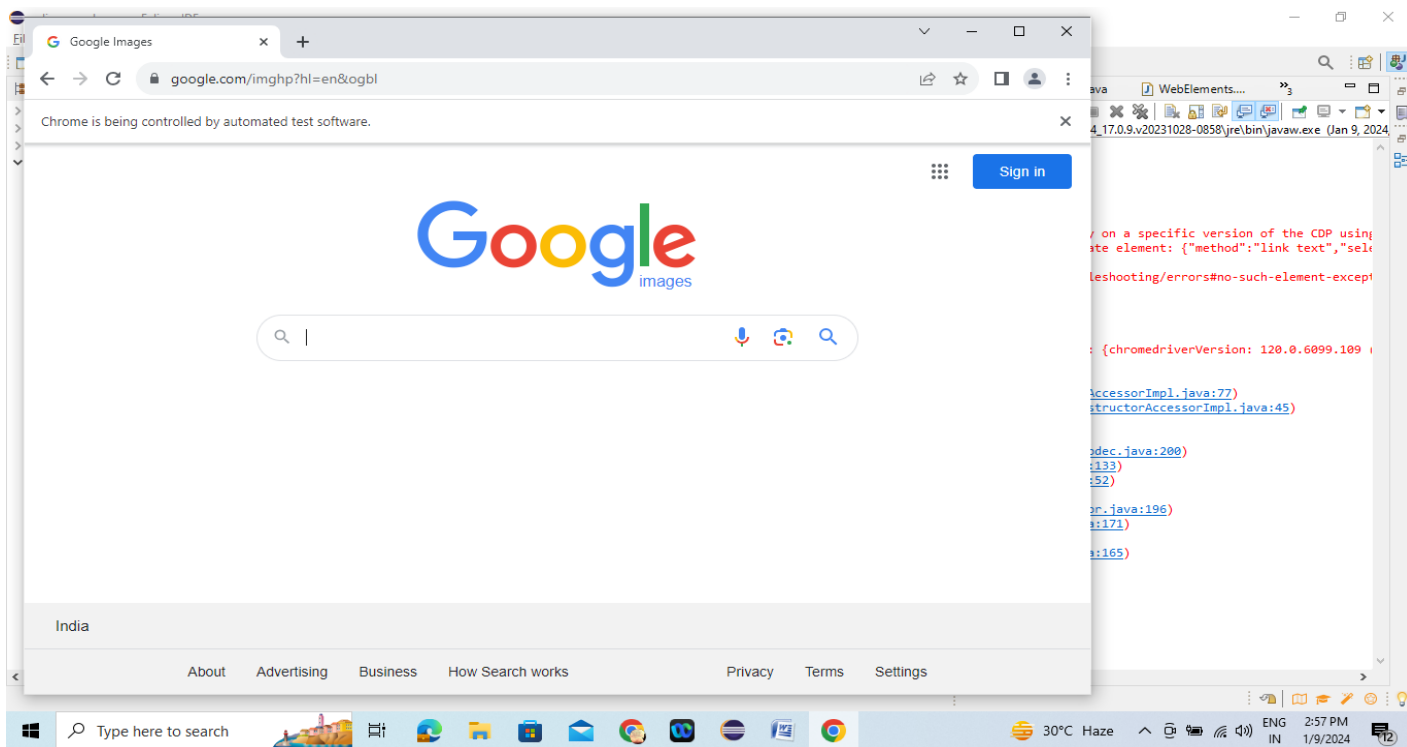
Handling Various Web Elements

NAME: HIRANMAI



The screenshot shows the Eclipse IDE with the 'WebElements.java' file open. The Package Explorer on the left shows the project structure, including 'src/main/java', 'src/main/resources', 'src/test/java', and 'src/test/resources'. The main editor displays the following Java code:

```
1 package Assisted_project;
2 import org.openqa.selenium.By;
3
4 import org.openqa.selenium.Keys;
5
6 import org.openqa.selenium.WebDriver;
7
8 import org.openqa.selenium.chrome.ChromeDriver;
9
10 import org.openqa.selenium.support.ui.Select;
11 import java.util.ArrayList;
12
13 public class WebElements {
14
15     public static void main(String[] args) {
16
17         // Set the path to the chromedriver executable
18         // System.setProperty("webdriver.chrome.driver", "\"C:\\Program Files\\Google\\Chrome\\Application\\chrome.exe\"");
19
20         // Launch the Chrome browser
21         WebDriver driver = new ChromeDriver();
22
23         // Navigate to Google homepage
24         driver.get("https://www.google.com");
25
26         // Step 4.5.1: Edit box
27         // Enter a value in the search box
28         driver.findElement(By.name("q")).sendKeys("Hello World");
29     }
30 }
```



