

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC – KỸ THUẬT MÁY TÍNH

LẬP TRÌNH WEB

BÀI TẬP LỚN SỐ 2

TP.Hồ Chí Minh, Tháng 12/2018

|  |  |
| --- | --- |
| GVHD: | Nguyễn Hữu Hiếu |
|  |  |
| Thành viên: | Lê Hữu Việt Anh - 1510054 |
|  | Mai Anh Đào - 1510625 |
|  | Nguyễn Thị Mỹ Duyên - 1510535  Bùi Phương Lan - 1511684 |

1. **Giới thiệu sản phẩm**
2. **Thiết kế hệ thống**
3. **Kiến trúc phần mềm: Sử dụng kiến trúc REST**

REST (Representational state transfer) định nghĩa các quy tắc kiến trúc để bạn thiết kế Web services chú trọng vào tài nguyên hệ thống, bao gồm các trạng thái tài nguyên được định dạng như thế nào và được chuyển tải qua HTTP thông qua số lượng lớn người dùng và được viết bởi những ngôn ngữ khác nhau. Nếu tính theo số dịch vụ mạng sử dụng, REST đã nổi lên trong vài năm qua như là một mô hình thiết kế dịch vụ chiếm ưu thế. Trong thực tế, REST đã có những ảnh hưởng lớn và gần như thay thế SOAP và WSDL vì nó đơn giản và dễ sử dụng hơn rất nhiều.

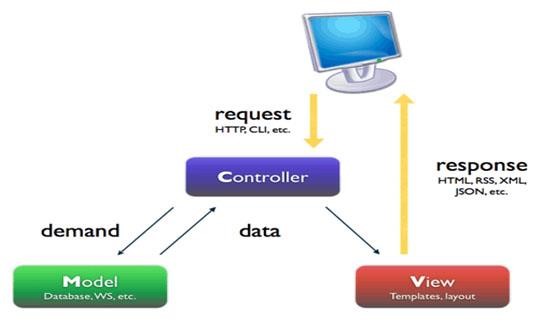
REST yêu cầu các nhà phát triển sử dụng phương thức HTTP một cách rõ ràng theo cách tương thích với giao thức chuẩn. Nguyên lý thiết kế REST cơ bản này thiết lập một ánh xạ 1-1 giữa các hành động tạo, đọc, cập nhật và xoá (CRUD) các quá trình vận hành và các phương thức HTTP. Theo cách ánh xạ này thì:

* Để tạo một tài nguyên trên máy chủ, bạn cần sử dụng phương thức POST.
* Để truy xuất một tài nguyên, sử dụng GET.
* Để thay đổi trạng thái một tài nguyên hoặc để cập nhật nó, sử dụng PUT.
* Để huỷ bỏ hoặc xoá một tài nguyên, sử dụng DELETE.

Những phản hồi (Response) trạng thái mà client có thể nhận được theo phương thức HTTP hoặc HTTPS:

* 200: OK - Yêu cầu thành công.
* 201: Created – Đã được tạo.
* 304: Not Modified – Dữ liệu chưa được định nghĩa.
* 400: Bad Request – Yêu cầu không đúng hoặc thiếu.
* 401: Unauthoried – Chưa xác thực.
* 403: Forbidden – Cấm truy cập dữ liệu.
* 404: Not Found – Không tìm thấy yêu cầu.
* 500: Internal Server Error – Lỗi máy chủ.

1. **Mô hình hệ thống: Sử dụng mô hình MVC (Model – View – Controller)**



*Hình 1. Mô hình MVC*

Mô hình MVC bao gồm 3 thành phần chính: Model, View và Controller.

* **Model:** Có nhiệm vụ thao tác với cơ sở dữ liệu, nghĩa là nó sẽ chứa tất cả các hàm, các phương thức truy vấn trực tiếp với dữ liệu và controller sẽ thông qua các hàm, phương thức đó để lấy dữ liệu rồi gửi qua View.
* **View:** Có nhiệm vụ tiếp nhận dữ liệu từ controller và hiển thị nội dung sang các đoạn mã HTML. Đây là thành phần giao diện.
* **Controller:** Đóng vài trò trung gian giữa Model và View. Nó có nhiệm vụ tiếp nhận yêu cầu từ client sau đó xử lý request, load model và gửi data qua view tương ứng rồi trả kết quả về cho client.

