

Selenium Integrated Development Environment

David Corcoran



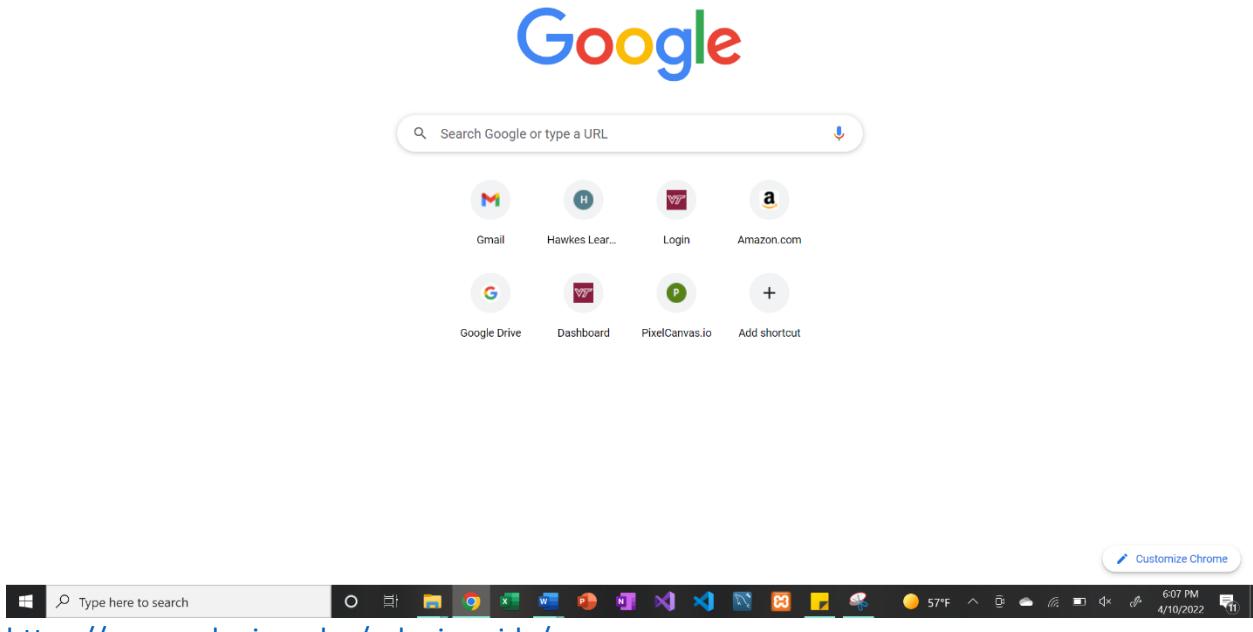
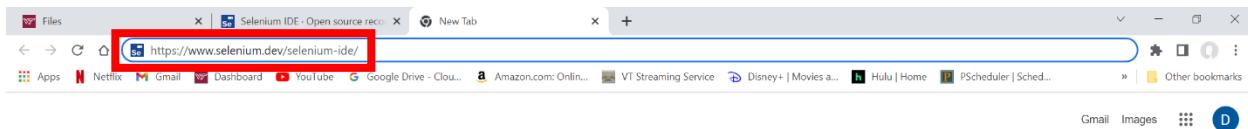
Table of Contents

Installation	3-4
Creating A Project	5-6
Saving a Project	7
Manipulating Automated Testing Files	
Creating Automated Tests.....	8-9
Renaming Automated Tests	10
Deleting Automated Tests	11
Test Suites	12-13
Recording a Test.....	14-22
Test Playback	23-29
Conclusion.....	30

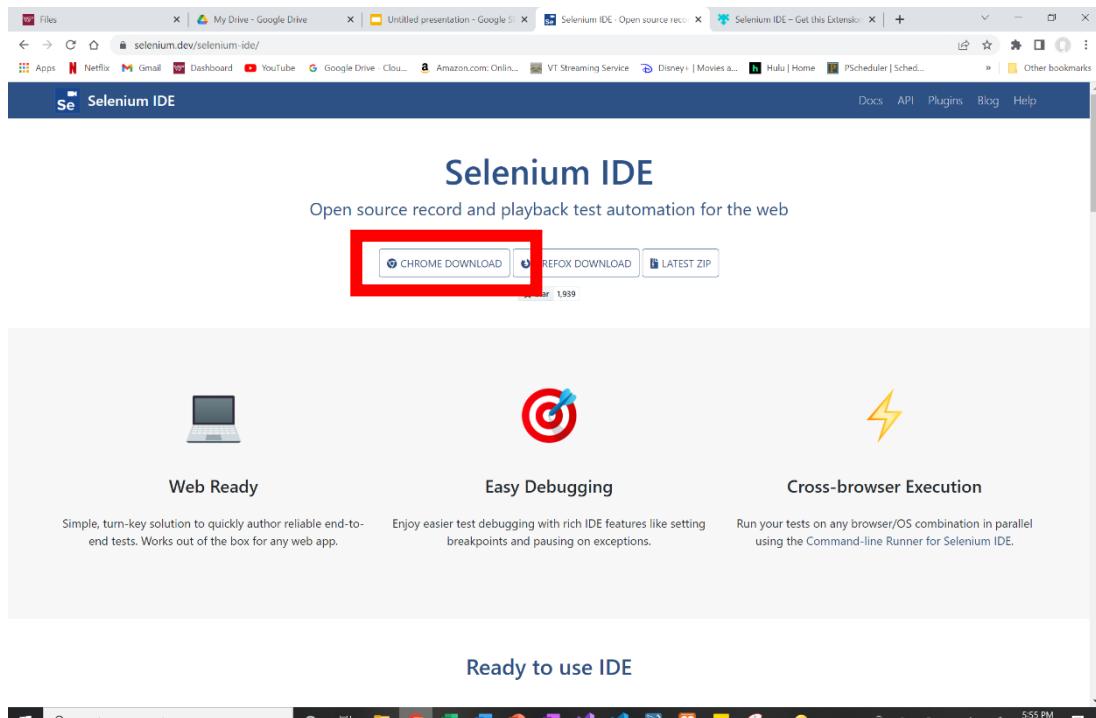
Installation

Selenium Integrated Development Environment (IDE) is an open source, record and playback automated testing software. It is implemented as a browser extension, available only on Mozilla Firefox and Google Chrome. For this demo, I will be using Chrome as my web browser. Selenium IDE allows its users to input and debug unique requirements tests on a pre-selected website. Steps for installation are as follows:

1. Open Google Chrome.
2. Navigate to <https://www.selenium.dev/selenium-ide/> in your selected browser.

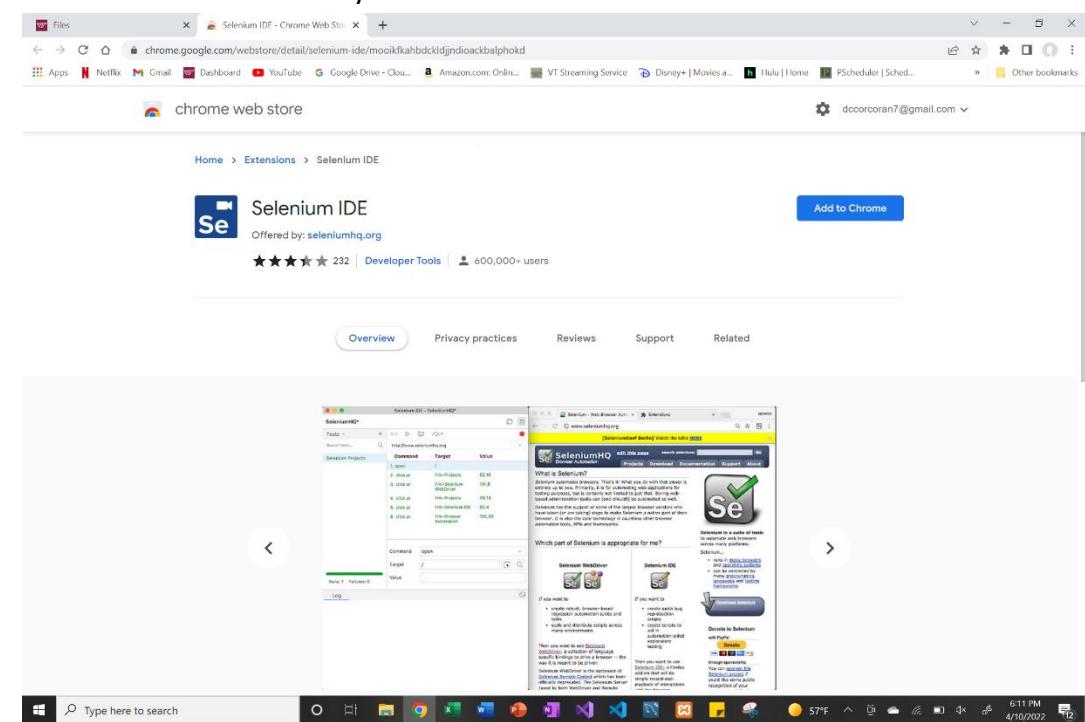


3. Click the “Chrome Download” button to be redirected to the Google Chrome Web Store.



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

4. Press the “Add to Chrome” button and confirm the action, allowing for the software to become an extension on your browser.

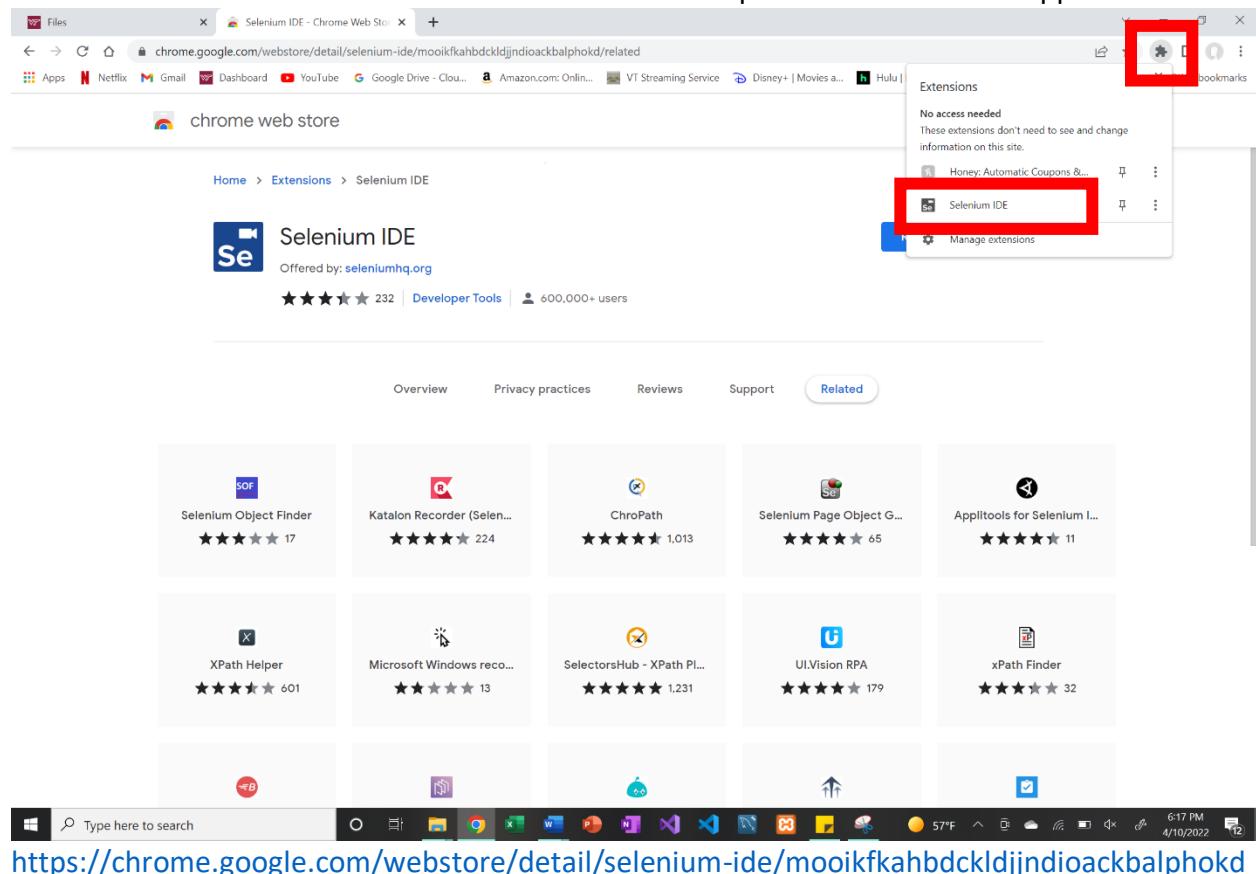


<https://chrome.google.com/webstore/detail/selenium-ide/mooikfahbdckldjndioackbalphokd>

