ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA HỆ THỐNG THÔNG TIN





BÁO CÁO ĐỒ ÁN CUỐI KÌ

PHÁT TRIỂN ỨNG DỤNG WEB

ĐỀ TÀI XÂY DỤNG WEBSITE BÁN QUẦN ÁO

**Giảng viên hướng dẫn**: Trình Trọng Tín

**Lớp**: IS207.I11

**Thành viên nhóm**:

Nguyễn Hoàng Nam MSSV 15520515

Nguyễn Xuân Sang MSSV 15520720

Trần Thị Xoan MSSV 15521038

Tp Hồ Chí Minh ngày 25 tháng 12 năm 2017

LỜI CÁM ƠN

Đầu tiên, chúng em xin gửi lời cảm ơn chân thành đến toàn thể quý thầy cô Trường Đại học Công nghệ Thông tin- Đại học Quốc giá TP.HCM và quý thầy cô khoa Hệ thống Thông tin đã tận tình truyền đạt những kiến thức cơ bản làm nền tảng cho chúng em thực hiện đề tài này.

Đặc biệt nhóm em xin gửi lời cảm ơn tới thầy Trình Trọng Tín. Qúy thầy cô đã trực tiếp hướng dẫn tận tình, sửa chữa và góp ý giúp nhóm hoàn thành tốt đồ án của mình.

Trong thời gian một học kỳ thực hiện đề tài, nhóm đã vận dụng những kiến thức tiếp thu được kết hợp với việc học hỏi và nghiên cứu những kiến thức mới để hoàn thiện đồ án một cách tốt nhất. Chính vì vậy nhóm em rất mong nhận được những sự góp ý chân thành từ phía thầy cô nhằm cải thiện đồ án cũng như kiến thức của nhóm trong tương lai.

Xin chân thành cảm ơn quý Thầy Cô!

NHẬN XÉT CỦA GIẢNG VIÊN

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

………………………………………………………………………………………….

MỤC LỤC

[CHƯƠNG 1 TỔNG QUAN ĐỀ TÀI 5](#_Toc501957121)

[1.1 Phát biểu bài toán 5](#_Toc501957122)

[1.1.1 Đặt vấn đề 5](#_Toc501957123)

[1.1.2 Mục tiêu 6](#_Toc501957124)

[1.1.3 Phạm vi 6](#_Toc501957125)

[1.1.4 Công cụ sử dụng 6](#_Toc501957126)

[CHƯƠNG 2 THIẾT KẾ HỆ THỐNG 7](#_Toc501957127)

[2.1 Mô hình Use-case 7](#_Toc501957128)

[2.1.1 Sơ đồ Use-case 7](#_Toc501957129)

[2.2 Danh sách các Actor 13](#_Toc501957130)

[2.3 Danh sách các Use-case 13](#_Toc501957131)

[2.4 Đặc tả Use-case và sơ đồ hoạt động (Activity Diagram) 15](#_Toc501957132)

[2.4.1 Quản lý thông tin tài khoản 16](#_Toc501957133)

[2.4.2 Đăng ký tài khoản 17](#_Toc501957134)

[2.4.3 Quên mật khẩu 19](#_Toc501957135)

[2.4.4 Tra cứu sản phẩm 20](#_Toc501957136)

[2.4.5 Thêm vào giỏ hàng 22](#_Toc501957137)

[2.4.6 Bình luận, đánh giá 23](#_Toc501957138)

[2.4.7 Quản lý giỏ hàng 25](#_Toc501957139)

[2.4.8 Đặt mua sản phẩm 27](#_Toc501957140)

[2.4.9 Đăng nhập 29](#_Toc501957141)

[2.4.10 Đăng xuất 31](#_Toc501957142)

[CHƯƠNG 3 THIẾT KẾ GIAO DIỆN 33](#_Toc501957143)

[3.1 Danh sách các trang chính trong chương trình 33](#_Toc501957144)

[3.2 Giao diện các trang 34](#_Toc501957145)

[3.2.1 Trang chủ 34](#_Toc501957146)

[3.2.2 Trang xem thông tin chi tiết sản phẩm 35](#_Toc501957147)

[3.2.3 Trang đăng nhập 36](#_Toc501957148)

[3.2.4 Trang giỏ hàng 36](#_Toc501957149)

[3.2.5 Trang thông tin tài khoản đăng nhập 37](#_Toc501957150)

[3.2.6 Trang địa chỉ giao hàng 37](#_Toc501957151)

[3.2.7 Trang báo cáo 38](#_Toc501957152)

[3.2.8 Trang quản lý sản phẩm 39](#_Toc501957153)

[3.2.9 Trảng quản lý nhóm sản phẩm 39](#_Toc501957154)

[3.2.10 Trang quản lý màu sắc 40](#_Toc501957155)

[3.2.11 Trang quản lý kích thước 40](#_Toc501957156)

[3.2.12 Trang quản lý khuyến mãi 41](#_Toc501957157)

# CHƯƠNG 1 TỔNG QUAN ĐỀ TÀI

## 1.1 Phát biểu bài toán

### 1.1.1 Đặt vấn đề

Hiện nay, với sự phát triển nhanh chóng của Internet và thương mại điện tử, việc bán hàng qua mạng đã trở thành một phương pháp kinh doanh tất yếu cần phải có trong lĩnh vực buôn bán. Khách hàng nếu muốn mua hàng phải lái xe một quãng đường rất dài và rất khó để chọn được mặt hàng mình ưng ý, và có khi phải đi nhiều cửa hàng để mua được món hàng mình thích. Về lâu về dài, sự bất tiện này sẽ làm ảnh hưởng đến việc kinh doanh của cửa hàng cũng như tụt hậu hơn với những cửa hàng khác khi họ áp dụng việc bán hàng qua mạng. Vì vậy việc tạo ra một hệ thống tương tác giữa khách hàng và cửa hàng đã trở thành một đề tài tất yếu mà mỗi người kinh doanh khi bắt đầu phải đặt ra đầu tiên.

Website bán quần áo đa cấp đã được ra đời nhằm giải quyết tình trạng trên. Một giải pháp mang tính thực tiễn nhằm giải quyết nhu cầu của người bán (cửa hàng quần áo) và người mua, đồng thời tạo nên kênh chính thức để quảng bá thương hiệu, hình ảnh cũng như sản phẩm của cửa hàng đến với người dùng.

### 1.1.2 Mục tiêu

Xây dựng website bán quần áo nasaxo.com đáp ứng các yêu cầu sau:

* Quản lý thông tin tài khoản người dùng.
* Đăng ký tài khoản cho người dùng và có chức năng quên mật khẩu.
* Tra cứu sản phẩm cho người dùng, ngoài ra còn có thêm chức năng bình luận đánh giá.
* Quản lý giỏ hàng cho người dùng khi người dùng cần mua sản phẩm.
* Cung cấp giao diện quản lý cho chủ của hàng.
* Quản lý các loại sản phẩm và các sản phẩm nhất định.
* Quản lý danh sách khách hàng.
* Quản lý đơn đặt hàng.
* Quản lý khuyến mãi.
* Thống kê doanh thu cho cửa hàng.
* Quản lý về các nội dung: kích cỡ, màu sắc.
* Quản lý địa chỉ của người dùng.

### 1.1.3 Phạm vi

Do được chuẩn bị khá kĩ lưỡng từ khâu khảo sát, phân tích yêu cầu cũng như đánh giá dự án. Phạm vi dự án sẽ bao gồm tất cả các hạng mục trên

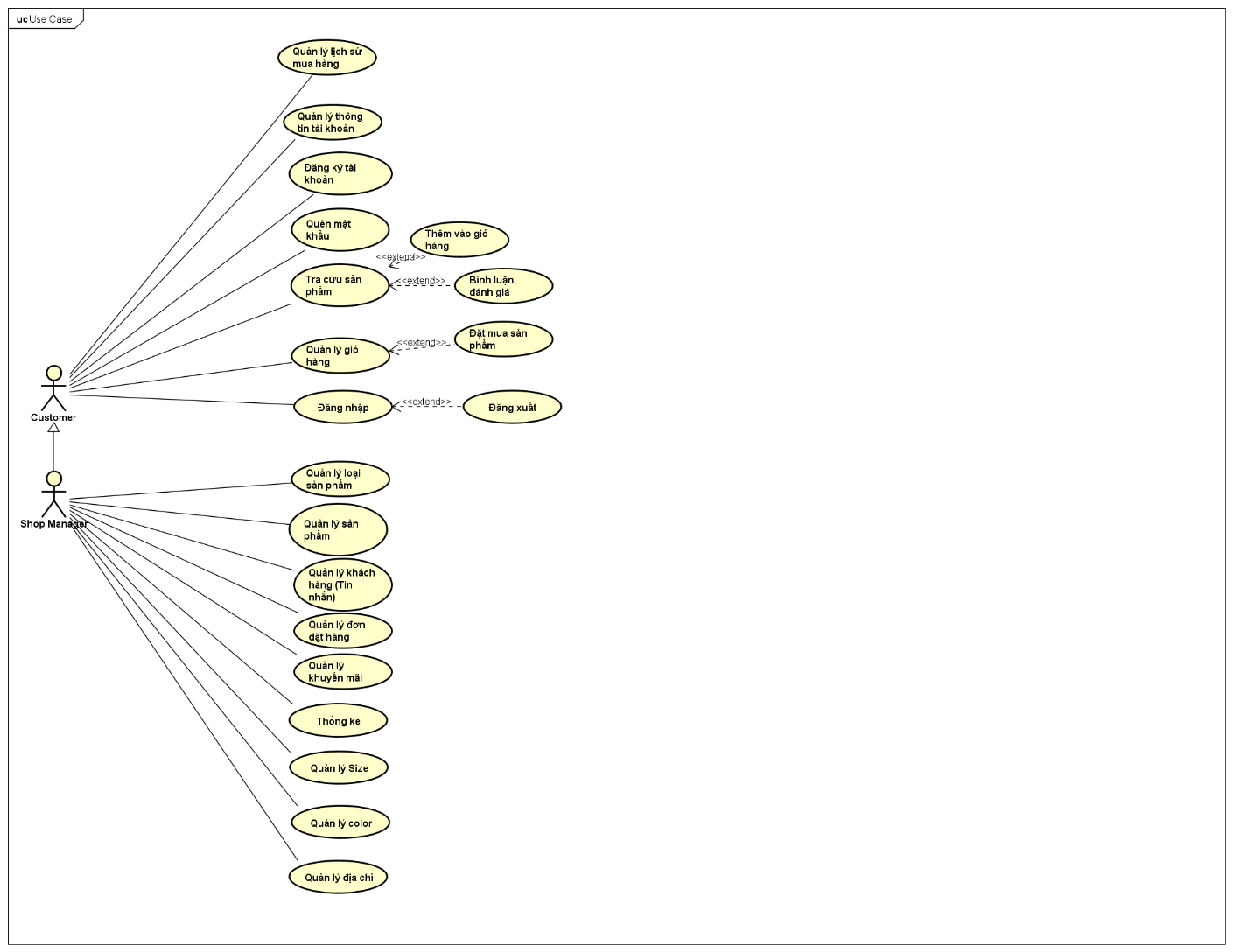
### 1.1.4 Công cụ sử dụng

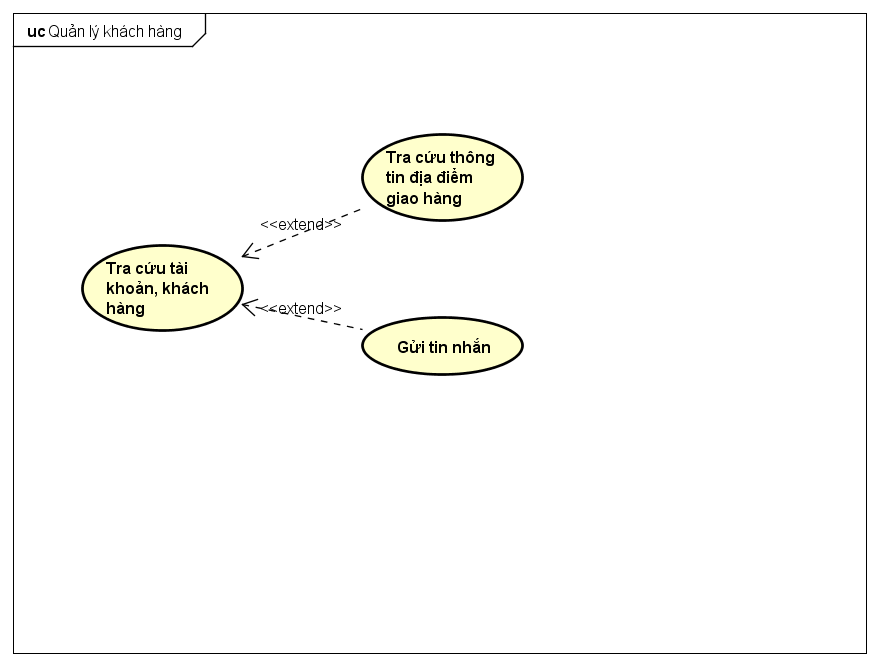
Trong quá trình thực hiện đồ án, nhóm đã sử dụng một số phần mềm phục vụ cho việc tìm hiểu và xây dựng đề tài bao gồm:

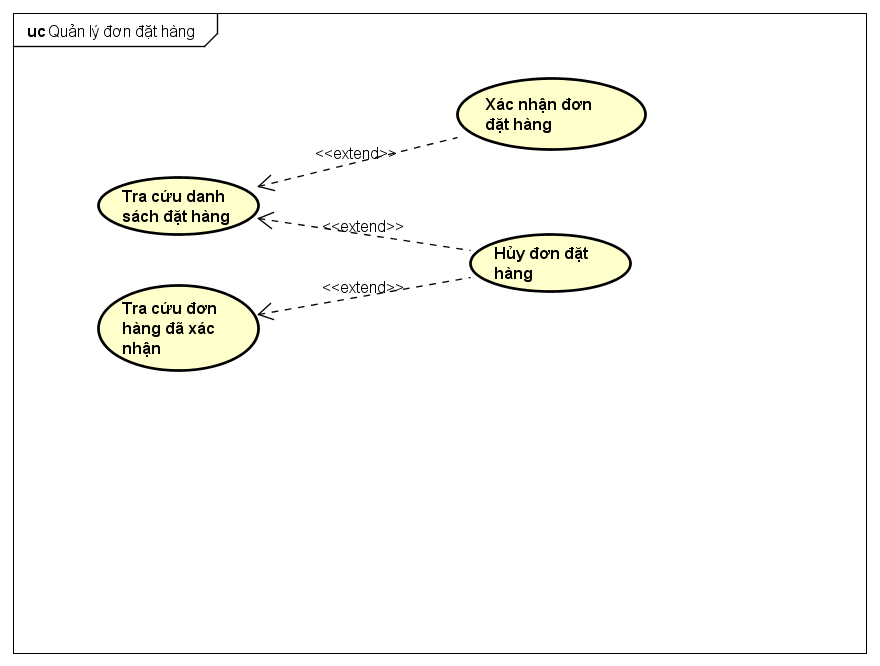
* Hệ quản trị cơ sở dữ liệu Mysql
* Adobe Photoshop CS6 phục vụ thiết kết kế giao diện
* Ngôn ngữ back-end PHP
* Trình soạn thảo Subline Text 3

