

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**



NGUYỄN BÁ ĐẠT

HỆ THỐNG API HỖ TRỢ BÁN HÀNG ONLINE

Mã số sinh viên: 1751010022

**ĐỒ ÁN NGÀNH
NGÀNH KHOA HỌC MÁY TÍNH**

Giảng viên hướng dẫn: DƯƠNG HỮU THÀNH

TP.HỒ CHÍ MINH, 2020

LỜI CẢM ƠN

Em xin chân thành cảm ơn trường Đại học Mở TP.HCM cùng khoa công nghệ thông tin đã tạo điều kiện cho em được học tập, rèn luyện, phát triển bản thân giúp em có được kiến thức, các kỹ năng cần thiết và nền tảng để phát triển sự nghiệp của em sau trong con đường tiếp theo.

Em cũng xin chân thành cảm ơn các quý thầy cô, đội ngũ giảng viên và công nhân viên của trường đã hỗ trợ, giúp đỡ, dạy dỗ em trong suốt quá trình học tập tại trường.

Em xin gửi lời cảm ơn chân thành đến Thầy Ths. Dương Hữu Thành đã giúp đỡ, hướng dẫn em tận tình trong quá trình nghiên cứu thực hiện đồ án và bài báo cáo này.

Với kinh nghiệm và trình độ còn hạn chế nên trong quá trình thực hiện đồ án vẫn còn nhiều thiếu sót, rất mong nhận được sự góp ý, chỉ dẫn của quý thầy cô.

Em xin kính chúc quý thầy cô thật nhiều sức khỏe và đạt được nhiều thành công, em xin chân thành cảm ơn.

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

MỤC LỤC

Chương 1. TỔNG QUAN.....	6
Chương 2. CƠ SỞ LÝ THUYẾT	7
2.1. Framework Django	7
2.2. MySQL Database.....	7
2.3. EAV Pattern.....	8
2.4. Code-first	10
2.5. API.....	11
2.6. RESTful API.....	11
2.7. OAuth 2.....	11
2.8. OTP and Mail.....	11
Chương 3. NỘI DUNG THỰC HIỆN VÀ KẾT QUẢ.....	12
3.1. Xác định yêu cầu hệ thống.....	12
3.2. Thiết kế cơ sở dữ liệu.....	14
3.3. Xây dựng trang quản trị (Admin)	18
3.3.1. Giai đoạn 1: Tạo cơ sở dữ liệu	18
3.3.2. Giai đoạn 2: Tạo trang Admin	19
3.4. Xây dựng các API theo yêu cầu hệ thống.....	25
3.4.1. API Product	31
3.4.2. API Customer	38
3.4.3. API Order	55
Chương 4. KẾT LUẬN	73
TÀI LIỆU THAM KHẢO	74

DANH MỤC HÌNH

Hình 1. Mô hình dữ liệu EAV	8
Hình 2. Sơ đồ VD mô hình EAV.....	9
Hình 3. Mô hình Code-first	11
Hình 4. Sơ đồ mô tả API	11
Hình 5. Sơ đồ usecase.....	12
Hình 6. Phân quyền nhân viên.....	12
Hình 7. Sơ đồ class	14
Hình 8. Cơ sở dữ liệu Product	15
Hình 9. Cơ sở dữ liệu Order	16
Hình 10. Các trạng thái của đơn hàng	16
Hình 11. Cơ sở dữ liệu Customer	17
Hình 12. Cấu trúc dự án.....	18
Hình 13. Danh sách API Product.....	25
Hình 14. Danh sách API Customer	26
Hình 15. Danh sách API Order.....	27
Hình 16. Ví dụ mẫu gọi API.....	30
Hình 17. API Get Category GET: Ok	31
Hình 18. API Get Manufacturer GET: Ok	32
Hình 19. API Get Discount GET: Ok.....	33
Hình 20. API Get Product GET: Ok.....	35
Hình 21. API Get Product Attribute: Ok	36
Hình 22. API Get Product Attribute Value: Ok	37
Hình 23. API Sign Up: Ok	38
Hình 24. API Sign Up: Tên tài khoản đã tồn tại	38
Hình 25. API Sign Up: Sai thông tin ngày sinh	39
Hình 26. API Sign Up: Sai thông tin SĐT	39
Hình 27. API Sign In: Ok	40
Hình 28. API Sign In: Sai thông tin đăng nhập.....	41
Hình 29. API Sign In: Thiếu thông tin đăng nhập.....	41
Hình 30. Thêm token xác thực vào headers của API	42
Hình 31. Xác thực token, token không tồn tại.....	42
Hình 32. Xác thực token, token hết hạn	42
Hình 33. API Sign Out: Ok	44
Hình 34. API Sign Out: Token quá hạn.....	44
Hình 35. API Get Customer Info: Ok.....	45
Hình 36. API Add Tel Number: Ok	45
Hình 37. API Edit Tel Number: Ok.....	46
Hình 38. API Edit Tel Number: Sai mã SĐT	46
Hình 39. API Delete Tel Number: Ok.....	46
Hình 40. API Add Ship Address: Ok	47
Hình 41. API Delete Ship Address: Ok.....	47
Hình 42. API Edit Ship Address: Ok.....	48
Hình 43. API Edit Ship Address: Sai mã địa chỉ.....	48
Hình 44. API Edit Customer.....	49

Hình 45. API Edit Customer: Sai mật khẩu.....	49
Hình 46. API Edit Customer: Sai dữ liệu ngày sinh.....	50
Hình 47. API Edit Customer: Gửi thiếu thông tin.....	50
Hình 48. API Change Password: Ok	51
Hình 49. API Change Password: Mật khẩu không khớp.....	51
Hình 50. API Change Password: Sai mật khẩu cũ	51
Hình 51. API Request Restore Password: Ok	52
Hình 52. Mail mã xác thực khôi phục mật khẩu	52
Hình 53. API Request Restore Password: Email không tồn tại.....	52
Hình 54. API Change Password OTP: Ok.....	54
Hình 55. API Get Cart: Ok	55
Hình 56. API Add Item to Cart: Ok.....	55
Hình 57. API Add Item to Cart: Sai kiểu dữ liệu	56
Hình 58. API Add Item to Cart: Mã sản phẩm không tồn tại.....	56
Hình 59. API Add Item to Cart: Số lượng không hợp lệ.....	56
Hình 60. API Add Item to Cart: Số lượng tăng tự động.....	57
Hình 61. API Del Item form Cart: Ok	57
Hình 62. API Del Item form Cart: Sản phẩm không có trong giỏ.....	57
Hình 63. API Customer Add Order: Ok	58
Hình 64. Mail thông báo KH đặt hàng thành công.....	58
Hình 65. API Get Customer Order (GET): Ok.....	62
Hình 66. API Get Customer Order (POST): DS đơn hàng có trạng thái PROCESSING	62
Hình 67. API Get Customer Order Detail: Ok	63
Hình 68. API Get Customer Order Detail: Mã đơn hàng không tồn tại.....	63
Hình 69. API Get Customer Order Detail: Mã đơn hàng không phải số	64
Hình 70. API Customer Cancel Order	64
Hình 71. Mail thông báo KH hủy đơn thành công	64
Hình 72. API Customer Return Order: Ok	66
Hình 73. Mail thông báo KH trả hàng thành công	66
Hình 74. API Admin Ship Order: Ok	69
Hình 75. Mail thông báo KH đơn hàng đang được vận chuyển đi.....	69
Hình 76. API Admin Done Order: Ok.....	71
Hình 77. Mail thông báo KH đơn hàng đã hoàn tất.....	71

Chương 1. TỔNG QUAN

Với tốc độ phát triển của ngành công nghệ thông tin nói chung và thương mại điện tử nói riêng thì nhu cầu của người tiêu dùng ngày càng cao, việc mua sắm không chỉ đơn thuần là tại các địa điểm bán hàng nữa mà còn trên các trang web mua sắm, ứng dụng mua hàng online. Việc phát triển một hệ thống web bán hàng cũng vô cùng phức tạp, mất nhiều thời gian, chi phí và không chỉ trên web mà còn các ứng dụng di động khác nữa. Vậy nếu mỗi lần muốn triển khai bán hàng trên một ứng dụng hay phương tiện khác, ta lại phải thiết kế và xây dựng lại hệ thống trên phương tiện đó? Với công nghệ hiện nay, chúng ta đã có giải pháp để giải quyết cho vấn đề đó là chỉ cần xây dựng một hệ thống API cho cửa hàng và khi triển khai bán hàng trên các phương tiện khác, ta chỉ cần gọi lại các API để sử dụng trên các phương tiện đó, như vậy ta có thể tiết kiệm được phần lớn thời gian và chi phí để phát triển và mở rộng hệ thống bán hàng của mình.

Về đề tài đồ án của em cũng chính là giải pháp cho vấn đề trên. Trong đồ án này, em sẽ thiết kế một hệ thống API hỗ trợ cho việc bán hàng online. Mặt hàng đầu tiên em hướng đến để thiết kế cơ sở dữ liệu cho hệ thống là các đồ dùng điện tử. Hệ thống sẽ hỗ trợ cho khách hàng việc xem chọn lựa hàng hóa cần mua và bỏ vào giỏ hàng của mình sau đó tiến hành đặt hàng khi đã tạo tài khoản trong hệ thống. Khách hàng cũng có thể quản lý các thông tin tài khoản và theo dõi các đơn hàng của mình. Ngoài ra em cũng sẽ xây dựng một web quản trị cùng các chức năng cơ bản để giúp chủ cửa hàng và nhân viên có thể quản lý các thông tin hàng hóa và đơn hàng của cửa hàng. Hướng phát triển của đề tài là xây dựng một hệ thống, một cơ sở dữ liệu có thể mở rộng được để bán tất cả mặt hàng có thể bán được và tích hợp được thêm các dịch vụ hỗ trợ thanh toán và giao hàng từ bên thứ ba. Để có thể phát triển hệ thống nhanh chóng em sử dụng một framework thuộc ngôn ngữ Python là Django cùng với hệ quản trị cơ sở dữ liệu MySQL cho đề tài của mình.

Về nội dung của các chương còn lại bao gồm: Chương 2 em sẽ giới thiệu về các cơ sở lý thuyết của những công nghệ em sử dụng để phát triển đề tài. Chương 3 là gồm các nội dung, giai đoạn, quá trình em thực hiện đồ án này và kết quả đạt được của từng giai đoạn đó. Chương 4 em sẽ tổng kết lại đề tài cũng như kết quả và những thứ em đã học, rút ra được sau quá trình làm đồ án này của mình.

Chương 2. CƠ SỞ LÝ THUYẾT

2.1. Framework Django

Django là một framework web bậc cao của ngôn ngữ Python, giúp việc phát triển xây dựng ứng dụng web nhanh chóng, gọn gàng, và có tính ứng dụng thực tế. Django được xây dựng bởi các nhà phát triển dày dặn kinh nghiệm, nó giúp xử lý được phần lớn sự phức tạp của việc phát triển web, vì vậy chúng ta có thể tập trung vào việc phát triển ứng dụng web của mình mà không phải xây dựng lại nền tảng đã có của Django. Dưới đây bao gồm các ưu điểm và cũng chính là lý do em lựa chọn Django:

- *Phát triển nhanh chóng:* Django được thiết kế để giúp các nhà phát triển đưa ứng dụng web của mình từ ý tưởng đến hoàn thiện nhanh nhất có thể.
Hỗ trợ đầy đủ tác vụ: Django được bao gồm rất nhiều tính năng mà bạn có thể sử dụng để xử lý các tác vụ phát triển web thông thường chẳng hạn như xác thực người dùng (User Authentication), Administration, Site map và nhiều tác vụ khác.
- *Đảm bảo tính bảo mật:* Django rất coi trọng việc bảo mật và giúp các nhà phát triển tránh được nhiều lỗi bảo mật phổ biến, chẳng hạn như SQL Injection, giả mạo yêu cầu cross-site. Hệ thống User Authentication của Django cung cấp cách bảo mật tốt nhất để quản lý tài khoản người dùng và mật khẩu của họ.
- *Khả năng mở rộng vượt trội:* Một số trang web doanh nghiệp lớn trên thế giới sử dụng Django để đáp ứng nhu cầu lưu lượng truy cập lớn vì khả năng mở rộng quy mô nhanh chóng và linh hoạt của nó.
- *Tính linh hoạt cao:* Các công ty, tổ chức và kể cả chính phủ đã sử dụng Django để xây dựng mọi thứ từ hệ thống quản lý nội dung đến mạng xã hội và cả nền tảng khoa học máy tính.

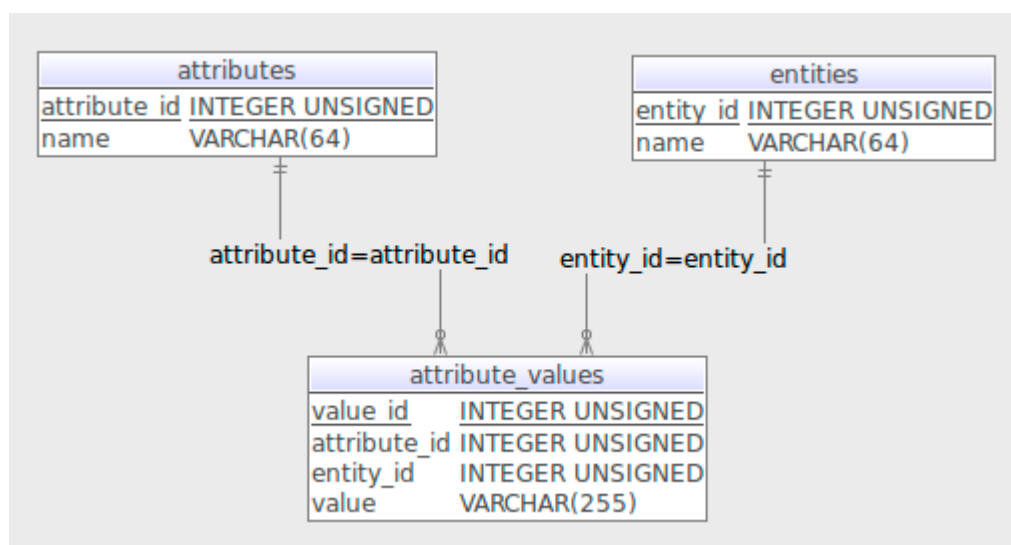
2.2. MySQL Database

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở. MySQL là một ngôn ngữ truy vấn có cấu trúc, là cơ sở dữ liệu quan hệ tổ chức dữ liệu vào một hay nhiều bảng dữ liệu mà các kiểu dữ liệu có thể sẽ liên kết với nhau và các liên kết này giúp tạo cấu trúc cho dữ liệu. SQL là một ngôn ngữ mà lập trình viên sử dụng để tạo, chỉnh sửa, trích xuất dữ liệu từ dữ liệu quan hệ, cũng như quản lý việc truy cập vào cơ sở dữ liệu.

2.3. EAV Pattern

Entity-Attribute-Value Pattern (dịch: mô hình thực thể - thuộc tính – giá trị) (EAV Pattern) là một mô hình dữ liệu có thể dễ dàng mở rộng số lượng thuộc tính của các thực thể.

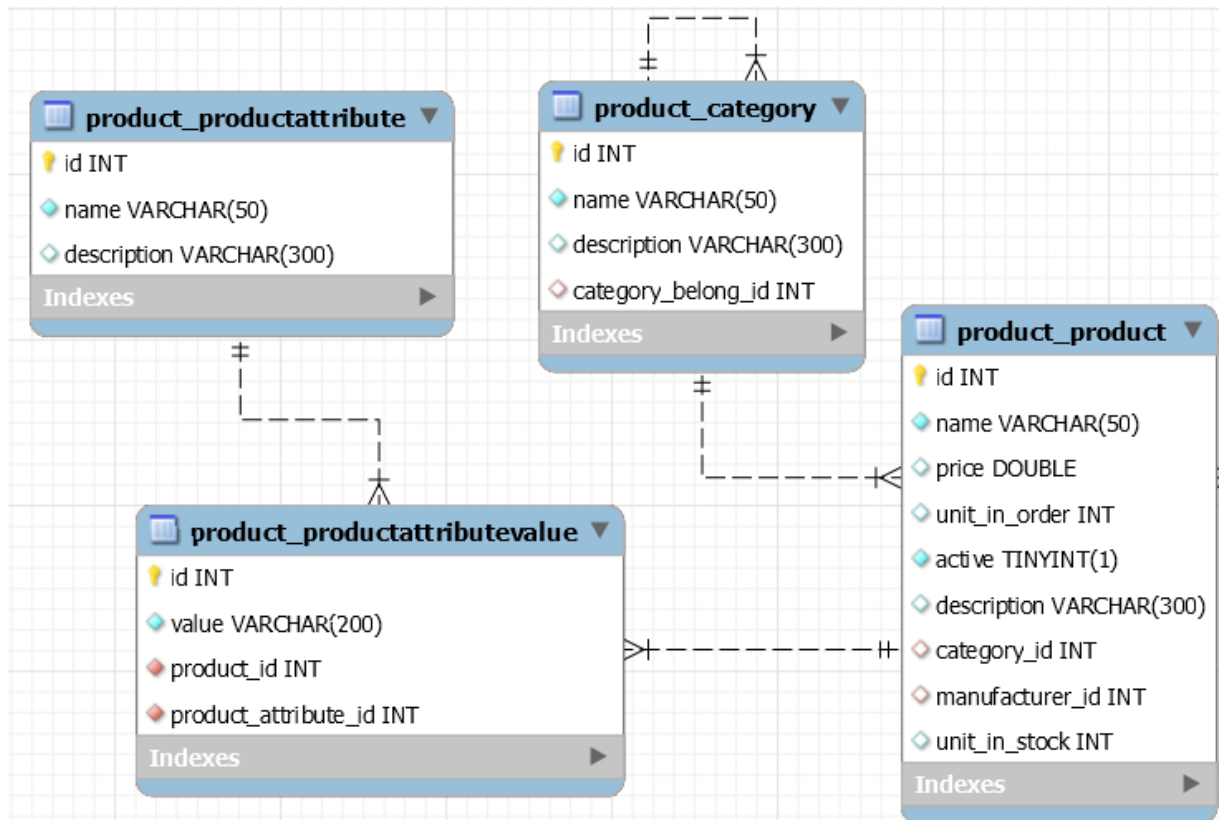
EAV là một cấu trúc thiết kế cơ sở dữ liệu trong Magento.



Hình 1. Mô hình dữ liệu EAV

- Entity: Bảng chứa thông tin cơ bản của thực thể.
- Attribute: Bảng chứa các thuộc tính ta có thể thêm mới vào thực thể về sau.
- Value: Bảng chứa giá trị với 2 khóa ngoại là thuộc tính là thực thể.

VD: Với cấu trúc bảng dữ liệu sau



Hình 2. Sơ đồ VD mô hình EAV

Ta có:

- Entity là bảng product_product.
- Attribute là bảng product_productattribute.
- Value là bảng product_productattributevalue.

Với cách thiết kế trên, hệ thống có thể dễ dàng mở rộng ra sau này, bán nhiều loại sản phẩm hơn (điện thoại, đồ gia dụng, quần áo,...). Giả sử công ty ban đầu chỉ bán điện thoại với 1 category là điện thoại với các product là các điện thoại khác nhau. Sau này mở rộng, công ty cần mở rộng hệ thống để bán thêm nhiều mặt hàng điện tử khác, category điện thoại khi này sẽ có category_belong là category đồ điện tử và trong category đồ điện tử sẽ bao gồm nhiều các category khác như laptop, tivi, tủ lạnh,... . Và các sản phẩm khi này không chỉ còn là điện thoại mà gồm nhiều mặt hàng khác với nhiều thuộc tính khác nhau thì thiết kế trên chỉ cần thêm 1 attribute mới vào và tạo value của attribute cho product đó.

Product:

id	name	price	unit_in_stock	unit_in_order	active
1	Asus ROG Zephyrus M GU502GU AZ090T	32990000	20	0	1

ProductAttribute:

id	name
1	Thông số màn hình
2	CPU
3	Ram
4	Ổ cứng
5	Pin
6	Ảnh

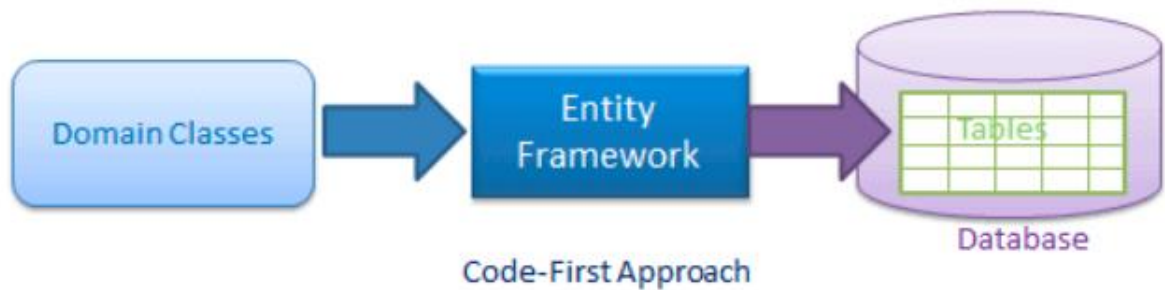
ProductAttributeValue:

id	value	product_id	product_attribute_id
1	15.6" FHD (1920 x 1080) IPS, 100% sRGB, 240...	1	1
2	Intel Core i7-9750H 2.6GHz up to 4.5GHz 12MB	1	2
3	16GB DDR4 2666MHz Onboard (1x SO-DIMM so...	1	3
4	512GB SSD PCIE G3X4 (Support RAID 0) (2 slots)	1	4
5	4 Cell 76WHr	1	5
6	https://product.hstatic.net/1000026716/produ...	1	6

Vì những đặc điểm trên nên em chọn mô hình EAV để thiết kế dữ liệu sản phẩm cho hệ thống của mình để có thể dễ dàng mở rộng, thêm mới các thuộc tính cho sản phẩm về sau.

2.4. Code-first

Code-first là một cách tạo dữ liệu bằng việc tập trung vào code, tạo các class của các thực thể cho hệ thống trước rồi từ đó entity framework sẽ tạo ra cơ sở dữ liệu từ code đó.

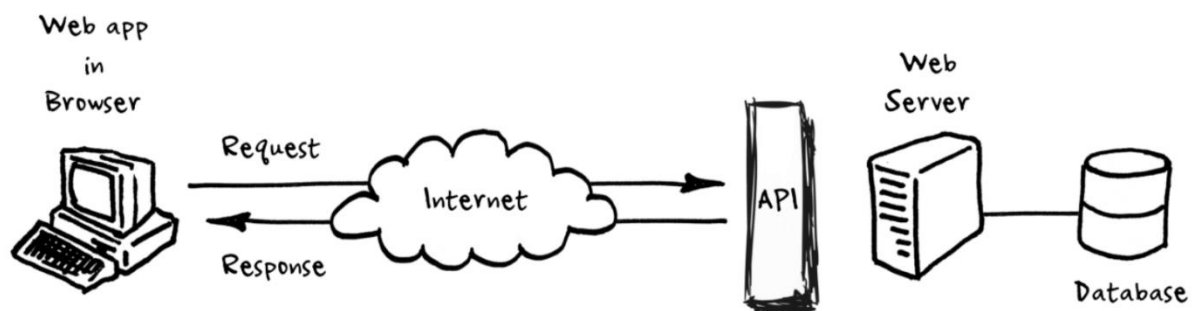


Hình 3. Mô hình Code-first

Với cách tiếp cận đó, ta có thể dễ dàng thay đổi thiết kế cơ sở dữ liệu và phát triển hệ thống một cách nhanh chóng.

2.5. API

Application Programming Interface (API) là một hàm dùng để truyền dữ liệu (giao tiếp) giữa một phần mềm này với phần mềm khác và trong đó, nó cũng chứa các điều khoản nhất định cho việc trao đổi đó.



Hình 4. Sơ đồ mô tả API

API không phải là cơ sở dữ liệu hay máy chủ, nó là một hàm hoặc đoạn mã dùng để quản lý các điểm truy cập cho máy chủ.