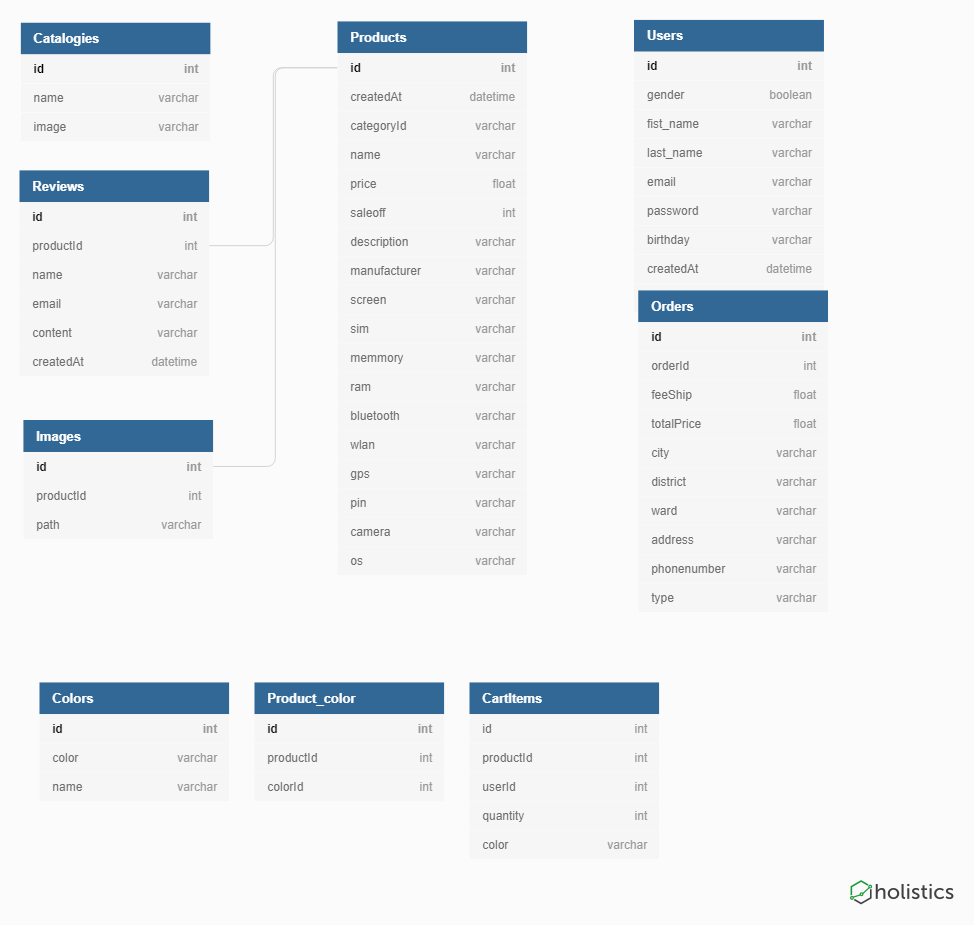
1. **Thiết kế cơ sở dữ liệu**

Sử dụng hệ quản trị cơ sở dữ liệu MySQL với các đối tượng như hình số 2 bên dưới.

Có các đối tượng là:

* Users: Lưu các tài khoản của người dùng bao gồm giới tính, họ, tên, email, ngày sinh, mật khẩu
* Categories: Lưu các danh mục sản phẩm của trang
* Reviews: Lưu các đánh giá của người dùng cho các sản phẩm
* Images: Lưu địa chỉ ảnh của các sản phẩm cũng như danh mục
* Products: Lưu các thông tin của sản phẩm
* Colors: Lưu các màu của sản phẩm
* CartItems: Lưu các sản phẩm khách hàng cho vào giỏ hàng
* Orders: Lưu các đơn hàng khách hàng đã đặt

Quan hệ giữa các đối tượng được thể hiện rõ trong lược đồ diagram bên dưới



*Hình 2. Lược đồ diagram database*

1. **Một số kỹ thuật được sử dụng**
2. **CORS (Cross-Origin Resource Sharing)**

Vì sử dụng kiến trúc RESTful API, nên việc truy cập API từ một host khác phải được sự cho phép của server. Vì vậy phải xét headers tương tự như code bên dưới:

header("Access-Control-Allow-Origin: \*");

header("Content-Type: application/json; charset=UTF-8");

header('Access-Control-Allow-Methods: DELETE');

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

1. **Hash mật khẩu**

Sử dụng MD5 để hash mật khẩu trước khi lưu vào database.

1. **Xác thực bằng JWT (JSON Web Token)**

JSON Web Token (JWT) là 1 tiêu chuẩn mở [(RFC 7519)](https://tools.ietf.org/html/rfc7519) định nghĩa cách thức truyền tin an toàn giữa các thành viên bằng 1 đối tượng JSON. Thông tin này có thể được xác thực và đánh dấu tin cậy nhờ vào "chữ ký" của nó. Phần chữ ký của JWT sẽ được mã hóa lại bằng [**HMAC**](https://en.wikipedia.org/wiki/Hash-based_message_authentication_code) hoặc [**RSA**.](https://en.wikipedia.org/wiki/RSA_(cryptosystem))

Những đặc điểm nổi bật của JWT:

* **Kích thước nhỏ**: JWT có thể được truyền thông qua URL, hoặc qua giao thức

POST, hay nhét vào bên trong phần HTTP Header. Kích thước nhỏ hơn ứng với công việc truyền tải sẽ nhanh hơn.

* **Khép kín**: Phần Payload (khối hàng) chứa toàn bộ những thông tin mà chúng ta cần tới, ví dụ như thông tin của người dùng (thay vì phải truy vấn cơ sở dữ liệu nhiều lần).

