|  |
| --- |
| ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯  TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI  KHOA CÔNG NGHỆ THÔNG TIN |
| **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC CHÍNH QUY**  **Ngành: Hệ thống thông tin** |
| **SỬ DỤNG CÔNG CỤ SELENIUM KIỂM THỬ WEBSITE THƯƠNG MẠI**  **Giảng viên hướng dẫn: TS. Lê Thị Anh**  **Sinh viên: Nguyễn Lê Huy MSV: 2019602593**  **Lớp: 2019DHHTTT01 Khoá: 14**  **HÀ NỘI - 2024** |

|  |
| --- |
| TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI  KHOA CÔNG NGHỆ THÔNG TIN  **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC CHÍNH QUY**  **Ngành: Hệ thống thông tin** |
| **SỬ DỤNG CÔNG CỤ SELENIUM KIỂM THỬ WEBSITE THƯƠNG MẠI**  **Giảng viên hướng dẫn: TS. Lê Thị Anh**  **Sinh viên: Nguyễn Lê Huy MSV: 2019602593**  **Lớp: 2019DHHTTT01 Khoá: 14**  **HÀ NỘI - 2024** |

1. **Tính cấp thiết của đề tài**

Ngày nay, công nghệ thông tin nói chung và công nghệ phần mềm nói riêng đang chiếm một vị trí quan trọng trong tiến trình công nghiệp hoá, hiện đại hoá đất nước. Song song với việc phát triển công nghệ phần mềm luôn tiềm ẩn những thách thức cho dành các doanh nghiệp, nhà phát triển phần mềm trong việc kiểm soát lỗi, chất lượng đầu ra của sản phẩm. Nguyên nhân gây thiệt hại về kinh tế và xã hội trong phần mềm không chỉ là lỗi về bảo mật mà còn là lỗi về một chức năng nào đó trong sản phẩm.

Kiểm thử phần mềm là một quá trình liên tục, xuyên suốt mọi giai đoạn phát triển phần mềm để đảm bảo rằng phần mềm thỏa mãn yêu cầu thiết kế và yêu cầu đó đáp ứng được nhu cầu của người sử dụng. Các kỹ thuật kiểm thử phần mềm đang được nghiên cứu và việc kiểm thử phần mềm trở thành quy trình bắt buộc trong các dự án phát triển phần mềm trên thế giới. Ngày nay xu hướng áp dụng tự động hoá đang được triển khai rộng rãi ở nhiều lĩnh vực, trong đó có kiểm thử phần mềm. Đặc biệt, khi kiểm thử phần mềm là công đoạn chiếm phần lớn thời gian trong quá trình phát triển dự án phần mềm thì sự ra đời của các công cụ kiểm thử tự động càng có ý nghĩa hơn bao giờ hết, giúp tiết kiệm thời gian, công sức và tiền bạc.

1. **Mục tiêu thực hiện đồ án**

Với mong muốn được tìm hiểu sâu về lĩnh vực kiểm thử phần mềm cũng như bổ sung thêm kiến thức cho lĩnh vực kiểm thử và bảo mật, em đã chọn đề tài “**Kiểm thử ứng dụng trên nền Web bằng công cụ Selenium** ”.

Mục tiêu đặt ra khi thực hiện đồ án:

1. Đưa ra hướng cài đặt, sử dụng hiệu quả công cụ
2. Ứng dụng các kiến thức đã học được để viết một kịch bản kiểm thử cho ứng dụng website cụ thể.

**LỜI CẢM ƠN**

Lời đầu tiên em xin chân thành cảm ơn các thầy, các cô trong khoa công nghệ thông tin đã tạo điều kiện thuận lợi cho em trong quá trình học tập tại trường cũng như trong thời gian thực hiện đồ án tốt nghiệp. Đặc biệt, em muốn gửi lời cảm ơn tới Cô Lê Thị Anh - Giảng viên trực tiếp hướng dẫn, chỉ bảo, giúp em khắc phục những khó khăn, thiếu sót để có thể hoàn thành các phần trong đồ án tốt nghiệp từ lý thuyết cho tới thực hành sử dụng công cụ.

Mặc dù đã cố gắng với tất cả nỗ lực của bản thân để hoàn thiện đồ án, nhưng do thời gian có hạn, năng lực và kinh nghiệm còn hạn chế nên đồ án không thể tránh khỏi những thiếu sót. Kính mong nhận được sự đóng góp ý kiến từ phía thầy cô, bạn bè để em có thể nâng cao kiến thức của bản thân, hoàn thiện đồ án được tốt hơn.

Em xin chân thành cảm ơn!

Hà Nội, ngày 21 tháng 12 năm 2024

sinh viên thực hiện

Huy

Nguyễn Lê Huy

[DANH MỤC HÌNH ẢNH 9](#_Toc8487)

[DANH MỤC TỪ VIẾT TẮT 10](#_Toc3239)

[CHƯƠNG 1: Tổng Quan Về Kiểm Thử 12](#_Toc2921)

[1.1. Khái quát về kiểm thử ứng dụng trên nền Web 12](#_Toc27344)

[1.1.1. Khái quát 12](#_Toc6980)

[1.1.2. Các loại ứng dụng web 12](#_Toc14554)

[1.1.3. Đặc điểm về chất lượng của một ứng dụng trên nền web 13](#_Toc17011)

[1.1.4. Khái niệm về ca thử nghiệm 15](#_Toc28939)

[1.2. Kiểm thử tự động và kiểm thử thủ công 18](#_Toc27595)

[1.2.1. Khái niệm 18](#_Toc28129)

[1.2.2. Đặc điểm 18](#_Toc16079)

[1.3. Công việc chính khi kiểm thử ứng dụng web 23](#_Toc31720)

[1.3.1. Kiểm thử chức năng 23](#_Toc21665)

[1.3.2. Kiểm thử khả năng sử dụng 25](#_Toc31579)

[1.3.3. Kiểm thử sự tương thích 26](#_Toc4523)

[1.3.4. Kiểm thử hiệu suất 27](#_Toc6062)

[1.3.5. Kiểm thử bảo mật 28](#_Toc12673)

[1.4. Một số công cụ hỗ trợ kiểm thử ứng dụng trên nền web 29](#_Toc3176)

[1.4.1. Công cụ kiểm thử hiệu năng 29](#_Toc16061)

[1.4.2. Công cụ kiểm thử bảo mật 30](#_Toc27146)

[1.4.3. Công cụ kiểm thử chức năng 31](#_Toc16112)

[CHƯƠNG 2. KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB BẰNG CÔNG CỤ SELENIUM 33](#_Toc8903)

[2.1. Giới thiệu về Selenium Webdriver 33](#_Toc23256)

[2.2. Các tính năng và chức năng chính của Selenium Webdriver 33](#_Toc9816)

[2.3. Quy trình làm việc của Selenium Webdriver 34](#_Toc19828)

[2.4. Mội số loại kiểm thử sử dụng Selenium Driver 35](#_Toc19772)

[2.4.1 Kiểm thử ứng dụng web 35](#_Toc30909)

[2.4.2 Kiểm thử cơ sở dữ liệu (database) 36](#_Toc5918)

[2.4.3 Kiểm thử bảo mật 38](#_Toc13569)

[2.4.4 Kiểm thử ứng dụng di động 38](#_Toc7477)

[2.4.5 Kiểm thử khả năng tương thích 39](#_Toc22579)

[2.4.6 Kiểm thử mạng 40](#_Toc24793)

[2.5. Giải quyết vấn đề Wait-time với Selenium Driver 40](#_Toc7348)

[2.5.1. Wait trong Katalon 40](#_Toc8108)

*[2.5.2.](#_Toc2906)* [Qiải quyết vấn đề liên quan đến Wait 41](#_Toc2906)

[2.5.3. Khi nào nên sử dụng lệnh Wait 42](#_Toc25085)

[2.6. Phát hiện các yếu tố với Xpath 43](#_Toc27332)

[2.6.1. Tổng quan về XPath 43](#_Toc23887)

[2.6.2. Xác định các yếu tố lồng nhau 43](#_Toc1596)

[2.7. So sánh các công cụ kiểm thử tự động 44](#_Toc21559)

[2.8. Tổng kết chương 47](#_Toc28853)

[CHƯƠNG 3. TRIỂN KHAI THỰC NGHIỆM KIỂM THỬ ỨNG DỤNG WEB 44](#_Toc9695)

[3.1. Triển khai thực nghiệm 44](#_Toc19621)

[3.1.1. Giới thiệu website 44](#_Toc4853)

[3.1.2. các chức năng chính của website 45](#_Toc10190)

*[3.1.2.1. Chức năng đăng nhập](#_Toc10966)* [45](#_Toc10966)

*[3.1.2.2. Chức năng đăng ký](#_Toc12853)* [47](#_Toc12853)

[3.1.2.3. chức năng xem chi tiết sản phẩm 50](#_Toc11048)

[3.1.2.4 chức năng thêm giỏ hàng 53](#_Toc4558)

[3.1.2.5 chức năng tìm kiếm 58](#_Toc24755)

[3.2. Triển khai kiểm thử chức năng chính bằng Selenium 63](#_Toc6216)

[3.2.1. Đăng ký 63](#_Toc4571)

[3.2.2. Đăng nhập 65](#_Toc17179)

[3.2.3. Xem chi tiết sản phẩm 68](#_Toc6308)

[3.2.4. Thêm giỏ hàng 68](#_Toc21145)

[3.2.5. Tìm kiếm sản phẩm 69](#_Toc20923)

[KẾT LUẬN 71](#_Toc30530)

[1. Kết quả đạt được 71](#_Toc24604)

[2. Hạn chế 71](#_Toc4782)

[3. Hướng phát triển 72](#_Toc1862)

[TÀI LIỆU THAM KHẢO 73](#_Toc18823)

[PHỤ LỤC 74](#_Toc5077)

[1. Mã script của các testcase 74](#_Toc28203)

[1.1. Register 74](#_Toc11803)

[1.2. SignUp 75](#_Toc20833)

**Mục Lục Hình ảnh:**

* **Hình 1.1**: Minh họa mẫu ca kiểm thử .......................................... 16
* **Hình 1.2**: So sánh phạm vi test tự động về rủi ro chất lượng sản phẩm ..... 19
* **Hình 1.3**: So sánh chi phí kiểm thử thủ công và kiểm thử tự động ......... 20
* **Hình 3.1**: Giao diện trang chủ website Trinnie Store ....................... 35
* **Hình 3.2**: Giao diện chức năng đăng nhập tài khoản của khách hàng ......... 36
* **Hình 3.3**: Giao diện chức năng đăng ký tài khoản của khách hàng .......... 38
* **Hình 3.4**: TestScript cho ca kiểm thử đăng ký nhưng thiếu địa chỉ ......... 40
* **Hình 3.5**: Thực thi ca kiểm thử đăng ký nhưng thiếu địa chỉ ............ 40
* **Hình 3.6**: TestScript cho ca kiểm thử đăng ký sai định dạng email ....... 41
* **Hình 3.7**: Thực thi ca kiểm thử đăng ký sai định dạng email ............ 41
* **Hình 3.8**: TestScript ca kiểm thử đăng ký với mật khẩu không đủ mạnh ... 41
* **Hình 3.9**: Thực thi ca kiểm thử đăng ký với mật khẩu không đủ mạnh .... 42
* **Hình 3.10**: TestScript cho ca kiểm thử đăng ký thành công ............. 42
* **Hình 3.11**: Thực thi ca kiểm thử đăng ký thành công ...................... 42
* **Hình 3.12**: TestScript cho ca kiểm thử đăng nhập với email sai ........ 43
* **Hình 3.13**: Thực thi ca kiểm thử đăng nhập với email sai ............... 43
* **Hình 3.14**: TestScript cho ca kiểm thử đăng nhập với password sai ...... 43
* **Hình 3.15**: Thực thi ca kiểm thử đăng nhập với password sai ......... 44
* **Hình 3.16**: TestScript cho ca kiểm thử đăng nhập thành công ........... 44
* **Hình 3.17**: Thực thi ca kiểm thử đăng nhập thành công .................. 44
* **Hình 3.18**: TestScript cho ca kiểm thử xem chi tiết sản phẩm .......... 68
* **Hình 3.19**: Xem chi tiết sản phẩm thành công ......................... 68
* **Hình 3.20**: TestScript cho ca kiểm thử thêm vào giỏ hàng ........... 69
* **Hình 3.21**: Thực thi thành công ca kiểm thử thêm vào giỏ hàng ...... 69
* **Hình 3.22**: TestScript cho ca kiểm thử tìm kiểm sản phẩm ............ 70
* **Hình 3.23**: Thực thi thành công ca kiểm thử tìm kiếm sản phẩm ...... 70

**DANH MỤC TỪ VIẾT TẮT**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **KÝ HIỆU** | **CỤM TỪ ĐẦY ĐỦ** | **Ý NGHĨA** |
| 1 | API | Application Programming  Interface | Giao diện lập trình ứng  dụng |
| 2 | CAPTCHA | Completely Automated  Public Turing test to tell  Computers and Humans  Apart | Trình kiểm tra tự động  phân biệt hành động của  máy tính với hành động  của người dùng |
| 3 | CSS | Cascading Style Sheets | Ngôn ngữ quy định cách  hiển thị của các phần tử  HTML |
| 4 | DOM | Document Object Model | Mô hình đối tượng tài  liệu |
| 5 | ERP | Enterprise Resource  Planning | Hệ thống hoạch định tài nguyên doanh nghiệp |
| 6 | HTML | HyperText Markup  Language | Ngôn ngữ đánh dấu siêu  văn bản |
| 7 | ISTQB | International Software  Testing Qualifications  Board | Tổ chức cung cấp chứng  chỉ kiểm thử phần mềm  có giá trị toàn cầu |
| 8 | RTM | Requirement Traceability  Matrix | Ma trận truy xuất nguồn  gốc các yêu cầu kiểm thử |
| 9 | SEO | Search Engine Optimization | Tối ưu hóa máy tìm kiếm |
| 10 | UI | User Interface | Giao diện người dùng |

**CHƯƠNG 1: Tổng Quan Về Kiểm Thử**

* 1. **Khái quát về kiểm thử ứng dụng trên nền Web**
     1. Khái quát

Khi mạng Internet ngày càng phát triển, môi trường mạng đem đến nhiều cơ hội kinh doanh, tiếp cận khách hàng thì hiển nhiên việc thiết kế website và các ứng dụng chạy trên nền Web là cần thiết để chiếm lĩnh thị trường. Các ứng dụng Web phát triển và đóng vai trò to lớn trong việc kết nối, trao đổi thông tin của nhiều doanh nghiệp.

Muốn có được sự thành công kể trên, trước hết các ứng dụng chạy trên nền Web phải có chất lượng tốt, hiệu năng cao, chưa kể tới các yếu tố về giao diện, trải nghiệm người dùng, v.v. Ngoài ra, chúng ta đều biết ứng dụng trên nền Web có những đặc thù khác biệt hoàn toàn so với ứng dụng di động, ứng dụng desktop, v.v. Ứng dụng trên nền Web không giới hạn chỉ ở điện thoại thông minh, máy vi tính hay máy tính bảng, mà được thiết kế để chạy trên nhiều nền tảng khác nhau. Mỗi nền tảng lại có những yêu cầu riêng về cấu hình, độ phân giải, đặc thù thao tác, v.v. Đó chính là những vấn đề lớn đặt ra cho các nhà phát triển phần mềm trong việc đảm bảo chất lượng cho các ứng dụng trên nền Web khi phải chạy trên đa nền tảng. Vì thế cần phải đưa ra một chiến lược hiệu quả cho kiểm thử, tránh những rủi ro, nâng cao chất lượng cho ứng dụng Web.

* + 1. Các loại ứng dụng web
* Ứng dụng Web tĩnh: Là loại ứng dụng Web hiển thị ít nội dung và không có tính linh hoạt. Ứng dụng Web tĩnh thường chỉ được xây dựng từ HTML, CSS và Javascript. Do không có cơ sở dữ liệu và công cụ điều khiển nội dung gián tiếp nên người quản trị không thể tùy ý thay đổi nội dung mà cần có kiến thức về HTML, CSS cơ bản để chỉnh sửa. Điểm cộng của loại website này là nội dung đơn giản, không mất nhiều thời gian, công sức để xây dựng do không phải xử lý những câu lệnh phức tạp. Tuy nhiên, do không có hệ thống hỗ trợ thay đổi nội dung nên việc cập nhật thông tin cho website gặp rất nhiều khó khăn, thậm chí phải bỏ ra chi phí lớn khi thay đổi 5 nhiều lần. Một số ví dụ cho Web tĩnh là những trang giới thiệu công ty, tổ chức, sản phẩm, khóa học ít có nhu cầu cập nhật thông tin, v.v.
* Ứng dụng Web động: So với web tĩnh thì Web động phức tạp hơn về mặt kỹ thuật khi xây dựng. Web động sử dụng cơ sở dữ liệu để hiển thị nội dung cũng như cho phép người dùng tương tác được với nội dung đó. Web động được chia làm 2 phần là back-end (dành cho người quản trị Web thay đổi, cập nhật nội dung) và front-end (dành cho người dùng truy cập). Hiện nay có rất nhiều ngôn ngữ lập trình được sử dụng để xây dựng Web động như Java, PHP, ASP.NET, VB.NET, Ruby, v.v. Đối với Web động, việc cập nhật nội dung là rất đơn giản và dễ dàng. Không những thế, một số hệ thống lớn hiện nay còn cho phép người quản trị có thể thay đổi giao diện Web trên trang quản trị mà không cần phải can thiệp trực tiếp vào mã nguồn. Đó là những lý do khiến cho Web động được sử dụng phổ biến hơn Web tĩnh.
  + 1. Đặc điểm về chất lượng của một ứng dụng trên nền web

Trước đây, kiểm thử phần mềm là lĩnh vực độc quyền của các ứng dụng desktop. Tuy nhiên, giờ đây nó đã trở thành một thuật ngữ bao gồm một loạt các nền tảng từ ứng dụng desktop, ứng dụng trên điện thoại thông minh, máy tính bảng cho đến ứng dụng chạy trên nền Web. Mỗi loại ứng dụng lại có những đặc trưng riêng về chất lượng, độ tin cậy, chức năng, môi trường cài đặt, yêu cầu người dùng, v.v. kéo theo việc kiểm thử trên từng loại ứng dụng sẽ khác nhau. Chính vì vậy, một chuyên gia về kiểm thử ứng dụng trên điện thoại thông minh hay ứng dụng desktop chưa chắc đã làm tốt công việc kiểm thử với ứng dụng trên nền Web. Những sự khác biệt có thể kể đến sau đây:

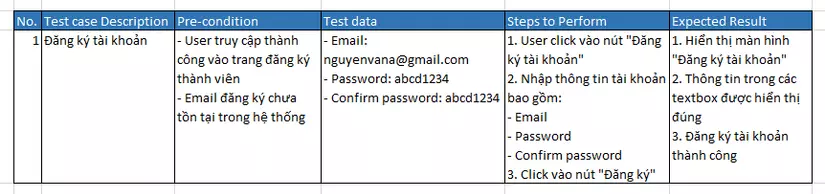
* *Ứng dụng trên nền Web sử dụng trên nhiều trình duyệt, không biết trước môi trường duyệt Web của người dùng:* Một ứng dụng Web chạy tốt trên trình duyệt Google Chrome nhưng trên Mozilla Firefox hay Safari thì có thể không như ý muốn. Đó là do mỗi trình duyệt được xây dựng trên kiến trúc khác nhau. Ngay cả khi hiện tại các trình duyệt đều đang cố gắng đưa ra chuẩn chung để dễ dàng hơn cho người lập trình, nhưng sự khác biệt khi khởi chạy ứng dụng trên nhiều trình duyệt khác nhau vẫn gây ra nhiều lo lắng cho lập trình viên và người làm kiểm thử. Đó là lý do chúng ta không khó bắt gặp những ứng dụng chạy trên nền Web ghi chú thích “Website chạy (tương thích) tốt nhất trên trình duyệt X v.v.”. Tuy nhiên, cách làm này không thật sự hiệu quả khi người dùng muốn sử dụng ứng dụng Web của chúng ta lại phải cài đặt trình duyệt được khuyến nghị. Để tránh cho sự bất tiện này đòi hỏi người làm kiểm thử phải triển khai ca kiểm thử trên nhiều trình duyệt khác nhau, kiểm tra độ tương thích và tìm ra những lỗi để lập trình viên đưa ra sự thay đổi cho phù hợp với mọi trình duyệt.
* *Ứng dụng trên nền Web thường có lượng truy cập lớn, nhiều người sử dụng trên cùng một thời điểm:* Với những ứng dụng Web có lượng người truy cập trung bình hoặc ít thì điều này không xảy ra vấn đề gì nghiêm trọng. Nhưng với những ứng dụng chạy trên nền Web có lượng người truy cập lớn, thực hiện nhiều thao tác truy vấn dữ liệu cùng lúc có thể sẽ dẫn tới việc server bị quá tải. Kiểm thử hộp trắng phát huy hiệu quả rất cao trong trường hợp này. Việc kiểm thử mã nguồn chương trình sẽ giúp loại bỏ được những dòng lệnh không hợp lý, gây tiêu tốn tài nguyên hệ thống và giúp cho ứng dụng Web có thể đáp ứng được lượng truy cập lớn cùng lúc tốt hơn. Công việc này cũng chính là kiểm thử hiệu năng, độ chịu lỗi của chương trình phần mềm.
* *Sự phụ thuộc vào tốc độ và sự ổn định của đường truyền Internet:* Đa số các ứng dụng Web đều cần sử dụng mạng Internet để tải các dữ liệu về, sau đó hiển thị lên trình duyệt. Nếu tốc độ đường truyền ổn định, việc duyệt Web không gây khó khăn gì. Tuy nhiên trên thực tế, tốc độ cũng như sự ổn định về đường truyền của người dùng là rất khó đoán biết, mỗi khu vực lại có sự khác nhau về đường truyền gây ảnh hưởng tới sự vận hành của ứng dụng Web. Chưa kể tới việc mạng có thể mất kết nối đột ngột khi đang thực hiện thao tác truy vấn sẽ dẫn tới những hậu quả rất khó lường nếu kiểm thử không tốt ở các trường hợp này, điển hình như các ứng dụng cho ngân hàng, hệ thống ERP, phần mềm phục vụ kế toán, v.v.
* *Sự cần thiết của SEO Web:* Đối với rất nhiều ứng dụng trên nền Web việc tối ưu SEO là một yêu cầu bắt buộc. Người sở hữu các website đều muốn website được thăng thứ hạng cao trên các máy tìm kiếm như Google, Bing, v.v. giúp ứng dụng Web của mình được nhiều người biết tới. Đây là một điểm mạnh giúp quảng bá ứng dụng trên nền Web dễ dàng hơn so với ứng dụng di động hay ứng dụng desktop. Trong thực thế, ngoài kiểm thử chức năng, hiệu năng, giao diện cho ứng dụng Web, kiểm thử viên còn phải chú trọng tới việc kiểm tra tối ưu SEO cho ứng dụng. Tuy nhiên việc tối ưu SEO lại không hề dễ dàng khi các máy tìm kiếm thường xuyên thay đổi thuật toán. Ngoài ra, nó còn liên quan tới chất lượng nội dung của ứng dụng Web để được máy tìm kiếm chú ý đến.
  + 1. Khái niệm về ca thử nghiệm

Ca kiểm thử là một khái niệm không thể thiếu trong kiểm thử phần mềm. Theo ISTQB “ca kiểm thử là một tập hợp các giá trị đầu vào, tiền điều kiện, các kết quả mong đợi và điều kiện kết thúc, được xây dựng cho mục đích hoặc điều kiện kiểm thử riêng biệt để kiểm tra tính đúng đắn của chương trình với yêu cầu của bản đặc tả yêu cầu phần mềm”. Hay nói cách khác, ca kiểm thử mô tả dữ liệu bao gồm: đầu vào, hành động hoặc sự kiện và kết quả đầu ra mong đợi (expected results) để xác định liệu 1 ứng dụng, hệ thống phần mềm hoặc một trong các tính năng của nó có hoạt động đúng như mong muốn hay không.

Cấu trúc của một ca kiểm thử thông thường bao gồm:

* *Test case ID:* Xác định số lượng trường hợp cần kiểm thử.
* *Function (Chức năng):* Các function có thể được chia nhỏ dựa theo chức năng của hệ thống nhằm giúp ca kiểm thử trở nên rõ ràng hơn.
* *Pre-condition:* Điều kiện đầu vào của ca kiểm thử, ví dụ như khi thực hiện kiểm thử form đăng nhập, pre-condition sẽ là form đăng nhập phải được hiển thị ra.
* *Test Data*: Dữ liệu đầu vào cần chuẩn bị trước khi kiểm thử.
* *Test Steps*: Mô tả chi tiết các bước thực hiện kiểm thử.
* *Expected Results:* Kết quả mong đợi sau khi thực hiện các bước kiểm thử.
* *Actual result*: Mô tả kết quả thực tế khi thực hiện kiểm thử trên môi trường của hệ thống. Actual result thường bao gồm ba giá trị: pass, fail và pending.
* *Comments:* Có thể chứa screenshot hoặc thông tin liên quan khi thực hiện ca kiểm thử.

Ngoài ra có thể có thêm một số cột như: Designed by (người thực hiện kiểm thử), Execute Date (ngày thực hiện kiểm thử), v.v. Mức độ chi tiết của ca kiểm thử sẽ phụ thuộc vào từng dự án và quy mô của công ty sản xuất phần mềm.



*Hình 1.1. Minh họa mẫu ca kiểm thử*

Một ca kiểm thử được cho là hiệu quả khi:

* Dựa vào ca kiểm thử có thể tìm thấy lỗi.
* Tìm được nhiều lỗi khó phát hiện.
* Chỉ ra được những điểm ban đầu mà khi thực hiện kiểm thử không tìm ra vấn đề.
* Ca kiểm thử cần có những bước thực hiện kiểm thử (Test steps) đơn giản, minh bạch, dễ hiểu.
* Các trường hợp thử nghiệm nên có giá trị, tóm tắt và ngắn.
* Các ca kiểm thử nên có sự liên kết: Mỗi ca kiểm thử cần được đánh số thứ tự (Test case ID) để đảm bảo ca kiểm thử đã bao phủ 100% bản đặc tả yêu cầu phần mềm.
* Ca kiểm thử có thể bảo trì: Nên viết ca kiểm thử sao cho khi có thay đổi, chỉnh sửa thì các bên liên quan có thể dễ dàng nhận thấy được sự thay đổi đó.
* Ca kiểm thử có tính ứng dụng cao.

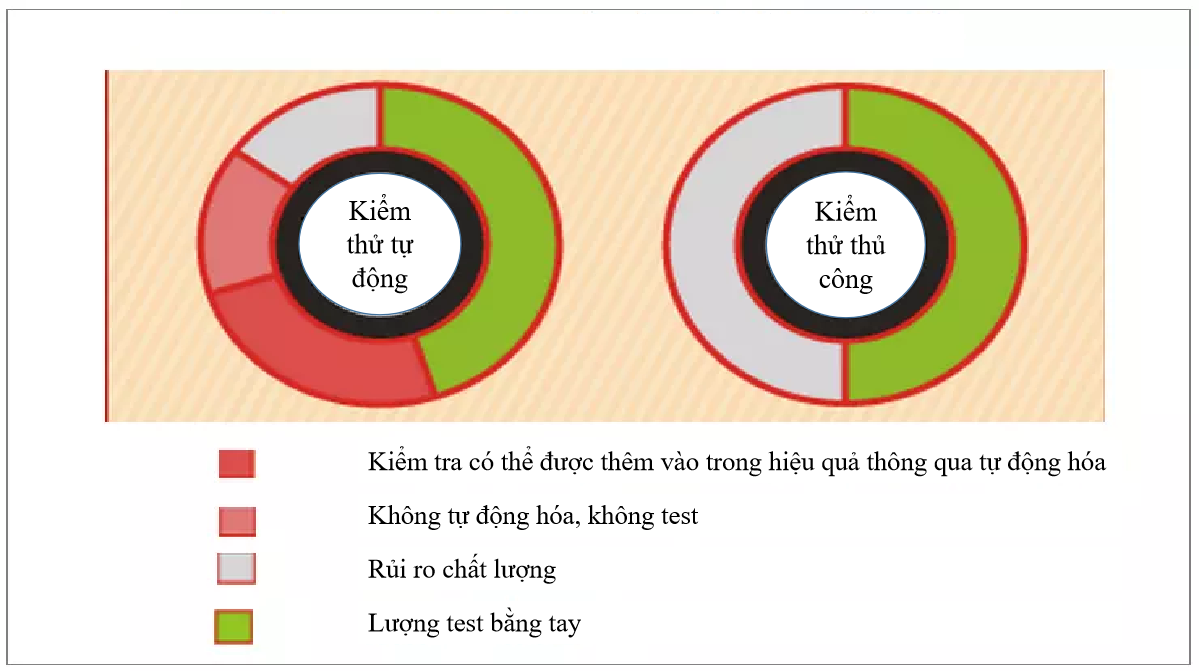
Tóm lại, ca kiểm thử được viết ra để kiểm tra hoạt động của các chức năng có đúng như mong muốn trong bản đặc tả yêu cầu phần mềm hay không. Khi viết ca kiểm thử nên cố gắng viết đơn giản, dễ hiểu nhưng phải đầy đủ các dữ liệu chuẩn cần có của một ca kiểm thử.

* 1. **Kiểm thử tự động và kiểm thử thủ công**
     1. Khái niệm
* Kiểm thử phần mềm tự động là thực hiện kiểm thử phần mềm bằng một chương trình đặc biệt với rất ít hoặc không có sự tương tác của con người, giúp cho người thực hiện việc kiểm thử phần mềm (tester) không phải lặp đi lặp lại các bước nhàm chán. Công cụ kiểm thử tự động có thể lấy dữ liệu từ file bên ngoài (excel, csv…) nhập vào ứng dụng, so sánh kết quả mong đợi (từ file excel, csv…) với kết quả thực tế và xuất ra báo cáo kết quả kiểm thử.
* Kiểm thử thủ công: là tester làm mọi công việc hoàn toàn bằng tay, từ viết testcase đến thực hiện test, mọi thao tác như nhập điều kiện đầu vào, thực hiện một số sự kiện khác như click nút và quan sát kết quả thực tế, sau đó so sánh kết quả thực tế với kết quả mong muốn trong test case, điền kết quả test. Hiện nay, phần lớn các tổ chức, các công ty phần mềm, hoặc các nhóm làm phần mềm đều thực hiện kiểm thử thủ công là chủ yếu.
  + 1. Đặc điểm

Ưu nhược điểm của hai quy trình:

*Bảng 1. 1 So sánh ưu nhược của kiểm thử thủ công và kiểm thử tự động*

|  |  |  |
| --- | --- | --- |
| **Kiểm thử** | **Điểm mạnh** | **Điểm yếu** |
| **Thủ công** | - Cho phép tester thực hiện việc kiểm thử khám phá.  - Thích hợp kiểm tra sản phẩm lần đầu tiên.  - Thích hợp kiểm thử trong trường hợp các test case chỉ phải thực hiện một số ít lần.  - Giảm được chi phí ngắn hạn. | - Tốn thời gian. Đối với mỗi lần release, người kiểm thử vẫn phải thực hiện lại một tập hợp các test case đã chạy dẫn đến sự mệt mỏi và lãng phí effort. |
| **Tự động** | - Thích hợp với trường hợp phải test nhiều lần cho một case, có tính ổn định và tin cậy cao hơn so với kiểm thử thủ công.  - Có thể thực hiện các thao tác lặp đi lặp lại (nhập dữ liệu, click, check kết quả...) giúp tester không phải làm những việc gây nhàm chán và dễ nhầm lẫn như vậy.  - Giảm chi phí đầu tư dài hạn. | - Tốn kém hơn kiểm thử tự động, chi phí đầu tư ban đầu lớn.  - Kiểm thử thủ công là không thể thay thế vì người ta không thể tự động hóa mọi thứ. |

**

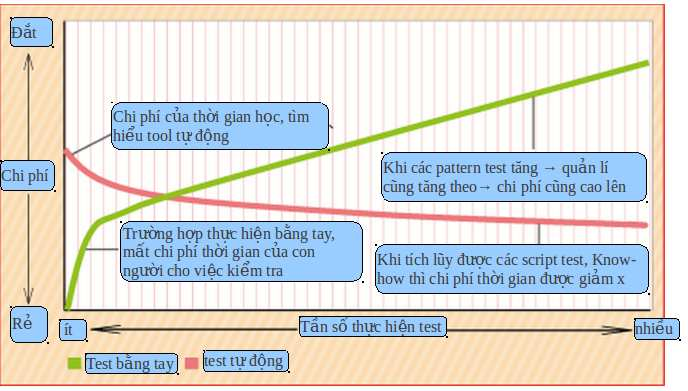
*Hình 1.2. So sánh phạm vi test tự động về rủi ro chất lượng sản phẩm*

Theo như bản đồ bên trên chỉ rõ tổng lượng kiểm tra (số hạng mục x thời gian) của test tự động và test bằng tay. Đối với test tự động có thể nâng cao được hiệu quả của việc kiểm tra, tổng số trường hợp kiểm tra chắc chắn được tăng lên.

* **Các kiểm nghiệm phù hợp và không phù hợp cho tự động hóa**

Nhìn vào bản đồ dưới đây ta có thể so sánh sự thiệt hơn giữa kiểm thử tự động và kiểm thử thủ công

Chi phí ban đầu cho kiểm thử tự động cao hơn so với kiểm thử bằng tay nhưng theo thời gian thì giá trị này bị đổi ngược lại. Tuy lợi ích mang lại là không nhỏ nhưng kiểm thử tự động cũng không hoàn toàn thay thế được kiểm thử bằng tay. Kiểm tra không có lợi ích của tự động hóa có nghĩa là dù có tự động hóa thì cũng không có ý nghĩa. Khi không mang lại ý nghĩa thì sẽ mang lại một kết quả không khả quan.



*Hình 1.3. So sánh chi phí kiểm thử thủ công và kiểm thử tự động*

* Kiểm tra thích hợp cho tự động hóa (hiệu quả mang lại lớn)
* Với những loại dưới đây thích hợp cho tự động hóa:
* Những kiểm tra cần thực hiện nhiều lần
* Thực hiện kiểm tra ở nhiều môi trường
* Đặc điểm kỹ thuật được xác định, test màn hình, chức năng không thay đổi trong tương lai.
* Thường xuyên thực hiện test xác nhận hoạt động cơ bản (chẳng hạn như di chuyển hệ thống)
* Test sự kết hợp của nhiều giá trị đầu vào ở một bước nào đó
* Kiểm tra nhiều màn hình của dữ liệu đầu vào
* Mục đầu vào ở nhiều màn hình đăng ký
* Kiểm thử không thích hợp cho tự động hóa (Không mang lại hiệu quả)
* Kiểm tra không có tính hồi quả
* Kiểm tra những hoạt động như test độ tin cậy, giới hạn, cạnh tranh…
* Việc thực hiện tự động không phải là ứng dụng cho tất cả các trường hợp test không thể tự động hóa cho tất cả các trường hợp thử nghiệm. Với nhiều trường hợp test không yêu cầu hồi quy, đặc điểm kỹ thuật luôn thay đổi thì tự động hóa không mang lại chút hiệu quả nào.

Nên sử dụng kiểm thử thủ công và thử nghiệm tự động khi:

* Kiểm thử thủ công là phù hợp nhất với các khu vực / kịch bản sau đây:
* Kiểm thử thăm dò: Đây là loại kiểm thử đòi hỏi phải thử nghiệm của kiến thức, kinh nghiệm, phân tích / logic kỹ năng, sáng tạo và trực giác. Xét nghiệm này được đặc trưng bởi các tài liệu ở đây kém bằng văn bản kỹ thuật, hoặc một thời gian ngắn để thực hiện. Chúng ta cần những kỹ năng của con người để thực hiện quá trình kiểm thử trong kịch bản này.
* Usability Testing: Đây là một lĩnh vực mà bạn cần để đo độ thân thiện, hiệu quả, hoặc thuận tiện phần mềm hoặc sản phẩm cho người dùng cuối. Ở đây, quan sát con người là yếu tố quan trọng nhất, do đó, một phương pháp thủ công là một lợi thế.
* Kiểm thử Ad-hoc: Trong kịch bản này, không có phương pháp cụ thể. Nó là một phương pháp hoàn toàn không có kế hoạch kiểm thử nơi sự hiểu biết và cái nhìn sâu sắc của các thử nghiệm là yếu tố quan trọng duy nhất.
* Kiểm thử tự động là phù hợp nhất với các khu vực / kịch bản sau đây:
* Kiểm thử hồi quy: Ở đây, kiểm thử tự động là phù hợp vì các thay đổi mã thường xuyên và khả năng chạy các hồi quy một cách kịp thời.
* Kiểm thử tải trọng: tự động kiểm thử cũng là cách tốt nhất để hoàn thành các thử nghiệm có hiệu quả khi nó đi kèm để tải thử nghiệm.
* Thực hiện lặp đi lặp lại: Thử nghiệm mà đòi hỏi phải thực hiện lặp đi lặp lại một công việc được tự động tốt nhất.
* Thử nghiệm tính năng: Tương tự như vậy, thử nghiệm mà đòi hỏi sự mô phỏng của hàng ngàn người dùng đồng thời đòi hỏi tự động hóa.
  1. **Công việc chính khi kiểm thử ứng dụng web**
     1. Kiểm thử chức năng

Kiểm thử chức năng yêu cầu kiểm thử viên thực hiện kiểm thử tất cả các link trong trang Web, định dạng được sử dụng trong các trang Web để gửi và nhận các thông tin cần thiết từ người dùng. Ngoài ra còn có kết nối cơ sở dữ liệu, kiểm tra cookie và xác minh HTML/CSS, v.v.

* *Kiểm thử giao diện:*

Trước khi lập trình viên bắt tay vào xây dựng mã nguồn sẽ luôn có một bản thiết kế UI quy định giao diện của ứng dụng Web. Mỗi thành phần textbox, button, image, link và bố cục trên ứng dụng Web đều được chỉ ra một cách cụ thể trong tài liệu này. Bản thiết kế UI thường sẽ kèm theo file .PSD – bản vẽ giao diện của ứng dụng Web sử dụng phần mềm Photoshop để lập trình viên dễ dàng xây dựng cũng như khách hàng có thể biết trước ứng dụng của mình sẽ hiển thị ra sao. Đây cũng là tài liệu không thể thiếu cho kiểm thử viên so sánh, đối chiếu giữa thiết kế và nội dung thực tế của ứng dụng hiển thị trên trình duyệt.

