**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------

**BÁO CÁO THỰC TẬP TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*:**

**XÂY DỰNG HỆ THỐNG KIỂM THỬ TỰ ĐỘNG HÓA CHO WEBSITE TRÊN NỀN TẢNG JOOMLA**

## Người hướng dẫn : THS. LÊ MINH HÓA

## Sinh viên thực hiện : PHẠM THỊ HẠNH

**Mã số sinh viên : N14DCCN304**

**Lớp : D14CQIS-01 Khoá : 2014 - 2019**

**Hệ : ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 08/2018**

# LỜI CẢM ƠN

Để có được kết quả như ngày hôm nay, trước tiên em xin gửi lời cảm ơn sâu sắc nhất tới Ths. Lê Minh Hóa về sự tận tình, tận tâm hướng dẫn, giúp đỡ em từ những ý tưởng ban đầu cũng như xuyên suốt quá trình nghiên cứu và thực hiện đồ án thực tập. Nhờ sự giúp đỡ của thầy mà em đã vượt qua được những khó khăn trong quá trình thực hiện đồ án thực tập để có được kết quả như ngày hôm nay.

Em cũng xin bày tỏ lòng biết ơn chân thành tới các thầy cô trong khoa Công nghệ thông tin II – Trường Học viện Công nghệ Bưu chính Viễn thông TP Hồ Chí Minh đã dìu dắt em qua suốt quá trình học tập, rèn luyện để em có đủ hiểu biết và kiến thức thực hiện đồ án thực tập.

Mặc dù đã cố gắng để hoàn thiện đề tài tốt nhất có thể, nhưng chắc chắn không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý của thầy cô và các bạn để đồ án thực tập của em được hoàn thiện hơn.

*Em xin chân thành cảm ơn!*

TP Hồ Chí Minh, ngày 5 tháng 8 năm 2018

Sinh viên

## Phạm Thị Hạnh

# MỞ ĐẦU

Các ứng dụng Web đã được phát triển và trở thành nền tảng kết nối thông tin thiết yếu trong nhiều doanh nghiệp, đóng vai trò quyết định trong thương mại điện tử và trao đổi thông tin. Để có thể đạt được điều này, các ứng dụng Web hoàn hảo cho những người đang và sẽ sử dụng ứng dụng đã trở thành một thử thách chính trong đảm bảo chất lượng. Kiểm thử các ứng dụng Web đã vượt quá giới hạn của kiểm thử những hệ thống phần mềm truyền thống. Như chúng ta đã biết, một ứng dụng Web thường có rất nhiều nhóm người sử dụng với các nền tảng khác nhau (hệ điều hành, trình duyệt,…), điều này dẫn tới việc kiểm thử ứng dụng Web cần phải có những phương pháp đặc biệt khác với phần mềm truyền thống.

Để tạo ra sản phẩm công nghệ thông tin hay phần mềm có chất lượng thì hoạt động kiểm thử phần mềm đóng vai trò rất quan trọng, trong khi đó hoạt động này lại tiêu tốn và chiếm tỷ trọng khá lớn công sức và thời gian trong một dự án. Do vậy nhu cầu tự động hóa quy trình kiểm thử phần mềm cũng được đăt ra.

Qua thực tế cho thấy, việc áp dụng kiểm thử tự động hợp lý sẽ mang lại thành công cho hoạt động kiểm thử phần mềm. Kiểm thử tự động giúp giảm bớt công sức thực hiện, tăng độ tin cậy, giảm sự nhàm chán và rèn luyện kỹ năng lập trình cho người kiểm thử viên.

Trên đây là lý do em chọn đề tài “**Xây dựng hệ thống kiểm thử tự động cho Website trên nền tảng Joomla**”.

## CHƯƠNG 1: TỔNG QUAN VỀ KIỂM THỬ

* 1. **Tổng quan về kiểm thử phần mềm**
     1. **Khái niệm**

Hiện nay có nhiều cách định nghĩa kiểm thử phần mềm và sau đây là một số cách định nghĩa chính:

* Kiểm thử phần mềm (kiểm tra, thử nghiệm) 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. Kiểm thử có thể cung cấp cho doanh nghiệp một quan điểm, một cách nhìn độc lập về phần mềm để từ đó cho đánh giá và thấu hiểu được những rủi ro trong quá trình triển khai phần mềm (*Theo Bách khoa toàn thư mở WIKIPEDIA*).
* Kiểm thử là quá trình vận hành hệ thống hoặc thành phần dưới những điều kiện xác định, quan sát ghi nhận kết quả và đưa ra đánh giá về hệ thống hoặc thành phần đó (*Theo Bảng chú giải thuật ngữ chuẩn IEEE của Thuật ngữ kỹ thuật phần mềm – IEEE Standard Glossary of Software Engineering Terminology*).
* Kiểm thử là quá trình thực thi một chương trình với mục đích tìm ra lỗi (*Theo Mayer – Nghệ thuật kiểm thử phần mềm – The art of software testing*).

Từ đây có thể định nghĩa một cách dễ hiểu như sau: Kiểm thử phần mềm là một tiến trình hay một tập hợp các tiến trình được thiết kế để đảm bảo mã hóa máy tính thực hiện theo cái mà chúng ta đã được thiết kế để làm, và không thực hiện bất cứ thứ gì không mong muốn. Đây là một bước quan trọng trong quá trình phát triển hệ thống, giúp con người xây dựng hệ thống và khách hàng thấy được hệ thống mới đã đáp ứng yêu cầu hay chưa để tiến hành phát triển và sửa chữa kịp thời.

* + 1. **Mục đích của kiểm thử phần mềm**
* Phát hiện và xác định lỗi ở các phần mềm được kiểm thử.
* Tiến hành sửa lỗi ở các phần mềm được kiểm thử và kiểm thử lại cho đến khi đạt một mức độ chất lượng phần mềm chấp nhận được.
* Thực thi những trường hợp kiểm thử một cách hiệu quả trong một giới hạn ngân sách và lịch trình cho phép.
* Để biên dịch một tài liệu về các lỗi phần mềm thường gặp nhằm mục đích ngăn ngừa và sửa lỗi.
* Đảm bảo rằng kết quả cuối cùng đáp ứng các yêu cầu kinh doanh và người sử dụng.
* Để đạt được sự tín nhiệm của khách hàng bằng cách cung cấp cho họ một sản phẩm chất lượng.

Kiểm thử phần mềm sẽ giúp hoàn thiện các ứng dụng phần mềm hoặc sản phẩm so với yêu cầu kinh doanh và người sử dụng. Nó là rất quan trọng để đảm bảo kiểm thử tốt để kiểm thử các ứng dụng phần mềm hoàn toàn và chắc chắn rằng nó hoạt động tốt và theo các thông số kỹ thuật.

Việc xác định phạm vi kiểm tra các trường hợp kiểm thử nên được thiết kế tốt với khả năng tối đa của việc tìm kiếm các lỗi hiệu quả và được tính toán là số bug báo cáo cho mỗi trường hợp kiểm thử.

* + 1. **Vai trò của kiểm thử phần mềm**

Việc tạo ra một sản phẩm phần mềm phải trải qua nhiều giai đoạn, người ta gọi là quy trình phát triển phần mềm. Quy trình phát triển phần mềm gồm các bước cơ bản sau:

* Requirements and Specifications: Phân tích yêu cầu và tài liệu đặc tả
* System analysis and Design: Phân tích hệ thống và thiết kế
* Coding and Unit Test: Thực thi và kiểm tra từng thành phần
* Test: Kiểm thử
* Deployment and Maintenance: Cài đặt và bảo trì

Như vậy, một sản phẩm phần mềm không chỉ đơn giản là các đoạn mã chương trình mà còn rất nhiều thành phần đằng sau nó. Vì vậy, việc mắc lỗi không chỉ xảy ra trong khi lập trình mà còn xảy ra cao hơn trong các công đoạn khác nhau của quy trình phát triển một sản phẩm phần mềm. Việc kiểm thử đóng vai trò quan trọng trong tất cả các phần tạo nên một sản phẩm phần mềm.

