KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

CHUYÊN ĐỀ ASP.NET

HỌC KỲ 5, NĂM HỌC 2025

ĐỀ TÀI:

XÂY DỰNG WEBSITE ĐẶT DỊCH VỤ CHĂM SÓC THÚ CƯNG TẠI NHÀ

Giáo viên hướng dẫn:
Họ tên: ThS. Đoàn Phước Miền

Sinh viên thực hiện:
Họ tên: Hồ Thanh Phong
MSSV: 170123426
Lớp: DK23TTC11

Trà Vinh, tháng 5 năm 2025

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Thành viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Em xin chân thành gửi lời cảm ơn sâu sắc đến quý thầy cô Bộ môn Công nghệ Thông tin, Trường Đại học Trà Vinh – những người đã không chỉ truyền dạy kiến thức chuyên môn mà còn hun đúc trong em tinh thần học hỏi, sự kiên trì và niềm đam mê với ngành nghề mà em đã chọn.

Em đặc biệt biết ơn thầy Đoàn Phước Miền, người đã dành nhiều thời gian, tâm huyết và sự tận tụy trong việc hướng dẫn, góp ý và đồng hành cùng em trong suốt quá trình thực hiện đồ án. Chính sự nhiệt tình, nghiêm túc và tận tâm của thầy đã giúp em từng bước vượt qua khó khăn, hoàn thiện đề tài này.

Mặc dù đã cố gắng hết sức, nhưng do hạn chế về kiến thức và kinh nghiệm thực tiễn, bài làm chắc chắn không tránh khỏi những thiếu sót. Em rất mong nhận được những ý kiến đóng góp chân thành từ quý thầy cô để em có thể học hỏi và hoàn thiện hơn trên con đường học tập và nghề nghiệp sau này.

Em xin chân thành cảm ơn!

MỤC LỤC

DANH MỤC HÌNH ẢNH – BẢNG BIỂU

TÓM TẮT

Về lý thuyết:

- Ứng dụng phát triển theo mô hình ASP.NET MVC.

- Sử dụng Entity Framework Core để thao tác dữ liệu dưới dạng ORM.

- Cơ sở dữ liệu chính: SQL Server.

- Tận dụng các tính năng: Routing, Razor View Engine, Model Binding trong ASP.NET Core.

- Áp dụng ASP.NET Identity để phân quyền người dùng.

- Hỗ trợ xử lý thời gian thực với SignalR.

Về thực nghiệm:

- Xây dựng website đặt dịch vụ chăm sóc thú cưng tại nhà, bao gồm: đặt lịch, quản lý dịch vụ.

- Dịch vụ phân loại theo các danh mục, hỗ trợ tìm kiếm tiện lợi.

- Giao diện thiết kế hiện đại, hỗ trợ responsive trên đa nền tảng.

- Quản trị viên có thể thêm/sửa/xóa dữ liệu động qua giao diện web.

- Dữ liệu xử lý qua Razor View, lưu trữ và truy xuất từ SQL Server.

MỞ ĐẦU

1. Lý do chọn đề tài

Hiện nay, nhu cầu chăm sóc thú cưng tại nhà ngày càng phổ biến trong nhiều hộ gia đình tại Việt Nam. Người nuôi không chỉ cần các dịch vụ y tế, làm đẹp cho thú cưng mà còn mong muốn được phục vụ tại nhà để thuận tiện và tiết kiệm thời gian. Tuy nhiên, thị trường dịch vụ này vẫn còn phân tán, chưa có nền tảng đặt dịch vụ chuyên biệt.

Nhằm giải quyết vấn đề đó, việc xây dựng một website đặt dịch vụ chăm sóc thú cưng tại nhà không chỉ đáp ứng nhu cầu thực tiễn mà còn là cơ hội ứng dụng những kiến thức đã học về ASP.NET Core MVC, Entity Framework, và các công nghệ web hiện đại như SignalR, AJAX, SQL Server vào một dự án thực tế. Vì vậy, em quyết định chọn đề tài: **"Xây dựng website đặt dịch vụ chăm sóc thú cưng tại nhà".**

Với mong muốn vận dụng được những kiến thức đã học vào thực tế, qua đó làm quen và đúc kết kinh nghiệm để sau này có thể xây dựng được các website thương mại, đáp ứng nhu cầu xã hội.

2. Mục tiêu nghiên cứu

Mục tiêu của đồ án là xây dựng một website cho phép người dùng:

- Đặt lịch dịch vụ chăm sóc thú cưng tại nhà (cạo lông, vệ sinh, khám định kỳ...).

- Tìm kiếm, xem chi tiết thông tin từng thú cưng, dịch vụ, và danh mục phân loại.

- Hỗ trợ xử lý thời gian thực và cải thiện trải nghiệm người dùng bằng SignalR.

3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu gồm:

- Các công nghệ lập trình web: ASP.NET Core MVC, Entity Framework, Razor View, SignalR.

- Các thao tác CRUD trong môi trường web.

- Cơ sở dữ liệu sử dụng SQL Server.

Phạm vi nghiên cứu:

- Website phục vụ hai nhóm người dùng chính: người đặt dịch vụ và quản trị viên.

- Người dùng có thể đăng ký, đăng nhập và đặt dịch vụ nhanh chóng.

- Quản trị viên có thể thêm/sửa/xóa các dịch vụ chăm sóc thú cưng, phân loại danh mục, và theo dõi các yêu cầu đặt lịch từ người dùng.

- Website triển khai mô hình MVC, xử lý dữ liệu qua Entity Framework và hỗ trợ responsive UI.

CHƯƠNG 1 TỔNG QUAN

1.1 Giới thiệu phần mềm mã nguồn mở

1.1.1 Định nghĩa

Phần mềm nguồn mở là những phần mềm mà mã nguồn được công khai, cho phép người dùng sử dụng, sửa đổi, phân phối lại mà không phải trả phí bản quyền. Các phần mềm nguồn mở mang lại sự linh hoạt cao, tiết kiệm chi phí và có cộng đồng phát triển rộng lớn hỗ trợ cập nhật, vá lỗi thường xuyên.

1.1.2 Ưu điểm của phần mềm nguồn mở

- Giảm chi phí triển khai và bảo trì.

- Dễ dàng tùy biến theo nhu cầu dự án.

- Cộng đồng lớn hỗ trợ nhanh chóng.

- Minh bạch và linh hoạt trong phát triển.

1.1.3 Lịch sử của phần mềm mã nguồn mở

Các cột mốc quan trọng trong lịch sử phần mềm mã nguồn mở bao gồm:

- 1984: Richard Stallman sáng lập dự án GNU (GNU’s Not Unix) – khởi đầu cho phong trào phần mềm tự do.

- 1991: Linus Torvalds phát hành phiên bản đầu tiên của Linux Kernel.

- 1997: GNU/Linux chiếm khoảng 25% thị phần máy chủ trên toàn thế giới.

- 1998: Công ty Netscape công bố mã nguồn trình duyệt Navigator, đánh dấu sự ra đời của thuật ngữ “Mã nguồn mở”.

- Thành lập OSI (Open Source Initiative) nhằm quảng bá phần mềm mã nguồn mở và chuẩn hóa các giấy phép đi kèm.

1.1.4 Lợi ích của phần mềm mã nguồn mở

Phần mềm mã nguồn mở mang lại nhiều lợi ích nổi bật như:

- Tính kinh tế: Không tốn chi phí bản quyền, phù hợp cho cá nhân và tổ chức nhỏ.

- Tính linh hoạt và tự do: Người dùng có quyền truy cập, chỉnh sửa, và phân phối lại mã nguồn.

- Tính an toàn: Do cộng đồng phát triển và kiểm tra liên tục nên lỗ hổng bảo mật dễ được phát hiện và khắc phục.

- Khả năng mở rộng và tích hợp tốt: Dễ dàng mở rộng và tích hợp với các hệ thống khác.

- Hỗ trợ từ cộng đồng mạnh mẽ: Nhiều diễn đàn, tài liệu và cộng đồng hỗ trợ rộng khắp.

1.1.5 Hạn chế của phần mềm mã nguồn mở

Dù có nhiều ưu điểm, phần mềm mã nguồn mở vẫn tồn tại một số hạn chế:

- Giao diện chưa thân thiện: Một số phần mềm có UI chưa trực quan hoặc kém hiện đại.

- Hạn chế về hỗ trợ kỹ thuật: Chủ yếu phụ thuộc vào cộng đồng, thiếu hỗ trợ chính thức.

- Không phù hợp với mọi đối tượng: Đôi khi khó tiếp cận với người dùng phổ thông hoặc doanh nghiệp lớn.

1.1.6 Những dự án mã nguồn mở thành công

Một số dự án mã nguồn mở tiêu biểu:

- Linux – hệ điều hành mã nguồn mở phổ biến nhất.

- Apache – máy chủ web được sử dụng rộng rãi.

- MySQL / PostgreSQL – hệ quản trị cơ sở dữ liệu.

- Mozilla Firefox – trình duyệt web mã nguồn mở.

- LibreOffice / OpenOffice – bộ công cụ văn phòng.

- VS Code – trình chỉnh sửa mã nguồn miễn phí của Microsoft.

1.1.7 Các loại giấy phép phần mềm mã nguồn mở

Một số giấy phép phổ biến trong mã nguồn mở:

- GPL (General Public License): Cho phép sử dụng, chỉnh sửa và phân phối lại miễn phí, với điều kiện mã nguồn phải được công khai.

- MIT License: Rất tự do, chỉ yêu cầu ghi nhận bản quyền.

- Apache License: Tương tự MIT, nhưng có điều khoản rõ hơn về bằng sáng chế.

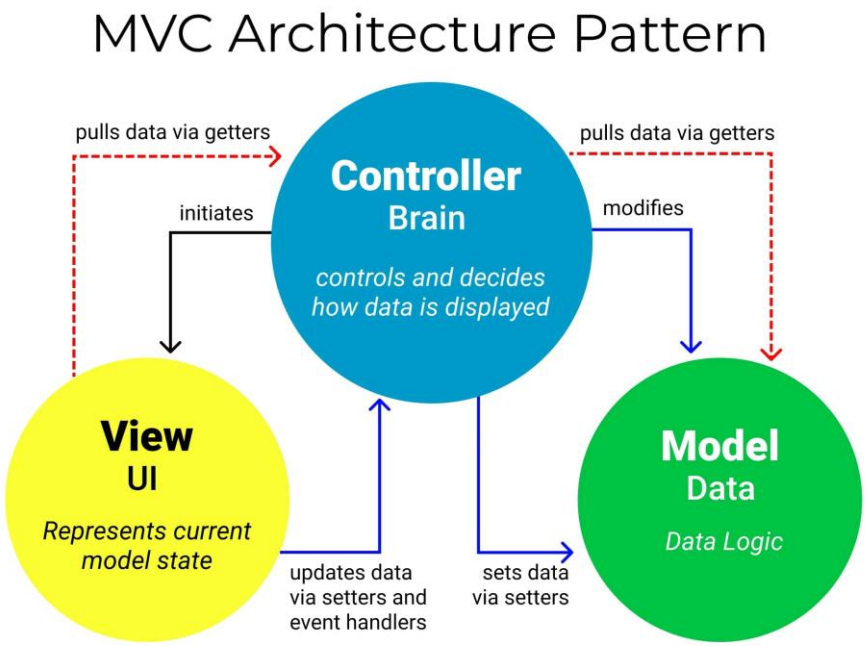
- BSD License: Đơn giản, cho phép tái sử dụng ngay cả trong phần mềm đóng.

- Mozilla Public License (MPL): Kết hợp giữa mở và đóng – mã được chỉnh sửa phải công khai, nhưng phần khác thì không bắt buộc.

