

HỌC VIỆN KỸ THUẬT MẬT MÃ  
KHOA AN TOÀN THÔNG TIN



ĐỒ ÁN MÔN HỌC  
**THỰC TẬP CƠ SỞ**

Đề tài:

**KIỂM THỬ TỰ ĐỘNG CHO MÔ HÌNH  
WEBSITE**

*Nhóm sinh viên thực hiện:* Vũ Tiến Đạt AT170609

Đặng Xuân Đức AT170612

Hoàng Hữu Ánh AT170604

Nhóm 11

*Giảng viên hướng dẫn:* ThS. Lê Đức Thuận

Hà Nội, 19-04-2023

HỌC VIỆN KỸ THUẬT MẬT MÃ  
KHOA AN TOÀN THÔNG TIN



ĐỒ ÁN MÔN HỌC  
**THỰC TẬP CƠ SỞ**

**Đề tài:**

**KIỂM THỬ TỰ ĐỘNG CHO MÔ HÌNH  
WEBSITE**

**Nhóm sinh viên thực hiện:** Vũ Tiến Đạt AT170609

Đặng Xuân Đức AT170612

Hoàng Hữu Ánh AT170604

Nhóm 11

**Giảng viên hướng dẫn:** ThS. Lê Đức Thuận

Hà Nội, 19-04-2023

# MỤC LỤC

|  |           |
|--|-----------|
| <b>MỤC LỤC .....</b>   | <b>1</b>  |
| <b>LỜI CẢM ƠN .....</b>  | <b>3</b>  |
| <b>LỜI MỞ ĐẦU .....</b>  | <b>4</b>  |
| <b>DANH MỤC TỪ VIẾT TẮT .....</b>                                  | <b>6</b>  |
| <b>DANH MỤC HÌNH ẢNH .....</b>                                     | <b>7</b>  |
| <b>CHƯƠNG 1. TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM .....</b>              | <b>10</b> |
| 1.1 Tổng quan về kiểm thử phần mềm.....                            | 10        |
| 1.1.1 Kiểm thử phần mềm .....                                      | 10        |
| 1.1.2 Phân loại và các kỹ thuật kiểm thử.....                      | 10        |
| 1.1.3 Mục đích của kiểm thử phần mềm .....                         | 11        |
| 1.1.4 Vai trò của Kiểm thử phần mềm .....                          | 11        |
| 1.2 Quy trình kiểm thử phần mềm .....                              | 12        |
| 1.2.1 Phân tích yêu cầu (Requirement analysis) .....               | 12        |
| 1.2.2 Lập kế hoạch (Test planning) .....                           | 13        |
| 1.2.3 Thiết kế kiểm thử (Test case development) .....              | 13        |
| 1.2.4 Thiết lập môi trường kiểm thử (Test environment set up)..... | 14        |
| 1.2.5 Thực thi (Test execution) .....                              | 15        |
| 1.2.6 Kết thúc (Test cycle closure).....                           | 15        |
| 1.3 Các mức kiểm thử .....   | 16        |
| 1.3.1 Kiểm tra mức đơn vị (Unit Test) .....                        | 17        |
| 1.3.2 Kiểm tra tích hợp (Integration Test) .....                   | 17        |
| 1.3.3 Kiểm tra hệ thống (System Test) .....                        | 18        |
| 1.3.4 Kiểm tra chấp nhận (Acceptance Test) .....                   | 19        |
| 1.4 Các kỹ thuật kiểm thử .....                                    | 20        |
| 1.4.1 Kỹ thuật kiểm thử tĩnh (Static Testing) .....                | 20        |
| 1.4.2 Kỹ thuật kiểm thử động (Dynamic Testing) .....               | 21        |
| 1.4.3 Kiểm thử hộp đen.....  | 22        |
| 1.4.4 Kiểm thử hộp trắng.....                                      | 23        |
| 1.4.5 Kiểm thử hộp xám.....  | 25        |
| 1.5 Tổng kết chương 1 .....  | 26        |
| <b>CHƯƠNG 2. KIỂM THỬ TỰ ĐỘNG.....</b>                             | <b>27</b> |
| 2.1 Tổng quan về kiểm thử tự động .....                            | 27        |
| 2.1.1 Khái niệm kiểm thử tự động.....                              | 27        |
| 2.1.2 Quy trình kiểm thử tự động.....                              | 27        |
| 2.1.3 Ưu và nhược điểm của kiểm thử tự động.....                   | 27        |
| 2.1.4 Các trường hợp nên áp dụng kiểm thử tự động.....             | 28        |
| 2.2 Kiểm thử website .....   | 29        |

|   |           |
|---|-----------|
| 2.2.1 Khái niệm kiểm thử website .....                    | 29        |
| 2.2.2 Các kỹ thuật kiểm thử Website.....                  | 30        |
| 2.3 Một số công cụ hỗ trợ kiểm thử website phổ biến ..... | 38        |
| 2.3.1 Tổng quan về Selenium .....                         | 38        |
| 2.3.2 Tổng quan về Serenity BDD và Cucumber .....         | 46        |
| 2.3.3 Tổng quan về Jmeter.....                            | 49        |
| 2.3.4 Tổng quan về Burp Suite.....                        | 51        |
| 2.4 Tổng kết chương 2 .....                               | 53        |
| <b>CHƯƠNG 3. THỰC NGHIỆM.....</b>                         | <b>54</b> |
| 3.1 Bài toán .....  | 54        |
| 3.2 Cấu trúc của 1 project Automation test.....           | 55        |
| 3.3 Xác định các ca kiểm thử .....                        | 56        |
| 3.3.1 Chức năng đăng ký .....                             | 56        |
| 3.3.2 Chức năng đăng nhập.....                            | 56        |
| 3.4 Kiểm thử Facebook .....                               | 56        |
| 3.4.1 Xác định các bước kiểm thử.....                     | 58        |
| 3.4.2 Thực nghiệm kiểm thử tự động .....                  | 63        |
| 3.5 Kiểm thử Web tự tạo .....                             | 68        |
| 3.5.1 Xác định các bước kiểm thử.....                     | 68        |
| 3.5.2 Thực nghiệm kiểm thử tự động .....                  | 75        |
| 3.6 Kiểm thử một số lỗ hổng web thông dụng .....          | 79        |
| 3.6.1 Lỗ hổng an ninh ứng dụng web.....                   | 79        |
| 3.6.2 Lỗ hổng SQL Injection .....                         | 80        |
| 3.6.3 Lỗ hổng an ninh XSS.....                            | 82        |
| 3.7 Tổng kết chương 3 .....                               | 84        |
| <b>KẾT LUẬN .....</b>                                     | <b>85</b> |
| <b>III.TÀI LIỆU THAM KHẢO.....</b>                        | <b>87</b> |

## **LỜI CẢM ƠN**

Nhóm chúng em xin gửi lời cảm ơn chân thành tới các thầy giáo, cô giáo trong Khoa Công nghệ thông tin đã tạo điều kiện thuận lợi cho nhóm em trong quá trình thực hiện đề tài.

Chúng em xin gửi lời cảm ơn đặc biệt đến Thạc sĩ thầy: Lê Đức Thuận đã nhiệt tình hướng dẫn và chỉ bảo nhóm trong suốt thời gian thực hiện đề tài, giúp chúng em có thể hoàn thành tốt đề tài chuyên đề cơ sở lần này.

**GIẢNG VIÊN HƯỚNG DẪN**

ThS. Lê Đức Thuận

**NHÓM SINH VIÊN THỰC HIỆN**

Vũ Tiến Đạt

Đặng Xuân Đức

Hoàng Hữu Ánh

# LỜI MỞ ĐẦU

## 1. Tính cấp thiết của đề tài.

Ngày nay tự động hóa được nghiên cứu và ứng dụng ở rất nhiều lĩnh vực, trong đó công nghệ phần mềm nói chung và kiểm thử phần mềm nói riêng cũng không ngoại lệ. Để tạo ra một sản phẩm công nghệ thông tin hay phần mềm có chất lượng thì hoạt động kiểm thử đóng một vai trò vô cùng quan trọng. Khi đó, quá trình kiểm thử truyền thống, thủ công tiêu tốn một lượng lớn thời gian, kinh phí và nhân lực trong dự án. Do vậy nhu cầu tự động hóa quy trình kiểm thử đã được đặt ra.

Thực tế cho thấy việc áp dụng kiểm thử tự động hợp lý sẽ mang lại nhiều thành công cho hoạt động kiểm thử phần mềm, ứng dụng. Kiểm thử tự động sẽ giúp giảm bớt công sức thực hiện, tăng độ tin cậy, giảm sự nhầm lẫn và rèn luyện được tốt kỹ năng lập trình cho nhân viên kiểm thử.

Với mong muốn có cái nhìn xác thực, rõ ràng hơn về kiểm thử ứng dụng Web và nghiên cứu sâu hơn kiểm thử tự động Selenium nhóm chúng em đã thống nhất và thực hiện đề tài “KIỂM THỬ TỰ ĐỘNG CHO MÔ HÌNH WEBSITE” làm đề bài báo cáo của nhóm mình.

## 2. Mục tiêu nghiên cứu của đề tài

Nghiên cứu kiểm thử tự động cho mô hình website.

## 3. Đối tượng và phạm vi nghiên cứu

Đối tượng: các ứng dụng web hoặc trang web

Phạm vi: Nghiên cứu các phương pháp, công cụ và kỹ thuật để thiết kế, triển khai và thực thi các kịch bản kiểm thử tự động

## 4. Các nhiệm vụ chính cần thực hiện

Nội dung nghiên cứu được tập trung vào các nội dung chính như sau:

- Khảo sát, tổng hợp kiến thức nền tảng về website
- Thiết kế và triển khai các kịch bản kiểm thử tự động
- Thực thi các kịch bản kiểm thử tự động
- Đánh giá và báo cáo kết quả kiểm thử

- Cài đặt, thử nghiệm, đánh giá

## **5. Kết quả dự kiến**

### **Lý thuyết:**

- Trình bày tổng quan về phần mềm, kiểm thử phần mềm, đưa ra được tiêu chí đánh giá một phần mềm tốt, tìm hiểu quy trình kiểm thử phần mềm, đưa ra được các mức kiểm thử phần mềm, các kỹ thuật trong kiểm thử phần mềm, các chiến lược trong kiểm thử cũng như nêu ra được các kỹ thuật kiểm thử của hộp đen.

- Tổng quan về kiểm thử tự động, một số công cụ hỗ trợ kiểm thử tự động.
- Tổng quan về một số lỗ hổng Web: SQLi và XSS.

### **Thực nghiệm:**

- Áp dụng kiến thức đã tìm hiểu để kiểm thử chức năng của ứng dụng Web.
- Viết script để kiểm thử tự động chức năng cơ bản của ứng dụng Web.
- Kiểm thử tự động SQLi và XSS

## DANH MỤC TỪ VIẾT TẮT

|         |  |
|---------|--|
| API     | Application Programming Interface  |
| CAPTCHA | Completely Automated Public Turing test to tell Computers and Humans Apart |
| CSS     | Cascading Style Sheets   |
| DOM     | Document Object Model  |
| HTML    | Hyper Text Markup Language   |
| SQLi    | Structured Query Language injection  |
| XSS     | Cross-site Scripting   |



## DANH MỤC HÌNH ẢNH

|  |    |
|--|----|
| Hình 1.1: Quy trình kiểm thử phần mềm                            | 12 |
| Hình 1.2: Các mức kiểm thử                                       | 16 |
| Hình 1.3: Các kỹ thuật kiểm thử                                  | 20 |
| Hình 1.4: Kiểm thử hộp đen                                       | 22 |
| Hình 1.5: Kiểm thử hộp trắng                                     | 23 |
| Hình 1.6: Kiểm thử hộp xám                                       | 25 |
| Hình 2.1: Kiểm thử khả năng sử dụng trên website                 | 32 |
| Hình 2.2: Kiểm thử sự tương thích                                | 33 |
| Hình 2.3: Kiểm thử website cần độ chính xác từ cơ sở dữ liệu     | 34 |
| Hình 2.4: Giao diện chiếm vai trò lớn với trong kiểm thử website | 35 |
| Hình 2.5: Kiểm thử hiệu năng có ảnh hưởng lớn đối với người dùng | 36 |
| Hình 2.6: Kiểm thử bảo mật website                               | 37 |
| Hình 2.7: Các phiên bản Selenium                                 | 38 |
| Hình 2.8: Các hàm phổ biến trong Selenium                        | 39 |
| Hình 2.9: Cách lấy Xpath dạng cơ bản                             | 40 |
| Hình 2.10: Ví dụ về cách lấy Xpath dạng cơ bản                   | 40 |
| Hình 2.11: Ví dụ về cách lấy Xpath dùng contains                 | 41 |
| Hình 2.12: Ví dụ về lấy Xpath dùng toán tử OR                    | 41 |
| Hình 2.13: Ví dụ về lấy Xpath dùng toán tử AND                   | 42 |
| Hình 2.14: Ví dụ về lấy Xpath dùng Start-with.                   | 42 |
| Hình 2.15: Ví dụ về cách lấy Xpath dùng text()                   | 43 |
| Hình 2.16: Ví dụ về lấy Xpath dùng following                     | 43 |
| Hình 2.17: Ví dụ về cách lấy Xpath dùng Ancestor                 | 44 |
| Hình 2.18: Ví dụ về cách lấy Xpath dùng Child.                   | 44 |
| Hình 2.19: Ví dụ về cách lấy Xpath dùng Following-sibling        | 45 |
| Hình 2.20: Ví dụ về cách lấy Xpath dùng Parent                   | 45 |
| Hình 2.21: Ví dụ về cách lấy Xpath dùng Self                     | 46 |
| Hình 2.22: Mô hình BDD   | 47 |
| Hình 2.23: Quy trình hoạt động của hệ thống Jmeter               | 51 |
| Hình 3.1 Cấu trúc của 1 project Automation                       | 55 |
| Hình 3.2: Giao diện Facebook                                     | 58 |
| Hình 3.3: Màn hình trang đăng ký                                 | 58 |
| Hình 3.4: Màn hình email không hợp lệ                            | 59 |
| Hình 3.5: Màn hình email không trùng khớp                        | 59 |

|   |    |
|---|----|
| Hình 3.6 Màn hình email trống                                   | 59 |
| Hình 3.7 Màn hình password trống                                | 59 |
| Hình 3.8 Màn hình email đã được sử dụng                         | 60 |
| Hình 3.9 Màn hình facebook                                      | 60 |
| Hình 3.10 Màn hình lỗi Email sai                                | 61 |
| Hình 3.11 Màn hình lỗi mật khẩu sai                             | 61 |
| Hình 3.12 Màn hình lỗi Email trống                              | 62 |
| Hình 3.13 Màn hình lỗi mật khẩu trống                           | 62 |
| Hình 3.14 Màn hình đăng nhập thành công                         | 63 |
| Hình 3.15 Scenario Outline đăng ký sai                          | 63 |
| Hình 3.16 Scenario Outline đăng ký Email đã sử dụng             | 63 |
| Hình 3.17 Trường hợp Email không hợp lệ                         | 64 |
| Hình 3.18 Trường hợp Email nhập lại không trùng khớp            | 64 |
| Hình 3.19 Trường hợp Email trống                                | 64 |
| Hình 3.20: Trường hợp mật khẩu trống                            | 65 |
| Hình 3.21 Trường hợp Email đã được sử dụng                      | 65 |
| Hình 3.22 Scenario Outline đăng nhập sai                        | 65 |
| Hình 3.23 Scenario Outline đăng nhập đúng                       | 66 |
| Hình 3.24 Trường hợp mật khẩu sai                               | 66 |
| Hình 3.25 Trường hợp Email sai                                  | 66 |
| Hình 3.26 Trường hợp Email trống                                | 67 |
| Hình 3.27 Trường hợp mật khẩu trống                             | 67 |
| Hình 3.28 Trường hợp đăng nhập đúng                             | 67 |
| Hình 3.29 Giao diện trang web                                   | 68 |
| Hình 3.30 Giao diện đăng ký                                     | 68 |
| Hình 3.31 Giao diện đăng ký với Email không hợp lệ              | 69 |
| Hình 3.32 Giao diện đăng ký với Email nhập lại không trùng khớp | 69 |
| Hình 3.33 Giao diện đăng ký với Email trống                     | 70 |
| Hình 3.34 Giao diện đăng ký với mật khẩu trống                  | 70 |
| Hình 3.35 Giao diện đăng ký với mật khẩu không đạt yêu cầu      | 71 |
| Hình 3.36 Giao diện đăng ký với Email đã được sử dụng           | 71 |
| Hình 3.37 Giao diện trang Web                                   | 72 |
| Hình 3.38 Giao diện đăng nhập với mật khẩu sai                  | 72 |
| Hình 3.39 Giao diện đăng nhập với Email không hợp lệ            | 73 |
| Hình 3.40 Giao diện đăng nhập với mật khẩu trống                | 73 |
| Hình 3.41 Giao diện với Email trống                             | 74 |
| Hình 3.42 Giao diện khi đăng nhập thành công                    | 74 |

|  |    |
|--|----|
| Hình 3.43 Scenario Outline                           | 75 |
| Hình 3.44 Trường hợp Email không hợp lệ              | 75 |
| Hình 3.45 Trường hợp Email nhập lại không trùng khớp | 75 |
| Hình 3.46 Trường hợp Email trống                     | 76 |
| Hình 3.47 Trường hợp mật khẩu trống                  | 76 |
| Hình 3.48 Trường hợp mật khẩu không hợp lệ           | 76 |
| Hình 3.49 Trường hợp Email đã được sử dụng           | 77 |
| Hình 3.50 Scenario Outline                           | 77 |
| Hình 3.51 Trường hợp mật khẩu sai                    | 77 |
| Hình 3.52 Trường hợp Email sai                       | 78 |
| Hình 3.53 Trường hợp mật khẩu trống                  | 78 |
| Hình 3.54 Trường hợp Email trống                     | 78 |
| Hình 3.55 Trường hợp đăng nhập thành công            | 79 |
| Hình 3.56: Sơ đồ minh họa tấn công SQLi              | 80 |
| Hình 3.57 Scenario Outline                           | 82 |
| Hình 3.58 Demo thành công                            | 82 |
| Hình 3.59 Scenario Outline                           | 83 |
| Hình 3.60 Demo thành công                            | 83 |

# CHƯƠNG 1. TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM

## 1.1 Tổng quan về kiểm thử phần mềm

### 1.1.1 Kiểm thử phần mềm

Kiểm thử phần mềm (Software Testing) là quy trình sử dụng để đánh giá, kiểm tra chất lượng phần mềm ở nhiều khía cạnh khác nhau dựa trên các yêu cầu của người sử dụng đối với sản phẩm.

Tiêu chuẩn để một phần mềm, ứng dụng đáp ứng được yêu cầu của khách hàng là:

- Đáp ứng được các yêu cầu đặt ra trước đó của nhà phát triển.
- Đáp ứng được mọi yêu cầu đầu vào trước đó.
- Thực hiện các chức năng trong thời gian cho phép.
- Cài đặt và chạy được trong môi trường thật.
- Đạt được các mục tiêu đề ra của các bên liên quan từ trước.

### 1.1.2 Phân loại và các kỹ thuật kiểm thử

Phân loại kiểm thử dựa vào các yếu tố: Chiến lược kiểm thử, phương pháp kiểm thử, kỹ thuật kiểm thử.

- Dựa vào chiến lược kiểm thử thì ta có thể chia kiểm thử thành hai loại:
  - +Kiểm thử thủ công.
  - +Kiểm thử tự động.
- Dựa vào phương pháp tiến hành kiểm thử, ta chia làm 2 loại:
  - +Kiểm thử tĩnh.
  - +Kiểm thử động.
- Dựa vào kỹ thuật kiểm thử ta chia kiểm thử thành 3 loại:
  - +Kiểm thử hộp đen.
  - +Kiểm thử hộp trắng.
  - +Kiểm thử hộp xám.

### **1.1.3 Mục đích của kiểm thử phần mềm**

Kiểm thử để tìm ra lỗi càng sớm càng tốt giúp tiết kiệm cho chi phí, thời gian, công sức cho việc sửa chữa phần mềm cũng như đưa ra các hướng khắc phục lỗi sao cho khi tung sản phẩm ra thị trường hoặc khách hàng sẽ đảm bảo rằng sản phẩm là tốt nhất.

Ngăn chặn các lỗi trong sản phẩm, dự án.

Kiểm tra xem tiêu chí, yêu cầu của khách hàng có được đáp ứng hay không.

Đảm bảo kết quả cuối cùng sẽ đáp ứng được yêu cầu của các bên liên quan.

Thể hiện sự uy tín đối với khách hàng bằng một sản phẩm chất lượng.

### **1.1.4 Vai trò của Kiểm thử phần mềm**

Kiểm thử phần mềm đóng vai trò quan trọng trong việc đánh giá, kiểm định và thu được chất lượng cao của sản phẩm phần mềm trong quá trình phát triển phần mềm. Kiểm thử sẽ giúp:

- Chỉ ra được các lỗi sai và sai sót trong quá trình phát triển phần mềm để đưa ra được một phần mềm hoàn hảo nhất.

- Có thể đánh giá chất lượng sản phẩm phần mềm dựa vào số lượng lỗi ta tìm thấy và một số đặc tính sau: tính đúng đắn, thời gian phản hồi, tính dễ sử dụng, tính bảo trì, ....

- Đảm bảo độ tin cậy của khách hàng, sự hài lòng của khách hàng đối với sản phẩm phần mềm.

- Đảm bảo chất lượng của sản phẩm.

- Là hoạt động cần thiết để tăng hiệu quả của ứng dụng.

- Đảm bảo rằng các ứng dụng không có bất kỳ kết quả nào thất bại hay lỗi.

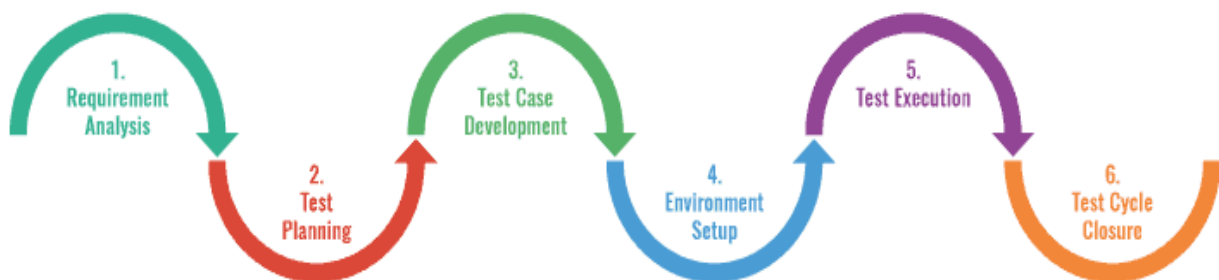
- Là một hoạt động thiết yếu giúp sản phẩm phát triển khi tung sản phẩm ra thị trường.

Kiểm thử đem lại sự tin tưởng tương đối với chất lượng sản phẩm phần mềm khi sản phẩm có ít lỗi hoặc không có lỗi nào được tìm thấy. Nếu lỗi tìm thấy và được sửa thì chất lượng sản phẩm phần mềm càng được tăng cao và giảm được chi phí cũng như thời gian trong quá trình phát triển, nâng cấp và bảo trì phần mềm,...

## 1.2 Quy trình kiểm thử phần mềm

Quy trình kiểm thử phần mềm (Software Testing Life Cycle) xác định các giai đoạn, các pha trong kiểm thử phần mềm. Tuy nhiên thì Quy trình kiểm thử phần mềm không có tiêu chuẩn cố định nào, nhưng về cơ bản thì Quy trình kiểm thử bao gồm các giai đoạn sau:

### Software Testing Life Cycle (STLC)



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

Các giai đoạn kiểm thử được thực hiện một cách tuần tự. Mỗi giai đoạn sẽ có những mục tiêu khác nhau, đầu vào và kết quả đầu ra khác nhau nhưng mục đích cuối cùng vẫn là đảm bảo chất lượng sản phẩm phần mềm tốt nhất.

### 1.2.1 Phân tích yêu cầu (*Requirement analysis*)

#### a) Đầu vào

Thu thập yêu cầu của khách hàng thành một bản đặc tả cho phần mềm, phần mềm có những chức năng chính nào, các tiêu chí của khách đối với đầu ra của phần mềm như nào, tài liệu thiết kế cho hệ thống.

#### b) Hoạt động

Thu thập các yêu cầu của khách hàng đối với phần mềm.

Xác định rõ được những chức năng chính của hệ thống là gì, phân tích chi tiết các yêu cầu của khách hàng.

Trong quá trình phân tích và xây dựng các chức năng chính của phần mềm, nếu có gì không hiểu sẽ liên hệ lại khách hàng để nắm rõ được chức năng của phần mềm.

#### c) Đầu ra

Đưa ra tài liệu chứa các câu hỏi và câu trả lời liên quan đến nghiệp vụ của hệ thống, tài liệu báo cáo tính khả thi, phân tích rủi ro của việc kiểm thử phần mềm.

### **1.2.2 Lập kế hoạch (Test planning)**

#### **a) Đầu vào**

Là các tài liệu liên quan đến nghiệp vụ của phần mềm, tài liệu báo cáo tính khả thi, phân tích rủi ro đã được phân tích ở giai đoạn phân tích yêu cầu.

#### **b) Hoạt động**

Xác định phạm vi của dự án: xác định xem thời gian để hoàn thành dự án trong bao lâu, bao gồm những công việc gì cần làm cho từng khoảng thời gian cụ thể để đưa ra được lịch trình thực hiện cho từng công việc nhỏ sao cho phù hợp với toàn bộ đội dự án.

Xác định phương pháp tiếp cận: xác định xem yêu cầu của khách hàng có khắt khe về sản phẩm, dùng kỹ thuật để phát triển phần mềm là gì, lĩnh vực của sản phẩm là lĩnh vực gì.

Đưa ra những phương pháp và kế hoạch phù hợp nhất cho cả quá trình thực hiện kiểm thử sản phẩm.

Xác định các nguồn lực của sản phẩm: cần bao nhiêu người tham gia dự án, ai sẽ test những phần nào, số lượng tester tham gia vào sản phẩm, tester có biết về lĩnh vực này không. Số lượng thiết bị: server, vision, máy tính, mobile để thực hiện test là bao nhiêu.

Lên kế hoạch cho công việc test: liệt kê các chức năng cần kiểm thử, để test chức năng đó cần những công việc gì và thực hiện test trong bao lâu, chức năng nào thực hiện trước, chức năng nào thực hiện sau, ai là người thực hiện test. Từ đó đưa ra được điều kiện để kết thúc kiểm thử một chức năng.

#### **c) Đầu ra**

Đưa ra được bản kế hoạch kiểm thử, các dự toán về chi phí, con người, tài nguyên cần cho dự án và đưa ra được tiến độ của dự án.

### **1.2.3 Thiết kế kiểm thử (Test case development)**

#### **a) Đầu vào**

Là các bản kế hoạch kiểm thử, các dự toán về chi phí, con người, tài nguyên cần cho dự án và đưa ra được tiến độ của dự án, các tài liệu đặc tả ban đầu.

#### b) Hoạt động

Đọc tài liệu: cần đọc lại tất cả các tài liệu để xác định rõ xem cần phải kiểm thử các chức năng với các luồng của nó là gì để đưa ra kịch bản kiểm thử và đầy đủ nhất.

Thiết kế các testcase/ checklist: tester bắt tay vào việc viết test case chi tiết dựa vào kế hoạch đã đưa ra và vận dụng các kỹ thuật thiết kế kịch bản kiểm thử. Test case cần bao phủ được tất cả các trường hợp kiểm thử có thể xảy ra cũng như đáp ứng đầy đủ các tiêu chí của sản phẩm. Đồng thời tester cũng cần đánh giá mức độ ưu tiên cho từng test case.

Chuẩn bị dữ liệu kiểm thử: Cùng với việc tạo ra các test case chi tiết, đội kiểm thử cũng cần chuẩn bị trước các dữ liệu kiểm thử cho các trường hợp cần thiết như test data, test script.

Review testcase/ checklist: Sau khi hoàn thành, các thành viên trong đội kiểm thử hoặc test leader cũng cần review lại test case đã tạo để có thể bổ sung, hỗ trợ lẫn nhau nhằm tránh những sai sót trong thiết kế test case và rủi ro về sau.

