

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ -
TIN HỌC THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN



**BÁO CÁO KẾT THÚC HỌC PHẦN
MÔN LẬP TRÌNH WEB NÂNG CAO**

**WEBSITE DỊCH VỤ TƯ VẤN KHÁM
BỆNH ONLINE BACSVOV**

Giảng viên hướng dẫn: **ThS. Lương Văn Minh**

Sinh viên thực hiện:

- | | |
|---------------------|------------|
| 1. Lương Tiên Đạt | 22DH114497 |
| 2. Nguyễn Lâm Phong | 22DH112739 |
| 3. Lê Văn Thiện | 22DH113465 |

Thành phố Hồ Chí Minh, tháng 08 năm 2025

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ -
TIN HỌC THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN



**BÁO CÁO KẾT THÚC HỌC PHẦN
MÔN LẬP TRÌNH WEB NÂNG CAO**

**WEBSITE DỊCH VỤ TƯ VẤN KHÁM
BỆNH ONLINE BACSIVOV**

Mã lớp học phần: **243123034408**

Năm học: **2024 – 2025**

Học kỳ: **3**

Sinh viên thực hiện:

- | | | |
|----|------------------|------------|
| 1. | Lương Tiến Đạt | 22DH114497 |
| 2. | Nguyễn Lâm Phong | 22DH112739 |
| 3. | Lê Văn Thiện | 22DH113465 |

Thành phố Hồ Chí Minh, tháng 08 năm 2025

TÓM TẮT ĐỀ TÀI

Đề tài "Xây dựng hệ thống đặt lịch khám VOV BÁC SĨ" là một ứng dụng web được phát triển nhằm hỗ trợ người dân có thể đặt lịch khám bệnh trực tuyến một cách dễ dàng, nhanh chóng và thuận tiện. Hệ thống được xây dựng trên nền tảng **ASP.NET Core** kết hợp với SQL Server để lưu trữ dữ liệu và đảm bảo hiệu năng hoạt động ổn định.

Ứng dụng chia thành ba nhóm người dùng chính: người bệnh (bệnh nhân), bác sĩ, và quản trị viên. Người bệnh có thể đăng ký tài khoản thông qua mã OTP (Firebase), tra cứu danh sách bác sĩ, lọc theo chuyên khoa, xem chi tiết hồ sơ và thực hiện đặt lịch khám. Bác sĩ có thể quản lý lịch khám, cập nhật thông tin trạng thái cuộc hẹn, xem thông tin bệnh nhân và đưa ra ghi chú chuyên môn. Quản trị viên có quyền quản lý toàn bộ hệ thống: thêm/sửa/xóa thông tin bác sĩ, bài viết chuyên môn, danh sách chuyên khoa, lịch khám và trạng thái khám.

Ngoài ra, hệ thống còn được tích hợp trình soạn thảo văn bản CKEditor nhằm giúp tạo nội dung bài viết về sức khỏe dễ đọc và thẩm mỹ hơn. Giao diện được thiết kế hiện đại, mô phỏng theo phong cách của các hệ thống y tế chuyên nghiệp như Vinmec, hướng tới trải nghiệm người dùng thân thiện và dễ sử dụng.

Đề tài đã hoàn thiện hầu hết các chức năng cơ bản và mở rộng cần thiết cho một hệ thống quản lý lịch khám, đồng thời đặt nền móng để tiếp tục phát triển trong tương lai như: tích hợp thanh toán trực tuyến, phản hồi đánh giá sau khám, nâng cấp giao diện tương tác và cải thiện hiệu năng tổng thể.

MỤC LỤC

DANH MỤC HÌNH	xi
DANH MỤC BẢNG	xiii
CHƯƠNG 1: Mở đầu	1
1.1. Giới thiệu chung về đề tài	1
1.1.1. Giới thiệu tổng quan về doanh nghiệp	1
1.1.2. Mô tả về đối tác	1
1.1.3. Tổng quan về quy trình nghiệp vụ	1
1.2. Lý do chọn đề tài	2
1.3. Lý do chọn nền tảng ASP.NET Core	3
1.4. Cấu trúc báo cáo	3
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	5
2.1. Giới thiệu ASP.NET Core	5
2.1.1. Application Frameworks	5
2.1.2. Utility Framework	5
2.1.3. ASP.NET Core Platform	5
2.1.4. Giới thiệu Code Editor	6
2.1.5. Tạo ASP.NET Core Project	6
2.1.6. Chạy chương trình	6
2.1.7. Hiểu về Endpoints, Routes và HTML Rendering	6
2.2. ASP.NET Core Application	7
2.3. ASP.NET Core Platform	7
2.3.1. Kiên trúc ASP.NET Core Platform	7
2.3.2. Middleware	7
2.4. Controller và Views trong ASP.NET Core	8
2.4.1. Khái niệm Views:	8
2.4.2. Razor Views	8
2.4.3. Razor Syntax	8
2.5. Controller và Views (Phần 2)	8
2.5.1. ViewBag	8
2.5.2. TempData	8

2.5.3.	Partial View.....	8
2.5.4.	Content-Encoding	8
2.6.	Tag Helper.....	9
2.6.1.	Các cài đặt cần thiết để sử dụng Tag Helper.....	9
2.6.2.	Giới thiệu Tag Helper cơ bản.....	10
2.6.3.	Áp dụng Tag Helper sẵn có.....	11
2.7.	Model Binding	11
2.7.1.	Khái niệm Model Binding.....	11
2.7.2.	Data Type	11
2.7.3.	Complex Type.....	11
2.8.	Docker	11
2.9.	Các khái niệm cơ bản về Blazor.....	12
2.9.1.	Blazor Server.....	12
2.9.2.	Blazor WebAssembly	12
2.9.3.	Giới thiệu ASP.NET Core Identity	12
CHƯƠNG 3: ĐẶC TẢ CHỨC NĂNG		13
3.1.	Trình bày sơ đồ tổng quát hệ thống	13
3.1.1.	Sơ đồ tổng quát	13
3.1.2.	Mô tả các actor	13
3.2.	Mô tả cụ thể từng usecase	14
3.2.1.	Sơ đồ chức năng của người dùng	14
3.2.2.	Sơ đồ chức năng của bệnh nhân.....	15
3.2.3.	Sơ đồ chức năng của bác sĩ	16
3.2.4.	Sơ đồ quản lý	17
3.3.	Mô tả về từng quy trình nghiệp vụ.....	17
3.3.1.	UC01 [Xem thông báo].....	21
3.3.2.	UC02 [Tìm kiếm thông tin]	21
3.3.3.	UC03 [Lọc thông tin].....	22
3.3.4.	UC08 [Sắp xếp danh sách bác sĩ].....	22
3.3.5.	UC09 [Đăng nhập]	23
3.3.6.	UC [Xem lại cuộc gọi]	24

3.3.7. UC12 [Quản lý thông tin cá nhân]	25
3.3.8. UC14 [Cập nhật thông tin cá nhân]	25
3.3.9. UC15 [Xóa tài khoản].....	26
3.3.10. UC16 [Quản lý giao dịch]	27
3.3.11. UC17 [Nạp tiền vào tài khoản]	27
3.3.12. UC18 [Rút tiền về ngân hàng]	29
3.3.13. UC21 [Đăng ký khám bệnh]	30
3.3.14. UC22 [Nhận cuộc gọi từ bác sĩ].....	31
3.3.15. UC23 [Quản lý đánh giá bác sĩ].....	32
3.3.16. UC24 [Áp dụng khuyến mãi].....	32
3.3.17. UC25 [Đăng ký tài khoản]	33
3.3.18. UC26 [Quản lý lịch hẹn]	34
3.3.19. UC27 [Thay đổi lịch khám]	35
3.3.20. UC28 [Hủy lịch khám].....	36
3.3.21. UC29 [Quản lý thông báo].....	37
3.3.22. UC31 [Xóa thông báo].....	37
3.3.23. UC32 [Đăng ký bác sĩ].....	38
3.3.24. UC33 [Quản lý hồ sơ bệnh nhân]	39
3.3.25. UC34 [Sửa đổi thông tin bệnh nhân]	40
3.3.26. UC35 [Quản lý lịch khám].....	40
3.3.27. UC38 [Hủy lịch khám]	41
3.3.28. UC39 [Chấp nhận lịch khám]	42
3.3.29. UC41 [Gọi điện với bệnh nhân].....	43
3.3.30. UC42 [Quản lý thông báo].....	43
3.3.31. UC43 [Xem chi tiết thông báo]	44
3.3.32. UC46 [Quản lý tài khoản người dùng]	45
3.3.33. UC48 [Sửa thông tin tài khoản]	45
3.3.34. UC49 [Khóa tài khoản]	46
3.3.35. UC50 [Xóa tài khoản]	46
3.3.36. UC51 [Xác nhận đăng ký bác sĩ]	47
3.3.37. UC52 [Quản lý hồ sơ y tế]	48

3.3.38. UC57 [Quản lý lịch khám].....	48
3.3.39. UC59 [Quản lý lương]	49
3.3.40. UC61 [Thêm lương].....	49
3.3.41. UC63 [Sửa lương]	50
3.3.42. UC64 [Quản lý thanh toán].....	51
3.3.43. UC65 [Hoàn phí khám bệnh]	51
3.3.44. UC70 [Quản lý thông tin bệnh viện].....	52
3.3.45. UC74 [Quản lý danh mục bài viết]	52
3.3.46. UC75 [Quản lý bài viết]	53
3.3.47. UC76 [Thêm bài viết]	54
3.3.48. UC79 [Quản lý danh mục chuyên ngành].....	55
3.3.49. UC83 [Quản lý danh mục khoa bệnh]	56
3.3.50. UC87 [Quản lý lịch sử cuộc gọi]	56
3.3.51. UC90 [Quản lý đánh giá phản hồi bệnh nhân]	57
CHƯƠNG 4: MÔ HÌNH HÓA YÊU CẦU VÀ API.....	58
4.1. Thiết kế ERD tổng quát của hệ thống	58
4.2. Triển khai CSDL trên MSSQL Server	58
4.3. Nhập dữ liệu mẫu	69
4.4. Mô tả các thực thể	76
4.4.1. Thực thể [PHANQUYEN].....	76
4.4.2. Thực thể [NGUOIDUNG]	76
4.4.3. Thực thể [BENHVIEN]	77
4.4.4. Thực thể [KHOABENH]	77
4.4.5. Thực thể [CHUYENNGANH].....	77
4.4.6. Thực thể [BACSI]	78
4.4.7. Thực thể [TRANGTHAICUOCHEM]	79
4.4.8. Thực thể [HOSO]	80
4.4.9. Thực thể [HINHANHBENH]	80
4.4.10. Thực thể [KHUYENMAI]	80
4.4.11. Thực thể [CUOCHENKHAM]	81
4.4.12. Thực thể [DANHGIA]	82

4.4.13. Thực thể [THANHTOAN].....	82
4.4.14. Thực thể [THONGBAO]	83
4.4.15. Thực thể [HOATDONG]	83
4.4.16. Thực thể [LICHSUHD]	83
4.4.17. Thực thể [LOAIBAIVIET]	84
4.4.18. Thực thể [BAIVIET].....	84
4.4.19. Thực thể [TRANGTHAIBACSI].....	84
4.5. Cài các Store Procedure	85
4.5.1. Đăng ký người dùng.....	85
4.5.2. Kiểm tra đăng nhập	86
4.5.3. Lấy thông tin của bài viết theo mã	87
4.5.4. Lấy thông tin bài viết theo mã khoa bệnh.....	89
4.5.5. Lấy thông tin chi tiết bác sĩ.....	90
4.5.6. Lọc bác sĩ theo nhiều tiêu chí.....	90
4.5.7. Lấy tất cả bình luận bằng id Bác sĩ	91
4.5.8. Lưu hồ sơ bệnh nhân	92
4.5.9. Lưu hình ảnh bệnh	93
4.5.10. Lưu cuộc hẹn khám.....	94
4.5.11. Lấy thông tin cuộc hẹn.....	95
4.5.12. Xóa cuộc hẹn.....	96
4.5.13. Nạp tiền vào tài khoản	97
4.5.14. Đè xuất top 5 bác sĩ có lịch khám nhiều nhất	98
4.5.15. Lấy danh sách thông báo theo mã nguồn	99
4.5.16. Lấy danh sách hồ sơ bệnh	100
4.6. Wireframe.....	102
4.7. Sơ đồ tuần tự (Sequence Diagram)	112
4.7.1. Sơ đồ tuần tự [Cập nhật thông tin cá nhân]	112
4.7.2. Sơ đồ tuần tự [Đăng ký tài khoản]	112
4.7.3. Sơ đồ tuần tự [Đăng nhập]	113
4.7.4. Sơ đồ tuần tự [Đăng ký bác sĩ].....	114
4.7.5. Sơ đồ tuần tự [Đăng ký lịch khám].....	115

4.7.6.	Sơ đồ tuần tự [Hủy lịch khám].....	115
4.7.7.	Sơ đồ tuần tự [Nạp tiền].....	116
4.7.8.	Sơ đồ tuần tự [Sửa bài viết]	117
4.7.9.	Sơ đồ tuần tự [Sửa khoa bệnh].....	118
4.7.10.	Sơ đồ tuần tự [Thêm bài viết]	118
4.7.11.	Sơ đồ tuần tự [Thêm Khoa bệnh].....	118
4.7.12.	Sơ đồ tuần tự [Xác nhận đăng ký bác sĩ]	119
4.7.13.	Sơ đồ tuần tự [Xóa bài viết]	119
4.7.14.	Sơ đồ tuần tự [xóa khoa bệnh]	120
4.8.	Class Diagram	121
4.8.1.	Tổng quan Kiến trúc	122
4.8.2.	Vai trò của từng nhóm lớp chính	123
4.8.3.	Mối quan hệ giữa các Class	124
4.8.4.	Kết luận	125
4.9.	Các API sử dụng trong chương trình	125
4.9.1.	Internal APIs (API nội bộ)	125
4.9.2.	External APIs (API bên ngoài)	128
4.9.3.	Bảo mật, hiệu suất và xử lý lỗi.....	128
CHƯƠNG 5:	THIẾT KẾ CHƯƠNG TRÌNH	130
5.1.	Cấu hình chương trình.....	130
5.1.1.	File appsettings.json:.....	130
5.1.2.	File launchSettings.json:	130
5.1.3.	File Program.cs:	132
5.1.4.	File DataModel.cs:	134
5.1.5.	File ErrorViewModel.cs:	136
5.2.	Chức năng chương trình.....	137
5.2.1.	Chức năng[Xem thông báo]	137
5.2.2.	Chức năng[Lọc thông tin]	138
5.2.3.	Chức năng[Xem danh sách bệnh viện]	139
5.2.4.	Chức năng [Xem danh sách bác sĩ].....	141
5.2.5.	Chức năng[Xem chi tiết bác sĩ].....	143

5.2.6. Chức năng [Xem đánh giá bác sĩ]	145
5.2.7. Chức năng[Đăng nhập]	146
5.2.8. Chức năng[Xem lịch sử khám bệnh]	147
5.2.9. Chức năng[Cập nhật thông tin cá nhân].....	148
5.2.10. Chức năng [Quản lý giao dịch]	157
5.2.11. Chức năng [Nạp tiền vào tài khoản]	161
5.2.12. Chức năng [Xem bài viết]	162
5.2.13. Chức năng [Đăng ký khám bệnh]	167
5.2.14. Chức năng [Đăng ký tài khoản]	179
5.2.15. Chức năng [Quản lý lịch hẹn khám]	186
5.2.16. Chức năng [Hủy lịch khám]	190
5.2.17. Chức năng[Đăng ký bác sĩ].....	191
5.2.18. Chức năng [Quản lí hồ sơ bệnh nhân]	216
5.2.19. Chức năng[Quản lí lịch khám].....	222
5.2.20. Chức năng [Xác nhận lịch khám]	225
5.2.21. Chức năng [Xác nhận đã hoàn thành]	228
5.2.22. Chức năng [Xác nhận đã bị hủy]	230
5.2.23. Chức năng [Cập nhật]	232
5.2.24. Chức năng [Quản lý thông báo]	235
5.2.25. Chức năng [Thông kê]	243
5.2.26. Chức năng [Quản lý tài khoản]	244
5.2.27. Chức năng [Quản lý bài viết]	245
5.2.28. Chức năng [Quản lý chuyên ngành]	250
5.2.29. Chức năng[Quản lý khoa bệnh]	254
5.2.30. Chức năng [Quản lý bệnh viện]	257
5.2.31. Chức năng [Quản lý bệnh nhân]	262
5.2.32. Chức năng [Quản lý bác sĩ].....	264
5.2.33. Chức năng [Quản lý lịch khám]	268
5.2.34. Chức năng [Quản lý đánh giá]	269
5.2.35. Chức năng [Thông kê]	270
5.2.36. Chức năng [Quản lý thông báo].....	272

5.2.37. Chức năng [Quản lý thanh toán]	274
CHƯƠNG 6: CHƯƠNG TRÌNH THỰC NGHIỆM	279
6.1. Chi tiết các giao diện người dùng	279
6.1.1. Giao diện [trang chủ]	279
6.1.2. Giao diện [danh sách bài viết].....	281
6.1.3. Giao diện [bài viết]	281
6.1.4. Giao diện [Lọc bác sĩ].....	282
6.1.5. Giao diện [danh sách bác sĩ]	283
6.1.6. Giao diện [chi tiết bác sĩ]	283
6.1.7. Giao diện [đăng ký tài khoản].....	284
6.1.8. Giao diện [đăng nhập].....	284
6.1.9. Giao diện [cập nhật thông tin cá nhân]	285
6.1.10. Giao diện [Quên mật khẩu].....	285
6.1.11. Giao diện [Đổi mật khẩu].....	286
6.1.12. Giao diện [nạp tiền].....	286
6.1.13. Giao diện [đăng ký lịch khám].	287
6.1.14. Giao diện [lịch sử khám].....	288
6.1.15. Giao diện [hủy lịch khám]	288
6.1.16. Giao diện [Dự đoán bệnh].....	289
6.2. Chi tiết các giao diện bác sĩ	290
6.2.1. Giao diện [đăng nhập].....	290
6.2.2. Giao diện [đăng ký bác sĩ]	290
6.2.3. Giao diện [Trang chủ/ thông kê]	294
6.2.4. Giao diện [Hồ sơ bệnh nhân]	297
6.2.5. Giao diện [lịch hẹn khám].....	297
6.2.6. Giao diện [đã hoàn thành khám]	298
6.2.7. Giao diện [đã hủy khám].....	298
6.2.8. Giao diện [thông báo]	299
6.3. Chi tiết các giao diện admin	299
6.3.1. Giao diện [sửa bài viết]	299
6.3.2. Giao diện [sửa khoa bệnh]	300

6.3.3.	Giao diện [thêm bài viết]	300
6.3.4.	Giao diện [thêm khoa bệnh]	301
6.3.5.	Giao diện [xác nhận đăng ký bác sĩ]	301
6.3.6.	Giao diện [xoá bài viết].....	302
6.3.7.	Giao diện [xoá khoa bệnh].....	302
6.3.8.	Giao dien [đánh giá phản hồi].....	303
6.3.9.	Giao diện [quản lý danh mục].....	303
6.3.10.	Giao diện [danh sách lịch khám].....	304
6.3.11.	Giao diện [quản lý thanh toán].....	304
6.3.12.	Giao diện [quản lý thông báo].....	305
6.3.13.	Giao diện [thống kê]	305
CHƯƠNG 7:	KẾT LUẬN	307
7.1.	Kết quả đạt được và chưa đạt được	307
7.1.1.	Kết quả đạt được	307
7.1.2.	Kết quả chưa đạt được	307
7.2.	Hướng phát triển mở rộng ứng dụng trong tương lai	307

DANH MỤC HÌNH

Hình 2.1: Sơ đồ UC tổng quát.....	13
Hình 2.2: Sơ đồ chức năng người dùng	14
Hình 2.3: Sơ đồ chức năng bệnh nhân	15
Hình 2.4: Sơ đồ chức năng bác sĩ	16
Hình 2.5: Sơ đồ chức năng quản lý	17
Hình 2.6: Activity Diagram [Đăng nhập]	24
Hình 2.7: Activity Diagram [Cập nhật thông tin cá nhân].....	26
Hình 2.8: Activity Diagram [Nạp tiền]	29
Hình 2.9: Activity Diagram [Đăng ký khám bệnh]	31
Hình 2.10: Activity Diagram [Đăng ký tài khoản]	34
Hình 2.11: Activity Diagram [Hủy lịch khám]	37
Hình 2.12: Activity Diagram [Đăng ký bác sĩ].....	39
Hình 2.13: Activity Diagram [Chấp nhận lịch khám].....	42
Hình 2.14: Activity Diagram [Gửi thông báo].....	44
Hình 2.15: Activity Diagram [Xác nhận đăng ký bác sĩ]	48
Hình 2.16: Activity Diagram [Thêm bài viết]	55
Hình 2.17: Sequence Diagram [Cập nhật thông tin cá nhân]	112
Hình 2.18: Sequence Diagram [Đăng ký tài khoản]	113
Hình 2.19: Sequence Diagram [Đăng nhập]	114
Hình 2.20: Sequence Diagram [Đăng ký bác sĩ].....	114
Hình 2.21: Sequence Diagram [Đăng ký lịch khám]	115
Hình 2.22: Sequence Diagram [Hủy lịch khám]	116
Hình 2.23: Sequence Diagram [Nạp tiền]	117
Hình 2.24: Sequence Diagram [Sửa bài viết]	117
Hình 2.25: Sequence Diagram [Sửa khoa bệnh].....	118
Hình 2.26: Sequence Diagram [Thêm bài viết]	118
Hình 2.27: Sequence Diagram [Thêm khoa bệnh].....	119
Hình 2.28: Sequence Diagram [Xác nhận đăng ký bác sĩ]	119
Hình 2.29: Sequence Diagram [Xóa bài viết]	120
Hình 2.30: Sequence Diagram [Xóa khoa bệnh]	120

Hình 3.1: Sơ đồ ERD 58

DANH MỤC BẢNG

Bảng 2.1: Danh sách các Actor tham gia sử dụng hệ thống	13
Bảng 2.2: Bảng Usecase.....	17

CHƯƠNG 1: MỞ ĐẦU

1.1. Giới thiệu chung về đề tài

1.1.1. Giới thiệu tổng quan về doanh nghiệp

Hệ thống quản lý bác sĩ VOVBacSi nhằm tạo ra một nền tảng chăm sóc sức khỏe trực tuyến với đầy đủ các chức năng như đặt lịch khám, tư vấn y tế từ xa, quản lý hồ sơ bệnh nhân và cung cấp thông tin y khoa. Trang web sẽ giúp kết nối bệnh nhân với bác sĩ một cách dễ dàng, an toàn và tiện lợi. Trong bối cảnh công nghệ thông tin phát triển mạnh mẽ, việc áp dụng hệ thống quản lý này đã trở thành yếu tố quan trọng để nâng cao chất lượng dịch vụ chăm sóc sức khỏe, mang đến sự tiện lợi cho bệnh nhân.

1.1.2. Mô tả về đối tác

Hệ thống VOVBacSi giúp quản lý thông tin bệnh nhân một cách chính xác và dễ dàng, đồng thời tích hợp các tính năng như đặt lịch hẹn trực tuyến, quản lý tài chính và theo dõi lịch sử điều trị. Điều này cho phép các bác sĩ và nhân viên y tế tập trung nhiều hơn vào việc chăm sóc bệnh nhân, từ chẩn đoán đến điều trị và theo dõi sức khỏe.

Một ưu điểm nổi bật của hệ thống là khả năng tăng cường tương tác giữa bác sĩ và bệnh nhân thông qua các hồ sơ y tế điện tử, giúp bệnh nhân dễ dàng truy cập thông tin cá nhân, đặt hẹn và nhận tư vấn từ bác sĩ thông qua chức năng video call một cách thuận tiện. Hệ thống cũng hỗ trợ tiêu chuẩn hóa quản lý thông tin y tế, giúp các phòng khám dễ dàng kết nối với các mạng lưới chăm sóc sức khỏe lớn hơn, nâng cao tính chuyên nghiệp và hiệu quả hoạt động.

1.1.3. Tổng quan về quy trình nghiệp vụ

Hệ thống cung cấp dịch vụ tư vấn khám chữa bệnh phát triển một hệ thống quản lý bác sĩ trực tuyến nhằm cung cấp nền tảng chăm sóc sức khỏe toàn diện, kết nối dễ dàng giữa bệnh nhân và bác sĩ. Hệ thống hỗ trợ quản lý thông tin bệnh nhân, đặt lịch hẹn trực tuyến, tư vấn y tế từ xa qua video call, và theo dõi lịch sử điều trị. Điều này giúp bác sĩ tối ưu hóa quy trình làm việc, đồng thời cải thiện trải nghiệm của bệnh nhân nhờ tính tiện lợi và an toàn.

Ngoài ra, dự án sẽ tích hợp một số nghiệp vụ quan trọng như:

- Quản lý lịch hẹn: Cho phép bệnh nhân đặt lịch khám trực tuyến dựa trên thời gian làm việc của bác sĩ.
 - Nghiệp vụ tư vấn từ xa: Bác sĩ có thể tư vấn trực tiếp cho bệnh nhân thông qua tính năng video call. Sau mỗi buổi tư vấn, bác sĩ có thể ghi nhận chẩn đoán và điều trị vào hồ sơ bệnh án điện tử của bệnh nhân, giúp dễ dàng theo dõi tình trạng sức khỏe về sau.
 - Quản lý thanh toán và hóa đơn: Hệ thống sẽ tích hợp tính năng quản lý tài chính, cho phép bệnh nhân thanh toán chi phí dịch vụ tư vấn trực tuyến, khám bệnh, hoặc mua thuốc qua các cổng thanh toán trực tuyến. Sau khi hoàn tất giao dịch, hệ thống sẽ tự động phát hành hóa đơn điện tử cho bệnh nhân.
 - Nghiệp vụ quản lý thông tin bệnh án: Bác sĩ có thể truy cập vào hồ sơ bệnh án điện tử của bệnh nhân, cập nhật kết quả chẩn đoán, chỉ định điều trị và theo dõi quá trình hồi phục. Bệnh nhân cũng có thể xem lại hồ sơ và kết quả khám của mình bất kỳ lúc nào.
 - Quản lý đánh giá và phản hồi: Bệnh nhân có thể đánh giá chất lượng dịch vụ và đưa ra phản hồi sau mỗi buổi tư vấn hoặc khám bệnh. Điều này giúp hệ thống cải thiện dịch vụ và hỗ trợ bác sĩ điều chỉnh quy trình làm việc để phục vụ tốt hơn. Bên cạnh đó, giao diện của hệ thống sẽ được thiết kế lại theo phong cách hiện đại, chuyên nghiệp và thân thiện với người dùng, tương tự như vinmec.com, nhằm nâng cao trải nghiệm người dùng và tăng tính chuyên nghiệp cho dịch vụ.
- Dự án không chỉ giúp nâng cao chất lượng chăm sóc sức khỏe mà còn tối ưu hóa quy trình quản lý và tăng cường hiệu quả kinh doanh cho các bác sĩ, đồng thời mở rộng phạm vi chăm sóc bệnh nhân mà không bị giới hạn về địa lý.

1.2. Lý do chọn đề tài

Hiện nay, việc đặt lịch khám bệnh truyền thống còn tồn tại nhiều bất cập như mất thời gian chờ đợi, quy trình rườm rà và chưa tối ưu cho bệnh nhân ở xa hoặc người không có thời gian đi lại. Với tình hình dịch bệnh như COVID-19 vừa qua, nhu cầu khám chữa bệnh từ xa lại càng trở nên cần thiết.

Đề tài này giúp sinh viên tiếp cận mô hình hệ thống thực tế, áp dụng những kiến thức đã học như lập trình web, thiết kế cơ sở dữ liệu, bảo mật và tối ưu hóa trải nghiệm

người dùng vào một sản phẩm có tính ứng dụng cao. Ngoài ra, đề tài còn mang ý nghĩa xã hội khi góp phần nâng cao chất lượng dịch vụ y tế thông qua công nghệ.

1.3. Lý do chọn nền tảng ASP.NET Core

ASP.NET Core là một nền tảng lập trình web hiện đại, mã nguồn mở, đa nền tảng và được Microsoft hỗ trợ lâu dài. Việc chọn ASP.NET Core cho đề tài dựa trên các lý do sau:

- **Hiệu suất cao, nhẹ, dễ triển khai** trên các nền tảng cloud như Azure, AWS.
- **Hỗ trợ kiến trúc MVC** giúp tách biệt rõ ràng giữa giao diện, xử lý logic và dữ liệu.
- **Tích hợp tốt với các công nghệ hiện đại** như Entity Framework Core, Identity, Razor Pages.
- **Bảo mật mạnh mẽ** với nhiều tùy chọn xác thực, phân quyền.
- **Cộng đồng hỗ trợ lớn** và có tài liệu hướng dẫn chi tiết.

Ngoài ra, nhóm đã có kiến thức nền tảng về C# và ASP.NET Core từ các học phần trước, nên việc chọn nền tảng này giúp rút ngắn thời gian học công nghệ mới mà vẫn đảm bảo chất lượng sản phẩm.

1.4. Cấu trúc báo cáo

. Báo cáo được chia thành 7 chương với nội dung cụ thể như sau:

- **Chương 1: Mở đầu** Giới thiệu tổng quan về đề tài, mục tiêu, lý do chọn đề tài và nền tảng công nghệ ASP.NET Core. Đồng thời trình bày cấu trúc của báo cáo.
- **Chương 2: Cơ sở lý thuyết** Trình bày các kiến thức nền tảng về ASP.NET Core, Docker, Blazor, và các công nghệ liên quan được sử dụng trong dự án (Middleware, MVC, Razor Syntax, Tag Helpers, v.v.).
- **Chương 3: Đặc tả chức năng** Mô tả chi tiết các yêu cầu chức năng của hệ thống thông qua sơ đồ use case, phân tích actor, và mô tả từng quy trình nghiệp vụ.
- **Chương 4: Mô hình hóa yêu cầu và API** Thiết kế cơ sở dữ liệu (ERD), triển khai CSDL trên SQL Server, mô tả các thực thể, store procedure, wireframe, sơ đồ tuần tự (sequence diagram), sơ đồ lớp (class diagram), và danh sách API.
- **Chương 5: Thiết kế chương trình** Trình bày cấu hình hệ thống (appsettings, launchSettings), cấu trúc mã nguồn, và triển khai chi tiết từng chức năng theo phân hệ người dùng (bệnh nhân, bác sĩ, quản trị).

- **Chương 6: Chương trình thực nghiệm** Minh họa giao diện người dùng, bác sĩ và quản trị viên, kèm theo giải thích hoạt động của các chức năng chính.
- **Chương 7: Kết luận** Tổng kết kết quả đạt được, những hạn chế và đề xuất hướng phát triển mở rộng trong tương lai.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu ASP.NET Core

ASP.NET Core là một nền tảng mã nguồn mở, đa nền tảng được Microsoft phát triển, phục vụ việc xây dựng các ứng dụng web hiện đại, hiệu suất cao. Đây là sự kế thừa và cải tiến mạnh mẽ từ ASP.NET truyền thống.

2.1.1. Application Frameworks

- Khái niệm Framework trong phát triển ứng dụng: Framework là một bộ công cụ hỗ trợ lập trình viên phát triển phần mềm hiệu quả, bao gồm các thư viện, công cụ, cấu trúc sẵn có để xử lý nhiều tác vụ phổ biến như routing, security, UI rendering,...
- So sánh một số Framework phổ biến:

Tiêu chí	ASP.NET Core	Laravel (PHP)	Django (Python)
Ngôn ngữ chính	C#	PHP	Python
Hiệu suất	Rất cao	Tốt	Tốt
Đa nền tảng	Có (Windows/Linux)	Có	Có
Kiến trúc hỗ trợ	MVC, Razor, Blazor	MVC	MTV
Hỗ trợ Microsoft	Có	Không	Không

2.1.2. Utility Framework

- ASP.NET Core hỗ trợ nhiều thư viện tiện ích như:
 - Entity Framework Core:** ORM dùng để thao tác CSDL qua mô hình đối tượng.
 - SignalR:** Hỗ trợ real-time communication.
 - Serilog / NLog:** Ghi log hiệu quả.
 - FluentValidation:** Kiểm tra dữ liệu đầu vào.

2.1.3. ASP.NET Core Platform

- Lịch sử phát triển ASP.NET Core: Ra mắt vào năm 2016, ASP.NET Core là sự thay thế cho ASP.NET Web Forms và MVC trước đó. Phiên bản mới nhất tích hợp nhiều công nghệ mới như Razor Pages, Blazor.
- Ưu điểm và đặc trưng nổi bật của ASP.NET Core:
 - Mã nguồn mở và cập nhật thường xuyên.
 - Hiệu năng cao, dễ triển khai cloud.

- Hỗ trợ Dependency Injection, bảo mật mạnh.
- Hoạt động trên Windows, macOS, Linux.

2.1.4. Giới thiệu Code Editor

- Visual Studio : Công cụ IDE đầy đủ tính năng, hỗ trợ quản lý project, thiết kế UI, debug.
- Visual Studio Code : Gọn nhẹ, linh hoạt
- Các extension hỗ trợ ASP.NET Core :
 - C# (OmniSharp)
 - ASP.NET Helper
 - Razor Syntax Highlighting

2.1.5. Tạo ASP.NET Core Project

- Hướng dẫn tạo project mẫu: Mở Visual Studio → Chọn “ASP.NET Core Web App (Model-View-Controller)” → .NET 6/7
- Cấu trúc thư mục ban đầu:
 - Controllers/
 - Views/
 - Models/
 - wwwroot/
 - appsettings.json
 - Program.cs

2.1.6. Chạy chương trình

- Cách chạy ứng dụng đầu tiên : Chọn “Run” hoặc nhấn Ctrl+F5, trình duyệt sẽ mở đường dẫn mặc định
- Giao diện mặc định và kiểm tra hoạt động : Giao diện ban đầu là template mặc định của ASP.NET Core MVC, xác nhận cấu hình đã hoạt động.

2.1.7. Hiểu về Endpoints, Routes và HTML Rendering

- **Routing:** Xác định URL tương ứng với controller/action.
- **Endpoints:** Các điểm kết nối HTTP mà hệ thống nhận request và trả response.
- **HTML Rendering:** Dữ liệu được hiển thị lên View thông qua Razor Engine.

2.2. ASP.NET Core Application

Là ứng dụng web chạy trên nền tảng ASP.NET Core, sử dụng kiến trúc MVC hoặc Razor Pages. Các thành phần cơ bản gồm:

- Model: dữ liệu và nghiệp vụ.
- View: giao diện người dùng.
- Controller: xử lý logic và điều hướng.

2.3. ASP.NET Core Platform

2.3.1. Kiến trúc ASP.NET Core Platform

Request Pipeline & Hosting:

- Request đi qua Middleware (Pipeline) → Controller xử lý → trả Response.
- Startup.cs hoặc Program.cs cấu hình pipeline và services.

2.3.2. Middleware

- Khái niệm Middleware: Là các thành phần trung gian xử lý request/response. Ví dụ: Authentication, Routing, Logging,...
- Cách tạo và sử dụng Middleware:

```
app.Use(async (context, next) =>
{
    // custom logic
    await next();
});
```

- Development Tools và C# Features

Development Tools (hay còn gọi là DevTools) là các công cụ và phần mềm được các nhà phát triển sử dụng để tạo, gỡ lỗi, kiểm tra và duy trì phần mềm và ứng dụng. Các công cụ này có thể được tích hợp sẵn trong trình duyệt web, hoặc là các ứng dụng độc lập.

C# (đọc là "C sharp") là một ngôn ngữ lập trình hiện đại, hướng đối tượng và an toàn về kiểu dữ liệu được phát triển bởi Microsoft. Nó được thiết kế để xây dựng nhiều loại ứng dụng chạy trên .NET.

- Các công cụ hỗ trợ phát triển
 - Swagger
 - Postman

- EF Core Tools
- SQL Server Management Studio
- Những tính năng C# được sử dụng trong project (nullable, async/await, LINQ...)
- async/await xử lý bất đồng bộ.
- LINQ truy vấn dữ liệu.
- Nullable cải thiện null safety.

2.4. Controller và Views trong ASP.NET Core

2.4.1. Khái niệm Views:

Trong MVC, View hiển thị dữ liệu cho người dùng.

2.4.2. Razor Views

- Giới thiệu Razor View Engine : Cho phép kết hợp C# vào HTML dễ dàng.
- Ưu điểm của Razor : Ngắn gọn, dễ đọc., Hiệu suất cao, Dễ bảo trì.

2.4.3. Razor Syntax

- Các cú pháp phổ biến:

vòng lặp : @foreach(var item in Model) { ... }

điều kiện: @if(condition) { ... }

render biến: @Model.Name

2.5. Controller và Views (Phần 2)

2.5.1. ViewBag

- Truyền dữ liệu từ Controller đến View: ViewBag.Name = "Thiện";

2.5.2. TempData

- Giữ dữ liệu tạm thời giữa các request: TempData["msg"] = "Đăng ký thành công";

2.5.3. Partial View

- Chia nhỏ UI thành các phần có thể tái sử dụng như Header, Footer

2.5.4. Content-Encoding

- Xử lý định dạng dữ liệu trả về:

Khi một ứng dụng client (ví dụ: trình duyệt hoặc ứng dụng di động) gửi yêu cầu đến một máy chủ hoặc API, máy chủ sẽ trả về dữ liệu. Dữ liệu này có thể ở nhiều định dạng khác nhau. Việc xử lý định dạng dữ liệu trả về là quá trình client diễn giải và sử dụng dữ liệu đó.

Các định dạng phổ biến:

- **JSON (JavaScript Object Notation):** Đây là định dạng phổ biến nhất cho các API web. Nó có cấu trúc văn bản dễ đọc cho người và dễ phân tích cho máy.
- **XML (eXtensible Markup Language):** Một định dạng khác dựa trên thẻ, tương tự như HTML. Nó từng rất phổ biến nhưng hiện nay ít được sử dụng hơn JSON cho các API mới.
- **HTML (HyperText Markup Language):** Trình duyệt nhận và hiển thị nội dung trang web dưới dạng HTML.
- **Dữ liệu dạng văn bản (Plain Text):** Dữ liệu đơn giản không có cấu trúc.
- Phân biệt Razor Pages và MVC

Razor Pages:

- **Mô hình:** Tập trung vào trang (Page-centric).
- **Cấu trúc:** Logic và giao diện được gộp chung cho mỗi trang (trong file .cshtml và .cshtml.cs).
- **Phù hợp cho:** Các ứng dụng đơn giản, tập trung vào biểu mẫu (forms) và các chức năng CRUD (tạo, đọc, sửa, xóa). Dễ học và phát triển nhanh hơn.

MVC (Model-View-Controller):

- **Mô hình:** Dựa trên mẫu thiết kế MVC.
- **Cấu trúc:** Tách biệt rõ ràng 3 thành phần: **Model** (dữ liệu), **View** (giao diện), và **Controller** (logic xử lý).
- **Phù hợp cho:** Các ứng dụng lớn, phức tạp, cần cấu trúc rõ ràng, dễ bảo trì, kiểm thử (test) và làm việc nhóm.

- Khi nào sử dụng Razor Pages :

Razor Pages phù hợp với CRUD nhỏ gọn. MVC tốt cho dự án lớn, tách biệt rõ vai trò.

2.6. Tag Helper

2.6.1. Các cài đặt cần thiết để sử dụng Tag Helper

Thêm dòng sau vào _ViewImports.cshtml:

```
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
```

2.6.2. Giới thiệu Tag Helper cơ bản

Tag Helpers là một tính năng của ASP.NET Core cho phép bạn viết mã phía máy chủ (server-side code) để tạo và sửa đổi các phần tử HTML trong các file Razor (.cshtml).

Về cơ bản, chúng trông giống như các thẻ HTML tiêu chuẩn nhưng được Razor engine xử lý trên máy chủ để sinh ra mã HTML cuối cùng. Điều này giúp mã trong file Razor trở nên **sạch sẽ, dễ đọc hơn** và rất thân thiện với các nhà thiết kế front-end vì cú pháp của nó rất giống với HTML thuần túy.

- Cách hoạt động của Tag Helper

1. **Xác định mục tiêu:** Tag Helpers nhắm vào các phần tử HTML dựa trên **tên thẻ** (ví dụ: <form>, <input>) hoặc **tên thuộc tính** (ví dụ: asp-for, asp-action).

2. **Xử lý trên máy chủ:** Khi Razor View được thực thi trên máy chủ, Razor engine sẽ tìm kiếm các Tag Helper này.

3. **Thực thi mã C#:** Mã C# tương ứng của Tag Helper sẽ được chạy. Nó có thể đọc các thuộc tính của Model, thực hiện logic, và sau đó sửa đổi hoặc tạo ra các thẻ HTML.

4. **Tạo ra HTML cuối cùng:** Kết quả là một đoạn mã HTML hoàn chỉnh được gửi đến trình duyệt của người dùng.

- Một số Tag Helper thường dùng

ASP.NET Core cung cấp nhiều Tag Helper tích hợp sẵn cho các tác vụ phổ biến:

Form Tag Helper (<form>):

Công dụng: Giúp tạo thẻ <form> và quản lý việc gửi dữ liệu.

Thuộc tính hay dùng: asp-controller, asp-action, asp-route-* để xác định Controller và Action sẽ xử lý form.

Input Tag Helper (<input>):

Công dụng: Tạo thẻ <input> được liên kết với một thuộc tính của Model.

Thuộc tính hay dùng: asp-for để liên kết với thuộc tính Model, tự động sinh ra id, name và các thuộc tính validation.

Label Tag Helper (<label>):

Công dụng: Tạo thẻ <label> cho một trường input.

Thuộc tính hay dùng: asp-for để hiển thị tên của thuộc tính Model.

Anchor Tag Helper (<a>):

Công dụng: Tạo liên kết (URL).

Thuộc tính hay dùng: asp-controller, asp-action, asp-page, asp-route-* để tạo URL động đến các trang hoặc action khác một cách an toàn.

Validation Tag Helpers (<div asp-validation-summary="..."> và):

Công dụng: Hiển thị thông báo lỗi xác thực (validation).

asp-validation-summary: Hiển thị một danh sách tóm tắt các lỗi.

asp-validation-for: Hiển thị lỗi cho một thuộc tính cụ thể.

Environment Tag Helper (<environment>):

Công dụng: Hiển thị nội dung tùy thuộc vào môi trường hiện tại của ứng dụng (Development, Staging, Production).

2.6.3. Áp dụng Tag Helper sẵn có

Sử dụng trong Form, Label, Input, Validation...

2.7. Model Binding

2.7.1. Khái niệm Model Binding

Tự động gán dữ liệu từ form vào model trong controller.

2.7.2. Data Type

```
public IActionResult Submit(string name, int age) { ... }
```

2.7.3. Complex Type

```
public IActionResult Submit(UserModel user) { ... }
```

2.8. Docker

Giới thiệu Docker và lý do sử dụng

- **Docker** là một nền tảng mã nguồn mở giúp bạn "đóng gói" một ứng dụng cùng với tất cả các thứ mà nó cần để chạy (như thư viện, công cụ hệ thống, mã nguồn và môi trường chạy) vào một đơn vị độc lập gọi là **container**.

- Cách triển khai ứng dụng ASP.NET Core với Docker

1. Tạo file Dockerfile
2. Build Image
3. Chạy Container

2.9. Các khái niệm cơ bản về Blazor

2.9.1. Blazor Server

- Chạy trên server, dùng SignalR.
- Phản hồi nhanh, phù hợp nội mạng.

2.9.2. Blazor WebAssembly

- Cơ chế hoạt động : Chạy trên trình duyệt.
- Khi nào nên sử dụng WebAssembly: Tự chủ, không phụ thuộc server.

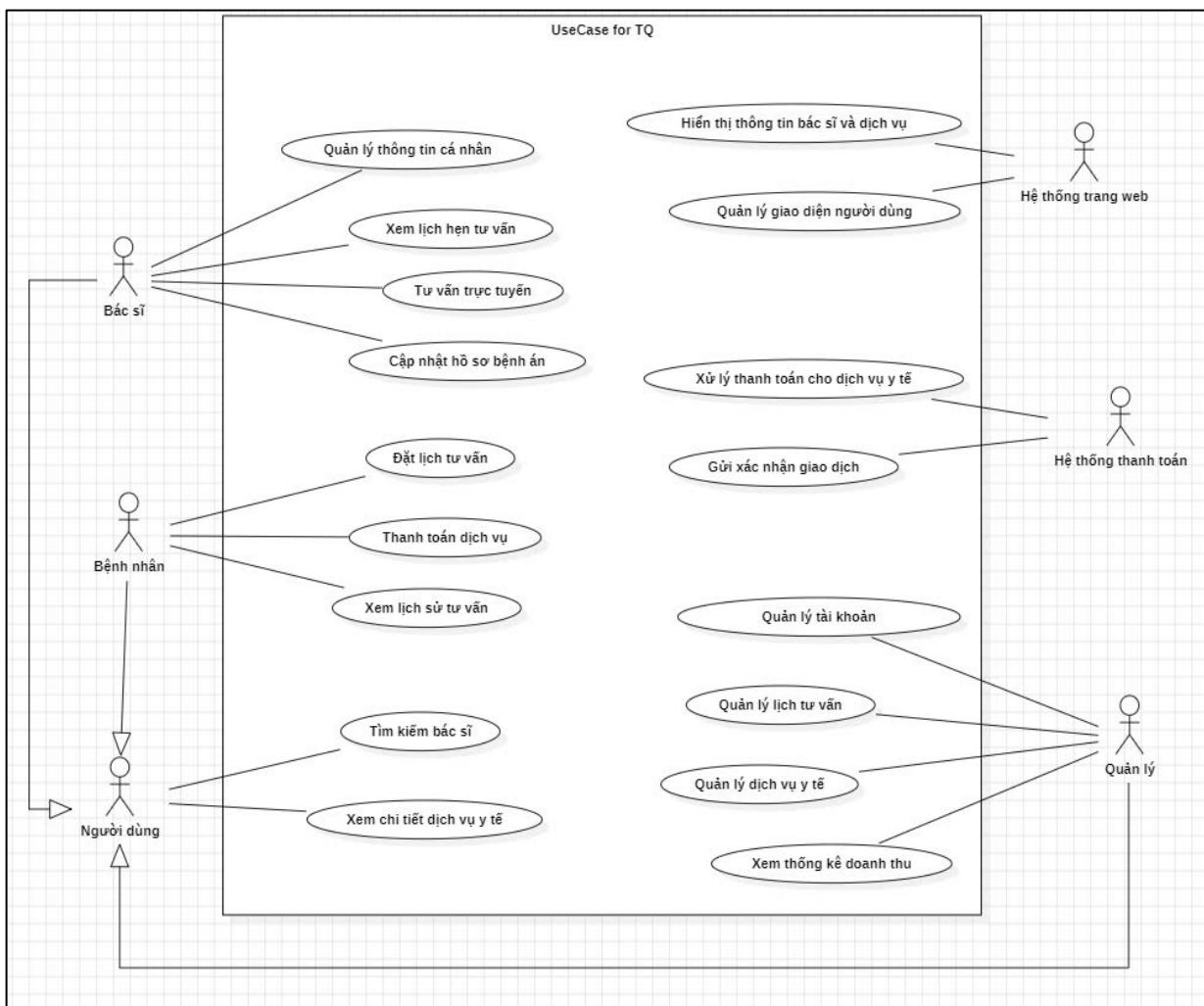
2.9.3. Giới thiệu ASP.NET Core Identity

- Quản lý người dùng trong ứng dụng: Cung cấp quản lý người dùng, phân quyền.
- Authentication & Authorization cơ bản: Hỗ trợ xác thực, đổi mật khẩu, xác minh 2 bước...

CHƯƠNG 3: ĐẶC TẢ CHỨC NĂNG

3.1. Trình bày sơ đồ tổng quát hệ thống

3.1.1. Sơ đồ tổng quát



Hình 3.1: Sơ đồ UC tổng quát

3.1.2. Mô tả các actor

Bảng 3.1: Danh sách các Actor tham gia sử dụng hệ thống

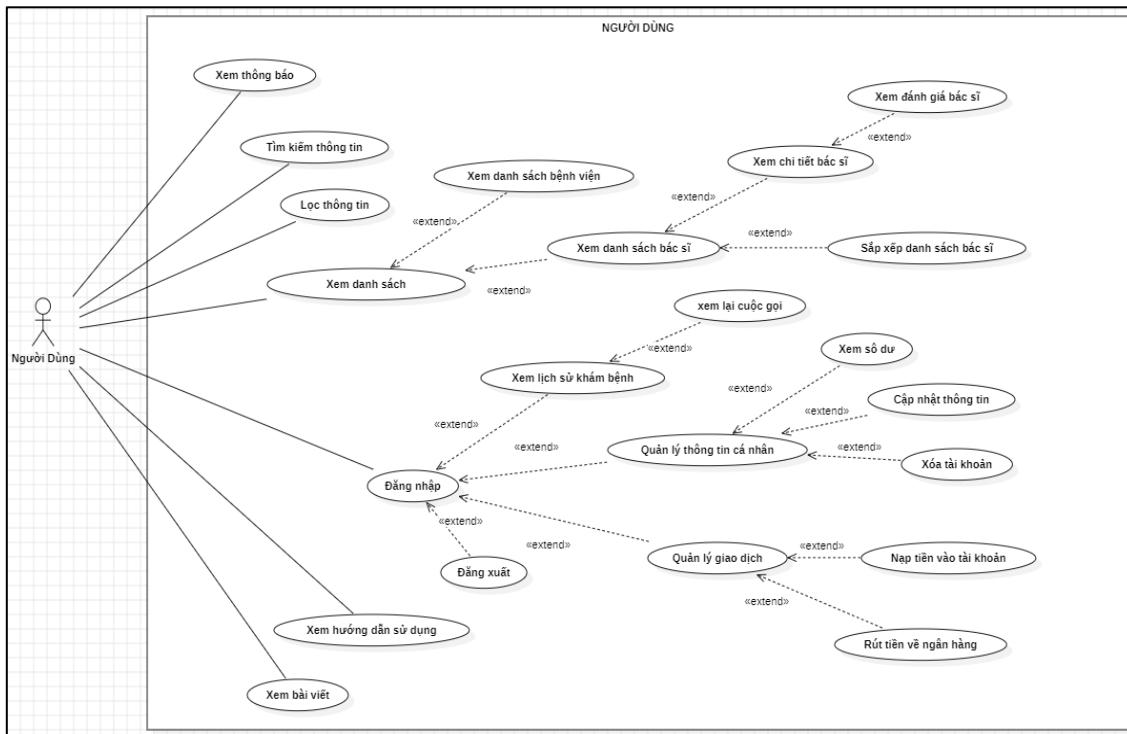
STT	Tên Actor	Quyền hạn
1	Hệ thống trang web	Quản lý toàn bộ hệ thống web.
2	Hệ thống thanh toán	Quản lý các chức năng giao dịch
3	Quản lý	Quản lý các chức năng thêm xóa sửa của hệ thống
4	Người dùng	Quản lý các chức năng chung của hệ thống

ĐẶC TẢ CHỨC NĂNG

5	Bác sĩ	Bác sĩ đăng ký hợp tác với admin để trở thành bác sĩ và quản lý các chức năng của bác sĩ.
6	Bệnh nhân	Bệnh nhân đã đăng ký tài khoản vào hệ thống và quản lý các chức năng của bệnh nhân

3.2. Mô tả cụ thể từng usecase

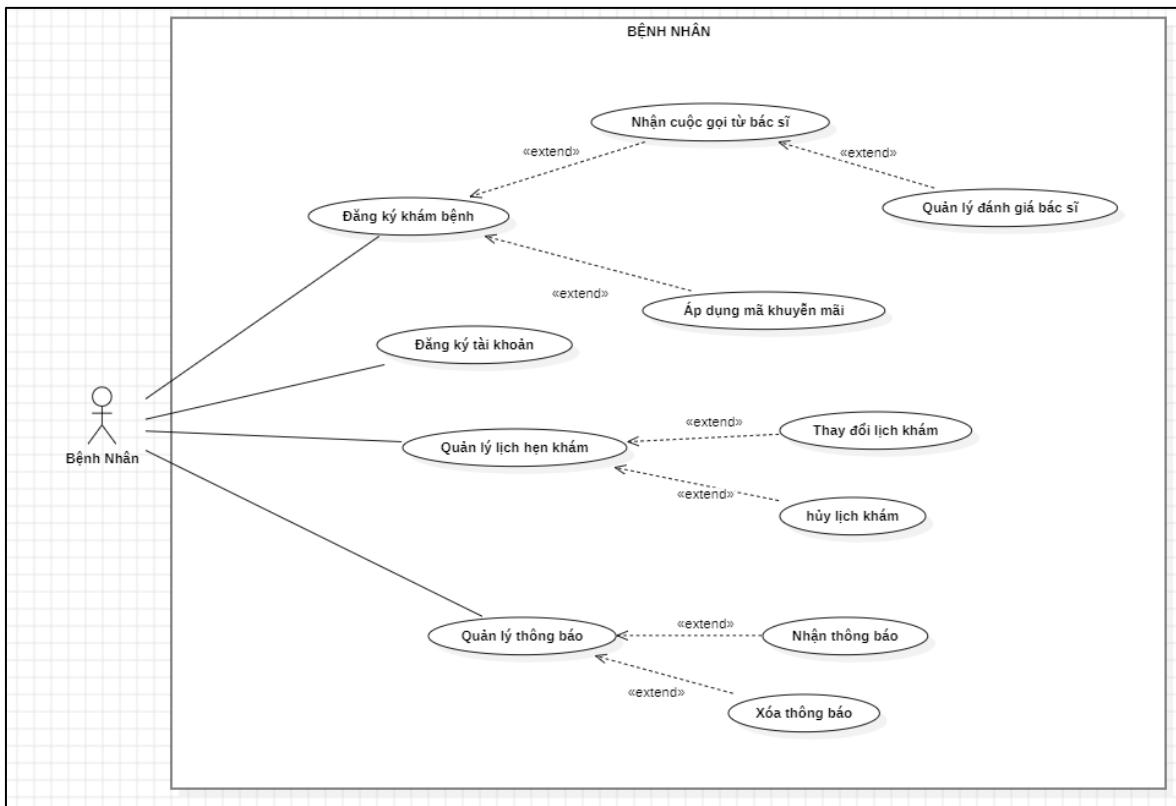
3.2.1. Sơ đồ chức năng của người dùng



Hình 3.2: Sơ đồ chức năng người dùng

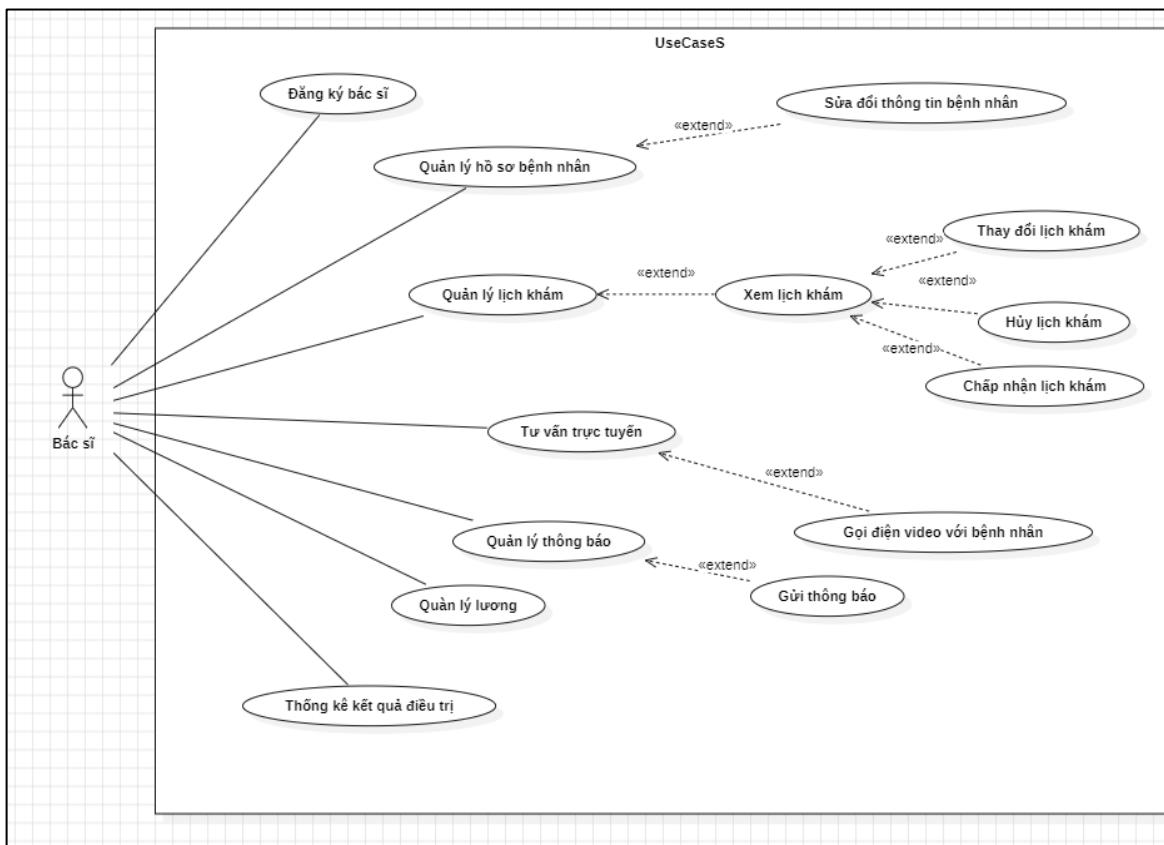
ĐẶC TẢ CHỨC NĂNG

3.2.2. Sơ đồ chức năng của bệnh nhân



Hình 3.3: Sơ đồ chức năng bệnh nhân

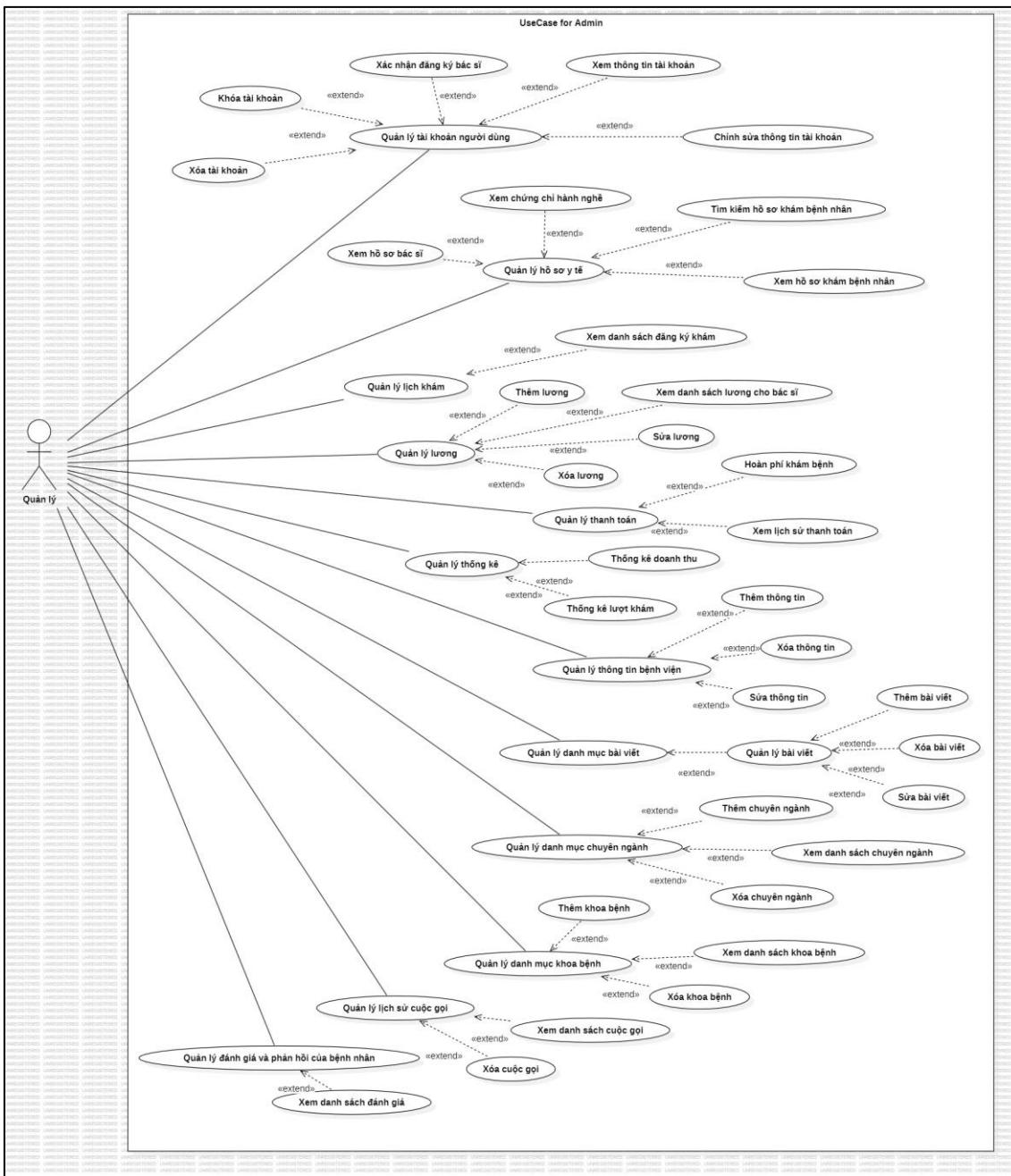
3.2.3. Sơ đồ chức năng của bác sĩ



Hình 3.4: Sơ đồ chức năng bác sĩ

ĐẶC TẢ CHỨC NĂNG

3.2.4. Sơ đồ quản lý



Hình 3.5: Sơ đồ chức năng quản lý

3.3. Mô tả về từng quy trình nghiệp vụ

Bảng 3.2: Bảng Usecase

Code	Sơ đồ chức năng	Tên Usecase
UC01	Sơ đồ người dùng	Xem thông báo
UC02	Sơ đồ người dùng	Tìm kiếm thông tin
UC03	Sơ đồ người dùng	Lọc thông tin

ĐẶC TẢ CHỨC NĂNG

UC04	Sơ đồ người dùng	Xem danh sách bệnh viện
UC05	Sơ đồ người dùng	Xem danh sách bác sĩ
UC06	Sơ đồ người dùng	Xem chi tiết bác sĩ
UC07	Sơ đồ người dùng	Xem đánh giá bác sĩ
UC08	Sơ đồ người dùng	Sắp xếp danh sách bác sĩ
UC09	Sơ đồ người dùng	Đăng nhập
UC10	Sơ đồ người dùng	Xem lịch sử khám bệnh
UC11	Sơ đồ người dùng	Xem lại cuộc gọi
UC12	Sơ đồ người dùng	Quản lý thông tin cá nhân
UC13	Sơ đồ người dùng	Xem số dư
UC14	Sơ đồ người dùng	Cập nhật thông tin
UC15	Sơ đồ người dùng	Xóa tài khoản
UC16	Sơ đồ người dùng	Quản lý giao dịch
UC17	Sơ đồ người dùng	Nạp tiền vào tài khoản
UC18	Sơ đồ người dùng	Rút tiền về ngân hàng
UC19	Sơ đồ người dùng	Xem hướng dẫn sử dụng
UC20	Sơ đồ người dùng	Xem bài viết
UC21	Sơ đồ bệnh nhân	Đăng ký khám bệnh
UC22	Sơ đồ bệnh nhân	Nhận cuộc gọi từ bác sĩ
UC23	Sơ đồ bệnh nhân	Quản lý đánh giá bác sĩ
UC24	Sơ đồ bệnh nhân	Áp dụng khuyến mãi
UC25	Sơ đồ bệnh nhân	Đăng ký tài khoản
UC26	Sơ đồ bệnh nhân	Quản lý lịch hẹn khám
UC27	Sơ đồ bệnh nhân	Thay đổi lịch khám
UC28	Sơ đồ bệnh nhân	Hủy lịch khám
UC29	Sơ đồ bệnh nhân	Quản lý thông báo
UC30	Sơ đồ bệnh nhân	Nhận thông báo
UC31	Sơ đồ bệnh nhân	Xóa thông báo
UC32	Sơ đồ bác sĩ	Đăng ký bác sĩ

ĐẶC TẢ CHỨC NĂNG

UC33	Sơ đồ bác sĩ	Quản lý hồ sơ bệnh nhân
UC34	Sơ đồ bác sĩ	Sửa đổi thông tin bệnh nhân
UC35	Sơ đồ bác sĩ	Quản lý lịch khám
UC36	Sơ đồ bác sĩ	Xem lịch khám
UC37	Sơ đồ bác sĩ	Thay đổi lịch khám
UC38	Sơ đồ bác sĩ	Hủy lịch khám
UC39	Sơ đồ bác sĩ	Chấp nhận lịch khám
UC40	Sơ đồ bác sĩ	Tư vấn trực tuyến
UC41	Sơ đồ bác sĩ	Gọi điện với bệnh nhân
UC42	Sơ đồ bác sĩ	Quản lý thông báo
UC43	Sơ đồ bác sĩ	Gửi thông báo
UC44	Sơ đồ bác sĩ	Quản lý lương
UC45	Sơ đồ bác sĩ	Thông kê kết quả điều trị
UC46	Sơ đồ quản lý	Quản lý tài khoản người dùng
UC47	Sơ đồ quản lý	Xem thông tin tài khoản
UC48	Sơ đồ quản lý	Sửa thông tin tài khoản
UC49	Sơ đồ quản lý	Khóa tài khoản
UC50	Sơ đồ quản lý	Xóa tài khoản
UC51	Sơ đồ quản lý	Xác nhận đăng ký bác sĩ
UC52	Sơ đồ quản lý	Quản lý hồ sơ y tế
UC53	Sơ đồ quản lý	Xem hồ sơ khám bệnh
UC54	Sơ đồ quản lý	Tìm kiếm hồ sơ khám bệnh
UC55	Sơ đồ quản lý	Xem chứng chỉ hành nghề
UC56	Sơ đồ quản lý	Xem hồ sơ bác sĩ
UC57	Sơ đồ quản lý	Quản lý lịch khám
UC58	Sơ đồ quản lý	Xem danh sách đăng ký khám
UC59	Sơ đồ quản lý	Quản lý lương
UC60	Sơ đồ quản lý	Xem danh sách lương
UC61	Sơ đồ quản lý	Thêm lương

ĐẶC TẢ CHỨC NĂNG

UC62	Sơ đồ quản lý	Xóa lương
UC63	Sơ đồ quản lý	Sửa lương
UC64	Sơ đồ quản lý	Quản lý thanh toán
UC65	Sơ đồ quản lý	Hoàn phí khám bệnh
UC66	Sơ đồ quản lý	Xem lịch sử thanh toán
UC67	Sơ đồ quản lý	Quản lý thống kê
UC68	Sơ đồ quản lý	Thống kê doanh thu
UC69	Sơ đồ quản lý	Thống kê lượt khám
UC70	Sơ đồ quản lý	Quản lý thông tin bệnh viện
UC71	Sơ đồ quản lý	Thêm thông tin bệnh viện
UC72	Sơ đồ quản lý	Sửa thông tin bệnh viện
UC73	Sơ đồ quản lý	Xóa thông tin bệnh viện
UC74	Sơ đồ quản lý	Quản lý danh mục bài viết
UC75	Sơ đồ quản lý	Quản lý bài viết
UC76	Sơ đồ quản lý	Thêm bài viết
UC77	Sơ đồ quản lý	Sửa bài viết
UC78	Sơ đồ quản lý	Xóa bài viết
UC79	Sơ đồ quản lý	Quản lý danh mục chuyên ngành
UC80	Sơ đồ quản lý	Thêm chuyên ngành
UC81	Sơ đồ quản lý	Sửa chuyên ngành
UC82	Sơ đồ quản lý	Xóa chuyên ngành
UC83	Sơ đồ quản lý	Quản lý danh mục khoa bệnh
UC84	Sơ đồ quản lý	Thêm khoa bệnh
UC85	Sơ đồ quản lý	Xem danh sách khoa bệnh
UC86	Sơ đồ quản lý	Xóa khoa bệnh
UC87	Sơ đồ quản lý	Quản lý lịch sử cuộc gọi
UC88	Sơ đồ quản lý	Xem danh sách cuộc gọi
UC89	Sơ đồ quản lý	Xóa cuộc gọi
UC90	Sơ đồ quản lý	Quản lý đánh giá phản hồi bệnh nhân

UC91	Sơ đồ quản lý	Xem danh sách đánh giá
------	---------------	------------------------

3.3.1. UC01 [Xem thông báo]

Tên UC	Xem thông báo
Mã UC	UC01
Tác nhân chính	Người dùng
Tóm tắt	Người dùng xem thông báo khi có thông báo mới đến
Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Hiển thị thông báo lên màn hình
Dòng sự kiện chính	<ol style="list-style-type: none"> Hệ thống hiện thị biểu tượng thông báo khi người dùng đã đăng nhập Người dùng nhấn vào biểu tượng thông báo trên màn hình Hệ thống hiển thị thông báo lên màn hình
Dòng sự kiện phụ	Thông báo “Bạn chưa có thông báo nào” khi chưa có thông báo đến người dùng

3.3.2. UC02 [Tìm kiếm thông tin]

Tên UC	Tìm kiếm thông tin
Mã UC	UC02
Tác nhân chính	Người dùng
Tóm tắt	Người dùng tìm kiếm tên bác sĩ
Mối quan hệ	
Tiền điều kiện	Người dùng đã vào trang lọc bác sĩ
Hậu điều kiện	Hiển thị thông tin chi tiết bác sĩ
Dòng sự kiện chính	<ol style="list-style-type: none"> Hệ thống load danh sách tên bác sĩ vào list bị ẩn Người dùng nhập tên bác sĩ vào ô tìm kiếm Hệ thống lắng nghe sự kiện nhập bàn phím Hệ thống hiện danh sách đề xuất tên bác sĩ trùng với những ký tự người dùng đã nhập

ĐẶC TẢ CHỨC NĂNG

	5. Người dùng nhấn vào tên bác sĩ dưới ô tìm kiếm 6. Hệ thống hiển thị thông tin chi tiết bác sĩ
Dòng sự kiện phụ	

3.3.3. UC03 [Lọc thông tin]

Tên UC	Lọc thông tin
Mã UC	UC03
Tác nhân chính	Người dùng
Tóm tắt	Người dùng lọc thông tin bác sĩ
Mối quan hệ	
Tiền điều kiện	Người dùng đã vào trang lọc bác sĩ
Hậu điều kiện	Hiển thị thông tin chi tiết bác sĩ
Dòng sự kiện chính	1. Người dùng chọn các option trên màn hình 2. Hệ thống đề xuất các ô select trong từng option 3. Người dùng nhấn nút lọc bác sĩ 4. Hệ thống hiển thị danh sách bác sĩ
Dòng sự kiện phụ	Thông báo “Rất tiếc! Chúng tôi không Tìm thấy bác sĩ” khi không tìm thấy kết quả phù hợp

3.3.4. UC08 [Sắp xếp danh sách bác sĩ]

Tên UC	Sắp xếp danh sách bác sĩ
Mã UC	UC08
Tác nhân chính	Người dùng
Tóm tắt	Người dùng sắp xếp danh sách bác sĩ theo tiêu chí lượt chò và phí khám
Mối quan hệ	
Tiền điều kiện	Người dùng đã vào trang lọc bác sĩ
Hậu điều kiện	Hiển thị danh sách bác sĩ

ĐẶC TẢ CHỨC NĂNG

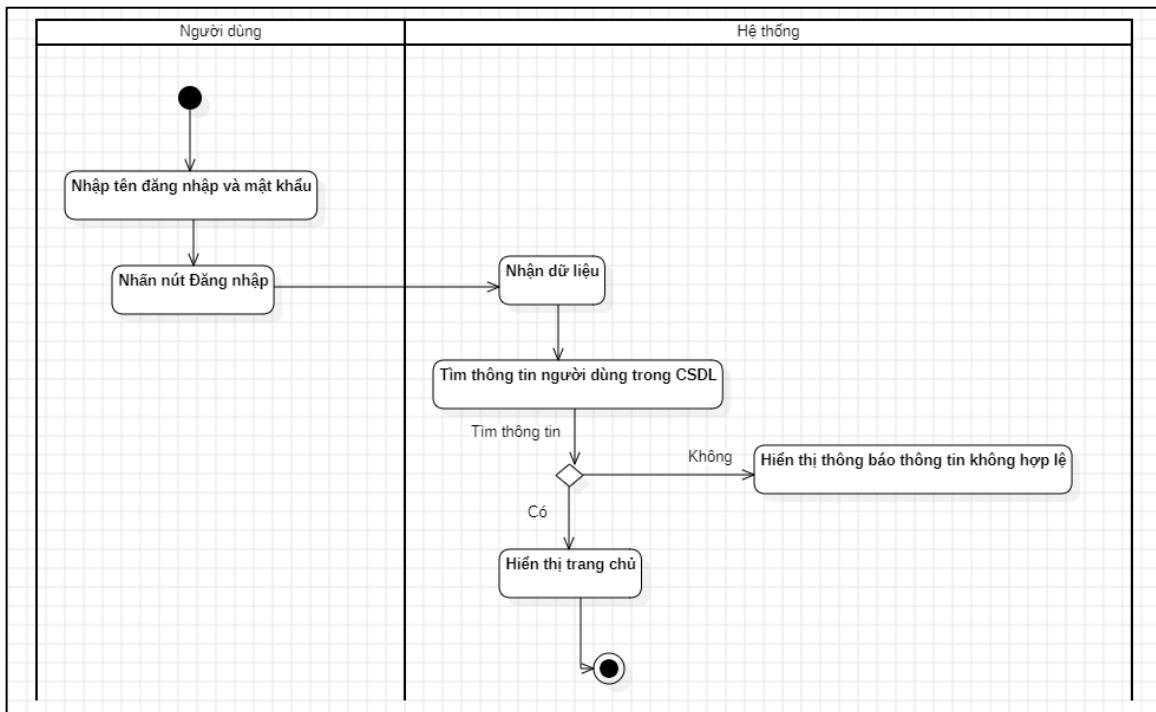
Dòng sự kiện chính	1. Người dùng chọn các option sắp xếp danh sách trên màn hình 2. Hệ thống đề xuất các ô select trong từng option 3. Hệ thống hiển thị danh sách bác sĩ tương ứng
Dòng sự kiện phụ	

3.3.5. UC09 [Đăng nhập]

Tên UC	Đăng nhập
Mã UC	UC09
Tác nhân chính	Người dùng
Tóm tắt	Người dùng sắp xếp danh sách bác sĩ theo tiêu chí lượt chờ và phí khám
Mối quan hệ	
Tiền điều kiện	Người dùng chưa đăng nhập vào hệ thống và đang ở trang đăng nhập
Hậu điều kiện	Hiển thị giao diện trang chủ
Dòng sự kiện chính	1. Người dùng truy cập vào trang đăng nhập của hệ thống. 2. Người dùng nhập tên đăng nhập và mật khẩu vào các trường tương ứng. 3. Người dùng nhấn nút "Đăng nhập" 4. Hệ thống kiểm tra và xác nhận thông tin đăng nhập 5. Hệ thống chuyển hướng đến giao diện trang chủ.
Dòng sự kiện phụ	Hệ thống hiển thị lại form đăng nhập khi không tìm thấy tài khoản.