* *Kiểm thử các liên kết và menu:*

Trong một ứng dụng Web có 2 loại liên kết: liên kết nội bộ (internal link) và liên kết ngoại bộ (external link). Cả 2 loại liên kết trên đều cần được kiểm tra xem chúng có hoạt động không? Có trỏ đến địa chỉ mong muốn không? Cần đảm bảo rằng các liên kết không tự trỏ đến vị trí của chính nó. Ngoài ra cũng cần xem xét thuộc tính “target” của các liên kết xem chúng có hoạt động đúng như bản thiết kế yêu cầu hay không.

* K*iểm thử các form nhập dữ liệu:*

Cần đảm bảo các trường nhập liệu được thiết kế đúng kiểu loại, có bộ lọc kiểm tra tính đúng đắn của dữ liệu nhập vào (validation) trước khi gửi đi (submit) tránh việc hacker có thể tận dụng lỗ hổng SQL Injection từ chính các form nhập liệu trên ứng dụng Web. Ngoài ra cũng cần đảm bảo sự toàn vẹn dữ liệu trong quá trình truyền tải thông tin từ trình duyệt tới server, nhất là đối với các ứng dụng thương mại điện tử, ngân hàng, v.v.

* *Kiểm thử lỗi cú pháp HTML/CSS:*

Ở bước tiếp theo, người kiểm thử cần xác định các thẻ CSS bị lỗi hoặc các thuộc tính, id, class được viết trong thẻ HTML không hợp lệ hoặc không thuộc bất kỳ thẻ CSS nào.

* *Kiểm thử cookie và session:*

Kiểm thử các ứng dụng đăng nhập trong phiên bằng cách cho phép và vô hiệu hóa các tập tin cookie. Có thể thử đưa lỗi vào ứng dụng Web bằng cách sử dụng một tên miền không phù hợp như cố tình truyền sai, thiếu tham số, v.v. Ngoài ra, cần kiểm tra khả năng bảo mật của ứng dụng Web bằng cách xóa các tập tin cookie có chọn lọc khi kiểm thử.

* *Kiểm thử nội dung đa ngôn ngữ:*

Bước kiểm thử này đặc biệt cần thiết với những ứng dụng Web hỗ trợ đa ngôn ngữ để đảm bảo thông tin khi dịch sang các ngôn ngữ khác nhau luôn được sát nghĩa, không bị tràn dòng khi dịch, các yếu tố về chính tả được tuân thủ.

* *Kiểm thử cơ sở dữ liệu (database):*

Kiểm tra kết nối tới cơ sở dữ liệu và các lỗi truy vấn có thể gặp phải, đảm bảo dữ liệu được cung cấp chính xác khi các chức năng xem thông tin, thêm, sửa, xóa, v.v. hoạt động.

* + 1. Kiểm thử khả năng sử dụng
* *Kiểm thử nội dung:*

Chúng ta cần đảm bảo nội dung trong ứng dụng được sắp xếp hợp lý và dễ hiểu với người dùng, không mắc các lỗi chính tả, các hình ảnh hiển thị chính xác về vị trí, kích thước. Ngoài ra cũng cần chú trọng tới màu sắc, font chữ phù hợp với mọi đối tượng sử dụng.

* *Kiểm thử logic các liên kết và hướng dẫn:*

Đối với người dùng lần đầu tiên truy cập một ứng dụng Web, họ luôn gặp những khó khăn nhất định trong việc sử dụng. Vì vậy cần kiểm tra xem các hướng dẫn, liên kết, thông báo đã được bố trí đầy đủ trên ứng dụng hay chưa? Tuy nhiên, việc xuất hiện quá nhiều hướng dẫn, thông báo ở mọi nơi trên ứng dụng cũng khiến người dùng rối mắt, không thoải mái khi sử dụng. Tốt nhất nên đảm bảo các hướng dẫn, thông báo đưa ra hết sức ngắn gọn nhưng đủ ý ngay tại nơi người dùng có thể gặp khó khăn khi sử dụng.

* *Kiểm thử văn hoá khu vực và đối tượng sử dụng:*

Điều này bắt nguồn từ đặc điểm riêng của từng lĩnh vực (ví dụ y khoa thường dùng màu sáng để thể hiện sự sạch sẽ), hoặc văn hóa riêng từng khu vực (người châu Á thường chuộng tông màu nóng và thiết kế cầu kỳ hơn châu Âu).

Thêm vào đó, trong quá trình kiểm thử phải luôn bảo đảm rằng chuẩn thiết kế ứng dụng Web của mình có thể được tìm thấy phổ biến ở nhiều ứng dụng Web khác cùng loại. Ví dụ như button Đăng nhập, Đăng xuất thường nằm ở góc trên bên phải và menu chính luôn nằm ở trên cho tất cả trang Web con. Nếu một ứng dụng Web trong lĩnh vực khoa học lại trình bày bằng font chữ cách điệu lòe loẹt, tiêu đề chạy ngang dọc, hoặc một ứng dụng Web dành cho trẻ em lại chỉ dùng 2 tông màu đen trắng buồn tẻ thì nên góp ý với bộ phận thiết kế.

* + 1. Kiểm thử sự tương thích

Một ứng dụng Web thường hỗ trợ nhiều thiết bị, môi trường khác nhau. Vì vậy kiểm thử độ tương thích của ứng dụng Web là một điều không dễ dàng khi công nghệ của các nền tảng thay đổi quá nhanh chóng.

* *Kiểm thử tương thích theo thiết bị, hệ điều hành:*

Khó có ứng dụng Web nào chạy hoàn hảo trên tất cả các môi trường, vì vậy người kiểm thử cần đặt ưu tiên cho những môi trường cần hỗ trợ để tiết kiệm thời gian cho việc kiểm thử.

Có hai điều cần lưu tâm nhất khi kiểm thử khả năng tương thích của ứng dụng với thiết bị, đó là: khung hình và khả năng hỗ trợ của thiết bị với các phiên bản HTML. Người kiểm thử cần truy cập tất cả các nội dung trên từng loại thiết bị, có thể xoay ngang, dọc màn hình (đối với thiết bị di động, máy tính bảng) để xem ứng dụng Web được hiển thị như thế nào, chạy thử từng chức năng trên ứng dụng để đảm bảo chúng hoạt động như mong muốn.

* *Kiểm thử tương thích với trình duyệt:*

Cần chạy thử ứng dụng trên một số trình duyệt phổ biến hiện nay như IE, Chrome, Firefox, Opera, Safari, v.v. để đảm bảo hoạt động chính xác trên các trình duyệt khác nhau.

Kiểm tra hoạt động các chức năng của ứng dụng khi thực hiện cài đặt, cấu hình bảo mật cho trình duyệt.

Mỗi trình duyệt lại có nhiều phiên bản cập nhật khác nhau, cần kiểm tra sự nhất quán của ứng dụng khi chạy trên các phiên bản đó.

Kiểm tra hoạt động của ứng dụng khi bật/tắt flash, cookie, java, v.v.

* + 1. Kiểm thử hiệu suất
* *Kiểm thử khả năng tải (Load test):*

Ở bước này cần xác định thời gian thực thi cho các hành động tương ứng với các chức năng trên ứng dụng. Công việc này cần được thực hiện ở nhiều thời điểm khác nhau (giờ cao điểm/thấp điểm) để có những đánh giá khách quan nhất về khả năng tải của ứng dụng.

* *Kiểm thử độ chịu lỗi (Stress test):*

Công việc này chính là kiểm tra sức chịu đựng của ứng dụng Web khi có lượng truy cập cao từ phía người dùng. Trong thực tế đó có thể là nhu cầu sử dụng thực sự của người dùng đối với ứng dụng hoặc khi máy chủ bị tấn công dưới dạng Ddos. Nói cách khác, người kiểm thử cần trả lời câu hỏi: Số lượng người truy cập cùng lúc là bao nhiêu sẽ đánh sập hệ thống? Hay đơn giản hơn là khi lượng người truy cập tăng lên ở các mức khác nhau, ứng dụng còn hoạt động ổn định hay không? Trả lời được các câu hỏi trên sẽ giúp cho ứng dụng Web khi đưa vào hoạt động tránh được những rủi ro không đáng có và lường trước những nguy cơ có thể xảy ra.

* + 1. Kiểm thử bảo mật

Ứng dụng Web là một trong những loại ứng dụng có nguy cơ bị tấn công cao nhất. Vì vậy, ngoài việc đảm bảo ứng dụng chạy đúng, ổn định cần phải kiểm tra nghiêm ngặt khả năng bảo mật của ứng dụng. Các công việc cần làm có thể kể đến như:

Kiểm tra độ tin cậy của việc phân quyền sử dụng trên ứng dụng.

Đưa lỗi vào bằng cách truyền các tham số không hợp lệ trên URL hay trong các form nhập liệu. Lỗ hổng SQL Injection được khai thác mạnh nhất thông qua các thành phần trên.

Kiểm tra khả năng truy cập trái phép đối với những thư mục bị cấm trên máy chủ của ứng dụng.

Kiểm tra hoạt động các bộ lọc (validation) khi sử dụng chức năng upload tệp tin, thư mục của ứng dụng (nếu có).

Kiểm tra độ xác thực khi nhập CAPTCHA trong ứng dụng (nếu có).

* 1. **Một số công cụ hỗ trợ kiểm thử ứng dụng trên nền web**

Công việc cần làm đối với một kiểm thử viên kiểm thử ứng dụng trên nền Web như đã nói ở phần trước là rất nhiều. Những công cụ kiểm thử ra đời để hỗ trợ cho các kiểm thử viên thực hiện công việc một cách nhanh chóng, bớt nhàm chán và giảm thiểu chi phí kiểm thử. Đồ án này sẽ giới thiệu một số công cụ hỗ trợ kiểm thử ứng dụng trên nền Web, phân loại dựa trên mục đích sử dụng.

* + 1. Công cụ kiểm thử hiệu năng

Dưới đây là danh sách một số công cụ kiểm thử hiệu năng được sử dụng rộng rãi nhất để đo hiệu suất ứng dụng Web và khả năng chịu tải của chúng. Các công cụ kiểm tra tải này sẽ đưa ra đánh giá về hiệu suất của ứng dụng trong thời gian có lưu lượng truy cập cao điểm.

* WebLoad: Cho phép thực hiện kiểm thử khả năng chịu tải và độ chịu lỗi của ứng dụng Web bằng cách sử dụng Ajax, Adobe Flex, .NET, Oracle. Forms, HTML5 và nhiều công nghệ khác. Điểm mạnh của WebLoad là là dễ sử dụng với các tính năng như cho phép ghi/phát lại dựa trên DOM, tương quan tự động và ngôn ngữ kịch bản Javascript. Công cụ này hỗ trợ thử nghiệm hiệu suất quy mô lớn với các kịch bản phức tạp và đưa ra những phân tích rõ ràng.
* Apache JMeter: Đây là một công cụ phát triển trên mã nguồn mở. Apache Jmeter được coi như một công cụ kiểm thử hiệu năng, có khả năng tích hợp với kế hoạch kiểm thử. Ngoài việc kiểm thử hiệu năng, Apache JMeter còn có thể sử dụng để kiểm tra các chức năng của ứng dụng Web.
* NeoLoad: Công cụ sử dụng để đo và phân tích hiệu suất của ứng dụng Web. NeoLoad phân tích hiệu suất của ứng dụng Web bằng cách tăng lưu lượng truy cập vào ứng dụng. Nhờ đó, kiểm thử viên có thể biết được năng lực chịu tải của ứng dụng. Công cụ này được viết trên nền Java, tương thích với nhiều hệ điều hành khác nhau và hỗ trợ hai ngôn ngữ: Tiếng Anh và tiếng Pháp.
* LoadStorm: Là một công cụ kiểm thử cho các ứng dụng Web và mobile. Điểm mạnh là nó có thể kiểm tra hiệu năng của ứng dụng dựa trên số lượng người dùng và lưu lượng truy cập. LoadStorm cũng có khả năng chịu tải rất tốt khi mà nó có thể giả lập hàng trăm nghìn đến hàng triệu user để tìm kiếm các breaking point (điểm dừng) trong ứng dụng. Các kịch bản kiểm thử của LoadStorm có thể được chỉnh sửa bởi kiểm thử viên.
  + 1. Công cụ kiểm thử bảo mật
* *Burp Suite*: Là một công cụ kiểm tra lỗ hổng bảo mật cho ứng dụng Web. Nó có nhiều công cụ tích hợp trong đó hai công cụ chính trong phiên bản miễn phí là Spider and Intruder. Spider được sử dụng để thu thập thông tin các trang của ứng dụng và Intruder được sử dụng để thực hiện các cuộc tấn công tự động trên ứng dụng Web. Burp có một công cụ bổ sung hiện nay được gọi là Burp Scanner được dùng trong việc quét các lỗ hổng có trong ứng dụng.
* *OWASP Zed Attack Proxy*: Tương tự như Burp Suite, OWASP Zed Attack Proxy là công cụ để thâm nhập, đánh giá an ninh mạng, bảo mật của các ứng dụng Web.
* *Nikto:* Công cụ đánh giá hệ thống Nikto là một máy quét lỗ hổng máy chủ Web mã nguồn mở. Nó phát hiện việc cài đặt phần mềm và cấu hình đã lỗi thời, các tệp tin có khả năng nguy hiểm, v.v.
* *Exploit-Me*: Là một công cụ kiểm tra bảo mật ứng dụng Web có thể tích hợp trên trình duyệt Firefox được thiết kế nhỏ gọn, dễ sử dụng. Exploit- Me bao gồm các gói: XSS-Me và SQL Inject-Me. Cross-Site Scripting (XSS) là một lỗ hổng được tìm thấy trong nhiều ứng dụng Web hiện nay. Lỗ hổng XSS có thể gây ra thiệt hại nghiêm trọng cho một ứng dụng Web. XSS-Me là công cụ giúp phát hiện ra các lỗ hổng XSS này. Trong khi đó, SQL Inject-Me được sử dụng để kiểm tra các lỗ hổng SQL Injection trong ứng dụng Web.
  + 1. Công cụ kiểm thử chức năng
* *Ranorex*: Công cụ kiểm thử tự động cho các ứng dụng Web, desktop và di động. Chỉ với một tài khoản, người dùng có thể sử dụng Ranorex để kiểm thử cho 3 loại ứng dụng kể trên. Việc tích hợp này sẽ giúp rút ngắn thời gian khi kiểm thử ứng dụng được thiết kế chạy trên nhiều nền tảng khác nhau. Tuy nhiên, bản trả phí của Ranorex khá đắt, lên tới 3500$/năm.
* *Selenium:* Là một trong những công cụ kiểm thử tự động ứng dụng Web mạnh mẽ nhất hiện nay. Selenium script có thể chạy trên hầu hết các trình duyệt hiện nay như IE, Chrome, Firefox, Safari, Opera và các hệ điều hành phổ biến như Windows, Mac, Linux. Trong thực tế, người ta thường sử dụng Selenium dưới dạng Add-on tích hợp trong trình duyệt Firefox, kết hợp cùng với Firebug để kiểm thử ứng dụng Web một cách hiệu quả nhất. Tuy chỉ có thể ghi lại (Record) hành động trên trình duyệt Firefox, nhưng có thể phát lại (Playback) trên nhiều trình duyệt phổ biến khác. Vì là công cụ mã nguồn mở nên Selenium có ưu thế lớn so với các công cụ kiểm thử tự động khác: có cộng đồng hỗ trợ mạnh mẽ và không phải trả phí bản quyền. Công cụ này hỗ trợ khá nhiều ngôn ngữ lập trình Web phổ biến hiện nay. Ngoài ra, Selenium được phát triển bởi Selenium team từ Google nên người dùng hoàn toàn yên tâm về chất lượng và độ tin cậy của Selenium.
* *Selenium Driver:* cũng như Selenium - là một công cụ miễn phí dành cho cộng đồng kiểm thử phần mềm tự động. Nếu Selenium Driver được xây dựng từ Selenium/Appium, tại sao chúng ta không dùng trực tiếp hai công cụ phổ biến trong cộng động kiểm thử này mà cần phải dùng đến Selenium Driver.

**CHƯƠNG 2. KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB BẰNG CÔNG CỤ SELENIUM**

**2.1. Giới thiệu về Selenium Webdriver**

Selenium Webdriver được coi là một trong những công cụ kiểm thử tự động mạnh mẽ và thông dụng nhất của bộ công cụ Selenium. Là khung tự động hóa trình duyệt của Selenium. Nó hỗ trợ một số ngôn ngữ lập trình như C#, Java, Perl, Ruby, PHP, JavaScript và Python. Khung tự động hóa này được thực hiện thông qua một trình điều khiển dành riêng cho trình duyệt và nó có thể giao tiếp trực tiếp với các trình duyệt và điều khiển chúng. Selenium Webdriver thực hiện automate tương tác với trình duyệt với hướng tiếp cận hiện đại và ổn định.

**2.2. Các tính năng và chức năng chính của Selenium Webdriver**

* WebDriver cho phép người dùng thực hiện kiểm thử tự động dựa trên web. WebDriver là một công cụ khác hoàn toàn và có những ưu điểm vượt trội so với Selenium RC.
* WebDriver hỗ trợ nhiều trình duyệt web, ngôn ngữ lập trình và môi trường kiểm thử.
* WebDriver giao tiếp trực tiếp với trình duyệt web và sử dụng khả năng tương thích riêng của trình duyệt để kiểm thử tự động.
* Khả năng hỗ trợ của WebDriver không chỉ giới hạn trong các hoạt động ngoại vi của người dùng truyền thống. Thay vào đó, nó còn hỗ trợ các cơ chế xử lý hiệu quả cho các hành động phức tạp của người dùng như xử lý dropdown, gọi Ajax, chuyển đổi giữa các cửa sổ, điều hướng, xử lý cảnh báo, v.v.
* WebDriver cho phép người dùng thực hiện kiểm thử mobile dựa trên web. Để hỗ trợ tương tự, WebDriver giới thiệu AndroidDriver và IphoneDriver.
* WebDriver nhanh hơn các công cụ khác của Selenium Suite vì nó thực hiện cuộc gọi trực tiếp tới trình duyệt mà không cần bất kỳ sự can thiệp nào từ bên ngoài.

**2.3. Quy trình làm việc của Selenium Webdriver**

Quy trình làm việc của Selenium webdriver gồm 3 bước cơ bản: viết kịch bản (hay còn gọi là script), kiểm tra kịch bản và chạy kịch bản.

Bước đầu tiên cần làm là viết script. Người viết sẽ tạo trước các hành động và lưu trữ nó trong thư viện. Khi làm việc với trình duyệt web, người viết sẽ xác định một số yếu tố như đối tượng và vị trí mà hành động đó cần thực hiện.

Sau đó xác định hết, người viết sẽ tạo một danh sách trình tự các hành động sẽ thực hiện trên trình duyệt web đó với đối tượng và vị trí xác định. Và cuối cùng là chạy thử. Với bước chạy thử, người viết sẽ kiểm tra được đối tượng đó có hành động theo đúng kịch bản hay không và đưa ra các sự bổ sung và sửa đổi.