#### c) Đầu ra

Sau khi hoàn thành thiết kế kịch bản kiểm thử, đội kiểm thử sẽ có các tài liệu bao gồm: test design, test case, check list, test data, test automation script.

### ***1.2.4 Thiết lập môi trường kiểm thử (Test environment set up)***

#### a) Đầu vào

Đầu vào của giai đoạn cài đặt môi trường kiểm thử là test plan, smoke test case, test data.

#### b) Hoạt động

Việc cài đặt môi trường kiểm thử là giai đoạn cũng rất quan trọng trong vòng đời phát triển phần mềm. Môi trường kiểm thử sẽ được quyết định dựa trên những yêu cầu của khách hàng, hay đặc thù của sản phẩm ví dụ như server/ client/ network,...

Tester cần chuẩn bị một vài test case để kiểm tra xem môi trường cài đặt đã sẵn sàng cho việc kiểm thử hay chưa. Đây chính là việc thực thi các smoke test case.



c) Đầu ra

Đầu ra của giai đoạn này là môi trường đã được cài đặt đúng theo yêu cầu, sẵn sàng cho việc kiểm thử và kết quả của smoke test case.

### **1.2.5 Thực thi (Test execution)**

a) Đầu vào

Tài liệu đầu vào của giai đoạn này là test plan, test design, test case, check list, test data, test automation script.

b) Hoạt động

Thực hiện các test case như thiết kế và mức độ ưu tiên đã đưa ra trên môi trường đã được cài đặt.

So sánh với kết quả mong đợi sau báo cáo các bug xảy ra lên tool quản lý lỗi và theo dõi trạng thái của lỗi đến khi được sửa thành công.

Thực hiện re-test để verify các bug đã được fix và regression test khi có sự thay đổi liên quan.

Trong quá trình thực hiện kiểm thử, kiểm thử viên cũng có thể hỗ trợ, đề xuất cho cả đội dự án để có giải pháp hợp lý và kết hợp công việc hiệu quả.

Đo và phân tích tiến độ: kiểm thử viên cũng cần kiểm soát chặt chẽ tiến độ công việc của mình bằng cách so sánh tiến độ thực tế với kế hoạch, nếu chậm cần phải điều chỉnh sao cho kịp tiến độ dự án, nếu nhanh cũng cần điều chỉnh vì có thể test lead lên kế hoạch chưa sát với thực tế dự án. Từ đó có thể sửa chữa test plan để phù hợp với tiến độ dự án đưa ra.

Report thường xuyên cho Project Management/ Project Manager và khách hàng về tình hình thực hiện dự án: Cung cấp thông tin trong quá trình kiểm thử đã làm được những chức năng nào, còn chức năng nào, hoàn thành được bao nhiêu phần trăm công việc, báo cáo các trường hợp phát sinh sớm, tránh ảnh hưởng tiến độ công việc của cả ngày.

c) Đầu ra

Đầu ra của giai đoạn này là test results (kết quả kiểm thử), defect reports ( danh sách các lỗi tìm được).

### **1.2.6 Kết thúc (Test cycle closure)**

a) Đầu vào

Đầu vào của giai đoạn đóng chu trình kiểm thử là bao gồm tất cả những tài liệu liên quan đã được tổng hợp, ghi chép và hoàn thiện đầy đủ trong suốt quy trình kiểm thử của dự án: tài liệu phân tích đặc tả yêu cầu, test plan, test results, defect reports, tài liệu Question & Answers,...

#### b) Hoạt động

Đây là giai đoạn cuối cùng trong quy trình kiểm thử phần mềm.

Ở giai đoạn này, Quality Assurance team thực hiện tổng kết, báo cáo kết quả về việc thực thi test case, bao nhiêu case pass/ fail, bao nhiêu case đã được fix, mức độ nghiêm trọng của lỗi, bao nhiêu lỗi cao/ thấp, lỗi còn nhiều ở chức năng nào, dev nào nhiều lỗi. Chức năng nào đã hoàn thành test/ chưa hoàn thành test/ trễ tiến độ bàn giao.

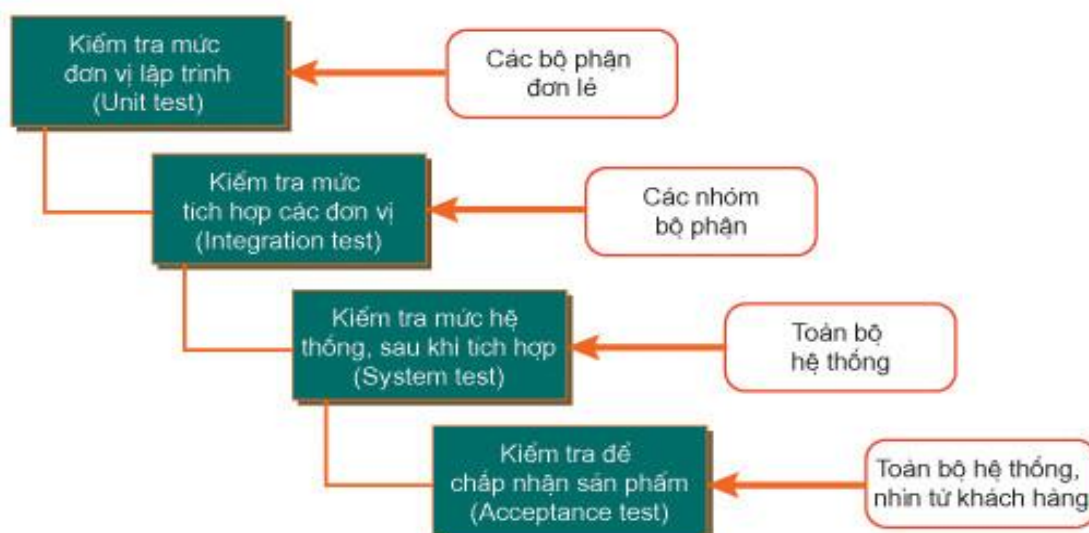
Đánh giá các tiêu chí hoàn thành như phạm vi kiểm tra, chất lượng, chi phí, thời gian, mục tiêu kinh doanh quan trọng.

Ngoài ra, giai đoạn này cũng thảo luận tất cả những điểm tốt, điểm chưa tốt và rút ra bài học kinh nghiệm cho những dự án sau, giúp cải thiện quy trình kiểm thử.

#### c) Đầu ra

Đầu ra của giai đoạn này bao gồm các tài liệu: Test report, Test results (final).

### 1.3 Các mức kiểm thử



Hình 1.2: Các mức kiểm thử

### **1.3.1 Kiểm tra mức đơn vị (Unit Test)**

Kiểm tra mức đơn vị (Unit Testing) hay còn là kiểm tra các thành phần, đề cập đến kiểm tra chức năng của một phần cụ thể của mã, thường ở mức chức năng.

Unit Testing được thực hiện trên một phần nhỏ hoặc một đơn vị code ( ví dụ như: một hàm, một class, thủ tục, phương thức...)

Các tool để test Unit: Nunit, Test Drivent.Net, ...

Cần hiểu biết về thiết kế chương trình và code.

Thực hiện bởi Lập trình viên (không phải kiểm thử viên).

Sử dụng phương pháp: Kiểm thử hộp trắng.

Mục đích của việc kiểm tra Unit test:

Tách riêng từng phần để kiểm tra và chứng minh các thành phần đó thực hiện chính xác các yêu cầu chức năng trong đặc tả.

Lỗi được sửa sớm trong chu trình phát triển phần mềm vì vậy tiết kiệm thời gian và chi phí sửa lỗi.

Mã nguồn được tái sử dụng nhiều hơn.

Tăng sự tin tưởng trong việc thay đổi hoặc bảo trì.

Mã nguồn đáng tin cậy hơn.

### **1.3.2 Kiểm tra tích hợp (Integration Test)**

Kiểm tra tích hợp (Integration Testing) là kiểm tra liệu tất cả các module được kết hợp hoặc chưa kết hợp lại cùng với nhau thực hiện hiện công việc có đạt được kết quả như tài liệu đặc tả đưa ra hay không( do mỗi lập trình viên thực hiện trên các module khác nhau. Khi mà họ hoàn thành đoạn code của họ, nhóm quản lý cấu hình ráp lại với nhau và chuẩn bị chạy chương trình. Các tester cần chắc rằng các module này bây giờ đã được kết hợp và làm việc theo như yêu cầu).

Thành phần có thể là: các module, các ứng dụng riêng lẻ, các ứng dụng client/server trên một mạng.

Integration Test có 2 mục tiêu chính:

Phát hiện lỗi giao tiếp xảy ra giữa các Unit.

Tích hợp các Unit đơn lẻ thành các hệ thống nhỏ (subsystem) và cuối cùng là nguyên hệ thống hoàn chỉnh (system) chuẩn bị cho kiểm tra ở mức hệ thống (System Test).

Có 4 loại kiểm tra trong Integration Test.

- Kiểm tra cấu trúc (structure): Tương tự White Box Test (kiểm tra nhằm bảo đảm các thành phần bên trong của một chương trình chạy đúng), chú trọng đến hoạt động của các thành phần cấu trúc nội tại của chương trình chẳng hạn các lệnh và nhánh bên trong.

- Kiểm tra chức năng (functional): Tương tự Black Box Test (kiểm tra chỉ chú trọng đến chức năng của chương trình, không quan tâm đến cấu trúc bên trong), chỉ khảo sát chức năng của chương trình theo yêu cầu kỹ thuật.

- Kiểm tra hiệu năng (performance): Kiểm tra việc vận hành của hệ thống.

- Kiểm tra khả năng chịu tải (stress): Kiểm tra các giới hạn của hệ thống.

Người thực hiện: Test level này thường là Tester thực hiện.

Trong Unit Test, lập trình viên cố gắng phát hiện lỗi liên quan đến chức năng và cấu trúc nội tại của Unit. Có một số phép kiểm tra đơn giản trên giao tiếp giữa Unit với các thành phần liên quan khác, tuy nhiên mọi giao tiếp liên quan đến Unit thật sự được kiểm tra đầy đủ khi các Unit tích hợp với nhau trong khi thực hiện Integration Test.

### **1.3.3 Kiểm tra hệ thống (System Test)**

Kiểm tra mức hệ thống (System Testing) là khi tester hoàn thành công việc test (các tester test ứng dụng trong các môi trường test, có nghĩa là tester test với dữ liệu test, không test trên dữ liệu thật, ứng dụng (phần mềm) phải được test trên môi trường thật).

Kể từ khi các tester test trong môi trường test với dữ liệu test, phải chắc chắn rằng ứng dụng làm việc tốt trong môi trường thật với dữ liệu thật.

Trong môi trường test thì một vài điều kiện không thể test hoặc thao tác giả được (ví dụ như máy tính cài nhiều phần mềm như máy tính của người dùng, không biết phần mềm này có xung đột với các ứng dụng, phần mềm khác không, truy nhập cũng không thể nhiều như khi ở môi trường thật). Tất cả sẽ khác nhau và cơ sở dữ liệu cũng khác nhau nên một số thao tác cũng không thể thực hiện được khi ứng dụng chuyển từ môi trường test sang môi trường thật.

Phân loại System test:

- *Kiểm thử chức năng (Functional Test)*: Là kiểm thử toàn bộ hệ thống, đảm bảo hệ thống hoạt động đúng theo yêu cầu được đưa ra trước đó.

- *Kiểm thử hiệu năng (Performance Test)*: Là kiểm tra sự tuân thủ của hệ thống với các yêu cầu được chỉ định về hiệu năng. Xác định những thuộc tính chất lượng của hệ thống như khả năng mở rộng, độ tin cậy...

- *Kiểm thử cơ sở dữ liệu (Database Test)*: Là kiểm tra dữ liệu hiển thị trên hệ thống có giống với dữ liệu trong cơ sở dữ liệu hay không?

- *Kiểm thử khả năng bảo mật (Security Test)*: Là kiểm tra hệ thống được bảo vệ an toàn, không bị đánh cắp dữ liệu, thông tin trước các tấn công từ bên ngoài.

- *Kiểm thử tính khả dụng (Usability Test)*: Kiểm tra tính thân thiện với người dùng và tính dễ sử dụng của hệ thống.

- *Kiểm tra tính tương thích (Compatibility Test)* : Là kiểm tra xem hệ thống có tương thích với các yếu tố khác của hệ thống mà nó sẽ hoạt động hay không? (Ví dụ: Trình duyệt, hệ điều hành, phần cứng).

- *Kiểm tra khả năng phục hồi (Recovery Test)*: Là kiểm tra hệ thống có khả năng khôi phục trạng thái ổn định khi gặp các sự cố bất thường không.

Sử dụng phương pháp: Kiểm thử hộp đen là phổ biến.

Người thực hiện: Test level này thường là Tester thực hiện.

Điểm khác nhau then chốt giữa Integration Test và System Test là System Test chú trọng các hành vi và lỗi trên toàn hệ thống, còn Integration Test chú trọng sự giao tiếp giữa các đơn thể hoặc đối tượng khi chúng làm việc cùng nhau. Thông thường ta phải thực hiện Unit Test và Integration Test để bảo đảm mọi Unit và sự tương tác giữa chúng hoạt động chính xác trước khi thực hiện System Test.

#### **1.3.4 Kiểm tra chấp nhận (Acceptance Test)**

Trong quá trình này, phần mềm sẽ được thực hiện kiểm tra từ người dùng để tìm ra nếu phần mềm đúng với sự mong đợi của người dùng và thực hiện chức năng đúng như mong đợi. Trong giai đoạn test này thì tester cũng có thể thực hiện hoặc khách hàng cũng có thể test.

Mục tiêu của kiểm thử này là để đánh giá sự tuân thủ của hệ thống với các yêu cầu nghiệp vụ và thẩm định xem đã có thể chấp nhận để bàn giao chưa.

Kiểm tra chấp nhận (Acceptance Testing) gồm 2 loại kiểm thử là:

- Alpha Test, người dùng kiểm thử phần mềm ngay tại nơi phát triển phần mềm, lập trình viên sẽ ghi nhận các lỗi hoặc phản hồi, và lên kế hoạch sửa chữa.

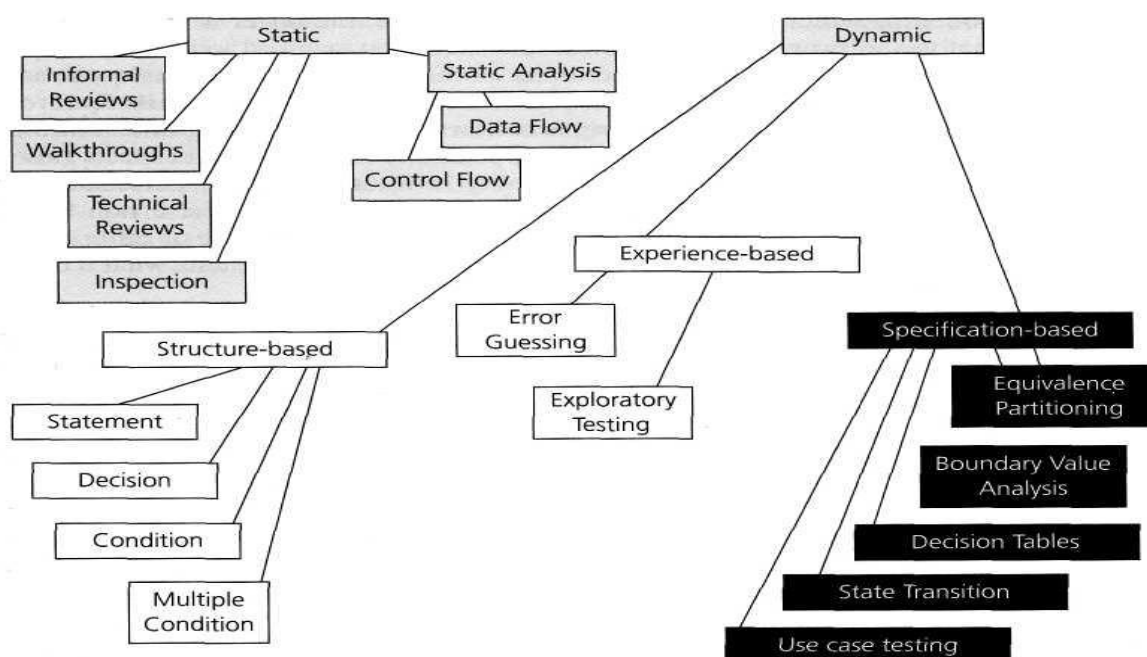
- Beta Test, phần mềm sẽ được gửi tới cho người dùng để kiểm thử ngay trong môi trường thực, lỗi hoặc phản hồi cũng sẽ gửi ngược lại cho lập trình viên để sửa chữa.

Sử dụng phương pháp: Kiểm thử hộp đen.

Người thực hiện: Test level này thường là khách hàng hoặc bên thứ ba.

=> Lưu ý không nhất thiết phải thực hiện tất cả các loại kiểm tra nêu trên. Tùy yêu cầu và đặc trưng của từng hệ thống, tùy khả năng và thời gian cho phép của dự án, khi lập kế hoạch, trưởng dự án sẽ quyết định áp dụng những loại kiểm tra nào.

## 1.4 Các kỹ thuật kiểm thử



Hình 1.3: Các kỹ thuật kiểm thử

### 1.4.1 Kỹ thuật kiểm thử tĩnh (Static Testing)

Kiểm thử tĩnh (Static Testing) là kiểm thử mà không thực thi mã nguồn hoặc không thực hiện chạy hệ thống phần mềm, kiểm tra, review các tài liệu đặc tả, tài liệu thiết kế, source code để tìm lỗi.

Mục tiêu chính của kiểm thử tĩnh là nâng cao chất lượng sản phẩm bằng việc tìm lỗi trong giai đoạn sớm của quy trình phát triển phần mềm.

Các loại Static Testing:

- *Informal review*: là quá trình đánh giá mà không cần hồ sơ cuộc họp, cũng không cần ghi chép lại nội dung cuộc họp. Informal review chủ yếu được thực hiện giữa 2 người ở bất kỳ đâu có thể là quán cà phê, căng-tin,... Phương pháp này không cần phải tuân theo chu trình gì, nên đem lại lợi ích khá nhanh chóng và không tốn kém chi phí.

- *Walkthroughs*: Đây là hình thức hướng dẫn, giải thích bởi người nắm rõ về logic phần mềm nhất, nhằm mục đích chuyển giao kiến thức tới những người tham gia vào chu trình test, qua đó mọi người sẽ có đạt được sự hiểu biết nhất định. Thường thì phương pháp này sẽ truyền tải qua một buổi họp, và có người ghi chép riêng.

- *Technical review*: Phương pháp này tập trung vào việc đánh giá kỹ thuật của phần mềm. Thường được dẫn dắt bởi moderator hoặc người có kiến thức kỹ thuật cùng với sự tham gia của các chuyên gia kỹ thuật. Đây là một cuộc thảo luận tập trung vào việc đạt được sự đồng thuận về nội dung kỹ thuật nhằm đưa ra quyết định, đánh giá sự thay thế, tìm kiếm lỗi, giải quyết vấn đề kỹ thuật...

- *Inspection*: Phương pháp này cũng được điều hành bởi moderator. Mục đích của nó là để xác định rõ vai trò của từng người trong quy trình cũng như các tiêu chuẩn đầu vào, đầu ra của phần mềm. Từ đó tìm kiếm lỗi cũng như tập hợp và phân tích để tối ưu hóa quy trình.

Kỹ thuật static có 2 phần:

- Review: diễn hình sử dụng tìm lỗi trong tài liệu như requirement, thiết kế, test case...

- Static analysis: code được viết bởi dev được phân tích (thường phân tích bằng tools).

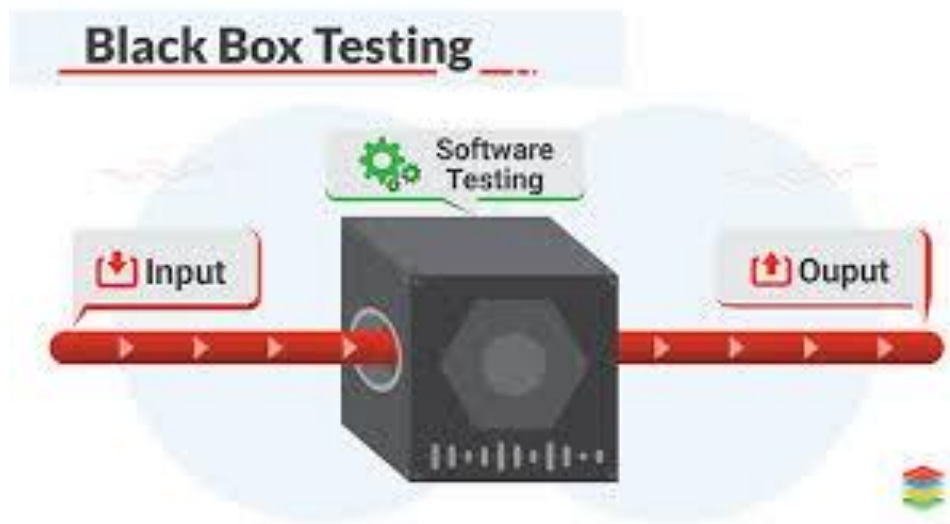
#### **1.4.2 Kỹ thuật kiểm thử động (Dynamic Testing)**

Kỹ thuật kiểm thử động (Dynamic Testing) được thực hiện khi code đang ở chế độ thực thi. Thử nghiệm động được thực hiện trong môi trường thực thi chạy chương trình ứng dụng. Khi code được thực thi, thì đầu vào được truyền một giá trị, kết quả hoặc đầu ra của việc thực hiện được so sánh với kết quả dự kiến ban đầu đã đưa ra. Với việc này chúng ta có thể quan sát được các hành vi chức năng của phần mềm, giám sát hệ thống bộ nhớ, thời gian phản hồi của CPU, hiệu suất của hệ thống. Kiểm thử dynamic còn được gọi là kiểm thử xác nhận (Validation testing), đánh giá sản phẩm.

Kiểm thử động gồm hai loại:

- Kiểm tra chức năng là quá trình kiểm tra xem phần mềm có thực hiện được các chức năng và hoạt động như mong đợi hay không.
- Kiểm tra phi chức năng, bao gồm các yêu cầu phi chức năng của phần mềm như tính bảo mật, khả năng sử dụng, hiệu suất, tương thích và độ tin cậy.

#### 1.4.3 Kiểm thử hộp đen



**Hình 1.4: Kiểm thử hộp đen**

Kiểm thử hộp đen (Black Box Testing) là một kỹ thuật để kiểm tra trong đó chức năng của ứng dụng được kiểm tra mà không cần xem cấu trúc mã bên trong, chi tiết triển khai và kiến thức về đường dẫn bên trong của phần mềm ra sao.

Chỉ quan tâm đầu vào và đầu ra mong muốn chứ không quan tâm đến code bên trong phần mềm.

Hộp đen ở trên có thể là bất kì hệ thống, phần mềm nào mà ta muốn kiểm tra.

##### 1.4.3.1 Các cách thực hiện kiểm thử hộp đen

Các yêu cầu và thông số kỹ thuật của phần mềm được kiểm tra.

Người kiểm thử chọn đầu vào hợp lệ để kiểm tra xem đầu ra có đúng như mong đợi hay không. Ngoài ra thì một số đầu vào không hợp lệ được chọn để xác nhận rằng hệ thống có thể phát hiện ra khi nhập sai.

Tester xác định đầu ra dự kiến cho tất cả các đầu vào.

Xây dựng các trường hợp kiểm thử với các đầu vào đã được lựa chọn ở bước lập kế hoạch.



Các trường hợp đưa ra đều được thử nghiệm.

Kiểm thử và so sánh các đầu ra thực tế với các điều kiện đã dự kiến trước đó.

Nếu có những sai sót gì sẽ được ghi lại và kiểm tra.

#### 1.4.3.2 Các loại thử nghiệm hộp đen

Kiểm thử chức năng (Functional testing) : thường liên quan đến các yêu cầu chức năng của hệ thống, phải đi theo đúng luồng trong phần đặc tả phần mềm.

Kiểm thử phi chức năng (Non-Functional testing) : không liên quan đến một chức năng cụ thể nào, nhưng các yêu cầu như hiệu suất, khả năng mở rộng, khả năng sử dụng của phần mềm.

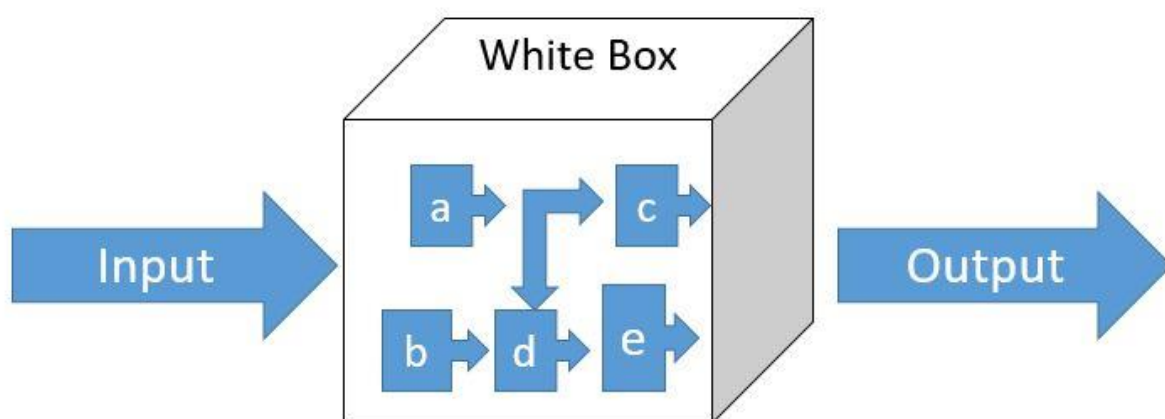
Kiểm thử hồi quy (Regression testing) : được thực hiện sau khi các lỗi đã được fix hoặc nâng cấp hoặc có bất kỳ bảo trì hệ thống nào khác để kiểm tra các chức năng mới không ảnh hưởng đến các chức năng hiện có.

#### 1.4.3.3 Các kỹ thuật kiểm thử hộp đen

Các kỹ thuật kiểm thử hộp đen phổ biến nhất:

- Phân vùng tương đương (Equivalence partitioning)
- Phân tích giá trị biên (Boundary value analysis)
- Bảng quyết định (Decision Tables)
- Đoán lỗi (Error guessing)

#### 1.4.4 Kiểm thử hộp trắng



**Hình 1.5: Kiểm thử hộp trắng**

Kiểm tra hộp trắng (White Box Testing - còn được gọi là thử nghiệm hộp rõ ràng, kiểm tra hộp kính, kiểm tra hộp trong suốt, thử nghiệm cấu trúc) là kiểm

tra cấu trúc nội bộ hoặc hoạt động của một ứng dụng và nó trái ngược với kiểm thử hộp đen.

Trong khi kiểm tra hộp trắng có thể được áp dụng tại mức độ test đơn vị - unit, tích hợp hệ thống- integration của quá trình kiểm thử phần mềm, thường được thực hiện ở mức đơn vị.

#### *1.4.4.1 Các loại kiểm thử hộp trắng*

White box testing có 2 loại kiểm thử chính:

- Kiểm thử đơn vị (Unit Testing): là quá trình test từng module nhỏ trong hệ thống để xác nhận rằng mỗi thành phần của phần mềm thực hiện đúng với thiết kế.

- Kiểm tra rò rỉ bộ nhớ (Testing for Memory Leaks): là điều cần thiết để tránh ứng dụng phần mềm chạy chậm gây ảnh hưởng đến chất lượng sản phẩm cũng như trải nghiệm người dùng.

#### *1.4.4.2 Các kỹ thuật kiểm thử hộp trắng*

Một số kỹ thuật kiểm thử hộp trắng phổ biến:

- Kiểm thử đường đi (Path testing): Kỹ thuật này kiểm tra tất cả các đường đi có thể trong mã nguồn của phần mềm. Người kiểm thử tạo ra các bảng liệt kê các đường đi trong mã nguồn để kiểm tra tính đầy đủ và logic của chương trình.

- Kiểm thử điều kiện (Condition testing): Kỹ thuật này kiểm tra các điều kiện có thể xảy ra trong mã nguồn của phần mềm, để đảm bảo rằng chương trình phản hồi đúng các trường hợp đó.

- Kiểm thử dòng điều khiển (Control flow testing): Kỹ thuật này kiểm tra các thực hiện của dòng điều khiển trong mã nguồn của phần mềm. Người kiểm thử tạo ra các bảng liệt kê các nhánh điều khiển và các điểm thực hiện trong chương trình để đảm bảo rằng chương trình thực hiện đúng các thực hiện đó.

