|  |  |
| --- | --- |
| 1-02 | BỘ GIÁO DỤC VÀ ĐÀO TẠO  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH** |

**ĐỒ ÁN MÔN HỌC**

**KIỂM THỬ VÀ ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM**

**KIỂM THỬ WEBSITE** **SHOP ACCOUNT GAME**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Chuyên ngành : **CÔNG NGHỆ PHẦN MỀM**

Giảng viên hướng dẫn : **Bùi Mạnh Toàn**

Sinh viên thực hiện :

Họ và tên MSSV

**Hồ Đại Minh**  **2180604350**

**Phan Thị Hằng** **2180605032**

**Liêu Đông Huy** **2180606424**

**Phạm Tấn Phát** **2180606267**

**Mã Hoàng Thái 2180606816**

Lớp: **21DTHC1**

TP. Hồ Chí Minh, 2024

# LỜI NÓI ĐẦU

Ngày nay, ứng dụng công nghệ thông tin vào đời sống được xem là một trong những yếu tố mang tính quyết định trong hoạt động của hầu hết tất cả mọi người, nó đóng vai trò hết sức quan trọng, có thể tạo ra những bước đột phá mạnh mẽ.

Cùng với sự phát triển không ngừng về kỹ thuật máy tính và mạng điện tử, công nghệ thông tin cũng được những công nghệ có đẳng cấp và lần lượt chinh phục hết đỉnh cao này đến đỉnh cao khác. Mạng Internet là một trong những sản phẩm có giá trị hết sức lớn lao và ngày càng trở nên một công cụ không thể thiếu, là nền tảng chính cho sự truyền tải, trao đổi thông tin trên toàn cầu. Giờ đây, mọi việc liên quan đến thông tin trở nên thật dễ dàng cho người sử dụng,…

Bằng Internet, chúng ta đã thực hiện được nhiều công việc với tốc độ nhanh hơn và chi phí thấp hơn nhiều so với cách thức truyền thống. Trong hoạt động sản xuất, kinh doanh, giờ đây, thương mại điện tử đã khẳng định được vai trò xúc tiến và thúc đẩy sự phát triển của doanh nghiệp. Đối với một trang website online, việc quảng bá và giới thiệu đến khách hàng các sản phẩm mới đáp ứng được nhu cầu của khách hàng sẽ là cần thiết. Vì vậy, chúng em đã thực hiện đồ án xây dựng “WEBSITE SHOP ACCOUNT GAME”.

MỤC LỤC

[LỜI NÓI ĐẦU 2](#_Toc28901)

[MỤC LỤC 3](#_Toc529)

[**CHƯƠNG 1: GIỚI THIỆU KIỂM THỬ 6**](#_Toc21387)

[1.1 Khái niệm 6](#_Toc5775)

[1.2 Các quy tắc trong kiểm thử 6](#_Toc5081)

[1.3 Các phương pháp trong kiểm thử 7](#_Toc17129)

[**CHƯƠNG 2: THIẾT KẾ TEST CASE 11**](#_Toc4894)

[2.1 Các phương pháp thiết kế test-case (Black-box Testing) 11](#_Toc21443)

[2.2 Kiểm thử hộp trắng (White-box Testing) 13](#_Toc5232)

[2.3 Mô tả đề tài và kế hoạch kiểm thử 13](#_Toc7494)

[*2.3.1 Mô tả đề tài* 13](#_Toc31267)

[*2.3.2 Kế hoạch kiểm thử* 14](#_Toc7386)

[**CHƯƠNG 3: TRIỂN KHAI KIỂM THỬ 16**](#_Toc27264)

[3.1 Mô tả các chức năng 16](#_Toc31519)

[3.2 Kiểm thử hộp đen 23](#_Toc10785)

[*3.2.1 Phương pháp phân vùng tương đương (Xem phim)* 23](#_Toc7757)

[*3.2.2 Kiểm thử phân tích giá trị biên (Đăng ký)* 24](#_Toc334)

[*3.2.3 Kiểm thử bảng quyết định (Tìm kiếm phim)* 27](#_Toc14059)

[*3.2.4 Kiểm thử dịch chuyển trạng thái (đăng nhập)* 29](#_Toc25256)

[3.1 Kiểm thử hộp trắng 31](#_Toc107)

[*3.1.1 Phương pháp phủ câu lệnh (Đăng nhập)* 31](#_Toc28707)

[*3.1.2 Phương pháp phủ nhánh (Chỉnh sửa phim)* 32](#_Toc549)

[*3.1.3 Phương pháp phủ đường dẫn (Đăng ký)* 33](#_Toc13323)

[*3.1.4 Phương pháp phủ điều kiện (Thêm phim)* 35](#_Toc1094)

[**CHƯƠNG 4: KIỂM THỬ TỰ ĐỘNG 38**](#_Toc18624)

[4.1 Công cụ kiểm thử tự động Selenium 38](#_Toc26374)

[4.2 Công cụ kiểm thử tự động TestComplete 41](#_Toc10671)

[4.3 Công cụ kiểm thử tự động OWASP ZAP 51](#_Toc12831)

[4.4 Công cụ kiểm thử tự động LambdaTest 59](#_Toc30482)

[**CHƯƠNG 5: KẾT LUẬN 63**](#_Toc18677)

[TÀI LIỆU THAM KHẢO 64](#_Toc20720)

# CHƯƠNG 1: GIỚI THIỆU KIỂM THỬ

## Khái niệm

Kiểm thử phần mềm là quá trình thực thi một hệ thống phần mềm để xác định xem phần mềm có đúng với đặc tả không và thực hiện trong môi trường như mong đợi hay không. Mục đích của kiểm thử phần mềm là để kiểm tra xem phần mềm có đáp ứng với nhu cầu của khách hàng và phù hợp với đặc tả yêu cầu, đảm bảo chất lượng và tính chính xác của ứng dụng

## Các quy tắc trong kiểm thử

Trong kiểm thử phần mềm có 7 nguyên tắc kiểm thử. Tìm hiểu chúng là 1 điều rất quan trọng bởi vì nó giúp tiết kiệm thời gian cũng như công sức truy lùng bug ẩn trong các ứng dụng của bạn. Có một câu hỏi đặt ra như thế này: "Bạn có thật sự hiểu thế nào là nguyên tắc?".

Về cơ bản, nguyên tắc chính là những quy định hoặc là luật mà chúng ta phải tuân theo. Hoặc bạn cũng có thể hiểu nguyên tắc là điều thiết yếu cần phải theo đuổi để phát triển hệ thống tốt nhất. Việc bỏ qua bất kỳ nguyên tắc nào cũng có thể làm giảm hiệu quả của việc thực hiện hệ thống. Vậy chúng ta sẽ nói về 7 nguyên tắc trong kiểm thử phần mềm:

* Kiểm thử đưa ra lỗi: Kiểm thử có thể cho thấy rằng phần mềm đang có lỗi, nhưng không thể chứng minh rằng phần mềm không có lỗi. Kiểm thử được thực hiện bằng những kĩ thuật khác nhau. Kiểm thử làm giảm xác suất lỗi chưa tìm thấy vẫn còn trong phần mềm, ngay cả khi đã kiểm thử nghiêm ngặt phần mềm vẫn có thể còn lỗi. Vì vậy chúng ta phải tìm được càng nhiều lỗi càng tốt.
* Kiểm thử cạn kiệt là không thể: Nguyên tắc này nói rằng kiểm tra mọi thứ trong phần mềm một cách trọn vẹn là không thể. Kiểm thử với tất cả các kết hợp đầu vào và đầu ra, với tất cả các kịch bản là không thể trừ phi nó chỉ bao gồm ít trường hợp thì có thể kiểm thử toàn bộ. Thay vì kiểm thử toàn bộ, việc phân tích rủi ro và dựa trên sự mức độ ưu tiên chúng ta có thể tập trung việc kiểm thử vào một số điểm cần thiết, có nguy cơ lỗi cao hơn.
* Kiểm thử càng sớm càng tốt: Thông thường, lỗi tập trung vào những module, thành phần chức năng chính của hệ thống. Nếu xác định được điều này bạn sẽ tập trung vào tìm kiếm lỗi quanh khu vực được xác định. Nó được coi là một trong những cách hiệu quả nhất để thực hiện kiểm tra hiệu quả.
* Sự tập trung của lỗi: Thông thường, lỗi tập trung vào những module, thành phần chức năng chính của hệ thống. Nếu xác định được điều này bạn sẽ tập trung vào tìm kiếm lỗi quanh khu vực được xác định. Nó được coi là một trong những cách hiệu quả nhất để thực hiện kiểm tra hiệu quả.
* Nghịch lí thuốc trừ sâu: Nếu bạn sử dụng cùng một tập hợp các trường hợp kiểm thử liên tục, sau đó một thời gian các trường hợp kiểm thử không tìm thấy lỗi nào mới. Hiệu quả của các trường hợp kiểm thử bắt đầu giảm xuống sau một số lần thực hiện, vì vậy luôn luôn chúng ta phải luôn xem xét và sửa đổi các trường hợp kiểm thử trên một khoảng thời gian thường xuyên.
* Kiểm thử phụ thuộc vào ngữ cảnh: Theo nguyên tắc này thì việc kiểm thử phụ thuộc vào ngữ cảnh và chúng ta phải tiếp cận kiểm thử theo nhiều ngữ cảnh khác nhau Nếu bạn đang kiểm thử ứng dụng web và ứng dụng di động bằng cách sử dụng chiến lược kiểm thử giống nhau, thì đó là sai. Chiến lược để kiểm thử ứng dụng web sẽ khác với kiểm thử ứng dụng cho thiết bị di động của Android.
* Không có lỗi - Sai lầm: Việc không tìm thấy lỗi trên sản phẩm không đồng nghĩa với việc sản phẩm đã sẵn sàng để tung ra thị trường. Việc không tìm thấy lỗi cũng có thể là do bộ trường hợp kiểm thử được tạo ra chỉ nhằm kiểm tra những tính năng được làm đúng theo yêu cầu thay vì nhằm tìm kiếm lỗi mới.

## Các phương pháp trong kiểm thử

* + 1. *Kiểm thử hộp đen*

Kiểm thử hộp đen là một phương pháp kiểm thử mà các tester không cần quan tâm đến các hoạt động bên trong hệ thống chạy ra sao, không cần quan tâm đến các dòng lệnh bên trong hệ thống hệ thống như thế nào. mà chỉ cần tập trung vào các giá trị đầu vào và các giá trị đầu ra của hệ thống có đúng với kết quả mong đợi của các trường hợp kiểm thử không để từ đó đánh giá chất lượng hệ thống. Chính vì cơ chế như vậy nên phương pháp này có các ưu nhược điểm như sau:

Ưu điểm:

* Được độ bao phủ lớn nhất khi thực hiện thiết kế các trường hợp kiểm thử sau này

- Hiệu quả khi số lượng các dòng lệnh của hệ thống lớn.

- Không cần truy cập vào các dòng lệnh

- Phân biệt được rõ quan điểm của người dùng với quan điểm của nhà phát triển

- Không đòi hỏi những kiến thức về ngôn ngữ lập trình ở các tester

Nhược điểm:

- Giới hạn ở độ bao phủ của các trường hợp kiểm thử

- Không hiệu quả bởi các tester bị giới hạn kiến thức về hệ thống

- Độ bao phủ thiếu vì tester không kiểm tra được các đoạn lệnh hoặc tập trung vào các dòng lệnh dễ xảy ra lỗi

- Khó để có thể thiết kế đầy đủ các trường hợp kiểm thử

* + 1. *Kiểm thử hộp trắng*

Kiểm thử hộp trắng là việc nghiên cứu cụ thể chi tiết từng luồng hoạt động cũng như các dòng lệnh bên trong hệ thống. Kiểm thử hộp trắng cũng được gọi dưới các cái tên khác như: Glass testing hay open-box testing. Kiểm thử hộp trắng đòi hỏi tester phải có kiến thức về ngôn ngữ lập trình. Tester sẽ cần phải nghiên cứu vào bên trong hê thống cụ thể là các dòng lệnh để tìm hiểu chúng có chạy đúng hay không. Dưới đây là các ưu nhược điểm của phương pháp này:

Ưu điểm:

* Các tester có kiến thức về ngôn ngữ lập trình sẽ dễ dàng phát hiện ra những lỗi ở các dòng lệnh
* Tối ưu hóa các dòng lệnh của hệ thống
* Các dòng lệnh không cần thiết hoặc các dòng lệnh có khả năng mang tới lỗi tiềm ẩn sẽ bị loại bỏ
* Các tester có kiến thức lập trình sau khi đã thực hiện phương pháp này sẽ dễ dàng đạt được độ bao phủ lớn nhất khi thực hiện thiết kế các trường hợp kiểm thử sau này

Nhược điểm:

* Việc sử dụng các tester có kiến thức về ngôn ngữ lập trình sẽ làm tăng giá thành để phát triển phần mềm
* Đôi lúc không khả thi khi kiểm tra chi tiết từng dòng lệnh để có thể từ đó phát hiện ra các lỗi tiềm ẩn của hệ thống, có rất nhiều các luồng không thể kiểm tra được
* Khó có thể duy trì phương pháp này liên tục, cần phải có tool chuyên dụng như tool về phân tích code hay tool về phát hiện và sửa lỗi
  + 1. *Kiểm thử hộp xám*

Kiểm thử hộp xám là một phương pháp kiểm thử mà đòi hỏi tester phải có một lượng kiến thức nhất định về các luồng hoạt động ở bên trong hệ thống. Khác với kiểm thử hộp đen, phương pháp mà tester chỉ quan tâm duy nhất để việc kiểm thử thông qua giao diện người dùng, kiểm thử hộp xám đòi hỏi tester phải truy cập vào các tài liệu thiết kế hệ thống cũng như hệ thống cơ sở dữ liệu của hệ thống. Do đó mà tester có thể chuẩn bị tốt hơn những dữ liệu cho việc kiểm thử cũng như các trường hợp kiểm thử trong quá trình lên kế hoạch kiểm thử hệ thống.

Ưu điểm:

* Vì là sự kết hợp giữa kiểm thử hộp trắng và kiểm thử hộp đen nên có được ưu điểm của cả hai phương pháp này

Trong phương pháp này các tester có thể thiết kế nên những trường hợp kiểm thử đặc biệt xung quanh các giao thức kết nối và các loại dữ liệu khác nhau.

- Việc kiểm thử được hoàn thành từ góc nhìn của người dùng chứ không phải từ nhà thiết kế

- Các tester sử dụng phương pháp này không dựa vào các dòng lệnh của hệ thống mà chủ yếu dựa trên các tài liệu định nghĩa giao diện cũng như các tài liệu đặc tả chức năng.

Nhược điểm:

* Vì phương pháp này không dựa trên việc truy cập code của hệ thống nên sẽ không tránh được việc độ bao phủ của các trường hợp kiểm thử bị giới hạn.
* Khi sử dụng phương pháp này thì nhiều trường hợp kiểm thử có thể bị dư thừa nếu mà những nhà thiết kế phần mềm tốn thời gian là vừa phải.
* Không thích hợp để kiểm tra các thuật toán trong hệ thống.
* Các miền dữ liệu có thể sẽ được test nếu các tester có kiến thức về nó.
  + 1. *So sánh các phương pháp kiểm thử*

|  |  |  |
| --- | --- | --- |
| **Kiểm thử hộp đen** | **Kiểm thử hộp trắng** | **Kiểm thử hộp xám** |
| Không cần quan tâm đến các luồng hoạt động trong hệ thống. | Cần nắm được toàn bộ các luồng hoạt động bên trong hệ thống. | Cần có kiến thức nhất định về các luồng hoạt động bên trong hệ thống |
| Được biết đến với các tên gọi khác như: closed-box testing, data-driven testing hoặc functional testing. | Được biết đến với các tên gọi khác như: clear-box testing hoặc code-based  testing. | Được biết đến với các tên gọi khác như: translucent testing. |
| Được thực hiện bởi người dùng cuối, kiểm thử viên và lập trình viên. | Thường thì được hoàn thành bởi kiểm thử viên và lập trình viên. | Được thực hiện bởi người dùng cuối, kiểm thử viên và lập trình viên. |

# CHƯƠNG 2: THIẾT KẾ TEST CASE

## 2.1 Các phương pháp thiết kế test-case (Black-box Testing)

## 2.1.1 Kỹ thuật phân vùng tương đương (Equivalence Partitioning)

Phân vùng tương đương (Equivalence Class): 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.

* Phân vùng tương đương: 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 đích : 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.
* Thiết kế ca kiểm thử cho phân vùng tương đương dựa trên sự đánh giá về các vùng tương đương với một điều kiện vào. Vùng tương đương biểu thị một tập cho các trạng thái hợp lệ hay không hợp lệ đối với điều kiện vào
* Thiết kế test case bằng phân vùng tương đương tiến hành theo hai bước:
  + Xác định các lớp tương đương
  + Xác định các ca kiểm thử

Nếu có lỗi xảy ra thì các giá trị khác trong class đồng giá trị cũng sẽ có lỗi giống

nhau.

## 2.1.2 Kỹ thuật phân tích giá trị biên (Boundary Value Analysis)

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.

* Test giá trị biên được thực hiện theo trình tự dưới đây:
  + Tìm ra đường biên
  + Quyết định giá trị biên
  + Quyết định giá trị để test
  + Giá trị biên.
  + Dưới giá trị biên. (Nếu là class đồng giá trị)
  + Trên 1 giá trị biên. (Nếu là class đồng giá trị)
* Boundary values:
  + Min – 1
  + Min
  + Max
  + Max + 1

### *2.1.3 Bảng quyết định (Decision Table)*

* Làm giảm số lượng test case không cần thiết so với 2 kỹ thuật trên vì nó loại trừ các phép kết hợp không cần thiết giữa các giá trị biến đầu vào.
* Liệt kê nguyên nhân (cause) – kết quả (result) trong một ma trận. Mỗi cột ma trận đại diện cho 1 phép kết hợp giữa các cause trong trong việc tạo ra 1 result.
* Các bước để tạo bảng quyết định
  + Liệt kê các nguyên nhân trong bảng quyết định
  + Tính tổng số lượng kết hợp giữa các cause
  + Điền vào các cột với tất cả các kết hợp có thể có
  + Rút bớt số lượng các phép kết hợp dư thừa
  + Kiểm tra các phép kết hợp có bao phủ hết mọi trường hợp hay không
  + Bổ sung kết quả vào bảng quyết định

### *2.1.4 Dịch chuyển trạng thái*

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.

## 2.2 Kiểm thử hộp trắng (White-box Testing)

Kiểm thử hộp trắng (White box testing) là một kỹ thuật xác minh giúp các kỹ sư phần mềm có thể sử dụng để kiểm tra mã code của họ hoạt động như dự kiến.

Có các loại white box testing đang tồn tại như sau:

* API testing (application programming interface) – Kiểm thử ứng dụng bằng cách sử dụng các hàm API public và private
* Code coverage – Là việc tạo các trường hợp test để thỏa mãn một số điều kiện bao phủ code - code coverage (ví dụ như, người thiết kế test có thể tạo ra các trường hợp test sao cho tất cả các câu lệnh của chương trình đều được thực thi ít nhất 1 lần)
* Fault injection methods – cải tiến bao phủ một trường hợp bằng cách đưa một số lỗi vào để test các đường dẫn code.
* Mutation testing methods
* Static testing - White box testing bao gồm tất cả các phương pháp kiểm thử tĩnh (ví dụ review code).

*Kiểm thử độ bao phủ*

Phương pháp kiểm thử white box cũng có thể được sử dụng để ước lượng tính trọn vẹn đầy đủ của các tập hợp kiểm thử (test suit) đã được tạo ra bằng phương pháp kiểm thử black box. Điều này cho phép nhóm sản xuất phần mềm xem xét lại các phần của hệ thống ít được test nhất và để chắc chắn rằng các chức năng quan trọng nhất đã được tập trung test kỹ.

*Hai hình thức chung của kiểm thử độ bao phủ code:*

* Bao phủ chức năng - Function coverage, dựa trên việc thực thi các chức năng.
* Bao phủ câu lệnh - Statement coverage, dựa trên số lượng các dòng lệnh đã được thực thi để hoàn thành kiểm thử.

## 2.3 Mô tả đề tài và kế hoạch kiểm thử

### *2.3.1 Mô tả đề tài*

Trong thời đại công nghệ phát triển mạnh mẽ, việc giải trí qua các nền tảng số đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày. Đề tài phát triển một website GameShop trực tuyến không chỉ đáp ứng nhu cầu giải trí đa dạng của người dùng mà còn mang lại giá trị kinh tế và cơ hội kinh doanh lớn cho các nhà phát triển.

Website ShopGame sẽ cung cấp một nền tảng nơi người dùng có thể dễ dàng truy cập và chọn lựa được các game thuộc nhiều thể loại khác nhau, từ hành động nhập vai, kinh dị đến giải đố.

