**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày …... tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

Em xin gửi lời cảm ơn chân thành đến cô Nguyễn Ngọc Đan Thanh - giảng viên Bộ môn Công nghệ thông tin - Khoa Kỹ thuật và Công nghệ của trường Đại học Trà Vinh đã giúp đỡ và hướng dẫn cho em rất nhiều trong quá trình tìm hiểu và thực hiện đồ án cơ sở ngành.

Tuy nhiên trong quá trình thực hiện đồ án cơ sỡ ngành, do chưa có nhiều kinh nghiệm và kiến thức còn hạn chế trong quá trình tìm hiểu cũng như làm đồ án nên em vẫn còn gặp khá nhiều hạn chế và thiếu sót. Rất mong nhận được sự quan tâm, nhận xét và những lời góp ý của các thầy/ cô giảng viên bộ môn để cho đề tài của em được hoàn thiện chỉnh chu hơn.

Em xin chân thành cảm ơn.

**MỤC LỤC**

[CHƯƠNG 1 TỔNG QUAN 9](#_Toc152895912)

[CHƯƠNG 2 NGHIÊN CỨU LÝ THUYẾT 11](#_Toc152895913)

[2.1 AngularJS 11](#_Toc152895914)

[2.1.1 Cách nhúng AngularJS 11](#_Toc152895915)

[2.1.2 AngularJS Expressions 11](#_Toc152895916)

[2.1.3 AngularJS Modules 13](#_Toc152895917)

[2.1.4 AngularJS Directives 14](#_Toc152895918)

[2.1.5 AngularJS Model 16](#_Toc152895919)

[2.1.6 AngularJS Data Binding 19](#_Toc152895920)

[2.1.7 AngularJS Controllers 21](#_Toc152895921)

[2.1.8 AngularJS Scopes 23](#_Toc152895922)

[2.1.9 AngularJS Filters 24](#_Toc152895923)

[2.1.10 AngularJS Services 26](#_Toc152895924)

[2.1.11 AngularJS Http 27](#_Toc152895925)

[2.1.12 AngularJS Tables 30](#_Toc152895926)

[2.1.13 AngularJS Select 31](#_Toc152895927)

[2.1.14 AngularJS SQL 33](#_Toc152895928)

[2.1.15 AngularJS DOM 35](#_Toc152895929)

[2.1.16 AngularJS Events 37](#_Toc152895930)

[2.1.17 AngularJS Forms 38](#_Toc152895931)

[2.1.18 AngularJS Validation 38](#_Toc152895932)

[2.1.19 AngularJS API 39](#_Toc152895933)

[2.1.20 AngularJS W3.CSS 40](#_Toc152895934)

[2.1.21 AngularJS Includes 40](#_Toc152895935)

[2.1.22 AngularJS Animations 41](#_Toc152895936)

[2.1.23 AngularJS Routing 41](#_Toc152895937)

[2.1.24 AngularJS Application 42](#_Toc152895938)

[2.2 PHP & MySQL 42](#_Toc152895939)

[2.2.1 PHP Connect to MySQL 42](#_Toc152895940)

[2.2.2 PHP Create a MySQL Database 43](#_Toc152895941)

[2.2.3 PHP MySQL Create Table 44](#_Toc152895942)

[2.2.4 PHP MySQL Insert Data 44](#_Toc152895943)

[2.2.5 PHP MySQL Get Last Inserted ID 45](#_Toc152895944)

[2.2.6 PHP MySQL Select Data 45](#_Toc152895945)

[2.2.7 PHP MySQL Use The WHERE Clause 45](#_Toc152895946)

[2.2.8 PHP MySQL Use The ORDER BY Clause 45](#_Toc152895947)

[2.2.9 PHP MySQL Delete Data 45](#_Toc152895948)

[2.2.10 PHP MySQL Update Data 46](#_Toc152895949)

[2.3 Kết luận 46](#_Toc152895950)

[CHƯƠNG 3 HIỆN THỰC HÓA NGHIÊN CỨU 47](#_Toc152895951)

[3.1 Mô tả bài toán 47](#_Toc152895952)

[3.2 Đặc tả yêu cầu hệ thống 47](#_Toc152895953)

[3.2.1 Yêu cầu chức năng 47](#_Toc152895954)

[3.2.2 Yêu cầu phi chức năng 48](#_Toc152895955)

[3.3 Thiết kế dữ liệu 48](#_Toc152895956)

[3.3.1 Lược đồ cơ sở dữ liệu 48](#_Toc152895957)

[3.3.2 Chi tiết các thực thể 49](#_Toc152895958)

[3.4 Thiết kế giao diện 50](#_Toc152895959)

[3.4.1 Sơ đồ website 50](#_Toc152895960)

[3.4.2 Giao diện website 51](#_Toc152895961)

[CHƯƠNG 4 KẾT QUẢ NGHIÊN CỨU 53](#_Toc152895962)

[4.1 Dữ liệu thử nghiệm 53](#_Toc152895963)

[4.2 Kết quả thử nghiệm 54](#_Toc152895964)

[4.2.1 Chức năng người dùng 54](#_Toc152895965)

[4.2.2 Chức năng quản trị 57](#_Toc152895966)

[4.3 Kết luận 58](#_Toc152895967)

[CHƯƠNG 5 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 59](#_Toc152895968)

[DANH MỤC TÀI LIỆU THAM KHẢO 60](#_Toc152895969)

[PHỤ LỤC 61](#_Toc152895970)

**DANH MỤC HÌNH ẢNH – BẢNG BIỂU**

[Bảng 3.1: Danh sách các thực thể 48](#_Toc154880609)

[Bảng 3.2 Chi tiết thực thể “sanpham” 49](#_Toc154880610)

[Bảng 3.3: Chi tiết thực thể “hinhanh” 49](#_Toc154880611)

[Bảng 3.4: Chi tiết thực thể “thuonghieu” 49](#_Toc154880612)

[Bảng 3.5: Chi tiết thực thể “nguoidung” 50](#_Toc154880613)

[Hình 3.1 Cơ sở dữ liệu 48](#_Toc154880593)

[Hình 3.2 Sitemap cửa hàng điện thoại Trung Nghĩa Store 50](#_Toc154880594)

[Hình 4.1 Bảng hình ảnh 53](#_Toc154880595)

[Hình 4.2 Bảng sản phẩm 53](#_Toc154880596)

[Hình 4.3 Bảng thương hiệu 53](#_Toc154880597)

[Hình 4.4 Bảng người dùng 53](#_Toc154880598)

[Hình 4.5 Xem trang chủ 54](#_Toc154880599)

[Hình 4.6 Xem chi tiết sản phẩm 54](#_Toc154880600)

[Hình 4.7 Chức năng lọc theo hãng 55](#_Toc154880601)

[Hình 4.8 Chức năng lọc theo giá 55](#_Toc154880602)

[Hình 4.9 Chức năng tìm kiếm 56](#_Toc154880603)

[Hình 4.10 Chức năng đăng nhập 56](#_Toc154880604)

[Hình 4.11 Quản lý thương hiệu 57](#_Toc154880605)

[Hình 4.12 Quản lý sản phẩm 57](#_Toc154880606)

[Hình 4.13 Quản lý người dùng 57](#_Toc154880607)

[Hình 4.14 Thống kê sản phẩm 58](#_Toc154880608)

**TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH**

Vấn đề nghiên cứu

Đồ án này tập trung vào việc nghiên cứu về AngularJS, một framework JavaScript mạnh mẽ và cách áp dụng nó trong phát triển ứng dụng web hiện nay. Vấn đề cơ bản là làm thế nào để có thể sử dụng các tính năng của AngularJS làm cho trang web trở nên hấp dẫn hơn, tăng cường tính linh hoạt và hiệu suất tốt hơn.

Hướng tiếp cận

Tôi đã lựa chọn một hướng tiếp cận tổng thể bao gồm hai phần chính: tìm hiểu lý thuyết về AngularJS và áp dụng nó vào thực tế. Đối với phần lý thuyết, tôi đã nghiên cứu cấu trúc MVC của AngularJS, các khái niệm như databinding, directives và dependency injection. Đồng thời, tôi đã tạo ra dự án mẫu để minh họa cách sử dụng AngularJS trong thực tế.

Cách giải quyết vấn đề

Để giải quyết vấn đề, tôi đã tiến hành một quá trình học tập chặt chẽ, đặt nền tảng cho việc tìm hiểu rõ về AngularJS. Bằng cách này tôi có thể xác định cách sử dụng các tính năng của nó để giải quyết những yêu cầu thường gặp trong phát triển ứng dụng web. Cùng với đó, việc xây dựng và triển khai các dự án thực tế giúp tôi kiểm tra và xác nhận hiệu suất của AngularJS trong thực tế.

Kết quả đạt được

Hiểu rõ hơn về AngularJS: Tôi đã hiểu về cấu trúc và các khái niệm quan trọng của AngularJS như databinding, directives, và dependency injection. Áp dụng thành công vào dự án của tôi, chứng minh khả năng ứng dụng linh hoạt của AngularJS trong môi trường phát triển ứng dụng web.

**MỞ ĐẦU**

Lý do chọn đề tài

AngularJS đã trở thành một phần không thể thiếu trong quá trình phát triển web hiện nay. Lựa chọn đề tài nghiên cứu về AngularJS là do thấy được sự ảnh hưởng và tầm quan trọng của nó đối với việc xây dựng ứng dụng web. Nó được ưa chuộng vì tính mạnh mẽ, tích hợp dễ dàng, và khả năng tạo ra trải nghiệm người dùng tốt. Một số công ty lớn như Google, Microsoft Office, Gmail, PayPal và Samsung đã sử dụng AngularJS để xây dựng và quản lý trang web của họ. Điều này cho thấy tầm quan trọng và sự phổ biến của AngularJS. Chính vì vậy, tôi quyết định thực hiện đồ án cơ sở ngành với đề tài “Tìm hiểu về AngularJS và thiết kế ứng dụng website minh họa”.

Mục đích nghiên cứu

Mục đích chính của đồ án này là hiểu rõ các kiến thức cơ bản về các tính năng và lợi ích của AngularJS, đồng thời áp dụng kiến thức đó vào việc xây dựng các ứng dụng web thực tế. Tôi mong muốn không chỉ có kiến thức lý thuyết mà còn có khả năng ứng dụng linh hoạt trong các dự án thực tế.

Đối tượng nghiên cứu

Đối tượng nghiên cứu của tôi là những kiến thức cơ bản và các tính năng quan trọng của AngularJS, cách để áp dụng kiến thức đó vào việc xây dựng các ứng dụng web thực tế.

Phạm vi nghiên cứu

Phạm vi nghiên cứu của tôi tập trung vào việc tìm hiểu về cấu trúc MVC của AngularJS, các khái niệm như databinding, dependency injection và directives.

# TỔNG QUAN

Sự bùng nổ của ứng dụng web động và phức tạp ngày nay đặt ra thách thức lớn đối với các nhà phát triển. Việc duy trì mã nguồn, quản lý dữ liệu và tương tác người dùng đòi hỏi một quy trình phát triển hiệu quả và linh hoạt. Trong bối cảnh này, AngularJS nổi lên như một công cụ mạnh mẽ để giúp giải quyết những thách thức này và mang lại trải nghiệm người dùng mượt mà.

AngularJS được phát triển và duy trì bởi Google, không chỉ là một framework JavaScript thông thường mà còn là một ứng dụng "Model View Controller" (MVC). Điều này có nghĩa là ứng dụng được chia thành ba phần chính để quản lý dễ dàng: Mô hình (Model), Xem (View) và Bộ điều khiển (Controller).

Mục tiêu của đồ án này là tìm hiểu về cách AngularJS hoạt động, cách ứng dụng nó vào quá trình phát triển một trang web. Dưới đây là các bước để hoàn thành đồ án.

**Bước 1**: Phân tích nhu cầu và thiết kế hệ thống

Phân tích nhu cầu: Bắt đầu bằng việc xác định yêu cầu chính của dự án bao gồm chức năng của trang web, nhu cầu kinh doanh và yêu cầu về dữ liệu.

Thiết kế hệ thống: Xây dựng một thiết kế hệ thống tổng quan, bao gồm cấu trúc của trang web, các chức năng chính và quy trình kết nối với cơ sở dữ liệu MySQL thông qua PHP.

**Bước 2**: Nghiên cứu AngularJS và PHP

Tìm hiểu AngularJS: Nghiên cứu cơ bản về AngularJS, bao gồm cấu trúc MVC, directives, databinding, và các tính năng quan trọng để hiểu rõ cách sử dụng AngularJS trong phát triển front-end của trang web.

Kết nối PHP với MySQL: Đọc tài liệu và học cách sử dụng PHP để kết nối và tương tác với cơ sở dữ liệu MySQL.

**Bước 3**: Thiết kế giao diện và Database

Thiết kế giao diện: Sử dụng công cụ thiết kế giao diện hoặc vẽ các mockup để định hình cấu trúc và trải nghiệm người dùng của trang web.

Thiết kế Database: Xác định cấu trúc của cơ sở dữ liệu MySQL bao gồm bảng, quan hệ và các truy vấn cần thiết cho ứng dụng.

**Bước 4**: Hiện thực hóa nghiên cứu

Cài đặt môi trường: Cài đặt và cấu hình môi trường phát triển, bao gồm máy chủ web (Laragon), PHP, MySQL và AngularJS.

Hiện thực hóa Front-end: Viết mã front-end bằng Html, Css, AngularJS áp dụng các khái niệm đã nghiên cứu như databinding và directives để xây dựng giao diện người dùng tương tác.

Hiện thực hóa Back-end: Viết mã back-end bằng PHP để xử lý các yêu cầu từ front-end, kết nối và thực hiện các truy vấn đến cơ sở dữ liệu MySQL.

Kiểm thử và Debug: Thực hiện kiểm thử chức năng và kiểm thử bảo mật để đảm bảo tính ổn định và an toàn của trang web.

# NGHIÊN CỨU LÝ THUYẾT

Trình bày cơ sở lý thuyết, lý luận, giả thiết khoa học và phương pháp nghiên cứu đã được sử dụng trong đồ án. [1] [1]

## AngularJS

### Cách nhúng AngularJS

AngularJS là một framework JavaScript. Nó có thể được thêm vào trang HTML bằng thẻ <script> trong phần header.

