

**ĐẠI HỌC QUỐC GIA HÀ NỘI**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**



**Đoàn Phương Thảo**

# **KIỂM THỬ TỰ ĐỘNG TRÊN ỨNG DỤNG WEB**

**KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY**

**Ngành: Khoa học máy tính**

**HÀ NỘI - 2019**

**ĐẠI HỌC QUỐC GIA HÀ NỘI  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

**Đoàn Phương Thảo**

# **KIỂM THỬ TỰ ĐỘNG TRÊN ỨNG DỤNG WEB**

**KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY**

**Ngành: Khoa học máy tính**

**Cán bộ hướng dẫn: TS. Đặng Văn Hưng**

**HÀ NỘI - 2019**

# TÓM TẮT

## **Tóm tắt:**

Trong xã hội hiện tại, xu hướng tự động hóa đang được triển khai rộng rãi ở nhiều lĩnh vực, trong đó có kiểm thử phần mềm. Đặc biệt, khi mà kiểm thử phần mềm vẫn tiêu tốn một lượng lớn thời gian, kinh phí và nhân lực trong một dự án phần mềm thì sự ra đời của các công cụ hỗ trợ kiểm thử tự động như Selenium, Quick Test Professional, NUnit, JUnit, Load Runner (thường dùng trong kiểm thử hiệu năng) càng có ý nghĩa hơn bao giờ hết. Selenium được biết đến là một bộ các công cụ kiểm thử tự động các ứng dụng Web, có thể kiểm thử trên nhiều trình duyệt, hỗ trợ nhiều ngôn ngữ lập trình, giao tiếp được với các công cụ kiểm thử khác nhau như Junit, TestNG (với Java) hay NUnit (với C#) và đặc biệt công cụ này là một bộ mã nguồn mở, do đó các tổ chức không cần tốn kinh phí mua bản quyền. Công cụ Selenium hiện được đánh giá là một trong những công cụ tốt nhất cho kiểm thử tự động các ứng dụng Web.

Với mong muốn có cái nhìn chính xác, rõ ràng hơn về quy trình kiểm thử phần mềm, đảm bảo chất lượng phần mềm và tiếp cận, sử dụng hiệu quả các công cụ kiểm thử tự động Selenium, đồng thời rèn kỹ năng làm việc tạo tiền đề định hướng cho tương lai sau khi ra trường nên em chọn đề tài “Kiểm thử tự động trên ứng dụng Web.”

**Từ khóa:** Tự động hóa, kiểm thử tự động, Selenium.

## LỜI CẢM ƠN

Để hoàn thành khóa luận này, em xin gửi lời cảm ơn sâu sắc nhất đến các thầy cô Trường Đại học Công Nghệ, đặc biệt là các thầy cô khoa Công Nghệ Thông Tin, những người đã dạy dỗ, chỉ bảo em trong suốt bốn năm đại học cũng như suốt quá trình làm khóa luận.

Em xin gửi lời cảm ơn đặc biệt đến giảng viên TS. Đặng Văn Hưng, người trực tiếp hướng dẫn tận tình, chu đáo để em có thể hoàn thành khóa luận này. Sau quá trình được thầy hướng dẫn, em đã học hỏi, đúc kết được nhiều kinh nghiệm quý báu.

Trong suốt quá trình làm khóa luận, em muốn gửi lời cảm ơn đến tập thể lớp K60–CAC, những người bạn đã luôn bên cạnh động viên, giúp đỡ em vượt qua khó khăn, thất bại. Bốn năm bên nhau không phải là quãng thời gian dài nhưng cũng không hề ngắn để xây dựng tình bạn, quãng thời gian tươi đẹp dưới mái trường đại học của chúng ta.

Cuối cùng, em xin được cảm ơn gia đình đã nuôi em khôn lớn để trở thành người có ích cho xã hội, là điểm tựa vững chắc, giúp em trưởng thành, vững bước trên con đường mình đã chọn.

Dù đã rất cố gắng trong quá trình tìm hiểu và hoàn thành khóa luận, nhưng do hạn chế về mặt thời gian cũng như những thiếu sót về kinh nghiệm và kiến thức, nên khóa luận không thể tránh khỏi những thiếu sót, em rất mong nhận được những đóng góp của thầy cô để khóa luận hoàn chỉnh.

Em xin chân thành cảm ơn!

Hà Nội, ngày 18 tháng 04 năm 2019

Sinh viên

*Đoàn Phương Thảo*

## **LỜI CAM ĐOAN**

Em xin cam đoan rằng những nghiên cứu về kiểm thử tự động bằng Selenium được trình bày trong luận án này là của em và chưa từng được nộp như một báo cáo khóa luận tại trường Đại học Công Nghệ - Đại học Quốc Gia Hà Nội hoặc bất kỳ trường đại học khác. Những gì em viết ra không sao chép từ các tài liệu, không sử dụng các kết quả của người khác mà không trích dẫn cụ thể.

Em xin cam đoan công cụ kiểm thử tự động em trình bày trong khoá luận là do em tự phát triển, không sao chép mã nguồn của người khác. Nếu sai em hoàn toàn chịu trách nhiệm theo quy định của trường Đại học Công Nghệ - Đại học Quốc Gia Hà Nội.

Hà Nội, tháng 04 năm 2019

Sinh viên

Đoàn Phương Thảo

## BẢNG THUẬT NGỮ VÀ TỪ VIẾT TẮT

Số thứ tự	Thuật ngữ và từ viết tắt	Mô tả
1	URL	Uniform Resource Locator
2	HTTP	Hypertext Transfer Protocol
3	HTML	HyperText Markup Language, hay là "Ngôn ngữ Đánh dấu Siêu văn bản"
4	RC	Remote Control
5	IDE	Integrated development environment
6	QTP	Quick Test Professional
7	Test case	Cá kiểm thử
8	UI	User Interface, hay là “Thiết kế giao diện người dùng”
9		
10		
11		
12		
13		
14		
15		
16		
17		

## DANH MỤC HÌNH ẢNH VÀ BẢNG BIỂU

Hình 1	Kiến trúc chung của một bộ kiểm thử tự động.....	6
Hình 2:	Các công cụ kiểm thử.....	11
Hình 3:	Kiến trúc Selenium RC.....	15
Hình 4:	Các thành phần cơ bản của Selenium IDE.....	17
Hình 5:	Webdriver tương tác trình duyệt.....	20
Hình 6:	Minh họa về hub và node của Selenium Grid.....	23
Hình 7:	Giao diện trang web Tiki.vn .....	26
Hình 8:	Màn hình tạo mới project Selenium IDE.....	31
Hình 9:	Màn hình tạo mới project Selenium IDE.....	32
Hình 10:	Màn hình nhập địa chỉ trang cần test Tiki.vn.....	32
Hình 11:	Màn hình chính trang Tiki.vn và vị trí chọn đăng nhập.....	33
Hình 12:	Trang đăng nhập và assert Text bằng Selenium IDE.....	33
Hình 13:	Kết quả chạy các test case cho trang đăng nhập bằng Selenium IDE.....	34
Hình 14:	Nhập từ cần tìm kiếm và nhấn tìm kiếm .....	34
Hình 15:	Lựa chọn mặt hàng Điện thoại – Máy tính bảng.....	35
Hình 16:	Click lựa chọn mặt hàng Ipad wifi .....	36
Hình 17:	Click button chọn mua Ipad wifi.....	36
Hình 18:	Kiểm tra thông tin mặt hàng đã chọn .....	37
Hình 19:	Kết quả thực hiện kiểm tra chức năng chọn hàng, xóa hàng .....	37
Hình 20:	Cấu trúc source code sử dụng Selenium WebDriver.....	38
Hình 21:	Thông tin các user để test .....	39
Hình 22:	Thông tin locatar xác định vị trí các phần tử trên trang web .....	39
Hình 23:	Thông tin của các test case, kết quả mong muốn .....	40
Hình 24:	Ví dụ một test case cho trang đăng nhập.....	40
Hình 25:	Kết quả các test case trang đăng nhập.....	41
Hình 26:	Các test case cho chức năng tìm kiếm.....	42
Hình 27:	Kết quả test case chức năng tìm kiếm.....	43
Hình 28:	Vị trí các phần tử web cho chức năng thêm hàng và xóa hàng.....	43
Hình 29:	Source code chọn một mặt hàng bằng webdriver.....	44
Hình 30:	Source code sinh test case chọn mặt hàng webdriver .....	44
Hình 31:	Kết quả test case chọn mặt hàng .....	44
Hình 32:	Source code kiểm thử selenium grid cho nhiều trình duyệt.....	45
Bảng 1:	So sánh Selenium với QTP .....	12
Bảng 2:	Ưu điểm QTP so với Selenium .....	13
Bảng 3:	So sánh hai phiên bản Selenium Grid .....	22
Bảng 4:	Các ca kiểm thử chức năng đăng nhập.....	28
Bảng 5:	Các ca kiểm thử cho chức năng tìm kiếm.....	30

# MỤC LỤC

<b>TÓM TẮT .....</b>	<b>i</b>
<b>LỜI CẢM ƠN .....</b>	<b>ii</b>
<b>LỜI CAM ĐOAN .....</b>	<b>iii</b>
<b>BẢNG THUẬT NGỮ VÀ TỪ VIẾT TẮT .....</b>	<b>iv</b>
<b>DANH MỤC HÌNH ẢNH VÀ BẢNG BIỂU.....</b>	<b>v</b>
<b>MỤC LỤC.....</b>	<b>vi</b>
<b>CHƯƠNG 1. GIỚI THIỆU .....</b>	<b>1</b>
1.1. Đặt vấn đề .....	1
1.2. Mục tiêu của đề tài .....	1
1.3. Các vấn đề được giải quyết.....	1
1.4. Bố cục của khóa luận.....	2
1.5. Phương pháp nghiên cứu .....	2
1.5.1. Phương pháp nghiên cứu lý thuyết.....	2
1.5.2. Phương pháp nghiên cứu thực nghiệm .....	2
<b>CHƯƠNG 2. TỔNG QUAN VỀ CÔNG CỤ KIỂM THỬ .....</b>	<b>3</b>
<b>TỰ ĐỘNG SELENIUM .....</b>	<b>3</b>
2.1. Kiến thức liên quan .....	3
2.1.1. Kiểm thử phần mềm.....	3
2.1.1.1. Định nghĩa về kiểm thử phần mềm.....	3
2.1.1.2. Mục đích của kiểm thử phần mềm .....	3
2.1.1.3. Các kỹ thuật kiểm thử phần mềm.....	3
2.1.1.4. Các mức kiểm thử:.....	4
2.1.1.5. Quy trình kiểm thử.....	5
2.1.2. Kiểm thử tự động .....	5



2.1.2.1.	Tổng quan về kiểm thử tự động.....	5
2.1.2.2.	Ưu nhược điểm của kiểm thử tự động .....	6
2.1.2.3.	Một số công cụ kiểm thử tự động.....	7
2.2.	Công cụ kiểm thử Selenium.....	9
2.2.1.	Tổng quan .....	9
2.2.2.	Một số đặc điểm của Selenium.....	9
2.2.3.	Các thành phần của Selenium.....	10
2.2.4.	So sánh giữa Selenium và QTP (Quick Test Professional).....	11
CHƯƠNG 3.	CÁC CÔNG CỤ SELENIUM VÀ ỨNG DỤNG .....	13
3.1.	SELENIUM REMOTE CONTROL (RC).....	14
3.1.1.	Tổng quan .....	14
3.1.2.	Ưu nhược điểm của RC .....	15
3.2.	SELENIUM IDE .....	16
3.2.1.	Tổng quan .....	16
3.2.2.	Ưu nhược điểm của Selenium IDE .....	16
3.2.3.	Các thành phần cơ bản của Selenium IDE .....	17
3.3.	SELENIUM WEBDRIVER .....	19
3.3.1.	Tổng quan .....	19
3.3.2.	Ưu nhược điểm của Selenium WebDriver .....	20
3.3.3.	Các thành phần cơ bản của Selenium WebDriver.....	21
3.4.	SELENIUM GRID .....	22
3.4.1.	Tổng quan .....	22
3.4.2.	Ưu nhược điểm của Selenium Grid .....	23
CHƯƠNG 4.	DEMO VÀ PHƯƠNG PHÁP LUẬN SỬ DỤNG CÁC CÔNG CỤ SELENIUM.....	25
4.1.	Mô tả bài toán.....	25

4.1.1.	Giới thiệu bài toán.....	25
4.1.2.	Hướng giải quyết bài toán .....	26
4.1.2.1.	Tổng quát.....	26
4.1.2.2.	Xây dựng ca kiểm thử cho các chức năng đăng nhập, tìm kiếm, đặt hàng, xóa hàng.     26	
4.2.	Áp dụng cho Selenium IDE .....	31
4.2.1.	Chức năng đăng nhập .....	31
4.2.2.	Chức năng tìm kiếm .....	34
4.2.3.	Chức năng đặt hàng và xóa hàng .....	35
4.3.	Áp dụng cho Selenium WebDriver .....	38
4.3.1.	Chức năng đăng nhập .....	38
4.3.2.	Chức năng tìm kiếm .....	41
4.3.3.	Chức năng đặt hàng, xóa hàng.....	43
4.4.	Áp dụng cho Selenium Grid. ....	45
4.5.	Đánh giá và đưa ra phương pháp luận.....	46
KẾT LUẬN .....		47
TÀI LIỆU THAM KHẢO .....		48

# CHƯƠNG 1. GIỚI THIỆU

Chương sẽ trình bày một cách tổng quan về mục tiêu, bố cục của khóa luận cũng như những vấn đề, phương pháp mà khóa luận giải quyết.

## 1.1. Đặt vấn đề

Theo các con số được thống kê, trong quá trình phát triển phần mềm, công việc kiểm thử thường chiếm từ 11% - 40% chi phí. Các dự án phần mềm đều mong muốn giảm chi phí về mặt thời gian, nhân lực mà vẫn đem lại hiệu quả cao và chất lượng tốt, do đó kiểm thử tự động ngày càng được áp dụng rộng rãi.

Có rất nhiều công cụ kiểm thử tự động phổ biến hiện nay, nhưng Selenium được đánh giá là một trong những công cụ tốt nhất để kiểm thử tự động một ứng dụng Web. Selenium cung cấp chức năng ghi tự động và phát lại, hỗ trợ hữu ích cho kiểm thử hồi quy. Điểm mạnh của Selenium là hỗ trợ trên nhiều nền tảng khác nhau, tích hợp trên nhiều trình duyệt, có thể thực hiện nhiều ca kiểm thử cùng lúc, có khả năng lưu các ca kiểm thử để sử dụng lại khi cần và cho phép người dùng chèn chú thích giữa kịch bản kiểm thử để hiểu rõ hơn nội dung kiểm thử. Selenium cũng hỗ trợ một lượng lớn các ngôn ngữ lập trình Web phổ biến hiện nay như C#, Java, Perl, PHP, Python, Ruby,... Selenium có thể kết hợp với một số công cụ khác nhưng với người dùng thông thường chỉ cần chạy tự động mà không cần cài thêm các công cụ hỗ trợ.

Selenium là một bộ các công cụ, mỗi công cụ đều có ưu nhược điểm khác nhau và cách sử dụng riêng. Điều đó đặt ra thách thức làm thế nào để sử dụng các công cụ Selenium một cách hiệu quả nhất và phát huy tối đa ưu điểm của chúng. Khóa luận này sẽ trình bày cách sử dụng các công cụ Selenium, ứng dụng vào bài toán thực tế và đề xuất phương pháp luận sử dụng hữu hiệu các công cụ này.

## 1.2. Mục tiêu của đề tài

Mục tiêu của đề tài là tìm hiểu cơ sở lý thuyết về kiểm thử, kiểm thử tự động và cách sử dụng các công cụ Selenium, so sánh ưu nhược điểm của từng công cụ, sử dụng các công cụ sinh test case cho ứng dụng thương mại điện tử phổ biến và đưa ra phương pháp luận sử dụng các công cụ đó.

## 1.3. Các vấn đề được giải quyết

Về lý thuyết, đề tài đã đưa ra phương pháp luận cách sử dụng các công cụ Selenium, tăng hiệu quả sử dụng các công cụ.

Hiểu rõ các ưu nhược điểm từng công cụ và áp dụng vào việc sinh test case cho website thương mại điện tử Tiki.vn

#### **1.4. Bố cục của khóa luận**

Bố cục của khóa luận được chia làm bốn chương với các nội dung chính như sau:

Chương 1 giới thiệu tổng quan về kiểm thử phần mềm và kiểm thử tự động. Chương này trình bày khái niệm kiểm thử phần mềm, mục đích và các kỹ thuật trong kiểm thử phần mềm. Ngoài ra còn đề cập đến một số công cụ kiểm thử tự động...