Mục tiêu của đề tài là tạo ra một Website thân thiện với người dùng, có giao diện đẹp mắt và dễ sử dụng, đáp ứng tốt nhu cầu giải trí của mọi người. Đây không chỉ là một dự án công nghệ mà còn là một cơ hội kinh doanh tiềm năng, mang lại lợi ích cho cả người dùng và nhà phát triển.

### *2.3.2 Kế hoạch kiểm thử*

Kiểm thử hộp đen

|  |  |
| --- | --- |
| **Kỹ thuật kiểm thử** | **Chức năng phần mềm** |
| * Phân vùng tương đương | * Xem chi tiết game |
| * Phân tích giá trị biên | * Đăng ký |
| * Bảng quyết định | * Mua tài khoản game |
| * Dịch chuyển trạng thái | * Đăng nhập |

Kiểm thử hộp trắng

|  |  |
| --- | --- |
| **Kỹ thuật kiểm thử** | **Chức năng phần mềm** |
| * Phủ câu lệnh | * Đăng nhập |
| * Phủ nhánh | * Chỉnh sửa thông tin tài khoản game |
| * Phủ đường | * Đăng ký |
| * Phủ điều kiện | * Thêm acc game mới vào hệ thống |

Kiểm thử tự động

|  |  |
| --- | --- |
| **Loại kiểm thử** | **Công cụ kiểm thử** |
| * Kiểm thử chức năng (Đăng nhập) | * Katalon |
| * Kiểm thử chức năng (Đăng ký) | * Katalon |
| * Kiểm thử giao diện | * Katalon |

# CHƯƠNG 3: ÁP DỤNG

## 3.1 Đặc tả

## 3.1.1 Đăng ký

**Mục tiêu**: Cung cấp cho người dùng khả năng tạo tài khoản cá nhân, đảm bảo tính bảo mật và dễ sử dụng. Việc đăng ký giúp hệ thống quản lý người dùng, lưu trữ lịch sử giao dịch.

**Các bước thực hiện:**

Bước 1: Nhập thông tin cá nhân

Email:

* Yêu cầu định dạng đúng của email (ví dụ: admin@gmail.com).
* Độ dài tối đa không quá 30 ký tự.
* Email phải là duy nhất trong hệ thống. Nếu email đã được đăng ký, hệ thống sẽ báo lỗi yêu cầu người dùng nhập một email khác.

Mật khẩu:

* Độ dài từ 8 đến 16 ký tự.
* Mật khẩu phải bao gồm ít nhất:
  + 1 ký tự chữ hoa (A-Z).
  + 1 ký tự chữ thường (a-z).
  + 1 chữ số (0-9).
  + 1 ký tự đặc biệt (ví dụ: !, @, #).
* Người dùng cần nhập mật khẩu hai lần để xác nhận và đảm bảo không có lỗi nhập liệu. Nếu mật khẩu không khớp, hệ thống sẽ yêu cầu nhập lại.

Bước 2: Xác nhận tính duy nhất của tài khoản

Hệ thống kiểm tra xem email đã được sử dụng hay chưa. Nếu email đã tồn tại trong hệ thống, thông báo lỗi sẽ được hiển thị với yêu cầu nhập email khác.

Bước 3: Kiểm tra tính hợp lệ của mật khẩu

Hệ thống sẽ xác nhận rằng mật khẩu thỏa mãn tất cả các tiêu chí bảo mật. Nếu mật khẩu không đáp ứng, người dùng sẽ nhận được thông báo yêu cầu thay đổi mật khẩu theo các tiêu chí đã đề ra.

Bước 4: Xác thực tài khoản qua Email (nếu có)

* Sau khi hoàn tất đăng ký, hệ thống sẽ gửi một email xác thực tới địa chỉ email đã đăng ký. Người dùng cần kiểm tra hộp thư và nhấn vào đường dẫn xác nhận để kích hoạt tài khoản.
* Khi xác thực thành công, tài khoản sẽ chính thức được kích hoạt và người dùng có thể đăng nhập vào hệ thống.

**Các ràng buộc bảo mật:**

* Tất cả các mật khẩu sẽ được mã hóa trước khi lưu trữ trong cơ sở dữ liệu để bảo vệ thông tin cá nhân người dùng.
* Các thông tin đăng ký chỉ được chấp nhận khi đáp ứng đầy đủ yêu cầu về định dạng và bảo mật.

|  |  |
| --- | --- |
| **Tên Use-Case** | Đăng ký tài khoản |
| **Actor** | Người dùng. |
| **Mô tả** | Use-case cho phép user tạo tài khoản để truy cập vào hệ thống để thực hiện những chức năng của mình. |
| **Điều kiện kích hoạt** | Khi người dùng chọn chức năng đăng ký tài khoản từ trang chủ của hệ thống. |
| **Hậu điều kiện** | Người dùng đăng ký thành công. |
| **Luồng sự kiện chính** | 1. Hệ thống hiển thị màn hình đăng ký. 2. Người dùng nhập thông tin cá nhân như tên , năm sinh email, tên đăng nhập và mật khẩu. 3. Hệ thống hiển thị kiểm tra thông tin đăng ký. 4. Nếu thành công hệ thống hiển thị thành công, tạo tài khoản mới và truy cập vào trang chủ bằng tài khoản vừa tạo. 5. Kết thúc Use-Case. |
| **Luồng sự kiện phụ** | **A1-Tài khoản không hợp lệ**: khi người dùng nhập sai tên thông tin cá nhân  1.Hệ thống hiển thị lại màn hình đăng nhập để người dùng nhập lại thông tin và màn hình sẽ thông báo sai thông tin cá nhân  2.Quay lại bước 2 trong luồng sự kiện chính. |

## 3.1.2 Đăng nhập

**Mục tiêu:** Cung cấp cho người dùng quyền truy cập vào tài khoản đã đăng ký để sử dụng các tính năng riêng như lưu trữ lịch sử, danh sách tài khoản game, và quản lý tài khoản cá nhân. Đăng nhập phải bảo đảm tính bảo mật cao và dễ dàng thao tác.

**Các bước thực hiện**:

Bước 1: Nhập thông tin đăng nhập

Email:

* Yêu cầu đúng định dạng email đã đăng ký và không quá 30 ký tự.
* Email phải tồn tại trong cơ sở dữ liệu của hệ thống.

Mật khẩu:

* Độ dài mật khẩu từ 8 đến 16 ký tự.
* Mật khẩu phải khớp với mật khẩu đã đăng ký.

Bước 2: Xác thực thông tin đăng nhập

* Hệ thống sẽ kiểm tra xem email có tồn tại trong cơ sở dữ liệu hay không. Nếu email không tồn tại, hệ thống sẽ thông báo rằng email chưa được đăng ký và yêu cầu người dùng tạo tài khoản mới.
* Hệ thống xác nhận tính chính xác của mật khẩu đã nhập:
* Nếu mật khẩu sai, hệ thống sẽ hiển thị thông báo yêu cầu nhập lại.

Bước 3: Đăng nhập thành công

Sau khi xác thực thành công, người dùng sẽ được chuyển hướng đến trang chính của hệ thống và có thể truy cập các tính năng cá nhân.

**Xử lý khi đăng nhập thất bại:**

Yêu cầu nhập lại.

**Các ràng buộc bảo mật:**

Mã hóa thông tin đăng nhập: Hệ thống mã hóa mật khẩu và các thông tin bảo mật khác để bảo vệ dữ liệu.

|  |  |
| --- | --- |
| **Tên Use-Case** | Đăng Nhập |
| **Actor** | Quản trị viên và người dùng. |
| **Mô tả** | Use-case cho phép admin hoặc user truy cập vào hệ thống để thực hiện những chức năng của mình. |
| **Điều kiện kích hoạt** | Khi người dùng chọn chức năng đăng nhập từ trang chủ của hệ thống. |
| **Tiền điều kiện** | Người dùng phải có tài khoản trên hệ thống. |
| **Hậu điều kiện** | Người dùng đăng nhập thành công. |
| **Luồng sự kiện chính** | 1. Hệ thống hiển thị màn hình đăng nhập. 2. Người dùng nhập tên đăng nhập và mật khẩu. 3. Hệ thống hiển thị kiểm tra thông tin đăng nhập. 4. Nếu thành công hệ thống hiển thị thành công và truy cập đến trang chủ. 5. Kết thúc Use-Case. |
| **Luồng sự kiện phụ** | **A1-Mật khẩu không hợp lệ**: khi người dùng nhập sai tên đăng nhập và mật khẩu  1.Hệ thống hiển thị lại màn hình đăng nhập để người dùng nhập lại thông tin và màn hình sẽ thông báo mật khẩu hoặc tài khoản bị sai.  2.Quay lại bước 2 trong luồng sự kiện chính.  **A2-Quên mật khẩu:** khi người dùng chọn chức năng quên mật khẩu trên màn hình đăng nhập  1.Hệ thống hiển thị màn hình để người dùng nhập email.  2.Người dùng nhập email và chọn chức năng lấy lại mật khẩu.  3.Hệ thống kiểm tra email hợp lệ và gửi liên kết để reset mật khẩu cho người dùng qua email.  4.Hệ thống hiển thị màn hình thông báo thành công  5.Use-case kết thúc. |

## 3.1.4 Mua bán acc game

Đây là tính năng cốt lõi của website ShopGame

* Giao dịch tài khoản trực tuyến: Người dùng có thể xem chi tiết tài khoản, bao gồm thông tin game, cấp độ, vật phẩm, và hình ảnh minh họa trước khi mua.
* Điều chỉnh hiển thị tài khoản: Hệ thống hỗ trợ sắp xếp, lọc tài khoản theo game, mức giá, cấp độ, và trạng thái (còn hàng, đã bán) để người dùng dễ tìm kiếm.
* Lịch sử giao dịch: Hệ thống lưu lại lịch sử mua bán tài khoản, giúp người dùng dễ dàng theo dõi các giao dịch đã thực hiện và tra cứu lại thông tin khi cần..

|  |  |
| --- | --- |
| **Tên Use-Case** | Mua bán acc game |
| **Actor** | Người dùng. |
| **Mô tả** | Use-case cho phép user truy cập vào website để chọn và mua |
| **Điều kiện kích hoạt** | Khi người dùng chọn acc game muốn mua từ trang chủ của hệ thống. |
| **Tiền điều kiện** | Người dùng phải có tài khoản trên hệ thống. |
| **Hậu điều kiện** | Người dùng mua acc game thành công. |
| **Luồng sự kiện chính** | 1. Hệ thống hiển thị màn hình trang chủ. 2. Người dùng chọn vào acc game cần mua 3. Hệ thống hiển thị kiểm tra thông tin acc game 4. Nếu thành công hệ thống hiển thị acc game 5. Kết thúc Use-Case. |

## 3.1.5 Quản lý trang Admin

Trang admin là công cụ quan trọng để quản trị viên quản lý toàn bộ acc game và tương tác của người dùng, đảm bảo chất lượng dịch vụ và bảo mật.

Quản lý tài khoản game:

* Cho phép admin thêm mới, xóa, và chỉnh sửa thông tin acc game bao gồm tiêu đề, mô tả, thể loại.

Báo cáo và Thống kê:

* Cung cấp báo cáo tổng quan về lượng người truy cập, số lượng tài khoản đã bán và các tài khoản game bán chạy nhất.
  + Thống kê doanh thu, lượng giao dịch theo thời gian, theo thể loại game, giúp admin đánh giá hiệu quả hoạt động và định hướng phát triển nội dung phù hợp.

Trang admin không chỉ là trung tâm quản lý nội dung mà còn giúp duy trì chất lượng, bảo mật cho hệ thống và tạo trải nghiệm tốt nhất cho người dùng.

|  |  |
| --- | --- |
| **Tên Use-Case** | Quản lý tài khoản game |
| **Actor** | Quản trị viên (Admin) |
| **Mô tả** | Use-case cho phép Admin:  1. Thêm mới tài khoản game vào hệ thống  2. Chỉnh sửa thông tin tài khoản game đã có  3. Xóa tài khoản game khỏi hệ thống |
| **Điều kiện kích hoạt** | Khi Admin chọn chức năng: Thêm mới, Cập nhật, hoặc Xóa acc game trên giao diện quản lý |
| **Tiền điều kiện** | Admin phải đăng nhập thành công vào hệ thống bằng tài khoản quản trị |
| **Hậu điều kiện** | Admin đã thêm mới, cập nhật hoặc xóa tài khoản game thành công |
| **Luồng sự kiện chính** | 1. Hệ thống hiển thị danh sách tất cả tài khoản game đã có. 2. Admin chọn một tài khoản game để thực hiện thao tác Thêm mới / Cập nhật / Xóa.   **2.1 Admin chọn chức năng Thêm mới:**  Hệ thống hiển thị biểu mẫu nhập thông tin tài khoản game.  Admin nhập các thông tin như: Tên game, server, rank, cấp độ, giá bán, mô tả, hình ảnh.  Hệ thống kiểm tra dữ liệu nhập vào.  Nếu hợp lệ, hiển thị thông báo “Thêm tài khoản thành công”.  **2.2 Admin chọn chức năng Cập nhật:**  Hệ thống hiển thị biểu mẫu chứa thông tin hiện tại của tài khoản game.  Admin chỉnh sửa thông tin cần cập nhật.  Hệ thống kiểm tra dữ liệu mới.  Nếu hợp lệ, hiển thị thông báo “Cập nhật thành công”.  **2.3 Admin chọn chức năng Xóa:**  Hệ thống hiển thị thông báo xác nhận xóa.  Admin nhấn xác nhận.  Hệ thống xóa tài khoản game khỏi hệ thống và hiển thị thông báo “Xóa thành công”.  Kết thúc use-case. |
| **Luồng sự kiện phụ** | A1 – Thêm mới thất bại:   * Khi Admin nhập thiếu thông tin hoặc sai định dạng. * Hệ thống hiển thị thông báo lỗi → Quay lại bước 2.1.   A2 – Cập nhật thất bại:   * Khi Admin nhập thiếu thông tin hoặc sai định dạng. * Hệ thống hiển thị thông báo lỗi → Quay lại bước 2.2. |

## 3.2 Kiểm thử hộp đen

*3.2.1 Phương pháp phân vùng tương đương (Xem tài khoản game)*

Các Test Case:

• Test Case 1: Xem chi tiết tài khoản khi đã đăng nhập

→ Tài khoản game hiển thị đầy đủ thông tin (tên game, rank, skin, giá, mô tả), hình ảnh rõ nét, không bị lỗi hoặc thiếu nội dung.

• Test Case 2: Xem tài khoản khi không có kết nối mạng

→ Hiển thị thông báo: "Không có kết nối mạng, vui lòng thử lại sau", không hiển thị chi tiết tài khoản.

• Test Case 3: Xem tài khoản khi mạng yếu hoặc chậm

→ Trang hiển thị chậm, hệ thống hiện thông báo "Đang tải...". Hình ảnh có thể nén lại để tải nhanh hơn.

• Test Case 4: Xem tài khoản có hình ảnh chất lượng cao

→ Hình ảnh tài khoản hiển thị sắc nét, đầy đủ, không bị vỡ hay mờ.

• Test Case 5: Xem tài khoản có hình ảnh chất lượng thấp

→ Hình ảnh hiển thị mờ hơn nhưng tải nhanh, vẫn đảm bảo đầy đủ thông tin cần thiết.

• Test Case 6: Xem tài khoản không tồn tại

→ Hiển thị thông báo: "Tài khoản không tồn tại hoặc đã bị gỡ", không hiển thị thông tin.

• Test Case 7: Xem tài khoản khi chưa đăng nhập (yêu cầu đăng nhập)

→ Hiển thị thông báo: "Vui lòng đăng nhập để xem chi tiết tài khoản" và chuyển đến trang đăng nhập.

• Test Case 8: Tiếp tục xem tài khoản đã xem trước đó

→ Hệ thống tự động chuyển đến đúng tài khoản trong lịch sử mà người dùng đã xem, không cần tìm lại.

• Test Case 9: Xem tài khoản bị giới hạn quyền truy cập

→ Hiển thị thông báo: "Bạn cần nâng cấp tài khoản để xem thông tin này" và không hiển thị chi tiết.

• Test Case 10: Xem tài khoản có mô tả tiếng Việt

→ Hiển thị mô tả tiếng Việt rõ ràng, đúng ngữ pháp, không lỗi font.

• Test Case 11: Xem tài khoản không có mô tả

→ Hiển thị mặc định thông báo: "Chưa có mô tả cho tài khoản này".

• Test Case 12: Xem tài khoản với thông tin đa ngôn ngữ

→ Cho phép chọn ngôn ngữ hiển thị khác (ví dụ: tiếng Anh), hệ thống hiển thị đúng ngôn ngữ đã chọn..

### *3.2.2 Kiểm thử phân tích giá trị biên (Đăng ký)*

L   : Trang Login

H  : Trang Home

M1: Vui lòng nhập username

M2: Vui lòng nhập password

M3: Username không hợp lệ

M4: Password không hợp lệ

M5: Vui lòng nhập SDT

M6: SDT không hợp lệ

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | **C1** | **C2** | **C3** | **C4** | **C5** | **C6** | **C7** |
| **Điều kiện** | Username |  | I | V | V | V | V | V |
| Password | - | - | B | I | V | V | V |
| SDT | - | - | - | - | B | I | V |
| **Hành động** | Thông báo lỗi | M1 | M3 | M2 | M4 | M5 | M6 |  |
| Chuyển trang | L | L | L | L | L | L | H |

Bảng quyết định:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cột** | **Đầu vào** | | | **Hành động** | |
| **username** | **password** | **SDT** | **Thông báo** | **Chuyển trang** |
| **C1** |  | Aa12345@ | 0232335643 | M1 | L |
| **C2** | vinh2233@gmail.com | Aa12345@ | 0232335643 | M3 | L |
| vinh2233@gmail.com | Aa12345@ | 0232335643 | M3 | L |
| vinh2233@gmail.com | Aa12345@ | 0232335643 | M3 | L |
| vinh2233@gmail.com | Aa12345@ | 0232335643 | M3 | L |
| **C3** | vinh2233@gmail.com |  | 0232335643 | M2 | L |
| **C4** | vinh2233@gmail.com | Aa123@  (6 ký tự) | 0232335643 | M4 | L |
| vinh2233@gmail.com | Aa34567890123456@  ( 17 ký tự) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABCDEFGH**  (chỉ chữ Hoa) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **!@#$%^&\***  (chỉ  ký tự đặc biệt) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **Abcdefgh**  (Chỉ chữ thường) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **12345678**  (chỉ số) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABCDabcd**  (chữ Hoa và chữ thường) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABCD!@#$**  (chữ Hoa và ký tự đặc biết) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABCD1234**  (chữ Hoa và số) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **abcd!@#$**  (chữ thường và ký tự đặc biệt) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **abcd1234**  (chữ thường và số) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **!@#$1234**  (ký tự đặc biệt và số) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABCabc!@**  (chữ Hoa, chữ thường và ký tự đặc biệt) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABCabc12**  (chữ Hoa , chữ thường và số ) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **ABC!@#12**  (chữ Hoa, ký tự đặc biệt và số) | 0232335643 | M4 | L |
| vinh2233@gmail.com | **abc!@#12**  (chữ thường, ký tự đặc biệt và số) | 0232335643 | M4 | L |
| **C5** | [vinh223366@gmail.com](mailto:vinh2233@gmail.com)  (đã tồn tại) | Aa12345@  (8 ký tự) | - | M5 | L |
| **C6** | vinh2233@gmail.com  (18 ký tự) | Aa12345@  (8 ký tự) | 0232335643434 | M6 | L |
| [vinhnguyen1233211232](mailto:vinhnguyen1233211232@gmail.com)  [@gmail.com](mailto:vinhnguyen1233211232@gmail.com)  (30 ký tự) | Aa3456789012345@  (16 ký tự) | 0232335 | M6 | L |
| **C7** | vinh2233@gmail.com  (18 ký tự) | Aa12345@  (8 ký tự) | 0232335643 |  | H |
| [vinhnguyen1233211232](mailto:vinhnguyen1233211232@gmail.com)  [@gmail.com](mailto:vinhnguyen1233211232@gmail.com)  (30 ký tự) | Aa3456789012345@  (16 ký tự | 0232335643 |  | H |

### *3.2.3 Kiểm thử bảng quyết định (Tìm kiếm phim)*

B   : Rỗng (không nhập từ khoá)

I : Nhập từ khoá không hợp lệ

V1: Nhập từ khoá hợp lệ (có phim trong csdl)

V2: Nhập từ khoá hợp lệ (không có phim trong csdl)

M1: Nhập thông tin phim cần tìm

M2: Từ khoá không hợp lệ

M3: Không có phim cần tìm

T   : Trang tìm kiếm phim

H  : Trang Home

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **C1** | **C2** | **C3** | **C4** |
| **Điều kiện** | B | I | V1 | V2 |
| **Hành động** | M1 | M2 |  | M3 |
| H | H | T | H |

*Bảng quyết định:*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Cột** | **Đầu vào** | **Hành động** | | | | |
| M1 | M2 | M3 | **H** | **T** |
| **C1** | “ ” | v | x | x | v | x |
| **C2** | "a" (1 ký tự) | x | v | x | v | x |
| “abcd…” (51 ký tự) | x | v | x | v | x |
| “@@@@” | x | v | x | v | x |
| “a@” | x | v | x | v | x |
| “A@” | x | v | x | v | x |
| “1@” | x | v | x | v | x |
| “aA@” | x | v | x | v | x |
| “1a@” | x | v | x | v | x |
| “Aa1@” | x | v | x | v | x |
| “1A@” | x | v | x | v | x |
| “xyz…” (60 ký tự) | x | v | x | v | x |
| **C3** | “Do” (2 ký tự) | x | x | x | x | v |
| “Doa” (3 ký tự) | x | x | x | x | v |
| “Do…” (49 ký tự) | x | x | x | x | v |
| “The…” (50 ký tự) | x | x | x | x | v |
| “do” | x | x | x | x | v |
| “Do” | x | x | x | x | v |
| “DO” | x | x | x | x | v |
| “1010” | x | x | x | x | v |
| “E123” | x | x | x | x | v |
| “e123” | x | x | x | x | v |
| “Ab123” | x | x | x | x | v |
| “Doraemon” (23 ký tự) | x | x | x | x | v |
| **C4** | “zabc” | x | x | v | v | x |
| “xy” | x | x | v | v | x |
| “xyz” | x | x | v | v | x |
| “xyz…” (49 ký tự) | x | x | v | v | x |
| “xyz…” (50 ký tự) | x | x | v | v | x |

### *3.2.4 Kiểm thử dịch chuyển trạng thái (đăng nhập)*

Các khía cạnh của hệ thống được mô tả thông qua lược đồ trạng thái. Hệ thống sẽ có nhiều trạng thái khác nhau, sự dịch chuyển từ một trạng thái này sang trạng thái khác được quyết định bởi một sự kiện nào đó.