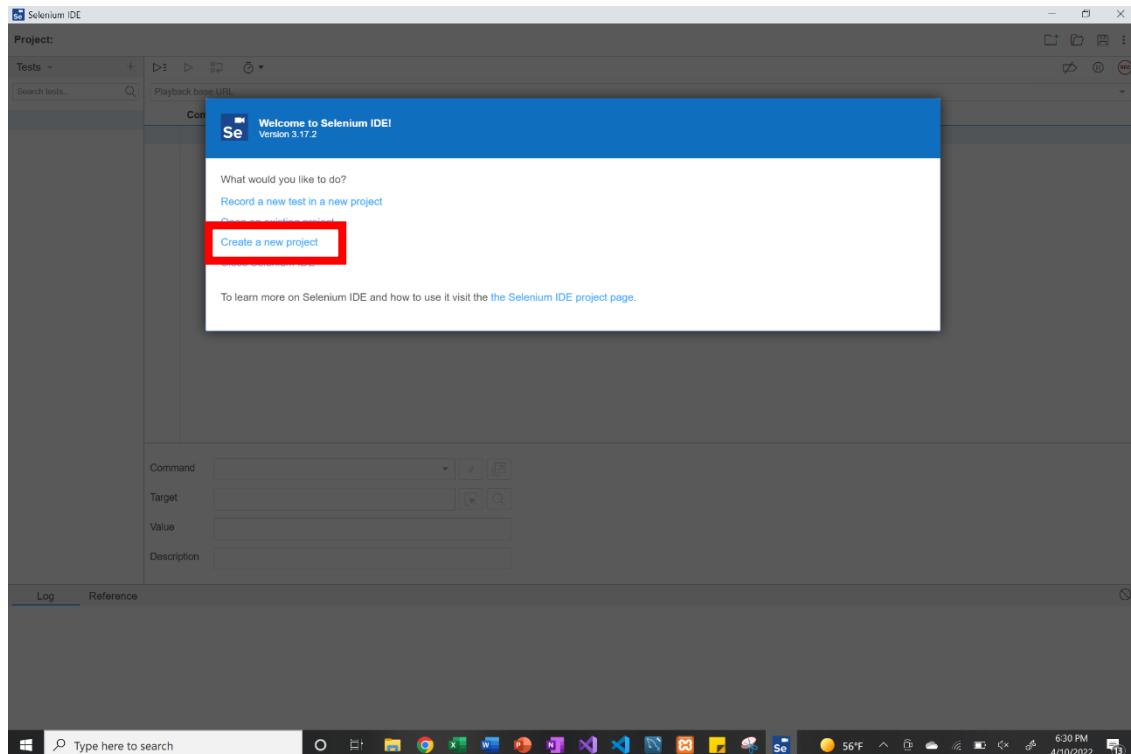
Creating A Project

Next, I will instruct how to open the Selenium IDE application and create a project file. To efficiently use Selenium IDE, the user must first create project, and subsequent file, where all related automated tests can be stored. Creating different project files can help organize a user's requirements tests, separating non-related tests to decrease confusion. Creating a project can be done as follows:

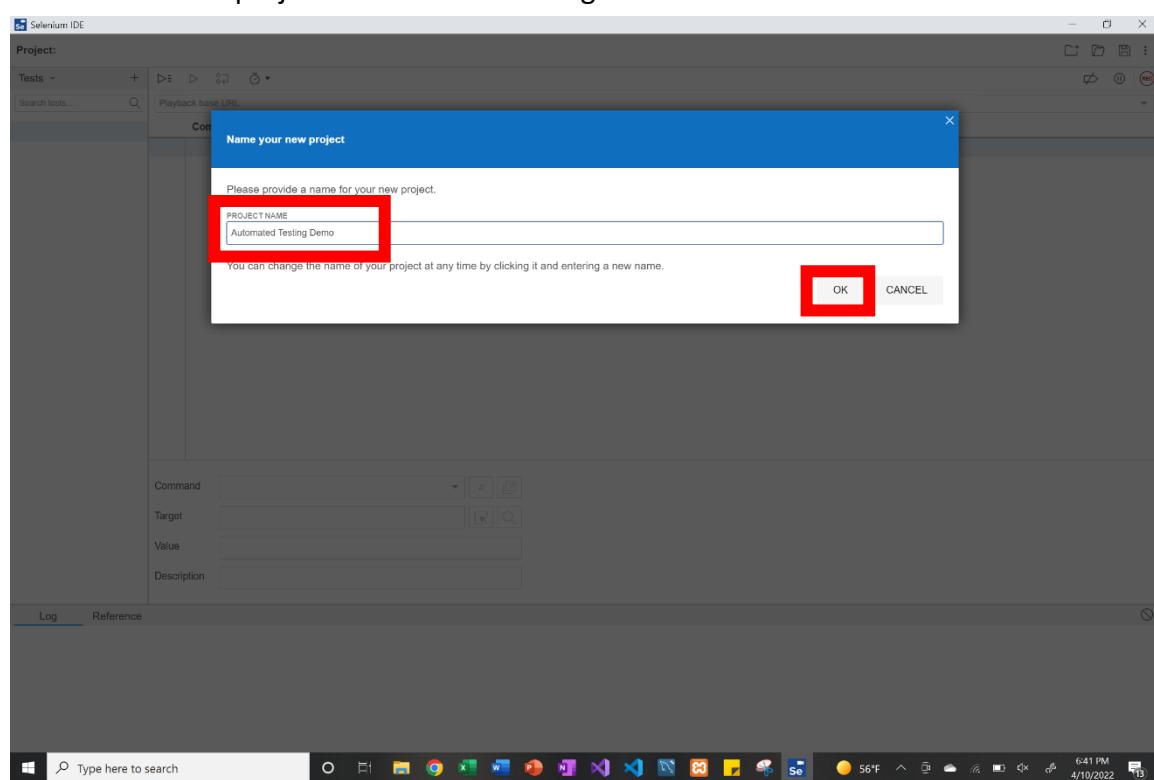
1. Click the puzzle icon in the top right of your screen. This is the hub for all the chrome extensions downloaded to your computer.
2. Click the button titled “Selenium IDE.” This will open the Selenium IDE application.



3. Click “Create a new project.” This will create a new project as a “.side” file. Because this program is completely new, you won’t have any existing project files to open.



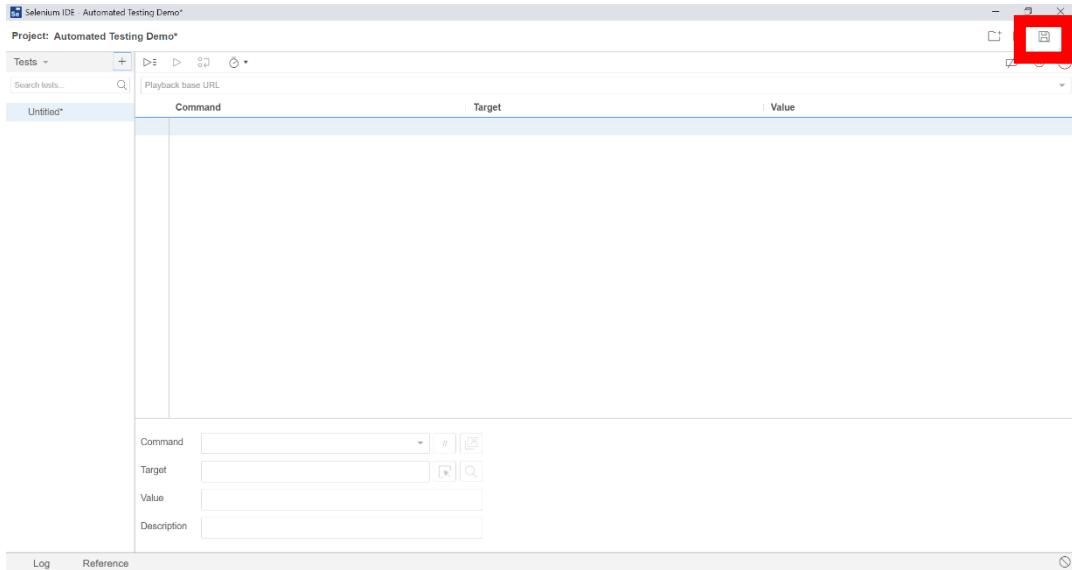
4. Title the new project “Automated Testing Demo” and click the “OK” button.



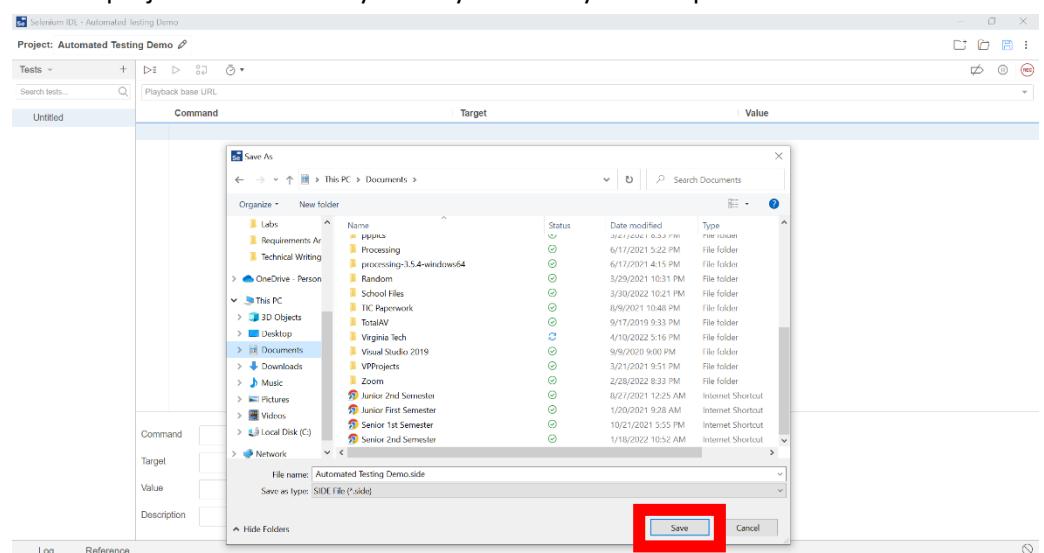
Saving A Project

Saving is important, as it allows the user to close the Selenium IDE application and return to work on it later. To save a project file, the instruction are as follows:

1. Click the save icon in the top right of Selenium IDE's application window



2. Save the project ".side" file anywhere you like in your computer files

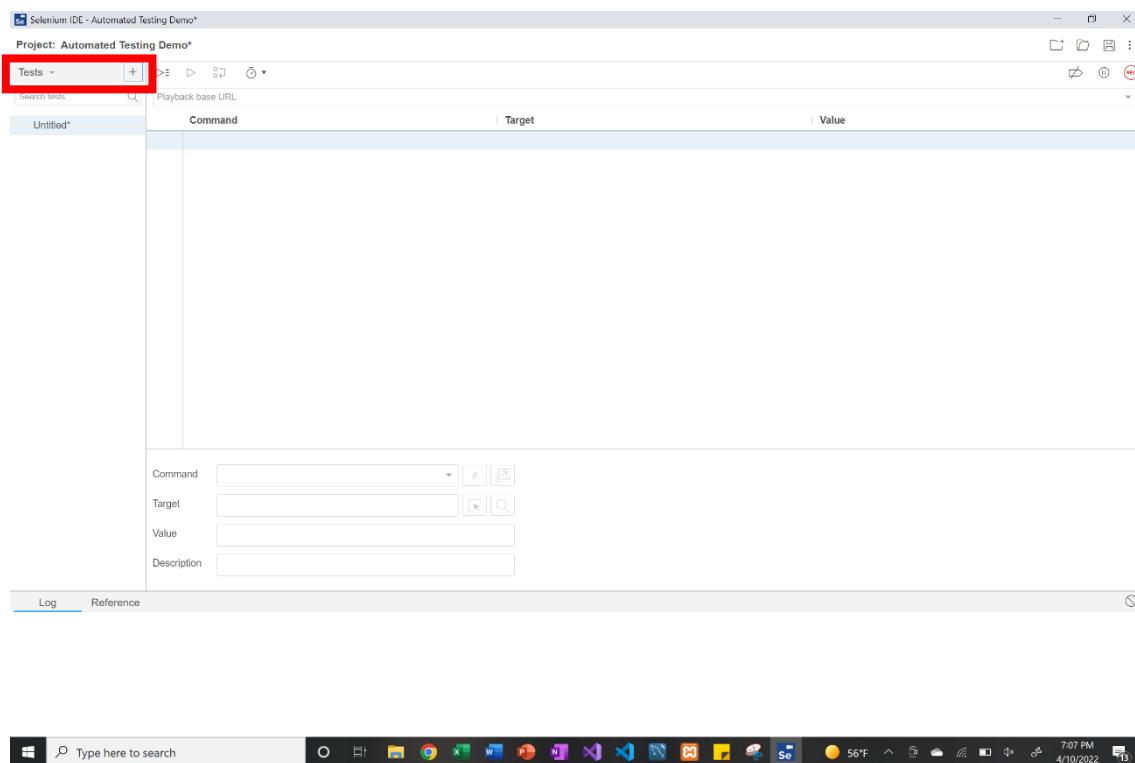


Manipulating Automated Testing Files

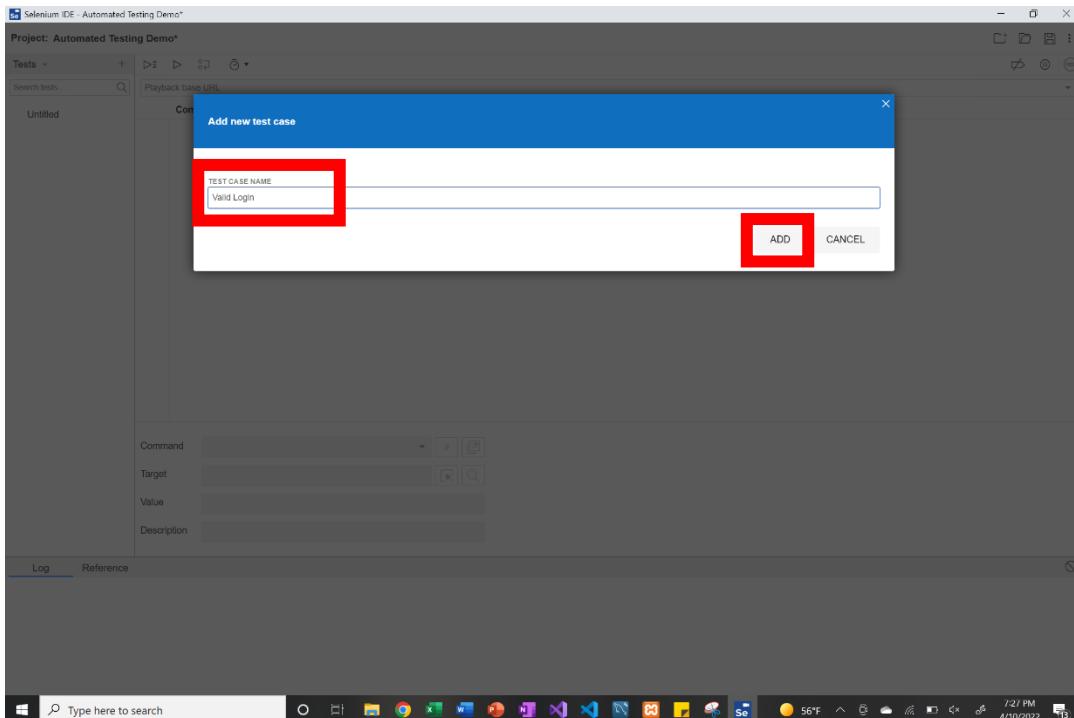
Next, I will provide instructions on how to create, rename, duplicate, delete, and save automated tests and test suites. A test suite is a collection of related automated test that the user has arranged in a collective group. The instructions on manipulating test files are as follows:

Creating Automated Tests

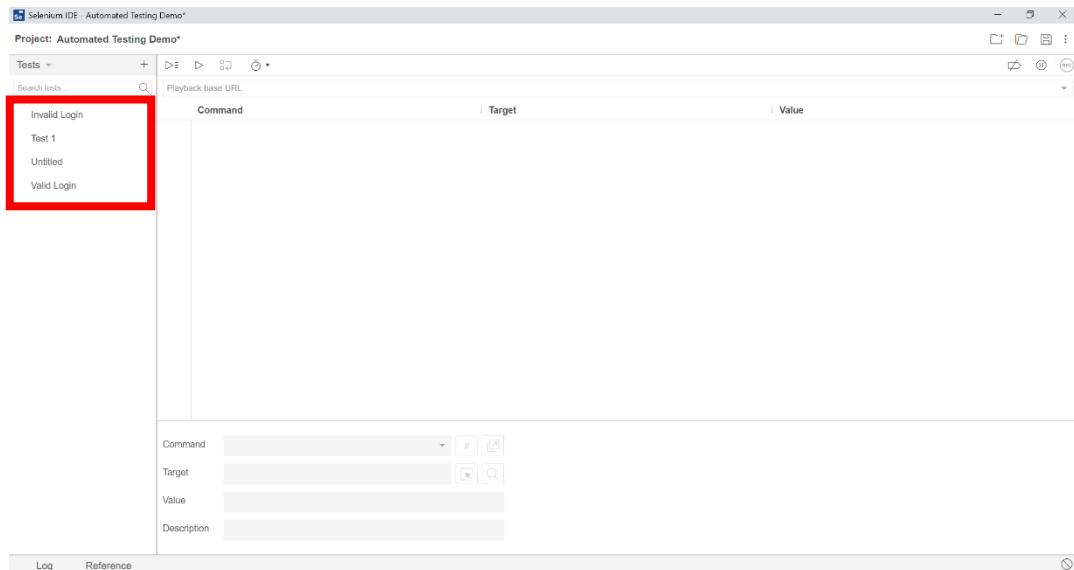
- When creating a new project in Selenium IDE, an “Untitled” first test is automatically created. However, to create a new automated test, press the plus button at the top left of the window.



2. Name the new automated test “Valid Login” and press the “Add” button. This test will automate a correct username/password combination and successfully admit the user into the system

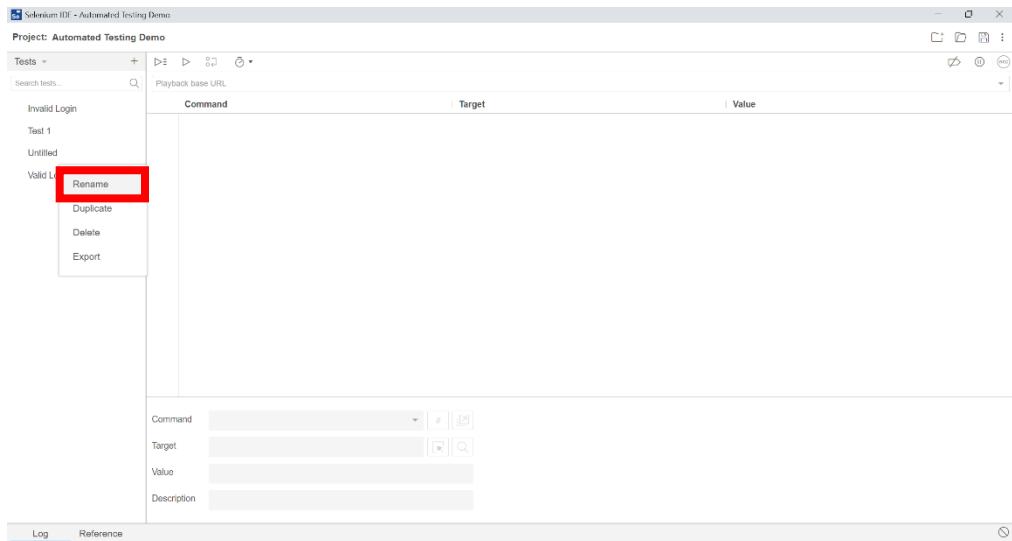


3. Create two more automated tests called “Invalid Login” and “Test 1.” “Invalid Login” will input incorrect username/password combinations, blocking the user from accessing the website. “Test 1” will be a throwaway test to demonstrate how to delete tests. New tests will appear in alphabetical order at the left side of the application window.

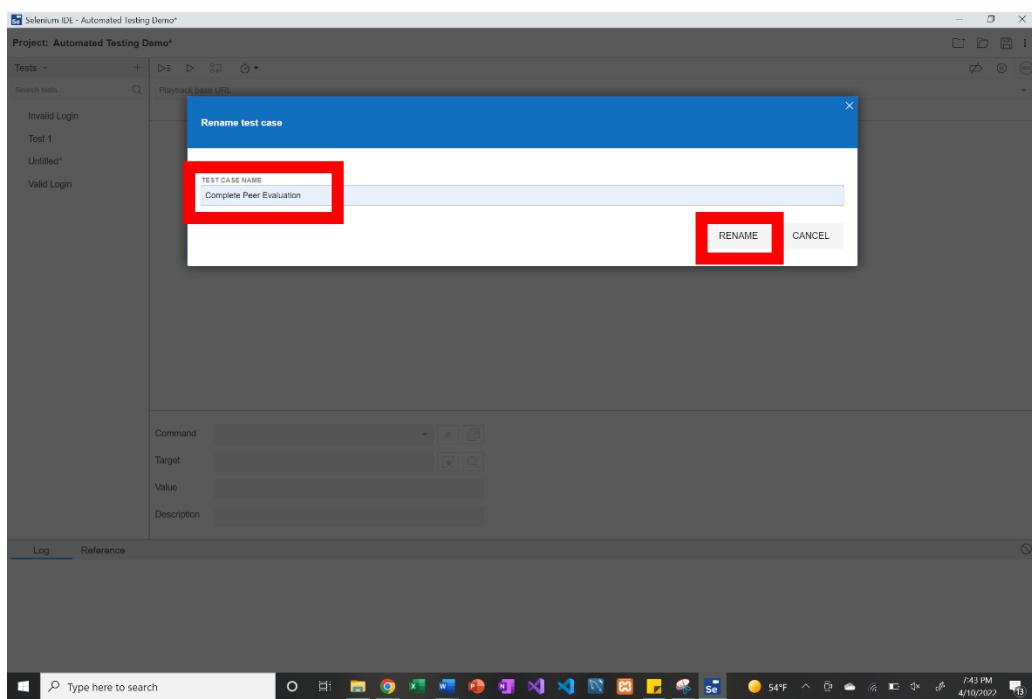


Renaming Automated Tests

1. Press the menu button for the “Untitled” test.
2. Press the “Rename” button on the dropdown to change the title of the test.

