|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **KHOA CÔNG NGHỆ THÔNG TIN**  ---------------------------------------  Ảnh có chứa văn bản, màu vàng, biểu tượng, vòng tròn  Mô tả được tạo tự động  BÁO CÁO THỰC NGHIỆM  HỌC PHẦN: KIỂM THỬ PHẦN MỀM  **SỬ DỤNG SELENIUM IDE KIỂM THỬ TỰ ĐỘNG GIAO DIỆN WEBSITE CANIFA**  Nhóm: 17   |  |  | | --- | --- | | **CBHD:***TS.* Hà Mạnh Đào | | | **Sinh viên:** | Bùi Quang Hiếu 2021602449 | |  | Nguyễn Hữu Hùng 2021603061 | |  | Nguyễn An Huy 2021603126 |   *Hà Nội – Năm 2024* |

**LỜI CẢM ƠN**

Trước tiên, chúng em xin bày tỏ lòng biết ơn sâu sắc đến giảng viên học phần "Kiểm thử phần mềm" – TS. Hà Mạnh Đào, người đã tận tình giảng dạy và chia sẻ những kiến thức vô cùng quý giá trong suốt quá trình học tập vừa qua. Sự tâm huyết và chuyên môn của thầy không chỉ giúp chúng em nâng cao hiểu biết mà còn truyền cảm hứng học tập và rèn luyện thái độ làm việc nghiêm túc, chuyên nghiệp.

Nhờ sự hướng dẫn chi tiết và tận tình từ thầy, cùng với những ý kiến đóng góp quý báu từ các bạn trong lớp, nhóm chúng em đã nỗ lực hoàn thành bài báo cáo này. Đây là cơ hội để chúng em củng cố kiến thức và phát triển các kỹ năng cần thiết, làm hành trang vững chắc cho chặng đường tương lai.

Chúng em rất mong nhận được những lời nhận xét và góp ý từ thầy để bài báo cáo được hoàn thiện hơn, cũng như giúp chúng em nâng cao năng lực cho các dự án tiếp theo.

Một lần nữa, chúng em xin chân thành cảm ơn thầy cùng các bạn đã đồng hành, hỗ trợ chúng em trong suốt quá trình học tập và thực hiện bài báo cáo.

Chúng em xin chân thành cảm ơn!

**Nhóm sinh viên thực hiện**

Nhóm 17

**MỤC LỤC**

[DANH MỤC CÁC BẢNG 6](#_Toc185979597)

[DANH MỤC HÌNH ẢNH 7](#_Toc185979598)

[PHẦN 1. MỞ ĐẦU 9](#_Toc185979599)

[1.1 Lý do chọn đề tài 9](#_Toc185979600)

[1.2 Mục tiêu của đề tài 10](#_Toc185979601)

[1.3 Giới hạn và phạm vi nghiên cứu 10](#_Toc185979602)

[1.4 Nội dung thực hiện 10](#_Toc185979603)

[1.5 Phương pháp tiếp cận 10](#_Toc185979604)

[PHẦN 2. KIẾN THỨC NỀN TẢNG 11](#_Toc185979605)

[2.1 Quy trình kiểm thử phần mềm 11](#_Toc185979606)

[2.1.1 Phân tích yêu cầu 11](#_Toc185979607)

[2.1.2 Lập kế hoạch 12](#_Toc185979608)

[2.1.3 Thiết kế kiểm thử 12](#_Toc185979609)

[2.1.4 Thiết lập môi trường kiểm thử 13](#_Toc185979610)

[2.1.5 Thực thi kiểm thử 13](#_Toc185979611)

[2.1.6 Kết thúc 14](#_Toc185979612)

[2.2 Kiểm thử hộp trắng 14](#_Toc185979613)

[2.2.1 Kiểm thử dòng điều kiện – Đồ thị dòng 15](#_Toc185979614)

[2.2.2 Kiểm thử dựa trên luồng điều kiện 17](#_Toc185979615)

[2.3 Kiểm thử hộp đen 18](#_Toc185979616)

[2.3.1 Đoán lỗi 19](#_Toc185979617)

[2.3.2 Kiểm thử dựa vào đồ thị nguyên nhân – kết quả 19](#_Toc185979618)

[2.3.3 Phân vùng tương đương 19](#_Toc185979619)

[2.3.4 Phân tích giá trị biên 21](#_Toc185979620)

[PHẦN 3. KẾT QUẢ NGHIÊN CỨU 22](#_Toc185979621)

[3.1 Tổng quan 22](#_Toc185979622)

[3.1.1 Khái niệm và mục đích của kiểm thử phần mềm 22](#_Toc185979623)

[3.1.1.1 Khái niệm về kiểm thử phần mềm 22](#_Toc185979624)

[3.1.1.2 Mục đích của việc kiểm thử phần mềm 23](#_Toc185979625)

[3.1.2 Sự quan trọng của kiểm thử phần mềm 23](#_Toc185979626)

[3.1.3 Các giai đoạn kiểm thử phần mềm 25](#_Toc185979627)

[3.1.3.1 Kiểm thử đơn vị (Unit tesing) 26](#_Toc185979628)

[3.1.3.2 Kiểm thử tích hợp (Integration Testing) 26](#_Toc185979629)

[3.1.3.3 Kiểm thử hệ thống (System Testing) 30](#_Toc185979630)

[3.1.3.4 Kiểm thử chấp nhận (Acceptance Testing) 31](#_Toc185979631)

[3.1.4 Các phương pháp kiểm thử phần mềm 32](#_Toc185979632)

[3.1.4.1 Kiểm thử tĩnh – Static testing: 32](#_Toc185979633)

[3.1.4.2 Kiểm thử động – Dynamic testing: 32](#_Toc185979634)

[3.2 TÌM HIỂU CÔNG CỤ KIỂM THỬ PHẦN NỀM 32](#_Toc185979635)

[3.2.1 Giới thiệu công cụ 32](#_Toc185979636)

[3.2.2 Đặc điểm 33](#_Toc185979637)

[3.2.3 Cài đặt và sử dụng công cụ Selenium IDE 33](#_Toc185979638)

[3.2.3.1 Cách cài đặt 33](#_Toc185979639)

[3.2.3.2 Cách sử dụng 34](#_Toc185979640)

[3.2.3.4 Thực hiện một kịch bản test case với Selenium IDE 35](#_Toc185979641)

[3.3 GIỚI THIỆU TRANG WEB CANIFA 37](#_Toc185979642)

[3.3.1 Giới thiệu 37](#_Toc185979643)

[3.3.2 Chức năng phần mềm 38](#_Toc185979644)

[3.4 Kiểm thử trang website Canifa 38](#_Toc185979645)

[3.4.1 Lập kế hoạch kiểm thử (test plan) 38](#_Toc185979646)

[Chiến lược kiểm thử: 38](#_Toc185979647)

[3.4.2 Kiểm thử tìm kiếm tin tức 41](#_Toc185979648)

[3.4.2.1 Phân tích thiết kế kiểm thử 41](#_Toc185979649)

[3.4.2.2 Thực hiện kiểm thử 43](#_Toc185979650)

[3.4.3 Kiểm thử chức năng giỏ hàng 48](#_Toc185979651)

[3.4.3.1 Phân tích thiết kế kiểm thử 48](#_Toc185979652)

[3.4.3.2 Thực hiện kiểm thử 49](#_Toc185979653)

[3.4.4 Kiểm thử chức năng tìm kiếm sản phẩm 53](#_Toc185979654)

[3.4.4.1 Phân tích thiết kế kiểm thử 53](#_Toc185979655)

[3.4.4.2 Thực hiện kiểm thử 55](#_Toc185979656)

[3.4.6 Báo cáo kiểm thử 59](#_Toc185979657)

[PHẦN 4. KIẾN THỨC LĨNH HỘI VÀ BÀI HỌC KINH NGHIỆM 61](#_Toc185979658)

[PHẦN 5. TÀI LIỆU THAM KHẢO 63](#_Toc185979659)

# DANH MỤC CÁC BẢNG

[Bảng 3.1 Kiểm thử chức năng 38](#_Toc185979534)

[Bảng 3.2 Kiểm thử giao diện người dùng. 39](#_Toc185979535)

[Bảng 3.3 Giai đoạn kiểm thử. 40](#_Toc185979536)

[Bảng 3.4 Các công cụ sử dụng. 40](#_Toc185979537)

[Bảng 3.5 Danh sách các test case (1) 42](#_Toc185979538)

[Bảng 3.6 Danh sách các test case (2) 49](#_Toc185979539)

[Bảng 3.7 Danh sách các test case (3) 54](#_Toc185979540)

[Bảng 3.8 Báo cáo kiểm thử 59](#_Toc185979541)

# DANH MỤC HÌNH ẢNH

[Hình 2. 1 : Vòng đời kiểm thử phần mềm 11](#_Toc185979549)

[Hình 2. 2 Mô hình kiểm thử hộp trắng 15](#_Toc185979550)

[Hình 2. 3 Các thành phần kiểm thử dòng điều kiện 15](#_Toc185979551)

[Hình 2. 4 Đồ thị của các cấu trúc lặp và rẽ nhánh 16](#_Toc185979552)

[Hình 2. 5 Ví dụ về đồ thị dòng điều khiển 16](#_Toc185979553)

[Hình 2. 6 Ví dụ về đồ thị dòng điều khiển 17](#_Toc185979554)

[Hình 2. 7 Ví dụ về dòng điều kiện 18](#_Toc185979555)

[Hình 2. 8 Kiểm thử hộp đen 19](#_Toc185979556)

[Hình 2. 9 Phân vùng tương đương 20](#_Toc185979557)

[Hình 2. 10 Phân tích giá trị biên 21](#_Toc185979558)

[Hình 3. 1 Các cấp độ kiểm thử phần mềm 25](#_Toc185979559)

[Hình 3. 2 Phương pháp Big Bang 27](#_Toc185979560)

[Hình 3. 3 Phương pháp Top Down. 28](#_Toc185979561)

[Hình 3. 4 Phương pháp Bottom Up. 29](#_Toc185979562)

[Hình 3. 5 Phương pháp Sandwich/Hybrid. 29](#_Toc185979563)

[Hình 3.6 Download Selenium IDE 33](#_Toc185979564)

[Hình 3.7 Kết quả cài đặt 34](#_Toc185979565)

[Hình 3. 8 Các thành phần của Selenium IDE. 34](#_Toc185979566)

[Hình 3. 9 Tạo Project mới. 35](#_Toc185979567)

[Hình 3. 10 Nhập URL của trang web. 35](#_Toc185979568)

[Hình 3. 11 Tiến hành record test case. 36](#_Toc185979569)

[Hình 3.12 Lưu test case 36](#_Toc185979570)

[Hình 3. 13 Chạy test case. 37](#_Toc185979571)

[Hình 3. 14 Click vào mục tin tức. 43](#_Toc185979572)

[Hình 3. 15 Nhập từ khóa. 43](#_Toc185979573)

[Hình 3. 16 Tìm kiếm. 44](#_Toc185979574)

[Hình 3. 17 Kết quả test case 1 (1). 44](#_Toc185979575)

[Hình 3. 18 Kết quả test case 2 (1). 45](#_Toc185979576)

[Hình 3. 19 Kết quả test case 3 (1). 45](#_Toc185979577)

[Hình 3. 20 Kết quả test case 4 (1). 46](#_Toc185979578)

[Hình 3. 21 Kết quả test case 5 (1). 46](#_Toc185979579)

[Hình 3. 22 Kết quả test case 6 (1). 47](#_Toc185979580)

[Hình 3. 23 Kết quả test case 7 (1). 47](#_Toc185979581)

[Hình 3. 24 Chọn một sản phẩm. 50](#_Toc185979582)

[Hình 3. 25 Chọn màu và kích cỡ cho sản phẩm. 50](#_Toc185979583)

[Hình 3. 26 Thêm vào giỏ hàng. 51](#_Toc185979584)

[Hình 3. 27 Xem giỏ hàng 51](#_Toc185979585)

[Hình 3. 28 Kết quả test case 1 (2). 52](#_Toc185979586)

[Hình 3. 29 Kết quả test case 2 (2). 52](#_Toc185979587)

[Hình 3. 30 Kết quả test case 3 (2). 53](#_Toc185979588)

[Hình 3. 31 Tìm kiếm từ khóa. 55](#_Toc185979589)

[Hình 3. 32 Kết quả tìm kiếm. 56](#_Toc185979590)

[Hình 3. 33 Kết quả test case 1 (3). 56](#_Toc185979591)

[Hình 3. 34 Kết quả test case 2 (3). 57](#_Toc185979592)

[Hình 3. 35 Kết quả test case 3 (3). 57](#_Toc185979593)

[Hình 3. 36 Kết quả test case 4 (3). 58](#_Toc185979594)

[Hình 3. 37 Kết quả test case 5 (3). 58](#_Toc185979595)

[Hình 3. 38 Kết quả test case 6 (3). 59](#_Toc185979596)

# PHẦN 1. MỞ ĐẦU

## 1.1 Lý do chọn đề tài

Trong giai đoạn phát triển của công nghệ thông tin, ngành công nghệ phần mềm đang ngày một chiếm vị trí quan trọng trong xu hướng phát triển kinh tế công nghiệp hóa, hiện đại hóa của đất nước ta. Cùng với sự phát triển của công nghệ phần mềm, lỗi phần mềm và chất lượng phần mềm luôn là thách thức lớn để phần mềm khi thực tế đã chứng minh, kiểm thử phần mềm là giai đoạn chiếm đến hơn 30%-70% thời gian, kinh phí và nguồn nhân lực phát triển dự án phần mềm (tùy theo loại phần mềm và lĩnh vực). Tuy nhiên ở Việt Nam hiện nay, việc kiểm thử phần mềm vẫn chưa thực sự được nhìn nhận đúng với tầm quan trọng của nó. Điều này thể hiện ở tỷ lệ kỹ sư kiểm thử phần mềm ở Việt Nam còn khá thấp, cứ 5 lập trình viên thì mới có 1 kỹ sư kiểm thử , trong khi tỷ lệ này theo chuẩn quốc tế là 3:1. Thêm vào đó, mức độ đáp ứng của kỹ sư kiểm thử phần mềm ở Việt Nam chưa cao. Nguyên nhân của việc này đến từ sự thiếu hụt các đơn vị đào tạo chuyên sâu về kiểm thử và nguyên nhân sâu xa vẫn là vấn đề kiểm thử phần mềm ở Việt Nam vẫn chưa được chuyên nghiệp hóa và đầu tư đúng mức.

Ngày nay, tự động hóa đang được nghiên cứu và ứng dụng trong nhiều lĩnh vực trong đó công nghệ phần mềm nói chung và kiểm thử phần mềm nói riêng cũng không ngoại lệ. 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 song song với kiểm thử truyền thống thủ công, sự ra đời của các công cụ hỗ trợ kiểm thử tự động như Quick Test Professional, Nunit, Junit, Load Runner ... là tất yếu. Selenium là một công cụ kiểm thử các ứng dụng web có khá nhiều ưu điểm như 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 như Junit, TestNG (với Java) hay Nunit (với C#), và ưu điểm đặc biệt của công cụ này là nó là một bộ mã nguồn mở, do đó các tổ chức sẽ không tốn kinh phí mua bản quyền. Tuy chưa được ứng dụng nhiều trong các tổ chức ở Việt Nam, song với Tìm hiểu về kiểm thử tự động và ứng dụng kiểm thử website sử dụng công cụ kiểm thử tự đông Selenium. Những̣ ưu điểm trên, Selenium hứa hẹn sẽ ngày càng phát triển và trở lên thông dụng hơn trong các tổ chức phát triển phần mềm ở nước ta.

## 1.2 Mục tiêu của đề tài

Đề tài hướng đến việc giúp người nghiên cứu nắm vững lý thuyết kiểm thử phần mềm, đặc biệt là kiểm thử tự động. Ngoài ra, đề tài còn tập trung vào việc tìm hiểu và sử dụng công cụ kiểm thử tự động Selenium. Cụ thể, công cụ Selenium IDE sẽ được ứng dụng để thực hiện kiểm thử giao diện website.

## 1.3 Giới hạn và phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài xoay quanh các nội dung lý thuyết về kiểm thử phần mềm, kiểm thử tự động và công cụ Selenium. Việc ứng dụng sẽ được giới hạn trong việc sử dụng Selenium IDE để kiểm thử giao diện của website, tập trung vào các chức năng cơ bản của công cụ.

## 1.4 Nội dung thực hiện

Nội dung của đề tài bao gồm việc trình bày lý thuyết cơ bản về kiểm thử tự động và kiểm thử website. Đồng thời, đề tài cũng cung cấp thông tin về công cụ Selenium và cách sử dụng nó. Cuối cùng, công cụ Selenium IDE sẽ được ứng dụng để kiểm thử giao diện website, minh họa quy trình kiểm thử tự động trong thực tế.

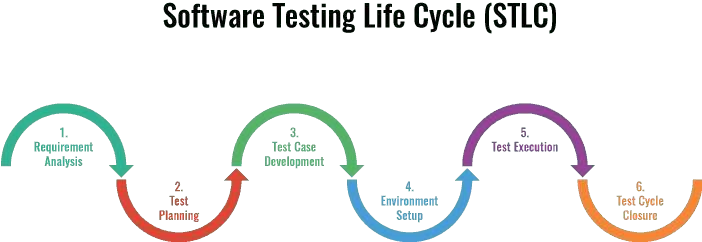
## 1.5 Phương pháp tiếp cận

Để thực hiện đề tài, các phương pháp nghiên cứu được áp dụng bao gồm: đọc tài liệu để thu thập thông tin, phân tích các mẫu kiểm thử để rút ra bài học, và thực nghiệm trực tiếp trên công cụ Selenium IDE để kiểm chứng lý thuyết. Các phương pháp này giúp đảm bảo tính chính xác và hiệu quả của kết quả nghiên cứu.

# PHẦN 2. KIẾN THỨC NỀN TẢNG

## 2.1 Quy trình kiểm thử phần mềm

Quy trình kiểm thử phần mềm xác định các giai đoạn, các pha trong kiểm thử phần mềm. Tuy nhiên thì Quy trình kiểm thử phần mềm – STLC không có tiêu chuẩn cố định nào, nhưng về cơ bản thì STLC bao gồm các giai đoạn sau:



Hình 2. 1 : Vòng đời kiểm thử phần mềm

Các giai đoạn kiểm thử được thực hiện một cách tuần tự. Mỗi giai đoạn sẽ có những mục tiêu khác nhau, đầu vào và kết quả đầu ra khác nhau nhưng mục đích cuối cùng vẫn là đảm bảo chất lượng sản phẩm phần mềm tốt nhất.

### 2.1.1 Phân tích yêu cầu

Đầu vào : Thu thập yêu cầu khách hàng thành một bản đặc tả cho phần mềm, phần mềm có những chức năng chính nào, các tiêu chí của khách hàng đối với đầu ra của phần mềm, tài liệu thiết kế cho hệ thống.

Hoạt động :

* Thu thập các yêu cầu của khách hàng đối với phần mềm.
* Xác định rõ chi tiết chức năng của hệ thống, phân tích chi tiết yêu cầu của khách hàng.
* Trong quá trình phân tích và xây dựng của phần mềm, nếu có vấn đề sẽ liên hệ với khách hàng để nắm rõ được chức năng của phần mềm.

Đầu ra : Đưa ra được nghiệp vụ của phần mềm, tài liệu báo cáo khả thi, phân tích rủi ro cho việc kiểm thử.

### 2.1.2 Lập kế hoạch

Đầu vào: Là các tài liệu liên quan đến nghiệp vụ của phần mềm, tài liệu báo cáo tính khả thi, phân tích rủi ro được phân tích ở trên.

Hoạt động: Xác định phạm vi của dự án: Thời gian để hoàn thành dự án, gồm những công việc gì cần thực hiện cho từng khoảng thời gian cụ thể để đưa ra lịch trình thực hiện cho từng công việc.

* Xác định phương pháp tiếp cận.
* Đưa ra những phương pháp và lên kế hoạch phù hợp nhất cho cả quá trình thực hiện kiểm thử sản phẩm.
* Xác định các nguồn lực của sản phẩm.
* Lên kế hoạch cho công việc kiểm thử.

Đầu ra: Đưa ra bản kế hoạch kiểm thử, các dự đoán về chi phí, con người, tài nguyên cân cho dự án và đưa ra tiến độ của dự án.

### 2.1.3 Thiết kế kiểm thử

Đầu vào: Các bản kế hoạch kiểm thử, các dự toán chi phí, con người, tài nguyên cần cho dự án và đưa ra tiến độ của dự án, các tài liệu đặc tả ban đầu.

Hoạt động:

* Đọc tài liệu: Đọc tất cả tài liệu để xác định rõ các chức năng phải kiểm thử với các luồng kiểm thử để đưa ra kịch bản kiểm thử đầy đủ nhất.
* Thiết kế các Testcase: Viết Testcase dựa trên kế hoạch kiểm thử và vận dụng các kỹ thuật thiết kế kịch bản kiểm thử. Test case cần bao phủ tất cả các trường hợp kiểm thử có thể xảy ra cũng như đáp ứng đầy đủ các tiêu chí của sản phẩm. Đồng thời đánh giá mức độ ưu tiên cho từng Test case.
* Chuẩn bị dữ liệu kiểm thử: Cùng với việc tạo ra các test case chi tiết, cũng cần chuẩn bị trước các dữ liệu kiểm thử cho các trường hợp cần thiết như test data, test script.
* Đánh giá Test case: Đánh giá các test case nhằm bổ sung, chỉnh sửa các sai sót trong thiết kế test case và rủi ro về sau.

Đầu ra: Các tài liệu gồm có: Test design, test plan, test case, check list, test data, test automation script.

### 2.1.4 Thiết lập môi trường kiểm thử

Đầu vào : Test plan, smoke test case, test data.

Hoạt động :

* Việc cài đặt môi trường kiểm thử là giai đoạn cũng rất quan trọng trong vòng đời phát triển phần mềm. Môi trường kiểm thử sẽ được quyết định dựa trên những yêu cầu của khách hàng, hay đặc thù của sản phẩm ví dụ như server/client/network, ...
* Tester cần chuẩn bị một vài test case để kiểm tra xem môi trường cài đặt đã sẵn sàng cho việc kiểm thử hay chưa. Đây chính là việc thực thi các smoke test case.

Đầu ra: Môi trường đã được cài đặt đúng theo yêu cầu, sẵn sàng cho việc kiểm thử và kết quả của smoke test case.

### 2.1.5 Thực thi kiểm thử

Đầu vào: Test plan, smoke test case, test data.

Hoạt động :

* Thực hiện các test case như thiết kế và mức độ ưu tiên đã đưa ra trên môi trường được cài đặt.
* So sánh với kết quả mong đợi sau báo cáo các bug xảy ra lên tool quản lý lỗi và theo dõi trạng thái của lỗi đến khi được sửa thành công.
* Thực hiện re-test để verify các bug đã được fix và regression test khi có sự thay đổi liên quan.
* Trong quá trình thực hiện kiểm thử, kiểm thử viên cũng có thể hỗ trợ, đề xuất cho cả đội dự án để có giải pháp hợp lý và kết hợp công việc hiệu quả.
* Phân tích tiến độ: kiểm thử viên cũng cần kiểm soát chặt chẽ tiến độ công việc của mình bằng cách so sánh tiến độ thực tế với kế hoạch, nếu chậm cần phải điều chỉnh sao cho kịp tiến độ dự án, nếu nhanh cũng cần điều chỉnh vì có thể test lead lên kế hoạch chưa sát với thực tế dự án. Từ đó có thể sửa chữa test plan để phù hợp với tiến độ dự án đưa ra.
* Báo cáo thường xuyên về tình hình thực hiện dự án: Cung cấp thông tin trong quá trình kiểm thử đã làm được những chức năng nào, còn chức năng nào, hoàn thành được bao nhiều phần trăm công việc, báo cáo các trường hợp phát sinh sớm, tránh ảnh hưởng tiến độ công việc của cả ngày.

Đầu ra: Kết quả kiểm thử, danh sách lỗi phát hiện được.