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>

AngularJS mở rộng các thuộc tính HTML bằng chỉ thị (directive) và liên kết dữ liệu với HTML bằng biểu thức (expressions).

Phần mở rộng (extend): AngularJS mở rộng HTML bằng **ng-directives**.

Lệnh **ng-app** xác định đâu là một ứng dụng AngularJS.

Lệnh **ng-model** liên kết giá trị của các điều khiển HTML như (input, select, textarea) với dữ liệu ứng dụng.

Lệnh **ng-bind** liên kết dữ liệu của ứng dụng với chế độ hiển thị trong HTML.

### AngularJS Expressions

AngularJS liên kết dữ liệu với HTML bằng cách sử dụng biểu thức.

Các biểu thức trong AngularJS cho phép chèn trực tiếp vào HTML. Điều này giúp cho phần hiển thị dữ liệu một cách đơn giản và linh hoạt trên giao diện người dùng.

Các biểu thức trong AngularJS được viết bên trong hai cặp ngoặc nhọn **{{ expression }}** hoặc bên trong cặp ngoặc kép của lệnh **ng-bind** **= "expression"**.

Biểu thức của AngularJS cũng giống với biểu thức của Javascript. Nó có thể chứa hằng số, toán tử và biến.

Tôi có thể viết biểu thức ở bất kì đâu mà tôi muốn vì AngularJS sẽ chỉ thực hiện xử lý và trả kết quả tại nơi biểu thức được viết.

Ví dụ: Tính tổng 5 + 5 và in ra kết quả tại vị trí nơi mà biểu thức được viết.

<!DOCTYPE html>  
<html>  
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>  
<body>  
  
 <div ng-app="">  
   <p>Tong cua hai so la: {{ 5 + 5 }}</p>  
 </div>  
  
</body>  
</html>

Kết quả sau khi chạy chương trình là.

Tong cua hai so la: 10

Nếu xóa lệnh **ng-app** thì HTML sẽ hiển thị biểu thức như cũ mà không giải quyết nó.

Kết quả sẽ được hiểu thị như sau.

Tong cua hai so la: {{ 5 + 5 }}

Ngoài ra AngularJS còn có thể thay đổi giá trị các thuộc tính của CSS.

Ví dụ: Thay đổi màu nền cho ô nhập dữ liệu.

<div ng-app="" ng-init="myCol='lightblue'">  
  
 <input style="background-color:{{myCol}}" ng-model="myCol">  
  
</div>

Kết quả sẽ được hiển thị như sau.



**Số trong AngularJS**

Ví dụ về số trong AngularJS.

<div ng-app="" ng-init="a=1; b=5">  
  
 <p>Tong cua a va b la: {{ a + b }}</p>  
  
</div>

Kết quả sau khi chạy chương trình là.

Tong cua a va b la: 6

**Chuỗi trong AngularJS**

Ví dụ về chuỗi trong AngularJS.

<div ng-app="" ng-init="firstName = Trung; lastName = 'Trung Nghia'">  
  
 <p>Full name is: {{ firstName + " " + lastName }}</p>  
  
</div>

Kết quả sau khi chạy chương trình là.

Full name is: Trung Nghia

**Đối tượng trong AngularJS**

Ví dụ về đối tượng trong AngularJS.

<div ng-app="" ng-init="person = {firstName: 'Trung', lastName: 'Nghia'}">  
  
 <p>The name is: {{ person.lastName }}</p>  
  
</div>

Kết quả sau khi chạy chương trình là.

The name is: Nghia

**Mảng trong AngularJS**

Ví dụ về mảng trong AngularJS.

<div ng-app="" ng-init="points=[1,15,30,45,60]">  
  
 <p>The third result is {{ points[2] }}</p>  
  
</div>

Kết quả sau khi chạy chương trình là:

The third result is 30

### AngularJS Modules

AngularJS Modules là cách tốt nhất để tổ chức mã nguồn. Nó được định nghĩa là một ứng dụng, nơi chứa các phần khác nhau của ứng dụng và là nơi chứa controller.

**Tạo một module**

Một module được tạo ra bằng cách sử dụng hàm **angular.module** và nó phải được đặt nằm trong cặp thẻ <script>.

<script>  
  
 var app = angular.module("myApp", []); </script>

**Thêm bộ điều khiển**

Một bộ điều khiển được thêm vào ứng dụng bằng lệnh **ng-controller**

<body>

<div ng-app="myApp" ng-controller="myCtrl">

{{ firstName + " " + lastName }}

</div>

<script>

var app = angular.module("myApp", []);

app.controller("myCtrl", function($scope) {

$scope.firstName = "Trung";

$scope.lastName = "Nghia";

});

</script>

</body>

Controller bên trong cặp thẻ <div> dùng để khai báo phần HTML bên trong nó sẽ sử dụng.

Controller bên trong cặp thẻ <script> dùng để tạo một controller với tên là "myCtrl", controller này sử dụng **$scope** để liên kết dữ liệu với HTML.

### AngularJS Directives

Directives cho phép mở rộng HTML cung cấp khả năng tùy chỉnh giao diện người dùng và tái sử dụng code. Các directives như **ng-model**, **ng-repeat** và **ng-show** giúp thực hiện các chức năng quan trọng như ràng buộc dữ liệu, lặp qua danh sách và điều kiển hiển thị.

Ví dụ về chỉ thị **ng-model**: lệnh này liên kết và lấy giá trị của **firstName** của **ng-init** để hiển thị giá trị ở **{{ firstName }} .**

<div ng-app="" ng-init="firstName='John'">  
 <p>Name: <input type="text" ng-model="firstName"></p>  
 <p>You wrote: {{ firstName }}</p>  
</div>

Ví dụ về chỉ thị **ng-repeat**

<div ng-app="" ng-init="names=[

{name:'Tran',country:'Cau Ngang'},

{name:'Trung',country:'Tieu Can'},

{name:'Nghia',country:'Tra Cu'}]">

//Sử dụng **ng-repeat** để lập qua các đối tượng ở trên để in kết quả ra màng hình.

<p>Looping with objects:</p>

<ul>

<li ng-repeat="i in names">

{{ i.name + ', ' + i.country }}</li>

</ul>

</div>

Kết quả sau khi sử dụng **ng-repeat** là.

Looping with objects:

Tran, Cau Ngang

Trung, Tieu Can

Nghia, Tra Cu

Chỉ thị **ng-app** xác định phần tử gốc của ứng dụng AngularJS. Lệnh này sẽ tự động khởi động ứng dụng khi một trang web được tải.

Chỉ thị **ng-init** xác định các giá trị ban đầu cho ứng dụng AngularJS.

Chỉ thị **ng-model** liên kết giá trị của HTML (đầu vào, chọn, vùng văn bản) với dữ liệu của ứng dụng. Chỉ thị này cũng có thể cung cấp xác thực loại cho dữ liệu ứng dụng (số, email, bắt buộc). Cung cấp trạng thái cho dữ liệu ứng dụng (không hợp lệ, bẩn, chạm, lỗi). Cung cấp các lớp CSS cho các phần tử HTML. Liên kết các phần tử HTML với các biểu mẫu HTML.

Tạo chỉ thị mới bằng cách sử dụng hàm **.directive**. Khi đặt tên cho một lệnh phải đặt tên theo kiểu lạc đà (tenBien) nhưng khi gọi nó phải sử dụng tên tách biệt (ten-bien).

Có thể gọi một lệnh bằng cách sử dụng

**element name**

<directive></directive>

**attribute**

<div directive></div>

**class**

<div class="directive"></div>

**comment**

<!-- directive: directive -->

### AngularJS Model

Model đại diện cho dữ liệu trong ứng dụng AngularJS. Nó giữ thông tin và cung cấp các phương thức để thao tác và cập nhật dữ liệu.

Với lệnh **ng-model** có thể liên kết giá trị của trường đầu vào với một biến được tạo trong AngularJS.

Ví dụ: Lệnh **ng-model**

<div ng-app="myApp" ng-controller="myCtrl">  
  Name: <input ng-model="name">  
</div>

<script>  
 var app = angular.module('myApp', []);  
 app.controller('myCtrl', function($scope) {  
   $scope.name = "Trung Nghia";  
 });  
</script>

Kết quả sẽ được hiểu thị như sau.



**Sự ràng buộc hai chiều**

Nếu người dùng thay đổi giá trị của trường đầu vào thì thuộc tính AngularJS cũng sẽ thay đổi theo giá trị của nó.

Giống như ví dụ ở trên nhưng khi tôi thay đổi giá trị của **$scope.name = "Trung Nghia"** thành **$scope.name = "Nghia Tran"** thì kết quả hiển thị cũng sẽ thay đổi theo vì nó bị ràng buộc.

Ví dụ: Thay đổi giá trị của **$scope.name = "Trung Nghia"** thành **$scope.name = "Nghia Tran" .**

<div ng-app="myApp" ng-controller="myCtrl">  
  Name: <input ng-model="name">  
</div>

<script>  
 var app = angular.module('myApp', []);  
 app.controller('myCtrl', function($scope) {  
   $scope.name = "Nghia Tran";  
 });  
</script>

Kết quả sẽ được hiểu thị như sau.



**Xác thực đầu vào của người dùng**

Lệnh **ng-model** cung cấp xác thực loại đầu vào cho dữ liệu ứng dụng (số, email, bắt buộc).

<form ng-app="" name="myForm">

Email:

<input type="email" name="myAddress" ng-model="text">

<span ng-show="myForm.myAddress.$error.email">Địa chỉ email không hợp lệ</span>

</form>

Kết quả trả về một ô input cho người dùng nhập email và AngularJS sẽ thông báo lỗi nếu như người dùng nhập email không hợp lệ.

**Trạng thái đơn đăng ký**

Lệnh **ng-model** cung cấp trạng thái cho dữ liệu ứng dụng (valid, dirty, touched).

Ví dụ: Một ô input để nhập email và các trạng thái hiển thị bên dưới.

<form ng-app="" name="myForm" ng-init="myText = 'trungnghia@myweb.com'">

Email:

<input type="email" name="myAddress" ng-model="myText" required>

<p>Sửa địa chỉ email và thử thay đổi trạng thái.</p>

<h1>Trạng thái</h1>

<p>Valid: {{myForm.myAddress.$valid}}</p>

<p>Dirty: {{myForm.myAddress.$dirty}}</p>

<p>Touched: {{myForm.myAddress.$touched}}</p>

</form>

Kết quả sẽ hiện thị ra một ô input cho người dùng nhập và trong khi nhập thì ba trạng thái trên sẽ liên tục thay đổi cho đến khi người dùng kết thúc việc nhập liệu. Sau đây là giải thích về ba trạng thái trên.

**{{myForm.myAddress.$valid}}**: Hiển thị trạng thái hợp lệ của trường nhập liệu. Nếu giá trị nhập vào là hợp lệ, biểu thức này sẽ trả về true hoặc ngược lại sẽ trả về false.

**{{myForm.myAddress.$dirty}}**: Hiển thị trạng thái "dirty" của trường nhập liệu. Một trường được coi là "dirty" nếu nó đã bị sửa đổi.

**{{myForm.myAddress.$touched}}**: Hiển thị trạng thái "touched" của trường nhập liệu. Một trường được coi là "touched" nếu nó đã được người dùng tương tác.

**Lớp CSS**

Lệnh ng-model cung cấp các lớp CSS cho các phần tử HTML tùy thuộc vào trạng thái của chúng.

Ví dụ: CSS cho ô input

<html>

<script

src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js">

</script>

<style>

input.ng-invalid {

background-color: lightblue;

}

</style>

<body>

<form ng-app="" name="myForm">

Enter your name:

<input name="myName" ng-model="myText" required>

</form>

</body>

</html>

Kết quả sẽ được hiển thị như sau.



Trạng thái ng-invalid áp dụng khi giá trị của trường nhập liệu không hợp lệ.

Một số trạng thái liên quan đến kiểm tra trường nhập liệu.

|  |  |
| --- | --- |
| ng-empty | Áp dụng khi giá trị của trường là trống (empty). |
| ng-not-empty | Áp dụng khi giá trị của trường không trống (khác empty). |
| ng-touched | Áp dụng khi trường đã được chạm vào (touched) bởi người dùng. Điều này xảy ra khi người dùng tương tác với trường nhập liệu, chẳng hạn như click vào nó. |
| ng-untouched | Áp dụng khi trường chưa được chạm vào (untouched). Nếu người dùng chưa tương tác với trường nhập liệu, nó sẽ ở trạng thái untouched. |
| ng-valid | Áp dụng khi giá trị của trường là hợp lệ theo các quy tắc xác định, chẳng hạn như định dạng email đúng. |
| ng-invalid | Áp dụng khi giá trị của trường không hợp lệ theo các quy tắc xác định. |
| ng-dirty | Áp dụng khi giá trị của trường đã được sửa đổi (dirty). Nếu người dùng thay đổi giá trị của trường nhập liệu, nó sẽ ở trạng thái dirty. |
| ng-pending | Áp dụng khi trường đang ở trạng thái đang chờ xác nhận hoặc đang trong quá trình xử lý. |

### AngularJS Data Binding

Data Binding là tính năng quan trọng của AngularJS, nó tự động đồng bộ hóa dữ liệu giữa mô hình và khung nhìn. Khi dữ liệu thay đổi thì giao diện người dùng cũng tự động được cập nhật.

**Data Model**

Các ứng dụng AngularJS thường có mô hình dữ liệu, mô hình dữ liệu là tập hợp dữ liệu có sẵn cho ứng dụng.

Ví dụ:

var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
  $scope.firstname = "Trung";  
  $scope.lastname = "Nghia";  
});

Trong AngularJS mô hình dữ liệu thường được sử dụng để lưu trữ và quản lý dữ liệu của ứng dụng. Trong ví dụ trên, mô hình dữ liệu được triển khai thông qua một controller và sử dụng biến $scope để chia sẻ dữ liệu giữa controller và view.

angular.module('myApp', []): Tạo một module có tên 'myApp'. Module này sẽ chứa tất cả các thành phần của ứng dụng như controllers, services, directives, và filters.

app.controller('myCtrl', function($scope) {...}): Tạo một controller có tên 'myCtrl' trong module 'myApp'. Controller này sử dụng biến $scope để chia sẻ dữ liệu với view liên kết với nó.