- Kiểm thử kiểu dữ liệu (Data type testing): Kỹ thuật này kiểm tra tính đúng đắn của các kiểu dữ liệu được sử dụng trong mã nguồn của phần mềm.

- Kiểm thử lỗi biên (Boundary testing): Kỹ thuật này kiểm tra việc xử lý các giá trị biên trong phần mềm. Người kiểm thử kiểm tra các giá trị biên để đảm bảo rằng phần mềm có thể xử lý đúng các giá trị này.

- Kiểm thử bộ nhớ (Memory testing): Kỹ thuật này kiểm tra việc sử dụng bộ nhớ của phần mềm. Người kiểm thử kiểm tra tính đúng đắn và hiệu suất của việc sử dụng bộ nhớ để đảm bảo rằng phần mềm không gây ra lỗi về bộ nhớ.

Các kỹ thuật kiểm thử hộp trắng có thể được sử dụng độc lập hoặc kết hợp với nhau để đảm bảo rằng phần mềm được kiểm tra đầy đủ và chính xác.

#### 1.4.5 Kiểm thử hộp xám



**Hình 1.6: Kiểm thử hộp xám**

Kiểm thử hộp xám (Gray Box Testing) là sự kết hợp giữa kiểm thử hộp đen và kiểm thử hộp trắng.

Mục đích của thử nghiệm này là để tìm ra được các lỗi nếu có do cấu trúc không đúng hoặc sử dụng không đúng cách của các ứng dụng.

##### 1.4.5.1 Tại sao phải kiểm thử hộp xám ?

Kiểm thử hộp xám có sự kết hợp giữa lợi ích của kiểm thử hộp đen và hộp trắng.

Kiểm thử hộp xám thường được sử dụng trong Kiểm thử tích hợp (Integration test). Tuy nhiên, dựa vào giải thuật và chức năng, nó cũng có thể được sử dụng ở nhiều mức kiểm thử khác nhau.

Cách tiếp cận của phương pháp này hết sức hữu ích khi kiểm tra các ứng dụng web.

Trong quá trình tiến hành kiểm thử hộp xám, người kiểm thử sẽ tạo input vào từ phía front-end, sau đó xác minh dữ liệu từ phía back-end.

##### 1.4.5.2 Kỹ thuật trong kiểm thử hộp xám

###### a. Kiểm thử ma trận (Matrix Testing)

Đây là loại kiểm thử nhằm xác định các biến đang tồn tại trong hệ thống. Và khi ma trận được ghi lại đầy đủ với các dữ liệu đầu vào cho tất cả các biến có

sẵn, nó sẽ trở nên như là một danh sách để đảm bảo rằng không bỏ sót một trường hợp nào.

**b. Kiểm thử hồi quy (Regression Testing)**

Khi có 1 chức năng mới thêm vào, để đảm bảo chức năng mới này không làm ảnh hưởng đến các chức năng khác trong hệ thống, chúng ta cần phải cân nhắc việc kiểm tra lặp lại những trường hợp đã từng kiểm thử rồi. Lúc này việc kiểm thử lặp lại được gọi là kiểm thử hồi quy.

**c. Kiểm thử mảng trực giao (Orthogonal Array Testing or OAT)**

Đây như là một kỹ thuật thống kê để tạo ra hoán vị các đầu vào, tạo ra các test case có phạm vi kiểm tra tối ưu để giảm công sức của con người trong giai đoạn lập kế hoạch thử nghiệm và thiết kế thử nghiệm.

**d. Kiểm thử mẫu (Pattern Testing)**

Kỹ thuật này được thực hiện trên lịch sử dữ liệu của các lỗi hệ thống trước đó. Không giống như thử nghiệm hộp đen, kiểm thử hộp xám đào sâu vào trong mã nguồn và xác định lý do tại sao sự cố xảy ra.

## **1.5 Tổng kết chương 1**

Chương 1 của bài thực tập đã trình bày được định nghĩa của Kiểm thử phần mềm, tiêu chuẩn để đánh giá một sản phẩm tốt và những vấn đề xung quanh kiểm thử phần mềm.

Các vấn đề cụ thể mà bài thực tập tiềm hiểu trong chương 1 là:

- Phân loại kiểm thử phần mềm.
- Quy trình kiểm thử phần mềm.
- Các mức kiểm thử phần mềm: kiểm thử mức đơn vị, mức tích hợp, mức hệ thống và mức chấp nhận.
- Các kỹ thuật kiểm thử: kỹ thuật kiểm thử tĩnh, kỹ thuật kiểm thử động.
- Các chiến lược kiểm thử: kiểm thử hộp đen, kiểm thử hộp trắng và kiểm thử hộp xám.

## CHƯƠNG 2. KIỂM THỬ TỰ ĐỘNG

### 2.1 Tổng quan về kiểm thử tự động

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

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

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

#### 2.1.1 Khái niệm kiểm thử tự động

Kiểm thử tự động là quá trình thực hiện một cách tự động các bước trong một kịch bản kiểm thử. Kiểm thử tự động bằng một công cụ nhằm rút ngắn thời gian kiểm thử.

Mục đích của kiểm thử tự động là giảm thiểu thời gian, công sức và kinh phí trong khi vẫn tăng độ tin cậy, tăng tính hiệu quả và giảm sự nhầm lẫn cho người kiểm thử trong quá trình kiểm thử sản phẩm phần mềm.

#### 2.1.2 Quy trình kiểm thử tự động

Quy trình kiểm thử tự động gồm 4 bước:

-Bước 1: Viết kịch bản kiểm thử, dùng công cụ kiểm thử để ghi lại các thao tác lên phần mềm cần kiểm tra và tự động sinh ra test script.

-Bước 2: Chỉnh sửa để kịch bản kiểm thử thực hiện kiểm tra theo đúng yêu cầu đặt ra, làm theo trường hợp kiểm thử cần thực hiện.

-Bước 3: Chạy kịch bản kiểm thử, giám sát hoạt động kiểm tra phần mềm của kịch bản kiểm thử.

-Bước 4: Kiểm tra kết quả thông báo sau khi thực hiện kiểm thử tự động. Sau đó bổ sung, chỉnh sửa những sai sót.

#### 2.1.3 Ưu và nhược điểm của kiểm thử tự động

a) Ưu điểm

**Độ tin cậy cao:** Nhờ sự ổn định vượt trội của công cụ kiểm thử tự động so với các hoạt động thủ công của người kiểm thử, đặc biệt trong trường hợp có quá nhiều ca kiểm thử cần được thực thi, nên độ tin cậy của kiểm thử tự động thường cao hơn so với kiểm thử thủ công.

**Khả năng lặp:** Khi phải thực thi một ca kiểm thử với 50 bộ dữ liệu đầu vào khác nhau, nếu thực thi cách thủ công bằng việc nhập dữ liệu và kiểm tra trong 50 lần thì sẽ rất tốn thời gian. Tuy nhiên, với việc sử dụng các công cụ kiểm thử tự động, chúng ta chỉ cần nhập dữ liệu vào một tệp Excel hoặc một kịch bản (script) rồi cho công cụ thực hiện và nhận được báo cáo kiểm thử từ công cụ này. Khi một ca kiểm thử phát hiện ra lỗi và chúng ta phải sửa nó và cần thực hiện lại tất cả các ca kiểm thử từ đầu. Trong trường hợp này, các công cụ kiểm thử tự động sẽ hoàn toàn trợ giúp chúng ta công việc này.

**Khả năng tái sử dụng:** Đây chính là một bộ kiểm thử tự động được nhiều người sử dụng với nhiều những phiên bản khác nhau và được gọi là tái tính sử dụng.

**Chi phí thấp:** Đặc biệt, chi phí sử dụng kiểm thử tự động khá hấp dẫn và phù hợp có thể tiết kiệm được nhiều chi phí cũng như thời gian nhân lực. Do quá trình kiểm thử nhanh hơn thủ công nên nhân lực sẽ được thực thi và bảo trì không nhiều.

**Tốc độ cao:** Với tốc độ kiểm thử nhanh hơn nhiều với tốc độ của con người. Những thực thi của một test case một cách thủ công có thể hoàn thành hay thực thi trong thời gian ngắn nhất một cách tự động.

#### b) Nhược điểm

Chi phí cao cho việc chuyển giao công nghệ và đào tạo nhân viên.

Tốn chi phí đầu tư lớn cho việc phát triển công cụ kiểm thử tự động.

Tốn chi phí và thời gian cho việc tạo các kịch bản kiểm thử và bảo trì các kịch bản kiểm thử.

Giai đoạn chuẩn bị kiểm thử yêu cầu nhiều nhân lực.

Khu vực kiểm thử tự động có thể không bao quát đầy đủ, không áp dụng được trong việc tìm lỗi mới của phần mềm.

#### ***2.1.4 Các trường hợp nên áp dụng kiểm thử tự động***

Không phải lúc nào cũng nên áp dụng kiểm thử tự động trong việc kiểm thử phần mềm, vì nhiều khi chi phí và thời gian cho việc kiểm thử tự động còn

lớn hơn nhiều so với kiểm thử thủ công. Dưới đây là một số trường hợp nên áp dụng phương pháp kiểm thử tự động để đạt được hiệu quả cao về thời gian, chi phí cũng như chất lượng.

Trường hợp không đủ tài nguyên: Là khi số lượng trường hợp kiểm thử lặp lại quá nhiều trên nhiều môi trường kiểm thử khác nhau, không có đủ nguồn nhân lực và chi phí để kiểm thử thủ công trong một giới hạn thời gian nào đó.

Trường hợp kiểm thử hồi quy: Trong quá trình phát triển phần mềm, nhóm lập trình thường đưa ra nhiều phiên bản phần mềm liên tiếp để kiểm thử. Thực tế cho thấy việc đưa ra các phiên bản phần mềm có thể là hàng ngày, mỗi phiên bản bao gồm những tính năng mới, hoặc tính năng cũ được sửa lỗi hay nâng cấp. Việc bổ sung hoặc sửa lỗi mã chương trình cho những tính năng ở phiên bản mới có thể làm cho những tính năng khác đã kiểm tra tốt chạy sai mặc dù phần mã chương trình của nó không hề chỉnh sửa. Để khắc phục điều này, đối với từng phiên bản, kiểm thử viên không chỉ kiểm tra chức năng mới hoặc được sửa, mà phải kiểm tra lại tất cả những tính năng đã kiểm tra tốt trước đó. Điều này khó khả thi về mặt thời gian nếu kiểm thử thủ công.

Trường hợp kiểm thử khả năng vận hành phần mềm trong môi trường đặc biệt: Đây là kiểm thử nhằm đánh giá xem vận hành của phần mềm có thỏa mãn yêu cầu đặt ra hay không. Thông qua đó kiểm thử viên có thể xác định được các yếu tố về phần cứng, phần mềm ảnh hưởng đến khả năng vận hành của hệ thống. Có thể liệt kê một số tình huống kiểm tra tiêu biểu thuộc loại này như sau:

Đo tốc độ trung bình xử lý một yêu cầu của eb server.

Thiết lập 1000 yêu cầu, đồng thời gửi đến eb server để kiểm tra tình huống 1000 người dùng truy xuất eb cùng lúc.

Xác định số yêu cầu tối đa được xử lý bởi eb server hoặc xác định cấu hình máy thấp nhất mà tốc độ xử lý của phần mềm vẫn có thể hoạt động ở mức cho phép.

## **2.2 Kiểm thử website**

### **2.2.1 Khái niệm kiểm thử website**

Kiểm thử website là tên gọi được đặt cho một quá trình kiểm thử phần mềm tập trung vào việc kiểm tra các ứng dụng web. Ứng dụng web cần được kiểm tra hoàn toàn trước khi đi vào hoạt động, điều này có thể giúp giải quyết các vấn đề trong ứng dụng web trước khi tiếp xúc với người dùng như các vấn đề về chức năng, bảo mật, các vấn đề dịch vụ web, các vấn đề tích hợp và khả năng xử lý lưu

lượng truy cập, trong quá trình kiểm thử website, cần cố gắng phát hiện ra lỗi có thể xảy ra trong hệ thống nhằm giải quyết kịp thời.

Hiểu theo cách đơn giản thì kiểm thử website chính là kiểm tra xem ứng dụng web có chứa những lỗi tiềm tàng nào không, trước khi chính thức đưa website đi vào sử dụng. Đây là một công việc liên quan đến lập trình web app, các khâu cần kiểm thử đó là bảo mật, chức năng, khả năng xử lý lưu lượng, hiệu suất trang web.

## **2.2.2 Các kỹ thuật kiểm thử Website**

### **2.2.2.1 Kiểm thử chức năng website**

Trong kiểm thử chức năng ( Functionality Testing) chúng ta cần kiểm tra từng thành phần hoạt động có như mong đợi hay không, vì vậy nó còn được gọi là “kiểm thử các thành phần”. Kiểm thử chức năng giúp kiểm tra các chức năng của thành phần ứng dụng, về cơ bản là để kiểm tra các chức năng được đề cập trong tài liệu mô tả chức năng cũng như kiểm tra xem ứng dụng phần mềm có đáp ứng được kỳ vọng của người dùng hay không.

Các hoạt động kiểm thử này bao gồm:

#### **a) Kiểm thử liên kết**

Kiểm tra tất cả các liên kết hỏng trên website và tất cả các liên kết đang hoạt động chính xác, bạn có thể kiểm tra các liên kết khác nhau trên website:

- Liên kết nội bộ
- Liên kết ngoài
- Liên kết mail
- Liên kết anchor

#### **b) Kiểm thử web form**

Đây là phần thiết yếu của bất kỳ kiểm thử website nào, mục đích chính của kiểm thử web form là lấy thông tin từ người sử dụng và lưu trữ vào cơ sở dữ liệu đồng thời tương tác với lượng dữ liệu ấy. Dưới đây là các trường hợp kiểm thử được nhắc tới trong kiểm thử web form:

-Điều đầu tiên là kiểm tra tính hợp lệ trên mỗi field của form, dưới đây là hai loại Validation cần được xem xét – “Client side” và “Server side” validations.

-Kiểm tra các giá trị mặc định.

-Kiểm tra tất cả các field bắt buộc.



-Kiểm tra nếu người dùng không nhập vào một field bắt buộc cần hiển thị một thông báo.

-Thêm và sửa thông tin bằng cách sử dụng form.

-Thứ tự các tab trên web form.

-Kiểm tra các giá trị mặc định của field.

-Form cần được định dạng tối ưu khả năng đọc.

-Kiểm tra số âm.

#### c) Kiểm thử cookie

Cookie là tập tin chứa thông tin hệ thống của người dùng, các tệp này được lưu ở vị trí mong muốn và được sử dụng bởi các trình duyệt. Các session đăng nhập, thông tin được lưu lại trong cookie (như session) và có thể được truy xuất cho các trang web. Người dùng có thể kích hoạt hoặc vô hiệu Cookies trong các tùy chọn trình duyệt, kiểm thử để kiểm tra xem cookie có được lưu trữ trong máy của người dùng ở định dạng mã hóa hay không.

-Kiểm tra ứng dụng bằng cách vô hiệu cookies.

-Kiểm tra ứng dụng sau khi hỏng các cookies.

-Kiểm tra hành vi của ứng dụng sau khi xóa tất cả cookie trên website.

-Kiểm tra cookie có hoạt động trên nhiều duyệt khác nhau hay không.

-Kiểm tra cookie cho đăng nhập xác thực có hoạt động hay không.

-Kiểm tra hành vi của ứng dụng sau khi xóa cookie (session) bằng cách xóa bộ nhớ cache hoặc sau khi cookie hết hạn.

-Kiểm tra đăng nhập vào ứng dụng sau khi xóa cookie (session).

#### d) Kiểm thử HTML và CSS

Kiểm thử này kiểm tra xem các công cụ tìm kiếm có thể thu thập dữ liệu trang web của bạn mà không xảy ra bất kỳ lỗi nào, bạn nên kiểm tra tất cả các lỗi cú pháp, màu sắc và tuân thủ theo tiêu chuẩn như W3C, ISO, ECMA, IETF, WS-I, OASIS.

Quy trình nghiệp vụ bao gồm:

- Kiểm tra luồng xử lý đảm bảo sự hoàn chỉnh của website.
- Kiểm tra các màn hình theo như tài liệu yêu cầu.

### 2.2.2.2 Kiểm thử khả năng sử dụng website

Usability testing đóng một vai trò quan trọng trong bất kỳ ứng dụng web, đảm bảo kiểm tra tất cả các test case xuất phát từ người dùng

Các hoạt động kiểm thử này bao gồm:

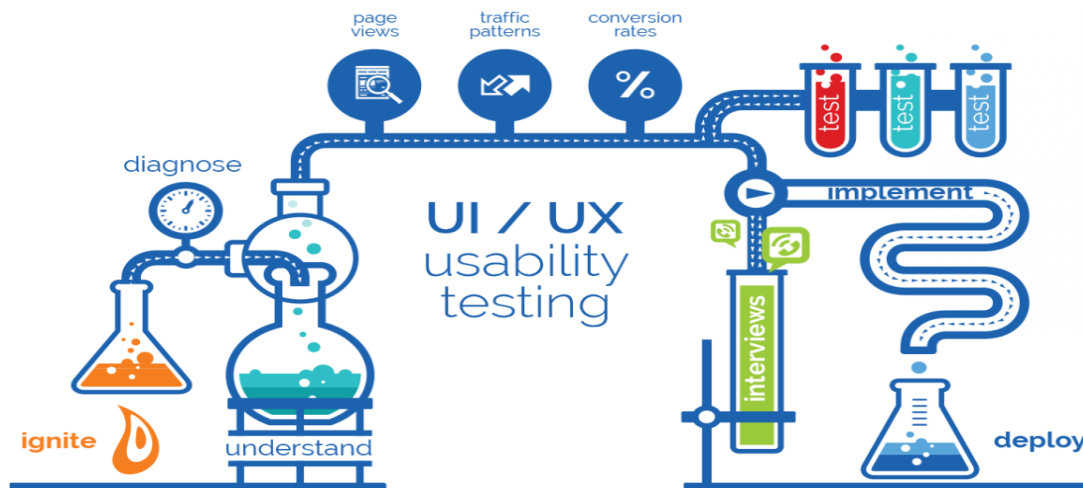
#### a) Kiểm tra điều hướng website

Tất cả các tùy chọn như UI/UX, menu, liên kết hoặc các button trên website phải hiển thị và có thể truy cập.

Điều hướng trang web dễ dàng sử dụng .

Nội dung hướng dẫn phải rõ ràng và phải đáp ứng được mục đích.

Tất cả tùy chọn trên header, footer và các điều hướng trái / phải phải nhất quán trên mỗi trang.



**Hình 2.1: Kiểm thử khả năng sử dụng trên website**

#### b) Kiểm tra nội dung website

- Không có lỗi chính tả hoặc ngữ pháp lỗi trong nội dung.
- Tích hợp Alt trong hình ảnh .
- Không có ảnh hỏng.
- Xác nhận tính hợp lệ tất cả giao diện người dùng.
- Thực hiện theo một số tiêu chuẩn về xây dựng nội dung trên trang web.
- Tất cả nội dung phải rõ ràng và dễ hiểu.
- Màu tối gây bất bình cho người sử dụng, vì vậy tránh sử dụng theme màu tối.

- Kích thước hình ảnh kích nên phù hợp.
- Anchor text phải hoạt động bình thường.

#### c) Kiểm thử sự tương thích

Đảm bảo làm thế nào ứng dụng làm việc trong các môi trường được hỗ trợ, sử dụng ứng dụng web trên các hệ điều hành khác nhau, khả năng tương thích của trình duyệt, khả năng tính toán của phần cứng, cơ sở dữ liệu và khả năng xử lý băng thông mạng. Kiểm thử tương thích đảm bảo rằng “ứng dụng web có hiển thị đúng trên các thiết bị khác nhau không?”. Điều này sẽ bao gồm:



**Hình 2.2: Kiểm thử sự tương thích**

#### d) Kiểm tra độ tương thích của trình duyệt

Các ứng dụng web được hiển thị khác nhau trên các trình duyệt khác nhau, mục tiêu của kiểm tra khả năng tương thích của trình duyệt là đảm bảo rằng không có lỗi nào xảy ra trên các trình duyệt web khác nhau trong khi hiển thị website. Bạn cần đảm bảo rằng ứng dụng web của bạn đang được hiển thị đúng trên các trình duyệt khác nhau cũng như kiểm tra AJAX, JavaScript và xác thực hoạt động chính xác.

e) Khả năng tương thích hệ điều hành

Công nghệ mới, sử dụng đồ họa mới hơn, các API khác nhau được sử dụng có thể không hoạt động trên nhiều hệ điều hành, bên cạnh đó các text field, button có thể hiển thị khác nhau trên hệ điều hành khác nhau. Vì vậy, kiểm thử website cần được thực hiện trên các hệ điều hành khác nhau như Windows, MAC, Solaris, Unix, Linux.

f) Trình duyệt web di động

Bạn cũng nên kiểm tra khả năng tương thích website trên điện thoại di động, đôi khi có thể xảy ra sự cố tương thích trên các trình duyệt điện thoại di động.

g) Kiểm thử cơ sở dữ liệu

Độ tin cậy của dữ liệu là một phần quan trọng trong việc kiểm thử cơ sở dữ liệu. Vì vậy, đối với các ứng dụng web nên được kiểm tra một cách kỹ lưỡng. Các hoạt động kiểm tra bao gồm:



**Hình 2.3: Kiểm thử website cần độ chính xác từ cơ sở dữ liệu**

- Kiểm tra nếu các truy vấn được thực hiện mà không xảy ra lỗi.
- Thêm mới, cập nhật hoặc xóa dữ liệu trong cơ sở dữ liệu nên duy trì tính toàn vẹn của dữ liệu.
- Truy vấn dữ liệu không nên mất quá nhiều thời gian.

-Kiểm tra việc load dữ liệu và kết quả nhận được với các câu truy vấn dài.

Dữ liệu nhận được trên cơ sở dữ liệu và hiển thị trên website có chính xác hay không.

#### h) Kiểm thử giao diện

Kiểm thử giao diện chủ yếu có ba lĩnh vực cần được kiểm tra: Web Server, Application server và Database server. Đảm bảo rằng tất cả các thông tin liên lạc giữa các server này phải được thực hiện đúng, xác minh kết nối giữa các máy chủ được thiết lập lại hoặc bị mất, kiểm tra xem có bất kỳ xung đột giữa lúc ứng dụng đang hoạt động, trả về bất kỳ lỗi từ web server hoặc database server đến application server sau đó được xử lý và cuối cùng là hiển thị kết quả tới người dùng.



**Hình 2.4: Giao diện chiếm vai trò lớn với trong kiểm thử website**

Web server: kiểm tra xem tất cả các yêu cầu web có đang được chấp nhận và không yêu cầu nào bị từ chối hoặc bị rò rỉ.

Application server: kiểm tra xem yêu cầu có đang gửi đúng đến server, lỗi có được bắt và hiển thị cho người quản trị.

Database server: kiểm tra kết quả truy vấn cơ sở dữ liệu.

#### i) Kiểm thử hiệu năng website



**Hình 2.5: Kiểm thử hiệu năng có ảnh hưởng lớn đối với người dùng**

Kiểm thử website làm việc dưới lượt tải nặng, được phân thành hai phần: kiểm tra tần suất, kiểm tra lượt tải. Bao gồm:

- Kiểm tra thời gian phản hồi của website với tốc độ kết nối khác nhau.
- Kiểm tra website có xử lý được nhiều yêu cầu người dùng vào cùng một thời điểm.
- Kiểm tra website có hoạt động tốt trong thời điểm lượt tải cao.
- Kiểm tra dữ liệu đầu vào lớn từ người dùng.
- Kiểm tra hành vi của website khi kết nối với cơ sở dữ liệu.
- Kiểm tra các phương pháp tối ưu hóa như giảm thời gian tải bằng cách bật bộ nhớ cache trên trình duyệt và phía máy chủ, nén gzip...

#### k) Kiểm thử bảo mật website



**Hình 2.6: Kiểm thử bảo mật website**

Website là một trong những loại trang web có nguy cơ bị tấn công cao nhất. Vì vậy, ngoài việc đảm bảo trang web chạy đúng, ổn định cần phải kiểm tra nghiêm ngặt khả năng bảo mật. Được thực hiện để đảm bảo rằng có bất kỳ rò rỉ thông tin nào về mã hoá dữ liệu hay không. Trong website thương mại điện tử, kiểm thử bảo mật đóng một vai trò rất quan trọng, nếu thông tin an toàn thì kiểm tra xem làm thế nào để lưu trữ các thông tin nhạy cảm như thẻ tín dụng, thanh toán hóa đơn... Các hoạt động kiểm tra sẽ bao gồm:

- Kiểm tra truy cập trái phép vào các trang an toàn, nếu người dùng thay đổi từ “https” sang “http” thì thông báo thích hợp sẽ được hiển thị và ngược lại.

- Kiểm tra việc truy cập các trang internal, nếu đăng nhập được yêu cầu thì người dùng nên được chuyển hướng đến trang đăng nhập hoặc thông báo thích hợp sẽ được hiển thị.

- Các thông tin liên quan đến giao dịch, thông báo lỗi, cố gắng đăng nhập nên được ghi vào file log.

- Kiểm tra các tệp tin có bị hạn chế tải xuống hay không.

- Kiểm tra các thư mục web hoặc tập tin web có thể truy cập được trừ khi không được cấu hình để tải xuống.

- Kiểm tra CAPTCHA đã được thêm vào và hoạt động bình thường cho đăng nhập để tự động ngăn chặn các đăng nhập hay chưa.

- Kiểm tra việc cố truy cập thông tin bằng cách thay đổi tham số trong chuỗi truy vấn. Ví dụ: nếu bạn đang chỉnh sửa thông tin và trên URL bạn thấy UserID = 123, hãy thử thay đổi các giá trị tham số này và kiểm xem ứng dụng có



cung cấp thông tin người dùng khác không, nên từ chối hiển thị cho trường hợp này để ngăn chặn việc xem thông tin người dùng khác.

-Kiểm tra session hết hạn sau thời gian được xác định nếu người dùng không thao tác trên website.

-Kiểm tra user/password không hợp lệ.

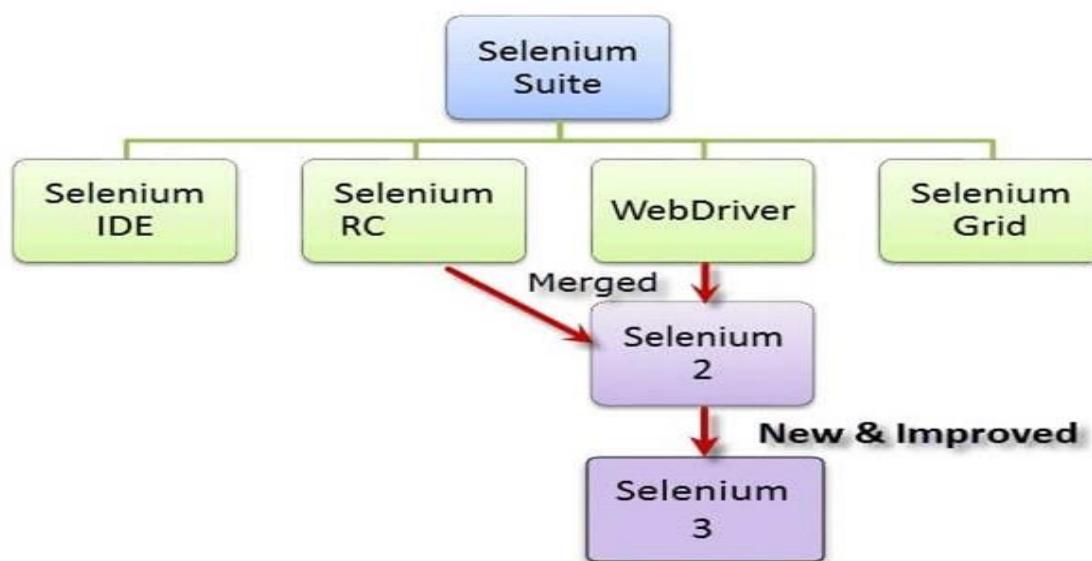
## 2.3 Một số công cụ hỗ trợ kiểm thử website phổ biến

### 2.3.1 Tổng quan về Selenium

#### 2.3.1.1 Khái niệm

Selenium (thường được viết tắt là SE) là một phần mềm mã nguồn mở, được phát triển bởi Jason Huggins, sau đó được tiếp tục phát triển bởi nhóm ThoughtWorks vào năm 2004.

