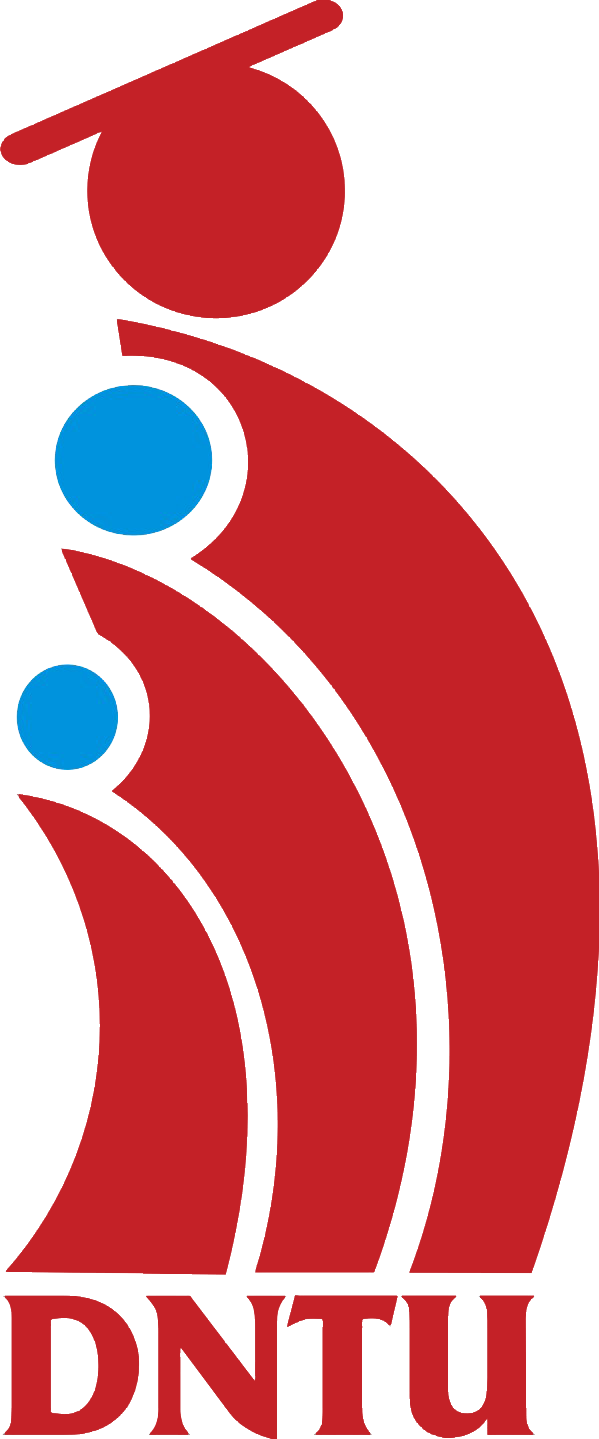
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐỒNG NAI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KIỂM TRA CUỐI KỲ**

**TIỂU LUẬN**

**TÊN ĐỀ TÀI:** PHẦN MỀM QUẢN LÝ ĐIỀU HÀNH CÔNG VIỆC

Tên học phần: Lập trình web

Mã học phần: 869.65422

Số tín chỉ: 3

Học kỳ: I , Năm học: 2024 – 2025.

Chuyên ngành: Công Nghệ Thông Tin

Khóa: 18 (2022 - 2026)

**Nhóm thực hiện:**

1. Nguyễn Đình Lực - 1822041377 – 22DTH6

2. Phạm Khánh Duy - 1822041255 – 22DTH6

**Giảng viên: Huỳnh Hoàng Tân**

**ĐỒNG NAI – 2024**

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

***Đồng Nai****,**ngày ….… tháng ….… năm …*

**Giảng viên hướng dẫn**

Mục Lục

[**I. Giới thiệu** 4](#_Toc181564298)

[**II. Cấu trúc phần mềm** 4](#_Toc181564299)

[1.Controller 4](#_Toc181564300)

[a.HomeController 4](#_Toc181564301)

[b.TaiKhoanController 4](#_Toc181564302)

[c.VanBanDenController 4](#_Toc181564303)

[d.VanBanDiController 4](#_Toc181564304)

[e.ThongKeController 4](#_Toc181564305)

[2.Models 4](#_Toc181564306)

[a.TaiKhoan 4](#_Toc181564307)

[b.VanBanDen 4](#_Toc181564308)

[c.VanBanDi 4](#_Toc181564309)

[d.ThongKe 4](#_Toc181564310)

[e.ErrorViewModel 4](#_Toc181564311)

[3.Views 4](#_Toc181564312)

[a.Home 4](#_Toc181564313)

[b.TaiKhoan 4](#_Toc181564314)

[c.VanBanDen 4](#_Toc181564315)

[d.VanBanDi 4](#_Toc181564316)

[e.ThongKe 4](#_Toc181564317)

[f.Shared 4](#_Toc181564318)

[4.Program 4](#_Toc181564319)

[**III. Phần mềm quản lý điều hành công việc.** 4](#_Toc181564320)

[**1.Đăng ký và đăng nhập tài khoản.** 4](#_Toc181564321)

[a.Đăng ký tài khoản: 4](#_Toc181564322)

[b.Đăng nhập: 4](#_Toc181564323)

[**2. Giao diện phần mềm:** 4](#_Toc181564324)

[**3.Các chức năng:** 4](#_Toc181564325)

[a.Văn bản đến: 4](#_Toc181564326)

[b.Văn bản đi: 4](#_Toc181564327)

[c.Thống kê văn bản: 4](#_Toc181564328)

[d.Quản lý tài khoản: 4](#_Toc181564329)

[Thao tác: 4](#_Toc181564330)

[e.Chuyển Hướng 4](#_Toc181564331)

[f.Đăng xuất: 4](#_Toc181564332)

[IV.Tài liệu tham khảo 5](#_Toc181564333)

# **I. Giới thiệu**

Hệ thống phần mềm quản lý văn bản đến và văn bản đi giúp tổ chức quản lý thông tin các văn bản hành chính một cách hiệu quả và chính xác. Với các chức năng quản lý và thống kê dễ dàng, hệ thống đảm bảo sự tiện lợi trong việc lưu trữ và tra cứu văn bản.

Phần mềm Điều Hành Quản Lý Công Việc là giải pháp giúp quản lý công việc một cách hiệu quả và tiện lợi. Phần mềm được thiết kế để hỗ trợ người dùng theo dõi và quản lý các văn bản cũng như các công việc hàng ngày một cách dễ dàng, từ đó giúp nâng cao năng suất và sự chuyên nghiệp trong quy trình làm việc.

**Mục tiêu của phần mềm:**

* **Quản lý văn bản**: Cung cấp các công cụ để lưu trữ, tìm kiếm, và thống kê các văn bản đến và đi, giúp người dùng dễ dàng theo dõi tình trạng và nội dung của từng tài liệu.
* **Quản lý công việc hàng ngày**: Giúp người dùng lên kế hoạch, theo dõi và đánh giá tiến độ công việc, từ đó tối ưu hóa thời gian và hiệu quả làm việc.

# **II. Cấu trúc phần mềm**

## **1.Controller**

### a.HomeController

using Microsoft.AspNetCore.Mvc;

using QuanLyCongViec.Models;

using System.Diagnostics;

namespace QuanLyCongViec.Controllers

{

public class HomeController : Controller

{

private readonly ILogger<HomeController> \_logger;

public HomeController(ILogger<HomeController> logger)

{

\_logger = logger;

}

public IActionResult Index()

{

return View();

}

public IActionResult Info()

{

return View();

}

public IActionResult Error()

{

var errorViewModel = new ErrorViewModel

{

RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier

};

return View(errorViewModel);

}

}

}

**Các chức năng của HomeController:**

Khởi tạo (HomeController): Controller này sử dụng dependency injection để nhận một logger (ILogger<HomeController>) phục vụ cho việc ghi log, giúp theo dõi hoạt động và xử lý lỗi trong ứng dụng.

Index: Phương thức Index trả về view mặc định cho trang chính của ứng dụng. Đây là điểm khởi đầu cho người dùng khi truy cập vào ứng dụng.

Info: Phương thức Info trả về một view có thể hiển thị thông tin về ứng dụng hoặc các thông tin khác liên quan. (Nội dung cụ thể của view sẽ phụ thuộc vào việc bạn đã thiết kế như thế nào).

Error: Phương thức Error xử lý việc hiển thị thông tin lỗi khi có sự cố xảy ra trong ứng dụng. Nó tạo một đối tượng ErrorViewModel và gán cho thuộc tính RequestId giá trị ID của yêu cầu hiện tại (nếu có). Sau đó, nó trả về view lỗi với thông tin chi tiết để người dùng có thể hiểu hơn về vấn đề gặp phải.

