**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**Môn học: Phát triển ứng dụng web**

**ĐỀ TÀI**

**QUẢN LÝ BÁN SÁCH BOOKSTORE**

**GV hướng dẫn : Bùi Chí Thành  
SV thực hiện : Nguyễn Hoàng Đức Khang  
MSSV: 63134290**

**Mục lục**

[CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI 3](#_Toc151931517)

[1.1. Lý do chọn đề tài: 3](#_Toc151931518)

[1.2. Mục tiêu của đề tài 3](#_Toc151931519)

[1.3. Giới hạn và phạm vi của đề tài 3](#_Toc151931520)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 4](#_Toc151931521)

[2.1. Tổng quan về ngôn ngữ Asp.NET MVC 4](#_Toc151931522)

[2.2. Tổng quan về SQL SERVER 8](#_Toc151931523)

[CHƯƠNG 3: NỘI DUNG THỰC HIỆN 9](#_Toc151931524)

[3.1 . Đối tượng sử dụng website: 9](#_Toc151931525)

[3.2. Các chức năng chính của website: 9](#_Toc151931526)

[3.3 Phân tích thiết kế hệ thống 10](#_Toc151931527)

[3.4 Thiết kế cơ sở dữ liệu 11](#_Toc151931528)

[3.4.1 Mô hình cơ sở dữ liệu 11](#_Toc151931529)

[3.4.2 Danh sách các bảng đối tượng 11](#_Toc151931530)

[3.5 Thiết kế giao diện 15](#_Toc151931531)

[CHƯƠNG 4: KẾT LUẬN 20](#_Toc151931532)

[4.1 Kết quả đạt được của đề tài 20](#_Toc151931533)

[4.2 Hạn chế của đề tài 20](#_Toc151931534)

[4.3 Hướng phát triển 21](#_Toc151931535)

# TỔNG QUAN VỀ ĐỀ TÀI

## Lý do chọn đề tài:

Công nghệ thông tin ngày càng phát triển, thương mại điện tử trở thành một thứ không thể thiếu đối với tất cả mọi người. Chỉ cần vào các trang dịch vụ thương mại điện tử, 1 cú click chuột và các nhà dịch vụ sẽ mang đến tận nhà cho bạn. Để góp phần đẩy mạnh sự phổ biến của thương mại điện tử ở Việt Nam, em đã quyết định làm 1 trang website quản lý và bán sách.

## Mục tiêu của đề tài

* Hỗ trợ khách hàng mua sắm trực tuyến, giải quyết nhược điểm của cách mua hàng truyền thống, mất thời gian.
* Hỗ trợ các admin trong việc quản lý khách hàng và kho hàng một cách dễ dàng hơn.

## Giới hạn và phạm vi của đề tài

* Cho phép người dùng đăng kí và đăng nhập tài khoản để mua hàng
* Tạo 1 giao diện giúp người dùng có thể xem sách và mua sách
* Cho phép admin quản lý: thêm, xóa, sửa và xem thông tin sách

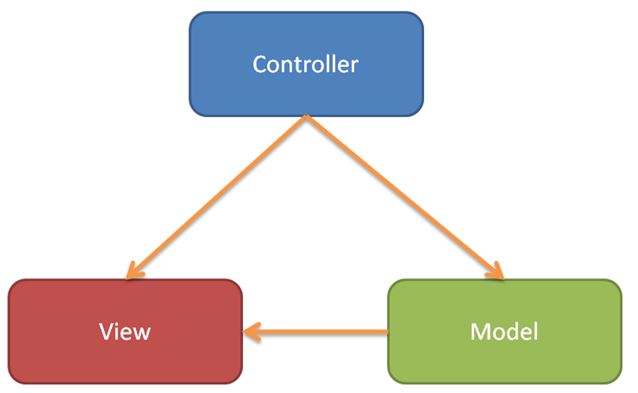
# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Tổng quan về ngôn ngữ Asp.NET MVC

ASP.Net MVC là một Framework sử dụng .Net Framework cho việc phát triển ứng dụng web động, trước khi ASP.Net MVC ra đời, lập trình viên sử dụng công nghệ ASP.Net Web Form trên nền tảng .Net Framework để phát triển ứng dụng Web động. ASP.Net MVC phát triển trên mẫu thiết kế chuẩn MVC, cho phép người sử dụng phát triển các ứng dụng phần mềm. **MVC** là tên một mẫu phát triển ứng dụng, phương pháp này chia nhỏ một ứng dụng thành ba thành phần để cài đặt, mỗi thành phần đóng một vai trò khác nhau và ảnh hưởng lẫn nhau, đó là models, views, và controllers.

MVC là một mẫu thiết kế (design pattern) chuẩn mà nhiều lập trình viên đã quen thuộc. Một số loại ứng dụng web sẽ thích hợp với kiến trúc MVC. Một số khác vẫn thích hợp với ASP.NET Web Forms và cơ chế postbacks. Đôi khi có những ứng dụng kết hợp cả hai kiến trúc trên.

Nền tảng MVC bao gồm các thành phần dưới đây :



Hình 1: Mẫu Model - View - Controller

**Models:** Các đối tượng Models là một phần của ứng dụng, các đối tượng này thiết lập logic của phần dữ liệu của ứng dụng. Thông thường, các đối tượng model lấy và lưu trạng thái của model trong CSDL. Ví dụ như, một đối tượng Product (sản phẩm) sẽ lấy dữ liệu từ CSDL, thao tác trên dữ liệu và sẽ cập nhật dữ liệu trở lại vào bảng Products ở SQL Server.

Trong các ứng dụng nhỏ, model thường là chỉ là một khái niệm nhằm phân biệt hơn là được cài đặt thực thụ, ví dụ, nếu ứng dụng chỉ đọc dữ liệu từ CSDL và gởi chúng đến view, ứng dụng không cần phải có tầng model và các lớp liên quan. Trong trường hợp này, dữ liệu được lấy như là một đối tượng model (hơn là tầng model).