Selenium là một bộ các công cụ hỗ trợ kiểm thử tự động các tính năng của ứng dụng web, bao gồm 4 phần: Selenium IDE, Selenium Remote Control (RC), Selenium Core và Selenium Grid.



Hình 2.7: Các phiên bản Selenium

Selenium hỗ trợ kiểm thử trên hầu hết các trình duyệt web phổ biến hiện nay như Firefox, Internet Explorer, Googlechrome và hỗ trợ trên rất nhiều ngôn ngữ lập trình phổ biến như C#, Java, Python, PHP. Không những vậy, Selenium còn có thể kết hợp với một số công cụ kiểm thử khác như Junit, Bromien, Nunit.



### 2.3.1.2 Đặc điểm

Cung cấp các điều khoản đề xuất khẩu ghi lại kịch bản trong các ngôn ngữ khác như Java, Ruby, RSpec, Python, C #, JUnit và TestNG

Nó có thể thực hiện nhiều bộ kiểm thử cùng một lúc.

Xác định phân tử sử dụng id, tên, đường dẫn X, v.v. + Lưu trữ các bộ kiểm thử như Ruby Script, HTML và bất kỳ định dạng nào khác.

Hỗ trợ tệp tin người dùng selenium-extensions.js

Cho phép để chèn ý kiến ở giữa của kịch bản để hiểu rõ hơn nội dung và mục đích của kịch bản kiểm thử.

### 2.3.1.3 Các hàm phổ biến trong Selenium

|   |  |
|---|--|
| <code>driver.get("URL")</code>                      | Đề điều hướng đến một trang web                                |
| <code>driver.findElement(...)</code>                | Tìm kiếm WebElement  |
| <code>element.sendKeys("inputtext")</code>          | Nhập một số văn bản vào textbox                                |
| <code>element.clear()</code>                        | Xóa nội dung textbox   |
| <code>driver.switchTo().window("windowName")</code> | Di chuyển con trỏ chuột từ cửa sổ này sang cửa sổ khác         |
| <code>driver.switchTo().window("windowName")</code> | Di chuyển con trỏ chuột từ cửa sổ này sang cửa sổ khác         |
| <code>driver.switchTo().alert()</code>              | Xử lý alert  |
| <code>driver.navigate().forward()</code>            | Chuyển hướng đến trang tiếp theo                               |
| <code>driver.navigate().back()</code>               | Chuyển hướng về trang trước                                    |
| <code>driver.close()</code>                         | Đóng trình duyệt hiện tại và các liên kết đến driver           |
| <code>driver.quit()</code>                          | Thoát driver và đóng tất cả các cửa sổ liên quan đến driver đó |

**Hình 2.8: Các hàm phổ biến trong Selenium**

### 2.3.1.4 Locator trong Selenium

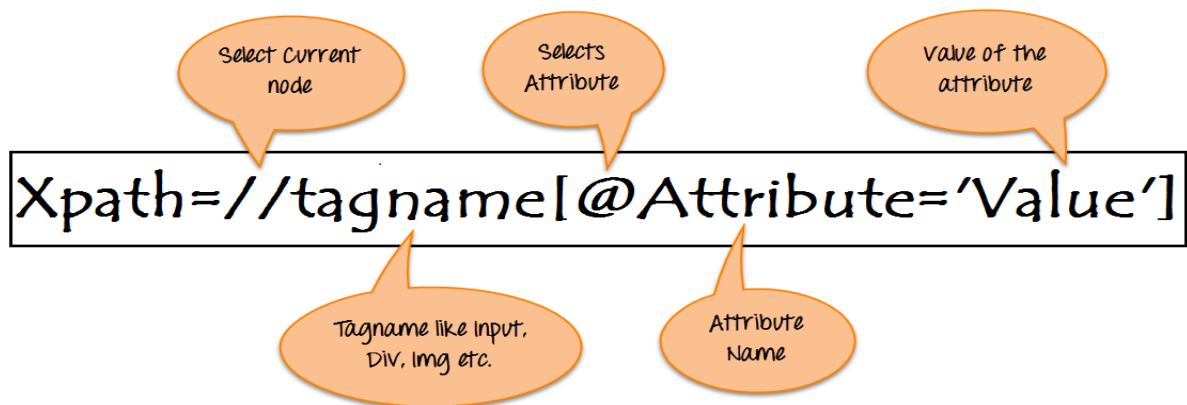
Locator có thể được gọi là địa chỉ để xác định một Web Element trong trang Web.

Có rất nhiều Web element nhưng phổ biến nhất trong số đó là:

- Text box
- Button
- Drop Down
- Hyperlink
- Check Box
- Radio Button

Các cách xác định Xpath

- Xpath dạng cơ bản



Hình 2.9: Cách lấy Xpath dạng cơ bản

Ví dụ:

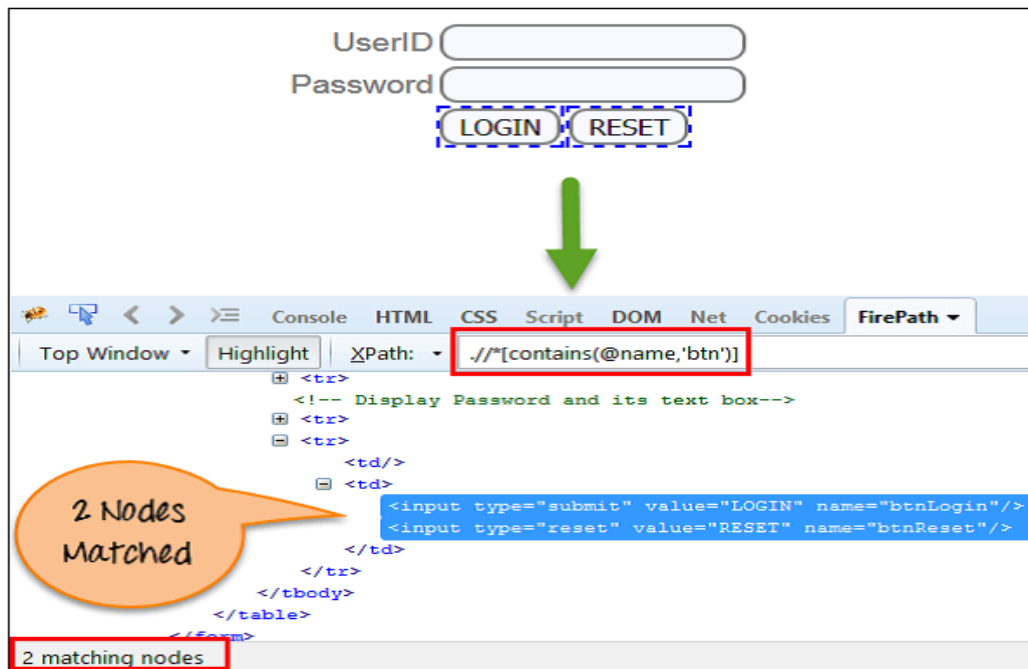
The screenshot shows a web browser with a login form. The form has fields for "UserID" and "Password", and buttons for "LOGIN" and "RESET". A message "User-ID must not be blank" is displayed next to the UserID field. The FirePath extension is open, showing the XPath `./input[@name='uid']` selected. The DOM tree shows the corresponding HTML element: `<input type='text' ... name='uid' />`. The console shows "1 matching node".

Examples of XPath expressions:

- `Xpath=//input[@type='text']`
- `Xpath= //label[@id='message23']`
- `Xpath= //input[@value='RESET']`
- `Xpath=//*[ @class='barone']`
- `Xpath=//a[@href='http://demo.guru99.com/']`
- `Xpath= //img[@src='//cdn.guru99.com/images/home/java.png']`

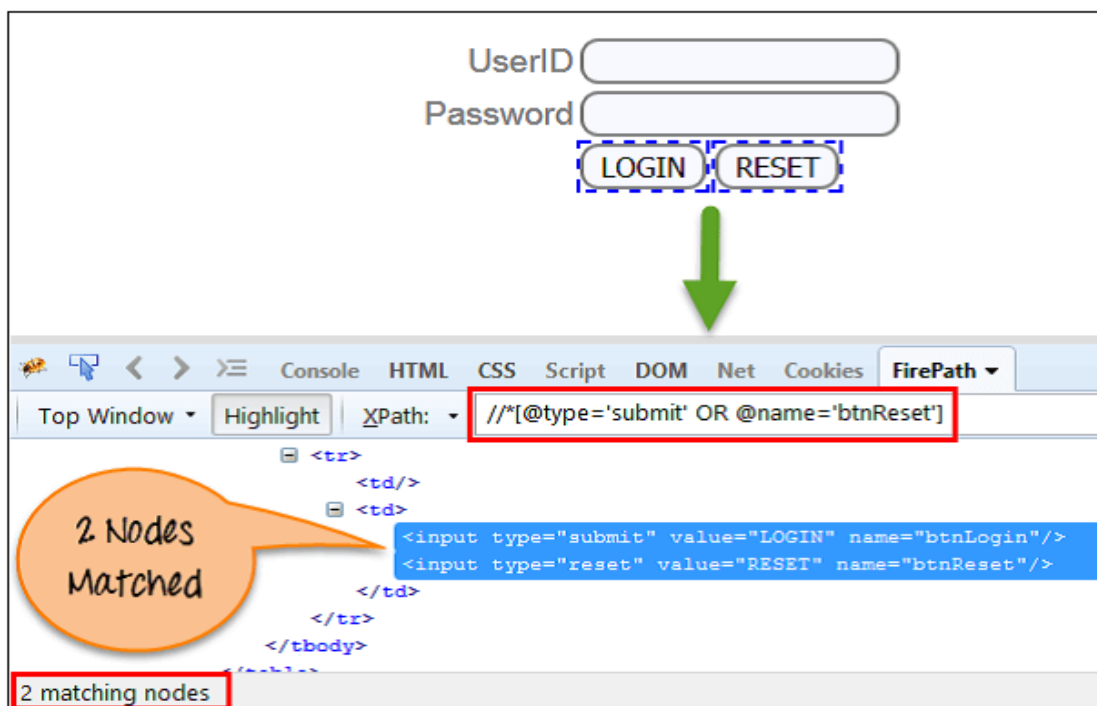
Hình 2.10: Ví dụ về cách lấy Xpath dạng cơ bản

- Xpath dạng nâng cao
  - Sử dụng contains:



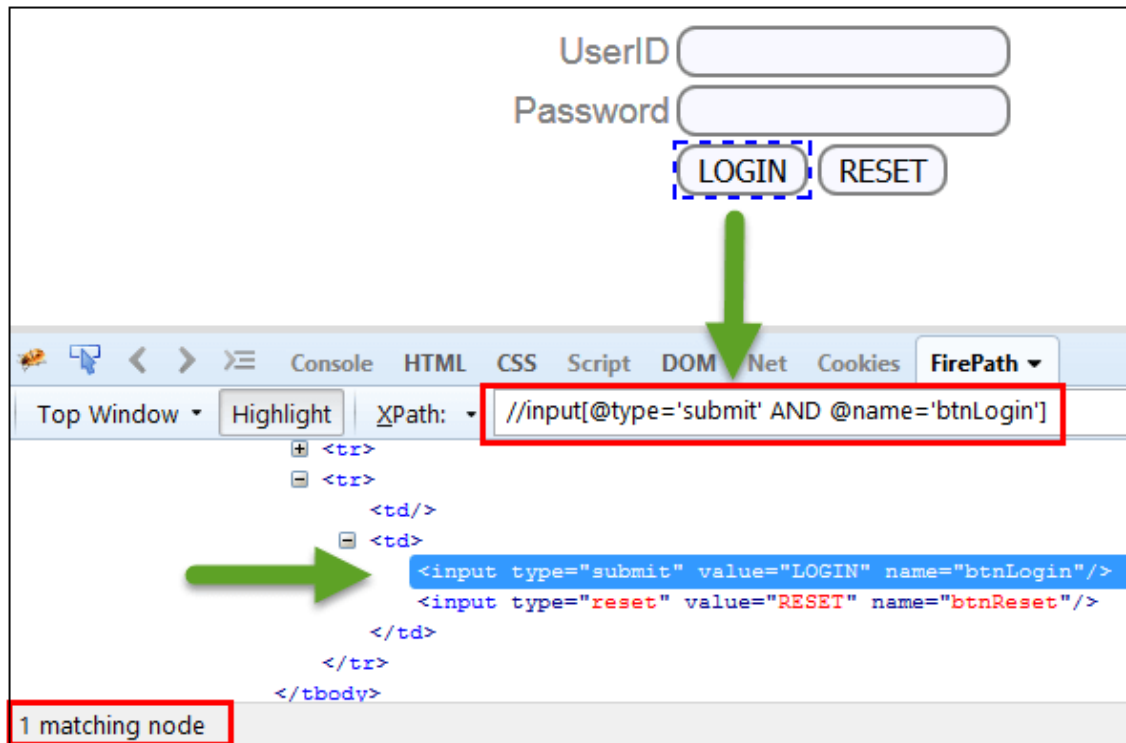
Hình 2.11: Ví dụ về cách lấy Xpath dùng contains

- Sử dụng OR:



Hình 2.12: Ví dụ về lấy Xpath dùng toán tử OR

➤ Sử dụng AND:



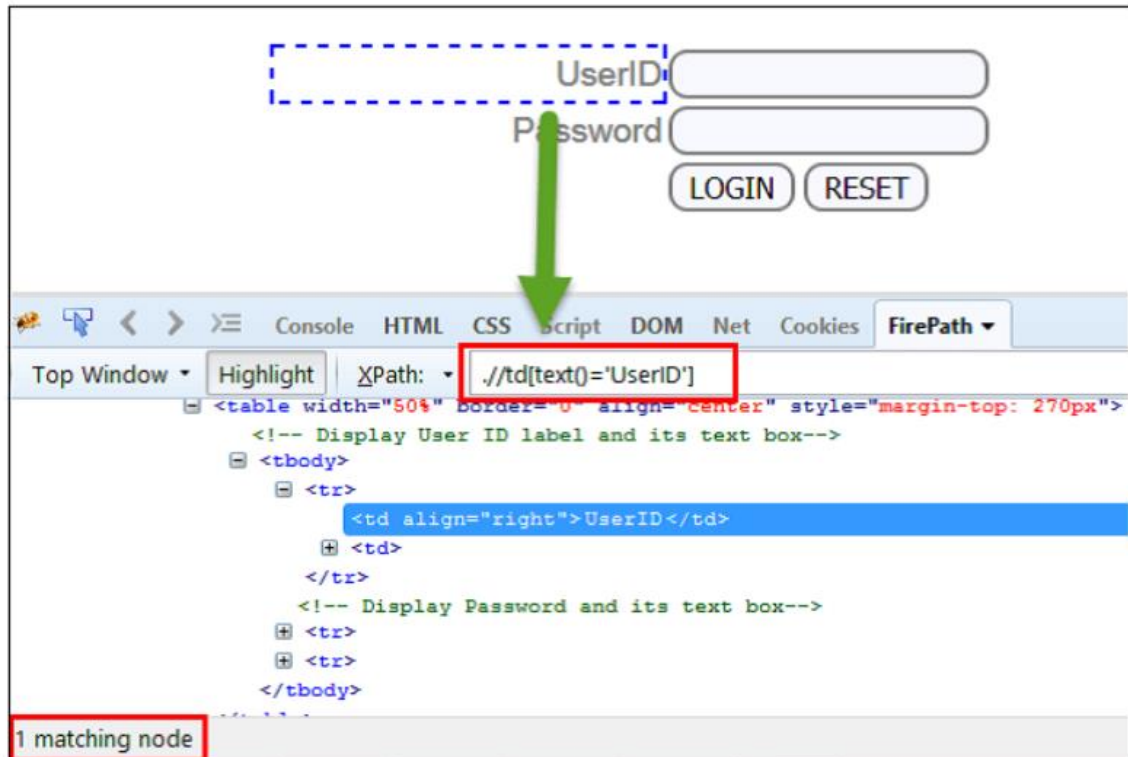
Hình 2.13: Ví dụ về lấy Xpath dùng toán tử AND

➤ Sử dụng Start-with:



Hình 2.14: Ví dụ về lấy Xpath dùng Start-with.

➤ Sử dụng text():



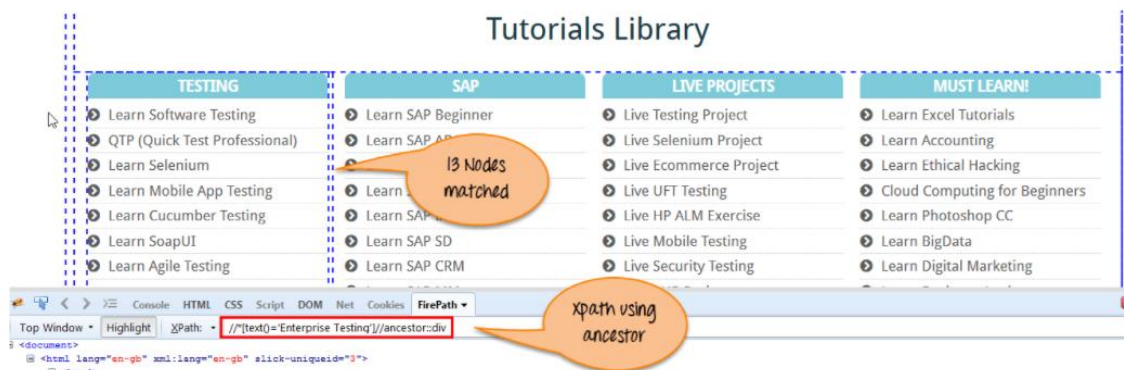
Hình 2.15: Ví dụ về cách lấy Xpath dùng text()

➤ Sử dụng thông qua phương thức axes: following



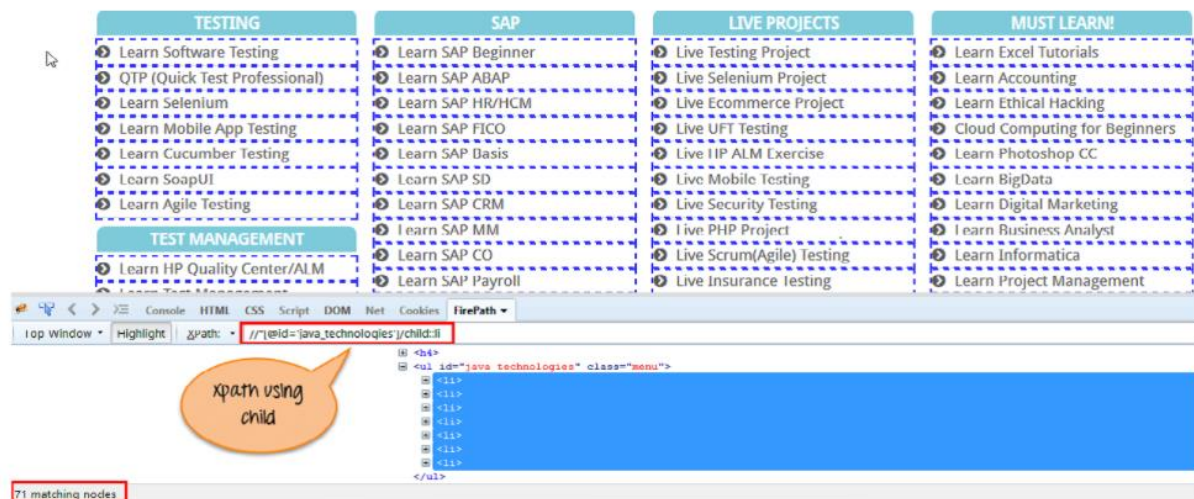
Hình 2.16: Ví dụ về lấy Xpath dùng following

➤ Sử dụng thông qua phương thức axes: **Ancestor**



Hình 2.17: Ví dụ về cách lấy Xpath dùng Ancestor

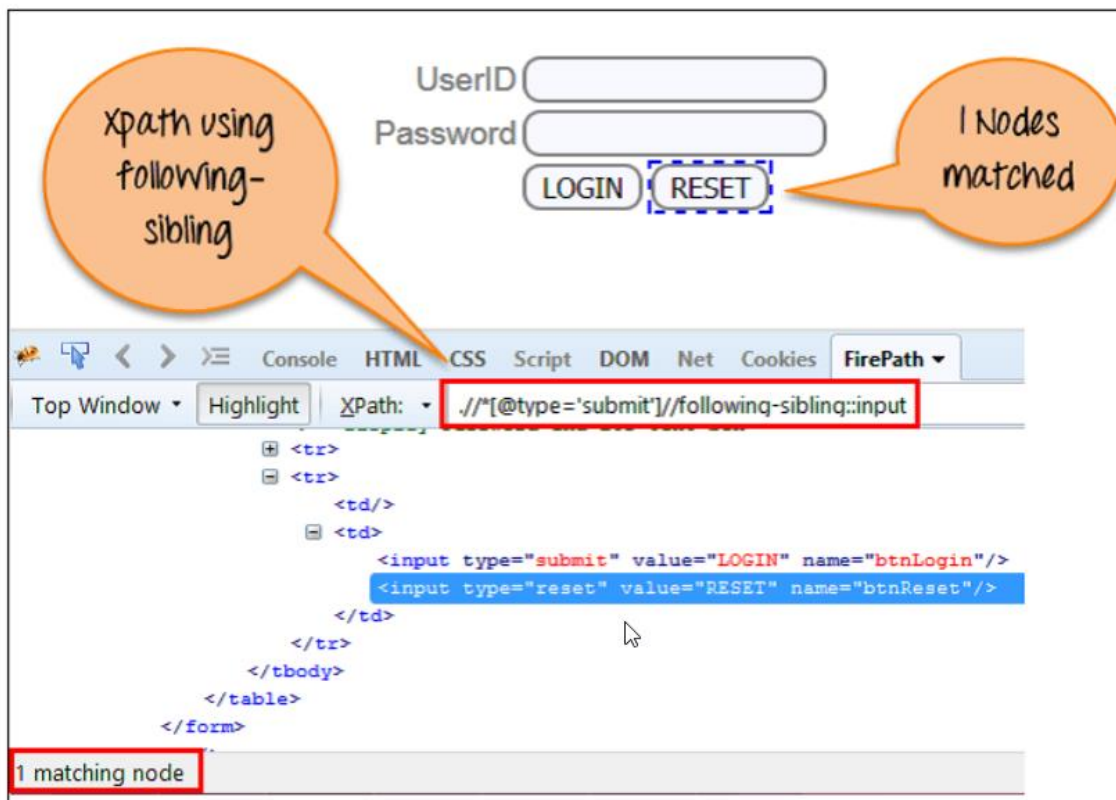
➤ Sử dụng thông qua phương thức axes: **Child**



Hình 2.18: Ví dụ về cách lấy Xpath dùng Child.

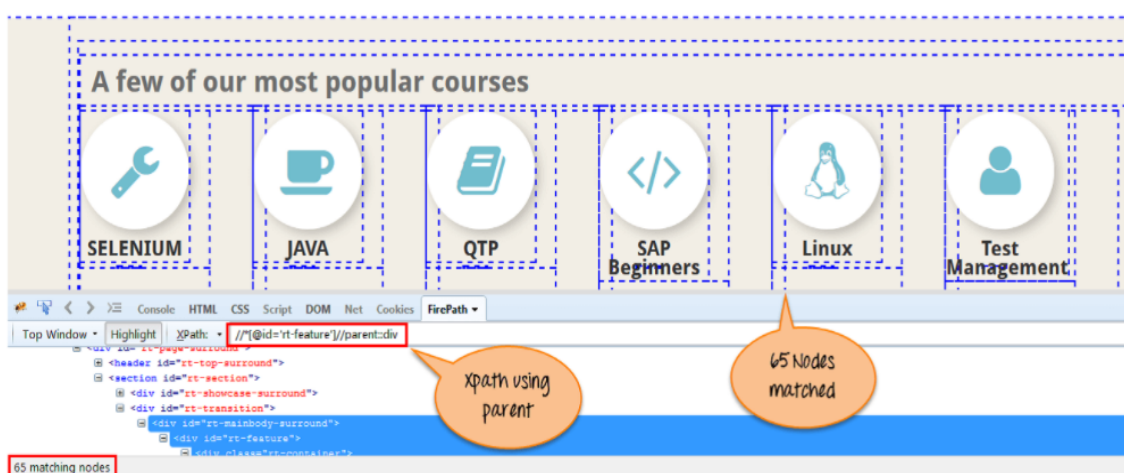


➤ Sử dụng thông qua phương thức axes: **Following-sibling**



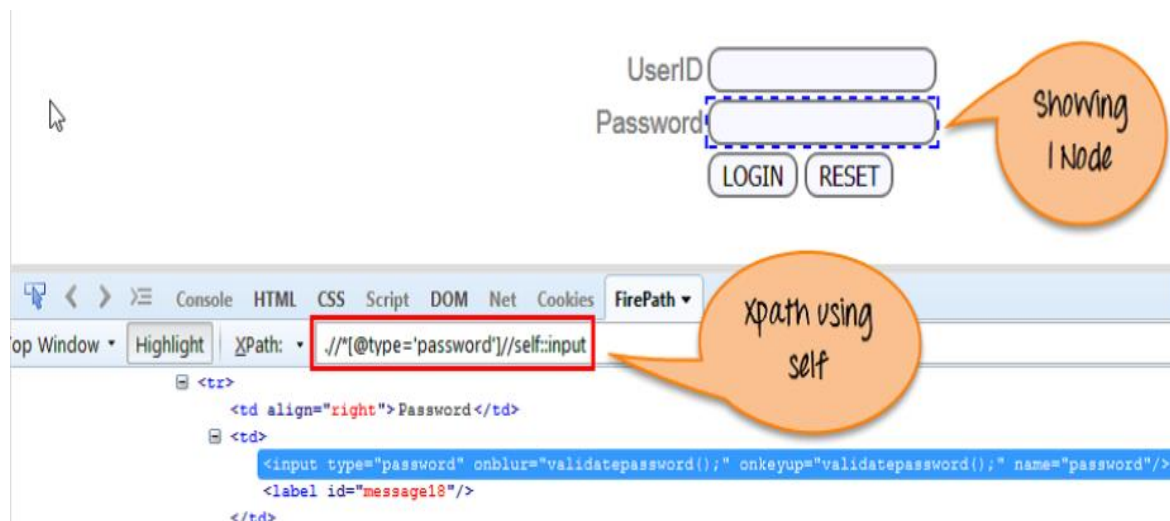
Hình 2.19: Ví dụ về cách lấy Xpath dùng Following-sibling

➤ Sử dụng thông qua phương thức axes: **Parent**



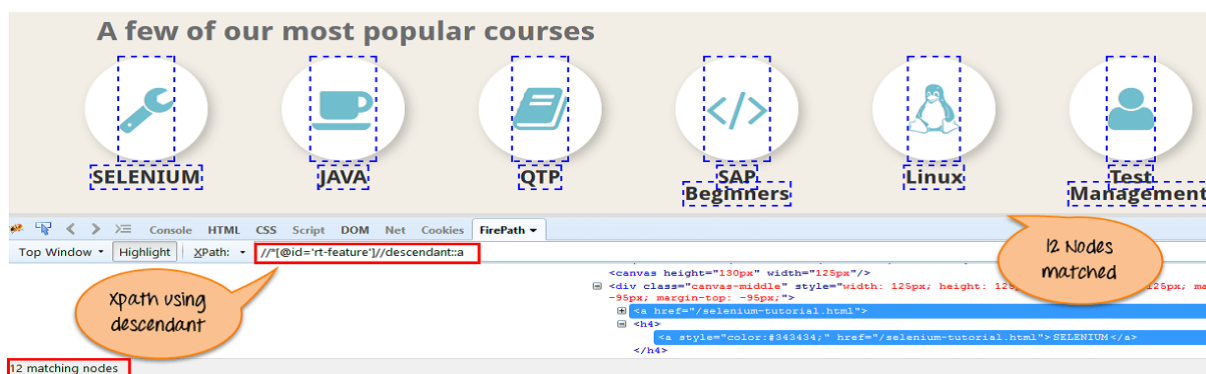
Hình 2.20: Ví dụ về cách lấy Xpath dùng Parent

➤ Sử dụng thông qua phương thức axes: **Self**



Hình 2.21: Ví dụ về cách lấy XPath dùng Self

➤ Sử dụng thông qua phương thức axes: **Descendant**



Hình 2.22: Ví dụ về cách lấy XPath dùng Descendant

## 2.3.2 Tổng quan về Serenity BDD và Cucumber

### 2.3.2.1 Serenity BDD

BDD (Behavior-Driven Development) là gì?