Sơ đồ hoạt động:

ĐẶC TẢ CHỨC NĂNG



Hình 3.6: Activity Diagram [Đăng nhập]

3.3.6. UC [Xem lại cuộc gọi]

Tên UC	Xem lại cuộc gọi
Mã UC	UC11
Tác nhân chính	Người dùng
Tóm tắt	Người dùng xem lại cuộc gọi đã khám
Mối quan hệ	
Tiền điều kiện	<p>Người dùng đã đăng nhập vào hệ thống và đã từng thực hiện cuộc gọi.</p> <p>Người dùng đang ở trang lịch sử khám</p>
Hậu điều kiện	Hiển thị video lên màn hình
Dòng sự kiện chính	<ol style="list-style-type: none"> Người dùng vào trang lịch sử khám Hệ thống hiển thị danh sách lịch sử khám Người dùng nhấn vào nút xem lại cuộc gọi Hệ thống hiển thị video
Dòng sự kiện phụ	Thông báo “Lịch sử cuộc gọi trống” khi không có cuộc gọi nào

ĐẶC TẢ CHỨC NĂNG

3.3.7. UC12 [Quản lý thông tin cá nhân]

Tên UC	Quản lý thông tin cá nhân
Mã UC	UC12
Tác nhân chính	Người dùng
Tóm tắt	Người dùng quản lý thông tin cá nhân
Mối quan hệ	Cập nhật thông tin, Xóa tài khoản
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, chỉnh sửa hoặc xóa tài khoản thành công
Dòng sự kiện chính	<ol style="list-style-type: none">Người dùng nhấn vào biểu tượng avatar trên màn hìnhHệ thống hiển thị thông tin chi tiết người dùngNgười dùng thực hiện các thao tác (thêm, xóa, sửa)Hệ thống thông báo thành công
Dòng sự kiện phụ	

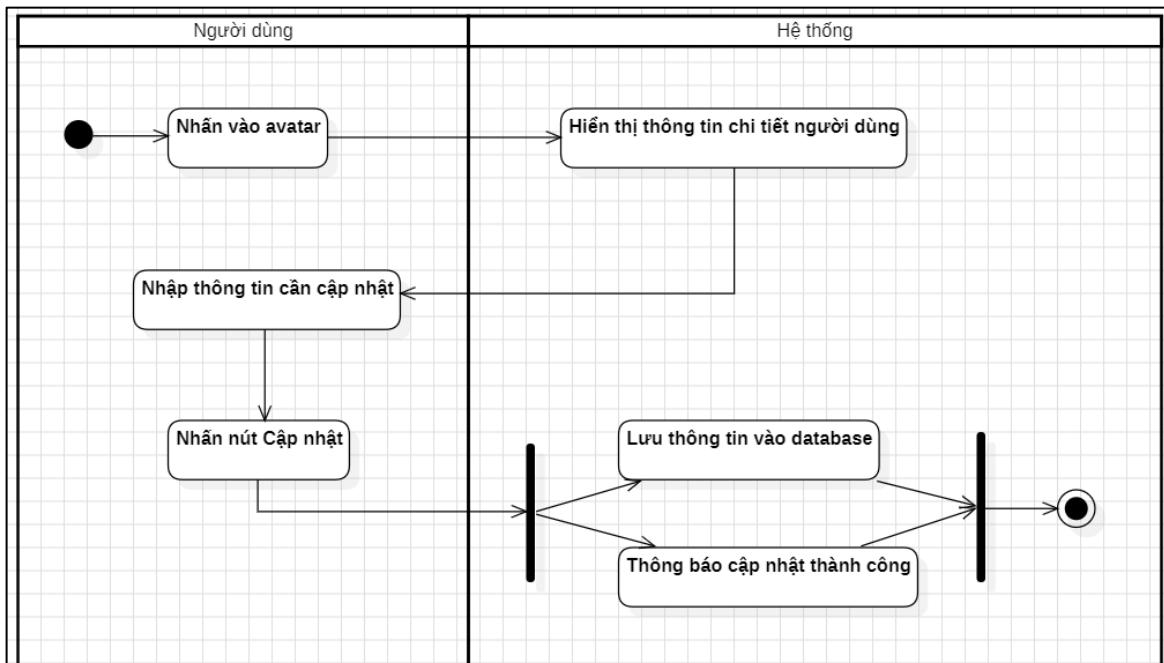
3.3.8. UC14 [Cập nhật thông tin cá nhân]

Tên UC	Cập nhật thông tin cá nhân
Mã UC	UC14
Tác nhân chính	Người dùng
Tóm tắt	Người dùng cập nhật thông tin cá nhân
Mối quan hệ	Quản lý thông tin cá nhân
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác chỉnh sửa tài khoản thành công
Dòng sự kiện chính	<ol style="list-style-type: none">Người dùng nhấn vào biểu tượng avatar trên màn hìnhHệ thống hiển thị thông tin chi tiết người dùngNgười dùng nhập thông tin vào các ô có trên màn hìnhNgười dùng nhấn nút “Cập nhật”Hệ thống hiển thị trang cá nhân đã được cập nhật

ĐẶC TẢ CHỨC NĂNG

Dòng sự kiện phụ	Hệ thống thông báo nhập lại khi người dùng nhập sai định dạng
------------------	---

Sơ đồ hoạt động:



Hình 3.7: Activity Diagram [Cập nhật thông tin cá nhân]

3.3.9. UC15 [Xóa tài khoản]

Tên UC	Xóa tài khoản
Mã UC	UC15
Tác nhân chính	Người dùng
Tóm tắt	Người dùng xóa tài khoản khi không sử dụng website
Mối quan hệ	Quản lý thông tin cá nhân
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xóa tài khoản thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> Người dùng nhấn vào biểu tượng avatar trên màn hình Hệ thống hiển thị thông tin chi tiết người dùng Người dùng nhấn nút “Xóa tài khoản” Hệ thống yêu cầu xác nhận lại Người dùng nhấn “Xác nhận” Hệ thống thông báo xóa thành công

ĐẶC TẢ CHỨC NĂNG

	7. Hệ thống chuyển sang giao diện trang chủ
Dòng sự kiện phụ	Hệ thống thông báo nhập lại khi người dùng nhập sai định dạng

3.3.10. UC16 [Quản lý giao dịch]

Tên UC	Quản lý giao dịch
Mã UC	UC16
Tác nhân chính	Người dùng
Tóm tắt	Người dùng quản lý thông tin giao dịch bao gồm nạp tiền, rút tiền
Mối quan hệ	Nạp tiền vào tài khoản, Rút tiền về ngân hàng
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác Nạp tiền vào tài khoản, Rút tiền về ngân hàng thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng di chuyển chuột đến “Giao dịch” trên màn hình 2. Hệ thống hiển thị các option quản lý giao dịch 3. Người dùng chọn thao tác muốn sử dụng 4. Người dùng thực hiện các thao tác (Nạp tiền, Rút tiền) 5. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.11. UC17 [Nạp tiền vào tài khoản]

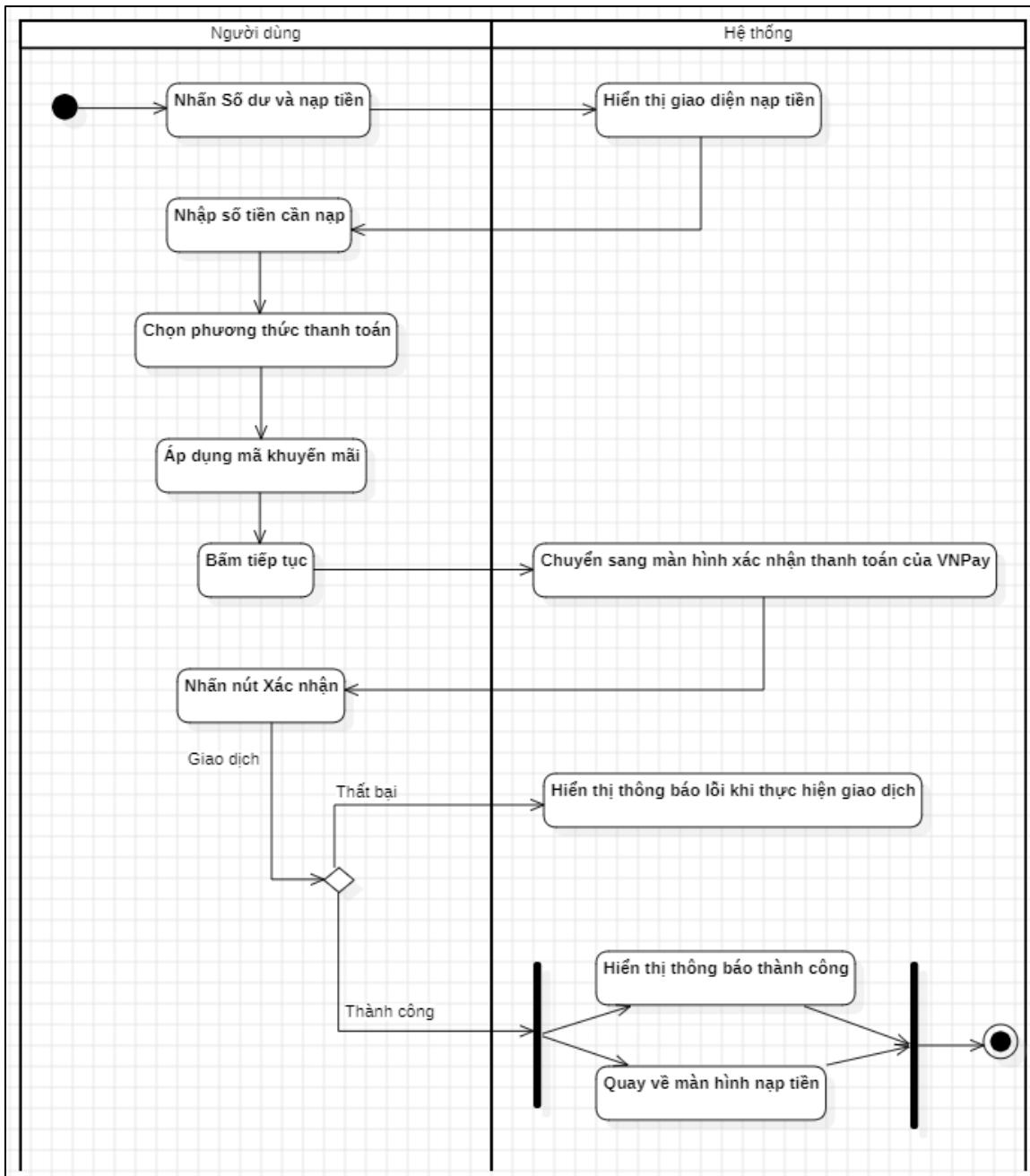
Tên UC	Nạp tiền vào tài khoản
Mã UC	UC17
Tác nhân chính	Người dùng
Tóm tắt	Người dùng nạp tiền thành công và hiển thị số dư
Mối quan hệ	Quản lý giao dịch, Áp dụng khuyến mãi
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác Nạp tiền vào tài khoản thành công

ĐẶC TẢ CHỨC NĂNG

Dòng sự kiện chính	<ol style="list-style-type: none">1. Người dùng nhấn “Số dư và nạp tiền”2. Hệ thống hiển thị giao diện nạp tiền3. Người dùng nhập số tiền4. Người dùng chọn phương thức thanh toán5. Người dùng thực hiện UC[Áp dụng khuyến mãi] nếu có6. Người dùng bấm tiếp tục7. Hệ thống kiểm tra dữ liệu hợp lệ8. Người dùng nhấn “submit”9. Hệ thống chuyển về màn hình nạp tiền
Dòng sự kiện phụ	

Sơ đồ hoạt động:

ĐẶC TẢ CHỨC NĂNG



Hình 3.8: Activity Diagram [Nạp tiền]

3.3.12. UC18 [Rút tiền về ngân hàng]

Tên UC	Rút tiền về ngân hàng
Mã UC	UC18
Tác nhân chính	Người dùng
Tóm tắt	Người dùng Rút tiền về ngân hàng và hiển thị số dư
Mối quan hệ	Quản lý giao dịch
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống

ĐẶC TẢ CHỨC NĂNG

Hậu điều kiện	Thực hiện các thao tác Rút tiền về ngân hàng thành công
Dòng sự kiện chính	1. Người dùng nhấn “Chuyển tiền” 2. Hệ thống hiển thị giao diện rút tiền 3. Người dùng nhập số tiền cần rút 4. Người dùng chọn phương thức thanh toán 5. Người dùng bấm tiếp tục 6. Hệ thống kiểm tra dữ liệu hợp lệ 7. Người dùng nhấn “Xác nhận” 8. Hệ thống thông báo thành công 9. Hệ thống chuyển về màn hình rút tiền
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi thực hiện giao dịch không thành công

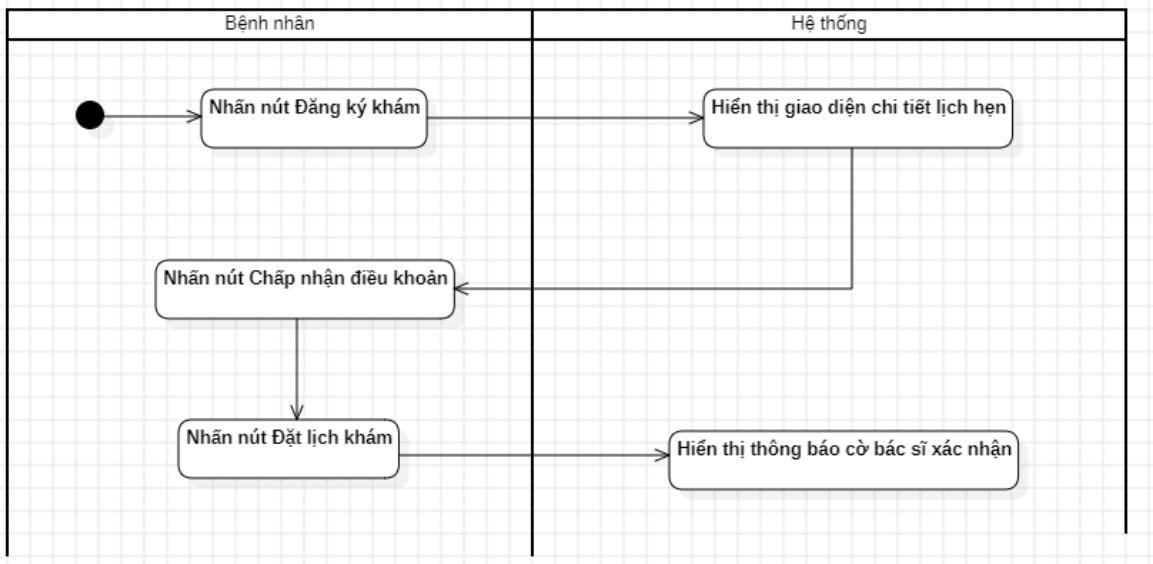
3.3.13. UC21 [Đăng ký khám bệnh]

Tên UC	Đăng ký khám bệnh
Mã UC	UC21
Tác nhân chính	Bệnh nhân
Tóm tắt	Bệnh nhân đặt lịch khám với bác sĩ
Mối quan hệ	UC17 [Nạp tiền vào tài khoản]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống Người dùng đang ở trang chi tiết bác sĩ
Hậu điều kiện	Thông báo “Xin vui lòng chờ bác sĩ xác nhận”
Dòng sự kiện chính	1. Người dùng nhấn nút “Đăng ký khám” 2. Hệ thống hiển thị giao diện chi tiết lịch hẹn 3. Người dùng nhập thông tin lịch khám 4. Người dùng nhấn nút “Chấp nhận điều khoản” 5. Người dùng nhấn nút “Đặt lịch khám” 6. Hệ thống hiển thị thông báo “Xin vui lòng chờ bác sĩ xác nhận”

ĐẶC TẢ CHỨC NĂNG

Dòng sự kiện chính	Hệ thống yêu cầu người dùng nhập đầy đủ thông tin khi thông tin bị thiếu
Dòng sự kiện phụ	Hệ thống yêu cầu nạp tiền khi số dư lớn hơn hoặc bằng số tiền khám

Sơ đồ hoạt động:



Hình 3.9: Activity Diagram [Đăng ký khám bệnh]

3.3.14. UC22 [Nhận cuộc gọi từ bác sĩ]

Tên UC	Nhận cuộc gọi từ bác sĩ
Mã UC	UC22
Tác nhân chính	Bệnh nhân
Tóm tắt	Bệnh nhân Nhận cuộc gọi từ bác sĩ
Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Hệ thống kết nối máy đến bác sĩ
Dòng sự kiện chính	<ol style="list-style-type: none"> Hệ thống hiển thị màn hình cuộc gọi Người dùng nhấn vào biểu tượng màu xanh trên màn hình Hệ thống kết nối với camera của bác sĩ
Dòng sự kiện phụ	

3.3.15. UC23 [Quản lý đánh giá bác sĩ]

Tên UC	Quản lý đánh giá bác sĩ
Mã UC	UC23
Tác nhân chính	Bệnh nhân
Tóm tắt	Bệnh nhân đánh giá bác sĩ sau khi nhận cuộc gọi
Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống Đã thực hiện cuộc gọi với bác sĩ và chưa đánh giá
Hậu điều kiện	Hệ thống thông báo đánh giá thành công
Dòng sự kiện chính	1. Người dùng vào trang lịch sử cuộc gọi 2. Người dùng chọn “đánh giá” 3. Hệ thống hiển thị form đánh giá 4. Người dùng nhập đánh giá và chọn số sao 5. Người dùng nhấn “Gửi” 6. Hệ thống lưu kết quả 7. Hệ thống báo đánh giá thành công
Dòng sự kiện phụ	

3.3.16. UC24 [Áp dụng khuyến mãi]

Tên UC	Áp dụng khuyến mãi
Mã UC	UC24
Tác nhân chính	Người dùng
Tóm tắt	Người dùng nạp tiền thành công và hiển thị số dư
Mối quan hệ	UC17 [Nạp tiền vào tài khoản]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác Nạp tiền và áp dụng mã giảm giá vào tài khoản thành công

ĐẶC TẢ CHỨC NĂNG

Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng nhập mã giảm giá 2. Người dùng tiếp tục thực hiện các bước ở 3. Người dùng nhấn “Xác nhận” 4. Hệ thống thông báo thành công 5. Hệ thống chuyển về màn hình nạp tiền
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi thực hiện giao dịch không thành công

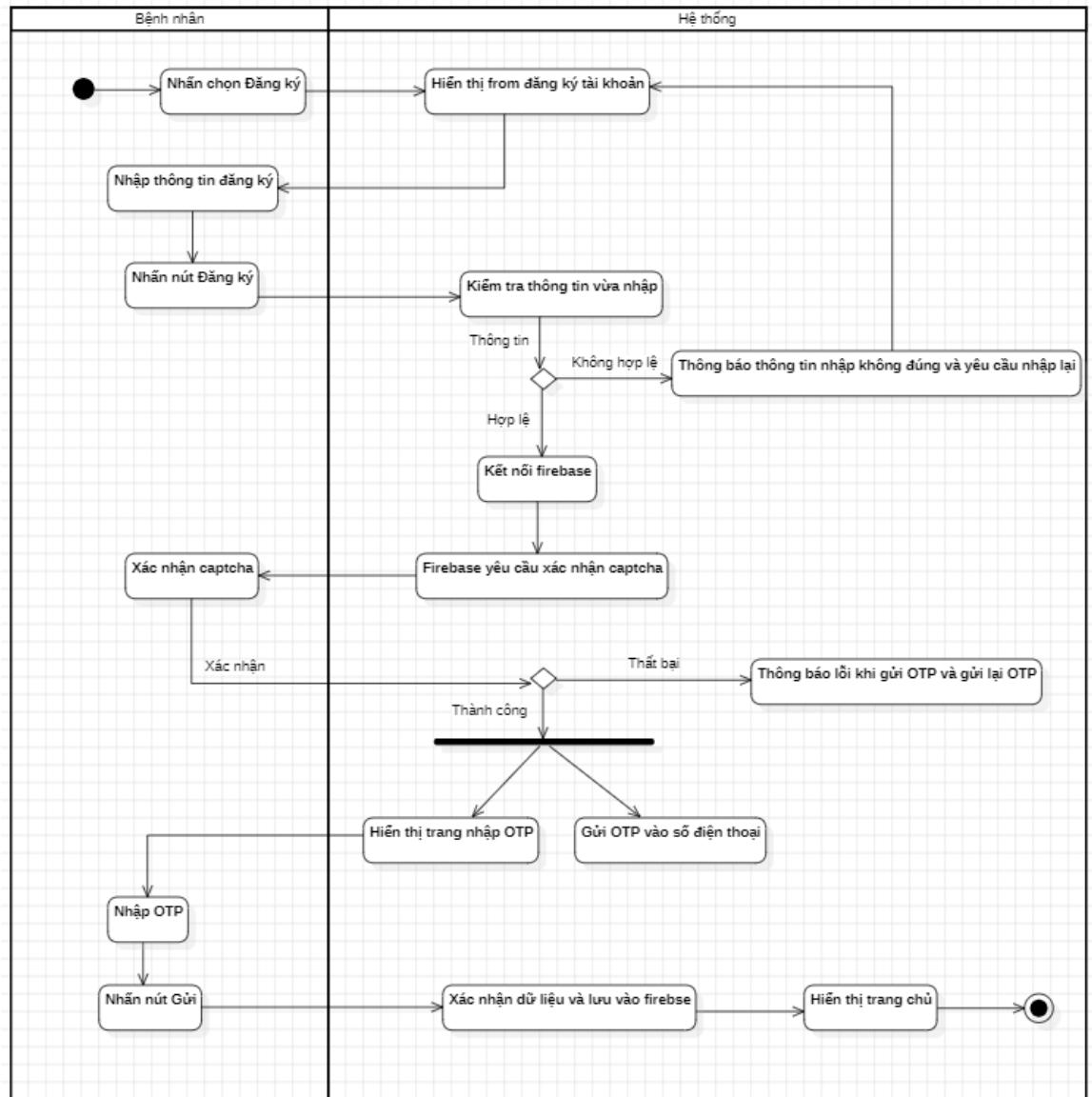
3.3.17. UC25 [Đăng ký tài khoản]

Tên UC	Đăng ký tài khoản
Mã UC	UC25
Tác nhân chính	Bệnh nhân
Tóm tắt	Bệnh nhân đăng ký tài khoản để sử dụng các dịch vụ của website
Mối quan hệ	
Tiền điều kiện	<p>Người dùng truy cập vào hệ thống.</p> <p>Người dùng chưa có tài khoản trên hệ thống.</p>
Hậu điều kiện	<p>Người dùng đã đăng ký thành công tài khoản</p> <ol style="list-style-type: none"> 1. Người dùng truy cập vào hệ thống. 2. Người dùng chọn "Đăng ký" 3. Hệ thống hiển thị form đăng ký tài khoản. 4. Người dùng nhập thông tin để đăng ký tài khoản. 5. Người dùng nhấn nút “Đăng ký”. 6. Hệ thống kiểm tra thông tin vừa nhập. 7. Hệ thống kết nối với dịch vụ firebase 8. Firebase yêu cầu xác nhận captcha 9. Người dùng xác nhận captcha 10. Hệ thống gửi OTP vào số điện thoại 11. Hệ thống chuyển màn hình sang trang nhập OTP 12. Người dùng dùng nhập OTP 13. Người dùng bấm gửi OTP
Dòng sự kiện chính	

ĐẶC TẢ CHỨC NĂNG

	14. Hệ thống xác nhận dữ liệu và lưu vào database 15. Hệ thống hiển thị giao diện trang chủ.
Dòng sự kiện phụ	Hệ thống yêu cầu nhập lại khi nhập sai định dạng dữ liệu Hệ thống thông báo lỗi khi gửi OTP không thành công

Sơ đồ hoạt động:



Hình 3.10: Activity Diagram [Đăng ký tài khoản]

3.3.18. UC26 [Quản lý lịch hẹn]

Tên UC	Quản lý lịch hẹn
Mã UC	UC26

ĐẶC TẢ CHỨC NĂNG

Tác nhân chính	Bệnh nhân
Tóm tắt	Người dùng quản lý lịch hẹn bao gồm xem, thay đổi, hủy lịch.
Mối quan hệ	UC27 [Thay đổi lịch khám], UC28 [Hủy lịch khám]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, thay đổi, hủy lịch thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng vào trang lịch sử khám 2. Hệ thống hiển thị giao diện lịch sử khám 3. Người dùng thực hiện các thao tác của UC27, UC28 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.19. UC27 [Thay đổi lịch khám]

Tên UC	Thay đổi lịch khám
Mã UC	UC27
Tác nhân chính	Bệnh nhân
Tóm tắt	Người dùng có thể chỉnh sửa thay đổi thông tin của lịch hẹn.
Mối quan hệ	UC26 [Quản lý lịch hẹn]
Tiền điều kiện	<p>Người dùng đã đăng nhập vào hệ thống Lịch khám đang trong trạng thái “Chờ xác nhận”</p>
Hậu điều kiện	Thực hiện thay đổi lịch thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng vào trang lịch sử khám 2. Hệ thống hiển thị giao diện lịch sử khám 3. Người dùng nhấn nút “Chỉnh sửa” 4. Hệ thống hiển thị chi tiết lịch khám 5. Người dùng thực hiện thao tác chỉnh sửa 6. Người dùng nhấn nút “Cập nhật” 7. Hệ thống lưu thông tin vào database 8. Hệ thống thông báo thành công

ĐẶC TẢ CHỨC NĂNG

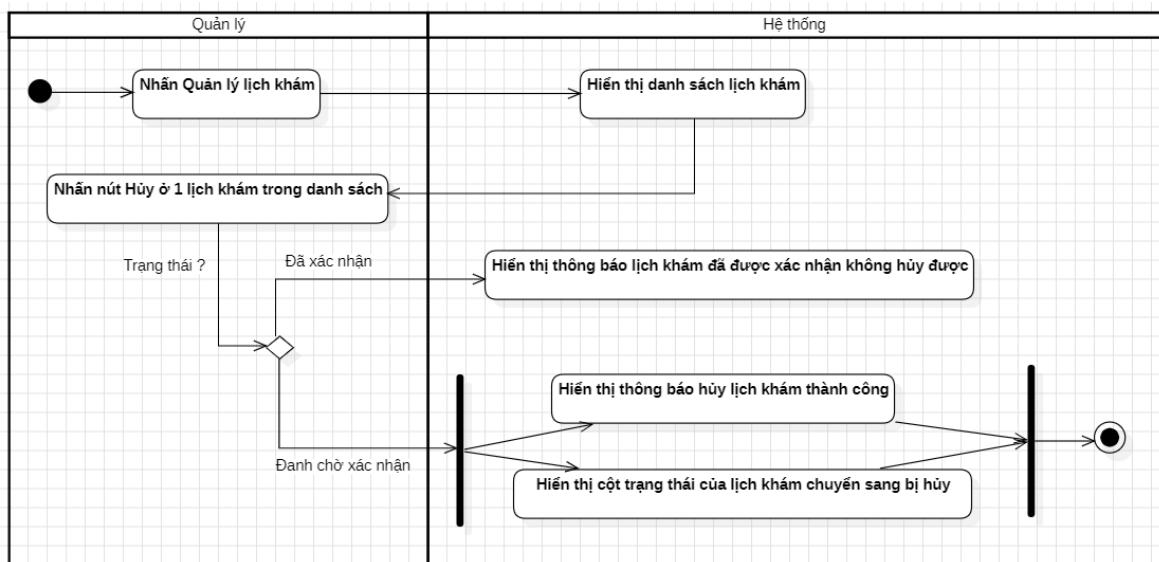
Dòng sự kiện phụ	
------------------	--

3.3.20. UC28 [Hủy lịch khám]

Tên UC	Hủy lịch khám
Mã UC	UC28
Tác nhân chính	Bệnh nhân
Tóm tắt	Người dùng có thể hủy lịch hẹn.
Mối quan hệ	UC26 [Quản lý lịch hẹn]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống Lịch khám đang trong trạng thái “Chờ xác nhận”
Hậu điều kiện	Thực hiện hủy lịch thành công
Dòng sự kiện chính	1. Người dùng vào trang lịch sử khám 2. Hệ thống hiển thị giao diện lịch sử khám 3. Người dùng nhấn nút “Hủy” 4. Hệ thống yêu cầu xác nhận lại 5. Người dùng nhấn nút “xác nhận” 6. Hệ thống xóa thông tin trong database 7. Hệ thống thông báo thành công
Dòng sự kiện phụ	

Sơ đồ hoạt động:

ĐẶC TẢ CHỨC NĂNG



Hình 3.11: Activity Diagram [Hủy lịch khám]

3.3.21. UC29 [Quản lý thông báo]

Tên UC	Quản lý thông báo
Mã UC	UC29
Tác nhân chính	Quản lý
Tóm tắt	Người dùng quản lý lịch hẹn bao gồm xem, tạo, xóa thông báo.
Mối quan hệ	UC31 [Xóa thông báo]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, tạo, xóa thông báo thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> Người dùng nhấn vào biểu tượng thông báo trên màn hình Hệ thống hiển thị màn hình thông báo Người dùng thực hiện thao tác UC31 nếu muốn Hệ thống thông báo thành công
Dòng sự kiện phụ	Hệ thống thông báo “Bạn chưa có thông báo nào” khi người dùng chưa có thông báo

3.3.22. UC31 [Xóa thông báo]

Tên UC	Xóa thông báo
Mã UC	UC31
Tác nhân chính	Bệnh nhân
Tóm tắt	Người dùng xóa thông báo khi không cần thiết.

ĐẶC TẢ CHỨC NĂNG

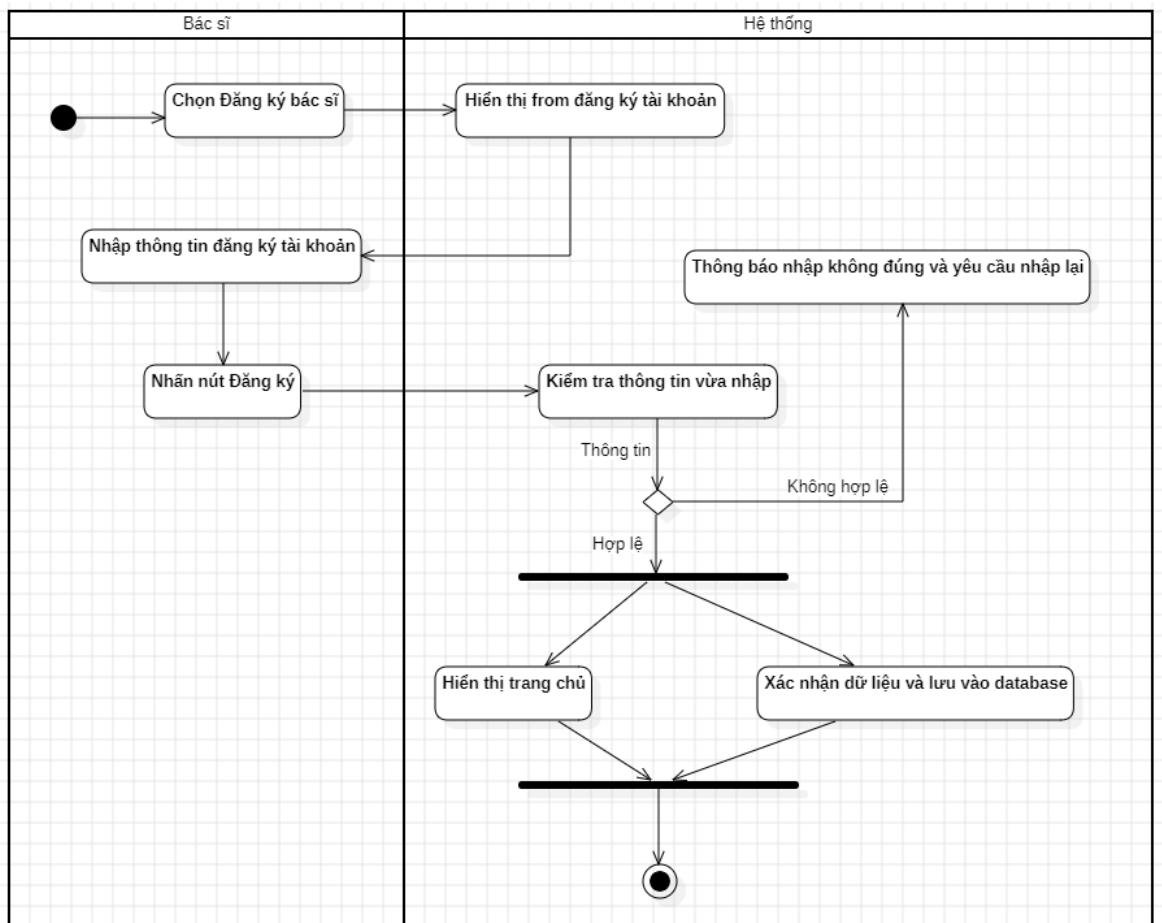
Mối quan hệ	UC29 [Quản lý thông báo]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xóa thông báo thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng nhấn nút xóa 2. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.23. UC32 [Đăng ký bác sĩ]

Tên UC	Đăng ký bác sĩ
Mã UC	UC32
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ đăng ký tài khoản để trở thành bác sĩ trong website
Mối quan hệ	
Tiền điều kiện	<p>Người dùng truy cập vào hệ thống.</p> <p>Người dùng chưa có tài khoản bác sĩ trên hệ thống.</p>
Hậu điều kiện	Người dùng đã đăng ký thành công tài khoản bác sĩ
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào hệ thống. 2. Người dùng chọn “Dành cho bác sĩ” 3. Hệ thống hiển thị form đăng nhập, đăng ký dành cho bác sĩ 4. Người dùng chọn "Đăng ký" 5. Hệ thống hiển thị form đăng ký tài khoản. 6. Người dùng nhập thông tin để đăng ký tài khoản. 7. Người dùng nhấn nút “Đăng ký”. 8. Hệ thống kiểm tra thông tin vừa nhập. 9. Hệ thống xác nhận dữ liệu và lưu vào database 10. Hệ thống hiển thị giao diện trang chủ.
Dòng sự kiện phụ	Hệ thống yêu cầu nhập lại khi nhập sai định dạng dữ liệu

ĐẶC TẢ CHỨC NĂNG

Sơ đồ hoạt động:



Hình 3.12: Activity Diagram [Đăng ký bác sĩ]

3.3.24. UC33 [Quản lý hồ sơ bệnh nhân]

Tên UC	Quản lý hồ sơ bệnh nhân
Mã UC	UC33
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ quản lý hồ sơ bệnh nhân bao gồm xem, thay đổi thông tin bệnh nhân.
Mối quan hệ	UC34 [Sửa đổi thông tin bệnh nhân]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, thay đổi thông tin bệnh nhân thành công
Dòng sự kiện chính	1. Bác sĩ nhấn vào nút “Hồ sơ bệnh nhân” 2. Hệ thống hiển thị danh sách hồ sơ bệnh nhân

ĐẶC TẢ CHỨC NĂNG

	3. Bác sĩ thực hiện thao tác UC34 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.25. UC34 [Sửa đổi thông tin bệnh nhân]

Tên UC	Sửa đổi thông tin bệnh nhân
Mã UC	UC34
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ quản lý lịch hẹn bao gồm xem, thay đổi thông tin bệnh nhân.
Mối quan hệ	UC34 [Sửa đổi thông tin bệnh nhân]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, thay đổi thông tin bệnh nhân thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Bác sĩ nhấn vào nút “Hồ sơ bệnh nhân” 2. Hệ thống hiển thị danh sách hồ sơ bệnh nhân 3. Bác sĩ nhấn vào nút chỉnh sửa 4. Hệ thống hiển thị chi tiết hồ sơ 5. Bác sĩ thực hiện chỉnh sửa hồ sơ 6. Bác sĩ nhấn cập nhật 7. Hệ thống thông báo thành công
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi bác sĩ nhập sai thông tin

3.3.26. UC35 [Quản lý lịch khám]

Tên UC	Quản lý lịch khám
Mã UC	UC35
Tác nhân chính	Bác sĩ

ĐẶC TẢ CHỨC NĂNG

Tóm tắt	Bác sĩ quản lý lịch khám bao gồm xem, chấp nhận, thay đổi lịch khám
Mối quan hệ	UC38 [Hủy lịch khám], UC39 [Chấp nhận lịch khám]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, chấp nhận, thay đổi lịch khám thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Bác sĩ nhấn vào nút “Lịch khám” 2. Hệ thống hiển thị danh sách lịch khám bệnh nhân 3. Bác sĩ thực hiện các thao tác, UC38, UC39 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	
Dòng sự kiện phụ	

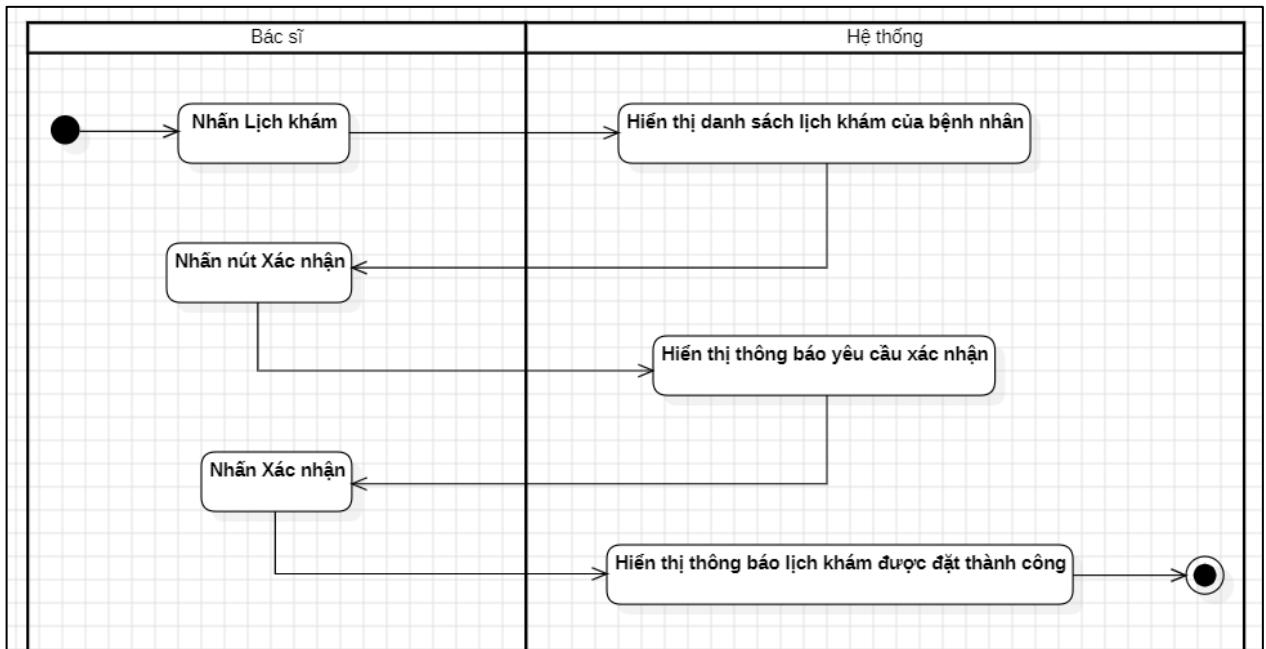
3.3.27. UC38 [Hủy lịch khám]

Tên UC	Hủy lịch khám
Mã UC	UC37
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ hủy lịch khám
Mối quan hệ	UC35 [Quản lý lịch khám]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác hủy lịch khám thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Bác sĩ nhấn vào nút “Lịch khám” 2. Hệ thống hiển thị danh sách lịch khám của bệnh nhân 3. Bác sĩ nhấn vào nút hủy 4. Hệ thống xác nhận hủy lịch 5. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.28. UC39 [Chấp nhận lịch khám]

Tên UC	Chấp nhận lịch khám
Mã UC	UC39
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ chấp nhận khám
Mối quan hệ	UC35 [Quản lý lịch khám]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác chấp nhận lịch khám thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Bác sĩ nhấn vào nút “Lịch khám” 2. Hệ thống hiển thị danh sách lịch khám của bệnh nhân 3. Bác sĩ nhấn vào nút “xác nhận” 4. Hệ thống xác nhận đặt lịch 5. Hệ thống thông báo thành công
Dòng sự kiện phụ	

Sơ đồ hoạt động:



Hình 3.13: Activity Diagram [Chấp nhận lịch khám]

ĐẶC TẢ CHỨC NĂNG

3.3.29. UC41 [Gọi điện với bệnh nhân]

Tên UC	Gọi điện với bệnh nhân
Mã UC	UC41
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ gọi điện tư vấn với bệnh nhân
Mối quan hệ	UC35 [Quản lý lịch khám]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống Bác sĩ đã chấp nhận lịch khám
Hậu điều kiện	Bác sĩ thực hiện cuộc gọi đến bác sĩ thành công
Dòng sự kiện chính	1. Bác sĩ nhấn vào nút “gọi điện” 2. Hệ thống hiển thị màn hình cuộc gọi 3. Hệ thống kết nối máy đến bệnh nhân và chờ bệnh nhân chấp nhận 4. Hệ thống hiển thị màn hình cuộc gọi đến bệnh nhân 5. Bác sĩ thực hiện cuộc gọi đến bệnh nhân
Dòng sự kiện phụ	

3.3.30. UC42 [Quản lý thông báo]

Tên UC	Quản lý thông báo
Mã UC	UC42
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ quản lý thông báo bao gồm xem, gửi thông báo
Mối quan hệ	UC43 [Xem chi tiết thông báo]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xem, gửi thông báo thành công
Dòng sự kiện chính	1. Bác sĩ nhấn vào nút “thông báo” 2. Hệ thống hiển thị danh sách thông báo 3. Bác sĩ thực hiện các thao tác UC43 4. Hệ thống thông báo thành công

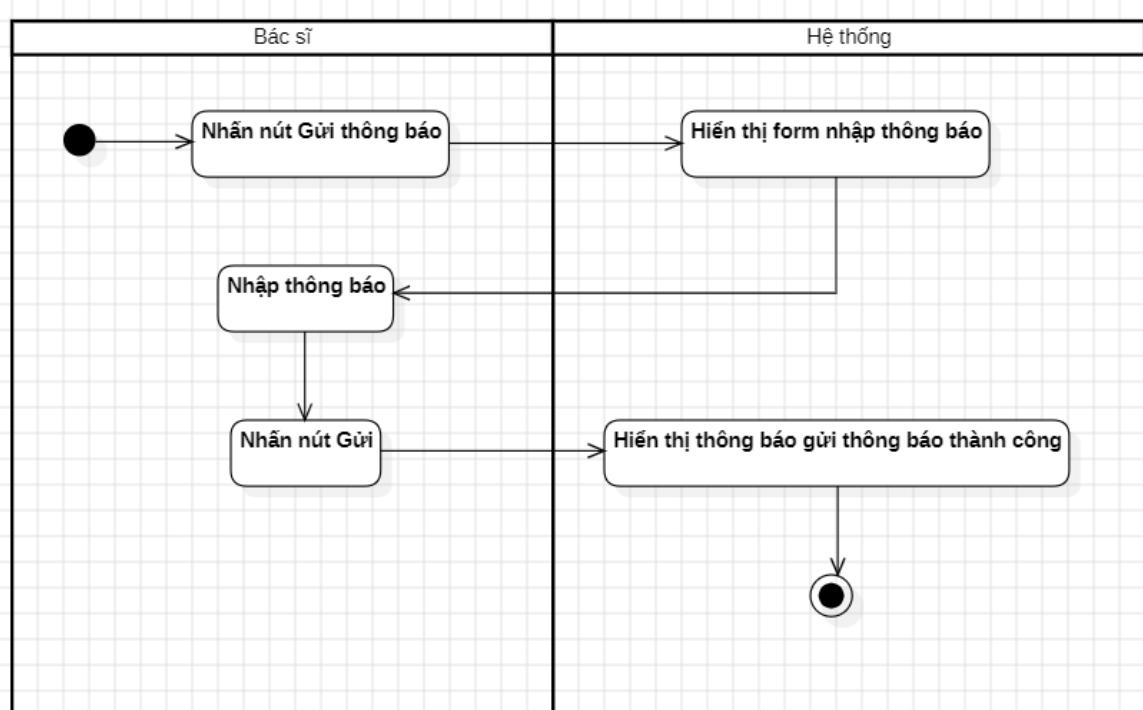
ĐẶC TẢ CHỨC NĂNG

Dòng sự kiện chính	
Dòng sự kiện phụ	

3.3.31. UC43 [Xem chi tiết thông báo]

Tên UC	Xem chi tiết thông báo
Mã UC	UC43
Tác nhân chính	Bác sĩ
Tóm tắt	Bác sĩ gửi thông báo đến bệnh nhân
Mối quan hệ	UC42 [Quản lý thông báo]
Tiền điều kiện	Bác sĩ đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác gửi thông báo thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> Bác sĩ nhấp vào nút “Xem chi tiết” Hệ thống hiển thị form chi tiết thông báo
Dòng sự kiện phụ	

Sơ đồ hoạt động:



Hình 3.14: Activity Diagram [Gửi thông báo]

3.3.32. UC46 [Quản lý tài khoản người dùng]

Tên UC	Quản lý tài khoản người dùng
Mã UC	UC46
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý tài khoản người dùng bao gồm xem, sửa, xóa, khóa tài khoản và xác nhận đơn đăng ký bác sĩ.
Mối quan hệ	UC48 [Sửa thông tin tài khoản], UC49 [Khóa tài khoản], UC50 [Xóa tài khoản], UC51 [Xác nhận đăng ký bác sĩ]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, sửa, xóa, khóa tài khoản và xác nhận đơn đăng ký thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý tài khoản” 2. Hệ thống hiển thị danh sách tài khoản người dùng 3. Bác sĩ thực hiện các thao tác UC48, UC49, UC50, UC51 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.33. UC48 [Sửa thông tin tài khoản]

Tên UC	Sửa thông tin tài khoản
Mã UC	UC48
Tác nhân chính	Quản lý
Tóm tắt	Quản lý sửa thông tin tài khoản
Mối quan hệ	UC46 [Quản lý tài khoản người dùng]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác thay đổi thông tin người dùng thành công.
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Tài khoản người dùng” 2. Hệ thống hiển thị danh sách tài khoản người dùng 3. Quản lý nhấn vào nút chỉnh sửa 4. Hệ thống hiển thị chi tiết tài khoản

ĐẶC TẢ CHỨC NĂNG

	5. Quản lý thực hiện chỉnh sửa tài khoản 6. Quản lý nhấn cập nhật 7. Hệ thống thông báo thành công
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi bác sĩ nhập sai thông tin

3.3.34. UC49 [Khóa tài khoản]

Tên UC	Khóa tài khoản
Mã UC	UC49
Tác nhân chính	Quản lý
Tóm tắt	Quản lý khóa tài khoản khi người dùng vi phạm điều khoản
Mối quan hệ	UC46 [Quản lý tài khoản người dùng]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác khóa tài khoản người dùng thành công.
Dòng sự kiện chính	1. Quản lý nhấn vào nút “Tài khoản người dùng” 2. Hệ thống hiển thị danh sách tài khoản người dùng 3. Quản lý nhấn vào nút “Khóa tài khoản” 4. Hệ thống yêu cầu xác nhận lại 5. Quản lý nhấn “Xác nhận” 6. Hệ thống thông báo thành công
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi bác sĩ nhập sai thông tin

3.3.35. UC50 [Xóa tài khoản]

Tên UC	Xóa tài khoản
Mã UC	UC50
Tác nhân chính	Quản lý
Tóm tắt	Quản lý xóa tài khoản khi người dùng yêu cầu
Mối quan hệ	UC46 [Quản lý tài khoản người dùng]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống

ĐẶC TẢ CHỨC NĂNG

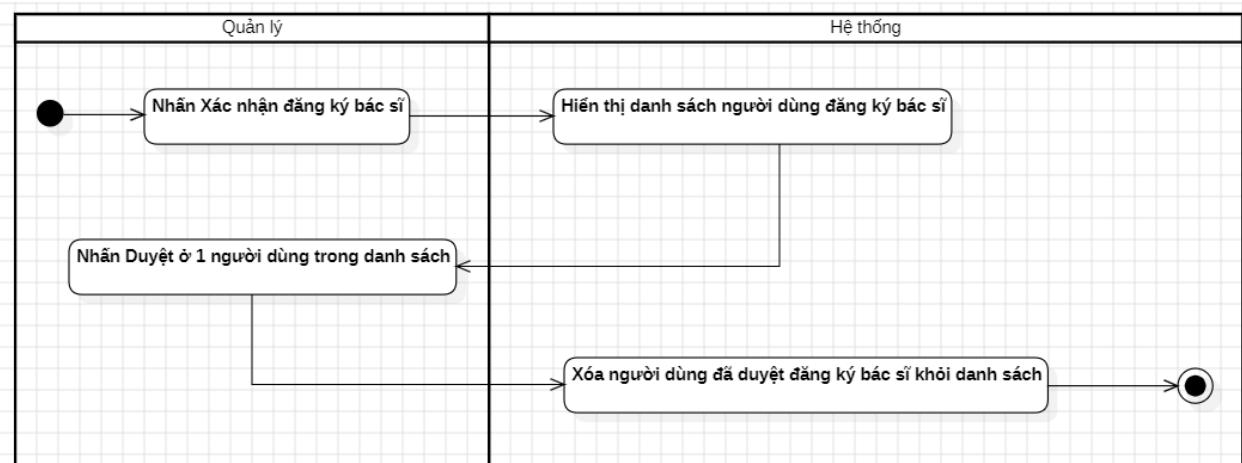
Hậu điều kiện	Thực hiện các thao tác xóa tài khoản người dùng thành công.
Dòng sự kiện chính	1. Quản lý nhấn vào nút “Tài khoản người dùng” 2. Hệ thống hiển thị danh sách tài khoản người dùng 3. Quản lý nhấn vào nút “Xóa tài khoản” 4. Hệ thống yêu cầu xác nhận lại 5. Quản lý nhấn “Xác nhận” 6. Hệ thống thông báo thành công
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi bác sĩ nhập sai thông tin

3.3.36. UC51 [Xác nhận đăng ký bác sĩ]

Tên UC	Xác nhận đăng ký bác sĩ
Mã UC	UC51
Tác nhân chính	Quản lý
Tóm tắt	Quản lý xác nhận đăng ký tài khoản của bác sĩ
Mối quan hệ	UC46 [Quản lý tài khoản người dùng]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác xóa tài khoản người dùng thành công.
Dòng sự kiện chính	1. Quản lý nhấn vào nút “quản lý hồ sơ bác sĩ” 2. Hệ thống hiển thị danh sách đơn chờ 3. Quản lý nhấn vào nút “Duyệt” 4. Hiển thị form xác nhận 5. Quản lý nhập số tiền khám cho bác sĩ 6. Quản lý bấm “xác nhận” 7. Hệ thống thông báo thành công
Dòng sự kiện phụ	

Sơ đồ hoạt động:

ĐẶC TẢ CHỨC NĂNG



Hình 3.15: Activity Diagram [Xác nhận đăng ký bác sĩ]

3.3.37. UC52 [Quản lý hồ sơ y tế]

Tên UC	Quản lý hồ sơ y tế
Mã UC	UC52
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý Hồ sơ y tế người dùng bao gồm xem, sửa hồ sơ.
Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, sửa hồ sơ thành công
Dòng sự kiện chính	1. Quản lý nhấn vào nút “Quản lý tài khoản” 2. Hệ thống hiển thị danh sách tài khoản người dùng 3. Quản lý thực hiện các thao tác xem, sửa hồ sơ 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.38. UC57 [Quản lý lịch khám]

Tên UC	Quản lý lịch khám
Mã UC	UC57
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý lịch khám người dùng bao gồm xem, sửa lịch khám.

