

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG**



ISO 9001:2015

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Vũ Công Minh

Giảng viên hướng dẫn: ThS. Nguyễn Trịnh Đông

HẢI PHÒNG - 2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB BẰNG CÔNG CỤ
SELENIUM

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Vũ Công Minh

Giảng viên hướng dẫn : ThS. Nguyễn Trịnh Đông

HẢI PHÒNG - 2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: Vũ Công Minh

Mã SV: 1412101135

Lớp: CT1801

Ngành: Công nghệ thông tin

Tên đề tài: Kiểm thử ứng dụng trên nền Web bằng công cụ Selenium

LỜI CẢM ƠN

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

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

Em xin chân thành cảm ơn!

Hải Phòng, ngày ... tháng ... năm 2018.

Sinh viên thực hiện

Vũ Công Minh

MỤC LỤC

LỜI CẢM ƠN	1
MỤC LỤC	5
MỞ ĐẦU	7
DANH MỤC HÌNH VẼ VÀ BẢNG BIỂU	9
DANH MỤC TỪ VIẾT TẮT.....	11
CHƯƠNG 1: PHẦN MỀM VÀ KIỂM THỬ PHẦN MỀM.....	12
1.1. Phần mềm và khái niệm liên quan	12
1.1.1. Phần mềm.....	12
1.1.2. Lỗi phần mềm	12
1.1.3. Yêu cầu của khách hàng.....	13
1.1.4. Đặc tả yêu cầu phần mềm	14
1.1.5. Chất lượng và độ tin cậy của phần mềm.....	15
1.2. Kiểm thử phần mềm.....	15
1.2.1. Khái niệm	15
1.2.2. Vai trò của kiểm thử phần mềm.....	16
1.2.3. Các cấp độ trong kiểm thử phần mềm	16
1.2.4. Quy trình kiểm thử phần mềm	18
1.2.5. Phân loại kiểm thử phần mềm.....	21
1.2.6. Các mức độ nghiêm trọng của lỗi	24
1.2.7. Ca kiểm thử	25
1.2.8. Kiểm thử tự động	27
1.2.9. Nguyên tắc quan trọng trong kiểm thử phần mềm	29
1.3. Các kỹ thuật xác định ca kiểm thử	31
1.3.1. Kỹ thuật phân vùng tương đương	31
1.3.2. Kỹ thuật phân tích giá trị biên.....	32
1.3.3. Đoán lỗi	34
1.3.4. Kỹ thuật chuyển trạng thái	34
1.4. Kết luận	34
CHƯƠNG 2: KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB	36
2.1. Khái quát về kiểm thử ứng dụng trên nền Web	36
2.1.1. Khái quát	36
2.1.2. Các loại ứng dụng Web.....	36

2.1.3. Đặc điểm về chất lượng của một ứng dụng trên nền Web.....	37
2.2. Công việc chính khi kiểm thử ứng dụng Web	39
2.2.1. Kiểm thử chức năng	39
2.2.2. Kiểm thử khả năng sử dụng	41
2.2.3. Kiểm thử sự tương thích	42
2.2.4. Kiểm thử hiệu suất	43
2.2.5. Kiểm thử bảo mật.....	44
2.3. Một số công cụ hỗ trợ kiểm thử ứng dụng trên nền Web	44
2.3.1. Công cụ kiểm thử hiệu năng	44
2.3.2. Công cụ kiểm thử bảo mật	45
2.3.3. Công cụ kiểm thử chức năng	46
2.4. Kết luận	47
CHƯƠNG 3: KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB BẰNG CÔNG CỤ SELENIUM	48
3.1. Công cụ kiểm thử tự động Selenium	48
3.1.1. Giới thiệu chung về Selenium.....	48
3.1.2. Selenium IDE	49
3.2. Một số công cụ hỗ trợ kiểm thử ứng dụng Web	61
3.2.1. Firebug	61
3.2.2. Monosnap	62
3.2.3. Công cụ quản lý lỗi (bug) MantisBT	63
3.3. Bài toán thực tế	66
3.3.1. Giới thiệu bài toán.....	66
3.3.2. Kiểm thử chức năng đăng ký tài khoản trên website https://id.zing.vn/ sử dụng công cụ Selenium IDE	67
3.4. Kết luận	73
KẾT LUẬN	74
DANH MỤC TÀI LIỆU THAM KHẢO.....	76

MỞ ĐẦU

Ngày nay, công nghệ thông tin nói chung và công nghệ phần mềm nói riêng đang chiếm một vị trí quan trọng trong tiến trình công nghiệp hoá, hiện đại hoá đất nước. Song song với việc phát triển công nghệ phần mềm luôn tiềm ẩn những thách thức cho dành các doanh nghiệp, nhà phát triển phần mềm trong việc kiểm soát lỗi, chất lượng đầu ra của sản phẩm. Tuy nhiên ở Việt Nam, số lượng các kiểm thử viên vẫn chưa đáp ứng được với nhu cầu của thị trường. Tại Hội nghị Quốc tế về kiểm thử phần mềm tự động (12/2011, TP. HCM), các chuyên gia đã nhận định: “Với đà tăng trưởng mạnh mẽ của ngành gia công phần mềm, trong vài năm tới, Việt Nam thiếu khoảng 10.000 kiểm thử viên.”

Bên cạnh đó, xu hướng áp dụng tự động hoá đang được triển khai rộng rãi ở nhiều lĩnh vực, trong đó có kiểm thử phần mềm. Đặc biệt, khi kiểm thử phần mềm là công đoạn chiếm phần lớn thời gian trong quá trình phát triển dự án phần mềm thì sự ra đời của các công cụ kiểm thử tự động càng có ý nghĩa hơn bao giờ hết, giúp tiết kiệm thời gian, công sức và tiền bạc. Selenium là một công cụ hỗ trợ kiểm thử tự động dành cho các ứng dụng Web, hoạt động trên hầu hết các trình duyệt phổ biến hiện nay như Firefox, Chrome, Internet Explorer, Safari, v.v. cũng như hỗ trợ số lượng lớn các ngôn ngữ lập trình Web phổ biến. Công cụ Selenium hiện được đánh giá là một trong những công cụ tốt nhất cho kiểm thử tự động các ứng dụng Web.

Với mong muốn được tìm hiểu sâu về lĩnh vực kiểm thử phần mềm cũng như trở thành một kỹ sư kiểm thử phần mềm sau khi tốt nghiệp đại học, em đã chọn đề tài “Kiểm thử ứng dụng trên nền Web bằng công cụ Selenium.” Trong quá trình làm đồ án, do còn hạn chế về thời gian và kinh nghiệm thực tế, em mong nhận được những góp ý chân thành từ thầy cô và các bạn.

Đề tài giới thiệu về lý thuyết kiểm thử phần mềm, các công cụ hỗ trợ kiểm thử tự động. Ngoài ra, đề tài đi sâu vào việc tìm hiểu, sử dụng các tính năng, công cụ của bộ phần mềm Selenium như:

- Đưa ra hướng dẫn cài đặt, sử dụng hiệu quả bộ công cụ.

- Ứng dụng các kiến thức đã học được để viết một kịch bản kiểm thử cho ứng dụng cụ thể.

Đề án được tổ chức làm 5 phần như sau:

- Mở đầu: Trình bày rõ lý do chọn đề tài, mục tiêu nghiên cứu đề án và bố cục của đề án.

- Chương 1: *Phần mềm và kiểm thử phần mềm*. Chương này trình bày các khái niệm cơ bản về phần mềm, kiểm thử phần mềm và các kỹ thuật kiểm thử phần mềm.

- Chương 2: *Kiểm thử ứng dụng trên nền Web*. Chương này trình bày chi tiết các khái niệm về kiểm thử ứng dụng Web, các công việc khi kiểm thử ứng dụng Web, giới thiệu một số công cụ hỗ trợ kiểm thử ứng dụng web.

- Chương 3: *Kiểm thử ứng dụng trên nền Web bằng công cụ Selenium*. Giới thiệu chung về Selenium, các cài đặt và sử dụng bộ công cụ, ứng dụng thực tế với Selenium.

- Kết luận: Phần này đưa ra những kết quả đề án đạt được, những thiếu sót chưa thực hiện được và hướng phát triển đề tài trong tương lai.

DANH MỤC HÌNH VẼ VÀ BẢNG BIỂU

Hình 1-1: Vòng đời của quá trình kiểm thử.	16
Hình 1-2: Quy trình kiểm thử phần mềm.	18
Hình 1-3: Xác định ca kiểm thử với kiểm thử hộp trắng.	22
Hình 1-4: Minh họa kỹ thuật kiểm thử hộp đen.	23
Hình 1-5: Biểu đồ Venn nguồn các ca kiểm thử.	23
Hình 1-6: Bảng phân loại mức độ nghiêm trọng của lỗi.	24
Hình 1-7: Minh họa mẫu ca kiểm thử đơn giản.	26
Hình 1-8: Minh họa kỹ thuật phân vùng tương đương.	32
Hình 1-9: Minh họa kỹ thuật phân tích giá trị biên.	33
Hình 3-1: Truy cập trang Download tiện ích Selenium IDE.	50
Hình 3-2: Tiến hành chọn phiên bản Selenium IDE để cài đặt.	51
Hình 3-3: Thêm tiện ích Selenium IDE vào Firefox.	51
Hình 3-4: Xác nhận cài đặt tiện ích Selenium IDE vào trình duyệt.	51
Hình 3-5: Khởi động lại trình duyệt Firefox để hoàn tất quá trình cài đặt.	52
Hình 3-6: Khởi chạy tiện ích Selenium IDE.	52
Hình 3-7: Giao diện khởi chạy Selenium IDE.	52
Hình 3-8: Giải thích một số chức năng, ký hiệu trong Selenium IDE.	53
Hình 3-9: Kịch bản kiểm thử được Selenium IDE lưu trữ dưới dạng HTML.	54
Hình 3-10: Chức năng tạo mới ca kiểm thử/bộ kiểm thử nằm trong menu File.	55
Hình 3-11: Minh họa thao tác lưu ca kiểm thử.	55
Hình 3-12: Minh họa thao tác lưu bộ kiểm thử.	56
Hình 3-13: Minh họa thao tác mở ca kiểm thử.	56
Hình 3-14: Minh họa thao tác mở bộ kiểm thử đã lưu.	56
Hình 3-15: Thiết lập điểm dừng cho ca kiểm thử.	57
Hình 3-16: Thiết lập điểm bắt đầu cho ca kiểm thử.	58
Hình 3-17: Minh họa thao tác chèn dòng lệnh mới.	59
Hình 3-18: Chèn nhận xét cho một dòng lệnh trong Selenium IDE.	59
Hình 3-19: Bảng liệt kê một số lệnh thường dùng trong Selenium IDE.	61
Hình 3-20: Giao diện tiện ích hỗ trợ kiểm thử Firebug.	61
Hình 3-21: Cài đặt công cụ Firebug trong trình quản lý Add-ons của Firefox.	62

Hình 3-22: Giao diện ứng dụng chụp ảnh màn hình Monosnap.....	63
Hình 3-23: Giao diện trang chủ MantisBT.	65
Hình 3-24: Giao diện hiện tại của website Zing ID.	66
Hình 3-25: Thực thi ca kiểm thử ZID_101 trên Selenium IDE.	68
Hình 3-26: Thực thi ca kiểm thử ZID_102 trên Selenium IDE.	71
Hình 3-27: Thực thi ca kiểm thử ZID_103 trên Selenium IDE.	72
Hình 3-28: Báo cáo lỗi thông qua công cụ MantisBT.	73

DANH MỤC TỪ VIẾT TẮT

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

CHƯƠNG 1: PHẦN MỀM VÀ KIỂM THỬ PHẦN MỀM

Chương đầu tiên của đồ án đi sâu vào việc tìm hiểu các khái niệm về phần mềm và kiểm thử phần mềm, giúp khái quát việc phân loại kiểm thử phần mềm, đưa ra các quy trình, mức độ, các kỹ thuật trong kiểm thử phần mềm.

1.1. Phần mềm và khái niệm liên quan

1.1.1. Phần mềm

Phần mềm thường được mô tả với ba bộ phận cấu thành:

- *Tập các lệnh (chương trình máy tính) trên máy tính khi được thực hiện sẽ tạo ra các dịch vụ và đem lại những kết quả mong muốn cho người dùng.*
- *Các cấu trúc dữ liệu (lưu giữ trên các bộ nhớ) làm cho chương trình thao tác hiệu quả với các thông tin thích hợp và nội dung thông tin được số hoá.*
- *Các tài liệu để mô tả các thao tác, cách sử dụng và bảo trì phần mềm (hướng dẫn sử dụng, tài liệu kỹ thuật, tài liệu phân tích, thiết kế, kiểm thử, v.v.) [1].*

1.1.2. Lỗi phần mềm

Lỗi phần mềm nhìn chung là sự không khớp giữa chương trình và đặc tả của nó, kéo theo những vấn đề xuất hiện trong các giai đoạn phát triển phần mềm.

Lỗi phần mềm thường xuất hiện ở các hình thức sau đây:

- Sai (Fault): Khi phần mềm gặp lỗi sẽ đưa đến những sai sót. Tuy nhiên, không dễ để phát hiện ra sai sót trong quá trình phát triển phần mềm. Sai lầm có thể xuất hiện ở ngay đầu quy trình phát triển phần mềm khi người phân tích, thiết kế bỏ sót thông tin dẫn tới thiếu chức năng mà lẽ ra cần phải có.

- **Thất bại (Failure):** *Thất bại* dễ nhận thấy nhất khi một lỗi được thực thi. Chúng thường xuất hiện dưới 2 dạng: thất bại có thể chạy được (ví dụ như mã nguồn) và thất bại chỉ liên kết với các lỗi về nhiệm vụ. Ngoài ra, có thể kể đến các *thất bại* liên quan tới các lỗi do bỏ quên. Chúng ta có thể hạn chế *thất bại* ngay tại bước đầu tiên của quy trình phát triển phần mềm nếu việc khảo sát được thực hiện tốt.
- **Sự cố (Incident):** *Sự cố* thường được liên kết với một thất bại. Tuy nhiên nó khác với thất bại ở chỗ *sự cố* luôn hiển thị cho người dùng hoặc kiểm thử viên biết về sự tồn tại của nó.
- **Thừa:** 1 số chức năng không có trong bản đặc tả yêu cầu phần mềm nhưng lại xuất hiện trong phần mềm được xây dựng.

Ngoài ra, còn xuất hiện 1 số lỗi phi chức năng như phần mềm khó sử dụng, tốc độ không đáp ứng yêu cầu (vấn đề hiệu năng) hay giao diện khó nhìn cũng dễ khiến cho người sử dụng nghĩ rằng phần mềm đang hoạt động không đúng.

1.1.3. Yêu cầu của khách hàng

Phần mềm được phát triển dựa trên nhu cầu của khách hàng. Chính vì lẽ đó, các chức năng của phần mềm được xây dựng dựa trên việc thu thập, phân tích, khảo sát nhu cầu của khách hàng thông qua những yêu cầu cụ thể. Đối với phần mềm, yêu cầu thường được tổng hợp từ nhiều người, nhiều tổ chức có mức độ chuyên môn và mức độ tham gia cũng như tương tác với phần mềm khác nhau trong môi trường hoạt động của nó.

Có thể phân loại yêu cầu của khách hàng cho sản phẩm phần mềm thành một số loại như sau:

- **Phân loại theo sản phẩm và tiến trình**

- *Yêu cầu sản phẩm:* là những đòi hỏi hay ràng buộc mà phần mềm phải thực hiện [2].

- *Yêu cầu tiến trình*: là những ràng buộc liên quan đến việc phát triển phần mềm (kỹ thuật sử dụng, mô hình phát triển, v.v.) [2].

Ví dụ: Khách hàng muốn phát triển một website làm bài thi trực tuyến. Lúc này, *yêu cầu sản phẩm* là xây dựng website thi trực tuyến với các tính năng như quản lý câu hỏi; quản lý đề thi; cho phép người dùng có thể tham gia làm bài thi; quản trị viên có thể duyệt các câu hỏi và bộ đề thi trước khi đăng lên website. Việc website được phát triển theo mô hình Agile hay mô hình thác nước chính là *yêu cầu tiến trình* của sản phẩm phần mềm.

- **Phân loại theo chức năng**

- *Yêu cầu chức năng*: đặc tả các chức năng mà phần mềm cần phải thực hiện.

- *Yêu cầu phi chức năng*: là các ràng buộc về giải pháp và chất lượng (hiệu năng, việc bảo trì, mức độ an toàn, bảo mật, v.v.).

- *Yêu cầu đặc tả các thuộc tính nổi bật*: là đặc tả cho các thuộc tính phụ thuộc vào sự vận hành, đặc biệt là kiến trúc hệ thống. Các thuộc tính này không thể xác định được cho từng thành phần đơn lẻ.

- **Phân loại theo tính kiểm định**

- Những yêu cầu mang tính mơ hồ, không thể kiểm định

- Những yêu cầu đã rõ ràng và có thể kiểm định được.

- **Phân loại theo phạm vi đặc tả**

- *Yêu cầu hệ thống*: đặc tả các cấu hình, cơ sở hạ tầng, phần cứng, phần mềm, con người, kỹ thuật, v.v. của toàn bộ hệ thống.

- *Yêu cầu phân mềm*: đặc tả các chức năng, giao diện, v.v. của các cấu phần phần mềm.

