

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN

=====***=====



ĐỒ ÁN TỐT NGHIỆP
NGÀNH KỸ THUẬT PHẦN MỀM
ĐỀ TÀI
KIỂM THỬ TỰ ĐỘNG BẰNG SELENIUM
VÀ ỨNG DỤNG KIỂM THỬ TRANG WEB
DIVIMENTI

GVHD	: TS. Nguyễn Mạnh Cường
Sinh viên	: Nông Minh Hiếu
Mã sinh viên	: 2020606420
Lớp: KTPM4	Khóa: 15

Hà Nội - 2024

MỤC LỤC

DANH MỤC BẢNG BIỂU	5
DANH MỤC HÌNH ẢNH	6
LỜI CẢM ƠN	6
MỞ ĐẦU	7
CHƯƠNG 1: TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM.....	8
1.1. Các khái niệm cơ bản.....	8
1.1.1 Kiểm thử phần mềm.....	8
1.1.2. Quy trình kiểm thử phần mềm	8
1.2. Kế hoạch kiểm thử -Test Plan.....	10
1.3. Test Case	13
1.5. Các cấp độ kiểm thử phần mềm.....	15
1.6. Phân loại kiểm thử	18
1.6.1. Khái niệm	18
1.6.2. Điểm khác nhau giữa kiểm thử thủ công và kiểm thử tự động.....	18
1.7. Quy trình kiểm thử tự động.....	21
1.7.1. Một số Test automation framework khác	23
1.7.2. Tại sao nên lựa chọn Automation testing	24
CHƯƠNG 2: CÔNG CỤ SELENIUM	25
2.1. Giới thiệu công cụ Selenium.....	25
2.1.1 Lịch sử phát triển	25
2.1.2 Khái niệm	25
2.1.3 Những tính năng của Selenium	25

2.1.4 Selenium bao gồm những công cụ nào?	26
2.1.5 Ưu điểm và nhược điểm của Selenium.....	27
2.2 Giới thiệu framework Mocha.....	28
2.3. Cài đặt Selenium Webdriver và môi trường kiểm thử.....	30
2.3.1. Cài đặt Node.js và npm	30
2.3.2. Cài đặt Selenium webdriver.....	33
2.3.3. Cài đặt framework Mocha.....	33
2.4. Cấu trúc của một test script automation trong nodejs.....	34
CHƯƠNG 3. THỰC HIỆN KIỂM THỬ WEBSITE.....	37
3.1. Giới thiệu chương trình.....	37
3.1.1 Giới thiệu về website.....	37
3.1.2 Chức năng của hệ thống.....	37
3.1.3 Một số màn hình giao diện của website.....	42
3.2. Kế hoạch kiểm thử	45
3.3. Kịch bản kiểm thử thủ công trong website	47
3.3.1. Testcase kiểm thử chức năng Đăng ký	49
3.3.2 Testcase kiểm thử chức năng Đăng nhập.....	51
3.3.3. Testcase kiểm thử chức năng Đăng bài	53
3.3.5 Test case chức năng Nhắn tin	56
3.3.6 Trường hợp kiểm thử chức năng Bình luận.....	57
3.4 Kiểm thử tự động các chức năng	59
3.4.1. Kiểm thử tự động chức năng Đăng ký	59
3.4.2. Kiểm thử tự động chức năng Đăng nhập	60

3.4.3 Kiểm thử tự động chức năng Đăng bài	63
3.4.4 Kiểm thử tự động chức năng Nhắn tin.....	64
3.4.5 Kiểm thử tự động chức năng Tìm kiếm	65
3.4.6 Kiểm thử tự động chức năng Bình luận	67
KẾT LUẬN	72
TÀI LIỆU THAM KHẢO.....	73

DANH MỤC BẢNG BIỂU

Bảng 1.1 Điểm khác nhau giữa kiểm thử thủ công và kiểm thử tự động	18
Bảng 3.1 Chức năng hệ thống về phía người dùng	37
Bảng 3.2 Chức năng hệ thống về phía người quản trị	42

DANH MỤC HÌNH ẢNH

Hình 1.1 Quy trình kiểm thử phần mềm	8
Hình 1.2 Các loại kiểm thử	11
Hình 1.3 Các cấp độ kiểm thử phần mềm.....	16
Hình 1.4 Quy trình kiểm thử tự động.....	21
Hình 2. 1 Những công cụ trong Selenium	26
Hình 2.2 Cài đặt Nodejs(1)	30
Hình 2.3: Cài đặt Nodejs(2)	30
Hình 2.4 Cài đặt Nodejs(3)	30
Hình 2.5 Cài đặt Nodejs(4)	31
Hình 2.6 Cài đặt Nodejs(5)	31
Hình 2.7 Cài đặt Nodejs(6)	31
Hình 2.8 Cài đặt Nodejs(7)	32
Hình 2. 9 Cài đặt Nodejs(8)	32
Hình 2.10 Kiểm tra version của nodejs và npm.....	32
Hình 2.11 Folder node_module sau khi cài đặt	33
Hình 2.12 Cài đặt Selenium webdriver.....	33
Hình 2.13 Cài đặt framework Mocha.....	33
Hình 2.14 Import các thư viện cần thiết.....	34
Hình 2.15 Thiết lập môi trường	34
Hình 2.16 Định nghĩa các test cases	35
Hình 2.17 Xử lý data và setup/teardown.....	35
Hình 2.18 Chạy các script	36
Hình 3.1 Giao diện màn hình Đăng ký	42
Hình 3.2 Giao diện màn hình Đăng nhập.....	43
Hình 3.3 Giao diện màn hình Trang chủ.....	43
Hình 3.4 Giao diện màn hình Trang cá nhân	44
Hình 3.5 Giao diện màn hình Trang cá nhân bạn bè	44
Hình 3.6 Kế hoạch kiểm thử website Divimenti.....	45
Hình 3.7 Trường hợp kiểm thử chức năng Đăng ký.....	50
Hình 3.8 Trường hợp kiểm thử chức năng Đăng nhập	52

Hình 3.9 Trường hợp kiểm thử chức năng Đăng bài	54
Hình 3.10 Trường hợp kiểm thử chức năng Tìm kiếm bạn bè	55
Hình 3.11 Trường hợp kiểm thử chức năng Nhắn tin.....	56
Hình 3.12 Trường hợp kiểm thử chức năng Bình luận.....	58
Hình 3.13 Sửa tên file test case Đăng ký	59
Hình 3.14 Kết quả kiểm thử tự động chức năng Đăng ký	59
Hình 3.15 Sửa tên file test case Đăng nhập	60
Hình 3.16 Kết quả kiểm thử tự động chức năng Đăng nhập	61
Hình 3.17 Sửa tên file chức năng Đăng bài	63
Hình 3.18 Kết quả kiểm thử tự động chức năng Đăng bài	63
Hình 3.19 Sửa tên file chức năng Nhắn tin.....	64
Hình 3.20 Kết quả kiểm thử tự động chức năng Nhắn tin	64
Hình 3.21 Kết quả kiểm thử tự động chức năng Đăng ký	65
Hình 3.22 Kết quả kiểm thử tự động chức năng Tìm kiếm bạn bè.....	66
Hình 3.23 Kết quả kiểm thử tự động chức năng Đăng ký	67
Hình 3.24 Kết quả kiểm thử tự động chức năng Bình luận	67
Hình 3.25 Test report	70

LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, sự giúp đỡ dù ít hay nhiều, dù là trực tiếp hay gián tiếp của người khác. Trong suốt thời gian từ khi bắt đầu học tập, em đã nhận được rất nhiều sự quan tâm, giúp đỡ của Thầy Cô, gia đình và bạn bè.

Trước hết em xin gửi tới các thầy các cô khoa Công Nghệ Thông Tin Trường Đại học Công Nghiệp Hà Nội lời chào trân trọng, lời chúc sức khỏe và lời cảm ơn sâu sắc. Với sự quan tâm, dạy dỗ, chỉ bảo tận tình chu đáo của thầy cô, đến nay em đã có thể hoàn thành đồ án tốt nghiệp, đề tài: ***“Kiểm thử tự động bằng Selenium và ứng dụng kiểm thử trang web Divimen”***.

Đặc biệt em xin chân thành cảm ơn thầy giáo TS. Nguyễn Mạnh Cường, người đã hướng dẫn, dìu dắt em tận tình để em hoàn thành tốt đề tài đồ án.

Sau cùng, em xin kính chúc các thầy cô dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

Trong quá trình hoàn thành đồ án tốt nghiệp không thể tránh khỏi thiếu sót, kính mong có sự góp ý từ thầy cô.

Em xin trân trọng cảm ơn!

Sinh viên thực hiện

Nông Minh Hiếu

MỞ ĐẦU

Trong những năm gần đây với sự phát triển rất mạnh mẽ của công nghệ thông tin, ngành công nghệ phần mềm đang chiếm một vị trí hết sức quan trọng trong xu hướng phát triển kinh tế công nghiệp hóa, hiện đại hóa của nước ta. Cùng với sự phát triển ấy các chương trình phần mềm ra đời ngày càng nhiều, đòi hỏi các nhà sản xuất phần mềm phải có một phương pháp để nâng cao chất lượng sản phẩm cũng như tối ưu hiệu suất làm việc để có thể cạnh tranh. Vì vậy kiểm thử phần mềm đang ngày càng đóng vai trò quan trọng trong ngành công nghiệp phát triển phần mềm không chỉ ở Việt Nam và trên thế giới. Với mong muốn có cái nhìn xác thực, rõ ràng hơn về quy trình kiểm thử phần mềm, đảm bảo chất lượng phần mềm và tiếp cận với các công cụ hỗ trợ kiểm thử, giải quyết phần nào vấn đề về thời gian, kinh phí trong việc tìm kiếm lỗi, quản lý lỗi khi tiến hành kiểm thử; đồng thời rèn kỹ năng làm việc, tạo tiền đề định hướng cho tương lai sau khi ra trường. Với những lý do như trên nên em đã lựa chọn đề tài ***“Kiểm thử tự động bằng Selenium và ứng dụng kiểm thử trang web Divimenti”***. Do kiến thức còn hạn hẹp nên bản thân em không tránh khỏi những sai sót, em rất mong nhận được sự đóng góp từ thầy cô để bài báo cáo được hoàn chỉnh hơn.

Nội dung báo cáo gồm 3 chương:

Chương 1: Tổng quan về Kiểm thử phần mềm

Từ kiến thức đã học từ nhà trường, doanh nghiệp, trình bày cơ sở lý thuyết về kiểm thử bao gồm: khái niệm, quy trình kiểm thử, các cấp độ kiểm thử.

Chương 2: Công cụ Selenium

Giới thiệu về công cụ Selenium đặc trưng, cách thức hoạt động, cài đặt, các thành phần và các chức năng thường được sử dụng.

Chương 3: Thực hiện kiểm thử Website

Từ những nội dung đã thu thập được về công cụ Selenium, tiến hành áp dụng vào thực hiện kiểm thử với trang web Devimenti. Qua đồ án lần này, em mong muốn cải thiện thêm kỹ năng kiểm thử phần mềm, cũng như ôn tập lại các kiến thức đã được học trong chương trình của nhà trường.

CHƯƠNG 1: TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM

1.1. Các khái niệm cơ bản

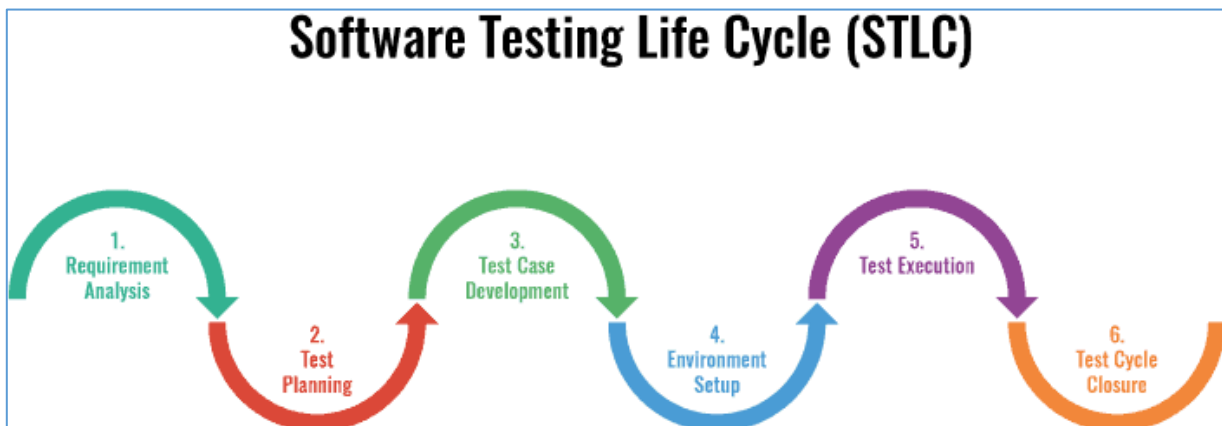
1.1.1 Kiểm thử phần mềm

Kiểm thử phần mềm có nhiều định nghĩa khác nhau đề xuất bởi nhiều tổ chức hay cá nhân khác nhau nhưng tổng quan thì:

Kiểm thử phần mềm là quá trình thực thi một chương trình với mục đích tìm ra lỗi và đảm bảo sản phẩm phần mềm đáp ứng chính xác, đầy đủ đúng theo yêu cầu của khách hàng, yêu cầu của sản phẩm đã đặt ra. Kiểm thử phần mềm cũng cung cấp mục tiêu, cái nhìn độc lập về phần mềm, điều này cho phép việc đánh giá và hiểu rõ các rủi ro khi thực thi phần mềm.

1.1.2. Quy trình kiểm thử phần mềm

Quy trình kiểm thử phần mềm xác định các giai đoạn/ pha trong kiểm thử phần mềm. Tuy nhiên, không có STLC tiêu chuẩn cố định nào trên thế giới, nhưng về cơ bản quy trình kiểm thử bao gồm những giai đoạn sau:



Hình 1.1 Quy trình kiểm thử phần mềm

Requirement analysis – Phân tích yêu cầu:

QA team (Quality Assurance team) có nhiệm vụ phân tích và xác định những yêu cầu của khách hàng, trong đó có yêu cầu về kiểm thử chức năng/phi chức năng của phần mềm. Trong quá trình phân tích, QA team có thể đặt ra câu hỏi để hiểu chính xác hơn về yêu cầu của sản phẩm, đồng thời hỗ trợ đưa ra giải pháp thích hợp cho khách hàng.

Test planning – Lập kế hoạch kiểm thử

Dựa vào tài liệu nhận được trong giai đoạn đầu, Test Lead hoặc Test Manager sẽ lên kế hoạch kiểm thử phần mềm cho QA team để xác định một số yếu tố

Phạm vi dự án: Thời gian thực hiện dự án bao lâu? Trong từng khoảng thời gian sẽ có những công việc gì?

Test case development – Thiết kế kịch bản cho quy trình kiểm thử

Trong giai đoạn này, các Tester sẽ đọc hiểu tất cả các tài liệu, từ đó xác định những việc cần làm, chức năng nào cần test hoặc không. Sau đó, dựa vào kế hoạch và kỹ thuật thiết kế kịch bản kiểm thử, Tester sẽ bắt đầu viết test case. Yêu cầu của test case: Thể hiện tất cả các trường hợp kiểm thử có thể phát sinh để đáp ứng yêu cầu sản phẩm. Ngoài test case, Tester cũng cần chuẩn bị các dữ liệu cần thiết khác như test data, test script, test design, test automation script.

Test environment set up – Thiết lập môi trường kiểm thử

Đây là một trong những giai đoạn đóng vai trò rất quan trọng trong Software Testing Life Cycle (vòng đời phát triển phần mềm). Dựa trên yêu cầu khách hàng và đặc thù của sản phẩm, môi trường kiểm thử sẽ được xác định. Tester cần chuẩn bị smoke test case để kiểm tra môi trường cài đặt đã đáp ứng yêu cầu và sẵn sàng cho giai đoạn kiểm thử tiếp theo hay chưa.

Test execution – Thực hiện kiểm thử

Theo test case đã thiết kế và môi trường kiểm thử đã hoàn tất cài đặt, Tester sẽ báo cáo bug lên tool quản lý lỗi và theo dõi đến khi fix bug thành công. Tiếp đó, Tester thực hiện retest để verify các fix bug và regression test trong trường hợp có sự thay đổi. Sau khi hoàn tất giai đoạn này, các chuyên viên kiểm thử cần có được test results (kết quả kiểm thử) và defect reports (danh sách các lỗi tìm được).

Test cycle closure – Đóng chu trình kiểm thử

Để đóng chu trình kiểm thử, QA team cần có được những tài liệu đã được tổng hợp và hoàn thiện từ những giai đoạn trước: tài liệu phân tích đặc tả yêu cầu, test plan, defect reports, test results... Tiếp đó, QA team sẽ tổng kết, báo

cáo về quá trình kiểm thử, có bao nhiêu bug đã được fix, bug có nghiêm trọng hay không, chức năng nào còn lỗi, chức năng nào đã hoàn thành...

1.2. Kế hoạch kiểm thử -Test Plan

Test plan là tài liệu chi tiết mô tả chiến lược kiểm thử, mục tiêu, lịch trình và ước tính cùng khả năng cung cấp các nguồn lực cho kiểm thử (nhân lực, phần mềm, phần cứng). Kế hoạch kiểm thử có vai trò như một bản kế hoạch chi tiết để thực hiện các hoạt động kiểm thử phần mềm. Hay quy trình xác định được quản lý và giám sát từng bước bởi Test Manager.

Các bước viết kế hoạch kiểm thử như sau:

Bước 1: Phân tích sản phẩm

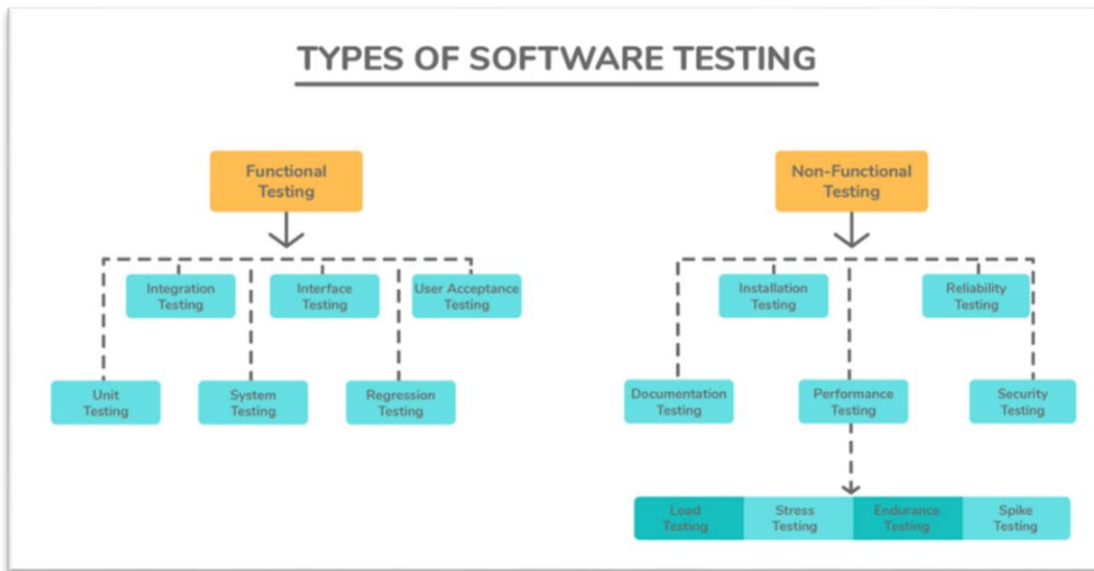
Đây là bước đầu tiên và cũng là bước quyết định cho các tiến trình kiểm thử tiếp theo. Để phân tích sản phẩm, người viết kế hoạch kiểm thử có thể dựa vào những câu hỏi sau đây:

- Ai là người sử dụng sản phẩm này?
- Sản phẩm được sử dụng để làm gì?
- Sản phẩm này sẽ làm việc như thế nào?
- Phần cứng và phần mềm của sản phẩm là gì?

Bước 2: Lập chiến lược kiểm thử

Thứ nhất, người lập kế hoạch kiểm thử phải xác định được phạm vi kiểm thử bao gồm các yêu cầu chính sách của khách hàng, ngân sách dự án, đặc điểm kỹ thuật sản phẩm, kỹ năng và trình độ của nhóm kiểm thử.