### b.TaiKhoanController

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using QuanLyCongViec.Data;

using QuanLyCongViec.Models;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Authorization;

namespace QuanLyCongViec.Controllers

{

public class TaiKhoanController : Controller

{

private readonly QuanLyCongViecContext \_context;

public TaiKhoanController(QuanLyCongViecContext context)

{

\_context = context;

}

// GET: TaiKhoan

public async Task<IActionResult> Index()

{

var userId = HttpContext.Session.GetString("UserId");

if (userId == null)

{

return RedirectToAction("Login", "TaiKhoan"); // Chuyển hướng đến trang đăng nhập nếu chưa đăng nhập

}

var taiKhoan = await \_context.TaiKhoan.FindAsync(int.Parse(userId));

if (taiKhoan == null)

{

return NotFound(); // Trả về lỗi nếu không tìm thấy tài khoản

}

return View(taiKhoan); // Trả về thông tin tài khoản

}

// GET: TaiKhoan/Edit/5

public async Task<IActionResult> Edit(int? id)

{

if (id == null)

{

return NotFound();

}

var taiKhoan = await \_context.TaiKhoan.FindAsync(id);

if (taiKhoan == null)

{

return NotFound();

}

return View(taiKhoan);

}

// POST: TaiKhoan/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("MaTaiKhoan,TenDangNhap,MatKhau,HoTen,Email")] TaiKhoan taiKhoan)

{

if (id != taiKhoan.MaTaiKhoan)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

// Tìm tài khoản hiện tại

var existingTaiKhoan = await \_context.TaiKhoan.AsNoTracking().FirstOrDefaultAsync(t => t.MaTaiKhoan == id);

if (existingTaiKhoan == null)

{

return NotFound();

}

// Giữ tên đăng nhập cũ

taiKhoan.TenDangNhap = existingTaiKhoan.TenDangNhap;

// Nếu mật khẩu rỗng, giữ mật khẩu cũ

if (string.IsNullOrEmpty(taiKhoan.MatKhau))

{

taiKhoan.MatKhau = existingTaiKhoan.MatKhau; // Giữ mật khẩu cũ

}

// Cập nhật thông tin tài khoản

\_context.Update(taiKhoan);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!TaiKhoanExists(taiKhoan.MaTaiKhoan))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

return View(taiKhoan);

}

// GET: TaiKhoan/Delete/5

public async Task<IActionResult> Delete(int? id)

{

if (id == null)

{

return NotFound();

}

var taiKhoan = await \_context.TaiKhoan

.FirstOrDefaultAsync(m => m.MaTaiKhoan == id);

if (taiKhoan == null)

{

return NotFound();

}

return View(taiKhoan);

}

// POST: TaiKhoan/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var taiKhoan = await \_context.TaiKhoan.FindAsync(id);

if (taiKhoan != null)

{

\_context.TaiKhoan.Remove(taiKhoan);

await \_context.SaveChangesAsync();

}

// Xóa thông tin phiên làm việc

HttpContext.Session.Clear();

return RedirectToAction("Login", "TaiKhoan"); // Quay lại trang đăng nhập

}

private bool TaiKhoanExists(int id)

{

return \_context.TaiKhoan.Any(e => e.MaTaiKhoan == id);

}

// GET: TaiKhoan/Login

[HttpGet]

[AllowAnonymous]

public IActionResult Login()

{

return View();

}

// POST: TaiKhoan/Login

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Login(string tenDangNhap, string matKhau)

{

var taiKhoan = await \_context.TaiKhoan

.FirstOrDefaultAsync(m => m.TenDangNhap == tenDangNhap && m.MatKhau == matKhau);

if (taiKhoan != null)

{

// Đăng nhập thành công, lưu thông tin vào session

HttpContext.Session.SetString("UserId", taiKhoan.MaTaiKhoan.ToString());

return RedirectToAction("Index", "Home");

}

// Thêm lỗi khi đăng nhập thất bại

ModelState.AddModelError("LoginFailed", "Tên đăng nhập hoặc mật khẩu không đúng.");

return View();

}

// POST: TaiKhoan/Logout

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Logout()

{

HttpContext.Session.Clear(); // Xóa thông tin phiên làm việc

return RedirectToAction("Index", "Home"); // Quay lại trang chính

}

// GET: TaiKhoan/Register

[HttpGet]

[AllowAnonymous]

public IActionResult Register()

{

return View();

}

// POST: TaiKhoan/Register

[HttpPost]

[ValidateAntiForgeryToken]

[AllowAnonymous]

public async Task<IActionResult> Register([Bind("TenDangNhap,MatKhau,HoTen,Email")] TaiKhoan taiKhoan)

{

if (ModelState.IsValid)

{

// Kiểm tra tên đăng nhập đã tồn tại

if (await \_context.TaiKhoan.AnyAsync(t => t.TenDangNhap == taiKhoan.TenDangNhap))

{

ModelState.AddModelError("TenDangNhap", "Tên đăng nhập đã tồn tại.");

return View(taiKhoan);

}

// Thêm tài khoản mới

\_context.Add(taiKhoan);

await \_context.SaveChangesAsync();

return RedirectToAction("Login");

}

ModelState.AddModelError("RegistrationFailed", "Vui lòng nhập đầy đủ và chính xác thông tin.");

return View(taiKhoan);

}

}

}

**Các chức năng của controller TaiKhoanController:**

Index: Phương thức này hiển thị thông tin tài khoản của người dùng đã đăng nhập. Đầu tiên, nó kiểm tra xem người dùng đã đăng nhập hay chưa thông qua Session. Nếu không có thông tin người dùng trong session, nó sẽ chuyển hướng đến trang đăng nhập (Login). Nếu có, nó sẽ tìm kiếm tài khoản dựa trên ID người dùng và trả về thông tin tài khoản cho view. Nếu không tìm thấy tài khoản, nó sẽ trả về lỗi 404.

Edit (GET): Phương thức này hiển thị trang chỉnh sửa thông tin tài khoản. Nó nhận ID tài khoản từ URL và kiểm tra tính hợp lệ. Nếu ID không hợp lệ (null), nó trả về lỗi 404. Nếu ID hợp lệ, nó tìm kiếm tài khoản trong cơ sở dữ liệu và trả về thông tin cho view chỉnh sửa.

Edit (POST): Phương thức này xử lý yêu cầu chỉnh sửa thông tin tài khoản. Nó nhận thông tin tài khoản từ form và kiểm tra tính hợp lệ của ID tài khoản. Nếu ID không khớp với tài khoản được gửi, nó sẽ trả về lỗi 404. Nếu dữ liệu hợp lệ, nó sẽ tìm kiếm tài khoản hiện tại trong cơ sở dữ liệu. Nếu tài khoản tồn tại, nó sẽ cập nhật thông tin mới, giữ nguyên tên đăng nhập và mật khẩu cũ nếu người dùng không nhập mật khẩu mới. Cuối cùng, nó lưu thay đổi vào cơ sở dữ liệu và chuyển hướng về trang Index.

Delete (GET): Phương thức này hiển thị trang xác nhận việc xóa tài khoản. Nó nhận ID tài khoản từ URL và kiểm tra tính hợp lệ. Nếu ID không hợp lệ hoặc không tìm thấy tài khoản, nó sẽ trả về lỗi 404. Nếu tài khoản tồn tại, nó sẽ trả về thông tin tài khoản để người dùng xác nhận việc xóa.

Delete (POST): Phương thức này xử lý xác nhận xóa tài khoản. Sau khi người dùng xác nhận, nó tìm kiếm tài khoản trong cơ sở dữ liệu và xóa tài khoản nếu tồn tại. Sau khi xóa, nó cũng xóa thông tin phiên làm việc khỏi session và chuyển hướng đến trang đăng nhập.

Login (GET): Phương thức này hiển thị trang đăng nhập cho người dùng. Đây là điểm khởi đầu cho người dùng mới hoặc những người đã đăng xuất.

Login (POST): Phương thức này xử lý yêu cầu đăng nhập. Nó nhận tên đăng nhập và mật khẩu từ form, kiểm tra thông tin tài khoản trong cơ sở dữ liệu. Nếu thông tin hợp lệ, nó sẽ lưu ID người dùng vào session và chuyển hướng đến trang chính (Index của Home). Nếu không, nó sẽ thông báo lỗi cho người dùng về tên đăng nhập hoặc mật khẩu không đúng.

