

**HỌC VIỆN CÔNG NGHỆ BUỔI CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN 2**



Báo cáo cuối kỳ

Đề tài: Xây dựng hệ thống bán thú cưng

Môn học: Phát triển phần mềm hướng dịch vụ

Giảng viên hướng dẫn: Huỳnh Trung Trụ

Thành viên:

N20DCCN034

Vũ Văn Lâm

N20DCCN035

Trần Gia Long

N20DCCN061

Đoàn Ngọc Tài

N20DCCN141

Hà Xuân Thành

Thông tin thành viên trong nhóm 15

Thông tin nhóm 15			
STT	Họ và tên	MSSV	Email
1	Vũ Văn Lâm	N20DCCN034	n20dccn034@student.ptithcm.edu.vn
2	Trần Gia Long	N20DCCN035	n20dccn035@student.ptithcm.edu.vn
3	Đoàn Ngọc Tài	N20DCCN061	n20dccn061@student.ptithcm.edu.vn
4	Hà Xuân Thành	N20DCCN141	n20dccn141@student.ptithcm.edu.vn

Bảng phân công		
STT	Họ và tên	Phân công
1	Vũ Văn Lâm	Lập trình Microservice, Restful API
2	Trần Gia Long	Lập trình app Android
3	Đoàn Ngọc Tài	Lập trình website (bán hàng)
4	Hà Xuân Thành	Lập trình website (quản trị)

Tài nguyên đồ án		
STT	Tên	Đường dẫn
1	Microservice	https://github.com/lamvu2010/Microservices-BanThuCung.git
2	App mobile	https://github.com/SKDragon18/App_PetShop.git
3	Website React JS	https://github.com/Rangdog/FrontendWebBanThucung.git

Mục lục

Chương I: Tổng quan	1
1. Giới thiệu về tài	1
2. Chức năng chính	1
a. Quản lý thông tin thú cưng, sản phẩm:	1
b. Quản lý thông tin nhân viên:	1
c. Giao dịch:	1
d. Quản lý tài khoản:	1
3. Đối tượng sử dụng.....	1
4. Khảo sát yêu cầu	2
Chương II: Phân tích và thiết kế	4
1. Bảng thuật ngữ.....	4
2. Xác định tác nhân	4
3. Yêu cầu chức năng	5
4. Yêu cầu phi chức năng	6
5. Thiết kế usecase.....	7
6. Thiết kế biểu đồ hoạt động.....	19
7. Thiết kế biểu đồ tuần tự	25
8. Thiết kế biểu đồ trạng thái.....	28
9. Thiết kế biểu đồ lớp	29
10. Thiết kế design pattern.....	32
11. Thiết kế kiến trúc hệ thống.....	34
a. Kiến trúc	34
b. Công nghệ sử dụng	35
12. Thiết kế cơ sở dữ liệu.....	39
Chương III: Tài liệu API.....	53
1. Các quy ước chung.....	53
2. Identity service	53
a. Bảng API.....	53
b. Chi tiết:	54
3. Center Service	82
a. Bảng API.....	82

b. Chi tiết:	84
4. Order Service	148
a. Bảng API.....	148
b. Chi tiết.....	148
Chương IV: Triển khai webservice trên docker	158
Chương V: Triển khai trên Website React Js	160
1. Quản lý	161
a. Load dữ liệu.....	161
b. Thêm dữ liệu.	163
c. Sửa thông tin.....	166
d. Xóa thông tin	168
e. Hình ảnh.....	169
2. Bán hàng	171
a. Trang chủ.....	171
b. Giỏ hàng.....	177
c. Đặt hàng.....	181
Chương VI: Triển khai trên Mobile Android	186
1. Quản lý	186
a. Load dữ liệu.....	186
b. Thêm dữ liệu.	190
c. Sửa thông tin.....	193
d. Xóa thông tin	195
e. Hình ảnh.....	197
2. Quản lý chi tiết	202
3. Đặt hàng và thanh toán	204
Chương VII: Tổng kết	211
Tài liệu kham khảo	212

Chương I: Tổng quan

1. Giới thiệu về đề tài

- Hệ thống bán thú cưng là một ứng dụng và một website dành cho các cửa hàng hay chuỗi chi nhánh thu mua và buôn bán thú cưng online. Đề tài tập trung vào việc phát triển một phần mềm đa năng, giúp người dùng quản lý thông tin về thú cưng, sản phẩm, dịch vụ và giao dịch trong cửa hàng. Phạm vi quy mô của đề tài là trên nhiều chi nhánh và có tối thiểu 1 người quản lý quản trị các chi nhánh.

2. Chức năng chính

a. Quản lý thông tin thú cưng, sản phẩm:

- + Các thao tác thêm, xóa, sửa về thú cưng/ sản phẩm ...
- + Hiển thị danh sách, tra cứu.
- + Nhập hàng.
- + Thay đổi giá cả.

b. Quản lý thông tin nhân viên:

- + Các thao tác thêm, xác nhận nghỉ và cập nhật thông tin nhân viên.
- + Quản trị các chi nhánh.

c. Giao dịch:

- + Người mua có thể đặt hàng, thanh toán.
- + Xem thông tin giao dịch trên hóa đơn.

d. Quản lý tài khoản:

- + Admin có thể reset và khóa tài khoản.

3. Đối tượng sử dụng

- Khách hàng
- Quản lý các chi nhánh
- Nhân viên cửa hàng
- Admin

4. Khảo sát yêu cầu

- Bối cảnh: Hệ thống gồm nhiều chi nhánh và có tối thiểu một quản lý quản trị các chi nhánh, các nhân viên thuộc từng chi nhánh có nhiệm vụ nhập hàng và trông coi cửa hàng, tiếp nhận đơn hàng, lập hóa đơn. Người dùng sử dụng app để mua đặt thú cưng hoặc sản phẩm trong hệ thống cửa hàng. Trong môi trường hệ thống, quản lý và admin vẫn được xem là một nhân viên.

- Quy trình tổ chức:

+ Nhân viên mới khi đến cửa hàng sẽ phải cung cấp các thông tin cá nhân cho người quản lý. Quản lý xem hồ sơ và nhập vào hệ thống. Các thông tin này chỉ được thay đổi bởi người quản lý. Thông tin cơ bản gồm: họ, tên, cản cước, số điện thoại, email và chức vụ. Khi phát hiện thông tin sai, cần thay đổi, nhân viên chủ động liên lạc với quản lý để cập nhật thông tin. Ngoài ra, khi nhân viên xin nghỉ, phải thông báo trước cho quản lý và quản lý tiến hành xóa nhân viên. Hành động xóa hợp lệ khi nhân viên chưa thực hiện bất kỳ thao tác nào với hệ thống. Khi xóa không thành công, chứng tỏ nhân viên đã hoạt động, quản lý cần yêu cầu admin để khóa tài khoản.

+ Đối với tài khoản nhân viên trong hệ thống sẽ được tạo đồng thời với thao tác thêm nhân viên của người quản lý, mật khẩu tạo là cố định, phân quyền là nhân viên, khi nhân viên nghỉ, tài khoản cũng bị khóa theo. Thông tin tài khoản của cả nhân viên và khách hàng đều bao gồm: tên đăng nhập, mật khẩu, trạng thái khóa, quyền và các thông tin lưu trữ như token, thời gian xác nhận và hết hạn phục vụ cho việc truy cập. Mỗi một tài khoản chỉ thuộc một nhân viên hoặc một khách hàng duy nhất. Tài khoản của nhân viên sẽ có tên đăng nhập tương ứng mã nhân viên, còn tài khoản của khách hàng sẽ có tên đăng nhập tùy ý người dùng đặt. Khi đăng ký tài khoản khách hàng cần cung cấp tối thiểu là địa chỉ và email, email để phục vụ cho việc gửi mã xác nhận hoặc cấp lại mật khẩu khi quên mật khẩu.

+ Toàn bộ tài khoản của hệ thống đều do nhân viên phụ trách vai trò admin quản trị. Chức vụ này cho phép reset mật khẩu và khóa tài khoản. Reset mật khẩu sẽ gửi mật khẩu ngẫu nhiên vào email của chủ tài khoản. Với việc khóa tài khoản, khi tài khoản bị khóa phải thông báo cho người dùng cố gắng đăng nhập khi đăng nhập (trong cả trường hợp nhân viên xin nghỉ).

+ Bên cạnh đó, chức vụ admin có thể nhận yêu cầu từ bộ phận cấp cao hơn của công ty hoặc chủ để phân quyền lại cho các tài khoản nhân viên gồm các chức vụ: nhân viên, quản lý và admin. Ghi chú: Admin không thể tự thao tác với tài khoản bản thân để nhằm đảm bảo trong hệ thống luôn luôn có một tài khoản admin đang hoạt động. Trong trường hợp admin nghỉ việc, admin phải phân quyền cho nhân viên khác là admin, và từ tài khoản admin mới phân quyền đó để xác nhận khóa tài khoản admin.

+ Khách hàng cần có các thông tin cơ bản sau: họ, tên, giới tính, ngày sinh, địa chỉ, số điện thoại, email và cản cước (nếu có). Sau khi có tài khoản, khách hàng sử dụng để đăng nhập hệ thống. Người dùng có thể tra cứu các thú cưng và sản phẩm của các chi nhánh. Lựa chọn để thêm giỏ hàng. Sau cùng người dùng chọn thanh toán, cung cấp địa chỉ, phương thức thanh toán, xác nhận đơn đặt. Khi lựa chọn phương thức thanh toán online và thanh toán thành công hoặc khách hàng

cửa hàng nhận được phần tiền, hệ thống lập hóa đơn. Thông tin đơn đặt gồm: số hiệu đơn, ngày lập, địa chỉ đặt, số điện thoại, chi nhánh, phương thức thanh toán, ghi nhận mã của khách hàng và cả chi tiết mua hàng. Hóa đơn gồm: Số hiệu hóa đơn, ngày lập, tổng giá trị của hóa đơn.

+ Nhân viên khi dùng app đăng nhập để lập đơn nhập hàng, ghi nhận thú cưng hoặc sản phẩm nhập vào của chi nhánh. Đơn nhập bao gồm: Mã đơn nhập, ngày lập, mã nhân viên nhập, mã chi nhánh và các chi tiết nhập hàng. Thủ cưng nhập về có thông tin cơ bản gồm: mã thú cưng, tên, trạng thái bán, chủ của thú cưng, mô tả, giá nhập, mã chi nhánh, giống. Về giống, một loại thú cưng có nhiều giống và một giống chỉ thuộc một loại, một giống có thể có nhiều thú cưng và thú cưng chỉ thuộc một giống. Sản phẩm nhập gồm các thông tin lưu trữ: mã sản phẩm tự sinh, tên, số lượng tồn, đơn giá nhập. Một loại sản phẩm gồm nhiều sản phẩm và sản phẩm chỉ thuộc một loại duy nhất.

+ Quản lý ngoài quản trị nhân viên, còn có thể quản trị thông tin thú cưng, sản phẩm, các đối tượng liên quan. Quản lý cập nhật giá bán cho thú cưng và sản phẩm. Quản lý có thể quản trị các bảng giá của từng chi nhánh. Bảng giá có chức năng thay đổi giá cả của thú cưng hoặc sản phẩm theo thời gian được cài đặt (thời gian bắt đầu và thời gian kết thúc), sau khoảng thời gian đó các thú cưng và sản phẩm trở về giá hiện tại (giá bình thường). Bảng giá có thông tin: mã bảng, thời gian bắt đầu, kết thúc và nội dung. Bảng giá được tạo phục vụ cho các đợt khuyến mãi của các chi nhánh cửa hàng. Quản lý có thể xem các hóa đơn, đơn nhập để thống kê, báo cáo doanh thu.

+ Ở các đối tượng nhân viên, khách hàng, thú cưng, sản phẩm đều có thể upload hình ảnh, avatar đại diện để nhằm làm trực quan hệ thống online.

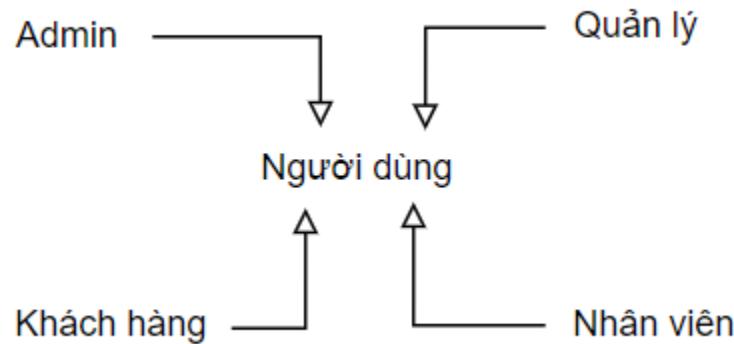
Chương II: Phân tích và thiết kế

1. Bảng thuật ngữ

STT	Tiếng Anh	Tiếng Việt	Nội dung
1	Customer	Khách hàng	Là người dùng đăng nhập phần mềm hệ thống để mua sản phẩm và thú cưng.
2	Staff	Nhân viên	Là người đang làm việc tại một chi nhánh của cửa hàng bán thú cưng. Nhân viên có nhiệm vụ nhập hàng.
3	Manager	Quản lý	Là chủ hoặc nhân viên cấp cao của công ty, nhân viên được phân bổ quản trị khu vực của hệ thống gồm nhiều chi nhánh, có trách nhiệm quản trị của các chi nhánh được giao. Quản lý có quyền quản trị các đối tượng đang tồn tại trong hệ thống ngoại trừ tài khoản.
4	Admin	Người quản trị hệ thống	Là người được tín nhiệm để giao cho vai trò quản lý mọi tài khoản trong hệ thống. Admin có quyền thay đổi phân quyền các tài khoản nhân viên. Đôi khi người chủ vừa là admin vừa là quản lý.
5	User	Người dùng	Là người dùng phần mềm hệ thống để thực hiện các tác vụ chuyên biệt.
6	Cart	Giỏ hàng	Nơi chứa các sản phẩm mà người dùng dự định mua trên hệ thống.
7	Payment System	Hệ thống thanh toán	Là hệ thống bên thứ 3 liên quan về vấn đề thanh toán tiền điện tử được tích vào phần mềm nhằm phục vụ thanh toán online cho khách hàng.

2. Xác định tác nhân

- Người dùng: đăng nhập, đăng xuất, đổi mật khẩu, quên mật khẩu, xem và thay đổi thông tin cho phép.
- Admin: quản trị tài khoản (reset, khóa tài khoản, phân quyền).
- Quản lý: quản trị thú cưng, quản trị sản phẩm, quản trị loại sản phẩm, loại thú cưng, giống, quản trị bảng giá, quản trị các đơn và hóa đơn, thống kê doanh thu.
- Nhân viên: nhập hàng (lập đơn nhập hàng), lập hóa đơn.
- Khách hàng: xem và trả cứu sản phẩm/ thú cưng, thêm vào giỏ hàng, thanh toán đơn hàng, xem các đơn hàng/ hóa đơn.



3. Yêu cầu chức năng

- + Đăng nhập: Hệ thống cho phép tài khoản tồn tại trong database đăng nhập hệ thống, dựa theo quyền của tài khoản để phân trang chức năng phù hợp, không hiển thị sai chức năng. Ngoài ra khi người dùng chọn ghi nhớ, hệ thống sẽ lưu thông tin tên đăng nhập và mật khẩu cùng với quyền vào file lưu trữ và phụ vụ cho đăng nhập lần sau.
- + Đổi mật khẩu: Khi người dùng yêu cầu đổi mật khẩu cần phải cung cấp chính xác mật khẩu cũ còn lưu trong database, cung cấp mật khẩu mới và mật khẩu nhập lại của mật khẩu mới để xác nhận thay đổi. Khi thay đổi, mật khẩu mới phải có hiệu lực ngay lập tức.
- + Quên mật khẩu: Khi người dùng quên mật khẩu cần phải cung cấp lại email đã đăng ký hoặc thuộc tài khoản đã quên. Khi cung cấp, hệ thống sẽ gửi mã xác minh, người dùng cần nhập đúng để xác minh tài khoản, khi xác minh thành công, hệ thống yêu cầu nhập mật khẩu mới và nhập lại chính xác để đổi mật khẩu hiện thời.
- + Quản lý tài khoản: Hệ thống cho phép admin reset mật khẩu của người dùng, mật khẩu ngẫu nhiên mới được gửi đến email người dùng, và cho phép khóa tài khoản, tài khoản bị khóa phải được thông báo khi đăng nhập và đăng nhập phải thất bại. Về phân quyền, admin có thể phân quyền cho các tài khoản nhân viên. Tài khoản nhân viên không thể thực hiện thao tác trên chính tài khoản của bản thân để đảm bảo trong hệ thống luôn có một tài khoản admin đang hoạt động.
- + Quản lý thứ cung: Hệ thống cho phép quản lý có thể thêm/ xóa/ sửa thông tin thứ cung toàn bộ ở các chi nhánh.
- + Quản lý sản phẩm: Hệ thống cho phép quản lý có thể thêm/ xóa/ sửa thông tin sản phẩm.
- + Quản lý loại sản phẩm: Hệ thống cho phép quản lý có thể thêm/ xóa/ sửa thông tin tên loại sản phẩm.

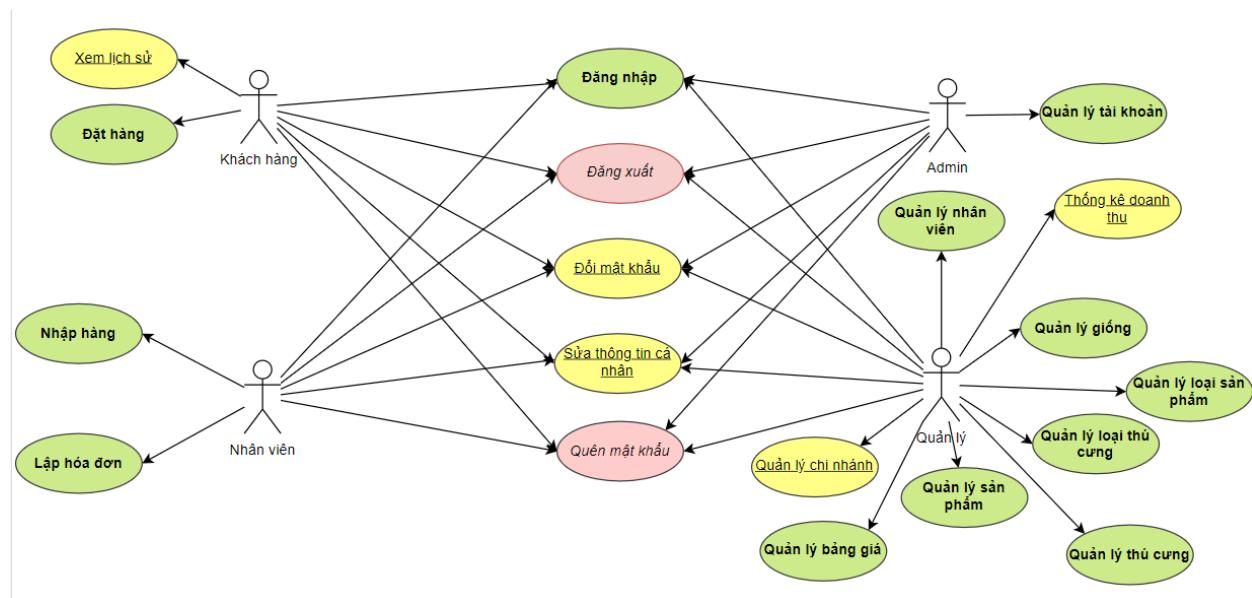
- + Quản lý loại thú cưng: Hệ thống cho phép quản lý có thể thêm/ xóa/ sửa thông tin tên loại thú cưng.
- + Quản lý giống: Hệ thống cho phép quản lý có thể thêm/ xóa/ sửa thông tin giống thú cưng.
- + Quản lý bảng giá: Hệ thống cho phép quản lý có thể thêm/ xóa/ sửa thông tin/ sửa thông tin chi tiết trong bảng giá theo từng chi nhánh.
- + Quản lý nhân viên: Hệ thống cho phép quản lý có thể thêm/ xóa (cho nghỉ)/ cập nhật thông tin nhân viên đang làm việc tại các chi nhánh.
- + Xem và thống kê doanh thu: Hệ thống cho phép quản lý xem các hóa đơn, thống kê thành doanh thu biểu đồ.
- + Nhập hàng: Hệ thống cho phép tài khoản nhân viên thực hiện việc lập hóa đơn nhập hàng, nhân viên cung cấp cho hệ thống các thông tin về sản phẩm nhập của chi nhánh trực thuộc.
- + Lập hóa đơn: Đối với lựa chọn phương thức thanh toán bằng tiền mặt của khách hàng, nhân viên lập hóa đơn khi nhận được phí thanh toán.
- + Xem và tra cứu sản phẩm/ thú cưng: Hệ thống cho phép người dùng là khách hàng xem các thông tin về sản phẩm có ở từng chi nhánh. Qua đó, người dùng thêm vào giỏ hàng hoặc bỏ ra.
- + Đặt hàng: Sau khi lập đơn đặt đã xác nhận, khách hàng tiến hành thanh toán bằng phương thức phần mềm cung cấp. Trong đó có sự tham gia của hệ thống thứ 3 bên thanh toán để xác thực và thanh toán, trả kết quả. Sau khi tiến hành tạo lập hóa đơn.
- + Xem lịch sử (Xem hóa đơn): Hệ thống cho phép khách hàng xem lại các đơn đặt/ hóa đơn mà mình đã thanh toán.

4. Yêu cầu phi chức năng

- Các thiết kế chức năng được phân bố đúng đắn, rõ ràng. Đảm bảo người không chuyên về công nghệ vẫn có thể dễ dàng sử dụng.
- Hệ thống thông báo lỗi một cách tường minh, tránh gây hiểu lầm cho người dùng.
- Hệ thống hiển thị chức năng đúng như phân quyền cấp phép cho mỗi tài khoản.
- Tốc độ truy xuất nhanh, nhỏ hơn 10 giây tính từ khi người dùng gửi yêu cầu.
- Có thể tái sử dụng mã nguồn (code) và chức năng hệ thống, tránh hiện tượng trùng lặp lượng lớn dòng code giống nhau ở nhiều nơi trong source gây ra lãng phí tài nguyên.
- Tính bảo mật: người dùng phải đăng nhập bằng mật khẩu cá nhân, và không cho phép sử dụng các chức năng không có trong phân quyền.

- Tính chịu tải: đáp ứng được lượng yêu cầu cao (bé hơn hoặc bằng 1000 máy truy cập) tại mọi thời điểm.
 - Dễ bảo trì và phát triển.

5. Thiết kế usecase



Biểu đồ usecase: Tổng hợp

Màu xanh/ in đậm: Đô ưu tiên 1

Màu vàng/ gạch dưới: Độ ưu tiên 2

Màu đỏ/ in nghiêng: Độ ưu tiên 3

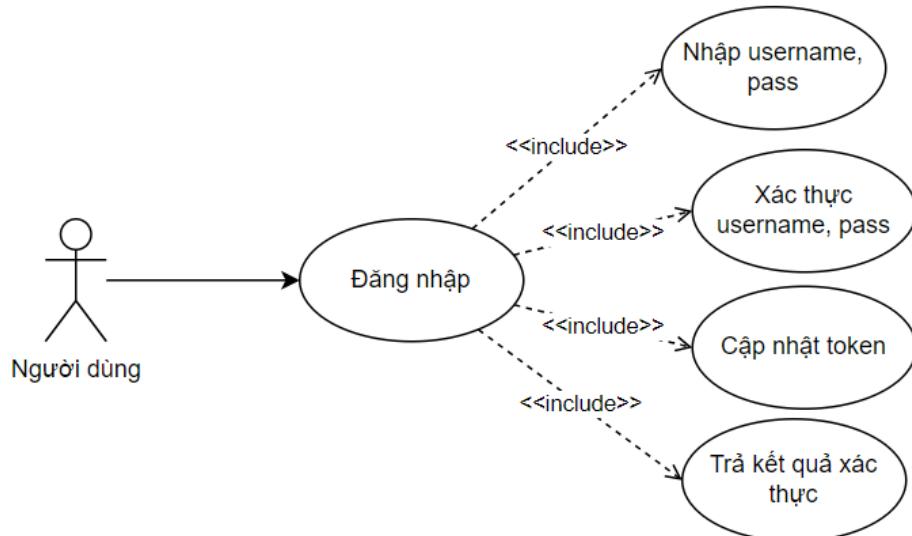
- Usecase 1:

Mô tả: Người dùng dùng tài khoản của mình để đăng nhập hệ thống theo phân quyền. Mỗi quyền sẽ được sử dụng chia sẻ khác nhau trong hệ thống.

Tên Use Case	Đăng nhập
Tác nhân chính	Người dùng
Điều kiện trước	Tài khoản phải tồn tại trong cơ sở dữ liệu
Đảm bảo tối thiểu	Cho phép đăng nhập nếu tài khoản không bị khóa
Điều kiện sau	Đăng nhập thành công

Chuỗi sự kiện chính

1. Người dùng nhập các trường username, password để đăng nhập.
2. Hệ thống xác thực username, password.
3. Báo lỗi nếu sai.
4. Hệ thống cập nhật token mới cho tài khoản.
5. Đăng nhập thành công khi không có lỗi.



Biểu đồ usecase: Đăng nhập.

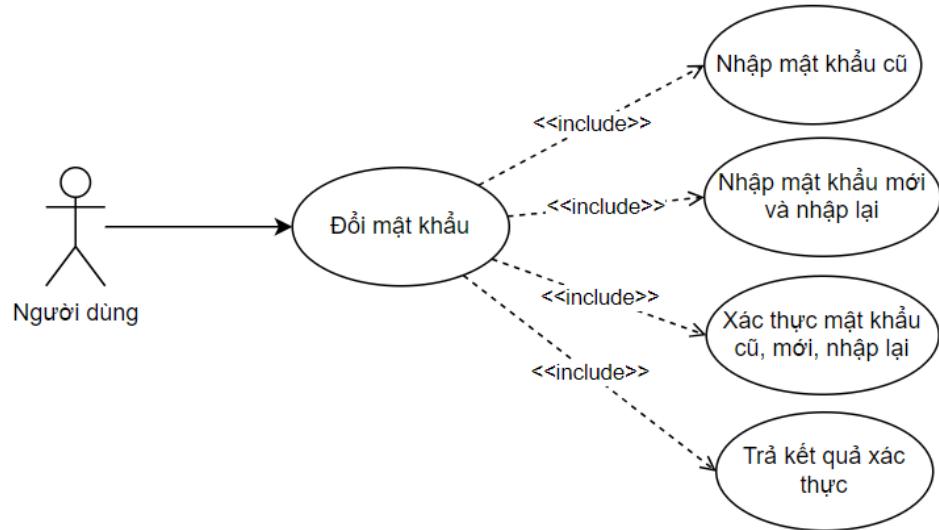
- UseCase 2:

Mô tả: Người dùng dùng tài khoản của mình để đăng nhập hệ thống theo phân quyền. Mỗi quyền sẽ được sử dụng chức năng khác nhau trong hệ thống.

Tên Use Case	Đổi mật khẩu
Tác nhân chính	Người dùng
Điều kiện trước	Tài khoản đang đăng nhập
Đảm bảo tối thiểu	Cho phép đổi khi thông tin cung cấp chính xác
Điều kiện sau	Đổi thành công

Chuỗi sự kiện chính

1. Người dùng nhập mật khẩu cũ để xác minh, mật khẩu mới và nhập lại mật khẩu mới.
2. Hệ thống xác thực mật khẩu cũ, mật khẩu mới và mật khẩu nhập lại
3. Báo lỗi nếu không thỏa mãn.
4. Thông báo cập nhật thành công khi thỏa mãn.

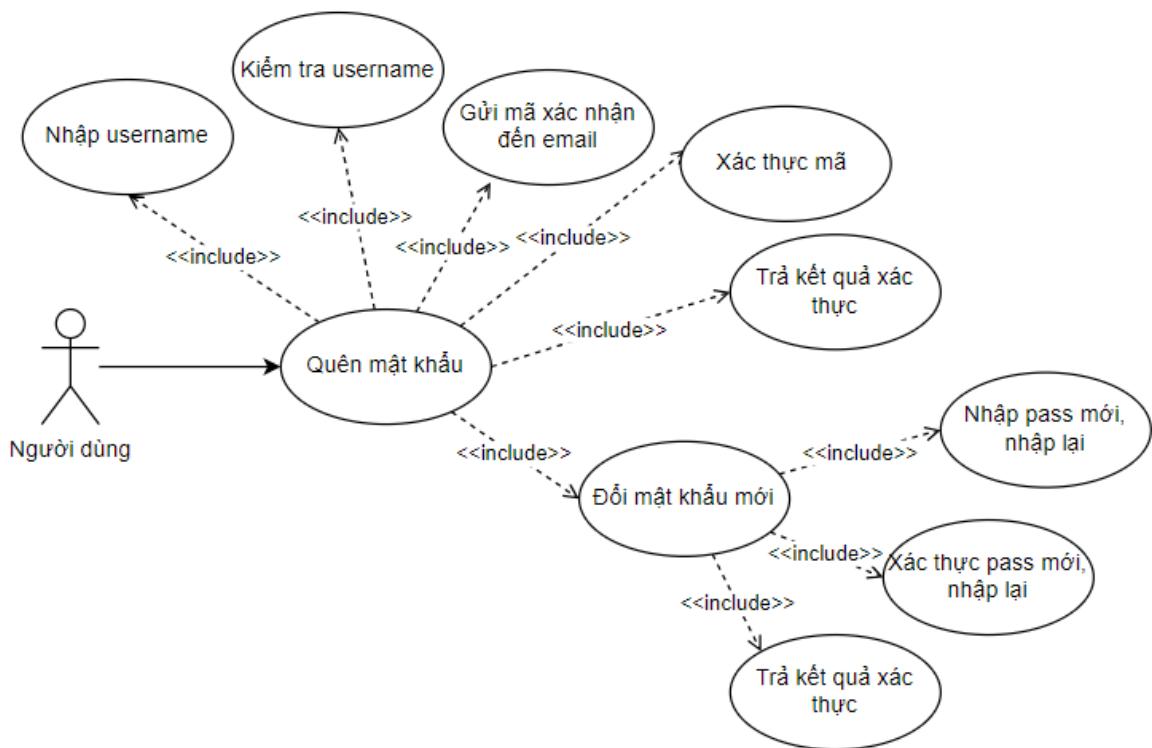


Biểu đồ usecase: Đổi mật khẩu.

- UseCase 3:

Mô tả: Người dùng cung cấp username để hệ thống xác thực, sau khi xác thực thành công, người dùng đổi mật khẩu mới.

Tên Use Case	Quên mật khẩu
Tác nhân chính	Người dùng
Điều kiện trước	Tài khoản chưa đăng nhập
Đảm bảo tối thiểu	Tài khoản phải tồn tại trong cơ sở dữ liệu và phải có thông tin email
Điều kiện sau	Đổi thành công
Chuỗi sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng cung cấp username 2. Hệ thống xác thực sự tồn tại của tài khoản, và gửi mã xác thực đến email 3. Báo lỗi nếu không thỏa mãn. 4. Khi xác thực mã xác thực thành công, hệ thống cho phép người dùng cập nhật lại mật khẩu mới. 5. Cập nhật lại mật khẩu mới khi mật khẩu mới và nhập lại trùng khớp, hoặc báo lỗi khi không khớp



Biểu đồ usecase: Quên mật khẩu.

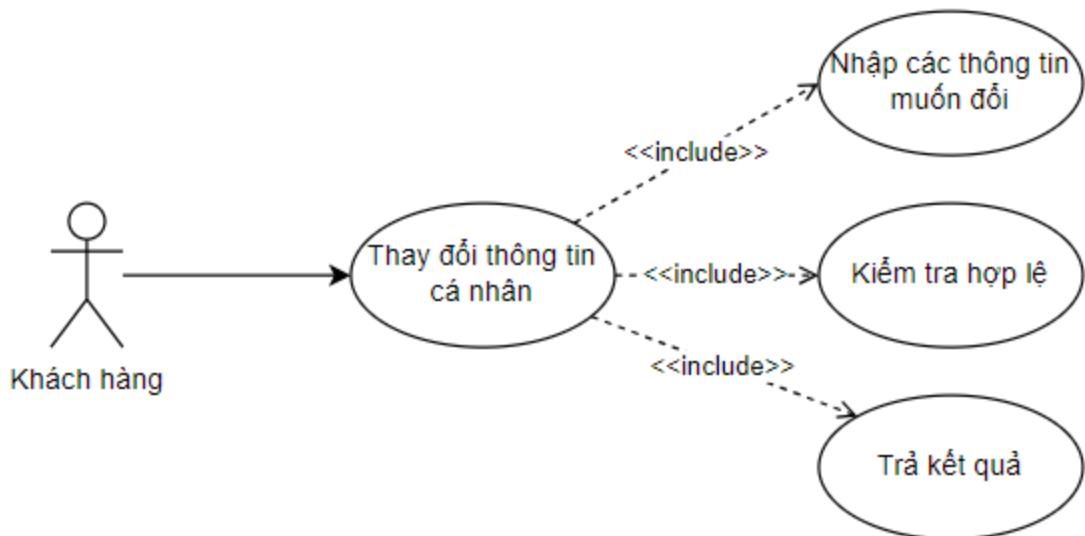
- Use Case 4:

Mô tả: Người dùng (chỉ Khách hàng) sau khi đăng nhập có thể xem và thay đổi những thông tin cá nhân cho phép sửa.

Tên Use Case	Thay đổi thông tin cá nhân
Tác nhân chính	Khách hàng
Điều kiện trước	Tài khoản khách hàng đang đăng nhập
Đảm bảo tối thiểu	Cho phép đổi các trường nếu nhập đúng dạng chuẩn.
Điều kiện sau	Đổi thành công

Chuỗi sự kiện chính

1. Khách hàng sau khi đăng nhập yêu cầu xem thông tin cá nhân.
2. Hệ thống tìm thông tin dữ liệu cá nhân của tài khoản liên quan để hiển thị.
3. Người dùng có thể yêu cầu thay đổi các thông tin cho phép.
4. Hệ thống mở khóa các trường cho phép nhập.
5. Người dùng nhập các trường muốn đổi.
6. Hệ thống thông báo đổi thành công và cập nhật vào cơ sở dữ liệu nếu nhập đúng và thông báo lỗi nếu nhập sai.



Biểu đồ usecase: Thay đổi thông tin cá nhân.

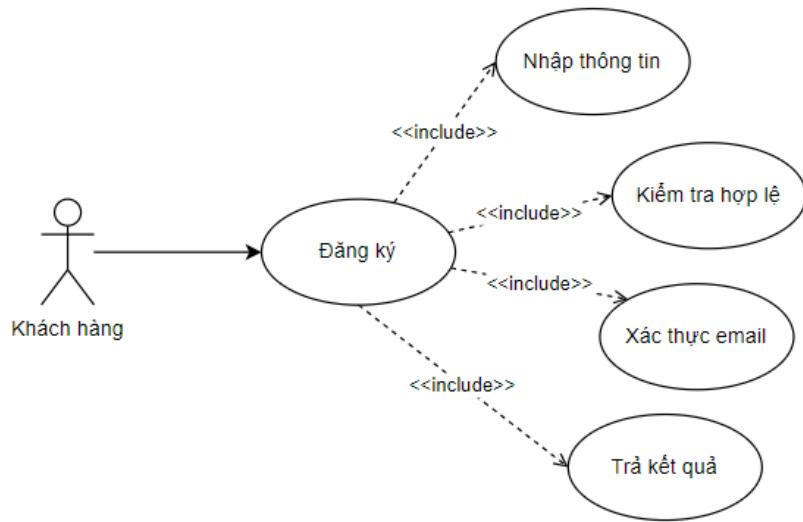
- Usecase 5:

Mô tả: Người dùng (chỉ Khách hàng) có thể đăng ký nhiều tài khoản. Một tài khoản tương ứng một mã khách hàng trong hệ thống.

Tên Use Case	Đăng ký
Tác nhân chính	Khách hàng
Điều kiện trước	Tài khoản chưa đăng nhập
Đảm bảo tối thiểu	Hệ thống rollback lại nếu lỗi giữa chừng
Điều kiện sau	Khách hàng đăng ký thành công

Chuỗi sự kiện chính

1. Khách hàng chọn chức năng đăng ký tài khoản nếu chưa có tài khoản.
2. Khách hàng nhập các thông tin mà hệ thống yêu cầu.
3. Hệ thống thực hiện việc xác thực thông tin của người dùng.
4. Hệ thống thông báo lỗi khi không thỏa mãn.
5. Hệ thống gửi email mã xác thực đến người dùng đăng ký.
6. Khách hàng cung cấp mã xác thực.
7. Hệ thống xác thực và thông báo đăng ký thành công.



Biểu đồ usecase: Đăng ký.

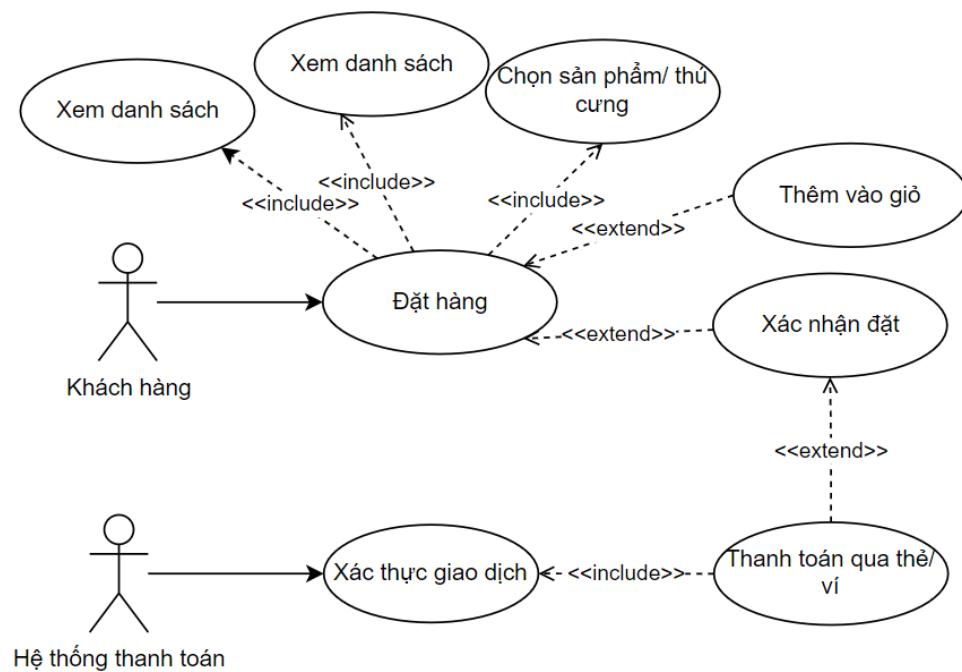
- Usecase 6:

Mô tả: Khách hàng xem và tìm kiếm sản phẩm, thú cưng muốn mua và có thể thêm vào giỏ hàng sau khi đăng nhập. Sau đó chọn các sản phẩm muốn đặt trong giỏ hàng, cung cấp địa chỉ giao hàng và thanh toán. Với thanh toán, hệ thống thanh toán sẽ xác thực, và xử lý thanh toán tiền điện tử do khách hàng chi trả, sử dụng hệ thống thanh toán bên thứ 3.

Tên Use Case	Đặt hàng
Tác nhân chính	Khách hàng, hệ thống thanh toán
Điều kiện trước	Khách hàng đã đăng nhập
Đảm bảo tối thiểu	Hệ thống rollback lại nếu đặt hàng lỗi giữa chừng
Điều kiện sau	Khách hàng đặt hàng thành công

Chuỗi sự kiện chính

- Khách hàng chọn chi nhánh để tra cứu các vật phẩm trong chi nhánh, thêm các sản phẩm, thú cưng vừa xem hoặc tìm kiếm vào giỏ hàng.
- Khách hàng chọn các sản phẩm, thú cưng muốn đặt trong giỏ hàng rồi tiến hành chọn địa chỉ giao hàng và chọn thanh toán, chi nhánh.
- Nếu người mua chọn thanh toán online thì yêu cầu người mua cung cấp phương thức thanh toán qua thẻ hoặc ví, sau đó hệ thống sẽ kiểm tra thông tin thanh toán online, nếu đúng thì tiến hành quá trình thanh toán, nếu sai thì thông báo lại cho người dùng.
- Thông báo cho người dùng việc đặt hàng thành công hoặc không thành công.

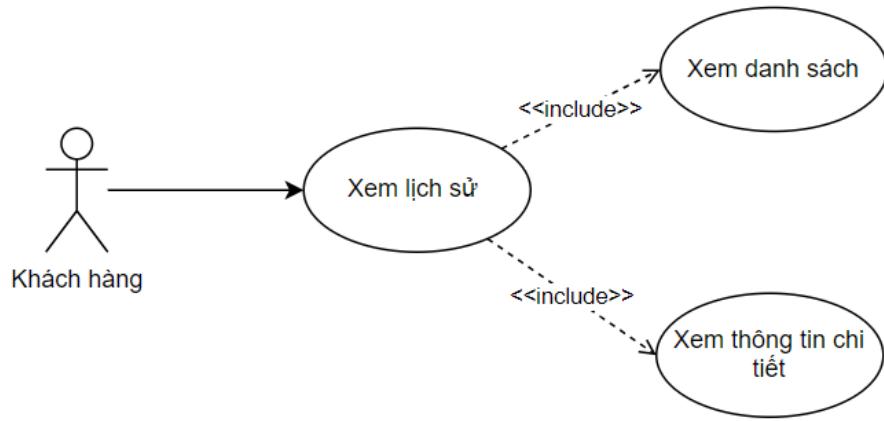


Biểu đồ usecase: Đặt hàng.

- UseCase 7:

Mô tả: Khách hàng xem các hóa đơn đã thanh toán.

Tên Use Case	Xem lịch sử (Xem hóa đơn)
Tác nhân chính	Khách hàng
Điều kiện trước	Khách hàng đã đăng nhập
Đảm bảo tối thiểu	Khách hàng phải có hóa đơn
Điều kiện sau	Khách hàng xem được danh sách
Chuỗi sự kiện chính	<ol style="list-style-type: none"> 1. Hệ thống hiển thị các hóa đơn của khách hàng. 2. Khách hàng xem thông tin hóa đơn của bản thân.

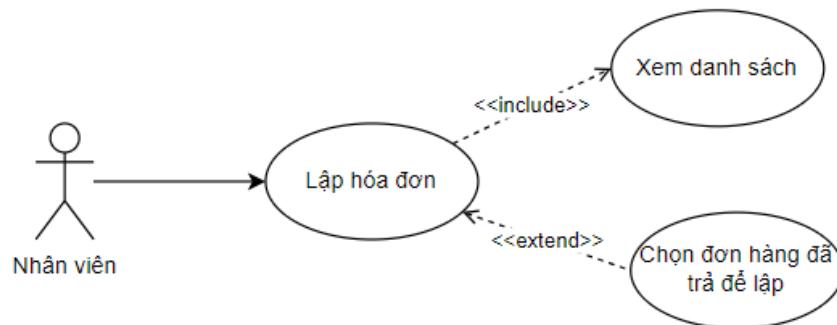


Biểu đồ usecase: Xem lịch sử hóa đơn.

- UseCase 8:

Mô tả: Nhân viên cập nhật trạng thái đơn hàng đã thanh toán khi nhận phí thanh toán tiền mặt của khách hàng. Hệ thống ghi nhận và lập hóa đơn.

Tên Use Case	Lập hóa đơn
Tác nhân chính	Nhân viên
Điều kiện trước	Nhân viên đã đăng nhập
Đảm bảo tối thiểu	Có hóa đơn chưa thanh toán
Điều kiện sau	Nhân viên cập nhật thành công
Chuỗi sự kiện chính	<ol style="list-style-type: none"> 1. Nhân viên duyệt đơn chọn chức năng đơn hàng trên giao diện chính sau khi đăng nhập thành công. 2. Hệ thống hiển thị danh sách đơn đặt hàng. 3. Nhân viên duyệt những đơn hàng chưa thanh toán để xác nhận thanh toán. 4. Hệ thống thông báo trả kết quả cập nhật.