2.6. RESTful API

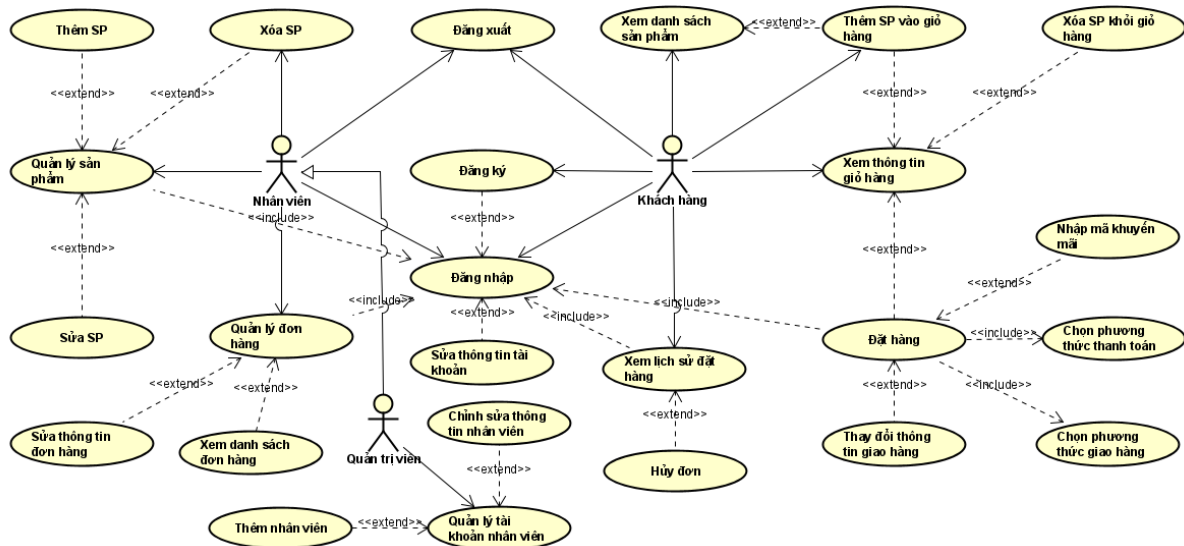
2.7. OAuth 2

2.8. OTP and Mail

Chương 3. NỘI DUNG THỰC HIỆN VÀ KẾT QUẢ

3.1. Xác định yêu cầu hệ thống

Hệ thống website bán hàng online gồm 2 đối tượng sử dụng chính là khách hàng và nhân viên. Từ đó xây dựng các chức năng chính và cơ bản để phục vụ cho hệ thống. Dưới đây là sơ đồ usecase của hệ thống:



Hình 5. Sơ đồ usecase

Theo sơ đồ trên, đối tượng quản lý của hệ thống gồm có quản trị viên và nhân viên. Quản trị viên là người có toàn quyền để quản lý các chức năng điều hành, quản lý. Nhân viên được chia ra nhiều loại nhân viên để quản lý từng thành phần của hệ thống như sau:



Hình 6. Phân quyền nhân viên

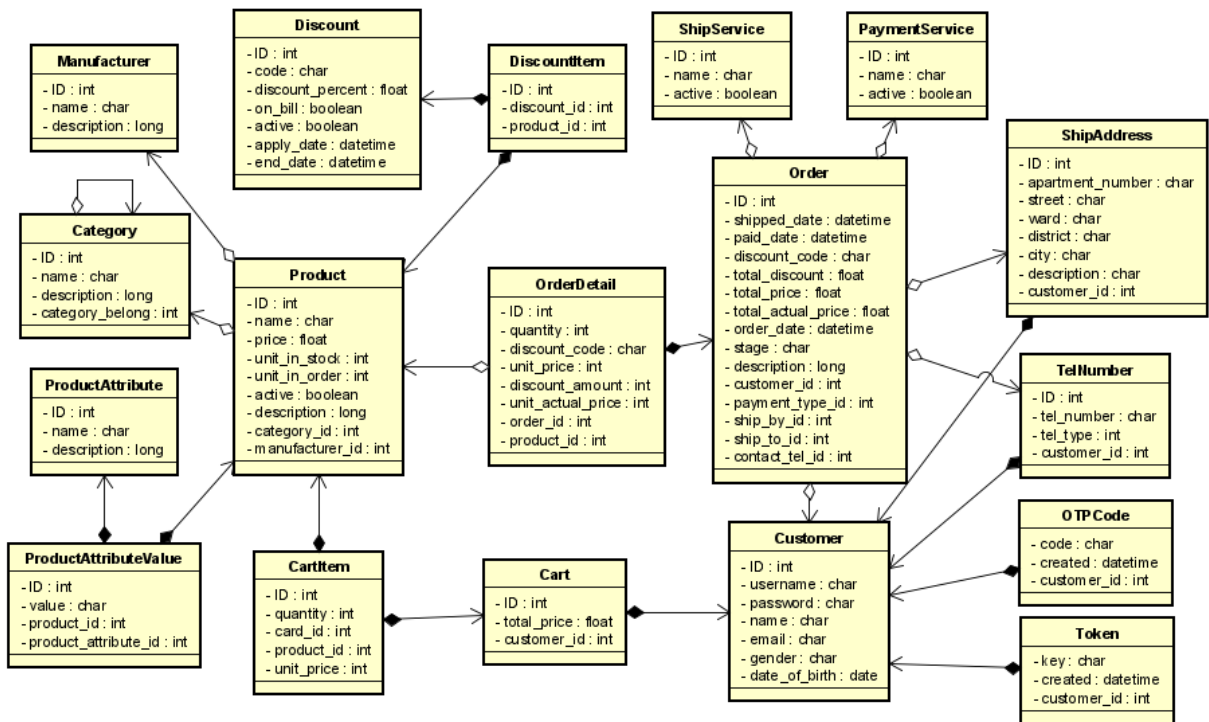
- Nhân viên dịch vụ: Quản lý các dịch vụ bên thứ 3 cung cấp cho hệ thống như các dịch vụ thanh toán, các dịch vụ giao hàng.
- Nhân viên kho: Quản lý các sản phẩm, nhà cung cấp, thương hiệu, loại sản phẩm, số lượng sản phẩm tồn kho và đang được đặt hàng.
- Nhân viên nhân sự: Quản lý nhân viên và phân quyền cho các nhân viên.
- Nhân viên thống kê đơn: Quản lý các đơn hàng, theo dõi đơn hàng, cập nhật trạng thái của các đơn hàng.

Khách hàng là đối tượng sử dụng hệ thống gồm các thao tác chính sau:

- Đăng ký, đăng nhập, đăng xuất, chỉnh sửa thông tin tài khoản.
- Xem danh sách các sản phẩm.
- Thêm sản phẩm vào giỏ hàng hoặc xóa sản phẩm khỏi giỏ hàng.
- Tiến hành đặt hàng tạo đơn hàng, trong lúc tạo đơn hàng phải thực hiện các công việc bao gồm chọn phương thức thanh toán và cung cấp thông tin giao hàng. Có thể nhập mã khuyến mãi nếu có.
- Xem lịch sử các đơn hàng đã đặt, theo dõi tình trạng các đơn hàng và có thể hủy đơn đối với các đơn hàng chưa thanh toán và còn trong tình trạng xử lý (PROCESSING).

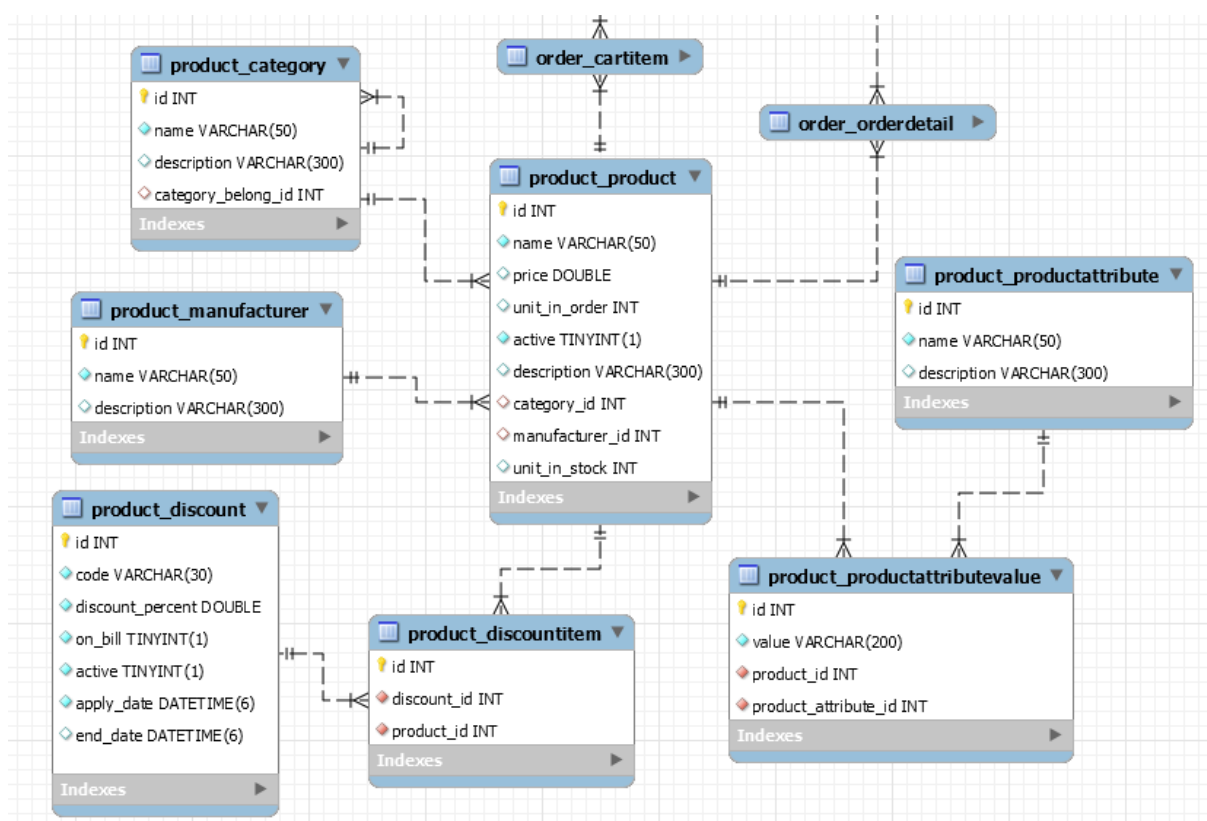
3.2. Thiết kế cơ sở dữ liệu

Từ các yêu cầu đã được xác định ở phần trên, em thiết kế được sơ đồ class như sau:



Hình 7. Sơ đồ class

Từ sơ đồ class trên em phân dữ liệu thành 3 nhóm chính gồm: Product, Order, Customer.



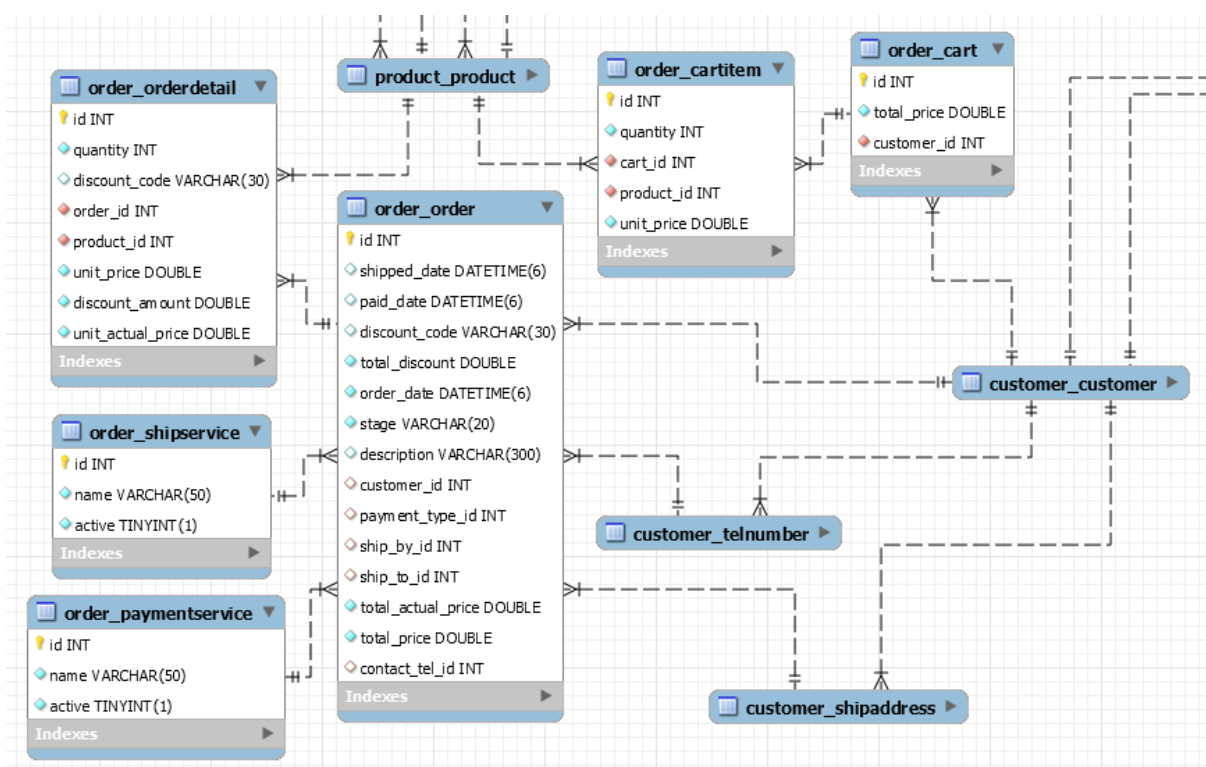
Hình 8. Cơ sở dữ liệu Product

Bảng Category và Manufacturer được dùng để phân loại cho sản phẩm, và bảng Category cũng được dùng để phân loại và gom nhóm chính nó (VD: danh mục linh kiện gồm nhiều danh mục khác như ram, ổ cứng,...).

Discount được dùng để xác định khuyến mãi được áp dụng trên các sản phẩm nào hoặc không áp dụng cho sản phẩm mà áp dụng cho tổng bill bằng thuộc tính on_bill (Nếu áp dụng on_bill sẽ không áp dụng cho sản phẩm và ngược lại).

Bảng ProductAttribute dùng để định nghĩa các thuộc tính của 1 sản phẩm. ProductAttributeValue dùng để xác định giá trị của thuộc tính và sản phẩm. Cách thiết kế này được dựa theo mô hình Entity-Attribute-Value (EAV) [1] giúp cho dữ liệu sản phẩm có thể dễ dàng thêm mới các thuộc tính về sau.

Bảng Product được dùng làm khóa ngoại cho các bảng OrderDetail và CartItem của dữ liệu Order trong phần tiếp theo.

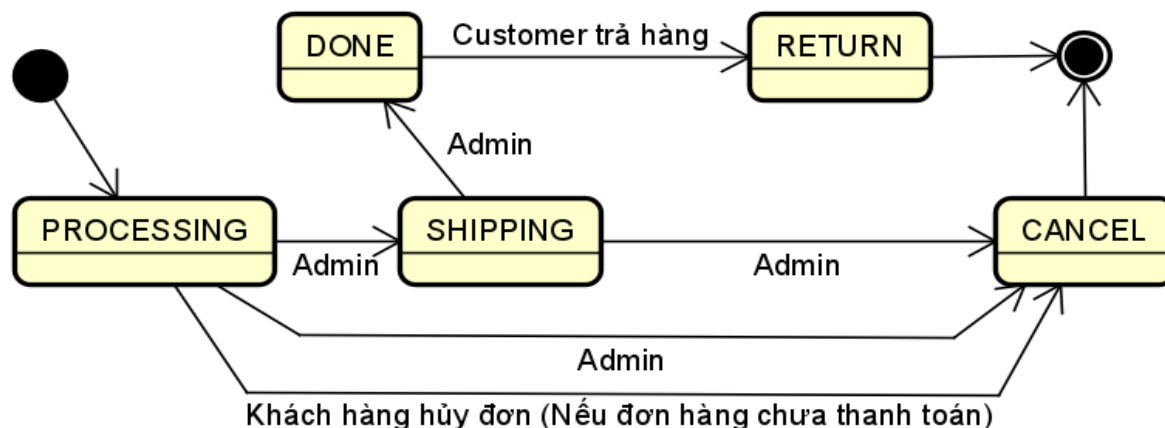


Hình 9. Cơ sở dữ liệu Order

Bảng ShipService và PaymentService được dùng để lưu trữ các thông tin dịch vụ cung cấp bởi bên thứ 3 cho hệ thống.

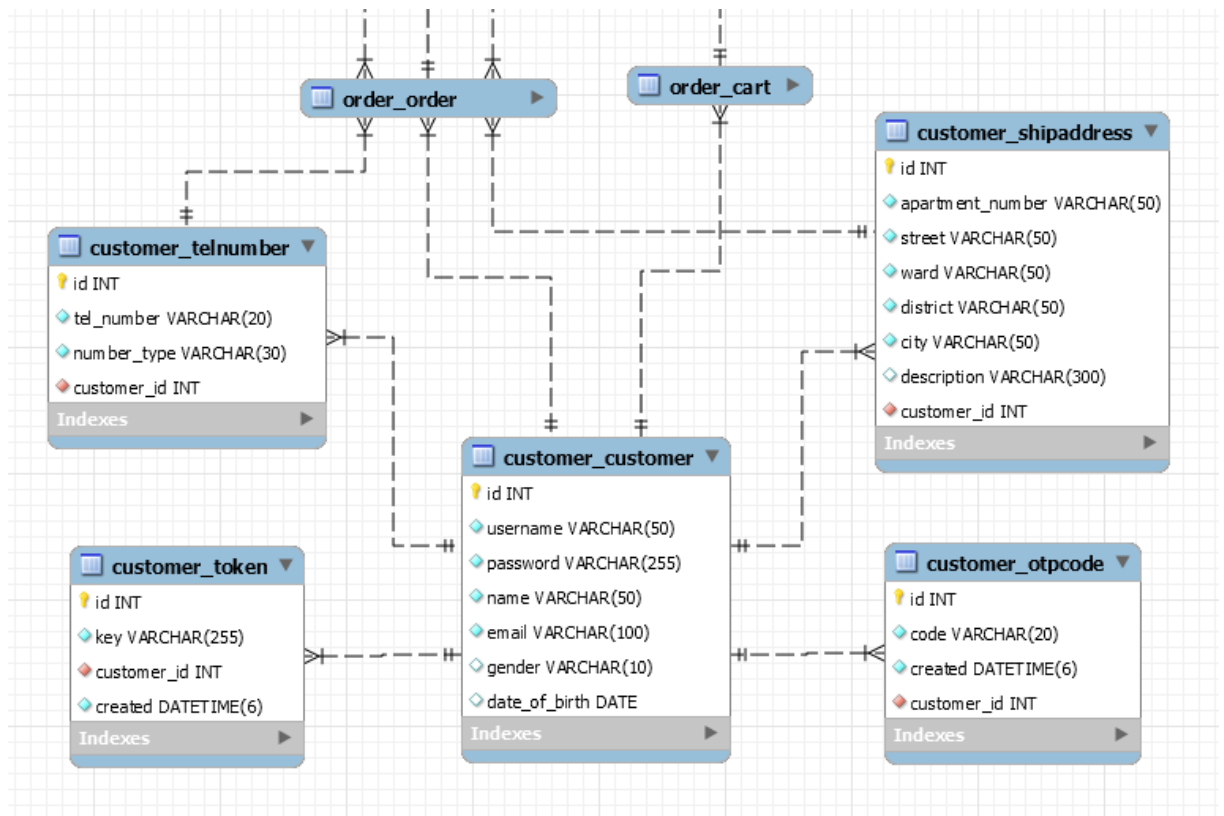
Bảng Cart được dùng để lưu các sản phẩm đã được thêm vào giỏ và hiện có trong giỏ hàng của khách hàng. Có khóa ngoại tham chiếu đến bảng Customer, xác định giỏ hàng đó của khách hàng nào.

Bảng Order dùng để lưu trữ thông tin đơn đặt hàng của khách hàng, các thông tin giao hàng (ship_to, contact_tel, customer_id) là khóa ngoại tham chiếu đến các bảng TelNumber, ShipAddress, Customer của dữ liệu Customer. Thông Stage của bảng Order gồm 5 trạng thái:



Hình 10. Các trạng thái của đơn hàng

Đơn hàng được khởi tạo bởi khách hàng và có trạng thái là PROCESSING. Nhân viên hệ thống sẽ theo dõi, cập nhật các trạng thái cho đơn hàng. Sau khi khách hàng nhận được hàng, đơn hàng sẽ có trạng thái DONE, lúc này khách hàng có thể trả lại hàng nếu không ưng ý. Khách hàng cũng có thể hủy đơn đặt hàng chỉ trong trường hợp đơn hàng đó chưa được thanh toán.



Hình 11. Cơ sở dữ liệu Customer

Bảng TelNumber và ShipAddress được dùng để giúp khách hàng lưu trữ nhiều thông tin để đặt hàng.

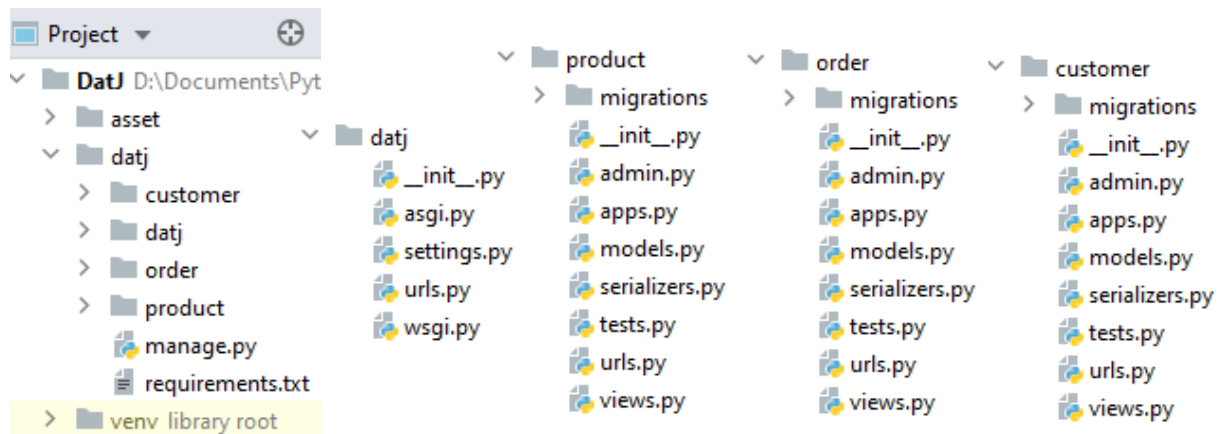
Bảng OTPCode được dùng để lưu mã xác nhận 1 lần, phục vụ cho việc khôi phục tài khoản nếu khách hàng bị quên mật khẩu, và thẻ dùng để xác thực cho các chức năng cần chứng thực.

Token được dùng để lưu trữ token xác thực để thực hiện các giao dịch, thêm sản phẩm vào giỏ, đặt hàng, thay đổi thông tin tài khoản.

3.3. Xây dựng trang quản trị (Admin)

3.3.1. Giai đoạn 1: Tạo cơ sở dữ liệu

Khởi tạo dự án với cấu trúc:



Hình 12. Cấu trúc dự án

Sử dụng phương pháp code-first [2] để tạo dữ liệu cho dự án. Đầu tiên tạo class (Là 1 bảng dữ liệu) cùng các thuộc tính đã được thiết kế ở các phần trên trong models

```
class Discount(models.Model):
    code = models.CharField(max_length=30, unique=True)
    discount_percent = models.FloatField(default=0)
    on_bill = models.BooleanField(default=False)
    active = models.BooleanField(default=True)
    apply_date = models.DateTimeField(default=datetime.now())
    end_date = models.DateTimeField(blank=True, null=True)

    def __str__(self):
        return self.code
```

Sau khi đã tạo các class theo thiết kế trước đó, nhờ vào framework Django em chạy các lệnh sau để tạo cơ sở dữ liệu vào hệ quản trị cơ sở dữ liệu MySQL

```
py manage.py makemigrations
py manage.py migrate
```

3.3.2. Giai đoạn 2: Tạo trang Admin

Tiếp đến, em sử dụng thư viện admin [3] của Django để tạo web với các chức năng cơ bản cho nhân viên.

Tại file admin em tạo 1 class ModelAdmin như sau:

```
class ProductAdmin(admin.ModelAdmin):
    inlines = [ProductAttributeValueInline]
    list_per_page = 12
    search_fields = ('name',)
    list_filter = ('category', 'manufacturer', 'active',)
```

Sau đó đăng ký model để chạy lên trang admin:

```
admin.site.register(Product, ProductAdmin)
```

Thao tác tương tự với các model khác và chỉnh sửa ModelAdmin theo yêu cầu để phù hợp với mô tả trước đó.

Sau khi hoàn tất, em dùng lệnh `py manage.py runserver` để chạy project và có được trang admin cơ bản như sau:

3.3.2.1. Quản lý các dịch vụ bên thứ ba:

Xem danh sách các dịch vụ

The screenshot shows the Django administration interface. The top header is 'Django administration' with a welcome message for 'VU01'. The breadcrumb trail is 'Home > Order > Payment services'. The left sidebar has a section 'ORDER' with two items: 'Payment services' (with a '+ Add' link) and 'Ship services' (with a '+ Add' link). The main content area is titled 'Select payment service to change'. It features a search bar, an 'Action:' dropdown menu, and a 'Go' button. Below these are five payment services listed with checkboxes: 'PAYMENT SERVICE', 'ATM Card', 'Master Card', 'Credit Card', and 'Momo'. At the bottom, it says '5 payment services'. On the right, there is a 'FILTER' sidebar with a 'By active' section containing 'All', 'Yes', and 'No' options. A '+ ADD PAYMENT SERVICE' button is located at the top right of the main content area.

Thêm mới dịch vụ

Django administration

WELCOME, **VU01**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Order > Payment services > Add payment service

ORDER

Payment services [+ Add](#)

Ship services [+ Add](#)

Add payment service

Name:

Zalo Pay

☒ Active

Save and add another

Save and continue editing

SAVE

Chỉnh sửa thông tin hoặc xóa dịch vụ

Django administration

WELCOME, **VU01**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Order > Payment services > Momo

ORDER

Payment services [+ Add](#)

Ship services [+ Add](#)

Change payment service

Name:

Momo

☒ Active

Delete

Save and add another

Save and continue editing

SAVE

HISTORY

3.3.2.2. Quản lý kiểm kê đơn hàng:

Xem danh sách đơn hàng

Django administration

WELCOME, **ADMIN**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Order > Orders

AUTHENTICATION AND AUTHORIZATION

Groups [+ Add](#)

Users [+ Add](#)

ORDER

Order details

Orders

Payment services [+ Add](#)

Ship services [+ Add](#)

PRODUCT

Categories [+ Add](#)

Discount items [+ Add](#)

Discounts [+ Add](#)

Manufacturers [+ Add](#)

Product attribute values [+ Add](#)

Product attributes [+ Add](#)

Select order to view

ORDER	STAGE	PAID DATE
Order-61-nbdat22-11/04/2020, 09:29:29	Processing	Nov. 4, 2020, 4:29 p.m.
Order-60-nbdat22-11/04/2020, 07:09:32	Cancel	-
Order-59-nbdat22-11/02/2020, 10:47:53	Cancel	-
Order-58-nbdat22-11/02/2020, 09:51:46	Return	-
Order-57-nbdat22-11/02/2020, 09:51:27	Done	Nov. 4, 2020, 5:23 p.m.
Order-56-nbdat22-11/02/2020, 09:45:35	Done	Nov. 2, 2020, 5:24 p.m.
Order-55-nbdat22-11/02/2020, 09:45:07	Done	Nov. 2, 2020, 5:24 p.m.
Order-54-nbdat22-11/02/2020, 09:42:11	Cancel	-
Order-53-nbdat22-11/02/2020, 09:38:50	Cancel	-
Order-52-nbdat22-11/02/2020, 09:37:04	Return	-
Order-51-nbdat22-10/30/2020, 11:07:38	Cancel	-
Order-50-nbdat22-10/30/2020, 11:05:09	Done	Oct. 30, 2020, 6:13 p.m.