Ví dụ như khi bạn thao tác trên web để đăng nhập facebook, đối tượng cần thực hiện hành động là nhập tên tài khoản và mật khẩu. Lúc này, ở đối tượng tên tài khoản và mật khẩu cần sử dụng đúng hành động nhập dữ liệu và sau đó là hành động click chuột. Việc kiểm tra này sẽ được thực hiện một cách tự động.

Chạy kịch bản là bước cuối cùng. Khi chạy, Selenium Webdriver sẽ gọi cho trình duyệt đó thông qua driver của chính trình duyệt đang kiểm tra. Khi gọi điện trực tiếp, các hành động sẽ được thực thi như sau:

* Mỗi HTTP sẽ được tạo ra khi chạy xong một hành động.
* Sau đó, HTTP sẽ gửi đến driver của trình duyệt, trong đó mỗi driver sẽ có các server để nhận HTTP.
* Server sẽ xác nhận HTTP và gửi lại thông tin.
* Bước cuối cùng, các server sẽ ghi lại hành động có trong kịch bản và thực thi hành động đó với mỗi khách hàng đang thực hiện hành động trên trình duyệt.

## 2.4. Mội số loại kiểm thử sử dụng Selenium Driver

### 2.4.1 Kiểm thử ứng dụng web

Kiểm thử chức năng yêu cầu kiểm thử viên thực hiện kiểm thử tất cả các link trong trang Web, định dạng được sử dụng trong các trang Web để gửi và nhận các thông tin cần thiết từ người dùng. Ngoài ra còn có kết nối cơ sở dữ liệu, kiểm tra cookie và xác minh HTML/CSS, v.v.

Kiểm thử giao diện: Trước khi lập trình viên bắt tay vào xây dựng mã nguồn sẽ luôn có một bản thiết kế UI quy định giao diện của ứng dụng Web. Mỗi thành phần textbox, button, image, link và bố cục trên ứng dụng Web đều được chỉ ra một cách cụ thể trong tài liệu này. Bản thiết kế UI thường sẽ kèm theo file.

PSD – bản vẽ giao diện của ứng dụng Web sử dụng phần mềm Photoshop để lập trình viên dễ dàng xây dựng cũng như khách hàng có thể biết trước ứng dụng của mình sẽ hiển thị ra sao. Đây cũng là tài liệu không thể thiếu cho kiểm thử viên so sánh, đối chiếu giữa thiết kế và nội dung thực tế của ứng dụng hiển thị trên trình duyệt.

Kiểm thử lỗi cú pháp HTML/CSS: Ở bước tiếp theo, người kiểm thử cần xác định các thẻ CSS bị lỗi hoặc các thuộc tính, id, class được viết trong thẻ HTML không hợp lệ hoặc không thuộc bất kỳ thẻ CSS nào.

Kiểm thử cookie và session: Kiểm thử các ứng dụng đăng nhập trong phiên bằng cách cho phép và vô hiệu hóa các tập tin cookie. Có thể thử đưa lỗi vào ứng dụng Web bằng cách sử dụng một tên miền không phù hợp như cố tình truyền sai, thiếu tham số, v.v. Ngoài ra, cần kiểm tra khả năng bảo mật của ứng dụng Web bằng cách xóa các tập tin cookie có chọn lọc khi kiểm thử.

### 2.4.2 Kiểm thử cơ sở dữ liệu (database)

Kiểm tra kết nối tới cơ sở dữ liệu và các lỗi truy vấn có thể gặp phải, đảm bảo dữ liệu được cung cấp chính xác khi các chức năng xem thông tin, thêm, sửa, xoá, v.v. hoạt động.

 Khi import các thư viện trên thì chúng ta có thể viết các method để kết nối đến Database và truy vấn dữ liệu:

|  |
| --- |
| *public class ConnectDB {*  *private static Connection connection = null;*  *@Keyword*  *def connectDB(String url, String dbname, String port, String username, String password){*  *//Load driver class for your specific database type*  *String conn = "jdbc:mysql://" + url + ":" + port + "/" + dbname*  *//Class.forName("org.sqlite.JDBC")*  *//String connectionString = "jdbc:sqlite:" + dataFile*  *if(connection != null && !connection.isClosed()){*  *connection.close()*  *}*  *connection=DriverManager.getConnection(conn,username, password)*  *return connection*  *}*  *@Keyword*  *def executeQuery(String queryString) {*  *Statement stm = connection.createStatement()*  *ResultSet rs = stm.executeQuery(queryString)*  *return rs*  *}*  *@Keyword*  *def closeDatabaseConnection() {*  *if(connection != null && !connection.isClosed()){*  *connection.close()*  *}*  *connection = null*  *}*  *@Keyword*  *def execute(String queryString) {*  *Statement stm = connection.createStatement()*  *boolean result = stm.execute(queryString)*  *return result*  *}*  *}* |

Bản chất của Selenium Driver là kiểm thử hướng từ khóa, nên các method được viết đều phải có [Keyword](https://viblo.asia/u/Keyword" \t "_blank) mới có thể add được vào các testcase.

### 2.4.3 Kiểm thử bảo mật

Ứng dụng Web là một trong những loại ứng dụng có nguy cơ bị tấn công cao nhất. Vì vậy, ngoài việc đảm bảo ứng dụng chạy đúng, ổn định cần phải kiểm tra nghiêm ngặt khả năng bảo mật của ứng dụng. Các công việc cần làm có thể kể đến như: Kiểm tra độ tin cậy của việc phân quyền sử dụng trên ứng dụng. Đưa lỗi vào bằng cách truyền các tham số không hợp lệ trên URL hay trong các form nhập liệu. Lỗ hổng SQL Injection được khai thác mạnh nhất thông qua các thành phần trên. Kiểm tra khả năng truy cập trái phép đối với những thư mục bị cấm trên máy chủ của ứng dụng. Kiểm tra hoạt động các bộ lọc (validation) khi sử dụng chức năng upload tệp tin, thư mục của ứng dụng (nếu có). Kiểm tra độ xác thực khi nhập CAPTCHA trong ứng dụng (nếu có).

### 2.4.4 Kiểm thử ứng dụng di động

Kiểm thử ứng dụng di động thường bắt đầu bằng các luồng kiểm tra giao diện người dùng. Tuy nhiên nếu giao diện ấy có đẹp, bắt mắt và hấp dẫn người dùng đến đâu nhưng các chức năng của sản phẩm đó không hoạt động đúng logic thì cũng coi như sản phẩm thất bại. Một sản phẩm phải đáp ứng được yêu cầu mà khách hàng đề ra.

Selenium Driver hỗ trợ thực thi các trường hợp thử nghiệm giống nhau trên cả ứng dụng máy tính để bàn và ứng dụng di động. Một trường hợp thử nghiệm được tạo thông qua recorder, ở chế độ thủ công hoặc chế độ tập lệnh, trên ứng dụng máy tính để bàn với Selenium Driver cũng có thể được thực thi trên ứng dụng dành cho thiết bị di động. Với cách thực hiện như sau:

* Hiểu ứng dụng đang được kiểm tra (AUT).
* Tạo một trường hợp thử nghiệm với Selenium Driver.
* Thực thi trường hợp thử nghiệm với ứng dụng trên thiết bị di động.

Cũng có thể đạt được trường hợp thử nghiệm mong muốn bằng cách sử dụng Script Mode

Chế độ tập lệnh:

****

*Hình 2.1: Chế độ tập lệnh trong Seleniium*

Luôn thực hiện lại test case trong Selenium Driver để đảm bảo rằng nó hoạt động bình thường.

### 2.4.5 Kiểm thử khả năng tương thích

Kiểm thử này được thực hiện chủ yếu dưới dạng hai ma trận của ứng dụng OS và ứng dụng Mô hình thiết bị. Thông thường, một danh sách các hệ điều hành được hỗ trợ (và đôi khi là các thiết bị) được cung cấp bởi chủ sở hữu sản phẩm hoặc khách hàng.

### 2.4.6 Kiểm thử mạng

Các tính năng chính của Kiểm thử mạng bao gồm:

Ứng dụng nên kết nối với dịch vụ trung gian để thực hiện quy trình.  
Trong quá trình kiểm thử mạng: yêu cầu / phản hồi đến / từ dịch vụ được kiểm tra cho các điều kiện khác nhau. Thử nghiệm này chủ yếu được thực hiện để xác minh thời gian phản hồi trong đó hoạt động được thực hiện như làm mới dữ liệu sau khi đồng bộ hóa hoặc tải dữ liệu sau khi đăng nhập lần đầu, v.v. Quá trình này thực hiện cho cả kết nối wifi mạnh, yếu và mạng dữ liệu di động.

## 2.5. Giải quyết vấn đề Wait-time với Selenium Driver

### 2.5.1. Wait trong Katalon

Thực hiện tự động thường không chính xác như kịch bản do nhiều yếu tố liên quan trong quá trình thực thi như độ ổn định mạng, băng thông internet, hiệu suất của AUT và hiệu suất của máy tính thực thi.

Sử dụng **delayed time** là một thực tế phổ biến trong các kịch bản kiểm thử tự động hóa để tạo tạm dừng ở giữa các bước tự động hóa khi bạn đợi các phần tử web load xong hoặc cho AUT trả lời. Dưới đây là một số ví dụ về lỗi kiểm tra do không đủ thời gian chờ đợi:

***False Fail:***

Một trong những thất bại phổ biến nhất là khi một script không thành công do hết thời gian chờ trên ứng dụng. Nó thường được gây ra bởi độ trễ mạng, yêu cầu cơ sở dữ liệu bị trì hoãn hoặc đơn giản vì hệ thống cần thêm thời gian để xử lý và phản hồi yêu cầu.

***Targeted element không có trên trang:***

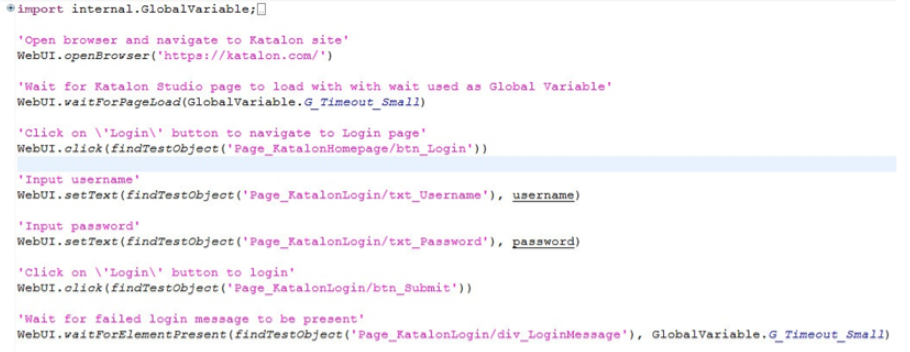
Loại lỗi này xảy ra khi chờ hiển thị hoặc hiển thị các phần tử trong trình duyệt. Ứng dụng có thể đang hoạt động nhưng một số thành phần nhất định có thể không được tải, do đó làm cho các test scripts thất bại.

### *2.5.2.* Qiải quyết vấn đề liên quan đến Wait

Katalon cung cấp các từ khóa dành riêng để trì hoãn một cách rõ ràng. Bạn có thể sử dụng các tùy chọn sau để giải quyết tình huống mình gặp phải một cách linh động.

* Wait For Page Load - Logic này sẽ đợi một trang tải hoàn toàn trước khi chạy một bước khác trong tập lệnh của bạn.
* Wait For Element Present - Đôi khi các yếu tố web mất nhiều thời gian hơn để xuất hiện trên trang. Từ khóa này tạm dừng thực thi cho đến khi targeted element xuất hiện trên trang. Khi phần tử xuất hiện thì việc test mới tiếp tục thực hiện hành động tiếp theo.
* Global variable - Biến này có phạm vi toàn cầu, có nghĩa là nó có thể nhìn thấy trong suốt chương trình. Do đó, bạn có thể sử dụng biến này trong các test scripts của mình theo thời gian phản hồi của ứng dụng web của bạn. Bạn có thể xem xét việc xác định 3 loại biến toàn cầu trong các test scripts của bạn, có các options như cho chờ đợi ngắn, trung bình và dài.

*Ví dụ:* Kịch bản dưới đây cho thấy việc sử dụng biến toàn cầu Global variable và việc sử dụng từ khóa được xây dựng trong Selenium Driver Wait For Element Present để cố ý chờ một phần tử thử nghiệm cụ thể.



*Hình 2.2: Ví dụ về Wait For Element Present*

### 2.5.3. Khi nào nên sử dụng lệnh Wait

Việc thực hiện kiểm thử khá thường xuyên không thành công do các ngoại lệ như ' NoSuchElementException ' , ' ElementNotFoundException ' , ' ElementNotVisibleException ' . Để xử lý các ngoại lệ này, việc đồng bộ hóa có điều kiện phải được thực hiện. Selenium Driver hỗ trợ nhiều tính năng tích hợp sẵn trong “ Wait for…. ” Từ khóa để đối phó với các ngoại lệ như vậy.

*Bảng 2-1: Bảng từ khóa Wait*

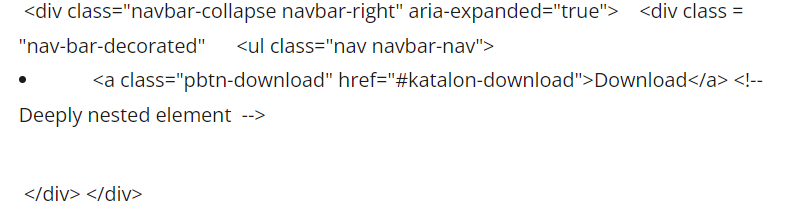
|  |  |
| --- | --- |
| Từ khóa | Miêu tả |
| Wait For Element Present | Đợi phần tử đã cho xuất hiện (xuất hiện) trong khoảng thời gian đã cho bằng giây. |
| Wait For Element Visible | Đợi đến khi phần tử web cụ thể hiển thị trong thời gian chờ. |
| Wait For Alert | Chờ cảnh báo của trình duyệt xuất hiện. |
| Wait for Element Clickable | Đợi phần tử đã cho có thể nhấp vào trong thời gian đã cho bằng giây. |

## 2.6. Phát hiện các yếu tố với Xpath

### 2.6.1. Tổng quan về XPath

Biểu thức XPath là một cơ chế để điều hướng và chọn một nút trong tài liệu XML. `Nó cũng có thể được sử dụng để định vị một phần tử HTML.

***Ví dụ***: Liên kết Tải xuống trong tập lệnh bên dưới là phần tử được lồng trong phần tử khác.



*Hình 2.3: Ví dụ về XPath*

***Vấn đề trong việc xác định các phần tử lồng nhau***

Rất khó để xác định phần tử lồng nhau, chẳng hạn như phần tử <a> trong tập lệnh ở trên. Để xác định XPath theo cách thủ công, chúng tôi cần có kiến thức vững chắc về cấu trúc DOM của trang web.

### 2.6.2. Xác định các yếu tố lồng nhau

Việc xác định XPath là một cách hiệu quả để tìm các phần tử lồng nhau mà các thuộc tính phổ biến như ID, Tên hoặc Lớp không thể nhận dạng được. Có hai cách để tìm XPath:

***Phát hiện XPath bằng các công cụ khác:*** Trình duyệt web thường có Adds-on hỗ trợ người dùng để xác định XPath. Tuy nhiên, nó có thể là một thách thức vì nó phụ thuộc vào rất nhiều công cụ.

***Phát hiện XPath bằng Selenium Driver:*** Selenium Driver có thể tạo và tối ưu hóa XPath cho các phần tử HTML, bất kể chúng lồng nhau như thế nào. Bạn có thể sử dụng các XPath này để xác định các phần tử mà không phải tìm kiếm thông qua cây DOM.

Ví dụ dưới đây minh họa cách Selenium Driver tạo và tối ưu hóa XPath tự động khi bạn gián điệp đối tượng Đăng ký ngay bây giờ (một đối tượng lồng nhau).

Đối phó với các yếu tố thay đổi động Một trong những nhiệm vụ khó khăn và tốn thời gian trong kiểm thử tự động hóa là sửa đổi các kịch bản thử nghiệm khi AUT được thay đổi, đặc biệt là trong các giai đoạn đầu của phát triển phần mềm. Nhà phát triển có thể thay đổi identifiers và elements khá thường xuyên từ bản dựng này sang bản dựng khác. Ngoài ra, trong quá trình thực hiện, các thành phần của AUT có thể thay đổi động.

Để đối phó với những thách thức này, những người làm kiểm thử tự động không nên đặt XPath cố định cho các phần tử trong các trường hợp kiểm thử, nhưng thay vào đó, XPaths sẽ tự động dựa trên các mẫu nhất định. Selenium Driver hỗ trợ tất cả các trục Xpath , chẳng hạn như

* following-sibling
* preceding-sibling
* contains
* descendant
* starts-with

## 2.7. So sánh các công cụ kiểm thử tự động

Selenium Driver ẩn tất cả các sự phức tạp về mặt kỹ thuật đằng sau hiện trường và cung cấp giao diện thân thiện với chế độ thủ công (người dùng có thể kéo-thả, chọn từ khóa và thử nghiệm các đối tượng để tạo thành các bước thử nghiệm) nhưng vẫn giữ vũ khí cần thiết cho những người dùng kỹ thuật mạnh mẽ sâu hơn vào mã hóa với chế độ kịch bản hỗ trợ đầy đủ các tiện ích phát triển như làm nổi bật cú pháp, đề xuất mã và gỡ lỗi. Selenium Driver được xây dựng như là một gói thống nhất bao gồm hầu như tất cả những thứ cần thiết như Java, Android SDK, …

*Bảng 2-2: So sánh ưu nhược điểm của Selenium Driver với một số công cụ*

|  |  |  |
| --- | --- | --- |
| **Công cụ** | **Điểm mạnh** | **Hạn chế** |
| Katalon Studio | - Không có phí cấp phép và bảo trì cần thiết (dịch vụ hỗ trợ chuyên dụng có trả phí có sẵn nếu cần).  - Tích hợp các framework và tính năng cần thiết để tạo và thực hiện các trường hợp kiểm thử nhanh.  - Được xây dựng dựa trên Selenium nhưng loại bỏ nhu cầu về các kĩ năng lập trình nâng cao cần thiết cho Selenium. | - Giải pháp mới nổi với một cộng đồng phát triển nhanh chóng.  - Bộ tính năng vẫn đang phát triển.  - Thiếu các lựa chọn cho các ngôn ngữ kịch bản |
| Selenium | - Mã nguồn mở, không có phí cấp phép và bảo trì.  - Cộng đồng người dùng và phát triển lớn và tích cực để kịp với các công nghệ phần mềm.  - Mở để tích hợp các công cụ và các framework khác để tăng cường khả năng của nó. | - Các nhóm kiểm thử cần phải có kĩ năng và kinh nghiệm lập trình tốt để thiết lập và tích hợp với Selenium với các công cụ và framework khác.  - Hỗ trợ chậm từ cộng đồng. |
| QTP/UFT | - Các tính năng kiểm thử tự động toàn diện được tích hợp vào một hệ thống duy nhất.  - Hỗ trợ người dùng chuyên dụng với một cộng đồng người dùng lớn được thành lập.  - Chỉ yêu cầu các kĩ năng lập trình cơ bản để bắt đầu với việc tạo và thực thi kiểm thử. | - Giải pháp tốn kém: giấy phép và phí bảo trì cao đáng kể.  - Chi phí cao để nâng cấp và các module bổ sung.  - Chỉ hỗ trợ VBScript |

***Đánh giá:***

Không có công cụ nào phù hợp cho tất cả để kiểm thử tự động. Rất khuyến khích người dùng kiểm thử đánh giá các công cụ khác nhau để chọn ra những gì sẽ đáp ứng tốt nhất nhu cầu kiểm thử tự động của họ. Các ngôn ngữ lập trình và công nghệ được sử dụng để phát triển phần mềm tiếp tục phát triển, cũng như các công cụ kiểm thử tự động, khiến chi phí trở thành một yếu tố quan trọng trong việc lựa chọn công cụ. Các nhà cung cấp thương mại thường tính phí nâng cấp công cụ, có thể là dáng kể nếu phần mềm của bạn sử dụng các công nghệ mới nổi và thường xuyên thay đổi.