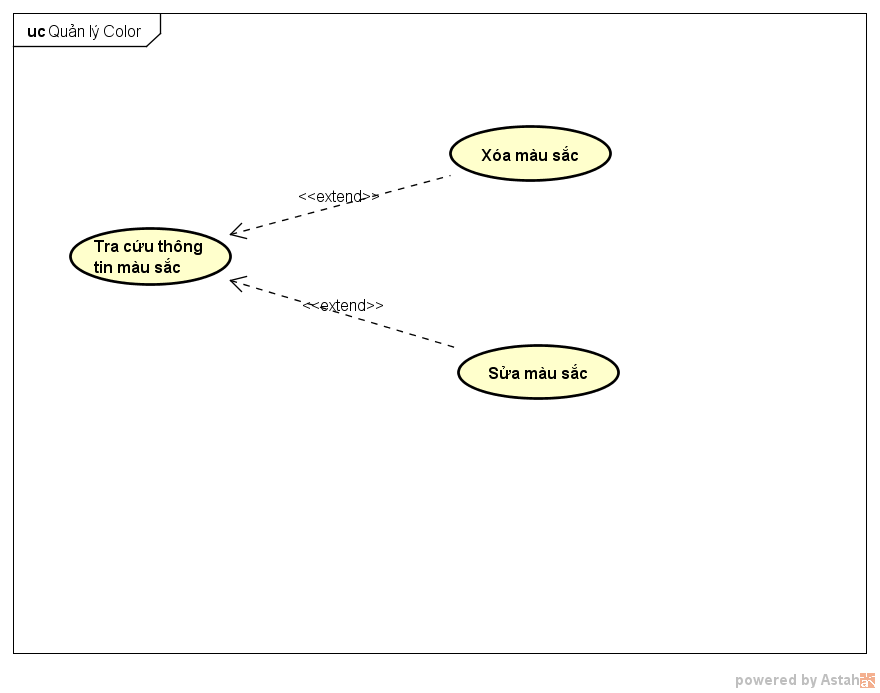
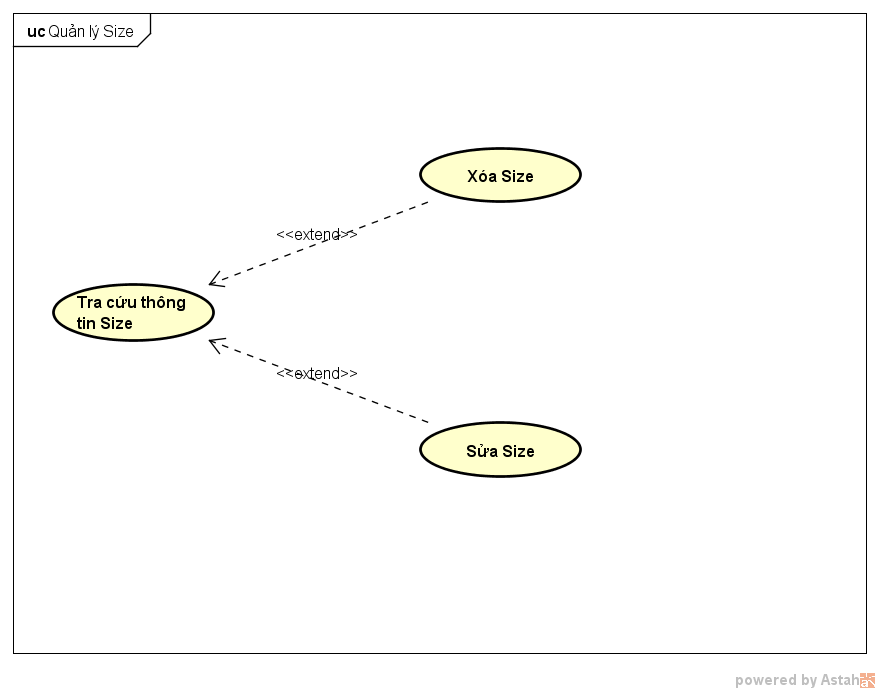
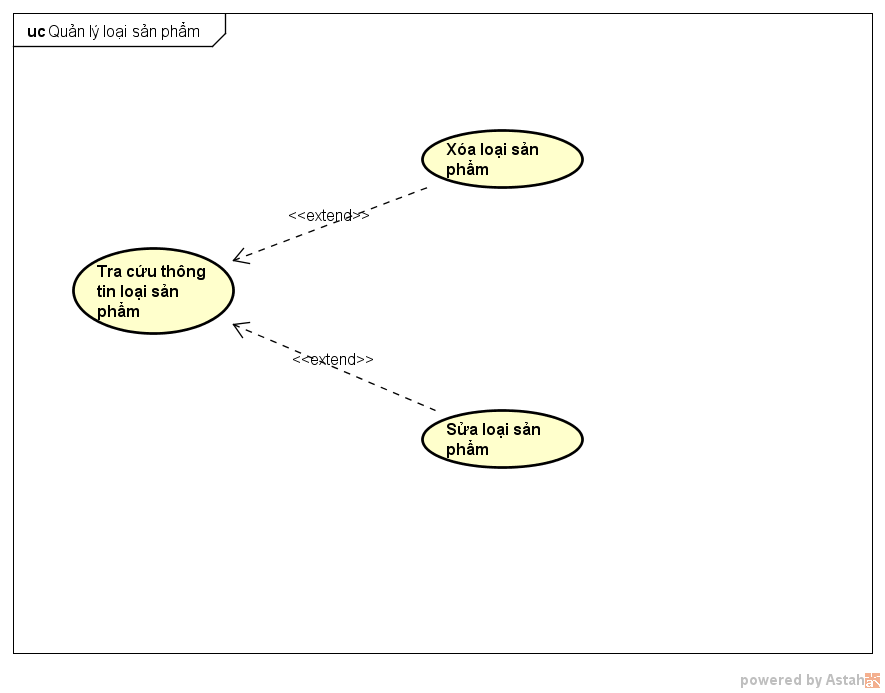
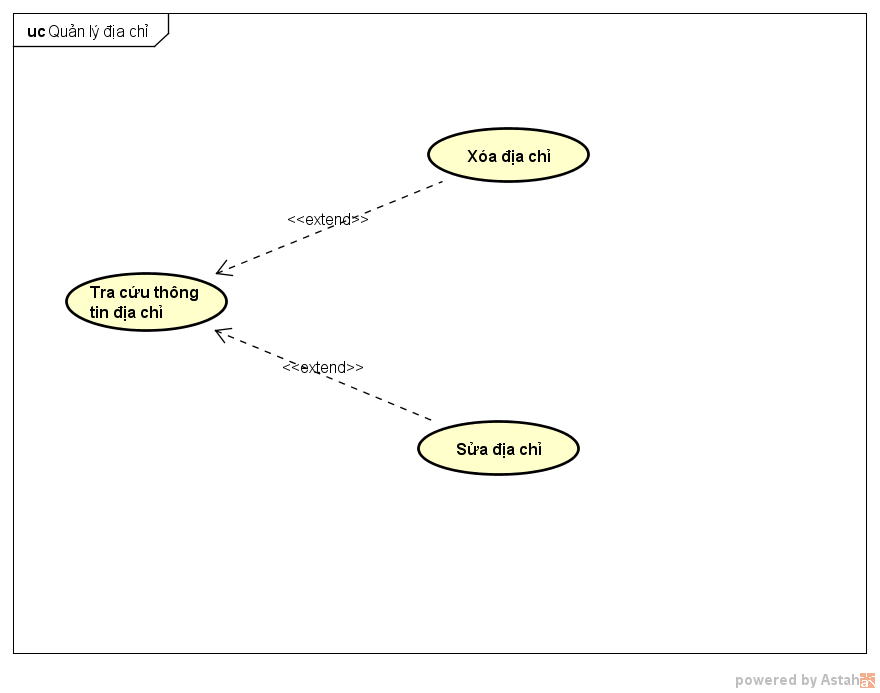
# CHƯƠNG 2 THIẾT KẾ HỆ THỐNG

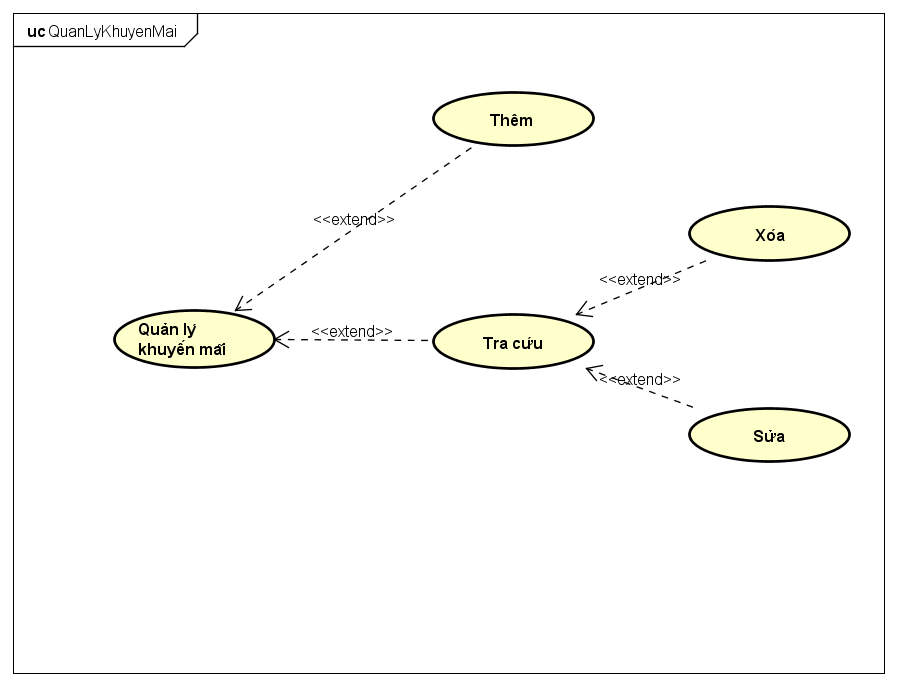
## 2.1 Mô hình Use-case

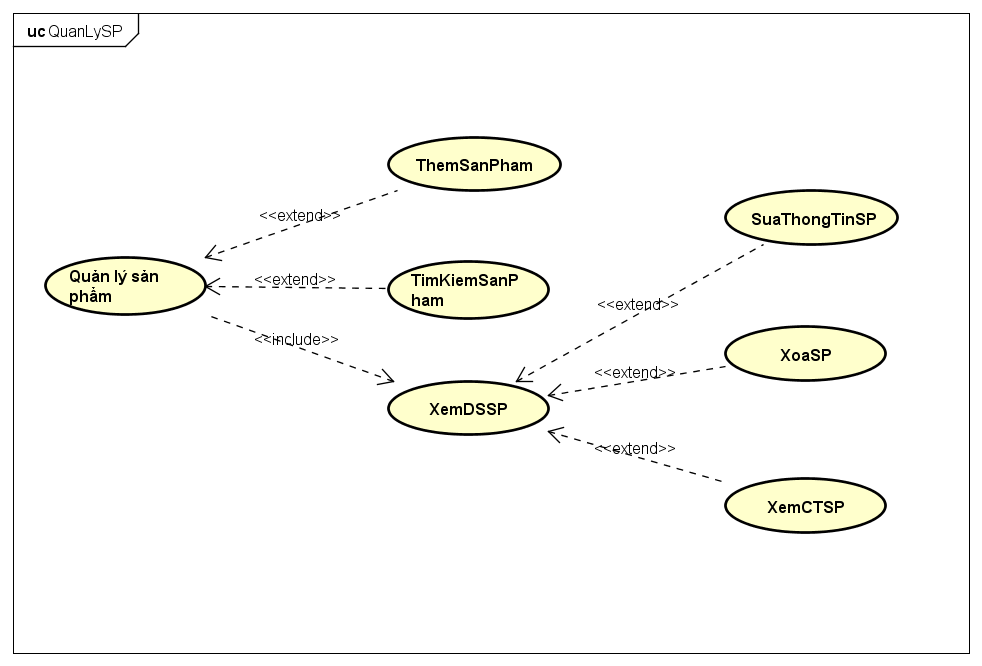


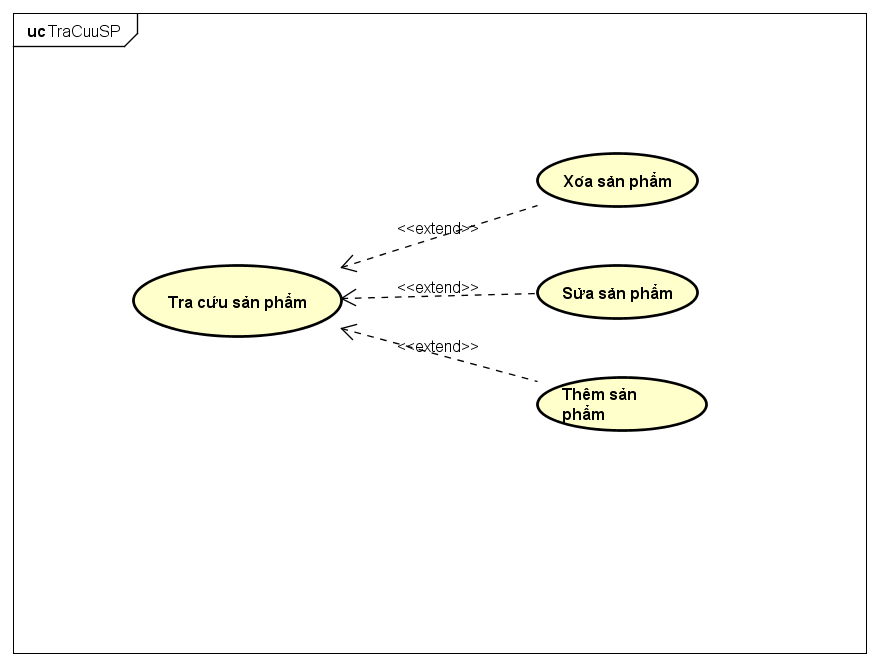












## 2.2 Danh sách các Actor

|  |  |  |
| --- | --- | --- |
| STT | Tên Actor | Ý nghĩa/Ghi chú |
| 1 | Người quản trị | Người quản trị hệ thống |
| 2 | Khách hàng | Khách hàng của của hàng bán quần áo |