$scope.firstname và $scope.lastname: Định nghĩa một thuộc tính firstname và lastname trên $scope và gán giá trị ban đầu cho nó.

**HTML view**

Vùng chứa HTML nơi ứng dụng AngularJS được hiển thị được gọi là dạng xem. Chế độ xem có quyền truy cập vào mô hình và có một số cách hiển thị dữ liệu mô hình trong chế độ xem. Có thể sử dụng cặp ngoặc nhọn {{..}} để liên kết Internal HTML của phần tử với thuộc tính mô hình đã chỉ định.

Ví dụ: Chế độ xem

<body>

<div ng-app="myApp" ng-controller="myCtrl">

<p>First name: {{firstname}}</p>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

$scope.firstname = "Trung";

$scope.lastname = "Nghia";

});

</script>

</body>

Kết quả hiện thị ra màn hình là

First name: Trung

**Chỉ thị** ng-model

Sử dụng ng-model để liên kết dữ liệu từ mô hình với chế độ xem.

Ví dụ: Liên kết mô hình với chế độ xem

<body>

<div ng-app="myApp" ng-controller="myCtrl">

<p>First name: {{firstname}}</p>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

$scope.firstname = "Trung";

$scope.lastname = "Nghia";

});

</script>

</body>

Kết quả sau khi chạy chương trình là.

First name: Trung

Ở ví dụ trên thì ng-model đóng vai trò là trung gian để kết nối module với chế độ xem hiển thị ra màn hình thông qua "firstname" .

### AngularJS Controllers

Controller là bộ điều khiển trong mô hình MVC của AngularJS. Nó quản lý logic ứng dụng, xử lý sự kiện từ giao diện và cập nhật mô hình.

Các ứng dụng AngularJS được điều khiển bởi bộ điều khiển. Lệnh ng-controller giúp xác định bộ điều khiển của ứng dụng.

Thêm bộ điều khiển vào ứng dụng bằng lệnh ng-controller .

Ví dụ:

<div ng-app="myApp" ng-controller="myCtrl">  
 First Name: <input type="text" ng-model="firstName"><br>  
 Last Name: <input type="text" ng-model="lastName"><br>  
 Full Name: {{firstName + " " + lastName}}  
</div>  
  
<script>  
 var app = angular.module('myApp', []);  
 app.controller('myCtrl', function($scope) {  
  $scope.firstName = "Trung";  
   $scope.lastName = "Nghia";  
 });  
</script>

Kết quả sẽ được hiển thị như sau.

Full Name: Trung Nghia

ng-app="myApp": Xác định nơi bắt đầu của ứng dụng AngularJS và chỉ định tên module là myApp mà ứng dụng sẽ sử dụng.

ng-controller="myCtrl": Là một bộ điều khiển có tên là myCtrl mà AngularJS sẽ sử dụng trong phạm vi của div này. Bất cứ thứ gì bên trong div này sẽ được quản lý và kiểm soát bởi bộ điều khiển này.

AngularJS sẽ gọi bộ điều khiển bằng đối tượng $scope. Bộ điều khiển tạo hai thuộc tính trong $scope là firstName và lastName. Sau đó lệnh ng-module kết nối để hiện thị kết quả ra màn hình.

**Phương pháp điều khiển**

Một bộ điều khiển cũng có thể có các phương thức (các biến là hàm).

Ví dụ:

<div ng-app="myApp" ng-controller="personCtrl">  
 First Name: <input type="text" ng-model="firstName"><br>  
 Last Name: <input type="text" ng-model="lastN ame"><br>  
 Full Name: {{fullName()}}  
</div>  
  
<script>  
 var app = angular.module('myApp', []);  
 app.controller('personCtrl', function($scope) {  
  $scope.firstName = "Trung";  
   $scope.lastName = "Nghia";  
   $scope.fullName = function() {  
     return $scope.firstName + " " + $scope.lastName;  
   };  
 });  
</script>

Kết quả sẽ được hiển thị như sau:

Full Name: Trung Nghia

Ở ví dụ này chỉ khác với ví dụ trên ở chỗ phương thức fullName sẽ nằm trong bộ điều khiển thay vì nằm ở ngoài div.

Nếu fullName được định nghĩa trong controller nó chỉ có thể được truy cập và sử dụng trong phạm vi của controller đó. Để truy cập và sử dụng fullName từ bên ngoài controller hoặc div cần sử dụng các cách khác như sử dụng $scope. Có thể thêm fullName vào $scope bằng cách gán nó vào một phương thức khác trong controller và sau đó truy cập vào fullName từ bất kỳ nơi nào sử dụng $scope.fullName().

### AngularJS Scopes

Scope là một đối tượng trong AngularJS giúp quản lý việc liên kết giữa controller và xem, là phần ràng buộc giữa HTML (chế độ xem) và JavaScript (bộ điều khiển). Nó định rõ phạm vi của biến và hàm, giúp tránh xung đột và tạo điều kiện cho tính encapsulation (đóng gói).

Để sử dụng $scope thì khi tạo bộ điều khiển trong AngularJS, nên truyền đối tượng $scope làm đối số.

Ví dụ:

<script>

var app = angular.module('myApp', []);  
 app.controller('myCtrl', function($scope) {  
   $scope.myName = "Nghia";  
 });

</script>

Khi thêm thuộc tính vào đối tượng $scope trong bộ điều khiển, chế độ xem ng-model sẽ có quyền truy cập vào các thuộc tính này.

Trong chế độ xem không thể sử dụng tiền tố $scope, chỉ tham chiếu đến tên thuộc tính như {{myName}}.

**Phạm vi gốc**

Tất cả các ứng dụng đều có $rootScopephạm vi được tạo trên phần tử HTML chứa lệnh ng-app. RootScope có sẵn trong toàn bộ ứng dụng.

Nếu một biến có cùng tên trong cả phạm vi hiện tại và trong rootScope, thì ứng dụng sẽ sử dụng biến đó trong phạm vi hiện tại.

Ví dụ: Một biến có tên "color" tồn tại trong cả phạm vi của bộ điều khiển và trong rootScope.

<body ng-app="myApp">  
 <p>The rootScope's color:</p>  
 <h1>{{color}}</h1>

<div ng-controller="myCtrl">  
   <p>The scope of the controller's color:</p>  
   <h1>{{color}}</h1>  
 </div>  
  
 <p>The rootScope's color is still:</p>  
 <h1>{{color}}</h1>  
  
 <script>

var app = angular.module('myApp', []);  
 app.run(function($rootScope) {  
   $rootScope.color = 'blue';  
 });  
 app.controller('myCtrl', function($scope) {  
    $scope.color = "red";  
 });

</script>  
</body>

Trong ví dụ trên phương thức run có biến tên là color và phương thức của bộ điều khiển cũng có biến tên là color. Trong trường hợp này biến color được sử dụng trong phạm vi hiện tại của controller (myCtrl) sẽ được ưu tiên hơn biến cùng tên trong rootScope khi hiển thị giá trị.

### AngularJS Filters

Filters cho phép xử lý và định dạng dữ liệu trước khi hiển thị nó trên giao diện người dùng. Các filters như currency, date và uppercase giúp làm cho dữ liệu trở nên dễ đọc và hiểu.

AngularJS cung cấp các bộ lọc để chuyển đổi dữ liệu:

|  |  |
| --- | --- |
| currency | Định dạng số thành định dạng tiền tệ. |
| date | Định dạng ngày theo định dạng được chỉ định. |
| filter | Chọn một tập hợp con các mục từ một mảng. |
| json | Định dạng một đối tượng thành chuỗi JSON. |
| limitTo | Giới hạn một mảng/chuỗi thành một số phần tử/ký tự được chỉ định. |
| lowercase | Định dạng chuỗi thành chữ thường. |
| number | Định dạng số thành chuỗi. |
| orderBy | Sắp xếp một mảng theo một biểu thức. |
| uppercase | Định dạng một chuỗi thành chữ hoa. |

**Thêm bộ lọc vào biểu thức**

Bộ lọc được thêm vào biểu thức bằng cách sử dụng ký tự | rồi đến bộ lọc.

Ví dụ: uppercase yourName thành chữ hoa.

<div ng-app="myApp" ng-controller="personCtrl"

<p>The name is {{ lastName | uppercase }}</p>

</div>

<script>

angular.module('myApp', []).controller('personCtrl', function($scope) {

$scope.firstName = "Trung",

$scope.lastName = "Nghia"

});

</script>

Kết quả sau khi sử dụng bộ lọc uppercase là:

The name is NGHIA

**Thêm bộ lọc vào chỉ thị**

Thêm bộ lọc vào chỉ thị tương tự như thêm vào biểu thức, sử dụng ký tự | rồi đến bộ lọc.

Ví dụ: orderBy sắp xếp một mảng.

<div ng-app="myApp" ng-controller="namesCtrl">

<p>Looping with objects:</p>

<ul>

<li ng-repeat="x in names | orderBy:'country'">

{{ x.name + ', ' + x.country }}

</li>

</ul>

</div>

<script>

angular.module('myApp', []).controller('namesCtrl', function($scope) {

$scope.names = [

{name:'Tran',country:'Tra Vinh'},

{name:'Trung',country:'HCM'},

{name:'Nghia',country:'Ha Noi'},

];

});

</script>

Kết quả sau khi sử dụng bộ lọc orderBy là.

Looping with objects:

Nghia, Ha Noi

Trung, HCM

Tran, Tra Vinh

### AngularJS Services

AngularJS Services cung cấp các dịch vụ có thể được chia sẻ giữa các thành phần trong ứng dụng. Điều này giúp tái sử dụng mã nguồn và tăng sự linh hoạt.

Dịch vụ $location có các phương thức trả về thông tin về vị trí của trang web hiện tại.

Ví dụ: sử dụng $location trong bộ điều khiển.

<div ng-app="myApp" ng-controller="myCtrl">

<p>The url of this page is:</p>

<h3>{{myUrl}}</h3>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope, $location) {

$scope.myUrl = $location.absUrl();

});

</script>

Kết quả trả về như sau:

The url of this page is:

https://www.w3schools.com/angular/tryit.asp?filename=try\_ng\_services

Đối tượng $location là một dịch vụ trong AngularJS được sử dụng để làm việc với URL. Nó cung cấp các phương thức và thuộc tính để truy cập và thao tác đường dẫn URL.

$location.absUrl(): Đây là một phương thức của $location được sử dụng để trả về đường dẫn URL tuyệt đối (absolute URL) của trang hiện tại.

Lưu ý rằng dịch vụ $location được chuyển đến bộ điều khiển dưới dạng đối số. Để sử dụng dịch vụ trong bộ điều khiển, nó phải được xác định là một phần phụ thuộc.

### AngularJS Http

Dịch vụ $http là một trong những dịch vụ được sử dụng phổ biến nhất trong các ứng dụng AngularJS, nó đưa ra yêu cầu tới máy chủ và cho phép ứng dụng xử lý phản hồi, giúp tương tác với các API thông qua HTTP. Nó cung cấp các phương thức như get, post để gửi và nhận dữ liệu từ máy chủ.

Ví dụ: Sử dụng $http để yêu cầu dữ liệu từ máy chủ.

Giả sử tệp tin "welcome.htm" chứa một chuỗi văn bản: "Welcome To AngularJS"

<div ng-app="myApp" ng-controller="myCtrl">

<p>Welcome</p

<h1>{{myWelcome}}</h1>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope, $http) {

$http.get("welcome.htm").then(function (response) {

$scope.myWelcome = response.data;

});

});

</script>

Kết quả trả về như sau:

Welcome

Welcome To AngularJS

$http là một dịch vụ trong AngularJS được sử dụng để thực hiện các yêu cầu HTTP. Phương thức $http.get("welcome.htm") là phương thức dùng để gửi một yêu cầu HTTP GET đến tệp tin "welcome.htm" để lấy nội dung.

Lưu ý rằng để thực thi đúng, tệp tin "welcome.htm" phải tồn tại trên máy chủ và được phản hồi bởi yêu cầu HTTP. Nếu không, biến myWelcome sẽ không có giá trị và không có gì được hiển thị trong phần tử <h1>.

Phản hồi từ máy chủ là một đối tượng có các thuộc tính sau:

|  |  |
| --- | --- |
| Thuộc tính | Ý nghĩa |
| .config | đối tượng được sử dụng để tạo ra yêu cầu. |
| .data | một chuỗi hoặc một đối tượng mang phản hồi từ máy chủ. |
| .headers | một chức năng được sử dụng để lấy thông tin tiêu đề. |
| .status | một số xác định trạng thái HTTP. |
| .statusText | một chuỗi xác định trạng thái HTTP. |

**JSON**

Dữ liệu nhận được từ phản hồi ​​sẽ ở định dạng JSON.

Ví dụ: định dạng JSON.

[

{

"id": "1",

"ten": "iphone15",

"gia": "25000000",

"mota": "Dien thoai iphone",

"hangxs\_id": "1"

}

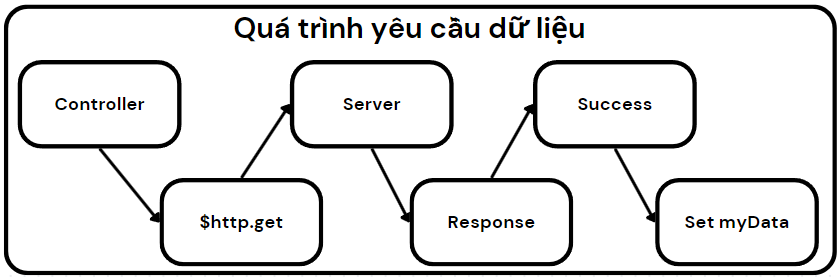
]

Ứng dụng xác định bộ điều khiển với đối tượng $scope và $http.

$http là một đối tượng XMLHttpRequest để yêu cầu dữ liệu ngoài.

Phương thức $http.get() đọc dữ liệu JSON từ một tập tin PHP (ví dụ: https://localhost/namefile/namefile.php).

Khi thành công, bộ điều khiển sẽ tạo một thuộc tính myData với dữ liệu có dạng JSON từ máy chủ.



