**Đề 2 – Product**

**Mô Tả Cách Thức Hoạt Động Source Code**

Lời đầu tiên, em xin cảm ơn Công ty TNHH LAMPART đã dành thời gian review CV của em và cho em cơ hội tham gia bài test PHP Intern. Sau đây em xin trình bày môi trường code và cách thức Source Code hoạt động.

* Môi trường dự án:

1. PHP version 7.4
2. Wampserver 64
3. MySQL version 8.0.31
4. Bootstrap version 4.6

* Phầm mềm code

Visual Studio Code

* Soruce Code (Xây dựng theo mô hình MVC)

1. Folder Config có File Config.php để khai báo điều kiện kết nối DATABASE.
2. Folder Controller có các Class Controller gồm các Function để kết nối với Model và điều hướng trang theo ý muốn.
3. Folder Message có Function ProductNotification để quản lý các thông báo khi thêm, sửa, copy sản phẩm thành công hoặc thất bại.
4. Folder Model có các Class Model gồm các Function để thực thi Logic, thực thi các câu truy vấn SQL.
5. Folder public gồm các folder css, img, js… Để xây dựng trang giao diện.
6. Folder System gồm:
   1. Folder Db có Class Database gồm các Function để kết nối và thao tác với DATABASE. Còn có Class QueryBuilder gồm các Function để thực thi các câu truy vấn Insert, Update, Select…
   2. Folder Library có file Funcs gồm các Function tiện ích để tái sử dụng ở nhiều nơi trong dự án.
   3. File Autoload để tự động Include các Class cần sử dụng trong dự án.
7. Folder view gồm các trang để hiện thị cho người xem.
   1. File layout có chức năng là Layout tổng.
8. File index để xác định yêu cầu từ người xem và điều hướng đến các Controller thông qua URL.

