BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**ĐỒ ÁN PHÁT TRIỂN ỨNG DỤNG WEB 2**

**XÂY DỰNG WEBSITE QUẢN LÝ HOMESTAY TRÁI BƠ (ĐÀ LẠT)**

**Sinh viên thực hiện : Võ Quốc Tỉnh - 63135736**

**Nguyễn Ngọc Gia Hưng - 63133173**

**Lớp : 63.CNTT.CLC**

**Giảng viên hướng dẫn : Mai Cường Thọ**

Khánh Hoà - 2024

# **1. Thu thập yêu cầu**

## **1.1. Mục tiêu**

Hệ thống trang web quản lý homestay Trái Bơ Đà Lạt dùng để đặt phòng online, quản lý phòng, người dùng, hóa đơn….

## **1.2. Lý do chọn đề tài:**

Trái Bơ homestay nằm trên đường Hai Bà Trưng, thành phố Đà Lạt. Trái Bơ homestay chuyên cung cấp các phòng ở lưu trú cỡ nhỏ và vừa cho các cá nhân hay gia đình du lịch xa đến với Đà Lạt muốn có một sự lựa chọn phù hợp về giá cho chỗ ở qua đêm hoặc dài hạn.

Trái Bơ homestay hiện tại đang trong giai đoạn phát triển, Trái Bơ homestay không chỉ muốn dừng lại ở việc đặt booking tại các trang booking lớn. Nên Trái Bơ homestay muốn có một website riêng giúp khách hàng có thuê booking ngay tại website của Trái bơ homestay. Bên cạnh đó website của Trái Bơ homestay có thể giúp quảng cáo chi tiết hơn cho các dịch vụ của mình.

**1.3 Mục tiêu và nhiệm vụ nghiên cứu**

Tạo dựng một website cho Trái Bơ homestay dựa trên các ngôn ngữ lập trình web và framework Spring Boot đáp ứng được các yêu cầu:

- Thiết kế lịch sự, đơn giản, sang trọng, nhưng vẫn đảm bảo sự tinh tế. Thiết kế website phải thể hiện được hình ảnh đẹp của Trái Bơ homestay.

- Tính năng đặt phòng thuận tiện, nhanh chóng.

- Tính năng quản lí đặt phòng.

- Trang giới thiệu Baobab homestay và các dịch vụ của

Baobab homestay...

- Website phải đáp ứng được tính bảo mật.

- Phải dễ sử dụng đối với người quản lí và cả người dùng.

## **1.4 Phương pháp nghiên cứu**

- Nghiên cứu lí thuyết: Nghiên cứu lí thuyết về thiết kế web (thiết kế giao diện với ngôn ngữ HTML, CSS, Javascript,...), lập trình web (ngôn ngữ Java, framework Spring boot CSDL MySQL).

Nghiên cứu thử nghiệm: Phân tích, tìm hiểu các kiểu thiết kế, xu hướng thiết kế cũng như phần mềm quản lí đặt phòng của các website thuộc lĩnh vực Homestay, Khách sạn, Nhà hàng, Khu nghỉ dưỡng để vận dụng lập trình xây dựng webssite cho Trái Bơ homestay một cách hợp lí.

## **1.5 Dự kiến kết quả**

Hoàn thành Website Baobab Homestay với chức năng đặt phòng cơ bản.

## **1.6 Ý nghĩa khoa học và ý nghĩa thực tiễn**

- Ý nghĩa khoa học:

Sinh viên tích lũy thêm kiến thức, kinh nghiệm về lập trình website full-stack. Tìm hiểu thêm về framework Springboot.

- Ý nghĩa thực tiễn .

Giúp Trái Bơ homestay có một hệ thống website đặt phòng hoàn chỉnh của riêng mình.

Ngoài những nội dung như Phần mở đầu, tổng kết, tài liệu tham khảo thì nội dung chính của đồ án gồm các chương như sau:

Chương 1. Cơ sở lý thuyết.

Chương 2. Phân tích thiết kế.

Chương 3. Website Demo

**Chương 1: CƠ SỞ LÝ THUYẾT**

## **1.1. Ngôn ngữ lập trinh Java**

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak.

Java được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

Java được tạo ra với tiêu chí “Viết (code) một lần, thực thi khắp nơi” (Write Once, Run Anywhere – WORA). Chương trình phần mềm viết bằng Java có thể chạy trên mọi nền tảng (platform) khác nhau thông qua một môi trường thực thi với điều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó.

## **1.2 Những đặc trưng cơ bản của Java**

1. Hướng đối tượng (Object-Oriented)

Java là một ngôn ngữ lập trình hướng đối tượng, điều này có nghĩa là nó tập trung vào các đối tượng (objects) và các thao tác trên đối tượng. Các nguyên tắc cơ bản của lập trình hướng đối tượng trong Java bao gồm:

Đóng gói (Encapsulation): Tất cả dữ liệu và các phương thức được đóng gói bên trong các lớp. Điều này giúp bảo vệ dữ liệu và ngăn chặn truy cập trái phép.

Kế thừa (Inheritance): Các lớp có thể kế thừa các thuộc tính và phương thức từ các lớp khác, giúp tái sử dụng mã và mở rộng chức năng.

Đa hình (Polymorphism): Cho phép các đối tượng khác nhau có thể được xử lý thông qua cùng một giao diện, cho phép thực hiện các thao tác khác nhau dựa trên kiểu thực tế của đối tượng.

Trừu tượng (Abstraction): Cung cấp cách để xử lý sự phức tạp bằng cách ẩn đi các chi tiết không cần thiết và chỉ hiển thị những phần quan trọng.

2. Độc lập nền tảng (Platform-Independent)

Một trong những đặc điểm nổi bật nhất của Java là tính độc lập nền tảng. Java biên dịch mã nguồn thành mã bytecode, và mã bytecode này có thể chạy trên bất kỳ nền tảng nào có cài đặt Java Virtual Machine (JVM). Điều này giúp Java đạt được khả năng "Viết một lần, chạy mọi nơi" (Write Once, Run Anywhere).

3. Bộ sưu tập phong phú (Rich Standard Library)

Java cung cấp một bộ thư viện chuẩn phong phú và mạnh mẽ, giúp các lập trình viên dễ dàng phát triển các ứng dụng mà không cần phải viết lại các chức năng cơ bản. Thư viện này bao gồm các lớp hỗ trợ cho:

Xử lý chuỗi ký tự

Xử lý tập tin và nhập/xuất

Giao tiếp mạng

Lập trình đa luồng

Giao diện người dùng đồ họa (GUI)

Cơ sở dữ liệu

4. Bảo mật (Security)

Java được thiết kế với các tính năng bảo mật tích hợp sẵn. JVM chạy trong một môi trường "sandbox" để hạn chế các hoạt động mà mã không tin cậy có thể thực hiện. Ngoài ra, Java còn cung cấp một hệ thống quản lý bảo mật chi tiết cho phép kiểm soát quyền truy cập vào các tài nguyên hệ thống.