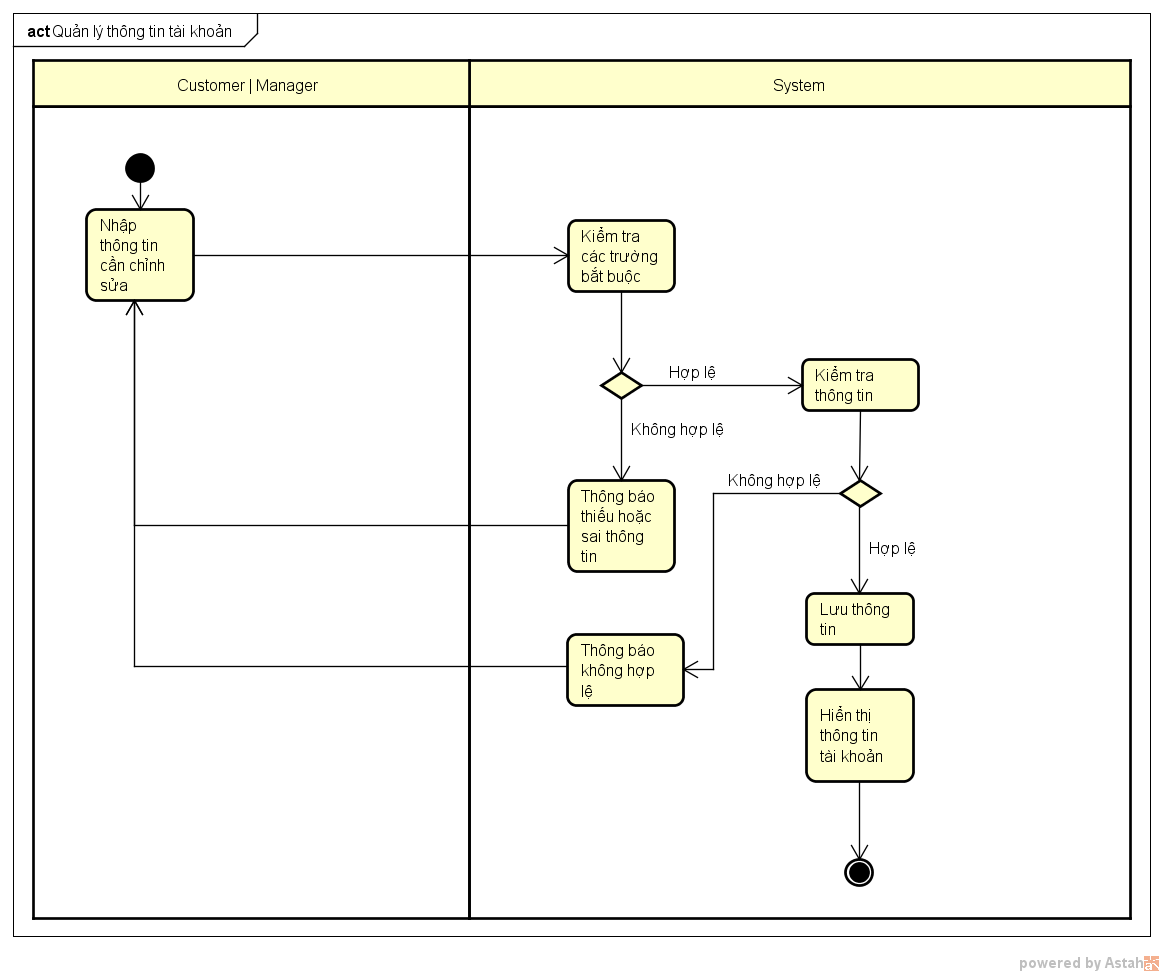
## 2.3 Danh sách các Use-case

|  |  |  |
| --- | --- | --- |
| STT | Tên Use-case | Ý nghĩa/Ghi chú |
| 1 | Quản lý thông tin tài khoản | Xem và chỉnh sửa thông tin tài khoản |
| 2 | Đăng ký tài khoản | Đăng ký tài khoản mới |
| 3 | Quên mật khẩu | Nhận lại mật khẩu khi quên |
| 4 | Tra cứu sản phẩm | Hiển thị danh sách sản phâm và xem các thông tin chi tiết một sản phẩm và thông tin liên quan |
| 5 | Thêm vào giỏ hàng | Thêm một sản phẩm vào giỏ hàng của tài khoản |
| 6 | Bình luận, đánh giá | Bình luận về sản phẩm và đánh giá số sao cho sản phẩm |
| 7 | Quản lý giỏ hàng | Xem danh sách giỏ hàng của tài khoản, xóa sản phẩm trong giỏ và chỉnh sửa số lượng |
| 8 | Đặt mua sản phẩm | Tiến hành đặt mua và hình thành hóa đơn, quản lý các địa điểm đã đặt giao hàng của tài khoản |
| 9 | Đăng nhập | Đăng nhập vào hệ thống bán hàng |
| 10 | Đăng xuất | Đăng xuất khỏi hệ thống |
| 11 | Quản lý loại sản phẩm | Thêm, xóa, sửa, tra cứu loại sản phẩm |
| 12 | Quản lý sản phẩm | Thêm, xóa, sửa, tra cứu sản phẩm |
| 13 | Quản lý khách hàng | Tra cứu thông tin khách hàng |
| 14 | Quản lý đơn đặt hàng | Tra cứu đơn đặt hàng và xác nhận giao hàng |
| 15 | Quản lý khuyến mãi | Thêm, xóa, sửa, tra cứu khuyến mãi |
| 16 | Thống kê | Thống kê doanh thu, thống kê mặt hàng bán chạy |
| 17 | Quản lý Size | Thêm, xóa, sửa, tra cứu các loại size |
| 18 | Quản lý color | Thêm, xóa, sửa, tra cứu các loại màu sắc |
| 19 | Quản lý địa chỉ | Thêm, xóa, sửa, tra cứu thành phố, huyện, phường xã |

## Đặc tả Use-case và sơ đồ hoạt động (Activity Diagram)

### 2.4.1 Quản lý thông tin tài khoản

* Tóm tắt: Người sử dụng: Quản lý hệ thống và khách hàng. Quản lý thông tin tài khoản của người dùng.
* Dòng sự kiện



* Các yêu cầu đặc biệt: Người dùng đã đăng nhập thành công.
* Trạng thái hệ thống khi bắt đầu thực hiện Use-case: Người dùng truy cập vào chức năng quản lý thông tin tài khoản
* Trạng thái hệ thống sau khi thực hiên Use-case: Chỉnh sửa thành công: Hệ thống cập nhật thông tin xuống cơ sở dữ liệu, hiển thị các dữ liệu đã chình sửa cho người dùng.
* Không thành công: Hệ thống hiển thị thông báo không thành công.
* Điểm mở rộng: Không có

### 2.4.2 Đăng ký tài khoản

* Tóm tắt: Người sử dụng: quản lý hệ thống và khách hàng. Người dùng đăng ký tài khoản mới.
* Dòng sự kiện:

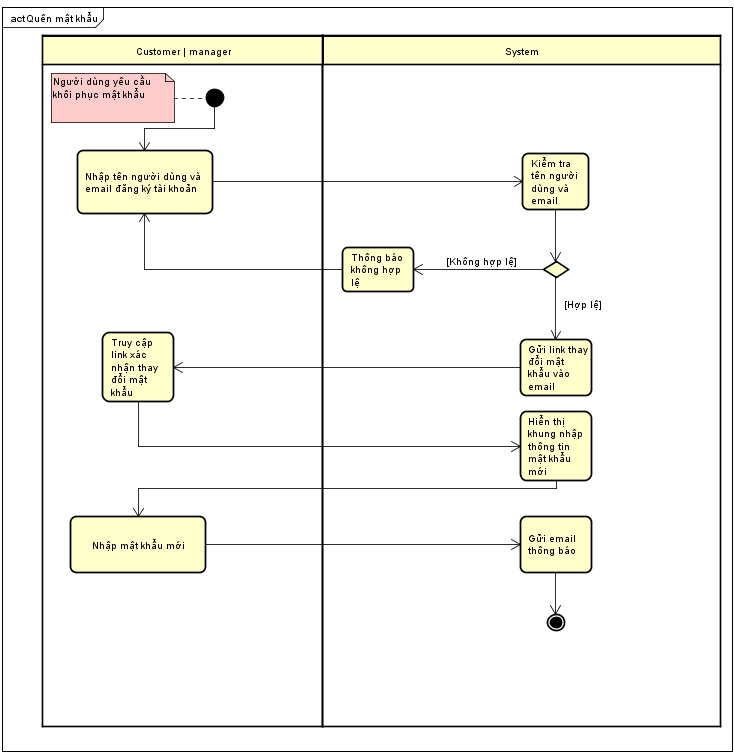


Các yêu cầu đặc biệt: Người dùng phải có tài khoản đăng nhập.

* Trạng thái hệ thống khi bắt đầu thực hiện Use-case: Chưa đăng nhập vào hệ thống.
* Điểm mở rộng: Đăng xuất khỏi hệ thống.

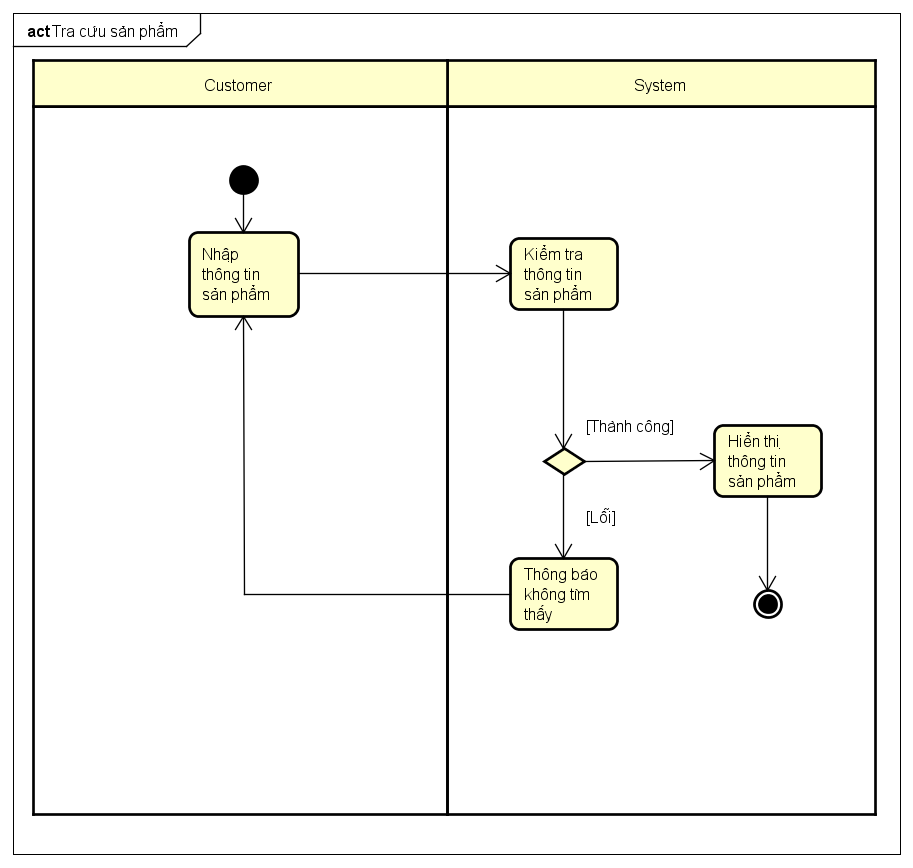
### 2.4.3 Quên mật khẩu

Dòng sự kiện:



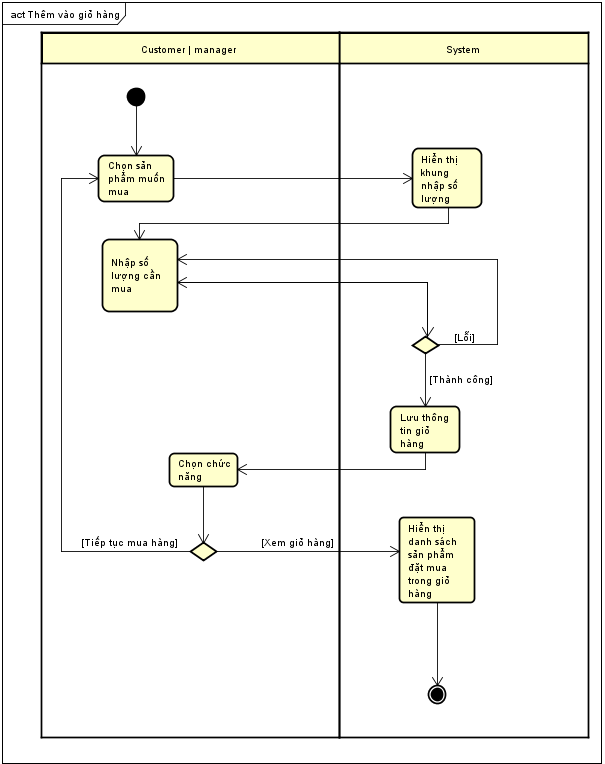
### 2.4.4 Tra cứu sản phẩm

Dòng sự kiện:



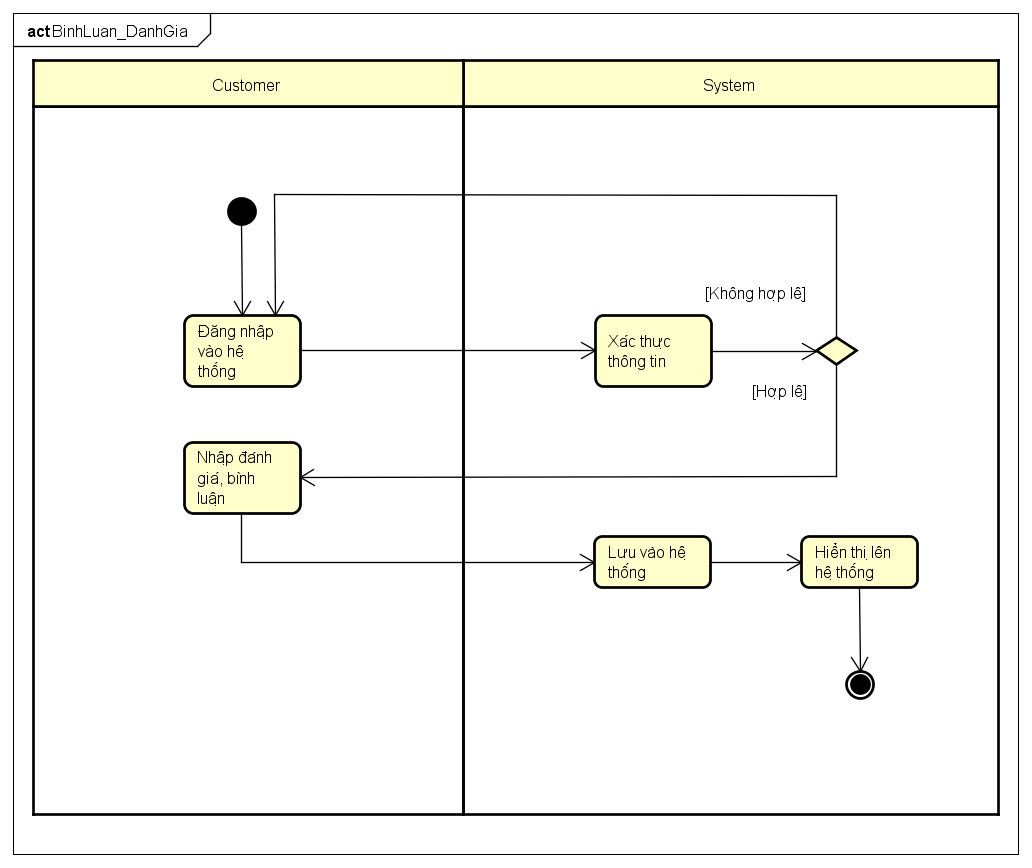
### 2.4.5 Thêm vào giỏ hàng

Dòng sự kiện



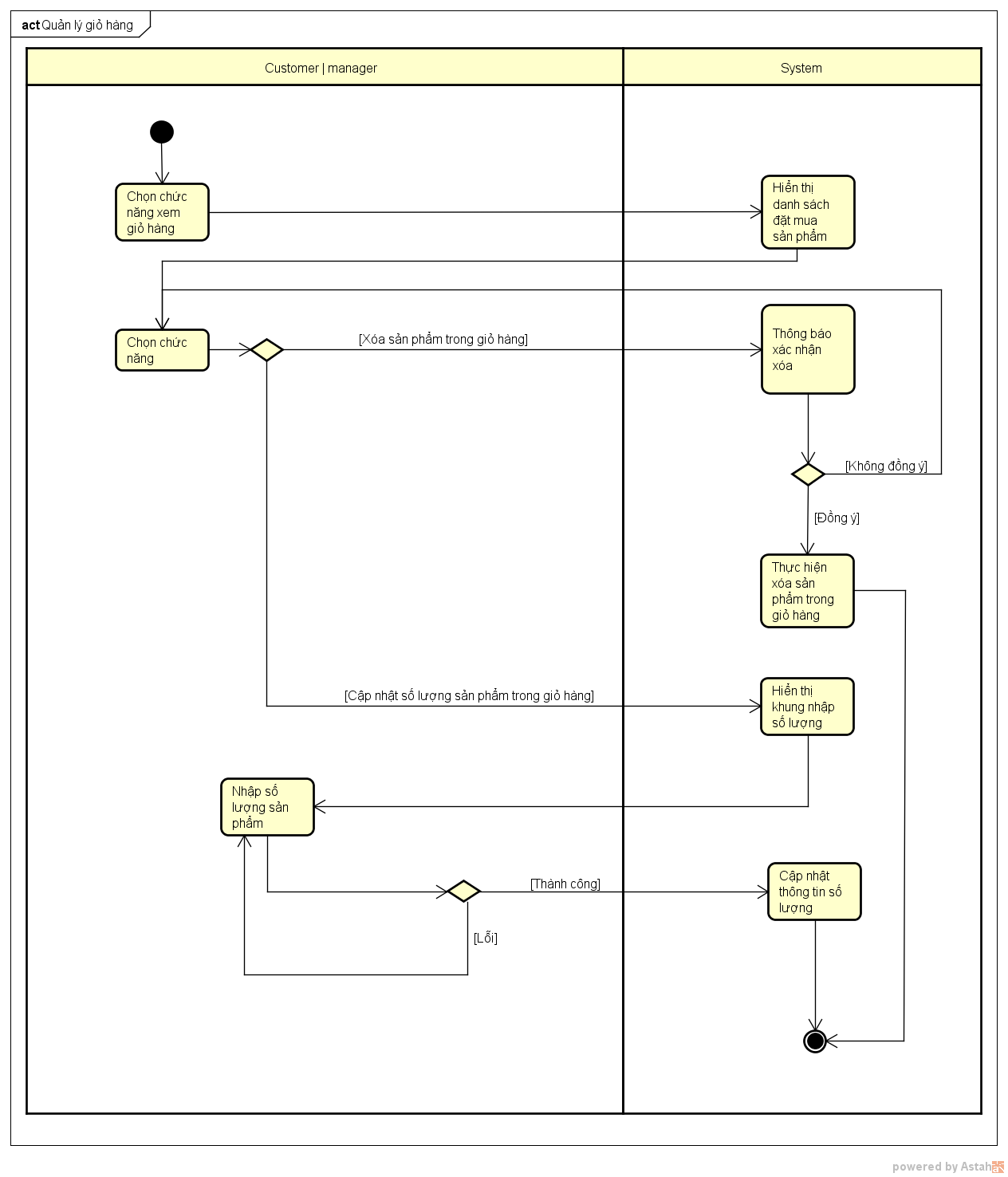
### 2.4.6 Bình luận, đánh giá

Dòng sự kiện



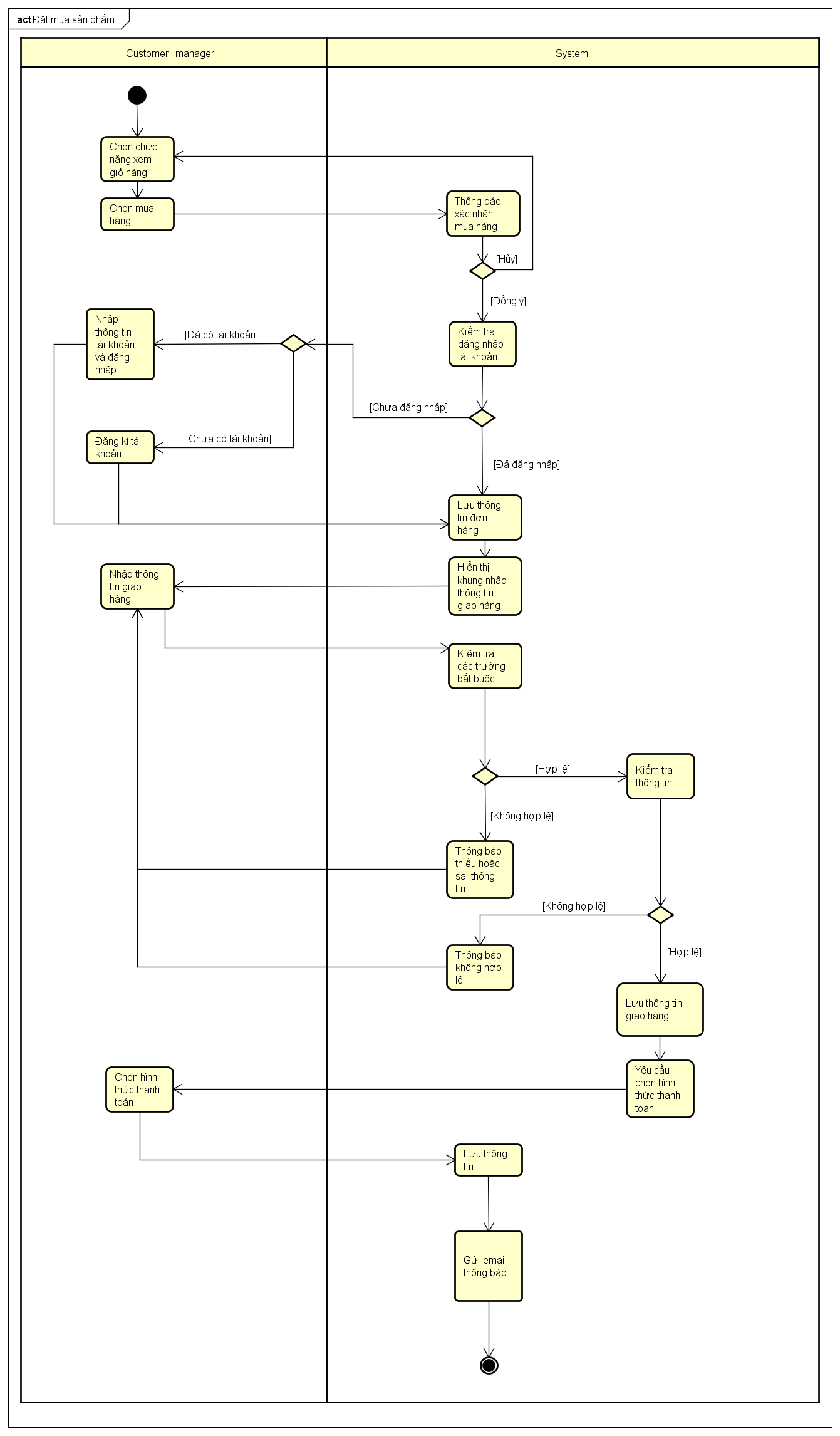
### 2.4.7 Quản lý giỏ hàng

Dòng sự kiện



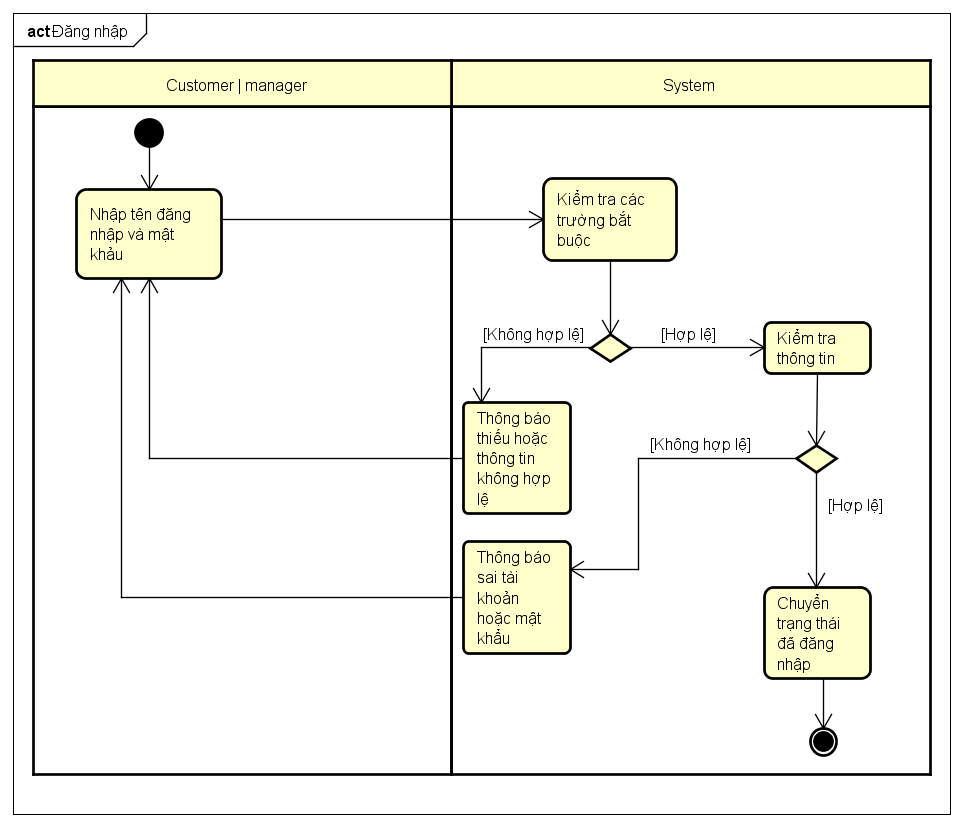
### 2.4.8 Đặt mua sản phẩm

Dòng sự kiện



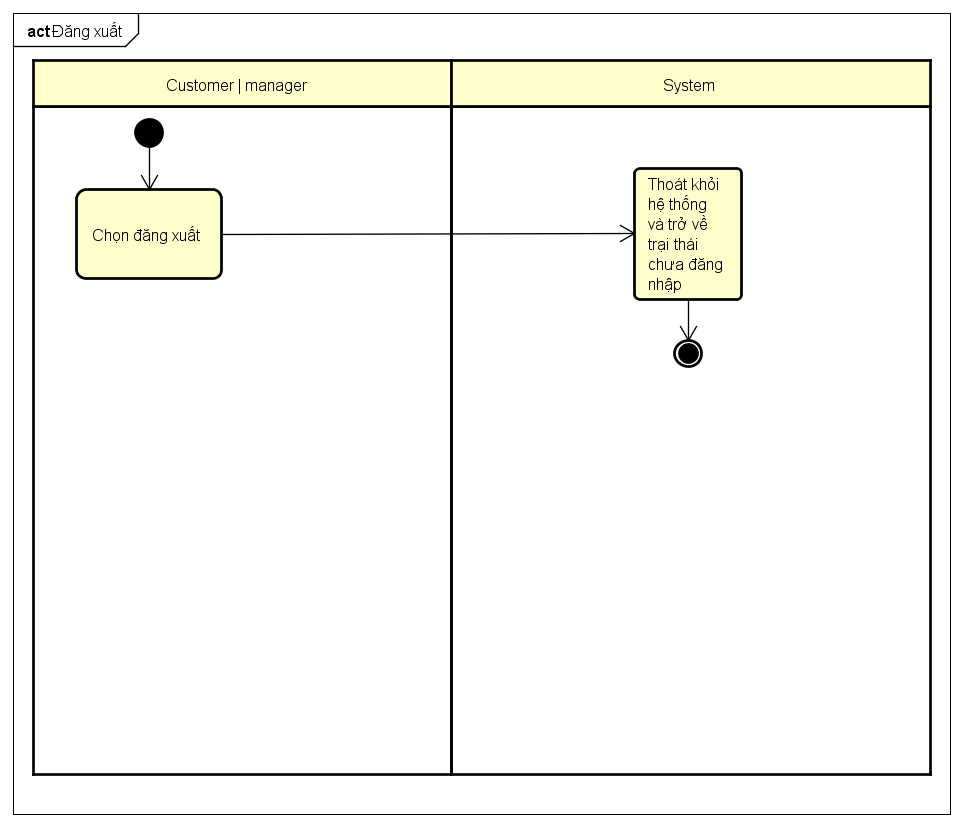
### 2.4.9 Đăng nhập

Dòng sự kiện

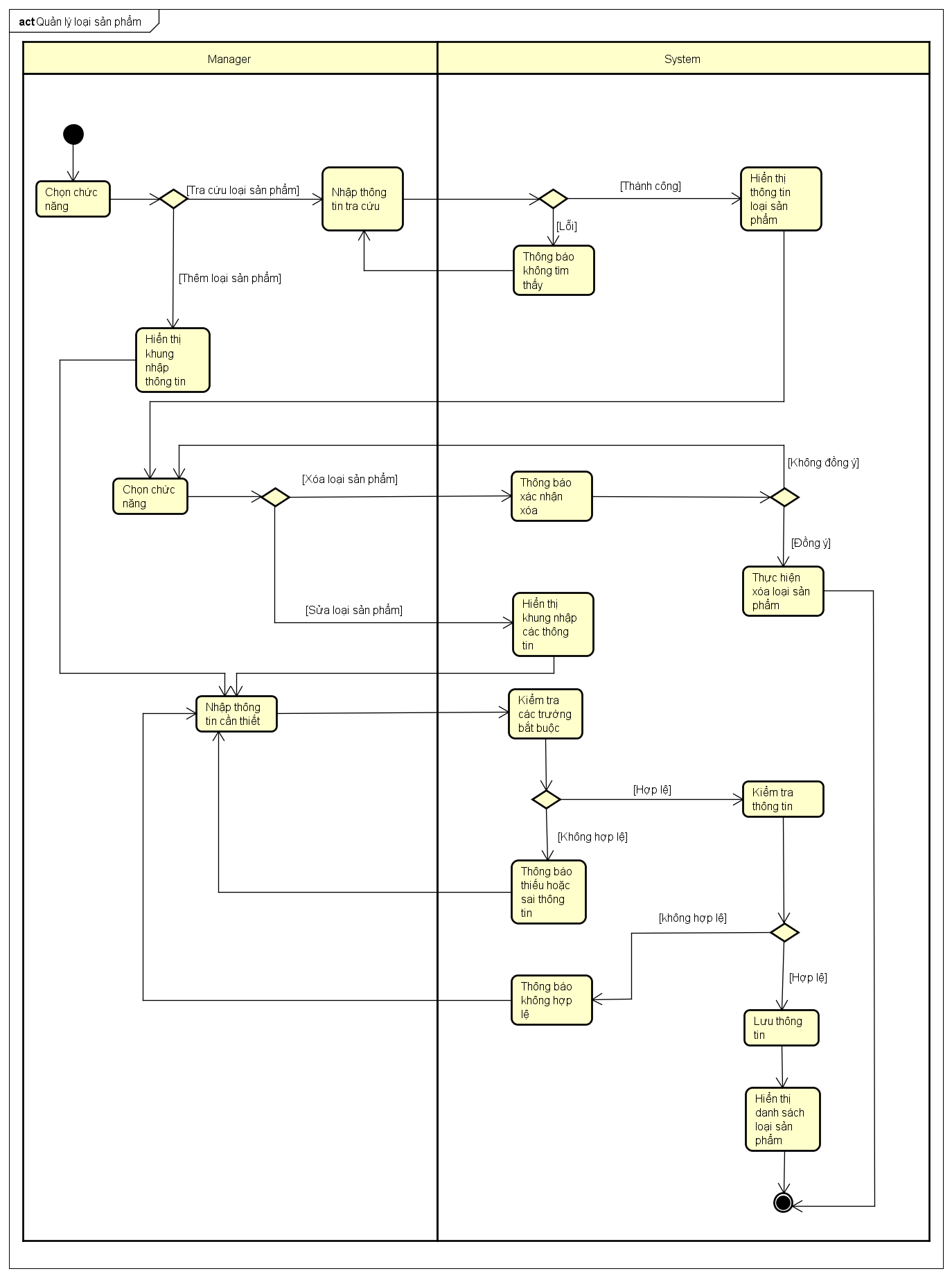


### 2.4.10 Đăng xuất

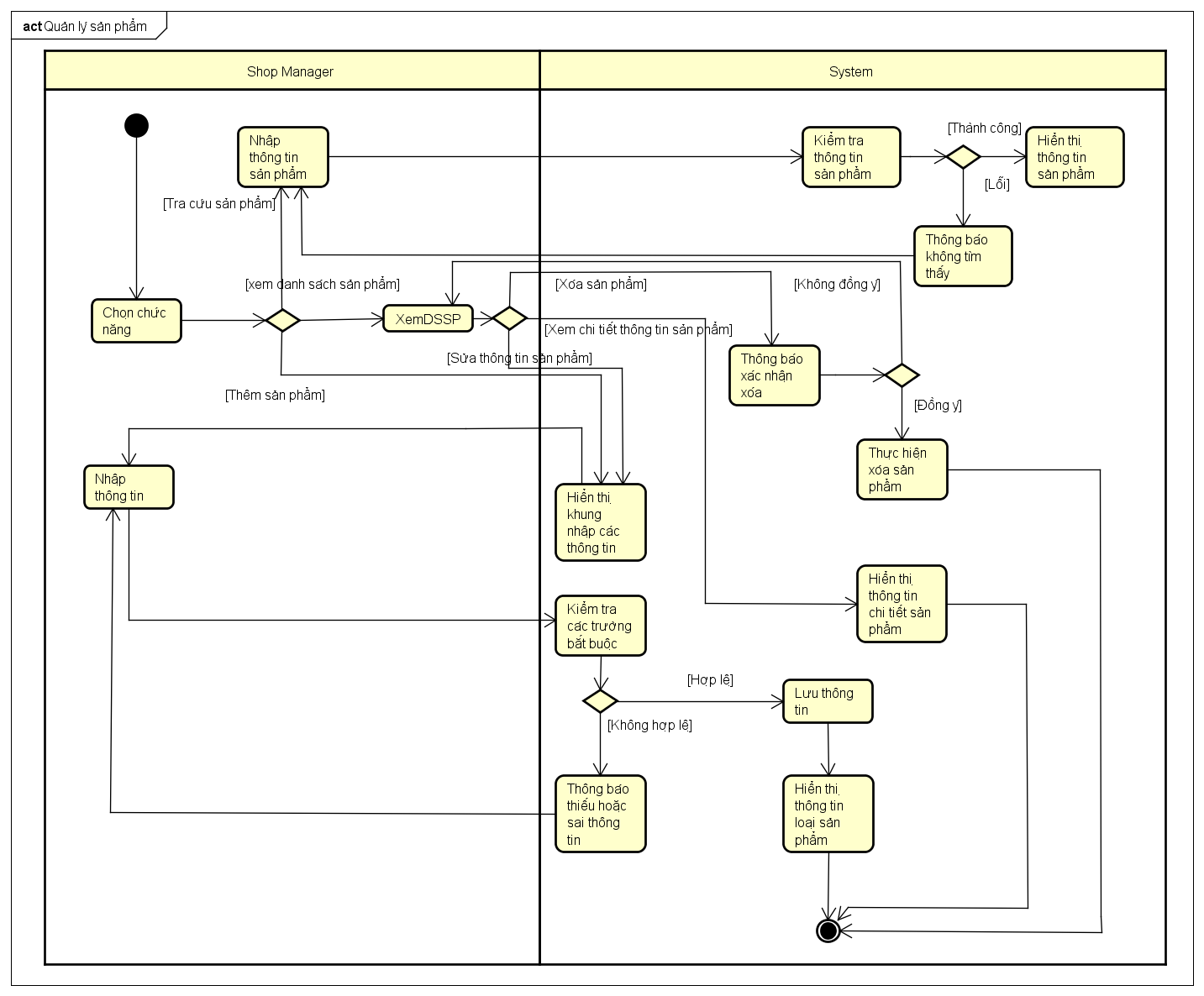
Dòng sự kiện



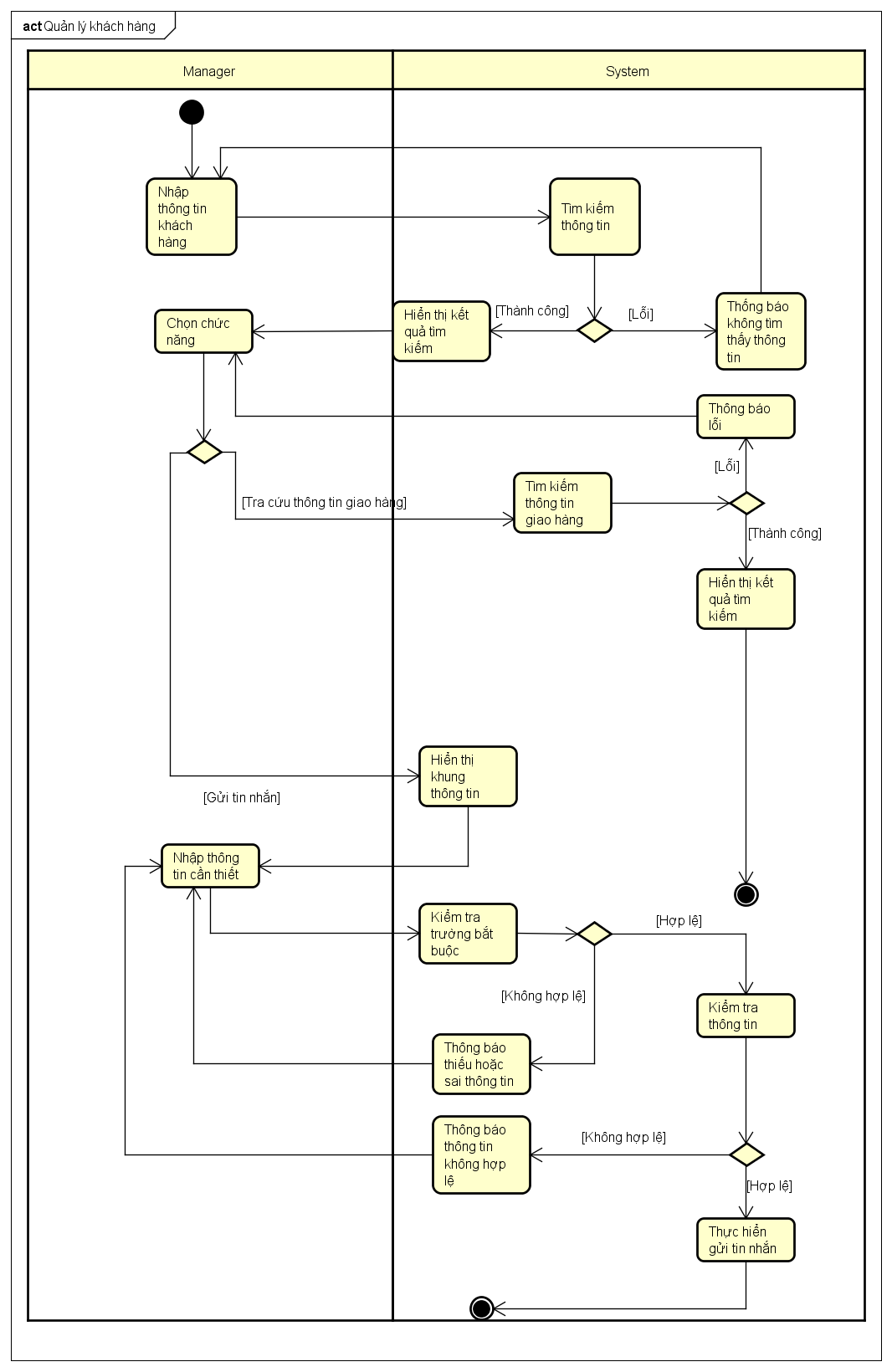
### 2.4.11 Quản lý loại sản phẩm



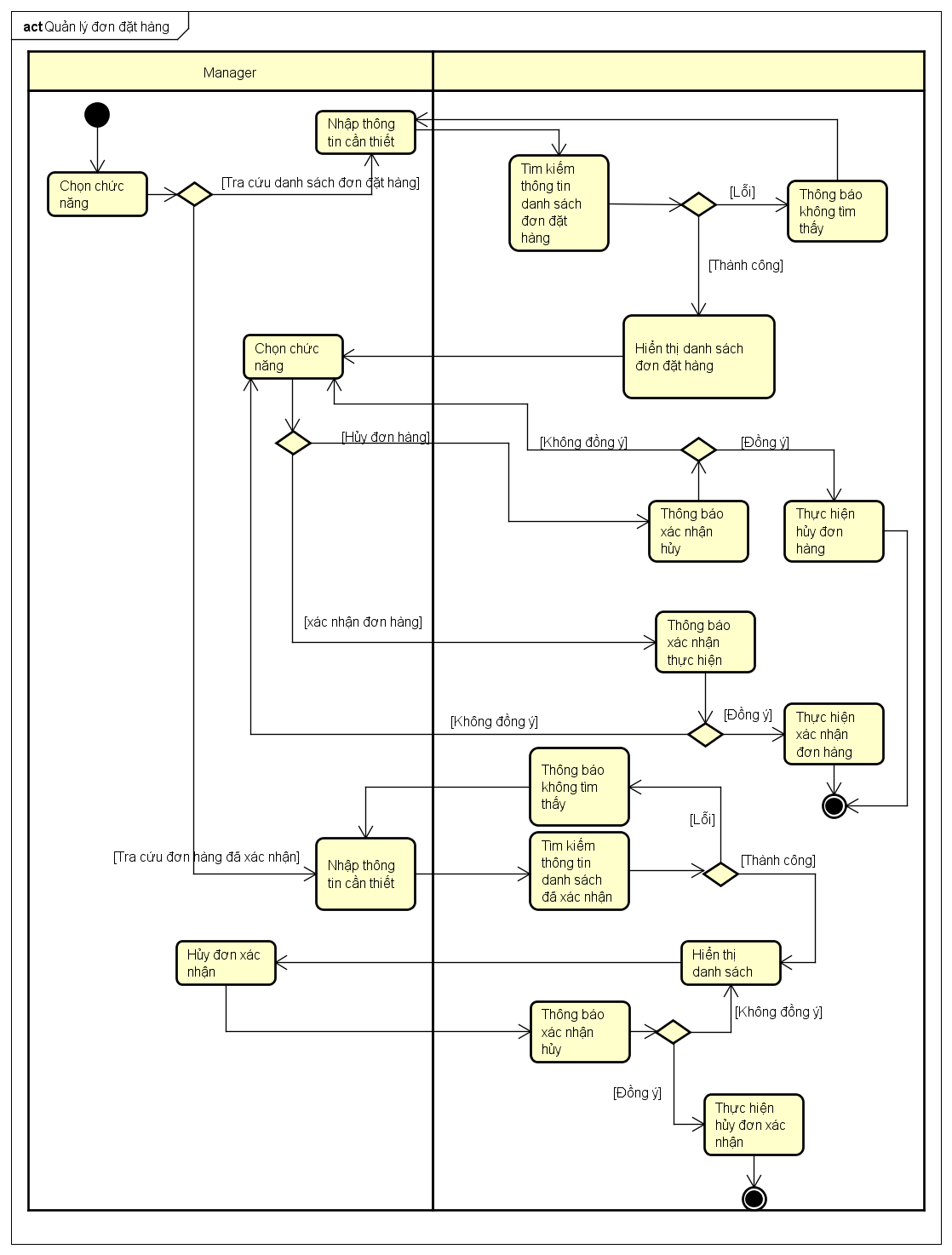
### 2.4.12 Quản lý sản phẩm



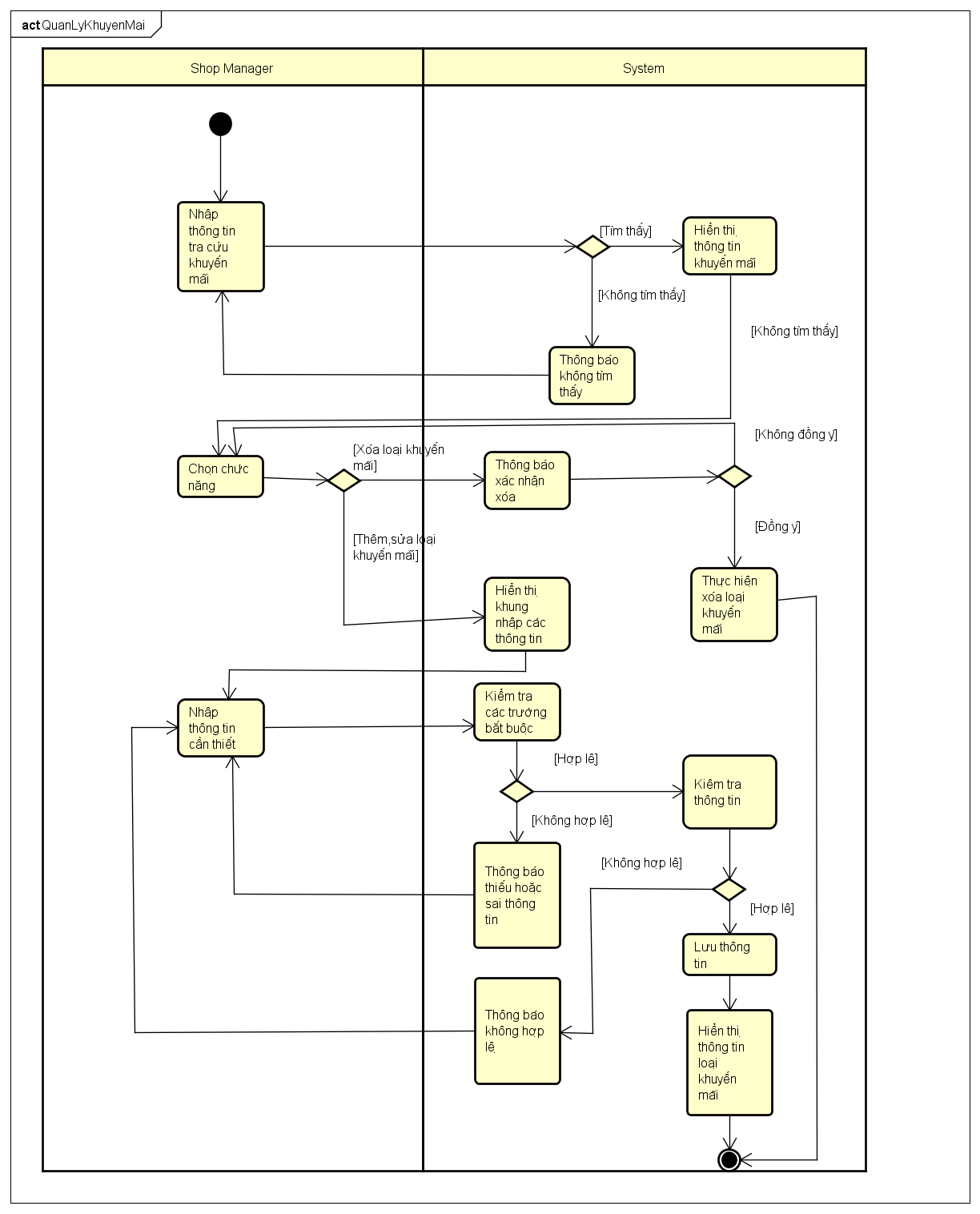
### 2.4.13 Quản lý khách hàng



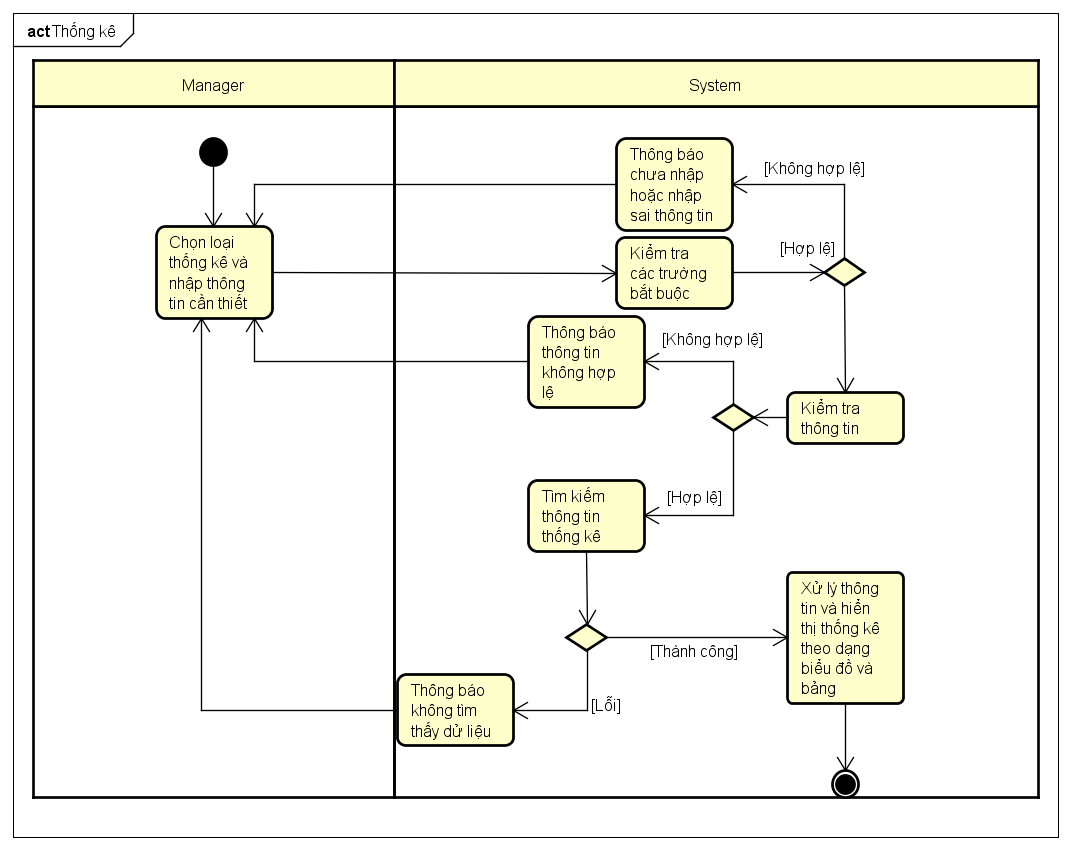
### 2.4.14 Quản lý đơn đặt hàng



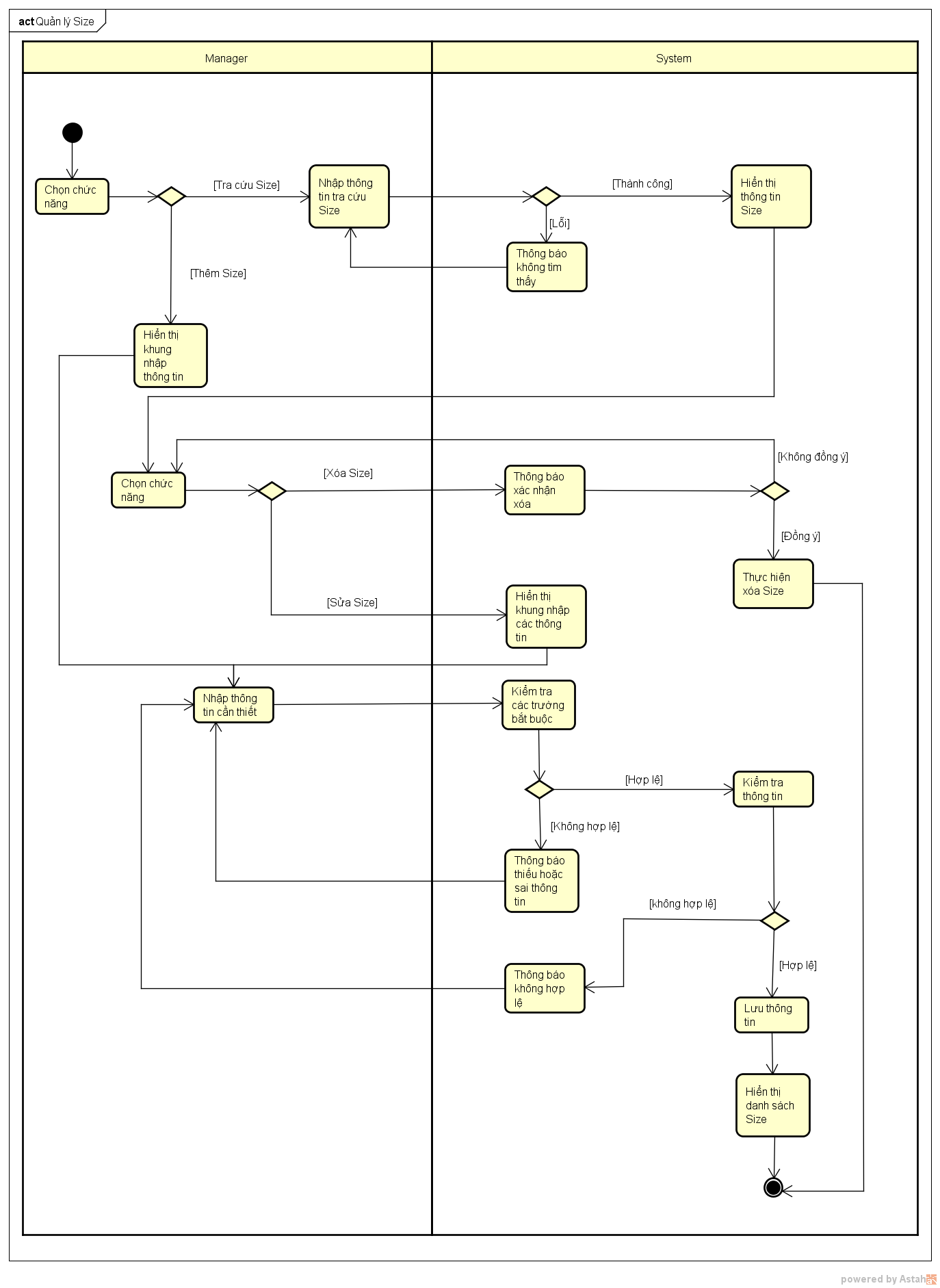
### 2.4.15 Quản lý khuyến mãi



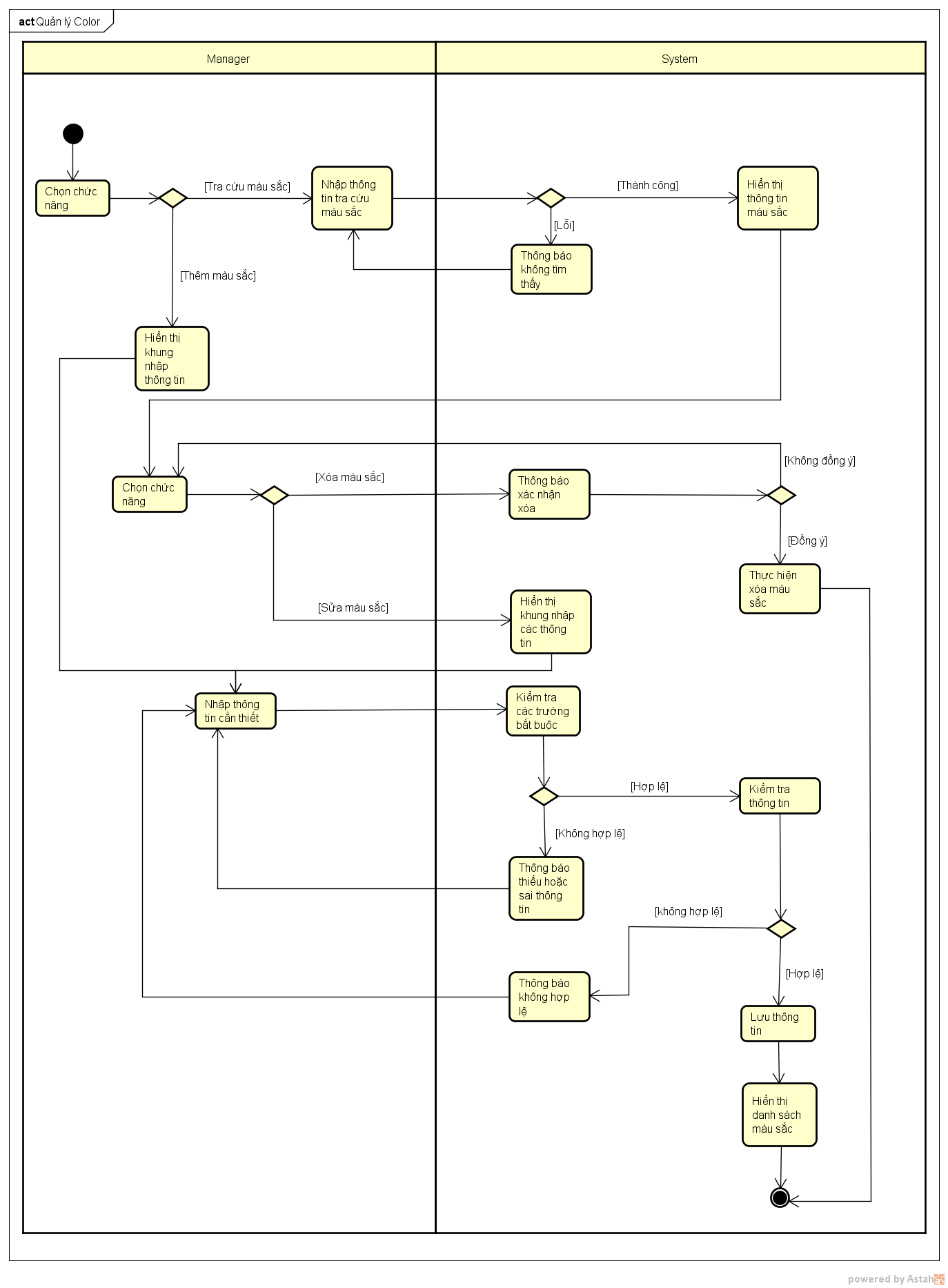
### 2.4.16 Thống kê



### 2.4.17 Quản lý size

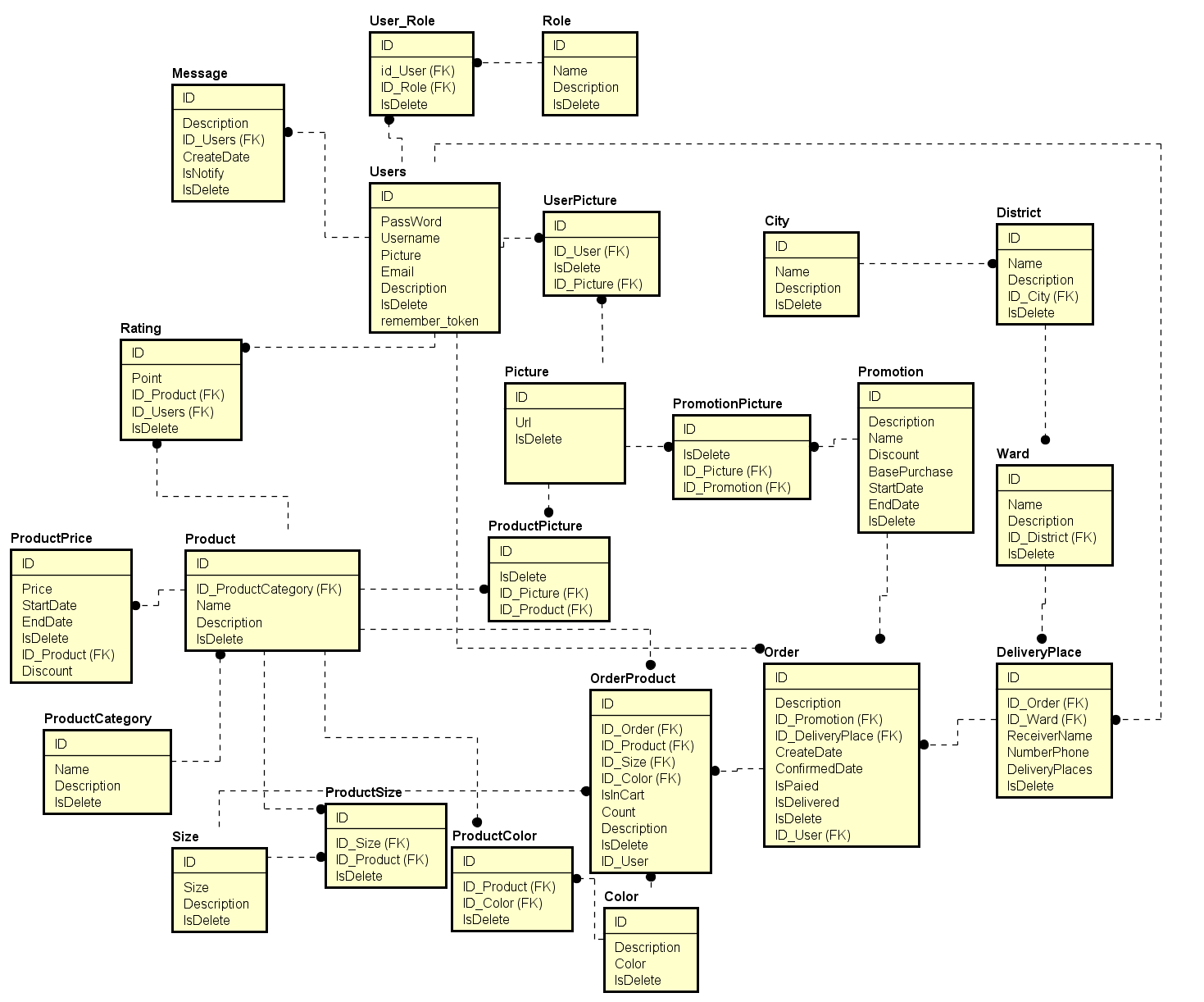


### 2.4.18 Quản lý color



### 2.4.19 Quản lý địa chỉ

* 1. **Mô hình CSDL**



# CHƯƠNG 3: XÂY DỰNG CÁC GIAO TÁC

## Trigger

Trigger là một đối tượng được định danh trong CSDL và được gắn chặt với một sự kiện xảy ra trên một bảng nào đó (điều này có nghĩa là nó sẽ được tự động thực thi khi xảy ra một sự kiện trên một bảng). Các sự kiện này bao gồm: chèn (Insert), xóa (Delete) hay cập nhật (Update) một bảng. Trigger được đưa vào MySQL từ phiên bản 5.0.2.

Trigger được sử dụng để tạo ra các công việc tự động thay cho việc phải làm thủ công bằng tay như: kiểm tra dữ liệu, đồng bộ hóa dữ liệu, đảm bảo các mối quan hệ giữa các bảng, ...

