|  |
| --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  KHƯƠNG CHÍ HƯỚNG 2021  **TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN**  **KHƯƠNG CHÍ HƯỚNG**  **KIỂM THỬ TỰ ĐỘNG CHO WEBSITE BÁN SÁCH  FAIRY TAIL SỬ DỤNG BDD FRAMEWORK**  **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**  **HƯNG YÊN - 2021** |

|  |
| --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN**  **KHƯƠNG CHÍ HƯỚNG**  **KIỂM THỬ TỰ ĐỘNG CHO WEBSITE BÁN SÁCH  FAIRY TAIL SỬ DỤNG BDD FRAMEWORK**  NGÀNH: CÔNG NGHỆ THÔNG TIN  CHUYÊN NGÀNH: KỸ THUẬT PHẦN MỀM  **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**  **NGƯỜI HƯỚNG DẪN**  **NGÔ THANH HUYỀN**  **HƯNG YÊN – 2021** |

NHẬN XÉT

**Nhận xét của giảng viên hướng dẫn:**

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

.......................................................................................................................................

**GIẢNG VIÊN HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

LỜI CAM ĐOAN

Em xin cam đoan đồ án tốt nghiệp “Kiểm thử tự động cho website bán sách Fairy Tail sử dụng BDD framework” là công trình nghiên cứu của bản thân. Những phần sử dụng tài liệu tham khảo trong đồ án đã được nêu rõ trong phần tài liệu tham khảo. Các số liệu, kết quả trình bày trong đồ án là hoàn toàn trung thực, nếu sai em xin chịu hoàn toàn trách nhiệm và chịu mọi kỷ luật của Bộ môn và Nhà trường đề ra.

*Hưng Yên, ngày … tháng … năm…..*

Sinh viên

…………………………………..

MỤC LỤC

NHẬN XÉT 1

LỜI CAM ĐOAN 2

MỤC LỤC 3

DANH SÁCH CÁC THUẬT NGỮ 7

DANH SÁCH BẢNG BIỂU 8

DANH SÁCH HÌNH VẼ 9

CHƯƠNG 1: MỞ ĐẦU 11

1.1 Lý do chọn đồ án 11

1.2 Mục tiêu của đồ án 11

1.2.1 Mục tiêu tổng quát 11

1.2.2 Mục tiêu cụ thể 12

1.3 Giới hạn và phạm vi của đồ án 12

1.3.1 Đối tượng nghiên cứu 12

1.3.2 Phạm vi nghiên cứu 12

1.4 Nội dung thực hiện 12

1.5 Phương pháp tiếp cận 13

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 14

2.1 Tổng quan về kiểm thử phần mềm 14

2.1.1 Các cấp độ kiểm thử phần mềm 14

Kiểm thử đơn vị (Unit Test) 14

Kiểm thử tích hợp (Integration Test) 14

Kiểm thử hệ thống (System Test) 15

Kiểm thử chấp nhận (Acceptance Test) 15

2.1.2 Kỹ thuật kiểm thử phần mềm 15

a) Kỹ thuật kiểm thử hộp đen (Black – box Testing) 16

Kỹ thuật kiểm thử hộp trắng (White – box Testing) 16

2.2 Kiểm thử tự động phần mềm 16

2.3 Kiểm thử chức năng 16

2.3.1 Định nghĩa 16

2.3.2 Mục đích 17

2.3.3 Các kỹ thuật thường dùng trong kiểm thử chức năng: 17

2.4 Kiểm thử tự động hướng hành vi (BDD framework) 17

2.5 Giới thiệu về Cucumber 18

CHƯƠNG 3: ĐẶC TẢ HỆ THỐNG PHẦN MỀM 21

3.1 Giới thiệu chung 21

3.1.1 Mục đích 21

3.1.2 Phạm vi áp dụng 21

3.1.3 Thông tin chung 21

3.1.4 Use Case của hệ thống 22

3.2 Các yêu cầu chức năng 23

3.2.1 Chức năng đăng ký 23

3.2.2 Chức năng đăng nhập 25

3.2.3 Màn hình chức năng quản lý sách 27

Chức năng [Tìm kiếm]: 27

Chức năng [Tạo mới]: 29

c) Chức năng [Sửa sách]: 30

d) Chức năng [Xóa sách]: cho phép xóa sách 32

e) Chức năng [Xem chi tiết]: 33

f) Chức năng [Cập nhật giá] 34

3.8.7. Chức năng [Cập nhật thể loại] 35

3.2.4 Quản lý quản lý đơn hàng 36

3.2.5 Quản lý đặt hàng 38

CHƯƠNG 4: TRIỂN KHAI KIỂM THỬ TỰ ĐỘNG 41

4.1 Thiết kế các yêu cầu kiểm thử 41

4.2 Xây dựng ca kiểm thử 46

4.2.1 Chức năng đăng nhập 46

4.2.2 Chức năng quản lý sản phẩm 48

4.2.3 Chức năng quản lý giỏ hàng 58

4.2.4 Chức năng quản lý đặt hàng 61

4.3 Xây dựng dữ liệu kiểm thử 66

4.4 Phương pháp xây dựng Framework 67

4.4.1 Xây dựng lớp CommonBase 67

4.4.2 Xây dựng lớp constant 98

4.4.3 Xây dựng các Feature file 99

4.4.4 Xây dựng lớp step\_definitions 101

4.4.5 Xây dựng lớp runner 118

4.5 Thực thi và báo cáo kiểm thử 119

4.5.1 Thực thi kiểm thử 119

4.5.2 Báo cáo kiểm thử 119

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 121

5.1 Kết quả đạt được 121

5.2 Hạn chế của đề tài 122

5.3 Hướng phát triển của đề tài 122

TÀI LIỆU THAM KHẢO 123

PHỤ LỤC 124

DANH SÁCH CÁC THUẬT NGỮ

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Từ đầy đủ | Giải thích |
| BDD | Behavior Driven Development | Quá trình phát triển phần mềm dựa trên phương pháp Agile |
| CSDL | Cơ sở dữ liệu |  |
| TDD | Test driven development | Là một phương thức làm việc, hay một quy trình viết mã hiện đại |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

DANH SÁCH BẢNG BIỂU

[Bảng 2-1. So sánh Cucumber với các công cụ khác 20](#_Toc75458877)

[Bảng 3-1. Mô tả các chức năng của hệ thống 22](#_Toc75458878)

DANH SÁCH HÌNH VẼ

[Hình 2-1. Các cấp độ kiểm thử phần mềm 14](#_Toc75458850)

[Hình 3-1. Màn hình form “đăng ký” 24](#_Toc75458851)

[Hình 3-2. Màn hình form “đăng nhập” 25](#_Toc75458852)

[Hình 3-3. Màn hình quản lý sách 27](#_Toc75458853)

[Hình 3-4. Ảnh minh hoa thêm sách 29](#_Toc75458854)

[Hình 3-5. Ảnh minh họa chức năng sửa sách 31](#_Toc75458855)

[Hình 3-6. Ảnh minh họa chức năng xóa sách 32](#_Toc75458856)

[Hình 3-7. Ảnh minh họa chức năng xem chi tiết sách 33](#_Toc75458857)

[Hình 3-8. Ảnh minh họa chức năng cập nhật giá 34](#_Toc75458858)

[Hình 3-9. Ảnh minh họa chức năng cập nhật thể loại 35](#_Toc75458859)

[Hình 3-10. Biểu đồ trạng thái của đơn hàng 36](#_Toc75458860)

[Hình 3-11. Màn hình quản lý đơn hàng 37](#_Toc75458861)

[Hình 3-12. Minh họa chức năng đặt hàng 38](#_Toc75458862)

[Hình 3-13. Hình minh họa chức năng thanh toán 39](#_Toc75458863)

[Hình 3-14. Ảnh minh họa bảng thanh toán 40](#_Toc75458864)

[Hình 4-1. Test design Quản lý sách - Thêm mới 41](#_Toc75458865)

[Hình 4-2. Test design Quản lý sách – Sửa 42](#_Toc75458866)

[Hình 4-3. Test design Quản lý sách – Xoá 43](#_Toc75458867)

[Hình 4-4. Test design Quản lý sách – Cập nhật giá 43](#_Toc75458868)

[Hình 4-5. Test design Đăng nhập 44](#_Toc75458869)

[Hình 4-6. Test design giỏ hàng 44](#_Toc75458870)

[Hình 4-7. Test design Đặt hàng 1 45](#_Toc75458871)

[Hình 4-8. Test design Đặt hàng 2 46](#_Toc75458872)

[Hình 4-9. Dữ liệu kiểm thử cho chức năng đăng nhập 66](#_Toc75458873)

[Hình 4-10. Dữ liệu kiểm thử cho chức năng tạo sách 66](#_Toc75458874)

[Hình 4-11. Các ca kiểm thử được thực thi 119](#_Toc75458875)

[Hình 4-12. Báo cáo tổng khi chạy xong các ca kiểm thử 120](#_Toc75458876)

# MỞ ĐẦU

* 1. Lý do chọn đồ án

Hiện nay, tự động hóa được ứng dụng trong rất nhiều lĩnh vực, mục đích thường đa dạng và tùy theo nhu cầu của từng lĩnh vực. Tuy nhiên, điểm chung nhất của giải pháp này là hướng đến giảm nhân lực, thời gian và nâng cao chất lượng của quá trình kiểm thử. Trong quá trình tìm hiểu cũng như được sự hướng dẫn tận tình từ phía thầy cô em đã thấy được những ưu điểm nổi trội về kiểm thử tự động nên đã chọn hướng đi của đề tài là “Kiểm thử tự động”.

Các ứng dụng Web được phát triển một cách mạnh mẽ nhưng yêu cầu của khách hàng đặt ra dường như chưa đáp ứng được. Câu hỏi quan trọng trong phát triển phần mềm đặt ra là: “Làm thế nào để đảm bảo được chất lượng cua các ứng dụng Web”. Kiểm thử gần như là phương pháp duy nhất để đảm bảo chất lượng cho các sản phẩm phần mềm trong các công ty phần mềm hiện nay. Giai đoạn kiểm thử là giai đoạn chiếm mất rất nhiều công sức và tiền của, chi phí cho giai đoạn này cũng thường chiếm tới 40% tổng các nố lực dành cho một dự án phát triển phần mềm, do đó việc áp dụng kiểm thử tự động khiến chúng ta có thể cắt giảm đi rất nhiều thời gian và chi phí không cần thiết.

Vì những lí do trên, em xin chọn đề tài là “Xây dựng Framework kiểm thử tự động hướng hành vi (BDD framework) và áp dụng kiểm thử tự động cho website bán sách Fairy Tail”.

* 1. Mục tiêu của đồ án
     1. Mục tiêu tổng quát

Đưa ra một cái nhìn tổng quan trong việc nghiên cứu, lựa chọn framework, công cụ kiểm thử tự động áp dụng trong từng trường hợp kiểm thử phần mềm tự động hiện nay, từ đó tìm ra ưu điểm và nhược điểm của kiểm thử tự động hướng hành vi (BDD framework).

* + 1. Mục tiêu cụ thể

Nghiên cứu framework kiểm thử tự động phần mềm trong ngành công nghiệp phần mềm: Tập trung nghiên cứu và xây dựng kiểm thử tự động hướng hành vi (BDD framework)

Xây dựng bộ thư viện, tài liệu kiểm thử hỗ trợ việc tự động hóa kiểm thử phần mềm áp dụng kiểm thử với trang web bán sách Fairy Tail.

* 1. Giới hạn và phạm vi của đồ án
     1. Đối tượng nghiên cứu

Kiểm thử tự động hướng hành vi (Behavior Driven Development Framework). Xây dựng bằng ngôn ngữ cucumber, selenium, webdriver.

Khách thể nghiên cứu: Đề tài sử dụng website bán sách Fairy Tail để làm đối

tượng thực hiện kiểm thử tự động

* + 1. Phạm vi nghiên cứu

Đề tài được nghiên cứu dựa trên ngôn ngữ java, cucumber. Áp dụng trực tiếp vào website bán sách Fairy Tail. Đề tài được nghiên cứ từ tháng 01/2021 tới tháng 06/2021

* 1. Nội dung thực hiện

Những phần chính mà đề tài sẽ đạt được trong quá trình nghiên cứu và xây dựng đề tài là:

- Nghiên cứu đặc tả phần mềm của website bán sách Fairy Tail. Từ đó thiết kế các trường hợp kiểm thử (Test Design), các ca kiểm thử tương ứng (Test Case), các bộ dữ liệu để thực hiện kiểm thử.

- Nghiên cứu và phát triển “Kiểm thử tự động hướng hành vi (BDD framework)”

- Áp dụng “Kiểm thử tự động hướng hành vi (BDD framework)” vào website bán sách Fairy Tail.

- Đưa ra được kết quả kiểm thử, kết luận về kiểm thử hướng hành vi (BDD framework)

* 1. Phương pháp tiếp cận

Để thực hiện được đề tài, đầu tiên chúng ta cần phải nghiên cứu về “Kiểm thử tự động” trước tiên. Tiếp theo chúng ta cần nắm rõ được các framework về kiểm thử tự động, đi sâu vào “Kiểm thử tự động hướng hành vi (BDD framework)”.

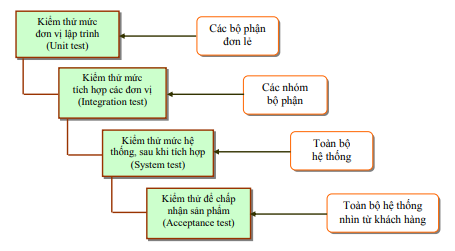
Tiếp theo chúng ta cần phân tích đặc tả phần mềm của website một cách chi tiết, để có thể nắm rõ được phần mềm, từ đó thiết kế ra các trường hợp kiểm thử (Test Design), các ca kiểm thử (Test Case), dữ liệu kiểm thử (Test Data).

# CƠ SỞ LÝ THUYẾT

* 1. Tổng quan về kiểm thử phần mềm

Kiểm thử phần mềm là quá trình thực thi một hệ thống phần mềm để xác định xem phần mềm có đúng với đặc tả không và thực hiện trong môi trường như mong đợi hay không. Mục đích của kiểm thử phần mềm là tìm ra lỗi chưa được phát hiện, tìm một cách sớm nhất và bảo đảm rằng lỗi sẽ được sửa. Mục tiêu của kiểm thử phần mềm là thiết kế tài liệu kiểm thử một cách có hệ thống và thực hiện nó sao cho có hiệu quả, nhưng tiết kiệm được thời gian, công sức và chi phí.

* + 1. Các cấp độ kiểm thử phần mềm



* 1. Các cấp độ kiểm thử phần mềm

Kiểm thử đơn vị (Unit Test)

Một đơn vị (Unit) là một thành phần phần mềm nhỏ nhất mà ta có thể kiểm thử được, ví dụ: các hàm (Function), thủ tục (Procedure), lớp (Class), hoặc các phương thức (Method).

Kiểm thử tích hợp (Integration Test)

Kiểm thử tích hợp kết hợp các thành phần của một ứng dụng và kiểm thử như một ứng dụng đã hoàn thành. Trong khi kiểm thử đơn vị kiểm tra các thành phần và Unit riêng lẻ thì kiểm thử tích hợp kết hợp chúng lại với nhau và kiểm tra sự giao tiếp giữa chúng.

Kiểm thử hệ thống (System Test)

Mục đích của kiểm thử hệ thống là kiểm thử xem thiết kế và toàn bộ hệ thống (sau khi tích hợp) có thỏa mãn yêu cầu đặt ra hay không.

Kiểm thử hệ thống kiểm tra cả các hành vi chức năng của phần mềm lẫn các yêu cầu về chất lượng như độ tin cậy, tính tiện lợi khi sử dụng, hiệu năng và bảo mật.

Kiểm thử hệ thống bắt đầu khi tất cả các bộ phận của phần mềm đã được tích hợp thành công.

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

Kiểm thử chấp nhận (Acceptance Test)

Mục đích của kiểm thử chấp nhận là kiểm thử khả năng chấp nhận cuối cùng để chắc chắn rằng sản phẩm là phù hợp và thỏa mãn các yêu cầu của khách hàng và khách hàng chấp nhận sản phẩm.

Trong giai đoạn kiểm thử chấp nhận thì người kiểm tra là khách hàng. Khách hàng sẽ đánh giá phần mềm với mong đợi theo những thao tác sử dụng quen thuộc của họ. Việc kiểm tra ở giai đoạn này có ý nghĩa hết sức quan trọng tránh cho việc hiểu sai yêu cầu cũng như sự mong đợi của khách hàng.[1]

* + 1. Kỹ thuật kiểm thử phần mềm

Mục tiêu của kiểm thử là phải thiết kế các trường hợp kiểm thử có khả năng cao nhất trong việc phát hiện nhiều lỗi với thời gian và công sức tối thiểu. Do đó có thể chia các kỹ thuật kiểm thử thành hai loại:

* Kỹ thuật kiểm thử hộp đen (Black – box Testing) hay còn gọi là kỹ thuật kiểm thử chức năng (Functional Testing).
* Kỹ thuật kiểm thử hộp trắng (White – box Testing) hay còn gọi là kỹ thuật kiểm thử cấu trúc (Structural Testing)
  + 1. Kỹ thuật kiểm thử hộp đen (Black – box Testing)

Kiểm thử hộp đen còn được gọi là kiểm thử hướng dữ liệu (data - driven) hay là kiểm thử hướng vào/ra (input/output driven).

Trong kỹ thuật này, người kiểm thử xem phần mềm như là một hộp đen. Người kiểm thử hoàn toàn không quan tâm đến cấu trúc và hành vi bên trong của chương trình. Người kiểm thử chỉ cần quan tâm đến việc tìm các hiện tượng mà phần mềm không hành xử theo đúng đặc tả của nó. Do đó, dữ liệu kiểm thử sẽ xuất phát từ đặc tả.

Kỹ thuật kiểm thử hộp trắng (White – box Testing)

Kiểm thử hộp trắng hay còn gọi là kiểm thử hướng logic, cho phép kiểm tra cấu trúc bên trong của phần mềm với mục đích bảo đảm rằng tất cả các câu lệnh và điều kiện sẽ được thực hiện ít nhất một lần. Người kiểm thử truy nhập vào mã nguồn chương trình và có thể kiểm tra nó, lấy đó làm cơ sở để hỗ trợ việc kiểm thử.

