# CSharp Selenium - Python - Session Notes

0201

Stage 1 - Programming language - C#

Stage 2 - Selenium (Web automation) & FLAUI or AutoIT (Windows Automation)

Stage 3 - Hybrid Framework (NUnit Framework, Data Driven Framework, Page Object Model, Keyword Driven Framework)

—-----------------------------------------------

Stage 4 - Python

Stage 5 - Playwright (Web automation)

Stage 6 - Hybrid Framework (Pytest Framework)

—-----------------------------------------------

Stage 7 - Git & Jenkins

—------------------------------------------------

Timing - 9 to 1

Break → 10:45 to 11

Session Notes, C# & Selenium Project - <https://github.com/balaji-githubstore/csharp-selenium-siemens-oct-2025.git>

Framework - <https://github.com/balaji-githubstore/csharp-hybrid-framework-siemens-oct-2025.git>

Python Pytest

<https://github.com/balaji-githubstore/python-pytest-siemens-oct-2025.git>

Selenium - <https://www.selenium.dev/>

* Automates only the web application
* Open Source
* Language Independency - Java, C#, Python, Javascript, Ruby

Selenium

1. Selenium IDE
   1. No need for programming knowledge
   2. Record and playback
   3. Plugin - chrome, firefox, edge
   4. Use only for simple scripting and exploratory testing
2. Selenium RC - Depreciated
   1. Architecture

Source code (C#+Selenium RC libs) → RC server (Turn ON/OFF) → Browser

1. Selenium WebDriver (>3.0)
   1. Architecture

Source code (C#+Selenium WebDriver libs) → Browser

1. Selenium Grid
   1. If you want to scale by distributing and running tests on several machines and managing multiple environments from a central point

### C# - Programming

1. Installation
   1. Download Visual studio IDE (Latest community)
   2. Open Visual Studio Installer
      1. Select .Net Desktop development
2. Architecture
   1. Console app

Source code (.csproj) → .dll/.exe → O/P

* 1. Class library/Unit Test project

Source code (.csproj) → .dll → O/P

.dll → Platform independent

C# Compiler → helps to convert Source code to the Microsoft intermediate layer (.dll)

CLI - Common Language Runtime → helps the operating system to understand the .dll/.exe

Compile time error → Source code (.csproj) → .dll/.exe

Run time error → .dll/.exe → O/P

1. Create a Console app project
2. Naming convention

UpperCamelCase - MyFirstProject

lowerCamelCase - myFirstProject

1. Structure of C#

Solution - UpperCamelCase

Project 1 - UpperCamelCase

Namespace - UpperCamelCase

Class - UpperCamelCase

Methods - UpperCamelCase

Variable - lowerCamelCase

Properties - UpperCamelCase

1. Datatypes
   1. Pre-defined datatypes
   2. Non-defined datatypes - a collection of pre-defined
      1. String - a collection of char
      2. Array
         1. Size is fixed
      3. User-defined datatypes
2. When we give a whole number, it will be considered as an int

When we give decimal value, it will be considered as a double

1. Debugging
   1. Continue
   2. Terminate
   3. Step over
   4. Step into
2. Conditional statements
   1. If
   2. Switch
3. Iterative statements
   1. For
   2. Foreach
   3. While
   4. Do while

1. Methods - Building block of the program
   1. Reusability
   2. Maintainence
2. Static Methods
   1. How to create a static method and call it?

//accessmodifier static returntype MethodName(arguments)

To call static method

Classname.MethodName()

1. Non-Static Methods
   1. How to create a non-static method and call it?
   2. To call the non-static method
      1. Create an object for that class
      2. objref.MethodName()
2. Variables
   1. Static Variable
      1. How to create and call it?
   2. Non-Static Variable
      1. How to create and call it?
3. Object
   1. Declaration
   2. Instanitiation - new
   3. Initialization
4. Class & Object
   1. Class - A class is a template, blueprint, or type from which objects are created
   2. Object
      1. An object is an instance of a class
      2. Every object has its own state (non-static variable) and behaviour (non-static method)
5. Access modifier
   1. Private - accessible within the class
   2. Internal - accessible within the assembly
   3. Protected - accessible through inheritance
   4. Protected internal - accessible within the assembly and also outside the assembly through inheritance
   5. Public - accessible anywhere
6. Properties in C#
   1. Encapsulation in C# is a fundamental principle of object-oriented programming that involves bundling data (fields) and the methods that operate on that data into a single unit
7. Constructor - Pre-requisite to the object
   1. Constructor name and class name should be the same. It is a kind of method with no return type, not even void.
   2. It gets called whenever you create an object.
   3. There will always be a default constructor that helps to load all non-static variables with default values.
   4. We can override the default constructor by creating our own constructor
      1. Without arguments
      2. With arguments
   5. If a constructor is explicitly created, then it must be called during object creation.
8. this
   1. this keyword - helps to distinguish between the instance and local variable
   2. this keyword - points to the current object
9. Constructor overloading/ static polymorphism/compile-time polymorphism