ĐẶC TẢ CHỨC NĂNG

Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, sửa lịch khám thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý lịch khám” 2. Hệ thống hiển thị danh sách lịch khám người dùng 3. Quản lý thực hiện các thao tác xem, sửa lịch khám 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.39. UC59 [Quản lý lương]

Tên UC	Quản lý lương
Mã UC	UC59
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý lương người dùng bao gồm xem, thêm, sửa, xóa lương
Mối quan hệ	UC61 [Thêm lương], UC62 [Xóa lương], UC63 [Sửa lương]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, thêm, sửa, xóa lương thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý lương” 2. Hệ thống hiển thị danh sách bác sĩ 3. Quản lý thực hiện các thao tác UC61, UC62, UC63 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.40. UC61 [Thêm lương]

Tên UC	Thêm lương
Mã UC	UC61
Tác nhân chính	Quản lý

ĐẶC TẢ CHỨC NĂNG

Tóm tắt	Quản lý Thêm lương cho bác sĩ
Mối quan hệ	UC59 [Quản lý lương]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác thay đổi thông tin người dùng thành công.
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý lương” 2. Hệ thống hiển thị danh sách bác sĩ 3. Quản lý nhấn vào nút Thêm 4. Hệ thống hiển thị chi tiết tài khoản 5. Quản lý thực hiện Thêm lương cho tài khoản 6. Quản lý nhấn tạo 7. Hệ thống thông báo thành công
Dòng sự kiện phụ	Hệ thống thông báo lỗi khi bác sĩ nhập sai thông tin

3.3.41. UC63 [Sửa lương]

Tên UC	Sửa lương
Mã UC	UC63
Tác nhân chính	Quản lý
Tóm tắt	Quản lý sửa lương cho bác sĩ
Mối quan hệ	UC59 [Quản lý lương]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác thay đổi thông tin người dùng thành công.
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý lương” 2. Hệ thống hiển thị danh sách bác sĩ 3. Quản lý nhấn vào nút sửa 4. Hệ thống hiển thị chi tiết tài khoản 5. Quản lý thực hiện sửa lương cho tài khoản 6. Quản lý nhấn tạo 7. Hệ thống thông báo thành công

ĐẶC TẢ CHỨC NĂNG

Dòng sự kiện phụ	Hệ thống thông báo lỗi khi bác sĩ nhập sai thông tin
------------------	--

3.3.42. UC64 [Quản lý thanh toán]

Tên UC	Quản lý thanh toán
Mã UC	UC64
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý thanh toán người dùng bao gồm xem, hoàn phí, thống kê thanh toán của người dùng
Mối quan hệ	UC65 [Hoàn phí khám bệnh], UC67 [Quản lý thống kê]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, hoàn phí, thống kê thanh toán thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý thanh toán” 2. Hệ thống hiển thị danh sách giao dịch 3. Quản lý thực hiện các thao tác UC65, UC66, UC67 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.43. UC65 [Hoàn phí khám bệnh]

Tên UC	Hoàn phí khám bệnh
Mã UC	UC65
Tác nhân chính	Quản lý
Tóm tắt	Quản lý hoàn phí khám bệnh cho người dùng.
Mối quan hệ	UC64 [Quản lý thanh toán]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao hoàn phí thanh toán thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý thanh toán” 2. Hệ thống hiển thị danh sách giao dịch 3. Quản lý nhấn nút hoàn phí

ĐẶC TẢ CHỨC NĂNG

	4. Hệ thống hiển thị form hoàn phí 5. Quản lý nhập thông tin hoàn phí 6. Quản lý nhấn Xác nhận 7. Hệ thống lưu giao dịch 8. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.44. UC70 [Quản lý thông tin bệnh viện]

Tên UC	Quản lý thông tin bệnh viện
Mã UC	UC70
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý thông tin bệnh viện bao gồm xem, thêm, sửa, xóa thông tin bệnh viện
Mối quan hệ	UC71 [Thêm thông tin bệnh viện], UC72 [Sửa thông tin bệnh viện], UC73 [Xóa thông tin bệnh viện]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, thêm, sửa, xóa thông tin bệnh viện thành công
Dòng sự kiện chính	1. Quản lý nhấn vào nút “Quản lý bệnh viện” 2. Hệ thống hiển thị danh sách bệnh viện 3. Quản lý thực hiện các thao tác UC71, UC72, UC73 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.45. UC74 [Quản lý danh mục bài viết]

Tên UC	Quản lý danh mục bài viết
Mã UC	UC74
Tác nhân chính	Quản lý

ĐẶC TẢ CHỨC NĂNG

Tóm tắt	Quản lý quản lý danh mục bài viết bao gồm xem, thêm, sửa, xóa danh mục bài viết
Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, thêm, sửa, xóa danh mục bài viết thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý danh mục bài viết” 2. Hệ thống hiển thị danh sách danh mục bài viết 3. Quản lý thực hiện các thao tác thêm, xóa, sửa danh mục bài viết 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.46. UC75 [Quản lý bài viết]

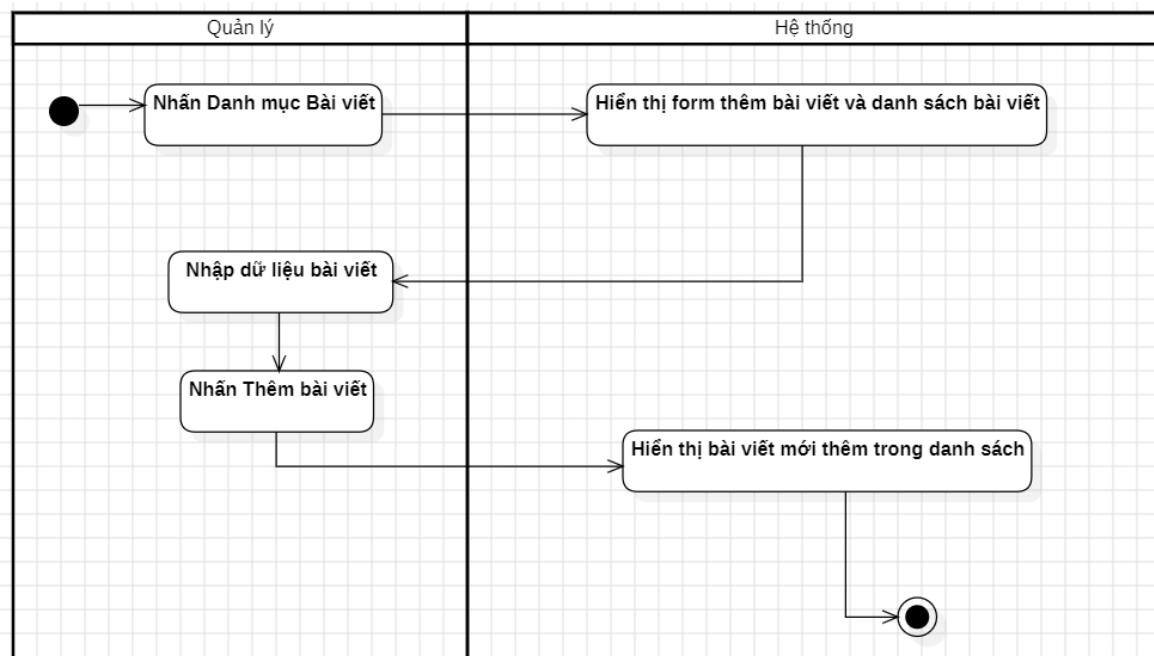
Tên UC	Quản lý bài viết
Mã UC	UC75
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý lương người dùng bao gồm xem, thêm, sửa, xóa lương
Mối quan hệ	UC76 [Thêm bài viết], UC77 [Sửa bài viết], UC78 [Xóa bài viết]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, thêm, sửa, xóa bài viết thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý bài viết” 2. Hệ thống hiển thị danh sách bác sĩ 3. Quản lý thực hiện các thao tác UC76, UC77, UC78 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.47. UC76 [Thêm bài viết]

Tên UC	Thêm bài viết
Mã UC	UC76
Tác nhân chính	Quản lý
Tóm tắt	Quản lý thêm bài viết
Mối quan hệ	UC75 [Quản lý bài viết]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao tác thêm bài viết thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý bài viết” 2. Hệ thống hiển thị danh sách bác sĩ 3. Quản lý nhấn nút thêm 4. Hệ thống hiển thị giao diện thêm bài viết 5. Quản lý chọn danh mục bài viết cho bài viết 6. Quản lý nhập bài viết 7. Quản lý nhấn nút đăng 8. Hệ thống lưu vào data 9. Hệ thống thông báo thành công
Dòng sự kiện phụ	

Sơ đồ hoạt động:

ĐẶC TẢ CHỨC NĂNG



Hình 3.16: Activity Diagram [Thêm bài viết]

3.3.48. UC79 [Quản lý danh mục chuyên ngành]

Tên UC	Quản lý danh mục chuyên ngành
Mã UC	UC79
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý lương người dùng bao gồm xem, thêm, sửa, xóa danh mục chuyên ngành
Mối quan hệ	UC80 [Thêm chuyên ngành], UC81 [Sửa chuyên ngành], UC82 [Xóa chuyên ngành]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, thêm, sửa, xóa danh mục chuyên ngành thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> Quản lý nhấn vào nút “Quản lý danh mục chuyên ngành” Hệ thống hiển thị danh sách danh mục chuyên ngành Quản lý thực hiện các thao tác UC80, UC81, UC82 Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.49. UC83 [Quản lý danh mục khoa bệnh]

Tên UC	Quản lý danh mục khoa bệnh
Mã UC	UC83
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý danh mục khoa bệnh người dùng bao gồm xem, thêm, sửa, xóa danh mục khoa bệnh
Mối quan hệ	UC84 [Thêm khoa bệnh], UC85 [Sửa chuyên ngành], UC86 [Xóa chuyên ngành]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem, thêm, sửa, xóa danh mục khoa bệnh thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý danh mục khoa bệnh” 2. Hệ thống hiển thị danh sách danh mục khoa bệnh 3. Quản lý thực hiện các thao tác UC84, UC85, UC86 4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.50. UC87 [Quản lý lịch sử cuộc gọi]

Tên UC	Quản lý lịch sử cuộc gọi
Mã UC	UC87
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý lịch sử cuộc gọi người dùng bao gồm xem, xóa lịch sử cuộc gọi
Mối quan hệ	UC89 [Xóa cuộc gọi]
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem xóa lương thành công
Dòng sự kiện chính	<ol style="list-style-type: none"> 1. Quản lý nhấn vào nút “Quản lý lịch sử cuộc gọi” 2. Hệ thống hiển thị danh sách lịch sử cuộc gọi 3. Quản lý thực hiện các thao tác UC89

ĐẶC TẢ CHỨC NĂNG

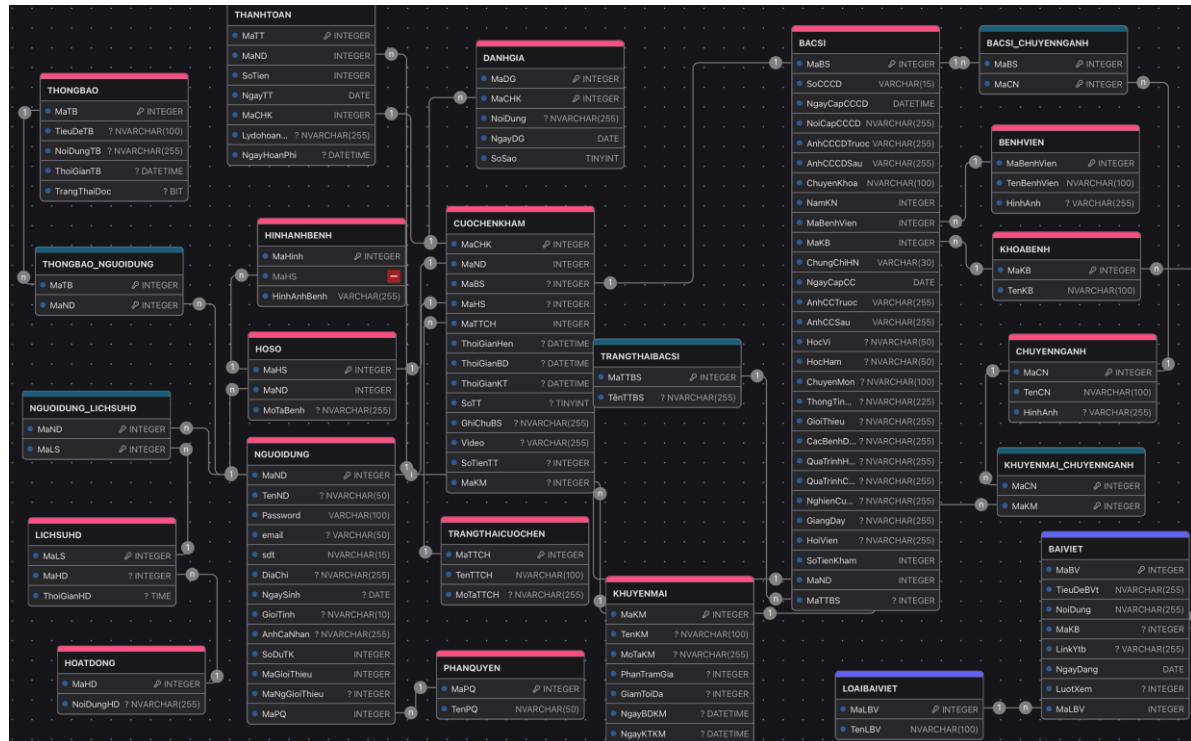
	4. Hệ thống thông báo thành công
Dòng sự kiện phụ	

3.3.51. UC90 [Quản lý đánh giá phản hồi bệnh nhân]

Tên UC	Quản lý đánh giá phản hồi bệnh nhân
Mã UC	UC90
Tác nhân chính	Quản lý
Tóm tắt	Quản lý quản lý đánh giá phản hồi bệnh nhân người dùng bao gồm xem đánh giá phản hồi bệnh nhân
Mối quan hệ	
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Thực hiện các thao xem đánh giá phản hồi bệnh nhân thành công
Dòng sự kiện chính	1. Quản lý nhấn vào nút “Quản lý đánh giá phản hồi bệnh nhân” 2. Hệ thống hiển thị danh sách đánh giá phản hồi bệnh nhân
Dòng sự kiện phụ	

CHƯƠNG 4: MÔ HÌNH HÓA YÊU CẦU VÀ API

4.1. Thiết kế ERD tổng quát của hệ thống



Hình 4.1: Sơ đồ ERD

4.2. Triển khai CSDL trên MSSQL Server

Tạo bảng cơ sở dữ liệu

```
use master

if exists (select * from sysdatabases where name =
'VOVBACSI')

    drop database VOVBACSI

go

create database VOVBACSI

go

use VOVBACSI

go

CREATE TABLE [PHANQUYEN] (
    [MaPQ] INTEGER NOT NULL IDENTITY,
    [TenPQ] NVARCHAR(50) NOT NULL,
    PRIMARY KEY ([MaPQ])
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
) ;  
GO  
  
CREATE TABLE [NGUOIDUNG] (  
    [MaND] INTEGER NOT NULL IDENTITY,  
    [TenND] NVARCHAR(50),  
    [Password] VARCHAR(50) NOT NULL,  
    [Email] VARCHAR(50),  
    [sdt] VARCHAR(15) NOT NULL,  
    [DiaChi] NVARCHAR(100),  
    [NgaySinh] DATE,  
    [GioiTinh] NVARCHAR(5),  
    [AnhCaNhan] VARCHAR(50),  
    [SoDuTK] INTEGER NOT NULL,  
    [MaGioiThieu] INTEGER NOT NULL,  
    [MaNgGioiThieu] INTEGER,  
    [MaPQ] INTEGER NOT NULL,  
    PRIMARY KEY ([MaND])  
);  
GO  
  
CREATE TABLE [KHOABENH] (  
    [MaKB] INTEGER NOT NULL IDENTITY,  
    [TenKB] NVARCHAR(50) NOT NULL,  
    [MoTa] NVARCHAR(500),  
    PRIMARY KEY ([MaKB])  
);  
GO  
CREATE TABLE [BENHVIEN] (  
    [MaBenzVien] INTEGER NOT NULL IDENTITY,  
    [TenBenzVien] NVARCHAR(150) NOT NULL,
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
[HinhAnh] VARCHAR(150) NOT NULL,  
PRIMARY KEY ([MaBenhVien])  
);  
GO  
  
CREATE TABLE [TRANGTHAIBACSI] (  
[MaTTBS] INTEGER NOT NULL IDENTITY UNIQUE,  
[TênTTBS] NVARCHAR,  
PRIMARY KEY ([MaTTBS])  
);  
GO  
  
CREATE TABLE [BACSI] (  
[MaBS] INTEGER NOT NULL IDENTITY,  
[SoCCCD] VARCHAR(20) NOT NULL,  
[NgayCapCCCD] DATETIME NOT NULL,  
[NoiCapCCCD] NVARCHAR(50) NOT NULL,  
[AnhCCCDTruoc] VARCHAR(150) NOT NULL,  
[AnhCCCDsau] VARCHAR(150) NOT NULL,  
[ChuyenKhoa] NVARCHAR(150) NOT NULL,  
[NamKN] INTEGER NOT NULL,  
[MaBenzVien] INTEGER NOT NULL,  
[MaKB] INTEGER NOT NULL,  
[ChungChiHN] NVARCHAR(150) NOT NULL,  
[NgayCapCC] DATE NOT NULL,  
[AnhCCTruoc] VARCHAR(150) NOT NULL,  
[AnhCCSau] VARCHAR(150) NOT NULL,  
[HocVi] NVARCHAR(50),  
[HocHam] NVARCHAR(50),  
[ChuyenMon] NVARCHAR(50),  
[ThongTinBangCap] NVARCHAR(max),
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
[GioiThieu] NVARCHAR(max),  
[CacBenhDieuTri] NVARCHAR(max),  
[QuaTrinhHoc] NVARCHAR(max),  
[QuaTrinhCongTac] NVARCHAR(max),  
[NghienCuuKH] NVARCHAR(max),  
[GiangDay] NVARCHAR(max),  
[HoiVien] NVARCHAR(max),  
[SoTienKham] INTEGER NOT NULL,  
[MaND] INTEGER NOT NULL,  
[MaTTBS] INTEGER NOT NULL,  
PRIMARY KEY ( [MaBS] )  
);  
GO  
  
CREATE TABLE [CHUYENNGANH] (  
    [MaCN] INTEGER NOT NULL IDENTITY,  
    [TenCN] NVARCHAR(100) NOT NULL,  
    [HinhAnh] VARCHAR(150) NOT NULL,  
    PRIMARY KEY ( [MaCN] )  
);  
GO  
  
CREATE TABLE [LOAIBAIVIET] (  
    [MaLBV] INTEGER NOT NULL IDENTITY,  
    [TenLBV] NVARCHAR(100) NOT NULL,  
    PRIMARY KEY ( [MaLBV] )  
);  
GO  
  
CREATE TABLE [BAIVIET] (  
    [MaBV] INTEGER NOT NULL IDENTITY,
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
[TieuDeBV] NVARCHAR(100) NOT NULL,  
[NoiDung] NVARCHAR(max) NOT NULL,  
[MaKB] INTEGER,  
[LinkYtb] VARCHAR(100),  
[NgayDang] DATE NOT NULL,  
[MaLBV] INTEGER NOT NULL,  
[LuotXem] INTEGER,  
PRIMARY KEY ([MaBV])  
);  
GO  
  
CREATE TABLE [HINHANHBENH] (  
[MaHinh] INTEGER NOT NULL IDENTITY,  
[MaHS] INTEGER NOT NULL,  
[HinhAnhBenh] VARCHAR(100) NOT NULL,  
PRIMARY KEY([MaHinh])  
);  
GO  
  
CREATE TABLE [HOSO] (  
[MaHS] INTEGER NOT NULL IDENTITY,  
[MaND] INTEGER NOT NULL,  
[MoTaBenh] NVARCHAR(max),  
PRIMARY KEY ([MaHS])  
);  
GO  
  
CREATE TABLE [TRANGTHAICUOCHE] (  
[MaTTCH] INTEGER NOT NULL IDENTITY,  
[TenTTCH] NVARCHAR(50) NOT NULL,  
[MoTaTTCH] NVARCHAR(200),
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
PRIMARY KEY ( [MaTTCH] )  
);  
GO  
  
CREATE TABLE [CUOCHECHENKHAM] (  
    [MaCHK] INTEGER NOT NULL IDENTITY,  
    [MaND] INTEGER NOT NULL,  
    [MaBS] INTEGER,  
    [MaHS] INTEGER,  
    [MaTTCH] INTEGER NOT NULL,  
    [ThoiGianHen] DATETIME,  
    [ThoiGianBD] DATETIME,  
    [ThoiGianKT] DATETIME,  
    [SoTT] TINYINT,  
    [GhiChuBS] NVARCHAR(200),  
    [Video] VARCHAR(200),  
    [SoTienTT] INTEGER,  
    [MaKM] INTEGER,  
    PRIMARY KEY ( [MaCHK] )  
);  
GO  
  
CREATE TABLE [KHUYENMAI] (  
    [MaKM] INTEGER NOT NULL IDENTITY,  
    [TenKM] NVARCHAR(100),  
    [MoTaKM] NVARCHAR(200),  
    [PhanTramGia] INTEGER,  
    [GiamToiDa] INTEGER,  
    [NgayBDKM] DATETIME,  
    [NgayKTkm] DATETIME,  
    PRIMARY KEY ( [MaKM] )
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
) ;  
GO  
  
CREATE TABLE [DANHGIA] (  
    [MaDG] INTEGER NOT NULL IDENTITY,  
    [MaCHK] INTEGER NOT NULL,  
    [NoiDung] NVARCHAR(200),  
    [NgayDG] DATE NOT NULL,  
    [SoSao] TINYINT NOT NULL,  
    PRIMARY KEY ([MaDG])  
) ;  
GO  
  
CREATE TABLE [THANHTOAN] (  
    [MaTT] INTEGER NOT NULL IDENTITY,  
    [MaND] INTEGER NOT NULL,  
    [SoTien] INTEGER NOT NULL,  
    [NgayTT] DATE NOT NULL,  
    [MaCHK] INTEGER NOT NULL,  
    PRIMARY KEY ([MaTT])  
) ;  
GO  
  
CREATE TABLE [THONGBAO] (  
    [MaTB] INTEGER NOT NULL IDENTITY,  
    [TieuDeTB] NVARCHAR(100),  
    [NoiDungTB] NVARCHAR(500),  
    [ThoiGianTB] DATETIME,  
    [TrangThaiDoc] BIT,  
    PRIMARY KEY ([MaTB])  
) ;
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
GO

CREATE TABLE [HOATDONG] (
    [MaHD] INTEGER NOT NULL IDENTITY,
    [NoiDungHD] NVARCHAR(100),
    PRIMARY KEY ([MaHD])
);

GO

CREATE TABLE [LICHSUHD] (
    [MaLS] INTEGER NOT NULL IDENTITY,
    [MaHD] INTEGER,
    [ThoiGianHD] TIME,
    PRIMARY KEY ([MaLS])
);

GO

CREATE TABLE [BACSI_CHUYENNGANH] (
    [MaBS] INTEGER NOT NULL,
    [MaCN] INTEGER NOT NULL,
    PRIMARY KEY ([MaBS], [MaCN])
);

GO

CREATE TABLE [KHUYENMAI_CHUYENNGANH] (
    [MaCN] INTEGER NOT NULL,
    [MaKM] INTEGER NOT NULL,
    PRIMARY KEY ([MaCN], [MaKM])
);

GO
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
CREATE TABLE [THONGBAO_NGUOIDUNG] (
    [MaTB] INTEGER NOT NULL,
    [MaND] INTEGER NOT NULL,
    PRIMARY KEY ([MaTB], [MaND])
);

GO

CREATE TABLE [NGUOIDUNG_LICHSUHD] (
    [MaND] INTEGER NOT NULL,
    [MaLS] INTEGER NOT NULL,
    PRIMARY KEY ([MaND], [MaLS])
);

GO

-- Khóa ngoại được điều chỉnh, tránh tạo khóa ngoại trên
-- chính bảng chứa khóa ngoại

ALTER TABLE [NGUOIDUNG]
ADD CONSTRAINT FK_NGUOIDUNG_PHANQUYEN FOREIGN
KEY ([MaPQ]) REFERENCES [PHANQUYEN] ([MaPQ]);

ALTER TABLE [BACSI]
ADD CONSTRAINT FK_BACSI_BENHVIEN FOREIGN
KEY ([MaBenzVien]) REFERENCES [BENHVIEN] ([MaBenzVien]);

ALTER TABLE [BACSI]
ADD CONSTRAINT FK_BACSI_KHOABENH FOREIGN KEY ([MaKB])
REFERENCES [KHOABENH] ([MaKB]);

ALTER TABLE [BACSI]
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
ADD      CONSTRAINT      FK_BACSI_TRANGTHAIBACSI      FOREIGN  
KEY ( [MaTTBS] )  REFERENCES  [TRANGTHAIBACSI] ( [MaTTBS] );  
  
ALTER TABLE  [BAIVIET]  
ADD      CONSTRAINT      FK_BAIVIET_LOAIBAIVIET      FOREIGN  
KEY ( [MaLBV] )  REFERENCES  [LOAIBAIVIET] ( [MaLBV] );  
  
ALTER TABLE  [BAIVIET]  
ADD CONSTRAINT  FK_BAIVIET_KHOABENH  FOREIGN KEY ( [MaKB] )  
REFERENCES  [KHOABENH] ( [MaKB] );  
  
ALTER TABLE  [CUOCHENKHAM]  
ADD      CONSTRAINT      FK_CUOCHENKHAM_NGUOIDUNG      FOREIGN  
KEY ( [MaND] )  REFERENCES  [NGUOIDUNG] ( [MaND] );  
  
ALTER TABLE  [CUOCHENKHAM]  
ADD CONSTRAINT  FK_CUOCHENKHAM_BACSI  FOREIGN KEY ( [MaBS] )  
REFERENCES  [BACSI] ( [MaBS] );  
  
ALTER TABLE  [CUOCHENKHAM]  
ADD CONSTRAINT  FK_CUOCHENKHAM_HOSO  FOREIGN KEY ( [MaHS] )  
REFERENCES  [HOSO] ( [MaHS] );  
  
ALTER TABLE  [CUOCHENKHAM]  
ADD CONSTRAINT  FK_CUOCHENKHAM_TRANGTHAICUOCHEN  FOREIGN  
KEY ( [MaTTCH] )  REFERENCES  [TRANGTHAICUOCHEN] ( [MaTTCH] );  
  
ALTER TABLE  [HOSO]  
ADD  CONSTRAINT  FK_HOSO_NGUOIDUNG  FOREIGN  KEY ( [MaND] )  
REFERENCES  [NGUOIDUNG] ( [MaND] );
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
ALTER TABLE [HINHANHBENH]
ADD CONSTRAINT FK_HINHANHBENH_HOSO FOREIGN KEY ([MaHS])
REFERENCES [HOSO] ([MaHS]);
```



```
ALTER TABLE [DANHGIA]
ADD CONSTRAINT FK_DANHGIA_CUOCHENKHAM FOREIGN
KEY ([MaCHK]) REFERENCES [CUOCHENKHAM] ([MaCHK]);
```



```
ALTER TABLE [THANHTOAN]
ADD CONSTRAINT FK_THANHTOAN_NGUOIDUNG FOREIGN
KEY ([MaND]) REFERENCES [NGUOIDUNG] ([MaND]);
```



```
ALTER TABLE [THANHTOAN]
ADD CONSTRAINT FK_THANHTOAN_CUOCHENKHAM FOREIGN
KEY ([MaCHK]) REFERENCES [CUOCHENKHAM] ([MaCHK]);
```



```
ALTER TABLE [BACSI_CHUYENNGANH]
ADD CONSTRAINT FK_BACSI_CHUYENNGANH_BACSI FOREIGN
KEY ([MaBS]) REFERENCES [BACSI] ([MaBS));
```



```
ALTER TABLE [BACSI_CHUYENNGANH]
ADD CONSTRAINT FK_BACSI_CHUYENNGANH_CHUYENNGANH FOREIGN
KEY ([MaCN]) REFERENCES [CHUYENNGANH] ([MaCN));
```



```
ALTER TABLE [KHUYENMAI_CHUYENNGANH]
ADD CONSTRAINT FK_KHUYENMAI_CHUYENNGANH_KHUYENMAI FOREIGN
KEY ([MaKM]) REFERENCES [KHUYENMAI] ([MaKM));
```



```
ALTER TABLE [KHUYENMAI_CHUYENNGANH]
ADD CONSTRAINT FK_KHUYENMAI_CHUYENNGANH_CHUYENNGANH FOREIGN
KEY ([MaCN]) REFERENCES [CHUYENNGANH] ([MaCN));
```

```
ALTER TABLE [THONGBAO_NGUOIDUNG]
ADD CONSTRAINT FK_THONGBAO_NGUOIDUNG_THONGBAO FOREIGN
KEY ([MaTB]) REFERENCES [THONGBAO] ([MaTB]);  
  
ALTER TABLE [THONGBAO_NGUOIDUNG]
ADD CONSTRAINT FK_THONGBAO_NGUOIDUNG_NGUOIDUNG FOREIGN
KEY ([MaND]) REFERENCES [NGUOIDUNG] ([MaND]);  
  
ALTER TABLE [NGUOIDUNG_LICHSUHD]
ADD CONSTRAINT FK_NGUOIDUNG_LICHSUHD_NGUOIDUNG FOREIGN
KEY ([MaND]) REFERENCES [NGUOIDUNG] ([MaND]);  
  
ALTER TABLE [NGUOIDUNG_LICHSUHD]
ADD CONSTRAINT FK_NGUOIDUNG_LICHSUHD_LICHSUHD FOREIGN
KEY ([MaLS]) REFERENCES [LICHсуHD] ([MaLS]);
```

4.3. Nhập dữ liệu mẫu

```
----- Nhập dữ liệu mẫu -----  
  
-- Dữ liệu mẫu cho bảng PHANQUYEN  
INSERT INTO PHANQUYEN (TenPQ)  
VALUES  
(N'Quản trị viên'),  
(N'Bệnh nhân'),  
(N'Bác sĩ');  
  
-- Dữ liệu mẫu cho bảng NGUOIDUNG
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
INSERT INTO NGUOIDUNG (TenND, Password, Email, sdt,
DiaChi, NgaySinh, GioiTinh, AnhCaNhan, SoDuTK,
MaGioiThieu, MaNgGioiThieu, MaPQ)
VALUES
(N'Nguyễn Văn Admin', '123456', 'vana@gmail.com',
N'0909123456', N'123 Lê Lợi, Quận 1, TP. HCM', '1990-
01-15', N'Nam', N'/images/van_a.jpg', 1000000, 101,
NULL, 1),
(N'Trần Thị Bệnh Nhân', 'abcdef', 'thib@gmail.com',
N'0909876543', N'456 Trường Chinh, Quận Tân Bình, TP.
HCM', '1985-05-20', N'Nữ', N'/images/thi_b.jpg',
1500000, 102, NULL, 2),
(N'Phạm Minh Bác Sĩ', '654321', 'minhc@gmail.com',
N'0909345678', N'789 Nguyễn Huệ, Quận 3, TP. HCM',
'1995-10-10', N'Nam', N'/images/minh_c.jpg', 2000000,
103, NULL, 3);

-- Dữ liệu mẫu cho bảng KHOABENH
INSERT INTO KHOABENH (TenKB, MoTa)
VALUES
(N'Nội tổng quát', N'Chuyên khám và điều trị các bệnh
lý nội khoa'),
(N'Nhi khoa', N'Khám và điều trị các bệnh về trẻ em'),
(N'Tai mũi họng', N'Chuyên về các bệnh tai, mũi và
họng');

-- Dữ liệu mẫu cho bảng BENHVIEN
INSERT INTO BENHVIEN (TenBenhVien, HinhAnh)
VALUES
(N'Bệnh viện Chợ Rẫy', '/bvl.png'),
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
(N'Bệnh viện Đại học Y Dược', '/bv2.png'),  
(N'Bệnh viện Từ Dũ', '/bv3.png');  
  
INSERT INTO CHUYENNGANH(TenCN, HinhAnh)  
VALUES  
(N'Viêm gan', '/chuyennganh/cn_1.jpg'),  
(N'Da liễu', '/chuyennganh/cn_2.jpg'),  
(N'Tai - Mũi - Họng', '/chuyennganh/cn_3.jpg');  
  
-- Dữ liệu mẫu cho bảng BACSI  
INSERT INTO BACSI (SoCCCD, NgayCapCCCD, NoiCapCCCD,  
AnhCCCDTruoc, AnhCCCDSau, ChuyenKhoa, NamKN,  
MaBenzVien, MaKB, ChungChiHN, NgayCapCC, AnhCCTruoc,  
AnhCCSau, HocVi, HocHam, ChuyenMon, ThongTinBangCap,  
GioiThieu, CacBenzDieuTri, QuaTrinhHoc,  
QuaTrinhCongTac, NghienCuuKH, GiangDay, HoiVien,  
SoTienKham, MaND)  
VALUES  
( '123456789', '2015-05-15', N'TP. Hồ Chí Minh',  
'/cccd/bs_1_truoc.jpg', '/cccd/bs_1_sau.jpg', N'Bác sĩ  
nội khoa', 10, 2, 1, '123ABC', '2010-06-01',  
'/cc/bs_1_truoc.jpg', '/cc/bs_1_sau.jpg', N'Tiến sĩ',  
N'Giáo sư', N'Chuyên khoa nội', N'Tiến sĩ y khoa', N'Có  
nhiều kinh nghiệm trong điều trị nội khoa', N'Bệnh tiêu  
hóa, hô hấp', N'Trường ĐH Y Dược TP. HCM', N'Bệnh viện  
Chợ Rẫy', N'10 nghiên cứu', N'Giảng dạy tại ĐH Y Dược',  
N'Hiệp hội Y khoa', 300000, 3);  
  
-- Dữ liệu mẫu cho bảng BACSI_CHUYENNGANH  
INSERT INTO BACSI_CHUYENNGANH (MaBS, MaCN)
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
VALUES
(5, 4),
(5, 5),
(5, 6);

-- Dữ liệu mẫu cho bảng LOAIBAIVIET
INSERT INTO LOAIBAIVIET (TenLBV)
VALUES
(N'Tư vấn sức khỏe'),
(N'Bệnh lý thường gặp'),
(N'Chuyên khoa và điều trị');

-- Dữ liệu mẫu cho bảng BAIVIET
INSERT INTO BAIVIET (TieuDeBV, Noidung, MaKB, LinkYtb,
NgayDang, MaLBV, LuotXem)
VALUES
(N'Cách chăm sóc sức khỏe trong mùa dịch', N'Hướng dẫn
chăm sóc sức khỏe mùa dịch COVID-19', 1,
'https://www.youtube.com/link1', '2023-07-01', 1, 100),
(N'Triệu chứng và cách điều trị viêm xoang', N'Bài viết
về triệu chứng và phương pháp điều trị viêm xoang', 3,
'https://www.youtube.com/link2', '2023-06-15', 2, 150),
(N'Chăm sóc trẻ sơ sinh', N'Các bước cơ bản trong chăm
sóc trẻ sơ sinh', 2, 'https://www.youtube.com/link3',
'2023-05-20', 3, 200);

-- Dữ liệu mẫu cho bảng KHUYENMAI
INSERT INTO KHUYENMAI (TenKM, MoTaKM, PhanTramGia,
GiamToiDa, NgayBDKM, NgayKTKM)
VALUES
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
(N'Giảm giá mùa hè', N'Khuyến mãi giảm giá 10% cho tất  
cả các dịch vụ', 10, 50000, '2023-06-01', '2023-07-01'),  
(N'Ưu đãi cho khách hàng mới', N'Giảm giá 20% cho lần  
khám đầu tiên', 20, 100000, '2023-08-01', '2023-09-01'),  
(N'Khuyến mãi tháng 9', N'Giảm giá 15% cho khách hàng  
đặt hẹn trước', 15, 75000, '2023-09-01', '2023-09-30');  
  
-- Dữ liệu mẫu cho bảng KHUYENMAI_CHUYENNGANH  
INSERT INTO KHUYENMAI_CHUYENNGANH (MaCN, MaKM)  
VALUES  
(4, 1),  
(5, 2),  
(6, 2);  
  
-- Dữ liệu mẫu cho bảng TRANGTHAICUOCCHEN  
INSERT INTO TRANGTHAICUOCCHEN (TenTTCH, MoTaTTCH)  
VALUES  
(N'Dang chờ xác nhận', NULL),  
(N'Dã xác nhận', NULL),  
(N'Dã hủy', NULL);  
  
-- Dữ liệu mẫu cho bảng HOSO  
INSERT INTO HOSO (MaND, MoTaBenh)  
VALUES  
(1, N'Hồ sơ điều trị bệnh tiêu hóa của bệnh nhân Nguyễn  
Văn A'),  
(2, N'Hồ sơ theo dõi sức khỏe cho trẻ em của Trần Thị  
B');  
  
-- Dữ liệu mẫu cho bảng HINHANH
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
INSERT INTO HINHANHBENH (MaHS, HinhAnhBenh)
VALUES
(1, '/images/hoso_1_anh1.jpg'),
(1, '/images/hoso_1_anh3.jpg'),
(2, '/images/hoso_3_anh4.jpg');

-- Dữ liệu mẫu cho bảng CUOCHENKHAM
INSERT INTO CUOCHENKHAM (MaND, MaBS, MaHS, MaTTCH,
ThoiGianHen, ThoiGianBD, ThoiGianKT, SoTT, GhiChuBS,
Video, SoTienTT, MaKM)
VALUES
(1, 5, 1, 1, '2023-09-15 14:00:00', '2023-09-15
14:15:00', '2023-09-15 14:45:00', 1, N'Bệnh nhân cần
theo dõi thêm', N'/videos/cuochen1.mp4', 300000, NULL),
(2, 5, 2, 2, '2023-09-16 09:00:00', '2023-09-16
09:15:00', '2023-09-16 09:45:00', 2, N'Khám định kỳ cho
trẻ', N'/videos/cuochen2.mp4', 250000, NULL);

-- Dữ liệu mẫu cho bảng DANHGIA
INSERT INTO DANHGIA (MaCHK, NoiDung, NgayDG, SoSao)
VALUES
(1, N'Dịch vụ rất tốt, bác sĩ nhiệt tình và chu đáo.', '2024-09-20', 5),
(2, N'Bác sĩ tư vấn kỹ càng, nhưng thời gian chờ hơi
lâu.', '2024-09-21', 4);

-- Dữ liệu mẫu cho bảng HOATDONG
INSERT INTO HOATDONG (NoiDungHD)
VALUES
(N'Thêm người dùng mới vào hệ thống.'),
(N'Cập nhật thông tin bác sĩ.'),
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
(N'Gửi thông báo cho người dùng.');

-- Dữ liệu mẫu cho bảng LICHSUHD
INSERT INTO LICHSUHD (MaHD, ThoiGianHD)
VALUES
(1, '10:30:00'),
(2, '11:15:00'),
(3, '14:00:00');

-- Dữ liệu mẫu cho bảng NGUOIDUNG_LICHSUHD
INSERT INTO NGUOIDUNG_LICHSUHD (MaND, MaLS)
VALUES
(1, 1),
(2, 2),
(1, 3);

-- Dữ liệu mẫu cho bảng THONGBAO
INSERT INTO THONGBAO (TieuDeTB, NoiDungTB, ThoiGianTB,
TrangThaiDoc)
VALUES
(N'Thông báo bảo trì hệ thống', N'Hệ thống sẽ bảo trì
vào lúc 22:00 đêm nay.', '2024-09-30 20:00:00', 0),
(N'Thông báo lịch khám mới', N'Cập nhật lịch khám cho
bác sĩ Nguyễn Văn A.', '2024-09-29 09:00:00', 0),
(N'Thông báo chương trình khuyến mãi', N'Nhận ngay ưu
đãi 20% cho lần khám đầu tiên.', '2024-09-28 08:30:00',
0);

-- Dữ liệu mẫu cho bảng THONGBAO_NGUOIDUNG
INSERT INTO THONGBAO_NGUOIDUNG (MaTB, MaND)
```

```

VALUES
(1, 2),
(2, 2),
(3, 2);

-- Dữ liệu mẫu cho bảng THANHTOAN
INSERT INTO THANHTOAN (MaND, SoTien, NgayTT, MaCHK)
VALUES
(1, 200000, '2024-09-20', 1),
(2, 150000, '2024-09-21', 2);

```

4.4. Mô tả các thực thể

4.4.1. Thực thể [PHANQUYEN]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaPQ	INT		Khóa chính	Mã phân quyền
2	TenPQ	NVARCHAR	50	Not Null	Tên phân quyền

4.4.2. Thực thể [NGUOIDUNG]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaND	INT		Khóa chính	Mã người dùng
2	TenND	NVARCHAR	50	Null	Tên người dùng
3	Password	VARCHAR	100	Not Null	Mật khẩu
4	email	VARCHAR	50	Null	email
5	sdt	VARCHAR	15	Not Null	Số điện thoại

MÔ HÌNH HÓA YÊU CẦU VÀ API

6	DiaChi	NVARCHAR	255	Null	Địa chỉ
7	NgaySinh	DATE		Null	Ngày sinh
8	GioiTinh	NVARCHAR	10	Null	Giới tính
9	AnhCaNhan	NVARCHAR	255	Null	Ảnh cá nhân
0	SoDuTK	INT		Not Null	Số dư tài khoản
1	MaGioiThieu	INT		Null	Mã giới thiệu
2	MaNguoiGioiThieu	INT		Not Null	Mã người giới thiệu
3	MaPQ	INT		Khóa ngoại	Mã phân quyền

4.4.3. Thực thể [BENHVIEN]

Sst	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaBenzVien	INT		Khóa chính	Mã bệnh viện
2	TenBenzVien	NVARCHAR	100	Not Null	Tên bệnh viện
3	HinhAnh	VARCHAR	225	Not Null	Hình ảnh

4.4.4. Thực thể [KHOABENH]

Sst	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaKB	INT		Khóa chính	Mã khoa bệnh
2	TenKB	NVARCHAR	100	Not Null	Tên khoa bệnh

4.4.5. Thực thể [CHUYENNGANH]

Sst	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú

MÔ HÌNH HÓA YÊU CẦU VÀ API

1	MaCN	INT		Khóa chính	Mã chuyên ngành
2	TenCN	NVARCHAR	100	Not Null	Tên chuyên ngành
3	HinhANh	VARCHAR	225	Not Null	Hình ảnh

4.4.6. Thực thể [BACSI]

Sđt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaBS	INT		Khóa chính	Mã bác sĩ
2	SoCCCD	VARCHAR	15	Not Null	Số căn cước công dân
3	NgayCapCCCD	Date		Not Null	Ngày cấp căn cước
4	NoiCapCCCD	NVARCHAR	255	Not Null	Nơi cấp căn cước
5	AnhCCCDTruoc	VARCHAR	255	Not Null	Ảnh căn cước trước
6	AnhCCCDsau	VARCHAR	255	Not Null	Ảnh căn cước sau
7	ChuyenKhoa	NVARCHAR	100	Not Null	Chuyên khoa
8	NamKN	INT		Not Null	Năm kinh nghiệm
9	MaBenzVien	INT		Not Null	Mã bệnh viện
0	MaKB	INT		Not Null	Mã khoa bệnh
1	ChungChiHN	NVARCHAR	150	Not Null	Chứng chỉ hành nghề
2	NgayCapCC	Date		Not Null	Ngày cấp chứng chỉ

MÔ HÌNH HÓA YÊU CẦU VÀ API

3	AnhCCTruoc	VARCHAR	255	Not Null	Ảnh chứng chỉ trước
4	AnhCCSau	VARCHAR	255	Not Null	Ảnh chứng chỉ sau
5	HocVi	NVARCHAR	50	Null	Học vị
6	HocHam	NVARCHAR	50	Null	Học hàm
7	ChuyenMon	NVARCHAR	50	Null	Chuyên môn
8	ThongTinBangCap	NVARCHAR	max	Null	Thông tin bằng cấp
9	GioiThieu	NVARCHAR	max	Null	Giới thiệu
0	CacBenhDieuTri	NVARCHAR	max	Null	Các bệnh điều trị
1	QuaTrinhHoc	NVARCHAR	max	Null	Quá trình học
2	NghienCuuKH	NVARCHAR	max	Null	Nghiên cứu khoa học
3	GiangDay	NVARCHAR	max	Null	Giảng dạy
4	HoiVien	NVARCHAR	max	Null	Hội viên
5	SoTienKham	INT		Null	Số tiền khám
6	MaND	INT		Khóa ngoại	Mã người dùng
7	MaTTBS	INT		Khóa ngoại	Mã trạng thái bác sĩ

4.4.7. Thực thể [TRANGTHAICUOCHEM]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaTTCH	INT		Khóa chính	Mã trạng thái
2	TenTTCH	NVARCHAR	100	Not Null	Tên trạng thái

MÔ HÌNH HÓA YÊU CẦU VÀ API

3	MoTaTTCH	NVARCHAR	255	Null	Mô tả trạng thái
---	----------	----------	-----	------	------------------

4.4.8. Thực thể [HOSO]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaHS	INT		Khóa chính	Mã hồ sơ
2	MaND	INT		Khóa ngoại	Mã người dùng
3	MoTaBenh	NVARCHAR	255	Null	Mô tả bệnh

4.4.9. Thực thể [HINHANHBENH]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaHinh	INT		Khóa chính	Mã hình
2	MaHS	INT		Khóa ngoại	Mã hồ sơ
3	HinhAnhBenh	VARCHAR	225	Not Null	Hình ảnh bệnh

4.4.10. Thực thể [KHUYENMAI]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaKM	INT		Khóa chính	Mã khuyến mãi
2	TenKM	NVARCHAR	100	Not Null	Tên khuyến mãi
3	MoTaKM	NVARCHAR	225	Not Null	Mô tả khuyến mãi

MÔ HÌNH HÓA YÊU CẦU VÀ API

4	PhanTramGiam	INT		Not Null	Phạm trâm giảm
5	GiamToiDa	INT		Not Null	Giảm tối đa
6	NgayBDKM	DATETIME		Not Null	Ngày bắt đầu
7	NgayKTKM	DATETIME		Not Null	Ngày kết thúc

4.4.11. Thực thể [CUOCHENKHAM]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaCHK	INT		Khóa chính	Mã hồ sơ
2	MaND	INT		Khóa ngoại	Mã người dùng
3	MaHS	INT		Khóa ngoại	Mã hồ sơ
	MaBS	INT		Khóa ngoại	Mã bác sĩ
4	MaTTCH	INT		Khóa ngoại	Mã trạng thái cuộc hẹn
5	ThoiDianHen	DATETIME		Not Null	Thời gian hẹn
6	ThoiGianBatDau	DATETIME		Null	Thời gian bắt đầu
7	ThoiGianKetThuc	DATETIME		Null	Thời gian kết thúc
8	SoTT	INT		Not Null	Số thứ tự
9	GhiChuBS	NVARCHAR	225	Null	Ghi chú bác sĩ
0	video	VARCHAR	225	Null	Video lưu lại
1	SoTienTT	INT		Null	Số tiền thanh toán

MÔ HÌNH HÓA YÊU CẦU VÀ API

2	MaKM	INT		Khóa ngoại	Mã khuyến mãi
---	------	-----	--	------------	---------------

4.4.12. Thực thể [DANHGIA]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
8	MaDG	INT		Khóa chính	Mã hồ sơ
9	MaCHK	INT		Khóa ngoại	Mã cuộc hẹn khám
0	NoiDung	NVARCHAR	225	Null	Nội dung
1	NgayDG	DATE		Not Null	Ngày đánh giá
2	SoSao	INT		Not Null	Số sao

4.4.13. Thực thể [THANHTOAN]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaTT	INT		Khóa chính	Mã hồ sơ
2	MaND	INT		Khóa ngoại	Mã người dùng
3	SoTien	INT		Not Null	Số tiền
4	NgayTT	DateTime		Not Null	Ngày thanh toán
5	MaCHK	INT		Khóa ngoại	Mã cuộc hẹn khám
6	LydoHoanPhi	NVARCHAR	200	Null	Lý do hoàn phí
7	NgayHoanPhi	DATETIME		NONE	Ngày hoàn phí

4.4.14. Thực thể [THONGBAO]

Số thứ tự	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaTB	INT		Khóa chính	Mã hồ sơ
2	TieuDeTB	NVARCHAR	100	Not Null	Tiêu đề
3	NoiDungTB	NVARCHAR	255	Not Null	Nội dung
4	ThoiGianTB	DATETIME		Not Null	Thời gian thông báo
5	TrangThaiDoc	BIT		Not Null	Trạng thái đọc

4.4.15. Thực thể [HOATDONG]

Số thứ tự	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
6	MaHD	INT		Khóa chính	Mã hồ sơ
7	NoiDungHD	NVARCHAR	225	Not Null	Nội dung

4.4.16. Thực thể [LICHGSUHD]

Số thứ tự	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
8	MaLS	INT		Khóa chính	Mã hồ sơ
9	MaHD	INT		Khóa ngoại	Mã hoạt động
0	ThoiGianHD	DATETIME		Not Null	Thời gian hoạt động

4.4.17. Thực thể [LOAIBAIVIET]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaLBV	INT		Khóa chính	Mã hồ sơ
2	TenLBV	NVARCHAR	100	Not Null	Tên loại bài viết

4.4.18. Thực thể [BAIVIET]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
3	MaBV	INT		Khóa chính	Mã hồ sơ
4	TieuDeBV	NVARCHAR	225	Not Null	Tiêu đề
5	NoiDung	NVARCHAR	225	Not Null	Nội dung
6	MaKB	INT		Not Null	Mã khoa bệnh
7	LinkYtb	VARCHAR	225	Not Null	Link Youtube
8	NgayDang	DATE		Not Null	Ngày đăng
9	LuotXem	INT		Not Null	Lượt xem
0	MaLBV	INT		Not Null	Mã loại bài viết

4.4.19. Thực thể [TRANGTHAIBACSI]

Stt	Thuộc tính	Kiểu dữ liệu	Kích thước	Ràng buộc	Ghi chú
1	MaTTBS	INT		Khóa chính	Mã trạng thái bác sĩ
2	TenTTBS	NVARCHAR	225	Not Null	Tên trạng thái bác sĩ

4.5. Cài các Store Procedure

4.5.1. Đăng ký người dùng

Chức năng: Được sử dụng để đăng ký một người dùng mới vào hệ thống. Nó nhận vào tên người dùng, mật khẩu và số điện thoại, sau đó tự động tạo một MaGioiThieu duy nhất và thêm người dùng vào bảng NGUOIDUNG với quyền mặc định là 2 và số dư tài khoản là 0. Cuối cùng, procedure này trả về thông tin chi tiết của người dùng vừa được tạo.

Mã nguồn:

```
-- Đăng ký người dùng
GO
CREATE OR ALTER PROCEDURE REGISTER
    @TenND NVARCHAR(50),
    @Password VARCHAR(50),
    @sdt VARCHAR(15)
AS
BEGIN
    DECLARE @MaGioiThieu INTEGER;
    SELECT @MaGioiThieu = ISNULL(MAX(MaGioiThieu), 0) +
1 FROM NGUOIDUNG;

    -- Thêm người dùng mới
    INSERT INTO NGUOIDUNG (TenND, Password, sdt, MaPQ,
SoDuTK, MaGioiThieu)
    VALUES      (@TenND,      @Password,      @sdt,      2,      0,
@MaGioiThieu);

    -- Trả về thông tin người dùng vừa tạo
    SELECT *
    FROM NGUOIDUNG
    WHERE sdt = @sdt;
```

```
END  
EXEC REGISTER N'Le Van Anh', '123123', '0808789789'
```

4.5.2. Kiểm tra đăng nhập

Chức năng: Dùng để xác thực thông tin đăng nhập của người dùng. Nó nhận vào số điện thoại và mật khẩu, sau đó tìm kiếm trong bảng NGUOIDUNG để kiểm tra xem có bản ghi nào khớp với cả hai thông tin này hay không và trả về toàn bộ thông tin của người dùng nếu tìm thấy.

Mã nguồn:

```
-- Kiểm tra đăng nhập  
GO  
CREATE OR ALTER PROCEDURE CheckLogin @sdt VARCHAR(15),  
@pass VARCHAR(100)  
AS  
BEGIN  
    Select *  
    from NGUOIDUNG  
    Where sdt = @sdt and Password =@pass  
END  
GO  
EXEC CheckLogin '0909123456', '123456'  
-- cập nhật thông tin người dùng  
GO  
CREATE OR ALTER PROCEDURE UpdateUserInfo  
    @MaND INT,  
    @TenND NVARCHAR(50),  
    @Email VARCHAR(50),  
    @NamSinh DATE,  
    @GioiTinh NVARCHAR(5),  
    @DiaChi NVARCHAR(100),
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
@AnhCaNhan VARCHAR(50) = NULL
AS
BEGIN
    DECLARE @CurrentAnhCaNhan VARCHAR(50);

    -- Lấy ảnh cá nhân hiện tại của người dùng
    SELECT @CurrentAnhCaNhan = AnhCaNhan
    FROM NGUOIDUNG
    WHERE MaND = @MaND;

    -- Nếu không truyền ảnh mới thì giữ nguyên ảnh cũ
    IF @AnhCaNhan IS NULL OR @AnhCaNhan = 'NULL'
    BEGIN
        SET @AnhCaNhan = @CurrentAnhCaNhan;
    END
    -- Cập nhật thông tin người dùng
    UPDATE NGUOIDUNG
    SET TenND = @TenND,
        Email = @Email,
        NgaySinh = @NamSinh,
        GioiTinh = @GioiTinh,
        DiaChi = @DiaChi,
        AnhCaNhan = @AnhCaNhan
    WHERE MaND = @MaND;
END
SELECT * from NGUOIDUNG
```

4.5.3. Lấy thông tin của bài viết theo mã

Chức năng: Được sử dụng để truy xuất thông tin chi tiết của một bài viết cụ thể. Nó nhận vào MaBV (mã bài viết) và trả về tất cả các trường của bài viết đó cùng với tên loại bài viết tương ứng bằng cách kết hợp dữ liệu từ hai bảng BAIVIET và LOAIBAIVIET.

Mã nguồn:

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
-- -- Lấy thông tin của bài viết theo mã
GO
CREATE OR ALTER PROCEDURE getBaiVietByID @MaBV INT
AS
BEGIN
    Select *, lbv.TenLBV
    from BAIVIET bv, LOAIBAIVIET lbv
    Where MaBV = @MaBV and bv.MaLBV = lbv.MaLBV
END
GO
EXEC getBaiVietByID 1
-- -- Lấy thông tin của bài viết theo mã loại bài viết
GO
CREATE OR ALTER PROCEDURE getBaiVietByIDMaLoai
    @MaLBV INT
AS
BEGIN
    SELECT
        bv.*, lbv.TenLBV
    FROM
        BAIVIET bv
    INNER JOIN
        LOAIBAIVIET lbv ON bv.MaLBV = lbv.MaLBV
    WHERE
        lbv.MaLBV = @MaLBV
    ORDER BY
        bv.NgayDang DESC; -- Sắp xếp bài viết theo ngày
đăng mới nhất
END
GO
```

```
EXEC getBaiVietByIDMaLoai 4
```

4.5.4. Lấy thông tin bài viết theo mã khoa bệnh

Chức năng: Được dùng để lấy danh sách các bài viết liên quan đến một khoa bệnh cụ thể. Nó nhận vào MaKB (mã khoa bệnh) và trả về tất cả các bài viết có liên kết với khoa bệnh đó, đồng thời sắp xếp kết quả theo ngày đăng giảm dần để hiển thị những bài viết mới nhất trước.

Mã nguồn:

```
-- -- Lấy thông tin của bài viết theo mã khoa bệnh
GO
CREATE OR ALTER PROCEDURE getBaiVietByIDMaKhoaBenh
    @MaKB INT
AS
BEGIN
    SELECT
        bv.*
    FROM
        BAIVIET bv
    INNER JOIN
        KHOABENH kb ON kb.MaKB = bv.MaKB
    WHERE
        kb.MaKB = @MaKB
    ORDER BY
        bv.NgayDang DESC; -- Sắp xếp bài viết theo ngày
        đăng mới nhất
END
GO
EXEC getBaiVietByIDMaKhoaBenh 2
exec getallBacSi
```

4.5.5. Lấy thông tin chi tiết bác sĩ

Chức năng: Được sử dụng để lấy thông tin chi tiết của một bác sĩ cụ thể. Nó nhận vào MaBS (mã bác sĩ) và trả về các thông tin như mã bác sĩ, mã người dùng, tên người dùng, tên bệnh viện, tên khoa bệnh và số tiền khám của bác sĩ đó bằng cách kết hợp dữ liệu từ các bảng BACSI, NGUOIDUNG, KHOABENH và BENHVIEN.

Mã nguồn:

```
-- chi tiết bác sĩ
GO
CREATE OR ALTER PROCEDURE DetDETAILLBACSI @MaBS INT
AS
BEGIN
    SELECT
        bs.*,
        nd.MaND,
        nd.TenND,
        nd.AnhCaNhan,
        bv.TenBenhVien,
        kb.TenKB
    FROM BACSI bs
    INNER JOIN NGUOIDUNG nd ON bs.MaND = nd.MaND
    INNER JOIN KHOABENH kb ON bs.MaKB = kb.MaKB
    INNER JOIN BENHVIEN bv ON bs.MaBenzVien = bv.MaBenzVien
    WHERE bs.MaBS = @MaBS
END
EXEC DetDETAILLBACSI 5
```

4.5.6. Lọc bác sĩ theo nhiều tiêu chí

Chức năng: Được sử dụng để lấy thông tin chi tiết của một bác sĩ cụ thể. Nó nhận vào MaBS (mã bác sĩ) và trả về các thông tin như mã bác sĩ, mã người dùng, tên

MÔ HÌNH HÓA YÊU CẦU VÀ API

người dùng, tên bệnh viện, tên khoa bệnh và số tiền khám của bác sĩ đó bằng cách kết hợp dữ liệu từ các bảng BACSI, NGUOIDUNG, KHOABENH và BENHVIEN.
Mã nguồn:

```
GO

CREATE OR ALTER PROCEDURE DetDETAILLBACSI1 @MaBS INT
AS
BEGIN
    -- Lấy danh sách bác sĩ theo các tiêu chí lọc
    SELECT
        bs.MaBS,
        nd.MaND,
        nd.TenND,
        bv.TenBenhVien,
        kb.TenKB,
        bs.SoTienKham
    FROM BACSI bs
    INNER JOIN NGUOIDUNG nd ON bs.MaND = nd.MaND
    INNER JOIN KHOABENH kb ON bs.MaKB = kb.MaKB
    INNER JOIN BENHVIEN bv ON bs.MaBenzVien = bv.MaBenzVien
    WHERE bs.MaBS = @MaBS
END
EXEC DetDETAILLBACSI1 5
```

4.5.7. Lấy tất cả bình luận bằng id Bác sĩ

Chức năng: Được dùng để lấy tất cả các bình luận và đánh giá về một bác sĩ cụ thể. Nó nhận vào MaBS (mã bác sĩ), sau đó truy xuất tên người dùng, ảnh cá nhân của người đánh giá, ngày đánh giá, nội dung bình luận và số sao từ các bảng BACSI, CUOCCHENKHAM, DANHGIA và NGUOIDUNG, chỉ trả về những cuộc hẹn đã có đánh giá.

Mã nguồn:

```
GO
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
CREATE OR ALTER PROCEDURE GetCommentBACSI @MaBS INT
AS
BEGIN
    SELECT
        nd.TenND,
        nd.AnhCaNhan,
        dg.NgayDG,
        dg.NoIDung,
        dg.SoSao
    FROM BACSI bs
    INNER JOIN CUOCHENKHAM chk ON chk.MaBS = bs.MaBS
    INNER JOIN DANHGIA dg ON chk.MaCHK = dg.MaCHK -- Chỉ
    lấy những dòng có đánh giá
    INNER JOIN NGUOIDUNG nd ON chk.MaND = nd.MaND -- --
    Lấy thông tin từ bệnh nhân
    WHERE bs.MaBS = @MaBS;
END
EXEC GetCommentBACSI 5
```

4.5.8. Lưu hồ sơ bệnh nhân

Chức năng: Được thiết kế để lưu trữ một hồ sơ bệnh mới vào hệ thống. Nó nhận vào mã người dùng (MaND) và mô tả bệnh (MoTaBenh), sau đó chèn dữ liệu này vào bảng HOSO. Procedure này sử dụng giao dịch để đảm bảo tính toàn vẹn dữ liệu, trả về mã hồ sơ vừa được tạo (MaHS) nếu thành công, hoặc thông báo lỗi nếu có bất kỳ vấn đề nào xảy ra trong quá trình lưu trữ.

Mã nguồn:

```
-- Lưu hồ sơ bệnh
GO
CREATE OR ALTER PROCEDURE SAVEHOSO
    @MaND INT,
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
@MoTaBenh NVARCHAR(MAX),  
@MaHS INT OUTPUT  
  
AS  
  
BEGIN  
    BEGIN TRANSACTION;  
    BEGIN TRY  
        INSERT INTO [HOSO] ([MaND], [MoTaBenh])  
        VALUES (@MaND, @MoTaBenh);  
        SET @MaHS = SCOPE_IDENTITY();  
        COMMIT TRANSACTION;  
        SELECT @MaHS AS MaHS;  
    END TRY  
    BEGIN CATCH  
        ROLLBACK TRANSACTION;  
        SELECT ERROR_MESSAGE() AS ErrorMessage;  
    END CATCH  
END;  
  
GO  
  
DECLARE @MaHS INT; EXEC SAVEHOSO 12, N'Hồ sơ điều trị  
bệnh tiêu hóa của bệnh nhân LuongTienDat', @MaHS = @MaHS  
OUTPUT;
```

4.5.9. Lưu hình ảnh bệnh

Chức năng: Được sử dụng để lưu trữ hình ảnh liên quan đến một hồ sơ bệnh cụ thể. Nó nhận vào MaHS (mã hồ sơ bệnh) và đường dẫn hoặc tên file của hình ảnh (HinhAnhBenh), sau đó chèn thông tin này vào bảng HINHANHBENH, liên kết hình ảnh với hồ sơ bệnh tương ứng.

Mã nguồn:

```
-- Lưu hình ảnh bệnh  
GO
```

```
CREATE PROCEDURE SAVEHINHANHBENH
    @MaHS INT,
    @HinhAnhBenh VARCHAR(100)
AS
BEGIN
    INSERT INTO [HINHANHBENH] ([MaHS], [HinhAnhBenh])
    VALUES (@MaHS, @HinhAnhBenh);
END;
GO
EXEC SAVEHINHANHBENH 1, 'vsbs';
```

4.5.10. Lưu cuộc hẹn khám

Chức năng: Được sử dụng để lưu trữ một cuộc hẹn khám mới. Nó nhận vào mã người dùng (MaND), mã bác sĩ (MaBS), mã hồ sơ bệnh (MaHS) và tùy chọn mã khuyến mãi (MaKM). Procedure này tự động gán thời gian hiện tại làm ThoiGianHen, tính toán SoTT (số thứ tự) dựa trên các cuộc hẹn hiện có của bác sĩ, và chèn thông tin cuộc hẹn vào bảng CUOCHEKKHAM với trạng thái mặc định là 1

Mã nguồn:

```
GO
CREATE PROCEDURE SAVECUOCHENKHAM
    @MaND INT,
    @MaBS INT,
    @MaHS INT,
    @MaKM INT = NULL
AS
BEGIN
    DECLARE @ThoiGianHen DATETIME = GETDATE(); -- Lấy
thời gian hiện tại
    DECLARE @SoTienTT INT = NULL;
    DECLARE @SoTT TINYINT;
```

```
-- Lấy số thứ tự từ danh sách cuộc hẹn của bác sĩ
SELECT @SoTT = COALESCE(MAX(SoTT), 0) + 1
FROM [CUOCHENKHAM]
WHERE MaBS = @MaBS;

INSERT INTO [CUOCHENKHAM] ([MaND], [MaBS], [MaHS],
[MaTTCH], [ThoiGianHen], [ThoiGianBD], [ThoiGianKT],
[SoTT], [GhiChuBS], [Video], [SoTienTT], [MaKM])
VALUES (@MaND, @MaBS, @MaHS, 1, @ThoiGianHen, Null,
Null, @SoTT, null, Null, @SoTienTT, @MaKM);

END;
EXEC SAVECUOCHENKHAM 12, 5, 4, null
```

4.5.11. Lấy thông tin cuộc hẹn

Chức năng: Được thiết kế để lấy thông tin về các cuộc hẹn khám của một người dùng cụ thể. Nó nhận vào MaND (mã người dùng) và trả về chi tiết cuộc hẹn như mã cuộc hẹn, tên bác sĩ, số thứ tự, mã và tên trạng thái cuộc hẹn, số tiền thanh toán, đường dẫn video và ghi chú của bác sĩ, bằng cách kết hợp dữ liệu từ các bảng CUOCHENKHAM, BACSI, NGUOIDUNG và TRANGTHAICUOCHEN.

Mã nguồn:

```
-- Lấy thông tin cuộc hẹn theo MaND
GO
CREATE OR ALTER PROCEDURE GetLichKhamInfoByMaND
    @MaND INT
AS
BEGIN
    SELECT
        chk.MaCHK,
        nd.TenND AS TenBacSi,
        chk.SoTT,
```

```
chk.MaTTCH,
tt.TenTTCH,
chk.SoTienTT,
chk.Video,
chk.GhiChuBS

FROM CUOCHENKHAM chk
INNER JOIN BACSI bs ON chk.MaBS = bs.MaBS
INNER JOIN NGUOIDUNG nd ON bs.MaND = nd.MaND -- Lấy
tên bác sĩ từ bảng NGUOIDUNG
INNER JOIN TRANGTHAICUOCHEN tt ON chk.MaTTCH =
tt.MaTTCH -- Lấy thông tin trạng thái cuộc hẹn
WHERE chk.MaND = @MaND
END
EXEC GetLichKhamInfoByMaND 12
```

4.5.12. Xóa cuộc hẹn

Chức năng: Được sử dụng để xóa một cuộc hẹn khám cùng với các dữ liệu liên quan. Nó nhận vào MaCHK (mã cuộc hẹn khám), sau đó thực hiện xóa cuộc hẹn, các hình ảnh bệnh và hồ sơ bệnh tương ứng. Procedure này sử dụng giao dịch để đảm bảo rằng tất cả các thao tác xóa đều thành công hoặc không có thao tác nào được thực hiện, duy trì tính toàn vẹn của dữ liệu.

Mã nguồn:

```
-- Xóa cuộc hẹn
GO
CREATE OR ALTER PROCEDURE DeleteCHKbyID
    @MaCHK INT
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION;
```

```
-- Lấy MaHS từ CUOCHENKHAM để xóa hình ảnh và hồ sơ liên quan
DECLARE @MaHS INT;
SELECT @MaHS = MaHS FROM CUOCHENKHAM WHERE MaCHK = @MaCHK;

-- Xóa cuộc hẹn khám trước
DELETE FROM CUOCHENKHAM
WHERE MaCHK = @MaCHK;

-- Xóa hình ảnh liên quan đến MaHS
DELETE FROM HINHANHBENH
WHERE MaHS = @MaHS;

-- Xóa hồ sơ liên quan đến MaHS
DELETE FROM HOSO
WHERE MaHS = @MaHS;
COMMIT TRANSACTION;

END TRY
BEGIN CATCH
    ROLLBACK TRANSACTION;
    THROW;
END CATCH
END
```

4.5.13. Nạp tiền vào tài khoản

Chức năng: Được sử dụng để nạp tiền vào tài khoản của người dùng. Nó nhận vào MaND (mã người dùng) và số tiền muốn nạp (@SoDuTK), sau đó cập nhật tăng số dư tài khoản hiện có của người dùng trong bảng NGUOIDUNG lên một lượng bằng số tiền nạp.

Mã nguồn:

```
-- Nạp tiền vào tài khoản
GO
CREATE OR ALTER PROCEDURE NapTien @MaND INT, @SoDuTK INT
AS
BEGIN
    UPDATE NGUOIDUNG
    SET SoDuTK = SoDuTK + @SoDuTK
    WHERE MaND = @MaND;
END
EXEC NapTien 12, 0
```

4.5.14. Đề xuất top 5 bác sĩ có lịch khám nhiều nhất

Chức năng: Được dùng để liệt kê các bác sĩ có số lượng cuộc hẹn khám nhiều nhất. Nó tính toán tổng số cuộc hẹn cho mỗi bác sĩ, sau đó trả về thông tin của 5 bác sĩ có số lượng cuộc hẹn cao nhất, bao gồm mã bác sĩ, tên bác sĩ, ảnh cá nhân và tổng số lịch hẹn, sắp xếp theo thứ tự giảm dần

Mã nguồn:

```
GO
CREATE OR ALTER PROCEDURE GetTop7Doctors
AS
BEGIN
    -- Tính tổng số lượng lịch hẹn của từng bác sĩ
    SELECT TOP 5
        BS.MaBS,
        ND.TenND AS TenBacSi,
        ND.ANHCANHAN,
        COUNT(CH.MaCHK) AS SoLuongLichHen
    FROM BACSI BS
    INNER JOIN NGUOIDUNG ND ON BS.MaND = ND.MaND
    INNER JOIN BENHVIEN BV ON BS.MaBenzVien = BV.MaBenzVien
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
INNER JOIN CUOCHENKHAM CH ON BS.MaBS = CH.MaBS
GROUP BY BS.MaBS, ND.TenND, ND.ANHCANHAN
ORDER BY SoLuongLichHen DESC
END
EXEC GetTop7Doctors
```

4.5.15. Lấy danh sách thông báo theo mã nguồn

Chức năng: Được sử dụng để lấy danh sách các thông báo dành cho một người dùng cụ thể. Nó nhận vào MaND (mã người dùng) và trả về các thông tin chi tiết của thông báo như mã thông báo, tiêu đề, nội dung, thời gian và trạng thái đọc, bằng cách kết hợp dữ liệu từ bảng THONGBAO và THONGBAO_NGUOIDUNG, sắp xếp theo thời gian

Mã nguồn:

```
-- Lấy danh sách thông báo theo mã người dùng
GO
CREATE OR ALTER PROCEDURE sp_GetThongBaoByMaND
    @MaND INT
AS
BEGIN
    SELECT
        tb.MaTB,           tb.TieuDeTB,           tb.NoDungTB,
        tb.ThoiGianTB,   tb.TrangThaiDoc
    FROM
        THONGBAO tb
    INNER JOIN
        THONGBAO_NGUOIDUNG tbn ON tb.MaTB = tbn.MaTB
    WHERE
        tbn.MaND = @MaND
    ORDER BY
        tb.ThoiGianTB DESC;
```

```
END;  
EXEC sp_GetThongBaoByMaND @MaND = 2;  
GO
```

4.5.16. Lấy danh sách hồ sơ bệnh

Chức năng: Được sử dụng để lấy danh sách các hồ sơ bệnh mà một bác sĩ cụ thể đã khám. Nó nhận vào MaBS (mã bác sĩ), sau đó truy xuất thông tin chi tiết của hồ sơ bệnh (mã hồ sơ, mô tả bệnh) và thông tin của người dùng liên quan (tên, email, số điện thoại, địa chỉ, giới tính, ảnh cá nhân), cùng với các hình ảnh bệnh liên quan. Dữ liệu được lấy từ các bảng HOSO, NGUOIDUNG, HINHANHBENH và CUOCHENKHAM, đảm bảo chỉ những hồ sơ có liên quan đến bác sĩ đó mới được trả về.

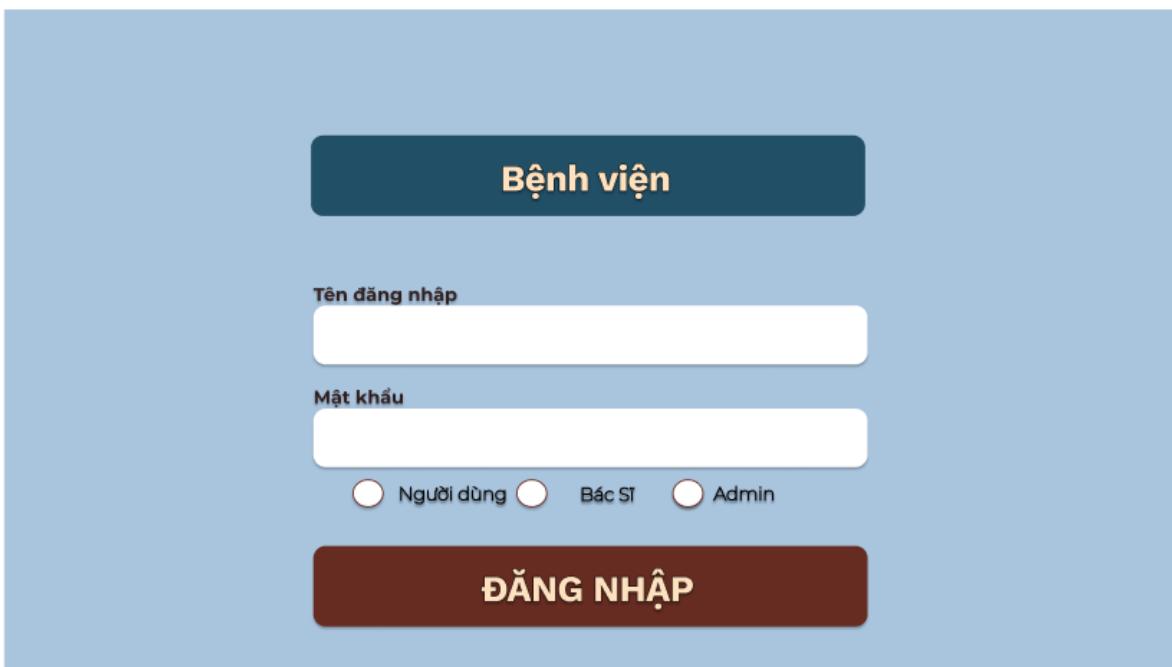
Mã nguồn:

```
CREATE PROCEDURE sp_LayDanhSachHoSoTheoMaBacSi  
    @MaBS INT  
AS  
BEGIN  
    SET NOCOUNT ON;  
  
    SELECT  
        hs.MaHS,  
        hs.MoTaBenh,  
        hs.MaND,  
        nd.TenND,  
        nd.Email,  
        nd.sdt,  
        nd.DiaChi,  
        nd.GioiTinh,  
        nd.AnhCaNhan,  
        hb.MaHinh,  
        hb.HinhAnhBenh  
    FROM HOSO hs
```

MÔ HÌNH HÓA YÊU CẦU VÀ API