1 2

19 orders

Show all

FILTER

By stage

All

Processing

Shipping

Done

Cancel

Return

By order date

Any date

Today

Past 7 days

This month

This year

By shipped date

Any date

Today

Past 7 days

This month

This year

3.3.2.3. Quản lý sản phẩm và thông tin khuyến mãi:

Xem danh sách sản phẩm (Tương tự khuyến mãi và các bảng khác)

Django administration WELCOME, KH001. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Product > Products

PRODUCT

- Categorys [+ Add](#)
- Discount items [+ Add](#)
- Discounts [+ Add](#)
- Manufacturers [+ Add](#)
- Product attribute values [+ Add](#)
- Product attributes [+ Add](#)
- Products** [+ Add](#)

Select product to change ADD PRODUCT +

Q

Action: 0 of 12 selected

<input type="checkbox"/>	PRODUCT
<input type="checkbox"/>	Chuột máy tính Asus TUF Gaming M3
<input type="checkbox"/>	Chuột gaming Logitech G403 Hero
<input type="checkbox"/>	Chuột máy tính không dây Logitech Mx Anywhere 2S
<input type="checkbox"/>	Chuột máy tính không dây Logitech M337
<input type="checkbox"/>	Chuột gaming không dây CORSAIR Corsair Dark Core R
<input type="checkbox"/>	Chuột máy tính Corsair Harpoon PRO RGB
<input type="checkbox"/>	Chuột gaming CorSAIR Ironclaw
<input type="checkbox"/>	Bàn phím Logitech K120
<input type="checkbox"/>	Bàn phím cơ Gaming Logitech G Pro X

FILTER

By category

- All
- Laptop
- Màn hình
- Bàn phím
- Chuột
- Thiết bị ngoại vi
-

By manufacturer

- All
- Asus
- Dell
- HP
- CORSAIR
- LOGITECH

Thêm sản phẩm mới (Tương tự khuyến mãi và các bảng khác)

Add product

Category:

Manufacturer:

Name:

Price:

Unit in stock:

Unit in order:

☒ Active

Description:

PRODUCT ATTRIBUTE VALUES




PRODUCT ATTRIBUTE	VALUE	DELETE?
<input type="text" value="-----"/> <input type="button" value="edit"/> <input type="button" value="+"/> <input type="button" value="x"/>	<input type="text"/>	<input type="button" value="x"/>
<input type="text" value="-----"/> <input type="button" value="edit"/> <input type="button" value="+"/> <input type="button" value="x"/>	<input type="text"/>	<input type="button" value="x"/>

[+ Add another Product attribute value](#)

Xóa hoặc chỉnh sửa thông tin sản phẩm (Tương tự khuyến mãi và các bảng khác)

Change product

[HISTORY](#)

Category: Laptop   

Manufacturer: Dell   

Name: Dell G3 Inspiron 3590 N5I5517W

Price: 22990000.0






















Unit in stock: 29

Unit in order: 0

☒ Active

Description:

PRODUCT ATTRIBUTE VALUES

PRODUCT ATTRIBUTE	VALUE	DELETE?
Intel Core i5-9300H 2.4GHz up to 4.1GHz 8MB		
CPU   	Intel Core i5-9300H 2.4GHz up to 4.1GHz 8MB	<input type="checkbox"/>
2 x 4GB DDR4 2666MHz		
Ram   	2 x 4GB DDR4 2666MHz	<input type="checkbox"/>
256GB SSD M.2 PCIE		
Ổ cứng   	256GB SSD M.2 PCIE	<input type="checkbox"/>
15.6" FHD (1920 x 1080) IPS, Anti-Glarec		
Thông số màn hình   	15.6" FHD (1920 x 1080) IPS, Anti-Glarec	<input type="checkbox"/>
3 Cell 56Whr		
Pin   	3 Cell 56Whr	<input type="checkbox"/>
https://product.hstatic.net/1000026716/product/ll-g3-inspiron-3590-n5i5517w_1_0c916cb24b5e4f328e249eb350768f22_master_3aa6f65982af408b913ab1f85f0f3b40.jpg		
Ảnh   	https://product.hstatic.net/1000026716/prod	<input type="checkbox"/>
-----   		<input checked="" type="checkbox"/>

[+ Add another Product attribute value](#)

[Delete](#)[Save and add another](#)[Save and continue editing](#)[SAVE](#)

3.3.2.4. Quản lý nhân sự:

Xem danh sách nhân viên

Django administration

WELCOME, **su01** [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

[Home](#) > [Authentication and Authorization](#) > [Users](#)

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add

Users

+ Add

Select user to change

Q

Search

Action:

Go

 0 of 5 selected

<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	admin	datzach31@gmail.com			✓
<input type="checkbox"/>	don01				✓
<input type="checkbox"/>	kho01				✓
<input type="checkbox"/>	su01				✓
<input type="checkbox"/>	vu01				✓

5 users

FILTER

By staff status

All

Yes

No

By superuser status

All

Yes

No

By active

All

Yes

No

ADD USER +

Thêm nhân viên mới

Add user

First, enter a username and password. Then, you'll be able to edit more user options.

Username:

nhanvienmoi

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password:

*

Password confirmation:

+

Enter the same password as before, for verification.

Save and add another

Save and continue editing

SAVE

Thêm, chỉnh sửa các thông tin của nhân viên

Change user

[HISTORY](#)

Username:	<input type="text" value="nhanviemmoi"/>
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.	
Password:	algorithm: pbkdf2_sha256 iterations: 216000 salt: 8bUri7***** hash: MykXUO*****
Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using this form .	

Personal info

First name:	<input type="text" value="Văn A"/>
Last name:	<input type="text" value="Nguyễn"/>
Email address:	<input type="text" value="test123@gmail.com"/>

Permissions

☒ **Active**
Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

☒ **Staff status**
Designates whether the user can log into this admin site.

Phân quyền cho nhân viên vừa thêm

Groups:

<div>Available groups ?</div> <div><input type="text" value="Filter"/></div> <div><div>Nhân viên dịch vụ</div><div>Nhân viên kho</div><div>Nhân viên nhân sự</div><div>Nhân viên thống kê đơn</div></div>	<div>Chosen groups ?</div> <div></div>
<div>Choose all ?</div>	<div>Remove all</div>

The groups this user belongs to. A user will get all permissions granted to each of their groups. Hold down "Control", or "Command" on a Mac, to select more than one.

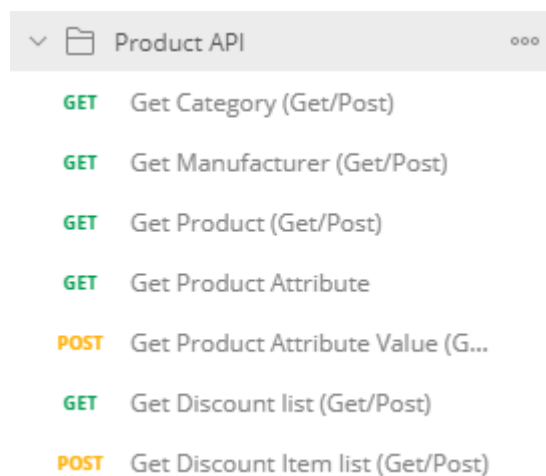
User permissions:

<div>Available user permissions ?</div> <div><input type="text" value="Filter"/></div> <div><div>admin log entry Can add log entry</div><div>admin log entry Can change log entry</div><div>admin log entry Can delete log entry</div><div>admin log entry Can view log entry</div><div>auth group Can add group</div><div>auth group Can change group</div><div>auth group Can delete group</div></div>	<div>Chosen user permissions ?</div> <div></div>
--	--

3.4. Xây dựng các API theo yêu cầu hệ thống

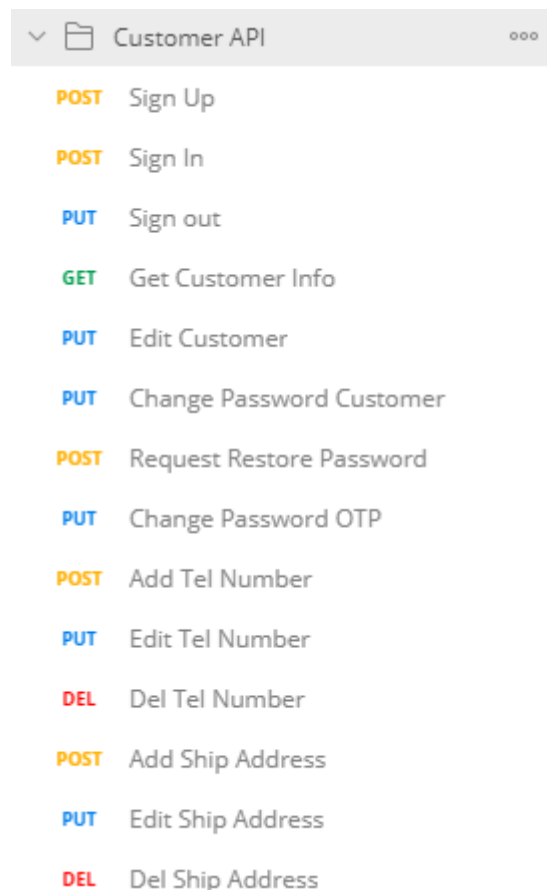
Dựa theo sơ đồ usecase và class ở trên, em phân làm 3 nhóm API chính là:

- API Product: Các API dùng để tải dữ liệu, thông tin về sản phẩm và các thông tin liên quan, xoay quanh đến sản phẩm đó (Như là Category, Manufacturer, Discount). Các API của Product sẽ bao gồm như sau:



Hình 13. Danh sách API Product

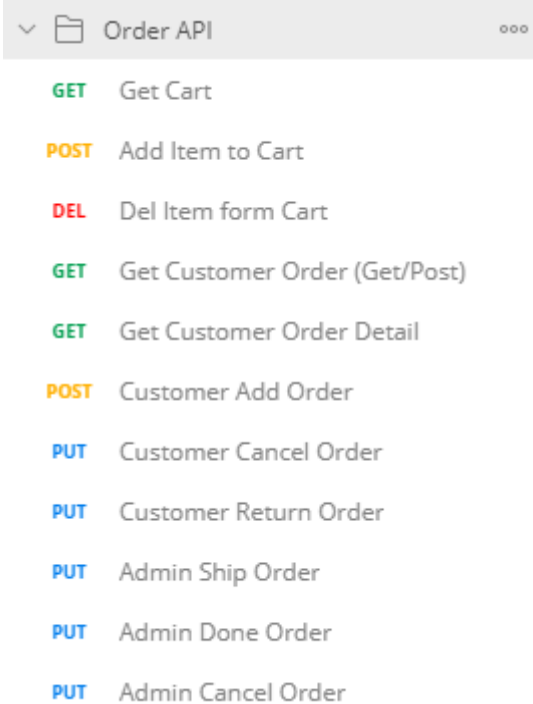
- API Customer: Các API này giúp khách hàng tương tác với hệ thống, giúp khách hàng đăng ký, đăng nhập, đăng xuất và sửa thông tin tài khoản, thêm mới hoặc sửa xóa các thông tin đặt hàng. Các API này sẽ được chứng thực bằng token [5]. API sẽ dùng thư viện mail [6] của Django để gửi mã xác thực 1 lần về mail của khách hàng khi có nhu cầu lấy lại mật khẩu. Các API của Customer sẽ bao gồm như sau:



	Customer API
POST	Sign Up
POST	Sign In
PUT	Sign out
GET	Get Customer Info
PUT	Edit Customer
PUT	Change Password Customer
POST	Request Restore Password
PUT	Change Password OTP
POST	Add Tel Number
PUT	Edit Tel Number
DEL	Del Tel Number
POST	Add Ship Address
PUT	Edit Ship Address
DEL	Del Ship Address

Hình 14. Danh sách API Customer

- API Order: Các API này được dùng để phục vụ khách hàng trong việc thêm, xóa sản phẩm từ giỏ hàng, đặt hàng, xem lịch sử thông tin đơn hàng, hủy đơn và trả hàng. Các API này sẽ được chứng thực bằng token [5]. Ngoài ra còn có các API dành cho nhân viên để thực hiện các thao tác cập nhật trạng thái của các đơn hàng. Các API này cũng sẽ được chứng thực bằng token [5] của nhân viên. Các API của Order sẽ bao gồm như sau:



Order API	
GET	Get Cart
POST	Add Item to Cart
DEL	Del Item form Cart
GET	Get Customer Order (Get/Post)
GET	Get Customer Order Detail
POST	Customer Add Order
PUT	Customer Cancel Order
PUT	Customer Return Order
PUT	Admin Ship Order
PUT	Admin Done Order
PUT	Admin Cancel Order

Hình 15. Danh sách API Order

Em dùng thư viện `rest_framework` [4] của Django để xây dựng, viết các API và các API này sẽ được viết theo chuẩn REST [7].

API được viết gồm 3 công đoạn:

- **Serializer:**

Tạo các serializer để đọc dữ liệu từ request khi gửi API lên server và trả dữ liệu về khi server response các API đó.

Code để tạo 1 serializer dùng để response từ server khi gọi 1 API:

```
class GetProAttributeValueSerializer(serializers.ModelSerializer):
    class Meta:
        model = ProductAttributeValue
        fields = ('pk', 'product', 'product_attribute', 'value',)
```

Code để tạo 1 serializer dùng để đọc dữ liệu từ request khi gửi API lên server:

```
class KeywordAttributeValueSerializer(serializers.Serializer):
    pk = serializers.IntegerField(default=0)
    product = serializers.IntegerField(default=0)
    product_attribute = serializers.IntegerField(default=0)
    value = serializers.CharField(max_length=100,
default="null")
```

- **Views:**

Tại đây gồm nhiều class để xử lý cho các API khi gửi lên server với method khác nhau (GET/ POST/ PUT/ DELETE/ ...).

APIView dùng để xử lý method POST để lấy và trả về dữ liệu giá trị thuộc tính của 1 sản phẩm:

Tạo class cho API `GetProAttributeValueAPIView`

```
class GetProAttributeValueAPIView(APIView):
```

Tạo hàm để xử lý method POST cho API trên

```
def post(self, request):
    mydata = KeywordAttributeValueSerializer(data=request.data)
    if not mydata.is_valid():
        return Response('Something wrong! Check your data',
            status=status.HTTP_400_BAD_REQUEST)

    pk = mydata.data['pk']
    product = mydata.data['product']
    product_attribute = mydata.data['product_attribute']
    value = mydata.data['value']
    list_product_attribute_value =
ProductAttributeValue.objects.all()

    if pk != 0:
        list_product_attribute_value =
list_product_attribute_value.filter(pk=pk)
    else:
        if product != 0:
            list_product_attribute_value =
list_product_attribute_value.filter(product=product)
            if product_attribute != 0:
                list_product_attribute_value =
list_product_attribute_value.filter(product_attribute=product_attrib
ute)
                if value != "null":
                    list_product_attribute_value =
list_product_attribute_value.filter(value__icontains=value)

    mydata =
GetProAttributeValueSerializer(list_product_attribute_value,
many=True)
    return Response(data=mydata.data, status=status.HTTP_200_OK)
```

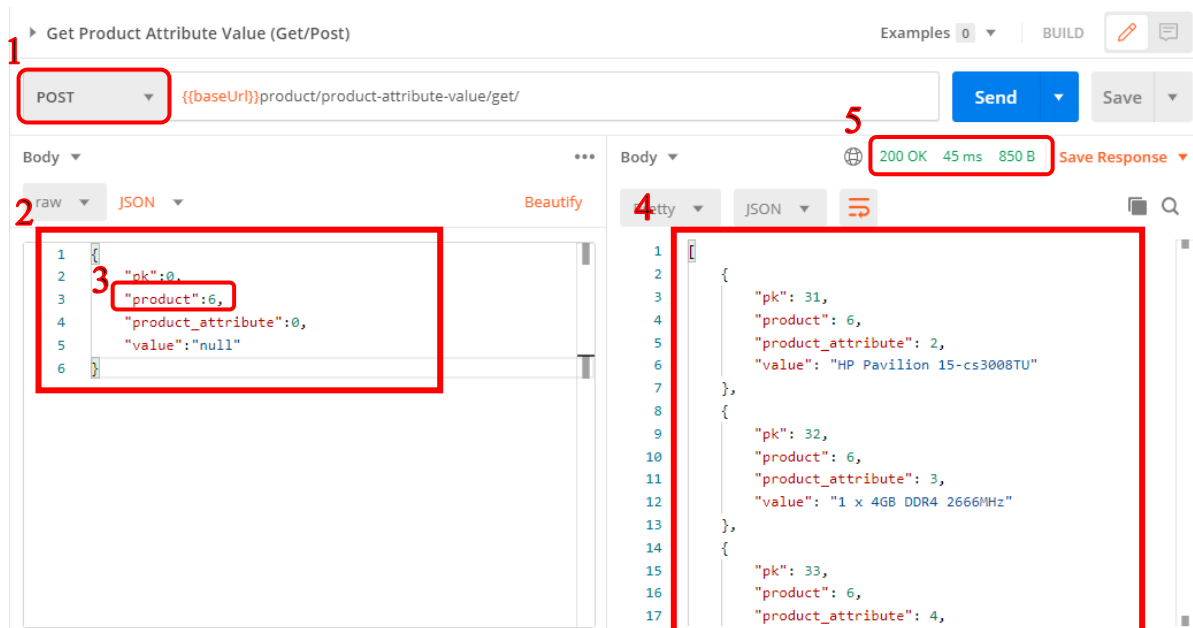
Sau các bước đó em đã có được 1 API xử lý lấy giá trị thuộc tính của sản phẩm theo các từ khóa được gửi lên server.

- Urls:

Đây là bước cuối cùng giúp cho ta có thể gọi API để sử dụng. Vào file urls của dự án, định nghĩa đường dẫn và chỉ định đường dẫn sẽ gọi đến API nào để xử lý bằng cách thêm đoạn code sau vào `urlpatterns = []`:

```
path('product/product-attribute-value/get/',  
GetProAttributeValueAPIView.as_view()),
```

Kết quả khi chạy API đã tạo trên:



Hình 16. Ví dụ mẫu gọi API

- 1: Gọi API với method POST
- 2: Danh sách dữ liệu gửi lên request
- 3: Tìm giá trị thuộc tính của sản phẩm có mã là 6
- 4: Kết quả trả về từ API
- 5: Status code trả về từ API

3.4.1. API Product

Cách làm tương tự với các bước trên, em có được các API sau:

GET Get Category (Get/Post)

API dùng để lấy danh sách các loại sản phẩm (GET), và lọc lấy theo điều kiện của dữ liệu gửi lên server (POST).

Kết quả:

Get Category (Get/Post) Examples 0 BUILD

GET {{baseUrl}}product/category/get/ Send Save

Params Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Pretty JSON

```
1 {
2   {
3     "pk": 1,
4     "name": "Laptop",
5     "description": null,
6     "category_belong": null
7   },
8   {
9     "pk": 2,
10    "name": "Màn hình",
11    "description": null,
12    "category_belong": null
13  },
14  {
15    "pk": 3,
16    "name": "Bàn phím",
17    "description": null,
```

Hình 17. API Get Category GET: Ok

Tìm loại sản phẩm với tên có chứa chuỗi “lap”.

Get Category (Get/Post) Examples 0 BUILD

POST {{baseUrl}}product/category/get/ Send Save

Body raw JSON Beautify

```
1 {
2   "pk": 0,
3   "name": "lap",
4   "category_belong": 0
5 }
```

Body Pretty JSON

```
1 {
2   {
3     "pk": 1,
4     "name": "Laptop",
5     "description": null,
6     "category_belong": null
7   },
8 }
```


GET Get Manufacturer (Get/Post)

API dùng để lấy danh sách các nhà sản xuất (GET), và lọc lấy theo điều kiện của dữ liệu gửi lên server (POST).

Kết quả:

GET `{{baseUrl}}product/manufacturer/get/` **Send** **Save**

Params **Query Params**

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body **200 OK 33 ms 504 B** **Save Response**

```
1 {
2   {
3     "pk": 1,
4     "name": "Asus",
5     "description": null
6   },
7   {
8     "pk": 2,
9     "name": "Dell",
10    "description": null
11  },
12  {
13    "pk": 3,
14    "name": "HP",
15    "description": null
16  },
17  }
```

Hình 18. API Get Manufacturer GET: Ok

Tìm nhà sản xuất với tên có chứa chuỗi “sus”.

POST `{{baseUrl}}product/manufacturer/get/` **Send** **Save**

Body **raw** **JSON** **Beautify**

```
1 {
2   "pk": 0,
3   "name": "sus"
4 }
```

Body **200 OK 30 ms 330 B** **Save Response**

```
1 {
2   {
3     "pk": 1,
4     "name": "Asus",
5     "description": null
6   },
7 }
```

GET Get Discount list (Get/Post)

API dùng để lấy danh sách các thông tin khuyến mãi (GET), và lọc lấy theo điều kiện của dữ liệu gửi lên server (POST).

Kết quả:

The screenshot shows a REST client interface for the endpoint `GET {{baseUrl}}product/discount/get/`. The response is a 200 OK with a status of 171 ms and 992 B. The response body is displayed in JSON format, showing two discount items:

```
[{"pk": 1, "code": "TEST01", "discount_percent": 0.1, "on_bill": true, "active": true, "apply_date": "2020-10-25T20:28:39+07:00", "end_date": "2020-10-27T20:32:01+07:00"}, {"pk": 2, "code": "Asus11", "discount_percent": 0.15, "on_bill": false, "active": true, "apply_date": "2020-10-28T17:07:41+07:00"}]
```

Hình 19. API Get Discount GET: Ok

Lấy khuyến mãi có mã 4 và điều kiện là giảm giá trên toàn bill.

The screenshot shows a REST client interface for the endpoint `POST {{baseUrl}}product/discount/get/`. The request body is a JSON object with the following values:

```
{ "pk": 4, "on_bill": false, "code": "", "active": null }
```

The response is a 200 OK with a status of 34 ms and 426 B. The response body is displayed in JSON format, showing one discount item:

```
[{"pk": 4, "code": "Mouse101", "discount_percent": 0.2, "on_bill": false, "active": true, "apply_date": "2020-11-01T22:36:43+07:00", "end_date": null}]
```

POST Get Discount Item list (Get/Post)

API dùng để lấy danh sách khuyến mãi của các sản phẩm (GET), và lọc lấy theo điều kiện của dữ liệu gửi lên server (POST).

Kết quả:

Get Discount Item list (Get/Post) Examples 0 BUILD

GET http://127.0.0.1:8000/product/discount-item/get/ Send Save

Params Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body 200 OK 40 ms 963 B Save Response

```
1 {
2   {
3     "pk": 5,
4     "discount": 2,
5     "product": 1
6   },
7   {
8     "pk": 6,
9     "discount": 2,
10    "product": 2
11  },
12  {
13    "pk": 7,
14    "discount": 2,
15    "product": 7
16  },
17  {
```

Tìm khuyến mãi của sản phẩm mã 22 với mã khuyến 4.

Get Discount Item list (Get/Post) Examples 0 BUILD

POST http://127.0.0.1:8000/product/discount-item/get/ Send Save

Body raw JSON Beautify

```
1 {
2   "pk": 0,
3   "discount": 4,
4   "product": 22
5 }
```

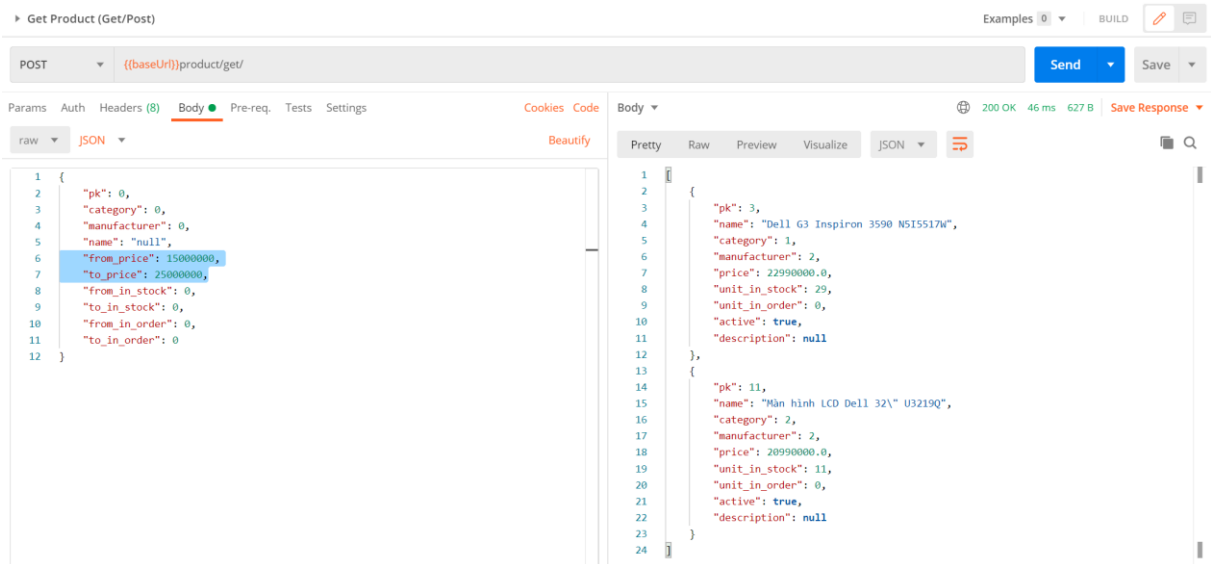
Body 200 OK 26 ms 324 B Save Response

```
1 {
2   {
3     "pk": 13,
4     "discount": 4,
5     "product": 22
6   }
7 }
```

GET Get Product (Get/Post)

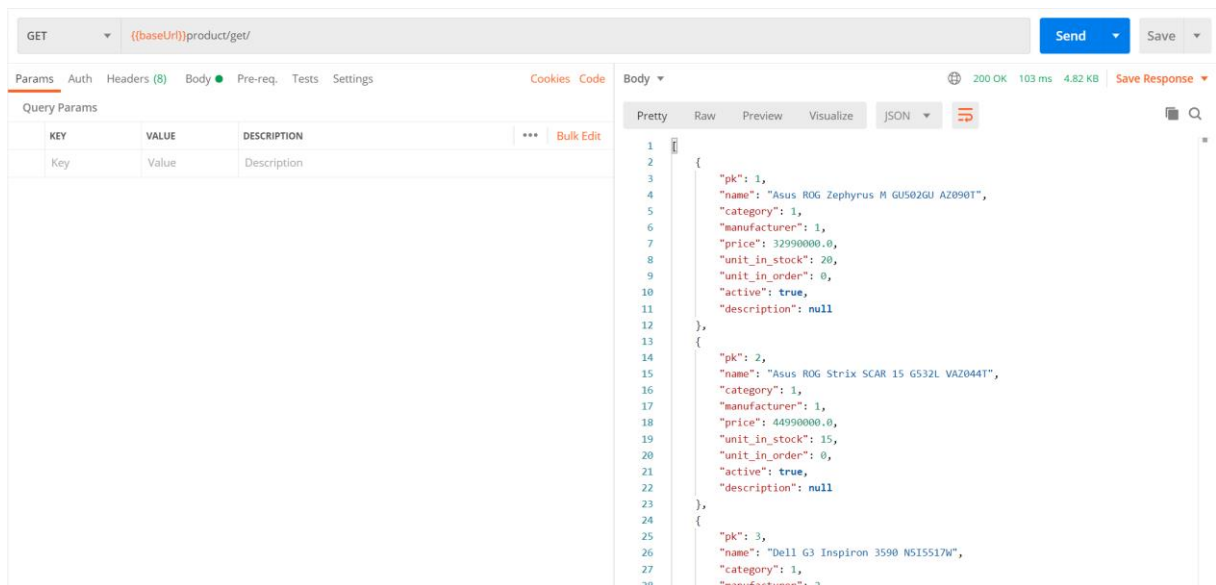
API dùng để lấy danh sách tất cả sản phẩm (GET), và lọc lấy theo điều kiện của dữ liệu gửi lên server (POST).