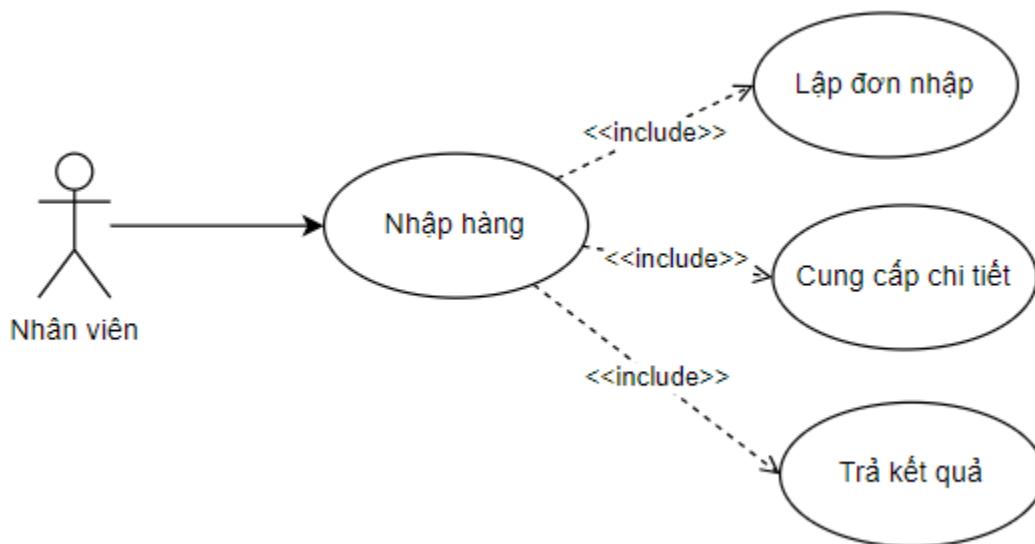


Biểu đồ usecase: Lập hóa đơn.

- Usecase 9:

Mô tả: Khi cần nhập hàng, nhân viên sẽ liên lạc với nhà cung cấp để nhập. Nhà cung cấp sẽ nhận và báo giá nhân viên. Phòng kế toán chi trả. Khi nhận hàng, nhân viên phải lập lại đơn nhập gồm các thông tin về sản phẩm hoặc thứ cung nhập.

Tên Use Case	Nhập hàng
Tác nhân chính	Nhân viên
Điều kiện trước	Nhân viên đã đăng nhập
Đảm bảo tối thiểu	Hệ thống rollback lại nếu lỗi giữa chừng
Điều kiện sau	Nhân viên nhập thành công
Chuỗi sự kiện chính	<ol style="list-style-type: none"> 1. Nhân viên nhận hàng và lập đơn tương ứng trên hệ thống. 2. Hệ thống ghi nhận các thông tin cung cấp và khởi tạo đơn nhập. 3. Hệ thống ghi nhận các chi tiết của đơn nhập. 4. Trả kết quả thông báo cho nhân viên.



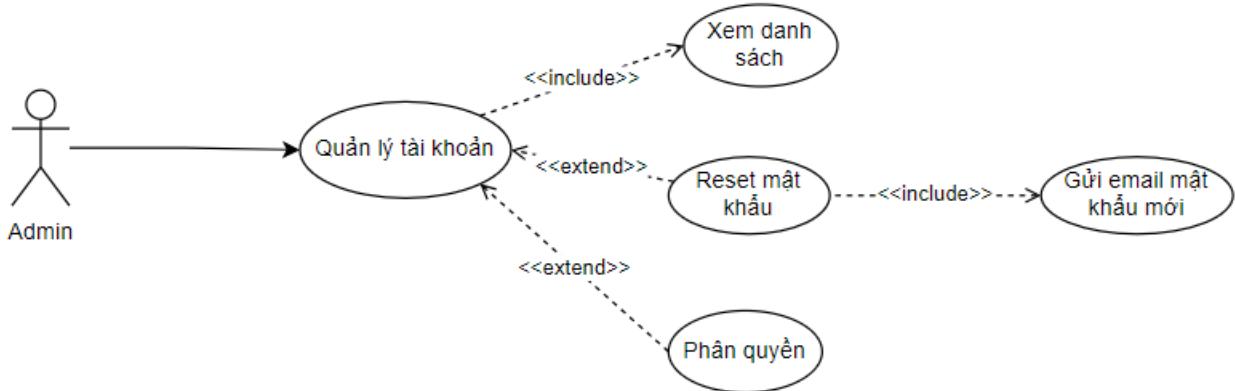
Biểu đồ usecase: Nhập hàng.

- Usecase 10:

Mô tả: Admin có thể reset mật khẩu và khóa các tài khoản trong hệ thống khi được yêu cầu. Ngoài ra admin có thể nhận yêu cầu của nhân viên cấp cao để thay đổi quyền tài khoản nhân viên: admin, quản lý, nhân viên.

Tên Use Case	Quản lý tài khoản
--------------	-------------------

Tác nhân chính	Admin
Điều kiện trước	Đăng nhập bằng tài khoản admin
Đảm bảo tối thiểu	Hệ thống rollback lại nếu lỗi, admin không thể thao tác trên chính tài khoản của mình.
Điều kiện sau	Cập nhật thành công
Chuỗi sự kiện chính	
<ul style="list-style-type: none"> - Reset mật khẩu: <ol style="list-style-type: none"> 1. Admin yêu cầu reset tài khoản chỉ định. 2. Hệ thống tạo mã ngẫu nhiên cho mật khẩu mới. 3. Hệ thống gửi mật khẩu mới đến email của tài khoản. - Khóa tài khoản: <ol style="list-style-type: none"> 1. Admin yêu cầu khóa tài khoản chỉ định. 2. Hệ thống cập nhật trạng thái tài khoản của tài khoản chỉ định. - Phân quyền: <ol style="list-style-type: none"> 1. Admin chọn phân quyền nhân viên cho tài khoản. 2. Hệ thống cập nhật quyền mới cho tài khoản nhân viên. 	



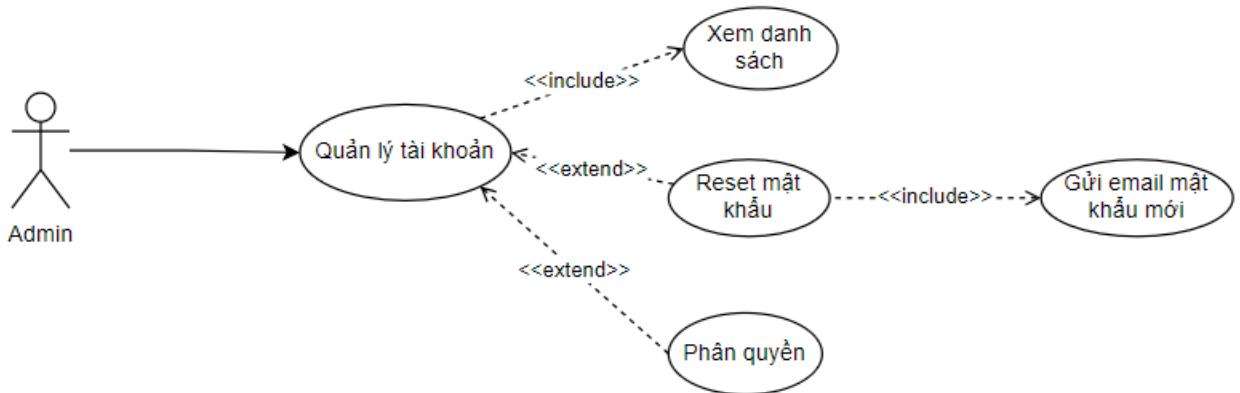
Biểu đồ usecase: Quản lý tài khoản.

- UseCase 11:

Mô tả: Admin có thể reset mật khẩu và khóa các tài khoản trong hệ thống khi được yêu cầu. Ngoài ra admin có thể nhận yêu cầu của nhân viên cấp cao để thay đổi quyền tài khoản nhân viên: admin, quản lý, nhân viên.

Tên Use Case	Quản lý tài khoản
Tác nhân chính	Admin
Điều kiện trước	Đăng nhập bằng tài khoản admin
Đảm bảo tối thiểu	Hệ thống rollback lại nếu lỗi, admin không thể thao tác trên chính tài khoản của mình.
Điều kiện sau	Cập nhật thành công
Chuỗi sự kiện chính	

- Reset mật khẩu:
 1. Admin yêu cầu reset tài khoản chỉ định.
 2. Hệ thống tạo mã ngẫu nhiên cho mật khẩu mới.
 3. Hệ thống gửi mật khẩu mới đến email của tài khoản.
- Khóa tài khoản:
 1. Admin yêu cầu khóa tài khoản chỉ định.
 2. Hệ thống cập nhật trạng thái tài khoản của tài khoản chỉ định.
- Phân quyền:
 1. Admin chọn phân quyền nhân viên cho tài khoản.
 2. Hệ thống cập nhật quyền mới cho tài khoản nhân viên.



Biểu đồ usecase: Quản lý tài khoản.

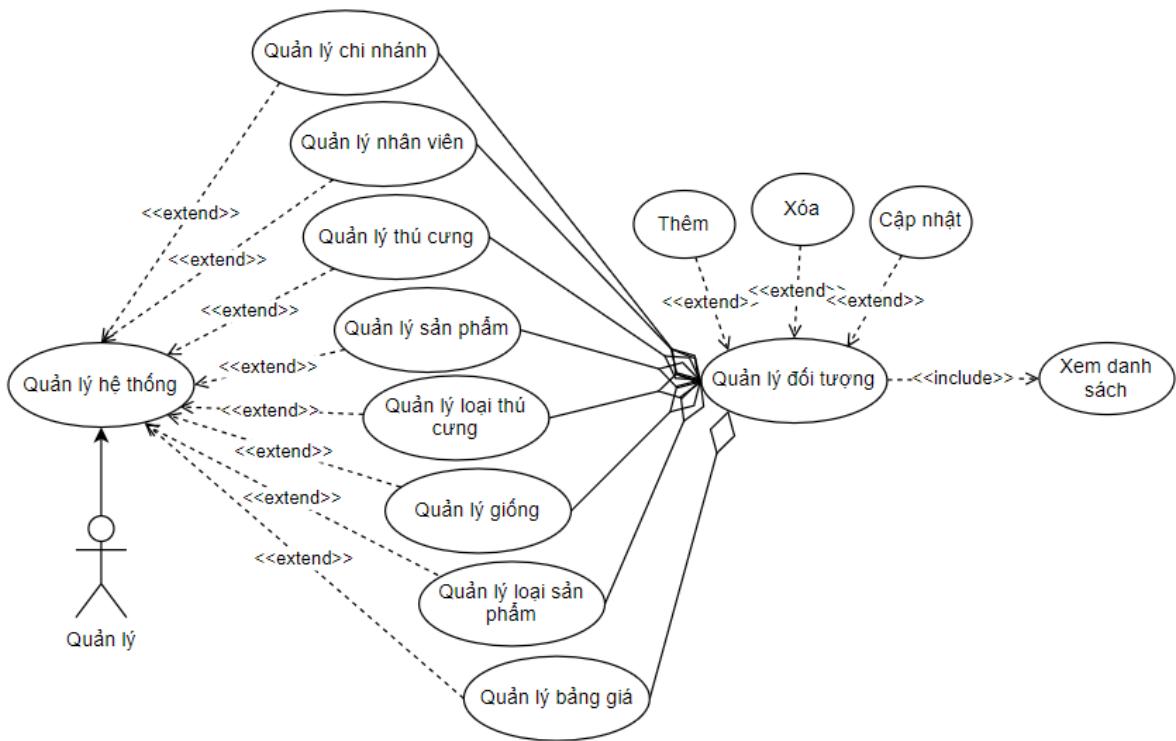
- Usecase 12:

Mô tả: Quản lý có thể quản trị các nhân viên và chi nhánh trong khu vực quản lý, sản phẩm, thú cưng, các đối tượng loại và giống, quản trị bảng giá ở các chi nhánh.

Tên Use Case	Quản lý hệ thống
Tác nhân chính	Quản lý
Điều kiện trước	Đăng nhập bằng tài khoản quản lý
Đảm bảo tối thiểu	Hệ thống rollback lại nếu lỗi
Điều kiện sau	Thao tác thành công khi input thỏa mãn

Chuỗi sự kiện chính

1. Quản lý đăng nhập hệ thống bằng tài khoản quản lý
2. Quản lý lựa chọn thao tác quản trị chi nhánh, nhân viên, sản phẩm, thú cưng, loại thú cưng, giống, loại sản phẩm hoặc bảng giá.
3. Hệ thống hiển thị danh sách tùy vào yêu cầu của quản lý.
4. Quản lý chọn thêm, xóa, hoặc cập nhật dòng thông tin đối tượng tương ứng.
5. Quản lý cung cấp thông tin với yêu cầu nhập và cập nhật, xem và xác nhận với xóa.
6. Hệ thống kiểm tra và trả kết quả thông qua thông báo.

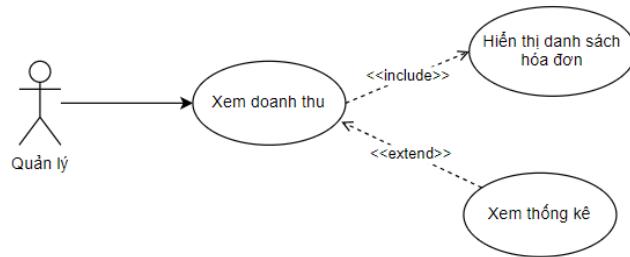


Biểu đồ usecase: Quản lý hệ thống.

- UseCase 13:

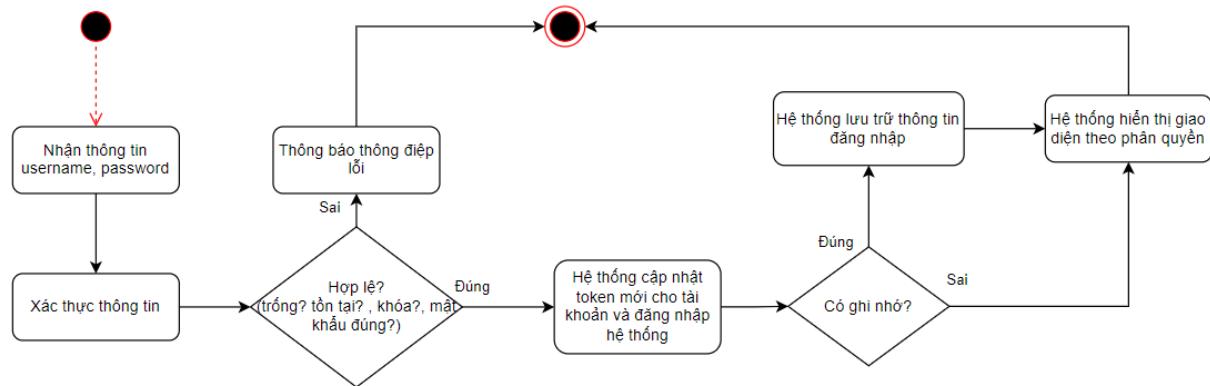
Mô tả: Quản lý có thể xem thông kê doanh thu chi nhánh theo hóa đơn tồn tại trong chi nhánh.

Tên Use Case	Thông kê doanh thu
Tác nhân chính	Quản lý
Điều kiện trước	Đăng nhập bằng tài khoản quản lý
Đảm bảo tối thiểu	Trong cơ sở dữ liệu tồn tại hóa đơn theo chi nhánh
Điều kiện sau	Hiển thị thành công nếu có tồn tại
Chuỗi sự kiện chính	<ol style="list-style-type: none"> Người dùng yêu cầu chọn chi nhánh muốn xem. Hệ thống lấy dữ liệu hóa đơn liên quan trong cơ sở dữ liệu và hiển thị. Hệ thống tính toán doanh thu và hiển thị.

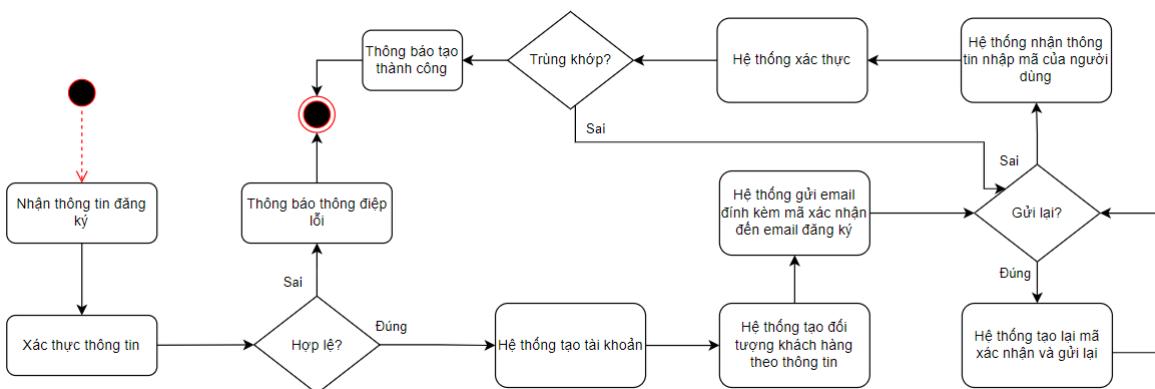


Biểu đồ usecase: Xem doanh thu.

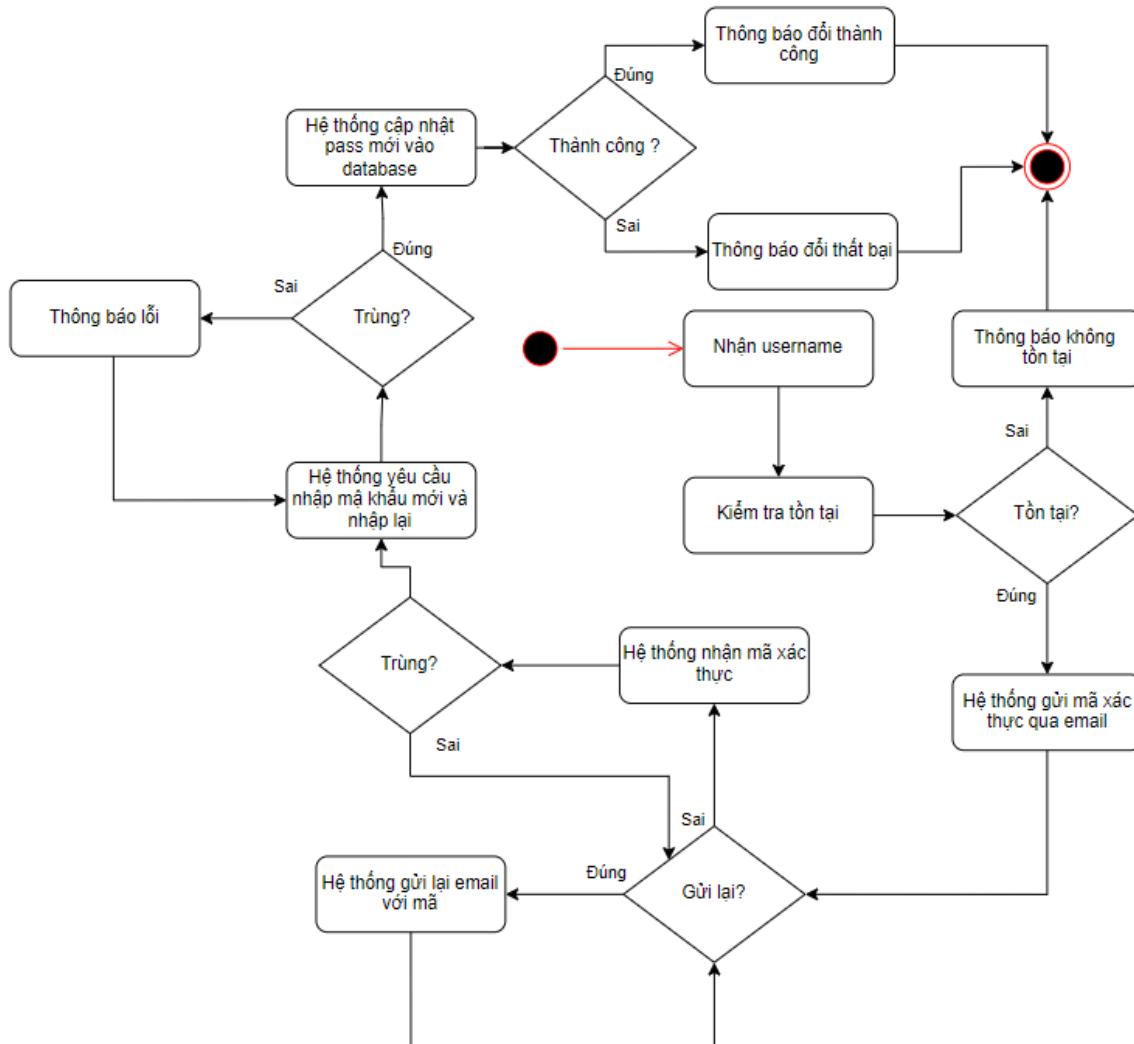
6. Thiết kế biểu đồ hoạt động



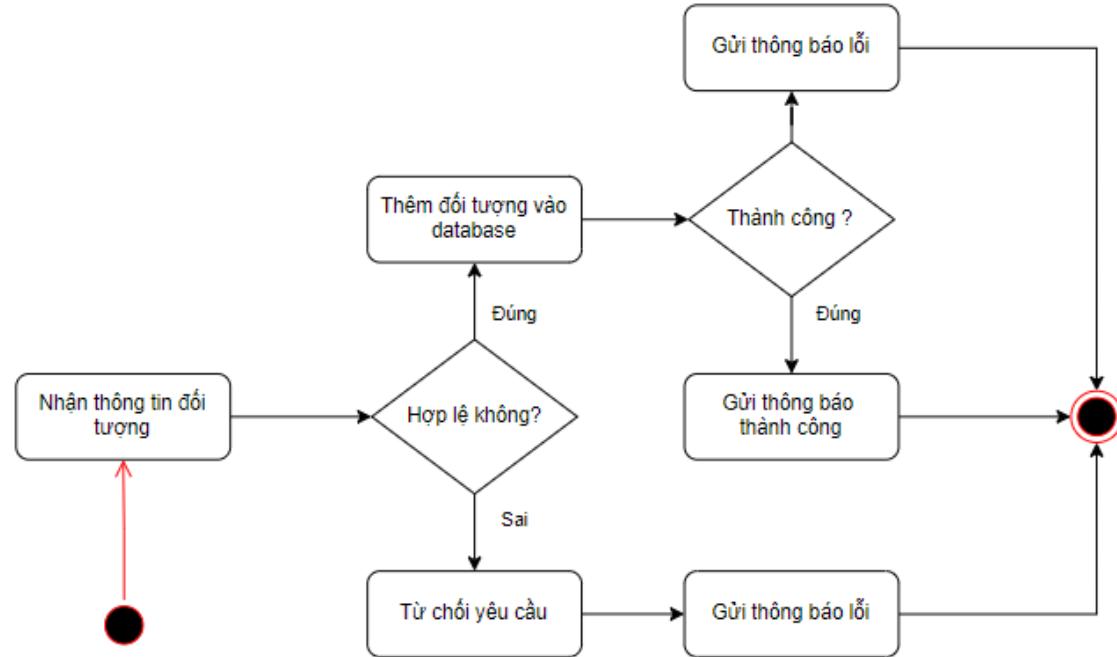
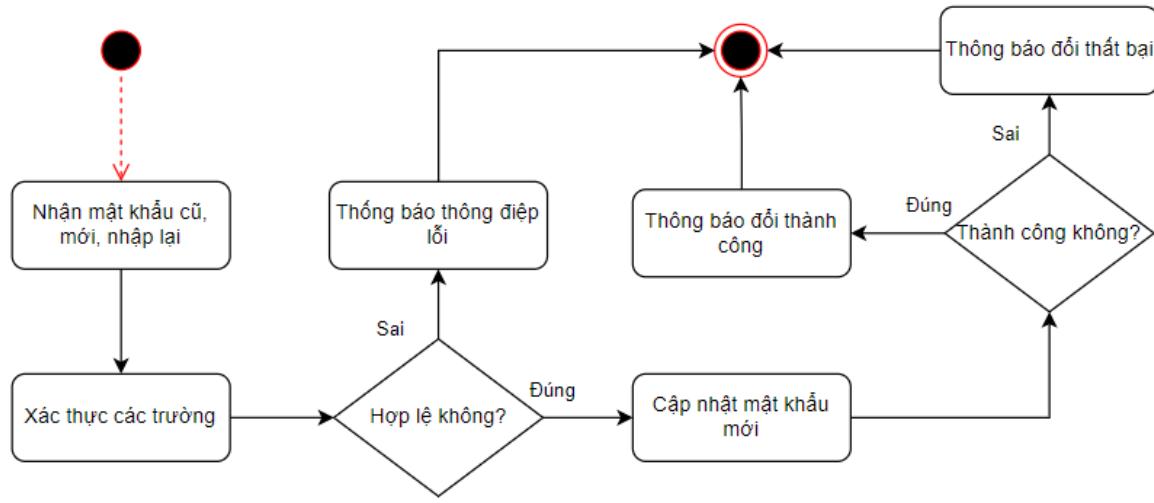
Biểu đồ hoạt động: Đăng nhập.



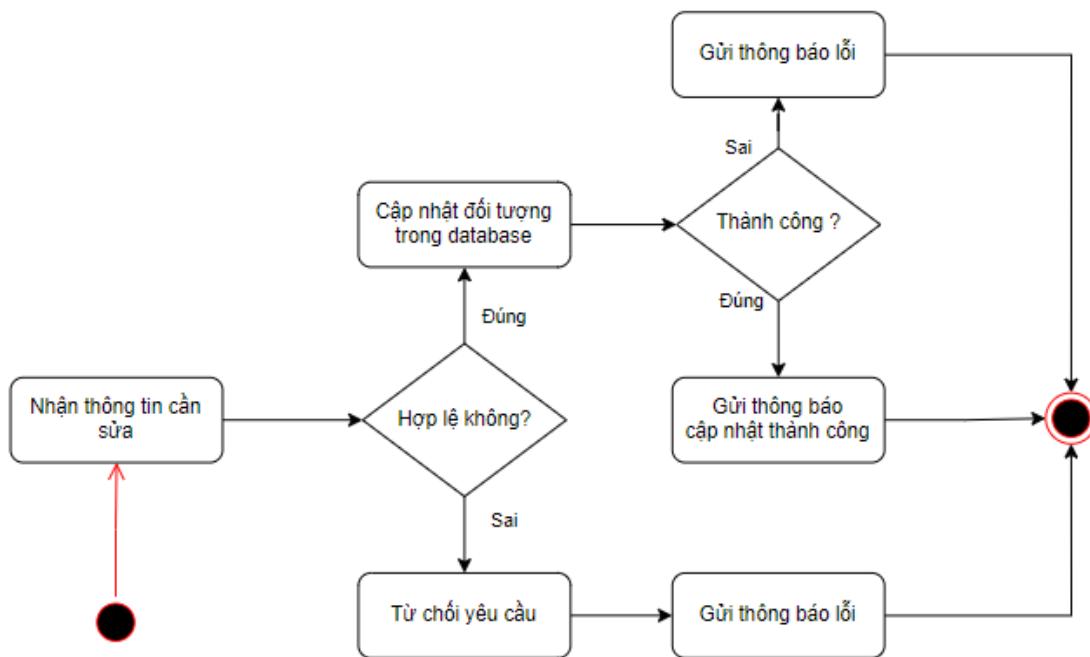
Biểu đồ hoạt động: Đăng ký.



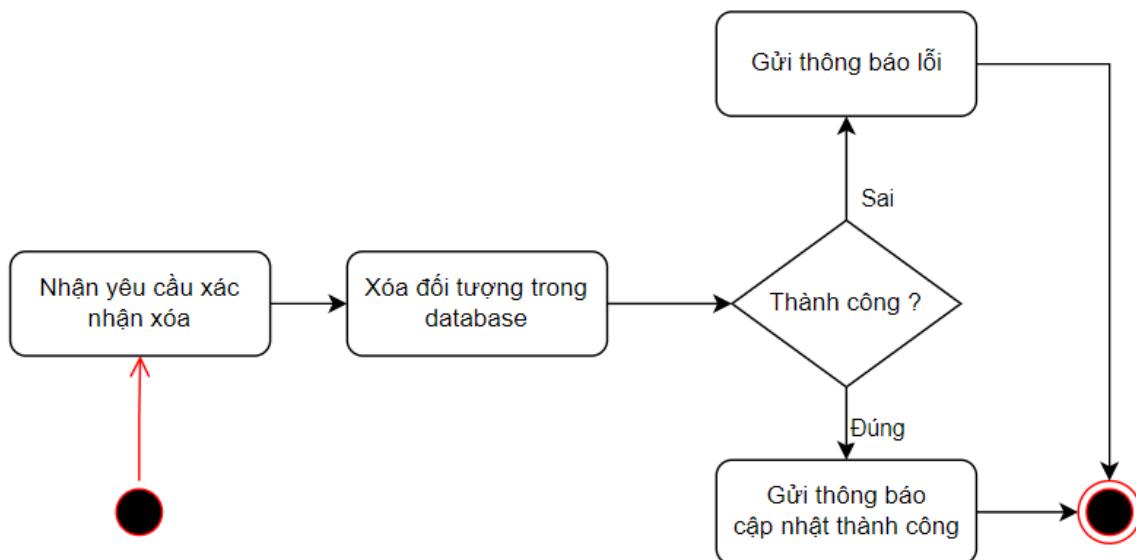
Biểu đồ hoạt động: Quên mật khẩu.



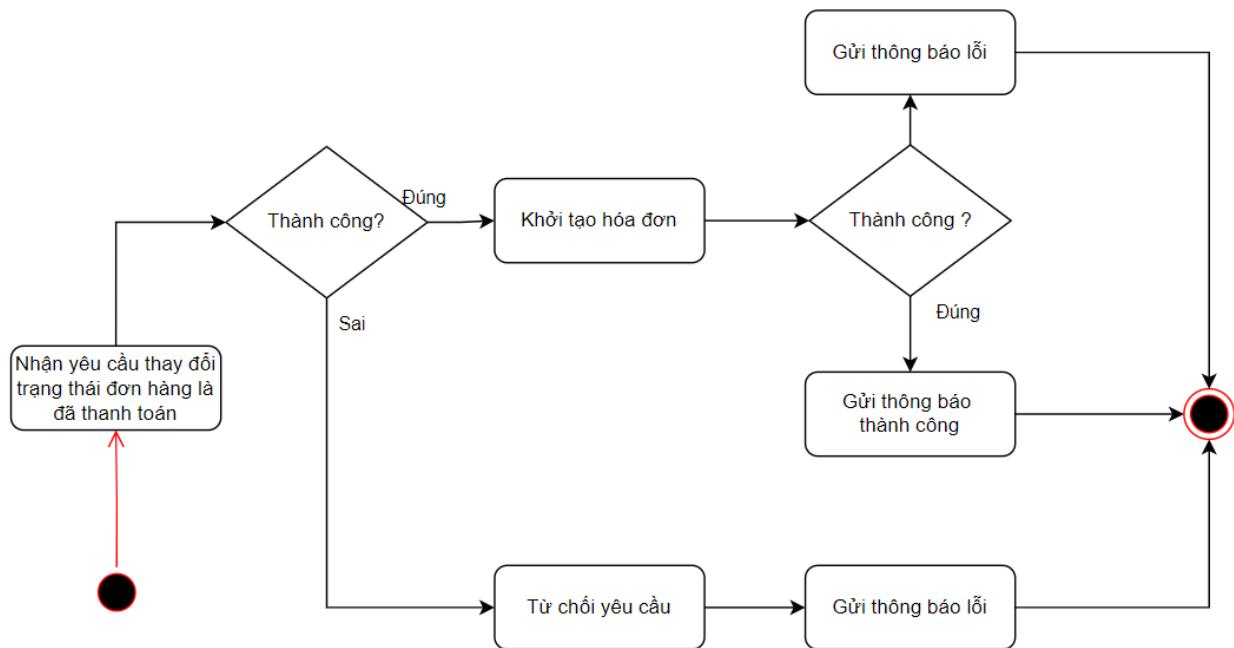
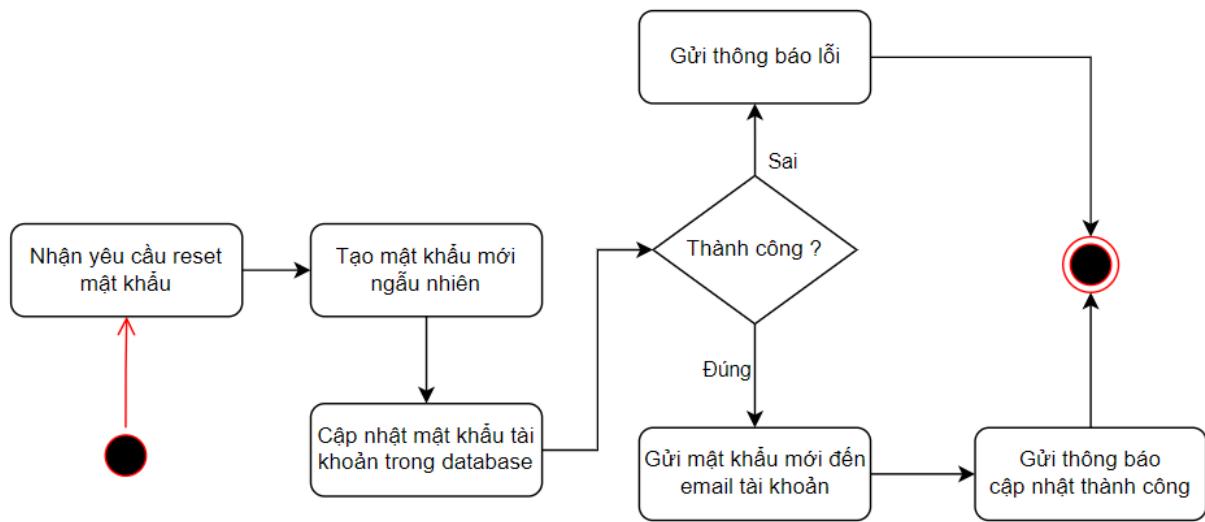
Biểu đồ hoạt động: Thêm đối tượng (nhân viên, chi nhánh, loại sản phẩm, loại thú cưng, giống, sản phẩm, thú cưng, bảng giá, phiếu nhập hàng, chi tiết bảng giá, chi tiết phiếu nhập).

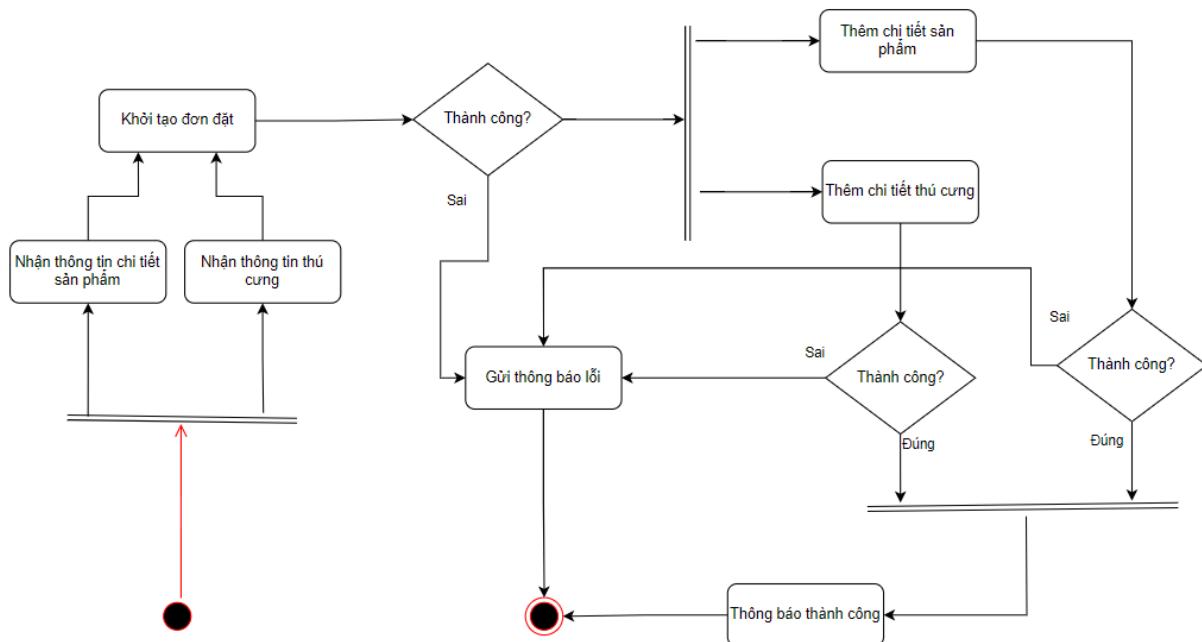


Biểu đồ hoạt động: Cập nhật thông tin đối tượng (nhân viên, chi nhánh, loại sản phẩm, loại thú cung, giống, sản phẩm, thú cung, bảng giá, giá trong chi tiết bảng giá, phân quyền, khóa tài khoản, thay đổi thông tin cá nhân).

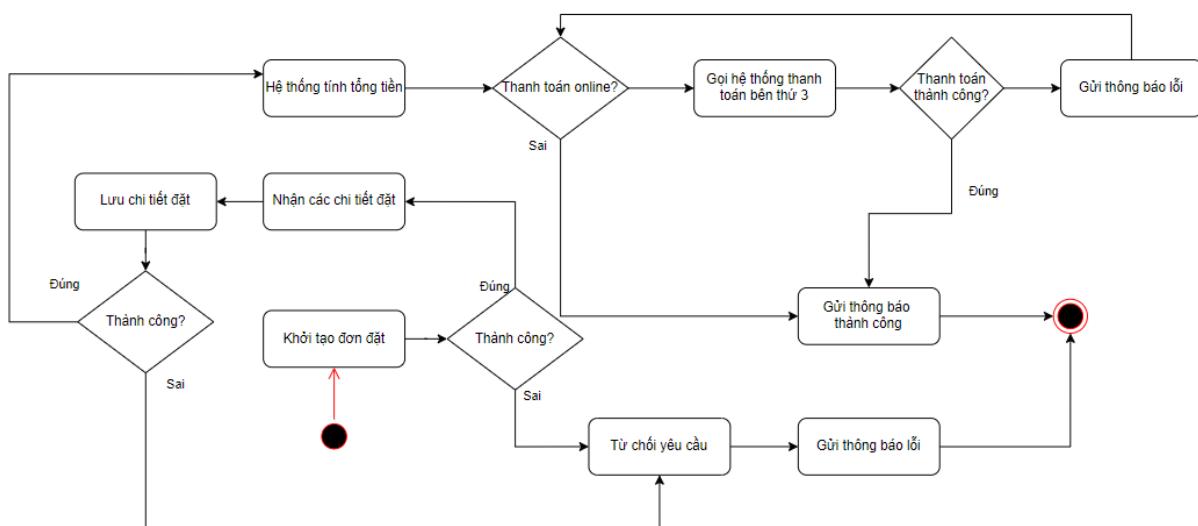


Biểu đồ hoạt động: Xóa đối tượng (nhân viên, chi nhánh, loại sản phẩm, loại thú cung, giống, sản phẩm, thú cung, bảng giá).

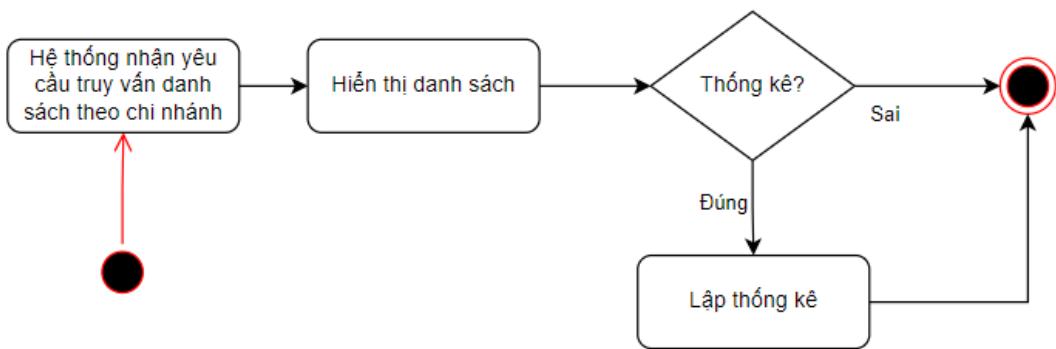




Biểu đồ hoạt động: Nhập hàng

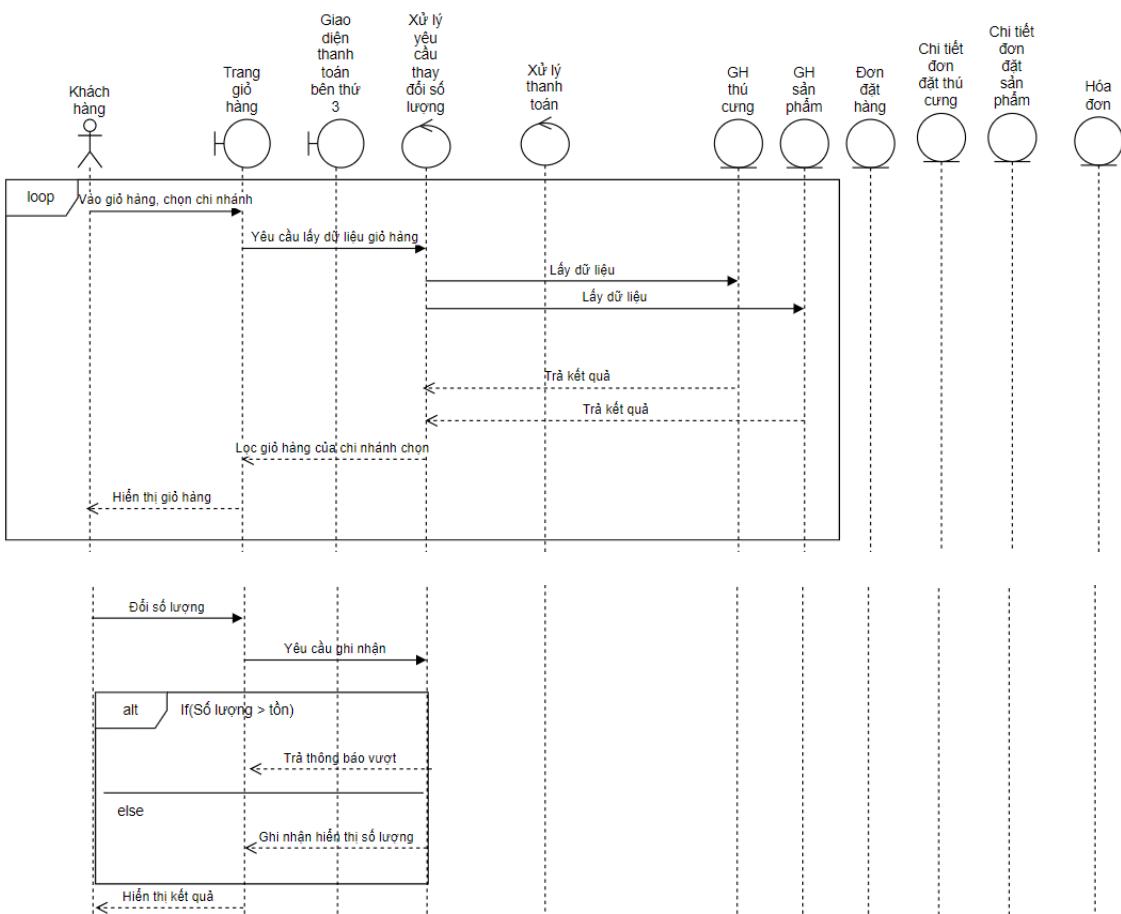


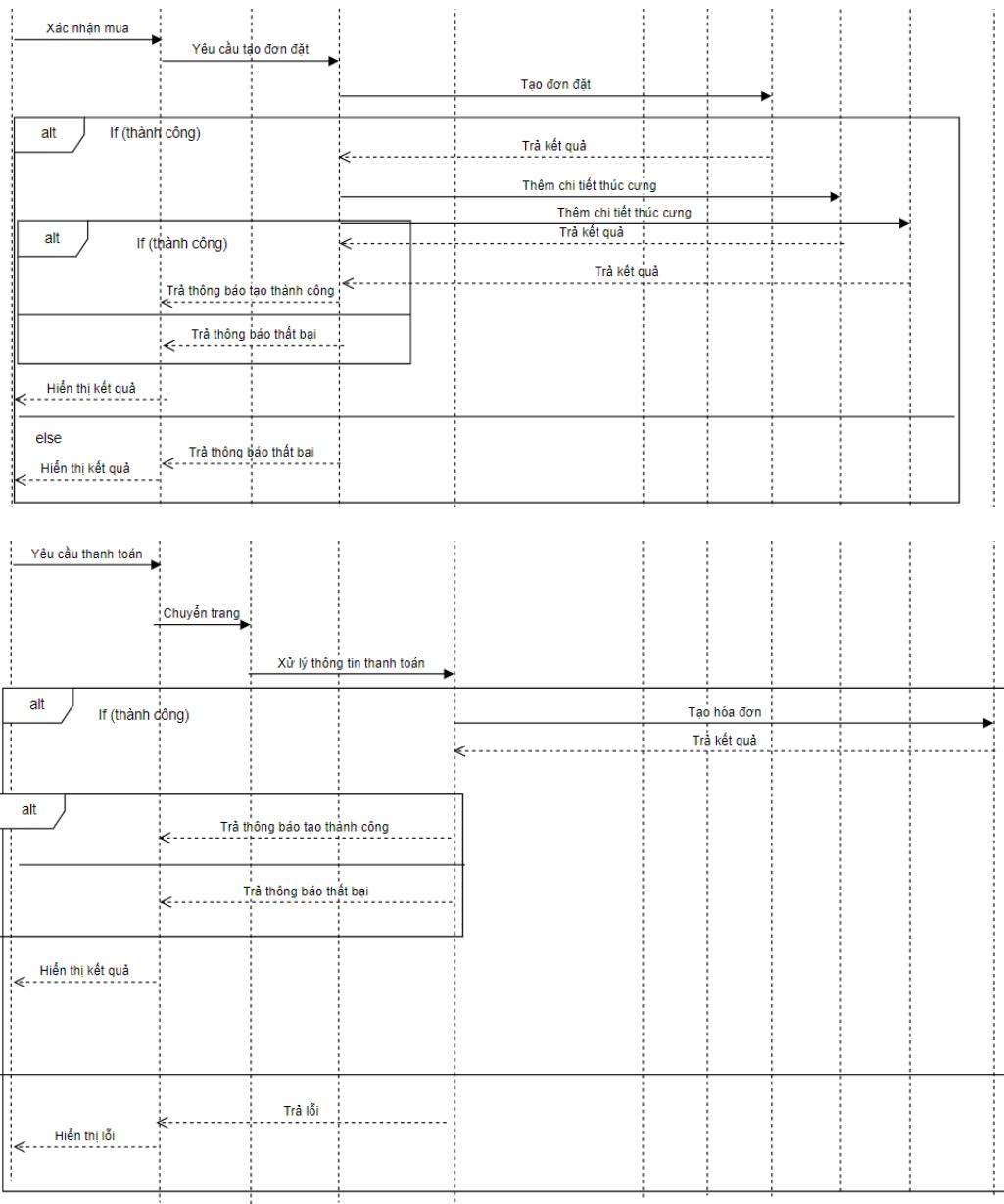
Biểu đồ hoạt động: Đặt hàng và thanh toán.