* 1. Kiểm thử tự động phần mềm

Trong [kiểm thử phần mềm](https://vi.wikipedia.org/wiki/Ki%E1%BB%83m_th%E1%BB%AD_ph%E1%BA%A7n_m%E1%BB%81m), kiểm thử tự động (tiếng Anh: test automation) là việc sử dụng [phần mềm](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m) đặc biệt (tách biệt với phần mềm đang được kiểm thử) để kiểm soát việc thực hiện các bài kiểm tra và so kết quả thực tế với kết quả dự đoán. Tự động kiểm thử có thể tự động hóa một số nhiệm vụ lặp đi lặp lại nhưng cần thiết trong một quá trình thử nghiệm đã được chính thức hóa, hay là các kiểm thử bổ sung nhưng sẽ khó thực hiện thủ công. Kiểm thử tự động là rất quan trọng cho [phân phối liên tục](https://vi.wikipedia.org/wiki/Ph%C3%A2n_ph%E1%BB%91i_li%C3%AAn_t%E1%BB%A5c) và [kiểm thử liên tục](https://vi.wikipedia.org/wiki/Ki%E1%BB%83m_th%E1%BB%AD_li%C3%AAn_t%E1%BB%A5c).

Một số nhiệm vụ kiểm thử phần mềm, như kiểm thử hồi quy giao diện cấp thấp rộng, có thể tốn nhiều thời gian và công sức để thực hiện. Ngoài ra, phương pháp thủ công có thể không phải lúc nào cũng có hiệu quả trong việc tìm kiếm các lớp lỗi. Kiểm tra tự động cung cấp một khả năng để thực hiện các loại kiểm thử một cách hiệu quả. Khi các bài kiểm thử tự động được phát triển, chúng có thể chạy nhanh và liên tục. [2]

* 1. Kiểm thử chức năng
     1. Định nghĩa

Kiểm thử chức năng là một loại kiểm thử hộp đen (black box) và test case của nó được dựa trên đặc tả của ứng dụng phần mềm/thành phần đang kiểm thử. Các chức năng được kiểm thử bằng cách nhập vào các giá trị nhập và kiểm tra kết quả đầu ra, và ít quan tâm đến cấu trúc bên trong của ứng dụng.

Nó là một qui trình cố gắng tìm ra các khác biệt giữa đặc tả bên ngoài của phần mềm và thực tế mà phần mềm cung cấp. Với các đặc tả bên ngoài của phần mềm là đặc tả chính xác về hành vi của phần mềm theo góc nhìn của người dùng. [3]

* + 1. Mục đích

Với kiểm thử đơn vị ta phát hiện sự khác biệt giữa đặc tả giao tiếp của đơn vị và thực tế mà đơn vị này cung cấp.

Với kiểm thử hệ thống ta chỉ ra rằng chương trình không tương thích với các mục tiêu ban đầu của nó. Thì:

Với kiểm thử chức năng ta sẽ hoàn thiện nốt phần cần xác minh còn lại là chỉ ra rằng chương trình không tương thích với các đặc tả bên ngoài của nó.

Các lợi ích: Tránh kiểm thử dư thừa. Ngăn chặn sự quan tâm nhiều vào quá nhiều loại lỗi tại từng thời điểm.

* + 1. Các kỹ thuật thường dùng trong kiểm thử chức năng:

**-** Kỹ thuật phân lớp tương đương (Equivalence Class Partitioning).

- Kỹ thuật dùng các bảng quyết định (Decision Tables).

- Kỹ thuật kiểm thử các bộ thần kỳ (Pairwise).

- Kỹ thật phân tích vùng miền (domain analysis).

- Kỹ thuật dựa trên đặc tả Use Case (Use case).

* 1. Kiểm thử tự động hướng hành vi (BDD framework)

**Đặc điểm cơ bản**

Behavior Driven Development (BDD) là một phần mở rộng của TDD. Giống như ở TDD, trong BDD chúng ta cũng viết các bài test trước và sau đó mới code ứng dụng để vượt qua những bài test đó. Sự khác biệt chính của BDD so với TDD chính là:

* Các bài test hay các test case sẽ được viết bằng ngữ pháp tiếng Anh và mô tả rất đơn giản
* Các bài test được xem như là hành vi của ứng dụng dựa trên những nhu cầu thực tế của người dùng với ứng dụng đó
* Sử dụng các ví dụ để làm rõ các yêu cầu của ứng dụng

Sự khác biệt này dẫn đến sự cần thiết phải có một ngôn ngữ có thể định nghĩa, trong một định dạng dễ hiểu để chúng ta có thể áp dụng BDD một cách hiệu quả nhất. Và Gherkin là ngôn ngữ sẽ làm việc này (sẽ giới thiệu cụ thể ở bài sau)

**Ưu điểm của mô hình hướng hành vi**

* Giúp xác định đúng yêu cầu của khách hàng: tài liệu được viết dưới dạng ngôn ngữ tự nhiên, bất kỳ đối tượng nào cũng có thể hiểu được. Khi đọc tài liệu này, khách hàng có thể dễ dàng nhận biết được lập trình viên có hiểu đúng yêu cầu của họ không và có phản hồi kịp thời.
* Là tài liệu sống của dự án: tài liệu này luôn được cập nhật khi có bất kỳ sự thay đổi nào nên tất cả các thành viên sẽ không bị miss thông tin khi phát triển hệ thống
* Nâng cao chất lượng phần mềm, tạo ra sản phẩm hữu ích: vì phát triển phần mềm theo hướng hành vi nên có thể focus vào việc tạo ra sản phẩm đúng với yêu cầu của khách hàng nhưng vẫn hữu ích cho người dùng.

**Nhược điểm**

* Yêu cầu hiểu sâu về số lượng lớn các khái niệm, vì vậy muốn tiếp cận với BDD thì yêu cầu developers cần hoàn toàn hiểu rõ về TDD.
* Vì nó là một khái niệm, biến nó thành một kỹ thuật thực hành hoặc kết nối nó với một bộ công cụ có nghĩa là hủy hoại nó.
  1. Giới thiệu về Cucumber

**Cucumber**

Cucumber, testing framework hỗ trợ Behavior Driven Development (BDD), cho phép người dùng định nghĩa hành vi hệ thống với ngữ nghĩa tiếng anh thông qua cú pháp Gherkin. Cucumber hướng tới việc viết test “as cool as cucumber” mà bất kỳ ai cũng có thể hiểu cho dù họ không có chuyên môn kĩ thuật. Ví dụ như các nền tảng quen thuộc như Selenium thì thường chỉ người viết test hoặc có kĩ năng lập trình mới hiểu được những gì đang test, còn khác hàng hoặc các bên liên quan thì không đọc ngay code để hiểu mà họ cần hiểu qua tài liệu.

Cucumber ban đầu được thực hiện dành riêng cho ngôn ngữ Ruby và sau đó được mở rộng sang Java, cả Ruby và Java đều sử dụng Junit để chạy test.

**Behavior Driven Development**

Trong BDD, người dùng (business analysts – người phân tích nghiệp vụ, product owners – người sỡ hửu sản phẩm) sẽ viết kịch bản(scenarios) hoặc acceptance test (kiểm thử chấp nhận) mô tả hành vi của hệ thống từ quan điểm của khách hàng trước và trong giai đoạn phát triển. Cucumber và BDD giải quyết hạn chế rất hay gặp trong các dự án phần mềm: mỗi người sẽ hiểu dự án theo một cách khác nhau.

BDD có khả năng tạo ra các kịch bản test dựa trên góc nhìn của bên phát triển cũng như góc nhìn của bên khác hàng. Ngay từ ban đầu, các thành viên dự án sẽ thảo luận để tạo ra các kịch bản trước, sau đó sẽ cài đặt dựa trên kịch bản đó, tất cả kịch bản test gần gũi với ngôn ngữ tiếng Anh, do đó nó đóng luôn vai trò của tài liệu.

**Workflow BDD**

Sau khi kịch bản test chạy, Cucumber sẽ đọc mã Gherkin từ file feature, sau đó nó sẽ tìm đoạn mã trong file step definition mô tả đúng với hành động trong file feature và thực hiện đoạn code, ở bước chạy code Cucumber có thể kết hợp với các framework khác như Ruby on Rails, Selenium, Spring, …

**Lợi ích**

* Giúp cho các bên liên quan đến dự án (stakeholders) có thể follow hoạt động test mà không cần kiến thức kĩ thuật chuyên môn
* Cucumber tập trung vào trải nghiệm người dùng cuối
* Style viết mã dễ bảo trì và thực hiện
* Công cụ hiệu quả cho kiểm thử
  1. So sánh Cucumber với các công cụ khác

| **Cucumber** | **HP ALM (QTP)** | **Selenium** |
| --- | --- | --- |
| Miễn phí | Trả phí | Miễn phí |
| Công cụ hỗ trợ Behaviour driven development BDD | Công cụ hỗ trợ Functional testing | Công cụ hỗ trợ Functional and Performance testing |
| Plugin hoạt động nhanh | Plugin hoạt động chậm hơn Cucumber và Selenium | Plugin hoạt động chậm hơn Cucumber |
| Hỗ trợ Java, Scala, Groovy | Chỉ hỗ trợ VB script | Hỗ trợ Java, .Net, Ruby |
| Dev, test viết script | Chỉ tester viết test script | Dev, test viết script |
| Chỉ support Web app | Support ứng dụng Web, desktop, client server app | Chỉ support web app |

**Thành phần của Cucumber**

Các project Cucumber luôn có một thư mục con tại thư mục gốc (root) project tên "features". Đây là nơi lưu trữ tất cả các features của projects, ngoài ra còn có các thư mục bổ sung (additional directories) và thư mục hỗ trợ (support directories). [4]

# ĐẶC TẢ HỆ THỐNG PHẦN MỀM

* 1. Giới thiệu chung
     1. Mục đích

Mục đích của tài liệu này để xác định chức năng và các yêu cầu khác về chương trình website cửa hàng bán sách Fairy Tail, bao gồm giới thiệu tổng quát về hệ thống, yêu cầu, tính ứng dụng, độ tin cậy và hiệu suất. Tài liệu này được dùng cho đội dự án để phát triển sản phẩm.

Phần 1: Sẽ giới thiệu chung về tài liệu.

Phần 2: Sẽ cung cấp thông tin tổng quát về hệ thống Website bán sách Fairy Tail

Phần 3: Sẽ mô tả các yêu cầu cụ thể của các chức năng, bao gồm input, output, các xử lý của chương trình.

Phần 4: Sẽ mô tả các yêu cầu khác liên quan đến tính ứng dụng của hệ thống (tính thân thiện với end user).

Phần 5: Sẽ mô tả các yêu cầu về tính toàn vẹn của dữ liệu, hiệu suất.

* + 1. Phạm vi áp dụng

Phần mềm này sẽ được áp dụng cho tất cả cả mọi người trên internet.

Tài liệu này sẽ mô tả đầy đủ yêu cầu về chức năng và các yêu cầu khác của hệ thống.

* + 1. Thông tin chung

Môi trường tối thiểu để chạy chương trình:

- Phần cứng: chipset 1GHz, Ram 512 MB, Network 10 Mb/s.

- Phần mềm: OS từ XP trở lên, .Net framework 3.5, MS Office 2000.

- Database server: MS SQL server 2008.

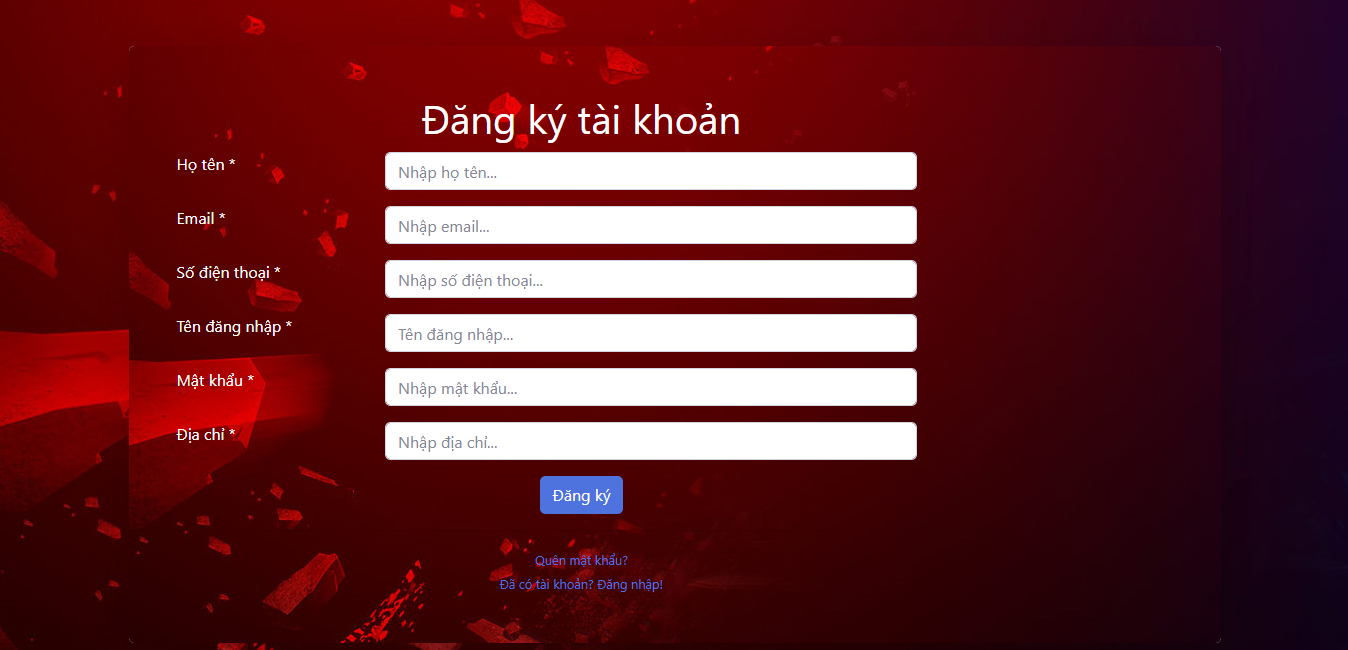
Cơ chế tìm kiếm chung của các chức năng là Tìm Kiếm thông qua từ khóa tìm kiếm. Nếu điều kiện tìm kiếm để trống thì sẽ coi là không tìm kiếm.

Khi sử dụng chương trình, admin có thể mở đồng thời nhiều màn hình khác nhau để sử dụng. Nếu màn hình đang mở thì sẽ không mở thêm mà sẽ hiển thị màn hình đó. Tại 1 thời điểm sẽ chỉ có 1 màn hình ở trạng thái được hiển thị.

* + 1. Use Case của hệ thống
  1. Mô tả các chức năng của hệ thống

|  |  |  |
| --- | --- | --- |
| STT | Chức năng | Mô tả |
| 1 | Quản lý loại sách | Quản lý thông tin loại sách bao gồm các chức năng: thêm, sửa, xóa, tìm kiếm. |
| 2 | Quản lý sách | Người quản trị có thể quản lý thông tin tất cả sách bao gồm cả thêm, sửa, xóa, tìm kiếm các sản phẩm trên website đang bán. |
| 4 | Quản lý giá | Người quản trị có thể quản lý thông tin về giá bao gồm thêm, sửa giá. |
| 5 | Quản lý tác giả | Người quản trị có thể quản lý thông tin về tác giả bao gồm thêm, sửa, tìm kiếm tác giả. |
| 6 | Quản ý nhà xuất bản | Người quản trị có thể quản lý thông tin về nhà xuất bản bao gồm thêm, sửa, xóa, tìm kiếm nhà xuất bản. |
| 7 | Quản lý đơn hàng chưa xác thực | Người quản trị quản lý các đơn hàng chưa xác thực: xác thực đơn hàng, hủy đơn hàng chưa xác thực. |
| 8 | Quản lý đơn hàng đã xác thực | Người quản trị quản lý các đơn hàng đã xác thực |
| 9 | Quản lý hóa đơn bán | Người quản trị quản lý các đơn hàng đã giao: ghi nhận thanh toán, ghi nhận giao hàng không thành công. |
| 10 | Quản lý nhà cung cấp | Khi nhập sách về bán, quản trị viên sẽ thêm thông tin nhà cung cấp sách vào hệ thống, khi thông tin của nhà cung cấp thay đổi thì sẽ phải cập nhật lại, và khi không còn nhập sách từ đó nhà cung cấp đó nữa thì xóa khỏi hệ thống |
| 11 | Quản lý nhập hàng | Khi có hàng mới nhập thêm về, quản trị viên sẽ tiến hành nhập thông tin hóa đơn nhập hàng, và nếu nội dung hóa đơn nhập có sai sót sẽ tiến hành cập nhật lại thông tin hóa đơn nhập hàng. Khi không còn muốn lưu trữ thông tin nhập hàng sẽ xóa khỏi hệ thống |
| 12 | Quản lý tin tức | Quản trị viên sẽ quản lý thông tin các tin tức liên quan đến sách của cửa hàng bao gồm: mã tin tức, tiêu đề, tên tin tức, nội dung, ảnh, ngày đăng, người đăng. Thêm tin tức mới, cập nhật tin, xóa tin. |
| 13 | Quản lý nhân viên | Người quản trị có thể quản lý thông tin về nhân viên bao gồm thêm, sửa, xóa, hiển thị nhân viên. |
| 14 | Quản lý khách hàng | Người quản trị có thể quản lý thông tin về khách hàng |
| 15 | Quản lý user | Người quản trị có thể quản lý thông tin về user |

* 1. Các yêu cầu chức năng
     1. Chức năng đăng ký



* 1. Màn hình form “đăng ký”

Chức năng [Đăng ký]: cho phép đăng ký tài khoản người dùng (tài khoản khách hàng) để có thể đăng nhập vào hệ thống.