**Các trạng thái:**

* Logged Out (Đã đăng xuất / chưa đăng nhập) - LO
* Logging In (Đang đăng nhập) - LingI
* Logged In (Đã đăng nhập) -LedI
* Login Failed (Đăng nhập thất bại) - LF

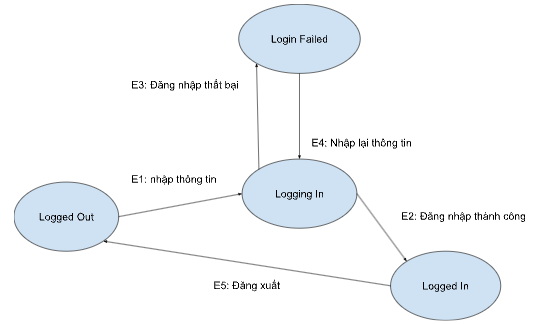
**Sự dịch chuyển:**

* Logged Out  → Logging In
* Logging In → Logged In, Login Failed
* Login Failed → Logging In
* Logged In → Logged Out

**Các sự kiện:**

* E1: Nhập thông tin đăng nhập
* E2: Thông tin chính xác → đăng nhập thành công
* E3: Thông tin không chính xác → đăng nhập thất bại
* E4: Nhập lại thông tin
* E5: Đăng xuất

**Các hành động:**



Sơ đồ lược đồ dịch chuyển

**Bảng lược đồ dịch chuyển:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Trạng thái** | **E1** | **E2** | **E3** | **E4** | **E5** |
| **LO** | LingI |  |  |  |  |
| **LingI** |  | LedI | LF |  |  |
| **LedI** |  |  |  |  | LO |
| **LF** |  |  |  | LingI |  |

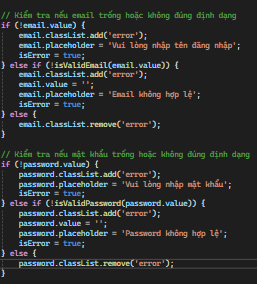
**Các TestCase:**

* Logged Out→ Logging In,
* Logging In→ Logged In,
* Logging In→ Login Failed,
* Login Failed→ Logging In,
* Logged In→ Logged Out

## 3.1 Kiểm thử hộp trắng

### *3.1.1 Phương pháp phủ câu lệnh (Đăng nhập)*

Phủ câu lệnh (statement coverage): mỗi câu lệnh được thực thi ít nhất một lần.



Chức năng đăng nhập

Bảng Test Case:

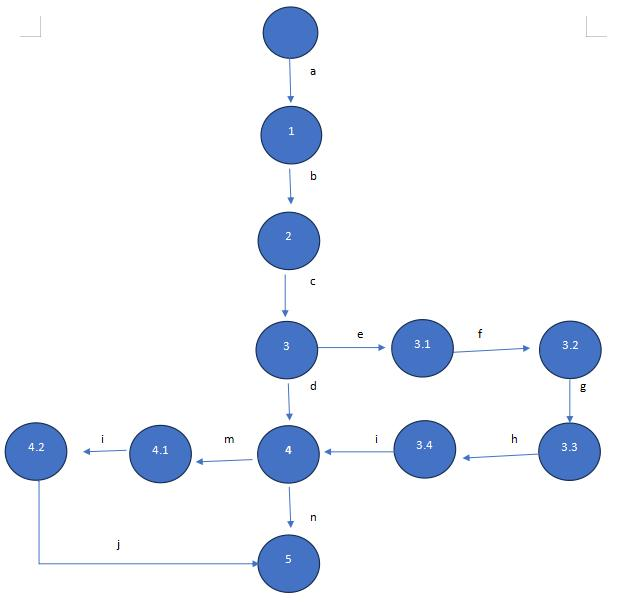
|  |  |  |
| --- | --- | --- |
| **Đầu vào** | | **Đầu ra mong muốn** |
| **Tài khoản** | **Mật khẩu** |
| admin@gmail.com | admin@123 | Đăng nhập thành công |
| admin@gmail.com | admin@124 | Mật khẩu không hợp lệ |
| admin | - | Email không hợp lệ |
| x | - | Vui lòng nhập tên đăng nhập |
| admin@gmail.com | Abcdxyz1298@LHK | Password không hợp lệ |
| admin@gmail.com | x | Vui lòng nhập mật khẩu |

### *3.1.2 Phương pháp phủ nhánh (Chỉnh sửa phim)*

Phủ nhánh có nghĩa là các cạnh được đi qua ít nhất một lần. Để phủ nhánh phải thiết kế dữ liệu kiểm thử sao cho mỗi nút vị từ (predicate) xảy ra tất cả các kết quả (true/false) có thể của nó, nên phủ nhánh còn gọi là phủ quyết định (decision coverage).



Chức năng chỉnh sửa phim



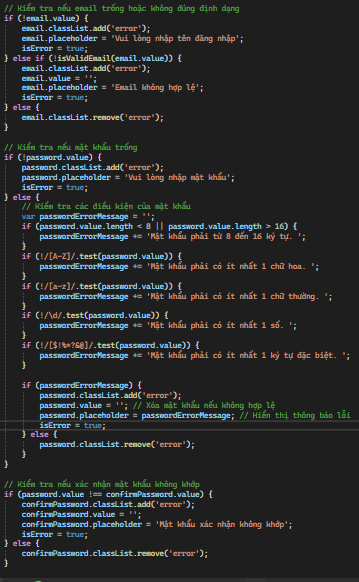
Sơ đồ kiểm thử phủ nhánh

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Đường dẫn** | **Đầu vào** | | | | **Đầu ra mong muốn** |
| **phimBo** | **anh** | **anhnen** | **theloais** |
| a,b,c,m,n | x | - | - | - | Không tìm được |
| A,,b,d | - | - | - | - | Không hợp lệ |
| A,b,c,d,l,,n,m | v | v | v | v | Thành công |
| A,b,c,l.n | x | x | x | - | Không thành công |
| A,b,c,n,m | v | v | v | - | Cập nhật thành công |
| A,b | v | x | x | x | Không update thể loại |

Bảng Test Case:

### *3.1.3 Phương pháp phủ đường dẫn (Đăng ký)*

Phủ đường dẫn (path coverage): mỗi đường dẫn qua ít nhất một lần. Đường là một tập các nhánh, nên phủ đường chắc chắn phủ nhánh, nhưng ngược lại chưa chắc đúng.



Chức năng đăng ký



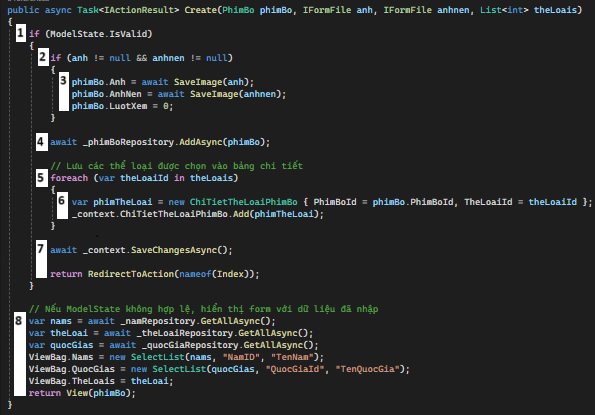
Sơ đồ kiểm thử phủ đường dẫn

Bảng Test Case:

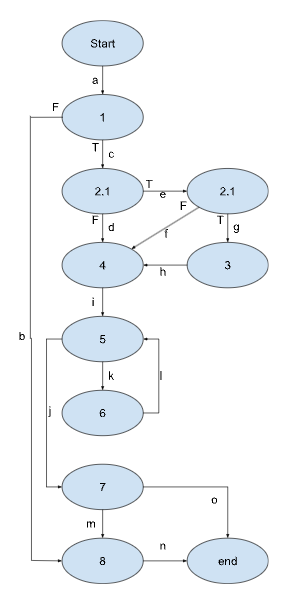
|  |  |  |  |
| --- | --- | --- | --- |
| **Đầu vào** | | | **Đầu ra mong muốn** |
| **Tài khoản** | **Mật khẩu** | **Xác nhận mk** |
| admin@gmail.com | admin@123 | admin@123 | Đăng ký thành công |
| admin@gmail.com | admin@123 | admin@124 | Mật khẩu xác nhận không khớp |
| admin | - | - | Email không hợp lệ |
| x | - | - | Vui lòng nhập tài khoản |
| admin@gmail.com | Abcdxyz1298@LHK | - | Mật khẩu từ 8 -16 ký tự |
| a@gmail.com | Abcde123 | - | Mật khẩu có ít nhât 1 chữ in hoa |
| b@gmail.com | ABCD1234 | - | Mật khẩu có ít nhất 1 chữ thường |
| c@gmail.com | ABCDabcd | - | Mật khẩu có ít nhất 1số |
| d@gmail.com | ABCD1234 | - | Mật khẩu có ít nhất 1 ký tự đặt biệt |
| e@gmail.com | x | - | Vui lòng nhập mật khẩu |

### *3.1.4 Phương pháp phủ điều kiện (Thêm phim)*

Phủ điều kiện (condition coverage): mỗi điều kiện trong các vị từ được thực hiện ít nhất một lần cho cả trường hợp true và false (không bắt buộc các kết hợp giữa chúng). Phủ điều kiện chứa chắc đảm bảo phủ các nhánh.



Chức năng thêm phim



Sơ đồ kiểm thử phủ điều kiện

Bảng Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Đường dẫn** | **Đầu vào** | | | | **Đầu ra mong muốn** |
| **phimBo** | **anh** | **anhnen** | **theloais** |
| a,c,e,g,h,i,k,l,j,o | V | V | V | V | Trả về trang Index |
| a,c,d,i,k,l,j,m,n | V | I | V | V | Hiển thị thông báo lỗi do ảnh không hợp lệ |
| a,c,e,f,i,k,l,j,m,n | V | V | I | V | Hiển thị thông báo lỗi do ảnh nền không hợp lệ |
| a,c,e,g,h,i,,j,m,n | V | V | V | I | Hiển thị thông báo lỗi do danh sách thể loại phim không hợp lệ |
| a,b,n | I | - | - | - | Trả về trang thêm với dữ liệu đã nhập |

# CHƯƠNG 4: KIỂM THỬ TỰ ĐỘNG

## 4.1 Công cụ kiểm thử tự động katalon

Kiểm thử chức năng Đăng nhập, mua acc game

### *4.1.1 Giới thiệu*

katalon Là một bộ công cụ toàn diện cho kiểm thử tự động hóa ứng dụng trên web và điện thoại di động. Công cụ này bao gồm một gói đầy đủ các tính năng mạnh mẽ giúp vượt qua những thách thức phổ biến trong tự động hóa thử nghiệm giao diện web, ví dụ như pop-up, iFrame và wait-time. Giải pháp thân thiện và linh hoạt này giúp tester thực hiện công tác kiểm tra tốt hơn, làm việc nhanh hơn và khởi chạy phần mềm chất lượng cao nhờ vào sự hông minh mà nó cung cấp cho toàn bộ quá trình tự động hóa kiểm thử.

*4.1.2 Các tính năng chính*

Tích hợp sẵn các công cụ test: Katalon tích hợp sẵn Selenium và Appium, giúp viết test nhanh mà không cần cài đặt rời rạc.

Giao diện kéo – thả dễ dùng: Hỗ trợ record & playback, người dùng không cần biết lập trình vẫn có thể tạo test case.

Hỗ trợ kiểm thử đa nền tảng: Kiểm thử được cả web, desktop, mobile (Android & iOS), và API trên cùng một nền tảng.

Tạo test suite dễ dàng: Dễ dàng nhóm nhiều test case thành một bộ test suite và chạy tự động theo lịch.

Hỗ trợ dữ liệu kiểm thử (Data-driven testing): Có thể kết nối file Excel, CSV, hoặc database để test với nhiều dữ liệu khác nhau.

Hệ thống báo cáo chi tiết: Tự động tạo report test bằng hình ảnh, log lỗi, biểu đồ kết quả trực quan.

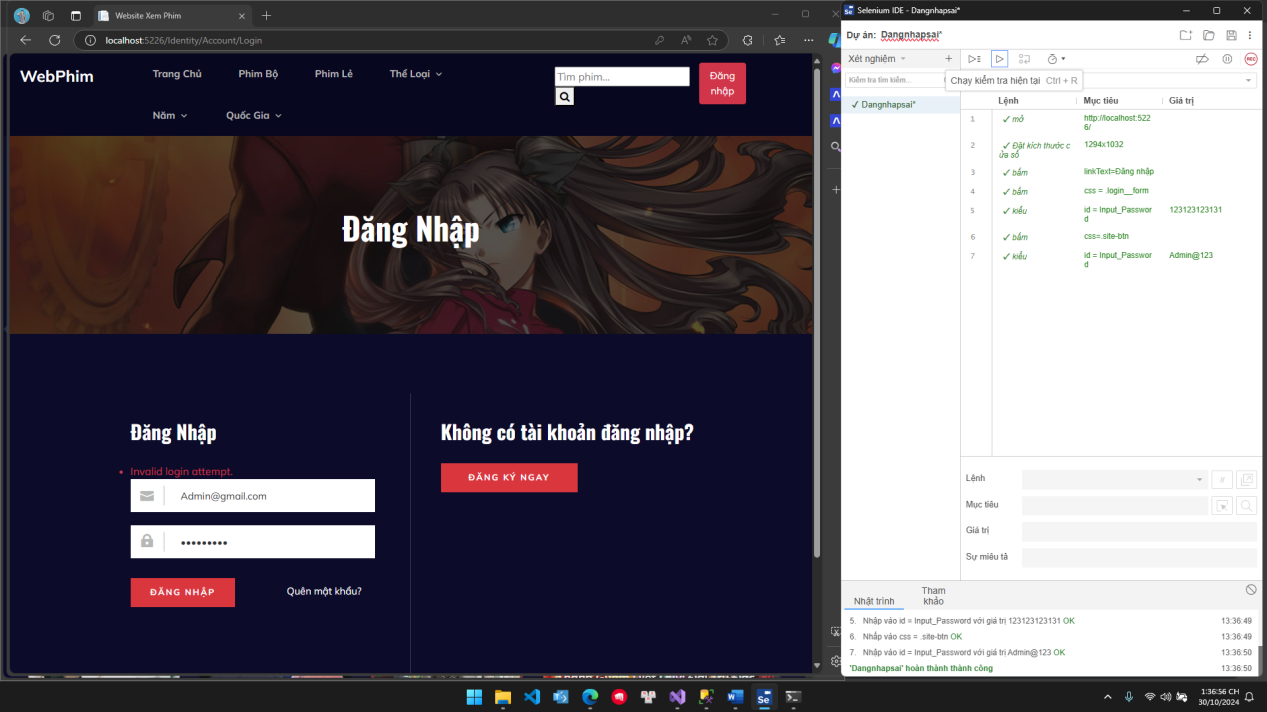
Tích hợp CI/CD: Dễ dàng kết nối với Jenkins, Git, GitLab CI/CD, Azure DevOps,… để triển khai kiểm thử tự động.

Đa dạng ngôn ngữ lập trình: Dùng ngôn ngữ Groovy (Java-based) nhưng cũng hỗ trợ người không rành code nhờ vào giao diện test script & manual mode.