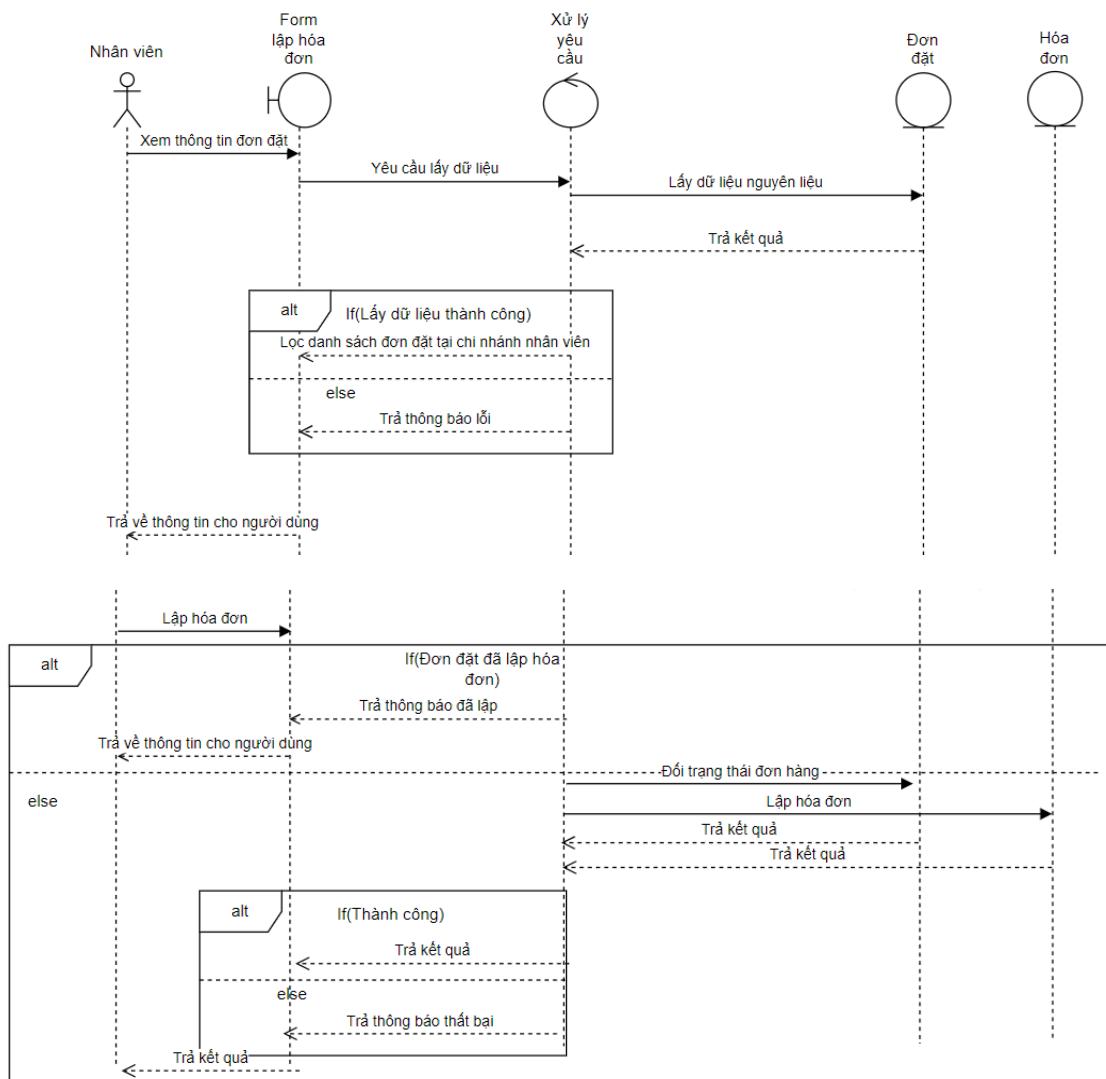
Biểu đồ hoạt động: Thông kê doanh thu.

7. Thiết kế biểu đồ tuần tự



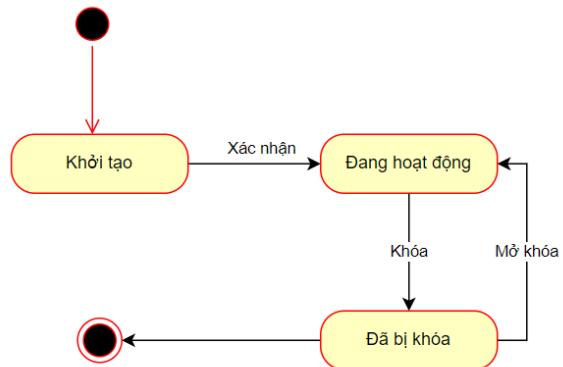


Biểu đồ tuần tự: Đặt hàng và thanh toán

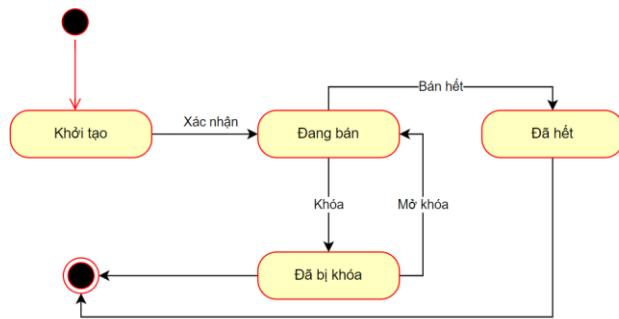


Biểu đồ hoạt động: Lập hóa đơn

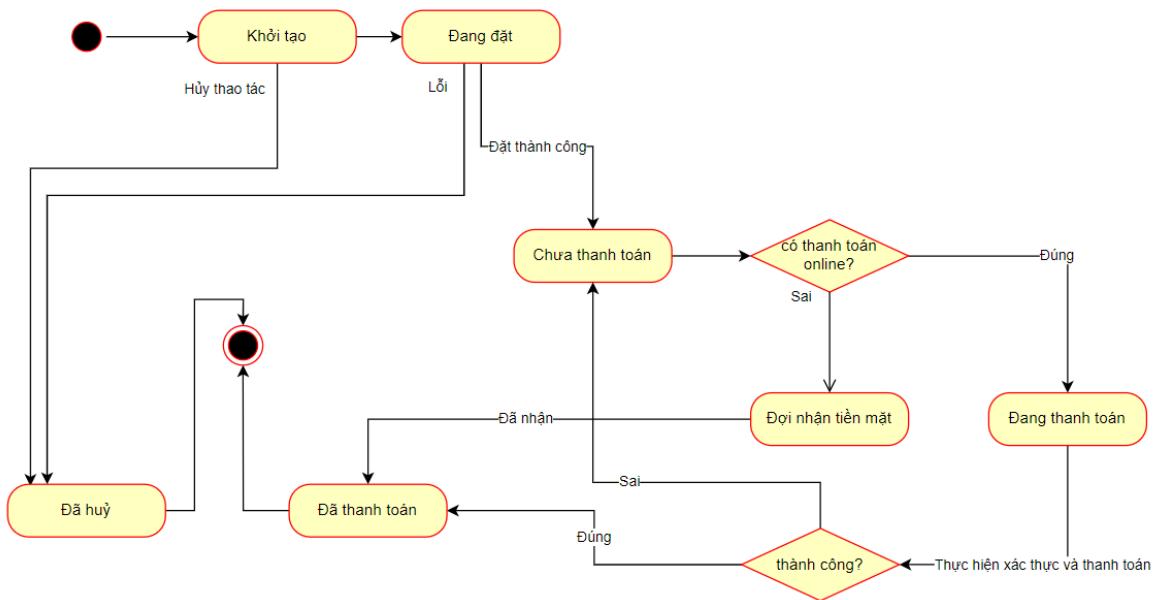
8. Thiết kế biểu đồ trạng thái



Biểu đồ trạng thái: Tài khoản

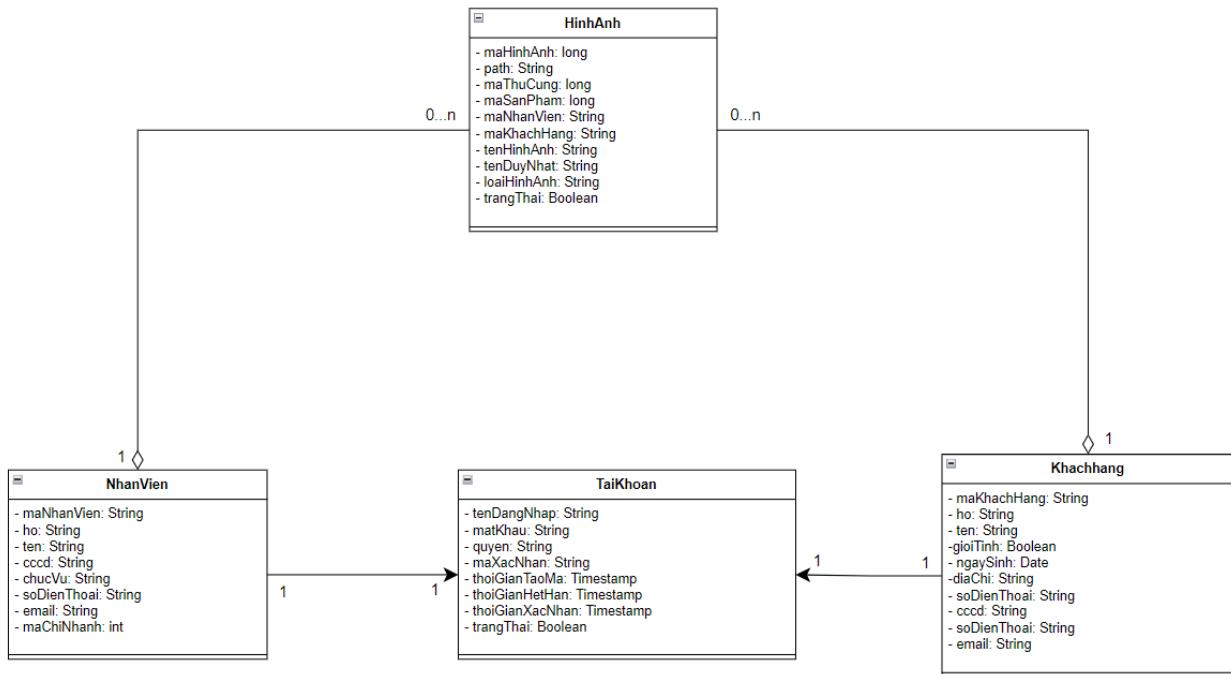


Biểu đồ trạng thái: Thú cưng

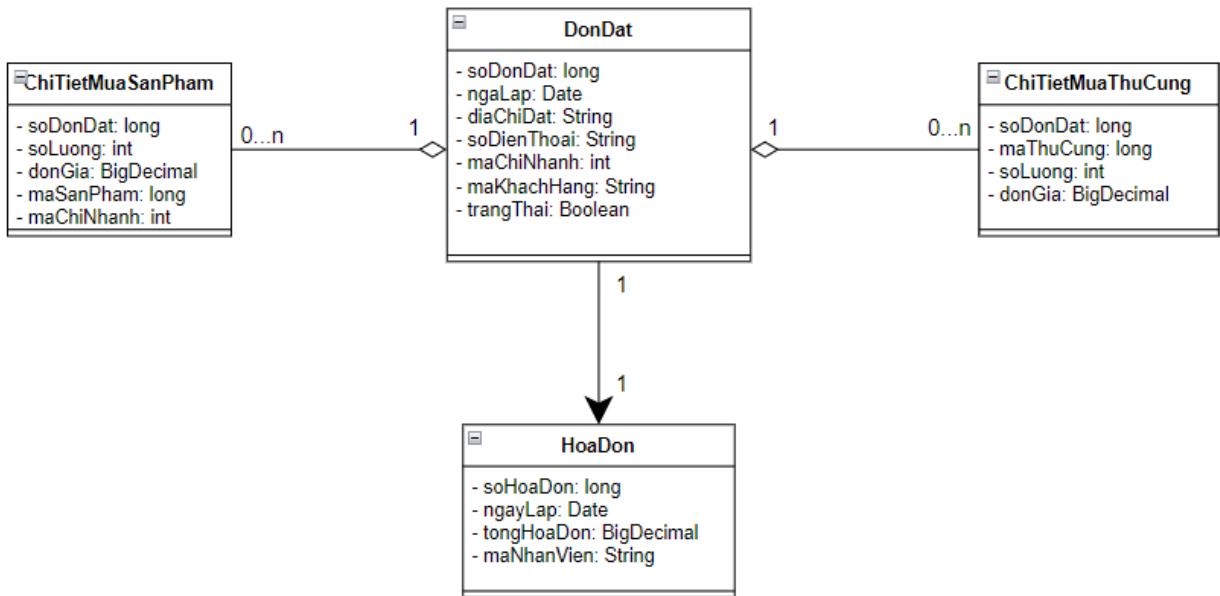


Biểu đồ trạng thái: Biểu đồ trạng thái đơn đặt hàng

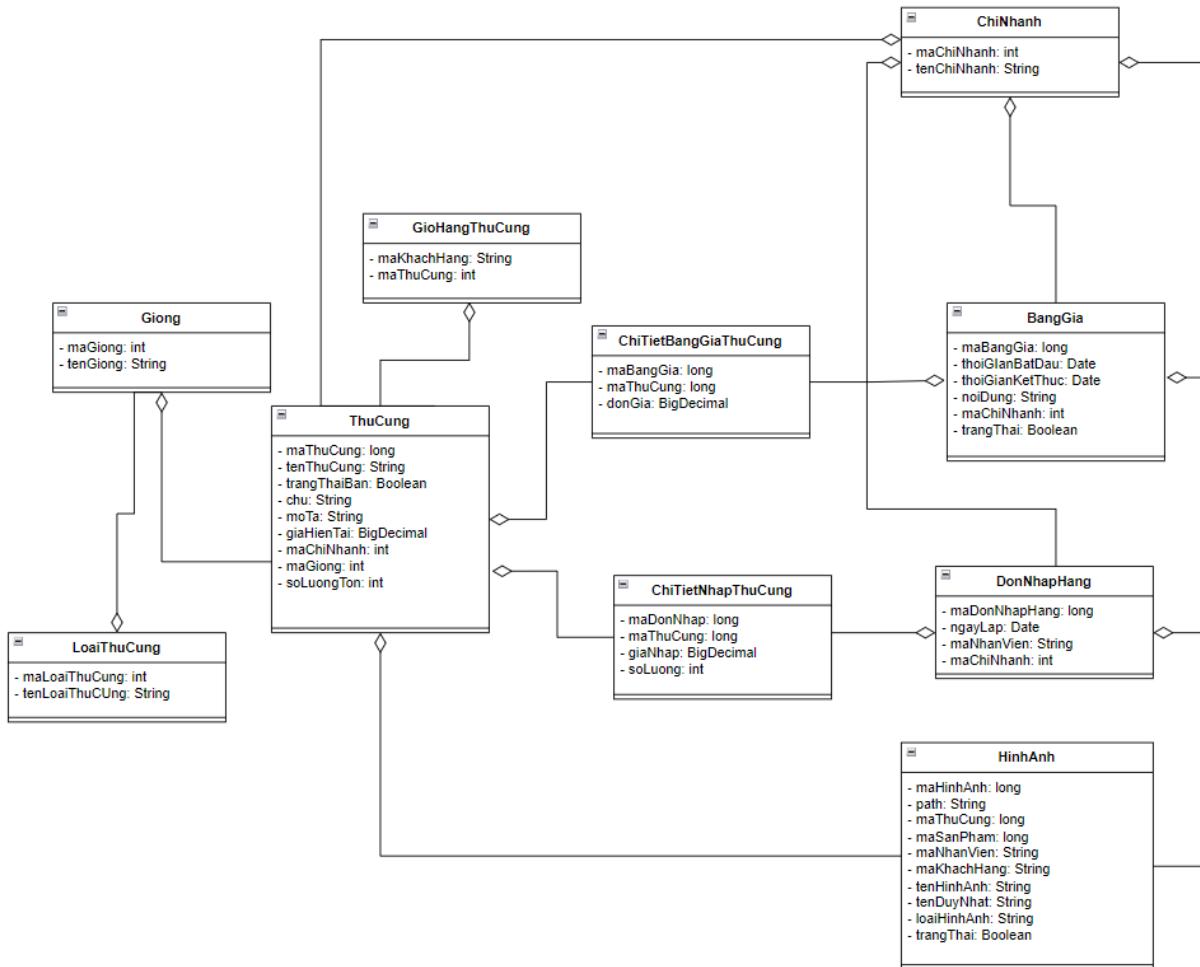
9. Thiết kế biểu đồ lớp



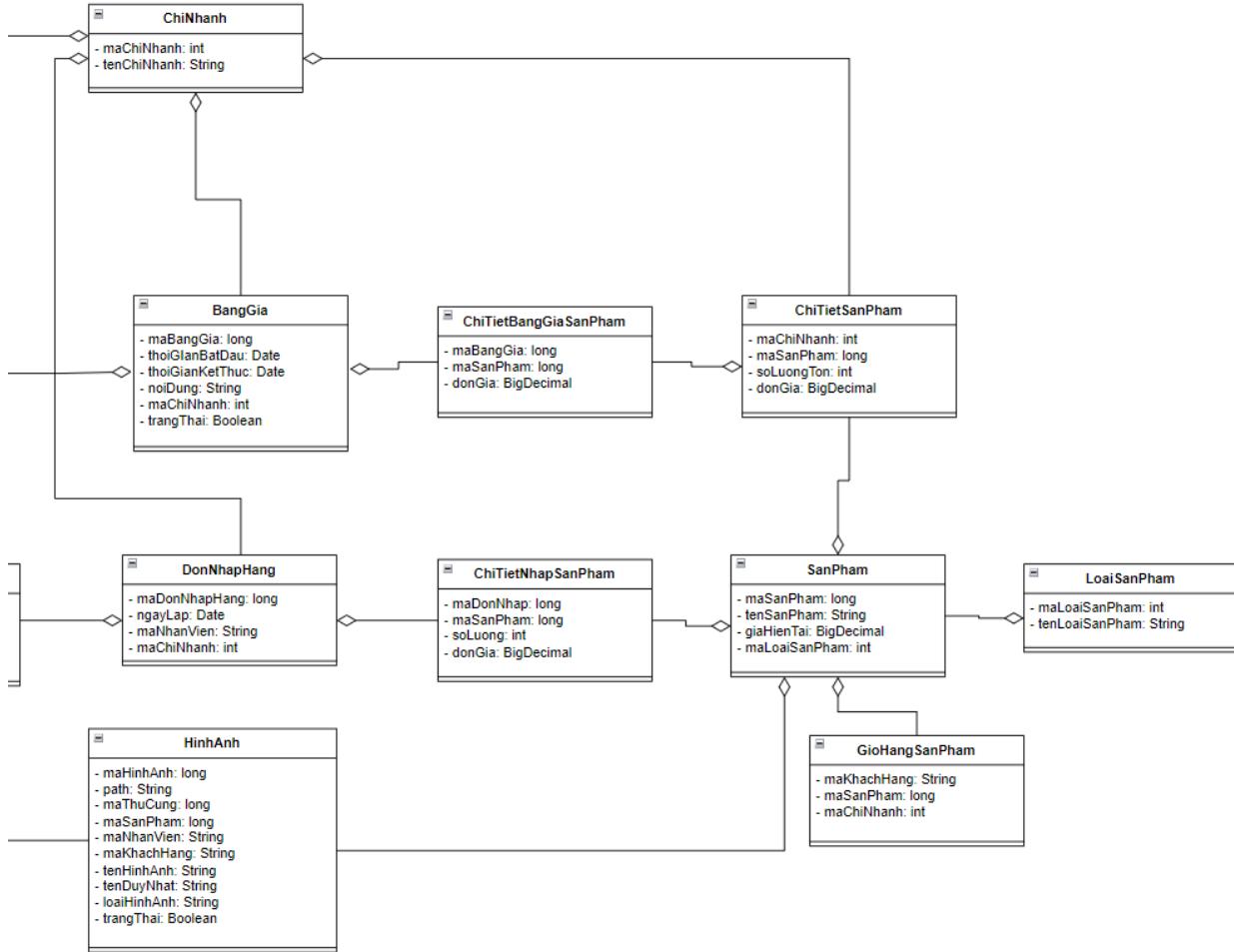
Biểu đồ lopus: Quan hệ giữa tài khoản với nhân viên, khách hàng.



Biểu đồ lopus: Quan hệ giữa các đối tượng đơn đặt.

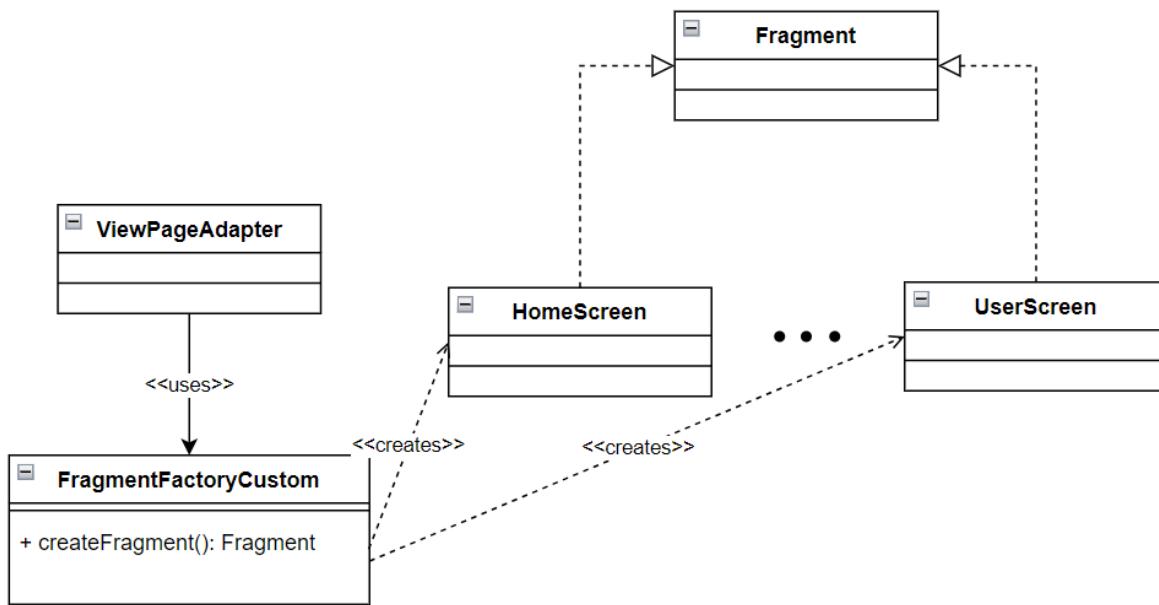


Biểu đồ lớp: Quan hệ giữa các đối tượng quản trị



Biểu đồ lớp: Quan hệ giữa các đối tượng quản trị 2

10. Thiết kế design pattern



Mẫu thiết kế: Method Factory

Mô tả: **FragmentFactoryCustom** có phương thức tạo các đối tượng fragment (là lớp cha của các trang view) dựa vào tham số tên truyền vào, mục đích nhằm thuận tiện trong việc phân quyền hiển thị tùy theo vai trò của người dùng phần mềm.

- Method factory:

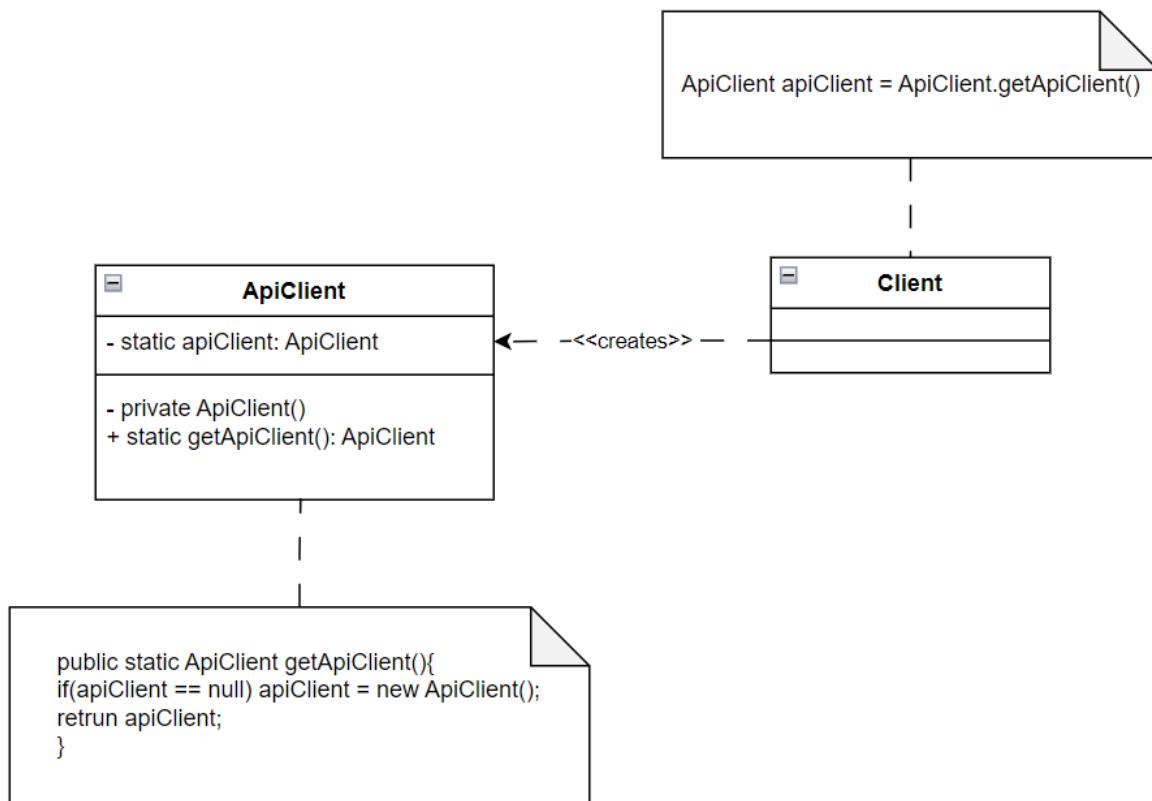
```
public Fragment createFragment(String fragmentName){  
    switch (fragmentName){  
        case "home":  
            return new HomeScreen();  
        case "cart":  
            return new CartScreen();  
        case "user":  
            return new UserScreen();  
        case "user2":  
            return new UserScreen2();  
        case "manage":  
            return new ManageScreen();  
        case "manage_account":  
            return new ManageUserScreen();  
        case "manage_employee":  
            return new ManageEmployeeScreen();  
        case "product_import":  
            return new ImportProductScreen();  
        case "manage_finance":  
            return new ManageFinanceScreen();  
        default:  
            return new Fragment()  
    }  
}
```

- Ứng dụng của mẫu vào ViewPagerAdapter:

```
public ViewPagerAdapter(@NonNull FragmentManager fm, int behavior, List<String> fragmentNameList) {
    super(fm, behavior);
    this.fragmentNameList=fragmentNameList;
    fragmentFactory=new FragmentFactoryCustom();
}

@NonNull
@Override
public Fragment getItem(int position) {
    return
fragmentFactory.createFragment(fragmentNameList.get(position));//Trả về fragment screen
}

@Override
public int getCount() {
    return fragmentNameList.size(); // trả về số item
}
```

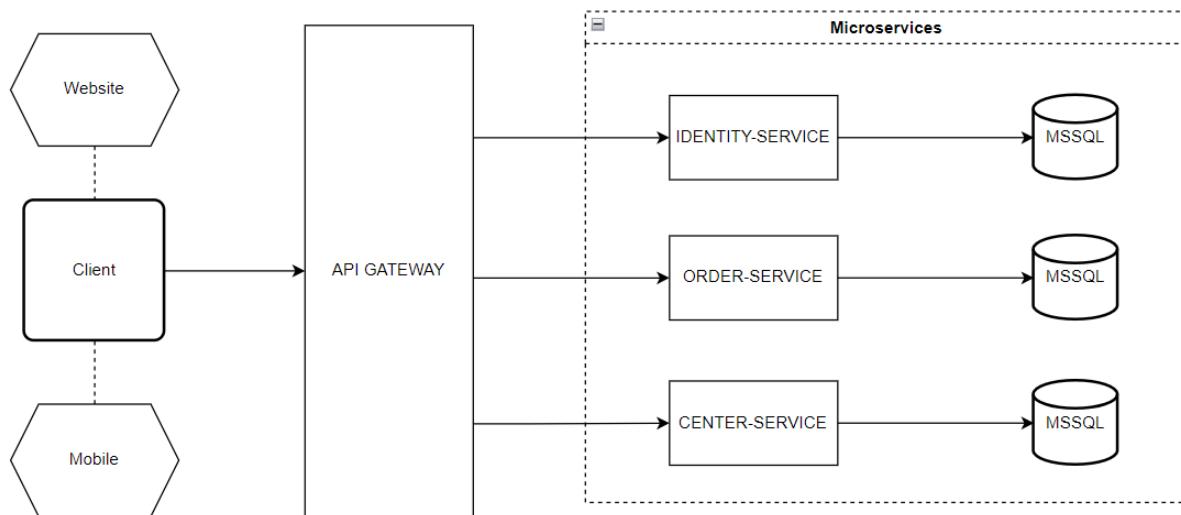


Mẫu thiết kế: Singleton

Mô tả: ApiClient là đối tượng dùng để kết nối tạo lời gọi api nhằm truy vấn và thao tác dữ liệu, áp dụng mẫu singleton lên đối tượng này có tác dụng duy trì hoạt động của một đối tượng tạo kết nối duy nhất trong quá trình sử dụng phần mềm, tránh lãng phí tài nguyên và thống nhất trong cách truyền token (bảo mật phân quyền khi gọi api).

11. Thiết kế kiến trúc hệ thống

a. Kiến trúc



- Hệ thống được xây dựng dựa theo mô hình kiến trúc Microservice, phân chia các database và service riêng lẻ hoạt động với mục đích khác nhau trong nghiệp vụ.

- API Gateway có chức năng routing các request đến các service con để truy cập tài nguyên. Ở đây cũng diễn ra hành động xác thực và phân quyền cho các tài khoản đăng nhập có quyền khác nhau. Bên cạnh đó, API Gateway cũng hỗ trợ việc cân bằng tải với số lượng truy cập lớn vào hệ thống, giúp hệ thống hoạt động ổn định hơn.

- Identity-service xây dựng với mục đích quản lý người dùng gồm Nhân viên với các chức vụ khác nhau, khách hàng, xác thực tài khoản, mật khẩu khi người dùng login vào hệ thống.

Identity-service cung cấp cơ chế xác thực với Json Web Token gửi cho người dùng sau khi hệ thống xác thực. Người dùng gửi request kèm token tới api để truy cập đến các endpoints cuối cùng.

- Order-service xây dựng với mục đích hỗ trợ việc đặt hàng của người dùng, việc order được xây dựng với một service riêng vì hành động này được dùng với một số lượng lớn request trong hệ

thống, xây dựng service riêng như vậy giúp giảm tải các request đến các service khác, giúp chương trình ổn định, hoạt động tốt hơn.

- Center-service xây dựng với mục đích quản trị các thành phần có trong hệ thống bán hàng như: bảng giá, đơn nhập hàng, thú cưng, sản phẩm.
- Các service truy cập đến các database riêng biệt cũng hỗ trợ cho việc lưu trữ dữ liệu một cách độc lập, mở rộng hệ thống dễ dàng hơn.

b. Công nghệ sử dụng

ba. Ngôn ngữ lập trình: Java Android (Frontend) và Java Spring Boot (Backend).

- Java là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay. Nó được sử dụng nhiều để phát triển phần mềm & web. Hiện nay, Java đã trở thành một ngôn ngữ phổ biến cho các ứng dụng di động. Còn Android được xem là nền tảng dựa trên các thiết bị điện thoại di động do Google phát triển [1].

- Trong đó, sự phát triển của Android hầu hết đều dựa trên Java. Một phần lớn các thư viện Java đều có sẵn trong nền tảng Android, chỉ một bộ phận thư viện khác tồn tại trong Android (cho giao diện người dùng,...) [1].

- Spring Boot là một framework Java được sử dụng để xây dựng các ứng dụng và dịch vụ web dễ dàng và nhanh chóng. Nền tảng cung cấp các cấu hình mặc định cho một số thư viện và bộ công cụ hỗ trợ xây dựng, triển khai, quản lý ứng dụng Spring-based. Cách Spring Boot hoạt động nhằm tối ưu hóa quy trình phát triển ứng dụng Java. Điều này sẽ giúp nhà phát triển tập trung vào việc xây dựng tính năng chính của ứng dụng mà không cần phải lo lắng về cấu hình phức tạp [2].

- Spring Boot xuất phát từ dự án Spring Framework, đây chính là một framework lập trình ứng dụng Java phổ biến. Spring Boot được phát triển bởi công ty Pivotal Software (nay thuộc VMware) với mục đích làm đơn giản hóa quá trình phát triển ứng dụng Spring. Công cụ sẽ giảm khối lượng công việc cần thiết để cấu hình và triển khai ứng dụng [2].

- Ưu điểm [2]:

- + Tự động cấu hình: Spring Boot sử dụng cơ chế cấu hình tự động thông minh, cho phép ứng dụng tự cấu hình dựa trên các thư viện và module được sử dụng.
- + Quản lý phụ thuộc: Spring Boot cung cấp các công cụ quản lý phụ thuộc mạnh mẽ như Maven hoặc Gradle, giúp quản lý các phụ thuộc của ứng dụng một cách hiệu quả.
- + Monitoring và quản lý: Spring Boot cung cấp các công cụ hỗ trợ giám sát và quản lý ứng dụng dễ dàng, bao gồm Spring Boot Actuator cho việc giám sát và quản lý ứng dụng.
- + Tối ưu hóa quá trình phát triển: Spring Boot cung cấp cấu hình mặc định thông minh và tự động, giúp giảm thiểu việc cấu hình thủ công và tối ưu quá trình phát triển ứng dụng Java.

+ Tích hợp tốt: Spring Boot tích hợp tốt với nhiều công nghệ và thư viện khác trong hệ sinh thái Spring Framework. Nền tảng cho phép hệ thống dễ dàng tích hợp các module và dịch vụ khác nhau mà không cần phải lo lắng về cấu hình phức tạp.

bb Framework: Spring Security và Spring Cloud Gateway

Spring Security là một framework mạnh mẽ và linh hoạt dùng để bảo mật các ứng dụng dựa trên nền tảng Java, đặc biệt là các ứng dụng sử dụng Spring Framework. Dưới đây là một số tính năng chính của Spring Security:

- **Authentication (Xác thực):** Spring Security hỗ trợ nhiều cơ chế xác thực khác nhau như Basic Authentication, Form-Based Authentication, OAuth2, và LDAP. Điều này giúp đảm bảo rằng chỉ những người dùng hợp lệ mới có thể truy cập vào hệ thống.
- **Authorization (Ủy quyền):** Sau khi người dùng được xác thực, Spring Security quản lý quyền truy cập của họ vào các tài nguyên dựa trên các vai trò và quyền hạn. Nó cung cấp kiểm soát truy cập ở mức URL, phương thức, và nhiều hơn nữa.
- **Protection Against Common Threats:** Spring Security bảo vệ ứng dụng khỏi các mối đe dọa phổ biến như Cross-Site Request Forgery (CSRF), Clickjacking, và Session Fixation.
- **Integration and Customization:** Spring Security tích hợp chặt chẽ với các dự án Spring khác và có thể được tùy chỉnh để đáp ứng các yêu cầu bảo mật đặc thù.

Spring Cloud Gateway là một API Gateway dựa trên Spring Framework, cung cấp một giải pháp điều phối lưu lượng mạnh mẽ và linh hoạt cho các hệ thống microservices. Dưới đây là một số tính năng chính của Spring Cloud Gateway:

- **Routing (Định tuyến):** Spring Cloud Gateway định tuyến các yêu cầu HTTP đến các dịch vụ backend dựa trên các quy tắc cấu hình. Điều này giúp phân phối lưu lượng đến đúng dịch vụ mục tiêu.
- **Load Balancing (Cân bằng tải):** Tích hợp với các công cụ cân bằng tải để phân phối yêu cầu giữa các instance của dịch vụ, giúp tăng cường khả năng chịu tải và tính sẵn sàng của hệ thống.
- **Filters (Bộ lọc):** Hỗ trợ các bộ lọc trước và sau khi xử lý yêu cầu, cho phép thực hiện các thao tác như xác thực, ủy quyền, ghi log, và chỉnh sửa yêu cầu/phản hồi.
- **Service Discovery (Khám phá dịch vụ):** Tích hợp với các công cụ khám phá dịch vụ như Eureka để tự động tìm kiếm và định tuyến đến các dịch vụ đang chạy.
- **Security Integration:** Tích hợp dễ dàng với Spring Security để bảo vệ các endpoint, đảm bảo rằng các yêu cầu được xác thực và ủy quyền trước khi được định tuyến đến dịch vụ backend.

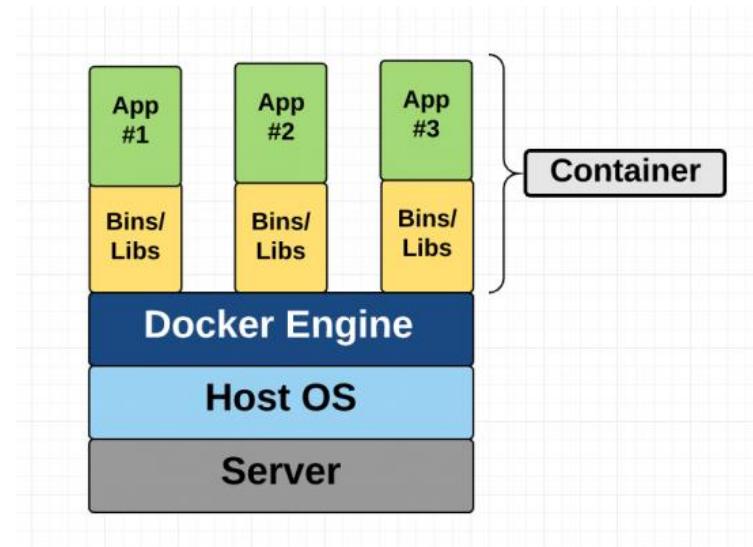
bc. Cơ sở dữ liệu: Microsoft SQL Server.

- SQL Server (hay Microsoft SQL Server) là một hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS) được phát triển bởi Microsoft [3].

- SQL Server cung cấp cho người dùng các công cụ và tính năng để quản lý, lưu trữ, xử lý các truy vấn dữ liệu, kiểm soát truy cập, xử lý giao dịch và hỗ trợ tích hợp dữ liệu từ nhiều nguồn khác nhau [3].
- Ngoài ra, SQL Server cũng cung cấp các công cụ để tạo báo cáo, phân tích và quản lý cơ sở dữ liệu trực quan thông qua giao diện người dùng hoặc các script lệnh SQL [3].
- SQL Server được xây dựng dựa trên SQL, một ngôn ngữ lập trình tiêu chuẩn để tương tác với cơ sở dữ liệu quan hệ. SQL Server được liên kết với Transact-SQL hoặc T-SQL, triển khai SQL của Microsoft có bổ sung một tập hợp các cấu trúc lập trình độc quyền [3].

bd. Triển khai service trên Docker

- Docker là nền tảng cung cấp cho các công cụ, service để các developers, admins systems có thể phát triển, thực thi, chạy các ứng dụng với containers. Hay nói một cách khác nó là một nền tảng để cung cấp cách để building, deploy và run các ứng dụng một cách dễ dàng trên nền tảng ảo hóa - "Build once, run anywhere" [4].
- Ứng dụng Docker chạy trong vùng chứa (container) có thể được sử dụng trên bất kỳ hệ thống nào: máy tính xách tay của nhà phát triển, hệ thống trên cơ sở hoặc trong hệ thống đám mây. Và là một công cụ tạo môi trường được "đóng gói" (còn gọi là Container) trên máy tính mà không làm tác động tới môi trường hiện tại của máy, môi trường trong Docker sẽ chạy độc lập [4].
- Docker có thể làm việc trên nhiều nền tảng như Linux, Microsoft Windows và Apple OS X [4].
- Một Container cung cấp ảo hóa ở cấp hệ điều hành bằng một khái niệm trùu tượng là “user space”. Container có thể chia sẻ host system’s kernel với các container khác [4].



Hình ảnh: Containter

- Các gói container là một user space bao gồm ứng dụng, system binaries và libraries mà không cần guest OS hoặc ảo hóa phần cứng như VMs. Đây là cái mà làm cho các container nhẹ hơn (lightweight). Các container sẽ chạy trên công nghệ cụ thể ở đây là Docker Engine [4].

be. Thư viện API phía Android: Retrofit 2

- Retrofit là thư viện vô cùng hữu ích cho việc kết nối internet và nhận dữ liệu từ server một cách dễ dàng và viết code theo mô hình chuẩn RESTful Webservices [5].