Thứ 2, người lập kế hoạch kiểm thử phải xác định được các loại kiểm thử mà nhóm kiểm thử sẽ dùng. Từng loại kiểm thử sẽ được xây dựng để tìm ra một loại lỗi cụ thể. Tùy theo sản phẩm hay loại tính năng trong giai đoạn kiểm thử mà người viết kế hoạch sẽ chọn các loại kiểm thử khác nhau.



Hình 1.2 Các loại kiểm thử

Bước 3: Xác định mục tiêu kiểm thử

Xác định mục tiêu kiểm thử được coi là mục tiêu tổng thể của toàn bộ dự án. Đó là tìm ra càng nhiều lỗi phần mềm càng tốt để đảm bảo rằng phần mềm không có lỗi trước khi phát hành. Việc xác định mục tiêu giúp cho việc kiểm thử diễn ra nhanh chóng và suôn sẻ hơn.

Nếu muốn xác định đối tượng kiểm thử thì cần phải liệt kê các tính năng của phần mềm cần kiểm thử (chức năng, hiệu năng, giao diện,...). Từ đó xác định mục tiêu của kiểm thử dựa trên các tính năng trên.

Bước 4: Xác định các tiêu chí kiểm thử

Tiêu chuẩn hoặc quy tắc để quá trình kiểm thử sản phẩm diễn ra đúng chuẩn đó chính là Test Criteria (Tiêu chí kiểm thử). Gồm có 2 loại tiêu chí chính:

- *Tiêu chí đình chỉ*: đây là tiêu chí khi phát hiện ra lỗi phần mềm trong quá trình kiểm thử. Nếu trong quá trình kiểm thử xuất hiện tiêu chí này thì quá trình kiểm thử sẽ dừng lại cho đến khi xác tiêu chí được xử lý.
- *Tiêu chí thoát kiểm thử*: Nhằm mục đích xác định các tiêu chí thể hiện sự hoàn thành thành công của giai đoạn kiểm thử. Exit Criteria là kết quả được hướng tới là mục tiêu của test để chuyển sang giai đoạn phát triển tiếp theo. Xác định tiêu chí kết thúc dựa trên 2 tỷ lệ:

- Run rate là tỉ số giữa số trường hợp đã kiểm thử trên tổng số trường hợp kiểm thử trên kế hoạch. Chỉ số này bắt buộc là 100%
- Pass rate là tỷ số giữa số các trường hợp kiểm thử pass trên số trường hợp được thực hiện kiểm thử. Yếu tố này phụ thuộc vào phạm vi dự án và Pass rate càng cao càng tốt.

Bước 5: Hoạch định nguồn lực

Các nhà quản lý kiểm thử nên liệt kê và xác định rõ nhân lực cùng tài nguyên hệ thống cho dự án để lên kế hoạch để chạy, hoàn thành dự án hợp lý.

Bước 6: Lập kế hoạch môi trường kiểm thử

Môi trường kiểm thử là một thiết lập của phần mềm và phần cứng để nhóm test sẽ tiến hành các trường hợp kiểm thử. Môi trường kiểm thử bao gồm: người dùng cuối, môi trường kinh doanh, môi trường chạy UI, máy chủ, ... Để thiết lập Test Environment thì bạn cần có sự hợp tác chặt chẽ giữa Test Team và Development Team. Môi trường kiểm thử lý tưởng sẽ cho phép Tester giám sát mọi biến động của phần mềm trong điều kiện sử dụng thực tế.

Bước 7: Lịch trình và dự toán

Ở bước này, cần phân chia cả quá trình thành những nhiệm vụ nhỏ để có thể dễ dàng phân phối thời gian chi tiết cho từng task, theo dõi tiến độ dự án và kiểm soát chi phí.

Bước 8: Xác định phân phối thử nghiệm

Khi tiến hành kiểm thử phần mềm thì các phân phối thử nghiệm là danh sách tất cả các tài liệu, công cụ. Cùng các hiện vật được khai thác để phát triển và dùng để hỗ trợ quá trình Test. Các loại thử nghiệm có thể phát hành gồm:

- Phân phối trước khi thử nghiệm gồm tài liệu kế hoạch kiểm tra và bộ thử nghiệm các trường hợp thử nghiệm.
- Phân phối trong quá trình test: tập lệnh thử nghiệm, dữ liệu test, ma trận truy xuất nguồn gốc cùng nhật ký thực thi và lỗi.
- Phân phối sau thử nghiệm: Kết quả thử nghiệm và báo cáo, quy trình cài đặt, ghi chú phát hành, báo cáo sai sót.

1.3. Test Case

Test case (trường hợp kiểm thử/ ca kiểm thử) là một trường hợp cần kiểm thử, nó bao gồm các thao tác/ hành động trên hệ thống, điều kiện cần (tiên quyết), các giá trị đầu vào, và kết quả mong đợi và các điều kiện kết thúc, được xây dựng cho mục đích hoặc điều kiện kiểm thử riêng biệt, như thực hiện một đường dẫn chương trình riêng hoặc để kiểm tra lại đúng với yêu cầu của spec. Một test case thì nên chỉ kiểm tra một trường hợp, một khía cạnh cụ thể nào đó.

Test case là đảm bảo rằng các tính năng của ứng dụng hoặc phần mềm được thiết kế hoạt động chính xác và đúng như kỳ vọng. Ngoài ra, test case còn giúp xác định phạm vi kiểm thử; phát hiện lỗi để cải thiện chất lượng phần mềm; hỗ trợ quá trình vận hành, bảo trì và cập nhật; ...

Cấu trúc:

1. Test case ID: Nó là duy nhất, là các chuỗi ký tự để định danh cho test case.
2. Function: Dựa theo chức năng của hệ thống có thể chia nhỏ các functions ra để tạo bộ test case rõ ràng hơn.
3. Description: Mô tả tóm tắt testcase sẽ được viết.
4. Pre-Condition: Bất cứ điều gì cần phải thiết lập bên ngoài của ứng dụng trước khi chúng ta có thể test được test case này. Ví dụ chúng ta cần đăng ký tài khoản trước khi đăng nhập
5. Test Case Procedure/ Execution Steps: Đây là mô tả chi tiết từng bước thực hiện và là danh mục quan trọng nhất trong một test case.
6. Test data: Những dữ liệu cần chuẩn bị để kiểm thử.
7. Expected Results: Mô tả kết quả mong đợi của một test case.
8. Result: thông thường sẽ là Pass, Fail hoặc Pending. Đây là kết quả thực tế khi thực hiện kiểm thử theo test case trên môi trường hệ thống.

Ngoài ra có thể thêm một số cột như Ngày thực hiện kiểm thử, Người thực hiện kiểm thử, ghi chú, ...

Cấu trúc của test case có thể thay đổi tùy theo dự án và công ty. Quan trọng nhất là test case phải đảm bảo có đủ các thông tin cần thiết để hiểu, thực hiện và đánh giá kết quả kiểm thử.

Phân loại:

Test case thường được chia thành 4 nhóm chính:

- GUI test case: test case về giao diện
- Positive test case: test case về việc nhập giá trị đúng
- Negative test case: test case về việc nhập giá trị sai
- Combination Test case: test case có nhiều bước đúng sai đan xen nhưng bước cuối cùng luôn đúng

Update Testcase: Khi tài liệu yêu cầu hoặc thiết kế hệ thống đã thay đổi, mình phải cập nhật lại test case dựa trên tài liệu mới nhất đó. Hoặc có khi phải dựa trên tài liệu của bản build hiện tại (không phải là bản mới nhất) nếu khách hàng yêu cầu:

- Thêm mới test case
- Sửa testcase theo tài liệu yêu cầu mới (latest requirements)
- Xóa testcase không còn liên quan đến tài liệu yêu cầu mới

Execute testcase:

- Nếu kết quả thực tế giống với kết quả mong đợi: ghi “Pass” vào cột Status của test case đó.
- Nếu kết quả thực tế không giống với kết quả mong đợi: ghi “False” vào cột Status của test case đó rồi post bug và ghi BugID vào cột BugID.
- Nếu có lý do nào đó làm cho test case nào đó không chạy được (ví dụ như không thực hiện được step nào đó, chức năng cần test chưa làm xong, hoặc có bug làm cho không test được nữa,..v...v...): Ghi “Pending” vào cột Status của test case đó và ghi chú lý do tại sao không test được vào cột Ghi chú.

Log Bug

Nội dung gồm các phần chính sau:

- 1) Title Bug: nên viết ngắn gọn mô tả đầy đủ ý nghĩa của lỗi, tốt nhất khoảng 60 từ trở lại. Tránh để các tiêu đề chung chung
- 2) Description:
 - Pre-condition: điều kiện tiên đề để thực hiện các steps
 - Procedure/Steps: mô tả step tái hiện lỗi. Nên đánh số cho từng bước

để dev có thể dễ dàng thực hiện lại.

- Expected result: kết quả mong đợi
- Actual result: kết quả thực tế
- Evidence: Image, video, logfile
- Môi trường xảy ra lỗi: IE, Chrome, bản build,...
- Mức độ thường xuyên xảy ra lỗi: thỉnh thoảng, luôn luôn...

3) Priority: Set mức độ ưu tiên High/Medium/Low

4) Severity: Mức độ nghiêm trọng Critical/High/Medium/Low

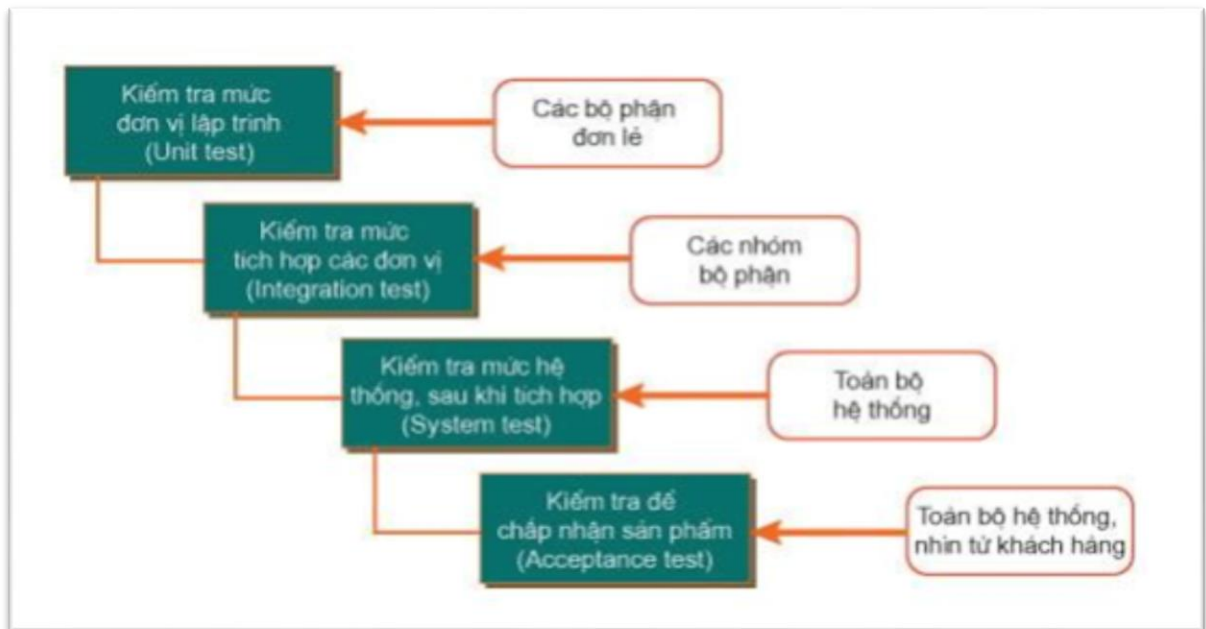
5) Assignee: Assign cho dev fix lỗi

1.5. Các cấp độ kiểm thử phần mềm

Cấp độ kiểm thử phần mềm hay mức độ kiểm thử phần mềm là một quá trình trong đó mọi thành phần của phần mềm hệ thống đều được kiểm tra. Thông qua các mức độ kiểm thử phần mềm sẽ giúp chúng ta đánh giá được chức năng của ứng dụng được những yêu cầu đã chỉ định hay chưa. Đồng thời thông qua mức độ kiểm thử phần mềm sẽ giúp chúng ta tìm và sửa lỗi nhằm đảm bảo sản phẩm tạo ra có chất lượng tốt nhất.

Tất cả các giai đoạn của quá trình phát triển phần mềm đều trải qua bốn mức kiểm thử phần mềm là:

- Kiểm thử đơn vị (Unit Testing) - Dev
- Kiểm thử tích hợp (Integration Testing) - Dev
- Kiểm thử hệ thống (System Testing) – Tester
- Kiểm thử chấp nhận (Acceptance Testing) - Tester



Hình 1.3 Các cấp độ kiểm thử phần mềm

Mức độ 1: Kiểm thử đơn vị - Unit Test

Kiểm thử đơn vị - Unit Testing là giai đoạn đầu tiên trong kiểm thử phần mềm. Với chức năng hoạt động đơn giản, không gây nhiều khó khăn trong việc kiểm thử, ghi nhận và phân tích kết quả do đó nếu phát hiện lỗi thì việc tìm kiếm nguyên nhân và sửa lỗi cũng đơn giản và tốn ít chi phí hơn. Tuy nhiên, kiểm thử mức đơn vị lại tốn nhiều thời gian để thực hiện, chưa phát hiện được các lỗi xảy ra khi tích hợp.

Mục tiêu:

- Xác định mỗi đơn vị phần mềm có đang thực hiện theo đúng thiết kế ban đầu hay không.
- Thông qua thử nghiệm sẽ giúp khắc phục những phát sinh do việc thay đổi hay bảo trì code.
- Unit Test giúp tiết kiệm chi phí, thời gian và thể diện khi phát hiện ra lỗi.

Mức độ 2: Integration Testing – Kiểm thử tích hợp

Mỗi dự án phần mềm được hoàn thành bởi rất nhiều module do nhiều người code khác nhau. Integration Testing là cấp độ kiểm thử phần mềm tích

hợp của các đơn vị riêng lẻ được kết hợp và thử nghiệm thành một nhóm thông qua việc tập trung vào kiểm tra truyền dữ liệu giữa các module.

Mục tiêu:

- Phát hiện lỗi giao tiếp xảy ra giữa các Unit cũng như lỗi của từng Unit
- Tích hợp các Unit đơn lẻ thành các hệ thống nhỏ và cuối cùng là nguyên hệ thống hoàn chỉnh chuẩn bị cho kiểm tra ở mức hệ thống

Một số phương pháp kiểm thử tích hợp:

- Phương pháp kiểm thử Bigbang
- Phương pháp kiểm thử Topdown
- Phương pháp kiểm thử Bottom up
- Phương pháp kiểm thử Sandwich

Mức độ 3: System Testing – Kiểm thử hệ thống

System Testing là giai đoạn thứ 3 của kiểm thử phần mềm cho phép phần mềm hoàn chỉnh và tích hợp được kiểm tra. System Testing tập trung nhiều hơn vào các chức năng của toàn bộ hệ thống. Kiểm thử hệ thống bao gồm kiểm thử chức và kiểm thử phi chức năng.

Mục tiêu:

- System Test kiểm tra thiết kế và toàn bộ hệ thống sau khi tích hợp có tuân thủ những yêu cầu đã được định sẵn trước đó .
- System Test kiểm thử cả các hành vi chức năng của phần mềm lẫn các yêu cầu về chất lượng như độ tin cậy, tính tiện lợi khi sử dụng, hiệu năng và bảo mật.

Mức độ 4: Acceptance Testing – Kiểm thử chấp nhận

Sau khi kiểm tra hệ thống đã sửa tất cả hoặc hầu hết các lỗi, hệ thống sẽ được gửi đến người dùng hoặc khách hàng để kiểm tra chấp nhận. Về cơ bản kiểm thử chấp nhận cũng khá giống kiểm thử hệ thống nhưng được thực hiện bởi khách hàng.

Mục đích của Acceptance Testing đó là xác nhận lại sự tin tưởng vào hệ thống, các đặc tính thuộc về chức năng hoặc phi chức năng của hệ thống.

Có 2 loại kiểm thử chấp nhận đó là Alpha Testing và Beta Testing.

- Kiểm thử alpha: được thực hiện tại nơi phát triển phần mềm bởi những người trong tổ chức nhưng không tham gia phát triển phần mềm.
- Kiểm thử beta: được thực hiện tại bởi khách hàng/ người dùng cuối tại địa điểm của người dùng cuối.

1.6. Phân loại kiểm thử

Có hai phương thức kiểm thử chính là Manual testing (kiểm thử thủ công) và Automation Testing (kiểm thử tự động). Kiểm thử tự động và kiểm thử thủ công là những bước vô cùng quan trọng để đảm bảo một dự án hoạt động trơn tru và hiệu quả. Do vậy vai trò của Tester trong cả hai lĩnh vực đều quan trọng như nhau.

1.6.1. Khái niệm

Kiểm thử thủ công: tester làm mọi công việc hoàn toàn bằng tay, từ viết test case đến thực hiện test, mọi thao tác như nhập điều kiện đầu vào, thực hiện một số sự kiện khác như click nút và quan sát kết quả thực tế, sau đó so sánh kết quả thực tế với kết quả mong muốn trong test case, điền kết quả test. Hiện nay, phần lớn các tổ chức, các công ty phần mềm, hoặc các nhóm làm phần mềm đều thực hiện kiểm thử thủ công là chủ yếu.

Kiểm thử tự động: thực hiện kiểm thử phần mềm bằng một chương trình đặc biệt với rất ít hoặc không có sự tương tác của con người, giúp cho người thực hiện việc kiểm thử phần mềm (tester) không phải lặp đi lặp lại các bước nhàm chán. Công cụ kiểm thử tự động có thể lấy dữ liệu từ file bên ngoài (excel, csv...) nhập vào ứng dụng, so sánh kết quả mong đợi (từ file excel, csv...) với kết quả thực tế và xuất ra báo cáo kết quả kiểm thử.

1.6.2. Điểm khác nhau giữa kiểm thử thủ công và kiểm thử tự động

Bảng 1.1 Điểm khác nhau giữa kiểm thử thủ công và kiểm thử tự động

Thông số	Kiểm thử tự động	Kiểm thử thủ công
Định nghĩa	Kiểm thử tự động sử dụng các công cụ tự động để thực hiện các trường hợp kiểm thử.	Các trường hợp kiểm thử được thực hiện bởi tester trên phần mềm

Thời gian xử lý	Kiểm thử tự động nhanh hơn đáng kể so với phương pháp kiểm thử thủ công.	Các trường hợp kiểm thử được thực hiện bởi tester trên phần mềm
Đầu tư ban đầu	Đầu tư ban đầu trong kiểm thử tự động cao hơn.	Đầu tư ban đầu trong kiểm thử thủ công là tương đối thấp hơn kiểm thử tự động.
Độ tin cậy	Kiểm thử tự động là một phương pháp đáng tin cậy, vì được thực hiện bởi các công cụ và scripts nên chính xác và không gây nhầm lẫn	Kiểm thử thủ công có thể bị nhầm lẫn và dễ bị lỗi.
Đầu tư	Đầu tư là cần thiết cho các công cụ kiểm thử cũng như các kỹ sư kiểm thử tự động hóa	Đầu tư là cần thiết cho nguồn nhân lực.
Hiệu quả chi phí	Không hiệu quả nếu kiểm thử hồi quy với số lượng nhỏ	Kết quả của kiểm thử thủ công thường được ghi lại trong Excel hoặc Word
Báo cáo Kiểm thử	Với kiểm thử tự động, tất cả các bên liên quan có thể đăng nhập vào hệ thống tự động và xem được kết quả kiểm thử	Kết quả của kiểm thử thủ công thường được ghi lại trong Excel hoặc Word
Giao diện đối với người dùng	Kiểm thử tự động không liên quan đến hành động của con người. Vì vậy, không phù hợp với người dùng và trải nghiệm tích cực khách hàng.	Phương pháp kiểm thử thủ công cho phép người dùng quan sát, có thể hữu ích để cung cấp hệ thống thân thiện với người dùng.
Kiểm thử hàng loạt	Có thể chạy hàng loạt các Script.	Kiểm thử thủ công không thể thực hiện hàng loạt.
Kiến thức lập trình	Kiến thức lập trình là bắt buộc để thực hiện kiểm thử tự động.	Không cần kiến thức lập trình trong Kiểm thử thủ công.