Mặt khác, các công cụ mã nguồn mở và phi thương mại không phải chịu phí bổ sung mà đòi hỏi nỗ lực và chuyên môn để tích hợp các nâng cấp mới. Thật khó để tìm thấy sự hỗ trợ và chuyên môn cần thiết để tích hợp các công cụ và framework khác nhau vào các giải pháp mã nguồn mở. Các công cụ mới nổi tích hợp với các framework mã nguồn mở, như Selenium, cung cấp một giải pháp thay thế khả thi cho cả các giải pháp kiểm thử tự động mã nguồn mở và thương mại

## 2.8. Tổng kết chương

Chương 2 đã trình bày một cách tổng quát về công cụ kiểm thử tự đông Selenium Driver, nêu nổi bật được cách sử dụng bộ công cụ và các ứng dụng thực tế với Katalon. Ở phần tiếp theo đây, em sẽ đi sâu về việc sử dụng công cụ Selenium Driver cho việc kiểm thử chức năng của website bán hàng online.

**CHƯƠNG 3. TRIỂN KHAI THỰC NGHIỆM KIỂM THỬ ỨNG DỤNG WEB**

**3.1. Triển khai thực nghiệm**

3.1.1. Giới thiệu website

Website “Trinnie Store” là website bán quần áo được phát triển bằng HTML CSS PHP bởi sinh viên chưa có nhiều kinh nghiệm, nên không thể tránh khỏi sai xót. Vì vậy, bản báo cáo này em mong muốn sẽ ứng dụng trực tiếp công cụ kiểm thử tự động Selenium WebDriver vào kiểm thử website này, để cho ứng dụng phát triển hoàn thiện hơn, đúng với yêu cầu đề ra.Website gồm các chức năng chính:

| **STT** | **Chức năng** | **Mô tả** | **Chi tiết** |
| --- | --- | --- | --- |

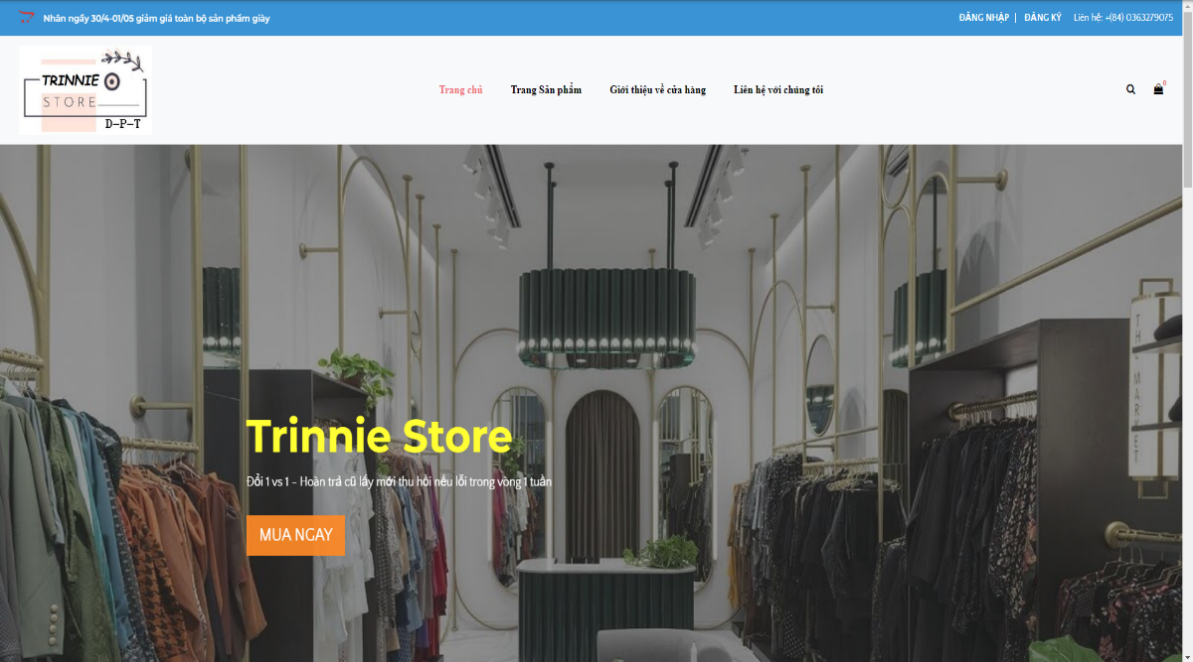
|  |  |  |  |
| --- | --- | --- | --- |
| 1 | Đăng nhập | Cho phép người dùng đăng nhập vào tài khoản để sử dụng các tính năng cá nhân hóa. | - Nhập email/username và mật khẩu. - Hỗ trợ đăng nhập qua Google/Facebook. - Tính năng quên mật khẩu và gửi email khôi phục. |

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | Đăng ký | Người dùng có thể tạo tài khoản mới. | - Nhập thông tin: họ tên, email, số điện thoại, mật khẩu. - Xác thực email qua mã OTP. - Yêu cầu người dùng đồng ý với điều khoản sử dụng trước khi hoàn tất đăng ký. |

|  |  |  |  |
| --- | --- | --- | --- |
| 3 | Thêm giỏ hàng | Cho phép người dùng thêm sản phẩm vào giỏ hàng để lưu trữ và xem lại trước khi thanh toán. | - Nút "Thêm vào giỏ hàng" trên danh sách và trang chi tiết sản phẩm. - Hiển thị tổng số sản phẩm và giá tiền trong giỏ hàng. - Chỉnh sửa số lượng hoặc xóa sản phẩm khỏi giỏ hàng. |

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | Tìm kiếm sản phẩm | Người dùng có thể tìm kiếm sản phẩm theo tên, danh mục hoặc từ khóa. | - Thanh tìm kiếm trực quan. - Gợi ý từ khóa khi nhập. - Lọc kết quả tìm kiếm theo giá, đánh giá và danh mục. |

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | Xem chi tiết sản phẩm | Hiển thị thông tin chi tiết của sản phẩm. | - Hình ảnh sản phẩm (có tính năng phóng to). - Tên sản phẩm, giá, mô tả, đánh giá của khách hàng. - Thông tin khuyến mãi (nếu có). - Nút "Thêm vào giỏ hàng" và |



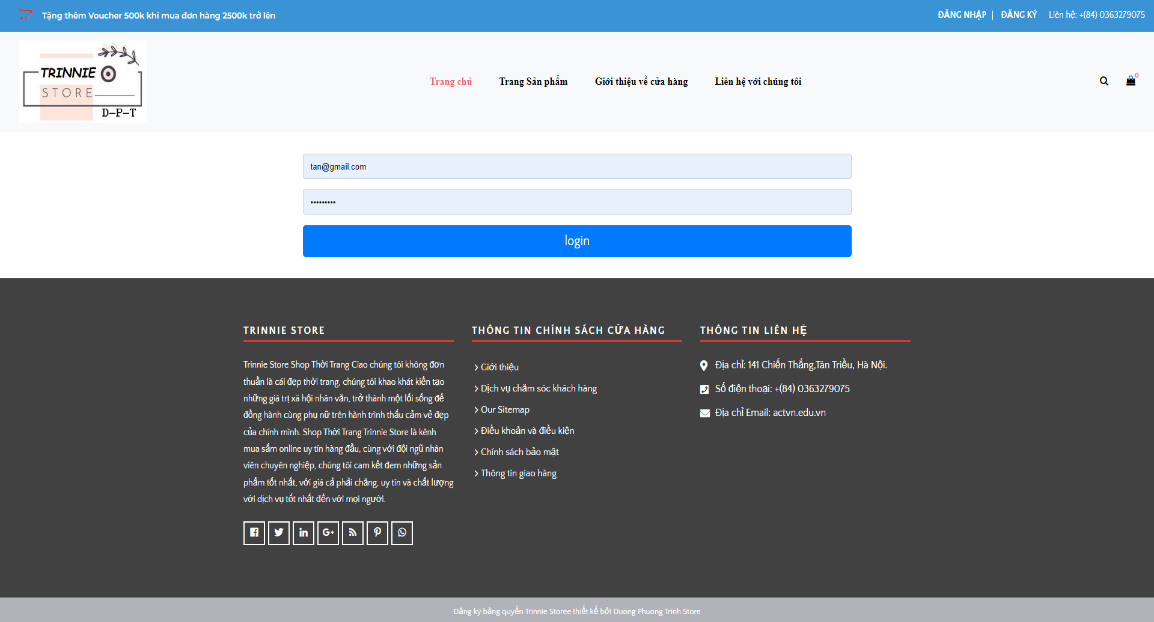
*Hình 3.1. Giao diện trang chủ website Trinnie Store*

3.1.2. các chức năng chính của website

Bản báo cáo sẽ ứng dụng Selenium WebDriver để thực hiện kiểm thử 2 chức năng cơ bản của ứng dụng là chức năng đăng nhập và chức năng đăng ký tài khoản của khách hàng.

*3.1.2.1. Chức năng đăng nhập*

Đây là một chức năng đã xác nhận thành viên của hệ thống. Khách hàng dùng tên truy cập và mật khẩu, xác nhận tài khoản của mình để có thể sử dụng các tính năng nâng cao của hệ thống.



*Hình 3.2. Giao diện chức năng đăng nhập tài khoản của khách hàng*

Mô tả yêu cầu:

*Bảng 3.1. Mô tả yêu cầu cho chứ năng đăng nhập*

|  |  |
| --- | --- |
| Username or Email | Email phải chứa @ |
| Tài khoản hoặc email đã đăng ký trước đó |
| Tài khoản hoặc email không được bỏ trống |
| Password | Password phải từ 8 ký tự trở lên |
| Password chỉ chứa các ký tự: chữ aA – zZ, các số 0 – 9 và \_, + |
| Password không được bỏ trống |

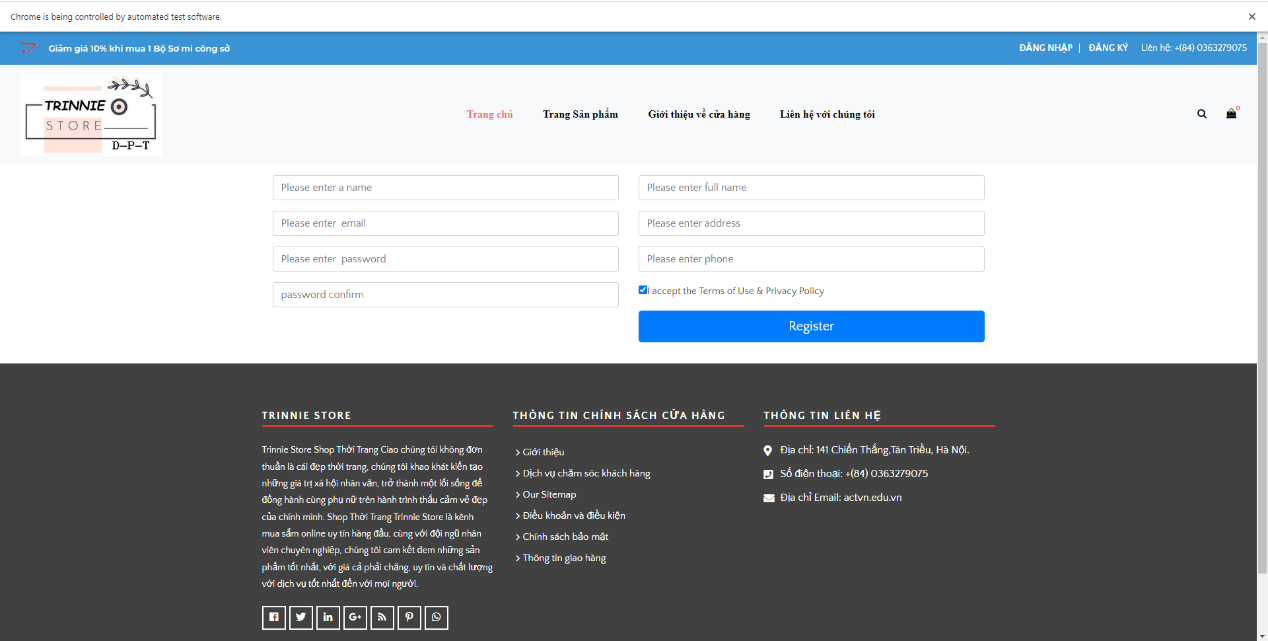
Các trường hợp cần kiểm thử cho chức năng đăng nhập:

*Bảng 3.2. Các trường hợp cần kiểm thử cho chức năng đăng nhập*

|  |  |  |  |
| --- | --- | --- | --- |
| Test Object  (Đối tượng kiểm thử) | Test Type  (Kiểu test) | Input(Condition)  (Điều kiện đầu vào) | Output(ExpectedResult)  (kết quả mong đợi) |
| Username or Email | Chức năng | Nhập sai tài khoản | Hiển thị thông báo “your username and password are wrong” |
| Password | Chức năng | Nhập sai Password | Hiển thị thông báo “your username and password are wrong” |
| Nhập tất cả các trường | Chức năng | Nhập đúng tất cả các trường | Đăng nhập thành công quay về trang chủ. Hiển thị tên người dùng vừa đăng ký |

*3.1.2.2. Chức năng đăng ký*

Đây là chức năng tạo tài khoản cá nhân của khách hàng. Để có được một tài khoản hợp lệ, khách hàng phải điền đầy đủ thông tin của mình theo form của hệ thống.



*Hình 3.3. Giao diện chức năng đăng ký tài khoản của khách hàng*

Mô tả yêu cầu:

*Bảng 3.3. Mô tả yêu cầu cho chức năng đăng ký*

|  |  |
| --- | --- |
| Name | Tên phải từ 5 đến 20 ký tự |
| Tài khoản không chứa các kí tự đặc biệt |
| Tài khoản không được bỏ trống |
| Password | Mật khẩu phải từ 8 ký tự trở lên |
| Mật khẩu chỉ chứa các ký tự : chữ aA-zZ , các số 0-9, và \_ , + |
| Mật khẩu không được bỏ trống |
| Email | Email không được bỏ trống |
| Định dạng: [X@gmail.com](mailto:X@gmail.com) |

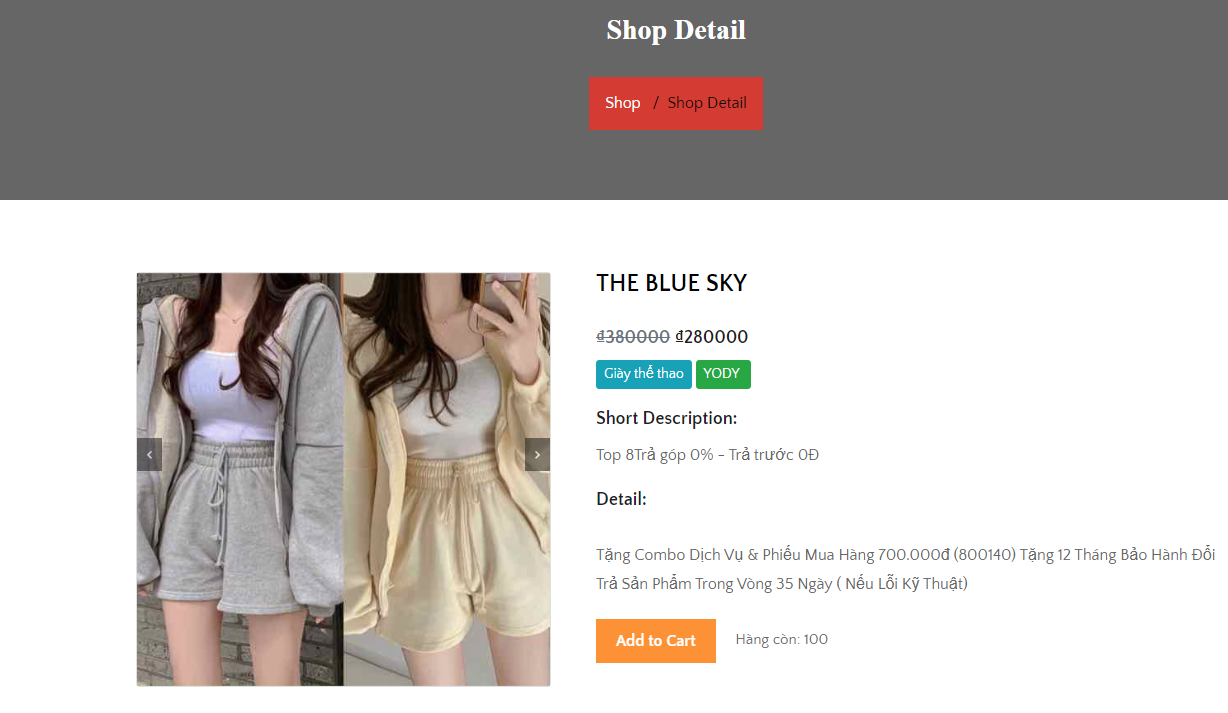
Các trường hợp cần kiểm thử cho chức năng đăng ký:

*Bảng 3.4. Các trường hợp cần kiểm thử cho chức năng đăng ký*

|  |  |  |  |
| --- | --- | --- | --- |
| Test Object  (Đối tượng kiểm thử) | Test Type  (Kiểu test) | Input(Condition)  (Điều kiện đầu vào) | Output(ExpectedResult)  (kết quả mong đợi) |
| Address | Chức năng | Không nhập Address | Hiển thị thông báo “Please enter address” |
| Email | Chức năng | Nhập sai định dạng email | Hiển thị thông báo “Email invalid” |
| Password | Chức năng | Nhập password không đủ mạnh | Hiển thị thông báo “Password must be at least 8 characters” |
| Nhập tất cả các trường | Chức năng | Nhập đúng tất cả các trường | Đăng ký thành công và đăng nhập luôn vào tài khoản vừa đăng ký, quay trở về trang chủ |
|  |  |  |  |