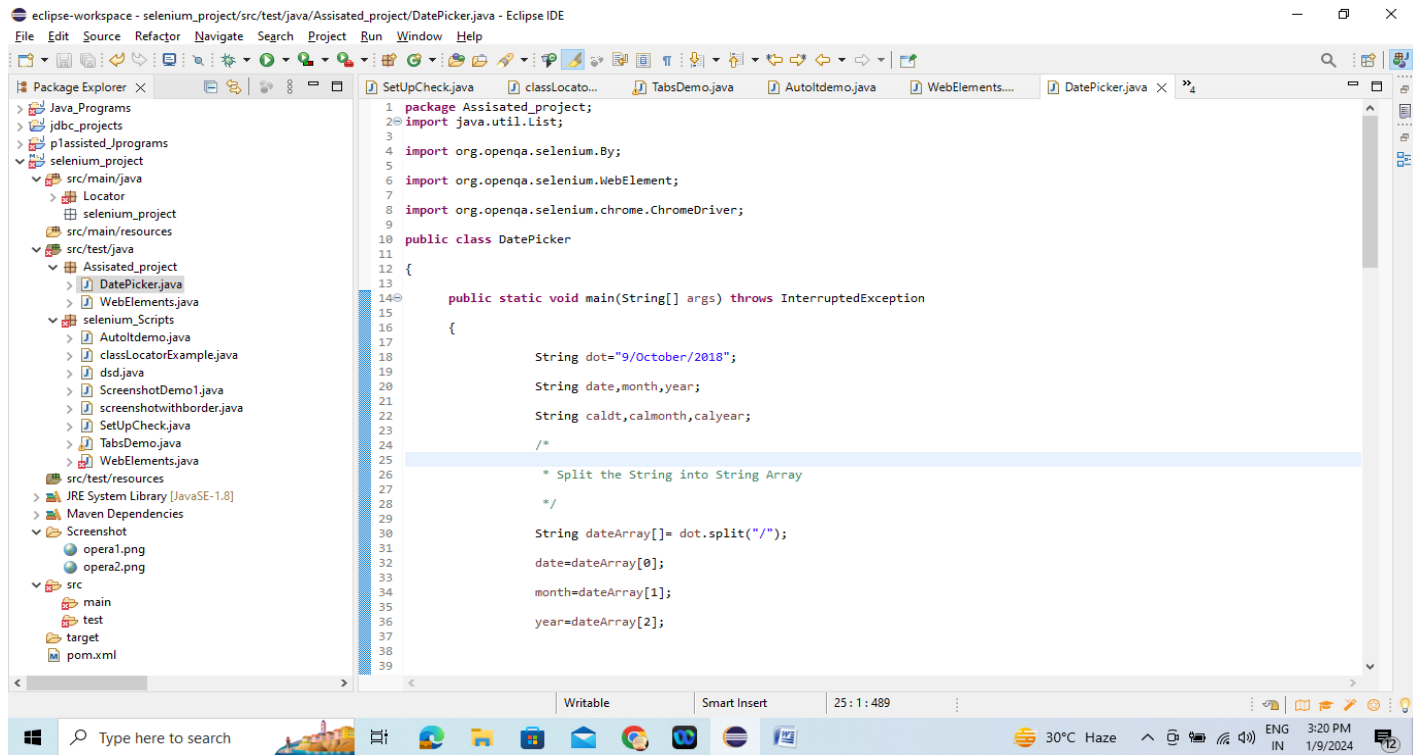
ASSISTED PRACTICE – 6

EMP ID -2587325

AUTOMATE CALENDER

ON WEB PAGE

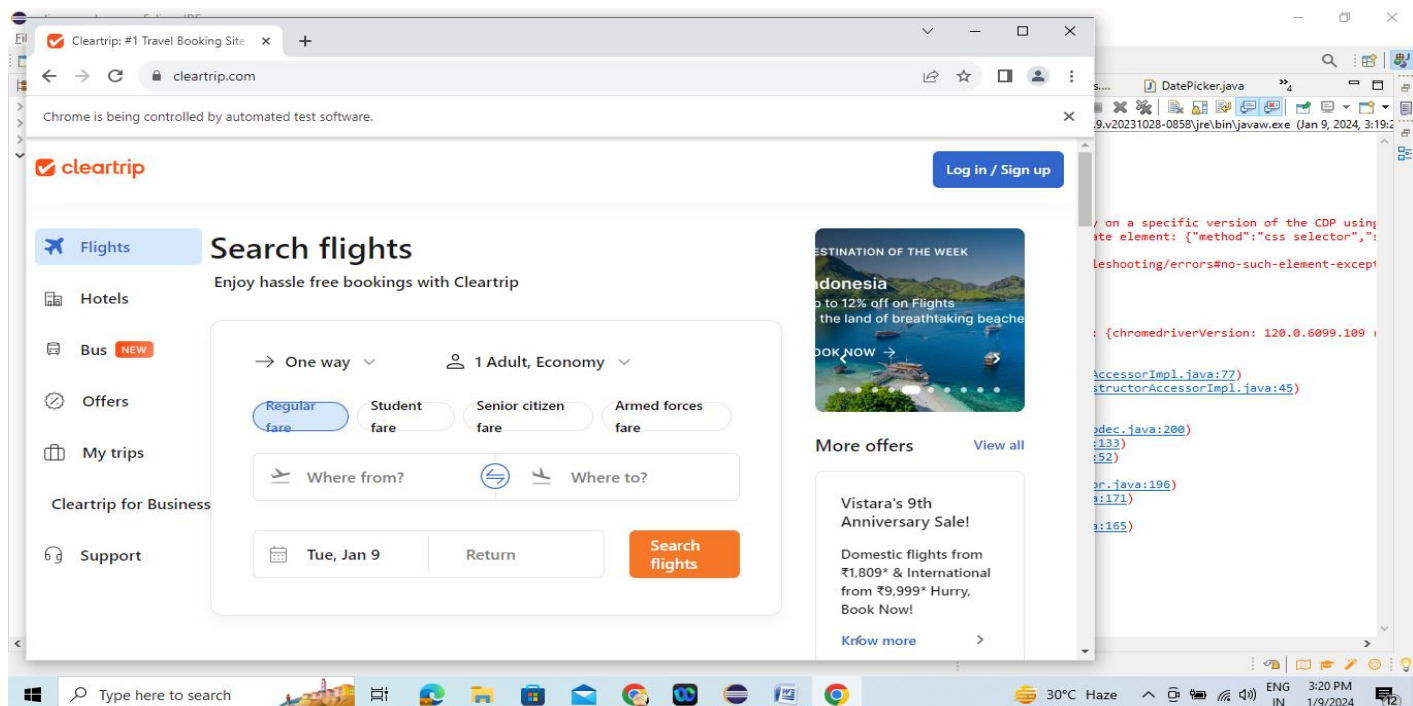
NAME: HIRANMAI



The screenshot shows the Eclipse IDE with the following details:

- Package Explorer:** Shows a project structure with packages like `src/main/java`, `src/main/resources`, `src/test/java`, and `src/test/resources`. The `Assisted_project` package is expanded, showing `DatePicker.java`.
- Code Editor:** Displays the `DatePicker.java` file with the following code:

```
1 package Assisted_project;
2 import java.util.List;
3
4 import org.openqa.selenium.By;
5
6 import org.openqa.selenium.WebElement;
7
8 import org.openqa.selenium.chrome.ChromeDriver;
9
10 public class DatePicker
11 {
12
13
14     public static void main(String[] args) throws InterruptedException
15     {
16
17         String dot="9/October/2018";
18         String date,month,year;
19         String caldt,calmonth,calyear;
20
21         /*
22          * Split the String into String Array
23          */
24         String dateArray[]= dot.split("/");
25         date=dateArray[0];
26         month=dateArray[1];
27         year=dateArray[2];
28     }
29 }
```
- Bottom Bar:** Shows the status bar with "Writable", "Smart Insert", and a progress indicator at 25:1:489.

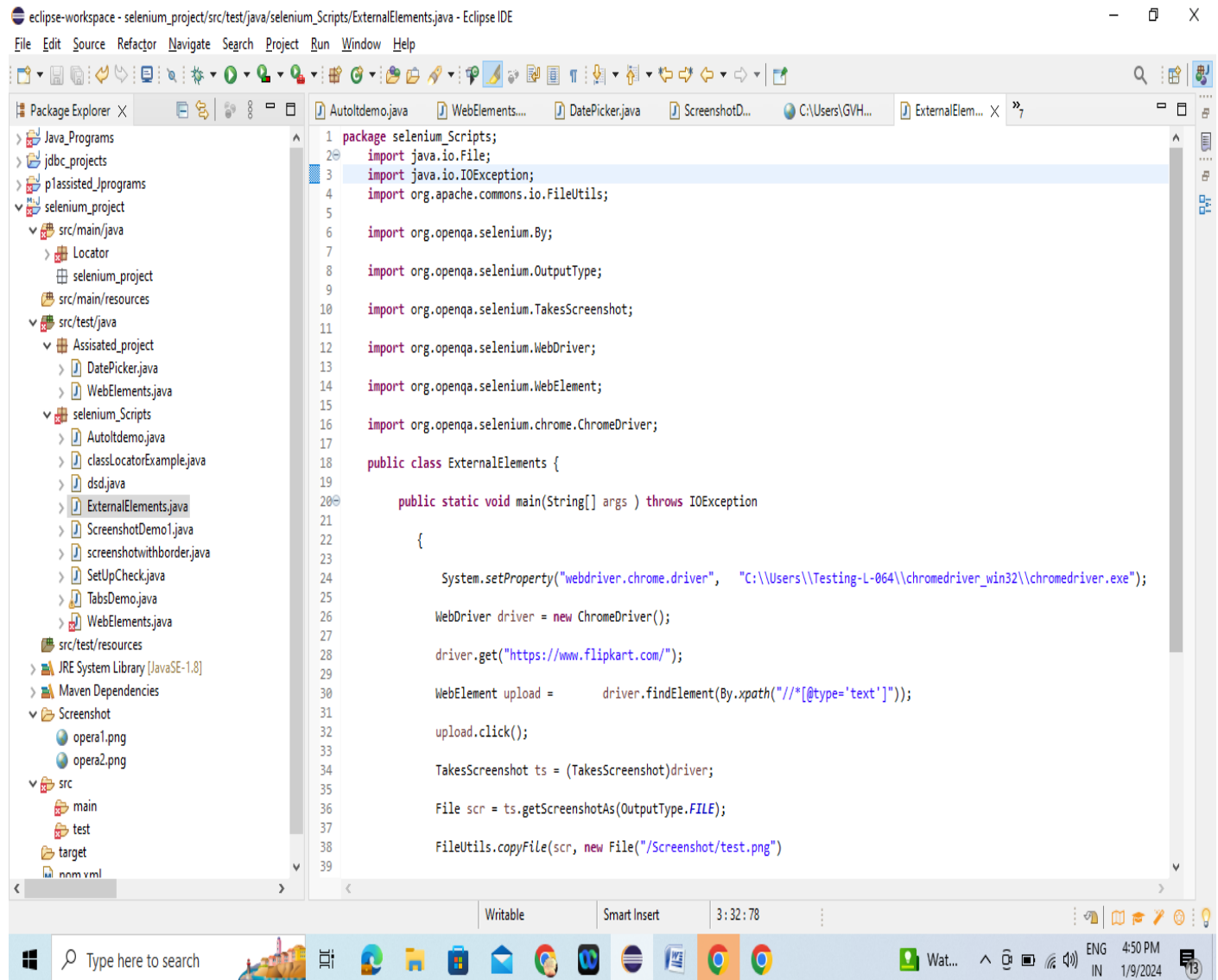


ASSISTED PROJECT – 7

EMP ID-2587325

Working with External Elements

NAME: HIRANMAI

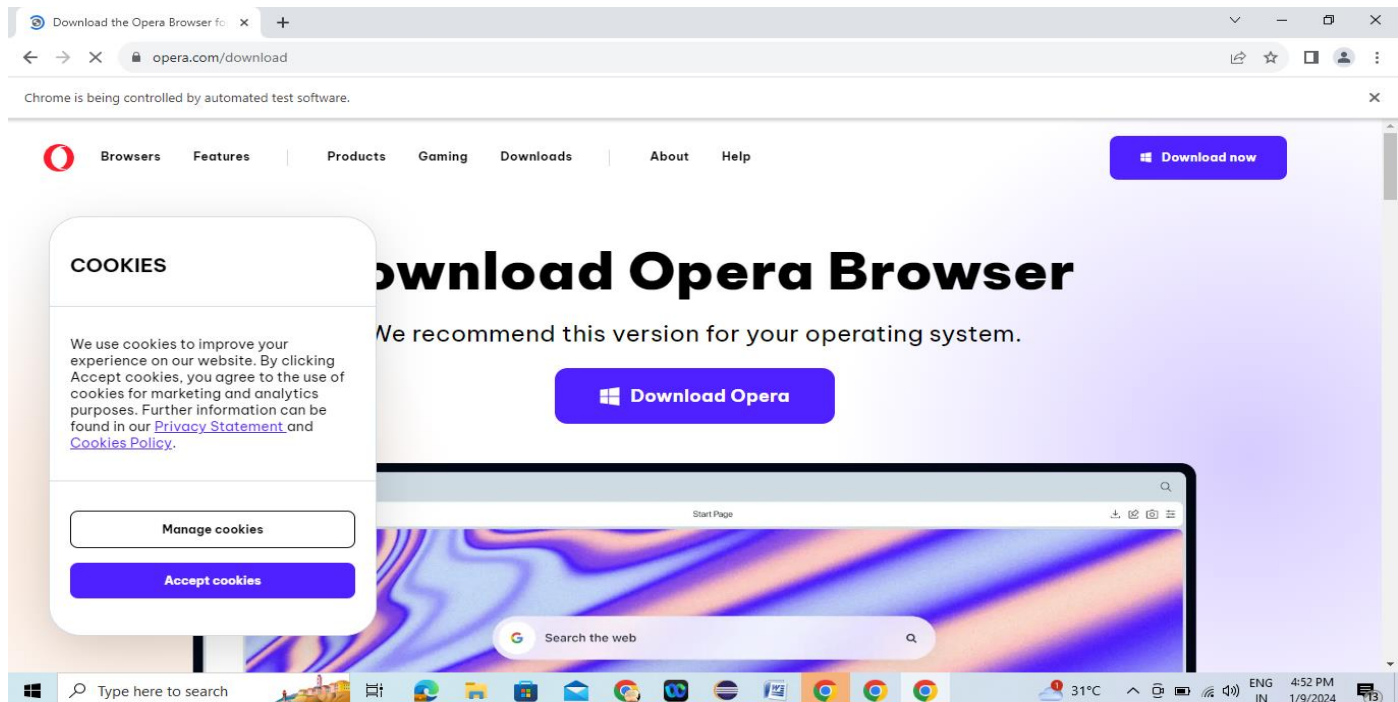
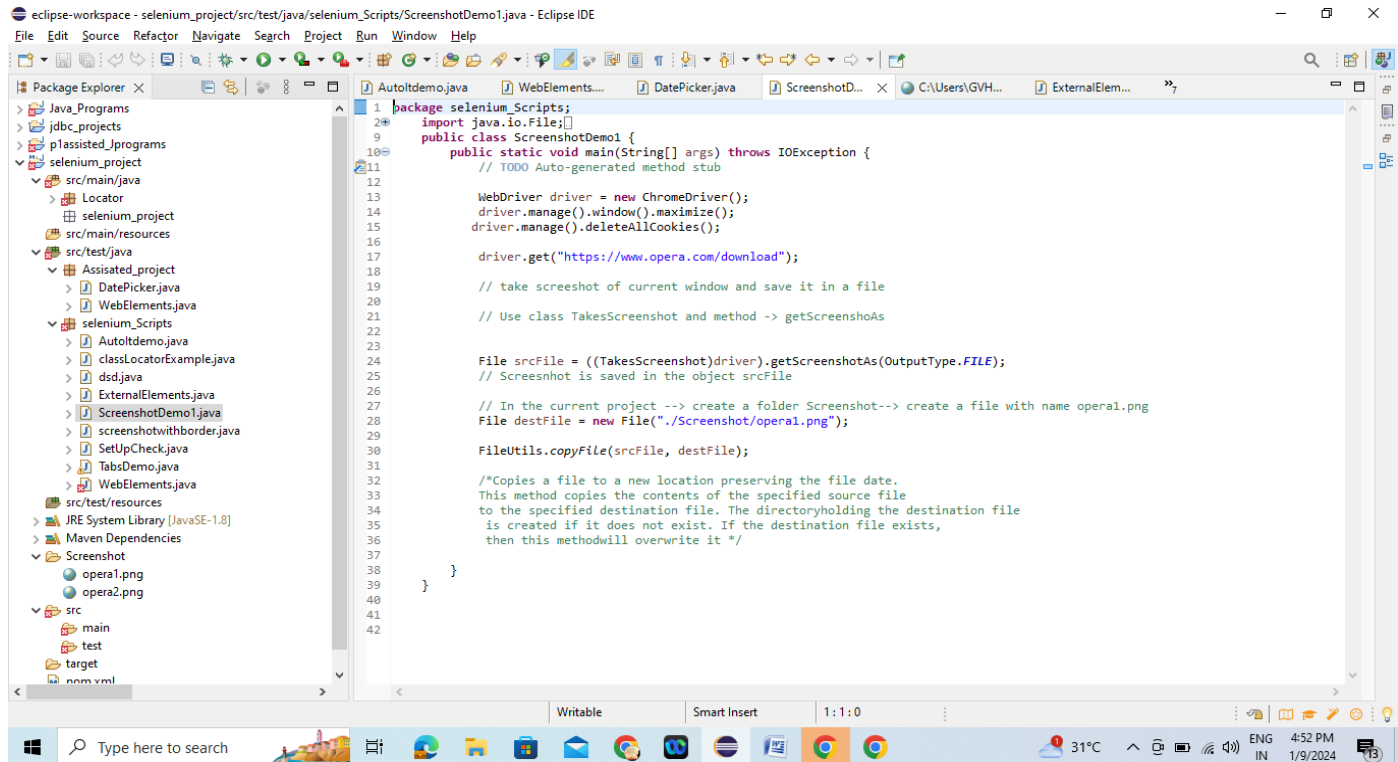


