**Exercise: Selenium IDE**

# **Number Calculator**

Record and/or manually add selenese commands, where needed to test the following functionalities of a Number Calculator.

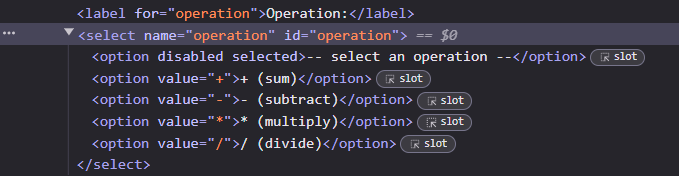
[**http://softuni-qa-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/number-calculator/**](http://softuni-qa-loadbalancer-2137572849.eu-north-1.elb.amazonaws.com/number-calculator/)

### **Input Validation**

* Ensure that the input fields for "First Number" and "Second Number" accept only numerical values.
* Verify that appropriate error messages are displayed when non-numeric values are entered.
* Check the application's behavior with missing input values (e.g., only one number entered).
* Test the application's behavior with both input fields left empty.

### **Operation Selection**

* Verify that the dropdown menu allows the selection of each operation:
  + Sum
  + Subtract
  + Multiply
  + Divide
* Think how to assert the value?



### **Calculation**

* Test the calculation functionality for each operation (addition, subtraction, multiplication, division) with valid inputs.
* Assert the correctness of the result displayed for each operation.

### **Reset Functionality**

* Ensure that the reset button clears all input fields and resets the operation selection.

### **Edge Cases**

* Test the application with edge cases:
* Dividing by zero (you will find a bug here; the correct text should read: Undefined but it's not)
* Test the application's response to negative numbers.
* Verify the application's handling of decimal numbers.
* Check if leading and trailing spaces in input fields are managed correctly.
* Try calculating with "Infinity".

# **CURA Health Care Service**

## Verify the Footer

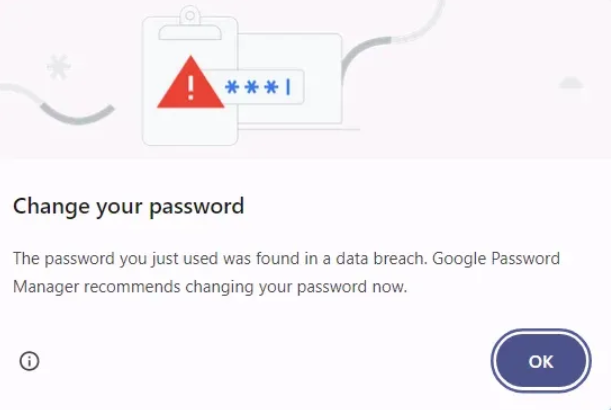
* Verify the text and elements in the footer of the page.
* Text is marked in Orange.
* Elements are marked in Green.

z

* For asserting the elements use "Present" command.  
  

## **Important:** Password breach warnings

**Google Password Manager**, a feature within **Chrome**, can interfere with your automated tests by displaying password breach warnings. These warnings, triggered by the use of hardcoded credentials, can disrupt the flow of your tests.



**You have three options for resolving the issue:**

### **Switch to Firefox Browser;**

**Install Selenium IDE on Firefox browser and run your tests there.**

### **Adding a pause in your test, so you'll have the time to close manually the pop-up;**

To manage the data breach pop-up, you need to manually pause the script execution.

**Follow these steps:**

* Determine the command after which the pop-up appears. In this case, it is after clicking the login button.
* Click on the number of the command that comes immediately after the command where the pop-up appears. For example, if the pop-up appears after clicking the login button, click directly on command number that follows the pop-up.



* Once you click on the command number, it should be highlighted with a blue bar, indicating that the script will pause execution at this step.