Chương 2 giới thiệu tổng quan về công cụ kiểm thử tự động Selenium, các thành phần trong Selenium

Chương 3 tìm hiểu chi tiết các công cụ trong Selenium, đưa ra ưu nhược điểm của mỗi công cụ.

Chương 4 Demo các công cụ với một trang web thương mại và rút ra phương pháp luận.

#### **1.5. Phương pháp nghiên cứu**

##### **1.5.1. Phương pháp nghiên cứu lý thuyết**

- Nghiên cứu lịch sử, cấu trúc, thành phần, cách sử dụng các công cụ của Selenium.
- Phân tích ưu nhược điểm của từng công cụ.
- Đánh giá so sánh các công cụ và đưa ra phương pháp luận.

##### **1.5.2. Phương pháp nghiên cứu thực nghiệm**

- Áp dụng các công cụ kiểm thử tự động của Selenium cho việc sinh test case trang web tiki.vn cho các chức năng đăng nhập, tìm kiếm, đặt hàng, xóa hàng.

## **CHƯƠNG 2. TỔNG QUAN VỀ CÔNG CỤ KIỂM THỬ TỰ ĐỘNG SILENIUM**

Chương này trình bày về tổng quan về các kiến thức liên quan đến kiểm thử phần mềm, kiểm thử tự động và giới thiệu tổng quan về công cụ kiểm thử tự động Selenium.

### **2.1. Kiến thức liên quan**

#### **2.1.1. Kiểm thử phần mềm**

##### **2.1.1.1. Định nghĩa về kiểm thử phần mềm**

Kiểm thử phần mềm là quá trình thẩm định phần mềm để xác định nó có được xây dựng đúng theo tài liệu yêu cầu và đáp ứng được nhu cầu của người sử dụng hay chưa.

##### **2.1.1.2. Mục đích của kiểm thử phần mềm**

Kiểm thử không chỉ giới hạn ở việc thực hiện một chương trình hoặc ứng dụng với mục đích đi tìm các lỗi phần mềm (bao gồm các lỗi và các thiếu sót) mà còn là một quá trình phê chuẩn và xác minh một chương trình máy tính/ứng dụng/sản phẩm nhằm:

- Tìm ra được càng nhiều lỗi càng tốt trong điều kiện về thời gian đã định và nguồn lực sẵn có.
- Chứng minh rằng sản phẩm phần mềm phù hợp với các đặc tả của nó.
- Xác thực chất lượng kiểm thử phần mềm đã dùng chi phí và nỗ lực tối thiểu.
- Thiết kế tài liệu kiểm thử một cách có hệ thống và thực hiện nó sao cho có hiệu quả, tiết kiệm được thời gian, công sức.

##### **2.1.1.3. Các kỹ thuật kiểm thử phần mềm**

Có 3 kỹ thuật kiểm thử phần mềm chính là: kiểm thử hộp đen, kiểm thử hộp trắng và kiểm thử hộp xám.

- Kiểm thử hộp đen:

Kỹ thuật kiểm thử hộp đen xem chương trình như là một “hộp đen”, trong đó người kiểm thử không quan tâm đến cấu trúc bên trong của chương trình mà chỉ quan tâm tới dữ liệu đầu vào và đầu ra sau khi được xử lý.

- Kiểm thử hộp trắng:

Kỹ thuật kiểm thử hộp trắng hay còn gọi là “kiểm thử cấu trúc” là kỹ thuật kiểm thử cho phép khảo sát kiến trúc bên trong của chương trình. Kiểm thử hộp trắng là chiến lược được thực hiện trên ba trong sáu loại kiểm thử cơ bản trong các giai đoạn kiểm thử phần mềm là: kiểm thử đơn vị, kiểm thử tích hợp và kiểm thử hồi quy. Mục tiêu của kiểm thử hộp trắng là kiểm thử bao phủ nhiều nhất các câu lệnh, điểm quyết định và các rẽ nhánh trong mã nguồn nếu có thể.

- Kiểm thử hộp xám:

Kiểm thử hộp xám là kỹ thuật kiểm thử có sự kết hợp giữa kiểm thử hộp đen và kiểm thử hộp trắng. Trong đó ta cũng quan tâm đến dữ liệu đầu vào và đầu ra giống như trong kiểm thử hộp đen, song lại đòi hỏi có sự truy cập đến cấu trúc dữ liệu và giải thuật để thiết kế các trường hợp kiểm thử.

#### **2.1.1.4. Các mức kiểm thử:**

Các mức của việc kiểm thử phản ánh mức độ trừu tượng được thấy trong mô hình thác nước của vòng đời của việc phát triển phần mềm.

- Kiểm thử đơn vị (Unit Test): nhóm phát triển có trách nhiệm thực hiện trước khi bàn giao chương trình cho nhóm kiểm thử.
- Kiểm thử chức năng (Functional Test), kiểm thử phi chức năng: nhóm kiểm thử có trách nhiệm thực hiện.
- Kiểm thử tích hợp (Integration Test): áp dụng thử nghiệm với các dự án CMMI. Điều kiện thực hiện kiểm thử tích hợp là phải có báo cáo kiểm thử chức năng với kết luận đạt tiêu chuẩn để tích hợp.
- Kiểu kiểm thử nghiệm thu (Acceptance Test): sẽ được thực hiện bởi phía khách hàng.

#### **2.1.1.5. Quy trình kiểm thử**

Quy trình kiểm thử phần mềm tổng quát:

- Bước 1. Lập kế hoạch kiểm thử (Test Planning)
- Bước 2. Phân tích và thiết kế các ca kiểm thử (Test analysis and Design).
- Bước 3. Thực thi kiểm thử (Test Executing).
- Bước 4. Báo cáo kiểm thử, đánh giá (Test Report and Evaluation).

#### **2.1.2. Kiểm thử tự động**

##### **2.1.2.1. Tổng quan về kiểm thử tự động**

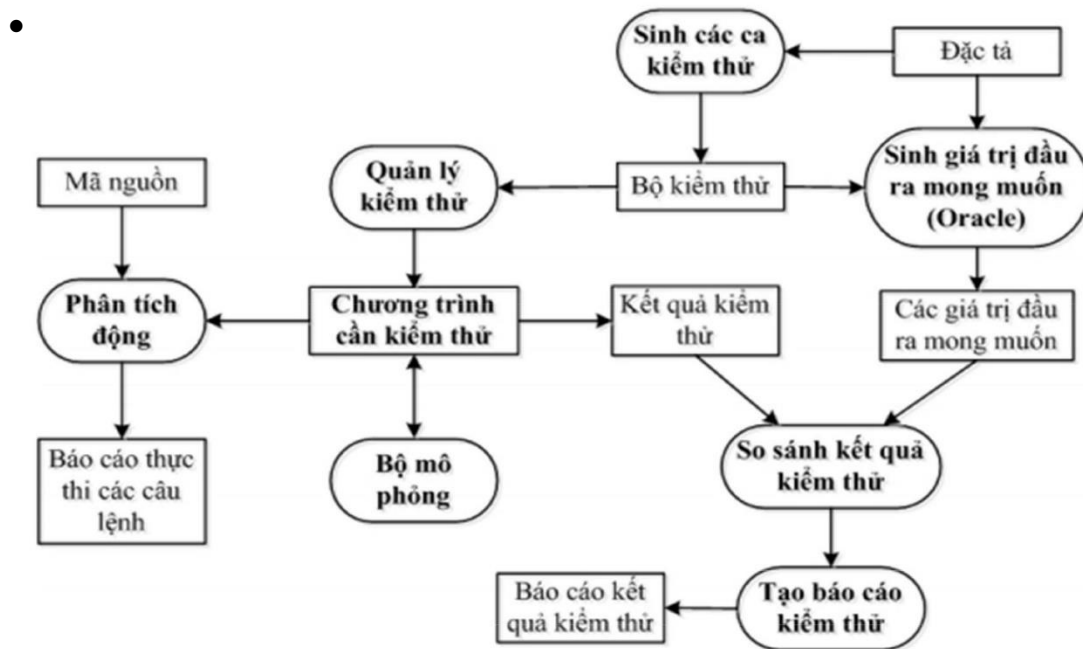
Kiểm thử tự động là quá trình thực hiện một cách tự động các bước trong một kịch bản kiểm thử. Kiểm thử tự động bằng một công cụ nhằm rút ngắn thời gian kiểm thử. Mục đích của kiểm thử tự động là giảm thiểu thời gian, công sức và kinh phí, tăng độ tin cậy, tăng tính hiệu quả và giảm sự nhầm lẫn cho người kiểm thử trong quá trình kiểm thử sản phẩm phần mềm.

Quy trình kiểm thử tự động gồm 4 bước:

- Bước 1: Viết kịch bản kiểm thử, dùng công cụ kiểm thử để ghi lại các thao tác lên phần mềm cần kiểm tra và tự động sinh ra một nhóm mã lệnh để đặc tả kịch bản (testscript).
- Bước 2. Chỉnh sửa để kịch bản kiểm thử thực hiện kiểm tra theo đúng yêu cầu đặt ra, làm theo trường hợp kiểm thử cần thực hiện.
- Bước 3. Chạy kịch bản kiểm thử, giám sát hoạt động kiểm tra phần mềm của kịch bản kiểm thử.
- Bước 4. Kiểm tra kết quả thông báo sau khi thực hiện kiểm thử tự động. Sau đó bổ sung, chỉnh sửa những sai sót.

Trong thực tế, có rất nhiều bộ công cụ hỗ trợ kiểm thử tự động được phát triển nhằm góp phần giải quyết các vấn đề khó khăn của quy trình kiểm thử. Hình 2 mô tả kiến trúc chung nhất của một bộ kiểm thử tự động. Trong kiến trúc này, các công

cụ kiểm thử được tích hợp trong một quy trình thống nhất nhằm hỗ trợ đầy đủ các hoạt động kiểm thử trong quy trình kiểm thử các sản phẩm của phần mềm.



Hình 1 Kiến trúc chung của một bộ kiểm thử tự động.

### 2.1.2.2. Ưu nhược điểm của kiểm thử tự động

- Ưu điểm:
  - Độ tin cậy cao (Reliability): Công cụ kiểm thử tự động có sự ổn định cao hơn so với con người, đặc biệt trong trường hợp có nhiều Test case, nên độ tin cậy cao hơn so với kiểm thử thủ công.
  - Khả năng lặp (Repeatability): Công cụ kiểm thử tự động ra đời là để giúp cho các kiểm thử viên không phải lặp đi lặp lại các thao tác (ví dụ: nhập dữ liệu, click, check kết quả,...) một cách nhàm chán với độ tin cậy và ổn định cao. Kiểm thử tự động rất hữu ích trong kiểm thử hồi quy.
  - Khả năng tái sử dụng (Reusability): Với một bộ kiểm thử tự động, người ta có thể sử dụng cho nhiều phiên bản ứng dụng khác nhau, đây chính là tính tái sử dụng.
  - Tốc độ cao (High speed): Do thực thi bởi công cụ nên tốc độ của kiểm thử tự động nhanh hơn nhiều so với tốc độ của con người.



- Chi phí thấp (Cost Reduction): Nếu áp dụng kiểm thử tự động đúng cách, người ta có thể tiết kiệm được nhiều chi phí, thời gian và nhân lực, do kiểm thử tự động nhanh hơn nhiều so với kiểm thử thủ công, đồng thời nhân lực cần để thực thi và bảo trì không nhiều.
- Nhược điểm:
  - Khó mở rộng, khó bảo trì (Poor scalability and maintainability): Trong cùng một dự án, để mở rộng phạm vi cho kiểm thử tự động khó hơn nhiều so với kiểm thử thủ công vì cập nhật hay chỉnh sửa yêu cầu nhiều công việc như debug, thay đổi dữ liệu đầu vào và cập nhật code mới.
  - Khả năng bao phủ thấp (Low coverage): Do khó mở rộng và đòi hỏi nhiều kỹ năng lập trình nên độ bao phủ của kiểm thử tự động thấp xét trên góc nhìn toàn dự án.
  - Vấn đề công cụ và nhân lực (Technology and people issues): Hiện nay cũng có nhiều công cụ hỗ trợ kiểm thử tự động khá tốt nhưng chúng vẫn còn nhiều hạn chế. Ngoài ra nhân lực đạt yêu cầu (có thể sử dụng thành thạo các công cụ này) cũng không nhiều.
  - Không thể thay thế hoàn toàn kiểm thử thủ công: Để thực hiện kiểm thử tự động, trước hết vẫn cần bàn tay con người thiết lập thao tác cho công cụ hay các đoạn kịch bản máy tính để thực thi. Đối với những ca kiểm thử chỉ thực hiện số ít lần thì việc mất thời gian tạo kịch bản kiểm thử tự động là không cần thiết. Chưa kể tới những ca kiểm thử với đặc thù riêng biệt mà kiểm thử tự động không làm được.

### **2.1.2.3. Một số công cụ kiểm thử tự động**

- Quick Test Professional (QTP)

Quick Test Professional là phần mềm kiểm soát việc kiểm thử tự động các chức năng của các sản phẩm phần mềm cần kiểm thử. Sản phẩm này bao gồm một tập các mô-đun có thể tương tác với nhau nhằm quản lý toàn bộ quy trình kiểm thử phần mềm. Quick Test Professional là một công cụ hỗ trợ kiểm thử hàm (kiểm thử chức năng) và cho phép tiến hành kiểm thử hồi quy một cách tự động.

- NUnit

NUnit là một testing framework mã nguồn mở được phát triển bởi một nhóm lập trình viên (Charlie Poole, Rob Prouse and Simone Busoli), tương tác trực tiếp với Visual Studio, đồng thời có thể chạy độc lập mà không phụ thuộc vào Visual Studio.

- JUnit

JUnit là một framework đơn giản dùng cho việc tạo các unit testing tự động, và chạy các test có thể lặp đi lặp lại. Nó chỉ là một phần của kiến trúc xUnit cho việc tạo các unit testing. JUnit là một chuẩn trên thực tế cho unit testing trong Java. JUnit tránh cho người lập trình phải làm đi làm lại những việc kiểm thử nhàm chán bằng cách tách biệt mã kiểm thử ra khỏi mã chương trình, đồng thời tự động hóa việc tổ chức và thi hành các bộ số kiểm thử.

- Load Runner

Load Runner giả lập một môi trường ảo gồm nhiều người dùng thực hiện các giao dịch cùng một lúc nhằm giám sát các thông số xử lý của phần mềm cần kiểm thử. Kết quả thống kê sẽ được lưu lại và cho phép kiểm thử viên thực hiện phân tích nhằm kiểm thử khả năng chịu tải và các yêu cầu phi chức năng khác của sản phẩm. Trong quá trình kiểm thử, Load Runner tự động tạo ra các kịch bản kiểm thử để lưu lại các thao tác người dùng tương tác lên phần mềm. Mỗi kịch bản này còn được xem là hoạt động của một người dùng ảo mà Load Runner giả lập. Ngoài ra, công cụ này còn cho phép tổ chức, điều chỉnh, quản lý và giám sát hoạt động kiểm tra khả năng chịu tải.

- Jmeter

Jmeter là một mã nguồn mở được viết bằng java. Công cụ để đo độ tải và performance của đối tượng, có thể sử dụng để kiểm thử hiệu năng trên cả nguồn tĩnh và nguồn động, có thể kiểm tra độ tải và hiệu năng trên nhiều loại server khác nhau như: Web – HTTP, HTTPS, SOAP, Database thông qua JDBC, LDAP, JMS, Mail – SMTP(S), POP3(S) and IMAP(S)...

Cách thức hoạt động của Jmeter: Giả lập một nhóm người dùng gửi các yêu cầu tới một máy chủ, nhận và xử lý các response từ máy chủ và trình diễn các kết quả đó cho người dùng dưới dạng bảng biểu, đồ thị, cây...

## **2.2. Công cụ kiểm thử Selenium**

### **2.2.1. Tổng quan**

Selenium là một công cụ kiểm thử phần mềm tự động, được phát triển bởi ThoughtWorks từ năm 2004 với tên ban đầu là JavaScriptTestRunner. Đến năm 2007, tác giả Jason Huggins rời ThoughtWorks và gia nhập Selenium team, một phần của Google và phát triển thành Selenium như hiện nay.

Selenium là một tập hợp mạnh mẽ của các công cụ hỗ trợ phát triển nhanh chóng của các thử nghiệm tự động hóa cho các ứng dụng dựa trên web. Selenium cung cấp một tập phong phú của các thử nghiệm chức năng đặc biệt hướng đến các nhu cầu của các thử nghiệm của một ứng dụng web. Các hoạt động này là rất linh hoạt, cho phép nhiều tùy chọn cho vị trí các thành phần UI và so sánh kết quả thử nghiệm dự kiến sẽ chống lại hành vi ứng dụng thực tế.