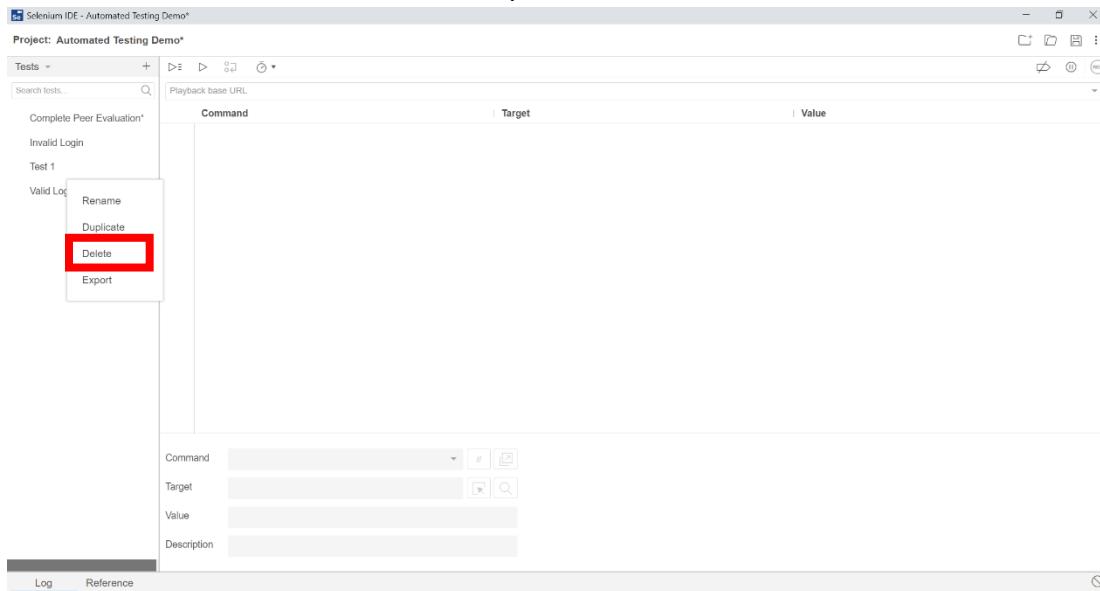


3. Rename the test to “Complete Peer Evaluation” and press the “Rename” button.

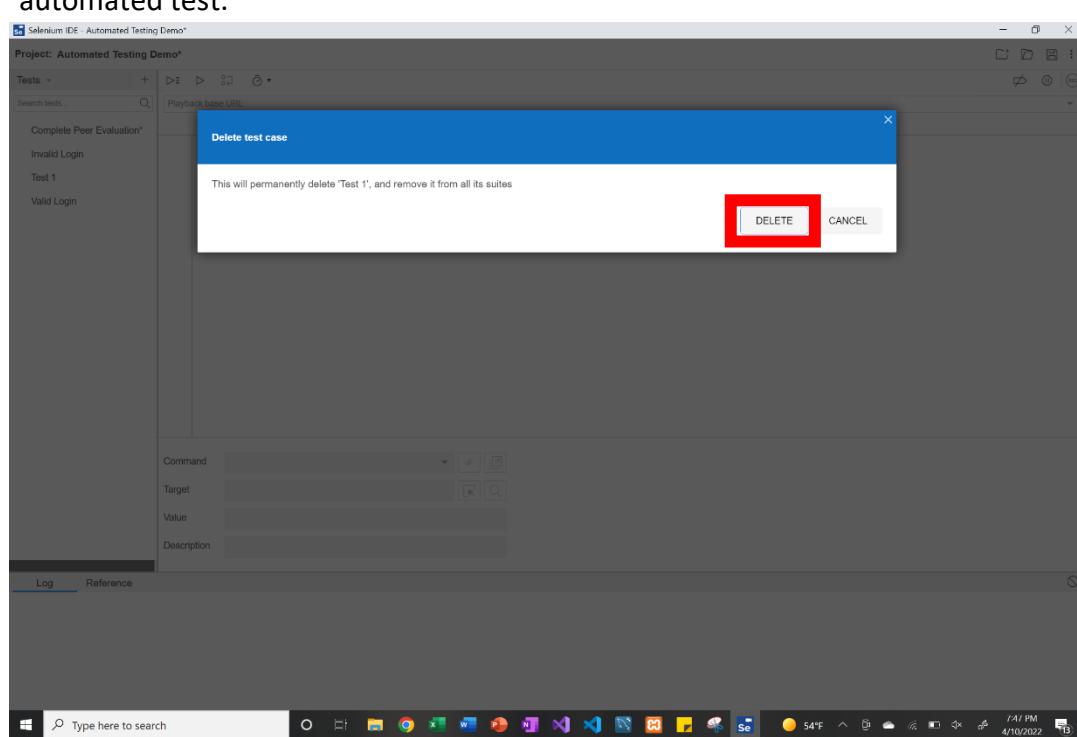


Deleting Automated Tests

1. Press the menu button for the “Test 1” test.
2. Press the “Delete” button on the dropdown to delete the test

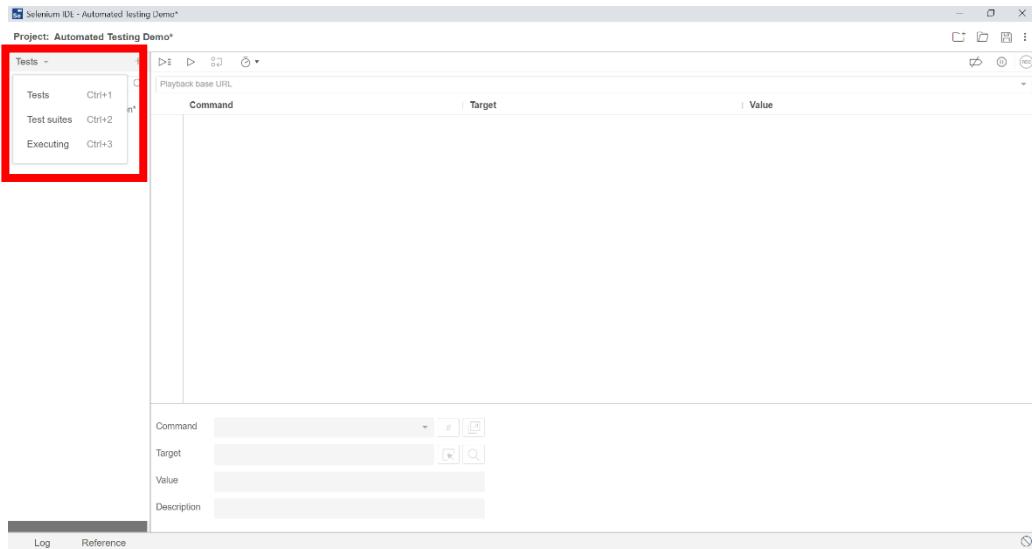


3. Press the “Delete” button on the verification window to permanently delete the automated test.

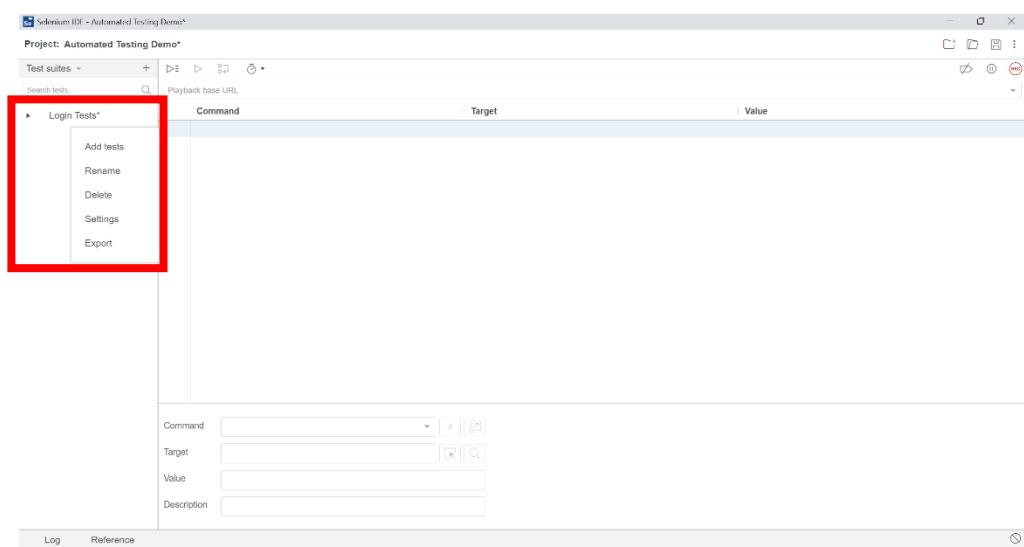


Test Suites

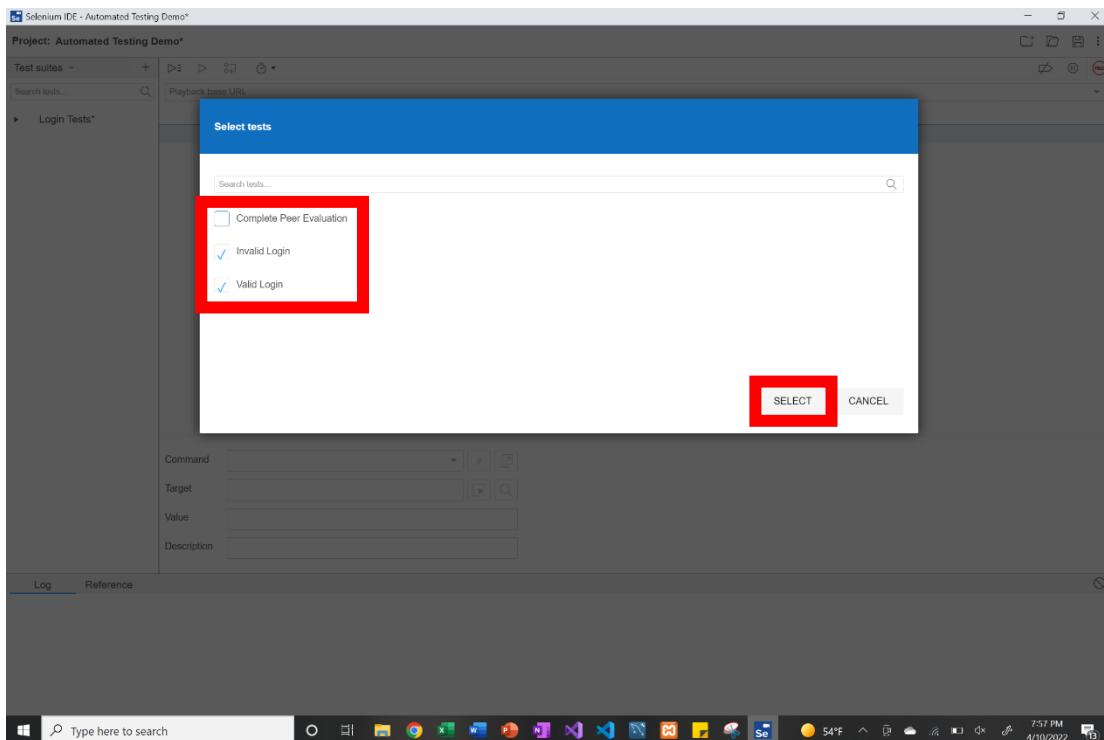
1. Click the dropdown arrow near the top left of the application window next to the word “Tests.” Right now, all test created in the project are shown in alphabetical order.
2. Press the “Test Suites” option int eh dropdown to display the current test suites, or test groupings.