The constructor to be called is resolved during compile time

Can create multiple constructors by changing in

* 1. Number of arguments
  2. Datatypes of arguments
  3. Sequence of arguments

1. Method overloading/static polymorphism/compile-time polymorphism

The method to be called is resolved during compile time

Can create multiple methods with the same name by changing in

* 1. Number of arguments
  2. Datatypes of arguments
  3. Sequence of arguments

1. Inheritance
   1. Reusability
   2. Maintenance

* When you want to reuse the methods and fields between the classes then we can implement inheritance
* While inheriting any class, if parent class contains only constructor with argument then we need to call that explicitly from child using base

1. Method overriding /Dynamic polymorphism / Run-time polymorphism
2. Abstract class
3. Interface
4. Collections
5. Var vs dynamic
   1. var → assign datatype during compile time
   2. dynamic → assign datatype during run time

var a = 10; //a will be registered for int // datatype will be autoassigned during compile time

a = 30;

var b = "hello"; //b is string

Console.WriteLine(b.ToUpper());

var res= Math.Sqrt(64); //res is double

res = 45;

res = 5.5;

var arr= DataSource.InvalidLoginData(); //arr as object[]

dynamic z = 10;

z = 2.4;

z = "kann";

dynamic b1 = "hello";

—----------------------------------------------------------------------------

### Selenium WebDriver

1. Create a Console app or NUnit test project

Unit test project → MSTest, XUnit, Nunit

1. Add Selenium WebDriver and Selenium support libraries from the NuGet package / Update all libraries if it is unit test project
2. Navigate to url
3. Get → title, url, page source
4. Click, type, Select
5. To inspect → tagname, attribute, text or not
6. Basic locator
   1. Id
   2. Name
   3. ClassName
   4. TagName
   5. LinkText
   6. PartialLinkText

FindElement → picks the first web element when there are duplicate locators

1. Advance locators
   1. XPath
   2. CSS
2. For page load - wait for page load to complete

FindElement --> check for presence of locator in 0.5s/500ms

1. Synchronization
   1. Unconditional wait (from C#)

Thread.Sleep(8000); //wait for 8s → Not recommended

* 1. Conditional wait (From selenium lib)
     1. Implicit wait
        1. Default implicit wait - 0s
        2. Applicable for all **FindElement** and **FindElements** methods
        3. Example: Implicit wait - 30s
           1. If the element is not present, it will check for 30 seconds, and then an exception will be thrown
           2. If the element is present, it will move ahead with the operation immediately
           3. Polling time → 0.5 (how freq it checks for elements within the 30s)
     2. Explicit wait
        1. Exact conditions
        2. Polling time → 0.5
     3. Fluent wait
        1. Ignore exceptions
        2. Modify polling time

1. Dropdown
   1. With select (Class - SelectElement)
      1. SelectByText()
      2. SelectByValue()
      3. SelectByIndex() → starts at 0
   2. Without select
      1. Click()
2. Multiple tabs/windows, frame, alert - SwitchTo()
3. Multiple tabs/windows
   1. driver.WindowHandles → gets all session id
4. Close() vs Quit()

driver.Close(); //close the current tab and current session

driver.Quit(); //close the current browser and also kill the process associated with it

1. Frame - embedding one HTML into another HTML
   1. Even though the locator is correct, we get OpenQA.Selenium.NoSuchElementException:
   2. Check for tagname - frame or iframe
   3. Switch to frame (any one option)
      1. Index
      2. String (name or id)
      3. IWebElement
2. To inspect → f12 or ctrl+shift+c

1. Alert
   1. Javascript alert
2. Upload
   1. Check for //input[@type='file']
3. Actions
   1. May not work on headless sometimes
   2. May not throw proper error
   3. Do not disturb the mouse/keyboard

Modifier keys - ctrl, alt, shift → KeyDown() and KeyUP()