### 2.1.6 Kết thúc

Đầu vào: Tất cả các tài liệu liên quan đã được tổng hợp, ghi chép và hoàn thiện đầy đủ trong suốt quá trình kiểm thử của dự án: Tài liệu phân tích đặc tả yêu cầu, kế hoạch kiểm thử, kết quả kiểm thử, danh sách lỗi, …

Hoạt động :

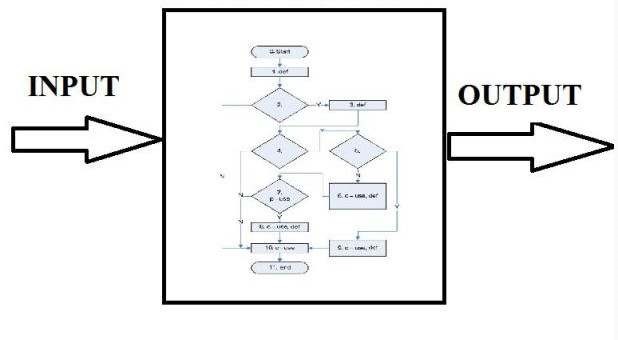
* Là giai đoạn cuối cùng trong quy trình kiểm thử phần mềm.
* tổng kết, báo cáo kết quả về việc thực thi test case, bao nhiêu case pass hoặc fail, bao nhiêu case đã được fix, mức độ nghiêm trọng của lỗi, bao nhiêu lỗi cao hoặc thấp, lỗi còn nhiều ở chức năng nào, dev nào nhiều lỗi. Chức năng nào đã hoàn thành test hoặc trễ tiến độ bàn giao.
* Đánh giá các tiêu chí hoàn thành như phạm vi kiểm tra, chất lượng, chi phí, thời gian, mục tiêu kinh doanh quan trọng.
* Thảo luận tất cả những điểm tốt, điểm chưa tốt và rút ra bài học kinh nghiệm cho những dự án sau, giúp cải thiện quy trình kiểm thử.

Đầu ra: Các tài liệu: Test report, test result

## 2.2 Kiểm thử hộp trắng

Kiểm thử Kiểm thử hộp trắng hay kiểm thử hướng logic cho phép khảo sát cấu trúc bên trong của chương trình. Kiểm thử viên sẽ truy cập vào cấu trúc dữ liệu và giải thuật bên trong chương trình (và cả mã lệnh thực hiện chúng).

Kiểm thử hộp trắng có thể được áp dụng tại mức độ test đơn vị - unit, tích hợp hệ thống- integration của quá trình kiểm thử phần mềm, thường được thực hiện ở mức đơn vị.



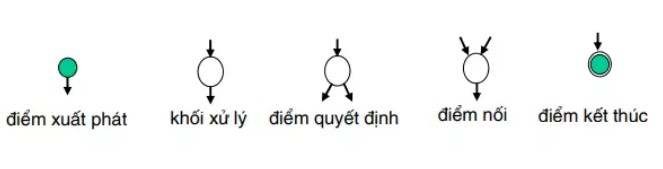
Hình 2. 2 Mô hình kiểm thử hộp trắng

### 2.2.1 Kiểm thử dòng điều kiện – Đồ thị dòng

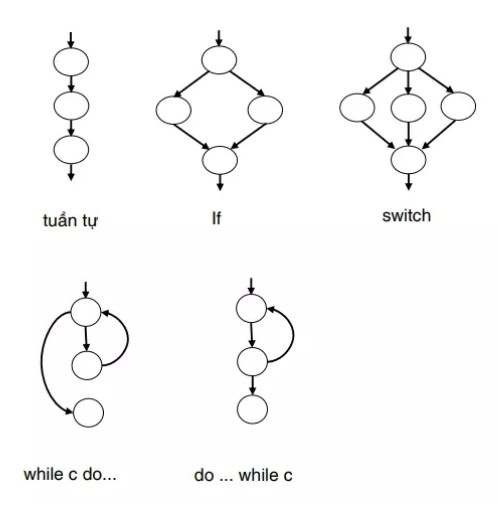
Là một kỹ thuật dùng trong kiểm thử hộp trắng được Tom McCabe đưa ra đầu tiên. Đồ thị dòng gần giống đồ thị luồng điều khiển của chương trình. Đây là một trong nhiều phương pháp mô tả giải thuật và là phương pháp trực quan cho ta thấy dễ dàng các thành phần của giải thuật và mối quan hệ trong việc thực hiện các thành phần này.

Kỹ thuật đường cơ bản – đồ thị dòng giúp những người thiết kế ca kiểm thử nhận được một độ phức tạp của một logic thủ tục.

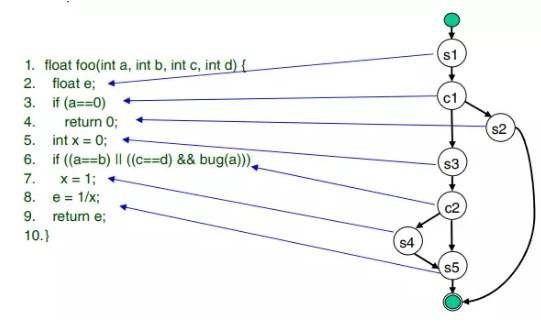
Gồm có 2 thành phần : Các nút và các cung kết nối giữa chúng.



Hình 2. 3 Các thành phần kiểm thử dòng điều kiện

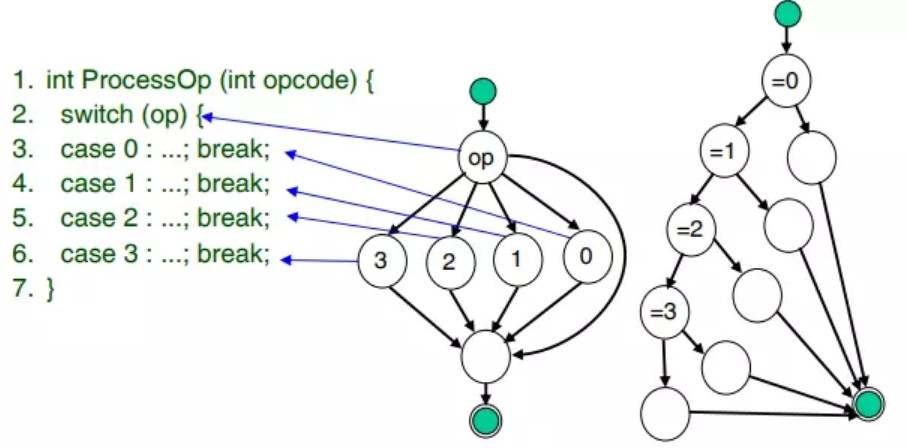


Hình 2. 4 Đồ thị của các cấu trúc lặp và rẽ nhánh



Hình 2. 5 Ví dụ về đồ thị dòng điều khiển

Nếu đồ thị dòng điều khiển chỉ chứa các nút quyết định nhị phân thì ta gọi nó là đồ thị dòng điều khiển nhị phân. Ta luôn có thể chi tiết hóa 1 đồ thị dòng điều khiển bất kỳ thành đồ thị dòng điều khiển nhị phân.



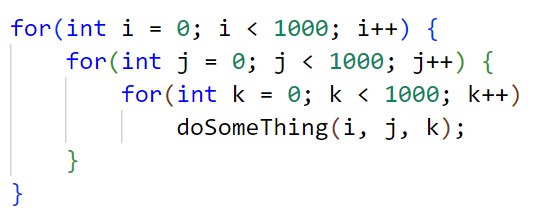
Hình 2. 6 Ví dụ về đồ thị dòng điều khiển

Độ phức tạp Cyclomatic C Độ phức tạp Cyclomatic C = V(G) của ₫ồ thị dòng điều khiển được tính bởi 1 trong các công thức sau: V(G) = E - N + 2, trong đó E là số cung, N là số nút của đồ thị. V(G) = P + 1, nếu là đồ thị dòng điều khiển nhị phân (chỉ chứa các nút quyết ₫ịnh luận lý - chỉ có 2 cung xuất True/False) và P số nút quyết định. Độ phức tạp Cyclomatic C chính là số đường thi hành tuyến tính độc lập của TPPM cần kiểm thử.

### 2.2.2 Kiểm thử dựa trên luồng điều kiện

Đường thi hành (Execution path): là 1 kịch bản thi hành đơn vị phần mềm tương ứng, cụ thể nó là danh sách có thứ tự các lệnh được thi hành ứng với 1 lần chạy cụ thể của đơn vị phần mềm, bắt đầu từ điểm nhập của đơn vị phần mềm đến điểm kết thúc của đơn vị phần mềm.

Mỗi TPPM có từ 1 đến n (có thể rất lớn) đường thi hành khác nhau. Mục tiêu của phương pháp kiểm thử luồng điều khiển là đảm bảo mọi đường thi hành của đơn vị phần mềm cần kiểm thử đều chạy đúng. Rất tiếc trong thực tế, công sức và thời gian để đạt mục tiêu trên đây là rất lớn, ngay cả trên những đơn vị phần mềm nhỏ.



Hình 2. 7 Ví dụ về dòng điều kiện

Trong ví dụ trên chỉ có một đường thi hành, nhưng rất dài (1 tỉ lần gọi hàm).

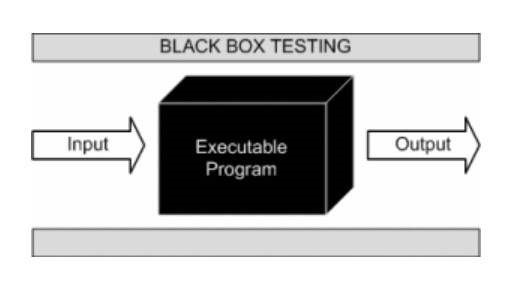
Ngoài 2 kỹ thuật kiểm thử trên thì còn có: Kiểm thử dựa trên luồng dữ liệu, kiểm thử đột biến, kiểm thử vòng lặp, kiểm thử memory leak, kiểm thử API.

## 2.3 Kiểm thử hộp đen

Kiểm thử hộp đem là một kỹ thuật để kiểm tra trong đó chức năng của ứng dụng được kiểm tra mà không cần xem cấu trúc mã bên trong, chi tiết triển khai và kiến thức về đường dẫn bên trong của phần mềm ra sao mà chỉ quan tâm đầu vào và đầu ra mong muốn chứ không quan tâm đến code bên trong phần mềm.

Kiểm thử hộp đen xem chương trình như là một “hộp đen”; bên trong mà người ta không thể nhìn thấy. Mục đích của kiểm thử viên là hoàn toàn không quan tâm về cách cư xử và cấu trúc bên trong của chương trình. Thay vào đó, tập trung vào tìm các trường hợp mà chương trình không thực hiện theo các đặc tả của nó. Phương pháp này cố gắng tìm ra các lỗi trong các loại sau:

* Chức năng không chính xác hoặc thiếu.
* Lỗi giao diện.
* Lỗi trong cấu trúc dữ liệu hoặc truy cập cơ sở dữ liệu bên ngoài.
* Hành vi hoặc hiệu suất lỗi.
* Khởi tạo và chấm dứt các lỗi.



Hình 2. 8 Kiểm thử hộp đen

Kiểm thử hộp đen chủ yếu là được thực hiện trong Function test và System test.

### 2.3.1 Đoán lỗi

Là một kỹ năng quan trọng của tester, thậm chí có thể gọi là nghệ thuật. Một kiệt tác của trực giác. Phương pháp này đặc biệt dựa vào kinh nghiệm và kiến thức của tester. Nhiều tester cố gắng đoán xem phần nào của hệ thống mà có khả năng ẩn chứa lỗi. Với phương pháp này, họ không cần một công cụ hay một kịch bản kiểm thử nào khi bắt đầu vào việc.

### 2.3.2 Kiểm thử dựa vào đồ thị nguyên nhân – kết quả

Cause Effect Graphing là một kỹ thuật thiết kế kiểm thử phần mềm liên quan đến việc xác định các trường hợp (điều kiện đầu vào) và các hiệu ứng (điều kiện đầu ra). Vì các hệ thống hiện nay đều được phát triển trên nền tảng OOP, do đó, chúng ta có thể có được một đồ thị các đối tượng mà hệ thống định nghĩa và kết nối. Từ đồ thị này, chúng ta dễ dàng biết các mối quan hệ của những đối tượng mà hệ thống xử lý, từ đó sẽ cho chúng ta các kịch bản kiểm thử phù hợp.

### 2.3.3 Phân vùng tương đương

Là một kỹ thuật kiểm thử phần mềm có liên quan đến phân chia các giá trị đầu vào thành các phân vùng hợp lệ và không hợp lệ, sau đó chúng ta sẽ viết ra các kịch bản kiểm thử cho từng phần, chọn giá trị đại diện từ mỗi phân vùng làm dữ liệu thử nghiệm.

Đây là kỹ thuật thực hiện test theo từng class đồng giá trị (tập hợp điều kiện cùng một thao tác).

Tập hợp giá trị input có cùng một kết quả xử lý, tập hợp thời gian có cùng một kết quả xử lý, tập hợp kết quả export được xử lý cùng một giá trị nhập.

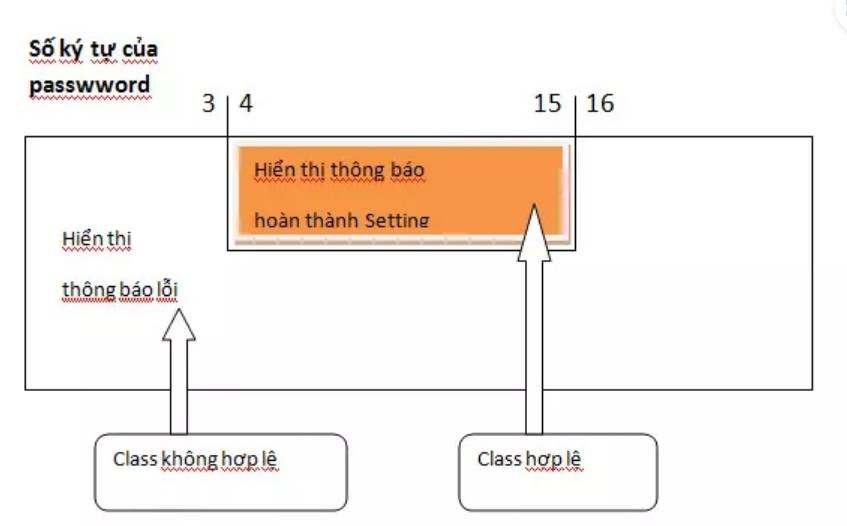
Mục đính là giảm đáng kể số lượng test case cần phải thiết kế vì với mỗi lớp tương đương ta chỉ cần test trên các phần tử đại diện. Chọn tối thiểu một giá trị đại diện từ các class đồng giá trị để tiến hành test. Nếu có lỗi xảy ra thì các giá trị khác trong class đồng giá trị cũng sẽ có lỗi giống nhau.

Ví dụ:

Spec yêu cầu nhập 4 <= password <= 15

1. Nhập đúng đưa ra mess hoàn thành thiết lập.
2. Nhập sai: mess yêu cầu nhập lại.

Như vậy ta sẽ thực hiện 2 testcase: một giá trị cho phần thông báo hoàn thành Setting và một giá trị phần thông báo lỗi.



Hình 2. 9 Phân vùng tương đương

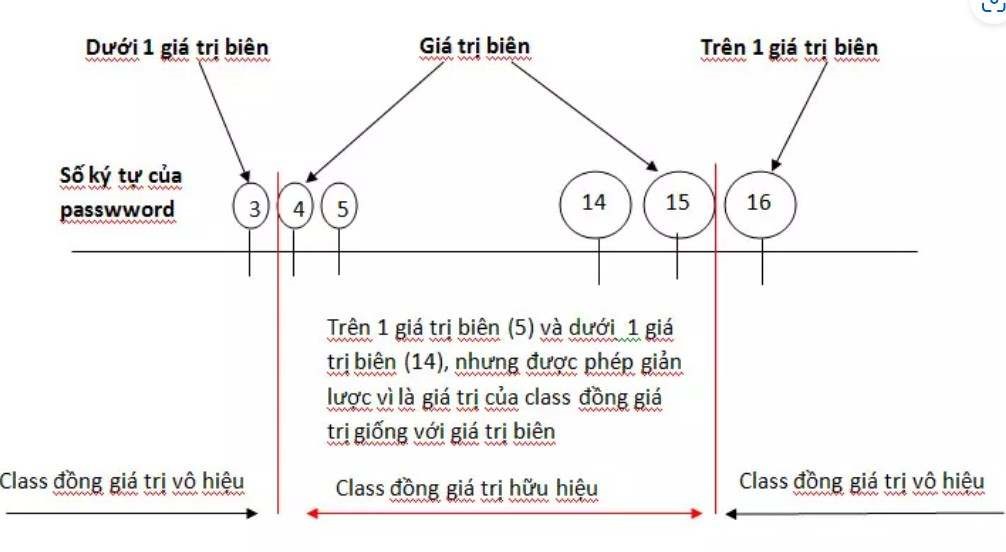
### 2.3.4 Phân tích giá trị biên

Boundary Value Analysis là một kỹ thuật kiểm thử phần mềm có liên quan đến việc xác định biên (ranh giới) của điều kiện mô tả cho các giá trị đầu vào và chọn giá trị ở biên và bên cạnh giá trị biên làm dữ liệu kiểm thử. Phương pháp phân tích giá trị biên sẽ đưa ra các giá trị đặc biệt, bao gồm loại dữ liệu, giá trị lỗi, bên trong, bên ngoài biên giá trị, lớn nhất và nhỏ nhất.

Những kỹ sư nhiều kinh nghiệm chắc chắn đã từng gặp phải các lỗi của hệ thống ngay tại giá trị biên. Đó là lý do tại sao phân tích giá trị biên lại quan trọng khi kiểm thử hệ thống.

Kiểm tra giá trị biên được thực hiện theo trình tự:

1. Tìm ra đường biên.
2. Quyết định giá trị biên.
3. Quyết định giá trị test.



Hình 2. 10 Phân tích giá trị biên

# PHẦN 3. KẾT QUẢ NGHIÊN CỨU

## 3.1 Tổng quan

### 3.1.1 Khái niệm và mục đích của kiểm thử phần mềm

#### 3.1.1.1 Khái niệm về kiểm thử phần mềm

Kiểm thử phần mềm là một quá trình xác định tính đúng đắn của phần mềm bằng cách xem xét tất cả các thuộc tính của nó (Độ tin cậy, Khả năng mở rộng, Tính khả chuyển, Khả năng sử dụng lại, Khả năng sử dụng) và đánh giá việc thực thi các thành phần phần mềm để tìm ra lỗi hoặc khiếm khuyết của phần mềm. Nói cách khác, đó là phương pháp kiểm tra xem sản phẩm phần mềm đó trên thực tế có phù hợp với các yêu cầu đã đặt ra hay không, và đảm bảo rằng không có lỗi hay khiếm khuyết. Nó bao gồm việc kiểm thử, phân tích, quan sát và đánh giá các khía cạnh khác nhau của sản phẩm. Người kiểm thử phần mềm (Tester) sử dụng kết hợp các công cụ thủ công và tự động. Sau khi tiến hành kiểm thử, Tester báo cáo kết quả cho team phát triển về các lỗi, khiếm khuyết hoặc các yêu cầu còn thiếu so với yêu cầu thực tế.

Kiểm thử phần mềm cần được thực hiện xuyên suốt quá trình xây dựng phần mềm để tìm ra các lỗi hoặc khiếm khuyết, từ đó có thể giải quyết được trước khi giao sản phẩm phần mềm. Sản phẩm phần mềm được kiểm thử chính xác đảm bảo độ tin cậy, bảo mật và hiệu suất cao, giúp tiết kiệm thời gian, hiệu quả chi phí và sự hài lòng của khách hàng. Một lý do nữa khiến việc kiểm thử ngày càng trở nên quan trọng đó là phát hiện khả năng tương thích với các thiết bị và nền tảng khác nhau.

Giả sử khi phát triển một trang web, Tester phải kiểm tra xem trang web có chạy trên độ phân giải thiết bị khác nhau, các trình duyệt khác nhau hay không? Những gì hoạt động tốt trên Chrome có thể không chạy tốt trên Safari hoặc Edge. Điều này làm phát sinh nhu cầu kiểm tra trình duyệt chéo, bao gồm kiểm tra tính tương thích của ứng dụng trên các trình duyệt khác nhau.

#### 3.1.1.2 Mục đích của việc kiểm thử phần mềm

Phát hiện càng nhiều lỗi càng tốt trong thời gian xác định trước khi bàn giao phần mềm tới tay khách hàng.

Chứng minh sản phẩm phần mềm phù hợp với các đặc tả của nó.

Đánh giá được chính xác chất lượng của sản phẩm phần mềm.

Giảm thiều chi phí bản trì sau khi bàn giao sản phẩn phần mềm tới khách hàng.

Đảm bảo rằng kết quả cuối cùng đáp ứng các yêu cầu kinh doanh và người sử dụng.

### 3.1.2 Sự quan trọng của kiểm thử phần mềm

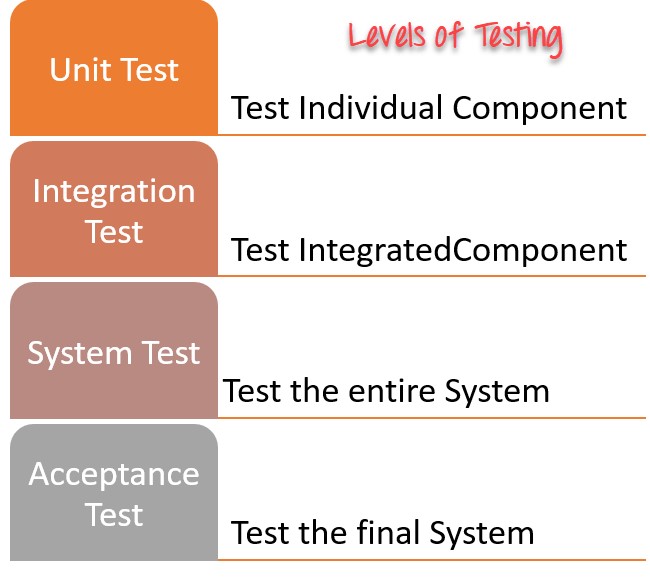
Kiểm thử rất quan trọng vì lỗi phần mềm có thể gây tốn kém hoặc thậm chí là nguy hiểm. Lỗi phần mềm có thể gây ra thiệt hại về tiền bạc và con người, và lịch sử có rất nhiều những ví dụ như vậy. Dưới đây là một số ví dụ cụ thể:

* Sự cố tàu thăm dò Mariner I (28/6/1962): Tàu thăm dò vũ trụ Mariner I có sứ mệnh bay đến sao Kim nhưng đã bị phá hủy 293 giây sau khi phóng do bay chệch hướng so với dự kiến ban đầu. Những cuộc điều tra sau đó đã khám phá ra rằng một công thức được viết trên giấy bằng bút chì đã không được chuyển đổi sang mã điện toán, khiến hệ thống máy tính tính toán sai đường đi của tên lửa.
* Sự cố máy gia tốc Therac-20 (1985-1987): Các chuyên thiết kế thuộc hãng AECL (Canada) đã thay thế khóa an toàn cơ học trong máy gia tốc Therac-20 bằng một phần mềm kiểm soát nhằm tăng độ tin cậy cho máy gia tốc Therac-25. Sự cố xảy ra khi nguồn năng lượng cao được kích hoạt và bia kim loại chưa xoay vào đúng vị trí nhưng phần mềm không phát hiện ra. Dòng điện cực cao được bắn thẳng vào bệnh nhân, gây cảm giác shock điện, phát nhiệt và cháy bức xạ. Có ít nhất 5 người chết và nhiều người bị thương nặng.
* Lỗi tràn bộ đệm trong Berkeley Unix năm 1988: Morris, sâu Internet đầu tiên, đã lây lan tới khoảng 2.000 - 6.000 máy tính của nhiều tổ chức ở Mỹ trong vòng chưa đầy một ngày bằng cách tận dụng lỗi tràn bộ nhớ trung gian. Mã lập trình get() được thiết kế để nhận dòng text qua mạng nhưng nó không được dự tính trước để hạn chế khối lượng nhập. Tận dụng khiếm khuyết này, một lượng lớn dữ liệu đã được gửi tới, tạo điều kiện cho phép sâu kiểm soát bất cứ hệ thống nào nó có thể kết nối. Các chuyên gia đã quyết định loại get() trong mã. Tuy nhiên, họ từ chối xóa get() trong thư viện nhập/xuất chuẩn của ngôn ngữ lập trình C và nó vẫn còn tồn tại đến ngày nay
* Ariane 5 Flight 501 năm 1996: Mã hoạt động cho tên lửa Ariane 4 được sử dụng lại trong phiên bản tiếp theo Ariane 5. Tuy nhiên, rắc rối đã xảy ra khi mã này thực hiện quá trình chuyển đổi số chứa dấu phấy động 64 bit sang ký hiệu số nguyên 16 bit. Động cơ trong Ariane 5 có tốc độ nhanh hơn đã khiến các số 64-bit trở lên lớn hơn so với Ariane 4, gây tình trạng quá tải và sập máy tính điều khiển. Hệ thống backup cũng gặp trục trặc ngay 0,05 giây sau đó và thiết bị xử lý cơ sở của tên lửa bị tiếp quá nhiều năng lượng.
* Phép chia dấu phẩy động (floating point) Intel Pentium năm 1993: Một lỗi silicon khiến chip Pentium nổi tiếng của hãng sản xuất thiết bị xử lý Mỹ gặp sai sót khi chia số floating-point trong một chuỗi cụ thể. Ví dụ, chia 4195835.0/3145727.0 sẽ ra 1.33374 thay vì 1.33382, sai số 0,0006%. Dù lỗi ảnh hưởng đến rất ít người sử dụng, nó trở thành một cơn ác mộng về PR (quan hệ cộng đồng) cho Intel. Đã có khoảng 3 - 5 triệu chip gặp khiếm khuyết lưu thông trên thị trường. Intel ban đầu chỉ cung cấp thiết bị xử lý mới cho những khách hàng chứng minh được rằng công việc của họ đòi hỏi độ chính xác cao. Tuy nhiên, công ty sau đó đã nhún nhường và chấp nhận thay thế chip cho bất cứ ai gửi phàn nàn. Tổng thiệt hại của Intel cho lỗi này là 475 triệu USD.