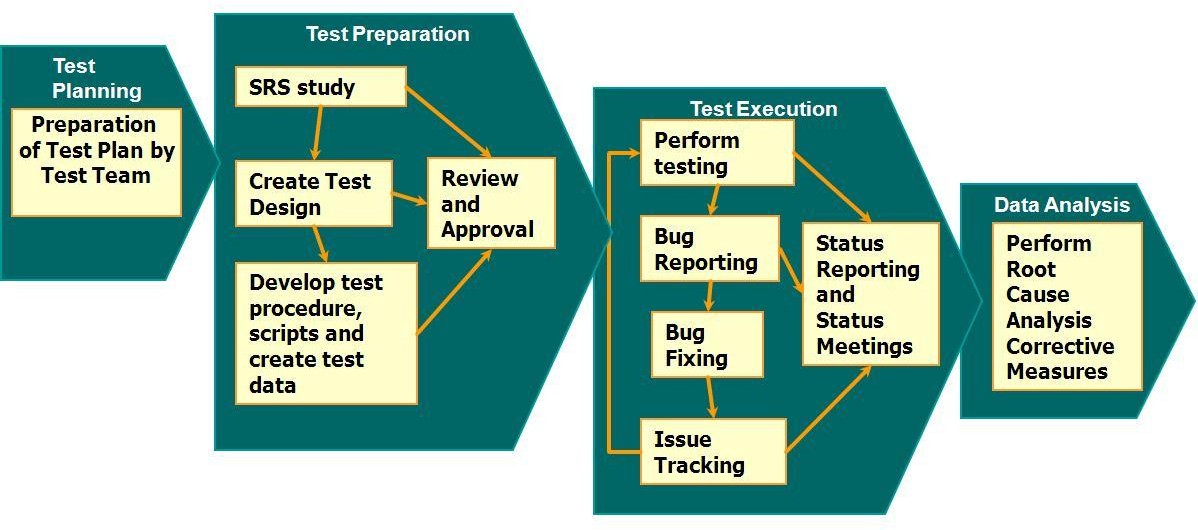
Kiểm thử phần mềm cho phép tạo ra những đánh giá khách quan về mức độ phù hợp của hệ thống các yêu cầu đã nêu và thông số kỹ thuật.

Kiểm tra xác nhận rằng hệ thống đáp ứng các yêu cầu khác nhau bao gồm: chức năng, hiệu suất, độ tin cậy, an toàn, khả năng sử dụng và như vậy. Việc xác nhận này được thực hiện để đảm bảo rằng chúng tôi đang xây dựng hệ thống phù hợp.

Xác nhận để đảm bảo đang xây dựng hệ thống phù hợp. Ngoài việc đưa ra quyết định, các thông tin từ các kiểm thử phần mềm giúp quản lý rủi ro.

**1.2** **Kiểm thử phần mềm**

**1.2.1. Quy trình kiểm thử phần mềm**



**Hình 1.1 – Quy trình kiểm thử phần mềm**

Quy trình kiểm thử phần mềm gồm các bước:

- Testing Planning – Lập kế hoạch kiểm thử: Nhiệm vụ quan trọng trong phần lập kế hoạch kiểm thử là xách định được các yếu tố sau:

* Các giai đoạn kiểm thử áp dụng cho dự án
* Các phương pháp kiểm thử
* Các công cụ kiểm thử
* Nguồn lực kiểm thử
* Tài nguyên môi trường kiểm thử, bao gồm các tài nguyên phần cứng và phần mềm
* Mốc bàn giao các tài liệu kiểm thử.

- Test Preparation – Chuẩn bị kiểm thử: Nhiệm vụ của gian đoạn này:

* Tìm hiểu nghiệp vụ của hệ thống phải kiểm thử
* Xây dựng kịch bản kiểm thử, phát triển các thủ tục và các kịch bản kiểm thử tự động
* Chuẩn bị dữ liệu kiểm thử
* Xem xét phê duyệt các tài liệu kiểm thử

- Test Execution: Thực thi kiểm thử:

* Thực thi kiểm thử dựa trên các kịch bản kiểm thử, testscript, thủ tục, dữ liệu có sẵn từ bước chuẩn bị kiểm thử.
* Tham gia quá trình quản lý lỗi: báo lỗi, sửa lỗi.

- Data Analysis – Phân tích dữ liệu kiểm thử:

* Báo cáo kiểm thử
* Phân tích nguyên nhân và đề xuất các hành động khắc phục.

**1.2.2. Vòng đời của Bugs**

**1.2.2.1. Thế nào là Bug/Defect/Error/Failure?**

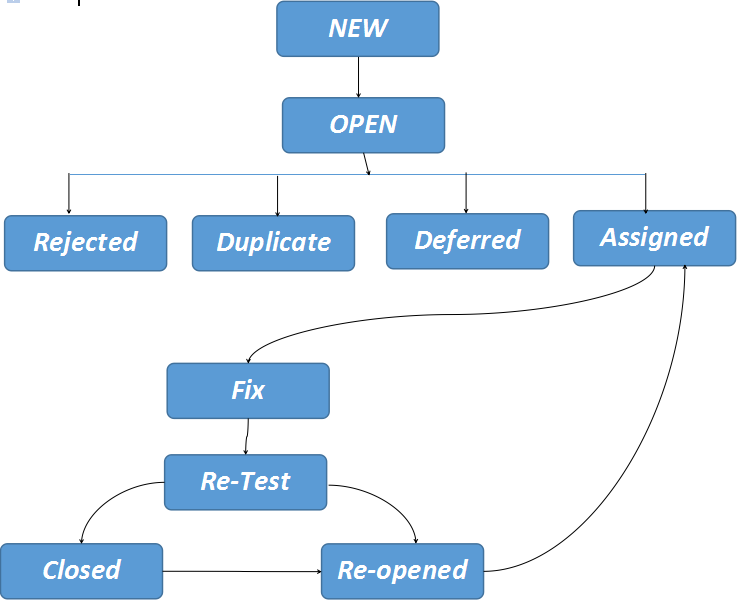
- Bug: là một khiếm khuyết trong một thành phần hoặc hệ thống mà nó có thể làm cho thành phần hoặc hệ thống này không thực hiện đúng chức năng yêu cầu của nó, ví dụ như thông báo sai hoặc định nghĩa dữ liệu không đúng. Một bug nếu gặp phải trong quá trình hệ thống hoạt động, có thể gây ra failure trong thành phần hoặc hệ thống đó. Hay đơn giản, Bug là điều không mong muốn của một chương trình hoặc của một phần cứng, làm cho chương trình không thực hiện đúng như mong đợi hoặc không lường trước, không phù hợp với thông số kĩ thuật.

- Defect: Lỗi trong quá trình phát triển hoặc lỗi logic(coding hoặc logic) làm cho chương trình hoạt động sai yêu cầu đề ra (về cơ bản là giống định nghĩa bug).

- Error: Là hành động của con người dẫn tới kết quả sai (Theo tài liệu IEEE 610).

- Failure: Chính là sự khác biệt giữa kết quả thực tế trên màn hình và kết quả mong đợi của một thành phần, hệ thống hoặc service nào đó. Không phải 100% failure là do bug gây ra, trong quá trình test cấu hình sai, test sai môi trường hoặc làm thiếu bước có thể dẫn đến failure. Tóm lại: con người gây ra error, mistake trong code, tài liệu,…=> dẫn đến có bug, defect hoặc fault trong code, tài liệu,…=> khi thực thi chương trình thì bắt gặp failure.