5. Đa luồng (Multithreading)

Java hỗ trợ lập trình đa luồng, cho phép nhiều luồng thực thi đồng thời. Điều này giúp các ứng dụng Java có thể xử lý nhiều tác vụ cùng một lúc, cải thiện hiệu suất và khả năng phản hồi.

## **1.3 Ưu và nhược điểm của Java**

**Ưu điểm của Java**

Độc lập nền tảng (Platform-Independent):

Java sử dụng Java Virtual Machine (JVM) để chuyển mã bytecode thành mã máy. Điều này cho phép các ứng dụng Java chạy trên bất kỳ nền tảng nào có JVM, từ Windows, macOS, Linux đến các hệ điều hành di động như Android.

Hướng đối tượng (Object-Oriented):

Java tuân thủ các nguyên tắc lập trình hướng đối tượng (OOP), giúp tạo ra các ứng dụng modular, dễ bảo trì và mở rộng.

Bảo mật cao (High Security):

Java cung cấp nhiều tính năng bảo mật tích hợp như sandboxing và cơ chế quản lý quyền truy cập, giúp bảo vệ ứng dụng khỏi các cuộc tấn công.

Bộ sưu tập phong phú (Rich Standard Library):

Java có một bộ thư viện chuẩn phong phú hỗ trợ nhiều chức năng khác nhau, từ xử lý chuỗi, nhập/xuất, đến lập trình mạng và giao diện người dùng.

Lập trình đa luồng (Multithreading):

Java hỗ trợ lập trình đa luồng, cho phép các ứng dụng thực hiện nhiều tác vụ đồng thời, cải thiện hiệu suất và khả năng phản hồi.

**Nhược điểm của Java**

Hiệu suất thấp hơn so với một số ngôn ngữ khác:

So với ngôn ngữ lập trình như C hoặc C++, Java có thể chậm hơn do phải qua lớp trung gian là JVM. Mặc dù các kỹ thuật tối ưu hóa như Just-In-Time (JIT) compilation đã cải thiện hiệu suất, nhưng Java vẫn không nhanh bằng các ngôn ngữ biên dịch trực tiếp.

Sử dụng nhiều bộ nhớ:

JVM và cơ chế thu gom rác của Java có thể tiêu tốn nhiều bộ nhớ hơn so với các ngôn ngữ khác, đặc biệt là trong các ứng dụng lớn hoặc trên các thiết bị hạn chế tài nguyên.

Giao diện người dùng đồ họa (GUI) không mạnh mẽ:

Mặc dù Java có các thư viện như Swing và JavaFX để tạo giao diện người dùng đồ họa, nhưng chúng không được coi là mạnh mẽ và tiện dụng như các công cụ GUI của các ngôn ngữ khác như C# (với .NET) hoặc JavaScript (với các framework như React hoặc Angular).

Quản lý bộ nhớ không tối ưu cho mọi tình huống:

Mặc dù thu gom rác giúp đơn giản hóa quản lý bộ nhớ, nhưng trong một số trường hợp cụ thể, lập trình viên không có kiểm soát tốt nhất đối với việc giải phóng bộ nhớ, có thể dẫn đến hiệu suất không tối ưu.

Cấu trúc ngôn ngữ phức tạp:

So với một số ngôn ngữ lập trình hiện đại như Python, Java có cú pháp và cấu trúc phức tạp hơn, có thể làm tăng thời gian học tập và phát triển.

## **1.4 Mô hình phát triển MVC**

❖Tổng quan về ngôn ngữ Asp.NET MVC

Asp.NET MVC rất quan trọng trong việc hiểu về cách mà mô hình Model- View

Controller (MVC) được sử dụng để cấu trúc hóa ứng dụng web. Nền tảng này là cột

mốc quan trọng cho việc tạo ra các ứng dụng web dựa trên mô hình MVC thay vì áp

dụng theo kiểu truyền thống của ASP.NET Web Forms. Asp.NET MVC có ưu điểm

nổi bật là nhẹ, giúp việc kiểm thử giao diện trở nên dễ dàng hơn so với việc làm điều

này trong ứng dụng Web Forms. Hơn nữa, nó còn tích hợp mượt mà các tính năng sẵn

có trong ASP.NET, tạo điều kiện thuận lợi cho việc phát triển ứng dụng web.

Nền tảng này được định nghĩa chính xác trong namespace System.Web.Mvc, và là

một phần không thể thiếu của namespace System.Web. Mô hình thiết kế MVC không

chỉ là một khái niệm quen thuộc mà còn là một phương pháp tiếp cận được nhiều lập

trình viên tin dùng. Có những loại ứng dụng web phù hợp với kiến trúc MVC, trong

khi những loại khác vẫn ưu tiên sử dụng ASP.NET Web Forms và cơ chế postbacks.

Đôi khi, các ứng dụng còn kết hợp cả hai kiến trúc này để phù hợp với nhu cầu cụ thể

của dự án.

❖Giớithiệu Asp.NET MVCFramework

ASP.NET MVC Framework là một nền tảng ứng dụng web được Microsoft phát triển,

cho phép nhà phát triển tạo ra các trang web động, ứng dụng web và dịch vụ web. Dựa

trên nền tảng ASP.NET, ASP.NET MVC áp dụng mô hình thiết kế MVC

(Model-View-Controller) để phân chia ứng dụng thành ba thành phần chính.

Models: Các đối tượng Models trong MVC định nghĩa logic cho phần dữ liệu của ứng

dụng. Thông thường, chúng thực hiện việc truy xuất và lưu trữ trạng thái của dữ liệu

trong CSDL. Ví dụ, một đối tượng Product có thể lấy thông tin từ CSDL, thực hiện

các thao tác và cập nhật dữ liệu vào bảng Products trên SQL Server. Trong các ứng

dụng nhỏ, model có thể chỉ là một khái niệm để phân biệt, không nhất thiết phải được

triển khai.

Views: Đây là các thành phần dùng để hiển thị giao diện người dùng (UI). Views

thường được tạo dựa trên thông tin từ các đối tượng model. Ví dụ, một view để cập

nhật bản Products sẽ hiển thị các hộp văn bản, danh sách thả xuống và các ô kiểm

dựa trên trạng thái hiện tại của đối tượng Product.

Controllers: Controllers là thành phần quản lý tương tác với người dùng, làm việc với

model và chọn view để hiển thị giao diện. Trong mô hình MVC, controller quản lý dữ

liệu người dùng gửi lên và truyền giá trị này đến model để xử lý dữ liệu từ CSDL.

Sự phân chia rõ ràng giữa các thành phần của MVC giúp giảm sự phức tạp của ứng

dụng. Logic giao diện thuộc về views, logic nhập liệu nằm trong controllers, và logic

tác vụ (Business logic) thuộc về models. Điều này giúp tập trung vào từng khía cạnh

cụ thể mà không cần quan tâm đến các phần khác.

Mô hình MVC cũng hỗ trợ việc kiểm thử ứng dụng một cách dễ dàng hơn so với mô

hình Web Forms. Việc phân tách thành phần và sử dụng interface cho phép kiểm thử

từng phần riêng biệt mà không cần phụ thuộc vào các yếu tố khác của ứng dụng.