1.2 Giới thiệu mô hình MVC

1.2.1 Tổng quát

- Mô hình MVC (Model – View – Controller) là một kiến trúc phần mềm phổ biến trong phát triển ứng dụng web. MVC phân tách các thành phần logic, giao diện, và xử lý điều khiển thành 3 lớp riêng biệt, giúp tổ chức mã nguồn rõ ràng, dễ bảo trì và mở rộng.



Hình 1.2.1-1 Mô hình MVC

- **Model**: Đại diện cho dữ liệu và logic nghiệp vụ. Là nơi xử lý tương tác với cơ sở dữ liệu, định nghĩa các đối tượng, và thực hiện các phép toán nghiệp vụ.

- **View**: Giao diện người dùng. Hiển thị dữ liệu từ Model và nhận tương tác từ người dùng. Trong ASP.NET Core, Razor View là công nghệ hiển thị chính.

- **Controller**: Điều phối hoạt động giữa Model và View. Nhận yêu cầu từ người dùng, xử lý logic phù hợp và trả về View kèm theo dữ liệu.

1.2.2 Ưu và nhược điểm

Ưu điểm:

- Tách biệt rõ ràng giữa giao diện, xử lý logic và dữ liệu.

- Dễ bảo trì, mở rộng, kiểm thử từng phần.

- Tái sử dụng được các thành phần code.

- Phù hợp cho ứng dụng có quy mô trung bình đến lớn.

Nhược điểm:

- Cấu trúc ban đầu phức tạp hơn các mô hình truyền thống.

- Đòi hỏi người lập trình hiểu rõ kiến trúc và nguyên tắc phân chia vai trò.

- Với dự án nhỏ, MVC có thể gây dư thừa về tổ chức mã.

CHƯƠNG 2 NGHIÊN CỨU LÝ THUYẾT

2.1 Tìm hiểu về nền tảng ASP.NET

2.1.1 Giới thiệu

ASP.NET là một nền tảng phát triển ứng dụng web mã nguồn mở do Microsoft phát triển, chạy trên nền tảng .NET. ASP.NET cho phép tạo ra các trang web động, dịch vụ web, và ứng dụng web hiện đại một cách nhanh chóng và bảo mật [1].

Hình 2.1.1-1 ASP.NET

Với sự hỗ trợ mạnh mẽ từ Visual Studio, ASP.NET giúp lập trình viên dễ dàng xây dựng ứng dụng theo mô hình MVC, Web API hoặc Razor Pages. Phiên bản ASP.NET Core còn hỗ trợ đa nền tảng (Windows, Linux, macOS) và tối ưu hóa hiệu suất.

2.1.2 Tính năng

- Hỗ trợ lập trình hướng đối tượng và tích hợp tốt với C# hoặc VB.NET.

- Tương thích với nhiều cơ sở dữ liệu như SQL Server, MySQL,...

- Hỗ trợ SignalR để xử lý realtime.

- Quản lý session, cookie, xác thực người dùng đơn giản.

- Có khả năng mở rộng, bảo mật tốt và hiệu năng cao.

2.1.3 Ưu và nhược điểm

Ưu điểm:

- Hiệu năng cao nhờ vào cơ chế biên dịch và caching [2].

- Tích hợp tốt với Visual Studio – môi trường phát triển mạnh mẽ.

- Hỗ trợ mô hình MVC tách biệt rõ logic – giao diện – dữ liệu.

- Hỗ trợ bảo mật: xác thực, phân quyền, mã hóa,...

- Dễ dàng bảo trì và mở rộng ứng dụng theo thời gian.

Nhược điểm:

- Yêu cầu môi trường Windows khi dùng bản ASP.NET truyền thống.

- Độ phức tạp ban đầu cao, cần kiến thức nền tảng vững.

- Hosting cho ASP.NET Core có thể ít phổ biến hơn so với PHP [3].

2.2 Các công cụ và công nghệ bổ trợ

2.2.1 SQL Server

SQL Server là hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System – RDBMS) do Microsoft phát triển. Đây là một nền tảng mạnh mẽ được sử dụng rộng rãi trong các hệ thống doanh nghiệp để lưu trữ và xử lý dữ liệu có cấu trúc.

Trong đồ án này, SQL Server đóng vai trò là nơi lưu trữ toàn bộ dữ liệu liên quan đến người dùng, dịch vụ, và lịch đặt hẹn chăm sóc thú cưng tại nhà. Việc tích hợp SQL Server với ASP.NET Core thông qua Entity Framework Core giúp tối ưu việc truy vấn, cập nhật và xóa dữ liệu mà không cần viết thủ công các câu lệnh SQL phức tạp.

Ngoài ra, SQL Server hỗ trợ tốt bảo mật, phân quyền và khả năng mở rộng cho các ứng dụng web quy mô vừa và lớn [4].

2.2.2 Entity Framework Core

Entity Framework Core (EF Core) là một thư viện ORM (Object-Relational Mapping) mã nguồn mở của Microsoft, giúp lập trình viên làm việc với cơ sở dữ liệu thông qua các đối tượng C# thay vì sử dụng SQL thuần.

EF Core hỗ trợ nhiều cơ sở dữ liệu khác nhau, trong đó SQL Server là mặc định. Với EF Core, lập trình viên có thể dễ dàng thao tác các bảng dữ liệu như một tập hợp các đối tượng trong ứng dụng – giúp mã nguồn rõ ràng, dễ bảo trì.

Trong dự án này, EF Core được sử dụng để thực hiện các thao tác CRUD (Create, Read, Update, Delete) như: thêm dịch vụ, cập nhật thông tin đặt lịch, xóa yêu cầu... bằng cách sử dụng các DbContext, DbSet, và LINQ [5].

2.2.3 SignalR

SignalR là một thư viện của ASP.NET Core cho phép truyền dữ liệu thời gian thực (real-time) giữa server và client. Điều này có nghĩa là bất kỳ thay đổi nào từ phía máy chủ đều có thể đẩy đến trình duyệt của người dùng ngay lập tức mà không cần reload trang.

Trong đề tài này, SignalR được ứng dụng để cập nhật realtime các lịch đặt dịch vụ chăm sóc thú cưng – ví dụ như khi người dùng gửi yêu cầu đặt lịch, quản trị viên sẽ nhận được thông báo ngay lập tức tại trang quản lý.

SignalR giúp nâng cao trải nghiệm người dùng và đảm bảo rằng hệ thống phản hồi nhanh, phù hợp với các hệ thống đòi hỏi cập nhật liên tục như bảng gọi dịch vụ, hệ thống chat hoặc thông báo nội bộ [6].

2.2.4 Visual Studio

Visual Studio là môi trường phát triển tích hợp (IDE) mạnh mẽ do Microsoft phát triển, hỗ trợ tốt cho việc xây dựng ứng dụng ASP.NET Core MVC.

Nó cung cấp đầy đủ công cụ như: trình soạn thảo mã, debugger, giao diện quản lý project, tích hợp Git, công cụ quản lý database, tạo migration cho Entity Framework, cũng như khả năng publish trực tiếp website.

Trong quá trình thực hiện đồ án, Visual Studio giúp tăng tốc độ lập trình, kiểm tra lỗi cú pháp, hỗ trợ refactor code và tổ chức cấu trúc thư mục rõ ràng. Đây là IDE lý tưởng cho lập trình viên .NET chuyên nghiệp.

2.3 Giao diện và công nghệ phía người dùng

2.3.1 Bootstrap

Bootstrap là một framework CSS mã nguồn mở, giúp xây dựng giao diện web hiện đại và responsive một cách dễ dàng. Với hệ thống lưới linh hoạt (Grid System), Bootstrap cho phép bố cục trang web hiển thị tốt trên nhiều loại thiết bị, từ desktop đến smartphone. Ngoài ra, thư viện còn cung cấp hàng trăm thành phần UI sẵn có như button, alert, modal, form… giúp tăng tốc quá trình phát triển mà vẫn đảm bảo tính nhất quán về thiết kế giao diện [7].

Trong đồ án này, Bootstrap được sử dụng để xây dựng toàn bộ giao diện người dùng, bao gồm cả trang người dùng và trang quản trị, đảm bảo tính trực quan và dễ sử dụng cho cả quản trị viên lẫn khách hàng.

2.3.2 JQuery

jQuery là thư viện JavaScript phổ biến, giúp đơn giản hóa các thao tác xử lý DOM, sự kiện (event), hiệu ứng (animation) và Ajax. Với cú pháp ngắn gọn và thân thiện, jQuery giúp giảm thời gian phát triển các chức năng tương tác trên trình duyệt [8].

2.3.3 Razor View Engine

Razor là công cụ hỗ trợ tạo giao diện (View Engine) được tích hợp sẵn trong ASP.NET MVC. Razor cho phép chèn trực tiếp mã C# vào HTML thông qua cú pháp @, giúp xử lý động dữ liệu hiển thị và tối ưu quá trình render phía server. Nó có tốc độ xử lý nhanh và giảm thiểu rủi ro lỗi so với việc thao tác chuỗi HTML thuần [9].

Razor là công nghệ chính được dùng để xây dựng các trang View trong hệ thống, bao gồm trang đăng ký lịch dịch vụ, trang quản trị, và các layout dùng chung.

CHƯƠNG 3 ĐÁNH GIÁ KẾT QUẢ

3.1 Mô tả bài toán

Hiện nay, nhu cầu chăm sóc thú cưng tại nhà đang ngày càng tăng cao, nhất là ở các khu đô thị lớn. Tuy nhiên, việc đặt lịch, theo dõi lịch sử chăm sóc và quản lý các dịch vụ còn thủ công và thiếu tính tiện lợi. Bài toán đặt ra là cần xây dựng một hệ thống website giúp:

- Người dùng có thể dễ dàng đặt lịch dịch vụ chăm sóc thú cưng tại nhà như tắm rửa, cắt móng, khám tổng quát,...

- Quản trị viên có thể quản lý danh sách dịch vụ, theo dõi các yêu cầu từ người dùng, xác nhận hoặc xóa lịch đặt.

- Giao diện website phải thân thiện, trực quan, hiển thị tốt trên nhiều loại thiết bị.

- Toàn bộ dữ liệu được lưu trữ và xử lý một cách hiệu quả trên nền tảng ASP.NET MVC kết hợp với SQL Server và Entity Framework Core.

3.2 Môi trường phát triển

Hệ thống được xây dựng và triển khai trong môi trường phát triển phần mềm phổ biến, bao gồm:

- Hệ điều hành: Windows 10/11

- IDE: Visual Studio 2022

- Ngôn ngữ lập trình: C# (ASP.NET Core MVC)

- Database: Microsoft SQL Server 2019

- ORM: Entity Framework Core

- Giao diện người dùng: Razor View, Bootstrap 5, jQuery

- Thư viện hỗ trợ: SignalR (cho xử lý bất đồng bộ real-time), Identity (quản lý người dùng)

- Trình duyệt kiểm thử: Google Chrome, Microsoft Edge

- Quản lý phiên bản: Git

Trong suốt quá trình phát triển, nhóm đã sử dụng Visual Studio làm công cụ chính để lập trình, debug, và kiểm tra hiệu năng của hệ thống. Các script SQL được viết thủ công và kiểm thử trực tiếp trên SQL Server Management Studio (SSMS).