Validate:

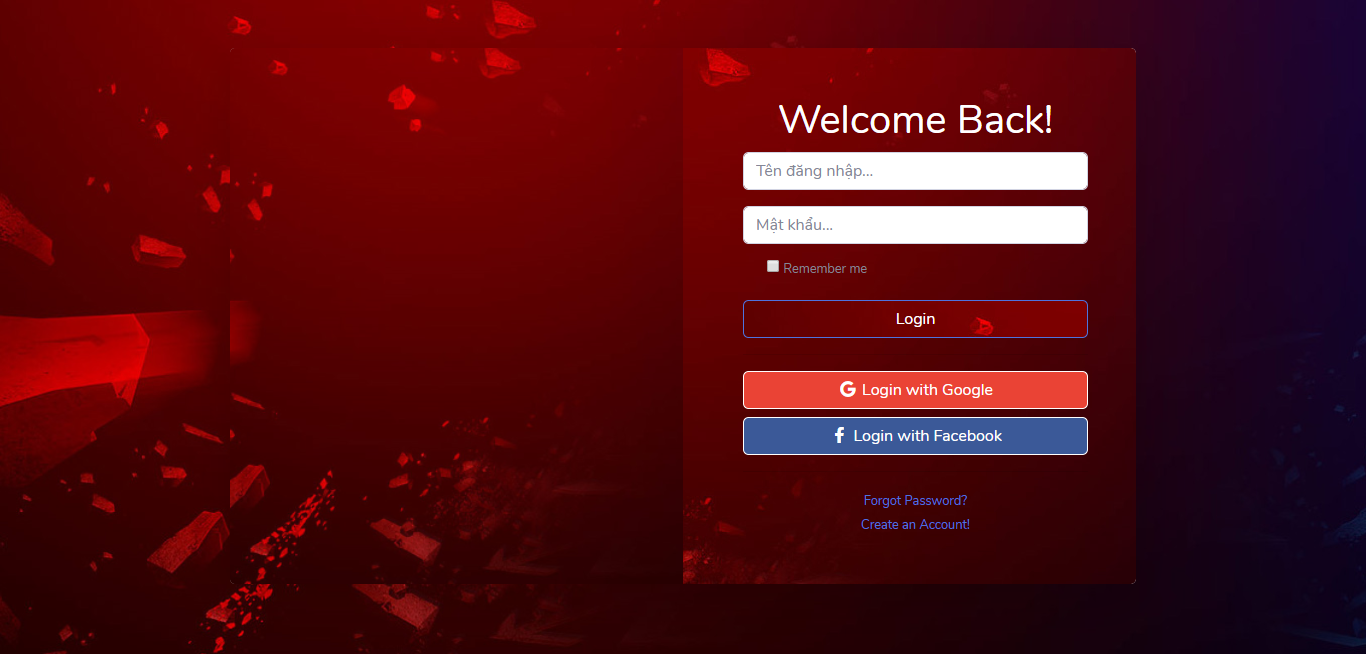
* Tên đăng nhập: length (50), khác rỗng
* Mật khẩu: kiểu mật khẩu (“\*\*\*\*\*”), length (32), khác rỗng
* Tên người dùng: length (50), khác rỗng
* Số điện thoại: length (15), kiểu number, khác rỗng
* Địa chỉ: length (500), khác rỗng
* Email: length (50), kiểu email (“xyz@gmail.com”)

Chức năng: Người dùng nhập thông tin vào các ô nhập liệu và nhấn nút [Đăng ký]

* Nếu có trường không hợp lệ về validate: hiện label màu đỏ ngay dưới ô nhập liệu không hợp lệ.
* Nếu [Email] đã có người sử sụng để đăng ký: hiện label màu đỏ dưới nút [Đăng ký] với nội dung: “Email đã có người đăng ký!”.
* Nếu tên tài khoản đã có người sử dụng: hiện label màu đỏ dưới nút [Đăng ký] với nội dung: “Tên tài khoản đã tồn tại”.
* Khi đăng ký thành công: hệ thống sẽ chuyển người dùng tới trang đăng nhập, tại đây người dùng có thể đăng nhập bằng tài khoản mình vừa đăng ký.
* Mật khẩu sau khi đăng kí sẽ được mã hóa bằng MD5 (là 1 hàm mã hóa 1 chiều của C#) và lưu mật khẩu sau khi mã hóa vào CSDL.

Chức năng: [Đã có tài khoản? sĐăng nhập!]

* Khi nhấn vào nút [Đã có tài khoản? Đăng nhập!] hệ thống sẽ chuyển về trang đăng nhập, tại đây người dùng có thể đăng nhập vào hệ thống với quyền quản trị viên, nhân viên hoặc quyền khách hàng.
  + 1. Chức năng đăng nhập



* 1. Màn hình form “đăng nhập”

**Các chức năng: [Đăng nhập], [Tạo tài khoản]**

Chức năng [Đăng nhập]: cho phép đăng nhập tài khoản vào hệ thống tùy theo quyền hạn của tài khoản.

Validate:

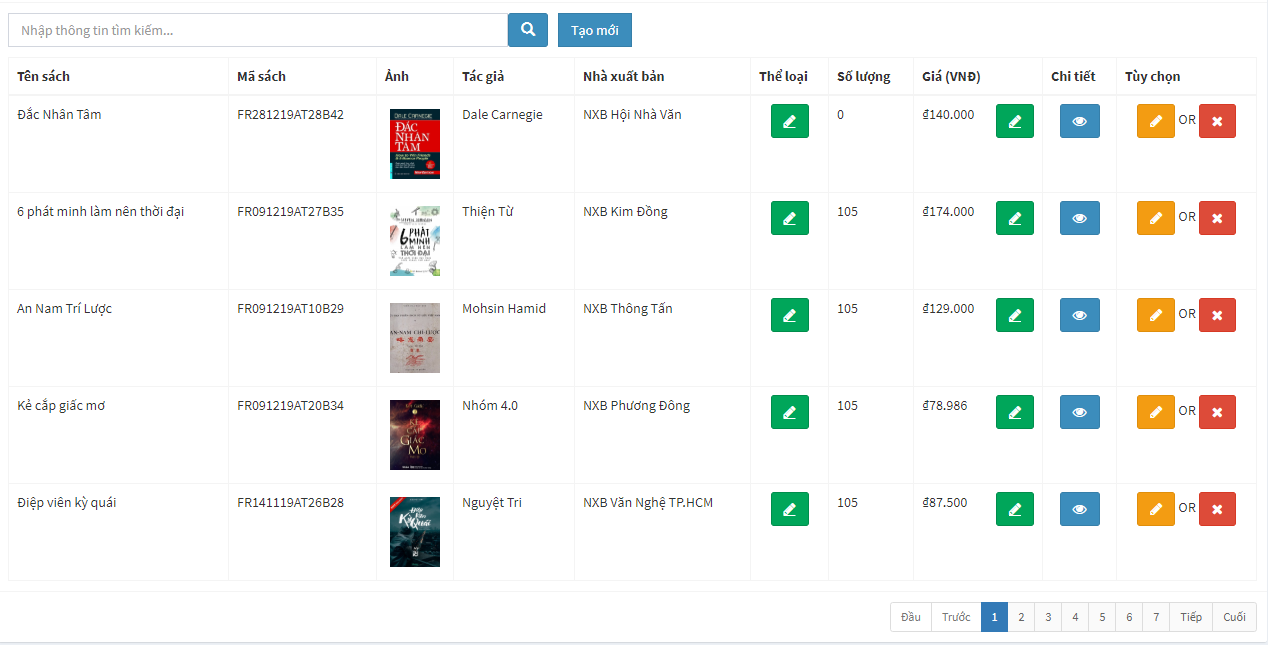
* Tên đăng nhập: length (50), khác rỗng
* Mật khẩu: kiểu mật khẩu (“\*\*\*\*\*”), length (32), khác rỗng

Chức năng: Người dùng nhập thông tin vào các ô nhập liệu và nhấn nút [Đăng nhập]

* Nếu có trường không hợp lệ về validate: hiện label màu đỏ ngay dưới ô nhập liệu không hợp lệ.
* Nếu [Tên đăng nhập] không tồn tại: hiện label màu đỏ dưới nút [Đăng ký] với nội dung: “Tên đăng nhập không tồn tại!”.
* Nếu [Tên đăng nhập] tồn tại nhưng mật khẩu sai, hiện label màu đỏ dưới nút [Đăng nhập] với nội dung: “Mật khẩu không khớp!”
* Nếu [Đăng nhập] thành công, chuyển người dùng đến trang tương ứng với quyền hạn của tài khoản (vd: với tài khoản có quyền hạn là admin hoặc nhân viên thì sẽ chuyển tới trang quản trị và hiển thị tên người đăng nhập, quyền hạn là khách hàng, sẽ chuyển tới trang khách hàng, tên khách được hiển thị).
* Khi đăng nhập, người dùng sẽ nhập mật khẩu và phần mật khẩu sẽ được mã hóa (MD5) rồi kiểm tra với CSDL.

Chức năng: [Tạo tài khoản]

* Khi nhấn vào nút [Tạo tài khoản!] hệ thống sẽ chuyển về trang đăng ký, tại đây người dùng có thể đăng ký tài khoản.
  + 1. Màn hình chức năng quản lý sách



* 1. Màn hình quản lý sách

**Các chức năng: [Tìm kiếm], [Tạo mới], [Sửa], [Xóa], [Xem chi tiết], [Cập nhật giá], [Cập nhật thể loại], [Chức năng phân trang]**

*Tiền điều kiện: đã đăng nhập với tài khoản admin hoặc nhân viên và đang ở View Quản lý sách*

Chức năng [Tìm kiếm]:

cho phép người dùng tìm kiếm chính xác theo: mã, tìm kiếm gần đúng theo: tên sách, tên tác giả, tên nhà xuất bản

Validate:

* Chuỗi tìm kiếm: minlength (0), maxlength (250)

Chức năng: Người dùng nhập thông tin vào ô tìm kiếm và nhấn nút **[Tìm kiếm]**

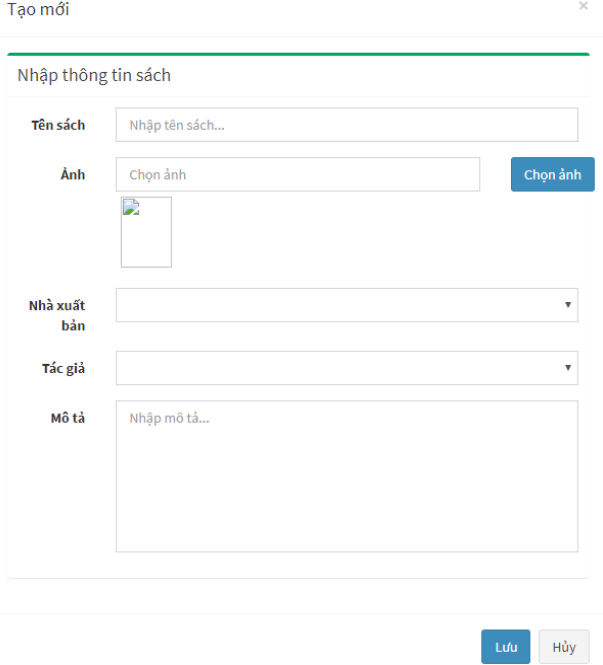
* Nếu ô tìm kiếm không hợp lệ về validate: hiện label màu đỏ ngay dưới ô tìm kiếm.
* Nếu chuỗi tìm kiếm thỏa mãn tìm kiếm gần đúng của 1 trong 3 thuộc tính: tên sách, tên tác giả, tên nhà xuất bản hoặc tìm kiếm chính xác: mã sách, thì hiển thị các đầu sách tương ứng dưới dạng table, kết quả có thể có nhiều bản ghi, nếu trên 5 bản ghi sẽ hiển thị phân trang.
* Nếu chuỗi tìm kiếm không thỏa mãn điều kiện tìm kiếm nào, hiển thị kết quả dưới dạng 1 table rỗng.
* Kết hợp kiểm thử chức năng tìm kiếm với phân trang.

Chức năng: [Phân trang] (kết hợp kiểm tra với chức năng tìm kiếm): cho phép hiển thị danh sách đầu sách dưới dạng các trang khi số bản ghi vượt quá 5 (mỗi trang sẽ hiển thị 5 bản ghi, nếu vượt quá 5 sẽ phân trang dữ liệu).

* TH1: số bản ghi < 5 bản ghi (tức có 1 trang):
  + chỉ hiển thị số trang: 1 dưới thanh phân trang, các nút “Đầu”, “Trước”, “Tiếp”, “Cuối” bị vô hiệu hóa
* TH2: số bản ghi > 5 bản ghi (tức có nhiều trang):
  + số trang tối đa sẽ hiển thị là: (số bản ghi/số bản ghi 1 trang) lấy phần nguyên + 1.
  + Tại trang 1: nút “Đầu”, “Trước” bị vô hiệu hóa, các nút khác trên thanh phân trang sử dụng bình thường, khi chọn “Tiếp” sẽ chuyển tới trang 2, khi chọn “Cuối” sẽ chuyển tới trang cuối cùng (tức số trang tối đa).
  + Tại trang n > 1: tất cả các nút tên thanh phân trang sử dụng bình thường, khi nhấn nút “Đầu” sẽ chuyển về trang 1, khi chọn “trước” sẽ chuyển về trang n-1, khi chọn “tiếp” sẽ chuyển tới trang n+1, khi chọn “cuối” sẽ chuyển tới trang cuối cùng.
  + Tại trang cuối cùng: nút “tiếp” và “cuối” bị vô hiệu hóa, khi chọn “trước” sẽ về trang n-1(tức trang trước trang cuối), khi chọn “đầu” sẽ về trang đầu tiên.

Chức năng [Tạo mới]:

Cho phép thêm sách



* 1. Ảnh minh hoa thêm sách

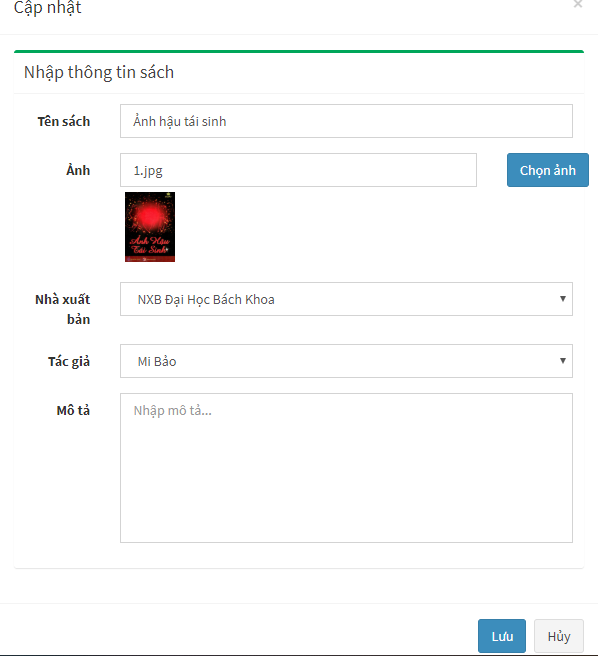
Validate:

* Tên sách: not null, maxlenght (250)
* Ảnh: not null, maxlenght (250)
* Nhà xuất bản: combobox, notnull
* Tác giả: combobox, notnull
* Mô tả: nvarchar(max), notnull

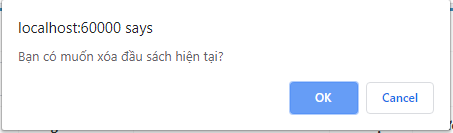
Chức năng:

* Người dùng chọn chức năng **[Tạo mới]**, một popup hiện ra như trên Hình 3-6, người dùng nhập/chọn thông tin vào các ô nhập liệu và nhấn nút **[Lưu]**
  + Nếu có trường không hợp lệ về validate: focus vào ô nhập liệu và hiển thị border màu đỏ tại ô nhập liệu đó.
  + Nếu thêm mới thất bại, hiển thị 1 alert: Thêm sách thất bại
* Nếu thêm mới thành công:
  + Lưu sách vào cơ sở dữ liệu, mã sách tự tăng, ngày tạo: ngày hiện tại, người tạo: lấy thông tin của tài khoản đang đăng nhập, metatile được tạo tự động (vd: tên sách là “Ảnh hậu tái sinh” thì metatile là: “anh-hau-tai-sinh”), giá sách = 0, khuyến mãi = 0%
  + 1 giá mới được tạo ra và lưu kèm với sách.
  + Mã sách được tạo theo công thức: “FR” + ngày + tháng + năm(tạo)+ “AT” + mã tác giả + “B” + mã sách.
  + Sách mới tạo sẽ không thuộc thể loại nào.
  + Hiển thị alert: Thêm sách thành công và load lại body của table (không load lại cả trang)
  + Khi người dùng nhấn dấu x, nút [Hủy] hoặc nhấn ra ngoài popup, popup biến mất, không có thay đổi trong CSDL.
    1. Chức năng [Sửa sách]:

Cho phép sửa sách

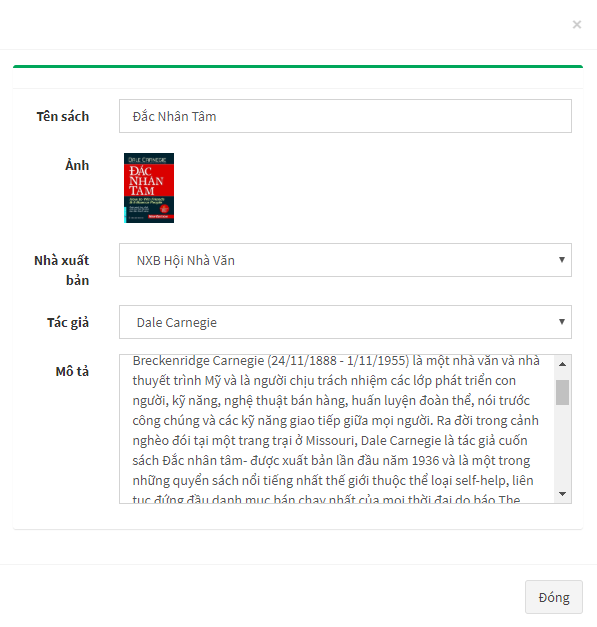


* 1. Ảnh minh họa chức năng sửa sách
* Validate:
  + Tên sách: not null, maxlenght (250)
  + Ảnh: not null, maxlenght (250)
  + Nhà xuất bản: combobox, notnull
  + Tác giả: combobox, notnull
  + Mô tả: nvarchar(max), notnull
* Chức năng:
  + Người dùng chọn chức năng **[Sửa sách]**, một popup hiện ra như trên Hình 3-5, thông tin về sách được hiển thị trên các trường, người dùng nhập/chọn thông tin vào các ô nhập liệu và nhấn nút **[Lưu]**
    - Nếu có trường không hợp lệ về validate: focus vào ô nhập liệu và hiển thị border màu đỏ tại ô nhập liệu đó.
    - Nếu thêm mới thất bại, hiển thị 1 alert: Sửa sách thất bại
  + Nếu thêm mới thành công:
    - Lưu sách vào cơ sở dữ liệu, ngày sửa: ngày hiện tại, người sửa: lấy thông tin của tài khoản đang đăng nhập, metatile được tạo tự động (vd: tên sách là “Ảnh hậu tái sinh” thì metatile là: “anh-hau-tai-sinh”)
    - Hiển thị alert: Sửa sách thành công và load lại body của table (không load lại cả trang)
  + Khi người dùng nhấn dấu x, nút [Hủy] hoặc nhấn ra ngoài popup, popup biến mất, không có thay đổi trong CSDL.
    1. Chức năng [Xóa sách]: cho phép xóa sách



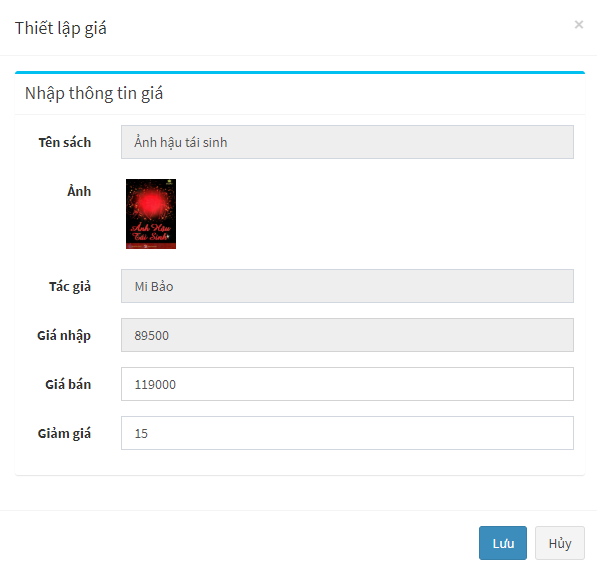
* 1. Ảnh minh họa chức năng xóa sách
* Chức năng:
  + người dùng chọn đầu sách muốn xóa và nhấn vào dấu x, 1 alert hiện ra với nội dung: Bạn có muốn xóa đầu sách hiện tại?
  + Người dùng chọn [OK], alert biến mất, hệ thống xóa đầu sách khỏi CSDL và load lại body của table
  + Người dùng chọn [Cancel], alert biến mất, không có đầu sách nào bị xóa khỏi CSDL
    1. Chức năng [Xem chi tiết]:

Cho phép xem chi tiết đầu sách.