Selenium là một mã nguồn mở và là một công cụ kiểm thử phần mềm tự động hóa để thử nghiệm các ứng dụng web. Nó có khả năng hoạt động trên nhiều các trình duyệt và hệ điều hành khác nhau. Selenium không chỉ là một công cụ duy nhất mà là một bộ các công cụ giúp những người kiểm thử tự động hóa các ứng dụng dựa trên web hiệu quả hơn.

### **2.2.2. Một số đặc điểm của Selenium**

- Selenium là một công cụ mã nguồn mở: Vì là mã nguồn mở nên chúng ta có thể sử dụng mà không phải lo lắng về phí bản quyền hay thời hạn sử dụng.
- Các thử nghiệm sau đó có thể được chạy trên các trình duyệt web hiện đại nhất.
- Selenium hỗ trợ chạy trên nhiều OS khác nhau, Selenium triển khai trên nền tảng Windows, Linux và Mac.
- Nó cho phép recording, editing and debugging tests.

- Kiểm thử có thể được exported ở hầu hết các ngôn ngữ ví dụ: HTML, Java, .Net, perl, ruby, ...
- Selenium có sự hỗ trợ của một số nhà cung cấp trình duyệt lớn hỗ trợ.
- Selenium hỗ trợ nhiều ngôn ngữ lập trình C#, Java, Python, PHP, Selenium còn có thể kết hợp với một số công cụ kiểm thử khác như Junit, Bromien, Nunit..

### 2.2.3. Các thành phần của Selenium

Selenium không phải là một tool hay một tiện ích đơn thuần, hơn thế nó là một package - gói - với một vài tool test, do đó nó giống một bộ hơn. Mỗi tool được thiết kế nhằm phục vụ các mục đích kiểm thử khác nhau và các yêu cầu về môi trường kiểm thử. Gói phần mềm bao gồm các tool sau:

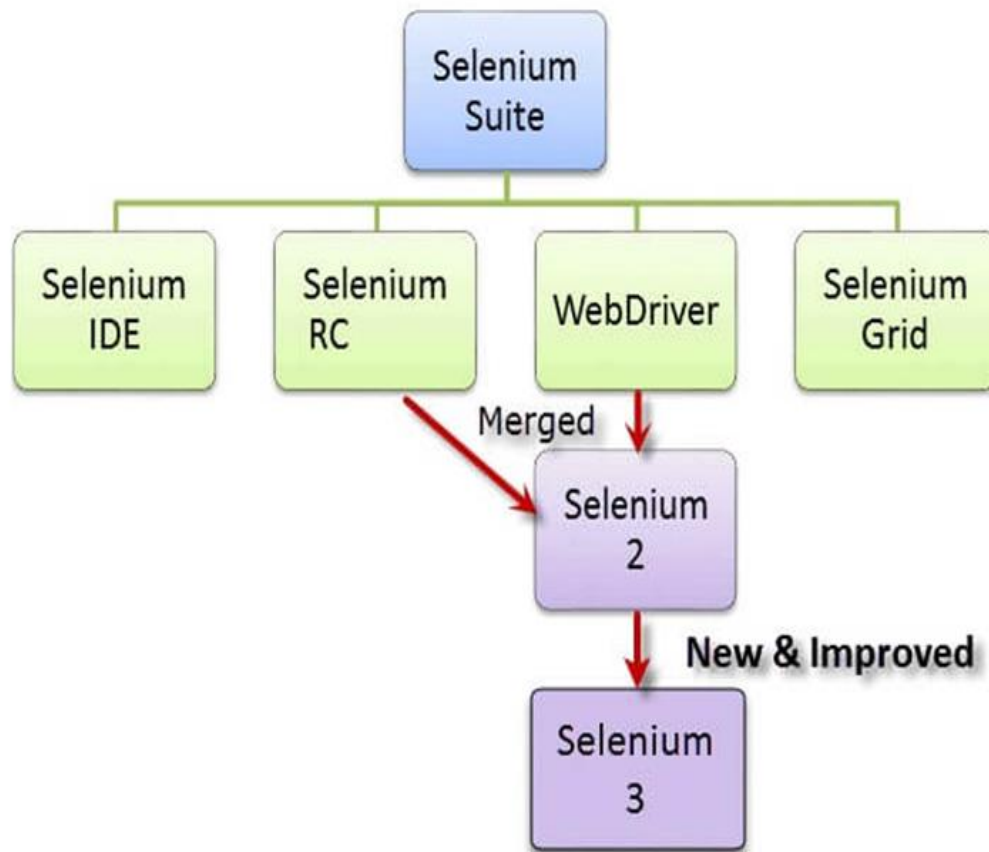
- Selenium Integrated Development Environment (IDE).
- Selenium Remote Control (RC).
- Selenium WebDriver.
- Selenium Grid.

Selenium IDE là một công cụ cho phép ghi lại một kịch bản và tái sử dụng kịch bản đó. Nó hoạt động như một Add-on của trình duyệt với giao diện trực quan, dễ sử dụng ngay cả với những kiểm thử viên không biết về code.

Selenium RC cho phép các nhà phát triển tự động hóa quá trình kiểm thử bằng cách sử dụng bất kỳ ngôn ngữ lập trình nào, phát huy tối đa thế mạnh của Selenium trong kiểm thử đơn vị. Để dễ dàng hơn cho việc kiểm thử, Selenium RC cung cấp các API và thư viện cho mỗi ngôn ngữ được hỗ trợ: HTML, Java, Perl, PHP, Ruby, Python, C#.

Selenium WebDriver là phiên bản kế nhiệm của Selenium RC. Cũng giống như Selenium RC, Selenium WebDriver hỗ trợ viết kịch bản kiểm thử bằng các ngôn ngữ khác nhau như Java, .NET, PHP, Python, Perl, Ruby và kiểm thử viên có thể sử dụng các điều kiện if, else hay các vòng lặp để tăng tính chính xác cho kịch bản kiểm thử. Selenium WebDriver có kiến trúc khá đơn giản, điều khiển trình duyệt trực tiếp từ hệ điều hành.

Selenium Grid là một hệ thống hỗ trợ kiểm thử viên thực thi kịch bản kiểm thử trên nhiều máy, nhiều trình duyệt một cách song song mà không cần chỉnh sửa kịch bản kiểm thử. Ban đầu, Selenium Grid chỉ hỗ trợ cho Selenium RC nhưng sau này đã xuất hiện trên cả Selenium WebDriver. Selenium Grid cho phép kiểm thử viên thực thi ca kiểm thử trên nhiều máy khác nhau với nhiều trình duyệt khác nhau. Đặc biệt hơn, Selenium Grid còn cung cấp khả năng kiểm thử với chế độ phân tán.



Hình 2: Các công cụ kiểm thử

Hiện tại, Selenium RC và WebDriver được hợp nhất thành một framework duy nhất để tạo ra Selenium 2. Còn Selenium 1 thì tham chiếu đến Selenium RC.

#### 2.2.4. So sánh giữa Selenium và QTP (Quick Test Professional)

QTP là một công cụ hỗ trợ kiểm thử hàm (kiểm thử chức năng) và cho phép tiến hành kiểm thử hồi quy một cách tự động. Hiện nay, QTP được sử dụng khá phổ

biến. So sánh với QTP, Selenium có nhiều ưu điểm vượt trội bên cạnh những điểm hạn chế.

Ưu điểm của Selenium so với QTP được thể hiện trong bảng dưới đây:

<b>Selenium</b>	<b>QTP</b>
Mã nguồn mở, miễn phí sử dụng.	Mất phí sử dụng.
Có thể chạy kiểm thử trên nhiều trình duyệt khác nhau.	Chỉ có thể chạy kiểm thử trên Firefox, Internet Explorer và Chrome.
Hỗ trợ các hệ điều hành khác nhau.	Chỉ có thể sử dụng trên Windows.
Hỗ trợ các thiết bị Mobile.	QTP hỗ trợ tự động hóa thử nghiệm ứng dụng trên điện thoại di động (iOS và Android) bằng giải pháp HP – HP Mobile Center.
Có thể thực hiện kiểm thử khi trình duyệt đang bị thu nhỏ.	Cần có ứng dụng bên dưới để có thể hiển thị trên desktop.
Có thể thực hiện nhiều ca kiểm thử song song.	Chỉ có thể thực hiện các ca kiểm thử song song khi sử dụng sản phẩm mất phí Quality Center.

*Bảng 1: So sánh Selenium với QTP*

Mặt khác, QTP cũng có những ưu điểm so với Selenium được thể hiện trong bảng sau:

<b>QTP</b>	<b>Selenium</b>
Có thể kiểm thử trên cả ứng dụng web và desktop.	Chỉ có thể kiểm thử trên ứng dụng web
Đi kèm với một kho lưu trữ đối tượng được xây dựng sẵn.	Không có kho lưu trữ đối tượng được xây dựng sẵn.

Tự động nhanh hơn Selenium vì nó là một IDE đầy đủ tính năng.	Tự động với tốc độ chậm hơn vì không có IDE gốc và chỉ IDE của bên thứ ba có thể được sử dụng và phát triển.
Có thể truy cập và điều khiển bên trong trình duyệt.	Không thể truy cập phần tử bên ngoài ứng dụng web đang được kiểm thử.
Cung cấp hỗ trợ khách hàng chuyên nghiệp.	Không có hỗ trợ người dùng chính thức nào đang được cung cấp.
Có khả năng tự xuất dữ liệu kiểm tra các định dạng bên ngoài.	Không có khả năng tự xuất dữ liệu thời gian chạy vào định dạng bên ngoài.
Hỗ trợ Thông số được xây dựng.	Thông số có thể được thực hiện thông qua lập trình nhưng rất khó thực hiện.
Test reports được tạo tự động.	Không hỗ trợ báo cáo test/bug.

*Bảng 2: Ưu điểm QTP so với Selenium*

Từ hai bảng so sánh trên, có thể thấy QTP có nhiều điểm tiên tiến hơn nhưng Selenium lại có lợi thế vượt trội trong ba lĩnh vực chính:

- Chi phí (vì Selenium hoàn toàn miễn phí)
- Tính linh hoạt (vì một số ngôn ngữ lập trình, trình duyệt và nền tảng mà nó có thể hỗ trợ)

Kiểm tra song song (điều mà QTP có thể nhưng chỉ với việc sử dụng Quality Center

### **CHƯƠNG 3. CÁC CÔNG CỤ SELENIUM VÀ ỨNG DỤNG**

Chương này trình bày các công cụ selenium như Selenium Remote Control, Selenium IDE, Selenium WebDriver, Selenium Grid. Phân tích ưu, nhược điểm, các thành phần cơ bản của các công cụ đó.

### **3.1. SELENIUM REMOTE CONTROL (RC)**

#### **3.1.1. Tổng quan**

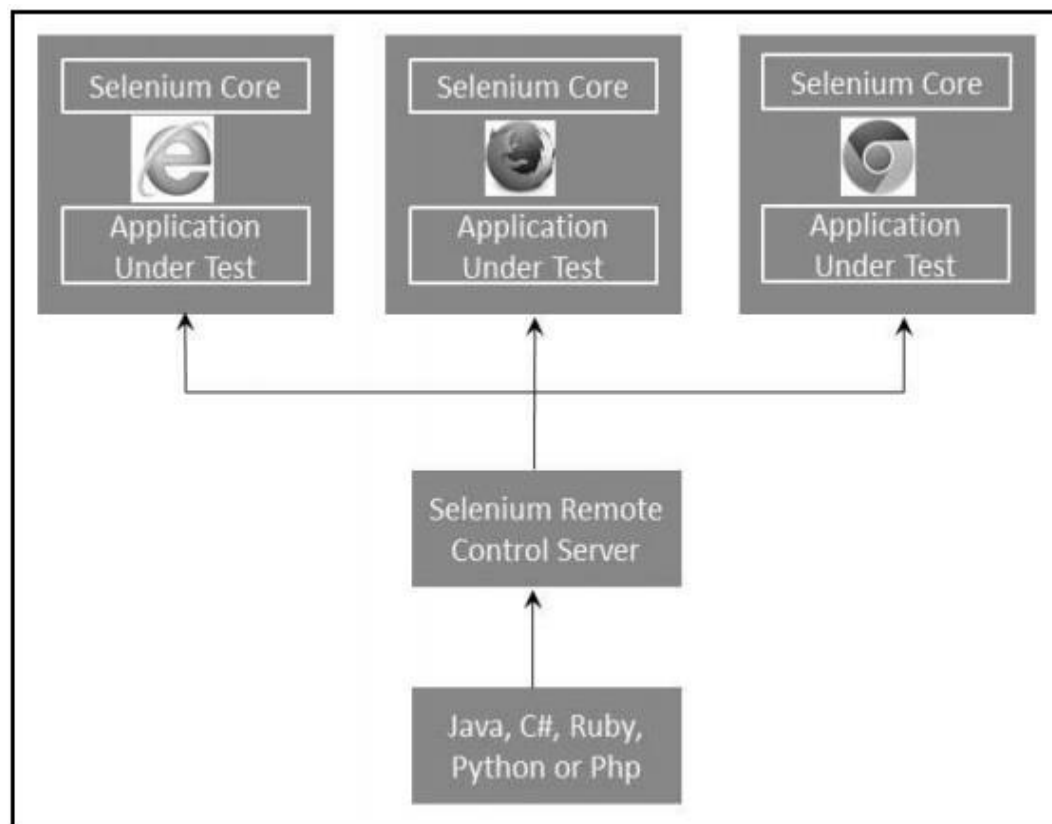
Selenium RC là dự án Selenium chính trong một thời gian dài trước khi Selenium WebDriver (Selenium 2.0) ra đời. Giờ đây Selenium RC hầu như không được sử dụng vì WebDriver cung cấp nhiều tính năng mạnh mẽ hơn. Tuy nhiên bạn vẫn có thể tiếp tục phát triển các script sử dụng RC.

Selenium RC cho phép các nhà phát triển tự động hóa kiểm tra sử dụng một ngôn ngữ lập trình cho tính linh hoạt tối đa và mở rộng trong việc phát triển logic thử nghiệm. Ví dụ, nếu trình ứng dụng trả về một tập kết quả của việc kiểm tra, và nếu chương trình thử nghiệm tự động cần chạy thử nghiệm trên mỗi phần tử trong tập hợp kết quả, hỗ trợ lặp đi lặp lại các ngôn ngữ lập trình có thể được sử dụng để chuyển đổi thông qua việc tập hợp kết quả, kêu gọi Selenium lệnh chạy thử nghiệm trên mỗi mục.

Khả năng sử dụng Selen-RC với một ngôn ngữ lập trình bậc cao để phát triển các trường hợp thử nghiệm cũng cho phép thử nghiệm tự động được tích hợp với một dự án xây dựng môi trường tự động.

Selenium RC cho phép chúng ta viết các kiểm thử giao diện của ứng dụng Web tự động với sự giúp đỡ của các ngôn ngữ lập trình như Java, C#, Perl, Python, PHP để tạo ra các ca kiểm thử phức tạp hơn như đọc và viết các tập tin, truy vấn cơ sở dữ liệu và gửi mail kết quả kiểm thử. Khả năng sử dụng Selen-RC với một ngôn ngữ lập trình bậc cao để phát triển các trường hợp thử nghiệm cũng cho phép thử nghiệm tự động được tích hợp với một dự án xây dựng môi trường tự động.





Hình 3: Kiến trúc Selenium RC

### 3.1.2. Ưu nhược điểm của RC

- Ưu điểm:
  - Hỗ trợ ngôn ngữ lập trình và cấu trúc.
  - Hỗ trợ đa trình duyệt và đa nền tảng (điều này tối ưu hơn rất nhiều so với Selenium IDE).
  - Hỗ trợ tạo các tiện ích người dùng như Generics/ Exceptions để tùy chỉnh Framework.
  - Hỗ trợ xử lý lỗi và kiểm tra Database.
  - Hỗ trợ test Data driven testing.
  - Hỗ trợ ghi log và chụp màn hình.
  - Hỗ trợ Framework testing giống như TestNG và Junit.
- Nhược điểm:
  - Kịch bản test không tương tác trực tiếp với trình duyệt, server Selenium RC cần được chạy để có thể tương tác.
  - Người sử dụng cần phải có kỹ năng lập trình (đây là nhược điểm lớn so với Selenium IDE).

- Không xử lý cảnh báo và điều hướng hiệu quả.
- Không hỗ trợ test các ứng dụng có nền tảng WAP (iphone/Android).
- Không thể xử lý tốt gọi Ajax.