```
INNER JOIN NGUOIDUNG nd ON hs.MaND = nd.MaND
LEFT JOIN HINHANHBENH hb ON hs.MaHS = hb.MaHS
WHERE hs.MaHS IN (
    SELECT DISTINCT MaHS
    FROM CUOCHENKHAM
    WHERE MaBS = @MaBS
)
ORDER BY hs.MaHS;
END;
GO
EXEC sp_LayDanhSachHoSoTheoMaBacSi 31
```

4.6. Wireframe



Hình 2: Giao diện Wireframe đăng nhập người dùng

Giao diện đăng nhập được thiết kế với tông màu xanh nhạt chủ đạo, tạo cảm giác thân thiện và dễ chịu cho người sử dụng. Bộ cục được sắp xếp gọn gàng, tập trung vào các trường thông tin cần thiết và các tùy chọn đăng nhập.

Các thành phần chính trên giao diện bao gồm:

1. Tiêu đề: Nằm ở vị trí trung tâm phía trên, nổi bật với chữ "Bệnh viện" màu vàng trên nền xanh đậm. Đây là biểu tượng nhận diện của hệ thống, cho biết mục đích sử dụng của giao diện.
2. Trường nhập liệu "Tên đăng nhập": Ô nhập liệu màu trắng, có đường viền rõ ràng, được đặt dưới nhãn "Tên đăng nhập". Đây là nơi người dùng điền thông tin tài khoản của mình.
3. Trường nhập liệu "Mật khẩu": Tương tự như trường tên đăng nhập, ô nhập liệu này cũng có màu trắng và được đặt dưới nhãn "Mật khẩu". Người dùng sẽ nhập mật khẩu của mình vào đây.
4. Các nút chọn vai trò (Radio buttons): Dưới hai trường nhập liệu là ba tùy chọn dạng nút radio, cho phép người dùng lựa chọn vai trò khi đăng nhập:
 - "Người dùng": Dành cho người dùng thông thường, có thể là bệnh nhân hoặc người thân.

- "Bác sĩ": Dành cho các bác sĩ hoặc chuyên gia y tế.
- "Admin": Dành cho quản trị viên hệ thống. Việc lựa chọn vai trò này có thể ảnh hưởng đến quyền truy cập và các chức năng mà người dùng có thể sử dụng sau khi đăng nhập thành công.

5. Nút "**ĐĂNG NHẬP**": Nút bấm chính để thực hiện quá trình đăng nhập, có màu nâu sẫm với chữ "**ĐĂNG NHẬP**" màu trắng được in hoa và làm nổi bật. Nút này được đặt ở vị trí cuối cùng, dễ nhận biết và thao tác, là điểm cuối của luồng đăng nhập.

Tổng thể, giao diện này được thiết kế theo hướng tối giản, trực quan và dễ sử dụng, nhằm mục đích cung cấp trải nghiệm đăng nhập hiệu quả cho người dùng trong môi trường bệnh viện.

Hình 3: Giao diện Wireframe thêm bác sĩ

Giao diện "Thêm Bác Sĩ" được thiết kế với tông màu xanh dương nhạt ở phần tiêu đề và nền trắng chủ đạo cho các vùng nhập liệu, tạo cảm giác chuyên nghiệp và rõ ràng. Bố cục được chia thành hai phần chính: thông tin chi tiết của bác sĩ ở bên trái và khu vực quản lý ảnh đại diện ở bên phải.

Các thành phần chính trên giao diện bao gồm:

1. Tiêu đề cửa sổ: Nằm ở góc trên cùng bên trái, hiển thị rõ "Thêm Bác Sĩ", cho biết chức năng của cửa sổ hiện tại. Bên cạnh đó là nút "X" ở góc trên cùng bên phải để đóng cửa sổ.

2. Thông tin chi tiết bác sĩ (phần bên trái):

- Mã bác sĩ (*): Trường nhập liệu với nhãn "Mã bác sĩ" và dấu (*) màu đỏ, cho thấy đây là trường bắt buộc. Giá trị mặc định là "BS01".
- Tên bác sĩ (*): Trường nhập liệu với nhãn "Tên bác sĩ" và dấu (*) màu đỏ, cũng là trường bắt buộc. Giá trị mặc định là "Hoàng Anh Hảo".
- Giới tính: Một ô chọn dạng dropdown (thanh cuộn xuống) với giá trị mặc định là "Nam", cho phép người dùng lựa chọn giới tính của bác sĩ.
- Ngày sinh: Trường nhập liệu dạng ngày tháng với giá trị mặc định "20/11/1975" và biểu tượng lịch bên cạnh, cho phép chọn ngày dễ dàng.
- Số CMND: Trường nhập liệu cho số chứng minh nhân dân với giá trị mặc định là "023456789".
- Ngày cấp: Trường nhập liệu dạng ngày tháng với giá trị mặc định "11/08/2025" và biểu tượng lịch, liên quan đến ngày cấp CMND.
- Vị trí công tác: Trường nhập liệu cho vị trí làm việc hoặc chuyên khoa với giá trị mặc định là "Khoa nội soi".
- Học vị: Trường nhập liệu cho học vị của bác sĩ với giá trị mặc định là "Tiến sĩ".

3. Khu vực ảnh đại diện (phần bên phải):

- Khung "Ảnh đại diện": Một khung lớn màu xám nhạt với biểu tượng hình người mặc định bên trong, gợi ý vị trí để tải ảnh đại diện của bác sĩ.
- Các nút thao tác ảnh:
 - Nút ba chấm "..." màu đen: Dùng để duyệt và tải ảnh lên từ máy tính.
 - Nút dấu "X" màu đỏ: Dùng để xóa ảnh đại diện đã tải lên.
- Thông báo định dạng ảnh: Dòng chữ "Chọn các ảnh có định dạng (.jpg, .jpeg, .png, .gif)" cung cấp thông tin về các định dạng ảnh được hỗ trợ.

4. Các nút điều khiển phía dưới:

- Nút "Lưu": Biểu tượng đĩa mềm và chữ "Lưu", dùng để lưu lại các thông tin đã nhập hoặc thay đổi.

MÔ HÌNH HÓA YÊU CẦU VÀ API

- Nút "Hủy bỏ": Biểu tượng dấu "X" và chữ "Hủy bỏ", dùng để đóng cửa sổ mà không lưu các thay đổi.
- Nút "Giúp": Biểu tượng dấu hỏi chấm "?" và chữ "Giúp", dùng để truy cập phần trợ giúp hoặc hướng dẫn sử dụng.

Tổng thể, giao diện này được thiết kế rõ ràng, trực quan, hỗ trợ người dùng nhập liệu thông tin bác sĩ một cách chi tiết và quản lý ảnh đại diện một cách hiệu quả.

The wireframe shows a user interface for managing physician salaries. On the left is a sidebar with the VINMEC logo and navigation links: Trang Chủ, Xem Lịch Làm Việc, Báo Cáo (with sub-links: Báo cáo giờ dạy, Báo cáo kết quả học, Báo cáo khác), Xem Đánh Giá, and Xem Lương (with sub-links: Lịch sử nhận lương, Lương hiện tại, Mức thưởng và phụ cấp). The main content area has a header 'Quản Lý Thông Tin | Thông tin bác sĩ' and a sub-header 'Quản lý lương (Tháng 9/2023)'. It includes a search bar ('Tim Kiếm'), a date filter ('Lọc theo năm') set to '08/10/2023 00:00', and a table listing salary details. The table columns are: Mã, Lương tháng, Tiền lương, Ngày nhận lương, and Trạng Thái. Each row contains a 'Đang tính' button, a 'Phản hồi' button, and three green buttons labeled 'Chưa nhận', 'Chi Tiết', and 'Phản hồi'. The table lists eight rows of salary data from August to February 2023, all marked as 'Đang tính' and 'Chưa nhận'.

Mã	Lương tháng	Tiền lương	Ngày nhận lương	Trạng Thái
longkh1	10/2023	-	-	Đang tính Phản hồi
longkh1	09/2023	120.000.000	-	Chưa nhận Chi Tiết Phản hồi
longkh1	08/2023	120.000.000	29-08-2023 17:31:23	Đã nhận Chi Tiết
longkh1	07/2023	120.000.000	30-07-2023 16:32:59	Đã nhận Chi Tiết
longkh1	06/2023	120.000.000	27-06-2023 19:30:23	Đã nhận Chi Tiết
longkh1	05/2023	120.000.000	29-05-2023 17:31:23	Đã nhận Chi Tiết
longkh1	04/2023	120.000.000	29-04-2023 17:31:23	Đã nhận Chi Tiết
longkh1	03/2023	120.000.000	29-03-2023 17:31:23	Đã nhận Chi Tiết
longkh1	02/2023	120.000.000	28-02-2023 17:31:23	Đã nhận Chi Tiết

Hình 4: Giao diện Wireframe lương bác sĩ

Giao diện quản lý được thiết kế với bố cục gồm một thanh điều hướng dọc (sidebar) ở bên trái và khu vực nội dung chính ở bên phải. Tông màu xanh đậm và trắng tạo cảm giác chuyên nghiệp và dễ đọc.

I. Thanh điều hướng dọc (Sidebar) bên trái: Thanh điều hướng được tổ chức thành các mục lớn với các chức năng liên quan được nhóm lại, giúp người dùng dễ dàng di chuyển giữa các phần của hệ thống. Các mục bao gồm:

- Logo: Nằm ở góc trên cùng bên trái, hiển thị logo "VINMEC - Healthcare System".
- Trang Chủ: Biểu tượng ngôi nhà và chữ "Trang Chủ", có vẻ là nút để quay về trang chính của hệ thống.
- Xem Lịch Làm Việc: Biểu tượng lịch và chữ "Xem Lịch Làm Việc".
- Báo cáo: Nhóm chức năng liên quan đến báo cáo, bao gồm:

- Báo cáo giờ dạy
- Báo cáo kết quả học
- Báo cáo khác
- Xem Đánh Giá: Biểu tượng sao và chữ "Xem Đánh Giá".
- Xem Lương: Nhóm chức năng liên quan đến lương, bao gồm:
 - Lịch sử nhận lương (đang được chọn và làm nổi bật)
 - Lương hiện tại
 - Mức thưởng và phụ cấp

II. Khu vực nội dung chính bên phải:

1. Thanh tiêu đề và thông tin người dùng:

- Nằm ở góc trên cùng bên phải, hiển thị "Xin chào: Nguyễn A", cho biết người dùng hiện tại đã đăng nhập vào hệ thống.

2. Thanh điều hướng ngang (Tab navigation):

- Gồm hai tab: "Quản Lý Thông Tin" và "Thông tin bác sĩ", cho phép người dùng chuyển đổi giữa các loại thông tin khác nhau. Tab "Quản lý lương (Tháng 9/2023)" là một phần của nội dung được hiển thị dưới tab "Quản Lý Thông Tin".

3. Khu vực quản lý lương:

- Tiêu đề: "Quản lý lương (Tháng 9/2023)", cho biết dữ liệu đang hiển thị là của tháng 9 năm 2023.
- Thanh tìm kiếm: Ô nhập liệu "Tìm kiếm" với biểu tượng kính lúp, cho phép người dùng tìm kiếm thông tin trong bảng lương.
- Bộ lọc "Lọc theo năm": Một ô chọn dạng dropdown (thanh cuộn xuống) với chữ "Lọc theo năm", cho phép người dùng lọc dữ liệu lương theo năm.
- Ô chọn ngày/thời gian: Một ô màu vàng với giá trị "08/10/2023 00:00" và biểu tượng lịch, có thể dùng để chọn một mốc thời gian cụ thể.

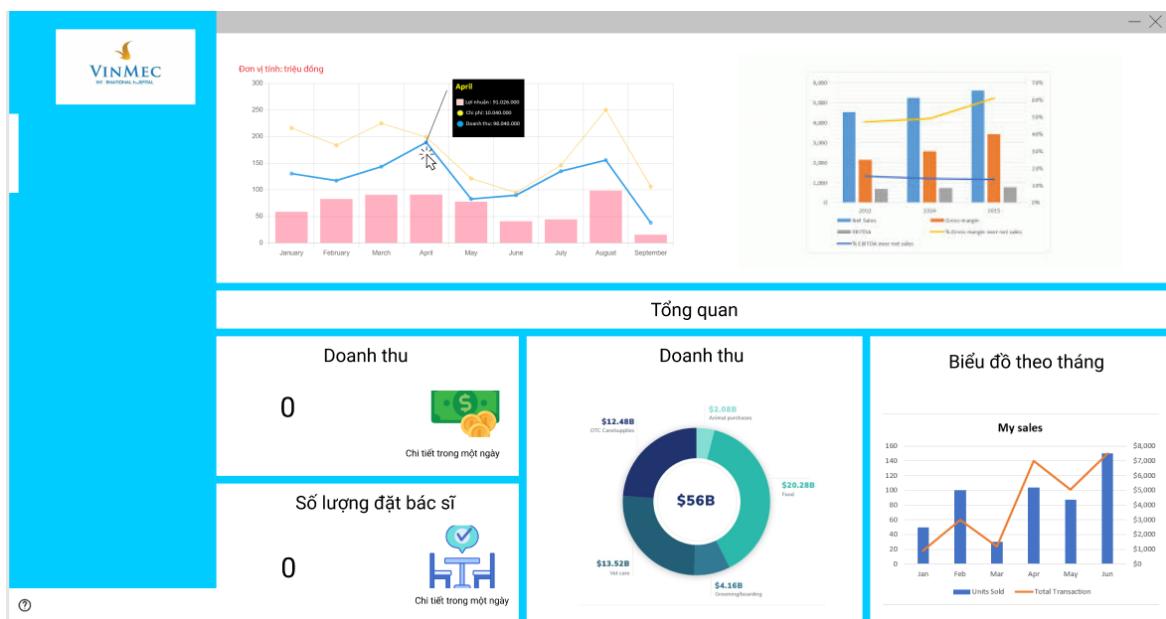
4. Bảng hiển thị thông tin lương: Bảng này liệt kê chi tiết các khoản lương theo tháng với các cột sau:

- Mã: Mã của người nhận lương (ví dụ: "longkhi").
- Lương tháng: Tháng và năm của khoản lương (ví dụ: "10/2023", "09/2023").
- Tiền lương: Số tiền lương nhận được (ví dụ: "120.000.000").

MÔ HÌNH HÓA YÊU CẦU VÀ API

- Ngày nhận lương: Ngày và thời gian mà khoản lương được nhận. Đối với tháng 10/2023, giá trị là "-", có thể hiểu là chưa nhận.
- Trạng Thái: Trạng thái của khoản lương.
- "Đang tính": Đối với lương tháng 10/2023.
- "Chưa nhận": Đối với lương tháng 09/2023.
- "Đã nhận": Đối với các khoản lương từ tháng 08/2023 trở về trước, được hiển thị bằng một nút màu xanh lá cây.
- Các nút hành động:
- Chi Tiết: Nút màu xanh dương cho phép xem thông tin chi tiết của khoản lương.
- Phản hồi: Nút màu tím cho phép gửi phản hồi về khoản lương. Nút này xuất hiện cho các khoản lương "Đang tính" và "Chưa nhận".

Tổng thể, giao diện này được thiết kế một cách khoa học và dễ sử dụng, cung cấp cái nhìn tổng quan và chi tiết về lịch sử nhận lương, đồng thời cho phép người dùng thực hiện các thao tác tìm kiếm, lọc và phản hồi liên quan đến thông tin lương của mình.



Hình 5: Giao diện Wireframe Trực quan hóa donah thu

Giao diện tổng quan này được thiết kế theo dạng bảng điều khiển (dashboard) với tông màu xanh dương chủ đạo, tạo cảm giác hiện đại và chuyên nghiệp. Bộ cục được chia thành nhiều phần, hiển thị các biểu đồ và số liệu thống kê quan trọng, giúp người dùng dễ dàng theo dõi hiệu suất tổng thể.

I. Thanh điều hướng dọc (Sidebar) bên trái:

- Logo: Nằm ở góc trên cùng bên trái, hiển thị logo "VINMEC - Healthcare System", biểu tượng nhận diện của hệ thống.
- Các mục điều hướng: Thanh điều hướng có màu xanh đậm hơn, nhưng các mục cụ thể không hiển thị rõ trong ảnh này, tuy nhiên có thể suy đoán đây là nơi chứa các liên kết đến các phần khác của hệ thống.

II. Khu vực nội dung chính bên phải:

1. Phần biểu đồ tổng thể:

- Tiêu đề: "Đơn vị tỉ triệu đồng", cho biết đơn vị tính của các số liệu trong biểu đồ.
- Biểu đồ đường và cột: Biểu đồ kết hợp hiển thị dữ liệu theo tháng từ tháng 1 đến tháng 9.
 - Các cột màu hồng nhạt: Biểu thị một loại dữ liệu (có thể là doanh thu hoặc số lượng).
 - Các đường kẻ (màu vàng và xanh dương): Biểu thị các chỉ số khác, có thể là xu hướng hoặc các dữ liệu liên quan.
- Khi di chuột (hiển thị con trỏ chuột và chú giải), biểu đồ hiển thị chi tiết dữ liệu của tháng "April" (tháng 4), bao gồm:

- "doanh thu": 234.334.334
- "SL bệnh nhân": 1.234
- "SL dịch vụ": 1.234
- "SL bác sĩ": 1.234

- Biểu đồ cột và đường nhỏ (góc trên bên phải): Một biểu đồ nhỏ hơn hiển thị dữ liệu theo năm (2012, 2013, 2014, 2015), có thể là biểu đồ so sánh doanh thu và chi phí hoặc các chỉ số khác theo thời gian.

2. Khu vực "Tổng quan": Phần này được chia thành các ô hiển thị các chỉ số quan trọng:

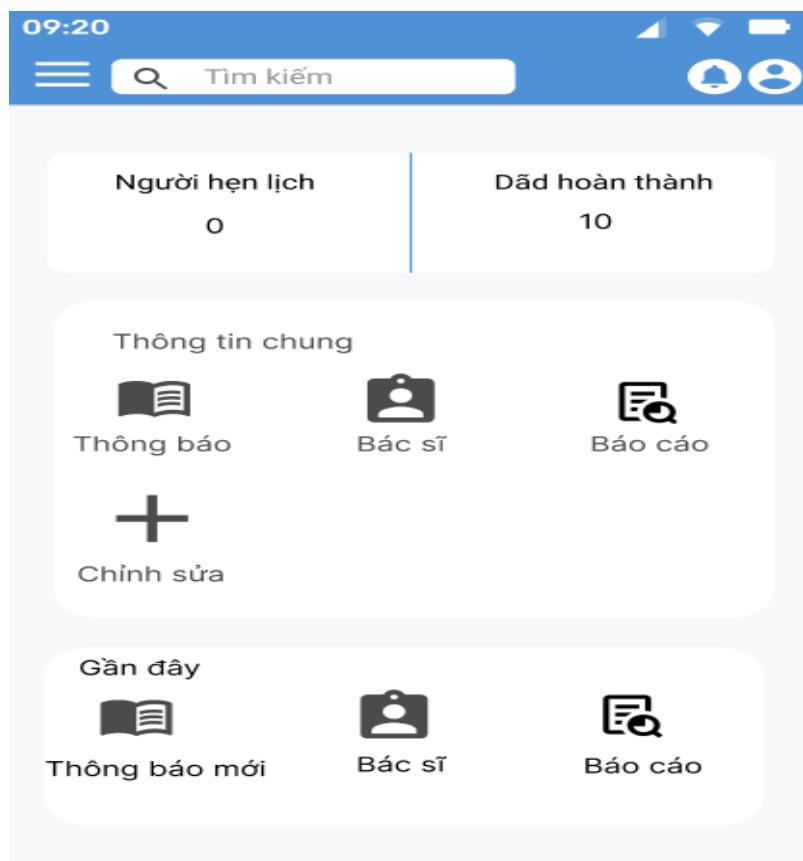
- Ô "Doanh thu":
 - Hiển thị số liệu "0" lớn, có thể là tổng doanh thu.
 - Biểu tượng tiền tệ màu xanh lá cây và dòng chữ "Chi tiết trong một ngày" gợi ý rằng đây là doanh thu trong ngày.
- Ô "Doanh thu" (biểu đồ tròn):

- Hiển thị biểu đồ hình tròn (donut chart) với giá trị trung tâm là "\$56B".
- Các phần của biểu đồ được chú thích bằng các con số và mô tả như "\$13.52B", "Outpatient", "Hospital purchase", "Food", "Housekeeping/Security", thể hiện tỷ lệ đóng góp của các nguồn doanh thu khác nhau.
- Ô "Biểu đồ theo tháng":
 - Tiêu đề "My sales".
 - Biểu đồ cột và đường (line chart) nhỏ hơn, hiển thị dữ liệu theo tháng (Jan, Feb, Mar, Apr, May, Jun, Jul) với các cột màu xanh đậm ("Units Sold") và đường màu cam ("Total Transaction").
- Ô "Số lượng bác sĩ":
 - Hiển thị số liệu "0" lớn.
 - Biểu tượng bàn ghế và dấu tick xanh, cùng dòng chữ "Chi tiết trong một ngày", cho thấy đây có thể là số lượng bác sĩ làm việc hoặc có mặt trong ngày.

III. Các nút điều khiển chung:

- Ở góc dưới cùng bên trái của giao diện, có biểu tượng cài đặt (bánh răng) và biểu tượng trợ giúp (dấu hỏi chấm), cho phép người dùng cấu hình hoặc tìm kiếm hỗ trợ. Tổng thể, giao diện này là một bảng điều khiển trực quan, cung cấp cái nhìn tổng quan về các hoạt động và hiệu suất của bệnh viện thông qua các biểu đồ và số liệu thống kê được sắp xếp khoa học.

MÔ HÌNH HÓA YÊU CẦU VÀ API



Hình 6: Giao diện Wireframe Màn hình chính

Giao diện tổng quan này được thiết kế theo dạng bảng điều khiển (dashboard) với tông màu xanh dương chủ đạo, tạo cảm giác hiện đại và chuyên nghiệp. Bộ cục được chia thành nhiều phần, hiển thị các biểu đồ và số liệu thống kê quan trọng, giúp người dùng dễ dàng theo dõi hiệu suất tổng thể.

I. Thanh điều hướng dọc (Sidebar) bên trái:

- Logo: Nằm ở góc trên cùng bên trái, hiển thị logo "VINMEC - Healthcare System", biểu tượng nhận diện của hệ thống.
- Các mục điều hướng: Thanh điều hướng có màu xanh đậm hơn, nhưng các mục cụ thể không hiển thị rõ trong ảnh này, tuy nhiên có thể suy đoán đây là nơi chứa các liên kết đến các phần khác của hệ thống.

II. Khu vực nội dung chính bên phải:

1. Phần biểu đồ tổng thể:

- Tiêu đề: "Đơn vị tỉ triệu đồng", cho biết đơn vị tính của các số liệu trong biểu đồ.
- Biểu đồ đường và cột: Biểu đồ kết hợp hiển thị dữ liệu theo tháng từ tháng 1 đến tháng 9.

- Các cột màu hồng nhạt: Biểu thị một loại dữ liệu (có thể là doanh thu hoặc số lượng).
- Các đường kẻ (màu vàng và xanh dương): Biểu thị các chỉ số khác, có thể là xu hướng hoặc các dữ liệu liên quan.
- Khi di chuột (hiển thị con trỏ chuột và chú giải), biểu đồ hiển thị chi tiết dữ liệu của tháng "April" (tháng 4), bao gồm:
 - "doanh thu": 234.334.334
 - "SL bệnh nhân": 1.234
 - "SL dịch vụ": 1.234
 - "SL bác sĩ": 1.234
- Biểu đồ cột và đường nhỏ (góc trên bên phải): Một biểu đồ nhỏ hơn hiển thị dữ liệu theo năm (2012, 2013, 2014, 2015), có thể là biểu đồ so sánh doanh thu và chi phí hoặc các chỉ số khác theo thời gian.

2. Khu vực "Tổng quan": Phần này được chia thành các ô hiển thị các chỉ số quan trọng:

- Ô "Doanh thu":
 - Hiển thị số liệu "0" lớn, có thể là tổng doanh thu.
 - Biểu tượng tiền tệ màu xanh lá cây và dòng chữ "Chi tiết trong một ngày" gợi ý rằng đây là doanh thu trong ngày.
- Ô "Doanh thu" (biểu đồ tròn):
 - Hiển thị biểu đồ hình tròn (donut chart) với giá trị trung tâm là "\$56B".
 - Các phần của biểu đồ được chú thích bằng các con số và mô tả như "\$13.52B", "Outpatient", "Hospital purchase", "Housekeeping/Security", thể hiện tỷ lệ đóng góp của các nguồn doanh thu khác nhau.
- Ô "Biểu đồ theo tháng":
 - Tiêu đề "My sales".
 - Biểu đồ cột và đường (line chart) nhỏ hơn, hiển thị dữ liệu theo tháng (Jan, Feb, Mar, Apr, May, Jun, Jul) với các cột màu xanh đậm ("Units Sold") và đường màu cam ("Total Transaction").
- Ô "Số lượng bác sĩ":
 - Hiển thị số liệu "0" lớn.

- Biểu tượng bàn ghế và dấu tick xanh, cùng dòng chữ "Chi tiết trong một ngày", cho thấy đây có thể là số lượng bác sĩ làm việc hoặc có mặt trong ngày.

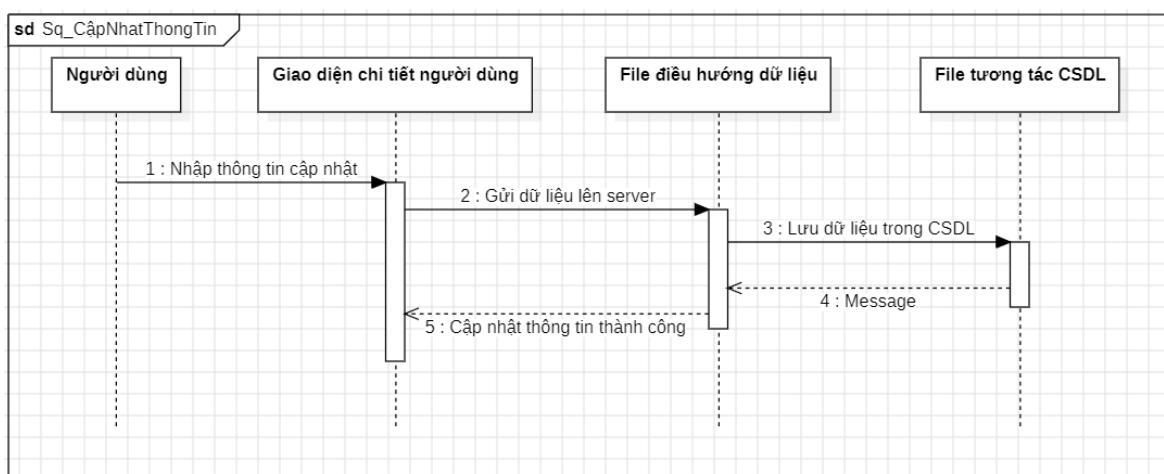
III. Các nút điều khiển chung:

- Ở góc dưới cùng bên trái của giao diện, có biểu tượng cài đặt (bánh răng) và biểu tượng trợ giúp (dấu hỏi chấm), cho phép người dùng cấu hình hoặc tìm kiếm hỗ trợ. Tổng thể, giao diện này là một bảng điều khiển trực quan, cung cấp cái nhìn tổng quan về các hoạt động và hiệu suất của bệnh viện thông qua các biểu đồ và số liệu thống kê được sắp xếp khoa học.

4.7. Sơ đồ tuần tự (Sequence Diagram)

4.7.1. Sơ đồ tuần tự [Cập nhật thông tin cá nhân]

- Mô tả: Người dùng chỉnh sửa thông tin cá nhân trong giao diện chi tiết người dùng
- Sơ đồ tuần tự:

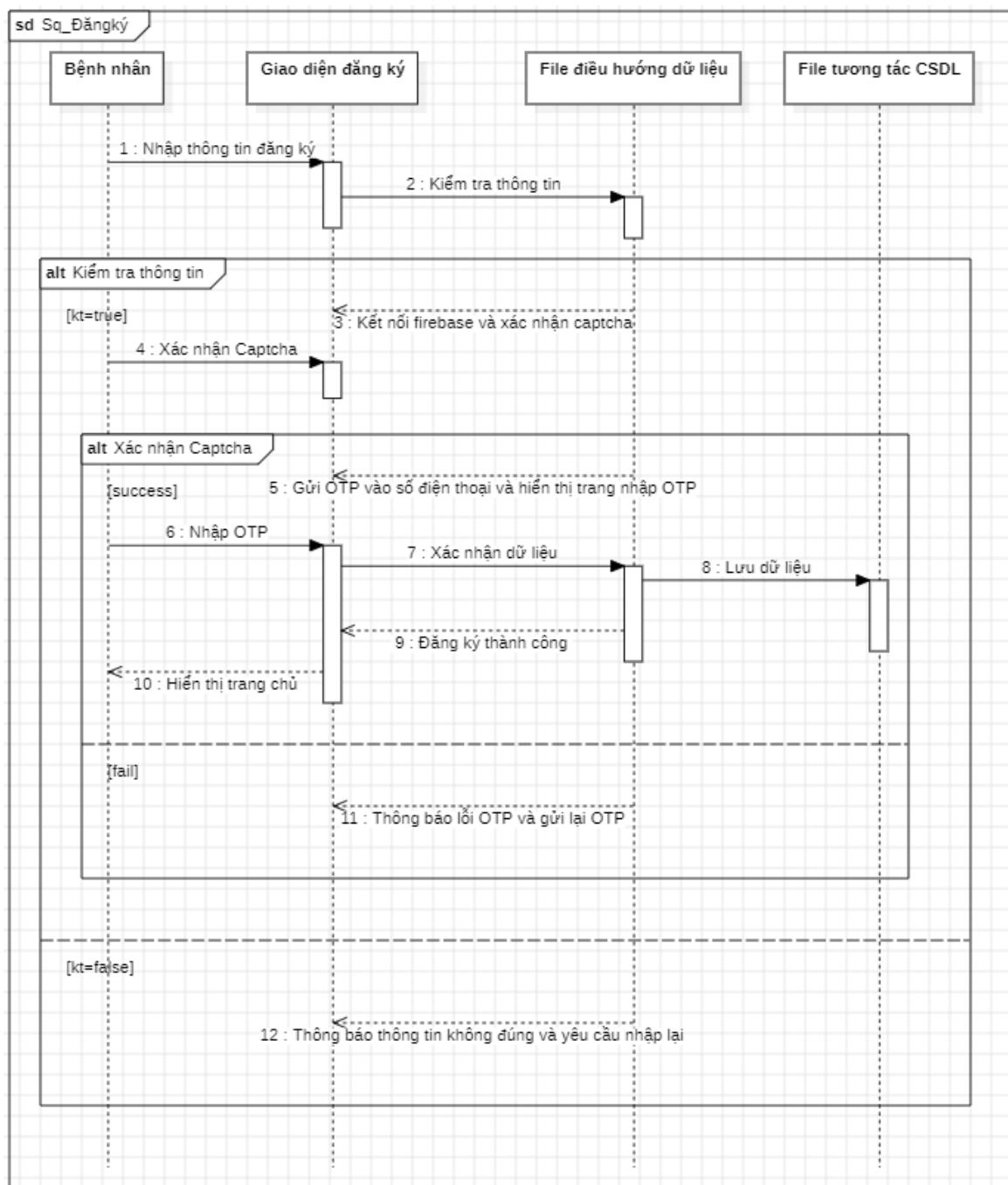


Hình 4.7: Sequence Diagram [Cập nhật thông tin cá nhân]

4.7.2. Sơ đồ tuần tự [Đăng ký tài khoản]

- Mô tả: Người đăng ký tài khoản để khám bệnh và xác thực bằng OTP
- Sơ đồ tuần tự:

MÔ HÌNH HÓA YÊU CẦU VÀ API

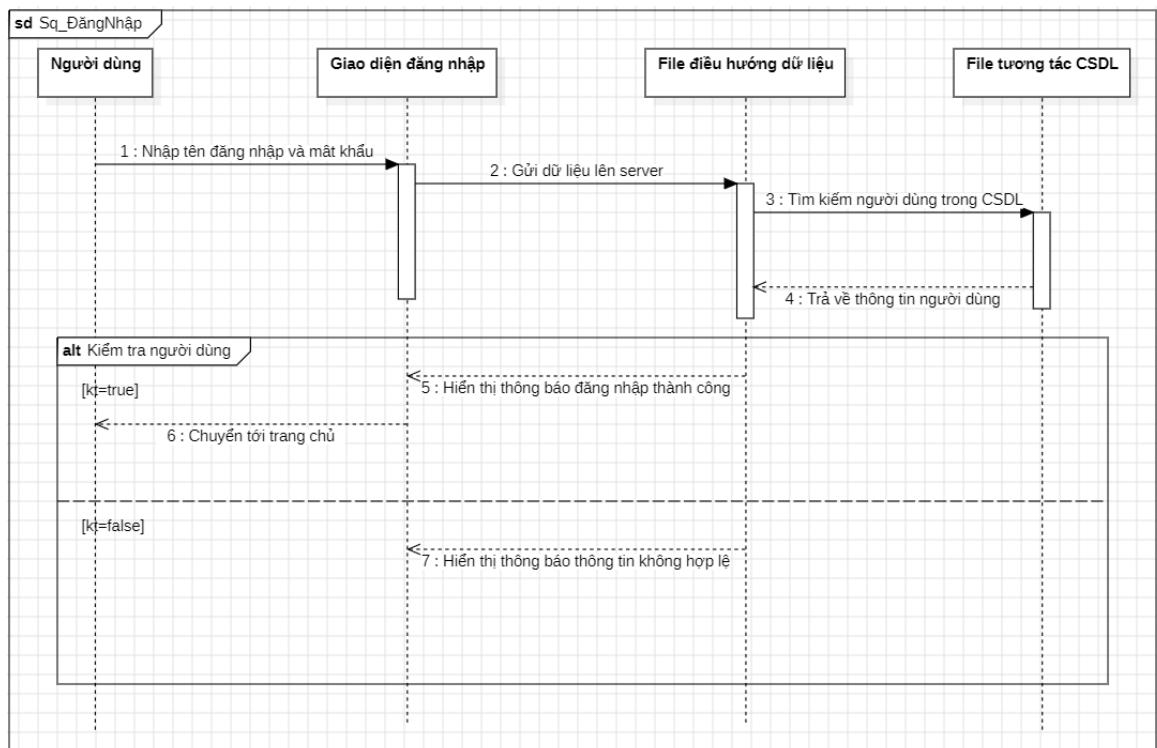


Hình 4.8: Sequence Diagram [Đăng ký tài khoản]

4.7.3. Sơ đồ tuần tự [Đăng nhập]

- Mô tả: Người dùng đăng nhập vào web để khám bệnh
- Sơ đồ tuần tự:

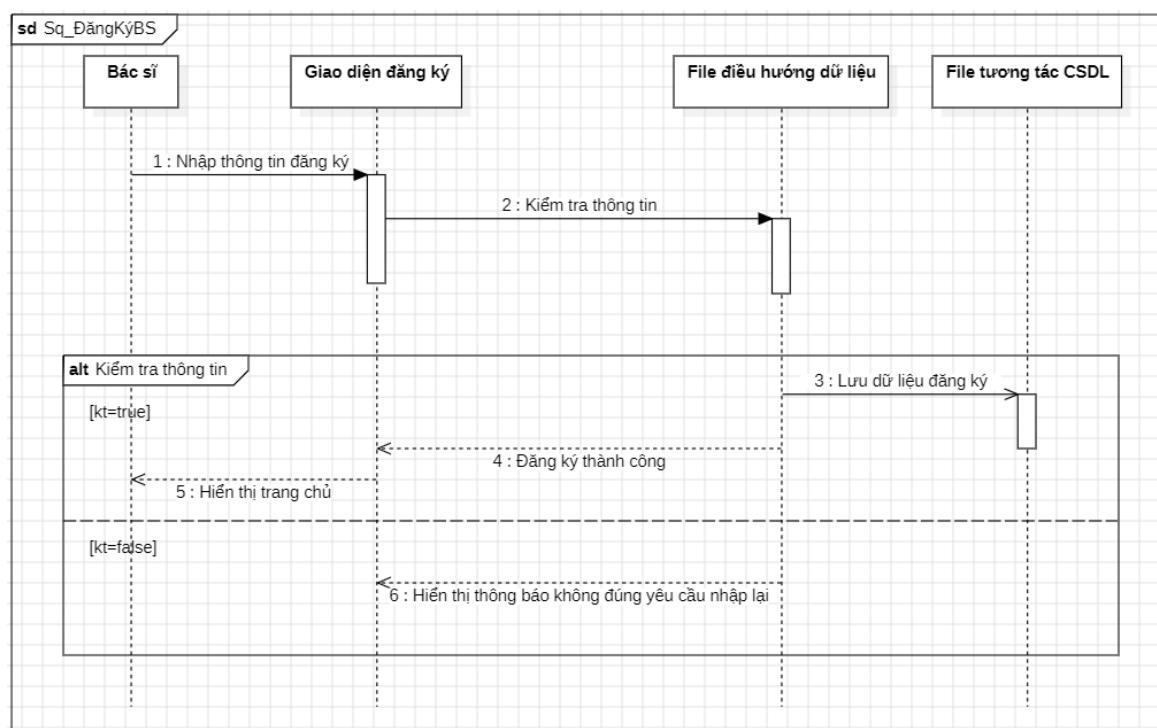
MÔ HÌNH HÓA YÊU CẦU VÀ API



Hình 4.9: Sequence Diagram [Đăng nhập]

4.7.4. Sơ đồ tuần tự [Đăng ký bác sĩ]

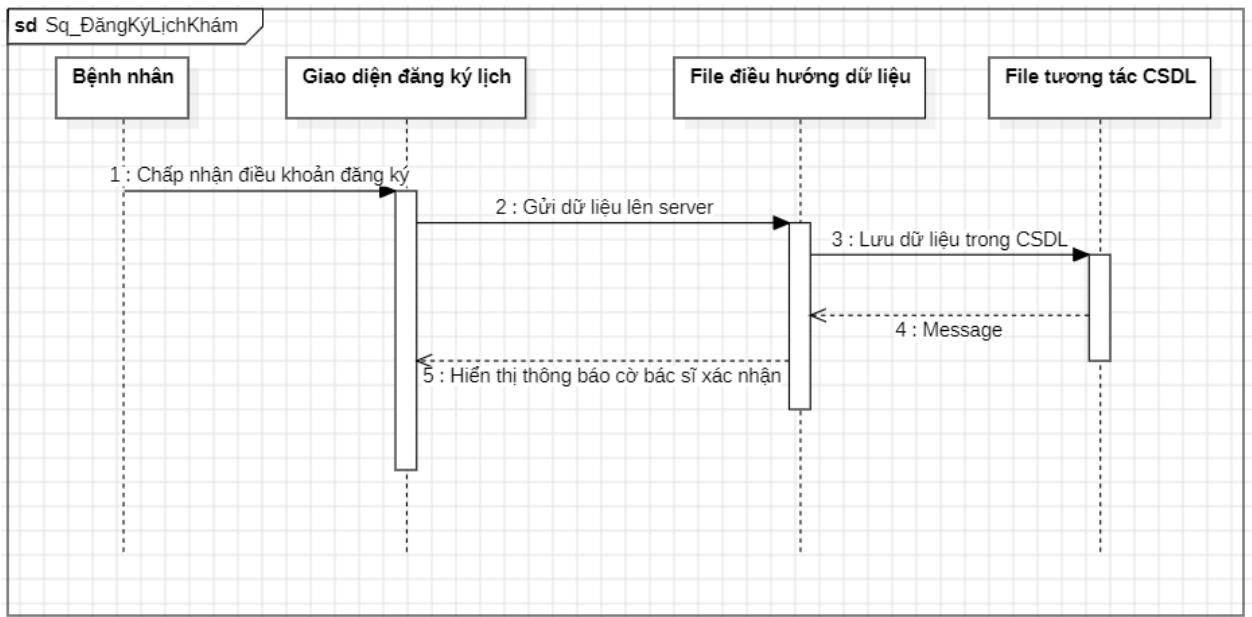
- Mô tả: Bác sĩ đăng ký tài khoản để khám bệnh
- Sơ đồ tuần tự:



Hình 4.10: Sequence Diagram [Đăng ký bác sĩ]

4.7.5. Sơ đồ tuần tự [Đăng ký lịch khám]

- Mô tả: Bệnh nhân đăng ký lịch khám với bác sĩ và đợi xác nhận từ bác sĩ
- Sơ đồ tuần tự:

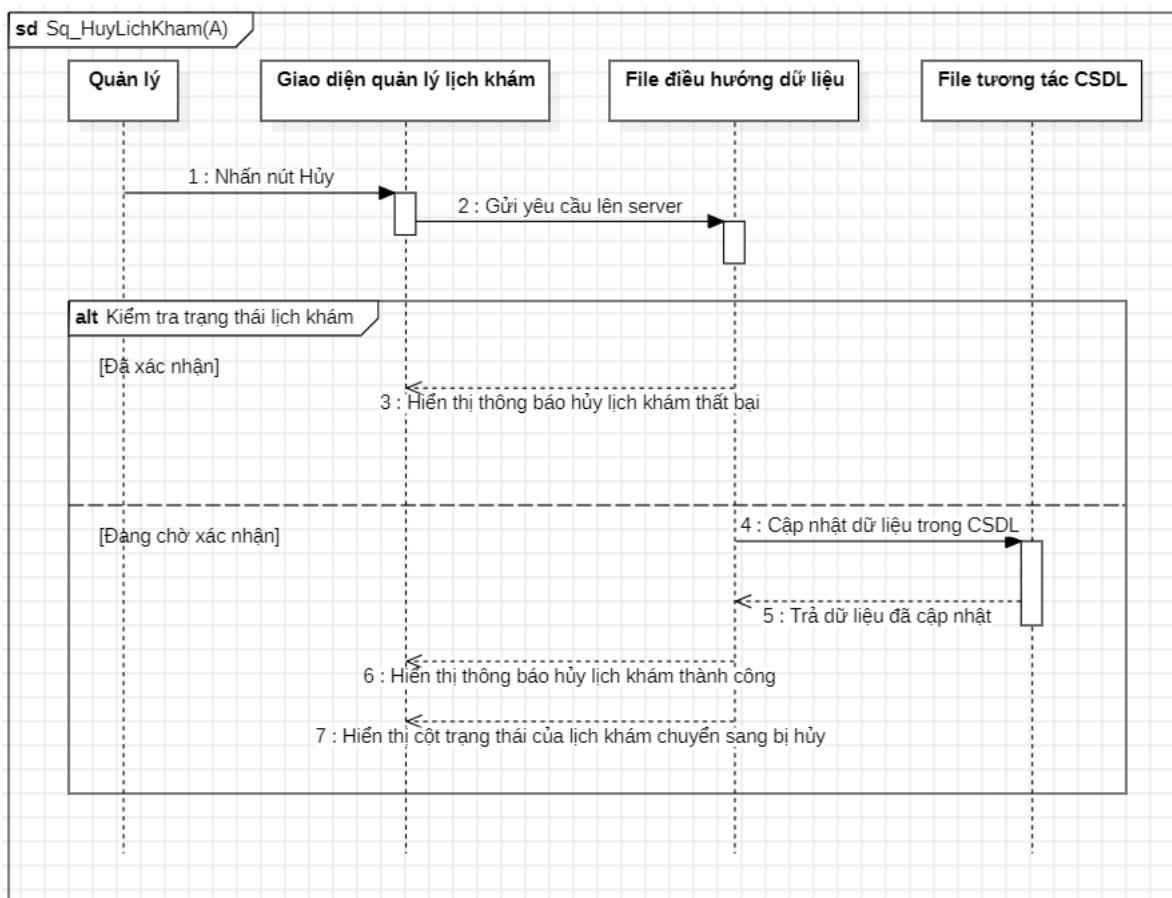


Hình 4.11: Sequence Diagram [Đăng ký lịch khám]

4.7.6. Sơ đồ tuần tự [Hủy lịch khám]

- Mô tả: Bệnh nhân hủy lịch khám khi trạng thái cuộc hẹn đang ở trạng thái chờ bác sĩ xác nhận
- Sơ đồ tuần tự:

MÔ HÌNH HÓA YÊU CẦU VÀ API

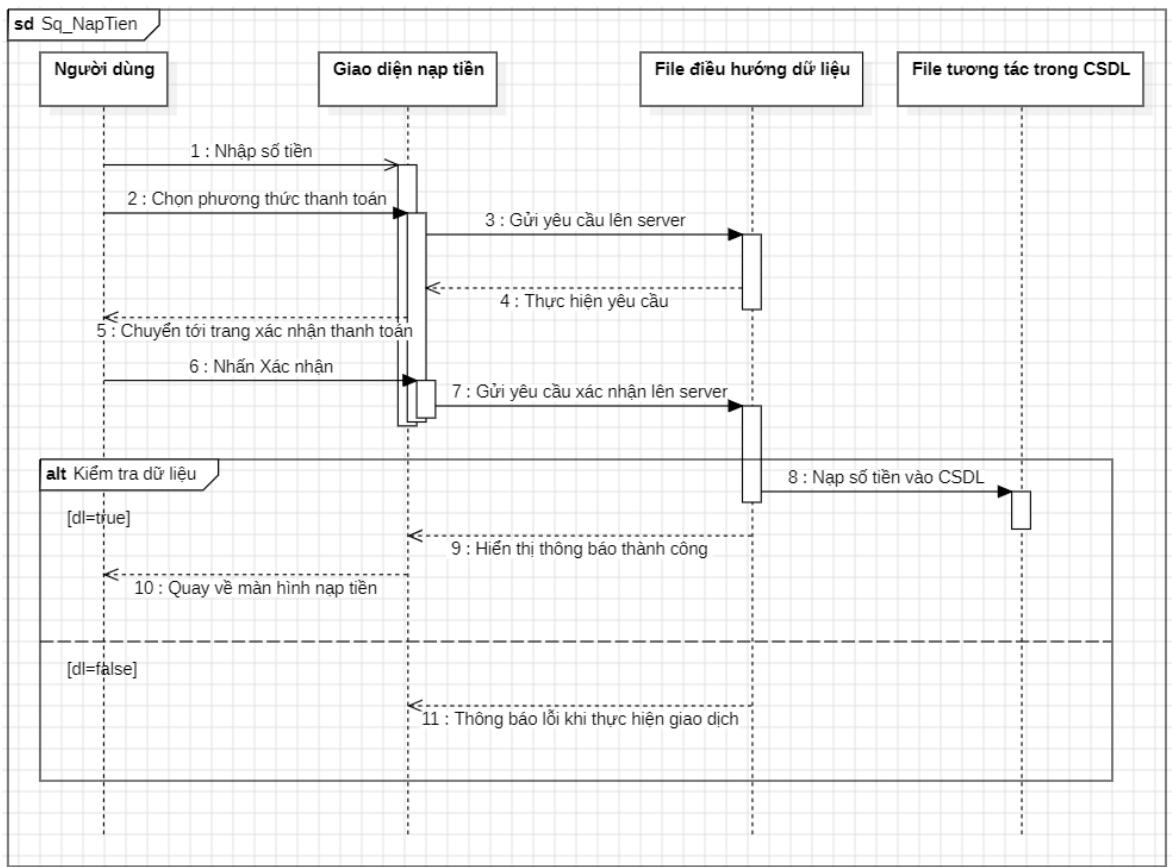


Hình 4.12: Sequence Diagram [Hủy lịch khám]

4.7.7. Sơ đồ tuần tự [Nạp tiền]

- Mô tả: Bệnh nhân thực hiện nạp tiền vào tài khoản
- Sơ đồ tuần tự:

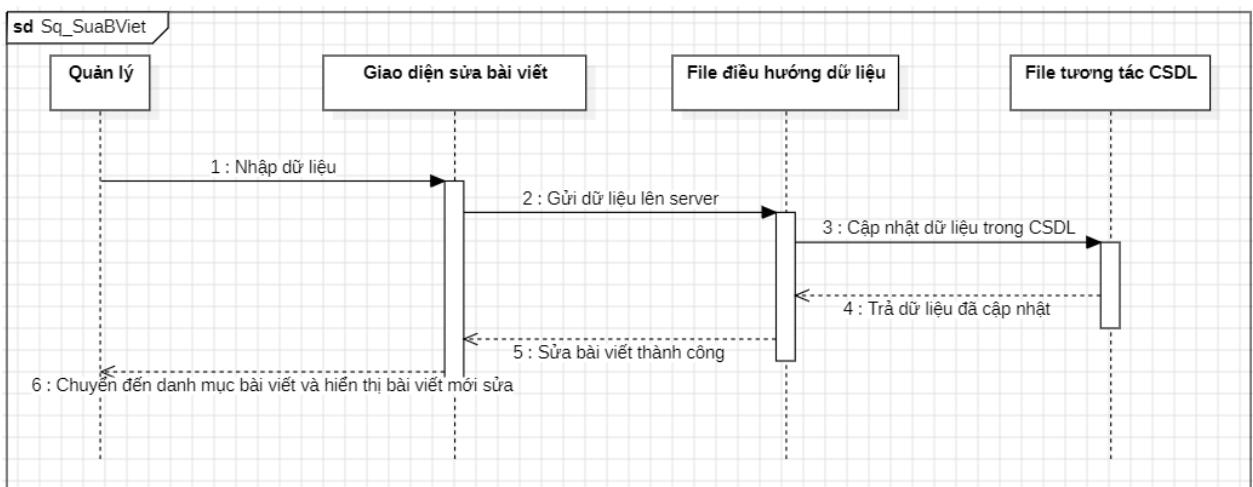
MÔ HÌNH HÓA YÊU CẦU VÀ API



Hình 4.13: Sequence Diagram [Nạp tiền]

4.7.8. Sơ đồ tuần tự [Sửa bài viết]

- Mô tả: Quản lý sửa bài viết đã đăng
- Sơ đồ tuần tự:

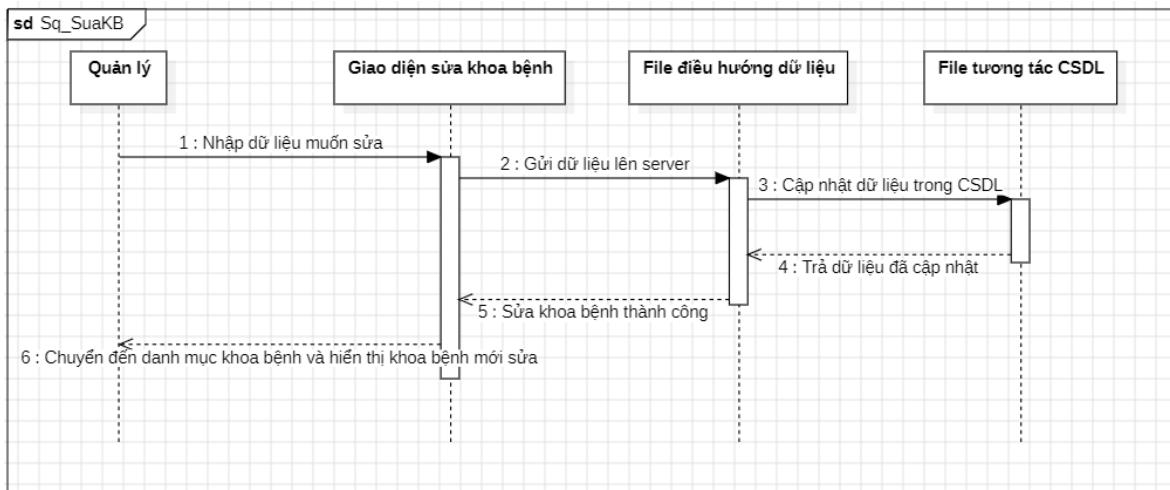


Hình 4.14: Sequence Diagram [Sửa bài viết]

MÔ HÌNH HÓA YÊU CẦU VÀ API

4.7.9. Sơ đồ tuần tự [Sửa khoa bệnh]

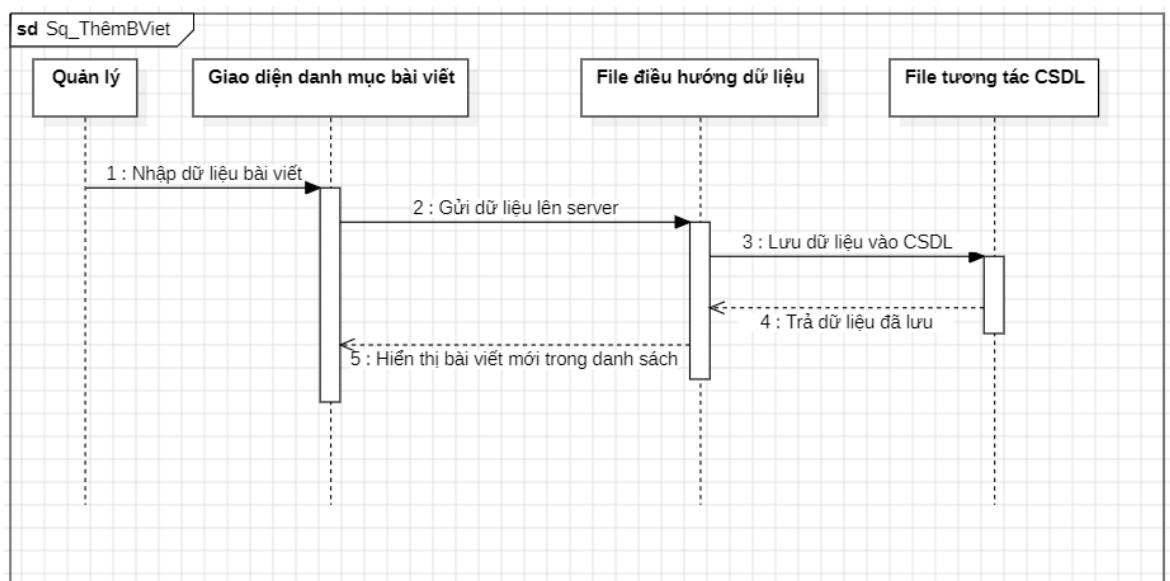
- Mô tả: Quản lý sửa khoa bệnh đã đăng
- Sơ đồ tuần tự:



Hình 4.15: Sequence Diagram [Sửa khoa bệnh]

4.7.10. Sơ đồ tuần tự [Thêm bài viết]

- Mô tả: Quản lý thực hiện thêm bài viết
- Sơ đồ tuần tự:

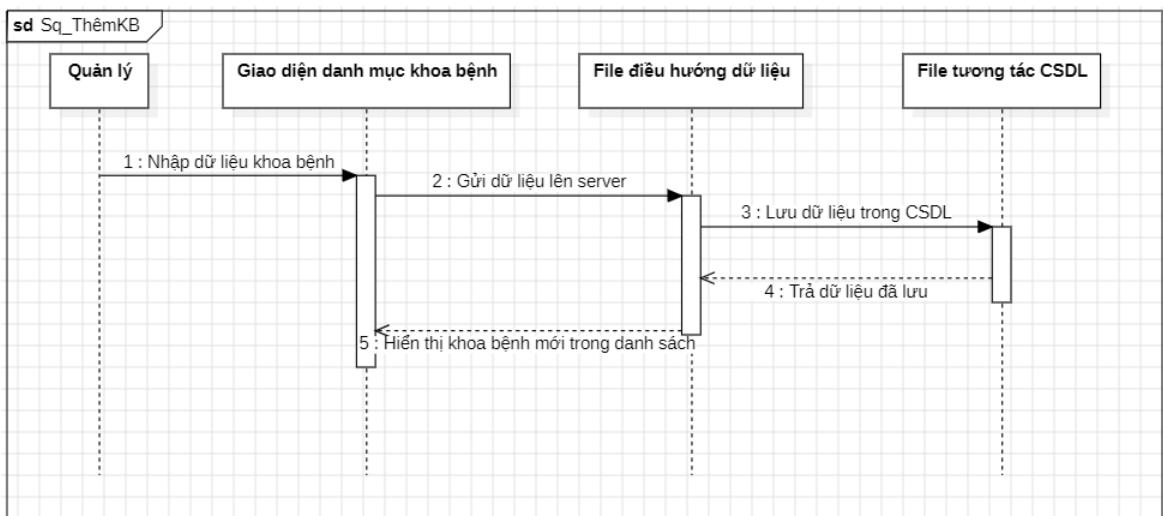


Hình 4.16: Sequence Diagram [Thêm bài viết]

4.7.11. Sơ đồ tuần tự [Thêm Khoa bệnh]

- Mô tả: Quản lý thực hiện thêm khoa bệnh
- Sơ đồ tuần tự:

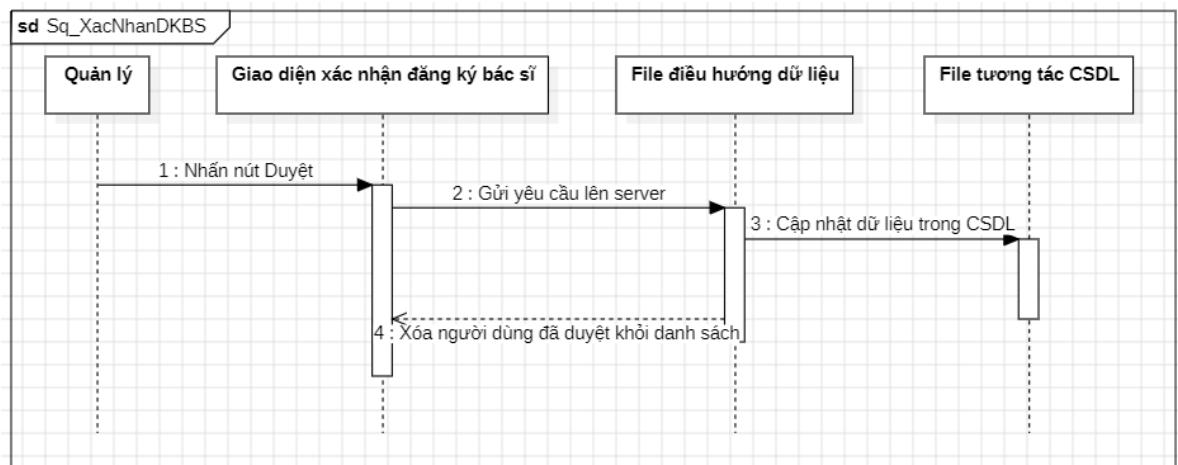
MÔ HÌNH HÓA YÊU CẦU VÀ API



Hình 4.17: Sequence Diagram [Thêm khoa bệnh]

4.7.12. Sơ đồ tuần tự [Xác nhận đăng ký bác sĩ]

- Mô tả: Quản lý xác nhận đơn đăng ký làm bác sĩ
- Sơ đồ tuần tự:

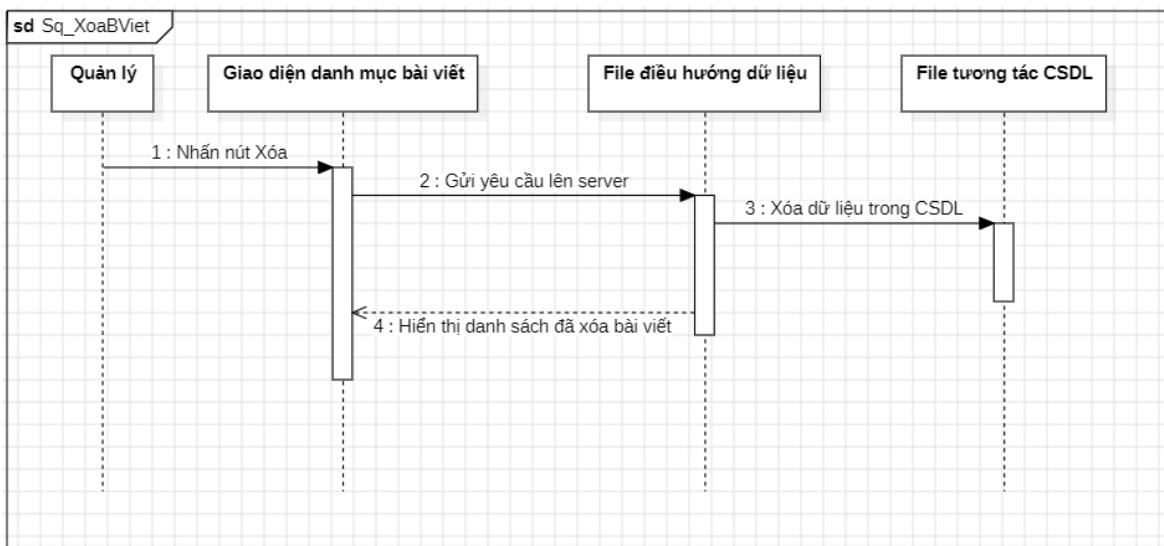


Hình 4.18: Sequence Diagram [Xác nhận đăng ký bác sĩ]

4.7.13. Sơ đồ tuần tự [Xóa bài viết]

- Mô tả: Quản lý xóa bài viết đã đăng
- Sơ đồ tuần tự:

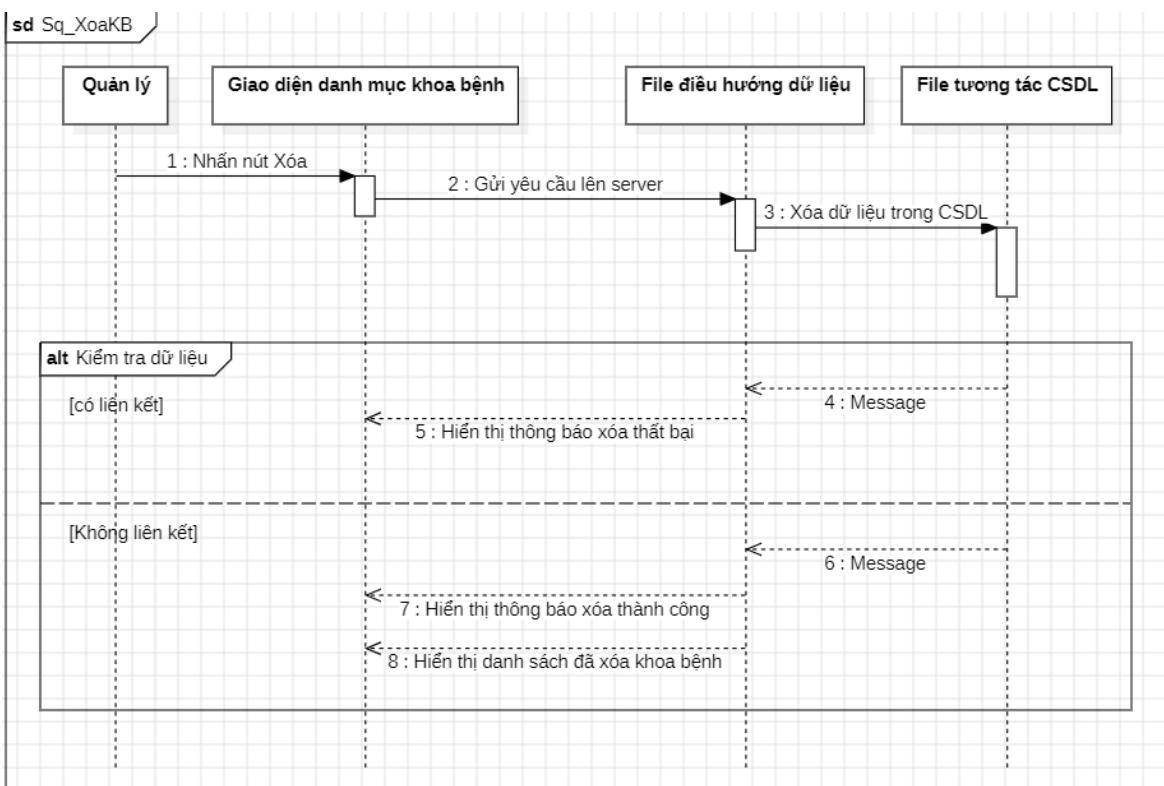
MÔ HÌNH HÓA YÊU CẦU VÀ API



Hình 4.19: Sequence Diagram [Xóa bài viết]

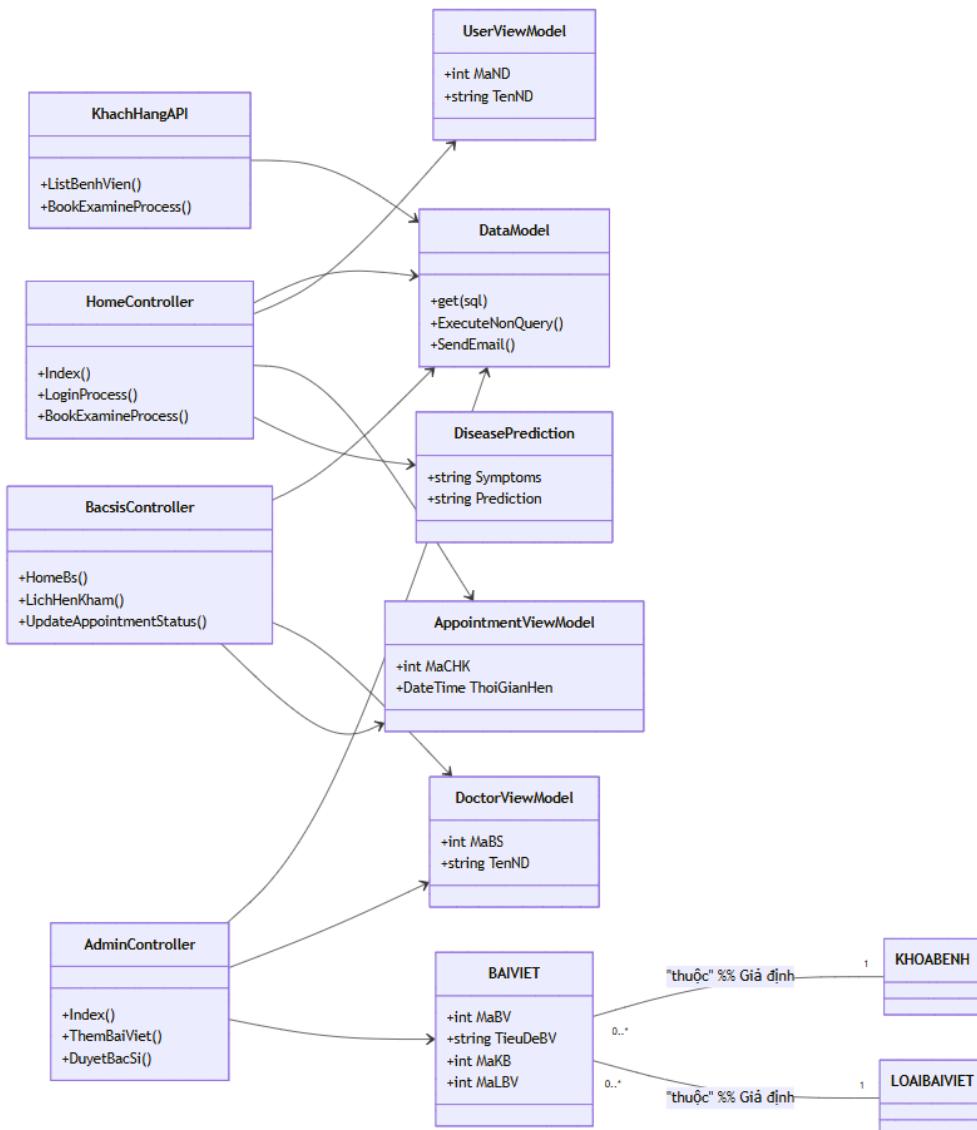
4.7.14. Sơ đồ tuần tự [xóa khoa bệnh]

- Mô tả: Quản lý xóa khoa bệnh đã đăng
- Sơ đồ tuần tự:

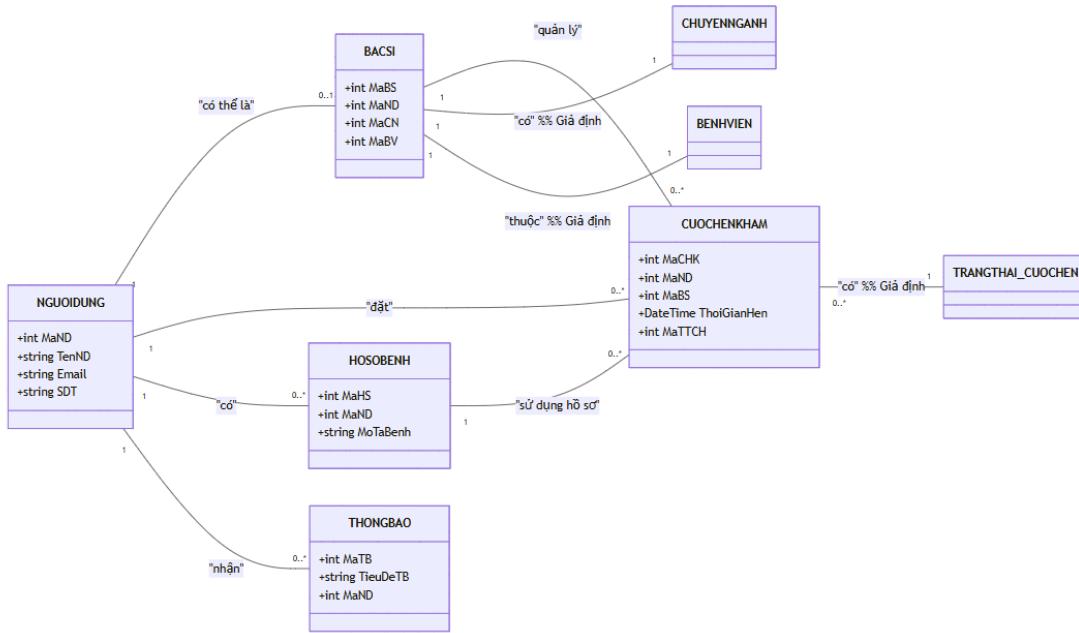


Hình 4.20: Sequence Diagram [Xóa khoa bệnh]

4.8. Class Diagram



Hình 21 Class Diagram



Hình 22 Class Diagram

Class Diagram này cung cấp một cái nhìn tổng quan súc tích về cấu trúc và mối quan hệ giữa các thành phần cốt lõi trong hệ thống website đặt lịch khám bệnh. Mặc dù đã được tối giản các chi tiết, nó vẫn truyền tải được bản chất kiến trúc phân lớp của hệ thống một cách hiệu quả.

4.8.1. Tổng quan Kiến trúc

Sơ đồ được tổ chức thành bốn nhóm lớp chính, mỗi nhóm đảm nhiệm một vai trò cụ thể trong hệ thống:

- Controllers: Các lớp chịu trách nhiệm xử lý logic nghiệp vụ và điều phối tương tác với người dùng.
- Models: Chứa các lớp xử lý logic nghiệp vụ phức tạp, truy cập dữ liệu và định nghĩa các đối tượng chính của hệ thống.
- ViewModels: Các đối tượng được thiết kế đặc biệt để chuyển đổi và truyền tải dữ liệu từ lớp Models đến giao diện người dùng (View).
- Database Entities: Đại diện trực tiếp cho cấu trúc các bảng trong cơ sở dữ liệu, nơi lưu trữ toàn bộ thông tin của hệ thống.

Sự phân chia này giúp hệ thống trở nên có tổ chức, dễ bảo trì và mở rộng trong tương lai.

4.8.2. Vai trò của từng nhóm lớp chính

1. Nhóm Controllers Các lớp trong nhóm này đóng vai trò "người điều hành", tiếp nhận yêu cầu từ người dùng và quyết định hành động tiếp theo.

- HomeController: Xử lý các chức năng quan trọng dành cho người dùng thông thường, bao gồm việc hiển thị trang chủ, xử lý quá trình đăng nhập và quản lý quy trình đặt lịch khám.

- AdminController: Đảm nhiệm các tác vụ quản trị hệ thống, như quản lý bài viết (thêm, sửa, xóa) và duyệt thông tin của bác sĩ.

- BacsisController: Cung cấp các chức năng chuyên biệt cho bác sĩ, bao gồm trang quản lý cá nhân, xem lịch hẹn khám và cập nhật trạng thái các cuộc hẹn.

- KhachHangAPI: Đây là một lớp API được thiết kế để phục vụ các ứng dụng khách hàng (ví dụ mobile app), cho phép họ xem danh sách bệnh viện và thực hiện quy trình đặt lịch khám một cách tự động.

2. Nhóm Models Nhóm Models là nơi chứa đựng các logic cốt lõi và tương tác trực tiếp với tầng dữ liệu hoặc các dịch vụ bên ngoài.

- DataModel: Đóng vai trò là lớp truy cập dữ liệu chính. Nó cung cấp các phương thức để thực hiện các truy vấn dữ liệu (get(sql)), thực thi các lệnh thao tác dữ liệu (ExecuteNonQuery()) và thậm chí cả việc gửi email (SendEmail()), giúp các Controllers không cần quan tâm đến chi tiết về database.

- DiseasePrediction: Đại diện cho một module hoặc dịch vụ liên quan đến việc dự đoán bệnh dựa trên các triệu chứng. Các thuộc tính như Symptoms và Prediction là những thông tin cốt lõi mà module này xử lý.

3. Nhóm ViewModels ViewModels là các "khung nhìn" dữ liệu được tùy chỉnh, giúp tối ưu hóa việc truyền tải thông tin đến giao diện người dùng mà không cần gửi toàn bộ đối tượng nghiệp vụ phức tạp.

- UserViewModel: Chứa các thông tin cơ bản của người dùng (Mã, Tên) cần thiết để hiển thị trên giao diện người dùng.

- DoctorViewModel: Tương tự, cung cấp thông tin cốt lõi về bác sĩ (Mã bác sĩ, Tên) để hiển thị.

- AppointmentViewModel: Tập hợp các dữ liệu cần thiết của một cuộc hẹn (Mã cuộc hẹn, Thời gian hẹn) để trình bày trên các màn hình liên quan.

4. Nhóm Database Entities Đây là những lớp phản ánh trực tiếp cấu trúc của các bảng trong cơ sở dữ liệu, đóng vai trò là nơi lưu trữ toàn bộ dữ liệu của hệ thống.

- NGUOIDUNG: Lưu trữ thông tin chung của tất cả người dùng hệ thống, là nền tảng cho các loại tài khoản khác.
- BACSI: Chứa thông tin chi tiết về các bác sĩ, bao gồm cả các liên kết đến thông tin chuyên ngành và bệnh viện công tác.
- CUOCHENKHAM: Ghi lại thông tin của từng cuộc hẹn, bao gồm người đặt, bác sĩ, thời gian và trạng thái hiện tại.
- HOSOBENH: Lưu trữ hồ sơ bệnh lý của từng người dùng, giúp bác sĩ có cái nhìn tổng quan về lịch sử sức khỏe bệnh nhân.
- BAIVIET: Quản lý các bài viết (ví dụ: tin tức, kiến thức y tế) trên hệ thống, được phân loại theo khoa bệnh và loại bài viết.
- THONGBAO: Lưu trữ các thông báo được gửi đến người dùng, đảm bảo thông tin luôn được truyền tải hiệu quả.

4.8.3. Mối quan hệ giữa các Class

Sơ đồ thể hiện rõ hai loại mối quan hệ chính:

- Mối quan hệ Phụ thuộc/Sử dụng (-->): Các lớp Controllers là trung tâm của việc tương tác, chúng sử dụng (phụ thuộc vào) DataModel để thực hiện các thao tác dữ liệu, và sử dụng các ViewModels để chuẩn bị dữ liệu gửi về giao diện. Chẳng hạn, HomeController cần DiseasePrediction để thực hiện chức năng dự đoán. Điều này cho thấy luồng điều khiển và trao đổi dữ liệu giữa các tầng của ứng dụng.
- Mối quan hệ Hiệp hội (Associations) giữa các Database Entities ("bản_số_A" -- "bản_số_B" ClassB : "tên_mối_quan_hệ"): Các mối quan hệ này minh họa cách các bảng trong cơ sở dữ liệu được liên kết với nhau, thể hiện logic nghiệp vụ:
 - NGUOIDUNG là một thực thể trung tâm, có thể liên kết với BACSI (một người dùng có thể là hoặc không là bác sĩ), "đặt" nhiều CUOCHENKHAM, "có" nhiều HOSOBENH, và "nhận" nhiều THONGBAO.
 - BACSI "quản lý" nhiều CUOCHENKHAM và liên kết với một CHUYENNGANH cũng như một BENHVIEN cụ thể.
 - HOSOBENH có thể được "sử dụng trong" nhiều CUOCHENKHAM.

- Mỗi CUOCHEKHAM "có" một TRANGTHAI_CUOCHEKHAM duy nhất (ví dụ: Đã xác nhận, Đã hủy).
- Các BAIVIET "thuộc" về một KHOABENH và một LOAIBAIVIET nhất định. Các "bản số" (như "1", "0..1", "0..*") chỉ rõ số lượng đối tượng tham gia vào mỗi phía của mối quan hệ, giúp xác định ràng buộc về tính toàn vẹn dữ liệu.

4.8.4. Kết luận

Class Diagram rút gọn này là một công cụ mạnh mẽ để hình dung kiến trúc tổng thể của hệ thống đặt lịch khám bệnh. Nó làm nổi bật các thành phần chính và cách chúng tương tác, cung cấp một cái nhìn rõ ràng và đủ chi tiết cho các mục đích báo cáo, thảo luận thiết kế mà không làm người xem bị quá tải bởi thông tin chi tiết không cần thiết trong giai đoạn tổng quan.

4.9. Các API sử dụng trong chương trình

Hệ thống sử dụng kiến trúc hiện đại, phân chia thành hai nhóm chính: API nội bộ (Internal APIs) và API bên ngoài (External APIs). Mỗi nhóm đảm nhiệm các chức năng riêng, hỗ trợ hoạt động đầy đủ và hiệu quả cho hệ thống quản lý người dùng, bác sĩ, lịch hẹn, và các dịch vụ hỗ trợ như AI và Email.

4.9.1. Internal APIs (API nội bộ)

1.1 Nhóm API xác thực người dùng

• Đăng nhập người dùng

- Endpoint: POST /Home/LoginProcess
- Tham số: Số điện thoại (10–11 số), mật khẩu (6–50 ký tự)
- Stored Procedure: CheckLogin
- Phản hồi: Thông tin người dùng và URL chuyển hướng hoặc thông báo lỗi

• Đăng nhập bác sĩ

- Endpoint: POST /Bacsis/KTBS
- Tham số: Tên đăng nhập (email/số điện thoại), mật khẩu
- Stored Procedure: CheckLoginBSs
- Phản hồi: Thông tin bác sĩ và URL chuyển hướng

• Đăng nhập quản trị viên

- Endpoint: POST /Admin/XuLyAdminLogin
- Tham số: Số điện thoại, mật khẩu

- Phản hồi: Thông tin quản trị viên và URL chuyển hướng

1.2 Nhóm API quản lý người dùng

- **Đăng ký tài khoản**

- Endpoint: POST /Home/RegisterProcess
- Tham số: Họ tên (2–100 ký tự), mật khẩu (6–50 ký tự), số điện thoại (độc nhất), email (độc nhất)
- Phản hồi: Thông báo đăng ký thành công hoặc thất bại

- **Cập nhật thông tin cá nhân**

- Endpoint: POST /Home/UpdateUserInfo
- Tham số: Thông tin cá nhân, ảnh đại diện (tối đa 2MB)
- Stored Procedure: UpdateUserInfo

- **Đổi mật khẩu**

- Endpoint: POST /Home/ChangePasswordProcess
- Tham số: Mật khẩu hiện tại, mật khẩu mới, xác nhận mật khẩu
- Stored Procedures: CheckCurrentPassword, UpdatePassword

1.3 Nhóm API đặt lại mật khẩu

- **Gửi mã OTP**

- Endpoint: POST /Home/ForgotPasswordProcess
- Tham số: Email
- Stored Procedure: CreateResetToken
- Phản hồi: Thông báo gửi mã OTP

- **Xác thực OTP**

- Endpoint: POST /Home/VerifyEmailProcess
- Tham số: Email, mã OTP (6 chữ số)
- Stored Procedure: ValidateResetToken

- **Đặt lại mật khẩu**

- Endpoint: POST /Home/ResetPasswordProcess
- Tham số: Email, token, mật khẩu mới, xác nhận mật khẩu
- Stored Procedure: ResetUserPassword

1.4 Nhóm API quản lý bác sĩ

- **Lấy danh sách bác sĩ**

MÔ HÌNH HÓA YÊU CẦU VÀ API

- Endpoint: GET /KhachHangAPI/BacSi

- Stored Procedure: getAllBacSi

- **Lấy top 7 bác sĩ**

- Endpoint: GET /Home/Index

- Stored Procedure: GetTop7Doctors

- **Lấy chi tiết bác sĩ**

- Endpoint: GET /Home/DetailDoctor?MaBS={MaBS}

- Stored Procedure: DetDETAILBACSI1

- **Tìm kiếm bác sĩ theo tiêu chí**

- Endpoint: POST /Home/FillterDoctorList

- **Đăng ký làm bác sĩ**

- Endpoint: POST /Bacsis/ThucHienDKBs

1.5 Nhóm API quản lý lịch hẹn

- **Đặt lịch khám**

- Endpoint: POST /Home/BookExamineProcess

- Tham số: Mã bác sĩ, mô tả bệnh, hình ảnh (tối đa 5 file, 5MB mỗi file)

- Stored Procedures: SAVEHOSO, SAVEHINHANHBENH,

- SAVECUOCHECHENKHAM

- **Lấy lịch sử khám**

- Endpoint: GET /Home/ExamineHistory

- **Hủy lịch hẹn**

- Endpoint: GET /Home/Cancel?maCHK={maCHK}

1.6 Nhóm API dashboard bác sĩ

- **Lấy danh sách lịch hẹn chờ xác nhận**

- Endpoint: GET /Bacsis/GetAllUnconfirmedAppointments

- **Cập nhật trạng thái lịch hẹn**

- Endpoint: POST /Bacsis/UpdateAppointmentStatus

- Trạng thái: 1 = Chờ xác nhận, 2 = Đã xác nhận, 3 = Hoàn thành, 4 = Hủy

1.7 Nhóm API quản trị hệ thống

- **Thêm bài viết**

- Endpoint: POST /Admin/ThemBaiViet

- **Duyệt đăng ký bác sĩ**
 - Endpoint: POST /Admin/DuyetBacSi
 - **Gửi thông báo**
 - Endpoint: POST /Admin/GuiThongBao
 - **Hoàn phí khám**
 - Endpoint: POST /Admin/HoanPhiKham
- 1.8 Nhóm API nội dung
- **Lấy danh sách bệnh viện**
 - Endpoint: GET /KhachHangAPI/Index
 - Stored Procedure: getAllBenhVien
 - **Lấy danh sách khoa bệnh**
 - Endpoint: GET /KhachHangAPI/KhoaBenh
 - Stored Procedure: getAllKhoaBenh
 - **Lấy bài viết theo khoa bệnh**
 - Endpoint: GET /Home/Article?MaBV={MaBV}
 - Stored Procedure: getBaiVietByIDMaKhoaBenh

4.9.2. External APIs (API bên ngoài)

2.1 API dự đoán bệnh bằng AI

- **Dự đoán bệnh**
 - Endpoint: POST http://localhost:8000/predict
 - Gửi mô tả triệu chứng dạng JSON
 - Phản hồi: Tên bệnh được hệ thống AI dự đoán

2.2 API gửi email SMTP

- **Gửi email**
 - Cấu hình trong tệp appsettings.json
 - Các mẫu email bao gồm: Email chào mừng và Email OTP đặt lại mật khẩu

4.9.3. Bảo mật, hiệu suất và xử lý lỗi

3.1 Bảo mật

- Sử dụng xác thực dựa trên session (thời gian hết hạn: 30 phút)
- Kiểm tra đầu vào chặt chẽ
- Ngăn chặn SQL Injection thông qua stored procedures

- Ngăn chặn XSS và kiểm tra định dạng file khi upload

3.2 Hiệu suất

- Sử dụng session caching
- Tối ưu cơ sở dữ liệu (index, stored procedures)
- Nén JSON và tối ưu hóa hình ảnh

3.3 Xử lý lỗi

- Định dạng lỗi chuẩn với mã trạng thái HTTP đầy đủ
- Thông báo lỗi chi tiết
- Ghi log và theo dõi hoạt động hệ thống

Tổng kết

Tổng cộng, hệ thống sử dụng hơn 20 API nội bộ và 2 API bên ngoài. Mỗi API đều có tài liệu chi tiết về endpoint, tham số, phản hồi và stored procedure tương ứng. Cấu trúc API rõ ràng, đảm bảo hiệu quả trong phát triển, vận hành và bảo trì hệ thống.

CHƯƠNG 5: THIẾT KẾ CHƯƠNG TRÌNH

5.1. Cấu hình chương trình

5.1.1. File appsettings.json:

Đây là file cấu hình mặc định của ứng dụng ASP.NET Core, dùng để lưu trữ các thiết lập quan trọng dưới dạng JSON. Mục "Logging" định nghĩa mức độ log hiển thị của hệ thống và các thành phần framework, giúp theo dõi và gỡ lỗi. "AllowedHosts": "*" cho phép ứng dụng chấp nhận yêu cầu từ bất kỳ tên miền nào. Ngoài ra, phần "Agora" chứa thông tin cấu hình liên quan đến dịch vụ bên thứ ba (ở đây là nền tảng truyền thông Agora), bao gồm AppId và AppCertificate để xác thực ứng dụng. File này giúp tách biệt phần cứng cấu hình ra khỏi mã nguồn, dễ dàng thay đổi mà không cần biên dịch lại.

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",
  "Agora": {
    "AppId": "YOUR_APP_ID",
    "AppCertificate": "YOUR_APP_CERTIFICATE"
  }
}
```