1.1.4. Đặc tả yêu cầu phần mềm

Từ yêu cầu của khách hàng và những yêu cầu bắt buộc khác, đặc tả yêu cầu phần mềm được viết ra để mô tả một cách chính xác các yêu cầu cần đáp ứng của sản phẩm phần mềm. Đây cũng chính là tài liệu cơ sở để lập trình viên, kiểm thử viên và các bộ phận khác dựa vào để phát triển phần mềm hoàn chỉnh, đúng với yêu cầu đặt ra ban đầu. Các khái niệm về lỗi đã nói ở

mục 1.1.2 cũng chính là đề cập đến việc phần mềm sau khi xây dựng hoạt động không đúng với bản đặc tả yêu cầu phần mềm. Tài liệu đặc tả yêu cầu phần mềm cũng cần cung cấp đầy đủ các thông tin về chi phí, rủi ro và lịch trình cho quá trình phát triển sản phẩm.

Đặc tả yêu cầu phần mềm được viết ra phục vụ rất nhiều đối tượng từ người dùng hệ thống, khách hàng đến các nhà phát triển và bảo trì phần mềm. Do đó, tài liệu đặc tả nên được viết bằng ngôn ngữ tự nhiên, sử dụng biểu đồ, bảng biểu để đảm bảo tính dễ hiểu, dễ sử dụng cho tất cả các đối tượng trên.

1.1.5. Chất lượng và độ tin cậy của phần mềm

Chất lượng của phần mềm trước hết là sự đáp ứng các yêu cầu đề ra trong bản đặc tả yêu cầu phần mềm. Có thể kể đến các yếu tố đại diện cho chất lượng phần mềm như: tính đúng đắn, tính hiệu quả, độ tin cậy, tính khả kiểm thử, dễ học, dễ sử dụng, dễ bảo trì... Ta có thể thấy độ tin cậy chỉ là một trong những yếu tố đánh giá chất lượng phần mềm. Tuy nhiên người kiểm thử lại hay nhầm lẫn giữa khái niệm chất lượng và độ tin cậy của phần mềm. Sau quá trình kiểm thử đảm bảo phần mềm có thể chạy ổn định, kiểm thử viên thường sẽ cho rằng phần mềm lúc này đã đạt chất lượng tốt.

Độ tin cậy của phần mềm là xác suất để phần mềm chạy không có thất bại trong một khoảng thời gian nhất định [3]. Ngoài ra, có thể dựa vào thời gian khắc phục sự cố để đánh giá độ tin cậy của phần mềm.

Trong phần tiếp theo, chúng ta sẽ tìm hiểu về khái niệm cũng như các vấn đề xung quanh việc *Kiểm thử phần mềm*.

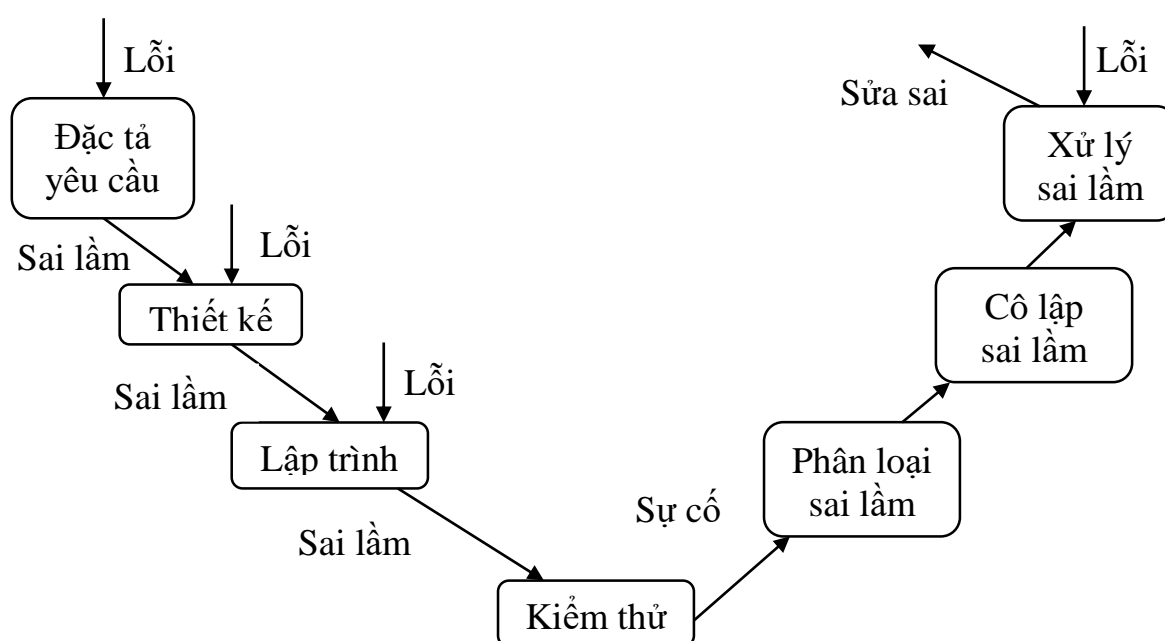
1.2. Kiểm thử phần mềm

1.2.1. Khái niệm

Kiểm thử phần mềm là một cuộc kiểm tra được tiến hành để cung cấp cho các bên liên quan thông tin về chất lượng của sản phẩm hoặc dịch vụ được kiểm thử [4]. Hiểu theo cách đơn giản hơn, kiểm thử phần mềm là quá trình tìm thất bại hoặc chứng tỏ việc tiến hành của phần mềm là đúng đắn.

1.2.2. Vai trò của kiểm thử phần mềm

Kiểm thử phần mềm chiếm một vị trí quan trọng trong việc nâng cao chất lượng cũng như độ tin cậy của phần mềm trong quá trình phát triển. Hoàn thành vòng quay “đưa lỗi vào – tìm lỗi – khử lỗi đi” của quy trình kiểm thử phần mềm sẽ thu lại được những cải tiến đáng kể cho chất lượng sản phẩm phần mềm. Việc biết được sản phẩm phần mềm tốt tới mức nào trước khi đưa vào sử dụng sẽ hạn chế tối đa những rủi ro gặp phải trong quá trình phát triển phần mềm.



Hình 1-1: Vòng đời của quá trình kiểm thử.

1.2.3. Các cấp độ trong kiểm thử phần mềm

Có rất nhiều cách để chia cấp độ kiểm thử phần mềm, nhưng tựu chung lại sẽ gồm 4 cấp độ sau:

Kiểm thử đơn vị: Cấp độ này chủ yếu do lập trình viên trực tiếp thực hiện. Phần mềm khi phát triển sẽ bao gồm nhiều đơn vị chức năng (hàm, phương thức) hợp thành. Mỗi lập trình viên sẽ đảm nhiệm việc phát triển một hay nhiều đơn vị chức năng. Kiểm thử đơn vị chính là việc lập trình viên sau khi hoàn thành code đơn vị chức năng của mình sẽ tiến hành kiểm thử chức năng đó một cách cô lập nhằm phát hiện ra lỗi và khắc phục trước khi tích

hợp với các đơn vị chức năng khác. Kiểm thử đơn vị thường được tiến hành theo 2 giai đoạn: kiểm thử đơn vị tĩnh và kiểm thử đơn vị động.

Kiểm thử tích hợp: Sau khi kiểm thử đơn vị được tiến hành bởi chính lập trình viên viết ra nó, các đơn vị chức năng sẽ được ghép lại với nhau để tạo thành hệ thống đầy đủ và có thể làm việc được. Các đơn vị chức năng hoạt động tốt khi ở trạng thái độc lập riêng rẽ, nhưng khi ghép lại sẽ có thể xuất hiện những lỗi về giao diện hoặc cho ra kết quả không đúng khi phải sử dụng dữ liệu từ những đơn vị chức năng khác. Đó chính là lý do tại sao phải tiếp tục kiểm thử để phát hiện ra những lỗi kể trên. Người ta thường chia bước kế tiếp này thành 2 giai đoạn: kiểm thử tích hợp và kiểm thử hệ thống. Ở mức kiểm thử tích hợp, các đơn vị chức năng được kết hợp lại với nhau và tiến hành kiểm thử chúng theo phương pháp tăng dần để đảm bảo cụm các đơn vị chức năng sẽ làm việc ổn định trong môi trường thử nghiệm.

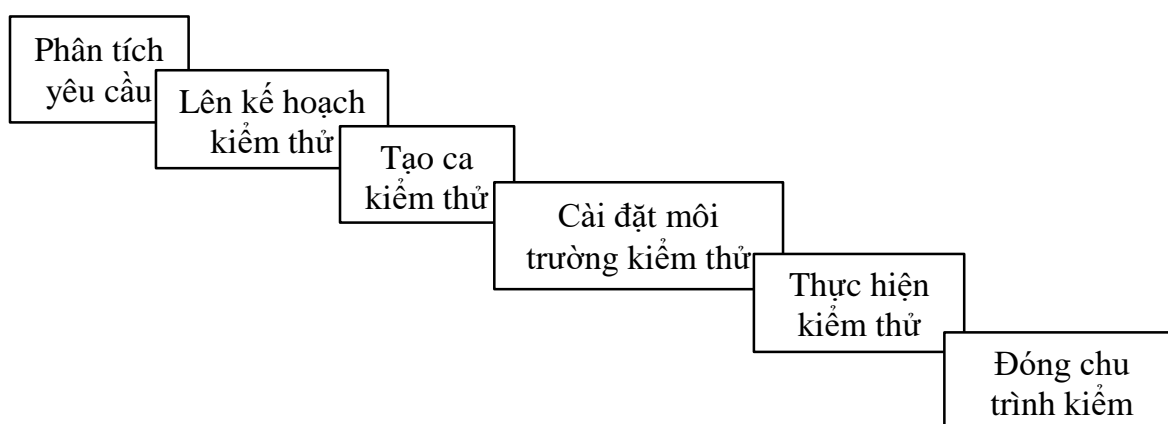
Kiểm thử hệ thống: Sau khi tất cả các đơn vị chức năng đã được tích hợp lại với nhau tạo thành một hệ thống hoàn chỉnh, kiểm thử hệ thống sẽ được thực thi để đảm bảo sản phẩm phần mềm đáp ứng đầy đủ các yêu cầu trong bản đặc tả yêu cầu phần mềm. Đây là công việc tốn nhiều công sức nhất trong quá trình kiểm thử phần mềm. Đồng thời cũng sử dụng nhiều kỹ thuật kiểm thử khác nhau như: kiểm thử giao diện người dùng, kiểm thử chức năng, kiểm thử hiệu năng, kiểm thử tính dễ dùng, v.v. để hoàn tất công việc kiểm thử trong cấp độ này.

Kiểm thử chấp nhận: Khi kiểm thử hệ thống hoàn tất, sản phẩm phần mềm được coi như đã sẵn sàng cho việc đưa vào sử dụng thực tế. Lúc này, phần mềm cần được tiến hành cấp độ kiểm thử cuối cùng – kiểm thử chấp nhận bởi chính khách hàng hay người sử dụng phần mềm. Tuy có phần tương tự như kiểm thử hệ thống nhưng mục đích chính của kiểm thử chấp nhận là quyết định việc đưa vào sử dụng chính thức sản phẩm phần mềm. Người ta dựa trên các số liệu thống kê thực tế về chất lượng, độ tin cậy của phần mềm để quyết định triển khai cho người dùng cuối. Kiểm thử chấp nhận thường

được thực hiện dưới hình thức cho một nhóm người dùng thử sản phẩm phần mềm để phát hiện các lỗi và nhận phản hồi từ người dùng. Trong đó, phiên bản *alpha* dành cho đội phát triển phần mềm và phiên bản *beta* được cung cấp cho người sử dụng thật để đưa ra đánh giá trong môi trường thực tế. Ở thời điểm hiện tại, kiểm thử chấp nhận được coi là cấp độ quy chuẩn bắt buộc không thể thiếu trong quy trình phát triển của nhiều sản phẩm phần mềm.

1.2.4. Quy trình kiểm thử phần mềm

Kiểm thử phần mềm 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. Các giai đoạn trong quy trình kiểm thử phần mềm được biểu diễn tổng quát bằng sơ đồ sau:



Hình 1-2: Quy trình kiểm thử phần mềm.

- **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, v.v.)

➤ *Hoạt động cụ thể:*

- Xác định loại kiểm thử sẽ thực hiện.

- Tổng hợp chi tiết về và mức độ tập trung thứ tự ưu tiên.
- Chuẩn bị RTM (Requirement Traceability Matrix – một tài liệu dưới dạng bảng sử dụng để theo dõi các yêu cầu của khách hàng và kiểm tra xem các yêu cầu này đã được đáp ứng đầy đủ hay chưa)
- Xác định môi trường kiểm thử.
- Phân tích khả năng sử dụng kiểm thử tự động.
 - *Tài liệu sử dụng:*
 - RTM.
 - Báo cáo về khả năng sử dụng kiểm thử tự động (nếu cần).
- **Lên kế hoạch kiểm thử:** Còn được gọi bằng tên khác là lên chiến lược thử nghiệm. Ở giai đoạn này, trưởng nhóm kiểm thử sẽ dự toán chi phí cho dự án cũng như chuẩn bị kế hoạch kiểm thử.
 - *Hoạt động cụ thể:*
 - Lựa chọn công cụ kiểm thử (test tool).
 - Lên kế hoạch về nhân sự và ấn định vai trò trách nhiệm cho từng người trong nhóm.
 - Phổ biến cho mọi người trong nhóm kiểm thử về yêu cầu dự án.
 - *Tài liệu sử dụng:*
 - Bản 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.
 - *Hoạt động cụ thể:*
 - Tạo ca kiểm thử.
 - Xác minh, kiểm tra lại các ca kiểm thử.
 - Tạo dữ liệu kiểm thử.
 - *Tài liệu sử dụng:*
 - Ca kiểm thử.

- Dữ liệu kiểm thử.
- **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.
 - *Hoạt động cụ thể:*
 - Hiểu được kiến trúc yêu cầu, thiết lập môi trường và chuẩn bị danh sách yêu cầu về phần cứng và phần mềm cho môi trường thử nghiệm.
 - Thiết lập môi trường kiểm thử.
- **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 hiện ở giai đoạn này sẽ được thông báo lại cho nhóm phát triển phần mềm để chỉnh sửa và thực hiện kiểm thử lại.
 - *Hoạt động cụ thể:*
 - Thực hiện kiểm thử theo kế hoạch.
 - Làm tài liệu về kết quả kiểm thử, cập nhật lại các lỗi trong ca kiểm thử.
 - Kiểm thử lại các lỗi đã được chỉnh sửa.
 - Kiểm tra để đóng lỗi.
 - *Tài liệu sử dụng:*
 - Ca kiểm thử (cập nhật kết quả).
 - Báo cáo lỗi.
- **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.

➤ *Hoạt động cụ thể:*

- Đánh giá việc hoàn thành quy trình kiểm thử dựa vào thời gian, mức độ bao phủ, chi phí và chất lượng.
- Chuẩn bị dữ liệu dựa trên các tiêu chí trên.
- Chuẩn bị báo cáo kết thúc kiểm thử.
- Báo cáo chất lượng sản phẩm cho khách hàng.
- Phân tích kết quả kiểm thử để tìm ra sự phân bố lỗi theo loại và mức độ nghiêm trọng.

➤ *Tài liệu sử dụng:*

- Báo cáo kết thúc kiểm thử.

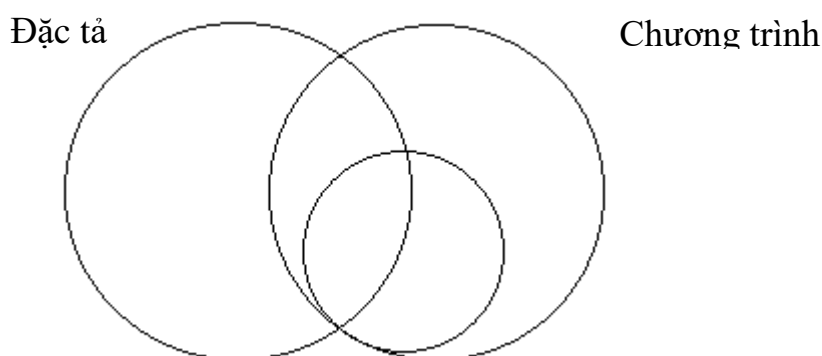
1.2.5. Phân loại kiểm thử phần mềm

Có 2 cách cơ bản để xác định các ca kiểm thử là kiểm thử tĩnh và kiểm thử động.

• **Kiểm thử tĩnh:** là một hình thức của kiểm thử phần mềm mà không cần thực thi chương trình. Điều này ngược với thử nghiệm động. Công việc chủ yếu là kiểm tra tính đúng đắn của mã lệnh, thuật toán hay tài liệu. Đây là loại kiểm thử được thực hiện bởi lập trình viên. Lỗi được phát hiện bằng kiểm thử tĩnh ít tốn kém để sửa chữa hơn so với lỗi phát hiện bằng kiểm thử động sẽ được đề cập dưới đây. Các lập trình viên có thể trao đổi mã nguồn chéo nhau hoặc làm việc một cách độc lập để thực hiện kiểm thử tĩnh.

• **Kiểm thử động:** Liên quan đến việc thực thi chương trình để phát hiện các lỗi, thất bại có thể có của chương trình hay tìm ra các vấn đề về hiệu năng hệ thống. Việc thực thi chương trình trên tất cả các dữ liệu đầu vào là không thể nên ta chỉ có thể chọn một tập con các dữ liệu đầu vào để thực thi hay nói cách khác là sinh ra các *ca kiểm thử*. Trong kiểm thử động, người ta chia làm 2 kỹ thuật: kiểm thử hộp trắng (kiểm thử cấu trúc) và kiểm thử hộp đen (kiểm thử chức năng).