Hình 2.1 Lưu đồ quá trình yêu cầu dữ liệu và cách dữ liệu được trả về và xử lý.

Controller: Đây là phần quản lý logic của ứng dụng AngularJS. Nó sẽ tạo một yêu cầu $http.get để lấy dữ liệu từ máy chủ.

$http.get: Đối tượng $http trong AngularJS sẽ gửi một yêu cầu Get đến máy chủ để lấy dữ liệu từ tập tin PHP được chỉ định.

Server: Đây là máy chủ nhận yêu cầu từ ứng dụng và xử lý nó. Trong trường hợp này, nó sẽ đọc dữ liệu từ tệp PHP và trả về cho ứng dụng.

Response: Đây là phản hồi từ máy chủ chứa dữ liệu được trả về từ tệp PHP.

Success: Nếu yêu cầu thành công và dữ liệu được trả về từ máy chủ, quá trình tiếp tục vào phần thành công.

Set myData: Trong phần thành công, bộ điều khiển sẽ tạo một thuộc tính myData và gán dữ liệu JSON từ máy chủ vào đó.

### AngularJS Tables

AngularJS Tables giúp tạo và quản lý các bảng dữ liệu trên giao diện người dùng. Nó tối ưu hóa quá trình hiển thị và tương tác với dữ liệu dạng bảng thông qua thuộc tính cho phép lặp ng-repeat.

Ví dụ: "thongtin.php" có JSON

{

"info": [

{

"Name": "Tran Trung Nghia",

"Country": "Viet Nam"

},

{

"Name": "Nghia Tran Trung",

"Country": "Portugal"

}

]

}

<body

<div ng-app="myApp" ng-controller="customersCtrl">

<table>

<tr ng-repeat="x in names">

<td>{{ x.Name }}</td>

<td>{{ x.Country }}</td>

</tr>

</table>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('customersCtrl', function($scope, $http) {

$http.get("customers.php")

.then(function (response) {$scope.names = response.data.info;});

});

</script>

</body>

Kết quả sẽ được hiển thì như sau.

Tran Trung Nghia Viet Nam

Nghia Tran Trung Portugal

### AngularJS Select

Select trong AngularJS là một thành phần quan trọng cho việc chọn lựa dữ liệu. Nó kết hợp với ng-model để giúp tự động cập nhật dữ liệu khi người dùng thay đổi lựa chọn.

Nếu muốn tạo danh sách thả xuống dựa trên một đối tượng hoặc một mảng trong AngularJS nên sử dụng lệnh ng-options.

Ví dụ: tạo một options trong AngularJS.

<body>

<div ng-app="myApp" ng-controller="myCtrl">

<select ng-model="selectedName" ng-options="x for x in names"></select>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

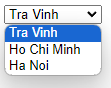
$scope.names = ["Tra Vinh", "Ho Chi Minh", "Ha Noi"];

});

</script>

</body>

Kết quả sẽ được hiển thị như sau.



Trong phần tử <select> thuộc tính ng-model="selectedName" được sử dụng để liên kết giá trị đã chọn trong menu dropdown với biến $scope.selectedName trong controller. Khi người dùng chọn một tên thành phố từ menu, giá trị tương ứng sẽ được gán vào biến $scope.selectedName.

Trong thuộc tính ng-options biểu thức "x for x in names"được sử dụng để tạo các tuỳ chọn trong menu dropdown từ mảng names được khai báo trong controller. Điều này có nghĩa là mỗi tên thành phố trong mảng names sẽ được hiển thị trong menu dropdown.

Tương tự, lệnh ng-repeat được sử dụng để tạo danh sách thả xuống tương tự như dropdown.

Ví dụ: tương tự với lệnh repeat.

<body>

<div ng-app="myApp" ng-controller="myCtrl">

<select><option ng-repeat="x in names">{{x}}</option></select>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

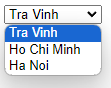
$scope.names = ["Emil", "Tobias", "Linus"];

});

</script>

</body>

Kết quả cũng tương tự như kết quả của dropdown.



### AngularJS SQL

AngularJS không tương tác trực tiếp với SQL nhưng có thể sử dụng để làm việc với dữ liệu từ máy chủ thông qua các API trong đó có thể có việc tương tác với cơ sở dữ liệu SQL.

AngularJS được dùng để hiển thị dữ liệu từ cơ sở dữ liệu. Chỉ cần đảm bảo dữ liệu ở định dạng JSON.

Ví dụ: có file dạng JSON như sau

<https://www.w3schools.com/js/Customers_MYSQL.php>

Dùng AngularJS để hiển thị dữ liệu MySQL thông qua tệp customers\_mysql.php

<body>

<div ng-app="myApp" ng-controller="customersCtrl">

<table>

<tr ng-repeat="x in names">

<td>{{ x.Name }}</td>

<td>{{ x.Country }}</td>

</tr>

</table>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('customersCtrl', function($scope, $http) {

$http.get("customers\_mysql.php")

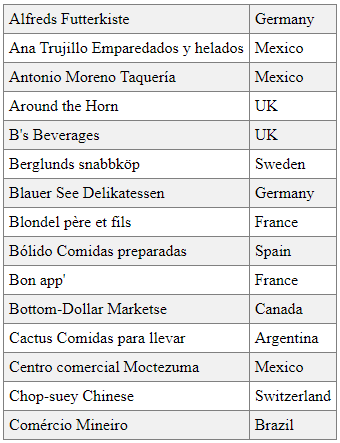
.then(function (response) {$scope.names = response.data.records;});

});

</script>

</body

Kết quả sẽ được hiển thị như sau:



Dùng AngularJS để kết nối và truy vấn dữ liệu từ tệp "customers\_mysql.php" bằng cách sử dụng $http.get(). Khi dữ liệu được trả về, nó được gán vào một biến $scope.names, các trường dữ liệu "Name" và "Country" được hiển thị trong bảng bằng cách sử dụng cú pháp {{ x.Name }} và {{ x.Country }}, vòng lặp sẽ lặp lại các hàng trong mảng names và hiển thị dữ liệu tương ứng trong các ô <td>.

**Mã máy chủ PHP và MySQL**

<?php  
header("Access-Control-Allow-Origin: \*");  
header("Content-Type: application/json; charset=UTF-8");  
$conn = new mysqli("myServer", "myUser", "myPassword", "Northwind");  
$result = $conn->query("SELECT CompanyName, City, Country FROM Customers");  
$outp = "";  
while($rs = $result->fetch\_array(MYSQLI\_ASSOC)) {  
  if ($outp != "") {$outp .= ",";}  
  $outp .= '{"Name":"'  . $rs["CompanyName"] . '",';  
  $outp .= '"City":"'   . $rs["City"]        . '",';  
  $outp .= '"Country":"'. $rs["Country"]     . '"}';  
}  
$outp ='{"records":['.$outp.']}';  
$conn->close();  
echo($outp);  
?>

Đoạn mã PHP ở trên là một tệp "customers\_mysql.php", tệp này thực hiện truy vấn CSDL và trả về dữ liệu JSON cho trang web AngularJS để hiển thị trong bảng. Đầu tiên, các dòng mã PHP đầu tiên thiết lập tiêu đề HTTP cho phép truy cập qua nguồn gốc khác (CORS) và đặt loại nội dung là JSON. Tiếp theo, kết nối đến cơ sở dữ liệu MySQL bằng cách sử dụng các thông số như tên máy chủ ("myServer"), tên người dùng ("myUser"), mật khẩu ("myPassword"), và tên cơ sở dữ liệu ("Northwind"). Sau đó, một câu truy vấn SQL được thực hiện để lấy dữ liệu từ bảng "Customers" trong cơ sở dữ liệu. Kết quả truy vấn được lặp lại thông qua vòng lặp while và mỗi hàng dữ liệu được biến đổi thành một đối tượng JSON. Tiếp theo, dữ liệu JSON được xây dựng bằng cách ghép các đối tượng JSON lại với nhau và được gán vào biến $outp. Mỗi đối tượng JSON trong mảng records sẽ chứa các trường dữ liệu như "Name", "City" và "Country" được lấy từ kết quả truy vấn. Cuối cùng, kết nối đến cơ sở dữ liệu được đóng và chuỗi JSON được gửi về cho trình duyệt thông qua hàm echo().

### AngularJS DOM

Module ng-dom giúp quản lý và tương tác với DOM (Document Object Model). Nó cung cấp các phương thức để thao tác với các phần tử trên trang web.

AngularJS có các chỉ thị để liên kết dữ liệu ứng dụng với các thuộc tính của các phần tử HTML DOM.

Lệnh ng-disabled liên kết dữ liệu của AngularJS với thuộc tính bị vô hiệu hóa của các phần tử HTML.

Ví dụ:

<body>

<div ng-app="" ng-init="mySwitch=true">

<p><button ng-disabled="mySwitch">Click Me!</button</p>

<p><input type="checkbox" ng-model="mySwitch"/>Button</p>

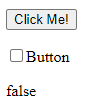
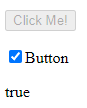
<p>{{ mySwitch }</p>

</div>

</body>

Kết quả được hiện thị như sau.

Khi không chọn và khi chọn

Thuộc tính ng-init="mySwitch=true" được sử dụng để khởi tạo biến mySwitch và đặt giá trị ban đầu là true.

Thuộc tính ng-disabled="mySwitch" được sử dụng để điều khiển trạng thái bật/tắt của nút dựa trên giá trị của biến mySwitch. Khi mySwitch là true, nút sẽ bị vô hiệu hóa (disable). Ngược lại, khi mySwitch là false, nút sẽ có thể được nhấp vào.

Phần tử <p> chứa một hộp kiểm <input type="checkbox">. Thuộc tính ng-model="mySwitch" được sử dụng để gắn kết giá trị của checkbox với biến mySwitch. Khi checkbox được chọn, giá trị của mySwitch sẽ là true. Ngược lại, khi checkbox không được chọn giá trị của mySwitch sẽ là false.

Phần tử <p> để hiển thị giá trị của biến mySwitch bằng cách sử dụng cú pháp {{ mySwitch }}.

**Chỉ thị ng-show**

Lệnh ng-show hiển thị hoặc ẩn một phần tử HTML dựa trên giá trị của ng-show.

Ví dụ:

<body>

<div ng-app=""

<p ng-show="true">I am visible.</p>

<p ng-show="false">I am not visible.</p>

</div>

</body>

Nếu giá trị của ng-show="false" thì kết quả sẽ không được hiển thị lên.

Nếu giá trị của ng-show="true" thì kết quả sẽ được hiển thị lên.

Chỉ thị ng-hide

Lệnh ng-hide ẩn hoặc hiển thị phần tử HTML. Tương tự như ng-show.

### AngularJS Events

Xử lý sự kiện là một phần quan trọng của phát triển web và AngularJS Events giúp quản lý và xử lý sự kiện từ người dùng.

Ta có thể thêm trình xử lý sự kiện AngularJS vào các phần tử HTML bằng cách sử dụng một hoặc nhiều lệnh sau:

|  |  |  |
| --- | --- | --- |
| ng-blur | ng-focus | ng-mouseleave |
| ng-change | ng-keydown | ng-mousemove |
| ng-click | ng-keypress | ng-mouseover |
| ng-copy | ng-keyup | ng-mouseup |
| ng-cut | ng-mousedown | ng-paste |
| ng-dblclick | ng-mouseenter |

**Chỉ thị ng-click**

Lệnh ng-click xác định mã AngularJS sẽ được thực thi khi phần tử được nhấp vào.

Ví dụ:

<body>

<div ng-app="myApp" ng-controller="myCtrl">

<button ng-click="count = count + 1">Click Me!</button>

<p>{{ count }}</p>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

$scope.count = 0;

});

</script>

</body>

Kết quả sẽ được hiển thị như sau.



Số 0 sẽ tăng dần lên khi ta click chuột vào nó, mỗi lần sẽ tăng 1 đơn vị.

### AngularJS Forms

AngularJS Forms giúp quản lý và kiểm soát các biểu mẫu trên giao diện người dùng. Nó cung cấp các directives và controllers để thực hiện các chức năng như validation và submit.

**Liên kết dữ liệu**

Điều khiển đầu vào cung cấp liên kết dữ liệu bằng cách sử dụng ng-model.

Ví dụ:

<body>

<div ng-app="myApp" ng-controller="formCtrl">

<form>

First Name: <input type="text" ng-model="firstname">

</form>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('formCtrl', function($scope) {

$scope.firstname = "Nghia";

});

</script>

</body>

Kết quả sẽ được hiển thị như sau.



### AngularJS Validation

Validation trong AngularJS giúp kiểm tra và đảm bảo tính đúng đắn của dữ liệu nhập vào từ người dùng. Các directives như ng-required và ng-pattern hỗ trợ quá trình này.

AngularJS cung cấp xác thực biểu mẫu phía máy khách.

AngularJS cũng chứa thông tin về việc chúng có được chạm vào và có được sửa đổi hay không.

**Các trường đầu vào có các trạng thái sau**

|  |  |
| --- | --- |
| Trạng thái | Ý nghĩa |
| $untouched | Nội dung vẫn chưa được chạm tới |
| $touched | Nội dung đã được chạm vào |
| $pristine | Nội dung vẫn chưa được sửa đổi |
| $dirty | Nội dung đã được sửa đổi |
| $invalid | Nội dung trường không hợp lệ |
| $valid | Nội dung trường hợp lệ |

**Các biểu mẫu có các trạng thái sau**

|  |  |
| --- | --- |
| $pristine | Chưa có trường nào được sửa đổi |
| $dirty | Một hoặc nhiều đã được sửa đổi |
| $invalid | Nội dung biểu mẫu không hợp lệ |
| $valid | Nội dung biểu mẫu hợp lệ |
| $submitted | Mẫu đơn được gửi |

Tất cả chúng đều là các thuộc tính có hai giá trị là true hoặc false.

### AngularJS API

API là viết tắt của Giao diện lập trình ứng dụng.

AngularJS API là tập hợp các phương thức và đối tượng mà AngularJS cung cấp để tương tác và quản lý ứng dụng.