Trigger trong MySQL chỉ được kích hoạt thông qua các câu lệnh SQL, chúng không được kích hoạt bằng các thay đổi được thực thi bởi các hàm API trong bảng mà không thông qua việc truyền các câu lệnh SQL tới máy chủ MySQL. Trong thực tế, chúng không được kích họat bằng các lệnh cập nhật thông qua NDB API.

Cú pháp của trigger trong MySQL:

CREATE [DEFINER = { user | CURRENT\_USER }]

TRIGGER trigger\_name trigger\_time trigger\_event

ON tbl\_name FOR EACH ROW trigger\_stmt

- DEFINER: xác định ngữ cảnh bảo mật sẽ được sử dụng khi kiểm tra quyền truy cập tại thời điểm trigger được kích họat.

- trigger\_name: Tên của trigger.

- trigger\_time: Xác định thời điểm thực thi của trigger. Thời điểm này có thể là BEFORE (trước) hoặc AFTER (sau) để chỉ báo rằng trigger sẽ được kích họat trước hoặc sau mỗi dòng được chỉnh sửa.

- trigger\_event: xác định kiểu câu lệnh sẽ kích họat trigger. Nó chỉ có thể nhận một trong các giá trị sau đây:

 INSERT: Trigger sẽ được kích họat khi có một dòng mới được chèn vào bảng thông qua các câu lệnh như: INSERT, LOAD DATA, và REPLACE.

 UPDATE: Trigger được kích họat khi có một dòng bị chỉnh sửa bởi câu lệnh UPDATE

 DELETE: Trigger được thực hiện khi một dòng bị xóa khỏi bảng. Chú ý rằng các câu lệnh DROP TABLE và TRUNCATE sẽ không kích họat được trigger này.

- tbl\_name: Tên bảng mà trigger sẽ họat động trên đó

- trigger\_stmt: Câu lệnh sẽ được thực thi khi trigger được kích họat. Nếu như cần kích họat nhiều câu lệnh, chúng ta cần phải đưa chúng vào cấu trúc khối lệnh BEGIN ... END