* Cách thức Source Code hoạt động

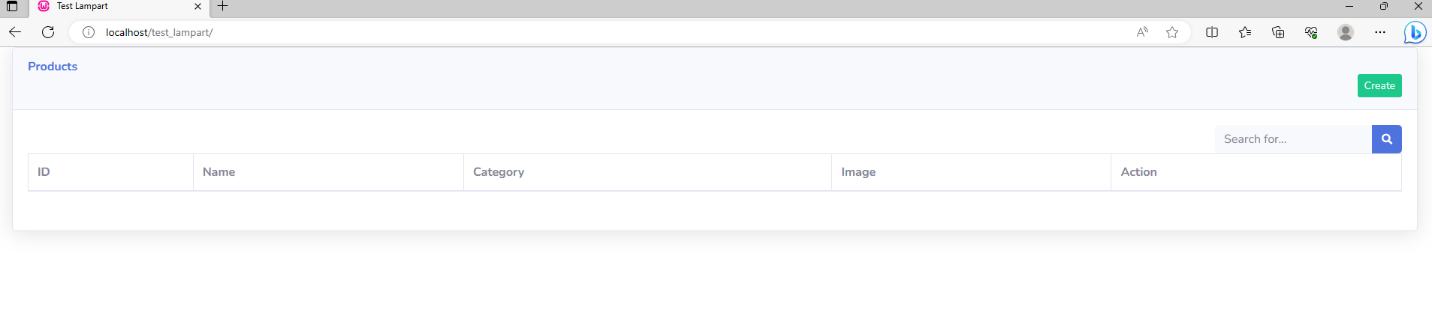
Request

Controller

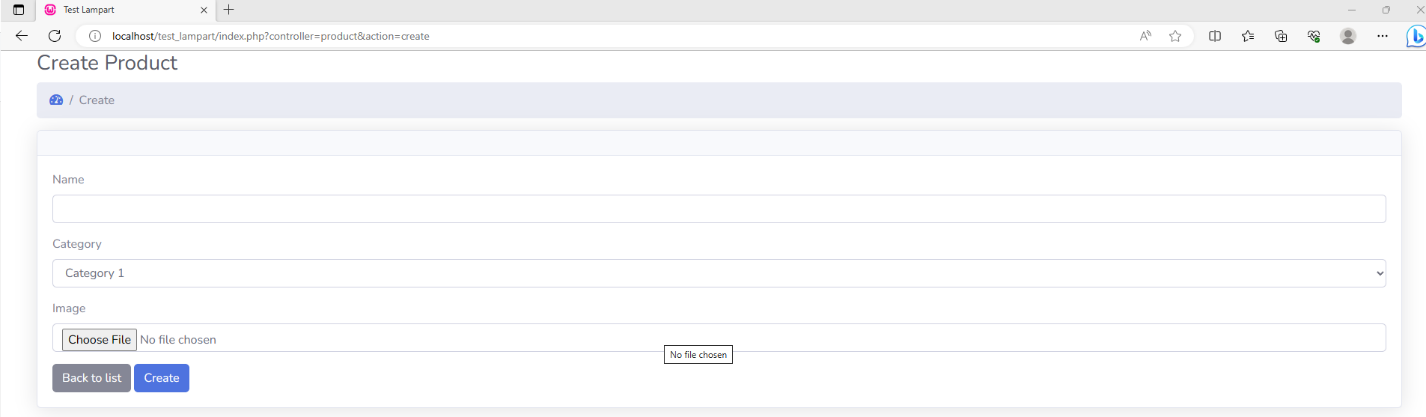
Model

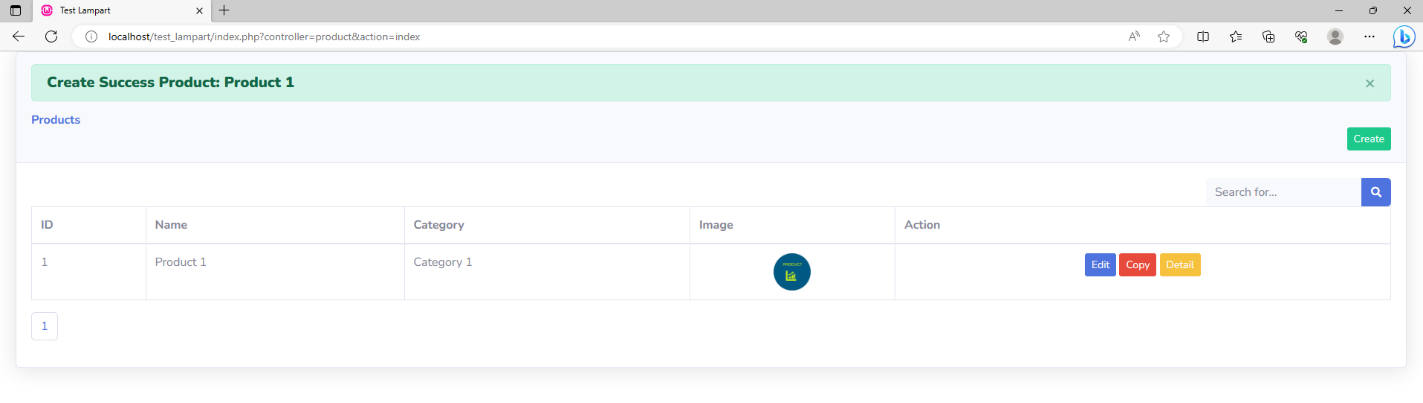
View

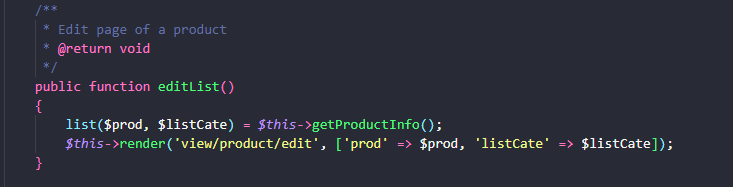
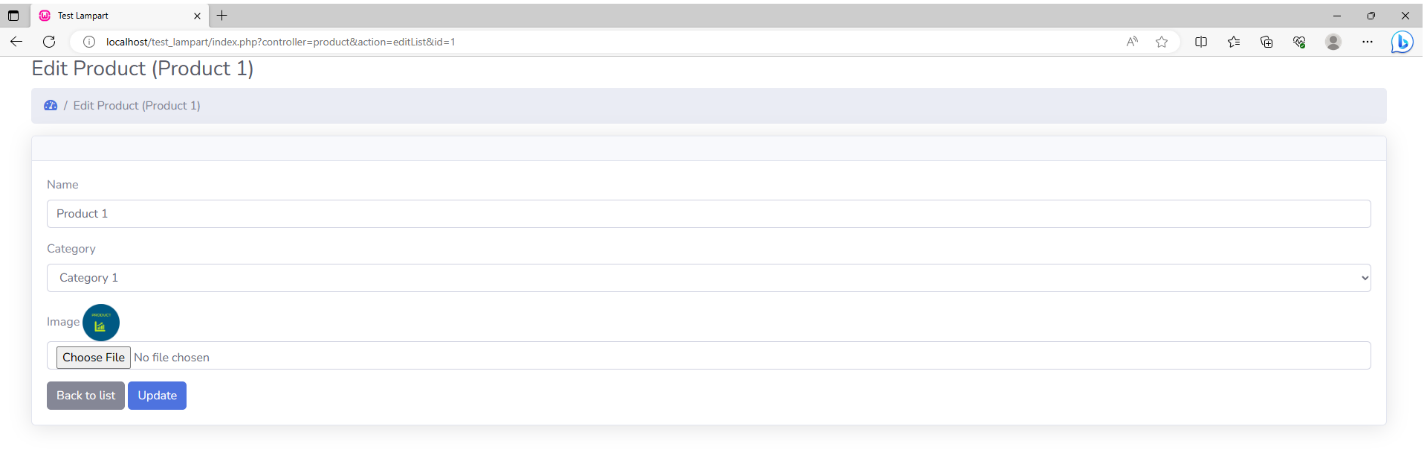
Database