**Views:**  là các thành phần dùng để hiển thị giao diện người dùng (UI). Thông thường, view được tạo dựa vào thông tin dữ liệu model. Ví dụ như, view dùng để cập nhật bảng Products sẽ hiển thị các hộp văn bản, drop-down list, và các check box dựa trên trạng thái hiện tại của một đối tượng Product.

**Controllers**: Controller là các thành phần dùng để quản lý tương tác người dùng, làm việc với model và chọn view để hiển thị giao diện người dùng. Trong một ứng dụng MVC, view chỉ được dùng để hiển thị thông tin, controller chịu trách nhiệm quản lý và đáp trả nội dung người dùng nhập và tương tác với người dùng. Ví dụ, controller sẽ quản lý các dữ liệu người dùng gởi lên (query-string values) và gởi các giá trị đó đến model, model sẽ lấy dữ liệu từ CSDL nhờ vào các giá trị này.

Mẫu MVC giúp bạn tạo được các ứng dụng mà chúng phân tách rạch ròi các khía cạnh của ứng dụng (logic về nhập liệu, logic xử lý tác vụ và logic về giao diện). Mẫu MVC chỉ ra mỗi loại logic kể trên nên được thiếp lập ở đâu trên ứng dụng. Logic giao diện (UI logic) thuộc về views. Logic nhập liệu (input logic) thuộc về controller. Và logic tác vụ (Business logic – là logic xử lý thông tin, mục đích chính của ứng dụng) thuộc về model. Sự phân chia này giúp bạn giảm bớt được sự phức tạp của ứng dụng và chỉ tập trung vào mỗi khía cạnh cần được cài đặt ở mỗi thời điểm. Ví dụ như bạn chỉ cần tập trung vào giao diện (views) mà không phải quan tâm đến logic xử lý thông tin của ứng dụng.

Để quản lý sự phức tạp của ứng dụng, mẫu MVC giúp cho chúng ta có thể kiểm thử ứng dụng dễ dàng hơn hẳn so với khi áp dụng mẫu Web Forms. Ví dụ, trong một ứng dụng ASP.NET Web Forms, một lớp thường được sử dụng để hiển thị thông tin xuất ra cho người dùng và đồng thời xử lý thông tin người dùng nhập. Việc xây dựng các bộ test tự động cho ứng dụng Web Forms là rất phức tạp, bởi để kiểm thử mỗi trang web, bạn phải khởi tạo đối tượng trang, khởi tạo tất cả các control được sử dụng trong trang và các lớp phụ thuộc trong ứng dụng. Và bởi vì có quá nhiều lớp cần được khởi tạo để chạy được trang, thật khó để có thể viết các test chỉ tập trung vào một khía cạnh nào đó của ứng dụng. Và vì thế, kiểm thử đối với các ứng dụng dứa trên nền tảng Web Forms sẽ khó khăn hơn nhiều so với khi áp dụng trên ứng dụng MVC. Hơn thế nữa, việc kiểm thử trên nền tảng Web Forms yêu cầu phải sử dụng đến web server. Nền tảng MVC phân tách các thành phần và sử dụng các interface (khái niệm giao diện trong lập trình hướng đối tượng), và nhờ đó có thể kiểm thử các thành phần riêng biệt trong tình trạng phân lập với các yếu tố còn lại của ứng dụng

Sự phân tách rạch ròi ba thành phần của ứng dụng MVC còn giúp cho việc lập trình diễn ra song song. Ví dụ như một lập trình viên làm việc với view, lập trình viên thứ hai lo cài đặt logic của controller và lập trình viên thứ ba có thể tập trung vào logic tác vụ của model tại cùng một thời điểm.

Lấy ví dụ một GUI Component (thành phần đồ họa người dùng ) đơn giản là checkbox. Checkbox có thành phần Model để quản lý trạng thái của nó là check hay uncheck, thành phần View để thể hiện nó với trạng thái tương ưng lên màn hình, và thành phần Controller để xử lý nhưng sự kiện khi có sự tương tác của người sử dụng hoặc các đối tượng khác lên checkbox.

Khi người sử dụng nhần chuột vào Check box , thành phần Controller của Checkbox sẽ xử lý sự kiện này, yêu cầu thành phần Model thay đổi dữ liệu trạng thái. Sau khi thay đổi trạng thái, thành phần Model phát thông điệp đến thành phần View và Controller. Thành phần View của Checkbox nhận được thông điệp sẽ cập nhật lại thể hiện của Checkbox, phản ánh chính sác trạng thái Checkbox do Model lưu giữ. Thành phần Controller nhận được thông điệp do Model gởi tới sẽ có nhưng tương tác phản hòi với người sử dụng nếu cần thiết.

Chẳng hạn: Để dễ thao tác, smarty sẽ dịch ngược các yêu cầu của bạn sang ngôn ngữ của nó. Sau đó chúng sẽ chuyển ngôn ngữ đó sang PHP và thao tác xử lý trên nó.  
Việc này sẽ làm ứng dụng chậm chạp, do cứ phải dịch qua, dịch lại một kịch bản. Trong khi, với sự kết hợp của PHP thuần, ứng dụng của bạn sẽ nhanh và ổn định hơn nhiều. Và view cũng là một phần trong việc nâng cấp những hạn chế ấy. Chúng giúp giảm thiểu tối đa quá trình biên dịch nhiều lần. Và làm cho ứng dụng trở nên mạnh mẽ và chuyên nghiệp hơn nhiều so với cách lập trình thuần.

