

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI



ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC
NGÀNH CÔNG NGHỆ THÔNG TIN

TÊN ĐỀ TÀI: NGHIÊN CỨU CÔNG CỤ KIỂM THỬ TỰ ĐỘNG
KATALON STUDIO VÀ ỨNG DỤNG KIỂM THỬ WEBSITE

CBHD: *Ts. Đỗ Mạnh Hùng*
Sinh viên: Đặng Thị Nguyên

Mã số sinh viên: 2019605472

ĐẶNG THỊ NGUYÊN

NGÀNH CÔNG NGHỆ THÔNG TIN

Hà Nội – Năm 2023

MỤC LỤC

DANH MỤC KÝ HIỆU, CHỮ VIẾT TẮT	1
DANH MỤC CÁC BẢNG BIÊU	2
DANH MỤC HÌNH ẢNH	3
MỞ ĐẦU.....	5
CHƯƠNG 1 - TỔNG QUAN VỀ PHẦN MỀM.....	7
1.1. Định nghĩa phần mềm	7
1.2. Tìm hiểu vòng đời phần mềm	8
1.3. Chất lượng phần mềm và đảm bảo chất lượng phần mềm	9
1.3.1. Chất lượng phần mềm là gì.....	9
1.3.2. Đảm bảo chất lượng phần mềm là gì	10
1.4. Lỗi phần mềm là gì?.....	10
1.4.1. Định nghĩa.....	10
1.4.2. Phân loại lỗi phần mềm.....	10
1.4.3. Các mức độ nghiêm trọng của lỗi	11
1.4.4. Quy trình xử lý lỗi phần mềm.....	12
1.4.5. Một số thông tin cần có về lỗi.....	13
CHƯƠNG 2 - TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM	14
2.1. Kiểm thử tự động và kiểm thử thủ công	14
2.1.1. Khái niệm.....	14
2.1.2. So sánh kiểm thử tự động và kiểm thử thủ công	15
2.1.3. Lý do cần kiểm thử tự động.....	19
2.1.4. Ca kiểm thử	21
2.2. Tìm hiểu chung về kiểm thử ứng dụng trên nền Web.....	23
2.2.1. Khái quát chung	23
2.2.2. Các loại ứng dụng Web	23
2.2.3. Đặc điểm về chất lượng của một ứng dụng trên nền Web....	24
2.2.4. Quy trình kiểm thử.....	26
2.3. Các công việc chính khi kiểm thử trên Web	27

2.3.1. Kiểm thử chức năng	27
2.3.2. Kiểm thử khả năng sử dụng	29
2.3.3. Kiểm thử sự tương thích	30
2.3.4. Kiểm thử hiệu suất	31
2.3.5. Kiểm thử bảo mật.....	32
2.4. Một số công cụ hỗ trợ kiểm thử tự động.....	32
2.4.1. Công cụ kiểm thử hiệu năng	32
2.4.2. Công cụ kiểm thử bảo mật	33
2.4.3. Công cụ kiểm thử chức năng	34
CHƯƠNG 3 – KIỂM THỬ WEBSITE BẰNG KATALON STUDIO.....	36
3.1. Giới thiệu về Katalon Studio	36
3.2. Các tính năng chính của Katalon Studio.....	36
3.3. Làm việc với Katalon Studio	37
3.4. Quy trình làm việc của Katalon Studio.....	38
3.4.1. Khởi tạo (INITIATE)	38
3.4.2. Triển khai (CREATE)	38
3.4.3. Hoạt động (OPERATE).....	38
3.4.4. Báo cáo (REPORT).....	39
3.4.5. Bảo trì (MAINTAIN).....	39
3.5. Cài đặt và cấu hình.....	39
3.5.1. Cài đặt	39
3.5.2. Cấu hình	42
3.6. Một số chức năng trong Katalon Studio	44
3.7. Cách viết một kịch bản với Katalon Studio	48
3.7.1. Cách chụp đối tượng trên màn hình (Lấy ID đối tượng).....	48
3.7.2. Viết kịch bản test với Katalon Studio	53
3.7.3. Một số plugin hỗ trợ kiểm thử ứng dụng Web.....	55
3.8. Ứng dụng Katalon Studio kiểm thử Website Wisami Go	59
3.8.1. Giới thiệu Website chấm công Wisami Go.....	59

3.8.2. Kiểm thử chức năng đăng ký, đăng nhập tài khoản công ty.	60
3.9. Ứng dụng kiểm thử website quản lý điểm trường THPT Liên Hà ..	70
3.9.1. Giới thiệu website	70
3.9.2. Kiểm thử chức năng thêm học sinh	71
KẾT LUẬN	74
TÀI LIỆU THAM KHẢO.....	76
PHỤ LỤC	77

DANH MỤC KÝ HIỆU, CHỮ VIẾT TẮT

STT	Ký hiệu	Cụm từ đầy đủ	Ý nghĩa
1	API	Application Programming Interface	Giao diện lập trình ứng dụng
2	CSS	Cascading Style Sheets	Ngôn ngữ quy định cách hiển thị của các phần tử HTML
3	IEEE	Institute of Electrical and Electronics Engineers	Viết tắt của tổ chức kỹ sư điện và điện tử
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	SEO	Search Engine Optimization	Tối ưu hóa máy tìm kiếm
9	UI	User Interface	Giao diện người dùng

DANH MỤC CÁC BẢNG BIỂU

Bảng 2. 1: So sánh ưu nhược điểm của hai loại kiểm thử	16
Bảng 3. 1: Các yêu cầu hệ thống khi cài đặt Katalon Studio.....	43
Bảng 3. 2: Danh sách trình duyệt hỗ trợ Katalon Studio	44
Bảng 3. 3: Một số chức năng trong Katalon Studio.....	45
Bảng 3. 4: Bảng chi tiết các chức năng trong Test Explorer.....	46
Bảng 3. 5: Bảng test case chức năng đăng ký.....	63
Bảng 3. 6: Bảng test case chức năng đăng nhập	67
Bảng 3. 7: Bảng test case thêm học sinh.....	72

DANH MỤC HÌNH ẢNH

Hình 1. 1: Các mức độ nghiêm trọng của lỗi	11
Hình 1. 2: Quy trình xử lý bug.....	12
Hình 2. 1: Kiểm thử thủ công.....	15
Hình 2. 2: So sánh phạm vi kiểm thử tự động rủi ro chất lượng sản phẩm	16
Hình 2. 3: So sánh chi phí kiểm thử tự động và kiểm thử thủ công	17
Hình 2. 4: Tỉ lệ phần trăm quá trình test	20
Hình 2. 5: Minh họa mẫu kiểm thử cơ bản	22
Hình 3. 1: Giao diện tạo tài khoản đăng ký trên Katalon Studio	40
Hình 3. 2: Download phiên bản Katalon Studio miễn phí	40
Hình 3. 3: Cài đặt ứng dụng Katalon trên máy tính.....	41
Hình 3. 4: Hình ảnh Katalon Studio khi khởi động	41
Hình 3. 5: Giao diện chính của Katalon Studio	42
Hình 3. 6: Giải thích một số chức năng trong Katalon Studio.....	44
Hình 3. 7: Giao diện test case trong Katalon Studio.....	47
Hình 3. 8: Giao diện khởi chạy Katalon Studio	49
Hình 3. 9: Giao diện chọn Skype Web	50
Hình 3. 10: Thêm URL của website cần test.....	51
Hình 3. 11: Sử dụng phím tắt để bắt đối tượng	52
Hình 3. 12: Lưu kết quả sau khi bắt đối tượng trong Katalon Studio.....	53
Hình 3. 13: Lựa chọn đối tượng trong Object Repository	54
Hình 3. 14: Kéo thả ID vào mục Object	54
Hình 3. 15: Kết quả test case nhập tài khoản đã đăng nhập	55
Hình 3. 16: Basic Report.....	56
Hình 3. 17: Text Encoder	56
Hình 3. 18: Text Encoder Pop up	57
Hình 3. 19: Text Encoder Realtime mode	57
Hình 3. 20: Text Encoder Normal mode	58

Hình 3. 21: Giao diện trang chủ Website chấm công Wisami Go	59
Hình 3. 22: Giao diện đăng nhập website Wisami Go.....	59
Hình 3. 23: Giao diện đăng ký website Wisami Go.....	60
Hình 3. 24: Giao diện sau khi đăng nhập.....	60
Hình 3. 25: Test case đăng ký trên Katalon Studio	64
Hình 3. 26: Data test đăng ký được import.....	64
Hình 3. 27: Test suite đăng ký.....	65
Hình 3. 28: Kết quả kiểm thử Test case đăng ký	65
Hình 3. 29: Test case đăng nhập trên Katalon Studio	68
Hình 3. 30: Data đăng nhập.....	68
Hình 3. 31: Test suite đăng nhập	69
Hình 3. 32: Kết quả test chức năng đăng nhập	69
Hình 3. 33: Giao diện đăng nhập của admin.....	70
Hình 3. 34: Giao diện quản lý học sinh.....	70
Hình 3. 35: Test case học sinh.....	72
Hình 3. 36: Data import của test case thêm học sinh.....	72
Hình 3. 37: Test suite ca kiểm thử Học sinh	73
Hình 3. 38: Kết quả sau khi thực hiện test	73

MỞ ĐẦU

1. Lý do chọn đề tài

Tên đề tài: Nghiên cứu công cụ kiểm thử tự động Katalon Studio và ứng dụng kiểm thử Website

Lý do chọn đề tài: Ngày nay xu hướng áp dụng tự động hóa 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. Katalon Studio là một trong những công cụ hỗ trợ kiểm thử tự động tốt nhất hiện nay cho các ứng dụng Web, hoạt động trên hầu hết các trình duyệt phổ biến như Firefox, Chrome, Internet Explorer, Safari, v.v. Công cụ cũng như hỗ trợ số lượng lớn các ngôn ngữ lập trình Web phổ biến. Với mong muốn được trở thành một kỹ sư kiểm thử phần mềm có cái nhìn rõ ràng hơn về kiểm thử tự động trong tương lai, cá nhân em đã lựa chọn đề tài “Nghiên cứu công cụ kiểm thử tự động Katalon Studio và ứng dụng kiểm thử Website” để làm đề tài cho tốt nghiệp đồ án của mình.

Vì thời gian thực hiện đồ án có hạn, do thời gian và kinh nghiệm thực tế còn hạn chế nên có những phần thực hiện chưa được tốt, em rất mong nhận được sự góp ý từ thầy cô và các bạn.

2. Mục đích

Có cái nhìn đúng đắn và sâu sắc hơn về các vấn đề cơ bản của công nghệ phần mềm, lỗi phần mềm và kiểm thử phần mềm.

Hiểu rõ về các thành phần của bộ công cụ Katalon Studio. Nắm được cách thức sử dụng của công cụ Katalon Studio.

Ứng dụng các kiến thức kiểm thử phần mềm và kiến thức về Katalon để viết kịch bản kiểm thử cho một phần mềm cụ thể.

3. Đối tượng và phạm vi nghiên cứu

Hướng đến các Website có nhiều tính năng lặp đi lặp lại nhiều lần, cần sử dụng kiểm thử tự động trong nhà trường và các Website đã được phát triển trong quá trình học.

4. Ý nghĩa

4.1. Ý nghĩa khoa học

Đánh giá được tầm hiểu biết của bản thân về tìm hiểu và sử dụng phần mềm kiểm thử cho Website. Biết được cách hoạt động của phần mềm kiểm thử tự động và áp dụng vào thực tế.

4.2. Ý nghĩa thực tiễn

Sau quá trình tìm hiểu nghiên cứu tài liệu, em đã có thể hiểu được tổng quan về công nghệ phần mềm, lỗi phần mềm, kiểm thử phần mềm nói chung và lĩnh vực web và tìm hiểu, tạo được testcase trên công cụ Katalon Studio.

CHƯƠNG 1 - TỔNG QUAN VỀ PHẦN MỀM

1.1. Định nghĩa phần mềm

Định nghĩa theo IEEE: Phần mềm là các chương trình máy tính, các thủ tục, các tài liệu có liên quan và các dữ liệu để vận hành của một hệ thống máy tính.

➤ Theo như định nghĩa của IEEE, phần mềm gồm 4 thành phần:

Chương trình máy tính (mã nguồn): Thành phần này giúp cho máy tính thực thi các ứng dụng được yêu cầu.

Các thủ tục: Các thủ tục được yêu cầu, để định nghĩa thứ tự và lịch trình mà các chương trình sẽ thực thi, các hàm triển khai, người thực thi các hành động cần thiết cho ứng dụng phần mềm.

Các tài liệu: Có rất nhiều những tài liệu cần thiết với nhân viên phát triển, người sử dụng và các nhân viên bảo trì như: tài liệu thiết kế, tài liệu phân tích, tài liệu hướng dẫn sử dụng, tài liệu hướng dẫn bảo trì.

Dữ liệu cần cho việc vận hành hệ thống: Dữ liệu bao gồm các biến, mã nguồn, danh sách tên thích ứng phần mềm với yêu cầu xác định của khách hàng để vận hành phần mềm.

➤ Định nghĩa công nghệ phần mềm

Công nghệ phần mềm là việc áp dụng các công cụ, kỹ thuật một cách hệ thống trong việc phát triển các ứng dụng dựa trên máy tính. Nói cách khác đó là việc áp dụng các quan điểm, các tiến trình có kỹ thuật và có bài bản vào để phát triển, vận hành và bảo trì phần mềm.

Về cơ bản, công nghệ phần mềm thực hiện các tác vụ chủ yếu sau: thiết kế phần mềm, xây dựng phần mềm, kiểm thử phần mềm và bảo trì phần mềm.

Mục tiêu của công nghệ phần mềm là tạo ra những phần mềm tốt, giảm đến mức tối thiểu sai sót xảy ra trong quá trình vận hành, cũng như tạo điều kiện thuận lợi trong việc bảo trì và nâng cấp phần mềm.

1.2. Tìm hiểu vòng đời phần mềm

Vòng đời phần mềm là một loạt các pha mà phần mềm phải trải qua từ việc khám phá các khái niệm đến khi phần mềm bị loại bỏ hoàn toàn. Mô hình vòng đời phần mềm gồm 7 pha:

Pha yêu cầu (requirements phase): Là pha đầu tiên trong quá trình xây dựng phần mềm, còn gọi là pha tìm hiểu khái niệm (concept exploration). Ở pha này, đại diện nhóm phát triển và khách hàng gặp nhau, khách hàng nêu ra những yêu cầu mà phần mềm phải có, đại diện nhóm phát triển ghi chép lại. Nếu dịch theo tiếng Anh “requirements phase” là pha yêu cầu. Tuy nhiên cách gọi này có thể hơi khó hiểu, do đó người ta thường gọi là pha xác định yêu cầu.

Pha đặc tả (specification phase): Pha này phân tích các yêu cầu của khách hàng. Mô tả các kết quả phân tích dưới dạng “tài liệu đặc tả”. Cuối giai đoạn này, kế hoạch quản lý dự án phần mềm được đưa ra, mô tả chi tiết quá trình phát triển phần mềm. Câu hỏi mà pha này cần trả lời là: “Phần mềm sẽ làm gì?”

Pha thiết kế (design phase): Là pha tiếp theo pha đặc tả. Căn cứ vào tài liệu đặc tả, pha này mô tả cách thức mà phần mềm thực hiện các công việc cụ thể. Pha này trả lời câu hỏi “Phần mềm sẽ được làm như thế nào?”. Pha thiết kế thường có 2 phần: thiết kế kiến trúc (architecture design) và thiết kế chi tiết (detail design). Thiết kế kiến trúc phân chia phần mềm thành các thành phần gọi là mô-đun. Thiết kế chi tiết là thiết kế từng mô-đun một cách chi tiết.

Pha cài đặt (implementation phase): Ở pha này, người ta thực hiện viết chương trình bằng một ngôn ngữ lập trình cụ thể.

Pha tích hợp (imtergration phase): Kết nối các mô-đun đã viết của chương trình thành một phần mềm thống nhất và chạy thử, chỉnh sửa cho đến khi phần mềm chạy tốt.

Pha bảo trì (maintenance phase): Khi chương trình đã được cài đặt trên máy của khách hàng vẫn có thể có lỗi phát sinh trong phần mềm cần loại trừ và điều chỉnh lại theo yêu cầu của khách hàng. Công việc bảo trì được chia thành hai loại: bảo trì sửa lỗi (software repair) và bảo trì cập nhật (software update). Bảo trì cập nhật lại được chia thành hai loại: Bảo trì hoàn thiện (perfective

maintenance) và bảo trì thích nghi (adaptive maintenance). Bảo trì hoàn thiện là sửa lỗi phần mềm thích nghi với môi trường mới. Người ta tính toán rằng bảo trì sửa lỗi và thích nghi chiếm thời gian gần bằng nhau và bằng 20% thời gian bảo trì, còn bảo trì hoàn thiện chiếm thời gian khoảng gấp 3 lần mỗi loại bảo trì kia (khoảng 60%).

Pha loại bỏ (retirement phase): Đến một thời điểm nào đó, do sự thay đổi của môi trường làm việc và một số yếu tố khác, chi phí bảo trì phần mềm sẽ lớn hơn rất nhiều so với chi phí phát triển một phần mềm mới. Đó là lúc việc loại bỏ phần mềm được thực hiện.

1.3. Chất lượng phần mềm và đảm bảo chất lượng phần mềm

1.3.1. Chất lượng phần mềm là gì

- Định nghĩa theo IEEE (1991):

Định nghĩa 1: Chất lượng phần mềm là một mức độ mà một hệ thống, thành phần hệ thống hay tiến trình đáp ứng được yêu cầu đã được đặc tả.

Định nghĩa 2: Chất lượng phần mềm là mức độ mà một hệ thống, thành phần hệ thống hay tiến trình đáp ứng được yêu cầu và sự mong đợi của khách hàng hay người sử dụng.

- Phân tích hai quan điểm của IEEE:

Theo quan điểm thứ nhất: Nếu theo quan điểm này, chúng ta sẽ bị phụ thuộc quá nhiều vào tài liệu đặc tả yêu cầu, dẫn đến nếu việc xác định yêu cầu bị sai, thiếu thì một phần mềm làm đúng với đặc tả chưa chắc đã là một phần mềm có chất lượng.

Theo quan điểm thứ hai: Khách hàng đôi khi không có nhiều kiến thức về công nghệ, họ có thể đưa ra những mong muốn hết sức vô lý và có thể thay đổi yêu cầu với phần mềm nhiều lần thậm chí thay đổi ngay trong giai đoạn cuối. Điều này gây rất nhiều khó khăn cho việc phát triển phần mềm.

- Định nghĩa theo Pressman:

Chất lượng phần mềm là sự phù hợp của các yêu cầu cụ thể về hiệu năng và chức năng, các tiêu chuẩn phát triển phần mềm được ghi lại rõ ràng bằng tài liệu với các đặc tính ngầm định của tất cả các phần mềm được phát triển chuyên nghiệp.

Định nghĩa của Pressman đề xuất ba yêu cầu với chất lượng phần mềm phải được đáp ứng khi phát triển phần mềm:

Các yêu cầu chức năng rõ ràng là nhân tố quyết định chất lượng đầu ra của phần mềm.

Các tiêu chuẩn chất lượng phần mềm sẽ được nói đến trong hợp đồng.

Các đặc tính ngầm định cần được đáp ứng trong quá trình phát triển cho dù không được nói đến rõ ràng trong hợp đồng.

1.3.2. Đảm bảo chất lượng phần mềm là gì

Định nghĩa theo Daniel Galin: Đảm bảo chất lượng phần mềm (Software Quality Assurance) là một tập hợp các hành động cần thiết được lên kế hoạch một cách hệ thống để cung cấp đầy đủ niềm tin rằng quá trình phát triển phần mềm phù hợp để thành lập các yêu cầu chức năng kỹ thuật cũng như các yêu cầu quản lý theo lịch trình và hoạt động trong giới hạn ngân sách.

1.4. Lỗi phần mềm là gì?

1.4.1. Định nghĩa

Định nghĩa lỗi phần mềm: Có rất nhiều định nghĩa về lỗi phần mềm nhưng có thể hiểu và phát biểu một cách đơn giản thì “Lỗi phần mềm là sự không khớp giữa chương trình và đặc tả của nó”.

1.4.2. Phân loại lỗi phần mềm

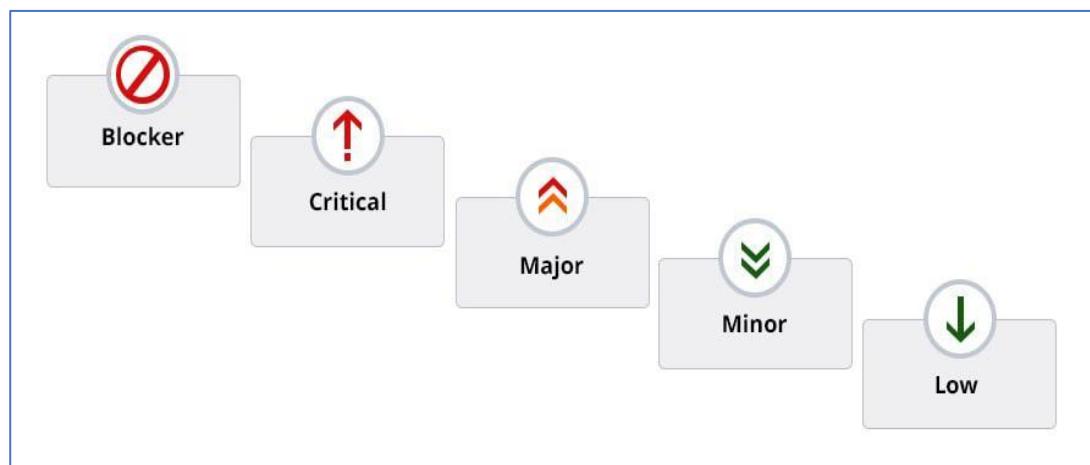
Dựa vào định nghĩa, ta có thể phân loại lỗi phần mềm thành 3 dạng:

- **Lỗi sai:** Sản phẩm phần mềm được xây dựng khác với đặc tả.
- **Lỗi thiếu:** Các yêu cầu của sản phẩm phần mềm đã có trong đặc tả nhưng lại không có trong sản phẩm thực tế.

- **Lỗi thừa:** Sản phẩm thực tế có những tính năng không có trong tài liệu đặc tả

1.4.3. Các mức độ nghiêm trọng của lỗi

Chương trình một khi đã xuất hiện lỗi đều kéo theo những hệ lụy nghiêm trọng. Một trong những cách phân loại mức độ nghiêm trọng của lỗi thường được sử dụng dựa trên tần suất xuất hiện: chỉ một lần, thỉnh thoảng, xuất hiện lại hay lặp đi lặp lại nhiều lần. Việc phân loại mức độ nghiêm trọng của lỗi sẽ giúp kiểm thử viên cũng như lập trình viên ý thức được đâu là lỗi cần được giải quyết trước, nhằm giảm thiểu tối đa những tổn thất về chi phí và nâng cao chất lượng cho sản phẩm phần mềm. Mức độ nghiêm trọng có thể có một số loại:



Hình 1. 1: Các mức độ nghiêm trọng của lỗi

Blocker: là một loại lỗi chặn kiểm tra thêm do ứng dụng hoặc phần mềm bị treo trong một môi trường cụ thể do lỗi.