### Danh sách các trigger

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | TÊN | THAO TÁC | BẢNG | NỘI DUNG |
| 1 | TG\_PRODUCT\_PRICE | INSERT | productprice | thêm record ở bảng productprice khi có 1 giá sản phẩm được thêm vào |
| 2 | TG\_UPDATE\_PRODUCT\_PRICE | UPDATE | productprice | cập nhật record ở bảng productprice khi có 1 giá sản phẩm được cập nhật thõa điều kiện |
| 3 | TRG\_UPDATE\_ORDER\_PRODUCT | UPDATE | order | cập nhật record ở bảng order khi có 1 đơn hàng sản phẩm được cập nhật thõa điều kiện |
| 4 | TRG\_DELETE\_ORDER\_PRODUCT | DELETE | order | xóa record ở bảng order khi có 1 đơn hàng sản phẩm được xóa thõa điều kiện |

### Mô tả các trigger

#### Trigger kiểm tra INSERT giá sản phẩm

Tên: TG\_PRODUCT\_PRICE

Thao tác: INSERT

Bảng: productprice

Nội dung: tại mỗi thời điểm chỉ có một giá sản phẩm, thêm record ở bảng productprice khi có 1 giá sản phẩm được thêm vào.

Mã SQL:

|  |
| --- |
| DELIMITER $$  CREATE TRIGGER TG\_PRODUCT\_PRICE AFTER INSERT ON productprice  FOR EACH ROW  BEGIN  SET @NBD := new.StartDate ;  SET @NKT := new.EndDate ;  if EXISTS(select pp.Price  FROM productprice PP  WHERE ((@NBD < PP.EndDate) AND ( @NBD > PP.StartDate))  OR ((@NKT < PP.EndDate)AND (@NKT > PP.StartDate))  OR ((pp.StartDate > @NBD) AND ( PP.EndDate < @NKT))  OR (@NBD > @NKT)  )  THEN  DELETE FROM productprice WHERE id = new.id;  END IF;  END$$  DELIMITER ; |

#### Trigger kiểm tra UPDATE giá sản phẩm

Tên: TG\_UPDATE\_PRODUCT\_PRICE

Thao tác: UPDATE

Bảng: productprice

Nội dung: tại mỗi thời điểm chỉ có một giá sản phẩm, cập nhật record ở bảng productprice khi có 1 giá sản phẩm được cập nhật.