Logout: Phương thức này xử lý yêu cầu đăng xuất. Khi người dùng thực hiện đăng xuất, thông tin phiên làm việc sẽ được xóa khỏi session và người dùng sẽ được chuyển hướng về trang chính.

Register (GET): Phương thức này hiển thị trang đăng ký cho người dùng mới. Đây là nơi người dùng có thể nhập thông tin để tạo tài khoản mới.

Register (POST): Phương thức này xử lý yêu cầu đăng ký tài khoản mới. Nó nhận thông tin tài khoản từ form và kiểm tra tính hợp lệ. Nếu tên đăng nhập đã tồn tại trong cơ sở dữ liệu, nó sẽ thông báo lỗi cho người dùng. Nếu thông tin hợp lệ, nó sẽ thêm tài khoản mới vào cơ sở dữ liệu và chuyển hướng người dùng đến trang đăng nhập để thực hiện việc đăng nhập.

### c.VanBanDenController

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using QuanLyCongViec.Data;

using QuanLyCongViec.Models;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

namespace QuanLyCongViec.Controllers

{

public class VanBanDenController : Controller

{

private readonly QuanLyCongViecContext \_context;

public VanBanDenController(QuanLyCongViecContext context)

{

\_context = context;

}

// Kiểm tra đăng nhập trước mỗi action

private bool IsUserLoggedIn()

{

return HttpContext.Session.GetString("UserId") != null;

}

// GET: VanBanDen

public async Task<IActionResult> Index()

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

return View(await \_context.VanBanDen.ToListAsync());

}

// GET: VanBanDen/Create

public IActionResult Create()

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

return View();

}

// POST: VanBanDen/Create

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("SoVanBan,NgayNhan,TrichYeu,DonViGui,NoiNhan")] VanBanDen vanBanDen)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (ModelState.IsValid)

{

try

{

\_context.Add(vanBanDen);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

catch (DbUpdateException ex)

{

ModelState.AddModelError("", "Có lỗi xảy ra khi lưu dữ liệu: " + ex.Message);

}

}

return View(vanBanDen);

}

// GET: VanBanDen/Edit/5

public async Task<IActionResult> Edit(int? id)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (id == null)

{

return NotFound();

}

var vanBanDen = await \_context.VanBanDen.FindAsync(id);

if (vanBanDen == null)

{

return NotFound();

}

return View(vanBanDen);

}

// POST: VanBanDen/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("MaVanBanDen,SoVanBan,NgayNhan,TrichYeu,DonViGui,NoiNhan")] VanBanDen vanBanDen)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (id != vanBanDen.MaVanBanDen)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(vanBanDen);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!VanBanDenExists(vanBanDen.MaVanBanDen))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

return View(vanBanDen);

}

// GET: VanBanDen/Delete/5

public async Task<IActionResult> Delete(int? id)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (id == null)

{

return NotFound();

}

var vanBanDen = await \_context.VanBanDen

.FirstOrDefaultAsync(m => m.MaVanBanDen == id);

if (vanBanDen == null)

{

return NotFound();

}

return View(vanBanDen);

}

// POST: VanBanDen/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

var vanBanDen = await \_context.VanBanDen.FindAsync(id);

if (vanBanDen != null)

{

\_context.VanBanDen.Remove(vanBanDen);

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

private bool VanBanDenExists(int id)

{

return \_context.VanBanDen.Any(e => e.MaVanBanDen == id);

}

}

}

**Các chức năng của VanBanDenController**

Khởi tạo (VanBanDenController): Controller này nhận vào một đối tượng QuanLyCongViecContext để truy cập cơ sở dữ liệu cho các thao tác với bảng VanBanDen.

Kiểm tra đăng nhập: Phương thức riêng IsUserLoggedIn() được sử dụng để kiểm tra xem người dùng đã đăng nhập hay chưa bằng cách kiểm tra giá trị trong session (UserId). Nếu chưa đăng nhập, người dùng sẽ được chuyển hướng đến trang đăng nhập.

Index: Phương thức Index hiển thị danh sách tất cả các văn bản đến (VanBanDen). Nếu người dùng chưa đăng nhập, họ sẽ được chuyển hướng đến trang đăng nhập. Nếu đã đăng nhập, phương thức sẽ truy vấn và trả về danh sách các văn bản đến cho view.

Create (GET): Phương thức này hiển thị trang để người dùng tạo mới một văn bản đến. Nó kiểm tra xem người dùng đã đăng nhập hay chưa, nếu chưa, sẽ chuyển hướng đến trang đăng nhập.

Create (POST): Phương thức này xử lý việc tạo mới một văn bản đến. Nó nhận thông tin từ form, kiểm tra tính hợp lệ của dữ liệu và thêm văn bản đến vào cơ sở dữ liệu nếu hợp lệ. Nếu có lỗi xảy ra trong quá trình lưu, thông báo lỗi sẽ được hiển thị cho người dùng.

Edit (GET): Phương thức này hiển thị trang chỉnh sửa thông tin của một văn bản đến dựa trên ID đã cho. Nếu người dùng chưa đăng nhập hoặc ID không hợp lệ, họ sẽ được chuyển hướng đến trang đăng nhập hoặc nhận lỗi 404 nếu không tìm thấy văn bản đến.

Edit (POST): Phương thức này xử lý việc cập nhật thông tin văn bản đến. Nó kiểm tra ID có khớp với văn bản đến được gửi hay không. Nếu hợp lệ, nó sẽ cập nhật thông tin văn bản trong cơ sở dữ liệu và chuyển hướng về trang danh sách. Nếu không tìm thấy văn bản hoặc xảy ra lỗi, sẽ có thông báo lỗi tương ứng.

Delete (GET): Phương thức này hiển thị trang xác nhận việc xóa một văn bản đến dựa trên ID. Nếu người dùng chưa đăng nhập hoặc ID không hợp lệ, họ sẽ được chuyển hướng đến trang đăng nhập hoặc nhận lỗi 404 nếu không tìm thấy văn bản.

Delete (POST): Phương thức này xử lý việc xác nhận xóa văn bản đến. Sau khi xác nhận, nó sẽ tìm và xóa văn bản từ cơ sở dữ liệu. Nếu văn bản không tồn tại, sẽ không làm gì và sẽ chuyển hướng về trang danh sách.

Kiểm tra tồn tại văn bản: Phương thức VanBanDenExists(int id) kiểm tra xem một văn bản đến với ID cụ thể có tồn tại trong cơ sở dữ liệu hay không.

### d.VanBanDiController

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using QuanLyCongViec.Data;

using QuanLyCongViec.Models;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

namespace QuanLyCongViec.Controllers

{

public class VanBanDiController : Controller

{

private readonly QuanLyCongViecContext \_context;

public VanBanDiController(QuanLyCongViecContext context)

{

\_context = context;

}

// Kiểm tra đăng nhập trước mỗi action

private bool IsUserLoggedIn()

{

return HttpContext.Session.GetString("UserId") != null;

}

// GET: VanBanDi

public async Task<IActionResult> Index()

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

return View(await \_context.VanBanDi.ToListAsync());

}

// GET: VanBanDi/Create

public IActionResult Create()

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

return View();

}

// POST: VanBanDi/Create

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("SoVanBan,NgayGui,TrichYeu,DonViNhan")] VanBanDi vanBanDi)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

// Kiểm tra xem số văn bản đã tồn tại hay chưa

bool exists = await \_context.VanBanDi.AnyAsync(v => v.SoVanBan == vanBanDi.SoVanBan);

if (exists)

{

ModelState.AddModelError("SoVanBan", "Số văn bản đã tồn tại. Vui lòng nhập số văn bản khác.");

}

if (ModelState.IsValid)

{

try

{

\_context.Add(vanBanDi);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

catch (DbUpdateException ex)

{

ModelState.AddModelError("", "Có lỗi xảy ra khi lưu dữ liệu: " + ex.Message);

}

}

return View(vanBanDi);

}

// GET: VanBanDi/Edit/5

public async Task<IActionResult> Edit(int? id)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (id == null)

{

return NotFound();

}

var vanBanDi = await \_context.VanBanDi.FindAsync(id);

if (vanBanDi == null)

{

return NotFound();

}

return View(vanBanDi);

}

// POST: VanBanDi/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("MaVanBanDi,SoVanBan,NgayGui,TrichYeu,DonViNhan")] VanBanDi vanBanDi)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (id != vanBanDi.MaVanBanDi)

{

return NotFound();

}

// Kiểm tra xem số văn bản đã tồn tại hay chưa (trừ văn bản hiện tại)

bool exists = await \_context.VanBanDi

.AnyAsync(v => v.SoVanBan == vanBanDi.SoVanBan && v.MaVanBanDi != id);

if (exists)