Một số IPA phổ biến.

|  |  |
| --- | --- |
| angular.lowercase() | Chuyển đổi một chuỗi thành chữ thường. |
| angular.uppercase() | Chuyển đổi một chuỗi thành chữ hoa. |
| angular.isString() | Trả về true nếu tham chiếu là một chuỗi. |
| angular.isNumber() | Trả về true nếu tham chiếu là số. |

Ví dụ: về angular.lowercase()

<body>

<div ng-app="myApp" ng-controller="myCtrl">

<p>{{ x1 }}</p>

<p>{{ x2 }}</p>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

$scope.x1 = "NGHIA";

$scope.x2 = angular.lowercase($scope.x1);

});

</script>

</body>

Kết quả sẽ được hiển thị như sau.

NGHIA

nghia

Sử dụng hàm angular.lowercase() để chuyển đổi giá trị của x1 thành chữ thường và gán vào biến x2.

### AngularJS W3.CSS

AngularJS có thể kết hợp với W3.CSS, một bộ framework CSS để tạo giao diện người dùng đẹp mắt và responsive. Để nhúng W3.CSS vào AngularJS hãy thêm dòng sau vào phần header:

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

### AngularJS Includes

Includes trong AngularJS cho phép tái sử dụng mã nguồn, tạo điều kiện cho sự lặp lại trong mã nguồn.

Với AngularJS có thể bao gồm nội dung HTML bằng lệnh ng-include.

Ví dụ: có một file index.html có nội dung là "Hello các bạn"

<body ng-app="">  
 <div ng-include="index.htm'"></div>  
</body>

Kết quả sẽ được hiển thị như sau.

Hello các bạn

Các tệp HTML đưa vào bằng lệnh ng-include cũng có thể chứa mã AngularJS.

### AngularJS Animations

Module ng-animate giúp thêm các hiệu ứng và animations vào giao diện.

Animations là khi việc chuyển đổi một phần tử HTML mang lại cho người dùng ảo giác về chuyển động.

Để làm cho ứng dụng có hoạt ảnh thì phải bao gồm thư viện AngularJS Animate.

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular-animate.js"></script>

Nếu ứng dụng có tên, hãy thêm ngAnimate làm phần phụ thuộc trong mô-đun ứng dụng của bạn.

<body ng-app="myApp">

<h1>Hide the DIV: <input type="checkbox" ng-model="myCheck"></h1>

<div ng-hide="myCheck"></div>

<script>

var app = angular.module('myApp', ['ngAnimate']);

</script>

</body>

### AngularJS Routing

AngularJS Routing là một phần quan trọng giúp quản lý và định hình các trang trong ứng dụng. Nó cho phép xác định các route và liên kết chúng với các controllers và templates tương ứng, tạo điều kiện cho việc chuyển đổi giữa các trang một cách mượt mà.

Module ngRoute giúp ứng dụng trở thành ứng dụng một trang.

Nếu muốn điều hướng đến các trang khác nhau trong ứng dụng của mình nhưng cũng muốn ứng dụng đó là SPA không cần tải lại trang, ta có thể sử dụng module ngRoute. ngRoute định tuyến ứng dụng đến các trang khác nhau mà không cần tải lại toàn bộ ứng dụng.

Để cho ứng dụng định tuyến sẳn sang phải thêm thư viện này vào phần header.

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular-route.js"></script>

Và sau đó phải thêm phần ngRoute phụ thuộc vào mô-đun của ứng dụng.

var app = angular.module("myApp", ["ngRoute"]);

Sử dụng $routeProvider để định cấu hình các tuyến khác nhau trong ứng dụng. Kết quả là khi chuyển trang khác thì không cần tải lại trang.

Ứng dụng cần một vùng chứa để đặt nội dung do định tuyến cung cấp.

Vùng chứa này là ng-view.

Có ba cách khác nhau để đưa ng-view vào ứng dụng .

Cách 1

<div ng-view></div>

Cách 2

<ng-view></ng-view>

Cách 3

<div class="ng-view"></div>

## PHP & MySQL

### PHP Connect to MySQL

**Mở kết nối tới MySQL**

Trước khi có thể truy cập dữ liệu trong cơ sở dữ liệu MySQL cần phải kết nối với máy chủ.

Kết nối với MySQL bằng PHP.

<?php  
$servername = "localhost";  
$username = "username";  
$password = "password";  
  
// Create connection  
$conn = new mysqli($servername, $username, $password);  
  
// Check connection  
if ($conn->connect\_error) {  
  die("Connection failed: " . $conn->connect\_error);  
}  
echo "Connected successfully";  
?>

Đoạn mã trên được dùng để tạo kết nối với MySQL bằng MySQLi trong PHP.

Các biến $servername, $username và $password được khai báo để lưu trữ thông tin kết nối tới cơ sở dữ liệu.

Hàm new mysqli() được gọi với các tham số là $servername, $username và $password để tạo ra một đối tượng kết nối MySQL mới và được lưu trong biến $conn.

Hàm $conn->connect\_error được sử dụng để kiểm tra xem kết nối đã thành công hay không. Nếu lỗi thì thì connect\_error trả về chuổi mô tả lỗi, nếu thành công thì Connected successfully sẽ được hiển thị thông qua lệnh echo.

**Đóng kết nối**

Kết nối sẽ tự động bị đóng khi tập lệnh kết thúc. Để đóng kết nối trước hãy sử dụng như sau.

$conn->close();

### PHP Create a MySQL Database

Câu lệnh CREATE DATABASE được sử dụng để tạo cơ sở dữ liệu trong MySQL.

Ví dụ: tạo cơ sở dữ liệu có tên "myDB".

<?php  
$servername = "localhost";  
$username = "username";  
$password = "password";  
  
// Create connection  
$conn = new mysqli($servername, $username, $password);  
// Check connection  
if ($conn->connect\_error) {  
  die("Connection failed: " . $conn->connect\_error);  
}  
  
// Create database  
$sql = "CREATE DATABASE myDB";  
if ($conn->query($sql) === TRUE) {  
  echo "Database created successfully";  
} else {  
  echo "Error creating database: " . $conn->error;  
}  
  
$conn->close();  
?>

Đoạn mã PHP trên được sử dụng để tạo một cơ sở dữ liệu MySQL mới trong hệ thống. Câu truy vấn "CREATE DATABASE myDB" được khai báo trong biến $sql. Câu truy vấn được thực thi bằng $conn->query($sql), nếu truy vấn thành công thì phương thức query trả về true và "Database created successfully" sẽ được hiển thị. Nếu lỗi thì phương thức query sẽ trả về false và báo lỗi chi tiết thông qua echo "Error creating database: " . $conn->error.

### PHP MySQL Create Table

Câu lệnh CREATE TABLE được sử dụng để tạo bảng trong MySQL.

Tôi sẽ tạo một bảng có tên "MyGuests" với năm cột: "id", "firstname", "lastname", "email" và "reg\_date".

CREATE TABLE MyGuests (  
id INT(6) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,  
firstname VARCHAR(30) NOT NULL,  
lastname VARCHAR(30) NOT NULL,  
email VARCHAR(50),  
reg\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP  
)

Câu truy vấn trên tạo ra một bảng "MyGuests" với các cột id, firstname, lastname, email và reg\_date, định nghĩa các ràng buộc và thuộc tính cho từng cột.

### PHP MySQL Insert Data

Sau khi cơ sở dữ liệu và bảng được tạo, ta có thể bắt đầu thêm dữ liệu vào.

Câu lệnh INSERT INTO được sử dụng để thêm bản ghi mới vào bảng MySQL.

INSERT INTO MyGuests (firstname, lastname, email)  
VALUES ('Trung', 'Nghia', 'trungnghia@gmail.com')

Sau khi câu truy vấn được thực thi nếu thành công thì các giá trị tương ứng sẽ được chèn vào "MyGuests".

### PHP MySQL Get Last Inserted ID

Nếu thực hiện chèn hoặc cập nhật trên bảng có trường AUTO\_INCREMENT, chúng tôi có thể nhận được ID của bản ghi được chèn/cập nhật cuối cùng ngay lập tức.

Trong bảng "MyGuests" cột "id" là trường AUTO\_INCREMENT.

CREATE TABLE MyGuests (  
id INT(6) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,  
firstname VARCHAR(30) NOT NULL,  
lastname VARCHAR(30) NOT NULL,  
email VARCHAR(50),  
reg\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP  
)

### PHP MySQL Select Data

Câu lệnh SELECT được sử dụng để chọn dữ liệu từ một hoặc nhiều bảng.

SELECT id, firstname, lastname FROM MyGuests

Hoặc có thể sử dụng ký tự \* để chọn tất cả các cột từ một bảng.

SELECT \* FROM MyGuests

### PHP MySQL Use The WHERE Clause

Mệnh đề WHERE được sử dụng để lọc các bản ghi. Mệnh đề WHERE được sử dụng để chỉ trích xuất những bản ghi đáp ứng một điều kiện đã chỉ định.

SELECT id, firstname, lastname FROM MyGuests WHERE lastname='Nghia'

### PHP MySQL Use The ORDER BY Clause

Mệnh đề ORDER BY được sử dụng để sắp xếp tập kết quả theo thứ tự tăng dần hoặc giảm dần. Mệnh đề ORDER BY mặc định sắp xếp các bản ghi theo thứ tự tăng dần. Để sắp xếp các bản ghi theo thứ tự giảm dần hãy sử dụng từ khóa DESC.

SELECT id, firstname, lastname FROM MyGuests ORDER BY lastname

### PHP MySQL Delete Data

Câu lệnh DELETE được sử dụng để xóa các bản ghi khỏi bảng.

DELETE FROM MyGuests WHERE id=1

### PHP MySQL Update Data

Câu lệnh UPDATE được sử dụng để cập nhật các bản ghi hiện có trong bảng.

UPDATE MyGuests SET lastname='Nghia' WHERE id=1

## Kết luận

AngularJS là một framework JavaScript của Google giúp xây dựng ứng dụng web động. Nó sử dụng mô hình MVC để quản lý dữ liệu và tương tác với người dùng, kết hợp với khả năng liên kết dữ liệu giúp tối ưu hóa hiệu suất phát triển. PHP là một ngôn ngữ lập trình phía server mạnh mẽ thường được sử dụng để xử lý logic back-end trong ứng dụng web. MySQL là hệ quản trị cơ sở dữ liệu phổ biến tương thích với PHP giúp lưu trữ và truy xuất dữ liệu hiệu quả. Sự kết hợp giữa AngularJS, PHP và MySQL có thể xây dựng một ứng dụng web động, kết nối giữa front-end và back-end một cách linh hoạt để mang lại trải nghiệm người dùng tốt và quản lý dữ liệu hiệu quả trên nền tảng web.

# HIỆN THỰC HÓA NGHIÊN CỨU

(Mô tả các bước nghiên cứu đã tiến hành, các bản thiết kế, cách thức cài đặt chương trình hoặc hiện thực hóa nghiên cứu. Đối với các đề tài ứng dụng có kết quả là sản phẩm phần mềm phải có hồ sơ thiết kế, cài đặt, ... theo các dạng lược đồ, mô hình phổ biến trong ngành. Nội dung đặc tả nhu cầu, phân tích thiết kế hệ thống cũng thể hiện trong chương này.)

## Mô tả bài toán

Về mục tiêu sẽxây dựng một trang web giới thiệu về điện thoại di động sử dụng AngularJS để tạo giao diện người dùng tương tác. Trang web này sẽ cho phép người dùng xem các sản phẩm điện thoại di động và thực hiện được một số thao tác như xem sản phẩm theo hãng, lọc theo giá, tìm kiếm và đăng nhập. Về giao diện người dùng sẽ có: trang chủ hiển thị danh sách các sản phẩm điện thoại mới nhất của cửa hàng, danh mục hiển thị danh sách sản phẩm theo từng hãng, sản phẩm chi tiết hiển thị thông tin chi tiết của một sản phẩm, chức năng lọc theo giá và hiển thị các sản phẩm theo tầm giá đã chọn, chức năng tìm kiếm và hiển thị các sản phẩm theo yêu cầu tìm kiếm, chức năng đăng nhập cho người dùng đăng nhập vào tài khoảng người quản trị. Về giao diện người quản trị sẽ có các chức năng: thêm thương hiệu sẽ hiện thị thương hiệu đã thêm có thể thêm xóa sửa, thêm sản phẩm sẽ hiển thị sản phẩm đã thêm có thể thêm xóa sửa, thêm người dùng sẽ hiển thị người dùng đã thêm có thể thêm xóa sửa, thống kê sản phẩm để thống kê số lượng sản phẩm đang có của từng hãng, đăng xuất sẽ đăng xuất khỏi trang người quản trị và trở về trang chủ. Kết nối với API để lấy dữ liệu sản phẩm bằng PHP và sữ dụng AngularJS để hiển thị sản phẩm lên trang web.Đặc tả yêu cầu hệ thống

### Yêu cầu chức năng

**Hiển thị danh sách và chi tiết sản phẩm**: Chức năng là hiển thị danh sách các sản phẩm và cho phép người dùng xem chi tiết từng sản phẩm.

**Tìm kiếm và lọc sản phẩm:** Chức năng là người dùng có thể tìm kiếm sản phẩm theo tên và lọc sản phẩm theo giá.

**Đăng nhập:** Chức năng là đăng nhập vào tài khoảng người quản trị.

**Thêm và quản lý sản phẩm (Chỉ Admin):** Chức năng là Admin có thể thêm xóa sửa sản phẩm, hãng, người dùng và thống kê sản phẩm.

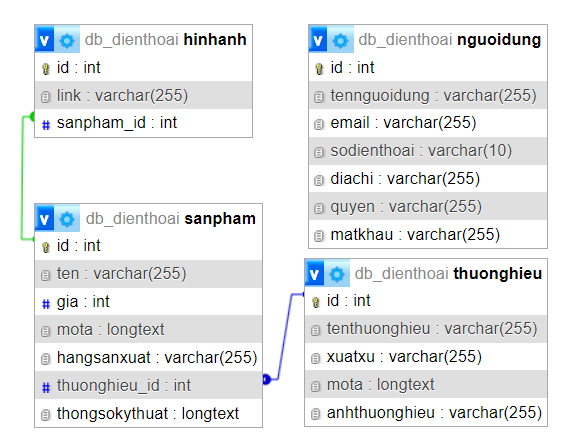
### Yêu cầu phi chức năng

**Giao diện người dùng thân thiện:** Yêu cầu một giao diện người dùng dễ sử dụng giúp người dùng dễ dàng tìm kiếm, lọc và xem thông tin chi tiết của sản phẩm.