ASSISTED PROJECT – 8

EMP ID-2587325

Screenshots and Browser profiles

NAME: HIRANMAI

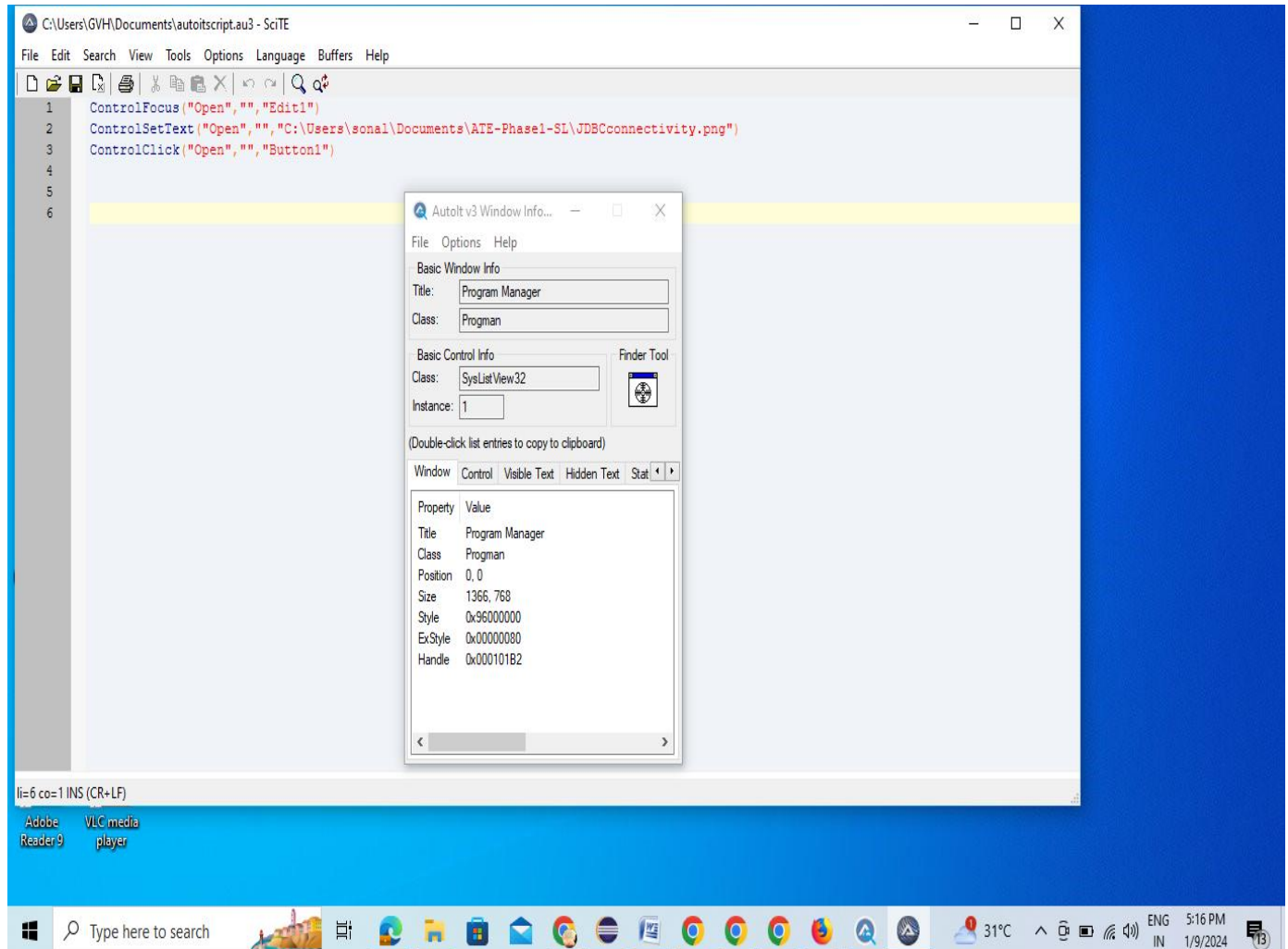