Remaining keys - SendKeys

1. CSS Selectors
2. Javascript
   1. To click on hidden elements
   2. To work on a read-only textbox
   3. To scroll the page

Option 1 - JavaScript - Click & type

document.querySelector('#authUser').click()

document.querySelector('#authUser').value='hhhhhashhas'

document.querySelector('#authUser').scrollIntoView()

Options 2 - JavaScript and IWebElement - Click & type

IWebElement ele1= driver.FindElement(By.XPath("//div[@id='book-flight']//input[@name='dateDeparture']"));

driver.ExecuteJavaScript("arguments[0].value='12 Dec 2023'", ele1);

1. JavaScript - fetch output from javascript

string value = driver.ExecuteJavaScript<string>("return document.querySelector(‘#email’).value");

Console.WriteLine(value);

1. Chromeoptions - presetting
2. Shadowroot element

—----------------------------------------------------------------------------

### Framework - C#

1. Unit Test Framework - NUnit
2. Data Driven Framework (Separating the [Test] methods and test data in different files)
3. Page Object Model

Implement a keyword-driven framework for effectively handling the web elements

Steps to create a framework

1. Create a new solution with nunit project
2. Update and add dependencies from nuget package manager
   1. Selenium WebDriver (WebDriver.dll)
   2. Selenium support (WebDriver.Support.dll)
   3. ClosedXML (Working with excel)
   4. Microsoft.Extension.Configuration
   5. Microsoft.Extension.Configuration.Binder
   6. Microsoft.Extension.Configuration.Json
3. Add Test folder, test class and [Test] method
4. Every [Test] should have at least one assertion. Assertion decides whether the [Test] passes or fails
5. Nunit attributes
   1. [Setup] → Runs before each [Test]
   2. [Teardown] → Runs after each [Test], even if [Test] fails
6. Used inheritance concepts for the browser configuration - AutomationWrapper class
7. Data Driven Framework - Separating the [Test] method from test data. So we can drive the [Test] with a different set of test methods
8. Can use Nunit attributes [TestCase] or [TestCaseSource]
9. Data Driven Activity: [TestCase]
   1. Create a [Test] with proper arguments and use it inside the method properly
   2. Use [TestCase] to supply the test data

[TestCase("john","john123", "Invalid credentials")]

1. Data Driven Activity: [TestCaseSource]
   1. Create a [Test] with proper arguments and use it inside the method properly
   2. Create a static method that returns object[]. Load all the test set inside the object[]
   3. Connect the [Test] with a static method using [TestCaseSource]
2. Excel Read
   1. Add ClosedXML dependencies from nuget package manager
   2. To read excel
      1. Open Excel
      2. Sheet
      3. Range
      4. Row & Column
3. Connect Excel with [TestCaseSource]
   1. Create a [Test] with proper arguments and use it inside the method properly
   2. Add the data to the excel based on the arguments supplied
   3. Create a static method that returns object[] from the GetSheetIntoObjectArray(string filePath,string sheetname)
   4. Connect the [Test] with a static method using [TestCaseSource]
4. Page Object Model - Design pattern
   1. Reuse
   2. Maintenance
   3. Readability

To implement the page object model

1. For each web page, we need to create a class - Page class
2. Operations/Actions should happen through methods - Page methods
3. Collecting the object repository (locators) at the class level
4. Keyword Driven Framework - WebDriver keywords
5. Reports
6. Configurations
7. Control using command line
   1. Run test groups
   2. Run all
   3. Run specific methods

—----------------------------------------------------------------------------

### GIT:

Git - Git is a free and open source distributed version control system

Architecture

Project/Solution (in local machine) --> local repository (in local machine) → remote repository (GitHub, GitLab, AWS CodeCommit, Bitbucket)

Git

Modified–staging–commit

Working with the git concepts (to push the code to GitHub)

1. git init → initialize the local repo (every solution will have its own local repo in C#)
2. git add . → stage the file to be committed
3. git commit -m “first commit” → update the local repo
4. git remote add origin1 <https://github.com/balaji-githubstore/csharp-selenium-siemens-oct-2025.git> → register the remote url with name origin
5. git push -u origin1 master

—-----------------------

Python

1. Install Python
   1. Add/verify below path to environment path variable
      1. C:\Program Files\Python314
      2. C:\Program Files\Python314\scripts
2. Datatypes
   1. Numeric
      1. Integer
      2. Float
   2. Sequence
      1. String
      2. List
      3. Tuple
         1. Faster
         2. Contains constant set of data (immutable)
   3. Dictionary
   4. Boolean
3. Conditional and iterative statements
4. Module
   1. Methods
   2. Variable
5. Class and Objects
   1. Static vs non-static variable
   2. Static vs non-static methods
   3. Property
6. Working on imports