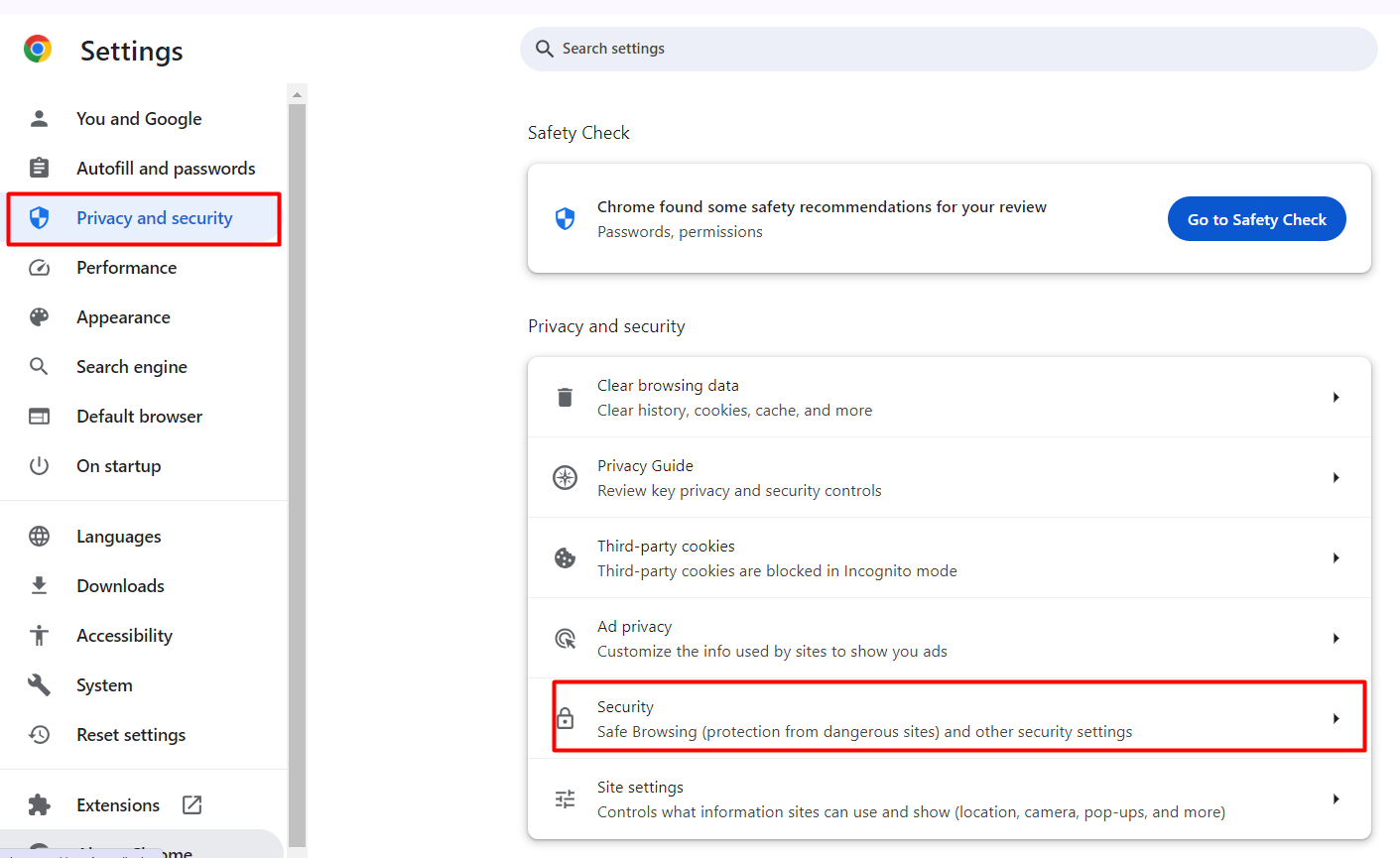


* Play the script. The script execution will pause at the highlighted command.
* While the script is paused, manually close the pop-up that appears.
* After closing the pop-up, resume the script execution, clicking on play button to continue with the remaining steps.

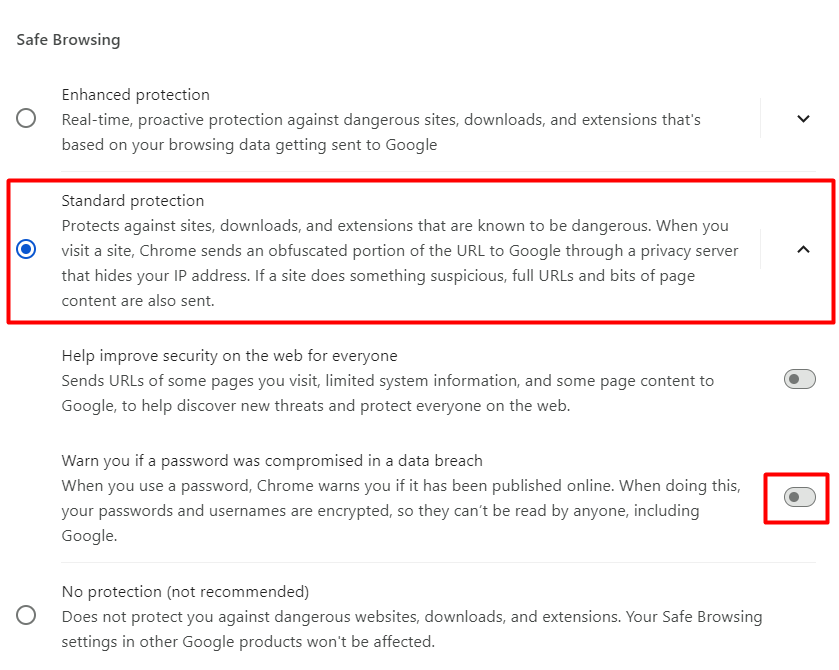
### **In your Chrome browser switch to "Standard protection" and disable these warnings temporarily.**