Sự phân rã ba thành phần trong MVC cũng tạo điều kiện cho việc lập trình song song.

Mỗi lập trình viên có thể tập trung vào một phần cụ thể của ứng dụng, làm việc đồng

thời như lập trình view, controller và model. Điều này giúp gia tăng hiệu suất và tập

trung hơn trong quá trình phát triển.

❖Lý do chọn Asp.NET MVC

Chủ yếu bắt nguồn từ những tính năng mạnh mẽ của nó:

Tách bạch các tác vụ và kiểm thử dễ dàng: ASP.NET MVC phân chia rõ ràng các tác

vụ trong ứng dụng như logic nhập liệu, logic kinh doanh và logic giao diện, giúp việc

kiểm thử trở nên thuận tiện. Việc áp dụng hướng phát triển TDD (Test-Driven

Development) dễ dàng thông qua việc cài đặt các tính năng chính dựa trên interface và

sử dụng đối tượng mô phỏng (mock objects) để kiểm thử.

Khả mở rộng và khả nhúng: ASP.NET MVC được thiết kế để linh hoạt và có thể tùy

chỉnh dễ dàng. Bạn có thể thay thế hoặc tùy chỉnh các thành phần như view engine, cơ

chế định tuyến URL, hay cách kết xuất tham số của action-method. Nó cũng hỗ trợ

Dependency Injection (DI) và Inversion of Control (IoC), giúp việc kiểm thử trở nên

dễ dàng hơn thông qua việc gắn các đối tượng vào một lớp thay vì buộc lớp đó tự khởi

tạo các đối tượng.

Ánh xạ URL mạnh mẽ: ASP.NET MVC cho phép xây dựng ứng dụng với địa chỉ

URL ngắn gọn, dễ đọc và tối ưu hóa cho việc tìm kiếm. Các URL không cần có phần

mở rộng của tên tập tin và hỗ trợ mẫu định dạng tên phù hợp với SEO và các tiêu

chuẩn RESTful.

Hỗ trợ các tính năng của ASP.NET: ASP.NET MVC tích hợp các tính năng mạnh mẽ

của ASP.NET như cơ chế xác thực, quản lý thành viên, quyền hạn, caching dữ liệu,

quản lý tình trạng ứng dụng, và nhiều tính năng khác.

Razor View Engine: ASP.NET MVC5 bổ sung Razor View Engine, giúp thiết kế view

nhanh chóng, tiết kiệm công sức hơn so với việc sử dụng Web Forms view engine.

Điều này tạo điều kiện thuận lợi hơn cho quá trình phát triển và quản lý các giao diện

của ứng dụng.

❖Hệ quản trị CSDL SQLServer

Khái niệm hệ quản trị cơ sở dữ liệu SQL Server

SQL Server là một hệ thống quản lý cơ sở dữ liệu (RDBMS) được sử dụng rộng rãi,

chủ yếu dùng Transact-SQL để trao đổi dữ liệu giữa máy tính của người dùng và máy

chủ SQL Server. Hệ thống RDBMS bao gồm các thành phần quan trọng như Cơ sở dữ

liệu (Databases), Bộ động cơ cơ sở dữ liệu (Database engine), và các ứng dụng dùng

để quản lý dữ liệu cũng như các phần khác trong RDBMS.

SQL Server được thiết kế tối ưu để hoạt động trên các môi trường cơ sở dữ liệu rất

lớn, có thể xử lý lượng dữ liệu lên đến hàng Tera-Byte và đồng thời phục vụ cho hàng

ngàn người dùng. Nó có khả năng mở rộng linh hoạt để đáp ứng nhu cầu của môi

trường làm việc có khối lượng dữ liệu lớn.

SQL Server không chỉ đứng độc lập mà còn có khả năng kết hợp hoạt động với các

máy chủ khác như Microsoft Internet Information Server (IIS), ECommerce Server,

hay Proxy Server. Sự linh hoạt này mở ra nhiều khả năng tích hợp và triển khai ứng

dụng, từ các trang web thương mại điện tử đến việc quản lý dữ liệu qua mạng.

Đặc điểm của SQL Server và đối tượng làm việc

SQL (Structured Query Language) là một ngôn ngữ chủ yếu sử dụng tiếng Anh để

tương tác với cơ sở dữ liệu. Điều này làm cho SQL trở thành một ngôn ngữ dễ tiếp

cận và sử dụng cho đa số các lập trình viên.

SQL là một ngôn ngữ không thủ tục, điều này có nghĩa là nó không đòi hỏi bạn phải

chỉ định cách thức truy cập cụ thể vào cơ sở dữ liệu. Những lệnh SQL rất trực quan và

ít gặp lỗi, giúp người sử dụng dễ dàng thao tác và truy vấn dữ liệu.

SQL cung cấp một bộ lệnh phong phú cho nhiều tác vụ liên quan đến dữ liệu:

● Chức năng thêm, cập nhật, xoá các hàng dữ liệu trong một bảng.

● Tạo, chỉnh sửa, thêm và xoá các đối tượng trong cơ sở dữ liệu như bảng, view,

trigger.

● Kiểm soát quyền truy cập vào cơ sở dữ liệu và các đối tượng trong đó để đảm

bảo bảo mật.

● Bảo đảm tính nhất quán và ràng buộc dữ liệu trong cơ sở dữ liệu.

● Đối với việc truy vấn dữ liệu, yêu cầu duy nhất là phải hiểu rõ cấu trúc của cơ

sở dữ liệu mà bạn đang làm việc.

● Nhờ vào tính linh hoạt và sức mạnh của mình, SQL đã trở thành một trong

những ngôn ngữ không thể thiếu trong việc quản lý và truy vấn dữ liệu trong

các hệ thống cơ sở dữ liệu quan trọng.

❖Bootstrap

Bootstrap là một framework front-end mã nguồn mở giúp xây dựng các trang web và

ứng dụng web nhanh chóng và dễ dàng. Nó cung cấp các thành phần giao diện người

dùng sẵn có, hệ thống lưới linh hoạt, thiết kế đáp ứng để hiển thị tốt trên mọi thiết bị,

và khả năng tùy chỉnh cao. Bootstrap cũng đảm bảo tương thích với các trình duyệt

phổ biến và có tài liệu chi tiết cùng với cộng đồng hỗ trợ lớn mạnh. Ban đầu được

phát triển bởi Twitter, Bootstrap hiện tại là một công cụ phổ biến trong cộng đồng phát

triển web.

❖Lý do dùng bootstrap

● Tiết kiệm thời gian và công sức: Bootstrap cung cấp sẵn các thành phần giao

diện người dùng, giúp giảm thiểu thời gian và công sức trong việc thiết kế và

phát triển giao diện web từ đầu.

● Thiết kế đáp ứng (Responsive Design): Với hệ thống lưới linh hoạt và các

thành phần đáp ứng, Bootstrap giúp tạo ra các trang web hiển thị tốt trên mọi

thiết bị và kích thước màn hình, từ điện thoại di động đến máy tính để bàn.