* Mở phần mềm Wampserver 64
* Mở trình duyệt, nhập đường dẫn localhost/phpmyadmin/, import file database được đính kèm và đặt tên database như trong file Config (test\_lampart). Hoặc có thể tự đặt tên theo ý muốn nhưng phải sửa tên DBNAME trong file Config.
* Mở trình duyệt nhập đường dẫn localhost/test\_lampart (do folder được đặt tên là test\_lampart).
* Khi truy cập vào đường dẫn trên yêu cầu sẽ được gửi đến file index.php, index.php sẽ dựa vào URL để chọn Controller tương ứng để xử lý, mặc định là controller = product và action = index.
* Hiện tại chưa có sản phẩm nào, Bấm vào Create để tạo sản phẩm mới, lúc này index.php sẽ điều hướng đến ProductController và chạy đến Function create().



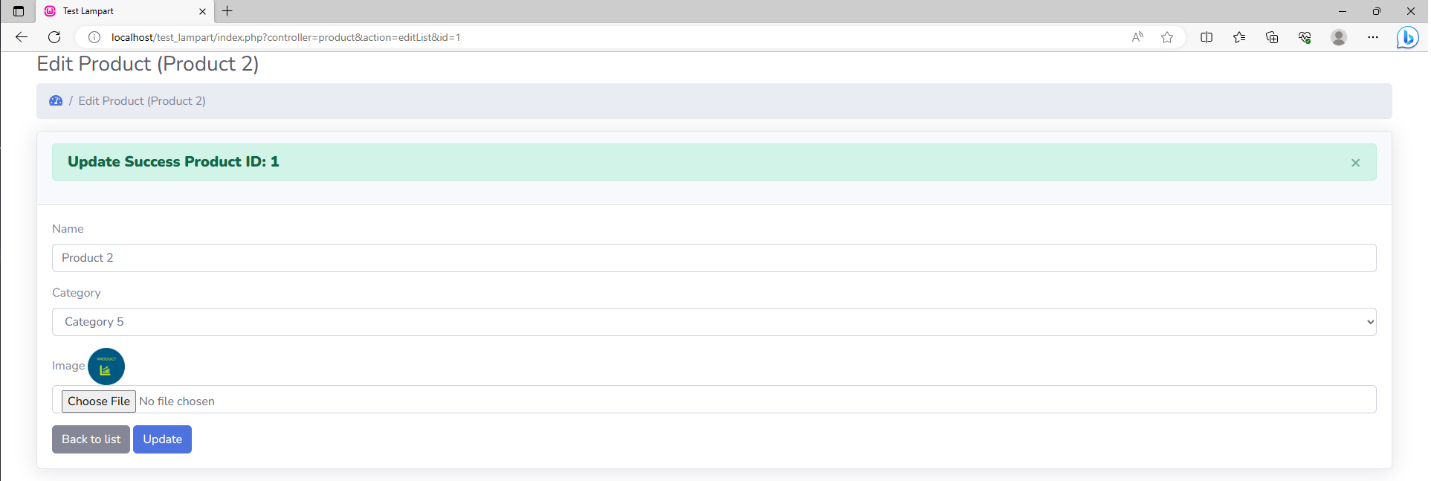
* Bảng categories các thành viên được tạo thủ công. Trong function create() biến $listCate được gọi từ Function listCategory trong Model Category.
* Ở giao diện, đặt tên sản phẩm, chọn danh mục, 1 sản phẩm chỉ có 1 danh mục nhưng 1 danh mục thì có nhiều sản phẩm. Sau khi bấm Create, yêu cầu sẽ được đưa đến hàm insert()