Kết quả:



Hình 20. API Get Product GET: Ok

Lấy các sản phẩm có giá từ 15000000 đến 25000000.



GET Get Product Attribute

API dùng để lấy danh sách tất cả thuộc tính của tất cả sản phẩm (GET).

Kết quả:

The screenshot shows a REST client interface for the endpoint `GET {{baseUrl}}product/product-attribute/get/`. The response status is 200 OK, with a response time of 22 ms and a body size of 1.37 KB. The response body is displayed in JSON format, showing an array of 6 product attributes.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 {
2   {
3     "pk": 1,
4     "name": "Thông số màn hình",
5     "description": null
6   },
7   {
8     "pk": 2,
9     "name": "CPU",
10    "description": null
11  },
12  {
13    "pk": 3,
14    "name": "Ram",
15    "description": null
16  },
17  {
18    "pk": 4,
19    "name": "Ổ cứng",
20    "description": null
21  },
22  {
23    "pk": 5,
24    "name": "Pin",
25    "description": null
26  },
27  {
28    "pk": 6,
29    "name": "Ảnh",
30    "description": null
31  }
32 }
```

Hình 21. API Get Product Attribute: Ok

POST Get Product Attribute Value (Get/Post)

API dùng để lấy danh sách tất cả giá trị thuộc tính của các sản phẩm (GET), và lọc lấy theo điều kiện của dữ liệu gửi lên server (POST).

Kết quả:

The screenshot shows a REST client interface with a GET request to the endpoint `{{baseUrl}}product/product-attribute-value/get/`. The response is a JSON array of product attributes, displayed in the 'Pretty' view. The response status is 200 OK, with a response time of 62 ms and a size of 15.65 KB.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 {
2   {
3     "pk": 1,
4     "product": 1,
5     "product_attribute": 1,
6     "value": "15.6\" FHD (1920 x 1080) IPS, 100% sRGB, 240Hz, 3ms, 300nits,
7       Pantone® Validated, NanoEdge"
8   },
9   {
10    "pk": 2,
11    "product": 1,
12    "product_attribute": 2,
13    "value": "Intel Core i7-9750H 2.6GHz up to 4.5GHz 12MB"
14  },
15  {
16    "pk": 3,
17    "product": 1,
18    "product_attribute": 3,
19    "value": "16GB DDR4 2666MHz Onboard (1x SO-DIMM socket, up to 32GB SDRAM)"
20  },
21  {
22    "pk": 4,
23    "product": 1,
24    "product_attribute": 4,
25    "value": "512GB SSD PCIE G3X4 (Support RAID 0) (2 slots)"
26  },
27 }
```

Hình 22. API Get Product Attribute Value: Ok

Lấy danh sách giá trị thuộc tính của sản phẩm có mã là 5.

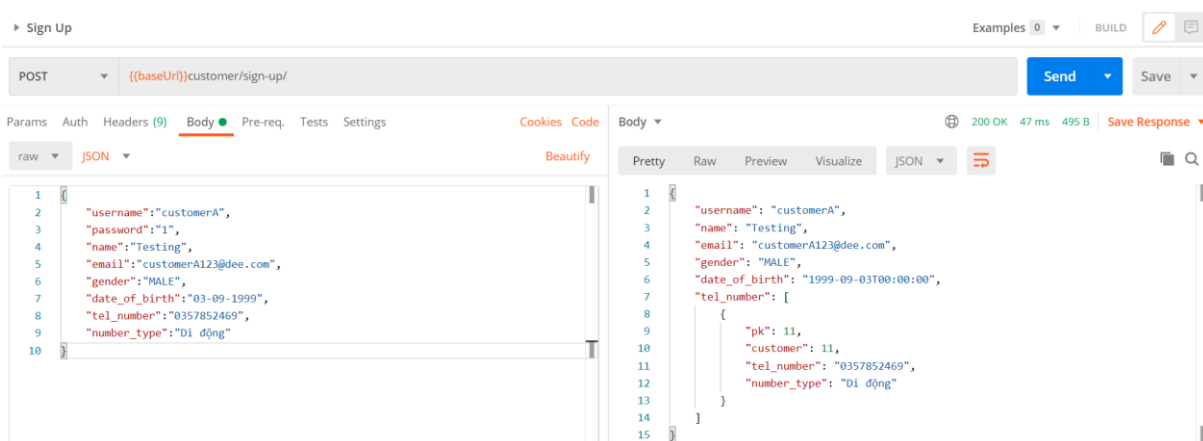
The screenshot shows a REST client interface with a POST request to the endpoint `{{baseUrl}}product/product-attribute-value/get/`. The request body is a JSON object: `{ "pk": 0, "product": 5, "product_attribute": 0, "value": "null" }`. The response is a JSON array of product attributes, displayed in the 'Pretty' view. The response status is 200 OK, with a response time of 30 ms and a size of 858 B.

```
1 {
2   {
3     "pk": 0,
4     "product": 5,
5     "product_attribute": 0,
6     "value": "null"
7   },
8   {
9     "pk": 25,
10    "product": 5,
11    "product_attribute": 2,
12    "value": "HP ENVY 13-aq1057TX"
13  },
14  {
15    "pk": 26,
16    "product": 5,
17    "product_attribute": 3,
18    "value": "1 x 8GB Onboard DDR4 2400MHz"
19  },
20  {
21    "pk": 27,
22    "product": 5,
23    "product_attribute": 4,
24    "value": "512GB SSD M.2 NVMe"
25  },
26  {
27    "pk": 28,
28    "product": 5,
29    "product_attribute": 1,
30    "value": "13.3\" IPS (1920 x 1080)"
31  },
32 }
```

3.4.2. API Customer

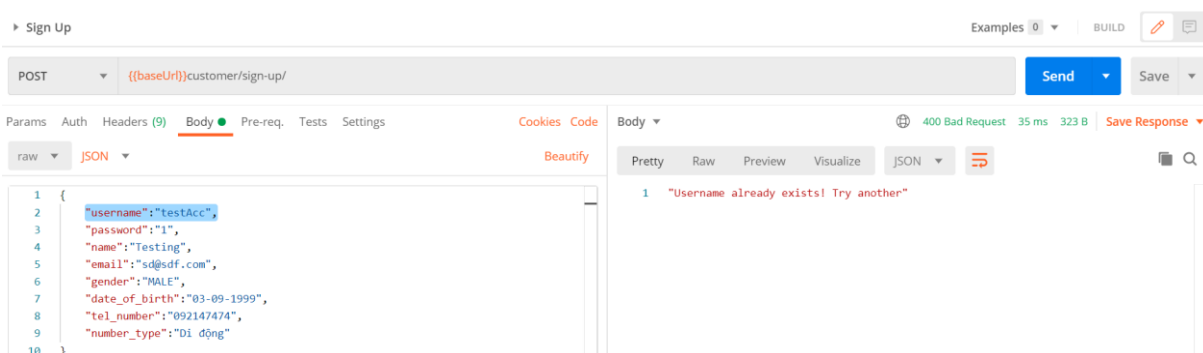
POST Sign Up

Đây là API với method POST giúp khách hàng tạo tài khoản mới vào hệ thống bằng việc cung cấp các thông tin cần thiết.

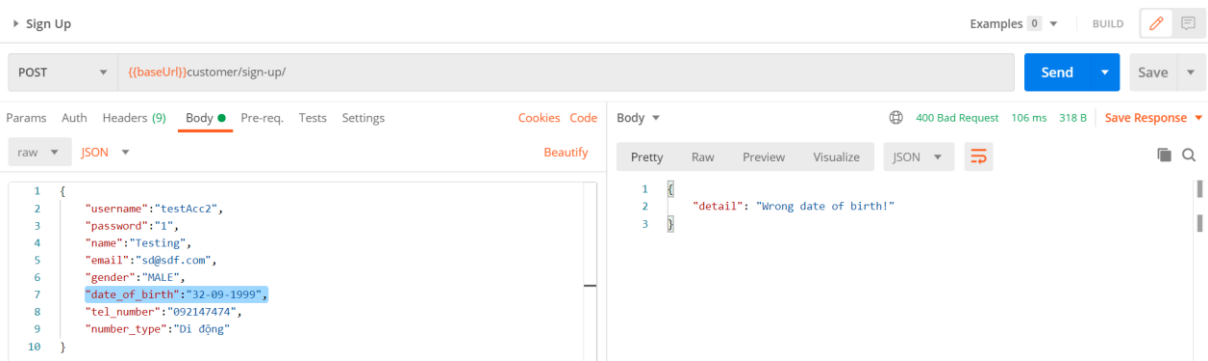


Hình 23. API Sign Up: Ok

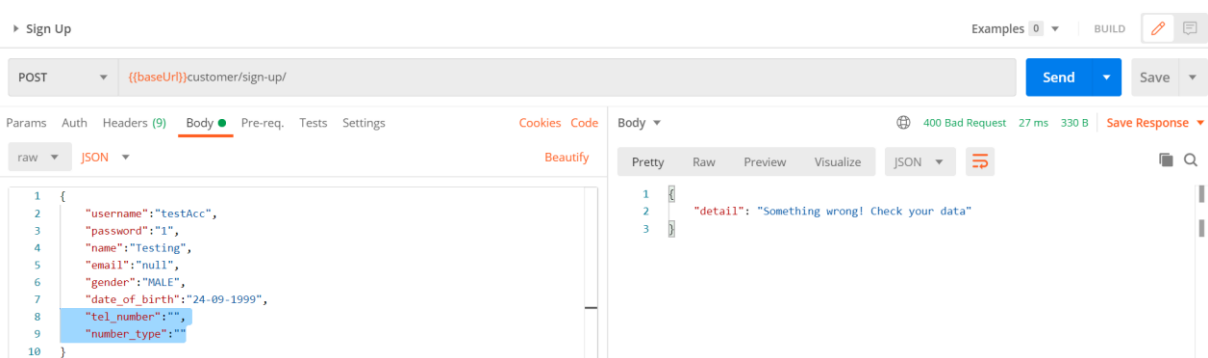
API sẽ báo lỗi nếu trùng tên tài khoản, email, thiếu hoặc sai các thông tin cần thiết cho việc tạo tài khoản.



Hình 24. API Sign Up: Tên tài khoản đã tồn tại



Hình 25. API Sign Up: Sai thông tin ngày sinh



Hình 26. API Sign Up: Sai thông tin SĐT

Trước khi lưu thông tin tài khoản khách hàng vào cơ sở dữ liệu, mật khẩu của khách hàng sẽ được mã hóa bằng hàm băm sha256 cung cấp bởi thư viện hashlib trong python để tăng tính bảo mật.

Đoạn code em dùng để băm mật khẩu

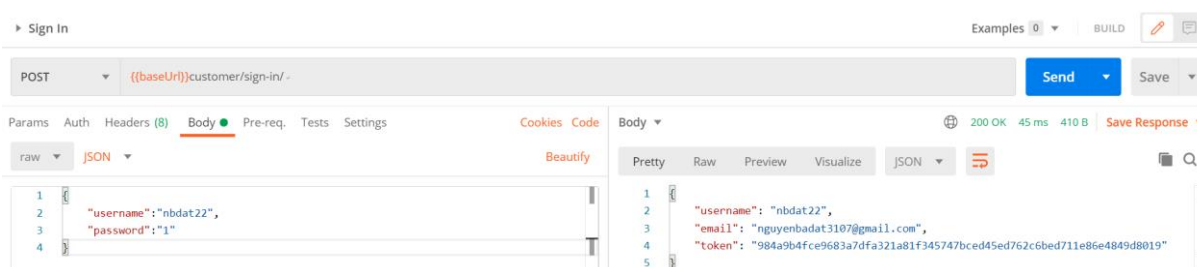
```
password = hashlib.sha256(password.strip().encode("utf-8")).hexdigest()
```

Kết quả sau khi tạo tài khoản khách hàng bằng API Sign Up:

11	customerA	6b86b273ff34fce19d6b804eff5a3f5747ada4ea...	Testing	customerA123@dee.com	MALE	1999-09-03
----	-----------	---	---------	----------------------	------	------------

POST Sign In

API phục vụ cho việc khách hàng đăng nhập vào hệ thống. Sau khi cung cấp thông tin đăng nhập hợp lệ, API sẽ trả về 1 token xác thực, khách hàng (web hoặc các ứng dụng khác) sẽ sử dụng token này để thực hiện các chức năng khác có yêu cầu xác thực bằng token.



Hình 27. API Sign In: Ok

Token được tạo ra bằng cách băm chuỗi ghép bởi tên đăng nhập của khách hàng và ngày giờ gửi API Sign In:

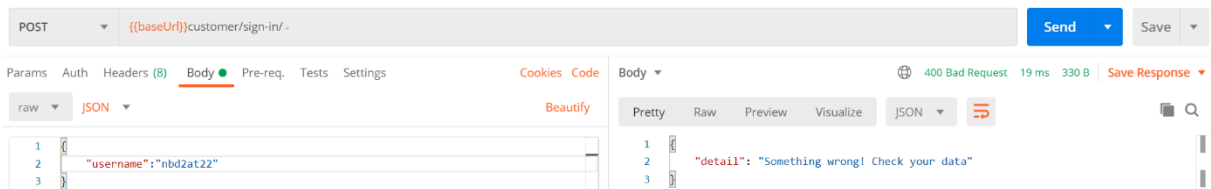
```
key = hashlib.sha256((customer.username +  
str(datetime.now()))).encode("utf-8")).hexdigest()
```

Token với khóa mới được tạo sẽ được lưu xuống cơ sở dữ liệu cùng với ngày giờ tạo khóa đó.

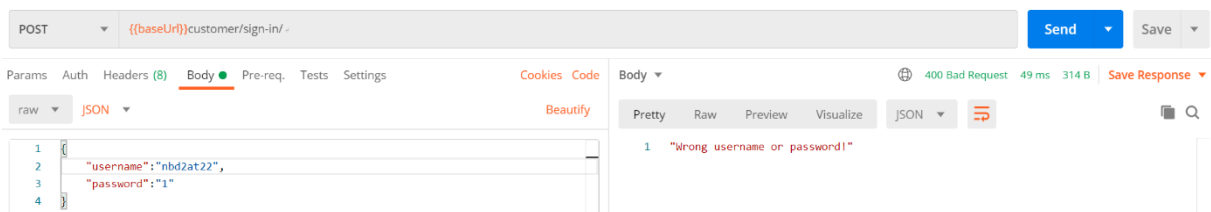
id	key	customer_id	created
1	984a9b4fce9683a7dfa321a81f345747bcd45e...	3	2020-11-03 15:24:55.409049

Mỗi khách hàng chỉ có một mã token duy nhất, sẽ được tạo mới mỗi lần đăng nhập vào hệ thống và các token đó sẽ tự hết hạn sau một tiếng đồng hồ.

API sẽ báo lỗi khi khách hàng nhập sai tên đăng nhập, mật khẩu hoặc thiếu một trong hai thông tin đó.



Hình 28. API Sign In: Sai thông tin đăng nhập

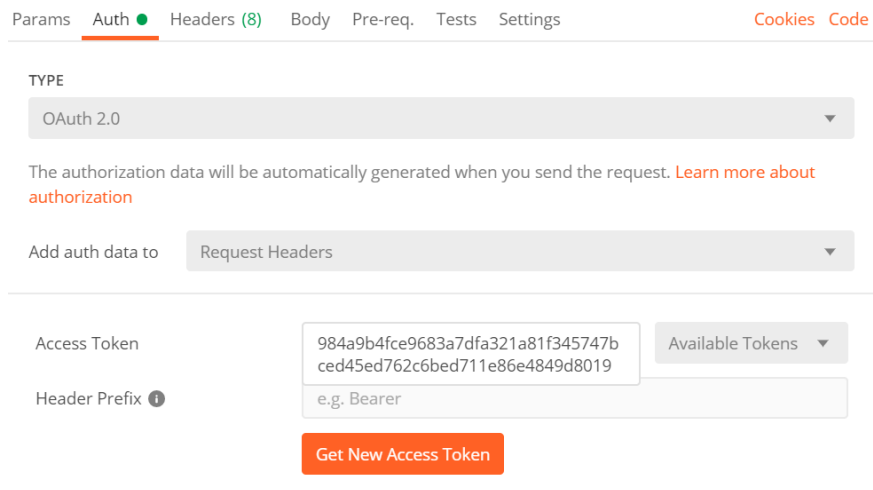


Hình 29. API Sign In: Thiếu thông tin đăng nhập

Xác thực token:

Sau chức năng Sign In, hệ thống đã có được token để xác thực cho các API tiếp theo.

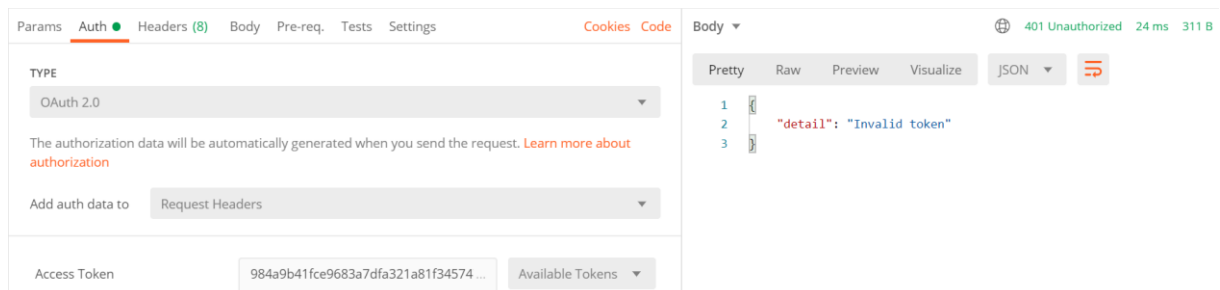
Các API có yêu cầu về xác thực sẽ phải thêm thông Authorization vào headers trước khi gọi API:



Hình 30. Thêm token xác thực vào headers của API

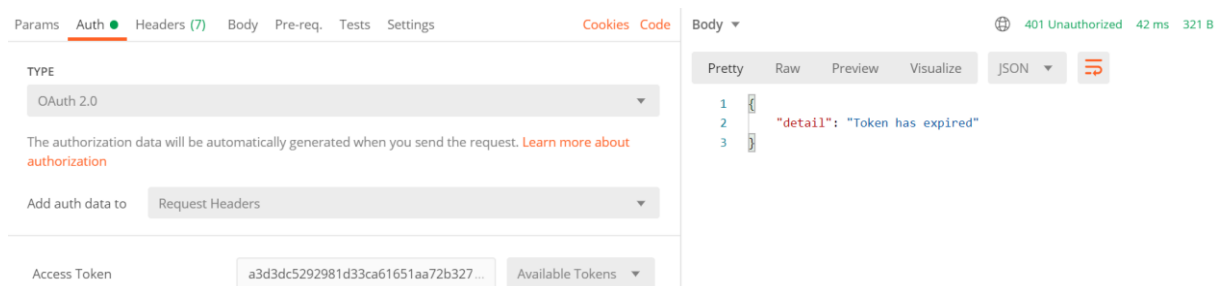
Khi gửi API đi, server sẽ kiểm tra token đó có hợp lệ hay không với các điều kiện:

- Token phải tồn tại trong kho dữ liệu.



Hình 31. Xác thực token, token không tồn tại

- Token vẫn còn hạn sử dụng (1 tiếng từ khi khách hàng đăng nhập tạo mới token).



Hình 32. Xác thực token, token hết hạn

Code dùng để xác thực token:

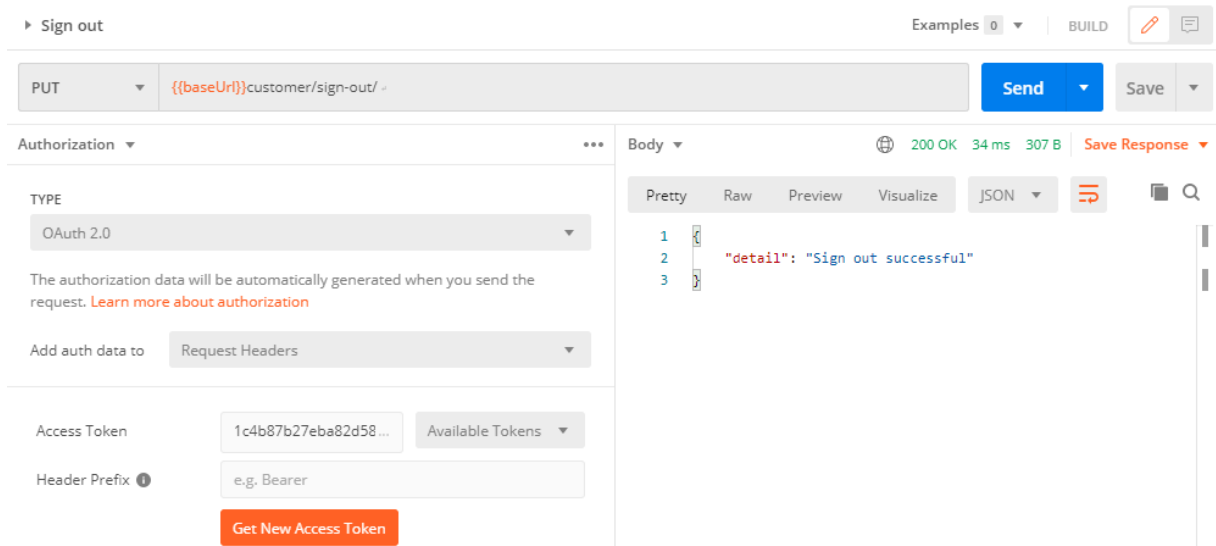
```
try:
    token = request.headers['Authorization']
except:
    return Response({
        "detail": "Token not found"
    },
    status=status.HTTP_203_NON_AUTHORITATIVE_INFORMATION)
token = Token.objects.filter(key=token).first()
if token is None:
    return Response({
        "detail": "Invalid token"
    }, status=status.HTTP_401_UNAUTHORIZED)
if (token.created + timedelta(hours=1)) <
datetime.now(timezone.utc):
    return Response({
        "detail": "Token has expired"
    }, status=status.HTTP_401_UNAUTHORIZED)
```

Sau khi xác thực token thành công, API được phép thực hiện chức năng của nó và API đó có thể lấy các thông tin tài khoản khách hàng từ token đó bằng đoạn code sau:

```
customer =
Customer.objects.filter(pk=token.customer.pk).first()
```

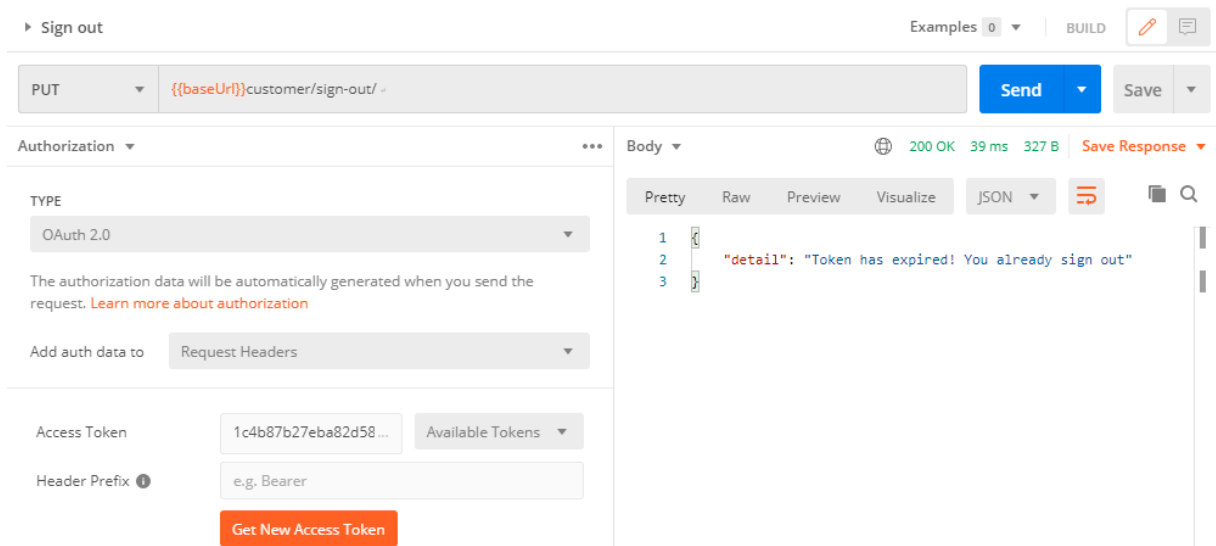
PUT Sign out [Xác thực token]

API giúp khách hàng đăng xuất bằng cách vô hiệu hóa token của khách hàng đó.



Hình 33. API Sign Out: Ok

API sẽ thông báo cho khách hàng khi token đó đã quá hạn (đã được sign out).