**Hiệu suất và tốc độ:** Đảm bảo trang web hoạt động mượt mà và có thời gian phản hồi nhanh nhằm cung cấp trải nghiệm người dùng tốt nhất có thể.

## **Kiểm thử đầy đủ**: Tiến hành kiểm thử để đảm bảo tính ổn định và xác nhận rằng tất cả các chức năng hoạt động đúng như mong đợi.Thiết kế dữ liệu

### Lược đồ cơ sở dữ liệu



Hình 3.1 Cơ sở dữ liệu

Bảng 3.1: Danh sách các thực thể

|  |  |  |
| --- | --- | --- |
| *STT* | *Tên thực thể* | *Diễn giải* |
| 1 | sanpham | Sản phẩm |
| 2 | hinhanh | Hình ảnh |
| 3 | thuonghieu | Thương hiệu |
| 4 | nguoidung | Người dùng |

### Chi tiết các thực thể

Bảng 3.2 Chi tiết thực thể “sanpham”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *STT* | *Thuộc tính* | *Diễn giải* | *Kiểu dữ liệu* | *Ràng buộc toàn vẹn* |
| 1 | id | ID điện thoại | integer | Khóa chính |
| 2 | ten | Tên điện thoại | varchar |  |
| 3 | gia | Giá bán điện thoại | integer |  |
| 4 | mota | Mô tả về điện thoại | longtext |  |
| 5 | hangsanxuat | Hãng sản xuất | varchar |  |
| 6 | thuonghieu\_id | Thương hiệu | integer | Khóa ngoại |
| 7 | thongsokythuat | Thông số kỹ thuật | varchar |  |

Bảng 3.3: Chi tiết thực thể “hinhanh”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *STT* | *Thuộc tính* | *Diễn giải* | *Kiểu dữ liệu* | *Ràng buộc toàn vẹn* |
| 1 | id | ID hình ảnh | integer | Khóa chính |
| 2 | link | Link hình ảnh | varchar |  |
| 3 | sanpham\_id | Sản phẩm | integer | Khóa ngoại |

Bảng 3.4: Chi tiết thực thể “thuonghieu”

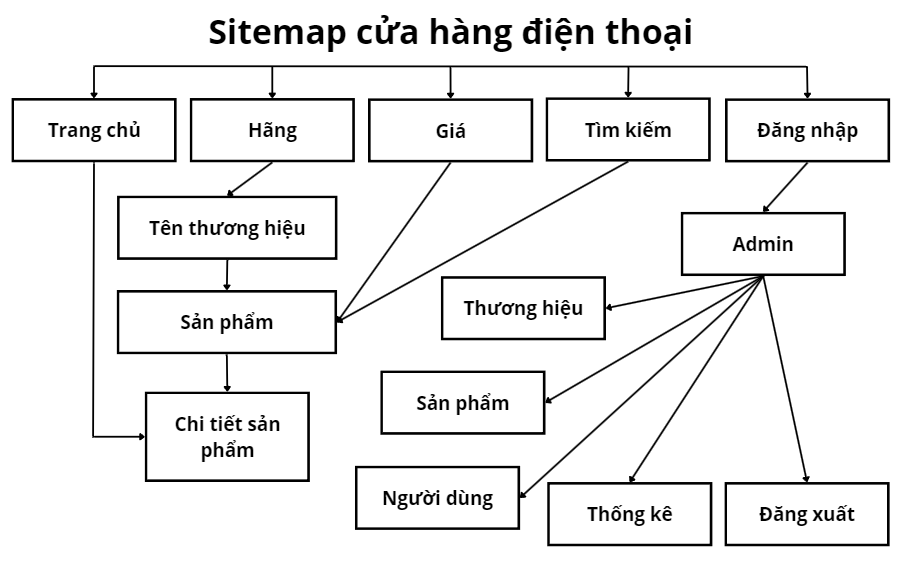
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *STT* | *Thuộc tính* | *Diễn giải* | *Kiểu dữ liệu* | *Ràng buộc toàn vẹn* |
| 1 | id | ID thương hiệu | integer | Khóa chính |
| 2 | tenthuonghieu | Tên thương hiệu | varchar |  |
| 3 | xuatxu | Xuất xứ | varchar |  |
| 4 | mota | Mô tả | longtext |  |
| 5 | anhthuonghieu | Ảnh thương hiệu | varchar |  |

Bảng 3.5: Chi tiết thực thể “nguoidung”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *STT* | *Thuộc tính* | *Diễn giải* | *Kiểu dữ liệu* | *Ràng buộc toàn vẹn* |
| 1 | id | ID người dùng | integer | Khóa chính |
| 2 | tennguoidung | Tên người dùng | varchar |  |
| 3 | email | Email người dùng | varchar |  |
| 4 | sodienthoai | Số điện thoại | varchar |  |
| 5 | diachi | Địa chỉ người dùng | varchar |  |
| 6 | quyen | Phân quyền | varchar |  |
| 7 | matkhau | Mật khẩu | varchar |  |

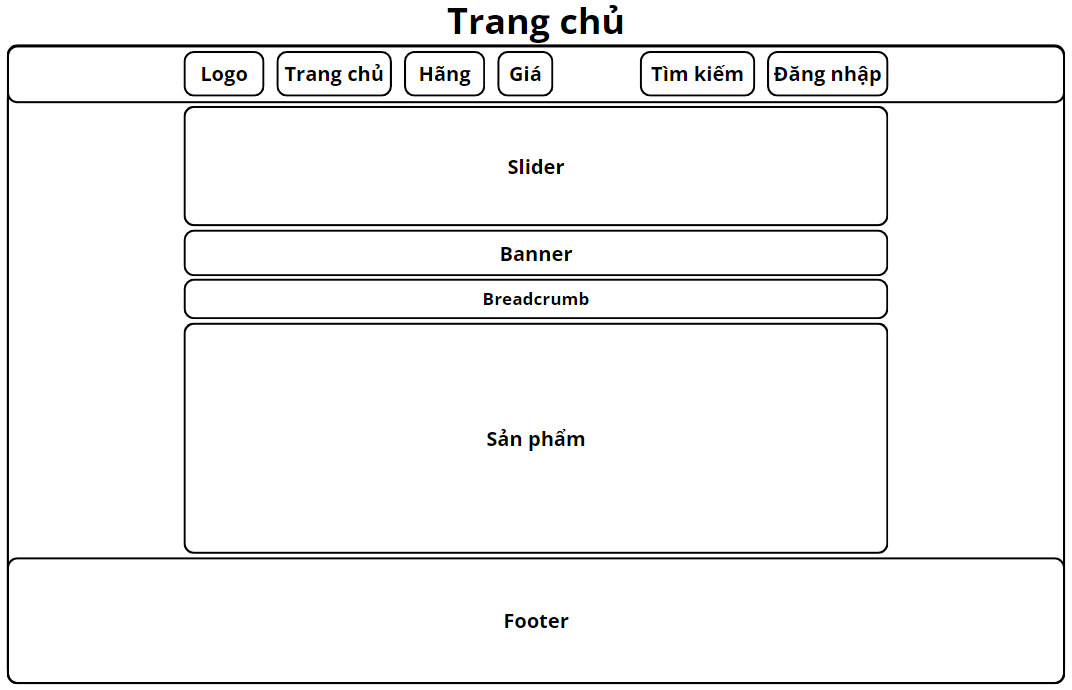
## Thiết kế giao diện

### Sơ đồ website

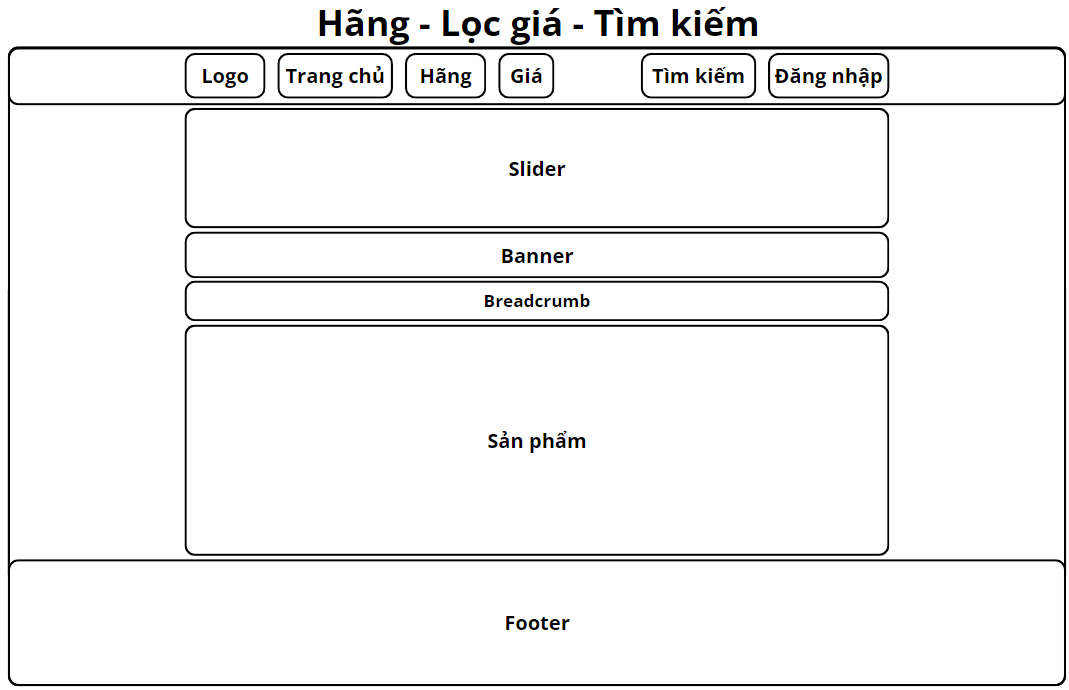


Hình 3.2 Sitemap cửa hàng điện thoại Trung Nghĩa Store

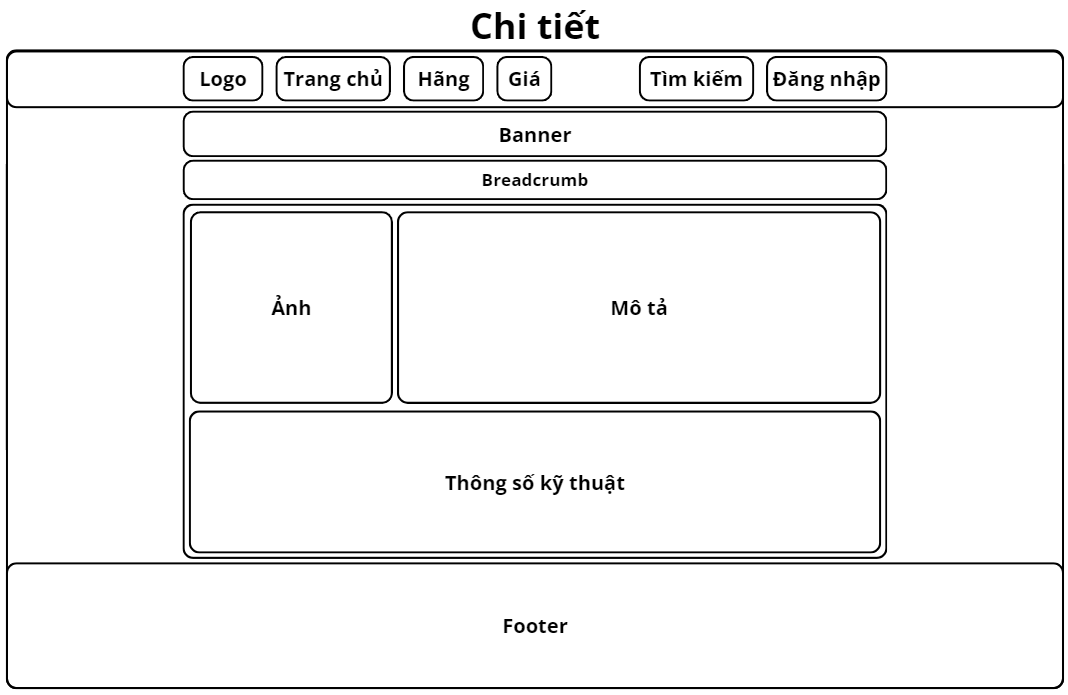
### Giao diện website

Trang chủ hiển thị danh sách các sản phẩm điện thoại mới nhất của cửa hàng. 

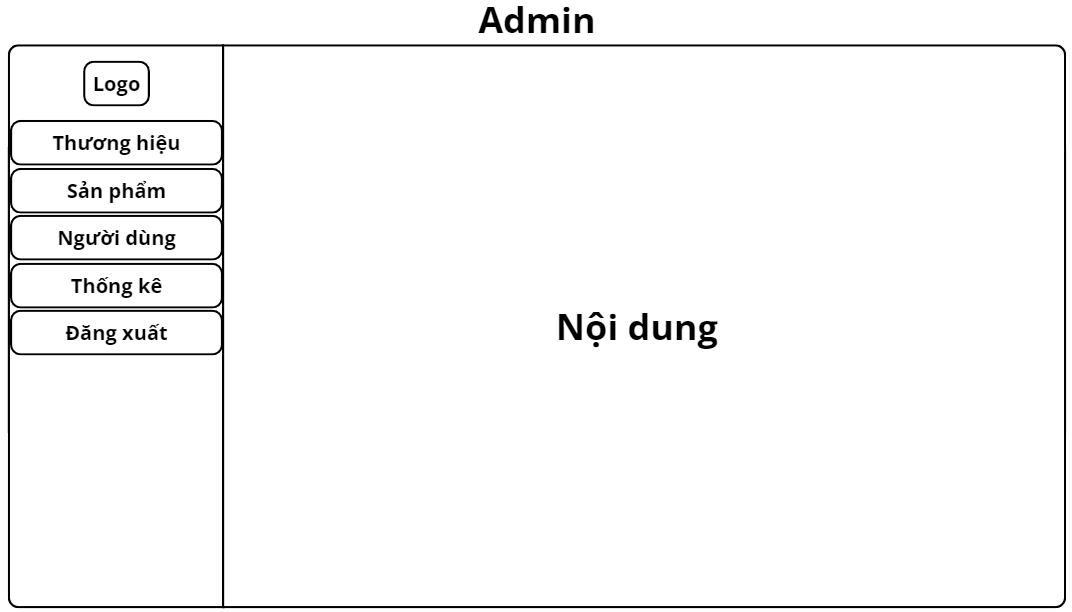
Danh mục hiển thị danh sách sản phẩm theo từng hãng, chức năng lọc theo giá và hiển thị các sản phẩm theo tầm giá đã chọn, chức năng tìm kiếm và hiển thị các sản phẩm theo yêu cầu tìm kiếm.