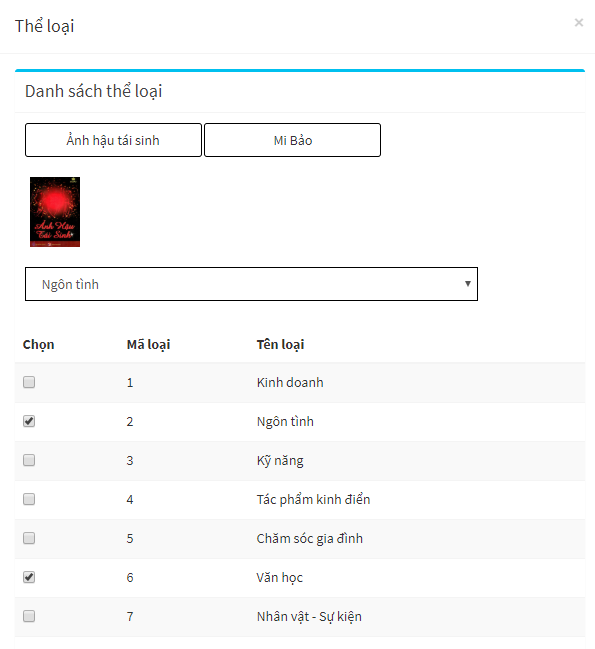
* 1. Ảnh minh họa chức năng xem chi tiết sách
* Người dùng chọn đầu sách muốn xóa và nhấn vào xem chi tiết, 1 popup hiện ra với thông tin sách đã chọn, các trường đều bị vô hiệu hóa.
* Người dùng chọn [Đóng], popup biến mất, không có thay đổi gì trong CSDL
  + 1. Chức năng [Cập nhật giá]

Cho phép cập nhật giá của sách:



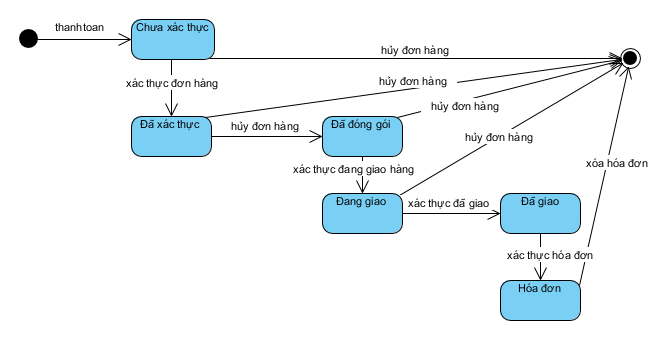
* 1. Ảnh minh họa chức năng cập nhật giá
* Validate:
  + Giá bán: double
  + Giảm giá: int, min: 0; max: 100
* Chức năng:
  + Người dùng chọn chức năng **[Cập nhật giá]**, một popup hiện ra như trên Hình 3-7, thông tin về sách được hiển thị trên các trường, các trường tên sách, tác giả, giá nhập bị vô hiệu hóa, người dùng nhập/chọn thông tin vào các ô nhập liệu và nhấn nút **[Lưu]**
    - Nếu có trường không hợp lệ về validate: focus vào ô nhập liệu và hiển thị border màu đỏ tại ô nhập liệu đó.
    - Nếu cập nhật thất bại, hiển thị 1 alert: Cập nhật giá thất bại
* Nếu thêm mới thành công:
  + Cập nhật sách vào cơ sở dữ liệu, ngày sửa: ngày hiện tại, người sửa: lấy thông tin của tài khoản đang đăng nhập, giảm giá = trường giảm giá đã nhập vào.
  + Hiển thị alert: “Cập nhật giá thành công” và load lại body của table (không load lại cả trang)
* Khi người dùng nhấn dấu x, nút [Hủy] hoặc nhấn ra ngoài popup, popup biến mất, không có thay đổi trong CSDL.
  + 1. Chức năng [Cập nhật thể loại]

Cho phép cập nhật thể loại cho mỗi đầu sách.

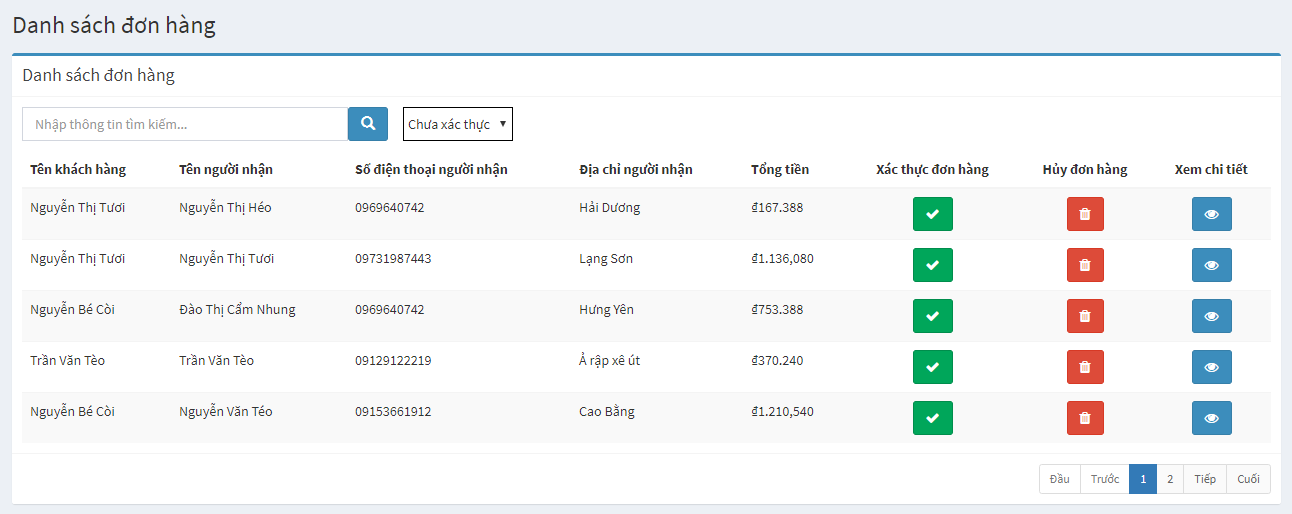


* 1. Ảnh minh họa chức năng cập nhật thể loại

* Người dùng chọn chức năng **[Cập nhật thể loại]**, một popup hiện ra như trên Hình 3-8, thông tin về sách được hiển thị trên các trường, các thể loại của sách được tích vào các dòng tương ứng, combobox chứa các thể loại của đầu sách đó, các trường tên sách, tác giả bị vô hiệu hóa
  + Người dùng tích vào trường muốn chọn, thông tin về thể loại đó được thêm vào combobox phía trên.
  + Người dùng bỏ tích vào 1 trường, thông tin về trường đó bị xóa khỏi combobox.
  + Người dùng nhấn [Lưu], lưu thể loại vào bảng sách thể loại, nếu thành công, hiển thị alert: Cập nhật thể loại thành công và load lại body của table, ngược lại hiển thị alert: Cập nhật thể loại thất bại và load lại trang đồng thời đóng popup.
  + Nếu người dùng dấu x, nút [Hủy] hoặc nhấn ra ngoài popup, popup biến mất, không có gì thay đổi trong CSDL.
    1. Quản lý quản lý đơn hàng



* 1. Biểu đồ trạng thái của đơn hàng

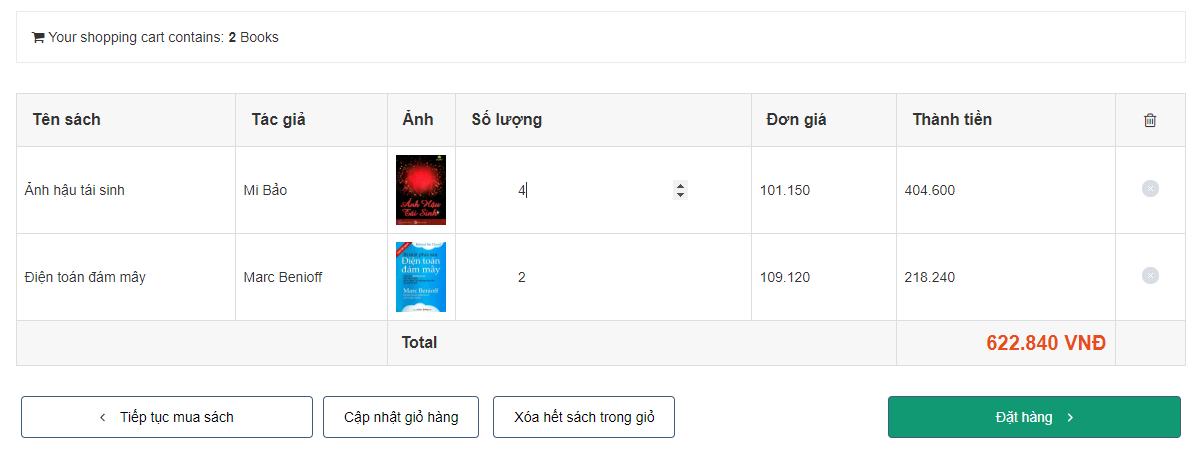


* 1. Màn hình quản lý đơn hàng

Các chức năng: Tìm kiếm, Xác thực đơn hàng, Xác thực đã đóng gói, Xác thực đang giao, Xác thực đã giao, Xác thực hóa đơn, Hủy đơn hàng, Xem chi tiết, Xóa hóa đơn, Chức năng phân trang, Xem đơn hàng theo trạng thái.

*Tiền điều kiện: đã đăng nhập với tài khoản admin hoặc nhân viên và đang ở View Quản lý đơn hàng*

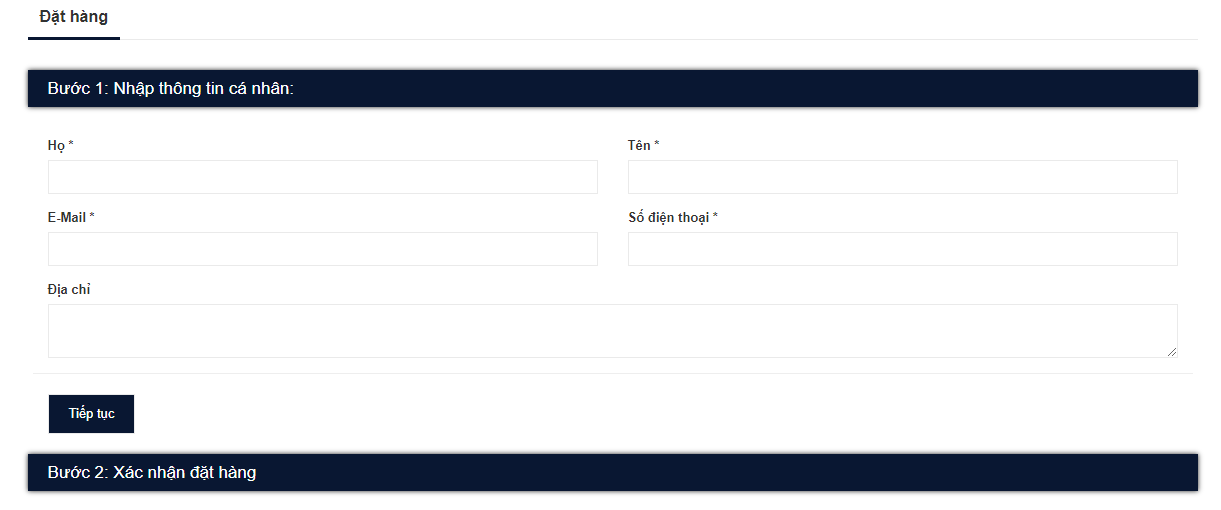
* Chức năng **[Tìm kiếm]**: cho phép người dùng tìm kiếm gần đúng theo: tên khách hàng, tên người nhận, địa chỉ người nhận; tìm kiếm chính xác theo: số điện thoại người nhận
  + Validate:
    - Chuỗi tìm kiếm: minlength (0), maxlength (250)
  + Chức năng: Người dùng nhập thông tin vào ô tìm kiếm và nhấn nút **[Tìm kiếm]**
    - Nếu ô tìm kiếm không hợp lệ về validate: hiện label màu đỏ ngay dưới ô tìm kiếm.
    - Nếu chuỗi tìm kiếm thỏa mãn tìm kiếm gần đúng của 1 trong 3 thuộc tính: tên sách, tên tác giả, tên nhà xuất bản hoặc tìm kiếm chính xác: số điện thoại, thì hiển thị các đầu sách tương ứng dưới dạng table, kết quả có thể có nhiều bản ghi, nếu trên 5 bản ghi sẽ hiển thị phân trang.
    - Nếu chuỗi tìm kiếm không thỏa mãn điều kiện tìm kiếm nào, hiển thị kết quả dưới dạng 1 table rỗng.
    1. Quản lý đặt hàng



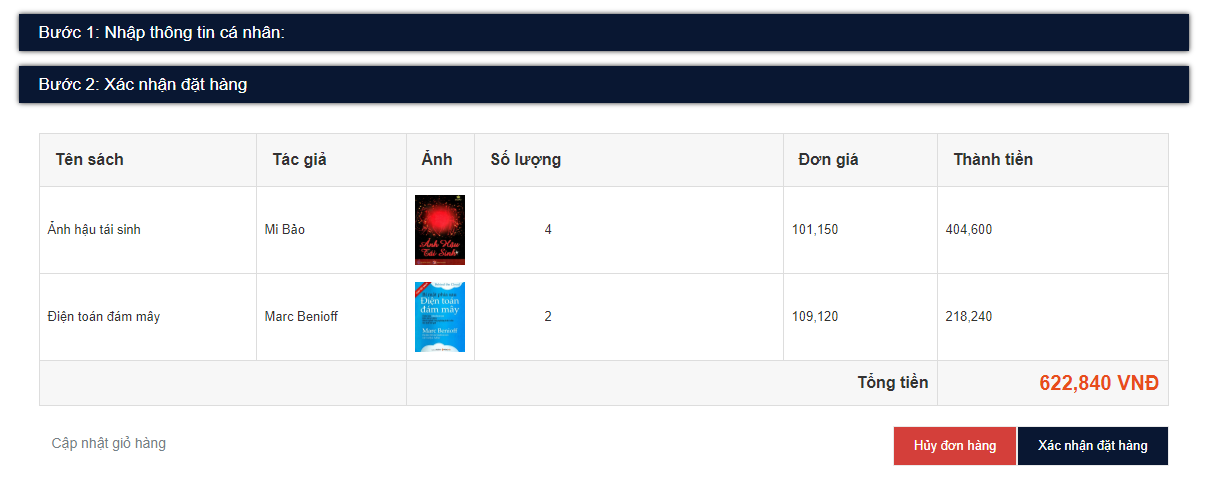
* 1. Minh họa chức năng đặt hàng

*Tiền điều kiện: đã đăng nhập với tài khoản khách hàng*

* Người dùng chọn đầu sách cần mua và nhấn thêm vào giỏ:
  + Nếu trong giỏ chưa có sách: đầu sách đã được thêm vào giỏ với số lượng là 1
  + Nếu trong giỏ đã có sách:
    - Nếu đầu sách đã chọn trùng với đầu sách đã có trong giỏ: số lượng đầu sách trong giỏ sẽ tăng thêm 1
    - Ngược lại: thêm 1 đầu sách vào giỏ với số lượng là 1
* Tại giỏ hàng: người dùng có thể: cập nhật giỏ hàng, xóa hết sách trong giỏ hàng, xóa 1 đầu sách chỉ định trong giỏ, tiếp tục mua sách, đặt hàng
  + Cập nhật giỏ hàng: người dùng có thể nhập số lượng sách thay đổi cho mỗi đầu sách vào các ô text (kiểu number -> dùng cho validate) và bấm vào cập nhật giỏ hàng, số lượng đầu sách trong giỏ sẽ thay đổi tương ứng.
  + Xóa hết sách trong giỏ hàng: khi chọn xóa hết sách trong giỏ thành công hiện 1 label chưa có đầu sách nào trong giỏ.
  + Xóa chỉ định 1 đầu sách trong giỏ: người dùng chọn đầu sách muốn xóa khỏi giỏ và bấm vào dấu x, đầu sách đã bị xóa khỏi giỏ
* Tiếp tục mua hàng: sau khi bấm vào, màn hình trở về trang chủ
* Đặt hàng: sau khi chọn thanh toán, màn hình sẽ chuyển tới trang thanh toán



* 1. Hình minh họa chức năng thanh toán
* Validate:
  + Họ, tên: length (50), khác rỗng
  + Số điện thoại: length (15), kiểu number, khác rỗng
  + Địa chỉ: length (500), khác rỗng
  + Email: length (50), kiểu email (“xyz@gmail.com”)
* Chức năng: Người dùng nhập các thông tin vào các trường nhập liệu
  + Nếu có trường không hợp lệ về validate, focus tại trường đó và viền đỏ xuất hiện
  + Ngược lại, người dùng bấm tiếp tục, sẽ chuyển xuống bảng thanh toán



* 1. Ảnh minh họa bảng thanh toán

Tại đây, người dùng có thể: cập nhật lại giỏ hàng, hủy đơn hàng, xác nhận thanh toán

* Hủy đơn hàng: người dùng bấm hủy đơn hàng, màn hình trở về trang chủ, các đầu sách trog giỏ bị xóa hết
* Cập nhật giỏ hàng: người dùng có thể nhập số lượng sách thay đổi cho mỗi đầu sách vào các ô text (kiểu number -> dùng cho validate) và bấm vào cập nhật giỏ hàng, số lượng đầu sách trong giỏ sẽ thay đổi tương ứng.
* Xác nhận thanh toán: sau khi bấm xác nhận thanh toán, người dùng sẽ nhận được 1 email về danh sách các đầu sách trong đơn hàng của mình và hiện 1 label: Thanh toán thành công, nếu thất bại sẽ hiện 1 label: thanh toán thất bại.