Mã SQL:

|  |
| --- |
| DELIMITER $$  CREATE TRIGGER TG\_UPDATE\_PRODUCT\_PRICE AFTER UPDATE ON productprice  FOR EACH ROW  BEGIN  SET @NBD := new.StartDate ;  SET @NKT := new.EndDate ;  if EXISTS(select pp.Price  FROM productprice PP  WHERE ((@NBD < PP.EndDate) AND ( @NBD > PP.StartDate))  OR ((@NKT < PP.EndDate)AND (@NKT > PP.StartDate))  OR ((pp.StartDate > @NBD) AND ( PP.EndDate < @NKT))  OR (@NBD > @NKT)  )  THEN  DELETE FROM productprice WHERE id = new.id;  END IF;  END$$ |

#### Trigger cập nhập nhật order\_product

Tên: TRG\_UPDATE\_ORDER\_PRODUCT

Thao tác: UPDATE

Bảng: order

Nội dung: cập nhật record ở bảng order khi có 1 đơn hàng được cập nhật.

Mã SQL:

|  |
| --- |
| DELIMITER $$  CREATE TRIGGER TRG\_UPDATE\_ORDER\_PRODUCT AFTER UPDATE ON `order`  FOR EACH ROW  BEGIN  SET @ID := new.id;  SET @isDelete := new.IsDelete;    if @IsDelete = 1 then  UPDATE `orderproduct`  SET IsDelete = 1  WHERE @ID = `orderproduct`.ID\_Order ;  END if;  END$$  DELIMITER ; |

#### Trigger xóa toàn bộ orderproduct liên quan

Tên: TRG\_DELETE\_ORDER\_PRODUCT

Thao tác: DELETE

Bảng: order

Nội dung: xóa toàn bộ orderproduct liên quan.

Mã SQL:

|  |
| --- |
| DELIMITER $$  CREATE TRIGGER TRG\_DELETE\_ORDER\_PRODUCT AFTER DELETE ON `order`  FOR EACH ROW  BEGIN  SET @ID := OLD.id;  SET @isDelete := OLD.IsDelete;  DELETE FROM `orderproduct`  WHERE @ID = `orderproduct`.ID\_Order ;  END$$  DELIMITER ; |

## Stored-procedure

### Stored procedure trong MySQL

Stored procedure là một hoặc một tập các lệnh SQL được biên dịch để thực hiện cùng lúc mỗi khi được gọi đến. Về bản chất giống với hàm, thủ tục trong các ngôn ngữ lập trình khác. Stored procedure đã được đưa vào MySQL từ phiên bản 5.0.

Stored procedure giúp giảm thời gian giao tiếp giữa các ứng dụng với hệ quản trị MySQL, bởi vì thay vì gửi nhiều câu lệnh dài thì ta chỉ cần gọi tới một thủ tục và trong thủ tục này sẽ thực hiện nhiều câu lệnh SQL.

Stored procudure sẽ giúp các ứng dụng nhìn minh bạch hơn, nghĩa là khi ta định nghĩa các thao tác xử lý vào một Stored thì công việc của các ngôn ngữ lập trình khác chỉ quan tâm đến tên thủ tục, các tham số truyền vào chứ không cần biết nó thực hiện như thế nào. Mỗi thủ tục sẽ có các mức độ truy cập, nghĩa là ta có thể cấp quyền sử dụng cho một User nào đó trong hệ quản trị.

Stored procedure cũng có một số nhược điểm. Nếu tạo ra quá nhiều Procedure thì hệ quản trị sẽ sử dụng bộ nhớ để lưu trữ các thủ tục này khá nhiều. Ngoài ra nếu thực hiện quá nhiều xử lý trong mỗi thủ tục thì đồng nghĩa với việc CPU sẽ làm việc nặng hơn. Một số hệ quản trị CSDL có những tool hỗ trợ Debug Store nhưng MySQL thì không có nên gây kho khăn trong quá trình kiểm lỗi.

Cú pháp khai báo Stored procedure:

|  |
| --- |
| DELIMITER $$  CREATE PROCEDURE procName([definition paramName typeName], ...)  BEGIN  -- xử lý  END; $$  DELIMITER ; |

- DELIMITER $$ và DELIMITER; : dùng để phân cách bộ nhớ lưu trữ thủ tục cache và mở ra một ô lưu trữ mới cho stored này. Đây là cú pháp bắt buộc.

- procName: tên của stored proocedure.

- [definition paramName typeName]: các khai báo tham số và kiểu dữ liệu. definition biểu hiện bởi IN, OUT hoặc INOUT.

- BEGIN ... END $$: nội dung xử lý của Stored procedure nằm trong đoạn lệnh này.

### Transaction trong MySQL

Transaction (giao tác) là một nhóm câu lệnh SQL, xử lý có tuần tự các thao tác trên cơ sở dữ liệu nhưng được xem như là một đơn vị duy nhất. Vì vậy, một transaction sẽ không được coi là thành công nếu như trong quá trình xử lý có một thao tác trong nó không được hoàn thành. Trường hợp này, transaction được coi là thất bại. Trong MySQL, transaction chỉ hoạt động với table hỗ trợ nó, phổ biến nhất là InnoDB.

Các tính chất của transaction:

- Atomicity (tính nguyên tố): Đảm bảo rằng tất cả các hoạt động trong phạm vi một đơn vị làm việc là thành công hoàn toàn. Ngược lại, transaction sẽ bị hủy ngay tại điểm lỗi. Và các các họat động trước sẽ được quay lại đúng trạng thái trước đó của nó.

- Consistency (tính nhất quán): Đảm bảo cơ sở dữ liệu thay đổi đúng trạng thái khi khi transaction kết thúc thành công.

- Isolation (tính độc lập): Cho phép transaction hoạt động một cách độc lập và “trong suốt” (transparent) với các transaction khác.

- Durability (tính bền vững): Đảm bảo kết quả hoặc tác động của transaction vẫn luôn tồn tại kể cả khi hệ thống bị lỗi.

Transaction được bắt đầu bằng câu lệnh START TRANSACTION; hoặc BEGIN WORK; và kết thúc khi có lệnh COMMIT, hoặc ROLLBACK. Lệnh COMMIT xác nhận việc thực hiện giao tác thành công và ghi nhận các thay đổi. Lệnh ROLLBACK khôi phục lại các thay đổi.

Cách khai báo transaction trong MySQL:

|  |
| --- |
| DELIMITER $$  CREATE PROCEDURE **tên**\_**procedure**(**danh sách tham số**)  BEGIN  -- Khai báo biến(nếu có)  -- Khai báo xử lý handler, nếu có lỗi sẽ rollback transaction và thông báo lỗi  DECLARE EXIT HANDLER FOR SQLEXCEPTION  BEGIN  ROLLBACK;  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'ERROR';  END;  -- Khai báo bắt đầu transaction  **START TRANSACTION;**  -- Nội dung transaction  -- Commit transaction  **COMMIT;**  END; $$  DELIMITER ; |

### Danh sách các stored-procedure

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | TÊN | INPUT PARAMETER | OUTPUT PARAMETER | Ý NGHĨA | GHI CHÚ |
| 1 | Sp\_change\_info |  |  |  |  |

### Mô tả các strored-procedure

#### Thay đổi thông tin của user, xóa cũ và làm lại cái mới picture, userpicture

Tên: sp\_change\_info

Nội dung:

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_change\_info` (  IN `user\_id` INT,  IN `user\_password` VARCHAR(191),  IN `user\_description` VARCHAR(191),  IN `picture\_url` VARCHAR(191))  sp\_change\_info:BEGIN  IF NOT EXISTS(SELECT \* FROM users WHERE users.id = `user\_id`) THEN  LEAVE sp\_change\_info;  END IF;  Update users SET users.Password = `user\_password`, users.Description = `user\_description` WHERE users.id = `user\_id`;  SET @PICTUREID := (SELECT ID\_Picture FROM userspicture WHERE ID\_Users = `user\_id`);  DELETE FROM userspicture WHERE ID\_Users = `user\_id`;  DELETE FROM picture WHERE id = @PICTUREID;  INSERT INTO picture(Url,IsDelete,created\_at) VALUES(`picture\_url`,0,CURRENT\_TIMESTAMP);  SET @PICTUREID := (SELECT MAX(id) FROM picture);  INSERT INTO userspicture(ID\_Users,ID\_Picture,IsDelete,created\_at) VALUES(`user\_id`,@PICTUREID,0,CURRENT\_TIMESTAMP);  END$$ |

#### Tạo hóa đơn, tự động cập nhật danh sách còn trong cart trong OrderProduct của user

Tên: sp\_create\_order

Nội dung:

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_create\_order` (  IN `vDescription` varchar(191),  IN `vID\_Promotion` int(10),  IN `vID\_DeliveryPlace` int(10),  IN `vID\_User` int(10),  IN `vConfirmDate` date,  IN `vIsPaied` tinyint(1),  IN `vIsDelivered` tinyint(1),  IN `vIsDelete` tinyint(1)  )  BEGIN  INSERT INTO `order`(Description,ID\_Promotion,ID\_DeliveryPlace,ID\_User,ConfirmDate,IsPaied,IsDelivered,IsDelete,CreateDate) VALUES(`vDescription`,`vID\_Promotion`,`vID\_DeliveryPlace`,`vID\_User`,`vConfirmDate`,`vIsPaied`,`vIsDelivered`,`vIsDelete`,CURRENT\_TIMESTAMP);  SET @OrID := (SELECT MAX(id) FROM `order`);  UPDATE `orderproduct` SET IsInCart = 0, ID\_Order = @OrID WHERE IsDelete = 0 AND ID\_User = `vID\_User` AND `IsInCart` = 1;  END$$ |

#### Tạo user, tự động tạo picture and role of users

Tên: sp\_change\_info

Nội dung:

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_create\_users` (  IN `vUsername` varchar(191),  IN `vPassword` varchar(191),  IN `vEmail` varchar(191),  IN `vDescription` varchar(191),  IN `vIsDelete` tinyint(1),  IN `vID\_Role` tinyint(1),  IN `picture\_url` varchar(191)  )  BEGIN  INSERT INTO `users`(Username,Password,Email,Description,IsDelete) VALUES(`vUsername`,`vPassword`,`vEmail`,`vDescription`,`vIsDelete`);  SET @MAXID := (SELECT MAX(id) FROM `users`);  INSERT INTO picture(Url,IsDelete,created\_at) VALUES(`picture\_url`,0,CURRENT\_TIMESTAMP);  SET @PICTUREID := (SELECT MAX(id) FROM picture);  INSERT INTO userspicture(ID\_Users,ID\_Picture,IsDelete,created\_at) VALUES(@MAXID,@PICTUREID,0,CURRENT\_TIMESTAMP);  INSERT INTO `user\_role`(`ID\_Users`, `ID\_Role`, `IsDelete`) VALUES (@MAXID, `vID\_Role`, 0);  SELECT @MAXID as id;  END$$ |

#### Tạo create product, khởi tạo product nếu không tồn tại Id and tạo picture, size, color by list array

Tên: sp\_create\_product

Nội dung: Tạo create product, khởi tạo product nếu không tồn tại Id and tạo picture, size, color by list array

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_create\_product` (  IN `vId` int(10),  IN `vID\_ProductCategory` int(10),  IN `vName` varchar(191),  IN `vDescription` varchar(191),  IN `vIsDelete` tinyint(1),  IN `vPrice` varchar(191),  IN `vDiscount` INT,  IN `vListColor` varchar(191),  IN `vListSize` varchar(191),  IN `vListPicture` varchar(191)  )  BEGIN  IF EXISTS (SELECT \* FROM `product` WHERE id = `vId`) THEN  UPDATE `product` SET ID\_ProductCategory = `vID\_ProductCategory`, Name = `vName`, Description=`vDescription`, IsDelete =`vIsDelete` WHERE id = `vId`;  SET @MAXID := `vId`;  ELSE  INSERT INTO `product`(`ID\_ProductCategory`, `Name`, `Description`, `IsDelete`) VALUES (`vID\_ProductCategory`, `vName`, `vDescription`, `vIsDelete`);  SET @MAXID := (SELECT MAX(id) FROM `product`);  END IF;  INSERT INTO `productprice`(`Price`, `StartDate`, `EndDate`, `IsDelete`, `ID\_Product`, `Discount`, `created\_at`) VALUES (`vPrice`, CURRENT\_TIMESTAMP, NULL, 0, @MAXID, vDiscount, CURRENT\_TIMESTAMP);  set @valueColor :=`vListColor`;  WHILE (LOCATE(',', @valueColor) > 0)  DO  SET @V\_DESIGNATION = SUBSTRING(@valueColor,1, LOCATE(',',@valueColor)-1);  SET @valueColor = SUBSTRING(@valueColor, LOCATE(',',@valueColor) + 1);  INSERT INTO `productcolor`(`ID\_Product`, `ID\_Color`, `IsDelete`) VALUES (@MAXID, @V\_DESIGNATION, 0);  END WHILE;  set @valueSize :=`vListSize`;  WHILE (LOCATE(',', @valueSize) > 0)  DO  SET @V\_DESIGNATION = SUBSTRING(@valueSize,1, LOCATE(',',@valueSize)-1);  SET @valueSize = SUBSTRING(@valueSize, LOCATE(',',@valueSize) + 1);  INSERT INTO `productcolor`(`ID\_Product`,`ID\_Size`, `IsDelete`) VALUES (@MAXID, @V\_DESIGNATION, 0);  END WHILE;  set @valuePicture :=`vListPicture`;  WHILE (LOCATE(',', @valuePicture) > 0)  DO  SET @V\_DESIGNATION = SUBSTRING(@valuePicture,1, LOCATE(',',@valuePicture)-1);  SET @valuePicture = SUBSTRING(@valuePicture, LOCATE(',',@valuePicture) + 1);  INSERT INTO picture(Url,IsDelete,created\_at) VALUES(@V\_DESIGNATION,0,CURRENT\_TIMESTAMP);  SET @PICTUREID := (SELECT MAX(id) FROM picture);  INSERT INTO `productprice`(`ID\_Product`,`ID\_Picture`,`IsDelete`) VALUES(@MAXID,@PICTUREID,0,CURRENT\_TIMESTAMP);  END WHILE;  select @MAXID as id;  END$$ |

#### Tạo khuyến mãi với picture

Tên: sp\_create\_promotion

Nội dung: Tạo khuyến mãi với picture

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_create\_promotion` (  IN `vId` int(10),  IN `vDescription` varchar(191),  IN `vName` varchar(191),  IN `vDiscount` varchar(191),  IN `vBasePurchase` double(8,2),  IN `vStartDate` date,  IN `vEndDate` date,  IN `vIsDelete` tinyint(1),  IN `vPicture` varchar(191)  )  BEGIN  IF EXISTS (SELECT \* FROM `promotion` WHERE id = `vId`) THEN  UPDATE `promotion` SET Description = `vDescription`, Name = `vName`, Discount=`vDiscount`, BasePurchase =`vBasePurchase`, StartDate =`vStartDate`, EndDate = `vEndDate`, IsDelete = `vIsDelete` WHERE id = `vId`;  SET @MAXID := `vId`;  ELSE  INSERT INTO `promotion`(`Description`, `Name`, `Discount`, `BasePurchase`, `StartDate`, `EndDate`, `IsDelete`) VALUES(`vDescription`, `vName`, `vDiscount`, `vBasePurchase`, `vStartDate`, `vEndDate`, `vIsDelete`);  SET @MAXID := (SELECT MAX(id) FROM `promotion`);  END IF;  INSERT INTO picture(Url,IsDelete,created\_at) VALUES(`vPicture`,0,CURRENT\_TIMESTAMP);  SET @PICTUREID := (SELECT MAX(id) FROM picture);  INSERT INTO `promotionpicture`(`ID\_Promotion`,`ID\_Picture`,`IsDelete`) VALUES(@MAXID,@PICTUREID,0);  select @MAXID as id;  END$$ |