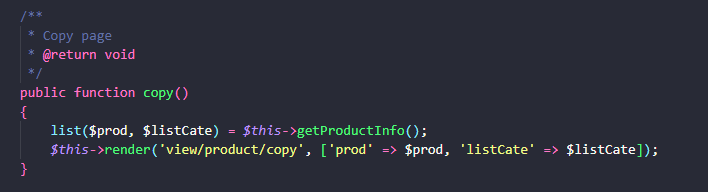
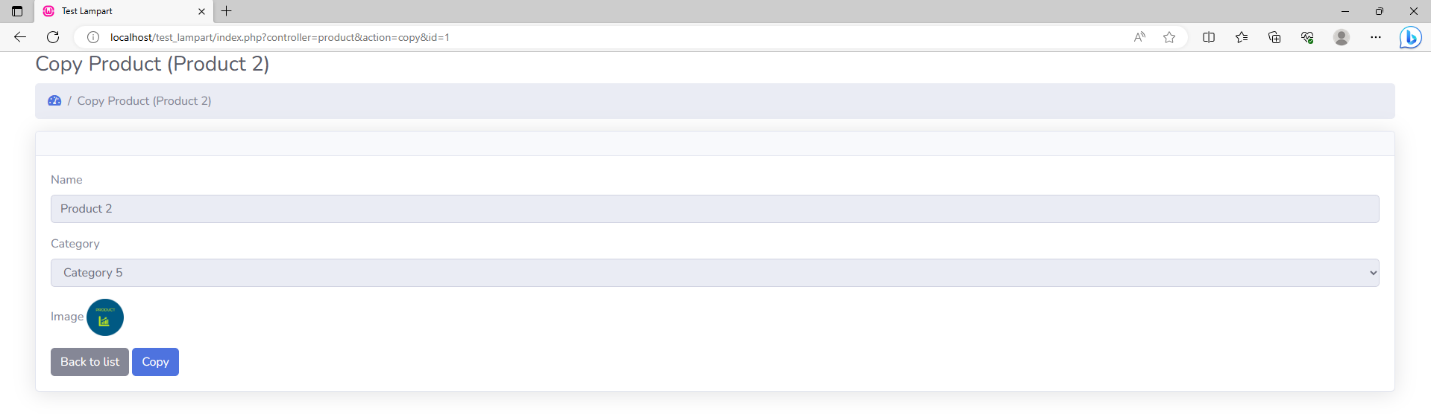
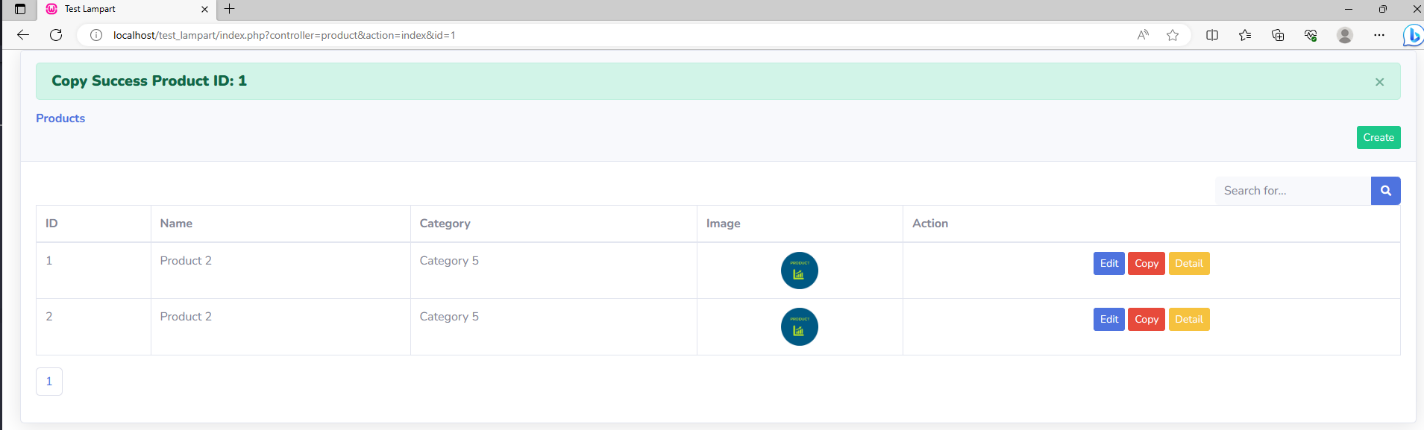


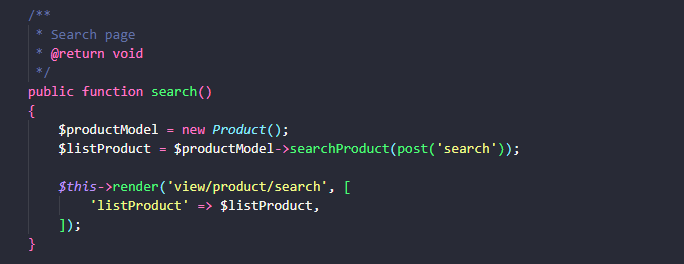
* Sau khi tạo xong 1 sản phẩm thành công thì sẽ được điều hướng về lại index, xuất hiện sản phẩm vừa tạo ở danh sách và kèm theo 1 thông báo.
* Bấm vào Edit để chỉnh sửa sản phẩm, lúc này index.php sẽ dựa vào URL gọi đến Function editList trong ProductController.
* Và điều hướng đến trang giao diện Edit, chỉnh sửa sản phẩm và bấm update. Lúc này Function updateList sẽ được gọi.