- **Kiểm thử hộp trắng:** là kỹ thuật kiểm thử dựa vào thuật toán, cấu trúc mã nguồn bên trong của chương trình với mục đích đảm bảo rằng tất cả các câu lệnh và điều kiện sẽ được thực hiện ít nhất một lần. Người kiểm thử truy cập vào mã nguồn chương trình và kiểm tra nó, lấy đó làm cơ sở để thực hiện việc kiểm thử. Kiểm thử hộp trắng bao gồm các công việc cơ bản: Kiểm thử đường dẫn, kiểm thử luồng điều khiển, kiểm thử nội bộ (xác nhận các tham số, vòng lặp), kiểm thử tính năng (kiểm tra thời gian xử lý, dữ liệu cụ thể). Tuy nhiên, việc kiểm thử hộp trắng tồn tại khá nhiều hạn chế như: không thể đảm bảo rằng chương trình đã tuân theo đặc tả, khó phát hiện được các lỗi do dữ liệu, thiếu đường dẫn, v.v. Như vậy, không thể chỉ sử dụng kiểm thử hộp trắng để kiểm thử chương trình.



Hình 1-3: Xác định ca kiểm thử với kiểm thử hộp trắng.

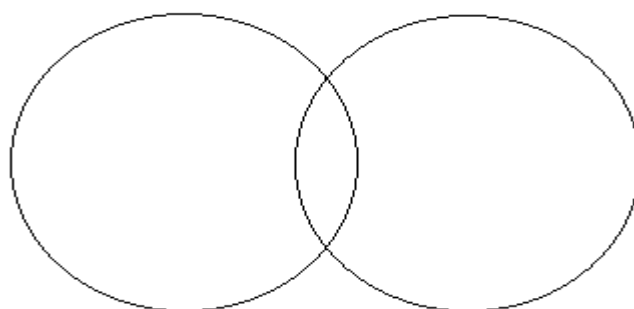
- **Kiểm thử hộp đen:** Là kỹ thuật kiểm thử dựa trên đầu vào và đầu ra của chương trình mà không quan tâm tới mã nguồn bên trong được viết ra sao. Với kỹ thuật này, kiểm thử viên xem phần mềm như là một hộp đen. Để thực hiện, kiểm thử viên sẽ xây dựng các nhóm giá trị đầu vào sao cho chúng có thể thực hiện đầy đủ các chức năng cần có của chương trình. Kiểm thử hộp đen sử dụng các phương pháp: phân tích giá trị biên, kiểm thử tính bền vững, kiểm thử trường hợp xấu nhất, kiểm thử phân lớp tương đương miền dữ liệu đầu vào, đầu ra, kiểm thử giá trị đặc biệt, kiểm thử dựa trên bảng quyết định. Tất cả

các phương pháp trên đều dựa trên thông tin xác định về các thành phần đang được kiểm thử.



Hình 1-4: Minh họa kỹ thuật kiểm thử hộp đen.

Cho tới nay, việc xác định kỹ thuật kiểm thử nào là tốt hơn trong 2 kỹ thuật: kiểm thử hộp đen và kiểm thử hộp trắng vẫn còn là một dấu hỏi lớn. Biểu đồ Venn sau đây sẽ giúp hình dung khái quát về mối liên hệ giữa kiểm thử hộp đen và kiểm thử hộp trắng trong thực tế kiểm thử hiện nay.



Kiểm thử hộp đen Kiểm thử hộp trắng

Hình 1-5: Biểu đồ Venn nguồn các ca kiểm thử.

Trước hết cần khẳng định mục đích của hai kỹ thuật trên đều là để xác định ca kiểm thử. Trong khi kiểm thử hộp trắng chỉ dùng đặc tả để xác định ca kiểm thử, thì kiểm thử hộp đen lại dùng mã nguồn chương trình để làm cơ sở xác định ca kiểm thử. Trong phần trình bày chi tiết về hai kỹ thuật đã nói ở trên đều cho thấy không có kỹ thuật nào là đủ tốt hoàn toàn. Cụ thể: Nếu tất cả các hành vi được nêu trong bản đặc tả yêu cầu phần mềm vẫn chưa được cài đặt thì kiểm thử hộp trắng sẽ không thể nhận biết được điều đó. Hay nếu các hành vi đã được cài đặt trong chương trình nhưng lại chưa có trong bản đặc tả yêu cầu phần mềm, kiểm thử hộp đen dường như bất lực trong trường

hợp này. Qua biểu đồ Venn, ta có thể khẳng định: việc kết hợp khéo léo cả hai kỹ thuật kiểm thử trên sẽ đem lại một kết quả tốt nhất trong kiểm thử phần mềm. Thực hiện song song kiểm thử hộp đen và kiểm thử hộp trắng sẽ hạn chế tối đa các khiếm khuyết có thể bị bỏ sót. Tuy nhiên, nếu biết được loại lỗi nào hay mắc phải hoặc những sai lầm thường thấy trong dự án đang được triển khai, ta hoàn toàn có thể chọn kỹ thuật kiểm thử thích hợp cho từng ca kiểm thử. Kinh nghiệm của kiểm thử viên sẽ được phát huy trong những trường hợp này.

1.2.6. 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 là 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. Hình 1.6 dưới đây minh họa các mức độ nghiêm trọng của lỗi dựa trên độ nghiêm trọng và hậu quả.

1	Nhẹ	Lỗi chính tả
2	Vừa	Hiểu lầm hoặc thừa thông tin
3	Khó chịu	Tên bị thiếu, cắt chữ hoặc hoá đơn có giá trị 0.0 đồng
4	Bực mình	Một vài giao dịch không được xử lý
5	Nghiêm trọng	Mất giao dịch
6	Rất nghiêm trọng	Xử lý giao dịch sai
7	Cực kỳ nghiêm trọng	Lỗi rất nghiêm trọng thường xuyên xảy ra
8	Quá quấy	Hủy hoại cơ sở dữ liệu
9	Thảm họa	Hệ thống dừng hoạt động
10	Dịch họa	Thảm họa chuyển sang mức lây lan

Hình 1-6: Bảng phân loại mức độ nghiêm trọng của lỗi.

1.2.7. 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, tiên 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*” [5]. Hay nói cách khác, ca kiểm thử mô tả dữ liệu bao gồm: đầu vào, hành động hoặc sự kiện và kết quả đầu ra mong đợi (expected results) để xác định liệu 1 ứng dụng, hệ thống phần mềm hoặc một trong các tính năng của nó có hoạt động đúng như mong muốn hay không.

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

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

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

Test Case #: 2.2	Test Case Name: Change PIN	Page: 1 of 1		
System: ATM	Subsystem: PIN			
Designed by: ABC	Design Date: 28/11/2004			
Executed by:	Execution Date:			
Short Description: Test the ATM Change PIN service				
Pre-conditions The user has a valid ATM card - The user has accessed the ATM by placing his ATM card in the machine The current PIN is 1234 The system displays the main menu				
Step	Action	Expected System Response	Pass/ Fail	Comment
1	Click the 'Change PIN' button	The system displays a message asking the user to enter the new PIN		
2	Enter '5555'	The system displays a message asking the user to confirm (re-enter) the new PIN		
3	Re-enter '5555'	The system displays a message of successful operation The system asks the user if he wants to perform other operations		
4	Click 'YES' button	The system displays the main menu		
5	Check post-condition 1			
Post-conditions 1. The new PIN '5555' is saved in the database				

Hình 1-7: Minh họa mẫu ca kiểm thử đơn giản.

Một ca kiểm thử được cho là hiệu quả khi:

- Dựa vào ca kiểm thử có thể tìm thấy lỗi.
- Tìm được nhiều lỗi khó phát hiện.
- Chỉ ra được những điểm ban đầu mà khi thực hiện kiểm thử không tìm ra vấn đề.
- Ca kiểm thử cần có những bước thực hiện kiểm thử (Test steps) đơn giản, minh bạch, dễ hiểu.
- Các trường hợp thử nghiệm nên có giá trị, tóm tắt và ngắn.
- Các ca kiểm thử nên có sự liên kết: Mỗi ca kiểm thử cần được đánh số thứ tự (Test case ID) để đảm bảo ca kiểm thử đã bao phủ 100% bản đặc tả yêu cầu phần mềm.

- Ca kiểm thử có thể bảo trì: Nên viết ca kiểm thử sao cho khi có thay đổi, chỉnh sửa thì các bên liên quan có thể dễ dàng nhận thấy được sự thay đổi đó.
- Ca kiểm thử có tính ứng dụng cao.

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

1.2.8. Kiểm thử tự động

Kiểm thử tự động là quá trình kiểm tra một hệ thống nào đó bằng các công cụ tự động hoá với dữ liệu đầu vào và đầu ra đã được xác định.

Công việc kiểm thử thường chiếm từ 11% - 40% chi phí cho quá trình phát triển phần mềm [6]. Hơn nữa, các dự án phần mềm đều mong muốn giảm chi phí về thời gian, nhân lực mà vẫn đem lại hiệu quả cao, chất lượng tốt. Đó chính là lý do kiểm thử tự động được áp dụng rộng rãi trong các quy trình phát triển phần mềm ngày nay.

Kiểm thử tự động đặc biệt phát huy tác dụng trong các trường hợp kiểm thử lặp đi lặp lại, kiểm thử hồi quy hay các ca kiểm thử có giá trị dữ liệu đầu vào rất lớn khiến cho việc kiểm thử thủ công gặp nhiều khó khăn. Đối với các trường hợp kiểm thử lặp đi lặp lại, nếu thực hiện thủ công sẽ gây ra sự nhàm chán cho người kiểm thử, dẫn tới năng suất lao động kém. Đó là chưa kể tới việc lặp đi lặp lại quy trình một cách thủ công hoàn toàn có thể dẫn tới sai sót. Ngược lại, nếu thay bằng kiểm thử tự động, dù có lặp đi lặp lại bao nhiêu lần thì cũng cho ra thao tác và kết quả chính xác. Điều này giúp chúng ta tránh được những rủi ro không đáng có và giảm đáng kể thời gian cho việc kiểm thử.

Một số công cụ kiểm thử tự động phổ biến hiện nay:

- *Selenium*: Công cụ kiểm thử phần mềm mã nguồn mở hỗ trợ kiểm thử tự động cho các ứng dụng Web. Selenium cung cấp chức năng ghi tự động và phát lại, hỗ trợ hữu ích cho kiểm thử hồi quy. Điểm mạnh của Selenium là hỗ trợ nhiều nền tảng khác nhau, tích hợp vào các trình duyệt phổ biến hiện nay, có thể thực hiện nhiều ca kiểm thử cùng lúc, có khả năng lưu các ca kiểm thử để sử dụng lại khi cần và cho phép người dùng chèn chú thích ở giữa kịch bản kiểm thử để hiểu rõ hơn nội dung kiểm thử. Selenium cũng hỗ trợ một lượng lớn các ngôn ngữ lập trình Web phổ biến hiện nay như C#, Java, Perl, PHP, Python, Ruby, v.v. Selenium có thể kết hợp với một số công cụ khác như Bromien và Junit nhưng với người dùng thông thường chỉ cần chạy tự động mà không cần cài thêm các công cụ hỗ trợ. Selenium hiện nay đang được cộng đồng đánh giá là một trong những công cụ tốt nhất cho kiểm thử tự động các ứng dụng Web.
- *QTP (HP UFT)*: Được sử dụng rộng rãi để kiểm thử chức năng và hồi quy. QTP sử dụng khái niệm kiểm thử từ khoá để đơn giản hoá việc tạo và bảo trì ca kiểm thử. Công cụ này hỗ trợ môi trường .NET và có cơ chế xác định đối tượng kiểm thử tốt. Đối với những kiểm thử viên không theo ngành kỹ thuật vẫn có thể sử dụng dễ dàng công cụ này.
- *Rational Function Tester*: Là 1 công cụ kiểm tra tự động hướng đối tượng có khả năng tự động kiểm tra dữ liệu, kiểm tra giao diện, và kiểm thử hồi quy. Rational Function Tester hỗ trợ rất nhiều giao thức và ứng dụng như Java, HTML, .NET, Windows, Visual Basic, v.v. Công cụ này cũng hỗ trợ ghi và phát lại các hành động theo yêu cầu. Nó cho phép các nhà phát triển tạo ra các kịch bản liên quan đến từ khoá để có thể được tái sử dụng. Bộ biên

tập Công cụ Java Developer Toolkit của Eclipse tạo điều kiện cho kiểm thử viên tạo mã thử nghiệm các đoạn mã trong Java với Eclipse.

- *WATIR*: Là một phần mềm kiểm tra mã nguồn mở để kiểm thử hồi quy. Watir hỗ trợ nhiều trình duyệt trên nhiều nền tảng khác nhau. Đồng thời cũng sử dụng một ngôn ngữ kịch bản hiện đại có đầy đủ các tính năng. Điểm mạnh của Watir là hỗ trợ ứng dụng Web được viết bởi bất kỳ ngôn ngữ nào.
- *SilkTest*: Silk Test được thiết kế để thực hiện kiểm thử chức năng và hồi quy. Nó là một ngôn ngữ hướng đối tượng giống như C++. SilkTest sử dụng các khái niệm về đối tượng, các class và sự kế thừa. Nó chuyển đổi các lệnh script thành các lệnh GUI. Trên cùng một máy, các lệnh có thể được chạy trên một máy từ xa hoặc máy chủ. Silktest có thể xác định chuyển động của con chuột cùng với các phím bấm. Nó có thể sử dụng cả phương pháp ghi và phát lại hoặc các phương pháp lập trình mô tả.

Dù có rất nhiều ưu điểm về mặt thời gian thực thi nhưng kiểm thử tự động cũng không thể thay thế hoàn toàn quá trình kiểm thử của con người. Để thực hiện kiểm thử tự động, trước hết vẫn cần bàn tay con người thiết lập thao tác cho công cụ hay các đoạn kịch bản máy tính để thực thi. Đối với những ca kiểm thử chỉ thực hiện số ít lần thì việc mất thời gian tạo kịch bản kiểm thử tự động là không cần thiết. Chưa kể tới những ca kiểm thử với đặc thù riêng biệt mà kiểm thử tự động không làm được. Thêm vào đó, không phải công cụ kiểm thử tự động nào cũng miễn phí và dễ sử dụng hay đưa vào triển khai rộng rãi.

1.2.9. Nguyên tắc quan trọng trong kiểm thử phần mềm

Có thể hiểu *nguyên tắc* là những quy định mà chúng ta phải tuân theo. Trong kiểm thử phần mềm, việc theo đuổi những nguyên tắc là điều cần thiết giúp chúng ta phát triển hệ thống một cách tốt nhất. Người ta đưa ra 7 nguyên

tắc kiểm thử quan trọng có thể dựa vào chúng để tiết kiệm thời gian, công sức và chi phí phát triển [5]. Cụ thể:

- 1) *Kiểm thử chỉ ra lỗi*: Kiểm thử có thể phát hiện ra lỗi của phần mềm, nhưng lại không thể chứng minh phần mềm hoàn toàn không có lỗi. Thực tế cho thấy, ngay cả khi kiểm thử một cách nghiêm ngặt phần mềm vẫn có thể xuất hiện lỗi. Vì vậy, cần phải tìm được ra càng nhiều lỗi càng tốt.
- 2) *Kiểm thử cạn kiệt là không thể*: Đó chính là việc chúng ta không thể kiểm tra với mọi dữ liệu đầu vào, đầu ra với toàn bộ các kịch bản của một phần mềm. Hay nói cách khác, kiểm tra mọi thứ trong chương trình một cách trọn vẹn là điều không thể làm được. Tuy nhiên, có thể phân tích rủi ro và dựa trên mức độ ưu tiên thông qua các mức độ nghiêm trọng của lỗi phần mềm để kiểm thử một số chức năng chính, thiết yếu của phần mềm.
- 3) *Kiểm thử càng sớm càng tốt*: Chi phí cho việc khắc phục, sửa lỗi sẽ tỷ lệ thuận với thời gian phát hiện ra lỗi đó. Không ít phần mềm khi chuẩn bị giao cho khách hàng mới phát hiện ra lỗi xuất hiện ngay từ bản đặc tả yêu cầu phần mềm hay bản phân tích thiết kế hệ thống. Điều này gây ra những thiệt hại không nhỏ cho quá trình phát triển phần mềm. Vì vậy, kiểm thử phần mềm ngay từ những giai đoạn đầu của quy trình phát triển là điều hết sức cần thiết.
- 4) *Xác định vị trí tập trung lỗi*: Lỗi thường tập trung nhiều ở những chức năng chính của phần mềm. Do đó, tập trung tìm ra lỗi ở những chức năng này sẽ giúp giảm thiểu thời gian kiểm thử. Đây là một trong những cách cơ bản và hiệu quả nhất trong kiểm thử phần mềm.
- 5) *Nghịch lý thuốc trừ sâu*: Dữ liệu của một hệ thống thay đổi liên tục và thường xuyên xuất hiện những hành vi mới từ phía người dùng. Nó giống như việc sử dụng một loại thuốc trừ sâu trong nhiều mùa vụ mà không để ý tới sự phát triển đa dạng của sâu bọ. Vì vậy, nếu giữ

nguyên ca kiểm thử cũ trong thời gian dài, chúng ta không thể tìm ra lỗi mới phát sinh. Việc xem xét và thay đổi các ca kiểm thử thường xuyên là điều cần thiết.

6) *Kiểm thử phụ thuộc vào ngữ cảnh*: Nguyên tắc này đề cập tới việc tiếp cận kiểm thử theo nhiều ngữ cảnh khác nhau. Diễn hình như việc kiểm thử ứng dụng Web và kiểm thử trên thiết bị di động không thể dùng chung một chiến lược kiểm thử. Mỗi môi trường kiểm thử sẽ có những đặc trưng riêng và đó là lý do tại sao cần xác định việc kiểm thử theo ngữ cảnh cho phù hợp.

7) *Phần mềm có lỗi bằng 0*: Việc không tìm thấy lỗi không có nghĩa là không tồn tại lỗi trong phần mềm. Chúng ta chỉ có thể hạn chế tối đa lỗi có thể gặp phải chứ không thể triệt tiêu toàn bộ lỗi có thể gặp phải.