### 3.1.3 Các giai đoạn kiểm thử phần mềm

Các bài kiểm tra được nhóm lại với nhau dựa trên vị trí chúng được thêm vào trong SDLC hoặc theo mức độ chi tiết của chúng. Nói chung, có bốn cấp độ kiểm thử: kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử hệ thống và kiểm thử chấp nhận. Mục đích của Cấp độ kiểm thử là làm cho việc kiểm thử phần mềm có hệ thống và dễ dàng xác định tất cả các trường hợp kiểm thử có thể có ở một cấp độ cụ thể.

* Kiểm thử đơn vị: Kiểm tra xem các thành phần phần mềm có đáp ứng đầy đủ các chức năng hay không.
* Kiểm thử tích hợp: Kiểm tra luồng dữ liệu từ một mô-đun này sang các mô-đun khác.
* Kiểm thử hệ thống: Đánh giá cả nhu cầu chức năng và phi chức năng cho việc kiểm thử.
* Kiểm thử chấp nhận người dùng cuối: Đánh giá cả nhu cầu chức năng và phi chức năng cho việc kiểm thử.



Hình 3. 1 Các cấp độ kiểm thử phần mềm

Có nhiều cấp độ kiểm tra khác nhau giúp kiểm tra hành vi và hiệu suất để kiểm thử phần mềm. Các cấp độ kiểm tra này được thiết kế để nhận ra các khu vực còn thiếu và điều hòa giữa các trạng thái của vòng đời phát triển. Trong các mô hình SDLC có các giai đoạn đặc trưng như thu thập yêu cầu, phân tích, thiết kế, mã hóa hoặc thực thi, thử nghiệm và triển khai. Tất cả các giai đoạn này đều trải qua quá trình cấp kiểm thử phần mềm.

#### 3.1.3.1 Kiểm thử đơn vị (Unit tesing)

Kiểm thử đơn vị là loại kiểm thử phần mềm trong đó các đơn vị/thành phần đơn lẻ của phần mềm sẽ được kiểm tra như: Hàm (Function), Lớp (Class), Phương thức (Method) và người thực hiện kiểm tra thường là các developer. Kiểm thử đơn vị được thực hiện trong quá trình phát triển ứng dụng. Lỗi ở level này thường được fix ngay sau khi chúng được tìm ra mà không cần lưu lại và quản lý như các test level khác.

Mục đích:

* Lỗi dễ được tìm thấy và được sửa sớm trong chu trình phát triển phần mềm vì vậy tiết kiệm thời gian và chi phí sửa lỗi.
* Tách riêng từng phần để kiểm tra và chứng minh các thành phần đó thực hiện chính xác các yêu cầu chức năng trong đặc tả.
* Mã code được tái sử dụng nhiều hơn.
* Giúp cho việc thay đổi và bảo trì trở nên dễ dàng hơn.
* Ngăn chăn lỗi xuất hiện trong các giai đoạn kiểm thử sau.̣
* Phương pháp sử dụng: Kiểm thử hộp trắng.

#### 3.1.3.2 Kiểm thử tích hợp (Integration Testing)

Kiểm thử tích hợp là loại kiểm thử trong đó các module phần mềm hay từng chức năng riêng lẻ được tích hợp logic và được kiểm tra theo nhóm chung với nhau. Mỗi dự án phần mềm gồm nhiều module, được code bởi nhiều người khác nhau, vì vậy kiểm thử tích hợp tập chung vào việc kiểm tra tương tác, truyền dữ liệu giữa các module hoặc đơn vị được tích hợp

Mục đích: Phát hiện lỗi tương tác xảy ra giữa các module. Tập chung chủ yếu vào các giao diện và thông tin giữa các module. Tích hợp các module đơn lẻ thành các hệ thống nhỏ.

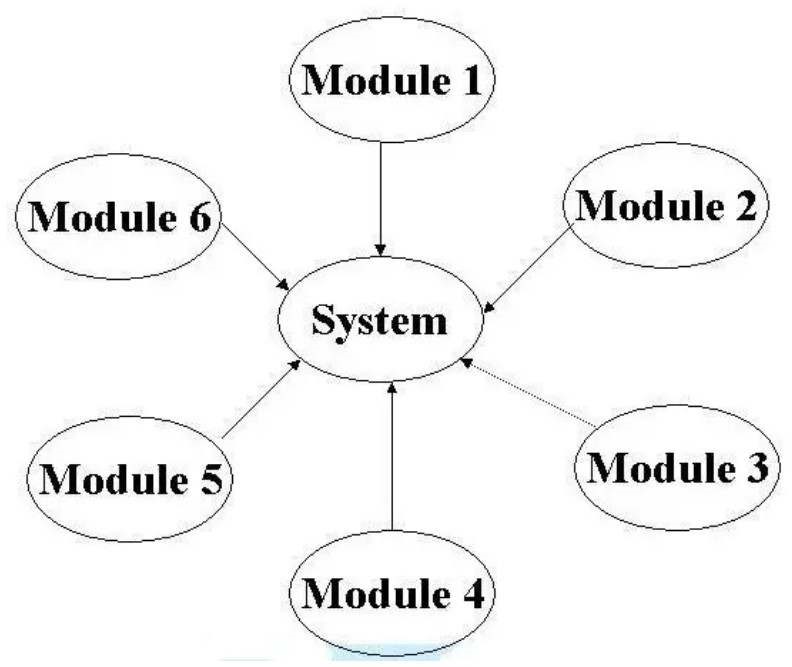
Một số phương pháp kiểm thử tích hợp:

1. Big Bang

Đây là phương pháp test tích hợp mà tất cả hoặc hầu hết các unit được kết hợp với nhau và cùng được kiểm thử. Phương pháp này được thực hiện khi team kiểm thử nhận được toàn bộ phần mềm.

Ưu điểm: phù hợp với các dự án nhỏ

Nhược điểm: có thể bỏ qua các bug giao diện nhỏ trong quá trình tìm bug, mất thời gian cho tích hợp hệ thống nên không có thời gian cho test



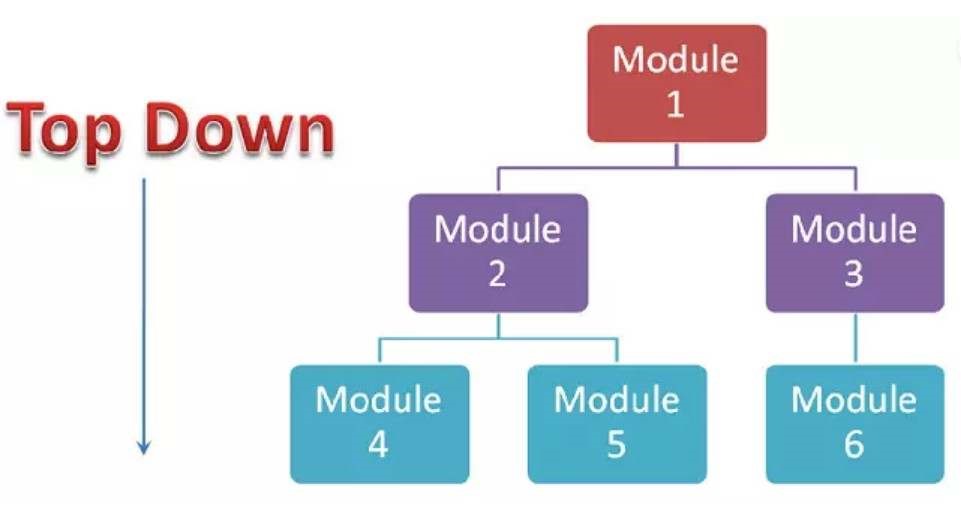
Hình 3. 2 Phương pháp Big Bang

1. Top down

Kiểm thử được thực hiện từ trên xuống dưới. Đơn vị cao nhất được kiểm thử đầu tiền, đơn vị thấp hơn được kiểm thử sau đó một cách tuần tự.

Ưu điểm: thu gọn bug dễ dàng hơn, module quan trọng được kiểm thử đầu tiên lỗi trong thiết kế lớn có thể được tìm thấy và cố định đầu tiên.

Nhược điểm: module ở mức độ thấp hơn sẽ được kiểm tra không đầy đủ.



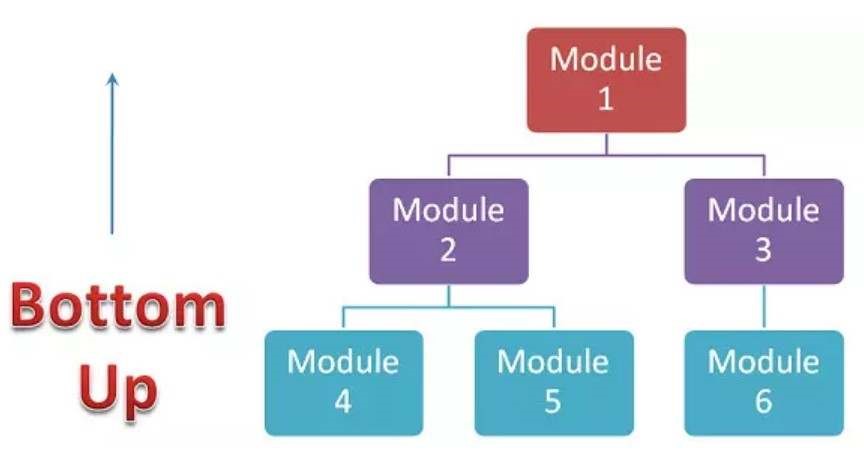
Hình 3. 3 Phương pháp Top Down.

1. Bottom up

Kiểm thử được thực hiện từ dưới lên trên. Đơn vị thấp nhất được kiểm thử đầu tiền, đơn vị cao hơn được kiểm thử sau đó.

Ưu điểm: thu gọn bug dễ dàng hơn, không mất thời gian để đợi các module được tích hợp.

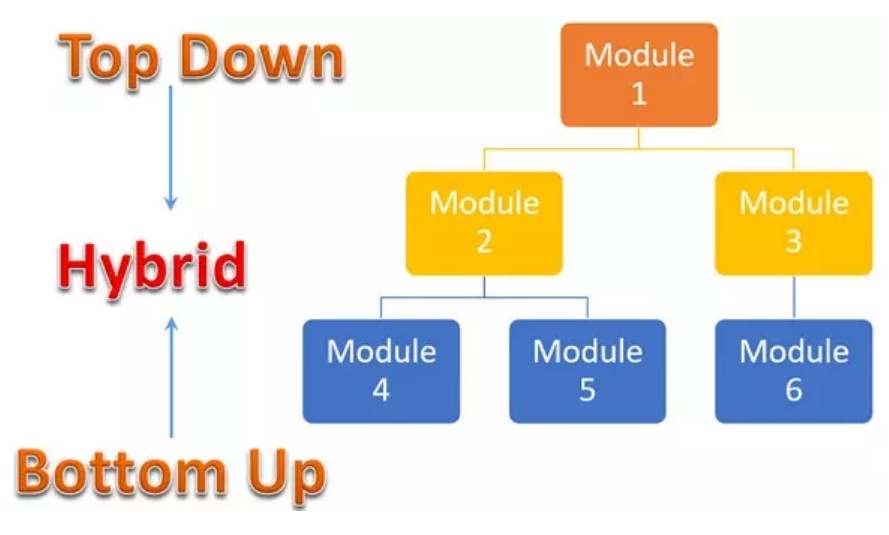
Nhược điểm: module quan trọng của hệ thống có thể dễ bị lỗi, không giữ được nguyên mẫu ban đầu của hệ thống.



Hình 3. 4 Phương pháp Bottom Up.

1. Sandwich

Còn gọi Phương thức gia tăng chức năng. Là sự kết hợp của hai phương pháp Top Down và Bottom Up. Ở đây, các module hàng đầu được kiểm tra với các module thấp hơn đồng thời các module thấp hơn được tích hợp với các module hàng đầu và được kiểm thử.



Hình 3. 5 Phương pháp Sandwich/Hybrid.

#### 3.1.3.3 Kiểm thử hệ thống (System Testing)

Kiểm thử hệ thống là thực hiện kiểm thử một hệ thống đã được tích hợp hoàn chỉnh để đảm bảo nó hoạt động đúng yêu cầu.

Mục đích: Đánh giá hệ thống có đáp ứng theo đúng yêu cầu nghiệp vụ, yêu cầu về chức năng theo bản đặc tả yêu cầu phần mềm đưa ra hay không.

Các loại kiểm thử hệ thống:

* Kiểm tra chức năng - Functional Testing: Kiểm thử chức năng nhằm đảm bảo chức năng phần mềm được vận hành theo đúng mục đích đưa ra.
* Kiểm tra khả năng phục hồi - Recoverability Testing: Nó được thực hiện bằng cách cố làm cho phần mềm bị crash hoặc fail, để đánh giá khả năng phục hồi của sản phẩm một cách nhanh chóng.
* Kiểm tra khả năng tương tác - Interoperability Testing: Nó đảm bảo khả năng phần mềm tương thích và tương tác với phần mềm hoặc hệ thống khác và các thành phần của chúng.
* Kiểm tra năng suất - Performance Testing: Nó được thực hiện để kiểm tra phản ứng, độ ổn định, khả năng mở rộng, độ tin cậy và các số liệu chất lượng khác của phần mềm dưới các khối lượng công việc khác nhau.
* Kiểm tra khả năng mở rộng - Scalability Testing: Để đảm bảo các khả năng mở rộng quy mô của hệ thống theo các thuật ngữ khác nhau như chia tỷ lệ người dùng, chia tỷ lệ theo địa lý và mở rộng tài nguyên.
* Kiểm tra độ tin cậy - Reliability Testing: Đảm bảo hệ thống có thể được vận hành trong thời gian dài hơn mà không phát triển lỗi.
* Kiểm tra hồi quy - Regression Testing: Đảm bảo hệ thống ổn định khi hệ thống tích hợp các hệ thống con và nhiệm vụ bảo trì khác nhau.
* Kiểm tra bảo mật - Security Testing: Để đảm bảo rằng hệ thống không cho phép truy cập trái phép vào dữ liệu và tài nguyên.
* Kiểm thử khả năng sử dụng - Usability Testing: kiểm thử khả năng sử dụng chủ yếu tập trung vào việc người dùng dễ dàng sử dụng ứng dụng, linh hoạt trong việc kiểm soát xử lý và khả năng của hệ thống để đáp ứng các mục tiêu.
* Người thực hiện: thường là khách hàng hoặc bên thứ ba. Người thực hiện: Tester.

#### 3.1.3.4 Kiểm thử chấp nhận (Acceptance Testing)

Kiểm thử chấp nhận chính thức liên quan đến yêu cầu và quy trình kinh doanh để xác định liệu hệ thống có đáp ứng tiêu chí chấp nhận hay không và cho phép người dùng, khách hàng hoặc tổ chức được ủy quyền khác xác định có chấp nhận hệ thống hay không.

Mục đích: Để nghiệm thu hệ thống trước khi hệ thống được đưa vào hoạt động.

Kiểm thử chấp nhận thường là trách nhiệm của người dùng hoặc khách hàng. Trong kiểm thử hệ thống, khách hàng sẽ kiểm tra xem phần mềm được viết có hoạt động đúng như mong đợi của mình không, có đảm bảo tính tiện dụng, hiệu suất hoạt động có như mong đợi không, có bảo mật tốt hay không….

Tìm lỗi không phải là trọng tâm chính trong kiểm thử chấp nhận, vì việc tìm lỗi đã được đội Developer và Tester thực hiện trong các giai đoạn kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử hệ thống rồi.

Kiểm thử chấp nhận được chia thành 2 loại:

* Kiểm thử alpha: được thực hiện tại nơi phát triển phần mềm bởi những người trong tổ chức nhưng không tham gia phát triển phần mềm.
* Kiểm thử beta: được thực hiện tại bởi khách hàng/ người dùng cuối tại địa điểm của người dùng cuối.

Sử dụng phương pháp: Kiểm thử hộp đen.

Người thực hiện: thường là khách hàng hoặc bên thứ ba.

### 3.1.4 Các phương pháp kiểm thử phần mềm

#### 3.1.4.1 Kiểm thử tĩnh – Static testing:

Quy về việc kiểm tra một số thứ mà nó không phải đang chạy, đang kiểm tra, đang xem lại. Đây là phương pháp kiểm thử phần mềm cần phải duyệt lại các yêu cầu và các đặc tả bằng tay bằng cách sử dụng giấy, bút để kiểm tra logic, các chi tiết mà không cần chạy chương trình. Cũng có thể tự động hóa kiểm thử tĩnh bằng cách thực hiện kiểm tra toàn bộ các chương trình được phân tích bằng cách sử dụng một trình thông dịch hoặc biên dịch để xác nhận tính hợp lệ của chương trình.

#### 3.1.4.2 Kiểm thử động – Dynamic testing:

Là quá trình kiểm thử, chạy và sử dụng phần mềm. Dùng máy để kiểm tra và chạy chương trình để kiểm tra lỗi của chương trình. Kiểm tra các hoạt động của câu lệnh.Chúng ta tiến hành biên dịch và chạy chương trình sau đó nhập các giá trị đầu vào và kiểm tra xem dữ liệu đầu ra có đúng không. Có bốn phương pháp kiểm thử động đã được nêu ở trên đó là:

* Kiểm thử đơn vị: Unit Tests
* Kiểm thử tích hợp: Intergration Tests
* Kiểm thử hệ thống: System Tests
* Kiểm thử chấp nhận sản phẩm: Acceptance Tests.

## 3.2 TÌM HIỂU CÔNG CỤ KIỂM THỬ PHẦN NỀM

### 3.2.1 Giới thiệu công cụ

Selenium (thường được viết tắt là SE) là một phần mềm mã nguồn mở, được phát triển bởiJason Huggins, sau đó được tiếp tục phát triển bởi nhóm ThoughtWorks vào năm 2004.

Selenium là một bộ các công cụ hỗ trợ kiểm thử tự động các tính năng của ứng dụng web, bao gồm 4 phần: Selenium IDE, Selenium Remote Control (RC), Selenium Core và Selenium Grid.

Selenium hỗ trợ kiểm thử trên hầu hết các trình duyệt web phổ biến hiện nay như Firefox, Internet Explorer, Googlechrome và hỗ trợ trên rất nhiều ngôn ngữ lập trình phổ biến như C#, Java, Python, PHP. Không những vậy, Selenium còn có thể kết hợp với một số công cụ kiểm thử khác như Junit, Bromien, Nunit.

### 3.2.2 Đặc điểm

Một số đặc điểm của Selenium IDE.

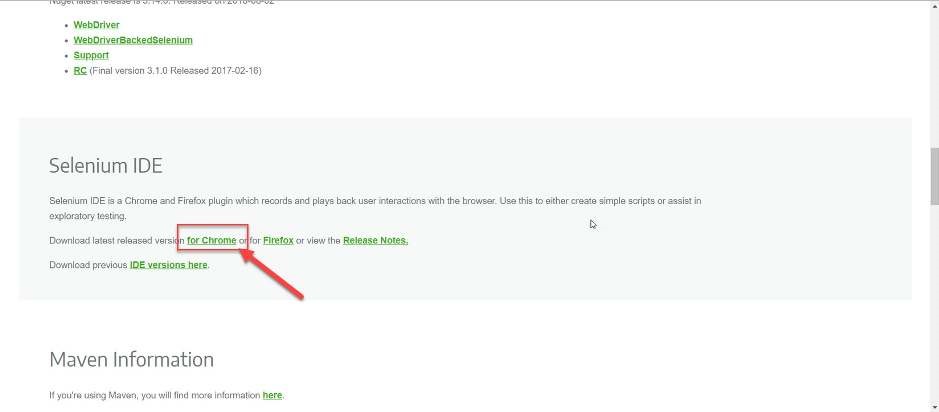
* Công cụ mã nguồn mở
* Phiên bản: Selenium IDE 3.17.4
* Công cụ hỗ trợ các loại kiểm thử giao diện của các website, hỗ trợ trên rất nhiều ngôn ngữ lập trình phổ biến như C#, Java, Python, PHP.
* Tương thích với các trình duyệt web phổ biến hiện tại, có thể cài đặt dễ dàng dưới dạng Extensions của trình duyệt đó.

### 3.2.3 Cài đặt và sử dụng công cụ Selenium IDE

#### 3.2.3.1 Cách cài đặt

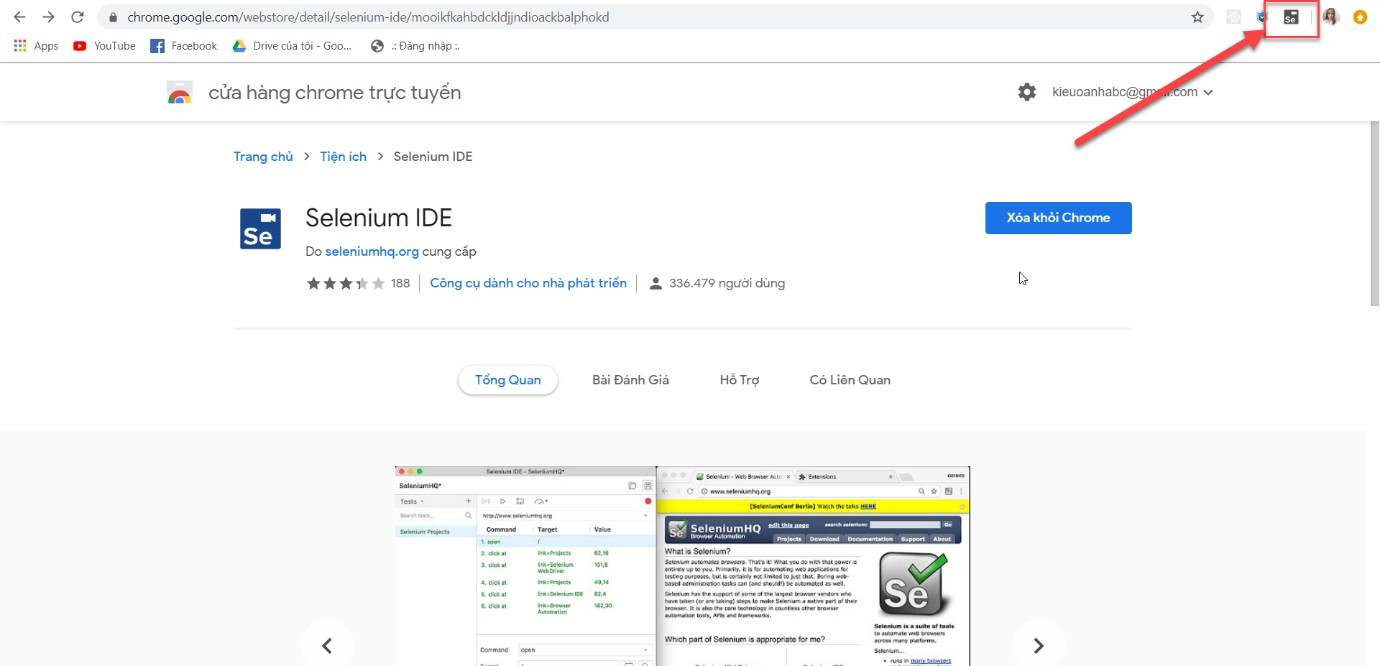
Bước 1: Truy cập vào trang web https://selenium.dev/downloads/ để download Selenium IDE.