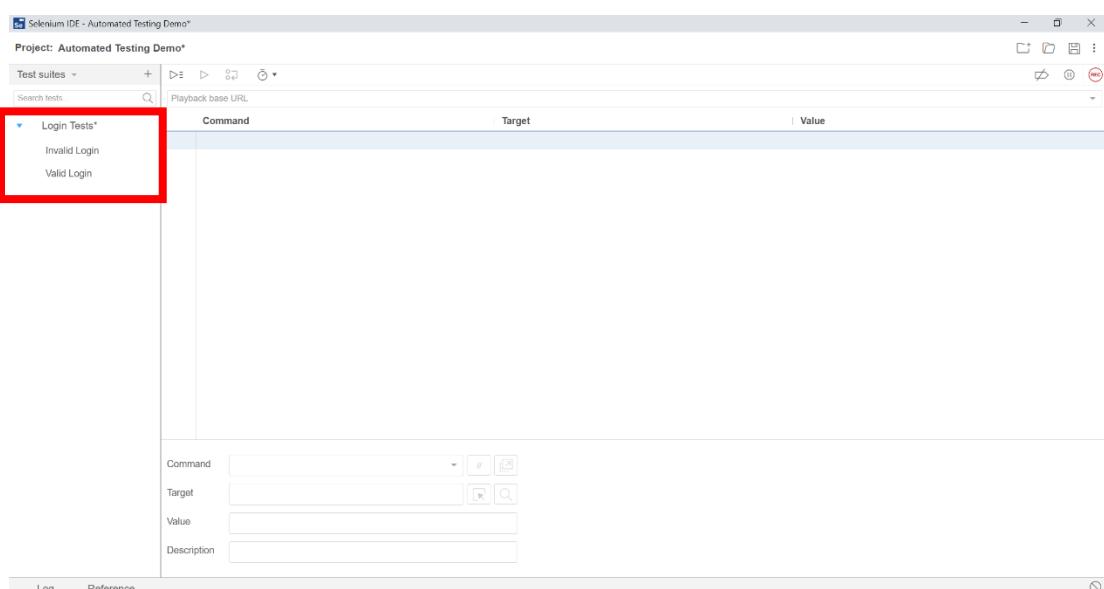
3. Press the menu button of “Default Suite,” and rename it “Login Tests” by using the same format used to rename the automated tests
4. Press the menu button of the “Login Tests” suite and click on the “Add Tests” option



5. Click the checkbox of both the “Invalid Login” and “Valid Login” tests. Then press “Select.” We are making sure to include only the tests that relate to one another. Both the “Valid Login” and Invalid Login” automated tests deal with logging into the system.



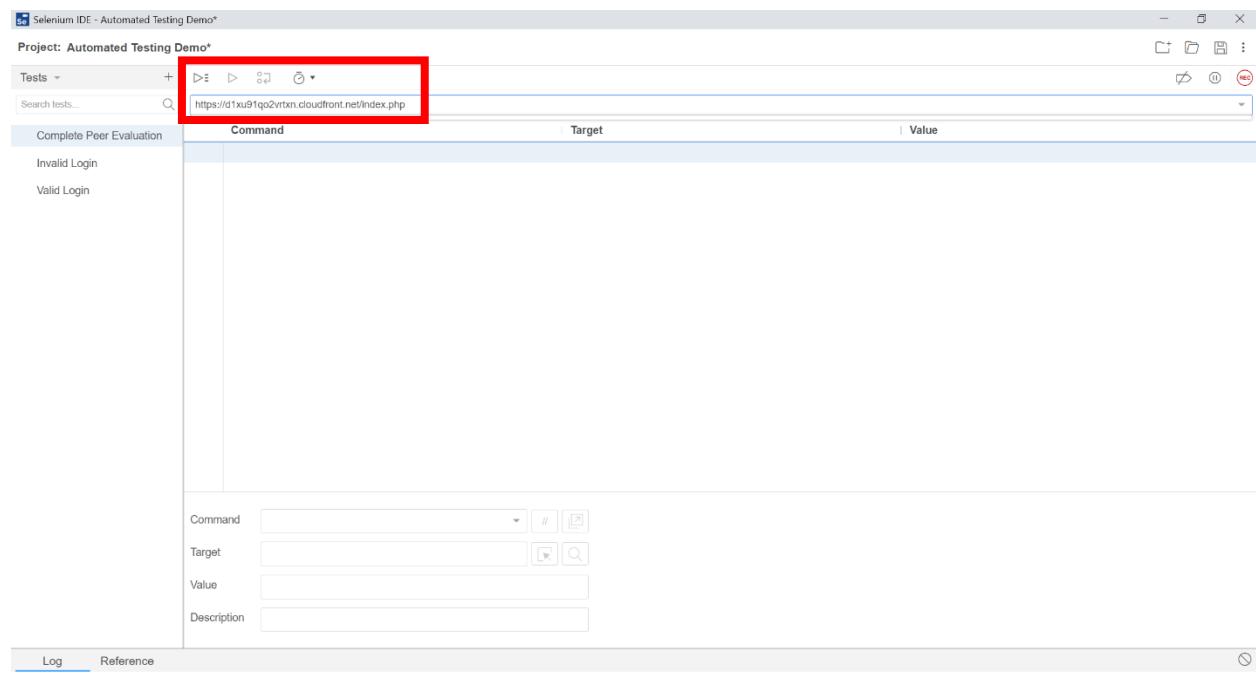
6. Press the dropdown arrow of the “Login Tests” suite to ensure that only the selected tests are assigned to that suite.



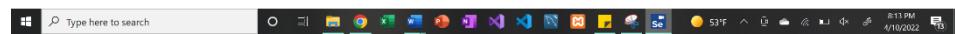
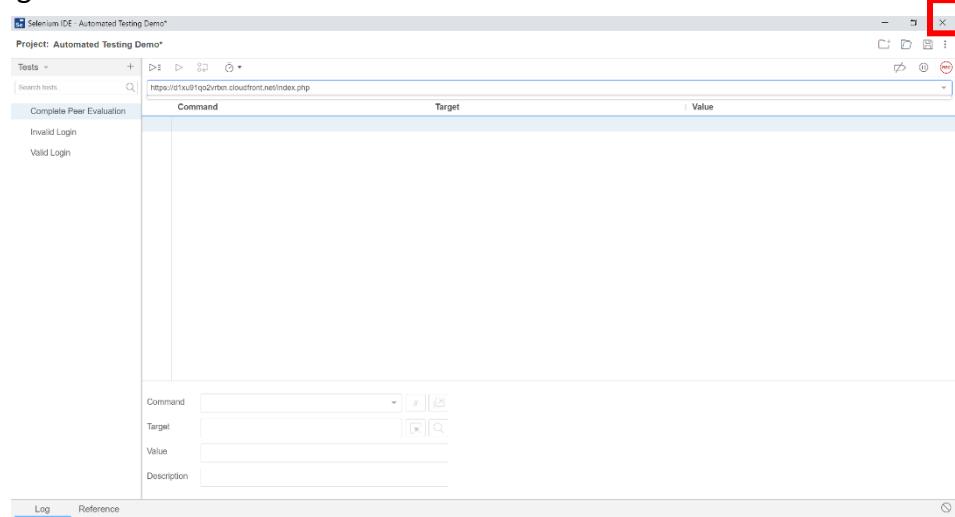
Recording a Test

Selenium IDE offers a feature in which the user can record themselves going through a selected website in real time. All actions done while in the live recording session will be automatically applied to the command line of the automated test. This eliminates the need to investigate the code of the website to locate the exact elements needed to test. Although there are a variety of testing commands, we will mostly be using “open,” “click,” “verify text,” “type,” and “submit.” The instructions to record an automated test are as follows:

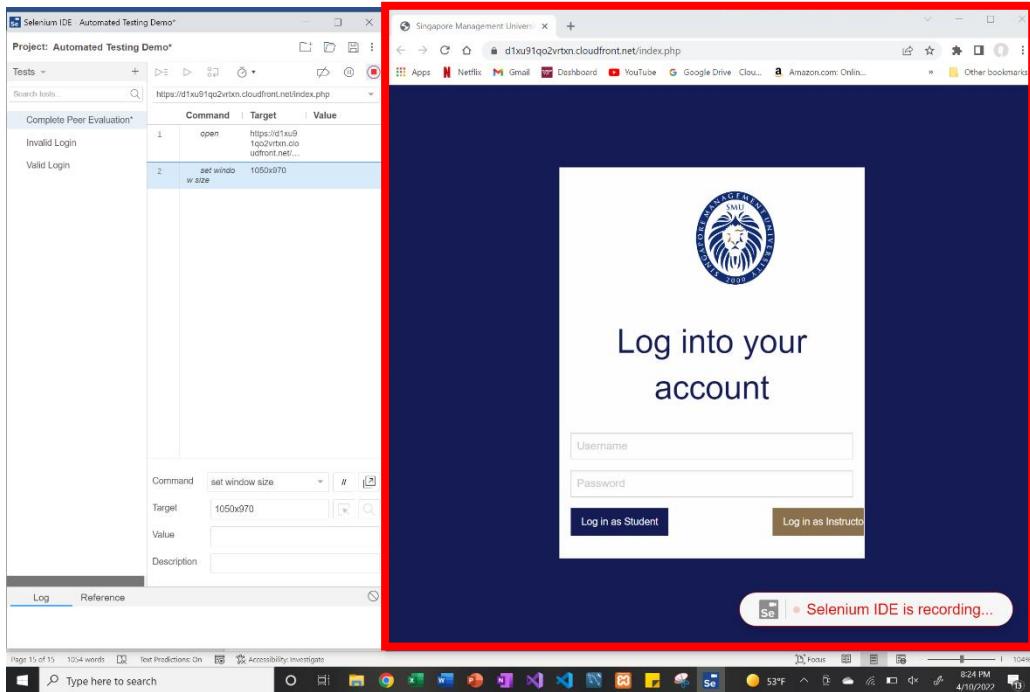
1. Return to the “Tests” display option and press the “Complete Peer Evaluation” automated test. We will be recording a test that will successfully submit a peer evaluation for a student in the SMU database.
2. Input the selected URL of the website that you are testing into the “Playback base URL” textbox at the top of Selenium IDE’s application window. For this demo, we will be using <https://d1xu91qo2vrtxn.cloudfront.net/index.php>, the URL for SMU Team 3’s peer review website.



3. After inputting the target URL, press the “REC” button at the top right of the application window. This will open a new chrome browser with the selected website and begin the recording



4. Once a chrome tab opens with the website in view, minimize the SMU website application to “1050x970.”
5. Resize the selenium IDE window to fit the remaining portion of your computer’s screen. This will allow you to perform actions on the SMU website and see the results as they come in on Selenium’s user interface.



6. Type “dcorcoran@smu.edu” into the username textbox and “password123” into the password textbox.
7. Press the “Log in as Student” button

The Selenium IDE application window shows a test case named "Complete Peer Evaluation". The "Valid Login" section contains the following steps:

	Command	Target	Value
1	✓ open	https://d1xu91qo2vrtxn.cloudfront.net/index.php	
2	✓ set window size	1050x970	
3	click	id=text	
4	type	id=text	dcorcoran@smu.edu
5	click	id=password	
6	type	id=password	password123

The browser screenshot shows the SMU login page with the logo and the text "Log into your account". The username field (dcorcoran@smu.edu) and password field (password123) are highlighted with red boxes. The "Log in as Student" button is also highlighted with a red box.

8. As an action is performed on the SMU website, the “Command”, “Target”, and “Value” items are automatically filled in on the Selenium IDE application window. This allows the user to complete the requirements for a test case much faster than manually inputted each item individually.

The Selenium IDE application window shows the same test case and steps as the previous screenshot. The "Valid Login" section is highlighted with a red box.

	Command	Target	Value
1	✓ open	https://d1xu91qo2vrtxn.cloudfront.net/index.php	
2	✓ set window size	1050x970	
3	click	id=text	
4	type	id=text	dcorcoran@smu.edu
5	click	id=password	
6	type	id=password	password123

The browser screenshot shows the SMU login page with the logo and the text "Log into your account". The username field (dcorcoran@smu.edu) and password field (password123) are highlighted with red boxes. The "Log in as Student" button is also highlighted with a red box.