1. **Các yêu cầu phi chức năng**
2. ***Tính toàn vẹn dữ liệu***

Khi có nhiều hơn 1 user cùng thay đổi 1 data trong DB thì chỉ user đầu tiên thực hiện được bình thường. Đối với các user khác sẽ không lưu lại thay đổi mà hiển thị thông báo và yêu cầu thực hiện lại xử lý.

1. ***Performace***

Yêu cầu thời gian mở website con bất kỳ không được chậm hơn 3 giây, và xử lý lưu thông tin không được chậm hơn 4 giây.

# TRIỂN KHAI KIỂM THỬ TỰ ĐỘNG

* 1. Thiết kế các yêu cầu kiểm thử

Map

Description automatically generated

* 1. Test design Quản lý sách - Thêm mới

A picture containing diagram

Description automatically generated

* 1. Test design Quản lý sách – Sửa

Diagram

Description automatically generated

* 1. Test design Quản lý sách – Xoá

Map

Description automatically generated

* 1. Test design Quản lý sách – Cập nhật giá

Map

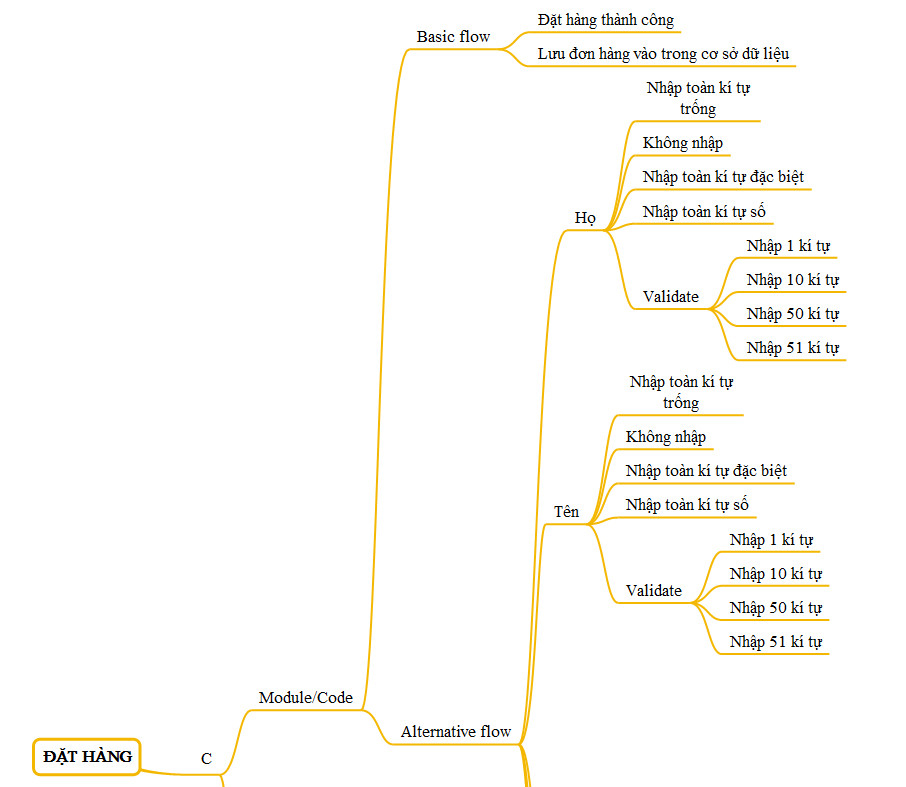
Description automatically generated with medium confidence

* 1. Test design Đăng nhập

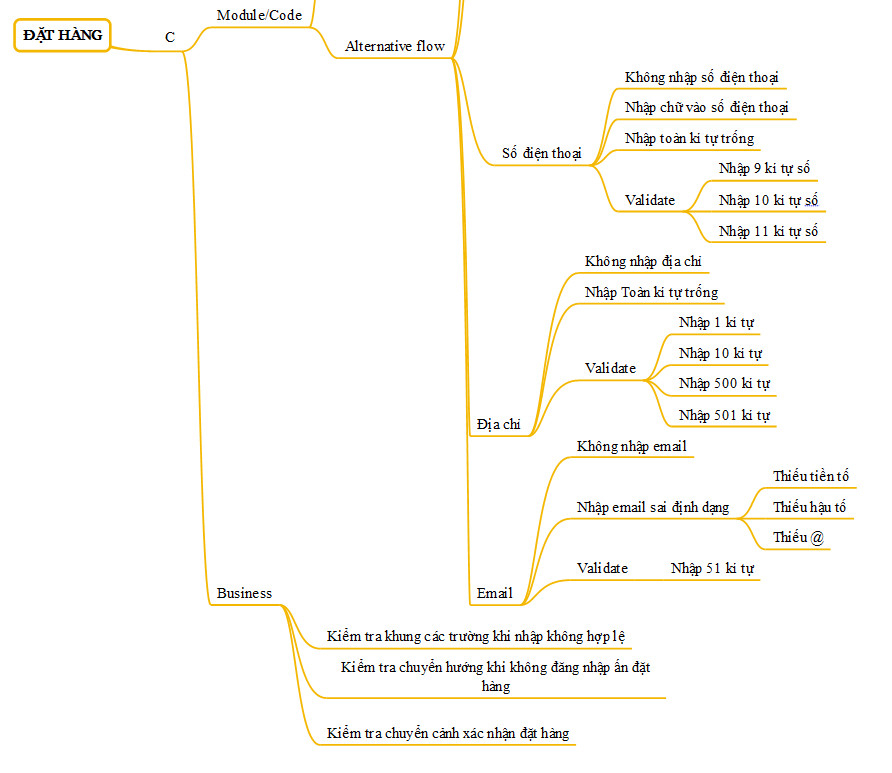
***A picture containing timeline

Description automatically generated***

* 1. Test design giỏ hàng



* 1. Test design Đặt hàng 1



* 1. Test design Đặt hàng 2
  2. Xây dựng ca kiểm thử
     1. Chức năng đăng nhập

|  |
| --- |
| @login Feature: login page test  *# đăng nhập thành công với tài khoản và mật khẩu chính xác* @basic  Scenario: Successfully logged in with an existing account  Given I navigate to admin page  When I enter an admin account "manh"  And I enter the password "manh123"  And I click on login button  Then I validate the title of website  And check validate the name of account  *# Không nhập tài khoản* @alternative  Scenario: Do not enter username  Given I navigate to admin page  And I enter the password "manh123"  And I click on login button  Then I validate the notification with value "Enter a valid username!"  *# Không nhập mật khẩu* @alternative  Scenario: Do not enter password  Given I navigate to admin page  And I enter an admin account "manh"  And I click on login button  Then I validate the notification with value "Enter a valid password!"  *# nhập tài khoản không tồn tại hoặc tài khoản tồn tại và mật khẩu sai* @alternative  Scenario Outline: Validate account and password  Given I navigate to admin page  When I enter an admin account "<username>"  And I enter the password "<password>"  And I click on login button  Then I validate the notification with value "<notification>"  Examples:  | *username* | *password* | *notification* |  *# Tài khoản không tồn tại* | manh123a | manh123 | Tài khoản không tồn tại! |  *# Mật khẩu sai* | manh | manhxxx | Mật khẩu không khớp! |  *# Tài khoản với độ rộng trên 50 kí tự* | 12345678901234567890123670 | manh123 | Tài khoản không được vượt quá 50 kí tự! |  *# Mật khẩu vượt quá 32 kí tự* | manh | 123456789012301234567890 | Mật khảu không được vượt quá 32 kí tự! | |

* + 1. Chức năng quản lý sản phẩm

|  |
| --- |
| @create Feature: Test function of product manage  @alternative  Scenario: Do not enter the name  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on save button  And I validate color of name textbox  @alternative  Scenario: Do not enter img link  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value "book name- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on save button  And I validate color of img textbox  @alternative  Scenario: Do not select the author  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value "book name- Hướng Test"  And Enter img link with value "iMG LINK- Hướng Test"  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on save button  And I validate color of author combobox  @alternative  Scenario: Do not select the publishing company  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value "book name- Hướng Test"  And Enter img link with value "IMG LINK- Hướng Test"  And Select the author  And Enter description with value "description- Hướng Test"  Then I click on save button  And I validate color of publish combobox  @alternative  Scenario: Do not enter the description  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value "book name- Hướng Test"  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  Then I click on save button  And I validate color of description combobox with "red"  @alternative  Scenario: Do not enter the name 2  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  And I click on save button  And I validate color of name textbox with "red"  And Enter book name with value "book name- Hướng Test"  And Clean text in descripton textbox  When I click on save button  And I validate color of description combobox with "red"  And I validate color of name textbox with "black"  @basic  Scenario Outline: Successfully added a new product  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value <string>  And Enter img link with value <string1>  And Select the author  And Select the publishing company  And Enter description with value <string2>  Then I click on save button  And Accept the alert  And I validate the name of product with value <string>  Examples:  | *string* | *string1* | *string2* |  | "book name- Hướng Test1" | "iMG LINK- Hướng Test" | "description- Hướng Test" |  | "book name- Hướng Test2" | "iMG LINK- Hướng Test" | "description- Hướng Test" |  @edit Feature: Test function of edit product  @alternative  Scenario: Do not enter the name  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value ""  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on update button  And I validate color of name textbox  @alternative  Scenario: Do not enter img link  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value "book name- Hướng Test"  And Enter img link with value ""  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on update button  And I validate color of img textbox   @alternative  Scenario: Do not enter the description  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value "book name- Hướng Test"  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value ""  Then I click on update button  And I validate color of description combobox with "red"  @alternative  Scenario: Do not enter the name 2  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value ""  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  And I click on update button  And I validate color of name textbox with "red"  And Enter book name with value "book name- Hướng Test"  And Clean text in descripton textbox  When I click on update button  And I validate color of description combobox with "red"  And I validate color of name textbox with "black"  @alternative  Scenario Outline: Successfully edit a product  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value <string>  And Enter img link with value <string1>  And Select the author  And Select the publishing company  And Enter description with value <string2>  Then I click on update button  And Accept the alert  And I validate the name of product with value <string>  Examples:  | *string* | *string1* | *string2* |  | "book name- Hướng Test1" | "iMG LINK- Hướng Test" | "description- Hướng Test" |  | "book name- Hướng Test2" | "iMG LINK- Hướng Test" | "description- Hướng Test" |  @edit Feature: Test function of update price  @basic  Scenario: Successfully update price  Given I login to the admin page  When Navigate to the product management page  And Click on the edit price button  And Enter the price with value "123454"  And Enter the discount with "11"  Then I click on update price button  And I validate alert notification with "Thiết lập giá thành công!"  And I accept alert  And I validate the price with value "123454"  @alternative  Scenario: invalid the price  Given I login to the admin page  When Navigate to the product management page  And Click on the edit price button  And Enter the price with value ""  And Enter the discount with "10"  Then I click on update price button  And I validate alert notification with "Thiết lập giá thất bại!"  And I accept alert   @alternative  Scenario Outline: invalid the discount  Given I login to the admin page  When Navigate to the product management page  And Click on the edit price button  And Enter the price with value "<price>"  And Enter the discount with "<discount>"  Then I click on update price button  And I validate color of discount with "<color>"  Examples:  | *price* | *discount* | *color* |  | 1000 | -1 | red |  | 1000 | 101 | red |  @alternative  Scenario: cancel update price  Given I login to the admin page  When Navigate to the product management page  And Click on the edit price button  And Enter the price with value "333333"  And Enter the discount with "10"  And I click on update price button  And I validate alert notification with "Thiết lập giá thành công!"  And I accept alert  And Click on the edit price button  And Enter the price with value "22222"  And Enter the discount with "10"  And I click on cancel update price button  And I validate the price with value "333333"  @alternative  Scenario: Do not enter the discount input  Given I login to the admin page  When Navigate to the product management page  And Click on the edit price button  And Enter the price with value "10000"  And Enter the discount with ""  Then I click on update price button  And I validate alert notification with "Thiết lập giá thất bại!"  And I accept alert  @alternative  Scenario: refresh before click on update price button  Given I login to the admin page  When Navigate to the product management page  And Click on the edit price button  And Enter the price with value "333333"  And Enter the discount with "10"  And I click on update price button  And I validate alert notification with "Thiết lập giá thành công!"  And I accept alert  And Click on the edit price button  And Enter the price with value "22222"  And Enter the discount with "10"  And I refresh this website  And I validate the price with value "333333"  @delete\_book Feature: Test delete function  @basic  Scenario: Successfully delete a book  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value "test xoa"  Then I click on update button  And Accept the alert  When Navigate to the product management page  And Click on the delete button of first book  And I accept alert  And I accept alert  And I validate the name of product different value "test xoa"  @alternative  Scenario: cancel delete a book  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value "test - Hướng"  Then I click on update button  And Accept the alert  When Navigate to the product management page  And Click on the delete button of first book  And I cancel alert  And I validate the name of product with value "test - Hướng"  @alternative  Scenario: refresh before accept delete a book  Given I login to the admin page  When Navigate to the product management page  And Click on the Edit button  And Enter book name with value "test - Hướng"  Then I click on update button  And Accept the alert  When Navigate to the product management page  And Click on the delete button of first book  And I refresh this website  And I validate the name of product with value "test - Hướng" |

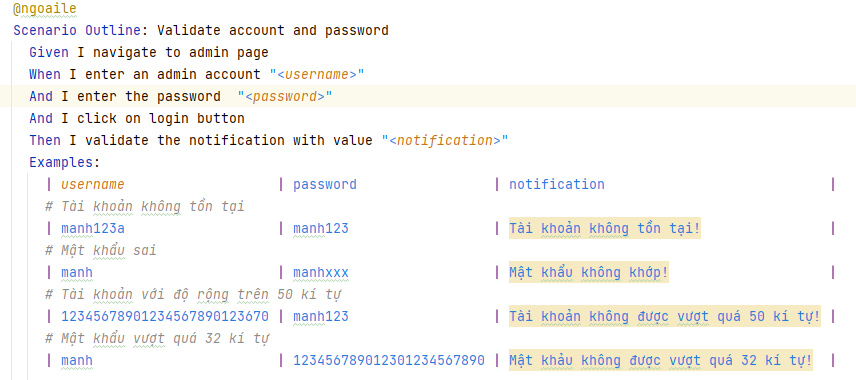
* + 1. Chức năng quản lý giỏ hàng

|  |
| --- |
| @edit Feature: Test function of shopping cart  @basic  Scenario: check item in cart when i add an item  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  When validate text next to cart icon with value "1 đầu sách"  @alternative  Scenario: check item in cart when we close browser and do not login  Given I navigate to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  When validate text next to cart icon with value "1 đầu sách"  And I close browser  And I navigate to the user page  When validate text next to cart icon with value ""  @alternative  Scenario: check item in cart when we close browser  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  When validate text next to cart icon with value "1 đầu sách"  And I close browser  And I login to the user page  When validate text next to cart icon with value "1 đầu sách"  @alternative  Scenario: check count of the item in the cart when we select that item twice  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  When validate text next to cart icon with value "2 đầu sách"  Then validate count of first book with "2"  And validate number of items in cart with "1"  And validate name of first item is "Đắc nhân tâm"  @alternative  Scenario: check the item in the cart when we select two different item  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  Then validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  @alternative  Scenario: check the delete all button  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  And validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  Then I click on Delete All  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng" |

* + 1. Chức năng quản lý đặt hàng

|  |
| --- |
| @order Feature: Test order function  @alternative  Scenario: do not enter surname of customer  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  And validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  When I click on Order button  And Enter the name with "a"  And Enter the email with "a"  And Enter the address with "a"  And Enter the phone number with "a"  Then I click on button continue  And i validate color of textbox surname with "red"  @alternative  Scenario: do not enter name of customer  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  And validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  When I click on Order button  And Enter the surname with "a"  And Enter the email with "a"  And Enter the address with "a"  And Enter the phone number with "a"  Then I click on button continue  And i validate color of textbox name with "red"  @alternative  Scenario: do not enter email of customer  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  And validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  When I click on Order button  And Enter the surname with "a"  And Enter the name with "a"  And Enter the address with "a"  And Enter the phone number with "a"  Then I click on button continue  And i validate color of textbox email with "red"  @alternative  Scenario: do not enter address of customer  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  And validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  When I click on Order button  And Enter the surname with "a"  And Enter the name with "a"  And Enter the email with "a"  And Enter the phone number with "a"  Then I click on button continue  And i validate color of textbox address with "red"  @alternative  Scenario: do not enter phone number of customer  Given I login to the user page  And Click on the cart button  And validate notification with value "Chưa có sản phẩm nào trong giỏ hàng"  And Enter search input with value "Đắc nhân tâm" and search  And I click on the book  And I click on add to card  And Enter search input with value "Loài người trong kỷ nguyên trí tuệ nhân tạo" and search  And I click on the book  And I click on add to card  And validate name of first item is "Đắc nhân tâm"  And validate the name of second item is "Loài người trong kỷ nguyên trí tuệ nhân tạo"  And validate count of first book with "1"  And validate number of items in cart with "2"  When I click on Order button  And Enter the surname with "a"  And Enter the name with "a"  And Enter the email with "a"  And Enter the address with "a"  Then I click on button continue  And i validate color of textbox phone number with "red" |