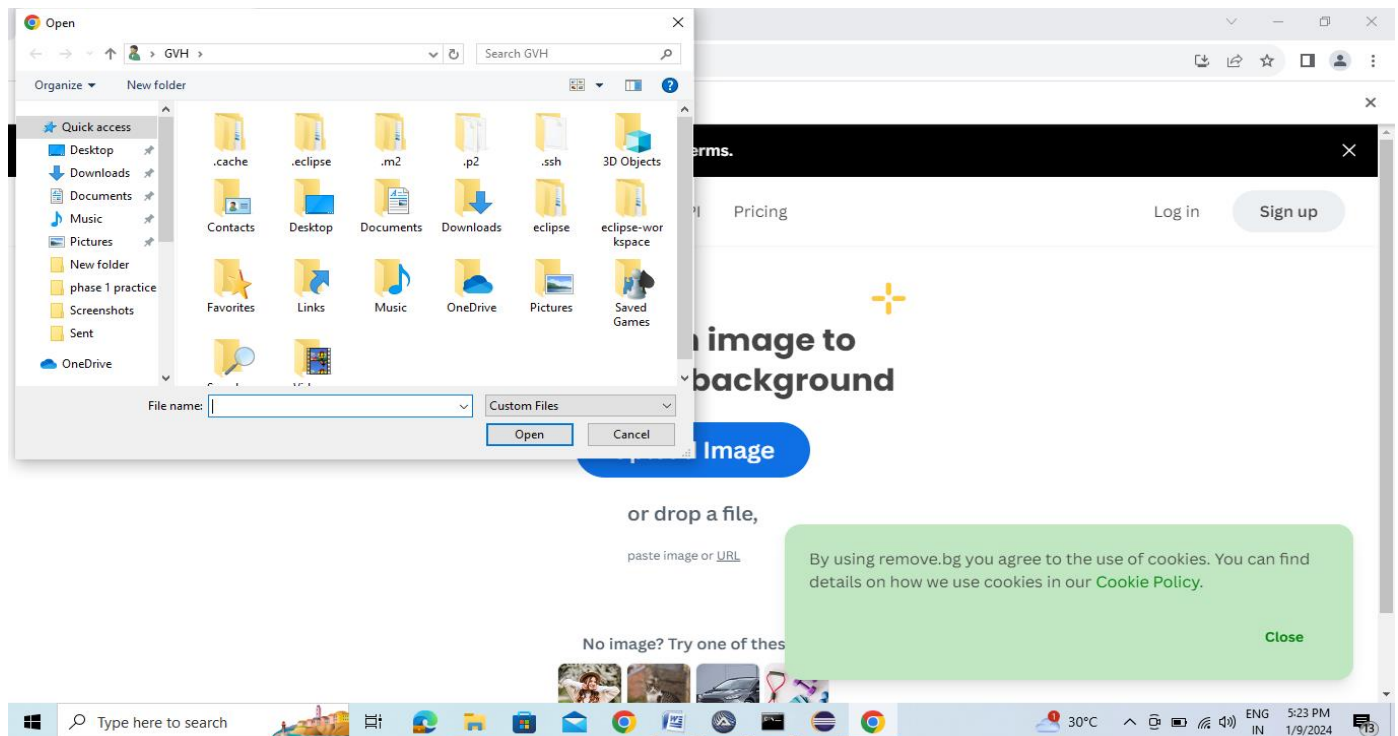
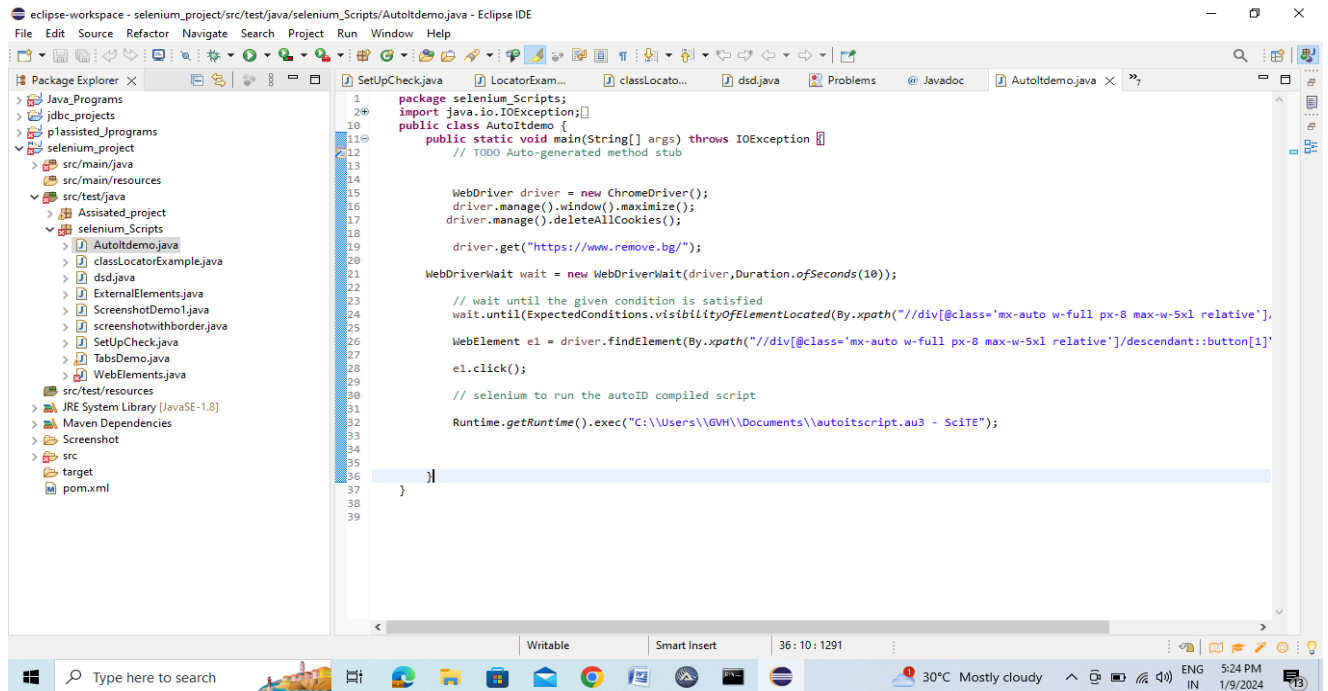
ASSISTED PROJECT – 9