9. Press on the first peer evaluation available. Since only the incomplete peer reviews assigned for “David Corcoran” are displayed, the names of students for available evaluations will change over time.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The browser window displays the 'Incomplete Peer Evaluations' page from Singapore Management University. The page lists four incomplete evaluations for the student 'Alex Williams'. Each evaluation has a due date and three student names: Alex Williams, John Doe, and Jane Doe. The Selenium IDE log on the left shows the following commands:

```

1 ✓ open https://d1xu91qo2vrbn.cloudfront.net/index.php
2 ✓ set window size 1050x970
3 click id=text doconoran@smu.edu
4 type id=password password123
5 click id=password password123
6 type id=password password123
7 click id=button

```

The status bar at the bottom of the Selenium IDE window indicates "Selenium IDE is recording".

10. Click on a selection for all 19 the radio buttons for the student being evaluated. I will be clicking the fourth radio button, “Always,” for all behavioral questions in the evaluation.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The browser window displays the 'Incomplete Peer Evaluations' page for the student 'Alex Williams'. The page title is 'Student being evaluated: Alex Williams'. Below it, there is a note: 'Please evaluate your teammate's performance based on the criteria below'. A red box highlights the list of 7 behavioral questions and their corresponding radio button options. The fourth option, 'Always', is selected for all questions. The Selenium IDE log on the left shows the following commands:

```

21 click css=cell:nth-child(13) > # Always
22 click css=cell:nth-child(14) > # Always
23 click css=cell:nth-child(15) > # Always
24 click css=cell:nth-child(16) > # Always
25 click css=cell:nth-child(17) > # Always
26 click css=cell:nth-child(18) > # Always
27 click css=cell:nth-child(19) > # Always

```

The status bar at the bottom of the Selenium IDE window indicates "Selenium IDE is recording".

11. Click on the “Command” dropdown of the Selenium application window and type “Verify Text”. Because there is no mouse action that can perform the “verify text” command, we will have to input it in manually. Ensure that the words contain all lowercase letters.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The Selenium IDE displays a test case titled 'Complete Peer Evaluation*' with a 'verify text' command highlighted. The web browser shows a form titled 'Incomplete Peer Evaluations' for evaluating a student named 'Alex Williams'. The 'Peer Evaluation' section asks for feedback on various criteria, each with a set of radio buttons from 'Never' to 'Always'. A red box highlights the 'verify text' command in the Selenium IDE's command list.

12. Click on the “Select target on page” button. This feature allows us to click on any element of the SMU website and assign that value to the “Target” textbox.

This screenshot is similar to the previous one, showing the Selenium IDE and a web browser. The Selenium IDE has the same test case and 'verify text' command selection. In the web browser, the 'Target' field of the 'verify text' command in the Selenium IDE is highlighted with a red box. The rest of the interface and content are identical to the first screenshot.

13. With the “Select target on page” button active, click the peer evaluation webpage’s title. This should say “Peer being evaluated: [Student’s Name].” By verifying that the label says the selected student’s name, we are checking that there is a connection to the student database. The element name will now appear in the “Target” textbox on the Selenium application window.

The screenshot displays two windows side-by-side. On the left is the "Selenium IDE - Automated Testing Demo" window, which is part of the Eclipse IDE. It shows a test case titled "Complete Peer Evaluation". The test case contains several "click" commands and one "verify text" command at step 28. The "verify text" command has a red box around its "Target" field, which is set to "css=h2". Below the command, there is a tooltip: "verify text locator, text" and "Soft assert the text of an element is present. The test will continue even if the verify fails." On the right is a web browser window titled "Incomplete Peer Evaluations" from the Singapore Management University website. The page header says "Peer Evaluation" and "Student being evaluated: Alex Williams". The main content area lists 7 criteria for evaluation, each with a radio button for "Never", "Sometimes", "Usually", "Regularly", or "Always". The "Always" option is selected for all criteria. A status bar at the bottom of the browser window indicates "Selenium IDE is recording".

14. Click on the next blank row after the “verify text” command. By doing this, the recording will write commands after the “verify text” command instead of before.

This screenshot shows the same setup as the previous one, but with a key difference: the "verify text" command at step 28 is now highlighted with a red box. This highlights the change made in step 14, where the user clicked on the next blank row after the command. The Selenium IDE interface shows the tooltip "Unknown command name provided." The browser window shows a success message: "You have successfully completed a peer evaluation for: Alex Williams". A "Go Back" button is visible below the message. The status bar at the bottom of the browser window still shows "Selenium IDE is recording".

15. Scroll to the bottom of the webpage and press the “Submit” button. This will submit the user form with the selected peer evaluation answers.

The Selenium IDE application window on the left displays a test case titled "Complete Peer Evaluation". The test case contains the following steps:

- Step 22: click css=cell:nth-c hild(14) > #AI ways
- Step 23: click css=cell:nth-c hild(15) > #AI ways
- Step 24: click css=cell:nth-c hild(16) > #AI ways
- Step 25: click css=cell:nth-c hild(17) > #AI ways
- Step 26: click css=cell:nth-c hild(18) > #AI ways
- Step 27: click css=cell:nth-c hild(19) > #AI ways
- Step 28: verify text css=h2

The browser window on the right shows a peer evaluation form with 19 questions. At the bottom of the form are two buttons: "Submit" and "Reset". A status message "Selenium IDE is recording" is visible in the bottom right corner of the browser window.

16. Verify that the confirmation text has a connection to the database by writing “verify text” into the “Command” dropdown of the Selenium IDE application window.

The Selenium IDE application window on the left displays a test case titled "Incomplete Peer Evaluations". The test case contains the following steps:

- Step 22: click css=cell:nth-c hild(14) > #AI ways
- Step 23: click css=cell:nth-c hild(15) > #AI ways
- Step 24: click css=cell:nth-c hild(16) > #AI ways
- Step 25: click css=cell:nth-c hild(17) > #AI ways
- Step 26: click css=cell:nth-c hild(18) > #AI ways
- Step 27: click css=cell:nth-c hild(19) > #AI ways
- Step 28: verify text css=h2
- Step 29: click id=button
- Step 30: verify text

The browser window on the right shows a confirmation message: "You have successfully completed a peer evaluation for: Alex Williams". Below this message is a "Go Back" button. A status message "Selenium IDE is recording" is visible in the bottom right corner of the browser window.

17. Click on the “Select target on page” button and select the [Student’s Name] element.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The browser window displays a success message: "You have successfully completed a peer evaluation for: Alex Williams". The Selenium IDE log shows a 'verify text' command with a target set to 'css=h3:nth-child(5)'.

18. Click on the “Go Back” button.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The browser window displays a success message: "You have successfully completed a peer evaluation for: Alex Williams". A red box highlights the "Go Back" button in the browser's footer.

19. Click on the “Log Out” button.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The browser displays the 'Incomplete Peer Evaluations' page from the Singapore Management University website. The test log in the IDE shows the following steps:

```

22 click css=.cell:nth-child(14) > #Always
23 click css=.cell:nth-child(15) > #Always
24 click css=.cell:nth-child(16) > #Always
25 click css=.cell:nth-child(17) > #Always
26 click css=.cell:nth-child(18) > #Always
27 click css=.cell:nth-child(19) > #Always
28 verify_text css=h2
29 click id=button
30 click id=button
    
```

The 'Log Out' button in the IDE is highlighted with a red box. The browser window shows a list of incomplete peer evaluations with due dates and student names. A status bar at the bottom of the screen indicates "Selenium IDE is recording".

20. Press the “REC” button again to close the recording. The complete peer evaluation test case has been fully recorded.

The screenshot shows the Selenium IDE interface on the left and a web browser window on the right. The browser displays the university's login page. The test log in the IDE shows the following steps:

```

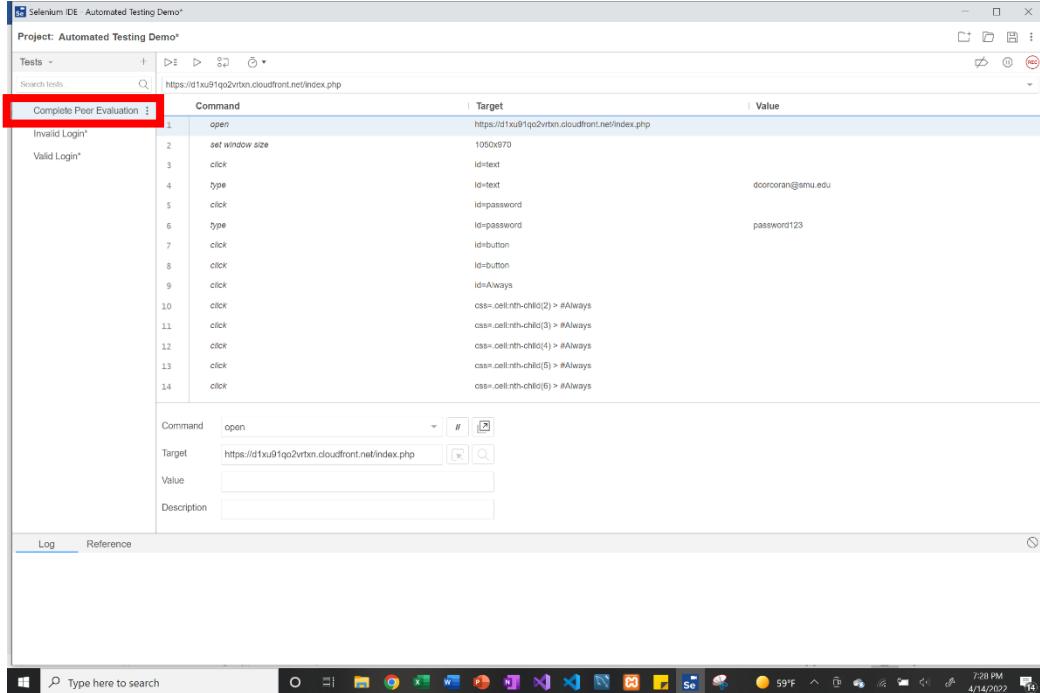
23 click css=.cell:nth-child(15) > #Always
24 click css=.cell:nth-child(16) > #Always
25 click css=.cell:nth-child(17) > #Always
26 click css=.cell:nth-child(18) > #Always
27 click css=.cell:nth-child(19) > #Always
28 verify_text css=h2
29 click id=button
30 click id=button
31 click xpath=//li[@id='menu']/a
    
```

The 'REC' button in the IDE is highlighted with a red box. The browser window shows the university's login page with the SMU logo and a 'Log into your account' message. A status bar at the bottom of the screen indicates "Selenium IDE is recording".

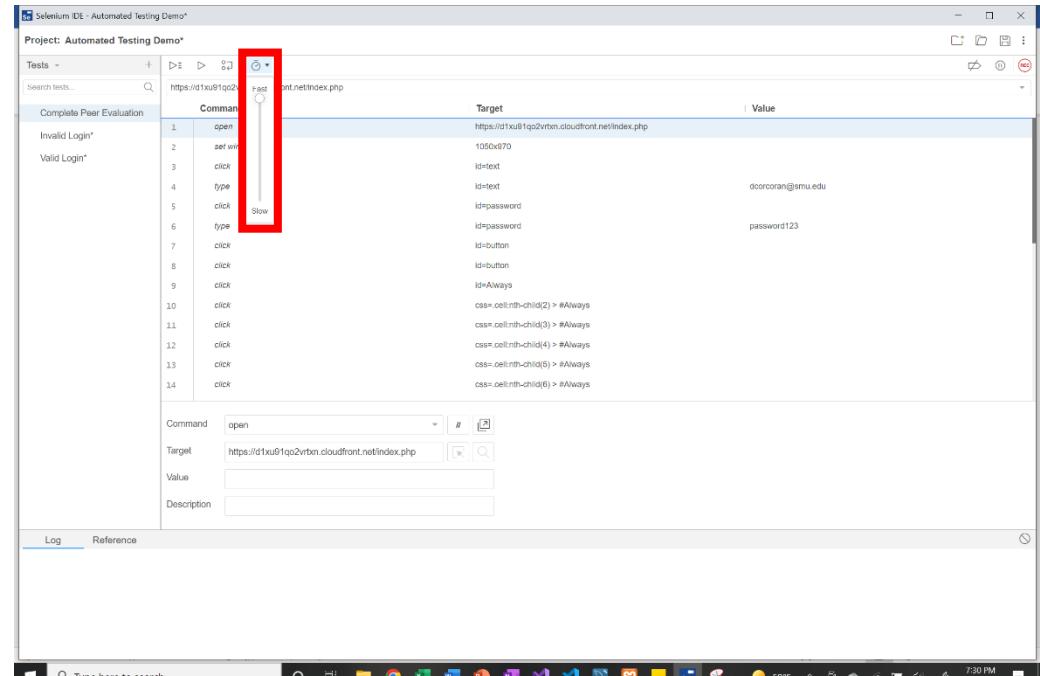
Test Playback

Replaying a test is very important in requirements testing. Replaying allows you to see your recorded actions in real time, or at a variable speed, as well as the running test's outcome.