Bước 2: Chọn phiên bản cần download.



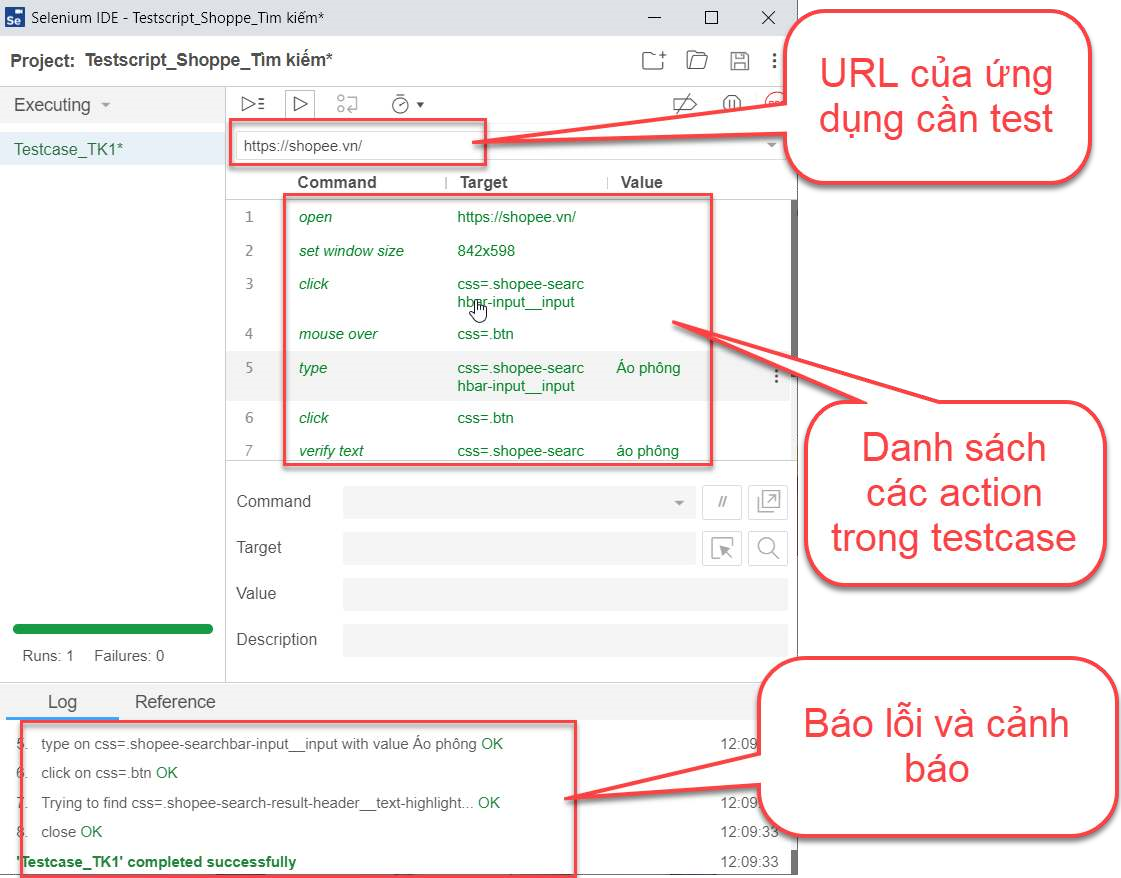
Hình 3.6 Download Selenium IDE

Bước 3: Sau khi chọn phiên bản cần download sau, màn hình sẽ được chuyển tới cửa hàng Chrome trực tuyến. Chọn “Thêm vào chrome” để cài đặt. Sau khi cài đặt sau phần mềm sẽ xuất hiện ở góc bên trái màn hình, như vậy là ta đã cài đặt được thành công công cụ Selenium IDE.



Hình 3.7 Kết quả cài đặt

#### 3.2.3.2 Cách sử dụng

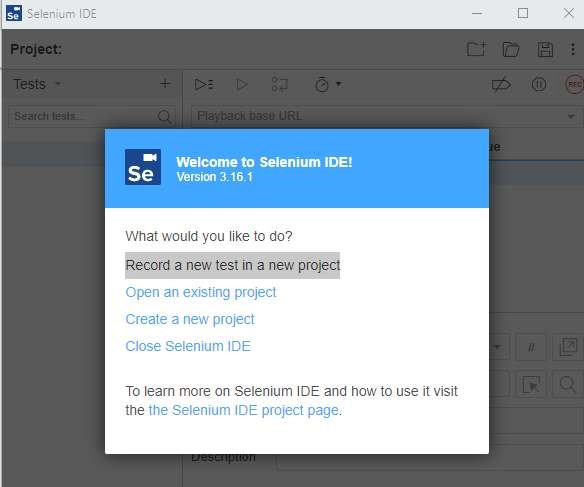


Hình 3. 8 Các thành phần của Selenium IDE.

#### 3.2.3.4 Thực hiện một kịch bản test case với Selenium IDE

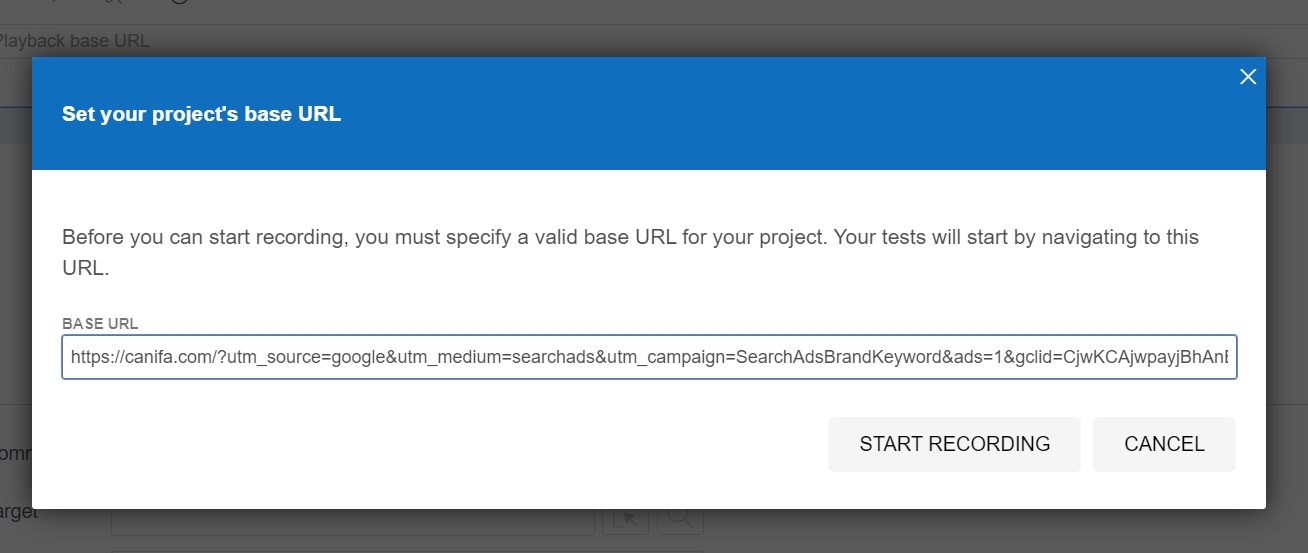
Bước 1: Vào một trình duyệt mà người dùng muốn thực hiện để kiểm thử trang web nhấn vào biểu tượng Selenium ở góc phải màn hình. Ở đây tôi sử dụng trình duyệt Chrome.

Bước 2: Chọn Record a new test in a new project và đặt tên cho project.



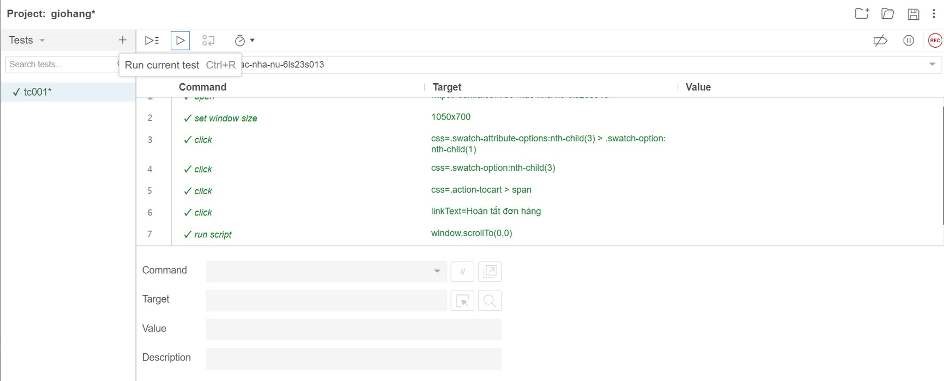
Hình 3. 9 Tạo Project mới.

Bước 3: Thay đổi mục Based URL thành URL của ứng dụng cần kiểm thử. Lấy ví dụ ứng dụng web cần kiểm thử có url là Canifa.com



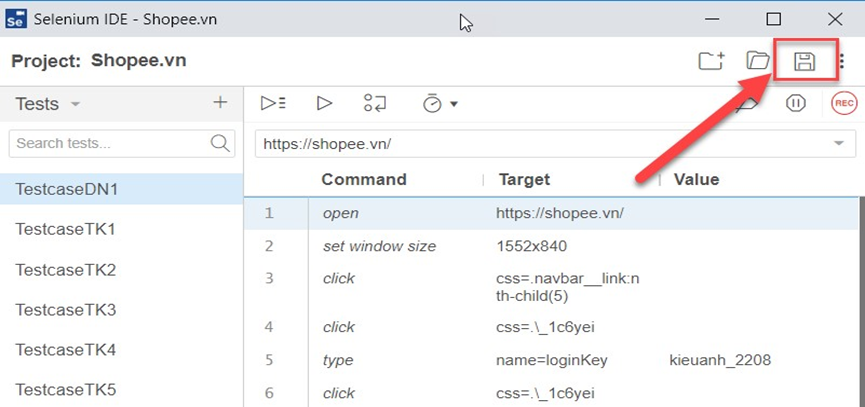
Hình 3. 10 Nhập URL của trang web.

Bước 4: Click vào nút thu “REC” và tiến hành các thao tác cần kiểm thử trên website canifa.com.



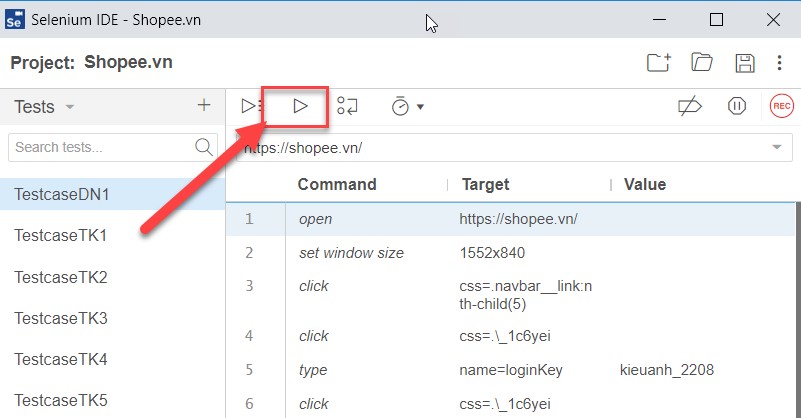
Hình 3. 11 Tiến hành record test case.

Bước 5: Lưu test case



Hình 3.12 Lưu test case

Bước 6: Để chạy thử test case đã được lưu, ta click vào nut Play trên màn hình Selenium.



Hình 3. 13 Chạy test case.

## 3.3 GIỚI THIỆU TRANG WEB CANIFA

### 3.3.1 Giới thiệu

Website bán bánh Canifa.vn trực tuyến là một website được xây dựng để phục vụ nhu cầu bán quần áo online của cửa hàng Canifa. Đến năm 2001 thì lúc này thương hiệu thời trang Canifa chính thức được ra đời. Đây được xem là cột mốc đáng nhớ và tự hào của công ty này trong ngành thời trang Việt Nam.

Canifa hiện là một trong những thương hiệu thời trang chất lượng được rất nhiều người tiêu dùng Việt Nam biết đến. Các sản phẩm của Canifa có độ phủ hầu hết đối tượng với độ tuổi từ 2-5 tuổi. Vì vậy, mẫu mã và chất liệu sản phẩm của Canifa luôn đa dạng để có thể đáp ứng đầy đủ nhu cầu khác nhau từ các khách hàng. Bên cạnh đó, các thiết kế trong sản phẩm của thương hiệu này hướng tới sự tiện dụng và bền bỉ hơn là theo xu hướng thời trang thay đổi hàng ngày. Mang đến niềm vui cho hàng triệu gia đình Việt.

Hiện nay, mọi người đều hướng đến cuộc sống năng động tích cực, Canifa mong muốn đồng hành cùng người tiêu dùng Việt trong hành trình sống đó. Canifa cam kết mang đến những sản phẩm đạt chất lượng cao nhất như một lời tri ân đối với sự yêu mến và tin dùng.

### 3.3.2 Chức năng phần mềm

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trải nghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

## 3.4 Kiểm thử trang website Canifa

### 3.4.1 Lập kế hoạch kiểm thử (test plan)

#### Chiến lược kiểm thử:

1. Kiểm thử chức năng (Functional Testing)

##### Bảng 3.1 Kiểm thử chức năng

|  |  |
| --- | --- |
| **Mục đích test** | Đảm bảo mục tiêu test đúng đắn của chức năng, bao gồm định hướng, dữ liệu đầu vào, xử lý và dữ liệu nhận được |
| **Cách thực hiện** | Thực hiện kiểm thử các chức năng, sử dụng dữ liệu hợp lệ và không hợp lệ để kiểm tra:   * Admin: đăng nhập thành công có thể xem thông tin, sửa, xóa, update tin tức và sản phẩm, đổi mật khẩu. * Khánh hàng: đăng nhập thành công có thể xem thông tin cá nhân, xem thông tin, tin tức sản phẩm, đặt hàng và đặt mua sản phẩm, đổi mật khẩu.   Thông báo hiển thị khi dữ liệu không hợp lệ: đăng nhập không thành công, hệ thống yêu cầu nhập lại. |
| **Điều kiện hoàn thành** | Toàn bộ kế hoạch kiểm thử đã được thực hiện. - Toàn bộ các lỗi phát hiện ra đã được ghi nhận. |
| **Các vấn đề đặc biệt:** |  |

1. Kiểm thử giao diện người dùng (User Interface Testing)

##### Bảng 3.2 Kiểm thử giao diện người dùng.

|  |  |
| --- | --- |
| **Mục đích kiểm thử** | Kiểm tra việc sử dụng thông qua mục tiêu test phản ánh đúng các chức năng và yêu cầu nghiệp vụ, bao gồm màn hình đến màn hình, trường dữ liệu đến trường dữ liệu và sử dụng các phương pháp truy cập (phím tabs, di chuột, tổ hợp phím) - Các đối tượng và thuộc tính màn hình như menus, size, position, state, và tập trung vào việc tương thích với chuẩn. |
| **Cách thực hiện** | Tạo ra các ca kiểm thử cho mỗi màn hình để kiểm tra việc sử dụng đúng cách và tình trạng các đối tượng cho mỗi màn hình và đối tượng của ứng dụng  Kiểm thử giao diện lúc khởi tạo. Lúc này ta test được giá trị mặc định, kích thước, màu sắc...của các đối tượng. Kiểm tra xem giá trị trong các đối tượng đó được đúng với bản yêu cầu đặc tả của khách hàng không. Trong thực tế có nhiều đối tượng không đúng với đặc tả, cần phải:   * Kiểm tra màu nền, chữ của toàn bộ đối tượng * Kiểm tra màu chữ, font, font size của tất cả các dòng văn bản đặt trong textbox   Thao tác đối với từng giao diện cho từng chức năng củaphần mềm đối với hệ thộng và khách hàng |
| **Điều kiện hoàn thành** | Mỗi màn hình được kiểm tra thành công đúng với phiên bản kiểm tra hoặc phạm vi chấp nhận được, không có sự sai lệch |
| **Các vấn đề đặc biệt** | Không phải toàn bộ các thuộc tính của các đối tượng đều truy cập được |

1. Giai đoạn kiểm thử

##### Bảng 3.3 Giai đoạn kiểm thử.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kiểu kiểm thử** |  | **Giai đoạn kiểm thử** | | |
| **Đơn vị** | **Tích hợp** | **Hệ thống** | **Chấp nhận** |
| Kiểm thử chức năng |  |  | X |  |
| Độ tin cậy |  |  | X |  |

1. Công cụ kiểm thử

##### Bảng 3.4 Các công cụ sử dụng.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mục đích** | **Công cụ** | **Nhà cung cấp/Tự xây dựng** | **Phiên bản** |
| Kế hoạch kiểm thử | MS Word | Microsoft | 2016 |
| Báo cáo kiểm thử | MS Word | Microsoft | 2016 |
| Kiểm thử giao diện | Selenium  IDE | ThoughtWorks | 3.17.2 |

1. Môi trường kiểm thử

Quá trình kiểm thử thực hiện trên môi trường Windows - Trình duyệt được sử dụng là Google Chrome.

### 3.4.2 Kiểm thử tìm kiếm tin tức

#### 3.4.2.1 Phân tích thiết kế kiểm thử

1. Giao diện chức năng

Thanh tìm kiếm tin tức trên website bao gồm một ô tìm kiếm và nút tìm kiếm.

Kết quả tìm kiếm chứa danh sách các tin tức, thông tin liên quan đến từ khóa tìm kiếm, thông tin sản phẩm.

Kết quả cụ thể được hiển thị ở trang chi tiết của nó khi bấm vào sản phẩm đó.

1. Mô tả giao diện và các điều kiện ràng buộc

Ô tìm kiếm: nhập từ khóa tìm kiếm.

Kết quả tìm kiếm: Hiển thị danh sách tin tức của sản phẩm có liên quan đến từ khóa tìm kiếm, có thể bao gồm thông tin về tên sản phẩm, giá, hình ảnh, đánh giá, v.v.

Trang chi tiết tin tức: Hiển thị thông tin chi tiết về tin tức sản phẩm, mô tả thông tin sản phẩm, lượt đánh giá và bình luận của người dùng.

1. Sử dụng các phương pháp kỹ thuật để lấy cơ sở tìm test case

Equivalence Partitioning (Phân vùng tương đương): Phân chia các loại từ khóa tìm kiếm thành các nhóm tương đương, ví dụ: từ khóa hợp lệ, từ khóa không hợp lệ, từ khóa trống, v.v.

Boundary Value Analysis (Phân tích giá trị biên): Tìm kiếm với các giá trị biên như từ khóa ngắn nhất, từ khóa dài nhất, từ khóa có chứa ký tự đặc biệt, v.v.

Error Guessing (Đoán lỗi): Dựa trên kiến thức và kinh nghiệm, đoán các lỗi tiềm ẩn hoặc phổ biến có thể xảy ra trong quá trình tìm kiếm.

1. Danh sách các test case

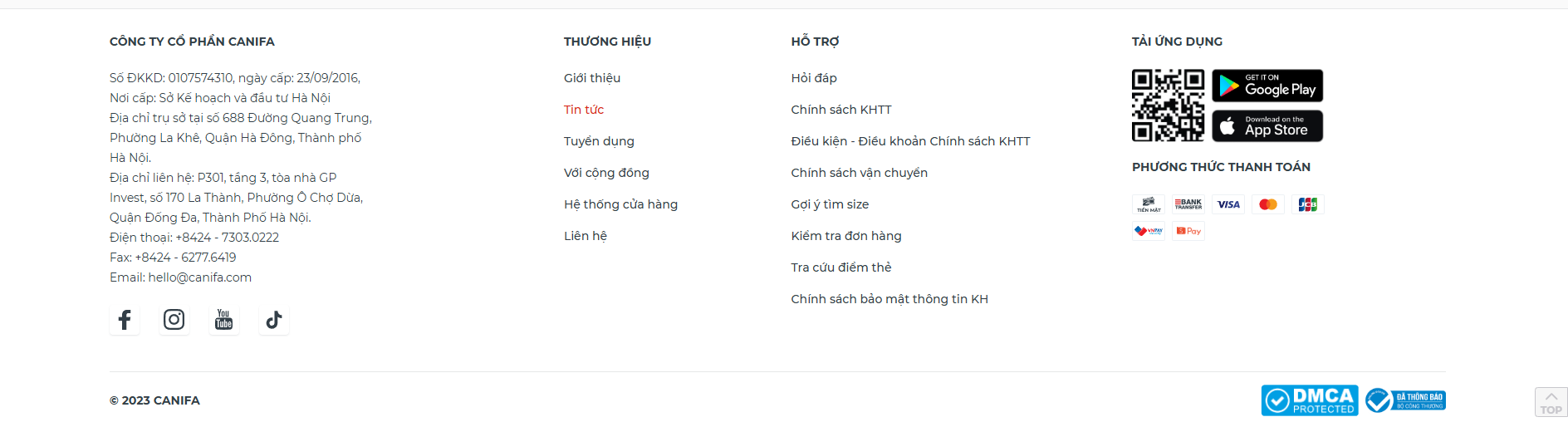
##### Bảng 3.5 Danh sách các test case (1)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Test case** | **Input** | **Output** | **Kết quả** |
| TC\_001 | Tìm kiếm với keyword đúng | Vải kaki | Tìm thấy thông tin từ khóa | Pass |
| TC\_002 | Tìm kiếm keyword sau | Quần jeaan | Không tìm thấy thông tin từ khóa | Pass |
| TC\_003 | Nhập keyword không tồn tại | Nhà cao tầng | Tìm thấy thông tin từ khóa | Pass |
| TC\_004 | Tìm kiếm keyword đã tồn tại bằng ngôn ngữ khác với không ngữ mặc định | Обувь | Không tìm thấy thông tin từ khóa | Pass |
| TC\_005 | Tìm kiếm với nhiều keyword | Áo dài, quần jean, giày nike | Tìm thấy thông tin từ khóa | Pass |
| TC\_006 | Tìm kiếm với keyword bỏ trống |  | Tìm thấy thông tin từ khóa | Pass |
| TC\_007 | Tìm kiếm với các ký tự đặc biệt | v@i | Không tìm thấy thông tin từ khóa | Pass |

#### 3.4.2.2 Thực hiện kiểm thử

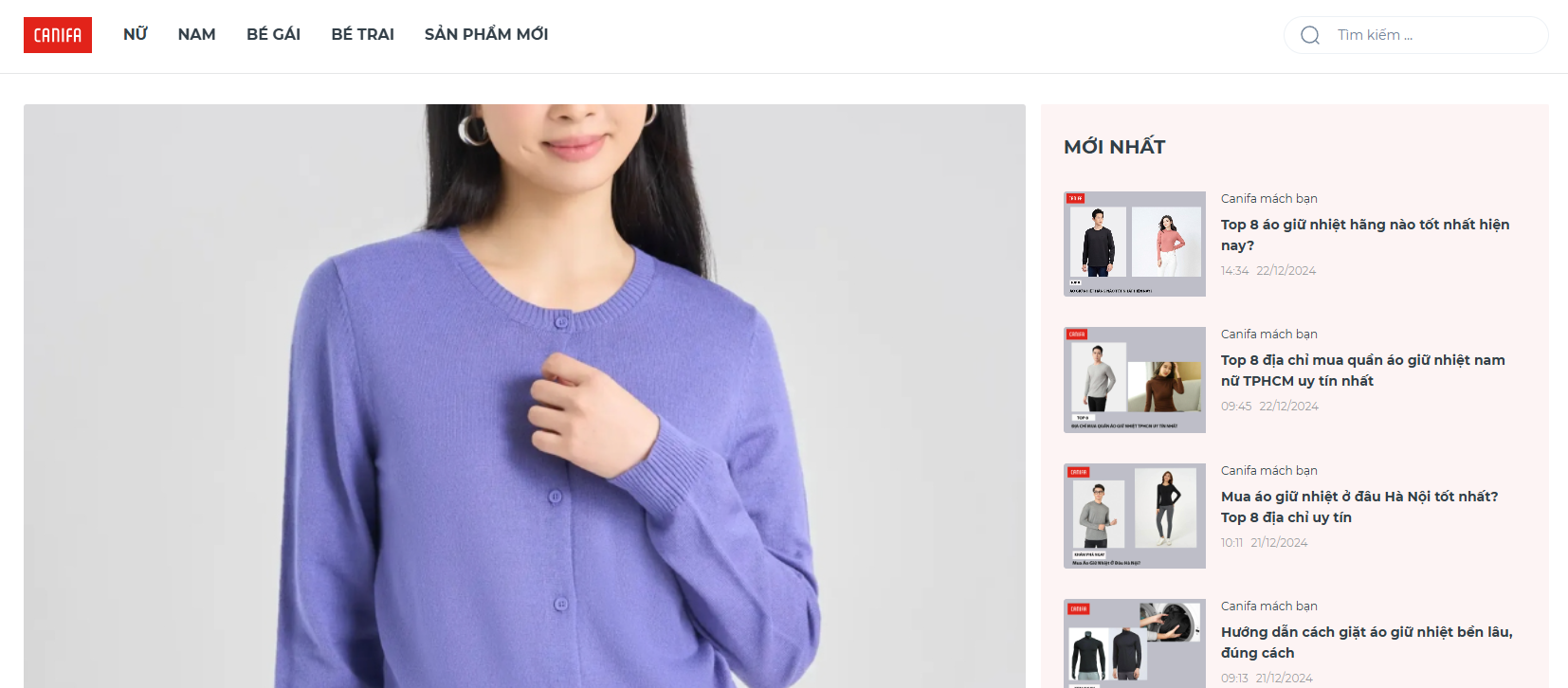
1. Các bước thực hiện

B1. Click vào mục ‘tin tức’