Critical: là các lỗi hay defect có thể làm dừng hoàn toàn chương trình hay hệ thống, hay là lỗi hỏng về bảo mật dữ liệu của khách hàng dẫn đến mất dữ liệu hoặc thiệt hại nghiêm trọng khác.

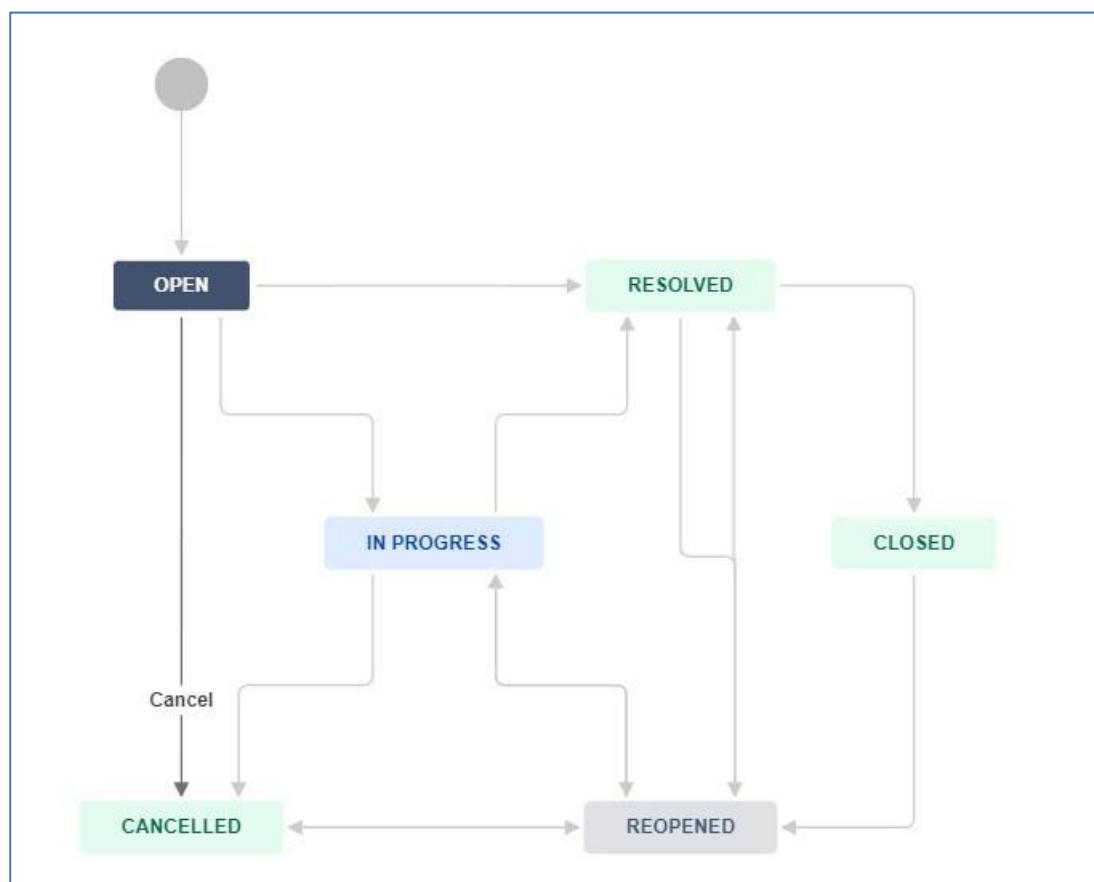
Major: Các lỗi này cũng là những lỗi làm chương trình hay các chức năng bị ngừng hoạt động, tuy nhiên vẫn có cách để thay thế và chấp nhận được. Khi đó chúng ta sẽ đặt mức độ ưu tiên ở Major.

Minor: Các defect này thường là các lỗi về giao diện, nó không cho chúng ta kết quả như thiết kế nhưng có thể sửa dễ dàng.

Low: Lỗi có mức độ nghiêm trọng thấp có ít ảnh hưởng đến hoạt động của chương trình. Chúng thường được tìm thấy trong quá trình kiểm tra giao diện người dùng

1.4.4. Quy trình xử lý lỗi phần mềm

Trước khi giới thiệu về quy trình xử lý lỗi phần mềm, đồ án sẽ trình bày về các trạng thái có thể có của lỗi.



Hình 1. 2: Quy trình xử lý bug

Trong đó:

- **Open:** Khi phát hiện ra lỗi, kiểm thử viên sẽ log bug và assigned đến cho lập trình viên. Để lập trình viên điều tra, xem xét
- **In Progress:** Lập trình viên bắt đầu thảo luận điều tra, tái hiện lại bug và thực hiện thay đổi để fix bug cho đúng yêu cầu

- **Resolved:** Lập trình viên đã xử lý lỗi
 - **Canceled:** Khi lập trình viên điều tra, và xác định lỗi không phải lỗi, lỗi được bỏ qua. Hay lỗi không thuộc phạm vi của dự án, hoặc sẽ được xử lý trong một giai đoạn khác của dự án
 - **Closed:** Trạng thái đóng, lỗi đã được sửa thành công
 - **Reopened:** Nếu như kiểm thử viên test lại mà bug đó vẫn xảy ra thì bug đó sẽ được gán là Re-Open và assign lại cho developer để fix lại
- Mỗi tổ chức phát triển phần mềm sẽ sử dụng một công cụ quản lý lỗi riêng, trong đó phải kể đến Jira hay Redmine hiện đang là những công cụ được sử dụng phổ biến và đem lại hiệu quả cho các tổ chức. Giúp các tổ chức tối ưu hóa việc quản lý bug, tránh bị thiếu bug.

1.4.5. Một số thông tin cần có về lỗi

Category: Thư mục lỗi dùng để phân loại lỗi, lỗi thuộc phần chức năng nào phải chọn đúng phần thư mục lỗi tương ứng để thuận tiện cho việc tra cứu, thống kê lỗi của chức năng.

Severity (mức độ nghiêm trọng của lỗi): Thông số này biểu hiện độ nghiêm trọng của lỗi.

Priority: Mức độ ưu tiên trong việc sửa lỗi.

Summary: Tóm tắt nội dung lỗi, có thể coi là tiêu đề của lỗi.

Description: Đây là phần mô tả lỗi, phải mô tả rõ 3 phần nội dung:

- Các bước thực hiện
- Kết quả trả về từ hệ thống
- Kết quả mong muốn

Comment: Dùng để đưa ra các lưu ý, trao đổi về lỗi của các thành viên trong dự án.

CHƯƠNG 2 - TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM

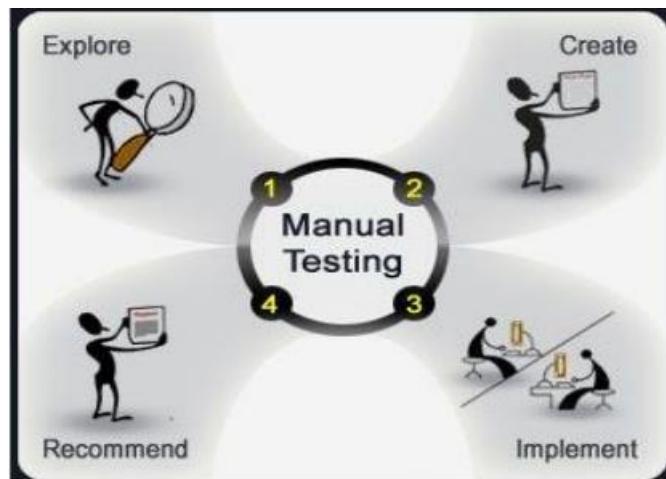
2.1. Kiểm thử tự động và kiểm thử thủ công

2.1.1. Khái niệm

Kiểm thử phần mềm (software testing): Là hoạt động nhằm tìm kiếm và phát hiện ra các lỗi của phần mềm, đảm bảo phần mềm chính xác, đúng và đầy đủ theo yêu cầu của khách hàng, yêu cầu của sản phẩm đã đặt ra. Software testing cũng cung cấp mục tiêu, cái nhìn độc lập về phần mềm điều này cho phép đánh giá và hiểu rõ các rủi ro khi thực thi phần mềm.

Kiểm thử thủ công (manual test): Là một loại kiểm thử phần mềm trong đó các trường hợp kiểm thử được người kiểm thử thực hiện theo cách thủ công mà không sử dụng bất kỳ công cụ tự động nào. Mục đích của Manual testing là xác định các lỗi, sự cố và khiếm khuyết trong ứng dụng phần mềm. Kiểm thử phần mềm thủ công là kỹ thuật nguyên thủy nhất trong tất cả các loại kiểm thử và nó giúp tìm ra các lỗi nghiêm trọng trong ứng dụng phần mềm. Tester làm mọi công việc hoàn toàn bằng tay, từ viết test case 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.

Kiểm thử tự động (automation test): 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 nhảm 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.



Hình 2. 1: Kiểm thử thủ công

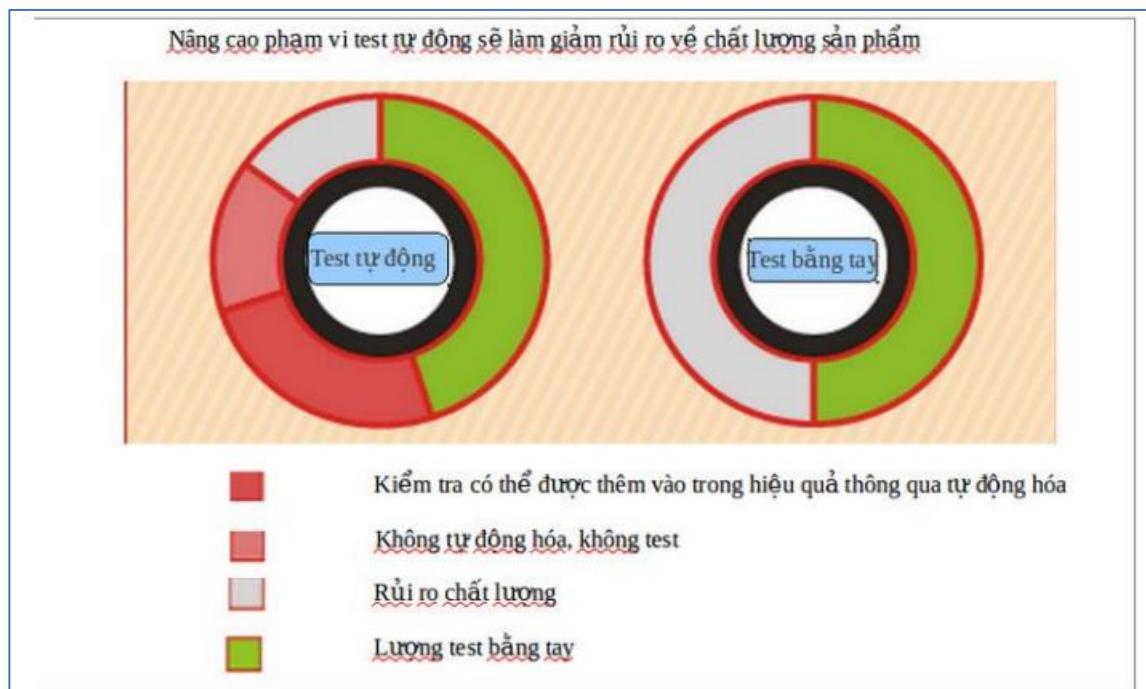
2.1.2. So sánh kiểm thử tự động và kiểm thử thủ công

Ưu nhược điểm của hai quy trình:

Kiểm thử	Điểm mạnh	Điểm yếu
Thủ công	<ul style="list-style-type: none"> - 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 	<ul style="list-style-type: none"> - 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
Tự động	<ul style="list-style-type: none"> - 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 	<ul style="list-style-type: none"> - Tốn kém hơn kiểm thử thủ công, chi phí đầu tư ban đầu lớn - Không thể thay thế kiểm thử thủ công vì người ta không thể tự động hóa mọi thứ

	liệu, click, check kết quả, ..) giúp tester không phải làm những thao tác nhiều lần gây nhảm chán và dễ nhầm lẫn - Giảm chi phí đầu tư dài hạn	
--	---	--

Bảng 2. 1: So sánh ưu nhược điểm của hai loại kiểm thử



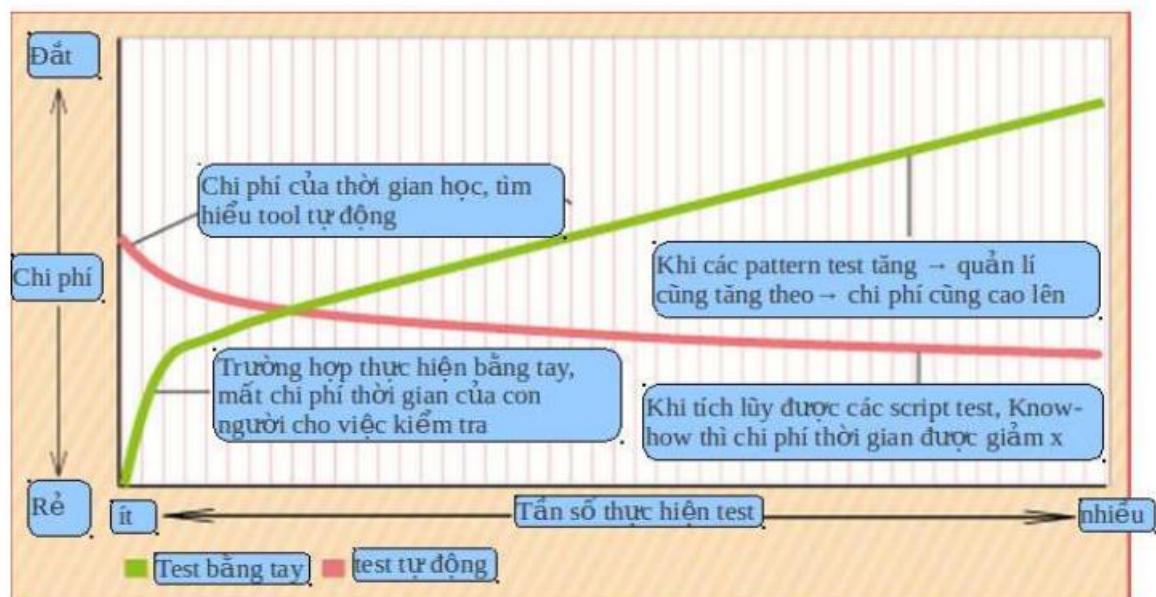
Hình 2. 2: So sánh phạm vi kiểm thử tự động rủi ro chất lượng sản phẩm

Theo như bản đồ hình 2.2 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 2. 3: So sánh chi phí kiểm thử tự động và kiểm thử thủ công

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 quy
- 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:

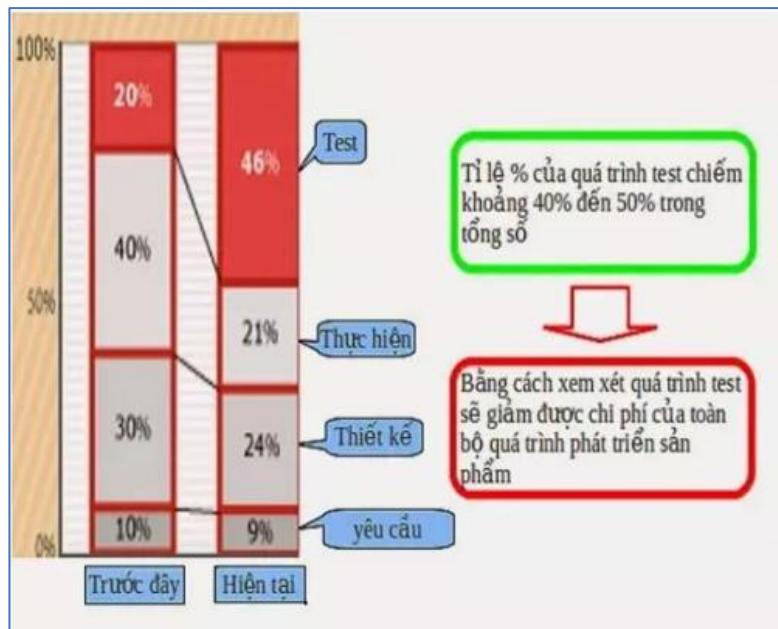
- Kiểm thử thủ công là phù hợp nhất với các khu vực / kịch bản sau đây:
 - Kiểm thử thăm dò: Đây là loại kiểm thử đòi hỏi phải thử nghiệm của kiến thức, kinh nghiệm, phân tích / logic kỹ năng, sáng tạo và trực giác.
 - Xét nghiệm này được đặc trưng bởi các tài liệu ở đây kém bằng văn bản kỹ thuật, hoặc 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.
- Thủ 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

2.1.3. Lý do cần kiểm thử tự động

2.1.2.1. Mục tiêu của kiểm thử tự động

Trong phát triển phần mềm, quá trình kiểm thử có ý nghĩa quan trọng nhất. Môi trường xung quanh việc phát triển phần mềm hiện nay được tìm kiếm đó là “Thời kì phát triển ngắn hạn”, “Kinh phí thấp”, “Chất lượng cao”. Đặc biệt, trong quá trình làm sản phẩm việc test chiếm khoảng 40% trong tổng số. Ngay cả khi vẫn duy trì chất lượng và hiệu quả, để đáp ứng nhu cầu cao hơn và chi phí thấp hơn thì điều này đặc biệt được quan tâm đề cao. Để khắc phục tình hình như vậy thì kiểm thử tự động đã và đang được chú ý tới.



Hình 2. 4: Tỉ lệ phần trăm quá trình test

2.1.2.2. Lý do nên chọn kiểm thử tự động

Khi phát triển phần mềm, việc thực hiện kiểm thử là bắt buộc, cho dù người thực hiện có thể là developer hoặc là tester. Vì thế, có kiến thức về kiểm thử, lựa chọn loại hình kiểm thử phù hợp với sản phẩm là điều cần thiết cho bất cứ người nào tham gia vào quá trình làm sản phẩm. Mỗi loại hình kiểm thử đều có điểm mạnh và điểm yếu riêng. Với kiểm thử tự động một số ưu điểm sau sẽ là những điều kiện tiên đề cho sự lựa chọn của mọi người.

- Cải thiện hiệu quả

Đầu tiên, lợi ích cụ thể được nói đến là “Nâng cao hiệu quả”. Khi cần kiểm tra hồi quy hay phải hao phí về mặt thời gian thì kiểm thử tự động mang lại hiệu quả rõ rệt (có thể thực hiện kiểm thử ngay cả khi không có người bắt kể ngày hay đêm)

- Cải thiện độ chính xác

Khi dùng kiểm thử tự động, dù có lặp đi lặp lại bao nhiêu lần thì cũng cho ra các thao tác và kết quả giống nhau. Do đó tránh được những rủi ro không cần thiết. Ngoài ra, nếu một lỗi được tìm thấy, nó có thể được tái tạo bằng cách đơn giản là thực hiện cùng một kịch bản tự động, dẫn đến cải thiện khả năng

tái lỗi. Kiểm thử tự động còn có tính năng các thao tác test được lưu lại tự động, dễ dàng kiểm tra và cưỡng chế lỗi trong thời gian kiểm thử

- Cải thiện chất lượng kiểm thử

2.1.4. Ca kiểm thử

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 kiện 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 results: Mô tả kết quả thực tế khi thực hiện kiểm thử trên môi trường của hệ thống. Actual results 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) ... 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.

KỊCH BẢN KIỂM THỬ								
Mã kiểm thử	Mục đích kiểm thử	Các bước thực hiện	Kết quả mong muốn	Chrome 56.0	Kết quả	Mã lỗi	Mô tả lỗi	Ghi chú
<i>Giao diện</i>								
<i>Giao diện chung</i>								
LOGIN_1	Kiểm tra màn hình ở trạng thái mặc định	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra các giá trị mặc định của các trường 4. Kiểm tra header, footer	Màn hình hiển thị thông tin sau: - Textbox Tên đăng nhập - Textbox Mật khẩu - Button Đăng nhập - Captcha Focus chuyển từ textbox Tên đăng nhập	P	P			
LOGIN_2	Kiểm tra tổng thể giao diện màn hình	1. Kiểm tra về bộ cục, font chữ, chỉnh tả, màu chữ 2. Kiểm tra trang bắt buộc phải có dấu (*)	1. Các label, textbox, combo có độ dài, rộng và khoảng cách bằng nhau, không xô lệch 2. Các label sử dụng cùng 1 loại font, có chất, cân bằng 3. Các trường hợp bắt buộc nhập phải có dấu (*) 4. Kiểm tra tất cả lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình 5. Hình ảnh hiển thị captcha phải rõ ràng, có khả năng đọc được	F	P	P	- TH hiển thị captcha và message cảnh báo đăng nhập sai cần chính xác đăng nhập cần giữa màn hình - Sửa label thành MÃ BẢO VỆ thay vì CAPTCHA như hiện tại - Bổ sung tooltip cho icon Đổi mã bảo vệ - Mã captcha cần hiển thị rõ ràng hơn, hiện tại hơi nhiều khó đọc	
LOGIN_3	Kiểm tra khi reload captcha	Click icon reload captcha	Đổi sang màn hình khác	P	P	P	455598	Không đổi được captcha trên Firefox
LOGIN_4	Kiểm tra thứ tự tab khi không chọn hiển thị captcha	Nhấn Tab liên tục	Con trỏ di chuyển lặp lại trên các trường input, button, hyperlink theo thứ tự trái sang phải, trên xuống dưới	P		P		
LOGIN_5	Kiểm tra thứ tự tab khi có hiển thị captcha	Nhấn Tab liên tục	Con trỏ di chuyển lặp lại trên các trường input, button, hyperlink theo thứ tự trái sang phải, trên xuống dưới	P		P		
LOGIN_6	Kiểm tra thực hiện chức năng chỉnh cửa màn hình khi nhấn Enter	Nhấn phím Enter	Thực hiện submit đăng nhập khi nhấn phím Enter	P		P		
LOGIN_7	Kiểm tra giao diện khi thu nhỏ, phóng to	1.Nhấn phím Ctrl + 2.Nhấn phím Ctrl -	Màn hình thu nhỏ, phóng to normal ứng và không bị vỡ giao diện	P		P		
	Textbox Tên đăng nhập							

Hình 2. 5: Minh họa mẫu kiểm thử cơ bản

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 kiểm thử nghiêm 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 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 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.

2.2. Tìm hiểu chung về kiểm thử ứng dụng trên nền Web

2.2.1. Khái quát chung

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... 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... 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... 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.

2.2.2. 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 nhiều lần.

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, ... Đối với Web động, việc cập nhật nội dung là rất đơn giản và dễ dàng. Không những thế, một số hệ thống 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.

2.2.3. 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 ... 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 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 được 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 khó bắt gặp những ứng dụng chạy trên nền Web ghi chú thích “Website chạy (tương thích) tốt nhất trên trình duyệt X”. 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 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 trống 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 đáo 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.

- 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 lại 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...

- **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... 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 tế, 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 đến chất lượng nội dung của ứng dụng để máy tìm kiếm chú ý đến.