Dựa theo 7 nguyên tắc trên đây giúp chúng ta có cái nhìn tổng quát về kiểm thử phần mềm cũng như đánh giá được tính hiệu quả của hoạt động kiểm thử đang triển khai.

1.3.Các kỹ thuật xác định ca kiểm thử

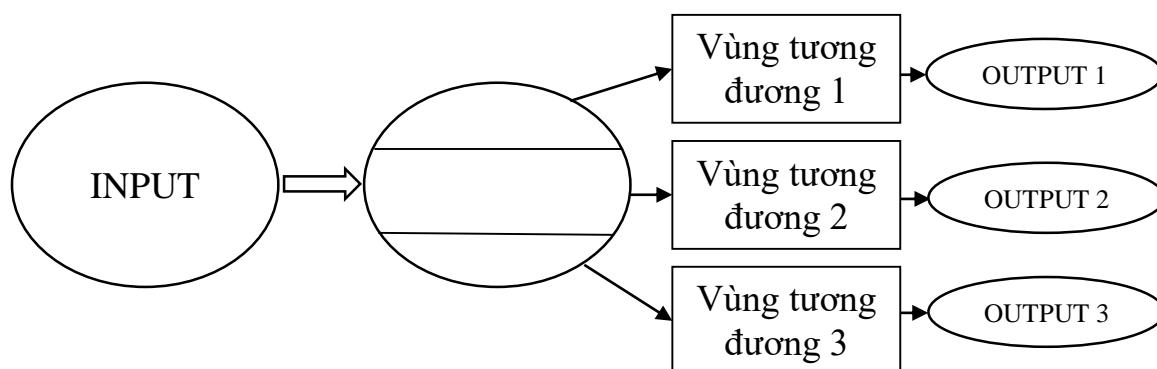
Trong quá trình kiểm thử phần mềm sẽ nảy sinh vô số trường hợp cần phải xét tới. Tuy nhiên, vì yếu tố chi phí, thời gian phát triển dự án nên người kiểm thử không thể tiến hành kiểm thử hết toàn bộ các giá trị đầu vào (Input). Lúc này, việc xác định tập các ca kiểm thử đặc trưng sẽ xây dựng sao cho có thể bao phủ được tối đa các trường hợp là điều vô cùng cần thiết. Phần này của đề án sẽ đề cập tới một số kỹ thuật xác định ca kiểm thử nhằm giải quyết vấn đề kể trên.

1.3.1. Kỹ thuật phân vùng tương đương

Kỹ thuật phân vùng tương đương có đặc điểm là:

- Chia miền dữ liệu đầu vào của một chương trình thành các vùng dữ liệu tương đương nhau.

- Tất cả các giá trị trong một vùng tương đương sẽ cho ra kết quả đầu ra giống nhau.
- Có thể chọn ra một giá trị đại diện trong một vùng tương đương để tiến hành kiểm thử.



Hình 1-8: Minh họa kỹ thuật phân vùng tương đương

Việc thiết kế ca kiểm thử bằng kỹ thuật phân lớp tương đương dựa trên nguyên tắc xác định số vùng tương đương hợp lệ và số vùng tương đương không hợp lệ.

Ví dụ, trường hợp kiểm thử một ô textbox chỉ cho phép nhập vào số ký tự trong khoảng [5 - 30]. Áp dụng nguyên tắc xác định số vùng tương đương ta sẽ có các ca kiểm thử sau:

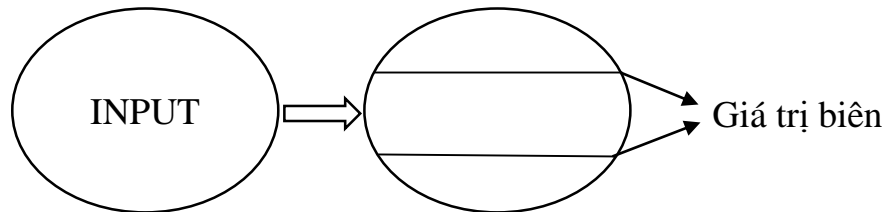
- Nhập vào một giá trị trong vùng tương đương không hợp lệ thứ nhất: Nhập 4 ký tự.
- Nhập vào một giá trị trong vùng tương đương hợp lệ: Nhập 6 ký tự.
- Nhập vào một giá trị trong vùng tương đương không hợp lệ thứ hai: Nhập 31 ký tự.

Như vậy với kỹ thuật trên, kiểm thử viên đã rút ngắn được số ca kiểm thử cần sinh ra so với việc phải kiểm thử toàn bộ các giá trị đầu vào.

1.3.2. Kỹ thuật phân tích giá trị biên

Phân tích giá trị biên tập trung vào các giá trị tại biên của miền xác định để xây dựng ca kiểm thử. Mục đích là tìm ra lỗi có thể xảy ra ở gần các giá trị

biên này. Phân tích giá trị biên chính là trường hợp đặc biệt của kỹ thuật phân vùng tương đương. Dựa trên những vùng giá trị tương đương, kiểm thử viên sẽ xác định giá trị biên giữa những vùng này và thiết kế ca kiểm thử phù hợp.



Hình 1-9: Minh họa kỹ thuật phân tích giá trị biên.

Với kỹ thuật phân tích giá trị biên, kiểm thử viên cần chú ý tới một số giá trị sau để sinh ca kiểm thử:

- Giá trị nhỏ nhất.
- Giá trị gần kề lớn hơn giá trị nhỏ nhất.
- Giá trị gần kề nhỏ hơn giá trị nhỏ nhất.
- Giá trị bình thường.
- Giá trị gần kề nhỏ hơn giá trị lớn nhất
- Giá trị lớn nhất.
- Giá trị gần kề lớn hơn giá trị lớn nhất

Ví dụ: Kiểm thử một textbox nhập tuổi cho phép nhập giá trị số trong khoảng [0 - 150]. Vậy có thể sinh ra các ca kiểm thử cho trường hợp này theo kỹ thuật phân tích giá trị biên như sau:

- Giá trị nhỏ nhất: 0
- Giá trị gần kề lớn hơn giá trị nhỏ nhất: 1
- Giá trị gần kề lớn hơn giá trị nhỏ nhất: -1
- Giá trị bình thường: 70
- Giá trị gần kề nhỏ hơn giá trị lớn nhất: 149
- Giá trị lớn nhất: 150
- Giá trị gần kề lớn hơn giá trị lớn nhất: 151

Như vậy, có thể thấy phân tích giá trị biên là kỹ thuật bổ sung cho kỹ thuật phân vùng tương đương, giúp kiểm thử viên sinh ca kiểm thử để kiểm tra các giá trị tại biên.

1.3.3. Đoán lỗi

Một kỹ thuật thiết kế ca kiểm thử khác là *đoán lỗi*. Kiểm thử viên phỏng đoán lỗi dựa trên trực giác và kinh nghiệm của mình, từ đó liệt kê các trường hợp có thể xảy ra lỗi và sinh ca kiểm thử. Khó có thể đưa ra một quy trình cho kỹ thuật kiểm thử đoán lỗi vì nó có tính trực giác cao và không thể dự đoán trước.

Trong một số trường hợp, kiểm thử viên có thể kết hợp với lập trình viên để tìm ra những trường hợp có thể bị bỏ sót trong quá trình viết đặc tả yêu cầu phần mềm và lập trình.

1.3.4. Kỹ thuật chuyển trạng thái

Kỹ thuật này dựa trên việc quan sát, theo dõi quá trình chuyển từ trạng thái này sang trạng thái khác khi có một hành động xảy ra trong chương trình phần mềm để phát hiện các lỗi có thể xảy ra mà các kỹ thuật trên có thể bỏ sót. Ví dụ điển hình cho kỹ thuật chuyển trạng thái là việc kiểm thử chức năng giỏ hàng trong các trang Web thương mại điện tử. Lỗi có thể xuất hiện mỗi khi thêm sản phẩm vào giỏ hàng, xóa sản phẩm khỏi giỏ hàng hay khi thanh toán các sản phẩm trong giỏ hàng đó. Công việc của kiểm thử viên là xem xét các điều kiện trạng thái, theo dõi quá trình chuyển đổi giữa các trạng thái, điều kiện nhập đầu vào và các sự kiện kích hoạt thay đổi trạng thái.

1.4. Kết luận

Chương 1 đã trình bày những khái niệm để có cái nhìn tổng quát về những vấn đề cơ bản xoay quanh phần mềm và kiểm thử phần mềm. Các vấn đề cụ thể bao gồm:

- Các định nghĩa về phần mềm, kiểm thử phần mềm.
- Vai trò của kiểm thử trong quá trình phát triển dự án phần mềm.

- Các cấp độ trong kiểm thử phần mềm.
- Quy trình kiểm thử phần mềm.
- Phân loại kiểm thử phần mềm.
- Liệt kê các mức độ nghiêm trọng của lỗi.
- Tổng quan về ca kiểm thử.
- Vai trò của kiểm thử tự động trong kiểm thử phần mềm hiện nay.
- Trình bày một số nguyên tắc quan trọng trong kiểm thử phần mềm.
- Liệt kê các kỹ thuật xác định ca kiểm thử.

CHƯƠNG 2: KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB

Kiểm thử ứng dụng trên nền Web là một lĩnh vực phổ biến trong ngành kiểm thử phần mềm. Chương thứ 2 của đề án sẽ đi sâu vào tìm hiểu khái niệm, các công việc cụ thể khi kiểm thử ứng dụng trên nền Web. Đồng thời, trong chương này cũng sẽ giới thiệu một số công cụ hỗ trợ kiểm thử ứng dụng Web.

2.1. Khái quát về kiểm thử ứng dụng trên nền Web

2.1.1. Khái quát

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

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

2.1.2. 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

khiến 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. Một số ví dụ cho Web tĩnh là những trang giới thiệu công ty, tổ chức, sản phẩm, khoá học ít có nhu cầu cập nhật thông tin, v.v.

- **Ứng dụng Web động:** So với web tĩnh thì Web động phức tạp hơn về mặt kỹ thuật khi xây dựng. Web động sử dụng cơ sở dữ liệu để hiển thị nội dung cũng như cho phép người dùng tương tác được với nội dung đó. Web động được chia làm 2 phần là back-end (dành cho người quản trị Web thay đổi, cập nhật nội dung) và front-end (dành cho người dùng truy cập). Hiện nay có rất nhiều ngôn ngữ lập trình được sử dụng để xây dựng Web động như Java, PHP, ASP.NET, VB.NET, Ruby, v.v. Đối với Web động, việc cập nhật nội dung là rất đơn giản và dễ dàng. Không những thế, một số hệ thống lớn hiện nay còn cho phép người quản trị có thể thay đổi giao diện Web trên trang quản trị mà không cần phải can thiệp trực tiếp vào mã nguồn. Đó là những lý do khiến cho Web động được sử dụng phổ biến hơn Web tĩnh.

2.1.3. 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, v.v. kéo theo việc kiểm thử trên từng loại ứng dụng sẽ khác nhau. Chính vì vậy, một chuyên gia về kiểm thử ứng dụng trên điện

thoại thông minh hay ứng dụng desktop chưa chắc đã làm tốt công việc kiểm thử với ứng dụng trên nền Web. Những sự khác biệt có thể kể đến sau đây:

- *Ứng dụng trên nền Web sử dụng trên nhiều trình duyệt, không biết trước môi trường duyệt Web của người dùng:* Một ứng dụng Web chạy tốt trên trình duyệt Google Chrome nhưng trên Mozilla Firefox hay Safari thì có thể không như ý muốn. Đó là do mỗi trình duyệt được xây dựng trên kiến trúc khác nhau. Ngay cả khi hiện tại các trình duyệt đều đang cố gắng đưa ra chuẩn chung để dễ dàng hơn cho người lập trình, nhưng sự khác biệt khi khởi chạy ứng dụng trên nhiều trình duyệt khác nhau vẫn gây ra nhiều lo lắng cho lập trình viên và người làm kiểm thử. Đó là lý do chúng ta không khó bắt gặp những ứng dụng chạy trên nền Web ghi chú thích “Website chạy (tương thích) tốt nhất trên trình duyệt X v.v.”. Tuy nhiên, cách làm này không thật sự hiệu quả khi người dùng muốn sử dụng ứng dụng Web của chúng ta lại phải cài đặt trình duyệt được khuyến nghị. Để tránh cho sự bất tiện này đòi hỏi người làm kiểm thử phải triển khai ca kiểm thử trên nhiều trình duyệt khác nhau, kiểm tra độ tương thích và tìm ra những lỗi để lập trình viên đưa ra sự thay đổi cho phù hợp với mọi trình duyệt.
- *Ứng dụng trên nền Web thường có lượng truy cập lớn, nhiều người sử dụng trên cùng một thời điểm:* Với những ứng dụng Web có lượng người truy cập trung bình hoặc ít thì điều này không xảy ra vấn đề gì nghiêm trọng. Nhưng với những ứng dụng chạy trên nền Web có lượng người truy cập lớn, thực hiện nhiều thao tác truy vấn dữ liệu cùng lúc có thể sẽ dẫn tới việc server bị quá tải. Kiểm thử hộp trắng phát huy hiệu quả rất cao trong trường hợp này. Việc kiểm thử mã nguồn chương trình sẽ giúp loại bỏ được những dòng lệnh không hợp lý, gây tiêu tốn tài nguyên hệ thống và giúp cho ứng dụng Web có thể đáp ứng được lượng truy cập lớn cùng lúc tốt hơn. Công việc này cũng chính là kiểm thử hiệu năng, độ chịu lỗi của chương trình phần mềm.

- *Sự phụ thuộc vào tốc độ và sự ổn định của đường truyền Internet:* Đa số các ứng dụng Web đều cần sử dụng mạng Internet để tải các dữ liệu về, sau đó hiển thị lên trình duyệt. Nếu tốc độ đường truyền ổn định, việc duyệt Web không gây khó khăn gì. Tuy nhiên trên thực tế, tốc độ cũng như sự ổn định về đường truyền của người dùng là rất khó đoán biết, mỗi khu vực lại có sự khác nhau về đường truyền gây ảnh hưởng tới sự vận hành của ứng dụng Web. Chưa kể tới việc mạng có thể mất kết nối đột ngột khi đang thực hiện thao tác truy vấn sẽ dẫn tới những hậu quả rất khó lường nếu kiểm thử không tốt ở các trường hợp này, điển hình như các ứng dụng cho ngân hàng, hệ thống ERP, phần mềm phục vụ kế toán, v.v.
- *Sự cần thiết của SEO Web:* Đối với rất nhiều ứng dụng trên nền Web việc tối ưu SEO là một yêu cầu bắt buộc. Người sở hữu các website đều muốn website được thăng thứ hạng cao trên các máy tìm kiếm như Google, Bing, v.v. giúp ứng dụng Web của mình được nhiều người biết tới. Đây là một điểm mạnh giúp quảng bá ứng dụng trên nền Web dễ dàng hơn so với ứng dụng di động hay ứng dụng desktop. Trong thực 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 tới chất lượng nội dung của ứng dụng Web để được máy tìm kiếm chú ý đến.

2.2. Công việc chính khi kiểm thử ứng dụng Web

2.2.1. Kiểm thử chức năng

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

- *Kiểm thử giao diện:*

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

- *Kiểm thử các liên kết và menu:*

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

- *Kiểm thử các form nhập dữ liệu:*

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

- *Kiểm thử lỗi cú pháp HTML/CSS:*

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

- *Kiểm thử cookie và session:*

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

- *Kiểm thử nội dung đa ngôn ngữ:*

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

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

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

2.2.2. 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 hoá khu vực và đối tượng sử dụng:*

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

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

2.2.3 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.

2.2.4. 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.

2.2.5. 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).

2.3. Một số công cụ hỗ trợ kiểm thử ứng dụng trên nền Web

Công việc cần làm đối với một kiểm thử viên kiểm thử ứng dụng trên nền Web như đã nói ở phần trước là rất nhiều. Những công cụ kiểm thử ra đời để hỗ trợ cho các kiểm thử viên thực hiện công việc một cách nhanh chóng, bớt nhàm chán và giảm thiểu chi phí kiểm thử. Đề 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.

2.3.1. 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.

2.3.2. 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 Inject-Me được sử dụng để kiểm tra các lỗ hổng SQL Injection trong ứng dụng Web.

2.3.3. Công cụ kiểm thử chức năng

- **BrowserStack:** Đây là một công cụ giúp kiểm thử hoạt động của các chức năng trên ứng dụng Web trên nhiều trình duyệt khác nhau. Ứng dụng Web có thể được kiểm tra bằng thao tác của người dùng hoặc tự động thông qua Selenium. Ngoài ra, BrowserStack còn cung cấp tính năng chụp ảnh màn hình ứng dụng Web trên 650 trình duyệt khác nhau và kiểm tra khả năng hiển thị responsive trên các loại màn hình.

- **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.

2.4. Kết luận

Chương 2 của đề án đã trình bày được các vấn đề cơ bản về kiểm thử ứng dụng trên nền Web:

- Khái quát về kiểm thử ứng dụng trên nền Web.
- Tìm hiểu các công việc cần làm khi kiểm thử một ứng dụng Web.
- Giới thiệu một số công cụ hỗ trợ kiểm thử ứng dụng Web, trong đó có **Selenium** – một công cụ kiểm thử phổ biến hiện nay. **Selenium** cũng sẽ là bộ công cụ được em sử dụng để nghiên cứu sâu hơn về cách cài đặt, sử dụng công cụ hỗ trợ cho việc kiểm thử ứng dụng Web.