EMP ID-2587325

Installation and configuration of AUTOIT

NAME: HIRANMAI



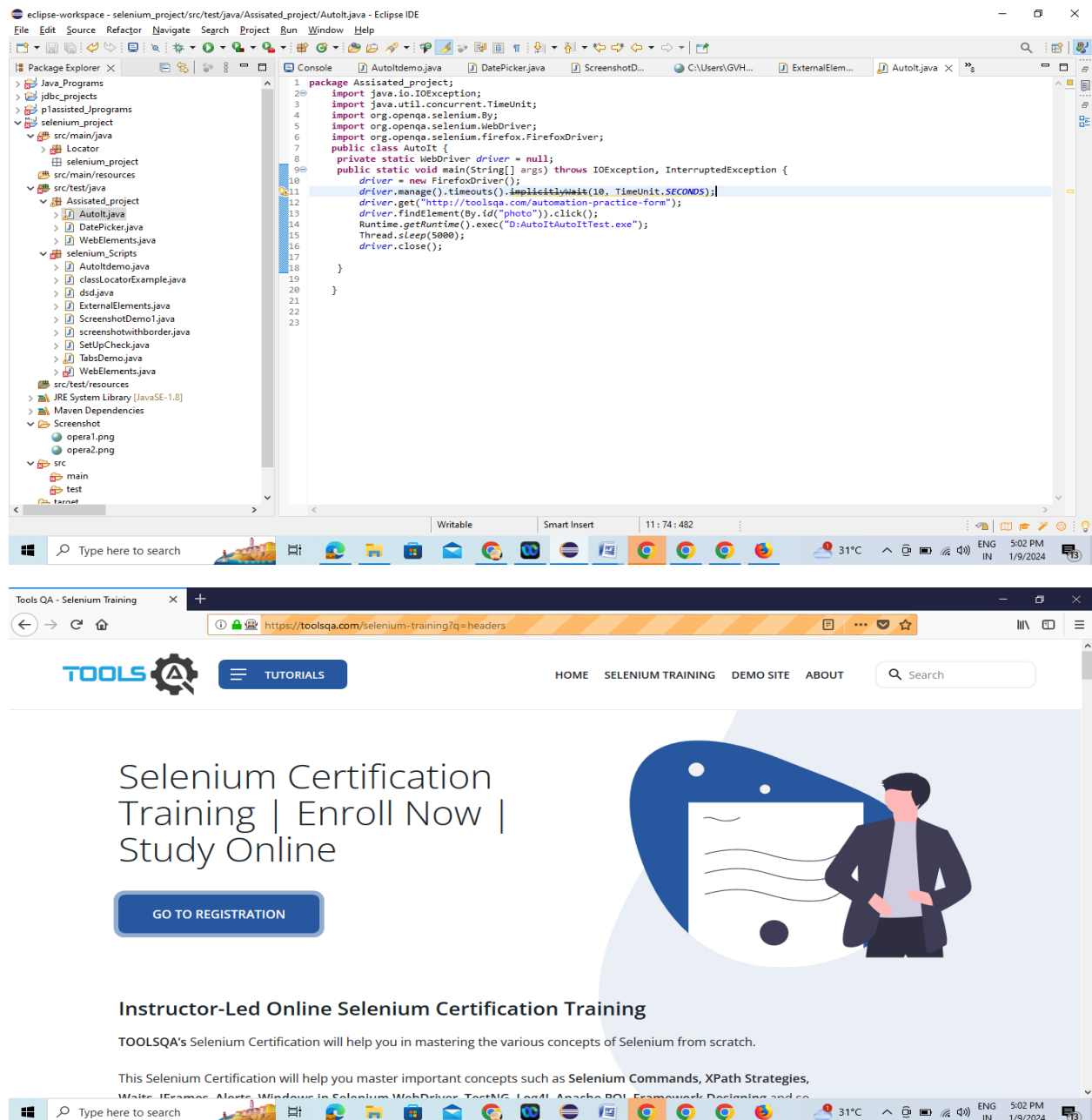


ASSISTED PROJECT – 10

EMP ID-2587325

Handling File Uploads.