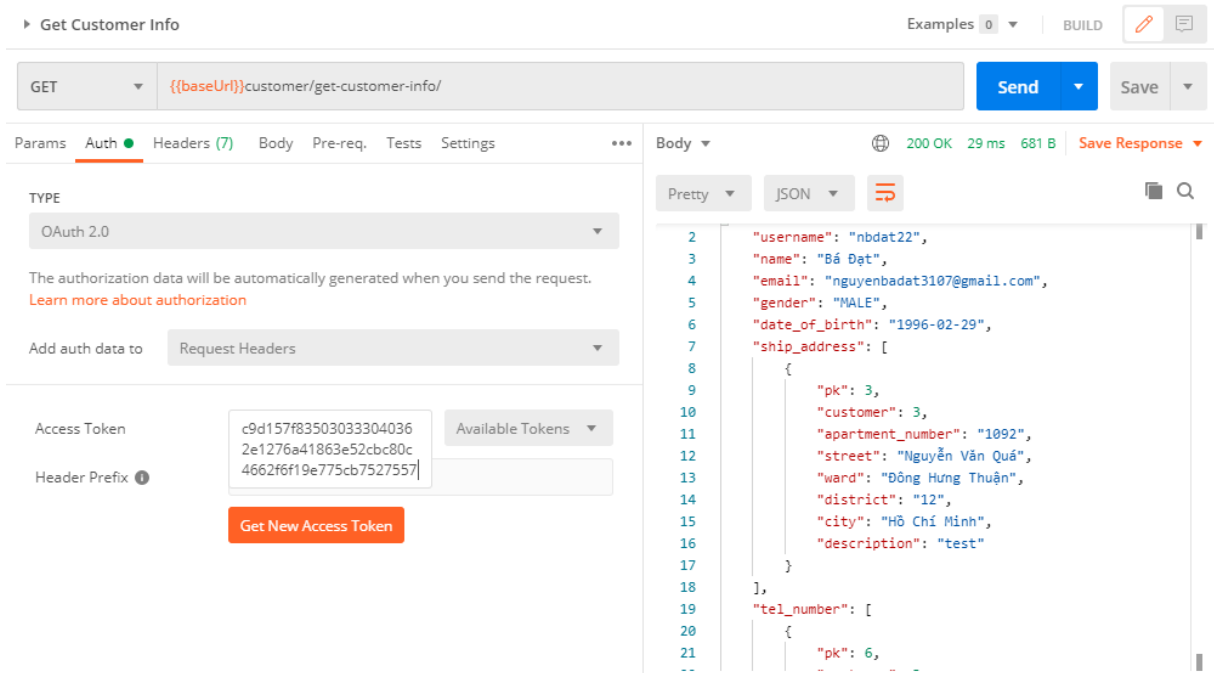


Hình 34. API Sign Out: Token quá hạn

GET Get Customer Info

[Xác thực token]

API dùng để lấy các thông tin tài khoản của khách hàng.

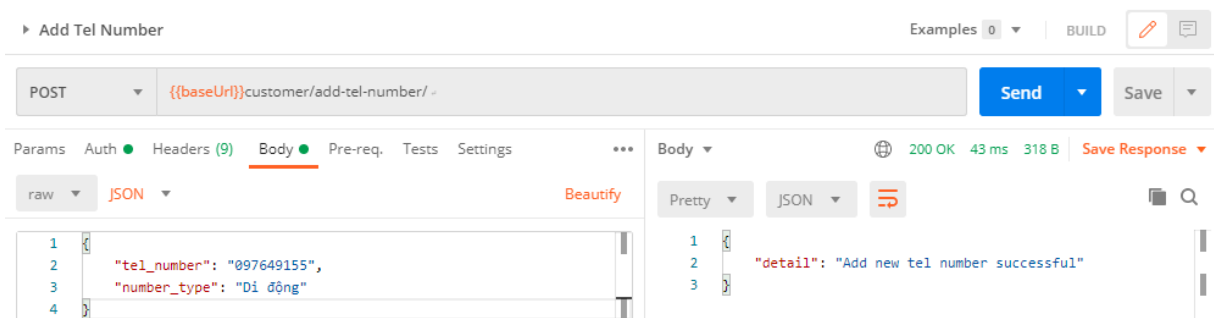


Hình 35. API Get Customer Info: Ok

POST Add Tel Number

[Xác thực token]

API thêm số điện thoại mới cho khách hàng. (Báo lỗi nếu không đủ thông tin)

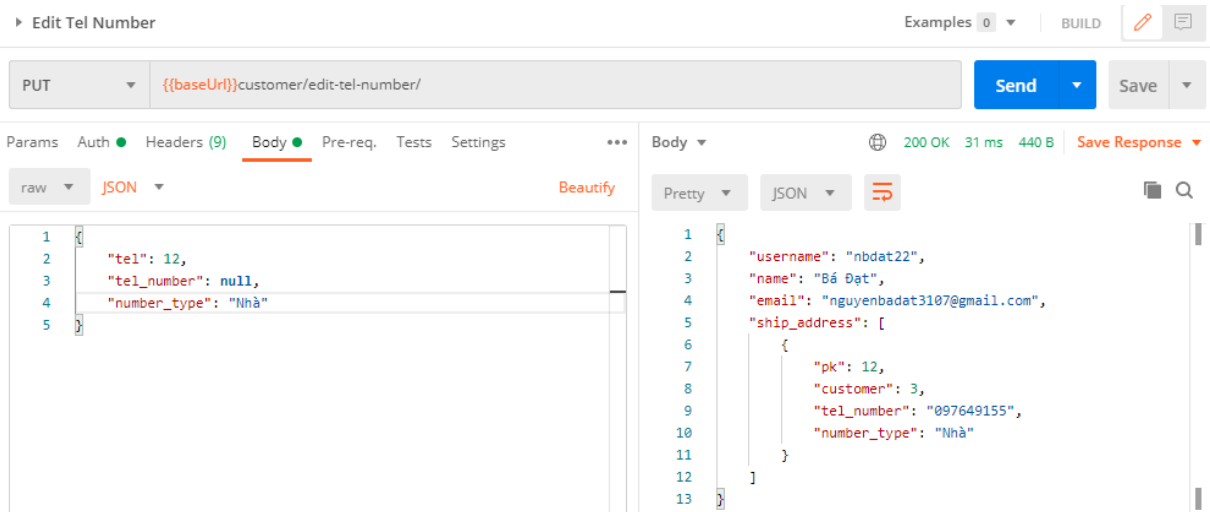


Hình 36. API Add Tel Number: Ok

PUT Edit Tel Number

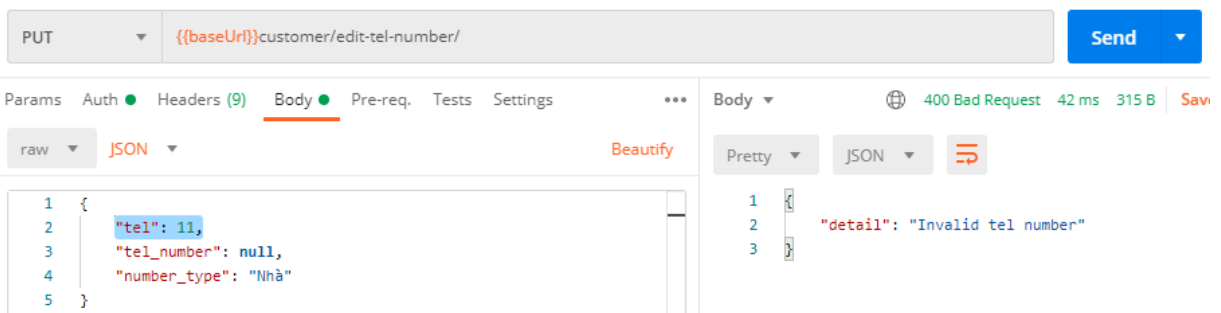
[Xác thực token]

API sửa thông tin số điện thoại của khách hàng.



Hình 37. API Edit Tel Number: Ok

API này cũng sẽ kiểm tra mã SĐT vừa nhận có phải thuộc sở hữu của khách hàng gửi API đó đi hay không qua token khách hàng vừa gửi đi. Nếu không khớp sẽ báo lỗi.

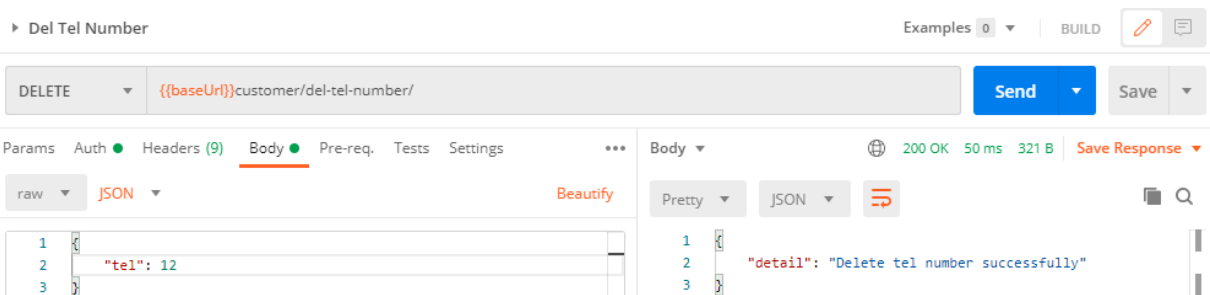


Hình 38. API Edit Tel Number: Sai mã SĐT

DEL Del Tel Number

[Xác thực token]

API xóa số điện thoại của khách hàng.

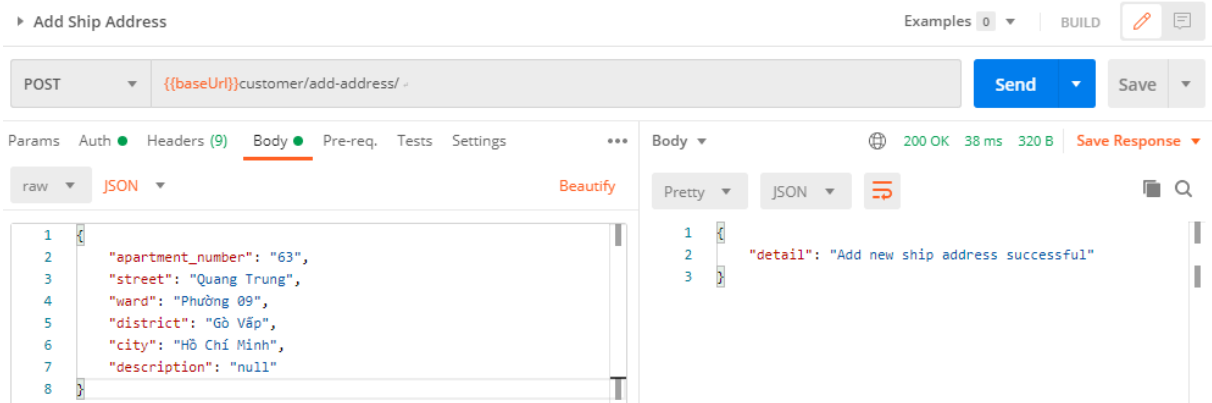


Hình 39. API Delete Tel Number: Ok

POST Add Ship Address

[Xác thực token]

API thêm thông tin địa chỉ giao hàng mới cho khách hàng. (Báo lỗi nếu không đủ thông tin)

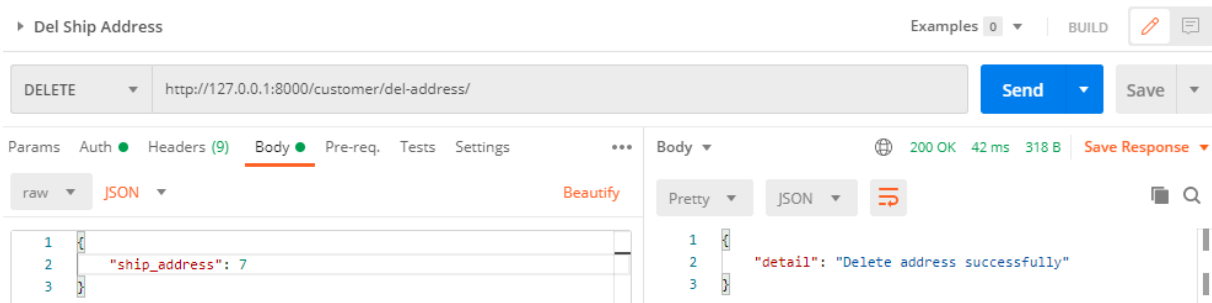


Hình 40. API Add Ship Address: Ok

DEL Del Ship Address

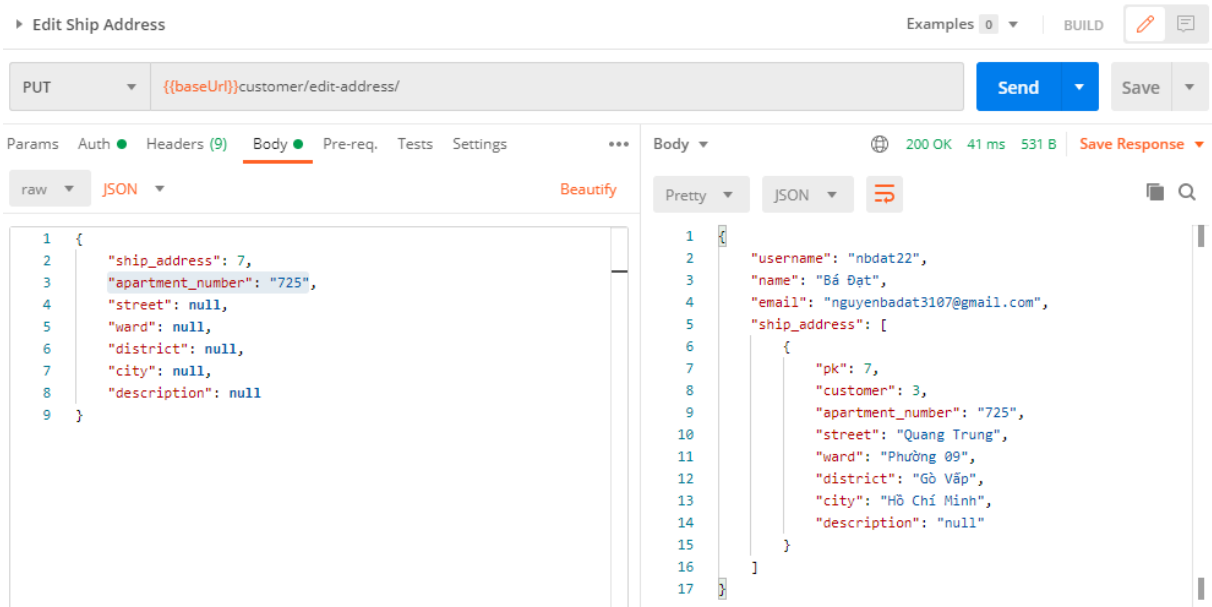
[Xác thực token]

API xóa địa chỉ giao hàng.



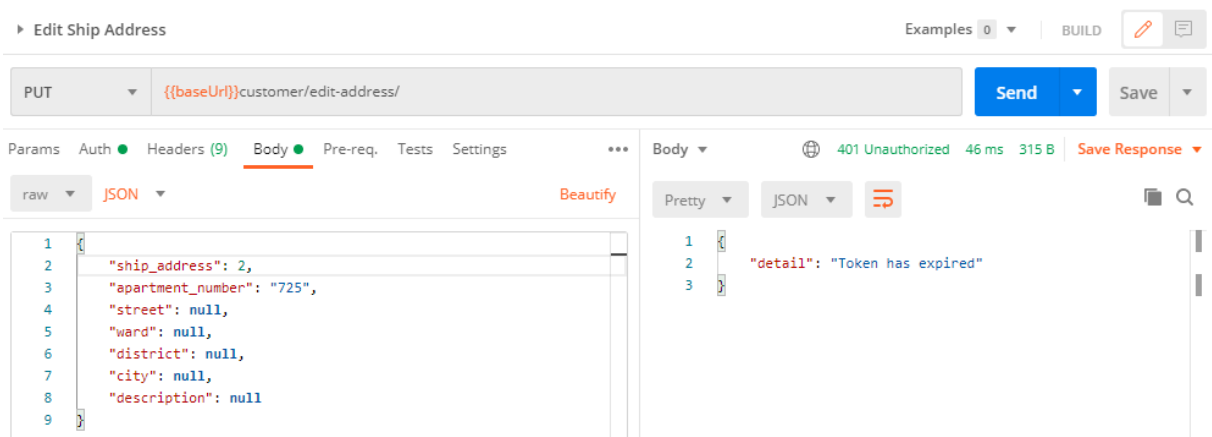
Hình 41. API Delete Ship Address: Ok

API dùng để sửa thông tin địa chỉ giao hàng. API chỉ sửa những thông tin gửi lên có giá trị khác null.



Hình 42. API Edit Ship Address: Ok

API này cũng sẽ kiểm tra mã địa chỉ vừa nhận có phải thuộc sở hữu của khách hàng gửi API đó đi hay không qua token khách hàng vừa gửi đi. Nếu không khớp sẽ báo lỗi.

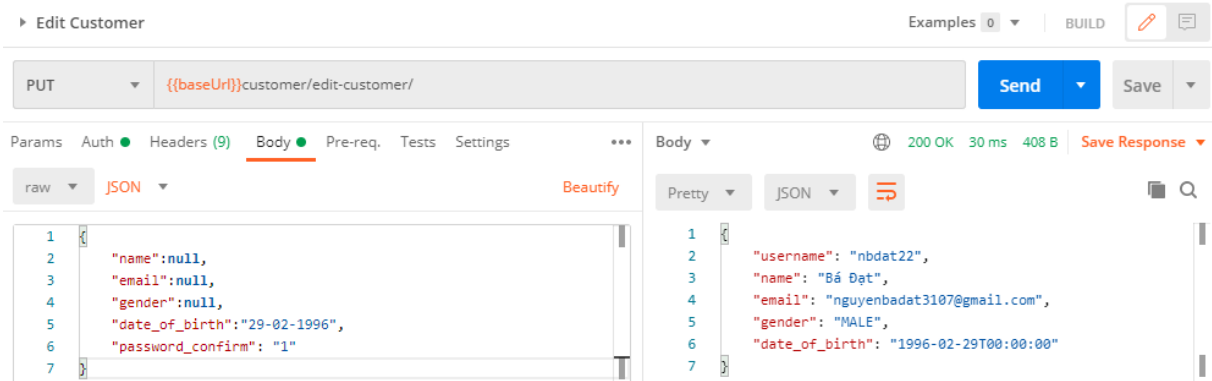


Hình 43. API Edit Ship Address: Sai mã địa chỉ

PUT Edit Customer

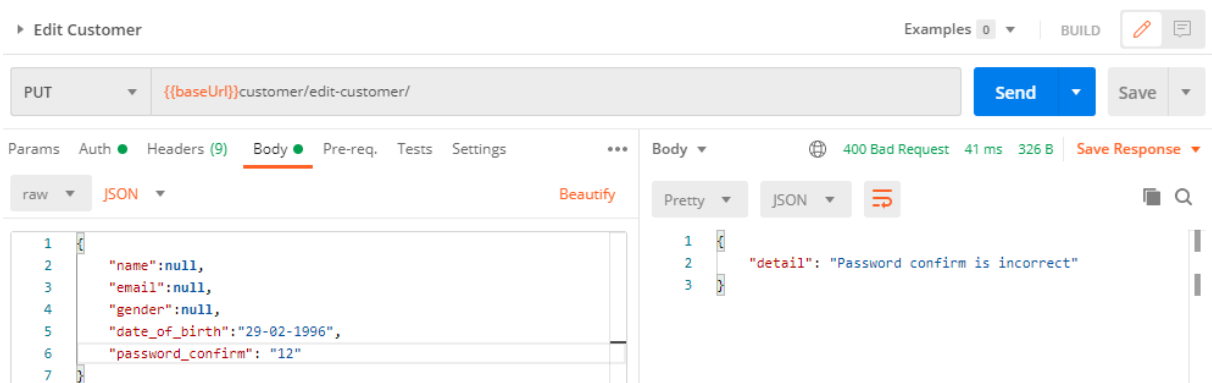
[Xác thực token]

API sửa thông tin tài khoản của khách hàng. API chỉ sửa những thông tin gửi đi có giá trị khác null.



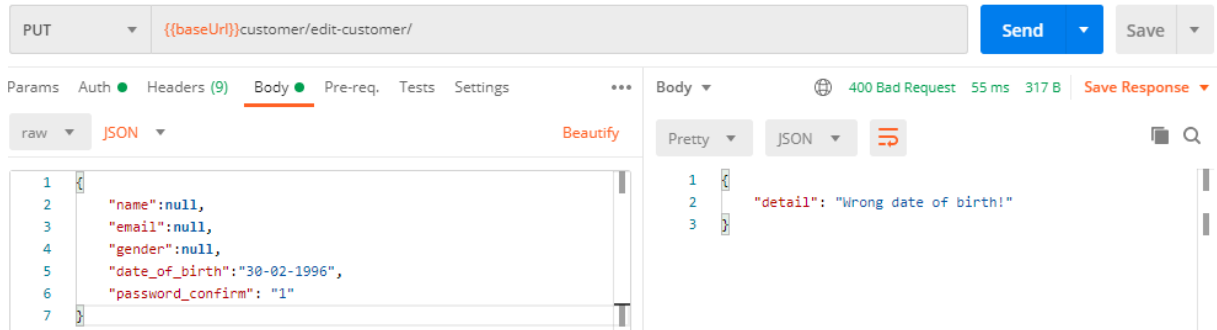
Hình 44. API Edit Customer

API báo lỗi nếu mật khẩu xác thực không đúng.

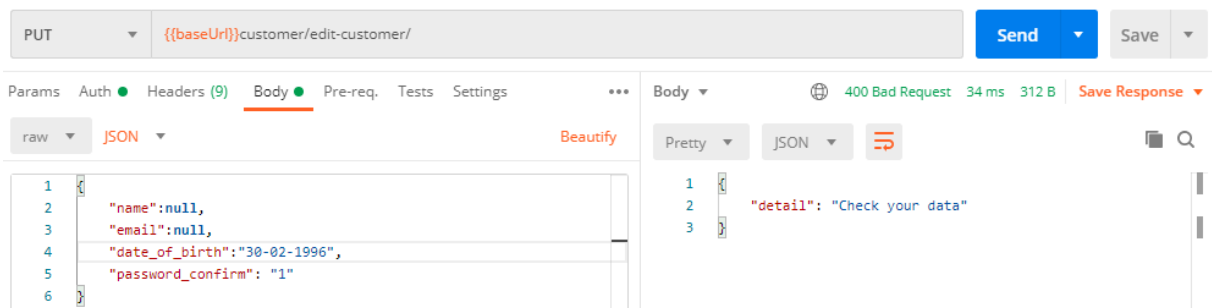


Hình 45. API Edit Customer: Sai mật khẩu

API báo lỗi nếu thiếu thông tin gửi đi hoặc sai dữ liệu ngày tháng năm sinh.



Hình 46. API Edit Customer: Sai dữ liệu ngày sinh

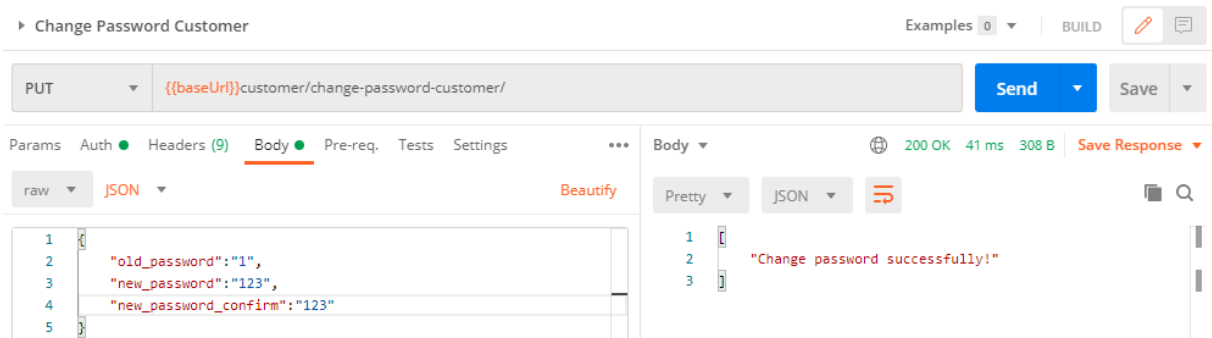


Hình 47. API Edit Customer: Gửi thiếu thông tin

PUT Change Password Customer

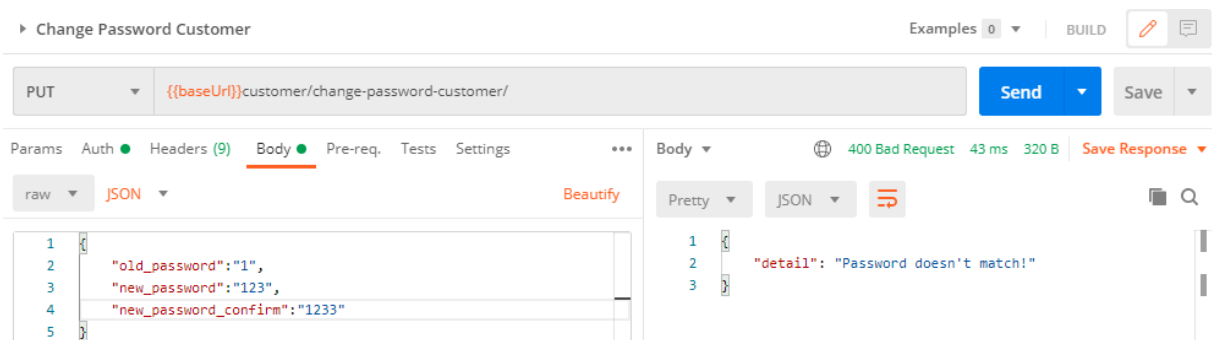
[Xác thực token]

API hỗ trợ khách hàng đổi mật khẩu tài khoản.



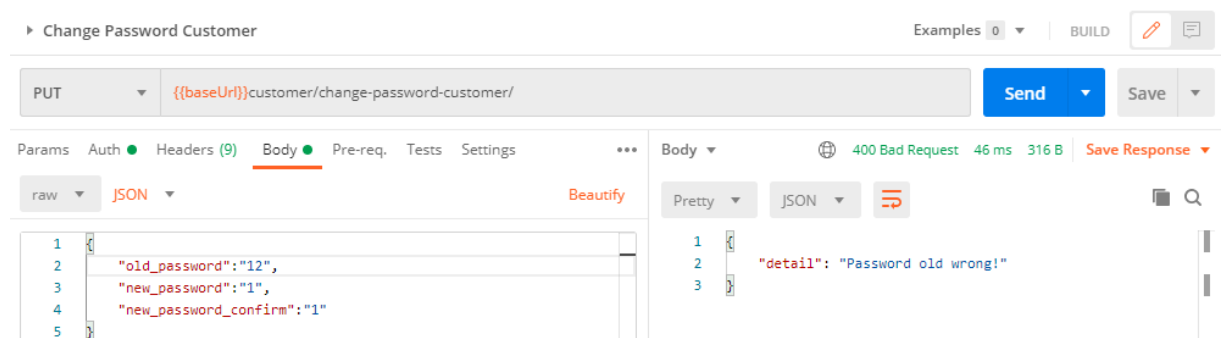
Hình 48. API Change Password: Ok

API báo lỗi nếu mật khẩu mới không trùng khớp.