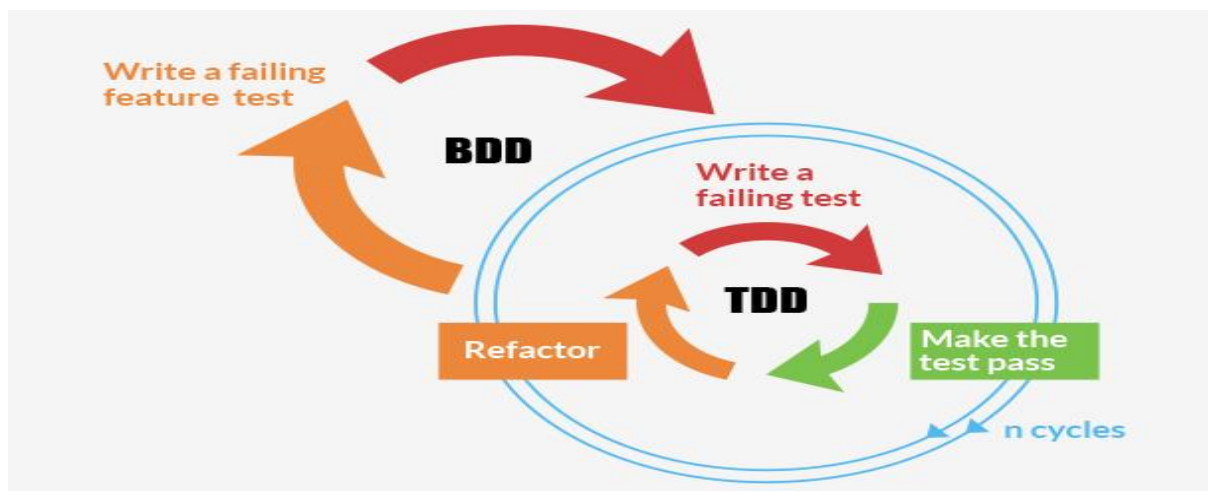
BDD (Behavior Driven Development) là một quá trình phát triển phần mềm dựa trên phương pháp Agile (phát triển phần mềm linh hoạt).

BDD là sự mở rộng của TDD (Test driven development). Thay vì tập trung vào phát triển phần mềm theo hướng kiểm thử, BDD tập trung vào phát triển phần mềm theo hướng hành vi.



Dựa vào requirement các kịch bản test (Scenarios) sẽ được viết trước dưới dạng ngôn ngữ tự nhiên và dễ hiểu nhất sau đó mới thực hiện cài đặt source code để pass qua tất cả các stories đó.

Những kịch bản test này được viết dưới dạng các feature file và đòi hỏi sự cộng tác từ tất cả các thành viên tham gia dự án hay stakeholder.



Hình 2.22: Mô hình BDD

### 2.3.2.2 Serenity with Cucumber

#### a) Cucumber

Cucumber là một công cụ kiểm thử tự động dựa trên việc thực thi các chức năng được mô tả dưới dạng ngôn ngữ tự nhiên, mục đích là để hỗ trợ cho việc viết BDD.

Điều này có nghĩa rằng kịch bản test unit (scenarios) sẽ được viết trước và thể hiện nghiệp vụ, sau đó source code mới được cài đặt để pass qua tất cả các stories đó.

Cucumber thực thi các feature file. Các feature files chứa các bước (step) thực thi, các bước này được viết bằng ngôn ngữ Gherkin.

#### b) Gherkin

Gherkin là 1 ngôn ngữ mà Cucumber đọc ngôn ngữ ấy chuyển thành test.

Gherkin khá dễ hiểu, người đọc có thể hiểu kịch bản và hành động mà không cần biết chi tiết chúng được cài đặt như thế nào.

#### **Có 2 quy tắc khi viết gherkin:**

- Một file Gherkin chỉ mô tả cho một feature
- Source file Gherkin là feature.

## Ngữ pháp của Gherkin:

- Ngữ pháp của Gherkin chia đầu vào thành các feature, scenario và step.
- Tất cả các file “ \*feature” được quy ước chỉ bao gồm một tính năng.
- Dòng bắt đầu với từ khóa “Feature:”
- Một tính năng thường gồm một danh sách các Scenario. Có thể viết bất cứ bạn muốn cho đến khi bắt đầu Scenario đầu tiên
- Có thể sử dụng tags để nhóm các feature và scenario lại với nhau, không lệ thuộc vào file và cấu trúc thư mục.

## Tips & keywords khi sử dụng gherkin

- Mỗi scenario bao gồm một danh sách các bước, các bước bắt đầu bằng các keyword như (Given, When, Then, But hoặc And).
- Bắt đầu sẽ là Feature sau đó đến Scenario hoặc Scenario Outline và Steps.

## Keywords

**Feature:** Mô tả 1 feature

Scenario: Mỗi 1 scenario thể hiện 1 testcase , 1 mục tiêu cụ thể. Mỗi 1 scenario sẽ gồm : Pre-conditions, step thực hiện và kết quả mong đợi. (Given-When-Then)

**Given:** Được sử dụng để mô tả ngữ cảnh ban đầu của hệ thống.

+ Mục đích của Given là đưa hệ thống vào một trạng thái đã biết trước khi sử dụng (hoặc hệ thống bên ngoài) bắt đầu tương tác với hệ thống (trong bước When).

+ Nếu bạn đã làm việc với use case, Givens là điều kiện tiên quyết.

+ Khi Cucumber thực thi bước Given, nó sẽ cấu hình hệ thống để được một trạng thái rõ ràng như là: tạo, cấu hình các đối tượng hoặc thêm dữ liệu test vào cơ sở dữ liệu.

**When:** Mục đích của When là để mô tả các sự kiện, hành động chính mà người dùng sử dụng.

'When' được sử dụng để mô tả một sự kiện, hoặc một hành động. Đây có thể là một người tương tác với hệ thống, hoặc nó có thể là một sự kiện được kích hoạt bởi một hệ thống khác.

**Then:** Mục đích của Then là quan sát kết quả. Các quan sát phải được liên quan đến các giá trị kinh doanh / lợi ích trong việc mô tả feature. Các quan sát phải kiểm tra đầu ra của hệ thống (một báo cáo, giao diện người dùng, tin nhắn,...)

'Then' được sử dụng để so sánh kết quả thực tế (hệ thống thực sự làm) với kết quả mong đợi (những gì các bước nói hệ thống là nghĩa vụ phải làm).

**And:** Thay thế cho các từ khóa Given/ When/ Then để làm cho chương trình mạch lạc hơn

### 2.3.3 Tổng quan về Jmeter

JMeter được xây dựng dựa trên plugin. Hầu hết các tính năng vượt trội của nó được áp dụng với các plugin. Các nhà phát triển có thể đơn giản hóa sử dụng JMeter bằng các plugin tùy chỉnh.

JMeter làm việc trên nhiều giao thức khác nhau:

- Web Services - SOAP( Simple Object Access Protocol) / XML – RPC
- Web – HTTP, HTTPS
- Database – JDBC driver
- Directory – LDAP( Lightweight Directory Access Protocol)
- Messaging Oriented service – JMS( Java Message Service)
- Service – POP3( Post Office Protocol version 3)
- IMAP( Internet Message Access Protocol)
- SMTP( Simple Mail Transfer Protocol)
- FTP( File Transfer Protocol).

### Các tính năng được hỗ trợ bởi JMeter

Mọi người thường hỏi tại sao chúng ta nên sử dụng JMeter? Vậy, để làm rõ điều này, chúng ta hãy cùng xem những tính năng "awesome" của JMeter:

*Ứng dụng mã nguồn mở:* JMeter là một ứng dụng mã nguồn mở miễn phí tạo điều kiện cho người dùng hoặc nhà phát triển sử dụng mã nguồn cho mục đích phát triển hoặc sửa đổi khác.

*Giao diện thân thiện với người dùng:* JMeter có giao diện như một native app thông thường. Nó rất đơn giản, dễ sử dụng và người dùng sẽ sớm làm quen với nó.

*Nền tảng độc lập:* Nó hoàn toàn là một ứng dụng desktop dựa trên ngôn ngữ Java. Đó là tại sao nó có thể chạy trên mọi nền tảng. Nó có khả năng mở rộng cao và có khả năng tải kiểm tra hiệu năng trong nhiều loại máy chủ khác nhau:

Web - HTTP, HTTPS, SOAP, Cơ sở dữ liệu thông qua JDBC, LDAP, JMS, Mail - POP3.

*Viết script của riêng mình:* Sử dụng plugin có thể viết trường hợp kiểm thử của riêng bạn để mở rộng quá trình kiểm thử. JMeter sử dụng trình soạn thảo văn bản để tạo một test plan và cung cấp ở định dạng XML.

*Hỗ trợ các phương pháp kiểm thử khác nhau:* JMeter hỗ trợ các phương pháp kiểm thử khác nhau như Load Testing, Distributed Testing, and Functional Testing, v.v.

*Mô phỏng:* JMeter có thể mô phỏng nhiều người dùng khác nhau với các luồng một cách đồng thời, tạo ra mức tải nặng đối với ứng dụng web đang được kiểm thử.

*Hỗ trợ đa giao thức:* JMeter có thể hoạt động trên kiểm thử ứng dụng web và kiểm thử máy chủ cơ sở dữ liệu và cũng hỗ trợ các giao thức như HTTP, JDBC, LDAP, SOAP, JMS và FTP.

*Script Test:* JMeter có khả năng thực hiện kiểm tra tự động hóa bằng Bean Shell & Selenium.

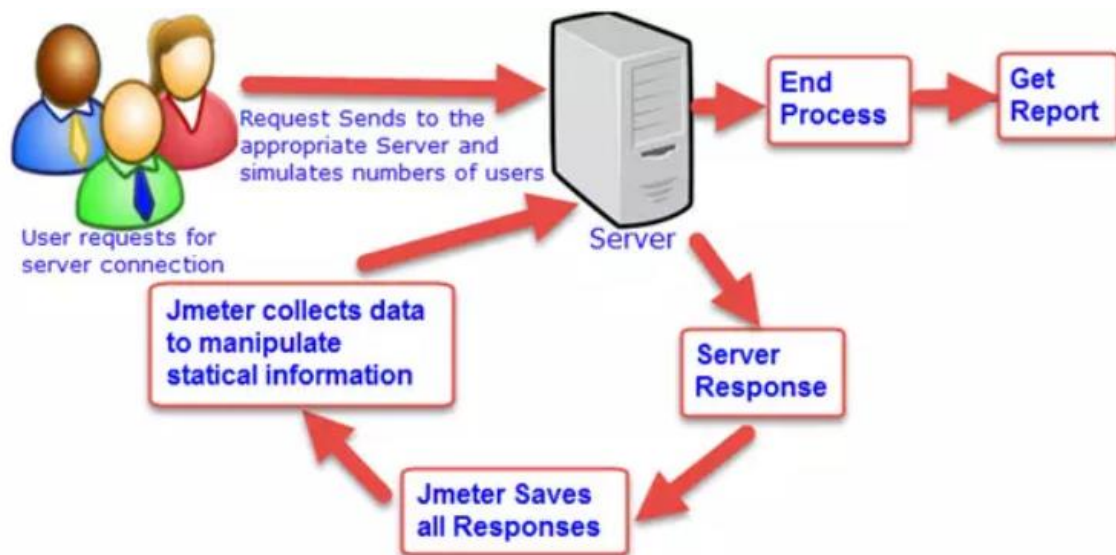
*Framework đa luồng:* Đó là một framework đa luồng đầy đủ cho phép lấy mẫu đồng thời theo nhiều luồng và lấy mẫu đồng thời các chức năng khác nhau bằng các nhóm luồng khác nhau.

*Kết quả kiểm thử dễ hiểu:* Nó trực quan hóa kết quả kiểm thử theo định dạng như biểu đồ, bảng, cây và file log rất đơn giản để hiểu.

*Dễ dàng cài đặt:* Trong Windows, chỉ cần chạy tệp “.bat” để sử dụng Jmeter. Trong Linux / Unix – Jmeter có thể được tiếp cận bằng cách nhấp vào tập lệnh shell Jmeter.

### **Quy trình hoạt động của hệ thống Jmeter**

JMeter mô phỏng số lượng người dùng gửi yêu cầu đến một máy chủ cho thấy hiệu suất / chức năng của một máy chủ / ứng dụng phù hợp thông qua các bảng, biểu đồ,...



**Hình 2.23: Quy trình hoạt động của hệ thống Jmeter**

User gửi request lên server > Request được gửi lên server với số lượng lớn > Server nhận, xử lý và phản hồi kết quả request > JMeter lưu tất cả các responses từ server > JMeter thu thập data để lấy thông tin thống kê rồi trả về server > Kết thúc chu trình làm việc > Lấy về báo cáo kết quả kiểm thử.

### **2.3.4 Tổng quan về Burp Suite**

#### **2.3.4.1 Khái niệm**

Burp Suite là một bộ công cụ kiểm thử bảo mật ứng dụng web. Nó được sử dụng để phát hiện các lỗ hổng bảo mật trong ứng dụng web bằng cách tấn công ứng dụng web và kiểm tra các phản hồi của nó.

#### **2.3.4.2 Các tính năng của Burp Suite**

Một số tính năng của Burp Suite:

-*Proxy Interceptor*: Burp Suite có thể được sử dụng như một máy chủ proxy để chuyển tiếp các yêu cầu và phản hồi giữa trình duyệt và máy chủ web. Khi hoạt động ở chế độ này, Burp Suite có thể phân tích các yêu cầu và phản hồi để phát hiện các lỗ hổng bảo mật.

-*Scanner*: Burp Suite có tính năng Scanner giúp tìm kiếm các lỗ hổng bảo mật trên ứng dụng web. Scanner có thể tìm kiếm các lỗ hổng bảo mật như XSS, SQL injection, và các lỗ hổng liên quan đến mã hóa.

-*Intruder*: Intruder là tính năng của Burp Suite cho phép thử nghiệm các yêu cầu của ứng dụng web với nhiều giá trị khác nhau để kiểm tra tính bảo mật

của ứng dụng. Intruder có thể được sử dụng để tấn công mật khẩu, tìm kiếm các lỗ hổng bảo mật khác và thử nghiệm hiệu suất ứng dụng.

-*Repeater*: Repeater cho phép kiểm tra và sửa đổi các yêu cầu và phản hồi của ứng dụng web. Điều này cho phép người dùng kiểm tra và sửa đổi các thao tác trên ứng dụng web một cách chi tiết.

-*Sequencer*: Sequencer là tính năng của Burp Suite cho phép kiểm tra chất lượng ngẫu nhiên của các chuỗi đánh giá. Nó có thể được sử dụng để kiểm tra tính bảo mật của các mã thông báo, mã bí mật hoặc mật khẩu.

-*Extensibility*: Burp Suite cho phép người dùng tạo các plugin hoặc mở rộng để thực hiện các chức năng tùy chỉnh. Người dùng có thể viết plugin để thực hiện các chức năng bổ sung như phát hiện các lỗ hổng bảo mật đặc biệt hoặc định dạng lại các yêu cầu và phản hồi.

#### 2.3.4.3 Tại sao lựa chọn Burpsuite đánh giá ứng dụng web?

-Hoàn toàn miễn phí: Burpsuite có 2 phiên bản đó là free và pro (mất phí). Bản pro sẽ có thêm chức năng scan web, nhưng với phiên bản miễn phí các bạn cũng có thể sử dụng hầu hết các chức năng chính của Burpsuite như: proxy server, web spider, intruder and repeater.

-Tính tiện lợi: Burpsuite tích hợp rất nhiều công cụ khác nhau nên người dùng sẽ tiện lợi hơn trong việc sử dụng. Không cần phải bật quá nhiều công cụ cùng một lúc bởi Burpsuite đã giúp các bạn làm được điều đó.

-Sử dụng dễ dàng: Để có thể chạy được nhiều ứng dụng Burpsuite người dùng chỉ cần cài một môi trường java sau đó click đúp vào file chạy là có thể sử dụng được. Do việc phát triển trên ngôn ngữ Java nên Burp có thiết kế giao diện vô cùng thân thiện với người dùng.

#### 2.3.4.4 Quy trình làm việc của Burp Suite

Quy trình làm việc với Burp Suite trong kiểm thử bảo mật ứng dụng web thường được chia thành các bước sau:

-*Cấu hình Burp Suite*: Đầu tiên, cần cấu hình Burp Suite để hoạt động như một máy chủ proxy để chuyển tiếp các yêu cầu và phản hồi giữa trình duyệt và máy chủ web. Để làm được điều này, ta có thể chọn menu Proxy -> Options, sau đó cấu hình các thiết lập như cổng proxy, tên miền và các tùy chọn khác.

-*Thiết lập các công cụ kiểm thử*: Tiếp theo, ta có thể thiết lập các công cụ kiểm thử như Scanner, Intruder, Repeater, Sequencer để thực hiện kiểm thử bảo

mật ứng dụng web. Các công cụ này sẽ giúp ta phát hiện các lỗ hổng bảo mật như SQL injection, Cross-site Scripting (XSS), và các lỗ hổng khác.

*-Thực hiện kiểm thử:* Sau khi đã thiết lập các công cụ kiểm thử, ta có thể thực hiện kiểm thử bảo mật ứng dụng web bằng cách sử dụng chúng. Ví dụ, ta có thể sử dụng Scanner để tự động phát hiện các lỗ hổng bảo mật, sử dụng Intruder để thử các giá trị khác nhau cho các tham số của yêu cầu, sử dụng Repeater để thực hiện lại các yêu cầu đã gửi trước đó với các thay đổi tùy ý, và sử dụng Sequencer để kiểm tra tính ngẫu nhiên của các giá trị.

*-Phân tích kết quả:* Khi đã hoàn thành các kiểm thử, ta cần phân tích kết quả được tạo ra bởi Burp Suite để tìm kiếm các lỗ hổng bảo mật trong ứng dụng web. Các kết quả này sẽ được hiển thị trong các tab của Burp Suite như Scanner, Intruder, Repeater, Sequencer.

*-Báo cáo kết quả:* Cuối cùng, ta cần tạo báo cáo về các lỗ hổng bảo mật tìm thấy và cung cấp cho nhóm phát triển để khắc phục. Burp Suite cung cấp các tùy chọn để tạo báo cáo với các định dạng khác nhau như HTML, XML, PDF.

*-Tối ưu hóa quá trình kiểm thử:* Sử dụng kết quả của các lần kiểm thử trước đó để tối ưu hóa quá trình kiểm thử

## **2.4 Tổng kết chương 2**

Ở chương 2 đã trình bày được các vấn đề cơ bản về kiểm thử tự động. Các vấn đề chính được trình bày bao gồm:

- Tổng quan về kiểm thử tự động.
- Ưu và nhược điểm của kiểm thử tự động.
- Một số công cụ hỗ trợ.
- Kiểm thử website.

-Kiểm thử tự động đang được quan tâm như một giải pháp hiệu quả và duy nhất nhằm cải thiện tính chính xác và hiệu quả cũng như giảm chi phí, rút ngắn thời gian trong quá trình kiểm thử các sản phẩm phần mềm.

## CHƯƠNG 3. THỰC NGHIỆM

### 3.1 Bài toán

Vấn đề đặt ra là kiểm thử chức năng cơ bản cho trang Web tự tạo và Facebook là chức năng đăng ký, đăng nhập.

Ứng dụng được kiểm thử trên trình duyệt: Microsoft Edge

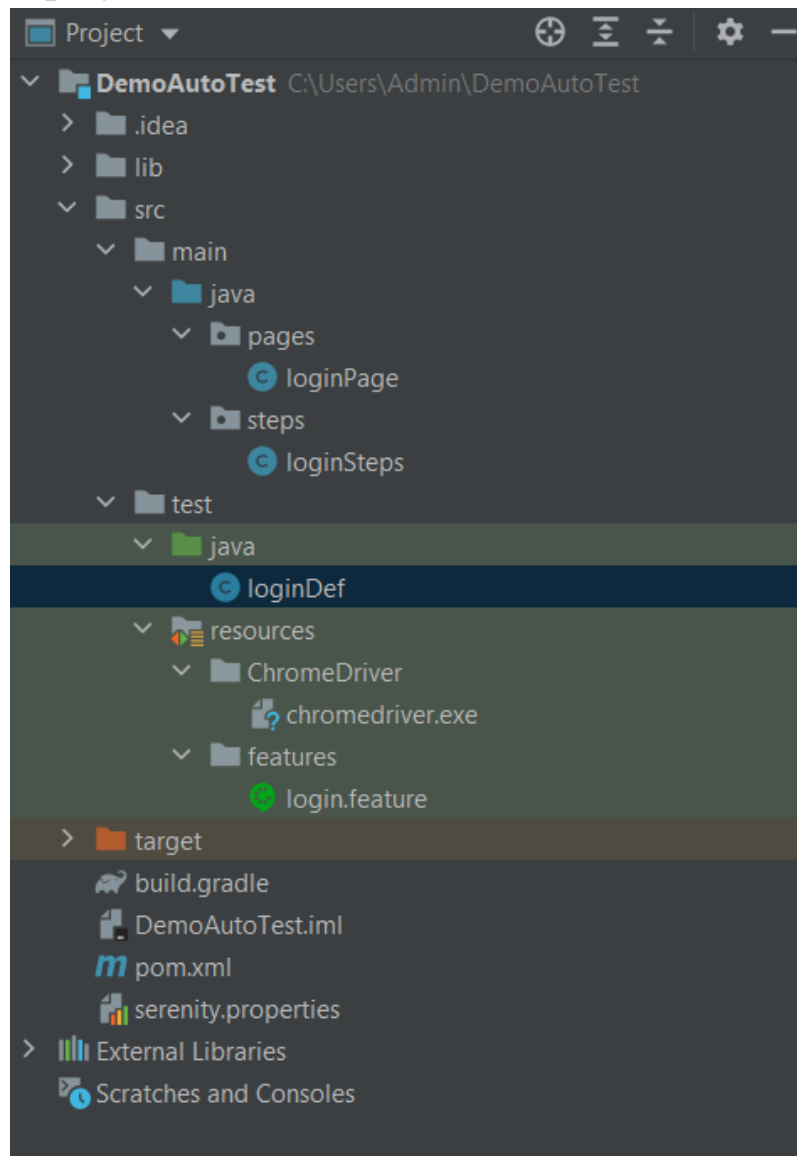
Trên phương diện là một người sử dụng, chúng em quan tâm tới việc tạo tài khoản và đăng nhập có thành công hay không.

Các yêu cầu cần kiểm tra:

-Đăng ký và đăng nhập với email, password hợp lệ và không hợp lệ.



### 3.2 Cấu trúc của 1 project Automation test



Hình 3.1 Cấu trúc của 1 project Automation

#### Features

- Bao gồm tất cả các feature nhỏ.
- Trong mỗi feature có những Scenario.
- Mỗi Scenario thể hiện 1 testcase, 1 mục tiêu cụ thể.
- Một Scenario bao gồm: Pre-conditions, step thực hiện và kết quả Step Definitions.
- Khi chạy file '.feature', mỗi step sẽ khớp với một Ruby code block được định nghĩa trước đó gọi là Step Definitions.

-Cucumber không thể biết làm thế nào để thực thi được Scenario. Nó cần Step Definition để biên dịch nguyên văn các bước Gherkin thành các hành động cái mà có thể tương tác với hệ thống.

-Khi Cucumber thực thi các Step trong Scenario nó sẽ tìm kiếm các Step Definition phù hợp để thực thi.

-Một Step Definition là một phần nhỏ của code với một pattern đính kèm.

-Pattern được sử dụng để liên kết các Step Definition với tất cả các Step phù hợp và code là cái mà Cucumber sẽ thực thi khi thấy Step.

### **Steps file**

Viết các step được call ở file Def vào file Steps bằng ngôn ngữ Java. Các step đó có thể chứa nhiều step, action nhỏ hơn.

### **Page file**

Viết file Pages bằng ngôn ngữ Java, định nghĩa từ action nhỏ nhất được call ở file Steps, chúng tương tác trực tiếp với Web Element.

## **3.3 Xác định các ca kiểm thử**

### **3.3.1 Chức năng đăng ký**

- Email không hợp lệ.
- Email nhập lại không giống.
- Email đã được sử dụng.
- Email trống.
- Password trống.
- Password không đạt yêu cầu.

### **3.3.2 Chức năng đăng nhập**

- Đăng nhập đúng.
- Email sai.
- Password sai.
- Email trống.
- Password trống.

## **3.4 Kiểm thử Facebook**

Facebook là một mạng xã hội trực tuyến được thành lập bởi Mark Zuckerberg vào năm 2004. Nó được phát triển như là một nền tảng để kết nối mọi

người với nhau, chia sẻ thông tin và tương tác trực tuyến. Từ đó, Facebook đã trở thành một trong những trang web được sử dụng nhiều nhất trên thế giới, với hàng tỷ người dùng đăng ký.

Facebook cho phép người dùng tạo hồ sơ cá nhân, kết bạn với những người khác và tham gia vào các nhóm và trang quan tâm. Người dùng có thể đăng thông tin cá nhân, chia sẻ hình ảnh, video và bài viết, đồng thời có thể thích, bình luận và chia sẻ nội dung của người khác. Nền tảng này cũng cung cấp các công cụ để kết nối doanh nghiệp với khách hàng và quảng cáo trực tuyến.

Facebook đã mở rộng và sở hữu nhiều dịch vụ và ứng dụng khác nhau. Các dịch vụ phổ biến khác của Facebook bao gồm Messenger, một ứng dụng nhắn tin riêng biệt; Instagram, một mạng xã hội chia sẻ hình ảnh và video; WhatsApp, một ứng dụng nhắn tin và gọi điện thoại; và Oculus, công ty sản xuất thiết bị thực tế ảo.

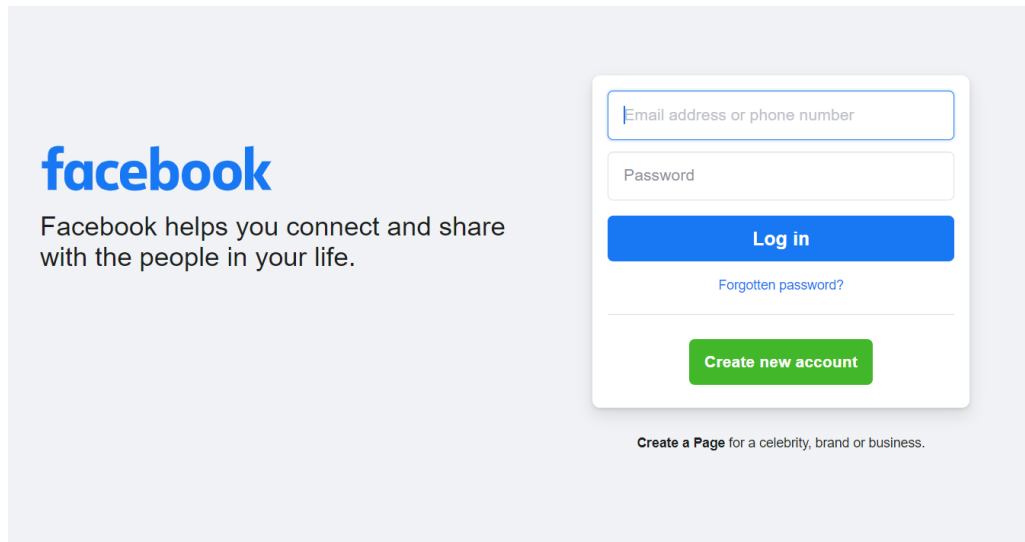
Tuy nhiên, Facebook cũng đã gặp phải nhiều tranh cãi về quyền riêng tư và bảo mật dữ liệu, cũng như ảnh hưởng của nền tảng này đến việc truyền thông và chính trị. Ngoài ra, Facebook cũng đã đối mặt với các vụ vi phạm pháp luật và sự chỉ trích về cách quản lý nội dung trên nền tảng của mình.

Dù vậy, Facebook vẫn tiếp tục là một trong những mạng xã hội lớn nhất và quan trọng nhất trên thế giới, ảnh hưởng đến cuộc sống và giao tiếp của hàng tỷ người dùng trên khắp thế giới.