**Tính năng của ASP.Net MVC**

* Tách bạch các tác vụ của ứng dụng (logic nhập liệu, business logic, và logic giao diện), dễ dàng kiểm thử và mặc định áp dụng hướng phát triển TDD. Tất cả các tính năng chính của mô hình MVC được cài đặt dựa trên interface và được kiểm thử bằng cách sử dụng các đối tượng mocks, mock object là các đối tượng mô phỏng các tính năng của những đối tượng thực sự trong ứng dụng. Bạn có thể kiểm thử unit-test cho ứng dụng mà không cần chạy controller trong tiến trình ASP.NET, và điều đó giúp unit test được áp dụng nhanh chóng và tiện dụng. Bạn có thể sử dụng bất kỳ nền tảng unit-testing nào tương thích với nền tảng .NET.
* MVC là một nền tảng khả mở rộng (extensible) & khả nhúng (pluggable). Các thành phần của ASP.NET MVC được thiết kể để chúng có thể được thay thế một cách dễ dàng hoặc dễ dàng tùy chỉnh. Bạn có thể nhúng thêm view engine, cơ chế định tuyến cho URL, cách kết xuất tham số của action-method và các thành phần khác. ASP.NET MVC cũng hỗ trợ việc sử dụng Dependency Injection (DI) và Inversion of Control (IoC). DI cho phép bạn gắn các đối tượng vào một lớp cho lớp đó sử dụng thay vì buộc lớp đó phải tự mình khởi tạo các đối tượng. IoC quy định rằng, nếu một đối tượng yêu cầu một đối tượng khác, đối tượng đầu sẽ lấy đối tượng thứ hai từ một nguồn bên ngoài, ví dụ như từ tập tin cấu hình. Và nhờ vậy, việc sử dụng DI và IoC sẽ giúp kiểm thử dễ dàng hơn.
* ASP.NET MVC có thành phần ánh xạ URL mạnh mẽ cho phép bạn xây dựng những ứng dụng có các địa chỉ URL xúc tích và dễ tìm kiếm. Các địa chỉ URL không cần phải có phần mở rộng của tên tập tin và được thiết kế để hỗ trợ các mẫu định dạng tên phù hợp với việc tối ưu hóa tìm kiếm (URL) và phù hợp với lập địa chỉ theo kiểu REST..
* Hỗ trợ các tính năng có sẵn của ASP.NET như cơ chế xác thực người dùng, quản lý thành viên, quyền, output caching và data caching, session và profile, quản lý tình trạng ứng dụng, hệ thống cấu hình…

2.2. Tổng quan về SQL SERVER

SQL Server là một hệ thống quản lý cơ sở dữ liệu RDBMS sử dụng Transact-SQL để trao đổi dữ liệu giữa Client computer và SQL Server computer. Một RDBMS bao gồm Databases, Database engine và các ứng dụng dùng để quản lý dữ liệu và các bộ phận khác nhau trong RDBMS.

SQL Server được tối ưu để có thể chạy trên môi trường cơ sở dữ liệu rất lớn (Very Large Database Environment) lên đến Tera -Byte và có thể phục vụ cùng lúc cho hàng ngàn User. SQL Server 2000 có thể kết hợp "ăn ý" với các Server khác như Microsoft Internet Information Server (IIS), ECommerce Server, Proxy Server...

Ưu điểm:

* Không cần code: Rất dễ dàng để quản lý các hệ thống cơ sở dữ liệu bằng việc sử dụng SQL chuẩn mà không cần phải viết bất cứ dòng code nào.
* Tiêu chuẩn được quy định rõ ràng: SQL sử dụng hai tiêu chuẩn ISO và ANSI, trong khi với các non-SQL database không có tiêu chuẩn nào được tuân thủ.
* Tính di động: SQL có thể được sử dụng trong chương trình trong PCs, servers, laptops, và thậm chí cả mobile phones.
* Ngôn ngữ tương tác: Language này có thể được sử dụng để giao tiếp với cơ sở dữ liệu và nhận câu trả lời cho các câu hỏi phức tạp trong vài giây.
* Multiple data views: Với sự trợ giúp của ngôn ngữ SQL, người dùng có thể tạo các hiển thị khác nhau về cấu trúc cơ sở dữ liệu và cơ sở dữ liệu cho những người dùng khác nhau.

Nhược điểm:

* Giao diện khó dùng: SQL có giao diện phức tạp khiến một số người dùng khó truy cập.
* Không được toàn quyền kiểm soát: Các lập trình viên sử dụng SQL không có toàn quyền kiểm soát cơ sở dữ liệu do các quy tắc nghiệp vụ ẩn.
* Thực thi: Hầu hết các chương trình cơ sở dữ liệu SQL đều có phần mở rộng độc quyền riêng của nhà cung cấp bên cạnh các tiêu chuẩn SQL.
* Giá cả: Chi phí vận hành của một số phiên bản SQL khiến một số lập trình viên gặp khó khăn khi tiếp cận.

# CHƯƠNG 3: NỘI DUNG THỰC HIỆN

## . Đối tượng sử dụng website:

* Đối với Admin: Admin có quyền thêm/sửa/xóa và xem danh sách chi tiết của sách.
* Đối với khách hàng: Khách hàng có thể xem sách, đăng nhập/đăng ký thông tin trên website.