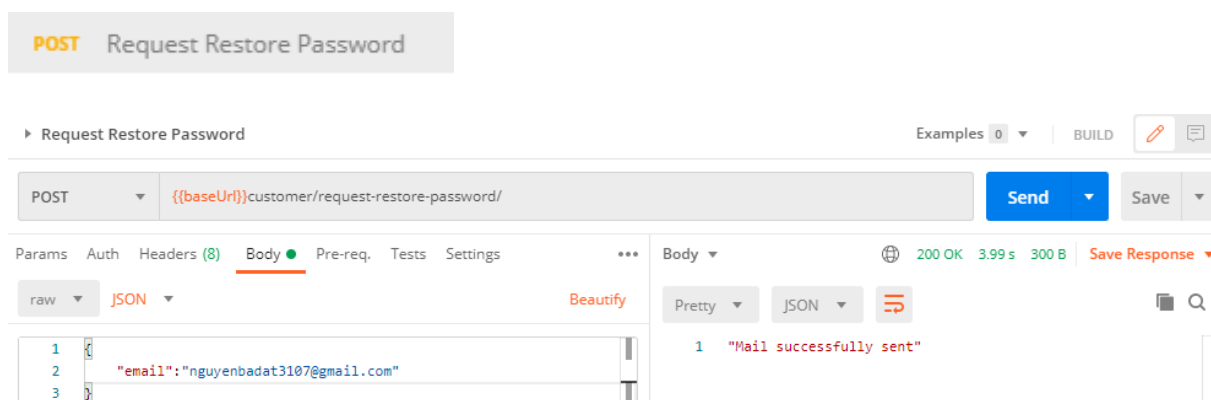
Hình 49. API Change Password: Mật khẩu không khớp

API báo lỗi nếu mật khẩu cũ không chính xác.

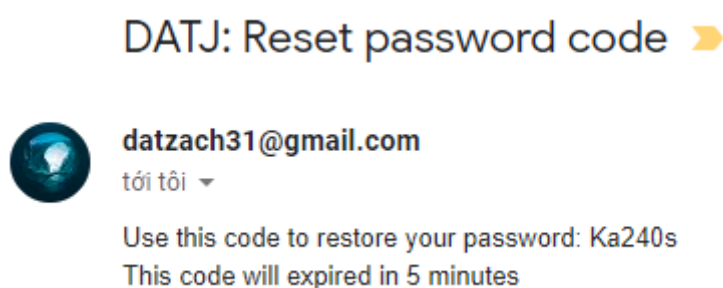


Hình 50. API Change Password: Sai mật khẩu cũ

Trong trường hợp khách hàng bị quên mật khẩu, họ có thể gọi API sau để gửi mã OTP qua email và dùng mã đó để khôi phục đặt lại mật khẩu mới.

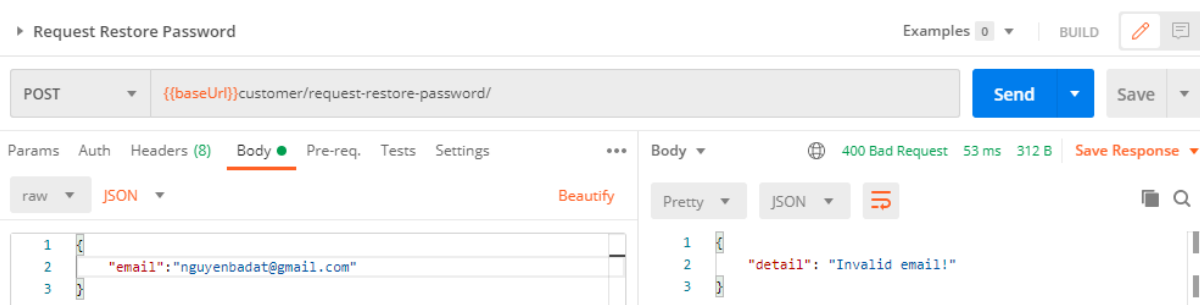


Hình 51. API Request Restore Password: Ok



Hình 52. Mail mã xác thực khôi phục mật khẩu

API sẽ kiểm tra email gửi đi có khớp với tài khoản khách hàng nào trong kho hay không, nếu email gửi đi đó không thuộc bất kì tài khoản nào hệ thống sẽ báo lỗi.



Hình 53. API Request Restore Password: Email không tồn tại

Nếu email hợp lệ, API sẽ tạo một đoạn mã xác thực liên kết đến tài khoản khách hàng có email vừa gửi kia và lưu đoạn mã đó xuống cơ sở dữ liệu. Sau khi tạo và lưu mã, đoạn mã đó sẽ được gửi cho khách hàng thông qua email vừa cung cấp, khách hàng sau đó có thể sử dụng đoạn mã đó để khôi phục đặt lại mật khẩu mới.

Code để tạo đoạn mã ngẫu nhiên gồm 6 ký tự:

```
letters_and_digits = string.ascii_letters + string.digits
code = ''.join((random.choice(letters_and_digits) for i in
range(6)))
```

Em dùng thư viện email [6] của Django để gửi thông tin khôi phục mật khẩu qua email của khách hàng bằng đoạn code sau:

```
content = "Use this code to restore your password: " + code +
"\nThis code will expired in 5 minutes"

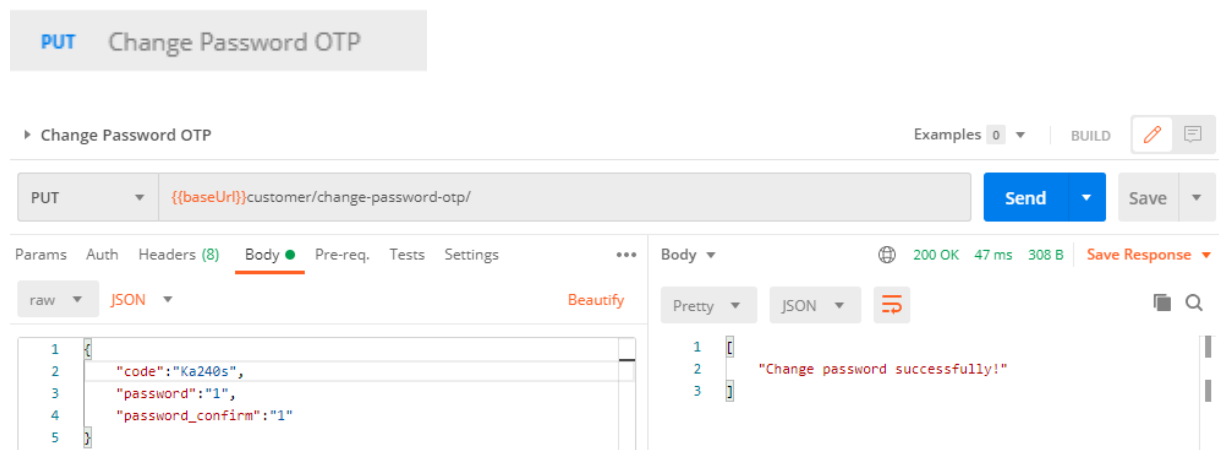
send_mail(
    'DATJ: Reset password code',
    content,
    settings.EMAIL_HOST_USER,
    [email],
    fail_silently=False,
)
```

`settings.EMAIL_HOST_USER` là email được dùng để gửi mail đi, được cấu hình tại file `settings` của dự án.