{

ModelState.AddModelError("SoVanBan", "Số văn bản đã tồn tại. Vui lòng nhập số văn bản khác.");

}

if (ModelState.IsValid)

{

try

{

\_context.Update(vanBanDi);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

catch (DbUpdateConcurrencyException)

{

if (!VanBanDiExists(vanBanDi.MaVanBanDi))

{

return NotFound();

}

else

{

throw;

}

}

}

return View(vanBanDi);

}

private bool VanBanDiExists(int id)

{

return \_context.VanBanDi.Any(e => e.MaVanBanDi == id);

}

// GET: VanBanDi/Delete/5

public async Task<IActionResult> Delete(int? id)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

if (id == null)

{

return NotFound();

}

var vanBanDi = await \_context.VanBanDi

.FirstOrDefaultAsync(m => m.MaVanBanDi == id);

if (vanBanDi == null)

{

return NotFound();

}

return View(vanBanDi);

}

// POST: VanBanDi/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

var vanBanDi = await \_context.VanBanDi.FindAsync(id);

if (vanBanDi != null)

{

\_context.VanBanDi.Remove(vanBanDi);

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

}

}

**Các chức năng của VanBanDiController**

Khởi tạo (VanBanDiController): Controller này khởi tạo một đối tượng QuanLyCongViecContext, cho phép truy cập vào cơ sở dữ liệu và thực hiện các thao tác liên quan đến văn bản đi (VanBanDi).

Kiểm tra đăng nhập: Phương thức riêng IsUserLoggedIn() được sử dụng để xác định xem người dùng đã đăng nhập hay chưa bằng cách kiểm tra session UserId. Nếu người dùng chưa đăng nhập, họ sẽ được chuyển hướng đến trang đăng nhập.

Index: Phương thức Index trả về danh sách tất cả các văn bản đi có trong cơ sở dữ liệu. Nếu người dùng chưa đăng nhập, họ sẽ được chuyển hướng đến trang đăng nhập. Nếu đã đăng nhập, phương thức sẽ trả về danh sách văn bản đi cho view.

Create (GET): Phương thức này hiển thị form để người dùng tạo mới một văn bản đi. Nó cũng kiểm tra xem người dùng đã đăng nhập hay chưa, nếu chưa sẽ chuyển hướng đến trang đăng nhập.

Create (POST): Phương thức này xử lý việc tạo mới một văn bản đi. Sau khi nhận dữ liệu từ form, nó kiểm tra xem số văn bản đã tồn tại trong cơ sở dữ liệu hay chưa. Nếu số văn bản đã tồn tại, sẽ có thông báo lỗi và dữ liệu không được lưu. Nếu không có lỗi, văn bản sẽ được thêm vào cơ sở dữ liệu và người dùng sẽ được chuyển hướng về trang danh sách.

Edit (GET): Phương thức này hiển thị form để chỉnh sửa thông tin của một văn bản đi dựa trên ID. Nếu người dùng chưa đăng nhập hoặc ID không hợp lệ, họ sẽ được chuyển hướng đến trang đăng nhập hoặc nhận lỗi 404 nếu không tìm thấy văn bản đi.

Edit (POST): Phương thức này xử lý việc cập nhật thông tin văn bản đi. Nó kiểm tra xem ID của văn bản có khớp với ID trong cơ sở dữ liệu hay không và kiểm tra xem số văn bản đã tồn tại hay chưa (trừ văn bản đang được chỉnh sửa). Nếu không có lỗi, thông tin sẽ được cập nhật trong cơ sở dữ liệu. Nếu xảy ra lỗi trong quá trình lưu, sẽ có thông báo lỗi cho người dùng.

Delete (GET): Phương thức này hiển thị trang xác nhận việc xóa một văn bản đi. Nếu người dùng chưa đăng nhập hoặc ID không hợp lệ, họ sẽ được chuyển hướng đến trang đăng nhập hoặc nhận lỗi 404 nếu không tìm thấy văn bản đi.

Delete (POST): Phương thức này xử lý việc xác nhận xóa văn bản đi. Sau khi xác nhận, nó sẽ tìm văn bản trong cơ sở dữ liệu và nếu tồn tại, nó sẽ xóa văn bản đó và lưu thay đổi. Người dùng sẽ được chuyển hướng về trang danh sách.

Kiểm tra tồn tại văn bản: Phương thức VanBanDiExists(int id) kiểm tra xem một văn bản đi với ID cụ thể có tồn tại trong cơ sở dữ liệu hay không.

### e.ThongKeController

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using QuanLyCongViec.Data;

using QuanLyCongViec.Models;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

namespace QuanLyCongViec.Controllers

{

public class ThongKeController : Controller

{

private readonly QuanLyCongViecContext \_context;

public ThongKeController(QuanLyCongViecContext context)

{

\_context = context;

}

// Kiểm tra đăng nhập trước khi truy cập vào action

private bool IsUserLoggedIn()

{

return HttpContext.Session.GetString("UserId") != null;

}

public async Task<IActionResult> Index(string searchString)

{

if (!IsUserLoggedIn())

{

return RedirectToAction("Login", "TaiKhoan");

}

// Truy xuất danh sách văn bản đến và đi

var vanBanDen = \_context.VanBanDen.AsQueryable();

var vanBanDi = \_context.VanBanDi.AsQueryable();

// Tìm kiếm theo điều kiện

if (!string.IsNullOrEmpty(searchString))

{

vanBanDen = vanBanDen.Where(v => v.NoiNhan.Contains(searchString)

|| v.DonViGui.Contains(searchString)

|| v.TrichYeu.Contains(searchString));

vanBanDi = vanBanDi.Where(v => v.DonViNhan.Contains(searchString)

|| v.TrichYeu.Contains(searchString));

}

var viewModel = new ThongKeViewModel

{

VanBanDen = await vanBanDen.ToListAsync(),

VanBanDi = await vanBanDi.ToListAsync()

};

return View(viewModel);

}

}

}

**Các chức năng của ThongKeController**

Khởi tạo (ThongKeController): Controller này được khởi tạo với một đối tượng QuanLyCongViecContext, cho phép truy cập vào cơ sở dữ liệu để thực hiện các thao tác thống kê liên quan đến văn bản đến (VanBanDen) và văn bản đi (VanBanDi).

Kiểm tra đăng nhập: Phương thức riêng IsUserLoggedIn() xác định xem người dùng đã đăng nhập hay chưa thông qua việc kiểm tra giá trị trong session UserId. Nếu người dùng chưa đăng nhập, họ sẽ được chuyển hướng đến trang đăng nhập.

Index: Phương thức Index thực hiện hai nhiệm vụ chính:

Kiểm tra đăng nhập: Trước tiên, nó xác minh người dùng đã đăng nhập hay chưa. Nếu không, người dùng sẽ được chuyển hướng đến trang đăng nhập.

Truy xuất và tìm kiếm văn bản:

Controller lấy danh sách tất cả các văn bản đến và văn bản đi từ cơ sở dữ liệu bằng cách sử dụng AsQueryable(), cho phép dễ dàng áp dụng các điều kiện tìm kiếm.

Nếu người dùng nhập vào một chuỗi tìm kiếm (searchString), controller sẽ lọc các văn bản dựa trên nội dung của các trường như NoiNhan, DonViGui, và TrichYeu cho văn bản đến, và DonViNhan cùng TrichYeu cho văn bản đi.

Kết quả tìm kiếm sau đó sẽ được đưa vào một view model ThongKeViewModel, bao gồm cả danh sách văn bản đến và văn bản đi.

Trả về view: Cuối cùng, controller trả về view Index, với view model chứa dữ liệu cần thiết để hiển thị cho người dùng.