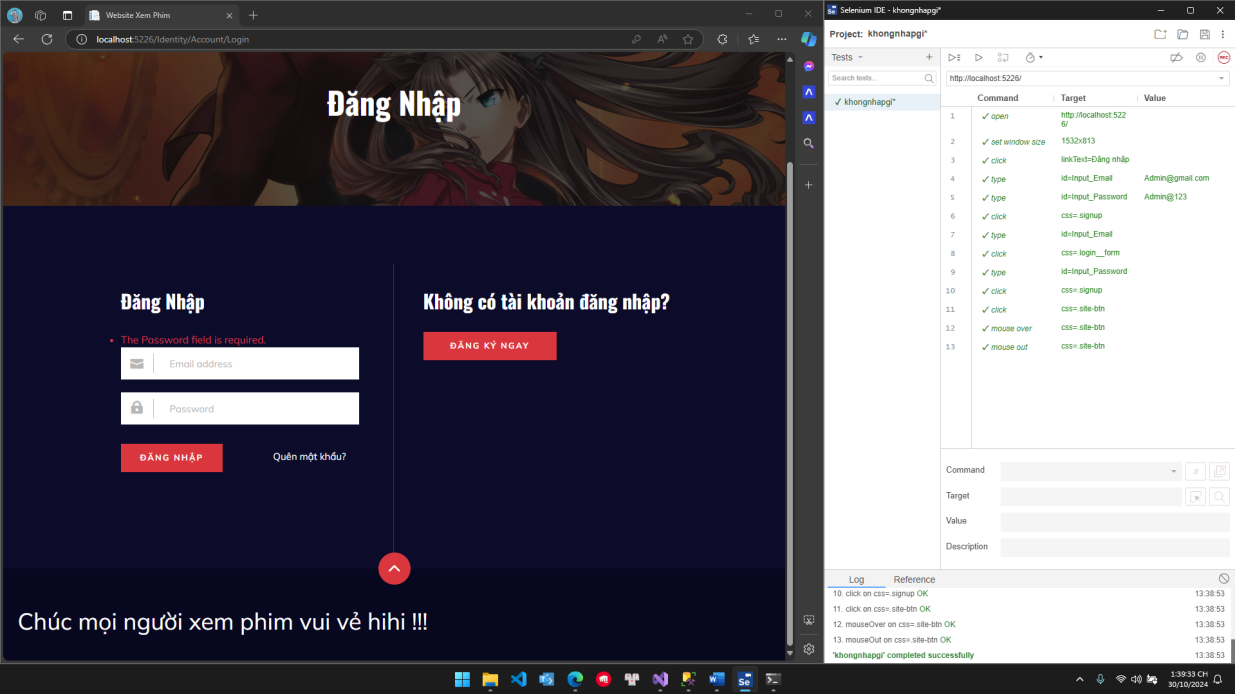
*4.1.3 Triển khai kiểm thử*

*Chức năng đăng nhập:*

Test Case đăng nhập thành công

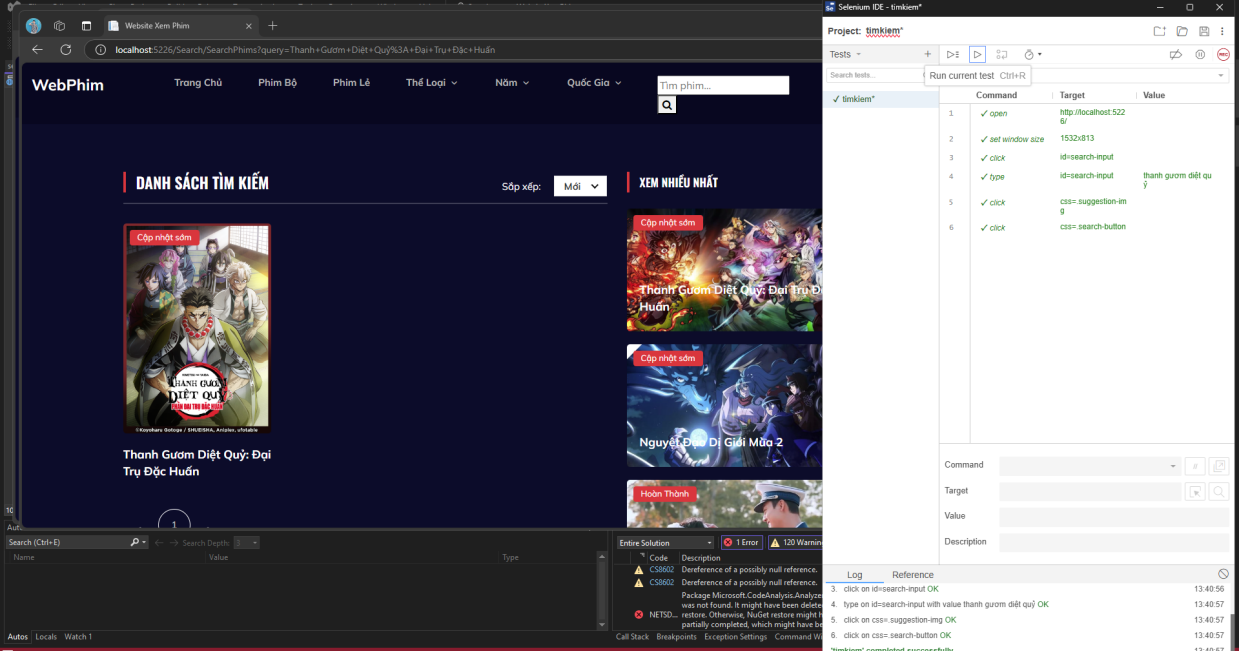


Test Case đăng nhập thất bại

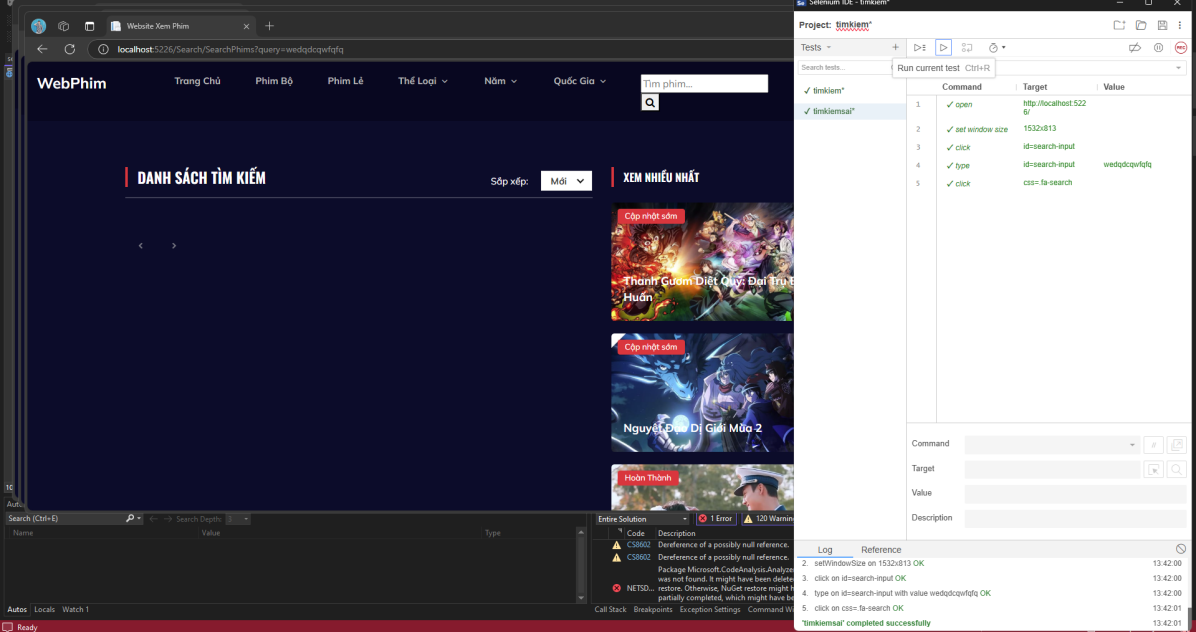


Không nhập thông tin

*Chức năng Tìm kiếm phim:*



Tìm kiếm đúng



Tìm kiếm sai

## 4.2 Công cụ kiểm thử tự động TestComplete

Kiểm tra chức năng Đăng ký

### *4.2.1 Giới thiệu*

TestComplete là một công cụ kiểm thử tự động hóa do SmartBear phát triển, cho phép kiểm thử ứng dụng trên nhiều nền tảng như web, mobile và desktop. Công cụ này hỗ trợ kiểm thử chức năng, kiểm thử hồi quy và kiểm thử giao diện người dùng. Với khả năng tương thích cao và các tính năng tiên tiến, TestComplete giúp các nhóm phát triển tăng tốc quá trình kiểm thử và cải thiện chất lượng sản phẩm.

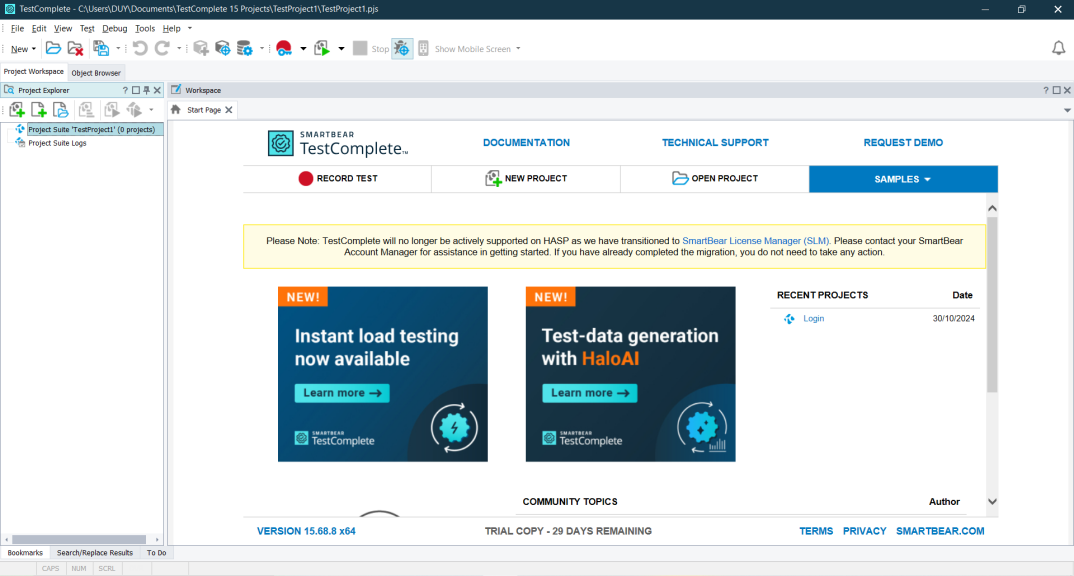
### *4.2.2 Các tính năng chính*

* **Hỗ trợ nhiều ngôn ngữ lập trình**: TestComplete hỗ trợ nhiều ngôn ngữ lập trình như JavaScript, Python, VBScript, JScript, DelphiScript, C++Script và C#Script, giúp người dùng linh hoạt trong việc lựa chọn ngôn ngữ phù hợp.
* **Ghi lại và phát lại**: Với tính năng ghi lại và phát lại, TestComplete cho phép người dùng tạo và tái sử dụng các kịch bản kiểm thử mà không cần kiến thức lập trình chuyên sâu.
* **Kiểm thử trên nhiều nền tảng:** TestComplete hỗ trợ kiểm thử các ứng dụng trên nhiều nền tảng, bao gồm Windows, web và mobile, giúp kiểm thử xuyên suốt và đồng nhất trên các thiết bị.
* **Tích hợp dễ dàng:** Công cụ này tích hợp với các công cụ CI/CD phổ biến như Jenkins, Git, và Azure DevOps, hỗ trợ cho quá trình kiểm thử tự động hóa liên tục.
* **Kiểm thử giao diện người dùng (UI Testing)**: TestComplete có thể kiểm thử UI tự động, cho phép kiểm tra các yếu tố giao diện người dùng, chức năng và phản hồi của ứng dụng.
* **Báo cáo chi tiết và phân tích**: TestComplete cung cấp báo cáo chi tiết và các công cụ phân tích kết quả kiểm thử, giúp người dùng xác định lỗi nhanh chóng và cải thiện hiệu suất kiểm thử.
* **Kiểm thử dựa trên dữ liệu:** Cho phép thực hiện kiểm thử với nhiều tập dữ liệu khác nhau, giúp kiểm tra các trường hợp đầu vào khác nhau của ứng dụng mà không cần tạo các trường hợp kiểm thử thủ công.
* **Kiểm thử dễ duy trì:** TestComplete cung cấp các công cụ quản lý và chỉnh sửa tập lệnh kiểm thử một cách dễ dàng, giúp người dùng cập nhật và bảo trì các kịch bản kiểm thử trong suốt vòng đời phát triển sản phẩm.

### *4.2.3 Tạo project*

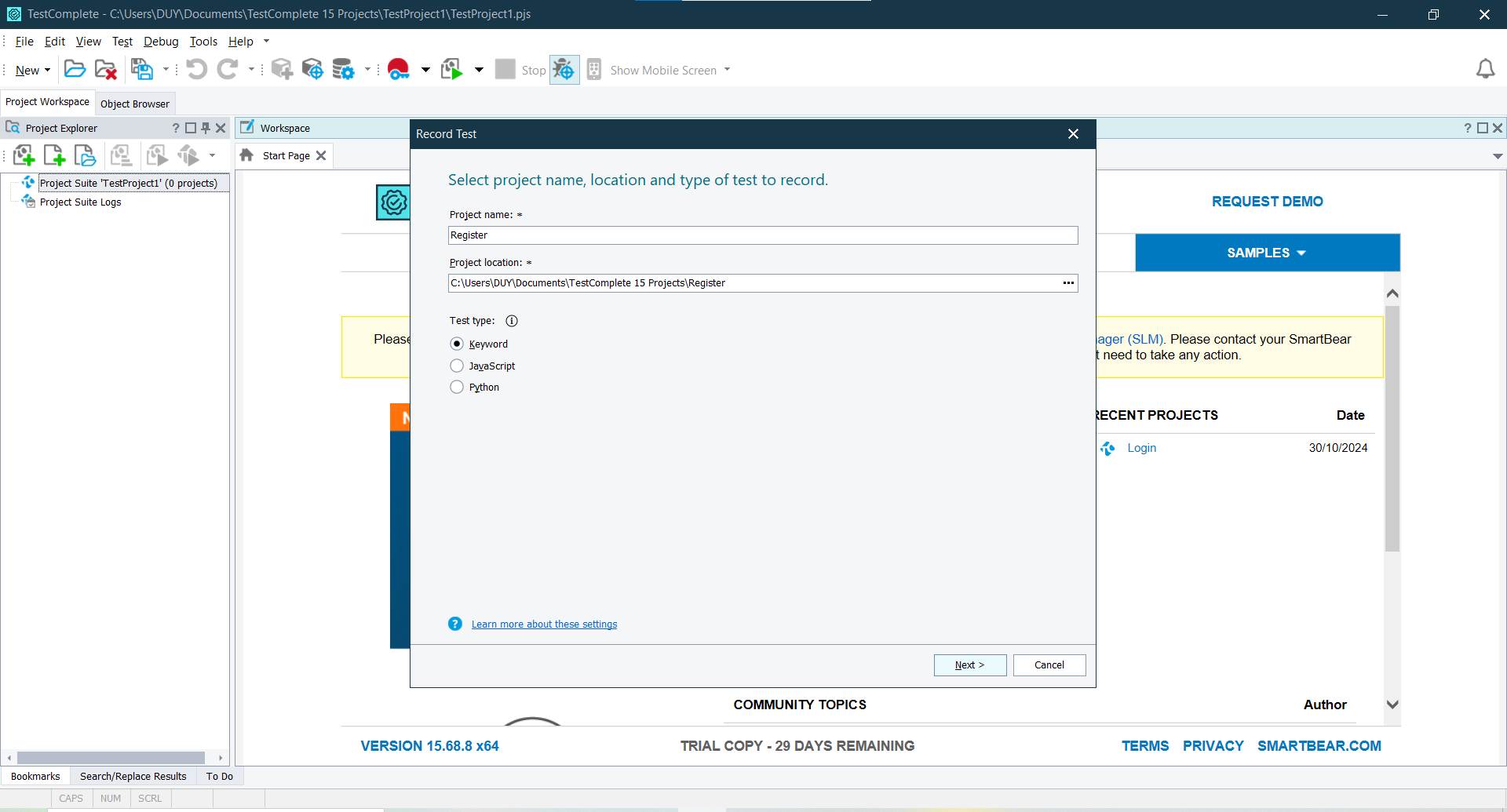
Để tạo một project trong TestComplete, bạn làm theo các bước như trong hình:

1. Mở TestComplete và vào giao diện chính.

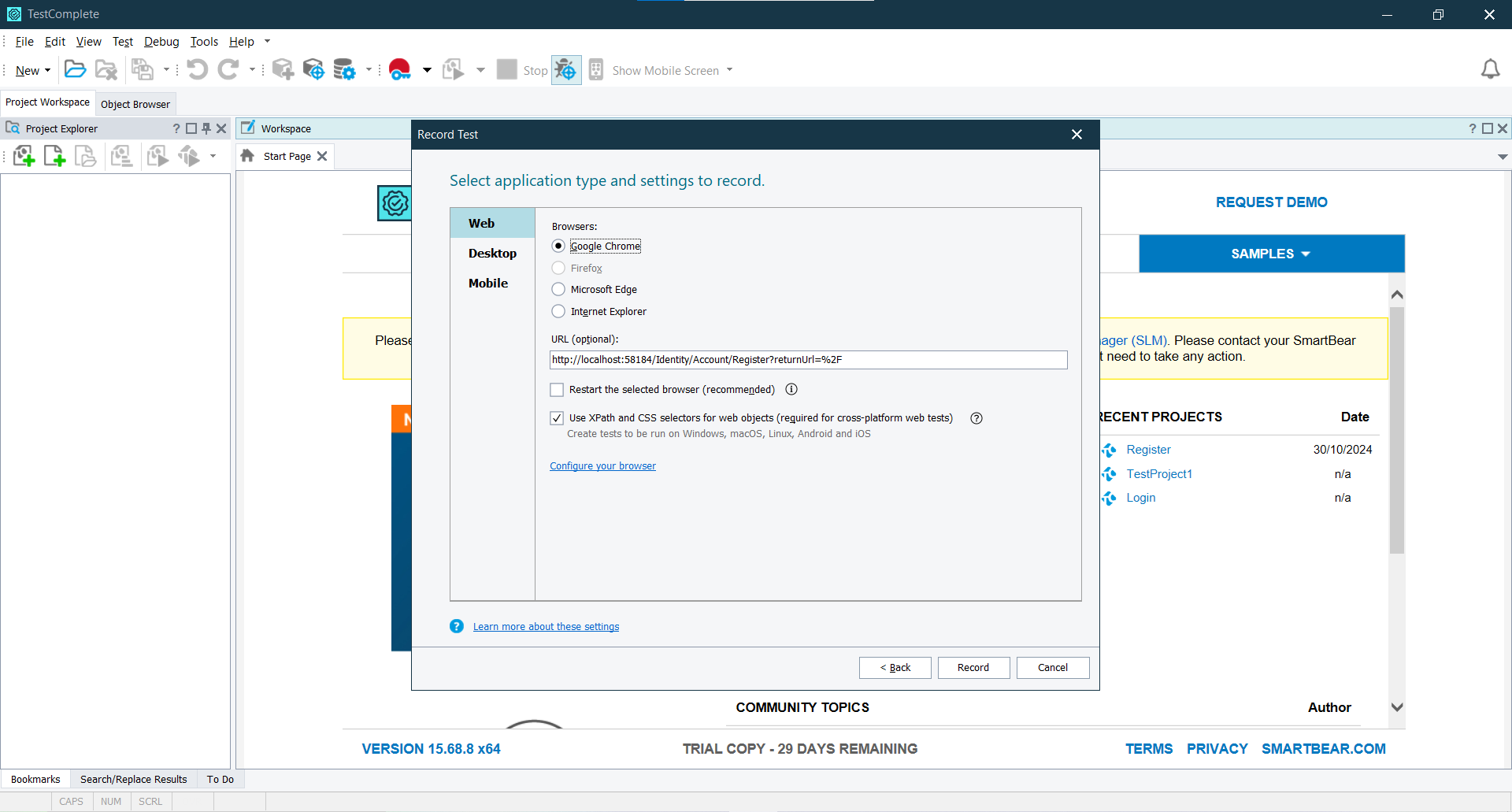


Giao diện ứng dụng

1. Chọn "New Project" ở trên để bắt đầu tạo một project mới.
2. Trong hộp thoại "Record Test", điền các thông tin sau:
   * Tên Project: Nhập tên mong muốn cho project.
   * Vị trí lưu Project: Chọn đường dẫn thư mục để lưu các tệp project.
   * Ngôn ngữ Script: Chọn ngôn ngữ lập trình bạn muốn sử dụng cho tự động hóa kiểm thử (các tùy chọn gồm JavaScript, Python, JScript, và VBScript).



1. Chọn **"Web"** làm loại kiểm thử, phù hợp nếu bạn đang làm việc với một ứng dụng web. Khi chọn loại **kiểm thử Web**, hệ thống sẽ yêu cầu bạn nhập **tên miền** của trang web cần kiểm thử. Nhập URL của trang web mà bạn muốn thực hiện kiểm thử tự động.



1. Sau khi nhập đầy đủ thông tin, nhấn "Next" để tiếp tục các cấu hình khác hoặc "Finish" nếu không cần thiết lập thêm.

TestComplete sẽ tạo project ở vị trí đã chỉ định, và bạn có thể bắt đầu ghi lại hoặc viết script cho các bài kiểm thử tự động từ đây.