3.1.2.3. chức năng xem chi tiết sản phẩm

Đây là chức năng xem chi tiêt sản phẩm

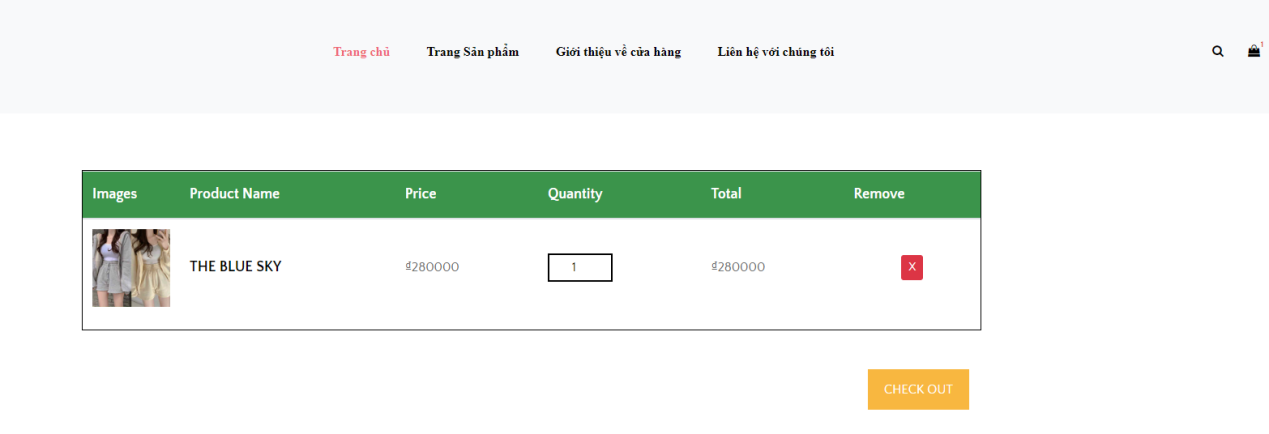


*Hình 3.4. Giao diện chức năng đăng ký tài khoản của khách hàng*

Bảng 3.5. Các trường hợp kiểm thử cho chức năng xem chi tiết sản phẩm

|  |  |  |  |
| --- | --- | --- | --- |
| Test Object  (Đối tượng kiểm thử) | Test Type  (Kiểu test) | Input(Condition)  (Điều kiện đầu vào) | Output(ExpectedResult)  (kết quả mong đợi) |
| XEMSANPHAM1 | Kiểm tra việc xem sản phẩm thành công | 1. Truy cập trang web và chọn một sản phẩm. 2. Hệ thống hiển thị thông tin chi tiết về sản phẩm bao gồm tên sản phẩm, mô tả, hình ảnh, giá cả và đánh giá của khách hàng khác. 3. Người dùng xem qua thông tin chi tiết về sản phẩm. 4. Nếu muốn, người dùng có thể thực hiện các hành động tiếp theo như đặt hàng hoặc thêm sản phẩm vào giỏ hàng. | 1. Hệ thống hiển thị thông tin chi tiết về sản phẩm chính xác và đầy đủ. 2. Người dùng có thể xem qua thông tin chi tiết về sản phẩm một cách dễ dàng. 3. Người dùng có thể thực hiện các hành động tiếp theo như đặt hàng hoặc thêm sản phẩm vào giỏ hàng. |
| XEMSANPHAM2 | Kiểm tra việc xem sản phẩm không thành công | 1.Truy cập trang web và chọn một sản phẩm không có thông tin chi tiết hoặc ảnh. 2. Hệ thống hiển thị thông báo lỗi. | 1. Hệ thống hiển thị thông báo lỗi rõ ràng và dễ hiểu. 2. Người dùng có thể hiểu được lý do tại sao hệ thống không thể hiển thị thông tin chi tiết về sản phẩm |
| XEMSANPHAM3 | Kiểm tra việc xem sản phẩm đã hết hàng hoặc không còn khả dụng | 1.Truy cập trang web và chọn một sản phẩm đã hết hàng hoặc không còn khả dụng. 2. Hệ thống hiển thị thông báo hết hàng hoặc không khả dụng và đề xuất các sản phẩm tương tự khác trong trường hợp sản phẩm đó không còn khả dụng. | 1. Hệ thống hiển thị thông báo hết hàng hoặc không khả dụng rõ ràng và dễ hiểu.. Người dùng có thể hiểu được lý do tại sao sản phẩm đã hết hàng hoặc không còn khả dụng. 2. Người dùng có thể xem các sản phẩm tương tự khác mà họ có thể quan tâm. |
| XEMSANPHAM4 | Kiểm tra xem sản phẩm có được hiển thị đúng cách trên các thiết bị khác nhau như máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. | 1 Truy cập trang web và chọn một sản phẩm. 2. Xem thông tin chi tiết về sản phẩm trên các thiết bị khác nhau như máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. | 1. Thông tin chi tiết về sản phẩm phải được hiển thị chính xác và đầy đủ trên tất cả các thiết bị. 2. Người dùng phải có thể xem thông tin chi tiết về sản phẩm một cách dễ dàng trên tất cả các thiết bị. |
| XEMSANPHAM5 | Kiểm tra xem sản phẩm có được hiển thị đúng cách trong các trình duyệt khác nhau như Chrome, Firefox, Safari và Edge. | 1. Truy cập trang web và chọn một sản phẩm. 2. Xem thông tin chi tiết về sản phẩm trong các trình duyệt khác nhau như Chrome, Firefox, Safari và Edge. | 1. Thông tin chi tiết về sản phẩm phải được hiển thị chính xác và đầy đủ trong tất cả các trình duyệt. 2. Người dùng phải có thể xem thông tin chi tiết về sản phẩm một cách dễ dàng trong tất cả các trình duyệt. |

3.1.2.4 chức năng thêm giỏ hàng



|  |  |  |  |
| --- | --- | --- | --- |
| Test Object  (Đối tượng kiểm thử) | Test Type  (Kiểu test) | Input(Condition)  (Điều kiện đầu vào) | Output(ExpectedResult)  (kết quả mong đợi) |
| THEMGIOHANG1 | Kiểm tra việc thêm sản phẩm vào giỏ hàng thành công | 1.Truy cập trang web và xem thông tin chi tiết về sản phẩm. Chọn nút "Thêm vào giỏ hàng" hoặc biểu tượng giỏ hàng. 1. Hệ thống thêm sản phẩm vào giỏ hàng và hiển thị thông báo xác nhận. 2. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng. 3.Nhấn nút "Đăng ký" | 1. Sản phẩm được thêm vào giỏ hàng thành công. 2. Hệ thống hiển thị thông báo xác nhận. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng |
| THEMGIOHANG2 | Kiểm tra việc thêm sản phẩm đã hết hàng hoặc không còn khả dụng vào giỏ hàng | 1. Truy cập trang web và xem giỏ hàng. 2. Chọn sản phẩm muốn xóa khỏi giỏ hàng. 3. Chọn nút "Xóa" hoặc "Bỏ chọn" để loại bỏ sản phẩm đó khỏi giỏ hàng. | 1. Hệ thống hiển thị thông báo hết hàng hoặc không khả dụng. 2. Hệ thống đề xuất các sản phẩm tương tự khác trong trường hợp sản phẩm đó không còn khả dụng. |
| THEMGIOHANG3 | Kiểm tra việc xóa sản phẩm khỏi giỏ hàng | 1. Truy cập trang web và xem giỏ hàng. 2. Chọn sản phẩm muốn xóa khỏi giỏ hàng. 3. Chọn nút "Xóa" hoặc "Bỏ chọn" để loại bỏ sản phẩm đó khỏi giỏ hàng. | 1. Sản phẩm được xóa khỏi giỏ hàng thành công. 2. Số lượng sản phẩm trong giỏ hàng được cập nhật. |
| THEMGIOHANG4 | Kiểm tra việc thay đổi số lượng sản phẩm trong giỏ hàng | 1. Truy cập trang web và xem giỏ hàng. 2. Chọn sản phẩm muốn thay đổi số lượng. 3. Nhập số lượng sản phẩm mới vào trường tương ứng. 4. Nhấn nút "Cập nhật". | 1. Số lượng sản phẩm được thay đổi thành công. 2. Tổng giá trị đơn hàng được cập nhật. |
| THEMGIOHANG5 | Kiểm tra việc thêm sản phẩm vào giỏ hàng khi người dùng chưa đăng nhập | 1. Truy cập trang web và xem thông tin chi tiết về sản phẩm. 2. Chọn nút "Thêm vào giỏ hàng" hoặc biểu tượng giỏ hàng. 3. Hệ thống yêu cầu người dùng đăng nhập hoặc đăng ký tài khoản trước khi thêm sản phẩm vào giỏ hàng. 4. Hệ thống yêu cầu người dùng đăng nhập hoặc đăng ký tài khoản trước khi thêm sản phẩm vào giỏ hàng. | 1. Hệ thống yêu cầu người dùng đăng nhập hoặc đăng ký tài khoản trước khi thêm sản phẩm vào giỏ hàng. 2. Người dùng có thể đăng nhập hoặc đăng ký tài khoản để thêm sản phẩm vào giỏ hàng |
| THEMGIOHANG6 | Kiểm tra việc thêm sản phẩm vào giỏ hàng trên các thiết bị khác nhau như máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. | 1. Truy cập trang web và xem thông tin chi tiết về sản phẩm. 2. Thêm sản phẩm vào giỏ hàng trên các thiết bị khác nhau như máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. | 1. Sản phẩm được thêm vào giỏ hàng thành công trên tất cả các thiết bị. 2. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng trên tất cả các thiết bị. |
| THEMGIOHANG\_7 | Kiểm tra việc thêm sản phẩm vào giỏ hàng trong các trình duyệt khác nhau như Chrome, Firefox, Safari và Edge. | 1. Truy cập trang đăng ký tài khoản trên hệ thống bán hàng trực tuyến. 2. Nhập thông tin đăng ký tài khoản với mật khẩu quá ngắn. 3. Nhấn nút "Đăng ký". | 1. Sản phẩm được thêm vào giỏ hàng thành công trong tất cả các trình duyệt. 2. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng trong tất cả các trình duyệt. |

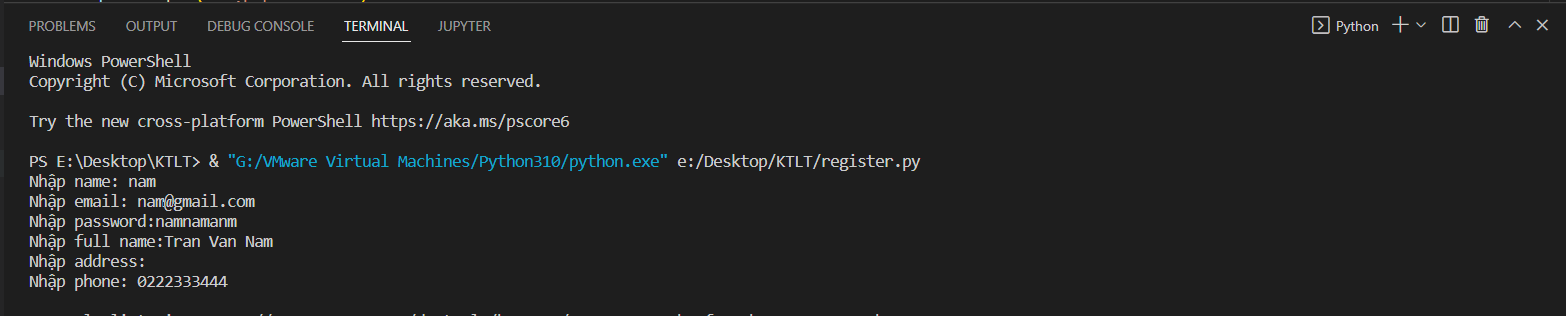
3.1.2.5 chức năng tìm kiếm

|  |  |  |  |
| --- | --- | --- | --- |
| Test Object  (Đối tượng kiểm thử) | Test Type  (Kiểu test) | Input(Condition)  (Điều kiện đầu vào) | Output(ExpectedResult)  (kết quả mong đợi) |
| THEMGIOHANG1 | Kiểm tra việc thêm sản phẩm vào giỏ hàng thành công | 1.Truy cập trang web và xem thông tin chi tiết về sản phẩm. Chọn nút "Thêm vào giỏ hàng" hoặc biểu tượng giỏ hàng. 1. Hệ thống thêm sản phẩm vào giỏ hàng và hiển thị thông báo xác nhận. 2. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng. 3.Nhấn nút "Đăng ký" | 1. Sản phẩm được thêm vào giỏ hàng thành công. 2. Hệ thống hiển thị thông báo xác nhận. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng |
| THEMGIOHANG2 | Kiểm tra việc thêm sản phẩm đã hết hàng hoặc không còn khả dụng vào giỏ hàng | 1. Truy cập trang web và xem giỏ hàng. 2. Chọn sản phẩm muốn xóa khỏi giỏ hàng. 3. Chọn nút "Xóa" hoặc "Bỏ chọn" để loại bỏ sản phẩm đó khỏi giỏ hàng. | 1. Hệ thống hiển thị thông báo hết hàng hoặc không khả dụng. 2. Hệ thống đề xuất các sản phẩm tương tự khác trong trường hợp sản phẩm đó không còn khả dụng. |
| THEMGIOHANG3 | Kiểm tra việc xóa sản phẩm khỏi giỏ hàng | 1. Truy cập trang web và xem giỏ hàng. 2. Chọn sản phẩm muốn xóa khỏi giỏ hàng. 3. Chọn nút "Xóa" hoặc "Bỏ chọn" để loại bỏ sản phẩm đó khỏi giỏ hàng. | 1. Sản phẩm được xóa khỏi giỏ hàng thành công. 2. Số lượng sản phẩm trong giỏ hàng được cập nhật. |
| THEMGIOHANG4 | Kiểm tra việc thay đổi số lượng sản phẩm trong giỏ hàng | 1. Truy cập trang web và xem giỏ hàng. 2. Chọn sản phẩm muốn thay đổi số lượng. 3. Nhập số lượng sản phẩm mới vào trường tương ứng. 4. Nhấn nút "Cập nhật". | 1. Số lượng sản phẩm được thay đổi thành công. 2. Tổng giá trị đơn hàng được cập nhật. |
| THEMGIOHANG5 | Kiểm tra việc thêm sản phẩm vào giỏ hàng khi người dùng chưa đăng nhập | 1. Truy cập trang web và xem thông tin chi tiết về sản phẩm. 2. Chọn nút "Thêm vào giỏ hàng" hoặc biểu tượng giỏ hàng. 3. Hệ thống yêu cầu người dùng đăng nhập hoặc đăng ký tài khoản trước khi thêm sản phẩm vào giỏ hàng. 4. Hệ thống yêu cầu người dùng đăng nhập hoặc đăng ký tài khoản trước khi thêm sản phẩm vào giỏ hàng. | 1. Hệ thống yêu cầu người dùng đăng nhập hoặc đăng ký tài khoản trước khi thêm sản phẩm vào giỏ hàng. 2. Người dùng có thể đăng nhập hoặc đăng ký tài khoản để thêm sản phẩm vào giỏ hàng |
| THEMGIOHANG6 | Kiểm tra việc thêm sản phẩm vào giỏ hàng trên các thiết bị khác nhau như máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. | 1. Truy cập trang web và xem thông tin chi tiết về sản phẩm. 2. Thêm sản phẩm vào giỏ hàng trên các thiết bị khác nhau như máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. | 1. Sản phẩm được thêm vào giỏ hàng thành công trên tất cả các thiết bị. 2. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng trên tất cả các thiết bị. |
| THEMGIOHANG\_7 | Kiểm tra việc thêm sản phẩm vào giỏ hàng trong các trình duyệt khác nhau như Chrome, Firefox, Safari và Edge. | 1. Truy cập trang đăng ký tài khoản trên hệ thống bán hàng trực tuyến. 2. Nhập thông tin đăng ký tài khoản với mật khẩu quá ngắn. 3. Nhấn nút "Đăng ký". | 1. Sản phẩm được thêm vào giỏ hàng thành công trong tất cả các trình duyệt. 2. Người dùng có thể tiếp tục mua sắm hoặc xem lại giỏ hàng để tiến hành đặt hàng trong tất cả các trình duyệt. |

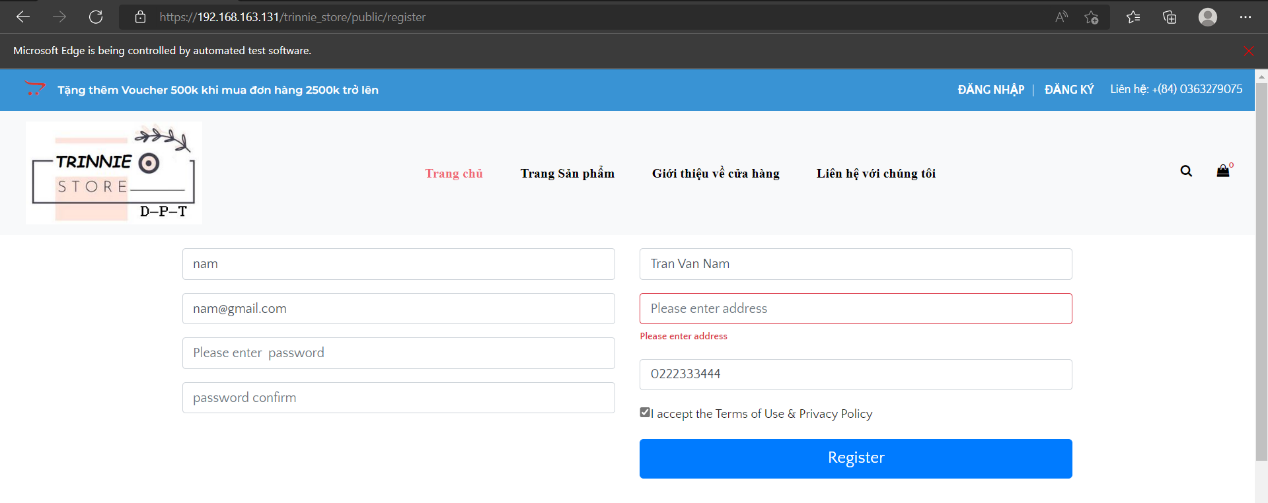
**3.2. Triển khai kiểm thử chức năng chính bằng Selenium**