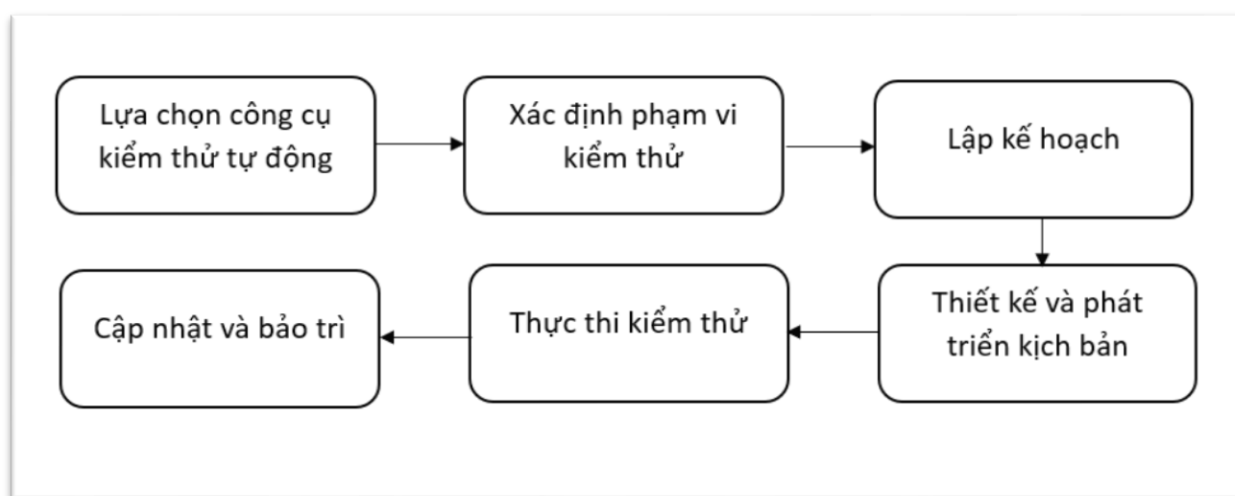
Thời hạn (Deadlines)	Các kiểm thử tự động không có rủi ro trễ Deadlines	Kiểm thử thủ công có nguy cơ trễ Deadlines cao.
Framework	Kiểm thử tự động hóa sử dụng các Framework như Data Drive, Keyword, Hybrid để tăng tốc quá trình tự động hóa.	Kiểm thử thủ công không sử dụng Framework nhưng có thể sử dụng guidelines, checklists, quy trình nghiêm ngặt để tạo ra một số test case nhất định.
Tài liệu	Kiểm thử tự động như một tài liệu để training về những test cases unit tự động. Một developer mới có thể tham khảo và nhanh chóng hiểu yêu cầu.	Các trường hợp kiểm thử thủ công không cung cấp giá trị training
Test Design	Kiểm thử đơn vị Tự động bắt buộc xây dựng thiết kế dựa trên kiểm thử	Kiểm thử đơn vị thủ công không bắt buộc thiết kế trong quá trình coding

1.7. Quy trình kiểm thử tự động

Việc kiểm thử tự động thì tốn kém hơn kiểm thử thủ công rất nhiều nếu chỉ thực hiện một lần. Để đạt được lợi ích, kiểm thử tự động cần phải được lựa chọn và thực thi một cách cẩn thận theo một tiến trình cụ thể.

Trước khi thực hiện kiểm thử tự động, kỹ sư kiểm thử và người quản lý phải có hiểu biết rõ ràng về kiểm thử tự động, bao gồm nhu cầu, mục tiêu, lợi ích, các vấn đề và thách thức.

Một tiến trình hiệu quả để thực hiện tự động hóa kiểm thử bao gồm các bước như ở hình 1.13



Hình 1.4 Quy trình kiểm thử tự động

Bước 1: Lựa chọn công cụ kiểm thử.

Việc lựa chọn công cụ kiểm thử tự động phù hợp thường phụ thuộc vào loại của ứng dụng đang được kiểm tra và môi trường mà ứng dụng đó sẽ chạy.

Bước 2: Xác định phạm vi kiểm thử tự động

Trong bước này, cần xem xét khoanh vùng các luồng nghiệp vụ kiểm thử phù hợp cho việc tự động hóa nó, chuẩn bị dữ liệu và môi trường diễn ra kiểm thử. Dưới đây là một số yếu tố cần xem xét khi xác định phạm vi thử nghiệm tự động

- Các tính năng chính, quan trọng của ứng dụng
- Các trường hợp kiểm thử có nhiều dữ liệu
- Các tính năng dùng chung (common) trên ứng dụng

- Những vùng khả thi về kỹ thuật mà công cụ đáp ứng được
- Những nghiệp vụ hay được tái sử dụng
- Mức độ phức tạp của các test cases
- Khả năng sử dụng các trường hợp kiểm thử giống nhau để test trên nhiều trình duyệt (với trường hợp kiểm thử ứng dụng web)

Bước 3: Lập kế hoạch.

Mục đích chính của bước này là lập kế hoạch xác định các đối tượng cần phải tự động, mục đích, chiến lược, yêu cầu, lịch trình, kinh phí. Trong thực tế, kế hoạch tự động hóa thường được tạo cho một sản phẩm, hoặc một dòng sản phẩm ngay ở giai đoạn đầu của tiến trình phát triển phần mềm.

Bước 4: Thiết kế và phát triển kịch bản kiểm thử tự động.

Ở giai đoạn này, người kiểm thử bắt tay vào thiết kế các kịch bản tự động và dùng công cụ được chọn để tạo ra các script tự động hóa. Các công việc điển hình như: Thiết kế Framework và các tính năng, thiết kế các kịch bản tự động, viết script và kiểm tra tính ổn định của script, xem xét tính đúng đắn của script so với thiết kế

Bước 5: Thực thi kiểm thử

Đây là giai đoạn áp chót trong quy trình kiểm thử tự động hóa. Khi đã tạo xong các script tự động, đã đến lúc chạy các script này để thực hiện kiểm thử ngay trên ứng dụng. Kết quả của việc thực thi các script thường sẽ được tổng hợp vào một báo cáo cho biết số lượng test cases PASS/FAIL và kèm theo các bằng chứng ghi lại hình ảnh/ trạng thái tại thời điểm xảy ra các lỗi được tìm thấy.

Bước 6: Cập nhật và bảo trì

Khi phần mềm được cập nhật thêm tính năng mới hoặc chỉnh sửa, thì bộ script sẽ được chạy để kiểm tra xem mức độ ảnh hưởng của các tính năng mới tới các tính năng hiện tại, liệu chúng có còn hoạt động đúng hay không. Khi phần mềm có cập nhật, những thay đổi trên phần mềm có thể làm cho bộ script không còn đúng, đây là lúc xem xét chỉnh sửa lại script cho phù hợp với những thay đổi.

1.7.1. Một số Test automation framework khác

Google EarlGrey

EarlGrey là một khung kiểm tra tự động hóa giao diện người dùng iOS gốc cho phép các nhà phát triển viết các bài kiểm tra ngắn gọn và rõ ràng. Các bài kiểm tra dễ dàng hơn để viết và duy trì. Nó có tính năng đồng bộ hóa tích hợp mạnh mẽ với giao diện người dùng, hình ảnh động, yêu cầu mạng, v.v.

Cucumber

Cucumber là một công cụ hướng hành vi chủ yếu được sử dụng để viết các bài kiểm tra chấp nhận cho các ứng dụng web. Nó cung cấp cho người dùng một thiết lập nhanh chóng và dễ dàng để bắt đầu và cũng cho phép họ sử dụng lại mã trong các thử nghiệm khác nhau. Khung công tác ban đầu được thực hiện trong Ruby và bây giờ được mở rộng sang khung công tác Java.

Appium

Appium chủ yếu được thiết kế để kiểm tra các ứng dụng di động. Nó được thiết kế theo cách mà chúng ta không phải biên dịch lại ứng dụng của mình hoặc sửa đổi nó theo bất kỳ cách nào để chạy thử nghiệm. Nó là một khuôn khổ đa nền tảng có thể được sử dụng để chạy trên các nền tảng khác nhau bằng cách sử dụng cùng một API.

Robot Framework

Đây là một trong những khuôn khổ tự động hóa kiểm tra chung nhất được sử dụng để phát triển theo hướng kiểm tra chấp nhận và kiểm tra chấp nhận. Robot Framework có thể được sử dụng trong các môi trường phân tán, không đồng nhất, nơi yêu cầu sử dụng các công nghệ và giao diện khác nhau. Đây là một khuôn mẫu đa nền tảng cung cấp cú pháp dữ liệu kiểm tra dạng bảng dễ sử dụng.

Khung cho phép tích hợp dễ dàng, ghi nhật ký chi tiết và báo cáo thử nghiệm rõ ràng. Điểm mạnh của Robot Framework chính là được viết trên nền tảng Python và được hỗ trợ bởi số lượng thư viện dành cho tester, Robot Framework rất dễ sử dụng cũng như viết test script và có thể chạy được trên mọi nền tảng khác nhau mà không cần chỉnh sửa test script.

Gauge

Gauge là một trong những công cụ tự động hóa thử nghiệm nhẹ tiên tiến cung cấp các tính năng đa nền tảng. Nó giới thiệu một cú pháp đơn giản, phong phú và linh hoạt và thực hiện thực thi theo hướng dữ liệu. Các trường hợp thử nghiệm trong khuôn khổ này có thể dễ dàng hiểu và duy trì, đồng thời nó có kiến trúc mô-đun cung cấp các plugin có khả năng mở rộng cao.

1.7.2. Tại sao nên lựa chọn Automation testing

- + Độ tin cậy cao: Công cụ kiểm thử tự động có sự ổn định cao vì hoạt động theo quy trình định sẵn, đặc biệt trong trường hợp nhiều test case, các bài kiểm tra tiêu chuẩn lặp đi lặp lại nhằm tránh không thể bỏ.

- + Khả năng lặp: Mình có thể test cách phần mềm xử lý (tính năng/hiệu năng) khi gặp tình huống chạy lặp đi lặp lại nhiều lần trên cùng script test giúp các Tester xử lý trường hợp lặp đi lặp lại các thao tác như: click, nhập dữ liệu, check kết quả,...) Đây còn gọi là performance/load testing.

- + Khả năng tái sử dụng: Các script có thể sử dụng lại và không cần script mới mọi lúc. Ngoài ra, các script có thể thực hiện lại các bước chính xác như những gì đã diễn ra trước đó.

- + Tiết kiệm thời gian: Automation test giúp chạy test nhanh hơn với tốc độ nhanh hơn ít nhất 10 lần so với tốc độ kiểm thử thủ công. Nếu cần 5 phút để thực thi một test case cách thủ công thì chỉ cần khoảng 30s để thực thi tự động.

- + Chi phí thấp: nếu áp dụng kiểm thử tự động đúng cách, chúng ta có thể tiết kiệm được nhiều chi phí về thời gian và nhân lực.

CHƯƠNG 2: CÔNG CỤ SELENIUM

2.1. Giới thiệu công cụ Selenium

2.1.1 Lịch sử phát triển

Selenium được phát triển bởi Jason Huggins làm việc tại ThoughtWorks có trụ sở tại Chicago có tên ban đầu là JavaScript Test Runner. Tự động kiểm tra bất kỳ ứng dụng nào là cốt lõi đối với phong cách của ThoughtWork, dựa trên khuynh hướng Agile của công ty này. Đây là tiền đề của Selenium IDE và Selenium RC.

Selenium WebDriver được phát triển Simon Stewart năm 2007, Nó không dựa vào JavaScript để thực hiện công việc nặng nhọc mà thay vào đó có một ứng dụng khách cho mỗi trình duyệt được mã hóa từ đầu. Nó cũng có API “cấp cao hơn” so với Selenium-RC và cho thấy nhiều hứa hẹn. Simon đã trình bày công cụ này tại GTAC và bắt đầu nghiên cứu khả năng tương thích với Selenium-RC, điều này dẫn đến kết luận rõ ràng rằng hai dự án nên hợp nhất.

2.1.2 Khái niệm

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 như: Firefox, Google Chrome, Microsoft edge. Selenium không chỉ là 1 công cụ độc lập mà là 1 bộ công cụ của phần mềm, mỗi bộ đều đáp ứng được nhu cầu kiểm thử khác nhau của 1 tổ chức. Selenium hoạt động bằng cách mô phỏng các thao tác của người dùng trên web page hoặc web element. Khi viết test bằng Selenium, tester chỉ định các hành động mà họ muốn người dùng thực hiện, Selenium sẽ tự động thực thi các hành động đó trên trình duyệt.

2.1.3 Những tính năng của Selenium

- Selenium IDE hỗ trợ tính năng playback giúp chúng ta có thể sử dụng các bài test của người khác và không cần phải biết ngôn ngữ script
- Selenium là một nền tảng kiểm thử dựa trên cloud giúp tester có thể lưu lại thao tác và xuất ra dưới dạng script đơn giản, dễ hiểu.
- Selenium hỗ trợ nhiều hệ điều hành, ngôn ngữ và trình duyệt khác nhau.
- Giúp chúng ta có thể chạy cùng lúc nhiều bài test để giảm thời gian và

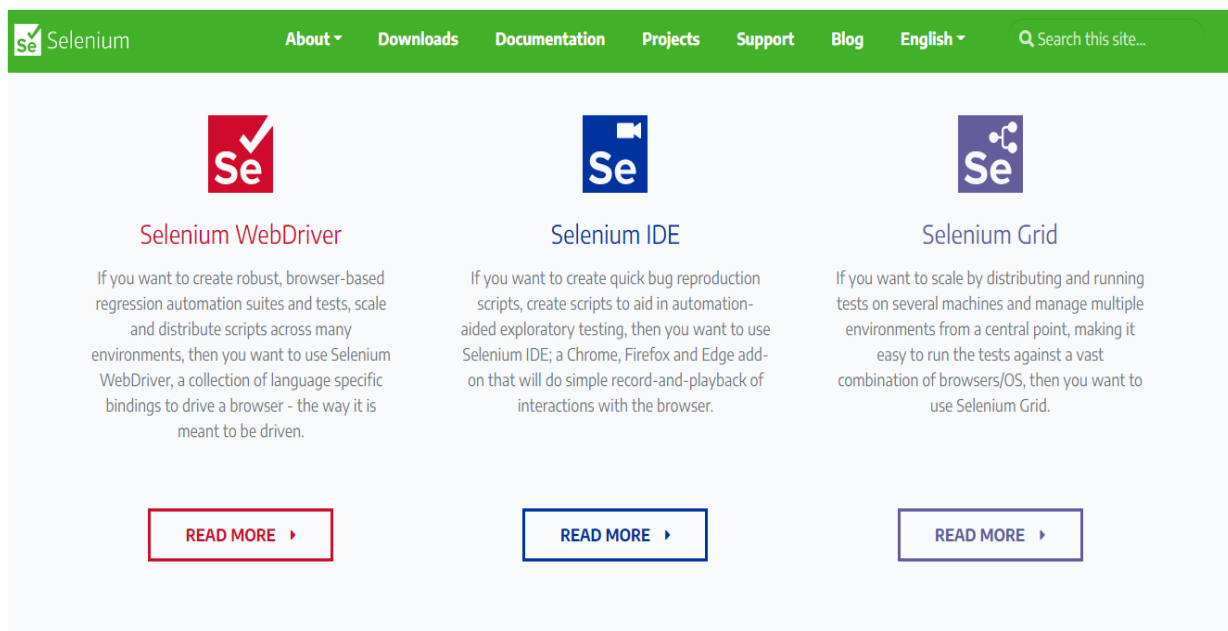
tăng hiệu quả

- Quá trình kiểm thử của Selenium hao tốn ít tài nguyên và yêu cầu cấu hình thiết bị thấp hơn các công cụ khác.
- Selenium WebDriver không yêu cầu cài đặt server, test script của chúng ta sẽ trực tiếp tương tác với trình duyệt.

2.1.4 Selenium bao gồm những công cụ nào?

Tính đến thời điểm hiện tại, Selenium không phải là một công cụ duy nhất. Selenium có đến 4 công cụ để chúng ta có thể tìm ra mục đích sử dụng và lựa chọn thích hợp.

- Selenium Integrated Development Environment (IDE)
- Selenium Remote Control
- Selenium WebDriver
- Selenium Grid



Hình 2. 1 Những công cụ trong Selenium

Selenium IDE

Sử dụng để học và tìm hiểu về các khái niệm kiểm tra tự động và Selenium như:

- Học và chọn các lệnh như: type, open, clickAndWait, assert, verify,...
- Học cách sử dụng bộ định vị như: id, name, xpath, css selector
- Tùy chỉnh JavaScript bằng cách sử dụng runScript

Selenium Remote Control

- Tạo test với ngôn ngữ dễ hiểu hơn Selenese
- Sử dụng để chạy thử nghiệm test trên nhiều trình duyệt khác nhau trên nhiều hệ điều hành khác nhau.
- Triển khai thêm nhiều môi trường thử nghiệm khác bằng cách kết hợp sử dụng với Selenium Grid.

Selenium WebDriver

- Dùng một ngôn ngữ nhất định trong quá trình kiểm thử
- Kiểm thử nhiều ứng dụng dựa trên nền Ajax
- Tạo bài test trên trình duyệt HtmlUnit
- Tạo kết quả kiểm tra tùy chỉnh

Selenium Grid

- Dùng để chạy các script của Selenium Remote Control trên nhiều hệ điều hành, nhiều trình duyệt khác nhau cùng lúc
- Giúp chúng ta có thể tạo ra một bộ test với số lượng lớn bài test nhỏ cùng lúc để tăng tốc hoàn thành việc kiểm tra.

2.1.5 Ưu điểm và nhược điểm của Selenium

Ưu điểm:

- Quá trình cài đặt và sử dụng rất đơn giản
- Yêu cầu cấu hình phần cứng thấp
- Hỗ trợ đa dạng hệ điều hành, trình duyệt và hỗ trợ nhiều ngôn ngữ lập trình
- Có bộ API hoàn thiện
- Tạo ra bộ test lớn với nhiều bài test nhỏ để tiết kiệm thời gian
- Hỗ trợ kiểm thử tự động thay cho người dùng thật giúp tiết kiệm nhiều nhân lực

Nhược điểm:

- Selenium IDE không hỗ trợ thực hiện tính toán hoặc câu lệnh phức tạp được
- Quá trình cài đặt Selenium Webdriver tốn thời gian và cần có nhiều kinh nghiệm
- Nếu chúng ta chạy quá nhiều test so với khả năng đáp ứng của phần cứng, chắc chắn quá trình test của chúng ta sẽ không có kết quả tốt và đôi khi sẽ đứng máy
- Không có bộ phận hỗ trợ kỹ thuật chuyên dụng. Vì thế, khi triển khai các dự án thương mại và bị lỗi, chúng ta sẽ cần phải tìm đơn vị thứ 3 hỗ trợ
- Chỉ hỗ trợ ứng dụng web.

2.2 Giới thiệu framework Mocha

Mocha là một khung kiểm tra JavaScript giàu tính năng chạy trên Node.js và trong trình duyệt, giúp việc kiểm tra không đồng bộ trở nên đơn giản và thú vị. Các thử nghiệm Mocha chạy tuần tự, cho phép báo cáo linh hoạt và chính xác, đồng thời ánh xạ các ngoại lệ chưa được phát hiện vào các trường hợp kiểm thử chính xác. Cấu trúc cơ bản của một tệp kiểm thử:

```
describe("hooks", function() => {
  //Những điều kiện không bắt buộc
  before(function() { // Chạy trước tất cả các bài kiểm thử});
  after(function() { // Chạy sau tất cả các bài kiểm thử});
  beforeEach(function() { //Chạy trước mỗi bài kiểm thử });
  afterEach(function() { //Chạy sau mỗi bài kiểm thử});
  // Test cases
  it("name", () => { // Các câu lệnh thực thi test case;});
});
```

Trong trình duyệt, các tệp kiểm tra được tải theo từng thẻ `<script>` và quá trình gọi `mocha.run()` bắt đầu ở bước 9 theo chế độ nối tiếp dưới đây:

1. Người dùng thực thi mocha
2. Tải các tùy chọn từ tập tin cấu hình (nếu có)
3. Mocha xử lý mọi tùy chọn dòng lệnh được cung cấp
4. Nếu tìm thấy các flags để thực thi tệp node, mocha sẽ sinh ra node trong một tiến trình con, tự thực thi với các flags và không sinh ra các

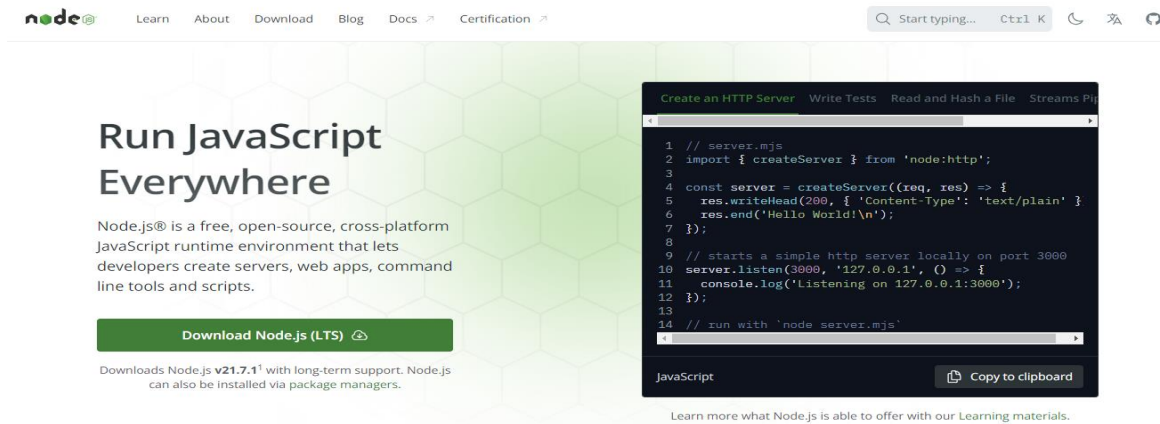
tiến trình con.