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_USE_TLS = True
EMAIL_PORT = 587
EMAIL_HOST_USER = '[user mail]'
EMAIL_HOST_PASSWORD = '[user mail password]'
```

`[email]` là địa chỉ dùng để gửi mail đến.

Sau khi khách hàng đã nhận được mail với đoạn mã khôi phục, họ có thể dùng API sau để đặt lại mật khẩu.

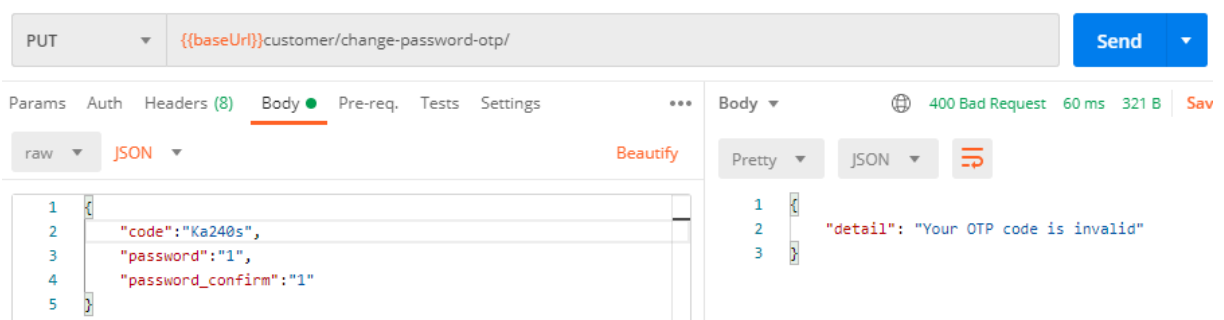


Hình 54. API Change Password OTP: Ok

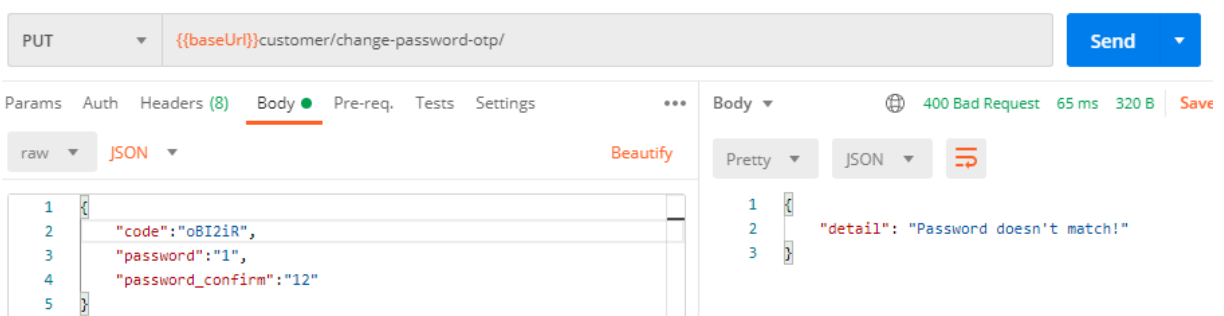
Do đoạn mã OTP đó đã được liên kết với tài khoản của khách hàng, nên API dễ dàng lấy thông tin khách hàng đó ra và thay đổi mật khẩu của họ sau khi API đã kiểm tra đoạn mã hợp lệ.

Sau khi đổi mật khẩu thành công, API sẽ vô hiệu hóa đoạn mã đó.

API sẽ báo lỗi khi mã xác thực không chính xác hoặc đã hết hạn (đã bị vô hiệu hóa).



API sẽ báo lỗi khi mật khẩu mới không khớp.

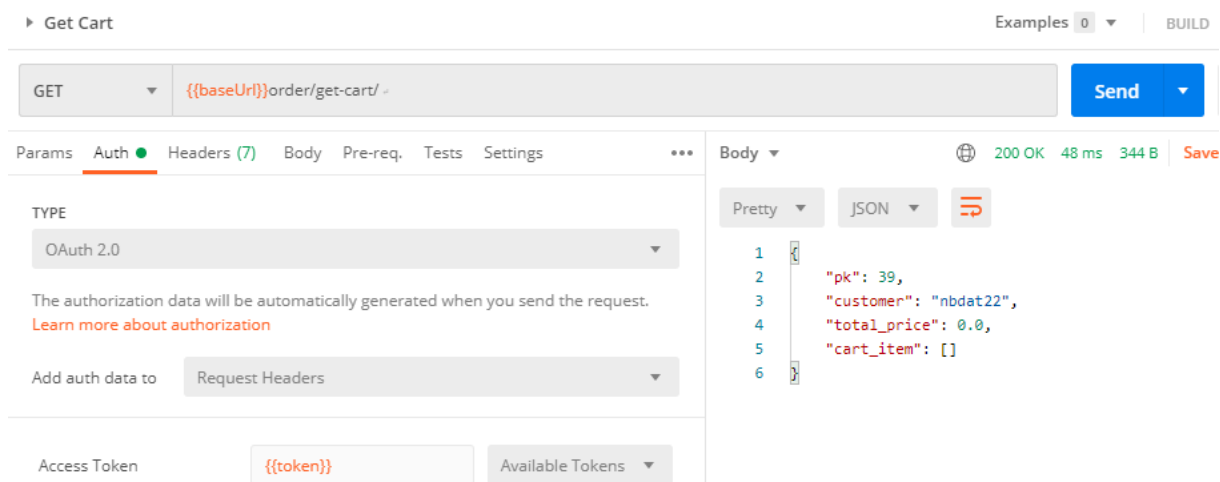


3.4.3. API Order

GET Get Cart

[Xác thực token]

API giúp khách hàng xem thông tin giỏ hàng hiện tại của mình.

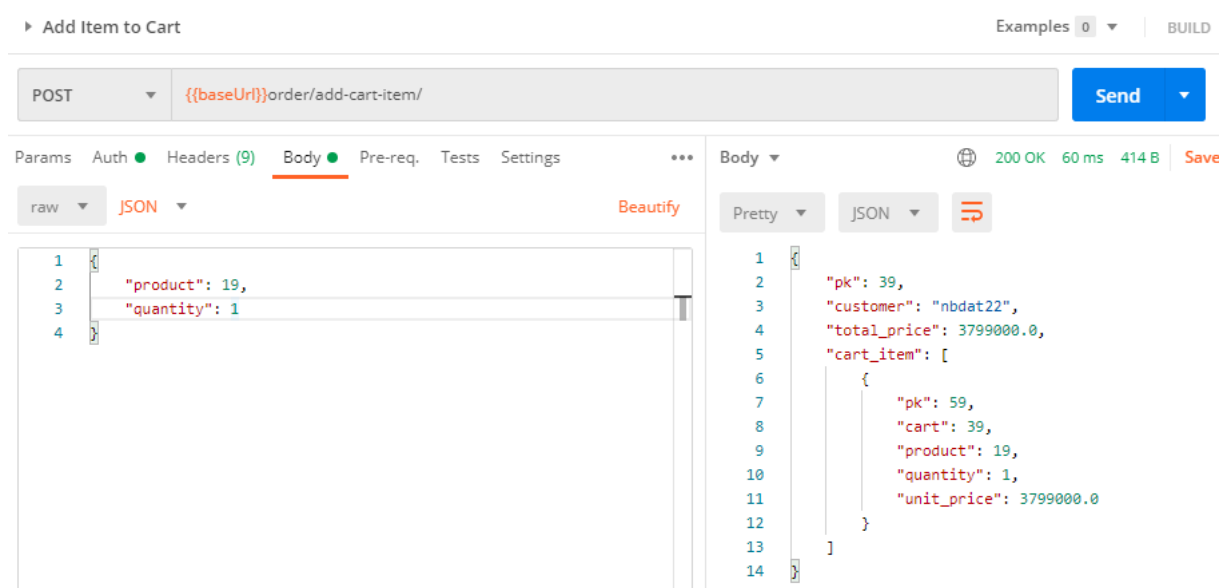


Hình 55. API Get Cart: Ok

POST Add Item to Cart

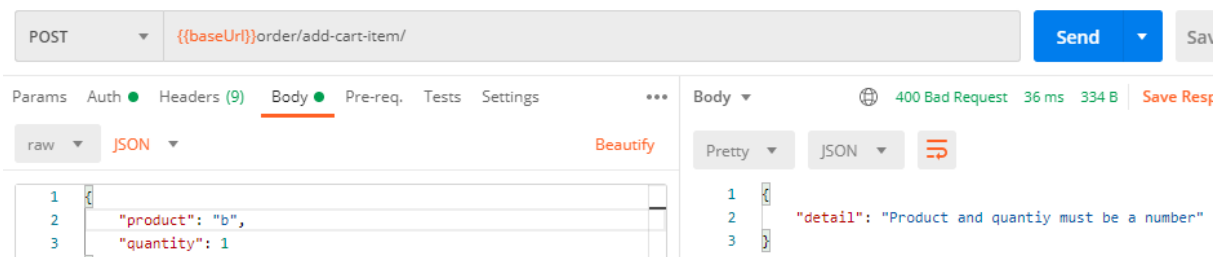
[Xác thực token]

API thêm sản phẩm vào giỏ hàng của khách hàng theo mã sản phẩm và số lượng cần thêm vào.

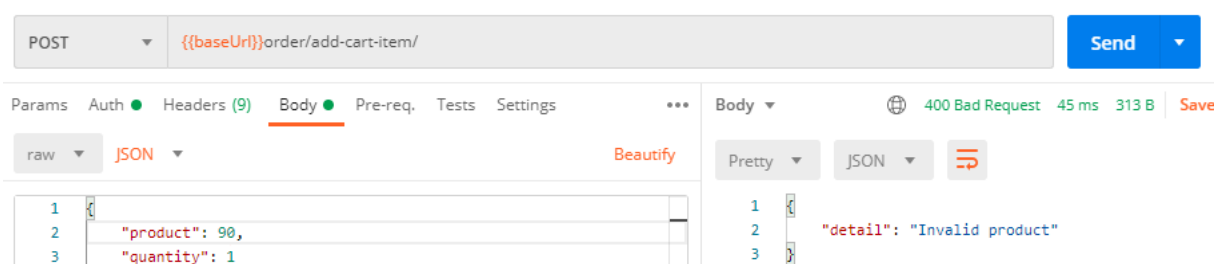


Hình 56. API Add Item to Cart: Ok

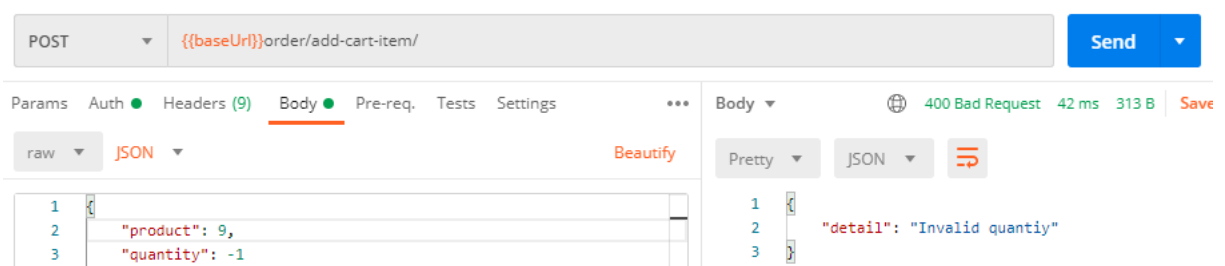
API sẽ báo lỗi khi dữ liệu gửi lên bị thiếu, mã sản phẩm không hợp lệ (Mã không tồn tại hoặc không phải số) hoặc số lượng không hợp lệ (Số lượng bé hơn 1 hoặc không phải là số).



Hình 57. API Add Item to Cart: Sai kiểu dữ liệu

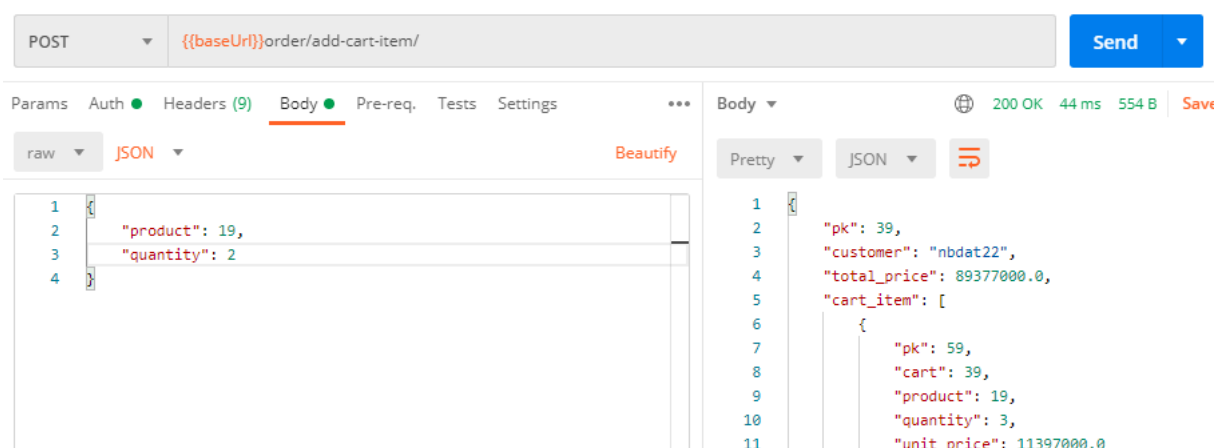


Hình 58. API Add Item to Cart: Mã sản phẩm không tồn tại



Hình 59. API Add Item to Cart: Số lượng không hợp lệ

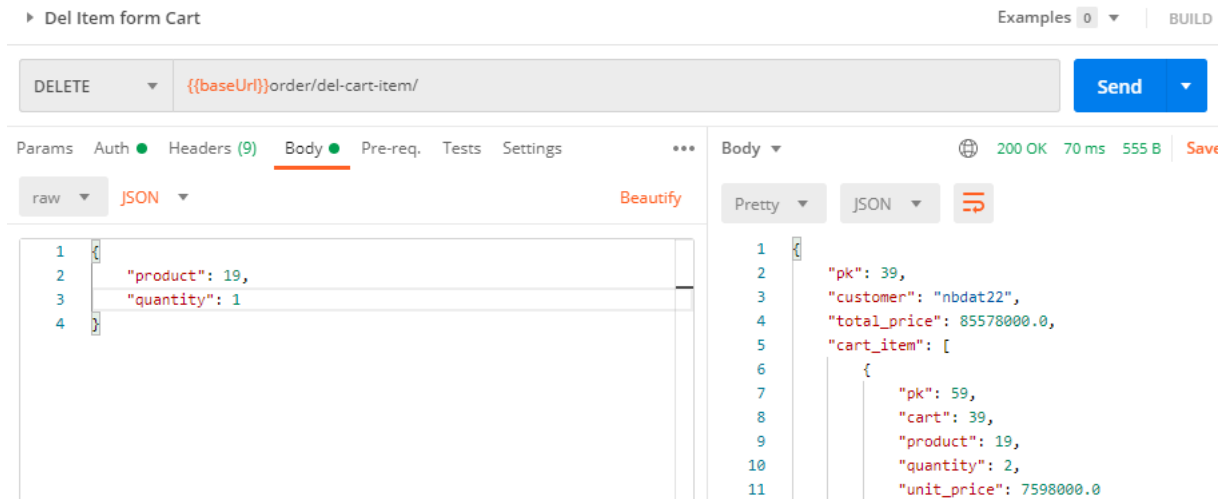
API sẽ tự động tăng số lượng nếu sản phẩm vừa thêm vào đã tồn tại trong giỏ.



Hình 60. API Add Item to Cart: Số lượng tăng tự động

DEL Del Item form Cart [Xác thực token]

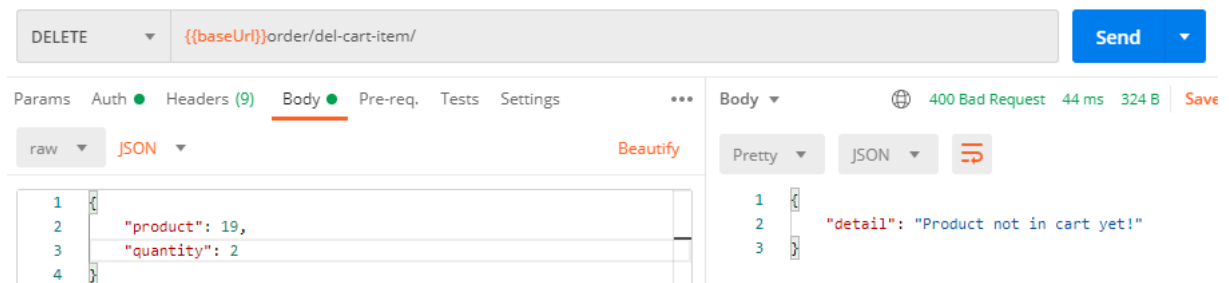
API xóa sản phẩm khỏi giỏ hàng của khách hàng theo mã sản phẩm và số lượng cần xóa.



Hình 61. API Del Item form Cart: Ok

API sẽ báo lỗi khi dữ liệu gửi lên bị thiếu, mã sản phẩm không hợp lệ (Mã không tồn tại hoặc không phải số) hoặc số lượng không hợp lệ (Số lượng bé hơn 1 hoặc không phải là số). (Tương tự như API Add Item to Cart ở trên)

API sẽ báo lỗi nếu sản phẩm muốn xóa không có trong giỏ.

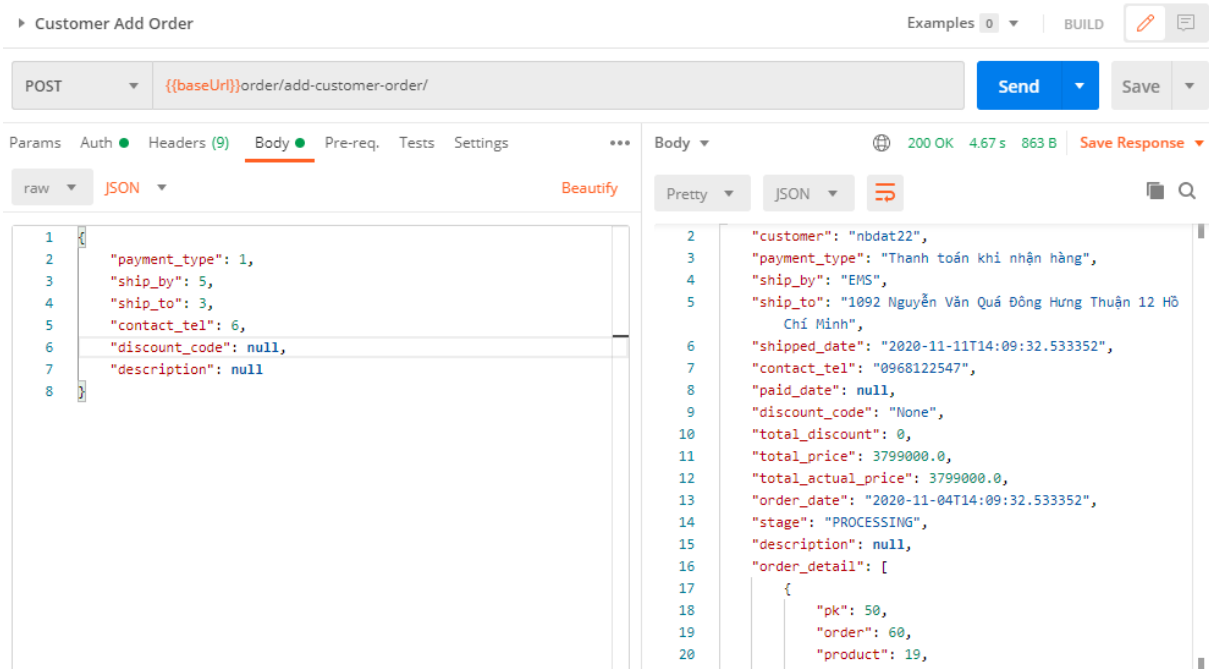


Hình 62. API Del Item form Cart: Sản phẩm không có trong giỏ

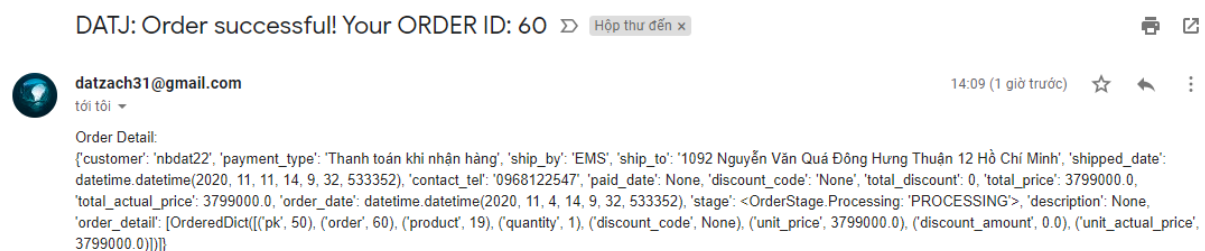
POST Customer Add Order

[Xác thực token]

API dùng để tạo đơn đặt hàng cho khách hàng và gửi mail thông báo đặt hàng thành công.



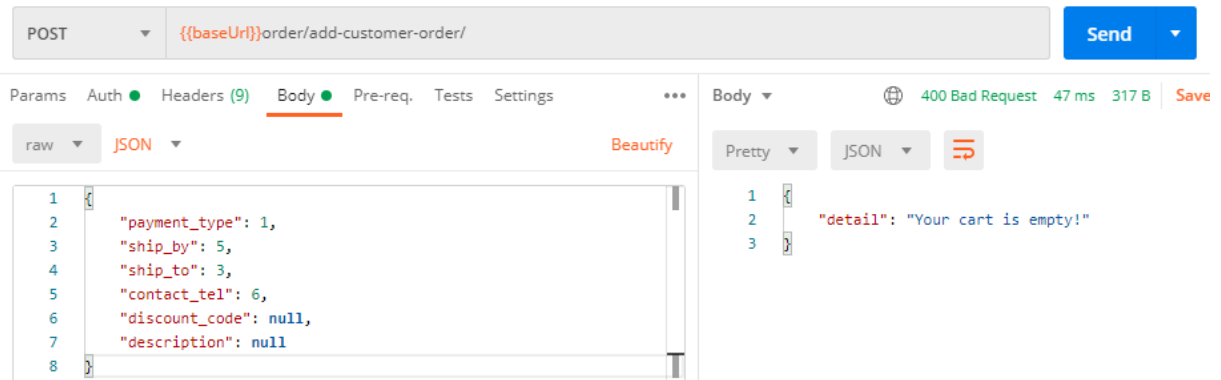
Hình 63. API Customer Add Order: Ok



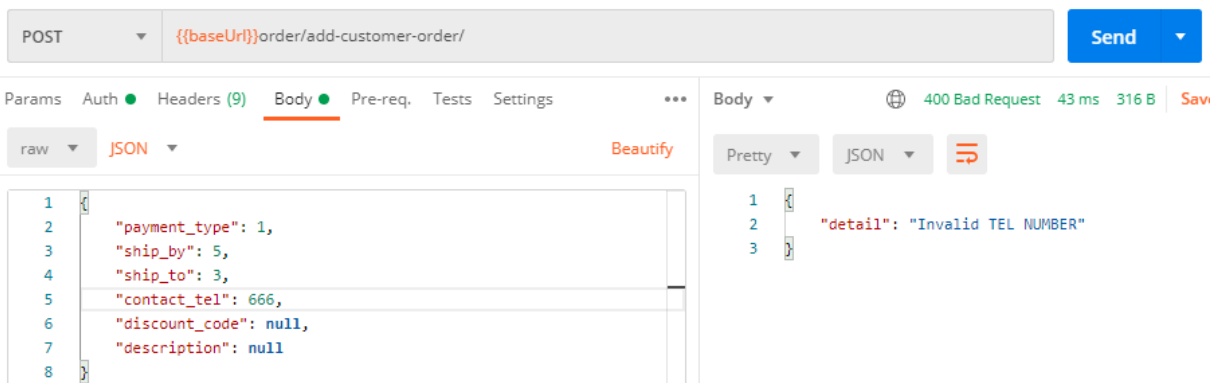
Hình 64. Mail thông báo KH đặt hàng thành công

Khách hàng đặt hàng thành công khi:

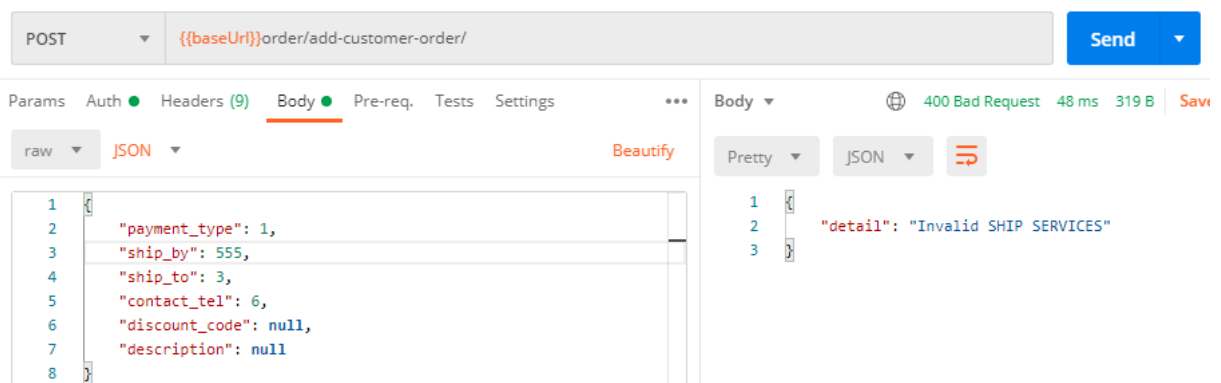
- Giỏ hàng không được phép rỗng.



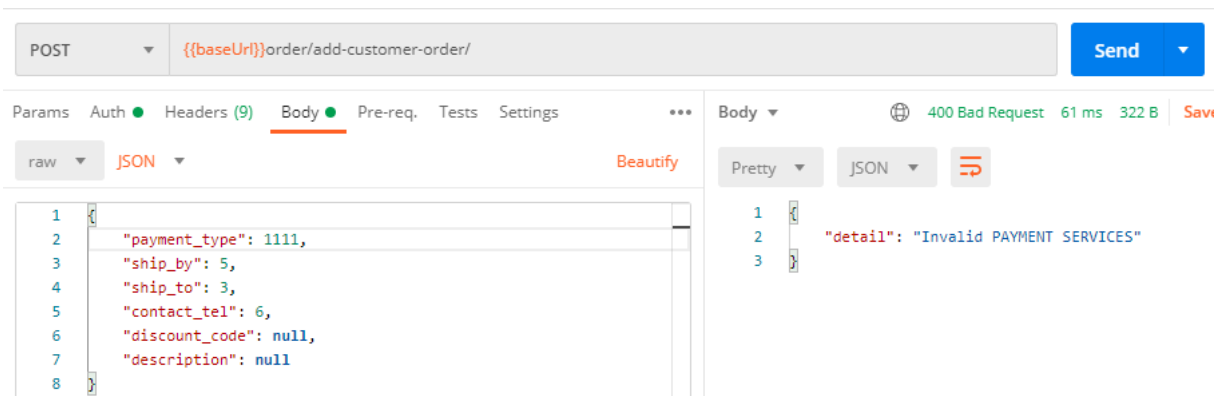
- Cung cấp đầy đủ thông tin để giao hàng (Địa chỉ giao hàng và điện thoại liên lạc) và thông tin giao hàng phải thuộc sở hữu của khách hàng đặt hàng.



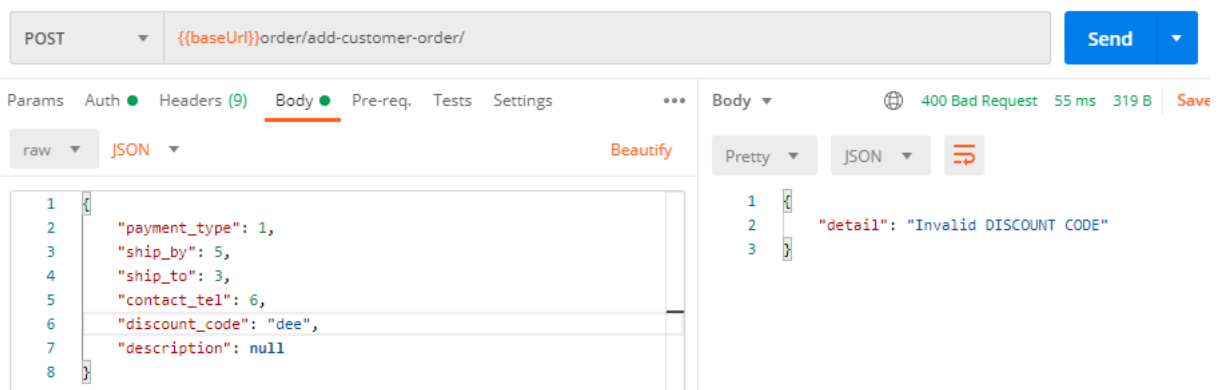
- Đã chọn được phương thức giao hàng mà hệ thống có hỗ trợ.



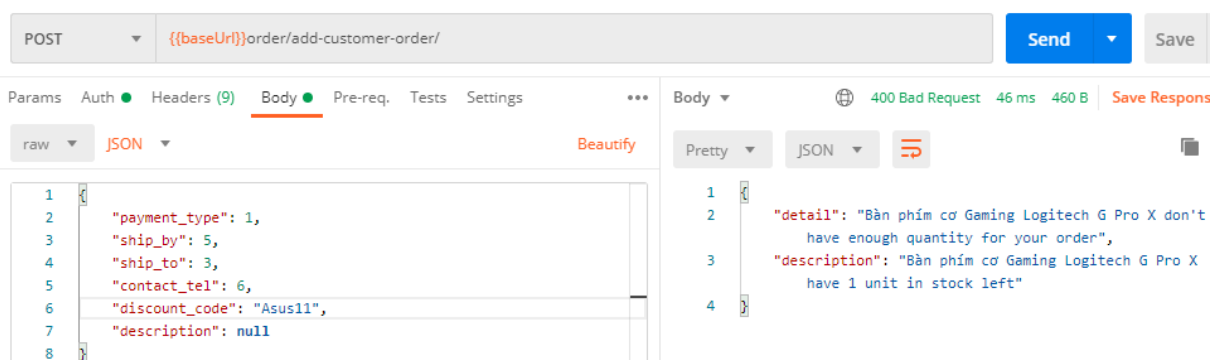
- Đã chọn được phương thức thanh toán mà hệ thống có hỗ trợ và phải hoàn tất thanh toán với các phương thức thanh toán trực tuyến.



- Nếu có mã giảm giá, phải nhập đúng mã giảm giá.



- Số lượng hàng tồn trong kho phải còn đủ số lượng để đáp yêu cầu đặt hàng.

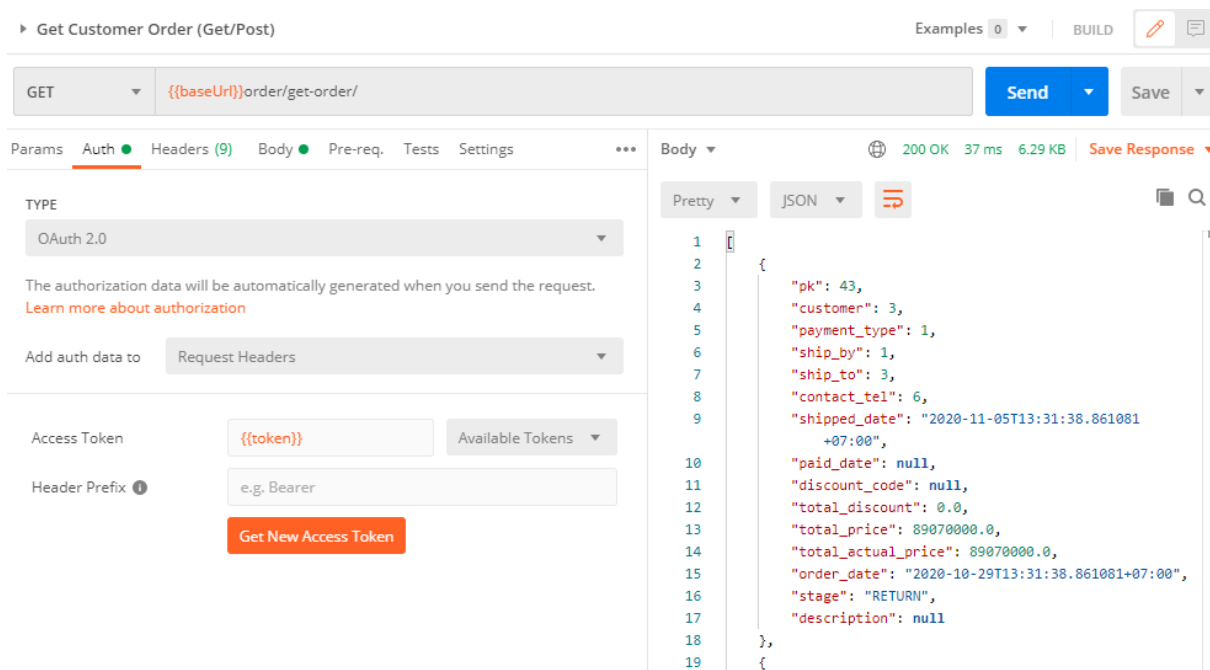


Sau khi kiểm tra các tính hợp lệ trên, API đã có thể tạo đơn hàng vào kho khi dữ liệu và các công việc để tạo đơn bao gồm:

- Tạo đơn hàng mới tại bảng Order với trạng thái PROCESSING cùng thông tin giảm giá nếu có.
- Đổ dữ liệu sản phẩm từ bảng CartItem sang bảng OrderDetail cùng với thông tin của mã giảm giá nếu có.
- Cập nhật số lượng unit_in_order tại bảng Product.
- Xóa dữ liệu tại bảng Cart và CartItem của khách hàng.
- Cập nhật và lưu lại tất cả thông tin.
- Gửi mail thông báo việc đặt hàng thành công cho khách hàng.

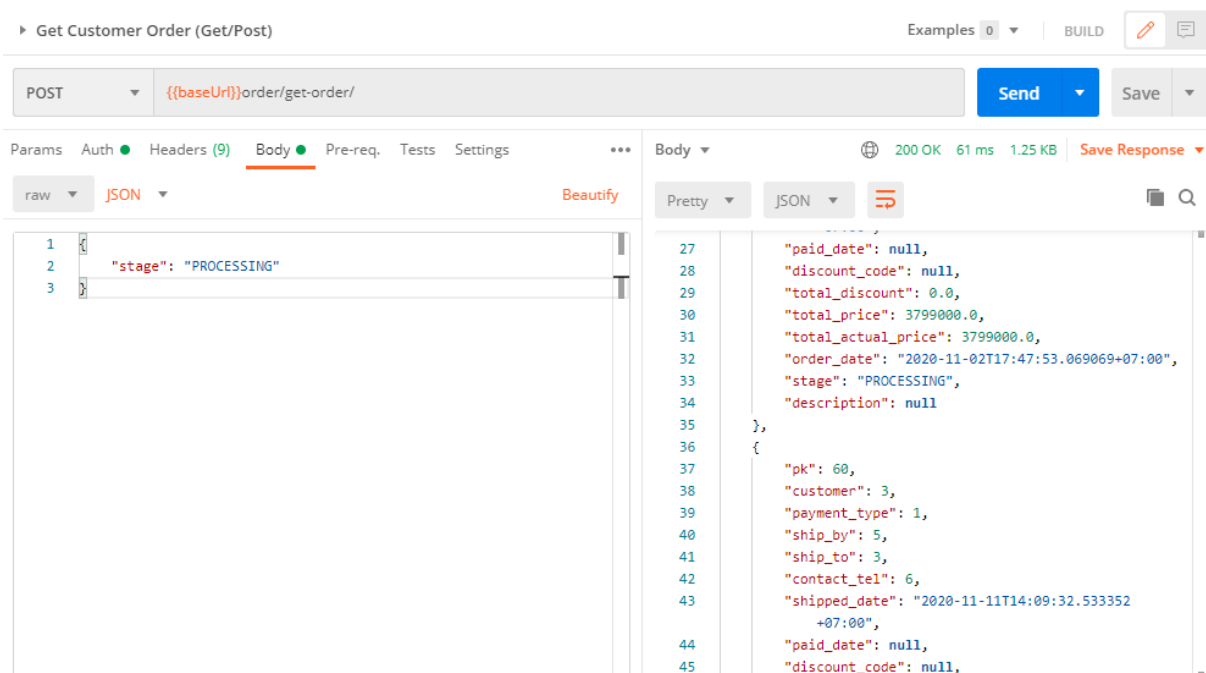
GET Get Customer Order (Get/Post) [Xác thực token]

API khi gọi bằng phương GET sẽ lấy danh sách tất cả đơn hàng của khách hàng.



Hình 65. API Get Customer Order (GET): Ok

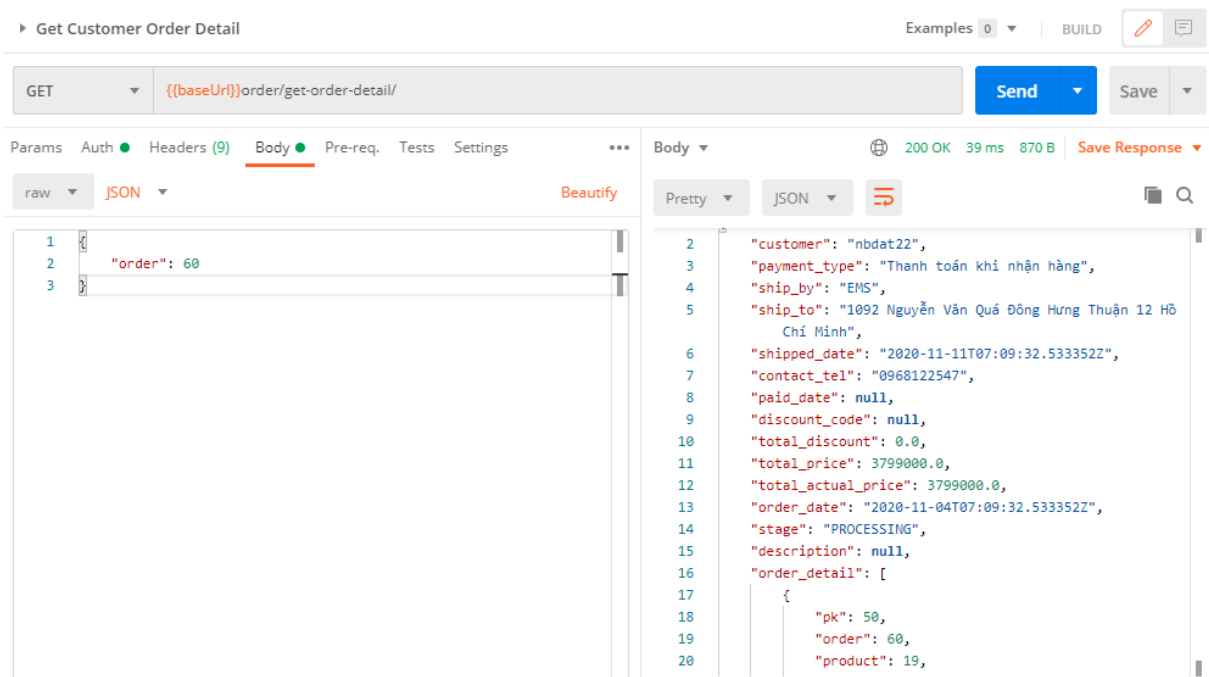
API khi gọi bằng phương thức POST sẽ lấy danh sách đơn hàng của khách hàng theo trạng thái được gửi đi.



Hình 66. API Get Customer Order (POST): DS đơn hàng có trạng thái PROCESSING

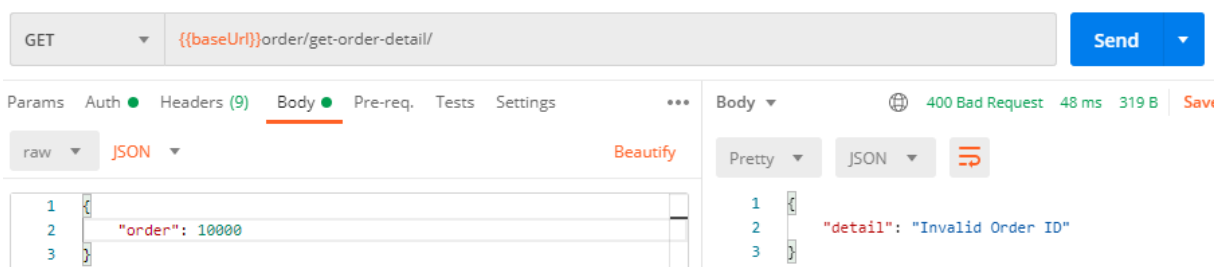
GET Get Customer Order Detail [Xác thực token]

API dùng để lấy thông tin chi tiết của đơn hàng theo mã đơn hàng.

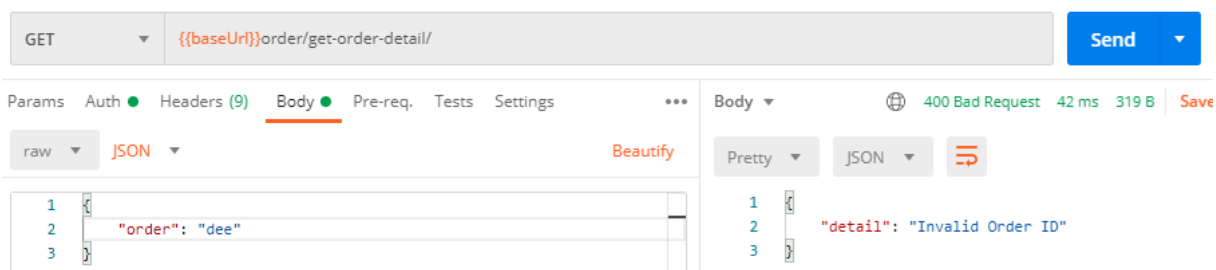


Hình 67. API Get Customer Order Detail: Ok

API sẽ báo lỗi khi nhận mã đơn hàng không hợp lệ (Mã đơn hàng không tồn hoặc mã không phải là số).



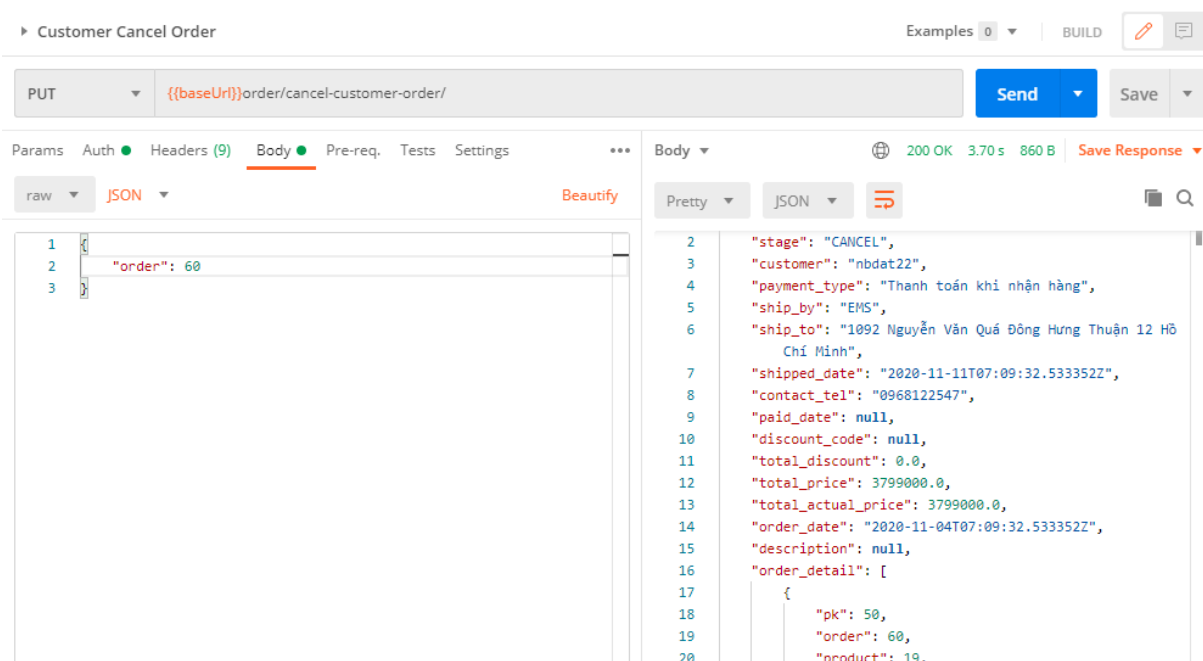
Hình 68. API Get Customer Order Detail: Mã đơn hàng không tồn tại



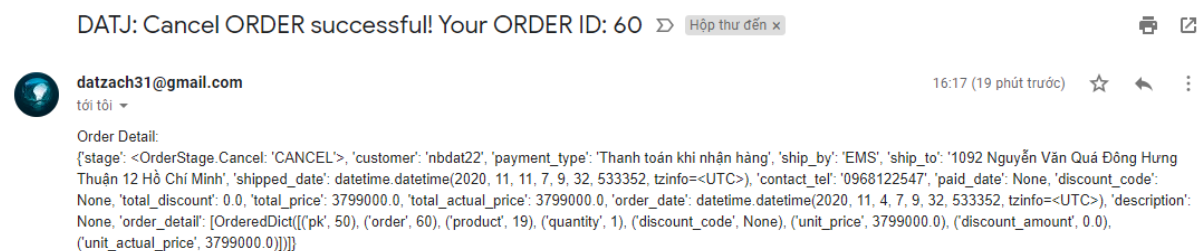
Hình 69. API Get Customer Order Detail: Mã đơn hàng không phải số

PUT Customer Cancel Order [Xác thực token]

API hỗ trợ khách hàng hủy đơn đặt hàng.



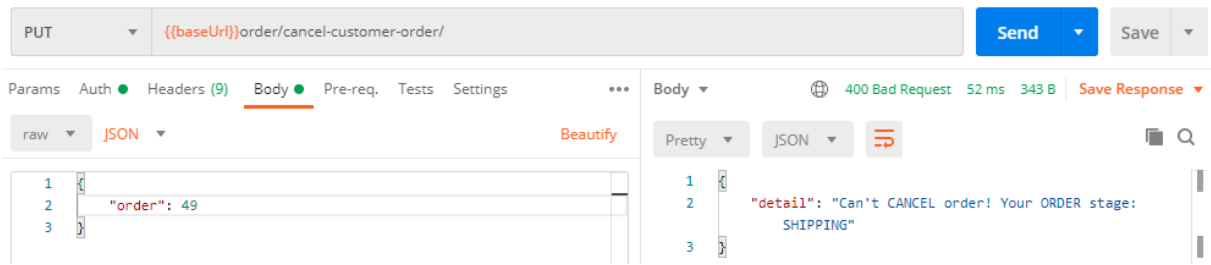
Hình 70. API Customer Cancel Order



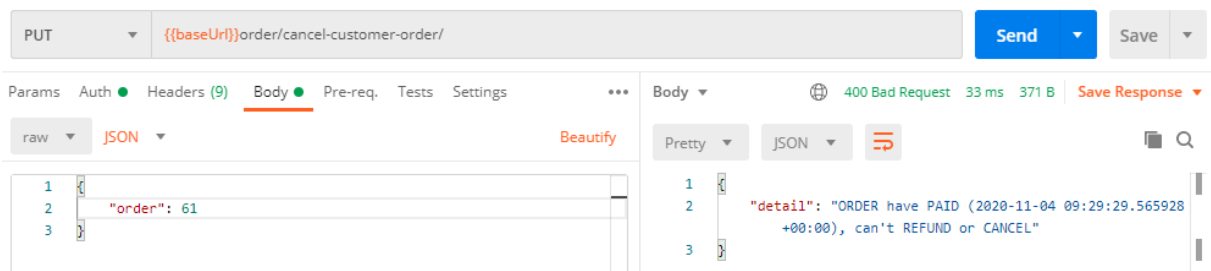
Hình 71. Mail thông báo KH hủy đơn thành công

Khách hàng chỉ được phép hủy đơn khi:

- Đơn hàng phải thuộc trạng thái PROCESSING (Đang xử lý).



- Đơn hàng chưa được thanh toán.



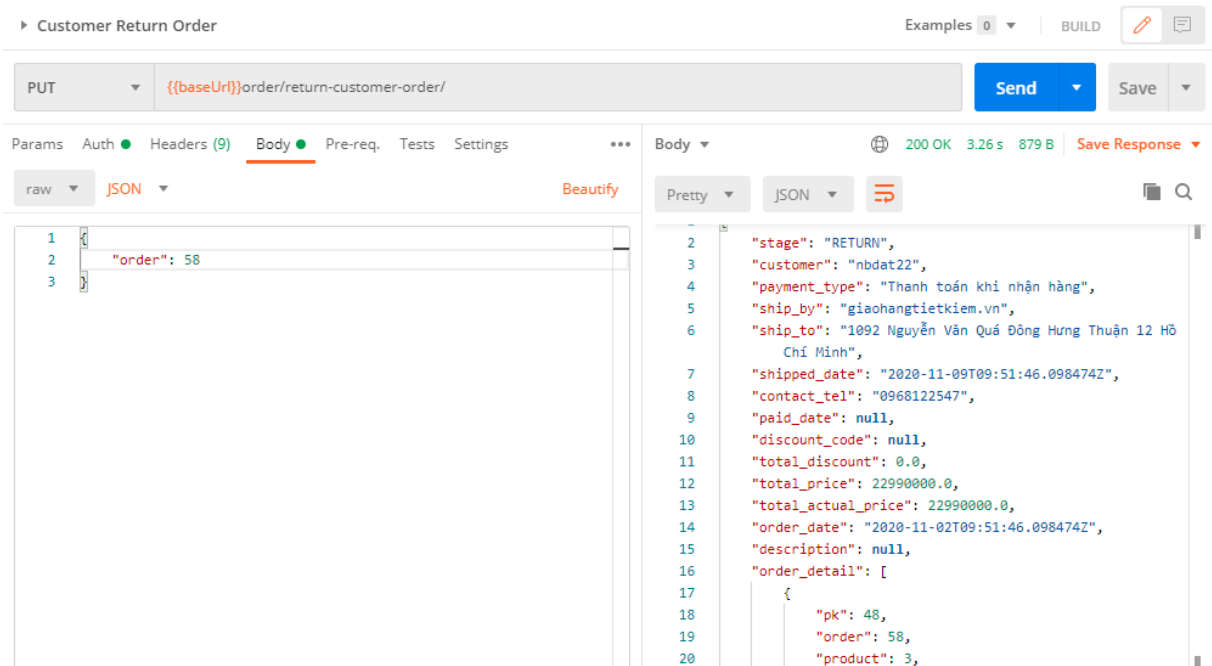