CHƯƠNG 3: KIỂM THỬ ỨNG DỤNG TRÊN NỀN WEB BẰNG CÔNG CỤ SELENIUM

Chương cuối của đồ án tập trung nghiên cứu về bộ công cụ hỗ trợ kiểm thử Selenium và một số công cụ liên quan khác phục vụ đắc lực cho công việc kiểm thử tự động ứng dụng trên nền Web. Cụ thể sẽ đi sâu vào tìm hiểu hướng dẫn cài đặt, thao tác sử dụng cơ bản kèm theo một số lệnh thông dụng của công cụ Selenium IDE trên trình duyệt Firefox. Ngoài ra chương này cũng đề cập tới việc phối hợp sử dụng Selenium IDE với một số công cụ khác như Monosnap, Firebug, Mantis trong những ca kiểm thử thực tế.

3.1. Công cụ kiểm thử tự động Selenium

3.1.1. Giới thiệu chung về Selenium

Selenium là một phần mềm mã nguồn mở - một công cụ kiểm thử phần mềm tự động để kiểm thử các ứng dụng trên nền Web. Năm 2004, Selenium được phát triển bởi ThoughtWorks với cái tên ban đầu JavaScriptTestRunner. Đến năm 2007, tác giả Jason Huggins rời ThoughtWorks và gia nhập Selenium Team (thuộc Google), từ đó tiếp tục phát triển Selenium như hiện nay.

Selenium không chỉ là một công cụ duy nhất mà là một bộ các công cụ giúp kiểm thử tự động các ứng dụng trên nền Web hiệu quả hơn, bao gồm 4 phần: Selenium IDE, Selenium RC, Selenium Grid, Selenium WebDriver.

Selenium IDE là một công cụ cho phép chúng ta ghi lại một kịch bản và tái sử dụng kịch bản đó. Nó hoạt động như một Add-on của trình duyệt Mozilla Firefox với giao diện trực quan, dễ sử dụng ngay cả với những kiểm thử viên không biết về code. Với Selenium IDE, chúng ta chỉ có thể ghi lại kịch bản (Record) trên trình duyệt Mozilla Firefox nhưng có thể tái sử dụng kịch bản này trên nhiều trình duyệt khác như Internet Explorer, Google Chrome, v.v.

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

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

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

Do thời gian nghiên cứu hạn chế nên đồ án chỉ trình bày về một phần của bộ công cụ Selenium là **Selenium IDE**. Đây cũng là công cụ được sử dụng phổ biến hiện nay trong việc kiểm thử các dự án phát triển ứng dụng Web bởi giao diện trực quan, dễ sử dụng và khả năng ghi/phát lại các ca kiểm thử một cách linh động của nó.

3.1.2. Selenium IDE

3.1.2.1. Giới thiệu

Selenium IDE là một tiện ích (extension) dùng để hỗ trợ kiểm thử tự động chạy trên trình duyệt Mozilla Firefox ban đầu được phát triển bởi

Shinya Kasatani người Nhật Bản. Công cụ này được phát triển bằng Javascript nên có khả năng tương tác với DOM, sử dụng được những cách gọi Javascript.

Selenium IDE cho phép ghi lại những hành động, sự kiện xảy ra trong quá trình kiểm thử bằng chức năng Record (ghi hành động) và Playback (phát lại).

Một số ưu điểm của Selenium IDE:

- Dễ dàng cài đặt và sử dụng.
- Không yêu cầu kinh nghiệm lập trình.
- Có thể debug, thiết lập breakpoint (điểm dừng), thêm comment vào kịch bản kiểm thử.

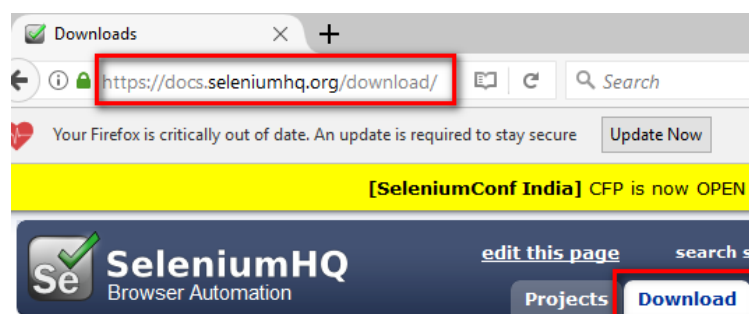
Bên cạnh đó Selenium IDE cũng tồn tại một số nhược điểm:

- Chỉ hỗ trợ chạy trên Mozilla Firefox (cần cấu hình để chạy được trên Chrome, Internet Explorer).
- Cần cài đặt thêm để đọc dữ liệu từ file .csv, .xml.
- Không hỗ trợ database, kiểm thử di động (mobile testing).
- Khó khăn khi xử lý những quy trình phức tạp.

3.1.2.2. Hướng dẫn cài đặt Selenium IDE

Đầu tiên người dùng nên kiểm tra xem trình duyệt Mozilla Firefox đã được cài đặt Selenium IDE hay chưa bằng cách mở trình duyệt này lên, chọn menu Tools. Nếu trong danh sách menu con không chứa Selenium IDE thì cần thực hiện những bước sau để tiến hành cài đặt.

- **Bước 1:** Tiến hành truy cập vào địa chỉ <http://docs.seleniumhq.org/>. Sau đó chọn menu *Download*.



Hình 3-1: Truy cập trang Download tiện ích Selenium IDE.

- **Bước 2:** Tìm tới mục Selenium IDE và chọn phiên bản mới nhất (latest released) hoặc những bản cũ hơn tùy theo nhu cầu sử dụng. Ở đây chọn phiên bản mới nhất Selenium IDE 2.9.1.

SafariDriver - DEPRECATED - use Apple's SafariDriver with Safari 10+

SafariDriver now requires manual installation of the extension prior to automation

- Latest release [2.48.0](#)
- [Wiki Page](#)

Selenium IDE

Selenium IDE is a Firefox plugin which records and plays back user interactions with the browser. Use this to either create simple scripts or assist in exploratory testing. It can also export Remote Control or WebDriver scripts, though they tend to be somewhat brittle and should be overhauled into some sort of Page Object-y structure for any kind of resiliency.

Download latest released version [from addons.mozilla.org](#) or view the [Release Notes](#) and then [install some plugins](#).

Download [previous versions here](#).

Hình 3-2: Tiến hành chọn phiên bản Selenium IDE để cài đặt.

- **Bước 3:** Tại website mới được mở ra, chọn “Add to Firefox” để thêm tiện ích Selenium IDE vào Firefox.

Selenium IDE

by [Jason Huggins](#), [Adam Goucher](#), [Shinya Kasatani](#), [Dave Hunt](#), [Samit Badle](#)

Restart Required  Experimental  Not compatible with Firefox Quantum 

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests.

[+ Add to Firefox](#)

Hình 3-3: Thêm tiện ích Selenium IDE vào Firefox.

- **Bước 4:** Tiếp tục chọn “Add” để xác nhận thao tác thêm tiện ích Selenium IDE vào trình duyệt.



addons.mozilla.org

This site would like to install an add-on in Firefox:

Selenium IDE

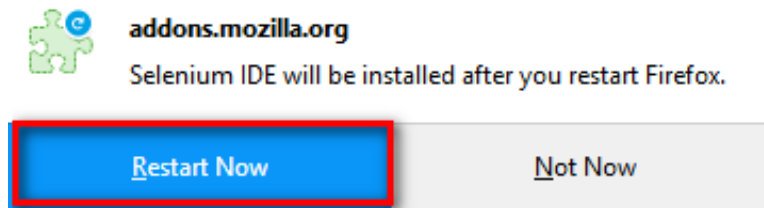
[Learn more...](#)

[Add](#)

[Cancel](#)

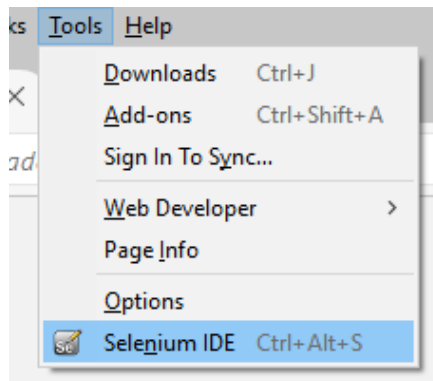
Hình 3-4: Xác nhận cài đặt tiện ích Selenium IDE vào trình duyệt.

- **Bước 5:** Khởi động lại trình duyệt để quá trình cài đặt hoàn tất.



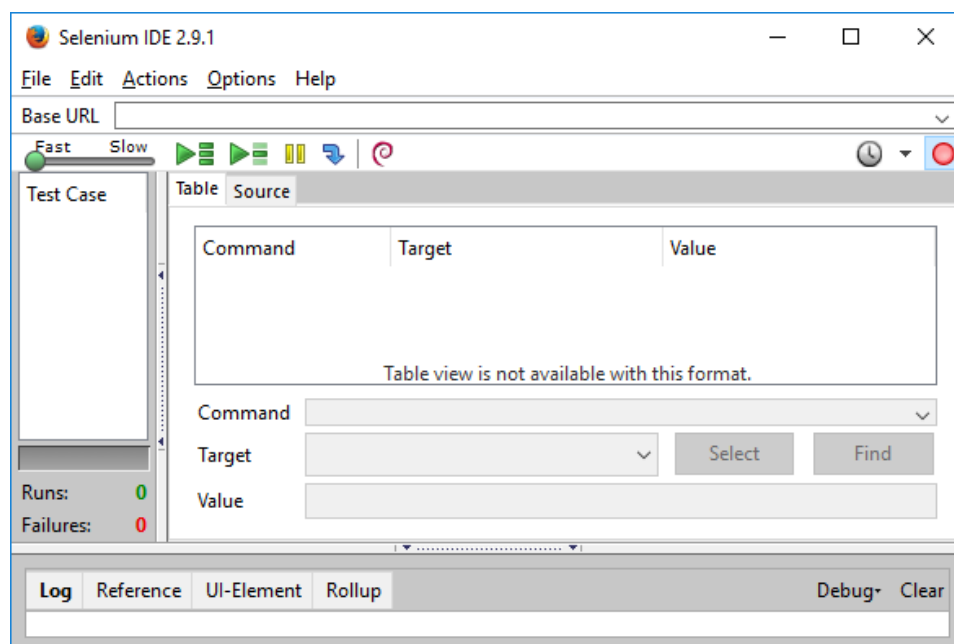
Hình 3-5: Khởi động lại trình duyệt Firefox để hoàn tất quá trình cài đặt.

Sau khi trình duyệt khởi động lại, tiến hành khởi chạy lần đầu để chắc chắn tiện ích Selenium IDE đã được cài đặt thành công bằng cách chọn menu *Tools*, tiếp tục chọn *Selenium IDE*.



Hình 3-6: Khởi chạy tiện ích Selenium IDE.

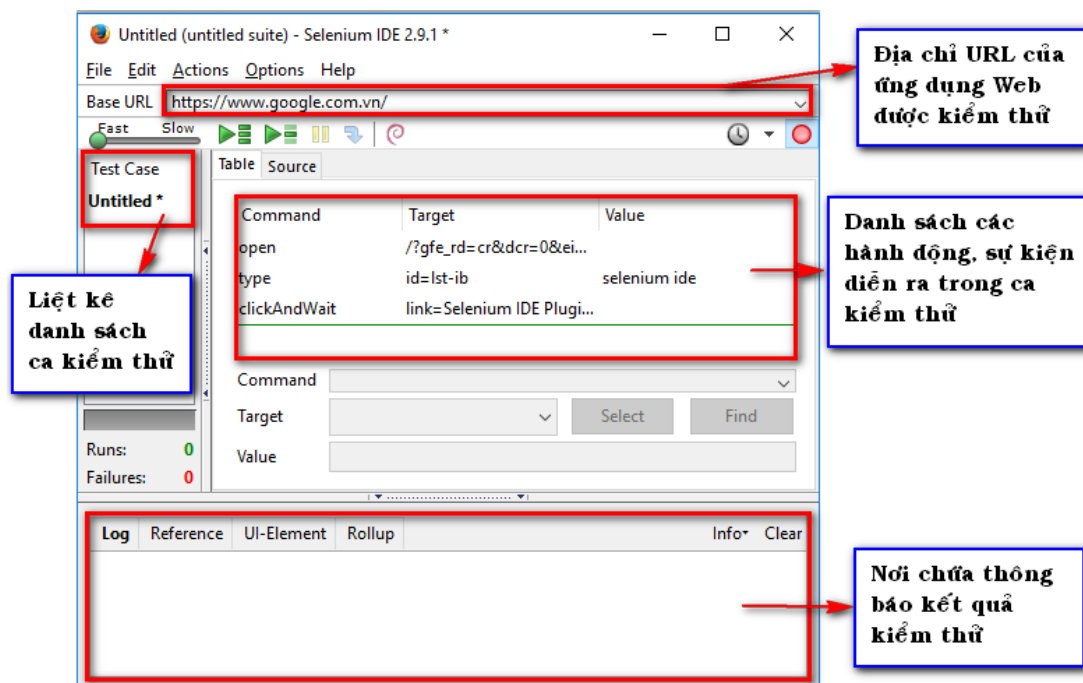
Cửa sổ tiện ích Selenium IDE được mở ra như Hình 3-7:



Hình 3-7: Giao diện khởi chạy Selenium IDE.

3.1.2.3. Một số chức năng trong Selenium IDE

Phần này của đề án sẽ giải thích và hướng dẫn cách sử dụng một số chức năng, ký hiệu trong tiện ích Selenium IDE.

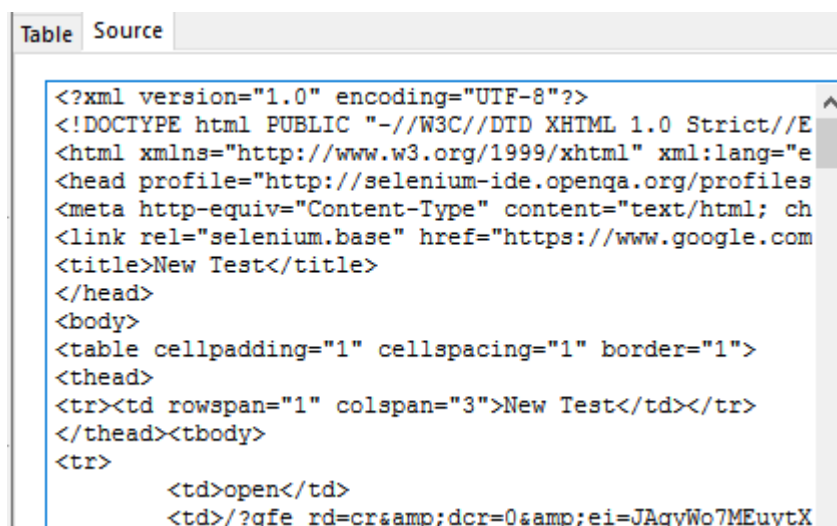


Hình 3-8: Giải thích một số chức năng, ký hiệu trong Selenium IDE.

- Base URL: Chứa địa chỉ URL của ứng dụng Web đang được kiểm thử.
- Thanh trượt : Điều chỉnh tốc độ thực thi ca kiểm thử tự động.
- Chạy bộ kiểm thử : Chạy tất cả các ca kiểm thử có trong bộ kiểm thử (test suite).
- Chạy một ca kiểm thử : Chạy ca kiểm thử hiện tại được chọn.
- Tạm dừng : Tạm dừng ca kiểm thử đang thực thi.
- Tiếp tục : Tiếp tục chạy ca kiểm thử vừa tạm dừng.
- Bỏ qua : Bỏ qua một bước ở vị trí nó đang dừng.
- Ghi (Record) : Ghi lại các thao tác trong ca kiểm thử. Mặc định khi khởi động Selenium IDE, chức năng ghi thao tác tự động được kích hoạt.
- Textbox *Command*: Lưu trữ các hành động (action) trong ca kiểm thử.
- Textbox *Target*: Đích đến của thao tác.

- **Textbox Value:** Giá trị đầu vào cho thao tác trong ca kiểm thử.

Tab *Source* là nơi hiển thị kịch bản ca kiểm thử dưới dạng HTML. Đây cũng chính là source của file HTML sau khi tiến hành lưu lại ca kiểm thử cho mục đích sử dụng lại.



```
Table Source
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//E
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="e
<head profile="http://selenium-ide.openqa.org/profiles
<meta http-equiv="Content-Type" content="text/html; ch
<link rel="selenium.base" href="https://www.google.com
<title>New Test</title>
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr><td rowspan="1" colspan="3">New Test</td></tr>
</thead><tbody>
<tr>
<td>open</td>
<td>?gfe_rd=cr&dc=0&ei=JAqyWo7MEuytX
```

Hình 3-9: Kịch bản kiểm thử được Selenium IDE lưu trữ dưới dạng HTML.

Ngoài ra, các thông báo cho thao tác vừa thực hiện, thông báo lỗi được thể hiện ở khung ghi *Log*.

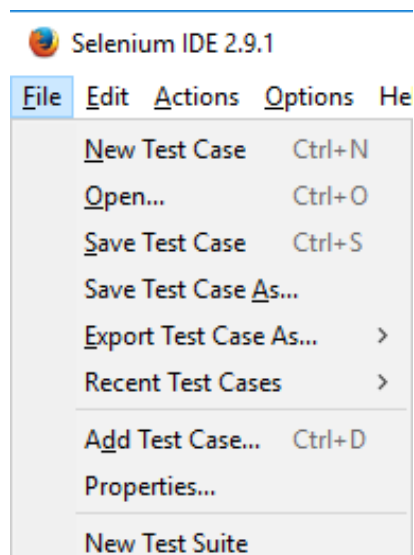
Trong Selenium IDE, các ca kiểm thử luôn có điểm bắt đầu. Điều này tương ứng với cách thiết kế *tiền điều kiện* (Pre-condition) trong ca kiểm thử. Điển hình nhất cho điểm bắt đầu của một ca kiểm thử với Selenium IDE là việc truy cập một ứng dụng Web để bắt đầu luồng công việc.