5.1.2. File launchSettings.json:

File launchSettings.json quy định cách ứng dụng ASP.NET Core khởi chạy khi chạy debug trong môi trường phát triển. Cấu hình "iisSettings" mô tả thiết lập cho IIS Express, trong đó bật xác thực ẩn danh và cung cấp URL cũng như cổng SSL mặc

định. Mục "profiles" định nghĩa nhiều cấu hình khởi chạy khác nhau như "http", "https" và "IIS Express", mỗi cấu hình có thể chỉ định URL ứng dụng, bật trình duyệt tự động, và đặt biến môi trường như "ASPNETCORE_ENVIRONMENT": "Development". File này chỉ dùng trong quá trình phát triển, không ảnh hưởng khi publish lên host thật.

```
{  
  "$schema": "http://json.schemastore.org/launchsettings.json",  
  "iisSettings": {  
    "windowsAuthentication": false,  
    "anonymousAuthentication": true,  
    "iisExpress": {  
      "applicationUrl": "http://localhost:26613",  
      "sslPort": 44390  
    }  
  },  
  "profiles": {  
    "http": {  
      "commandName": "Project",  
      "dotnetRunMessages": true,  
      "launchBrowser": true,  
      "applicationUrl": "http://localhost:5068",  
      "environmentVariables": {  
        "ASPNETCORE_ENVIRONMENT": "Development"  
      }  
    },  
    "https": {  
      "commandName": "Project",  
      "dotnetRunMessages": true,  
      "launchBrowser": true,  
      "applicationUrl": "https://localhost:7123;http://localhost:5068",  
      "environmentVariables": {  
        "ASPNETCORE_ENVIRONMENT": "Development"  
      }  
    }  
  }  
}
```

```
"ASPNETCORE_ENVIRONMENT": "Development"  
}  
},  
"IIS Express": {  
    "commandName": "IISExpress",  
    "launchBrowser": true,  
    "environmentVariables": {  
        "ASPNETCORE_ENVIRONMENT": "Development"  
    }  
}  
}  
}
```

5.1.3. File Program.cs:

Đây là file cấu hình chính của ứng dụng ASP.NET Core theo mô hình Minimal Hosting Model (từ .NET 6 trở lên). Trong phần này, ứng dụng được cấu hình để hỗ trợ các dịch vụ như MVC (AddControllersWithViews), session (AddSession), SignalR (giao tiếp thời gian thực) và cache (AddDistributedMemoryCache). Pipeline xử lý yêu cầu HTTP được thiết lập bao gồm: chuyển hướng HTTPS, phục vụ file tĩnh, routing, kích hoạt session và phân quyền truy cập. Hệ thống định tuyến cho controller Admin và Home cũng được khai báo rõ ràng. Cuối cùng, ứng dụng được khởi chạy bằng app.Run() để bắt đầu lắng nghe các request từ người dùng.

```
var builder = WebApplication.CreateBuilder(args);  
  
// Thêm các dịch vụ cần thiết vào container.  
builder.Services.AddControllersWithViews();  
builder.Services.AddDistributedMemoryCache(); // Cài đặt bộ nhớ đệm phân tán  
builder.Services.AddSession(options =>  
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
options.IdleTimeout = TimeSpan.FromMinutes(30); // Thời gian hết hạn của session
options.Cookie.HttpOnly = true; // Cookie chỉ có thể truy cập từ HTTP
options.Cookie.IsEssential = true; // Đảm bảo cookie cần thiết cho ứng dụng });
builder.Services.AddHttpContextAccessor();

// Thêm SignalR vào container
builder.Services.AddSignalR();

var app = builder.Build();

// Cấu hình pipeline yêu cầu HTTP.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error"); // Xử lý lỗi khi ứng dụng không phải môi trường phát triển
    app.UseHsts(); // Sử dụng HTTP Strict Transport Security (HSTS)
}

app.UseHttpsRedirection(); // Chuyển hướng tất cả các yêu cầu HTTP sang HTTPS
app.UseStaticFiles(); // Cung cấp các tệp tĩnh (CSS, JavaScript, hình ảnh, v.v.)
app.UseRouting(); // Cấu hình routing cho ứng dụng

// Kích hoạt session middleware
app.UseSession();

app.UseAuthorization(); // Kích hoạt middleware kiểm tra quyền truy cập

// Định tuyến các yêu cầu tới các controller
app.MapControllerRoute(
```

THIẾT KẾ CHƯƠNG TRÌNH

```
name: "admin",
pattern: "Admin/{action=AdminLogin}/{id?}",
defaults: new { controller = "Admin" });

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

// Định tuyến SignalR Hub

app.Run(); // Khởi chạy ứng dụng
```

5.1.4. File DataModel.cs:

Lớp DataModel chịu trách nhiệm kết nối và truy vấn dữ liệu từ cơ sở dữ liệu SQL Server. Biến connectionStrings lưu chuỗi kết nối đến cơ sở dữ liệu trên dịch vụ somee.com. Phương thức get() nhận vào một câu lệnh SQL và thực hiện truy vấn, sau đó duyệt qua từng dòng kết quả và lưu dữ liệu vào ArrayList. Trước khi thêm vào danh sách, dữ liệu được xử lý qua hàm Xulydulieu() nhằm loại bỏ hoặc mã hóa các ký tự đặc biệt như dấu nháy, xuống dòng, hoặc dấu phẩy, giúp tránh lỗi hoặc lỗ hổng bảo mật khi hiển thị trên giao diện. File này đóng vai trò như một tầng truy cập dữ liệu thủ công (Data Access Layer).

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Collections;
using System.Data.SqlClient;
using Microsoft.Data.SqlClient;
```

THIẾT KẾ CHƯƠNG TRÌNH

```
namespace DoAnCNPM.Models
{
    public class DataModel
    {
        private string connectionStrings = "workstation
id=VOVBACSI.mssql.somee.com;packet
size=4096;user
id=LuongDat_SQLLogin_1;pwd=123456789;data
source=VOVBACSI.mssql.somee.com;persist security info=False;initial
catalog=VOVBACSI;TrustServerCertificate=True";
        public ArrayList get(String sql)
        {
            ArrayList datalist = new ArrayList();
            SqlConnection connection = new SqlConnection(connectionStrings);

            SqlCommand command = new SqlCommand(sql, connection);
            connection.Open();
            using (SqlDataReader r = command.ExecuteReader())
            {
                while (r.Read())
                {
                    ArrayList row = new ArrayList();
                    for (int i = 0; i < r.FieldCount; i++)
                    {
                        row.Add(Xulydulieu(r.GetValue(i).ToString()));
                    }
                    datalist.Add(row);
                }
            }
            connection.Close();
        }
    }
}
```

```
        return datalist;
    }

    public string Xulydulieu( string text){
        String s = text.Replace(", ", "&44;");
        s = s.Replace("'", "&34;");
        s = s.Replace("''", "&39;");
        s = s.Replace("\r", "");
        s = s.Replace("\n", "");
        return s;
    }

}
```

5.1.5. File ErrorViewModel.cs:

Lớp ErrorViewModel được sử dụng để truyền thông tin lỗi từ Controller đến View. Thuộc tính RequestId đại diện cho mã yêu cầu của lỗi, còn ShowRequestId giúp xác định có nên hiển thị mã lỗi này không (chỉ hiển thị khi khác null hoặc rỗng). Ngoài ra, thuộc tính Message chứa nội dung thông báo lỗi tùy chỉnh. Lớp này thường được dùng trong trang hiển thị lỗi (Error.cshtml) nhằm hỗ trợ việc kiểm tra và gỡ lỗi dễ dàng hơn khi ứng dụng xảy ra sự cố.

```
namespace DoAnCNPM.Models;

public class ErrorViewModel
{
    public string? RequestId { get; set; }

    public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);
    public string? Message { get; set; }
```

}

5.2. Chức năng chương trình

5.2.1. Chức năng[Xem thông báo]

Controller: Phương thức LayoutShare có vai trò trung tâm trong việc chuẩn bị dữ liệu nền cho giao diện chung (layout) của toàn bộ website. Khi được thực thi, nó trước hết kiểm tra phiên đăng nhập của người dùng qua Session. Sau đó, phương thức này tạo một đối tượng DataModel để thực hiện hai truy vấn chính đến cơ sở dữ liệu SQL Server: lấy danh sách tất cả khoa bệnh để hiển thị trên menu, và lấy danh sách thông báo cá nhân nếu người dùng đã đăng nhập. Dữ liệu này sau đó được lưu vào ViewBag và ViewData để các thành phần trên layout như header hay thanh điều hướng có thể sử dụng và hiển thị động.

```
public IActionResult LayoutShare()
{
    var taikhoan = _session.GetString("taikhoan");
    ViewData["TaiKhoan"] = taikhoan;

    DataModel db = new DataModel();
    ViewBag.listKB = db.get("EXEC getAllKhoaBenh");

    // PHẦN CODE CỦA CHỨC NĂNG XEM THÔNG BÁO
    if (!string.IsNullOrEmpty(taikhoan))
    {
        var MaND = taikhoan;
        ViewBag.ListTB = db.get($"EXEC sp_GetThongBaoByMaND @MaND =
        {MaND}");
    }
    return View(); // Phương thức này thực chất không trả về View,
                  // nó chỉ chuẩn bị dữ liệu. Lệnh return này sẽ không bao giờ được gọi.
}
```

View: Đoạn mã này tạo ra một hộp thoại thông báo (pop-up) có thể tái sử dụng trên toàn bộ trang web. Phần **HTML** định nghĩa một div được ẩn mặc định, được tạo kiểu bằng CSS để hiển thị như một cửa sổ pop-up nổi ở trung tâm màn hình. Phần **JavaScript** cung cấp hàm showThongBao để điều khiển hộp thoại này. Hàm nhận vào nội dung tin nhắn và trạng thái (thành công/thất bại) để hiển thị thông báo với màu sắc tương ứng (xanh lá/đỏ). Khi hàm được gọi, hộp thoại sẽ hiện ra và người dùng có thể đóng nó bằng cách nhấp vào nút "Đóng".

5.2.2. Chức năng[Lọc thông tin]

Controller :Action FillterDoctor chịu trách nhiệm **khởi tạo và hiển thị trang tìm kiếm ban đầu**. Nó truy vấn cơ sở dữ liệu để lấy danh sách *tất cả* bác sĩ (getAllBacSi) và danh sách tên bác sĩ (ListNameDoc) để dùng cho gợi ý tìm kiếm. Các danh sách này được đưa vào ViewBag và trả về View FillterDoctor.cshtml, hiển thị bộ lọc và kết quả ban đầu cho người dùng.

```
// Action để hiển thị trang có các bộ lọc
public IActionResult FillterDoctor()
{
    LayoutShare();
    DataModel db = new DataModel();
    // Lấy danh sách tên bác sĩ (có thể để làm gợi ý tìm kiếm)
    ViewBag.ListNameDoc = db.get("SELECT MaBS, nd.TenND FROM BACSI
bs, NGUOIDUNG nd where bs.MaND = nd.MaND");
    // Lấy danh sách tất cả bác sĩ ban đầu
    ViewBag.ListDoc = db.get("Exec getAllBacSi");

    return View();
}

// Action để xử lý và hiển thị kết quả sau khi lọc
```

```
public IActionResult FillterDoctorList(string khuvuc, string phikham, string
khoabenh, string hocham)
{
    LayoutShare();
    DataModel db = new DataModel();
    // Xử lý chuỗi đầu vào để tương thích với câu lệnh SQL
    if(khuvuc != "Null"){
        khuvuc = "N'" + khuvuc + "'";
    }
    if(hocham != "Null" ){
        hocham = "N'" + hocham + "'";
    }
    // Thực thi Stored Procedure lọc bác sĩ với các tham số
    ViewBag.ListDocFill = db.get($"EXEC FILTER_BACSI {khuvuc},
{phikham}, {khoabenh}, {hocham};");
}

return View();
}
```

View: **Về cấu trúc (HTML)**, trang bao gồm hai thành phần chính: một **form bộ lọc** cho phép người dùng chọn các tiêu chí (khu vực, phí khám,...) và gửi đi; và một **khung hiển thị danh sách bác sĩ**. Ban đầu, khung này sẽ hiển thị tất cả bác sĩ được truyền từ ViewBag.ListDoc. Mỗi bác sĩ được trình bày dưới dạng một "card" thông tin chi tiết, bao gồm hình ảnh, tên, chuyên khoa, phí khám và đánh giá sao.

5.2.3. Chức năng[Xem danh sách bệnh viện]

Controller : Action Index() là phương thức chính cho trang chủ của website. Khi người dùng truy cập trang chủ, phương thức này sẽ được gọi. Nó thực hiện các nhiệm vụ sau:

Gọi LayoutShare() để tải dữ liệu chung cho toàn trang.

Truy vấn cơ sở dữ liệu để lấy **danh sách bệnh viện** (getAllBenhVien) và **danh sách các bác sĩ nổi bật** (GetTop7Doctors).

Lưu các dữ liệu này vào ViewBag.

Cuối cùng, nó trả về một View(), là một trang HTML được hiển thị trực tiếp trên trình duyệt của người dùng.

KhachHangAPI.cs (Dành cho API)

Controller này không trả về giao diện đồ họa. Thay vào đó, nó có nhiệm vụ **cung cấp dữ liệu thô dưới định dạng JSON** cho các ứng dụng khác (ví dụ: ứng dụng di động, hoặc một website khác) sử dụng.

Phương thức Index() là một API endpoint đơn giản, nó truy vấn danh sách bệnh viện và trả về trực tiếp dưới dạng một mảng JSON.

Phương thức ListBenhVien() là một endpoint được thiết kế có cấu trúc hơn. Nó cũng lấy danh sách bệnh viện nhưng trả về một đối tượng JSON phức tạp hơn, thường bao gồm một cờ trạng thái (ví dụ: true để báo thành công) và dữ liệu đi kèm (listBV). Đây là một cách thiết kế API phổ biến để client có thể dễ dàng kiểm tra trạng thái của yêu cầu.

```
// File: HomeController.cs (Dùng cho giao diện Web)
public IActionResult Index()
{
    LayoutShare();
    DataModel db = new DataModel();
    // Lấy danh sách bệnh viện để hiển thị trên trang chủ
    ViewBag.ListBV = db.get("EXEC getAllBenhVien");
    ViewBag.ListBS5 = db.get("EXEC GetTop7Doctors");

    return View();
}

// =====

// File: KhachHangAPI.cs (Dùng cho API)
public JsonResult Index()
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
DataManager db =new DataManager();
ArrayList a = db.get("EXEC getAllBenhVien");
return Json(a);
}

public JsonResult ListBenhVien()
{
    try
    {
        // Lấy danh sách bệnh viện từ database
        var listBV = db.get("EXEC getAllBenhVien");
        // ... (lấy thêm danh sách bác sĩ)
        return Json(new object[]
        {
            true,
            listBV, // Dữ liệu danh sách bệnh viện
            // ...
        });
    }
    catch (Exception ex)
    {
        // ...
    }
}
```

5.2.4. Chức năng [Xem danh sách bác sĩ]

Trên giao diện Web (HomeController)

Chức năng xem danh sách bác sĩ được tích hợp ở hai vị trí chính trên website:

Trang chủ (Index action): Để thu hút người dùng, trang chủ chỉ hiển thị một danh sách chọn lọc 7 bác sĩ nổi bật nhất. Dữ liệu này được lấy thông qua stored procedure GetTop7Doctors và truyền tới View qua ViewBag.ListBS5.

THIẾT KẾ CHƯƠNG TRÌNH

Trang tìm kiếm bác sĩ (FillterDoctor action): Trang này cung cấp một danh sách đầy đủ và toàn diện tất cả các bác sĩ có trong hệ thống. Dữ liệu được lấy bằng stored procedure getAllBacSi và được sử dụng làm danh sách ban đầu trước khi người dùng áp dụng các bộ lọc.

Cung cấp qua API (KhachHangAPI)

Để phục vụ cho các ứng dụng bên ngoài (ví dụ: ứng dụng di động), một API endpoint đã được xây dựng.

Endpoint BacSi(): Action này cũng gọi đến stored procedure getAllBacSi để lấy toàn bộ danh sách bác sĩ. Tuy nhiên, thay vì trả về một trang HTML, nó trả về một JsonResult, tức là dữ liệu bác sĩ được cung cấp dưới dạng dữ liệu thô JSON, giúp các ứng-dụng khác có thể dễ dàng đọc và xử lý.

```
// File: HomeController.cs (Dùng cho giao diện Web)
public IActionResult Index()
{
    LayoutShare();
    DataModel db = new DataModel();
    // Lấy danh sách 7 bác sĩ hàng đầu để hiển thị trên trang chủ
    ViewBag.ListBS5 = db.get("EXEC GetTop7Doctors");

    return View();
}

public IActionResult FillterDoctor()
{
    LayoutShare();
    DataModel db = new DataModel();
    // Lấy toàn bộ danh sách bác sĩ cho trang lọc
    ViewBag.ListDoc = db.get("Exec getAllBacSi");
```

```
return View();  
}  
  
// ======  
  
// File: KhachHangAPI.cs (Dùng cho API)  
public JsonResult BacSi()  
{  
    ArrayList a = db.get("EXEC getAllBacSi");  
    return Json(a);  
}
```

5.2.5. Chức năng[Xem chi tiết bác sĩ]

Action **DetailDoctor** có nhiệm vụ xây dựng trang **hồ sơ chi tiết của một bác sĩ** để hiển thị cho người dùng trên website. Khi người dùng chọn xem một bác sĩ, phương thức này được gọi với tham số là mã bác sĩ (MaBS). Nó sẽ thực hiện hai lần truy vấn cơ sở dữ liệu:

Lấy thông tin cá nhân, chuyên môn của bác sĩ qua stored procedure **DetDETAILBACSI**.

Lấy tất cả các bình luận, đánh giá liên quan đến bác sĩ đó qua stored procedure **GetCommentBACSI**.

Cả hai khôi dữ liệu này được đưa vào ViewBag và trả về một View() hoàn chỉnh để hiển thị đầy đủ trên trình duyệt. Cung cấp qua API

Action **BookExamine** trong API controller phục vụ một mục đích khác: **cung cấp dữ liệu bác sĩ dưới dạng JSON** cho các ứng dụng khác, ví dụ như khi người dùng chuẩn bị đặt lịch khám trên ứng dụng di động. Phương thức này cũng nhận vào MaBS nhưng chỉ truy vấn thông tin chi tiết cần thiết của bác sĩ (**DetDETAILBACSI1**). Thay vì trả về một trang web, nó trả về một **JsonResult** với cấu trúc chuẩn: một cờ báo thành công (**true**), một thông điệp, và dữ liệu chi tiết của bác sĩ. Điều này giúp các ứng dụng khác dễ dàng nhận và xử lý thông tin.

THIẾT KẾ CHƯƠNG TRÌNH

```
// File: HomeController.cs (Dùng cho giao diện Web)
public IActionResult DetailDoctor(string MaBS)
{
    LayoutShare();
    DataModel db = new DataModel();

    // Lấy thông tin chi tiết của bác sĩ
    ViewBag.ListDBS = db.get($"EXEC DetDETAILBACSI {MaBS}");

    // Lấy cả bình luận/đánh giá của bác sĩ này
    ViewBag.ListComment = db.get($"EXEC GetCommentBACSI {MaBS}");
    return View();
}

// =====

// File: KhachHangAPI.cs (Dùng cho API, trong chức năng chuẩn bị đặt lịch)
[HttpGet]
public JsonResult BookExamine(string MaBS)
{
    try
    {
        // ... (lấy thông tin người dùng)

        // Lấy thông tin chi tiết bác sĩ
        var doctorDetails = db.get($"EXEC DetDETAILBACSI1 {MaBS}");

        return Json(new object[]
        {
            true,
```

```
"Data retrieved successfully",
// ...,
doctorDetails // Dữ liệu chi tiết bác sĩ
});
}

catch (Exception ex)
{
    // ...
}
```

5.2.6. Chức năng [Xem đánh giá bác sĩ]

Chức năng này được thực hiện trong `HomeController` để lấy và hiển thị các đánh giá của bệnh nhân về một bác sĩ cụ thể trên trang chi tiết của họ.

Khi người dùng xem trang chi tiết của một bác sĩ (được xác định bằng `MaBS`), dòng code `ViewBag.ListComment = db.get($"EXEC GetCommentBACSI {MaBS}")`; sẽ được thực thi. Dòng này gọi phương thức `get` của lớp `DataModel` để thực thi một stored procedure trong SQL Server có tên là `GetCommentBACSI`. Nó truyền mã bác sĩ (`MaBS`) vào làm tham số để truy vấn và lấy ra tất cả các bình luận, đánh giá liên quan đến duy nhất bác sĩ đó. Kết quả (danh sách các bình luận) sau đó được gán vào `ViewBag.ListComment` để View có thể truy cập và hiển thị lên giao diện web.

```
// File: HomeController.cs

public IActionResult DetailDoctor(string MaBS)
{
    LayoutShare();
    DataModel db = new DataModel();

    // Lấy thông tin chi tiết của bác sĩ
    ViewBag.ListDBS = db.get($"EXEC DetDETAILBACSI {MaBS}");
```

```
// Dòng code này thực hiện chức năng xem đánh giá  
ViewBag.ListComment = db.get($"EXEC GetCommentBACSI {MaBS}");  
return View();  
}
```

5.2.7. Chức năng[Đăng nhập]

Action **LoginProcess** được đánh dấu với [HttpPost], nghĩa là nó chỉ được kích hoạt khi người dùng gửi thông tin từ form đăng nhập.

Phương thức nhận vào hai tham số là số điện thoại (sdt) và mật khẩu (password) do người dùng nhập.

Nó thực thi stored procedure **CheckLogin** trong cơ sở dữ liệu để kiểm tra xem thông tin đăng nhập có hợp lệ hay không.

Nếu xác thực thành công (nghĩa là CheckLogin trả về kết quả), thông tin định danh của người dùng sẽ được lưu vào Session với key là "taikhoan". Việc lưu vào Session giúp hệ thống ghi nhớ trạng thái đăng nhập của người dùng. Sau đó, người dùng được chuyển hướng đến trang chủ.

Nếu xác thực thất bại, người dùng sẽ được chuyển hướng trở lại trang đăng nhập để thử lại.

```
[HttpPost]  
public IActionResult LoginProcess(string sdt, string password)  
{  
    DataModel db = new DataModel();  
    var list = db.get("EXEC CheckLogin '" + sdt + "','" + password + "'");  
  
    if (list.Count > 0 && list[0] is ArrayList arrayList && arrayList.Count > 1)  
    {  
        var userInfo = arrayList[0].ToString() ?? "Unknown";  
        _session.SetString("taikhoan", userInfo);  
        return RedirectToAction("Index", "Home");  
    }  
}
```

```
    }
    else
    {
        return RedirectToAction("Login", "Home");
    }
}
```

5.2.8. Chức năng[Xem lịch sử khám bệnh]

Action này có nhiệm vụ hiển thị trang lịch sử khám bệnh của người dùng.

Đầu tiên, nó lấy thông tin người dùng đang đăng nhập từ `Session`. Sau đó, nó sử dụng mã người dùng này để thực thi stored procedure `GetLichKhamInfoByMaND` nhằm truy vấn tất cả các lịch khám (bao gồm đã khám, sắp khám, và đã hủy) của họ từ cơ sở dữ liệu. Danh sách các lịch khám này được gán vào `ViewBag.ListLichKham` và được truyền tới View để hiển thị chi tiết cho người dùng.

Hủy Lịch khám

Action này được gọi khi người dùng quyết định hủy một cuộc hẹn đã đặt.

Nó nhận vào mã của cuộc hẹn cần hủy (`maCHK`) làm tham số. Bên trong, nó thực thi stored procedure `DeleteCHKbyID` để xóa hoặc cập nhật trạng thái "đã hủy" cho cuộc hẹn tương ứng trong cơ sở dữ liệu. Sau khi thực hiện xong, nó sẽ tự động **chuyển hướng** người dùng trở lại trang lịch sử khám (`ExamineHistory`) để họ có thể thấy danh sách lịch hẹn đã được cập nhật.

```
// File: HomeController.cs

// Hiển thị trang lịch sử khám
public IActionResult ExamineHistory()
{
    LayoutShare();

    var taikhoan = HttpContext.Session.GetString("taikhoan");
```

```
var MaND = taikhoan;

DataModel db = new DataModel();
// Lấy thông tin các lịch khám của người dùng
ViewBag.ListLichKham = db.get($"EXEC GetLichKhamInfoByMaND
{MaND}");

return View();
}

// Xử lý hủy một lịch khám
public IActionResult Cancel(string maCHK)
{
    DataModel db = new DataModel();
    // Thực thi lệnh xóa cuộc hẹn khám
    ViewBag.List = db.get("EXEC DeleteCHKbyID " + maCHK);

    return RedirectToAction("ExamineHistory", "Home");
}
```

5.2.9. Chức năng[Cập nhật thông tin cá nhân]

- Controller: Controller này xử lý logic cho trang cá nhân của người dùng. Phương thức PersonalPage() truy xuất thông tin tài khoản từ session, sau đó sử dụng DataModel để lấy dữ liệu người dùng từ cơ sở dữ liệu dựa trên manD. Dữ liệu này (nếu có) được lưu vào ViewBag để hiển thị trên View. Phương thức UpdateUserInfo() (dùng [HttpPost]) nhận dữ liệu người dùng từ form, bao gồm cả ảnh đại diện. Nó xử lý lưu trữ ảnh vào thư mục wwwroot/Uploads, định dạng lại ngày sinh, và cuối cùng, gọi một stored procedure UpdateUserInfo thông qua DataModel để cập nhật thông tin vào cơ sở dữ liệu trước khi chuyển hướng người dùng trở lại trang cá nhân.

```
public IActionResult PersonalPage()
{
    LayoutShare();
    DataModel db = new DataModel();
    var taikhoan = HttpContext.Session.GetString("taikhoan");
    ViewData["TaiKhoan"] = taikhoan;

    if (!string.IsNullOrEmpty(taikhoan))
    {
        var result = db.get("SELECT * from NGUOIDUNG where manD=" +
taikhoan);
        if (result != null && result.Count > 0)
        {
            ViewBag.UserInfo = result[0]; // Lấy dòng đầu tiên của kết quả
        }
    }
    return View();
}

[HttpPost]
public IActionResult UpdateUserInfo(string MaND, string TenND, string Email,
                                    string NamSinh, string GioiTinh,
                                    string DiaChi, IFormFile Hinhcanhan)
{
    DataModel db = new DataModel();
    int manD = int.Parse(MaND);
    DateTime parsedDate = DateTime.Parse(NamSinh);
    string formattedDate = parsedDate.ToString("yyyy-MM-dd");
```

```
string nameFile = "NULL";
if(Hinhcanhan != null){
    // lấy tên tệp
    nameFile = Path.GetFileName(Hinhcanhan.FileName);

    // Đường dẫn để lưu tệp
    string uploadsFolder = Path.Combine(Directory.GetCurrentDirectory(),
"wwwroot", "Uploads");
    Directory.CreateDirectory(uploadsFolder); // Tạo thư mục nếu chưa tồn tại
    string filePath = Path.Combine(uploadsFolder, nameFile);
    // Lưu tệp
    using (var stream = new FileStream(filePath, FileMode.Create))
    {
        Hinhcanhan.CopyTo(stream);
    }
}

db.get($"EXEC UpdateUserInfo {manD}, N'{TenND}', '{Email}',
'formattedDate', N'{GioiTinh}', N'{DiaChi}', '{nameFile}' ");

return RedirectToAction("PersonalPage", "Home");
}
```

- View: View này thiết kế giao diện cho trang cá nhân, bao gồm việc hiển thị thông tin người dùng và form chỉnh sửa. Nó sử dụng Bootstrap (row, col-md-4, col-md-8, card, form-control) để tạo bố cục và phong cách responsive. Dữ liệu người dùng được truyền từ Controller thông qua ViewBag.UserInfo và được hiển thị động trong các trường form. Đặc biệt, nó hiển thị ảnh đại diện, tên, mã giới thiệu, và số dư. Form UpdateUserInfo cho phép người dùng chỉnh sửa thông tin cá nhân và tải lên ảnh mới, sau đó gửi dữ liệu trở lại Controller để xử lý cập nhật.

THIẾT KẾ CHƯƠNG TRÌNH

```
@using System.Collections

@{
    ViewData["Title"] = "Trang cá nhân";
    Layout = "~/Views/Shared/_Layout.cshtml";

}

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

<div class="slide-img" >
    <div style="overflow: hidden;">
        <div style="overflow: hidden;">
            <svg
                preserveAspectRatio="none"
                viewBox="0 0 1200 120"
                xmlns="http://www.w3.org/2000/svg"
                style="fill: #2ea8ff; width: 139%; height: 265px;">
                >
                <path
                    d="M0 0v46.29c47.79 22.2 103.59 32.17 158 28 70.36-5.37 136.33-33.31
206.8-37.5 73.84-4.36 147.54 16.88 218.2 35.26 69.27 18 138.3 24.88 209.4 13.08
36.15-6 69.85-17.84 104.45-29.34C989.49 25 1113-14.29 1200 52.47V0z"
                    opacity=".25"
                />
                <path
                    d="M0 0v15.81c13 21.11 27.64 41.05 47.69 56.24C99.41 111.27 165 111
224.58 91.58c31.15-10.15 60.09-26.07 89.67-39.8 40.92-19 84.73-46 130.83-
49.67 36.26-2.85 70.9 9.42 98.6 31.56 31.77 25.39 62.32 62 103.63 73 40.44 10.79
81.35-6.69 119.13-24.28s75.16-39 116.92-43.05c59.73-5.85 113.28 22.88 168.9
38.84 30.2 8.66 59 6.17 87.09-7.5 22.43-10.89 48-26.93 60.65-49.24V0z"
                />
            </svg>
        </div>
    </div>
</div>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    opacity=".5"
/>
<path d="M0 0v5.63C149.93 59 314.09 71.32 475.83 42.57c43-7.64 84.23-
20.12 127.61-26.46 59-8.63 112.48 12.24 165.56 35.4C827.93 77.22 886 95.24
951.2 90c86.53-7 172.46-45.71 248.8-84.81V0z" />
</svg>
</div>
</div>
</div>
<div class="container-wrapper anim">
<div class="container form-personal ">
<div class="row g-4">
<div class="col-md-4">
<div class="card h-100" style="padding: 0;">
<div class="card-body text-center" style="padding: 0;">
@if (ViewBag.UserInfo != null)
{
    var user = ViewBag.UserInfo as ArrayList;
    if (user != null && user.Count > 1)
    {
        <div style="padding: 25px;">
            
            <h2      class="card-subtitle"      mb-2      text-
muted">@user[1]</h2>
            <p class="card-text" style="font-weight: bold; font-size:
20px">Mã giới thiệu: @user[10]</p>
            <button class="btn-changepass">Đổi mật khẩu</button>
        </div>
        <h3 style="color: white;">
    
```

```
@{
    // Định dạng số tiền đẹp hơn
    var amount = Convert.ToInt32(user[9]);
    var SoTien = amount.ToString("#,##0");
}

Số dư:
<span style="color: #fff003;">
    @SoTien đ
</span>
</h3>
}

}
</div>
</div>
</div>

<div class="col-md-8">
    <div class="card h-100 " style="min-width: 100%">
        <div class="card-body ">
            <h2 class="card-title">Thông tin cá nhân</h2>
            <form      asp-controller="Home"      asp-action="UpdateUserInfo"
method="post" enctype="multipart/form-data">
                @if (ViewBag.UserInfo != null)
                {
                    var user = ViewBag.UserInfo as ArrayList;
                    if (user != null && user.Count > 1)
                    {
                        <input type="text" class="form-control" id="fullName"
value="@user[0]" name="MaND" style="display: none;">
                        <div class="mb-3">

```

THIẾT KẾ CHƯƠNG TRÌNH

```
<label for="fullName" class="form-label">Họ và  
tên:</label>  
  
<input type="text" class="form-control"  
value="@user[1]" name="TenND">  
  
</div>  
  
<div class="mb-3">  
  
    <label for="email" class="form-label">Email:</label>  
  
    <input type="text" class="form-control" id="email"  
value="@user[3]" name="Email">  
  
</div>  
  
    @if (user[6] != null &&  
DateTime.TryParse(user[6].ToString(), out DateTime birthDate))  
  
    {  
  
        var formattedDate = birthDate.ToString("yyyy-MM-  
dd"); // Định dạng yyyy-MM-dd  
  
        <div class="mb-3">  
  
            <label for="birthYear" class="form-label">Năm  
sinh:</label>  
  
            <input type="date" class="form-control"  
id="birthYear" name="NamSinh" value="@formattedDate" />  
  
</div>  
  
    }  
    else  
    {  
  
        <div class="mb-3">  
  
            <label for="birthYear" class="form-label">Năm  
sinh:</label>  
  
            <input type="date" class="form-control"  
id="birthYear" name="NamSinh" />  
    }
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</div>
}

<div class="mb-3">
    <label for="gender" class="form-label">Giới
tính:</label>
    <select class="form-select" id="gender"
name="GioiTinh">
        <option selected>@user[7]</option>
        <option value="Nam">Nam</option>
        <option value="Nữ">Nữ</option>
        <option value="Khác">Khác</option>
    </select>
</div>
<div class="mb-3">
    <label for="address" class="form-label">Địa
chi:</label>
    <input type="text" class="form-control" id="address"
value="@user[5]" name="DiaChi">
</div>
<div class="mb-3">
    <label for="phone" class="form-label">Số     điện
thoại:</label>
    <input type="tel" class="form-control" id="phone"
value="@user[4]" name="SDT">
</div>

<div class="mb-3">
    <label for="avar" class="form-label">Ảnh     cá
nhân:</label>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<input type="file" class="form-control" id="avar"
name="Hinhcanhan">
</div>

<button type="submit" class="btn btn-primary">Cập
nhật</button>
}

}

</form>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</script>

function formatDate(dateStr) {
    const date = new Date(dateStr);
    const day = String(date.getDate()).padStart(2, '0');
    const month = String(date.getMonth() + 1).padStart(2, '0'); // Tháng bắt đầu
từ 0
    const year = date.getFullYear();
    return `${day}/${month}/${year}`;
}

document.getElementById('birthYear').value = formatDate(rawDate);
</script>
```

5.2.10. Chức năng [Quản lý giao dịch]

- Controller: Controller này chịu trách nhiệm hiển thị trang quản lý tài khoản và số dư. Phương thức AccountBank() đầu tiên gọi LayoutShare() để thiết lập layout chung, sau đó lấy thông tin tài khoản người dùng từ Session. Dựa vào mã người dùng, nó truy vấn cơ sở dữ liệu bằng DataModel để lấy chi tiết thông tin người dùng. Nếu tìm thấy thông tin, dòng dữ liệu đầu tiên sẽ được lưu vào ViewBag.UserInfo để hiển thị trên View, giúp người dùng xem số dư hiện tại của họ. Hiện tại, chưa có phương thức AccountBankProcess được cung cấp để xử lý logic nạp tiền.

```
public IActionResult AccountBank()
{
    LayoutShare();
    DataModel db = new DataModel();
    var taikhoan = HttpContext.Session.GetString("taikhoan");
    ViewData["TaiKhoan"] = taikhoan;

    if (!string.IsNullOrEmpty(taikhoan))
    {
        var result = db.get("SELECT * from NGUOIDUNG where manD=" +
taikhoan);
        if (result != null && result.Count > 0)
        {
            ViewBag.UserInfo = result[0]; // Lấy dòng đầu tiên của kết quả
        }
    }
    return View();
}
```

- View: View này được thiết kế để người dùng kiểm tra số dư hiện tại và thực hiện các giao dịch nạp tiền. Giao diện có breadcrumb giúp điều hướng, hiển thị số dư tài

THIẾT KẾ CHƯƠNG TRÌNH

khoản của người dùng một cách rõ ràng. Phần nạp tiền bao gồm một trường nhập số tiền, các nút gợi ý nhanh các mức nạp phổ biến và nhiều tùy chọn phương thức thanh toán (Mobile banking, Thẻ ATM nội địa, Thẻ tín dụng/Ghi nợ) thông qua cổng VNPAY, được trình bày một cách trực quan. Người dùng cũng có thể nhập mã khuyến mại. JavaScript được sử dụng để làm nổi bật phương thức thanh toán được chọn và tự động điền số tiền khi nhấn nút gợi ý.

```
@{  
    ViewData["Title"] = "Tài khoản";  
    Layout = "~/Views/Shared/_Layout.cshtml";  
}  
  
<link rel="stylesheet" href("~/css/Article.css" asp-append-version="true">  
  
<div class="container center">  
    <div class="bread-flow" style="margin-top: 10px;">  
        <nav aria-label="breadcrumb">  
            <ol class="breadcrumb">  
                <li class="breadcrumb-item" asp-controller="Home" asp-action="Index">Trang chủ</a></li>  
                <li class="breadcrumb-item active" aria-current="page">Số dư & nạp  
                    tiền</li>  
            </ol>  
        </nav>  
    </div>  
    <div class="sodu">  
        <h3 class="">  
            @{  
                // Định dạng số tiền đẹp hơn  
                var amount1 = Convert.ToInt32(ViewBag.UserInfo[9]);  
            }  
        </h3>  
    </div>  
</div>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
var SoTienUser = amount1.ToString("#,##0");

}

Số dư:  

<span style="color: #fff003;">  

    @SoTienUser đ  

</span>  

</h3>  

</div>  

<div class="account">  

    <div class="title-content_accou">  

        <h2>  

            NẠP TIỀN  

        </h2>  

    </div>  

    <form method="post" action("~/Home/AccountBankProcess"  

        enctype="multipart/form-data">  

        <input type="text" name="SoDuTK" id="amountInput" class="form-control  

            mb-3" placeholder="Số tiền cần nạp (VND)">  

        <div class="btn-group mb-4" role="group">  

            <button type="button" class="btn btn-outline-secondary"  

                onclick="setAmount(200000)">200,000 VND</button>  

            <button type="button" class="btn btn-outline-secondary"  

                onclick="setAmount(500000)">500,000 VND</button>  

            <button type="button" class="btn btn-outline-secondary"  

                onclick="setAmount(1000000)">1,000,000 VND</button>  

        </div>  

        <h2>Phương thức thanh toán</h2>  

        <div class="payment-option selected">
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<input type="radio" name="paymentMethod" id="mobileBanking" checked>
    
    <label for="mobileBanking">
        <strong>Mobile banking</strong><br>
        Cỗng thanh toán VNPay
    </label>
</div>

<div class="payment-option">
    <input type="radio" name="paymentMethod" id="atmCard">
    
    <label for="atmCard">
        <strong>Thẻ ATM nội địa</strong><br>
        Cỗng thanh toán VNPay
    </label>
</div>

<div class="payment-option">
    <input type="radio" name="paymentMethod" id="creditCard">
    
    <label for="creditCard">
        <strong>Thẻ tín dụng/Ghi nợ</strong><br>
        Cỗng thanh toán VNPay
    </label>
</div>
<h2 class="fm-text">Mã khuyến mại</h2>
```

```
<input type="text" class="form-control mb-3" placeholder="Hệ thống ưu tiên  
áp dụng mã khuyến mại nếu có">  
  
<input type="submit" class="btn btn-primary btn-lg w-100">  
</form>  
</div>  
  
<script>  
    document.querySelectorAll('.payment-option').forEach(option => {  
        option.addEventListener('click', function() {  
            document.querySelectorAll('.payment-option').forEach(opt =>  
                opt.classList.remove('selected'));  
            this.classList.add('selected');  
            this.querySelector('input[type="radio"]').checked = true;  
        });  
    });  
    function setAmount(amount) {  
        document.getElementById("amountInput").value = amount;  
    }  
</script>
```

5.2.11. Chức năng [Nạp tiền vào tài khoản]

Controller: Controller này xử lý logic khi người dùng gửi yêu cầu nạp tiền. Phương thức AccountBankProcess() (sử dụng [HttpPost]) nhận vào số tiền cần nạp (SoDuTK) từ form. Nó lấy mã người dùng (MaND) từ Session hiện tại, sau đó sử dụng DataModel để thực thi stored procedure NapTien trong cơ sở dữ liệu, truyền vào mã người dùng và số tiền nạp. Sau khi thực hiện xong thao tác nạp tiền, Controller sẽ chuyển hướng người dùng trở lại trang AccountBank để họ có thể xem số dư cập nhật của mình.

[HttpPost]

```
public IActionResult AccountBankProcess(string SoDuTK)
{
    var taikhoan = HttpContext.Session.GetString("taikhoan");
    ViewData["TaiKhoan"] = taikhoan;
    var MaND = taikhoan;

    DataModel db = new DataModel();
    db.get($"EXEC NapTien {MaND}, {SoDuTK}");

    return RedirectToAction("AccountBank", "Home");
}
```

5.2.12. Chức năng [Xem bài viết]

Controller: Controller này quản lý việc hiển thị các bài viết và danh sách bài viết. Có ba phương thức IActionResult: Article(string MaBV): Lấy thông tin chi tiết của một bài viết cụ thể dựa trên MaBV (Mã Bài Viết) và lưu vào ViewBag.ListBV. ListArticleLoaiBV(string MaLBV): Lấy danh sách các bài viết thuộc một loại bài viết cụ thể dựa trên MaLBV (Mã Loại Bài Viết) và lưu vào ViewBag.ListLBV. ArticleLoaiBV(string MaBV): Tương tự Article, cũng lấy thông tin chi tiết bài viết theo MaBV và lưu vào ViewBag.ListBV. Cả ba phương thức đều sử dụng DataModel để truy vấn cơ sở dữ liệu thông qua các stored procedure tương ứng và gọi LayoutShare() để đảm bảo layout chung.

```
public IActionResult Article(string MaBV)
{
    LayoutShare();

    DataModel db = new DataModel();
```

```
// Sử dụng tham số hóa để tránh lỗi
ViewBag.ListBV = db.get($"EXEC getBaiVietByIDMaKhoaBenh
{MaBV}");  
  
return View();
}  
  
public IActionResult ListArticleLoaiBV(string MaLBV)
{
    LayoutShare();  
  
DataModel db = new DataModel();  
  
// Sử dụng tham số hóa để tránh lỗi
ViewBag.ListLBV = db.get($"EXEC getBaiVietByIDMaLoai {MaLBV}");  
  
return View();
}  
  
public IActionResult ArticleLoaiBV(string MaBV)
{
    LayoutShare();  
  
DataModel db = new DataModel();  
  
// Sử dụng tham số hóa để tránh lỗi
ViewBag.ListBV = db.get($"EXEC getBaiVietByID {MaBV}");  
  
return View();
}
```

View: View này được thiết kế để hiển thị nội dung chi tiết của một bài viết. Nó bao gồm một phần tiêu đề và hình ảnh cover, cùng với breadcrumb để người dùng dễ dàng điều hướng. Nội dung chính của bài viết được hiển thị, bao gồm tiêu đề bài viết và nội dung chi tiết (HTML) được lấy từ ViewBag.ListBV. Đặc biệt, View này còn hỗ trợ nhúng video YouTube bằng cách chuyển đổi URL video thành định dạng embed để hiển thị trong iframe, mang lại trải nghiệm đa phương tiện cho người đọc.