**1.2.2.2. Vòng đời của Bugs**



**Hình 1.2 - Vòng đời của Bug**

Mô tả từng trạng thái của Bug:

1. NEW: Khi tester thực thi testcase và đầu ra của testcase đó không đúng như kết quả mong đợi, thì sẽ gọi đó là bug. Do đó bug này cần phải được fix, nhưng tester không phải là người fix bug mà là lập trình viên sẽ fix nó => khi tester tìm thấy bug thì nó sẽ có trạng thái là NEW.
2. OPEN: Lỗi được log lên bởi tester. Team lead cần xác minh lại bug đó xem có đúng là bug hay không, thì bug có trạng thái OPEN.
3. REJECTED(từ chối): Một bug được đánh dấu là Rejected khi bug đó không hợp lệ. Nghĩa là thình thoảng tester có thể hiểu sai chức năng và có thể đánh dấu chức năng là bug. Trong trường hợp này, bug sẽ bị reject sau khi team lead kiểm tra lại.
4. DUPLICATE(trùng lặp): Nếu bug là hợp lệ, thì sau đó team lead sẽ kiểm tra xem lỗi đó đã được log người khác hay chưa. Nếu đã có người khác log nó, thì team lead sẽ đánh dấu nó là duplicate. Còn nếu nó chưa được báo cáo bởi tester khác thì team lead sẽ thực hiện tìm kiếm nó trong scope => nếu cùng một bung được báo cáo bởi hai hay nhiều tester thì lỗi được báo cáo sau sẽ được đánh dấu là duplicate.
5. DEFERRED(hoãn lại): Nếu bug không bị duplicate, lại không thuộc bản release hiện tại thì sẽ được đánh dấu là deferred. Deferred là một bug nhưng nó sẽ được sửa chữa trong bản release tương lai =>Khi một bug là một phần của bản release tương lai thì nó sẽ được đánh dấu là deferred.
6. ASSIGNED(gán Bug): Khi bug tìm thấy là hợp lệ, duy nhất và thuộc bản release hiện tại, thì team lead sẽ gán bug đó cho Developer.
7. FIX: Khi nhận được bug từ team lead, developer sẽ thực hiện thay đổi để fix bug cho đúng với yêu cầu, và đẩy lại cho tester kiểm tra lịa lỗi đó.
8. RE-TESTING (test lại): Sau khi fix bug xong, và chức năng/tính năng đã sẵn sang để kiểm thử, thì tester sẽ thực hiện lại những testcase lỗi và xác minh lại xem nó đã chạy đúng hay chưa. Việc này gọi là Re-Testing.
9. CLOSED: Khi bug đã được fix, đã được kiểm thử lại và nó chạy đúng như yêu cầu thì tester sẽ đánh dấu nó là Closed.
10. RE-OPEND: Có 2 tình huống mà chúng ta cần phải re-open lại bug:

* Tình huống 1: Khi developer fix bug và tester thực hiện test lại nó,nhưng sau khi re-test, bug đó vẫn xảy ra thì tester sẽ Re-opend lại bug và assign cho developer.
* Tình huống 2: Có trường hợp lỗi đã fix và được close xuất hiện lại. Trong trường hợp này, tester cần Re-opend lại bug đã close và gán nó cho developer

**1.3. Kiểm thử ứng dụng Web**

**1.3.1. Tổng quan**

Các ứng dụng Web đã được phát triển và trở thành một nền tảng kết nối thông tin thiết yếu trong nhiều doanh nghiệp. Các ứng dụng Web đóng vai trò quyết định của thương mại điện tử, trao đổi thông tin.

Để có thể đạt được điều này các ứng dụng Web cần phải có hiệu năng cao, đáng tin cậy,… Việc đưa ra một ứng dụng web hoàn hảo cho những người đang và sẽ sử dụng ứng dụng đã trở thành một thử thách chính trong đảm bảo chất lượng.

Kiểm thử là một trong những công việc quan trọng để đánh giá chất lượng của một sản phẩm và đương nhiên là các ứng dụng Web cũng không phảo là ngoại lệ. Các phương pháp kiểm thử thông thường là các kỹ thuật tập trung vào đánh giá các chức năng yêu cầu của ứng dụng.

Tuy nhiên, không thể nào tập trung được vào hết tất cả các chức năng yêu cầu. Bởi có rất nhiều chức năng quan trọng cho người sử dụng ứng dụng như: tính hiệu năng, tính dễ sử dụng, độ tin cậy và tính bảo mật cần được xem xét. Những yêu cầu và mong đợi của người sử dụng, những vấn đề về nền tảng và cấu hình, mô hình nghiệp vụ, sự phát triển và chi phí cho việc kiểm thử là những vấn đề thường hay gặp phải và thay đổi liên tục đổi xuyên suốt chu trình của một ứng dụng Web. Vì thế, cần thiết phải phát triển một chiến lược hiệu quả cho việc kiểm thử mà có thể bao quát được giới hạn tổng thể và rộng lớn của những yêu cầu, chức năng cho một ứng dụng Web qua đó có thể giúp cho việc cài đặt, hoàn thành ứng dụng cũng như tránh đư ợc các rủi ro có thể gặp.

**1.3.2. Đặc điểm về chất lượng của một ứng dụng Web**

Người sử dụng không chỉ mong đợi chương trình của họ sẽ vận hành một cách ổn định, chính xác mà họ còn yêu cầu một số chức năng nào đó phải luôn sẵn sàng trên 24 giờ trong 1 ngày và 7 ngày trong tuần. Hơn nữa, người sử dụng còn mong đợi ở phần mềm những ưu điểm sau: tính dễ sử dụng, độ tin cậy, tốc độ, tương thích với các hệ thống khác nhau và tương thích với các phiên bản trong tương lai. Còn với một ứng dụng Web, thì những yêu cầu về chất lượng bao gồm:

* + - * Yêu cầu về chức năng: Sự hiện diện của các chức năng đáp ứng những yêu cầu được xác định. Các yêu cầu cần có nữa là tính phù hợp, chính xác, khả năng tương tác, tuân thủ và bảo mật.
      * Yêu cầu về độ tin cậy: Khả năng của một ứng dung để duy trì sự hiệu quả của nó trong một điều kiện cụ thể và trong một khoảng thời gian xác định.
      * Yêu cầu về khả năng sử dụng: Tính dễ sử dụng và hiệu quả của một ứng dụng. Vấn đề này có thể được thẩm định bởi một nhóm người dùng giả định.
      * Yêu cầu về hiệu quả: Tỷ lệ giữa mức độ hiệu quả của một ứng dụng và các tài nguyên mà nó sử dụng trong các điều kiện cụ thể.Các yêu cầu về chất lượng đóng một vai trò thiết yếu khi thử nghiệm các ứng dụng Web. Mặc dù nhìn chung thì chúng tương tự như những yêu cầu về chất lượng cho các hệ thống phần mềm truyền thống. Tuy nhiên, chúng có thể có mức độ đòi hỏi cao hơn về chiều sâu.

Do ý nghĩa quan trọng của các đặc điểm về chất lượng và sự khác biệt ở cách mà chúng được kiểm thử, nhiều phương pháp để kiểm thử một ứng dụng Web tập trung vào một vài đặc điểm. Tuy nhiên, tất cả các đặc điểm đều quan trọng đối với một ứng dụng Web. Và công việc kiểm thử phải đảm bảo được những yêu cầu cài đặt thành công.

**1.3.3. Kiểm thử Website**

Các trang web là các ứng dụng client/server (các máy chủ web và “trình duyệt” của khách hàng) cơ bản. Cần xem xét đến sự tương tác giữa các trang html, dịch vụ web, thông tin liên lạc đã được mã hóa, kết nối Internet, tường lửa, các ứng dụng chạy trong các trang web (như javascript, flash, plug-in các ứng dụng), nhiều ứng dụng có thể chạy ở phía máy chủ, vv… Ngoài ra, có rất nhiều loại máy chủ và các trình duyệt, các phiên bản khác nhau của từng, nhỏ nhưng đôi khi sự khác biệt giữa chúng, các biến đổi trong tốc độ kết nối, công nghệ thay đổi nhanh chóng, các tiêu chuẩn và giao thức. Kết quả cuối cùng là kiểm thử các trang web có thể trở thành một nỗ lực lớn đang diễn ra. Những điều cần xem xét khác có thể bao gồm:

- Tải mong muốn trên máy chủ, loại hiệu năng yêu cầu. Những loại công cụ cần thiết để thử nghiệm hiệu năng.