3.2.1. Đăng ký

Trường hợp 1: Đăng ký nhưng thiếu địa chỉ

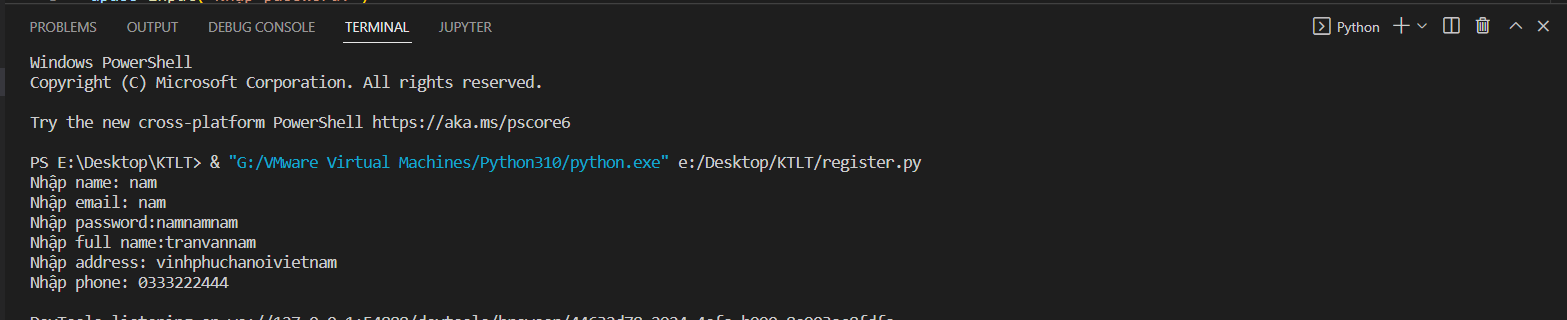


*Hình 3.4. TestScript cho ca kiểm thử đăng ký nhưng thiếu địa chỉ*

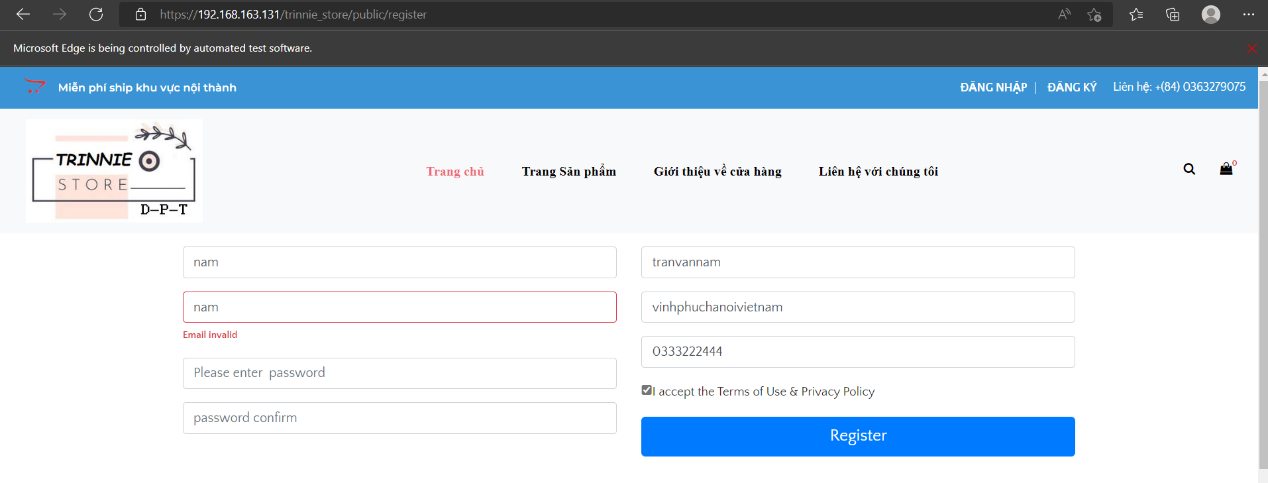


*Hình 3.5. Thực thi ca kiểm thử đăng ký nhưng thiếu địa chỉ*

Trường hợp 2: Đăng ký sai định dạng email

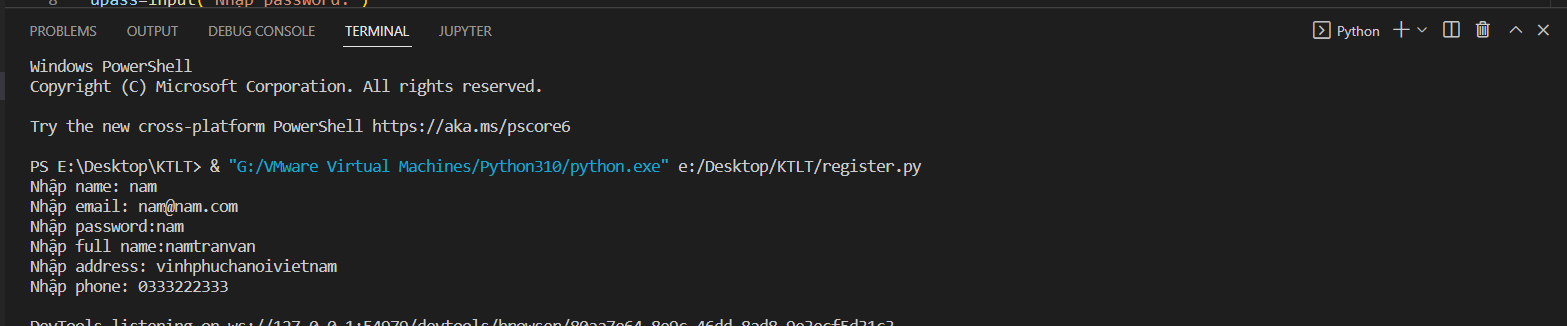


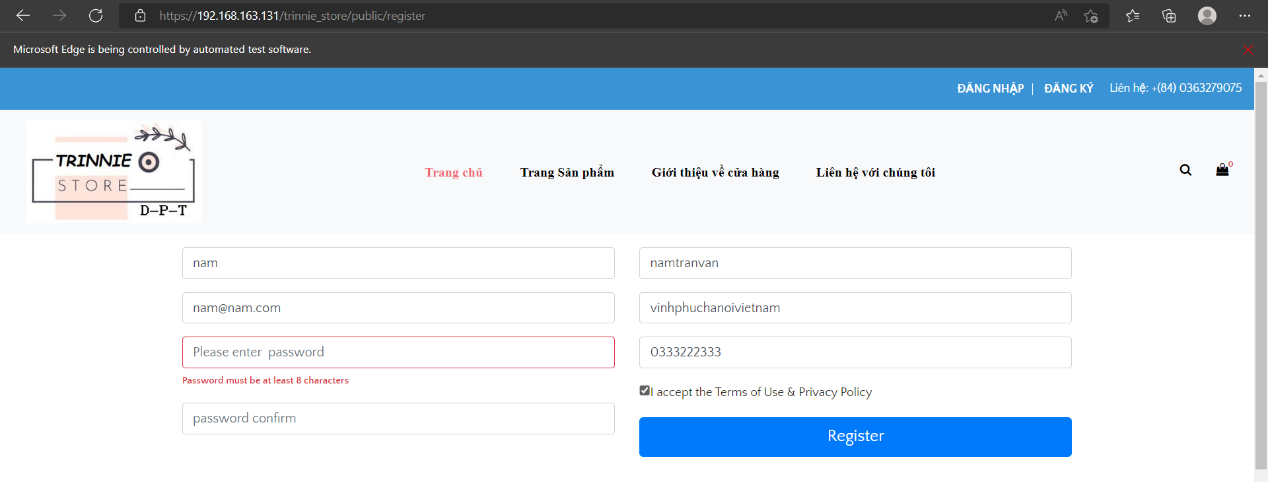
*Hình 3.6. TestScript cho ca kiểm thử đăng ký sai định dạng email*



*Hình 3.7. Thực thi ca kiểm thử đăng ký sai định dạng email*

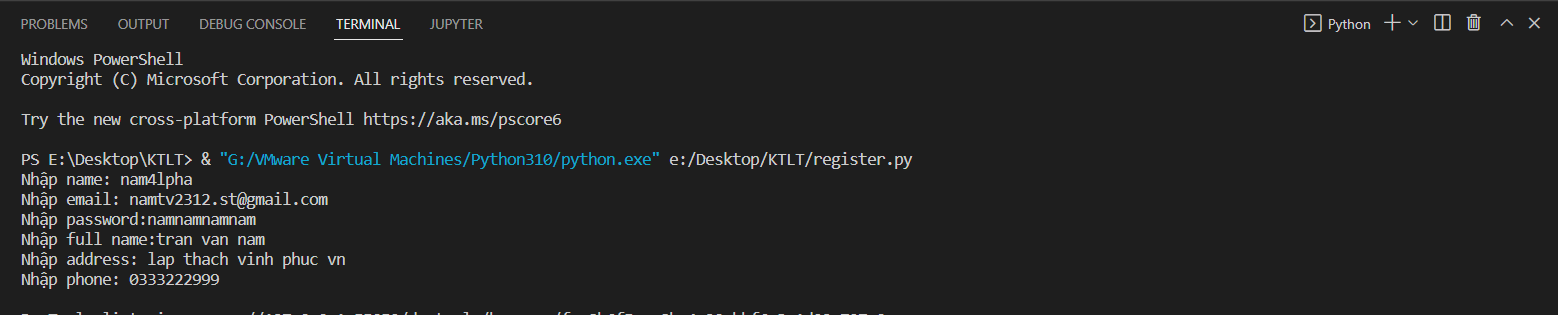
Trường hợp 3: Đăng ký với mật khẩu không đủ mạnh

 *Hình 3.8. TestScript cho ca kiểm thử đăng ký với mật khẩu không đủ mạnh*

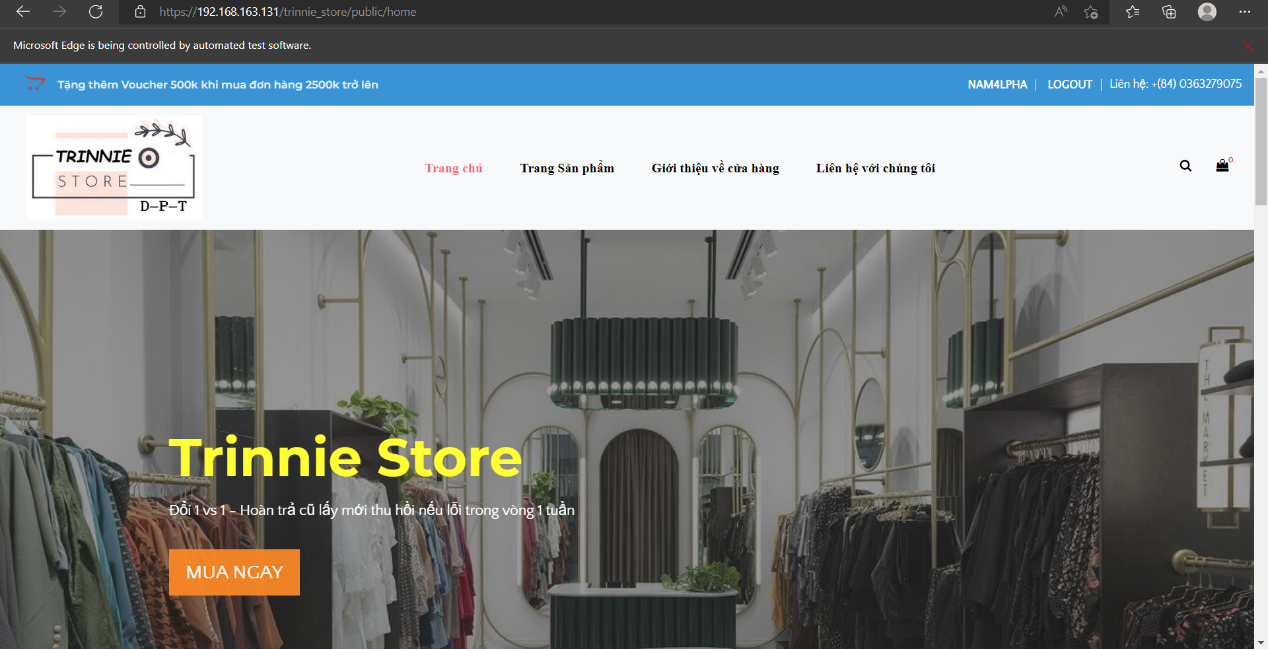


*Hình 3.9. Thực thi ca kiểm thử đăng ký với mật khẩu không đủ mạnh*

Trường hợp 4: Đăng ký thành công



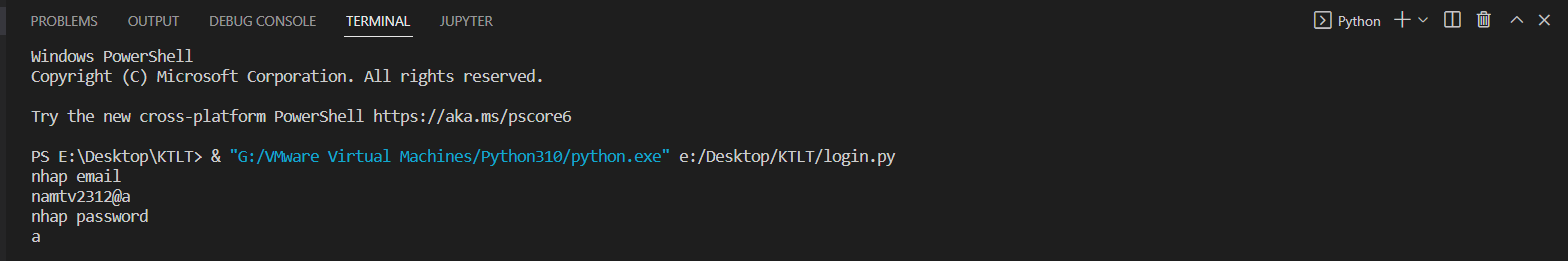
*Hình 3.10. TestScript cho ca kiểm thử đăng ký thành công*



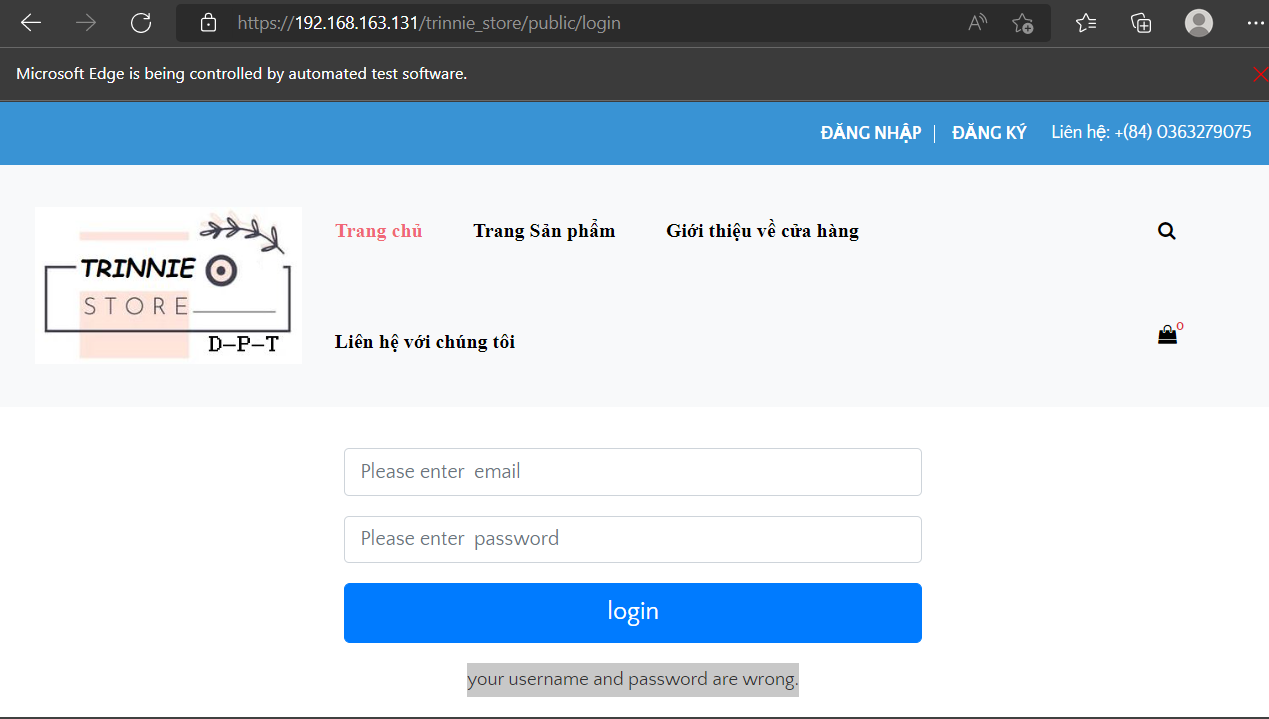
*Hình 3.11. Thực thi ca kiểm thử đăng ký thành công*

3.2.2. Đăng nhập

Trường hợp 1: Đăng nhập với email sai

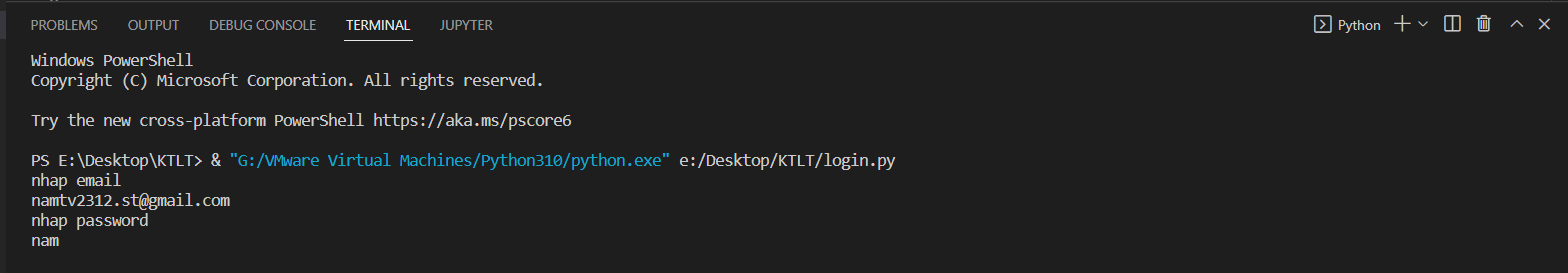


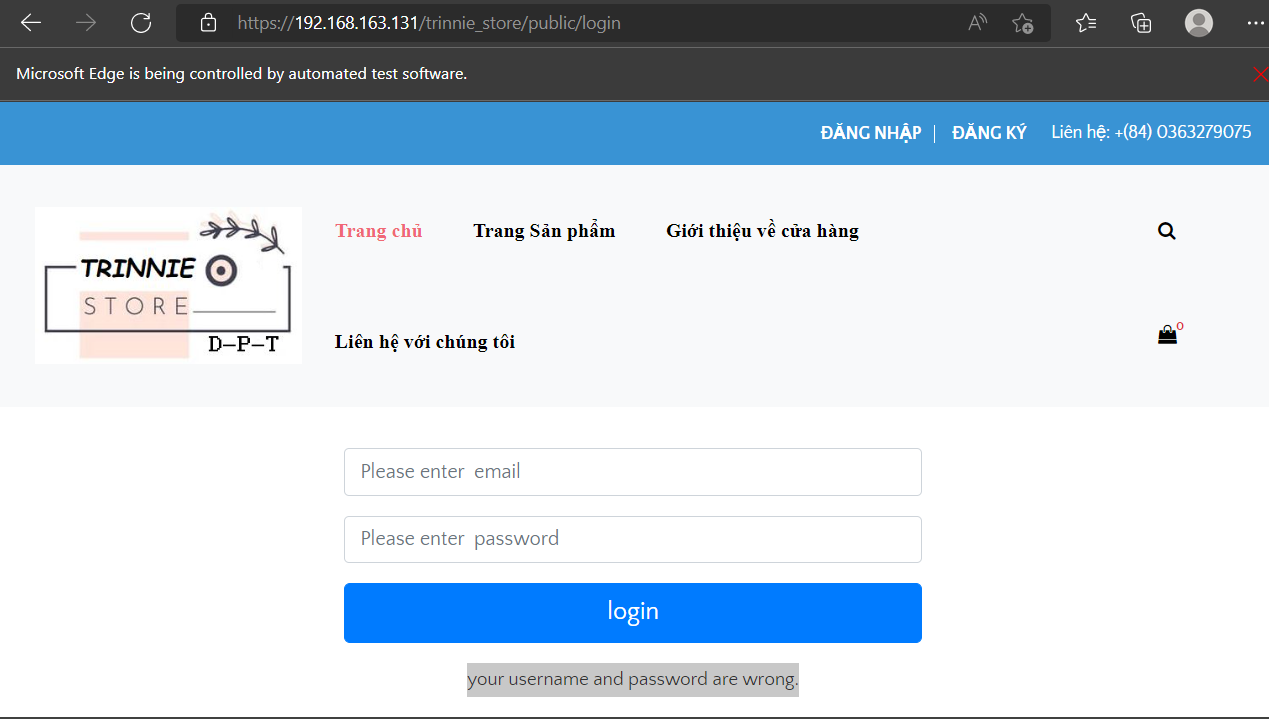
*Hình 3.12. TestScript cho ca kiểm thử đăng nhập với email sai*