## **3.2. SELENIUM IDE**

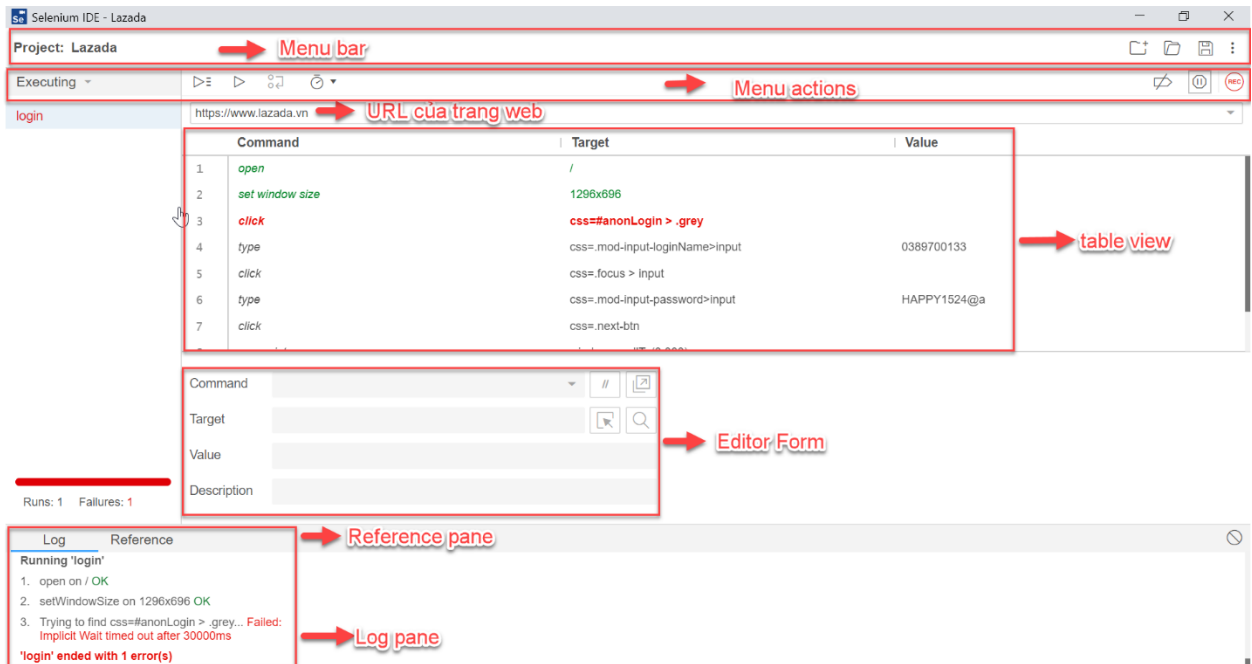
### **3.2.1. Tổng quan**

Selenium IDE là môi trường phát triển tích hợp cho việc xây dựng trường hợp thử nghiệm Selenium. Nó hoạt động như một trình duyệt Firefox add-on hoặc Chrome add-on và cung cấp một giao diện để sử dụng để phát triển và chạy trường hợp kiểm thử cá nhân, bộ kiểm tra toàn bộ. Selenium IDE có một tính năng ghi lại, sẽ giữ tài khoản của người sử dụng khi chúng được thực hiện và lưu trữ chúng như là một kịch bản tái sử dụng để phát sử dụng. Nó cũng có một menu ngữ cảnh (nhấn chuột phải) tích hợp với trình duyệt Firefox hoặc Chrome, cho phép người dùng chọn từ một danh sách xác nhận và xác minh cho các vị trí đã chọn. Selenium IDE cũng cung cấp chỉnh sửa đầy đủ các trường hợp thử nghiệm cho chính xác hơn và kiểm soát. Mặc dù Selenium IDE chỉ là một add-on, các bộ kiểm thử tạo ra cũng có thể được chạy cho các trình duyệt khác bằng cách sử dụng Selenium RC và chỉ định tên của bộ ứng dụng thử nghiệm trên dòng lệnh.

### **3.2.2. Ưu nhược điểm của Selenium IDE**

- Ưu điểm:
  - Dễ dàng ghi và chạy lại các thao tác.
  - Không yêu cầu người sử dụng có kiến thức lập trình.
  - Có khả năng ghi log bằng việc sử dụng file logging plug-in trên Firefox.
  - Linh hoạt và có khả năng mở rộng.
- Nhược điểm:
  - Không thể tích hợp trên tất cả các trình duyệt.
  - Không hỗ trợ lặp lại và các câu lệnh có điều kiện.
  - Không hỗ trợ sửa lỗi.
  - Không hỗ trợ test các kịch bản độc lập hoặc theo nhóm.
  - Không hỗ trợ test dữ liệu trong database.

### 3.2.3. Các thành phần cơ bản của Selenium IDE



Hình 4: Các thành phần cơ bản của Selenium IDE

Chi tiết từng tính năng gồm các phần như sau:

- Menu bar: được đặt trên đầu cửa sổ Selenium IDE và thường gồm:
  - Tên project: tên của project hiện tại
  - Create new project: tạo một project mới
  - Open project: mở một project có sẵn
  - Save project: lưu project hiện tại
  - Help menu: liệt kê một loạt các tài liệu chính thức và các ghi chú phát hành để hỗ trợ người dùng.
- Menu actions:
  - Run all tests: Tùy chọn để chạy lại tất cả các testcase Selenium IDE có liên quan đến bộ kiểm thử hiện tại.
  - Run current tests: Tùy chọn chạy lại testcase Selenium IDE hiện tại đã được ghi lại/ tạo ra bởi người dùng.
  - Stop test execution: Tùy chọn kết thúc testcase đang chạy.
  - Pause test execution: Tùy chọn tạm dừng testcase đang chạy.
  - Test execution speed: Tùy chọn điều chỉnh tốc độ chạy testcase.
  - Disable Breakpoint : Tùy chọn bỏ qua tất cả các Breakpoint.
  - Pause on exceptions: Tạm dừng chạy testcase khi javascript bị lỗi.

- Start/ Stop recording: Lựa chọn cho phép người dùng bắt đầu hoặc dừng thao tác ghi hành động của người dùng. Nếu biểu tượng có màu hồng nhưng rỗng màu ở giữa: bắt đầu ghi, biểu tượng được phủ kín màu đỏ: dừng ghi. Selenium IDE mở ra luôn ở chế độ mặc định là record.
- Address bar: chứa URL của trang web đang test.
- Table view: Đây là chế độ view mặc định của Selenium IDE. Test case được hiển thị dạng bảng. Mỗi hành động của người dùng trong bảng là 1 sự kết hợp của "Command" - lệnh; "Target" - đích; "Value" - giá trị, trong đó lệnh - hành động người dùng, mục tiêu - các phần tử web được xác định duy nhất và giá trị - dữ liệu test tương ứng. Bên cạnh việc ghi lại thì nó cũng cho phép người dùng thêm, tạo hoặc chỉnh sửa các câu lệnh với sự hỗ trợ của form editor hiện tại phía dưới.
- Editor Form: cho phép người dùng chọn bất kỳ lệnh nào và các lệnh gợi ý có liên quan có thể tự động điền vào. Nút Select cho phép người dùng chọn bất cứ phần tử web và vị trí của nó có thể tự động nạp vào trường mục tiêu. Button Find cho phép người dùng tìm kiếm các phần tử web trên trang web với mục tiêu đã được xác định. Value là dữ liệu kiểm thử được nhập vào với mục tiêu là để chúng ta kiểm thử kịch bản.
- Log Pane: vùng ghi log, cung cấp thông tin chi tiết về việc thực thi hiện tại theo dạng thông báo với cấp độ ghi log theo thời gian thực. Do đó, message log cho phép user có thể debug - gỡ lỗi - các vấn đề trong trường hợp thực thi test case thất bại.
- Reference Pane: đưa ra các mô tả ngắn gọn về các lệnh Selenses được chọn hiện tại với các đối số chi tiết của nó.

*Một số câu lệnh hay dùng trong Selenium IDE*

Tên lệnh	Ý nghĩa
<b>open</b>	<b>Đi đến một trang Web theo URL xác định.</b>
<b>click</b>	<b>Hoàn thành hành động click chuột.</b>
<b>click and wait</b>	<b>Hoàn thành hành động click chuột và đợi tải một trang Web mới.</b>
<b>verify title/ assert title</b>	<b>Xác minh tiêu đề trang mong đợi.</b>

<b>verify text present</b>	<b>Xác minh giá trị một đoạn văn bản ở vị trí nào đó trên trang.</b>
<b>verify element present</b>	<b>Xác minh thành phần giao diện người dùng được mong đợi, được định nghĩa bởi thẻ HTML là tồn tại trên trang.</b>
<b>verify text</b>	<b>Xác minh văn bản mong đợi và các thẻ HTML tương ứng trên trang.</b>
<b>verify table</b>	<b>Xác minh nội dung mong đợi của một bảng.</b>
<b>waitForPage to load</b>	<b>Tạm dừng thực thi ca kiểm thử cho tới khi việc tải trang Web mới được hoàn tất. Lệnh này được tự động gọi khi sử dụng lệnh clickAndWait.</b>
<b>wait for element present</b>	<b>Tạm dừng thực thi ca kiểm thử cho tới khi các yếu tố giao diện người dùng mong đợi trên trang Web xuất hiện.</b>

### **3.3. SELENIUM WEBDRIVER**

#### **3.3.1. Tổng quan**

Selenium WebDriver là một công cụ để kiểm thử tự động các ứng dụng web. Nó thường được gọi là Selenium 2.0. WebDriver sử dụng một framework cơ bản khác biệt trong khi Selenium RC sử dụng Javascript Selenium-Core nhúng vào trong trình duyệt. WebDriver tương tác trực tiếp với các trình duyệt và không cần bất kỳ trung gian nào, không giống như Selenium RC phụ thuộc vào một máy chủ. WebDriver được sử dụng trong ngữ cảnh sau:

- Kiểm thử đa trình duyệt, bao gồm cải thiện chức năng cho trình duyệt mà không được hỗ trợ tốt bởi Selenium RC (Selenium 1.0).
- Điều khiển nhiều frame, nhiều cửa sổ trình duyệt, nhiều popup và alert. Điều hướng trang phức hợp.

- Điều hướng người dùng nâng cao như kéo-thả (drag-and-drop) AJAX-based UI elements.



*Hình 5: Webdriver tương tác trình duyệt*

### 3.3.2. Ưu nhược điểm của Selenium WebDriver

- Ưu điểm:
  - Selenium WebDriver không yêu cầu có server Selenium để chạy kịch bản
  - Tương tác trực tiếp với trình duyệt.
  - Giao diện hướng đối tượng.
  - Hỗ trợ công cụ tìm kiếm động.
  - Cung cấp tiện ích và class hỗ trợ trong việc xử lý cảnh báo, điều hướng, gọi Ajax và dropdowns.
  - Hỗ trợ test các ứng dụng WAP (iphone/android).
- Nhược điểm:
  - Yêu cầu API phức tạp.
  - Không hỗ trợ test mobile.
  - Việc migrate từ Selenium Remote Control sang WebDriver khá phức tạp.

- Không hỗ trợ kiểm thử trên ứng dụng có sử dụng đối tượng flash/flex.

### 3.3.3. Các thành cơ bản của Selenium WebDriver

- Cấu tạo của một trang web: Đơn vị cấu thành cơ bản của 1 trang web là phần tử web. Mỗi phần tử web có nhiều thành phần, trong đó thường bao gồm 4 thành phần chính:
  - Thẻ mở (Opening Tag): thành phần đánh dấu bắt đầu một phần tử Web và định nghĩa phần tử nào được sử dụng.
  - Các thuộc tính (Attributes): các thông tin về phần tử Web, tất cả các phần tử Web đều có 4 thuộc tính cơ bản sau:
    - id: Đặc tả định danh duy nhất của phần tử Web.
    - class: Đặc tả một hay nhiều lớp của phần tử Web.
    - style: Đặc tả CSS style cho phần tử Web.
    - title: Đặc tả thông tin mở rộng về phần tử Web.
  - Nội dung (Element Content): thành phần nằm giữa thẻ mở và thẻ đóng để hiển thị nội dung của phần tử web.
  - Thẻ đóng (Closing Tag): thành phần đánh dấu kết thúc một phần tử web, nằm ở vị trí cuối cùng của phần tử.

Ví dụ minh họa các thành phần cơ bản của phần tử web như sau:

Thẻ mở	Thuộc tính	Nội dung	Thẻ đóng
<p>		Đây là một đoạn văn	</p>
<a id="lnk" href="default.html">	id, href	Đây là một đường dẫn	</a>

- Một số câu lệnh cơ bản hay được sử dụng của Selenium WebDriver:
  - driver.getUrl(); // mở một trang web mới trong trình duyệt hiện tại.
  - driver.getTitle(); // lấy tiêu đề của trang web hiện tại.
  - driver.getCurrentUrl(); // lấy Url của trang hiện tại đã được tải lên trình duyệt.
  - driver.getPageSource(); // lấy source của trang được tải cuối cùng.
  - driver.close(); // đóng cửa sổ hiện tại của trình duyệt.

- `driver.quit();` // thoát khỏi trình duyệt và tắt cả các cửa sổ đã mở.
- `driver.navigate().refresh();` // làm mới trình duyệt.
- `driver.findElementById("");`
- `pageLoadTimeout();` // set thời gian load Timeout cho điều khiển javascript.
- Một số câu lệnh điều hướng trình duyệt sử dụng
  - `navigate().forward();`
  - `navigate().back();`
  - `implicitlyWait();` // để chờ các cảnh báo.
  - `fluentWait();` // để xác định tối đa mức thời gian để chờ một điều khiển thực thi.

### 3.4. SELENIUM GRID

#### 3.4.1. Tổng quan

Selenium Grid cho phép người dùng thực thi kiểm thử song song trên nhiều máy tính khác nhau với nhiều trình duyệt khác nhau.

Selenium Grid cho phép thực thi kiểm thử với chế độ phân tán, sử dụng chung một code base. Do đó, hard code không cần thiết phải có mặt trên tất cả các máy được sử dụng để thực thi kiểm thử. Selenium Grid có 2 phiên bản là Grid 1 và Grid 2 được mô tả trong bảng dưới đây:

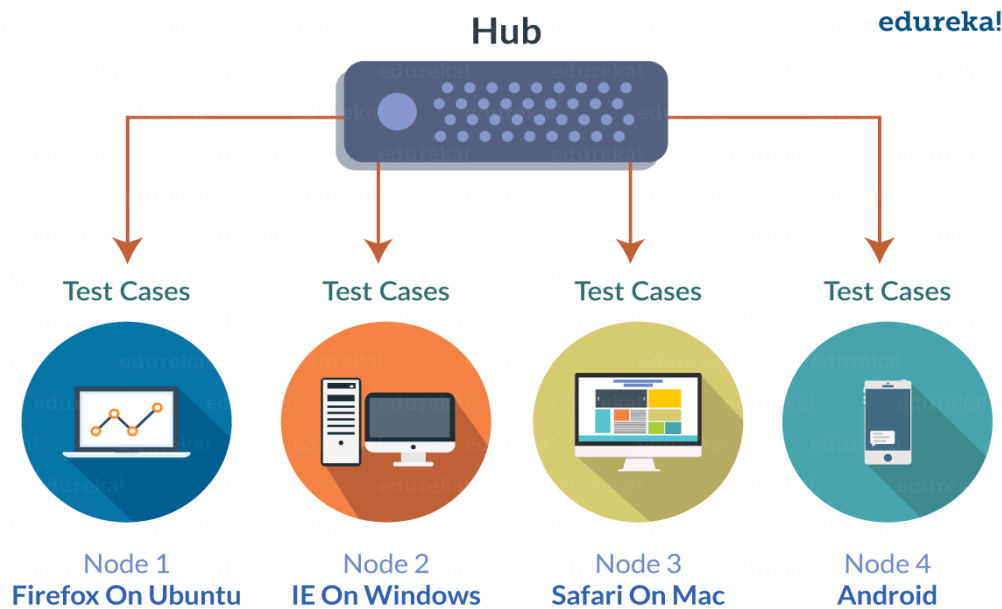
Selenium Grid 1	Selenium Grid 2
Có một bộ điều khiển riêng biệt với Selenium RC server.	Đi kèm luôn trong bộ Selenium server.
Chỉ hỗ trợ Selenium RC commands và script.	Hỗ trợ cả Selenium RC và Selenium WebDriver script.
Chỉ cho phép automate cho 1 trình duyệt trên 1 remote control.	Với mỗi remote control có thể automate lên tới 5 trình duyệt.
Cần phải cài đặt và cấu hình Apache Ant trước khi sử dụng.	Không yêu cầu cài đặt Apache Ant trước khi sử dụng.

