**Vishal Salvi - 2019230069**

**Shivam Pawar - 2019230068**

**Batch C**

**EXPERIMENT NO. 8**

**Aim​:** To Perform automating testing of web application using selenium automated testing tool

**Theory:**

## **What Is Selenium?**

Selenium is an open-source tool that is used for automating the tests carried out on web browsers (Web applications are tested using any web browser).

Wait, before you get carried away, let me re-iterate that, only testing of web applications is possible with Selenium. We can neither test any desktop (software) application nor test any mobile application using Selenium.

It’ a bummer right? I can feel your pain. But don’t worry, there are many tools for testing software and mobile applications like: IBM’s RFT, HP’s QPT, Appium and many more. But, the focus of this blog is, testing dynamic web applications and why Selenium is the best for that purpose.

### **What are the advantages of Selenium?**

Since Selenium is open-source, there is no licensing cost involved, which is a major advantage over other testing tools. Other reasons behind Selenium’s ever growing popularity are:



* Test scripts can be written in any of these programming languages: [Java](https://www.edureka.co/blog/java-tutorial/), [Python](https://www.edureka.co/blog/videos/python-tutorial/), [C#](https://www.edureka.co/blog/c-sharp-tutorial/), [PHP](https://www.edureka.co/blog/php-tutorial-for-beginners/), [Ruby](https://www.edureka.co/blog/ruby-on-rails-tutorial/), [Perl](https://www.edureka.co/blog/videos/learn-perl-the-jewel-of-scripting-languages/) & .Net
* Tests can be carried out in any of these OS: Windows, Mac or Linux
* Tests can be carried out using any browser: Mozilla Firefox, Internet Explorer, Google Chrome, Safari or Opera
* It can be integrated with tools such as [TestNG](https://www.edureka.co/blog/dataprovider-in-testng/) & [JUnit](https://www.edureka.co/blog/junit-tutorial/) for managing test cases and generating reports
* It can be integrated with Maven, [Jenkins](https://www.edureka.co/blog/jenkins-tutorial/) & Docker to achieve Continuous Testing

**But there surely has to be shortcomings right?**

* We can use Selenium only to test web applications. We cannot test desktop applications or any other software
* There is no guaranteed support available for Selenium. We need to leverage on the available customer communities
* It is not possible to perform testing on images. We need to integrate Selenium with Sikuli for image based testing
* There is no native reporting facility. But we can overcome that issue by integrating it with frameworks like TestNG or JUnit

## **Need For Software Testing**

Software testing is where it all boils down to. Today’s world of technology is completely dominated by machines, and their behavior is controlled by the software powering it. Will the machines behave exactly as we want them to? Everytime? Everywhere? The answer to these questions lie in software testing.

At the end of the day, it is the software application’s success rate which is going to control your business growth. The same thing can be said even for web applications because most businesses today are completely reliant on the internet.

Take for example, any e-commerce company. Be it Amazon or E-Bay or Flipkart, they rely on the customer traffic on their web sites and traffic on their web based mobile applications for business.

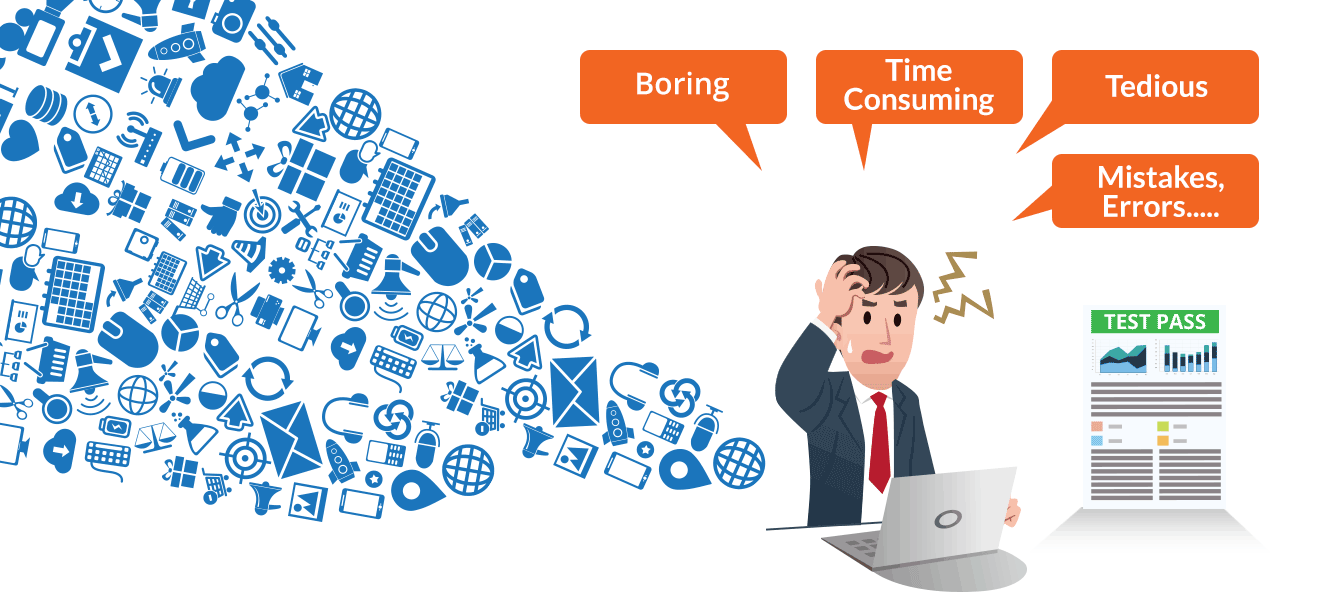
Imagine, if something catastrophic happens like the prices of a number of products being capped off at 10$, all because of a small bug in a “not so easily readable” part of the code. Then what can be done, and how can we prevent it the next time?

By testing the code before deployment right? So, that is the need for software testing. But, what is Selenium? Is it a software testing tool? Well, Selenium is an automation testing tool!

**Software testing is of two types:** Manual Testing & Automation Testing. Selenium was founded as an automation testing tool to overcome the drawbacks/ limitations of Manual testing.

## **Challenges With Manual Testing**

Manual testing means the (web) application is tested manually by QA testers. Tests need to be performed manually in every environment, using a different data set and the success/ failure rate of every transaction should be recorded.

Look at the above image of a poor chap, who manually verifies the transactions recorded. The challenges he is facing cause fatigue, boredom, delay in work, mistakes and errors because of manual effort. This lead to the invention of Selenium (automation testing tool).

## 

## 

## 

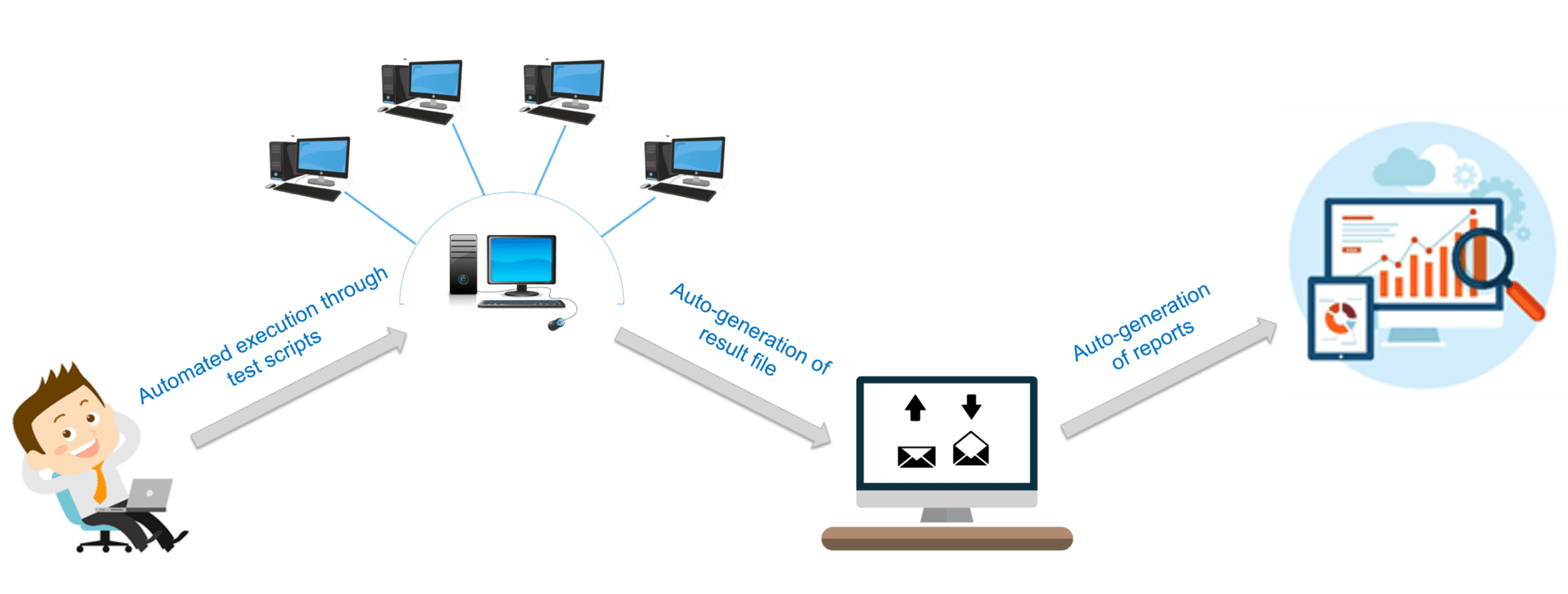
## **Automation Testing Beats Manual Testing**

Automation testing beats manual testing every time. Why? Because it is faster, needs less investment in human resource, it is not prone to errors, frequent execution of tests is possible, supports lights out execution, supports regression testing and also functional testing.

Let’s take a similar example to the one mentioned earlier. Suppose there is a login page and we need to verify if all the login attempts are successful, then it will be really easy to write a piece of code which will validate if all the transaction/ login attempts are a success or not (automated test case execution).

Moreover, these tests can be configured in such a way that they are tested in different environments and web browsers. What else can be done? You can automate the generation of result file, by scheduling it for a particular time during the day. Then you can also automate the generation of reports based on those results and what not.

The key point is that automation testing makes a tester’s job a whole lot simpler. Check out the image below which shows a more relaxed environment in which the same tester is working.

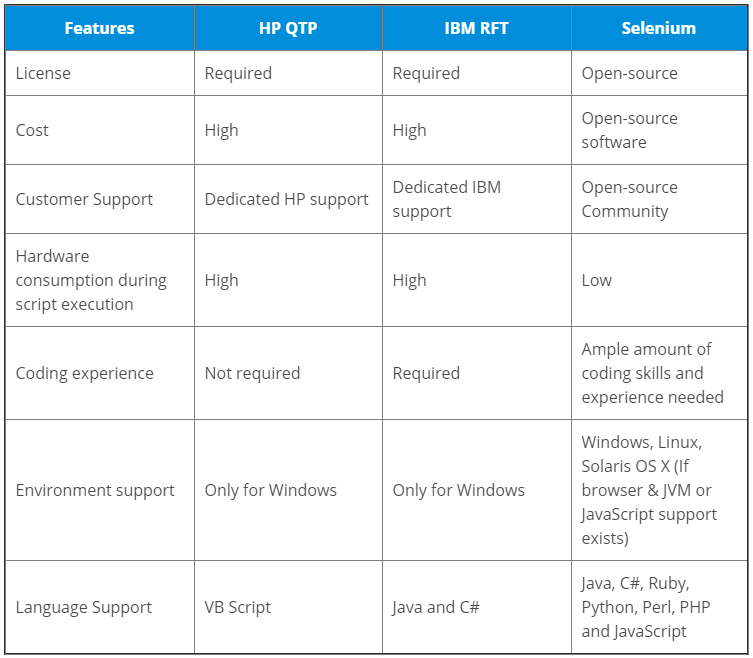


Now let us see where Selenium stands in the market.

## 

## 

## **Selenium vs QTP vs RFT**

It is pretty clear from the above table why Selenium is the most preferred tool. But there are many flavors in Selenium and you should know which is the appropriate right Selenium tool for your need.