Sau khi kiểm tra đơn hàng được phép xóa, API sẽ thực thi tiếp các công việc sau để xóa đơn hàng:

- Đổi trạng thái đơn hàng từ PROCESSING sang CANCEL.
- Cập nhật lại số lượng `unit_in_order` tại bảng Product.
- Gửi mail thông báo việc hủy đơn thành công cho khách hàng.

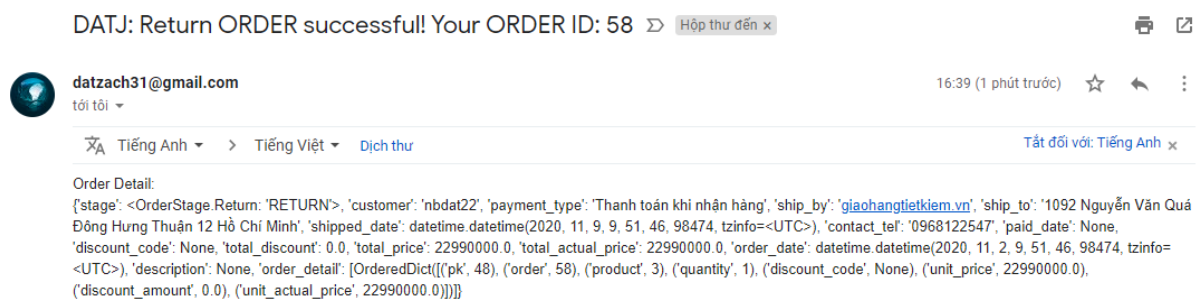
PUT Customer Return Order

[Xác thực token]

API hỗ trợ khách hàng trả hàng.

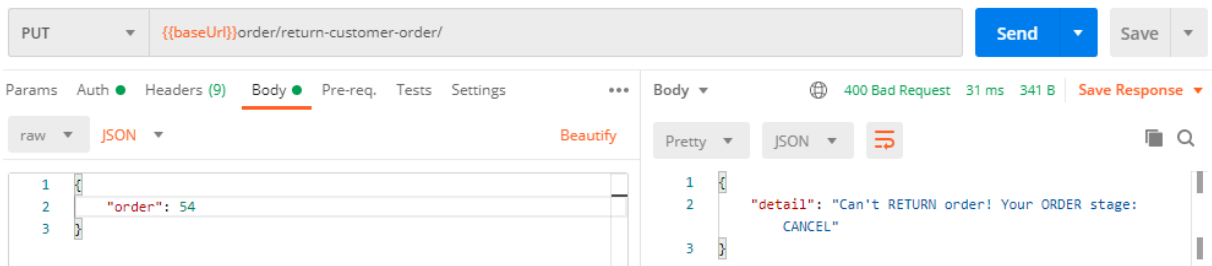


Hình 72. API Customer Return Order: Ok



Hình 73. Mail thông báo KH trả hàng thành công

Khách hàng chỉ được phép trả hàng khi đơn hàng ở trạng thái DONE (Đã hoàn tất).



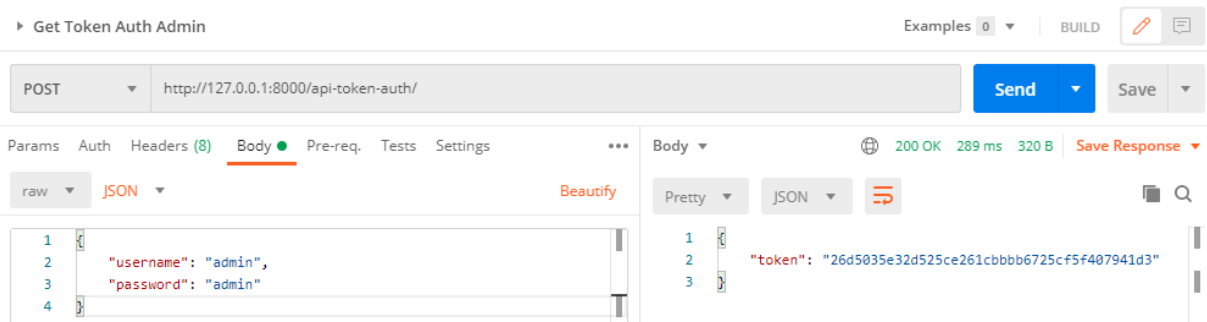
Sau khi API kiểm tra đơn hàng có thể trả về, API sẽ thực thi các công việc:

- Cập nhật trạng thái đơn hàng là RETURN và xóa thông tin ngày thanh toán.
- Cập nhật dữ liệu số lượng `unit_in_stock` tại bảng Product.
- Gửi mail thông báo việc trả hàng thành công cho khách hàng.

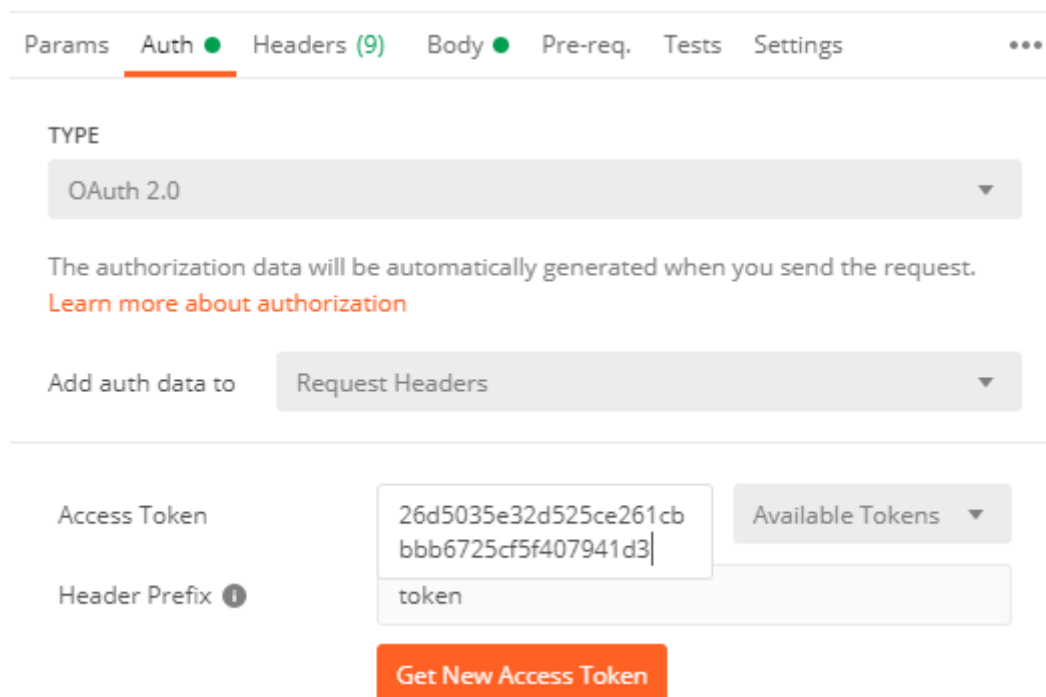
Các API tiếp theo dành cho nhân viên sử dụng hệ thống để cập nhật trạng thái của các đơn hàng và được xác thực token với thư viện

`rest_framework.authentication` của **Django**

Dùng API sau và đăng nhập với tài khoản nhân viên để tạo token:



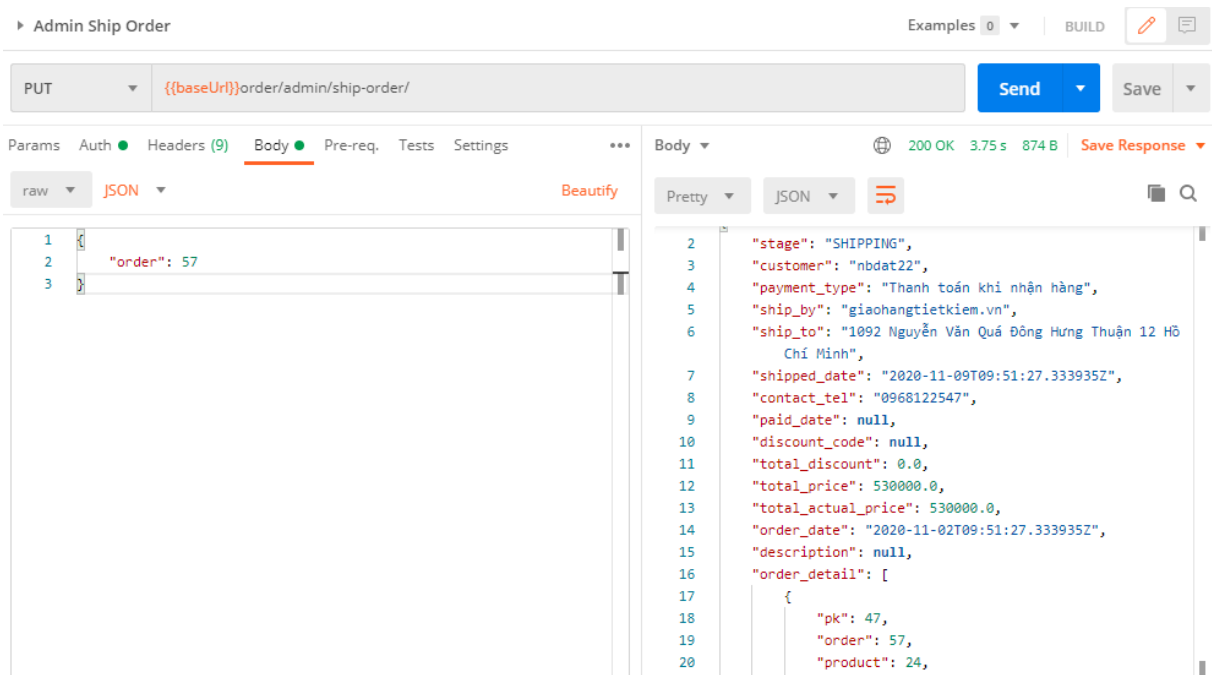
Dùng token đó để xác thực khi gọi API:



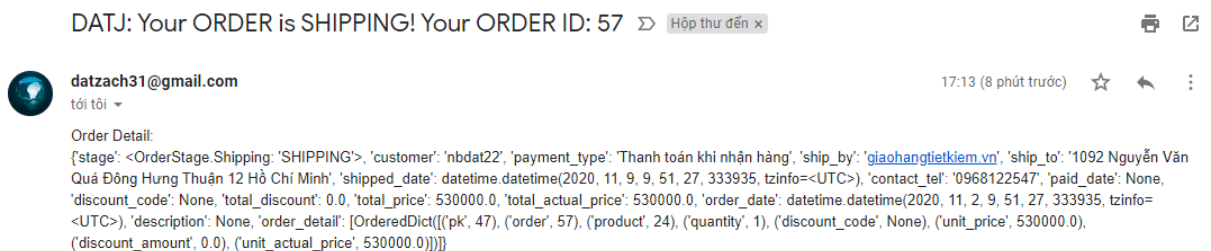
PUT Admin Ship Order

[Xác thực token nhân viên]

API giúp nhân viên cập nhật trạng thái đơn hàng thành SHIPPING.

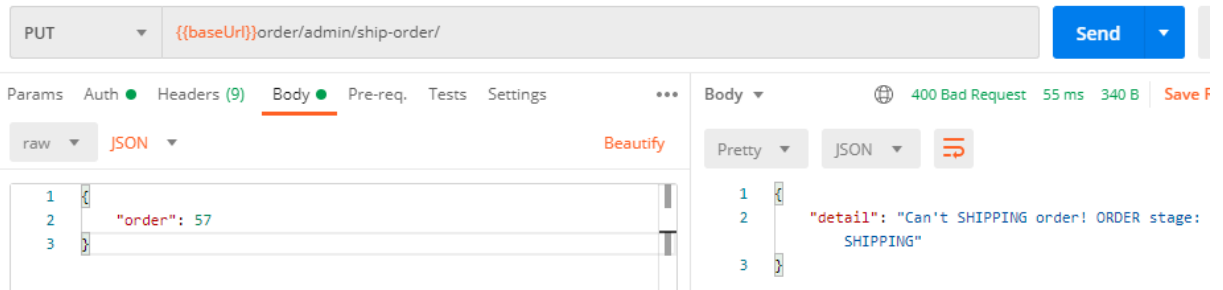


Hình 74. API Admin Ship Order: Ok



Hình 75. Mail thông báo KH đơn hàng đang được vận chuyển đi

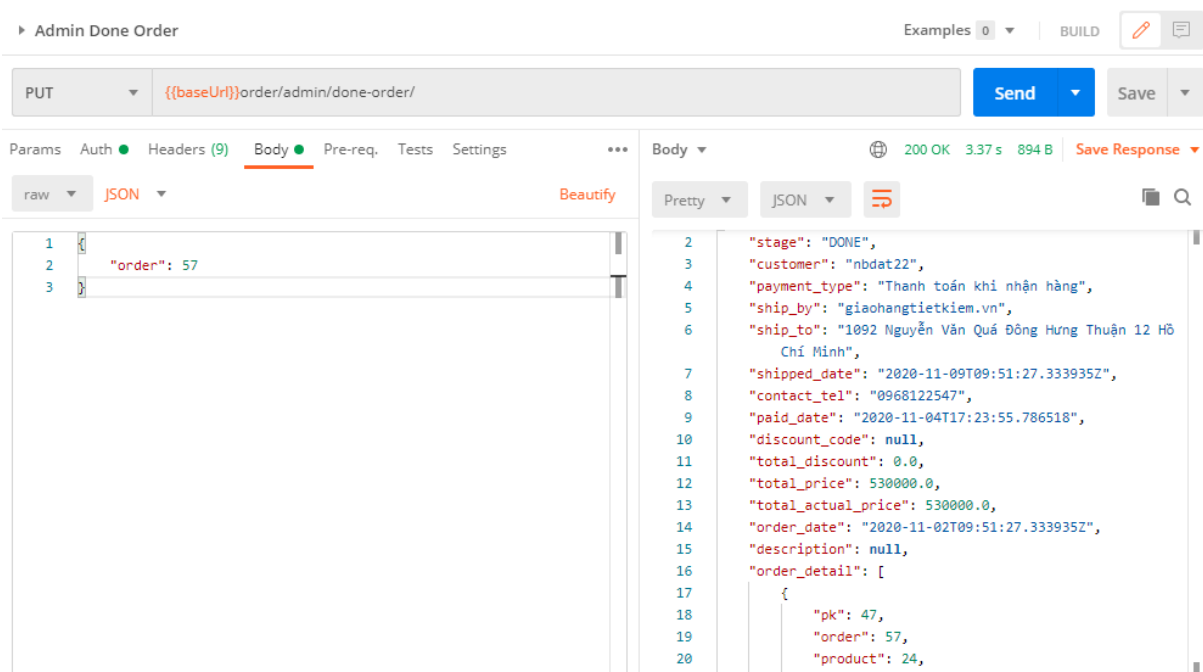
Đơn hàng chỉ được chuyển trạng thái giao hàng khi đơn hàng đang ở trạng thái PROCESSING.



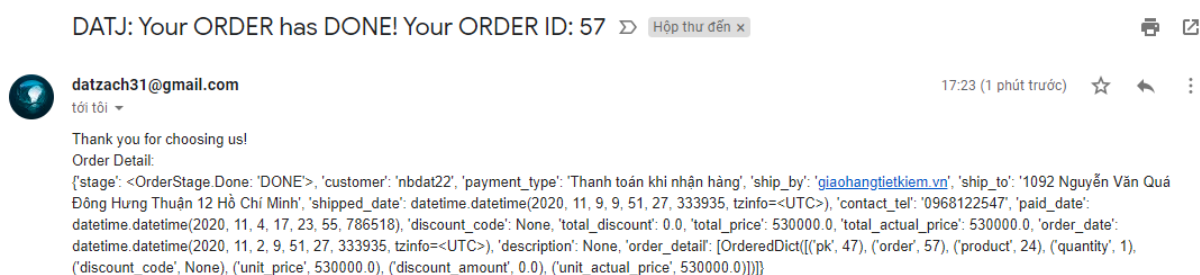
Sau khi API kiểm tra đơn hàng có thể giao, API sẽ thực thi các công việc:

- Cập nhật trạng thái đơn hàng từ PROCESSING sang SHIPPING.
- Cập nhật dữ liệu số lượng `unit_in_order` và `unit_in_stock` tại bảng Product.
- Gửi mail thông báo đơn hàng đang được giao cho khách hàng.

API giúp nhân viên cập nhật trạng thái đơn hàng thành DONE.

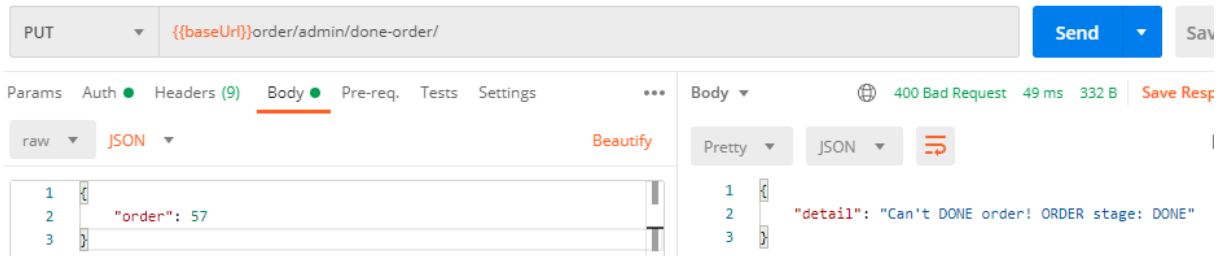


Hình 76. API Admin Done Order: Ok



Hình 77. Mail thông báo KH đơn hàng đã hoàn tất

Đơn hàng chỉ được chuyển trạng thái hoàn tất khi đơn hàng đang ở trạng thái SHIPPING.



Sau khi API kiểm tra đơn hàng có thể hoàn tất, API sẽ thực thi các công việc:

- Chuyển trạng thái đơn hàng từ SHIPPING sang DONE.
- Cập nhật ngày thanh toán hóa đơn nếu khách hàng thanh toán khi nhận hàng.
- Gửi mail thông báo đơn hàng hoàn tất cho khách hàng.

Chương 4. KẾT LUẬN

Sau khi hoàn thiện đồ án, em đã có được một hệ thống API được chia làm 3 nhóm API chính hỗ trợ khách hàng xem, chọn lựa hàng hóa (API Product), giúp khách hàng quản lý các thông tin tài khoản tại hệ thống (API Customer) và theo dõi các đơn hàng, đặt hàng, trả hàng và hủy đơn theo như cầu của họ. Cùng với đó là một web quản trị với các chức năng cơ bản dành cho chủ cửa hàng và các nhân viên với các thao tác quản lý hàng hóa, khuyến mãi, nhân sự và xem, kiểm kê các đơn hàng. Với các API cơ bản đó đã có thể ứng dụng vào một web bán hàng nhỏ, xử lý được các thao tác chọn mua hàng cho khách hàng. Và với cách thiết kế dữ liệu cho Product theo mô hình EAV em có thể dễ dàng mở rộng để bán được các mặt hàng khác về sau. Các API cũng được bảo mật bằng phương pháp OAuth 2, các API chỉ có thể gọi và thực thi được khi khách hàng đã đăng nhập hệ thống thành công và được cấp một token của riêng mình để thao tác với các API, chức năng khác của hệ thống. Cùng với tính năng gửi mail giúp khách hàng phục hồi mật khẩu dễ dàng và hệ thống thông báo kịp thời cho khách hàng về tình trạng các đơn hàng của mình.

Hệ thống của em còn nhiều mặt hạn chế bao gồm cách thiết kế dữ liệu của nhóm bảng Order (Order, OrderDetail, PaymentService, ShipService), việc trả hàng, giao các món hàng riêng lẻ trong đơn hàng chính, khó tích hợp được các dịch vụ bên thứ ba, chưa có các chức năng thống kê đơn hàng, hàng hóa và còn nhiều thiết sót khác em chưa thể nhận ra do trình độ còn hạn chế của bản thân. Mong nhận được sự góp ý, nhận xét, đánh giá từ quý thầy cô để em có thể nhận ra khuyết điểm và khắc phục những chỗ chưa hoàn thiện đó để phát triển tốt hơn trong tương lai.

Sau khi thực hiện đồ án và hoàn thiện phần báo cáo này, em đã có thêm được nhiều kiến thức chuyên môn mới cùng các kỹ năng tìm hiểu, nghiên cứu khi gặp các vấn đề mới, giải quyết vấn đề độc lập và quản lý phân chia thời gian, sắp xếp các công việc cần làm để thực hiện một dự án.

TÀI LIỆU THAM KHẢO

- [1] <https://viblo.asia/p/tim-hieu-ve-entity-attribute-value-pattern-eav-structural-pattern-Eb85okV452G>
 - Truy cập lần cuối vào “22/10/2020”
- [2] <https://www.entityframeworktutorial.net/code-first/what-is-code-first.aspx>
 - “03/11/2020”
- [3] <https://docs.djangoproject.com/en/3.1/ref/contrib/admin/>
 - Truy cập lần cuối vào “21/10/2020”
- [4] <https://www.django-rest-framework.org/tutorial/quickstart/>
 - Truy cập lần cuối vào “29/10/2020”
- [5] <https://www.django-rest-framework.org/api-guide/authentication/>
 - Truy cập lần cuối vào “02/11/2020”
- [6] <https://docs.djangoproject.com/en/3.1/topics/email/>
 - Truy cập lần cuối vào “03/11/2020”
- [7] <https://medium.com/eway/nguy%C3%AAn-t%E1%BA%AFC-thi%E1%BA%BFt-k%E1%BA%BF-rest-api-23add16968d7>
 - Truy cập lần cuối vào “04/11/2020”
- [8] <https://www.altexsoft.com/blog/engineering/what-is-api-definition-types-specifications-documentation/>
 - Truy cập lần cuối vào “06/11/2020”

---HẾT---