Sản phẩm chi tiết hiển thị thông tin chi tiết của một sản phẩm.

****

Chức năng đăng nhập cho người dùng đăng nhập vào tài khoảng người quản trị. Người quản trị có thể thêm xóa sửa thương hiệu, sản phẩm, người dùng, thống kê sản phẩm và đăng xuất khỏi trang quản trị.

****

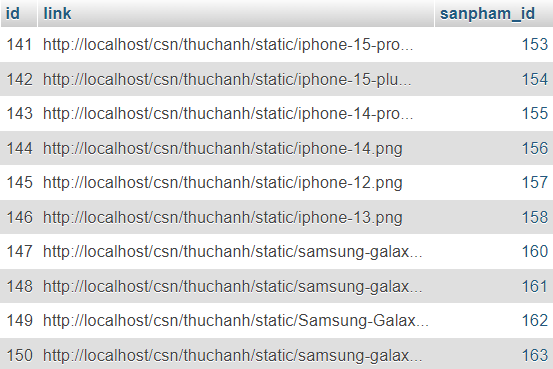
# KẾT QUẢ NGHIÊN CỨU

Trình bày các kết quả đạt được sau quá trình thực hiện đồ án. Có thể đánh giá về hiệu năng, trải nghiệm người dùng, hoặc trình bày các giao diện chức năng của nghiên cứu ở phần này.

## Dữ liệu thử nghiệm

Dữ liệu thử nghiệm được tham khảo từ nguồn: <https://www.thegioididong.com/>

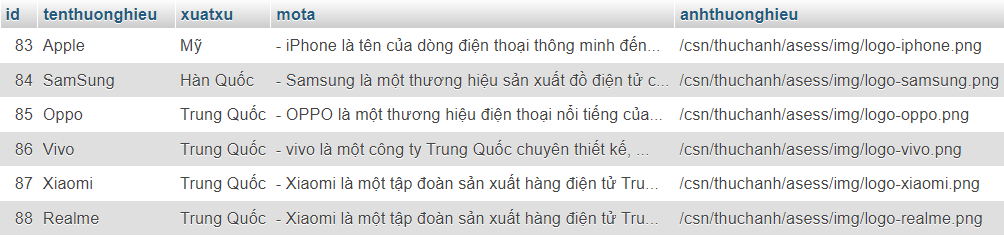
|  |  |  |
| --- | --- | --- |
| 141 | http://localhost/csn/thuchanh/static/iphone-15-pro-max.png | 153 |
| 142 | http://localhost/csn/thuchanh/static/iphone-15-plus.png | 154 |
| 143 | http://localhost/csn/thuchanh/static/iphone-14-pro-max.png | 155 |
| 144 | http://localhost/csn/thuchanh/static/iphone-14.png | 156 |
| 145 | http://localhost/csn/thuchanh/static/iphone-12.png | 157 |
| 146 | http://localhost/csn/thuchanh/static/iphone-13.png | 158 |
| 147 | http://localhost/csn/thuchanh/static/samsung-galaxy-a54.png | 160 |
| 148 | http://localhost/csn/thuchanh/static/samsung-galaxy-m54.png | 161 |
| 149 | http://localhost/csn/thuchanh/static/Samsung-Galaxy-S21-FE.png | 162 |
| 150 | http://localhost/csn/thuchanh/static/samsung-galaxy-s23-fe-5g.png | 163 |
| 151 | http://localhost/csn/thuchanh/static/samsung-galaxy-z-flip5.png | 164 |
| 152 | http://localhost/csn/thuchanh/static/oppo-a79-5g.png | 165 |
| 156 | http://localhost/csn/thuchanh/static/oppo-reno10.png | 169 |
| 166 | http://localhost/csn/thuchanh/static/oppo-reno8-pro.png | 168 |



Hình 4.1 Bảng hình ảnh



Hình 4.2 Bảng sản phẩm



Hình 4.3 Bảng thương hiệu



Hình 4.4 Bảng người dùng

## Kết quả thử nghiệm

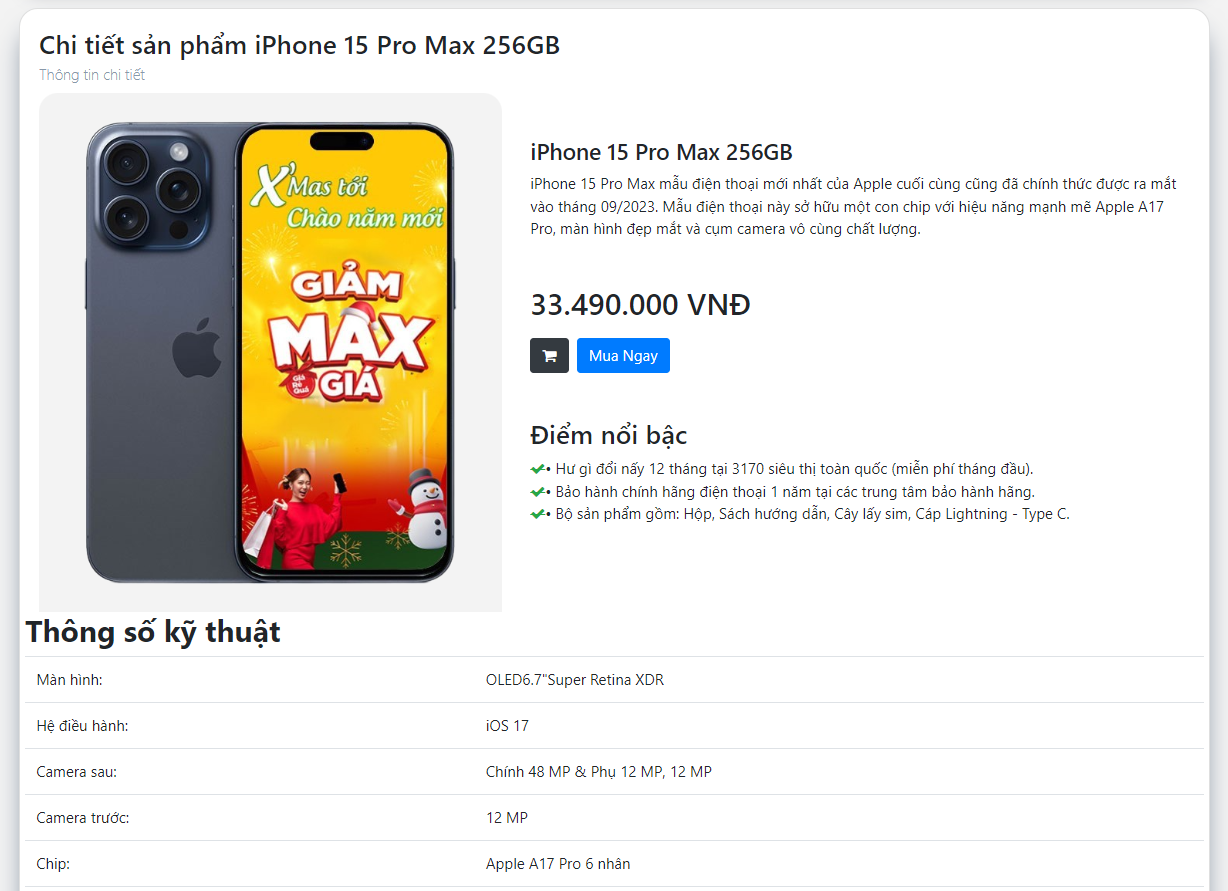
### Chức năng người dùng

Xem sản phẩm: người dùng có thể xem các sản phẩm hiện đang có ở trang chủ.



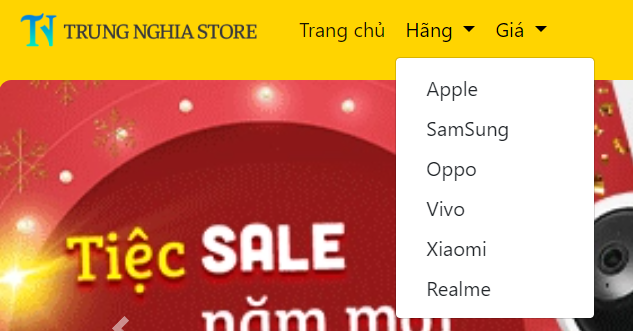
Hình 4.5 Xem trang chủ

Xem chi tiết sản phẩm: người dùng có thể xem chi tiết sản phẩm sau khi chọn.



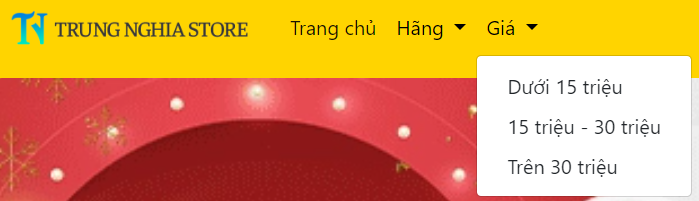
Hình 4.6 Xem chi tiết sản phẩm

Lọc theo hãng: người dùng có thể chọn hãng theo nhu cầu và trang web sẽ hiện thị các sản phẩm thuộc hãng mà người dùng đã chọn.



Hình 4.7 Chức năng lọc theo hãng

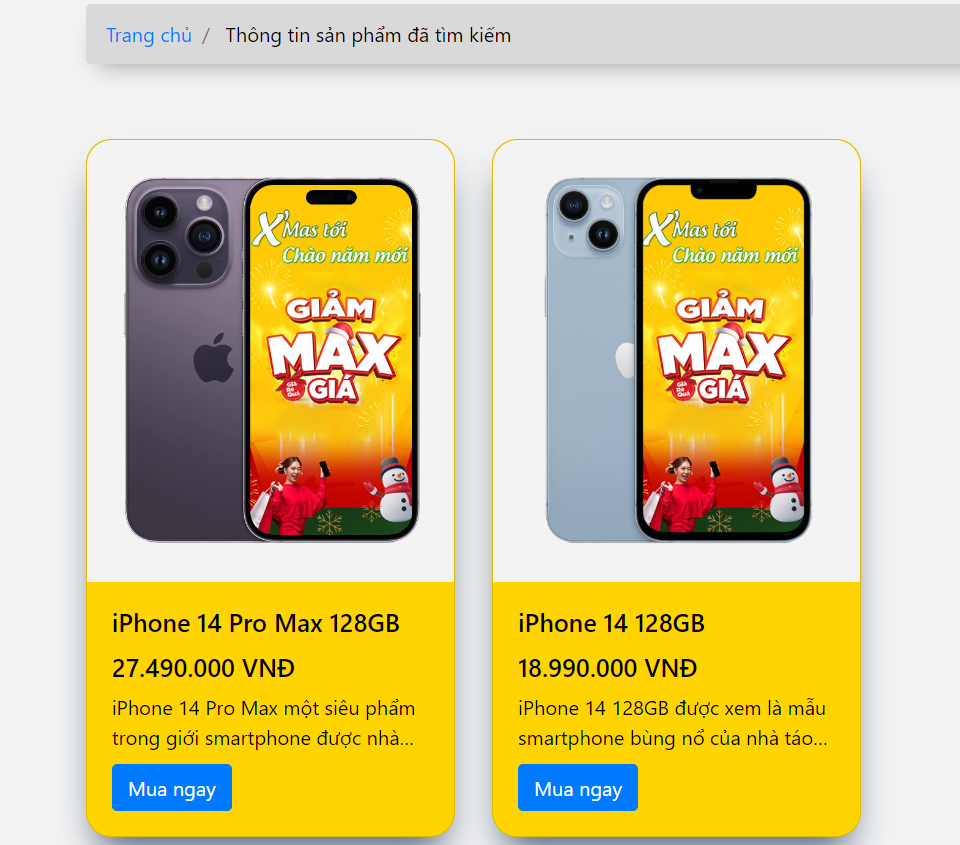
Lọc theo giá: người dùng có thể chọn giá theo phân khúc và trang web sẽ hiện thị các sản phẩm có giá thuộc phân khúc mà người dùng đã chọn.



Hình 4.8 Chức năng lọc theo giá

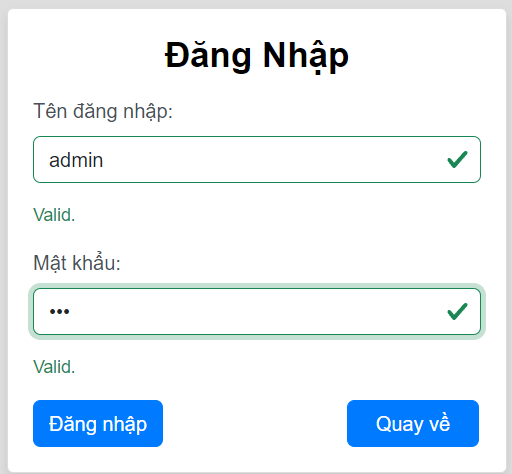
Tìm kiếm: người dùng có thể tìm kiếm tên của sản phẩm và trang web sẽ hiện thị các sản phẩm có tên như người dùng đã tìm kiếm.

Ví dụ như người dùng nhập vào ô tìm kiếm với nội dung: ‘iphone 14’ thì kết quả sẽ hiển thị như sau.



Hình 4.9 Chức năng tìm kiếm

Chức năng đăng nhập: người dùng chỉ mới được đăng nhập tài khoảng admin.