## **Selenium Suite Of Tools**

* Selenium RC (Now depreciated)
* Selenium IDE
* Selenium Grid
* Selenium WebDriver

### **What are the components of Selenium?**

### **Selenium RC (Remote Control)**

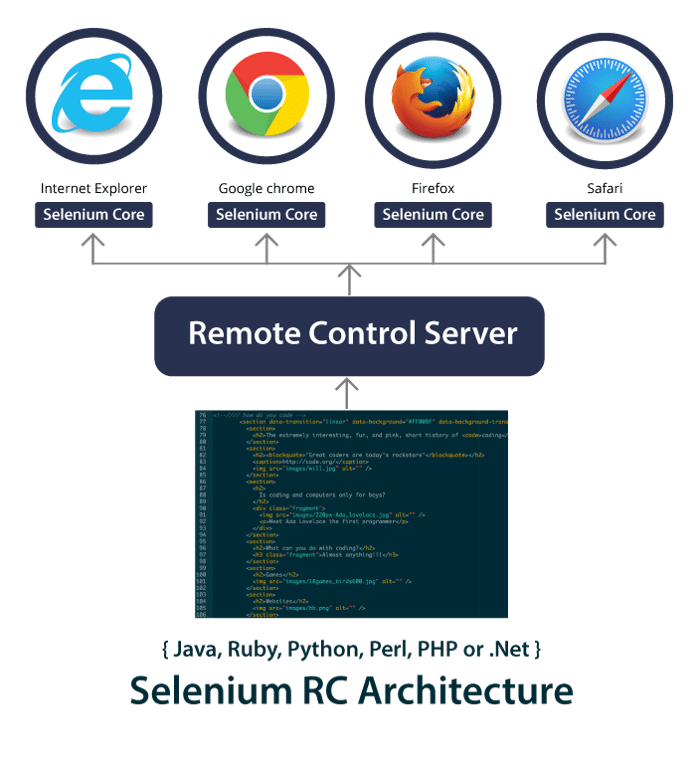
Before I talk about the details of Selenium RC, I would like to go a step back and talk about the first tool in the Selenium project. Selenium Core was the first tool. But, Selenium Core hit a roadblock in terms of cross-domain testing because of the same-origin policy. The same-origin policy prohibits JavaScript code from accessing web elements that are hosted on a different domain compared to where the JavaScript was launched.

To overcome the same-origin policy issue, testers needed to install local copies of both Selenium Core (a JavaScript program) and the webserver containing the web application being tested so they would belong to the same domain. This lead to the birth of Selenium RC, which is accredited to then ThoughtWork’s engineer, Paul Hammant.

RC overcame the problem by involving an HTTP proxy server to “trick” the browser into believing that Selenium Core and the web application being tested come from the same domain. Thus making RC a two-component tool.

1. Selenium RC Server
2. Selenium RC Client – Library containing your programming language code

RC Server communicates using simple HTTP GET/ POST requests. Look at the below image for understanding the RC architecture.



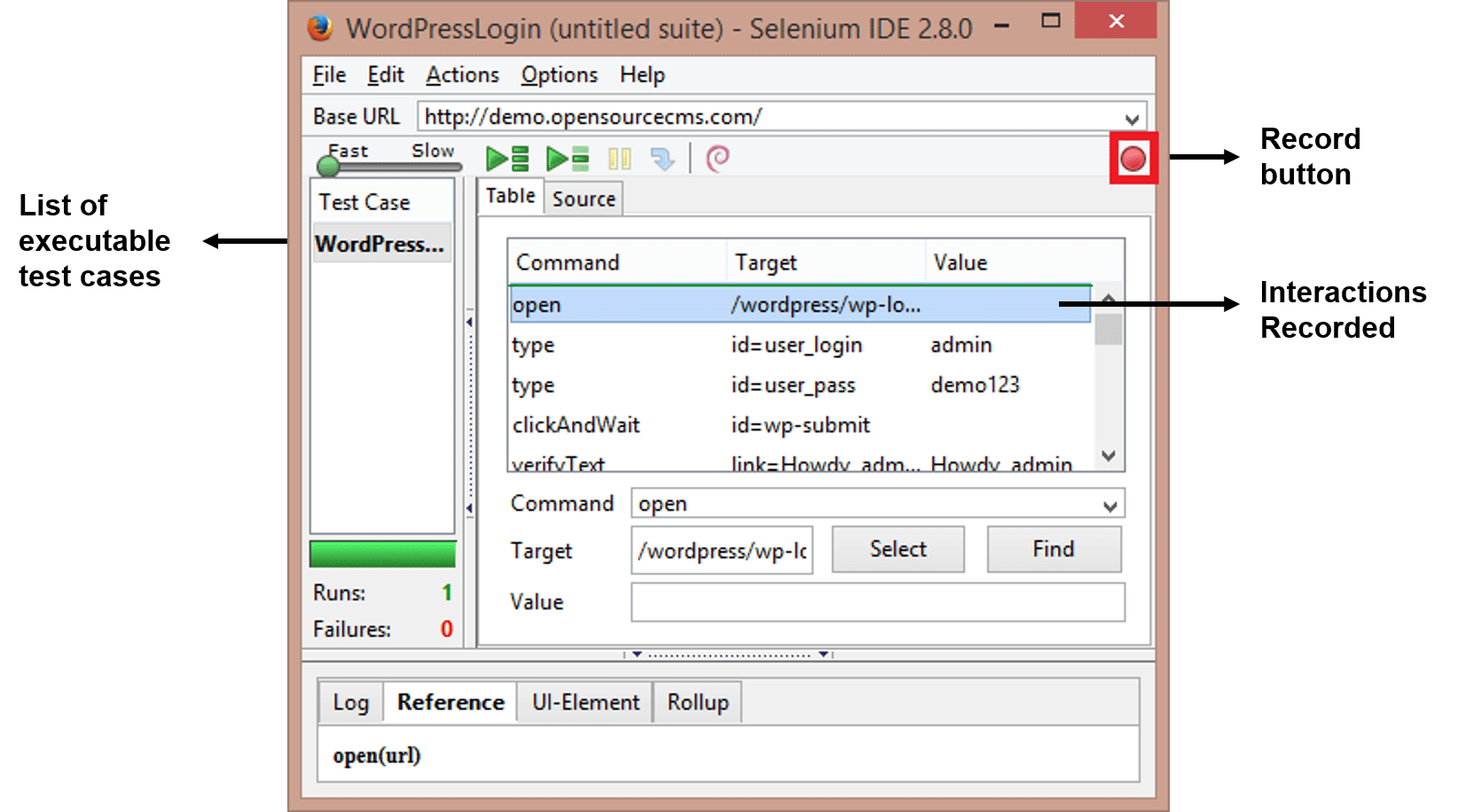
Selenium project’s flagship tool was Selenium RC as it was their first tool and it could be used to write test cases in different programming languages. But the drawback with RC is that every communication with the RC server is time consuming and hence RC is very slow. So slow, that it would sometimes take hours to complete single tests.

From Selenium v3 onwards, RC has been depreciated and moved to legacy package. You can however download and work with RC, but unfortunately you cannot avail support for it. But on the flip side, why would you want to use a tool which is outdated, especially when there is a more efficient tool called Selenium WebDriver. Before I talk about WebDriver, let me discuss about IDE and Grid, which are the other tools that make up Selenium v1.

### **Selenium IDE (Integrated Development Environment)**

In 2006, Shinya Kastani from Japan had donated his Selenium IDE prototype to Apache’s Selenium project. It was a Firefox plugin for faster creation of test cases. IDE implemented a record and playback model wherein, test cases are created by recording the interactions which the user had with the web browser. These tests can then be played back any number of times.

The advantage with Selenium IDE is that, tests recorded via the plugin can be exported in different programming languages like: Java, Ruby, Python etc. Check out the below screenshot of Firefox’s IDE plugin.



**But the associated shortcomings of IDE are:**

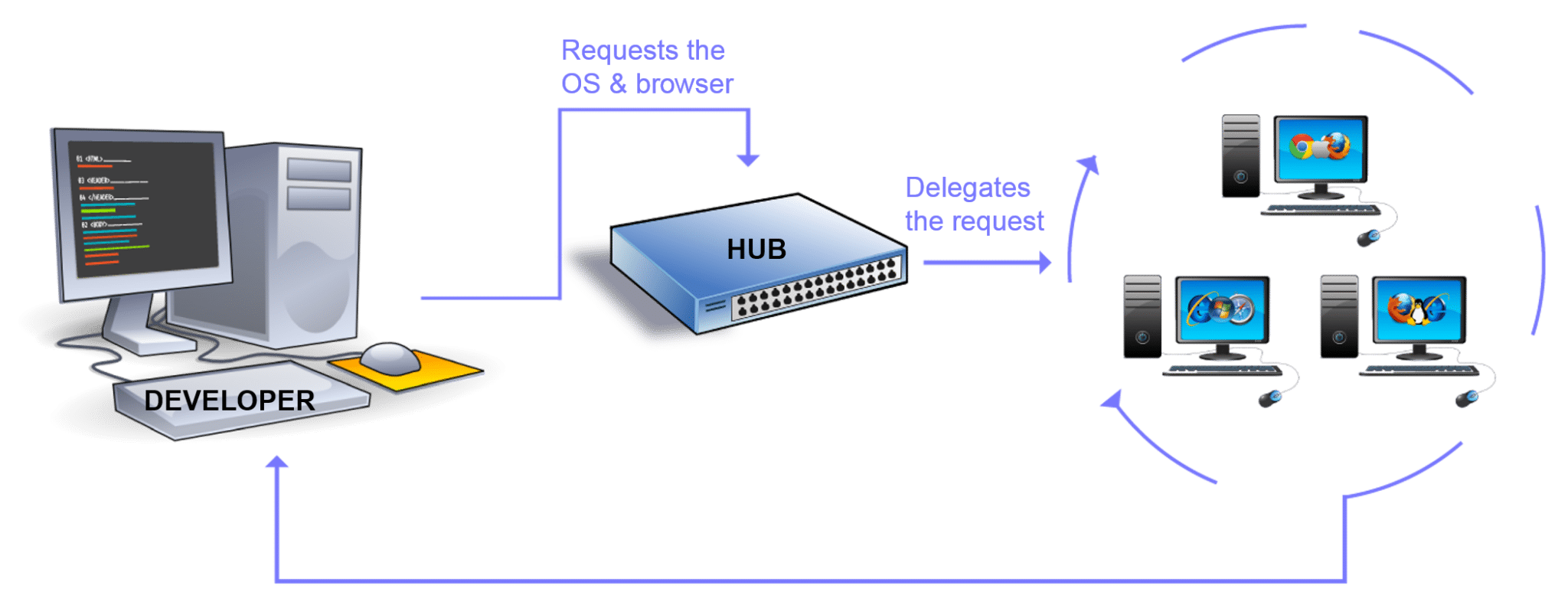
* Plug-in only available for Mozilla Firefox; not for other browsers
* It is not possible to test dynamic web applications; only simple tests can be recorded
* Test cases cannot be scripted using programming logic
* Does not support Data-Driven testing

These were some of the aspects of Selenium IDE. Let me now talk about Selenium Grid.

### **What is Selenium Grid**

Selenium Grid was developed by Patrick Lightbody and initially called HostedQA (initially a part of Selenium v1) and it was used in combination with RC to run tests on remote machines. In fact, with Grid, multiple test scripts can be executed at the same time on multiple machines.

Parallel execution is achieved with the help of Hub-Node architecture. One machine will assume the role of Hub and the others will be the Nodes. Hub controls the test scripts running on various browsers inside various operating systems. Test scripts being executed on different Nodes can be written in different programming languages.



Grid is still in use and works with both WebDriver and RC. However, maintaining a grid with all required browsers and operating systems is a challenge. For this, there are multiple online platforms that provide an online Selenium Grid that you can access to run your selenium automation scripts. For example, you can use [LambdaTest](http://bit.ly/2UZOaa5). It has more than 2000 browser environments over which you can run your tests and truly automate cross-browser testing.

### 

### **What is Selenium WebDriver**

Founded by Simon Stewart in 2006, Selenium WebDriver was the first cross-platform testing framework that could control the browser from OS level. In contrast to IDE, Selenium WebDriver provides a programming interface to create and execute test cases. Test cases are written such that, web elements on web pages are identified and then actions are performed on those elements.