### *4.2.4 Test case*

L   : Trang đăng kí

H  : Trang Home

M1: Vui lòng nhập username

M2: Vui lòng nhập password

M3: Username không hợp lệ

M4: Password không hợp lệ

M5: Vui lòng nhập mật khẩu xác nhận

M6: Mật khẩu xác nhận không khớp

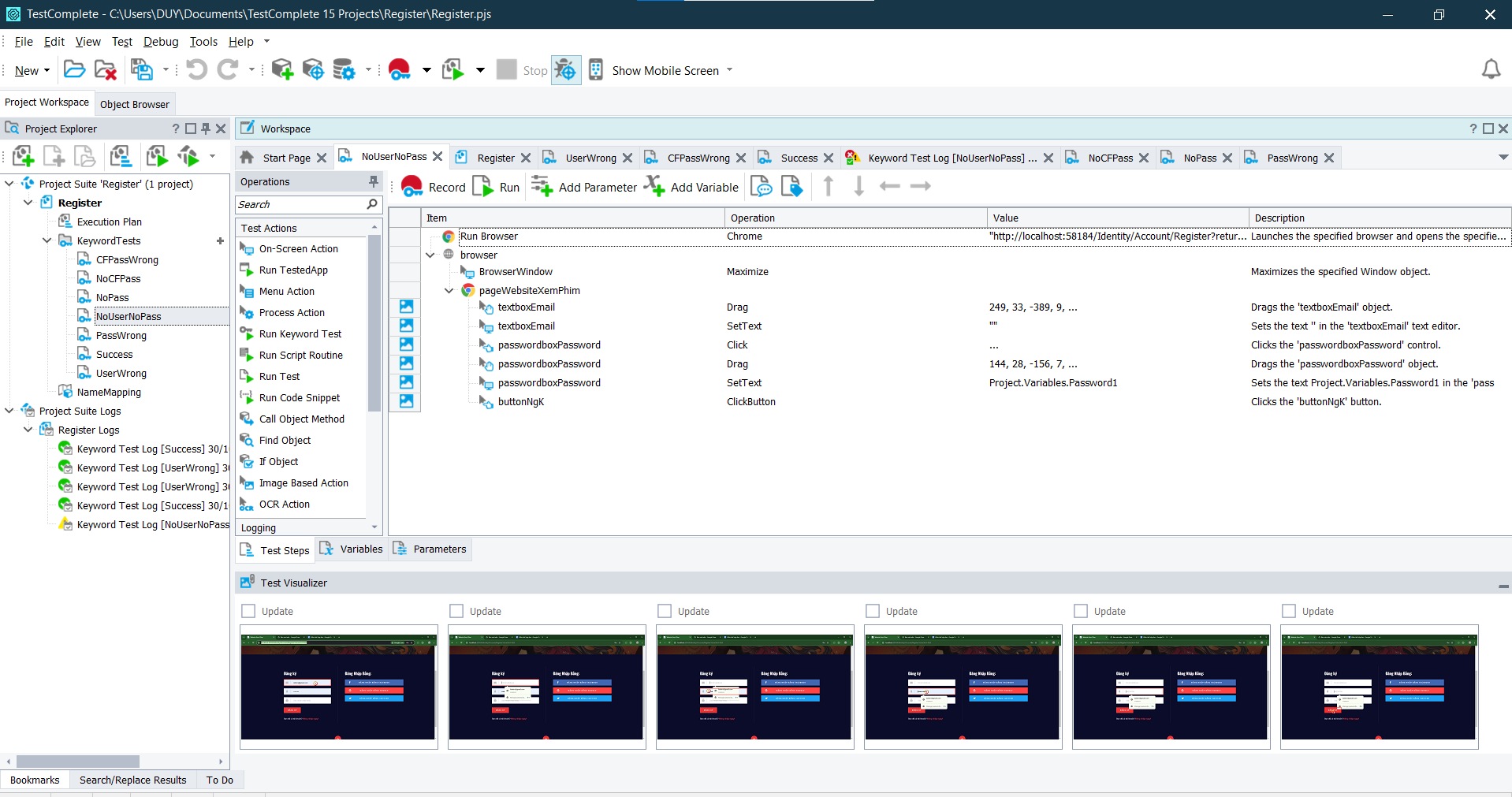
I : Nhập sai

V: Nhập đúng

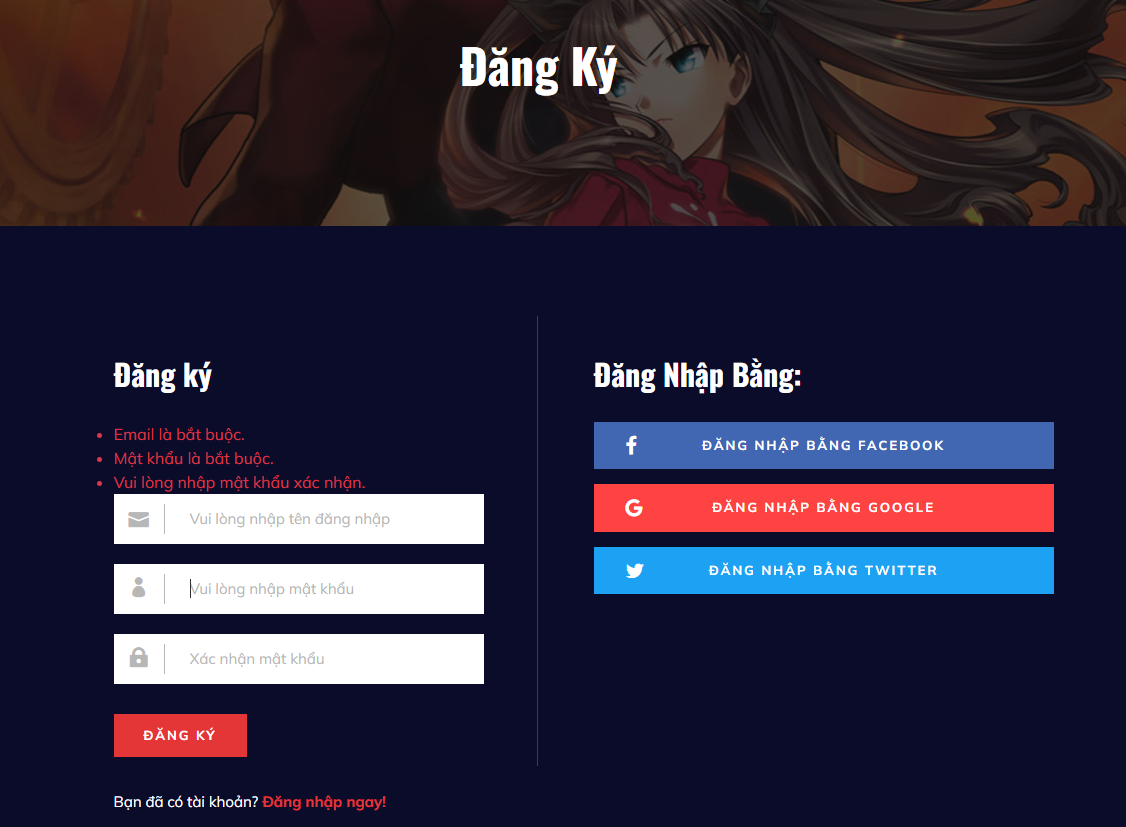
B: Bỏ trống

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | C1 | C2 | C3 | C4 | C5 | C6 | C7 |
| Điều kiện | Username  (email) |  | I | V | V | V | V | V |
| Password | - | - | B | I | V | V | V |
| XÁC NHẬN Password | - | - | - | - | B | I | V |
| Hành động | Thông báo lỗi | M1 | M3 | M2 | M4 | M5 | M6 |  |
| Chuyển trang | L | L | L | L | L | L | H |

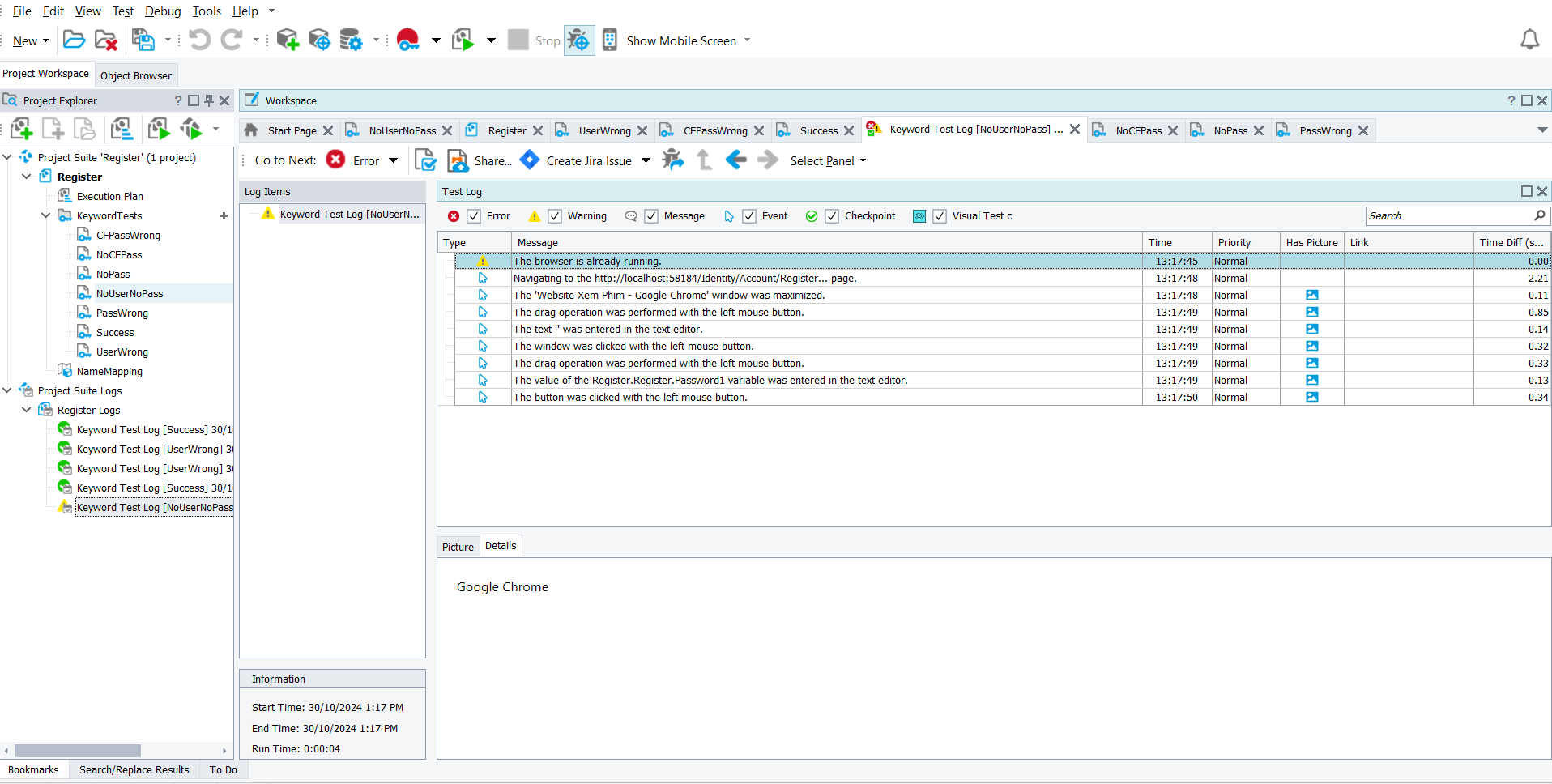
### *4.2.5 Demo*



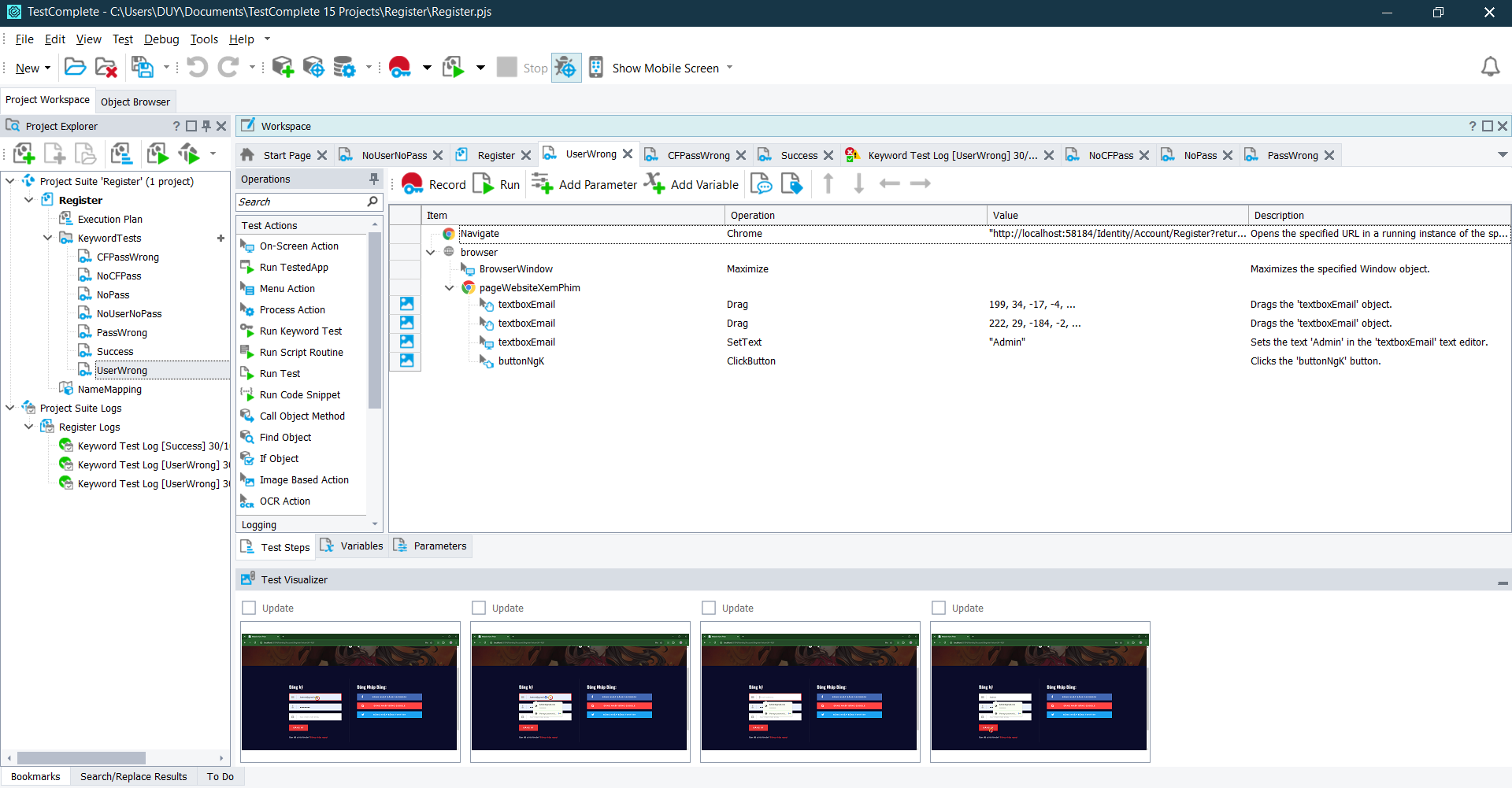
Các bước thực hiện kiểm thử tự động sau khi ghi lại testcase NoUserNoPass



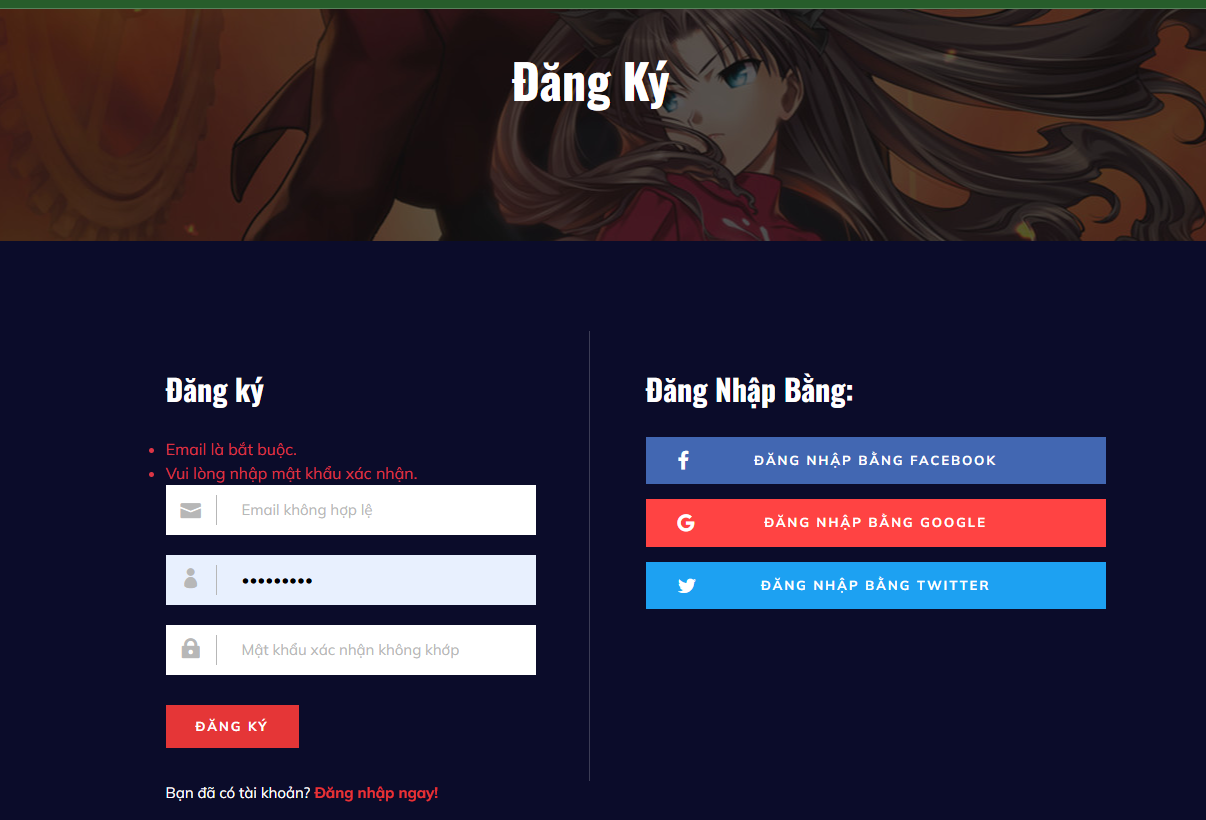
Kết quả đúng với test case NoUserNoPass



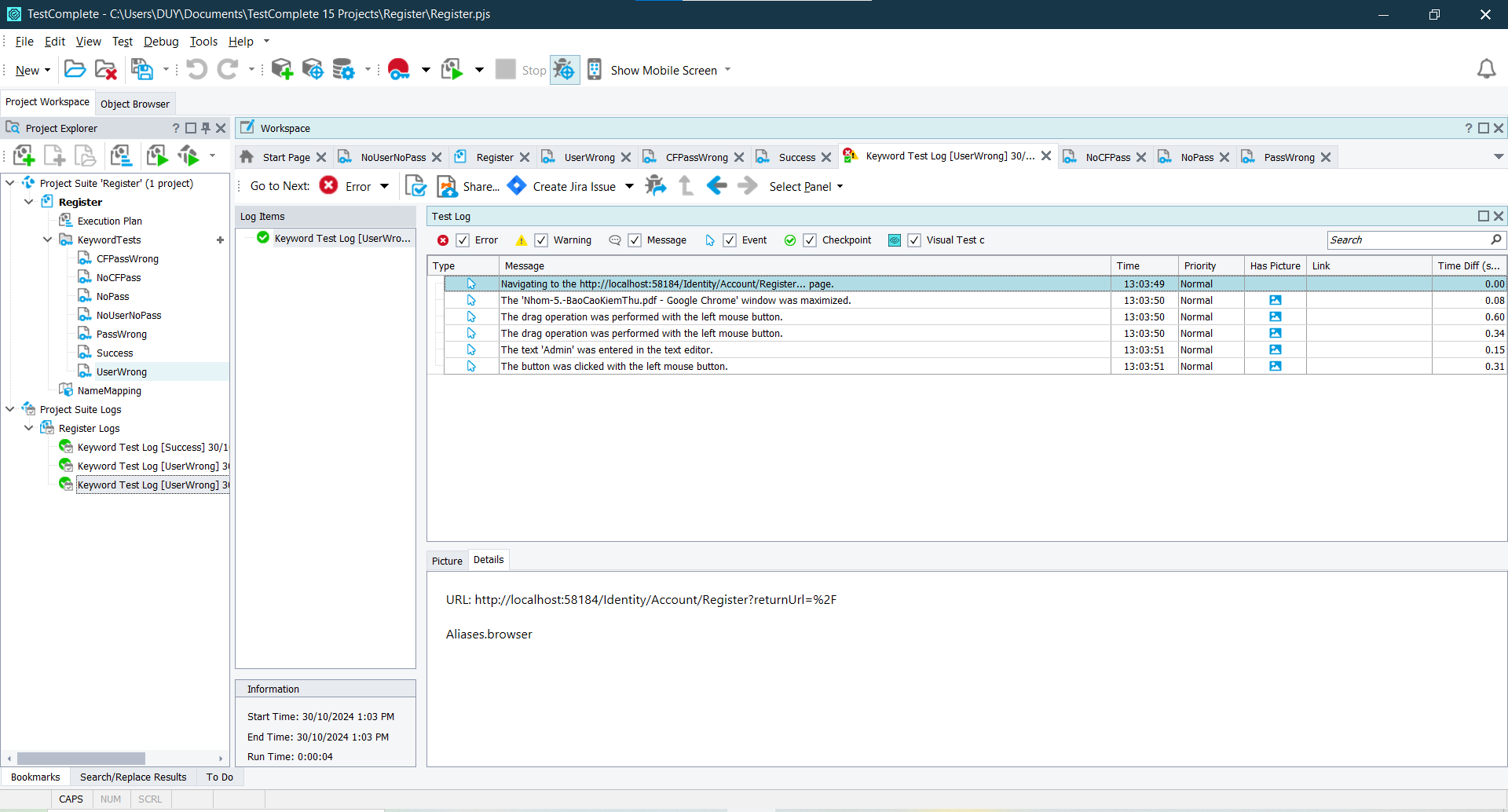
Kết quả testcase NoUserNoPass



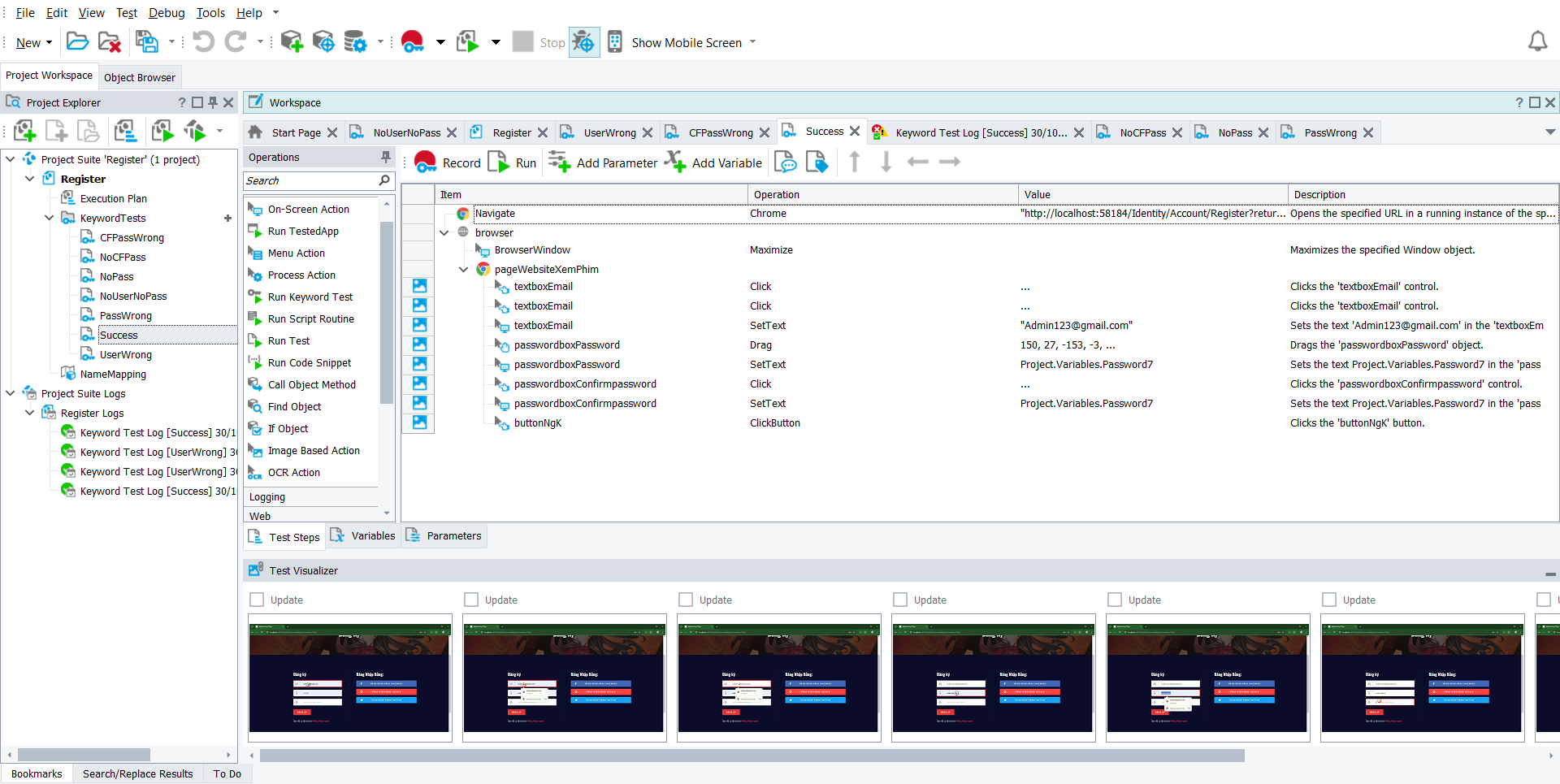
Các bước thực hiện kiểm thử tự động sau khi ghi lại testcase UserWrong