## **2.Models**

### a.TaiKhoan

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace QuanLyCongViec.Models

{

public class TaiKhoan

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int MaTaiKhoan { get; set; }

[Required(ErrorMessage = "Tên đăng nhập là bắt buộc")]

[Display(Name = "Tên đăng nhập")]

public string TenDangNhap { get; set; }

[Required(ErrorMessage = "Mật khẩu là bắt buộc")]

[Display(Name = "Mật khẩu")]

[DataType(DataType.Password)]

public string MatKhau { get; set; }

[Required(ErrorMessage = "Họ tên là bắt buộc")]

[Display(Name = "Họ tên")]

public string HoTen { get; set; }

[EmailAddress(ErrorMessage = "Email không hợp lệ")]

public string Email { get; set; }

}

}

### b.VanBanDen

using System;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace QuanLyCongViec.Models

{

public class VanBanDen

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int MaVanBanDen { get; set; }

[Required(ErrorMessage = "Số văn bản là bắt buộc")]

[Display(Name = "Số văn bản")]

public string SoVanBan { get; set; }

[Required(ErrorMessage = "Ngày nhận là bắt buộc")]

[Display(Name = "Ngày nhận")]

[DataType(DataType.Date)]

public DateTime NgayNhan { get; set; }

[Required(ErrorMessage = "Trích yếu là bắt buộc")]

[Display(Name = "Trích yếu")]

public string TrichYeu { get; set; }

[Required(ErrorMessage = "Đơn vị gửi là bắt buộc")]

[Display(Name = "Đơn vị gửi")]

public string DonViGui { get; set; }

[Required(ErrorMessage = "Nơi nhận là bắt buộc")]

[Display(Name = "Nơi nhận")]

public string NoiNhan { get; set; }

}

}

### c.VanBanDi

using System;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace QuanLyCongViec.Models

{

public class VanBanDi

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int MaVanBanDi { get; set; }

[Required(ErrorMessage = "Số văn bản là bắt buộc")]

[Display(Name = "Số văn bản")]

public string SoVanBan { get; set; }

[Required(ErrorMessage = "Ngày gửi là bắt buộc")]

[Display(Name = "Ngày gửi")]

[DataType(DataType.Date)]

public DateTime NgayGui { get; set; }

[Required(ErrorMessage = "Trích yếu là bắt buộc")]

[Display(Name = "Trích yếu")]

public string TrichYeu { get; set; }

[Required(ErrorMessage = "Đơn vị nhận là bắt buộc")]

[Display(Name = "Đơn vị nhận")]

public string DonViNhan { get; set; }

}

}

### d.ThongKe

using System.Collections.Generic;

namespace QuanLyCongViec.Models

{

public class ThongKeViewModel

{

public List<VanBanDen> VanBanDen { get; set; }

public List<VanBanDi> VanBanDi { get; set; }

}

}

### e.ErrorViewModel

A screenshot of a computer

Description automatically generated

Lớp ErrorViewModel sử dụng để hiển thị thông tin lỗi cho người dùng, giúp họ có được thông tin cụ thể hơn về lỗi mà họ gặp phải trong ứng dụng.

## **3.Views**

### a.Home

**Index(Trang Chính)**

@{

ViewData["Title"] = "Trang Chủ";

}

<div class="d-flex flex-column justify-content-center align-items-center vh-100">

<div class="text-center">

<h1 class="display-4 mb-4">Chào mừng đến với hệ thống quản lý công việc</h1>

<p class="lead mb-4">Giải pháp tốt nhất để quản lý công việc của bạn một cách hiệu quả và tiện lợi.</p>

@if (Context.Session.GetString("UserId") == null)

{

<**a** **asp-controller**="TaiKhoan" **asp-action**="Login" class="btn btn-primary btn-lg">Đăng nhập</**a**>

}

else

{

<p>Bạn đã đăng nhập thành công!</p>

}

</div>

</div>

**Info(Trang Giới Thiệu)**

@{

ViewData["Title"] = "Giới Thiệu";

}

<div class="container mt-4">

<h2 class="text-center mb-4">Phần mềm điều hành quản lý công việc</h2>

<div class="text-center mb-4">

<p>Đây là phần mềm giúp quản lý công việc một cách hiệu quả và tiện lợi, giúp người dùng dễ dàng theo dõi và quản lý các văn bản, công việc hàng ngày.</p>

</div>

<h4 class="text-center mb-3">Thông tin thực hiện:</h4>

<ul class="list-unstyled text-center">

<li>Nguyễn Đình Lực - 1822041377</li>

<li>Phạm Khánh Duy - 1822041255</li>

</ul>

<h4 class="text-center mb-3">Giảng viên hướng dẫn:</h4>

<p class="text-center">Huỳnh Hoàng Tân</p>

<h4 class="text-center mt-4">Cảm ơn!</h4>

</div>

### b.TaiKhoan

**Index**

@model QuanLyCongViec.Models.TaiKhoan

@{

ViewData["Title"] = "Trung tâm tài khoản";

}

<h1>Trung tâm tài khoản</h1>

<table class="table">

<thead>

<tr>

<th>Tên đăng nhập</th>

<th>Họ tên</th>

<th>Email</th>

<th>Thao tác</th>

</tr>

</thead>

<tbody>

<tr>

<td>

@Html.DisplayFor(model => model.TenDangNhap)

</td>

<td>

@Html.DisplayFor(model => model.HoTen)

</td>

<td>

@Html.DisplayFor(model => model.Email)

</td>

<td>

<**a** **asp-action**="Edit" **asp-route-id**="@Model.MaTaiKhoan">Sửa</**a**> |

<**a** **asp-action**="Delete" **asp-route-id**="@Model.MaTaiKhoan">Xóa</**a**>

</td>

</tr>

</tbody>

</table>

<p>

<**a** **asp-controller**="Home" **asp-action**="Index">Quay lại trang chính</**a**>

</p>

**Login**

@{

ViewData["Title"] = "Đăng Nhập";

}

<div class="d-flex justify-content-center align-items-center vh-100">

<div class="text-center login-form">

<h2 class="mb-4">Đăng nhập</h2>

<!-- Thông báo lỗi đăng nhập -->

@if (ViewData.ModelState.ContainsKey("LoginFailed"))

{

<div class="alert alert-danger">

@ViewData.ModelState["LoginFailed"].Errors[0].ErrorMessage

</div>

}

<**form** **asp-controller**="TaiKhoan" **asp-action**="Login" method="post">

<div class="form-group mb-3">

<label for="tenDangNhap">Tên đăng nhập</label>

<input type="text" class="form-control" id="tenDangNhap" name="tenDangNhap" required />

</div>

<div class="form-group mb-4">

<label for="matKhau">Mật khẩu</label>

<input type="password" class="form-control" id="matKhau" name="matKhau" required />

</div>

<button type="submit" class="btn btn-primary btn-lg w-100">Đăng nhập</button>

<p class="mt-3">Chưa có tài khoản? <**a** **asp-controller**="TaiKhoan" **asp-action**="Register">Đăng ký ngay</**a**></p>

</**form**>

</div>

</div>

**Edit**

@model QuanLyCongViec.Models.TaiKhoan

@{

ViewData["Title"] = "Sửa Tài Khoản";

}

<h1>Sửa Tài Khoản</h1>

<**form** **asp-action**="Edit">

<**input** **type**="hidden" **asp-for**="MaTaiKhoan" />

<div class="form-group">

<**label** **asp-for**="TenDangNhap"></**label**>

<**input** **asp-for**="TenDangNhap" class="form-control" readonly />

<**span** **asp-validation-for**="TenDangNhap" class="text-danger"></**span**>

</div>

<div class="form-group">

<**label** **asp-for**="MatKhau"></**label**>

<**input** **asp-for**="MatKhau" class="form-control" />

<**span** **asp-validation-for**="MatKhau" class="text-danger"></**span**>

</div>

<div class="form-group">

<**label** **asp-for**="HoTen"></**label**>

<**input** **asp-for**="HoTen" class="form-control" />

<**span** **asp-validation-for**="HoTen" class="text-danger"></**span**>

</div>

<div class="form-group">

<**label** **asp-for**="Email"></**label**>

<**input** **asp-for**="Email" class="form-control" />

<**span** **asp-validation-for**="Email" class="text-danger"></**span**>

</div>

<button type="submit" class="btn btn-primary">Lưu</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

**Detele**

@model QuanLyCongViec.Models.TaiKhoan

@{

ViewData["Title"] = "Xóa tài khoản";

}

<h1>Xóa tài khoản</h1>

<h4>Bạn có chắc chắn muốn xóa tài khoản này?</h4>

<div>

<h5>@Model.HoTen</h5>

<p>Địa chỉ email: @Model.Email</p>

</div>

<**form** **asp-action**="Delete">

<**input** **type**="hidden" **asp-for**="MaTaiKhoan" />

<div class="form-group">

<input type="submit" value="Xóa" class="btn btn-danger" />

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</div>

</**form**>

**Register**

@{

ViewData["Title"] = "Đăng Ký Tài Khoản";

}

<div class="d-flex justify-content-center align-items-center vh-100">

<div class="text-center login-form">

<h2 class="mb-4">Đăng Ký Tài Khoản</h2>

@if (!ViewData.ModelState.IsValid)

{

<div class="alert alert-danger">

@foreach (var error in ViewData.ModelState.Values.SelectMany(v => v.Errors))

{

<div>@error.ErrorMessage</div>

}

</div>

}

<**form** **asp-controller**="TaiKhoan" **asp-action**="Register" method="post">

<div class="form-group mb-3">

<label for="tenDangNhap">Tên đăng nhập</label>

<input type="text" class="form-control" id="tenDangNhap" name="TenDangNhap" required />