### 3.4.1 Xác định các bước kiểm thử

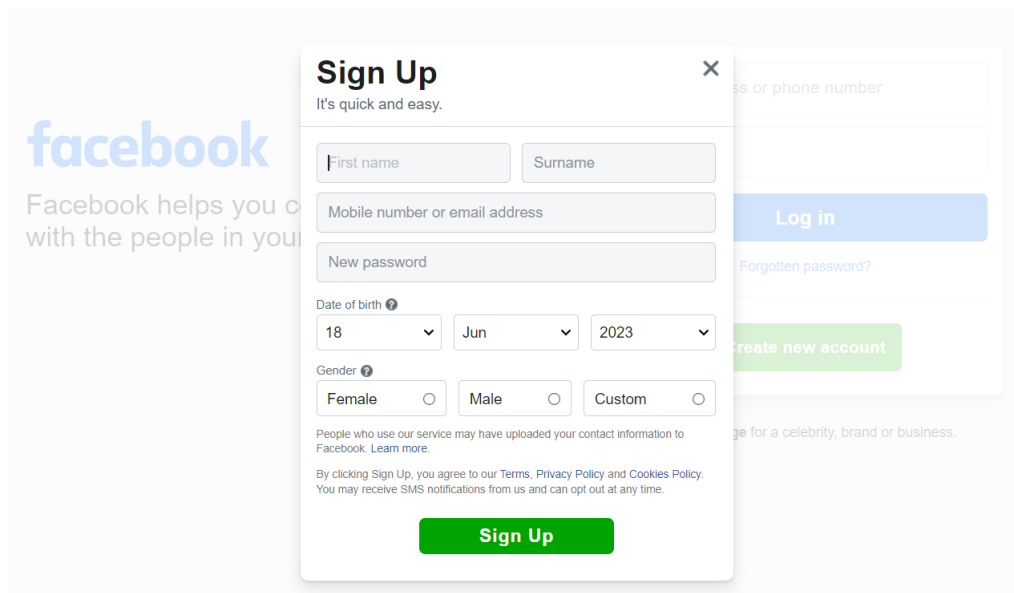
a) Chức năng đăng ký:

Bước 1: Mở trang chủ Facebook



Hình 3.2: Giao diện Facebook

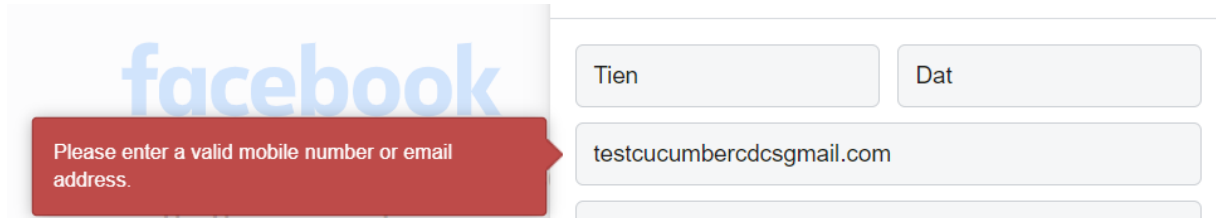
Bước 2: Đăng ký



Hình 3.3: Màn hình trang đăng ký

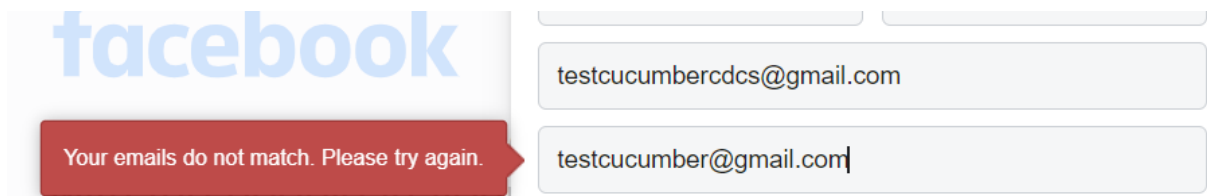
### Bước 3: Kiểm tra kết quả

-Trường hợp 1: Email không hợp lệ



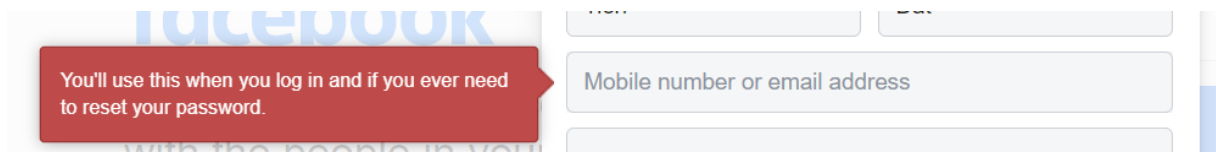
**Hình 3.4: Màn hình email không hợp lệ**

-Trường hợp 2: Email nhập lại không trùng khớp.



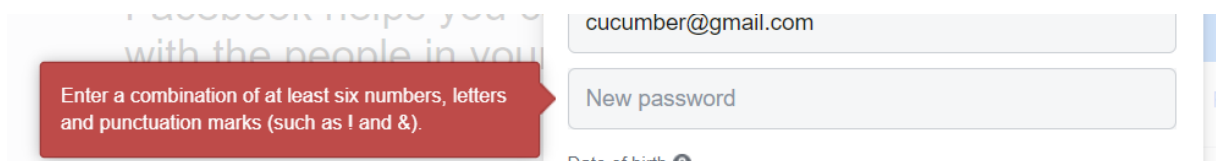
**Hình 3.5: Màn hình email không trùng khớp**

-Trường hợp 3: Email trống.



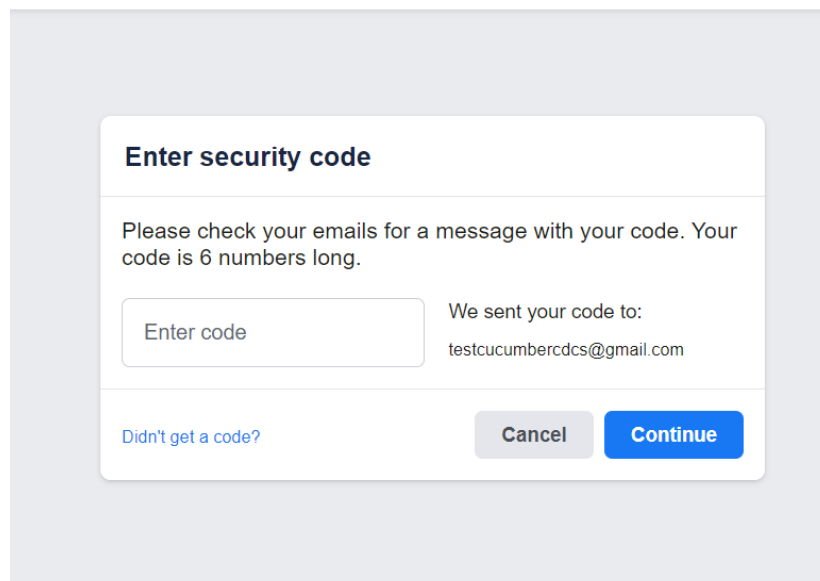
**Hình 3.6 Màn hình email trống**

-Trường hợp 4: Password trống.



**Hình 3.7 Màn hình password trống**

-Trường hợp 5: Email đã được sử dụng

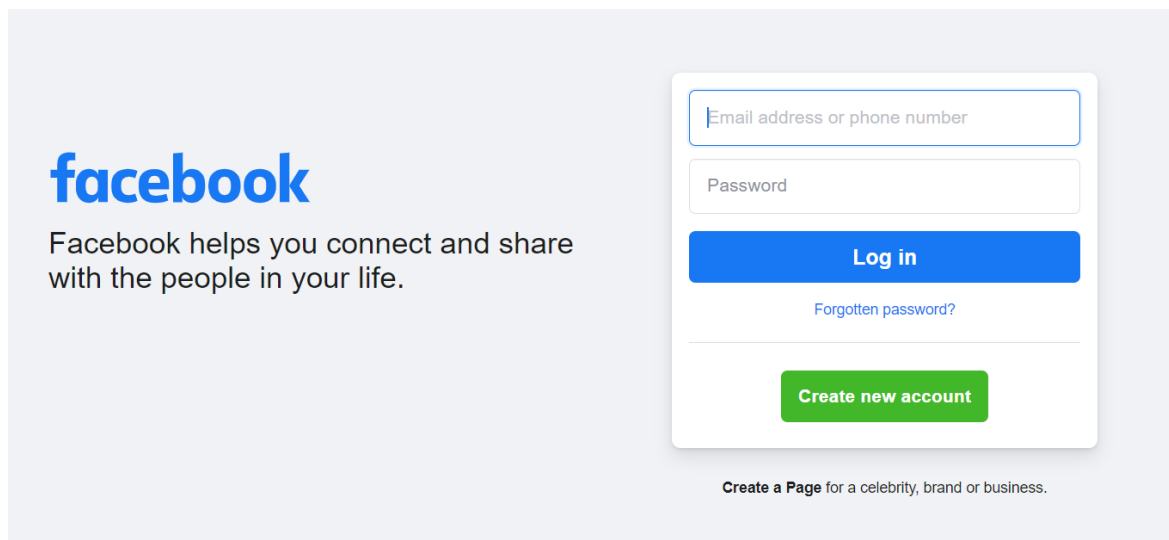


The screenshot shows a white modal box titled "Enter security code" on a light gray background. Inside the box, the text reads: "Please check your emails for a message with your code. Your code is 6 numbers long." Below this is a text input field with the placeholder "Enter code". To the right of the input field, it says "We sent your code to:" followed by the email address "testcucumberdc@gmail.com". At the bottom left of the modal is a link "Didn't get a code?". At the bottom right are two buttons: a gray "Cancel" button and a blue "Continue" button.

**Hình 3.8 Màn hình email đã được sử dụng**

b) Chức năng đăng nhập

Bước 1: Mở trang chủ Facebook



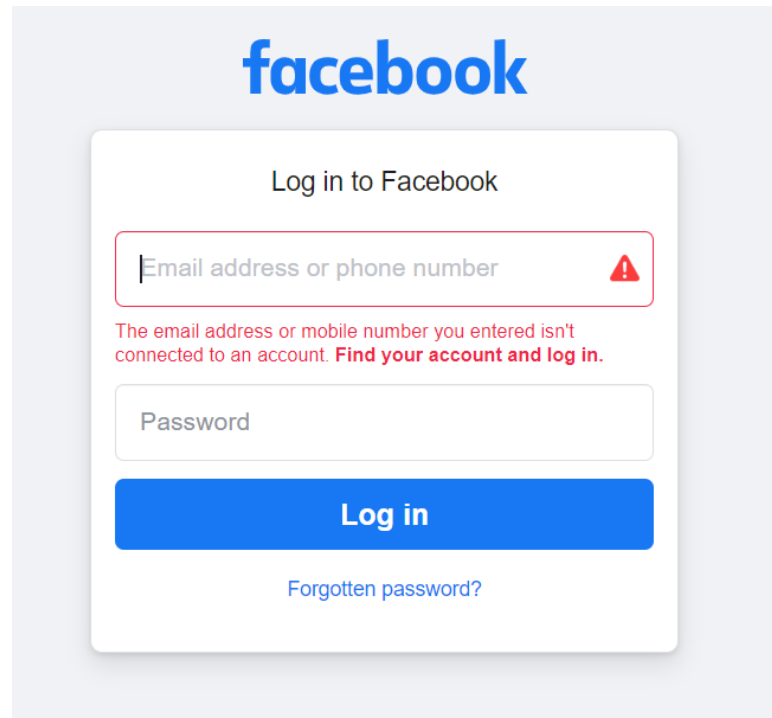
The screenshot shows the Facebook login page. On the left, the Facebook logo is displayed above the text "Facebook helps you connect and share with the people in your life." On the right, there is a white login box. It contains two input fields: "Email address or phone number" and "Password". Below these fields is a blue "Log in" button. Under the "Log in" button is a link "Forgotten password?". At the bottom of the login box is a green "Create new account" button. Below the login box, there is a link "Create a Page for a celebrity, brand or business."

**Hình 3.9 Màn hình facebook**

Bước 2: Đăng ký

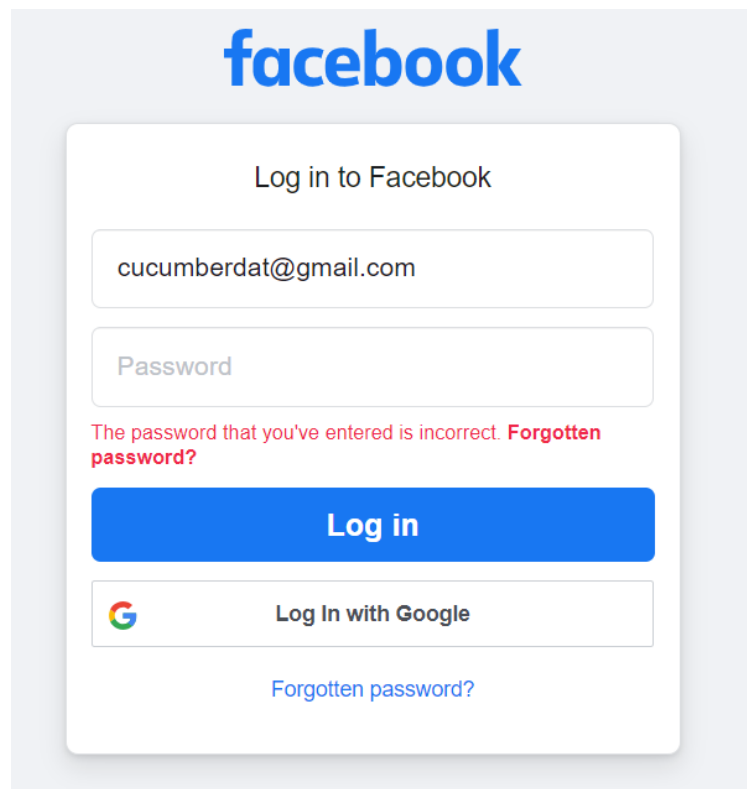
Bước 3: Kiểm tra kết quả

-Trường hợp Email sai.



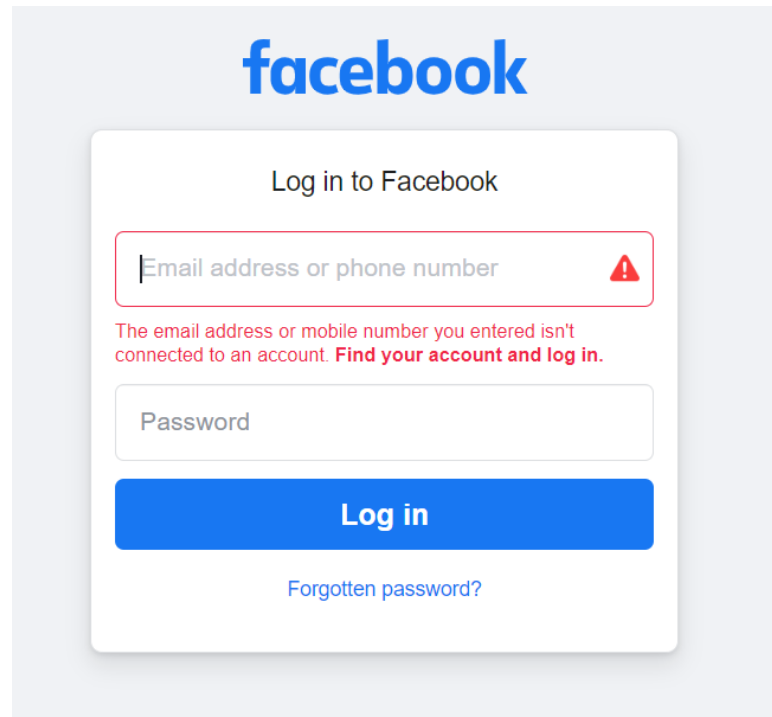
**Hình 3.10 Màn hình lỗi Email sai**

-Trường hợp mật khẩu sai.



**Hình 3.11 Màn hình lỗi mật khẩu sai**

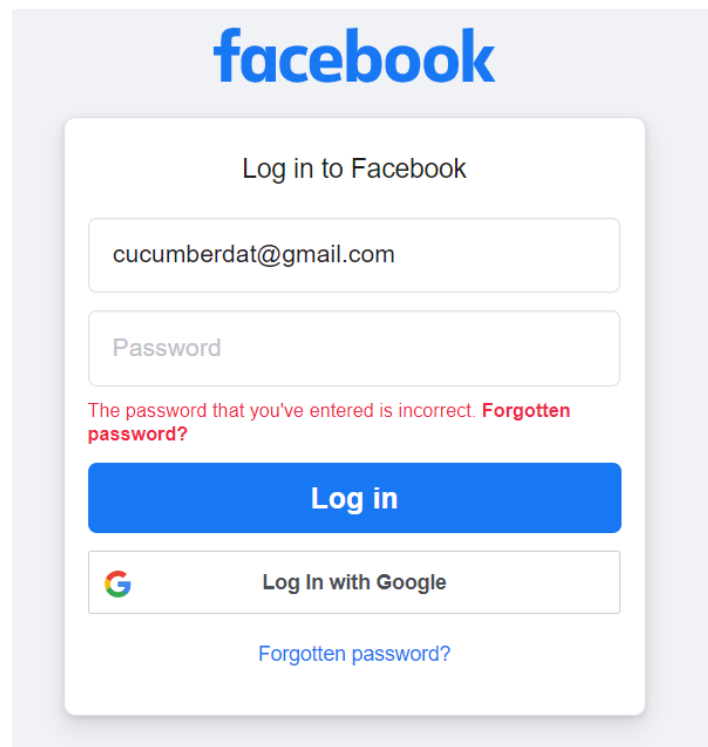
-Trường hợp Email trống.



The image shows the Facebook login interface. At the top is the Facebook logo. Below it is the text "Log in to Facebook". There are two input fields: "Email address or phone number" and "Password". The "Email address or phone number" field is highlighted with a red border and a red warning icon. Below this field, a red error message reads: "The email address or mobile number you entered isn't connected to an account. Find your account and log in." The "Log in" button is blue and located below the password field. A link for "Forgotten password?" is at the bottom.

**Hình 3.12 Màn hình lỗi Email trống**

-Trường hợp mật khẩu trống.

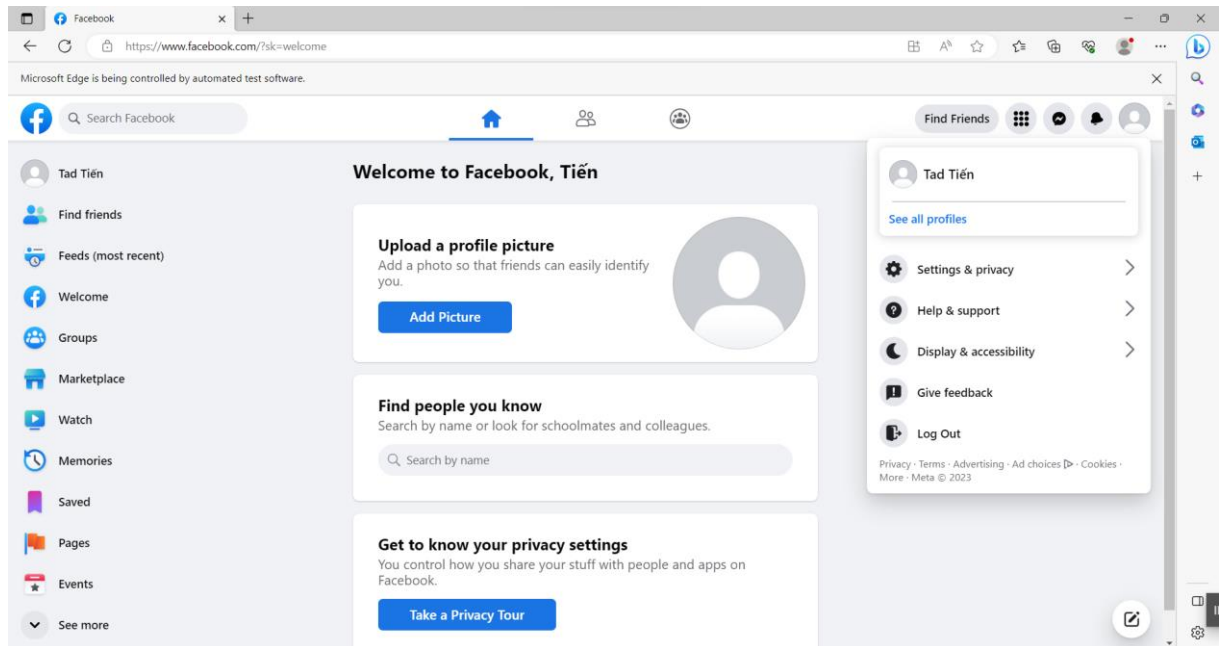


The image shows the Facebook login interface. At the top is the Facebook logo. Below it is the text "Log in to Facebook". There are two input fields: "Email address or phone number" and "Password". The "Password" field is highlighted with a red border and a red warning icon. Below this field, a red error message reads: "The password that you've entered is incorrect. Forgotten password?". The "Log in" button is blue and located below the password field. A link for "Forgotten password?" is at the bottom. Below the "Log in" button is a "Log In with Google" button with the Google logo.

**Hình 3.13 Màn hình lỗi mật khẩu trống**



-Trường hợp đăng nhập đúng.



Hình 3.14 Màn hình đăng nhập thành công

### 3.4.2 Thực nghiệm kiểm thử tự động

#### a) Demo chức năng đăng ký

Các trường hợp kiểm thử chính:

```
Scenario Outline: Dang ky sai
  Given mo trang dang ky facebook
  When dienthong tin voi "<Mobile number or email address>" va "<New password>" va "<email2>"
  And dang ky
  And kiemtra voi "<Loi>"
  Examples:
    | Mobile number or email address | email2 | New password | Loi
    | testcucumbercdcs@gmail.com | | SubAXvSW-yQnT7h | Please enter a valid mobile number or email
    | testcucumbercdcs@gmail.com | testcucumber@gmail.com | SubAXvSW | Your emails do not match. Please try again.
    | | | SubAXvSW-yQnT7h | You'll use this when you log in and if you e
    | cucumber@gmail.com | cucumber@gmail.com | | Enter a combination of at least six numbers,
```

Hình 3.15 Scenario Outline đăng ký sai

```
Scenario Outline: Dang ky email trung
  Given motrangdang ky facebook
  When dienthongtinvoi "<Mobile number or email address>" va "<New password>" va "<email2>"
  And dang ky2
  And kiemtra2 voi "<Loi>"
  Examples:
    | Mobile number or email address | email2 | New password | Loi
    | testcucumbercdcs@gmail.com | testcucumbercdcs@gmail.com | SubAXvSW | Please check your emails for a message with your code.
```

Hình 3.16 Scenario Outline đăng ký Email đã sử dụng

Kết quả từng trường hợp thử nghiệm:

- Trường hợp Email không hợp lệ:

|   |              |
|---|--------------|
| ✓ mo trang dang ky facebook   | 5 sec 501 ms |
| ✓ dienthong tin voi "testcumbercdcs@gmail.com" va "5ubAXvSW-yQnT7h" va "" | 1 sec 238 ms |
| ✓ dang ky   | 47 ms        |
| ✓kiemtra voi "Please enter a valid mobile number or email address."       | 242 ms       |

**Hình 3.17 Trường hợp Email không hợp lệ**

B1: Nhập Email không hợp lệ “testcumbercdcs@gmail.com”

B2: Đăng ký và kiểm tra lỗi “Please enter a valid mobile number or email address.” có hiển thị lên màn hình hay không.

- Trường hợp Email nhập lại không trùng khớp:

|  |              |
|--|--------------|
| ✓ mo trang dang ky facebook  | 4 sec 478 ms |
| ✓ dienthong tin voi "testcumbercdcs@gmail.com" va "5ubAXvSW" va "testcumber@gmail.com" | 1 sec 201 ms |
| ✓ dang ky  | 65 ms        |
| ✓kiemtra voi "Your emails do not match. Please try again."                             | 266 ms       |

**Hình 3.18 Trường hợp Email nhập lại không trùng khớp**

B1: Nhập Email nhập lại không khớp “testcumber@gmail.com”

B2: Đăng ký và kiểm tra lỗi “Your emails do not match. Please try again.” có hiển thị lên màn hình hay không.

- Trường hợp Email trống:

|   |              |
|---|--------------|
| ✓ mo trang dang ky facebook   | 4 sec 277 ms |
| ✓ dienthong tin voi "" va "5ubAXvSW-yQnT7h" va ""   | 1 sec 693 ms |
| ✓ dang ky   | 91 ms        |
| ✓kiemtra voi "You'll use this when you log in and if you ever need to reset your password." | 95 ms        |

**Hình 3.19 Trường hợp Email trống**

B1: Để Email trống

B2: Đăng ký và kiểm tra lỗi “You'll use this when you log in and if you ever need to reset your password.” có hiển thị lên màn hình hay không.

- Trường hợp mật khẩu trống:

|  |              |
|--|--------------|
| ✓ mo trang dang ky facebook  | 4 sec 355 ms |
| ✓ dienthong tin voi "cucumber@gmail.com" va "" va "cucumber@gmail.com"   | 2 sec 197 ms |
| ✓ dang ky  | 96 ms        |
| ✓ kiểm tra voi "Enter a combination of at least six numbers, letters and punctuation marks (such as ! and &)." | 42 ms        |

**Hình 3.20: Trường hợp mật khẩu trống**

B1: Để mật khẩu trống

B2: Đăng ký và kiểm tra lỗi “Enter a combination of at least six numbers, letters and punctuation marks (such as ! and &).” có hiển thị lên màn hình hay không.

- Trường hợp Email đã được sử dụng:

|   |              |
|---|--------------|
| ✓ motrangdang ky facebook   | 5 sec 429 ms |
| ✓ dienthongtinvoi "testcumbercdcs@gmail.com" va "5ubAXvSW" va "testcumbercdcs@gmail.com"              | 1 sec 527 ms |
| ✓ dang ky2  | 87 ms        |
| ✓ kiểm tra2 voi "Please check your emails for a message with your code. Your code is 6 numbers long." | 4 sec 96 ms  |

**Hình 3.21 Trường hợp Email đã được sử dụng**

B1: Nhập Email đã được sử dụng

B2: Đăng ký và kiểm tra lỗi “Please check your emails for a message with your code. Your code is 6 numbers long.” có hiển thị lên màn hình hay không.

## b) Demo chức năng đăng nhập

Các trường hợp kiểm thử chính:

| Scenario Outline: Dang nhap sai                            |           |          |   |
|--|-----------|----------|---|
| Given mo trang facebook                                    |           |          |   |
| When dien thong tin dang nhap voi "<email>" va "<matkhau>" |           |          |   |
| And dang nhap  |           |          |   |
| And kiểm tra "<thongbao>"                                  |           |          |   |
| Examples:  |           |          |   |
| email  | matkhau   | thongbao |   |
| cucumberdat@gmail.com                                      | 654321    |          | The password that you've entered is incorrect. Forgotten password?            |
| cucumbercdcs@gmail.com                                     | 654321@Aa |          | The email address or mobile number you entered isn't connected to an account. |
|  | 654321@Aa |          | The email address or mobile number you entered isn't connected to an account. |
| cucumberdat@gmail.com                                      |           |          | The password that you've entered is incorrect. Forgotten password?            |

**Hình 3.22 Scenario Outline đăng nhập sai**

```

Scenario Outline: Dang nhap dung
    Given mo trang fbook
    When dien thong tin voi "<email>" va "<matkhau>"
    And dangnhap
    And kiem tra
    Examples:
        | email | matkhau |
        | cucumberdat@gmail.com | 654321@Aa |

```

**Hình 3.23 Scenario Outline đăng nhập đúng**

Kết quả từng trường hợp thử nghiệm:

-Trường hợp mật khẩu sai:

|   |              |
|---|--------------|
| ✓ mo trang facebook   | 6 sec 829 ms |
| ✓ dien thong tin dang nhap voi "cucumberdat@gmail.com" va "654321"              | 385 ms       |
| ✓ dang nhap   | 3 sec 158 ms |
| ✓ kiem tra "The password that you've entered is incorrect. Forgotten password?" | 1 sec 766 ms |

**Hình 3.24 Trường hợp mật khẩu sai**

B1: Nhập Email "[cucumberdat@gmail.com](mailto:cucumberdat@gmail.com)" và mật khẩu sai "654321"

B2: Đăng nhập và kiểm tra lỗi "The password that you've entered is incorrect. Forgotten password?" có hiện hay không.