Python - Playwright

1. Install playwright via pip
   1. pip install playwright
2. Playwright → Browser → context → page (tab)
3. Click, type, select
4. Auto wait
   1. <https://playwright.dev/python/docs/actionability>
5. Multiple tabs/windows, frame, alert
6. Frame
   1. Use page.frame\_locator()
7. Multiple tabs/windows
8. Alert
9. Shadow root
   1. Use css selector
10. Upload
11. CSS Selector
    1. CSS selectors pierce open shadow DOM.
    2. Playwright adds custom pseudo-classes like :visible, :has-text(), :has(), :is(), :nth-match() and more.

*# using text to find*

page.locator("text=Create an account").click()

page.locator("a:has-text('Create an account')").click()

page.locator("a",has\_text='Create an account').click()

Framework in Python - using pytest

1. Install libraries
   1. pip install pytest
   2. pip install assertpy
   3. pip install pandas
2. Create test module, class, methods using pytest
3. Add fixtures for browser configuration
4. Add AutomationWrapper for inheritance
5. Data Driven Framework using - pytest.mark.parametrize(

Reference

1 GB - 1024 MB

1 MB - 1024 KB

1 KB - 1024 B

1 B - 8 bits

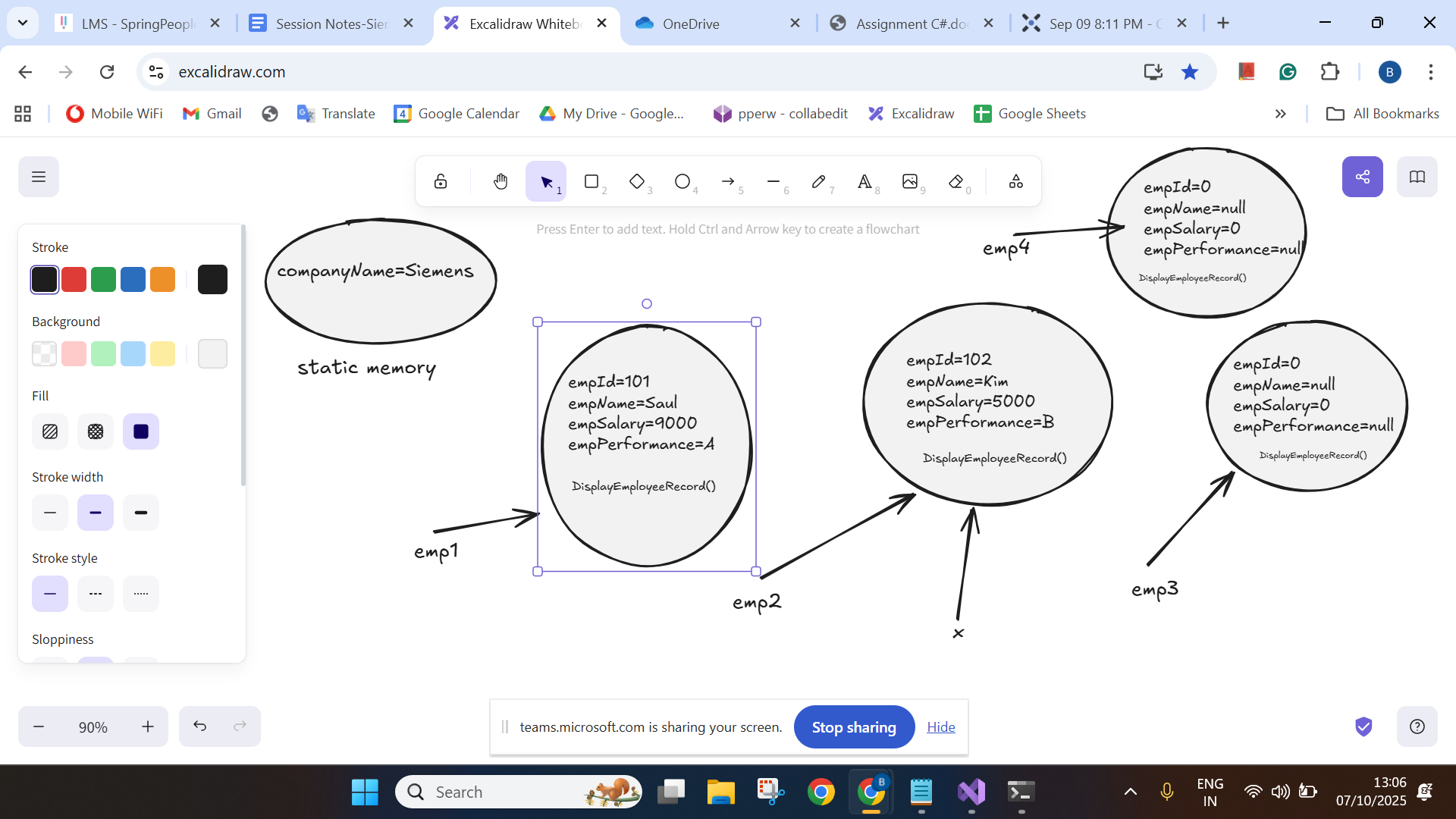
1 bit - 0 or 1

Exception in C#

1. System.IndexOutOfRangeException:

Exception in Selenium C#

1. NoSuchElementException
2. InvalidSelectorException - wrong syntax of the locator
3. ElementClickInterceptedException: - Some other element is hiding the target element.
4. ElementNotIteractableException -
5. NoSuchWindowException: 'no such window: target window already closed
6. OpenQA.Selenium.NoAlertPresentException : no such alert
7. OpenQA.Selenium.StaleElementReferenceException: 'stale element reference: stale element not found



# Assignments:

# **Task 1(Important)**

1. Navigate onto <https://www.salesforce.com/in/form/signup/freetrial-sales/>
2. Enter first name as “John”
3. Enter last name as “wick”
4. Enter work email as “john@gmail.com”
5. Select Job title as “IT Manager”
6. Select Employees as “101-500 employees”
7. Select country as “United Kingdom”
8. Do not fill the phone number