3.1.2.4. Thao tác cơ bản với Selenium IDE

- **Tạo ca kiểm thử và bộ kiểm thử mới**


Với những trường hợp chỉ cần tạo ca kiểm thử đơn lẻ, ta có thể sử dụng thao tác File / New Test Case.

Tuy nhiên, dù chỉ kiểm thử một form đơn giản cũng cần tạo ra khá nhiều ca kiểm thử khác nhau. Lúc này ta cần tạo ra một bộ kiểm thử (Test Suite) chứa các ca kiểm thử trong đó để dễ dàng quản lý bằng cách chọn File / New Test Suite.



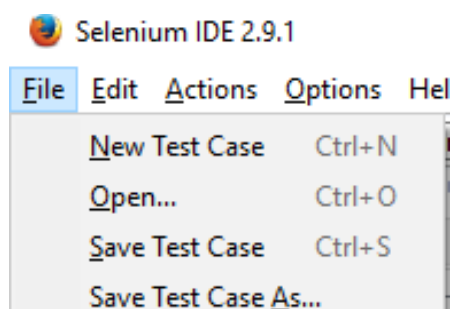
Hình 3-10: Chức năng tạo mới ca kiểm thử/bộ kiểm thử nằm trong menu File.

- **Khởi động chức năng ghi (Record)**

Mặc định khi mở tiện ích Selenium IDE thì chức năng ghi đã được kích hoạt. Có thể dễ dàng nhận ra trạng thái kích hoạt bằng giao diện trực quan của tiện ích. Để kiểm tra, di chuyển con trỏ chuột vào nút Record , nếu xuất hiện thông báo *Click to Record* tương đương với chức năng ghi chưa được kích hoạt.

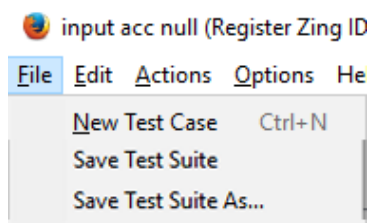
- **Lưu ca kiểm thử/bộ kiểm thử**

Sau khi đã hoàn thành các thao tác với một ca kiểm thử, ta có thể lưu chúng lại bằng cách chọn File / Save Test Case (Ctrl+S). Trường hợp muốn lưu ca kiểm thử đã tồn tại dưới một tên khác, ta chọn File / Save Test Case As...



Hình 3-11: Minh họa thao tác lưu ca kiểm thử.

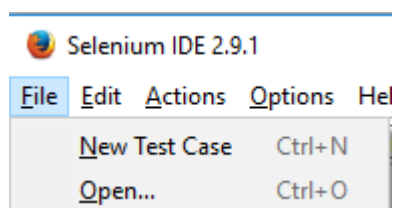
Việc lưu lại một bộ kiểm thử cũng tương tự, ta chọn File / Save Test Suite hoặc File / Save Test Suite As... khi muốn lưu ca kiểm thử đã tồn tại dưới tên khác.



Hình 3-12: Minh họa thao tác lưu bộ kiểm thử.

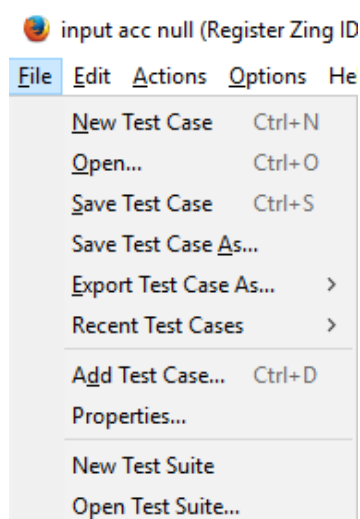
- **Mở ca kiểm thử/bộ kiểm thử đã lưu**

Chọn File / Open và tìm tới thư mục lưu ca kiểm thử để mở một ca kiểm thử đã lưu.



Hình 3-13: Minh họa thao tác mở ca kiểm thử.




Trường hợp muốn mở một bộ kiểm thử, chọn File / Open Test Suite... và tìm tới thư mục lưu bộ kiểm thử, tiếp tục chọn bộ kiểm thử cần thực thi sau đó chọn Open.

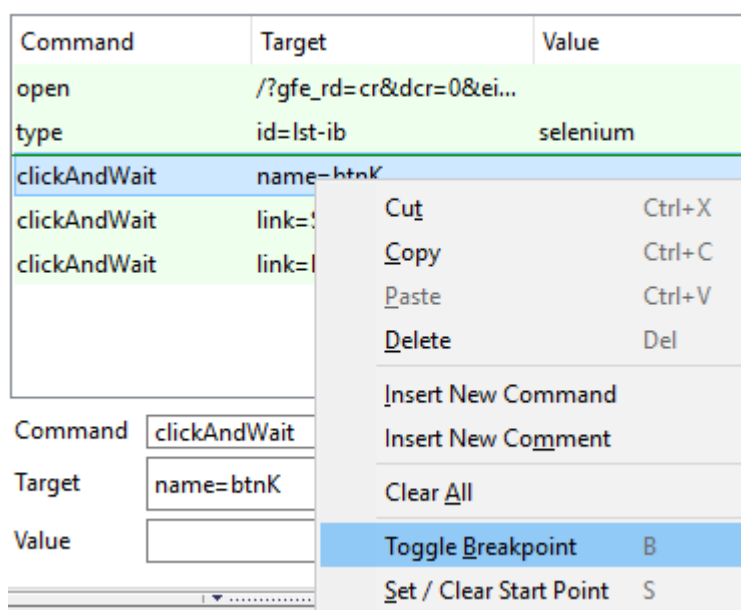


Hình 3-14: Minh họa thao tác mở bộ kiểm thử đã lưu.

• Chạy các ca kiểm thử

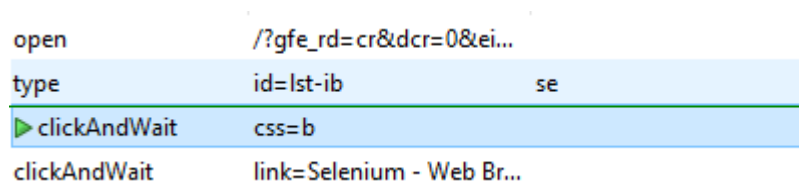
Selenium IDE có nhiều tùy chọn phù hợp cho việc thực thi ca kiểm thử, tạm dừng, chạy từng dòng lệnh đơn lẻ (step), thiết lập điểm ngắt (breakpoint) và chạy tất cả các ca kiểm thử có trong một bộ kiểm thử. Điều này giúp kiểm thử viên dễ dàng làm chủ các thao tác trong ca kiểm thử của mình.

- *Chạy một ca kiểm thử*: Chọn biểu tượng Play current test case  để chạy ca kiểm thử đang chọn.
- *Tạm dừng/tiếp tục chạy ca kiểm thử*: Chọn Pause  khi muốn tạm dừng ca kiểm thử đang chạy và Resume  để tiếp tục chạy ca kiểm thử đang bị tạm dừng.
- *Thiết lập điểm dừng (Toggle breakpoint)*: Selenium IDE hỗ trợ thiết lập các điểm dừng khi chạy một ca kiểm thử. Cụ thể, kiểm thử viên có thể xác định điểm dừng tại một dòng lệnh ở giữa ca kiểm thử nhằm mục đích kiểm tra tình trạng của ứng dụng Web tại điểm dừng đó. Để thiết lập điểm dừng, chọn dòng lệnh mà tại đó ca kiểm thử sẽ dừng hoạt động, click chuột phải chọn Toggle breakpoint.



Hình 3-15: Thiết lập điểm dừng cho ca kiểm thử.

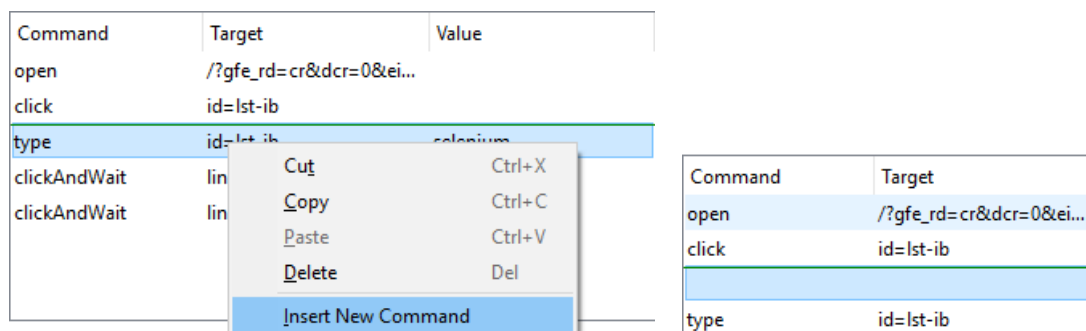
- **Thiết lập điểm bắt đầu (Start Point):** Ngược lại với việc thiết lập điểm dừng, Selenium IDE hỗ trợ thiết lập điểm bắt đầu cho ca kiểm thử tại vị trí bất kỳ mà kiểm thử viên mong muốn. Giả sử ca kiểm thử bắt đầu bằng việc truy cập vào một ứng dụng Web, đăng nhập và thực hiện các thao tác với ứng dụng. Tuy nhiên kiểm thử viên chỉ muốn kiểm tra các thao tác sau khi đã đăng nhập thành công. Lúc này kiểm thử viên hoàn toàn có thể thiết lập điểm bắt đầu ngay sau khi hệ thống đã duyệt qua thao tác đăng nhập. Chọn dòng lệnh tại vị trí bắt đầu mong muốn, sau đó click chuột phải, chọn Set / Clear Start Point để thiết lập điểm bắt đầu cho ca kiểm thử. Tại dòng lệnh điểm bắt đầu sẽ xuất hiện biểu tượng ▶ thông báo thiết lập thành công. Tiếp tục chọn Set / Clear Start Point một lần nữa để xóa thiết lập điểm bắt đầu.



Hình 3-16: Thiết lập điểm bắt đầu cho ca kiểm thử.

- **Chạy từng dòng lệnh (step):** Sau khi thực hiện thao tác tạm dừng (Pause) hoặc ca kiểm thử đã chạy đến điểm dừng (breakpoint), kiểm thử viên có thể chạy từng dòng lệnh tiếp theo để kiểm tra việc chuyển trạng thái của ứng dụng Web. Selenium IDE cung cấp tính năng này bằng việc chọn biểu tượng ↴. Mỗi lần chọn tương ứng với một dòng lệnh được thực thi.
- **Chèn dòng lệnh mới:** Selenium IDE hỗ trợ việc bổ sung dòng lệnh mới vào bất kỳ vị trí nào trong ca kiểm thử. Điều này giúp kiểm thử viên linh động hơn trong việc thiết kế ca kiểm thử phù hợp với yêu cầu. Để làm được điều này, click chuột phải lên một dòng lệnh, Selenium IDE sẽ tự động thêm một dòng lệnh trống mới lên bên trên

dòng lệnh vừa chọn. Sau đó bạn có thể tiến hành cài đặt các thuộc tính cho lệnh mới như bình thường.



Hình 3-17: Minh họa thao tác chèn dòng lệnh mới.

- **Chèn nhận xét (comment):** Giống như công việc lập trình, comment rõ ràng tại các dòng lệnh sẽ giúp quản lý và tái sử dụng ca kiểm thử hiệu quả hơn. Để thực hiện, click chuột phải tại dòng lệnh cần comment và chọn Insert New Command, sau đó gõ nội dung comment vào textbox Command.

Command	Target	Value
open	/?gfe_rd=cr&dcr=0&ei...	
This is a comment		
click	id=lst-ib	
type	id=lst-ib	selenium

Hình 3-18: Chèn nhận xét cho một dòng lệnh trong Selenium IDE.

3.1.2.5. Các câu lệnh trong Selenium IDE – Selenese

Các câu lệnh trong Selenium IDE thường được gọi là Selenese. Selenium IDE cung cấp cho kiểm thử viên một tập lệnh phong phú để kiểm thử ứng dụng trên nền Web. Trong Selenium IDE, kiểm thử viên có thể kiểm tra các yếu tố giao diện người dùng, nội dung, liên kết bị hỏng, dữ liệu đầu vào. Ngoài ra, Selenium IDE còn hỗ trợ kiểm tra kích thước cửa sổ, vị trí chuột, cảnh báo (alert), cửa sổ popup, xử lý sự kiện và nhiều tính năng khác cho ứng dụng Web.

Một lệnh của Selenium IDE thường có 3 thành phần: Actions, Accessors và Assertions.

- Action (hành động): là các thao tác chung của ứng dụng. Ví dụ như click vào một liên kết hay chọn và chờ liên kết tải xong (ClickAndWait). Nếu Action không thành công hoặc có lỗi xảy ra, việc thực thi ca kiểm thử sẽ bị dừng lại.
- Accessors: Kiểm tra trạng thái của ứng dụng và lưu kết quả vào các biến. Kiểm thử viên có thể kiểm tra và lưu trữ tiêu đề trang Web (storeTitle), các phần tử được chọn (storeElementPresent).
- Assertions (xác minh): Assertions gần tương tự như Accessors, tuy nhiên nó xác định trạng thái của ứng dụng cho phù hợp với với kết quả mong đợi. Ví dụ như đảm bảo tiêu đề trang là đúng với thiết kế (verifyTitle) hay xác minh tính đúng đắn của giá trị tại một textbox (verifyValue).

Assertions được chia làm 3 loại: assert, verify và waitFor. Khi một assert thất bại thì việc kiểm thử sẽ dừng lại. Khi một verify không thành công, việc kiểm thử vẫn được tiếp tục tiến hành nhưng sẽ xuất hiện thông báo lỗi. Các lệnh waitFor chờ đợi một số điều kiện được thực thi (hữu ích với việc kiểm thử ứng dụng Ajax). WaitFor thành công nếu điều kiện đúng và việc kiểm thử sẽ bị dừng lại nếu điều kiện không được thực thi trong thời gian chờ (thời gian chờ có thể được thiết lập dễ dàng thông qua thuộc tính SetTimeout).

Bảng sau đây cung cấp 1 số lệnh thường dùng trong Selenium IDE:

Tên lệnh	Ý nghĩa
Open	Đi đến một trang Web theo URL xác định.
Click	Hoàn thành hành động click chuột.
ClickAnd Wait	Hoàn thành hành động click chuột và đợi tải một trang Web mới.
verifyTitle/ assertTitle	Xác minh tiêu đề trang mong đợi.
verifyTextPresent	Xác minh giá trị một đoạn văn bản ở vị trí nào đó trên trang.

verifyElementPresent	Xác minh thành phần giao diện người dùng được mong đợi, được định nghĩa bởi thẻ HTML là tồn tại trên trang.
verifyText	Xác minh văn bản mong đợi và các thẻ HTML tương ứng trên trang.
verifyTable	Xác minh nội dung mong đợi của một bảng.
waitForPageToLoad	Tạm dừng thực thi ca kiểm thử cho tới khi việc tải trang Web mới được hoàn tất. Lệnh này được tự động gọi khi sử dụng lệnh clickAndWait.
waitForElementPresent	Tạm dừng thực thi ca kiểm thử cho tới khi các yếu tố giao diện người dùng mong đợi trên trang Web xuất hiện.

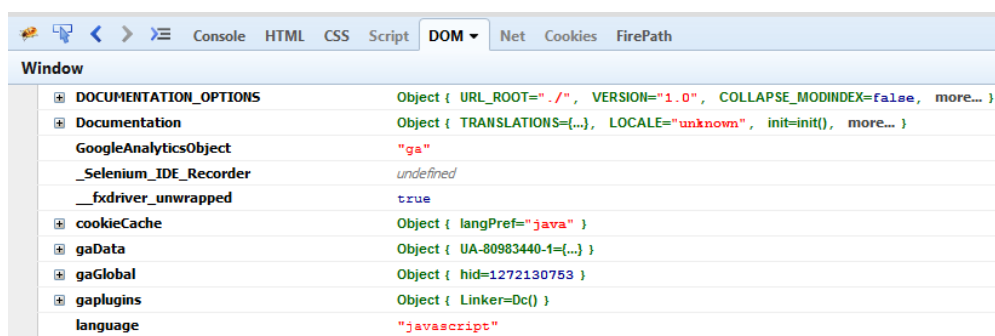
Hình 3-19: Bảng liệt kê một số lệnh thường dùng trong Selenium IDE.

3.2. Một số công cụ hỗ trợ kiểm thử ứng dụng Web

Ngoài công cụ kiểm thử tự động Selenium IDE, kiểm thử viên còn cần tới 1 số công cụ hỗ trợ khác, có thể là Add-on trên trình duyệt Firefox hoặc các ứng dụng desktop nhằm mục đích phục vụ đặc lực cho quá trình kiểm thử ứng dụng trên nền Web. Phần này sẽ giới thiệu về tiện ích FireBug trên Firefox và ứng dụng chụp ảnh màn hình Monosnap.

3.2.1. Firebug

Firebug là một phần mở rộng miễn phí trên trình duyệt Firefox cung cấp cho người dùng các công cụ phục vụ cho việc phát triển website. Firebug có thể dùng để chỉnh sửa HTML/CSS và gỡ lỗi Javascript trực tiếp trên trình duyệt, thêm CSS cho thiết kế responsive. Firebug là một công cụ không thể thiếu trong quá trình xây dựng website của các lập trình viên, kiểm thử viên.