● Tính nhất quán: Sử dụng Bootstrap đảm bảo sự nhất quán trong thiết kế giao

diện người dùng, vì các thành phần và phong cách đã được chuẩn hóa và kiểM

tra kỹ lưỡng.

● Dễ dàng tùy chỉnh: Bootstrap có thể được tùy chỉnh theo nhu cầu cụ thể của dự

án, từ việc thay đổi màu sắc, kiểu chữ đến các biến số trong tệp CSS hoặc sử

dụng phiên bản Sass để tùy chỉnh sâu hơn.

● Hệ sinh thái phong phú: Bootstrap có một hệ sinh thái phong phú với nhiều tiện

ích mở rộng, plugin và các template sẵn có, giúp mở rộng khả năng và tính

năng của trang web.

● Tài liệu và cộng đồng hỗ trợ mạnh mẽ: Bootstrap đi kèm với tài liệu chi tiết và

dễ hiểu, cùng với cộng đồng lớn mạnh giúp hỗ trợ và chia sẻ kiến thức, giúp

giải quyết các vấn đề một cách nhanh chóng.

● Tương thích với các trình duyệt: Bootstrap được thiết kế để hoạt động tốt trên

các trình duyệt phổ biến hiện nay, đảm bảo trải nghiệm người dùng nhất quán

trên các nền tảng khác nhau.

● Mã nguồn mở và miễn phí: Là một dự án mã nguồn mở, Bootstrap hoàn toàn

miễn phí để sử dụng và có thể được đóng góp và cải tiến bởi cộng đồng phát

triển toàn cầu

# **Chương 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG**

# **2.1 Đặt vấn đề**

Trong bối cảnh phát triển mạnh mẽ của ngành du lịch hiện nay, nhu cầu tìm kiếm và lưu trú tại các homestay ngày càng tăng cao. Homestay không chỉ mang lại cho du khách trải nghiệm gần gũi, thân thiện với người dân địa phương mà còn giúp họ khám phá văn hóa, phong tục tập quán đặc trưng của từng vùng miền. Tuy nhiên, việc quản lí homestay gặp nhiều khó khăn, từ việc quản lí đặt phòng, lịch trình khách đến, dịch vụ đi kèm cho đến xử lí các vấn đề phát sinh trong quá trình lưu trú.

Bây giờ có rất nhiều dịch vụ như booking (Đặt phòng trước), đa phần các home stay sẽ lựa chọn dịch vụ này để quản lí, hoặc tiếp cận đến với khách hàng.Để đáp ứng nhu cầu đó, việc ứng dụng công nghệ vào quản lí homestay trở nên vô cùng cần thiết. Một hệ thống quản lí homestay hiệu quả không chỉ giúp chủ homestay tiết kiệm thời gian, công sức mà còn nâng cao chất lượng dịch vụ ở hiện tại và tương lai, tối ưu hóa quy trình hoạt động và tạo sự hài lòng cho khách hàng. Chính vì vậy, việc phát triển một trang web quản lí homestay với các tính năng thông minh, tiện ích và dễ sử dụng là điều cần thiết để hỗ trợ các chủ homestay trong việc quản lí và phát triển kinh doanh một cách bền vững.

# **2.2 Danh sách các tác nhân sử dụng hệ thống**

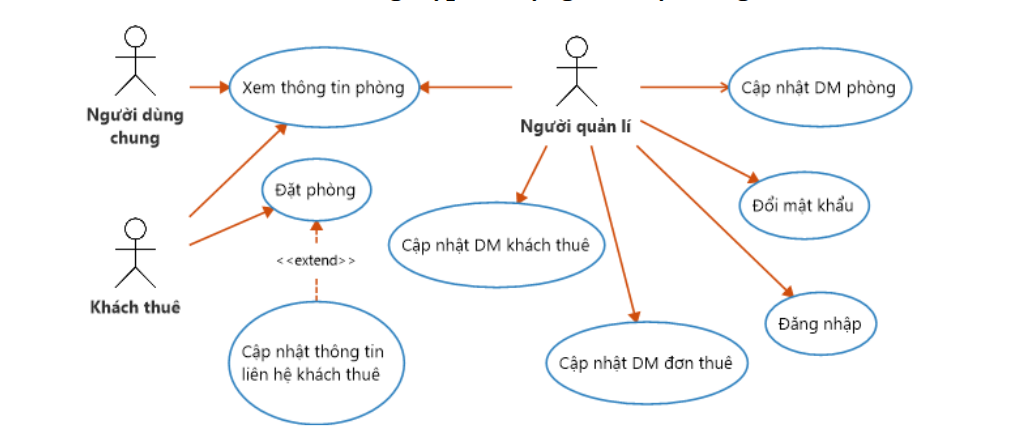
| STT | Tác nhân | Ý nghĩa |
| --- | --- | --- |
| 1 | Người dùng chung | Người truy cập vào website |
| 2 | Khách thuê | Người truy cập vào website và đặt phòng trên website |
| 3 | Người quản lí | Người quản lý hệ thống(Chủ homestay) |

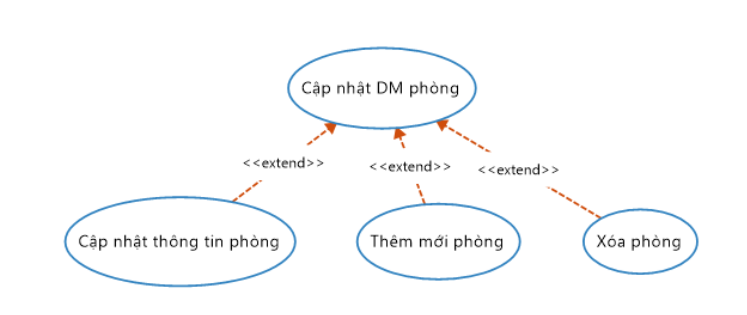
- Người dùng chung: tác nhân có thể xem các thông tin trên website như thông tin phòng ở nhưng không đặt phòng.

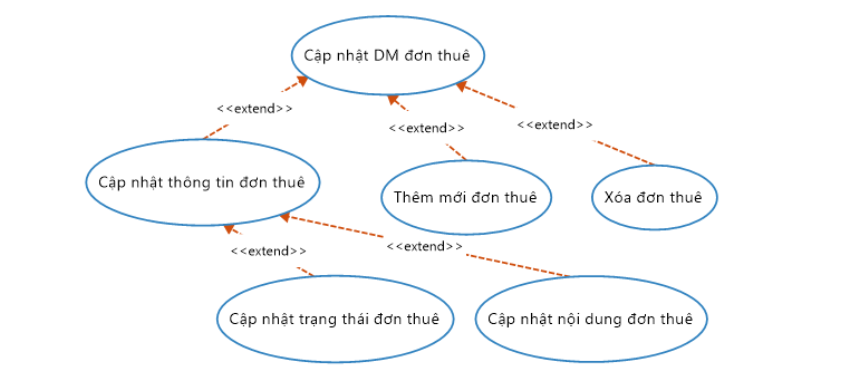
- Khách thuê: tác nhân có thể xem các thông tin trên website giống người dùng chung nhưng là người đặt phòng trực tuyến. Có thể đã từng đặt trước đó hoặc là khách thuê mới.