-Trường hợp Email sai:

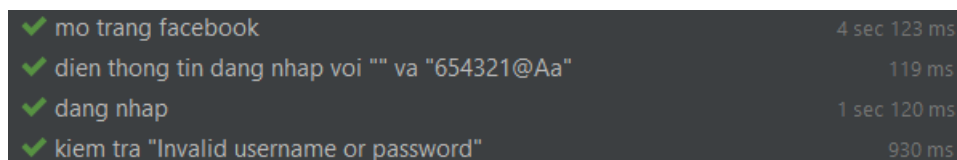
|   |              |
|---|--------------|
| ✓ mo trang facebook   | 5 sec 817 ms |
| ✓ dien thong tin dang nhap voi "cucumbercdcs@gmail.com" va "654321@Aa"              | 361 ms       |
| ✓ dang nhap   | 2 sec 928 ms |
| ✓ kiem tra "The email address or mobile number you entered isn't connected to an a" | 1 sec 400 ms |

**Hình 3.25 Trường hợp Email sai**

B1: Nhập Email "[cucumbercdcs@gmail.com](mailto:cucumbercdcs@gmail.com)"

B2: Đăng nhập và kiểm tra lỗi "The email address or mobile number you entered isn't connected to an account. Find your account and log in." có hiện hay không.

-Trường hợp Email trống:



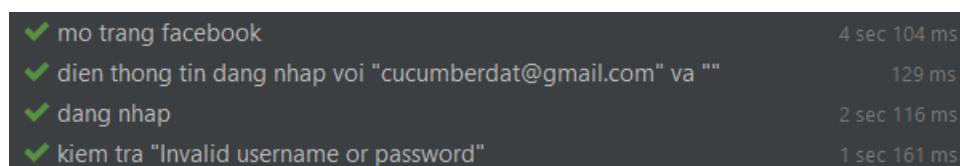
|  |              |
|--|--------------|
| ✓ mo trang facebook                              | 4 sec 123 ms |
| ✓ dien thong tin dang nhap voi "" va "654321@Aa" | 119 ms       |
| ✓ dang nhap                                      | 1 sec 120 ms |
| ✓ kiem tra "Invalid username or password"        | 930 ms       |

**Hình 3.26 Trường hợp Email trống**

B1: Để trống Email

B2: Đăng nhập và kiểm tra lỗi “The email address or mobile number you entered isn't connected to an account. Find your account and log in.” có hiện hay không.

-Trường hợp mật khẩu trống:



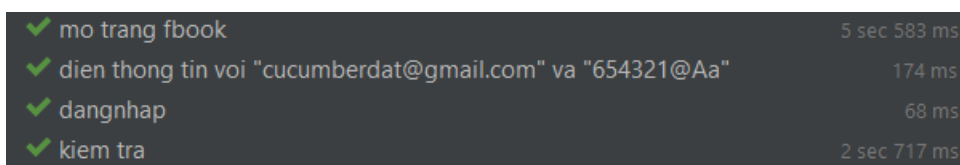
|  |              |
|--|--------------|
| ✓ mo trang facebook  | 4 sec 104 ms |
| ✓ dien thong tin dang nhap voi "cucumberdat@gmail.com" va "" | 129 ms       |
| ✓ dang nhap  | 2 sec 116 ms |
| ✓ kiem tra "Invalid username or password"                    | 1 sec 161 ms |

**Hình 3.27 Trường hợp mật khẩu trống**

B1: Để trống mật khẩu.

B2: Đăng nhập và kiểm tra lỗi “The password that you've entered is incorrect. Forgotten password?” có hiện hay không.

-Trường hợp đăng nhập đúng:



|   |              |
|---|--------------|
| ✓ mo trang fbook  | 5 sec 583 ms |
| ✓ dien thong tin voi "cucumberdat@gmail.com" va "654321@Aa" | 174 ms       |
| ✓ dangnhap  | 68 ms        |
| ✓ kiem tra  | 2 sec 717 ms |

**Hình 3.28 Trường hợp đăng nhập đúng**

B1: Nhập thông tin chính xác

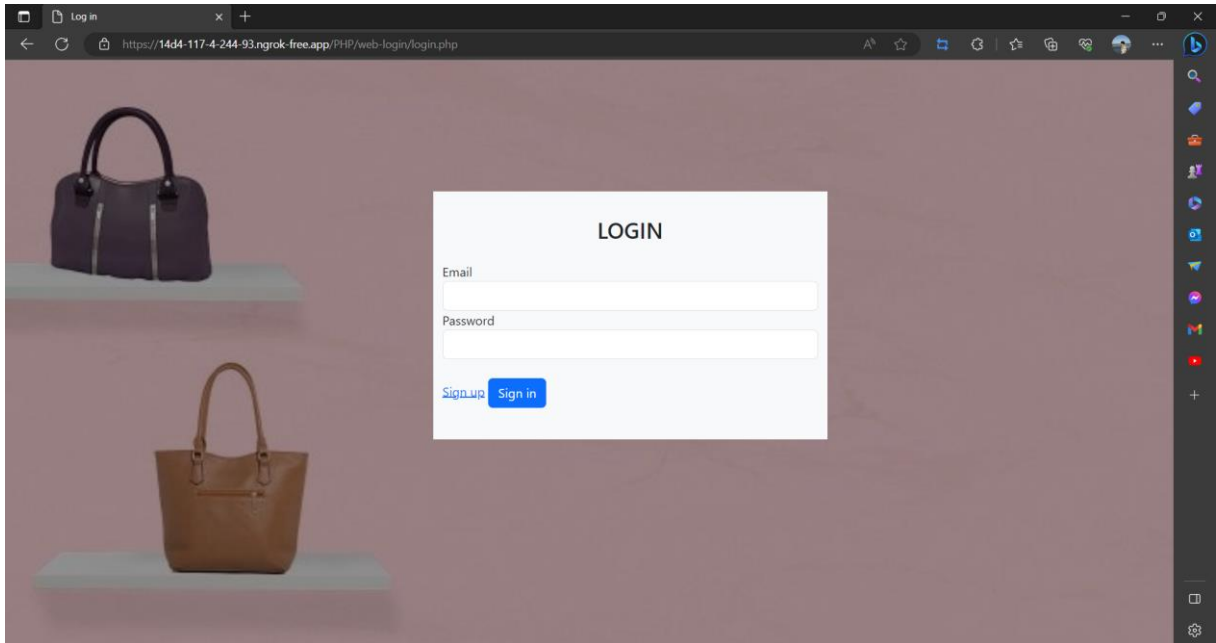
B2: Đăng nhập và kiểm tra có đăng nhập được vào trang web hay không.

## 3.5 Kiểm thử Web tự tạo

### 3.5.1 Xác định các bước kiểm thử

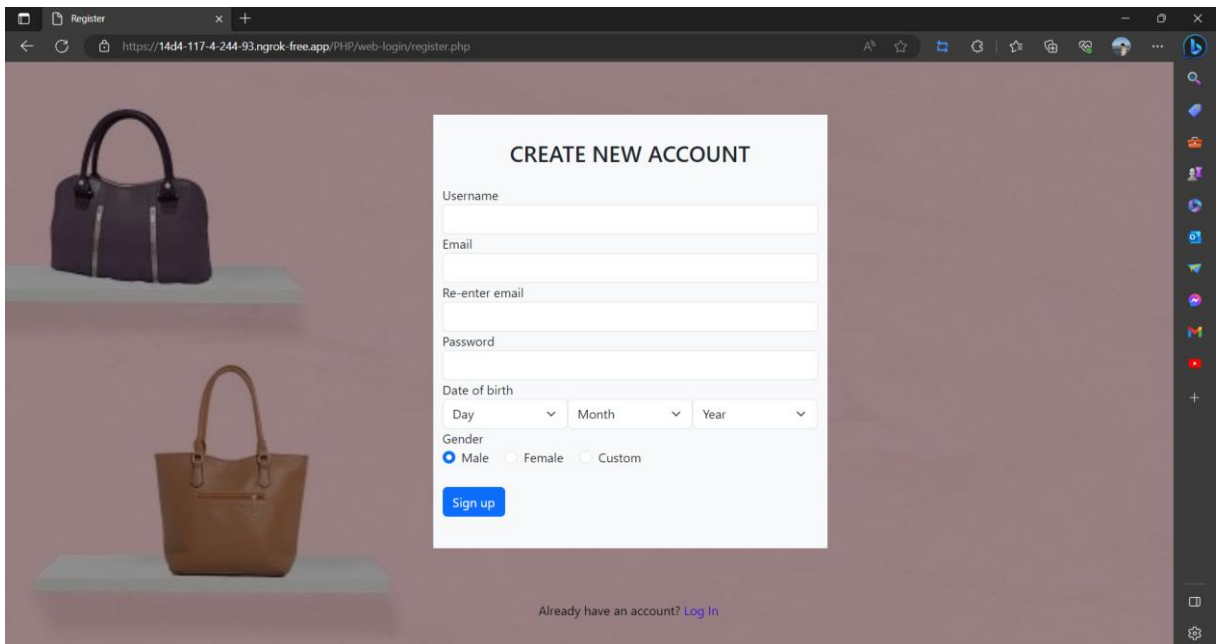
#### a) Chức năng đăng ký

##### Bước 1: Mở trang web



Hình 3.29 Giao diện trang web

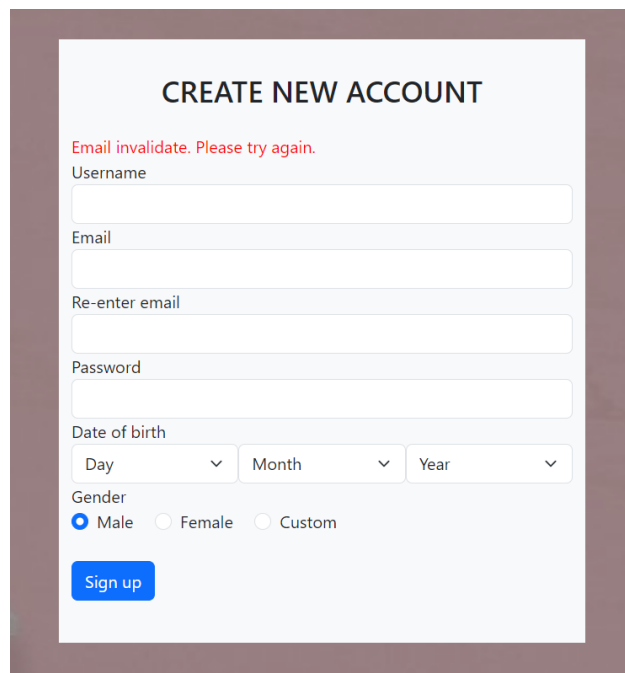
##### Bước 2: Đăng ký



Hình 3.30 Giao diện đăng ký

### Bước 3: Kiểm tra

#### -Trường hợp 1: Email không hợp lệ



**CREATE NEW ACCOUNT**

Email invalidate. Please try again.

Username

Email

Re-enter email

Password

Date of birth

Day Month Year

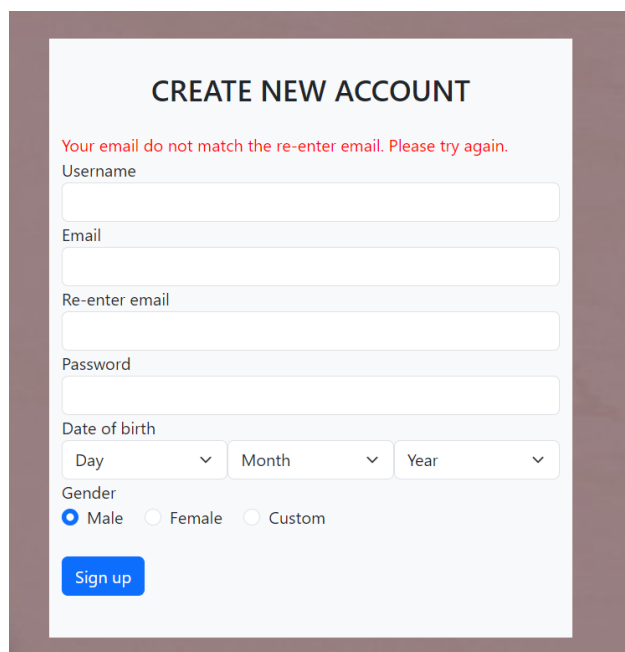
Gender

☒ Male ☐ Female ☐ Custom

Sign up

**Hình 3.31** Giao diện đăng ký với Email không hợp lệ

#### -Trường hợp 2: Email nhập lại không trùng khớp



**CREATE NEW ACCOUNT**

Your email do not match the re-enter email. Please try again.

Username

Email

Re-enter email

Password

Date of birth

Day Month Year

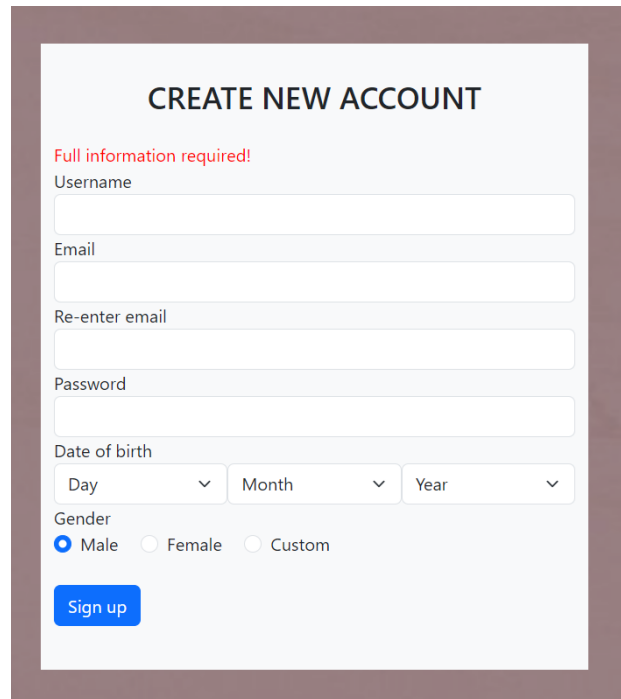
Gender

☒ Male ☐ Female ☐ Custom

Sign up

**Hình 3.32** Giao diện đăng ký với Email nhập lại không trùng khớp

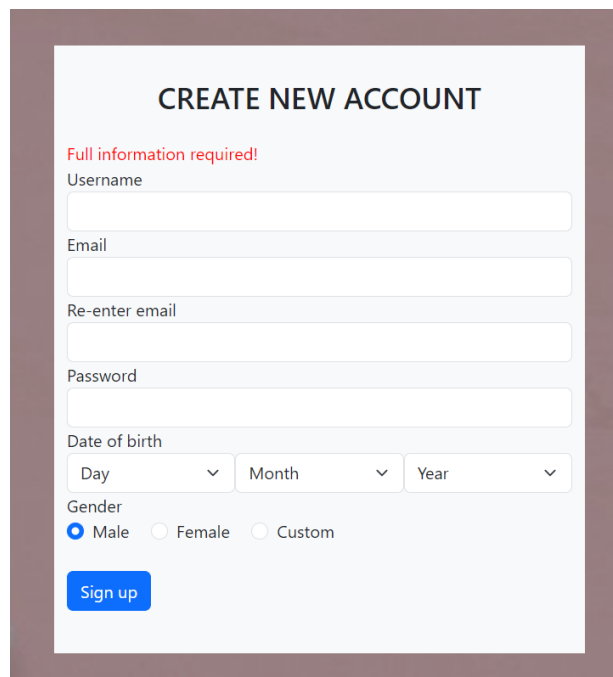
-Trường hợp 3: Email trống



The screenshot shows a 'CREATE NEW ACCOUNT' form. At the top, the title 'CREATE NEW ACCOUNT' is centered. Below it, a red error message 'Full information required!' is displayed. The form contains several input fields: 'Username', 'Email', 'Re-enter email', and 'Password'. The 'Email' field is highlighted with a red border, indicating it is the source of the error. Below the 'Email' field, there are three dropdown menus for 'Date of birth' (Day, Month, Year). At the bottom, there are radio buttons for 'Gender' (Male, Female, Custom) and a blue 'Sign up' button.

**Hình 3.33** Giao diện đăng ký với Email trống

-Trường hợp 4: Mật khẩu trống

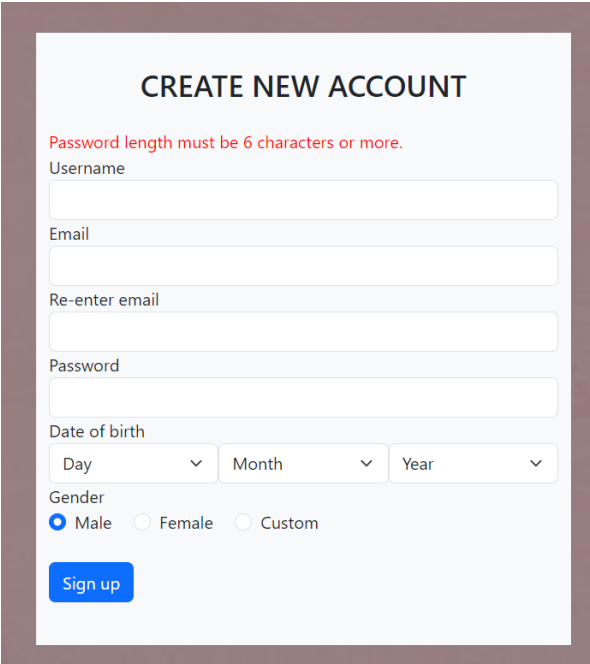


The screenshot shows the same 'CREATE NEW ACCOUNT' form as in Figure 3.33. The red error message 'Full information required!' is still present. In this case, the 'Password' field is highlighted with a red border, indicating it is the source of the error. All other fields and elements, including the 'Email' field, are in their default state.

**Hình 3.34** Giao diện đăng ký với mật khẩu trống



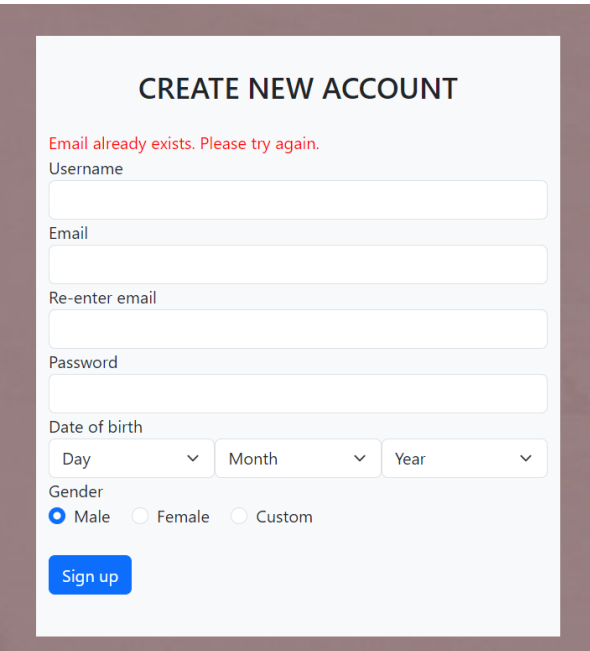
-Trường hợp 5: Mật khẩu không đạt yêu cầu



The screenshot shows a web form titled "CREATE NEW ACCOUNT". At the top, a red error message states: "Password length must be 6 characters or more." Below this, the form contains several input fields: "Username", "Email", "Re-enter email", and "Password". The "Password" field is currently empty. Below the password field are three dropdown menus for "Date of birth" labeled "Day", "Month", and "Year". At the bottom of the form, there is a "Gender" section with three radio buttons: "Male" (which is selected), "Female", and "Custom". A blue "Sign up" button is located at the bottom left of the form area.

Hình 3.35 Giao diện đăng ký với mật khẩu không đạt yêu cầu

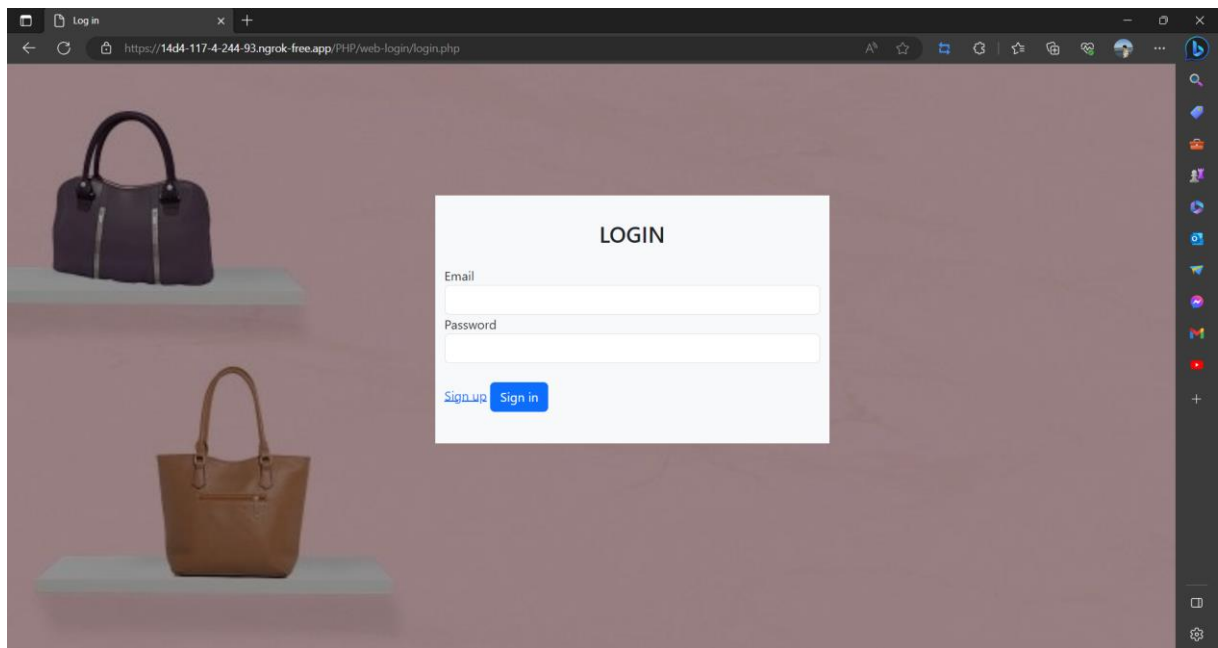
-Trường hợp 6: Email đã được sử dụng



This screenshot shows the same "CREATE NEW ACCOUNT" form as in the previous image. In this instance, a red error message at the top states: "Email already exists. Please try again." The "Email" and "Re-enter email" fields are filled with the same text. The "Password" field remains empty. The "Date of birth" dropdowns and the "Gender" section (with "Male" selected) are also visible. The blue "Sign up" button is at the bottom left.

Hình 3.36 Giao diện đăng ký với Email đã được sử dụng

b) Chức năng đăng nhập  
Bước 1: Mở trang web

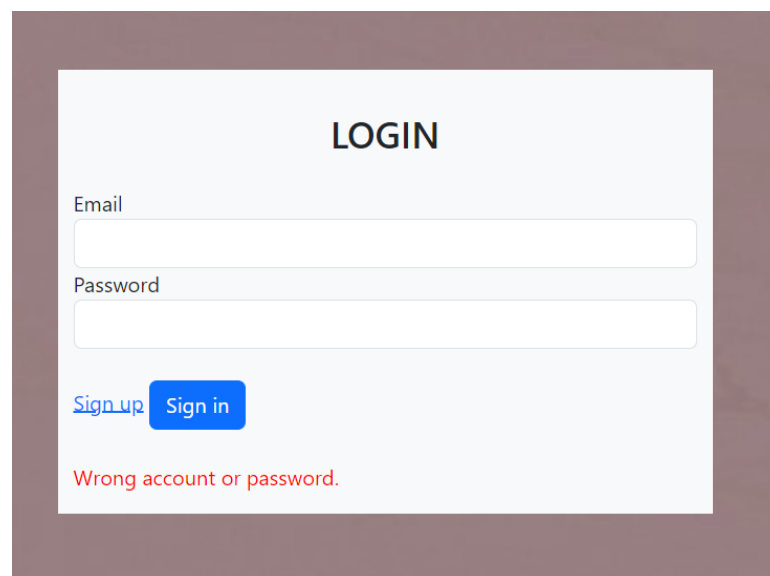


Hình 3.37 Giao diện trang Web

Bước 2: Đăng nhập

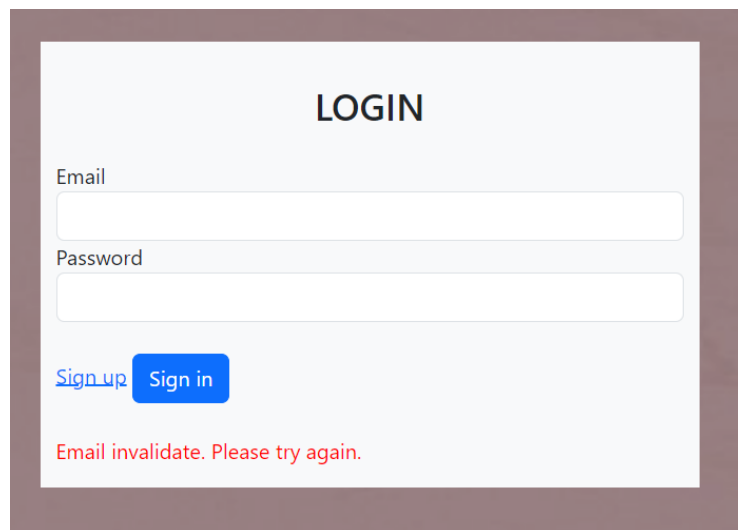
Bước 3: Kiểm tra

-Trường hợp 1: Mật khẩu sai



Hình 3.38 Giao diện đăng nhập với mật khẩu sai

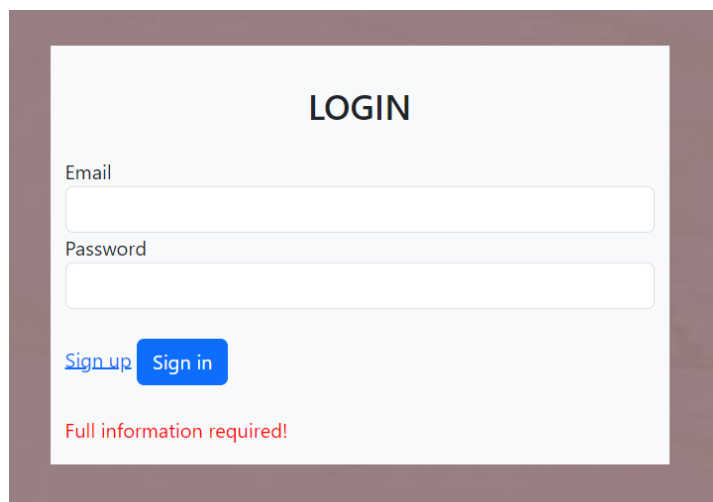
-Trường hợp 2: Email sai



The screenshot shows a login interface with the title "LOGIN" at the top. Below the title are two input fields: "Email" and "Password". The "Email" field is empty, and the "Password" field is also empty. Below the input fields, there is a link "Sign up" and a blue button "Sign in". At the bottom, a red error message reads "Email invalidate. Please try again."

**Hình 3.39** Giao diện đăng nhập với Email không hợp lệ

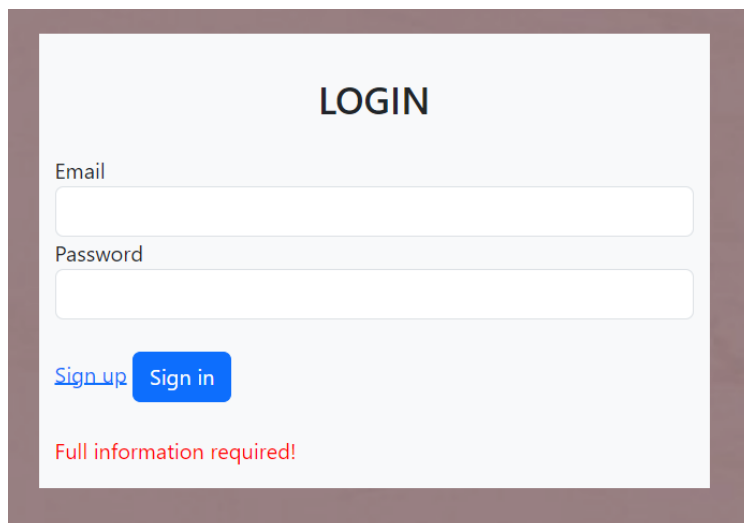
-Trường hợp 3: Mật khẩu trống



The screenshot shows a login interface with the title "LOGIN" at the top. Below the title are two input fields: "Email" and "Password". The "Email" field is empty, and the "Password" field is also empty. Below the input fields, there is a link "Sign up" and a blue button "Sign in". At the bottom, a red error message reads "Full information required!"