9. Click on check box
10. Click on start my free trial
11. Get the error message displayed “Enter a valid phone number”

# **Task 3 (Important)**

1. Navigate onto https://www.medibuddy.in/

2. Close if any popup and Click on Login

3. Click on I have Corporate Account

4. Click on Learn More

5. Click on Skip

6. Click on Login using Username & Password

7. Enter username as john

8. Enter password as john123

9. Click on show password

10. Click log in

11. Get the error message shown and print it in terminal

# **Task 4**

1. Navigate onto https://nasscom.in/

2. Click on Login and then click on register

3. Enter First name as admin

4. Enter Last name as pass

5. Enter email address as admin@gmail.com

6. Enter company name as Google

7. Select IT Consulting from dropdown

8. No need to automate CAPTCHA

9. Click on Register

### Multiple tabs

# **Task 1 (Important) - Multiple tabs**

1. Navigate onto https://www.citigroup.com/global/about-us/global-presence/india

2. Close if any pop up comes

3. Click on My Account

4. Click on Banking with citi

5. In new tab

6. Enter userid as john123

7. Click on signup

8. Get the error displayed for password

# **Task 2**

1. Navigate to<https://secure1.inmotionhosting.com/index/login>

2. Click on visit our support center

3. Click on “No Thanks” if any popup

4. Search for “diskspace”

5. Get the text “Search Results for: diskspace” and print it

Task 3

1. <https://eq-sol-ops-us-fd-main.azurefd.net/informational/contact-us/>
2. Fill the form with some random data and clear it off

Frame

# **Task 1 (Important)**

1. Navigate onto http://demo.openemr.io/b/openemr/

2. Update username as admin

3. Update password as pass

4. Select language as English (Indian)

5. Click on the login button

6. Click on Patient Click New Search

7. Add the first name, last name

8. Update DOB as today's date

driver.findElement(By.id("form\_DOB")).sendKeys("2024-01-12");

9. Update the gender

10. . Click on the create new patient button above the form

11. . Click on confirm create new patient button.