Dưới đây là 1 vài cách áp dụng với JWT:

* **Authentication**: Đây là kịch bản phổ biến nhất cho việc sử dụng JWT. Một khi người dùng đã đăng nhập vào hệ thống thì những request tiếp theo từ phía người dùng sẽ chứa thêm mã JWT, cho phép người dùng quyền truy cập vào các đường dẫn, dịch vụ, và tài nguyên mà cần phải có sự cho phép nếu có mã

Token đó. Phương pháp này không bị ảnh hưởng bởi Cross-Origin Resource Sharing (CORS) do nó không sử dụng cookie.

* **Trao đổi thông tin**: JSON Web Token là 1 cách thức để truyền tin an toàn giữa các thành viên với nhau, nhờ vào phần "chữ ký" của nó. Người nhận có thể biết được người gửi là ai thông qua phần chữ ký. Ngoài ra, chữ ký được tạo ra bằng việc kết hợp cả phần header và payload lại nên ta có thể xác nhận được chữ ký có bị giả mạo hay không.

Cấu trúc của JSON Web Token:

JSON Web Token bao gồm 3 phần, được ngăn cách nhau bởi dấu chấm (.):

* Header
* Payload
* Signature (chữ ký)

Tổng quát thì nó có dạng như sau: xxxxx.yyyyy.zzzzz

**Header**: Phần Header dùng để khai báo kiểu chữ ký và thuật toán mã hóa sẽ dùng cho cái token của chúng ta.

**Payload (Claims)**: Nơi chứa các nội dung của thông tin (claim). Thông tin truyền đi có thể là mô tả của 1 thực thể (ví dụ như người dùng) hoặc cũng có thể là các thông tin bổ sung thêm cho phần Header. Nhìn chung, chúng được chia làm 3 loại: reserved, public và private.

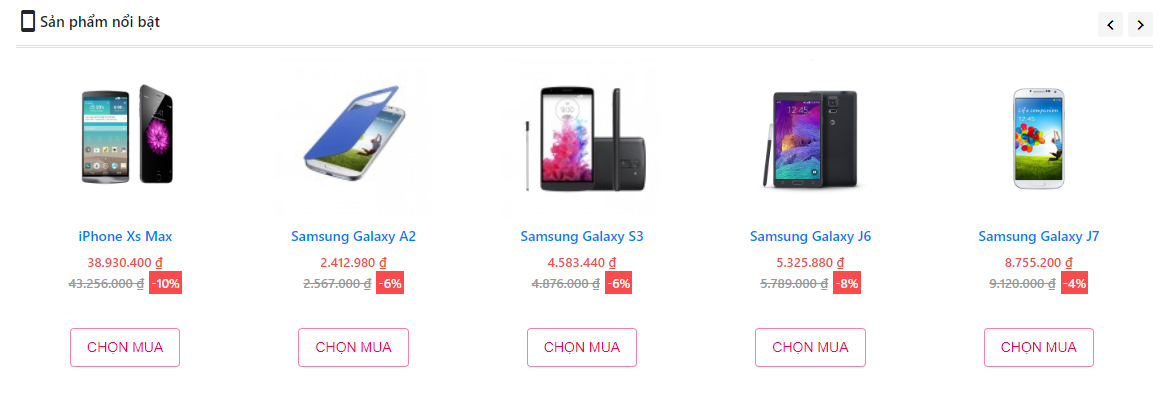
* Reserved: là những thông tin đã được quy định ở trong [IANA JSON Web Token Claims registry.](https://tools.ietf.org/html/draft-ietf-oauth-json-web-token-32#section-10.1) Các khóa của claim đều chỉ dài 3 ký tự vì mục đích giảm kích thước của Token. Chúng bao gồm:
* iss (issuer): tổ chức phát hành token.
* sub (subject): chủ đề của token.
* aud (audience): đối tượng sử dụng token.
* exp (expired time): thời điểm token sẽ hết hạn.
* nbf (not before time): token sẽ chưa hợp lệ trước thời điểm này.
* iat (issued at): thời điểm token được phát hành, tính theo UNIX time.
* jti: JWT ID
* Public: Khóa nên được quy định ở trong IANA JSON Web Token Registry hoặc là 1 URI có chứa không gian tên không bị trùng lặp.
* Private: Phần thông tin thêm dùng để truyền qua giữa các máy thành viên.