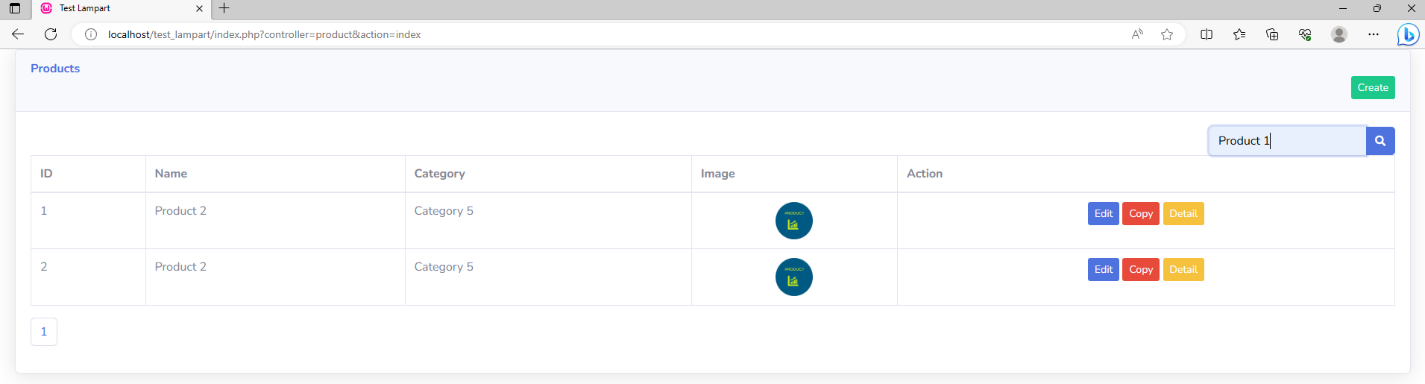
* Function nè sẽ update lại sản phẩm khi thay đổi tên, danh mục hoặc hình ở ngoài giao diện, nếu không nhập gì thì mặc định sẽ giữ nguyên giá trị của sản phẩm.