12. . Print the text from alert box (if any error before handling alert, add 5 5-second wait)

13. . Handle alert

14. Close the Happy Birthday popup

15. Get the added patient name and print it in the console.

Final Task

# **Task 1 (Important)**

1. <https://www.malaysiaairlines.com/us/en/home.html>

# **Task 2 (Important) - shadowroot element**

1. Navigate onto https://www.royalcaribbean.com/account/signin

2. Click Create an account

4. First Name as Dennis

5. Last Name as Rich

6. Select Month as April

7. Day as 4

8. Year as 1990

9. Country as India

10. Any email address and password

11. Select as “What was your first car's make or model?”

12. Type answer

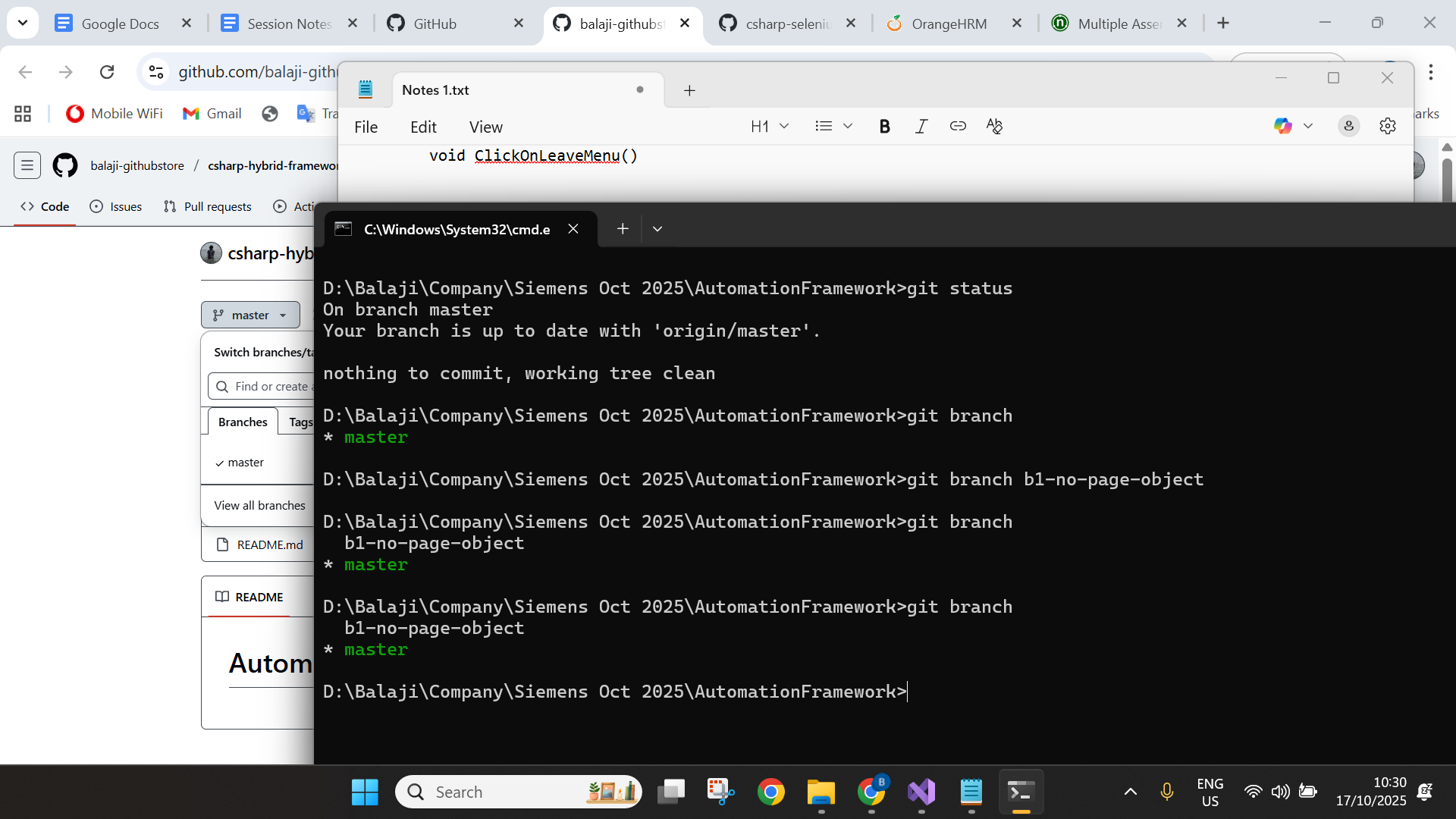
13. Accept the terms and condition

14. Click done

—

Created a branch **b1-no-page-object** on Oct 17, 2025, at 10:29 AM IST → It will create a copy from the existing checkout branch (we created a copy from master branch)

**B2-no-webdriver-keywords** on Oct 17, 2025, at 12:22 AM IST (created a copy from checkout branch which is master)



sherin.thomas@siemens.com