**Signature**:

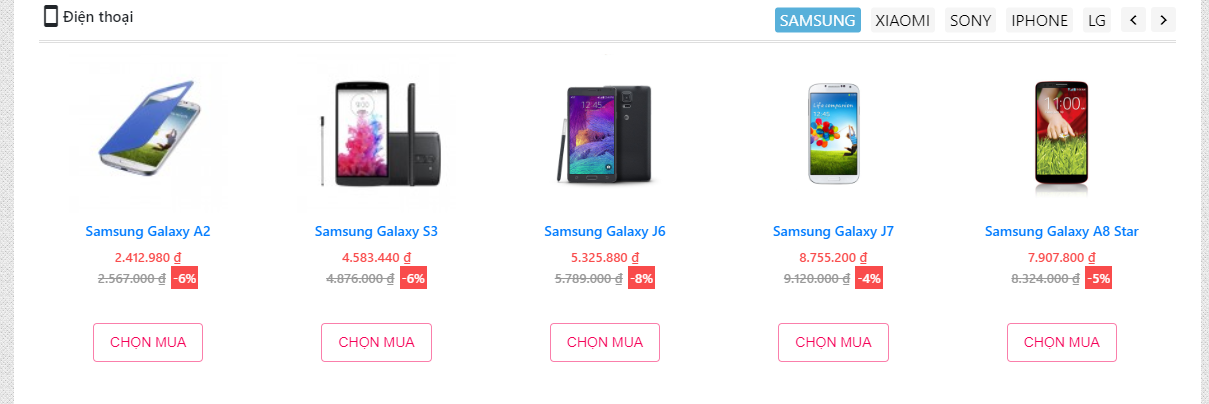
Phần chữ ký được tạo bằng cách kết hợp 2 phần Header + Payload, rồi mã hóa nó lại bằng 1 giải thuật encode, càng phức tạp thì càng tốt, ví dụ như HMAC SHA-256.

1. **Các chức năng chính**
2. Hiển thị top sản phẩm

Các sản phẩm nổi bật được chọn (hiện tại đang lấy theo những sản phẩm mới nhất)



1. Hiển thị top sản phẩm theo danh mục

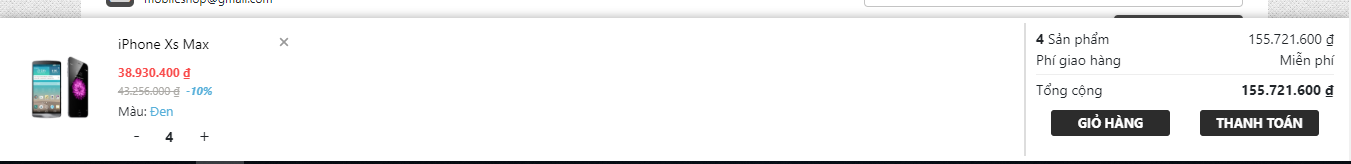
Tương tự như sản phẩm nổi bật, nhưng khách hàng có thể lọc theo danh mục.

1. Chọn mua nhanh

Khách hàng có thể xem thông tin nhanh của sản phẩm qua các card nhỏ và chọn mua mà không cần phải vào trang chi tiết

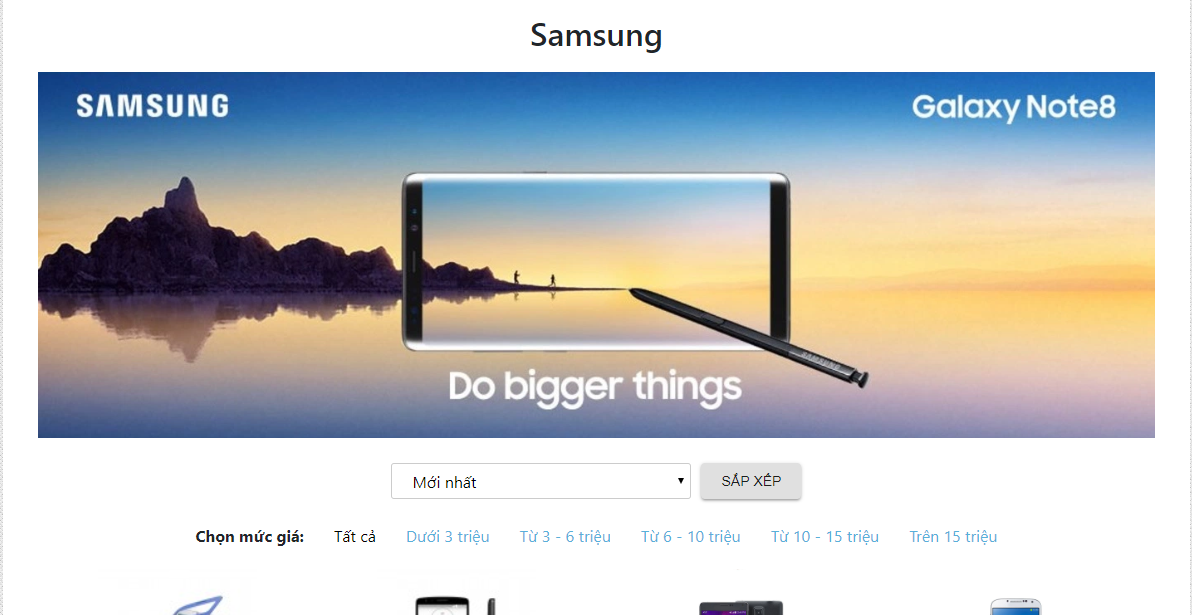
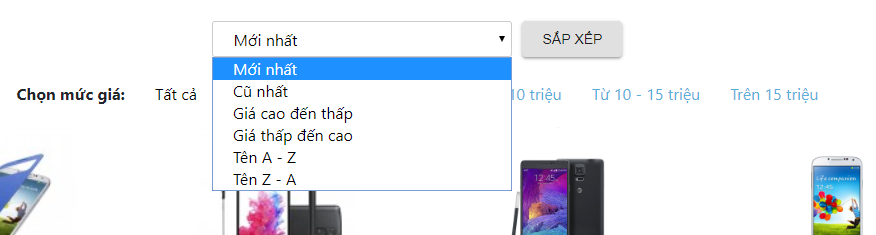


1. Xem giỏ hàng nhanh

Sau khi chọn mua, khách hàng có thể xem giỏ hàng của mình ngay tại trang hiện tại, đồng thời có thể tăng giảm số lượng.