#### Thống kê doanh thu theo thời gian

Tên: sp\_statistic\_revenue

Nội dung: Thống kê doanh thu theo thời gian

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_statistic\_revenue` (  IN `vStartDate` DATE,  IN `vEndDate` DATE  )  BEGIN  SELECT o.id,o.CreateDate,IF(m.BasePurchase IS NOT NULL AND m.BasePurchase <= SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100)),SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100))\*(100-m.Discount)/100,SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100))) Price FROM `order` o JOIN `orderproduct` as op ON o.id = op.ID\_Order  JOIN `product` p ON op.ID\_Product = p.id  JOIN `productprice` pr ON p.id = pr.ID\_Product AND ((pr.StartDate <= o.CreateDate AND pr.EndDate > o.CreateDate) || (pr.StartDate <= o.CreateDate AND pr.EndDate IS NULL))  LEFT JOIN `promotion` m ON o.ID\_Promotion = m.id  WHERE `vStartDate` <= o.CreateDate AND o.CreateDate <= `vEndDate` AND o.IsPaied = 1  GROUP BY o.CreateDate,o.id  ORDER BY o.CreateDate ASC;  END$$ |

#### Thống kê theo sản phẩm theo khoản thời gian

Tên: sp\_statistic\_bestsell

Nội dung: Thống kê theo sản phẩm theo khoản thời gian

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_statistic\_bestsell` (  IN `vStartDate` DATE,  IN `vEndDate` DATE  )  BEGIN  SELECT p.id,o.CreateDate,p.Name,count(DISTINCT o.id) as `count`,IF(m.BasePurchase IS NOT NULL AND m.BasePurchase <= SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100)),SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100))\*(100-m.Discount)/100,SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100))) Price FROM `order` o JOIN `orderproduct` as op ON o.id = op.ID\_Order  JOIN `product` p ON op.ID\_Product = p.id  JOIN `productprice` pr ON p.id = pr.ID\_Product AND ((pr.StartDate <= o.CreateDate AND pr.EndDate > o.CreateDate) || (pr.StartDate <= o.CreateDate AND pr.EndDate IS NULL))  LEFT JOIN `promotion` m ON o.ID\_Promotion = m.id  WHERE `vStartDate` <= o.CreateDate AND o.CreateDate <= `vEndDate` AND o.IsPaied = 1  GROUP BY o.CreateDate,p.id,p.Name  ORDER BY o.CreateDate ASC;  END$$ |

#### Thống kê theo khách hàng theo khoản thời gian

Tên: sp\_statistic\_bestuser

Nội dung: Thống kê theo khách hàng theo khoản thời gian

Tham số đầu vào:

Tham số đầu ra:

Mã SQL:

|  |
| --- |
| CREATE PROCEDURE `sp\_statistic\_bestuser` (  IN `vStartDate` DATE,  IN `vEndDate` DATE  )  BEGIN  SELECT u.Username,count(DISTINCT o.id) as `count`,IF(m.BasePurchase IS NOT NULL AND m.BasePurchase <= SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100)),SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100))\*(100-m.Discount)/100,SUM(op.Count\*(pr.Price \* (100- pr.Discount) / 100))) Price FROM `order` o JOIN `orderproduct` as op ON o.id = op.ID\_Order  JOIN `product` p ON op.ID\_Product = p.id  JOIN `productprice` pr ON p.id = pr.ID\_Product AND ((pr.StartDate <= o.CreateDate AND pr.EndDate > o.CreateDate) || (pr.StartDate <= o.CreateDate AND pr.EndDate IS NULL))  LEFT JOIN `promotion` m ON o.ID\_Promotion = m.id  JOIN `users` u ON o.ID\_User = u.id  WHERE `vStartDate` <= o.CreateDate AND o.CreateDate <= `vEndDate` AND o.IsPaied = 1  GROUP BY u.Username  ORDER BY u.Username ASC;  END$$  DELIMITER ; |

# CHƯƠNG 4: THIẾT KẾ GIAO DIỆN

## 3.1 Danh sách các trang chính trong chương trình

|  |  |
| --- | --- |
| STT | Tên trang |
| 1 | Trang chủ |
| 2 | Trang xem thông tin sản phẩm |
| 3 | Trang giỏ hàng |
| 4 | Trang địa chỉ giao hàng |
| 5 | Trang quản lý admin |
| 6 | Trang báo cáo |
| 7 | Trang quản lý sản phẩm |
| 8 | Thông tin tài khoản đăng nhập |
| 9 | Trang quản lý nhóm sản phẩm |
| 10 | Trang quản lý màu sắc |
| 11 | Trang quản lý kích thước |
| 12 | Trang quản lý khuyến mãi |

## 3.2 Giao diện các trang

### 3.2.1 Trang chủ

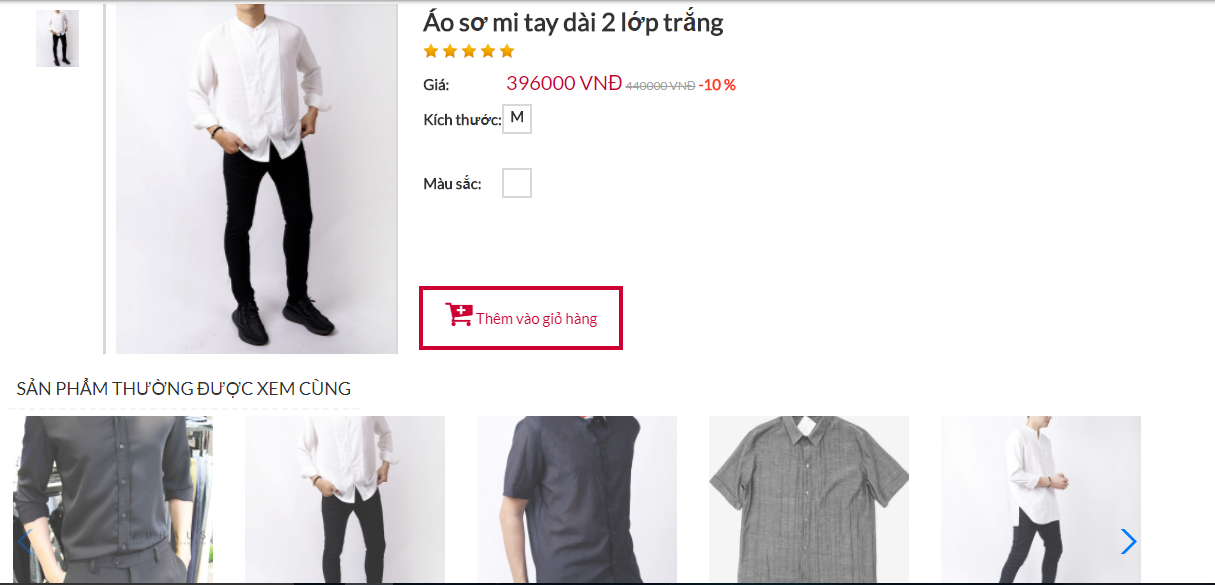
A person posing for the camera

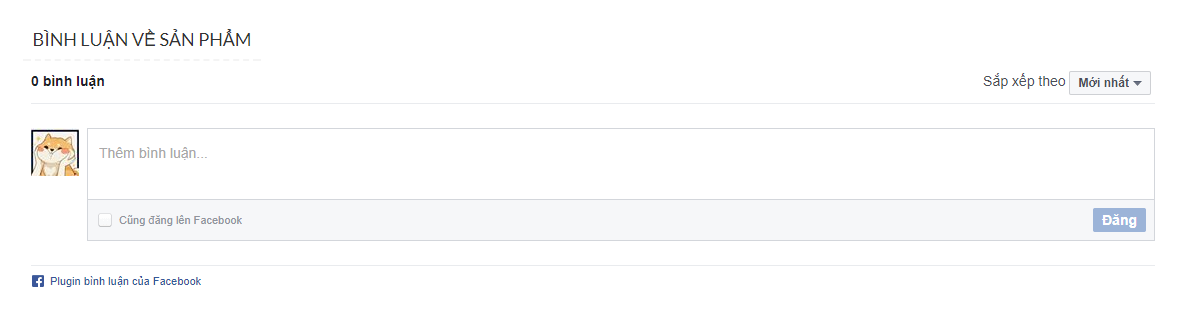
Description generated with very high confidenceA group of people standing in a room

Description generated with very high confidenceA group of people standing in a room

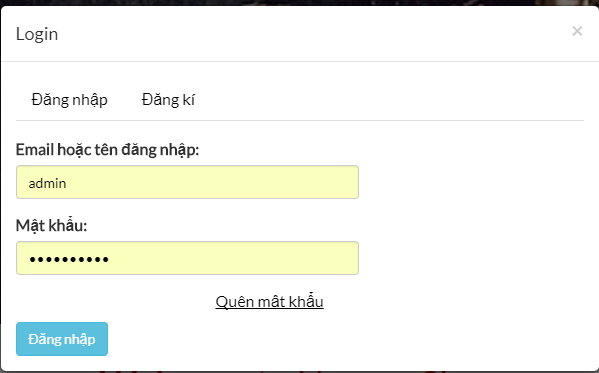
Description generated with very high confidence

### 3.2.2 Trang xem thông tin chi tiết sản phẩm





### 3.2.3 Trang đăng nhập

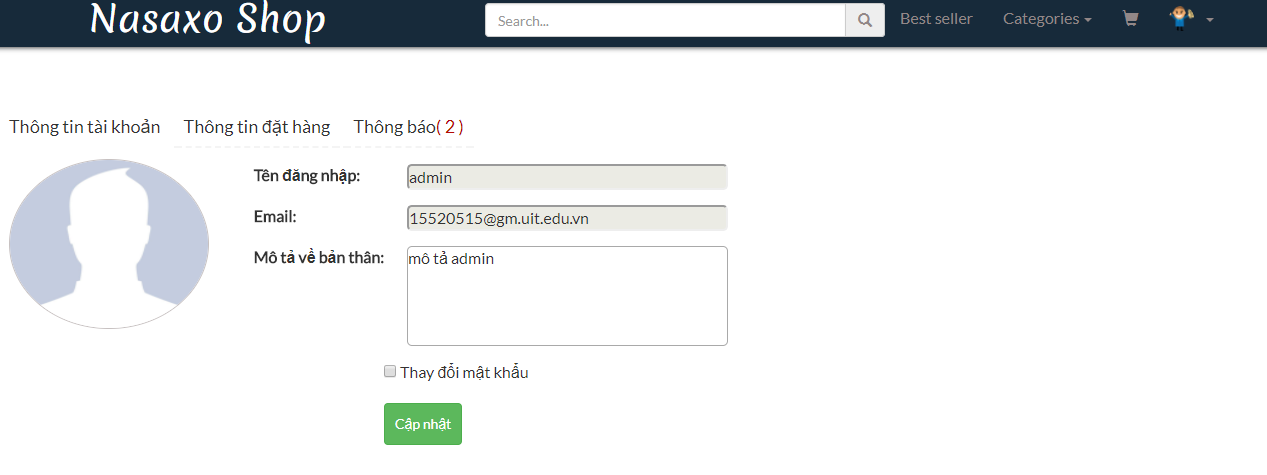


### 3.2.4 Trang giỏ hàng

A screenshot of a cell phone

Description generated with very high confidence

### 3.2.5 Trang thông tin tài khoản đăng nhập



### 3.2.6 Trang địa chỉ giao hàng

A screenshot of a cell phone

Description generated with very high confidence

### 3.2.7 Trang báo cáo

A screenshot of a cell phone

Description generated with very high confidenceA screenshot of a cell phone

Description generated with very high confidence

### 3.2.8 Trang quản lý sản phẩm

A screenshot of a computer screen

Description generated with very high confidence

### 3.2.9 Trảng quản lý nhóm sản phẩm

A screenshot of a computer

Description generated with very high confidence

### 3.2.10 Trang quản lý màu sắc

A screenshot of a computer

Description generated with very high confidence

### 3.2.11 Trang quản lý kích thước

A screenshot of a computer

Description generated with very high confidence

### 3.2.12 Trang quản lý khuyến mãi

A screenshot of a computer

Description generated with very high confidence

# CHƯƠNG 4 FRAMEWORK LARAVEL

## 4.1 Lịch sử phát triển

* Được phát triển bởi Taylor Otwell
* 9/6/2011 phát hành bản Laravel beta và tiếp đó là **Laravel 1**. Laravel 1 bao gồm các tính năng như xác thực, bản địa hóa, model, view, session, định tuyến và các cơ cấu khác, nhưng vẫn còn thiếu controller, điều này làm nó chưa thật sự là một MVC framework đúng nghĩa.
* 9/2011 phát hành bản Laravel 2. Được năng cấp lên một số chức năng như hỗ trợ controller hỗ trợ Inversion of Control ([IoC](https://en.wikipedia.org/wiki/Inversion_of_control)), hệ thống template Blade. Bên cạnh đó, có một nhược điểm là hỗ trợ cho các gói của nhà phát triển bên thứ 3 bị gỡ bỏ.
* 2/2012 phát hàng bản Laravel 3 hỗ trợ thêm một số chức năng nổi bật như giao diện dòng lệnh (CLI) tên “Artisan”, hỗ trợ nhiều hơn cho hệ thống quản trị cơ sở dữ liệu, chức năng ánh xạ cơ sở dữ liệu Migration, hỗ trợ “bắt sự kiện” trong ứng dụng, và hệ thống quản lý gói gọi là “Bundles”
* 5/2013 phát hàng bản Laravel 4 Lần này thực sự là sự lột xác của Laravel framework, di chuyển và tái cấu trúc các gói hỗ trợ vào một tập được phân phối thông qua Composer, một chương trình quản lý gói thư viện phụ thuộc độc lập của PHP
* 2/2015 phát hành bản Laravel 5 giới thiệu cấu trúc cây thư mục nội bộ cho phát triển ứng dụng mới. Hỗ trợ lập lịch định kỳ, một lớp trừu tượng gọi là “Flysystem” cho phép điều khiển việc lưu trữ từ xa đơn giản như lưu trữ trên máy local
* 6/2015 phát hàng bản Laravel 5.1 nhận được hỗ trợ dài hạng (LTS) với một kết hoạch fix bug lên tới 2 năm vào hỗ trợ vá lỗi bảo mật lên tới 3 năm

## 4.2 Ưu điểm nổi bật

* Mã nguồn mở và hoàn toàn free
* Sử dụng **Blade template** và các công cụ như **Elixir** giúp bạn dễ dàng quản lý các assets( biên dịch, ghép nối, nén ...), cũng như chia nhỏ code HTML ra thành nhiều phần. Giúp cho việc thiết kế cũng như quản lý các assets trở nên vô cùng đơn giản
* Được xây dựng dựa trên hơn 20 thư viện khác nhau. Dễ dàng mở rộng
* Sử dụng MVC pattern để xây dựng framework
* Thay đổi tùy chọn và cấu hình để phù hợp với nhiều môi trường. Các developer cũng có thể viết các gói (package) có thể là một tính năng hay một đoạn code gì đó. Nó dễ dàng được khai báo trong Laravel thông qua các provider

# CHƯƠNG 5 KẾT LUẬN

## 5.1 Môi trường phát triển ứng dụng

* Hệ điều hành: Microsoft windows 10
* Hệ quản trị CSDL: MySQL
* Phần mềm sử dụng: sublime text, xampp

## 5.2 Kết quả đạt được

### 5.2.1 Kết quả

Website bán quần áo giúp cho người bán hàng có thể đăng sản phẩm của mình lên website, quản lý khách hàng, quản lý các chương trình khuyến mãi, báo cáo doanh thu sản phẩm… Sau đây là một số chức năng cơ bản đạt được của website

* Quản lý sản phẩm, nhóm sản phẩm, màu sắc, kích thước
* Quản lý khuyến mãi
* Quản lý giỏ hàng
* Báo cáo thống kê
* Quản lý địa chỉ giao hàng
* Quản lý giỏ hàng
* Quản lý khách hàng
* Quản lý tài khoản

### 5.2.2 Hạn chế

* Chưa thanh toán online qua ngân hàng

## 5.3 Tài liệu đính kèm

* Link github: https://github.com/hoangnam1997/WebsiteBanQuanAo