5. Mocha tải các mô-đun được chỉ định bởi *--require*.
6. Mocha xác thực mọi trình báo cáo hoặc giao diện được tải qua *--require* hoặc bằng cách khác.
7. Mocha phát hiện ra các tập tin kiểm thử, khi không có tệp hoặc thư mục nào, nó sẽ tìm các tệp có phần mở rộng *.js*, *.mjs* hoặc *.cjs* trong mục test (không tệp con của nó), liên quan đến thư mục làm việc hiện tại.
8. Giao diện bdd sẽ tải các tệp không theo thứ tự cụ thể. Khi các tệp được tải, Mocha sẽ tìm nhưng không thực thi phần nào trong bộ test suite đó.
9. Mocha chạy các thiết lập được cài đặt (nếu có)
10. Mocha bắt đầu với bộ “root” để thực thi
11. Mocha thực thi một lần “before all” cho cả bộ trong mỗi lần chạy
12. Với mỗi trường hợp kiểm thử, Mocha sẽ thực thi “before each”, sau đó thực thi các câu lệnh bên trong (bao gồm cả báo cáo) và cuối cùng là thực thi “after each”.
13. Nếu trong bộ test có bộ con, sẽ lặp lại các bước 10,11,12 cho mỗi bộ con. Mỗi bộ con đều kế thừa “before each” và “after each” được xác định trong phần cha của nó.
14. Mocha thực thi “after all”
15. Mocha in phần kết quả

2.3. Cài đặt Selenium Webdriver và môi trường kiểm thử

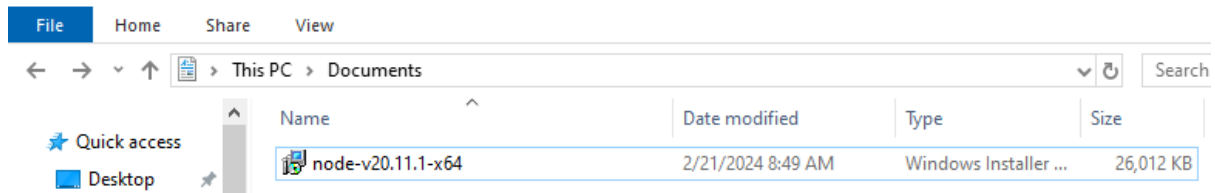
2.3.1. Cài đặt Node.js và npm

Bước 1: Truy cập “<https://nodejs.org/en/download>” và tải xuống Node.js phiên bản mới nhất phù hợp với thiết bị.

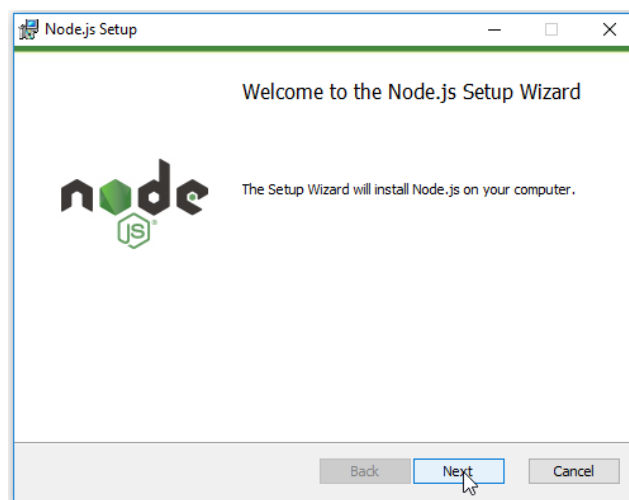


Hình 2.2 Cài đặt Nodejs(1)

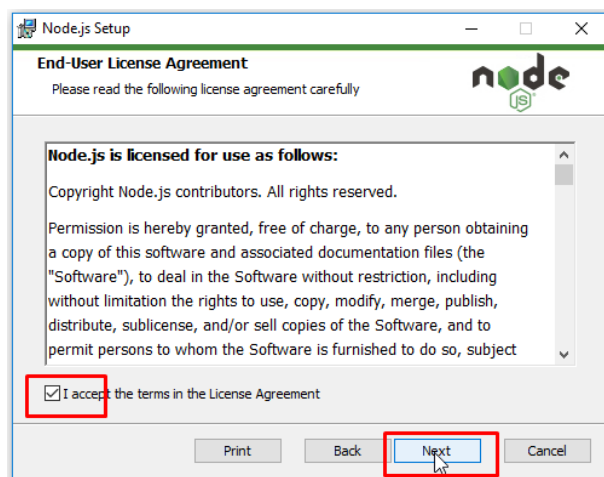
Bước 2: Click đúp vào file .msi vừa tải về để tiến hành cài đặt.



Hình 2.3: Cài đặt Nodejs(2)

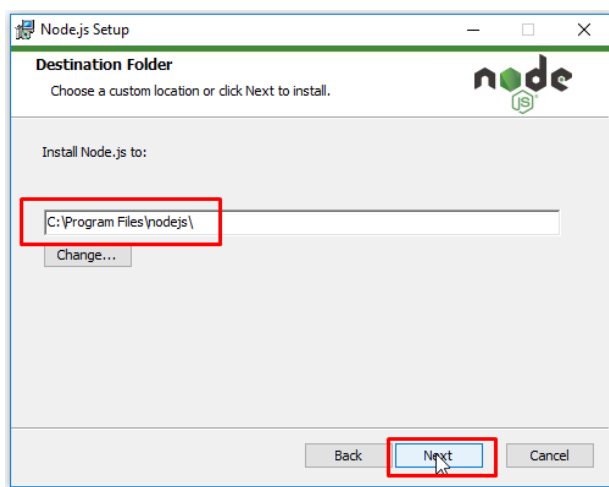


Hình 2.4 Cài đặt Nodejs(3)

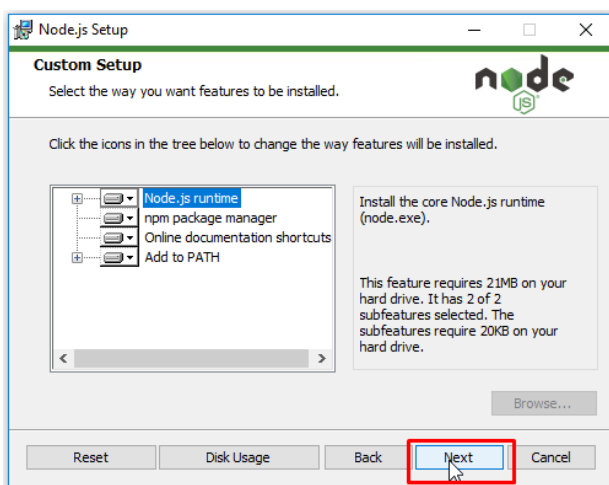


Hình 2.5 Cài đặt Nodejs(4)

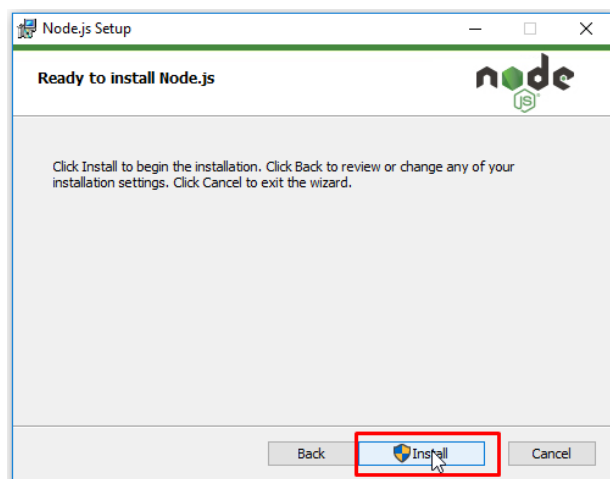
Bước 3: Chọn thư mục để cài đặt nodejs



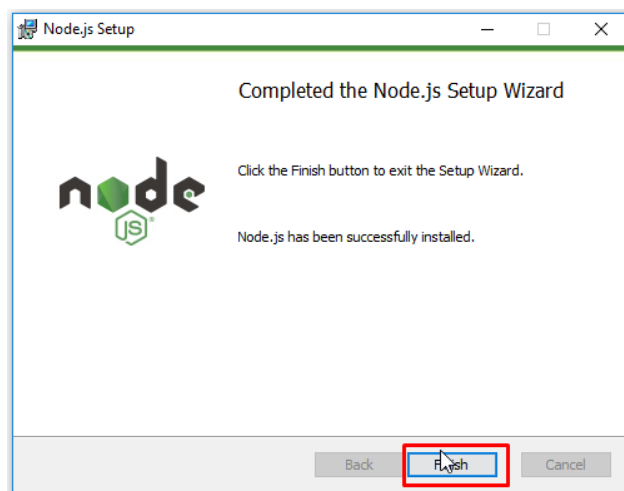
Hình 2.6 Cài đặt Nodejs(5)



Hình 2.7 Cài đặt Nodejs(6)



Hình 2.8 Cài đặt Nodejs(7)



Hình 2. 9 Cài đặt Nodejs(8)

Bước 4: Kiểm tra version của nodejs và npm

Để kiểm tra version của nodejs và npm, ta mở cửa sổ terminal của Visual Studio Code và nhập lệnh.

- Lệnh “`node -v`” để kiểm tra version nodejs

```

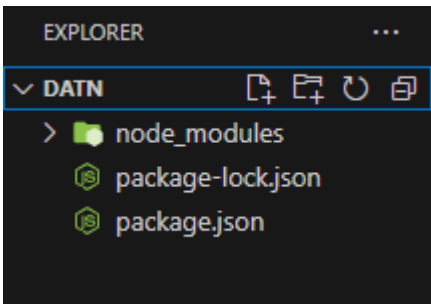
PROBLEMS  OUTPUT  TERMINAL  ...
PS D:\STUDY\DATN> node -v
v20.11.1
PS D:\STUDY\DATN> npm -v
10.2.4
PS D:\STUDY\DATN>

```

Hình 2.10 Kiểm tra version của nodejs và npm

- Lệnh “`npm -v`” để kiểm tra version npm

Sau khi cài đặt xong sẽ xuất hiện folder “`node_module`” chứa thông tin của dự án



Hình 2.11 Folder `node_module` sau khi cài đặt

2.3.2. Cài đặt Selenium webdriver

Điền lệnh “`npm install selenium-webdriver`” vào terminal trong Visual Studio Code

```
PS D:\STUDY\DATN> npm install selenium-webdriver
added 16 packages in 3s
PS D:\STUDY\DATN> 
```

Hình 2.12 Cài đặt Selenium webdriver

2.3.3. Cài đặt framework Mocha

Điền lệnh “`npm install --save-dev mocha`” vào terminal trong Visual Studio Code

```
PS D:\STUDY\123\NMHieu> npm install --save-dev mocha
up to date, audited 664 packages in 2s
144 packages are looking for funding
  run `npm fund` for details
1 moderate severity vulnerability
To address all issues, run:
  npm audit fix
```

Hình 2.13 Cài đặt framework Mocha

2.4. Cấu trúc của một test script automation trong nodejs

- *Import các thư viện cần thiết*

Import các module hoặc thư viện như framework test, công cụ assertion (chai, expect), và thư viện điều khiển browser automation (như WebDriver hoặc Puppeteer).

```
const { expect } = require('chai');  
const { Builder, By, Key, until } = require('selenium-webdriver');
```

Hình 2.14 Import các thư viện cần thiết

- *Thiết lập môi trường*

Khai báo và khởi tạo các thành phần cần thiết (driver, URL, hoặc dữ liệu giả lập).

```
const url = "https://example.com";  
let driver;  
  
before(async function () {  
    driver = await new Builder().forBrowser('chrome').build();  
});  
  
after(async function () {  
    await driver.quit();  
});
```

Hình 2.15 Thiết lập môi trường

- *Định nghĩa các test cases*

Sử dụng các khối describe và it để tổ chức test case

```
describe('Test Automation Example', function () {
  this.timeout(30000); // Đặt timeout cho các test case

  it('should load the webpage and verify the title', async function () {
    await driver.get(url);
    const title = await driver.getTitle();
    expect(title).toEqual('Example Domain');
  });

  it('should find and click a button', async function () {
    const button = await driver.findElement(By.css('button#submit'));
    await button.click();
    const confirmationText = await driver.findElement(By.css('.confirmation')).getText();
    expect(confirmationText).toEqual('Form submitted!');
  });
});
```

Hình 2.16 Định nghĩa các test cases

- *Xử lý data và setup/teardown*

Setup: Chuẩn bị trước khi chạy các test case.

Teardown: Dọn dẹp sau khi test hoàn tất.

```
beforeEach(async function () {
  console.log("Preparing for test case...");
});

afterEach(async function () {
  console.log("Cleaning up after test case...");
});
```

Hình 2.17 Xử lý data và setup/teardown

- *Chạy các script*

Tích hợp script này vào hệ thống chạy test (như npm test) bằng cách cấu hình file package.json.

```
"scripts": {  
  "test": "mocha tests/**/*.test.js"  
}
```

Hình 2.18 Chạy các script

CHƯƠNG 3. THỰC HIỆN KIỂM THỬ WEBSITE

3.1. Giới thiệu chương trình

3.1.1 Giới thiệu về website

Website Divimenti được phát triển để người dùng có thể chia sẻ thông tin và kết nối với nhiều người dùng khác ở mọi lúc, mọi nơi, mọi lứa tuổi mà chỉ cần có kết nối Internet. Với giao diện đơn giản và dễ sử dụng, Divimenti cho phép người dùng chia sẻ văn bản, hình ảnh và nhiều nội dung đa dạng khác chỉ trong vài cú nhấp chuột. Ngoài ra, Divimenti còn cung cấp tính năng nhắn tin để người dùng có thể trò chuyện cùng với bạn bè một cách nhanh chóng, thuận tiện và miễn phí.

3.1.2 Chức năng của hệ thống

Mỗi người dùng, tùy thuộc vào là người quản trị hay người dùng sẽ có duy nhất một tên đăng nhập và mật khẩu để sử dụng cho việc đăng nhập tài khoản trên hệ thống.

Về phía người dùng:

Bảng 3.1 Chức năng hệ thống về phía người dùng

Chức năng	Mô tả
Đăng ký	<ul style="list-style-type: none"> - Tài khoản: bắt buộc nhập, nhập tối đa 100 kí tự. Hiện thị mặc định: “Tài khoản” Nếu bỏ trống cảnh báo: Vui lòng điền vào trường này! Không nhập được quá 100 ký tự - Email: bắt buộc nhập, nhập tối đa 254 kí tự. Hiện thị mặc định: “Email” Có @. Nếu không có @, báo lỗi "Enter a valid email address." Nếu bỏ trống cảnh báo: Vui lòng điền vào trường này! Không nhập được quá 254 ký tự - Số điện thoại: bắt buộc nhập, là số, nhập tối đa 15 kí tự. Hiện thị mặc định: “Số điện thoại” Nếu bỏ trống cảnh báo: Vui lòng điền vào trường này!

	<p>- Mật khẩu: bắt buộc nhập, có ít nhất 8 ký tự bao gồm chữ hoa, chữ thường, số và ký tự đặc biệt. Nếu không đủ báo lỗi: "Passwords must have at least 8 characters, including uppercase letters, lowercase letters, numbers and special characters to ensure security!"</p> <p>Nếu bỏ trống cảnh báo: Vui lòng điền vào trường này!</p> <p>Các ký tự chuyển thành chấm tròn sau khi nhập.</p> <p>Hiện thị mặc định: “Mật khẩu”</p> <p>- Button "Hiện": ban đầu button hiển thị "Hiện"</p> <p>Click lần 1 button chuyển "Ẩn", các ký tự đã nhập hiển thị.</p> <p>Click lần 2 button chuyển thành "Hiện", các ký tự chuyển thành chấm tròn.</p> <p>Nếu textbox không có ký tự nào thì hiển thị mặc định "Mật khẩu"</p> <p>-Tên: bắt buộc nhập, nhập tối đa 100 kí tự.</p> <p>Hiện thị mặc định: “Tên”</p> <p>Nếu bỏ trống cảnh báo: Vui lòng điền vào trường này!</p> <p>Không nhập được quá 100 ký tự</p> <p>- Họ: bắt buộc nhập, nhập tối đa 100 kí tự.</p> <p>Hiện thị mặc định: “Họ”</p> <p>Nếu bỏ trống cảnh báo: Vui lòng điền vào trường này!</p> <p>Không nhập được quá 100 ký tự</p> <p>- Ngày sinh: điền hoặc chọn . Nếu nhập ngày sinh lỗi thì thông báo "Enter a valid date."</p> <p>- Giới tính: mặc định chọn “Male”, người dùng có thể chọn 1 trong 3 lựa chọn Male, Female, Other.</p> <p>- Button "Đăng ký":</p> <p>Nếu bỏ trống tất cả các trường và click “Đăng ký” thì báo lỗi "Vui lòng điền vào trường này" trường đầu tiên chưa nhập từ trên xuống.</p>
--	---

	<p>Nếu người dùng đăng ký tài khoản với có tên trùng với tài khoản khác trong cơ sở dữ liệu thì hiển thị thông báo "User with this Username already exists."</p> <p>Nếu người dùng đăng ký tài khoản với có Email trùng với tài khoản khác trong cơ sở dữ liệu thì hiển thị thông báo "User with this Email already exists."</p> <p>Nếu người dùng đăng ký tài khoản với có tên trùng với tài khoản khác trong cơ sở dữ liệu thì hiển thị thông báo "User with this Phone already exists."</p> <p>- Link "Đăng nhập": click vào link thì chuyển sang trang đăng nhập</p>
Đăng nhập	<p>- Tài khoản: bắt buộc nhập, nhập tối đa 100 kí tự. Hiển thị mặc định: "Tài khoản" Nếu bỏ trống báo lỗi: "Vui lòng điền vào trường này" Không nhập được quá 100 ký tự.</p> <p>- Mật khẩu: bắt buộc nhập tối thiểu 8 kí tự có phân biệt chữ hoa chữ thường. Hiển thị mặc định: "Mật khẩu" Sau khi nhập, kí tự chuyển thành chấm tròn. Nếu bỏ trống báo lỗi: "Vui lòng điền vào trường này"</p> <p>- Button "Login": báo lỗi tại trường đầu tiên từ trên xuống khi người dùng chưa điền đầy đủ thông tin. Hiển thị thông báo "Please enter a correct username and password. Note that both fields may be case-sensitive!" khi tài khoản người dùng không tồn hoặc mật khẩu nhập không đúng</p> <p>- Link "Đăng ký": khi người dùng click vào link sẽ chuyển sang trang đăng ký.</p>
Xem trang cá nhân	<p>Người dùng click vào ảnh đại diện ở góc trên bên phải màn hình tại màn hình trang chủ, hiển thị màn hình trang cá nhân của người dùng bao gồm ảnh đại diện, ảnh bìa,</p>