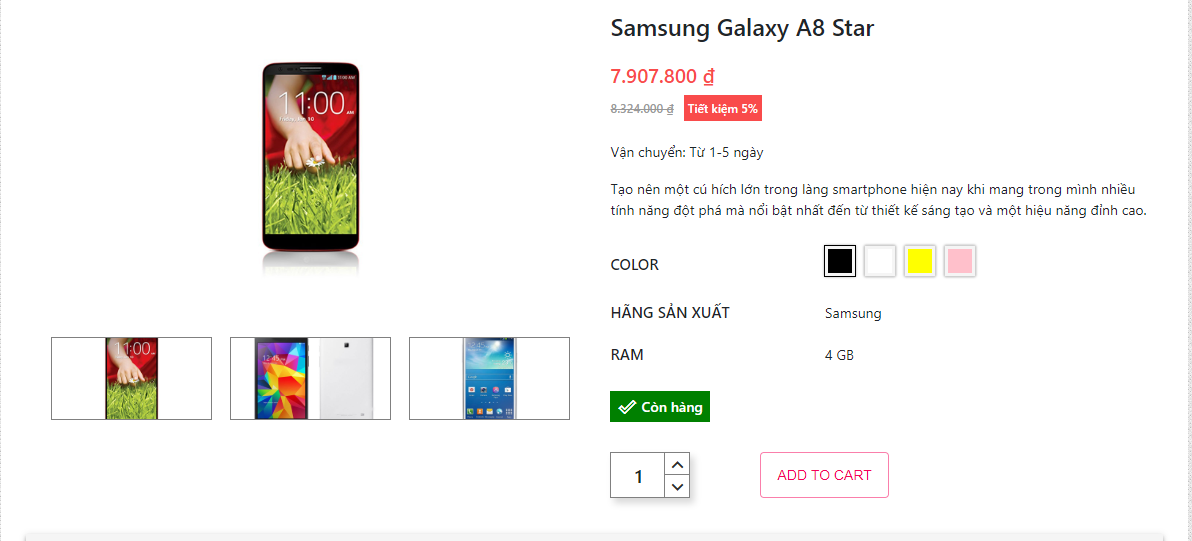
1. Xem sản phẩm của từng danh mục, có chức năng phân loại theo giá

Sau khi khách hàng chọn danh mục trên menu, các sản phẩm của danh mục đó sẽ được hiển thị, khách hàng có thể bấm từng khoảng giá hoặc để phân loại, cũng như chọn cách sắp xếp và có thể xem những trang tiếp nếu có nhiều sản phẩm.