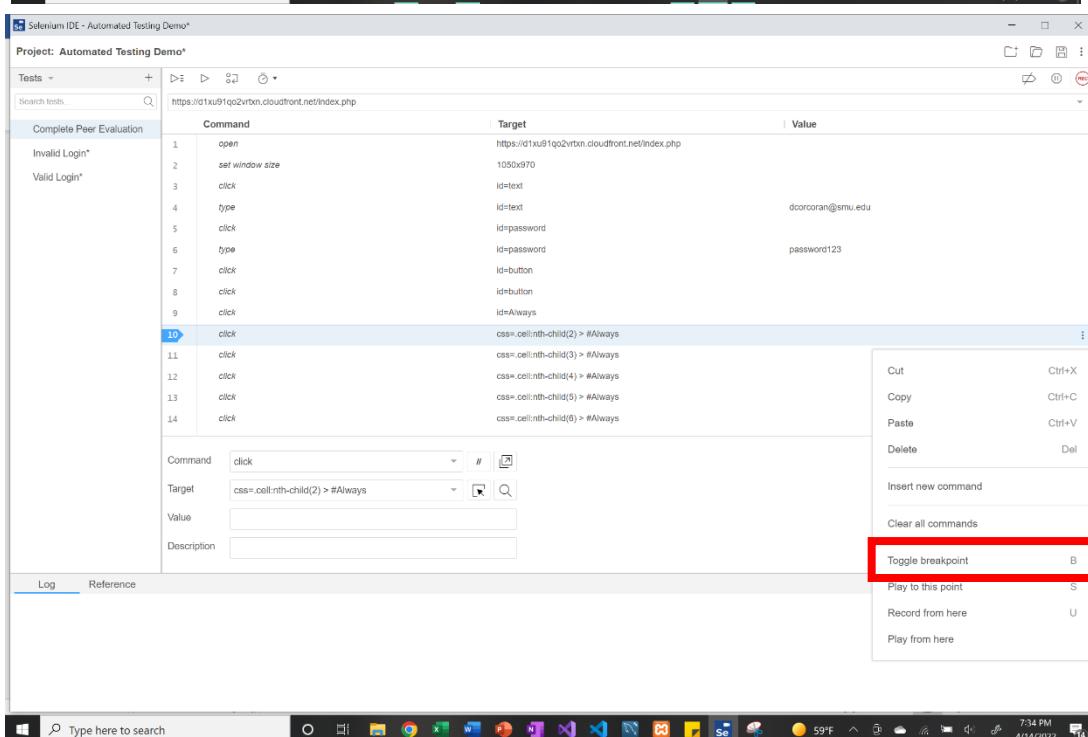
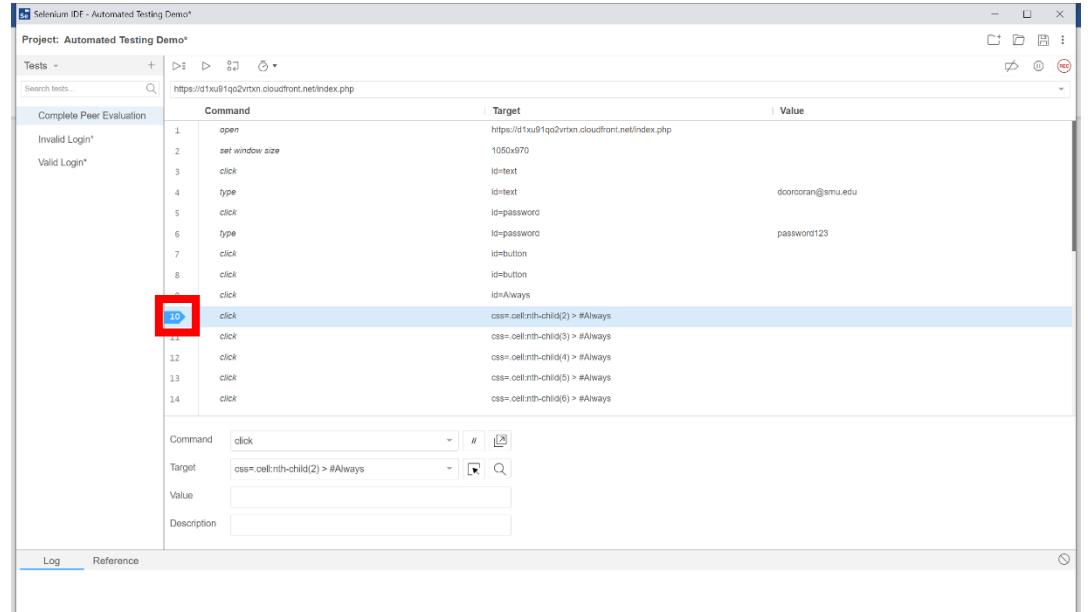
- Click the “Complete Student Evaluation” test in the test case pane. The entire test case should already be completed from the previous steps.



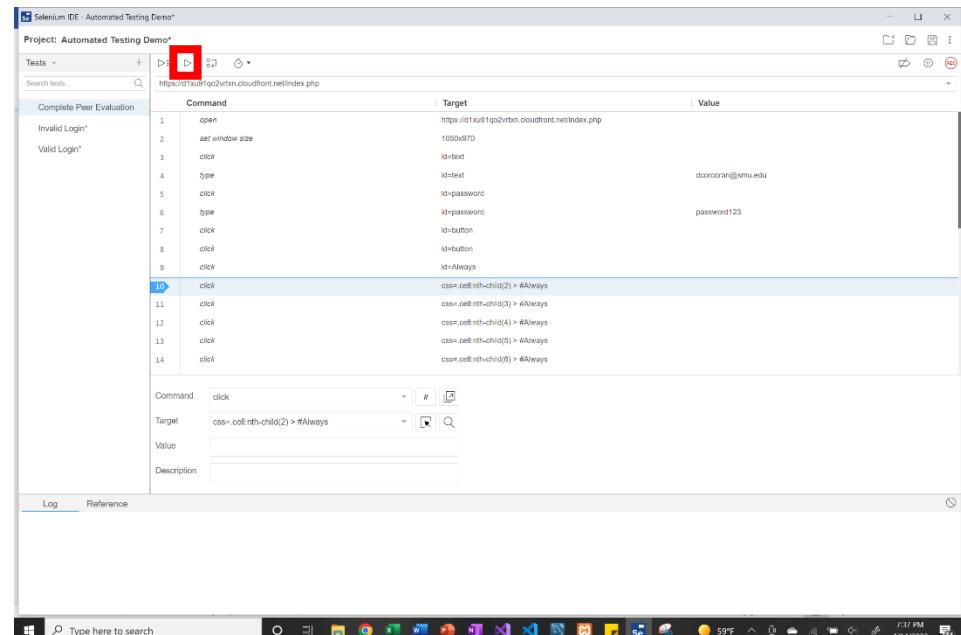
- Set the replay speed with the slider at the top of the screen. You are able to slow down or speed up the replay of the test. I will put my test on the fastest setting.



3. Set a breakpoint on step 10 by clicking the step number on the left side of testing dashboard. Alternatively, you can click the menu button at the right of the step and select “toggle breakpoint” from there. Breakpoints stop pause the breakpoint in the middle of testing.



4. The first play button with three horizontal lines is a “play all” button. This button will replay all tests one after another. Because we have only completed one test, we will skip over this function.
5. Press the play button, the small triangle button at the top of the selenium IDE window. The replay will begin automatically once pressed.



6. The replay will automatically stop at the previously selected breakpoint. A message saying “Paused in Debugger” will appear at the top of Selenium IDE.

The screenshot shows the Selenium IDE and a browser window side-by-side. The Selenium IDE shows the test steps, and the browser window shows the 'Incomplete Peer Evaluations' page for 'Student being evaluated: John Doe'. Both windows have red boxes highlighting the "Paused in debugger" message at the top.

7. Every step previously played that pass should be highlighted in green. Steps that don't pass will be highlighted in red.
8. If at any time you want to stop the test, you can press the square button at the top of the testing page. However, do not press the button now.

The screenshot shows the Selenium IDE interface with a test titled "Complete Peer Evaluation". The test log lists 11 steps:

- 1. ✓ open https://d1xu91qp2vrtxn.cloudfront.net/STUDENT_form.php?user=John%20Doe&eval=107
- 2. ✓ set window size 1050x970
- 3. ✓ click #id-text
- 4. ✓ type #id-text doocoran@samu.edu
- 5. ✓ click #idpassword
- 6. ✓ type #idpassword password123
- 7. ✓ click #id-button
- 8. ✓ click #id-button
- 9. ✓ click #idAlways
- 10. click css=.cell:nth-child(2) > #Always
- 11. click css=.cell:nth-child(3) > #Always

Step 10 is highlighted in yellow, indicating it is the current step. The browser window to the right shows the "Incomplete Peer Evaluations" page for "Student being evaluated: John Doe".

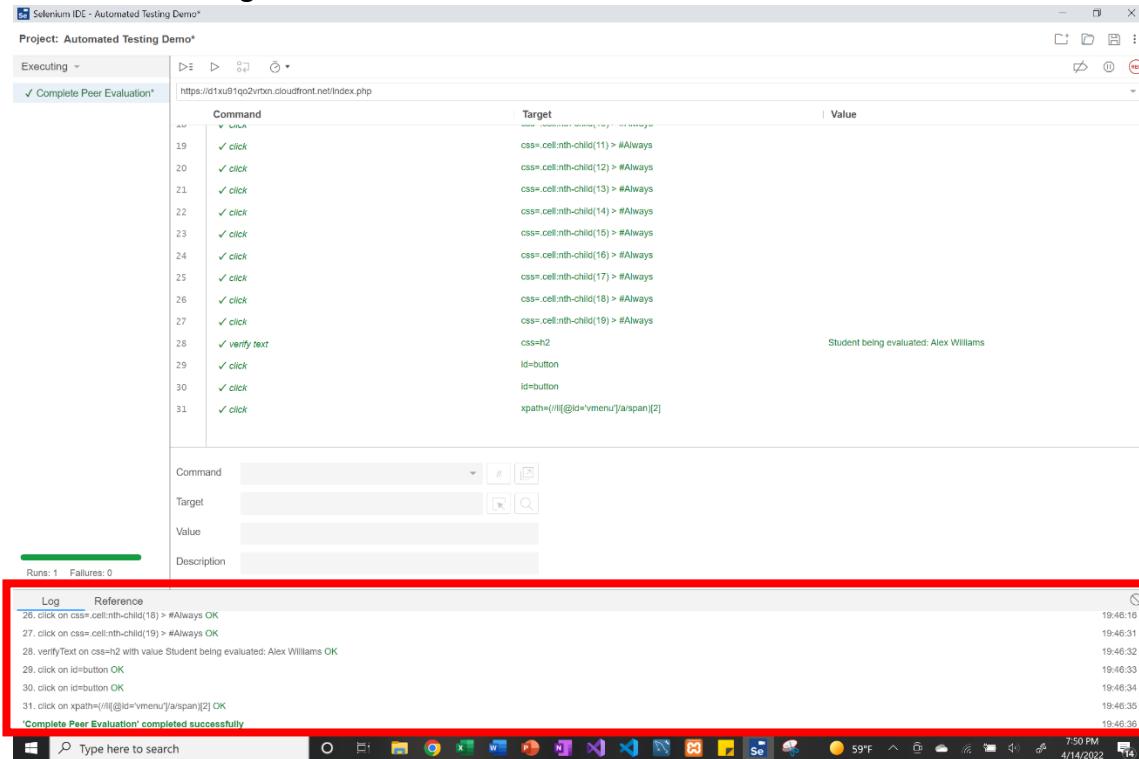
9. Resume the test by pressing the pause button, or the two vertical lines at the top of Selenium.