	Tên tài khoản, btn "Interacs", button “Rechange”, hiển thị các bài viết đã đăng(nếu có)
Xem trang chủ	Người dùng click vào logo "Divimenti" hoặc btn "New feed" ở góc trên bên trái màn hình, hiển thị màn hình trang chủ
Đăng bài	<p>Người dùng có thể nhập liệu và chọn ảnh và click btn "Share" để đăng bài</p> <p>-Caption: nhập nội dung bài viết. Hiển thị mặc định: “What’s on your mind?”</p> <p>- Chọn ảnh: chọn 1 ảnh từ thiết bị</p> <p>- Button “Đăng”: đăng bài. Nếu không có dữ liệu nào thì không đăng được bài viết mới.</p> <p>Người dùng cũng có thể xóa bài viết đã đăng bằng cách click vào icon ba chấm ở góc trên bên phải bài viết và chọn “Delete”</p>
Bình luận	<p>Người dùng có thể bình luận/ xem bình luận trên những bài đăng của mình hoặc bạn bè</p> <p>- Comment: nhập nội dung bình luận.</p> <p>- Button “Post”: bình luận. Nếu không có ký tự nào thì không bình luận được</p>
Tìm kiếm bạn bè	<p>Hiển thị mặc định : Start typing to search...</p> <p>-Textbox: nhập tối đa 100 ký tự</p> <p>Thanh search hiển thị trên màn hình trang chủ, người dùng nhập liệu vào thanh search và nhấn Enter trên bàn phím để tìm kiếm.</p> <p>Sau khi tìm kiếm, nếu có tài khoản người dùng khớp với nội dung tìm kiếm thì hệ thống sẽ hiển thị lên màn hình "Search result for: + nội dung trên ô search" và danh sách người dùng.</p> <p>Nếu không nhập ký tự nào và Enter thì hiển thị tất cả người dùng</p>

Nhắn tin	<p>Tại màn hình trang cá nhân của người dùng khác, người dùng hover vào button "Interacs", màn hình hiển thị danh sách các actions để chọn bao gồm: Follow, Add friend, Become a fan, Send message. Người dùng chọn "Send Message" để nhắn tin.</p> <p>Trường hợp người dùng và người dùng khác đã có tin nhắn, click vào icon tin nhắn ở góc trên bên phải hoặc button "Chat" ở sidebar bên trái tại bất kì màn hình nào, sau đó chọn cuộc hội thoại</p> <p>Hiển thị mặc định: "Type your message"</p> <ul style="list-style-type: none"> - InputMessage: nhập tin nhắn - Button "Send": gửi tin nhắn. <p>Nếu không có ký tự nào thì không gửi được</p>
Kết bạn	<p>Tại màn hình trang cá nhân của người dùng khác, người dùng hover vào button "Interacs", màn hình hiển thị danh sách các action để chọn. Người dùng chọn "Add friend", hệ thống sẽ gửi thông báo đến người dùng đó. Nếu người dùng đó chấp nhận kết bạn thì 2 người dùng trở thành bạn bè.</p>
Theo dõi	<p>Tại màn hình trang cá nhân của người dùng khác, người dùng hover vào button "Interacs", màn hình hiển thị danh sách các action để chọn. Người dùng chọn "Follow", action Follow chuyển thành Unfollow, hệ thống lưu mối quan hệ của 2 người dùng</p>
Trở thành fan	<p>Tại màn hình trang cá nhân của người dùng khác, người dùng hover vào button "Interacs", màn hình hiển thị danh sách các action để chọn. Người dùng chọn "Become a Fan", action chuyển thành Unfollow, hệ thống lưu mối quan hệ của 2 người dùng</p>

Về phía người quản trị:

Bảng 3.2 Chức năng hệ thống về phía người quản trị

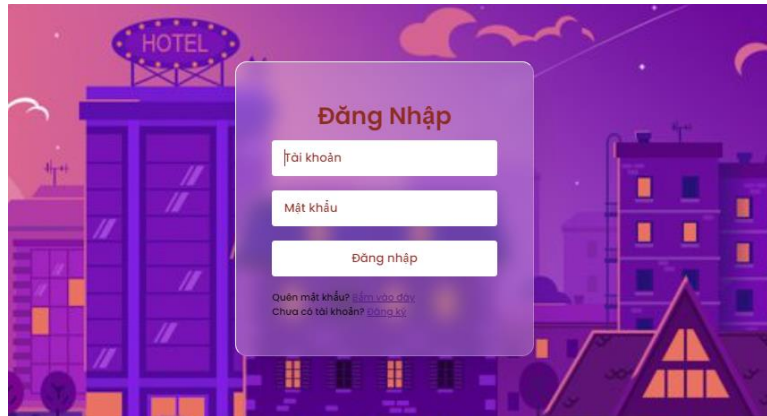
Chức năng	Mô tả
Quản lý tài khoản người dùng	Người quản trị có thể thêm, sửa, xóa tài khoản người dùng
Quản lý bài viết	Người quản trị có thể thêm, sửa, xóa bài viết
Quản lý tin nhắn	Người quản trị có thể thêm, sửa, xóa tin nhắn
Quản lý bình luận	Người quản trị có thể thêm, sửa, xóa bình luận
Quản lý túi	Người quản trị có thể thêm, sửa, xóa túi
Quản lý thẻ nạp	Người quản trị có thể thêm, sửa, xóa thẻ nạp
Tất cả các chức năng của người dùng.	

3.1.3 Một số màn hình giao diện của website

- Giao diện màn hình Đăng ký:

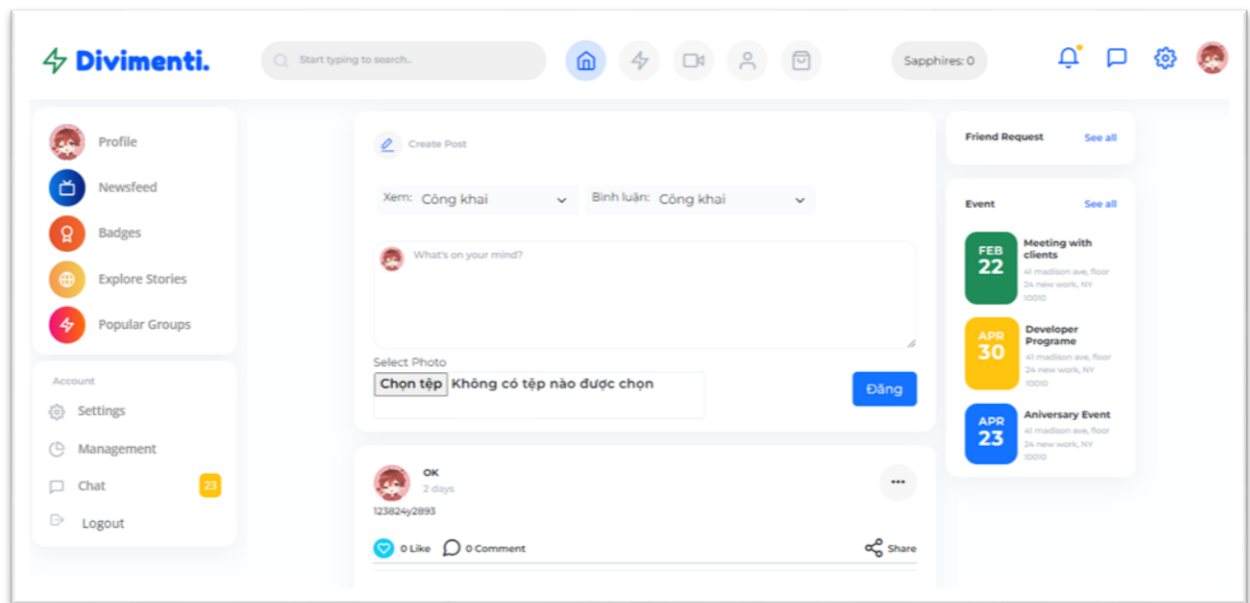
Hình 3.1 Giao diện màn hình Đăng ký

- Giao diện màn hình Đăng nhập



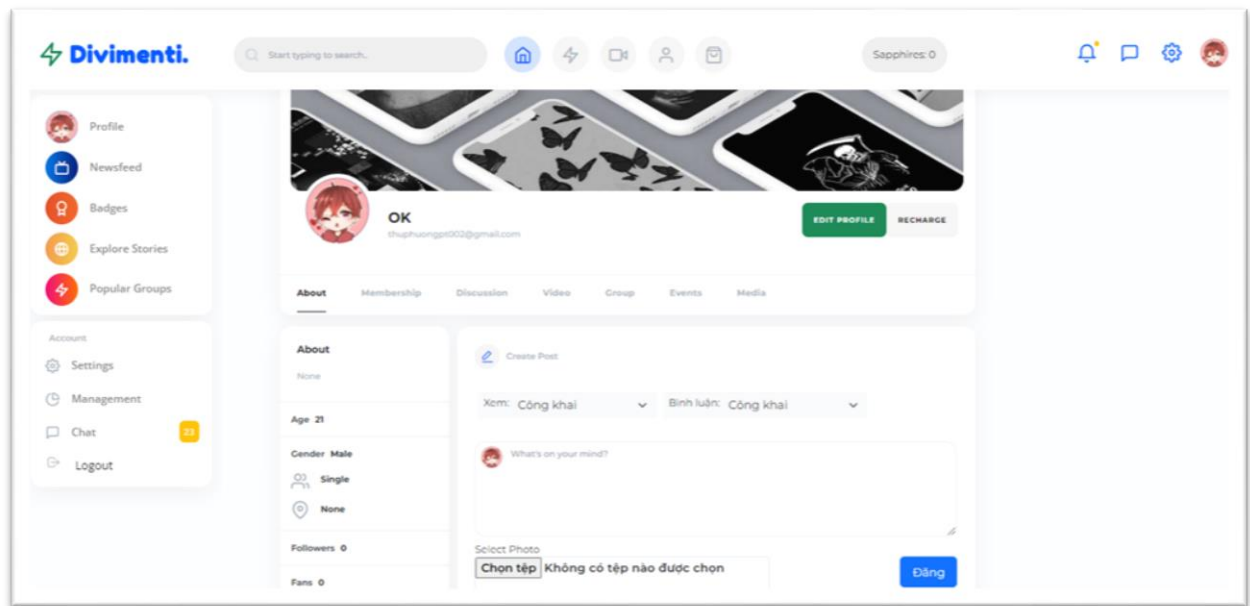
Hình 3.2 Giao diện màn hình Đăng nhập

- Giao diện màn hình trang chủ



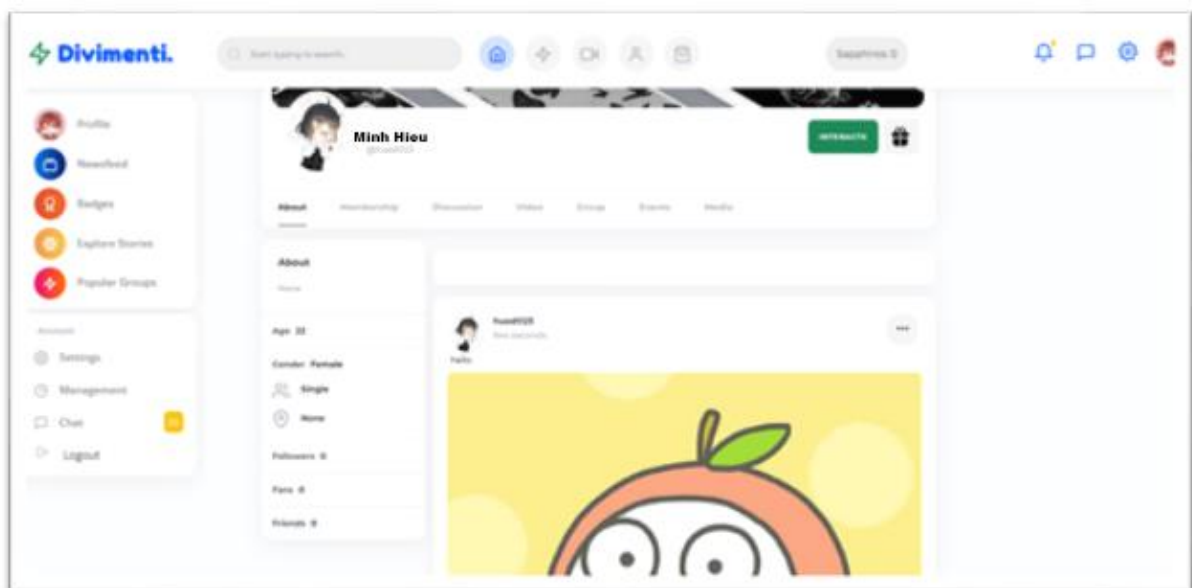
Hình 3.3 Giao diện màn hình Trang chủ

- Giao diện màn hình Trang cá nhân



Hình 3.4 Giao diện màn hình Trang cá nhân

- Giao diện màn hình trang cá nhân bạn bè



Hình 3.5 Giao diện màn hình Trang cá nhân bạn bè

3.2. Kế hoạch kiểm thử

Mục tiêu của kế hoạch kiểm thử là đảm bảo rằng website Divimenti hoạt động đúng với yêu cầu, thân thiện với người dùng và bảo mật trước khi phát hành. Phạm vi kiểm thử bao gồm các chức năng chính: Đăng ký, Đăng nhập, Đăng bài, Bình luận, Tìm kiếm bạn bè, Nhắn tin. Loại kiểm thử sẽ được thực hiện đó là kiểm thử chức năng. Công cụ được sử dụng là Selenium Webdriver. Dữ liệu kiểm thử sẽ được xây dựng dựa trên tình huống thực tế của người dùng. Kế hoạch được thực hiện trong 9 tuần, với các mốc nhiệm vụ bao gồm: Tìm hiểu và mô tả các chức năng (Đăng ký, Đăng nhập, Đăng bài, Bình luận, Tìm kiếm bạn bè, Nhắn tin), Review chức năng, Viết test case cho các chức năng, Thiết kế và phát triển các kịch bản tự động và Thực hiện kiểm thử.

Sau khi đã xác định mục tiêu và phạm vi cho kế hoạch kiểm thử ta tiến hành sử dụng phần mềm Microsoft Excel để hoàn thiện bảng kế hoạch kiểm thử hoàn chỉnh, sau khi hoàn thiện kế hoạch kiểm thử kết quả thu được như hình bên dưới.

TEST PLAN					
Task	Start date	Due date	Done criteria	Status	Note
Tìm hiểu và mô tả các chức năng - Đăng ký - Đăng nhập - Đăng bài - Bình luận - Tìm kiếm bạn bè - Nhắn tin	22/10/2024	05/11/2024	Hoàn thành mô tả chức năng và ràng buộc của từng control	DONE	
Review chức năng	06/11/2024	12/11/2024	Làm rõ được các công dụng của tất cả các control và những ràng buộc trong chức năng đó	DONE	
Viết test case cho các chức năng	13/11/2024	25/11/2024	Viết được ít nhất 100 TCs và có độ bao phủ trên 85%	DONE	
Thiết kế và phát triển các kịch bản tự động	26/11/2024	15/12/2024	Các script phù hợp với website, dễ nhìn, dễ quản lý	DONE	
Thực hiện kiểm thử	16/12/2024	22/12/2024	Có báo cáo cho biết số lượng test cases PASS/FAIL	DONE	

Hình 3.6 Kế hoạch kiểm thử website Divimenti

Trong bảng kế hoạch kiểm thử (Test plan) hoàn chỉnh của trang web Divimenti sẽ bao gồm các đề mục giống như “Hình 3.6” trong đó:

- **Task:** Mô tả nhiệm vụ cụ thể cần thực hiện.
- **Start Date:** Ngày bắt đầu của từng nhiệm vụ.

- **Due Date:** Ngày hoàn thành dự kiến.
- **Done Criteria:** Tiêu chí hoàn thành công việc.
- **Status:** Trạng thái của nhiệm vụ ("DONE", "IN PROGRESS").
- **Note:** Ghi chú.

Đối với các đề mục sẽ là các nhiệm vụ (Task), thời gian bắt đầu (Start Date), thời gian kết thúc (Due Date), tiêu chí hoàn thành công việc (Done Criteria), trạng thái của nhiệm vụ (Status) và ghi chú (Note) tương ứng, cụ thể:

- Nhiệm vụ Tìm hiểu và mô tả các chức năng có thời gian bắt đầu từ 22/10/2024 kết thúc vào 05/11/2024, tiêu chí hoàn thành của nhiệm vụ này là hoàn thành mô tả các chức năng và ràng buộc từng control, trạng thái của nhiệm vụ này là DONE và không có ghi chú nào.
- Nhiệm vụ Review chức năng, thời gian bắt đầu của nhiệm vụ này là từ ngày 06/11/2024 và kết thúc vào ngày 12/11/2024, tiêu chí hoàn thành của nhiệm vụ này là làm rõ được các công dụng của tất cả các control và những ràng buộc trong chức năng đó, trạng thái của nhiệm vụ này DONE và không có ghi chú nào.
- Nhiệm vụ Viết test case cho các chức năng, thời gian bắt đầu từ ngày 13/11/2024 kết thúc vào ngày 25/11/2024, tiêu chí hoàn thành cho nhiệm vụ này là viết được ít nhất 100 TCs và có độ bao phủ trên 85%, trạng thái của chức năng này là DONE và không có ghi chú nào.
- Nhiệm vụ Thiết kế và phát triển các kịch bản tự động, thời gian bắt đầu từ ngày 26/11/2024 và kết thúc vào ngày 15/12/2024, tiêu chí hoàn thành cho nhiệm vụ này là các script phù hợp với website, dễ nhìn, dễ quản lý, trạng thái DONE và không có ghi chú nào.
- Nhiệm vụ Thực hiện kiểm thử, thời gian bắt đầu từ ngày 16/12/2024, kết thúc vào ngày 22/12/2024, tiêu chí chấp nhận cho nhiệm vụ này là có báo cáo cho biết số lượng test cases PASS/FAIL, trạng thái DONE và không có ghi chú.

3.3. Kịch bản kiểm thử thủ công trong website

Xác định các đề mục có trong kịch bản bên cạnh đó áp dụng một số kỹ thuật trong xây dựng kịch bản kiểm thử như: Phân vùng tương đương, phân tích giá trị biên, đoán lỗi, kiểm thử chuyển trạng thái, bảng quyết định để thực hiện xây dựng các kịch bản kiểm thử cho từng chức năng của trang web Divimenti. Trong đó kịch bản kiểm thử của trang web sẽ bao gồm các tiêu đề sau:

- **ID**
 - Mô tả: Đây là mã định danh duy nhất cho từng test case.
 - Ý nghĩa: Giúp dễ dàng theo dõi và quản lý từng trường hợp kiểm thử.
 - Ví dụ: "RE_001", "RE_002" đại diện cho các trường hợp kiểm thử liên quan đến module "Register".
- **Group Name**
 - Mô tả: Tên nhóm kiểm thử (phân nhóm theo chức năng hoặc loại kiểm thử).
 - Ý nghĩa: Tổ chức các test case thành từng nhóm để quản lý.
 - Ví dụ:
 - Trigger: Nhóm kiểm thử về các hành động kích hoạt như nhấn nút, link.
 - UIDesign: Nhóm kiểm thử về thiết kế giao diện (font, size, màu sắc).
- **Sub Group Name**
 - Mô tả: Nhóm con trong từng nhóm kiểm thử, mô tả chi tiết hơn.
 - Ý nghĩa: Làm rõ nội dung hoặc phạm vi của nhóm kiểm thử chính.
 - Ví dụ: Trong "Trigger", sub-group có thể bao gồm "Link" hoặc "Button".
- **Test Case Description**
 - Mô tả: Mô tả ngắn gọn và chi tiết về mục đích kiểm thử.
 - Ý nghĩa: Đưa ra thông tin về chức năng, hành vi hoặc điều kiện kiểm thử.

- Ví dụ: "Truy cập màn hình Đăng ký bằng link", "Kiểm tra số lượng các fields trên màn hình".
- **Preconditions**
 - Mô tả: Điều kiện tiên quyết cần thiết để thực hiện test case.
 - Ý nghĩa: Đảm bảo rằng môi trường và trạng thái hệ thống sẵn sàng cho bài kiểm thử.
 - Ví dụ: “Click vào link”, “Mở màn hình Đăng ký”.
- **Test Case Procedure**
 - Mô tả: Các bước chi tiết để thực hiện test case.
 - Ý nghĩa: Cung cấp hướng dẫn cụ thể để kiểm thử viên có thể thực hiện từng bước mà không nhầm lẫn.
 - Ví dụ:
 - Click vào nút “Register”.
 - Xem màn hình đăng ký hiện ra.
- **Test Data**
 - Mô tả: Dữ liệu cần thiết để thực hiện kiểm thử.
 - Ý nghĩa: Đảm bảo rằng các test case được kiểm tra với dữ liệu cụ thể và có thể tái tạo.
 - Ví dụ: URL truy cập (<http://127.0.0.1:8000/user/register>).
- **Expected Output**
 - Mô tả: Kết quả mong đợi nếu chức năng hoạt động đúng.
 - Ý nghĩa: Là cơ sở để so sánh với kết quả thực tế và đánh giá trạng thái của test case (Pass/Fail).
 - Ví dụ: “Hiển thị trang Đăng ký”, “Hiển thị đúng số lượng fields trên form”.
- **Result**
 - Mô tả: Trạng thái của test case sau khi kiểm thử.
 - Ý nghĩa: Ghi nhận kết quả thực tế, giúp đánh giá mức độ hoàn thành của chức năng hoặc tính năng.
 - Trạng thái sẽ bao gồm:
 - Pass: Kết quả thực tế khớp với kết quả mong đợi.