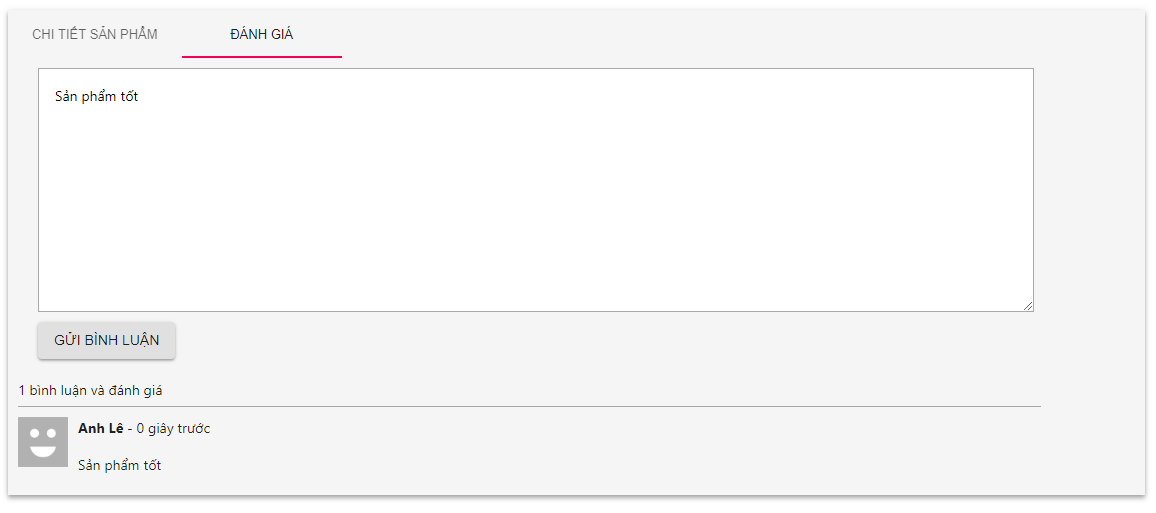


1. Xem trang chi tiết của sản phẩm

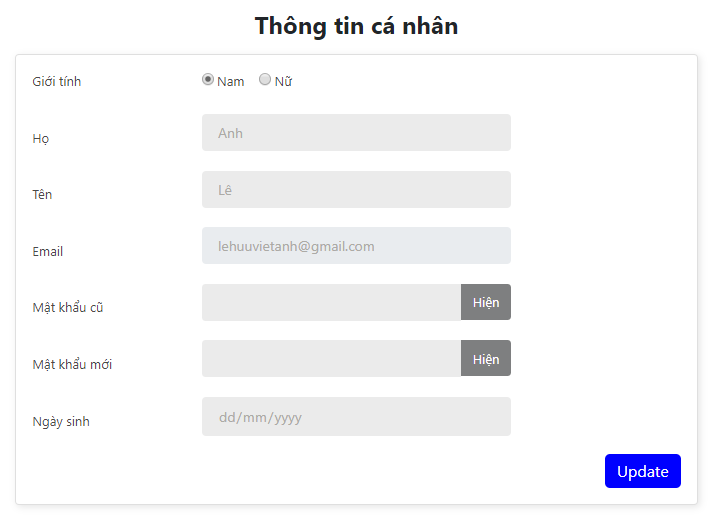
Khách hàng có thể bấm vào từng sản phẩm để hiển thị chi tiết thông tin sản phẩm đó, có thể chọn mua trong này.



1. Xem và đánh giá

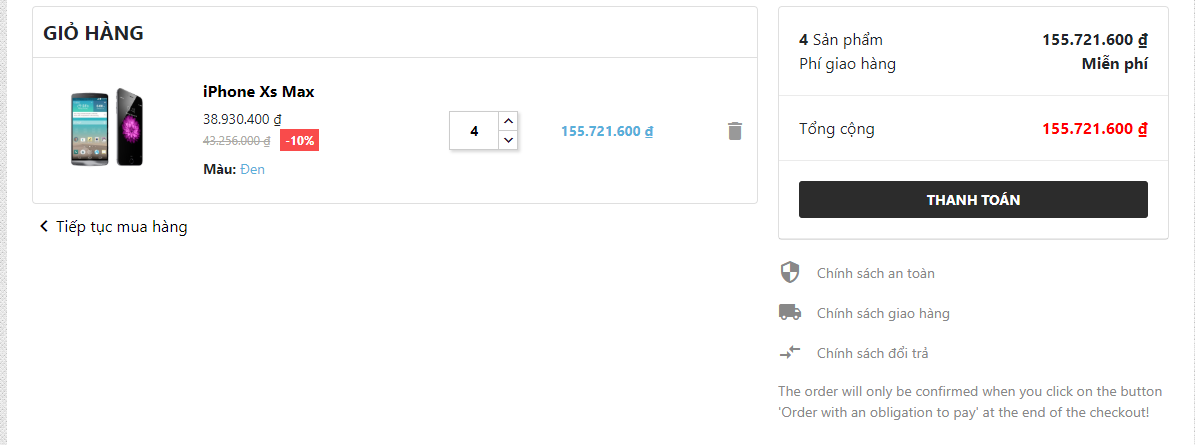
Khách hàng có thể xem và đánh giá sản phẩm dù đã đăng nhập hoặc chưa. Nếu chưa đăng nhập thì sẽ phải nhập họ tên và email.

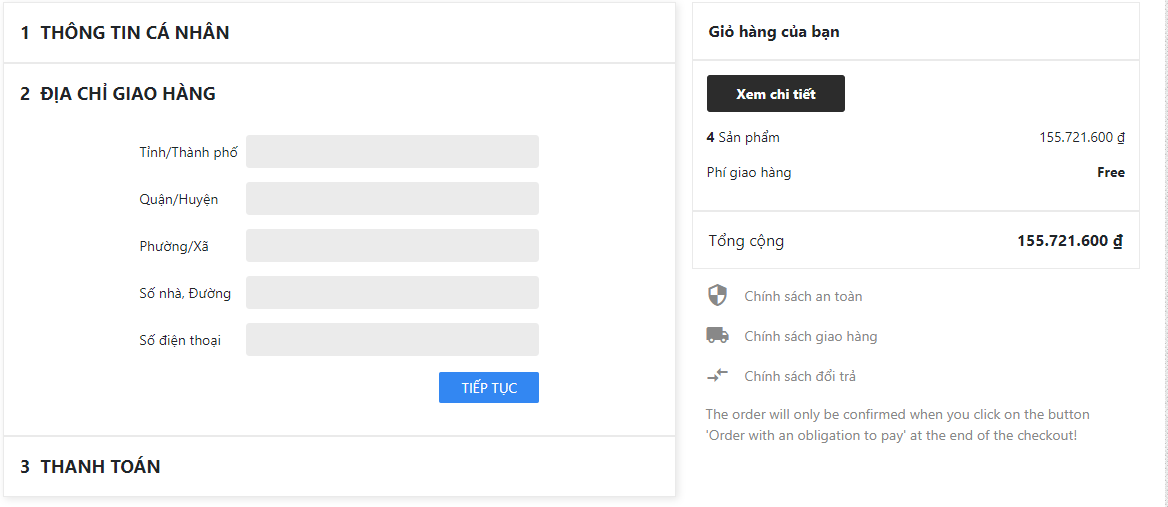
1. Xem thông tin các nhân và chỉnh sửa

Khách hàng có thể xem thông tin cá nhân của mình nếu đã đăng nhập, cũng như chỉnh sửa thông tin và mật khẩu.