**If you choose the third option, follow the steps below:**

* Click on the three-dot menu in the top-right corner of Chrome.
* Select "Settings" from the dropdown menu.
* In the left-hand menu, click on "Privacy and security".
* Click on "Security" under the Privacy and security section.



* Under "Safe Browsing", select "Standard protection".
* This setting allows you to disable the password breach warning feature that pops up when you use hardcoded credentials for testing.
* Scroll down to the section "Warn you if a password was compromised in a data breach".
* Toggle the switch to turn off this warning.

a

**Remember**

**Switch Back After Testing**: For your security, remember to switch back to your preferred protection level after you finish testing.

**Educational Purpose:** This adjustment is only recommended for the duration of your learning and testing activities.

By following these steps, you can ensure a smoother and uninterrupted testing process while working with Selenium IDE on this website. **This issue is specific to the Chrome browser and its development with Google Password Manager, which is designed to enhance security but can interfere with testing scenarios.**

## Login and Logout

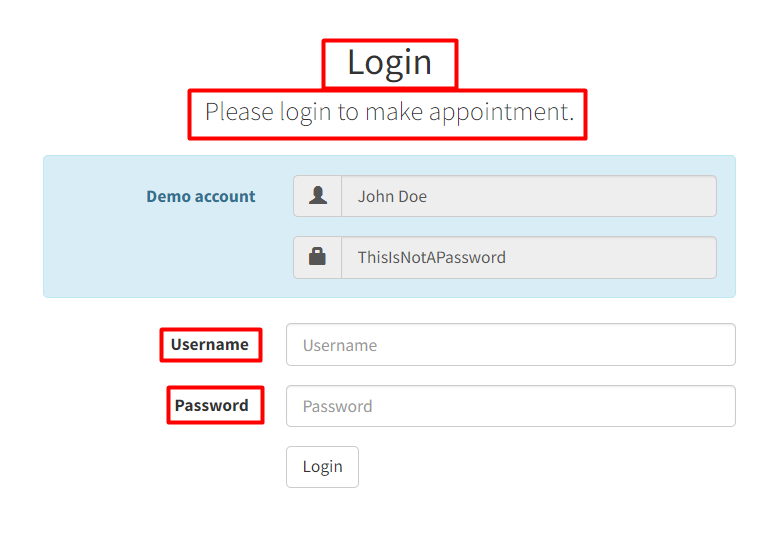
**Successfully log in and log out of the website.**

**Use commands that don't stop the script if a step fails.**

**Write the test manually. Don't use the record option.**

**Check the reference section for each command to see the explanation for what it does.**

* On the home page, check that the "Make Appointment" button is visible and that its text is correct.
* Go to the login page by clicking the "Make Appointment" button.
* On the login page, check these elements:
  + Login
  + Please login to make appointment.
  + Username
  + Password

a

* Log in with the correct details:
  + Username: John Doe
  + Password: ThisIsNotAPassword
* After logging in, verify that you are taken to a different page. You can choose any element on that page to check.
* Log out by clicking on the menu and selecting logout.
* After logging out, ensure you are redirected to the home page.
* Click the "Make Appointment" button again and confirm that it takes you to the login page.

**OPTIONAL:** Try using developer tools to copy text for text assertions directly from there. This will help you get used to using developer tools.

## Make an Appointment

**Use commands that will not stop the script, if a given step fails.**

**Write script manually. Without using record option.**

**Use dev tools.**

* Click on the button "Make Appointment".
* Log in.
* Verify that the title "Make an appointment" is displayed.
* In the Facility field, select "Hongkong CURA Healthcare Center".
* In "Healthcare program" select value None.
* Enter a date. (Type the date. Don't use the select option from the calendar. Trust me, it's easier.)
* Enter any value in the "Comment" field.
* Press the button "Book appointment".
* First, verify that you are on the meeting confirmation page. Use title "Appointment Confirmation".
* On the meeting confirmation page, verify that the values you previously selected are displayed there.
* Verify one by one:
* Does Facility have the selected value "Hongkong Cura Healthcare Center".
* Does "Apply for hospital readmission" have value "No".
* Is the Healthcare program set to "None".
* Is the date you selected the same.
* Is the comment you entered the same.
* Press the "Go to homepage" button.
* Verify that you have been correctly redirected to the home page.
* Log out.