* + - * Đối tượng mục tiêu. Loại và phiên bản nào của trình duyệt sẽ được họ sử dụng, và mức độ bao quát sẽ kiểm thử cho các thay đổi. Loại kết nối. Họ nằm trong nội bộ của một tổ chức hoặc bên ngoài internet.
      * Những hiệu năng nào mong muốn trên phía máy khách.
      * Thời gian nghỉ cho server và việc bảo trì/nâng cấp nội dung sẽ cho phép.
      * Loại bảo mật nào (firewalls, encryption, passwords, functionality, v.v…) sẽ được yêu cầu và mong muốn.
      * Các yêu cầu quốc tế hóa/nội địa hóa/ngôn ngữ.
      * Mức độ tin cậy kết nối Internet của trang web. Và sự ảnh hưởng đến hệ thống dự phòng hoặc các yêu cầu kết nối dự phòng.
      * Qui trình sẽ được yêu cầu để quản lý cập nhật nội dung trang web, và các yêu cầu cho việc bảo trì, theo dõi và điều khiển nội dung, đồ họa, link, v.v… của trang web.
      * HTML và mô tả chi tiết. Mức độ chặt chẽ, và các thay đổi dự định.
      * Tiêu chuẩn hoặc yêu cầu đối với sự giao diện và/hoặc đồ họa của một phần hoặc một trang web.
      * Quá trình phát triển thực tế/tiêu chuẩn sử dụng cho các thành phần và nhận dạng của trang web (components và identifiers), nó có thể tác động đáng kể đến việc kiểm thử tự động.
      * Liên kết nội bộ và bên ngoài sẽ được xác nhận và cập nhật và mức độ thường xuyên.
      * Cả hệ thống sản phẩm có thể được kiểm thử hết, hoặc sẽ tách riêng một phần hệ thống. Trình duyệt caching (lưu bộ nhớ đệm), thay đổi thiết lập tùy chọn trong trình duyệt, thay đổi kết nối, và vấn đề "tắc nghẽn giao thông" của Internet trong thế giới thực sẽ được tính toán cho vào kiểm thử
      * Mở rộng hay tùy biến nhật ký (log) của máy chủ và các yêu cầu báo cáo; chúng có được coi là một phần của hệ thống.
      * Các thành phần như flash, applet, javascript, ActiveX component, v.v… bảo trì, theo dõi

**1.3.4. Phân loại kiểm thử ứng dụng Website**

Một ứng dụng web thường có rất nhiều nhóm người sử dụng với nhiều nền tảng khác nhau (hệ điều hành, trình duyệt…), người ta cũng rất khó có thể đoán được số lượng người sử dụng một ứng dụng web là bao nhiêu, rồi thời gian hồi đáp yêu cầu của người sử dụng đối với ứng dụng là một trong những yếu tố mang tính quyết định thành bại của ứng dụng…dẫn đến việc kiểm thử ứng dụng web sẽ có những khác biệt nhất định so với kiểm thử phần mềm truyền thống. Trong đó, kiểm thử giao diện người dùng, kiểm thử hiệu năng và kiểm thử bảo mật là những loại kiểm thử mà ứng dụng web cần chú trọng. Dưới đây người viết xin giới thiệu các bước kiểm thử ứng dụng web:

**1.3.4.1. Kiểm thử chức năng.**

Việc kiểm thử chức năng yêu cầu tester thực hiện test 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 cookies và xác minh HTML/CSS.

* + - * + Kiểm tra các liên kết (links) bao gồm kiểm tra liên kết ngoài trang web, liên kết nội bộ, liên kết tới các vị trí trong cùng trang, liên kết sử dụng để gửi email tới admin hoặc người dùng khác trong trang…
        + Kiểm tra form của các trang: form là phần cơ bản của các trang web, nên cần được kiểm tra một cách kỹ càng, bao gồm các yêu cầu sau:

Không nhập gì vào các trường bắt buộc thì sao?

Giá trị mặc định của các trường là gì?

Nhập đầu vào không đúng validate của các trường thì sao?

Thao tác trên các trường: xem, nhập, lưu, sửa, xóa…

- Kết nối cơ sở dữ liệu: yêu cầu kiểm tra tính toàn vẹn của dữ liệu khi tạo, sửa, xóa form…hoặc bất cứ chức năng nào có liên quan tới cơ sở dữ liệu. Khi truy vấn cơ sở dữ liệu thì kết quả trả về có tương ứng với kết quả nhìn thấy được trên giao diện không, dữ liệu có được lấy và cập nhật chính xác không?...

* + - * + Kiểm thử cookies: cookies là các tệp được tạo bởi trang web đã truy cập để lưu trữ thông tin duyệt web, như các tùy chọn trang web hoặc thông tin đăng nhập của người dùng. Người dùng có thể tùy chỉnh trên trình duyệt nhằm quản lý cookies, thực hiện các thao tác cho phép lưu, hoặc xóa, hoặc chặn…để kiểm tra các tính năng lưu hoặc không lưu trạng thái đăng nhập, tính năng bảo mật của ứng dụng web.
        + Xác minh HTML/CSS: việc xác minh này đặc biệt quan trọng khi developer thực hiện tối ưu hóa trang web cho các công cụ tìm kiếm, chủ yếu liên quan tới lỗi cú pháp HTML. Tester sẽ kiểm tra xem trang web có được nhận diện bởi các công cụ tìm kiếm khác nhau hay không (ví dụ: Google, Yahoo, Bing…)

**1.3.4.2.Kiểm thử tính khả dụng**

Tính khả dụng của trang web được định nghĩa là trang web dễ sử dụng, có hướng dẫn sử dụng rõ ràng, rành mạch, mỗi trang đều có menu chính và menu này phải nhất quán. Tester cần lưu ý những điều này.

Ngoài ra, khi kiểm thử tính khả dụng, tester còn cần thực hiện kiểm thử các điều khiển chuyển hướng (như button, text box, text link, bread crum…), nội dung của trang web phải dễ hiểu và thân thiện với người sử dụng. Không có lỗi chính tả, không khó để đọc chữ, hình ảnh được sắp xếp gọn gàng, hợp lý.

**1.3.4.3.Kiểm thử giao diện**

Các giao diện chính bao gồm:

* + - * + Giao diện web server và server ứng dụng
        + Giao diện server ứng dụng và giao diện server dữ liệu

Kiểm tra tất cả các tương tác giữa các server. Nếu server dữ liệu hoặc web server trả lại bất kỳ báo lỗi nào cho bất kỳ truy vấn nào từ server ứng dụng thì ngay lập tức server ứng dụng phải nhận được và cho hiển thị cảnh báo tới người dùng. Kiểm tra các trường hợp giao dịch bị ngắt đột ngột do người dùng, hoặc kết nối tới server bị gián đoạn, bị khởi động lại…

**1.3.4.4. Kiểm thử khả năng tương thích**

Yêu cầu thực hiện các kiểm tra như sau:

* + - * + Tương thích với trình duyệt (trên máy tính và điện thoại di động): Người dùng khác nhau có thể sử dụng trình duyệt khác nhau tùy theo nhu cầu, thói quen của họ. Cần phải kiểm tra ứng dụng web trên càng nhiều trình duyệt càng tốt (IE, Firefox, Chrome, Safari, Opera…) để kiểm tra sự tương thích. Kiểm tra trên cả các phiên bản khác nhau của trình duyệt. Kiểm tra trên cả trình duyệt của thiết bị điện thoại thông minh. Nếu ứng dụng chạy tốt hơn, hoặc có ưu tiên tương thích hơn với trình duyệt nào đó thì cần có thông báo tới người dùng.
        + Tương thích với hệ điều hành: một số chức năng của ứng dụng có thể không tương thích với một số hệ điều hành, hoặc có những lưu ý khác khi sử dụng, điều này cần phải được kiểm tra kỹ và thông báo cho người dùng được biết.
        + Tương thích với các thiết bị ngoại vi (máy in…): khi người dùng có lệnh in trang thì phải đảm bảo tính chính xác của fonts, cỡ chữ, cỡ giấy…mà người dùng đã chọn.