1. Xem chi tiết giỏ hàng

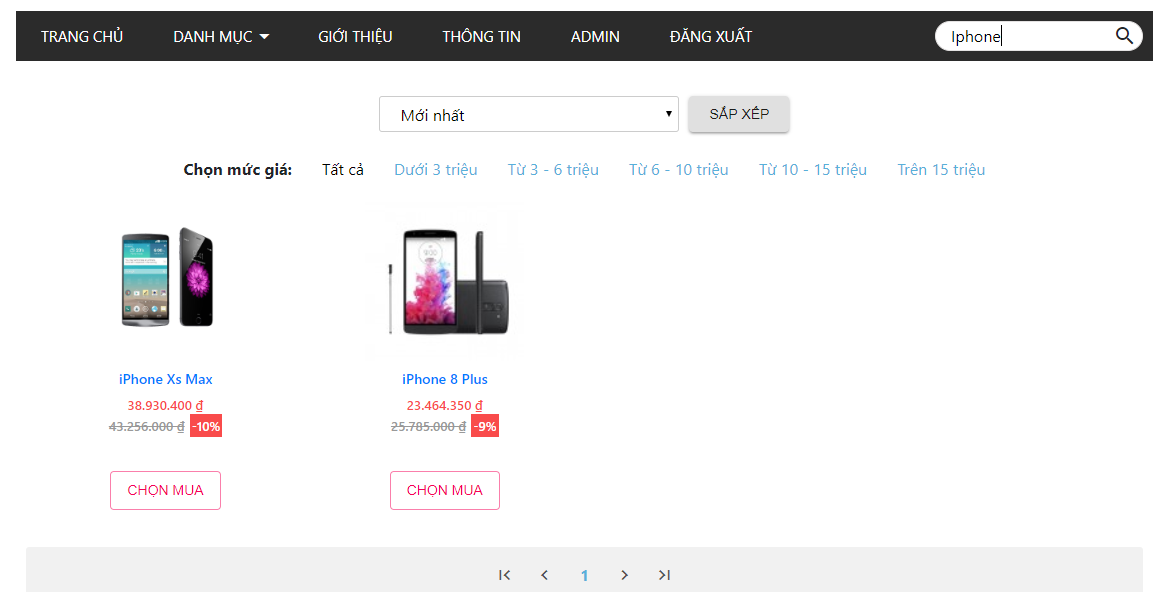
Khách hàng có thể xem chi tiết giỏ hàng, từ đây có thể tiếp mục mua hoặc chuyển sang thanh toán.

1.  Thanh toán

Điền các thông tin và chọn phương thức đặt hàng, thanh toán.

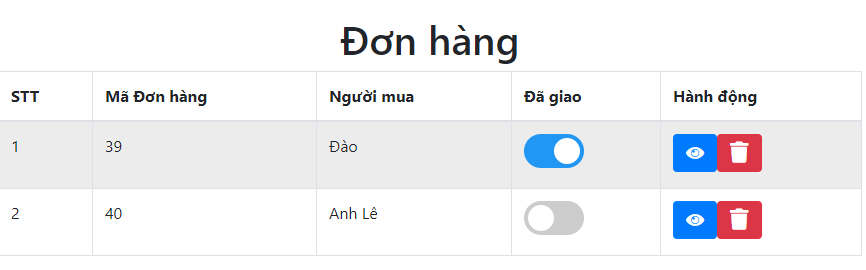
1. Tìm kiếm sản phẩm

Khách hàng có thể tìm kiếm sản phẩm theo tên.



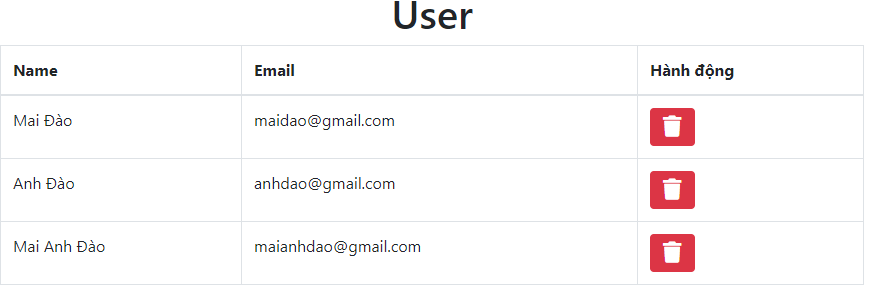
1. Admin – Xem đơn hàng của khách

Admin có thể xem đơn hàng và tình trạng đơn hàng đã được gửi hay chưa.