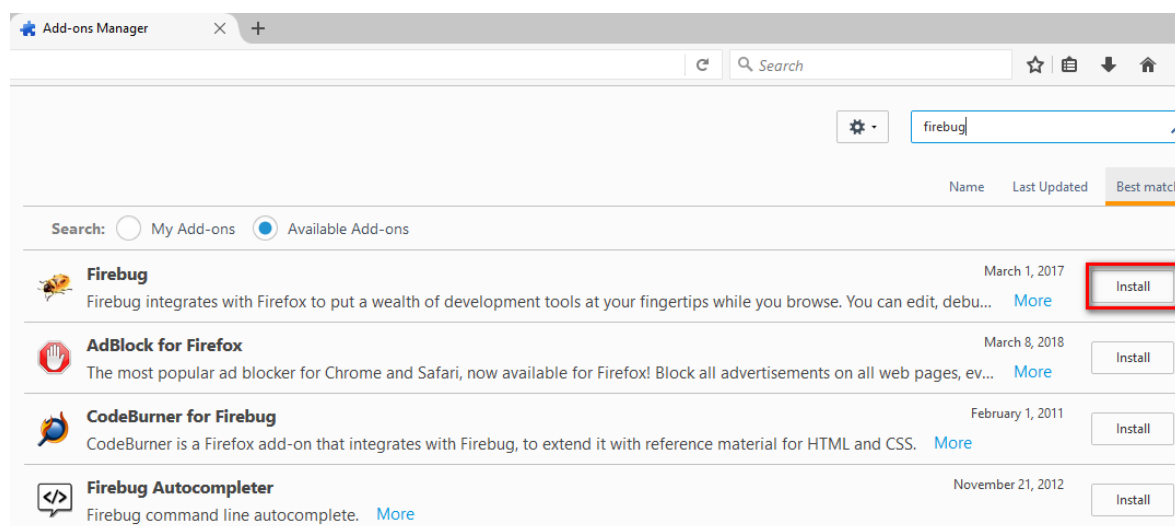


Hình 3-20: Giao diện tiện ích hỗ trợ kiểm thử Firebug.

Một số tab thường dùng trong Firebug:

- **Console:** Chứa các thông báo lỗi/cảnh báo (warning) của các đoạn mã Javascript có trong ứng dụng Web. Đây cũng là phần được các kiểm thử viên quan tâm nhất.
- **HTML:** Phần này hiển thị các thẻ HTML của ứng dụng Web đang duyệt. Thoạt nhìn nó có vẻ giống chức năng Xem mã nguồn (View Source) tích hợp sẵn trong các trình duyệt. Tuy nhiên, điểm khác biệt là nó được hỗ trợ rất nhiều tính năng kèm theo giúp người dùng dễ dàng kiểm tra và sửa mã nguồn ngay trên trình duyệt để theo dõi các thay đổi.
- **CSS:** Chứa toàn bộ mã nguồn CSS của ứng dụng Web. Người dùng có thể thay đổi các mã nguồn CSS ngay tại đây.
- **DOM:** Theo dõi các đối tượng HTML.

Để cài đặt tiện ích Firebug, truy cập vào phần quản lý Add-ons của trình duyệt Firefox, trong tab Extensions gõ tên tiện ích Firebug vào khung Search và chọn Install.

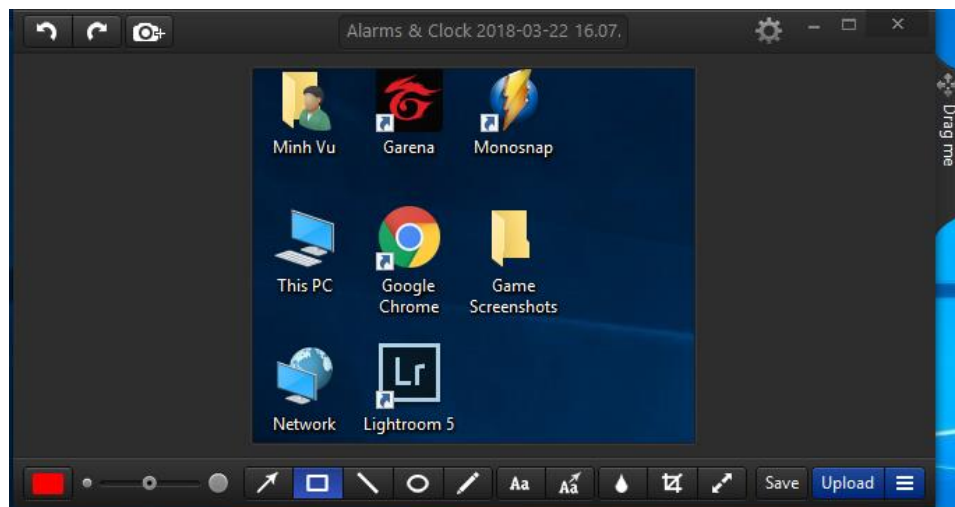


Hình 3-21: Cài đặt công cụ Firebug trong trình quản lý Add-ons của Firefox.

3.2.2. Monosnap

Monosnap là ứng dụng phần mềm mang đến cho người dùng một phương thức đơn giản nhất để chụp màn hình hay chụp ảnh với camera kết nối mạng và cũng cho phép chỉnh sửa ảnh. Đối với kiểm thử viên, Monosnap

rất hữu ích khi dùng để chụp màn hình các lỗi xảy ra trong quá trình kiểm thử, phục vụ cho quá trình làm Bug Report gửi tới các bộ phận khác.



Hình 3-22: Giao diện ứng dụng chụp ảnh màn hình Monosnap.

Ưu điểm của Monosnap là cho phép thao tác nhanh bằng phím tắt, có nhiều lựa chọn cho phép chụp toàn màn hình hoặc từng khu vực trên màn hình. Ngoài ra, ứng dụng này cũng hỗ trợ chỉnh sửa ảnh vừa chụp với các công cụ vừa đủ, không quá phức tạp, giúp người dùng có thể thao tác nhanh và cho chất lượng ảnh sau khi lưu lại khá tốt. Đây cũng là một ứng dụng với dung lượng khá nhẹ và cho phép download miễn phí tại trang chủ <http://monosnap.com>

3.2.3. Công cụ quản lý lỗi (bug) MantisBT

MantisBT tên gọi đầy đủ là Mantis Bug Tracker được sử dụng như một công cụ quản lý lỗi trong dự án phần mềm. Công cụ này được xây dựng trên mã nguồn mở và rất hữu ích cho những người trực tiếp tham gia vào quá trình phát triển phần mềm như: lập trình viên, người quản lý dự án, kiểm thử viên.

Một số ưu điểm của MantisBT có thể kể đến như:

- Hoàn toàn miễn phí do được phát triển từ mã nguồn mở.
- Dễ dàng cài đặt.
- Chạy trên nền Web-based nên tương thích với mọi trình duyệt Web.
- Có thể quản lý nhiều dự án cùng lúc.

- Hỗ trợ đa ngôn ngữ.
- Tích hợp chức năng gửi email thông báo.
- Chức năng tìm kiếm dễ dàng và đơn giản.

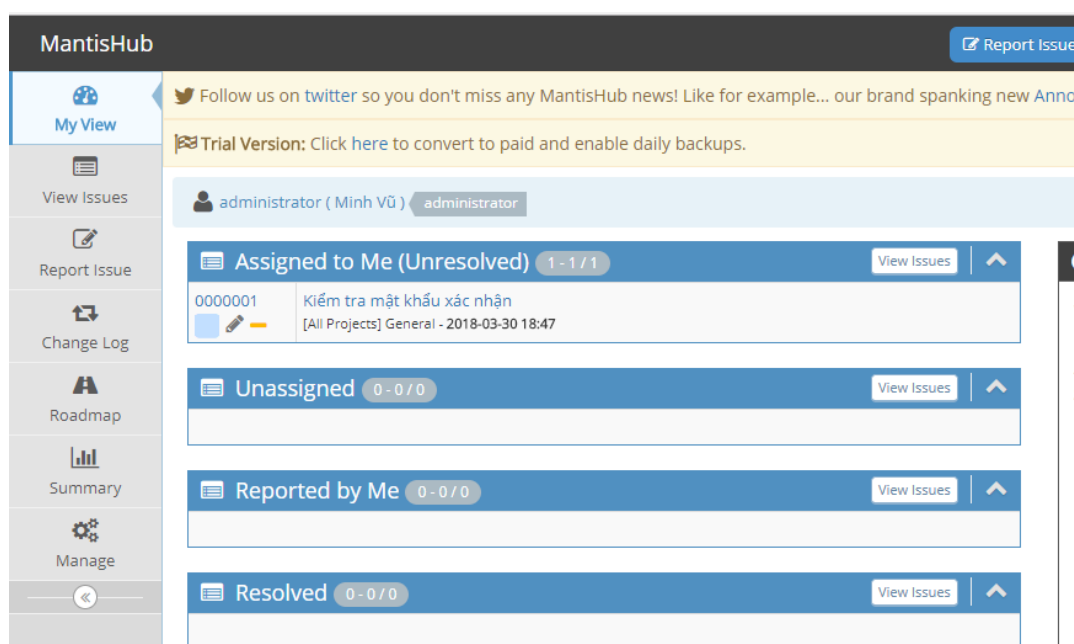
Tiến hành truy cập vào website <https://www.mantisbt.org/> để tải về bộ cài đặt MantisBT hoặc sử dụng bản demo trong vòng 14 ngày. Bộ cài đặt MantisBT có thể cấu hình dễ dàng để chạy trên bất kỳ máy chủ nào hỗ trợ PHP.

Khi sử dụng MantisBT cần chú ý tới một số khái niệm thường gặp sau đây:

- *Issues*: Được hiểu như là lỗi (thường gọi là *bug*), sai sót trong chương trình hoặc tài liệu của dự án.
- *Report Issue*: Báo cáo lỗi, sai sót được tìm thấy trong chương trình, tài liệu của dự án.
- *Severity*: Mức độ nghiêm trọng của lỗi (issue/bug), bao gồm 4 mức: Mức 1: *Block / Crash*, Mức 2: *Major*, Mức 3: *Minor*, Mức 4: *Tweak / Text / Trivial / Feature*.
- *Status*: Mô tả trạng thái của lỗi (issue/bug), bao gồm:
 - *New*: Lỗi mới được báo cáo, chưa xử lý hoặc đang trong giai đoạn xử lý.
 - *Resolved*: Lỗi đã xử lý xong.
 - *Closed*: Lỗi sau khi xử lý đã được người báo cáo lỗi đó kiểm tra lại và xác nhận không còn lỗi.
 - *Feedback*: Lỗi đang chờ phản hồi.
 - *Confirmed*: Lỗi đang chờ xác nhận lại.
 - *Assigned*: Lỗi đã được bàn giao cho người phụ trách lỗi cụ thể.
 - *Acknowledged*: Lỗi đã được người phụ trách chấp nhận xử lý.
- *Resolution*: Tình trạng giải quyết lỗi, bao gồm:
 - *Open*: Lỗi vừa được tạo.
 - *Fixed*: Lỗi đã được xử lý.
 - *Reopened*: Lỗi vẫn xảy ra sau khi đã được xử lý và được mở lại để tiếp tục xử lý lỗi trên.
 - *Not Fixable*: Lỗi không thể xử lý được.

- Unable to produce: Lỗi không bị phát sinh thêm lỗi khác.
- Duplicate: Lỗi đã bị trùng với một lỗi khác.
- Cancelled: Việc xử lý lỗi bị hủy bỏ.
- Suspended: Lỗi bị treo, chưa được xử lý.

Sau khi cài đặt và đăng nhập thành công, giao diện My View được mở ra với các thống kê chi tiết về việc quản lý lỗi: hiển thị lỗi đang chịu trách nhiệm xử lý, lỗi đã xử lý xong, lỗi đã báo cáo, v.v.



Hình 3-23: Giao diện trang chủ MantisBT.

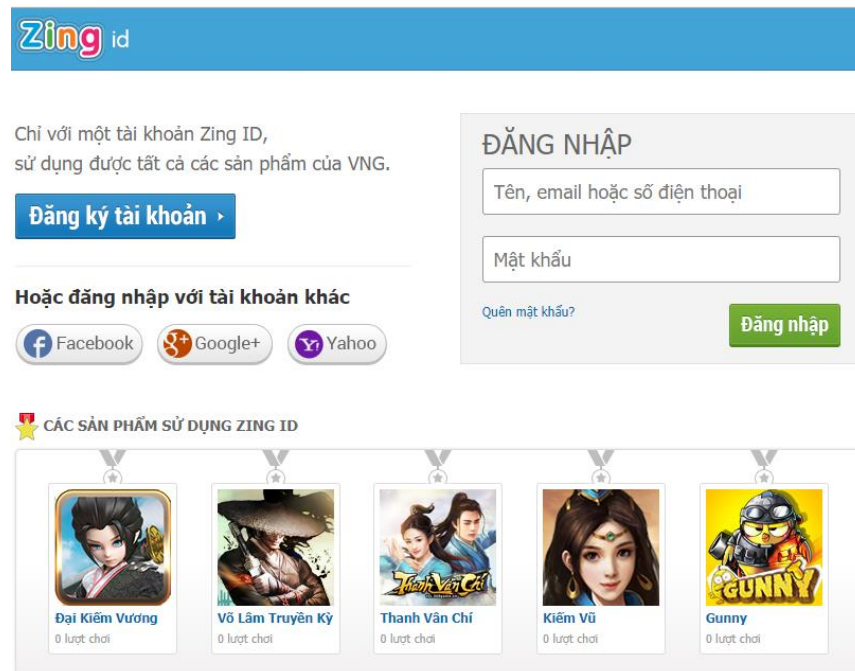
Người dùng có thể xem lỗi đang có được sắp xếp theo ID, ngày giờ tại mục View Issues. Khi click vào ID của một lỗi, MantisBT cho phép xem hoặc chỉnh sửa các thông tin liên quan tới lỗi đó. Để báo cáo lỗi, chọn Report Issue và điền vào các thông tin về lỗi đã phát hiện ra.

Với thiết kế đơn giản, trực quan, hoàn toàn miễn phí cùng hệ thống chức năng vừa đủ cho việc quản lý lỗi, MantisBT hiện được sử dụng rất nhiều trong các dự án kiểm thử cũng như phát triển phần mềm.

3.3. Bài toán thực tế

3.3.1. Giới thiệu bài toán

Zing ID là tài khoản dùng để truy cập tất cả các sản phẩm của Công ty Cổ phần VNG. Hiện tại, các sản phẩm sử dụng Zing ID bao gồm: Đại Kiếm Vương, Võ lâm truyền kỳ, Thanh Vân Chí, Kiếm Vũ, Gunny và mạng xã hội Zing Me. Địa chỉ truy cập hiện tại của Zing ID là <https://id.zing.vn/>.



Hình 3-24: Giao diện hiện tại của website Zing ID.

Website <https://id.zing.vn/> có các chức năng:

- Hỗ trợ người dùng đăng ký và đăng nhập vào tài khoản Zing ID của mình.
- Cho phép người dùng xem và cập nhật thông tin cá nhân trong tài khoản Zing ID.
- Cho phép người dùng tra cứu thông tin kết nối tài khoản Zing ID với các tài khoản Facebook, Google, Yahoo.
- Hỗ trợ người dùng khi gặp các sự cố trong quá trình sử dụng sản phẩm của Zing.
- Cung cấp thông tin về sự kiện mới của các sản phẩm được phát hành bởi Zing.

Trong khuôn khổ đề án, do hạn chế về kinh nghiệm và thời gian tìm hiểu công cụ, em sẽ trình bày quá trình kiểm thử chức năng đăng ký tài khoản trong website <https://id.zing.vn/>. Quá trình kiểm thử sẽ diễn ra như công việc thực tế của một kiểm thử viên, trong đó có: sinh ca kiểm thử, tiến hành kiểm thử trên công cụ kiểm thử tự động và sử dụng Selenium IDE 2.9.1 cài đặt trên trình duyệt Mozilla Firefox phiên bản 54.0.1 64-bit.

3.3.2. Kiểm thử chức năng đăng ký tài khoản trên website <https://id.zing.vn/> sử dụng công cụ Selenium IDE

Khi đăng ký tài khoản tại website <https://id.zing.vn/>, người dùng được yêu cầu nhập vào: họ tên, tên tài khoản, mật khẩu. Các thông tin còn lại về ngày sinh, giới tính có thể cập nhật sau cùng với email, số điện thoại, v.v. Khi người dùng nhập thiếu một trong các thông tin trên, nhập tên tài khoản đã được sử dụng hay nhập mật khẩu không hợp lệ, hệ thống sẽ đưa ra thông báo cụ thể.

Đề án sẽ sử dụng kỹ thuật kiểm thử đoán lỗi để kiểm tra trường hợp người dùng nhập thiếu thông tin đăng ký (không nhập họ tên, tên tài khoản hoặc mật khẩu). Bên cạnh đó, kỹ thuật kiểm thử biên và kiểm thử vùng tương đương được dùng để kiểm tra thao tác nhập mật khẩu của người dùng (mật khẩu quá ngắn, mật khẩu hợp lệ).

3.3.2.1. Ca kiểm thử trường hợp nhập thiếu tên đăng nhập

Test Case ID: ZID_101.

Test Designed by: Vũ Công Minh.

Test Priority: High.

Test Designed date: 15/03/2018.

Module Name: Zing ID register.

Test Executed by: Selenium IDE.