- Fail: Kết quả thực tế không khớp với kết quả mong đợi.

3.3.1. Testcase kiểm thử chức năng Đăng ký

Dựa vào tài liệu đặc tả của phần mềm và phương pháp xây dựng kịch bản kiểm thử từ đó thực hiện xây dựng kịch bản kiểm thử cho chức năng Đăng ký. Cụ thể đối với chức năng Đăng ký của trang web Divimenti thực hiện viết các test case tập trung vào các trạng thái thay đổi của chức năng này, kết quả như hình bên dưới.

Module Code	Register							
Test requirement	Đăng ký thành công							
Tester	Nguyễn Minh Hiếu							
Result	Pass	Fail	Untested	N/A	Number of Test			
	64	0	0	0	64			
ID	Group Name	Sub Group Name	Test Case Description	Precondition	Test Case Procedure	Test data	Expected Output	Result
RE_001	Trigger	Link	Truy cập màn hình Đăng ký bằng link		1. Click vào link	http://127.0.0.1:8000/user/register	1. hiển thị trang Đăng ký	Pass
RE_002		Button	Truy cập màn hình Đăng ký bằng cách click button Đăng ký trên màn hình Đăng nhập		1. Click button "Register" trên màn hình đăng nhập		1. hiển thị trang Đăng ký	Pass
RE_003			Kiểm tra số lượng các fields trên màn hình		1. Kiểm tra số lượng các fields		1. Số lượng các fields trên màn hình như sau: - text box: Tài khoản, Email, Số điện thoại, Mật khẩu, Tên, Họ - button: Đăng ký, Hiện	Pass
RE_004	UI/Design		Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Hiện thị đúng theo design Design	Pass
RE_005			Kiểm tra trạng thái		1. Kiểm tra trạng thái các item		1. Các item đều ở đúng trạng thái enable/disable	Pass
RE_006			Kiểm tra giá trị default		1. Kiểm tra các giá trị default		1. Các text box đều hiển thị trắng, có tên trường: Tài khoản, Email, Mật khẩu, Số Điện Thoại, Tên, Họ	Pass
RE_007	Validation	All	Kiểm tra behavior của hệ thống khi bỏ trống toàn bộ thông tin		1. Click "Đăng ký"		1. Báo lỗi: "Vui lòng điền vào trường này."	Pass
RE_008		Tài khoản	Bỏ trống thông tin Tài khoản	Các trường khác được điền đầy đủ	1. Bỏ trống thông tin trường Tài khoản 2. Click "Đăng ký"		2. Báo lỗi: "Vui lòng điền vào trường này."	Pass

RE_060			Click 2 lần khi trường Mật khẩu có dữ liệu	Trường Mật khẩu có dữ liệu	1. Click lần 1 button 2. Click lần 2 button		1. Button "Hiện" đổi thành "Ẩn", textbox Mật khẩu hiển thị các ký tự đã nhập 2. Button "Ẩn" đổi thành "Hiện", textbox Mật khẩu hiển thị chấm tròn	Pass	▼
RE_061		button "Đăng ký"	Kiểm tra behavior của Đăng ký btn khi nhập đầy đủ các thông tin hợp lệ		1. Nhập đầy đủ thông tin hợp lệ cho tất cả các trường 2. Click "Đăng ký"		2. Lưu thông tin user vào DB và Hiển thị trang "Đăng nhập"	Pass	▼
RE_062			Kiểm tra behavior của Đăng ký btn khi nhập các thông tin không hợp lệ		1. Nhập thông tin không hợp lệ cho 1 trường bắt buộc bất kỳ 2. Click "Đăng ký"		2. Báo lỗi tại trường không hợp lệ, đăng ký không thành công	Pass	▼
RE_063		Link "Đăng nhập"	Kiểm tra behavior của link "Đăng nhập" khi nhập đầy đủ các thông tin hợp lệ		1. Nhập đầy đủ thông tin hợp lệ cho tất cả các trường 2. Click "Đăng nhập"		2. Chuyển sang trang Đăng nhập	Pass	▼
RE_064			Kiểm tra behavior của link "Đăng nhập" khi nhập các thông tin không hợp lệ		1. Nhập thông tin không hợp lệ cho 1 trường bắt buộc bất kỳ 2. Click "Đăng nhập"		2. Chuyển sang trang Đăng nhập	Pass	▼

Hình 3.7 Trường hợp kiểm thử chức năng Đăng ký

Sau khi hoàn thành xong kịch bản kiểm thử cho chức năng Đăng ký như “*Hình 3.7*” đã đưa ra được 64 trường hợp kiểm thử (test case), tiêu đề và mô tả các bước thực thi của các trường hợp kiểm thử (test case) khá rõ ràng, kết quả mong đợi (Expected Output) mô tả đầy đủ, chi tiết.

3.3.2 Testcase kiểm thử chức năng Đăng nhập

Dựa vào tài liệu đặc tả của phần mềm và phương pháp xây dựng kịch bản kiểm thử từ đó thực hiện xây dựng kịch bản kiểm thử cho chức năng Đăng nhập. Cụ thể đối với chức năng Đăng nhập của trang web Divimenti thực hiện viết các test case tập trung vào các trạng thái thay đổi và hành động phản hồi của chức năng này, kết quả đạt được như hình bên dưới.

Module	Login							
Test requireme	Đăng nhập thành công, chuyển đến trang chủ							
Tester	Nguyễn Minh Hiếu							
Result	Pass	Fail	Untested	N/A	Number of Test			
	21	2	0	0	23			

ID	Group Name	Sub Group Name	Test Case Description	Precondition	Test Case Procedure	Test data	Expected Output	Result
Login_001	Trigger	Link	Kiểm tra khi truy cập trang bằng link		1. Click link	http://127.0.0.1:8000/user/login	1. hiển thị trang đăng nhập	Pass
Login_002		Button	Kiểm tra khi click button "Login"		1. Click button "Login" trên màn hình đăng ký		1. hiển thị trang đăng nhập	Pass
Login_003	UI/Design		Kiểm tra số lượng các fields trên màn hình		1. Kiểm tra số lượng các fields		1. Số lượng các fields trên màn hình như sau: - text box: Tài khoản, Mật khẩu - button: Login	Pass
Login_004			Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Hiển thị đúng theo design	Pass
Login_005			Kiểm tra trạng thái		1. Kiểm tra trạng thái các item		1. Các item đều ở đúng trạng thái enable/disable	Pass
Login_006			Kiểm tra giá trị default		1. Kiểm tra các giá trị default		1. Các text box đều hiển thị trắng, có tên trường Tài khoản, mật khẩu	Pass
Login_008		Tài khoản	Bỏ trống thông tin Tài khoản		1. Bỏ trống thông tin trường Tài khoản		1. Báo lỗi: "Vui lòng nhập vào trường này"	Pass
Login_009			Nhập 1 ký tự vào trường Tài khoản		1. Nhập 1 ký tự vào trường Tài khoản	h	1. Hiện thị ký tự đã nhập trên textbox Tài khoản	Pass
Login_010			Nhập 10 ký tự vào trường Tài khoản		1. Nhập 10 ký tự vào trường Tài khoản	huehue1234	1. Hiện thị ký tự đã nhập trên textbox Tài khoản	Pass
Login_011			Nhập 100 ký tự vào trường Tài khoản		1. Nhập 100 ký tự vào trường Tài khoản	hue123.... (100 ký tự)	1. Hiện thị ký tự đã nhập trên textbox Tài khoản	Pass
Login_012			Nhập 101 ký tự vào trường Tài khoản		1. Nhập 101 ký tự vào trường Tài khoản	hue123.... (101 ký tự)	1. Không nhập được ký tự thứ 101	Pass
Login_013		Mật khẩu	Bỏ trống thông tin Mật khẩu		1. Bỏ trống thông tin trường Mật khẩu		2. Báo lỗi: "Vui lòng nhập vào trường này"	Pass
Login_014			Nhập 1 ký tự vào trường Mật khẩu		1. Nhập 1 ký tự vào trường Mật khẩu	h	1. Hiện thị 1 chấm tròn	Pass
Login_015	Validation		Nhập 5 ký tự vào trường Password		1. Nhập 5 ký tự vào trường Password 2. Click ra khỏi textbox Password	12345	1. Hiện thị 5 chấm tròn	Pass
Login_016			Nhập 8 ký tự vào trường Password		1. Nhập 8 ký tự vào trường Password 2. Click ra khỏi textbox Password	12345678	1. Hiện thị 8 chấm tròn	Pass

Login_019			Kiểm tra behavior của Đăng nhập btn khi nhập Tài khoản hoặc mật khẩu không đúng trong DB		1. Nhập thông tin không hợp lệ cho 1 trường bắt buộc bất kỳ 2. Click "Đăng nhập"	2. Báo lỗi "Please enter a correct username and password. Note that both fields may be case-sensitive."	Pass
Login_020		link Đăng ký	Kiểm tra behavior của link "Đăng ký" khi nhập đầy đủ các thông tin hợp lệ		1. Nhập đầy đủ thông tin hợp lệ cho tất cả các trường 2. Click "Đăng ký"	2. Chuyển sang trang Đăng ký	Pass
Login_021			Kiểm tra behavior của link "Đăng ký" khi nhập các thông tin không hợp lệ		1. Nhập thông tin không hợp lệ cho 1 trường bắt buộc bất kỳ 2. Click "Đăng nhập"	2. Chuyển sang trang Đăng ký	Pass
Login_022		link Quên mật khẩu	Kiểm tra behavior của link "Đăng ký" khi nhập đầy đủ các thông tin hợp lệ		1. Nhập đầy đủ thông tin hợp lệ cho tất cả các trường 2. Click "Quên mật khẩu"	2. Hệ thống gửi mã OTP về số điện thoại đã đăng ký và chuyển sang trang nhập mã OTP	Fail
Login_023			Kiểm tra behavior của link "Đăng ký" khi nhập các thông tin không hợp lệ		1. Nhập thông tin không hợp lệ cho 1 trường bắt buộc bất kỳ 2. Click "Quên mật"	2. Hệ thống gửi mã OTP về số điện thoại đã đăng ký và chuyển sang trang nhập mã OTP	Fail

Hình 3.8 Trường hợp kiểm thử chức năng Đăng nhập

Sau khi hoàn thành xong kịch bản kiểm thử cho chức năng Đăng nhập như “Hình 3.8” đã xây dựng và đưa ra được 23 trường hợp kiểm thử (test case), các tiêu đề và mô tả các bước thực thi của các trường hợp kiểm thử (test case) khá rõ ràng, kết quả mong đợi (Expected Output) mô tả đầy đủ, chi tiết.

3.3.3. Testcase kiểm thử chức năng Đăng bài

Dựa vào tài liệu đặc tả của phần mềm và phương pháp xây dựng kịch bản kiểm thử từ đó thực hiện xây dựng kịch bản kiểm thử cho chức năng Đăng bài. Cụ thể đối với chức năng Đăng bài của trang web Divimenti thực hiện viết các test case tập trung vào các trạng thái thay đổi và hành động phản hồi của chức năng này, kết quả đạt được như hình bên dưới.

Module Code	Post							
Test requirement	Đăng bài thành công							
Tester	Nguyễn Minh Hiếu							
Result	Pass	Fail	Untested	N/A	Number of Test cases			
	28	0	0	0	28			

ID	Group Name	Sub Group Name	Test Case Description	Precondition	Test Case Procedure	Test data	Expected Output	Result
POST_001	Trigger	Paste link	Truy cập màn hình bằng cách copy link trên cùng trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000/	1. hiển thị trang chủ	Pass
POST_002			Truy cập vào màn hình bằng cách copy paste link khác trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000/	1. hiển thị trang chủ	Pass
POST_003	UI/Design		Kiểm tra số lượng các fields trên màn hình		1. Kiểm tra số lượng các fileds		1. Số lượng các fields trên màn hình như sau: - text box: caption - items: Chọn ảnh	Pass
POST_004			Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Hiển thị đúng theo design Design	Pass
POST_005			Kiểm tra trạng thái		1. Kiểm tra trạng thái các item		1. Các item đều ở đúng trạng thái enable/disable	Pass

POST_007	Validation		Bỏ trống các thông tin		1. Click "Đăng"		Không đăng bài được	Pass
POST_008			Bỏ trống textbox + chọn ảnh		1. Điền tất cả các thông tin 2. Click "Đăng"		2. Hiển thị bài đăng mới ngay bên dưới Thời gian hiển thị "a few seconds ago" Nội dung là kí tự vừa	Pass
POST_009			Nhập 1 kí tự vào textbox + không chọn ảnh		1. Nhập 1 kí tự vào textbox 2. Click "Đăng"	h	2. Hiển thị bài đăng mới ngay bên dưới Thời gian hiển thị "few seconds ago" Nội dung là kí tự vừa	Pass
POST_010			Nhập 1 kí tự vào textbox + chọn ảnh		1. Nhập 10 kí tự vào textbox 2. Chọn ảnh 3. Click "Đăng"	h	2. Hiển thị bài đăng mới ngay bên dưới Thời gian hiển thị "a few seconds ago" Nội dung là kí tự vừa	Pass
POST_011			Nhập 10 kí tự vào textbox + không chọn ảnh		1. Nhập 1 kí tự vào textbox 2. Click "Đăng"	huehue1235	2. Hiển thị bài đăng mới ngay bên dưới Thời gian hiển thị "a few seconds ago" Nội dung là kí tự vừa	Pass

POST_022		Combo box "Chế độ bình luận"	Chọn "Người theo dõi"		1. Chọn "Người theo dõi"		1. Hiện thị "Người theo dõi", chỉ người theo dõi bình luận được bài viết	Pass	▼
POST_023			Chọn "Người ủng hộ"		1. Chọn "Người ủng hộ"		1. Hiện thị "Người ủng hộ", chỉ người ủng hộ bình luận được bài viết	Pass	▼
POST_024			Chọn "Bạn bè"		1. Chọn "Bạn bè"		1. Hiện thị "Bạn bè", chỉ bạn bè bình luận được bài viết	Pass	▼
POST_025			Chọn "Chỉ mình tôi"		1. Chọn "Chỉ mình tôi"		1. Hiện thị "Chỉ mình tôi", chỉ chủ tài khoản bình luận được bài viết	Pass	▼
POST_026			Chọn "Công khai"		1. Chọn "Công khai"		1. Hiện thị "Công khai", tất cả mọi người đều bình luận được bài viết	Pass	▼
POST_027		button "Đăng"	Kiểm tra behavior của Đăng btn khi nhập caption hoặc ảnh		1. Nhập aption hoặc chọn ảnh 2. Click "Đăng"	dfgdfgdgde	2. Đăng bài thành công	Pass	▼
POST_028			Kiểm tra behavior của Đăng btn khi không nhập caption và ảnh		1. Bỏ trống caption và ảnh 2. Click "Đăng"		Không đăng bài được	Pass	▼

Hình 3.9 Trường hợp kiểm thử chức năng Đăng bài

Sau khi hoàn thành xong kịch bản kiểm thử cho chức năng Đăng bài như “Hình 3.9” đã xây dựng và đưa ra được 28 trường hợp kiểm thử (test case), các tiêu đề và mô tả các bước thực thi của các trường hợp kiểm thử (test case) khá rõ ràng, kết quả mong đợi (Expected Output) mô tả đầy đủ, chi tiết.

3.3.4 Test case kiểm thử chức năng Tìm kiếm bạn bè

Dựa vào tài liệu đặc tả của phần mềm và phương pháp xây dựng kịch bản kiểm thử từ đó thực hiện xây dựng kịch bản kiểm thử cho chức năng Tìm kiếm bạn bè. Cụ thể đối với chức năng Tìm kiếm bạn bè của trang web Divimenti thực hiện viết các test case tập trung vào các trạng thái thay đổi và hành động phản hồi của chức năng này, kết quả đạt được như hình bên dưới.

Module Code	Search							
Test requirement	Tìm kiếm bạn bè							
Tester	Nguyễn Minh Hiếu							
Result	Pass	Fail	Untested	N/A	Number of Test cases			
	11	0	0	0	11			

ID	Group Name	Sub Group Name	Test Case Description	Precondition	Test Case Procedure	Test data	Expected Output	Result
Search_001	Trigger	Paste link	Truy cập màn hình bằng cách copy link trên cùng trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000/	1. hiển thị trang chủ	Pass
Search_002			Truy cập vào màn hình bằng cách copy paste link khác trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000/	1. hiển thị trang chủ	Pass
Search_003	UI/Design		Kiểm tra số lượng các fields trên màn hình		1. Kiểm tra số lượng các fields		1. Số lượng các fields trên màn hình như sau: - text box: Searchbox - icon Search	Pass
Search_004			Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Hiển thị đúng theo design Design	Pass
Search_005			Kiểm tra trạng thái		1. Kiểm tra trạng thái các item		1. Các item đều ở đúng trạng thái enable/disable	Pass
Search_006			Kiểm tra giá trị default		1. Kiểm tra các giá trị default		1. Textbox hiển thị "Start typing to search..."	Pass
Search_007	Validation	Search	Bỏ trống ô Search		1. Bỏ trống ô Search 2. Nhấn phím Enter		2. Hiển thị tất cả người dùng	Pass
Search_008			Nhập 1 ký tự vào ô Search và có kết quả tìm kiếm		1. Nhập 1 ký tự vào ô Search 2. Nhấn phím Enter	h	1. Hiển thị nội dung nhập trên ô Search 2. Hiển thị "Search result for: h" và danh sách người	Pass
Search_009			Nhập 10 ký tự vào ô Search và có kết quả tìm kiếm		1. Nhập 10 ký tự vào ô Search 2. Nhấn phím Enter	huehue1234	1. Hiển thị nội dung nhập trên ô Search 2. Hiển thị "Search result for: huehue1234" và danh sách người dùng	Pass
Search_010			Nhập 100 ký tự vào ô Search và không có kết quả tìm kiếm		1. Nhập 100 ký tự vào ô Search 2. Nhấn phím Enter	hue123.... (100 ký tự)	1. Hiển thị nội dung nhập trên ô Search 2. Hiển thị "Search result for: hdgfhgdghd...." và danh	Pass
Search_011			Nhập 101 ký tự vào ô Search		1. Nhập 101 ký tự vào ô Search	hue123.... (101 ký tự)	Không nhập dc ký tự 101	Pass

Hình 3.10 Trường hợp kiểm thử chức năng Tìm kiếm bạn bè

Sau khi hoàn thành xong kịch bản kiểm thử cho chức năng Tìm kiếm bạn bè như “Hình 3.10” đã xây dựng và đưa ra được 11 trường hợp kiểm thử (test case), các tiêu đề và mô tả các bước thực thi của các trường hợp kiểm thử (test case) khá rõ ràng, kết quả mong đợi (Expected Output) mô tả đầy đủ, chi tiết.

3.3.5 Test case chức năng Nhắn tin

Dựa vào tài liệu đặc tả của phần mềm và phương pháp xây dựng kịch bản kiểm thử từ đó thực hiện xây dựng kịch bản kiểm thử cho chức năng Nhắn tin. Cụ thể đối với chức năng Nhắn tin của trang web Divimenti thực hiện viết các test case tập trung vào các trạng thái thay đổi của chức năng này, kết quả đạt được như hình bên dưới.