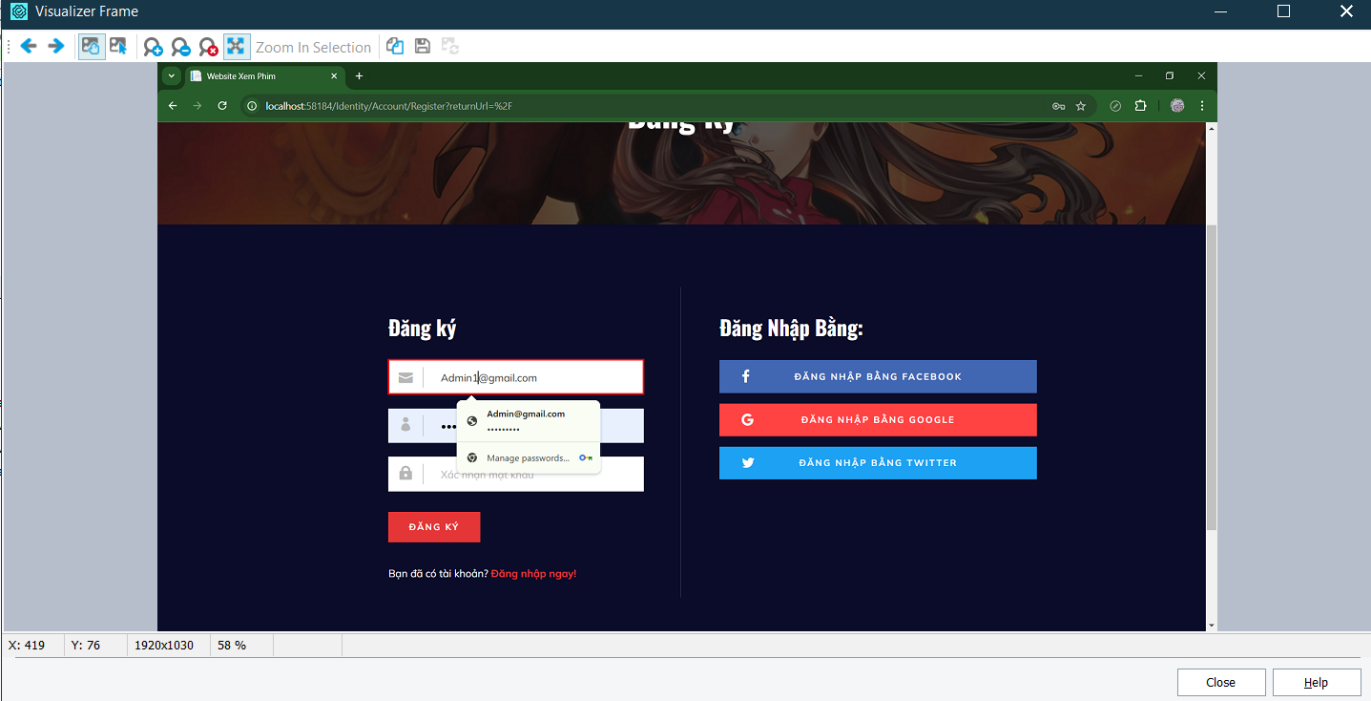
Kết quả đúng với mong muốn testcase UserWrong



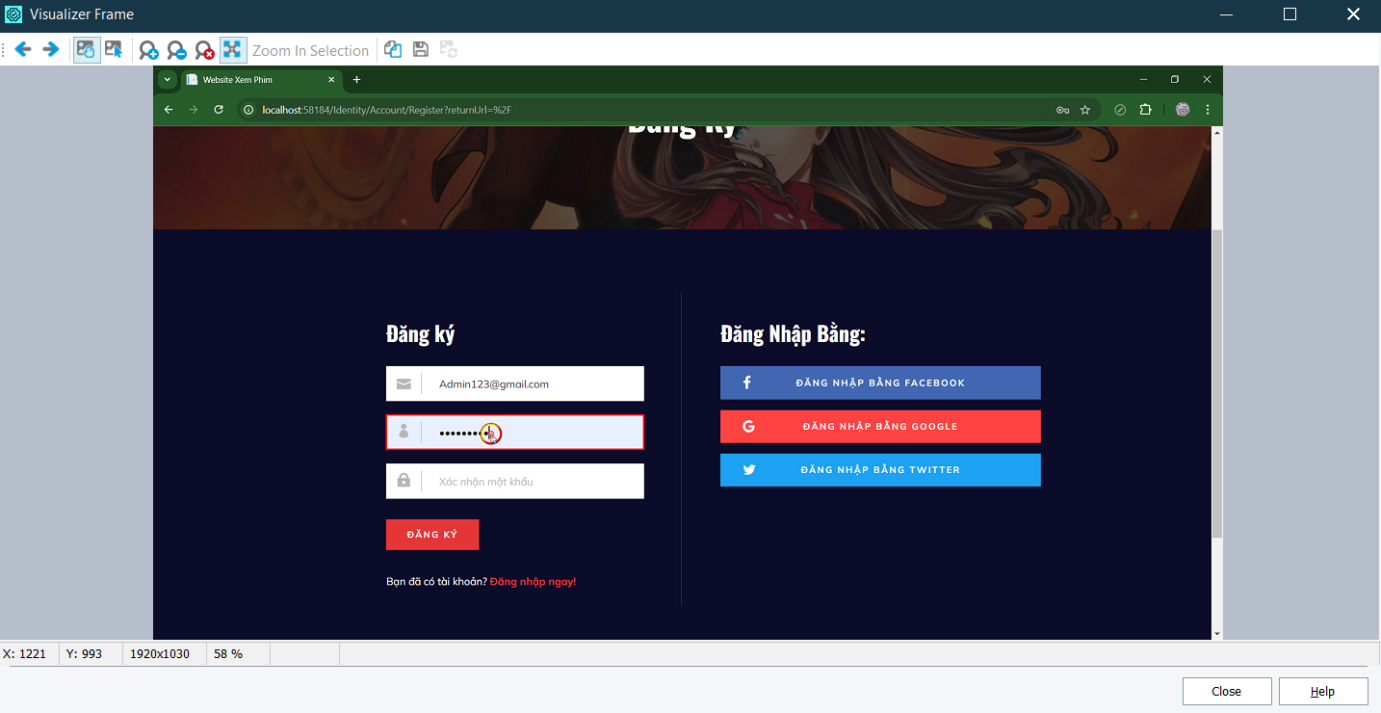
Kết quả testcase UserWrong



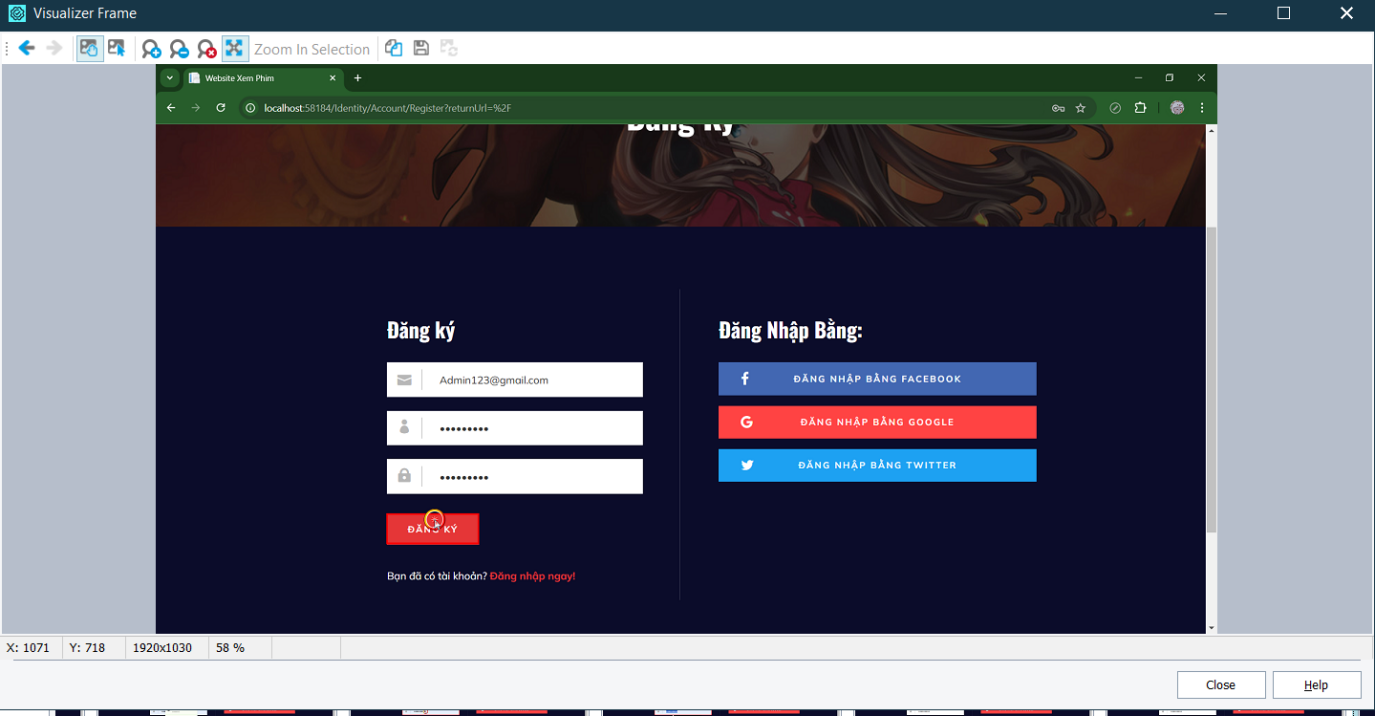
Các bước thực hiện kiểm thử tự động sau khi ghi lại testcase Success



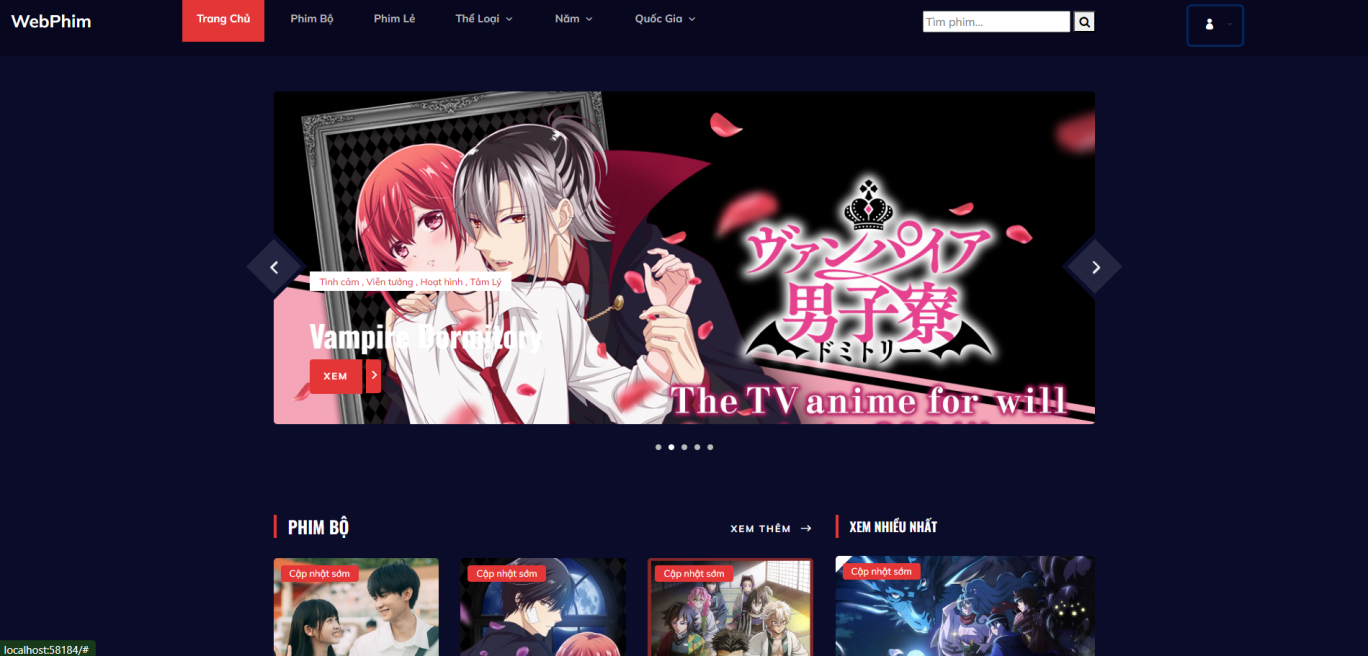
Bước 1: Nhập user của testcase Success



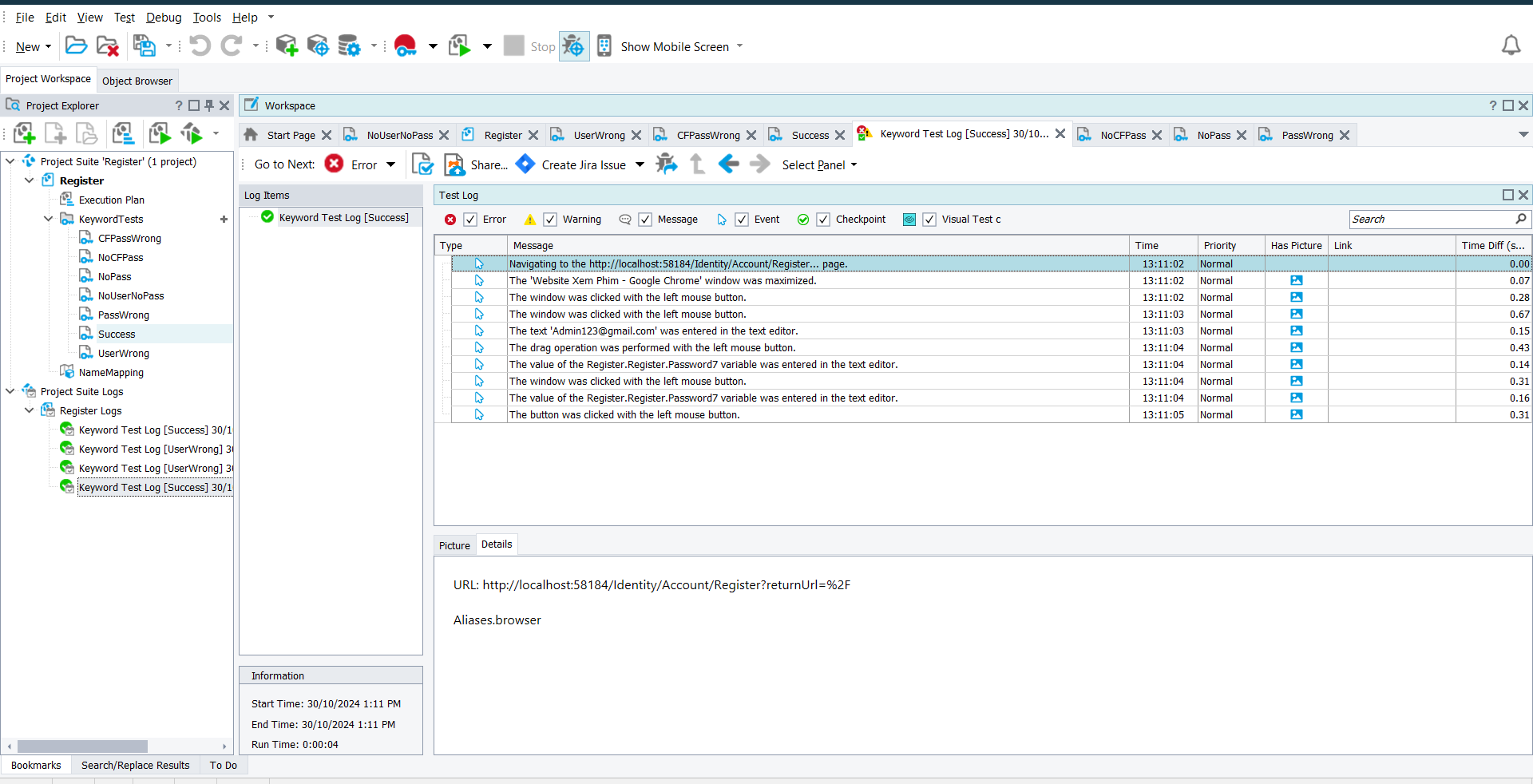
Bước 2: Nhập pass của testcase Success



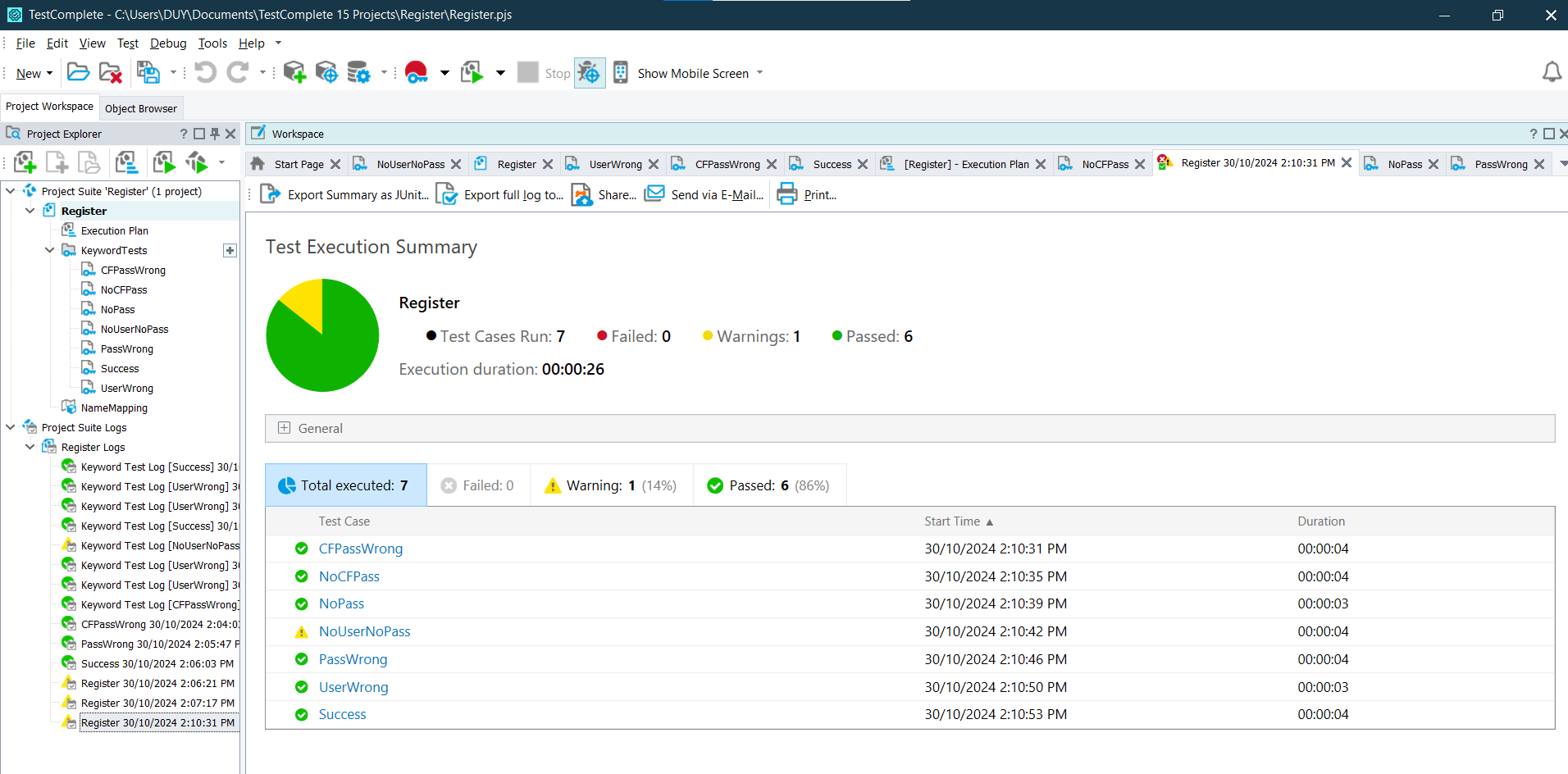
Bước 3: Nhập mật khẩu xác nhận và bấm đăng ký của testcase Success



Kết quả đúng với test case Success



Kết quả test case Success



Biểu đồ kết quả khi kiểm thử 7 testcase đăng ký

## 4.3 Công cụ kiểm thử tự động OWASP ZAP

**Kiểm thử bảo mật (Security Testing)** là quá trình đánh giá các khía cạnh bảo mật của hệ thống phần mềm hoặc ứng dụng để xác định và xử lý các điểm yếu có thể bị khai thác, nhằm bảo vệ hệ thống khỏi các cuộc tấn công từ bên ngoài. Đây là một bước quan trọng trong phát triển phần mềm, giúp đảm bảo rằng sản phẩm không dễ bị tổn thương và tuân thủ các tiêu chuẩn an toàn.