- Người quản lí: tác nhân có thể sử dụng các chức năng của hệ thống như cập nhật thông tin phòng, kiểm tra thông tin các đơn thuê phòng, kiểm tra hoặc cập nhật thông tin khách thuê,...

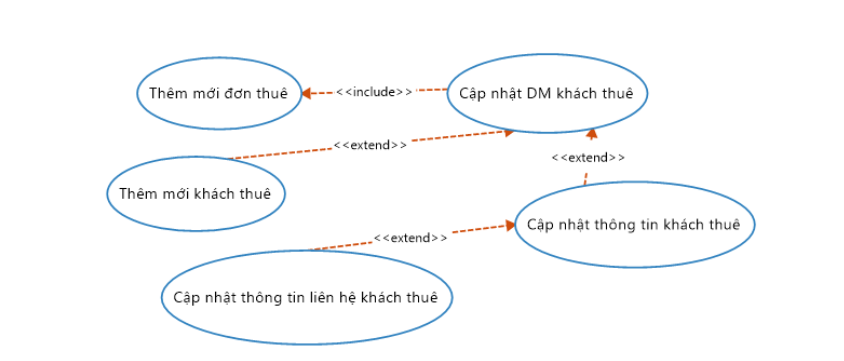
# **2.3 Các biểu đồ trường hợp của sử dụng hệ thống**



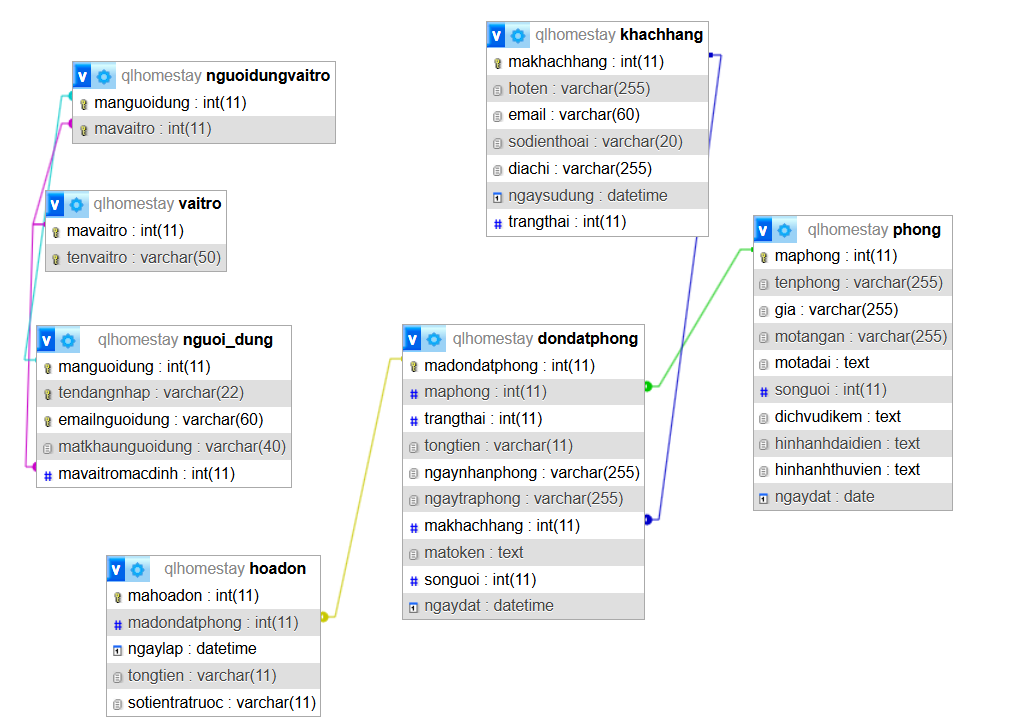








# **2.4 Biểu đồ lớp**



# **2.5 Từ điển cơ sở dữ liệu**

Bảng phong

| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| maphong | int(11) | mã phòng(khóa chinh) |
| tenphong | varchar(255) | tên phòng |
| giaphong | vachar(255) | giá phòng |
| motangan | varchar(255) | mô tả ngắn |
| motadai | varchar(255) | mô tả dài |
| songuoi | int(11) | số người |
| dichvudikem | text | dịch vụ đi kèm |
| hinhanhdaidien | text | hình ảnh đại diện |
| hinhanhthuvien | text | hình ảnh đại diện |

bảng khách hàng

| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| makhachhang | int(11) | mã khách hàng(khóa chính) |
| hoten | vachar(255) | họ tên |
| email | varchar(60) | email |
| sodienthoai | varchar(20) | số điện thoại |
| diachi | vachar(255) | địa chỉ |
| trangthai | int(11) | trạng thái |

bảng nguoi\_dung

| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| manguoidung | int(11) | mã người dùng |
| tendangnhap | vachar(22) | tên đăng nhập |
| emailnguoidung | varchar(60) | email người dùng |
| matkhaunguoidung | varchar(40) | mật khẩu người dùng |
| mavaitromacdinh | int(11) | mã vai trò mặc định(khóa ngoại) |

bảng nguoidungvaitro

| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| manguoidung | int(11) | mã người dùng |
| mavaitro | int(11) | mã vai trò |

bảng hoadon

| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| mahoadon | int(11) | mã hóa đơn |
| dondatphong | int(11) | đơn đặt phòng |
| ngaylap | datetime | ngày lập |
| tongtien | varchar(11) | tổng tiền |
| sotientratruoc | varchar(11) | số tiền trả trước |