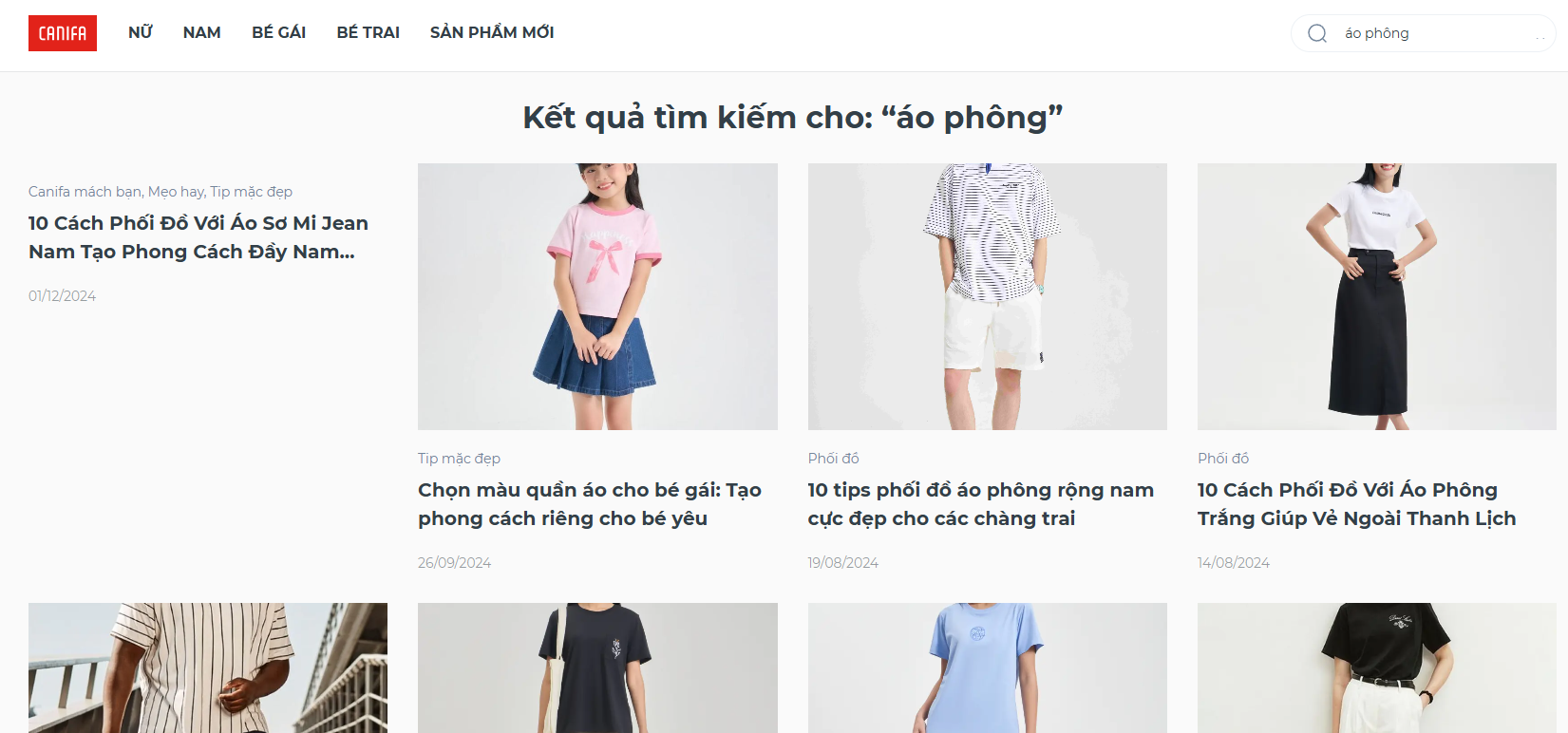
Hình 3. 14 Click vào mục tin tức.

B2. Click vào thanh tìm kiếm và nhập từ khóa



Hình 3. 15 Nhập từ khóa.

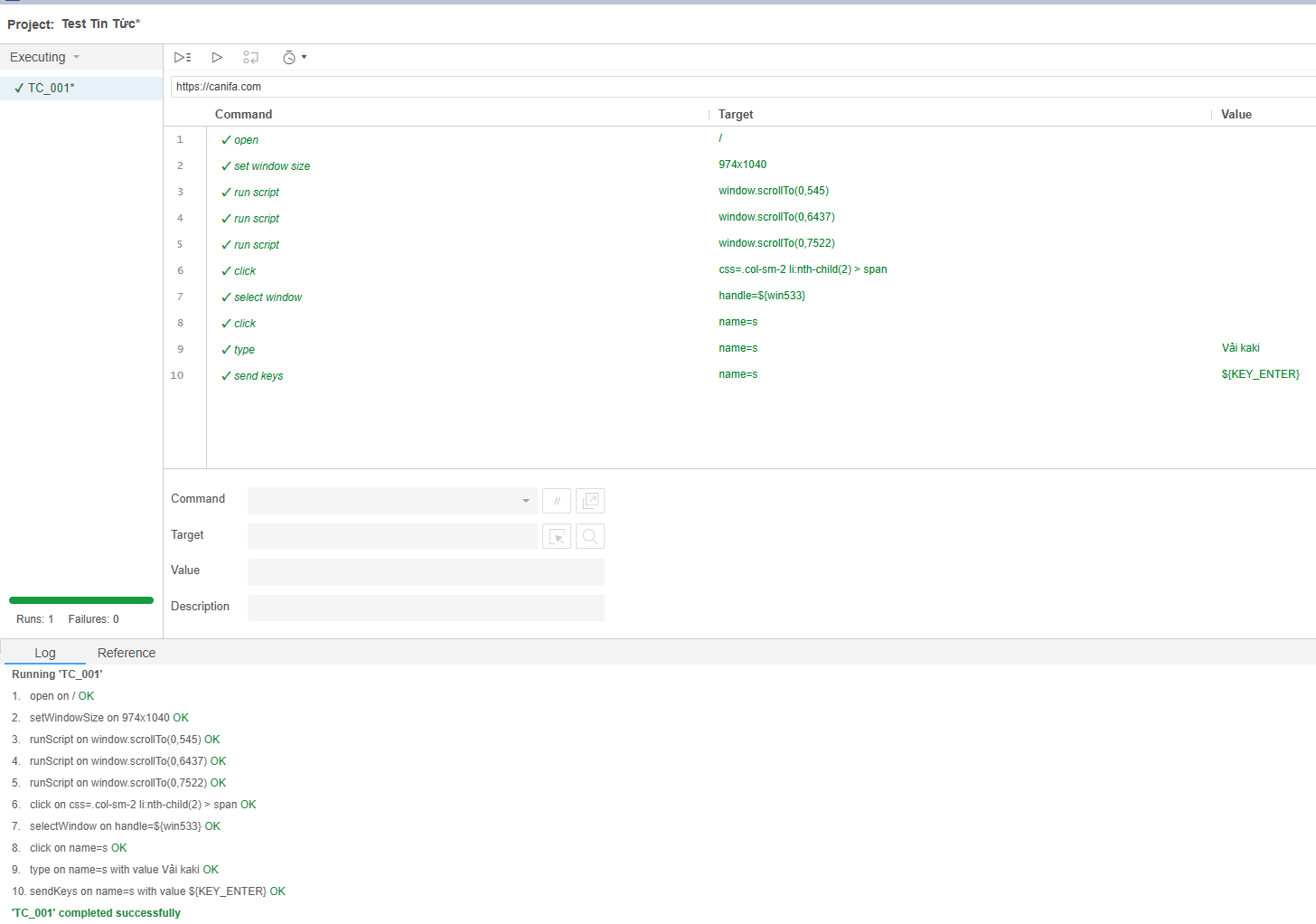
B3. Kích vào biểu tượng kính lúp để tìm kiếm.



Hình 3. 16 Tìm kiếm.

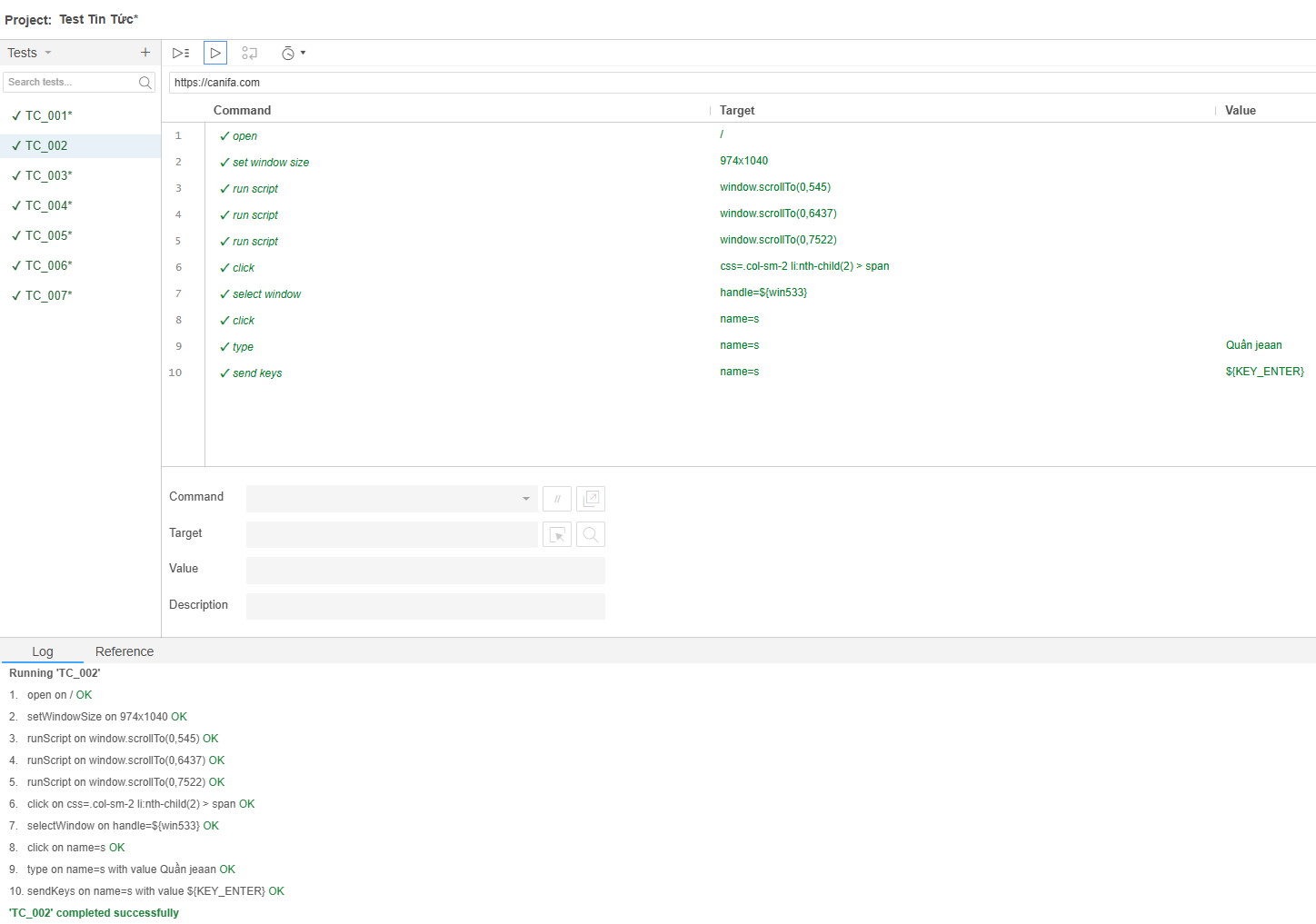
1. Kết quả các test case

TC\_001:



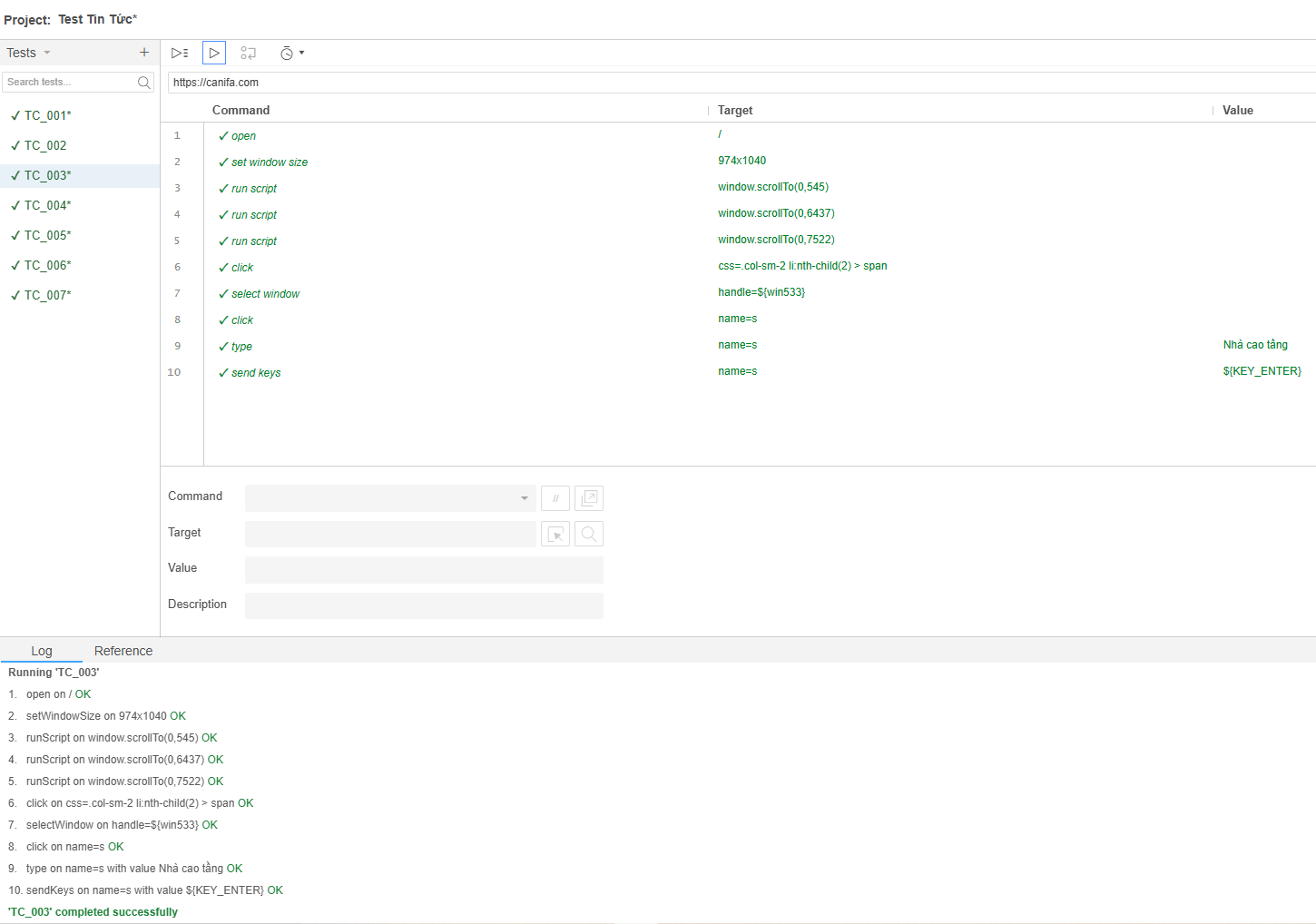
Hình 3. 17 Kết quả test case 1 (1).

TC\_002:



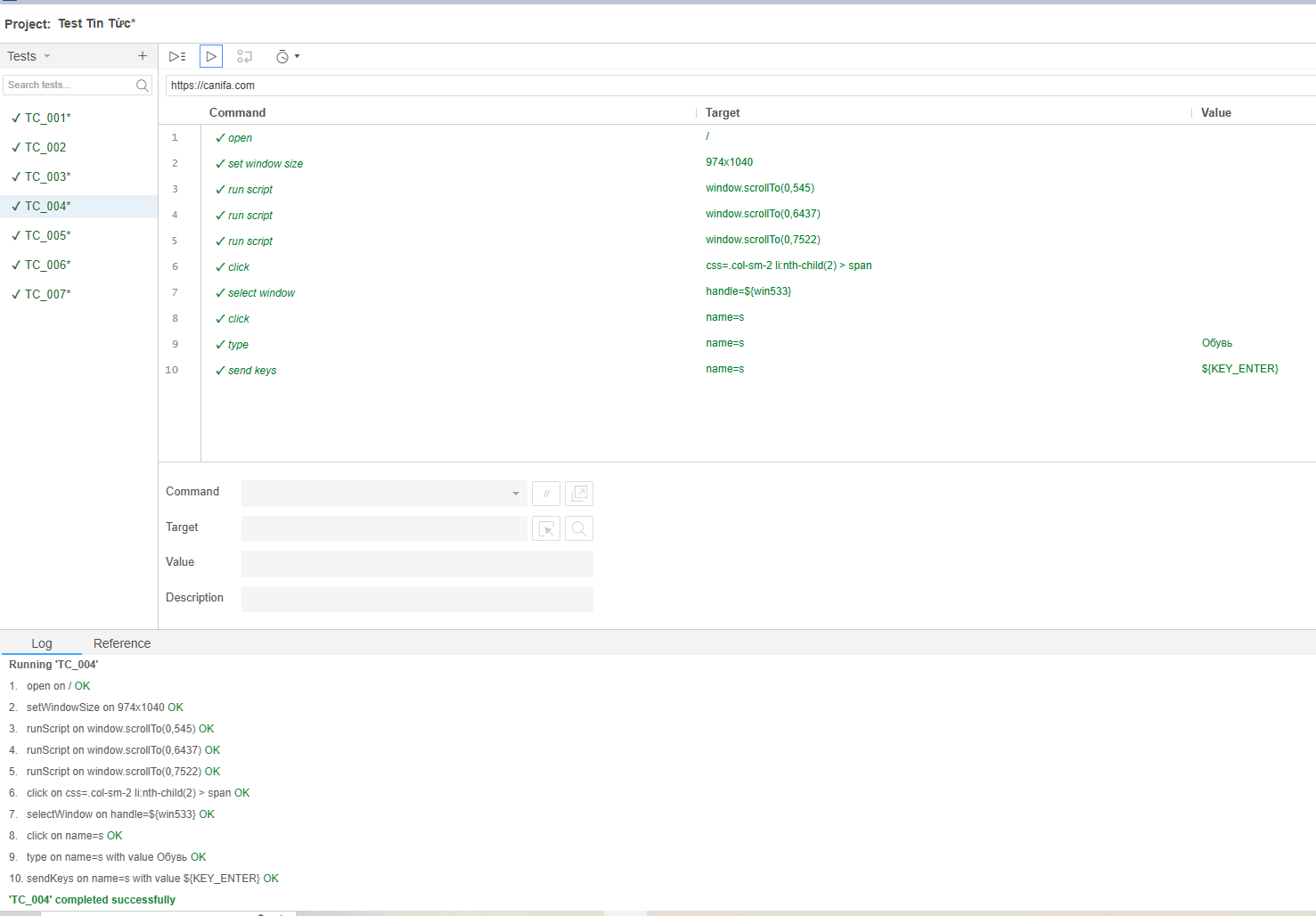
Hình 3. 18 Kết quả test case 2 (1).

TC\_003:



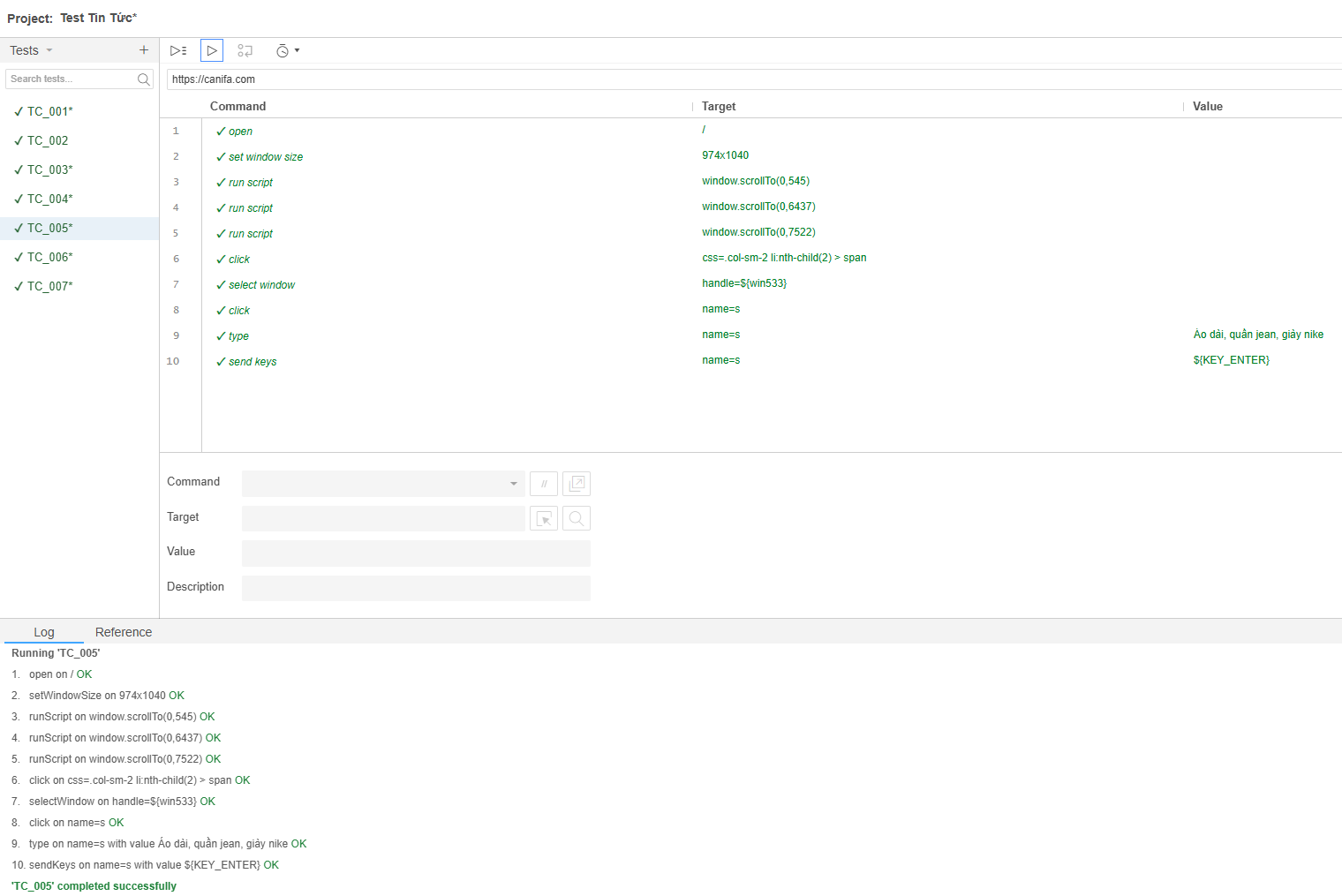
Hình 3. 19 Kết quả test case 3 (1).