Module Code	Send_Message							
Test requirement	Nhắn tin							
Tester	Nguyễn Minh Hiếu							
Result	Pass	Fail	Untested	N/A	Number of Test cases			
	13	0	0	0	13			

ID	Group Name	Sub Group Name	Test Case Description	Precondition	Test Case Procedure	Test data	Expected Output	Result
SM_001	Trigger	Paste link	Truy cập màn hình bằng cách copy link trên cùng trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000/chat/quanghuy/	1. hiển thị cuộc trò chuyện	Pass
SM_002			Truy cập vào màn hình bằng cách copy paste link khác trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000/chat/quanghuy/	1. hiển thị cuộc trò chuyện	Pass
SM_003		icon Chat	Truy cập màn hình bằng cách click icon Chat ở góc trên bên phải màn hình	Đã login vào hệ thống	1. Click icon		1. hiển thị cuộc trò chuyện	Pass

SM_009	Validation	InputMessage	Bỏ trống textbox InputMessage		1. Bỏ trống textbox InputMessage 2. Click "Send"		2. Không gửi được tin nhắn	Pass
SM_010			Nhập 1 ký tự vào textbox InputMessage		1. Nhập 1 ký tự vào textbox InputMessage 2. Click "Send"	a	1. Hiện thị các ký tự đã nhập trên box 2. Tin nhắn nhảy lên hộp thoại với nội dung như đã nhập	Pass
SM_011			Nhập 20 ký tự vào textbox InputMessage		1. Nhập 20 ký tự vào textbox InputMessage 2. Click "Send"	hjsjskaleiwjx kaldjs	1. Hiện thị các ký tự đã nhập trên box 2. Tin nhắn nhảy lên hộp thoại với nội dung như đã nhập	Pass
SM_012			Nhập 200 ký tự vào textbox InputMessage		1. Nhập 200 ký tự vào textbox InputMessage 2. Click "Send"	sdstdsf.. (200 ký tự)	1. Hiện thị các ký tự đã nhập trên box 2. Tin nhắn nhảy lên hộp thoại với nội dung như đã nhập	Pass
SM_013			Gửi tin nhắn bằng phím Enter		1. Nhập 20 ký tự vào textbox InputMessage 2. Click phím Enter	sdstdsf.. (20 ký tự)	1. Hiện thị các ký tự đã nhập trên box 2. Tin nhắn nhảy lên hộp thoại với nội dung như đã nhập	Pass

Hình 3.11 Trường hợp kiểm thử chức năng Nhắn tin

Sau khi hoàn thành xong kịch bản kiểm thử cho chức năng Đăng bài như “Hình 3.11” đã xây dựng và đưa ra được 13 trường hợp kiểm thử (test case), các tiêu đề và mô tả các bước thực thi của các trường hợp kiểm thử (test case) khá rõ ràng, kết quả mong đợi (Expected Output) mô tả đầy đủ, chi tiết.

3.3.6 Trường hợp kiểm thử chức năng Bình luận

Dựa vào tài liệu đặc tả của phần mềm và phương pháp xây dựng kịch bản kiểm thử từ đó thực hiện xây dựng kịch bản kiểm thử cho chức năng Bình luận. Cụ thể đối với chức năng Bình luận của trang web Divimenti thực hiện viết các test case tập trung vào các trạng thái thay đổi của chức năng này, kết quả đạt được như hình bên dưới.

Module Code	Comment							
Test requirement	Bình luận							
Tester	Nguyễn Minh Hiếu							
Result	Pass	Fail	Untested	N/A	Number of Test			
	12	0	0	0	12			

ID	Group Name	Sub Group Name	Test Case Description	Precondition	Test Case Procedure	Test data	Expected Output	Result
CM_001	Trigger	Paste link	Truy cập màn hình bằng cách copy link trên cùng trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8000	1. hiển thị cuộc trò chuyện	Pass
CM_002			Truy cập vào màn hình bằng cách copy paste link khác trình duyệt	Đã login vào hệ thống	1. Click link	http://127.0.0.1:8001	1. hiển thị cuộc trò chuyện	Pass
CM_005	UI/Design		Kiểm tra số lượng các fields trên màn hình		1. Kiểm tra số lượng các fields		1. Số lượng các fields trên màn hình như sau: - textbox: InputComment - button: Post	Pass
CM_006			Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Kiểm tra vị trí, font chữ, size chữ, màu sắc		1. Hiển thị đúng theo design Design	Pass
CM_007			Kiểm tra trạng thái		1. Kiểm tra trạng thái các item		1. Các item đều ở đúng trạng thái enable/disable	Pass

CM_009	Validation	InputComment	Bỏ trống textbox		1. Bỏ trống textbox 2. Click "Post"		2. Không bình luận được	Pass
CM_010			Nhập 1 ký tự vào textbox		1. Nhập 1 ký tự vào textbox 2. Click "Post"	a	1. Hiện thị các ký tự đã nhập trên box 2. Bình luận hiển thị với nội dung như đã nhập	Pass
CM_011			Nhập 10 ký tự vào textbox		1. Nhập 10 ký tự vào textbox 2. Click "Post"	1234567890	1. Hiện thị các ký tự đã nhập trên box 2. Bình luận hiển thị với nội dung như đã nhập	Pass
CM_012			Nhập 100 ký tự vào textbox		1. Nhập 100 ký tự vào textbox 2. Click "Post"	sdsdsfsff.. (100 ký tự)	1. Hiện thị các ký tự đã nhập trên box 2. Bình luận hiển thị với nội dung như đã nhập	Pass
CM_013	Bussiness	button "Post"	Kiểm tra behavior của Đăng btn khi nhập caption hoặc ảnh		1. Nhập nội dung bình luận 2. Click "Post"	dfgdfgdgdge	2. Bình luận thành công	Pass
CM_014			Kiểm tra behavior của Đăng btn khi không nhập caption và ảnh		1. Bỏ trống nội dung bình luận 2. Click "Post"		Không bình luận được	Pass

Hình 3.12 Trường hợp kiểm thử chức năng Bình luận

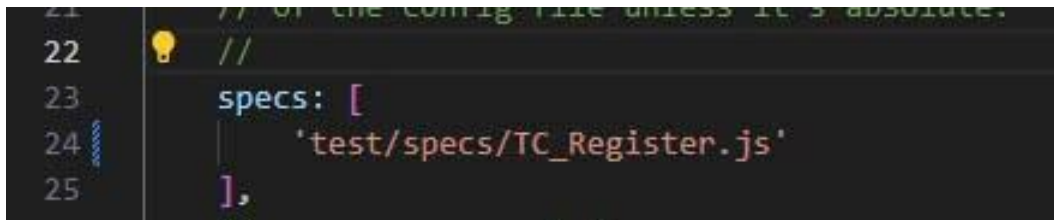
Sau khi hoàn thành xong kịch bản kiểm thử cho chức năng Bình luận như “Hình 3.12” đã xây dựng và đưa ra được 14 trường hợp kiểm thử (test case), các tiêu đề và mô tả các bước thực thi của các trường hợp kiểm thử (test case) khá rõ ràng, kết quả mong đợi (Expected Output) mô tả đầy đủ, chi tiết.

3.4 Kiểm thử tự động các chức năng

Dựa vào kịch bản kiểm thử đã xây dựng cho từng chức năng của trang web sau đó sử dụng công cụ Selenium WebDriver áp dụng test một số chức năng trên trang web Divimenti.

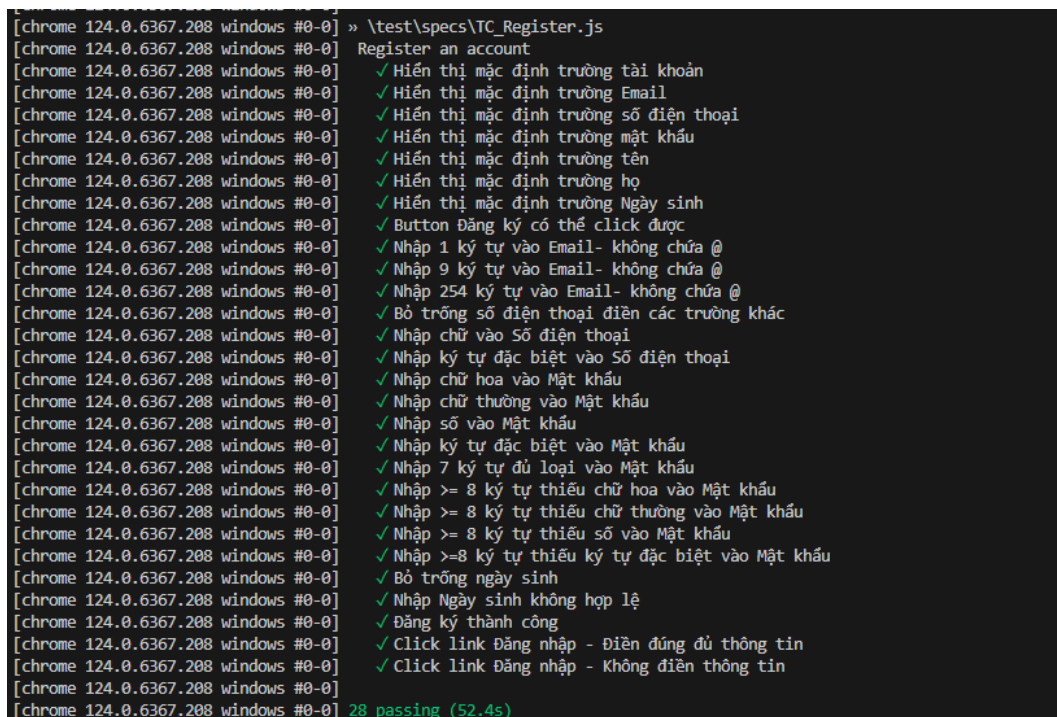
3.4.1. Kiểm thử tự động chức năng Đăng ký

- Truy cập vào file wdio.conf.js
- Tiếp theo thực hiện sửa tên file tại mục spec



Hình 3.13 Sửa tên file test case Đăng ký

- Bật terminal trở đến file và chạy lệnh “*npm run wdio*”
- Hệ thống tự động bật trình duyệt chrome lên và thực hiện chạy file test script
- Sau khi thực hiện hoàn tất quá trình chạy các script thu được kết quả như hình sau:

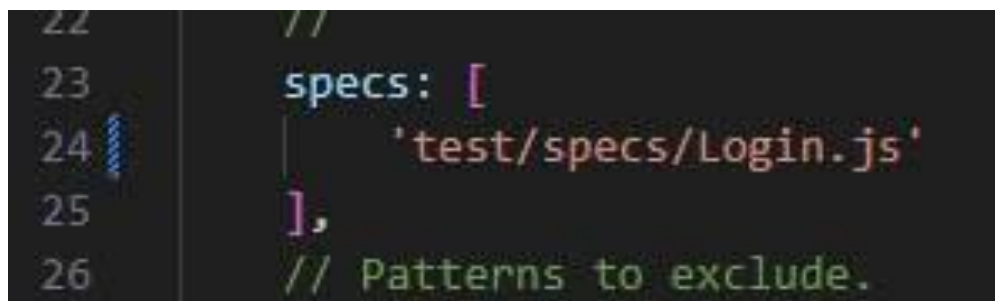


Hình 3.14 Kết quả kiểm thử tự động chức năng Đăng ký

- Đánh giá kết quả:
 - ✓ Sau khi thực hiện chạy các script thu được kết quả như “Hình 3.14” cho thấy rằng tất cả 28 trường hợp kiểm thử (test case) liên quan đến chức năng đăng ký tài khoản của website Divimenti đều được thực thi thành công (passing) mà không gặp lỗi.
 - ✓ Các kịch bản kiểm thử bao quát đầy đủ các trường hợp quan trọng, từ kiểm tra các trường mặc định (Email, Số điện thoại, Mật khẩu, Ngày sinh, v.v.) đến các kịch bản kiểm tra logic đầu vào (như định dạng email, độ dài và thành phần mật khẩu).
 - ✓ Thời gian thực hiện 52.4 giây cho 28 test case là hợp lý, cho thấy quá trình kiểm thử tự động được tối ưu hóa.

3.4.2. Kiểm thử tự động chức năng Đăng nhập

- Truy cập vào file wdio.conf.js
- Tiếp theo thực hiện sửa tên file tại mục spec thành “test/specs/Login.js”



```

22 //
23 specs: [
24   'test/specs/Login.js'
25 ],
26 // Patterns to exclude.

```

Hình 3.15 Sửa tên file test case Đăng nhập

- Bật terminal trở đến file và chạy lệnh “npm run wdio”
- Hệ thống tự động bật trình duyệt chrome lên và thực hiện chạy file test script
- Sau khi thực hiện hoàn tất quá trình chạy các script thu được kết quả như hình bên dưới.

```

Login application
✓ Hiện thị mặc định trường Tài khoản
✓ Hiện thị mặc định trường Mật khẩu
✓ Button Đăng nhập có thể click được
✓ Username không chính xác
✓ Username chính xác - Password không chính xác
✓ Password không chính xác
✓ Password chính xác chữ thường
✓ Login OK
✓ Click link Đăng ký - Điền đúng đủ thông tin
✓ Click link Đăng ký - Không điền thông tin
✗ Click link Quên mật khẩu - Điền đúng đủ thông tin
✗ Click link Quên mật khẩu - Không điền thông tin

10 passing (48.6s)
2 failing

```

```

[chrome 131.0.6778.140 windows #0-0]
[chrome 131.0.6778.140 windows #0-0] 1) Login application Click link Quên mật khẩu - Điền đúng đủ thông tin
[chrome 131.0.6778.140 windows #0-0] Expect $('[type="submit"]') not to exist

Expected [not]: "not exist"
Received      : "exist"
[chrome 131.0.6778.140 windows #0-0] Error: Expect $('[type="submit"]') not to exist
[chrome 131.0.6778.140 windows #0-0]
[chrome 131.0.6778.140 windows #0-0] Expected [not]: "not exist"
[chrome 131.0.6778.140 windows #0-0] Received      : "exist"
[chrome 131.0.6778.140 windows #0-0]
[chrome 131.0.6778.140 windows #0-0] 2) Login application Click link Quên mật khẩu - Không điền thông tin
[chrome 131.0.6778.140 windows #0-0] Expect $('[type="submit"]') not to exist

Expected [not]: "not exist"
Received      : "exist"
[chrome 131.0.6778.140 windows #0-0] Error: Expect $('[type="submit"]') not to exist
[chrome 131.0.6778.140 windows #0-0]
[chrome 131.0.6778.140 windows #0-0] Expected [not]: "not exist"
[chrome 131.0.6778.140 windows #0-0] Received      : "exist"
[chrome 131.0.6778.140 windows #0-0]

```

Hình 3.16 Kết quả kiểm thử tự động chức năng Đăng nhập

- Đánh giá kết quả:
 - ✓ Sau khi thực hiện chạy các script thu được kết quả như “Hình 3.16” cho thấy rằng có 10/12 test cases PASS và 2/12 test case FAIL liên quan đến chức năng đăng ký tài khoản của website Divimenti cụ thể:

Diễn tích cực

- Độ ổn định:
 - 10 bài kiểm thử khác đều đã thành công, chứng minh các chức năng cơ bản như đăng nhập với tài khoản, mật khẩu đúng/sai và các trạng thái UI đã hoạt động ổn định.

- Thời gian chạy kiểm thử:
 - Tổng thời gian chạy chỉ mất 48.6 giây, chứng minh bộ kiểm thử có hiệu suất tốt.

Điểm cần cải thiện:

- Về hệ thống ứng dụng:
 - Luồng xử lý Quên mật khẩu:
 - Cần xem xét lại logic và trạng thái giao diện khi nhấn vào link "Quên mật khẩu".
 - Kiểm tra xem nút submit có cần tồn tại ở những bước này hay không.
- Về kịch bản kiểm thử:
 - Xác minh lại yêu cầu nghiệp vụ:
 - Đảm bảo kịch bản kiểm thử phù hợp với yêu cầu thiết kế.
 - Nếu nút submit xuất hiện là đúng, cần sửa kịch bản để khớp với hành vi thực tế.
- Về môi trường kiểm thử:
 - Đảm bảo không có lỗi do môi trường như dữ liệu không đồng bộ, cấu hình không khớp giữa môi trường dev/test.
- Đề xuất cải thiện và hướng giải quyết
 - Kiểm tra logic phía ứng dụng:
 - Kiểm tra xử lý trạng thái UI khi nhấn “Quên mật khẩu”.
 - Đảm bảo trạng thái nút submit (ẩn/hiện) phản ánh đúng yêu cầu thiết kế.
 - Cập nhật kiểm thử tự động:
 - Nếu hành vi nút submit là đúng, cập nhật test case để tránh kiểm tra trạng thái sai lệch.
 - Thêm kiểm thử chi tiết hơn:
 - Mở rộng kiểm thử để bao gồm:
 - Các hành động tiếp theo sau khi nhấn “Quên mật khẩu”.
 - Xử lý thông báo lỗi hoặc thông báo xác nhận khi người dùng không điền thông tin.

3.4.3 Kiểm thử tự động chức năng Đăng bài

- Truy cập vào file wdio.conf.js
- Tiếp theo thực hiện sửa tên file tại mục spec thành “test/specs/Post.js”



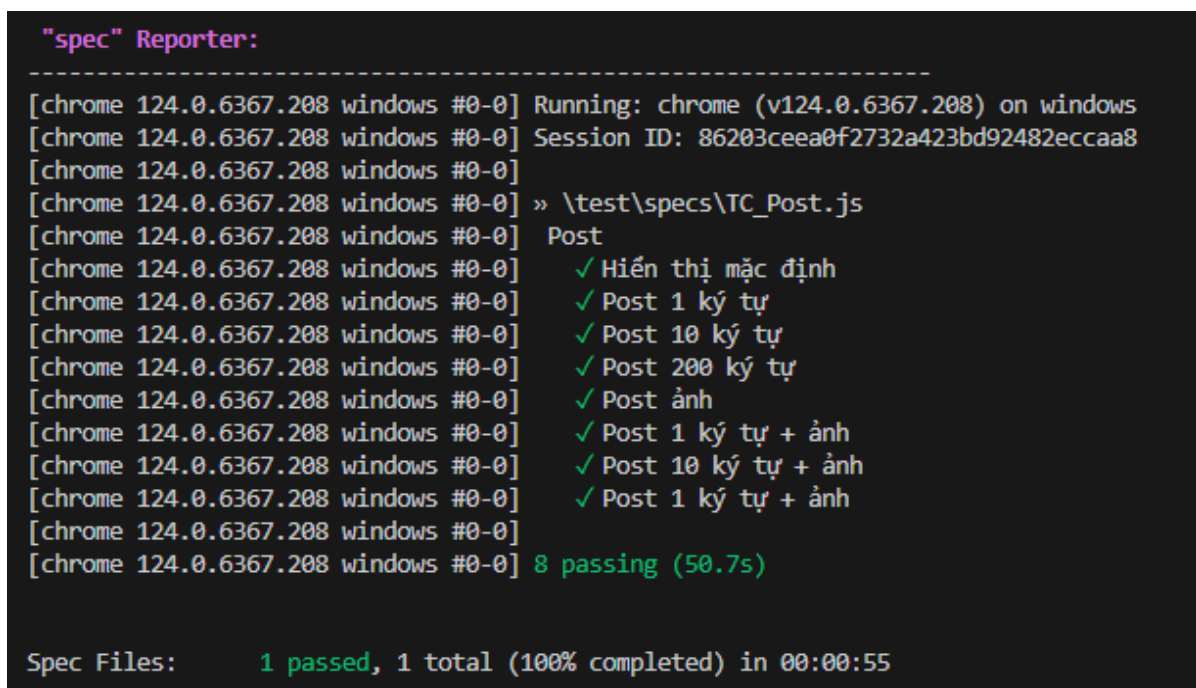
```

22 //
23 specs: [
24     'test/specs/Post.js'
25 ],

```

Hình 3.17 Sửa tên file chức năng Đăng bài

- Bật terminal trở đến file và chạy lệnh “*npm run wdio*”
- Hệ thống sẽ tự động bật trình duyệt chrome lên và thực hiện chạy file test script
- Sau khi thực hiện hoàn tất quá trình chạy các script thu được kết quả như hình bên dưới.



```

"spec" Reporter:
-----
[chrome 124.0.6367.208 windows #0-0] Running: chrome (v124.0.6367.208) on windows
[chrome 124.0.6367.208 windows #0-0] Session ID: 86203ceea0f2732a423bd92482eccaa8
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] » \test\specs\TC_Post.js
[chrome 124.0.6367.208 windows #0-0] Post
[chrome 124.0.6367.208 windows #0-0]   ✓ Hiển thị mặc định
[chrome 124.0.6367.208 windows #0-0]   ✓ Post 1 ký tự
[chrome 124.0.6367.208 windows #0-0]   ✓ Post 10 ký tự
[chrome 124.0.6367.208 windows #0-0]   ✓ Post 200 ký tự
[chrome 124.0.6367.208 windows #0-0]   ✓ Post ảnh
[chrome 124.0.6367.208 windows #0-0]   ✓ Post 1 ký tự + ảnh
[chrome 124.0.6367.208 windows #0-0]   ✓ Post 10 ký tự + ảnh
[chrome 124.0.6367.208 windows #0-0]   ✓ Post 1 ký tự + ảnh
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] 8 passing (50.7s)

Spec Files:      1 passed, 1 total (100% completed) in 00:00:55

```

Hình 3.18 Kết quả kiểm thử tự động chức năng Đăng bài

- Đánh giá kết quả:
 - ✓ Sau khi thực hiện chạy các script thu được kết quả như “Hình 3.18” cho thấy rằng tất cả 8 trường hợp kiểm thử (test case) liên quan đến chức năng Đăng bài của website Divimenti đều được thực thi thành công (passing) mà không gặp lỗi. Điều này cho thấy tính năng đang hoạt động đúng theo kỳ vọng.
 - ✓ Bộ test case khá chi tiết và bao phủ nhiều trường hợp sử dụng, đặc biệt cả với văn bản ngắn, dài và kết hợp với hình ảnh.
 - ✓ Thời gian thực thi cho 8 test cases là 50,7s khá tối ưu.