</div>

<div class="form-group mb-3">

<label for="matKhau">Mật khẩu</label>

<input type="password" class="form-control" id="matKhau" name="MatKhau" required />

</div>

<div class="form-group mb-3">

<label for="hoTen">Họ tên</label>

<input type="text" class="form-control" id="hoTen" name="HoTen" required />

</div>

<div class="form-group mb-4">

<label for="email">Email</label>

<input type="email" class="form-control" id="email" name="Email" required />

</div>

<button type="submit" class="btn btn-primary btn-lg w-100">Đăng Ký</button>

<p class="mt-3">Đã có tài khoản? <**a** **asp-controller**="TaiKhoan" **asp-action**="Login">Đăng nhập ngay</**a**></p>

</**form**>

</div>

</div>

### c.VanBanDen

**Index**

@model IEnumerable<QuanLyCongViec.Models.VanBanDen>

@{

ViewData["Title"] = "Danh Sách Văn Bản Đến";

}

<h2>Danh Sách Văn Bản Đến</h2>

<table class="table">

<thead>

<tr>

<th>Số Văn Bản</th>

<th>Ngày Nhận</th>

<th>Trích Yếu</th>

<th>Đơn Vị Gửi</th>

<th>Nơi Nhận</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr>

<td>@item.SoVanBan</td>

<td>@item.NgayNhan.ToString("dd/MM/yyyy")</td>

<td>@item.TrichYeu</td>

<td>@item.DonViGui</td>

<td>@item.NoiNhan</td>

<td>

<**a** **asp-action**="Edit" **asp-route-id**="@item.MaVanBanDen" class="btn btn-warning">Sửa</**a**>

<**a** **asp-action**="Delete" **asp-route-id**="@item.MaVanBanDen" class="btn btn-danger">Xóa</**a**>

</td>

</tr>

}

</tbody>

</table>

<**a** **asp-action**="Create" class="btn btn-primary">Tạo Văn Bản Đến Mới</**a**>

**Create**

@model QuanLyCongViec.Models.VanBanDen

@{

ViewData["Title"] = "Tạo Văn Bản Đến Mới";

}

<div class="container">

<h2>Tạo Văn Bản Đến Mới</h2>

<**form** **asp-action**="Create" method="post">

<div class="form-group mb-3">

<label for="soVanBan">Số Văn Bản</label>

<input type="text" class="form-control" id="soVanBan" name="SoVanBan" required />

</div>

<div class="form-group mb-3">

<label for="ngayNhan">Ngày Nhận</label>

<input type="date" class="form-control" id="ngayNhan" name="NgayNhan" required />

</div>

<div class="form-group mb-3">

<label for="trichYeu">Trích Yếu</label>

<textarea class="form-control" id="trichYeu" name="TrichYeu" rows="3" required></textarea>

</div>

<div class="form-group mb-4">

<label for="donViGui">Đơn Vị Gửi</label>

<input type="text" class="form-control" id="donViGui" name="DonViGui" required />

</div>

<div class="form-group mb-4">

<label for="noiNhan">Nơi Nhận</label>

<input type="text" class="form-control" id="noiNhan" name="NoiNhan" required />

</div>

<button type="submit" class="btn btn-primary">Gửi Văn Bản</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

</div>

**Edit**

@model QuanLyCongViec.Models.VanBanDen

@{

ViewData["Title"] = "Chỉnh Sửa Văn Bản Đến";

}

<div class="container">

<h2 class="my-4">Chỉnh Sửa Văn Bản Gửi Đến</h2>

@if (!ViewData.ModelState.IsValid)

{

<div class="alert alert-danger">

@foreach (var error in ViewData.ModelState.Values.SelectMany(v => v.Errors))

{

<div>@error.ErrorMessage</div>

}

</div>

}

<**form** **asp-action**="Edit" method="post">

<input type="hidden" name="MaVanBanDen" value="@Model.MaVanBanDen" />

<div class="form-group mb-3">

<label for="soVanBan">Số Văn Bản</label>

<input type="text" class="form-control" id="soVanBan" name="SoVanBan" value="@Model.SoVanBan" required />

</div>

<div class="form-group mb-3">

<label for="ngayGui">Ngày Nhận</label>

<input type="date" class="form-control" id="ngayNhan" name="NgayNhan" value="@Model.NgayNhan.ToString("yyyy-MM-dd")" required />

</div>

<div class="form-group mb-3">

<label for="trichYeu">Trích Yếu</label>

<textarea class="form-control" id="trichYeu" name="TrichYeu" rows="3" required>@Model.TrichYeu</textarea>

</div>

<div class="form-group mb-4">

<label for="donViNhan">Đơn Vị Gửi</label>

<input type="text" class="form-control" id="donViGui" name="DonViGui" value="@Model.DonViGui" required />

</div>

<div class="form-group mb-4">

<label for="donViNhan">Nơi Nhận</label>

<input type="text" class="form-control" id="noiNhan" name="NoiNhan" value="@Model.NoiNhan" required />

</div>

<button type="submit" class="btn btn-primary">Cập Nhật Văn Bản</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

</div>

**Detele**

@model QuanLyCongViec.Models.VanBanDen

<h2>Xác Nhận Xóa</h2>

<p>Bạn có chắc chắn muốn xóa văn bản này: <strong>@Model.SoVanBan</strong>?</p>

<dl class="row">

<dt class="col-sm-2">Số Văn Bản</dt>

<dd class="col-sm-10">@Model.SoVanBan</dd>

<dt class="col-sm-2">Ngày Nhận</dt>

<dd class="col-sm-10">@Model.NgayNhan.ToShortDateString()</dd>

<dt class="col-sm-2">Trích Yếu</dt>

<dd class="col-sm-10">@Model.TrichYeu</dd>

<dt class="col-sm-2">Đơn Vị Gửi</dt>

<dd class="col-sm-10">@Model.DonViGui</dd>

<dt class="col-sm-2">Nơi Nhận</dt>

<dd class="col-sm-10">@Model.NoiNhan</dd>

</dl>

<**form** **asp-action**="Delete" method="post">

<input type="hidden" name="id" value="@Model.MaVanBanDen" />

<button type="submit" class="btn btn-danger">Xóa</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

### d.VanBanDi

**Index**

@model IEnumerable<QuanLyCongViec.Models.VanBanDi>

@{

ViewData["Title"] = "Danh Sách Văn Bản Đi";

}

<div class="container">

<h2 class="my-4">Danh Sách Văn Bản Đi</h2>

<div class="mb-3">

<**a** **asp-action**="Create" class="btn btn-primary">Tạo Văn Bản Đi Mới</**a**>

</div>

@if (!Model.Any())

{

<p>Hiện chưa có văn bản nào được gửi đi.</p>

}

else

{

<table class="table">

<thead>

<tr>

<th>STT</th>

<th>Số Văn Bản</th>

<th>Ngày Gửi</th>

<th>Trích Yếu</th>

<th>Đơn Vị Nhận</th>

<th>Thao Tác</th>

</tr>

</thead>

<tbody>

@foreach (var item in Model.Select((value, index) => new { value, index }))

{

<tr>

<td>@(item.index + 1)</td>

<td>@Html.DisplayFor(m => item.value.SoVanBan)</td>

<td>@Html.DisplayFor(m => item.value.NgayGui)</td>

<td>@Html.DisplayFor(m => item.value.TrichYeu)</td>

<td>@Html.DisplayFor(m => item.value.DonViNhan)</td>

<td>

<**a** **asp-action**="Edit" **asp-route-id**="@item.value.MaVanBanDi">Sửa</**a**> |

<**a** **asp-action**="Delete" **asp-route-id**="@item.value.MaVanBanDi">Xóa</**a**>

</td>

</tr>

}

</tbody>

</table>

}

</div>

**Create**

@model QuanLyCongViec.Models.VanBanDi

@{

ViewData["Title"] = "Tạo Văn Bản Gửi Đi Mới";

}

<div class="container">

<h2 class="my-4">Tạo Văn Bản Gửi Đi Mới</h2>

@if (!ViewData.ModelState.IsValid)

{

<div class="alert alert-danger">

@foreach (var error in ViewData.ModelState.Values.SelectMany(v => v.Errors))

{

<div>@error.ErrorMessage</div>

}

</div>

}

<**form** **asp-action**="Create" method="post">

<div class="form-group mb-3">

<label for="soVanBan">Số Văn Bản</label>

<input type="text" class="form-control" id="soVanBan" name="SoVanBan" required />