WebDriver is an upgrade to RC because it is much faster. It is faster because it makes direct calls to the browser. RC on the other hand needs an RC server to interact with the web browser. Each browser has its own driver on which the application runs. The different WebDrivers are:

* Firefox Driver (Gecko Driver)
* Chrome Driver
* Internet Explorer Driver
* Opera Driver
* Safari Driver and
* HTM Unit Driver

#### **Benefits Of Selenium WebDriver**

* Support for 7 programming languages: JAVA, C#, PHP, Ruby, Perl, Python and .Net.
* Supports testing on various browsers like: Firefox, Chrome, IE, Safari
* Tests can be performed on different operating systems like: Windows, Mac, Linux, Android, iOS
* Overcomes limitations of Selenium v1 like file upload, download, pop-ups & dialogs barrier

#### **Short-comings Of Selenium WebDriver**

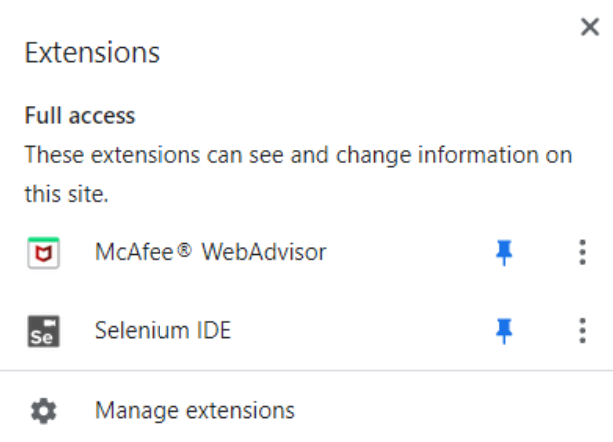
* Detailed test reports cannot be generated
* Testing images is not possible

No matter the challenge, these shortcomings can be overcome by integrations with other frameworks. For testing images, Sikuli can be used, and for generating detailed test reports, TestNG can be used.

**Implementation:**

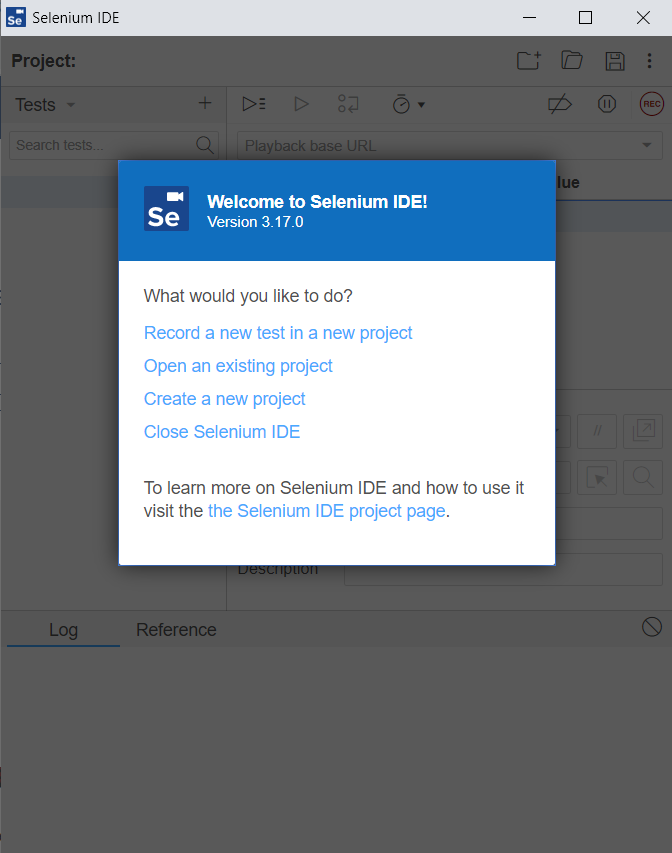
**Steps in creating a Test case for our project:**

**1. Click on the Selenium IDE extension:**

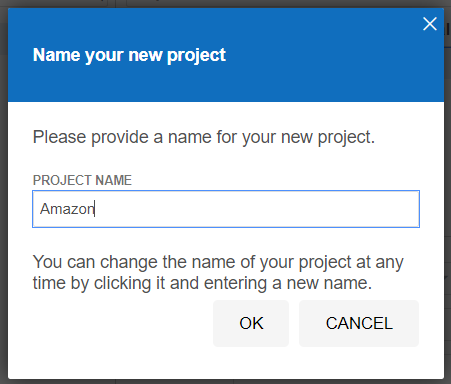
****

****

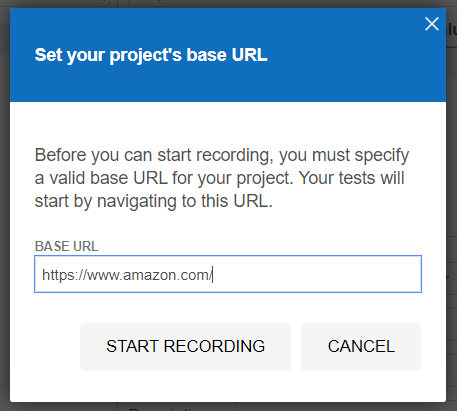
**2. Select “record a new test in a new project”:**

****

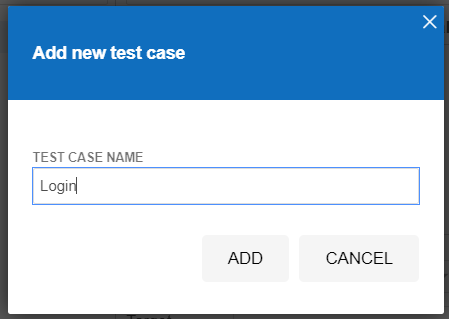
**3. Enter name of your Project:**

****

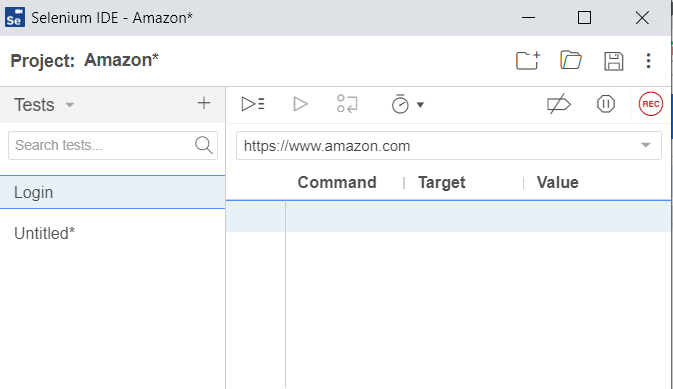
**4. Enter base url and start recording:**