The screenshot shows the Selenium IDE interface with a test titled "Complete Peer Evaluation". The test log lists 11 steps:

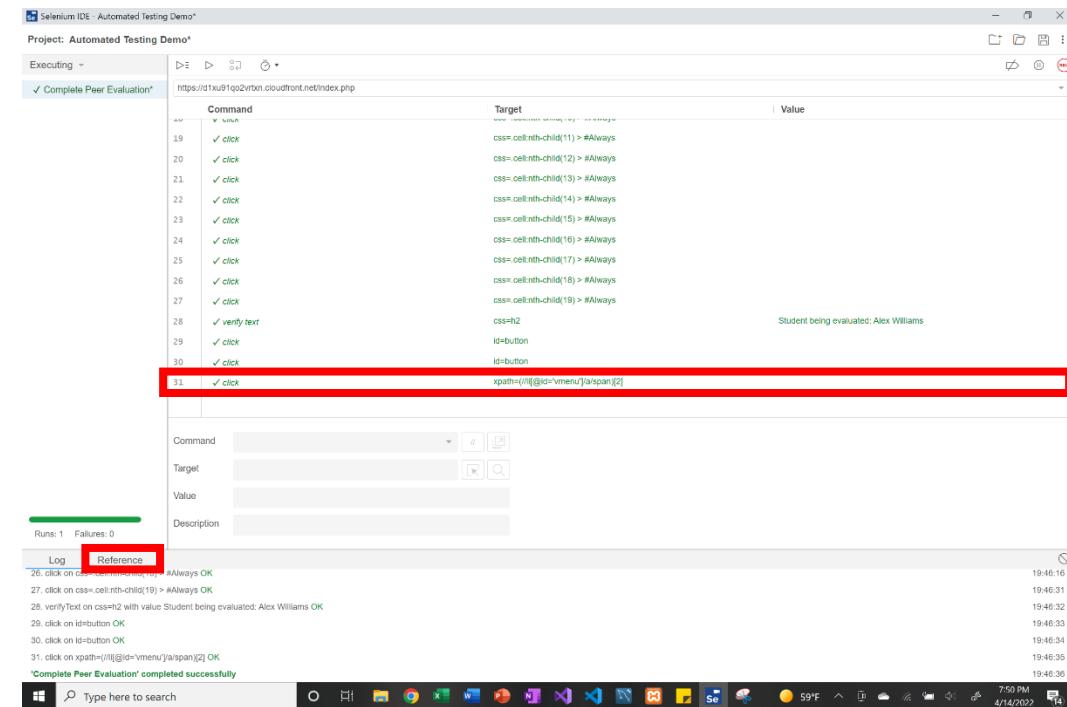
- 1. ✓ open https://d1xu91qp2vrtxn.cloudfront.net/STUDENT_form.php?user=John%20Doe&eval=107
- 2. ✓ set window size 1050x970
- 3. ✓ click #id-text
- 4. ✓ type #id-text doocoran@samu.edu
- 5. ✓ click #idpassword
- 6. ✓ type #idpassword password123
- 7. ✓ click #id-button
- 8. ✓ click #id-button
- 9. ✓ click #idAlways
- 10. click css=.cell:nth-child(2) > #Always
- 11. click css=.cell:nth-child(3) > #Always

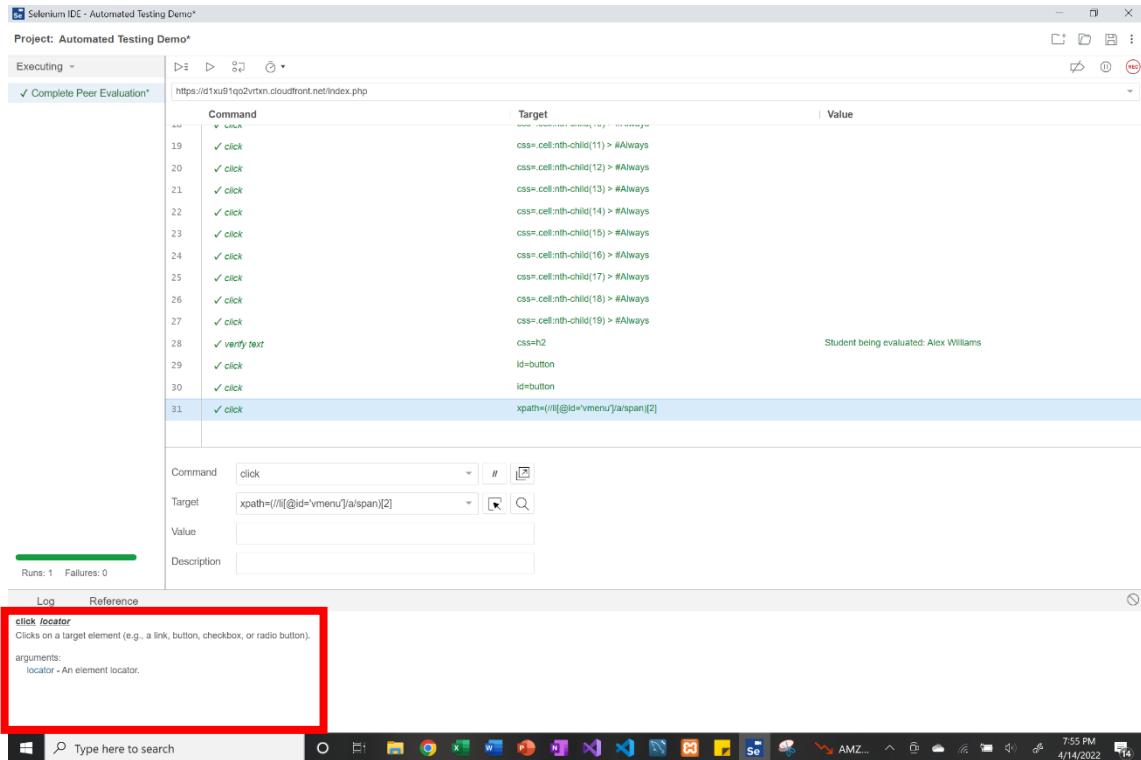
Step 10 is highlighted in yellow, indicating it is the current step. The browser window to the right shows the "Incomplete Peer Evaluations" page for "Student being evaluated: John Doe".

10. Once the replay is complete, the Selenium window will take control. The test will automatically stop once the last step is reached.
11. The “Log” keeps track of the steps taken and decides whether they have passed or failed. The “Log” is situated at the bottom on Selenium.

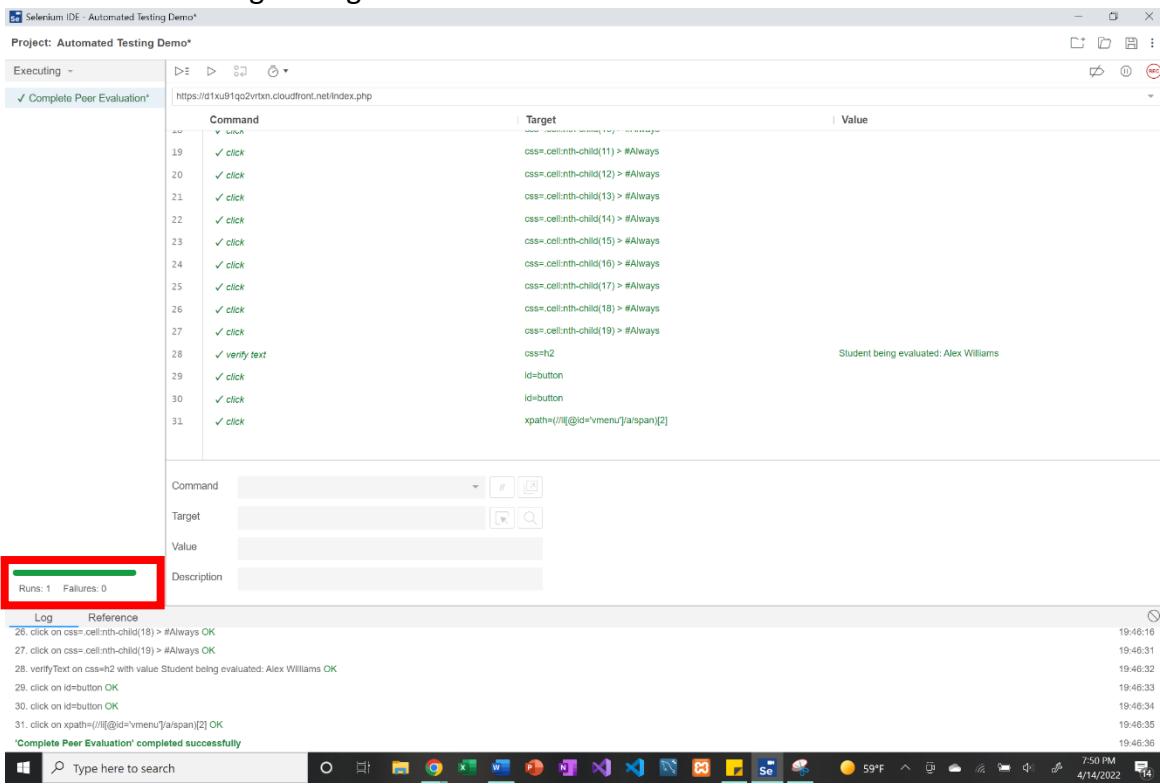


12. Click the last step and press the “reference” tab. Pressing the “Reference” tab next to the “Log” tab brings the user to a description of the highlighted step. In this cas it is the “click” command.





13. The number of runs and failures is also depicted at the bottom of Selenium's testing pane. This shows the result of the previously run tests and how many problems occurred during testing.



14. Click the comment button on the last test step. This button should look like two forward slashes in a row. This action nullifies the step altogether and will put the icon next to the step number.

The screenshot shows the Selenium IDE interface with a test case titled "Complete Peer Evaluation*". The test steps listed are:

- 19. ✓ click
- 20. ✓ click
- 21. ✓ click
- 22. ✓ click
- 23. ✓ click
- 24. ✓ click
- 25. ✓ click
- 26. ✓ click
- 27. ✓ click
- 28. ✓ verify text
- 29. ✓ click
- 30. ✓ click
- 31. ✓ click

The step at index 31 is highlighted with a red box. Below the list, there is a detailed view of the step:

Command	//click	Enable/Disable command
Target	xpath=(//li[@id='vmenu']/a/span)[2]	
Value		
Description		

A red box highlights the "Enable/Disable command" button. The status bar at the bottom right indicates "Student being evaluated: Alex Williams".

Conclusion

Selenium IDE is a very helpful automated testing program. The software allows requirements analysts to run automated tests using a record and playback function, significantly reducing the time it can take to test website functionality. Selenium IDE is a completely free service with a wide variety of features useful in software testing. It has a simple graphical layout, does not require previous programming knowledge, and is easy to understand. Because the program is a browser extension, it is very easy to install on any computer with internet connection and either Mozilla Firefox or Google Chrome. I highly recommend Selenium IDE to anyone in need of an automated testing technology especially if they have limited knowledge on the subject and technical requirements.