Test Title: Chức năng đăng ký

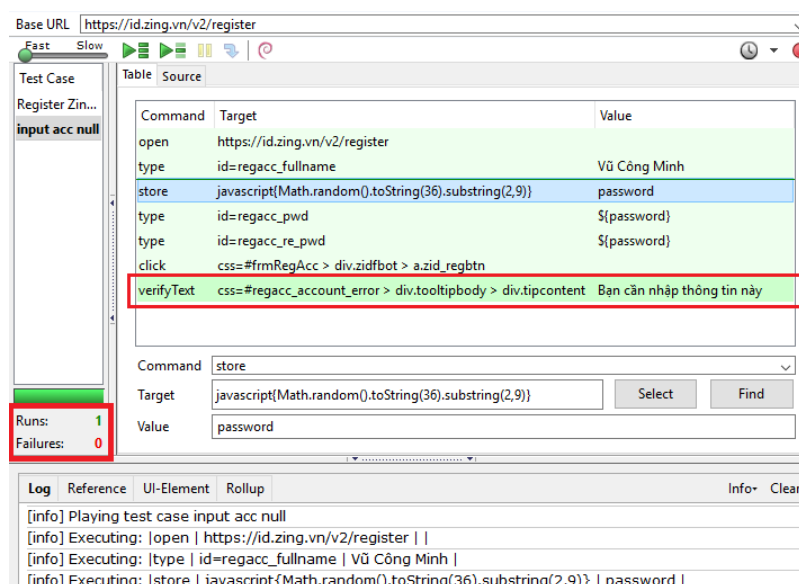
Test Executed date: 15/03/2018.

thiếu thông tin.

Description: Kiểm tra chức năng đăng ký khi người dùng nhập thiếu thông tin (tên đăng nhập).

Pre-condition: Mở sẵn trình duyệt. Truy cập trang đăng ký tài khoản Zing ID trên trình duyệt.

Step	Action	Test Data	Expected Result	Actual Result	Pass / Fail
1	Mở trang đăng ký tài khoản	id.zing.vn/v2/register	Truy cập website thành công.	Truy cập website thành công.	Pass
2	Nhập họ tên hợp lệ	Vũ Công Minh	Hiện thị họ tên vừa nhập	Hiện thị họ tên vừa nhập	Pass
3	Nhập mật khẩu hợp lệ	Ký tự ngẫu nhiên	Nhập thành công	Nhập thành công	Pass
4	Nhập xác nhận mật khẩu	Khớp với mật khẩu đã nhập	Nhập thành công	Nhập thành công	Pass
5	Click <i>Đăng ký</i>	Button Đăng ký	Yêu cầu nhập tên đăng nhập	Yêu cầu nhập tên đăng nhập	Pass



Hình 3-25: Thực thi ca kiểm thử ZID_101 trên Selenium IDE.

Mỗi ca kiểm thử với Selenium IDE đều có thể xây dựng kịch bản theo ba thuộc tính: *Command*, *Target*, *Value* trong tab Table hoặc viết dưới dạng HTML trong tab Source. Cụ thể với ca kiểm thử này, kịch bản kiểm thử dưới dạng HTML bao gồm các thành phần như sau:

- Mã nguồn bắt buộc

```
<!-- Phần mã nguồn do Selenium IDE tự sinh ra -->
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<link rel="selenium.base" href="https://id.zing.vn/v2/register" />
```

```
<title>New Test</title> <!--Tiêu đề/Tên ca kiểm thử -->
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr><td rowspan="1" colspan="3">New Test</td></tr>
</thead><tbody>

<!-- KHU VỰC VIẾT MÃ NGUỒN TÙY CHỈNH CHO TỪNG CA KIỂM THỬ -->

</tbody></table>
</body>
</html>
```

• Mã nguồn tùy chỉnh theo kịch bản của ca kiểm thử

```
<!-- Mở giao diện trang đăng ký Zing ID -->
<tr>
  <td>open</td>
  <td>https://id.zing.vn/v2/register</td>
  <td></td>
</tr>
<!-- Nhập tên tài khoản -->
<tr>
  <td>type</td>
  <td>id=regacc_fullname</td>
  <td>Vũ Công Minh</td>
</tr>
<!-- Mã lệnh javascript tự sinh mật khẩu ngẫu nhiên -->
<tr>
  <td>store</td>
  <td>javascript{Math.random().toString(36).substring(2,9)}</td>
  <td>password</td>
</tr>
<!-- Nhập vào mật khẩu tự sinh ra ở bước trước -->
<tr>
  <td>type</td>
  <td>id=regacc_pwd</td>
  <td>${password}</td>
</tr>
<!-- Nhập mật khẩu xác nhận -->
<tr>
  <td>type</td>
  <td>id=regacc_re_pwd</td>
  <td>${password}</td>
</tr>
<!-- Click vào nút Đăng ký. Ở đây do nút Đăng ký không có thuộc tính
ID/name nên xác định bằng các thuộc tính khác như class CSS, các viết
thẻ HTML -->
<tr>
  <td>click</td>
  <td>css=#frmRegAcc &gt; div.zidfbot &gt; a.zid_regbtn</td>
  <td></td>
</tr>
```

```
<!-- Xác nhận thông báo lỗi -->
<tr>
  <td>verifyText</td>
  <td>css=#regacc_account_error &gt; div.tooltipbody &gt; div.tipcontent</td>
  <td>Bạn cần nhập thông tin này</td>
</tr>
```

3.3.2.2. Ca kiểm thử trường hợp nhập mật khẩu không hợp lệ

Test Case ID: ZID_102.

Test Designed by: Vũ Công Minh.

Test Priority: Medium.

Test Designed date: 15/03/2018.

Module Name: Zing ID register.

Test Executed by: Selenium IDE.

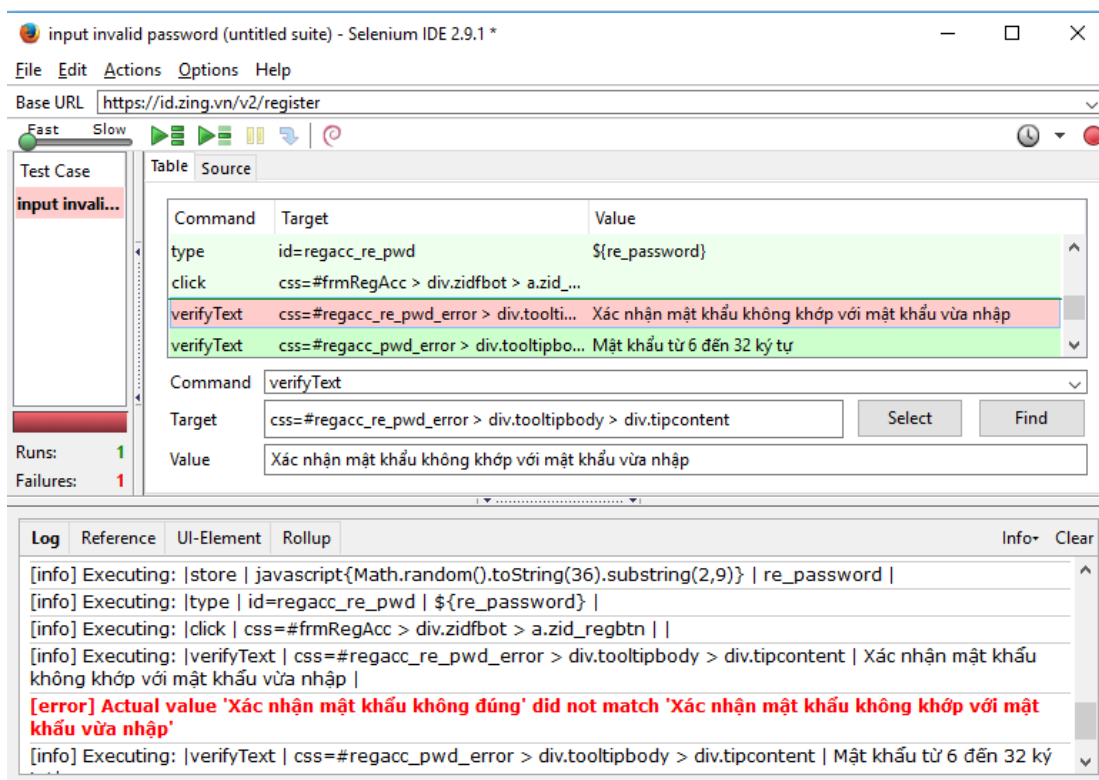
Test Title: Nhập mật khẩu không hợp lệ.

Test Executed date: 15/03/2018.

Description: Kiểm tra chức năng đăng ký khi người dùng nhập mật khẩu không hợp lệ (mật khẩu nhỏ hơn 6 ký tự và xác nhận mật khẩu không khớp).

Pre-condition: Mở sẵn trình duyệt. Truy cập trang đăng ký tài khoản Zing ID trên trình duyệt.

Step	Action	Test Data	Expected Result	Actual Result	Pass / Fail
1	Mở trang đăng ký tài khoản	id.zing.vn/v2/register	Truy cập website thành công.	Truy cập website thành công.	Pass
2	Nhập họ tên hợp lệ	Vũ Công Minh	Hiện thị họ tên vừa nhập	Hiện thị họ tên vừa nhập	Pass
3	Nhập tên tài khoản hợp lệ	seleniumhpu	Nhập thành công	Nhập thành công	Pass
4	Nhập mật khẩu không hợp lệ	Ký tự ngẫu nhiên nhỏ hơn 6 ký tự	Báo mật khẩu nhập vào không hợp lệ	Báo mật khẩu nhập vào không hợp lệ	Pass
5	Nhập xác nhận mật khẩu	Ký tự ngẫu nhiên lớn hơn 6 ký tự, không khớp với mật khẩu	Báo lỗi “Xác nhận mật khẩu không khớp với mật khẩu vừa nhập”	Báo lỗi “Xác nhận mật khẩu không đúng”	Fail
6	Click Đăng ký	Button Đăng ký	Báo lỗi mật khẩu không hợp lệ / mật khẩu và xác nhận mật khẩu không khớp nhau	Hiện thị đầy đủ các thông báo lỗi như mong đợi	Pass



Hình 3-26: Thực thi ca kiểm thử ZID_102 trên Selenium IDE.

3.3.2.3. Ca kiểm thử trường hợp đăng ký thành công

Test Case ID: ZID_103.

Test Designed by: Vũ Công Minh.

Test Priority: Medium.

Test Designed date: 15/03/2018.

Module Name: Zing ID register.

Test Executed by: Selenium IDE.

Test Title: Đăng ký thành công.

Test Executed date: 15/03/2018.

Description: Kiểm tra chức năng đăng ký khi người dùng nhập các thông tin hợp lệ, hoàn tất việc đăng ký thành công.

Pre-condition: Mở sẵn trình duyệt. Truy cập trang đăng ký tài khoản Zing ID trên trình duyệt.

Step	Action	Test Data	Expected Result	Actual Result	Pass / Fail
1	Mở trang đăng ký tài khoản	id.zing.vn/v2/register	Truy cập website thành công.	Truy cập website thành công.	Pass
2	Nhập họ tên hợp lệ	Vũ Công Minh	Hiện thị họ tên vừa nhập	Hiện thị họ tên vừa nhập	Pass
3	Nhập tên tài khoản hợp lệ	selenium113	Nhập thành công	Nhập thành công	Pass

4	Nhập mật khẩu không hợp lệ	abc12345	Nhập thành công	Nhập thành công	Pass
5	Nhập xác nhận mật khẩu	abc12345	Nhập thành công	Nhập thành công	Pass
6	Click Đăng ký	Button Đăng ký	Chuyển sang trang thông tin tài khoản và thông báo đăng ký thành công	Chuyển sang trang thông tin tài khoản và thông báo đăng ký thành công	Pass

Command	Target	Value
open	https://id.zing.vn/v2/register	
type	id=regacc_fullname	Vũ Công Minh
type	id=regacc_account	selenium113
type	id=regacc_pwd	abc12345
type	id=regacc_re_pwd	abc12345
click	css=#frmRegAcc > div.zidfbot > a.zid_regbtn	
verifyText	//body/div[6]	Thông báo Chúc mừng bạn! Bạn đã ...

Hình 3-27: Thực thi ca kiểm thử ZID_103 trên Selenium IDE.

Như vậy, sau khi thực hiện kiểm thử chức năng đăng ký tài khoản Zing ID với ba ca kiểm thử được đưa ra bằng tiện ích Selenium IDE đã xuất hiện thông báo lỗi với ca kiểm thử ZID_102. Cụ thể:

Tổng số ca kiểm thử: 03.

Số ca kiểm thử lỗi: 01.

Lúc này, kiểm thử viên cần thực hiện báo cáo lỗi thông qua công cụ quản lý lỗi MantisBT. Quản lý dự án sẽ nhận được thông báo lỗi qua email và trên giao diện của công cụ này. Sau đó, tùy thuộc vào đặc thù của dự án, người quản lý dự án cần phân công/gán lỗi cho một cá nhân/bộ phận cụ thể để thực hiện xác minh và xử lý lỗi được thông báo.

The screenshot shows the 'View Issue Details' page in MantisBT. At the top, there are buttons for 'Send a Reminder', 'Jump to Notes', and 'Jump to History'. Below this is a table with columns: ID, Project, Category, View Status, Date Submitted, and Last Update. The issue ID is 0000003, Project is MyProject, Category is [All Projects] General, View Status is public, Date Submitted is 2018-03-31 04:25, and Last Update is 2018-03-31 04:26. Below the table are fields for Reporter (hpuSelenium), Assigned To (administrator), Priority (low), Severity (minor), Status (assigned), and Resolution (open). A Summary section contains the text: '0000003: [Đăng ký] Thông báo lỗi sai'. The Description section contains: 'Xuất hiện thông báo lỗi sai với đặc tả khi xác nhận mật khẩu không khớp với mật khẩu. Thông báo lỗi đúng: "Xác nhận mật khẩu không khớp với mật khẩu vừa nhập". Thông báo lỗi sai: "Xác nhận mật khẩu không đúng".' The Tags section shows 'No tags attached.' The Attach Tags section has a text input field, a dropdown for 'Existing tags', and an 'Attach' button. At the bottom, there are buttons for 'Edit', 'Assign To:', 'Change Status To:' (set to 'new'), 'Monitor', 'Stick', 'Clone', 'Close', 'Move', and 'Delete'.

Hình 3-28: Báo cáo lỗi thông qua công cụ MantisBT.

3.4. Kết luận

Chương 3 của đề án đã giới thiệu chung về bộ công cụ kiểm thử tự động Selenium cũng như đi sâu vào tìm hiểu tiện ích Selenium IDE trên trình duyệt Mozilla Firefox. Các nội dung cụ thể trong chương 3 bao gồm:

- Giới thiệu tổng quan về công cụ kiểm thử tự động Selenium.
- Tìm hiểu chi tiết về tiện ích Selenium IDE: cách cài đặt, phạm vi ứng dụng, cách sử dụng cơ bản.
- Giới thiệu một số công cụ hỗ trợ kiểm thử ứng dụng Web hữu ích kết hợp cùng với Selenium trong công việc kiểm thử.
- Ứng dụng kiến thức đã nghiên cứu về công cụ kiểm thử tự động Selenium IDE để kiểm thử chức năng đăng ký tài khoản của ứng dụng Web <https://id.zing.vn/>. Đồng thời thực hiện báo cáo lỗi (Report Issue) thông qua công cụ MantisBT.

KẾT LUẬN

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

Sau 03 tháng thực hiện nghiên cứu đề tài, dưới sự hướng dẫn tận tình của Thạc sỹ Nguyễn Trịnh Đông, đồ án của em đã đạt được những kết quả sau:

❖ Kết quả đạt được

- Trình bày đầy đủ tổng quan về phần mềm, lỗi phần mềm, đặc tả yêu cầu phần mềm, chất lượng và độ tin cậy của phần mềm cũng như các vấn đề liên quan tới kiểm thử phần mềm.
- Chỉ ra được sự khác biệt của công việc kiểm thử ứng dụng trên nền Web với công việc kiểm thử trên các môi trường khác.
- Tìm hiểu chi tiết cách cài đặt và sử dụng tiện ích Selenium IDE trên trình duyệt Mozilla Firefox.
- Áp dụng kiến thức đã tìm hiểu để kiểm thử chức năng đăng ký tài khoản của ứng dụng Web <https://id.zing.vn/> bằng tiện ích Selenium IDE.
- Đồ án là một tài liệu tổng hợp các vấn đề trong kiểm thử phần mềm nói chung, kiểm thử ứng dụng trên nền Web nói riêng và có thể xem như tài liệu hướng dẫn sử dụng Selenium IDE một cách cơ bản nhất bằng tiếng Việt để tham khảo.

❖ Hạn chế

Trong thời gian qua, em đã cố gắng hết sức để tìm hiểu thực hiện đề tài. Tuy nhiên với kinh nghiệm và thời gian hạn chế nên không thể tránh khỏi những thiếu sót trong đồ án. Cụ thể:

- Đồ án mới tập trung nghiên cứu sâu 1 tiện ích trong bộ công cụ kiểm thử tự động Selenium.

- Chưa nghiên cứu được các kỹ thuật nâng cao khi sử dụng Selenium IDE.
- Chỉ áp dụng kiểm thử duy nhất chức năng đăng ký tài khoản của ứng dụng Web Zing ID.

❖ Hướng phát triển của đề tài

Với mong muốn trở thành một kiểm thử viên kiểm thử phần mềm, trong thời gian tới em sẽ tiếp tục tìm hiểu, nghiên cứu sâu hơn các vấn đề của kiểm thử phần mềm, nhất là bộ công cụ Selenium để có thể tiến bộ hơn nữa trong lĩnh vực mà mình theo đuổi.

DANH MỤC 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] P. Bourque, SWEBOK V3.0, IEEE.
- [3] Phạm Ngọc Hùng - Trương Anh Hoàng - Đặng Văn Hưng, Giáo trình Kiểm thử phần mềm, 2014.
- [4] Wikipedia, Bách khoa toàn thư mở.
- [5] ISTQB, <http://istqbexamcertification.com/>.
- [6] ISTQB, Worldwide Software Testing Practices Report, 2015-2016.
- [7] IEEE, ISTQB Glossary of Testing Terms 2.1.