* 1. Xây dựng dữ liệu kiểm thử



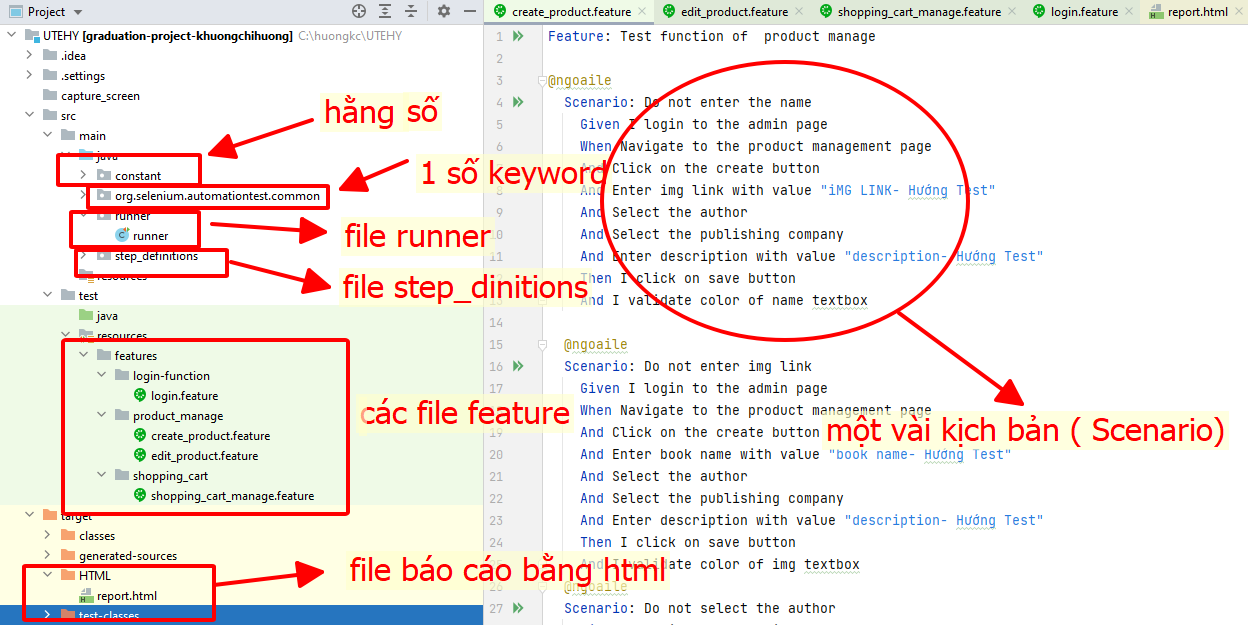
* 1. Dữ liệu kiểm thử cho chức năng đăng nhập



* 1. Dữ liệu kiểm thử cho chức năng tạo sách
  2. Phương pháp xây dựng Framework

Kiểm thử tự động hướng hành vi (BDD framework) bao gồm 2 phần chính:

* Feature file: Mỗi Feature gồm nhiều Scenario, bắt đầu bằng từ khóa “Feature:”. Mỗi Feature là 1 chức năng - Mỗi Scenario gồm nhiều step, bắt đầu bằng từ khóa “Scenario:”. Mỗi Scenario là một testcase. - Mỗi step sẽ bắt đầu bằng các keyword như Given, When, Then, But hoặc And Trong đó: - “Given”: Mô tả ngữ cảnh ban đầu của hệ thống - “When”: Mô tả hành vi - “Then”: Mô tả kết quả - “And”, “But”: Kết hợp nhiều step giống nhau Ví dụ về một file feature hoàn chỉnh
* Step Definition file: có thể viết bằng nhiều ngôn ngữ lập trình khác nhau. Hiện tại thì mình sử dụng ngôn ngữ Java. Trong file này, có các hàm tương ứng với các step ở “**Feature file**” để giúp chương trình chạy theo đúng kịch bản mà chúng ta muốn.



* + 1. Xây dựng lớp CommonBase

**Ý nghĩa:** Lớp CommonBase cung cấp các keyword hỗ trợ cho file step\_definitions. Dưới đây là code của lớp CommonBase

|  |
| --- |
| package org.selenium.automationtest.common;  import java.io.File; import java.net.Inet4Address; import java.text.DateFormat; import java.text.SimpleDateFormat; import java.util.ArrayList; import java.util.Calendar; import java.util.List;  import io.cucumber.java.After; import org.apache.commons.io.FileUtils; import org.openqa.selenium.Alert; import org.openqa.selenium.JavascriptExecutor; import org.openqa.selenium.Keys; import org.openqa.selenium.NoAlertPresentException; import org.openqa.selenium.NoSuchElementException; import org.openqa.selenium.By; import org.openqa.selenium.ElementNotVisibleException; import org.openqa.selenium.OutputType; import org.openqa.selenium.StaleElementReferenceException; import org.openqa.selenium.WebDriver; import org.openqa.selenium.remote.RemoteWebDriver; import org.openqa.selenium.WebDriverException; import org.openqa.selenium.WebElement; import org.openqa.selenium.chrome.ChromeDriver; import org.openqa.selenium.chrome.ChromeOptions; import org.openqa.selenium.firefox.FirefoxDriver; import org.openqa.selenium.firefox.FirefoxProfile; import org.openqa.selenium.ie.InternetExplorerDriver; import org.openqa.selenium.interactions.Actions; import org.openqa.selenium.remote.DesiredCapabilities; import org.openqa.selenium.safari.SafariDriver; import org.openqa.selenium.support.ui.ExpectedCondition; import org.openqa.selenium.support.ui.ExpectedConditions; import org.openqa.selenium.support.ui.Select; import org.openqa.selenium.support.ui.WebDriverWait; import org.openqa.selenium.TakesScreenshot; import org.testng.Assert; import org.testng.annotations.AfterTest; import static org.selenium.automationtest.common.TestLogger.\*;  public class CommonBase {  public WebDriver driver;  protected String baseCoreSolr = "dbmapactiveinfo";  protected int DEFAULT\_TIMEOUT = 180000;  protected int WAIT\_INTERVAL = 1000;  public int loopCount = 0;  public final int ACTION\_REPEAT = 5;  public Actions action;  public void waitForPageLoaded(WebDriver driver) {  ExpectedCondition<Boolean> expectation = new ExpectedCondition<Boolean>() {  public Boolean apply(WebDriver driver) {  return ((JavascriptExecutor) driver)  .executeScript("return document.readyState").toString()  .equals("complete");  }  };  try {  Thread.*sleep*(1000);  WebDriverWait wait = new WebDriverWait(driver, (DEFAULT\_TIMEOUT / 1000));  wait.until(expectation);  } catch (Throwable error) {  Assert.*fail*("Timeout khi cho trang web hoan thanh load");  }  }  */\*\*  \* Open page  \*  \* @param pageUrl  \* @param driver  \*/* public void openPage(String pageUrl, WebDriver driver) {  if (pageUrl != null) {  driver.get(pageUrl);  }  waitForPageLoaded(driver);  pause(1000);  }  */\*\*  \* Open page at not loaded status, as clear cache  \*  \* @param pageUrl  \* @param driver  \*/* public void openPageNotLoad(String pageUrl, WebDriver driver) {  if (pageUrl != null) {  driver.get(pageUrl);  pause(2000);  }  }  */\*\*  \* pause driver in timeInMillis  \*  \* @param timeInMillis  \*/* public void pause(long timeInMillis) {  try {  Thread.*sleep*(timeInMillis);  } catch (InterruptedException e) {  e.printStackTrace();  }  }  */\*\*  \* @param locator  \* @return  \*/* public WebElement getElement(Object locator) {  By by = locator instanceof By ? (By) locator : By.*xpath*(locator  .toString());  WebElement elem = null;  try {  elem = driver.findElement(by);  } catch (NoSuchElementException e) {  checkCycling(e, 10);  pause(WAIT\_INTERVAL);  getElement(locator);  } catch (StaleElementReferenceException e) {  checkCycling(e, 10);  pause(WAIT\_INTERVAL);  getElement(locator);  }  return elem;  }  */\*\*  \* get a display element in web page  \*  \* @param locator  \* @return  \*/* public WebElement getDisplayedElement(Object locator) {  By by = locator instanceof By ? (By) locator : By.*xpath*(locator  .toString());  WebElement e = null;  try {  if (by != null)  e = driver.findElement(by);  if (e != null) {  if (isDisplay(by))  return e;  }  } catch (NoSuchElementException ex) {  checkCycling(ex, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  getDisplayedElement(locator);  } catch (StaleElementReferenceException ex) {  checkCycling(ex, 10);  pause(WAIT\_INTERVAL);  getDisplayedElement(locator);  } finally {  loopCount = 0;  }  return null;  }  */\*\*  \* @param locator  \* @return  \*/* public List<WebElement> getListElement(Object locator) {  By by = locator instanceof By ? (By) locator : By.*xpath*(locator  .toString());  List<WebElement> elementOptions;  try {  elementOptions = driver.findElements(by);  return elementOptions;  } catch (NoSuchElementException ex) {  checkCycling(ex, 10);  pause(WAIT\_INTERVAL);  getListElement(locator);  } catch (StaleElementReferenceException ex) {  checkCycling(ex, 10);  pause(WAIT\_INTERVAL);  getListElement(locator);  } finally {  loopCount = 0;  }  return null;  }  */\*\*  \* lay cac gia tri thuoc tinh cua 1 mang doi tuong element  \*  \* @param locator  \* @param attribute  \* @return  \*/* public String[] getAttOfListElement(Object locator, String attribute) {  String[] att = new String[20];  List<WebElement> list;  list = getListElement(locator);  if (list.size() > 0) {  for (int i = 0; i < list.size(); i++) {  att[i] = list.get(i).getAttribute(attribute);  }  }  return att;  }  public String[] getTextOfListElement(Object locator) {  String[] att = new String[20];  List<WebElement> list;  list = getListElement(locator);  if (list.size() > 0) {  for (int i = 0; i < list.size(); i++) {  att[i] = list.get(i).getText();  }  }  return att;  }  public String getSizeOfListElement(Object locator) {  return String.*valueOf*(getListElement(locator).size());  }  */\*\*  \* checking an element is displayed in web page  \*  \* @param locator  \* @return  \*/* public boolean isDisplay(Object locator) {  boolean bool = false;  WebElement e = getElement(locator);  try {  if (e != null)  bool = e.isDisplayed();  } catch (StaleElementReferenceException ex) {  checkCycling(ex, 10);  pause(WAIT\_INTERVAL);  isDisplay(locator);  } finally {  loopCount = 0;  }  return bool;  }  */\*\*  \* check repeat times  \*  \* @param e  \* @param loopCountAllowed  \*/* public void checkCycling(Exception e, int loopCountAllowed) {  *info*("Co exception xay ra: " + e.getClass().getName());  if (loopCount > loopCountAllowed) {  driver.manage().deleteAllCookies();  Assert.*fail*("Qua thoi gian nhung khong thay hoac thay doi tuong "  + e.getMessage());  }  *info*("Lap lai lan thu " + loopCount);  loopCount++;  }  */\*\*  \* get an element that present in Web Page  \*  \* @param locator  \* @param opParams  \* @return  \*/* public WebElement getElementPresent(Object locator, int... opParams) {  WebElement elem = null;  int timeout = opParams.length > 0 ? opParams[0] : DEFAULT\_TIMEOUT;  int isAssert = opParams.length > 1 ? opParams[1] : 1;  int notDisplayE = opParams.length > 2 ? opParams[2] : 0;  for (int tick = 0; tick < timeout / WAIT\_INTERVAL; tick++) {  if (notDisplayE == 2) {  elem = getElement(locator);  } else {  elem = getDisplayedElement(locator);  }  if (null != elem)  return elem;  pause(WAIT\_INTERVAL);  }  if (isAssert == 1) {  String date = getDateTime("yyyyMMddHHmmss");  *info*("date");  captureScreen(driver, "Loi\_" + date + ".jpg");  assert false : ("Qua thoi gian " + timeout  + "ma khong tim thay doi tuong " + locator);  quitDriver(driver);  }  return null;  }  public void type(Object object, String value) {  WebElement element = getElement(object);  WebDriverWait wait = new WebDriverWait(driver, 10);  try {  *// WebElement element = getElementPresent(locator, 10000, 0);* if (element != null) {  wait.until(ExpectedConditions.*visibilityOf*(element));  element.clear();  element.sendKeys(value);  } else {  wait.until(ExpectedConditions.*visibilityOf*(element));  element.sendKeys(value);  }  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  type(element, value);  } catch (NoSuchElementException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  type(element, value);  } catch (ElementNotVisibleException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  type(element, value);  } finally {  loopCount = 0;  }  }  */\*\*  \* @param locator  \* @param value  \*/* public void inputTextJavaScript(Object locator, String value) {  WebElement e = getElementPresent(locator, DEFAULT\_TIMEOUT, 1, 2);  try {  ((JavascriptExecutor) driver).executeScript(  "arguments[0].innerHTML = '" + value + "'", e);  } catch (StaleElementReferenceException ex) {  pause(1000);  inputTextJavaScript(locator, value);  }  }  */\*\*  \* get value of element in web page  \*  \* @param locator  \* @return  \*/* public String getValue(Object locator, Object... opParams) {  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  try {  return getElementPresent(locator, DEFAULT\_TIMEOUT, 1, notDisplay)  .getAttribute("value");  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  return getValue(locator);  } finally {  loopCount = 0;  }  }  */\*\*  \* click on an element  \*  \* @param locator  \* @param opParams  \*/* public void click(Object locator, Object... opParams) {  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  Actions actions = new Actions(driver);  try {  WebElement element = getElementPresent(locator, DEFAULT\_TIMEOUT, 1,  notDisplay);  if (element.isEnabled()) {  actions.click(element).perform();  } else {  *info*("Element is not enabled");  *// click(locator, notDisplay);* }  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  click(locator, notDisplay);  } catch (ElementNotVisibleException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  click(locator, notDisplay);  } catch (NoSuchElementException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  click(locator, notDisplay);  } finally {  loopCount = 0;  }  }  public String getAlertText() {  *//AlertPresent();* String text = "";  try {  Alert alert = driver.switchTo().alert();  text = alert.getText();   } catch (NoAlertPresentException ex) {  *info*("No Alert present");  ;  }  return text;  }  public void AlertPresent() {  WebDriverWait wait = new WebDriverWait(driver, 300 */\*timeout in seconds\*/*);  if (wait.until(ExpectedConditions.*alertIsPresent*()) == null)  System.*out*.println("alert was not present");  else  System.*out*.println("alert was present");  }  public void clickJavascript(Object locator, Object... opParams) {  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  try {  WebElement element = getElementPresent(locator, DEFAULT\_TIMEOUT, 1,  notDisplay);  ((JavascriptExecutor) driver).executeScript(  "arguments[0].scrollIntoView(true);arguments[0].click();",  element);  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  clickJavascript(locator, opParams);  }  }  */\*\*  \* @param locator  \* @param opParams  \* @return  \*/* public WebElement waitForElementNotPresent(Object locator, int... opParams) {  WebElement elem = null;  int timeout = opParams.length > 0 ? opParams[0] : DEFAULT\_TIMEOUT;  int isAssert = opParams.length > 1 ? opParams[1] : 1;  int notDisplayE = opParams.length > 2 ? opParams[2] : 0;  for (int tick = 0; tick < timeout / WAIT\_INTERVAL; tick++) {  if (notDisplayE == 2) {  elem = getElement(locator);  } else {  elem = getDisplayedElement(locator);  }  if (elem == null) {  return null;  }  pause(WAIT\_INTERVAL);  }  if (isAssert == 1) {  assert false : ("Timeout after " + timeout  + "ms waiting for element not present: " + locator);  }  return elem;  }  */\*\*  \* @param locator  \* @param opParams  \*/* public void check(Object locator, int... opParams) {  int notDisplayE = opParams.length > 0 ? opParams[0] : 0;  Actions actions = new Actions(driver);  try {  WebElement element = getElementPresent(locator, DEFAULT\_TIMEOUT, 1,  notDisplayE);  boolean a = element.getAttribute("class").contains(  "ui-state-active");  if (!element.isSelected() && !a) {  actions.click(element).perform();  } else {  *info*("Element " + locator + " is already checked.");  }  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  check(locator);  } finally {  loopCount = 0;  }  }  */\*\*  \* @param locator  \* @param opParams  \*/* public void uncheck(Object locator, int... opParams) {  int notDisplayE = opParams.length > 0 ? opParams[0] : 0;  Actions actions = new Actions(driver);  try {  WebElement element = getElementPresent(locator, DEFAULT\_TIMEOUT, 1,  notDisplayE);  if (element.isSelected()) {  actions.click(element).perform();  } else {  *info*("Element " + locator + " is already unchecked.");  }  } catch (StaleElementReferenceException e) {  checkCycling(e, 5);  pause(1000);  uncheck(locator);  } finally {  loopCount = 0;  }  }  public void cancelAlert() {  try {  Alert alert = driver.switchTo().alert();  alert.dismiss();   } catch (NoAlertPresentException ex) {  *info*("No Alert present");  ;  }  }  */\*\*  \* get absolute path of file  \*  \* @param relativeFilePath  \* @return  \*/* public String getAbsoluteFilePath(String relativeFilePath) {  String curDir = System.*getProperty*("user.dir");  String absolutePath = curDir + relativeFilePath;  return absolutePath;  }  */\*\*  \* @param locator  \*/* public void doubleClickOnElement(Object locator) {  Actions actions = new Actions(driver);  try {  WebElement element = getElementPresent(locator);  actions.doubleClick(element).perform();  } catch (StaleElementReferenceException e) {  checkCycling(e, 5);  pause(1000);  doubleClickOnElement(locator);  } finally {  loopCount = 0;  }  }  */\*\*  \* get text of element  \*  \* @param locator  \* @return  \*/* public String getText(Object locator) {  WebElement element = null;  try {  element = getElementPresent(locator);  return element.getText();  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  return getText(locator);  } finally {  loopCount = 0;  }  }  */\*\*  \* @param locator  \* @param safeToSERE  \* @param opParams  \*/* public void mouseOver(Object locator, boolean safeToSERE,  Object... opParams) {  WebElement element;  Actions actions = new Actions(driver);  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  try {  if (safeToSERE) {  for (int i = 1; i < ACTION\_REPEAT; i++) {  *info*("Thuc hien mouserover repeat lan thu " + i);  element = getElementPresent(locator, 2000, 0, notDisplay);  *info*("Doi tuong " + element);  if (element == null) {  pause(WAIT\_INTERVAL);  } else {  *info*("Thuc hien action");  actions.moveToElement(element).build().perform();  break;  }  }  } else {  element = getElementPresent(locator);  actions.moveToElement(element).build().perform();  }  } catch (StaleElementReferenceException e) {  checkCycling(e, DEFAULT\_TIMEOUT / WAIT\_INTERVAL);  pause(WAIT\_INTERVAL);  mouseOver(locator, safeToSERE);  } finally {  loopCount = 0;  }  }   */\*\*  \* @param locator  \* @param opParams  \*/* public void mouseOverAndClick(Object locator, Object... opParams) {  WebElement element;  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  Actions actions = new Actions(driver);  try {  element = getElementPresent(locator, DEFAULT\_TIMEOUT, 1, notDisplay);  actions.moveToElement(element).click(element).build().perform();  } catch (StaleElementReferenceException e) {  mouseOverAndClick(locator, opParams);  }  }  */\*\*  \* quit driver if driver existed  \*  \* @param dr  \*/* public void quitDriver(WebDriver dr) {  if (dr.toString().contains("null")) {  System.*out*.print("All Browser windows are closed ");  } else {  driver.manage().deleteAllCookies();  dr.quit();  }  }  */\*\*  \* switch to a frame  \*  \* @param locator  \* @param opParams  \*/* public void switchToFrame(Object locator, Object... opParams) {  *info*("Switch to frame " + locator);  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  try {  driver.switchTo().frame(  getElementPresent(locator, DEFAULT\_TIMEOUT, 1, notDisplay));  } catch (Exception e) {  switchToFrame(locator, notDisplay);  }  }  */\*\*  \* back to main frame  \*/* public void switchToParentFrame() {  try {  driver.switchTo().defaultContent();  } catch (Exception e) {  switchToParentFrame();  }  }  */\*\*  \* accept unexpected alert  \*/* public void acceptAlert() {  try {  Alert alert = driver.switchTo().alert();  alert.accept();   } catch (NoAlertPresentException ex) {  *info*("No Alert present");  ;  }  }  */\*\*  \* get datetime  \*  \* @param format  \*/* public String getDateTime(String format) {  DateFormat dateFormat = new SimpleDateFormat(format);  Calendar cal = Calendar.*getInstance*();  String dateTime = dateFormat.format(cal.getTime());  *info*("time at moment is " + dateTime);  return dateTime;  }  */\*\*  \* @param object  \*/* public void clickTab(By object) {  if (object != null) {  WebElement e = getElementPresent(object);  e.sendKeys(Keys.*TAB*);  }  }  */\*\*  \* @param xpath  \* @param att  \* @return  \*/* public String getAttributeFromJavaScript(String xpath, String att) {  WebElement e = getElementPresent(xpath);  JavascriptExecutor executor = (JavascriptExecutor) driver;  String value = (String) executor.executeScript(  " return arguments[0].getAttribute('" + att + "')", e);  *info*("value" + value);  return value;  }  */\*\*  \* @return  \*/* public String getIpOfMachinhe() {  String ip = "";  try {  ip = Inet4Address.*getLocalHost*().getHostAddress();  *info*("IP of local machine: " + ip);  } catch (Exception e) {  *info*("Exeption: " + e);  }  return ip;  }  */\*\*  \* @param locator  \* @param opParams  \*/* public void scrollToElement(Object locator, Object... opParams) {  int notDisplay = (Integer) (opParams.length > 0 ? opParams[0] : 0);  WebElement element = getElementPresent(locator, DEFAULT\_TIMEOUT, 1,  notDisplay);  ((JavascriptExecutor) driver).executeScript(  "arguments[0].scrollIntoView(true);", element);  }  */\*\*  \* type cho cac input dang so  \*  \* @param locator  \* @param text  \* @param validate  \*/* public void typeHinput(Object locator, String text, boolean validate) {  if (text != null) {  if (getElementPresent(locator) != null) {  try {  WebElement e = getElementPresent(locator);  e.click();  e.sendKeys(Keys.*CONTROL* + "a");  e.sendKeys(text);  for (int i = 0; i < 5; i++) {  String am = getValue(locator, 2);  if (am != null) {  if (am.equalsIgnoreCase(text)) {  break;  } else {  e.sendKeys(Keys.*CONTROL* + "a");  e.sendKeys(text);  }  }  }  } catch (StaleElementReferenceException ex) {  typeHinput(locator, text, validate);  }  }  }  }   */\*\*  \* compare 2 string  \*  \* @param s1  \* @param s2  \*/* public void verifyCompare(String s1, String s2) {  if (s1 != "" && s1 != null && s2 != null && s2 != "") {  Assert.*assertFalse*(!s1.equalsIgnoreCase(s2),  "So sanh khong bang nhau: " + s1 + " va " + s2);  } else if ((s1 == "" || s1 == null) && (s2 == "" || s2 == null)) {  *info*("2 truong du lieu can so sanh deu null");  } else {  Assert.*fail*("Du lieu so sanh co 1 truong bi null");  }  }   public void verifyNotEqual(String s1, String s2) {  if (s1 != "" && s1 != null && s2 != null && s2 != "") {  Assert.*assertFalse*(s1.equalsIgnoreCase(s2),  "So sanh bang nhau: " + s1 + " va " + s2);  } else if ((s1 == "" || s1 == null) && (s2 == "" || s2 == null)) {  *info*("2 truong du lieu can so sanh deu null");  } else {  Assert.*fail*("Du lieu so sanh co 1 truong bi null");  }  }  public void verifyContains(String s1, String s2) {  if (s1 != null && s2 != null) {  Assert.*assertFalse*(!s2.contains(s1), "Chuỗi " + s1  + " không nằm trong chuỗi " + s2);  } else {  Assert.*assertFalse*(!s2.contains(s1), "Chuỗi " + s1  + " không nằm trong chuỗi " + s2);  }  }  */\*\*  \* @param xpath  \* @param option  \*/* public void selectOptionFromCombobox(String xpath, String option) {  if (option != null) {  String locator = xpath.replaceAll("&option", option);  click(locator);  waitForElementNotPresent(locator, 10000, 0);  }  }  public void captureScreen(WebDriver driver, String fileName) {  try {  File scrFile = ((TakesScreenshot) driver)  .getScreenshotAs(OutputType.*FILE*);  String dir = System.*getProperty*("user.dir");  FileUtils.*copyFile*(scrFile, new File(dir + "\\capture\_screen\\"  + fileName));  } catch (Exception e) {  *info*("Khong capture duoc man hinh");  }  }  public String trimCharactor(String input, String trim) {  *info*("Xau can xu ly trim: " + input);  if (input != "" && input != null && trim != "") {  if (trim == ".") {  return input.replaceAll("\\.", "");  } else {  return input.replaceAll(trim, "");  }  } else  return "";  }  */\*\*  \* @param file  \* @param filePath  \*/* public void uploadFile(Object file, String filePath) {  WebElement e = getElement(file);  *info*("Upload file "  + getAbsoluteFilePath("\\file\_to\_upload\\" + filePath));  e.sendKeys(getAbsoluteFilePath("\\file\_to\_upload\\" + filePath));  }  public void selectOptionByValue(Object cbb, String value) {  WebElement e = getElementPresent(cbb);  Select dropdown = new Select(e);  dropdown.selectByValue(value);  }  public static Boolean isVisibleInViewport(WebElement element, WebDriver driver) {  return (Boolean) ((JavascriptExecutor) driver).executeScript(  "var elem = arguments[0], " +  " box = elem.getBoundingClientRect(), " +  " cx = box.left + box.width / 2, " +  " cy = box.top + box.height / 2, " +  " e = document.elementFromPoint(cx, cy); " +  "for (; e; e = e.parentElement) { " +  " if (e === elem) " +  " return true; " +  "} " +  "return false; "  , element);  }  public void switchNewTab(int... index) {  int tab = index.length > 0 ? index[0] : 1;  ArrayList<String> tabs2 = new ArrayList<String>(  driver.getWindowHandles());  driver.switchTo().window(tabs2.get(tab));  }  public void verifyNotContains(String s1, String s2) {  if (s1 != null && s2 != null && s2.contains(s1)) {  *info*("Fail do chuoi " + s2 + " van chua chuoi " + s1);  Assert.*assertFalse*(s2.contains(s1));  }  }  public void enter(Object locator) {  if (locator != null) {  WebElement e = getElementPresent(locator);  e.sendKeys(Keys.*ENTER*);  }  } } |