# **Saucedemo**

Navigate to the URL: <https://www.saucedemo.com>

## \* Login with Invalid User and Retry

Create a test script that attempts to log in with an invalid username, checks the error message, and then retries with a valid username. Use conditional logic to handle the error scenario.

**Login Attempt with Invalid User:**

* Locate the username field and enter user123.
* Locate the password field and enter secret\_sauce.
* Click on the login button.

**Verify Error Message:**

* Assert that the error message displayed is: "Epic sadface: Username and password do not match any user in this service".

**Conditional Retry:**

* Hint: Use store text and if structure for this part.
* Use the if command to check if the error message is present.
* If the error message is found, print "Wrong username" and retry with valid credentials.

**Login Attempt with Valid User:**

* Locate the username field and enter standard\_user.
* Locate the password field and enter secret\_sauce.
* Click on the login button.

**Verify Successful Login:**

* Assert that the page title is "Products".
* Print "Successful login".

## Add item to the shopping cart

Create a test script that logs into the application, adds an item to the cart, and verifies the cart content.

Try this one by yourself.

## Remove item from the shopping cart

Create a test script that logs into the application, adds an item to the cart, verifies the cart content, and then removes the item from the cart.

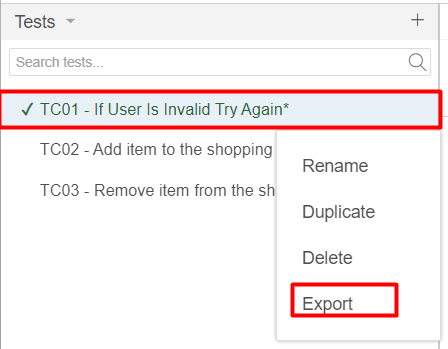
Try this one by yourself.

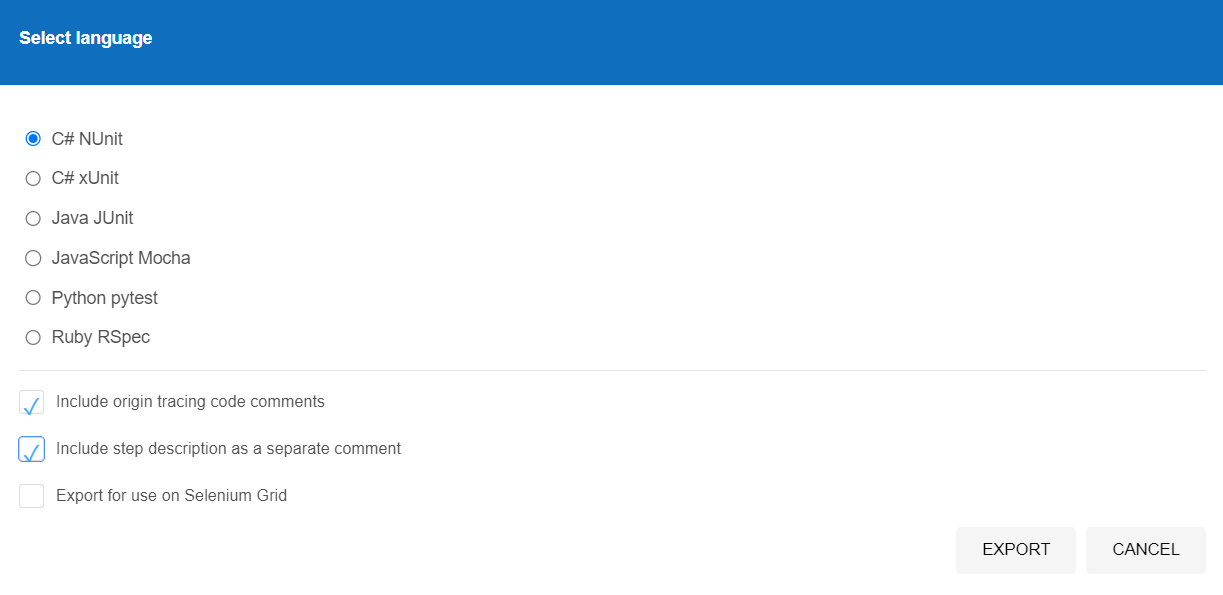
## Explore

<https://www.saucedemo.com/> is one of the best sites for practice. Explore it, if you feel like it.