3.3 Thiết kế

3.3.1 Xác định Actor

Trong hệ thống website đặt dịch vụ chăm sóc thú cưng tại nhà, các tác nhân (actor) tham gia tương tác với hệ thống bao gồm các vai trò người dùng khác nhau, mỗi actor đảm nhận một chức năng cụ thể, được mô tả như sau:

STT	Tên Actor	Mô tả
1	Khách	Là người dùng chưa đăng ký, có thể tìm kiếm và xem thông tin dịch vụ. Có thể đăng ký tài khoản để sử dụng các chức năng nâng cao.
2	Thành viên	Là người dùng đã đăng ký tài khoản. Có thể tìm kiếm dịch vụ, xem chi tiết, chỉnh sửa thông tin cá nhân, đánh giá, đặt lịch chăm sóc và quản lý các yêu cầu đặt lịch.
3	Quản trị viên	Có quyền cao nhất trong hệ thống. Có thể quản lý dịch vụ, tài khoản thành viên, xem thống kê và điều hành toàn bộ hoạt động của website

3.3.2 Sơ đồ Usecase tổng quát

Biểu đồ ca sử dụng (use case diagram) mô tả tập hợp các ca sử dụng, các tác nhân và những quan hệ giữa chúng. Các biểu đồ ca sử dụng mô tả cái nhìn tĩnh về hệ thống dưới con mắt của người sử dụng. Các biểu đồ ca sử dụng rất quan trọng để nắm bắt các chức năng của hệ thống

a) ***Ca sử dụng (usecase)***

Bước đầu tiên của phân tích yêu cầu là xác định các ca sử dụng của hệ thống. Một ca sử dụng là một tương tác giữ hệ thống và môi trường. Tập hợp các ca sử dụng là mô tả toàn bộ hệ thống cần xây dựng. Một ca sử dụng tương ứng với một chức năng

của hệ thống dưới góc nhìn của người sử dụng. Một ca sử dụng có thể lớn hoặc nhỏ. Một ca sử dụng chỉ ra làm thế nào một mục tiêu của người sử dụng được thoả mãn bởi hệ thống. Cần phân biệt các mục tiêu của người sử dụng và các tương tác của họ với hệ thống.

+ Mục tiêu: cái mà người sử dụng mong đợi

+ Tương tác: kỹ thuật cho phép đáp ứng mục tiêu.

Thực tế, chúng ta xác định các mục tiêu trước, sau đó chọn tập hợp các tương tác đáp ứng các mục tiêu đó.

b) *Tác nhân (Actor)*

Tác nhân đóng vai trò một người sử dụng hoặc một thực thể bên ngoaì tương tác với hệ thống. Cần phải phân biệt tác nhân (actor) và người sử dụng (user):

+ Nhiều người sử dụng có thể tương ứng một tác nhân

+ Một người sử dụng có thể tương ứng với nhiều tác nhân khác nhau

Tác nhân không nhất thiết luôn luôn là con người. Tác nhân có thể là môi trường, hệ thống khác, thực thể bên ngoài tương tác với hệ thống

c) *Đặc tả ca sử dụng*

Đặc tả điển hình của ca sử dụng:

- Ca sử dụng: tên ca sử dụng thường bắt đầu bởi động từ

- Các tác nhân: danh sách các tác nhân liên quan

- Mô tả: tóm tắt các xử lý cần thực hiện Đặt tả ca sử dụng có thể thêm:

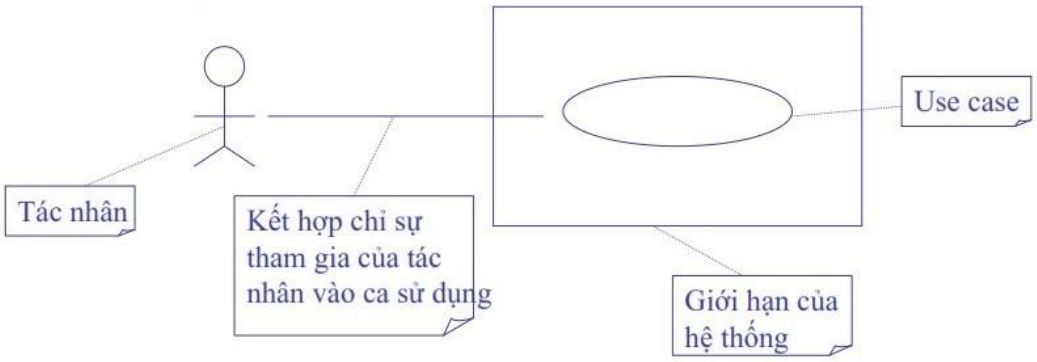
- Tham chiếu (reference) đến mục liên quan trong đặc tả yêu cầu.

- Điều kiện trước và điều kiện sau khi thực hiện ca sử dụng

Ngoài ra, đối với ca sử dụng ta có thể xây dựng một kịch bản (scenario) hành động mô tả các sự kiện sảy ra. Kịch bản gồm: gồm các sự kiện chính và các sự kiện ngoại lệ. Các sự kiện chia làm 2 luồng: Luồng tương ứng với các tác nhân và luồng tương ứng với hệ thống

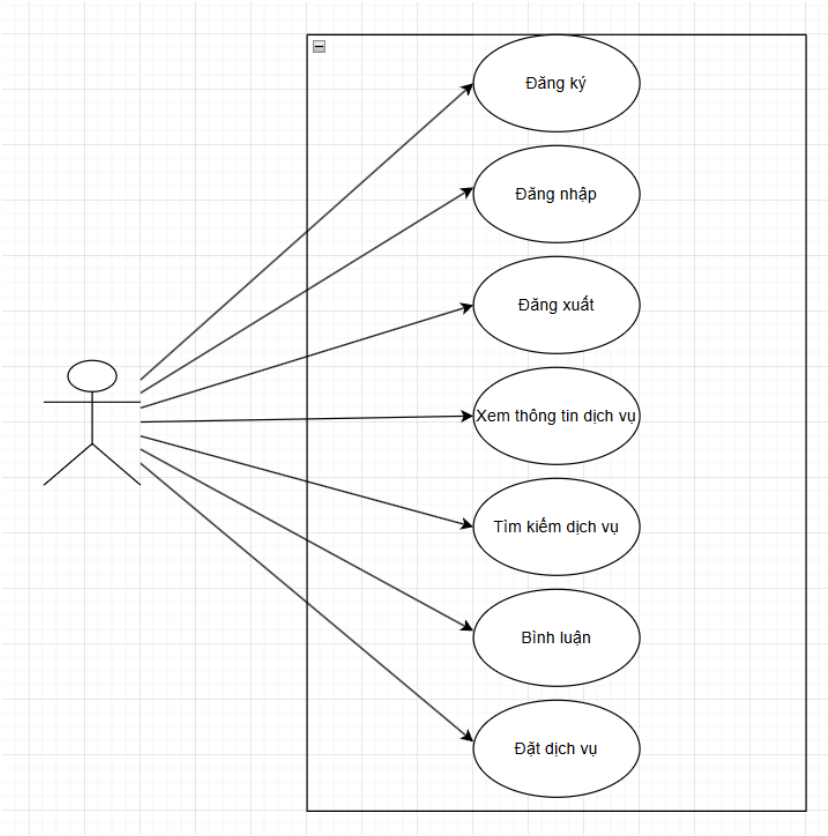
d) *Biểu đồ ca sử dụng*

Biểu đồ ca sử dụng mô tả quan hệ giữa các tác nhân và các ca sử dụng của một hệ thống. Ký hiệu như sau:



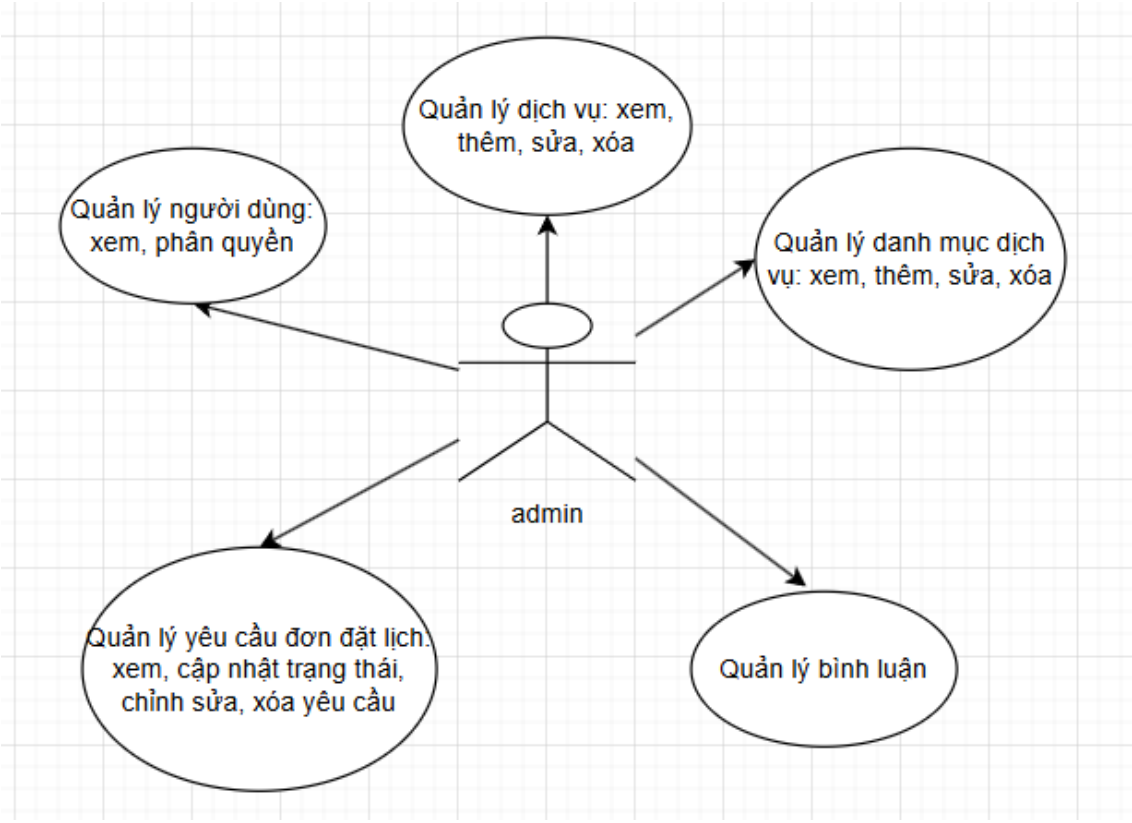
Hình 3.3.2-1 Các kí hiệu của biểu đồ ca sử dụng

3.3.3 Sơ đồ Usecase tổng quát khách hàng



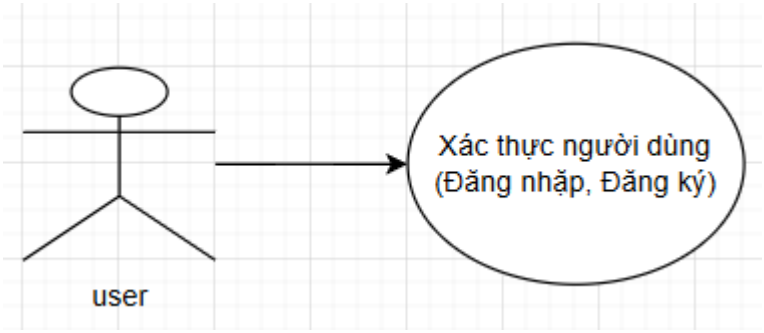
Hình 3.3.3-1 Sơ đồ usecase tổng quát khách hàng

3.3.4 Sơ đồ Usecase tổng quát admin



Hình 3.3.4-1 Sơ đồ usecase tổng quát admin

3.3.5 Sơ đồ Usecase xác thực người dùng



Hình 3.3.5-1 Sơ đồ usecase xác thực người dùng

- Mục đích: Cho phép người dùng đăng ký tài khoản mới hoặc đăng nhập để sử dụng hệ thống.