* + 1. Xây dựng lớp constant

**Ý nghĩa:** lưu một số hằng số, biến mặc định cho dự án.

Dưới đây là code của lớp constant:

|  |
| --- |
| package constant; public class base\_constant {  public final static String *ADMIN\_URL*="http://113.160.133.144:20201/Admin/Login";  public final static String *USER\_URL*="http://113.160.133.144:20201";  public final static String *PRODUCT\_MANAGE\_URL*="http://113.160.133.144:20201/Admin/Book/Manage";  public final static String *ADMIN\_ID*="manh";  public final static String *ADMIN\_PASSWORD*="manh123";  public final static String *USER\_ID*="huongkhuong";  public final static String *USER\_PASSWORD*="huong123"; } |

* + 1. Xây dựng các Feature file

Ý nghĩa: đưa ra kịch bản các luồng nghiệp vụ để từ đó người kiểm thử tự động sẽ có thể viết các step\_definitions tương ứng đúng như mong muốn của kịch bản.

Dưới đây là 1 file Feature mẫu:

|  |
| --- |
| Feature: Test function of product manage @ngoaile  Scenario: Do not enter the name  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter img link with value "iMG LINK- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on save button  And I validate color of name textbox  @ngoaile  Scenario: Do not enter img link  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value "book name- Hướng Test"  And Select the author  And Select the publishing company  And Enter description with value "description- Hướng Test"  Then I click on save button  And I validate color of img textbox  @thanhcong Scenario Outline: Successfully added a new product  Given I login to the admin page  When Navigate to the product management page  And Click on the create button  And Enter book name with value <string>  And Enter img link with value <string1>  And Select the author  And Select the publishing company  And Enter description with value <string2>  Then I click on save button  And Accept the alert  And I validate the name of product with value <string>  Examples:  | *string* | *string1* | *string2* |  | "book name- Hướng Test1" | "iMG LINK- Hướng Test" | "description- Hướng Test" |  | "book name- Hướng Test2" | "iMG LINK- Hướng Test" | "description- Hướng Test" | |

* + 1. Xây dựng lớp step\_definitions

**Ý nghĩa**: Lớp này xây dựng các hàm tương ứng với từng bước ở **Feature file**. 1 hàm ở file này có thể dùng cho nhiều **Scenario**.