</div>

<div class="form-group mb-3">

<label for="ngayGui">Ngày Gửi</label>

<input type="date" class="form-control" id="ngayGui" name="NgayGui" required />

</div>

<div class="form-group mb-3">

<label for="trichYeu">Trích Yếu</label>

<textarea class="form-control" id="trichYeu" name="TrichYeu" rows="3" required></textarea>

</div>

<div class="form-group mb-4">

<label for="donViNhan">Đơn Vị Nhận</label>

<input type="text" class="form-control" id="donViNhan" name="DonViNhan" required />

</div>

<button type="submit" class="btn btn-primary">Gửi Văn Bản</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

</div>

**Edit**

@model QuanLyCongViec.Models.VanBanDi

@{

ViewData["Title"] = "Chỉnh Sửa Văn Bản Gửi Đi";

}

<div class="container">

<h2 class="my-4">Chỉnh Sửa Văn Bản Gửi Đi</h2>

@if (!ViewData.ModelState.IsValid)

{

<div class="alert alert-danger">

@foreach (var error in ViewData.ModelState.Values.SelectMany(v => v.Errors))

{

<div>@error.ErrorMessage</div>

}

</div>

}

<**form** **asp-action**="Edit" method="post">

<input type="hidden" name="MaVanBanDi" value="@Model.MaVanBanDi" />

<div class="form-group mb-3">

<label for="soVanBan">Số Văn Bản</label>

<input type="text" class="form-control" id="soVanBan" name="SoVanBan" value="@Model.SoVanBan" required />

</div>

<div class="form-group mb-3">

<label for="ngayGui">Ngày Gửi</label>

<input type="date" class="form-control" id="ngayGui" name="NgayGui" value="@Model.NgayGui.ToString("yyyy-MM-dd")" required />

</div>

<div class="form-group mb-3">

<label for="trichYeu">Trích Yếu</label>

<textarea class="form-control" id="trichYeu" name="TrichYeu" rows="3" required>@Model.TrichYeu</textarea>

</div>

<div class="form-group mb-4">

<label for="donViNhan">Đơn Vị Nhận</label>

<input type="text" class="form-control" id="donViNhan" name="DonViNhan" value="@Model.DonViNhan" required />

</div>

<button type="submit" class="btn btn-primary">Cập Nhật Văn Bản</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

</div>

**Detele**

@model QuanLyCongViec.Models.VanBanDi

<h2>Xác Nhận Xóa</h2>

<p>Bạn có chắc chắn muốn xóa văn bản này: <strong>@Model.SoVanBan</strong>?</p>

<dl class="row">

<dt class="col-sm-2">Số Văn Bản</dt>

<dd class="col-sm-10">@Model.SoVanBan</dd>

<dt class="col-sm-2">Ngày Nhận</dt>

<dd class="col-sm-10">@Model.NgayGui.ToShortDateString()</dd>

<dt class="col-sm-2">Trích Yếu</dt>

<dd class="col-sm-10">@Model.TrichYeu</dd>

<dt class="col-sm-2">Đơn Vị Nhận</dt>

<dd class="col-sm-10">@Model.DonViNhan</dd>

</dl>

<**form** **asp-action**="Delete" method="post">

<input type="hidden" name="id" value="@Model.MaVanBanDi" />

<button type="submit" class="btn btn-danger">Xóa</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Hủy</**a**>

</**form**>

### e.ThongKe

@model ThongKeViewModel

<h2>Thống Kê Văn Bản</h2>

<**form** **asp-action**="Index" method="get" class="mb-4">

<div class="d-flex align-items-end">

<label for="searchString" class="me-2">Tìm kiếm:</label>

<input type="text" name="searchString" class="form-control me-2" placeholder="Nhập từ khóa..." />

<button type="submit" class="btn btn-primary">Tìm kiếm</button>

</div>

</**form**>

<h3>Văn Bản Đến</h3>

<table class="table">

<thead>

<tr>

<th>Số Văn Bản</th>

<th>Ngày Nhận</th>

<th>Trích Yếu</th>

<th>Đơn Vị Gửi</th>

<th>Nơi Nhận</th>

</tr>

</thead>

<tbody>

@if (Model.VanBanDen.Any())

{

foreach (var item in Model.VanBanDen)

{

<tr>

<td>@item.SoVanBan</td>

<td>@item.NgayNhan.ToShortDateString()</td>

<td>@item.TrichYeu</td>

<td>@item.DonViGui</td>

<td>@item.NoiNhan</td>

</tr>

}

}

else

{

<tr>

<td colspan="5" class="text-center">Không có dữ liệu văn bản đến.</td>

</tr>

}

</tbody>

</table>

<h3>Văn Bản Đi</h3>

<table class="table">

<thead>

<tr>

<th>Số Văn Bản</th>

<th>Ngày Gửi</th>

<th>Trích Yếu</th>

<th>Đơn Vị Nhận</th>

</tr>

</thead>

<tbody>

@if (Model.VanBanDi.Any())

{

foreach (var item in Model.VanBanDi)

{

<tr>

<td>@item.SoVanBan</td>

<td>@item.NgayGui.ToShortDateString()</td>

<td>@item.TrichYeu</td>

<td>@item.DonViNhan</td>

</tr>

}

}

else

{

<tr>

<td colspan="4" class="text-center">Không có dữ liệu văn bản đi.</td>

</tr>

}

</tbody>

</table>

### f.Shared

**Layout**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>@ViewData["Title"] - Quản lý điều hành công việc</title>

<**link** rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />

<**link** rel="stylesheet" **href**="~/css/site.css" **asp-append-version**="true" />

<style>

body {

background-color: #f8f9fa;

}

header {

background-color: #007bff;

color: white;

}

.navbar-brand, .nav-link {

color: white !important;

}

.nav-link:hover {

color: #ffc107 !important;

}

footer {

background-color: #343a40;

color: white;

text-align: center;

padding: 15px 0;

}

</style>

</head>

<body>

<header>

<nav class="navbar navbar-expand-lg navbar-light">

<div class="container">

<**a** class="navbar-brand" **asp-controller**="Home" **asp-action**="Index">Quản lý điều hành công việc</**a**>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav ms-auto">

@if (Context.Session.GetString("UserId") != null) // Kiểm tra nếu đã đăng nhập

{

<li class="nav-item">

<**a** class="nav-link" **asp-controller**="VanBanDen" **asp-action**="Index">Văn bản đến</**a**>

</li>

<li class="nav-item">

<**a** class="nav-link" **asp-controller**="VanBanDi" **asp-action**="Index">Văn bản đi</**a**>

</li>

<li class="nav-item">

<**a** class="nav-link" **asp-controller**="ThongKe" **asp-action**="Index">Thống kê</**a**>

</li>

<li class="nav-item">

<**a** class="nav-link" **asp-controller**="TaiKhoan" **asp-action**="Index">Quản lý tài khoản</**a**>

</li>

<li class="nav-item">

<**a** class="nav-link" **asp-controller**="Home" **asp-action**="Info">Giới Thiệu</**a**>

</li>

}

</ul>

<div>

@if (Context.Session.GetString("UserId") != null) // Kiểm tra nếu đã đăng nhập

{

<**form** **asp-controller**="TaiKhoan" **asp-action**="Logout" method="post" class="d-inline">

<button type="submit" class="btn btn-danger ms-3">Đăng xuất</button>

</**form**>

}

</div>

</div>

</div>

</nav>

</header>

<div class="container mt-4">

<main role="main" class="pb-3">

@RenderBody()

</main>

</div>

<footer class="border-top footer text-muted">

<div class="container">

&copy; LD - Phần mềm quản lý điều hành công việc

</div>

</footer>

<**script** src="~/lib/jquery/dist/jquery.min.js"></**script**>

<**script** src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></**script**>

<**script** **src**="~/js/site.js" **asp-append-version**="true"></**script**>

@await RenderSectionAsync("Scripts", required: false)

</body>

</html>

## **4.Program**

A screenshot of a computer program

Description automatically generated

Khởi tạo ứng dụng:

Tạo một builder cho ứng dụng web để cấu hình các dịch vụ cần thiết.

Đăng ký dịch vụ:

MVC: Cho phép sử dụng mô hình MVC để xử lý các yêu cầu HTTP.

Session: Cấu hình dịch vụ session để lưu trữ dữ liệu giữa các yêu cầu của người dùng.

Entity Framework: Đăng ký DbContext để kết nối và tương tác với cơ sở dữ liệu SQL Server.

Cấu hình pipeline xử lý yêu cầu:

Xử lý lỗi: Cấu hình xử lý lỗi cho môi trường không phải phát triển.

Chuyển hướng HTTPS: Đảm bảo tất cả yêu cầu được chuyển hướng sang HTTPS.

Phục vụ tệp tĩnh: Cho phép ứng dụng phục vụ các tệp như CSS, JS, và hình ảnh.