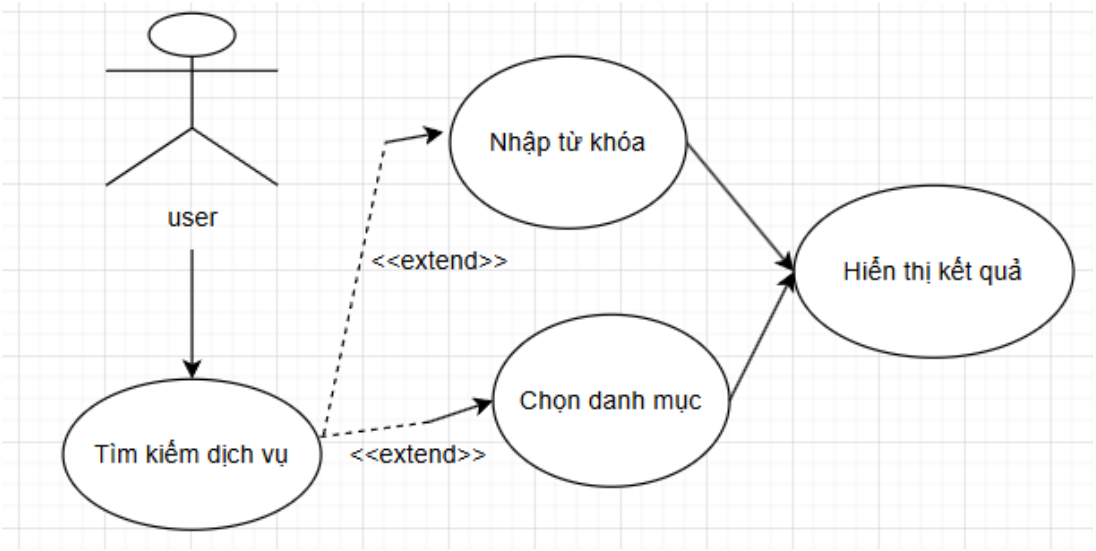
- Tác nhân: Khách hàng (user)

- Điều kiện:

+ Với đăng ký: người dùng chưa có tài khoản.

+ Với đăng nhập: người dùng đã có tài khoản hợp lệ (email + mật khẩu hợp lệ).

3.3.6 Sơ đồ Usecase chức năng Tìm kiếm dịch vụ



Hình 3.3.6-1 Sơ đồ usecase chức năng tìm kiếm dịch vụ

- Mục đích: Cho phép người dùng tra cứu các dịch vụ chăm sóc thú cưng theo từ khóa hoặc danh mục mong muốn.

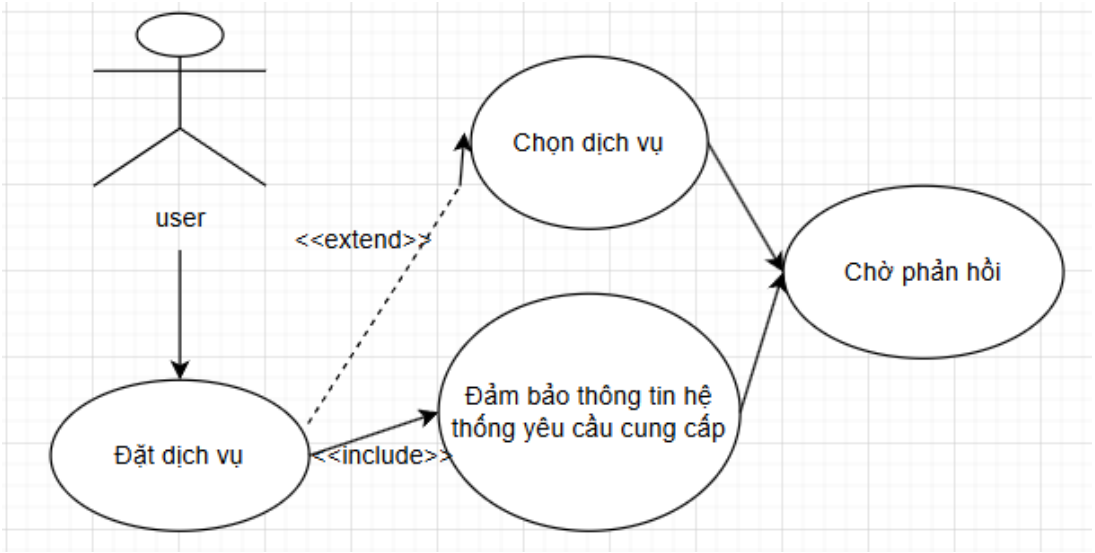
- Tác nhân: Khách hàng, Thành viên

- Điều kiện:

+ Người dùng đã truy cập website.

+ Dữ liệu dịch vụ đã được thêm vào hệ thống.

3.3.7 Sơ đồ Usecase chức năng đặt dịch vụ



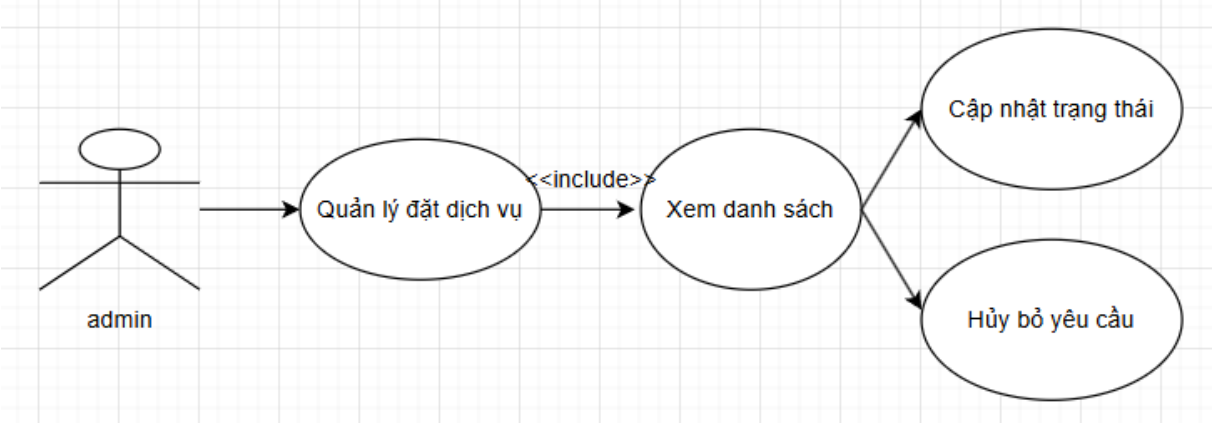
Hình 3.3.7-1 Sơ đồ usecase chức năng đặt dịch vụ

- Mục đích: Cho phép người dùng chọn dịch vụ và xác nhận đặt lịch chăm sóc thú cưng tại nhà.

- Tác nhân: Khách hàng (user)

- Điều kiện: Người dùng cung cấp đầy đủ thông tin để hệ thống yêu cầu.

3.3.8 Sơ đồ Usecase chức năng quản lý đặt dịch vụ (admin)



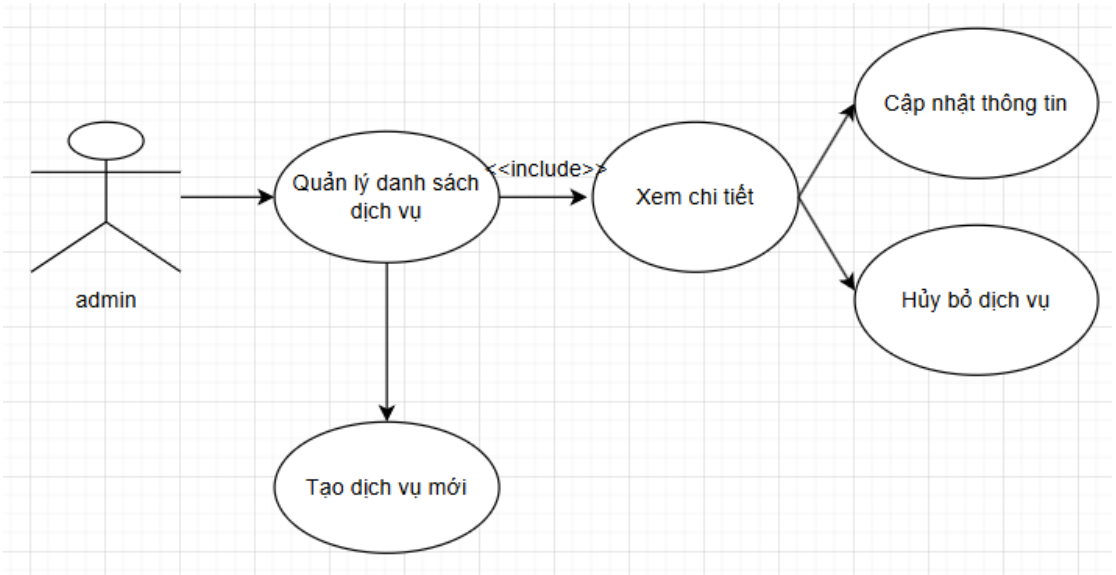
Hình 3.3.8-1 Sơ đồ usecase chức năng quản lý đặt dịch vụ

- Mục đích: Giúp quản trị viên xem, cập nhật trạng thái, hoặc hủy các đơn đặt lịch mà người dùng đã gửi.

- Tác nhân: Quản trị viên (Admin)

- Điều kiện: Quản trị viên đã đăng nhập vào hệ thống.

3.3.9 Sơ đồ Usecase chức năng quản lý dịch vụ (admin)



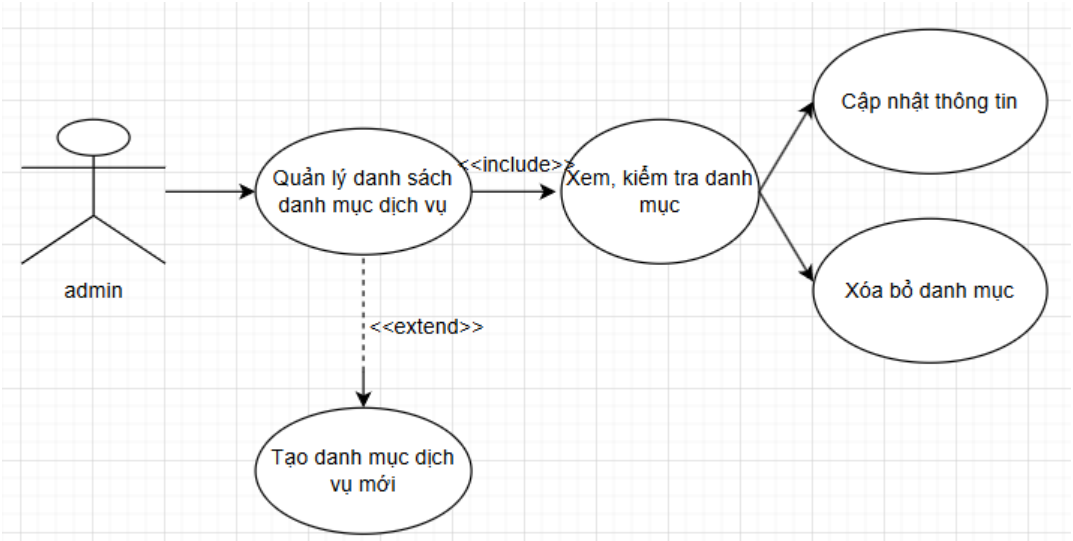
Hình 3.3.9-1 Sơ đồ usecase quản lý dịch vụ

- Mục đích: Giúp quản trị viên xem, thêm, cập nhật thông tin, hoặc hủy các dịch vụ website hỗ trợ.

- Tác nhân: Quản trị viên (Admin)

- Điều kiện: Quản trị viên đã đăng nhập vào hệ thống.