*Hình 3.13. Thực thi ca kiểm thử đăng nhập với email sai*

Trường hợp 2: Đăng nhập với password sai

 *Hình 3.14. TestScript cho ca kiểm thử đăng nhập với password sai*

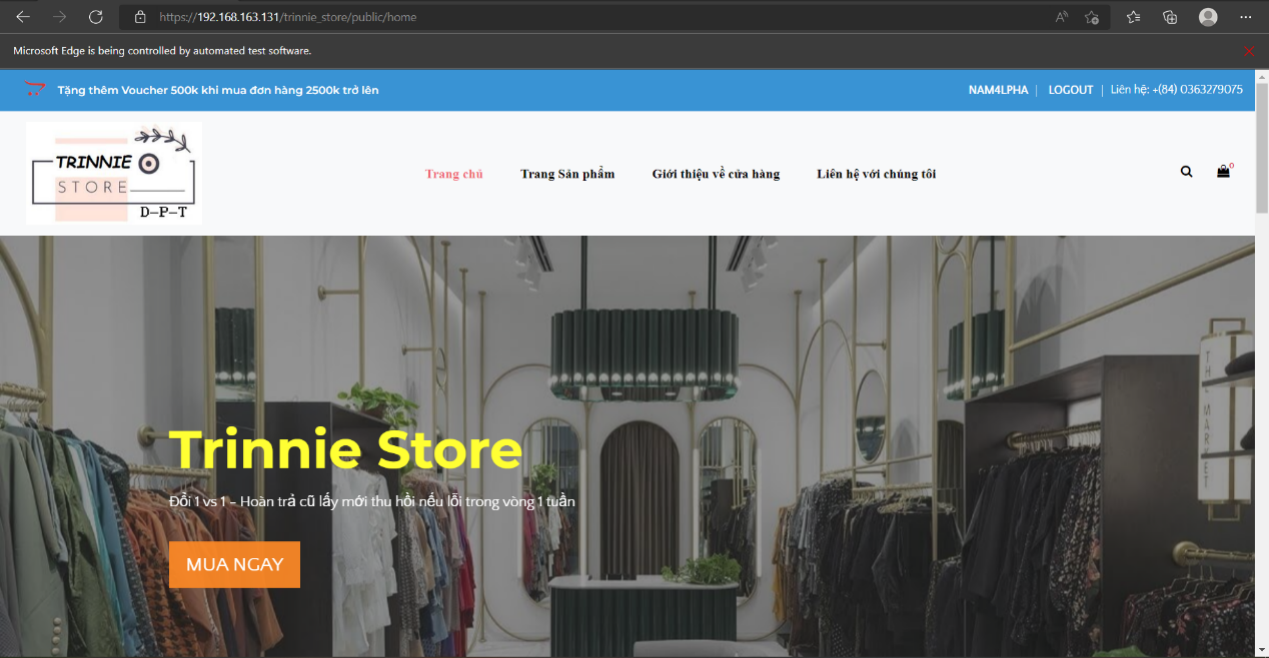


*Hình 3.15. Thực thi ca kiểm thử đăng nhập với password sai*

Trường hợp 3 : Đăng nhập thành công

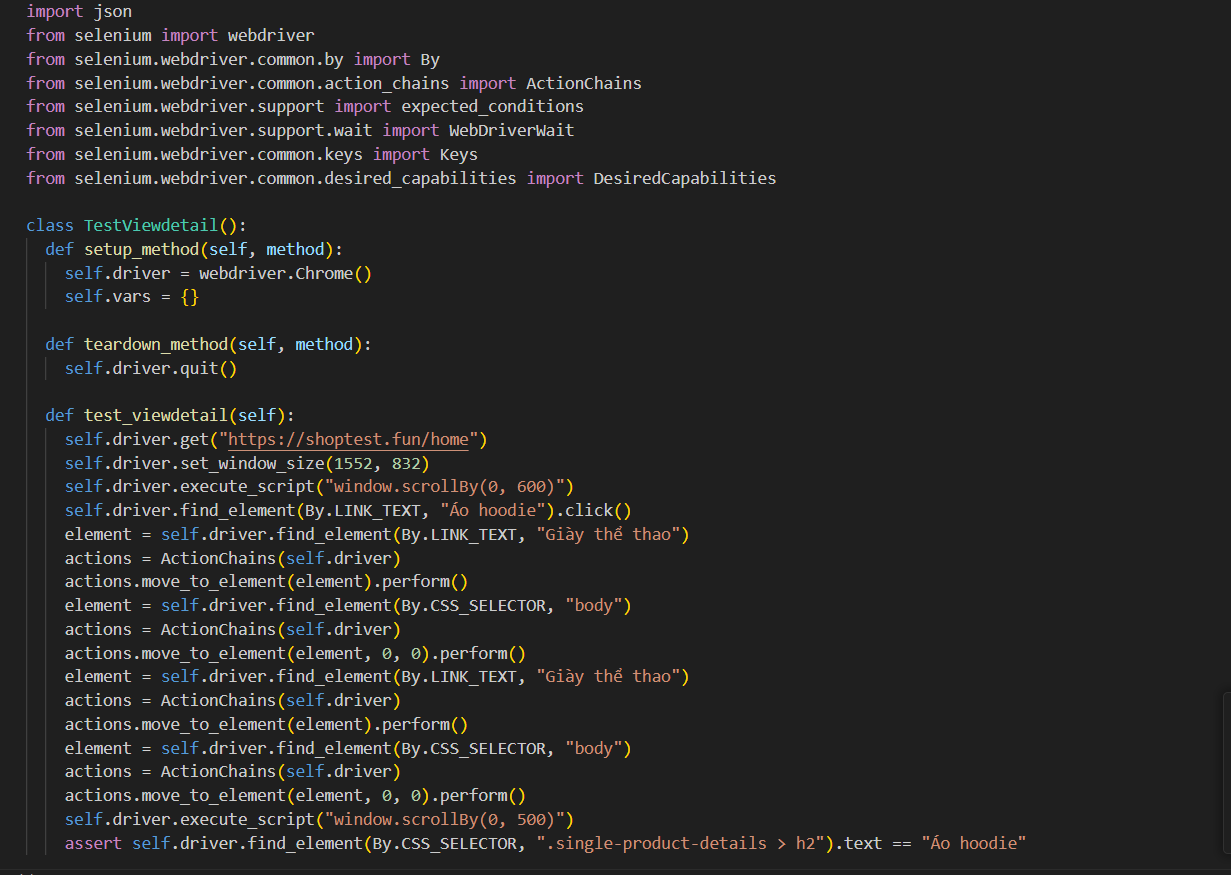


*Hình 3.16. TestScript cho ca kiểm thử đăng nhập thành công*



*Hình 3.17. Thực thi ca kiểm thử đăng nhập thành công*

3.2.3. Xem chi tiết sản phẩm

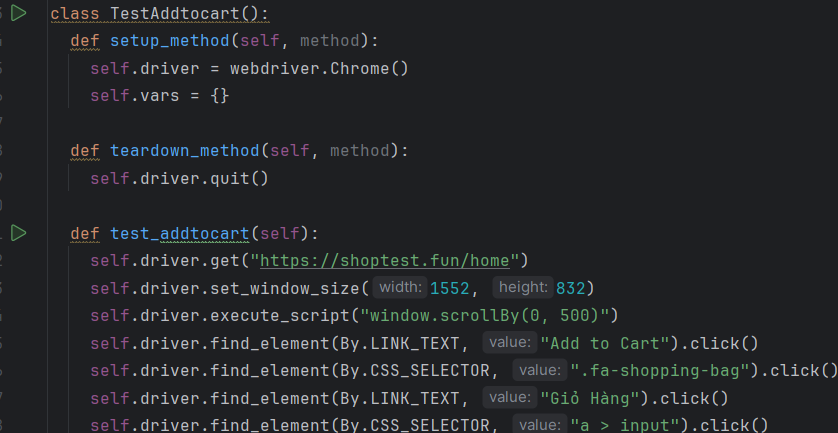


*Hình 3.18. TestScript cho ca kiểm thử xem chi tiết sản phầm*



*Hình 3.19. Xem chi tiết sản phẩm thành công*

3.2.4. Thêm giỏ hàng

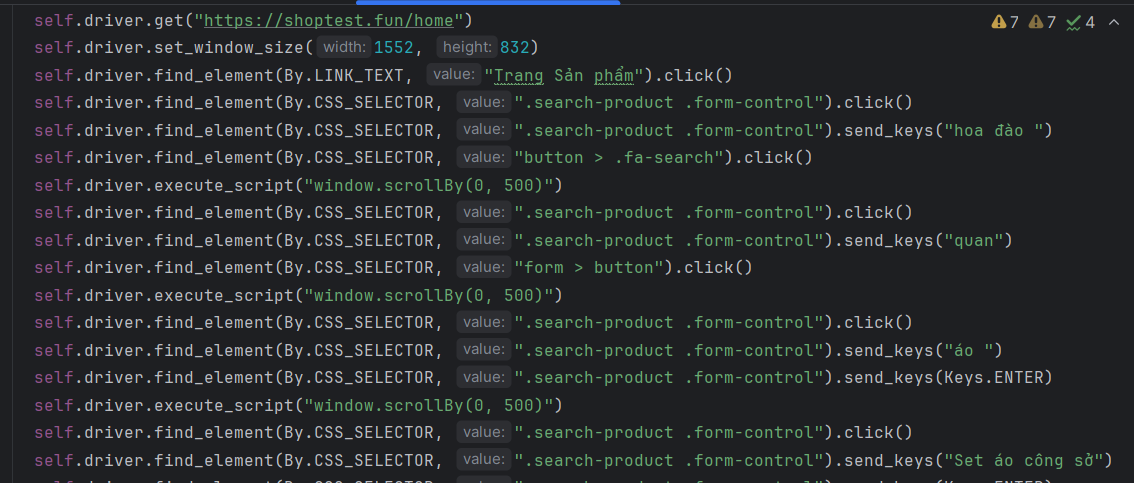


*Hình 3.20. TestScript cho ca kiểm thử thêm vào giỏ hàng*

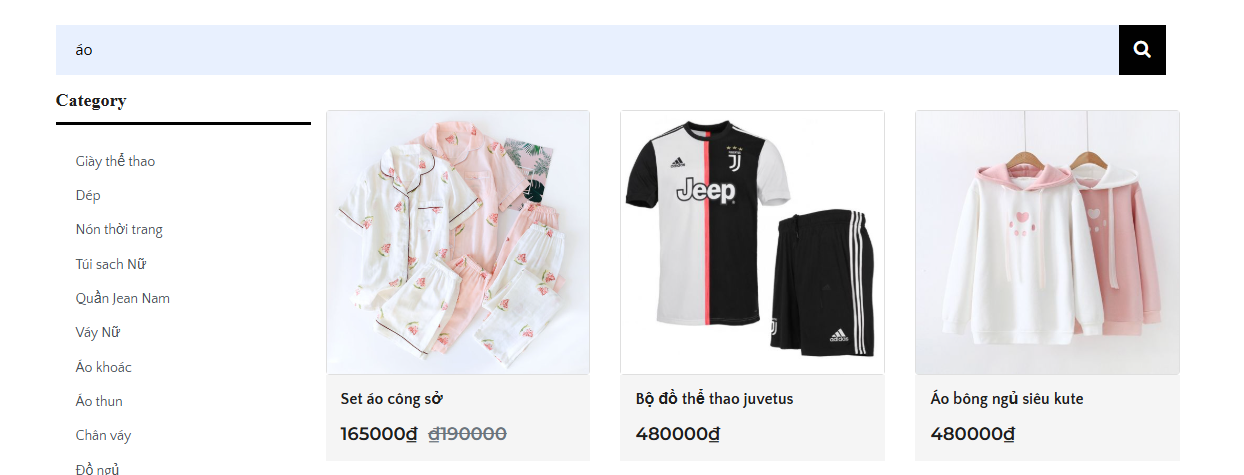


*Hình 3.21. Thực thi thành công ca kiểm thử thêm vào giỏ hàng*

3.2.5. Tìm kiếm sản phẩm



*Hình 3.22. TestScript cho ca kiểm thử tìm kiểm sản phẩm*



*Hình 3.23. Thực thi thành công ca kiểm thử tìm kiếm sản phẩm*

**KẾT LUẬN**

Kiểm thử phần mềm nói chung và kiểm thử ứng dụng trên nền Web nói riêng là một vấn đề hết sức quan trọng đối với các tổ chức phát triển phần mềm hiện nay. Trong quá trình thực hiện báo cáo của em do thời gian nghiên cứu và kinh nghiệm bản thân còn hạn chế nên một số phần của báo cáo nghiên cứu chưa được sâu.

Với sự hướng dẫn tận tình của …….., báo cáo của em đã đạt được những kết quả sau:

**Kết quả đạt được**

* Đã tìm hiểu, nghiên cứu và trình bày tổng quan về lĩnh vực kiểm thử phần mềm nói chung và lĩnh vực web nói riêng.
* Trình bày đầy đủ tổng quan về phần mềm, lỗi phần mềm, đặc tả yêu cầu phần mềm, chất lượng và độ tin cậy của phần mềm cũng như các vấn đề liên quan tới kiểm thử phần mềm.
* Chỉ ra được sự khác biệt của công việc kiểm thử ứng dụng trên nền Web với công việc kiểm thử trên các môi trường khác.

**Hạn chế**

Do trình độ, khả năng và thời gian còn hạn chế nên báo cáo của nhóm còn tồn tại một số điểm hạn chế:

* Mặc dù đã tìm tòi, cũng như chủ động trong việc tìm kiếm nguồn tài liệu, các báo cáo nước ngoài bằng tiếng Anh để nghiên cứu, nhưng việc thể hiện lại bằng tiếng Việt còn nhiều bối rối khi chưa truyền tải chính xác các thuật ngữ chuyên ngành.
* Trong khuôn khổ báo cáo chưa thể trình bày đầy đủ các tính năng, và nghiên cứu các kỹ thuật nâng cao được dử dụng trong Selenium WebDriver.
* Nguồn tài liệu bằng tiếng Việt không nhiều, đặc biệt là tài liệu chuyên sâu.
* Chỉ áp dụng kiểm thử được chức năng đăng ký, đăng nhập tài khoản người dùng.

**Hướng phát triển**

* Cần nghiên cứu sâu hơn về Selenium WebDriver trong lĩnh vực an toàn thông tin.
* Kết hợp các công cụ kiểm thử tự động để cải tiến quy trình kiểm thử. Đảm bảo ứng dụng web có thể được kiểm tra về chức năng cững như bảo mật.

**TÀI LIỆU THAM KHẢO**

1. Trương Anh Hoàng, Đặng Văn Hưng, Phạm Ngọc Hùng, Giáo trình kiểm thử phần mềm, Hà Nội: Đại học Quốc gia Hà Nội, Tháng 1 năm 2014.
2. Marijn Haverbeke. Eloquent JavaScript, Third Edition. No Starch Pree 2018.
3. Arnon Axelrod. Complete Guide to Test Automation. Apress 2018.
4. KMS Technology Releases Katalon Studio, Free Intelligent Test Automation Toolset, <https://www.katalon.com/resources-center/blog/kms-technology-releases-katalon-studio-free-intelligent-test-automation-toolset/>, truy cập ngày 18/12/2023.
5. 7 Principles Of Software Testing: Defect Clustering And Pareto Principle, <https://www.softwaretestinghelp.com/7-principles-of-software-testing>, truy cập ngày 20/12/2023.

**PHỤ LỤC**

1. **Mã script của các testcase**
   1. Register

|  |
| --- |
| from selenium import webdriver  from selenium.webdriver.common.keys import Keys  from selenium.webdriver.edge.service import Service  from selenium.webdriver.common.by import By  uname=input("Nhập name: ")  uemail=input("Nhập email: ")  upass=input("Nhập password:")  ufname=input("Nhập full name:")  uaddress=input("Nhập address: ")  uphone=input("Nhập phone: ")  s=Service("G:\VMware Virtual Machines\edgedriver\_win64\msedgedriver.exe")  driver=webdriver.Edge(service=s)  driver.get("https://192.168.163.131/trinnie\_store/public/home")  driver.maximize\_window()  driver.find\_element(By.XPATH,"/html/body/div[1]/div[1]/div/div[1]/div[2]/div[2]/ul/li[2]/a").click()  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[1]/div[1]/input").send\_keys(uname)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[1]/div[2]/input").send\_keys(uemail)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[1]/div[3]/input").send\_keys(upass)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[1]/div[4]/input").send\_keys(upass)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[2]/div[1]/input").send\_keys(ufname)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[2]/div[2]/input").send\_keys(uaddress)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[2]/div[3]/input").send\_keys(uphone)  driver.find\_element(By.XPATH,value="/html/body/div[1]/div[3]/div/form/div/div[2]/div[5]/button").click() |

* 1. SignUp

|  |
| --- |
| from selenium import webdriver  from selenium.webdriver.common.by import By  # from selenium.webdriver.support.ui import WebDriverWait  # from selenium.webdriver.support import expected\_conditions as EC  # from webdriver\_manager.firefox import GeckoDriverManager  # from webdriver\_manager.chrome import ChromeDriverManager  from selenium.webdriver.chrome.service import Service  import time  import os  ###########################################################  #enter the link to the website you want to automate login.  website\_link="https://192.168.163.131/trinnie\_store/public/login"  #enter your login email  print("nhap email")  email = input()  # email="tan@gmail.com"  #enter your login password  print("nhap password")  password = input()  # password="05042001a"  ###########################################################  #enter the element for email input field  xpath\_for\_email="/html/body/div/div[3]/div/form/div[1]/input"  #enter the element for password input field  xpath\_for\_password="/html/body/div/div[3]/div/form/div[2]/input"  #enter the element for submit button  xpath\_for\_submit="/html/body/div/div[3]/div/form/div[3]/button"  ###########################################################  ###Firefox browser  # browser = webdriver.Firefox(executable\_path=GeckoDriverManager().install())   #for Firefox user  # browser = webdriver.Safari()  #for macOS users[for others use chrome vis chromedriver]  # browser.get((website\_link))  ###Chrome driver  driver = webdriver.Edge(service=Service("G:\VMware Virtual Machines\edgedriver\_win64\msedgedriver.exe"))  driver.get(website\_link)  try:      email\_element = driver.find\_element(By.XPATH, xpath\_for\_email)      email\_element.send\_keys(email)      password\_element  = driver.find\_element(By.XPATH, xpath\_for\_password)      password\_element.send\_keys(password)      if not password\_element:          print("nono nono")      signInButton = driver.find\_element(By.XPATH, xpath\_for\_submit)      signInButton.click()        #### to quit the browser uncomment the following lines ####      # time.sleep(3)      # browser.quit()      # time.sleep(1)      # browserExe = "Safari"      # os.system("pkill "+browserExe)  except Exception:      #### This exception occurs if the element are not found in the webpage.      print("Some error occured :(")      #### to quit the browser uncomment the following lines ####      # browser.quit()      # browserExe = "Safari"      # os.system("pkill "+browserExe) |