*Bảng 3: So sánh hai phiên bản Selenium Grid*



Selenium Grid bao gồm 2 thành phần chính là Hub và Nodes:

- Hub: có thể hiểu là máy chủ server, chứa hard code và là nơi gửi lệnh điều khiển các máy khác trong mô hình thực thi kiểm thử. Hub chỉ có thể được set up duy nhất trên một máy tính.
- Nodes: là các Selenium instances được kết nối vào Hub để thực thi các kịch bản kiểm thử. Có thể có nhiều Nodes trong một mô hình Grid. Các Nodes có thể được set up trên nhiều máy tính với nhiều trình duyệt khác nhau.



Hình 6: Minh họa về hub và node của Selenium Grid

### 3.4.2. Ưu nhược điểm của Selenium Grid

- Ưu điểm:
  - Selenium Grid cho phép chạy nhiều ca kiểm thử trên các trình duyệt web, hệ điều hành và máy khác nhau. Điều này đảm bảo tính tương thích của ứng dụng được kiểm thử trên nhiều trình duyệt web, hệ điều hành và kiến trúc phần cứng.
  - Rút ngắn thời gian hoàn thành kiểm thử vì nó có khả năng chạy song song trên nhiều trình duyệt. Ví dụ: Với một ca kiểm thử thông thường

trên 5 trình duyệt khác nhau sẽ tốn thời gian gấp 5 lần nếu chúng ta không áp dụng kiểm thử bằng Selenium Grid.

- Nhược điểm:
  - Chi phí cho việc kiểm thử cao hơn so với các công cụ khác vì Selenium yêu cầu nhiều thiết bị và trình duyệt...
  - Yêu cầu người sử dụng phải có kiến thức lập trình.

## CHƯƠNG 4. DEMO VÀ PHƯƠNG PHÁP LUẬN SỬ DỤNG CÁC CÔNG CỤ SELENIUM

Chương này trình bày áp dụng các công cụ kiểm thử tự động của selenium sinh testcase cho trang web thương mại điện tử <https://tiki.vn>, đánh giá và đưa ra phương pháp luận sử dụng các công cụ Selenium.

### 4.1. Mô tả bài toán

#### 4.1.1. Giới thiệu bài toán

Kiểm thử tự động cho các ca kiểm thử cơ bản với trang web điện tử thương mại Tiki.vn

Các chức năng kiểm thử tự động bao gồm: đăng nhập, tìm kiếm, đặt hàng, xóa hàng.

#### 4.1.2. Giới thiệu chung về dự án

Trang web thương mại điện tử là kênh bán hàng hiệu quả cho các công ty, doanh nghiệp hiện nay, đặc biệt là các sàn thương mại điện tử, các trang thương mại điện tử tổng hợp. Các loại website này vốn đã phổ biến ở các nước phát triển từ lâu như Ebay, Amazon, Alibaba... Ở Việt Nam, trong những năm gần đây, Tiki.vn được biết đến là một trong những website thương mại điện tử hàng đầu hiện nay bên cạnh các thương hiệu khác như Lazada.vn, Adayroi.com,...

Trang web Tiki.vn cung cấp các sản phẩm gồm 10 ngành hàng: sách, điện thoại - máy tính bảng, thiết bị số - phụ kiện, điện gia dụng, nhà cửa - đời sống,... với các mặt hàng uy tín và mức giá hợp lý. Các chính sách giao hàng - vận chuyển, đổi trả, mua trả góp cũng được đánh giá là phù hợp với người dùng.

Khách hàng cần đăng ký một tài khoản để có thể đặt mua hàng trực tuyến và nhận các chính sách ưu đãi, hỗ trợ từ hệ thống.



Hình 7: Giao diện trang web Tiki.vn

## 4.1.2. Hướng giải quyết bài toán

### 4.1.2.1. Tổng quát

- Bước 1: Xây dựng các ca kiểm thử cơ bản cho các chức năng sẽ test.
- Bước 2: Viết mã lệnh cho các chức năng theo các ca kiểm thử đã tạo.
- Bước 3: Thực hiện chạy lệnh, ghi lỗi, gỡ lỗi và lập báo cáo kiểm thử.

### 4.1.2.2. Xây dựng ca kiểm thử cho các chức năng đăng nhập, tìm kiếm, đặt hàng, xóa hàng.

- Các ca kiểm thử cho chức năng đăng nhập

ID	Mục đích kiểm thử	Tiền điều kiện	Các bước thực hiện	Kết quả mong muốn	Kết quả kiểm thử
TC_001	Kiểm tra đăng nhập khi để trống trường Email	<ul style="list-style-type: none"> <li>- Màn hình Login được hiển thị.</li> <li>- Có kết nối mạng.</li> </ul>	<ol style="list-style-type: none"> <li>1. Để trống trường email</li> <li>2. Nhập dữ liệu đúng cho trường</li> </ol>	3. Thông báo lỗi được hiển thị: "Vui lòng nhập Email hoặc"	

			Mật Khẩu. 3. Nhấn [Đăng Nhập] button.	Số điện thoại"	
TC_002	Kiểm tra đăng nhập khi để trống trường mật khẩu	- Màn hình Login được hiển thị. - Có kết nối mạng.	1. Để trống trường mật khẩu 2. Nhập dữ liệu đúng cho trường email. 3. Nhấn [Đăng Nhập] button.	3. Thông báo lỗi được hiển thị: "Mật khẩu không chính xác"	
TC_003	Kiểm tra đăng nhập khi để trống cả 2 trường Email và mật khẩu	- Màn hình Login được hiển thị. - Có kết nối mạng.	1. Để trống cả 2 trường email và mật khẩu 3. Nhấn [Đăng Nhập] button.	3. Thông báo lỗi được hiển thị: "Vui lòng nhập Email hoặc Số điện thoại "	
TC_004	Kiểm tra đăng nhập với email sai định dạng	- Màn hình Login được hiển thị. - Có kết nối mạng.	1. Email nhập vào sai định dạng 2. Nhập dữ liệu đúng cho trường Mật Khẩu. 3. Nhấn [Đăng	3. Thông báo lỗi được hiển thị: "Số điện thoại không hợp lệ"	

			Nhập] button.		
TC_005	Kiểm tra đăng nhập với email và mật khẩu không cùng một cặp	- Màn hình Login được hiển thị. - Có kết nối mạng.	1. Email và mật khẩu nhập vào không cùng một cặp với dữ liệu lưu trong DB 2. Nhấn [Đăng Nhập] button.	3. Thông báo lỗi được hiển thị: " <b>Số điện thoại không hợp lệ</b> " hoặc " <b>Mật khẩu không chính xác</b> "	
TC_006	Kiểm tra đăng nhập thành công	- Màn hình Login được hiển thị. - Có kết nối mạng.	1. Email và mật khẩu nhập vào cùng một cặp với dữ liệu lưu trong DB 2. Nhấn [Đăng Nhập] button.	3. Đăng nhập thành công.	

Bảng 4: Các ca kiểm thử chức năng đăng nhập

- Các ca kiểm thử cho chức năng tìm kiếm

ID	Mục đích kiểm thử	Tiền điều kiện	Các bước thực hiện	Kết quả mong muốn	Kết quả kiểm thử
TC_001	Kiểm tra tìm kiếm khi nhập	- Màn hình hiển thị được đăng nhập	1. Nhập vào trường	2. Trả về kết quả tìm kiếm cho “sách”: số	

	chữ thường	sẵn với một tài khoản. - Có kết nối mạng.	Tìm kiếm “sách”  2. Nhấn [Tìm kiếm] button.	kết quả/ số giấy	
TC_002	Kiểm tra tìm kiếm khi nhập chữ hoa	- Màn hình hiển thị được đăng nhập sẵn với một tài khoản. - Có kết nối mạng.	1. Nhập vào trường Tìm kiếm “SÁCH”  2. Nhấn [Tìm kiếm] button.	2. Trả về kết quả tìm kiếm cho “SÁCH”: số kết quả/ số giấy.  Kết quả đúng như TC_001	
TC_003	Kiểm tra tìm kiếm khi nhập chữ hoa thường lẫn lộn	- Màn hình hiển thị được đăng nhập sẵn với một tài khoản. - Có kết nối mạng.	1. Nhập vào trường Tìm kiếm “SáCH”  2. Nhấn [Tìm kiếm] button.	2. Trả về kết quả tìm kiếm cho “SáCH”: số kết quả/ số giấy.  Kết quả đúng như TC_001	
TC_004	Kiểm tra tìm kiếm khi nhập thêm dấu cách ở đầu text	- Màn hình hiển thị được đăng nhập sẵn với một tài khoản. - Có kết nối mạng.	1. Nhập vào trường Tìm kiếm “Sách”  2. Nhấn [Tìm kiếm] button.	2. Trả về kết quả tìm kiếm cho “Sách”: số kết quả/ số giấy.  Kết quả đúng như TC_001	

Bảng 5: Các ca kiểm thử cho chức năng tìm kiếm

- Các ca kiểm thử cho chức năng chọn hàng, xóa hàng

ID	Mục đích kiểm thử	Tiền điều kiện	Các bước thực hiện	Kết quả mong muốn	Kết quả kiểm thử
TC_001	Kiểm tra lựa chọn một mặt hàng	<ul style="list-style-type: none"> <li>- Màn hình hiển thị được đăng nhập sẵn với một tài khoản.</li> <li>- Có kết nối mạng.</li> </ul>	<ol style="list-style-type: none"> <li>1. Nhấn chọn Điện thoại – Máy tính bảng</li> <li>2. Nhấn chọn mặt hàng Ipad Wifif 128G.</li> <li>3. Nhấn [CHỌN MUA].</li> <li>4. Nhấn [Giỏ hàng] để chuyển đến trang giỏ hàng.</li> <li>5. Kiểm tra tên hàng hóa đã lựa chọn</li> </ol>	1. Giỏ hàng chứa sản phẩm được chọn.	
TC_002	Xóa một mặt hàng trong giỏ hàng	<ul style="list-style-type: none"> <li>- Trong giỏ hàng chứa ít nhất một mặt hàng</li> <li>- Có kết nối mạng</li> </ul>	<ol style="list-style-type: none"> <li>1. Nhấn [xóa] mặt hàng</li> <li>2. Kiểm tra số lượng mặt hàng còn lại</li> </ol>	Xóa thành công sản phẩm trong giỏ hàng	



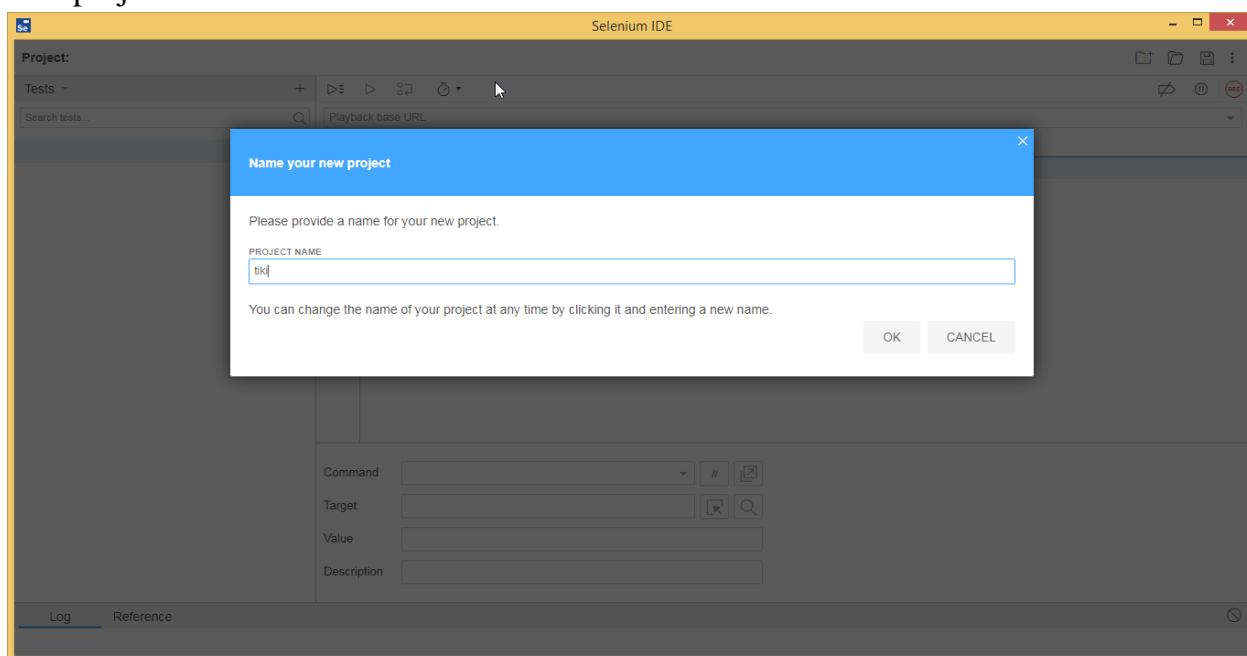
			trong giỏ hàng		
--	--	--	----------------	--	--

## 4.2. Áp dụng cho Selenium IDE

### 4.2.1. Chức năng đăng nhập

Sử dụng Selenium IDE tạo 6 test case kiểm tra chức năng đăng nhập cho trang Tiki.vn. Các bước thực hiện tạo 1 test case sử dụng chức năng record and play back như sau:

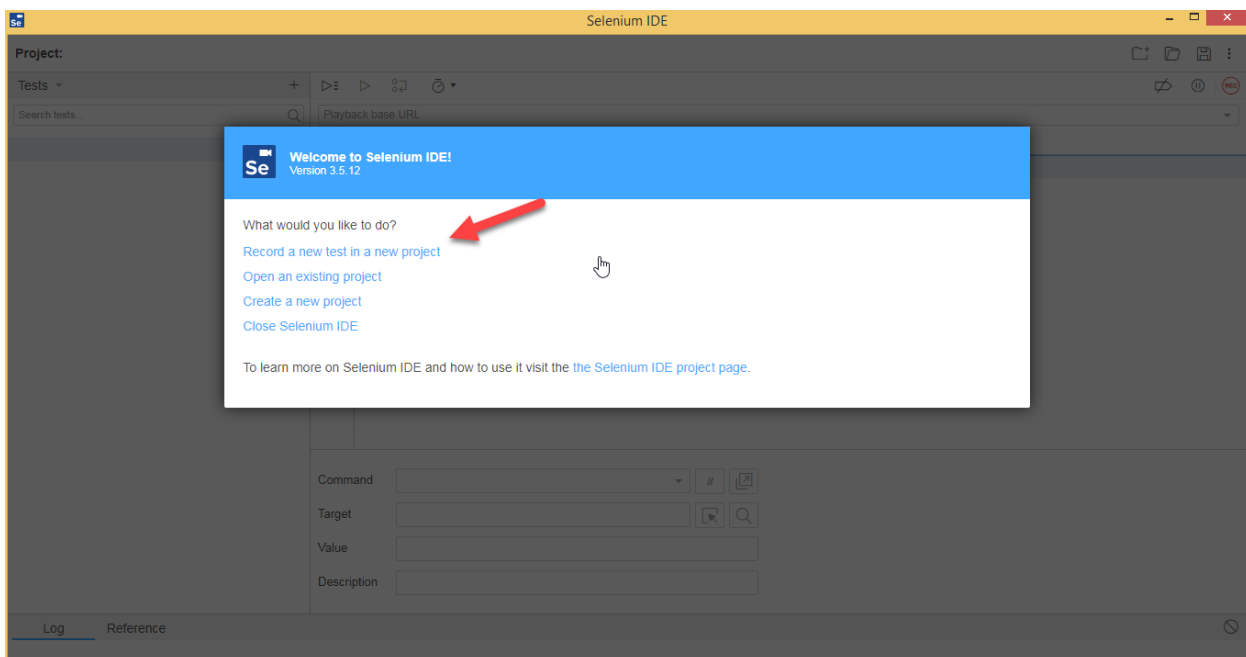
Chọn Selenium IDE extension trên trình duyệt, click chọn record a new test in new project:



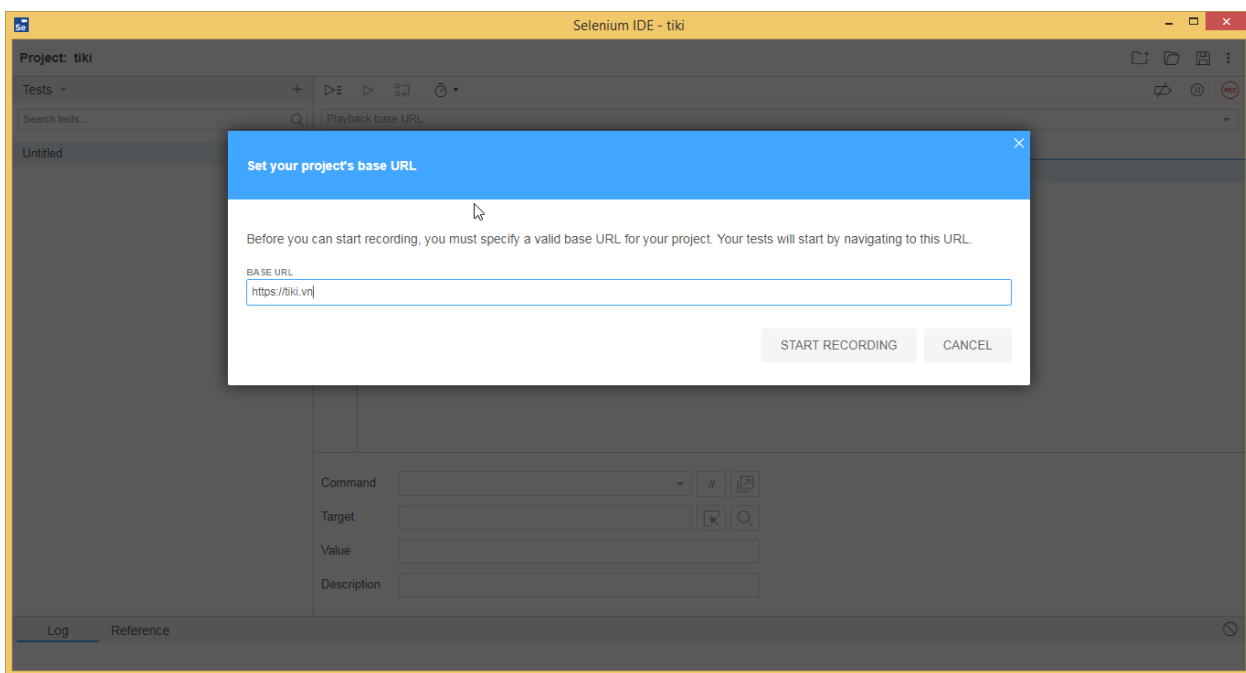
*Hình 8: Màn hình tạo mới project Selenium IDE*

Điền tên project vào trong ô project name, tại đây project name là tiki.