3.3.10 Sơ đồ Usecase chức năng quản lý danh mục dịch vụ (admin)



Hình 3.3.10-1 Sơ đồ usecase chức năng quản lý danh mục dịch vụ

- Mục đích: Giúp quản trị viên xem, kiểm tra, thêm, cập nhật thông tin, hoặc xóa các danh mục dịch vụ website hỗ trợ, tối ưu hóa nhu cầu tìm kiếm.

- Tác nhân: Quản trị viên (Admin)

- Điều kiện: Quản trị viên đã đăng nhập vào hệ thống.

3.3.11 Thiết kế cơ sở dữ liệu

Danh sách các bảng dữ liệu:

services	Thông tin về các dịch vụ cung cấp (tên, mô tả, hình ảnh...)
calls	Lịch đặt dịch vụ của khách hàng (tên, email, ngày giờ, dịch vụ, trạng thái, ghi chú)
categories	Danh mục các dịch vụ cho từng loài
comments	Bình luận/đánh giá từ người dùng về dịch vụ

idennity	Tài khoản người dùng (username, email, password hash...)
roleclaims	Quyền cụ thể gắn với từng vai trò (claim cho role)
roles	Vai trò của người dùng (Admin, User...)
userclaims	Quyền riêng gắn với từng người dùng
userlogins	Lịch sử đăng nhập, provider, thông tin xác thực ngoài
userroles	Liên kết người dùng với vai trò
usertokens	Token xác thực (dùng reset mật khẩu, xác minh...)

Mô tả chi tiết các bảng dữ liệu:

+ Bảng services:

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
Id	int	Khóa chính, tự động tăng	Mã dịch vụ
Name	nvarchar(50)	Not null	Tên dịch vụ
PicturePath	nvarchar(100)	Có thể null	Đường dẫn hình ảnh
Description	nvarchar(500)	Có thể null	Mô tả chi tiết dịch vụ
CategoryId	int	Khóa ngoại	Liên kết đến bảng danh mục dịch vụ

+ Bảng calls:

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
Id	int	Khóa chính, tự động tăng	Mã cuộc gọi

Name	nvarchar	Not null	Tên người gửi yêu cầu
Email	nvarchar	Not null	Email người gửi yêu cầu
Problem	nvarchar	Not null	Nội dung yêu cầu, vấn đề, ghi chú
CallTime	datetime2	Not null	Thời gian gửi yêu cầu
Answered	bit	Not null	Trạng thái phản hồi (0/1)
AnswerTime	datetime2	Có thể null	Thời gian phản hồi (nếu có)

+ Bảng categories:

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
Id	int	Khóa chính, tự động tăng	Mã danh mục dịch vụ
Name	nvarchar	Not null	Tên danh mục (ví dụ: Mammal, Birds, ...)

+ Bảng comments:

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
Id	int	Khóa chính, tự động tăng	Mã bình luận
Content	nvarchar	Not null	Nội dung bình luận
AnimalId	int	Khóa ngoại	Liên kết đến bảng services (Id)

+ Bảng idennity:

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	uniqueidentifier	Khóa chính	ID người dùng
UserName	nvarchar	Không null	Tên người dùng
NormalizedUserName	nvarchar		Tên chuẩn hóa
Email	nvarchar		Email người dùng
NormalizedEmail	nvarchar		Email chuẩn hóa
EmailConfirmed	bit		Email đã xác nhận chưa (0/1)
PasswordHash	nvarchar		Mật khẩu đã mã hóa
SecurityStamp	nvarchar		Token bảo mật
ConcurrencyStamp	nvarchar		Token kiểm tra đồng bộ
PhoneNumber	nvarchar		Số điện thoại
PhoneNumberConfirmed	bit		Số điện thoại đã xác nhận chưa (0/1)
TwoFactorEnabled	bit		Có bật xác thực 2 bước không
LockoutEnd	datetimeoffset		Thời điểm bị khóa tài khoản
LockoutEnabled	bit		Có cho phép khóa tài khoản không

AccessFailedCount	int		Số lần đăng nhập sai
Discriminator	nvarchar		Phân biệt kiểu user (nếu dùng kế thừa)

+ Bảng roleclaims:

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
Id	int	Khóa chính	Mã định danh cho mỗi claim
RoleId	nvarchar	Có thể null	Id vai trò được gán quyền
ClaimType	nvarchar	Có thể null	Loại quyền (claim)
ClaimValue	nvarchar	Có thể null	Giá trị của quyền

+ Bảng roles:

Tên trường	**Kiểu dữ liệu**	**Ràng buộc**	**Mô tả**
Id	nvarchar(900)	Khóa chính	Mã định danh của vai trò
Name	nvarchar	Có thể null	Tên vai trò
Normalized Name	nvarchar	Có thể null	Tên vai trò chuẩn hóa
ConcurrencyStamp	nvarchar	Có thể null	Dùng để kiểm tra đồng bộ dữ liệu

+ Bảng userclaims:

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	int	Tự động tăng	ID định danh cho mỗi claim
ClaimType	nvarchar	Có thể null	Loại quyền yêu cầu
ClaimValue	nvarchar	Có thể null	Giá trị của quyền yêu cầu
UserId	nvarchar(900)	Khóa chính	Mã định danh người dùng

+ Bảng userlogins:

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
UserId	nvarchar(900)	Khóa chính	Mã người dùng
LoginProvider	nvarchar	Có thể null	Nhà cung cấp đăng nhập (Google, Facebook...)
ProviderKey	nvarchar	Có thể null	Khóa định danh của nhà cung cấp
ProviderDisplayName	nvarchar	Có thể null	Tên hiển thị của nhà cung cấp
CallTime			
Answered			
AnswerTime			

+ Bảng userroles:

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
UserId	nvarchar(900)	Khóa chính	Mã người dùng
RoleId	nvarchar(900)	Khóa chính	Mã vai trò (quyền) được gán

+ Bảng usertokens:

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
UserId	nvarchar(900)	Khóa chính	Mã người dùng
LoginProvider	nvarchar	Khóa chính	Nhà cung cấp đăng nhập (Google, Facebook...)
Name	nvarchar	Khóa chính	Tên token
Value	nvarchar		Giá trị token

3.3.12 Cài đặt và chạy ứng dụng

a) ***Công cụ cần thiết***

Để triển khai và chạy ứng dụng, cần cài đặt các phần mềm sau:

- Visual Studio 2022 (hoặc phiên bản mới hơn), cài kèm workload ASP.NET and web development

- .NET SDK phiên bản X.X.X (trùng với version trong project)

- SQL Server (có thể dùng bản Express) và SQL Server Management Studio (SSMS) để quản lý database

- Postman (nếu muốn test API)

b) ***Các bước cài đặt và chạy project***

- Bước 1: Tải mã nguồn

+ Mở Visual Studio → File → Open → chọn file .sln của project

(Git: git clone https://github.com/ThanhPhong2611/ASPNET-DK23TTC11-hothanhphong-petcare.git)

- Bước 2: Cấu hình chuỗi kết nối

+ Mở file appsettings.json

+ Chỉnh sửa chuỗi kết nối tại trường "DefaultConnection" sao cho đúng với SQL Server trên máy: "ConnectionStrings": {

"DefaultConnection":
"Server=localhost;Database=TenCSDL;Trusted_Connection=True;"

}

- Bước 3: Khởi tạo cơ sở dữ liệu

+ Mở Package Manager Console trong Visual Studio (Tools > NuGet Package Manager > PMC)

+ Chạy các lệnh sau: Update-Database

(Nếu chưa có database, Entity Framework sẽ tự tạo theo các migration có sẵn.)

- Bước 4: Chạy ứng dụng

+ Nhấn F5 hoặc bấm nút Start để khởi chạy

+ Ứng dụng sẽ mở trình duyệt tại địa chỉ như: https://localhost:5001

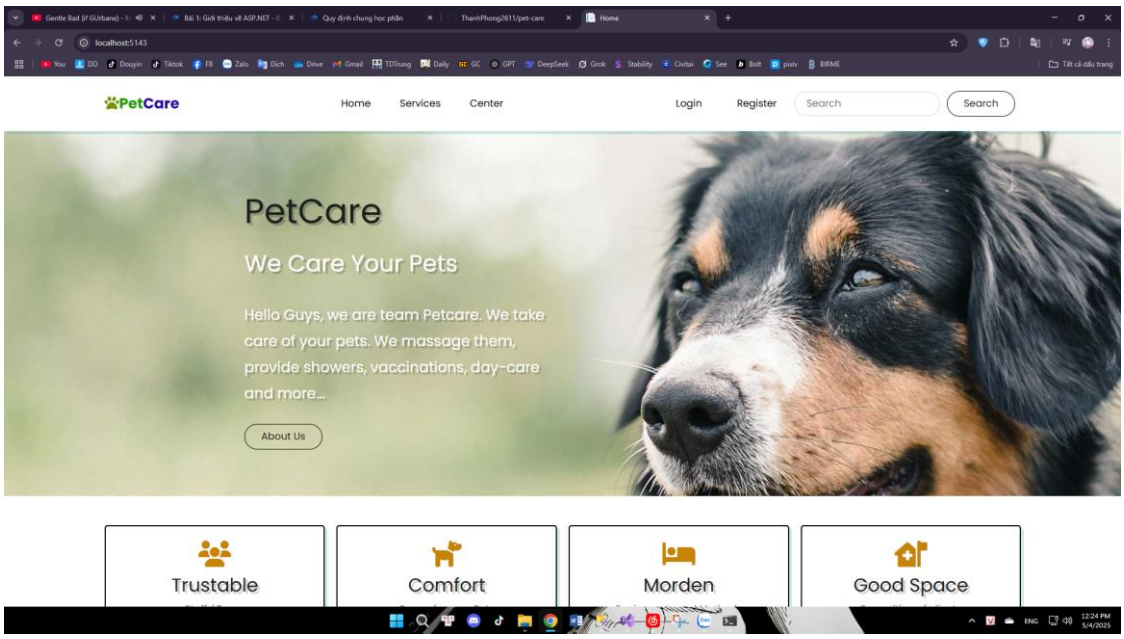
* Admin Account: Admin, 123456aA

CHƯƠNG 4 KẾT LUẬN

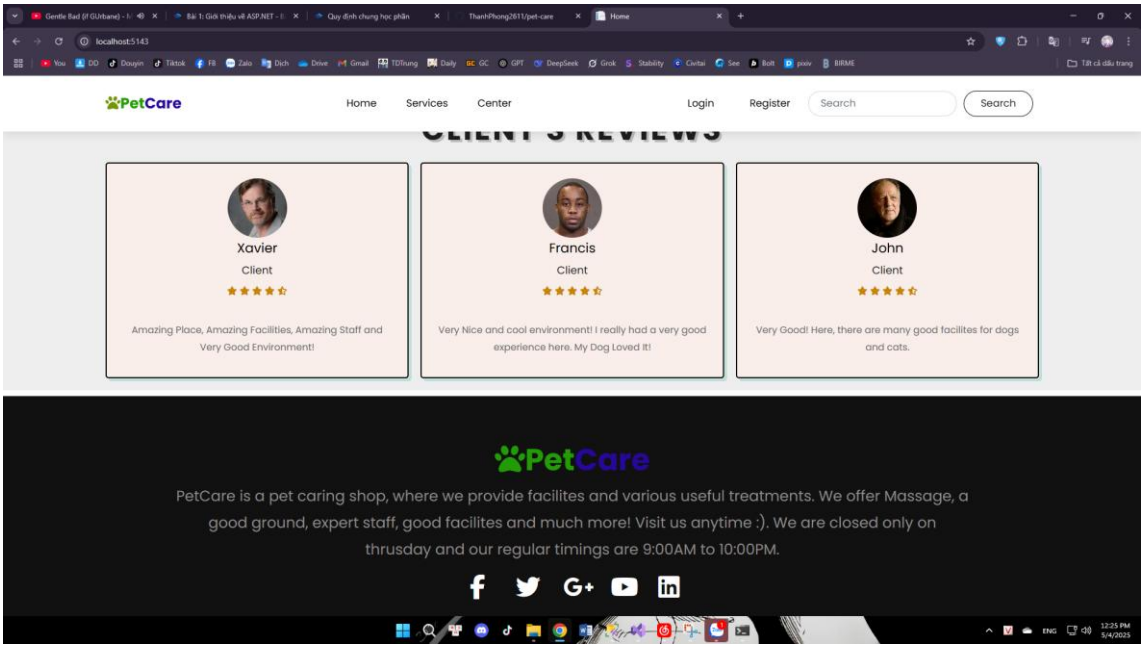
4.1 Giao diện website và chức năng phía khách hàng