**Hình 3.40** Giao diện đăng nhập với mật khẩu trống

-Trường hợp 4: Email trống



LOGIN

Email

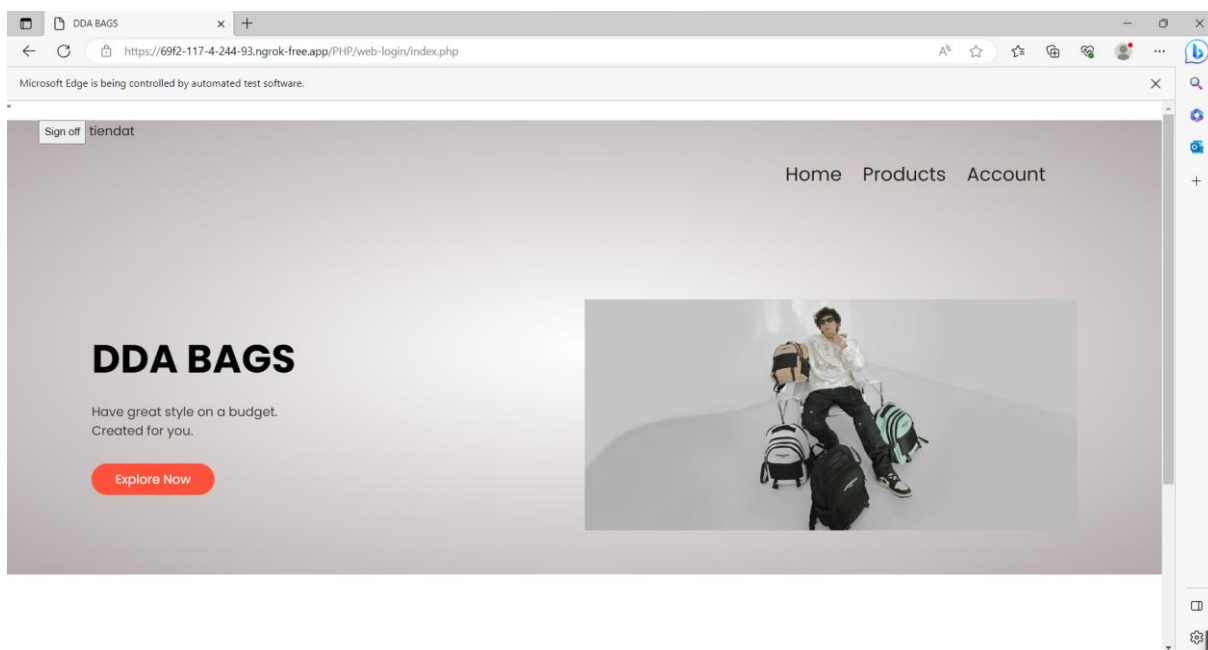
Password

[Sign up](#) [Sign in](#)

Full information required!

**Hình 3.41** Giao diện với Email trống

-Trường hợp 5: Đăng nhập thành công



**Hình 3.42** Giao diện khi đăng nhập thành công

### 3.5.2 Thực nghiệm kiểm thử tự động

#### a) Demo chức năng đăng ký

Các trường hợp kiểm thử chính:

|   |                            |           |   |  |
|---|----------------------------|-----------|---|--|
| Scenario Outline: dang ky sai                               |                            |           |   |  |
| Given mo trang web  |                            |           |   |  |
| When dang ky  |                            |           |   |  |
| And dien thong tin voi "<email1>" va "<email2>" va "<pass>" |                            |           |   |  |
| Then kiem tra voi "<Loi>"                                   |                            |           |   |  |
| Examples:   |                            |           |   |  |
| email1  | email2                     | pass      | Loi   |  |
| testcucumbercdcs@gmail.com                                  | testcucumbercdcs@gmail.com | 123456@Aa | Email invalidate. Please try again.                           |  |
| testcucumber@gmail.com                                      | testcumber@gmail.com       | 123456@Aa | Your email do not match the re-enter email. Please try again. |  |
|   |                            | 123456@Aa | Full information required!                                    |  |
| testcucumbercdcs@gmail.com                                  | testcucumbercdcs@gmail.com |           | Full information required!                                    |  |
| testcucumber@gmail.com                                      | testcucumber@gmail.com     | 12345     | Password length must be 6 characters or more.                 |  |
| testcucumbercdcs@gmail.com                                  | testcucumbercdcs@gmail.com | 123456@Aa | Email already exists. Please try again.                       |  |

Hình 3.43 Scenario Outline

Kết quả từng trường hợp thử nghiệm:

-Trường hợp Email không hợp lệ

|  |              |
|--|--------------|
| ✓ mo trang web   | 5 sec 526 ms |
| ✓ dang ky  | 1 sec 286 ms |
| ✓ dien thong tin voi "testcucumbercdcs@gmail.com" va "testcucumbercdcs@gmail.com" va "123456@Aa" | 752 ms       |
| ✓ kiem tra voi "Email invalidate. Please try again."   | 40 ms        |

Hình 3.44 Trường hợp Email không hợp lệ

B1: Nhập Email không hợp lệ “testcucumbercdcs@gmail.com”

B2: Đăng ký và kiểm tra lỗi “Email invalidate. Please try again.” có hiển thị lên màn hình hay không.

-Trường hợp Email nhập lại không trùng khớp

|  |              |
|--|--------------|
| ✓ mo trang web   | 4 sec 463 ms |
| ✓ dang ky  | 2 sec 71 ms  |
| ✓ dien thong tin voi "testcucumber@gmail.com" va "testcumber@gmail.com" va "123456@Aa" | 1 sec 203 ms |
| ✓ kiem tra voi "Your email do not match the re-enter email. Please try again."         | 33 ms        |

Hình 3.45 Trường hợp Email nhập lại không trùng khớp

B1: Nhập lại Email không trùng khớp “testcucumbercdcs@gmail.com”

B2: Đăng ký và kiểm tra lỗi “Your email do not match the re-enter email. Please try again.” có hiển thị lên màn hình hay không.

### -Trường hợp Email trống

|  |              |
|--|--------------|
| ✓ mo trang web                               | 4 sec 966 ms |
| ✓ dang ky                                    | 886 ms       |
| ✓ dien thong tin voi "" va "" va "123456@Aa" | 1 sec 112 ms |
| ✓kiem tra voi "Full information required!"   | 31 ms        |

**Hình 3.46 Trường hợp Email trống**

B1: Để Email trống

B2: Đăng ký và kiểm tra lỗi “Full information required!” có hiển thị lên màn hình hay không.

### -Trường hợp mật khẩu trống

|   |              |
|---|--------------|
| ✓ mo trang web  | 4 sec 451 ms |
| ✓ dang ky   | 1 sec 881 ms |
| ✓ dien thong tin voi "testcucumbercdcs@gmail.com" va "testcucumbercdcs@gmail.com" va "" | 798 ms       |
| ✓kiem tra voi "Full information required!"  | 27 ms        |

**Hình 3.47 Trường hợp mật khẩu trống**

B1: Để mật khẩu trống

B2: Đăng ký và kiểm tra lỗi “Full information required!” có hiển thị lên màn hình hay không.

### -Trường hợp mật khẩu không hợp lệ

|  |              |
|--|--------------|
| ✓ mo trang web   | 5 sec 331 ms |
| ✓ dang ky  | 827 ms       |
| ✓ dien thong tin voi "testcucumber@gmail.com" va "testcucumber@gmail.com" va "12345" | 757 ms       |
| ✓kiem tra voi "Password length must be 6 characters or more."                        | 29 ms        |

**Hình 3.48 Trường hợp mật khẩu không hợp lệ**

B1: Nhập mật khẩu “12345”

B2: Đăng ký và kiểm tra lỗi “Password length must be 6 characters or more.” có hiển thị lên màn hình hay không.

-Trường hợp Email đã được sử dụng

|  |              |
|--|--------------|
| ✓ mo trang web   | 4 sec 493 ms |
| ✓ dang ky  | 841 ms       |
| ✓ dien thong tin voi "testcucumbercdcs@gmail.com" va "123456@Aa" | 747 ms       |
| ✓ kiem tra voi "Email already exists. Please try again."         | 42 ms        |

Hình 3.49 Trường hợp Email đã được sử dụng

B1: Nhập Email [testcucumbercdcs@gmail.com](mailto:testcucumbercdcs@gmail.com) đã được sử dụng

B2: Đăng ký và kiểm tra lỗi “Email already exists. Please try again.” có hiển thị lên màn hình hay không.

b) Demo chức năng đăng nhập

Các trường hợp kiểm thử chính:

|    |  |
|----|--|
| 17 | Scenario Outline: dang nhap sai  |
| 18 | Given motrang  |
| 19 | When dien thongtin voi "<email>" va "<pass>"                                 |
| 20 | Then Kiem tra "<Loi>"  |
| 21 | Examples:  |
| 22 | email   pass   Loi   |
| 23 | testcucumbercdcs@gmail.com   123456   Wrong account or password.             |
| 24 | testcucumbercdcs@gmail.com   123456@Aa   Email invalidate. Please try again. |
| 25 | testcucumbercdcs@gmail.com     Full information required!                    |
| 26 | 123456@Aa   Full information required!                                       |
| 27 |  |
| 28 | Scenario Outline: dang nhap dung   |
| 29 | Given motrangweb   |
| 30 | When dienthongtin voi "<email>" va "<pass>"                                  |
| 31 | Then Kiem tra  |
| 32 | Examples:  |
| 33 | email   pass   |
| 34 | testcucumbercdcs@gmail.com   123456@Aa                                       |

Hình 3.50 Scenario Outline

Kết quả từng trường hợp thử nghiệm:

-Trường hợp mật khẩu sai

|  |              |
|--|--------------|
| ✓ motrang  | 7 sec 924 ms |
| ✓ dien thongtin voi "testcucumbercdcs@gmail.com" va "123456" | 397 ms       |
| ✓ Kiem tra "Wrong account or password."                      | 52 ms        |

Hình 3.51 Trường hợp mật khẩu sai

B1: Nhập Email “[testcucumbercdcs@gmail.com](mailto:testcucumbercdcs@gmail.com)” và mật khẩu sai “123456”

B2: Đăng nhập và kiểm tra lỗi “Wrong account or password.” có hiển thị lên màn hình hay không.

-Trường hợp Email sai

|   |             |
|---|-------------|
| ✓ motrang   | 5 sec 54 ms |
| ✓ dien thongtin voi "testcucumbercdcs@gmail.com" va "123456@Aa" | 383 ms      |
| ✓ Kiem tra "Email invalidate. Please try again."                | 53 ms       |

**Hình 3.52 Trường hợp Email sai**

B1: Nhập Email sai “[testcucumbercdcs@gmail.com](mailto:testcucumbercdcs@gmail.com)”

B2: Đăng nhập và kiểm tra lỗi “Email invalidate. Please try again.” có hiển thị lên màn hình hay không.

-Trường hợp mật khẩu trống

|  |              |
|--|--------------|
| ✓ motrang  | 4 sec 913 ms |
| ✓ dien thongtin voi "testcucumbercdcs@gmail.com" va "" | 383 ms       |
| ✓ Kiem tra "Full information required!"                | 52 ms        |

**Hình 3.53 Trường hợp mật khẩu trống**

B1: Để trống mật khẩu

B2: Đăng nhập và kiểm tra lỗi “Full information required!” có hiển thị lên màn hình hay không.

-Trường hợp Email trống

|   |              |
|---|--------------|
| ✓ motrang                               | 4 sec 879 ms |
| ✓ dien thongtin voi "" va "123456@Aa"   | 322 ms       |
| ✓ Kiem tra "Full information required!" | 43 ms        |

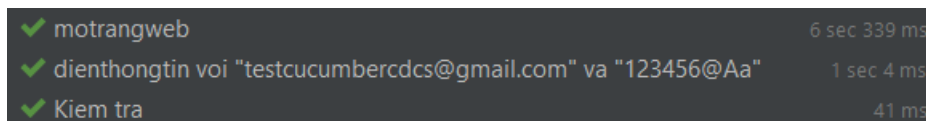
**Hình 3.54 Trường hợp Email trống**

B1: Để trống Email



B2: Đăng nhập và kiểm tra lỗi “Full information required!” có hiển thị lên màn hình hay không.

-Trường hợp đăng nhập thành công



|  |              |
|--|--------------|
| ✓ motrangweb   | 6 sec 339 ms |
| ✓ dienthongtin voi "testcucumbercdcs@gmail.com" va "123456@Aa" | 1 sec 4 ms   |
| ✓ Kiểm tra   | 41 ms        |

Hình 3.55 Trường hợp đăng nhập thành công

B1: Điền đúng thông tin

B2: Đăng nhập và kiểm tra có đăng nhập được hay không.

### 3.6 Kiểm thử một số lỗ hổng web thông dụng

#### 3.6.1 Lỗ hổng an ninh ứng dụng web

Với sự phát triển không ngừng của các ứng dụng web thì các cuộc tấn công ứng dụng web cũng phát triển hết sức phức tạp. Điều này đã đặt ra vấn đề cấp thiết là cần làm thế nào để đảm bảo an toàn thông tin cho ứng dụng web, thông tin của người sử dụng. Hiện tại có khá nhiều phần mềm hỗ trợ lập trình viên, chuyên viên quản trị mạng tìm kiếm lỗ hổng của ứng dụng web. Tuy nhiên, các phần mềm này không theo kịp sự phát triển nhanh chóng của các ứng dụng web, không thể ngăn chặn hoàn toàn các cuộc tấn công với phương thức đa dạng, nguy hiểm trong bối cảnh lỗ hổng ứng dụng web bị phát hiện ngày càng nhiều.

Thống kê cho thấy, hơn 90% các ứng dụng web tồn tại các lỗ hổng an ninh và 75% cuộc tấn công mạng tập trung vào các ứng dụng web. Dự án mở về an ninh ứng dụng web OWASP đã phân loại 10 lỗ hổng ứng dụng web phổ biến và nguy hiểm nhất hiện nay gồm: Nhúng mã (Injection); Xác thực hay quản lý phiên thiếu chính xác; Thực thi mã Script xấu (XSS); Đối tượng tham chiếu không an toàn; Sai sót trong cấu hình an ninh; Lộ dữ liệu nhạy cảm; Điều khiển truy cập mức chức năng không an toàn; Tấn công giả mạo (CSRF); Sử dụng thành phần chứa lỗ hổng đã công khai; Chuyển hướng và chuyển tiếp không an toàn.

### 3.6.2 Lỗ hổng SQL Injection

#### a) Giới thiệu SQL Injection

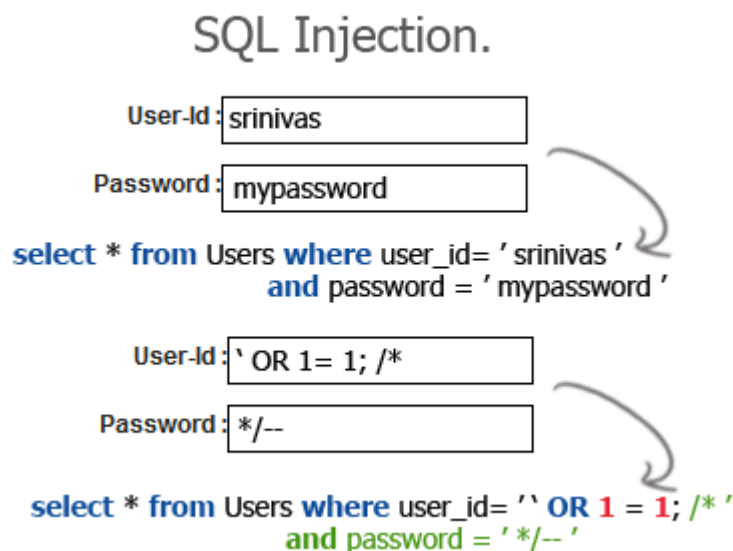
SQLi là lỗ hổng trong việc kiểm tra dữ liệu đầu vào của các ứng dụng web và các thông báo lỗi của hệ quản trị CSDL trả về, được tin tặc khai thác bằng cách tiêm các mã SQL để thực thi câu lệnh bất hợp pháp, đăng nhập mà không cần tên tài khoản và mật khẩu, thực hiện truy cập từ xa, xóa dữ liệu, lấy quyền quản trị của máy chủ.... Trong quy trình kiểm tra và xác nhận tính hợp pháp của tài khoản đó, hệ thống hoặc ứng dụng web tương ứng sẽ chạy 1 câu lệnh truy vấn có dạng như sau:

```
SELECT * FROM Users WHERE User_id=" OR 1=1; /* and password='*/--'
```

Nếu kẻ tấn công chèn đoạn mã '**OR 1=1; /\*** vào giá trị User-Id và **\*/--** vào giá trị Password, câu lệnh truy vấn sẽ có dạng:

```
SELECT * FROM Users WHERE User_id='srinivas' AND Password='mypassword';
```

Đây là câu lệnh có điều kiện luôn đúng, do đó kẻ tấn công có thể lấy toàn bộ dữ liệu của bảng Users.



Hình 3.56: Sơ đồ minh họa tấn công SQLi

#### b) Kiểm thử phát hiện SQLi

Các toán tử để tự động phát hiện lỗ hổng SQLi chia làm 03 dạng [3, 4, 5]:

- Toán tử thay đổi hành vi: **or '1'='1, and '1'='1, ;DROP ALL TABLES;--**
- Toán tử thay đổi cú pháp câu lệnh SQL: **', ", ) , -- , \* , - , #.**

- Toán tử làm mờ:
  - + Thay whitespace bằng ký tự +, /\*\*/, hoặc giá trị unicode %20,%09,%0a, %0b, %0c, %0d, %a0;
  - + Thay dấu ' bằng %27;
  - + Thay 1=1 bằng not false=!!1;
  - + Thay select bằng SELECT hoặc SeLeCt, sel/\*comment here\*/ect.

Sau khi thêm các toán tử sau giá trị đầu vào, gửi yêu cầu đến máy chủ web, phân tích trang phản hồi nếu xuất hiện các dấu hiệu như sau thì website đó có lỗi SQLi:

- Xuất hiện thông báo lỗi từ máy chủ web;
- Xuất hiện thông báo lỗi ẩn trong webservice;
- Chuyển hướng tới trang web khác;
- Báo lỗi 500 (Internal Server Error);
- Không hiển thị gì hoặc hiển thị khác so với trang ban đầu;
- Nếu điều kiện đúng thì hiển thị trang ban đầu, điều kiện sai thì hiển thị lỗi hoặc khác so với trang ban đầu.

Với từng hệ quản trị CSDL khác nhau hoặc dạng lỗi SQLi khác nhau sẽ xuất hiện thông báo lỗi khác nhau [11,13,16].

- Đối với CSDL MySQL xuất hiện các lỗi:
  - + *You have an error in your SQL syntax;*
  - + *mysql\_fetch\_array();*
  - + *mysql\_fetch\_assoc().*
- Đối với CSDL Ms SQL Server xuất hiện các lỗi:
  - + *Microsoft SQL Native Client error;*
  - + *Microsoft OLE DB Provider for SQL Server;*
  - + *Unclosed quotation mark after the character string;*
  - + *Microsoft OLE DB Provider for ODBC Drivers;*
  - + *ODBC SQL Server Driver;*
  - + *Unclosed quotation mark after the character string.*
- Đối với CSDL oracle xuất hiện lỗi: *SQL command not properly ended.*

- Đối với CSDL Postgre SQL xuất hiện các lỗi:
  - + *Query failed: ERROR: syntax error at or near;*
  - + *PSQLErrorException : ERROR;*
  - + *Free and Open Source Software (FOSS).*
- Đối với lỗi SQLi dạng Error based xuất hiện lỗi: Invalid query: The used SELECT statements have a different number of columns.
- Đối với lỗi SQLi dạng Blind: Ta thêm sau giá trị đầu vào chuỗi " **and (true)**" trả về đúng trang ban đầu; Nếu thêm chuỗi " **and (false)**" trả về khác trang ban đầu.

### c) Demo trên Web tự tạo

-Chèn đoạn mã ' or 1 = 1 --' vào sau Email

```

37 >> Scenario Outline: SQLi
38   Given motrang web
39   When dienthongtin voi "<email>" voi "<pass>"
40   Then Kiemtra
41   Examples:
42     | email | pass |
43     | testcucumbercdcs@gmail.com' or 1 = 1 --' | querty1234567 |
44

```

Hình 3.57 Scenario Outline

|  |              |
|--|--------------|
| ✓ Test Results   | 9 sec 533 ms |
| ✓ Cucumber   | 9 sec 533 ms |
| ✓ Web  | 9 sec 533 ms |
| ✓ SQLi   | 9 sec 533 ms |
| ✓ Examples   | 9 sec 533 ms |
| ✓ Example #1.1   | 9 sec 533 ms |
| ✓ motrang web  | 8 sec 673 ms |
| ✓ dienthongtin voi "testcucumbercdcs@gmail.com' or 1 = 1 --" voi "querty1234567" | 811 ms       |
| ✓ Kiemtra  | 49 ms        |

Hình 3.58 Demo thành công

### 3.6.3 Lỗ hổng an ninh XSS

#### a) Giới thiệu lỗ hổng an ninh XSS

XSS là một kiểu tấn công cho phép tin tặc chèn những đoạn script độc hại (thường là javascript hoặc HTML) vào website và thực thi trong trình duyệt của người dùng nhằm đánh cắp những thông tin quan trọng như cookie, mật khẩu...

Những phương pháp tin tặc có thể khai thác qua lỗi XSS:

- Đánh cắp cookie: tin tặc có thể lấy được cookie của người dùng và sử dụng thông tin đánh cắp để giả mạo phiên truy cập hoặc lấy những thông tin nhạy cảm khác được lưu trong cookie.

- Keylogging: tin tặc có thể ghi lại những thao tác gõ phím của người dùng bằng cách sử dụng sự kiện `addEventListener` trong Javascript nhằm đánh cắp các thông tin nhạy cảm, lấy mật khẩu truy cập website hoặc mã số thẻ tín dụng...

- Phishing: tin tặc có thể tạo ra những website giả lừa người dùng đăng nhập để đánh cắp mật khẩu.

## b) Quá trình phát hiện lỗ hổng XSS

Quá trình phát hiện lỗ hổng XSS theo phương pháp hộp đen gồm 03 giai đoạn:

- Xác định vị trí trên website cho phép nhập giá trị đầu vào;
- Chèn các đoạn mã script độc hại vào vị trí nhập giá trị đầu vào;
- Kiểm tra nếu script được thực thi hoặc xuất hiện script trong websource thì website đó có lỗ hổng XSS.

Hiện tại, người lập trình đã thực hiện nhiều biện pháp để lọc script hoặc cấu hình firewall chặn các script độc hại, tuy nhiên vẫn có thể vượt qua việc lọc bằng cách thực hiện mã hóa ký tự metadata hoặc đổi sang mã ASCII... Luận văn đã tổng hợp được 433 payloads để phát hiện và khai thác XSS.

## c) Demo trên Web tự tạo

-Chèn đoạn mã `<script>alert('XSS')</script>` vào sau Email



Hình 3.59 Scenario Outline

|   |              |
|---|--------------|
| ✓ mot rang web  | 6 sec 155 ms |
| ✓ dien thongtin voi "testcucumbercdcs@gmail.com<script>alert('XSS')</script>" voi "querty1234567" | 509 ms       |
| ✓ Kiemtraalert voi "XSS"  | 17 ms        |

Hình 3.60 Demo thành công

### 3.7 Tổng kết chương 3

Chương 3 của báo cáo đã đi sâu vào tìm hiểu tiện ích Serenity BDD with Cucumber trên trình duyệt Microsoft Edge và ứng dụng những kiến thức đã tìm hiểu để thực hiện kiểm thử chức năng đăng ký, đăng nhập của trang web tự tạo và Facebook.

Các nội dung cụ thể trong chương 3 bao gồm:

- Xác định các ca kiểm thử và thiết kế kịch bản.
- Ứng dụng các kiến thức đã nghiên cứu về công cụ kiểm thử tự động Serenity with Cucumber để kiểm thử chức năng đăng nhập, đăng ký.
- Báo cáo kết quả thử nghiệm

## KẾT LUẬN

Kiểm thử phần mềm trên nền Web hay kiểm thử phần mềm trên nền tảng nào thì kiểm thử cũng là một vấn đề quan trọng đối với việc phát triển phần mềm hiện nay.

Trong quá trình thực hiện bài báo cáo thực tập này do thời gian nghiên cứu và tìm hiểu kiến thức cũng như kinh nghiệm của nhóm chúng em còn hạn chế nên một số phần của bài thực tập vẫn chưa được trình bày sâu và kỹ.

Sau khoảng thời gian nghiên cứu đề tài, dưới sự hướng dẫn tận tình, chu đáo, sát sao của Thầy Lê Đức Thuận, bài thực tập của nhóm đã đạt được những kết quả sau:

### **Kết quả đạt được**

- Trình bày tổng quan về phần mềm, kiểm thử phần mềm, đưa ra được tiêu chí đánh giá một phần mềm tốt, tìm hiểu quy trình kiểm thử phần mềm, đưa ra được các mức kiểm thử phần mềm, các kỹ thuật trong kiểm thử phần mềm, các chiến lược trong kiểm thử cũng như nêu ra được các kỹ thuật kiểm thử của hộp đen.
- Tổng quan về kiểm thử tự động, một số công cụ hỗ trợ kiểm thử tự động.
- Tìm hiểu chi tiết cách cài đặt và sử dụng tiện ích Selenium, Serenity BDD with Cucumber trên trình duyệt Microsoft Edge.
- Áp dụng kiến thức đã tìm hiểu để kiểm thử tự động chức năng đăng nhập, đăng ký của trang Web tự tạo và Facebook bằng tiện ích Selenium, Serenity BDD with Cucumber, kiểm thử lỗ hổng SQLi và XSS.
- Bài thực tập là một tài liệu tổng hợp các vấn đề trong kiểm thử phần mềm nói chung, kiểm thử ứng dụng trên nền Web nói riêng và có thể xem như tài liệu hướng dẫn sử dụng Selenium một cách cơ bản nhất bằng tiếng Việt để tham khảo.

### **Hạn chế**

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

- Báo mới tập trung nghiên cứu sâu 1 tiện ích trong bộ công cụ kiểm thử tự động Serenity BDD with Cucumber.
- Chưa nghiên cứu được các kỹ thuật nâng cao khi sử dụng Selenium, Serenity BDD with Cucumber.

## **Hướng phát triển**

Với mong muốn tìm hiểu sâu hơn về kiểm thử phần mềm cũng như bộ công cụ Selenium, Serenity BDD with Cucumber, trong thời gian tới nhóm chúng em sẽ tiếp tục tìm hiểu, nghiên cứu sâu hơn để có thể tiến bộ hơn nữa trong lĩnh vực mà chúng em tìm hiểu.



### III.TÀI LIỆU THAM KHẢO

- [1] Phạm Ngọc Hùng, Trương Anh Hoàng, Đặng Văn Hưng. Giáo trình kiểm thử phần mềm. *Nhà xuất bản Đại học quốc gia Hà Nội*, 2014.
- [2] Arnon Axelrod. Complete Guide to Test Automation. 2018.
- [3] Linda G. Hayes. Automated Testing Handbook. *Software Testing Inst*, 2004.
- [4]<https://viblo.asia/p/quy-trinh-kiem-thu-phan-mem-software-testing-life-cycle-stlc-Qbq5QLvmlD8>
- [5] <https://www.softwaretestingclass.com/software-testing-life-cycle-stlc/>
- [6]<https://jobs.hybrid-technologies.vn/blog/xac-dinh-vi-tri-element-voi-automation-test/>
- [7] [Lỗ hổng XSS và demo khai thác lỗ hổng XSS \(viblo.asia\)](#)
- [8] [Các kỹ thuật khai thác SQL Injection \(viblo.asia\)](#)