bảng vaitro

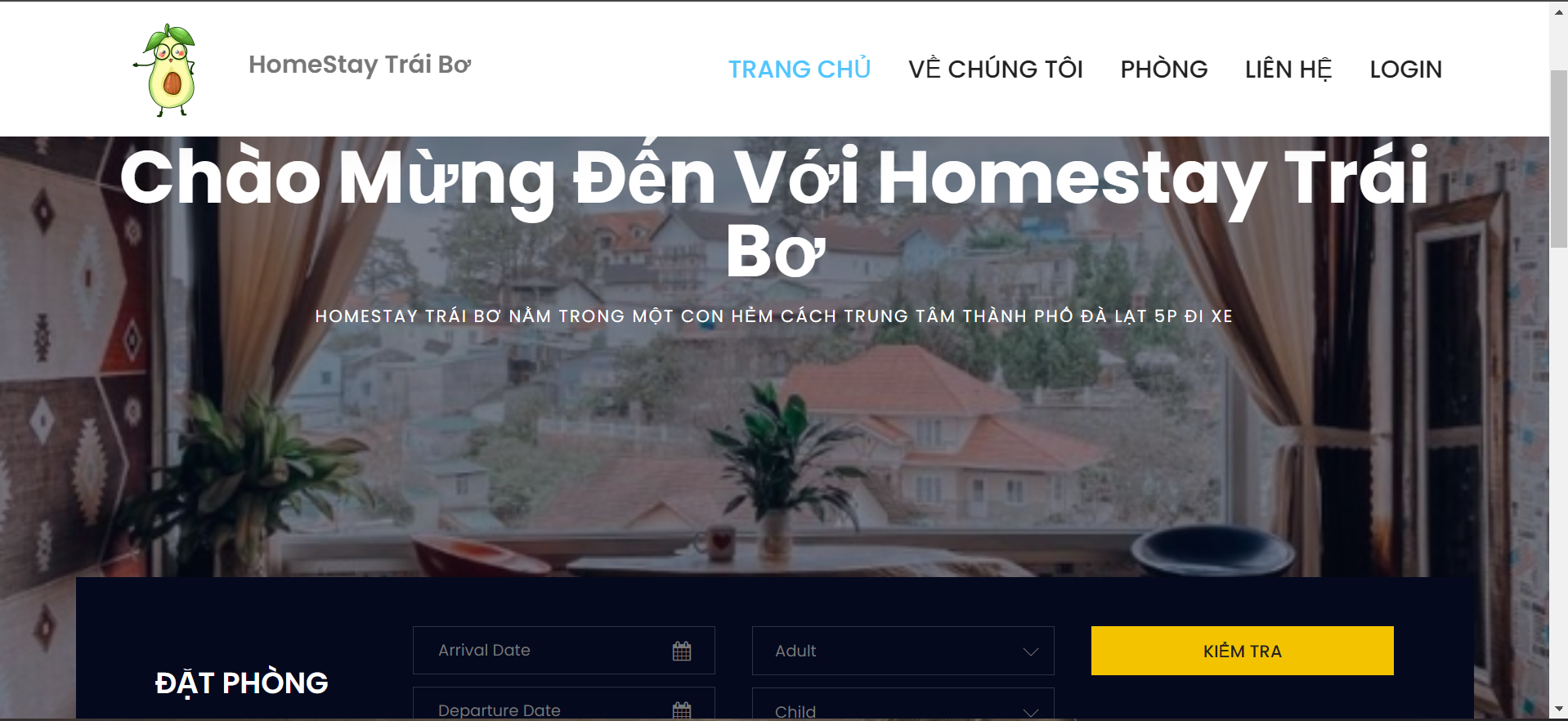
| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| mavaitro | int(11) | mã vai trò |
| tenvaitro | varchar(50) | tên vai trò |