4.1.1 Trang chủ website

Giao diện trang chủ của website:



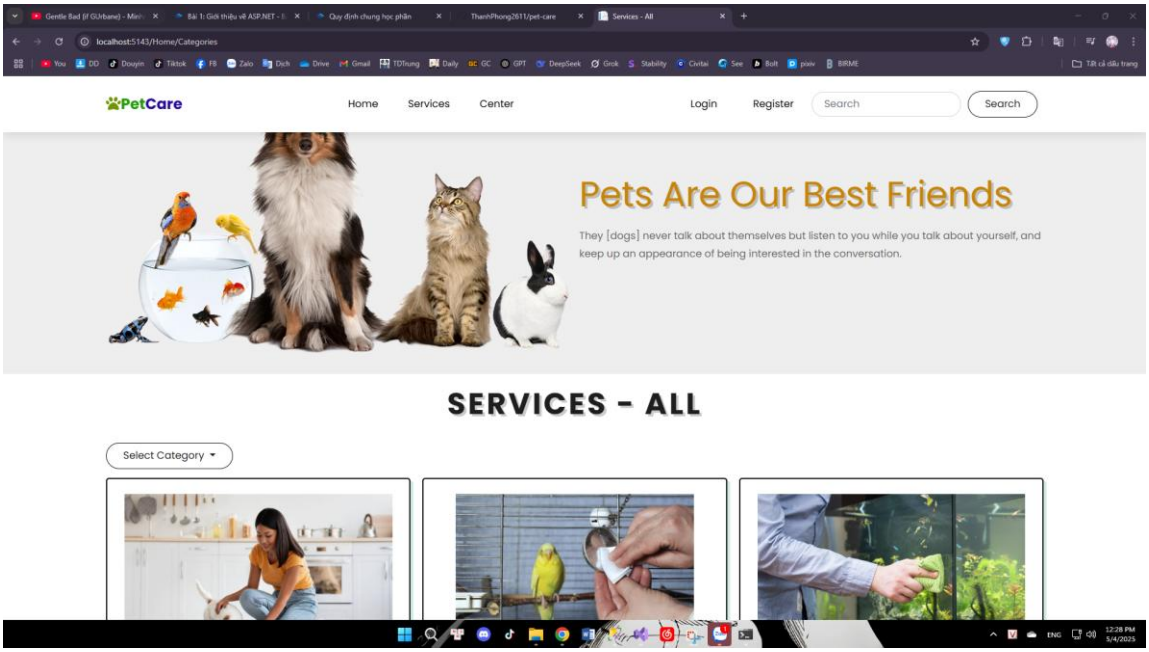
Hình 4.1.1-1 Giao diện trang chủ Website



Hình 4.1.1-2 Giao diện trang chủ Website

4.1.2 Giao diện trang dịch vụ

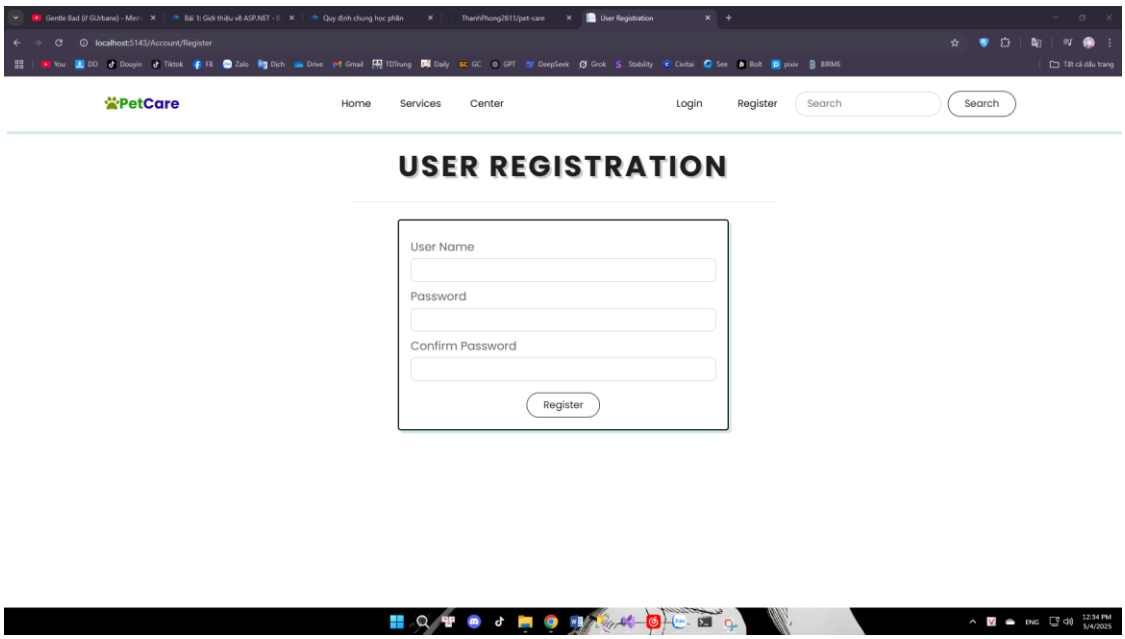
Giao diện trang dịch vụ của website:



Hình 4.1.2-1 Giao diện trang dịch vụ

4.1.3 Giao diện trang đăng ký tài khoản

Giao diện cho phép người dùng nhập User Name, Password và xác nhận mật khẩu.Thông tin được kiểm tra hợp lệ phía client trước khi gửi lên server. Sau khi đăng ký thành công, tài khoản sẽ được lưu trong hệ thống với vai trò mặc định là User.

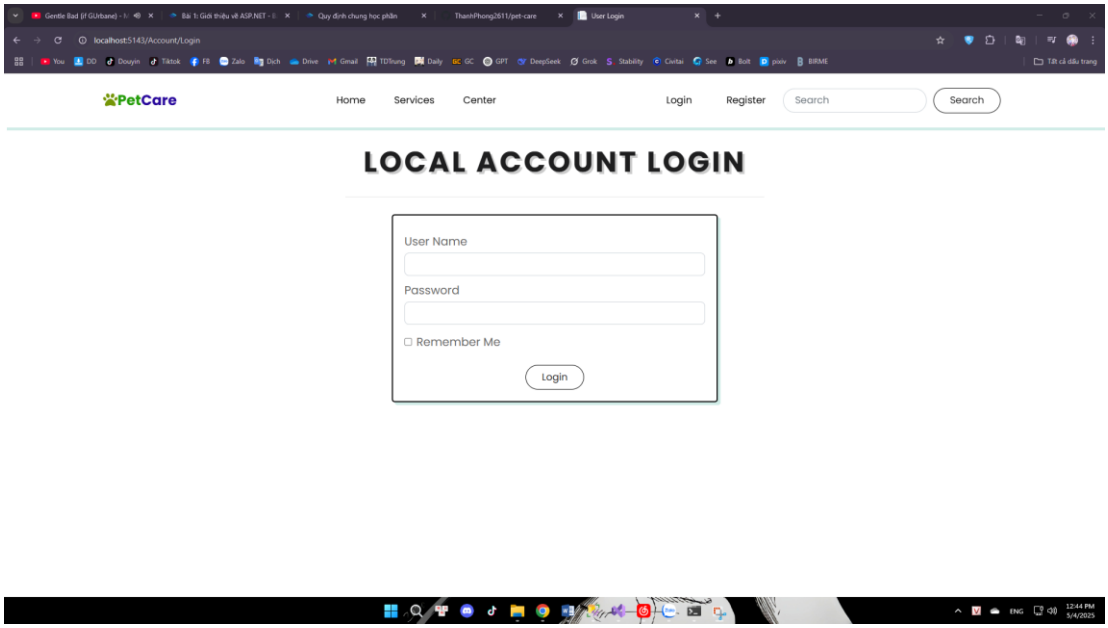


Hình 4.1.3-1 Giao diện trang đăng ký tài khoản

4.1.4 Giao diện trang đăng nhập

Trang đăng nhập cho phép người dùng nhập email và mật khẩu để truy cập hệ thống.

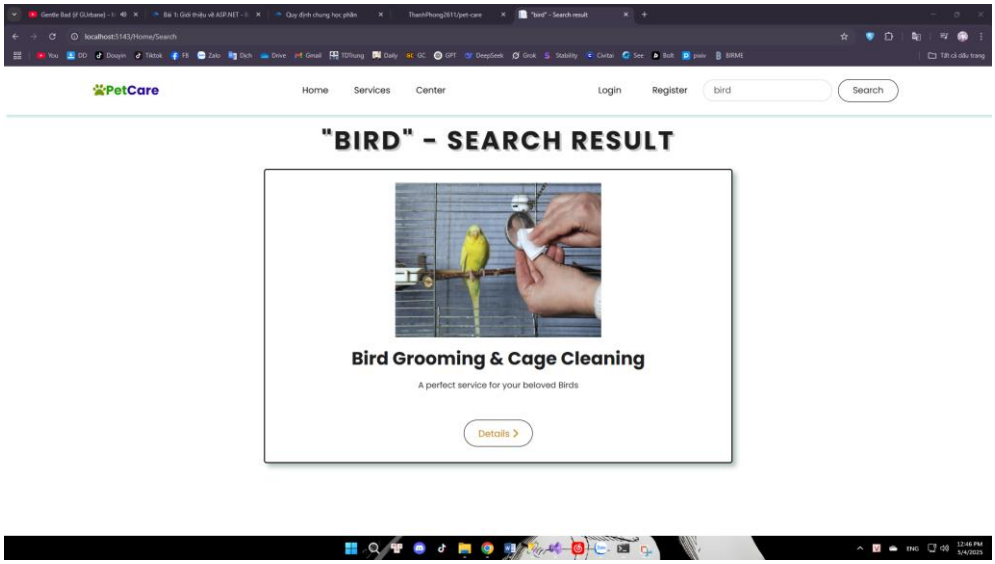
Nếu thông tin hợp lệ, hệ thống sẽ điều hướng đến giao diện phù hợp với vai trò người dùng.