**1.3.4.5. Kiểm thử hiệu năng**

Bao gồm các yêu cầu:

* + - * + Kiểm thử tải: kiểm thử hiệu năng của ứng dụng với các tốc độ kết nối mạng khác nhau. Kiểm thử khi có nhiều người dùng cùng truy cập hoặc cùng yêu cầu một trang xem hệ thống có thể duy trì hoạt động được không? Hoặc kiểm thử khi người dùng tải lên hoặc tải xuống một số lượng dữ liệu đặc biệt lớn…
        + Kiểm thử áp lực: tức là việc đẩy hệ thống ra ngoài giới hạn của nó, thử làm gián đoạn trang web bằng cách tăng lượng tải cao hơn và kiểm tra xem hệ thống phản ứng như thế nào và phục hồi như thế nào.

**1.3.4.6. Kiểm thử bảo mật**

Một số case cho kiểm thử bảo mật web như sau:Gõ trực tiếp url vào thanh địa chỉ của trình duyệt mà không qua đăng nhập. Các trang nội bộ phải không được mở.

* + - * + Sau khi đăng nhập và mở các trang nội bộ, thay đổi url trực tiếp bằng cách đổi tham số ID của trang tới trang thuộc quyền người dùng đã đăng nhập khác. Truy cập phải bị từ chối bởi người dùng này không thể xem trang thống kê của người dùng khác.
        + Thử các giá trị đầu vào không hợp lệ trong các trường username, password. Hệ thống phải báo lỗi.
        + Các thư mục web, các tệp tin không được truy nhập trực tiếp mà không có tùy chọn “Download”.
        + Kiểm tra CAPTCHA cho các đăng nhập tự động
        + Tất cả các phiên giao dịch, các thông báo lỗi, các hành vi cố gắng xâm phạm an ninh phải ghi trong log và lưu tại web server.

**CHƯƠNG 2: KIỂM THỬ TỰ ĐỘNG**

**2.1. Tổng quan về kiểm thử tự động**

Kiểm thử phần mềm tốn nhiều chi phí nhân công, thời gian. Trong một số dự án, chi phí kiểm thử phần mềm chiếm 50% tổng giá trị của dự án. Nếu cần ứng dụng an toàn hơn, chi phí kiểm thử còn cao hơn nữa.

Do đó một trong các mục tiêu của kiểm thử là tự động hóa nhiều, nhờ đó mà giảm thiểu chi phí, giảm lỗi, đặc biệt giúp việc kiểm thử hồi qui dễ dàng và nhanh chóng hơn.

Tự động hóa việc kiểm thử là dùng phần mềm điều khiển việc thi hành kiểm thử, so sánh kết quả có được với kết quả mong muốn, thiết lập các điều kiện đầu vào, các kiểm soát kiểm thử và các chức năng báo cáo kết quả...

**2.1.1. Khái niệm**

Kiểm thử tự động là việc sử dụng các công cụ để thực hiện các test case. Kiểm thử tự động cũng có thể nhập dữ liệu thử nghiệm vào hệ thống kiểm thử, so sánh kết quả mong đợi với kết quả thực tế và tạo ra các báo cáo kiểm thử chi tiết. Là quá trình thực hiện một cách tự động các bước trong bản kiểm thử, nhằm rút ngắn thời gian kiểm thử.

**2.1.2. Kiểm thử tự động sử dụng khi nào?**

Kiểm thử tự động được sử dụng khi:

* + - * + Các trường hợp kiểm thử được thực hiện lặp đi lặp lại để đảm bảo tính năng của

phần mềm/sản phẩm.

- Thực hiện ở các trường hợp mà kiểm thử thủ công khó thực hiện.

- Các trường hợp kiểm thử cần tốn nhiều thời gian.

**2.1.3. Kiểm thử tự động sử dụng ở đâu?**

- Kiểm thử tự động sử dụng trong các giai đoạn kiểm thử:

* Unit Testing (Kiểm thử đơn vị).
* Integration Testing (Kiểm thử tích hợp).
  + - * + Sử dụng trong các loại kiểm thử:
  + Smoke Testing (Kiểm thử khói).
  + Functional Testing (Kiểm thử chức năng).
  + Regression Testing (Kiểm thử hồi quy).
    - * + Và sử dụng trong kỹ thuật kiểm thử: Black Box Testing (Kiểm thử hộp đen).

**2.1.4. Tại sao phải kiểm thử tự động?**

- Kiểm thử tự động sử dụng các công cụ có thể ghi lại bộ kiểm tra này và phát lại nó theo yêu cầu.

- Tiết kiệm thời gian kiểm thử.

- Tự động hóa không cần can thiệp của con người nên có thể chạy tự động kiểm tra mà không cần giám sát.

- Tự động tăng tốc độ thực hiện kiểm tra

- Tự động hóa giúp tăng phạm vi kiểm tra

- Kiểm tra bằng tay có thể trở nên nhàm chán và do đó dễ bị lỗi.

- Cải thiện độ chính xác

- Nhanh hơn 70% so với kiểm tra thủ công

* + - * + Giảm sự nhàm chán cho con người.
        + Rèn luyện kỹ năng lập trình cho lập trình viên.

**2.2. Các công cụ được sử dụng trong kiểm thử tự động**

Có rất nhiều công cụ được sử dụng để tiến hành kiểm thử tự động, sau đây em xin giới thiệu về một số công cụ em sử dụng trong quá trình chạy automation testing.

**2.2.1. Selenium**

**2.2.1.1. Selenium là gì?**

Selenium là bộ kiểm thử tự động miễn phí (mã nguồn mở) dành cho các ứng dụng web trên các trình duyệt và nền tảng khác nhau. Selenium không chỉ là một công cụ duy nhất mà còn là một bộ các công cụ giúp những người kiểm thử tự động hóa các ứng dụng dựa trên web hiệu quả hơn.

**2.2.1.2. Thành phần của Selenium**

Selenium là một bộ phần mềm, mỗi bộ đáp ứng nhu cầu kiểm thử khác nhau. Selenium gồm có 4 phần: Selenium IDE – Selenium Remote Control (Selenium 1) – Web Driver – Selenium Grid.

1. Selenium IDE: Selenium Integreted Development Environment (IDE) là một plugin của Firefox cho phép người kiểm thử ghi lại những hành động của họ theo quy trình làm việc mà họ cần phải kiểm tra.
2. Selenium Remote Control: là framework thử nghiệm chủ đạo cho phép nhiều hành động của trình duyệt hơn và thực hiện tuyến tính. Nó sử dụng toàn bộ chức năng của ngôn ngữ lập trình như Java, C#, PHP, Python, Ryby và PERL để tạo nhiều kiểm tra phức tạp hơn.
3. Web Driver: là sự kế thừa của Selenium RC để gửi các lệnh trực tiếp tới trình duyệt và xuất ra các kết quả.
4. Selenium Grid: là một công cụ được dùng để chạy các kiểm tra song song qua các máy khác nhau và các trình duyệt khác nhau cùng một lúc mà kết quả trong thời gian thực thi giảm thiểu.

**2.2.1.3. Các đặc điểm của Selenium**

- Mã nguồn mở. Phải nói điểm này là điểm mạnh nhất của Selenium khi so sánh với các test tool khác. Vì là mã nguồn mở nên chúng ta có thể sử dụng mà không phải lo lắng về phí bản quyền hay thời hạn sử dụng.