## 3.2. Các chức năng chính của website:

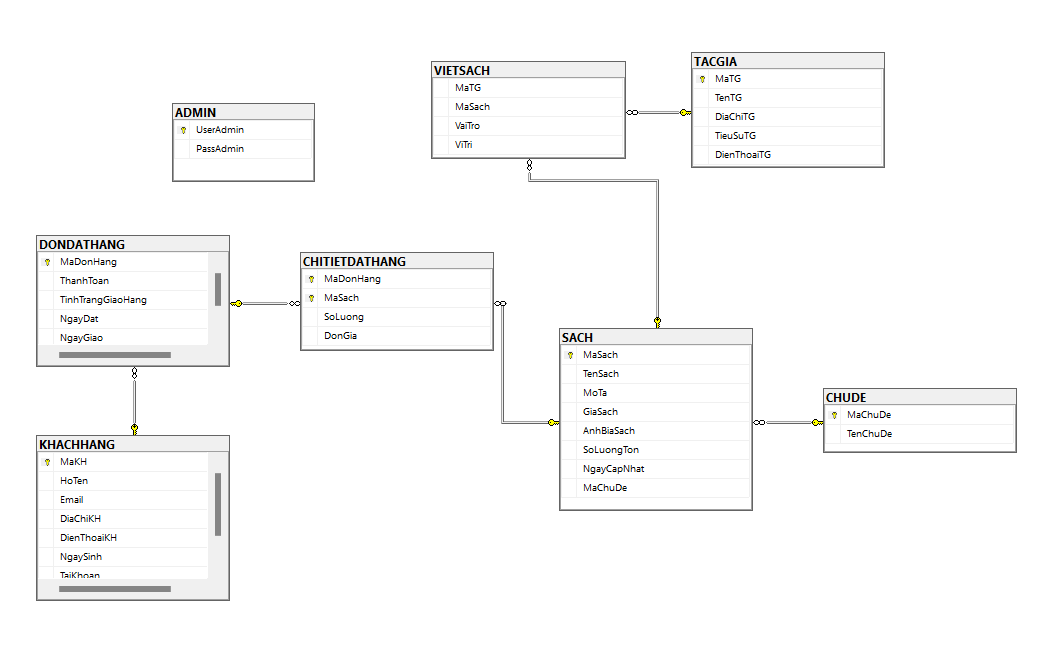
Chức năng đối với Admin:

* Đăng nhập vào trang Admin
* Thêm sách
* Xóa sách
* Sửa sách
* Xem chi tiết thông tin sách
* Kiểm tra danh sách sách

Chức năng đối với Khách hàng:

* Đăng ký/Tạo tài khoản
* Đăng nhập
* Xem thông tin sách
* Xem danh sách sách
* Tìm kiếm sách thông qua các chủ đề của sách

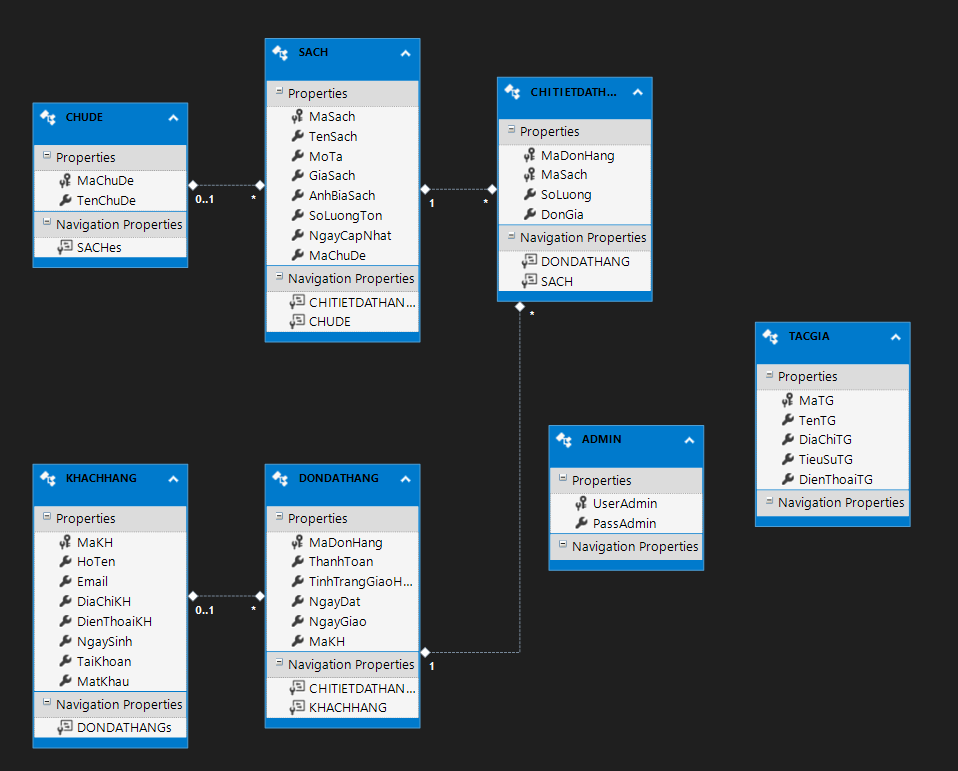
3.3 Phân tích thiết kế hệ thống



Hình 2: Sơ đồ Class Diagram

3.4 Thiết kế cơ sở dữ liệu

3.4.1 Mô hình cơ sở dữ liệu



###### Hình 3: Biểu đồ dữ liệu quan hệ

3.4.2 Danh sách các bảng đối tượng

1. **Bảng KHÁCH HÀNG**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaKH | Varchar(50) | Khóa chính | Mã khách hàng |
| HoTen | Nvarchar(50) |  | Họ tên khách hàng |
| Email | Varchar(70) |  | Email khách hàng |
| DiaChiKH | Nvarchar(100) |  | Địa chỉ khách hàng |
| DienThoaiKH | Varchar(30) |  | Số điện thoại khách hàng |
| NgaySinh | datetime |  | Ngày sinh của khách hàng |
| TaiKhoan | Varchar(50) |  | Tài khoản đăng nhập của khách hàng |
| MatKhau | Varchar(50) |  | Mật khẩu đăng nhập của khách hàng |

1. **Bảng chủ đề**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaChuDe | INT | Khóa chính | Mã chủ đề |
| TenChuDe | Nvarchar(50) |  | Tên chủ đề |