2.2.4. Quy trình kiểm thử

Kiểm thử phần mềm/ứng dụng bao gồm nhiều giai đoạn với sự phối hợp của nhiều bên liên quan chứ không chỉ là một hoạt động đơn lẻ. Chính vì thế cần có quy trình kiểm thử phần mềm để làm rõ các công đoạn, các bước kiểm thử, người chịu trách nhiệm và khi nào việc kiểm thử được tiến hành trong toàn bộ quy trình phát triển phần mềm. Nói cách khác, quy trình kiểm thử phần mềm chính là chuỗi các hoạt động được tiến hành để thực hiện việc kiểm thử.

- **Phân tích yêu cầu:** Nhóm kiểm thử sẽ tương tác với các bên liên quan để hiểu rõ những yêu cầu cụ thể cần cho việc kiểm thử. Các yêu cầu

có thể là chức năng (xác định phần mềm cần phải làm những gì) hoặc phi chức năng (hiệu năng, tính bảo mật hệ thống, màu sắc...)

- Lên kế hoạch kiểm thử: Còn được gọi bằng tên khác là lên chiến lược kiểm thử. Ở giai đoạn này, trưởng nhóm kiểm thử sẽ được dự toán chi phí cho dự án cũng như chuẩn bị kế hoạch kiểm thử.
- Tạo ca kiểm thử: Giai đoạn này cần phải tạo, xác minh, kiểm tra lại các ca kiểm thử. Dữ liệu kiểm thử cũng được tạo và xác định trong giai đoạn này.
- Cài đặt môi trường kiểm thử: Môi trường kiểm thử quyết định bởi các điều kiện phần cứng và phần mềm trong từng dự án. Thiết lập môi trường kiểm thử có thể thực hiện song song với giai đoạn sinh ca kiểm thử và là một tiêu chí quan trọng trong quá trình kiểm thử. Tuy nhiên, nhóm kiểm thử có thể không cần tham gia vào giai đoạn này nếu đã có các bên liên quan khác hỗ trợ, nhiệm vụ của nhóm kiểm thử chỉ là yêu cầu môi trường kiểm thử cần thiết.
- Thực hiện kiểm thử: Nhóm kiểm thử thực hiện kiểm thử theo kế hoạch và danh sách ca kiểm thử đã chuẩn bị từ giai đoạn trước. Các lỗi phát triển ở giai đoạn này sẽ được thông báo lại cho nhóm phát triển phần mềm để chỉnh sửa và thực hiện kiểm thử lại.
- Đóng chu trình kiểm thử: Nhóm kiểm thử sẽ họp, thảo luận và phân tích những bài học rút ra sau quá trình kiểm thử, đưa ra chiến lược cho những lần kiểm thử kế tiếp hoặc chia sẻ kinh nghiệm cho những dự án tương tự.

2.3. Các công việc chính khi kiểm thử trên Web

2.3.1. Kiểm thử chức năng

Khái niệm: Kiểm tra các tính năng (feature), các hành vi hoạt động của ứng dụng để đảm bảo ứng dụng đúng với các thông số kỹ thuật của sản phẩm,

chỉ tập trung vào các đầu ra (output) tương ứng với các đầu vào (input) kèm theo đó là các điều kiện đi cùng

Mục đích: Để xác minh xem sản phẩm hay ứng dụng có đáp ứng được các đặc điểm chức năng như đã được đề cập trong tài liệu phát triển của dự án hay không.

Quan điểm test chung:

- 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 bộ 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 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 viên sẽ cần kiểm tra tất cả các trường bắt buộc, các dấu, message thông báo, báo lỗi, tất cả các dữ liệu bên trong combobox/list có được sắp xếp đúng, các trường hợp input giá trị, thao tác của các chức năng. Kiểm tra xem khi bất kỳ chức năng nào người dùng thao tác nhưng bị thất bại thì có chuyển hướng đến trang thông báo lỗi hay không.

Quan điểm test cần lưu ý:

- Trong một ứng dụng Web có hai loại liên kết: liên kết nội bộ (internal link) và liên kết ngoại bộ (external link). Cả hai loại liên kết đều cần được kiểm tra xem chúng có hoạt động không? Có trả đến địa chỉ mong muốn không? Cần đảm bảo rằng các liên kết không tự trả đến vị trí của chính nó. Cần xem xét thuộc tính “target” của các liên kết xem chúng có hoạt động đúng như bản thiết kế yêu cầu hay không. Ngoài ra còn có Email links – cần đảm bảo rằng nếu người dùng nhấp vào liên kết email thì ứng dụng email mặc định sẽ được mở và địa chỉ To có thể được điền trước. Với Broken links – liên kết gãy, liên kết

bị hỏng hay được gọi là liên kết chết. Các liên kết này không được liên kết với bất kỳ trang nào trong số các trang nội bộ hoặc các trang bên ngoài của trang web.

- Kiểm thử form web: thường được sử dụng nhiều nhất trong các trang web (chú ý vào các phần hiển thị, validate, và các button). Kiểm tra logic validation cho từng trường. Kiểm tra các trường mật khẩu không hiển thị nội dung mật khẩu. Kiểm tra giá trị đầu vào không hợp lệ của từng trường. Validation phản hồi của một form submit.

- Kiểm thử session và quản lý cookie: Kiểm tra phiên đăng nhập ứng dụng/trang web bằng cách bật và tắt cookie. Kiểm tra xem cookie có được reset lại giữa các phiên bản trình duyệt hay không. Kiểm tra bảo mật ứng dụng bằng cách chọn lọc xóa cookie trong test operates.

- 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, xoá, v.v. hoạt động.

2.3.2. 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 hóa 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 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.

2.3.3. 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

2.3.4. 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.

2.3.5. 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. Lỗi hỏng 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. Một số công cụ hỗ trợ kiểm thử tự động

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 nhảm 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.

2.4.1. 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 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.

2.4.2. 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 InjectMe được sử dụng để kiểm tra các lỗ hổng SQL Injection trong ứng dụng Web.

2.4.3. 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.

- Katalon Studio: cũng như Selenium - là một công cụ miễn phí dành cho cộng đồng kiểm thử phần mềm tự động. Nếu Katalon Studio được xây dựng từ Selenium/Appium, tại sao chúng ta không dùng trực tiếp hai công cụ phổ biến trong cộng đồng kiểm thử này mà cần phải dùng đến Katalon Studio.

CHƯƠNG 3 – KIỂM THỬ WEBSITE BẰNG KATALON STUDIO

3.1. Giới thiệu về Katalon Studio

Katalon Studio là một bộ công cụ toàn diện để kiểm thử tự động ứng dụng Web và Mobile. Công cụ này bao gồm một gói đầy đủ các tính năng mạnh mẽ giúp vượt qua các thách thức phổ biến trong kiểm thử tự động giao diện người dùng web, ví dụ: cửa sổ bật lên, iFrame và thời gian chờ. Giải pháp thân thiện và linh hoạt này giúp cho người kiểm tra tốt hơn, làm việc nhanh hơn và khởi chạy phần mềm chất lượng cao nhờ vào sự thông minh mà nó cung cấp cho toàn bộ quy trình kiểm thử tự động.

3.2. Các tính năng chính của Katalon Studio

Triển khai đơn giản (Simple deployment): gói triển khai duy nhất, gắn kết chứa mọi thứ cần để triển khai một công cụ kiểm thử tự động mạnh mẽ.

Cài đặt nhanh chóng và dễ dàng (Quick & easy set-up): không chỉ cung cấp cài đặt đơn giản, Katalon Studio còn giúp tester dễ dàng thiết lập môi trường.

Người kiểm thử có thể chạy kịch bản kiểm thử đầu tiên của họ khá nhanh chóng bằng cách sử dụng các mẫu và tập lệnh kiểm thử dựng sẵn của nó, chẳng hạn như kho đối tượng và thư viện từ khóa.

Kết quả nhanh hơn và tốt hơn (Faster & Better results): các mẫu dựng sẵn với hướng dẫn rõ ràng giúp người kiểm tra nhanh chóng xây dựng và chạy các kịch bản kiểm thử tự động. Có thể thực hiện từng bước một tốc độ và hiệu quả, từ thiết lập dự án, tạo kiểm thử, thực hiện, tạo báo cáo và bảo trì.

Chế độ linh hoạt (Flexible modes): có thể sử dụng bản ghi và từ khóa để xây dựng bài kiểm thử tự động, trong khi có IDE đầy đủ để xây dựng tập Lệnh nâng cao.

Dễ sử dụng (Ease of use): ngay cả thủ công với kinh nghiệm lập trình tối thiểu cũng có thể khai thác lợi ích của nó một cách dễ dàng.

Ứng dụng đa trình duyệt (Cross-browser application): Katalon Studio hỗ trợ nhiều nền tảng: Windows 32 và 64 (7, 8, và 10) và OS X 10.5+.

3.3. Làm việc với Katalon Studio

Katalon Studio là một giải pháp tự động hóa sâu sắc, thân thiện với người dùng, được đặc trưng bởi sự đơn giản và tốc độ. Nó rất hữu ích cho các nhóm và cá nhân người kiểm thử tự động dành ít nỗ lực nhất từ việc thiết lập một dự án mới đến thực hiện các kiểm thử và sau đó giám sát kết quả thực hiện. Mỗi quy trình công việc được cung cấp rất nhiều khả năng và tùy chỉnh để bảo trì dễ dàng và mở rộng dự án:

- Cấu trúc được xác định trước (Pre-defined structure): test cases, test suites, test objects, reports. Người kiểm tra không cần phải dành hàng giờ để xác định và duy trì chúng sau này.
- Từ khóa (Custom keywords) tùy chỉnh cung cấp tính linh hoạt trong việc thêm các từ khóa bổ sung để kiểm tra AUT hiệu quả cho các mục đích thử nghiệm cụ thể và phức tạp.
- Hỗ trợ các nhu cầu kiểm thử chính: web, mobile và API.
- Thực hiện nhiều bộ kiểm thử cùng một lúc với bộ sưu tập bộ kiểm thử.
- Mở rộng dòng CI hiện tại một cách dễ dàng với việc thực hiện chế độ bàn điều khiển mà không cần nỗ lực. Thực hiện dòng lệnh có thể được tạo ra nhanh chóng bằng cách sử dụng tính năng 'Generate Command Line for console mode'.
- Giám sát kết quả thực hiện dễ dàng với chế độ xem bảng hoặc chế độ xem cây trong/ sau khi thực hiện.
- Báo cáo chi tiết bộ kiểm thử giảm thời gian phân tích kết quả. Có thể xuất nó sang các định dạng khác nhau như CSV, PDF, HTML và lưu trữ để sử dụng sau.

3.4. Quy trình làm việc của Katalon Studio

Điều hành một công việc tuyển tính:

3.4.1. Khởi tạo (INITIATE)

- Các mẫu dự án tích hợp: bằng cách cung cấp các mẫu dựng sẵn để tổ chức các trường hợp kiểm thử, lưu trữ đối tượng và từ khóa, Katalon Studio giúp việc kiểm thử dễ dàng hơn cho người kiểm tra.
- Nhiều khả năng: hỗ trợ đầy đủ kiểm thử cho Web, Android, iOS, và API trên tất cả các hệ điều hành.
- Tích hợp trên công cụ không rắc rối: dễ dàng kết hợp với Jenkins, GIT và JIRA với các plug-in gốc.

3.4.2. Triển khai (CREATE)

- Tạo kiểm thử tự động: ghi lại các hành động và tạo các kịch bản tự động bằng các từ khóa tích hợp.
- Tập lệnh Hi-end: cho phép xây dựng kịch bản kiểm thử nâng cao hoặc từ khóa tùy chỉnh dễ dàng và hiệu quả.
- Thu thập thông tin đối tượng: máy ghi âm tiên tiến phát hiện các thuộc tính đối tượng một cách hiệu quả để tối đa hóa nhận dạng.

3.4.3. Hoạt động (OPERATE)

- Thực hiện kiểm thử (Test execution) mạnh mẽ: chạy các trường hợp kiểm thử hoặc bộ kiểm thử bằng cách sử dụng nhiều cấu hình và bộ dữ liệu.
- Tính linh hoạt trong thực thi: cung cấp bảng điều khiển tích hợp CI với các tham số khác nhau để thực hiện từ xa. Chạy thử nghiệm trên nhiều trình duyệt và hệ điều hành OS cục bộ hoặc với Sauce Labs và BrowserStack.

- Xử lý lỗi linh hoạt và thực hiện lại tự động: bao gồm các quy tắc thời gian chạy để tự động xử lý các luồng thực thi phức tạp.

3.4.4. Báo cáo (REPORT)

- Báo cáo có sẵn trong một số định dạng: với ghi nhật kí nâng cao, dữ liệu gỡ lỗi và ảnh chụp màn hình.
- Báo cáo thực hiện Bespoke: được tích hợp với quy trình thông báo của bạn.
- Nhật kí Selenium và Appium cải tiến: với các tính năng phân tích được cải tiến để cải thiện chiến lược tự động hóa.

3.4.5. Bảo trì (MAINTAIN)

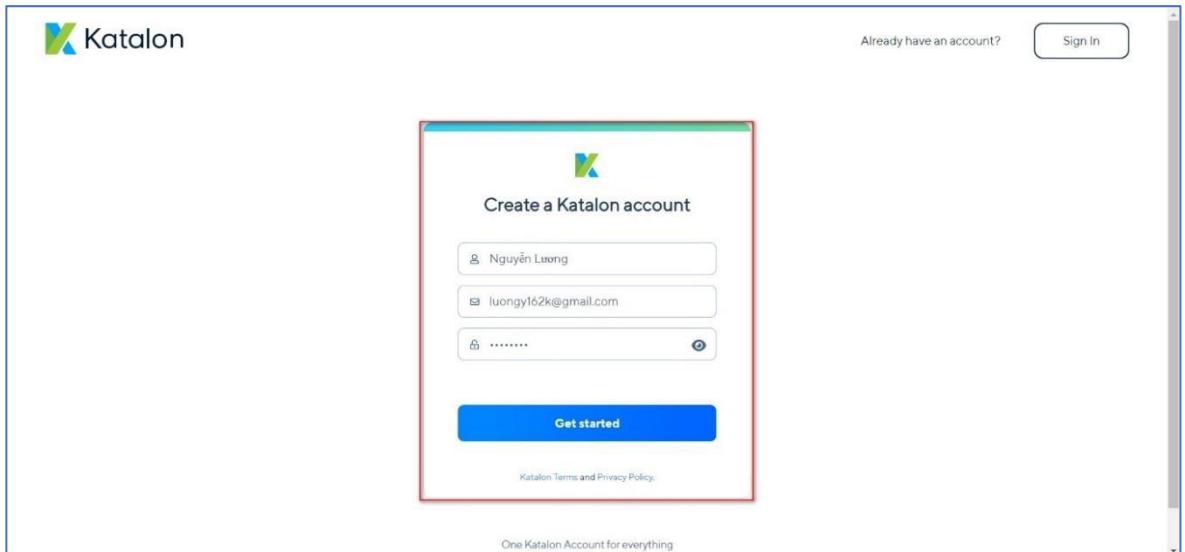
- Bảo trì đối tượng kiểm thử thông minh: tự động cập nhật tất cả các trường hợp và bộ kiểm thử liên quan khi các đối tượng được thay đổi.
- Tổ chức kiểm thử hiệu quả: cho phép dễ dàng quản lý và bảo trì các bài kiểm thử, dữ liệu và từ khóa.

3.5. Cài đặt và cấu hình

3.5.1. Cài đặt

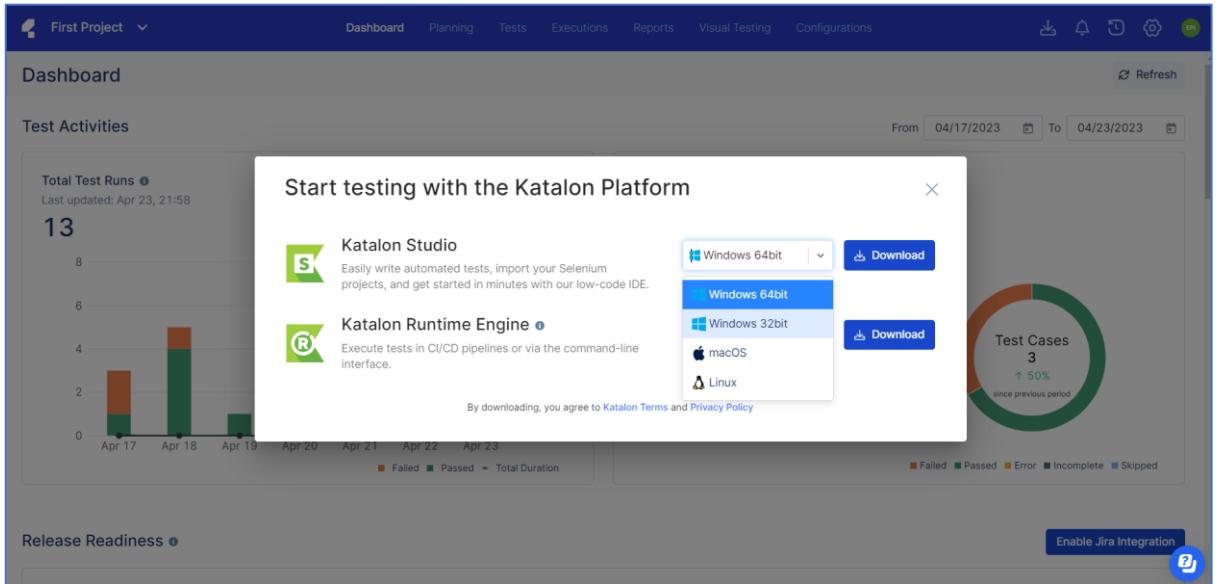
Bước 1: Truy cập link và download Katalon Studio tại: katalon.com

Trước khi download cần tạo một tài khoản trên trang chủ của Katalon để có thể tải và sử dụng miễn phí phần mềm.



Hình 3. 1: Giao diện tạo tài khoản đăng ký trên Katalon Studio

Có 4 phiên bản để download: Windows 64, Windows 32, macOS, Linux 64. Tùy vào hệ điều hành của máy bạn cài mà có thể tải đúng phiên bản cho hệ điều hành.



Hình 3. 2: Download phiên bản Katalon Studio miễn phí

Bước 2: Sau khi download về máy, tiến hành giải nén và mở ứng dụng trên máy tính. Để khởi động Katalon Studio, nhấn đúp vào katalon.exe

This PC > Data (D) > ĐATN > Katalon_Studio_Windows_64-8.6.0				
	Name	Date modified	Type	Size
(G:)	configuration	23/03/10 3:24 PM	File folder	
	features	23/03/10 3:24 PM	File folder	
	jre	23/03/10 3:24 PM	File folder	
	p2	23/03/10 3:23 PM	File folder	
	plugins	23/03/10 3:24 PM	File folder	
	artifacts.xml	23/03/10 3:24 PM	XML Document	87 KB
	eclipse.exe	23/03/10 3:23 PM	Application	128 KB
	katalon.exe	23/03/10 4:18 PM	Application	419 KB
(E:)	katalon.ini	23/03/10 3:24 PM	Configuration setti...	1 KB
(G:)				

Hình 3. 3: Cài đặt ứng dụng Katalon trên máy tính

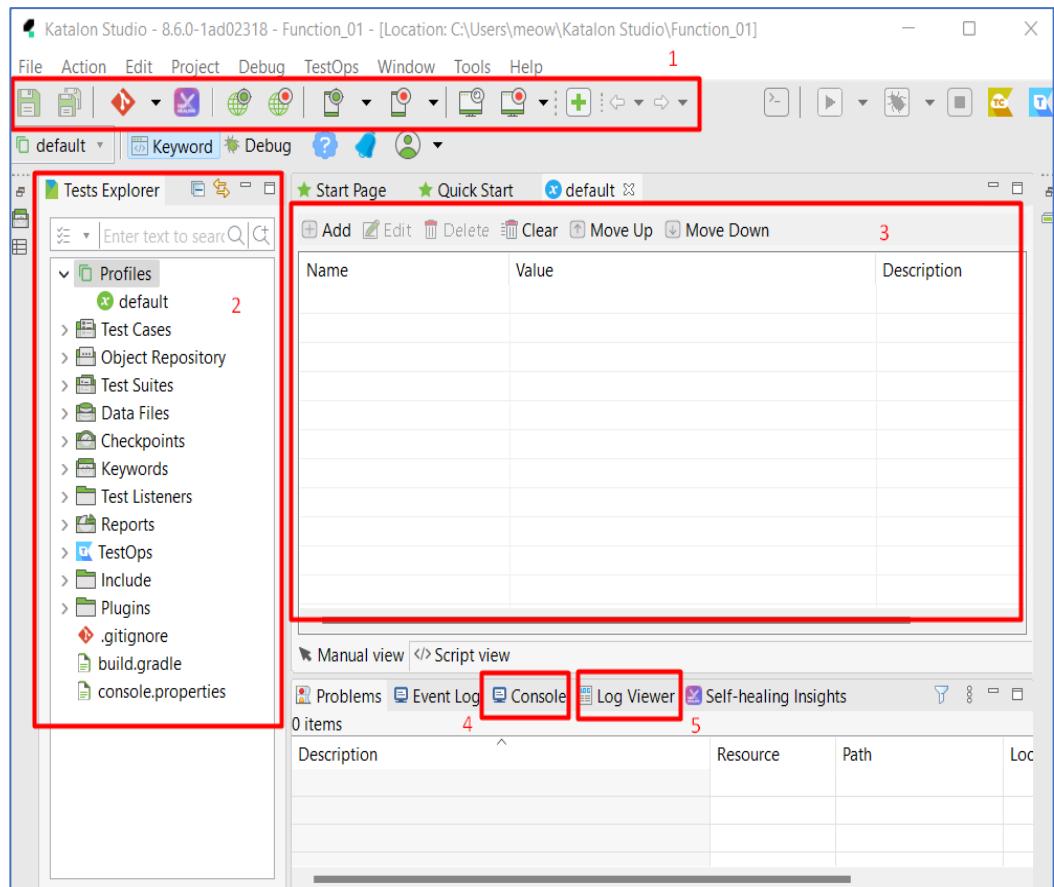
Bước 3: Ứng dụng sẽ được hiển thị màn hình như hình 3.4:



Hình 3. 4: Hình ảnh Katalon Studio khi khởi động

Ở lần hiển thị đầu tiên, cửa sổ kích hoạt Katalon Studio, nhằm mục đích kiểm tra tài khoản đang sử dụng là phiên bản enterprise, free hay đang trial. Ứng dụng sẽ yêu cầu nhập email và mật khẩu đã đăng ký.

Bước 4: Sau khi đăng nhập vào được giao diện chính, chọn New Project, sau khi điền thông tin cho Project của bạn, sẽ hiển thị giao diện như hình bên dưới



Hình 3. 5: Giao diện chính của Katalon Studio

Trong đó:

1. Thanh Toolbar: Chứa tất cả các tính năng quan trọng thường xuyên sử dụng để làm việc
2. Test Explorer: Các thư mục về cấu trúc dự án
3. Danh sách các bước trong Test case
- 4 và 5. Hiển thị các kết quả test, hỗ trợ chế độ Console và Log Viewer

3.5.2. Cấu hình

❖ Các yêu cầu hệ thống

	Yêu cầu
Hệ điều hành	Windows 7, Windows 8, Windows 10, macOS 10.11+, Linux (Ubuntu based)
CPU	Bộ xử lý 1 Ghz hoặc nhanh hơn 32 bit (x86) hoặc 64 bit (x64)

Bộ nhớ	Tối thiểu 1 GB RAM (32 bit) hoặc 4 GB RAM (64 bit). Khuyến nghị là 4 GB Ram (32 bit) và 8 GB RAM (64 bit)
Ổ cứng	Ít nhất 1 GB dung lượng ổ cứng khả dụng. Cần thêm dung lượng đĩa phụ thuộc vào mã nguồn dự án và các báo cáo thực hiện được tạo.