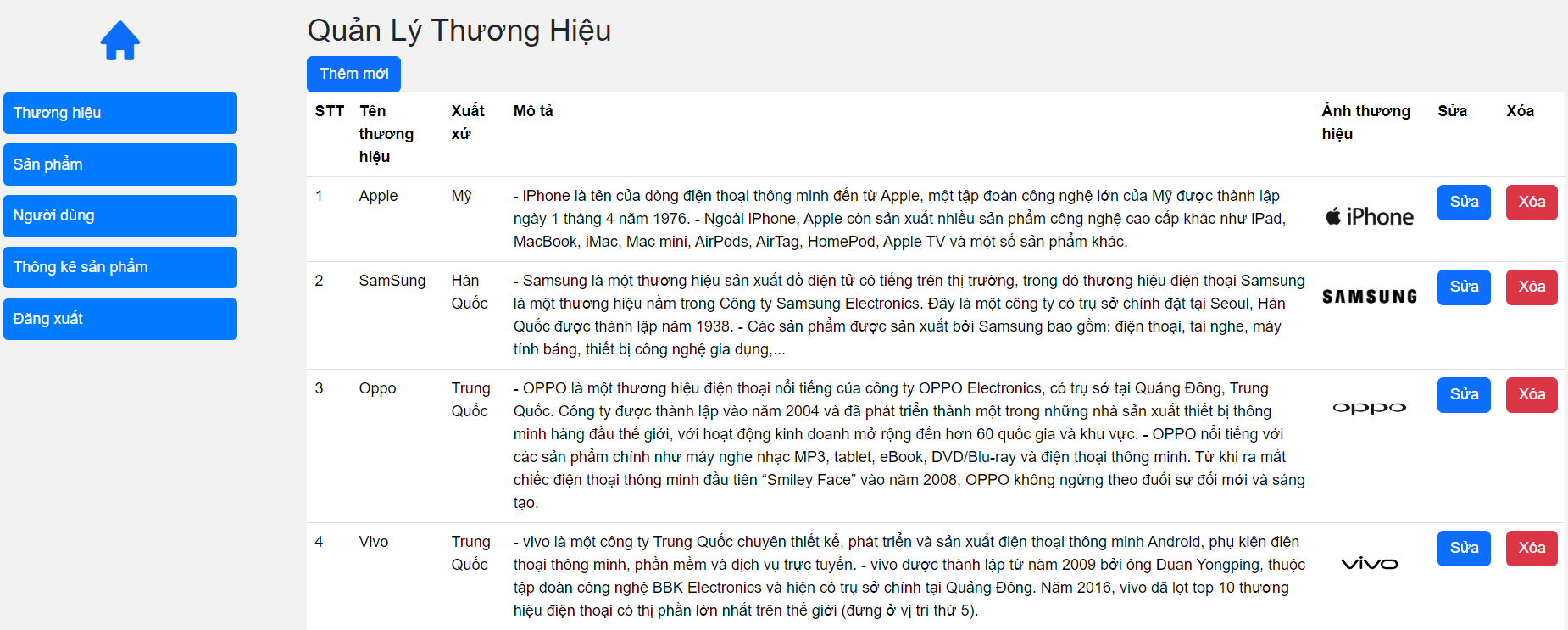


Hình 4.10 Chức năng đăng nhập

### Chức năng quản trị

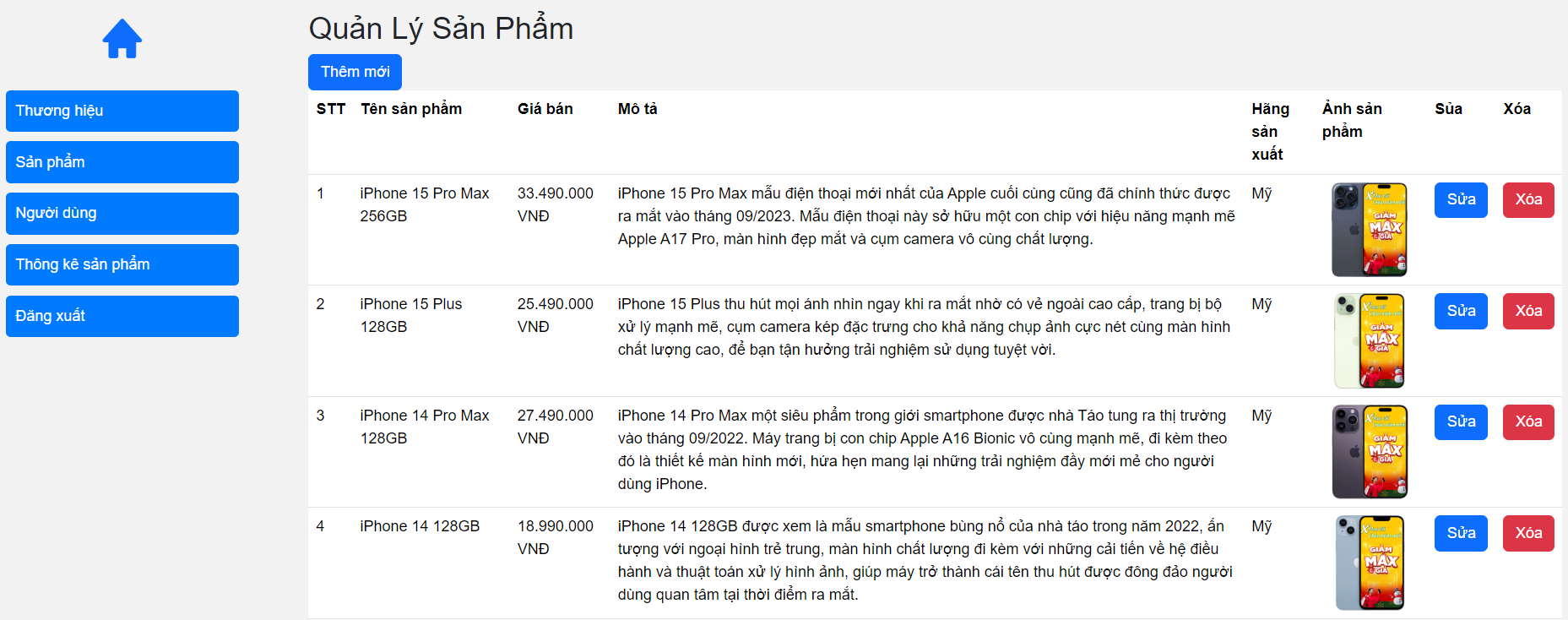
#### Quản lý dữ liệu

Quản lý thương hiệu: hiển thị các thương hiệu đã thêm và có thể thêm xóa sửa.



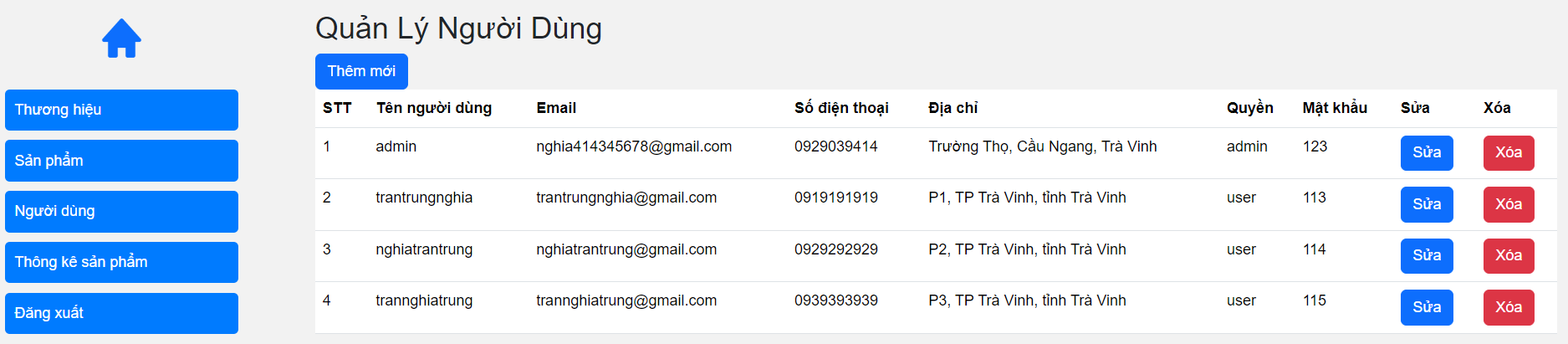
Hình 4.11 Quản lý thương hiệu

Quản lý sản phẩm: hiển thị các sản phẩm đã thêm và có thể thêm xóa sửa.



Hình 4.12 Quản lý sản phẩm

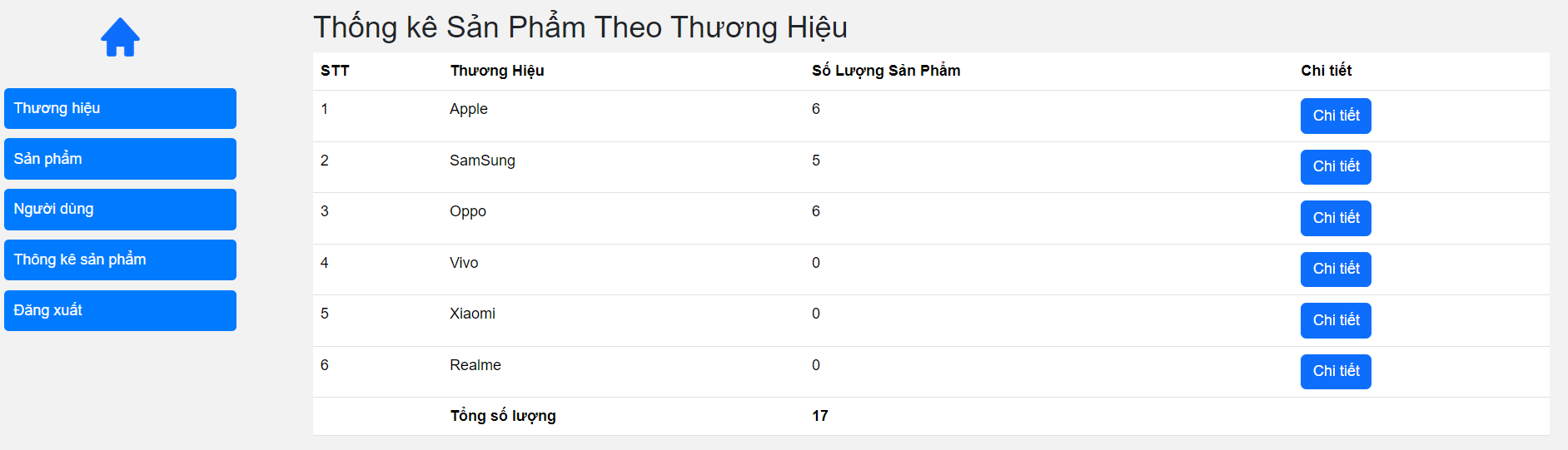
Quản lý người dùng: hiển thị thông tin của người dùng và có thể thêm xóa sửa.



Hình 4.13 Quản lý người dùng

#### Tìm kiếm và thống kê

Thống kê sản phẩm theo hãng: thống kê số lượng của sản phẩm theo từng hãng và có thể thêm xóa sửa theo từng hãng.



Hình 4.14 Thống kê sản phẩm

## Kết luận

Sau một thời gian tìm hiểu và thực hành thì hệ thống của trang web cơ bản đã được hoàn thiện. Hệ thống hoạt động khá ổn định, thử nghiệm các chức năng có trên hệ thống bằng nguồn dữ liệu mẫu đã thu thập cho ra kết quả khá tốt, các chức năng của người dùng và người quản trị hoạt bình thường tuy nhiên hệ thống vẫn chưa đạt hiệu suất cao do chưa có nhiều kinh nghiệm.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận: Trình bày những kết quả đạt được, những đóng góp mới và những đề xuất mới. Phần kết luận cần ngắn gọn, không có lời bàn và bình luận thêm.

Hướng phát triển: Kiến nghị về những hướng nghiên cứu tiếp theo.

## Kết luận

### Kết quả đạt được

Học được kiến thức mới về framework AngularJS, PHP và MySQL.

Thiết kế được một trang web giới thiệu điện thoại bằng AngularJS.

Chức năng lọc, tìm kiếm, đăng nhập cho người dùng và chức năng thêm, xóa, sửa, thống kê cho người quản trị.

~~Đóng góp mới: ??~~

~~Đề xuất mới: Thêm chức năng mua hàng và thêm vào giỏ hàng.~~

## Hướng phát triển

Thêm các chức năng mới cho trang web như: đăng ký, thêm vào giỏ hàng, mua hàng, bình luận, chat.

Tối ưu hóa sự tương thích của trang web đối với nhiều loại thiết bị.

Tối ưu hóa trang web cho trải nghiệm người dùng tốt hơn.

# DANH MỤC TÀI LIỆU THAM KHẢO

[1] W3Schools "AngularJS Tutorial" W3Schools Online Web Tutorials. [6/11/2023]. Url: <https://www.w3schools.com/angular/default.asp>

[2] W3Schools "W3Schools PHP Tutorial" W3Schools Online Web Tutorials. [10/11/2023]. URL: <https://www.w3schools.com/php/default.asp>

[3] W3Schools "W3Schools MySQL Tutorial" W3Schools Online Web Tutorials. [10/11/2023]. URL: <https://www.w3schools.com/mysql/default.asp>

# PHỤ LỤC

**Các phần mềm và công cụ đã cài đặt**

Visual Studio Code (VSCode): Trình soạn thảo mã nguồn mở, hỗ trợ nhiều ngôn ngữ lập trình và tích hợp với nhiều tiện ích mở rộng (extension). Url: <https://code.visualstudio.com/>

Google Chrome: Trình duyệt web phổ biến được phát triển bởi Google. Url: <https://www.google.com/chrome/>

Laragon: Môi trường phát triển tổng quát nhanh chóng và mạnh mẽ dành cho PHP, Node.js, Python, Java. Url: <https://laragon.org/download/index.html>

# References

|  |  |
| --- | --- |
| [1] | N. v. Sơn, Nhập môn Công nghệ thông tin, TP. HỒ chí Minh: Lao động, 2015. |
| [2] | N. Thanh, Kỹ thuật lập trình, Trà Vinh: Cần Thơ, 2014. |