Điền đường dẫn trang web cần test: <https://tiki.vn>, click start recording

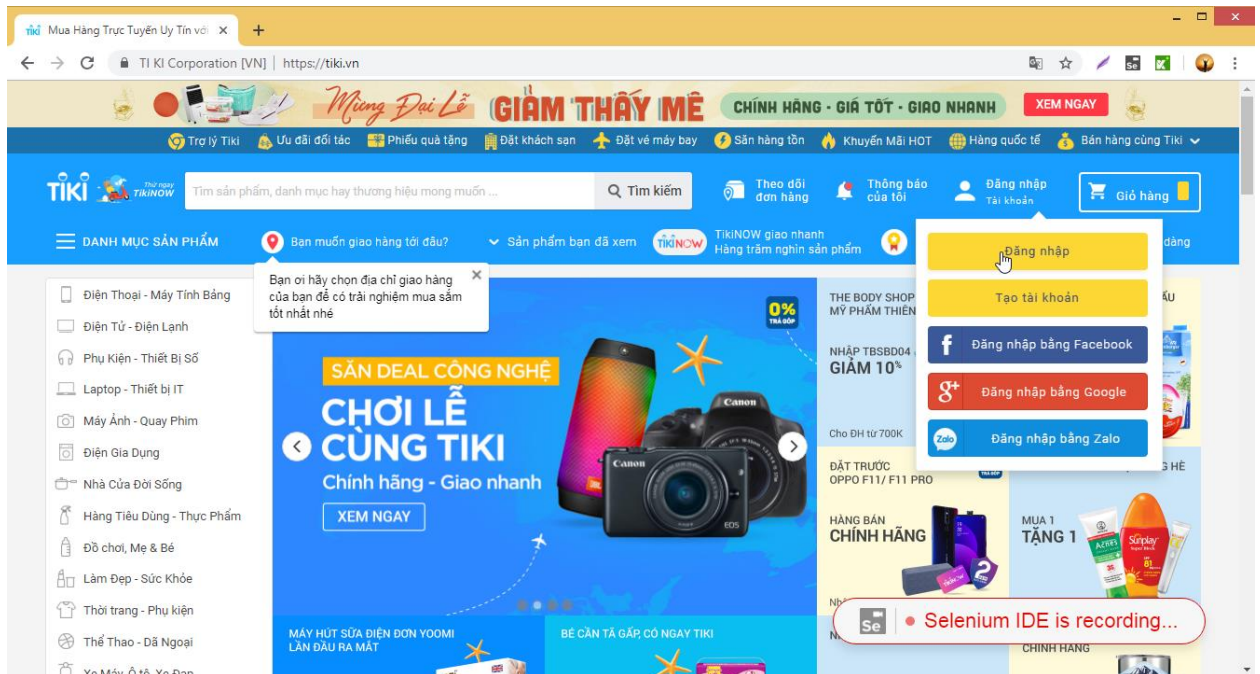


*Hình 9: Màn hình tạo mới project Selenium IDE*



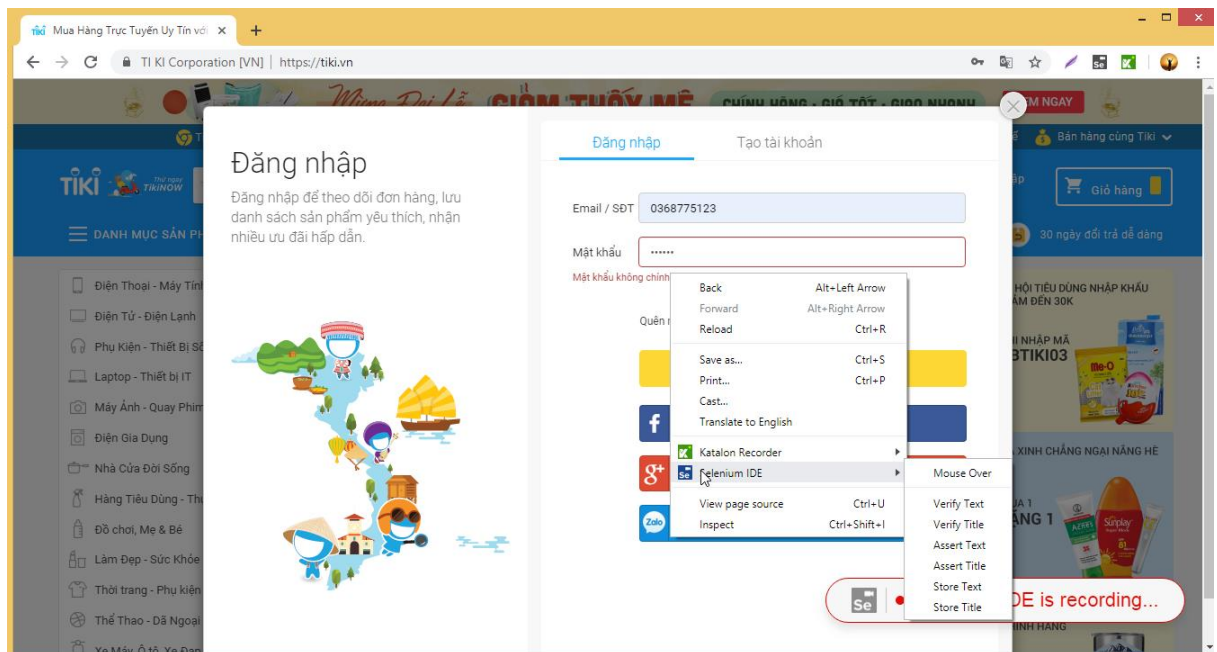
*Hình 10: Màn hình nhập địa chỉ trang cần test Tiki.vn*

Chọn đăng nhập



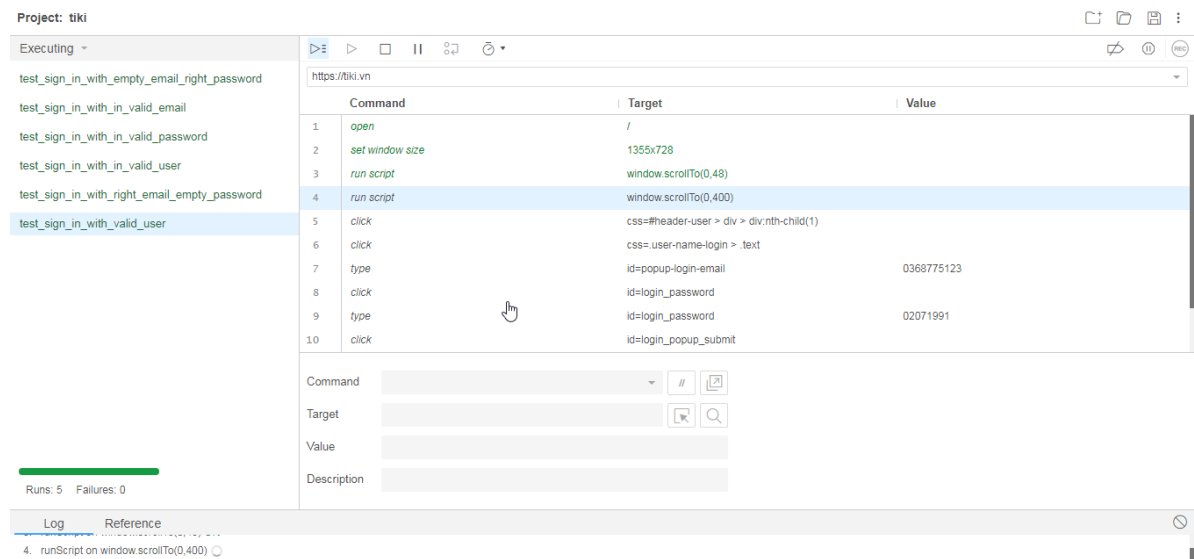
Hình 11: Màn hình chính trang Tiki.vn và vị trí chọn đăng nhập

Đăng nhập với giá trị số điện thoại đúng và giá trị mật khẩu sai, trên màn hình xuất hiện thông báo mật khẩu không đúng. Click chuột phải, chọn Selenium IDE và click Assert text. Hoàn thành xây dựng một test case kiểm tra sai password sử dụng Selenium IDE.



Hình 12: Trang đăng nhập và assert Text bằng Selenium IDE

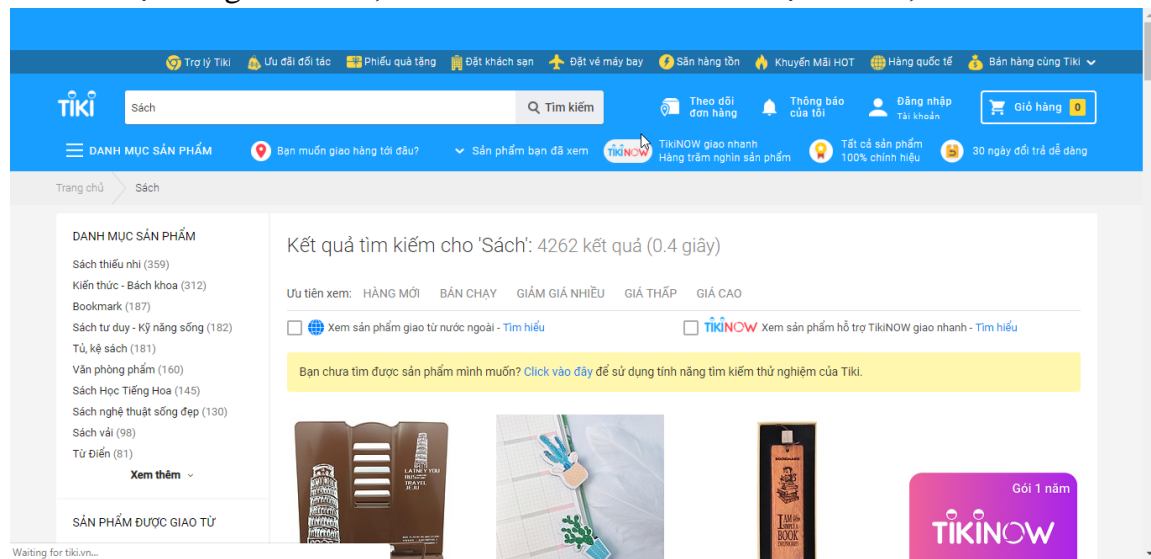
Thực hiện tương tự đối với 5 test case còn lại, thêm vào test suite và play back thu được kết quả như sau:



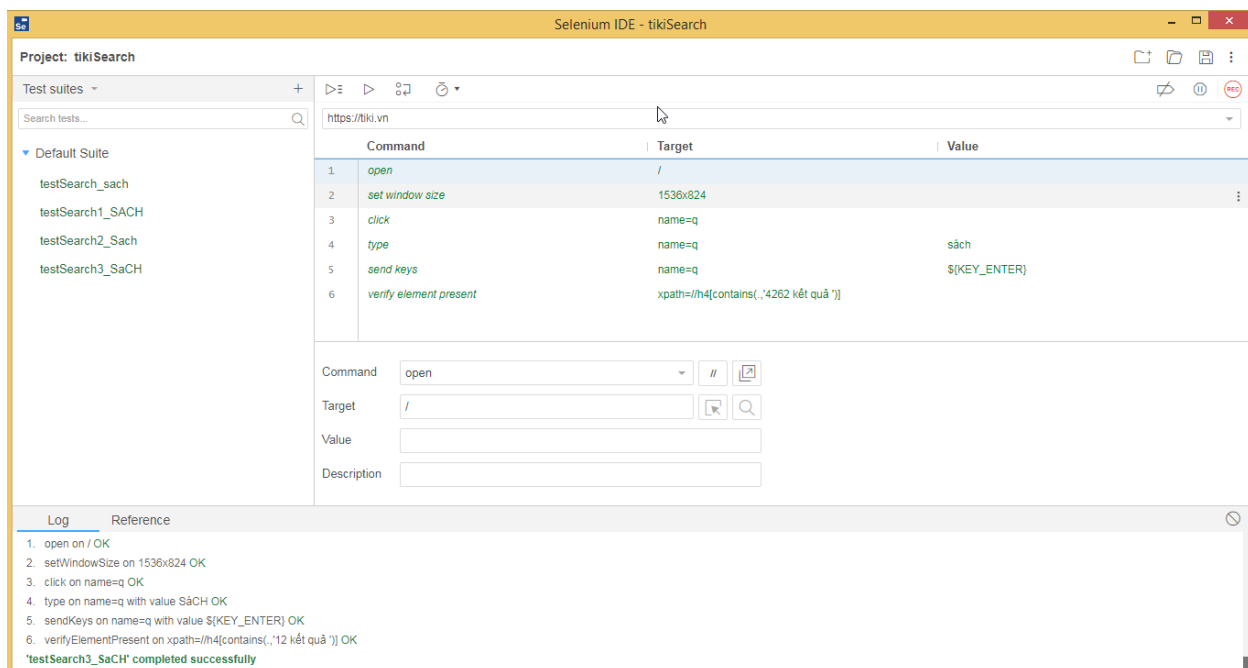
Hình 13: Kết quả chạy các test case cho trang đăng nhập bằng Selenium IDE

#### 4.2.2. Chức năng tìm kiếm

Mở Selenium IDE, thực hiện tạo test suit chứa các test case cho chức năng search. Tại trang tìm kiếm, điền vào thanh tìm kiếm ký tự “Sách”, nhấn search.