- Cộng đồng hỗ trợ. Vì là mã nguồn mở nên Selenium có một cộng đồng hỗ trợ khá mạnh mẽ. Bên cạnh đó, Google là nơi phát triển Selenium nên chúng ta hoàn toàn có thể yên tâm về sự hổ trợ miễn phí khi có vấn đề về Selenium. Tuy nhiên, đây cũng là một điểm yếu của Selenium. Cơ bản vì là hàng miễn phí, cộng đồng lại đông nên một vấn đề có thể nhiều giải pháp, và có thể một số giải pháp là không hữu ích. Mặc khác, chúng ta không thể hối thúc hay ra deadline cho sự hỗ trợ.

- Selenium hỗ trợ nhiều ngôn ngữ lập trình.

- Selenium hỗ trợ chạy trên nhiều OS khác nhau với mức độ chỉnh sửa script hầu như là không có. Thực sự thì điều này phụ thuộc phần lớn vào khả năng viết script của chúng ta.

- Chạy test case ở backround. Khi chúng ta thực thi một test scrpit, chúng ta hoàn toàn có thể làm việc khác trên cùng một PC. Điều này hỗ trợ chúng ta không cần tốn quá nhiều tài nguyên máy móc khi chạy test script.

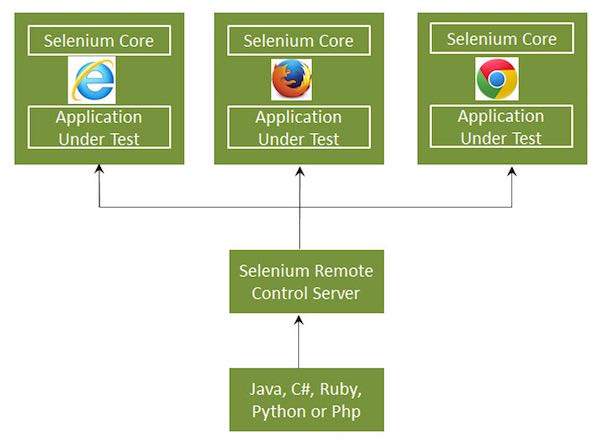
- Không hỗ trợ Win app. Selenium thực sự chỉ hỗ trợ chúng ta tương tác với Browser mà không hỗ trợ chúng ta làm việc với các Win app, kể cả Win dialog như Download/Upload – ngoại trừ Browser Alarm. Vậy nên, để xử lý các trường hợp cần tương tác với hệ thống hay một app thứ ba, chúng ta cần một hay nhiều thư viện khác như AutoIt hay Coded UI.

**2.2.1.4. Cấu trúc Selenium RC**

Selenium Remote Control (RC) là dự án Selenium chính trong một thời gian dài trước khi Selenium WebDriver (Selenium 2.0) ra đời. RC cho phép chúng ta viết các kiểm thử giao diện của ứng dụng Web tự động với sự giúp đỡ của các ngôn ngữ lập trình như Java, C#, Perl, Python, PhP để tạo ra các ca kiểm thử phức tạp hơn như đọc và viết các tập tin, truy vấn cơ sở dữ liệu và gửi mail kết quả kiểm thử.

Selenium RC làm việc theo cách mà thư viện client có thể giao tiếp với Selenium RC Server thông qua mỗi Selenium Command để thi hành. Sau đó Server thông qua Selenium Command tới trình duyệt để sử dụng các lệnh Selenium – Core Javascript.

Các trình duyệt thực thi Selenium Command sử dụng trình thông dịch Javascript của nó.

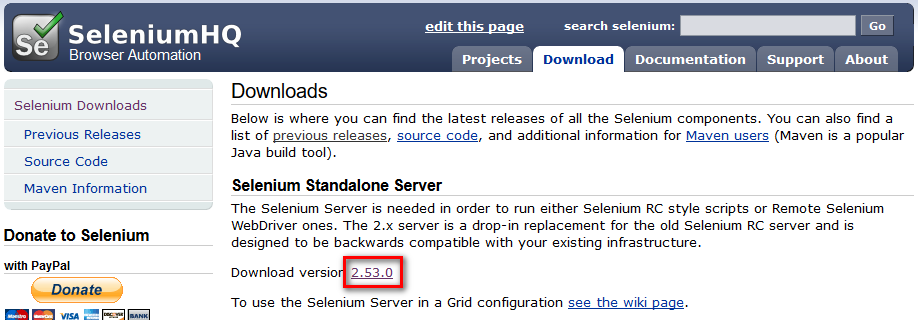


**Selenium RC đi kèm theo hai phần:**

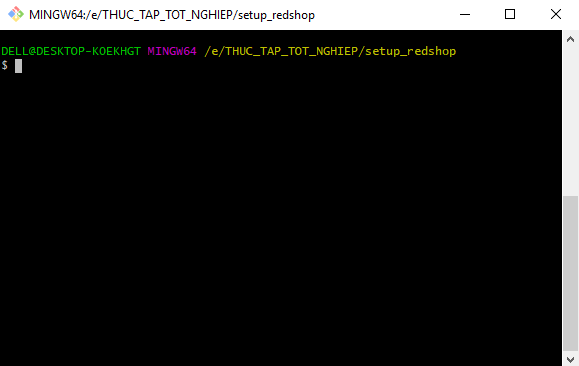
* + - * + Selenium Server khởi động và tắt các trình duyệt. Thêm vào đó, nó diễn dịch và thực thi các lệnh Selenium. Nó cũng hoạt động như một HTTP proxy bằng cách chặn và xác minh các tin nhắn HTTP được thông qua, giữa trình duyệt và ứng dụng kiểm thử.
        + Client libraries: cung cấp một giao diện giữa mỗi một ngôn ngữ lập trình (Java, C#, Perl, Python, PHP) với Selenium-RC Server. Ví dụ: Eclipse, IntelliJ IDEA, PhpStorm,…

**Cấu hình Selenium RC:**

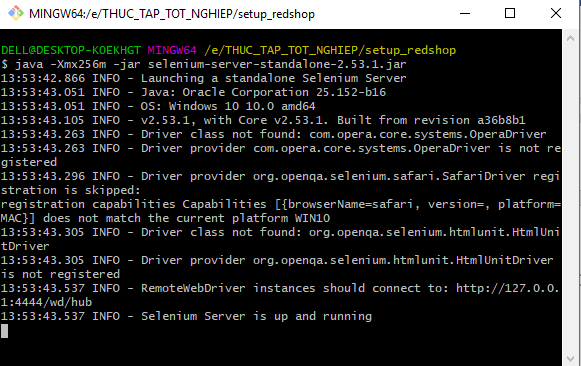
**Bước 1:** Truy cập trang để tải về Selenium Server bằng cách bấm vào phiên bản hiện tại như hình dưới đây.



**Bước 2:** Sauk hi tải chúng về ta cần khởi động Selenium Server bằng cách mở Command Pormpt hoặc Git Bash và điều hướng tới thư mục chứa file JAR đã tải về.