TC\_004:



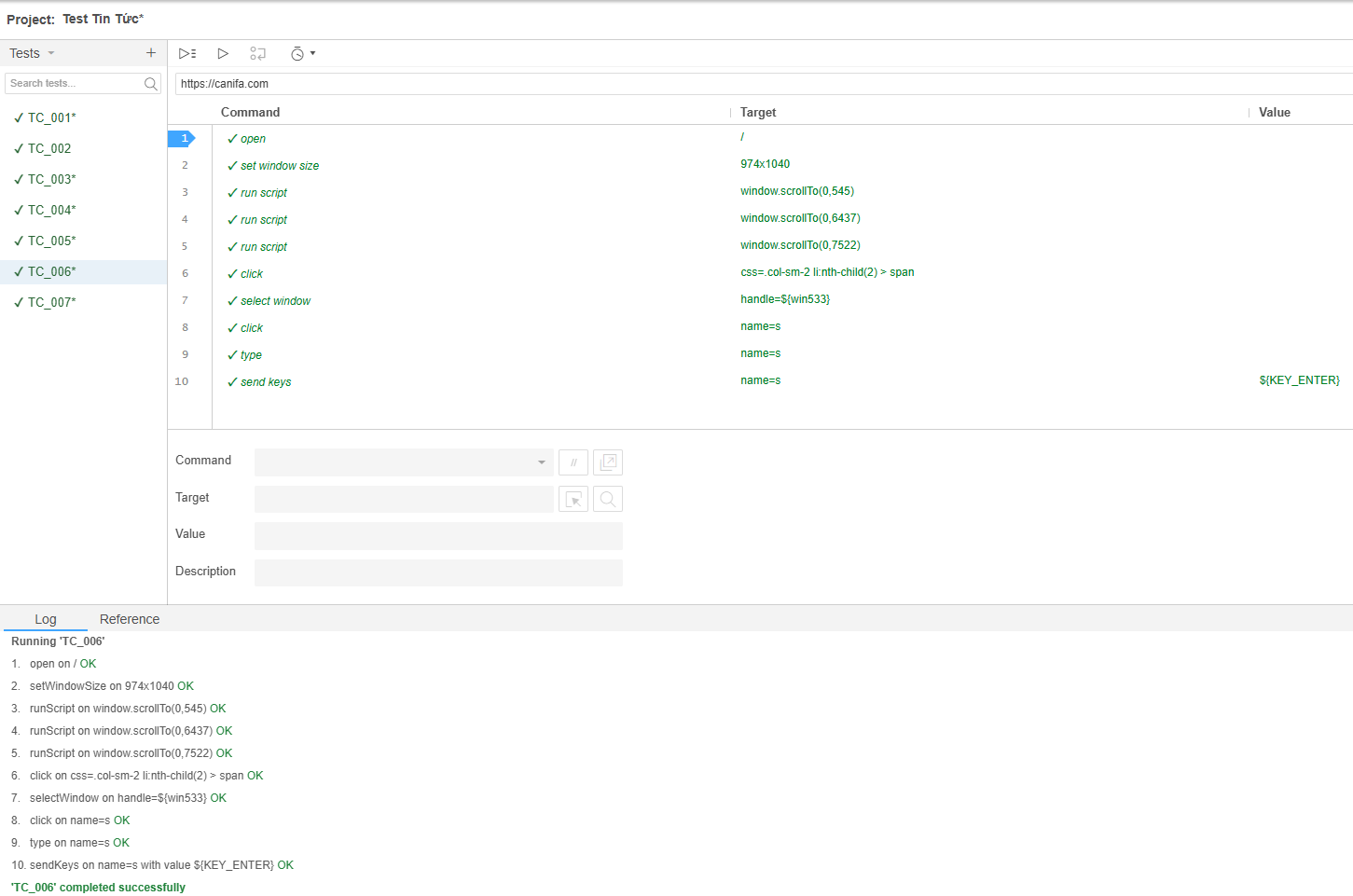
Hình 3. 20 Kết quả test case 4 (1).

TC\_005:



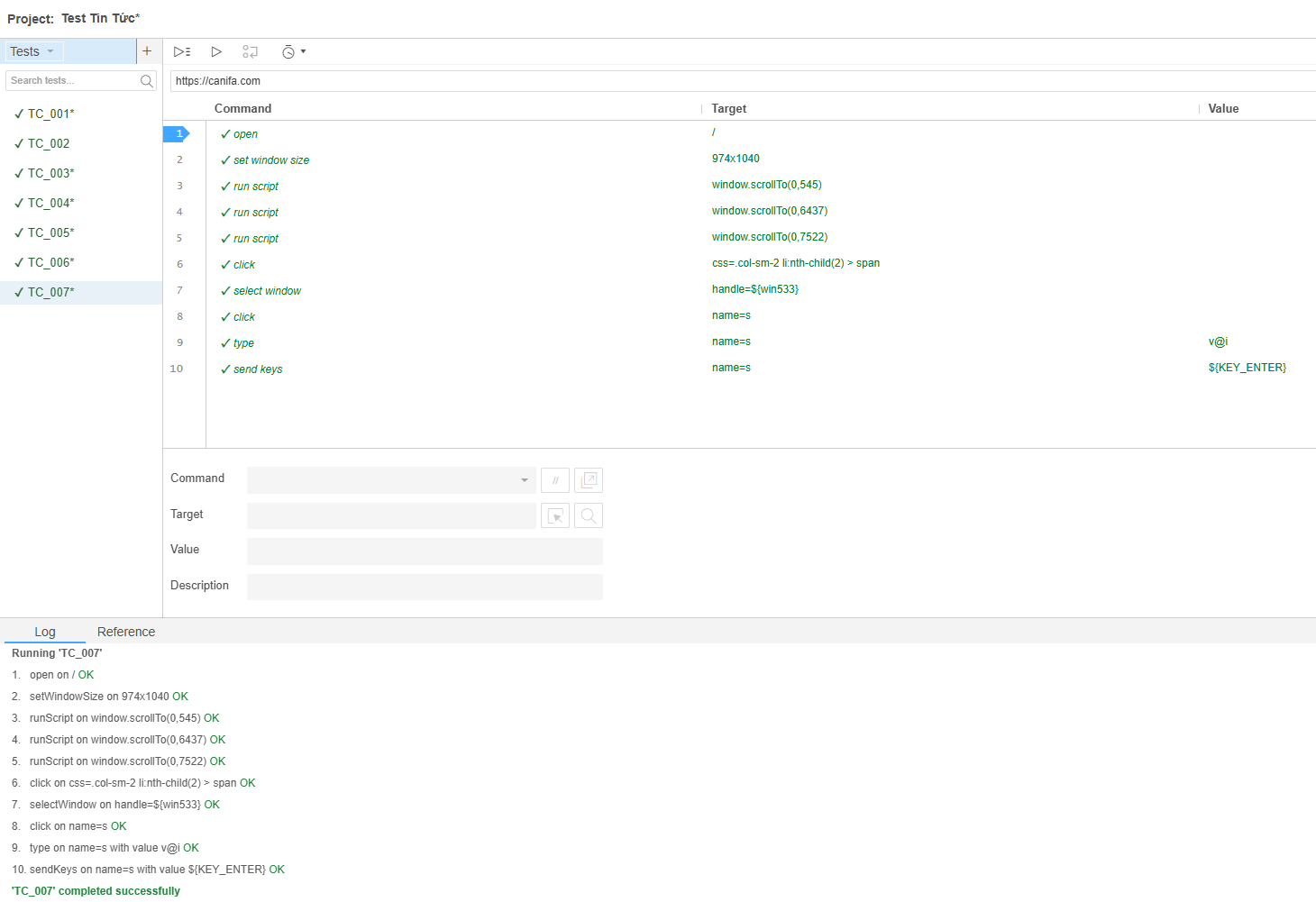
Hình 3. 21 Kết quả test case 5 (1).

TC\_006



Hình 3. 22 Kết quả test case 6 (1).

TC\_007



Hình 3. 23 Kết quả test case 7 (1).

Kết luận:

Website canifa.com sử dụng thuật toán search tên like – có nghĩa là kiểm tra các keyword có xuất hiện chính xác trong tin tức sản phẩm hay không. Số lượng pass test-case: 100%

### 3.4.3 Kiểm thử chức năng giỏ hàng

#### 3.4.3.1 Phân tích thiết kế kiểm thử

1. Giao diện chức năng

Giao diện chức năng giỏ hàng của Canifa gồm các tính năng như xem giỏ hàng, xem sản phẩm vào giỏ hàng, thêm và bớt số lượng sản phẩm trong giỏ hàng, xóa sản phẩm khỏi giỏ hàng.

Người dùng cần phải đăng nhập bằng tài khoản cá nhân vào trang web của Canifa để có thể sử dụng chức năng này.

Các sản phẩm có trong giỏ hàng cần thể hiện rõ thông tin về tên, số lượng, giá tiền của từng sản phẩm, tổng tiền và các thông tin liên quan khác.

1. Các điều kiện ràng buộc

Phải đăng nhập vào trang web trước khi sử dụng chức năng này.

Phải chọn kích thước sản phẩm mới có thể thêm sản phẩm vào giỏ hàng.

1. Phương pháp kỹ thuật lấy cơ sở tìm test case

Để lấy cơ sở tìm TC, chúng ta có thể sử dụng các phương pháp kỹ thuật như Boundary Value Analysis (Phân tích giá trị biên), Equivalence Partitioning (Phân vùng tương đương), Error Guesing (Đoán lỗi), …

* Boundary Value Analysis (BVA): phương pháp này tập trung vào các giá trị biên của các đầu vào, giúp xác định các giá trị đầu vào một cách chính xác và hiệu quả hơn.
* Equivalence Partitioning (EP): phương pháp này chia tập hợp các giá trị đầu vào thành các tập tương đương.
* Error Guessing là một phương pháp kiểm thử phần mềm dựa trên kinh nghiệm và khả năng đoán lỗi của người kiểm thử. Phương pháp này giúp tìm ra các lỗi tiềm ẩn hoặc phổ biến có thể xảy ra trong quá trình tìm kiếm, dựa trên kiến thức và kinh nghiệm của người kiểm thử.

1. Danh sách các test case

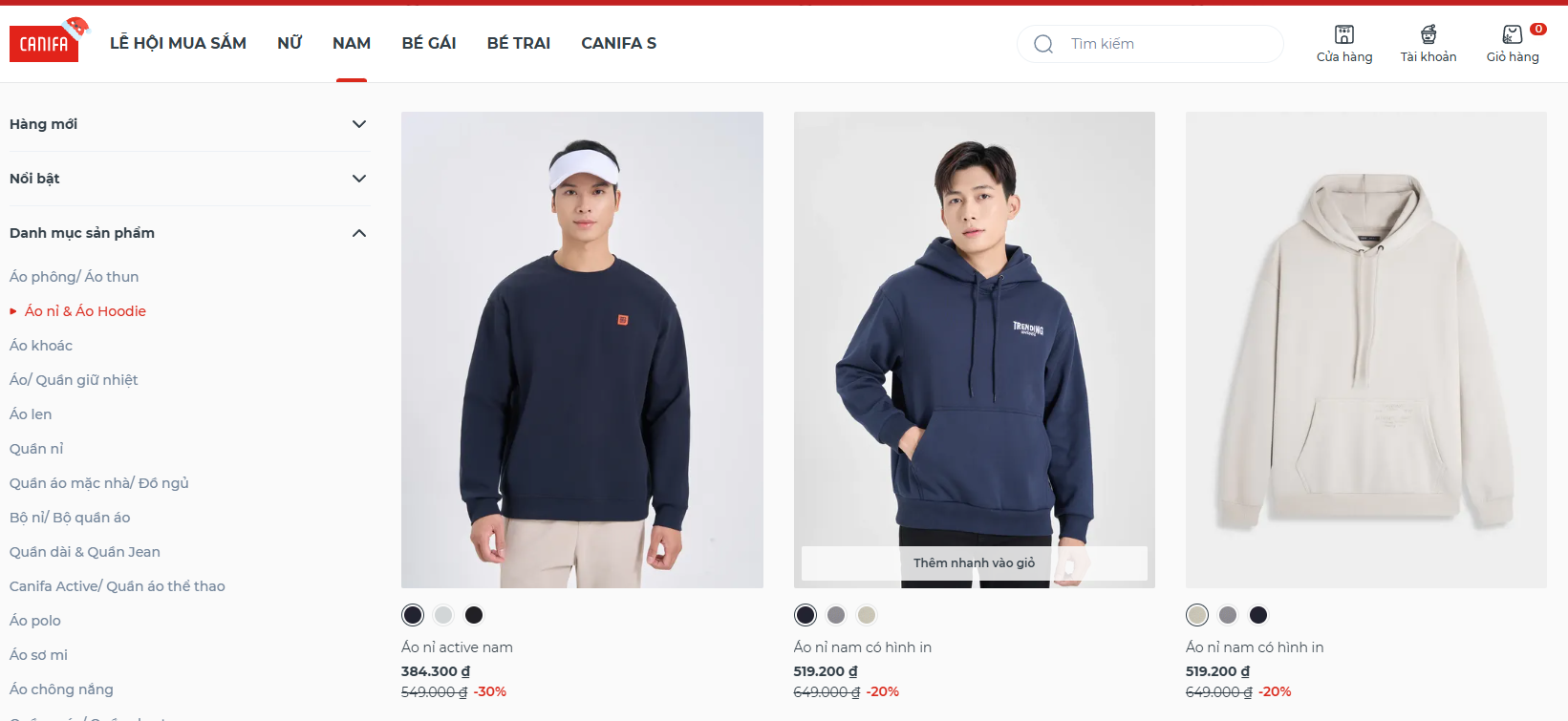
##### Bảng 3.6 Danh sách các test case (2)

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Input** | **Output** | **Kết quả** |
| TC\_001 | Chọn “màu sắc”  Chọn “size”  Chọn “Thêm vào giỏ hàng  Chọn “Xem giỏ hàng”  Chọn “xóa đơn hàng” | Thêm và xóa hàng ở giỏ hàng thành công | Pass |
| TC\_002 | Chọn “màu sắc”  Chọn “size” hết hàng  Chọn “thêm vào giỏ hàng” | Thêm hàng vào giỏ hàng không thành công | Pass |
| TC\_003 | Chọn “màu sắc”  Chọn “Thêm vào giỏ hàng” | Thêm hàng vào giỏ hàng không thành công | Pass |

#### 3.4.3.2 Thực hiện kiểm thử

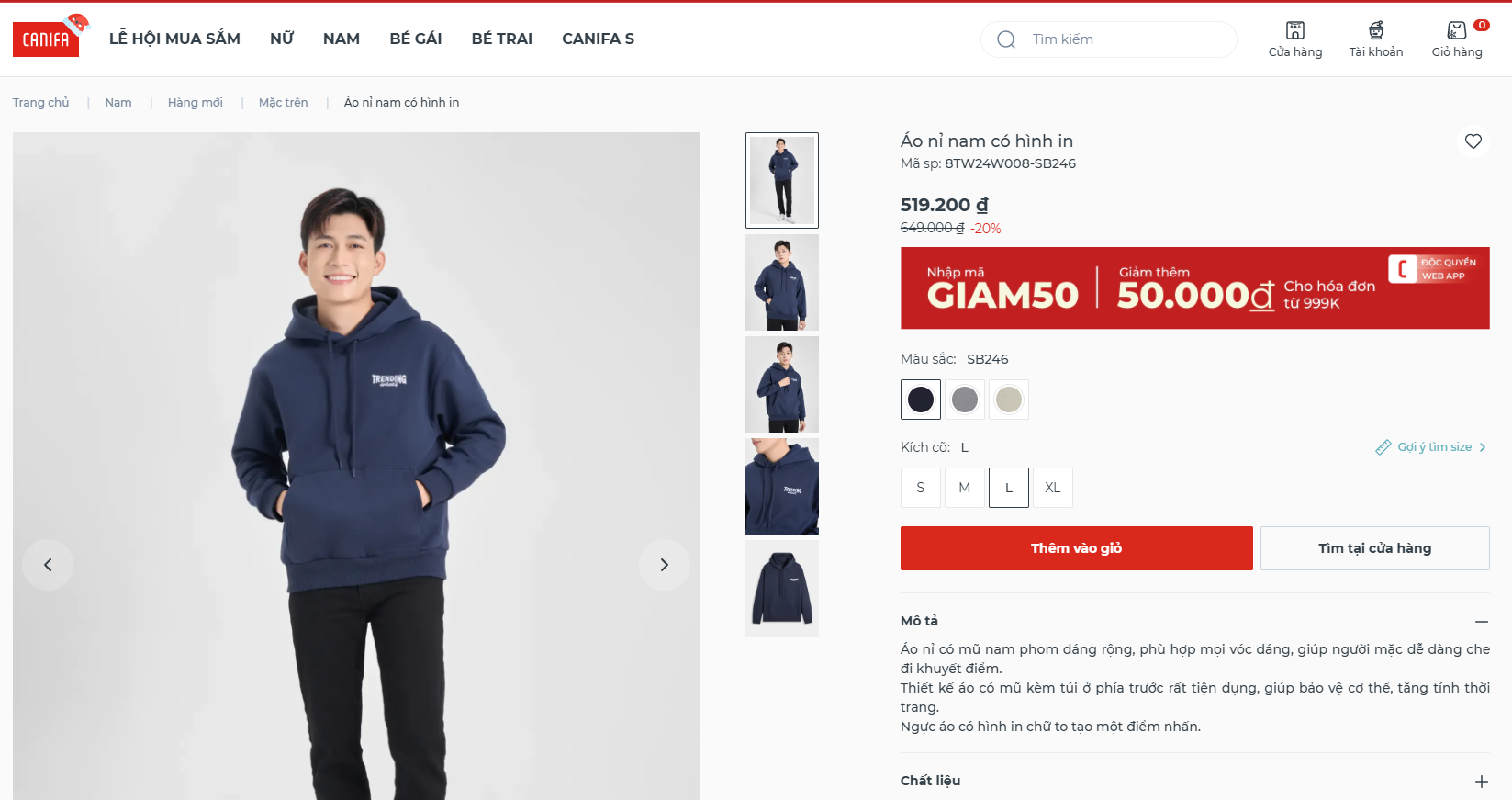
1. Các bước thực hiện

B1. Chọn một sản phẩm



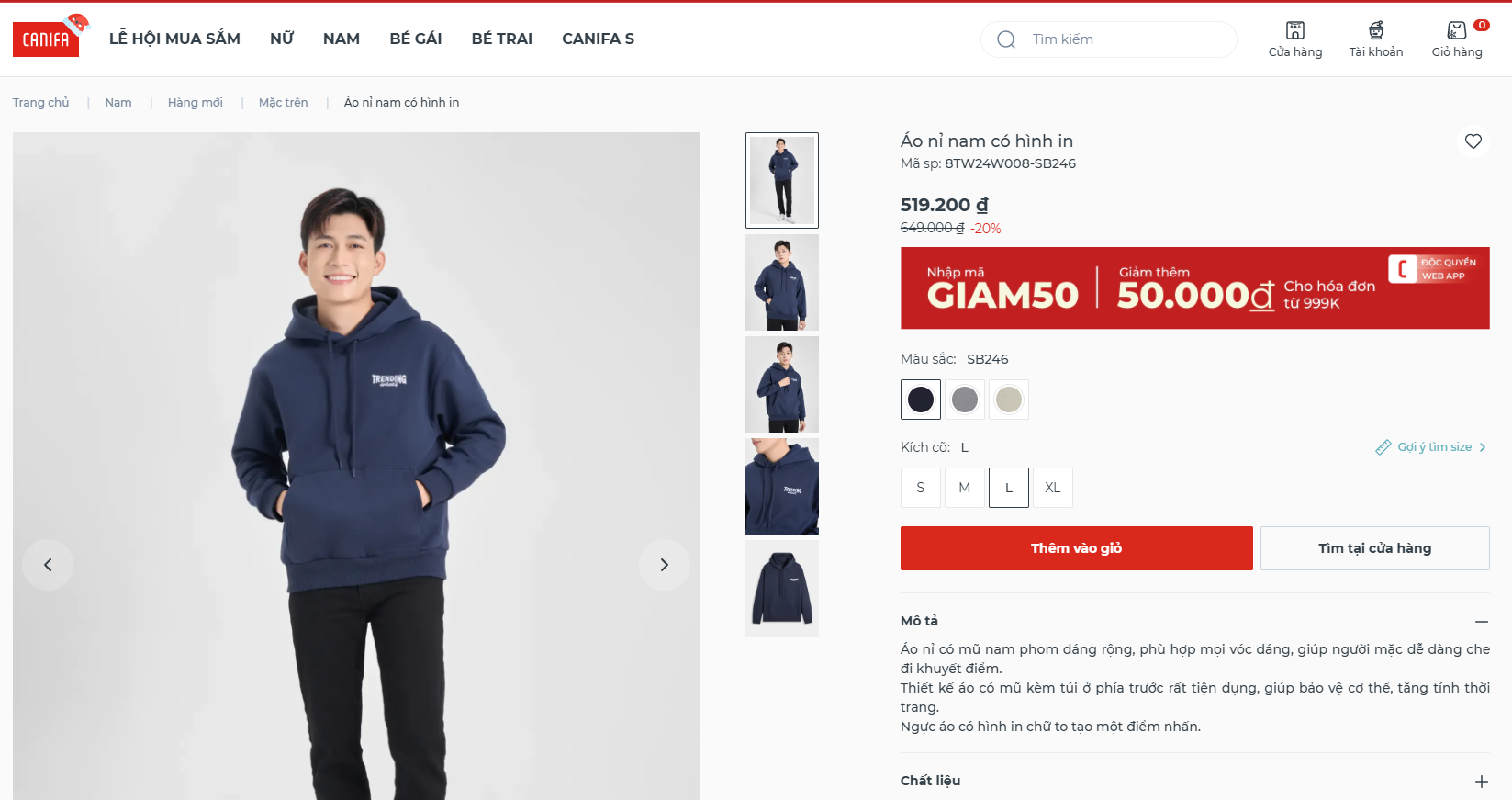
Hình 3. 24 Chọn một sản phẩm.

B2. Chọn màu và kích cỡ sản phẩm



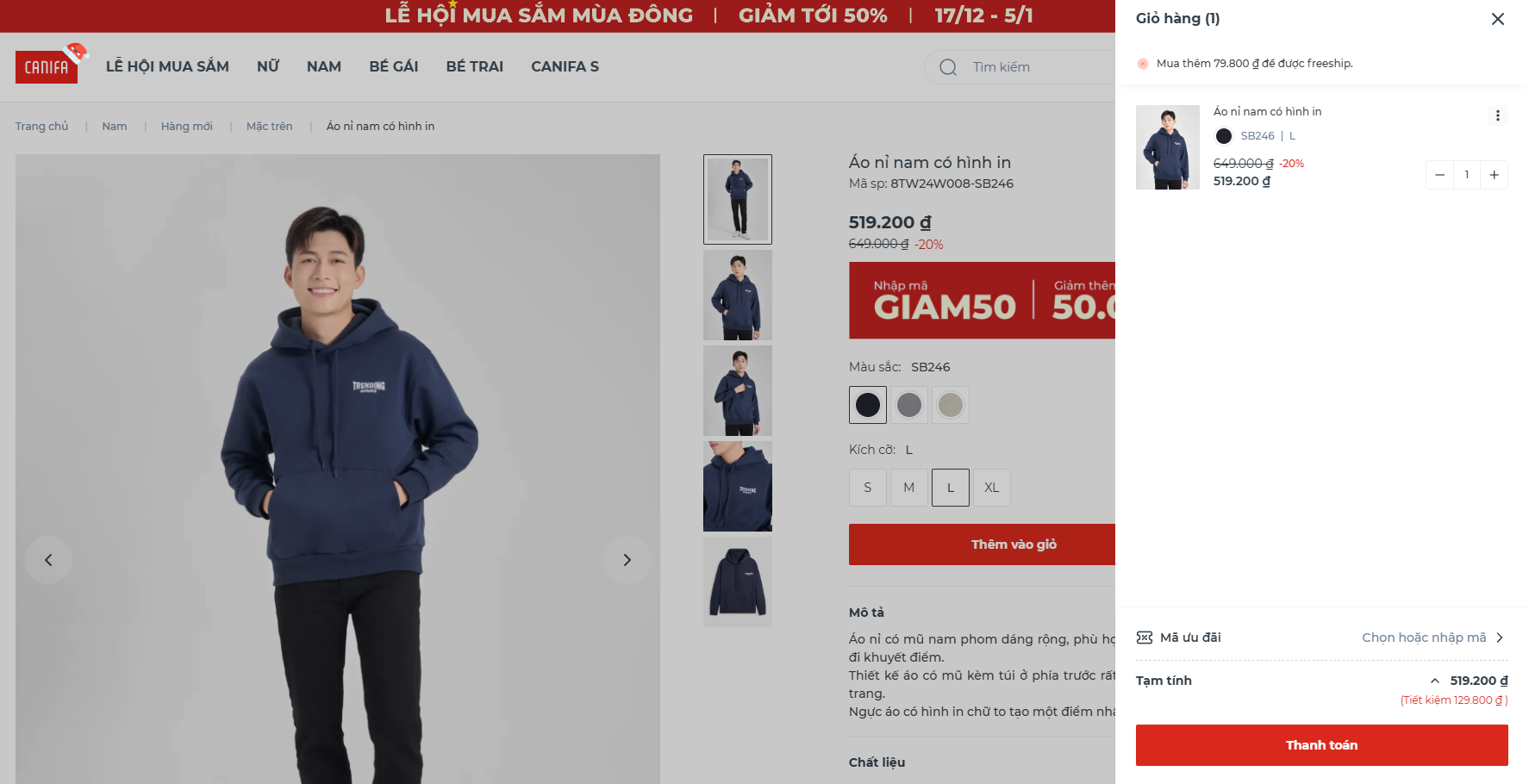
Hình 3. 25 Chọn màu và kích cỡ cho sản phẩm.