1. **Bảng Sách**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaSach | INT | Khóa chính | Mã sách |
| TenSach | Nvarchar(100) |  | Tên sách |
| MoTa | Nvarchar(500) |  | Mô tả của sách |
| GiaSach | Decimal(18,0) |  | Giá sách |
| AnhBiaSach | Varchar(50) |  | Ảnh bìa sách |
| SoLuongTon | INT |  | Số lượng tồn |
| NgayCapNhat | DateTime |  | Ngày cập nhật |
| MaChuDe | INT | Khóa ngoại | Mã chủ đề |

**d. Bảng tác giả**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaTG | INT | Khóa chính | Mã tác giả |
| TenTG | Nvarchar(50) |  | Tên tác giả |
| DiaChiTG | Nvarchar(100) |  | Địa chỉ tác giả |
| TieuSuTG | Nvarchar(500) |  | Tiểu sử tác giả |
| DienThoaiTG | Varchar(30) |  | Số điện thoại tác giả |

**e. Bảng Viết sách**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaTG | INT | Khóa chính, Khóa ngoại | Mã tác giả |
| MaSach | INT | Khóa chính, Khóa ngoại | Mã sách |
| VaiTro | Nvarchar(50) |  | Vai trò |
| ViTri | Nvarchar(50) |  | Vị trí |

**f. Bảng Đơn đặt hàng**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaDonHang | int | Khóa chính | Mã đơn hàng |
| MaKH | int | Khóa ngoại | Mã khách hàng |
| TinhTrangGiaoHang | bit |  | Tình trạng giao hàng |
| NgayDat | DateTime |  | Ngày đặt |
| NgayGiao | DateTime |  | Ngày giao |
| ThanhToan | bit |  | Thanh toán |

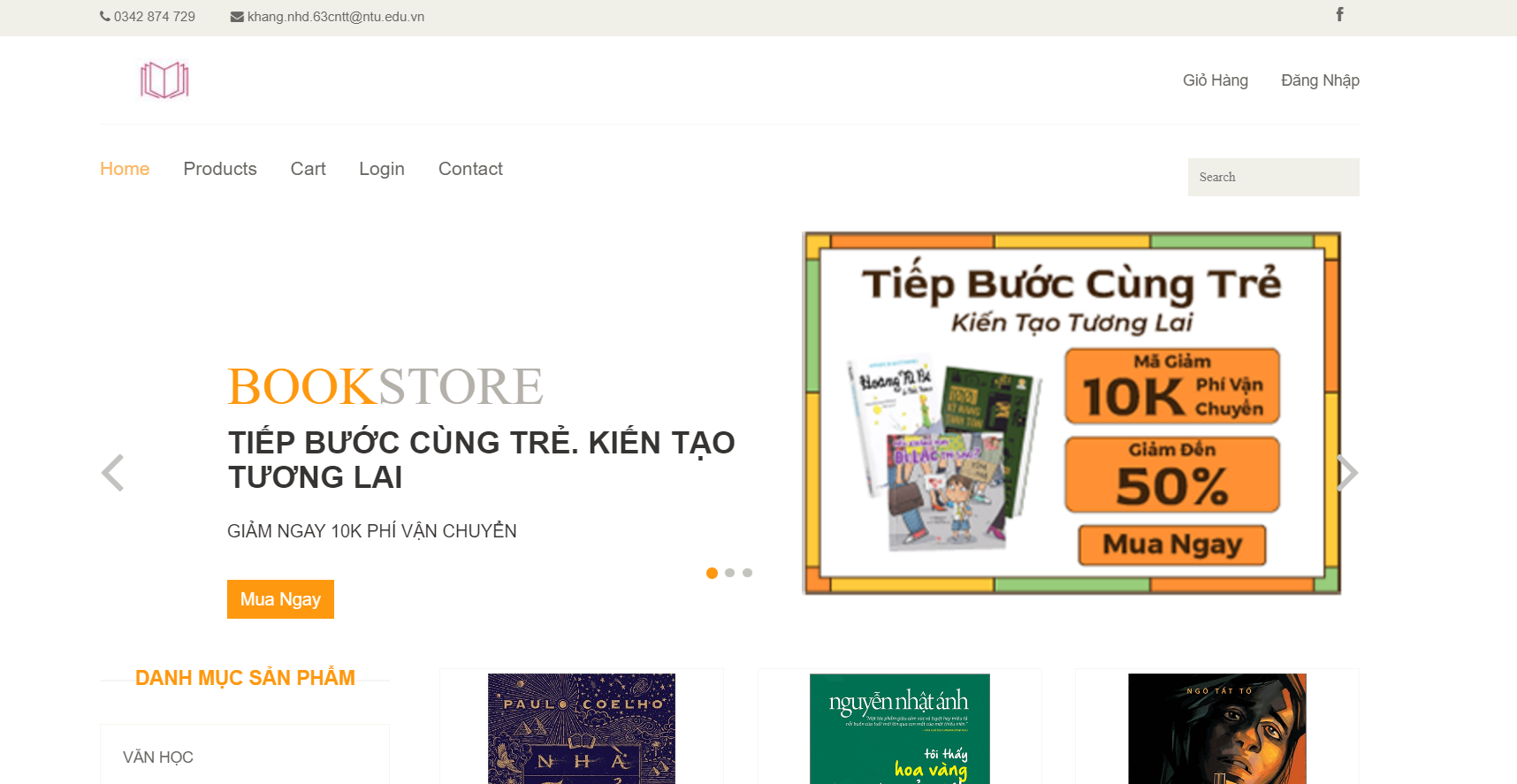
***g*. Bảng Chi tiết đặt hàng**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| MaDonHang | INT | Khóa chính, khóa ngoại | Mã đơn hàng |
| MaSach | INT | Khóa chính, khóa ngoại | Mã sách |
| SoLuong | Int |  | Số lượng |
| DonGia | Decimal(18,0) |  | Giá |