|  |
| --- |
| package step\_definitions.login\_function; import constant.base\_constant; import io.cucumber.java.After; import io.cucumber.java.en.Given; import io.cucumber.java.en.Then; import io.cucumber.java.en.When; import io.github.bonigarcia.wdm.WebDriverManager; import org.openqa.selenium.By; import org.openqa.selenium.chrome.ChromeDriver; import org.selenium.automationtest.common.CommonBase; public class login\_steps extends CommonBase {  Object input\_id = By.*id*("UserName");  Object input\_pass = By.*id*("PassWord");  Object button\_login = By.*xpath*("//button[@type='submit']");  Object label\_user\_name = By.*xpath*("//img[@class='user-image']/following-sibling::span");  Object notification\_missing\_username = By.*xpath*("//div[@class='validation-summary-errors text-danger']//li[1]");  @Given("I navigate to admin page")  public void i\_navigate\_to\_admin\_page() {  WebDriverManager.*chromedriver*().setup();  driver = new ChromeDriver();  driver.manage().window().maximize();  openPage(base\_constant.*ADMIN\_URL*, driver);  waitForPageLoaded(driver);  }  @When("I enter an admin account {string}")  public void i\_enter\_an\_existing\_admin\_account(String string) {  type(input\_id, string);  }  @When("I enter the password {string}")  public void i\_enter\_the\_password(String string) {  type(input\_pass, string);  }  @When("I click on login button")  public void i\_click\_on\_login\_button() {  click(button\_login);  }  @Then("I validate the title of website")  public void i\_validate\_the\_title\_of\_website() {  String title\_text = driver.getTitle();  verifyCompare(title\_text, "Trang chủ");  }  @Then("check validate the name of account")  public void check\_validate\_the\_name\_of\_account() {  String username\_text = getText(label\_user\_name);  verifyCompare(username\_text, "Tran Tiên Manh");  }  @Then("I validate the notification with value {string}")  public void i\_validate\_the\_notification\_with\_value(String string) {  String username\_text = getText(notification\_missing\_username);  verifyCompare(username\_text, string);  }  @Then("I close the browser and driver")  public void i\_close\_the\_browser\_and\_driver() {  driver.close();  driver.quit();  }  @After  public void afterScenario() {  try {  driver.close();  driver.quit();  } catch (Exception e) {  }  } }  package step\_definitions.product\_manage; import constant.base\_constant; import io.cucumber.java.After; import io.cucumber.java.en.And; import io.cucumber.java.en.Then; import io.cucumber.java.en.When; import io.github.bonigarcia.wdm.WebDriverManager; import org.openqa.selenium.By; import org.openqa.selenium.chrome.ChromeDriver; import org.selenium.automationtest.common.CommonBase; public class product\_steps extends CommonBase {  Object input\_id = By.*id*("UserName");  Object input\_pass = By.*id*("PassWord");  Object button\_login = By.*xpath*("//button[@type='submit']");  Object button\_create = By.*xpath*("//button[text()='Tạo mới']");  Object input\_name = By.*id*("name");  Object input\_img = By.*id*("image");  Object select\_publishing = By.*id*("pub");  Object select\_author = By.*id*("author");  Object input\_desc = By.*id*("desc");  Object button\_add = By.*id*("btnAdd");  Object label\_name\_created = By.*xpath*("//tbody/tr[1]/td[1]");  Object button\_edit = By.*xpath*("//table[@class='table table-bordered']/tbody/tr[1]/td[last()]/a[1]");  Object button\_update = By.*id*("btnUpdate");  Object button\_edit\_price = By.*xpath*("//table[@class='table table-bordered']/tbody/tr[1]/td[8]/a");  Object label\_price = By.*xpath*("//table[@class='table table-bordered']/tbody/tr[1]/td[8]");  Object input\_price = By.*id*("priceout");  Object input\_discount = By.*id*("promotionforPrice");  Object button\_update\_price = By.*id*("btnUpdatePrice");  Object button\_cancel\_update\_price = By.*xpath*("//button[@id='btnUpdatePrice']//following-sibling::button");  Object button\_delete\_first\_book = By.*xpath*("//table[@class='table table-bordered']/tbody/tr[1]/td[10]/a[@class='btn btn-danger']");  @When("I login to the admin page")  public void i\_login\_to\_the\_admin\_page() {  WebDriverManager.*chromedriver*().setup();  driver = new ChromeDriver();  driver.manage().window().maximize();  openPage(base\_constant.*ADMIN\_URL*, driver);  type(input\_id, "manh");  type(input\_pass, "manh123");  click(button\_login);  waitForPageLoaded(driver);  }  @When("Navigate to the product management page")  public void navigate\_to\_the\_product\_management\_page() {  openPage(base\_constant.*PRODUCT\_MANAGE\_URL*, driver);  waitForPageLoaded(driver);  }  @When("Click on the create button")  public void click\_on\_the\_create\_button() {  click(button\_create);  }  @When("Enter book name with value {string}")  public void enter\_book\_name\_with\_value(String string) {  type(input\_name, string);  }  @When("Enter img link with value {string}")  public void enter\_img\_link\_with\_value(String string) {  type(input\_img, string);  }  @When("Select the author")  public void select\_the\_author() {  selectOptionByValue(select\_author, "1");  }  @When("Select the publishing company")  public void select\_the\_publishing\_company() {  selectOptionByValue(select\_publishing, "1");   }  @Then("Accept the alert")  public void accept\_the\_alert() {  driver.switchTo().alert().accept();  pause(2000);  }  @Then("I click on save button")  public void i\_click\_on\_save\_button() {  click(button\_add);  pause(2000);  }  @Then("I validate color of name textbox")  public void i\_validate\_color\_of\_name\_textbox() {  String attribute = getAttributeFromJavaScript("//\*[@id='name']", "style");  verifyContains("red", attribute);  }  @When("Enter description with value {string}")  public void enter\_description\_with\_value(String string) {  type(input\_desc, string);  }  @Then("I validate the name of product with value {string}")  public void i\_validate\_the\_name\_of\_product\_with\_value(String string) {  String product\_text = getText(label\_name\_created);  verifyCompare(product\_text, string);  }  @After  public void afterScenario() {  try {  driver.close();  driver.quit();  } catch (Exception e) {  }  }  @And("I validate color of img textbox")  public void iValidateColorOfImgTextbox() {  String attribute = getAttributeFromJavaScript("//\*[@id='image']", "style");  verifyContains("red", attribute);  }  @And("I validate color of author combobox")  public void iValidateColorOfAuthorCombobox() {  String attribute = getAttributeFromJavaScript("//\*[@id='author']", "style");  verifyContains("red", attribute);  }  @And("I validate color of publish combobox")  public void iValidateColorOfPublishCombobox() {  String attribute = getAttributeFromJavaScript("//\*[@id='pub']", "style");  verifyContains("red", attribute);  }  @And("I validate color of description combobox with {string}")  public void iValidateColorOfDescriptionComboboxWith(String arg0) {  String attribute = getAttributeFromJavaScript("//\*[@id='desc']", "style");  verifyContains(arg0, attribute);  }  @And("I validate color of name textbox with {string}")  public void iValidateColorOfNameTextboxWith(String arg0) {  String attribute = getAttributeFromJavaScript("//\*[@id='name']", "style");*// border-coler: red* verifyContains(arg0, attribute);  }  @And("Clean text in descripton textbox")  public void cleanTextInDescriptonTextbox() {  type(input\_desc, "");  }  @And("Click on the Edit button")  public void clickOnTheEditButton() {  click(button\_edit);  pause(2000);  }  @Then("I click on update button")  public void iClickOnUpdateButton() {  click(button\_update);  pause(2000);  }  @And("Click on the edit price button")  public void clickOnTheEditPriceButton() {  click(button\_edit\_price);  }  @And("Enter the price with value {string}")  public void enterThePriceWithValue(String arg0) {  type(input\_price, arg0);  }  @And("Enter the discount with {string}")  public void enterTheDiscountWith(String arg0) {  type(input\_discount, arg0);  }  @Then("I click on update price button")  public void iClickOnUpdatePriceButton() {  click(button\_update\_price);  pause(2000);  }  @And("I validate alert notification with {string}")  public void iValidateAlertNotification(String arg0) {  String a = getAlertText();  verifyCompare(arg0, a);  }  @And("I validate color of discount with {string}")  public void iValidateColorOfPriceWith(String arg0) {  String attribute = getAttributeFromJavaScript("//input[@id='promotionforPrice']", "style");  verifyContains(arg0, attribute);  }  @And("I accept alert")  public void iAcceptAlert() {  acceptAlert();  pause(3000);  }  @And("I click on cancel update price button")  public void iClickOnCancelUpdatePriceButton() {  click(button\_cancel\_update\_price);  }  @And("I validate the price with value {string}")  public void iValidateThePriceWithValue(String arg0) {  String text = getText(label\_price);  text = text.replace(".", "");  text = text.replace("₫", "");  verifyCompare(arg0, text);  }  @And("I refresh this website")  public void iRefreshThisWebsite() {  try {  driver.navigate().refresh();   } catch (Exception e) {  }  }  @And("Click on the delete button of first book")  public void clickOnTheDeleteButtonOfFirstBook() {  click(button\_delete\_first\_book);  }  @And("I validate the name of product different value {string}")  public void iValidateTheNameOfProductWithoutValue(String arg0) {  String text = getText(label\_name\_created);  verifyNotEqual(text, arg0);  }  @And("I cancel alert")  public void iCancelAlert() {  cancelAlert();  } }  package step\_definitions.cart\_manage; import constant.base\_constant; import io.cucumber.java.After; import io.cucumber.java.en.And; import io.cucumber.java.en.Given; import io.cucumber.java.en.Then; import io.cucumber.java.en.When; import io.github.bonigarcia.wdm.WebDriverManager; import org.openqa.selenium.By; import org.openqa.selenium.WebElement; import org.openqa.selenium.chrome.ChromeDriver; import org.selenium.automationtest.common.CommonBase; public class cart\_steps extends CommonBase {  Object input\_id = By.*id*("UserName");  Object input\_pass = By.*id*("PassWord");  Object button\_login = By.*xpath*("//button[@type='submit']");  Object input\_search = By.*xpath*("//input[@name='keyword']");  Object button\_search = By.*xpath*("//button[@type='submit' and @class='pull-right btn-search']");  Object button\_book\_1 = By.*xpath*("//div[@class='product-featured clearfix']//ul[@class='product-list']/li[1]");  Object button\_icon\_cart = By.*xpath*("//i[@class='fa fa-cart-plus fa-2x']");  Object button\_add\_to\_cart = By.*xpath*("//div[@class='button-group']/a[@title='Add to Cart']");  Object label\_notification\_chuacosanpham = By.*xpath*("//span[@class='alert alert-danger']");  Object number\_of\_items\_in\_cart = By.*xpath*("//div[@class='heading-counter warning']/span[2]/b");  String coutn\_of\_the\_first\_book = "//div[@class='order-detail-content']//tbody/tr[1]//input";  String coutn\_of\_the\_second\_book = "//div[@class='order-detail-content']//tbody/tr[2]//input";  Object button\_delete\_from\_list = By.*xpath*("//a[text()='Delete item']");  Object button\_first\_book\_name = By.*xpath*("//div[@class='order-detail-content']//tbody/tr[1]//td[1]");  Object button\_second\_book\_name = By.*xpath*("//div[@class='order-detail-content']//tbody/tr[2]//td[1]");  Object button\_delete\_all = By.*id*("btn\_deleteAll");  Object button\_order = By.*id*("btn\_payment");  Object input\_surname = By.*id*("firstname");  Object input\_name = By.*id*("lastname");  Object input\_email = By.*id*("email");  Object input\_address = By.*id*("address");  Object input\_phone = By.*id*("phone");  Object button\_continue = By.*id*("button-payment-method");  @Given("I login to the user page")  public void i\_login\_to\_the\_user\_page() {  WebDriverManager.*chromedriver*().setup();  driver = new ChromeDriver();  driver.manage().window().maximize();  openPage(base\_constant.*ADMIN\_URL*, driver);  type(input\_id, "huongkhuong");  type(input\_pass, "huong123");  click(button\_login);  waitForPageLoaded(driver);  }  @Given("I navigate to the user page")  public void iNavigateToTheUserPage() {  WebDriverManager.*chromedriver*().setup();  driver = new ChromeDriver();  driver.manage().window().maximize();  openPage(base\_constant.*USER\_URL*, driver);  }  @Given("Click on the cart button")  public void click\_on\_the\_cart\_button() {  click(button\_icon\_cart);  waitForPageLoaded(driver);  }  @Given("Enter search input with value {string} and search")  public void enter\_search\_input\_with\_value\_and\_search(String string) {  type(input\_search, string);  click(button\_search);  waitForPageLoaded(driver);  }  @Given("I click on the book")  public void i\_click\_on\_the\_book() {  click(button\_book\_1);  waitForPageLoaded(driver);  }  @Given("I click on add to card")  public void i\_click\_on\_add\_to\_card() {  WebElement element = getElementPresent(button\_add\_to\_cart);  if (*isVisibleInViewport*(element, driver)) {  click(button\_add\_to\_cart);  } else {  scrollToElement(button\_add\_to\_cart);  click(button\_add\_to\_cart);  waitForPageLoaded(driver);  }  }  @When("I close browser")  public void i\_close\_browser() {  try {  driver.close();  driver.quit();  } catch (Exception e) {  }  }  @Then("validate count of first book with {string}")  public void validateCoutnOfTheItemIs(String arg0) {  String text = getAttributeFromJavaScript(coutn\_of\_the\_first\_book, "value");  verifyCompare(text, arg0);  }  @Then("validate name of first item is {string}")  public void validateNameOfFirstItemIs(String arg0) {  String text = getText(button\_first\_book\_name);  verifyCompare(text, arg0);  }  @And("validate notification with value {string}")  public void validateNotificationWithValue(String arg0) {  String text = getText(label\_notification\_chuacosanpham);  verifyCompare(text, arg0);  }  @After  public void afterScenario() {  try {  driver.close();  driver.quit();  } catch (Exception e) {  }  }  @When("validate text next to cart icon with value {string}")  public void validateTextNextToCartIconWithValue(String arg0) {  String text = getText(button\_icon\_cart);  verifyCompare(text, arg0);  }  @And("validate the name of second item is {string}")  public void validateTheNameOfSecondItemIs(String arg0) {  String text = getText(button\_second\_book\_name);  verifyCompare(text, arg0);  }  @And("validate number of items in cart with {string}")  public void validateNumberOfItemsInCartWith(String arg0) {  String text = getText(number\_of\_items\_in\_cart);  verifyCompare(text, arg0);  }  @When("I click on Order button")  public void iClickOnOrderButton() {  click(button\_order);  }  @And("Enter the surname with {string}")  public void enterTheSurnameWith(String arg0) {  type(input\_surname, arg0);  }  @And("Enter the name with {string}")  public void enterTheNameWith(String arg0) {  type(input\_name, arg0);  }  @And("Enter the email with {string}")  public void enterTheEmailWith(String arg0) {  type(input\_email, arg0);  }  @And("Enter the address with {string}")  public void enterTheAddressWith(String arg0) {  type(input\_address, arg0);  }  @And("Enter the phone number with {string}")  public void enterThePhoneNumberWith(String arg0) {  type(input\_phone, arg0);  }  @Then("I click on button continue")  public void iClickOnButtonContinue() {  click(button\_continue);  }  @And("i validate color of textbox surname with {string}")  public void iValidateColorOfTextboxSurnameWith(String arg0) {  String attribute = getAttributeFromJavaScript("//input[@id='firstname']", "style");*// border-coler: red* verifyContains(arg0, attribute);  }  @And("i validate color of textbox name with {string}")  public void iValidateColorOfTextboxNameWith(String arg0) {  String attribute = getAttributeFromJavaScript("//input[@id='lastname']", "style");*// border-coler: red* verifyContains(arg0, attribute);  }  @And("i validate color of textbox address with {string}")  public void iValidateColorOfTextboxAddressWith(String arg0) {  String attribute = getAttributeFromJavaScript("//input[@id='address']", "style");*// border-coler: red* verifyContains(arg0, attribute);  }  @And("i validate color of textbox email with {string}")  public void iValidateColorOfTextboxEmailWith(String arg0) {  String attribute = getAttributeFromJavaScript("//input[@id='email']", "style");*// border-coler: red* verifyContains(arg0, attribute);  }  @And("i validate color of textbox phone number with {string}")  public void iValidateColorOfTextboxPhoneNumberWith(String arg0) {  String attribute = getAttributeFromJavaScript("//input[@id='phone']", "style");*// border-coler: red* verifyContains(arg0, attribute);  }  @Then("I click on Delete All")  public void iClickOnDeleteAll() {  click(button\_delete\_all);  } } |

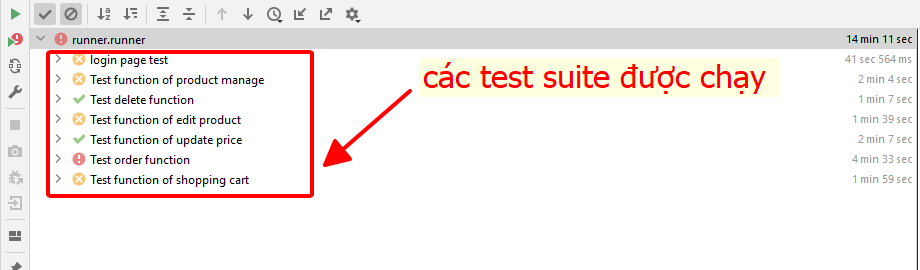
* + 1. Xây dựng lớp runner

**Ý nghĩa:** lớp này giúp ta cài đặt, nhóm, điều khiển được các testcase muốn chạy, tùy chỉnh các loại báo cáo.

|  |
| --- |
| package runner; import org.junit.runner.RunWith;  import io.cucumber.junit.CucumberOptions; import io.cucumber.junit.Cucumber;  @RunWith(Cucumber.class)  @CucumberOptions(features="src/test/resources/features",  glue= {"step\_definitions"},  tags = "@ngoaile",  plugin={"pretty","html:target/HTML/report.html"},  publish = true ) public class runner { } |

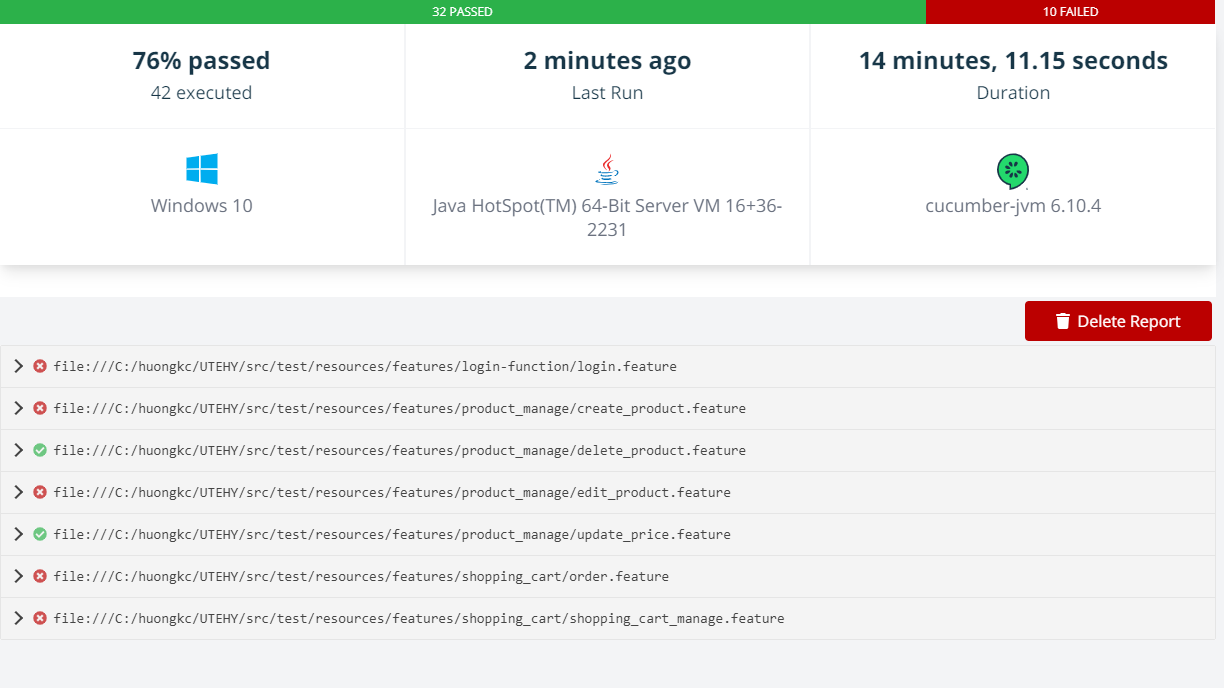
* 1. Thực thi và báo cáo kiểm thử
     1. Thực thi kiểm thử

Để thực thi các ca kiểm thử hướng hành vi, người dùng chỉ cần điều chỉnh lại file runner.java, để chọn lọc các ca kiểm thử mong muốn chạy, loại báo cáo report ra.



* 1. Các ca kiểm thử được thực thi
     1. Báo cáo kiểm thử

Sau khi thực thi các ca kiểm thử, framework sẽ tự report ra một dạng báo cáo theo mình tùy chỉnh ở thư mục: *target/HTML/report.html*



* 1. Báo cáo tổng khi chạy xong các ca kiểm thử

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

* 1. Kết quả đạt được

Thông qua việc tìm hiểu lý thuyết về kiểm thử và kiểm thử tự động, cũng như áp dụng lý thuyết vào việc xây dựng framework kiểm thử tự động, đồ án đã đạt được những kết quả như sau:

* Đã hoàn thành được mục tiêu của đồ án đó là: Đã xây dựng thành công framework kiểm thử tự động hướng hành vi và áp dụng vào kiểm thử cho website bán sách Fairy Tail.
* Tổng hợp lý thuyết về kiểm thử và kiểm thử tự động, lợi ích và thách thức của kiểm thử tự động các phương pháp luận tiếp cận thử động, các bước cần phải làm khi muốn áp dụng tự động hóa kiểm thử.
* Đồ án đã áp dụng kiến thức để xây dựng kiểm thử tự động cho các chức năng quan trọng của phần mềm, góp phần giảm chi phí, nguồn lực và thời gian thực hiện kiểm thử...

Thông qua đồ án này, tôi nhận thấy rằng kiểm thử tự động là một giải pháp tốt trong việc nâng cao năng suất chất lượng của kiểm thử. Đồng thời cũng giúp nâng cao kỹ năng và kiến thức cho kiểm thử viên. Góp phần giảm thời gian phát triển sản phẩm mà vẫn đảm bảo được chất lượng của phần mềm. Kiểm thử tự động là một thị trường rất tiềm năng không chỉ cho các nhà đầu tư mà còn là một lĩnh vực đang rất khát nhân lực ở Việt Nam, từ đó mang đến nhiều cơ hội cho các kiểm thử viên và các bạn trẻ đang ngồi trên ghế nhà trường.

Tuy nhiên, để phát huy tốt nhất khả năng của kiểm thử phần mềm tự động, cần phải lựa chọn các chức năng tự động hóa một cách cẩn thận, hiệu quả. Không phải chức năng nào cũng có thể tự động hóa và mang lại hiệu quả. Việc lựa chọn đúng chức năng không những làm giảm thiểu chi phí tự động hóa mà còn nâng cao hiệu quả của tự động kiểm thử.

* 1. Hạn chế của đề tài

Do tập chung tìm hiểu, xây dựng và phát triển dự án theo hướng hành vi là chủ yếu các hạn chế của hướng hành vi vẫn chưa thể khắc phục, hiện tại framework chỉ áp dụng cho các ứng dụng web nên phạm vi áp dụng của framework còn hạn hẹp. Báo cáo của framework chưa phân tích được chi tiết kết quả của các ca kiểm thử.

* 1. Hướng phát triển của đề tài

Kiểm thử tự động sẽ chỉ đem lại lợi ích qua quá trình sử dụng lâu dài. Do vậy, khi có ý định sử dụng kiểm thử tự động, thì cần phải khuyến khích việc sử dụng kiểm thử tự động, cũng như cơ chế, chính sách để phát triển kiểm thử tự động. Có như vậy, kiểm thử tự động mới thực sự đem lại hiệu quả to lớn cho tổ chức. Trong tương lai, có thể tập trung nghiên cứu theo các hướng sau:

* Thứ nhất, áp dụng tự động hóa cho việc tạo ra các bài kiểm thử, tạo dữ liệu kiểm thử tự động.
* Thứ hai, áp dụng các chức năng khác trong toàn bộ các bài kiểm thử liên quan đến kiểm thử hồi qui bao gồm thay đổi nhiều dữ liệu, thêm mới nhiều dữ liệu, các chức năng khác trong thao tác người dùng của hệ thống.
* Thứ ba, tiếp tục nghiên cứu sửa đổi, mở rộng framework kiểm thử tự động hiện tại để có thể kiểm thử tự động cho các ứng dụng khác như ứng dụng trên windows. Sửa đổi và nâng cấp để báo cáo của framework có thể tiện lợi tối ưu cho người dùng.

TÀI LIỆU THAM KHẢO

x

|  |  |
| --- | --- |
| [1] | https://viblo.asia/p/tim-hieu-ve-cac-cap-do-kiem-thu-test-levels-4P856drAZY3 |
| [2] | https://vi.wikipedia.org/wiki/Kiểm\_thử\_phần\_mềm |
| [3] | https://vi.wikipedia.org/wiki/Kiểm\_thử\_tự\_động |
| [4] | https://viblo.asia/p/tim-hieu-ve-kiem-thu-chuc-nang-functionality-testing-bJzKmLVB59N |

PHỤ LỤC

1. Đường dẫn tới dự án

https://github.com/sotatek-huongkhuong/graduation\_project