Định tuyến: Thiết lập khả năng định tuyến cho các yêu cầu HTTP đến các controller tương ứng.

Khởi động ứng dụng:

Chạy ứng dụng để bắt đầu lắng nghe các yêu cầu từ người dùng.

# **III. Phần mềm quản lý điều hành công việc.**

## **1.Đăng ký và đăng nhập tài khoản.**

### a.Đăng ký tài khoản:

A screenshot of a computer

Description automatically generatedMô tả: Người dùng nhập thông tin bao gồm tên đăng nhập, mật khẩu, họ tên, và email, sau đó nhấn "Đăng ký". Hệ thống sẽ kiểm tra xem tên đăng nhập có bị trùng với bất kỳ tài khoản nào khác không. Nếu tên đăng nhập không trùng, hệ thống sẽ kiểm tra định dạng email và tính hợp lệ. Người dùng cần nhập email chính xác để hoàn thành quá trình đăng ký.

### b.Đăng nhập:

A screenshot of a computer

Description automatically generatedMô tả: Sau khi đăng ký tài khoản thành công, hệ thống sẽ chuyển người dùng đến giao diện đăng nhập. Người dùng nhập tên đăng nhập và mật khẩu. Hệ thống kiểm tra tính hợp lệ của thông tin đăng nhập, đối chiếu với cơ sở dữ liệu. Nếu thông tin chính xác, người dùng sẽ được phép truy cập vào hệ thống. Nếu không, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại thông tin.

## 2. Giao diện phần mềm:

Mô tả: Giao diện sau khi đăng nhập thành công sẽ là màn hình chính của phần mềm, cung cấp các chức năng quản lý và điều hướng cho người dùng.

A screenshot of a computer

Description automatically generated

## 3.Các chức năng:

### **a.Văn bản đến:**

Mô tả: Người dùng sẽ thấy được thông tin danh sách các văn bản đến. Gồm số văn bản, ngày nhận, trích yếu, đơn vị gửi và nơi nhận. Ở mỗi văn bản đến đều có thể sửa, xóa các văn bản đến. Cũng như tạo văn bản đến mới.

A screenshot of a computer

Description automatically generated

#### **Thao tác sửa văn bản đến:**

Người dùng có thể sửa các thông tin của văn bản đến. Sau khi sửa xong thì nhấn vào cập nhật văn bản nếu không muốn sửa thì nhấn vào hủy để kết thúc quá trình sửa văn bản đến.

A screenshot of a computer

Description automatically generated

#### **Thao tác xóa văn bản đến:**

Mô tả: Người dùng có thể xóa văn bản đến bằng thao tác **xóa**. Nếu người dùng không muốn thực hiện thao tác **xóa**, họ có thể nhấn **Hủy** để thoát khỏi quá trình này.

A screenshot of a computer

Description automatically generated

### **b.Văn bản đi:**

Mô tả: Người dùng sẽ thấy được thông tin danh sách các văn bản đi. Gồm stt, số văn bản,ngày gửi, trích yếu, đơn vị nhận. Ở mỗi văn bản đi đều có thao tác sửa, xóa các văn bản đi. Cũng như tạo văn bản đi mới.

A screenshot of a computer

Description automatically generated

#### **Thao tác sửa văn bản đi:**

Người dùng có thể sửa các thông tin của văn bản đi. Sau khi sửa xong thì nhấn vào cập nhật văn bản nếu không muốn sửa thì nhấn vào hủy để kết thúc quá trình sửa văn bản đến.

A screenshot of a computer

Description automatically generated

#### **Thao tác xóa văn bản đi:**

Mô tả: Người dùng có thể xóa văn bản đi bằng thao tác **xóa**. Nếu người dùng không muốn thực hiện thao tác **xóa**, họ có thể nhấn **hủy** để thoát khỏi quá trình này.

A screenshot of a computer

Description automatically generated

### **c.Thống kê văn bản:**

#### **Thống Kê**

Mô tả: Chức năng này cho phép người dùng xem và quản lý danh sách văn bản đến và đi trong phần mềm. Người dùng có thể tìm kiếm các văn bản bằng từ khóa và xem thông tin chi tiết về từng văn bản trong các bảng được phân loại rõ ràng.

A screenshot of a computer

Description automatically generated

#### Tìm kiếm

* **Thanh tìm kiếm**: Người dùng có thể nhập từ khóa (ví dụ: đơn vị gửi, nơi nhận) để tìm kiếm nhanh các văn bản liên quan. Từ khóa có thể áp dụng cho cả văn bản đến và văn bản đi.
* **Ô nhập từ khóa**: Ô trống cho phép người dùng nhập bất kỳ từ khóa nào liên quan đến văn bản để thu hẹp phạm vi tìm kiếm và tìm kiếm nhanh chóng các tài liệu cần thiết.

**A screenshot of a computer

Description automatically generated**

#### Văn Bản Đến

Danh sách các văn bản đến từ các đơn vị, bao gồm thông tin chi tiết như sau:

* **Số Văn Bản**: Mã số duy nhất để nhận diện văn bản.
* **Ngày Nhận**: Ngày nhận văn bản từ đơn vị gửi.
* **Trích Yếu**: Mô tả ngắn gọn hoặc nội dung chính của văn bản.
* **Đơn Vị Gửi**: Tên phòng ban hoặc đơn vị đã gửi văn bản.
* **Nơi Nhận**: Phòng ban hoặc cá nhân nhận văn bản.

#### Văn Bản Đi

Danh sách các văn bản đi từ đơn vị người dùng đến các đơn vị khác, bao gồm:

* **Số Văn Bản**: Số của văn bản.
* **Ngày Gửi**: Ngày gửi văn bản.
* **Trích Yếu**: Mô tả ngắn về nội dung văn bản.
* **Đơn Vị Nhận**: Tên đơn vị hoặc phòng ban nhận văn bản.

Chức năng **Thống Kê Văn Bản** giúp người dùng dễ dàng theo dõi, quản lý, và tìm kiếm các văn bản đến và đi, hỗ trợ lưu trữ thông tin và đảm bảo tính nhất quán trong việc quản lý tài liệu của tổ chức.

### **d.Quản lý tài khoản:**

Mô tả: Chức năng này cho phép quản lý tài khoản người dùng trong hệ thống, hiển thị thông tin cơ bản của từng tài khoản và cung cấp các tùy chọn chỉnh **sửa** hoặc **xóa**.

**Bảng Thông Tin Tài Khoản**

**Tên đăng nhập**: Tên đăng nhập của người dùng, được sử dụng để truy cập vào hệ thống.

**Họ tên**: Họ và tên đầy đủ của người dùng.

**Email**: Địa chỉ email của người dùng, dùng cho việc liên lạc và khôi phục tài khoản nếu cần.

A screenshot of a computer

Description automatically generated

Thao tác: Cung cấp các tùy chọn quản lý tài khoản:

#### ***Sửa:***

Cho phép chỉnh sửa thông tin tài khoản, bao gồm mật khẩu,họ tên và email.

Sau khi sửa thông tin xong nhấn **lưu** để hoàn tất quá trình sửa đổi tài khoản.

A screenshot of a computer

Description automatically generated

#### ***Xóa:***

Xóa tài khoản khỏi hệ thống (hệ thống sẽ xác nhận lại một lần nữa để chắc chắn người dùng có muốn xóa tài khoản hay không), loại bỏ quyền truy cập của người dùng này. Nếu không muốn xóa tài khoản người dùng chỉ cần nhấn **hủy** để thoát quá trình xóa tài khoản.

A screenshot of a computer

Description automatically generated

### e.Chuyển Hướng

**Quay lại trang chính**: Liên kết để quay lại trang chính giúp người dùng dễ dàng điều hướng trở lại giao diện tổng quan của phần mềm.

A screenshot of a computer

Description automatically generated

### **f.Đăng xuất:**

Mô tả: Sau khi nhấn đăng xuất giao diện đăng nhập sẽ xuất hiện. Giúp người dùng dễ dàng thay đổi các tài khoản khác nhau.

# IV.Tài liệu tham khảo

* **ASP.NET Core 8.0 Documentation (Microsoft)**:

<https://learn.microsoft.com/vi-vn/aspnet/core/?view=aspnetcore-8.0>

* **Hướng dẫn lập trình ASP.NET Core MVC**:

<https://tedu.com.vn/series/hoc-aspnet-core-can-ban.html>

<https://xuanthulab.net/asp-net-core-mvc-tao-ung-dung-mvc-dau-tien-c-csharp.html>

<https://tuhocict.com/bai-giang-huong-dan-tu-hoc-lap-trinh-asp-net-core/>