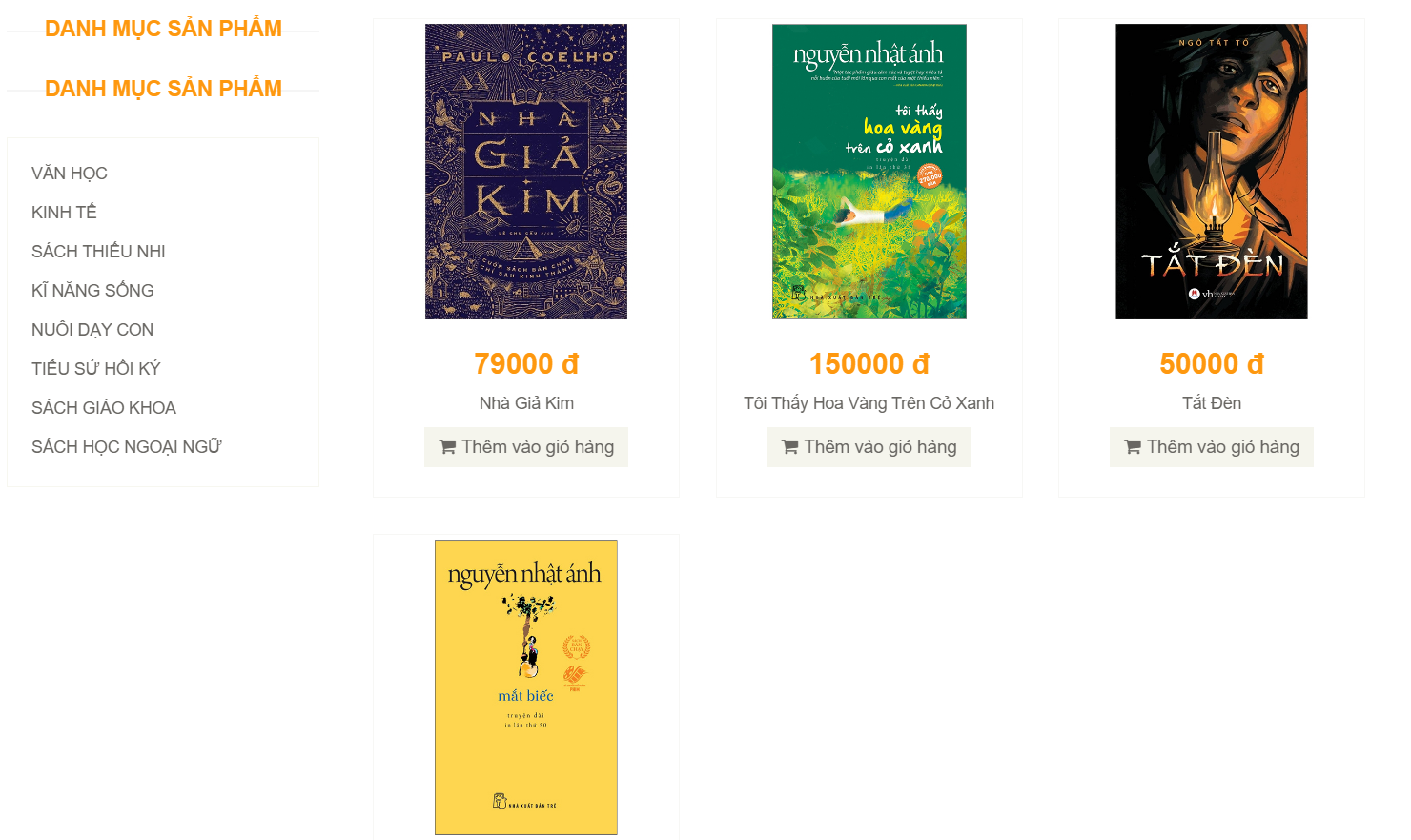
**h. Bảng Admin**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Khóa | Mô tả |
| UserAdmin | Varchar(50) | Khóa chính | Tên đăng nhập Admin |
| PassAdmin | Varchar(50) |  | Mật khẩu Admin |