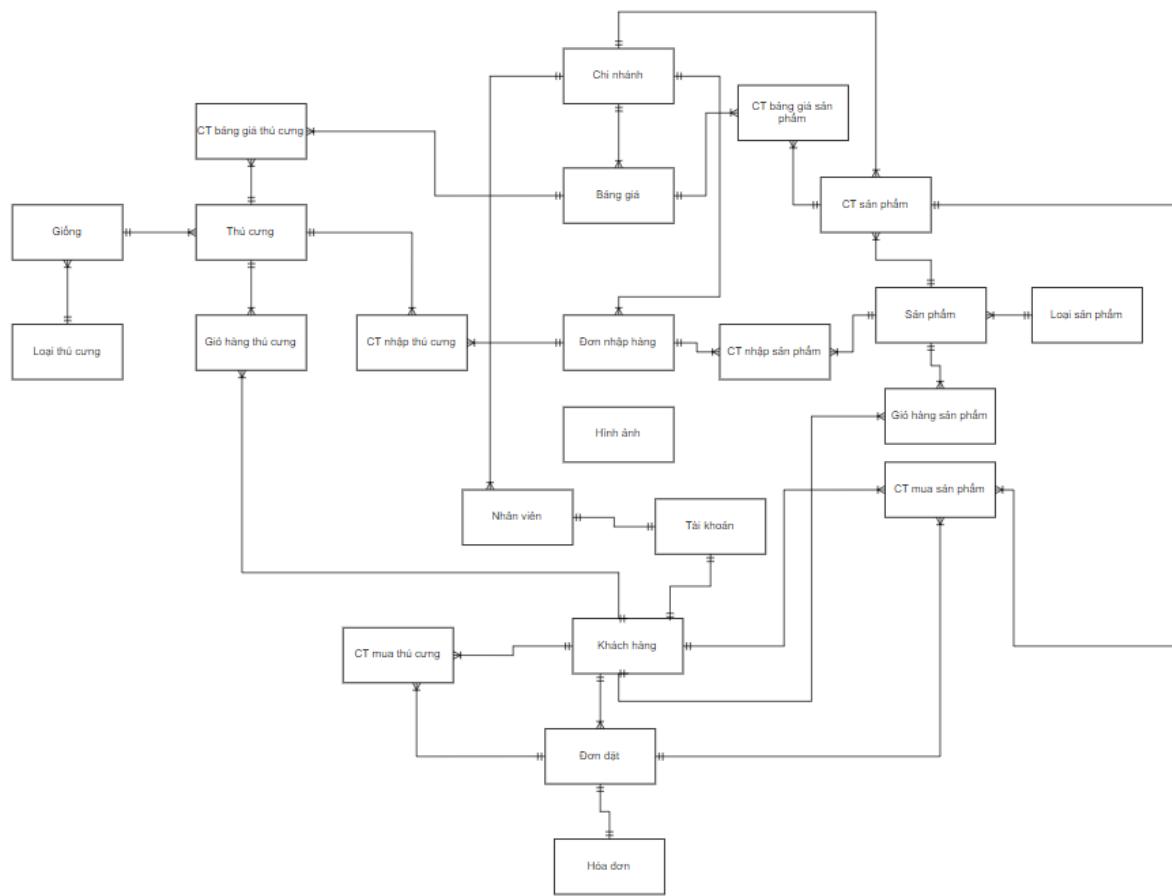
- Retrofit là một HTTP client type-safe cho Android, Java và kotlin được phát triển bởi Square. Retrofit giúp dễ dàng kết nối đến một dịch vụ REST trên web bằng cách chuyển đổi API thành Java Interface [5].

- Giống như hầu hết các phần mềm mã nguồn mở khác, Retrofit được xây dựng dựa trên một số thư viện mạnh mẽ và công cụ khác. Retrofit không tích hợp bất kỳ một bộ chuyển đổi JSON nào để phân tích từ JSON thành các đối tượng Java. Thay vào đó nó đi kèm với các thư viện chuyển đổi JSON để xử lý điều đó [5].

- Một số thư viện hỗ trợ:

- Gson: com.squareup.retrofit:converter-gson
- Jackson: com.squareup.retrofit:converter-jackson
- Moshi: com.squareup.retrofit:converter-moshi

12. Thiết kế cơ sở dữ liệu



Mô hình ERD của toàn hệ thống

Ghi chú:

Gạch dưới: Khóa chính

In nghiêng: Khóa ngoại

a. Bảng Bảng giá.

**BANGGIA(MABANGGIA, THOIGIANBATDAU, THOIGIANKETTHUC,
NOIDUNG, *MACHINHANH*, TRANGTHAI)**

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MABANGGIA	long		Primary key	
2	THOIGIANBATDAU	datetime			
3	THOIGIANKETTHUC	DATETIME			
4	NOIDUNG	TEXT			
5	MACHINHANH	INT		Foreign key	
6	TRANGTHAI	BOOLEAN			1: Đang áp dụng 0: Không áp dụng

b. Bảng Chi Nhánh.

CHINHANH(MACHINHANH,TENCHINHANH)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MACHINHANH	int		Primary key	
2	TENCHINHANH	nvarchar	50		

c. **Bảng Chi tiết bảng giá sản phẩm.**

CTBANGGIASANPHAM (MABANGGIA, MASANPHAM, DONGIA)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MABANGGIA	long		Primary key, Foreign key	
2	MASANPHAM	long		Primary key, Foreign key	
3	DONGIA	money			

d. **Bảng Chi tiết bảng giá thú cưng.**

CTBANGGIATHUCUNG(MABANGGIA, MATHUCUNG, DONGIA)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MABANGGIA	long		Primary key, Foreign key	
2	MATHUCUNG	long		Primary Key, Foreign key	
3	DONGIA	money			

e. **Bảng Chi tiết mua sản phẩm.**

**CHITIETMUASANPHAM(SODONDAT, MASANPHAM, MACHINHANH,
SOLUONG, DONGIA)**

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	<u>SODONDAT</u>	Long		Primary key, Foreign key	
2	<u>MASANPHAM</u>	Long		Primary key, Foreign key	
3	<u>MACHINHANH</u>	int		Primary key, Foreign key	
4	<u>SOLUONG</u>	int			
5	<u>DONGIA</u>	money			

f. **Bảng Chi tiết mua thú cưng.**

CHITIETMUATHUCUNG(MADONDAT, MATHUCUNG, DONGIA,SOLUONG)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	<u>MADONDAT</u>	Long		Primary key, Foreign key	
2	<u>MATHUCUNG</u>	Long		Primary key, Foreign key	
4	<u>SOLUONG</u>	int			

5	DONGIA	money			
---	--------	-------	--	--	--

g. Bảng Chi tiết nhập sản phẩm.

CTNHAPSANPHAM(MADONNHAP, MASANPHAM, SOLUONG, DONGIA)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MADONNHAP	LONG		Primary key, Foreign key	
2	MASANPHAM	LONG		Primary key, Foreign key	
3	SOLUONG	int			
4	DONGIA	money			

h. Bảng Chi tiết nhập thú cưng.

CTNHAPTHUCUNG(MADONNHAP, MATHUCUNG, SOLUONG, DONGIA)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MADONNHAP	Long		Primary key, Foreign key	
2	MATHUCUNG	Long		Primary key, Foreign key	
3	SOLUONG	Int			
4	DONGIA	money			

i. Bảng Chi tiết sản phẩm.

CTSANPHAM (MASANPHAM, MACHINHANH, SOLUONGTON, DONGIA)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MASANPHAM	long		Primary key, Foreign key	
2	MACHINHANH	Int		Primary key, Foreign key	
3	SOLUONGTON	Int			
4	DONGIA	money			

j. Bảng Đơn đặt.

**DONDAT(SODONDAT, NGAYLAP, DIACHIDAT,
SODIENTHOAI,MACHINHANH,MAKHACHHANG)**

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	SODONDAT	Long		Primary key	
2	NGAYLAP	Date			
3	DIACHIDAT	Nvarchar	200		
4	SODIENTHOAI	nvarchar	10		
5	MACHINHANH	int			

6	MAKHACHHANG	nvarchar	50		
---	-------------	----------	----	--	--

k. Bảng Đơn nhập hàng.

DONNHAPHANG (MADONNNHAPHANG, NGAYLAP, MANHANVIEN, MACHINHANH)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MADONNNHAPHANG	Long		Primary key	
2	NGAYLAP	Date			
3	MANHANVIEN	nvarchar	50		
4	MACHINHANH	int			

l. Bảng Giống.

GIONG(MAGIONG, TENGIONG)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MAGIONG	Int		Primary key	
2	TENGIONG	Nvarchar	50		

m. Bảng Hình ảnh.

HINHANH(MAHINHANH, PATH, MATHUCUNG, MANHANVIEN, MASANPHAM, MAKHACHHANG, TENHINHANH, TENDUYNHAT, LOAIHINHANH, TRANGTHAI)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MAHINHANH	Int		Primary key	
2	PATH	Nvarchar	100		
3	MATHUCUNG	LONG			
4	MANHANVIEN	Nvarchar	50		
5	MASANPHAM	Long			
6	MAKHACHHANG	nvarchar	50		
7	TENHINHANH	Nvarchar	100		
8	TENDUYNHAT	Nvarchar	100		
9	LOAIHINHANH	Nvarchar	50		
10	TRANGTHAI	boolean			

n. Bảng Hóa đơn.

HOADON(SOHOADON, NGAYLAP, TONGHOADON)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	SOHOADON	long		Primary key	
2	NGAYLAP	Date			
	TONGHOADON	money			

o. Bảng Khách hàng.

KHACHHANG(MAKHACHHANG, HO, TEN, GIOITINH, NGAYSINH, DIACHI, SODIENTHOAI, EMAIL, CCCD)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MAKHACHHANG	nvarchar	50	Primary key	
2	HO	nvarchar	50		
3	TEN	nvarchar	50		
4	GIOITINH	nvarchar	50		
5	NGAYSINH	Date			
6	DIACHI	nvarchar	200		
7	SODIENTHOAI	nvarchar	10		

8	EMAIL	nvarchar	50		
9	CCCD	nvarchar	12		

p. Bảng Loại sản phẩm.

LOAISANPHAM(MALO AISANPHAM, TENLOAISANPHAM)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MALO AISANPHAM	Int		Primary key	
2	TENLOAISANPHAM	nvarchar	50		

q. Bảng Loại thú cưng.

LOAITHUCUNG(MALO AITHUCUNG, TENLOAITHUCUNG)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MALO AITHUCUNG	Int		Primary key	
2	TENLOAITHUCUNG	nvarchar	50		

r. Bảng Loại sản phẩm.

LOAISANPHAM(MALO AISANPHAM, TENLOAISANPHAM)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	<u>MALO AISANPHAM</u>	Int		Primary key	
2	<u>TENLOAISANPHAM</u>	nvarchar	50		

s. Bảng Nhân viên.

NHANVIEN(MANHANVIEN, HO, TEN, CHUCVU, SODIENTHOAI, EMAIL, CCCD, MACHINHANH)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	<u>MANHANVIEN</u>	nvarchar	50	Primary key	
2	<u>HO</u>	nvarchar	50		
3	<u>TEN</u>	nvarchar	50		
4	<u>CHUCVU</u>	nvarchar	50		
5	<u>SODIENTHOAI</u>	nvarchar	10		
6	<u>EMAIL</u>	nvarchar	50		
7	<u>CCCD</u>	nvarchar	12		
8	<u>MACHINHANH</u>	INT			

t. Bảng Sản phẩm.

SANPHAM(MASANPHAM, TENSANPHAM, GIAHIENTAI, MALO AISANPHAM)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MASANPHAM	Int		Primary key	
2	TENSANPHAM	nvarchar	50		
3	GIAHIENTAI	MONEY			
4	MALO AISANPHAM	INT			

u. Bảng Tài khoản.

**TAIKHOAN(TENDANGNHAP, MATKHAU, QUYEN, MAXACNHAN,
THOIGIANTAOMA, THOIGIANHETHAN, THOIGIANXACNHAN, TRANGTHAI)**

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	TENDANGNHAP	nvarchar		Primary key	
2	MATKHAU	nvarchar	50		
3	QUYEN	nvarchar			
4	MAXACNHAN	nvarchar			
5	THOIGIANTAOMA	Datetime			
6	THOIGIANHETHAN	Datetime			

7	THOIGIANXACNHAN	Datetime			
8	TRANGTHAI	Boolean			

v. Bảng Thú cưng.

THUCUNG(MATHUCUNG, TENTHUCUNG, TRANGTHAI, CHU, MOTA, GIAHIENTAI, MACHINHANH, MAGIONG)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MATHUCUNG	Long		Primary key	
2	TENTHUCUNG	nvarchar	50		
3	TRANGTHAI	Boolean			
4	CHU	nvarchar			
5	MOTA	nvarchar			
6	GIAHIENTAI	money			
7	MACHINHANH	int			
8	MAGIONG	int			

x. Bảng Giỏ hàng thú cưng.

GHTHUCUNG(MAKHACHHANG, MATHUCUNG)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú

1	MAKHACHHANG	nvarchar	50	Primary key	
2	MATHUCUNG	LONG		Primary key	

y. **Bảng Giỏ hàng sản phẩm.**

GHTHUCUNG(MAKHACHHANG, MATHUCUNG)

STT	Thuộc tính	Kiểu dữ liệu	Độ dài	Ràng buộc	Ghi chú
1	MAKHACHHANG	nvarchar	50	Primary key	
2	MASANPHAM	LONG		Primary key	
3	MACHINHANH	int		Primary key	

Chương III: Tài liệu API

1. Các quy ước chung

Bảng mã		
STT	Code	Message
1	200	OK
2	400	BAD_REQUEST
3	403	FORBIDDEN
4	404	NOT_FOUND
5	409	CONFLICT
6	500	INTERNAL_SERVER_ERROR
7	502	BAD_GATEWAY
8	503	SERVICE_UNAVAILABLE

Đường dẫn chung (base_url): <http://localhost:8989>

2. Identity service

a. Bảng API

STT	Path	Method	Yêu cầu đầu vào	Mô tả
1	identity/login	Post	LoginDTO	Xác thực đăng nhập
2	identity/nhanvien	Get	Không	Lấy danh sách nhân viên
3	identity/nhanvien/{id}	Get	String	Lấy nhân viên cụ thể
4	identity/nhanvien	Post	NhanVienDTO	Thêm nhân viên
5	identity/nhanvien	Put	NhanVienDTO	Sửa thông tin nhân viên
6	identity/nhanvien	Delete	String	Xóa nhân viên
7	identity/khachhang	Get	Không	Lấy danh sách khách hàng
8	identity/khachhang/{id}	Get	String	Lấy khách hàng cụ thể
9	identity/khachhang	Put	KhachHangDTO	Sửa thông tin khách hàng
10	identity/tk	Get	Không	Lấy danh sách tài khoản
11	identity/tk/{id}	Get	String	Lấy tài khoản cụ thể
12	identity/tk	Put	TaiKhoanDTO	Sửa thông tin tài khoản
13	identity/tk/reset	Put	TaiKhoanDTO	Reset mật khẩu tài khoản
14	identity/register	Post	RegisterRequest	Đăng ký tài khoản
15	identity/register /confirm	Post	ConfirmEmailRequest	Xác thực tài khoản email
16	identity/register /getVerifiedCode	Post	RegisterRequest	Gửi lại mã xác thực

17	identity/hinhanh	Post	RequestParam: image - MultipartFile[] maNhanVien - String maKhachHang - String	Lưu hình ảnh khách hàng/nhân viên
18	identity/hinhanh/get	Post	long []	Nhận hình ảnh khách hàng/nhân viên
19	identity/forget	Post	String	Gửi mã xác thực tài khoản
20	identity/forget/confirm	Post	ConfirmEmailRequest	Xác thực mã
21	identity/forget	Put	TaiKhoanDTO	Cập nhật mật khẩu mới

b. Chi tiết:

STT1: Xác thực đăng nhập

POST: {base_url} /identity/login

Đầu vào: @Body LoginDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
tenDangNhap	String	Có	Tên đăng nhập
matKhau	String	Có	Mật khẩu

Ví dụ:

```
{
    "tenDangNhap": "NV1",
    "matKhau": "123"
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoginResponse	Thành công
400	"Vui lòng điền đầy đủ thông tin"	Thiếu thông tin cung cấp
	"Tài khoản bị khóa!"	Tài khoản đã bị khóa trong hệ thống

404	"Tài khoản không tồn tại"	Tài khoản không tồn tại trong hệ thống
-----	---------------------------	--

LoginResponse Body:

Thuộc tính	Kiểu	Mô tả
tenDangNhap	String	Tên đăng nhập
quyen	String	Quyền của tài khoản thuộc 1 trong 4 TH: “nhanvien”, “khachhang”, “quanly”, “admin”
token	String	Token xác thực phân quyền api

Ví dụ:

```
{
    "tenDangNhap": "NV1",
    "quyen": "admin",
    "token":
    "eyJhbGciOiJIUzUxMiJ9.eyJyb2xIjoiYWRtaW4iLCJzdWIiOiJOVjEiLCJpYXQiOjE3MTgxODUzNDYsImV4
    cCI6MTcxODIyMTM0Nn0.ruDjxIFcitONmhfPhatPAqHN_i5b3PL87u8OZdWetzT7iGpcX1-xUidECsa-
    s7377Vrs1L113JU-RqMWUNOFlw"
}
```

STT2: Lấy danh sách nhân viên

GET: {base_url} /identity/nhanvien

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<NhanVienDTO>	Thành công

NhanVienDTO Body:

Thuộc tính	Kiểu	Mô tả
maNhanVien	String	Mã nhân viên
ho	String	Họ nhân viên

ten	String	Tên nhân viên
cccd	String	Căn cước công dân
chucVu	String	Chức vụ
soDienThoai	String	Số điện thoại
email	String	Email
maChiNhanh	int	Mã chi nhánh
hinhAnh	List<Long>	Danh sách id hình ảnh của nhân viên

Ví dụ:

```
[
  {
    "maNhanVien": "N100",
    "ho": "24",
    "ten": "13",
    "cccd": "2132",
    "chucVu": "nhanvien",
    "soDienThoai": "75466533",
    "email": "sf@gmail.com",
    "maChiNhanh": 1,
    "hinhAnh": null
  },
  {
    "maNhanVien": "NV1",
    "ho": "Vũ Văn",
    "ten": "Lâm",
    "cccd": "1234567890",
    "chucVu": "admin",
    "soDienThoai": "0917",
    "email": "lamvu",
    "maChiNhanh": 1,
    "hinhAnh": null
  }
]
```

STT3: Lấy nhân viên cụ thể

GET: {base_url} /identity/nhanvien/{id}

Đầu vào: @Path String id

Ví dụ: {base_url} /identity/nhanvien/NV1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	NhanVienDTO	Thành công
404	"Entity not found"	Không có nhân viên

NhanVienDTO Body:

Thuộc tính	Kiểu	Mô tả
maNhanVien	String	Mã nhân viên
ho	String	Họ nhân viên
ten	String	Tên nhân viên
cccd	String	Căn cước công dân
chucVu	String	Chức vụ
soDienThoai	String	Số điện thoại
email	String	Email
maChiNhanh	int	Mã chi nhánh
hinhAnh	List<Long>	Danh sách id hình ảnh của nhân viên

Ví dụ:

```
{  
    "maNhanVien": "NV1",  
    "ho": "Vũ Văn",  
    "ten": "Lâm",  
    "cccd": "1234567890",  
    "chucVu": "admin",  
    "soDienThoai": "0917",  
    "email": "lamvu",  
    "maChiNhanh": 1,  
    "hinhAnh": null  
}
```

STT4: Thêm nhân viên

POST: {base_url} /identity/nhanvien

Đầu vào: @Body NhanVienDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maNhanVien	String	Có	Mã nhân viên
ho	String	Có	Họ nhân viên
ten	String	Có	Tên nhân viên
cccd	String	Có	Căn cước công dân
chucVu	String	Có	Chức vụ
soDienThoai	String	Có	Số điện thoại
email	String	Có	Email
maChiNhanh	int	Có	Mã chi nhánh
hinhAnh	List<Long>	Không	Danh sách id hình ảnh của nhân viên

Ví dụ:

```
{  
    "maNhanVien": "NV1",  
    "ho": "Vũ Văn",  
    "ten": "Lâm",  
    "cccd": "1234567890",  
    "chucVu": "admin",  
    "soDienThoai": "0917",  
    "email": "lamvu",  
    "maChiNhanh": 1,  
    "hinhAnh": null  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	NhanVienDTO	Thành công
400	"Thêm thất bại"	Lưu nhân viên hoặc tài khoản thất bại

NhanVienDTO Body:

Thuộc tính	Kiểu	Mô tả
maNhanVien	String	Mã nhân viên
ho	String	Họ nhân viên
ten	String	Tên nhân viên
cccd	String	Căn cước công dân
chucVu	String	Chức vụ
soDienThoai	String	Số điện thoại
email	String	Email
maChiNhanh	int	Mã chi nhánh
hinhAnh	List<Long>	Danh sách id hình ảnh của nhân viên

Ví dụ:

```
{  
    "maNhanVien": "NV1",  
    "ho": "Vũ Văn",  
    "ten": "Lâm",  
    "cccd": "1234567890",  
    "chucVu": "admin",  
    "soDienThoai": "0917",  
    "email": "lamvu",  
    "maChiNhanh": 1,  
    "hinhAnh": null  
}
```

STT5: Sửa nhân viên

PUT: {base_url} /identity/nhanvien

Đầu vào: @Body NhanVienDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maNhanVien	String	Có	Mã nhân viên
ho	String	Có	Họ nhân viên
ten	String	Có	Tên nhân viên
cccd	String	Có	Căn cước công dân
chucVu	String	Có	Chức vụ
soDienThoai	String	Có	Số điện thoại
email	String	Có	Email
maChiNhanh	int	Có	Mã chi nhánh
hinhAnh	List<Long>	Không	Danh sách id hình ảnh của nhân viên

Ví dụ:

```
{  
    "maNhanVien": "NV1",  
    "ho": "Vũ Văn",  
    "ten": "Lâm",  
    "cccd": "1234567890",  
    "chucVu": "admin",  
    "soDienThoai": "0917",  
    "email": "lamvu",  
    "maChiNhanh": 1,  
    "hinhAnh": null  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	NhanVienDTO	Thành công
400	"Nhân viên không tồn tại"	Không tồn tại nhân viên trong hệ thống

NhanVienDTO Body:

Thuộc tính	Kiểu	Mô tả
maNhanVien	String	Mã nhân viên
ho	String	Họ nhân viên
ten	String	Tên nhân viên
cccd	String	Căn cước công dân
chucVu	String	Chức vụ
soDienThoai	String	Số điện thoại
email	String	Email
maChiNhanh	int	Mã chi nhánh
hinhAnh	List<Long>	Danh sách id hình ảnh của nhân viên

Ví dụ:

```
{  
    "maNhanVien": "NV1",  
    "ho": "Vũ Văn",  
    "ten": "Lâm",  
    "cccd": "1234567890",  
    "chucVu": "admin",  
    "soDienThoai": "0917",  
    "email": "lamvu",  
    "maChiNhanh": 1,  
    "hinhAnh": null  
}
```

STT6: Xóa nhân viên

DELETE: {base_url} /identity/nhanvien/{id}

Đầu vào: @Path String id

Ví dụ: {base_url} /identity/nhanvien/NV1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xóa thành công"	Thành công
400	"Nhân viên không tồn tại"	Không tồn tại nhân viên trong hệ thống
	"Xóa thất bại"	Thao tác xóa thất bại

STT7: Lấy danh sách khách hàng

GET: {base_url} /identity/khachhang

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<KhachHangDTO>	Thành công

KhachHangDTO Body:

Thuộc tính	Kiểu	Mô tả
maKhachHang	String	Mã khách hàng (Tên đăng nhập)
ho	String	Họ khách hàng
ten	String	Tên khách hàng
gioiTinh	Boolean	1: nam 0: nữ
ngaySinh	Date	Ngày sinh khách hàng
soDienThoai	String	Số điện thoại
email	String	Email liên kết của khách hàng
cccd	String	Căn cước công dân của khách hàng
diaChi	String	Địa chỉ liên lạc của khách hàng
hinhAnh	List<Long>	Danh sách id hình ảnh của khách hàng

Ví dụ:

```
[
  {
    "maKhachHang": "lamvu10000",
    "ho": "vu",
    "ten": "lam",
    "gioiTinh": true,
    "ngaySinh": null,
    "soDienThoai": "1234567",
    "email": "longsk123456789@gmail.com",
    "cccd": "1234567890",
    "diaChi": null,
    "hinhAnh": null
  },
  {
    "maKhachHang": "lamvu1020",
    "ho": "lam",
    "ten": "vu",
    "gioiTinh": false,
    "ngaySinh": "1990-01-01",
    "soDienThoai": "0987654321",
    "email": "test@gmail.com",
    "cccd": "12345678901234567890",
    "diaChi": "Hà Nội",
    "hinhAnh": [1, 2, 3]
  }
]
```

```

    "gioiTinh": true,
    "ngaySinh": null,
    "soDienThoai": "1234567890",
    "email": "longsk123456789@gmail.com",
    "cccd": "1234567890",
    "diaChi": null,
    "hinhAnh": null
  ]
}

```

STT8: Lấy khách hàng cụ thể

GET: {base_url} /identity/khachhang/{id}

Đầu vào: @Path String id

Ví dụ: {base_url} /identity/khachhang/lamvu2010

Đầu ra:

Code	Kết quả	Nguyên nhân
200	KhachHangDTO	Thành công
400	"Entity not found"	Không tìm thấy khách hàng

KhachHangDTO Body:

Thuộc tính	Kiểu	Mô tả
maKhachHang	String	Mã khách hàng (Tên đăng nhập)
ho	String	Họ khách hàng
ten	String	Tên khách hàng
gioiTinh	Boolean	1: nam 0: nữ
ngaySinh	Date	Ngày sinh khách hàng
soDienThoai	String	Số điện thoại
email	String	Email liên kết của khách hàng

cccd	String	Căn cước công dân của khách hàng
diaChi	String	Địa chỉ liên lạc của khách hàng
hinhAnh	List<Long>	Danh sách id hình ảnh của khách hàng

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "ho": "Vũ",
    "ten": "Lâm",
    "gioiTinh": false,
    "ngaySinh": "2002-06-13",
    "soDienThoai": "09172",
    "email": "lamvune",
    "cccd": "123456",
    "diaChi": "97 Man Thiện",
    "hinhAnh": [
        4
    ]
}
```

STT9: Sửa thông tin khách hàng

PUT: {base_url} /identity/khachhang

Đầu vào: @Body KhachHangDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng (Tên đăng nhập)
ho	String	Có	Họ khách hàng
ten	String	Có	Tên khách hàng
gioiTinh	Boolean	Có	1: nam 0: nữ
ngaySinh	Date	Không	Ngày sinh khách hàng

soDienThoai	String	Không	Số điện thoại
email	String	Có	Email liên kết của khách hàng
cccd	String	Không	Căn cước công dân của khách hàng
diaChi	String	Không	Địa chỉ liên lạc của khách hàng
hinhAnh	List<Long>	Không	Danh sách id hình ảnh của khách hàng

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "ho": "Vũ",
    "ten": "Lâm",
    "gioiTinh": false,
    "ngaySinh": "2002-06-13",
    "soDienThoai": "09172",
    "email": "lamvune",
    "cccd": "123456",
    "diaChi": "97 Man Thiện",
    "hinhAnh": [
        4
    ]
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	KhachHangDTO	Thành công
400	“Khách hàng không tồn tại”	Không có khách hàng trong hệ thống

KhachHangDTO Body:

Thuộc tính	Kiểu	Mô tả
maKhachHang	String	Mã khách hàng (Tên đăng nhập)
ho	String	Họ khách hàng

ten	String	Tên khách hàng
gioiTinh	Boolean	1: nam 0: nữ
ngaySinh	Date	Ngày sinh khách hàng
soDienThoai	String	Số điện thoại
email	String	Email liên kết của khách hàng
cccd	String	Căn cước công dân của khách hàng
diaChi	String	Địa chỉ liên lạc của khách hàng
hinhAnh	List<Long>	Danh sách id hình ảnh của khách hàng

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "ho": "Vũ",
    "ten": "Lâm",
    "gioiTinh": false,
    "ngaySinh": "2002-06-13",
    "soDienThoai": "09172",
    "email": "lamvune",
    "cccd": "123456",
    "diaChi": "97 Mân Thiện",
    "hinhAnh": [
        4
    ]
}
```

STT10: Lấy danh sách tài khoản

GET: {base_url} /identity/tk

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân

200	List<TaiKhoanDTO>	Thành công
-----	-------------------	------------

TaiKhoanDTO Body:

Thuộc tính	Kiểu	Mô tả
tenDangNhap	String	Tên đăng nhập hệ thống
matKhau	String	Mật khẩu của tài khoản
quyen	String	Phân quyền của tài khoản trên hệ thống
trangThai	Boolean	Trạng thái hoạt động của tài khoản: 0: Khóa 1: Đang hoạt động
maXacNhan	String	Mã xác thực của tài khoản
thoiGianTaoMa	LocalDateTime	Thời gian khi tạo mã xác thực
thoiGianHetHan	LocalDateTime	Thời gian hết hạn của mã. Thường là thoiGianTaoMa + 5 phút
thoiGianXacNhan	LocalDateTime	Thời gian xác thực mã

Ví dụ:

```
[
  {
    "tenDangNhap": "customer",
    "matKhau": "12356",
    "quyen": "khachhang",
    "trangThai": true,
    "maXacNhan": null,
    "thoiGianTaoMa": null,
    "thoiGianHetHan": null,
    "thoiGianXacNhan": null
  },
  {
    "tenDangNhap": "lamvu10000",
    "matKhau": "123",
    "quyen": "khachhang",
  }
]
```

```

        "trangThai": false,
        "maXacNhan": null,
        "thoiGianTaoMa": null,
        "thoiGianHetHan": null,
        "thoiGianXacNhan": null
    }
]

```

STT11: Lấy tài khoản cụ thể

GET: {base_url} /identity/tk/{id}

Đầu vào: @Path String id

Ví dụ: {base_url} /identity/khachhang/lamvu2010

Đầu ra:

Code	Kết quả	Nguyên nhân
200	TaiKhoanDTO	Thành công
400	"Tài khoản không tồn tại"	Tài khoản không tồn tại trong hệ thống

TaiKhoanDTO Body:

Thuộc tính	Kiểu	Mô tả
tenDangNhap	String	Tên đăng nhập hệ thống
matKhau	String	Mật khẩu của tài khoản
quyen	String	Phân quyền của tài khoản trên hệ thống
trangThai	Boolean	Trạng thái hoạt động của tài khoản: 0: Khóa 1: Đang hoạt động
maXacNhan	String	Mã xác thực của tài khoản
thoiGianTaoMa	LocalDateTime	Thời gian khi tạo mã xác thực

thoiGianHetHan	LocalDateTime	Thời gian hết hạn của mã. Thường là thoiGianTaoMa + 5 phút
thoiGianXacNhan	LocalDateTime	Thời gian xác thực mã

Ví dụ:

```
{
    "tenDangNhap": "customer",
    "matKhau": "12356",
    "quyen": "khachhang",
    "trangThai": true,
    "maXacNhan": null,
    "thoiGianTaoMa": null,
    "thoiGianHetHan": null,
    "thoiGianXacNhan": null
}
```

STT12: Sửa thông tin tài khoản

PUT: {base_url} /identity/tk

Đầu vào: @Body TaiKhoanDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
tenDangNhap	String	Có	Tên đăng nhập hệ thống
matKhau	String	Có	Mật khẩu của tài khoản
quyen	String	Không	Phân quyền của tài khoản trên hệ thống
trangThai	Boolean	Có	Trạng thái hoạt động của tài khoản: 0: Khóa 1: Đang hoạt động
maXacNhan	String	Không	Mã xác thực của tài khoản
thoiGianTaoMa	LocalDateTime	Không	Thời gian khi tạo mã xác thực
thoiGianHetHan	LocalDateTime	Không	Thời gian hết hạn của mã. Thường là thoiGianTaoMa + 5 phút

thoiGianXacNhan	LocalDateT ime	Không	Thời gian xác thực mã
-----------------	-------------------	-------	-----------------------

Ví dụ:

```
{
    "tenDangNhap": "customer",
    "matKhau": "12356",
    "quyen": "khachhang",
    "trangThai": false,
    "maXacNhan": null,
    "thoiGianTaoMa": null,
    "thoiGianHetHan": null,
    "thoiGianXacNhan": null
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	TaiKhoanDTO	Thành công
400	"Tài khoản không tồn tại"	Không có tài khoản trong hệ thống
	"Cập nhật thất bại"	Lưu thông tin tài khoản thất bại

TaiKhoanDTO Body:

Thuộc tính	Kiểu	Mô tả
tenDangNhap	String	Tên đăng nhập hệ thống
matKhau	String	Mật khẩu của tài khoản
quyen	String	Phân quyền của tài khoản trên hệ thống
trangThai	Boolean	Trạng thái hoạt động của tài khoản: 0: Khóa 1: Đang hoạt động
maXacNhan	String	Mã xác thực của tài khoản

thoiGianTaoMa	LocalDateTime	Thời gian khi tạo mã xác thực
thoiGianHetHan	LocalDateTime	Thời gian hết hạn của mã. Thường là thoiGianTaoMa + 5 phút
thoiGianXacNhan	LocalDateTime	Thời gian xác thực mã

Ví dụ:

```
{
    "tenDangNhap": "customer",
    "matKhau": "12356",
    "quyen": "khachhang",
    "trangThai": false,
    "maXacNhan": null,
    "thoiGianTaoMa": null,
    "thoiGianHetHan": null,
    "thoiGianXacNhan": null
}
```

STT13: Reset mật khẩu

PUT: {base_url} /identity/tk/reset

Đầu vào: @Body TaiKhoanDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
tenDangNhap	String	Có	Tên đăng nhập hệ thống
matKhau	String	Không	Mật khẩu của tài khoản
quyen	String	Không	Phân quyền của tài khoản trên hệ thống
trangThai	Boolean	Không	Trạng thái hoạt động của tài khoản: 0: Khóa 1: Đang hoạt động
maXacNhan	String	Không	Mã xác thực của tài khoản
thoiGianTaoMa	LocalDateTime	Không	Thời gian khi tạo mã xác thực

thoiGianHetHan	LocalDateTime	Không	Thời gian hết hạn của mã. Thường là thoiGianTaoMa + 5 phút
thoiGianXacNhan	LocalDateTime	Không	Thời gian xác thực mã

Ví dụ:

```
{
    "tenDangNhap": "customer",
    "matKhau": "12356",
    "quyen": "khachhang",
    "trangThai": true,
    "maXacNhan": null,
    "thoiGianTaoMa": null,
    "thoiGianHetHan": null,
    "thoiGianXacNhan": null
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	String matKhau (Mật khẩu reset của tài khoản)	Thành công
400	"Tài khoản không tồn tại"	Không có tài khoản trong hệ thống
	"Cập nhật thất bại"	Lưu thông tin mật khẩu tài khoản thất bại

STT14: Đăng ký tài khoản khách hàng

POST: {base_url} /identity/register

Đầu vào: @Body RegisterRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
ho	String	Có	Họ khách hàng
ten	String	Có	Tên khách hàng
tenDangNhap	String	Có	Tên tài khoản để đăng nhập chương trình

matKhau	String	Có	Mật khẩu của tài khoản
cccd	String	Không	Căn cước công dân
soDienThoai	String	Không	Số điện thoại liên lạc khách hàng
email	String	Có	Email liên kết tài khoản khách hàng. Để gửi mã xác thực.

Ví dụ:

```
{
    "ho": "Trần",
    "ten": "Gia Long",
    "tenDangNhap": "trangialong",
    "matKhau": "123",
    "cccd": null,
    "soDienThoai": null,
    "email": "trangialong@gmail.com"
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"ten dang nhap null"	Nhận về tên đăng nhập null
	"Tên đăng nhập đã tồn tại"	Đã tồn tại tài khoản trùng tên đăng nhập trong hệ thống
	"Email không đúng"	Email không đúng định dạng chuẩn abc@gmail.com
	String maXacNhan (Mã xác nhận tài khoản)	Thành công
400	e.getMessage()	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gặp phải.

STT15: Xác thực tài khoản đăng ký

POST: {base_url} /identity/register/confirm

Đầu vào: @Body ConfirmEmailRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
tenDangNhap	String	Có	Tên tài khoản để đăng nhập chương trình
maXacNhan	String	Có	Mã xác nhận

Ví dụ:

```
{
    "tenDangNhap": "trangialong",
    "maXacNhan": "444f8dd1-4"
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Tài khoản không tồn tại, vui lòng thử lại"	Tài khoản chưa tạo trong hệ thống
	"Tài khoản đã được xác nhận"	Tài khoản đã được xác nhận đăng ký trước đó rồi
	"Mã xác nhận hết hạn"	Mã xác nhận đã hết hạn (Quá 5 phút)
	"Xác nhận thành công, đăng nhập để tiếp tục"	Thành công
	"Mã xác nhận không chính xác"	Mã xác nhận nhận được không khớp với trong cơ sở dữ liệu
400	e.getMessage()	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gấp phải.

STT16: Gửi lại mã xác nhận vào email

POST: {base_url} /identity/register/getVerifiedCode

Đầu vào: @Body RegisterRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
ho	String	Có	Họ khách hàng
ten	String	Có	Tên khách hàng
tenDangNhap	String	Có	Tên tài khoản để đăng nhập chương trình
matKhau	String	Có	Mật khẩu của tài khoản
cccd	String	Không	Căn cước công dân
soDienThoai	String	Không	Số điện thoại liên lạc khách hàng
email	String	Có	Email liên kết tài khoản khách hàng. Để gửi mã xác thực.

Ví dụ:

```
{
    "ho": "Trần",
    "ten": "Gia Long",
    "tenDangNhap": "trangialong",
    "matKhau": "123",
    "cccd": null,
    "soDienThoai": null,
    "email": "trangialong@gmail.com"
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	String maXacNhan (Mã xác nhận tài khoản)	Thành công
400	e.getMessage()	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gấp phai.

STT17: Lưu hình ảnh nhân viên hoặc khách hàng

POST: {base_url} /identity/hinhanh

Đầu vào:

- + @RequestParam("image") MultipartFile[]
- + @RequestParam("maNhanVien") String
- + @RequestParam("maKhachHang") String

Tham số truyền vào có dạng:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
image	MultipartFile	Có	Nhiều file hình ảnh
maNhanVien	String	Không	Mã nhân viên
maKhachHang	String	Không	Mã khách hàng

Ví dụ: Lưu 1 hình ảnh của khách hàng lamvu2010.

```
image = Multipart(1),  
maNhanVien = null,  
maKhachHang = lamvu2010
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Lưu thành công"	Thành công
400	"Lưu thất bại"	Có lỗi trong quá trình lưu giữa các hình ảnh

STT18: Lấy về các hình ảnh của nhân viên hoặc khách hàng

POST: {base_url} /identity/hinhanh/get

Đầu vào: @Body long[]

Ví dụ: Lấy hình có id 1, 2 và 3

```
[1,2,3]
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<HinhAnhDTO>	Thành công
400	"Lỗi lấy hình"	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gấp phai.

HinhAnhDTO Body:

Thuộc tính	Kiểu	Mô tả
maHinhAnh	long	Id hình ảnh đã truyền
path	String	Đường dẫn hình ảnh
tenHinhAnh	String	Tên hình ảnh khi lưu
tenDuyNhat	String	Tên phân biệt giữa các hình ảnh trong thư mục
loaiHinhAnh	String	Loại hình ảnh
source	byte[]	Đoạn byte mã hóa hình ảnh. Khi nhận từ phía client cần nhận bằng String và dùng phương pháp giải mã Base64 để nhận về dạng byte [] ban đầu.

Ví dụ:

```
[
  {
    "maHinhAnh": 1,
    "path": null,
    "tenHinhAnh": null,
    "tenDuyNhat": null,
    "loaiHinhAnh": null,
    "source":
    "/9j/4AAQSkZJRgABAQEAYABgAAD/4QVoRXhpZgAASUkqAAgAAAALAA4BAgAaAAAAkgAAAEZH AwABAAAABQAAA
    E1H AwABAAAAZAAAAJycAQACAgAArAAAAJiCAgAIAAAArgIAAJ2cAQ A+AAAAtgIAADsB AgAqAAA9AIAAJ6cAQB
    sAAAAHgMAAGmHBAABAAAigMAACWIBAABAAA6AQAAWgBAABAAA WgUAAAAAAABbB GJ1bSDhuqNuaCB0aGnDq
    m4gbmhpw6puAFYAYQBwAGEALgB2AG4AIAAtACAASADZHmkAIA BOAGcAaADHHiAAUwApASAATgBoAGkAvx5wAC...
    "
  }
]
```

STT19: Gửi code để xác thực tài khoản (Quên mật khẩu)

POST: {base_url} /identity/forget

Đầu vào: @Body String (username)

Ví dụ:

```
"trangialong"
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"ten dang nhap null"	Nhận về tên đăng nhập null
	"Tên đăng nhập không tồn tại"	Không tồn tại tài khoản trong hệ thống có tên đăng nhập phù hợp
	String maXacNhan (Mã xác nhận tài khoản)	Thành công
400	e.getMessage()	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gấp phải.

STT20: Xác thực tài khoản đã quên

POST: {base_url} /identity/forget/confirm

Đầu vào: @Body ConfirmEmailRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
tenDangNhap	String	Có	Tên tài khoản để đăng nhập chương trình
maXacNhan	String	Có	Mã xác nhận

Ví dụ:

```
{
    "tenDangNhap": "trangialong",
    "maXacNhan": "444f8dd1-4"
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Tài khoản không tồn tại, vui lòng thử lại"	Tài khoản chưa tạo trong hệ thống
	"Mã xác nhận hết hạn"	Mã xác nhận đã hết hạn (Quá 5 phút)
	"Xác nhận thành công"	Thành công
	"Mã xác nhận không chính xác"	Mã xác nhận nhận được không khớp với trong cơ sở dữ liệu
400	e.getMessage()	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gấp phải.