****

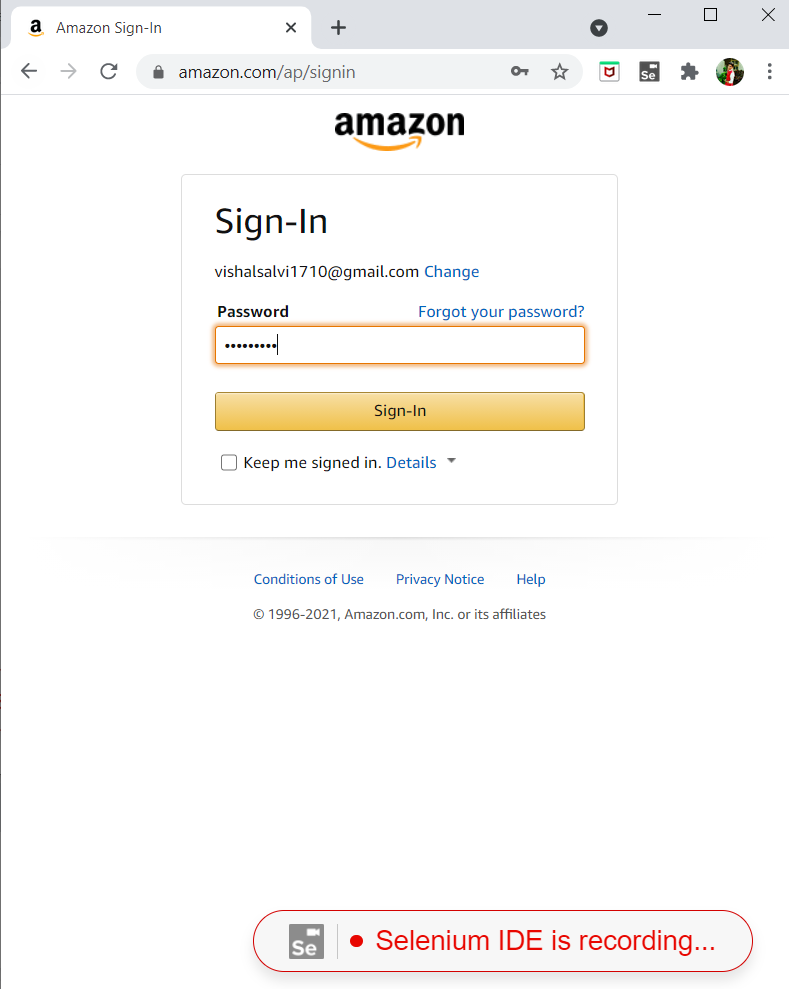
**5. Enter name of test case:**

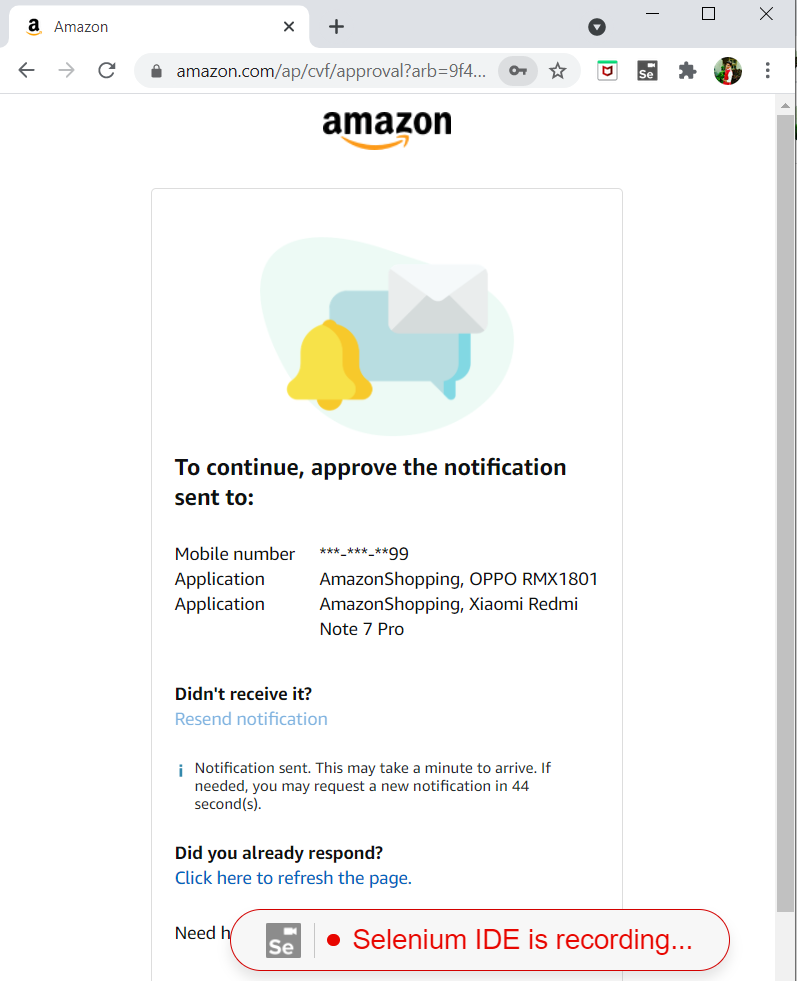
****

**6. Click on start recording**

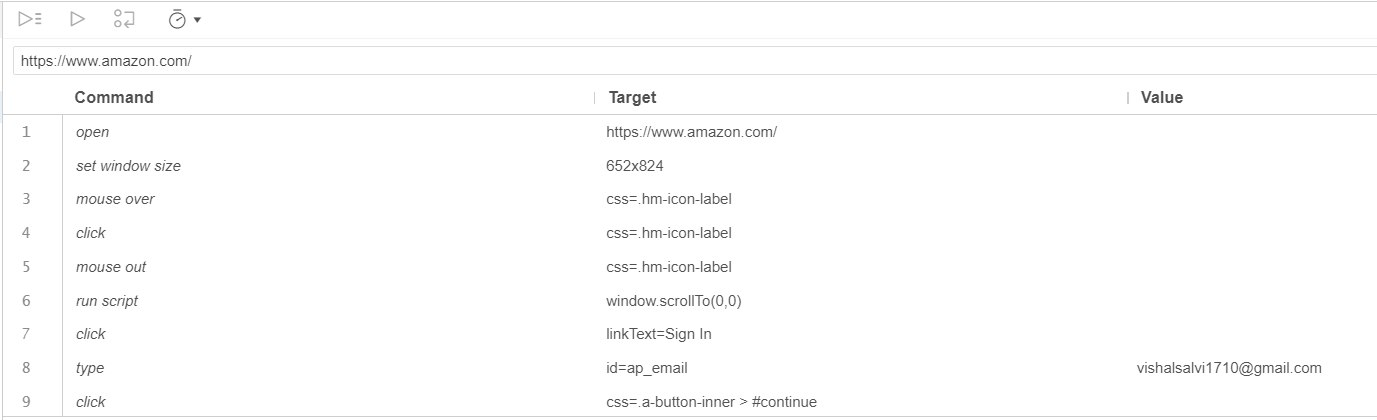
****

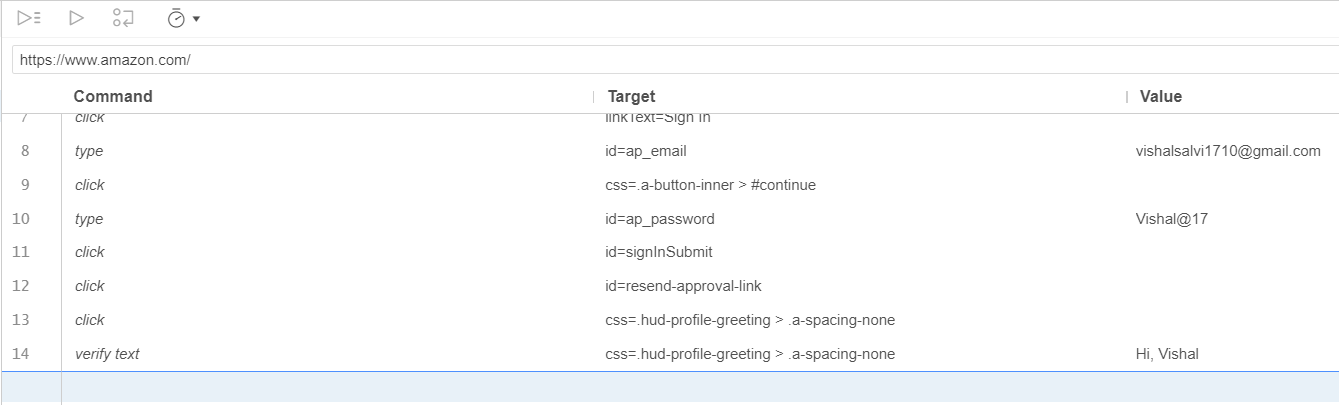
**Log in:**

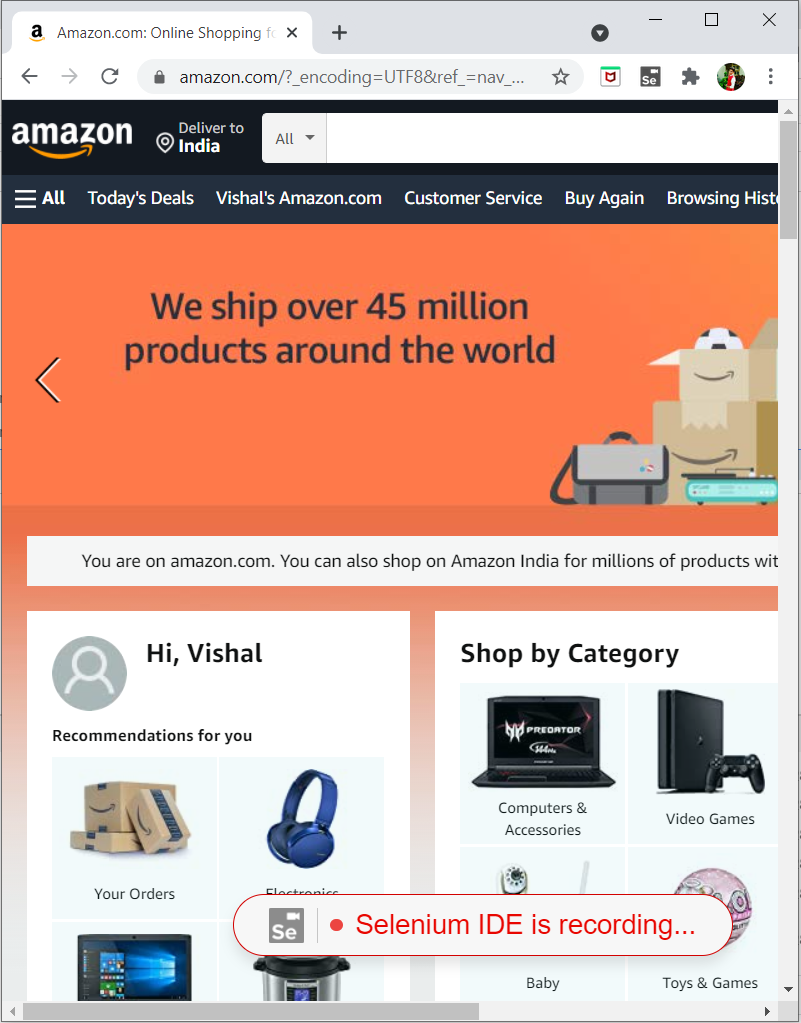
****

****

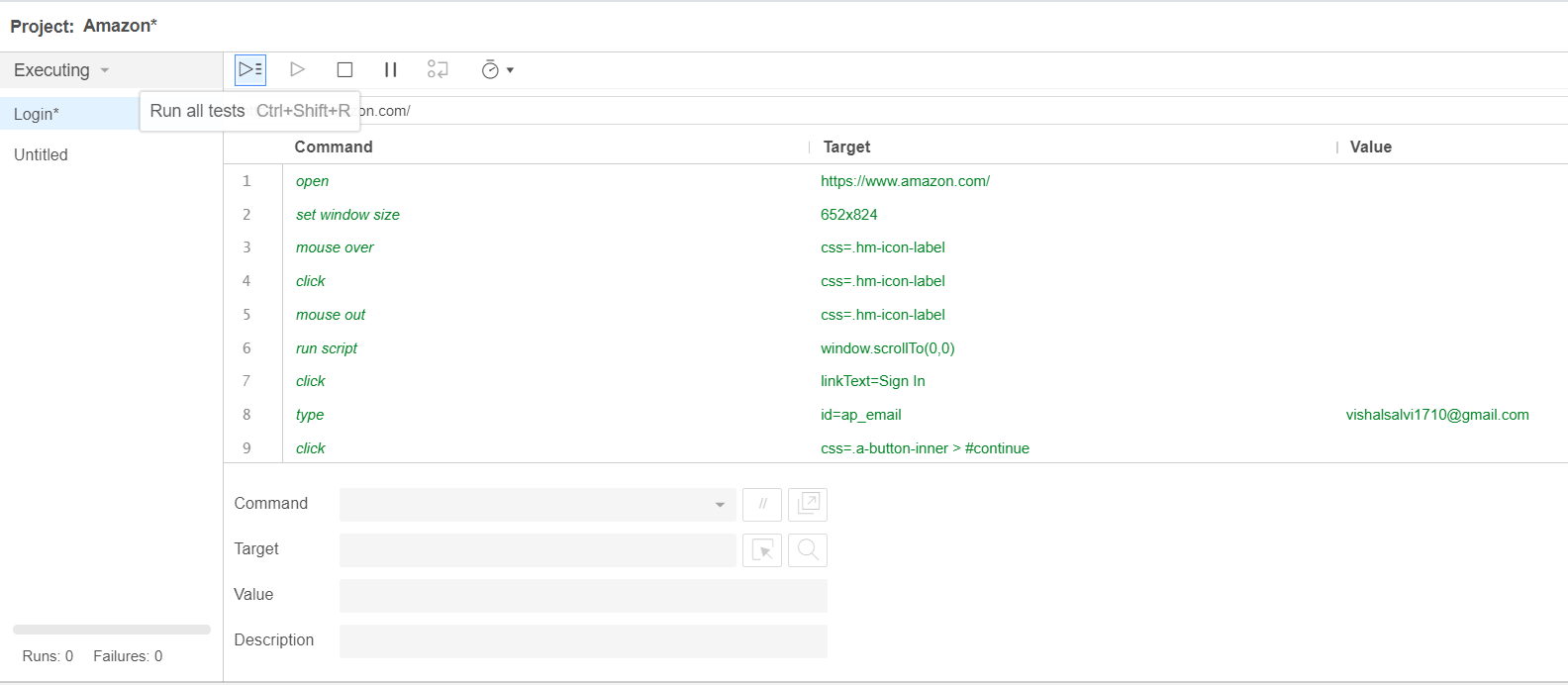
**Command:**

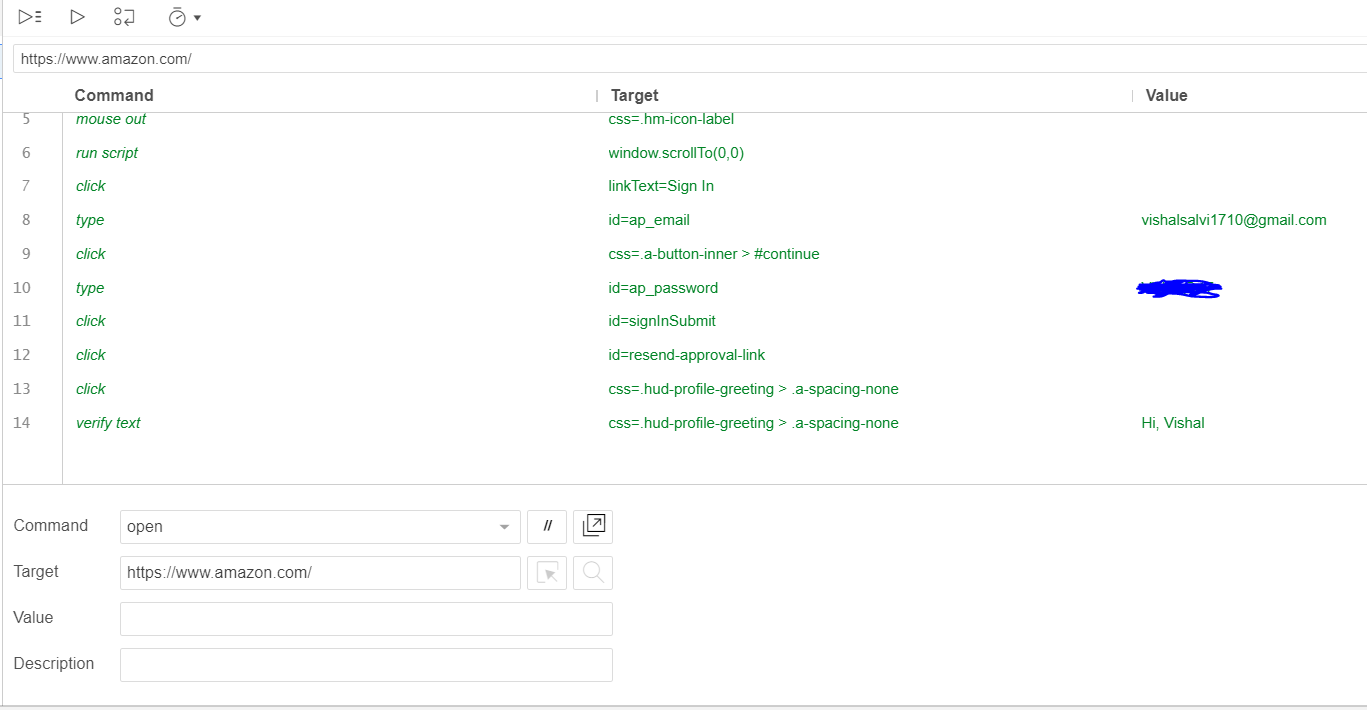
****

****

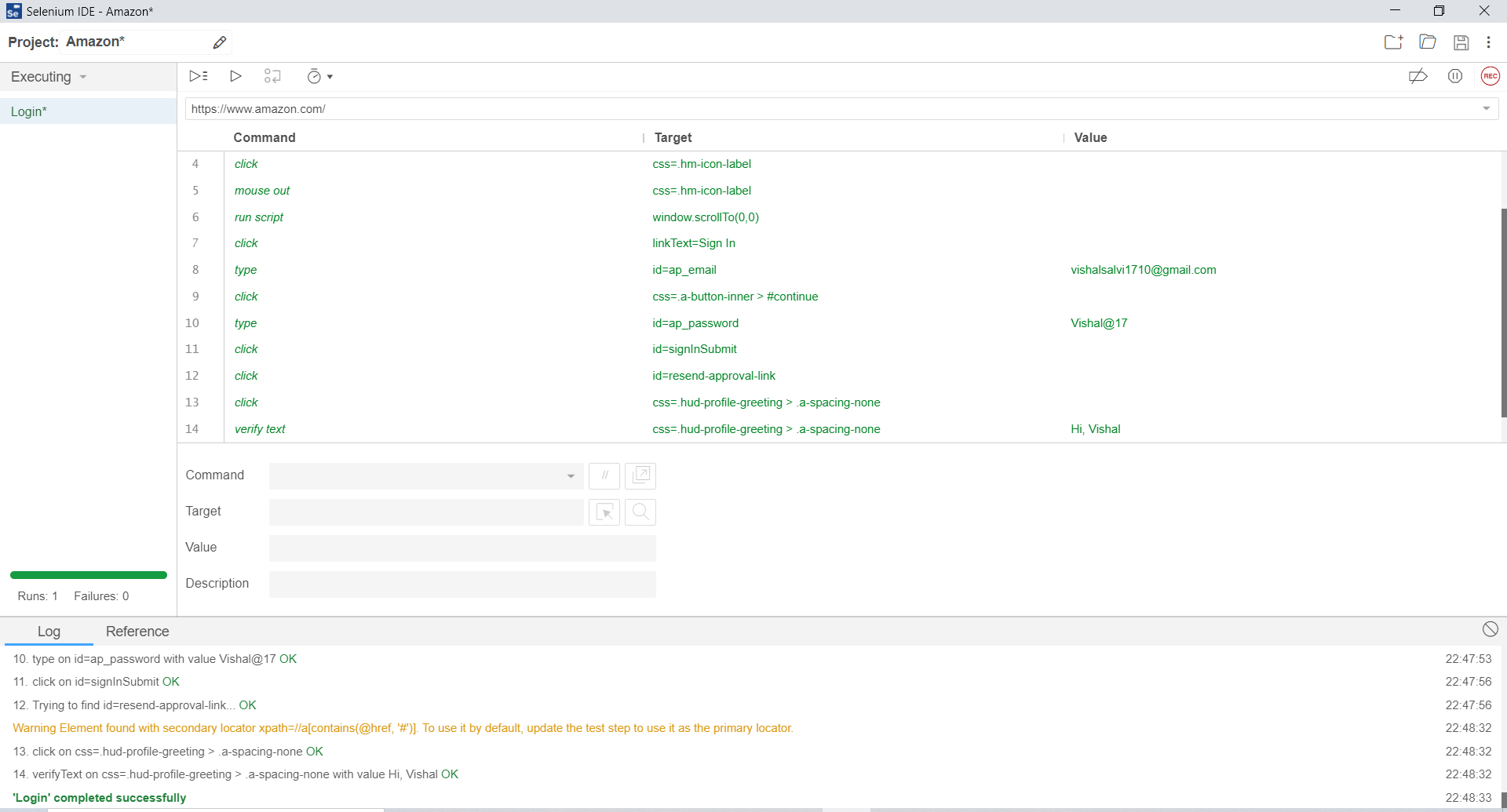