B3. Click Thêm vào giỏ



Hình 3. 26 Thêm vào giỏ hàng.

B4. Xem giỏ hàng

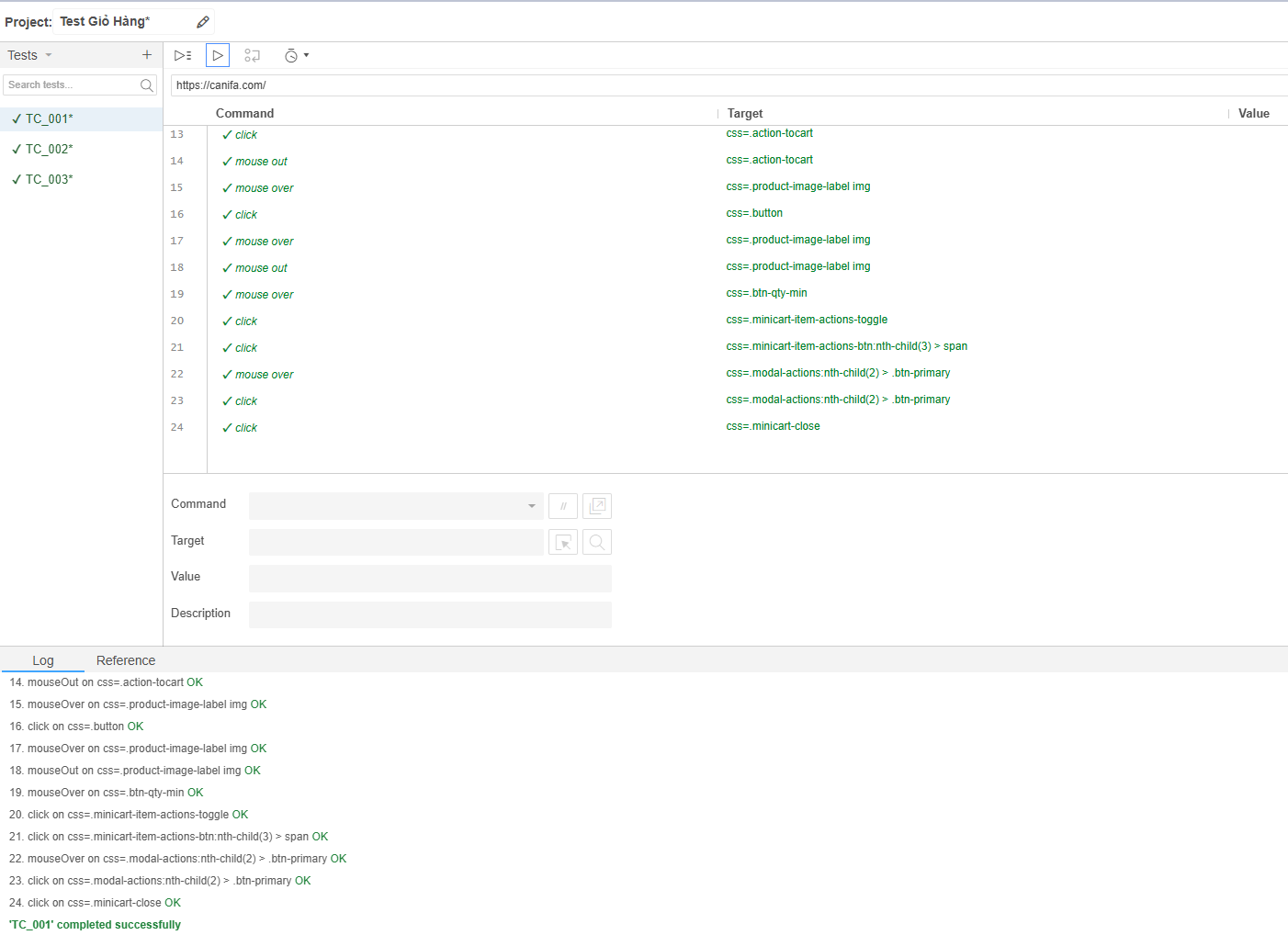


Hình 3. 27 Xem giỏ hàng

B5. Click vào biểu tượng “x” ở sản phẩm để xóa sản phẩm khỏi giỏ hàng.

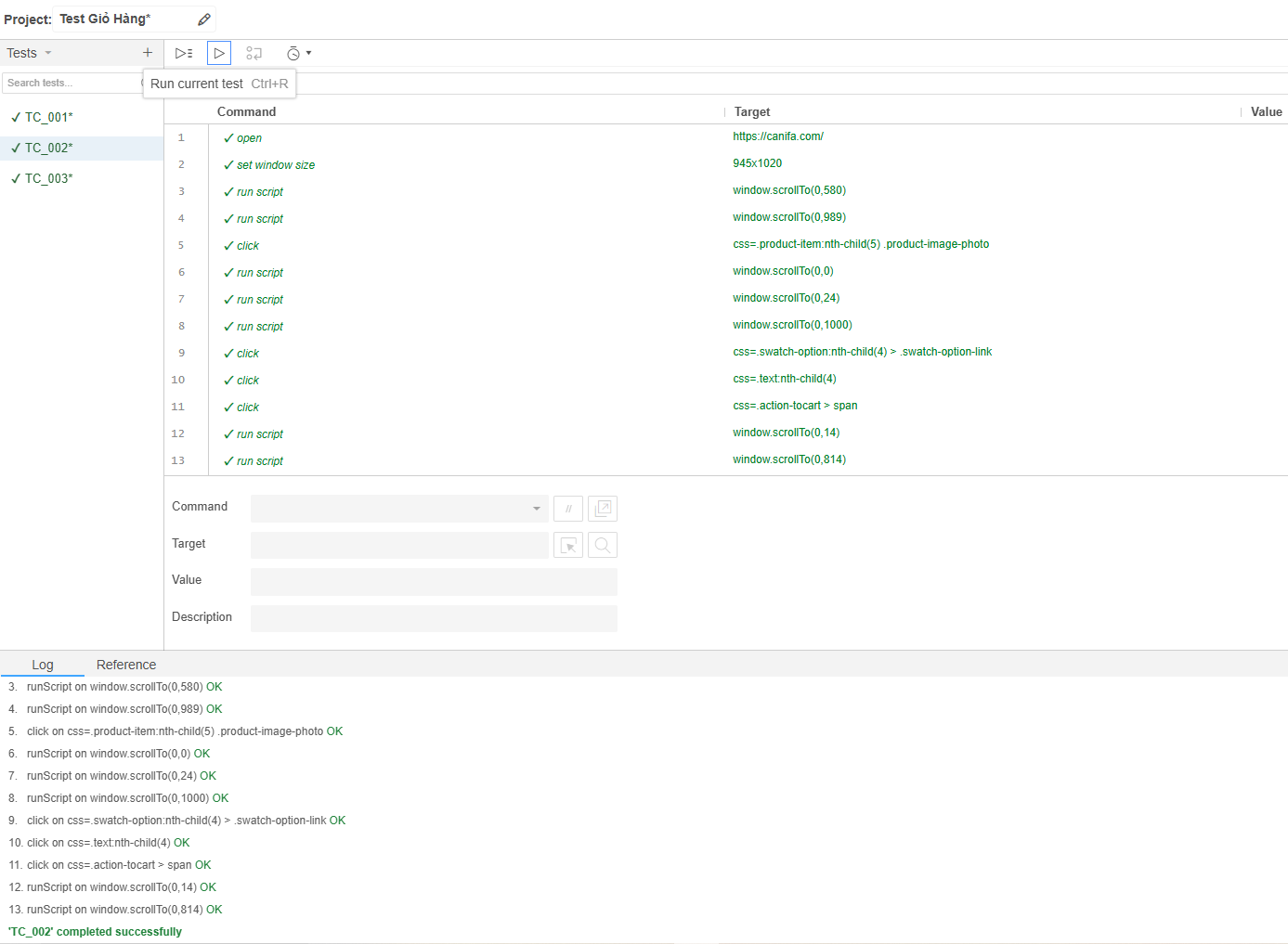
1. Kết quả các test case

TC\_001:



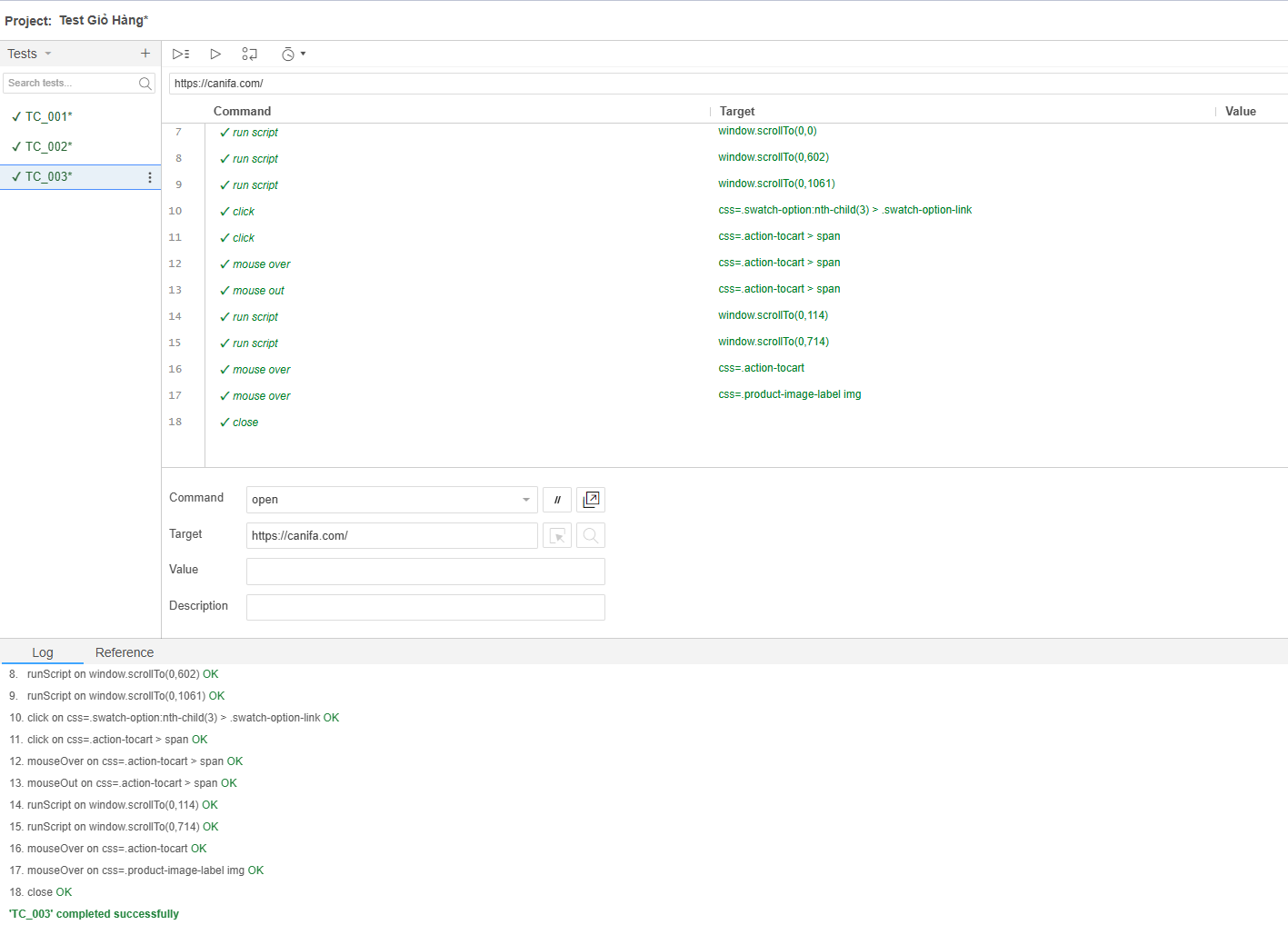
Hình 3. 28 Kết quả test case 1 (2).

TC\_002:



Hình 3. 29 Kết quả test case 2 (2).

TC\_003:



Hình 3. 30 Kết quả test case 3 (2).

Số lượng pass test-case: 100%

Tỷ lệ này cho thấy rằng việc sử dụng

kiểm thử tự động đã giúp đảm bảo tính chính xác và đáng tin cậy của quá trình kiểm thử.

### 3.4.4 Kiểm thử chức năng tìm kiếm sản phẩm

#### 3.4.4.1 Phân tích thiết kế kiểm thử

1. Giao diện chức năng

Giao diện chức năng tìm kiếm thường bao gồm 1 thanh tìm kiếm và một biểu tượng kính lúp.

Giao diện hiển thị kết quả tìm kiếm chứa danh sách sản phẩm liên quan đến từ khóa tìm kiếm.

Các sản phẩm được hiện thị bao gồm: tên sản phẩm, giá tiền, ảnh minh họa, …

1. Mô tả giao diện và các điều kiện ràng buộc

Ô tìm kiếm: Có điều kiện bắt buộc phải nhập từ khóa tìm kiếm.

Kết quả tìm kiếm: Hiển thị danh sách sản phẩm có liên quan đến từ khóa tìm kiếm bao gồm thông tin về tên sản phẩm, giá, hình ảnh.

Sử dụng các phương pháp kỹ thuật để lấy cơ sở làm test case:

* Equivalence Partitioning (Phân vùng tương đương): Phân chia các loại từ khóa tìm kiếm thành các nhóm tương đương, ví dụ: từ khóa hợp lệ, từ khóa không hợp lệv.v.
* Error Guessing (Đoán lỗi): Dựa trên kiến thức và kinh nghiệm, đoán các lỗi tiềm ẩn hoặc phổ biến có thể xảy ra trong quá trình tìm kiếm.
* Boundary Value Analysis (Phân tích giá trị biên): Tìm kiếm với các giá trị biên như từ khóa có ký tự đặc biệt, từ khóa không có ký tự đặc biệt, v.v.

1. Danh sách test case

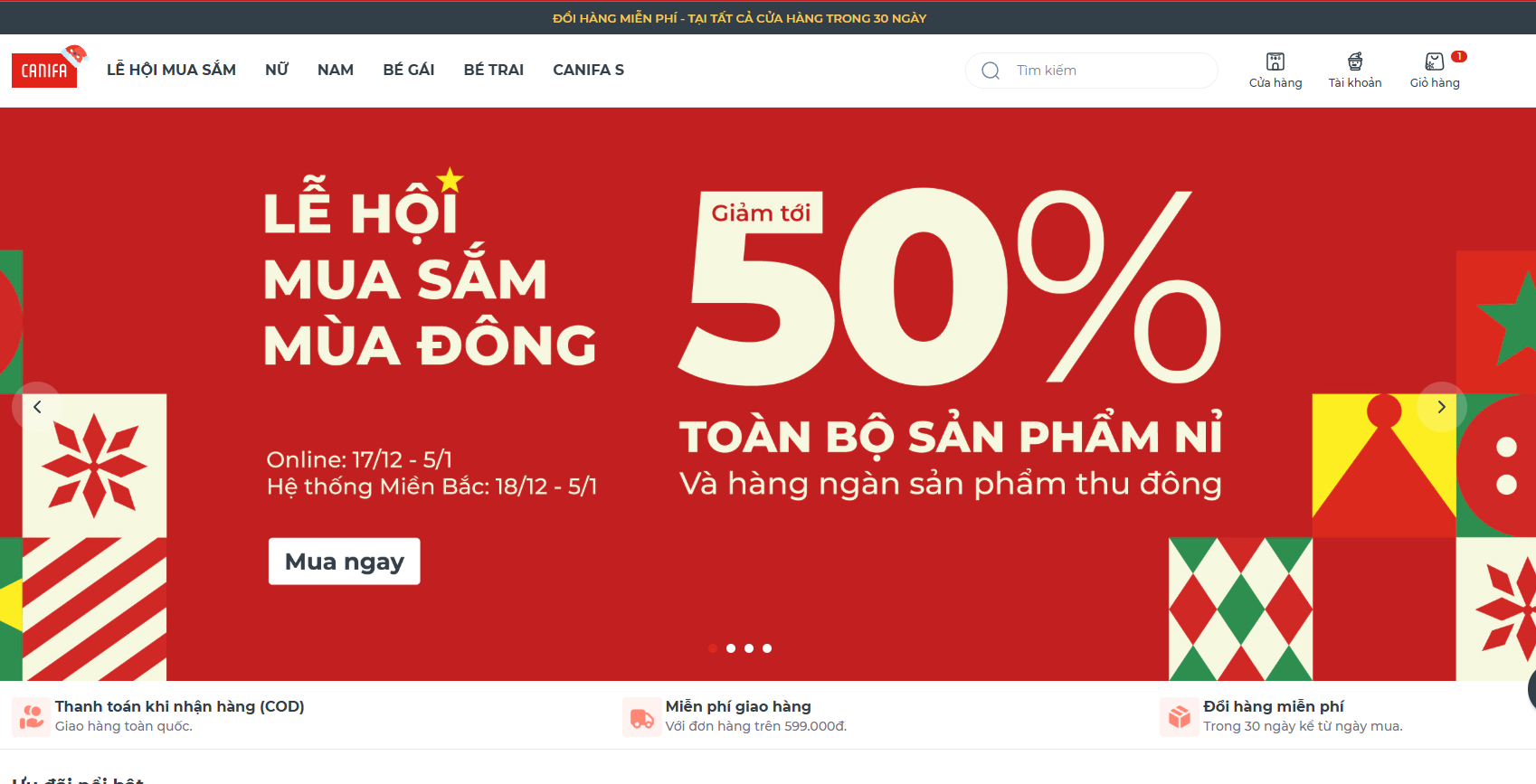
##### Bảng 3.7 Danh sách các test case (3)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Mô tả test case** | **Input** | **Output** | **Notice** |
| TC\_001 | Tìm kiếm với keyword liên quan | Áo phông | Các sản phẩm có liên quan đến từ khóa | Pass |
| TC\_002 | Tìm kiếm với keyword  không liên quan | Tay cầm chơi game | Không tìm  thấy sản  phẩm | Pass |
| TC\_003 | Tìm kiếm với keyword bị sai chính tả | Quaan | Không tìm  thấy sản  phẩm | Pass |
| TC\_004 | Tìm kiếm với keyword đúng nhưng với ngôn ngữ khác. | Shirt | Các sản phẩm có liên quan đến từ khóa | Pass |
| TC\_005 | Tìm kiếm với nhiều keyword | Quần, giày | Không tìm  thấy sản  phẩm | Pass |
| TC\_006 | Keyword có chứa ký tự đặc biệt | Qu@n | Không tìm  thấy sản  phẩm | Pass |

#### 3.4.4.2 Thực hiện kiểm thử

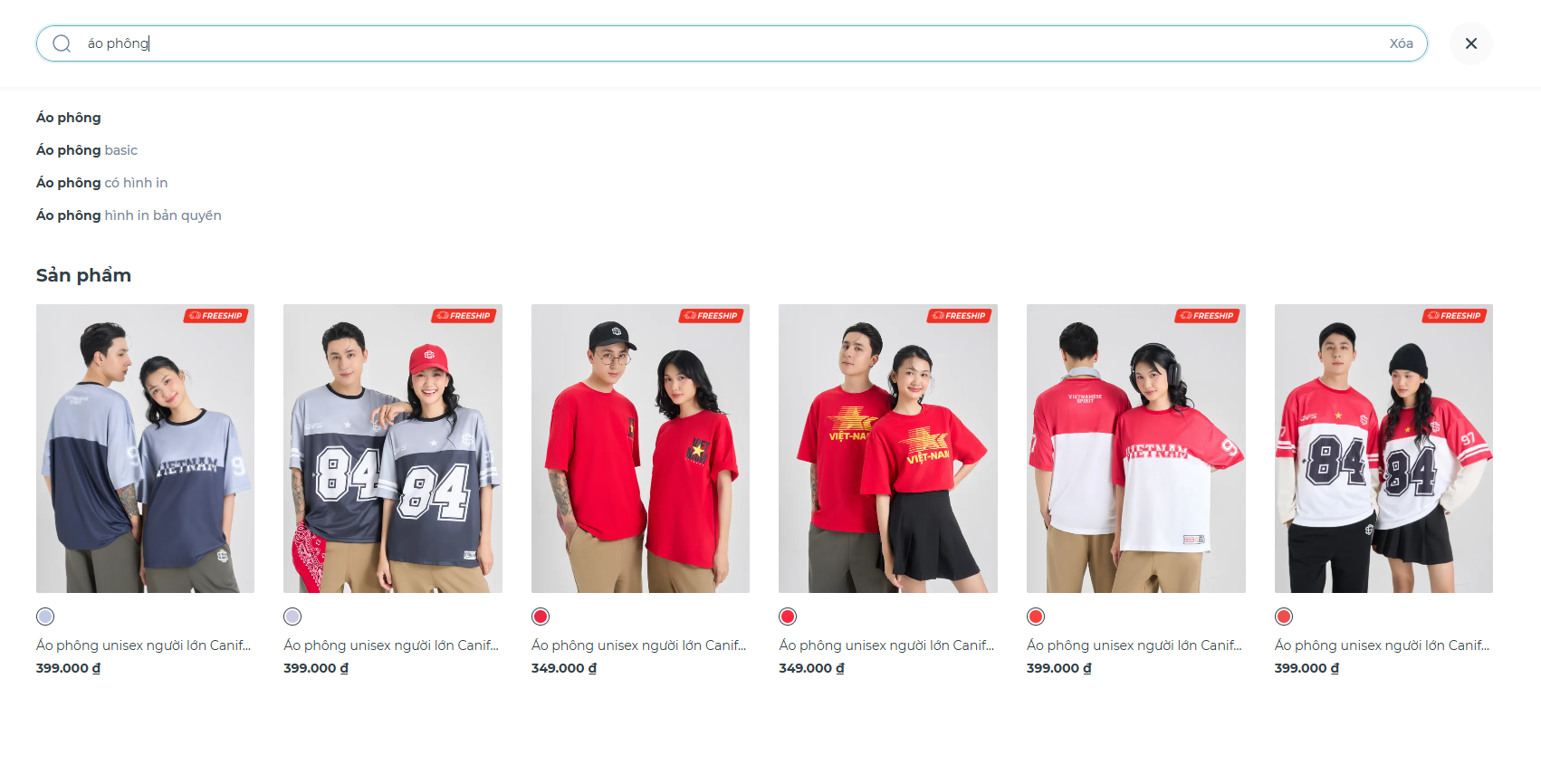
1. Các bước thực hiện

Bước 1: Click vào thanh tìm kiếm



Hình 3. 31 Tìm kiếm từ khóa.