Bảng 3. 1: Các yêu cầu hệ thống khi cài đặt Katalon Studio

❖ Môi trường hỗ trợ: Trình duyệt

Trình duyệt	Version on Windows	Version on MacOS	Chú ý
Internet Explorer	9, 10, 11	N/A	Cấu hình IE bắt buộc: cấu hình Internet Explorer
Microsoft Edge	Hiện hành	N/A	Tham khảo trang này để biết trạng thái hiện tại của Edge WebDriver: https://learn.microsoft.com/en-us/archive/microsoft-edge/legacy/developer/
Firefox	56+	Để sử dụng Firefox 57 với Katalon Studio, vui lòng sử dụng Katalon	

		Studio v5.1+	
Google Chrome	58+		
Opera	Không hỗ trợ		
Safari	5.1+	9, 10, 11	

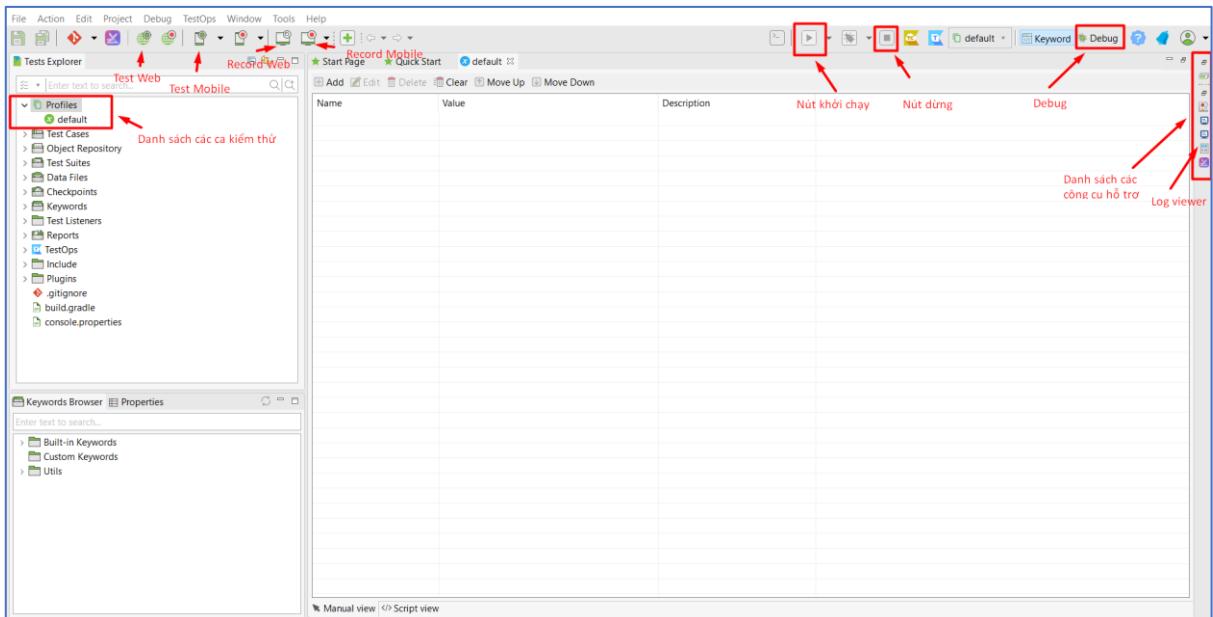
Bảng 3. 2: Danh sách trình duyệt hỗ trợ Katalon Studio

❖ Cấu hình Katalon Studio trên trình duyệt Chrome

Trong phạm vi đồ án môn học, nhóm tác giả chỉ hướng dẫn cách cấu hình của Katalon Studio với trình duyệt Chrome (vì Chrome được sử dụng phổ biến hơn các trình duyệt còn lại).

Cần cài đặt tiện ích mở rộng trình ghi tự động Katalon cho Chrome để có chụp các đối tượng trên trình duyệt web (Web Object Spy) và ghi - phát lại (Record & Playback) là: Katalon Automation Recorder

3.6. Một số chức năng trong Katalon Studio



Hình 3. 6: Giải thích một số chức năng trong Katalon Studio

Spy Web		Khởi tạo quá trình test manual trên web
		Ghi lại các thao tác trong ca kiểm thử trên Web
		Khởi động ca kiểm thử đang thực thi
		Tạm dừng ca kiểm thử đang thực thi
Debug		Sửa lỗi ca kiểm thử đang thực thi
Test Cases		Danh sách các ca kiểm thử
Test Suites		Tập hợp các test case
		Trình log Viewer
Add		Thêm một ca kiểm thử
Delete		Xóa một ca kiểm thử
		Thêm thuộc tính cho object
Textbox <i>Input</i>		Giá trị đầu vào cho thao tác trong ca kiểm thử
Textbox <i>Object</i>		Đối tượng đến của thao tác
Textbox <i>Item</i>		Các hành động (action) trong ca kiểm thử

Bảng 3. 3: Một số chức năng trong Katalon Studio

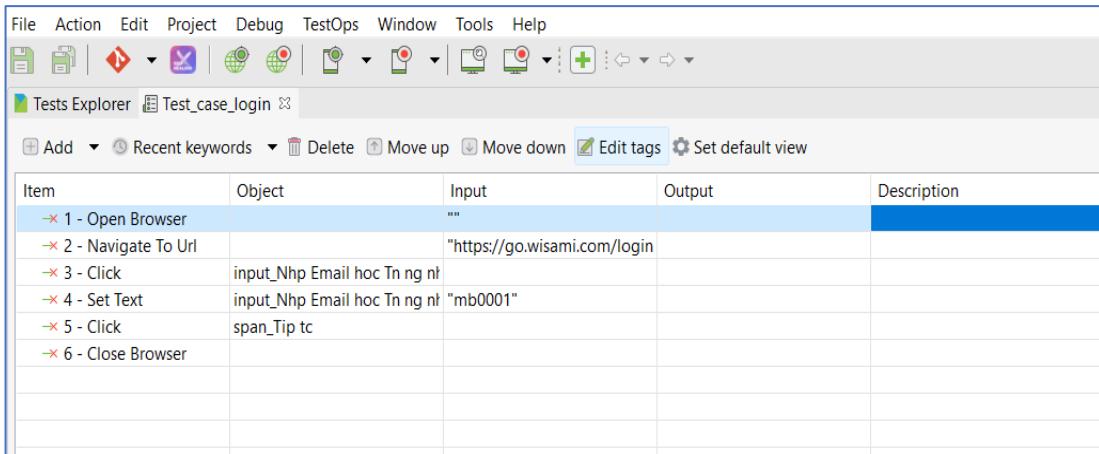
Test Explorer cho phép bạn duyệt qua cấu trúc của dự án và truy cập vào tất cả các thao tác kiểm thử của bạn. Sử dụng menu ngữ cảnh trên chế độ xem, bạn có thể tạo các thao tác mới, sắp xếp các mục của chế độ xem hoặc kéo thả chúng vào các chế độ xem trình chỉnh sửa nhất định nếu cần.

Group	Mô tả

Profile	Liệt kê tất cả các hồ sơ thực thi của dự án hiện tại
Test cases	Liệt kê tất cả các trường hợp kiểm thử trong dự án hiện tại
Object repository	Liệt kê tất cả các đối tượng kiểm thử của dự án hiện tại
Test suites	Liệt kê tất cả các bộ kiểm thử và bộ sưu tập kiểm thử của dự án hiện tại
Data files	Liệt kê tất cả dữ liệu thử nghiệm của dự án hiện tại
Checkpoints	Liệt kê tất cả các điểm kiểm tra của dự án hiện tại
Keywords	Liệt kê tất cả các từ khóa tùy chỉnh của dự án hiện tại
Test listeners	Liệt kê tất cả các trình nghe kiểm thử của dự án hiện tại
Reports	Liệt kê tất cả các báo cáo đã tạo của dự án hiện tại
TestOps	Chứa tất cả các lần chạy thử nghiệm từ máy chủ TestOps
Include	Chứa tệp tính năng Cucumber và định nghĩa bước
Plugin	Chứa tất cả các tệp plugin của dự án hiện tại

Bảng 3. 4: Bảng chi tiết các chức năng trong Test Explorer

Theo mặc định, Tests Explorer hiển thị tất cả các thao tác kiểm thử. Từ phiên bản 7.0.0 trở đi, bạn có thể tùy chỉnh Test Explorer bằng cách chọn Project -> Setting -> Explorer. Bỏ chọn các phần mềm kiểm thử mà bạn muốn ẩn, sau đó vào Apply và Close.



Hình 3. 7: Giao diện test case trong Katalon Studio

Trường hợp kiểm thử là một tập hợp các hành động được thực thi để xác minh một tính năng hoặc chức năng cụ thể của ứng dụng phần mềm của bạn. Khi bạn mở một trường hợp kiểm thử, trình chỉnh sửa trường hợp kiểm thử chứa thông tin chi tiết của trường hợp kiểm thử đó trong các tab:

- Tab Manual: Hiển thị chế độ xem thủ công, nơi cấu hình cơ bản theo hướng từ khóa cho phép bạn tạo các kiểm thử tự động mà không cần viết mã
- Tab Script: Hiển thị dạng xem tập lệnh, nơi người dùng nâng cao có nền tảng lập trình có thể sửa đổi tập lệnh bằng ngôn ngữ Groovy hoặc Java
- Tab Variables: Hiển thị tất cả các biến được xác định cho trường hợp thử nghiệm đó
 - Tab Variable (Script mode): Hiển thị tất cả các biến được xác định cho trường hợp kiểm thử đó ở chế độ tập lệnh
 - Data Binding: Cho phép người dùng tiến hành ràng buộc dữ liệu ở cấp trường hợp kiểm thử
 - Tab Integration: Hiển thị tích hợp được định cấu hình của người dùng trong dự án, ví dụ: qTest, Jira, Azure DevOps...
 - Tab Properties: Hiển thị thông tin chung về trường hợp kiểm thử, bao gồm Mô tả và Nhận Xét

Một số lệnh cơ bản trong Katalon Studio:

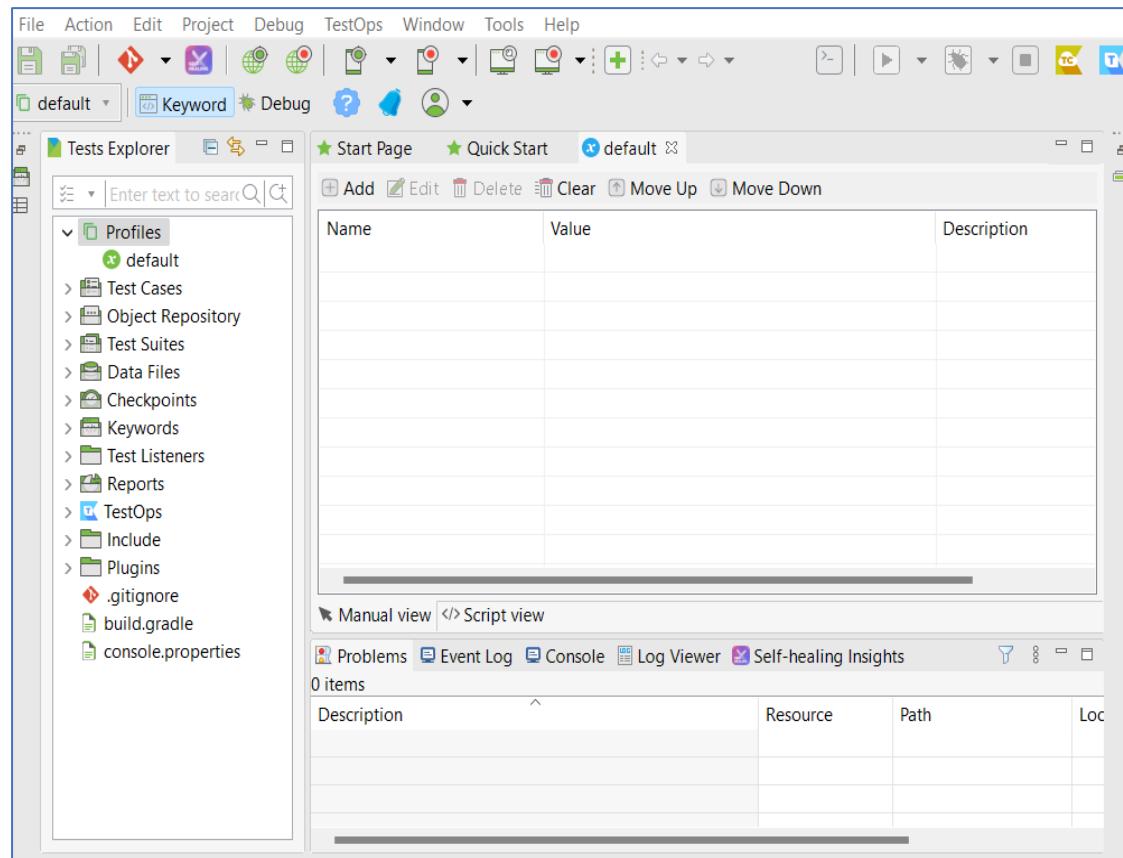
- WebUI.openBrowser (URL): Mở trình duyệt
- WebUI.navigateToURL (URL): Điều hướng trang Web

- WebUI.click (object): Click vào object
- WebUI.setText (object, text): Điền text vào object
- WebUI.getText (object): Lấy ra giá trị text của object
- findTestObject (id): Tìm Object theo id đã lưu trong Katalon
- WebUI.uploadFile (object, filePath): Upload file vào object dựa theo filePath
- WebUI.callTestCase(findTestCase("id"), [:], FailureHandling.STOP_ON_FAILURE): Gọi testcase đã viết dừng nếu testcase fail
- WebUI.verifyElementVisible(object, FailureHandling.STOP_ON_FAILURE): Kiểm tra xem Object có hiển thị hay không

3.7. Cách viết một kịch bản với Katalon Studio

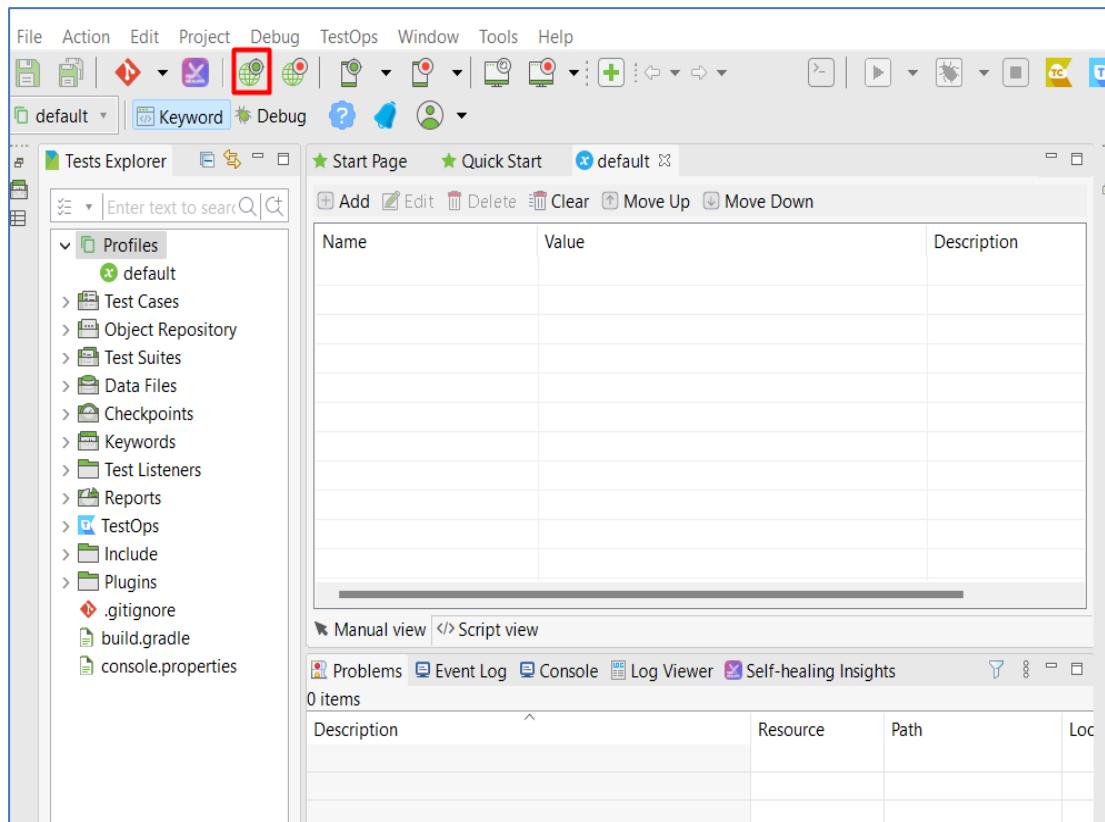
3.7.1. Cách chụp đối tượng trên màn hình (Lấy ID đối tượng)

Bước 1: Mở Katalon Studio



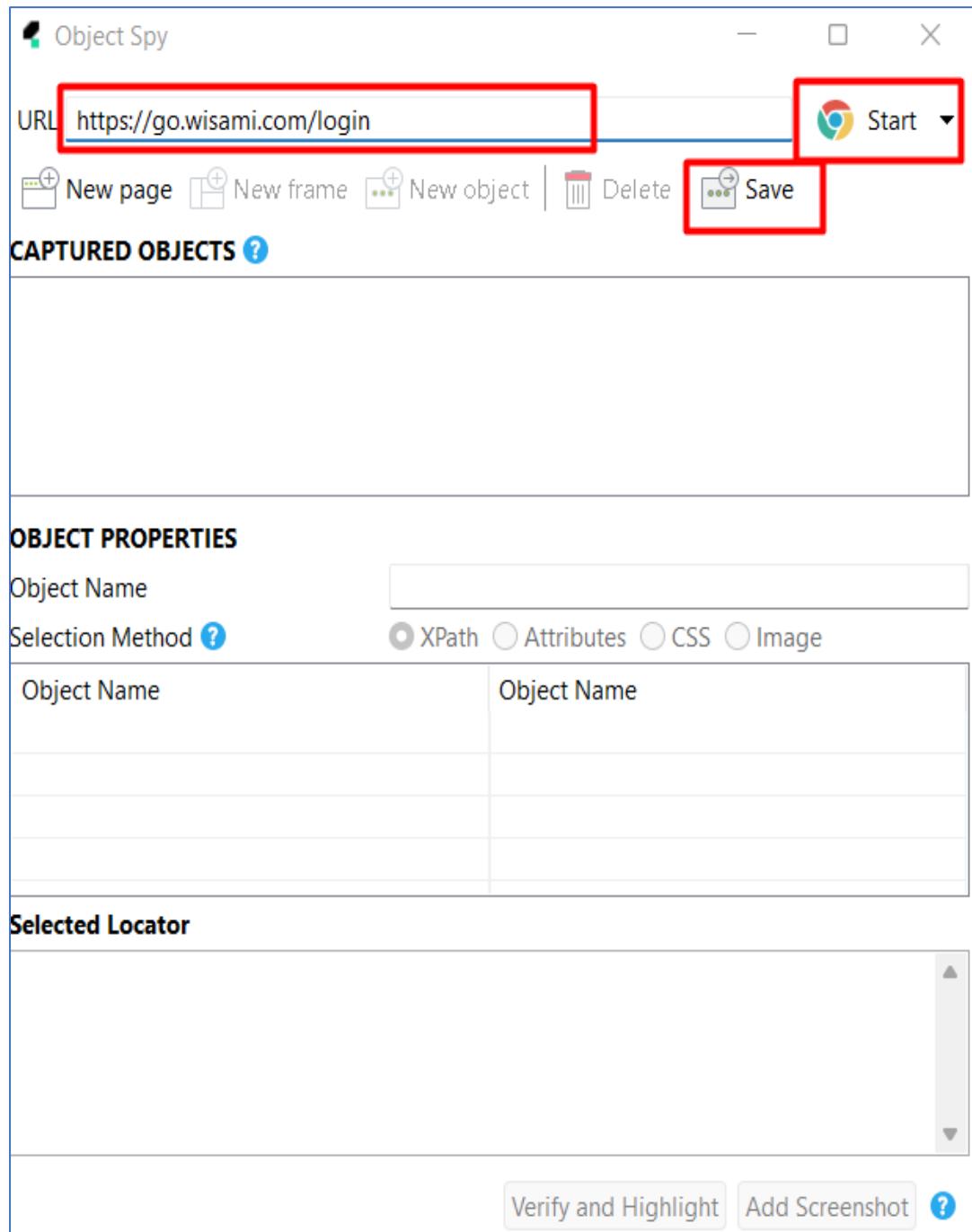
Hình 3. 8: Giao diện khởi chạy Katalon Studio