****

**Testing login case:**

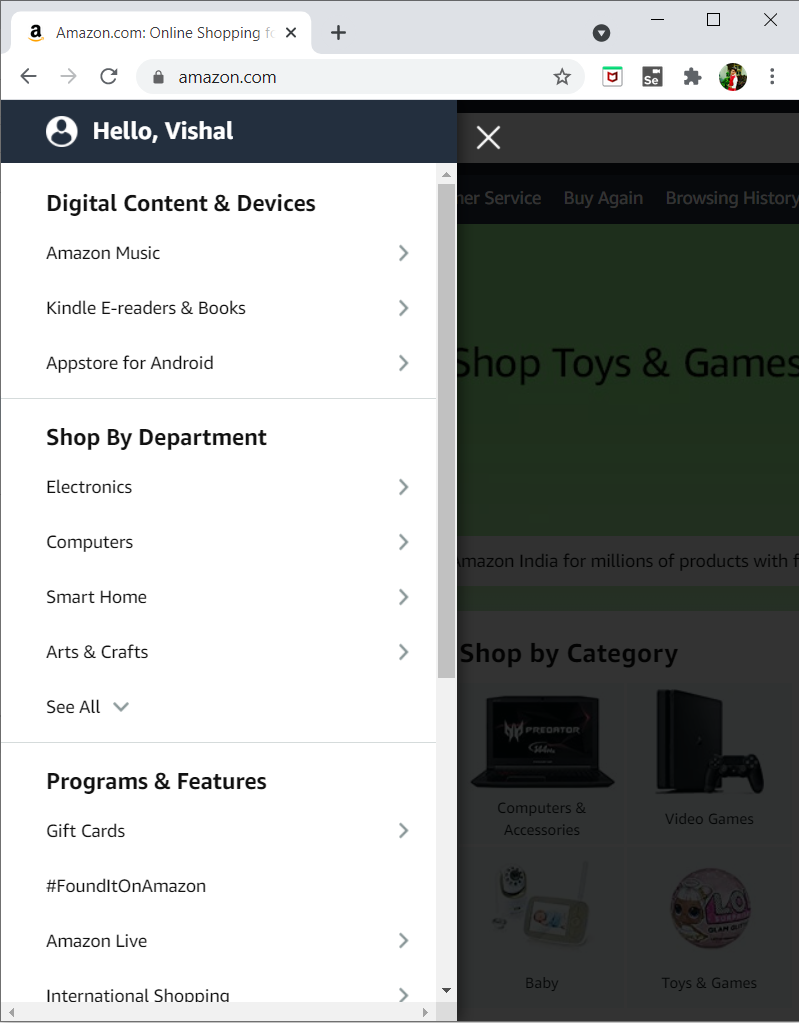
****

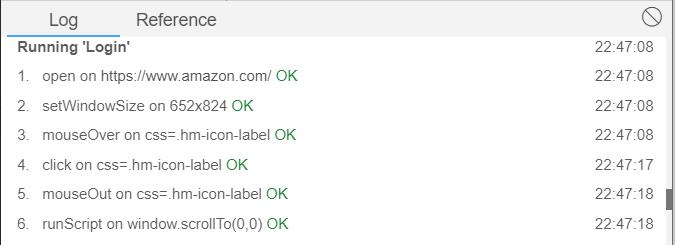
****

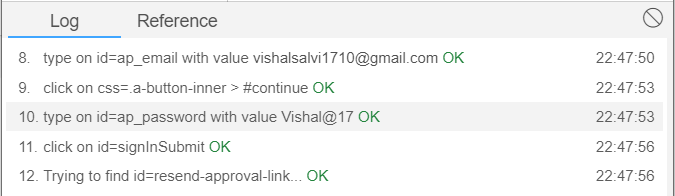
**Test case Successful:**

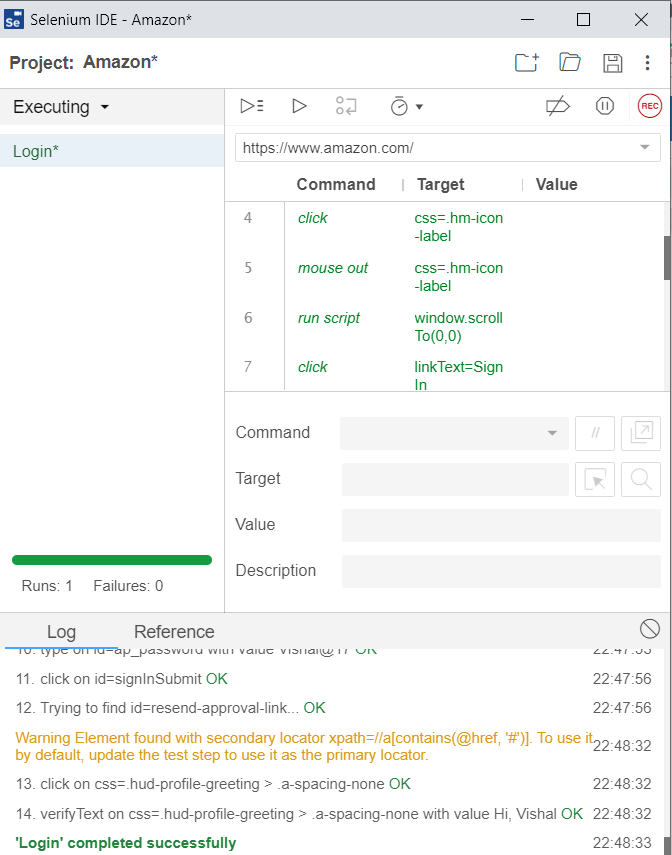
****

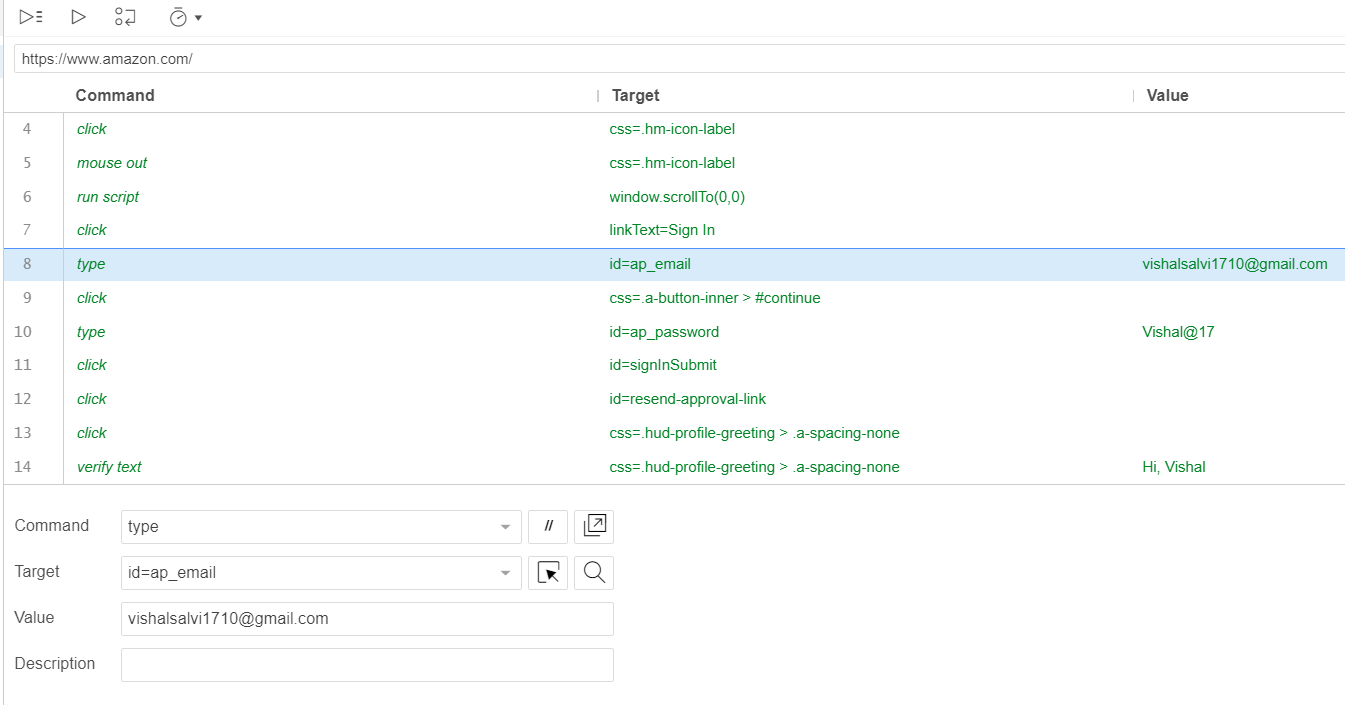
**Result:**

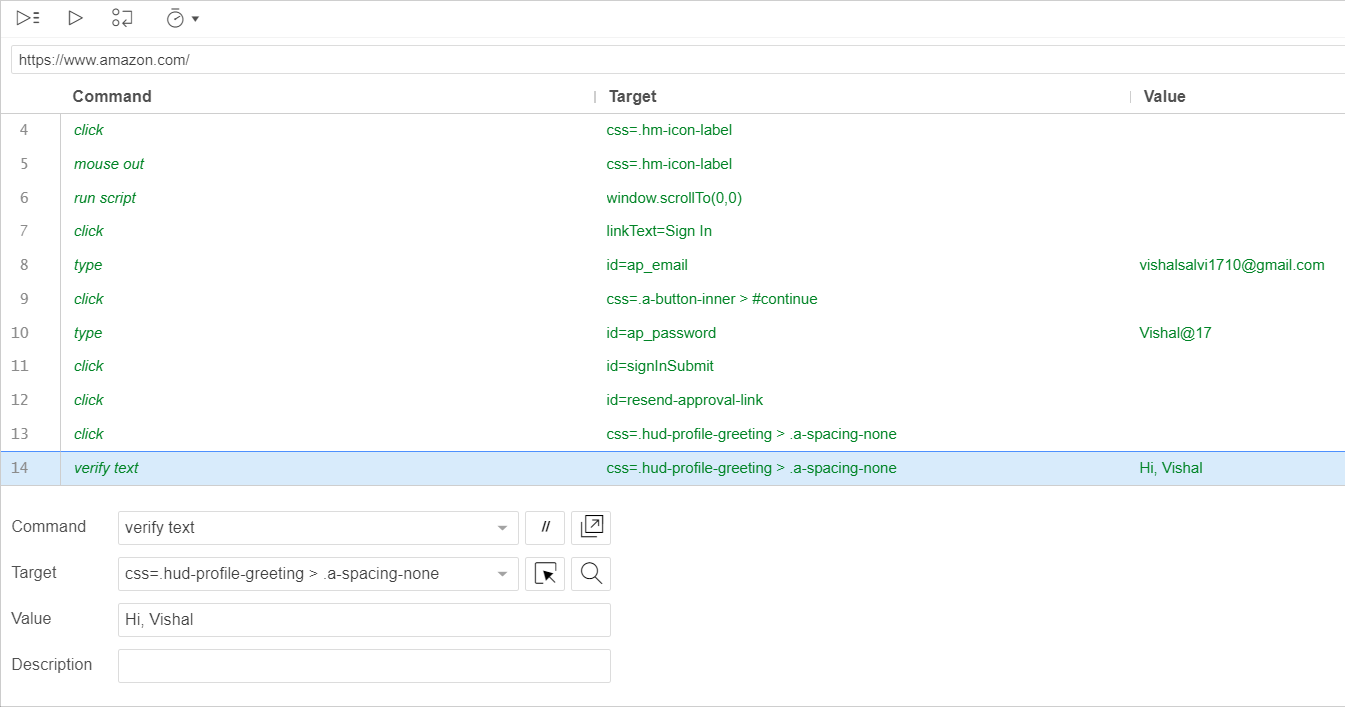
****

****

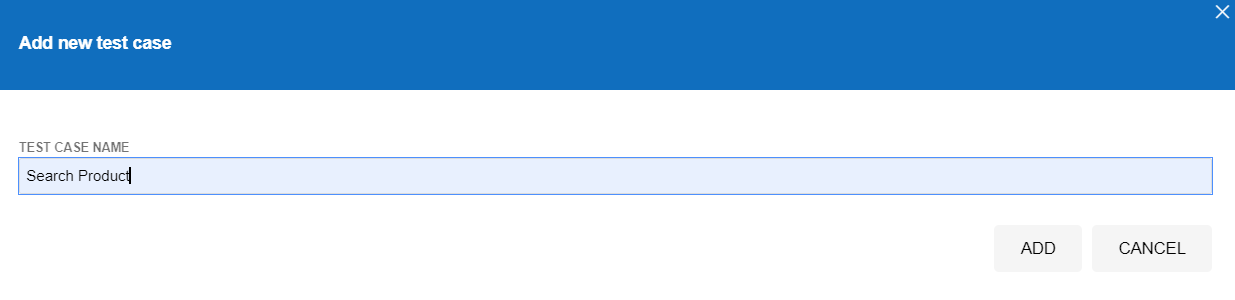
****

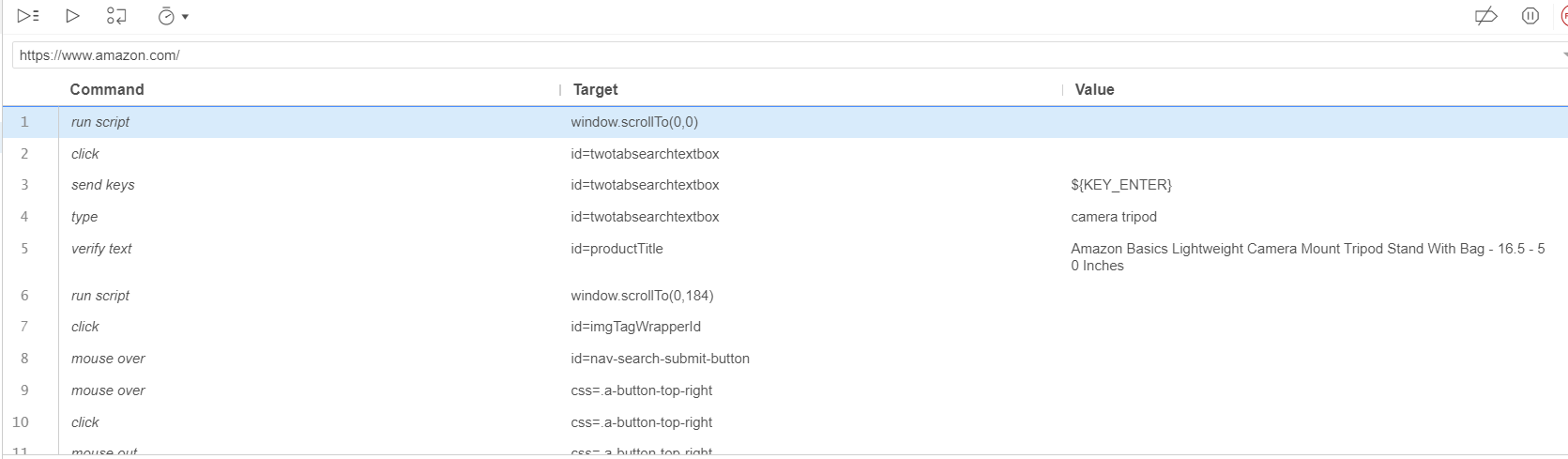