3.4.4 Kiểm thử tự động chức năng Nhắn tin

- Truy cập vào file wdio.conf.js
- Tiếp theo thực hiện sửa tên file tại mục spec thành “test/specs/Chat.js”

```

23     specs: [
24         'test/specs/Chat.js'
25     ],

```

Hình 3.19 Sửa tên file chức năng Nhắn tin

- Bật terminal trở đến file và chạy lệnh “`npm run wdio`”
- Hệ thống sẽ tự động bật trình duyệt chrome lên và thực hiện chạy file test script
- Sau khi thực hiện hoàn tất quá trình chạy các script thu được kết quả như hình bên dưới.

```

"spec" Reporter:
-----
[chrome 124.0.6367.208 windows #0-0] Running: chrome (v124.0.6367.208) on windows
[chrome 124.0.6367.208 windows #0-0] Session ID: aff7f82576c5f3512d142669f0947d56
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] » \test\specs\TC_Chats.js
[chrome 124.0.6367.208 windows #0-0] Nhắn tin cho bạn bè
[chrome 124.0.6367.208 windows #0-0]   ✓ Hiển thị mặc định
[chrome 124.0.6367.208 windows #0-0]   ✓ Nhập 1 ký tự vào Message box
[chrome 124.0.6367.208 windows #0-0]   ✓ Nhập 100 ký tự vào Message box
[chrome 124.0.6367.208 windows #0-0]   ✓ Gửi bằng phím Enter
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] 4 passing (8.7s)

Spec Files:      1 passed, 1 total (100% completed) in 00:00:12

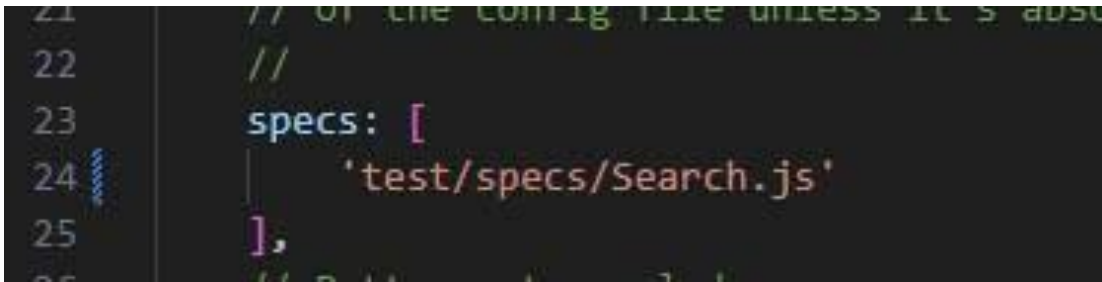
```

Hình 3.20 Kết quả kiểm thử tự động chức năng Nhắn tin

- Đánh giá kết quả:
 - ✓ Sau khi thực hiện chạy các script thu được kết quả như “Hình 3.20” cho thấy rằng tất cả 4 trường hợp kiểm thử (test case) liên quan đến chức năng Nhấn tin của website Divimenti đều được thực thi thành công (passing) mà không gặp lỗi. Điều này cho thấy tính năng đang hoạt động đúng theo kỳ vọng.
 - ✓ Thời gian chạy 8.7s, nhanh và hiệu quả.
 - ✓ Bộ test case khá chi tiết và bao phủ nhiều trường hợp sử dụng, đặc biệt cả với văn bản ngắn, dài và kết hợp với hình ảnh.

3.4.5 Kiểm thử tự động chức năng Tìm kiếm

- Truy cập vào file wdio.conf.js
- Tiếp theo thực hiện sửa tên file tại mục spec thành “test/specs/Search.js”



```

21 // Of the config file unless it's above
22 //
23 specs: [
24   'test/specs/Search.js'
25 ],

```

Hình 3.21 Sửa tên file chức năng Tìm kiếm

- Bật terminal trở đến file và chạy lệnh “*npm run wdio*”
- Hệ thống sẽ tự động bật trình duyệt chrome lên và thực hiện chạy file test script
- Sau khi thực hiện hoàn tất quá trình chạy các script thu được kết quả như hình bên dưới.

```

"spec" Reporter:
-----
[chrome 124.0.6367.208 windows #0-0] Running: chrome (v124.0.6367.208) on windows
[chrome 124.0.6367.208 windows #0-0] Session ID: b7a84a139aa2d122fe4461124e1a29f8
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] » \test\specs\TC_Search.js
[chrome 124.0.6367.208 windows #0-0]   Tìm kiếm bạn bè
[chrome 124.0.6367.208 windows #0-0]     ✓ Bỏ trống ô search
[chrome 124.0.6367.208 windows #0-0]     ✓ Nhập 1 ký tự vào ô search ô search
[chrome 124.0.6367.208 windows #0-0]     ✓ Nhập 10 ký tự vào ô search ô search
[chrome 124.0.6367.208 windows #0-0]     ✓ Nhập 100 ký tự vào ô search
[chrome 124.0.6367.208 windows #0-0] 4 passing (12.2s)

Spec Files:      1 passed, 1 total (100% completed) in 00:00:17

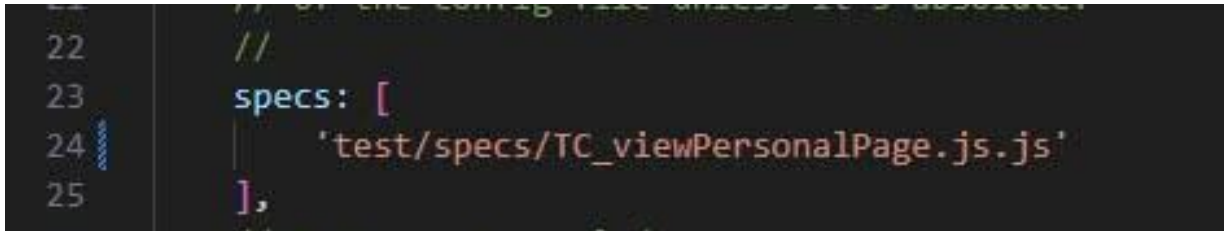
```

Hình 3.22 Kết quả kiểm thử tự động chức năng Tìm kiếm bạn bè

- Đánh giá kết quả:
 - ✓ Sau khi thực hiện chạy các script thu được kết quả như “Hình 3.22” cho thấy rằng tất cả 4 trường hợp kiểm thử (test case) liên quan đến chức năng Tìm kiếm bạn bè của website Divimenti bao gồm các trường hợp: Bỏ trống ô search, Nhập 1 ký tự vào ô search, Nhập 10 ký tự vào ô search, Nhập 100 ký tự vào ô search đều được thực thi thành công (passing) mà không gặp lỗi. Điều này cho thấy tính năng đang hoạt động đúng theo kỳ vọng.
 - ✓ Thời gian chạy cho 4 trường hợp kiểm thử là 12,2s từ đó đánh giá được quá trình chạy nhanh và hiệu quả.
 - ✓ Qua kết quả chạy trên có thể cho thấy được bộ test case khá chi tiết và bao phủ nhiều trường hợp sử dụng, đặc biệt cả với văn bản ngắn, dài và kết hợp với hình ảnh.

3.4.6 Kiểm thử tự động chức năng Bình luận

- Truy cập vào file wdio.conf.js
- Tiếp theo thực hiện sửa tên file tại mục spec thành “test/specs/TC_viewPersonalPage.js”



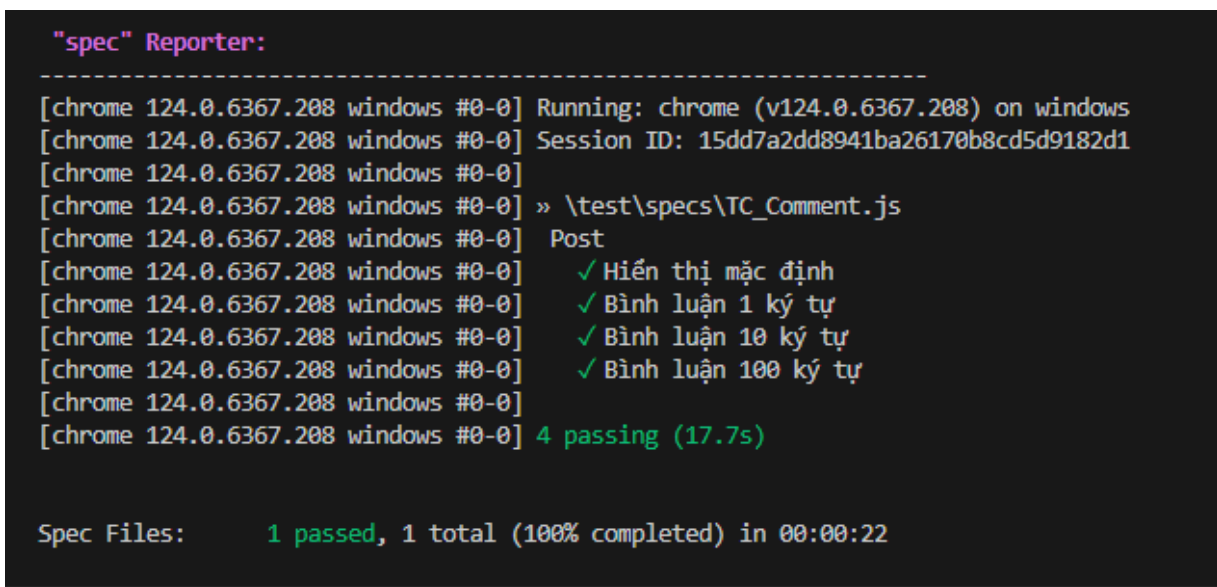
```

22 //
23 specs: [
24     'test/specs/TC_viewPersonalPage.js.js'
25 ],

```

Hình 3.23 Sửa tên file chức năng bình luận

- Bật terminal trở đến file và chạy lệnh “npm run wdio”
- Hệ thống sẽ tự động bật trình duyệt chrome lên và thực hiện chạy file test script
- Sau khi thực hiện hoàn tất quá trình chạy các script thu được kết quả như hình bên dưới.



```

"spec" Reporter:
-----
[chrome 124.0.6367.208 windows #0-0] Running: chrome (v124.0.6367.208) on windows
[chrome 124.0.6367.208 windows #0-0] Session ID: 15dd7a2dd8941ba26170b8cd5d9182d1
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] » \test\specs\TC_Comment.js
[chrome 124.0.6367.208 windows #0-0] Post
[chrome 124.0.6367.208 windows #0-0]   ✓ Hiển thị mặc định
[chrome 124.0.6367.208 windows #0-0]   ✓ Bình luận 1 ký tự
[chrome 124.0.6367.208 windows #0-0]   ✓ Bình luận 10 ký tự
[chrome 124.0.6367.208 windows #0-0]   ✓ Bình luận 100 ký tự
[chrome 124.0.6367.208 windows #0-0]
[chrome 124.0.6367.208 windows #0-0] 4 passing (17.7s)

Spec Files:      1 passed, 1 total (100% completed) in 00:00:22

```

Hình 3.24 Kết quả kiểm thử tự động chức năng Bình luận

- Đánh giá kết quả:
 - ✓ Sau khi thực hiện chạy các script thu được kết quả như “Hình 3.24” cho thấy rằng tất cả 4 trường hợp kiểm thử (test case) liên quan đến chức năng Bình luận của website Divimenti bao gồm các trường hợp:

Hiển thị mặc định, Bình luận 1 ký tự, Bình luận 10 ký tự, Bình luận 100 ký tự đều được thực thi thành công (passing) mà không gặp lỗi. Điều này cho thấy tính năng đang hoạt động đúng theo kỳ vọng.

- ✓ Thời gian chạy cho 4 trường hợp kiểm thử là 17,7s từ đó đánh giá được quá trình chạy khá nhanh và hiệu quả.
- ✓ Qua kết quả chạy trên có thể cho thấy được bộ test case khá chi tiết và bao phủ nhiều trường hợp sử dụng, đặc biệt cả với văn bản ngắn và dài.

3.5 Đánh giá kết quả kiểm thử

Sau khi thực hiện chạy các script cho tất cả các chức năng và thu được kết quả tiến hành viết tài liệu báo cáo kiểm thử (Test report) cho trang web Divimenti. Trong đó bao gồm các mục:

- **No (Số thứ tự)**
 - Mô tả: Đây là số thứ tự của từng module trong bảng.
 - Ý nghĩa: Giúp sắp xếp và dễ dàng tra cứu thông tin của từng module kiểm thử.
- **Module code (Mã module)**
 - Mô tả: Tên hoặc mã đại diện cho từng module của hệ thống cần kiểm thử.
 - Trong báo cáo này, các module đó là “Đăng ký”, “Đăng nhập”, “Tìm kiếm bạn bè”, “Nhắn tin”, “Đăng bài”, “Bình luận”.
- **Pass (Số test case thực thi thành công)**
 - Mô tả: Số lượng test cases đã thực hiện và **đạt yêu cầu** (kết quả kiểm thử khớp với kết quả mong đợi).
 - Ý nghĩa: Đây là số lượng bài kiểm thử hoạt động đúng chức năng mà không phát sinh lỗi.
- **Fail (Số test case thực thi thất bại)**
 - Mô tả: Số lượng test cases đã thực hiện nhưng **không đạt yêu cầu** (kết quả thực tế không khớp với kết quả mong đợi).
 - Ý nghĩa: Đây là số lỗi hoặc vấn đề cần được đội phát triển khắc phục.
- **Untested (Chưa kiểm thử)**

- Mô tả: Số lượng test cases **chưa được thực thi** (do thiếu thời gian, không đủ dữ liệu hoặc lỗi môi trường kiểm thử).
- Ý nghĩa: Cho thấy mức độ bao phủ kiểm thử chưa hoàn chỉnh, cần cải thiện để đạt 100% test coverage.
- **N/A (Không áp dụng)**
 - Mô tả: Số lượng test cases **không liên quan hoặc không áp dụng** cho module này (ví dụ: kiểm thử chức năng không tồn tại trong module cụ thể).
 - Ý nghĩa: Những trường hợp kiểm thử này không được xem xét trong phạm vi module.
- **Number of test cases (Tổng số test cases)**
 - Mô tả: Tổng số lượng test cases đã được định nghĩa cho từng module.
 - Ý nghĩa: Đây là tổng bài kiểm thử (bao gồm Pass, Fail, Untested, và N/A) trong một module.
- **Sub total (Tổng phụ)**
 - Mô tả: Tổng số lượng bài kiểm thử (Pass, Fail, Untested, N/A) của tất cả các module trong báo cáo.
 - Ý nghĩa: Dùng để tổng hợp nhanh kết quả kiểm thử cho toàn hệ thống.
- **Test coverage (Độ bao phủ của các test case)**
 - Mô tả: Phần trăm tổng số test cases được **thực thi** (Pass + Fail), so với tổng số test cases được định nghĩa.
 - Ý nghĩa: Cho thấy mức độ bao phủ của quá trình kiểm thử.
- **Test successful coverage (Tỉ lệ kiểm thử thành công)**
 - Mô tả: Phần trăm số bài kiểm thử thành công (Pass) so với tổng số bài kiểm thử đã thực thi.
 - Ý nghĩa: Đánh giá chất lượng của hệ thống dựa trên số lượng bài kiểm thử đạt yêu cầu.

Sau khi xác định các đề mục cần thiết và quan trọng trong tài liệu báo cáo kiểm thử (test plan), tiến hành sử dụng công cụ Microsoft Excel để hoàn thiện báo cáo, kết quả thu được như hình bên dưới.

TEST REPORT						
Project Name	Diviment					
Creator	Nông Minh Hiếu					
No	Module code	Pass	Fail	Untested	N/A	Number of test cases
1	Đăng ký	64	0	0	0	64
2	Đăng nhập	21	2	0	0	23
3	Đăng bài	28	0	0	0	28
4	Tìm kiếm bạn bè	11	0	0	0	11
5	Nhắn tin	13	0	0	0	13
6	Bình luận	12	0	0	0	12
Sub total		85	2	0	0	87
Test coverage				100,00	%	
Test successful coverage				97,70	%	

Hình 3.25 Test report

Sau khi hoàn thiện báo cáo kiểm thử (test plan) được kết quả như “Hình 3.25” từ đó có thể đánh giá được độ bao phủ của các test case và chất lượng của phần mềm, cụ thể:

- **Test coverage:** 100%, nghĩa là toàn bộ các test cases đã được thực hiện.
- **Test successful coverage:** 97,70%, cho thấy mức độ thành công khá cao.
- Tổng số test cases: 87.

Trong đó:

- 85 test cases pass.
- test cases fail (ở module Đăng nhập).
- Không có test case nào chưa thực hiện (Untested).

Điểm mạnh:

- Hoàn thành 100% các test cases, đảm bảo không bỏ sót các trường hợp kiểm thử.
- Mức độ thành công cao 97,7% chứng tỏ chất lượng phần mềm khá tốt.
- Phân loại kết quả kiểm thử từng module, dễ dàng đánh giá module nào hoạt động chưa ổn định.

Hạn chế:

- Module “Đăng nhập”: Có 2 test cases bị fail.

Đề xuất:

- Xử lý lỗi trong module Đăng nhập:
 - Xem xét chi tiết log của 2 test cases thất bại trong module này.
 - Phân tích nguyên nhân (bug logic, sai sót thiết kế, dữ liệu đầu vào không hợp lệ, v.v.).
- Kiểm tra lại toàn bộ module Đăng nhập:
 - Sau khi sửa lỗi, thực hiện kiểm thử lại module này để đảm bảo không còn phát sinh thêm lỗi.
- Tiếp tục bổ sung test cases:
 - Xem xét mở rộng các kịch bản kiểm thử phức tạp hơn (negative cases, stress testing, v.v.) để nâng cao độ tin cậy của hệ thống.

KẾT LUẬN

Kết luận:

Sau thời gian tìm hiểu và ứng dụng kiểm thử tự động em nhận thấy những vấn đề đã đạt được:

- Kiểm thử phần mềm rất quan trọng trong việc đảm bảo chất lượng của phần mềm. Việc nghiên cứu lựa chọn các kỹ thuật và chiến lược kiểm thử nhằm giúp việc thực hiện kiểm thử hiệu quả, giảm chi phí và thời gian.
- Ứng dụng thành công để truyền dữ liệu đầu vào cho kiểm thử tự động chức năng đăng ký, thêm mới sinh viên trên nền tảng Web. Tạo được kịch bản kiểm thử (Testcases) và chạy được bản kế hoạch.
- Có thêm định hướng về ngành nghề kiểm thử phần mềm.

Hạn chế:

Tuy nhiên, thời gian có hạn và kinh nghiệm thực tế của bản thân em chưa nhiều nên đề tài không tránh khỏi những thiếu sót:

- Chưa nghiên cứu lập trình nâng cao với Selenium
- Chưa kiểm thử được về mặt giao diện, hiệu năng,...
- Chưa kiểm thử những chức năng có nhiều các loại control sử dụng nền tảng web.

Hướng phát triển đề tài:

- Những nhược điểm trên cũng là hướng phát triển tương lai của đề án.
- Có thêm định hướng về ngành nghề kiểm thử phần mềm.

Trong thời gian tới em sẽ tiếp tục và nghiên cứu sâu hơn về kiểm thử để kiểm thử tự động sử dụng Selenium hỗ trợ vào công việc tương lai sau này.

TÀI LIỆU THAM KHẢO

- [1] Hoàng Quang Huy (2016), Giáo trình kiểm thử phần mềm, Nhà xuất bản Thống kê, Hà Nội
- [2] Mocha – the fun, simple, flexible JavaScript test framework, <https://mochajs.org/>
- [3] Selenium, selenium.dev
- [4] WebDriverIO - Next-gen browser and mobile automation test framework for Node.js, <http://webdriver.io/>
- [5] Kiểm thử tự động là gì, <https://hocjavascript.net/kiem-thu/>