Bước 2: Chọn biểu tượng Spy Web



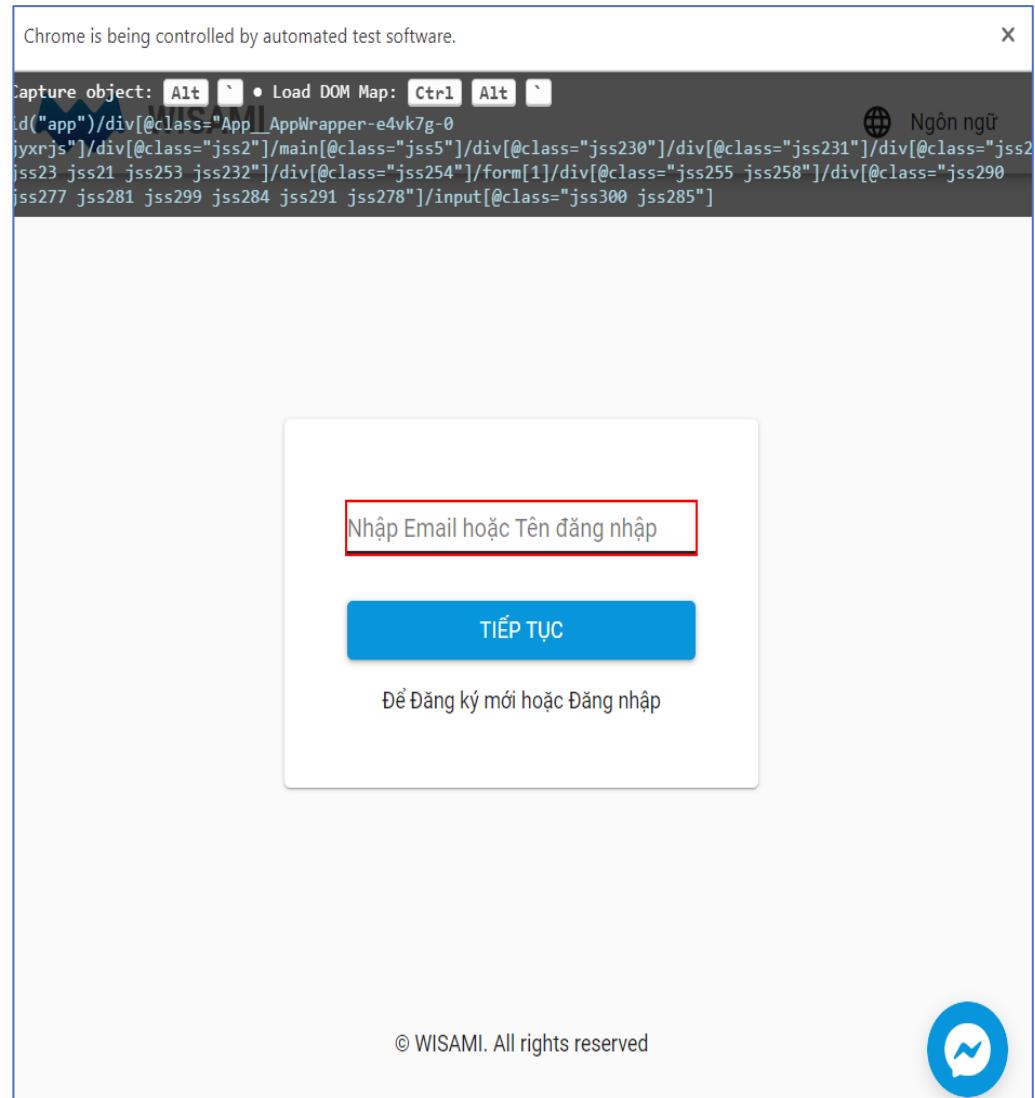
Hình 3. 9: Giao diện chọn Skype Web

Bước 3: Thêm URL của website cần test



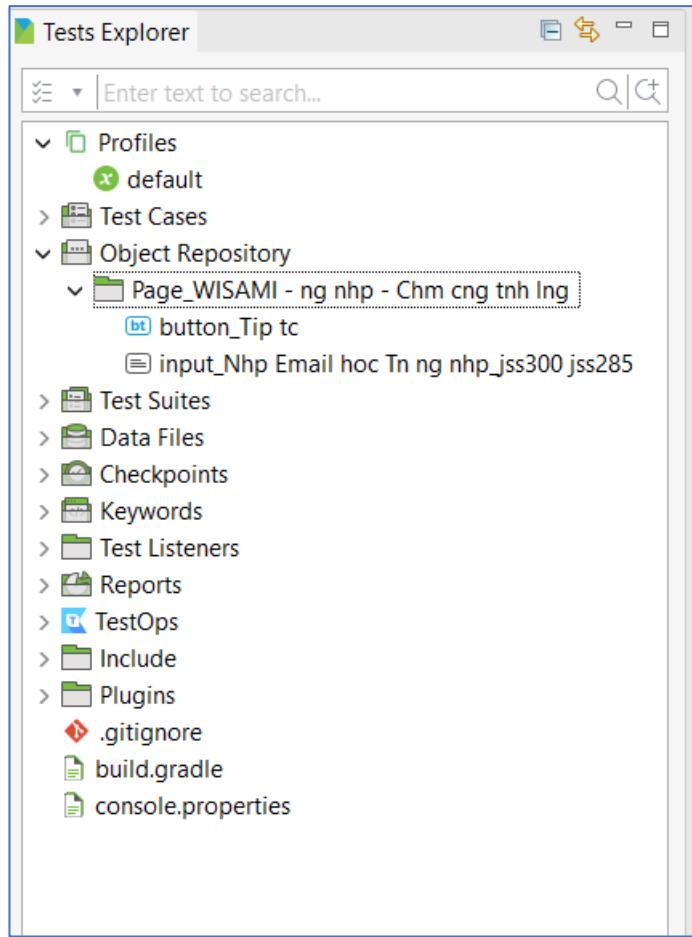
Hình 3. 10: Thêm URL của website cần test

Thử thực nghiệm với trang web nào thì dán liên kết của trang web đó vào ô URL, như trong hình là web thực hiện build trên local. Sau đó click Start trên trình duyệt Chrome. Katalon sẽ điều hướng đến trang web mà bạn điền link. Lúc này bạn có thể thực hiện chụp các đối tượng trên màn hình. Chụp các đối tượng bằng cách chỉ con trỏ chuột vào đối tượng đó rồi nhấn tổ hợp phím alt + ~. Hình 3.11:



Hình 3. 11: Sử dụng phím tắt để bắt đối tượng

Sau khi chụp các đối tượng, ID của các đối tượng này sẽ được lưu trong Object Repository, ân Lưu bằng cách chọn nút Save.



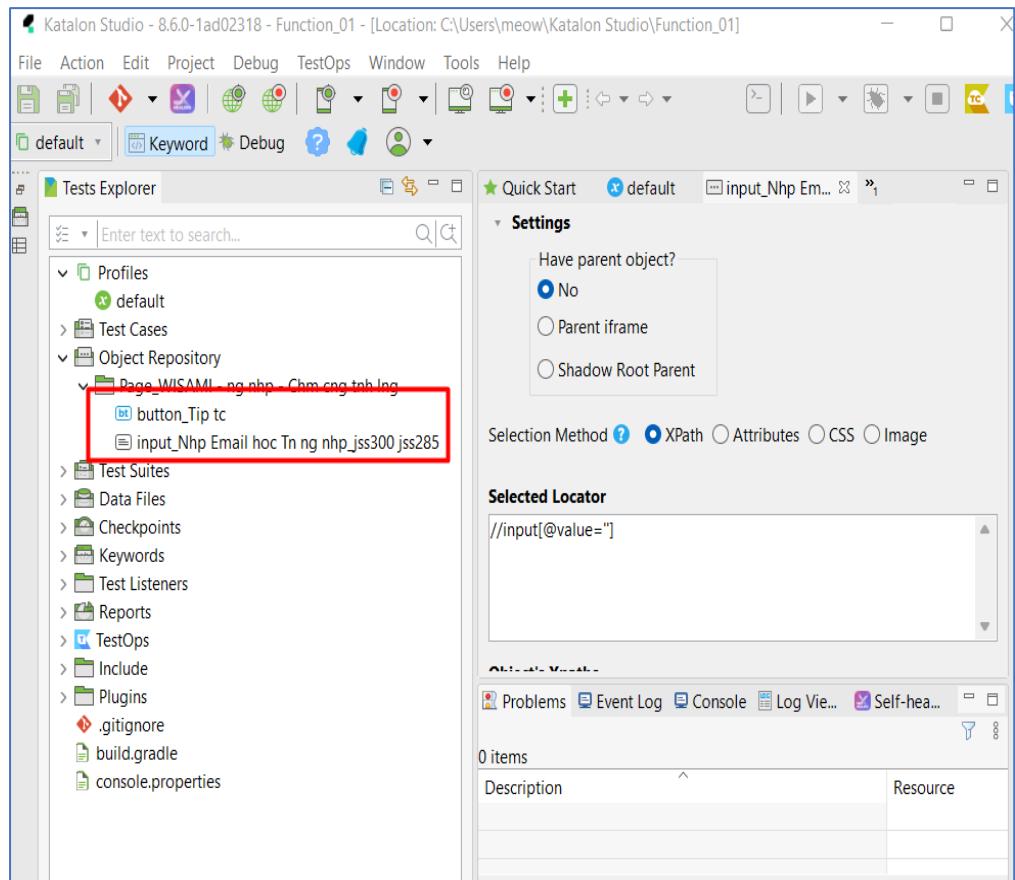
Hình 3. 12: Lưu kết quả sau khi bắt đối tượng trong Katalon Studio

3.7.2. Viết kịch bản test với Katalon Studio

Katalon Studio hỗ trợ người dùng 2 chế độ để thiết lập kịch bản test:

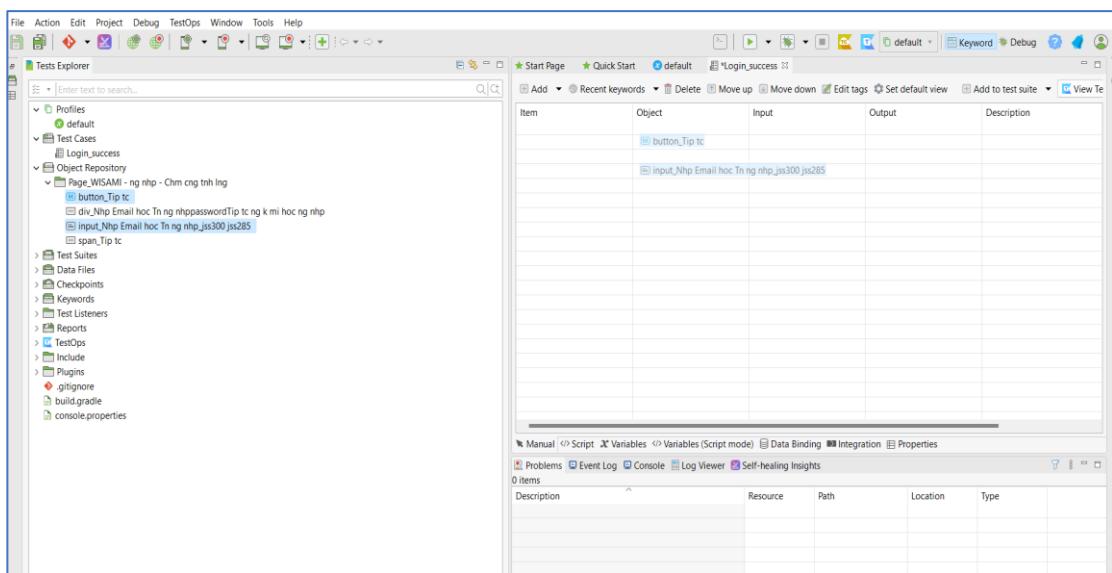
Manual view và Script view:

- Manual View: Ở chế độ manual view này, click vào các đối tượng trong mục Object Repository



Hình 3. 13: Lựa chọn đối tượng trong Object Repository

Thả các ID vào mục Object

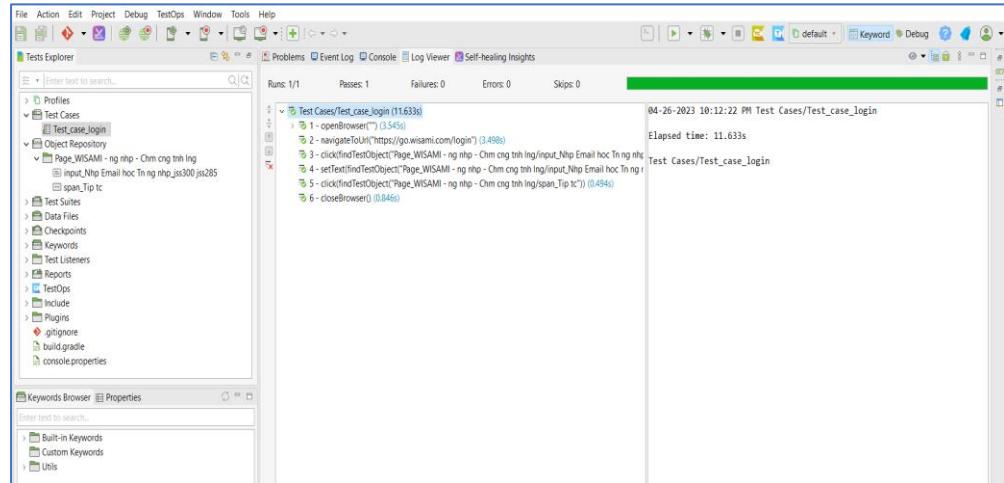


Hình 3. 14: Kéo thả ID vào mục Object

➤ Script View:

- Ở chế độ này, Katalon Studio cho phép nhập các câu lệnh để thực hiện chạy kịch bản test.

- Đầu tiên, cần bật trình duyệt với câu lệnh là: openBrowser
- Tiếp theo thực hiện các bước như hình và chạy Test case



Hình 3. 15: Kết quả test case nhập tài khoản đã đăng nhập

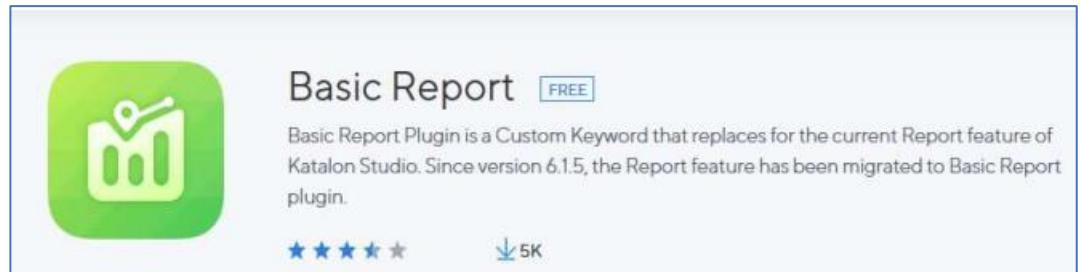
3.7.3. Một số plugin hỗ trợ kiểm thử ứng dụng Web

3.7.3.1. Basic Report

Basic Report là một Custom Keyword thay thế cho tính năng report hiện tại của Katalon Studio. Kể từ phiên bản 6.1.5, tính năng Báo cáo đã được di chuyển sang plugin Báo cáo cơ bản. Người dùng cần cài đặt plugin này để tiếp tục sử dụng tính năng này.

Basic Report cung cấp:

- Tạo báo cáo tự động từ báo cáo Test Suite sau mỗi lần thực hiện thử nghiệm với các định dạng khác nhau: HTML, CSV, JUnit và PDF.
- Cho phép người dùng xuất thủ công báo cáo bộ sưu tập Test Suite và Test Suite sang HTML, CSV, JUnit và PDF. Nhấp chuột phải vào báo cáo và chọn xuất dưới dạng bất kỳ



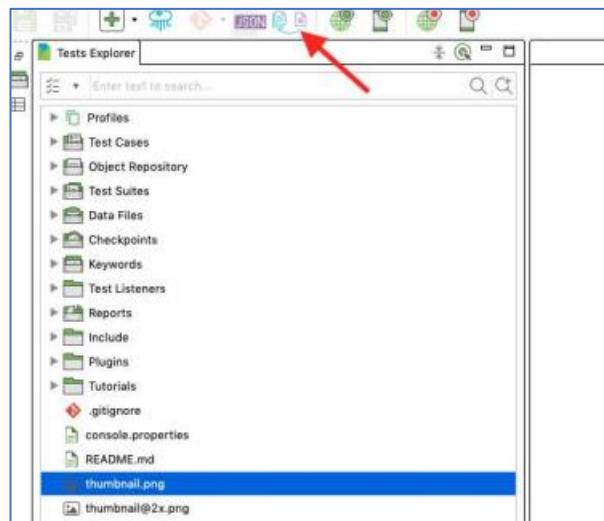
Hình 3. 16: Basic Report

3.7.3.2. Text Encoder for Katalon Studio

Text Encoder for Katalon Studio cho phép bạn mã hóa / giải mã văn bản bằng các thuật toán khác nhau. Bộ mã hóa bao gồm:

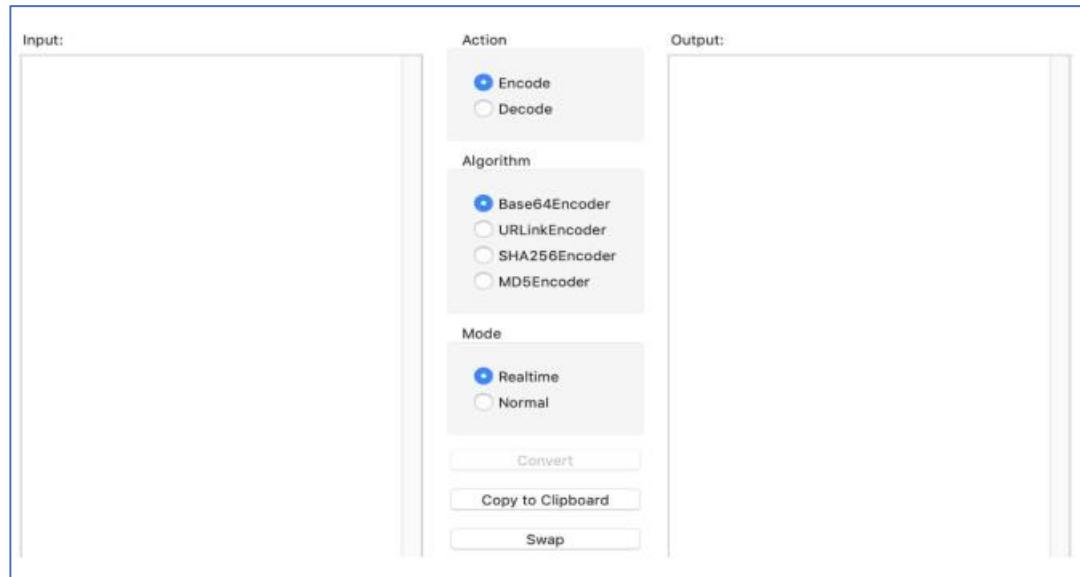
- Bộ giải mã Base64
- Bộ giải mã URLLinkEncoder
- Bộ giải mã SHA256

Sau khi tải xuống plugin, nhập vào biểu tượng Encoder hóa dưới thanh công cụ



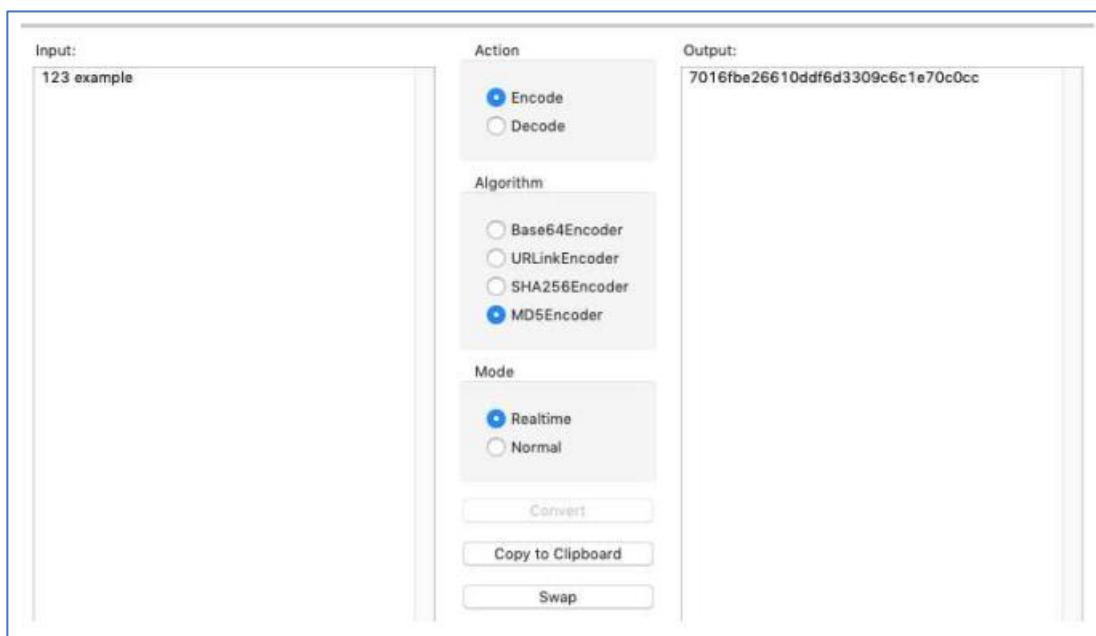
Hình 3. 17: Text Encoder

Sẽ có một cửa sổ bật lên hiển thị Mã hóa văn bản



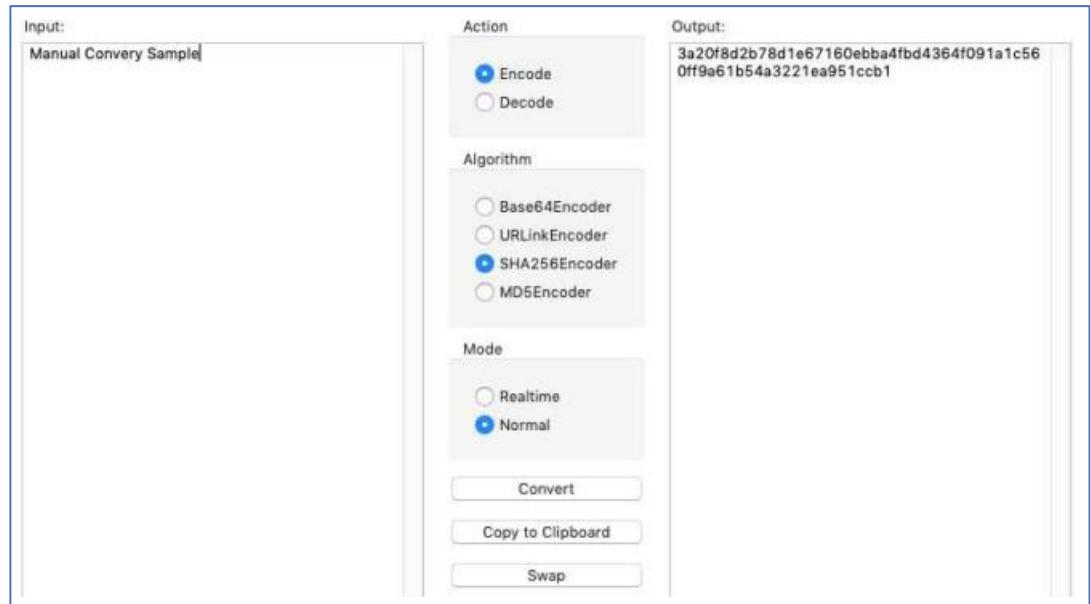
Hình 3. 18: Text Encoder Pop up

Trên chế độ 'Thời gian thực', Đầu ra sẽ được tạo tự động sau khi bạn nhập văn bản mong muốn



Hình 3. 19: Text Encoder Realtime mode

Trên chế độ 'Bình thường', để mã hóa hoặc giải mã văn bản, bạn phải nhập vào biểu tượng 'Chuyển đổi'



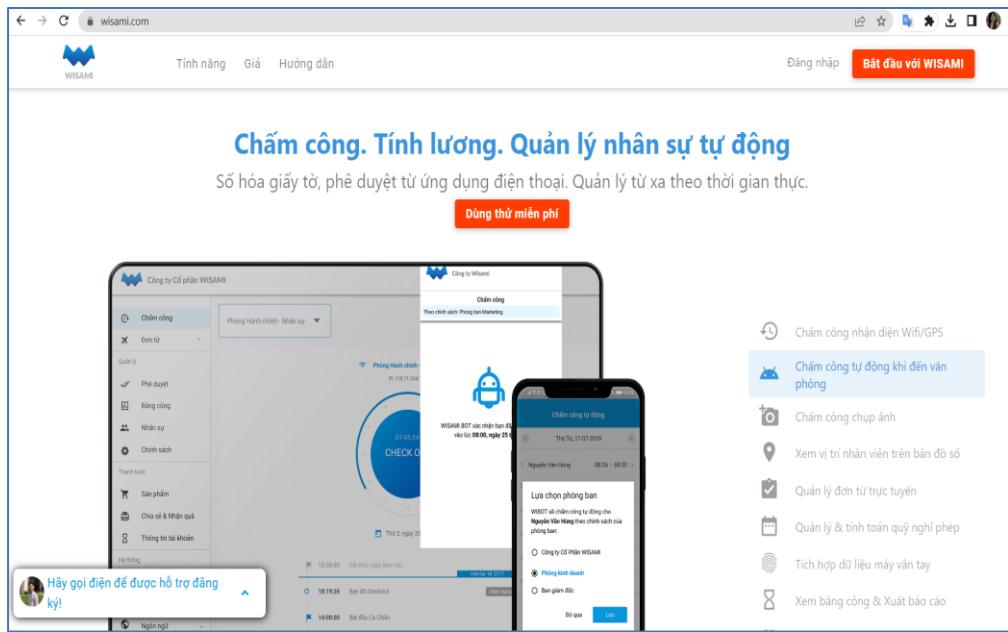
Hình 3. 20: Text Encoder Normal mode

Ngoài ra còn nhiều plugin hỗ trợ trong Katalon Studio như:

- Excel Keywords: viết kiểm tra tự động hóa bằng cách sử dụng tệp excel (.xls và .xlsx) dễ dàng hơn nhiều bằng cách đọc, viết và so sánh các tệp Excel.
- Read and Write Gmail Messages Keywords: kết nối tài khoản email và thao tác nội dung thư theo yêu cầu kiểm tra
- Get Authorization Value Form API Keywords: Plugin này rất hữu ích để nhận giá trị Ủy quyền từ API
- Jira Integration: Đóng bộ hóa các trường hợp kiểm tra và nội dung Gherkin của BDD từ các vấn đề của Jira. Tự động tạo các trường hợp thử nghiệm thông qua Ngôn ngữ truy vấn Jira (JQL) trong Katalon Studio.
- Slack Integration: plugin này hiện đang hỗ trợ cả kênh riêng và kênh công cộng.