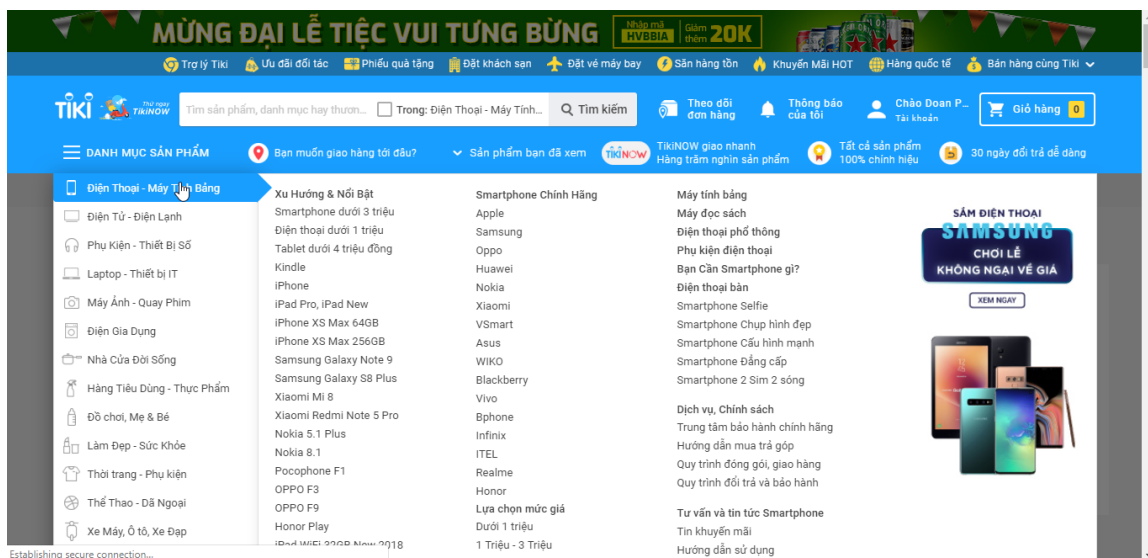


Hình 14: Nhập từ cần tìm kiếm và nhấn tìm kiếm

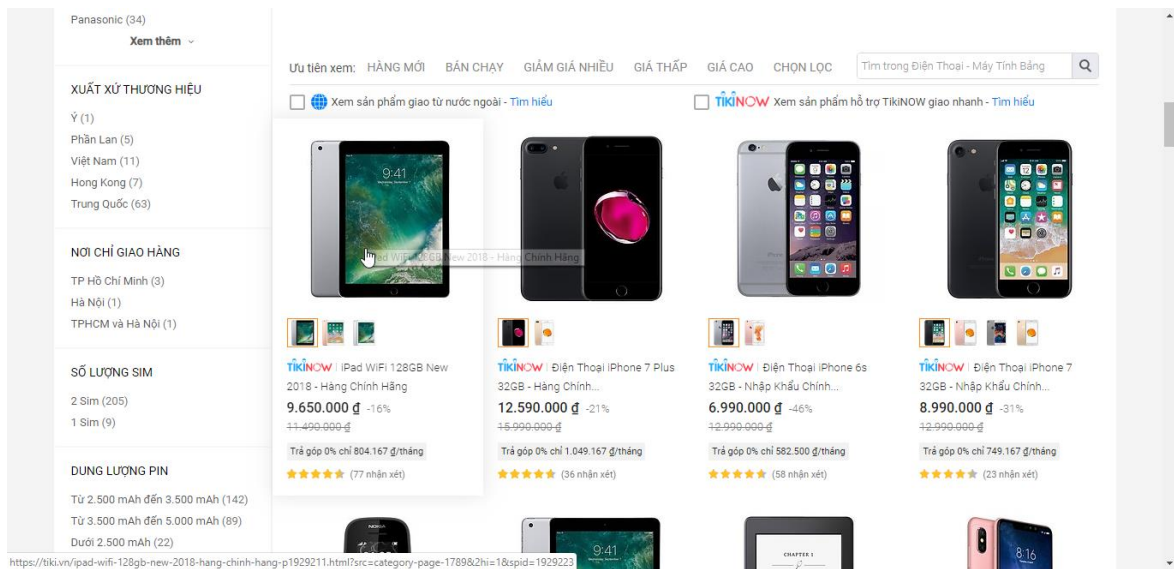


### 4.2.3. Chức năng đặt hàng và xóa hàng

Tại giao diện chính của trang tiki, lựa chọn Điện thoại – Máy tính bảng trong menu Danh mục sản phẩm.



Hình 15: Lựa chọn mặt hàng Điện thoại – Máy tính bảng

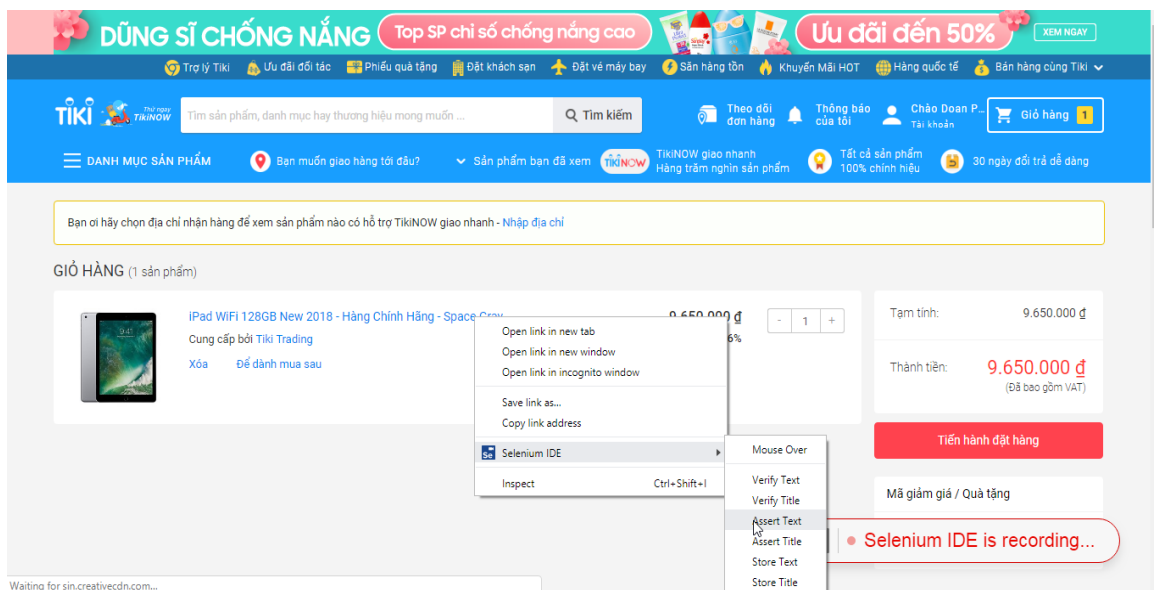


Hình 16: Click lựa chọn mặt hàng Ipad wifi



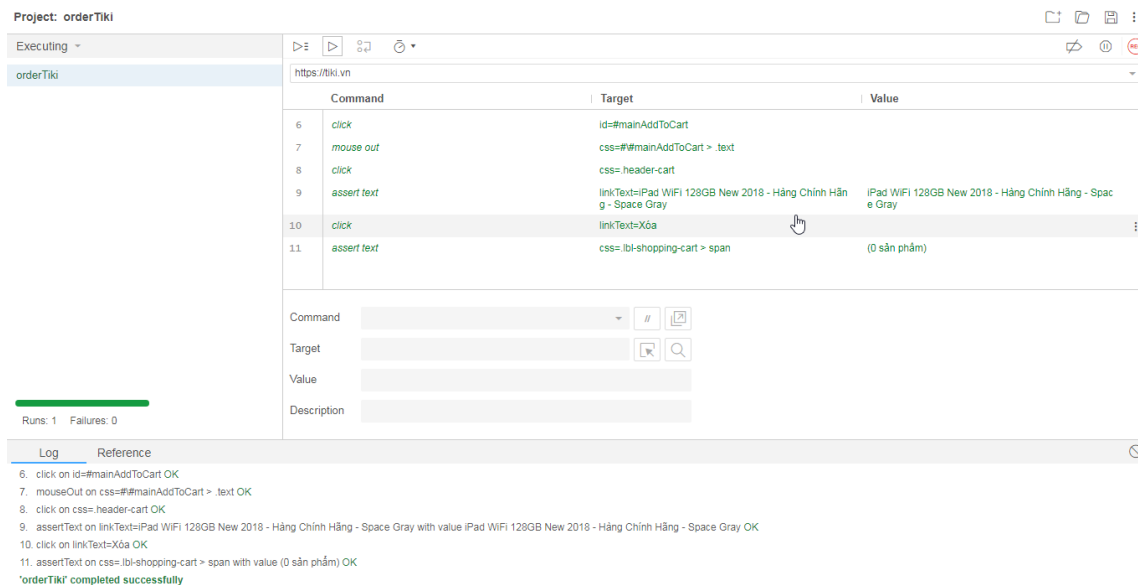
Hình 17: Click button chọn mua Ipad wifi

Vào trang cart và kiểm tra kết quả chọn mặt hàng. Bằng cách chuột phải tại tên của mặt hàng, lựa chọn Selenium IDE và assert text. Sau khi kiểm tra thông tin mặt hàng, tiến hành chọn nút xóa và kiểm tra số lượng sản phẩm trong giỏ hàng.



Hình 18: Kiểm tra thông tin mặt hàng đã chọn

Kết quả cho test case chọn hàng, xóa hàng như sau:



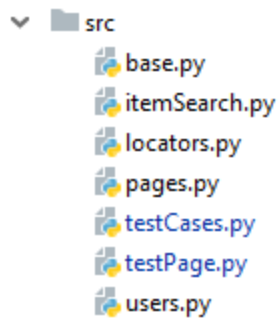
Hình 19: Kết quả thực hiện kiểm tra chức năng chọn hàng, xóa hàng



### 4.3. Áp dụng cho Selenium WebDriver

Dưới đây là cấu trúc source code sử dụng Selenium WebDriver. Cấu trúc bao gồm 7 file chính:

- base.py: Chứa các phương thức cơ bản dùng chung cho tất cả các test case.
- locators: Chứa các định danh phần tử trên giao diện.
- pages: Định nghĩa các lớp tương ứng với mỗi page trên giao diện, cung cấp các phương thức lấy thông tin phục vụ sinh testcase.
- testCases: Chứa danh sách tên của các test case cần thực hiện
- testPage: Là file định nghĩa các test case và hàm main sinh test case, tạo report.
- users: chứa thông tin của các user khác nhau phục vụ test trang login.



Hình 20: Cấu trúc source code sử dụng Selenium WebDriver

#### 4.3.1. Chức năng đăng nhập

Để test cho trang đăng nhập, đầu vào cần có là danh sách các user với các giá trị email và password khác nhau được lưu tại file users.py:



```
users.py x
1 from operator import itemgetter
2
3 # we can store test data in this module like users
4
5 users = [
6     {"name": "DoanThao", "email": "0368775123", "password": "02071991"},
7     {"name": "empty_value", "email": "", "password": ""},
8     {"name": "invalid_email", "email": "staff@test.com", "password": "02071991"},
9     {"name": "invalid_password", "email": "0368775123", "password": "qwerty1234"},
10    {"name": "empty_email_right_password", "email": "", "password": "02071991"},
11    {"name": "right_email_empty_password", "email": "0368775123", "password": ""},
12]
13
14 def get_user(name):
15     try:
16         return (user for user in users if user["name"] == name).next()
17     except:
18         raise
19     print("\n    User %s is not defined, enter a valid user.\n" % name)
```

Hình 21: Thông tin các user để test

Trên trang đăng nhập, để xác định được định danh của các thành phần như ô nhập email, ô nhập password, nút submit, vị trí hiển thị lỗi sử dụng file locators.py. Vị trí của các thành phần có thể định danh bằng id, class\_name, css\_selector...

```
locators.py x
1 from selenium.webdriver.common.by import By
2
3 # for maintainability we can separate web objects by page name
4
5
6 class MainPageLocators(object):...
13
14
15 class LoginPageLocators(object):
16     PASSWORD = (By.ID, 'login_password')
17     EMAIL = (By.ID, 'popup-login-email')
18     SUBMIT = (By.ID, 'login_popup_submit')
19     ERROR_MESSAGE = (By.CSS_SELECTOR, '#popup_password > .help-block')
20     ERROR_MESSAGE_EMAIL = (By.CSS_SELECTOR, '#popup_login > .help-block')
21     ERROR_MESSAGE_NO_VALUE = (By.CSS_SELECTOR, '#popup_login > small.help-block')
22     NAME_USER = (By.CSS_SELECTOR, '#header-user > div > div > b')
23
24 class CartLocators(object):...
27
```

Hình 22: Thông tin locator xác định vị trí các phần tử trên trang web

Thông tin về mức độ quan trọng, tên, mong muốn của từng test case sẽ được lưu tại testCases.py. Với chức năng đăng nhập thông tin của các test case có số thứ tự từ 4 đến 9.

```
locators.py x testCases.py x
3 # some manual testers (test analysts) can handle this more efficiently
4 # we can obtain test cases from test management tools, I used this for my previous project: http://www.inflectra.com/SpiraTest/Integrations/Unit-Test-Frameworks
5 # We can also record the result of test cases to test management tool
6
7 # for maintainability, we can separate web test cases by page name but I just listed every case in same array
8
9 def test_cases(number):
10     return testCases[number]
11
12
13 testCases = [
14     # [severity, description]
15     ['0', 'Low', 'when user goes to main page, page should be loaded'],
16     ['1', 'Normal', 'In Main page, when user search "pin dell insprison3537" button, he should see result for "62 items"'],
17     ['2', 'Normal', 'In Main page, when user click "Sing up" button, he should see Sign up Page'],
18     ['3', 'High', 'In Main page, when user click "Sing in" button, he should see Sign in Page'],
19     ['4', 'Normal', 'In Login Page, when user login with a empty email, right password, he should see Error Message: "Vui lòng nhập Email hoặc Số điện thoại"'],
20     ['5', 'Normal', 'In Login Page, when user login with a right email, empty password, he should see Error Message: "Mật khẩu không chính xác"'],
21     ['6', 'Normal', 'In Login Page, when user login with a empty email, empty password he should see Error Message: "Vui lòng nhập Email hoặc Số điện thoại"'],
22     ['7', 'Normal', 'In Login Page, when user login with a in-valid email, right password he should see Error Message: "Số điện thoại không hợp lệ"'],
23     ['8', 'Normal', 'In Login Page, when user login with a right email, wrong password he should see Error Message: "Mật khẩu không chính xác"'],
24     ['9', 'Low', 'In Login Page, when user login with a valid user, he should see Home Page'],
25     ['10', 'Normal', 'when user goes to main page, the title of page is Shopping online - Buy online on Lazada.vn'],
26     ['11', 'Normal', 'In Main page, when user click CUSTOMER CARE, he should see Customer Care page'],
27     ['12', 'Normal', 'In Cart page, when user click choose an item, he should see that item in Cart Page'],
28     ['13', 'Normal', 'In Cart page, when user click delete an item, he should not see that item in Cart Page'],
29     ['14', 'Normal', 'In Main page, when user search "sách" button, he should see result for 12 kết quả (0.05 giây)'],
30     ['15', 'Normal', 'In Main page, when user search "SÁCH" button, he should see result for 4200 kết quả (0.11 giây)'],
31     ['16', 'Normal', 'In Main page, when user search "SÁCH" button, he should see result for 4200 kết quả (0.11 giây)'],
32     ['17', 'Normal', 'In Main page, when user search "Sách" button, he should see result for 4199 kết quả (0.11 giây)'],
33 ]
```

Hình 23: Thông tin của các test case, kết quả mong muốn

Tại testPage.py tổng hợp kết quả và sinh test case

```
testPage.py x
42 class TestLoginPage(unittest.TestCase):
43
44     def setUp(self):
45         self.driver = webdriver.Chrome('../driver/chromedriver.exe')
46         self.driver.get('https://tiki.vn')
47
48     def test_sign_in_with_empty_email_right_password(self):
49         print("\n" + str(test_cases(4)))
50         mainPage = MainPage(self.driver)
51         loginPage = mainPage.click_sign_in_button()
52         result = loginPage.login_with_no_value("empty_email_right_password")
53         self.assertIn("Vui lòng nhập Email hoặc Số điện thoại", result)
54
55     def test_sign_in_with_right_email_empty_password(self):...
56
57     def test_sign_in_with_in_valid_user(self):...
58
59     def test_sign_in_with_in_valid_email(self):...
60
61     def test_sign_in_with_in_valid_password(self):...
62
63     def test_sign_in_with_valid_user(self):...
64
65     def tearDown(self):
66         self.driver.close()
```

Hình 24: Ví dụ một test case cho trang đăng nhập

Kết quả thu được lưu lại dưới định dạng html:

## Unittest Results

Start Time: 2019-04-17 05:39:46

Duration: 219.31 s

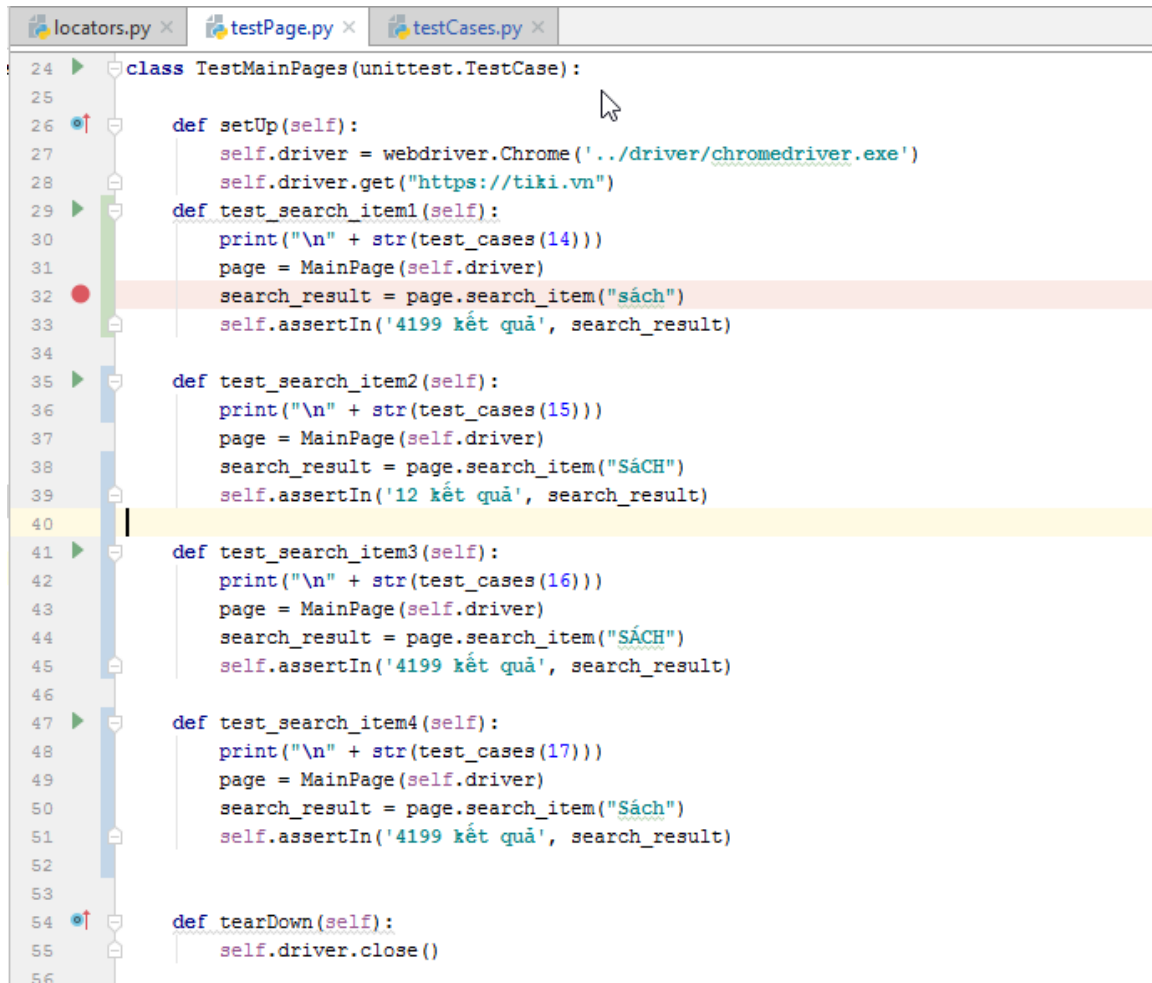
Summary: Total: 6, Pass: 6

__main__.TestLoginPage		Status
test_sign_in_with_empty_email_right_password		Pass Hide
['4', 'Normal', 'In Login Page, when user login with a empty email, right password, he should see Error Message: "Vui lòng nhập Email hoặc Số điện thoại"']		
test_sign_in_with_in_valid_email		Pass Hide
['7', 'Normal', 'In Login Page, when user login with a in-valid email, right password he should see Error Message: "Số điện thoại không hợp lệ"']		
test_sign_in_with_in_valid_password		Pass Hide
['8', 'Normal', 'In Login Page, when user login with a right email, wrong password he should see Error Message: "Mật khẩu không chính xác"']		
test_sign_in_with_in_valid_user		Pass Hide
['6', 'Normal', 'In Login Page, when user login with a empty email, empty password he should see Error Message: "Vui lòng nhập Email hoặc Số điện thoại"']		
test_sign_in_with_right_email_empty_password		Pass Hide
['5', 'Normal', 'In Login Page, when user login with a right email, empty password, he should see Error Message: "Mật khẩu không chính xác"']		
test_sign_in_with_valid_user		Pass Hide
['9', 'Low', 'In Login Page, when user login with a valid user, he should see Home Page']		
Total: 6, Pass: 6 -- Duration: 219.31 s		

*Hình 25: Kết quả các test case trang đăng nhập*

### 4.3.2. Chức năng tìm kiếm

Trong class TestMainPages chứa 4 phương thức tương ứng test case cho chức năng search:



```
24 class TestMainPages(unittest.TestCase):
25
26     def setUp(self):
27         self.driver = webdriver.Chrome('../driver/chromedriver.exe')
28         self.driver.get("https://tiki.vn")
29     def test_search_item1(self):
30         print("\n" + str(test_cases(14)))
31         page = MainPage(self.driver)
32         search_result = page.search_item("sách")
33         self.assertIn('4199 kết quả', search_result)
34
35     def test_search_item2(self):
36         print("\n" + str(test_cases(15)))
37         page = MainPage(self.driver)
38         search_result = page.search_item("SÁCH")
39         self.assertIn('12 kết quả', search_result)
40
41     def test_search_item3(self):
42         print("\n" + str(test_cases(16)))
43         page = MainPage(self.driver)
44         search_result = page.search_item("SÁCH")
45         self.assertIn('4199 kết quả', search_result)
46
47     def test_search_item4(self):
48         print("\n" + str(test_cases(17)))
49         page = MainPage(self.driver)
50         search_result = page.search_item("Sách")
51         self.assertIn('4199 kết quả', search_result)
52
53
54     def tearDown(self):
55         self.driver.close()
56
```

Hình 26: Các test case cho chức năng tìm kiếm

Kết quả thu được với các test case cho chức năng search như sau:

## Unittest Results

Start Time: 2019-04-17 21:38:10

Duration: 142.97 s

Summary: Total: 4, Pass: 4

__main__.TestMainPages	Status
test_search_item1	Pass Hide
['14', 'Normal', 'In Main page, when user search "sách" button, he should see result for 12 kết quả (0.05 giây)']	
test_search_item2	Pass Hide
['15', 'Normal', 'In Main page, when user search "SÁCH" button, he should see result for 4200 kết quả (0.11 giây)']	
test_search_item3	Pass Hide
['16', 'Normal', 'In Main page, when user search "SÁCH" button, he should see result for 4200 kết quả (0.11 giây)']	
test_search_item4	Pass Hide
['17', 'Normal', 'In Main page, when user search "Sách" button, he should see result for 4199 kết quả (0.11 giây)']	
Total: 4, Pass: 4 -- Duration: 142.97 s	

Hình 27: Kết quả test case chức năng tìm kiếm

### 4.3.3. Chức năng đặt hàng, xóa hàng

Các giá trị định vị các phần tử trên trang web, CART vị trí nút bấm chuyển sang trang giỏ hàng, MAIN\_NAV xác định vị trí của mặt hàng Điện thoại – Máy tính bảng, ITEM chỉ ra vị trí của mặt hàng cần lựa chọn, ADD\_TO\_CARD xác định nút bấm thêm hàng vào giỏ hàng, NAME\_ITEM vị trí ghi tên của mặt hàng.

```
class CartLocators(object):  
    CART = (By.CSS_SELECTOR, '.header-cart')  
    MAIN_NAV = (By.XPATH, '//span[contains(., "Điện Thoại - Máy Tính Bảng")]')  
    ITEM = (By.CSS_SELECTOR, '.product-box-list > .product-item:nth-child(2) .content')  
    ADD_TO_CARD = (By.ID, '#mainAddToCart')  
    NAME_ITEM = (By.CSS_SELECTOR, '.badge-tikinow-a>.name>a')
```

Hình 28: Vị trí các phần tử web cho chức năng thêm hàng và xóa hàng

Sau khi xác định được các phần tử trên trang web tiến hành thực hiện đặt hàng bằng Selenium WebDriver, thông qua các lệnh find\_element tìm phần tử web, click thực hiện thao tác nhấn vào button.

```
def choose_an_item(self):
    self.find_element(*self.locator.MAIN_NAV).click()
    self.find_element(*self.locator.ITEM).click()
    self.find_element(*self.locator.ADD_TO_CART).click()
    time.sleep(3)
    self.find_element(*self.locator.CART).click()
    return self.find_element(*self.locator.NAME_ITEM).text
```

Hình 29: Source code chọn một mặt hàng bằng webdriver

Tổng hợp kết quả tại trang testPage bằng cách tạo class TestCartPage trong đó chứa hàm test\_choose\_item thực hiện chọn mặt hàng và kiểm tra tên mặt hàng đã được chọn bằng lệnh assertIn.

```
class TestCartPage(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome('../driver/chromedriver.exe')
        self.driver.get('https://tiki.vn/')

    def test_choose_item(self):
        print("\n" + str(test_cases(10)))
        cartPage = CartPage(self.driver)
        result = cartPage.choose_an_item()
        time.sleep(3)
        self.assertIn("iPad WiFi 128GB New 2018 - Hàng Chính Hãng - Space Gray", result)

    def tearDown(self):
        self.driver.close()
```

Hình 30: Source code sinh test case chọn mặt hàng webdriver

Kết quả thu được như sau:

## Unittest Results

Start Time: 2019-04-24 13:15:25

Duration: 92.51 s

Summary: Total: 1, Pass: 1

__main__.TestCartPage		Status
test_choose_item		Pass <input type="button" value="Hide"/>
['10', 'Normal', 'when user choose an item, he should see it in the cart']		
Total: 1, Pass: 1 -- Duration: 92.51 s		

Hình 31: Kết quả test case chọn mặt hàng

#### 4.4. Áp dụng cho Selenium Grid.

Trong phần này, để thực hiện sinh ca kiểm thử cho nhiều trình duyệt cùng lúc bằng cách sử dụng Selenium Grid. Trước hết cần tạo hub server bằng cách chạy lệnh dưới đây:

```
java -jar selenium-server-standalone-3.141.59.jar -role hub
```

Trong đó selenium-server-standalone-3.141.59.jar là file chạy server cần download về. Tiếp theo, tạo các node liên kết với hub thông qua port 4444 với hai lệnh như sau tại hai command line khác nhau:

```
java -jar selenium-server-standalone-3.141.59.jar -role node -hub  
http://local_ip:4444/grid/register/ -port 3456
```

```
java -jar selenium-server-standalone-3.141.59.jar -role node -hub  
http://local_ip:4444/grid/register/ -port 4567
```

Trong hai câu lệnh trên, local\_ip là địa chỉ ip của máy tính hiện tại đang dùng. Sau khi đã có hai node tiến hành tạo file source code test\_tiki\_order.py chứa ca kiểm thử dùng để chạy trên nhiều trình duyệt:

```
class TikiTestOrder(unittest.TestCase):  
  
    def setUp(self):  
        self.driver = webdriver.Remote(  
            command_executor='http://127.0.0.1:4444/wd/hub',  
            desired_capabilities={'browserName': 'chrome', 'javascriptEnabled': True})  
  
    def test_order(self):  
        driver = self.driver  
        # Here we will implement login into github (PART 1)  
        driver.get("https://tiki.vn")  
        driver.find_element_by_xpath('//span[contains(., "Điện Thoại - Máy Tính Bảng")]').click()  
        driver.find_element_by_css_selector('.product-box-list > .product-item:nth-child(2) .content').click()  
        driver.find_element_by_id('#mainAddToCart').click()  
        driver.find_element_by_class_name('.header-cart').click()  
        driver.assertIn('iPad WiFi 128GB New 2018 - Hàng Chính Hãng - Space Gray',  
            driver.find_element_by_css_selector('.badge-tikinow-a>.name>').text)  
  
    def tearDown(self):  
        self.driver.close()  
  
if __name__ == "__main__":  
    unittest.main()
```

Hình 32: Source code kiểm thử selenium grid cho nhiều trình duyệt

Cuối cùng tiến hành chạy sinh ca kiểm thử trên nhiều trình duyệt, mở hai command line khác nhau và chạy lệnh sau:

```
python test_tiki_order.py
```

Kết quả thu được cho việc sinh test case tương tự như mục 3 webdriver

#### 4.5. Đánh giá và đưa ra phương pháp luận

Như vậy cả bốn công cụ đều có những hạn chế và điểm mạnh riêng. Trong từng trường hợp, người dùng có thể lựa chọn một trong bốn công cụ để thực thi kiểm thử một cách hiệu quả nhất.

- Selenium IDE: một add-on của trình duyệt, phù hợp với các ca kiểm thử đơn giản và không đòi hỏi người sử dụng phải có kiến thức về lập trình.
  - Xuất các trường hợp thử nghiệm ở các định dạng khác nhau.
  - Tạo các ca kiểm thử có ít hoặc không có kiến thức về lập trình.
  - Phù hợp tạo các ca kiểm thử không quá phức tạp và các bộ kiểm thử có thể xuất sau đó đến Selenium RC hoặc Selenium WebDriver.
- Selenium RC:
  - Phù hợp với các ca kiểm thử cần thực hiện trên nhiều trình duyệt khác nhau (ngoại trừ HtmlUnit) trên các hệ điều hành khác nhau.
  - Để triển khai kiểm thử trên nhiều môi trường sử dụng Selenium Grid.
  - Để kiểm tra ứng dụng trên một trình duyệt mới hỗ trợ JavaScript.
  - Để kiểm tra các ứng dụng web với các kịch bản dựa trên AJAX phức tạp.
  - Hiện tại Selenium RC không còn được sử dụng nữa.
- Selenium WebDriver:
  - Để sử dụng một ngôn ngữ lập trình nhất định trong việc thiết kế các trường hợp kiểm thử.
  - Sử dụng khi cần thao tác với database.
  - Tạo test case với điều kiện và vòng lặp phức tạp.
  - Để kiểm tra các ứng dụng có nhiều chức năng dựa trên AJAX.
  - Để thực hiện các bài kiểm tra trên trình duyệt HtmlUnit.
  - Tạo ra các kết quả kiểm tra tùy chỉnh.
- Selenium Grid:
  - Để chạy các kịch bản Selenium RC trong nhiều trình duyệt và hệ điều hành cùng lúc.
  - Thực thi một bộ phần mềm kiểm thử không lỗi, cần phải hoàn thành trong thời gian sớm nhất có thể.



## KẾT LUẬN

Kiểm thử tự động là phương pháp hữu dụng và có nhiều ưu điểm vượt trội trong kiểm thử phần mềm. Không thể phủ nhận những lợi ích to lớn của kiểm thử tự động mặc dù không phải dự án nào cũng phù hợp để áp dụng. Nhờ kiểm thử tự động, các kiểm thử viên không phải lặp đi lặp lại quá trình kiểm thử thủ công nhàm chán mà vẫn đảm bảo độ tin cậy, tính ổn định cao và có thể tái sử dụng. Khóa luận đã nghiên cứu, phân tích qua đó đề xuất phương pháp sử dụng hiệu quả các công cụ Selenium. Xét tổng quan, mỗi công cụ Selenium đều có ưu nhược điểm riêng và phù hợp với từng trường hợp kiểm thử. Selenium IDE phù hợp với các ca kiểm thử đơn giản và kiểm thử viên có ít kiến thức lập trình. Selenium WebDriver phù hợp với những ca kiểm thử có điều kiện và vòng lặp phức tạp hơn, phải thao tác với cơ sở dữ liệu. Selenium Grid phù hợp với những ca kiểm thử yêu cầu về mặt thời gian nên cần chạy song song trên nhiều trình duyệt và hệ điều hành cùng một lúc.

Tuy nhiên, trong quá trình thực hiện nghiên cứu em nhận thấy có những trường hợp kiểm thử chưa thể áp dụng bằng công cụ Selenium nói riêng hay tự động hóa nói chung. Trong tương lai, em sẽ cố gắng nghiên cứu thêm để tìm các phương pháp khắc phục những hạn chế của công cụ kiểm thử Selenium và áp dụng vào các bài toán phức tạp hơn trong thực tế, góp phần tăng tính hiệu quả của Selenium trong kiểm thử tự động.

## **TÀI LIỆU THAM KHẢO**

- [1] <https://www.softwaretestingclass.com/introduction-to-selenium-grid/>
- [2] <https://www.tutorialspoint.com/selenium>
- [3] <https://viblo.asia/p/bai-1-gioi-thieu-ve-selenium-aWj538VwK6m>
- [4] <https://www.seleniumhq.org/>
- [5] <https://gist.github.com/dzitkowskik/0fc641cf59af0dc3de62>