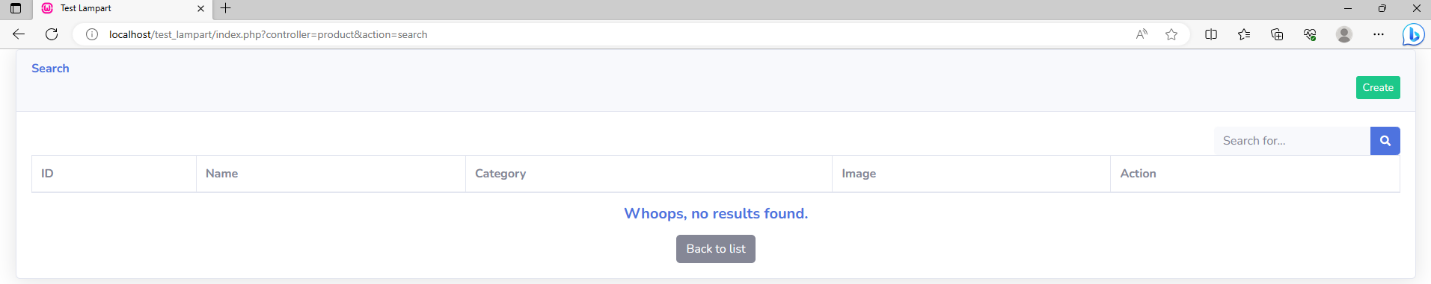


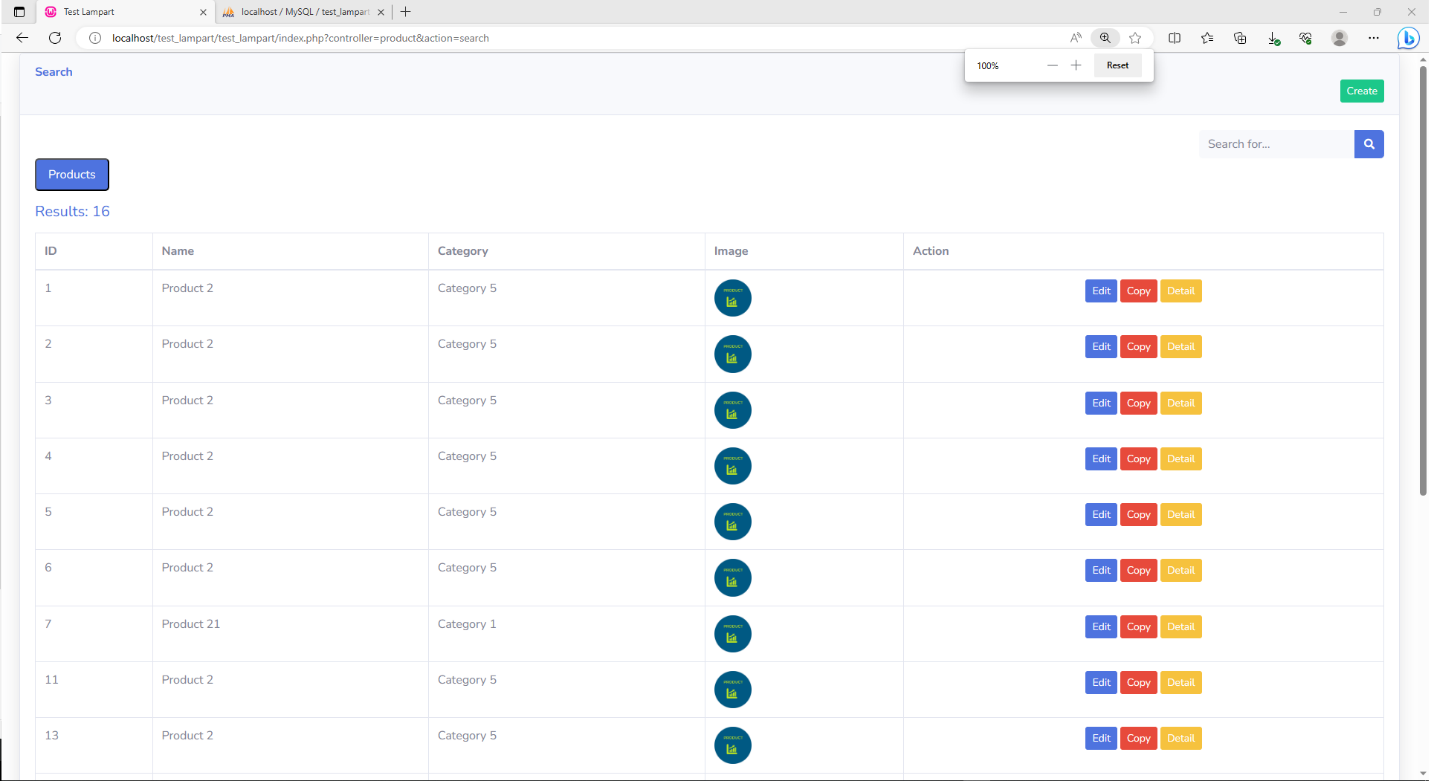
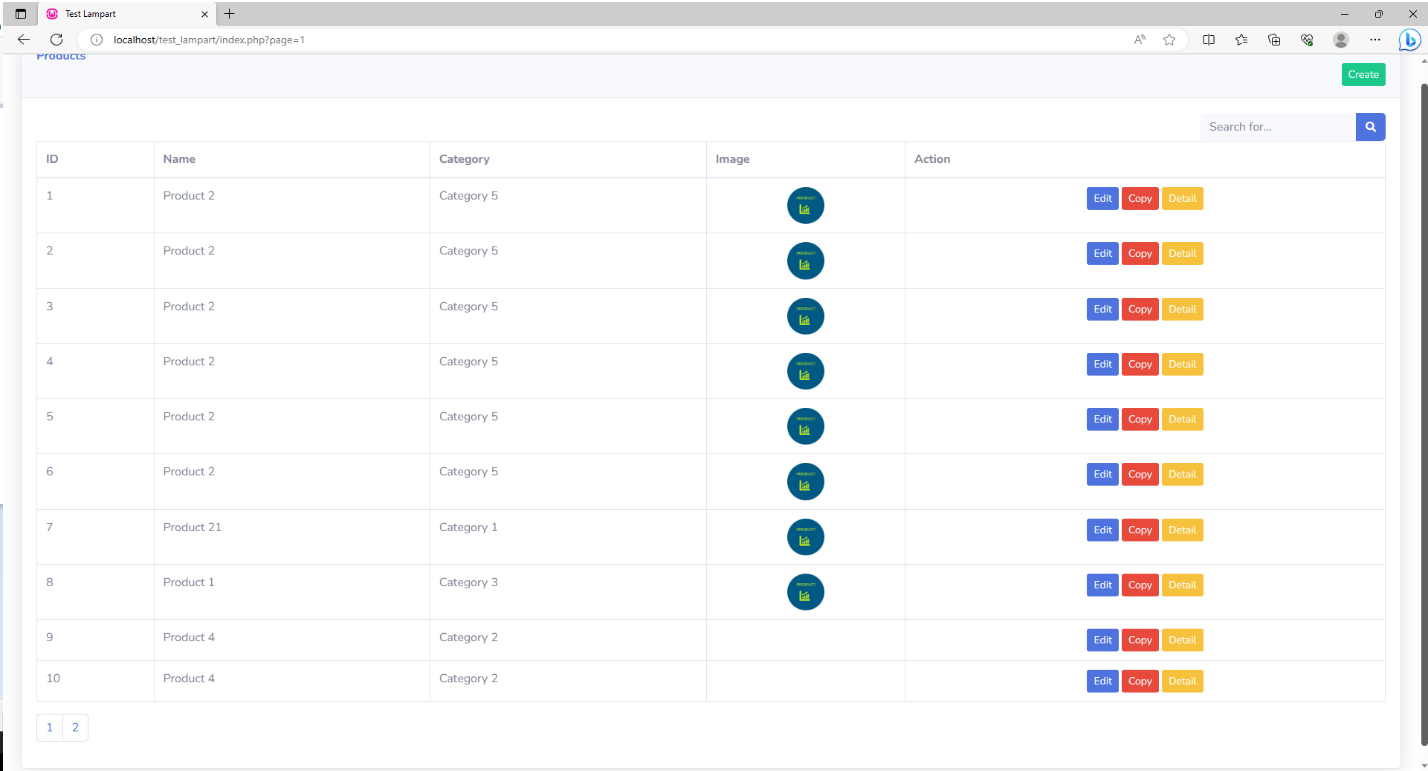
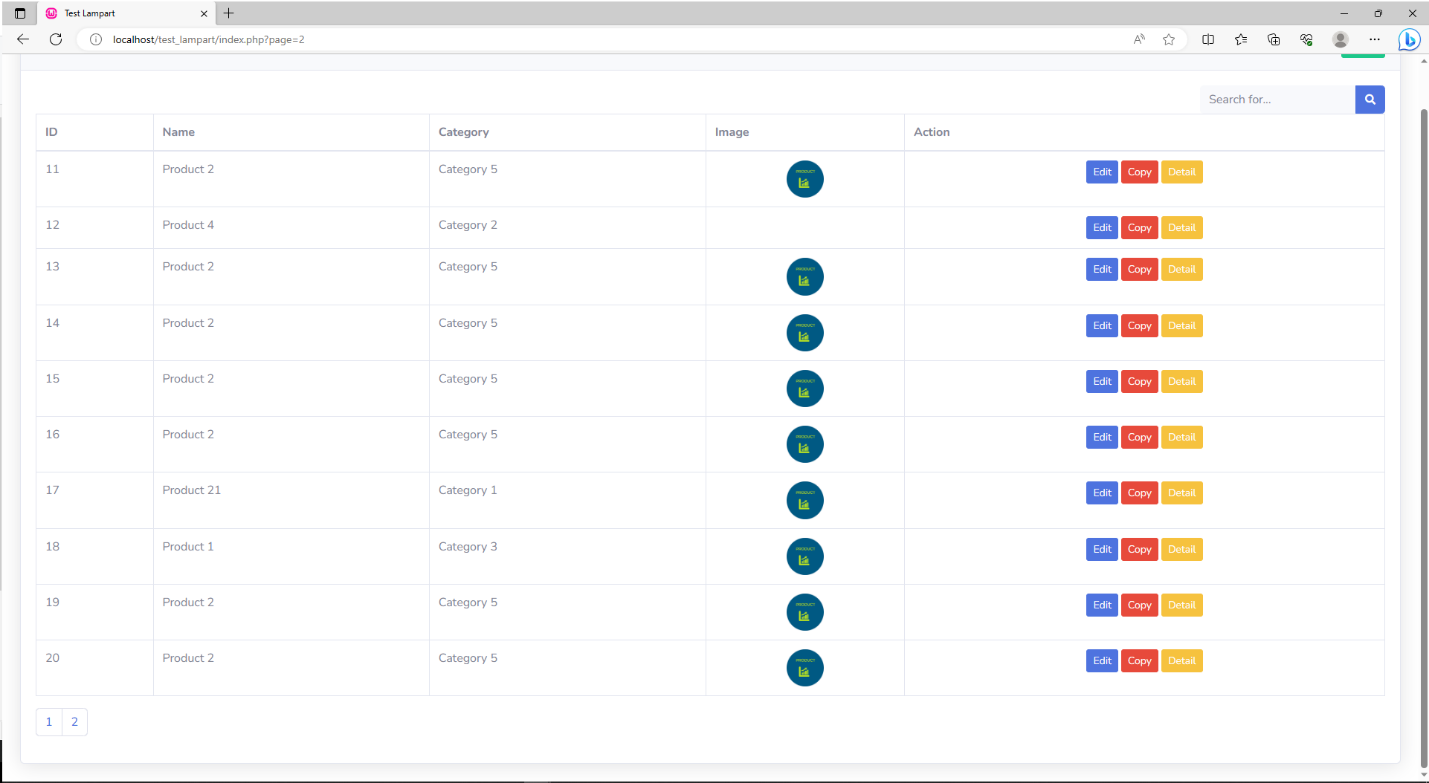
* Sau khi update thành công sẽ có 1 thông báo hiện ra.
* Giờ quay lại trang index và bấm copy, Funcion copy trong ProductController sẽ được gọi và sẽ được điều hướng đến giao diện copy.
* Bấm vào copy, Function copyInsert sẽ được gọi đến vào thêm thành viên vào database.
* Sau khi copy thành công sẽ điều hướng lại trang và kèm 1 thông báo
* Quay lại trang index và bấm vào detail để xem chi tiết sản phẩm. Lúc này index.php sẽ điều hướng đến ProductController, gọi Function detail và điều hướng đến trang Detail theo URL.
* Quay lại trang index và gõ vào thanh search để tìm sản phẩm theo tên hoặc tìm theo danh mục. Lúc này index.php sẽ điều hướng đến ProductController, gọi function search và điều hướng đến trang Search theo URL.