3.8. Ứng dụng Katalon Studio kiểm thử Website Wisami Go

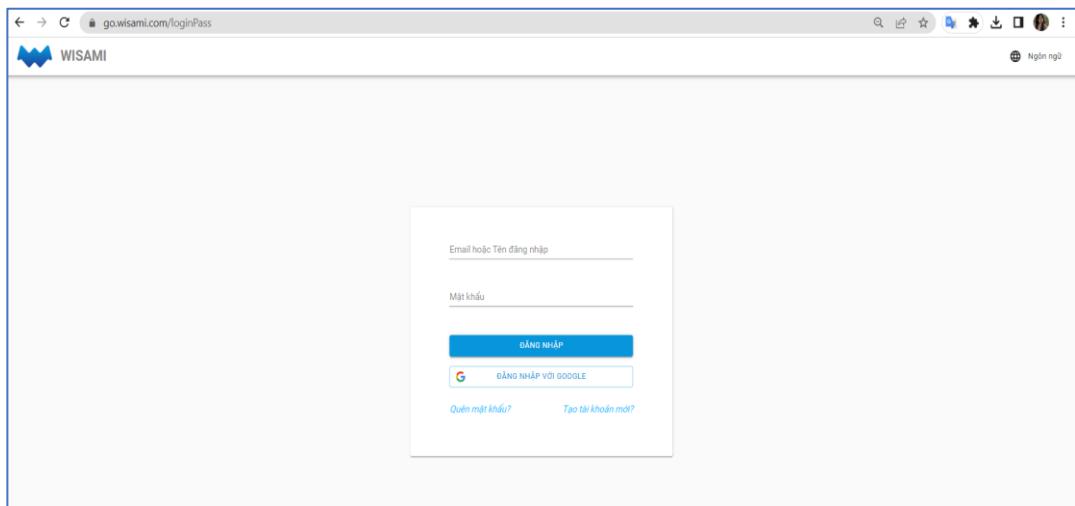
3.8.1. Giới thiệu Website chấm công Wisami Go



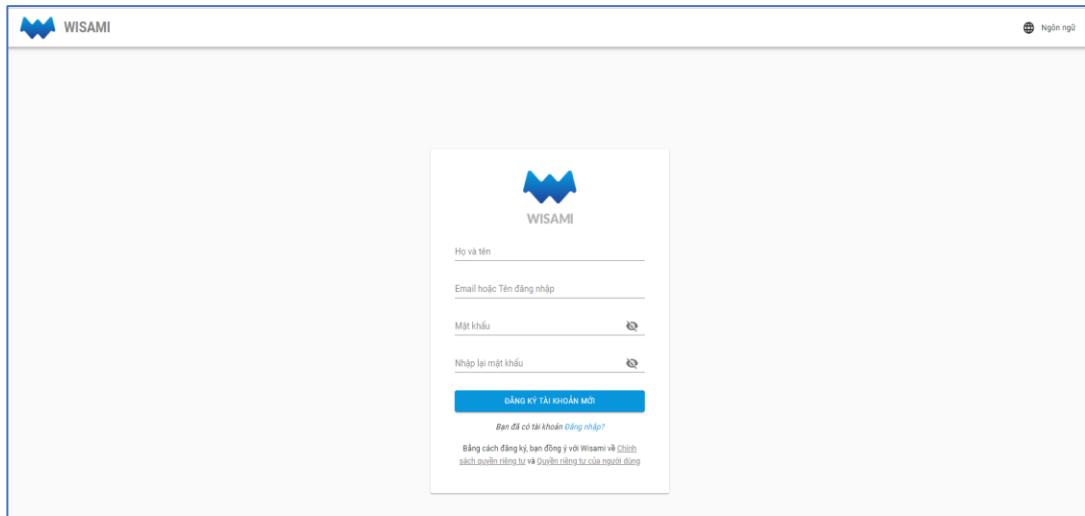
Hình 3. 21: Giao diện trang chủ Website chấm công Wisami Go

WISAMI là phần mềm Chấm công, tính lương cho Cửa hàng, Nhà hàng, Công ty văn phòng, Nhà máy khu công nghiệp.

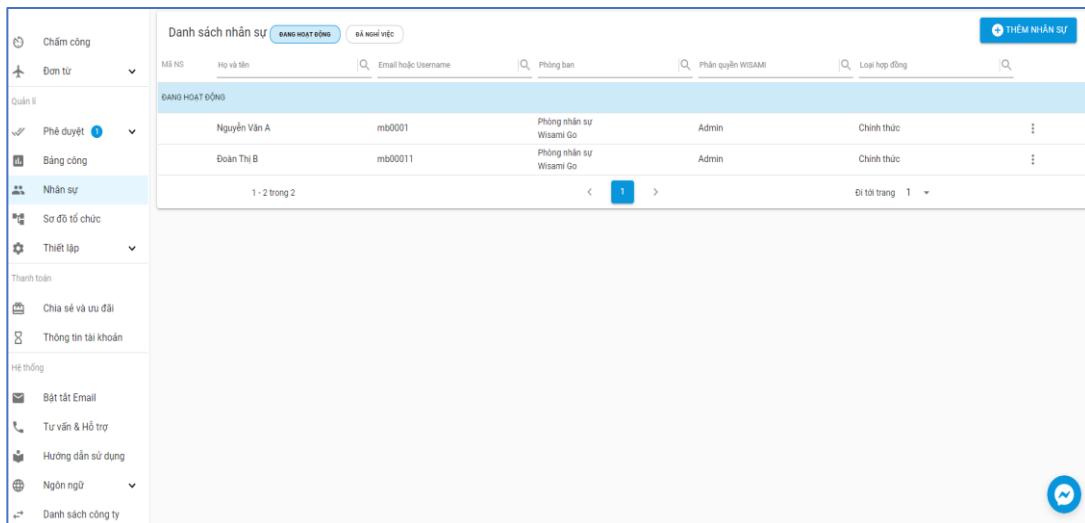
Giao diện đăng nhập, đăng ký của phần mềm:



Hình 3. 22: Giao diện đăng nhập website Wisami Go



Hình 3. 23: Giao diện đăng ký website Wisami Go



Hình 3. 24: Giao diện sau khi đăng nhập

3.8.2. Kiểm thử chức năng đăng ký, đăng nhập tài khoản công ty

Thông tin đăng ký, đăng nhập website:

- Để đăng ký tài khoản (admin công ty mới), người dùng được yêu cầu nhập các thông tin bao gồm: Họ và tên, Email (Tên đăng nhập), Mật khẩu và Nhập lại mật khẩu.
- Để đăng nhập, người dùng cần phải nhập thông tin bao gồm: Tên đăng nhập và mật khẩu đã đăng ký

Có 2 loại người dùng:

- Người dùng với vai trò nhân viên: có thể xem thông tin cá nhân trong tài khoản của mình

- Người dùng với vai trò quản trị viên (admin): có thể quản lý nhân viên, quản lý chấm công, phê duyệt, ...

Ứng dụng các kỹ thuật kiểm thử đoán lỗi kiểm tra các trường hợp:

- Người dùng đăng ký tài khoản thiếu thông tin
- Người dùng đăng ký tài khoản thành công
- Người dùng đăng nhập tài khoản thiếu thông tin
- Người dùng đăng nhập tài khoản thành công

3.8.2.1. Ca kiểm thử người dùng đăng ký

Test case:

Testcase ID	Testcase Description	Steps To Perform	Expect Result	Actual Result
Đăng_ký_01	Kiểm tra trường hợp đăng ký nhập thiếu trường “Họ và tên”	1. Khởi động trình duyệt 2. Tại trang đăng ký, nhập các thông tin: - Email hoặc tên đăng nhập: guyendt0612@gmail.com - Mật khẩu: 12345678 - Nhập lại mật khẩu: 12345678 3. Click button “Đăng ký tài khoản mới”	Đăng ký không thành công	Pass
Đăng_ký_02	Kiểm tra trường hợp đăng ký nhập thiếu	1. Khởi động trình duyệt 2. Tại trang đăng ký, nhập các thông tin:	Đăng ký không	Pass

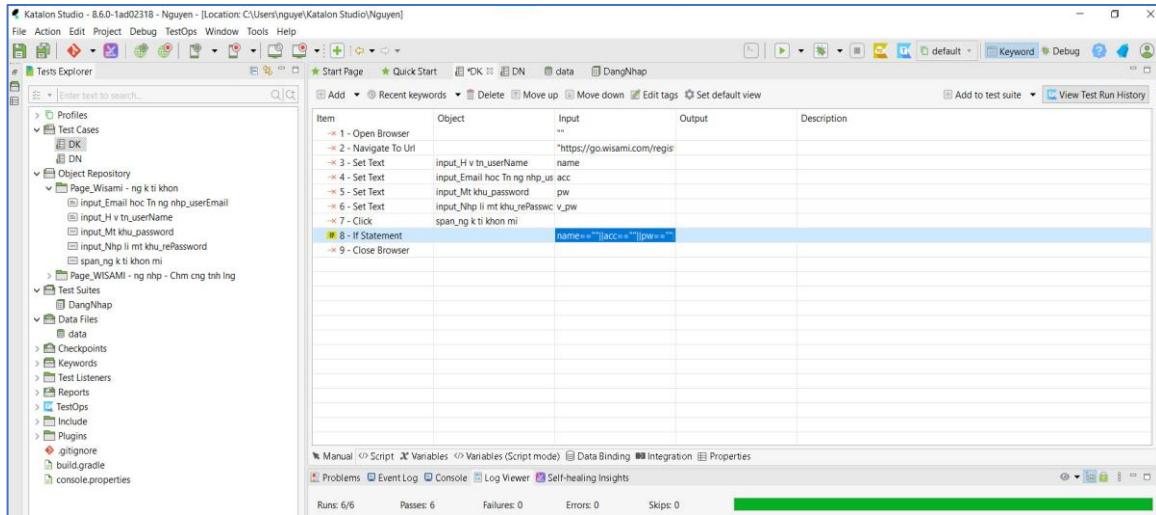
	“Email hoặc Tên đăng nhập”	<ul style="list-style-type: none"> - Họ và tên: Đặng Thị Nguyên - Mật khẩu: 12345678 - Nhập lại mật khẩu: 12345678 3. Click button “Đăng ký tài khoản mới” 	thành công	
Đăng_ký_03	Kiểm tra trường hợp đăng ký nhập thiếu “Mật khẩu”	<ul style="list-style-type: none"> 1. Khởi động trình duyệt 2. Tại trang đăng ký, nhập các thông tin: <ul style="list-style-type: none"> - Họ và tên: Đặng Thị Nguyên - Email và Tên đăng nhập: nguyendt0612@gmail.com - Nhập lại mật khẩu: 12345678 3. Click button “Đăng ký tài khoản mới” 	Đăng ký không thành công	Pass
Đăng_ký_04	Kiểm tra trường hợp đăng ký nhập thiếu “Nhập lại mật khẩu”	<ul style="list-style-type: none"> 1. Khởi động trình duyệt 2. Tại trang đăng ký, nhập các thông tin: <ul style="list-style-type: none"> - Họ và tên: Đặng Thị Nguyên - Email và Tên đăng nhập: nguyendt0612@gmail.com - Mật khẩu: 12345678 3. Click button “Đăng ký tài khoản mới” 	Đăng ký không thành công	Pass

Đăng_ký_05	Kiểm tra trường hợp nhập email đã đăng ký rồi	1. Khởi động trình duyệt 2. Tại trang đăng ký, nhập các thông tin: - Họ và tên: Đặng Thị Nguyên - Email và Tên đăng nhập: nguyen.nd102@gmail.com - Mật khẩu: 12345678 3. Click button “Đăng ký tài khoản mới”	Đăng ký không thành công	Pass
Đăng_ký_06	Kiểm tra trường hợp đăng ký thành công	1. Khởi động trình duyệt 2. Tại trang đăng ký, nhập các thông tin: - Họ và tên: Đặng Thị Nguyên - Email và Tên đăng nhập: nguyendt0612@gmail.com - Mật khẩu: 12345678 - Nhập lại mật khẩu: 12345678 3. Click button “Đăng ký tài khoản mới”	Đăng ký thành công	Pass

Bảng 3. 5: Bảng test case chìrc năng đăng ký

Tiến hành tạo Test case với Katalon Studio có các cách như tạo Test case từ Record Web, viết test case trên tab Manual, hoặc viết trên tab Script. Ở đây áp dụng tạo test case bằng cách Record Web, lấy được các thành phần của trang Web, và tiến hành chỉnh sửa trên tab Manual để hoàn chỉnh testcase cho phần Đăng ký

Testcase Test_Register được tiến hành tạo và với dữ liệu người dùng đăng ký được import từ file excel: data.xlsx.



Hình 3. 25: Test case đăng ký trên Katalon Studio

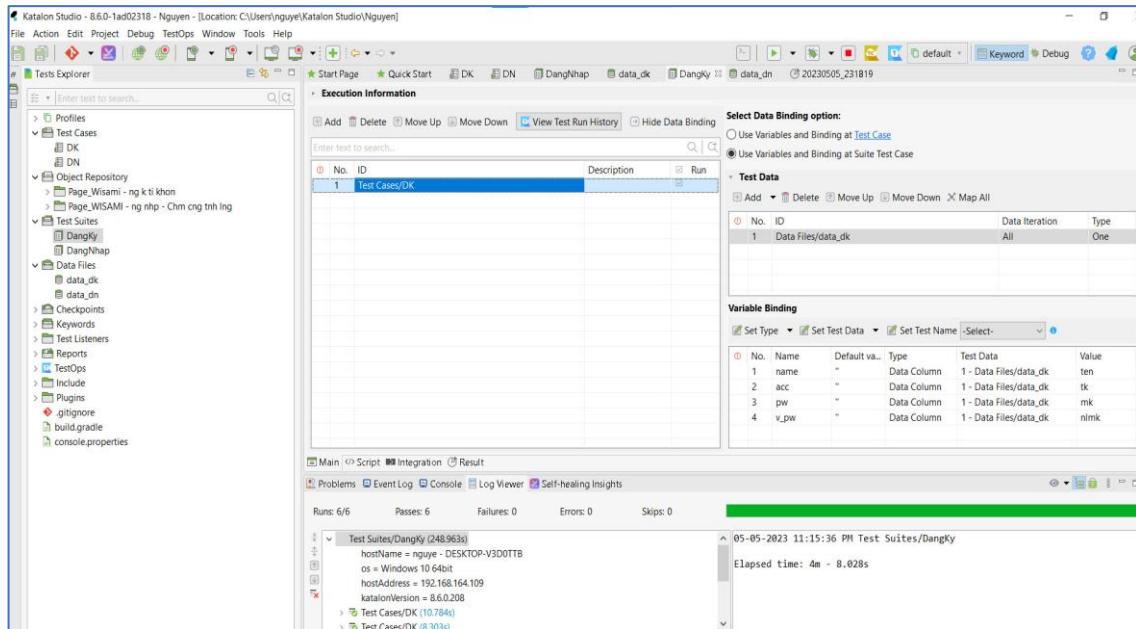
Để tiến hành import data, trên Test Explorer Click chuột phải vào Data Files -> New -> Nhập Name và chọn Data Type -> Click OK

Chọn file excel từ Browser tiến hành import, dữ liệu từ file excel sẽ được import lên Katalon Studio.

No.	ten	tk	mk	nimk
1		nguyendt0612@gmail.com	12345678	12345678
2	Dặng Thị Nguyễn	nguyendt0612@gmail.com	12345678	12345678
3	Dặng Thị Nguyễn	nguyendt0612@gmail.com		12345678
4	Dặng Thị Nguyễn	nguyendt0612@gmail.com	12345678	
5	Dặng Thị Nguyễn	nguyen.ndt02@gmail.com	12345678	12345678
6	Dặng Thị Nguyễn	nguyendt0612@gmail.com	12345678	12345678

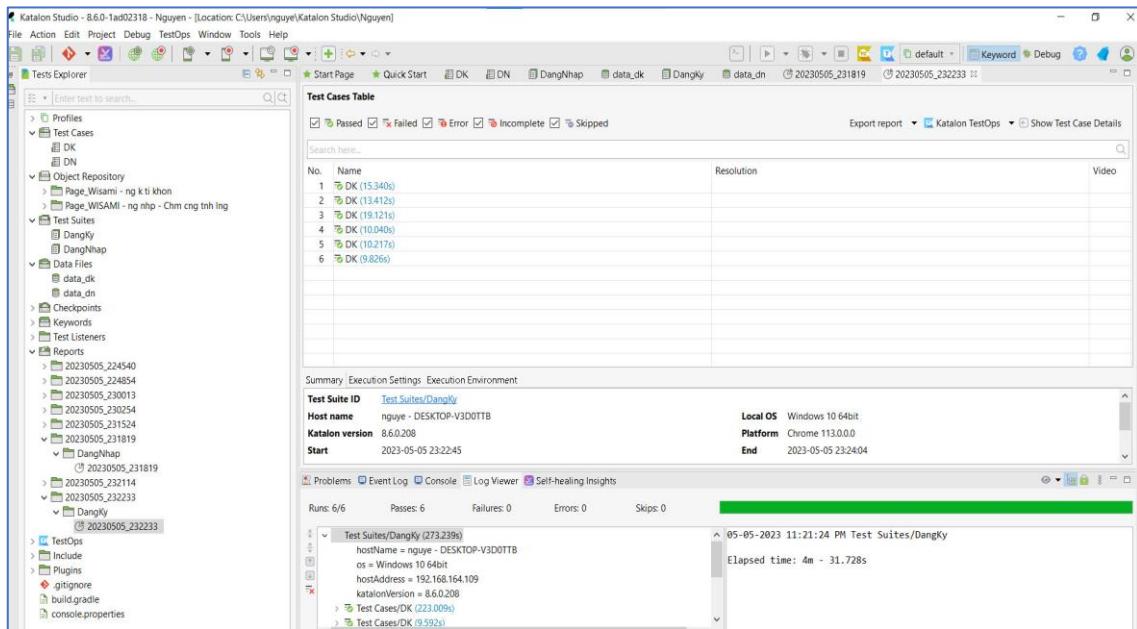
Hình 3. 26: Data test đăng ký được import

Tiếp đó tạo một test suite cho phần đăng ký này. Sau đó thêm mục test case đã tạo vào. Chọn Show Data Binding và thêm file excel cần import



Hình 3. 27: Test suite đăng ký

Sau khi import data từ file excel ta tiến hành chạy kiểm thử với công cụ ta thu được kết quả hình 3.28:



Hình 3. 28: Kết quả kiểm thử Test case đăng ký

Kết quả: Ca kiểm thử với Katalon Studio thực thi 6 trường hợp, pass 6. Trong trường hợp này do người kiểm thử đã viết test case bắt lỗi bằng key word “Verify Element Present”. Trong khi người dùng có bỏ trống thông tin nào sẽ không thể đăng ký tài khoản. Chức năng đăng ký tài khoản được hoạt động tốt.

Sau khi chạy kiểm thử thành công trên công cụ Katalon Studio, công cụ sẽ tích hợp báo cáo trên Reports, hỗ trợ báo cáo bằng các loại file HTML, CSV (Summary), CSV (Details), PDF. Với chức năng đăng ký em chọn báo cáo bằng HTML, hiển thị đầy đủ thông tin của các test case đã thực hiện. Qua báo cáo có thể thấy được Hostname thực hiện, thực hiện trên local nào, version Katalon Studio và Browser. Thời gian thực hiện các test case, kết quả của các test case.