STT21: Cập nhật mật khẩu mới sau khi xác thực (Quên mật khẩu)

PUT: {base_url} /identity/forget

Đầu vào: @Body TaiKhoanDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
tenDangNhap	String	Có	Tên đăng nhập hệ thống
matKhau	String	Có	Mật khẩu của tài khoản
quyen	String	Không	Phân quyền của tài khoản trên hệ thống
trangThai	Boolean	Không	Trạng thái hoạt động của tài khoản: 0: Khóa 1: Đang hoạt động
maXacNhan	String	Không	Mã xác thực của tài khoản

thoiGianTaoMa	LocalDateTime	Không	Thời gian khi tạo mã xác thực
thoiGianHetHan	LocalDateTime	Không	Thời gian hết hạn của mã. Thường là thoiGianTaoMa + 5 phút
thoiGianXacNhan	LocalDateTime	Không	Thời gian xác thực mã

Ví dụ:

```
{
    "tenDangNhap": "customer",
    "matKhau": "mkmoi",
    "quyen": "khachhang",
    "trangThai": true,
    "maXacNhan": null,
    "thoiGianTaoMa": null,
    "thoiGianHetHan": null,
    "thoiGianXacNhan": null
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Cập nhật thành công"	Thành công
400	"Tài khoản không tồn tại"	Không có tài khoản trong hệ thống
	"Cập nhật thất bại"	Lưu thông tin mật khẩu tài khoản thất bại

3. Center Service

a. Bảng API

STT	Path	Method	Yêu cầu đầu vào	Mô tả
1	center/banggia	Get	Không	Lấy danh sách bảng giá
2	center/banggia /apply/{id}	Get	long	Áp dụng bảng giá trên một chi nhánh
3	center/banggia	Post	BangGiaDTO	Thêm bảng giá
4	center/banggia	Put	BangGiaDTO	Sửa thông tin bảng giá
5	center/banggia/{id}	Delete	long	Xóa bảng giá
6	center/ct-san-pham	Get	Không	Lấy các thông tin sản phẩm bày bán
7	center/ct-san-pham /bang-gia	Get	Không	Lấy chi tiết sản phẩm trong bảng giá
8	center/ct-san-pham	Post	List<BangGiaSan PhamRequest>	Cập nhật chi tiết giá trong bảng giá sản phẩm
9	center/ct-san-pham/upload	Post	long	Upload hàng loạt sản phẩm trong chi nhánh lên bảng giá
10	center/ct-thu-cung	Get	Không	Lấy các thông tin thú cung bày bán
11	center/ct-thu-cung/bang-gia	Get	Không	Lấy chi tiết thú cung trong bảng giá
12	center/ct-thu-cung	Post	List<BangGiaThu CungRequest>	Cập nhật chi tiết giá trong bảng giá thú cung
13	center/ct-thu-cung/upload	Post	long	Upload hàng loạt thú cung trong chi nhánh lên bảng giá
14	center/chinhanh	Get	Không	Lấy danh sách chi nhánh
15	center/chinhanh/{id}	Get	int	Lấy thông tin chi nhánh
16	center/chinhanh	Post	String	Thêm chi nhánh
17	center/chinhanh	Put	ChiNhanhDTO	Sửa tên chi nhánh
18	center/chinhanh/{id}	Delete	int	Xóa chi nhánh
19	center/gio-hang/thu-cung	Post	GioHangRequest	Lấy thú cưng trong giỏ hàng
20	center/gio-hang/san-pham	Post	GioHangRequest	Lấy sản phẩm trong giỏ hàng
21	center/gio-hang/them-thu-cung	Post	GhThuCungRequest	Thêm thú cưng vào giỏ
22	center/gio-hang/them-san-pham	Post	GhSanPhamRequest	Thêm sản phẩm vào giỏ
23	center/gio-hang/bo-thu-cung	Post	GhThuCungRequest	Bỏ thú cưng khỏi giỏ hàng
24	center/gio-hang/bo-san-pham	Post	GhSanPhamRequest	BỎ sản phẩm khỏi giỏ hàng

25	center/gio-hang/bo-tat-ca	Post	GioHangRequest	Bỏ tất cả sản phẩm, thú cưng ra khỏi giỏ hàng
26	center/giong	Get	Không	Lấy danh sách giống
27	center/giong	Post	GiongDTO	Thêm giống
28	center/giong	Put	GiongDTO	Sửa giống
29	center/giong/{id}	Delete	int	Xóa giống
30	center/hinhanh	Post	RequestParam: image - MultipartFile[] maNhanVien - String maKhachHang – String maThuCung – long maSanPham – long	Lưu hình ảnh
31	center/hinhanh/get	Post	long[]	Lấy hình ảnh
32	center/loaisanpham	Get	Không	Lấy danh sách loại sản phẩm
33	center/loaisanpham/{id}	Get	int	Lấy loại sản phẩm cụ thể
34	center/loaisanpham	Post	String	Thêm loại sản phẩm
35	center/loaisanpham	Put	LoaiSanPhamDTO	Sửa tên loại sản phẩm
36	center/loaisanpham/{id}	Delete	int	Xóa loại sản phẩm
37	center/loaithucung	Get	Không	Lấy danh sách loại thú cưng
38	center/loaithucung/{id}	Get	int	Lấy loại thú cưng cụ thể
39	center/loaithucung	Post	String	Thêm loại thú cưng
40	center/loaithucung	Put	LoaiThuCungDTO	Sửa loại thú cưng
41	center/loaithucung/{id}	Delete	int	Xóa loại thú cưng
42	center/nhaphang	Get	Không	Lấy danh sách đơn nhập
43	center/nhaphang/tao-phieu-nhap	Post	DonNhapHangDTO	Tạo phiếu nhập hàng
44	center/nhaphang/them-san-pham	Post	List<DonNhapSanPhamRequest>	Thêm chi tiết phiếu nhập hàng sản phẩm
45	center/nhaphang/them-thu-cung	Post	List<DonNhapThuCungRequest>	Thêm chi tiết phiếu nhập hàng thú cưng
46	center/sanpham	Get	Không	Lấy danh sách sản phẩm
47	center/sanpham/{sanpham}/{chinhanh}	Get	Long, int	Lấy thông tin sản phẩm cụ thể
48	center/sanpham	Post	SanPhamDTO	Thêm sản phẩm
49	center/sanpham	Put	SanPhamDTO	Sửa sản phẩm
50	center/sanpham/{sanpham}/{chinhanh}	Delete	Long, int	Xóa sản phẩm

51	center/sanpham /soluongton	Put	SanPhamDTO	Cập nhật số lượng tồn
52	center/thucung	Get	Không	Lấy danh sách thú cưng
53	center/thucung/{ id }	Get	long	Lấy thú cưng cụ thể
54	center/thucung	Post	ThuCungDTO	Thêm thú cưng
55	center/thucung	Put	ThuCungDTO	Sửa thú cưng
56	center/thucung/{ id }	Delete	long	Xóa thú cưng
57	center/thucung/soluongton	Put	ThuCungDTO	Cập nhật số lượng tồn

b. Chi tiết:

STT1: Lấy danh sách bảng giá

GET: {base_url} /center/banggia

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<BangGiaDTO>	Thành công

BangGiaDTO Body:

Thuộc tính	Kiểu	Mô tả
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc chương trình
noiDung	String	Nội dung chương trình/ đợt khuyến mãi
trangThai	Boolean	Trạng thái áp dụng: 1 – Đang áp dụng 0 – Không áp dụng
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh của bảng giá

Ví dụ:

```
[
  {
    "maBangGia": 1,
    "thoiGianBatDau": "2024-05-01T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-05-14T17:00:00.000+00:00",
    "noiDung": "Giảm giá 20% tháng nam sale 20%",
    "trangThai": false,
    "chiNhanh": {
      "maChiNhanh": 1,
      "tenChiNhanh": "Man Thiện"
    }
  },
  {
    "maBangGia": 2,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-20T17:00:00.000+00:00",
    "noiDung": "Tết natal",
    "trangThai": false,
    "chiNhanh": {
      "maChiNhanh": 1,
      "tenChiNhanh": "Man Thiện"
    }
  }
]
```

STT2: Lấy bảng giá cụ thể

GET: {base_url} /center/banggia/{id}

Đầu vào: @Path long id

Ví dụ: {base_url} /center/banggia/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	BangGiaDTO	Thành công
400	"Mã bảng giá không tồn tại!"	Không tồn tại bảng giá có id thỏa mãn

BangGiaDTO Body:

Thuộc tính	Kiểu	Mô tả
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc chương trình
noidung	String	Nội dung chương trình/ đợt khuyến mãi
trangThai	Boolean	Trạng thái áp dụng: 1 – Đang áp dụng 0 – Không áp dụng
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh của bảng giá

Ví dụ:

```
{
    "maBangGia": 1,
    "thoiGianBatDau": "2024-05-01T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-05-14T17:00:00.000+00:00",
    "noidung": "Chương trình khuyến mãi tháng 5",
    "trangThai": true,
    "chiNhanh": {
        "id": 1,
        "ten": "Chi Nhánh A"
    }
}
```

```

    "noiDung": "Giảm giá đặc biệt tháng nam sale 20%",
    "trangThai": false,
    "chiNhanh": {
        "maChiNhanh": 1,
        "tenChiNhanh": "Man Thiện"
    }
}

```

STT3: Thêm bảng giá

POST {base_url} /center/banggia

Đầu vào: @Body BangGiaDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maBangGia	long	Không	Mã bảng giá
thoiGianBatDau	Timestamp	Có	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Có	Thời gian kết thúc chương trình
noiDung	String	Có	Nội dung chương trình/ đợt khuyến mãi
trangThai	Boolean	Có	Trạng thái áp dụng: 1 – Đang áp dụng 0 – Không áp dụng
chiNhanh	ChiNhanhDTO	Có	Thông tin chi nhánh của bảng giá

Ví dụ:

```
{
    "maBangGia": null,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-20T17:00:00.000+00:00",
    "noiDung": "Sinh nhật",
    "trangThai": false,
    "chiNhanh": {
        "maChiNhanh": 1,
        "tenChiNhanh": "Man Thiện"
    }
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	BangGiaDTO	Thành công
400	"Thêm thất bại"	Không tồn tại bảng giá sau khi tạo

BangGiaDTO Body:

Thuộc tính	Kiểu	Mô tả
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc chương trình
noidung	String	Nội dung chương trình/ đợt khuyến mãi
trangThai	Boolean	Trạng thái áp dụng: 1 – Đang áp dụng 0 – Không áp dụng
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh của bảng giá

Ví dụ:

```
{
    "maBangGia": 2,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-20T17:00:00.000+00:00",
    "noidung": "Sinh nhật",
    "trangThai": false,
    "chiNhanh": {
        "maChiNhanh": 1,
        "tenChiNhanh": "Man Thiện"
    }
}
```

STT4: Sửa bảng giá

PUT: {base_url} /center/banggia

Đầu vào: @Body BangGiaDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maBangGia	long	Có	Mã bảng giá
thoiGianBatDau	Timestamp	Có	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Có	Thời gian kết thúc chương trình
noiDung	String	Có	Nội dung chương trình/ đợt khuyến mãi
trangThai	Boolean	Có	Trạng thái áp dụng: 1 – Đang áp dụng 0 – Không áp dụng
chiNhanh	ChiNhanhDTO	Có	Thông tin chi nhánh của bảng giá

Ví dụ:

```
{
    "maBangGia": 2,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-20T17:00:00.000+00:00",
    "noiDung": "Sinh nhật tháng 6",
    "trangThai": true,
    "chiNhanh": {
        "maChiNhanh": 1,
        "tenChiNhanh": "Man Thiện"
    }
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	BangGiaDTO	Thành công
400	"Bảng giá không tồn tại"	Không tồn tại bảng giá trong hệ thống

BangGiaDTO Body:

Thuộc tính	Kiểu	Mô tả
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc chương trình
noiDung	String	Nội dung chương trình/ đợt khuyến mãi
trangThai	Boolean	Trạng thái áp dụng: 1 – Đang áp dụng 0 – Không áp dụng
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh của bảng giá

Ví dụ:

```
{  
    "maBangGia": 2,  
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",  
    "thoiGianKetThuc": "2024-06-20T17:00:00.000+00:00",  
    "noiDung": "Sinh nhật tháng 6",  
    "trangThai": true,  
    "chiNhanh": {  
        "maChiNhanh": 1,  
        "tenChiNhanh": "Man Thiện"  
    }  
}
```

STT5: Xóa bảng giá

DELETE: {base_url} /center/banggia/{id}

Đầu vào: @Path long id

Ví dụ: {base_url} /center/banggia/1

Đầu ra:

Code	Kết quả	Nguyên nhân
400	"Bảng giá không tồn tại"	Không tồn tại bảng giá có id thỏa trong hệ thống
	"Xóa thất bại"	Bảng giá vẫn tồn tại sau khi có thao tác xóa
200	"Xóa thành công"	Thành công

STT6: Lấy thông tin các sản phẩm bày bán

GET: {base_url} /center/ct-san-pham

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<BangGiaSanPhamDTO>	Thành công
400	"Không có dữ liệu"	Không có danh sách sản phẩm

BangGiaSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
maLoaiSanPham	int	Mã loại sản phẩm
tenLoaiSanPham	String	Tên loại sản phẩm
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc

maChiNhanh	int	Mã chi nhánh của sản phẩm
tenChiNhanh	String	Tên chi nhánh
giaHienTai	BigDecimal	Giá hiện tại sản phẩm
giaKhuyenMai	BigDecimal	Giá khuyến mãi (nếu có đợt)
soLuongTon	long	Số lượng tồn sản phẩm
hinhAnh	byte[]	Hình ảnh dạng byte của sản phẩm

Ví dụ:

```
[
  {
    "maSanPham": 3,
    "tenSanPham": "Thức ăn hạt mềm zenith",
    "maLoaiSanPham": 1,
    "tenLoaiSanPham": "Đồ ăn",
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 1,
    "tenChiNhanh": "Man Thiện",
    "giaHienTai": null,
    "giaKhuyenMai": null,
    "soLuongTon": 1,
    "hinhAnh": null
  },
  {
    "maSanPham": 4,
    "tenSanPham": "Thức ăn hạt ANF",
    "maLoaiSanPham": 1,
    "tenLoaiSanPham": "Đồ ăn",
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 1,
    "tenChiNhanh": "Man Thiện",
    "giaHienTai": 100000.0000,
    "giaKhuyenMai": 100000.0000,
    "soLuongTon": 62,
    "hinhAnh": null
  }
]
```

STT7: Lấy thông tin chi tiết sản phẩm trong bản giá

GET: {base_url} /center/ct-san-pham/bang-gia

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<BangGiaSanPhamDTO>	Thành công
400	"Không có dữ liệu"	Không có danh sách chi tiết sản phẩm

BangGiaSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
maLoaiSanPham	int	Mã loại sản phẩm
tenLoaiSanPham	String	Tên loại sản phẩm
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc
maChiNhanh	int	Mã chi nhánh của sản phẩm
tenChiNhanh	String	Tên chi nhánh
giaHienTai	BigDecimal	Giá hiện tại sản phẩm
giaKhuyenMai	BigDecimal	Giá khuyến mãi (nếu có đợt)
soLuongTon	long	Số lượng tồn sản phẩm

hinhAnh	byte[]	Hình ảnh dạng byte của sản phẩm
---------	--------	---------------------------------

Ví dụ:

```
[
  {
    "maSanPham": 3,
    "tenSanPham": "Thúc ăn hạt mềm zenith",
    "maLoaiSanPham": 1,
    "tenLoaiSanPham": "Đồ ăn",
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 1,
    "tenChiNhanh": "Man Thiện",
    "giaHienTai": null,
    "giaKhuyenMai": null,
    "soLuongTon": 1,
    "hinhAnh": null
  },
  {
    "maSanPham": 4,
    "tenSanPham": "Thúc ăn hạt ANF",
    "maLoaiSanPham": 1,
    "tenLoaiSanPham": "Đồ ăn",
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 1,
    "tenChiNhanh": "Man Thiện",
    "giaHienTai": 100000.0000,
    "giaKhuyenMai": 100000.0000,
    "soLuongTon": 62,
    "hinhAnh": null
  }
]
```

STT8: Đổi chi tiết giá sản phẩm trong bảng giá

POST: {base_url} /center/ct-san-pham

Đầu vào: @Body List<BangGiaSanPhamRequest>

BangGiaSanPhamRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maBangGia	long	Có	Mã bảng giá
maSanPham	long	Có	Mã sản phẩm
donGia	BigDecimal	Có	donGia

Ví dụ:

```
[ {  
    "maBangGia": 7,  
    "maSanPham": 7,  
    "donGia": 100.000  
, {  
    "maBangGia": 7,  
    "maSanPham": 8,  
    "donGia": 150.000  
}]
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Cập nhật thành công"	Thành công
400	"Cập nhật thất bại"	Bị lỗi try catch cả hàm

STT9: Upload hàng loạt sản phẩm lên bảng giá

POST: {base_url} /center/ct-san-pham/upload

Đầu vào: @Body long (maBangGia)

Ví dụ: 1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Upload thành công"	Thành công
400	"Không nhận được mã bảng giá"	Mã bảng giá null hoặc bằng 0
	"Upload thất bại"	Lỗi trong lưu chi tiết

STT10: Lấy thông tin các thú cưng bày bán

GET: {base_url} /center/ct-thu-cung

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List< BangGiaThuCungDTO >	Thành công
400	"Không có dữ liệu"	Không có danh sách thú cưng bày bán

BangGiaSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc
maChiNhanh	int	Mã chi nhánh
tenChiNhanh	String	Tên chi nhánh
maThuCung	long	Mã thú cưng
tenThuCung	String	Tên thú cưng
moTa	String	Mô tả

maGiong	int	Mã giống
tenGiong	String	Tên giống
giaHienTai	BigDecimal	Giá hiện tại
giaKhuyenMai	BigDecimal	Giá khuyến mãi (nếu có)
soLuongTon	int	Số lượng tồn khi bán theo bầy/ bê
hinhAnh	byte[]	Hình ảnh thú cưng dạng byte

Ví dụ:

```
[
  {
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 1,
    "tenChiNhanh": "Man Thiện",
    "maThuCung": 1,
    "tenThuCung": "Chó Husky",
    "moTa": "Husky 3 tháng tuổi, màu nâu nh?", 
    "maGiong": 1,
    "tenGiong": "Husky",
    "giaHienTai": 10000000.0000,
    "giaKhuyenMai": 10000000.0000,
    "soLuongTon": 1,
    "hinhAnh": null
  },
  {
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 2,
    "tenChiNhanh": "Trần Thị Hoa",
    "maThuCung": 2,
    "tenThuCung": "Chó Chihuahua",
    "moTa": "Chihuahua 2 tháng tuổi, 4kg",
    "maGiong": 2,
    "tenGiong": "Chihuahua",
    "giaHienTai": 5000000.0000,
    "giaKhuyenMai": 5000000.0000,
    "soLuongTon": 1,
  }
]
```

```

        "hinhAnh": null
    } ]

```

STT11: Lấy thông tin chi tiết thú cung trong bản giá

GET: {base_url} /center/ct-thu-cung/bang-gia

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<BangGiaSanPhamDTO>	Thành công
400	"Không có dữ liệu"	Không có danh sách chi tiết thú cung

BangGiaThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maBangGia	long	Mã bảng giá
thoiGianBatDau	Timestamp	Thời gian bắt đầu áp dụng
thoiGianKetThuc	Timestamp	Thời gian kết thúc
maChiNhanh	int	Mã chi nhánh
tenChiNhanh	String	Tên chi nhánh
maThuCung	long	Mã thú cung
tenThuCung	String	Tên thú cung
moTa	String	Mô tả
maGiong	int	Mã giống
tenGiong	String	Tên giống
giaHienTai	BigDecimal	Giá hiện tại

giaKhuyenMai	BigDecimal	Giá khuyến mãi (nếu có)
soLuongTon	int	Số lượng tồn khi bán theo bầy/ bể
hinhAnh	byte[]	Hình ảnh thú cưng dạng byte

Ví dụ:

```
[
  {
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 1,
    "tenChiNhanh": "Man Thiện",
    "maThuCung": 1,
    "tenThuCung": "Chó Husky",
    "moTa": "Husky 3 tháng tuổi, màu nâu nh?", 
    "maGiong": 1,
    "tenGiong": "Husky",
    "giaHienTai": 10000000.0000,
    "giaKhuyenMai": 10000000.0000,
    "soLuongTon": 1,
    "hinhAnh": null
  },
  {
    "maBangGia": 4,
    "thoiGianBatDau": "2024-06-10T17:00:00.000+00:00",
    "thoiGianKetThuc": "2024-06-27T17:00:00.000+00:00",
    "maChiNhanh": 2,
    "tenChiNhanh": "Trần Thị Hoa",
    "maThuCung": 2,
    "tenThuCung": "Chó Chihuahua",
    "moTa": "Chihuahua 2 tháng tuổi, 4kg",
    "maGiong": 2,
    "tenGiong": "Chihuahua",
    "giaHienTai": 5000000.0000,
    "giaKhuyenMai": 5000000.0000,
    "soLuongTon": 1,
    "hinhAnh": null
  }
]
```

STT12: Đổi chi tiết giá thú cung trong bảng giá

POST: {base_url} /center/ct-thu-cung

Đầu vào: @Body List<BangGiaThuCungRequest>

BangGiaThuCungRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maBangGia	long	Có	Mã bảng giá
maThuCung	long	Có	Mã thú cung
donGia	BigDecimal	Có	donGia

Ví dụ:

```
[ {
    "maBangGia": 7,
    "maThuCung": 7,
    "donGia": 100.000
}, {
    "maBangGia": 7,
    "maThuCung": 8,
    "donGia": 150.000
}]
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Cập nhật thành công"	Thành công
400	"Cập nhật thất bại"	Bị lỗi try catch cả hàm

STT13: Upload hàng loạt thú cung lên bảng giá

POST: {base_url} /center/ct-thu-cung/upload

Đầu vào: @Body long (maBangGia)

Ví dụ: 1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Upload thành công"	Thành công
400	"Không nhận được mã bảng giá"	Mã bảng giá null hoặc bằng 0
	"Upload thất bại"	Lỗi trong lưu chi tiết

STT14: Lấy danh sách chi nhánh

GET: {base_url} /center/chinhanh

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<ChiNhanhDTO>	Thành công

ChiNhanhDTO Body:

Thuộc tính	Kiểu	Mô tả
maChiNhanh	int	Mã chi nhánh
tenChiNhanh	String	Tên chi nhánh

Ví dụ:

```
[  
  {  
    "maChiNhanh": 1,  
    "tenChiNhanh": "Man Thiện"  
  },  
  {  
    "maChiNhanh": 2,  
    "tenChiNhanh": "Trần Thị Hoa"  
  }  
]
```

STT15: Lấy chi nhánh cụ thể

GET: {base_url} /center/chinhanh/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/chinhanh/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	ChiNhanhDTO	Thành công
404	"Entity not found"	Không có chi nhánh

ChiNhanhDTO Body:

Thuộc tính	Kiểu	Mô tả
maChiNhanh	int	Mã chi nhánh
tenChiNhanh	String	Tên chi nhánh

Ví dụ:

```
{  
    "maChiNhanh": 1,  
    "tenChiNhanh": "Man Thiện"  
}
```

STT16: Thêm chi nhánh

POST: {base_url} /center/chinhanh

Đầu vào: @Body String (tenChiNhanh)

Ví dụ: Man Thiệp

Đầu ra:

Code	Kết quả	Nguyên nhân
200	ChiNhanhDTO	Thành công
400	BAD_REQUEST	Chi nhánh tạo bị null hoặc có mã chi nhánh từ 0 trở xuống
409	"Entity is existed"	Đã tồn tại tên chi nhánh trong hệ thống

ChiNhanhDTO Body:

Thuộc tính	Kiểu	Mô tả
maChiNhanh	int	Mã chi nhánh
tenChiNhanh	String	Tên chi nhánh

Ví dụ:

```
{  
    "maChiNhanh": 1,  
    "tenChiNhanh": "Man Thiện"  
}
```

STT17: Sửa chi nhánh

PUT: {base_url} /center/chinhanh

Đầu vào: @Body ChiNhanhDTO

ChiNhanhDTO Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maChiNhanh	int	Có	Mã chi nhánh
tenChiNhanh	String	Có	Tên chi nhánh

Ví dụ:

```
{  
    "maChiNhanh": 1,  
    "tenChiNhanh": "97 Man Thiện"
```

}

Đầu ra:

Code	Kết quả	Nguyên nhân
200	ChiNhanhDTO	Thành công
404	"Entity not found"	Không tồn tại chi nhánh trong hệ thống

ChiNhanhDTO Body:

Thuộc tính	Kiểu	Mô tả
maChiNhanh	int	Mã chi nhánh
tenChiNhanh	String	Tên chi nhánh

Ví dụ:

```
{  
    "maChiNhanh": 1,  
    "tenChiNhanh": "Man Thiện"  
}
```

STT18: Xóa chi nhánh

DELETE: {base_url} /center/chinhanh/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/chinhanh/3

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Success to delete entity"	Thành công
400	"Failed to delete entity"	Xóa thất bại, chi nhánh vẫn còn tồn tại
404	"Entity not found"	Không tồn tại chi nhánh trong hệ thống

STT19: Lấy thú cưng trong giỏ hàng

POST: {base_url} /center/gio-hang/thu-cung

Đầu vào: @Body GioHangRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maChiNhanh	int	Có	Mã chi nhánh

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "maChiNhanh": 1
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<ThuCungDTO>	Thành công

ThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maThuCung	long	Mã thú cưng
tenThuCung	String	Tên thú cưng
trangThaiBan	int	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Chủ của thú cưng (nếu có)
moTa	String	Mô tả chi tiết

giaHienTai	BigDecimal	Giá hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh
giong	GiongDTO	Thông tin giống thú cưng
hinhAnh	List<Long>	Danh sách hình ảnh của thú cưng
soLuongTon	int	Số lượng tồn thú cưng

Ví dụ:

```
[
  {
    "maThuCung": 8,
    "tenThuCung": "Tắc kè hoa",
    "trangThaiBan": 0,
    "chu": "Chưa xác định",
    "moTa": "Tắc kè hoa, d? an, kêu thu?ng xuyên",
    "giaHienTai": 200000.0000,
    "giaKM": null,
    "chiNhanh": {
      "maChiNhanh": 1,
      "tenChiNhanh": "Man Thiện"
    },
    "giong": {
      "maGiong": 13,
      "tengiong": "Tắc kè",
      "loaiThuCung": {
        "maLoaiThuCung": 4,
        "tenLoaiThuCung": "Bò sát"
      }
    },
    "hinhAnh": null,
    "soLuongTon": 5
  }
]
```

STT20: Lấy sản phẩm trong giỏ hàng

POST: {base_url} /center/gio-hang/san-pham

Đầu vào: @Body GioHangRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maChiNhanh	int	Có	Mã chi nhánh

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "maChiNhanh": 1
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<SanPhamDTO>	Thành công

SanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
giaHienTai	BigDecimal	Giá bán hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Mã chi nhánh

soLuongTon	long	Số lượng tồn của sản phẩm tại chi nhánh
------------	------	---

Ví dụ:

```
[  
  {  
    "maSanPham": 4,  
    "tenSanPham": "Thức ăn hạt ANF",  
    "giaHienTai": 100000.0000,  
    "giaKM": null,  
    "loaiSanPham": {  
      "maLoaiSanPham": 1,  
      "tenLoaiSanPham": "Đồ ăn"  
    },  
    "hinhAnh": null,  
    "maChiNhanh": 1,  
    "soLuongTon": 62  
  }  
]
```

STT21: Thêm thú cưng vào trong giỏ hàng

POST: {base_url} /center/gio-hang/them-thu-cung

Đầu vào: @Body GhThuCungRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maThuCung	long	Có	Mã thú cưng

Ví dụ:

```
{  
  "maKhachHang": "lamvu2010",  
  "maThuCung": 1  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Thêm thành công"	Thành công
400	"Thêm thất bại" + e.getMessage()	Lỗi thêm thú cưng vào giỏ hàng

STT22: Thêm sản phẩm vào trong giỏ hàng

POST: {base_url} /center/gio-hang/them-san-pham

Đầu vào: @Body GhSanPhamRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maSanPham	long	Có	Mã sản phẩm
maChiNhanh	int	Có	Mã chi nhánh

Ví dụ:

```
{  
    "maKhachHang": "lamvu2010",  
    "maSanPham": 1,  
    "maChiNhanh": 1  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Thêm thành công"	Thành công
400	"Thêm thất bại" + e.getMessage()	Lỗi thêm sản phẩm vào giỏ hàng

STT23: Bỏ thú cưng ra giỏ hàng

POST: {base_url} /center/gio-hang/bo-thu-cung

Đầu vào: @Body GhThuCungRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maThuCung	long	Có	Mã thú cưng

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "maThuCung": 1
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xóa thành công"	Thành công
400	"Giỏ hàng thú cưng không tồn tại"	Trong giỏ hàng người dùng không có thú cưng đã truyền

STT24: Bỏ sản phẩm vào trong giỏ hàng

POST: {base_url} /center/gio-hang/bo-san-pham

Đầu vào: @Body GhSanPhamRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maSanPham	long	Có	Mã sản phẩm
maChiNhanh	int	Có	Mã chi nhánh

Ví dụ:

```
{
    "maKhachHang": "lamvu2010",
    "maSanPham": 1,
    "maChiNhanh": 1
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xóa thành công"	Thành công
400	"Giỏ hàng sản phẩm không tồn tại"	Trong giỏ hàng người dùng không có sản phẩm ứng với chi nhánh đã truyền

STT25: Bỏ tất cả mặt hàng ra khỏi giỏ hàng

POST: {base_url} /center/gio-hang/bo-tat-ca

Đầu vào: @Body GioHangRequest

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maKhachHang	String	Có	Mã khách hàng
maChiNhanh	int	Có	Mã chi nhánh

Ví dụ:

```
{  
    "maKhachHang": "lamvu2010",  
    "maChiNhanh": 1  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xóa thành công"	Thành công
400	"Xóa thất bại"	Lỗi trong việc xóa từng chi tiết sản phẩm và thủ cung

STT26: Lấy danh sách giống

GET: {base_url} /center/giong

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<GiongDTO>	Thành công

GiongDTO Body:

Thuộc tính	Kiểu	Mô tả
maGiong	int	Mã giống
tenGiong	String	Tên giống
loaiThuCung	LoaiThuCungDTO	Thông tin về loại thú cưng

Ví dụ:

```
[  
 {  
     "maGiong": 1,  
     "tengiong": "Husky",  
     "loaiThuCung": {  
         "maLoaiThuCung": 3,  
         "tenLoaiThuCung": "Chó"  
     }  
 } ]
```

STT27: Thêm giống

POST: {base_url} /center/giong

Đầu vào: @Body GiongDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maGiong	int	Không	Mã giống
tenGiong	String	Có	Tên giống
loaiThuCung	LoaiThuCungDTO	Có	Thông tin về loại thú cưng

Ví dụ:

```
{  
    "maGiong": null,  
    "tengiong": "Husky",  
    "loaiThuCung": {  
        "maLoaiThuCung": 3,  
        "tenLoaiThuCung": "Chó"  
    }  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	GiongDTO	Thành công
400	"Thêm thất bại"	Không tồn tại sau khi thêm

GiongDTO Body:

Thuộc tính	Kiểu	Mô tả
maGiong	int	Mã giống
tenGiong	String	Tên giống
loaiThuCung	LoaiThuCungDTO	Thông tin về loại thú cưng

Ví dụ:

```
{  
    "maGiong": 1,  
    "tengiong": "Husky",  
    "loaiThuCung": {  
        "maLoaiThuCung": 3,  
        "tenLoaiThuCung": "Chó"  
    }  
}
```

STT28: Sửa giống

PUT: {base_url} /center/giong

Đầu vào: @Body GiongDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maGiong	int	Có	Mã giống
tenGiong	String	Có	Tên giống
loaiThuCung	LoaiThuCungDTO	Có	Thông tin về loại thú cưng

Ví dụ:

```
{
    "maGiong": 1,
    "tengiong": "Husky",
    "loaiThuCung": {
        "maLoaiThuCung": 3,
        "tenLoaiThuCung": "Chó"
    }
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	GiongDTO	Thành công
400	"Giống không tồn tại"	Giống không tồn tại trong hệ thống

GiongDTO Body:

Thuộc tính	Kiểu	Mô tả
maGiong	int	Mã giống
tenGiong	String	Tên giống
loaiThuCung	LoaiThuCungDTO	Thông tin về loại thú cưng

Ví dụ:

```
{
    "maGiong": 1,
    "tengiong": "Husky",
    "loaiThuCung": {
```

```

        "maLoaiThuCung": 3,
        "tenLoaiThuCung": "Chó"
    }
}

```

STT29: Xóa gióng

DELETE: {base_url} /center/giong/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/giong/3

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xóa thành công"	Thành công
400	"Gióng không tồn tại"	Gióng có id không thỏa tồn tại trong hệ thống

STT30: Lưu hình ảnh thú cưng và sản phẩm

POST: {base_url} /center/hinhanh

Đầu vào:

- + @RequestParam("image") MultipartFile[]
- + @RequestParam("maNhanVien") String
- + @RequestParam("maKhachHang") String
- + @RequestParam("maThuCung") Long
- + @RequestParam("maSanPham") Long

Tham số truyền vào có dạng:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
image	MultipartFile	Có	Nhiều file hình ảnh
maNhanVien	String	Không	Mã nhân viên

maKhachHang	String	Không	Mã khách hàng
maThuCung	Long	Không	Mã thú cưng
maSanPham	Long	Không	Mã sản phẩm

Ví dụ: Lưu 1 hình ảnh của khách hàng lamvu2010.

```
image = Multipart(1),
maNhanVien = null,
maKhachHang = null,
maThuCung = null,
maSanPham = null,
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Lưu thành công"	Thành công
400	"Lưu thất bại"	Có lỗi trong quá trình lưu giữa các hình ảnh

STT31: Lấy về các hình ảnh của thú cưng hoặc sản phẩm

POST: {base_url} /center/hinhanh/get

Đầu vào: @Body long[]

Ví dụ: Lấy hình có id 1, 2 và 3

[1, 2, 3]

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<HinhAnhDTO>	Thành công
400	"Lỗi lấy hình"	Lỗi try catch về dòng lệnh. In ra chi tiết lỗi gấp phai.

HinhAnhDTO Body:

Thuộc tính	Kiểu	Mô tả
maHinhAnh	long	Id hình ảnh đã truyền
path	String	Đường dẫn hình ảnh
tenHinhAnh	String	Tên hình ảnh khi lưu
tenDuyNhat	String	Tên phân biệt giữa các hình ảnh trong thư mục
loaiHinhAnh	String	Loại hình ảnh
source	byte[]	Đoạn byte mã hóa hình ảnh. Khi nhận từ phía client cần nhận bằng String và dùng phương pháp giải mã Base64 để nhận về dạng byte [] ban đầu.

Ví dụ:

```
[  
 {  
   "maHinhAnh": 1,  
   "path": null,  
   "tenHinhAnh": null,  
   "tenDuyNhat": null,  
   "loaiHinhAnh": null,  
   "source":  
     "/9j/4AAQSkZJRgABAQEAYABgAAD/4QVoRXhpZgAASUkqAAgAAAAALAA4BAgAaAAAAkgAAAEZH AwABAAAABQAAA  
     E1H AwABAAAAZAAAAJycAQACAgAArAAAAJiCAgAIAAAArgIAAJ2cAQ A+AAAAtgIAADsB AgAqAAA9AIA AJ6cAQ B  
     sAAAAHgMAAGmHBA BAAA AigMAACWIBA BAAA A6AQAA WgBA BAAA A WgUAAA ABBbGJ1bSDhuqNuaCB0aGnDq  
     m4gbmhpw6puAFYAYQBwAGEALgB2AG4AIAAtACAASADZHmkAIABOAGcAaADHHiAAUwApASAATgBoAGkAvx5wAC...  
   "  
 } ]
```

STT32: Lấy danh sách loại sản phẩm

GET: {base_url} /center/loaisanpham

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<LoaiSanPhamDTO>	Thành công

LoaiSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiSanPham	int	Mã loại sản phẩm
tenLoaiSanPham	String	Tên loại sản phẩm

Ví dụ:

```
[  
  {  
    "maLoaiSanPham": 1,  
    "tenLoaiSanPham": "Đồ ăn"  
  },  
  {  
    "maLoaiSanPham": 2,  
    "tenLoaiSanPham": "Thức uống"  
  }  
]
```

STT33: Lấy loại sản phẩm cụ thể

GET: {base_url} /center/loaisanpham/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/loaisanpham/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoaiSanPhamDTO	Thành công
404	"Entity not found"	Không tìm thấy loại sản phẩm trong hệ thống

LoaiSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiSanPham	int	Mã loại sản phẩm
tenLoaiSanPham	String	Tên loại sản phẩm

Ví dụ:

```
{  
    "maLoaiSanPham": 1,  
    "tenLoaiSanPham": "Đồ ăn"  
}
```

STT34: Thêm loại sản phẩm

POST: {base_url} /center/loaisanpham

Đầu vào: @Body String (tenLoaiSanPham)

Ví dụ: Đồ ăn cho pet

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoaiSanPhamDTO	Thành công
400	"Cannot save entity. Try again"	Loại sản phẩm tạo bị null hoặc có mã từ 0 trở xuống
409	"Entity is existed"	Đã tồn tại tên loại sản phẩm trong hệ thống

LoaiSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiSanPham	int	Mã loại sản phẩm
tenLoaiSanPham	String	Tên loại sản phẩm

Ví dụ:

```
{  
    "maLoaiSanPham": 1,  
    "tenLoaiSanPham": "Đồ ăn"  
}
```

STT35: Sửa loại sản phẩm

PUT: {base_url} /center/loaisanpham

Đầu vào: @Body LoaiSanPhamDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maLoaiSanPham	int	Có	Mã loại sản phẩm
tenLoaiSanPham	String	Có	Tên loại sản phẩm

Ví dụ:

```
{  
    "maLoaiSanPham": 1,  
    "tenLoaiSanPham": "Đồ ăn cho pet"  
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoaiSanPhamDTO	Thành công
404	"Entity not found"	Không tồn tại loại sản phẩm trong hệ thống

LoaiSanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiSanPham	int	Mã loại sản phẩm
tenLoaiSanPham	String	Tên loại sản phẩm

Ví dụ:

```
{  
    "maLoaiSanPham": 1,  
    "tenLoaiSanPham": "Đồ ăn"  
}
```

STT36: Xóa loại sản phẩm

DELETE: {base_url} /center/loaisanpham/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/loaisanpham/3

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Success to delete entity"	Thành công
400	"Failed to delete entity"	Xóa thất bại, loại sản phẩm vẫn còn tồn tại
404	"Entity not found"	Không tồn tại loại sản phẩm trong hệ thống

STT37: Lấy danh sách loại thú cưng

GET: {base_url} /center/loraithucung

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<LoaiThuCungDTO>	Thành công

LoaiThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiThuCung	int	Mã loại thú cưng
tenLoaiThuCung	String	Tên loại thú cưng

Ví dụ:

```
[  
  {  
    "maLoaiThuCung": 2,  
    "tenLoaiThuCung": "Mèo"  
  },  
  {  
    "maLoaiThuCung": 3,  
    "tenLoaiThuCung": "Chó"  
  }  
]
```

STT38: Lấy loại thú cưng cụ thể

GET: {base_url} /center/loaithucung/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/loaithucung/2

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoaiThuCungDTO	Thành công
404	"Entity not found"	Không tìm thấy loại thú cưng trong hệ thống

LoaiThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiThuCung	int	Mã loại thú cưng
tenLoaiThuCung	String	Tên loại thú cưng

Ví dụ:

```
{  
  "maLoaiThuCung": 2,  
  "tenLoaiThuCung": "Mèo"  
}
```

STT39: Thêm loại thú cưng

POST: {base_url} /center/loithucung

Đầu vào: @Body String (tenLoaiThuCung)

Ví dụ: Chó ngao Tây Tạng

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoaiThuCungDTO	Thành công
400		Loại thú cưng tạo bị null hoặc có mã từ 0 trở xuống
409	"Entity is existed"	Đã tồn tại tên loại sản phẩm trong hệ thống

LoaiThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiThuCung	int	Mã loại thú cưng
tenLoaiThuCung	String	Tên loại thú cưng

Ví dụ:

```
{
    "maLoaiThuCung": 2,
    "tenLoaiThuCung": "Mèo"
}
```

STT40: Sửa loại thú cưng

PUT: {base_url} /center/loithucung

Đầu vào: @Body LoaiThuCungDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maLoaiThuCung	int	Có	Mã loại thú cưng

tenLoaiThuCung	String	Có	Tên loại thú cưng
----------------	--------	----	-------------------

Ví dụ:

```
{
    "maLoaiThuCung": 2,
    "tenLoaiThuCung": "Mèo Mung"
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	LoaiThuCungDTO	Thành công
404	"Entity not found"	Không tồn tại loại thú cưng trong hệ thống

LoaiThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maLoaiThuCung	int	Mã loại thú cưng
tenLoaiThuCung	String	Tên loại thú cưng

Ví dụ:

```
{
    "maLoaiThuCung": 2,
    "tenLoaiThuCung": "Mèo Mung"
}
```

STT41: Xóa loại thú cưng

DELETE: {base_url} /center/loaithucung/{id}

Đầu vào: @Path int id

Ví dụ: {base_url} /center/loaithucung/3

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Success to delete entity"	Thành công
400	"Failed to delete entity"	Xóa thất bại, loại thú cưng vẫn còn tồn tại
404	"Entity not found"	Không tồn tại loại thú cưng trong hệ thống

STT42: Lấy danh sách đơn nhập hàng

GET: {base_url} /center/nhaphang

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<DonNhapHangDTO>	Thành công

DonNhapHangDTO Body:

Thuộc tính	Kiểu	Mô tả
maDonNhapHang	long	Mã đơn nhập
ngayLap	Timestamp	Ngày giờ lập đơn
maNhanVien	String	Mã nhân viên lập đơn
chiNhanhDTO	ChiNhanhDTO	Thông tin chi nhánh liên quan

Ví dụ:

```
[  
  {  
    "maDonNhapHang": 1,  
    "ngayLap": "2024-06-10T07:24:50.490+00:00",  
    "maNhanVien": "NV3",  
    "chiNhanhDTO": {  
      "maChiNhanh": 2,  
      "tenChiNhanh": "Trần Thị Hoa"  
    }  
  }  
]
```

```

    }
]
```

STT43: Tao đơn nhập hàng

POST: {base_url} /center/nhaphang/tao-phieu-nhap

Đầu vào: @Body DonNhapHangDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maDonNhapHang	long	Không	Mã đơn nhập
ngayLap	Timestamp	Không	Ngày giờ lập đơn
maNhanVien	String	Có	Mã nhân viên lập đơn
chiNhanhDTO	ChiNhanhDTO	Có	Thông tin chi nhánh liên quan

Ví dụ:

```
{
    "maDonNhapHang": null,
    "ngayLap": null,
    "maNhanVien": "NV3",
    "chiNhanhDTO": {
        "maChiNhanh": 2,
        "tenChiNhanh": "Trần Thị Hoa"
    }
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	DonNhapHangDTO	Thành công
400	"Tạo thất bại" + e.getMessage()	Lỗi tạo đơn. Đính kèm chi tiết lỗi trên thông báo.

DonNhapHangDTO Body:

Thuộc tính	Kiểu	Mô tả
maDonNhapHang	long	Mã đơn nhập
ngayLap	Timestamp	Ngày giờ lập đơn
maNhanVien	String	Mã nhân viên lập đơn
chiNhanhDTO	ChiNhanhDTO	Thông tin chi nhánh liên quan

Ví dụ:

```
{
    "maDonNhapHang": 1,
    "ngayLap": "2024-06-10T07:24:50.490+00:00",
    "maNhanVien": "NV3",
    "chiNhanhDTO": {
        "maChiNhanh": 2,
        "tenChiNhanh": "Trần Thị Hoa"
    }
}
```

STT44: Thêm chi tiết sản phẩm vào phiếu nhập

POST: {base_url} /center/nhaphang/them-san-pham

Đầu vào: @Body List<DonNhapSanPhamRequest>

DonNhapSanPhamRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maDonNhapHang	long	Có	Mã đơn nhập
maSanPham	long	Có	Mã sản phẩm
soLuong	int	Có	Số lượng nhập sản phẩm
donGia	BigDecimal	Có	Đơn giá nhập sản phẩm

Ví dụ:

```
[      {
    "maDonNhapHang": 1,
    "maSanPham": 1,
    "soLuong": 1,
    "donGia": 100.000},
{
    "maDonNhapHang": 1,
    "maSanPham": 2,
    "soLuong": 1,
    "donGia": 200.000}

]
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Thêm thành công"	Thành công
400	e.getMessage()	Lỗi thêm chi tiết. Kèm chi tiết lỗi mã lệnh.