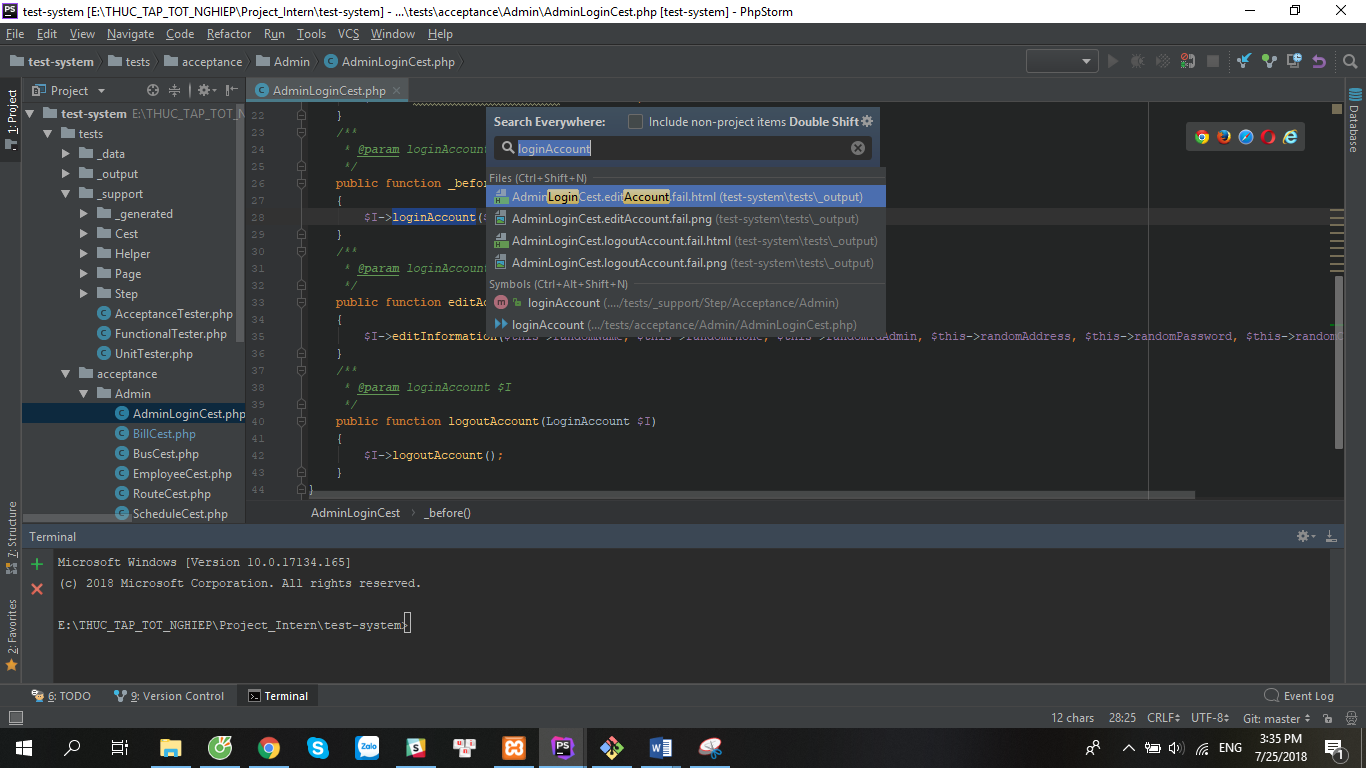
**Bước 3:** Để khởi động server, sử dụng câu lệnh “java -Xmx256m -jar selenium-server-standalone-2.53.1.jar”. Nếu JDK đã được cài đặt thành công, bạn sẽ nhận được như hình bên dưới:



Sau khi khởi động Server chúng ta có thể bắt đầu viết các script Selenium RC.

**2.2.2. JetBrains PhpStorm**

JetBrains PhpStorm là một môi trường phát triển tích hợp (IDE) đặc biệt dành cho các nhà phát triển web, những người cần những công cụ thích hợp để chỉnh sửa PHP, HTML, CSS, JavaScript và các tập tin XML.



PhpStorm giúp người dùng tạo và chỉnh sửa mã nguồn bất kể ngôn ngữ lập trình mà họ đang sử dụng. Như bất kỳ trình soạn thảo IDE khác, nó đi kèm với các tính năng cơ bản như đánh dấu trang, hoàn thành mã, phóng to thu nhỏ, các điểm ngắt…Tuy nhiên, nó chứa các tính năng khác nhau như phân tích mã, nhanh chóng chuyển hướng để làm cho công việc của bạn dễ dàng hơn.

**2.2.3. Joomla!**

**2.2.3.1. Giới thiệu về Joomla!**

Joomla! là một hệ thống quản lý nội dung miễn phí và mã nguồn mở (CMS) để xuất bản nội dung web. Qua nhiều năm Joomla! đã giành được nhiều giải thưởng. Nó được xây dựng trên một khung công tác ứng dụng web của mô hình-khung nhìn-điều khiển có thể được sử dụng độc lập với CMS cho phép bạn xây dựng các ứng dụng trực tuyến mạnh mẽ.

Joomla! là một trong những phần mềm trang web phổ biến nhất, nhờ cộng đồng các nhà phát triển và tình nguyện viên toàn cầu, đảm bảo nền tảng thân thiện với người dùng, có thể mở rộng, đa ngôn ngữ, có thể truy cập, đáp ứng, tối ưu hóa công cụ tìm kiếm và hơn thế nữa.

**2.2.3.2.** **Hệ thống quản lý nội dung (CMS) là gì?**

Định nghĩa của CMS là một ứng dụng (dựa trên web), cung cấp khả năng cho nhiều người dùng với các cấp quyền khác nhau để quản lý (tất cả hoặc một phần) nội dung, dữ liệu hoặc thông tin của dự án trang web hoặc ứng dụng mạng nội bộ. Quản lý nội dung đề cập đến việc tạo, chỉnh sửa, lưu trữ, xuất bản, cộng tác, báo cáo, phân phối nội dung trang web, dữ liệu và thông tin.

**2.2.3.3.** **Joomla! trong thế giới thực (trực tuyến)**

Joomla! được sử dụng trên toàn thế giới để cung cấp năng lượng cho hàng triệu trang web thuộc mọi hình dạng và kích thước.

Khám phá các ví dụ về các công ty sử dụng Joomla! Có thể được sử dụng cho:

- Trang web hoặc cổng công ty, mạng nội bộ và mạng nội bộ

- Trang web doanh nghiệp nhỏ

- Tạp chí, báo và tạp chí trực tuyến

- Thương mại điện tử và đặt chỗ trực tuyến

- Trang web chính phủ, phi lợi nhuận và tổ chức

- Trang web hoặc cổng thông tin dựa trên cộng đồng, trường học và nhà thờ

- Trang chủ cá nhân hoặc gia đình ...

**2.2.3.4.** **Là một cơ quan web, làm thế nào để Joomla! giúp tôi?**

Nếu bạn là đại lý phát triển trang web cho khách hàng của bạn, Joomla! là công cụ hoàn hảo cho bạn (hãy xem các lợi ích và tính năng của chúng tôi). Được thiết kế để dễ cài đặt và thiết lập ngay cả khi bạn không phải là người dùng nâng cao. Với đường cong học tập ngắn (chúng tôi cũng cung cấp đào tạo video miễn phí), bạn sẽ có thể nhanh chóng xây dựng trang web cho khách hàng của mình. Sau đó, với số lượng lệnh tối thiểu, bạn có thể trao quyền cho khách hàng của mình để dễ dàng quản lý các trang web của chính họ.

Nếu khách hàng của bạn cần chức năng chuyên biệt, Joomla! có khả năng mở rộng cao và hàng nghìn phần mở rộng (phần lớn miễn phí theo giấy phép GPL) có sẵn trong Joomla!

**2.2.3.5.** **Là một nhà phát triển, tôi có thể sử dụng Joomla! theo cách nâng cao hơn?**

Một số công ty và tổ chức có yêu cầu vượt xa những gì có sẵn trong Joomla!. Trong những trường hợp đó, khung ứng dụng mạnh mẽ của Joomla giúp các nhà phát triển dễ dàng tạo ra các tiện ích tinh vi mở rộng sức mạnh của Joomla thành các hướng hầu như không giới hạn.

Joomla! Framework cho phép các nhà phát triển xây dựng một cách nhanh chóng và dễ dàng:

* Hệ thống kiểm soát hàng tồn kho
* Công cụ báo cáo dữ liệu
* Cầu ứng dụng
* Danh mục sản phẩm tùy chỉnh
* Tích hợp hệ thống thương mại điện tử
* Danh bạ kinh doanh phức tạp
* Hệ thống đặt trước
* Công cụ truyền thông

Kể từ Joomla! dựa trên PHP và MySQL, bạn đang xây dựng các ứng dụng mạnh mẽ trên nền tảng mở mà mọi người đều có thể sử dụng, chia sẻ và hỗ trợ.