# **Export Code and Run it in Visual Studio**

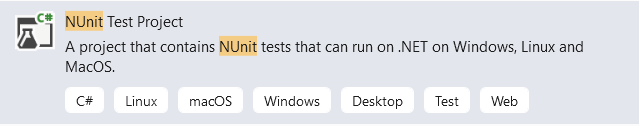
Let's export our "Login with Invalid User and Retry" Test and try to run it in Visual Studio.



a

**Setting Up Visual Studio:**

* Open Visual Studio.
* Create a new project by selecting File > New > Create a new project.
* Search for NUnit Test Project.

a

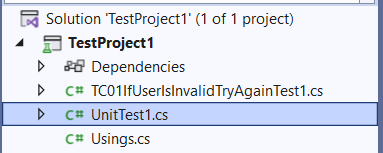
* Click Next, set the project name, and click Create.

**Adding Selenium Test Cases to the Project:**

* Right-click on the project in Solution Explorer.
* Select Add > Existing Item.
* Navigate to the location of the exported Selenium test suite and add it.

**Delete Unnecessary Class:**

* Delete UnitTest.cs



**Adding NuGet Packages:**

* Right-click on the project and select Manage NuGet Packages.
* In the Browse tab, search for **Selenium.WebDriver** and click Install.
* Search for **Chrome Web Driver** and click install.

**Rebuilding the Solution:**

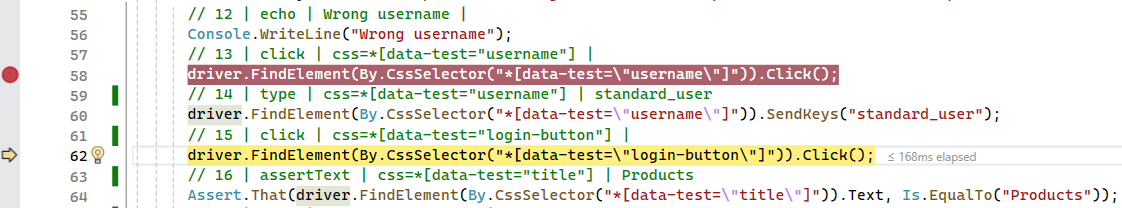
* After installing the necessary packages, rebuild the solution.
* Check the Output window for any errors.
* Check if the targeted URL is correct. Selenium IDE tends to add too many slashes /

**Running the Test Case:**

* Open the Test Explorer by going to Test > Test Explorer.
* Click Run to execute the test.

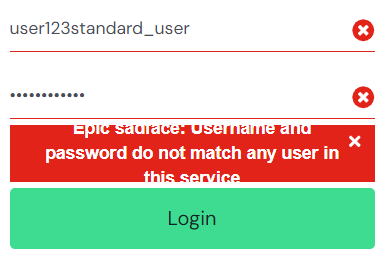
**The Test fails. Why?**

* Let's debug the test.



**What happens after the Wrong username was printed?**

* Since Selenium IDE doesn't have a clear command. Selenium WebDriver doesn't know that it has to clear the text field, before typing the correct username.

a

**We have to add this step manually in our code.**

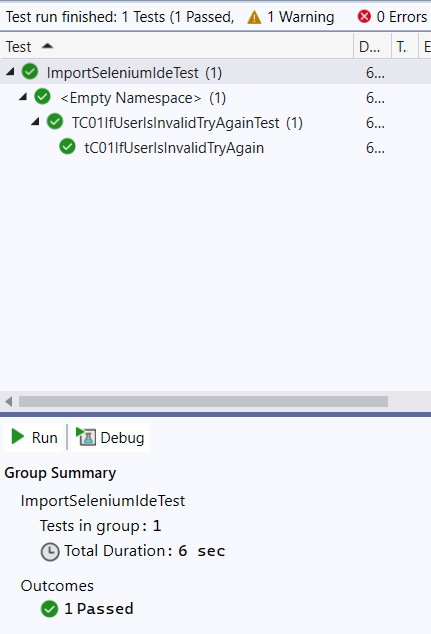
* You can replace this line of code:

**driver.FindElement(By.CssSelector("\*[data-test=\"username\"]")).Click();**

* With this one:

**driver.FindElement(By.CssSelector("\*[data-test=\"username\"]")).Clear();**

**Run the test again! 😊**

**a**