****

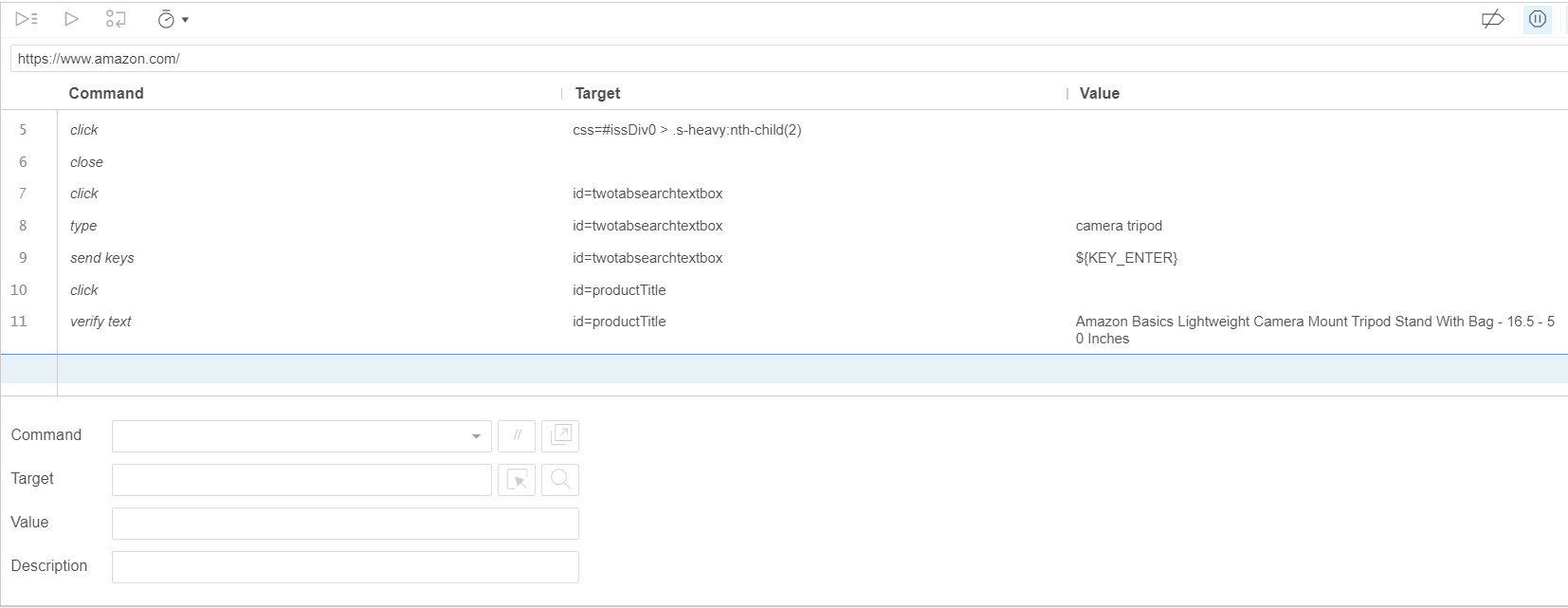
****

****

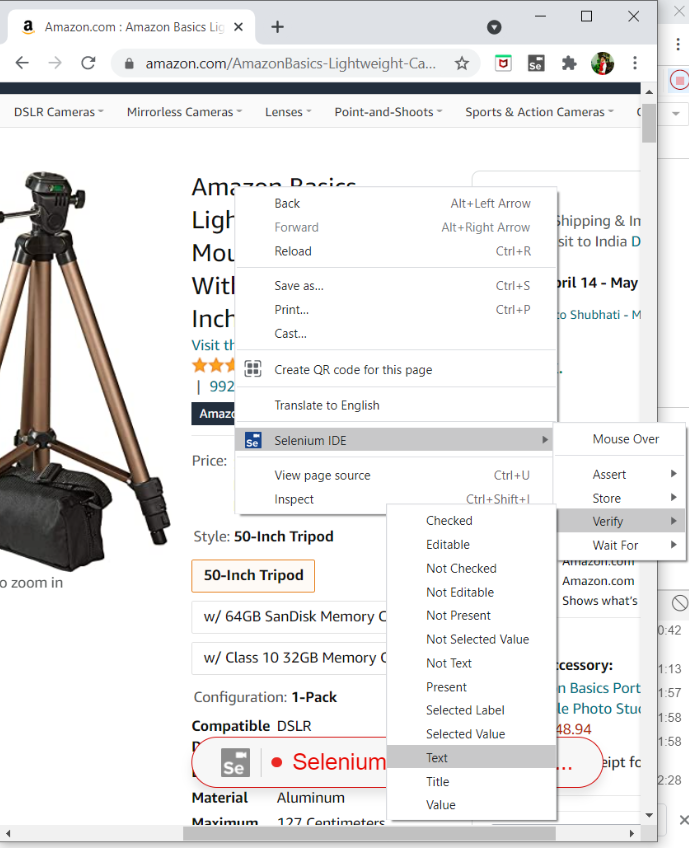
**Add new test case: Search Product**

****

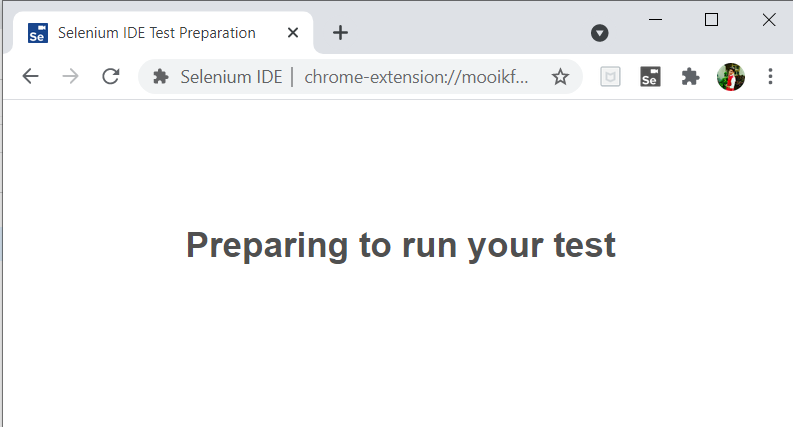
****

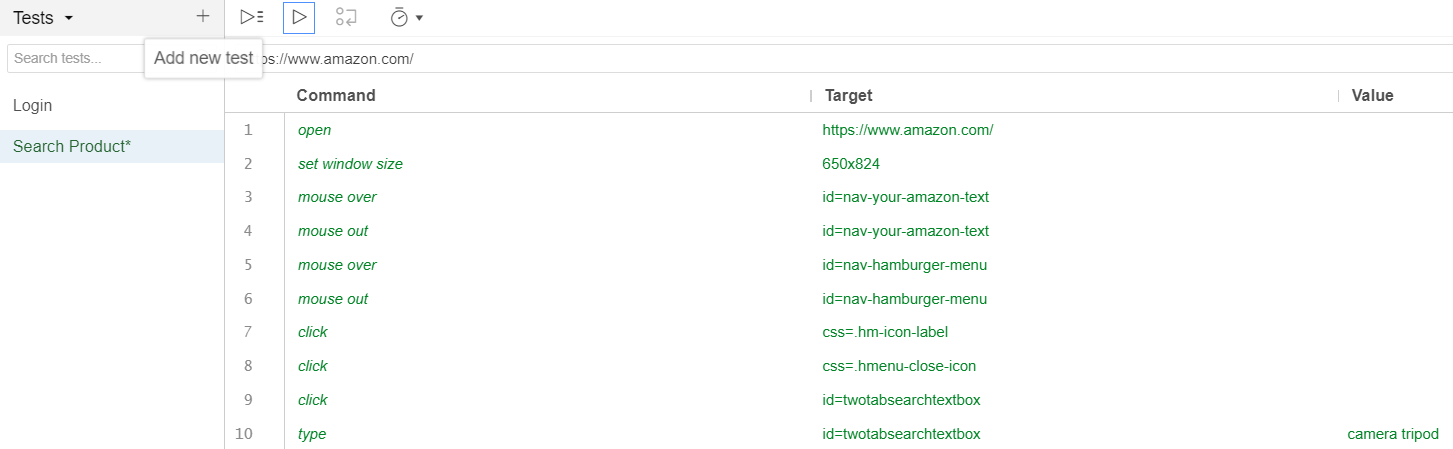
****

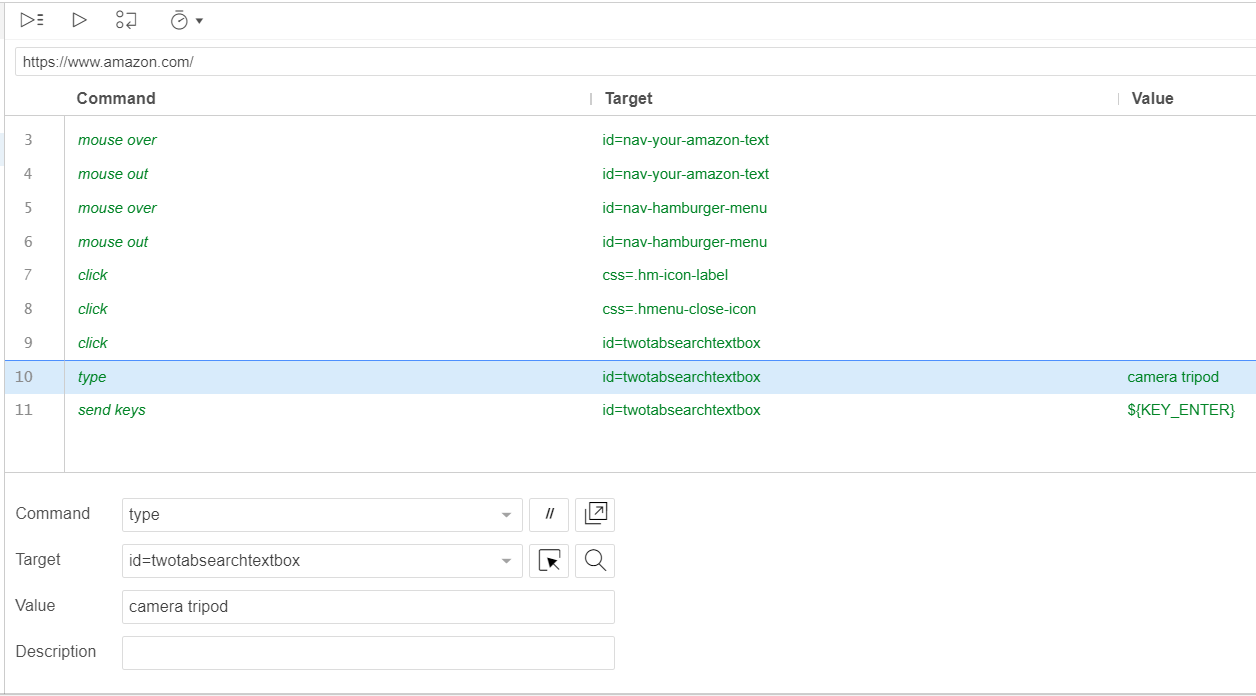
**Verify text:**

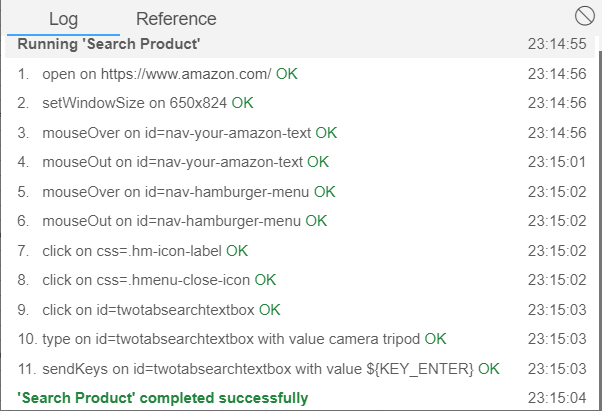
****

**Running test case:**

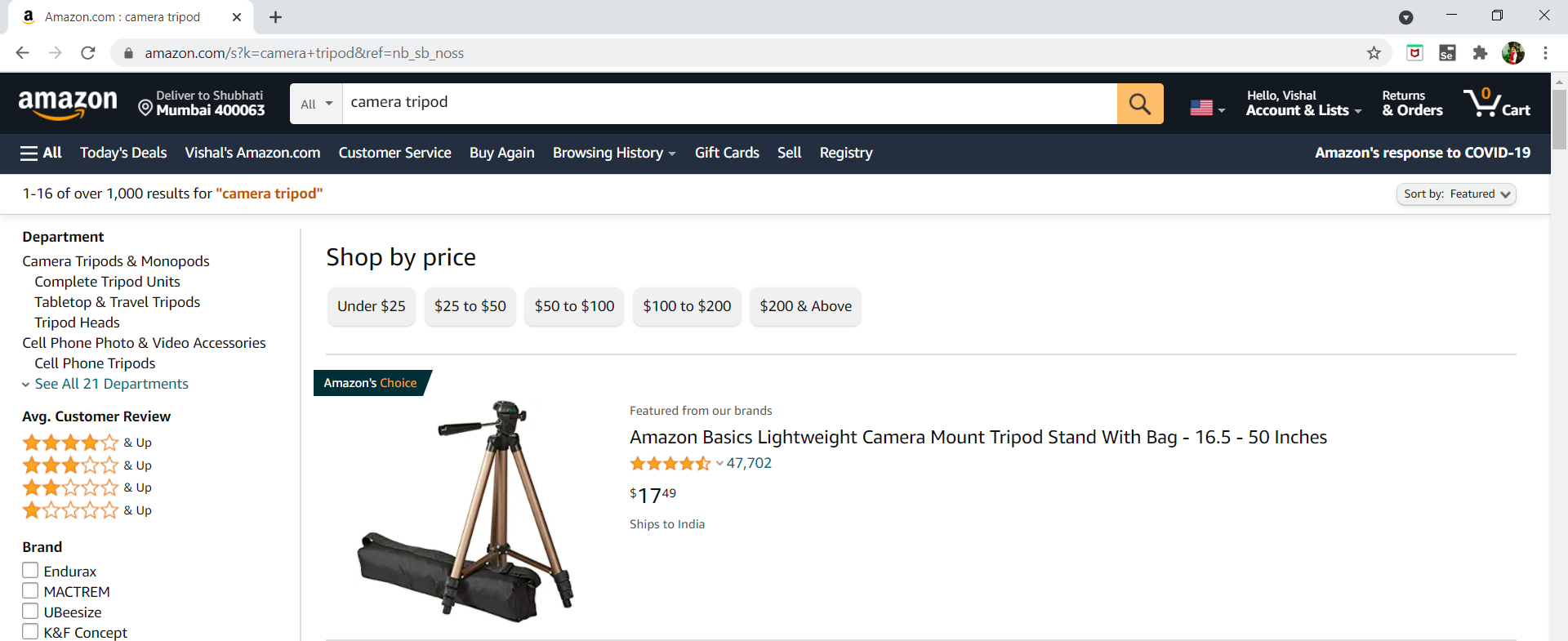
****

****

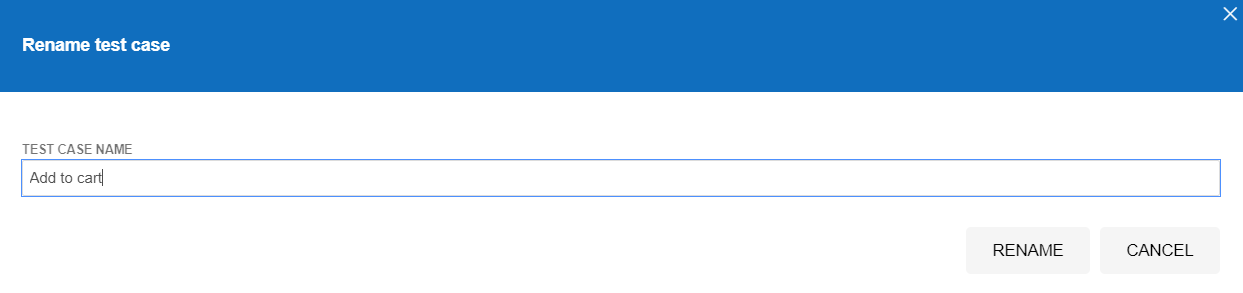