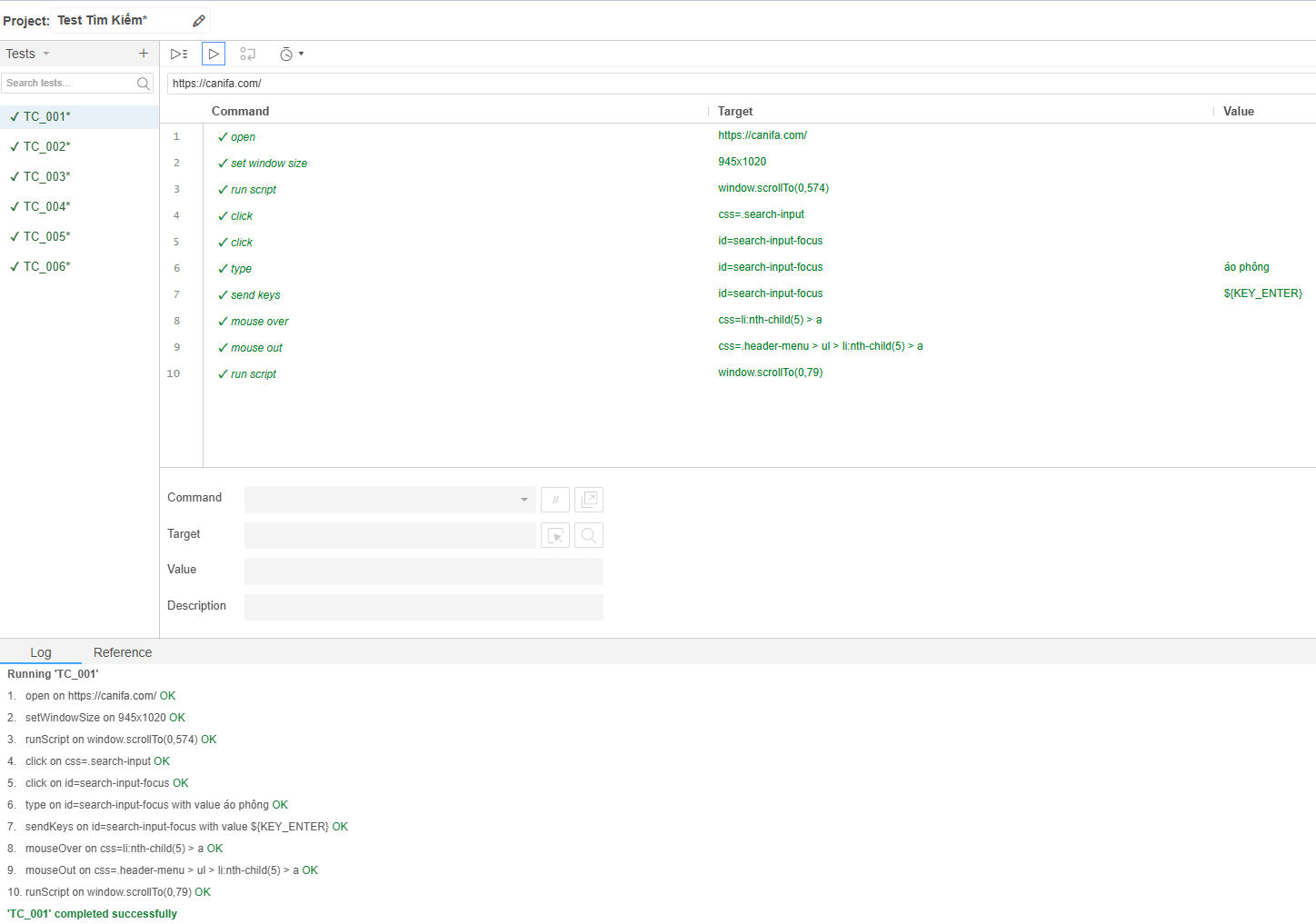
Bước 2: Nhập từ khóa vào thanh tìm kiếm Bước 3: Nhấn enter và nhận kết quả



Hình 3. 32 Kết quả tìm kiếm.

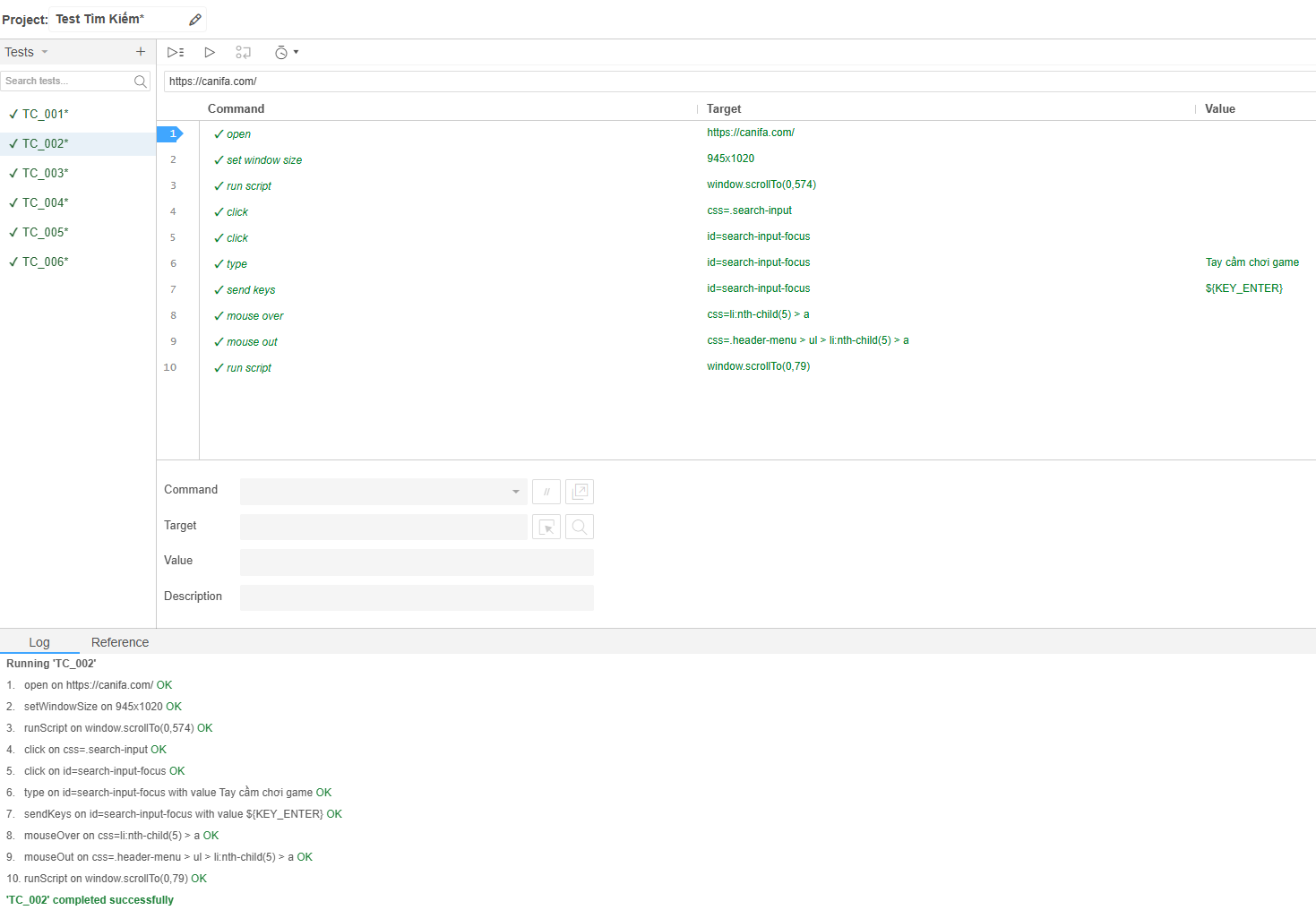
1. Kết quả các test case

TC\_001:



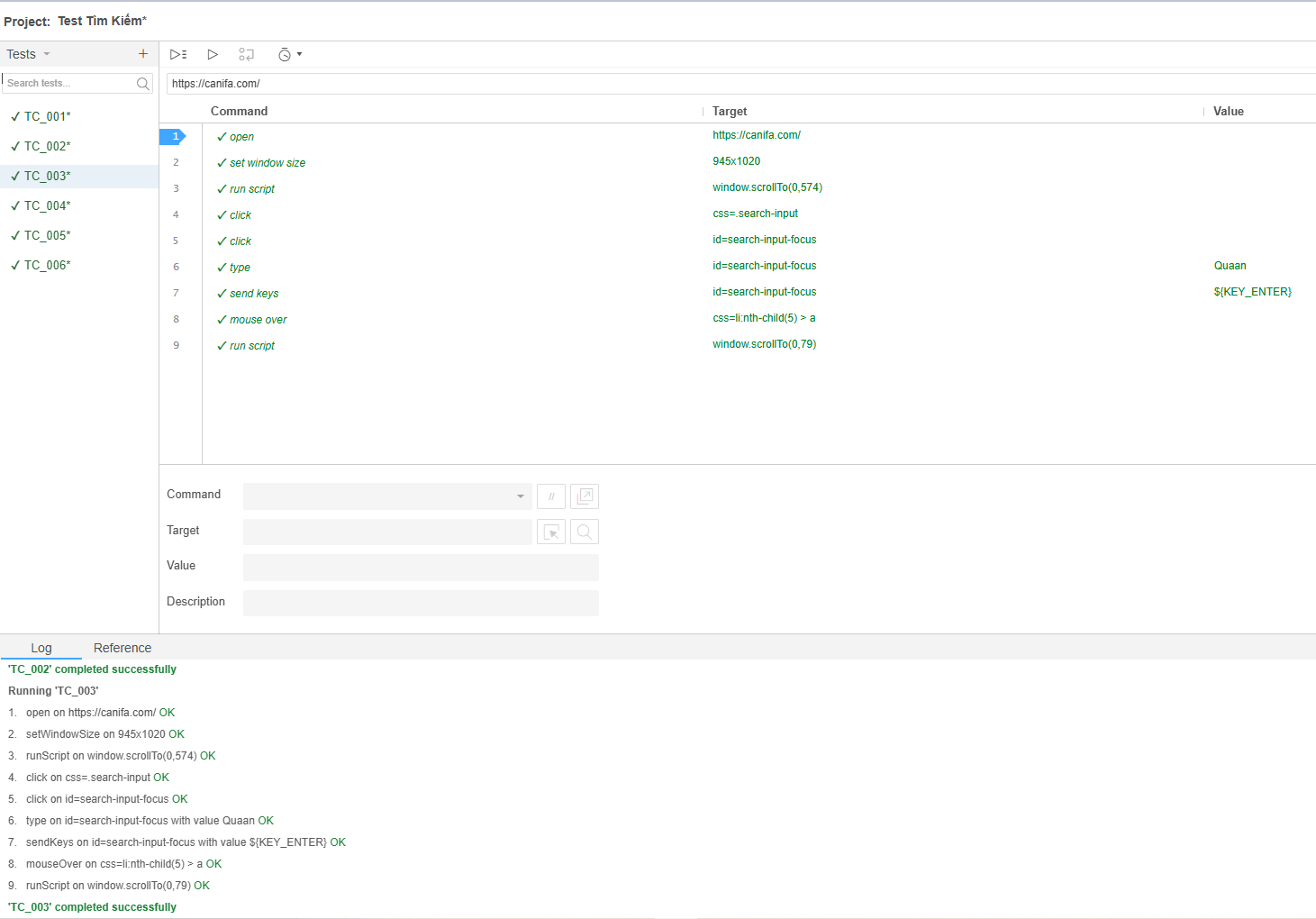
Hình 3. 33 Kết quả test case 1 (3).

TC\_002:



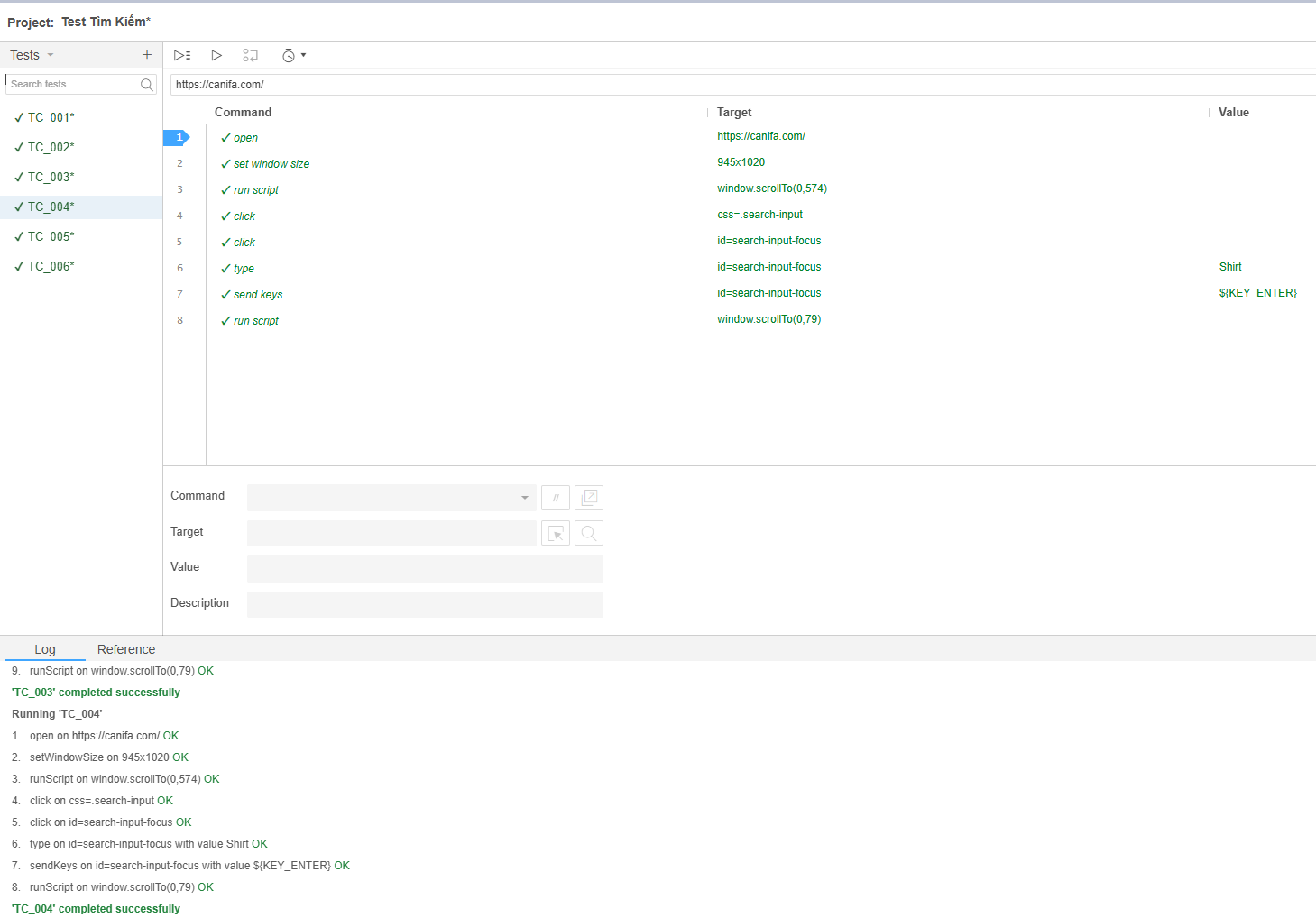
Hình 3. 34 Kết quả test case 2 (3).

TC\_003:



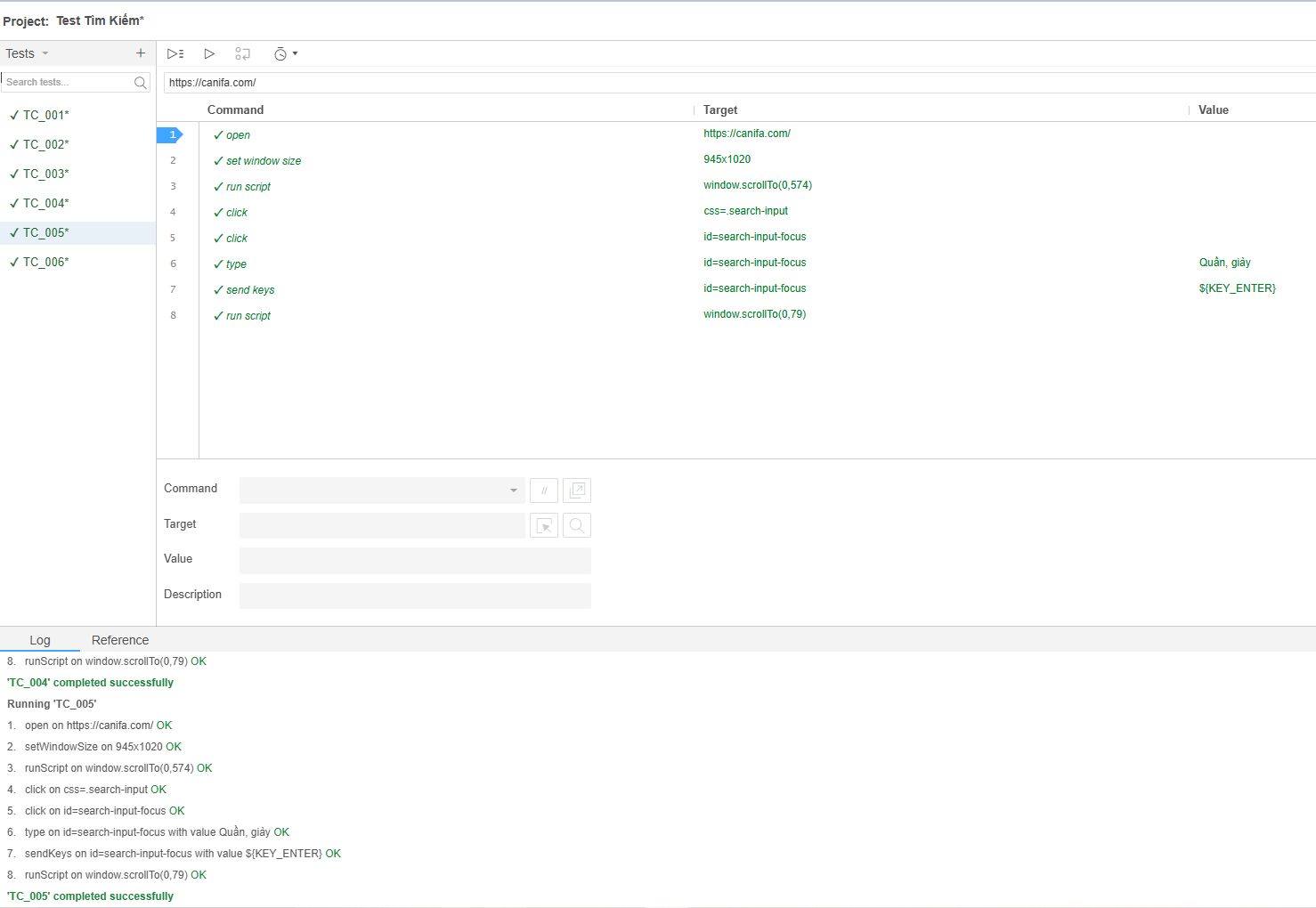
Hình 3. 35 Kết quả test case 3 (3).

TC\_004:



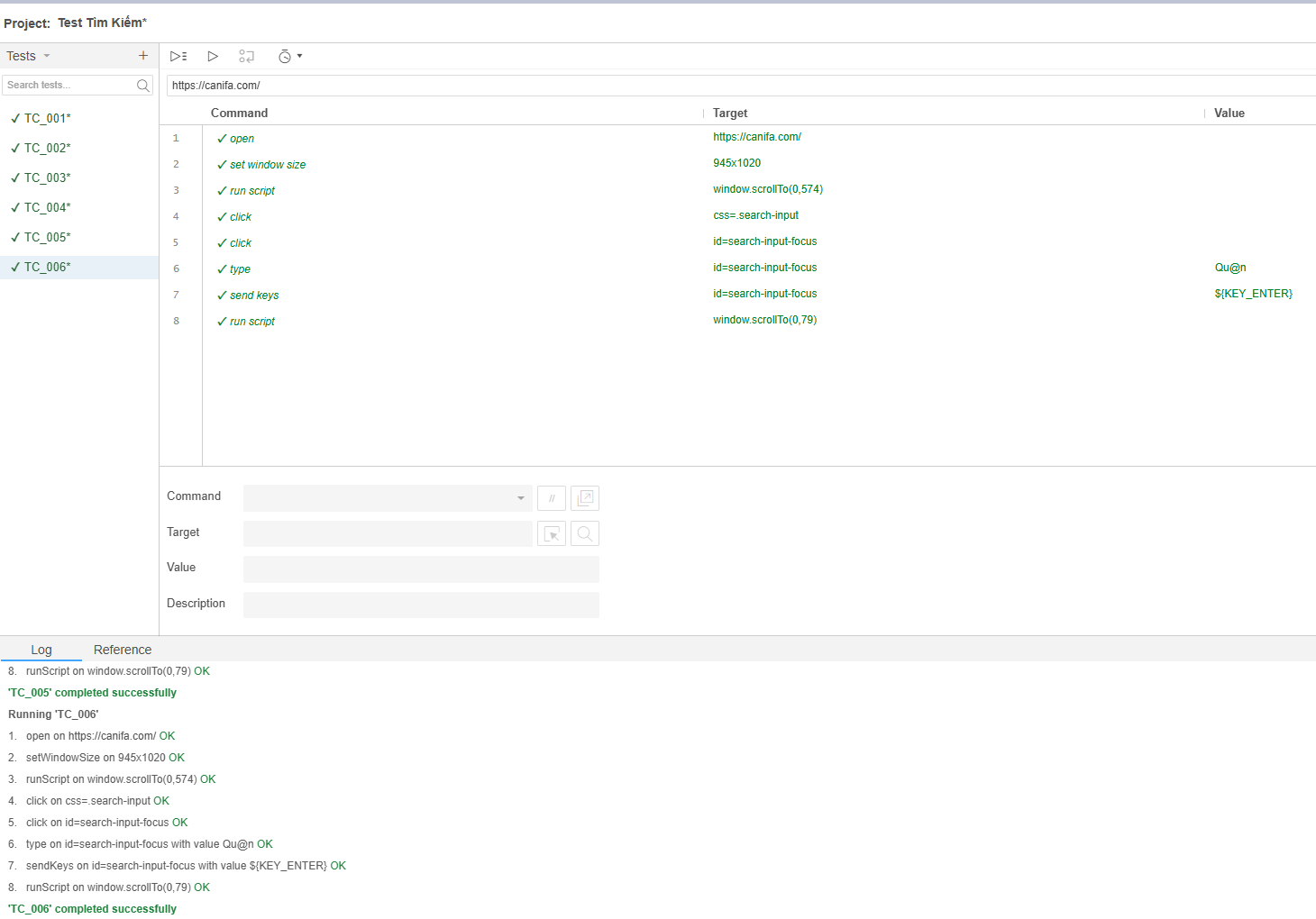
Hình 3. 36 Kết quả test case 4 (3).

TC\_005:



Hình 3. 37 Kết quả test case 5 (3).

TC\_006:



Hình 3. 38 Kết quả test case 6 (3).

Kết luận:

Trang wed canifa.com sử dụng thuật toán search tên like – có nghĩa là kiểm tra các keyword có xuất hiện chính xác trong tên sản phẩm hay không.

Số lượng pass test case: 100%

### 3.4.6 Báo cáo kiểm thử

##### Bảng 3.8 Báo cáo kiểm thử

|  |  |  |  |
| --- | --- | --- | --- |
| **Chức năng** | **Tổng số test case** | **Tỷ lệ thành công** | **Kết Luận** |
| Tìm kiếm tin tức | 7 | 100% | Hoạt động chính  xác và đáng tin cậy |
| Chức năng giỏ hàng | 3 | 100% | Hoạt động chính  xác và đáng tin cậy |
| Tìm kiếm sản phẩm | 6 | 100% | Hoạt động chính  xác và đáng tin cậy |

# PHẦN 4. KIẾN THỨC LĨNH HỘI VÀ BÀI HỌC KINH NGHIỆM

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trải nghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trảinghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trảinghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trảinghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trảinghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

Thông qua website và ứng dụng Canifa, người tiêu dùng có được trảinghiệm mua sắm đa kênh thuận tiện, dễ dàng và an toàn hơn. Khách hàng không cần xếp hàng, dễ dàng thanh toán, chủ động theo dõi và kiểm soát đơn hàng... Đồng thời, người dùng có thể cập nhật hạng mức thẻ thành viên một cách nhanh chóng, chính xác.

# PHẦN 5. TÀI LIỆU THAM KHẢO

[1] Alsmadi,Izzat 2012, " Advanced automated software testing "

[2] Spillner,Andreas 2014, " Software testing foundations:A study guide for the certified tester exam ".

[3] Thạc Bình Cường 2008, " Bài giảng: Kiểm thử và bảo đảm chất lượng phần mềm".

[4] Trần Tiến Dũng 2019, " Giáo trình công cụ và môi trường phát triển phần mềm".