STT45: Thêm chi tiết thú cưng vào phiếu nhập

POST: {base_url} /center/nhaphang/them-thu-cung

Đầu vào: @Body List<DonNhapThuCungRequest>

DonNhapThuCungRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maDonNhapHang	long	Có	Mã đơn nhập
maThuCung	long	Có	Mã thú cưng
giaNhap	BigDecimal	Có	Giá nhập mỗi con
soLuong	int	Có	Số lượng nhập thú cưng (có thể theo bầy/bé)

Ví dụ:

```
[    {
        "maDonNhapHang": 1,
        "maThuCung": 1,
        "giaNhap": 100.000,
        "soLuong": 1
    },
    {
        "maDonNhapHang": 1,
        "maThuCung": 2,
        "giaNhap": 200.000,
        "soLuong": 2
    }
]
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Thêm thành công"	Thành công
400	e.getMessage()	Lỗi thêm chi tiết. Kèm chi tiết lỗi mã lệnh.

STT46: Lấy danh sách sản phẩm

GET: {base_url} /center/sanpham

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<SanPhamDTO>	Thành công

SanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
giaHienTai	BigDecimal	Giá bán hiện tại

giaKM	BigDecimal	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Mã chi nhánh
soLuongTon	long	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
[
  {
    "maSanPham": 4,
    "tenSanPham": "Thức ăn hạt ANF",
    "giaHienTai": 100000.0000,
    "giaKM": null,
    "loaiSanPham": {
      "maLoaiSanPham": 1,
      "tenLoaiSanPham": "Đồ ăn"
    },
    "hinhAnh": null,
    "maChiNhanh": 1,
    "soLuongTon": 62
  }
]
```

STT47: Lấy sản phẩm cụ thể

GET: {base_url} /center/sanpham/{sanpham}/{chinhanh}

Đầu vào: @Path Long sanpham, @Path int chinhanh

Ví dụ: {base_url} /center/sanpham/1/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	SanPhamDTO	Thành công
404	"Id khong ton tai"	Không tồn tại một chi tiết sản phẩm (sản phẩm và chi nhánh) trong hệ thống

SanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
giaHienTai	BigDecimal	Giá bán hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Mã chi nhánh
soLuongTon	long	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
{  
    "maSanPham": 1,  
    "tenSanPham": "Thức ăn hạt ANF",  
    "giaHienTai": 100000.0000,  
    "giaKM": null,  
    "loaiSanPham": {  
        "maLoaiSanPham": 1,  
        "tenLoaiSanPham": "Đồ ăn"  
    },  
    "hinhAnh": null,  
    "maChiNhanh": 1,  
    "soLuongTon": 62  
}
```

STT48: Thêm sản phẩm

POST: {base_url} /center/sanpham

Đầu vào: @Body SanPhamDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maSanPham	long	Không	Mã sản phẩm
tenSanPham	String	Có	Tên sản phẩm
giaHienTai	BigDecimal	Có	Giá bán hiện tại
giaKM	BigDecimal	Không	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Có	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Không	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Có	Mã chi nhánh
soLuongTon	long	Có	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
{
    "maSanPham": null,
    "tenSanPham": "Thức ăn hạt ANF",
    "giaHienTai": 100000.0000,
    "giaKM": null,
    "loaiSanPham": {
        "maLoaiSanPham": 1,
        "tenLoaiSanPham": "Đồ ăn"
    },
    "hinhAnh": null,
    "maChiNhanh": 1,
    "soLuongTon": 100
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	SanPhamDTO	Thành công
400	"Chi nhánh không tồn tại"	Không có dữ liệu về chi nhánh trong hệ thống

	"Thêm thất bại"	Lỗi thêm và lưu thông tin sản phẩm
--	-----------------	------------------------------------

SanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
giaHienTai	BigDecimal	Giá bán hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Mã chi nhánh
soLuongTon	long	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
{
    "maSanPham": null,
    "tenSanPham": "Thức ăn hạt ANF",
    "giaHienTai": 100000.0000,
    "giaKM": null,
    "loaiSanPham": {
        "maLoaiSanPham": 1,
        "tenLoaiSanPham": "Đồ ăn"
    },
    "hinhAnh": null,
    "maChiNhanh": 1,
    "soLuongTon": 100
}
```

STT49: Sửa sản phẩm

PUT: {base_url} /center/sanpham

Đầu vào: @Body SanPhamDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maSanPham	long	Có	Mã sản phẩm
tenSanPham	String	Có	Tên sản phẩm
giaHienTai	BigDecimal	Có	Giá bán hiện tại
giaKM	BigDecimal	Không	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Có	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Không	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Có	Mã chi nhánh
soLuongTon	long	Không	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
{
    "maSanPham": 1,
    "tenSanPham": "Thúc ăn hạt ANF",
    "giaHienTai": 200000.0000,
    "giaKM": null,
    "loaiSanPham": {
        "maLoaiSanPham": 1,
        "tenLoaiSanPham": "Đồ ăn"
    },
    "hinhAnh": null,
    "maChiNhanh": 2,
    "soLuongTon": 100
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	SanPhamDTO	Thành công
400	SanPhamDTO (Empty)	Không tồn tại sản phẩm phù hợp mã sản phẩm

SanPhamDTO Body:

Thuộc tính	Kiểu	Mô tả
maSanPham	long	Mã sản phẩm
tenSanPham	String	Tên sản phẩm
giaHienTai	BigDecimal	Giá bán hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Mã chi nhánh
soLuongTon	long	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
{  
    "maSanPham": 1,  
    "tenSanPham": "Thức ăn hạt ANF",  
    "giaHienTai": 200000.0000,  
    "giaKM": null,  
    "loaiSanPham": {  
        "maLoaiSanPham": 1,  
        "tenLoaiSanPham": "Đồ ăn"  
    },  
    "hinhAnh": null,  
    "maChiNhanh": 2,  
    "soLuongTon": 100  
}
```

STT50: Xóa sản phẩm

DELETE: {base_url} /center/sanpham/{ sanpham }/{ chinhanh }

Đầu vào: @Path Long sanpham, @Path int chinhanh

Ví dụ: {base_url} /center/sanpham/1/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xoa thanh cong"	Thành công
400	"Id khong ton tai"	Trong chi nhánh không có id sản phẩm được tạo hoặc không có chi nhánh nào phù hợp.
	"Xoa that bai"	Sản phẩm vẫn còn tồn tại sau khi xóa

STT51: Cập nhật số lượng tồn sản phẩm

PUT: {base_url} /center/sanpham/soluongton

Đầu vào: @Body SanPhamDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maSanPham	long	Có	Mã sản phẩm
tenSanPham	String	Không	Tên sản phẩm
giaHienTai	BigDecimal	Không	Giá bán hiện tại
giaKM	BigDecimal	Không	Giá khuyến mãi (nếu có)
loaiSanPham	LoaiSanPhamDTO	Không	Thông tin loại của sản phẩm
hinhAnh	List<Long>	Không	Danh sách id hình ảnh của sản phẩm
maChiNhanh	int	Có	Mã chi nhánh
soLuongTon	long	Có	Số lượng tồn của sản phẩm tại chi nhánh

Ví dụ:

```
{
    "maSanPham": 1,
    "tenSanPham": null,
    "giaHienTai": null,
    "giaKM": null,
    "loaiSanPham": null,
```

```

        "hinhAnh": null,
        "maChiNhanh": 2,
        "soLuongTon": 100
    }
}

```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Cập nhật thành công"	Thành công
400	"Cập nhật thất bại"	Lưu chi tiết số lượng tồn ở chi tiết sản phẩm thất bại

STT52: Lấy danh sách thú cưng

GET: {base_url} /center/thucung

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<ThuCungDTO>	Thành công

ThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maThuCung	long	Mã thú cưng
tenThuCung	String	Tên thú cưng
trangThaiBan	int	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Chủ của thú cưng (nếu có)
moTa	String	Mô tả chi tiết

giaHienTai	BigDecimal	Giá hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh
giong	GiongDTO	Thông tin giống thú cưng
hinhAnh	List<Long>	Danh sách hình ảnh của thú cưng
soLuongTon	int	Số lượng tồn thú cưng

Ví dụ:

```
[
  {
    "maThuCung": 8,
    "tenThuCung": "Tắc kè hoa",
    "trangThaiBan": 0,
    "chu": "Chưa xác định",
    "moTa": "Tắc kè hoa, d? an, kêu thu?ng xuyên",
    "giaHienTai": 200000.0000,
    "giaKM": null,
    "chiNhanh": {
      "maChiNhanh": 1,
      "tenChiNhanh": "Man Thiện"
    },
    "giong": {
      "maGiong": 13,
      "tengiong": "Tắc kè",
      "loaiThuCung": {
        "maLoaiThuCung": 4,
        "tenLoaiThuCung": "Bò sát"
      }
    },
    "hinhAnh": null,
    "soLuongTon": 5
  }
]
```

STT53: Lấy thú cưng cụ thể

GET: {base_url} /center/thucung/{id}

Đầu vào: @Path long id

Ví dụ: {base_url} /center/thucung/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	ThuCungDTO	Thành công
404	"Thú cưng không tồn tại"	Không tồn tại thú cưng có id thỏa mãn trong hệ thống

ThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maThuCung	long	Mã thú cưng
tenThuCung	String	Tên thú cưng
trangThaiBan	int	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Chủ của thú cưng (nếu có)
moTa	String	Mô tả chi tiết
giaHienTai	BigDecimal	Giá hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh
giong	GiongDTO	Thông tin giống thú cưng
hinhAnh	List<Long>	Danh sách hình ảnh của thú cưng
soLuongTon	int	Số lượng tồn thú cưng

Ví dụ:

```
{  
    "maThuCung": 8,  
    "tenThuCung": "Tắc kè hoa",  
    "trangThaiBan": 0,  
    "chu": "Chưa xác định",  
    "moTa": "Tắc kè hoa, d? an, kêu thu?ng xuyên",  
    "giaHienTai": 200000.0000,  
    "giaKM": null,  
    "chiNhanh": {  
        "maChiNhanh": 1,  
        "tenChiNhanh": "Man Thiện"  
    },  
    "gioong": {  
        "maGiong": 13,  
        "tengioong": "Tắc kè",  
        "loaiThuCung": {  
            "maLoaiThuCung": 4,  
            "tenLoaiThuCung": "Bò sát"  
        }  
    },  
    "hinhAnh": null,  
    "soLuongTon": 5  
}
```

STT54: Thêm thú cưng

POST: {base_url} /center/thucung

Đầu vào: @Body ThuCungDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maThuCung	long	Không	Mã thú cưng
tenThuCung	String	Có	Tên thú cưng
trangThaiBan	int	Có	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Có	Chủ của thú cưng (nếu có)

moTa	String	Có	Mô tả chi tiết
giaHienTai	BigDecimal	Có	Giá hiện tại
giaKM	BigDecimal	Không	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Có	Thông tin chi nhánh
giong	GiongDTO	Có	Thông tin giống thú cưng
hinhAnh	List<Long>	Không	Danh sách hình ảnh của thú cưng
soLuongTon	int	Không	Số lượng tồn thú cưng

Ví dụ:

```
{
    "maThuCung": 8,
    "tenThuCung": "Tắc kè hoa",
    "trangThaiBan": 0,
    "chu": "Chưa xác định",
    "moTa": "Tắc kè hoa, d? an, kêu thu?ng xuyên",
    "giaHienTai": 200000.0000,
    "giaKM": null,
    "chiNhanh": {
        "maChiNhanh": 1,
        "tenChiNhanh": "Man Thiện"
    },
    "giong": {
        "maGiong": 13,
        "tengiong": "Tắc kè",
        "loaiThuCung": {
            "maLoaiThuCung": 4,
            "tenLoaiThuCung": "Bò sát"
        }
    },
    "hinhAnh": null,
    "soLuongTon": 5
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	ThuCungDTO	Thành công
400	"Them that bai"	Lỗi lưu thú cưng

ThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maThuCung	long	Mã thú cưng
tenThuCung	String	Tên thú cưng
trangThaiBan	int	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Chủ của thú cưng (nếu có)
moTa	String	Mô tả chi tiết
giaHienTai	BigDecimal	Giá hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh
giong	GiongDTO	Thông tin giống thú cưng
hinhAnh	List<Long>	Danh sách hình ảnh của thú cưng
soLuongTon	int	Số lượng tồn thú cưng

Ví dụ:

```
{  
    "maThuCung": 8,  
    "tenThuCung": "Tắc kè hoa",  
    "trangThaiBan": 0,  
    "chu": "Chưa xác định",  
    "moTa": "Tắc kè hoa, d? an, kêu thu?ng xuyên",  
    "giaHienTai": 200000.0000,  
    "giaKM": null,  
    "chiNhanh": {  
        "maChiNhanh": 1,  
        "tenChiNhanh": "Man Thiện"  
    },  
    "gioong": {  
        "maGiong": 13,  
        "tengiong": "Tắc kè",  
        "loaiThuCung": {  
            "maLoaiThuCung": 4,  
            "tenLoaiThuCung": "Bò sát"  
        }  
    },  
    "hinhAnh": null,  
    "soLuongTon": 5  
}
```

STT55: Sửa thú cưng

PUT: {base_url} /center/thucung

Đầu vào: @Body ThuCungDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maThuCung	long	Có	Mã thú cưng
tenThuCung	String	Có	Tên thú cưng
trangThaiBan	int	Có	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Có	Chủ của thú cưng (nếu có)

moTa	String	Có	Mô tả chi tiết
giaHienTai	BigDecimal	Có	Giá hiện tại
giaKM	BigDecimal	Không	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Có	Thông tin chi nhánh
giong	GiongDTO	Có	Thông tin giống thú cưng
hinhAnh	List<Long>	Không	Danh sách hình ảnh của thú cưng
soLuongTon	int	Không	Số lượng tồn thú cưng

Ví dụ:

```
{
    "maThuCung": 8,
    "tenThuCung": "Tắc kè hoa",
    "trangThaiBan": 0,
    "chu": "Chưa xác định",
    "moTa": "Tắc kè hoa, d? an, kêu thu?ng xuyên",
    "giaHienTai": 200000.0000,
    "giaKM": null,
    "chiNhanh": {
        "maChiNhanh": 1,
        "tenChiNhanh": "Man Thiện"
    },
    "giong": {
        "maGiong": 13,
        "tengiong": "Tắc kè",
        "loaiThuCung": {
            "maLoaiThuCung": 4,
            "tenLoaiThuCung": "Bò sát"
        }
    },
    "hinhAnh": null,
    "soLuongTon": 5
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	ThuCungDTO	Thành công
400	"Thú cưng khong ton tai"	Không tồn tại thú cưng phù hợp mã thú cưng trong hệ thống

ThuCungDTO Body:

Thuộc tính	Kiểu	Mô tả
maThuCung	long	Mã thú cưng
tenThuCung	String	Tên thú cưng
trangThaiBan	int	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Chủ của thú cưng (nếu có)
moTa	String	Mô tả chi tiết
giaHienTai	BigDecimal	Giá hiện tại
giaKM	BigDecimal	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Thông tin chi nhánh
giong	GiongDTO	Thông tin giống thú cưng
hinhAnh	List<Long>	Danh sách hình ảnh của thú cưng
soLuongTon	int	Số lượng tồn thú cưng

Ví dụ:

```
{  
    "maThuCung": 8,  
    "tenThuCung": "Tắc kè hoa",  
    "trangThaiBan": 0,  
    "chu": "Chưa xác định",  
    "moTa": "Tắc kè hoa, đ? an, kêu thu?ng xuyên",  
    "giaHienTai": 200000.0000,  
    "giaKM": null,  
    "chiNhanh": {  
        "maChiNhanh": 1,  
        "tenChiNhanh": "Man Thiện"  
    },  
    "giong": {  
        "maGiong": 13,  
        "tengiong": "Tắc kè",  
        "loaiThuCung": {  
            "maLoaiThuCung": 4,  
            "tenLoaiThuCung": "Bò sát"  
        }  
    },  
    "hinhAnh": null,  
    "soLuongTon": 5  
}
```

STT56: Xóa thú cưng

DELETE: {base_url} /center/thucung/{id}

Đầu vào: @Path long id

Ví dụ: {base_url} /center/thucung/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Xóa thành công"	Thành công
400	"Thú cưng không tồn tại"	Thú cưng không có id nằm trong hệ thống
	"Xóa thất bại"	Thú cưng vẫn còn tồn tại sau khi xóa

STT57: Cập nhật số lượng tồn thú cưng

PUT: {base_url} /center/thucung/soluongton

Đầu vào: @Body ThuCungDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maThuCung	long	Có	Mã thú cưng
tenThuCung	String	Không	Tên thú cưng
trangThaiBan	int	Không	Trạng thái bán của thú cưng: 0 – Chưa khóa 1 – Đã khóa
chu	String	Không	Chủ của thú cưng (nếu có)
moTa	String	Không	Mô tả chi tiết
giaHienTai	BigDecimal	Không	Giá hiện tại
giaKM	BigDecimal	Không	Giá khuyến mãi (nếu có)
chiNhanh	ChiNhanhDTO	Không	Thông tin chi nhánh
giong	GiongDTO	Không	Thông tin giống thú cưng
hinhAnh	List<Long>	Không	Danh sách hình ảnh của thú cưng
soLuongTon	int	Có	Số lượng tồn thú cưng

Ví dụ:

```
{
    "maThuCung": 8,
    "tenThuCung": null,
    "trangThaiBan": null,
    "chu": null,
    "moTa": null,
    "giaHienTai": null,
```

```

    "giaKM": null,
    "chiNhanh": null,
    "giuong": null,
    "hinhAnh": null,
    "soLuongTon": 5
}

```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	"Cập nhật thành công"	Thành công
400	"Cập nhật thất bại"	Lưu số lượng tồn ở thú cưng thất bại

4. Order Service

a. Bảng API

STT	Path	Method	Yêu cầu đầu vào	Mô tả
1	order/don-dat	Get	Không	Lấy danh sách đơn đặt
2	order/don-dat	Post	DonDatDTO	Tạo đơn đặt hàng
3	order/dat-hang	Get	Không	Lấy danh sách hóa đơn
4	order/dat-hang/{id}	Get	long	Lấy hóa đơn chi tiết
5	order/dat-hang/sp	Post	List<DonDatSanPhamRequest>	Thêm chi tiết đơn đặt hàng về sản phẩm
6	order/dat-hang/tc	Post	List<DonDatThuCongRequest>	Thêm chi tiết đơn đặt hàng về thú cưng
7	order/dat-hang/hoa-don	Post	HoaDonRequest	Thêm hóa đơn
8	order/dat-hang/thanhtien/{id}	Get	long	Tính thành tiền đơn đặt hàng

b. Chi tiết

STT1: Lấy danh sách đơn đặt hàng

GET: {base_url} /order/don-dat

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân
200	List<DonDatDTO>	Thành công

DonDatDTO Body:

Thuộc tính	Kiểu	Mô tả
soDonDat	long	Mã số đơn đặt hàng
ngayLap	Date	Ngày lập đơn đặt hàng
diaChi	String	Địa chỉ giao hàng
soDienThoai	String	Số điện thoại liên lạc
maChiNhanh	Integer	Mã chi nhánh mua hàng
maKhachHang	String	Mã khách mua hàng (Tên đăng nhập)
trangThai	Boolean	Trạng thái thanh toán: 1: Đã thanh toán và có hóa đơn 2: Chưa thanh toán và không có hóa đơn

Ví dụ:

```
[
  {
    "soDonDat": 1,
    "ngayLap": "2024-06-03T17:00:00.000+00:00",
    "diaChi": "23/4 Trần Thị Hoa",
    "soDienThoai": "09172133223",
    "maChiNhanh": 1,
    "maKhachhang": "lamvu2010",
    "trangThai": true
  },
  {
    "soDonDat": 38,
    "ngayLap": "2024-06-07T06:51:55.853+00:00",
    "diaChi": "28 Đại Nam",
    "soDienThoai": "09012390213",
    "maChiNhanh": 2,
    "maKhachhang": "lamvu2010",
    "trangThai": true
  }
]
```

STT2: Tao đơn đặt hàng

POST: {base_url} /order/don-dat

Đầu vào: @Body DonDatDTO

Thuộc tính	Kiểu	Bắt buộc	Mô tả
soDonDat	long	Không	Mã số đơn đặt hàng
ngayLap	Date	Không	Ngày lập đơn đặt hàng
diaChi	String	Có	Địa chỉ giao hàng
soDienThoai	String	Có	Số điện thoại liên lạc
maChiNhanh	Integer	Có	Mã chi nhánh mua hàng
maKhachHang	String	Có	Mã khách mua hàng (Tên đăng nhập)
trangThai	Boolean	Không	Trạng thái thanh toán: 1: Đã thanh toán và có hóa đơn 2: Chưa thanh toán và không có hóa đơn

Ví dụ:

```
{
    "soDonDat": null,
    "ngayLap": null,
    "diaChi": "128 Ngõ Quyền P2, Q5, TPHCM",
    "soDienThoai": "0395387645",
    "maChiNhanh": 1,
    "maKhachhang": "lamvu2010",
    "trangThai": null
}
```

Đầu ra:

Code	Kết quả	Nguyên nhân
200	DonDatDTO	Thành công
400	"Thêm thất bại"	Lưu đơn đặt thất bại

DonDatDTO Body:

Thuộc tính	Kiểu	Mô tả
soDonDat	long	Mã số đơn đặt hàng
ngayLap	Date	Ngày lập đơn đặt hàng
diaChi	String	Địa chỉ giao hàng
soDienThoai	String	Số điện thoại liên lạc
maChiNhanh	Integer	Mã chi nhánh mua hàng
maKhachHang	String	Mã khách mua hàng (Tên đăng nhập)
trangThai	Boolean	Trạng thái thanh toán: 1: Đã thanh toán và có hóa đơn 2: Chưa thanh toán và không có hóa đơn

Ví dụ:

```
{  
    "soDonDat": 100,  
    "ngayLap": "2024-06-03T17:00:00.000+00:00",  
    "diaChi": "128 Ngô Quyền P2, Q5, TPHCM",  
    "soDienThoai": "0395387645",  
    "maChiNhanh": 1,  
    "maKhachhang": "lamvu2010",  
    "trangThai": false  
}
```

STT3: Lấy danh sách hóa đơn

GET: {base_url} /order/dat-hang

Đầu vào: Không

Đầu ra:

Code	Kết quả	Nguyên nhân

200	List<HoaDonDTO>	Thành công
-----	-----------------	------------

HoaDonDTO Body:

Thuộc tính	Kiểu	Mô tả
donDat	DonDatDTO	Đơn đặt liên kết với hóa đơn
maHoaDon	long	Mã số hóa đơn
ngayLap	Timestamp	Ngày giờ lập hóa đơn
tongHoaDon	BigDecimal	Tổng số tiền tính trên hóa đơn
maNhanVien	String	Mã nhân viên lập hóa đơn (nếu thanh toán bằng tiền mặt)

Ví dụ:

```
[
  {
    "donDat": {
      "soDonDat": 1,
      "ngayLap": "2024-06-03T17:00:00.000+00:00",
      "diaChi": "23/4 Trần Thị Hoa",
      "soDienThoai": "0917",
      "maChiNhanh": 1,
      "maKhachhang": "lamvu2010",
      "trangThai": true
    },
    "maHoaDon": 1,
    "ngayLap": "2024-06-06T03:02:50.080+00:00",
    "tongHoaDon": 18250000.000,
    "maNhanVien": "NV3"
  },
  {
    "donDat": {
      "soDonDat": 38,
      "ngayLap": "2024-06-07T06:51:55.853+00:00",
      "diaChi": "kfjsk",
      "soDienThoai": "sfvsf",
      "maChiNhanh": 1,
      "maKhachhang": "lamvu2010",
      "trangThai": true
    },
  }
]
```

```

    "maHoaDon": 38,
    "ngayLap": "2024-06-07T07:46:51.163+00:00",
    "tongHoaDon": 232.0000,
    "maNhanVien": "NV3"
}
]

```

STT4: Lấy hóa đơn cụ thể

GET: {base_url} /order/dat-hang/{id}

Đầu vào: @Path long

Ví dụ: {base_url} /order/dat-hang/1

Đầu ra:

Code	Kết quả	Nguyên nhân
200	HoaDonDTO	Thành công
400	"Không tìm thấy hóa đơn, đơn đặt"	Nếu không tìm thấy đơn đặt hoặc hóa đơn trong cơ sở dữ liệu

HoaDonDTO Body:

Thuộc tính	Kiểu	Mô tả
donDat	DonDatDTO	Đơn đặt liên kết với hóa đơn
maHoaDon	long	Mã số hóa đơn
ngayLap	Timestamp	Ngày giờ lập hóa đơn
tongHoaDon	BigDecimal	Tổng số tiền tính trên hóa đơn
maNhanVien	String	Mã nhân viên lập hóa đơn (nếu thanh toán bằng tiền mặt)

Ví dụ:

```

{
    "donDat": {
        "soDonDat": 1,
        "ngayLap": "2024-06-03T17:00:00.000+00:00",
        ...
    }
}

```

```

        "diaChi": "23/4 Trần Thị Hoa",
        "soDienThoai": "0917",
        "maChiNhanh": 1,
        "maKhachhang": "lamvu2010",
        "trangThai": true
    },
    "maHoaDon": 1,
    "ngayLap": "2024-06-06T03:02:50.080+00:00",
    "tongHoaDon": 18250000.0000,
    "maNhanVien": "NV3"
}

```

STT5: Thêm đơn đặt chi tiết sản phẩm

POST: {base_url} /order/dat-hang/sp

Đầu vào: @Body List<DonDatSanPhamRequest>

DonDatSanPhamRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maDonDat	long	Có	Mã số đơn đặt hàng
maSanPham	long	Có	Mã sản phẩm đặt
donGia	BigDecimal	Có	Đơn giá sản phẩm đặt
soLuong	int	Có	Số lượng đặt
maChiNhanh	int	Có	Mã chi nhánh đặt

Ví dụ:

```
[
{
    "maDonDat":90,
    "maSanPham":11,
    "donGia":50000.0000,
    "soLuong":2,
    "maChiNhanh":2
},
{
    "maDonDat":90,
    "maSanPham":12,
```

```

        "donGia":50000.0000,
        "soLuong":2,
        "maChiNhanh":2
    }
]

```

Đầu ra:

Code	Kết quả	Nguyên nhân
500	"Thêm thất bại: " + e.getMessage()	Lưu chi tiết thất bại kèm theo chi tiết lỗi dòng lệnh
400	"Thêm thất bại: DonDat không tìm thấy"	Không tìm thấy đơn đặt
	"Thêm thất bại"	Lỗi try catch toàn hàm
200	"Thêm thành công"	Thành công

STT6: Thêm đơn đặt chi tiết thú cưng

POST: {base_url} /order/dat-hang/tc

Đầu vào: @Body List<DonDatThuCungRequest>

DonDatThuCungRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
maDonDat	long	Có	Mã số đơn đặt hàng
maThuCung	long	Có	Mã thú cưng đặt
donGia	BigDecimal	Có	Đơn giá sản phẩm đặt
soLuong	int	Có	Số lượng đặt

Ví dụ:

```

[
{
    "maDonDat":90,
    "maThuCung":11,
    "donGia":50000.0000,

```

```

        "soLuong":1
    },
{
    "maDonDat":90,
    "maThuCung":12,
    "donGia":50000.0000,
    "soLuong":1
}
]

```

Đầu ra:

Code	Kết quả	Nguyên nhân
400	"Thêm thất bại"	Lỗi try catch toàn hàm
200	"Thêm thành công"	Thành công

STT7: Lập hóa đơn

POST: {base_url} /order/dat-hang/hoa-don

Đầu vào: @Body HoaDonRequest

HoaDonRequest Body:

Thuộc tính	Kiểu	Bắt buộc	Mô tả
soHoaDon	long	Có	Mã số hóa đơn (Mã số đơn đặt hàng)
maNhanVien	String	Có	Mã nhân viên lập

Ví dụ:

```

{
    "maDonDat":90,
    "maNhanVien":"NV1"
}

```

Đầu ra:

Code	Kết quả	Nguyên nhân
404	"Đơn đặt không tồn tại"	Đơn đặt không tồn tại trong hệ thống

400	"Thêm thất bại"	Lỗi update đơn đặt và tạo hóa đơn
200	"Thêm thành công"	Thành công

STT8: Tính thành tiền

GET: {base_url} /order/dat-hang/thanhtien/{id}

Đầu vào: @Path long

Ví dụ: {base_url} /order/dat-hang/thanhtien/90

Đầu ra:

Code	Kết quả	Nguyên nhân
200	String thanhTien (Thành tiền của đơn đặt hàng)	Thành công

Chương IV: Triển khai webservice trên docker

Docker build flow:



Jib build flow:



Mô hình triển khai docker với Spring Boot

- Mô hình sử dụng thư viện Jib Java để thực hiện việc đóng gói project và gửi lên Docker Hub. Thay vì phải làm các bước tuân tự việc tạo Docker file, đóng gói file jar, build thành Docker Image sau đó đẩy lên DockerHub thì Jib hỗ trợ cho chúng ta toàn bộ công đoạn phức tạp đó bằng các dependencies và plugins cấu hình sẵn trong pom.xml.

- Cấu hình pom.xml:

```
<plugin>
    <groupId>com.google.cloud.tools</groupId>
    <artifactId>jib-maven-plugin</artifactId>
    <version>3.4.3</version>
    <configuration>
        <from>
            <image>eclipse-temurin:17.0.4.1_1-jre</image>
        </from>
        <to>
            <image>registry.hub.docker.com/lamvu2010/${project.artifactId}</image>
            <auth>
                <username>${env.DOCKER_USERNAME}</username>
                <password>${env.DOCKER_PASSWORD}</password>
            </auth>
        </to>
    </configuration>
</plugin>
```

- Sau đó chúng ta kéo Images và Container về để chạy. Sử dụng Docker Desktop để quản lý container trực quan hơn.

Containers [Give feedback](#)

Container CPU usage [i](#)

No containers are running.

Container memory usage [i](#)

No containers are running.

[Show charts](#)

<input type="checkbox"/>	Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	project		Exited		N/A	23 minutes ago	▶ ⋮ ✖
<input type="checkbox"/>	identity-se f0cf3496084	lamvu2010/identity-se	Exited (143)		N/A	23 minutes ago	▶ ⋮ ✖
<input type="checkbox"/>	center-ser 5b8db85c05	lamvu2010/center-ser	Exited (143)		N/A	23 minutes ago	▶ ⋮ ✖
<input type="checkbox"/>	order-serv c18bb565b3	lamvu2010/order-serv	Exited (143)		N/A	23 minutes ago	▶ ⋮ ✖
<input type="checkbox"/>	api-gatew 5d2a3338c7	lamvu2010/api-gatew	Exited (143)	8989:8989	N/A	23 minutes ago	▶ ⋮ ✖
<input type="checkbox"/>	service-re 8888:8888	lamvu2010/service-re	Exited (143)	8761:8761	N/A	23 minutes ago	▶ ⋮ ✖

Showing 7 items

- Cuối cùng dùng Docker Desktop để chạy container.

Chương V: Triển khai trên Website React Js

Cấu hình AxiosInstance để lấy dữ liệu thông qua API, sử dụng thư viện axios:

```
const baseUrl = 'http://localhost:8989/'
const AxiosInstance = axios.create({
    baseURL: baseUrl,
    timeout: 5000,
    headers:{
        "Content-type": "application/json",
        accept: "application/json",
    }
})

AxiosInstance.interceptors.request.use(
    (config) => {
        const token = localStorage.getItem('Token')
        if(token){
            config.headers.Authorization = `Token ${token}`
        }
        else{
            config.headers.Authorization = ``
        }
        return config;
    }
)

AxiosInstance.interceptors.response.use(
    (response) => {
        return response
    },
    (error) => {
        if(error.response && error.response.status === 401){
            localStorage.removeItem('Token')
        }
    }
)
```

1. Quản lý

Quản lý sản phẩm và thú cưng, loại sản phẩm, loại thú cưng, giống, bảng giá, chi nhánh và nhân viên, các chức năng có thao tác quản lý tương đương nhau, bao gồm: Danh sách, Thêm, Xóa, Sửa.

a. Load dữ liệu

API lấy dữ liệu trong Spring Boot:

```
@GetMapping  
public ResponseEntity<List<SanPhamDTO>> getAll() {  
    List<Ctsanpham> list = chiTietSanPhamService.findAll();  
    List<SanPhamDTO> dtoList = new ArrayList<>();  
    for (Ctsanpham item : list) {  
        SanPhamDTO sanPhamDTO = convertToDTO(item);  
        dtoList.add(sanPhamDTO);  
    }  
    return new ResponseEntity<>(dtoList, HttpStatus.OK);  
}
```

Convert dữ liệu theo chuẩn giao tiếp DTO:

```
public SanPhamDTO convertToDTO(Ctsanpham ctsanpham) {  
    SanPhamDTO sanPhamDTO = new SanPhamDTO();  
    if (ctsanpham == null) {  
        return sanPhamDTO;  
    }  
    sanPhamDTO.setMaSanPham(ctsanpham.getSanpham().getMasanpham());  
    sanPhamDTO.setTenSanPham(ctsanpham.getSanpham().getTensanpham());  
    sanPhamDTO.setGiaHienTai(ctsanpham.getSanpham().getGiahientai());  
    sanPhamDTO.setLoaiSanPham(new LoaiSanPhamDTO());  
    if (ctsanpham.getSanpham().getLoaisanpham() != null) {  
  
        sanPhamDTO.getLoaiSanPham().setMaLoaiSanPham(ctsanpham.getSanpham().getLoaisanpham().getMaloaisanpham());  
  
        sanPhamDTO.getLoaiSanPham().setTenLoaiSanPham(ctsanpham.getSanpham().getLoaisanpham().getTenloaisanpham());  
    }  
    if (ctsanpham.getSanpham().getHinhanh() != null && !ctsanpham.getSanpham().getHinhanh().isEmpty()) {  
        sanPhamDTO.setHinhAnh(new ArrayList<>());  
        for (Hinhanh item : ctsanpham.getSanpham().getHinhanh()) {  
            sanPhamDTO.getHinhAnh().add(item.getMahinhanh());  
        }  
    }  
    if (ctsanpham.getChinhanh() != null) {  
        sanPhamDTO.setMaChiNhanh(ctsanpham.getChinhanh().getMachinhanh());  
    }  
    sanPhamDTO.setSoLuongTon(ctsanpham.getSoluongton());  
    return sanPhamDTO;  
}
```

Khi đó, ở Webservice sẽ gọi đến API để lấy danh sách:

```
const getProduct = async ()=>{
  try{
    const res = await AxiosInstance.get("/center/sanpham")
    setProducts(res.data)
  }
  catch(err){
    console.error(err);
  }
}
```

Sau đó sẽ đổ dữ liệu của danh sách vào bảng:

```
<TableBody>
  {products.map((product, index) => (
    <TableRow key={index}
      onClick={() => handleRowClick(index)}
      sx={{ backgroundColor: selectedBreedIndex === index ?
"#f0f0f0" : "inherit" }}
    >
      <TableCell>{product.maSanPham}</TableCell>
      <TableCell>{product.hinhAnh && product.hinhAnh[0] &&
imageUrls[product.hinhAnh[0]] ? (
        <img src={imageUrls[product.hinhAnh[0]]}
          alt="Hình ảnh" />
        ) : ''}</TableCell>
      <TableCell>{product.tenSanPham}</TableCell>
      <TableCell>{product.giaHienTai}</TableCell>
      <TableCell>{product.giaKM}</TableCell>
      <TableCell>{product.maChiNhanh}</TableCell>
      <TableCell>{product.loaiSanPham.tenLoaiSanPham}</Tabl
eCell>
      <TableCell>{product.soLuongTon}</TableCell>
    </TableRow>
  ))}
</TableBody>
```

Giao diện:

The screenshot shows a left sidebar with various management options: Quản lý nhân viên, Quản lý chi nhánh, Quản lý thú cưng, Quản lý sản phẩm (highlighted), Quản lý giá, Quản lý loại, Quản lý loài, Quản lý giống, Xem danh thu, and Đăng xuất. To the right is a main content area with a search bar and buttons for THÊM, SỬA, and XÓA. Below is a table listing products with columns: Mã sản phẩm, Hình ảnh, Tên sản phẩm, Giá, Giá khuyến mãi, Mã chi nhánh, loại sản phẩm, and Số lượng tồn.

Mã sản phẩm	Hình ảnh	Tên sản phẩm	Giá	Giá khuyến mãi	Mã chi nhánh	loại sản phẩm	Số lượng tồn
3		Thức ăn hạt mềm zenith	100000		1	Đồ ăn	28
4		Thức ăn hạt ANF	100000		1	Đồ ăn	65
5		Thức ăn hạt Natural Core	400000		1	Đồ ăn	100
6		Thức ăn cho chó Smartheart	300000		1	Đồ ăn	100
7		Thức ăn hạt Ganador	200000		1	Đồ ăn	150
8		Sữa Royal Canin	600000		1	Thức uống	150
9		Sữa Úc pet own	1000000		1	Thức uống	170

b. Thêm dữ liệu.

Tạo API thêm dữ liệu trong Spring Boot, cụ thể là thêm sản phẩm:

```
// Them san pham
@PostMapping
public ResponseEntity<?> insert(@RequestBody SanPhamDTO sanPhamDTO) {
    try {
        Sanpham sp = new Sanpham();
        Ctsanpham ctsanpham = new Ctsanpham();
        CtsanphamPK ctsanphamPK = new CtsanphamPK();

        sp.setTensanpham(sanPhamDTO.getTenSanPham());
        sp.setGiahientai(sanPhamDTO.getGiaHienTai());

        sp.setLoaisanpham(loaiSanPhamService.findById(sanPhamDTO.getLoaiSanPham().getMaLoaiSanPham()).orElse(null));
        sp = sanPhamService.save(sp);
        ctsanphamPK.setMasanpham(sp.getMasanpham());
        ctsanphamPK.setMachinhanh(sanPhamDTO.getMaChiNhanh());

        if (!chinhNhanhService.existById(sanPhamDTO.getMaChiNhanh())) {
            return new ResponseEntity<>("Chi nhánh không tồn tại",
HttpStatus.BAD_REQUEST);
        }
        Chinhanh chinhanh =
        chinhNhanhService.findById(sanPhamDTO.getMaChiNhanh()).get();

        ctsanpham.setId(ctsanphamPK);
        ctsanpham.setChinhanh(chinhanh);
        ctsanpham.setSanpham(sp);
        ctsanpham.setSoluongton(sanPhamDTO.getSoLuongTon());
        chiTietSanPhamService.save(ctsanpham);
        SanPhamDTO sanPhamDTO1 = convertToDTO(ctsanpham);
        return new ResponseEntity<>(sanPhamDTO1, HttpStatus.OK);
    } catch (Exception e) {
```

```

        System.out.println(e.getMessage());
        return new ResponseEntity<>("Thêm thất bại", HttpStatus.BAD_REQUEST);
    }
}

```

- Sau đó, Webservice sẽ lấy dữ liệu thông qua API:

```

const handleFormSubmit = async(data) => {

    const { hinhAnh, ...rest } = data;
    try{
        const res = await AxiosInstance.post("/center/sanpham", rest)
        if(res.status === 200){
            try{
                console.log(res.data.maSanPham)
                const pushimage = await AxiosInstance.post("/center/image", {
                    "image":hinhAnh.get('image'),
                    "maSanPham": res.data.maSanPham
                },{
                    headers: {
                        'Content-Type': 'multipart/form-data',
                    },
                })
                if(pushimage.status === 200){
                    enqueueSnackbar('Thêm sản phẩm thành công', {variant : 'success',
autoHideDuration: 3000} )
                    getProduct()
                }
            }
            catch(e){
                enqueueSnackbar('Lỗi khi đẩy ảnh lên', {variant : 'error',
autoHideDuration: 3000} )
                console.log(e)
            }
        }
    }
    catch(err){
        enqueueSnackbar('Lỗi thêm sản phẩm', {variant : 'error', autoHideDuration:
3000} )
        console.log(err)
    }
    handleDialogClose();
};

```

- Giao diện khi thêm dữ liệu, cụ thể là sản phẩm:

Lý Nhân Viên

Thêm Sản Phẩm

Tên sản phẩm: Nhà Cho Chó

Chi Nhánh: Man Thiện

Loại sản phẩm: Nhà ở

Giá Hiện Tại: 150000

Số lượng tồn: 100

CHỌN HÌNH ẢNH

HỦY THÊM

Tên sản phẩm	Giá	Giá khuyến mãi	Mã chi nhánh	loại sản phẩm	Số lượng tồn
--------------	-----	----------------	--------------	---------------	--------------

Nhà Cho Chó	150000	1	Nhà ở	100
-------------	--------	---	-------	-----

Đã có dữ liệu khi thêm vào.

c. Sửa thông tin

Tạo API thêm dữ liệu trong Spring Boot, cụ thể là sửa sản phẩm:

```
//Sua san pham
@PutMapping
public ResponseEntity<?> update(@RequestBody SanPhamDTO sanPhamDTO) {
    Sanpham sp = new Sanpham();
    if (!sanPhamService.existsById(sanPhamDTO.getMaSanPham())) {
        return new ResponseEntity<>(sp, HttpStatus.BAD_REQUEST);
    }
    sp.setMasanpham(sanPhamDTO.getMaSanPham());
    sp.setGiahientai(sanPhamDTO.getGiaHienTai());
    sp.setTensanpham(sanPhamDTO.getTenSanPham());

    sp.setLoaisanpham(loaiSanPhamService.findById(sanPhamDTO.getLoaiSanPham().get
    MaLoaiSanPham()).orElse(null));
    sp = sanPhamService.save(sp);
    Ctsanpham ctsanpham = chiTietSanPhamService.findById(new
    CtsanphamPK(sanPhamDTO.getMaChiNhanh(), sanPhamDTO.getMaSanPham())).get();
    SanPhamDTO sanPhamDTO1 = convertToDTO(ctsanpham);
    return new ResponseEntity<>(sanPhamDTO1, HttpStatus.OK);
}
```

Webservice gọi đến API để thao tác sửa dữ liệu:

```
const handleEditFormSubmit = async(data) => {
    console.log("Form Data:", data);
    const { hinhAnh, ...rest } = data;
    console.log(rest)
    if(hinhAnh.get('image')){
        try{
            const res = await AxiosInstance.put("/center/sanpham", rest)
            if(res.status === 200){
                try{
                    const pushimage = await AxiosInstance.post("/center/image", {
                        "image":hinhAnh.get('image'),
                        "maSanPham": res.data.maThuCung
                    },{
                        headers: {
                            'Content-Type': 'multipart/form-data',
                        },
                    })
                    if(pushimage.status === 200){
                        enqueueSnackbar('Sửa sản phẩm thành công', {variant : 'success',
                        autoHideDuration: 3000} )
                        getProduct()
                    }
                }
            }
        }
    }
}
```

```

        }
        catch(e){
            console.log(e)
            enqueueSnackbar('Lỗi đẩy ảnh', {variant : 'error', autoHideDuration: 3000} )
        }
    }
}
catch(err){
    console.log(err)
    enqueueSnackbar('Lỗi cập nhập sản phẩm', {variant : 'error', autoHideDuration: 3000} )
}
else{
    try{
        const res = await AxiosInstance.put("/center/sanpham", rest)
        if(res.status === 200){
            enqueueSnackbar('sửa sản phẩm thành công', {variant : 'success', autoHideDuration: 3000} )
            getProduct()
        }
    }
    catch(err){
        console.log(err)
        enqueueSnackbar('Lỗi cập nhập sản phẩm', {variant : 'error', autoHideDuration: 3000} )
    }
}
handleEditDialogClose();
};

```

Giao diện khi sửa:

The screenshot shows a form for editing a product. The fields are as follows:

- Mã sản phẩm: 3
- Tên sản phẩm: Thức ăn hạt mềm zenith
- Chi Nhánh: Man Thiện
- Loại sản phẩm: Đồ ăn
- Giá Hiện Tại: 100000

Below the form is a blue button labeled "CHỌN HÌNH ẢNH". At the bottom right are two buttons: "HỦY" and "SỬA".

Sau khi sửa: Sản phẩm có id là 3 đã được đổi tên là MELANA.

3	Thức ăn hạt mềm MELANA	100000	1	Đồ ăn	28
---	------------------------	--------	---	-------	----

d. Xóa thông tin

Tập API xóa sản phẩm trong Spring Boot:

```
// Xoa san pham
@DeleteMapping("/{sanpham}/{chinhanh}")
public ResponseEntity<?> delete(@PathVariable Long sanpham, @PathVariable int chinhanh) {
    CtsanphamPK ctsanphamPK = new CtsanphamPK(chinhanh, sanpham);
    boolean spTonTai = chiTietSanPhamService.existsById(ctsanphamPK);
    if (spTonTai == false) {
        return new ResponseEntity<>("Id khong ton tai",
HttpStatus.NOT_FOUND);
    }
    Optional<Ctsanpham> ctsanpham =
chiTietSanPhamService.findById(ctsanphamPK);
    chiTietSanPhamService.delete(ctsanpham.orElse(null));
    spTonTai = chiTietSanPhamService.existsById(ctsanphamPK);
    if (spTonTai == true) {
        return new ResponseEntity<>("Xoa that bai", HttpStatus.BAD_REQUEST);
    } else {
        return new ResponseEntity<>("Xoa thanh cong", HttpStatus.OK);
    }
}
```

Webservice sẽ gọi đến API để thao tác xóa:

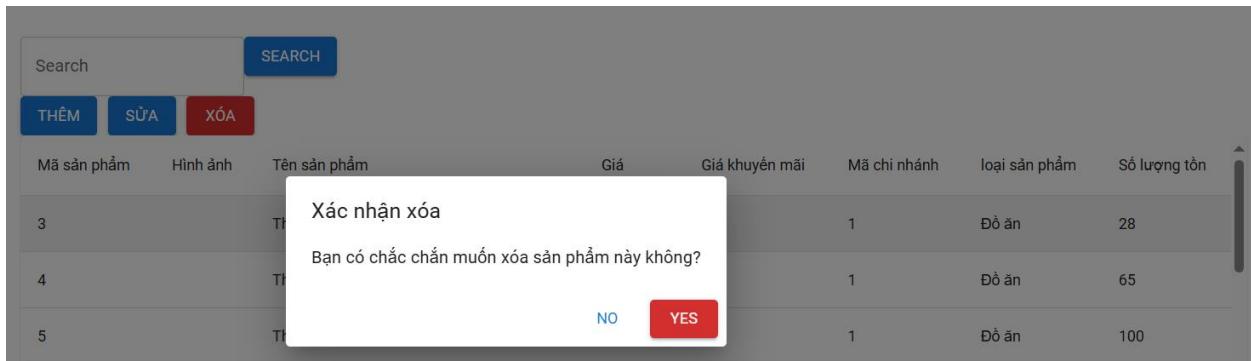
```
const handleConfirmDelete = async (product) => {
    try {
        const res = await
AxiosInstance.delete(`/center/sanpham/${product.maSanPham}/${product.maChiNhanh}`);
    };
}
```