NAME: HIRANMAI



The image displays two screenshots. The top screenshot shows the Eclipse IDE interface with a Java project named 'Assisted_project'. The Package Explorer on the left shows the project structure, including 'src/main/java' and 'src/test/java'. The main editor window shows the code for 'Autolt.java', which is a Selenium WebDriver script. The code includes imports for Selenium WebDriver, WebDriverWait, and FirefoxDriver, and defines a 'main' method that initializes a WebDriver, navigates to a URL, clicks an element, and sleeps for 5000ms. The bottom screenshot shows a web browser displaying the ToolsQA website. The website has a navigation bar with 'HOME', 'SELENIUM TRAINING', 'DEMO SITE', and 'ABOUT'. The main content area features a large heading 'Selenium Certification Training | Enroll Now | Study Online' and a 'GO TO REGISTRATION' button. Below this, there is a section titled 'Instructor-Led Online Selenium Certification Training' with a brief description of the training.

```
1 package Assisted_project;
2 import java.io.IOException;
3 import java.util.concurrent.TimeUnit;
4 import org.openqa.selenium.By;
5 import org.openqa.selenium.WebDriver;
6 import org.openqa.selenium.firefox.FirefoxDriver;
7 public class Autolt {
8     private static WebDriver driver = null;
9     public static void main(String[] args) throws IOException, InterruptedException {
10         driver = new FirefoxDriver();
11         driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
12         driver.get("http://toolsqa.com/automation-practice-form");
13         driver.findElement(By.id("photo")).click();
14         Runtime.getRuntime().exec("D:\\Autolt\\AutoltTest.exe");
15         Thread.sleep(5000);
16         driver.close();
17     }
18 }
19
20 }
21
22
23
```

ToolsQA - Selenium Training

https://toolsqa.com/selenium-training?q=headers

TOOLSQA TUTORIALS

HOME SELENIUM TRAINING DEMO SITE ABOUT

Search

Selenium Certification Training | Enroll Now | Study Online

[GO TO REGISTRATION](#)

Instructor-Led Online Selenium Certification Training

TOOLSQA's Selenium Certification will help you in mastering the various concepts of Selenium from scratch.

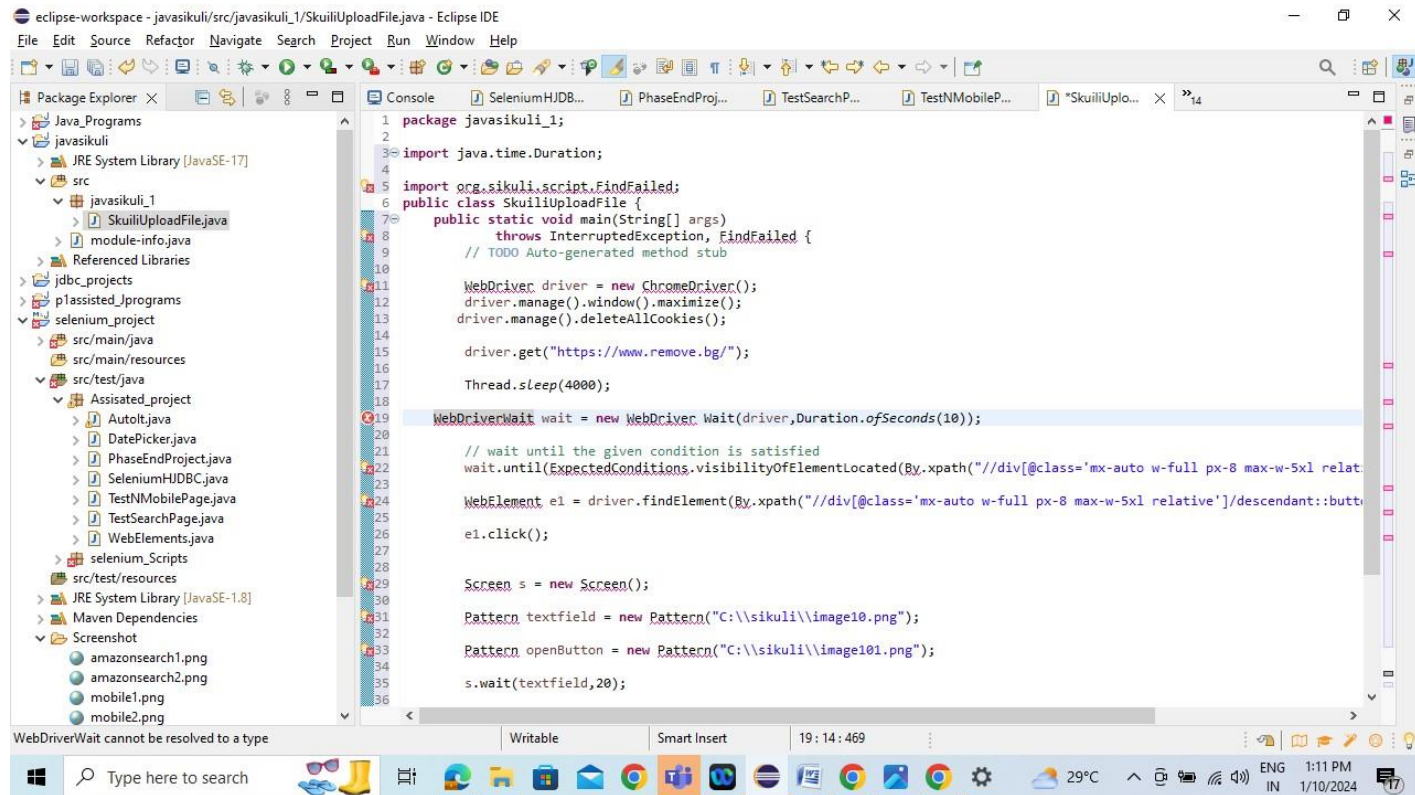
This Selenium Certification will help you master important concepts such as Selenium Commands, XPath Strategies, Waits, Iframes, Alerts, Windows in Selenium WebDriver, TestNG, Log4j, Apache POI, Framework, Designing and so

ASSISTED PROJECT – 11

EMP ID-2587325

Sikuli for UI Testing.

NAME: HIRANMAI



ASSISTED PROJECT – 11

EMP ID-2587325

Selenium and JDBC

NAME: HIRANMAI