3.8.2.2. Người dùng đăng nhập

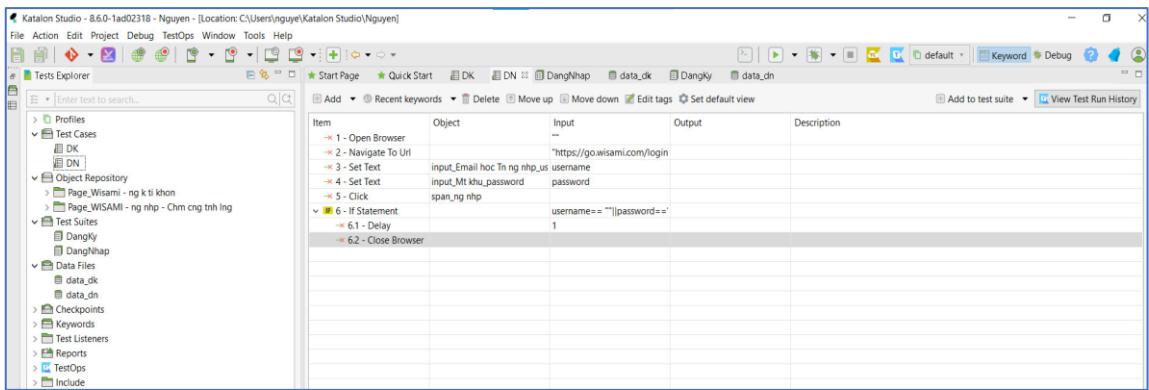
Test case:

Testcase ID	Testcase Description	Steps To Perform	Expect Result	Actual Result
Đăng_nhập_01	Kiểm tra trường hợp nhập thiếu “Email hoặc Tên đăng nhập”	1. Khởi động trình duyệt 2. Tại trang đăng nhập, nhập thông tin: - Mật khẩu 3. Click button “Đăng nhập”	Đăng nhập không thành công	Pass
Đăng_nhập_02	Kiểm tra trường hợp nhập thiếu “Mật khẩu”	1. Khởi động trình duyệt 2. Tại trang đăng nhập, nhập thông tin: - “Email hoặc Tên đăng nhập” 3. Click button “Đăng nhập”	Đăng nhập không thành công	Pass

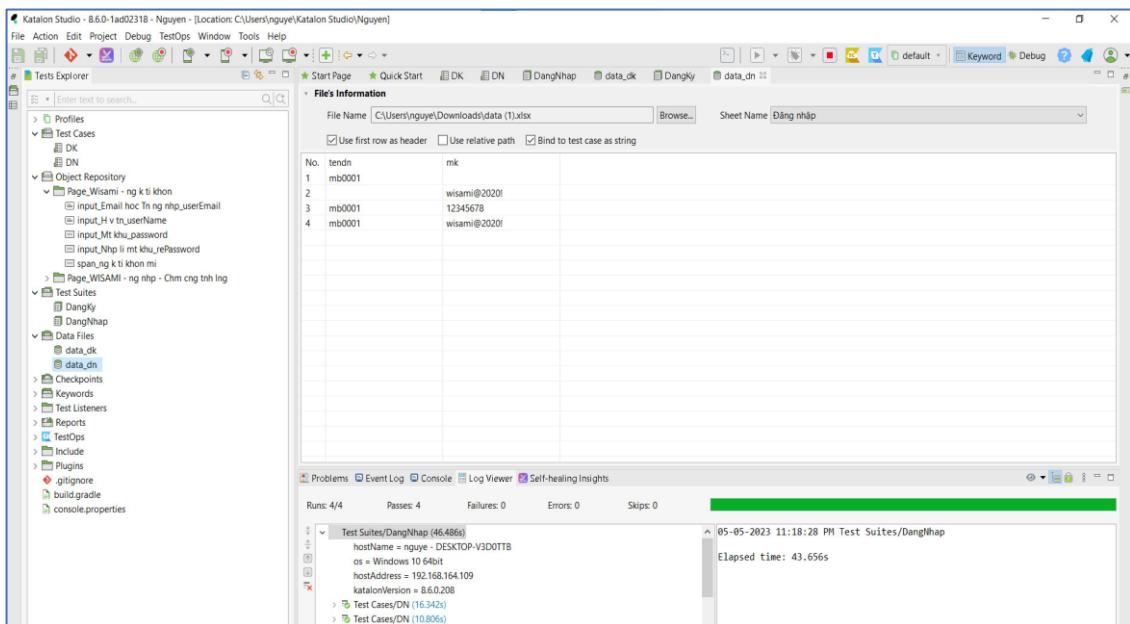
Đăng_nhập_03	Kiểm tra trường hợp đăng nhập bằng google	1. Khởi động trình duyệt 2. Tại trang đăng nhập, nhập thông tin: - “Email hoặc Tên đăng nhập” 3. Nhập sai “Mật khẩu”	Đăng nhập không thành công	Pass
Đăng_nhập_04	Kiểm tra trường hợp đăng nhập thành công	1. Khởi động trình duyệt 2. Tại trang đăng nhập, nhập thông tin: - Email hoặc Tên đăng nhập: mb0001 - Mật khẩu: wisami@2020! 3. Click button “Đăng nhập”	Đăng nhập thành công, hiển thị màn hình chọn công ty chấm công	Pass

Bảng 3. 6: Bảng test case chức năng đăng nhập

Tiến hành tạo Test Case với Katalon Studio với dữ liệu người dùng đăng ký được import từ file excel có tên là data.xlsx

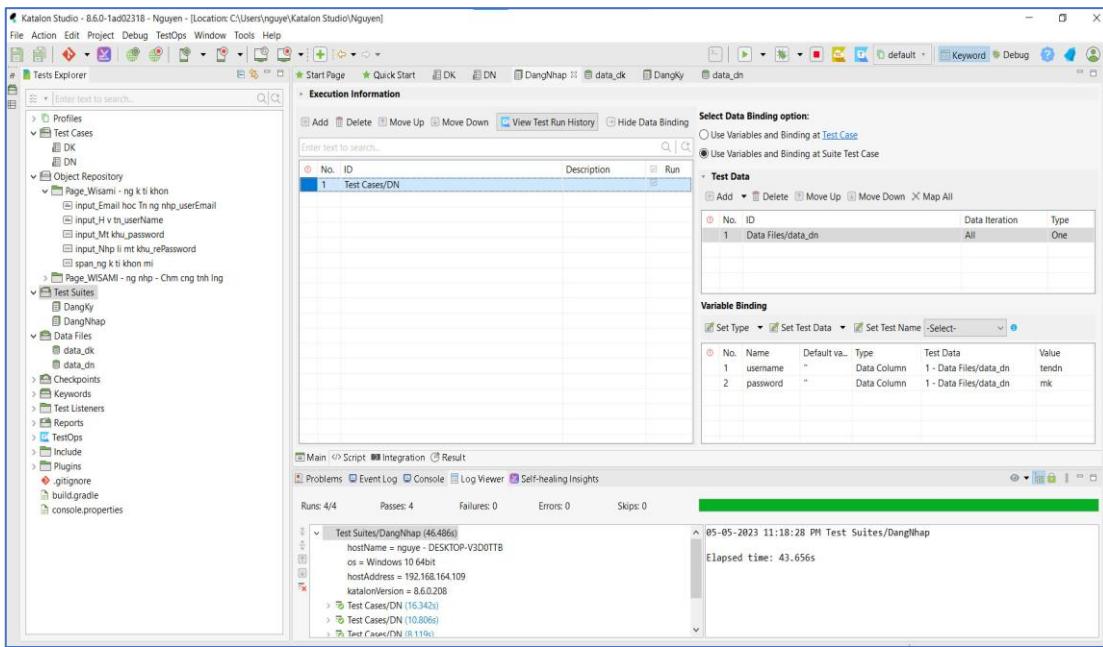


Hình 3. 29: Test case đăng nhập trên Katalon Studio



Hình 3. 30: Data đăng nhập

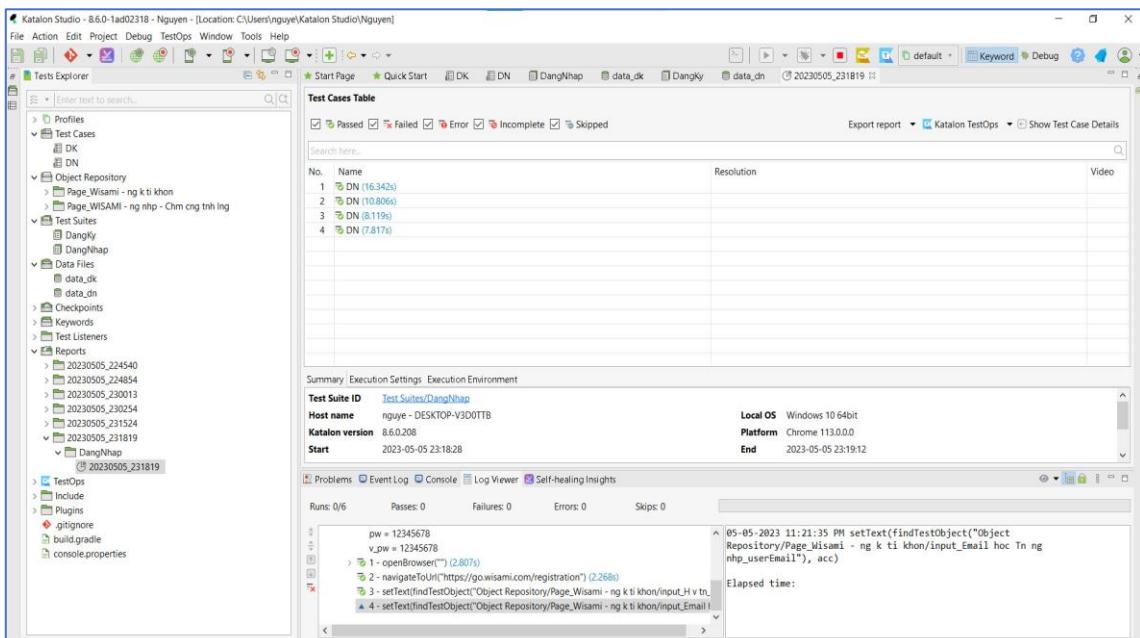
Sau khi import data từ file excel ta tiến hành chạy thử nghiệm Test suite với các trường hợp user trên, ta thu được kết quả hình 3.31:



Hình 3. 31: Test suite đăng nhập

Kết quả: Ca thử nghiệm với Katalon Studio thực thi 4 trường hợp, pass 4.

Công cụ kiểm tra khi người dùng nhập thông tin đăng nhập sai (mật khẩu), kết quả trả về đúng với test case mong muốn. Kiểm tra ca chức năng đăng nhập thành công.

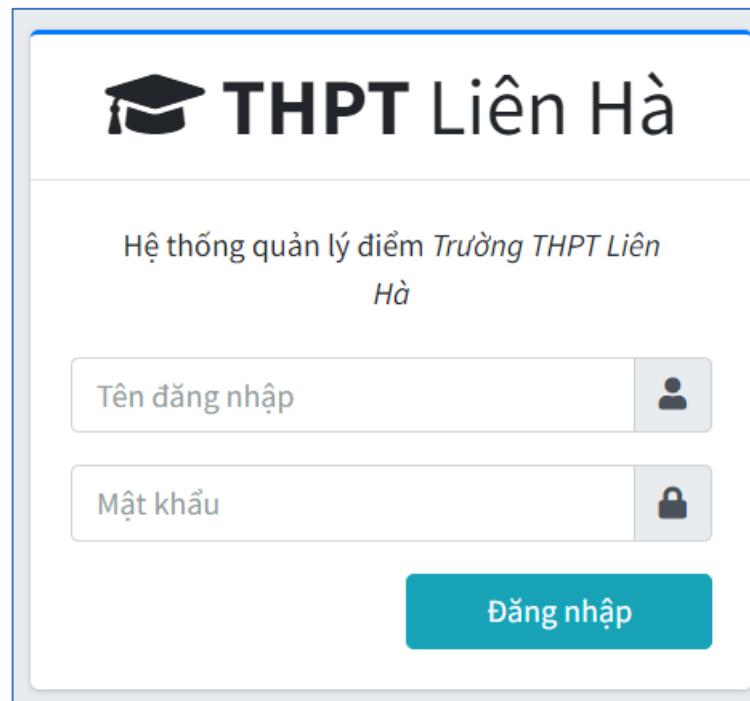


Hình 3. 32: Kết quả test chức năng đăng nhập

3.9. Ứng dụng kiểm thử website quản lý điểm trường THPT Liên Hà

3.9.1. Giới thiệu website

Giao diện đăng nhập, quản lý học sinh:



Hình 3. 33: Giao diện đăng nhập của admin

Mã	Họ tên học sinh	Ngày sinh	Giới tính	Nơi sinh	Diện ưu tiên	Dân tộc	TP Gia Đình	Sửa	Xoá
1	Dương Quang Hai	2005-09-01	Nam	Hà Tây	Không có	Thái	Nông dân		
2	Dương Phương Nam	2005-08-11	Nam	Cầu Giấy	Không có	Kinh	Công nhân		
3	Dương Quang Minh	2005-02-20	Nam	Mỹ Đức	Không có	Kinh	Công chức/ Viên chức		
4	Dương Quang Vinh	2005-05-12	Nam	Ứng Hòa	Dân tộc thiểu số	Tày	Tiểu thương		
5	Dương Quốc Khánh	2005-09-21	Nam	Đan Phượng	Không có	Kinh	Nông dân		

Các
hàng
tên
mỗi
trang:
1
đến
5
của
1899

Hình 3. 34: Giao diện quản lý học sinh

3.9.2. Kiểm thử chức năng thêm học sinh

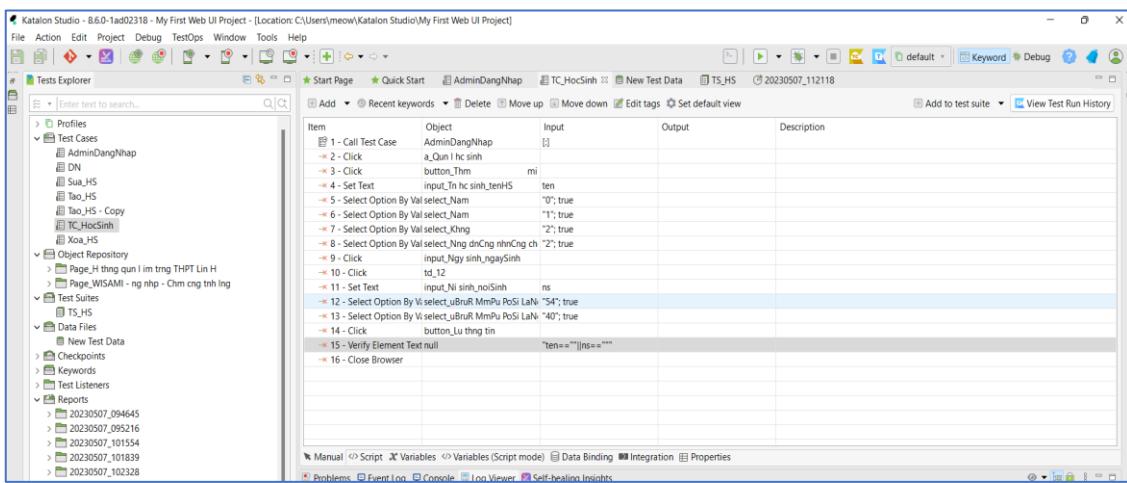
Testcase ID	Testcase Description	Steps To Perform	Expect Result	Actual Result
Hoc_sinh_01	Kiểm tra thêm học sinh thành công	1. Truy cập trang web 2. Chọn đăng nhập admin 3. Click submenu: Quản lý học sinh 4. Click button “Thêm mới” 5. Điền đầy đủ thông tin học sinh 6. Click “Lưu thông tin”	Thêm học sinh thành công	Pass
Hoc_sinh_02	Kiểm tra sửa Tên học sinh	1. Truy cập trang web 2. Chọn đăng nhập admin 3. Click submenu: Quản lý học sinh 4. Click button “Sửa” 7. Sửa Tên học sinh Click “Lưu”	Sửa thành công	Pass
Hoc_sinh_03	Kiểm tra xóa học sinh chưa có điểm	1. Truy cập trang web 2. Chọn đăng nhập admin	Xóa thành công	Pass

		<p>3. Click submenu: Quản lý học sinh</p> <p>4. Click button “Xóa”</p> <p>1. Click “Lưu”</p>	
--	--	--	--

Bảng 3. 7: Bảng test case thêm học sinh

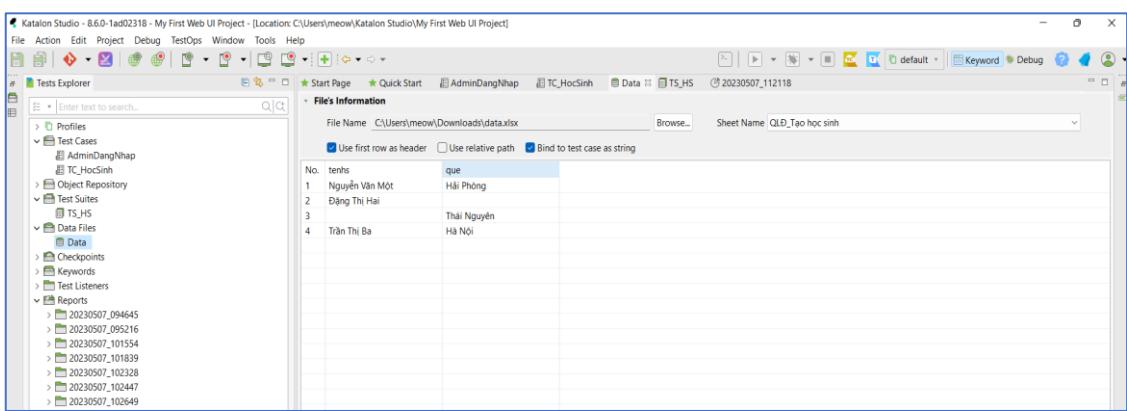
Thiết kế test case học sinh trên Katalon Studio (Thêm, Sửa, Xóa học sinh).

Trong trường hợp này, có thể Call Test Case Admin Đăng nhập để tránh lặp lại bước đăng nhập nhiều lần.



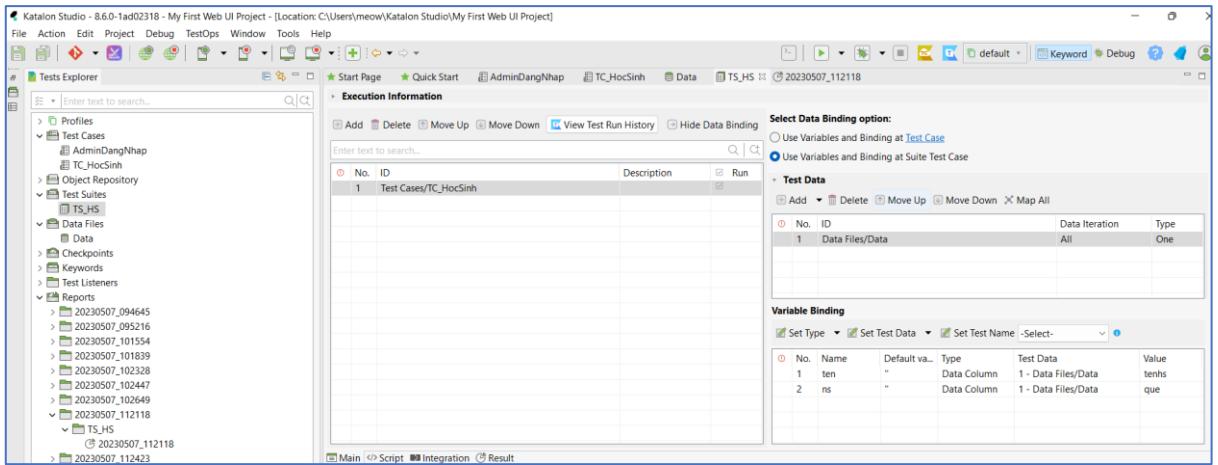
Hình 3. 35: Test case học sinh

Import file data excel để thực hiện test

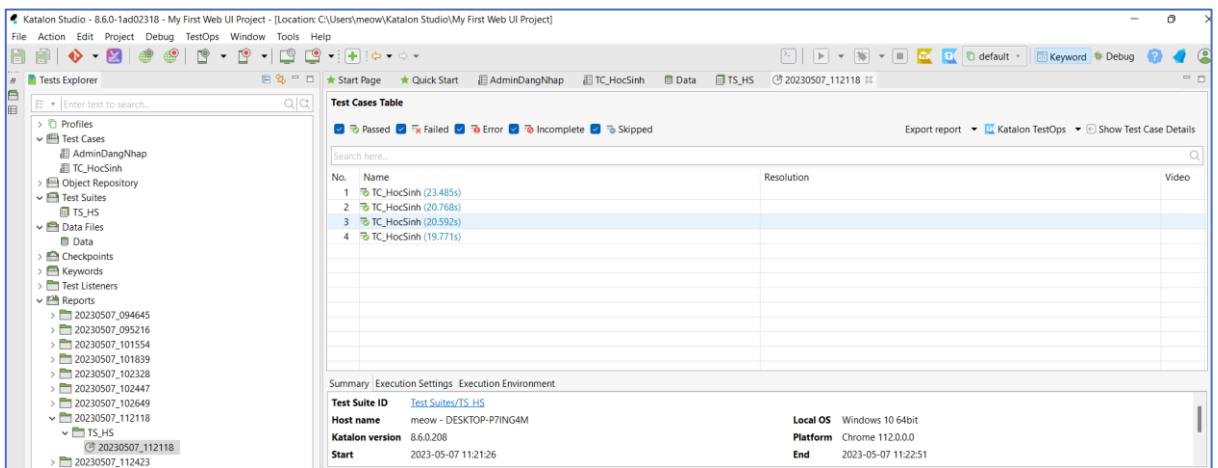


Hình 3. 36: Data import của test case thêm học sinh

Tạo Test suite, sau đó chạy thử và thu kết quả hình 3.38:



Hình 3. 37: Test suite ca kiểm thử Học sinh



Hình 3. 38: Kết quả sau khi thực hiện test

Kết quả test ca kiểm thử Học sinh, 4 test case, pass 4. Đáp ứng đúng kết quả mong muốn của Test case, vậy nên tính năng Thêm học sinh hoạt động bình thường.

KẾT LUẬN

Kết quả đạt được:

Sau quá trình tìm hiểu, nghiên cứu và tập hợp lại kiến thức, em đã đạt được một số kết quả sau bài báo cáo đồ án này như sau:

- Trình bày tổng quan về phần mềm, công nghệ phần mềm, lỗi phần mềm, và các vấn đề liên quan đến kiểm thử phần mềm
- Giới thiệu về công cụ Katalon Studio, các thao tác và các chức năng cơ bản của công cụ
- Áp dụng được từ những kiến thức đã tìm hiểu và học được thực hành trực tiếp với website chấm công Wisami Go và website quản lý điểm trường THPT Liên Hà
- Đồ án này có thể coi như là một tài liệu xúc tích, tổng hợp các vấn đề của kiểm thử và có thể coi là một phần mềm hướng dẫn sử dụng Katalon Studio bằng tiếng Việt có thể tham khảo

Hạn chế:

Do trình độ, khả năng và thời gian còn hạn chế nên báo cáo của em vẫn còn rất nhiều hạn chế như:

- 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 cao hơn được sử dụng trong Katalon Studio như test API.
- Trang Web chưa thực sự hoàn thiện và còn nhiều thiếu sót
- Kỹ thuật viết script vẫn còn yếu

Tuy vẫn còn nhiều hạn chế trong bài báo cáo, nhưng em đã rất cố gắng, chủ động tìm hiểu tài liệu để có thể hiểu khái quát về công cụ Katalon Studio. Trong thời gian tới em sẽ nghiên cứu kỹ hơn về Katalon Studio trong lĩnh vực an toàn thông tin. Đảm bảo ứng dụng Web có thể được kiểm tra về chức năng

cũng như về bảo mật. Để có thể tạo ra những trang Web/phần mềm an toàn, bảo mật và có tính ứng dụng cao.