```
@{  
    ViewData["Title"] = "Thông tin";  
    Layout = "~/Views/Shared/_Layout.cshtml";  
  
}  
  
<link rel="stylesheet" href("~/css/Article.css" asp-append-version="true">  
  
<div class="cover_list_news">  
    <div class="name_cate_cover animcc">Bài viết</div>  
    <img src("~/images/bg_article.jpg" alt="Cấp cứu">  
</div>  
  
<div class="container">  
  
<div class="bread-flow">  
    <nav aria-label="breadcrumb">  
        <ol class="breadcrumb">  
            <li class="breadcrumb-item"><a asp-controller="Home" asp-action="Index">Trang chủ</a></li>  
            <li class="breadcrumb-item active" aria-current="page">@ViewBag.ListBV[0][1]</li>  
        </ol>  
    </nav>  
</div>
```

```
<div class="content title-content_arti">
    <h2>@ViewBag.ListBV[0][1]</h2>
    @{
        var url_raw = ViewBag.ListBV[0][4];
        var url_embed = url_raw.Contains("watch?v=") ?
            url_raw.Replace("watch?v=", "embed/") : url_raw;
    }
    <div class="center">
        <iframe width="400px" height="210px"
            src="@url_embed"
            title="YouTube video"
            frameborder="0"
            allow="accelerometer; autoplay; clipboard-write; encrypted-media;
gyroscope; picture-in-picture"
            allowfullscreen>
        </iframe>
    </div>
    <div class="ck ck-editor__main" role="presentation">
        <div class="article-content">
            @Html.Raw(ViewBag.ListBV[0][2])
        </div>
    </div>

    </div>
</div>
@{
    ViewData["Title"] = "@ViewBag.ListBV[0][9]";
    Layout = "~/Views/Shared/_Layout.cshtml";
}
```

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

<div class="cover_list_news">
    <div class="name_cate_cover animcc">@ViewBag.ListBV[0][9]</div>
    
</div>

<div class="container">

    <div class="bread-flow">
        <nav aria-label="breadcrumb">
            <ol class="breadcrumb">
                <li class="breadcrumb-item"><a asp-controller="Home" asp-action="Index">Trang chủ</a></li>
                <li class="breadcrumb-item"><a href="~/Home>ListArticleLoaiBV?MaLBV=@ViewBag.ListBV[0][6]">@ViewBag.ListBV[0][9]</a></li>
                <li class="breadcrumb-item active" aria-current="page">@ViewBag.ListBV[0][1]</li>
            </ol>
        </nav>
    </div>

    <div class="content title-content_arti">
        <h2>@ViewBag.ListBV[0][1]</h2>
        @{
            var url_raw = ViewBag.ListBV[0][4];
            var url_embed = url_raw.Contains("watch?v=") ? url_raw.Replace("watch?v=", "embed/") : url_raw;
        }
    </div>

```

```
}

<div class="center">

    <iframe width="400px" height="210px"
        src="@url_embed"
        title="YouTube video"
        frameborder="0"
        allow="accelerometer; autoplay; clipboard-write; encrypted-media;
gyroscope; picture-in-picture"
        allowfullscreen>

    </iframe>
</div>

<div class="ck ck-editor__main" role="presentation" style="margin: 20px
0;">

    <div class="article-content">
        @Html.Raw(ViewBag.ListBV[0][2])
    </div>
</div>

</div>
```

5.2.13. Chức năng [Đăng ký khám bệnh]

Controller: Controller này quản lý quy trình đặt lịch khám cho người dùng. BookExamine(string MaBS): Phương thức này hiển thị trang đặt lịch khám. Nó thực hiện ba tác vụ chính: thiết lập layout chung thông qua LayoutShare(), lấy thông tin tài khoản của người dùng từ Session và truy vấn cơ sở dữ liệu để hiển thị số dư, và cuối cùng là lấy thông tin chi tiết của bác sĩ được chọn (MaBS) để hiển thị trên form đặt lịch. BookExamineProcess(string MaBS, string MoTaBenh, List<IFormFile> HinhanhBenhs): Đây là phương thức xử lý khi người dùng gửi yêu cầu đặt lịch. Nó lấy mã người dùng từ Session, sau đó thực hiện các bước sau: Lưu hồ sơ bệnh án:

THIẾT KẾ CHƯƠNG TRÌNH

Gọi stored procedure SAVEHOSO để lưu mô tả bệnh và lấy lại MaHS (Mã Hồ Sơ) vừa tạo. Lưu hình ảnh bệnh: Duyệt qua danh sách các hình ảnh được tải lên, lưu từng tệp vào thư mục wwwroot/Uploads, và sau đó gọi stored procedure SAVEHINHANHBENH để lưu thông tin đường dẫn hình ảnh vào cơ sở dữ liệu, liên kết với MaHS đã tạo. Lưu cuộc hẹn: Cuối cùng, gọi stored procedure SAVECUOCHEKHAM để lưu thông tin cuộc hẹn vào cơ sở dữ liệu, bao gồm mã người dùng, mã bác sĩ, và mã hồ sơ bệnh án. Sau khi hoàn tất, người dùng sẽ được chuyển hướng về trang chủ

```
public IActionResult BookExamine(string MaBS)
{
    LayoutShare();

    DataModel db = new DataModel();
    var taikhoan = HttpContext.Session.GetString("taikhoan");
    ViewData["TaiKhoan"] = taikhoan;

    if (!string.IsNullOrEmpty(taikhoan))
    {
        var result = db.get("SELECT * from NGUOIDUNG where manD=" +
taikhoan);
        if (result != null && result.Count > 0)
        {
            ViewBag.UserInfo = result[0]; // Lấy dòng đầu tiên của kết quả
        }
    }

    ViewBag.ListDBS = db.get($"EXEC DetDETAILBACSI1 {MaBS}");

    return View();
}
```

```
[HttpPost]
```

```
public IActionResult BookExamineProcess(string MaBS, string MoTaBenh,  
List<IFormFile> HinhAnhBenhs)  
{  
    DataModel db = new DataModel();  
  
    var taikhoan = HttpContext.Session.GetString("taikhoan");  
    ViewData["TaiKhoan"] = taikhoan;  
    var MaND = taikhoan;  
  
    var result = db.get($"DECLARE @MaHS INT; EXEC SAVEHOSO  
{MaND}, N'{MoTaBenh}', @MaHS = @MaHS OUTPUT; SELECT @MaHS;");  
    if (result != null && result.Count > 0)  
    {  
        ViewBag.MaHS = result[0]; // Lấy dòng đầu tiên của kết quả  
    }  
    var MaHS = int.Parse(ViewBag.MaHS[0].ToString());  
  
    foreach (var file in HinhAnhBenhs)  
    {  
        // lấy tên tệp  
        string nameFile = Path.GetFileName(file.FileName);  
  
        // Đường dẫn để lưu tệp  
        string uploadsFolder = Path.Combine(Directory.GetCurrentDirectory(),  
"wwwroot", "Uploads");  
        Directory.CreateDirectory(uploadsFolder); // Tạo thư mục nếu chưa tồn tại  
        string filePath = Path.Combine(uploadsFolder, nameFile);  
        // Lưu tệp  
        using (var stream = new FileStream(filePath, FileMode.Create))
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    file.CopyTo(stream);  
}  
  
db.get($"EXEC SAVEHINHANHBENH {MaHS}, '{nameFile}");  
}  
  
db.get($"EXEC SAVECUOCHEKHAM {MaND}, {MaBS}, {MaHS},  
null");  
  
return RedirectToAction("Index", "Home");  
}
```

View: View này cung cấp giao diện để người dùng đặt lịch hẹn khám với bác sĩ. Thông tin hiển thị: Trang hiển thị thông tin số dư hiện tại của người dùng, thông tin chi tiết của bác sĩ (tên, bệnh viện, chuyên khoa) và form điền thông tin khách hàng (họ tên, giới tính, số điện thoại, email, ngày sinh). Mô tả bệnh và hình ảnh: Người dùng có thể nhập lý do khám và tải lên nhiều hình ảnh liên quan đến bệnh. Phí khám: Hiển thị rõ ràng phí khám của bác sĩ. Xác nhận và Điều khoản: Có một checkbox để xác nhận đọc và đồng ý với điều khoản dịch vụ. Logic đặt lịch và kiểm tra số dư: JavaScript được sử dụng để xử lý nút "Đặt lịch khám". Nó kiểm tra các trường bắt buộc của form, so sánh số dư của người dùng với phí khám. Nếu số dư không đủ, một modal thông báo sẽ hiện lên gợi ý nạp tiền. Nếu đủ, một modal xác nhận khác sẽ xuất hiện trước khi form được gửi đến Controller để xử lý. Hiển thị động: Dữ liệu người dùng và bác sĩ từ ViewBag được sử dụng để điền sẵn vào các trường form, giúp tối ưu trải nghiệm người dùng.

```
@using System.Collections
```

THIẾT KẾ CHƯƠNG TRÌNH

```
@{  
    ViewData["Title"] = "Đặt lịch";  
    Layout = "~/Views/Shared/_Layout.cshtml";  
}  
  
<link rel="stylesheet" href("~/css/Article.css" asp-append-version="true">  
  
<div class="cover_list_news">  
    <div class="name_cate_cover animcc">Đặt Lịch KHÁM</div>  
    <img src("~/images/bg_article.jpg" alt="Đặt lịch">  
</div>  
  
{  
    var bn = ViewBag.UserInfo;  
    var bs = ViewBag.ListDBS[0];  
}  
  
<div class="container" id="loginForm">  
  
<div class="bread-flow">  
    <nav aria-label="breadcrumb">  
        <ol class="breadcrumb">  
            <li class="breadcrumb-item"><a asp-controller="Home" asp-action="Index">Trang chủ</a></li>  
            <li class="breadcrumb-item"><a asp-controller="Home" asp-action="FillterDoctor">Tìm bác sĩ</a></li>  
            <li class="breadcrumb-item" href="~/Home/DetailDoctor?MaBS=@int.Parse(bs[0])">Thông tin bác sĩ</a></li>  
            <li class="breadcrumb-item active" aria-current="page">Đặt Lịch khám</li>  
        </ol>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</nav>
</div>
<div class="sodu">
    <h3 class="">
        @{
            // Định dạng số tiền đẹp hơn
            var amount1 = Convert.ToInt32(ViewBag.UserInfo[9]);
            var SoTienUser = amount1.ToString("#,##0");
        }
    Số dư:
    <span style="color: #fff003;">
        @SoTienUser đ
    </span>
    </h3>
</div>
<div class="form-booking ">
    <form id="bookingForm" class="book-form" asp-controller="Home" asp-action="BookExamineProcess" method="post" enctype="multipart/form-data">
        <div class="title-content_arti">
            <h2>
                Chi tiết lịch hẹn
            </h2>
        </div>
        <h4 class="mb-3">Thông tin bác sĩ</h4>
        <div class="boder-ex">
            <div class="row mb-3">
                <div class="col-md-4">
                    <label for="doctor" class="form-label">Bác sĩ</label>
                    <input type="text" class="form-control" id="doctor" disabled value="@bs[2]">
                </div>
            </div>
        </div>
    </form>
</div>
```

```
</div>

<div class="col-md-4">
    <label for="hospital" class="form-label">Bệnh viện/phòng
khám</label>
    <input type="text" class="form-control" id="hospital" disabled
value="@bs[3]">
</div>

<div class="col-md-4">
    <label for="department" class="form-label">Chuyên khoa</label>
    <input type="text" class="form-control" id="department" disabled
value="@bs[4]">
</div>
</div>

<h4 class="mb-3">Thông tin khách hàng</h4>
<div class="border-ex">
    <div class="row mb-3 ">
        <div class="col-md-6">
            <label for="fullName" class="form-label required">Họ và
tên</label>
            <input type="text" class="form-control" id="fullName"
placeholder="Họ và tên" required value="@bn[1]">
</div>
        <div class="col-md-6">
            <label class="form-label required">Giới tính:</label>
            <div>
                <div class="form-check form-check-inline">
```

```

<input class="form-check-input" type="radio"
name="gender" id="male" value="Nam" @(bn[7] == "Nam" ? "checked" : "")>
    <label class="form-check-label" for="male">Nam</label>
</div>
<div class="form-check form-check-inline">
    <input class="form-check-input" type="radio"
name="gender" id="female" value="Nữ" @(bn[7] == "Nữ" ? "checked" : "")>
        <label class="form-check-label" for="female">Nữ</label>
    </div>
</div>
</div>

<div class="row mb-3">
    <div class="col-md-6">
        <label for="phone" class="form-label required">Số điện
thoại</label>
        <input type="tel" class="form-control" id="phone"
placeholder="Nhập số điện thoại" value="@bn[4]" required>
    </div>
    <div class="col-md-6">
        <label for="email" class="form-label">Email</label>
        <input type="email" class="form-control" id="email"
placeholder="Nhập email" value="@bn[3]">
    </div>
</div>
@if (bn[6] != null)
{
    DateTime birthDate = DateTime.Parse(bn[6].ToString());
}

```

THIẾT KẾ CHƯƠNG TRÌNH

```
var formattedDate = birthDate.ToString("yyyy-MM-dd"); // Định  
dạng yyyy-MM-dd  
  
<div class="mb-3">  
    <label for="dob" class="form-label required">Ngày tháng năm  
sinh</label>  
    <input type="date" class="form-control" id="dob"  
value="@formattedDate" required>  
    </div>  
}  
else  
{  
    <div class="mb-3">  
        <label for="dob" class="form-label required">Ngày tháng năm  
sinh</label>  
        <input type="date" class="form-control" id="dob" required>  
        </div>  
}  
  
<div class="mb-3">  
    <label for="reason" class="form-label required">Lý do khám</label>  
    <textarea class="form-control" id="reason" rows="3"  
placeholder="Triệu chứng của bạn" name="MoTaBenh" required></textarea>  
    </div>  
    <div class="mb-3">  
        <label for="dob" class="form-label required">Hình ảnh bệnh</label>  
        <input type="file" class="form-control" id="dob" required  
name="HinhAnhBenhs" multiple>  
    </div>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</div>

<div class="mb-3 fee" style="text-align: end;">
    @{
        // Định dạng số tiền đẹp hơn
        var amount = Convert.ToInt32(bs[5]);
        var SoTien = amount.ToString("#,##0");
    }

    <label for="department" class="form-label">Phí khám:</label>
    <span>@SoTien</span>
    <span>đ</span>
</div>

<div class="mb-3 form-check">
    <input type="checkbox" class="form-check-input" id="termsAgreement" required>
        <label class="form-check-label" for="termsAgreement">Tôi đã đọc và xác nhận <a href="#">Điều khoản dịch vụ</a> của Bệnh viện .</label>
    </div>
    <input type="text" value="@bs[0]" name="MaBS" style="display: none;">
    <div id="insufficientFundsModal" style="display: none; position: fixed; top: 50%; left: 50%; transform: translate(-50%, -50%); padding: 20px; background-color: white; border: 1px solid #ccc; box-shadow: 0 5px 15px rgba(0,0,0,0.3); z-index: 1000; min-width: 300px; border-radius: 8px;">
        <h4 style="margin-bottom: 15px; color: #dc3545;">Thông báo</h4>
        <p style="margin-bottom: 20px;">Bạn cần nạp thêm tiền để đặt khám</p>
        <div style="display: flex; justify-content: space-between;">
            <button onclick="closeInsufficientFundsModal()" style="padding: 8px 15px; background-color: #6c757d; color: white; border: none; border-radius: 4px; cursor: pointer;">Đóng</button>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<a href="~/Home/AccountBank" style="padding: 8px 15px; background-color: #007bff; color: white; border: none; border-radius: 4px; cursor: pointer;">Nạp tiền</a>

</div>
</div>

<div class="center" >
    <button type="button" class="btn-send-book" onclick="validateAndCheckBalance()">Đặt lịch khám</button>
</div>

</form>

<!-- Modal -->
<div id="confirmationModal" style="display: none; position: fixed; top: 50%; left: 50%; transform: translate(-50%, -50%); padding: 20px; background-color: white; border: 1px solid #ccc; box-shadow: 0 5px 15px rgba(0,0,0,0.3); z-index: 1000;">
    <p>Xin vui lòng chờ bác sĩ xác nhận</p>
    <button onclick="closeModal()">Close</button>
</div>

<!-- Overlay -->
<div id="modalOverlay" style="display: none; position: fixed; top: 0; left: 0; width: 100%; height: 100%; background-color: rgba(0, 0, 0, 0.5); z-index: 999;"></div>

</div>

</div>

<script>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
window.onload = function() {
    document.getElementById("loginForm").scrollIntoView({
        behavior: "smooth",
        block: "start"
    });
};

function validateAndCheckBalance() {
    const form = document.getElementById('bookingForm');
    if (form.checkValidity()) {
        // Lấy số tiền của người dùng và phí khám
        const userBalance = @ViewBag.UserInfo[9];
        const examFee = @bs[5];

        if (userBalance >= examFee) {
            // Nếu đủ tiền thì hiện modal xác nhận và submit form
            showModal();
            form.submit();
        } else {
            // Nếu không đủ tiền thì hiện modal thông báo
            showInsufficientFundsModal();
        }
    } else {
        // Kích hoạt validation mặc định của form
        form.reportValidity();
    }
}

function showInsufficientFundsModal() {
    document.getElementById('insufficientFundsModal').style.display = 'block';
```

```
document.getElementById('modalOverlay').style.display = 'block';  
}  
  
function closeInsufficientFundsModal() {  
    document.getElementById('insufficientFundsModal').style.display = 'none';  
    document.getElementById('modalOverlay').style.display = 'none';  
}  
  
function redirectToAccountBank() {  
    window.location.href = '~/Home/AccountBank';  
}  
  
function showModal() {  
    document.getElementById('confirmationModal').style.display = 'block';  
    document.getElementById('modalOverlay').style.display = 'block';  
}  
  
function closeModal() {  
    document.getElementById('confirmationModal').style.display = 'none';  
    document.getElementById('modalOverlay').style.display = 'none';  
}  
  
</script>
```

5.2.14. Chức năng [Đăng ký tài khoản]

- Controller: Controller này quản lý quá trình đăng ký người dùng. Phương thức Register() chỉ đơn giản là hiển thị trang đăng ký. Phương thức RegisterProcess() (dùng [HttpPost]) nhận tên người dùng, mật khẩu và số điện thoại từ form đăng ký. Nó sử dụng DataModel để thực thi stored procedure REGISTER trong cơ sở dữ liệu. Nếu đăng ký thành công và trả về thông tin người dùng, Controller sẽ lưu tên người

THIẾT KẾ CHƯƠNG TRÌNH

dùng vào Session và chuyển hướng về trang chủ. Ngược lại, nếu đăng ký thất bại, nó sẽ đặt một thông báo lỗi vào ViewBag và chuyển hướng người dùng trở lại trang đăng ký.

```
public IActionResult Register()
{
    LayoutShare();
    return View();
}

[HttpPost]
public IActionResult RegisterProcess(string TenND, string Password, string
sdt)
{
    DataModel db = new DataModel();

    var list = db.get($"EXEC REGISTER N'{TenND}', '{Password}', '{sdt}'");

    if (list.Count > 0 && list[0] is ArrayList arrayList && arrayList.Count >= 2)
    {
        string userName = arrayList[0]?.ToString() ?? "Unknown";
        HttpContext.Session.SetString("taikhoan", userName);

        return RedirectToAction("Index", "Home");
    }
    else
    {
        ViewBag.Error = "Đăng ký không thành công. Vui lòng thử lại.";
        return RedirectToAction("Register", "Home");
    }
}
```

- View: View này tạo giao diện cho trang đăng ký người dùng. Nó bao gồm một form với các trường nhập liệu cho số điện thoại, tên người dùng, mật khẩu và nhập lại mật khẩu, cùng với các thông báo validation trực quan. Giao diện được thiết kế hiện đại với hình ảnh minh họa bên cạnh form. Khi người dùng gửi form, JavaScript sẽ chặn sự kiện submit mặc định, thực hiện validation cơ bản tại phía client (kiểm tra trường trống, khớp mật khẩu). Sau đó, nó gửi dữ liệu form đến RegisterProcess Controller bằng Fetch API. Tùy thuộc vào phản hồi từ server, người dùng sẽ được chuyển hướng hoặc nhận thông báo lỗi.

```
@{  
    ViewData["Title"] = "Đăng ký";  
    Layout = "~/Views/Shared/_Layout.cshtml";  
}  
  
<link rel="stylesheet" href("~/css/Article.css" asp-append-version="true">  
  
<div class="wrapper">  
    <div id="loginForm" class="container main">  
        <div class="form-login">  
            <div class="col-6 side-image">  
                <img src "~/images/bg_login.jpg"/>  
            </div>  
            <div class="col-6 right">  
                <div class="input-box">  
                    <h2>Đăng ký</h2>  
  
                    <form class="needs-validation" input-box" novalidate  
                        id="registrationForm">  
                        <!-- Existing form fields -->  
                        <div class="input-field">
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<input type="tel" class="input-info" id="sdt" required="" name="sdt">
    <label for="sdt">Số điện thoại</label>
    <div class="invalid-feedback">* Vui lòng nhập số điện thoại hợp lệ </div>
    <div class="valid-feedback">
        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-check-circle" viewBox="0 0 16 16">
            <path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14m0 1A8 8 0 1 0 8 0a8 8 0 0 0 0 16" />
            <path d="m10.97 4.97-.02.022-3.473 4.425-2.093-2.094a.75.75 0 0 0-1.06 1.06L6.97 11.03a.75.75 0 0 0 1.079-.02l3.992-4.99a.75.75 0 0 0-1.071-1.05" />
        </svg>
    </div>
</div>

<div class="input-field">
    <input type="text" class="input-info" id="username" required="" name="TenND">
        <label for="username">Tên người dùng</label>
        <div class="invalid-feedback">* Vui lòng nhập tên.</div>
        <div class="valid-feedback">
            <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-check-circle" viewBox="0 0 16 16">
                <path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14m0 1A8 8 0 1 0 8 0a8 8 0 0 0 0 16" />
                <path d="m10.97 4.97-.02.022-3.473 4.425-2.093-2.094a.75.75 0 0 0-1.06 1.06L6.97 11.03a.75.75 0 0 0 1.079-.02l3.992-4.99a.75.75 0 0 0-1.071-1.05" />
            </svg>
        </div>
</div>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</svg>
</div>
</div>

<div class="input-field">
    <input type="password" class="input-info" id="pass"
required="" name="Password">
        <label for="pass">Mật khẩu</label>
        <div class="invalid-feedback">* Phải gồm (chữ hoa, chữ
thường, số và ký tự đặc biệt).</div>
        <div class="valid-feedback">
            <svg xmlns="http://www.w3.org/2000/svg" width="16"
height="16" fill="currentColor" class="bi bi-check-circle" viewBox="0 0 16 16">
                <path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14m0 1A8 8 0 1
0 8 0a8 8 0 0 0 16" />
                <path d="m10.97 4.97-.02.022-3.473 4.425-2.093-
2.094a.75.75 0 0 0-1.06 1.06L6.97 11.03a.75.75 0 0 0 1.079-.0213.992-4.99a.75.75
0 0 0-1.071-1.05" />
            </svg>
        </div>
    </div>

    <div class="input-field">
        <input type="password" class="input-info" id="repass"
required="">
        <label for="repass">Nhập lại mật khẩu</label>
        <div class="invalid-feedback">* Mật khẩu không khớp.</div>
        <div class="valid-feedback">
            <svg xmlns="http://www.w3.org/2000/svg" width="16"
height="16" fill="currentColor" class="bi bi-check-circle" viewBox="0 0 16 16">
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14m0 1A8 8 0 1  
0 8 0a8 8 0 0 0 0 16" />  
  
<path d="m10.97 4.97-02.022-3.473 4.425-2.093-  
2.094a.75.75 0 0 0-1.06 1.06L6.97 11.03a.75.75 0 0 0 1.079-.02l3.992-4.99a.75.75  
0 0 0-1.071-1.05" />  
  
</svg>  
</div>  
</div>  
  
<div class="input-field">  
    <button type="submit" class="submit" >Đăng ký</button>  
</div>  
</form>  
<div class="signin">  
    <span>  
        Bạn đã có tài khoản?  
        <a asp-controller="Home" asp-action="Login">  
            Đăng nhập  
        </a>  
    </span>  
    </div>  
    </div>  
    </div>  
    </div>  
    </div>  
    </div>  
  
<script>  
    // Function to submit registration  
    function submitRegistration() {
```

THIẾT KẾ CHƯƠNG TRÌNH

```
const formData = new FormData(document.getElementById('registrationForm'));

fetch('/Home/RegisterProcess', {
    method: 'POST',
    body: formData,
})
.then(response => {
    if (response.redirected) {
        window.location.href = response.url;
    } else {
        return response.json();
    }
})
.then(data => {
    if (data && data.success) {
        alert('Đăng ký thành công!');
        window.location.href = '/Home/Login';
    } else if (data) {
        alert(data.message || 'Đăng ký thất bại. Vui lòng thử lại.');
    }
})
.catch(error => {
    console.error('Error:', error);
    alert('Có lỗi xảy ra. Vui lòng thử lại sau hoặc liên hệ hỗ trợ.');
});

}

// Event listeners
document.addEventListener('DOMContentLoaded', function() {
```

```
document.getElementById('registrationForm').addEventListener('submit',  
function(e) {  
    e.preventDefault();  
  
    const phoneInput = document.getElementById('sdt');  
    const username = document.getElementById('username').value;  
    const password = document.getElementById('pass').value;  
    const repassword = document.getElementById('repass').value;  
  
    if (!phoneInput.value || !username || !password || password !== repassword)  
    {  
        alert('Vui lòng kiểm tra lại thông tin đăng ký.');//  
        return;  
    }  
  
    submitRegistration();  
});  
});  
</script>
```

5.2.15. Chức năng [Quản lý lịch hẹn khám]

- Controller: Controller này có nhiệm vụ hiển thị lịch sử khám bệnh của người dùng. Phương thức ExamineHistory() thực hiện các bước sau: Đầu tiên, nó gọi LayoutShare() để thiết lập layout chung cho trang. Tiếp theo, Controller lấy mã người dùng (MaND) từ Session hiện tại. Sau đó, nó khởi tạo một đối tượng DataModel và sử dụng nó để gọi stored procedure GetLichKhamInfoByMaND trong cơ sở dữ liệu, truyền vào MaND để lấy tất cả thông tin lịch khám của người dùng. Kết quả trả về từ cơ sở dữ liệu được lưu vào ViewBag.ListLichKham để truyền sang View. Cuối cùng, Controller trả về View để hiển thị dữ liệu lịch sử khám.

THIẾT KẾ CHƯƠNG TRÌNH

```
public IActionResult ExamineHistory()
{
    LayoutShare();

    var taikhoan = HttpContext.Session.GetString("taikhoan");
    ViewData["TaiKhoan"] = taikhoan;
    var MaND = taikhoan;

    DataModel db = new DataModel();
    ViewBag.ListLichKham = db.get($"EXEC GetLichKhamInfoByMaND
{MaND}");

    return View();
}
```

- View: View này được thiết kế để hiển thị lịch sử các cuộc hẹn khám bệnh của người dùng dưới dạng bảng. Điều hướng: Trang có một breadcrumb ở đầu để người dùng dễ dàng điều hướng về trang chủ. Tiêu đề: Tiêu đề lớn "LỊCH SỬ KHÁM" giúp nhận diện nội dung trang. Bảng lịch sử: Thông tin lịch khám được trình bày trong một bảng table table-striped với các cột: Số thứ tự, Tên bác sĩ, Số thứ tự khám, Ghi chú bác sĩ, Thanh toán, và Trạng thái. Hiển thị dữ liệu động: View lặp qua danh sách lịch khám (ViewBag.ListLichKham) để hiển thị từng cuộc hẹn. Số thứ tự được tự động tăng. Xử lý trạng thái: Cột "Trạng thái" hiển thị một nút "Hủy lịch" nếu trạng thái cuộc hẹn là "1" (có thể là trạng thái chờ xác nhận hoặc chưa khám). Khi người dùng nhấp vào "Hủy lịch", một JavaScript confirm sẽ hiện ra để xác nhận hành động hủy. Các trạng thái khác (không phải "1") sẽ hiển thị trực tiếp giá trị trạng thái từ cơ sở dữ liệu

```
@{
```

```
ViewData["Title"] = "Lịch sử khám";
Layout = "~/Views/Shared/_Layout.cshtml";
}

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

<div class="container centerls">
    <div class="bread-flow" style="margin-top: 10px;">
        <nav aria-label="breadcrumb">
            <ol class="breadcrumb">
                <li class="breadcrumb-item"><a asp-controller="Home" asp-action="Index">Trang chủ</a></li>
                <li class="breadcrumb-item active" aria-current="page">Lịch sử khám</li>
            </ol>
        </nav>
    </div>

    <div class="title-content_arti">
        <h2>
            LỊCH SỬ KHÁM
        </h2>
    </div>

    <div class="table-lsk">
        <table class="table table-striped">
            <thead>
                <tr>
                    <th scope="col">STT</th>
                    <th scope="col">Tên bác sĩ</th>
                    <th scope="col">STT khám</th>
                </tr>
            </thead>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<th scope="col">Ghi chú bác sĩ</th>
<th scope="col">Thanh toán</th>
<th scope="col">Trạng thái</th>
</tr>
</thead>
<tbody>
@{
    int stt = 1;
}
@foreach(var a in ViewBag.ListLichKham){
<tr>
    <th scope="row">@stt</th>
    <td>@a[1]</td>
    <td>@a[2]</td>
    <td>@a[7]</td>
    <td>@a[5]</td>
    <td>
        @if(a[3] == "1")
        {
            <a href="@Url.Action("Cancel", "Home")?maCHK=@a[0]" onclick="return validateForm();">
                Hủy lịch
            </a>
        }
        else if(a[3] == "1"){
    }
    else{
        @a[4]
    }
}
```

```
</td>
</tr>
stt = stt + 1;
}

</tbody>
</table>
</div>
</div>

<script>
function validateForm() {
    return confirm('Bạn có chắc chắn muốn hủy lịch này không?');
}
</script>
```

5.2.16. Chức năng [Hủy lịch khám]

- Controller: Controller này xử lý yêu cầu hủy bỏ một cuộc hẹn khám bệnh. Phương thức Cancel(string maCHK) nhận vào mã cuộc hẹn khám (maCHK) từ yêu cầu. Nó tạo một đối tượng DataModel và sử dụng nó để thực thi stored procedure DeleteCHKbyID trong cơ sở dữ liệu, truyền maCHK để xóa (hoặc cập nhật trạng thái hủy) cuộc hẹn tương ứng. Sau khi lệnh thực thi xong, Controller sẽ chuyển hướng người dùng trở lại trang "Lịch sử khám" (ExamineHistory) để họ có thể thấy sự thay đổi (lịch hẹn đã bị hủy hoặc không còn hiển thị).

```
public IActionResult Cancel(string maCHK)
{
    DataModel db = new DataModel();
    ViewBag.List = db.get("EXEC DeleteCHKbyID " + maCHK);

    return RedirectToAction("ExamineHistory", "Home");
```

}

5.2.17. Chức năng[Đăng ký bác sĩ]

- **Controller:** Phương thức DangKyBs() thuộc một Controller dùng để hiển thị giao diện đăng ký tài khoản bác sĩ. Trong hàm này, dữ liệu từ ba bảng KHOABENH, BENHVIEN, và CHUYENNGANH được truy vấn bằng phương thức db.get(...) và truyền sang View thông qua ViewBag với các tên kb, bv, cn. Mục đích là để hiển thị danh sách các khoa bệnh, bệnh viện, và chuyên ngành dưới dạng các dropdown hoặc danh sách chọn khi bác sĩ đăng ký. Sau khi lấy dữ liệu, hàm trả về View tương ứng với trang đăng ký bác sĩ.

```
public IActionResult DangKyBs()
{
    ViewBag.kb = db.get("select * from KHOABENH");
    ViewBag.bv = db.get("select * from BENHVIEN");
    ViewBag.cn = db.get("select * from CHUYENNGANH");
    return View();
}
```

- **Controller** Thực hiện đăng ký bác sĩ: Phương thức ThucHienDKBs xử lý toàn bộ quá trình đăng ký tài khoản bác sĩ sau khi người dùng điền form ở giao diện DangKyBs.cshtml. Hàm nhận nhiều thông tin đầu vào như tên đăng nhập, mật khẩu, thông tin cá nhân, chuyên môn, hình ảnh và tài liệu đính kèm. Các tệp ảnh được lưu vào thư mục wwwroot/Uploads, sau đó thực thi stored procedure DangKyBacSi để lưu toàn bộ thông tin vào cơ sở dữ liệu. Nếu quá trình thành công, hệ thống chuyển hướng sang trang DKTC (đăng ký thành công), còn nếu xảy ra lỗi, người dùng được trả về lại trang đăng ký kèm theo ghi log lỗi nội bộ.

```
public IActionResult ThucHienDKBs(
    string username,
    string password,
    string email,
    string phone,
```

THIẾT KẾ CHƯƠNG TRÌNH

```
string diaChi,  
DateTime birthyear,  
string gender,  
IFormFile hinhanh,  
string cmtnumber,  
DateTime issuedate,  
string issueplace,  
IFormFile cmtimagefront,  
IFormFile cmtimageback,  
string chonChuyenKhoa,  
int namKinhNghiem,  
string noiCongTac,  
string khoa,  
string soChungChi,  
DateTime ngayThangCap,  
IFormFile anhChungChi1,  
IFormFile anhChungChi2,  
string hocVi,  
string hocHam,  
string chuyenMon,  
string thontinbangcap,  
string gioithieu,  
string cacBenhDieuTri,  
string quaTrinhhoc,  
string quaTrinhCongTac,  
string nghienCuuKhoaHoc,  
string giangDay,  
string hoiVienCongTac)  
{  
try
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    DataModel db = new DataModel();  
    // Kiểm tra tệp hình ảnh và lưu vào thư mục "Hinh"  
    if (hinhanh != null && hinhanh.Length > 0)  
    {  
        string filename = Path.GetFileName(hinhanh.FileName); // Lấy tên tệp  
        string path = Path.Combine(Directory.GetCurrentDirectory(),  
"wwwroot", "Uploads", filename); // Đường dẫn lưu tệp  
        using (var stream = new FileStream(path, FileMode.Create))  
        {  
            hinhanh.CopyTo(stream); // Lưu tệp  
        }  
    }  
  
    // Tương tự cho các tệp khác như cmtimagefront, cmtimageback,  
    anhChungChi1, anhChungChi2  
    // Lưu tên tệp vào cơ sở dữ liệu  
    if (cmtimagefront != null && cmtimagefront.Length > 0)  
    {  
        string cmtFrontFileName =  
Path.GetFileName(cmtimagefront.FileName);  
        string cmtFrontPath = Path.Combine(Directory.GetCurrentDirectory(),  
"wwwroot", "Uploads", cmtFrontFileName);  
        using (var stream = new FileStream(cmtFrontPath, FileMode.Create))  
        {  
            cmtimagefront.CopyTo(stream);  
        }  
    }  
  
    if (cmtimageback != null && cmtimageback.Length > 0)
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    string cmtBackFileName =  
        Path.GetFileName(cmtimageback.FileName);  
    string cmtBackPath = Path.Combine(Directory.GetCurrentDirectory(),  
        "wwwroot", "Uploads", cmtBackFileName);  
    using (var stream = new FileStream(cmtBackPath, FileMode.Create))  
    {  
        cmtimageback.CopyTo(stream);  
    }  
}  
  
int randomValue = new Random().Next(1, 10000);  
  
// Sử dụng stored procedure để thực hiện thêm thông tin bác sĩ vào cơ sở  
dữ liệu  
string result = "EXEC DangKyBacSi N'" + username + "','" +  
    + password + "','" +  
    + email + "','" +  
    + phone + "', N'" +  
    + diaChi + "','" +  
    + birthyear.ToString("yyyy-MM-dd") + "', N'" +  
    + gender + "', N'" +  
    + hinhanh.FileName + ", 0 , " +  
    + randomValue +", NULL, 2 ,"  
    + cmtnumber + "','" +  
    + issuedate.ToString("yyyy-MM-dd") + "', N'" +  
    + issueplace + "', N'" +  
    + cmtimagefront.FileName + "','" +  
    + cmtimageback.FileName + "','" +  
    + chonChuyenKhoa + "','" +
```

THIẾT KẾ CHƯƠNG TRÌNH

```
+ namKinhNghiem + ", "
+ noiCongTac + ", "
+ khoa + ", "
+ soChungChi + ", "
+ ngayThangCap.ToString("yyyy-MM-dd") + "", N""
+ anhChungChi1.FileName + "", N""
+ anhChungChi2.FileName + "", N""
+ hocVi + "", N""
+ hocHam + "", N""
+ chuyenMon + "", N""
+ thontinbangcap + "", N""
+ gioithieu + "", N""
+ cacBenhDieuTri + "", N""
+ quaTrinhhoc + "", N""
+ quaTrinhCongTac + "", N""
+ nghienCuuKhoaHoc + "", N""
+ giangDay + "", N""
+ hoiVienCongTac + "", 0 ,2 ;";
db.get(result);

}

catch (Exception ex)
{
    _logger.LogError(ex, "Có lỗi xảy ra khi thực hiện đăng ký bác sĩ.");
    // Nếu có lỗi, giữ lại trang DangKyBs và thông báo lỗi
    return RedirectToAction("DangKyBs", "Bacsis");
}

return RedirectToAction("DKTC", "Bacsis");
```

```
}
```

- **View:** Giao diện DangkyBs.cshtml là trang dùng để bác sĩ đăng ký tài khoản trên hệ thống. View này hiển thị đầy đủ các trường thông tin cần thiết như tên, mật khẩu, năm sinh, số CMT, nơi công tác, chứng chỉ hành nghề, chuyên môn, học hàm, học vị,... Dữ liệu các dropdown như "Bệnh viện", "Khoa", "Chuyên khoa" được lấy từ ViewBag do Controller cung cấp. Trang cũng có modal hiển thị điều khoản sử dụng, bắt buộc bác sĩ đọc và đồng ý trước khi đăng ký. Ngoài HTML, trang còn sử dụng JavaScript để kiểm tra các điều kiện như định dạng mật khẩu, số điện thoại, tuổi, và hiển thị ảnh xem trước khi upload. Khi gửi form, dữ liệu được đẩy về ThucHienDKBs trong BacsisController để xử lý đăng ký.

```
@{  
    ViewData["Title"] = "DangkyBs";  
    <link rel="stylesheet" href="~/css/Dangky.css" asp-append-version="true" />  
}  
  
<body>  
  
    <!-- Modal Điều Khoản -->  
    <div id="modalTerms" class="modal">  
        <div class="modal-content">  
            <span class="close" id="closeModal">&times;</span>  
            <h2>Điều Khoản Sử Dụng</h2>  
            <p>Trước khi đăng ký mở tài khoản và tham gia ứng dụng VOVBACSI,  
            vui lòng đọc kỹ các Điều Khoán...</p>  
            <!-- Thêm nội dung điều khoản ở đây -->  
            <textarea rows="10" cols="50" readonly>  
                ĐIỀU KHOẢN THỎA THUẬN KHI THAM GIA ỦNG DỤNG  
                VOVBACSI  
            </textarea>  
        </div>  
    </div>  
}</body>
```

Trước khi đăng ký mở tài khoản tư vấn, khám chữa bệnh trên ứng dụng VOVBACSI (Sau đây gọi tắt là ứng dụng), vui lòng đọc kỹ các Điều Khoản và điều kiện dưới đây để hiểu rõ quyền lợi và nghĩa vụ hợp pháp của mình trong công tác tư vấn khám chữa bệnh trực tuyến.

Nếu bạn không đồng ý hoặc chưa hiểu rõ những điều khoản này cũng như những quy định pháp luật có liên quan, vui lòng không tạo tài khoản tư vấn trên ứng dụng, và hãy gọi cho chúng tôi theo số điện thoại 19001289 để được hỗ trợ.

Nguyên tắc chính:

- Người tham gia tư vấn, khám chữa bệnh trên ứng dụng phải là bác sĩ, có giấy phép hành nghề khám chữa bệnh hợp pháp, tự nguyện tham gia thực hiện tư vấn về sức khỏe, khám chữa bệnh và các vấn đề về y học trên nền tảng mạng internet với các ứng dụng online và offline trên phương tiện điện thoại thông minh, máy tính, và các phương tiện khác mà có thể cài đặt được ứng dụng.

- Các bác sĩ tham gia ứng dụng phải được sự đồng ý của đơn vị quản lý ứng dụng mới có thể tạo được tài khoản trên ứng dụng và phải cung cấp cho đơn vị quản lý ảnh chụp hoặc bản sao chứng minh thư,(hoặc căn cước công dân) và chứng chỉ hành nghề khám chữa bệnh hợp pháp của mình.

- Thỏa thuận này quy định và ràng buộc giữa hai bên đối với tất cả mọi vấn đề, quyền lợi, trách nhiệm...liên quan đến ứng dụng doctor24 và bacsi24.

1. Quyền lợi và trách nhiệm của Bác sĩ

- Có trách nhiệm điền đầy đủ các thông tin bắt buộc trên trang thông tin cá nhận của mình một cách trung thực, chính xác.

THIẾT KẾ CHƯƠNG TRÌNH

- Cung cấp ảnh chụp hoặc bản sao căn cước công dân và chứng chỉ hành nghề hợp pháp cho đơn vị quản lý ứng dụng.
- Chịu trách nhiệm về nội dung tư vấn, khám chữa bệnh của mình cho bệnh nhân
- Mọi hoạt động thực hiện trên ứng dụng không được vi phạm pháp luật. Quá trình tư vấn, khám chữa bệnh cho người bệnh luôn y đức, hòa nhã và trung thực.
- Không được tư vấn, khám chữa bệnh trong lĩnh vực không thuộc phạm vi chuyên môn của mình.
- Không được có những hành vi xúc phạm tới nhận phẩm của người khác.
- Đảm bảo về tình trạng phương tiện cài đặt ứng dụng của mình hoạt động tốt bao gồm cả vấn đề bảo mật.
- Không được cho người khác, nhất là những người không có chứng chỉ hành nghề khám chữa bệnh sử dụng tài khoản của mình.
- Phải giữ bí mật các thông tin của bệnh nhân trên ứng dụng.
- Được chủ động bố trí thời gian tư vấn.
- Được toàn quyền đặt mức phí tư vấn theo trình độ, khả năng, thời gian tư vấn.
- Được nhận 60% số tiền người nhận tư vấn trả qua ứng dụng, chưa khấu trừ các khoản phải nộp do nhà nước quy định nếu có (như Thuế thu nhập cá nhân...vvv nếu có).

THIẾT KẾ CHƯƠNG TRÌNH

- Thời hạn thanh toán cho bác sĩ (chủ tài khoản) là 03 tháng một lần kể từ khi tài khoản được mở trên ứng dụng.
 - Được tùy ý thay đổi mật khẩu đăng nhập tài khoản của mình.
 - Được bảo mật các thông tin riêng tư và nhận thân.
2. Quyền lợi và trách nhiệm của đơn vị quản lý ứng dụng
- Đảm bảo mức độ hoạt động tối ưu nhất của ứng dụng cho bác sĩ thực hiện hoạt động tư vấn, khám chữa bệnh trên ứng dụng
 - Đảm bảo lưu trữ các tư liệu, nội dung tư vấn, khám chữa bệnh của bác sĩ khi thực hiện hoạt động trên ứng dụng theo quy định.
 - Chịu trách nhiệm kiểm soát, lưu trữ tài khoản trên ứng dụng và thanh toán cho Bác sĩ theo đúng thỏa thuận.
 - Chịu trách nhiệm giữ bí mật các thông tin của Bác sĩ và bệnh nhân.
 - Chịu trách nhiệm thu hộ nhà nước các khoản phải nộp theo quy định (nếu có) của bác sĩ khi tư vấn, khám chữa bệnh trên ứng dụng.
 - Được giữ lại 40% chi phí tư vấn khám chữa bệnh của bệnh nhân trả cho Bác sĩ để chi trả các chi phí như : internet, đường truyền, chăm sóc khách hàng, bảo dưỡng hệ thống...và các chi phí khác
 - Được bảo lưu quyền quản trị, nếu phát hiện chủ tài khoản vi phạm bất cứ quy định nào đơn vị quản lý sẽ xóa tài khoản trên ứng dụng

THIẾT KẾ CHƯƠNG TRÌNH

- Được quyền nâng cấp và cập nhật các phiên bản mới nhất khi cần thiết
 - Sau khi kiểm tra và xác thực thông tin của bác sĩ thì quản trị viên sẽ thông báo tới Bác sĩ (chủ tài khoản) bằng điện thoại hoặc email
3. Thanh toán
- Đơn vị quản lý ứng dụng sẽ thanh toán cho Bác sĩ 03 tháng một lần kể từ khi Bác sĩ bắt đầu được mở tài khoản trên ứng dụng.
 - Số tiền thanh toán cho Bác sĩ là số tiền sau khi đã trừ đi số tiền đơn vị quản lý được giữ lại và số tiền thuế thu nhập cá nhân, hoặc số tiền khác mà Bác sĩ phải nộp theo quy định của luật pháp nếu có.
 - Việc thanh toán này sẽ được thực hiện sau khi hai bên xác nhận số liệu đối soát bằng Email hoặc văn bản giấy.
4. Bất khả kháng
- Trong các trường hợp bất khả kháng như thiên tai, địch họa...hoặc các trường hợp bị đứt kết nối do các nhà mạng cung cấp internet – không thuộc lỗi của đơn vị quản lý ứng dụng thì đơn vị quản lý ứng dụng sẽ khôi phục ngay hoạt động của ứng dụng khi bất khả kháng đó kết thúc
5. Thỏa thuận chung
- Hai bên cam kết những điều khoản trên là hoàn toàn tự nguyện và cam kết thực hiện đúng các điều khoản đó

- Trong trường hợp có vấn đề phát sinh hai bên cùng thống nhất giải quyết, nếu không giải quyết được sẽ đưa ra tòa án thành phố Hà Nội để giải quyết
- Mọi thắc mắc xin gọi về số điện thoại hotline: 1900 1289 để được hỗ trợ.

Tôi đã đọc và đồng ý với tất cả các điều khoản và điều kiện trên cũng như bất kỳ điều khoản nào được chỉnh sửa sau này. Bằng cách click vào nút “ĐỒNG Ý” khi đăng ký tạo tài khoản tư vấn khám chữa bệnh trực tuyến trên ứng dụng, tôi hiểu rằng tôi đang tạo chữ ký điện tử mà nó có giá trị và hiệu lực tương tự như chữ ký tôi ký bằng tay.

```
</textarea>
<br>
<button id="agreeButton">Tôi đồng ý</button>
</div>
</div>

<h2>Đăng ký bác sĩ</h2>
<div class="form-container">
    <form action="/Bacsis/ThucHienDKBs" method="post"
        enctype="multipart/form-data">

        <label for="hinhanh">Ảnh đại diện</label>
        <input type="file" id="hinhanh" name="hinhanh" accept="image/*"
            required>
        <div id="preview-container" style="margin-top: 10px;">
            <img id="preview" src="" alt="Ảnh xem trước" style="max-width: 200px; max-height: 200px; display: none; border: 1px solid #ccc; padding: 5px;">
        </div>
    </form>
</div>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<label for="username">Tên Người dùng <span class="required">(Bắt buộc  
nhập)</span></label>  
<input type="text" id="username" name="username" required>  
  
<div>  
    <label for="password">Mật khẩu <span class="required">(Bắt buộc  
nhập)</span></label>  
    <input type="password" id="password" name="password" required>  
    <p id="password-error" style="color: red; font-size: 12px; margin: 5px  
0 0;"></p>  
</div>  
  
<div>  
    <label for="confirm-password">Nhập lại mật khẩu <span  
class="required">(Bắt buộc nhập)</span></label>  
    <input type="password" id="confirm-password" name="confirm-  
password" required>  
    <p id="confirm-password-error" style="color: red; font-size: 12px;  
margin: 5px 0 0;"></p>  
</div>  
  
<label for="birthyear">Năm sinh <span class="required">(Bắt buộc  
nhập)</span></label>  
<input type="date" id="birthyear" name="birthyear" required>  
    <p id="birth-error" style="color: red; font-size: 12px; margin: 5px 0  
0;"></p>  
  
<label for="cmtnumber">Số CMT <span class="required">(Bắt buộc  
nhập)</span></label>
```

```
<input type="text" id="cmtnumber" name="cmtnumber" placeholder="Số CMT" required pattern="\d*" title="Vui lòng nhập số">

    <input type="date" id="issuedate" name="issuedate" placeholder="Ngày cấp" required>
        <input type="text" id="issueplace" name="issueplace" placeholder="Nơi cấp" required>

    <div>
        <label for="cmt-image-front">Ảnh CMT mặt trước <span class="required">(Bắt buộc nhập)</span></label>
            <input type="file" id="cmt-image-front" name="cmtimagefront" accept="image/*" required>
                <div id="preview-front-container" style="margin-top: 10px;">
                    <img id="preview-front" src="" alt="Xem trước mặt trước" style="max-width: 200px; max-height: 200px; display: none; border: 1px solid #ccc; padding: 5px;">
                </div>
            </div>
        </div>

    <div>
        <label for="cmt-image-back">Ảnh CMT mặt sau <span class="required">(Bắt buộc nhập)</span></label>
            <input type="file" id="cmtimageback" name="cmtimageback" accept="image/*" required>
                <div id="preview-back-container" style="margin-top: 10px;">
                    <img id="preview-back" src="" alt="Xem trước mặt sau" style="max-width: 200px; max-height: 200px; display: none; border: 1px solid #ccc; padding: 5px;">
                </div>
            </div>
        </div>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</div>

<label for="gender">Giới tính <span class="required">(Bắt buộc  
nhập)</span></label>

<select id="gender" name="gender" required>
    <option value="">Chọn giới tính</option>
    <option value="Nam">Nam</option>
    <option value="Nữ">Nữ</option>
</select>

<div>
    <label for="email">Email <span class="required">(Bắt buộc  
nhập)</span></label>
    <input type="email" id="email" name="email" required>
        <div id="email-error" style="color: red; font-size: 12px; display:  
none;">Email không hợp lệ hoặc không tồn tại.</div>
    </div>

<div>
    <label for="phone">Số điện thoại <span class="required">(Bắt buộc  
nhập)</span></label>
    <input type="text" id="phone" name="phone" required>
        <div id="phone-error" style="color: red; font-size: 12px; display: none;">  
        Số điện thoại không đúng định dạng Việt Nam. Phải bắt đầu bằng 03,  
        05, 07, 08, hoặc 09 và có độ dài 10 hoặc 11 chữ số.
    </div>
</div>

<label for="diaChi">Địa chỉ</label>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<input type="text" id="diaChi" name="diaChi" placeholder="Địa chỉ">

    <label for="namKinhNghiem">Năm kinh nghiệm <span class="required">(Bắt buộc nhập)</span></label>
    <input type="number" id="namKinhNghiem" name="namKinhNghiem" required>

    <!--Nơi công tác-->
    <label for="noiCongTac">Nơi công tác <span class="required">(Bắt buộc chọn)</span></label>
    <select id="noiCongTac" name="noiCongTac" required>
        <option value="">-- Hãy chọn bệnh viện công tác --</option>
        @foreach (var benhVien in ViewBag.bv)
        {
            <option value="@benhVien[0]">@benhVien[1]</option>
        }
    </select>

    <label for="khoa">Khoa <span class="required">(Bắt buộc chọn)</span></label>
    <select id="khoa" name="khoa" required>
        <option value="">-- Hãy chọn khoa --</option>
        @if (ViewBag.kb != null)
        {
            foreach (var kb in ViewBag.kb)
            {
                <option value="@kb[0]">@kb[1]</option>
            }
        }
        else
        {
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<option value="">Không có khoa nào</option>
}

</select>

<label for="chuyenKhoa">Chuyên khoa <span class="required">(Bắt buộc  
chọn)</span></label>
<select id="chuyenKhoa" name="chuyenKhoa" required>
    <option value="">-- Hãy chọn chuyên khoa --</option>
    @if (ViewBag.cn != null)
    {
        foreach (var ck in ViewBag.cn)
        {
            <option value="@ck[1]">@ck[1]</option>
        }
    }
    else
    {
        <option value="">Không có chuyên khoa nào</option>
    }
</select>

<label for="soChungChi">Số chứng chỉ hành nghề <span  
class="required">(Bắt buộc nhập)</span></label>
<input type="text" id="soChungChi" name="soChungChi" required >

<label for="ngayThangCap">Ngày tháng cấp <span class="required">(Bắt  
buộc nhập)</span></label>
<input type="date" id="ngayThangCap" name="ngayThangCap" required>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<label for="noiCap">Nơi cấp <span class="required">(Bắt buộc  
nhập)</span></label>  
  
<input type="text" id="noiCap" name="noiCap" required>  
  
<div>  
    <label for="anhChungChi1">Ảnh chứng chỉ hành nghề 1 <span  
class="required">(Bắt buộc nhập)</span></label>  
    <input type="file" id="anhChungChi1" name="anhChungChi1"  
accept="image/*" required>  
    <div id="preview-container1" style="margin-top: 10px;">  
        <img id="preview1" src="" alt="Xem trước chứng chỉ 1" style="max-  
width: 200px; max-height: 200px; display: none; border: 1px solid #ccc; padding:  
5px;">  
    </div>  
</div>  
  
<div>  
    <label for="anhChungChi2">Ảnh chứng chỉ hành nghề 2 <span  
class="required">(Bắt buộc nhập)</span></label>  
    <input type="file" id="anhChungChi2" name="anhChungChi2"  
accept="image/*" required>  
    <div id="preview-container2" style="margin-top: 10px;">  
        <img id="preview2" src="" alt="Xem trước chứng chỉ 2" style="max-  
width: 200px; max-height: 200px; display: none; border: 1px solid #ccc; padding:  
5px;">  
    </div>  
</div>  
  
<label for="hocVi">Học vị</label>  
<select id="hocVi" name="hocVi">
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<option value="">-- Hãy chọn học vị --</option>
<option value="Cử nhân">Cử nhân</option>
<option value="Thạc sĩ">Thạc sĩ</option>
<option value="Tiến sĩ">Tiến sĩ</option>
<option value="Giáo sư">Giáo sư</option>
<option value="Phó giáo sư">Phó giáo sư</option>
</select>

<label for="hocHam">Học hàm</label>
<select id="hocHam" name="hocHam">
    <option value="">-- Hãy chọn học hàm --</option>
    <option value="Phó giáo sư">Phó giáo sư</option>
    <option value="Giáo sư">Giáo sư</option>
</select>

<label for="chuyenMon">Chuyên môn <span class="required">(Bắt buộc  
nhập)</span></label>
<select id="chuyenMon" name="chuyenMon" required>
    <option value="">-- Hãy chọn chuyên môn --</option>
    <option value="Bác sĩ">Bác sĩ</option>
    <option value="Y tá">Y tá</option>
    <option value="Dược sĩ">Dược sĩ</option>
    <option value="Điều dưỡng viên">Điều dưỡng viên</option>
</select>

<label for="cacBenzDieuTri">Các bệnh điều trị <span  
class="required">(Bắt buộc nhập)</span></label>
<textarea id="cacBenzDieuTri" name="cacBenzDieuTri" required></textarea>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<label for="gioithieu">Giới thiệu <span class="required">(Bắt buộc  
nhập)</span></label>  
<textarea id="gioithieu" name="gioithieu" required></textarea>  
  
<label for="thontinbangcap">Thông tin bằng cấp</label>  
<textarea id="thontinbangcap" name="thontinbangcap" required></textarea>  
  
<label for="quaTrinhhoc">Quá trình Học</label>  
<textarea id="quaTrinhhoc" name="quaTrinhhoc" required></textarea>  
  
<label for="quaTrinhCongTac">Quá trình công tác</label>  
<textarea id="quaTrinhCongTac" name="quaTrinhCongTac" required></textarea>  
  
<label for="khamChuaBenh">Khám chữa bệnh</label>  
<textarea id="khamChuaBenh" name="khamChuaBenh" required></textarea>  
  
<label for="nghienCuuKhoaHoc">Nghiên cứu khoa học</label>  
<textarea id="nghienCuuKhoaHoc" name="nghienCuuKhoaHoc" required></textarea>  
  
<label for="giangDay">Giảng dạy</label>  
<textarea id="giangDay" name="giangDay" required></textarea>  
  
<label for="hoiVienCongTac">Hội viên công tác</label>  
<textarea id="hoiVienCongTac" name="hoiVienCongTac" required></textarea>
```

```
<label for="quanLy">Quản lý</label>
<textarea id="quanLy" name="quanLy" required></textarea>

<button type="submit" class="submit-button" >Đăng ký</button>

<a href="@Url.Action("Index","Home")" class="quaylai-button">Quay lại
Trang chủ</a>
</form>
</div>
<script>
// Hàm thiết lập xem trước ảnh
function setupImagePreview(inputId, previewId) {
    const fileInput = document.getElementById(inputId);
    const previewImage = document.getElementById(previewId);

    fileInput.addEventListener("change", function () {
        const file = fileInput.files[0]; // Lấy file đã chọn

        if (file) {
            const reader = new FileReader();

            // Khi file được đọc xong, hiển thị ảnh
            reader.onload = function (e) {
                previewImage.src = e.target.result; // Gán URL ảnh
                previewImage.style.display = "block"; // Hiển thị ảnh
            };

            reader.readAsDataURL(file); // Đọc file dưới dạng URL
        } else {
            // Nếu không có ảnh, ẩn ảnh xem trước
        }
    });
}
```

THIẾT KẾ CHƯƠNG TRÌNH

```
previewImage.src = "";
previewImage.style.display = "none";
}
});
}

// Sử dụng setupImagePreview cho từng trường input file
setupImagePreview("hinhanh", "preview"); // Xem trước ảnh đại diện
setupImagePreview("cmt-image-front", "preview-front"); // Xem trước ảnh CMT
mặt trước
setupImagePreview("cmtimageback", "preview-back"); // Xem trước ảnh CMT
mặt sau
setupImagePreview("anhChungChi1", "preview1"); // Xem trước ảnh chứng chỉ 1
setupImagePreview("anhChungChi2", "preview2"); // Xem trước ảnh chứng chỉ 2

const passwordInput = document.getElementById("password");
const confirmPasswordInput = document.getElementById("confirm-
password");
const passwordError = document.getElementById("password-error");
const confirmPasswordError = document.getElementById("confirm-password-
error");

// Kiểm tra mật khẩu khi gõ
passwordInput.addEventListener("input", function () {
    const password = passwordInput.value;
    const passwordRegex = /^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$/;

    if (!passwordRegex.test(password)) {
        passwordError.textContent = "Mật khẩu phải gồm ít nhất 8 ký tự, bao gồm
cả chữ và số.";
    }
})
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    } else {
        passwordError.textContent = "";
    }
});

// Kiểm tra mật khẩu nhập lại khi gõ
confirmPasswordInput.addEventListener("input", function () {
    const password = passwordInput.value;
    const confirmPassword = confirmPasswordInput.value;

    if (confirmPassword !== password) {
        confirmPasswordError.textContent = "Mật khẩu nhập lại không khớp.";
    } else {
        confirmPasswordError.textContent = "";
    }
});

// Lấy các phần tử modal và các nút
const modal = document.getElementById("modalTerms");
const closeModal = document.getElementById("closeModal");
const closeButton = document.getElementById("closeButton");
const agreeButton = document.getElementById("agreeButton");

// Hiển thị modal khi trang được tải (có thể thay đổi tùy vào yêu cầu)
window.addEventListener("load", function() {
    modal.style.display = "block"; // Mở modal khi trang tải
});

// Xử lý khi nhấn nút "Tôi đồng ý"
agreeButton.addEventListener("click", function() {
```

THIẾT KẾ CHƯƠNG TRÌNH

```
modal.style.display = "none"; // Đóng modal
alert("Cảm ơn bạn đã đồng ý với các điều khoản.");
});

// Xử lý khi nhấn nút "Hủy bỏ" nhưng không đóng modal
closeButton.addEventListener("click", function() {
    // Không làm gì ở đây, chỉ dừng hành động mặc định của nút
    // Điều này sẽ không đóng modal
});

// Xử lý khi bấm nút đóng (×)
closeModal.addEventListener("click", function() {
    modal.style.display = "none"; // Đóng modal khi nhấn nút ×
});

// Xử lý khi bấm ngoài modal để đóng
window.addEventListener("click", function(event) {
    if (event.target === modal) {
        modal.style.display = "none"; // Đóng modal nếu nhấn bên ngoài
    }
});

// Đăng ký thành công
function handleSubmit() {
    // Hiển thị thông báo trong console
    console.log("Đăng ký thành công, chờ đơn xác nhận từ người quản lý");

    // Nếu muốn ngừng việc gửi form, bạn có thể thêm dòng này
    event.preventDefault();
}
```

THIẾT KẾ CHƯƠNG TRÌNH

```
// Kiểm tra lần cuối khi submit
    document.getElementById("passwordForm").addEventListener("submit",
function (event) {
    const password = passwordInput.value;
    const confirmPassword = confirmPasswordInput.value;
    const passwordRegex = /^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$/;

    // Nếu có lỗi thì ngăn gửi form
    if (!passwordRegex.test(password) || confirmPassword !== password) {
        event.preventDefault();
    }
});

//kiểm tra tuổi
const birthInput = document.getElementById("birthyear");
const birthError = document.getElementById("birth-error");

// Kiểm tra khi ngày sinh thay đổi
birthInput.addEventListener("change", function () {
    const birthDate = new Date(birthInput.value);
    const today = new Date();

    // Tính tuổi
    const age = today.getFullYear() - birthDate.getFullYear();
    const monthDiff = today.getMonth() - birthDate.getMonth();
    const dayDiff = today.getDate() - birthDate.getDate();

    // Kiểm tra nếu chưa đủ 18 tuổi
});
```

```
if (age < 18 || (age === 18 && (monthDiff < 0 || (monthDiff === 0 && dayDiff < 0)))) {  
    birthError.textContent = "Bạn phải đủ 18 tuổi.";  
} else {  
    birthError.textContent = "";  
}  
});  
  
const phoneInput = document.getElementById("phone");  
const phoneError = document.getElementById("phone-error");  
  
// Kiểm tra số điện thoại khi gõ  
phoneInput.addEventListener("input", function () {  
    const phone = phoneInput.value;  
    const phoneRegex = /^(0[3|5|7|8|9])+([0-9]{8})$/;  
  
    if (!phoneRegex.test(phone)) {  
        phoneError.textContent = "Số điện thoại không đúng định dạng Việt Nam.  
Phải bắt đầu bằng 03, 05, 07, 08, hoặc 09 và có độ dài 10 hoặc 11 chữ số.";  
    } else {  
        phoneError.textContent = ""; //Ẩn thông báo lỗi nếu hợp lệ  
    }  
});  
  
</script>  
  
</body>
```

- **View:** View DKTC.cshtml là trang thông báo cho người dùng sau khi đăng ký tài khoản bác sĩ thành công. Giao diện đơn giản, không sử dụng layout chung (Layout = null) để tạo cảm giác nhẹ nhàng và tập trung vào thông điệp xác nhận. Trang hiển thị dòng thông báo cảm ơn và cung cấp nút "Quay lại trang chủ", liên kết đến action

Index của HomeController. Giao diện này giúp khép lại quá trình đăng ký một cách rõ ràng, đồng thời mang lại trải nghiệm thân thiện cho người dùng mới.

```
@{  
    ViewData["Title"] = "DKTC";  
    Layout = null;  
}  
  
<h2>Đăng ký thành công!</h2>  
<p>Cảm ơn bạn đã đăng ký. Chúng tôi sẽ liên hệ với bạn sớm nhất.</p>  
<a href="@Url.Action("Index", "Home")" class="btn btn-success">Quay lại trang chủ</a>
```

5.2.18. Chức năng [Quản lý hồ sơ bệnh nhân]

- **Controller :** Phương thức HoSoBN() được sử dụng để hiển thị danh sách hồ sơ bệnh nhân của bác sĩ hiện đang đăng nhập. Hàm lấy userId từ Session (dựa trên thông tin đăng nhập trước đó), sau đó gọi stored procedure sp_LayDanhSachHoSoTheoMaBacSi để lấy danh sách hồ sơ tương ứng từ cơ sở dữ liệu. Kết quả được gán vào ViewBag.list và truyền sang View để hiển thị. Mục tiêu của phương thức là cho phép bác sĩ theo dõi các hồ sơ bệnh án của bệnh nhân mà họ đang phụ trách.

```
public IActionResult HoSoBN()  
{  
  
    var userId = _session.GetString("userId");  
    ViewBag.list = db.get("EXEC sp_LayDanhSachHoSoTheoMaBacSi "+  
    userId);  
    return View();  
}
```

- **Controller:** Giao diện HoSoBN.cshtml hiển thị danh sách các hồ sơ bệnh nhân gắn với bác sĩ đang đăng nhập. View sử dụng layout riêng Bs_Layout.cshtml và nhận dữ liệu danh sách từ ViewBag.list do Controller cung cấp. Mỗi hồ sơ hiển thị thông

THIẾT KẾ CHƯƠNG TRÌNH

tin như mã, mô tả bệnh, tên bệnh nhân, số điện thoại, giới tính, địa chỉ, ảnh đại diện và ảnh mô tả bệnh. Trang có chức năng tìm kiếm bệnh nhân theo số điện thoại và sửa thông tin mô tả bệnh bằng modal popup. Ngoài ra, còn có nút “xóa” để bác sĩ xóa hồ sơ bệnh nhân sau khi xác nhận. Các thao tác đều được hỗ trợ bởi JavaScript giúp nâng cao trải nghiệm người dùng.

```
public IActionResult ThemHs(string patientPhone, string patientDescription)
{

```

```
    ViewBag.list = db.get("exec AddPatientRecordByPhone "+ patientPhone
+ ",N" + patientDescription + "");
    return RedirectToAction("HoSoBN", "Bacsis");
}
```

- **Controller:** Xóa hồ sơ: Phương thức XoaHs(string id) xử lý yêu cầu xóa hồ sơ bệnh nhân theo mã hồ sơ (id). Khi bác sĩ nhấn nút xóa ở giao diện HoSoBN, phương thức này sẽ được gọi và thực hiện truy vấn EXEC DeletePatientRecord để xóa hồ sơ trong cơ sở dữ liệu. Sau khi xóa, người dùng được chuyển hướng trở lại trang danh sách hồ sơ bệnh nhân (HoSoBN) để cập nhật lại dữ liệu. Đây là chức năng giúp bác sĩ loại bỏ các hồ sơ không cần thiết một cách trực tiếp trên giao diện.

```
public IActionResult XoaHs(string id){
```

```
    ViewBag.list=db.get("EXEC DeletePatientRecord "+ id);
    return RedirectToAction("HoSoBN", "Bacsis");
}
```

- **Controller** Sửa hồ sơ: Phương thức Suahoso xử lý việc cập nhật mô tả bệnh cho một hồ sơ bệnh nhân cụ thể. Khi bác sĩ chỉnh sửa thông tin trên giao diện HoSoBN và gửi form, phương thức này được gọi với hai tham số patientId và patientDescription. Sau đó, stored procedure UpdatePatientRecord được thực thi để cập nhật mô tả bệnh trong cơ sở dữ liệu. Dữ liệu trả về được gán tạm vào ViewBag.list nhưng không sử dụng, và sau cùng hệ thống điều hướng người dùng trở về trang danh sách hồ sơ (HoSoBN) để xem thông tin mới cập nhật.

```
public IActionResult Suahoso(string patientId, string patientDescription){  
  
    ViewBag.list=db.get("EXEC UpdatePatientRecord "+ patientId +",N"+  
patientDescription + "");  
  
    return RedirectToAction("HoSoBN", "Bacsis");  
}
```

- **View:** Giao diện HoSoBN.cshtml hiển thị danh sách các hồ sơ bệnh nhân gắn với bác sĩ đang đăng nhập. View sử dụng layout riêng Bs_Layout.cshtml và nhận dữ liệu danh sách từ ViewBag.list do Controller cung cấp. Mỗi hồ sơ hiển thị thông tin như mã, mô tả bệnh, tên bệnh nhân, số điện thoại, giới tính, địa chỉ, ảnh đại diện và ảnh mô tả bệnh. Trang có chức năng tìm kiếm bệnh nhân theo số điện thoại và sửa thông tin mô tả bệnh bằng modal popup. Ngoài ra, còn có nút “xóa” để bác sĩ xóa hồ sơ bệnh nhân sau khi xác nhận. Các thao tác đều được hỗ trợ bởi JavaScript giúp nâng cao trải nghiệm người dùng.

```
@{  
    ViewData["Title"] = "HoSoBN";  
    Layout = "~/Views/Shared/Bs_Layout.cshtml";  
    <link rel="stylesheet" href("~/css/hosobn.css" asp-append-version="true" />  
}  
  
<body>  
    <!-- Phàn tìm kiếm theo số điện thoại -->  
    <div class="search-container">  
        <input type="text" id="searchInput" placeholder="Nhập số điện thoại bệnh nhân" />  
        <button onclick="searchPatient()">Tìm kiếm</button>  
    </div>  
  
    <!-- Danh sách hồ sơ bệnh nhân -->  
    <div class="patient-list">
```

```
<h2>Danh Sách Hồ Sơ Bệnh Nhân</h2>
<table>
    <thead>
        <tr>
            <th>Mã hồ sơ</th>
            <th>Mô Tả bệnh</th>
            <th>Tên Bệnh Nhân</th>
            <th>Số Điện Thoại</th>
            <th>Địa Chỉ</th>
            <th>Giới Tính</th>
            <th>Ảnh cá nhân</th>
            <th>Ảnh Mô Tả bệnh</th>
            <th>Hành động</th>
        </tr>
    </thead>
    <tbody id="patientTableBody">
        @foreach (var hs in ViewBag.list) {
            <tr>
                <td>@hs[0]</td>
                <td>@hs[1]</td>
                <td>@hs[3]</td>
                <td>@hs[5]</td>
                <td>@hs[6]</td>
                <td>@hs[7]</td>
                <td>
                    
                </td>
                <td>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
  
</td>  
  
<td>  
    <a href="~/Bacsis/XoaHs?id=@hs[0]" onclick="return  
confirm('Bạn có chắc chắn muốn xóa không?')">xóa</a>  
    <button onclick="editPatient(@hs[0], '@hs[5]',  
'@hs[1]')">sửa</button>  
</td>  
</tr>  
<br>  
</tbody>  
</table>  
</div>  
  
<!-- Modal Sửa Hồ Sơ -->  
<div id="editPatientFormModal" class="modal" style="display: none;">  
    <div class="modal-content">  
        <span class="close" onclick="hideForm()">&times;</span>  
        <h3>Sửa Hồ Sơ Bệnh Nhân</h3>  
  
        <!-- Form sửa hồ sơ bệnh nhân -->  
        <form id="editPatientForm" asp-controller="Bacsis" asp-action="Suahoso"  
method="post">  
            <input type="hidden" id="patientId" name="patientId" />  
  
            <label>Số Điện Thoại:</label>  
            <input type="text" id="editPatientPhone" name="patientPhone" readonly  
/>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<label>Mô Tả Bệnh:</label>
<textarea id="editPatientDescription"
name="patientDescription"></textarea>

<button type="submit">Cập Nhật</button>
</form>
</div>
</div>

<script src="~/js/hosobn.js"></script> <!-- Đường dẫn tới JavaScript -->

<script>
    // Hàm tìm kiếm bệnh nhân theo số điện thoại
    function searchPatient() {
        // Lấy giá trị từ ô nhập liệu tìm kiếm
        var searchInput = document.getElementById('searchInput').value.toLowerCase();

        // Lấy tất cả các hàng trong bảng bệnh nhân
        var rows = document.querySelectorAll('#patientTableBody tr');

        // Lặp qua tất cả các hàng và kiểm tra số điện thoại
        rows.forEach(function(row) {
            var phoneCell = row.cells[3]; // Cột số điện thoại (index 3)
            var phone = phoneCell.textContent || phoneCell.innerText;

            // Kiểm tra nếu số điện thoại có chứa chuỗi tìm kiếm
            if (phone.toLowerCase().includes(searchInput)) {
                row.style.display = ""; // Hiển thị hàng
            } else {
        
```

```
row.style.display = 'none'; // Ẩn hàng
}
});
}

// Hàm để mở form sửa bệnh nhân với dữ liệu đã có
function editPatient(id, phone, description) {
    // Cập nhật tiêu đề form sửa
    document.getElementById('editPatientFormModal').style.display = 'block';

    // Đưa dữ liệu vào form sửa
    document.getElementById('patientId').value = id;
    document.getElementById('editPatientPhone').value = phone;
    document.getElementById('editPatientDescription').value = description;
}

// Hàm để ẩn form
function hideForm() {
    document.getElementById('editPatientFormModal').style.display = 'none';
}

</script>
</body>
```

5.2.19. Chức năng[Quản lý lịch khám]

- **Controller** lịch hẹn khám: Phương thức LichHenKham() có nhiệm vụ hiển thị danh sách lịch hẹn khám bệnh của người dùng hiện tại. Hàm lấy userId từ session (tức người đang đăng nhập), sau đó gọi stored procedure GetAllCuocHenByMaND để truy xuất toàn bộ các cuộc hẹn liên quan đến người dùng đó. Kết quả được truyền

qua ViewBag.list để hiển thị trên giao diện View. Chức năng này giúp người dùng theo dõi tất cả lịch hẹn đã đặt với bác sĩ một cách trực quan và thuận tiện.

```
public ActionResult LichHenKham()
{
    var userId = _session.GetString("userId");
    ViewBag.list = db.get("EXEC GetAllCuocHenByMaND " + userId);
    return View();
}
```

- View LichHenKham.cshtml: Giao diện LichHenKham.cshtml hiển thị danh sách các cuộc hẹn **chưa được xác nhận** mà bác sĩ cần xử lý. View sử dụng layout chung Bs_Layout.cshtml và dữ liệu từ ViewBag.list, được truyền từ Controller LichHenKham. Mỗi hàng trong bảng thể hiện thông tin chi tiết của một cuộc hẹn như mã cuộc hẹn, tên người dùng, số điện thoại, trạng thái, thời gian hẹn và mô tả bệnh. Giao diện có hai nút chức năng: **Xác nhận** và **Hủy**, mỗi nút gửi form tương ứng đến phương thức Updatecuochen của BacsisController để cập nhật trạng thái cuộc hẹn. View này hỗ trợ bác sĩ thao tác nhanh, thuận tiện mà không cần chuyển trang hay thao tác phức tạp.