Các loại kiểm thử bảo mật phổ biến

* Kiểm thử thâm nhập (Penetration Testing)
* Quét lỗ hổng (Vulnerability Scanning)
* Kiểm thử bảo mật ứng dụng (Application Security Testing)
* Kiểm thử bảo mật mạng (Network Security Testing)
* Kiểm thử bảo mật API
* Kiểm thử cấu hình (Configuration Testing)
* **Kiểm thử ứng dụng web (Web application Testing)**: kiểm tra tính bảo mật của các ứng dụng web, bao gồm kiểm tra các lỗ hổng bảo mật phổ biến như Croos-Site Scripting (XSS), SQL Injection, Cross Site Request Forgery (CSRF), và kiểm tra tính bảo mật của phương thức xử lý dữ liệu và quyền truy cập.

### *4.3.1 Giới thiệu*

**OWASP ZAP (Zed Attack Proxy)** là một công cụ mã nguồn mở do OWASP (Open Web Application Security Project) phát triển, dùng để kiểm thử bảo mật cho các ứng dụng web. ZAP đặc biệt nổi bật bởi khả năng quét lỗ hổng và cung cấp môi trường kiểm thử bảo mật thân thiện cho các nhà phát triển cũng như chuyên gia bảo mật.

Công cụ chủ yếu được thiết kế để kiểm thử bảo mật cho các ứng dụng web và các API. ZAP hỗ trợ nhiều loại ứng dụng khác nhau, từ ứng dụng truyền thống (web) đến các hệ thống sử dụng giao diện API hiện đại, và có thể sử dụng để kiểm thử nhiều loại cấu trúc và công nghệ web khác nhau như HTML, JavaScript, JSON, XML, REST API, SOAP API, vv.

ZAP là công cụ miễn phí và được hỗ trợ đa nền tảng (Windows, macOS, Linux, Docker, API REST).

### *4.3.2 Các tính năng chính*

* Quét lỗ hổng tự động: có thể tự động quét và phát hiện các lỗ hổng bảo mật phổ biến, chẳng hạn như SQL Injection, Cross-Site Scripting (XSS), Remote File Inclusion (RFI), vv.
* Proxy và Interceptor: hoạt động như một proxy trung gian, cho phép chặn và chỉnh sửa các yêu cầu và phản hồi HTTP/HTTPS, điều này giúp kiểm thử các điểm yếu bảo mật và nghiên cứu các hành vi của ứng dụng.
* Kiểm thử thâm nhập (Penetration Testing): ZAP hỗ trợ kiểm thử thâm nhập bán tự động và thủ công, cho phép kiểm tra tính bảo mật ở nhiều mức độ khác nhau.
* Spidering: giúp tự động khám phá và lập bản đồ toàn bộ các liên kết và tài nguyên trong ứng dụng web để tìm các điểm tấn công.
* Active Scan và Passive Scan: hỗ trợ quét chủ động để kiểm tra các lỗ hổng có thể khai thác và quét thụ động nhằm phát hiện các điểm yếu bảo mật mà không gây ảnh hưởng đến ứng dụng.

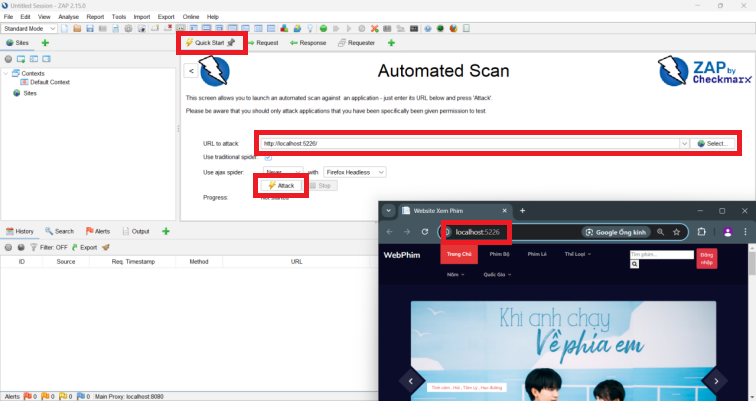
*4.3.3 Triển khai kiểm thử*

**Cài đặt OWASP ZAP:**

* Vào <https://www.zaproxy.org/> và tải xuống phiên bản phù hợp với hệ điều hành sau đó cài đặt.
* ZAP chỉ hỗ trợ JDK 11 nhỏ nhất, nên cài đặt các phiên bản mới nhất (<https://www.oracle.com/>).

**Khởi động OWASP SAP:**

Chạy chương trình web xem phim và khởi động OWASP ZAP



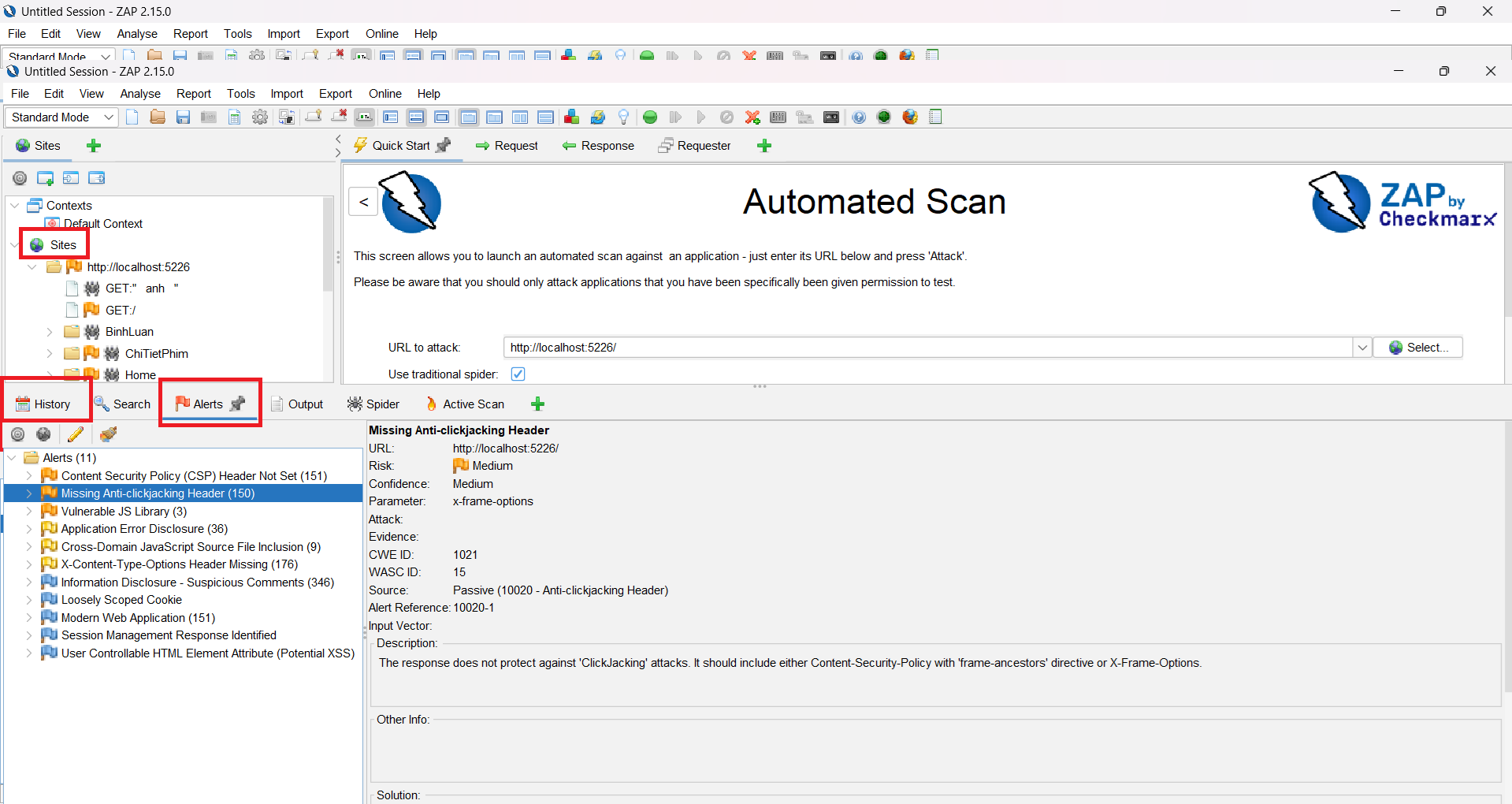
Giao diện người dùng chính của OWASP ZAP

**Xác định và quét mục tiêu:**

* Trong OWASP ZAP, điều hướng đến tab "Quick Start".
* Trong phần "URL to attack", nhập URL của ứng dụng web kiểm thử (http://localhost:5226/).
* Nhấn nút "Attack" để bắt đầu quét ứng dụng web.

**Quét và kiểm tra bảo mật:**

OWASP ZAP sẽ bắt đầu quét ứng dụng web và tìm kiếm các lỗ hổng bảo mật phổ biến. Có thể theo dõi tiến trình quét trong tab "Alerts", nơi hiển thị các cảnh báo và lỗ hổng bảo mật được tìm thấy



Kết quả trả về sau khi ZAP quét tự động

**Xem và xử lý kết quả:**

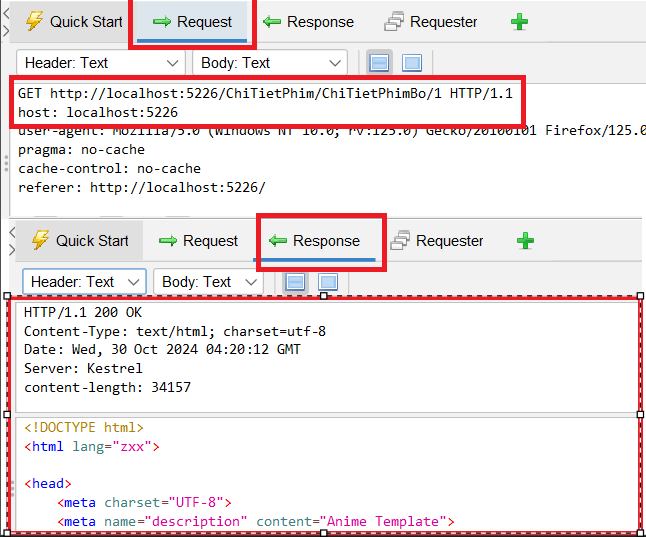
Sau khi quét hoàn tất, bạn có thể xem kết quả quét trong các tab "Sites", "History" và trong tab "Alerts" - sẽ thấy danh sách các cảnh báo và lỗ hổng bảo mật đã được tìm thấy, có thể nhấp vào từng mục để xem chi tiết và thông tin liên quan.

***Alerts (Cảnh báo)***

* Mức độ nghiêm trọng (Severity): Đánh giá mức độ nguy hiểm của lỗ hổng bảo mật, thông thường được phân loại thành các mức như Critical, High, Medium, Low.
* Loại lỗ hổng (Vulnerability type): Mô tả loại lỗ hổng bảo mật đã được phát hiện, ví dụ: Cross-Site Scripting (XSS), SQL Injection, Insecure Direct Object References, và nhiều loại khác.
* URL liên quan: Liên kết đến trang web hoặc tài nguyên có chứa lỗ hổng bảo mật.
* Mô tả: Chi tiết về lỗ hổng bảo mật đã được phát hiện, bao gồm cách tấn công có thể được thực hiện và cách khắc phục.

Màu sắc của cảnh báo cho biết mức độ nghiêm trọng của các vấn đề bảo mật:

* + **Màu đỏ**: Cảnh báo nghiêm trọng (High severity). Đây là những vấn đề bảo mật có nguy cơ cao, có thể dễ dàng bị khai thác và gây ra tổn thất lớn. Nên ưu tiên xử lý các cảnh báo màu đỏ.
  + **Màu cam**: Cảnh báo trung bình (Medium severity). Đây là những vấn đề bảo mật có nguy cơ trung bình. Mặc dù không nguy hiểm bằng các cảnh báo nghiêm trọng, cũng nên xem xét giải quyết chúng để tăng cường tính bảo mật của ứng dụng.
  + **Màu vàng**: Cảnh báo nhẹ (Low severity). Đây là những vấn đề có nguy cơ thấp và ít có khả năng bị khai thác. Tuy nhiên, nếu có thời gian, việc khắc phục chúng cũng giúp cải thiện bảo mật tổng thể.
  + **Màu xanh lam**: Cảnh báo thông tin (Informational). Đây không phải là vấn đề bảo mật mà chỉ là các thông tin bổ sung về ứng dụng, chẳng hạn như nhận diện phiên bản phần mềm hoặc các thông tin cần lưu ý. Bạn có thể bỏ qua các cảnh báo này nếu chúng không ảnh hưởng trực tiếp đến bảo mật.



* + **Request (Yêu cầu)**: OWASP ZAP tạo ra các request (phương thức HTTP: GET/POST/PUT/DELETE, URL, Header, Body) để mô phỏng các hành động người dùng hoặc kiểm tra các điểm yếu trong ứng dụng, như XSS, SQL Injection, CSRF, và nhiều lỗ hổng khác.
  + **Response (Phản hồi)** là phản hồi HTTP từ máy chủ web sau khi nhận được yêu cầu. Response chứa các thông tin Status Code, Header, Body.

***Sites (Trang web)***

* URL: Địa chỉ trang web đã được quét.
* Số lần quét: Số lần trang web đã được quét trong quá trình kiểm tra bảo mật.
* Số lỗ hổng: Số lỗ hổng bảo mật đã được phát hiện trong trang web.
* Trạng thái: Trạng thái hiện tại của trang web, ví dụ: hoạt động, không hoạt động, bị chặn.

***History (Lịch sử)***

* Thời gian quét: Thời điểm và thời gian mà quá trình quét đã diễn ra.
* Số lỗ hổng: Tổng số lỗ hổng bảo mật đã được phát hiện trong quá trình quét.
* Thông tin liên quan: Các thông tin khác liên quan đến quá trình quét, ví dụ: thiết bị hoặc công cụ sử dụng.

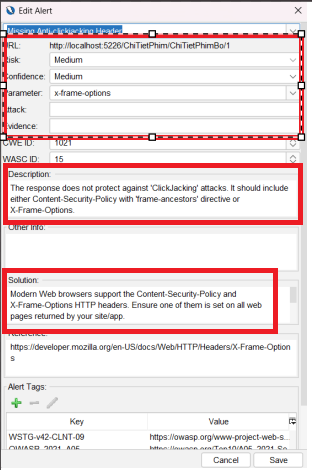
***Statistics (Thống kê)***

* Số lượng trang web: Tổng số trang web đã được quét.
* Số lỗ hổng: Tổng số lỗ hổng bảo mật đã được phát hiện trong tất cả các trang web.
* Mức độ nghiêm trọng: Thống kê về số lượng lỗ hổng theo mức độ nghiêm trọng.
* Các thông tin thống kê khác: Có thể bao gồm tỷ lệ lỗ hổng theo loại, biểu đồ thống kê, và thông tin liên quan khác.

**Phân tích kết quả:**

Xem chi tiết từng Cảnh báo và phân tích kết quả, danh sách cảnh báo đã quét được:

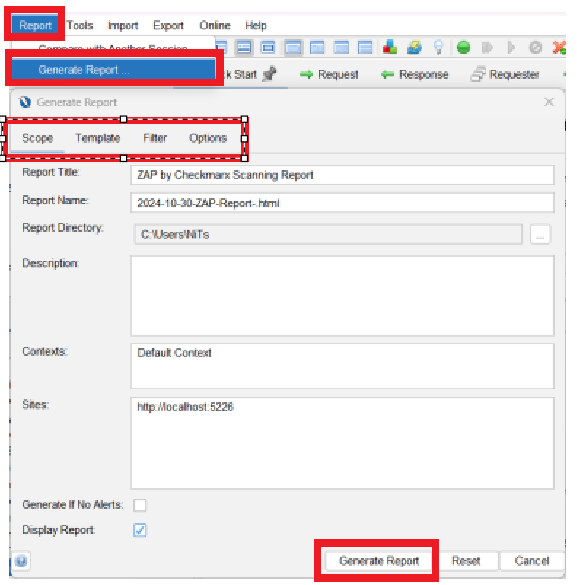
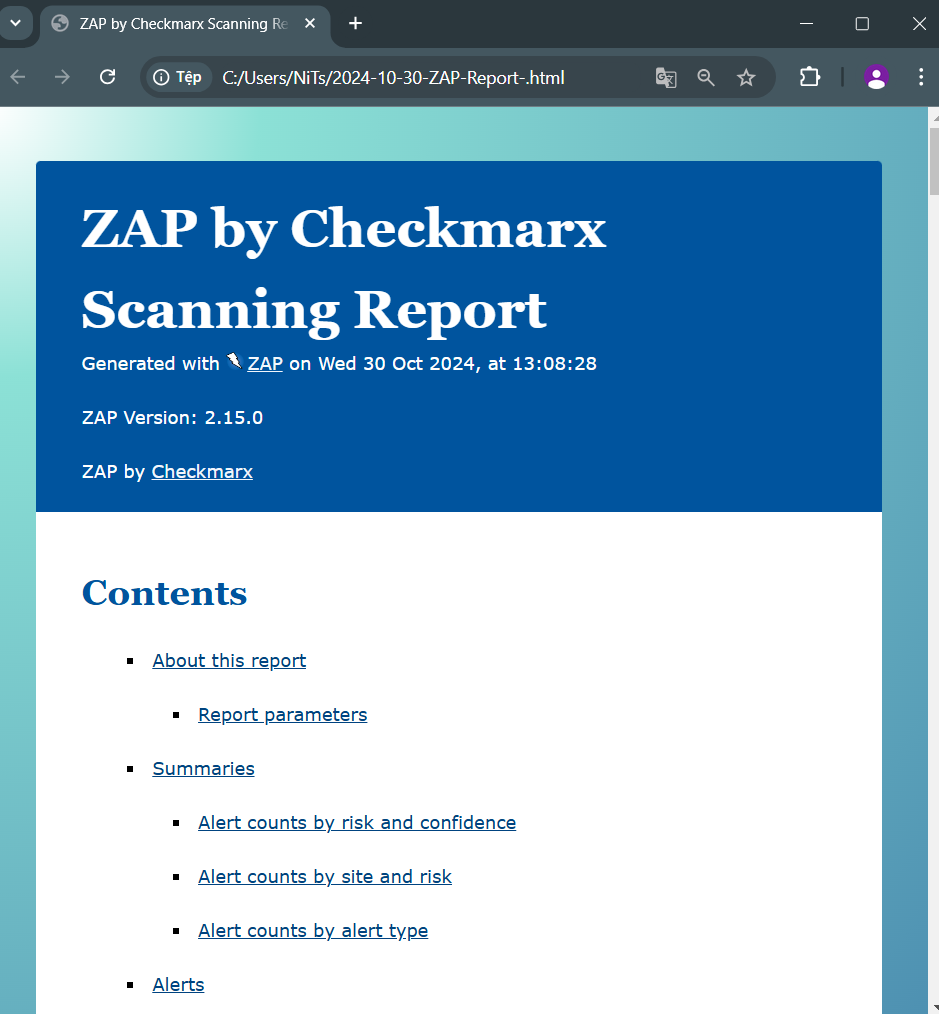
* **Content Security Policy (CSP) Header Not Set**: CSP là một cơ chế bảo mật giúp ngăn ngừa các cuộc tấn công XSS bằng cách hạn chế các nguồn nội dung có thể được tải vào trang web. Cảnh báo này cho biết không có CSP header nào được thiết lập, điều này có thể khiến trang web dễ bị tấn công XSS.
* **Missing Anti-clickjacking Header**: Cảnh báo này liên quan đến thiếu header X-Frame-Options. Header này ngăn chặn tấn công clickjacking bằng cách không cho phép nội dung của trang web được nhúng vào iframe trên các trang web khác.
* **Vulnerable JS Library**: Cảnh báo này cho biết trang web của bạn đang sử dụng một hoặc nhiều thư viện JavaScript có lỗ hổng bảo mật đã biết, nên cập nhật lên phiên bản mới nhất của các thư viện này.
* **Application Error Disclosure**: Cảnh báo này xuất hiện khi ứng dụng tiết lộ thông tin lỗi có thể hữu ích cho kẻ tấn công. Ví dụ: lỗi chi tiết về mã nguồn hoặc lỗi kết nối cơ sở dữ liệu có thể được tiết lộ cho người dùng.
* **Cross-Domain JavaScript Source File Inclusion**: Cảnh báo này liên quan đến việc bao gồm các tệp JavaScript từ các tên miền khác. Điều này có thể dẫn đến các vấn đề bảo mật nếu các nguồn bên ngoài không được tin tưởng.
* **X-Content-Type-Options Header Missing:** Header X-Content-Type-Options (với giá trị nosniff) giúp ngăn ngừa tấn công MIME-sniffing, trong đó trình duyệt cố gắng đoán loại nội dung của tệp dựa trên nội dung thay vì dựa vào loại MIME. Thiếu header này có thể cho phép các cuộc tấn công XSS.
* **Information Disclosure - Suspicious Comments**: Cảnh báo này được tạo ra khi ZAP phát hiện các bình luận trong mã nguồn có chứa thông tin nhạy cảm hoặc đáng ngờ (chẳng hạn như ghi chú phát triển hoặc thông tin về cấu trúc mã).
* **Loosely Scoped Cookie**: Cảnh báo này xuất hiện khi cookie của trang web có phạm vi quá rộng, có thể truy cập được từ nhiều miền hoặc thư mục hơn mức cần thiết. Điều này có thể làm tăng nguy cơ tấn công CSRF và tấn công giả mạo cookie.
* **Modern Web Application**: Đây là một cảnh báo tổng quát cho biết rằng ứng dụng web của bạn là ứng dụng hiện đại với các tính năng có thể yêu cầu bảo mật bổ sung.
* **Session Management Response Identified:** Cảnh báo này cho biết có các phiên (session) quản lý đã được xác định trong ứng dụng. Quản lý phiên không an toàn có thể dẫn đến nhiều loại tấn công, chẳng hạn như chiếm đoạt phiên.
* **User Controllable HTML Element Attribute (Potential XSS**): Cảnh báo này liên quan đến các thuộc tính HTML mà người dùng có thể kiểm soát được (ví dụ như qua input form). Nếu không kiểm tra và lọc các đầu vào này, trang web có thể dễ bị tấn công XSS.



thông tin chi tiết Cảnh báo

**Tạo báo cáo**

* Có thể tạo báo cáo với tùy biến template của bản report, có thể dùng định dạng HTML để xuất file, có thể chọn định dạng khác như XML:
* Chọn **Report** (trên thanh tiêu đề) - Chọn **Generate Report** (Tuỳ chỉnh định dạng ở form) - chọn **Generate Report** để xuất file.

Form **Generate Report và File HTML báo cáo - phân tích kết quả**

**Kết quả**

**Sau khi triển khai kiểm thử, chạy quét tự động và nhận lại chi tiết các cảnh báo, phân tích kết quả thì trang web còn nhiều lỗ hỏng bảo mật c**ó thể dẫn đến các cuộc tấn công như XSS, CSRF, clickjacking, chiếm đoạt phiên, và tiết lộ thông tin nhạy cảm, gây nguy hiểm cho bảo mật và quyền riêng tư của người dùng.

Trang web có thể tấn công các lỗ hỏng khác do công cụ không thể quét được, cần có các biện pháp đúng cách để bảo vệ trang web khỏi các tấn công. Đặc biệt là tấn công SQL Injection - tấn cống vào dữ liệu để lấy thông tin.

## 4.4 Công cụ kiểm thử tự động LambdaTest

Kiểm thử giao diện

### *4.4.1 Giới thiệu*

LambdaTest là một nền tảng kiểm thử đám mây cho phép các nhà phát triển và kiểm thử viên kiểm thử ứng dụng web và di động trên nhiều trình duyệt và hệ điều hành khác nhau. Với LambdaTest, bạn có thể kiểm thử cross-browser (kiểm thử đa trình duyệt) mà không cần thiết lập hạ tầng vật lý. Đây là một công cụ tiện lợi trong việc đảm bảo ứng dụng hoạt động mượt mà trên mọi thiết bị và trình duyệt, từ đó cải thiện trải nghiệm người dùng.

### *4.4.2 Các tính năng chính*

* Kiểm thử trên nhiều trình duyệt và hệ điều hành:Hỗ trợ kiểm thử trên hơn 3000 trình duyệt và hệ điều hành khác nhau, bao gồm cả các phiên bản cũ. Kiểm thử trên các hệ điều hành phổ biến như Windows, macOS, Android và iOS.
* Kiểm thử tự động (Automation Testing): Tích hợp với Selenium, Cypress, Playwright, và Appium, cho phép bạn chạy các bài kiểm thử tự động trên nền tảng đám mây. Giúp tăng tốc độ kiểm thử nhờ khả năng chạy song song (parallel testing).
* Kiểm thử thủ công (Manual Testing): Cho phép kiểm thử thủ công trên giao diện của các thiết bị ảo (virtual devices). Hỗ trợ việc kiểm tra và gỡ lỗi giao diện ngay trên các môi trường mô phỏng.
* Kiểm thử trên thiết bị thực (Real Device Testing): Hỗ trợ kiểm thử trên các thiết bị thực như điện thoại, máy tính bảng, để đảm bảo kết quả kiểm thử chính xác hơn.
* Chụp ảnh màn hình và ghi chú lỗi: Tính năng chụp ảnh màn hình tự động và ghi chú lỗi giúp theo dõi và xử lý các vấn đề giao diện trên từng thiết bị và trình duyệt cụ thể.
* Tích hợp CI/CD: LambdaTest có thể tích hợp với các công cụ CI/CD phổ biến như Jenkins, CircleCI, và GitLab để tự động hóa kiểm thử, giúp cải thiện quy trình phát triển.
* Báo cáo kiểm thử và phân tích: Cung cấp báo cáo chi tiết về kết quả kiểm thử, giúp dễ dàng phát hiện lỗi và cải thiện hiệu suất.
* Kiểm thử Geolocation: Hỗ trợ kiểm thử từ các vị trí địa lý khác nhau, giúp đảm bảo ứng dụng hiển thị nội dung phù hợp với vị trí người dùng.

*4.4.3 Triển khai kiểm thử*

Nhấp vào nút  **'Cấu hình đường hầm'** . Tải xuống tệp zip nhị phân bằng cách nhấp vào  **'Liên kết tải xuống'** . Nhấn  nút **'Sao chép'**  trong biểu mẫu mở ra để sao chép toàn bộ chuỗi vào bảng tạm của hệ thống.

A screenshot of a computer

Description automatically generated

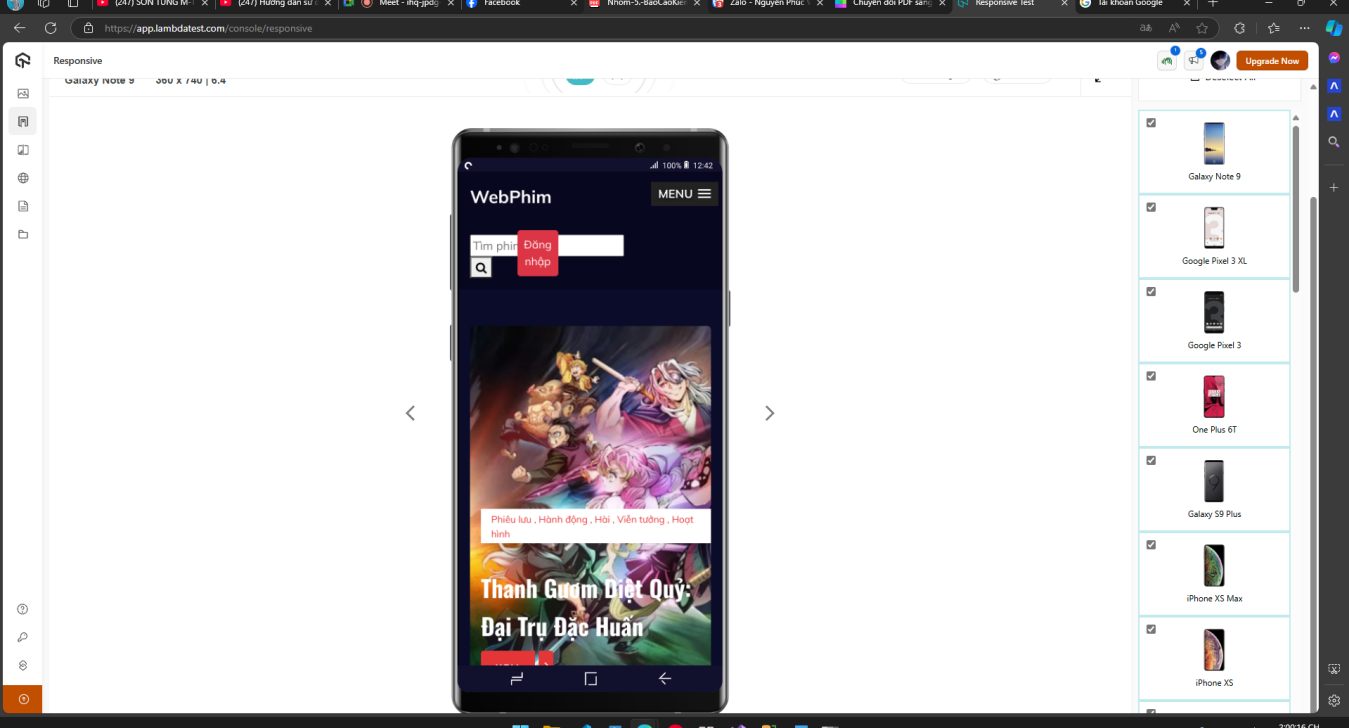
Giải nén tệp zip đã tải xuống. Vào  **'Command Prompt'**  và định tuyến con trỏ đến cùng thư mục mà bạn đã giải nén tệp zip.

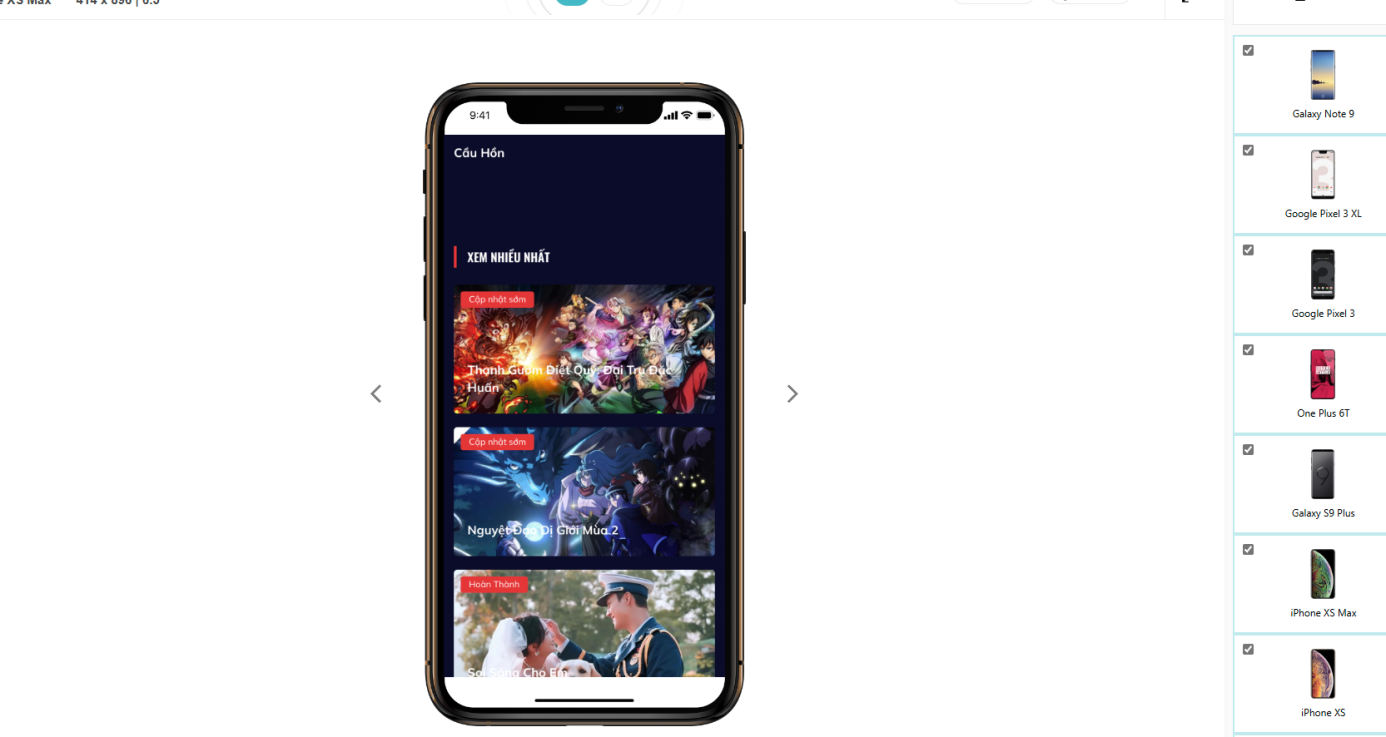
A screenshot of a computer

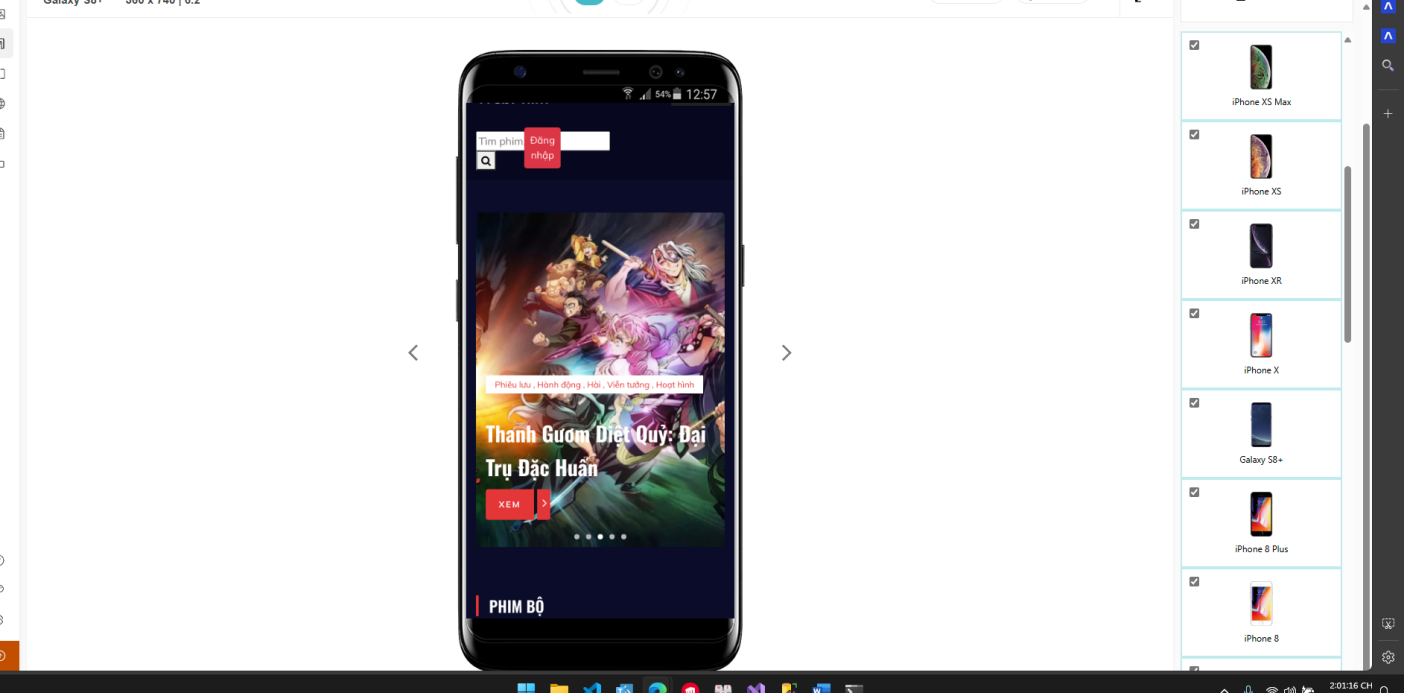
Description automatically generated

Test UI Trên Điện Thoại

Thực hiện trên web localhost







# CHƯƠNG 5: KẾT LUẬN

## 5.1 Kết luận:

### *5.1.1 Kết quả đạt được:*

- Trình bày được các kiến thức cơ bản về kiểm thử phần mềm nói chung và kiểm

thử phần mềm tự động cho các ứng dụng Web nói riêng.

- Hiểu được cách cài đặt, thiết lập và sử dụng công cụ kiểm thử tự động.

- Áp dụng các kiến thức đã tìm hiểu vào thực hiện kiểm thử tự động các chức

năng chính của Website xem phim.

### *5.1.2 Hạn chế:*

- Phần mềm chưa đạt trạng thái tốt nhất.

- Chỉ mới kiểm thử được một vài ứng dụng nhỏ.

- Chưa nghiên cứu được nhiều phương pháp kiểm thử khác.

- Về phương diện chung, phần mềm có thể đi vào hoạt động nhưng cần bổ sung thêm ràng buộc và các tính năng yêu cầu. m thử tự động để có thể vận dụng kiểm thử các ứng dụng lớn hơn trong thực tiễn.

## 5.2 Nhận xét

Kiểm thử phần mềm là một hoạt động quan trọng trong đảm bảo chất lượng phần mềm để đảm bảo độ tin cậy và chất lượng của phần mềm. Mục đích chính của kiểm thử là phát hiện ra các lỗi phần mềm để từ đó khắc phục và sửa chữa. Việc kiểm thử không thể khẳng định được rằng các chức năng của sản phẩm đúng trong mọi điều kiện, mà chỉ có thể khẳng định rằng nó không hoạt động đúng trong những điều kiện cụ thể.

Trong thời gian tìm hiểu về kiểm thử phần mềm nói chung và công cụ kiểm thử nói riêng nhóm em đã hiểu được kiểm thử phần mềm là giai đoạn rất quan trọng trong quy trình sản xuất phần mềm .Sự áp dụng mới chỉ thực hiện trên một bài toán nhỏ. Nhóm em xin trân thành cảm ơn thầy **Dương Thành Phết** đã giúp đỡ chúng em thực hiện tốt đề tài này.

# TÀI LIỆU THAM KHẢO

[3] <https://github.com/microsoft/playwright>

[4] https://playwright.dev/docs/api/class-playwright

[5] https://www.zaproxy.org/docs/

[6] https://cheatsheetseries.owasp.org

[7] [https://community.smartbear.c](https://www.youtube.com/watch?v=Dre2iVOCqhY)om

[8] [https://smartbear.com/academy](https://www.youtube.com/watch?v=Dre2iVOCqhY)/

[9] https://github.com/SeleniumHQ/selenium

[10] [https://www.selenium.dev/documentation/webdriver](https://www.youtube.com/watch?v=Dre2iVOCqhY)/

[11] <https://github.com/LambdaTest>

[12] [https://www.lambdatest.com/learning-hub](https://github.com/LambdaTest)/