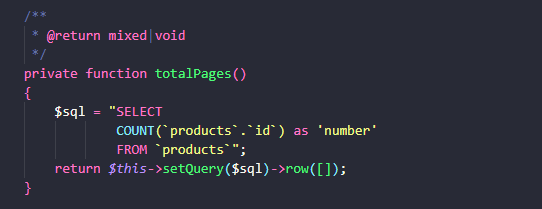


* Trong Function này sẽ gọi đến Function searchProduct ở trong Model Product.
* Function searchProduct trong Model Product. Function này sử dụng truy vấn liên kết 2 bảng và LIKE để tìm kết quả gần đúng.
* Tìm Product 1 trong thanh search và bấm tìm. Lúc này sẽ điều hướng đến trang search.

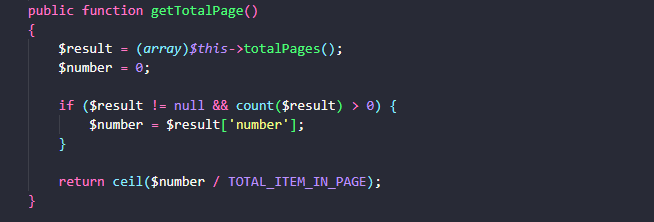


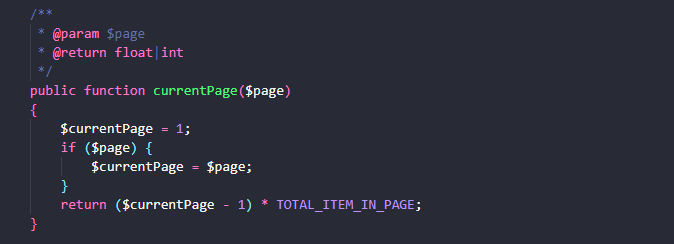
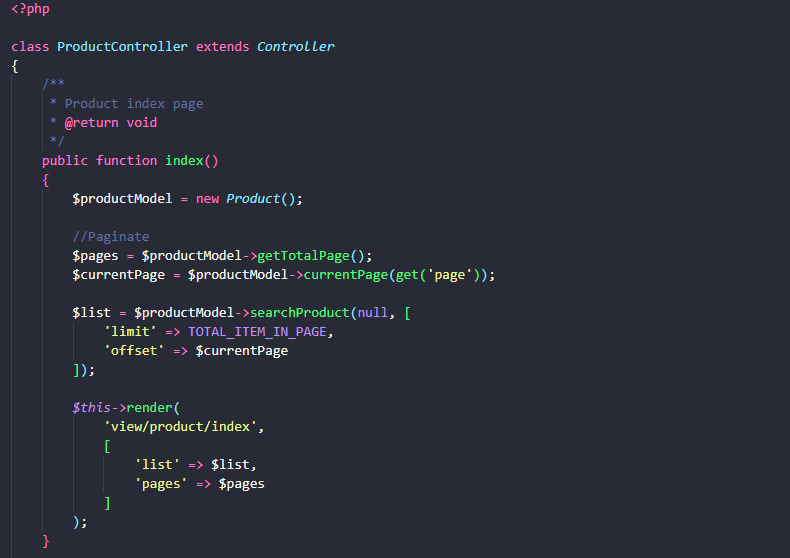


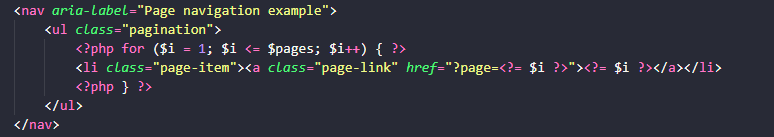
* Không có kết quả tìm kiếm Product 1. Vậy hãy tìm Product 2.
* Hiển thị 16 kết quả tìm kiếm liên quan đến Product 2.
* Giờ tạo hoặc copy ra 20 sản phẩm.
* Trang 1 sẽ hiển thị 10 sản phẩm có ID từ 1 đến 10.
* Trang 2 sẽ hiển thị sản phẩm có ID từ 11 đến trở đi.
* Để phân trang, em sử dụng 3 Function totalPages(), getTotalPage(), currentPage($page) được viết trong Model Product.
* Function totalPages() để tìm ra tổng sản phẩm có trong bảng products.



* Function getTotalPage () để tính ra số trang tương ứng cho 1 trang 10 sản phẩm. Function ceil để làm tròn số nếu $pages ra số lẻ.



* Funcion currentPage($page) để tính số ID sản phẩm được hiển thị ở trang nào.
* Để hiện thị 10 sản phẩm cho 1 trang, thực hiện câu truy vấn ở Function searchProduct($searchKey = null, $paging = []).
* Phân trang ở ProductController, Function index
* Ở Function này khởi tạo class Product, sau đó gọi đến các function getTotalPage(), currentPage($page) và searchProduct(). Và kèm theo data $list và $pages ra view/product/index.
* Hiện thị phân trang ở ngoài view/product/index.



* Chạy vòng lặp for cho $i = 1 và $i <= $pages ($pages data được đưa ra view từ Function index trong ProductController).

Trên đây là phần giải thích cách hoạt động Source Code của em. Do đây là lần đầu em làm file Document như thế này nên không thể tránh khỏi thiếu sót. Nếu Anh/Chị đọc file này có khó hiểu mong Anh/Chị góp ý giúp em.

Một lần nữa em chân thành cảm ơn Công Ty TNHH LAMPART đã tạo cơ hội cho em tham gia bài test này.