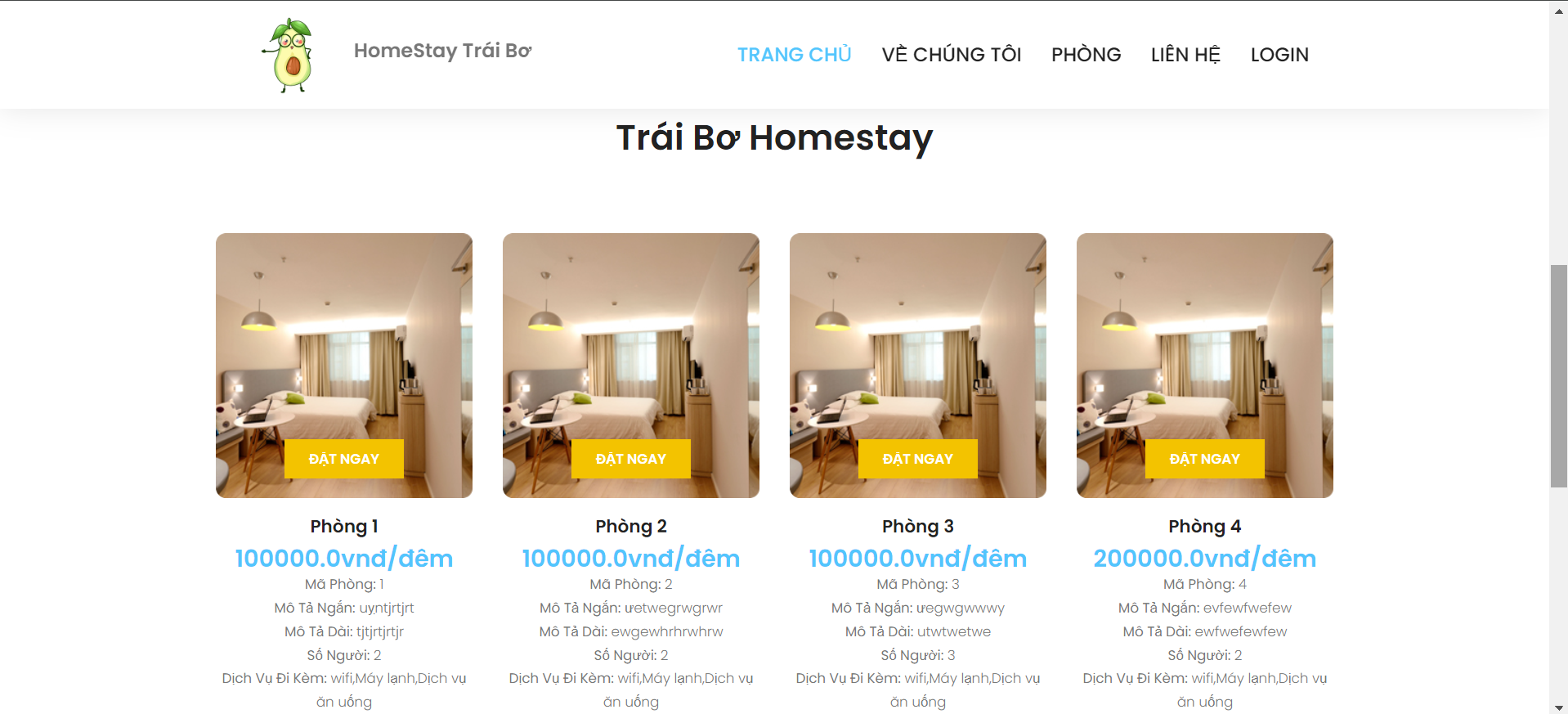
bảng dondatphong

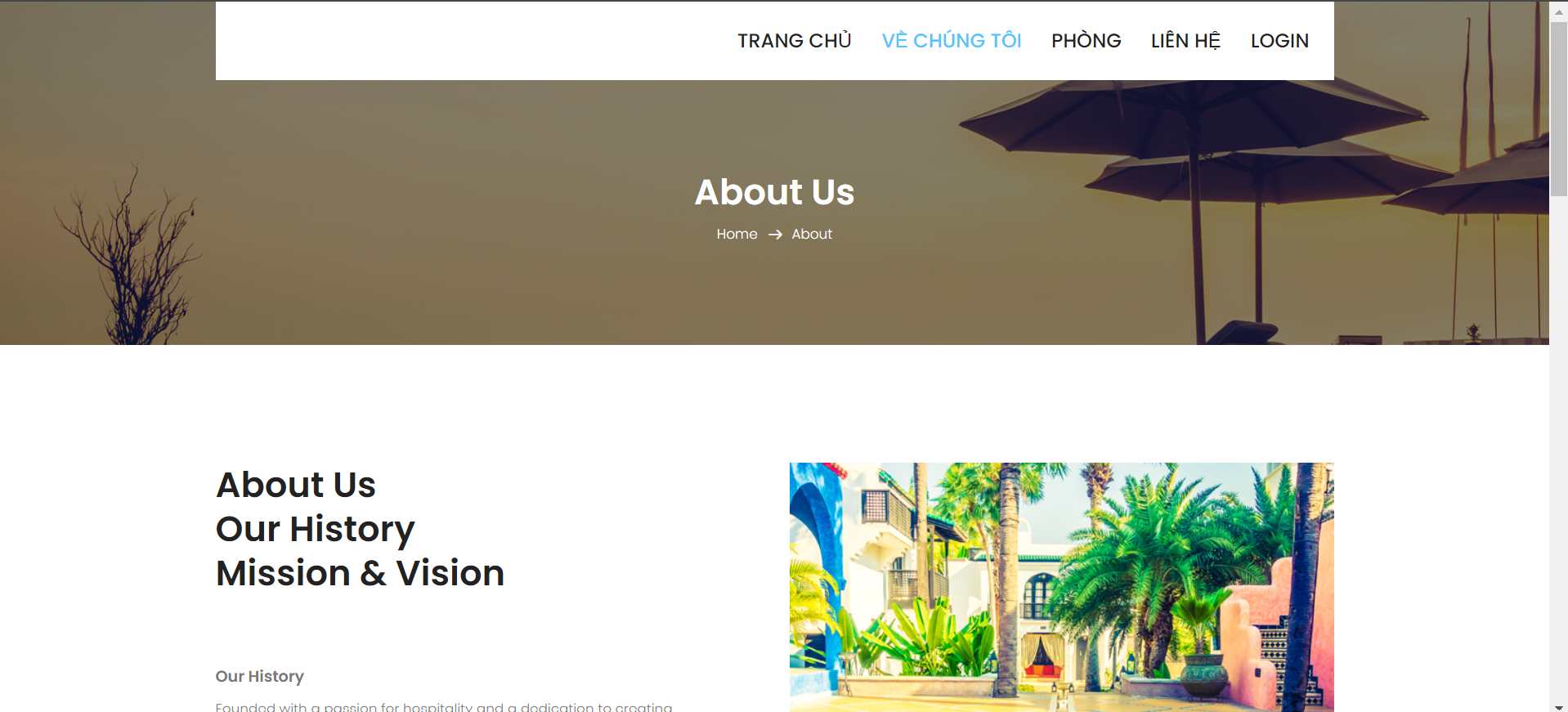
| Tên thuộc tính | Kiểu dữ liệu | Chú thích |
| --- | --- | --- |
| madondatphong | int(11) | mã hóa đơn |
| maphong | int(11) | đơn đặt phòng |
| trangthai | int(11) | ngày lập |
| tongtien | varchar(11) | tổng tiền |
| ngaynhanphong | varchar(255) | số tiền trả trước |
| ngaytraphong | varchar(255) | ngày trả phòng |
| makhachhang | int(11) | mã khách hàng |
| matoken | text | mã token |
| songuoi | int(11) | số người |
| ngaydat | datetime | ngày đặt |