```

if (res.status === 200) {
    enqueueSnackbar('Xóa sản phẩm thành công', { variant: 'success',
autoHideDuration: 3000 });
    getProduct()
}
} catch (err) {
    enqueueSnackbar('Lỗi khi xóa sản phẩm', { variant: 'error',
autoHideDuration: 3000 });
    console.error(err);
}
setIsConfirmOpen(false);
setSelectedBreedIndex(null);
};

```

Giao diện khi xóa: Chọn sản phẩm có id số 3 để xóa:



Sản phẩm có id là 3 đã bị xóa:

Mã sản phẩm	Hình ảnh	Tên sản phẩm	Giá	Giá khuyến mãi	Mã chi nhánh	loại sản phẩm	Số lượng tồn
4		Thức ăn hạt ANF	100000		1	Đồ ăn	65
5		Thức ăn hạt Natural Core	400000		1	Đồ ăn	100
6		Thức ăn cho chó Smartheart	300000		1	Đồ ăn	100
7		Thức ăn hạt Ganador	200000		1	Đồ ăn	150
8		Sữa Royal Canin	600000		1	Thức uống	150
9		Sữa Úc pet own	1000000		1	Thức uống	170
10		Sữa Petlac thú cưng	2000000		1	Thức uống	80

e. Hình ảnh.

- Hình ảnh trong Spring Boot cho phép nhận về nhiều file ảnh khác nhau và lưu vào thư mục cụ bộ bên trong project, trên database chỉ cần lưu trữ đường dẫn và tên hình ảnh. Hình ảnh được liên kết và sử dụng trong 2 service center và identity (sản phẩm, thú cưng và nhân viên, khách hàng).

Hàm lưu hình ảnh trong Spring Boot:

```
@PostMapping("saveImage")
public ResponseEntity<?> saveImage(@RequestParam("image") MultipartFile[] file,
                                         @RequestParam(value = "maNhanVien", required = false) String maNhanVien,
                                         @RequestParam(value = "maKhachHang", required = false) String maKhachHang,
                                         @RequestParam(value = "maThuCung", required = false) Long maThuCung,
                                         @RequestParam(value = "maSanPham", required = false) Long maSanPham)
    throws IOException {
    try {
        for (MultipartFile item : file) {
            storageService.uploadImageToFileSystem(item, maNhanVien, maKhachHang, maThuCung, maSanPham);
        }
        return new ResponseEntity<?>(body: "Lưu thành công", HttpStatus.OK);
    } catch (IOException e) {
        return new ResponseEntity<?>(body: "Lưu thất bại", HttpStatus.BAD_REQUEST);
    }
}
```

Webservice gọi đến API lưu hình ảnh:

```
try{
    const pushimage = await AxiosInstance.post("/center/image", {
        "image":hinhAnh.get('image'),
        "maSanPham": res.data.maThuCung
    },{
        headers: {
            'Content-Type': 'multipart/form-data',
        },
    })
    if(pushimage.status === 200){
        enqueueSnackbar('Sửa sản phẩm thành công', {variant : 'success', autoHideDuration: 3000} )
        getProduct()
    }
}
catch(e){
    console.log(e)
    enqueueSnackbar('Lỗi đẩy ảnh', {variant : 'error', autoHideDuration: 3000} )
}
```

Tạo API để lấy hình ảnh:

```
@PostMapping("/get")
public ResponseEntity<?> sendImage(@RequestBody long[] id) throws IOException {
    MultiValueMap<String, Object> body = new LinkedMultiValueMap<>();
    HttpHeaders headers = new HttpHeaders();
    headers.setContentType(MediaType.MULTIPART_FORM_DATA);

    for (long itemId : id) {
        byte[] image = storageService.downloadImageFromFileSystem(itemId);
        ByteArrayResource imageResource = new ByteArrayResource(image) {
            @Override
            public String getFilename() {
                return "image-" + itemId; // Cung cấp một tên duy nhất cho
                                           // mỗi file
            }
        }
        body.add("image", imageResource);
    }
    return ResponseEntity.ok(body);
}
```

```

        };
        body.add("images", imageResource);
    }
    return new ResponseEntity<>(body, headers, HttpStatus.OK);
}

```

Webservice gọi đến API để lấy hình ảnh:

```

const getHinhAnh = async (id) => {
    try {
        const res = await AxiosInstance.post("/center/hinhanh/get", [id]);
        if (res.status === 200) {
            const base64Image = res.data[0].source;
            const blob = base64ToBlob(base64Image, 'image/jpeg');
            const imageUrl = URL.createObjectURL(blob);
            return imageUrl;
        }
    } catch (e) {
        // console.log(e);
    }
    return null;
}

```

2. Bán hàng

a. Trang chủ.

- Hệ thống lấy danh sách các thú cưng và sản phẩm hiển thị lên trang mua hàng của người dùng.
- Thú cưng và sản phẩm sẽ được hiển thị theo Chi Nhánh. Khách hàng có thể chọn Chi Nhánh để xem các sản phẩm và thú cưng có ở Chi Nhánh đó.

Chọn Chi Nhánh

Trần Thị Hoa

Man Thiện

Trần Thị Hoa

Nhỏ gáy Revolution nội ngoại ký sinh Thuốc 40000 VND 50000 VND Số lượng tồn: 90	Nhỏ gáy IMIDA Plus ngừa ve rận Thuốc 50000 VND Số lượng tồn: 100	Nhỏ gáy trị ve rận, bọ chét Frontline Thuốc 70000 VND Số lượng tồn: 60	Nhỏ gáy ngừa ve rận, giun, ghẻ Boardline Thuốc 100000 VND Số lượng tồn: 150
THÊM VÀO GIỎ HÀNG	THÊM VÀO GIỎ HÀNG	THÊM VÀO GIỎ HÀNG	THÊM VÀO GIỎ HÀNG

- Xử lý thao tác lấy danh sách thú cưng và sản phẩm ở Spring Boot:

- Lấy sản phẩm:

```
// Lay danh sach san pham ban
@GetMapping
public ResponseEntity<?> getDanhSachSanPhamBan() {
    List<Map<?, ?>> list = bangGiaSanPhamService.danhSachSanPhamBan();
    List<BangGiaSanPhamDTO> dtoList = new ArrayList<>();
    if (list.isEmpty()) {
        return new ResponseEntity<>("Không có dữ liệu",
        HttpStatus.BAD_REQUEST);
    }
    for (Map<?, ?> item : list) {
        BangGiaSanPhamDTO bangGiaSanPhamDTO = new BangGiaSanPhamDTO();
        if (item.get("MASANPHAM") != null) {
            bangGiaSanPhamDTO.setMaSanPham((long) item.get("MASANPHAM"));
            bangGiaSanPhamDTO.setTenSanPham((String) item.get("TENSANPHAM"));
            bangGiaSanPhamDTO.setGiaHienTai((BigDecimal)
item.get("GIAHIENTAI"));
            bangGiaSanPhamDTO.setSoLuongTon((long) item.get("SOLUONGTON"));
            Sanpham sanpham =
sanPhamService.findById(bangGiaSanPhamDTO.getMaSanPham()).get();
            List<Hinhanh> hinhanhList = sanpham.getHinhanh();
            if(hinhanhList!=null&&hinhanhList.size()!=0){
                long idHinhAnh = hinhanhList.get(0).getMahinhanh();
                try{
                    byte[] image =
storageService.downloadImageFromFileSystem(idHinhAnh);
                    bangGiaSanPhamDTO.setHinhAnh(image);
                }
                catch(Exception e){
                    System.out.println(e.getMessage());
                    bangGiaSanPhamDTO.setHinhAnh(null);
                }
            }
        }
    }
}
```

```

        }
    else{
        bangGiaSanPhamDTO.setHinhAnh(null);
    }

}

if (item.get("MALOAI SAN PHAM") != null) {
    bangGiaSanPhamDTO.setMaLoaiSanPham((int)
item.get("MALOAI SAN PHAM"));
    bangGiaSanPhamDTO.setTenLoaiSanPham((String)
item.get("TENLOAI SAN PHAM"));
}
if (item.get("MABANGGIA") != null) {
    bangGiaSanPhamDTO.setMaBangGia((long) item.get("MABANGGIA"));
    bangGiaSanPhamDTO.setThoiGianBatDau((Timestamp)
item.get("THOIGIANBATDAU"));
    bangGiaSanPhamDTO.setThoiGianKetThuc((Timestamp)
item.get("THOIGIANKETTHUC"));
    bangGiaSanPhamDTO.setGiaKhuyenMai((BigDecimal)
item.get("GIAKM"));
}
if (item.get("MACHINHANH") != null) {
    bangGiaSanPhamDTO.setMaChiNhanh((int) item.get("MACHINHANH"));
    bangGiaSanPhamDTO.setTenChiNhanh((String)
item.get("TENCHINHANH"));
}
dtoList.add(bangGiaSanPhamDTO);
}
return new ResponseEntity<>(dtoList, HttpStatus.OK);
}

```

- Lấy thú cưng:

```

@GetMapping
public ResponseEntity<?> getChiTietBangGiaThuCung() {
    List<Map<?, ?>> result = bangGiaThuCungService.danhSachThuCungBan();
    if (result.isEmpty()) {
        return new ResponseEntity<>("Không có dữ liệu",
HttpStatus.BAD_REQUEST);
    }
    List<BangGiaThuCungDTO> dtoList = new ArrayList<>();
    for (Map<?, ?> item : result) {
        BangGiaThuCungDTO bangGiaThuCungDTO = new BangGiaThuCungDTO();
        if (item.get("MABANGGIA") != null) {
            bangGiaThuCungDTO.setMaBangGia((long) item.get("MABANGGIA"));
            bangGiaThuCungDTO.setThoiGianBatDau((Timestamp)
item.get("THOIGIANBATDAU"));
            bangGiaThuCungDTO.setThoiGianKetThuc((Timestamp)
item.get("THOIGIANKETTHUC"));
            bangGiaThuCungDTO.setGiaKhuyenMai((BigDecimal)
item.get("GIAKM"));
        }
        if (item.get("MACHINHANH") != null) {
            bangGiaThuCungDTO.setMaChiNhanh((int) item.get("MACHINHANH"));
            bangGiaThuCungDTO.setTenChiNhanh((String)
item.get("TENCHINHANH"));
        }
    }
}

```

```

        if (item.get("MATHUCUNG") != null) {
            bangGiaThuCungDTO.setMaThuCung((long) item.get("MATHUCUNG"));
            bangGiaThuCungDTO.setTenThuCung((String) item.get("TENTHUCUNG"));
            bangGiaThuCungDTO.setMoTa((String) item.get("MOTA"));
            bangGiaThuCungDTO.setSoLuongTon((int) item.get("SOLUONGTON"));
            bangGiaThuCungDTO.setGiaHienTai((BigDecimal)
                item.get("GIAHIENTAI"));
            Thucung thucung =
                thuCungService.findById(bangGiaThuCungDTO.getMaThuCung()).get();
            List<Hinhanh> hinhanhList = thucung.getHinhanh();
            if (hinhanhList!=null&&hinhanhList.size()!=0){
                long idHinhAnh = hinhanhList.get(0).getMahinhanh();
                try{
                    byte[] image =
                        storageService.downloadImageFromFileSystem(idHinhAnh);
                    bangGiaThuCungDTO.setHinhAnh(image);
                }
                catch (Exception e){
                    System.out.println(e.getMessage());
                    bangGiaThuCungDTO.setHinhAnh(null);
                }
            } else{
                bangGiaThuCungDTO.setHinhAnh(null);
            }
        }
        if (item.get("MAGIONG") != null) {
            bangGiaThuCungDTO.setMaGiong((int) item.get("MAGIONG"));
            bangGiaThuCungDTO.setTenGiong((String) item.get("TENGIONG"));
        }
        dtoList.add(bangGiaThuCungDTO);
    }
    return new ResponseEntity<>(dtoList, HttpStatus.OK);
}

```

-Ở frontend ReactJs, gọi API lấy danh sách thú cưng và sản phẩm trả về từ Spring Boot:

```

useEffect(() => {
  const fetchProducts = async () => {
    try {
      const response = await AxiosInstance.get("/center/ct-san-pham");
      setProducts(response.data);
    } catch (error) {
      console.error("Error fetching products:", error);
    }
  };

  const fetchPets = async () => {
    try {
      const response = await AxiosInstance.get("/center/ct-thu-cung");
      setPets(response.data);
    } catch (error) {
      console.error("Error fetching pets:", error);
    }
  };
}

```

```

        console.log(response.data)
    } catch (error) {
        console.error("Error fetching pets:", error);
    }
};

fetchProducts();
fetchPets();
}, []);

```

- Khách hàng có thể thêm sản phẩm hoặc thú cưng ngay tại giỏ hàng hoặc trong trang chi tiết của giỏ hàng hoặc thú cưng:



Thức ăn hạt mềm zenith

500000 VND 1000000 VND

Loại : Đồ ăn

Số lượng tồn: 28

THÊM VÀO GIỎ HÀNG

- Khi thêm sản phẩm vào giỏ hàng sẽ gọi đến API xử lý việc thêm sản phẩm và thú cưng vào giỏ hàng trong Spring Boot. Mọi thao tác liên quan đến giỏ hàng sẽ thông qua API:

```
@RequestMapping("/center/gio-hang")
```

- Xử lý thêm sản phẩm và thú cưng vào giỏ hàng:

```
@PostMapping("them-thu-cung")
public ResponseEntity<?> themThuCung(@RequestBody GhThuCungRequest
ghThuCungRequest) {
    try {
        gioHangService.themThuCung(ghThuCungRequest);
        return new ResponseEntity<>("Thêm thành công", HttpStatus.OK);
    } catch (IllegalStateException e) {
        return new ResponseEntity<>("Thêm thất bại" + e.getMessage(),
HttpStatus.BAD_REQUEST);
    }
}

@PostMapping("them-san-pham")
public ResponseEntity<?> themSanPham(@RequestBody GhSanPhamRequest
ghSanPhamRequest) {
    try {
        gioHangService.themSanPham(ghSanPhamRequest);
        return new ResponseEntity<>("Thêm thành công", HttpStatus.OK);
    } catch (IllegalStateException e) {
        return new ResponseEntity<>("Thêm thất bại" + e.getMessage(),
HttpStatus.BAD_REQUEST);
    }
}
```

- Khi đó, ở ReactJs sẽ gọi đến 2 API này để xử lý hàm thêm sản phẩm và thú cưng:

```
const addToCartProducts = async (id) => {
    try {
        await AxiosInstance.post("/center/gio-hang/them-san-pham", {
            maKhachHang: maKhachHang,
            maSanPham: id,
            maChiNhanh: selectedBranch,
        });
        alert("Đã thêm sản phẩm vào giỏ hàng!");
    } catch (error) {
        console.error("Error adding product to cart:", error);
        alert("Có lỗi xảy ra khi thêm sản phẩm vào giỏ hàng!");
    }
};

const addToCartPets = async (id) => {
    try {
        await AxiosInstance.post("/center/gio-hang/them-thu-cung", {
            maKhachHang: maKhachHang,
```

```

        maThuCung: id,
    });
    alert("Đã thêm thú cưng vào giỏ hàng!");
} catch (error) {
    console.error("Error adding pet to cart:", error);
    alert("Có lỗi xảy ra khi thêm thú cưng vào giỏ hàng!");
}
};

```

b. Giỏ hàng.

- Sau khi thêm sản phẩm vào giỏ hàng, khách hàng có thể xem các sản phẩm trong giỏ hàng và tiến hành đặt mua. Giỏ hàng được hiển thị theo Chi Nhánh và khách hàng sẽ đặt mua theo Chi Nhánh đó.

Sản phẩm				
Hình ảnh	Sản Phẩm	Đơn Giá	Số Lượng tồn	
	Thức ăn hạt mềm zenith	500000	28	
	Thức ăn hạt ANF	80000	65	

Thú cưng				
Hình ảnh	Thú Cưng	Đơn Giá	Số Lượng Tồn	
	Mèo Anh lông dài	3000000	5	

- Hàm xử lý thao tác lấy danh sách sản phẩm và thú cưng trong giỏ hàng trong Spring Boot:

```

@PostMapping("/thu-cung")
public ResponseEntity<?> getThuCung(@RequestBody GioHangRequest gioHangRequest) {
    List<ThuCungDTO> list = new ArrayList<>();
    list = gioHangService.thuCungTrongGiohang(gioHangRequest.getMaChiNhanh(), gioHangRequest.getMaKhachHang());
    return new ResponseEntity<>(list, HttpStatus.OK);
}

@PostMapping("/san-pham")
public ResponseEntity<?> getSanPham(@RequestBody GioHangRequest gioHangRequest) {

```

```

List<SanPhamDTO> list = new ArrayList<>();
list = gioHangService.sanPhamTrongGioHang(gioHangRequest.getMaChiNhanh(), 
gioHangRequest.getMaKhachHang());
return new ResponseEntity<>(list, HttpStatus.OK);
}

```

- Khi đó, ở ReactJs sẽ gọi đến 2 API này để lấy danh sách sản phẩm và thú cưng:

```

useEffect(() => {
  const fetchCartItems = async () => {
    if (maKhachHang && selectedBranch) {
      try {
        const responseProducts = await AxiosInstance.post(
          "/center/gio-hang/san-pham",
        {
          maKhachHang: maKhachHang,
          maChiNhanh: selectedBranch,
        }
      );
      const responsePets = await AxiosInstance.post(
        "/center/gio-hang/thu-cung",
      {
        maKhachHang: maKhachHang,
        maChiNhanh: selectedBranch,
      }
      );
      setCartPets(responsePets.data);
      setCartProducts(responseProducts.data);
      console.log(responseProducts.data);
      console.log(responsePets.data);
    } catch (error) {
      console.error("Error fetching cart items:", error);
    }
  }
  fetchCartItems();
}, [maKhachHang, selectedBranch]);

```

- Khách hàng cũng có thể xóa những sản phẩm và thú cưng ra khỏi giỏ hàng.

Tạo hàm xử lý trong Spring Boot:

```

@PostMapping("bo-thu-cung")
public ResponseEntity<?> boThuCung(@RequestBody GhThuCungRequest
ghThuCungRequest) {

```

```

GhthucungPK ghthucungPK = new GhthucungPK();
ghthucungPK.setMakhachhang(ghThuCungRequest.getMaKhachHang());
ghthucungPK.setMathucung(ghThuCungRequest.getMaThuCung());
boolean tontai = gioHangService.existsByIdThuCung(ghthucungPK);
if(!tontai){
    return new ResponseEntity<>("Giỏ hàng thú cung không tồn
tại", HttpStatus.BAD_REQUEST);
}
else{
    gioHangService.deleteByIdThuCung(ghthucungPK);
    return new ResponseEntity<>("Xóa thành công", HttpStatus.OK);
}
}

@PostMapping("bo-san-pham")
public ResponseEntity<?> boSanPham(@RequestBody GhSanPhamRequest
ghSanPhamRequest) {
    GhsanphamPK ghsanphamPK = new GhsanphamPK();
    ghsanphamPK.setMachinhanh(ghSanPhamRequest.getMaChiNhanh());
    ghsanphamPK.setMakhachhang(ghSanPhamRequest.getMaKhachHang());
    ghsanphamPK.setMasanpham(ghSanPhamRequest.getMaSanPham());

    boolean tontai = gioHangService.existsByIdSanPham(ghsanphamPK);
    if(!tontai){
        return new ResponseEntity<>("Giỏ hàng sản phẩm không tồn
tại", HttpStatus.BAD_REQUEST);
    }
    else{
        gioHangService.deleteByIdSanPham(ghsanphamPK);
        return new ResponseEntity<>("Xóa thành công", HttpStatus.OK);
    }
}
}

```

Khi đó, sẽ gọi đến 2 API xóa để thực hiện thao tác xóa sản phẩm:

```

const handleDeleteProduct = async (id) => {
    try {
        await AxiosInstance.post("/center/gio-hang/bo-san-pham", {
            maKhachHang: maKhachHang,
            maSanPham: id,
            maChiNhanh: selectedBranch,
        });
        alert("Đã xóa sản phẩm khỏi giỏ hàng!");
    }

    const responseProducts = await AxiosInstance.post(
        "/center/gio-hang/san-pham",
        {
            maKhachHang: maKhachHang,
            maChiNhanh: selectedBranch,
        }
    );
}

```

```

        setCartProducts(responseProducts.data);
    } catch (error) {
        console.error("Error deleting to cart:", error);
        alert("Có lỗi xảy ra khi xóa sản phẩm!");
    }
};

const handleDeletePet = async (id) => {
    try {
        await AxiosInstance.post("/center/gio-hang/bo-thu-cung", {
            maKhachHang: maKhachHang,
            maThuCung: id,
        });
        alert("Đã xóa thú cưng khỏi giỏ hàng!");

        const responsePets = await AxiosInstance.post(
            "/center/gio-hang/thu-cung",
            {
                maKhachHang: maKhachHang,
                maChiNhanh: selectedBranch,
            }
        );

        setCartPets(responsePets.data);
    } catch (error) {
        console.error("Error deleting to cart:", error);
        alert("Có lỗi xảy ra khi xóa sản phẩm!");
    }
};

```

c. Đặt hàng.

- Khách hàng tiến hành đặt hàng, sẽ hiển thị ra giao diện để xác nhận đơn hàng:

Xác nhận đơn hàng

Thông tin giao hàng:	Thông tin sản phẩm:
Họ và tên	Sản phẩm
Số điện thoại	Thức ăn hạt mềm zenith
Tỉnh/Thành phố	Thức ăn hạt mềm zenith
Quận/Huyện	Thức ăn hạt ANF
Phường/Xã	Thú cưng
Địa chỉ cụ thể	Mèo Anh lông dài
Phương thức thanh toán	Tổng cộng: 3,580,000đ
Thanh toán khi nhận hàng	ĐẶT HÀNG

- Trang đặt hàng sẽ gọi đến API sản phẩm trong giỏ hàng:

```
useEffect(() => {
  const fetchCartItems = async () => {
    if (maKhachHang && selectedBranch) {
      try {
        const responseProducts = await AxiosInstance.post(
          "/center/gio-hang/san-pham",
          {
            maKhachHang: maKhachHang,
            maChiNhanh: selectedBranch,
          }
        );

        const responsePets = await AxiosInstance.post(
          "/center/gio-hang/thu-cung",
          {
            maKhachHang: maKhachHang,
            maChiNhanh: selectedBranch,
          }
        );
        setCartPets(responsePets.data);
        setCartProducts(responseProducts.data);

        const initialProductQuantities = responseProducts.data.reduce(
          (acc, item) => {
            acc[item.maSanPham] = 1;
            return acc;
          }
        );
      } catch (error) {
        console.error(error);
      }
    }
  };
});
```

```

        },
        {}
    );

    const initialPetQuantities = responsePets.data.reduce((acc, item) => {
        acc[item.maThuCung] = 1;
        return acc;
    }, {});

    setProductQuantities(initialProductQuantities);
    setPetQuantities(initialPetQuantities);
} catch (error) {
    console.error("Error fetching cart items:", error);
}
};

fetchCartItems();
}, [maKhachHang, selectedBranch]);

```

- Thao tác đặt hàng sẽ được xử lý gọi đến 3 API trong Spring Boot: Tạo đơn hàng, Thêm vào chi tiết đơn hàng và chi tiết sản phẩm:

```

@PostMapping
public ResponseEntity<?> insert(@RequestBody DonDatDTO donDatDTO) {
    Dondat dondat = new Dondat();
    dondat.setDiachidat(donDatDTO.getDiaChi());
    dondat.setMachinhanh(donDatDTO.getMaChiNhanh());
    dondat.setNgaylap(Timestamp.valueOf(LocalDateTime.now()));
    dondat.setMakhachhang(donDatDTO.getMaKhachhang());
    dondat.setSodienthoai(donDatDTO.getSoDienThoai());
    dondat.setTrangthai(Boolean.FALSE);
    try {
        dondat = donDatService.save(dondat);
        DonDatDTO donDatDTO2 = donDatService.convertToDTO(dondat);
        return new ResponseEntity<>(donDatDTO2, HttpStatus.OK);
    } catch (Exception e) {
        return new ResponseEntity<>("Thêm thất bại", HttpStatus.BAD_REQUEST);
    }
}

```

Thêm vào Chi tiết sản phẩm:

```

@PostMapping("/sp")
public ResponseEntity<?> themSP(@RequestBody List<DonDatSanPhamRequest> list)
{
    try {
        for (DonDatSanPhamRequest item : list) {
            Ctmuasanpham ctmuasanpham = new Ctmuasanpham();
            CtmuasanphamPK ctmuasanphamPK = new CtmuasanphamPK();
            ctmuasanphamPK.setSodondat(item.getMaDonDat());

```

```

        ctmuasanphamPK.setMachinhanh(item.getMaChiNhanh());
        ctmuasanphamPK.setMasanpham(item.getMaSanPham());
        ctmuasanpham.setId(ctmuasanphamPK);
        ctmuasanpham.setSoluong(item.getSoLuong());
        ctmuasanpham.setDongia(item.getDonGia());

        Optional<DonDat> dondat =
donDatService.findById(item.getMaDonDat());

        if (dondat.isPresent()) {
            DonDat dondat1 = dondat.get();
            System.out.println(dondat1.getMakhachhang());
            ctmuasanpham.setDonDat(dondat1);
            ctmuasanpham.setMasanpham(item.getMaSanPham());
            ctmuasanpham.setMachinhanh(item.getMaChiNhanh());

            try {
                ctmuasanpham = ctMuaSanPhamService.save(ctmuasanpham);
            } catch (Exception e) {
                e.printStackTrace();
                return new ResponseEntity<>("Thêm thất bại: " +
e.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR);
            }
        } else {
            System.out.println("DonDat not found");
            return new ResponseEntity<>("Thêm thất bại: DonDat không tìm
thấy", HttpStatus.BAD_REQUEST);
        }
    }
}

```

Thêm vào chi tiết đơn đặt thú cưng:

```

@PostMapping("/tc")
public ResponseEntity<?> themTC(@RequestBody List<DonDatThuCungRequest> list)
{
    try {
        for (DonDatThuCungRequest item : list) {
            Ctmuathucung ctmuathucung = new Ctmuathucung();
            CtmuathucungPK ctmuathucungPK = new CtmuathucungPK();
            ctmuathucungPK.setMadondat(item.getMaDonDat());
            ctmuathucungPK.setMathucung(item.getMaThuCung());
            ctmuathucung.setId(ctmuathucungPK);
            ctmuathucung.setDongia(item.getDonGia());
            ctmuathucung.setSoluong(item.getSoLuong());

            ctmuathucung.setDonDat(donDatService.findById(item.getMaDonDat()).get());
            ctmuathucung = ctMuaThuCungService.save(ctmuathucung);
        }
        return new ResponseEntity<>("Thêm thành công", HttpStatus.OK);
    } catch (Exception e) {

```

```

        System.out.println(e.getMessage());
        return new ResponseEntity<>("Thêm thất bại", HttpStatus.BAD_REQUEST);
    }
}

```

-Ở ReactJs sẽ gọi đến các API này để xử lý đặt hàng:

```

const placeOrder = async () => {
    try {
        const orderInfoData = {
            diaChi:
                specificAddress +
                ", " +
                selectedWard.full_name +
                ", " +
                selectedDistrict.full_name +
                ", " +
                selectedProvince.full_name,
            soDienThoai: phoneNumber,
            maChiNhanh: selectedBranch,
            maKhachhang: maKhachHang,
        };
        console.log(orderInfoData);

        const orderResponse = await AxiosInstance.post(
            "/order/don-dat",
            orderInfoData
        );
        const maDonDat = orderResponse.data.soDonDat;
        // Tạo danh sách các sản phẩm từ cartProducts

        const orderProducts = cartProducts.map((product) => ({
            maDonDat: maDonDat,
            soLuong: productQuantities[product.maSanPham] || 1,
            donGia: product.giaKM ? product.giaKM : product.giaHienTai,
            maSanPham: product.maSanPham,
            maChiNhanh: selectedBranch,
        }));
        await AxiosInstance.post("/order/dat-hang/sp", orderProducts);

        const orderPets = cartPets.map((pet) => ({
            maDonDat: maDonDat,
            soLuong: petQuantities[pet.maThuCung] || 1,
            donGia: pet.giaKM ? pet.giaKM : pet.giaHienTai,
            maThuCung: pet.maThuCung,
        }));
    }
}

```

```
}));

console.log(orderPets);
await AxiosInstance.post("/order/dat-hang/tc", orderPets);
alert("Đã đặt hàng thành công!");

const responseCart = await AxiosInstance.post(
  "/center/gio-hang/bo-tat-ca",
{
  maKhachHang: maKhachHang,
  maChiNhanh: selectedBranch,
}
);
setCartPets(responseCart.data);
setCartProducts(responseCart.data);

navigate("/cart");
} catch (error) {
  alert("Đặt hàng thất bại!");
  console.error("Đặt hàng thất bại:", error.message);
  // Hiển thị thông báo lỗi cho người dùng
}
};
```

Chương VI: Triển khai trên Mobile Android

1. Quản lý

Quản lý sản phẩm và thú cưng, loại sản phẩm, loại thú cưng, giống, bảng giá, chi nhánh và nhân viên, các chức năng có thao tác quản lý tương đương nhau.

a. Load dữ liệu

- API lấy danh sách sản phẩm:

```
@GetMapping  
    public ResponseEntity<List<SanPhamDTO>> getAll() {  
        List<Ctsanpham> list = chiTietSanPhamService.findAll();  
        List<SanPhamDTO> dtoList = new ArrayList<>();  
        for (Ctsanpham item : list) {  
            SanPhamDTO sanPhamDTO = convertToDTO(item);  
            dtoList.add(sanPhamDTO);  
        }  
        return new ResponseEntity<>(dtoList, HttpStatus.OK);  
    }
```

- Đoạn mã convert dữ liệu sang dạng chuẩn giao tiếp DTO

```
public SanPhamDTO convertToDTO(Ctsanpham ctsanpham) {  
    SanPhamDTO sanPhamDTO = new SanPhamDTO();  
    if (ctsanpham == null) {  
        return sanPhamDTO;  
    }  
    sanPhamDTO.setMaSanPham(ctsanpham.getSanpham().getMasanpham());  
    sanPhamDTO.setTenSanPham(ctsanpham.getSanpham().getTensanpham());  
    sanPhamDTO.setGiaHienTai(ctsanpham.getSanpham().getGiahientai());  
    sanPhamDTO.setLoaiSanPham(new LoaiSanPhamDTO());  
    if (ctsanpham.getSanpham().getLoaisanpham() != null) {  
        sanPhamDTO.getLoaiSanPham().setMaLoaiSanPham(ctsanpham.getSanpham().getLoaisanpham().getMaloaisanpham());  
        sanPhamDTO.getLoaiSanPham().setTenLoaiSanPham(ctsanpham.getSanpham().getLoaisanpham().getTenloaisanpham());  
    }  
    if (ctsanpham.getSanpham().getHinhanh() != null && !ctsanpham.getSanpham().getHinhanh().isEmpty()) {  
        sanPhamDTO.setHinhAnh(new ArrayList<>());  
        for (Hinhanh item : ctsanpham.getSanpham().getHinhanh()) {  
            sanPhamDTO.getHinhAnh().add(item.getMahinhanh());  
        }  
    }  
    if (ctsanpham.getChinhanh() != null) {  
        sanPhamDTO.setMaChiNhanh(ctsanpham.getChinhanh().getMachinhanh());  
    }  
    sanPhamDTO.setSoLuongTon(ctsanpham.getSoluongton());  
    return sanPhamDTO;  
}
```

- Đoạn mã Retrofit ở Android phía client để lấy dữ liệu thông qua API:

+ Cấu hình Retrofit theo mẫu Singleton:

```
public class ApiClient {  
    private static final String API_PATH ="http://10.252.9.249:8989/";  
    private static final Gson GSON =new GsonBuilder()  
        .setLenient()  
        .setDateFormat("yyyy-MM-dd'T'HH:mm:ss.SSSX")  
        .create();  
    private static ApiClient apiClient;  
    private String authToken = "";  
    private Retrofit retrofit = null;  
  
    private ApiClient(){  
    }  
  
    public void setAuthToken(String authToken){  
        this.authToken=authToken;  
    }  
  
    public static ApiClient getApiClient(){  
        if(apiClient==null){  
            apiClient=new ApiClient();  
        }  
        return apiClient;  
    }  
  
    public Retrofit getRetrofit(){  
        if(retrofit == null){  
            OkHttpClient client = new OkHttpClient.Builder().addInterceptor(new Interceptor() {  
                @Override  
                public Response intercept(Chain chain) throws IOException {  
                    Request newRequest = chain.request().newBuilder()  
                        .addHeader("Authorization", "Bearer " + authToken)  
                        .build();  
                    return chain.proceed(newRequest);  
                }  
            }).build();  
            retrofit= new Retrofit.Builder()  
                .baseUrl(API_PATH)  
                .addConverterFactory(GsonConverterFactory.create(GSON))  
                .build();  
        }  
        return retrofit;  
    }  
    public void deleteRetrofit(){  
        retrofit=null;  
    }  
}
```

+ Cấu hình interface SanPhamService:

```
public interface SanPhamService {  
    @GET("center/sanpham")  
    Call<List<SanPham>> getAll();  
  
    @GET("center/sanpham/{id}")  
    Call<SanPham> getOneById(@Path("id") String id);  
  
    @POST("center/sanpham")  
    Call<SanPham> insert(@Body SanPham sanPham);  
  
    @PUT("center/sanpham")  
    Call<SanPham> update(@Body SanPham sanPham);  
  
    @DELETE("center/sanpham/{sanpham}/{chinhanh}")  
    Call<ResponseBody> delete(@Path("sanpham") Long maSanPham,  
        @Path("chinhanh") int maChiNhanh);  
  
    @PUT("center/sanpham/soluongton")  
    Call<ResponseBody> updateSL(@Body SanPham sanPham);  
}
```

+ Gọi Retrofit để tạo service:

```
ApiClient apiClient = ApiClient.getApiClient();
```

```
SanPhamService sanPhamService = apiClient.getRetrofit().create(SanPhamService.class);
```

+ Hàm sử dụng Retrofit đọc dữ liệu:

```
public void DocDL(){  
    System.out.println("DocDLSanPham");  
    sanPhamService.getAll().enqueue(new Callback<List<SanPham>>() {  
        @Override  
        public void onResponse(Call<List<SanPham>> call, Response<List<SanPham>> response) {  
            if (response.code() == 200) {  
                data.clear();  
                for (SanPham x : response.body()) {  
                    data.add(x);  
                }  
                sanPhamManageAdapter.notifyDataSetChanged();  
            } else {  
                try {  
                    int code = response.code();  
                    String message = response.message();  
                    String error = response.errorBody().string();  
                    SendMessage.sendMessageFail(mView.getContext(), code, error, message);  
                } catch (Exception e) {  
                    SendMessage.sendCatch(mView.getContext(), e.getMessage());  
                }  
            }  
        }  
        @Override  
        public void onFailure(Call<List<SanPham>> call, Throwable throwable) {  
            SendMessage.sendMessageFail(mView.getContext(), throwable);  
        }  
    });  
}
```

- Dữ liệu sau khi đã vào ArrayList lưu theo đối tượng load thì sẽ được set cho adapter để hiển thị lên đối tượng ListView trong Android Studio.

```
sanPhamManageAdapter=new  
SanPhamManageAdapter(mView.getContext(),R.layout.item_sanpham_manage,  
        data,chiNhanhMap, loaiSanPhamList, tenLoaiSanPham);  
lvSanPham.setAdapter(sanPhamManageAdapter);
```

- Kết quả giao diện:



b. Thêm dữ liệu.

- Hàm thêm dữ liệu sản phẩm mới trong Spring Boot:

```
@PostMapping
public ResponseEntity<?> insert(@RequestBody SanPhamDTO sanPhamDTO) {
    try {
        Sanpham sp = new Sanpham();
        Ctsanpham ctsanpham = new Ctsanpham();
        CtsanphamPK ctsanphamPK = new CtsanphamPK();

        sp.setTensanpham(sanPhamDTO.getTenSanPham());
        sp.setGiahientai(sanPhamDTO.getGiaHienTai());

        sp.setLoaisanpham(loaiSanPhamService.findById(sanPhamDTO.getLoaiSanPham()).getMaLoaiSanPham()
            .orElse(null));
        sp = sanPhamService.save(sp);
        ctsanpham.setMasanpham(sp.getMasanpham());
        ctsanpham.setMachinhanh(sanPhamDTO.getMaChiNhanh());

        if (!chiNhanhService.existById(sanPhamDTO.getMaChiNhanh())) {
            return new ResponseEntity<>("Chi nhánh không tồn tại",
                HttpStatus.BAD_REQUEST);
        }
        ChiNhanh chinhanh = chiNhanhService.findById(sanPhamDTO.getMaChiNhanh()).get();

        ctsanpham.setId(ctsanphamPK);
        ctsanpham.setChinhanh(chinhanh);
        ctsanpham.setSanpham(sp);
        ctsanpham.setSoluongton(sanPhamDTO.getSoLuongTon());
        chiTietSanPhamService.save(ctsanpham);
        SanPhamDTO sanPhamDT01 = convertToDTO(ctsanpham);
        return new ResponseEntity<>(sanPhamDT01, HttpStatus.OK);
    } catch (Exception e) {
        System.out.println(e.getMessage());
        return new ResponseEntity<>("Thêm thất bại", HttpStatus.BAD_REQUEST);
    }
}
```

- Dùng Retrofit để thêm dữ liệu mới:

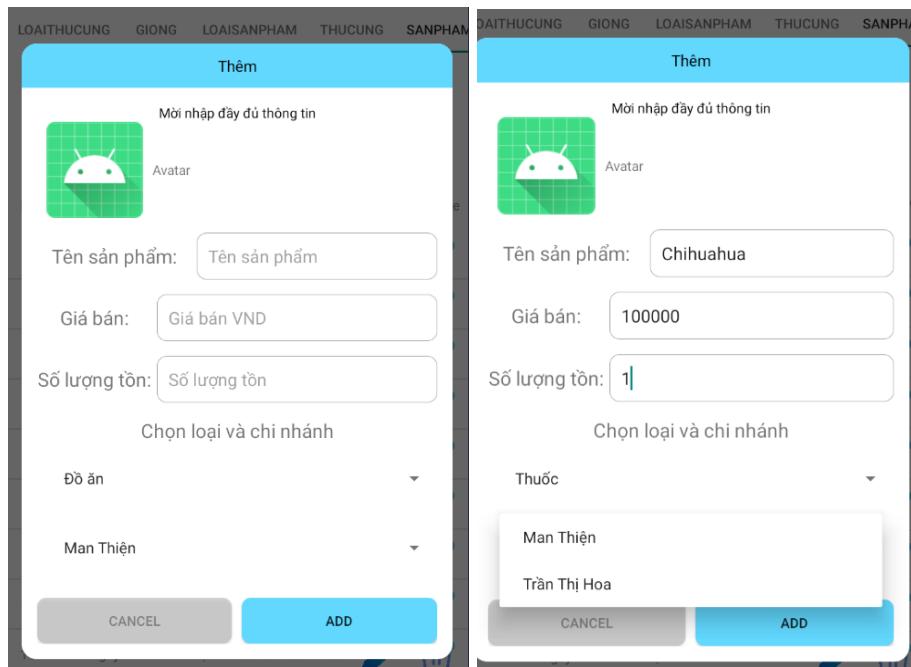
```
sanPhamService.insert(sanPham).enqueue(new Callback<SanPham>() {
    @Override
    public void onResponse(Call<SanPham> call, Response<SanPham> response) {
        if(response.code()== 200){
            SanPham sanPhamMoi = response.body();
            if (mUri!=null){
                sendImage("", "", "", String.valueOf(sanPhamMoi.getMaSanPham()));
            }
            DocDL();
            Toast.makeText(mView.getContext(), "Thêm thành công", Toast.LENGTH_SHORT).show();
            dialog.dismiss();
        } else{
            try {
                int code = response.code();
                String message = response.message();
                String error = response.errorBody().string();
                SendMessage.sendMessageFail(mView.getContext(),code,error,message);
            } catch (Exception e) {
                SendMessage.sendCatch(mView.getContext(),e.getMessage());
            }
        }
    }

    @Override
    public void onFailure(Call<SanPham> call, Throwable throwable) {
        SendMessage.sendApiFail(mView.getContext(),throwable);
    }
});
```

- sanPham được thêm từng field dựa trên những EditText và input khác mà Android Studio cung cấp như Spinner...

```
sanPham.setTenSanPham(tenSanPham);
sanPham.setGiaHienTai(convertBigDecimal(giaHienTai));
sanPham.setSoLuongTon(Integer.parseInt(soLuongTon));
for(Map.Entry<Integer, String> x:chiNhanhMap.entrySet()) {
    if (x.getValue().equals(tenChiNhanh)) {
        sanPham.setMachiNhanh(x.getKey());
        break;
    }
}
for(LoaiSanPham x: loaiSanPhamList){
    if(x.getTenLoaiSanPham().equals(tenLoaiSanPham)){
        sanPham.setLoaiSanPham(x);
        break;
    }
}
```

- Kết quả giao diện thêm:



- Khi thêm thành công, hệ thống dùng Toast để thông báo cho người dùng:

#	Tên sản phẩm	Chi nhánh	Giá	Số	Edit	Delete
1	Thức ăn hạt	Man Thiện	Liên hệ	30		
2	Thức ăn hạt	Man Thiện	100000.0000	65		
3	Thức ăn hạt	Man Thiện	400000.0000	100		
4	Thức ăn	Man Thiện	300000.0000	100		
5	Thức ăn hạt	Man Thiện	200000.0000	150		
6	Sữa Royal	Man Thiện	600000.0000	150		
7	Sữa Úc pet	Man Thiện	1000000.0000	170		
8	Sữa Petlac thú	Man Thiện	2000000.0000	80		
9	Nhỏ gáy	Man Thiện	50000.0000	80		

c. Sửa thông tin

- Hàm xử lý lần gọi ở Spring Boot:

```
@PutMapping
public ResponseEntity<?> update(@RequestBody SanPhamDTO sanPhamDTO) {
    Sanpham sp = new Sanpham();
    if (!sanPhamService.existsById(sanPhamDTO.getMaSanPham())) {
        return new ResponseEntity<>(sp, HttpStatus.BAD_REQUEST);
    }
    sp.setMasanpham(sanPhamDTO.getMaSanPham());
    sp.setGiahtientai(sanPhamDTO.getGiaHienTai());
    sp.setTensanpham(sanPhamDTO.getTenSanPham());

    sp.setLoaisanpham(loaiSanPhamService.findById(sanPhamDTO.getLoaiSanPham()).getMaLoaiSanPham().orElse(null));
    sp = sanPhamService.save(sp);
    Ctsanpham ctsanpham = chiTietSanPhamService.findById(new
    CtsanphamPK(sanPhamDTO.getMaChiNhanh(), sanPhamDTO.getMaSanPham())).get();
    SanPhamDTO sanPhamDTO1 = convertToDTO(ctsanpham);
    return new ResponseEntity<>(sanPhamDTO1, HttpStatus.OK);
}
```

- Phần xử lý sửa thông tin sản phẩm cũng tương tự như thêm một sản phẩm mới, trước đó hệ thống cần cập nhật các field của sản phẩm lên trên EditText, Spinner... để hiển thị thông tin dòng sản phẩm trong hệ thống. Lần gọi update sản phẩm từ Retrofit SanPhamService.