- **Controller** đã xác nhận: Phương thức DaXacNhan() có chức năng hiển thị danh sách các cuộc hẹn khám bệnh đã được xác nhận dành cho người dùng hiện tại. Hàm lấy userId từ Session, sau đó thực hiện stored procedure GetAllCuocHenByMaND да XN để truy xuất những lịch hẹn đã được bác sĩ xác nhận. Dữ liệu kết quả được đưa vào ViewBag.list và trả về View để hiển thị. Chức năng này giúp người dùng dễ dàng kiểm tra các lịch hẹn đã được chấp thuận, phục vụ cho việc chủ động đi khám.

```
@{
    ViewData["Title"] = "LichHenKham";
    Layout = "~/Views/Shared/Bs_Layout.cshtml";
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<link rel="stylesheet" href="~/css/HenLichKham.css" asp-append-version="true" />
}

<h2> Chưa xác nhận </h2>

<div class="container">
    <table class="table table-striped">
        <thead>
            <tr>
                <th>Mã Cuộc Hẹn</th>
                <th>Tên Người Dùng</th>
                <th>SĐT</th>
                <th>Trạng thái</th>
                <th>Thời Gian Hẹn</th>
                <th>Mô Tả Bệnh</th>
                <th>Chức năng</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var appointment in ViewBag.list)
            {
                <tr>
                    <td>@appointment[0]</td>
                    <td>@appointment[2]</td>
                    <td>@appointment[3]</td>
                    <td>@appointment[7]</td>
                    <td>@appointment[8]</td>
                    <td>@appointment[15]</td>
                    <td>
                        <div class="button-container">
                            <form action="~/Bacsis/Updatecuochen" method="post">
```

```

<input type="hidden" name="id" value="@appointment[0]" />
<input type="hidden" name="matt" value="2" />
<button type="submit">Xác nhận</button>
</form>
<form action="~/Bacsis/Updatecuochen" method="post">
<input type="hidden" name="id" value="@appointment[0]" />
<input type="hidden" name="matt" value="3" />
<button class="huy">Hủy</button>
</form>
</div>
</td>
</tr>
}
</tbody>
</table>
</div>

```

5.2.20. Chức năng [Xác nhận lịch khám]

- **Controller DaXacNhan:** Phương thức có nhiệm vụ lấy danh sách tất cả các cuộc hẹn đã được bác sĩ xác nhận của người dùng đang đăng nhập. Hàm lấy userId từ Session, sau đó gọi stored procedure GetAllCuocHenByMaNDDaXN để truy vấn các lịch hẹn ở trạng thái "Đã xác nhận" từ cơ sở dữ liệu. Dữ liệu kết quả được lưu trong ViewBag.list và truyền sang View tương ứng để hiển thị. Chức năng này giúp người dùng theo dõi những cuộc hẹn sắp đến hoặc đã được đồng ý, hỗ trợ quá trình chuẩn bị và tra cứu lịch khám một cách thuận tiện.

```

public ActionResult DaXacNhan()
{
    var userId = _session.GetString("userId");
    ViewBag.list = db.get("EXEC GetAllCuocHenByMaNDDaXN " + userId);
    return View();
}

```

}

- **View:** Giao diện DaXacNhan.cshtml hiển thị danh sách các cuộc hẹn đã được bác sĩ xác nhận. Dữ liệu được lấy từ ViewBag.list (do Controller DaXacNhan() cung cấp) và trình bày trong bảng có các cột như mã cuộc hẹn, tên người dùng, số điện thoại, trạng thái, thời gian và mô tả bệnh. Mỗi cuộc hẹn đều có hai nút chức năng: "Hoàn thành" – gửi form đến UpdatecuochenTT để chuyển trạng thái sang hoàn tất và cập nhật tài khoản; "Hủy" – gọi Updatecuochen để đổi trạng thái sang hủy. Giao diện này giúp bác sĩ theo dõi và xử lý các cuộc hẹn đang chờ thực hiện một cách trực tiếp và dễ dàng.

```
@{  
    ViewData["Title"] = "DaXacNhan";  
    Layout = "~/Views/Shared/Bs_Layout.cshtml";  
    <link rel="stylesheet" href("~/css/HenLichKham.css" asp-append-version="true" />  
}  
  
<h2>Đã xác nhận </h2>  
<div class="container">  
    <table class="table table-striped">  
        <thead>  
            <tr>  
                <th>Mã Cuộc Hẹn</th>  
                <th>Tên Người Dùng</th>  
                <th>SĐT</th>  
                <th>Trạng thái</th>  
                <th>Thời Gian Hẹn</th>  
                <th>Mô Tả Bệnh</th>  
                <th>Chức năng</th>  
            </tr>  
        </thead>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</thead>
<tbody>
    @foreach (var appointment in ViewBag.list)
    {
        <tr>
            <td>@appointment[0]</td>
            <td>@appointment[2]</td>
            <td>@appointment[3]</td>
            <td>@appointment[7]</td>
            <td>@appointment[8]</td>
            <td>@appointment[15]</td>
            <td>
                <div class="button-container">
                    <form action="/Bacsis/UpdatecuochenTT" method="post">
                        <input type="hidden" name="id" value="@appointment[0]" />
                        <input type="hidden" name="matt" value="4" />
                        <button type="submit">Hoàn thành</button>
                    </form>
                    <form action="/Bacsis/Updatecuochen" method="post">
                        <input type="hidden" name="id" value="@appointment[0]" />
                        <input type="hidden" name="matt" value="3" />
                        <button class="huy">Hủy</button>
                    </form>
                </div>
            </td>
        </tr>
    }
</tbody>
</table>
</div>
```

5.2.21. Chức năng [Xác nhận đã hoàn thành]

- **Controller** đã hoàn thành: Phương thức DaHoanThanh() đảm nhiệm việc hiển thị danh sách các cuộc hẹn khám bệnh **đã hoàn tất** của người dùng đang đăng nhập. Sau khi lấy userId từ Session, hàm thực hiện stored procedure GetAllCuocHenByMaNDDaHT, có nhiệm vụ truy vấn các lịch hẹn đã được bác sĩ xác nhận và thực hiện xong. Kết quả được gán vào ViewBag.list, sau đó truyền sang View để hiển thị thông tin dưới dạng bảng hoặc danh sách. Mục tiêu của chức năng này là cho phép bệnh nhân dễ dàng tra cứu lại lịch sử khám bệnh đã thực hiện, phục vụ cho các nhu cầu theo dõi sức khỏe cá nhân hoặc tái khám.

```
public ActionResult DaHoanThanh()
{
    var userId = _session.GetString("userId");
    ViewBag.list = db.get("EXEC GetAllCuocHenByMaNDDaHT " + userId);
    return View();
}
```

- **View:** View DaHoanThanh.cshtml hiển thị danh sách các cuộc hẹn khám bệnh đã hoàn tất của bác sĩ. Giao diện sử dụng layout Bs_Layout.cshtml và dữ liệu được truyền từ ViewBag.list. Bảng hiển thị các thông tin như mã cuộc hẹn, tên bệnh nhân, số điện thoại, trạng thái, thời gian hẹn, mô tả bệnh và số tiền đã thanh toán. Đây là phần quan trọng giúp bác sĩ theo dõi lại các cuộc khám đã diễn ra, kiểm tra lịch sử tư vấn cũng như xác minh thông tin thanh toán. Tất cả thông tin được trình bày trực quan và dễ theo dõi, hỗ trợ công tác thống kê và báo cáo hiệu quả.

```
@{
    ViewData["Title"] = "DaHoanThanh";
    Layout = "~/Views/Shared/Bs_Layout.cshtml";
    <link rel="stylesheet" href="~/css/HenLichKham.css" asp-append-version="true" />
```

THIẾT KẾ CHƯƠNG TRÌNH

```
}

<h2>Đã hoàn thành </h2>

<div class="container">

    <table class="table table-striped">
        <thead>
            <tr>
                <th>Mã Cuộc Hẹn</th>
                <th>Tên Người Dùng</th>
                <th>SĐT</th>
                <th>Trạng thái</th>
                <th>Thời Gian Hẹn</th>
                <th>Mô Tả Bệnh</th>
                <th>Số tiền thanh toán</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var appointment in ViewBag.list)
            {
                <tr>
                    <td>@appointment[0]</td>
                    <td>@appointment[2]</td>
                    <td>@appointment[3]</td>
                    <td>@appointment[7]</td>
                    <td>@appointment[8]</td>
                    <td>@appointment[15]</td>
                    <td>@appointment[14]</td>
                </tr>
            }
        </tbody>
    </table>
```

```
</div>
```

5.2.22. Chức năng [Xác nhận đã bị hủy]

- **Controller** đã bị hủy: Phương thức DaBiHuy() dùng để hiển thị các lịch hẹn đã bị hủy của người dùng đang đăng nhập. Sau khi lấy userId từ session, hệ thống gọi stored procedure GetAllCuocHenByMaNDDaHuy để lấy ra toàn bộ các lịch hẹn mà người bệnh hoặc bác sĩ đã hủy vì lý do nào đó. Dữ liệu sau khi lấy về được lưu vào ViewBag.list để hiển thị trên View. Tính năng này giúp người dùng dễ dàng kiểm tra lại các cuộc hẹn không thành công, từ đó có thể đặt lại lịch mới hoặc liên hệ lại với bác sĩ khi cần thiết.

```
public ActionResult DaBiHuy()
{
    var userId = _session.GetString("userId");
    ViewBag.list = db.get("EXEC GetAllCuocHenByMaNDDaHuy " + userId);
    return View();
}
```

- **View:** View DaBiHuy.cshtml hiển thị danh sách các cuộc hẹn khám bệnh **đã bị hủy** của người dùng. Dữ liệu được truyền từ ViewBag.list, do Controller DaBiHuy() cung cấp. Giao diện sử dụng layout dùng chung Bs_Layout.cshtml và hiển thị các cột thông tin như mã cuộc hẹn, tên người dùng, số điện thoại, trạng thái, thời gian hẹn và mô tả bệnh. Mục đích chính là để bác sĩ có thể theo dõi lại các cuộc hẹn không thành công, xác định nguyên nhân và hỗ trợ người bệnh đặt lại lịch khi cần thiết. Giao diện này đóng vai trò như một phần trong hệ thống quản lý lịch sử khám chữa bệnh.

```
@{
    ViewData["Title"] = "DaBiHuy";
    Layout = "~/Views/Shared/Bs_Layout.cshtml";
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<link rel="stylesheet" href="~/css/HenLichKham.css" asp-append-version="true" />
}

<h2> Đã bị hủy </h2>
<div class="container">
    <table class="table table-striped">
        <thead>
            <tr>
                <th>Mã Cuộc Hẹn</th>
                <th>Tên Người Dùng</th>
                <th>SĐT</th>
                <th>Trạng thái</th>
                <th>Thời Gian Hẹn</th>
                <th>Mô Tả Bệnh</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var appointment in ViewBag.list)
            {
                <tr>
                    <td>@appointment[0]</td>
                    <td>@appointment[2]</td>
                    <td>@appointment[3]</td>
                    <td>@appointment[7]</td>
                    <td>@appointment[8]</td>
                    <td>@appointment[15]</td>
                </tr>
            }
        </tbody>
    </table>
```

```
</div>
```

5.2.23. Chức năng [Cập nhật]

- **Controller** cập nhật cuộc hẹn: Phương thức Updatecuochen(string id, string matt) dùng để cập nhật trạng thái của một cuộc hẹn khám bệnh. Tham số id đại diện cho mã cuộc hẹn, còn matt là mã trạng thái cần cập nhật (ví dụ: đã xác nhận, hoàn thành, đã hủy...). Khi phương thức được gọi, nó thực thi stored procedure UpdateMaTTCH, truyền vào hai giá trị này để cập nhật trạng thái tương ứng trong cơ sở dữ liệu. Sau khi cập nhật xong, hệ thống tự động chuyển hướng về trang LichHenKham để người dùng xem danh sách cập nhật mới nhất. Chức năng này thường được bác sĩ hoặc hệ thống sử dụng để thay đổi tiến trình xử lý của một cuộc hẹn.

```
public ActionResult Updatecuochen(string id, string matt)
{
    // Lấy thời gian hiện tại
    db.get("Exec UpdateMaTTCH "+ id +"," + matt);
    return RedirectToAction("LichHenKham","Bacsis");
}
```

- **Controller UpdatecuochenTT**: Phương thức UpdatecuochenTT(string id, string matt) xử lý việc cập nhật trạng thái cuộc hẹn kèm theo cập nhật số dư tài khoản cho bệnh nhân và bác sĩ sau khi hoàn tất cuộc hẹn. Sau khi lấy userId từ phiên đăng nhập (Session), hàm đầu tiên gọi stored procedure UpdateMaTTCH để cập nhật trạng thái cuộc hẹn có mã id sang trạng thái mới matt. Tiếp theo, nó gọi tiếp UpdateSoDuTKForUserAndDoctor, truyền vào userId và id, nhằm thực hiện nghiệp vụ cộng/trừ số dư tài khoản dựa trên kết quả tư vấn. Sau khi hoàn thành cả hai bước, hệ thống chuyển hướng trở lại trang LichHenKham để hiển thị danh sách mới nhất. Đây là bước xử lý sau cùng trong một cuộc hẹn, đảm bảo cả trạng thái hệ thống và tài chính đều được cập nhật đồng bộ.

```
public ActionResult UpdatecuochenTT(string id,string matt)
```

```
{  
    var userId = _session.GetString("userId");  
    db.get("Exec UpdateMaTTCH "+ id +"," + matt);  
    db.get("Exec UpdateSoDuTKForUserAndDoctor "+ userId +"," + id);  
    return RedirectToAction("LichHenKham","Bacsis");  
}
```

- **Controller GetAllConfirmedAppointments:** Phương thức trả về danh sách tất cả các cuộc hẹn đã được xác nhận của người dùng hiện tại dưới dạng dữ liệu JSON. Sau khi lấy userId từ Session, phương thức gọi stored procedure “Xem tat ca lich hen da xac nhan” để truy xuất toàn bộ các cuộc hẹn đã được bác sĩ xác nhận. Kết quả được lưu trong biến list và trả về thông qua Json(list). Phương thức này thường được dùng trong các tình huống cần kết nối dữ liệu bằng Ajax hoặc API phía client-side, ví dụ như khi xây dựng giao diện lịch bằng JavaScript mà không cần reload toàn bộ trang.

```
public JsonResult GetAllConfirmedAppointments()  
{  
    var userId = _session.GetString("userId");  
    var list = db.get($"EXEC Xemtatcalichhendaxacnhan {userId}");  
    return Json(list);  
}
```

- **Controller GetAllCompletedAppointments:** Phương thức dùng để trả về danh sách các cuộc hẹn đã hoàn thành của người dùng hiện tại dưới dạng JSON. Sau khi lấy userId từ session, hàm thực hiện stored procedure “Xem tat ca lich hen da hoan thanh” để truy vấn các lịch hẹn đã được bác sĩ hoàn tất. Kết quả được lưu trong biến list và trả về bằng Json(list). Phương thức này rất hữu ích trong các giao diện hiện đại như dashboard, biểu đồ thống kê hoặc lịch điện tử, nơi dữ liệu cần được tải bất đồng bộ mà không cần tải lại toàn trang.

```
public JsonResult GetAllCompletedAppointments()  
{  
    var userId = _session.GetString("userId");  
}
```

```
var list = db.get($"EXEC Xemtatcalichhendahoanthanh {userId}");  
return Json(list);  
}
```

- **Controller GetAllCancelledAppointments:** Phương thức dùng để trả về danh sách cuộc hẹn đã hoàn thành của người dùng hiện tại dưới dạng JSON. Sau khi lấy userID từ session, hàm thực hiện stored procedure “xem tat ca lich hen da hoan thanh” để truy vấn các lịch hẹn đã được bác sĩ hoàn tất. Kết quả được lưu trong biến list và trả về bằng Json(list). Phương thức này rất hữu ích trong các giao diện hiện đại như dashboard, biểu đồ thống kê hoặc lịch điện tử, nơi dữ liệu cần được tải bất đồng bộ mà không cần tải lại toàn trang.

```
public JsonResult GetAllCancelledAppointments()  
{  
    var userId = _session.GetString("userId");  
    var list = db.get($"EXEC Xemtatcalichhendahuy {userId}");  
    return Json(list);  
}
```

- **Controller UpdateAppointmentStatus:** thực hiện cập nhật trạng thái của một cuộc hẹn từ phía giao diện người dùng thông qua gọi bất đồng bộ (AJAX). Khi nhận được id của cuộc hẹn và status mới, hàm gọi stored procedure UpdateMaTTCH để cập nhật thông tin trong cơ sở dữ liệu. Sau khi thực hiện xong, hàm trả về một đối tượng JSON chứa { success = true } để phía client biết rằng thao tác đã thành công. Phương thức này thường được dùng kết hợp với JavaScript để cập nhật trạng thái nhanh chóng trên giao diện mà không cần reload lại trang.

```
public JsonResult UpdateAppointmentStatus(int id, string status)  
{  
    db.get($"EXEC UpdateMaTTCH {id}, '{status}'");  
    return Json(new { success = true });  
}
```

5.2.24. Chức năng [Quản lý thông báo]

Controller: Hàm ThongBao() là một action trong ASP.NET MVC dùng để lấy danh sách thông báo của người dùng hiện tại. Đầu tiên, nó lấy userId từ session (tức là người đang đăng nhập). Sau đó, nó gọi stored procedure sp_GetThongBaoByMaND từ SQL Server thông qua phương thức db.get() để truy xuất các thông báo tương ứng với người dùng này. Kết quả được gán vào ViewBag.ThongBaos để truyền sang View và hiển thị. Cuối cùng, action trả về view tương ứng.

```
public ActionResult ThongBao()
{
    var userId = _session.GetString("userId");
    ViewBag.ThongBaos = db.get("EXEC sp_GetThongBaoByMaND " + userId );
    return View();
}
```

Controller: Đây là một API dạng GET trả về dữ liệu dưới dạng JSON, được dùng để lấy danh sách tất cả các lịch hẹn đã được xác nhận của người dùng hiện tại. Hàm truy xuất userId từ session để biết người dùng nào đang đăng nhập, sau đó gọi stored procedure Xemtatcalichhendaxacnhan với userId làm tham số để truy vấn dữ liệu từ cơ sở dữ liệu. Kết quả trả về là một danh sách (list) được gửi về client dưới dạng JsonResult, thường được sử dụng trong AJAX hoặc các frontend framework như React/Vue.

```
[HttpGet]
public JsonResult GetAllConfirmedAppointments()
{
    var userId = _session.GetString("userId");
    var list = db.get($"EXEC Xemtatcalichhendaxacnhan {userId}");
    return Json(list);
}
```

Controller: API GetNotifications() là một phương thức HttpGet dùng để lấy danh sách thông báo của người dùng hiện tại từ cơ sở dữ liệu. Phương thức này truy xuất userId từ session, sau đó thực hiện stored procedure sp_GetThongBaoByMaND để

lấy các thông báo tương ứng. Kết quả được trả về dưới dạng JSON thông qua Json(notifications), cho phép frontend (như jQuery, AJAX, hoặc các framework JS) dễ dàng xử lý và hiển thị thông báo theo thời gian thực hoặc khi người dùng cần.

```
[HttpGet]  
public JsonResult GetNotifications()  
{  
    var userId = _session.GetString("userId");  
    var notifications = db.get($"EXEC sp_GetThongBaoByMaND {userId}");  
    return Json(notifications);  
}
```

View: Giao diện này hiển thị danh sách các thông báo của người dùng dưới dạng bảng. View sử dụng biến ViewBag.ThongBaos (được truyền từ Controller) để hiển thị từng dòng thông báo, bao gồm tiêu đề và thời gian. Mỗi dòng có một nút "Xem Chi Tiết", khi nhấp vào sẽ hiển thị nội dung đầy đủ của thông báo trong một Modal popup được tạo sẵn trong HTML. CSS được dùng để cải thiện bố cục bảng và modal, giúp giao diện rõ ràng, dễ nhìn. Ngoài ra, JavaScript/jQuery xử lý sự kiện nhấp nút và hiển thị modal tương tác một cách mượt mà.

```
@{  
    ViewData["Title"] = "Thông Báo";  
    Layout = "~/Views/Shared/Bs_Layout.cshtml";  
}  
  
<style>  
/* Styling improvements for table */  
.table {  
    margin-top: 20px;  
    border-collapse: collapse;  
    width: 100%;  
}
```

```
.table th, .table td {  
    text-align: left;  
    padding: 12px;  
}  
  
.table th {  
    background-color: #007bff;  
    color: white;  
    font-weight: bold;  
    text-transform: uppercase;  
}  
  
.table tr:nth-child(even) {  
    background-color: #f2f2f2;  
}  
  
.table tr:hover {  
    background-color: #d9edf7;  
    cursor: pointer;  
}  
  
.btn-primary {  
    background-color: #007bff;  
    border: none;  
    font-size: 14px;  
    padding: 8px 12px;  
    border-radius: 4px;  
    transition: background-color 0.3s ease;  
}  
  
.btn-primary:hover {
```

```
background-color: #0056b3;  
color: #fff;  
}  
  
/* Styling for the modal */  
.modal {  
display: none; /* Initial hidden state */  
position: fixed; /* Fix the modal to the screen */  
top: 0;  
left: 0;  
width: 100%;  
height: 100%;  
z-index: 1050; /* Ensure modal is on top */  
background-color: rgba(0, 0, 0, 0.5); /* Dimmed background */  
justify-content: center;  
align-items: center;  
}  
  
.modal-content {  
padding: 20px;  
border-radius: 8px;  
background-color: white;  
max-width: 500px;  
width: 100%;  
}  
  
.modal-header {  
background-color: #007bff;  
color: white;  
font-size: 18px;  
padding: 10px 15px;
```

THIẾT KẾ CHƯƠNG TRÌNH

```
border-radius: 5px 5px 0 0;  
}  
  
.modal-footer {  
    padding: 10px;  
    text-align: center;  
}  
  
.close {  
    background: none;  
    border: none;  
    color: white;  
    font-size: 20px;  
    position: absolute;  
    top: 10px;  
    right: 15px;  
}  
  
</style>  
  
<div class="container mt-4">  
    <h2 class="text-center mb-4">Danh Sách Thông Báo</h2>  
    <table class="table table-hover">  
        <thead>  
            <tr>  
                <th>#</th>  
                <th>Tiêu Đề</th>  
                <th>Thời Gian</th>  
                <th>Thao Tác</th>  
            </tr>  
        </thead>  
        <tbody>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
<tbody>
    @if (ViewBag.ThongBaos != null && ViewBag.ThongBaos.Count > 0)
    {
        foreach (var tb in ViewBag.ThongBaos)
        {
            <tr>
                <td>@tb[0]</td>
                <td>@tb[1]</td>
                <td>@Convert.ToDateTime(tb[3]).ToString("yyyy-MM-dd
HH:mm:ss")</td>
                <td>
                    <button
                        class="btn btn-primary btn-sm btn-detail"
                        data-id="@tb[0]"
                        data-title="@tb[1]"
                        data-content="@tb[2]">
                        Xem Chi Tiết
                    </button>
                </td>
            </tr>
        }
    }
    else
    {
        <tr>
            <td colspan="4" class="text-center text-muted">Không có thông báo
nào.</td>
        </tr>
    }
</tbody>
</table>
```

THIẾT KẾ CHƯƠNG TRÌNH

```
</div>

<!-- Modal -->
<div class="modal" id="detailModal">
  <div class="modal-content">
    <div class="modal-header">
      <h5 class="modal-title" id="modalTitleLabel">Chi Tiết Thông Báo</h5>
      <button type="button" class="close" id="closeModal" aria-label="Close">
        &times;
      </button>
    </div>
    <div class="modal-body">
      <p><strong>Tiêu Đề:</strong> <span id="modalTitle"></span></p>
      <p><strong>Nội Dung:</strong></p>
      <p id="modalContent"></p>
    </div>
    <div class="modal-footer">
      <button type="button" class="btn btn-secondary" id="closeBtn">Đóng</button>
    </div>
  </div>
</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function () {
  // Gắn sự kiện cho nút "Xem Chi Tiết"
  $(document).on('click', '.btn-detail', function () {
    // Lấy dữ liệu từ các thuộc tính data
    const title = $(this).data('title');
    const content = $(this).data('content');
```

THIẾT KẾ CHƯƠNG TRÌNH

```
// Kiểm tra dữ liệu trước khi hiển thị
if (!title || !content) {
    alert("Thông tin thông báo không đầy đủ!");
    return;
}

// Hiển thị thông tin trong modal
$('#modalTitle').text(title);
$('#modalContent').text(content);

// Hiển thị modal (đưa modal vào chế độ hiển thị)
$('#detailModal').fadeIn();
});

// Đóng modal khi nhấn nút "Đóng" hoặc "X"
$('#closeModal, #closeBtn').click(function () {
    $('#detailModal').fadeOut();
});

// Đóng modal khi nhấn bên ngoài modal
$(window).click(function (event) {
    if ($(event.target).is('#detailModal')) {
        $('#detailModal').fadeOut();
    }
});
});

</script>
```

5.2.25. Chức năng [Thống kê]

Controller: Hàm ThongKe là một action trong controller dùng để hiển thị thông tin thống kê lịch hẹn theo tháng trong năm mà người dùng chọn. Hàm lấy userId từ session, sau đó gọi stored procedure sp_XemThongTinNguoiDung để lấy thông tin người dùng và ThongKeCuocHenTheoThang để lấy danh sách các cuộc hẹn theo tháng tương ứng với năm (nam) được truyền vào. Dữ liệu kết quả được gán vào ViewBag.luong và ViewBag.list, rồi trả về View HomeBs.cshtml để hiển thị. Action này hỗ trợ bác sĩ xem thống kê nhanh chóng theo năm.

```
public IActionResult ThongKe(string nam)
{
    var userId = _session.GetString("userId");
    ViewBag.luong = db.get("exec sp_XemThongTinNguoiDung "+userId);
    ViewBag.list = db.get("EXEC ThongKeCuocHenTheoThang " + userId +","+
nam );
    return View("HomeBs");
}
```

Controller Doanh Thu: Hàm DoanhThu() là một action đơn giản thuộc controller, có nhiệm vụ trả về view hiển thị doanh thu. Trong action này không thực hiện xử lý logic hay truy vấn dữ liệu từ cơ sở dữ liệu. Việc tính toán hoặc hiển thị doanh thu (nếu có) sẽ được thực hiện trong View tương ứng hoặc thông qua các API/JavaScript gọi động sau khi trang được render. Đây là action thường dùng để hiển thị trang thống kê doanh thu cho người dùng (ví dụ như bác sĩ hoặc quản trị viên).

```
Public ActionResult DoanhThu()
{
    return View();
}
```

View: View này chỉ thiết lập tiêu đề trang ("DoanhThu") và chỉ định layout chung Bs_Layout.cshtml, vốn chứa các thành phần giao diện tái sử dụng (header, sidebar, footer) cho bác sĩ. Bởi không có HTML hay logic render dữ liệu bên trong, trang này đóng vai trò như một “khung” rỗng: nội dung chi tiết về doanh thu sẽ được chèn động

sau, chẳng hạn bằng partial view, AJAX/JavaScript gọi API, hoặc trong các khối section được định nghĩa ở layout. Cách tách rời này giúp giữ layout gọn gàng, hỗ trợ tái sử dụng và cho phép nhóm phát triển bổ sung biểu đồ doanh thu, bảng thống kê hay bộ lọc thời gian mà không cần chỉnh sửa file layout hoặc controller.

```
@{  
    ViewData["Title"] = "DoanhThu";  
    Layout = "~/Views/Shared/Bs_Layout.cshtml";  
}
```

5.2.26. Chức năng [Quản lý tài khoản]

Phương thức `AdminLogin()` là một hành động trong Controller có nhiệm vụ trả về giao diện (View) chứa form đăng nhập. Khi người dùng truy cập trang này, hệ thống khởi tạo một đối tượng `DataModel` để chuẩn bị cho các truy vấn cơ sở dữ liệu sau này (nếu có), sau đó trả về View tương ứng.

```
public IActionResult AdminLogin()  
{  
    DataModel db = new DataModel();  
    return View();  
}
```

Phương thức này đơn giản chỉ trả về View, nơi người quản trị có thể nhập số điện thoại và mật khẩu để tiến hành đăng nhập.

Khi người dùng nhập số điện thoại và mật khẩu rồi nhấn đăng nhập, dữ liệu sẽ được gửi đến phương thức `XuLyAdminLogin()` thông qua phương thức POST. Hàm này tiếp nhận hai tham số: `sdt` (số điện thoại) và `password` (mật khẩu), sau đó sử dụng stored procedure có tên `CheckAdminLogin` để kiểm tra thông tin.

```
[HttpPost]  
public IActionResult XuLyAdminLogin(string sdt, string password)  
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    DataModel db = new DataModel();

    ViewBag.list = db.get("EXEC CheckAdminLogin '" + sdt + "', '" + password +
"';"); ["Message"] = "Thông báo đã được gửi thành công!";
}
```

Dòng trên cho thấy cách gọi stored procedure trong SQL Server thông qua một chuỗi truy vấn. Hàm `db.get(...)` sẽ thực hiện truy vấn và trả về danh sách kết quả (nếu có). Danh sách này được gán vào `ViewBag.list`.

Nếu danh sách `ViewBag.list` có ít nhất một bản ghi (nghĩa là thông tin đăng nhập là hợp lệ), hệ thống sẽ lưu trạng thái đăng nhập vào `TempData`, đồng thời chuyển hướng người dùng đến trang quản lý chính

```
if (ViewBag.list != null && ViewBag.list.Count > 0)
{
    TempData["AdminLoggedIn"] = true;
    TempData["Success"] = "Đăng nhập thành công!";
    return RedirectToAction("Index", "Admin");
}
```

Ngược lại, nếu không có dữ liệu nào được trả về từ truy vấn, hệ thống hiểu rằng thông tin đăng nhập không hợp lệ. Khi đó, hệ thống sẽ lưu thông báo lỗi vào `TempData["Error"]` và chuyển hướng ngược lại trang đăng nhập để người dùng thử lại.

5.2.27. Chức năng [Quản lý bài viết]

Chức năng quản lý bài viết giúp admin thực hiện các thao tác thêm, sửa, xóa, tìm kiếm và hiển thị chi tiết bài viết trong hệ thống. Toàn bộ các thao tác đều sử dụng stored procedure hoặc truy vấn SQL trực tiếp, đồng thời hỗ trợ người dùng thông qua thông báo (`TempData`) và hiển thị danh sách dữ liệu bằng `ViewBag`.

Phương thức `DMBaiViet()` có nhiệm vụ lấy danh sách các loại bài viết (`LOAIBAIVIET`), danh sách bài viết (`BAIVIET`) và danh sách khoa bệnh (`KHOABENH`) từ cơ sở dữ liệu. Dữ liệu được lưu vào `ViewBag` để hiển thị trong View tương ứng.

```
public IActionResult DMBaiViet()
{
}
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    DataModel db = new DataModel();
    ViewBag.listLBV = db.get("SELECT * from LOAIBAIVIET");
    ViewBag.listBV = db.get("SELECT * from BAIVIET");
    ViewBag.listKB = db.get("SELECT * from KHOABENH");
    return View();
}
```

Phương thức `ThemBaiViet()` xử lý việc thêm mới bài viết. Hệ thống thực hiện kiểm tra các trường bắt buộc như tiêu đề, nội dung và mã loại bài viết. Nếu thiếu, sẽ hiển thị thông báo lỗi. Ngược lại, thông tin hợp lệ sẽ được xử lý và gửi đến stored procedure `AddBAIVIET` để lưu vào cơ sở dữ liệu.

```
[HttpPost]
public IActionResult ThemBaiViet(string tieudebv, string noidungbv, string makb,
    string linkytb, string malbv)
{
    try
    {
        if (string.IsNullOrEmpty(tieudebv) || string.IsNullOrEmpty(noidungbv)
            || string.IsNullOrEmpty(malbv))
        {
            TempData["Error"] = "Vui lòng điền đầy đủ thông tin bắt buộc";
            return RedirectToAction("DMBaiViet");
        }

        DataModel db = new DataModel();
        DateTime ngaydang = DateTime.Now;

        string maKhoaBenhParam = string.IsNullOrEmpty(makb) || makb == "NULL" ? "NULL" : makb;
        string linkYoutubeParam = string.IsNullOrEmpty(linkytb) ? "NULL" : $"'{linkytb}'";

        tieudebv = tieudebv.Replace("'", "''");
        noidungbv = noidungbv.Replace("'", "''");
```

THIẾT KẾ CHƯƠNG TRÌNH

```
string query = $@"EXEC AddBAIVIET  
    @TieuDeBV = N'{tieudebv}',  
    @NoiDung = N'{noidungbv}',  
    @MaKB = {maKhoaBenhParam},  
    @LinkYtb = {linkYoutubeParam},  
    @NgayDang = '{ngaydang:yyyy-MM-dd}',  
    @MaLBV = {malbv},  
    @LuotXem = 0";  
  
ViewBag.list = db.get(query);  
return RedirectToAction("DMBaiViet", "Admin");  
}  
catch (Exception)  
{  
    return RedirectToAction("DMBaiViet", "Admin");  
}
```

Phương thức XoaBaiViet() nhận vào id bài viết và thực hiện gọi stored procedure DeleteBAIVIET để xóa bài viết khỏi cơ sở dữ liệu.

```
[HttpPost]  
public IActionResult XoaBaiViet(string id)  
{  
    DataModel db = new DataModel();  
    ViewBag.list = db.get("EXEC DeleteBAIVIET " + id + ";");  
    return RedirectToAction("DMBaiViet", "Admin");  
}
```

Phương thức TimBaiViet() có chức năng lấy chi tiết bài viết theo MaBV (mã bài viết) để hiển thị lên form chỉnh sửa. Đồng thời, phương thức cũng lấy danh sách khoa bệnh và loại bài viết để điền vào các dropdown.

```
public IActionResult TimBaiViet(string mabv)  
{  
    try
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    DataModel db = new DataModel();  
    ViewBag.list = db.get($"SELECT * FROM BAIVIET WHERE MaBV = {mabv}");  
    ViewBag.listKB = db.get("SELECT * FROM KHOABENH");  
    ViewBag.listLBV = db.get("SELECT * FROM LOAIBAIVIET");  
    return View();  
}  
catch (Exception)  
{  
    TempData["Error"] = "Có lỗi xảy ra khi tìm bài viết";  
    return RedirectToAction("DMBaiViet");  
}  
}
```

Phương thức `ChiTietBaiViet()` dùng để hiển thị đầy đủ thông tin một bài viết theo mã. Đồng thời, nó lấy thêm danh sách loại bài viết và khoa bệnh để bổ sung thông tin liên quan.

```
public IActionResult ChiTietBaiViet(string mabv)  
{  
    DataModel db = new DataModel();  
    ViewBag.list = db.get("EXEC FindBAIVIET_ID " +mabv+ ";");  
    ViewBag.listLBV = db.get("SELECT * from LOAIBAIVIET");  
    ViewBag.listKB = db.get("SELECT * from KHOABENH");  
    return View();  
}
```

Phương thức `SuaBaiViet()` xử lý logic cập nhật bài viết. Trước khi thực hiện, phương thức kiểm tra dữ liệu đầu vào và loại bỏ các ký tự gây lỗi trong câu lệnh SQL (escaping '). Sau đó, dữ liệu được gửi đến stored procedure `UpdateBAIVIET`.

```
[HttpPost]  
public IActionResult SuaBaiViet(string mabv, string tieudebv, string noidungbv,  
    string makb, string linkytb, string malbv, string luotxem)  
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
try
{
    if (string.IsNullOrEmpty(mabv) || string.IsNullOrEmpty(tieudebv)
        || string.IsNullOrEmpty(noidungbv) || string.IsNullOrEmpty(malbv))
    {
        TempData["Error"] = "Vui lòng điền đầy đủ thông tin bắt buộc";
        return RedirectToAction("TimBaiViet", new { mabv = mabv });
    }

    DataModel db = new DataModel();
    DateTime ngaydang = DateTime.Now;

    string maKhoaBenhParam = string.IsNullOrEmpty(makb) ? "NULL" : makb;
    string linkYoutubeParam = string.IsNullOrEmpty(linkytb) ? "NULL" :
$""{linkytb.Replace("", "")}";
    tieudebv = tieudebv.Replace("", "");
    noidungbv = noidungbv.Replace("", "");

    string query = $@"EXEC UpdateBAIVIET
        @MaBV = {mabv},
        @TieuDeBV = N'{tieudebv}',
        @NoiDung = N'{noidungbv}',
        @MaKB = {maKhoaBenhParam},
        @LinkYtb = {linkYoutubeParam},
        @NgayDang = '{ngaydang:yyyy-MM-dd}',
        @MaLBV = {malbv},
        @LuotXem = {luotxem}";

    ViewBag.list = db.get(query);

    TempData["Success"] = "Cập nhật bài viết thành công";
    return RedirectToAction("DMBaiViet");
}
```

```
    catch (Exception)
    {
        TempData["Error"] = "Có lỗi xảy ra khi cập nhật bài viết";
        return RedirectToAction("TimBaiViet", new { mabv = mabv });
    }
}
```

5.2.28. Chức năng [Quản lý chuyên ngành]

Trong hệ thống quản trị, quản lý chuyên ngành là một trong những chức năng quan trọng mà Admin cần đảm trách. Toàn bộ quy trình này được thực hiện trong controller thông qua các hành động như hiển thị danh sách chuyên ngành, thêm mới, tìm kiếm, chỉnh sửa và xóa chuyên ngành. Bên dưới là phần giải thích chi tiết từng chức năng, lồng ghép theo từng đoạn mã cụ thể.

```
public IActionResult DMChuyenNganh()
{
    DataModel db = new DataModel();
    ViewBag.listCN = db.get("SELECT * from CHUYENNGANH");
    return View();
}
```

Phương thức DMChuyenNganh() chịu trách nhiệm hiển thị danh sách các chuyên ngành trong hệ thống. Đầu tiên, một đối tượng DataModel được khởi tạo để phục vụ cho việc truy xuất dữ liệu từ cơ sở dữ liệu. Sau đó, phương thức gọi lệnh SQL SELECT * FROM CHUYENNGANH thông qua hàm db.get() để lấy tất cả các chuyên ngành có trong bảng CHUYENNGANH. Kết quả trả về được lưu vào ViewBag.listCN, cho phép truyền dữ liệu sang View để hiển thị danh sách chuyên ngành trên giao diện người dùng. Cuối cùng, phương thức trả về view tương ứng.

```
[HttpPost]
public IActionResult ThemChuyenNganh(string tencn, IFormFile hinhanh)
{
    if (hinhanh == null || hinhanh.Length == 0)
    {
        TempData["Error"] = "Vui lòng tải lên hình ảnh hợp lệ!";
    }
}
```

THIẾT KẾ CHƯƠNG TRÌNH

```
        return RedirectToAction("DMChuyenNganh", "Admin");  
    }
```

Phương thức `ThemChuyenNganh()` là một action POST được dùng để xử lý dữ liệu khi Admin thêm mới một chuyên ngành. Hai tham số đầu vào là `tencn` (tên chuyên ngành) và `hinhanh` (file hình ảnh đại diện). Ở đầu phương thức, có một điều kiện kiểm tra tính hợp lệ của ảnh tải lên. Nếu ảnh không tồn tại hoặc rỗng, hệ thống sẽ thông báo lỗi thông qua `TempData["Error"]` và quay lại trang danh mục chuyên ngành.

```
try  
{  
    DataModel db = new DataModel();  
    string namefile = Path.GetFileName(hinhanh.FileName);
```

Trong khối `try`, hệ thống tiếp tục khởi tạo `DataModel` để tương tác với cơ sở dữ liệu. Tiếp đó, dùng `Path.GetFileName()` để lấy tên file ảnh từ đối tượng `hinhanh`.

```
string uploadsFolder = Path.Combine(Directory.GetCurrentDirectory(), "wwwroot",  
"Uploads");  
    Directory.CreateDirectory(uploadsFolder);  
    string filePath = Path.Combine(uploadsFolder, namefile);
```

Tại đây, hệ thống xác định đường dẫn thư mục để lưu file ảnh, cụ thể là trong thư mục `wwwroot/Uploads`. Nếu thư mục chưa tồn tại, dòng `Directory.CreateDirectory(uploadsFolder)` sẽ tự động tạo mới để đảm bảo không xảy ra lỗi khi lưu file.

```
using (var stream = new FileStream(filePath, FileMode.Create))  
{  
    hinhanh.CopyTo(stream);  
}
```

Sau khi xác định xong đường dẫn, hệ thống tiến hành ghi file bằng cách tạo luồng `FileStream` để ghi dữ liệu ảnh vào thư mục đích.

```
db.get($"EXEC sp_ThemChuyenNganh N'{tencn}', '{namefile}';");
```

THIẾT KẾ CHƯƠNG TRÌNH

```
        TempData["Message"] = "Thêm chuyên ngành thành công!";
    }
    catch (Exception ex)
    {
        TempData["Error"] = $"Có lỗi xảy ra: {ex.Message}";
    }

    return RedirectToAction("DMChuyenNganh", "Admin");
}
```

Sau khi ảnh được lưu, hệ thống gọi stored procedure `sp_ThemChuyenNganh` để chèn bản ghi mới vào cơ sở dữ liệu với các tham số gồm tên chuyên ngành và tên file ảnh. Nếu có lỗi xảy ra trong quá trình thêm, hệ thống sẽ hiển thị thông báo lỗi chi tiết. Dù thành công hay thất bại, phương thức đều điều hướng người dùng về trang danh sách chuyên ngành.

```
public IActionResult TimChuyenNganh(string macn)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC sp_TimChuyenNganhTheoMa " + macn + ";");
    return View();
}
```

Phương thức `TimChuyenNganh()` cho phép Admin tìm kiếm chuyên ngành theo mã (`macn`). Sau khi khởi tạo `DataModel`, phương thức thực hiện truy vấn bằng stored procedure `sp_TimChuyenNganhTheoMa`, truyền vào mã chuyên ngành để lọc dữ liệu. Kết quả trả về được lưu vào `ViewBag.list` để hiển thị trong View tương ứng.

```
[HttpPost]
public IActionResult SuaChuyenNganh(string macn, string tench, IFormFile
hinhanh)
{
    if (hinhanh == null || hinhanh.Length == 0)
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    TempData["Error"] = "Vui lòng tải lên hình ảnh hợp lệ!";  
    return RedirectToAction("DMChuyenNganh", "Admin");  
}
```

Phương thức `SuaChuyenNganh()` cho phép chỉnh sửa thông tin chuyên ngành dựa vào mã chuyên ngành (`macn`). Hệ thống kiểm tra tính hợp lệ của ảnh tương tự như khi thêm mới. Nếu ảnh không hợp lệ, người dùng sẽ nhận thông báo và được điều hướng trở lại.

```
try  
{  
    DataModel db = new DataModel();  
    string namefile = Path.GetFileName(hinhanh.FileName);  
    string uploadsFolder = Path.Combine(Directory.GetCurrentDirectory(),  
    "wwwroot", "Uploads");  
    Directory.CreateDirectory(uploadsFolder);  
    string filePath = Path.Combine(uploadsFolder, namefile);  
  
    using (var stream = new FileStream(filePath, FileMode.Create))  
    {  
        hinhanh.CopyTo(stream);  
    }  
  
    db.get($"EXEC sp_SuaChuyenNganh {macn}, N'{tencn}', N'{namefile}';");  
    TempData["Message"] = "Sửa chuyên ngành thành công!";  
}  
catch (Exception ex)  
{  
    TempData["Error"] = $"Có lỗi xảy ra: {ex.Message}";  
}  
  
return RedirectToAction("DMChuyenNganh", "Admin");
```

THIẾT KẾ CHƯƠNG TRÌNH

}

Sau khi lưu ảnh, hệ thống gọi thủ tục `sp_SuaChuyenNganh` với ba tham số: mã chuyên ngành, tên chuyên ngành mới, và tên file ảnh mới. Đây là quy trình cập nhật lại toàn bộ thông tin chuyên ngành. Nếu có lỗi, thông báo sẽ được hiển thị lên màn hình. Việc cập nhật được xử lý trọn vẹn trong một khối `try-catch` để tránh làm gián đoạn luồng hoạt động của hệ thống.

```
[HttpPost]
public IActionResult XoaChuyenNganh(string macn)
{
    DataModel db = new DataModel();
    try
    {
        db.get("EXEC sp_XoaChuyenNganh " + macn + ";");
        TempData["Message"] = "Xóa chuyên ngành thành công!";
    }
    catch (Exception)
    {
        TempData["Error"] = "Chuyên ngành được chọn có liên kết với bác sĩ. Không
xóa thành công !!!!";
    }
    return RedirectToAction("DMChuyenNganh", "Admin");
}
```

Cuối cùng, phương thức `XoaChuyenNganh()` hỗ trợ Admin xóa một chuyên ngành khỏi hệ thống. Mã chuyên ngành được truyền vào, sau đó gọi stored procedure `sp_XoaChuyenNganh` để thực hiện việc xóa trên cơ sở dữ liệu. Tuy nhiên, nếu chuyên ngành này có liên kết với bác sĩ (khóa ngoại), thì việc xóa sẽ thất bại và lỗi được bắt thông qua khối `catch`. Hệ thống lúc này sẽ hiển thị thông báo cảnh báo để người dùng biết rằng chuyên ngành không thể xóa vì đang liên kết với dữ liệu khác.

5.2.29. Chức năng[Quản lý khoa bệnh]

Trong hệ thống quản lý bệnh viện, chức năng quản lý thông tin khoa bệnh là một phần quan trọng thuộc quyền hạn của Admin. Toàn bộ logic xử lý nghiệp vụ của

THIẾT KẾ CHƯƠNG TRÌNH

chức năng này được hiện thực trong controller thông qua các phương thức được khai báo trong lớp AdminController. Các phương thức này đảm nhiệm việc hiển thị danh sách khoa bệnh, thêm mới, chỉnh sửa, xóa và tìm kiếm thông tin khoa bệnh. Dưới đây là phân tích chi tiết từng phần chức năng.

Đầu tiên là phương thức DMKhoaBenz, chịu trách nhiệm lấy toàn bộ danh sách các khoa bệnh từ cơ sở dữ liệu và truyền sang View để hiển thị:

```
public IActionResult DMKhoaBenz()
{
    DataModel db = new DataModel();
    ViewBag.listKB = db.get("SELECT * from KHOABENH");
    return View();
}
```

Trong đoạn code trên, một đối tượng DataModel được khởi tạo để thực hiện truy vấn dữ liệu. Phương thức get sẽ thực hiện câu lệnh SQL "SELECT * from KHOABENH" nhằm lấy tất cả các dòng dữ liệu từ bảng KHOABENH. Kết quả truy vấn được lưu vào ViewBag.listKB để truyền dữ liệu sang giao diện hiển thị danh sách các khoa bệnh. Tiếp theo, phương thức ThemKhoaBenz đảm nhiệm vai trò thêm mới một khoa bệnh vào hệ thống. Phương thức này được đánh dấu [HttpPost], cho thấy nó sẽ được gọi khi người dùng thực hiện hành động gửi biểu mẫu (form submit).

```
[HttpPost]
public IActionResult ThemKhoaBenz(string tenkb, string mota)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC sp_ThemKhoaBenz N'" + tenkb + "", N'" + mota +
"';");
    return RedirectToAction("DMKhoaBenz", "Admin");
}
```

Khi người dùng điền thông tin tên khoa bệnh (tenkb) và mô tả (mota) rồi bấm “Thêm”, hệ thống sẽ thực thi stored procedure sp_ThemKhoaBenz, đồng thời truyền tham số đầu vào là tên và mô tả khoa bệnh. Sau khi thực hiện thêm thành công,

THIẾT KẾ CHƯƠNG TRÌNH

phương thức sẽ chuyển hướng người dùng về trang DMKhoaBenh để cập nhật lại danh sách khoa bệnh.

Trong trường hợp cần xóa một khoa bệnh, hệ thống sử dụng phương thức XoaKhoaBenh. Việc xóa một bản ghi có thể phát sinh lỗi do ràng buộc khóa ngoại trong cơ sở dữ liệu, do đó phương thức này được xử lý với try-catch để đảm bảo tính ổn định cho chương trình.

```
[HttpPost]
public IActionResult XoaKhoaBenh(string id)
{
    DataModel db = new DataModel();
    try
    {
        db.get("EXEC sp_XoaKhoaBenh " + id + ";");
        TempData["Message"] = "Xóa khoa bệnh thành công!";
    }
    catch (Exception)
    {
        TempData["Error"] = "Khoa bệnh được chọn có liên kết dữ liệu. Không xóa
thành công !!!!";
    }
    return RedirectToAction("DMKhoaBenh", "Admin");
}
```

Nếu khoa bệnh được xóa thành công, một thông báo thành công sẽ được lưu vào TempData["Message"]. Nếu có lỗi phát sinh (ví dụ khoa bệnh đó còn liên kết với bảng khác như bác sĩ, bệnh nhân, v.v.), thì thông báo lỗi sẽ được đưa ra để cảnh báo người dùng.

Ngoài ra, chức năng tìm kiếm khoa bệnh theo mã cũng được hỗ trợ thông qua phương thức TimKhoaBenh. Phương thức này thực hiện truy vấn theo mã khoa bệnh (makb) do người dùng nhập vào.

```
public IActionResult TimKhoaBenh(string makb)
{
}
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC sp_TimKhoaBenhTheoMa " + makb + ";");
    return View();
}
```

Stored procedure `sp_TimKhoaBenhTheoMa` được gọi để lọc ra các dòng dữ liệu phù hợp với mã khoa bệnh đã nhập. Kết quả tìm kiếm được truyền sang View thông qua `ViewBag.list`.

Cuối cùng là phương thức `SuaKhoaBenh`, thực hiện cập nhật thông tin cho một khoa bệnh cụ thể.

[HttpPost]

```
public IActionResult SuaKhoaBenh(string makb, string tenkb, string mota)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC sp_SuaKhoaBenh " + makb + ",N'" + tenkb + "'",
    N'" + mota + "'");
    return RedirectToAction("DMKhoaBenh", "Admin");
}
```

Khi người dùng nhập các thông tin cần sửa và gửi biểu mẫu, hệ thống sẽ gọi stored procedure `sp_SuaKhoaBenh` và truyền vào ba tham số: mã khoa bệnh (`makb`), tên mới (`tenkb`) và mô tả mới (`mota`). Sau khi cập nhật thành công, trang web sẽ tự động quay về trang danh sách để người dùng kiểm tra kết quả.

Tóm lại, toàn bộ chức năng "Quản lý khoa bệnh" trong phần Controller của Admin được xây dựng dựa trên nguyên lý tách biệt logic xử lý dữ liệu (truy vấn database thông qua stored procedure) và hiển thị (View). Cách tiếp cận này không chỉ giúp mã nguồn dễ bảo trì, mà còn đảm bảo an toàn dữ liệu và hiệu quả trong quá trình quản lý.

5.2.30. Chức năng [Quản lý bệnh viện]

Trong hệ thống quản lý thông tin y tế, chức năng quản lý bệnh viện đóng vai trò trung tâm, cho phép người quản trị (admin) thực hiện các thao tác CRUD – tức là tạo, đọc, cập nhật và xóa thông tin về bệnh viện. Toàn bộ các chức năng này được triển khai

THIẾT KẾ CHƯƠNG TRÌNH

trong controller thông qua một loạt các action method, tương tác trực tiếp với cơ sở dữ liệu bằng cách gọi các stored procedure được viết sẵn, đảm bảo hiệu năng và tính bảo mật.

Phương thức đầu tiên, `DMBenhVien`, đảm nhận nhiệm vụ hiển thị danh sách tất cả các bệnh viện đang được lưu trữ trong cơ sở dữ liệu:

```
public IActionResult DMBenhVien()
```

```
{  
    DataModel db = new DataModel();  
    ViewBag.list = db.get("SELECT * from BENHVIEN");  
    return View();  
}
```

Tại đây, lớp `DataModel` là một lớp trung gian giúp thực thi câu lệnh SQL `SELECT * from BENHVIEN`, nhằm truy vấn toàn bộ bảng `BENHVIEN`. Dữ liệu sau đó được lưu vào `ViewBag.list`, để hiển thị lên giao diện người dùng. Đây là bước cơ bản để người quản trị có thể xem được danh sách các bệnh viện hiện có trước khi thực hiện thao tác chỉnh sửa hay xóa.

Khi cần thêm mới bệnh viện, phương thức `ThemBenhVien` sẽ được gọi sau khi người dùng gửi form. Phương thức này yêu cầu người dùng nhập tên bệnh viện và upload một hình ảnh đại diện. Đoạn xử lý như sau:

```
[HttpPost]  
public IActionResult ThemBenhVien(string tenbv, IFormFile hinhanh)  
{  
    if (hinhanh == null || hinhanh.Length == 0)  
    {  
        TempData["Error"] = "Vui lòng tải lên hình ảnh hợp lệ!";  
        return RedirectToAction("DMBenhVien", "Admin");  
    }  
  
    try  
    {  
        DataModel db = new DataModel();  
        // Logic insert data into BENHVIEN table  
    }  
}
```

```
string namefile = Path.GetFileName(hinhanh.FileName);
string uploadsFolder = Path.Combine(Directory.GetCurrentDirectory(),
"wwwroot", "Uploads");
Directory.CreateDirectory(uploadsFolder);
string filePath = Path.Combine(uploadsFolder, namefile);

using (var stream = new FileStream(filePath, FileMode.Create))
{
    hinhanh.CopyTo(stream);
}

db.get($"EXEC AddBenhVien N'{tenbv}', '{namefile}');");
 TempData["Message"] = "Thêm bệnh viện thành công!";
}

catch (Exception ex)
{
    TempData["Error"] = $"Có lỗi xảy ra: {ex.Message}";
}

return RedirectToAction("DMBenhVien", "Admin");
}
```

Toàn bộ thao tác trong phương thức này có thể chia làm hai phần chính: kiểm tra tính hợp lệ của file hình ảnh và xử lý lưu trữ, sau đó là gọi stored procedure `AddBenhVien`. Đầu tiên, file hình ảnh được kiểm tra về mặt tồn tại và dung lượng. Nếu file không hợp lệ, hệ thống trả về thông báo lỗi. Nếu hợp lệ, tên tệp được lấy bằng `Path.GetFileName`, sau đó được lưu vào thư mục `wwwroot/Uploads` bằng luồng `FileStream`. Cuối cùng, thông tin bệnh viện bao gồm tên (`tenbv`) và tên tệp ảnh (`namefile`) được truyền vào stored procedure để thêm vào cơ sở dữ liệu. `TempData` dùng để lưu thông báo thành công hoặc lỗi và sẽ hiển thị ở View sau khi redirect.

THIẾT KẾ CHƯƠNG TRÌNH

Với chức năng tìm kiếm bệnh viện theo mã, hệ thống cung cấp phương thức `TimBenhVien`, cho phép lọc ra một bản ghi cụ thể trong bảng `BENHVIEN` dựa vào `mabv` (mã bệnh viện) người dùng nhập vào:

```
public IActionResult TimBenhVien(string mabv)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC sp_TimBenhVienTheoMa " + mabv + ";");
    return View();
}
```

Ở đây, hệ thống gọi stored procedure `sp_TimBenhVienTheoMa` để tìm kiếm, sau đó hiển thị kết quả trong View thông qua `ViewBag.list`. Đây là một ví dụ điển hình của việc tách biệt logic xử lý và giao diện hiển thị theo mô hình MVC.

Tiếp đến là chức năng sửa thông tin bệnh viện, được triển khai trong phương thức `SuaBenhVien`. Tương tự như thêm mới, chức năng này yêu cầu người dùng nhập mã bệnh viện (`mabv`), tên mới (`tenbv`) và upload hình ảnh mới. Nếu hình ảnh không hợp lệ, hệ thống sẽ từ chối xử lý. Quá trình lưu trữ file cũng giống như trong chức năng thêm mới:

```
[HttpPost]
public IActionResult SuaBenhVien(string mabv, string tenbv, IFormFile hinhanh)
{
    if (hinhanh == null || hinhanh.Length == 0)
    {
        TempData["Error"] = "Vui lòng tải lên hình ảnh hợp lệ!";
        return RedirectToAction("DMBenhVien", "Admin");
    }

    try
    {
        DataModel db = new DataModel();
        string namefile = Path.GetFileName(hinhanh.FileName);
```

THIẾT KẾ CHƯƠNG TRÌNH

```
string uploadsFolder = Path.Combine(Directory.GetCurrentDirectory(),
"wwwroot", "Uploads");
Directory.CreateDirectory(uploadsFolder);
string filePath = Path.Combine(uploadsFolder, namefile);

using (var stream = new FileStream(filePath, FileMode.Create))
{
    hinhanh.CopyTo(stream);
}

db.get($"EXEC sp_SuaBenhVien {mabv}, N'{tenbv}', N'{namefile}');");
 TempData["Message"] = "Sửa bệnh viện thành công!";
}

catch (Exception ex)
{
    TempData["Error"] = $"Có lỗi xảy ra: {ex.Message}";
}

return RedirectToAction("DMBenhVien", "Admin");
}
```

Phần stored procedure `sp_SuaBenhVien` sẽ cập nhật tên và hình ảnh bệnh viện dựa trên mã bệnh viện truyền vào. Toàn bộ các thao tác đều được bảo vệ bằng khôi `try-catch`, giúp chương trình không bị gián đoạn nếu xảy ra lỗi.

Cuối cùng là chức năng xóa bệnh viện, được thực hiện qua phương thức `xoaBenhVien`. Đây là một phương thức đơn giản nhưng cũng được xử lý lỗi cẩn thận để đảm bảo tính toàn vẹn dữ liệu:

```
[HttpPost]
public IActionResult XoaBenhVien(string mabv)
{
    DataModel db = new DataModel();
    try
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    db.get("EXEC sp_XoaBenhVien " + mabv + ";");  
    TempData["Message"] = "Xóa bệnh viện thành công!";  
}  
catch (Exception)  
{  
    TempData["Error"] = "Bệnh viện được chọn không xóa được!!!!";  
}  
return RedirectToAction("DMBenhVien", "Admin");  
}
```

Ở đây, nếu bệnh viện đang liên kết với dữ liệu khác trong hệ thống (chẳng hạn như bác sĩ, chuyên ngành...), cơ sở dữ liệu sẽ từ chối xóa do ràng buộc khóa ngoại. Trong trường hợp đó, hệ thống sẽ hiển thị thông báo lỗi cụ thể thông qua TempData["Error"], giúp người dùng hiểu lý do thao tác thất bại.

5.2.31. Chức năng [Quản lý bệnh nhân]

Trong hệ thống quản lý bệnh viện, **Admin** có quyền truy xuất, tìm kiếm, xem chi tiết và xóa thông tin bệnh nhân. Các chức năng này được hiện thực hóa trong Controller thông qua những phương thức sau đây: HoSoBenhNhan, ChiTietBenhNhan, SreachBenhNhan, và XoaBenhNhan.

```
public IActionResult HoSoBenhNhan()
```

```
{  
    DataModel db = new DataModel();  
    ViewBag.list = db.get("EXEC GetPatientsWithoutDoctor");  
    return View();  
}
```

Phương thức HoSoBenhNhan được dùng để hiển thị danh sách các bệnh nhân hiện có trong hệ thống, cụ thể là những bệnh nhân chưa có bác sĩ điều trị. Đầu tiên, một đối tượng DataModel được khởi tạo nhằm thực hiện việc tương tác với cơ sở dữ liệu. Sau đó, phương thức get() được gọi với câu lệnh SQL EXEC GetPatientsWithoutDoctor,

THIẾT KẾ CHƯƠNG TRÌNH

thực thi stored procedure cùng tên. Kết quả trả về là một danh sách các bệnh nhân được lưu vào `ViewBag.list` để truyền xuống view và hiển thị.

```
public IActionResult ChiTietBenhNhan(string id)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC GetPatientDetailsByUser_Id " + id + ";");
    return View();
}
```

Phương thức `ChiTietBenhNhan` cho phép Admin truy xuất chi tiết hồ sơ của một bệnh nhân cụ thể, thông qua mã định danh (`id`) của bệnh nhân đó. Cũng tương tự như trên, đối tượng `DataModel` được sử dụng để gửi truy vấn đến cơ sở dữ liệu. Truy vấn được gọi là `EXEC GetPatientDetailsByUser_Id` cộng chuỗi với `id`, tức là truyền tham số động vào stored procedure để lấy thông tin chi tiết. Dữ liệu thu được được đưa vào `ViewBag.list` để phục vụ việc hiển thị chi tiết hồ sơ trên giao diện người dùng.

Lưu ý: việc nối chuỗi trực tiếp vào câu truy vấn SQL như trên có thể gây nguy cơ SQL Injection. Trong thực tế, nên dùng các tham số SQL để bảo mật hơn.

```
public IActionResult SreachBenhNhan(string tenbn)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC SearchPatient N'" + tenbn + "'"); 
    return View();
}
```

Phương thức `SreachBenhNhan` thực hiện chức năng tìm kiếm bệnh nhân theo tên. Khi người dùng nhập tên bệnh nhân cần tìm, chuỗi ký tự đó sẽ được truyền vào phương thức. Stored procedure `SearchPatient` sẽ được thực thi với tham số là tên bệnh nhân (`tenbn`). Từ khóa `N'...'` được sử dụng để đảm bảo truy vấn có thể xử lý các ký tự Unicode (ví dụ: tiếng Việt). Kết quả sẽ được đẩy về `ViewBag.list` để hiển thị kết quả tìm kiếm cho Admin.

```
[HttpPost]
```

```
public IActionResult XoaBenhNhan(string mabn)
{
    DataModel db = new DataModel();
    try
    {
        // Gọi stored procedure để xóa bệnh nhân theo mã
        db.get("EXEC DeletePatientById " + mabn + ";");
        TempData["Message"] = "Xóa bệnh nhân thành công!";
    }
    catch (Exception)
    {
        // Nếu xảy ra lỗi (do khóa ngoại), thông báo không thể xóa
        TempData["Error"] = "Bệnh nhân được chọn không xóa được!!!!";
    }
    return RedirectToAction("HoSoBenhNhan", "Admin");
}
```

Đây là phương thức xử lý chức năng xóa thông tin bệnh nhân khỏi hệ thống. Dữ liệu được truyền vào là mã bệnh nhân (mabn). Phương thức sử dụng try-catch để bắt lỗi khi thực thi truy vấn xóa:

- Nếu stored procedure DeletePatientById thực hiện thành công, hệ thống thông báo thành công bằng TempData["Message"].
- Nếu xảy ra lỗi, thường là do ràng buộc khóa ngoại (foreign key) – ví dụ, bệnh nhân còn lịch hẹn hoặc hồ sơ khám – thì hệ thống sẽ báo lỗi và không cho xóa.

Sau khi xử lý, Admin được điều hướng về lại trang danh sách bệnh nhân bằng lệnh RedirectToAction("HoSoBenhNhan", "Admin").

5.2.32. Chức năng [Quản lý bác sĩ]

Trong hệ thống quản lý, chức năng dành cho Admin đóng vai trò trọng tâm trong việc theo dõi, xử lý và duyệt các thông tin liên quan đến đội ngũ bác sĩ. Mỗi hành động đều được định nghĩa rõ ràng trong controller dưới dạng các phương thức, sử dụng mô hình MVC kết hợp với stored procedure từ SQL Server để truy xuất và thao tác dữ liệu. Dưới đây là phần giải thích cụ thể từng chức năng.

THIẾT KẾ CHƯƠNG TRÌNH

Phương thức HoSoBacSi đảm nhận vai trò hiển thị danh sách các bác sĩ trong hệ thống, dựa theo mã phân quyền:

```
public IActionResult HoSoBacSi(string mapq)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC GetDoctorDetailsByRoleId " +3+ ";");
    return View();
}
```

Tuy tham số mapq được truyền vào nhưng ở đây hàm gọi trực tiếp mã quyền là 3, tương ứng trung cho quyền của bác sĩ trong hệ thống. DataModel là lớp hỗ trợ kết nối cơ sở dữ liệu, và phương thức get sẽ thực thi stored procedure GetDoctorDetailsByRoleId để lấy danh sách bác sĩ, kết quả sau đó được lưu vào ViewBag.list để truyền xuống view hiển thị trên giao diện người dùng.

Khi người quản trị muốn xem cụ thể thông tin của một bác sĩ, họ sẽ sử dụng chức năng ChiTietBacSi. Phương thức này nhận vào một id là mã người dùng để gọi stored procedure tương ứng:

```
public IActionResult ChiTietBacSi(string id)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC GetDoctorDetailsByUser_Id " +id+ ";");
    return View();
}
```

Câu lệnh SQL "EXEC GetDoctorDetailsByUser_Id " sẽ lấy thông tin chi tiết của bác sĩ dựa vào mã người dùng (User_Id), bao gồm tên, chuyên ngành, mức giá khám, thông tin liên hệ, v.v. Thông tin này sau đó cũng được đẩy vào ViewBag.list để render ra giao diện một cách trực quan.

Chức năng SreachBacSi cho phép Admin tìm bác sĩ dựa vào tên, sử dụng stored procedure SearchDoctorByName. Đây là tính năng rất quan trọng khi hệ thống có hàng trăm bác sĩ và cần tìm kiếm nhanh chóng:

```
public IActionResult SreachBacSi(string tenbs)
```

THIẾT KẾ CHƯƠNG TRÌNH

```
{  
    DataModel db = new DataModel();  
    ViewBag.list = db.get("EXEC SearchDoctorByName N'" +tenbs+ "'");  
    return View();  
}
```

Từ khóa N' trước chuỗi truyền vào để đảm bảo hỗ trợ Unicode – rất cần thiết khi làm việc với tiếng Việt. Kết quả tìm được sẽ được hiển thị như bảng danh sách đã lọc. Xóa bác sĩ là một thao tác nhạy cảm, vì có thể dẫn đến mất dữ liệu quan trọng. Phương thức XoaBacSi được định nghĩa như sau:

```
[HttpPost]  
public IActionResult XoaBacSi(string mabs)  
{  
    DataModel db = new DataModel();  
    try  
    {  
        db.get("EXEC DeleteDoctor " + mabs + ");"  
        TempData["Message"] = "Xóa bác sĩ thành công!";  
    }  
    catch (Exception)  
    {  
        TempData["Error"] = "Bác sĩ được chọn không xóa được!!!!";  
    }  
    return RedirectToAction("HoSoBenhNhan", "Admin");  
}
```

Hệ thống sử dụng try-catch để xử lý lỗi, ví dụ nếu bác sĩ đang còn bệnh nhân điều trị thì sẽ xảy ra lỗi liên quan đến ràng buộc khóa ngoại. Khi đó, hệ thống sẽ thông báo không thể xóa. Nếu xóa thành công, một thông báo thành công sẽ hiển thị. Sau khi xóa, hệ thống chuyển hướng về trang HoSoBenhNhan để cập nhật lại giao diện.

```
public IActionResult DonDangKyBacSi()  
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    DataModel db = new DataModel();
    ViewBag.listDDK = db.get("EXEC GetPatientsByDoctor");
    return View();
}
```

Stored procedure GetPatientsByDoctor có vẻ lấy danh sách những người đang chờ được duyệt làm bác sĩ, có thể là bệnh nhân trước đó có nguyện vọng trở thành bác sĩ. Kết quả được lưu ở ViewBag.listDDK.

Để kiểm tra kỹ hơn từng hồ sơ đăng ký, Admin có thể vào chức năng ChiTietDonDangKy:

```
public IActionResult ChiTietDonDangKy(string mabs)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC GetDoctorDetails " + mabs + ";");
    return View();
}
```

Hệ thống sẽ gọi thủ tục GetDoctorDetails, lấy toàn bộ thông tin mà người đăng ký cung cấp – giúp Admin đưa ra quyết định có duyệt hay không.

Khi một đơn đăng ký hợp lệ, Admin sẽ duyệt và thiết lập số tiền khám thông qua phương thức DuyetBacSi:

```
[HttpPost]
public IActionResult DuyetBacSi(string mabs, string sotienkham) {
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC ConfirmDoctor " + mabs + "," + sotienkham + ";");
    return RedirectToAction("DonDangKyBacSi", "Admin");
}
```

Admin truyền vào mã bác sĩ và mức giá khám, sau đó hệ thống gọi stored procedure ConfirmDoctor để cập nhật dữ liệu. Sau khi xử lý xong, người dùng được chuyển về danh sách đơn đăng ký để tiếp tục các yêu cầu khác.

Nếu phát hiện hồ sơ không hợp lệ, Admin có thể hủy yêu cầu đăng ký bác sĩ bằng:

```
[HttpPost]
```

```
public IActionResult HuyDonDangKy(string mabs)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC DeleteDoctorRegistration " + mabs + ";");
    return RedirectToAction("DonDangKyBacSi", "Admin");
}
```

Phương thức này gọi stored procedure DeleteDoctorRegistration, xóa thông tin đăng ký dựa vào mã bác sĩ. Sau khi xóa, trang sẽ load lại danh sách đơn để tiếp tục duyệt.

5.2.33. Chức năng [Quản lý lịch khám]

Trong hệ thống quản lý bệnh viện này, chức năng Lịch khám bệnh đóng vai trò rất quan trọng giúp quản trị viên có thể theo dõi toàn bộ lịch hẹn khám của các bệnh nhân với bác sĩ. Phương thức LichKhamBenh() thuộc controller AdminController chịu trách nhiệm đảm nhiệm chức năng này. Khi phương thức này được gọi, nó sẽ thực hiện lấy dữ liệu lịch hẹn từ cơ sở dữ liệu thông qua việc thực thi một stored procedure có tên là GetAppointmentDetails. Cụ thể, phương thức sử dụng lớp DataModel, đây là lớp trung gian để xử lý tương tác với cơ sở dữ liệu. Đầu tiên, nó tạo một đối tượng DataModel mới và sau đó gọi hàm get() để thực thi câu lệnh "EXEC GetAppointmentDetails", câu lệnh này sẽ trả về toàn bộ dữ liệu liên quan đến lịch khám bệnh của các bệnh nhân đang có trong hệ thống. Kết quả truy vấn này được lưu trữ vào biến ViewBag.listLH, để truyền dữ liệu sang view tương ứng, cho phép hiển thị dữ liệu một cách trực quan cho người dùng. Sau cùng, phương thức trả về một view thông qua lệnh return View();. View này có thể là một trang web chứa bảng liệt kê tất cả các lịch hẹn đã được xác lập, bao gồm thông tin về bệnh nhân, bác sĩ phụ trách, thời gian khám, phòng khám và trạng thái của cuộc hẹn. Đây là chức năng cần thiết giúp cho quản trị viên dễ dàng kiểm soát và điều phối hoạt động khám chữa bệnh trong bệnh viện, đảm bảo quá trình làm việc được trơn tru và không bị chồng chéo lịch.

```
// Lịch khám
public IActionResult LichKhamBenh()
{
    DataModel db = new DataModel();
```

```
ViewBag.listLH = db.get("EXEC GetAppointmentDetails");
return View();
}
// End Lịch khám
```

5.2.34. Chức năng [Quản lý đánh giá]

Chức năng Đánh giá phản hồi trong hệ thống quản lý này đóng vai trò như một cầu nối giữa bệnh nhân và bệnh viện, cho phép bệnh nhân để lại các nhận xét, góp ý, hoặc đánh giá chất lượng dịch vụ sau khi đã trải nghiệm quy trình khám chữa bệnh. Đây là một chức năng rất quan trọng giúp ban quản trị bệnh viện lắng nghe ý kiến từ người bệnh, từ đó cải thiện chất lượng dịch vụ và nâng cao trải nghiệm tổng thể cho bệnh nhân trong tương lai.