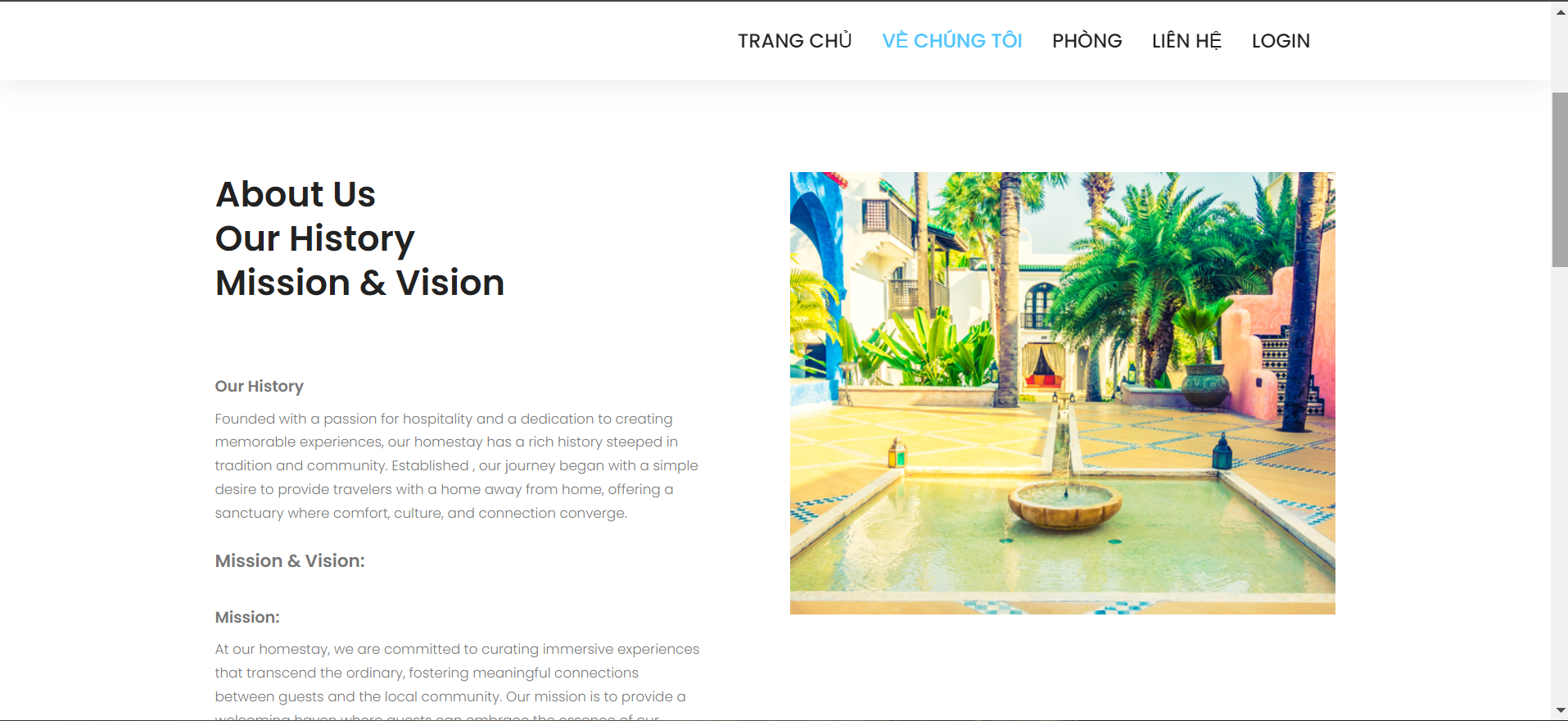
**Chương 3: Website Demo**

# **3.1 Trang cho khách hàng**

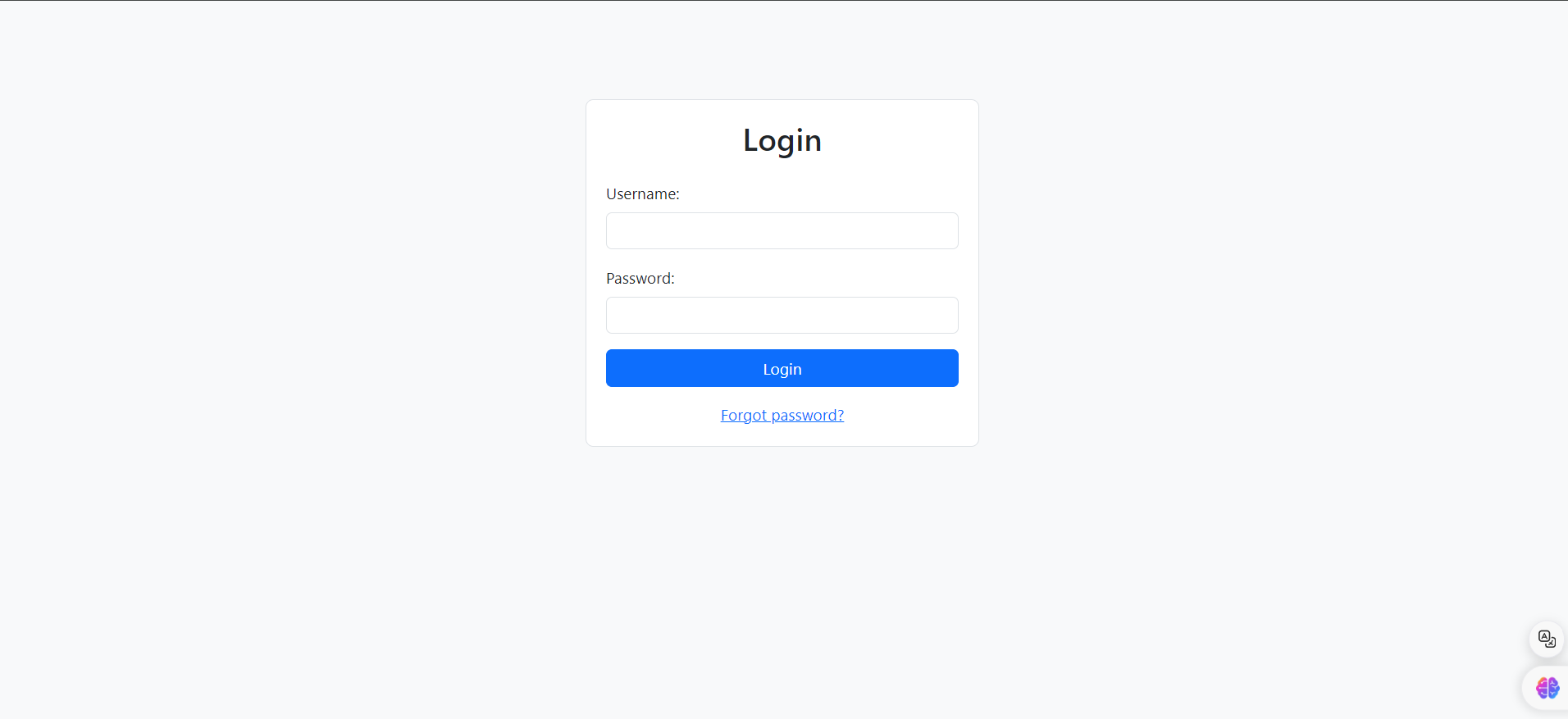


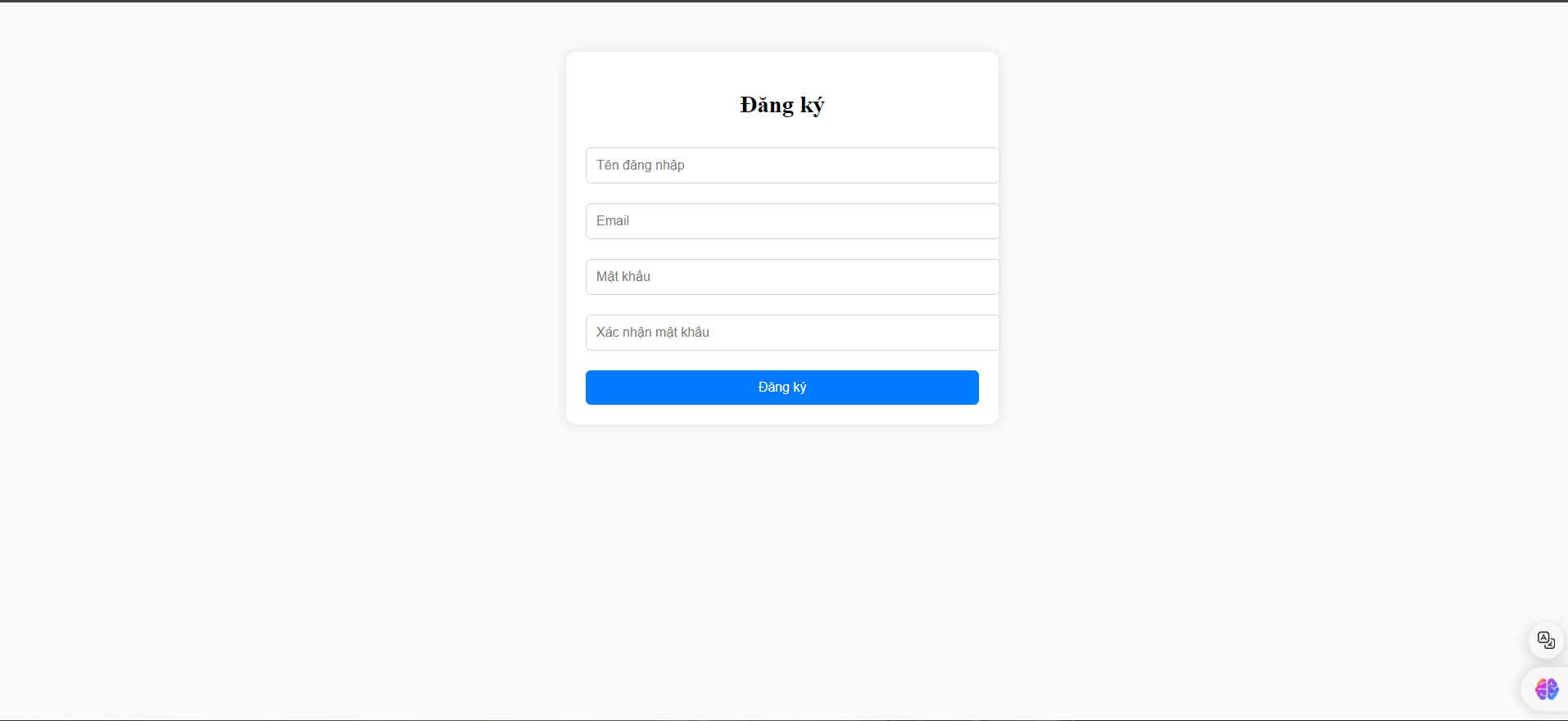






# **3.2 Đăng ký, đăng nhập**





# **3.3 Trang quản trị**