Hình 4.1.4-1 Giao diện trang đăng nhập

4.1.5 Giao diện trang tìm kiếm

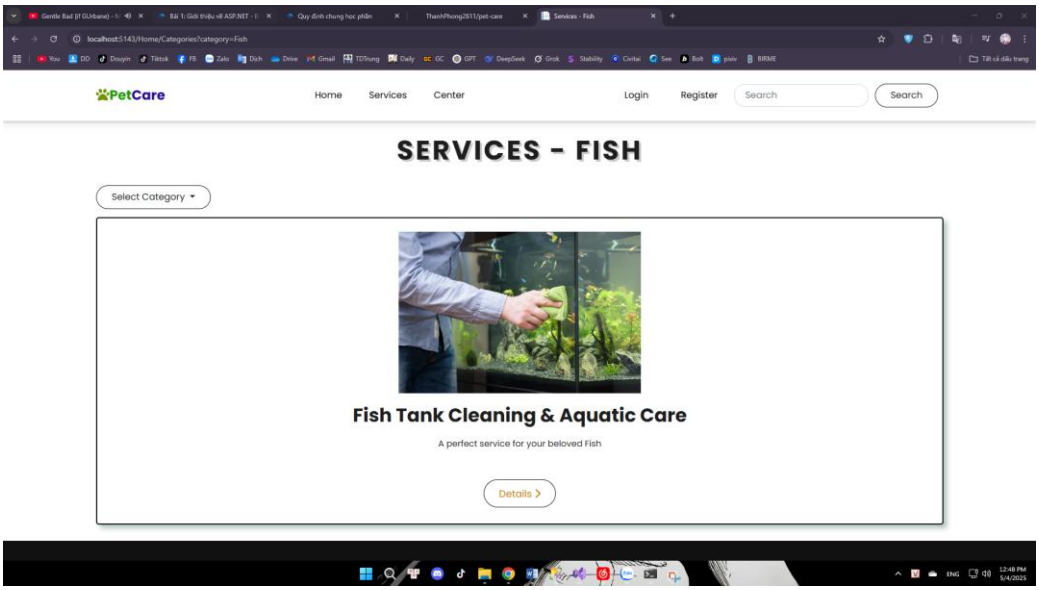
Người dùng nhập từ khóa liên quan đến dịch vụ muốn tìm, hệ thống sẽ lọc và hiển thị các kết quả phù hợp.



Hình 4.1.5-1 Giao diện trang tìm kiếm

4.1.6 Giao diện trang tìm kiếm theo danh mục

Người dùng chọn danh mục dịch vụ có sẵn, hệ thống sẽ hiển thị tất cả các dịch vụ thuộc danh mục đó.



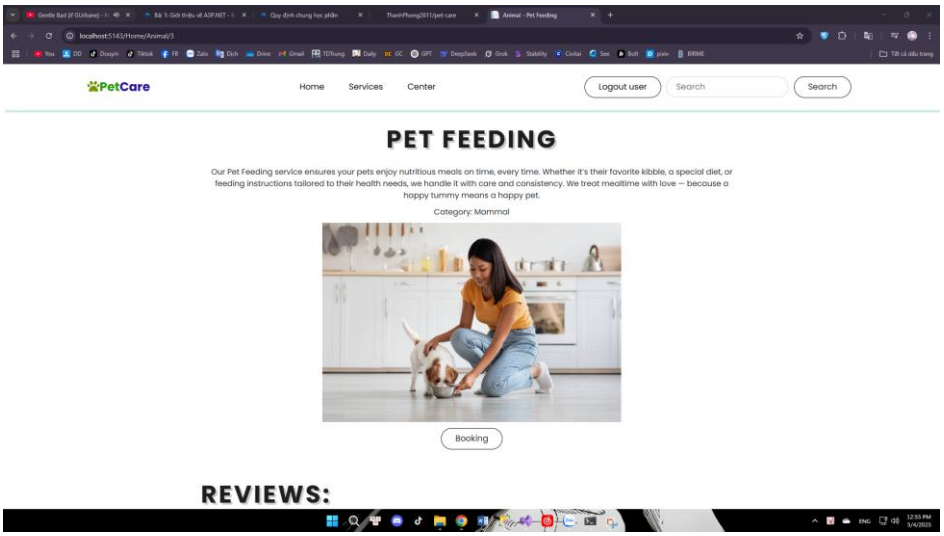
Hình 4.1.6-1 Giao diện trang tìm kiếm theo danh mục

4.1.7 Giao diện trang chi tiết dịch vụ

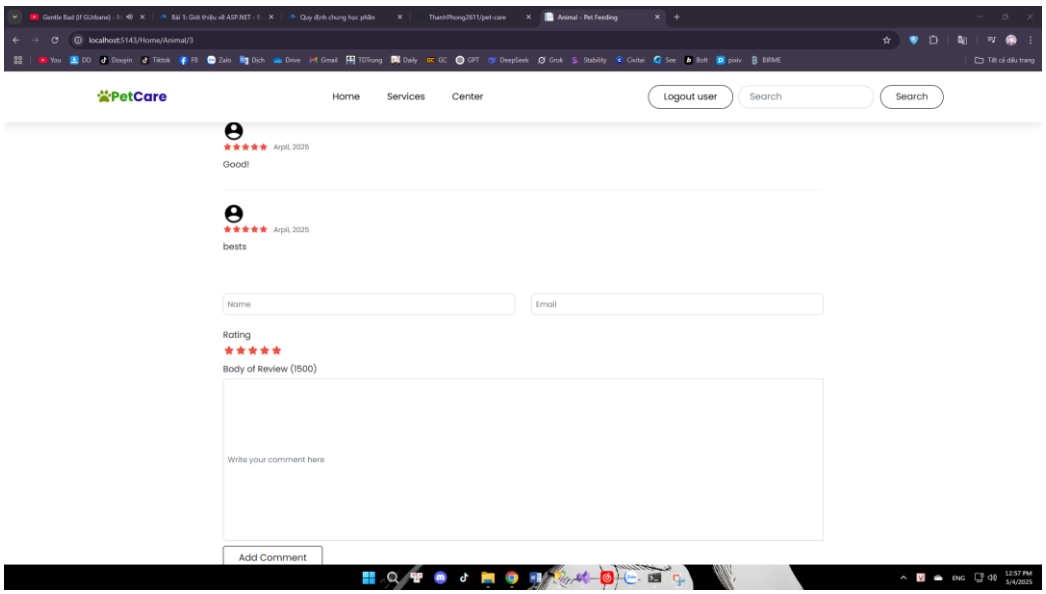
Trang hiển thị thông tin chi tiết về dịch vụ như tên, mô tả, hình ảnh và danh mục. Người dùng có thể:

+ Đặt lịch sử dụng dịch vụ thông qua nút đặt lịch (Booking).

+ Xem và gửi đánh giá (Reviews) bằng cách nhập bình luận trực tiếp bên dưới.



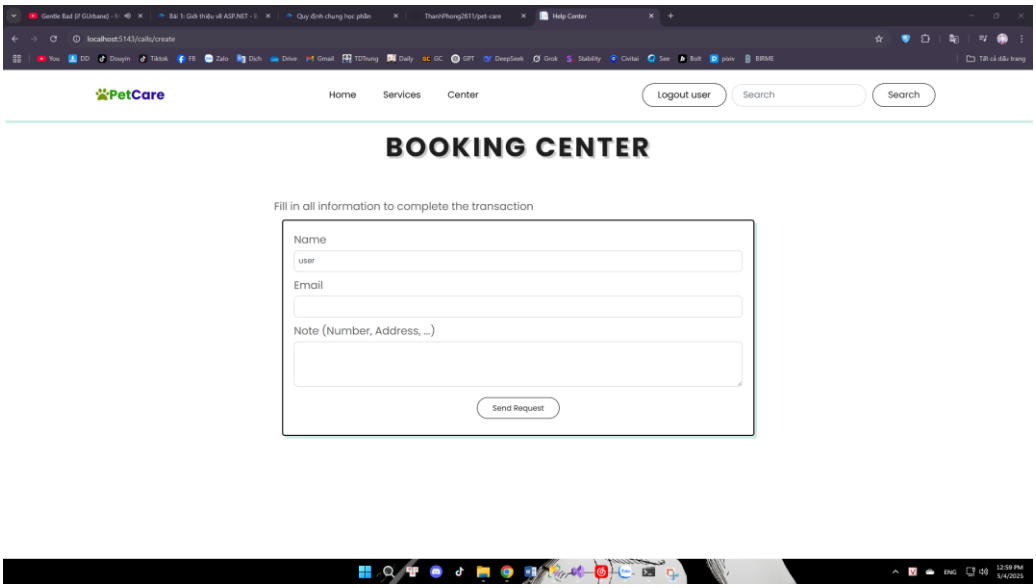
Hình 4.1.7-1 Giao diện trang chi tiết dịch vụ



Hình 4.1.7-2 Giao diện trang chi tiết dịch vụ

4.1.8 Giao diện trang xác thực đặt lịch

Ở trang này, người dùng cần cung cấp thông tin đầy đủ trước khi gửi yêu cầu đặt lịch, yêu cầu sẽ được gửi đi cùng thời gian thực.



Hình 4.1.8-1 Giao diện trang xác nhận đặt lịch

4.2 Giao diện website và chức năng của quản trị viên

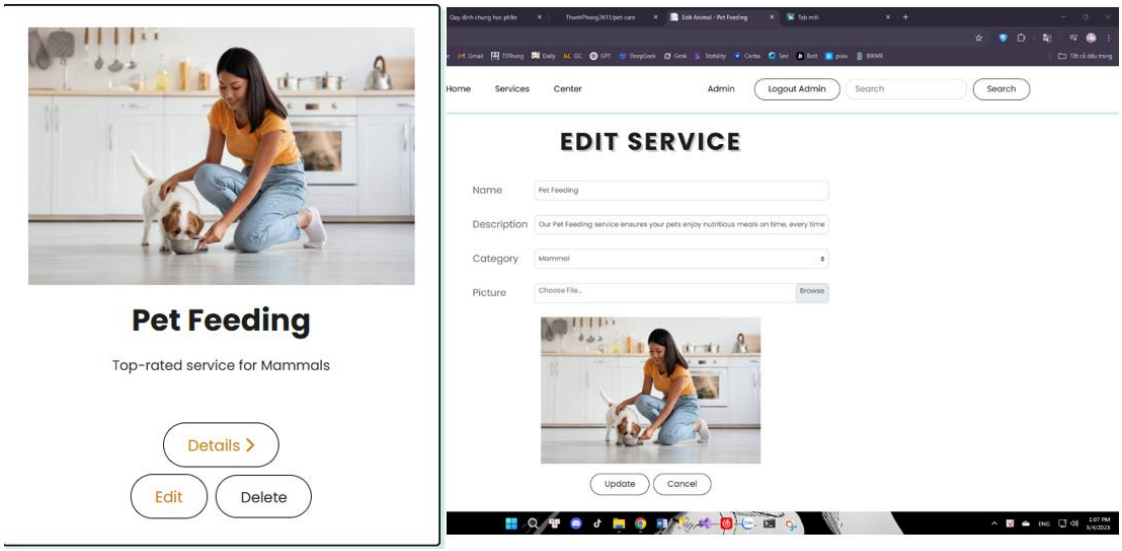
4.2.1 Giao diện và chức năng quản lý danh sách dịch vụ

Cho phép quản trị viên thực hiện các thao tác:

+ Xem, truy cập danh sách dịch vụ.

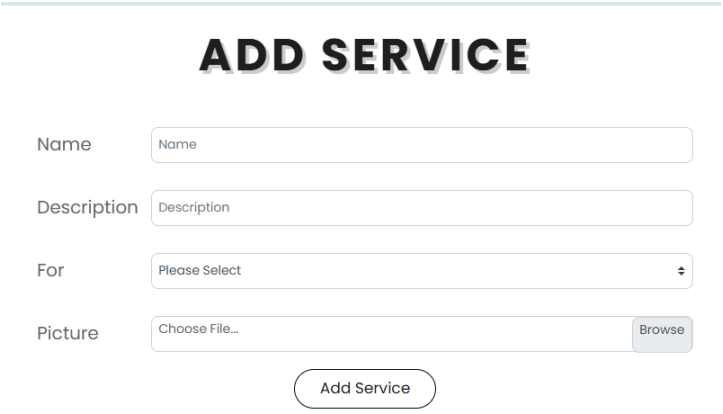
+ Chỉnh sửa thông tin dịch vụ (tên, mô tả, danh mục, ảnh).