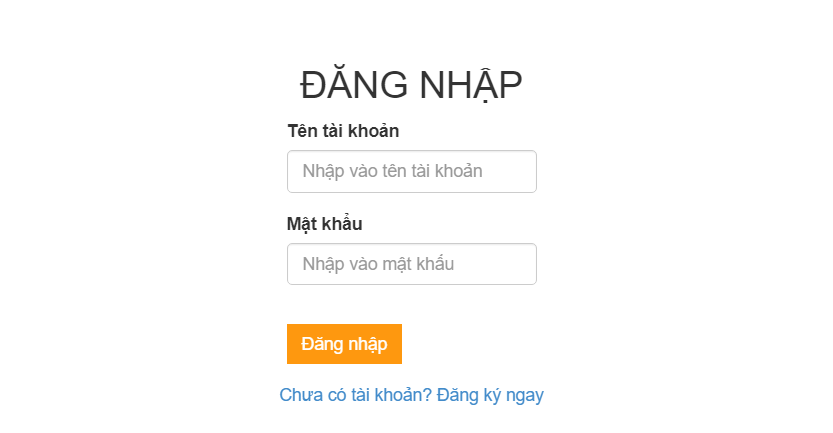
3.5 Thiết kế giao diện



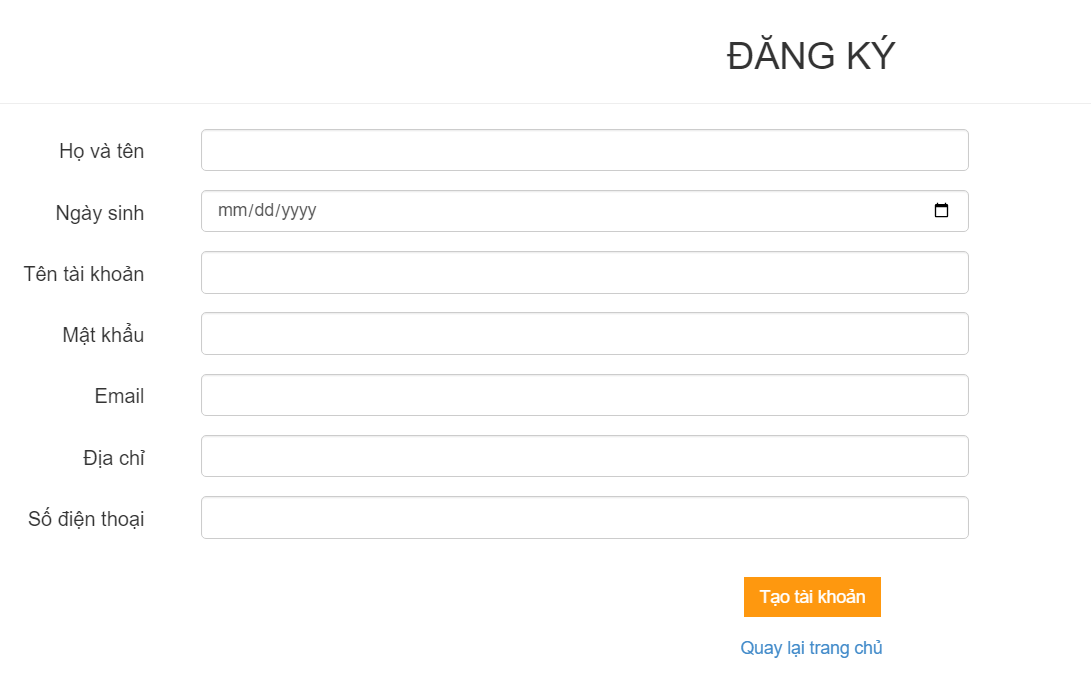
Hình 4: Giao diện trang đăng nhập



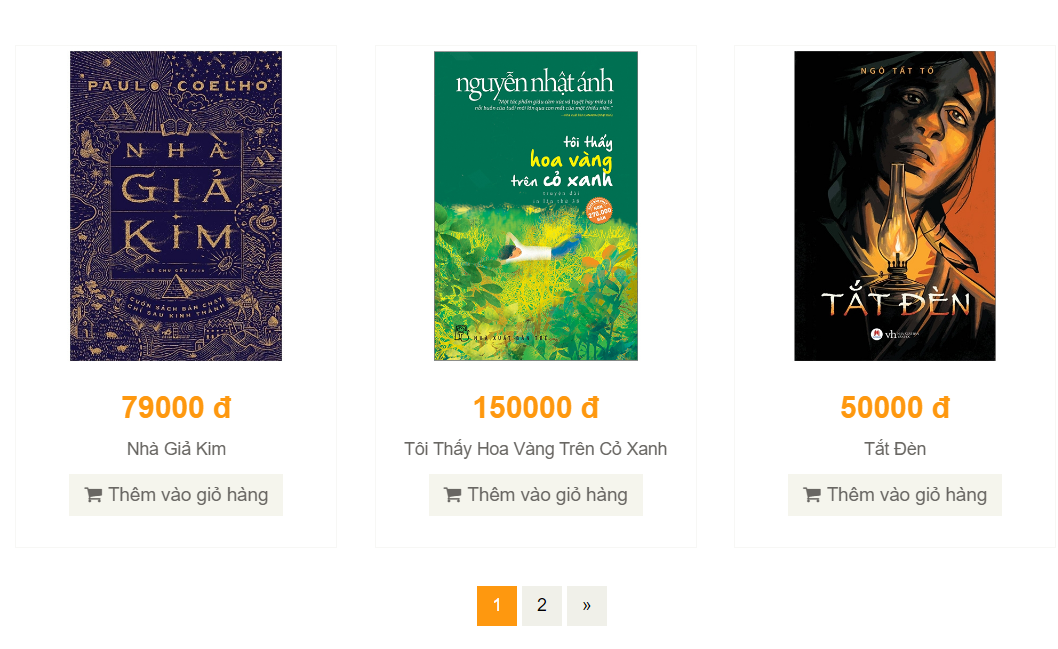
Hình 5: Tìm kiếm sách theo danh mục “Văn học”