```
sanPhamService.update(sanPhamTemp).enqueue(new Callback<SanPham>() {
    @Override
    public void onResponse(Call<SanPham> call, Response<SanPham> response) {
        if(response.code() == 200) {
            if(soLuongTon!=sanPhamTemp.getSoLuongTon()){
                capNhatSoLuongTon(sanPhamTemp);
            } else{
                notifyDataSetChanged();
                Toast.makeText(mView.getContext(), "Cập nhật thành công", Toast.LENGTH_SHORT).show();
            }
        } else{
            try {
                int code = response.code();
                String message = response.message();
                String error = response.errorBody().string();
                SendMessage.sendMessageFail(mView.getContext(),code,error,message);
            } catch (Exception e) {
                SendMessage.sendCatch(mView.getContext(),e.getMessage());
            }
        }
    }
    @Override
    public void onFailure(Call<SanPham> call, Throwable throwable) {
        SendMessage.sendApiFail(mView.getContext(),throwable);
    }
});
```

- Kết quả:

The image shows two identical-looking update forms side-by-side. Both forms have a blue header bar with the word "Sửa" (Edit) in white. Below the header, there are four pairs of input fields, each consisting of a label and a text input box. The first pair is "Mã sản phẩm:" with value "7". The second pair is "Tên sản phẩm:" with value "Thức ăn hạt Ganador". The third pair is "Giá bán:" with value "200000.0000". The fourth pair is "Số lượng tồn:" with value "150". Below these input fields are two dropdown menus labeled "Chọn loại và chi nhánh" with options "Đồ ăn" and "Man Thiện". At the bottom of each form are two buttons: a grey "CANCEL" button and a blue "UPDATE" button.

- Thông báo thành công:

The image shows a table of products with a modal overlay. The modal has a blue header bar with the word "THÊM" (Add). The table below has columns: Số (Number), Tên sản phẩm (Product Name), Chi nhánh (Branch), Giá (Price), Số (Quantity), Edit icon, and Delete icon. The data rows are:

Số	Tên sản phẩm	Chi nhánh	Giá	Số	Edit	Delete
9	Chihuahua	Man Thiện	100000.0000	1		
1	Nhỏ gáy	Trần Thị Hoa	50000.0000	90		
2	Nhỏ gáy	Trần Thị Hoa	50000.0000	100		
3	Nhỏ gáy trí ve	Trần Thị Hoa	70000.0000	60		
4	Nhỏ gáy ngứa	Trần Thị Hoa	100000.0000	150		
5	Cát đậu nành	Trần Thị Hoa	70000.0000	152		
6	Cây lăn lồng	Trần Thị Hoa	150000.0000	82		
7	Bim tâ cho	Trần Thị Hoa	200000.0000	99		
8	Tấm lót sàn	Trần Thị Hoa	24234	130		

A light blue circular overlay at the bottom center contains the text "Cập nhật thành công" (Update successful).

d. Xóa thông tin.

- Hàm xử lý gọi thao tác xóa ở Spring Boot:

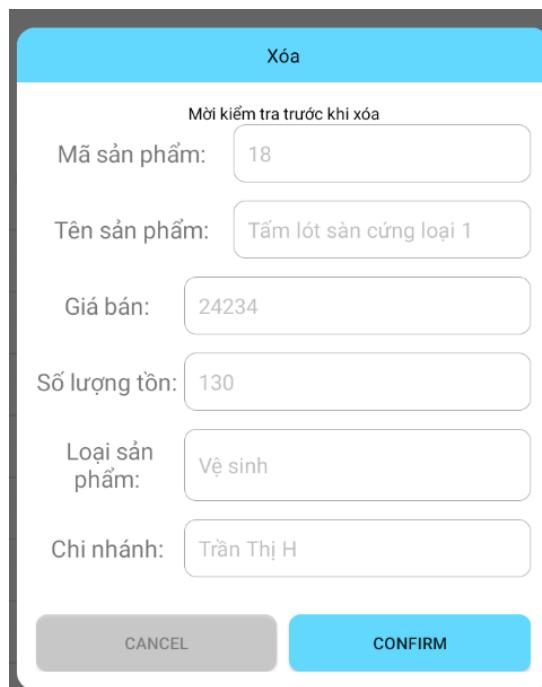
```
@DeleteMapping("/{sanpham}/{chinhanh}")
    public ResponseEntity<?> delete(@PathVariable Long sanpham,@PathVariable int chinhanh) {
        CtsanphamPK ctsanphamPK = new CtsanphamPK(chinhanh,sanpham);
        boolean spTonTai = chiTietSanPhamService.existsById(ctsanphamPK);
        if (spTonTai == false) {
            return new ResponseEntity<>("Id khong ton tai", HttpStatus.NOT_FOUND);
        }
        Optional<Ctsanpham> ctsanpham = chiTietSanPhamService.findById(ctsanphamPK);
        chiTietSanPhamService.delete(ctsanpham.orElse(null));
        spTonTai = chiTietSanPhamService.existsById(ctsanphamPK);
        if (spTonTai == true) {
            return new ResponseEntity<>("Xoa that bai", HttpStatus.BAD_REQUEST);
        } else {
            return new ResponseEntity<>("Xoa thanh cong", HttpStatus.OK);
        }
    }
```

- Khi xóa thông thường hệ thống sẽ tạo một dialog để hiển thị những thông tin người dùng xem qua và xác nhận xóa.

- Mã lệnh Retrofit xử lý thao tác xóa:

```
btnConfirm.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        sanPhamService.delete(sanPham.getMaSanPham(),sanPham.getMaChiNhanh()).enqueue(new Callback<ResponseBody>() {
            @Override
            public void onResponse(Call<ResponseBody> call, Response<ResponseBody> response) {
                try{
                    if(response.code() == 200){
                        String result = response.body().string();
                        data.remove(sanPham);
                        notifyDataSetChanged();
                        Toast.makeText(mView.getContext(),result,Toast.LENGTH_SHORT).show();
                    }
                    else{
                        try {
                            int code = response.code();
                            String message = response.message();
                            String error = response.errorBody().string();
                            SendMessage.sendMessageFail(mView.getContext(),code,error,message);
                        } catch (Exception e) {
                            SendMessage.sendCatch(mView.getContext(),e.getMessage());
                        }
                    }
                }
                catch (Exception e){
                    SendMessage.sendCatch(mView.getContext(),e.getMessage());
                }
            }
            @Override
            public void onFailure(Call<ResponseBody> call, Throwable throwable) {
                SendMessage.sendApiFail(mView.getContext(),throwable);
            }
        });
        dialog.dismiss();
    }
});
```

- Kết quả:



- Thông báo trả về:

OAITHUCUNG GIONG LOAISANPHAM THUCUNG SANPHAM

THÊM

Mã	Tên sản phẩm	Chi nhánh	Giá	Số	Edit	Delete
18	Tấm lót sàn	Man Thiện	Liên hệ	10		
19	Chihuahua	Man Thiện	100000.0000	1		
11	Nhỏ gáy	Trần Thị Hoa	50000.0000	90		
12	Nhỏ gáy	Trần Thị Hoa	50000.0000	100		
13	Nhỏ gáy trị ve	Trần Thị Hoa	70000.0000	60		
14	Nhỏ gáy ngứa	Trần Thị Hoa	100000.0000	150		
15	Cát đậu nành	Trần Thị Hoa	70000.0000	152		
16	Cây lăn lông	Trần Thị Hoa	150000.0000	82		
17	Bim tă cho	Trần Thị Hoa	200000.0000	99		

Xoá thành công

e. Hình ảnh

- Hình ảnh trong Spring Boot cho phép nhận về nhiều file ảnh khác nhau và lưu vào thư mục cũ bộ bên trong project, trên database chỉ cần lưu trữ đường dẫn và tên hình ảnh. Hình ảnh được liên kết và sử dụng trong 2 service center và identity (sản phẩm, thú cưng và nhân viên, khách hàng).

- Hàm lưu hình ảnh trong Spring Boot:

```
@PostMapping
public ResponseEntity<?> saveImage(@RequestParam("image") MultipartFile[] file,
                                         @RequestParam(value = "maNhanVien", required = false) String maNhanVien,
                                         @RequestParam(value = "maKhachHang", required = false) String maKhachHang,
                                         @RequestParam(value = "maThuCung", required = false) Long maThuCung,
                                         @RequestParam(value = "maSanPham", required = false) Long maSanPham
) throws IOException {
    try {
        for (MultipartFile item : file) {
            storageService.uploadImageToFileSystem(item, maNhanVien, maKhachHang, maThuCung, maSanPham);
        }
        return new ResponseEntity<>("Lưu thành công", HttpStatus.OK);
    } catch (IOException e) {
        return new ResponseEntity<>("Lưu thất bại", HttpStatus.BAD_REQUEST);
    }
}
```

- Với Android Studio, ta cần cấp phép quyền truy cập bộ nhớ cho hệ thống:

+ Cấu hình Manifest:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

<application
    android:name="com.example.petshopapp.paypal.PaypalApp"
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.AppCompat.Light.NoActionBar"
    android:usesCleartextTraffic="true"
    tools:targetApi="31">
```

+ Hàm kiểm tra và hỏi quyền truy cập:

```
private void onClickRequestPermission(){
    if(Build.VERSION.SDK_INT< Build.VERSION_CODES.M){
        getGallery();
        return;
    }
    if(this.getContext().checkSelfPermission(Manifest.permission.READ_EXTERNAL_STORAGE)==
PackageManager.PERMISSION_GRANTED){
        getGallery();
    }
    else{
        String [] permission = {Manifest.permission.READ_EXTERNAL_STORAGE};
        this.requestPermissions(permission,MY_REQUEST_CODE);
    }
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if(requestCode==MY_REQUEST_CODE){
        if(grantResults.length>0&&grantResults[0]==PackageManager.PERMISSION_GRANTED){
            getGallery();
        }
    }
}
```

- Hàm lấy hình trong thư viện điện thoại (Android Studio):

```
private void getGallery(){
    Intent intent = new Intent(Intent.ACTION_PICK);
    intent.setType("image/*");
    mActivityResultLauncher.launch(Intent.createChooser(intent, "Select picture"));
}
```

- Hàm nhận kết quả trả về khi đã chọn hình:

```
private ActivityResultLauncher<Intent> mActivityResultLauncher = registerForActivityResult(
    new ActivityResultContracts.StartActivityForResult(),
    new ActivityResultCallback<ActivityResult>() {
        @Override
        public void onActivityResult(ActivityResult result) {
            Log.e(NhanVienTab.class.getName(),"onActivityResult");
            if(result.getResultCode() == Activity.RESULT_OK){
                Intent data = result.getData();
                if(data == null ){
                    return;
                }
                Uri uri = data.getData();
                mUri=uri;
                try{
                    bitmap = MediaStore.Images.Media.getBitmap(getApplicationContext().getContentResolver(),uri);
                    ivAvatar.setImageBitmap(bitmap);
                } catch (FileNotFoundException e) {
                    Log.e("FileNotFoundException", e.getMessage());
                    Toast.makeText(getApplicationContext(),"FileNotFoundException" + e.getMessage(), Toast.LENGTH_SHORT).show();
                    bitmap=null;
                    ivAvatar.setImageResource(R.mipmap.ic_launcher);
                } catch (IOException e) {
                    Log.e("IOException", e.getMessage());
                    Toast.makeText(getApplicationContext(),"IOException" + e.getMessage(), Toast.LENGTH_SHORT).show();
                    bitmap=null;
                    ivAvatar.setImageResource(R.mipmap.ic_launcher);
                }
            }
        }
    });
};
```

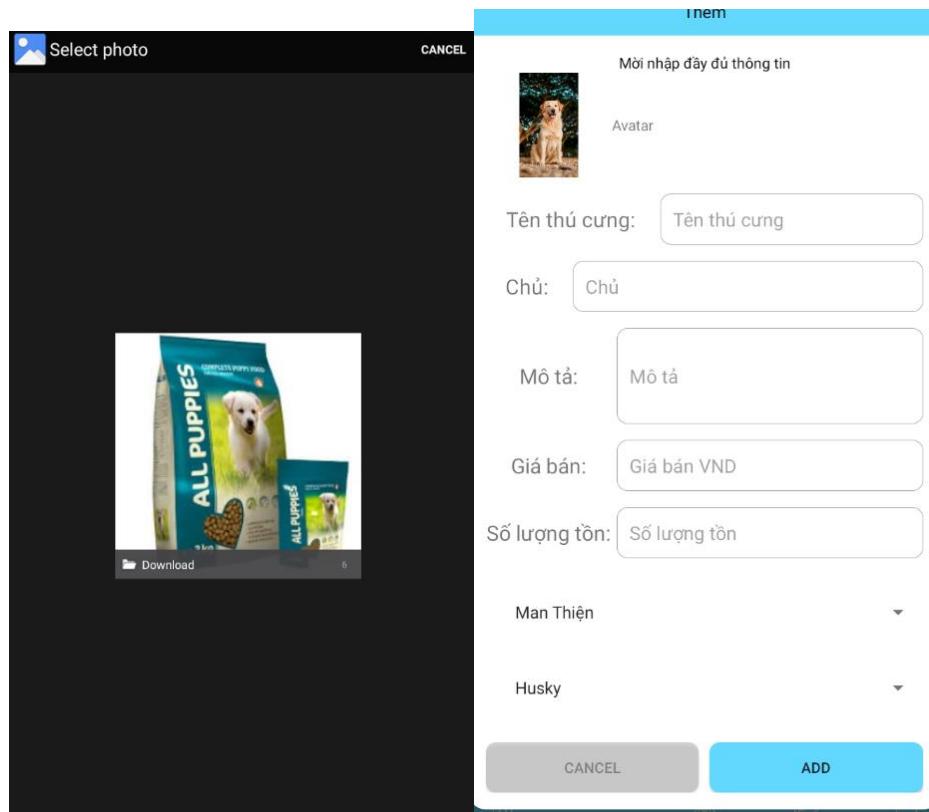
- Lệnh gọi Retrofit để lưu hình:

```
private void sendImage(String maNhanVien, String maKhachHang, String maThuCung, String maSanPham){
    RequestBody requestBodyMaNhanVien = RequestBody.create(MediaType.parse("multipart/form-data"), maNhanVien);
    RequestBody requestBodyMaKhachHang = RequestBody.create(MediaType.parse("multipart/form-data"), maKhachHang);
    RequestBody requestBodyMaThuCung = RequestBody.create(MediaType.parse("multipart/form-data"), maThuCung);
    RequestBody requestBodyMaSanPham = RequestBody.create(MediaType.parse("multipart/form-data"), maSanPham);

    String imgRealPath = RealPathUtil.getRealPath(this.getContext(), mUri);
    File file = new File(imgRealPath);
    RequestBody requestBodyAvatar = RequestBody.create(MediaType.parse("multipart/form-data"), file);
    MultipartBody.Part multipartBodyAvatar = MultipartBody.Part.createFormData(Const.KEY_IMAGE, file.getName(),
        requestBodyAvatar);

    hinhAnhService.saveImageCenter(multipartBodyAvatar, requestBodyMaNhanVien, requestBodyMaKhachHang,
        requestBodyMaThuCung, requestBodyMaSanPham).enqueue(new Callback<ResponseBody>() {
        @Override
        public void onResponse(Call<ResponseBody> call, Response<ResponseBody> response) {
            try{
                if(response.code() == 200){
                    String result = response.body().string();
                    Toast.makeText(mView.getContext(),result,Toast.LENGTH_SHORT).show();
                }
                else{
                    try {
                        int code = response.code();
                        String message = response.message();
                        String error = response.errorBody().string();
                        SendMessage.sendMessageFail(mView.getContext(),code,error,message);
                    } catch (Exception e) {
                        SendMessage.sendCatch(mView.getContext(),e.getMessage());
                    }
                }
            }
            catch (Exception e){
                SendMessage.sendCatch(mView.getContext(),e.getMessage());
            }
        }
        @Override
        public void onFailure(Call<ResponseBody> call, Throwable throwable) {
            SendMessage.sendApiFail(mView.getContext(),throwable);
        }
    });
}
```

- Kết quả giao diện sau khi lấy được hình ảnh:



- Với việc nhận hình ảnh, hàm yêu cầu truyền đi một mảng long[] và trả về danh sách hình ảnh cùng với định dạng byte, hàm xử lý ở API:

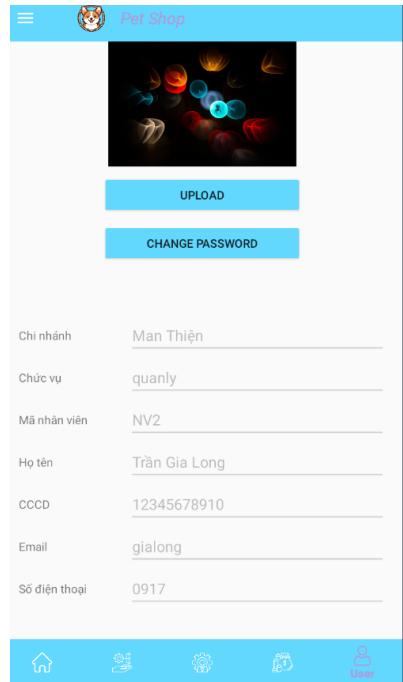
```
@PostMapping("/get")
public ResponseEntity<?> sendImage(@RequestBody long[] id) throws IOException {
    try{
        List<HinhAnhDTO> hinhanhDTOS = new ArrayList<>();
        for (long itemId : id) {
            byte[] image = storageService.downloadImageFromFileSystem(itemId);
            HinhanhDTO hinhanhDTO = new HinhanhDTO();
            hinhanhDTO.setMaHinhAnh(itemId);
            hinhanhDTO.setSource(image);
            hinhanhDTOS.add(hinhanhDTO);
        }
        return new ResponseEntity<>(hinhanhDTOS, HttpStatus.OK);
    }
    catch(Exception e){
        System.out.println(e.getMessage());
        return new ResponseEntity<>("Lỗi lấy hình", HttpStatus.BAD_REQUEST);
    }
}
```

- Hàm call API:

```
private void getImage(){
    if(khachHang!=null && khachHang.getHinhAnh()!=null&&khachHang.getHinhAnh().size()!=0){
        long idHinh = khachHang.getHinhAnh().get(0);
        hinhanhService.getImage(new long[]{idHinh}).enqueue(new Callback<List<HinhAnh>>() {
            @Override
            public void onResponse(Call<List<HinhAnh>> call, Response<List<HinhAnh>> response) {
                try{
                    if(response.code() == 200){
                        List<HinhAnh> list = response.body();
                        String source = list.get(0).getSource();
                        bitmap= ImageInteract.convertStringToBitmap(source);
                        if(bitmap == null){
                            Toast.makeText(getApplicationContext(),"Bitmap null",Toast.LENGTH_SHORT).show();
                            return;
                        }
                        ivAvatar.setImageBitmap(bitmap);
                    }
                    else{
                        String message="Lỗi: "+String.valueOf(response.code())
                        +"\\n"+Chi tiết: "+ response.errorBody().string();
                        Log.e("ERROR","Call api fail: "+message);
                    }
                } catch (Exception e){
                    SendMessage.sendCatch(mView.getContext(),e.getMessage());
                }
            }

            @Override
            public void onFailure(Call<List<HinhAnh>> call, Throwable throwable) {
                SendMessage.sendApiFail(mView.getContext(),throwable);
            }
        });
    }
}
```

- Kết quả giao diện trang cá nhân của User với hình ảnh được lấy bằng lệnh gọi trên:



2. Quản lý chi tiết

- Với chức năng quản lý bảng giá và nhập hàng, thông tin cần thể hiện chi tiết của một phiếu hoặc bảng, cho phép người dùng nhập liệu hoặc thay đổi chi tiết bên trong.

- Mã sử dụng FragmentManager để chuyển trang fragment:

```
private void openChiTietTab(long idBangGia, int maChiNhanh){
    llND.setVisibility(View.GONE);
    BangGiaChiTietTab newFragment = new BangGiaChiTietTab();
    Bundle bundle = new Bundle();
    bundle.putLong("idBangGia", idBangGia);
    bundle.putInt("maChiNhanh", maChiNhanh);
    newFragment.setArguments(bundle);
    FragmentManager fragmentManager = getActivity().getSupportFragmentManager();
    fragmentManager.addOnBackStackChangedListener(new FragmentManager.OnBackStackChangedListener() {
        private int previousCount = fragmentManager.getBackStackEntryCount();
        @Override
        public void onBackStackChanged() {
            int currentCount = fragmentManager.getBackStackEntryCount();
            if (currentCount < previousCount) {
                Toast.makeText(mView.getContext(), "Trở về thành công", Toast.LENGTH_SHORT).show();
                close();
            }
            previousCount = currentCount;
        }
    });
    FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
    fragmentTransaction
        .replace(R.id.flContainer, newFragment)
        .addToBackStack(null)
        .commit();
}
```

- Kết quả giao diện chi tiết bảng giá:

Thú cưng		Sản phẩm	
1	Chó Husky	Husky	10000000.0000 <u>6000000.00</u>
2	Chó Chihuahua	Chihuahua	5000000.0000 <u>4000000.00</u>
3	Chó poodle	Poodle	2000000.0000 <u>1600000.00</u>
4	Chó Alashka	Alaska	15000000.0000 <u>12000000.00</u>
5	Mèo Xiêm	Xiêm	1000000.0000 <u>800000.00</u>
Thú cưng		Sản phẩm	
3	Thức ăn hạt mềm zenith	Đồ ăn	500000.0000 <u>3</u>
4	Thức ăn hạt ANF	Đồ ăn	100000.0000 <u>80000.0000</u>
5	Thức ăn hạt Natural Core	Đồ ăn	400000.0000 <u>300000.0000</u>
6	Thức ăn cho chó Smartheart	Đồ ăn	300000.0000 <u>200000.0000</u>

- Kết quả thông báo khi lưu sửa:

Thú cưng		Sản phẩm	
1	Chó Husky	Husky	10000000.0000 <u>6000000.00</u>
2	Chó Chihuahua	Chihuahua	5000000.0000 <u>40500.0000</u>
3	Chó poodle	Poodle	2000000.0000 <u>1600000.00</u>
4	Chó Alashka	Alaska	15000000.0000 <u>12000000.00</u>
5	Mèo Xiêm	Xiêm	1000000.0000 <u>800000.00</u>
Thú cưng		Sản phẩm	
3	Thức ăn hạt mềm zenith	Đồ ăn	500000.0000 <u>3</u>
4	Thức ăn hạt ANF	Đồ ăn	100000.0000 <u>80000.0000</u>
5	Thức ăn hạt Natural Core	Đồ ăn	400000.0000 <u>300000.0000</u>
6	Thức ăn cho chó Smartheart	Đồ ăn	300000.0000 <u>200000.0000</u>

- Hiển thị Toast thông báo và tông xanh nút save để người dùng nhận diện bảng đã tác động.

3. Đặt hàng và thanh toán

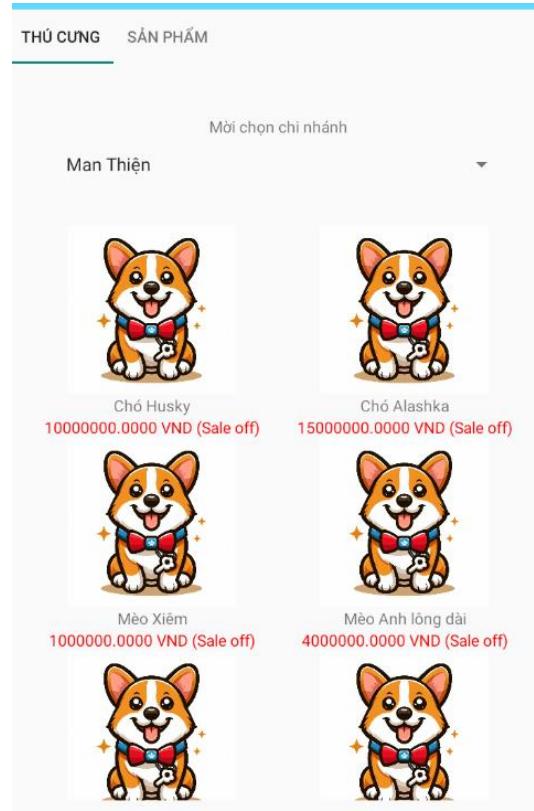
- Hệ thống lấy danh sách các thú cưng và sản phẩm hiển thị lên trang mua hàng của người dùng.
- Xử lý ở Spring Boot trả về danh sách sản phẩm bán:

```
@GetMapping
public ResponseEntity<?> getDanhSachSanPhamBan() {
    List<Map<?, ?>> list = bangGiaSanPhamService.danhSachSanPhamBan();
    List<BangGiaSanPhamDTO> dtoList = new ArrayList<>();
    if (list.isEmpty()) {
        return new ResponseEntity<>("Không có dữ liệu", HttpStatus.BAD_REQUEST);
    }
    for (Map<?, ?> item : list) {
        BangGiaSanPhamDTO bangGiaSanPhamDTO = new BangGiaSanPhamDTO();
        if (item.get("MASANPHAM") != null) {
            bangGiaSanPhamDTO.setMaSanPham((long) item.get("MASANPHAM"));
            bangGiaSanPhamDTO.setTenSanPham((String) item.get("TENSANPHAM"));
            bangGiaSanPhamDTO.setGiaHienTai((BigDecimal) item.get("GIAHIENTAI"));
            bangGiaSanPhamDTO.setSoLuongTon((long) item.get("SOLUONGTON"));
            Sanpham sanpham =
sanPhamService.findById((long)bangGiaSanPhamDTO.getMaSanPham());
            if(hinhanhList!=null&&hinhanhList.size()!=0){
                long idHinhAnh = hinhanhList.get(0).getMahinhanh();
                try{
                    byte[] image =
storageService.downloadImageById(idHinhAnh);
                }
                catch(Exception e){
                    System.out.println(e.getMessage());
                    bangGiaSanPhamDTO.setHinhAnh(null);
                }
            }else{
                bangGiaSanPhamDTO.setHinhAnh(null);
            }
        }
        if (item.get("MALOAISANPHAM") != null) {
            bangGiaSanPhamDTO.setMaLoaiSanPham((int) item.get("MALOAISANPHAM"));
            bangGiaSanPhamDTO.setTenLoaiSanPham((String) item.get("TENLOAISANPHAM"));
        }
        if (item.get("MABANGGIA") != null) {
            bangGiaSanPhamDTO.setMaBangGia((long) item.get("MABANGGIA"));
            bangGiaSanPhamDTO.setThoiGianBatDau((Timestamp)
item.get("THOIGIAbangGiaSanPhamDTO.setThoiGianKetThuc((Timestamp)
item.get("THOIGIAbangGiaSanPhamDTO.setGiaKhuyenMai((BigDecimal) item.get("GIAKM")));
        }
        if (item.get("MACHINHANH") != null) {
            bangGiaSanPhamDTO.setMaChiNhanh((int) item.get("MACHINHANH"));
            bangGiaSanPhamDTO.setTenChiNhanh((String) item.get("TENCHINHANH"));
        }
        dtoList.add(bangGiaSanPhamDTO);
    }
    return new ResponseEntity<>(dtoList, HttpStatus.OK);
}
```

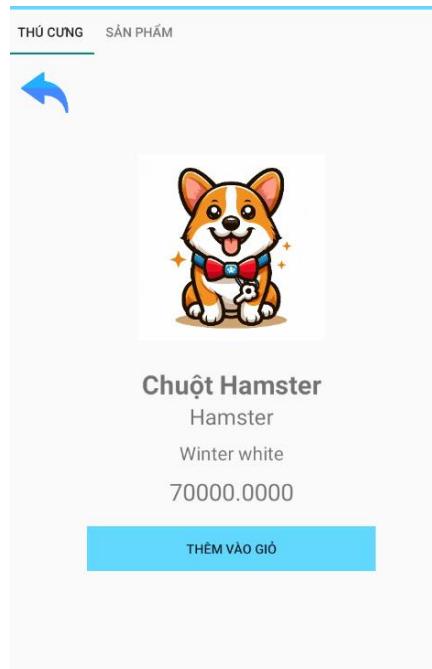
- Gọi api sản phẩm bán dựa trên chi nhánh chọn:

```
private void DocDL(String tenChiNhanh){
    bangGiaService.getAllSP().enqueue(new Callback<List<BangGiaSanPham>>() {
        @Override
        public void onResponse(Call<List<BangGiaSanPham>> call, Response<List<BangGiaSanPham>> response) {
            if (response.code() == 200) {
                data.clear();
                for (BangGiaSanPham x : response.body()) {
                    if (x.getTenChiNhanh() != null && x.getTenChiNhanh().equals(tenChiNhanh)) data.add(x);
                }
                if (muaSanPhamAdapter != null) {
                    muaSanPhamAdapter.notifyDataSetChanged();
                }
            } else {
                try {
                    int code = response.code();
                    String message = response.message();
                    String error = response.errorBody().string();
                    SendMessage.sendMessageFail(mView.getContext(), code, error, message);
                } catch (Exception e) {
                    SendMessage.sendCatch(mView.getContext(), e.getMessage());
                }
            }
        }
        @Override
        public void onFailure(Call<List<BangGiaSanPham>> call, Throwable throwable) {
            SendMessage.sendApiFail(mView.getContext(), throwable);
        }
    });
}
```

- Kết quả giao diện:



- Trang chi tiết:



- Hàm xử lý thêm thú cưng/ sản phẩm vào giỏ hàng:

```
private void themVaoGio(long maThuCung){
    GioHangThuCungGui gioHangThuCungGui = new GioHangThuCungGui();
    gioHangThuCungGui.setMaKhachHang(maKhachHang);
    gioHangThuCungGui.setMaThuCung(maThuCung);
    gioHangService.themThuCung(gioHangThuCungGui).enqueue(new Callback<ResponseBody>() {
        @Override
        public void onResponse(Call<ResponseBody> call, Response<ResponseBody> response) {
            try{
                if(response.code() == 200){
                    String result = response.body().string();
                    Toast.makeText(mView.getContext(),result,Toast.LENGTH_SHORT).show();
                }
                else{
                    try {
                        int code = response.code();
                        String message = response.message();
                        String error = response.errorBody().string();

                        SendMessage.sendMessageFacade((Exception e),code,error,message);
                        SendMessage.sendCatch(mView.getContext(),e.getMessage());
                    }
                    catch (Exception e){
                        SendMessage.sendCatch(mView.getContext(),e.getMessage());
                    }
                }
            }
            catch (Exception e){
                SendMessage.sendCatch(mView.getContext(),e.getMessage());
            }
        }
        @Override
        public void onFailure(Call<ResponseBody> call, Throwable throwable) {
            SendMessage.sendApiFail(mView.getContext(),throwable);
        }
    });
}
```

- Giao diện giỏ hàng của người dùng:

Man Thiện				ĐẶT HÀNG	
Thú cưng					
Mã	Tên	Tên giống	Số lượng		
1	Chó Husky	Husky	-	1	+
5	Mèo Xiêm	Xiêm	-	1	+
7	Rồng nam mỹ	Rồng nam mỹ	-	1	+
9	Chuột Hamster	Hamster	-	1	+
Sản phẩm					
Mã	Tên	Tên loại	Số lượng		
3	Thức ăn hạt mềm zenith	Đồ ăn	-	1	+
4	Thức ăn hạt ANF	Đồ ăn	-	1	+

- Dòng lệnh bắt sự kiện thay đổi số lượng muôn mua của khách hàng:

```
int soLuong = donDatThuCungGui.getSoLuong();
int soLuongMax = thuCung.getSoLuongTon();
soLuong += 1;
if(soLuong>Integer.MAX_VALUE) return;
if(soLuong>soLuongMax){
    Toast.makeText(mView.getContext(), "Vượt quá số lượng
tồn",Toast.LENGTH_SHORT).show();
    return;
}
donDatThuCungGui.setSoLuong(soLuong);
notifyDataSetChanged();
```

- Quá trình xử lý đặt hàng sẽ bắt đầu từ việc tạo đơn hàng theo thông tin mà khách hàng cung cấp như số điện thoại, địa chỉ giao hàng, chi nhánh mua và mã khách hàng (là tên đăng nhập). Sau khi tạo thành công, hệ thống hiển thị nút thanh toán bằng Paypal, người dùng có thể thanh toán trực tiếp online hoặc thanh toán bằng tiền mặt khi nhận hàng (trường hợp này người dùng chỉ cần thoát).

- Cấu hình với API của Paypal (hệ thống thanh toán bên thứ 3):

+ Cài đặt thư viện:

```
implementation("com.paypal.checkout:android-sdk:1.1.0")
implementation("commons-codec:commons-codec:1.14")
```

+ Setting maven:

```
dependencyResolutionManagement {
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
        maven{
            url =uri("https://cardinalcommerceprod.jfrog.io/artifactory/android")
            credentials{
                username = "paypal_sgerritz"
                password =
"AKCp8jQ8tAahqpT5JjZ4FRP2mW7GMoFZ674kGqHmupTesKeAY2G8NcmPKLuTxTGkKjDLRzDUQ"
            }
        }
        maven("https://jitpack.io")
    }
}
```

+ Đường dẫn đến app phụ thuộc trong Manifest:

```
android:name="com.example.petshopapp.paypal.PaypalApp"
```

+ Tạo lớp đúng như đường dẫn chỉ định trên:

```
public class PaypalApp extends Application {
    @Override
    public void onCreate() {
        super.onCreate();
        PayPalCheckout.setConfig(new CheckoutConfig(
            this,
            "AT9all-Y7gGi5pPmCAbHYsFZap-89ms_ygCfZgTvTcM59B9p5j9t1vpqUgPdUUleCt16Cx0ga0lpZSW3",
            Environment.SANDBOX,
            CurrencyCode.USD,
            UserAction.PAY_NOW,
            "com.example.petshopapp://paypalpay"

        ));
    }
}
```

Trong đó cung cấp ID Client của tài khoản thụ hưởng tiền chuyển tới.

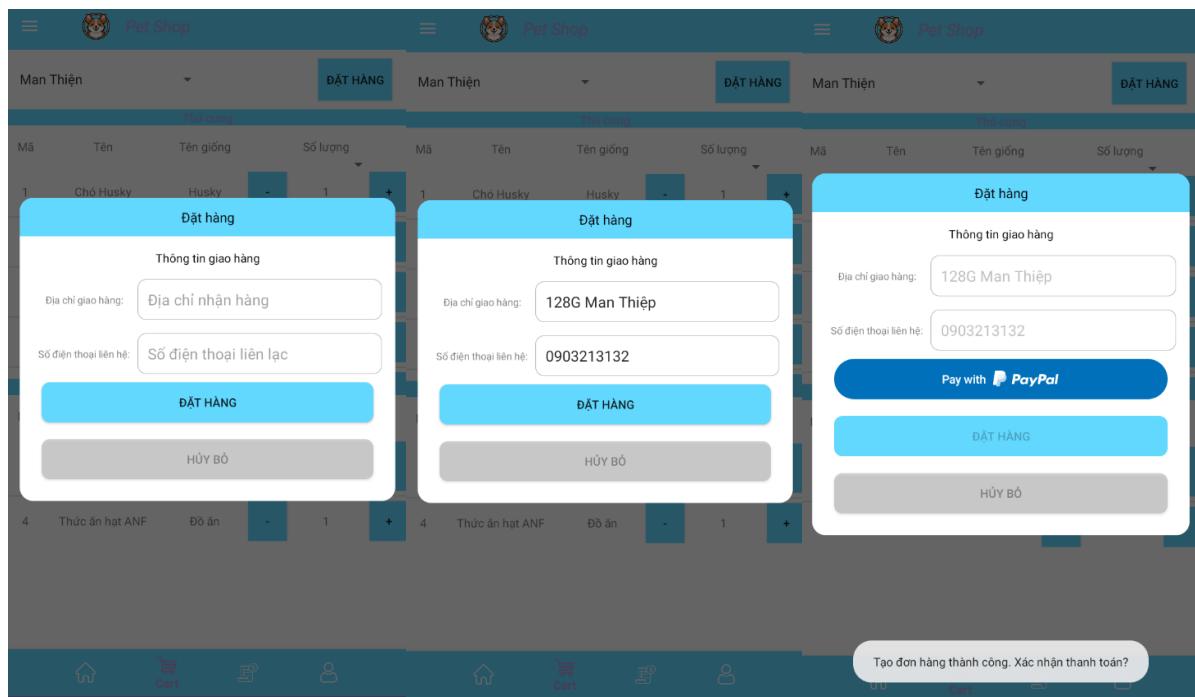
+ Để có thẻ thanh toán, giao diện cần cung cấp một đối tượng mà Paypal cung cấp:

```
<com.paypal.checkout.paymentbutton.PaymentButtonContainer  
    android:id="@+id/payment_button_container"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_margin="10dp"  
    android:gravity="center"  
    app:paypal_button_color="blue"  
    app:paypal_button_enabled="true"  
    app:paypal_button_label="pay"  
    app:paypal_button_shape="pill"  
    app:paypal_button_size="small" />
```

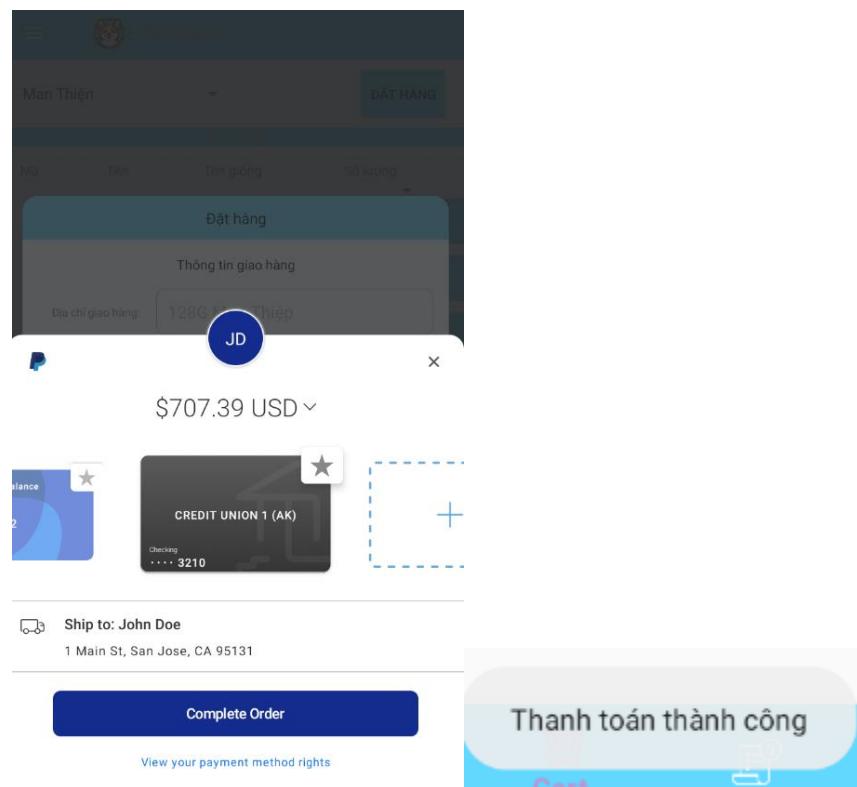
- Để xử lý thanh toán, hệ thống cần gọi api để tính thành tiền của tổng đơn đặt hàng của khách hàng, sau khi có thành tiền, cần chuyển đổi giá trị tiền tệ sang USD và gọi requestPaypal để chuyển sang các hoạt động thanh toán tiếp theo của Paypal.

```
private void requestPaypal(String gia, long maDonDat){  
    paymentButtonContainer.setup(  
        new CreateOrder() {  
            @Override  
            public void create(@NotNull CreateOrderActions createOrderActions) {  
                ArrayList<PurchaseUnit> purchaseUnits = new ArrayList<>();  
                purchaseUnits.add(  
                    new PurchaseUnit.Builder()  
                        .amount(  
                            new Amount.Builder()  
                                .currencyCode(CurrencyCode.USD)  
                                .value(gia)  
                                .build()  
                        )  
                        .build()  
                );  
                OrderRequest order = new OrderRequest(  
                    OrderIntent.CAPTURE,  
                    new AppContext.Builder()  
                        .userAction(UserAction.PAY_NOW)  
                        .build(),  
                    purchaseUnits  
                );  
                createOrderActions.create(order, (CreateOrderActions.OnOrderCreated) null);  
            }  
        },  
        new OnApprove() {  
            @Override  
            public void onApprove(@NotNull Approval approval) {  
                approval.getOrderActions().capture(new OnCaptureComplete() {  
                    @Override  
                    public void onCaptureComplete(@NotNull CaptureOrderResult result) {  
                        taoHoaDon(maDonDat); //Tạo hóa đơn  
                        Log.i("CaptureOrder", String.format("CaptureOrderResult: %s", result));  
                        if(dialog!=null)dialog.dismiss();  
                    };  
                });  
            }  
        },  
    );  
}
```

- Kết quả giao diện:



- Giao diện thanh toán với tài khoản thanh toán trên môi trường ảo Sandbox:



Chương VII: Tổng kết

- Trong suốt quá trình thực hiện đồ án “xây dựng hệ thống bán thú cưng”, nhóm đã tập trung vào việc phát triển một hệ thống toàn diện dựa trên kiến trúc microservice, đáp ứng các nhu cầu cơ bản của việc quản lý một cửa hàng bán thú cưng hiện đại. Hệ thống cung cấp các chức năng không chỉ cho người quản lý mà còn là người quản trị, nhân viên hệ thống và khách hàng, thể hiện tính đa năng và hữu dụng trong việc quản trị thông tin hệ thống một cách toàn vẹn.

- Kết quả đạt được:

1. Phát triển hệ thống: Nhóm đã xây dựng thành công ứng dụng thân thiện với người dùng, với giao diện trực quan và dễ sử dụng, các chức năng hoạt động ổn định, không có hiện tượng mất mát dữ liệu hoặc lỗi không hoạt động.
2. Tích hợp công nghệ hiện đại: Việc sử dụng công nghệ như Spring boot cho backend hữu hiệu về khả năng truy vấn, cho thấy đạt hiệu suất tốt, dễ dàng mở rộng và bảo trì. Nhờ vào áp dụng theo microservice, tạo api, mã code còn có thể tái sử dụng ở nhiều nơi, mở rộng lĩnh vực hệ thống.
3. Xây dựng ứng dụng di động: Xây dựng được phần mềm ở nền tảng di động như Android nhằm tiếp cận được nhiều người dùng hơn.
4. Xây dựng website: Xây dựng được website thân thiện với người dùng.

- Những hạn chế và hướng phát triển:

1. Cải thiện hiệu suất truy vấn: Tối ưu hóa hơn trong việc truy vấn cơ sở dữ liệu và gọi api, tránh khiến cho tốc độ phản hồi dữ liệu chậm.
 2. Mở rộng tính năng: Trong tương lai, đề tài cần phát triển thêm nhiều chức năng hơn để phù hợp với môi trường và bối cảnh thực tế, ngoài ra, còn có những điểm có thể khai thác như trang trò chuyện giữa chủ với khách, mô hình học máy dự đoán thú cưng, bộ đề xuất sản phẩm thông minh ... nhằm để tăng tính sử dụng và giá trị của sản phẩm.
- Trong tương lai, việc tiếp thu phát triển và nâng cấp hệ thống sẽ giúp phần mềm ngày càng hoàn thiện, phục vụ tốt hơn cho cả người bán và mua thú cưng, từ đó mang lại giá trị lợi ích thực tế hơn với cộng đồng và trở thành công cụ đắc lực cho kinh doanh bán thú cưng.

Tài liệu kham khảo

- [1] Admin, "Java Android Là Gì? Để Lập Trình Android Bằng Java Bạn Cần Biết Nhũng Gì?", CODEGYM, [Online]. Available: <https://codegym.vn/blog/java-android-la-gi-de-lap-trinh-android-bang-java-ban-can-biet-nhung-gi/>. [Accessed 3 6 2024].
- [2] N. Liên, "Spring boot là gì? Cách sử dụng Spring boot làm dự án như thế nào nhanh nhất?," fptshop.vn, 16 1 2024. [Online]. Available: <https://fptshop.com.vn/tin-tuc/danh-gia/spring-boot-la-gi-172291>.
- [3] Pum, "SQL Server là gì? Cách tải & cài đặt Microsoft SQL Server," 200LabBlog, 16 9 2023. [Online]. Available: <https://200lab.io/blog/sql-server-la-gi/>. [Accessed 3 6 2024].
- [4] D. V. Phu, "Tìm hiểu cơ bản về Docker," VIBLO, 20 1 2020. [Online]. [Accessed 13 6 2024].
- [5] N. T. T. Yen, "[Android Library]: Tìm hiểu Retrofit2," VIBLO, 18 10 2018. [Online]. Available: <https://viblo.asia/p/android-library-tim-hieu-retrofit2-bJzKmXGB59N>. [Accessed 3 6 2024].