+ Xóa dịch vụ khỏi hệ thống nếu không còn phù hợp.



Hình 4.2.1-1 Ảnh ghép - Giao diện, chức năng - Quản lý dịch vụ

Giao diện Form thêm dịch vụ:



Hình 4.2.1-2 Giao diện, chức năng - quản lý dịch vụ

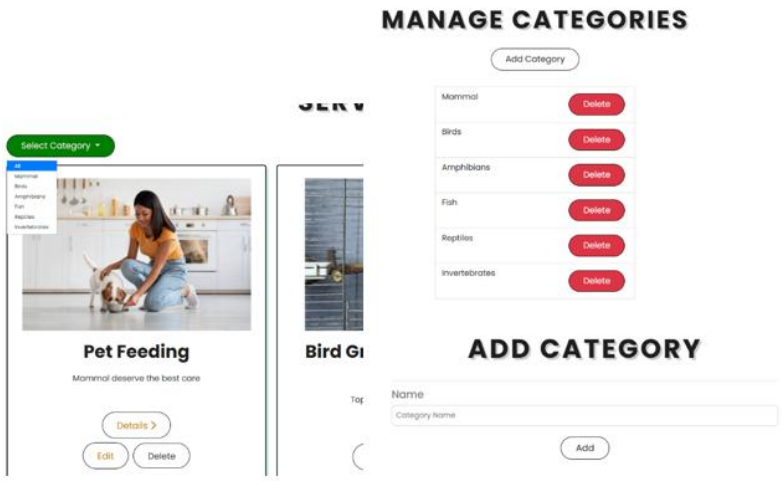
4.2.2 Giao diện và chức năng quản lý danh sách danh mục

Quản trị viên có thể thực hiện các thao tác:

+ Xem danh sách các danh mục hiện có.

+ Tạo mới danh mục để phân loại dịch vụ rõ ràng hơn.

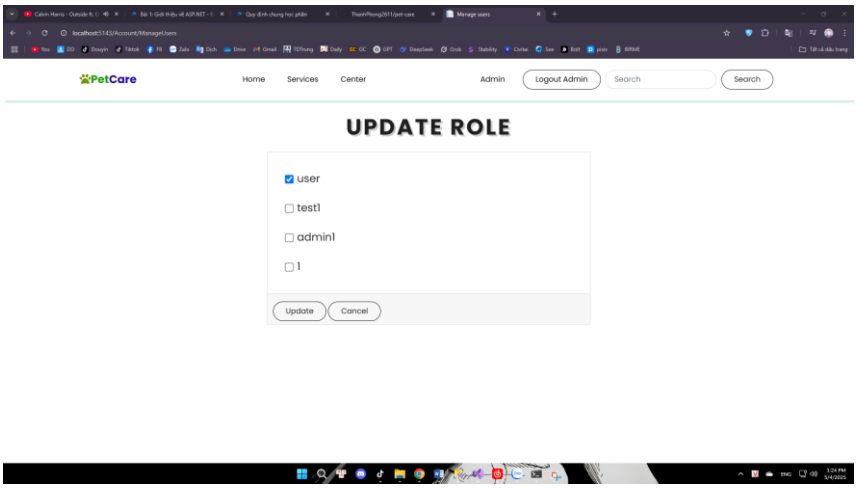
+ Xóa danh mục nếu không còn sử dụng (sau khi đảm bảo không liên kết với dịch vụ nào).



Hình 4.2.2-1 Ảnh ghép - Danh sách danh mục, xóa, thêm mới

4.2.3 Giao diện và chức năng quản lý phân quyền

Chức năng này cho phép quản trị viên cập nhật phân quyền sử dụng hệ thống quản trị website.



Hình 4.2.3-1 Quản lý phân quyền

4.2.4 Giao diện và chức năng quản lý đặt dịch vụ

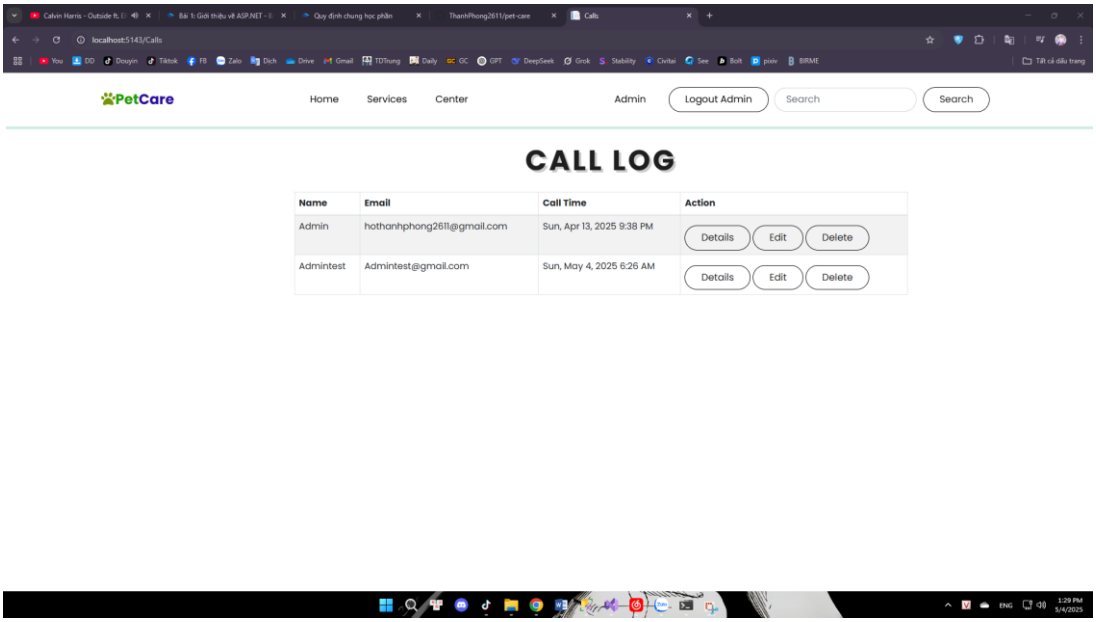
Tính năng này cho phép quản trị viên giám sát và điều phối các yêu cầu đặt dịch vụ từ người dùng. Bao gồm:

+ Xem danh sách các đơn đặt dịch vụ từ khách hàng, hiển thị thông tin như tên khách, thời gian đặt, trạng thái xử lý…

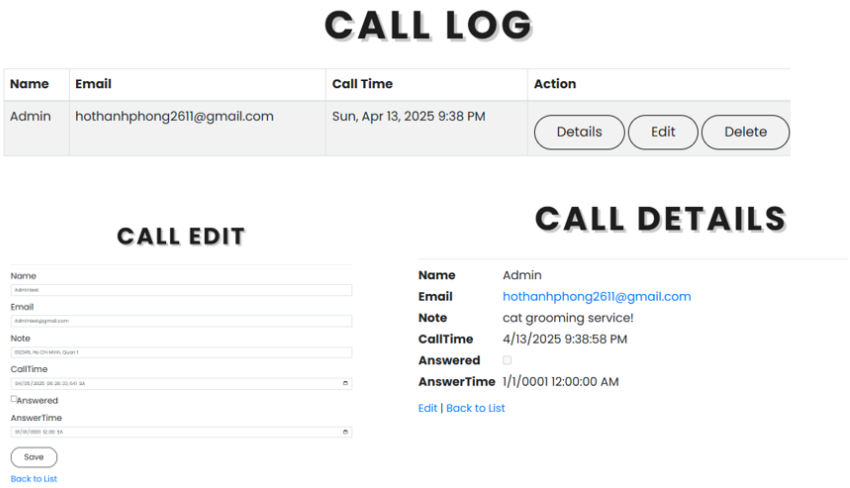
+ Xem chi tiết từng đơn đặt lịch để kiểm tra thông tin liên hệ, mô tả yêu cầu hoặc phản hồi từ khách hàng.

+ Cập nhật trạng thái đơn đặt (đang chờ, đã xác nhận, hoàn tất, từ chối…).

+ Xóa đơn đặt lịch nếu phát hiện yêu cầu không hợp lệ hoặc bị hủy.



Hình 4.2.4-1 Giao diện trang quản lý đặt dịch vụ



Hình 4.2.4-2 Ảnh ghép - Chức năng - Quản lý đặt dịch vụ

CHƯƠNG 5 HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Đồ án đã hoàn thành hệ thống website đặt dịch vụ chăm sóc thú cưng tại nhà với các chức năng cơ bản như: đăng ký/đăng nhập, tìm kiếm dịch vụ, đặt lịch, đánh giá, và chức năng quản trị dành cho admin (quản lý dịch vụ, danh mục, đơn đặt, phân quyền).

Hệ thống được xây dựng bằng ASP.NET Core, giao diện thân thiện, dễ sử dụng, đáp ứng tốt nhu cầu người dùng và hỗ trợ thao tác quản lý hiệu quả.

5.2 Hạn chế

- Chưa tích hợp tính năng thanh toán trực tuyến, người dùng chỉ đặt lịch mà chưa thể thanh toán qua hệ thống.

- Chưa có hệ thống thông báo (notification) khi đơn đặt dịch vụ được xử lý.

- Chưa hỗ trợ responsive tốt trên mọi thiết bị di động.

- Chưa tích hợp được hệ thống định vị, địa chỉ khách hàng để thuận tiện cho dịch vụ chăm sóc thú cưng tại nhà.

- Giao diện và trải nghiệm người dùng còn đơn giản, chưa có sự tối ưu về mặt thẩm mỹ và tiện ích.

- Tính năng phân quyền quản trị còn cơ bản, chưa hỗ trợ phân nhóm chi tiết.

5.3 Hướng phát triển

- Tích hợp cổng thanh toán online (Momo, ZaloPay, VNPAY...).

- Xây dựng hệ thống thông báo qua email hoặc trong ứng dụng.

- Thiết kế lại giao diện theo hướng hiện đại, thân thiện hơn với người dùng.

- Mở rộng chức năng quản lý khách hàng, lịch sử đặt dịch vụ chi tiết hơn.

- Tối ưu hệ thống trên mobile, cải thiện hiệu suất và tốc độ tải trang.

- Nâng cấp phần quản trị với dashboard thống kê, biểu đồ và phân quyền nâng cao.

DANH MỤC TÀI LIỆU THAM KHẢO

Tài liệu tham khảo:

[1] Microsoft Docs – ASP.NET Overview, https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-8.0, ngày truy cập: 04/05/2025.

[2] W3Schools – ASP.NET Tutorial, https://www.w3schools.com/asp/, ngày truy cập: 04/05/2025.

[3] Hostinger – ASP.NET vs PHP Hosting Comparison, https://www.hostinger.com/tutorials/php-vs-asp-net, ngày truy cập: 04/05/2025.

[4] Microsoft SQL Server, https://www.microsoft.com/en-us/sql-server, ngày truy cập: 04/05/2025.

[5] Microsoft Entity Framework Core, https://learn.microsoft.com/en-us/ef/core/, ngày truy cập: 04/05/2025.

[6] ASP.NET SignalR, https://learn.microsoft.com/en-us/aspnet/core/signalr/introduction, ngày truy cập: 04/05/2025.

[7] Bootstrap Framework, https://getbootstrap.com/, ngày truy cập: 04/05/2025.

[8] jQuery, https://jquery.com/, ngày truy cập: 04/05/2025.

[9] Razor Pages – Microsoft Docs, https://learn.microsoft.com/en-us/aspnet/core/mvc/views/razor, ngày truy cập: 04/05/2025.

PHỤ LỤC