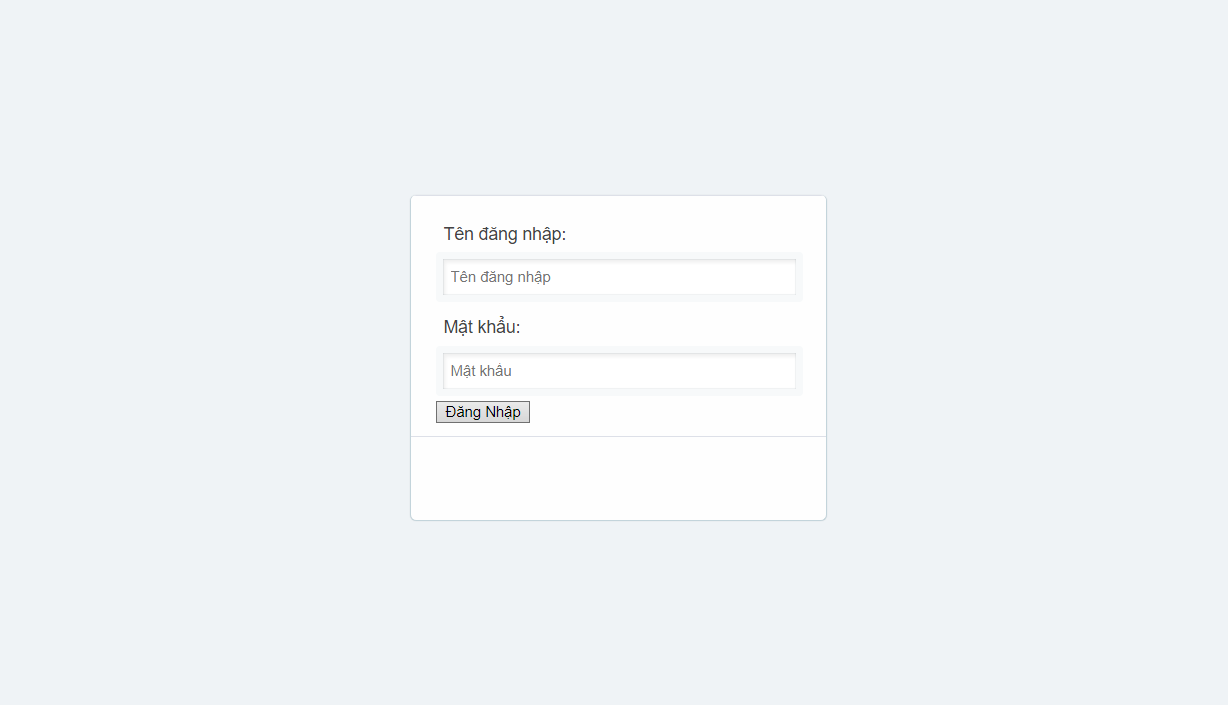
Hình 6: Giao diện đăng nhập vào trang web



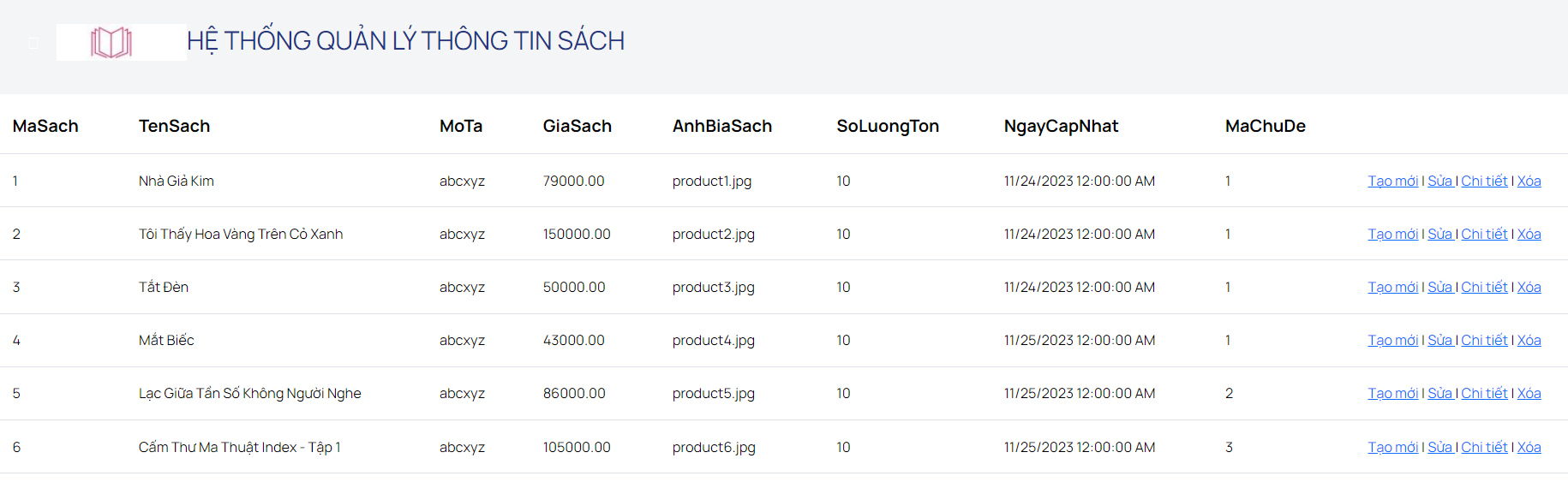
Hình 7: Giao diện đăng ký của trang web



Hình 8: Tính năng phân trang của giao diện trang chủ



Hình 9: Giao diện đăng nhập của Admin



Hình 10: Giao diện quản lý sách



Hình 11: Giao diện thêm sách mới



Hình 12: Giao diện chỉnh sửa sách



Hình 13: Giao diện xóa sách

# CHƯƠNG 4: KẾT LUẬN

4.1 Kết quả đạt được của đề tài

* Xây dựng được website Quản lý và bán sách.
* Giao diện dễ nhìn, dễ sử dụng.
* Thêm được các chức năng cần thiết cho hệ thống: thêm, sửa, xóa thông tin.

4.2 Hạn chế của đề tài

* Chưa thêm được chức năng Giỏ hàng và thanh toán, và 1 số tính năng nâng cao khác.
* Giao diện còn sơ sài, chưa thu hút người xem.

4.3 Hướng phát triển

* Tối ưu trang web, cải thiện giao diện trang web.
* Thêm được cái tính năng nâng cao.

Trong quá trình xây dựng phần mềm với kinh nghiệm và kiến thức còn hạn chế nên báo cáo không tránh khỏi được những sai sót, demo xây dựng còn chưa tốt, rất mong sự góp ý của thầy. Em xin chân thành cảm ơn.

**TÀI LIỆU THAM KHẢO**

* Bùi Chí Thành, Giáo trình phát triển ứng dụng Web.
* <https://tedu.com.vn/khoa-hoc-mien-phi/khoa-hoc-lap-trinh-aspnet-mvc-25.html>
* https://hiepsiit.com/detail/aspxmvc/asp-net-mvc