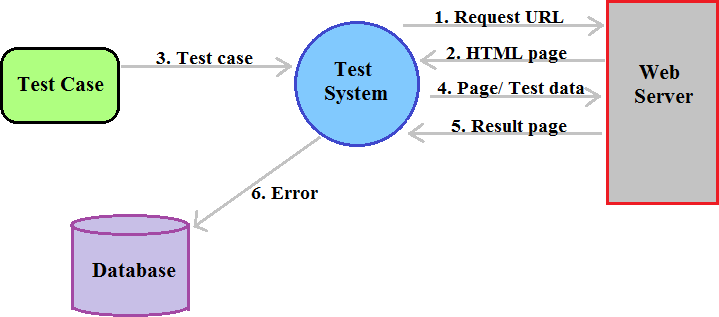
**2.3. Quy trình kiểm thử tự động.**

Bảng sau mô tả rõ hơn các bước thực hiện kiểm thử tự động:

|  |  |  |
| --- | --- | --- |
| **SST** | **Bƣớc thực hiện** | **Mô tả** |
| 1 | Tạo kịch bản kiểm thử | Giai đoạn này dùng công cụ kiểm thử để ghi lại các thao tác lên phần mềm cần kiểm tra và tự động sinh ra kịch bản kiểm thử. |
| 2 | Chỉnh sửa kịch bản kiểm thử | Chỉnh sửa để kịch bản kiểm thử thực hiện kiểm tra theo đúng yêu cầu đặt ra. Cụ thể, làm theo trường hợp kiểm thử cần thực hiện. |
| 3 | Chạy kịch bản kiểm thử | Chạy kịch bản kiểm thử để kiểm tra phần mềm có đưa ra đúng như kết quả mong muốn không |
| 4 | Đánh giá kết quả | Đánh giá kết quả sau khi chạy kịch bản kiểm thử. Kiểm tra và theo dõi lỗi trong đặc tả của kiểm thử tự động. |

## Bảng 2.1 – Quy trình kiểm thử tự động

**2.4. Quy trình kiểm thử tự động của website**



**Hình 2.1 – Quy trình kiểm thử tự động trên website**

Quy trình kiểm thử tự động trên website tuân theo 6 bước:

* Bước 1: Test System gửi yêu cầu URL của website cần test đến Web Server.
* Bước 2: Web Server gửi trả lại một HTML page để biết được các phần tử của website đó.
* Bước 3: Gửi dữ liệu cần kiểm thử đến Test System
* Bước 4: Test System gửi dữ liệu test đến Web Server
* Bước 5: Web Server trả về kết quả sau khi test
* Bước 6: Kết quả đó được đưa vào trong Database. Database là nơi lưu trữ thời gian thực hiện, nguyên nhân dẫn đến kết quả không mong đợi.

Kết quả sau khi chạy kiểm thử tự động

* Pass: Test System hoạt động đúng theo mô tả trong test case, không phát sinh các trường hợp ngoại lệ, đạt kết quả theo như kết quả mong đợi.
* Fail: Test System sau khi hoạt động có kết quả khác kết quả mong đợi được mô tả trong test case tự động. Thường do nhiều yếu tố dẫn đến việc này: dữ liệu sai, không kết nối được với Web Server, … Khi kết quả Fail thì phải phân tích được nguyên nhân dẫn đến kết quả không được mong đợi, để đưa vào Data Base.

**2.5. Test Case trong kiểm thử tự động**

- Test Case là tài liệu để kiểm thử phần mềm.

- Test Case cho phần mềm giúp hướng dẫn người kiểm tra thông qua một chuỗi các bước để xác nhận xem một ứng dụng phần mềm có lỗi và hoạt động theo yêu cầu của người dùng cuối hay không.

- Khi viết các trường hợp kiểm tra, điều quan trọng là đặt mình vào vị trí của người dùng và bao gồm tất cả các chi tiết cần thiết.

- Test Case dùng để test tự động có thể được lựa chọn bằng cách sử dụng các tiêu chí sau đây:

* Rủi ro cao (high risk) – Business Critical test case.
* Những Test Case mà có thể được thực hiện lặp lại.
* Những Test Case rất tẻ nhạt hoặc khó thực hiện bằng tay.
* Test Case tốn thời gian

- Các loại không phù hợp để tự động hóa:

* Test Case được thiết kế mới và chưa được thực hiện bằng tay ít nhất một lần.
* Test Case mà yêu cầu được thay đổi thường xuyên.

**2.6. Ưu điểm và nhược điểm của kiểm thử tự động**

**2.6.1. Ưu điểm của kiểm thử tự động**

* Độ tin cậy cao (Reliability)
* Khả năng lặp (Repeatability): nếu bạn thực thi bằng kiểm thử tự động, chỉ cần nhập dữ liệu cho script chạy và đợi cho đến khi nhận được báo cáo. Với độ ổn định cao, bạn hoàn toàn có thể tin tưởng vào kết quả thực thi của công cụ kiểm thử tự động.
* Khả năng tái sử dụng (Reusability): với một bộ kiểm thử tự động, chúng ta có thể sử dụng cho nhiều phiên bản ứng dụng khác nhau.
* Nhanh (Fast): nếu cần 5 phút để thực thi một test case cách thủ công, có thể bạn cần chưa đầy 30s để thực thi cách tự động.
* Chi phí thấp (Cost Reduction): Nếu áp dụng kiểm thử tự động đúng cách, chúng ta có thể tiết kiệm được rất nhiều chi phí, thời gian và nhân lực.
* Cải thiện hiệu quả: Đầu tiên, lợi ích cụ thể được nói đến là ” Nâng cao hiệu quả”. Khi cần kiểm tra hồi quy hay phải hao phí về mặt thời gian thì kiểm thử tự động mang lại hiệu quả rõ rệt (có thể thực hiện kiểm thử ngay cả khi không có người bất kể ngày hay đêm).
* Cải thiện độ chính xác: Khi dùng kiểm thử tự động, dù có lặp đi lặp lại bao nhiêu lần thì cũng cho ra các thao tác và kết quả giống nhau. Do đó tránh được những rủi ro không cần thiết.Ngoài ra, nếu một lỗi được tìm thấy, nó có thể được tái tạo bằng cách đơn giản là thực hiện cùng một kịch bản tự động, dẫn đến cải thiện khả năng tái lỗi. Kiểm thư tự động còn có tính năng các thao tác test được lưu lại tự động, dễ dàng kiểm tra và cưỡng chế lỗi trong thời gian kiểm thử.

**2.6.2. Nhược điểm của kiểm thử tự động.**

* Các công cụ kiểm thử tự động mặc dù rất thuận tiện về nhiều phương diện nhưng thực tế dù như thế nào đi chăng nữa thì nó cũng không phải là một công cụ có thể thay thế hoàn toàn quá trình kiểm thử. Để thực hiện các thiếp lập tự động thì vẫn cần có con người, phải bỏ công sức, tiền bạc và thời gian.
* Khó mở rộng, khó bảo trì (Poor scalability and maintainability): Trong cùng một dự án, để mở rộng phạm vi cho kiểm thử tự động là khó hơn nhiều so với kiểm thử cách thủ công. Kiểm thử tự động lại không đơn giản như vậy, cập nhật hay chỉnh sửa yêu cầu rất nhiều công việc như debug, thay đổi dữ liệu đầu vào, và cập nhật code mới.
* Vấn đề công cụ và nhân lực (Technology vs. people issues): Cho đến nay công cụ hỗ trợ kiểm thử tự động đã có những bước phát triển mạnh mẽ, chúng ta có các công cụ rất tốt như QTP, Selenium, Test Complete, LoadTest, Jmeter, Visual Studio, … Nhưng nhìn chung vẫn còn rất nhiều mặt hạn chế. Ngoài ra nguồn nhân lực đạt yêu cầu cũng không nhiều.

**CHƯƠNG 3: THIẾT KẾ HỆ THỐNG**