Hướng phát triển:

- Ứng dụng kiểm thử cho nhiều phần mềm hơn
- Có kế hoạch xử lý rủi ro khi kiểm thử một trang web
- Phát triển bản thân thêm về kỹ thuật test

TÀI LIỆU THAM KHẢO

- [1] Nguyễn Văn Vy - Nguyễn Việt Hà, Giáo trình Kỹ nghệ phần mềm, Nhà xuất bản Giáo dục Việt Nam, 2009
- [2]<https://viblo.asia/p/kiem-thu-tu-dong-va-kiem-thu-thu-cong-nen-sudung-khi-nao-EoDkQqEqkbV>
- [3] <https://www.slideshare.net/qnv96/n-kim-th-phn-mm>
- [4] <https://docs.katalon.com/katalon-studio/docs/index.html>
- [5] <https://www.katalon.com/>
- [6]<https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-ofkatalon-studio-automation-testing-tool/>

PHỤ LỤC

1. Danh sách các report trong quá trình kiểm thử

1.1. Đăng ký Test Log

DangKy Test Log

Execution Environment

Host name: ngyue - DESKTOP-V3D0TTB
 Local OS: Windows 10 64bit
 Katalon version: 8.6.0.208
 Browser: Chrome 113.0.0.0

Test Execution Log

TEST SUITE DangKy

Full Name:	DangKy
Start / End / Elapsed:	2023-05-05 23:22:45.585 / 2023-05-05 23:24:04.771 / 00:01:19.186
Status:	6 test total, 6 passed, 0 failed, 0 error, 0 incomplete, 0 skipped

TEST CASE Test Cases/DK

Full Name:	DangKy/Test Cases/DK
Start / End / Elapsed:	2023-05-05 23:22:46.315 / 2023-05-05 23:23:01.655 / 00:00:15.340
Status:	PASSED

DATA BINDING

- TEST STEP: openBrowser("")
- TEST STEP: navigateToUrl("https://go.wisami.com/registration")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_H v tr_userName"), "name")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Email hoc Tr ng nhp_userEmail"), "acc")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Mt khu_password"), "pw")
- TEST STEP: click(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Nhp l mt khu_rePassword"), "v_pw")
- TEST STEP: if (name == "" || acc == "" || pw == "" || v_pw == "")
- TEST STEP: closeBrowser()

TEST CASE Test Cases/DK

Full Name:	DangKy/Test Cases/DK
Start / End / Elapsed:	2023-05-05 23:23:01.770 / 2023-05-05 23:23:15.182 / 00:00:13.412
Status:	PASSED

TEST CASE Test Cases/DK

Full Name:	DangKy/Test Cases/DK
Start / End / Elapsed:	2023-05-05 23:23:01.770 / 2023-05-05 23:23:15.182 / 00:00:13.412
Status:	PASSED

DATA BINDING

- TEST STEP: openBrowser("")
- TEST STEP: navigateToUrl("https://go.wisami.com/registration")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_H v tr_userName"), "name")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Email hoc Tr ng nhp_userEmail"), "acc")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Mt khu_password"), "pw")
- TEST STEP: click(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Nhp l mt khu_rePassword"), "v_pw")
- TEST STEP: if (name == "" || acc == "" || pw == "" || v_pw == "")
- TEST STEP: closeBrowser()

TEST CASE Test Cases/DK

Full Name:	DangKy/Test Cases/DK
Start / End / Elapsed:	2023-05-05 23:23:15.345 / 2023-05-05 23:23:34.466 / 00:00:19.121
Status:	PASSED

DATA BINDING

- TEST STEP: openBrowser("")
- TEST STEP: navigateToUrl("https://go.wisami.com/registration")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_H v tr_userName"), "name")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Email hoc Tr ng nhp_userEmail"), "acc")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Mt khu_password"), "pw")
- TEST STEP: setText(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_Nhp l mt khu_rePassword"), "v_pw")
- TEST STEP: click(findTestObject("Object Repository/Page_Wisami - ng k ti khon/input_ng k ti khon m"))
- TEST STEP: if (name == "" || acc == "" || pw == "" || v_pw == "")
- TEST STEP: closeBrowser()

TEST CASE Test Cases/DK		Expand All
Full Name:	DangKy/Test Cases/DK	
Start / End / Elapsed:	2023-05-05 23:23:34.556 / 2023-05-05 23:23:44.596 / 00:00:10.040	
Status:	PASSED	
DATA BINDING		
TEST STEP openBrowser("")		
TEST STEP navigateToUrl("https://go.wisami.com/registration")		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_H v ln_userName"), name)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Email hoc Tr ng nhp_userEmail"), acc)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Mt khu_password"), pw)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Nhp l mt khu_mPassword"), v_pw)		
TEST STEP click(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_ng k ti khon m"))		
TEST STEP if(name == "" acc == "" pw == "" v_pw == "")		
TEST STEP closeBrowser()		
TEST CASE Test Cases/DK		Expand All
Full Name:	DangKy/Test Cases/DK	
Start / End / Elapsed:	2023-05-05 23:23:44.668 / 2023-05-05 23:23:54.883 / 00:00:10.217	
Status:	PASSED	
DATA BINDING		
TEST STEP openBrowser("")		
TEST STEP navigateToUrl("https://go.wisami.com/registration")		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_H v ln_userName"), name)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Email hoc Tr ng nhp_userEmail"), acc)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Mt khu_password"), pw)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Nhp l mt khu_mPassword"), v_pw)		
TEST STEP click(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_ng k ti khon m"))		
TEST STEP if(name == "" acc == "" pw == "" v_pw == "")		
TEST STEP closeBrowser()		

TEST CASE Test Cases/DK		Expand All
Full Name:	DangKy/Test Cases/DK	
Start / End / Elapsed:	2023-05-05 23:23:54.945 / 2023-05-05 23:24:04.771 / 00:00:09.826	
Status:	PASSED	
DATA BINDING		
TEST STEP openBrowser("")		
TEST STEP navigateToUrl("https://go.wisami.com/registration")		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_H v ln_userName"), name)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Email hoc Tr ng nhp_userEmail"), acc)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Mt khu_password"), pw)		
TEST STEP setText(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_Nhp l mt khu_mPassword"), v_pw)		
TEST STEP click(findTestObject("Object Repository/Page_Wisami - ng k ti khonInput_ng k ti khon m"))		
TEST STEP if(name == "" acc == "" pw == "" v_pw == "")		
TEST STEP closeBrowser()		

1.2. Đăng nhập Test Log

DangNhap Test Log	
Execution Environment	
Host name: nguye - DESKTOP-V3D0TTB	
Local OS: Windows 10 64bit	
Katalon version: 8.6.0.208	
Browser: Chrome 113.0.0.0	
Test Execution Log	
TEST SUITE DangNhap	
Full Name:	DangNhap
Start / End / Elapsed:	2023-05-05 23:18:28.504 / 2023-05-05 23:19:12.160 / 00:00:43.656
Status:	4 test total, 4 passed, 0 failed, 0 error, 0 incomplete, 0 skipped
TEST CASE Test Cases/DN	
Full Name:	DangNhap/Test Cases/DN
Start / End / Elapsed:	2023-05-05 23:18:28.958 / 2023-05-05 23:18:45.300 / 00:00:16.342
Status:	PASSED
DATA BINDING	
TEST STEP openBrowser("")	
TEST STEP navigateToUrl("https://go.wisami.com/loginPass")	
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrInput_Email hoc Tr ng nhp_userName"), username)	
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrInput_Mt khu_password"), password)	
TEST STEP click(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrSpan_ng nhp"))	
TEST STEP if(username == "" password == "")	

DangNhap Test Cases/DN	
Full Name:	DangNhap/Test Cases/DN
Start / End / Elapsed:	2023-05-05 23:18:45.334 / 2023-05-05 23:18:56.140 / 00:00:10.806
Status:	PASSED
DATA BINDING	
TEST STEP openBrowser("")	
TEST STEP navigateToUrl("https://go.wisami.com/loginPass")	
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrInput_Email hoc Tr ng nhp_userName"), username)	
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrInput_Mt khu_password"), password)	
TEST STEP click(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrSpan_ng nhp"))	
TEST STEP if(username == "" password == "")	
TEST CASE Test Cases/DN	
Full Name:	DangNhap/Test Cases/DN
Start / End / Elapsed:	2023-05-05 23:18:56.168 / 2023-05-05 23:19:04.287 / 00:00:08.119
Status:	PASSED
DATA BINDING	
TEST STEP openBrowser("")	
TEST STEP navigateToUrl("https://go.wisami.com/loginPass")	
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrInput_Email hoc Tr ng nhp_userName"), username)	
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrInput_Mt khu_password"), password)	
TEST STEP click(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm cng trh lngrSpan_ng nhp"))	
TEST STEP if(username == "" password == "")	

TEST CASE Test Cases/ĐN		Expand All
Full Name:	DangNap/Test Cases/ĐN	
Start / End / Elapsed:	2023-05-05 23:19:04.343 / 2023-05-05 23:19:12.160 / 00:00:07.817	
Status:	PASSED	
DATA BINDING		
TEST STEP openBrowser("")		
TEST STEP navigateToUrl("https://tgo.wisami.com/loginPass")		
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm crng trnh lnput_Email hoc Tn ng nhp_username"), "username")		
TEST STEP setText(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm crng trnh lnput_Mt khu_password"), "password")		
TEST STEP click(findTestObject("Object Repository/Page_WISAMI - ng nhp - Chm crng trnh lnput_ng nhp"))		
TEST STEP if (username == "" password == "")		

1.3. Thêm học sinh Test Log

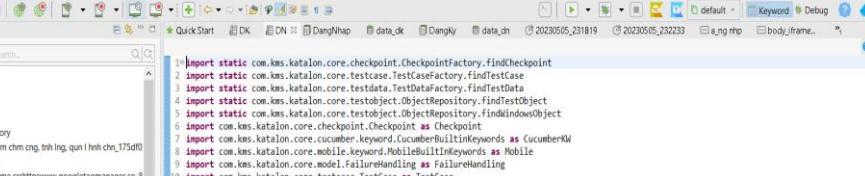
TS_HS Test Log	
Execution Environment	
Host name:	meow - DESKTOP-P7ING4M
Local OS:	Windows 10 64bit
Katalon version:	8.6.0.208
Browser:	Chrome 112.0.0.0
Test Execution Log	
TEST SUITE TS_HS	
Full Name:	TS_HS
Start / End / Elapsed:	2023-05-07 11:21:26.607 / 2023-05-07 11:22:51.960 / 00:01:25.353
Status:	4 test total, 4 passed, 0 failed, 0 error, 0 incomplete, 0 skipped
TEST CASE Test Cases/TC_HocSinh	
Full Name:	TS_HS/Test Cases/TC_HocSinh
Start / End / Elapsed:	2023-05-07 11:21:27.231 / 2023-05-07 11:21:50.716 / 00:00:23.485
Status:	PASSED
DATA BINDING	
TEST STEP openBrowser("")	
TEST STEP navigateToUrl("http://localhost/QuanLyDiemTHPT/login.php")	
TEST STEP setText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Tng THPT Lin H_username"), "admin")	
TEST STEP setEncryptedText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Tng THPT Lin H_password"), "RAIVpfpDOg-")	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_ng nhp"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Ha_Qun l hc sinh"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_Them m"))	
TEST STEP setText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Tn hc sinh_tenHS"), ten)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Nam N"), "0", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Nam N"), "1", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Khng_95a33"), "2", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Ng_drcng_nhnCng_clic_Vin chciTu thng_b50e16"), "2", true)	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Ng_sinh_ngaySinh"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_12"))	
TEST STEP setText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Ng_drcng_nhnCng_clic_Vin chciTu thng_b50e16"), ten)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Nam N"), "0", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Nam N"), "1", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Khng_95a33"), "2", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Ng_drcng_nhnCng_clic_Vin chciTu thng_b50e16"), "2", true)	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Ng_sinh_ngaySinh"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_12"))	
TEST CASE Test Cases/TC_HocSinh	
Full Name:	TS_HS/Test Cases/TC_HocSinh
Start / End / Elapsed:	2023-05-07 11:21:50.749 / 2023-05-07 11:22:11.517 / 00:00:20.768
Status:	PASSED
DATA BINDING	
TEST STEP openBrowser("")	
TEST STEP navigateToUrl("http://localhost/QuanLyDiemTHPT/login.php")	
TEST STEP setText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Tng THPT Lin H_username"), "admin")	
TEST STEP setEncryptedText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Tng THPT Lin H_password"), "RAIVpfpDOg-")	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_ng nhp"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Ha_Qun l hc sinh"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_Them m"))	
TEST STEP setText(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Tn hc sinh_tenHS"), ten)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Nam N"), "0", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Nam N"), "1", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Khng_95a33"), "2", true)	
TEST STEP selectOptionByValue(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hselect_Ng_drcng_nhnCng_clic_Vin chciTu thng_b50e16"), "2", true)	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hlnput_Ng_sinh_ngaySinh"))	
TEST STEP click(findTestObject("Object Repository/Page_H thng qun l im trng THPT Lin Hbuton_12"))	

<ul style="list-style-type: none">☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ng_ytinh", "noSinh"]☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hid_12"]☒ TEST STEP setTextIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ng_noSinh", "ns"]☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_uBrU MmPu PoSl LaNgCngB YC LaOlMngL_0254e1", "54", true]☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_uBrU MmPu PoSl LaNgCngB YC LaOlMngL_0254e1", "40", true]☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hbutton_Lu_thingtin"]☒ TEST STEP closeBrowser()	Expand All
<p>☒ TEST CASE: Test Cases/Tc_HocSinh</p> <p>Full Name: TS_HS/Test Cases/Tc_HocSinh</p> <p>Start / End / Elapsed: 2023-05-07 11:22:31.548 / 2023-05-07 11:22:32.140 / 00:00:20.592</p> <p>Status: PASSED</p> <p>☐ DATA BINDING</p> <p>☐ openBrowser("")</p> <p>☒ TEST STEP navigateToUrl("http://localhost/QuanLyDiemTHPT/login.php")</p> <p>☒ TEST STEP setTextIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Truong THPT Lin H_username", "admin"]</p> <p>☒ TEST STEP setEncryptedTextIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Truong THPT Lin H_password", "RAIVflpD0g="]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hbutton_ng_nhp"]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ng_hinh"]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ng_hinh"]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hbutton_Them_m"]</p> <p>☒ TEST STEP setTextIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ten_hc_sinh_tenHS", "ten"]</p> <p>☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_Nam_N", "0", true]</p> <p>☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_Nam_N", "1", true]</p> <p>☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_Khang_95a33f", "2", true]</p> <p>☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_Ng_nhCng_cic_Vin_chcTu_thng_b50e1f", "2", true]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ng_ytinh_ngaySinh"]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hid_12"]</p> <p>☒ TEST STEP setTextIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hinput_Ng_noSinh", "ns"]</p> <p>☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_uBrU MmPu PoSl LaNgCngB YC LaOlMngL_0254e1", "54", true]</p> <p>☒ TEST STEP selectOptionByValue/findTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hselected_uBrU MmPu PoSl LaNgCngB YC LaOlMngL_0254e1", "40", true]</p> <p>☒ TEST STEP clickIfNotTestObject["Object Repository/Page_H_thng_qun_l_i_mng THPT Lin Hbutton_Lu_thingtin"]</p> <p>☒ TEST STEP closeBrowser()</p>	

Test Case: Test Cases/TC_HocSinh		Expand All
Full Name:	TS_HS/Test Cases/TC_HocSinh	
Start / End / Elapsed:	2023-05-07 11:22:32.189 / 2023-05-07 11:22:51.960 / 00:00:19.771	
Status:	PASSED	
DATA BINDING:		
# TEST STEP: openBrowser("")		
# TEST STEP: navigateToUrl("http://localhost:QuanLyDiemTHPT/Login.php")		
# TEST STEP: setText(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Trng_THPT_LinH_username"), "admin")		
# TEST STEP: setEncryptedText(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Trng_THPT_LinH_password"), "RAIVpjljD0g=")		
# TEST STEP: click(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_ng_nhp"))		
# TEST STEP: click(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Qui_hc_sinh"))		
# TEST STEP: click(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHbutton_Them_moi"))		
# TEST STEP: setText(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Tr_hc_sinh_tenHS"), ten)		
# TEST STEP: selectOptionByValue(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHSelect_Nam_N"), "0", true)		
# TEST STEP: selectOptionByValue(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHSelect_Nam_N"), "1", true)		
# TEST STEP: selectOptionByValue(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHSelect_Khang_95e33f"), "2", true)		
# TEST STEP: selectOptionByValue(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHSelect_Ngu_dnCng_nhnCng_csc_Vin_chitieu_thang_b50e16"), "2", true)		
# TEST STEP: click(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Ngy_sinh_ngaySinh"))		
# TEST STEP: click(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Lu_thing"))		
# TEST STEP: setText(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHinput_Ni_sinh_noSinh"), ns)		
# TEST STEP: selectOptionByValue(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHSelect_uBruR_MmPu_PoSi_LaNgj_CngB_YC_LaoMngL_0254e1"), "54", true)		
# TEST STEP: selectOptionByValue(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHSelect_uBruR_MmPu_PoSi_LaNgj_CngB_YC_LaoMngL_0254e1"), "40", true)		
# TEST STEP: click(findTestObject("Object Repository/Page_H_thang_quan_lam_tring_THPT_LinHbutton_Lu_thing_lnt"))		
# TEST STEP: closeBrowser()		

2. Mã script của các test case

2.1. ĐĂNG KÝ



The screenshot shows the Katalon Studio interface with the following details:

- File Menu:** File, Action, Edit, Project, Debug, TestOps, Window, Tools, Help.
- Toolbar:** Includes icons for Run, Stop, Refresh, Save, and various test types like UI, API, Database, and Keyword.
- Sidebar:** Shows the project structure:
 - Profiles
 - Test Cases
 - DK
 - DN
 - Object Repository
 - Page_Pbn_mm cmh cng_tnh_lng_iinh_chv_175df0
 - a_ng_rhp
 - body_scriptswpsw.googletagmanager.co_B
 - button_Bt ui_WISAM
 - button_Dng_th min_h
 - div_S h ng_t ph_dyt_t ph dng th inq_o_500bte
 - Page_WISAMI_ng_kt_khon
 - Page_WISAM_ng_rhp - Chm cng trh lng
 - Test Suites
 - DangKy
 - DangNhap
 - Data File
 - data_dk
 - data_dn
 - Checkpoints
 - Keywords
 - Test Listeners
 - Reports
 - 20230505_231450
 - 20230506_231054
- Code Editor:** Displays Java code for a Cucumber test step. The code uses annotations like @Test, @CucumberTest, and @TestNG to define test cases. It includes imports for Page objects, WebDriver, and various keyword classes from the katalon.core package. The code also includes Selenium WebDriver initialization and navigation logic.

```

import com.kms.katalon.core.testdata.TestData as TestData
import com.kms.katalon.core.testing.keyword.WebUIBuiltInKeywords as TestNGKW
import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webservice.keyword.WebUIBuiltInKeywords as WebUI
import com.kms.katalon.core.windows.keyword.WindowsBuiltInKeywords as Windows
import internal.GlobalVariable as GlobalVariable
import org.openqa.selenium.Keys as Keys
WebUI.openBrowser('')
WebUI.navigateToUrl('https://go.wisami.com/loginPass')
WebUI.setText(findTestObject('Object Repository/Page_WISAMI - ng nhp - Chm cng trn lhng/input_email_hoc Tn ng nhp_userName'), 'username')
WebUI.setText(findTestObject('Object Repository/Page_WISAMI - ng nhp - Chm cng trn lhng/input_Mt khu_password'), 'password')
WebUI.click(findTestObject('Object Repository/Page_WISAMI - ng nhp - Chm cng trn lhng/span_ng nhp'))
if (username == "" || password == "") {
    WebUI.delay(1)
}
WebUI.closeBrowser()

```

2.2. Đăng nhập

```

import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint
import static com.kms.katalon.core.testcase.TestCaseFactory.findTestCase
import static com.kms.katalon.coretestdata.TestDataFactory.findTestData
import static com.kms.katalon.core.testobject.ObjectRepository.findTestObject
import static com.kms.katalon.core.testobject.ObjectRepository.findWindowsObject
import com.kms.katalon.core.checkpoint.Checkpoint as Checkpoint
import com.kms.katalon.core.cucumber.keyword.CucumberBuiltInKeywords as CucumberKW
import com.kms.katalon.core.mobile.keyword.MobileBuiltInKeywords as Mobile
import com.kms.katalon.core.model.FailureHandling as FailureHandling
import com.kms.katalon.core.testcase.TestCase as TestCase
import com.kms.katalon.core.testdata.TestData as TestData
import com.kms.katalon.core.testing.keyword.TestNGBuiltInKeywords as TestNGKW
import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webui.keyword.WebUIBuiltInKeywords as WebUI
import com.kms.katalon.core.windows.keyword.WindowsBuiltInKeywords as Windows
import internal.GlobalVariable as GlobalVariable
import org.openqa.selenium.Keys as Keys
WebUI.openBrowser('')
WebUI.navigateToUrl('https://go.wisami.com/registration')
WebUI.setText(findTestObject('Object Repository/Page_Wisami - ng k t khon/input_H v tn_userName'), 'name')
WebUI.setText(findTestObject('Object Repository/Page_Wisami - ng k t khon/input_Email_hoc Tn ng nhp_userEmail'), 'acc')

```

```

import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webservice.keyword.WebUIBuiltInKeywords as WebUI
import com.kms.katalon.core.windows.keyword.WindowsBuiltInKeywords as Windows
import internal.GlobalVariable as GlobalVariable
import org.openqa.selenium.Keys as Keys
WebUI.openBrowser('')
WebUI.navigateToUrl('https://go.wisami.com/registration')
WebUI.setText(findTestObject('Object Repository/Page_Wisami - ng k t khon/input_H v tn_userName'), 'name')
WebUI.setText(findTestObject('Object Repository/Page_Wisami - ng k t khon/input_Email_hoc Tn ng nhp_userEmail'), 'acc')
WebUI.setText(findTestObject('Object Repository/Page_Wisami - ng k t khon/input_Mt khu_password'), 'pw')
WebUI.click(findTestObject('Object Repository/Page_Wisami - ng k t khon/span_ng k t khon mi'))
if (name == "" || acc == "" || pw == "" || v_pw == "") {
}
WebUI.closeBrowser()

```

2.3. Thêm học sinh

The screenshot shows the Katalon Studio interface with the following details:

- Top Bar:** Katalon Studio - 8.6.0-Tad02318 - My First Web UI Project [Location: C:\Users\meow\Katalon Studio\My First Web UI Project]
- Menu Bar:** File, Action, Edit, Project, Debug, TestOps, Window, Tools, Help
- Toolbar:** Includes icons for Run, Stop, Refresh, Save, and various test-related functions.
- Left Sidebar (Tree View):**
 - Profiles
 - Test Cases
 - AdminDangNhap
 - TC_HocSinh
 - Object Repository
 - Test Suites
 - T5_HS
 - Data Files
 - Data
 - Checkpoints
 - Keywords
 - Test Listeners
 - Reports
 - 20230507_094645
 - 20230507_095216
 - 20230507_101554
 - 20230507_101839
 - 20230507_102328
 - 20230507_102447
 - 20230507_102649
 - 20230507_112118
 - T5_HS

- Right Side (Code Editor):** Displays the script content for the T5_HS test suite.

```
11 import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint()
26
28* Some methods below are samples for using Setup/TearDown in a test suite.
30
31 /**
32 * Setup test suite environment.
33 */
34@Setup(skipped = true) // Please change skipped to be false to activate this method.
35 def setup() {
36     // Put your code here.
37 }
38
39 /**
40 * Clean test suites environment.
41 */
42@TearDown(skipped = true) // Please change skipped to be false to activate this method.
43 def tearDown() {
44     // Put your code here.
45 }
46
47 /**
48 * Run before each test case starts.
49 */
50@SetupTestCase(skipped = true) // Please change skipped to be false to activate this method.
51 def setupTestCase() {
52     // Put your code here.
53 }
```

The screenshot shows the Katalon Studio interface. The left sidebar contains the 'Tests Explorer' tree view, which includes sections for Profiles, Test Cases (with sub-items AdminDangNhap and TC_HocSinh), Object Repository, Test Suites (with sub-item TS_HS), Data File, Checkpoints, Keywords, Test Listeners, and Reports (with sub-items 20230507_094645, 20230507_095216, 20230507_101554, 20230507_101839, 20230507_102328, 20230507_102447, 20230507_102649, 20230507_112118, and TS_HS). The right panel displays a Groovy script editor with code related to test suites and teardown methods. The top menu bar includes File, Action, Edit, Project, Debug, TestOps, Window, Tools, Help, and a toolbar with various icons.