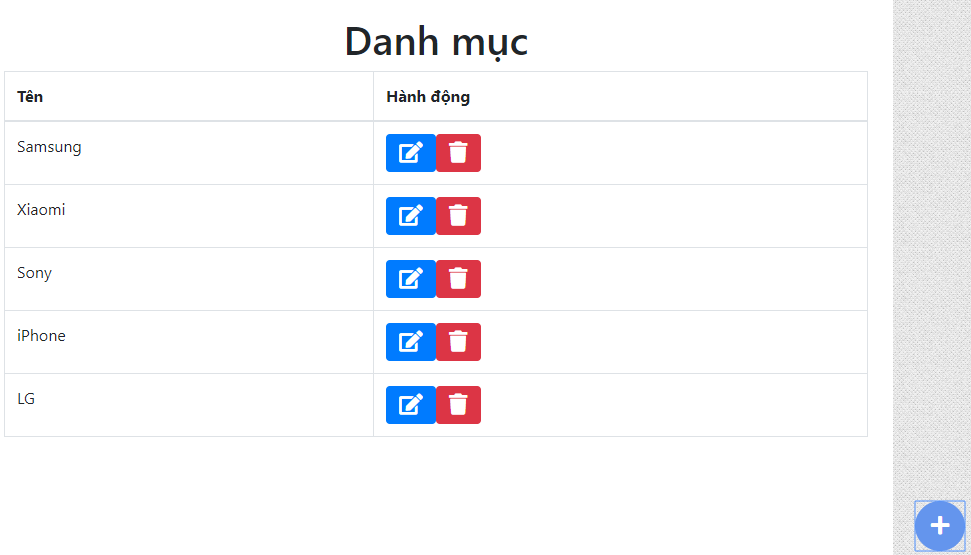
1. Admin – Xem danh sách người dùng

Admin có thể xem danh sách người dùng và có quyền xóa những người dùng mình muốn.

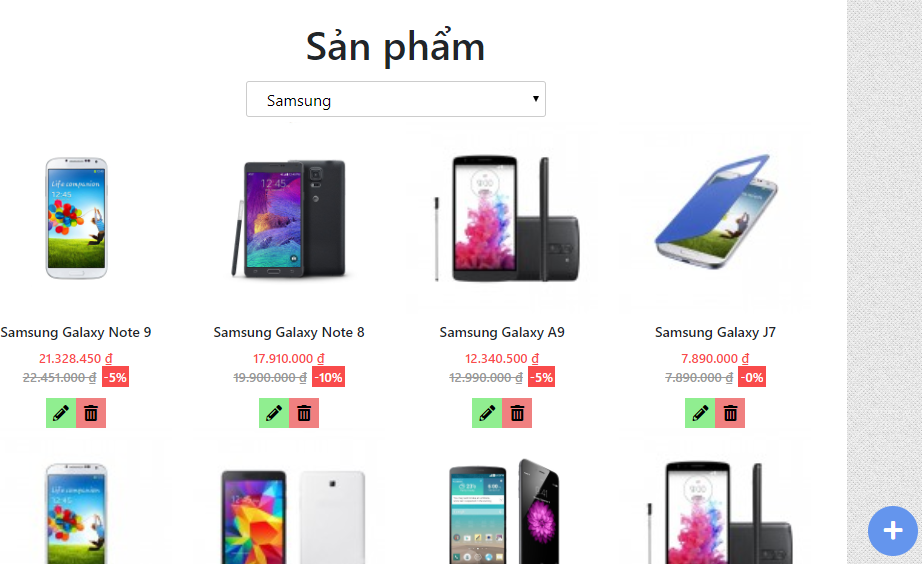


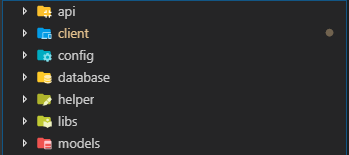
1. Admin – Quản lý danh mục

Admin có thể xem, thêm xóa sửa đối với các danh mục của sản phẩm



1. Admin – Quản lý sản phẩm

Cũng như danh mục, admin có thể xem và thêm, xóa, sửa đối với sản phẩm của từng danh mục.

1. **Cấu trúc thư mục code**

Thư mục được thiết kế theo mô hình MVC. Với api đóng vai trò là controller, xử lý tất cả các request từ client. Models xử lý các tác vụ liên quan đến database như đọc, xóa, sửa, lấy dữ liệu. Client đại diện cho view, chỉ làm nhiệm vụ hiển thị giao diện và tương tác với người dùng.

1. **Cài đặt ứng dụng và các phiên bản hỗ trợ**
2. **Bảng phân công công việc**

|  |  |  |
| --- | --- | --- |
| **Tên** | **Vai trò** | **Những mục đã thực hiện** |
| 1. Lê Hữu Việt Anh | Nhóm trưởng | Làm báo cáo |
| 1. Nguyễn Thị Mỹ Duyên | Thành viên | Làm báo cáo phần Oracle |
| 1. Mai Anh Đào | Thành viên | Làm báo cáo phần MySQL |
| 1. Bùi Phương Lan | Thành viên |  |