****

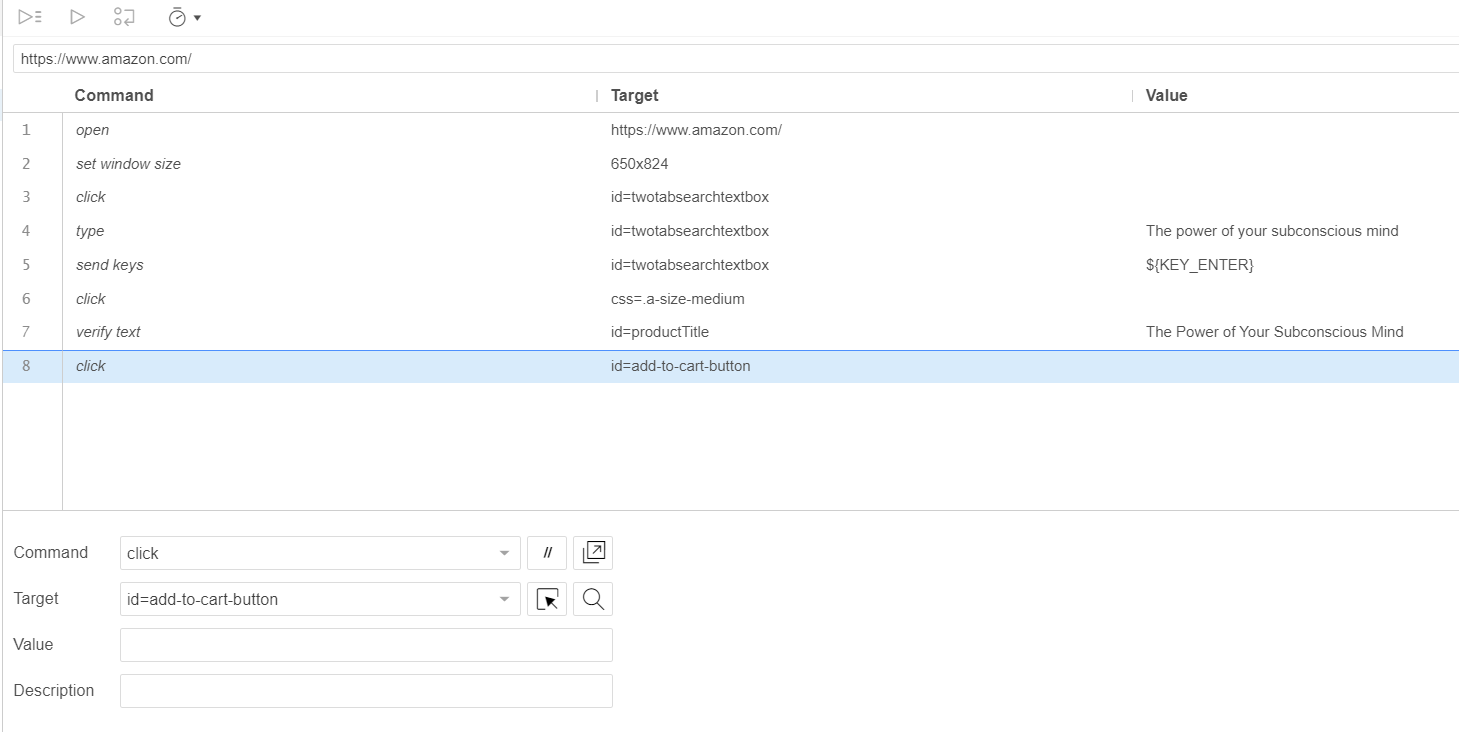
****

**Result of test case: Search Product**

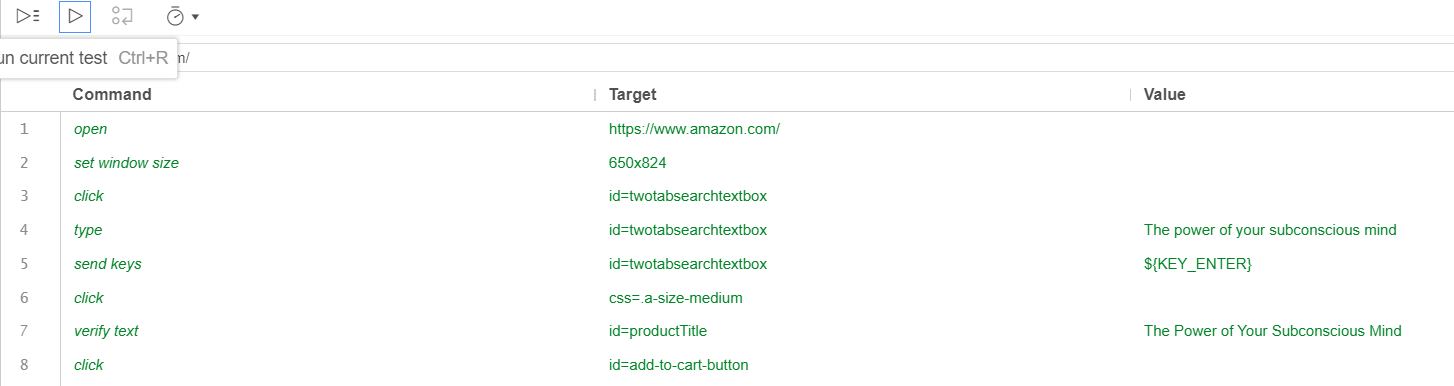
****

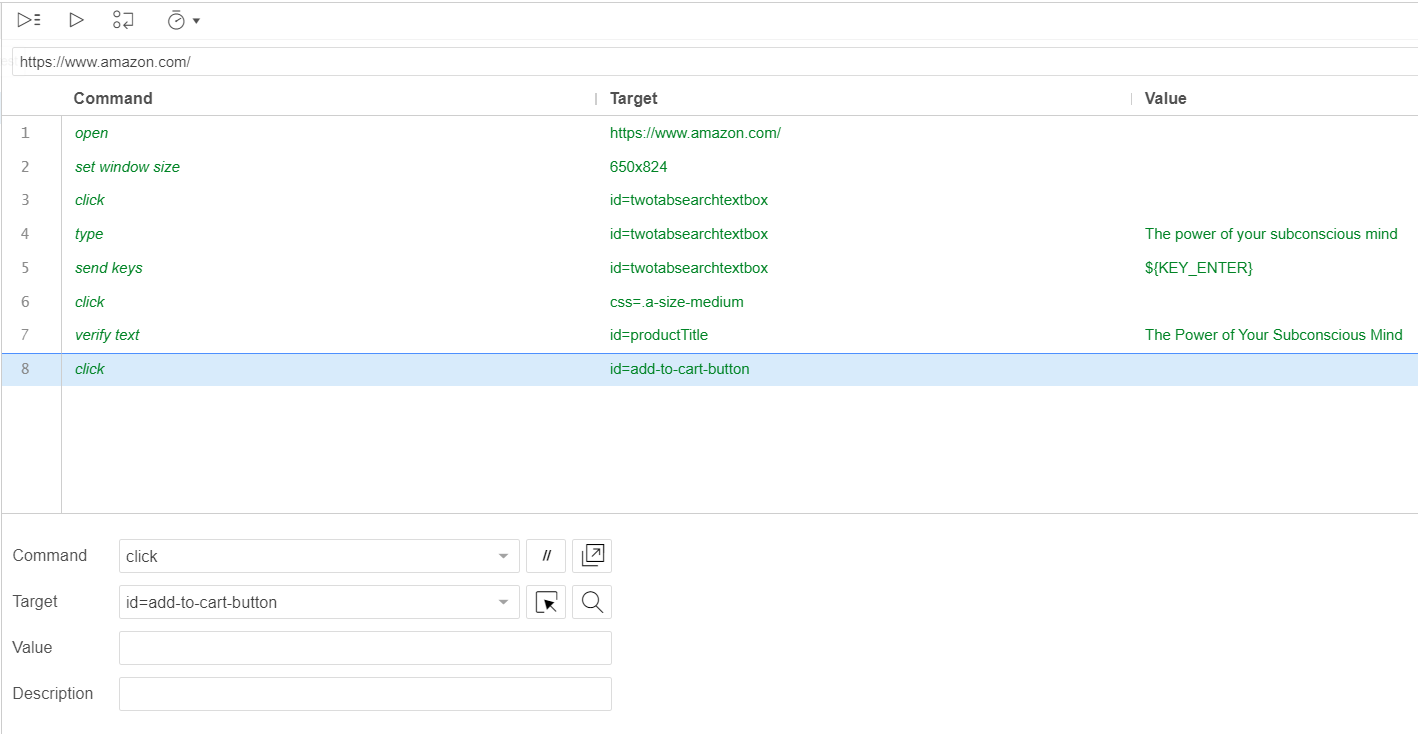
**Add New Test Case: Add to cart**

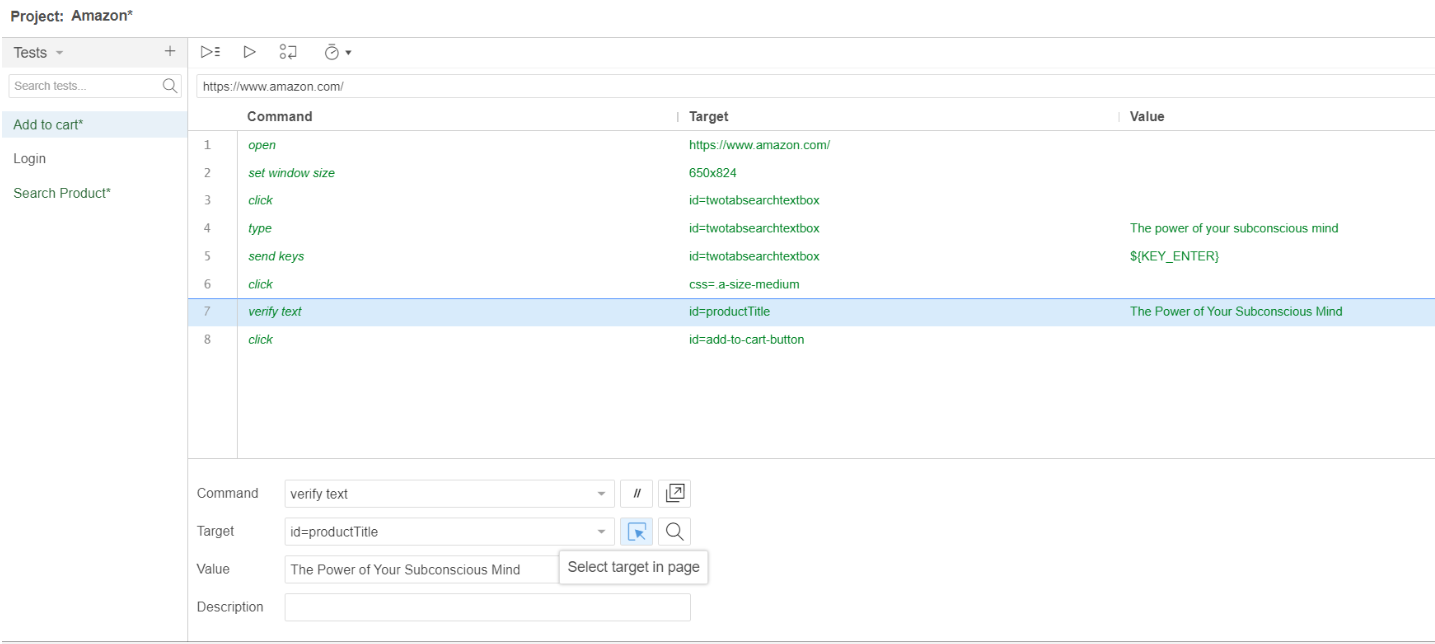
****

****

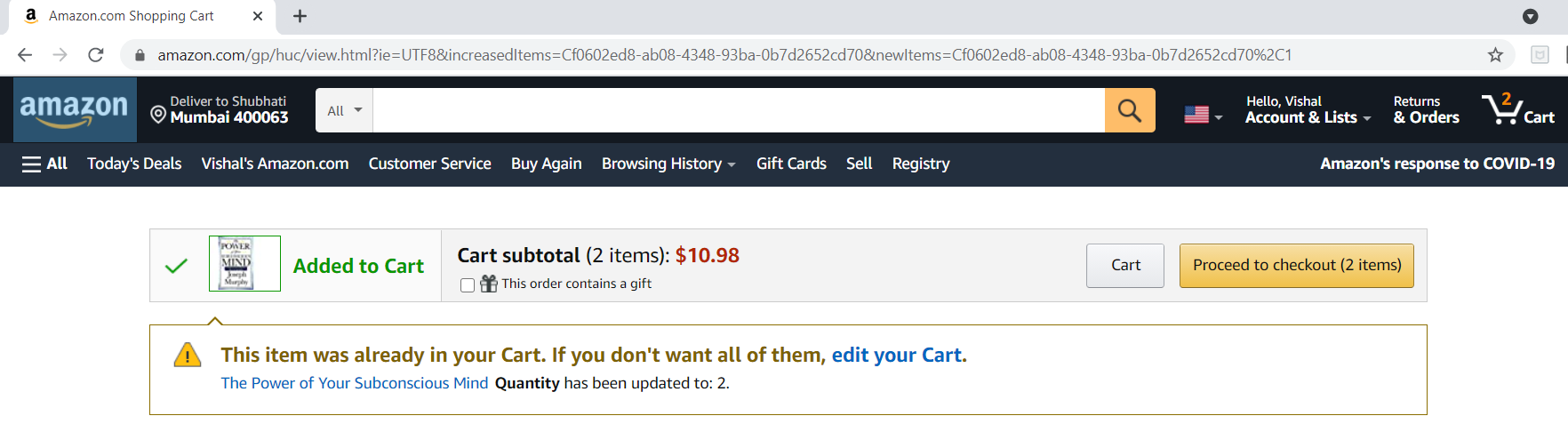
**Run test case:**

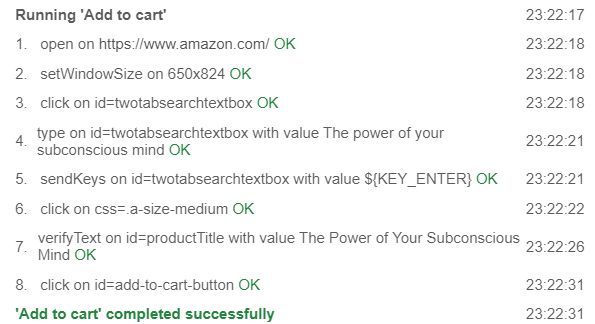
****

****

****

**Result of test case: Add to cart**

****

****

**Conclusion:**

In this experiment, we understood the concept of automation testing in a project. We implemented automated testing for our project amazon.com.Selenium IDE does have some limitations as a testing framework, but is a valuable tool in its own right and deserves a place in the toolbox of any testing team.

**References:**

**1)**[**https://www.edureka.co/blog/what-is-selenium/**](https://www.edureka.co/blog/what-is-selenium/)