Trong đoạn mã thuộc controller, phương thức `DanhGiaPhanHoi()` được sử dụng để xử lý yêu cầu truy cập vào trang hiển thị danh sách đánh giá từ phía người dùng. Khi phương thức này được gọi, hệ thống sẽ khởi tạo một đối tượng `DataModel`, đây là lớp được xây dựng để quản lý việc truy vấn và tương tác với cơ sở dữ liệu. Sau khi khởi tạo, hệ thống thực hiện một truy vấn đến cơ sở dữ liệu thông qua stored procedure `GetAllRatings`, với mục đích lấy về toàn bộ thông tin các đánh giá đã được ghi nhận trong hệ thống. Câu lệnh "`EXEC GetAllRatings`" được truyền vào phương thức `get()` của đối tượng `DataModel`, và kết quả trả về sẽ được lưu trong `ViewBag.listDG`. Việc sử dụng `ViewBag` ở đây là để truyền dữ liệu một cách nhanh chóng và linh hoạt từ controller sang view tương ứng.

Khi view được render ra, dữ liệu trong `ViewBag.listDG` sẽ được sử dụng để hiển thị danh sách các đánh giá. Mỗi đánh giá có thể bao gồm nội dung như tên người đánh giá, thời gian gửi phản hồi, nội dung bình luận, mức độ hài lòng (thường thể hiện bằng số sao), cũng như có thể kèm theo các đề xuất cải thiện. Việc quản trị viên có thể xem được toàn bộ phản hồi từ người bệnh sẽ giúp họ nhận diện được những điểm yếu trong hệ thống hiện tại, từ đó đưa ra những điều chỉnh kịp thời. Không những vậy, những phản hồi tích cực còn là minh chứng rõ ràng cho chất lượng dịch vụ của bệnh viện, giúp xây dựng hình ảnh và uy tín đối với cộng đồng.

Tóm lại, phương thức DanhGiaPhanHoi () chính là một phần quan trọng trong chuỗi quản lý chất lượng, giúp bệnh viện tiến gần hơn đến mô hình chăm sóc sức khỏe hiện đại, đặt bệnh nhân làm trung tâm, và không ngừng cải tiến dựa trên phản hồi thực tế từ người sử dụng dịch vụ.

```
// Đánh giá
public IActionResult DanhGiaPhanHoi()
{
    DataModel db = new DataModel();
    ViewBag.listDG = db.get("EXEC GetAllRatings ");
    return View();
}
// End Đánh giá
```

5.2.35. Chức năng [Thông kê]

Một trong những chức năng quan trọng của hệ thống quản lý y tế là Thông kê, giúp ban quản trị có thể theo dõi hiệu quả hoạt động của cơ sở y tế qua từng khoảng thời gian. Trong hệ thống này, chức năng thông kê được hiện thực thông qua một action mang tên ThongKe trong AdminController. Dưới đây là đoạn mã nguồn cụ thể của action này:

```
//Thông kê
public IActionResult ThongKe()
{
    DataModel db = new DataModel();

    ViewBag.listTKLK = JsonConvert.SerializeObject(db.get("EXEC
ThongKeLuotKhamTheoThang"));
    ViewBag.listTKDT = JsonConvert.SerializeObject(db.get("EXEC
ThongKeDoanhThuKhachTheoThang"));

    return View();
}
```

THIẾT KẾ CHƯƠNG TRÌNH

//End Thống kê

Về mặt chức năng, phương thức `ThongKe()` chịu trách nhiệm lấy dữ liệu thống kê từ cơ sở dữ liệu bằng cách sử dụng hai thủ tục lưu sẵn (Stored Procedures). Đầu tiên, đối tượng `DataModel` được khởi tạo để có thể thực hiện các truy vấn cơ sở dữ liệu. `DataModel` ở đây là lớp trung gian giúp kết nối và thực hiện các câu lệnh SQL từ controller.

```
ViewBag.listTKLK = JsonConvert.SerializeObject(db.get("EXEC  
ThongKeLuotKhamTheoThang"));
```

được dùng để lấy dữ liệu thống kê lượt khám bệnh theo từng tháng. Lệnh SQL "EXEC ThongKeLuotKhamTheoThang" sẽ gọi một thủ tục lưu sẵn từ SQL Server – thủ tục này trả về danh sách số lượt khám trong từng tháng, có thể bao gồm các cột như tháng, năm và số lượt khám. Kết quả truy vấn được trả về dưới dạng một danh sách đối tượng, nhưng trước khi truyền sang View, danh sách này được chuyển đổi thành dạng JSON bằng thư viện `JsonConvert.SerializeObject()`. Việc này là cần thiết trong trường hợp dữ liệu sẽ được hiển thị trên biểu đồ trong View bằng JavaScript – chẳng hạn như thư viện Chart.js, nơi định dạng JSON là lý tưởng để thao tác với dữ liệu.

Tương tự, dòng lệnh thứ hai:

```
ViewBag.listTKDT = JsonConvert.SerializeObject(db.get("EXEC  
ThongKeDoanhThuKhachTheoThang"));
```

sẽ lấy dữ liệu về doanh thu từ bệnh nhân theo từng tháng. Thủ tục "ThongKeDoanhThuKhachTheoThang" có thể truy vấn tổng số tiền mà mỗi bệnh nhân đã thanh toán hoặc tổng doanh thu theo tháng, hỗ trợ ban quản trị theo dõi tình hình tài chính của cơ sở khám chữa bệnh. Dữ liệu này cũng được chuyển về định dạng JSON và lưu vào `ViewBag.listTKDT`, để View sử dụng hiển thị biểu đồ trực quan.

Cuối cùng, phương thức `ThongKe()` trả về View tương ứng thông qua lệnh `return View();`. Ở View đó (thường là `ThongKe.cshtml`), ta có thể dùng các biểu đồ để thể hiện trực quan hai nguồn dữ liệu kể trên – một biểu đồ đường cho

lượt khám bệnh, và một biểu đồ cột cho doanh thu, giúp người quản trị nắm bắt xu hướng phát triển của phòng khám theo thời gian.

Tổng thể, action ThongKe() không chỉ giúp hiển thị dữ liệu quan trọng về hoạt động khám bệnh và doanh thu, mà còn thể hiện rõ khả năng kết hợp giữa truy vấn SQL, xử lý dữ liệu phía backend (C#), và frontend (biểu đồ) – tạo ra một hệ thống thống kê mạnh mẽ, minh bạch và trực quan trong hệ thống quản lý bệnh viện.

5.2.36. Chức năng [Quản lý thông báo]

Trong hệ thống quản lý bệnh viện, chức năng gửi và quản lý thông báo đóng vai trò vô cùng quan trọng nhằm cung cấp các cập nhật, hướng dẫn hoặc cảnh báo cần thiết đến người dùng (như bệnh nhân, y bác sĩ hay nhân viên y tế). Trong đoạn code thuộc lớp AdminController, chức năng quản lý thông báo được hiện thực với ba hành động chính là hiển thị danh sách thông báo, gửi thông báo mới, và xóa thông báo đã có.

```
public IActionResult ThongBaoBV()
{
    DataModel db = new DataModel();
    ViewBag.listTB = db.get("EXEC LayDanhSachThongBao ");
    ViewBag.listND = db.get("SELECT * from NGUOIDUNG ");
    return View();
}
```

Trong phương thức ThongBaoBV, một đối tượng DataModel được tạo ra để truy xuất dữ liệu từ cơ sở dữ liệu. Phương thức db.get(...) được sử dụng để gọi stored procedure LayDanhSachThongBao nhằm lấy tất cả các thông báo đã được gửi trước đó, và kết quả được gán vào ViewBag.listTB để hiển thị ở phần giao diện. Đồng thời, truy vấn "SELECT * from NGUOIDUNG" cũng được thực hiện để lấy thông tin người dùng – thường được dùng trong dropdown hoặc các biểu mẫu để chọn người nhận thông báo. Tất cả dữ liệu này sẽ được truyền sang view tương ứng để hiển thị.

```
[HttpPost]
public IActionResult GuiThongBao(string tieudetb, string noidungtb, string thoigiantb, string mand)
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
try
{
    DateTime parsedTime = DateTime.Parse(thoigiantb);
    DataModel db = new DataModel();
    db.get($"EXEC GuiThongBaoChoNguoiDung N'{tieudetb}', N'{noidungtb}',
    '{parsedTime}', '{mand}']");
    TempData["Message"] = "Thông báo đã được gửi thành công!";
}
catch (Exception ex)
{
    TempData["Error"] = "Có lỗi xảy ra khi gửi thông báo: " + ex.Message;
}

return RedirectToAction("ThongBaoBV", "Admin");
}
```

Phương thức GuiThongBao được gọi khi người quản trị gửi một biểu mẫu tạo thông báo mới. Các tham số đầu vào như tieudetb, noidungtb, thoigiantb, và mand là lần lượt tương ứng với tiêu đề, nội dung, thời gian gửi, và mã người dùng. Trước tiên, chuỗi thời gian được chuyển đổi sang kiểu DateTime để đảm bảo định dạng phù hợp với dữ liệu trong SQL Server. Sau đó, câu lệnh EXEC GuiThongBaoChoNguoiDung sẽ gọi đến stored procedure để thực thi việc chèn dữ liệu vào bảng thông báo. Kết quả thành công hoặc thất bại được phản hồi lại cho người dùng bằng TempData, từ đó có thể hiển thị ở view để [HttpPost]

```
public IActionResult XoaThongBao(string matb)
{
    DataModel db = new DataModel();
    try
    {
```

THIẾT KẾ CHƯƠNG TRÌNH

```
        db.get("EXEC XoaThongBao " + matb + ";");
        TempData["Message"] = "Xóa thông báo thành công!";
    }
    catch (Exception)
    {
        TempData["Error"] = "Thông báo được chọn không xóa được!!!!";
    }
    return RedirectToAction("ThongBaoBV", "Admin");
}
```

Đối với việc xóa thông báo, phương thức `XoaThongBao` sẽ nhận vào `matb` – là mã thông báo cần xóa. Hàm `db.get(...)` tiếp tục được sử dụng để gọi stored procedure `XoaThongBao`, thực hiện việc xóa thông báo tương ứng trong cơ sở dữ liệu. Quá trình này được bọc trong khôi `try-catch` nhằm xử lý ngoại lệ, ví dụ như khi thông báo đang được tham chiếu bởi bảng khác, hoặc có lỗi truy cập dữ liệu. Sau đó, người dùng được chuyển hướng lại về trang danh sách thông báo để cập nhật giao diện.

5.2.37. Chức năng [Quản lý thanh toán]

Trong phần quản trị của hệ thống quản lý bệnh viện, chức năng thống kê được triển khai thông qua action `ThongKe()` trong controller. Khi người dùng truy cập vào trang thống kê, hệ thống sẽ tạo một đối tượng `DataModel` để kết nối với cơ sở dữ liệu. Từ đó, hai stored procedure `ThongKeLuotKhamTheoThang` và `ThongKeDoanhThuKhachTheoThang` sẽ được gọi để lấy dữ liệu thống kê lượt khám và doanh thu theo từng tháng. Kết quả sau khi truy vấn được chuyển thành dạng JSON thông qua `JsonConvert.SerializeObject`, rồi gán vào `ViewBag` để truyền xuống View, nơi dữ liệu sẽ được hiển thị dưới dạng biểu đồ hoặc bảng thống kê.

```
public IActionResult ThongKe()
{
    DataModel db = new DataModel();
    ViewBag.listTKLK = JsonConvert.SerializeObject(db.get("EXEC
ThongKeLuotKhamTheoThang"));
    ViewBag.listTKDT = JsonConvert.SerializeObject(db.get("EXEC
ThongKeDoanhThuKhachTheoThang"));
}
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    return View();
}
```

Tiếp theo là chức năng quản lý và gửi thông báo cho người dùng, được xử lý trong action `ThongBaoBV()` và `GuiThongBao()`. Trong `ThongBaoBV()`, hệ thống sẽ lấy danh sách các thông báo đã được gửi bằng cách gọi stored procedure `LayDanhSachThongBao`, đồng thời truy vấn thêm danh sách người dùng từ bảng `NGUOIDUNG` để phục vụ việc chọn người nhận thông báo. Hai danh sách này sẽ được truyền xuống View thông qua `ViewBag`.

```
public IActionResult ThongBaoBV()
{
    DataModel db = new DataModel();
    ViewBag.listTB = db.get("EXEC LayDanhSachThongBao");
    ViewBag.listND = db.get("SELECT * from NGUOIDUNG");
    return View();
}
```

Khi admin muốn gửi một thông báo mới, action `GuiThongBao()` sẽ được gọi với các tham số như tiêu đề, nội dung, thời gian gửi và mã người nhận. Thời gian được truyền vào dưới dạng chuỗi sẽ được chuyển đổi sang kiểu `DateTime`. Sau đó, stored procedure `GuiThongBaoChoNguoiDung` được gọi để lưu thông báo vào hệ thống. Nếu thành công, một thông báo xác nhận sẽ được lưu trong `TempData`, ngược lại, nếu có lỗi xảy ra, thông báo lỗi cũng được hiển thị.

```
[HttpPost]
public IActionResult GuiThongBao(string tieudetb, string noidungtb, string thoigiantb, string mand)
{
    try
    {
        DateTime parsedTime = DateTime.Parse(thoigiantb);
        DataModel db = new DataModel();
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    db.get("$"EXEC GuiThongBaoChoNguoiDung N'{tieudetb}', N'{noidungtb}',  
'{parsedTime}', '{mand}");  
    TempData["Message"] = "Thông báo đã được gửi thành công!";  
}  
catch (Exception ex)  
{  
    TempData["Error"] = "Có lỗi xảy ra khi gửi thông báo: " + ex.Message;  
}  
return RedirectToAction("ThongBaoBV", "Admin");  
}
```

Ngoài ra, hệ thống cũng hỗ trợ xóa thông báo thông qua action `XoaThongBao()`, trong đó stored procedure `XoaThongBao` sẽ được gọi với mã thông báo cần xóa. Nếu thao tác thành công, một thông báo xác nhận được trả về, còn nếu thất bại, hệ thống sẽ gửi thông báo lỗi cho người quản trị.

```
[HttpPost]  
public IActionResult XoaThongBao(string matb)  
{  
    DataModel db = new DataModel();  
    try  
    {  
        db.get("EXEC XoaThongBao " + matb + ";");  
        TempData["Message"] = "Xóa thông báo thành công!";  
    }  
    catch (Exception)  
    {  
        TempData["Error"] = "Thông báo được chọn không xóa được!!!!";  
    }  
    return RedirectToAction("ThongBaoBV", "Admin");  
}
```

THIẾT KẾ CHƯƠNG TRÌNH

Một phần quan trọng nữa là chức năng thanh toán được xử lý bởi action `DSThanhToan()` và `HoanPhiKham()`. Trong `DSThanhToan()`, hệ thống gọi stored procedure `LayDanhSachThanhToan` để truy xuất danh sách các giao dịch thanh toán hiện có và truyền danh sách này xuống view thông qua `ViewBag`.

```
public IActionResult DSThanhToan()
{
    DataModel db = new DataModel();
    ViewBag.litsTT = db.get("EXEC LayDanhSachThanhToan");
    return View();
}
```

Trong trường hợp người quản trị cần hoàn phí khám cho bệnh nhân (ví dụ: do hủy lịch khám hoặc lỗi hệ thống), action `HoanPhiKham()` sẽ được thực thi. Hệ thống sẽ gọi stored procedure `HoanPhiKham` với mã thanh toán cụ thể, thực hiện hoàn lại phí và sau đó chuyển hướng về danh sách thanh toán để cập nhật lại giao diện.

```
[HttpPost]
public IActionResult HoanPhiKham(string matt)
{
    DataModel db = new DataModel();
    ViewBag.list = db.get("EXEC HoanPhiKham " + matt);
    return RedirectToAction("DSThanhToan", "Admin");
}
```

Cuối cùng, để xử lý các lỗi không mong muốn trong toàn hệ thống, một action `Error()` đã được định nghĩa. Action này sẽ trả về view lỗi mặc định (`Error.cshtml`) để đảm bảo trải nghiệm người dùng không bị gián đoạn khi có lỗi xảy ra trong quá trình thực thi các chức năng quản trị.

```
[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore
= true)]
public IActionResult Error()
{
```

THIẾT KẾ CHƯƠNG TRÌNH

```
    return View();
}
```

CHƯƠNG 6: CHƯƠNG TRÌNH THỰC NGHIỆM

6.1. Chi tiết các giao diện người dùng

6.1.1. Giao diện [trang chủ]

The screenshot shows the homepage of the VOV BacSi24 website. At the top, there is a navigation bar with links for 'Cấp cứu', 'Chuyên khoa', 'Hướng dẫn sử dụng', 'Tài khoản', 'Giao dịch', 'Về VOVBacSi24', and 'Chuyên trang sức khỏe'. There is also a search bar and a notification bell icon.

Main content area:

- Chăm sóc bằng tài năng, y đức và sự thấu cảm**: A large banner with a doctor's photo and text about the service.
- VOV Bác sĩ cam kết về trách nhiệm xã hội và đóng góp tích cực vào sự phát triển bền vững của cộng đồng**: A statement from the service.
- Xem thêm**: A button to view more information.
- Gọi tổng đài**: A link to call the service.
- Đặt Lịch Hẹn**: A link to book an appointment.
- Tìm bác sĩ**: A link to search for doctors.
- Times City (+84) 243 974 3556**
- Hải Phòng (+84) 225 730 9888**
- Hà Long (+84) 203 382 8188**
- Đà Nẵng (+84) 236 371 1111**
- Nha Trang (+84) 258 390 0168**
- Tp. Hồ Chí Minh (+84) 283 622 1166**
- Phú Quốc (+84) 297 398 5588**
- Chọn theo tên, chuyên môn và nhiều hơn thế**: A dropdown menu for filtering doctors.
- Đội ngũ bác sĩ uy tín**: A section featuring a doctor's photo and text about the quality of the medical staff.
- Tư vấn y tế nhanh chóng**: A section featuring a doctor's photo and text about the quick consultation service.
- Tại sao nên chọn VOV Bác sĩ 24?**: A section listing reasons to choose the service, each with an icon and text.
- Đội ngũ bác sĩ uy tín**: Icons of a doctor's hands and a stethoscope.
- Dịch vụ trực tuyến tiện lợi**: Icons of a smartphone and a laptop.
- Công nghệ y tế tiên tiến**: Icons of a brain and a DNA helix.

CHƯƠNG TRÌNH THỰC NGHIỆM

Danh sách các bệnh viện hàng đầu Việt Nam

Bệnh viện Chợ Rẫy - Cập nhật

Bệnh viện Đại học Y Dược

Bệnh viện Từ Dũ

Bệnh viện FV

Bệnh viện Quân Y

Top các bác sĩ hàng đầu

VOV Bác sĩ 24 tự hào quy tụ đội ngũ bác sĩ hàng đầu với nhiều năm kinh nghiệm, chuyên môn cao, luôn tận tâm trong việc chăm sóc sức khỏe cho bệnh nhân và được các bệnh nhân thường xuyên lựa chọn.

Cập nhật

Được

Top các bác sĩ hàng đầu

VOV Bác sĩ 24 tự hào quy tụ đội ngũ bác sĩ hàng đầu với nhiều năm kinh nghiệm, chuyên môn cao, luôn tận tâm trong việc chăm sóc sức khỏe cho bệnh nhân và được các bệnh nhân thường xuyên lựa chọn.

Xem thêm

Hệ thống VOVBacSi24
ĐÀI TIẾNG NÓI VIỆT NAM

Dịch vụ
Chuyên khoa

Theo dõi chúng tôi



CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.2. Giao diện [danh sách bài viết]

The screenshot shows the homepage of the VOVBacSi24 website. At the top, there is a blue header bar with the text "Dành cho bác sĩ". Below the header, the VOV BacSi24 logo is on the left, followed by a search bar containing the placeholder "Tim kiếm..." and icons for a magnifying glass and a bell. To the right of the search bar is a user profile icon. The main content area has a light blue background. It features several horizontal cards, each containing a title and a small red circular badge with a white number (e.g., 82). The titles include: "Khám và điều trị theo tiêu chuẩn Mỹ", "Phẫu thuật thần kinh", "CHẨN THƯƠNG CHỈNH HÌNH - Y HỌC THỂ THAO", and "CẤP CỨU - HỒI SỨC TÍCH CỰC". Below these cards, a larger card displays the text "Đồng hành cùng ba mẹ chăm sóc và nâng niu những mầm xanh!".

6.1.3. Giao diện [bài viết]

The screenshot shows a specific article page from the VOVBacSi24 website. The top navigation bar is identical to the homepage. The main content area features a large, dark-toned photograph of a person's hands, with the word "CẤP CỨU" overlaid in large white capital letters. Below the image, the breadcrumb navigation shows "Trang chủ / Cấp cứu / SỨC KHOẺ TỔNG QUÁ". The main title of the article is "SỨC KHOẺ TỔNG QUÁ", displayed prominently in red capital letters. Below the title is a video player interface with the VOVBacSi24 logo, a play button, and the text "Bệnh mạch vành" and "TẬN TÂM VÌ SỨC KHỎE NGƯỜI VIỆT". At the bottom of the video player, there are download links for "App Store" and "GOOGLE PLAY".

CHƯƠNG TRÌNH THỰC NGHIỆM

Tổng quan

Trong bối cảnh nhịp sống bận rộn và sự thay đổi không ngừng của môi trường sống&44; cơ thể chúng ta có thể đã xuất hiện dấu hiệu tiềm ẩn của các bệnh lý nghiêm trọng mà không thể nhận biết bằng mắt thường. Ngoài ra&44; những thói quen không lành mạnh như làm việc quá sức&44; thức khuya&44; sử dụng bia rượu&44; căng thẳng kéo dài cũng có thể là nguyên nhân dẫn đến các bệnh lý nguy hiểm như đột quỵ&44; ung thư&44; tim mạch&44; xương khớp...

Lợi ích kiểm tra sức khỏe định kỳ

Kiểm tra sức khỏe tổng quát định kỳ với đầy đủ các hạng mục là rất cần thiết&44; mang lại nhiều lợi ích quan trọng như:

Được đánh giá tình trạng sức khỏe tổng quát của bản thân

Giúp phát hiện sớm những thay đổi bất thường

Ngăn chặn kịp thời các biến chứng nguy hiểm

Tăng cơ hội chữa khỏi bệnh hoàn toàn&44; giảm chi phí điều trị

Được tư vấn để duy trì sức khỏe tốt&44; cải thiện chất lượng cuộc sống

Khám sức khỏe tại Vinmec

Thấu hiểu nhu cầu chăm sóc sức khỏe chủ động của khách hàng&44; Vinmec đã đầu tư trang thiết bị hiện đại&44; kết hợp với đội ngũ y bác sĩ giàu kinh nghiệm không chỉ giúp khách hàng tầm soát bệnh mà còn mang đến một trải nghiệm khác biệt về khám chữa bệnh chủ động.

Đồng thời&44; Vinmec cũng cung cấp các dịch vụ khám sức khỏe toàn diện&44; từ cơ bản đến chuyên sâu theo tiêu chuẩn Quốc tế&44; đáp ứng mọi nhu cầu kiểm tra sức khỏe của Quý khách hàng và gia đình.

Hệ thống VOVBacSi24

ĐÀI TIẾNG NÓI VIỆT NAM



KÊNH SỨC KHỎE - MỚI THƯỜNG - AN TOÀN THỰC PHẨM

Tập trung vì sức khỏe người Việt

Toà nhà CT1AB Mễ Trì Plaza
Nam Từ Liêm - Hà Nội

Hotline: 1900 1289

Dịch vụ

Chuyên khoa

Gói dịch vụ

Bảo hiểm

Đặt lịch hẹn

Theo dõi chúng tôi



6.1.4. Giao diện [Lọc bác sĩ]

The screenshot shows the homepage of the VOVBacSi24 website. At the top, there is a navigation bar with links for 'Cấp cứu', 'Chuyên khoa', 'Hướng dẫn sử dụng', 'Tài khoản', 'Giao dịch', 'Về VOVBacSi24', and 'Chuyên trang sức khỏe'. A search bar with the placeholder 'Tim kiếm...' is located above a large banner image featuring a close-up of two hands. The banner has the text 'DANH SÁCH BÁC SĨ' overlaid. Below the banner, there is a search form with dropdown menus for 'Khu vực', 'Phi khám', 'khoa bệnh', 'Học hàm', and a green button labeled 'Lọc bác sĩ'. There is also a text input field for 'Nhập tên bác sĩ...' and a green 'Tim kiếm' button. Below this, there is a section titled 'Danh sách bác sĩ' with two entries: 'Phạm Minh Bác Sĩ' and 'đẹp'.

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.5. Giao diện [danh sách bác sĩ]

Danh sách bác sĩ

 Phạm Minh Bác Sĩ Chuyên khoa: Nội khoa Đơn vị: Bệnh viện Chợ Rẫy - Cập nhật Phi khám: 300,000đ ★★★★★	 Hoàng Hương Diễm Chuyên khoa: Nội khoa Đơn vị: Bệnh viện Chợ Rẫy - Cập nhật Phi khám: 250,000đ ★★★★★
 Phương Chuyên khoa: Tai mũi họng Đơn vị: Bệnh viện Quân Y Phi khám: 150,000đ ★★★★★	 Bắc Chuyên khoa: Nội khoa Đơn vị: Bệnh viện Đại học Y Dược Phi khám: 100,000đ ★★★★★
 ew Chuyên khoa: Chấn thương Đơn vị: Bệnh viện Bạch mai Phi khám: 250,000đ ★★★★★	 jghg Chuyên khoa: Nội khoa Đơn vị: Bệnh viện Đại học Y Dược Phi khám: 250,000đ ★★★★★
 hns	 Khôi Lê

6.1.6. Giao diện [chi tiết bác sĩ]



Phạm Minh Bác Sĩ
Chuyên môn: Chuyên khoa nội
Phi khám: 300,000 đ
★★★★★

Chuyên môn
Bác sĩ nội khoa

Nơi làm việc
Bệnh viện Chợ Rẫy - Cập nhật

Dịch vụ
Bệnh tiêu hóa&44; hô hấp

Quá trình đào tạo

Kinh nghiệm làm việc

Công trình nghiên cứu

Thành viên của tổ chức

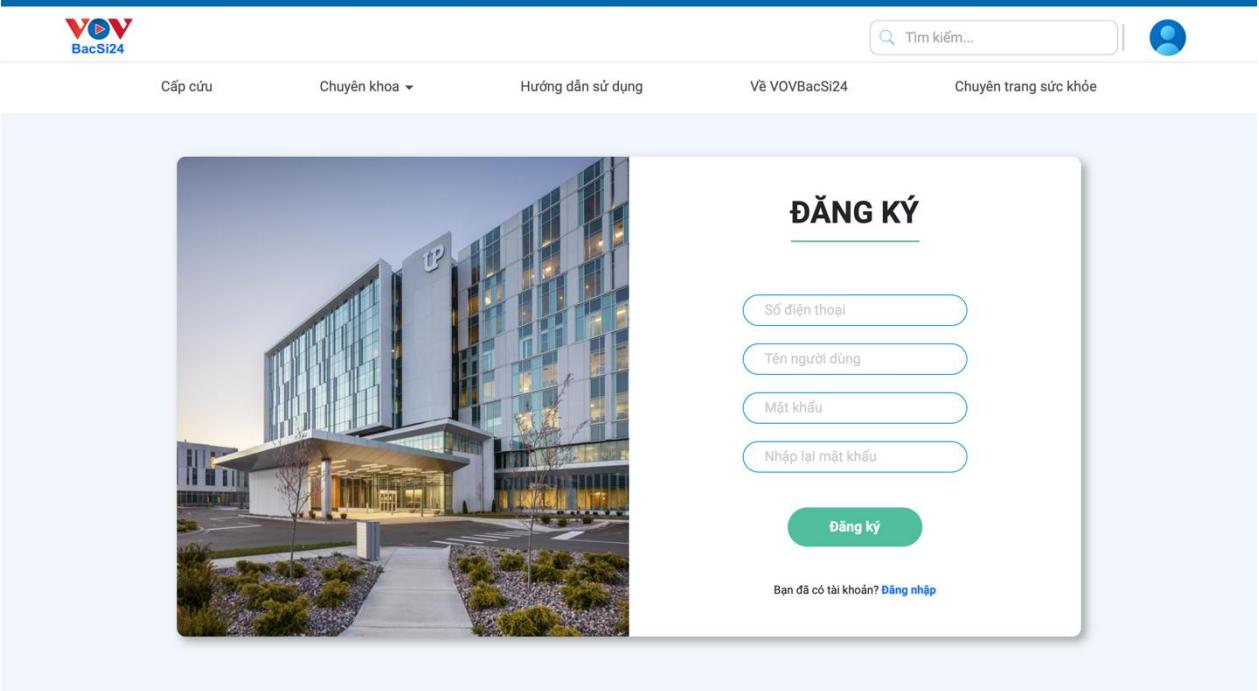
Đánh giá

 **Nguyễn Văn Adminvsd**
Đăng ngày: 9/20/2024 12:00:00 AM
★★★★★
Dịch vụ rất tốt&44; bác sĩ nhiệt tình và chu đáo.

 **Trần Thị Bệnh Nhân Cập Nhật**
Đăng ngày: 9/21/2024 12:00:00 AM
★★★★★
Bác sĩ tư vấn kỹ càng&44; nhưng thời gian chờ hơi lâu.

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.7. Giao diện [đăng ký tài khoản]



VOV
BacSi24

Cấp cứu Chuyên khoa ▾ Hướng dẫn sử dụng Về VOVBacSi24 Chuyên trang sức khỏe

ĐĂNG KÝ

Số điện thoại

Tên người dùng

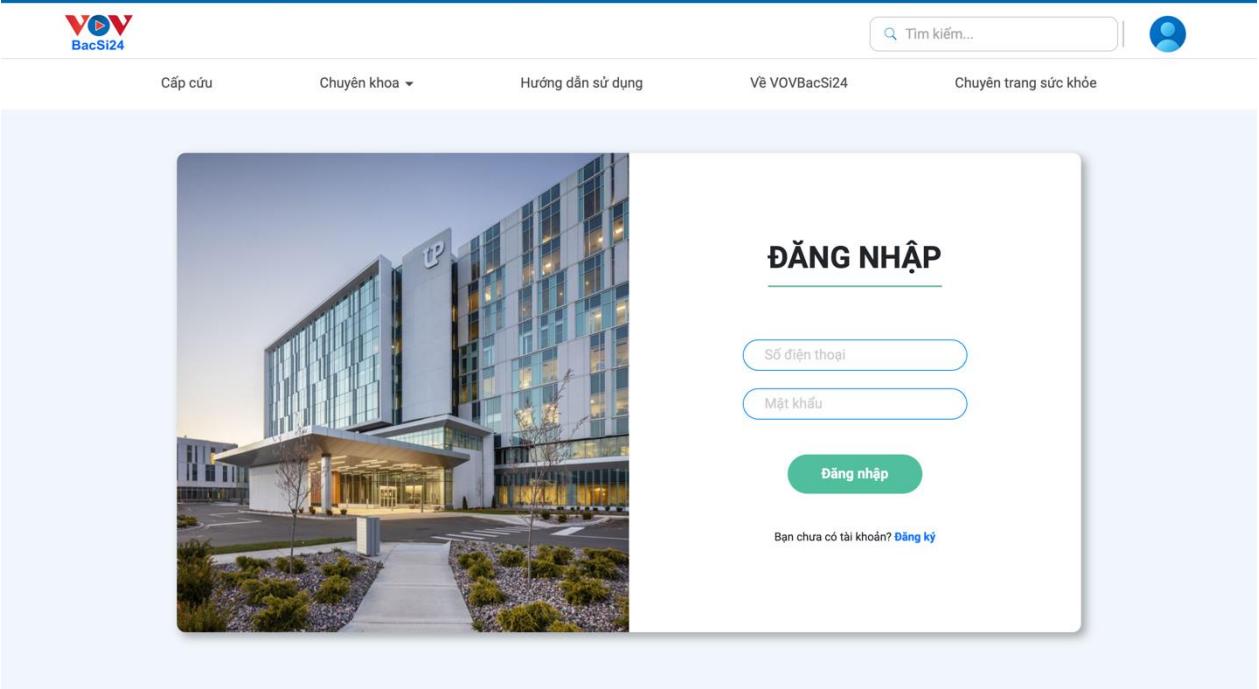
Mật khẩu

Nhập lại mật khẩu

Đăng ký

Bạn đã có tài khoản? [Đăng nhập](#)

6.1.8. Giao diện [đăng nhập]



VOV
BacSi24

Cấp cứu Chuyên khoa ▾ Hướng dẫn sử dụng Về VOVBacSi24 Chuyên trang sức khỏe

ĐĂNG NHẬP

Số điện thoại

Mật khẩu

Đăng nhập

Bạn chưa có tài khoản? [Đăng ký](#)

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.9. Giao diện [cập nhật thông tin cá nhân]



Lương Tiến Đạt

Mã giới thiệu: 104

[Đổi mật khẩu](#)

Số dư: 400,020 đ

THÔNG TIN CÁ NHÂN

Họ và tên:

Email:

Năm sinh:

Giới tính:

Địa chỉ:

Số điện thoại:

Ảnh cá nhân:

Choose File

[Cập nhật](#)

6.1.10. Giao diện [Quên mật khẩu]

Dành cho bác sĩ



Để thoát khỏi chế độ toàn màn hình, hãy nhấn và giữ [Thoát](#)

[Tìm kiếm...](#) 

Cấp cứu

Chuyên khoa ▾

Hướng dẫn sử dụng

Về VOVBacSi24

Chuyên trang sức khỏe



QUÊN MẬT KHẨU

Nhập email để nhận mã xác thực

[Gửi mã xác thực](#)

[← Quay lại đăng nhập](#)

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.11. Giao diện [Đổi mật khẩu]

The screenshot shows a mobile application interface for changing a password. At the top center is a blue padlock icon above the text "ĐỔI MẬT KHẨU". Below this, a message reads: "Vui lòng nhập mật khẩu hiện tại và mật khẩu mới". There are two input fields: the first for the "Mật khẩu hiện tại" (current password) and the second for the "Mật khẩu mới" (new password). Both fields have an "eye" icon to show/hide the password. A note below the new password field states: "Mật khẩu phải có ít nhất 6 ký tự" (password must contain at least 6 characters). Below the new password field is a checked checkbox labeled "Xác nhận mật khẩu mới" (confirm new password), followed by another input field with an "eye" icon. A note below this field says: "Độ mạnh mật khẩu" (password strength). At the bottom is a large blue "Đổi mật khẩu" (Change password) button with a lock icon, and a smaller white "← Quay lại" (Back) button.

6.1.12. Giao diện [nạp tiền]

The screenshot shows a mobile application interface for top-up payments. At the top, a blue bar displays the balance: "Số dư: 400,020 đ". Below this is a red header with the text "NẠP TIỀN". The main form has a light gray background. It starts with a text input field for "Số tiền cần nạp (VND)". Below it is a horizontal button bar with three options: "200,000 VND", "500,000 VND", and "1,000,000 VND". The next section is titled "Phương thức thanh toán" (Payment method) and contains three buttons: "Mobile banking" (selected, showing "Cổng thanh toán VNPay"), "Thẻ ATM nội địa" (VNPay QR logo), and "Thẻ tín dụng/Ghi nợ" (VNPay QR logo). The final sections are "Mã khuyến mại" (Promotion code) with a text input field and a "Submit" button at the bottom.

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.13. Giao diện [đăng ký lịch khám]

Trang chủ / Tim bác sĩ / Thông tin bác sĩ / Đặt Lịch khám

Số dư: 400,020 đ

Chi tiết lịch hẹn

Thông tin bác sĩ

Bác sĩ

Phạm Minh Bác Sĩ

Bệnh viện/phòng khám

Bệnh viện Chợ Rẫy - Cập nhật

Chuyên khoa

Nội khoa

Thông tin khách hàng

Họ và tên *

Lương Tiến Đạt

Giới tính: *

Nam Nữ

Số điện thoại *

0707456456

Email

ldat123@gmail.com

Ngày tháng năm sinh *

18/12/2004

Lý do khám *

Lý do khám

Triệu chứng của bạn

Hình ảnh bệnh *

Choose Files

no files selected

Phí khám: 300,000 đ

Tôi đã đọc và xác nhận [Điều khoản dịch vụ](#) của Bệnh viện.

Đặt lịch khám

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.14. Giao diện [lịch sử khám]

The screenshot shows a web-based application interface for medical treatment history. At the top, there is a navigation bar with links for 'Cấp cứu', 'Chuyên khoa', 'Hướng dẫn sử dụng', 'Tài khoản', 'Giao dịch', 'Về VOVBacSi24', and 'Chuyên trang sức khỏe'. A search bar and a user profile icon are also present. The main content area is titled 'LỊCH SỬ KHÁM' and displays a table of treatment records. The table has columns for 'STT', 'Tên bác sĩ', 'STT khám', 'Ngày đặt lịch', 'Ghi chú bác sĩ', 'Thanh toán', and 'Trạng thái'. The data in the table is as follows:

STT	Tên bác sĩ	STT khám	Ngày đặt lịch	Ghi chú bác sĩ	Thanh toán	Trạng thái
1	Hoàng Hương Diễm	0	11/23/2024 4:40:33 AM		175000	Đã hoàn thành
2	Phạm Minh Bác Sĩ	0	11/23/2024 11:38:40 PM		210000	Đã hoàn thành
3	Hoàng Hương Diễm	1	11/24/2024 2:49:41 AM			Đã hủy
4	Hoàng Hương Diễm	2	11/24/2024 3:02:59 AM			Đã hoàn thành
5	Hoàng Hương Diễm	0	11/24/2024 3:17:08 AM		175000	Đã hoàn thành
6	Hoàng Hương Diễm	0	11/24/2024 3:19:36 AM		175000	Đã hoàn thành
7	Hoàng Hương Diễm	0	11/24/2024 3:29:01 AM		250000	Đã hoàn thành
8	Phạm Minh Bác Sĩ	0	11/24/2024 7:38:44 PM		300000	Đã hoàn thành
9	Phạm Minh Bác Sĩ	0	2/28/2025 11:46:11 PM		300000	Đã hoàn thành
10	Bác	0	3/1/2025 12:05:20 AM		100000	Đã hoàn thành
11	Bác	0	3/1/2025 12:25:53 AM		100000	Đã hoàn thành
12	Phạm Minh Bác Sĩ	0	3/2/2025 1:30:52 AM		300000	Đã hoàn thành

6.1.15. Giao diện [hủy lịch khám]

This screenshot shows the same treatment history page as above, but with a modal dialog box overlaid on the row for entry number 11. The dialog box contains the question 'Bạn có chắc chắn muốn hủy lịch này không?' (Are you sure you want to cancel this appointment?). It includes 'Cancel' and 'OK' buttons. The table data remains the same as in the previous screenshot.

STT	Tên bác sĩ	STT khám	Ngày đặt lịch	Ghi chú bác sĩ	Thanh toán	Trạng thái
4	Hoàng Hương Diễm	2	11/24/2024 3:02:59 AM			Đã hoàn thành
5	Hoàng Hương Diễm	0	11/24/2024 3:17:08 AM		175000	Đã hoàn thành
6	Hoàng Hương Diễm	0	11/24/2024 3:19:36 AM		175000	Đã hoàn thành
7	Hoàng Hương Diễm	0	11/24/2024 3:29:01 AM		250000	Đã hoàn thành
8	Phạm Minh Bác Sĩ	0	11/24/2024 7:38:44 PM		300000	Đã hoàn thành
9	Phạm Minh Bác Sĩ	0	2/28/2025 11:46:11 PM		300000	Đã hoàn thành
10	Bác	0	3/1/2025 12:05:20 AM		100000	Đã hoàn thành
11	Bác	0	3/1/2025 12:25:53 AM		100000	Đã hoàn thành
12	Phạm Minh Bác Sĩ	0		Bạn có chắc chắn muốn hủy lịch này không?	300000	Đã hoàn thành
13	Hoàng Hương Diễm	0		<input type="button" value="Cancel"/> <input type="button" value="OK"/>	250000	Đã hoàn thành
14	Hoàng Hương Diễm	0	3/2/2025 10:19:01 AM		250000	Đã hoàn thành
15	Phạm Minh Bác Sĩ	7	3/13/2025 12:45:06 PM			Đã xác nhận
16	Phạm Minh Bác Sĩ	8	3/13/2025 12:45:46 PM			Đã xác nhận
17	Phạm Minh Bác Sĩ	9	3/13/2025 12:48:07 PM			Đã hủy
18	Phạm Minh Bác Sĩ	10	3/13/2025 1:00:29 PM			Đã xác nhận
19	Phạm Minh Bác Sĩ	11	3/13/2025 1:03:09 PM			Đã xác nhận
20	Phạm Minh Bác Sĩ	12	3/13/2025 1:07:36 PM			Đã hủy
21	Phạm Minh Bác Sĩ	13	3/13/2025 1:09:35 PM			<input type="button" value="Hủy lịch"/>
22	Phạm Minh Bác Sĩ	14	3/13/2025 1:12:57 PM			<input type="button" value="Hủy lịch"/>

CHƯƠNG TRÌNH THỰC NGHIỆM

6.1.16. Giao diện [Dự đoán bệnh]

The screenshot displays two side-by-side instances of a web-based AI symptom checker. Both instances have a similar layout with a header, navigation bar, and a main content area for input and results.

Header: Dành cho bác sĩ (For doctors) in the top right corner. The top left features the VOV BacSi24 logo.

Navigation Bar: Includes links for Trang chủ (Home), Cấp cứu (Emergency), Chuyên khoa (Specialties), Hướng dẫn sử dụng (Usage guide), Về VOVBacSi24 (About VOVBacSi24), and Chuyên trang sức khỏe (Health specialty page).

Main Content Area - Top Instance:

- Title:** Ứng dụng Dự đoán bệnh từ triệu chứng (Application for Predicting Disease from Symptoms).
- Text:** Nhập mô tả triệu chứng bạn đang gặp phải để hệ thống AI gợi ý các bệnh lý có thể liên quan.
- Note:** Lưu ý: Kết quả chỉ mang tính chất tham khảo.
- Input Field:** Mô tả triệu chứng của bạn: (sốt, đau rát cổ họng, ho nhiều đờm)
- Button:** Dự đoán bệnh (Predict disease).
- Feedback:** Bạn nên quan tâm đến sức khỏe hơn, theo dõi thêm các triệu chứng. (You should pay attention to your health, monitor additional symptoms.)
- Section:** Kết quả dự đoán (Prediction results).
- Note:** Lưu ý: Kết quả chỉ mang tính chất tham khảo.
- Input Field:** Mô tả triệu chứng của bạn: (sốt, đau rát cổ họng, ho nhiều đờm)
- Button:** Dự đoán bệnh (Predict disease).

Main Content Area - Bottom Instance:

- Feedback:** Bạn nên quan tâm đến sức khỏe hơn, theo dõi thêm các triệu chứng. (You should pay attention to your health, monitor additional symptoms.)
- Section:** Kết quả dự đoán (Prediction results).
- Result Card 1:** Khả năng cao: viêm họng (High probability: Throat inflammation) - Độ tin cậy: 54.4% (Confidence: 54.4%).
- Result Card 2:** Khả năng thấp: viêm phổi (Low probability: Pneumonia) - Độ tin cậy: 19.7% (Confidence: 19.7%).

6.2. Chi tiết các giao diện bác sĩ

6.2.1. Giao diện [đăng nhập]



VOV Doctor24

[Đăng nhập / Login](#) [Đăng ký / Register](#)

Số điện thoại

Mật khẩu / Password

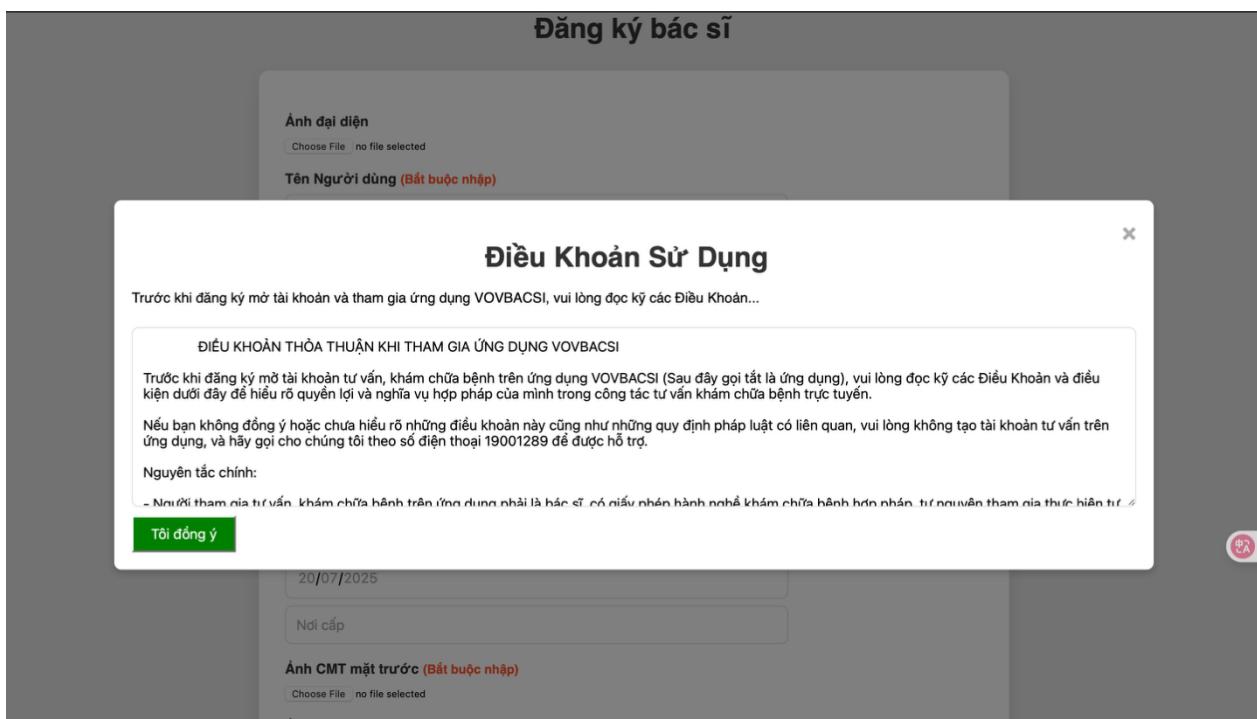
[Ghi nhớ tài khoản / Remember me](#) [Quên mật khẩu / Forgot password](#)

[ĐĂNG NHẬP / LOGIN](#)

Hoặc đăng nhập bằng / Or login by [Google](#)

Chưa có tài khoản? [Đăng ký ngay](#)
Didn't have account? [Register](#)

6.2.2. Giao diện [đăng ký bác sĩ]



Đăng ký bác sĩ

Ảnh đại diện
 Choose File no file selected

Tên Người dùng (Bắt buộc nhập)

Điều Khoản Sử Dụng

Trước khi đăng ký mở tài khoản và tham gia ứng dụng VOVBACSI, vui lòng đọc kỹ các Điều Khoản...

ĐIỀU KHOẢN THỎA THUẬN KHI THAM GIA ỨNG DỤNG VOVBACSI

Trước khi đăng ký mở tài khoản tư vấn, khám chữa bệnh trên ứng dụng VOVBACSI (Sau đây gọi tắt là ứng dụng), vui lòng đọc kỹ các Điều Khoản và điều kiện dưới đây để hiểu rõ quyền lợi và nghĩa vụ hợp pháp của mình trong công tác tư vấn khám chữa bệnh trực tuyến.

Nếu bạn không đồng ý hoặc chưa hiểu rõ những điều khoản này cũng như những quy định pháp luật có liên quan, vui lòng không tạo tài khoản tư vấn trên ứng dụng, và hãy gọi cho chúng tôi theo số điện thoại 19001289 để được hỗ trợ.

Nguyên tắc chính:

- Người tham gia tư vấn, khám chữa bệnh trên ứng dụng phải là bác sĩ có giấy phép hành nghề khám chữa bệnh hợp pháp, từ người tham gia thực hiện tư vấn.

[Tôi đồng ý](#)

20/07/2025

Nơi cấp

Ảnh CMND mặt trước (Bắt buộc nhập)
 Choose File no file selected

Ảnh CMND mặt sau (Để lựa chọn)

CHƯƠNG TRÌNH THỰC NGHIỆM

Đăng ký bác sĩ

Ảnh đại diện
 Choose File no file selected

Tên Người dùng (Bắt buộc nhập)

Mật khẩu (Bắt buộc nhập)

Nhập lại mật khẩu (Bắt buộc nhập)

Năm sinh (Bắt buộc nhập)

Số CMT (Bắt buộc nhập)

Ảnh CMT mặt trước (Bắt buộc nhập)
 Choose File no file selected

Ảnh CMT mặt sau (Bắt buộc nhập)
 Choose File no file selected

Giới tính (Bắt buộc nhập)

Email (Bắt buộc nhập)

Số điện thoại (Bắt buộc nhập)

Địa chỉ

Năm kinh nghiệm (Bắt buộc nhập)

Nơi công tác (Bắt buộc chọn)

Khoa (Bắt buộc chọn)

Chuyên khoa (Bắt buộc chọn)

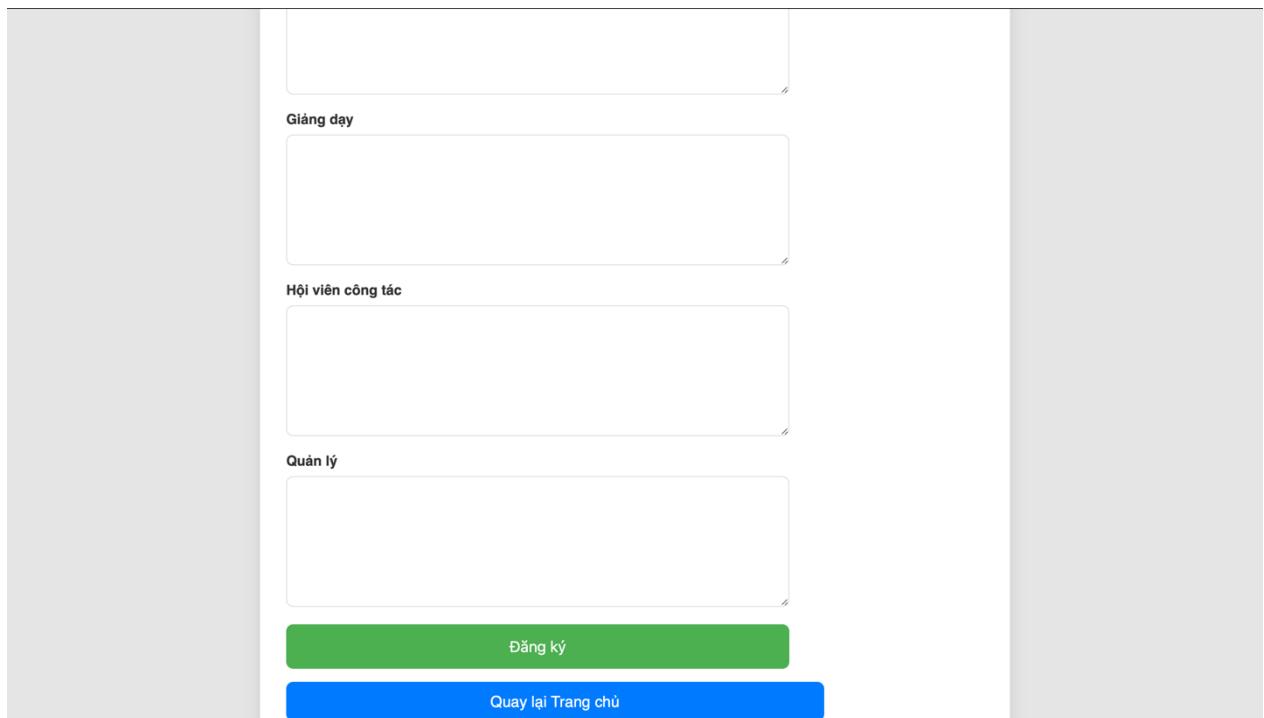
Số chứng chỉ hành nghề (Bắt buộc nhập)

Ngày tháng cấp (Bắt buộc nhập)

CHƯƠNG TRÌNH THỰC NGHIỆM

<p>Ngày tháng cấp (Bắt buộc nhập) 24/11/2024</p> <p>Nơi cấp (Bắt buộc nhập) <input type="text"/></p> <p>Ảnh chứng chỉ hành nghề 1 (Bắt buộc nhập) Choose File no file selected</p> <p>Ảnh chứng chỉ hành nghề 2 (Bắt buộc nhập) Choose File no file selected</p> <p>Học vị -- Hãy chọn học vị --</p> <p>Học hàm -- Hãy chọn học hàm --</p> <p>Chuyên môn (Bắt buộc nhập) -- Hãy chọn chuyên môn --</p> <p>Các bệnh điều trị (Bắt buộc nhập) <input type="text"/></p> <p>Giới thiệu (Bắt buộc nhập) <input type="text"/></p>	<p>Thông tin bằng cấp <input type="text"/></p> <p>Quá trình Học <input type="text"/></p> <p>Quá trình công tác <input type="text"/></p> <p>Khám chữa bệnh <input type="text"/></p>
--	--

CHƯƠNG TRÌNH THỰC NGHIỆM



Giảng dạy

Hội viên công tác

Quản lý

Đăng ký

Quay lại Trang chủ

CHƯƠNG TRÌNH THỰC NGHIỆM

6.2.3. Giao diện [Trang chủ/ thống kê]

The screenshots illustrate the 'Bacsi24' application's user interface, specifically the main dashboard and a statistical report page.

Main Dashboard (Top Screenshot):

- Header:** Chào đêm khuya Bác Sĩ: Phạm Minh | 10:40:05 CH | Thứ Bảy, 26 07 2025
- Left Sidebar:** Dashboard, Đã hoàn thành, Đã bị hủy, Thông báo, Quản lý cá nhân, Thống kê.
- Statistics:** Tổng thu nhập **5.150.000 VNĐ**, Tổng chờ khám **1**, Tổng đang khám **1**.
- Pending Appointments (Danh sách chờ khám):**

STT	Tên bệnh nhân	Thời gian đặt lịch	Giờ khám	Trạng thái	Chi tiết	Thao tác
1	Đào Huy Hoàng	24/07/2025	23:07:39	Đang chờ khám	Hỗ trợ	Khám Hủy
- Upcoming Appointments (Lịch khám):**

STT	Tên bệnh nhân	Thời gian đặt lịch	Trạng thái	Chi tiết	Thao tác
1	Lương Trung Đạt	28/11/2024 12:00:00 SA	Đang khám	Hỗ trợ	Hoàn thành Hủy

Statistical Report (Bottom Screenshot):

- Header:** Chào đêm khuya Bác Sĩ: Phạm Minh | 10:40:57 CH | Thứ Bảy, 26 07 2025
- Left Sidebar:** Dashboard, Đã hoàn thành, Đã bị hủy, Thông báo, Quản lý cá nhân, Thống kê.
- Statistics:** Tổng doanh thu **4.360.000 VNĐ**, Tổng thu nhập **5.150.000 VNĐ**.
- Summary Metrics:** Đã hoàn thành **11**, Đã hủy **24**, Tổng doanh thu **3.600.000 VNĐ**.
- Filter Options:** Chọn Năm (2025), Chọn loại biểu đồ (Biểu đồ đường), Thống kê.
- Line Chart:** Số Lượng Cuộc Hẹn (Number of Appointments) over Months (Tháng 1 to Tháng 12). The chart shows a sharp peak in March (Tháng 3) reaching approximately 28 appointments, followed by a low point in April (Tháng 4) and another smaller peak in July (Tháng 7) reaching about 6 appointments.

CHƯƠNG TRÌNH THỰC NGHIỆM

Chào đón khuya Bác Sĩ Phạm Minh | 10:44:15 CH | Thứ Bảy, 26/07/2025

Bacsit24 Bác Sĩ

Tổng doanh thu
4.360.000 VNĐ

Tổng thu nhập
5.150.000 VNĐ

Đã hoàn thành
11

Đã hủy
24

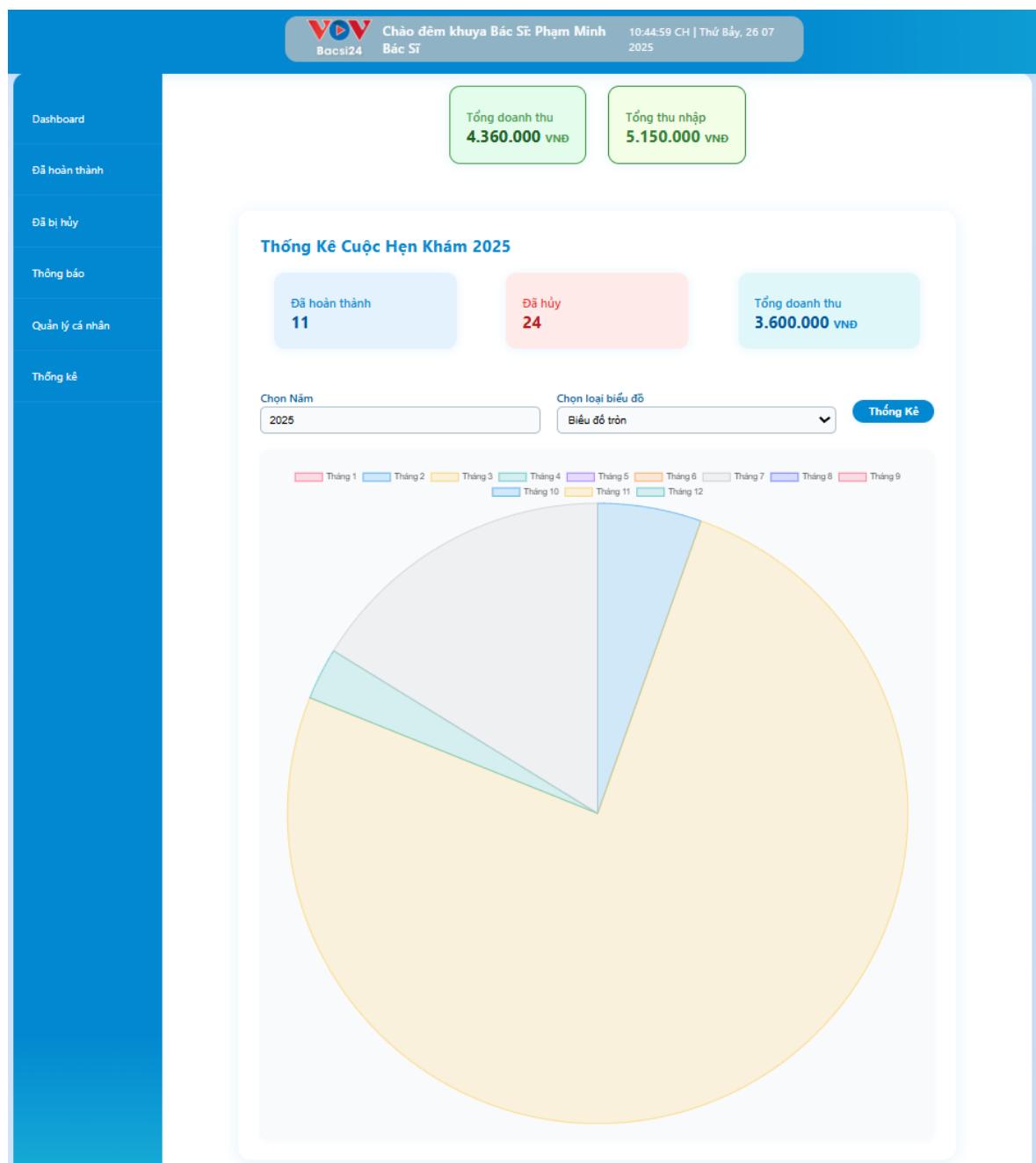
Tổng doanh thu
3.600.000 VNĐ

Thống Kê Cuộc Hẹn Khám 2025

Chọn Năm: 2025 | Chọn loại biểu đồ: Biểu đồ cột | Thống Kê

Tháng	Số Lượng Cuộc Hẹn
Tháng 2	2
Tháng 3	28
Tháng 4	1
Tháng 5	0
Tháng 6	0
Tháng 7	6
Tháng 8	0
Tháng 9	0
Tháng 10	0
Tháng 11	0
Tháng 12	0

CHƯƠNG TRÌNH THỰC NGHIỆM



CHƯƠNG TRÌNH THỰC NGHIỆM

6.2.4. Giao diện [Hồ sơ bệnh nhân]

STT	Tên bệnh nhân	Bác sĩ	Địa chỉ	Giới tính				
2	Hồ sơ điều trị bệnh theo dõi sức khỏe cho Trần Thị Bệnh Nhân Cập Nhật	Trần Thị Bệnh Nhân Cập Nhật	0909876543	456 Trường Chinh&44; Quận Tân Bình&44; TP. HCM	Nữ			xóa sửa
11	gdfgfdsgd	Lương Tiến Đạt	0707456456	123 Phạm Văn Đồng&44; Thủ Đức	Nam			xóa sửa
11	gdfgfdsgd	Lương Tiến Đạt	0707456456	123 Phạm Văn Đồng&44; Thủ Đức	Nam			xóa sửa
24	dsadsa	Lương Tiến Đạt	0707456456	123 Phạm Văn Đồng&44; Thủ Đức	Nam			xóa sửa

6.2.5. Giao diện [lịch hẹn khám]

STT	Tên bệnh nhân	Thời gian đặt lịch	Giờ khám	Trạng thái	Chi tiết	Thao tác
1	Đào Huy Hoàng	24/07/2025	23:07:39	Đang chờ khám	Hỗ trợ	Khám Hủy

STT	Tên bệnh nhân	Thời gian đặt lịch	Trạng thái	Chi tiết	Thao tác
1	Lương Trung Đạt	28/11/2004 12:00:00 SA	Đang khám	Hỗ trợ	Hoàn thành Hủy

CHƯƠNG TRÌNH THỰC NGHIỆM

6.2.6. Giao diện [đã hoàn thành khám]

The screenshot shows the 'Completed Exams' section of the application. At the top right, it displays 'Chào đêm khuya Bác Sĩ: Phạm Minh' and the date '10:50:13 CH | Thứ Bảy, 26/07/2025'. On the left sidebar, there are navigation items: Dashboard, Đã hoàn thành (Completed), Đã bị hủy (Cancelled), Thông báo (Notifications), Quản lý cá nhân (Personal Management), and Thông kê (Statistics). The main content area shows a summary box with 'Số lượng lịch đã hoàn thành: 15' and 'Tổng Thu nhập 5.150.000 VNĐ'. Below this is a table titled 'Lịch khám đã hoàn thành' with columns: STT, Tên bệnh nhân, Thời gian đặt lịch, Giờ khám, Trạng thái, Chi tiết, and Số tiền thanh toán. The table lists 7 completed exams for patient 'Lương Trung Đạt' at time 4, each costing 300,000 VNĐ.

STT	Tên bệnh nhân	Thời gian đặt lịch	Giờ khám	Trạng thái	Chi tiết	Số tiền thanh toán
1	Nguyễn Anh Huy	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	300.000 VNĐ
2	asdfgh	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	250.000 VNĐ
3	Lương Trung Đạt	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	210.000 VNĐ
4	Lương Trung Đạt	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	300.000 VNĐ
5	asdfgh	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	300.000 VNĐ
6	Lương Trung Đạt	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	300.000 VNĐ
7	Lương Trung Đạt	4	Đã hoàn thành	Đã hoàn thành	Hỗ trợ	300.000 VNĐ

6.2.7. Giao diện [đã hủy khám]

The screenshot shows the 'Cancelled Exams' section of the application. At the top right, it displays 'Chào đêm khuya Bác Sĩ: Phạm Minh' and the date '10:51:47 CH | Thứ Bảy, 26/07/2025'. The left sidebar has the same navigation items as the previous section. The main content area shows a summary box with 'Tổng lượt hủy: 24'. Below this is a table titled 'Lịch khám đã bị hủy' with columns: STT, Tên bệnh nhân, Thời gian đặt lịch, Giờ khám, Trạng thái, and Mô tả bệnh. The table lists 7 cancelled exams for patient 'Lương Trung Đạt' at time 3, all with the status 'Đã hủy' and the note 'Đau đầu chóng mặt'.

STT	Tên bệnh nhân	Thời gian đặt lịch	Giờ khám	Trạng thái	Mô tả bệnh
1	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt
2	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt
3	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt
4	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt
5	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt
6	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt
7	Lương Trung Đạt	3	Đã hủy	Đã hủy	Đau đầu chóng mặt

6.2.8. Giao diện [thông báo]

#	TIÊU ĐỀ	THỜI GIAN	THAO TÁC
21	4	2025-03-30 09:18:03	Xem Chi Tiết
19	3	2025-03-30 09:17:40	Xem Chi Tiết

6.3. Chi tiết các giao diện admin

6.3.1. Giao diện [sửa bài viết]

CHƯƠNG TRÌNH THỰC NGHIỆM

6.3.2. Giao diện [sửa khoa bệnh]

6.3.3. Giao diện [thêm bài viết]

CHƯƠNG TRÌNH THỰC NGHIỆM

6.3.4. Giao diện [thêm khoa bệnh]

Danh mục Khoa Bệnh

Tên Khoa Bệnh:

Mô Tả:

Thêm Khoa Bệnh

Danh Sách Khoa Bệnh

Mã Khoa Bệnh	Tên Khoa Bệnh	Mô Tả	Thao Tác
1	Nội khoa	Chuyên khám và điều trị các bệnh lý nội khoa mới.	Xóa Sửa
2	Nhi khoa	Khám và điều trị các bệnh về trẻ em	Xóa Sửa

6.3.5. Giao diện [xác nhận đăng ký bác sĩ]

Danh Sách Đơn Đăng Ký Bác Sĩ

Mã Bác Sĩ	Tên Bác Sĩ	Chuyên Khoa	Giới Tính	Hành Động
57	hee		Nữ	<input checked="" type="checkbox"/> Duyệt <input type="checkbox"/> Từ Chối

CHƯƠNG TRÌNH THỰC NGHIỆM

6.3.6. Giao diện [xoá bài viết]

The screenshot shows a software interface with a sidebar on the left containing icons for patient files, doctor files, and various management modules like appointment management, information management, call history, statistics, and feedback. The main area is titled 'Danh Sách Khoa Bệnh' (List of Medical Specialties) and displays a table with three entries:

Mã Khoa Bệnh	Tên Khoa Bệnh	Mô Tả	Thao Tác
1	Nội khoa	Chuyên khám và điều trị các bệnh lý nội khoa mới.	Xóa Sửa
2	Nhi khoa	Khám và điều trị các bệnh về trẻ em	Xóa Sửa
3	Tai mũi họng	Chuyên về các bệnh tai&44; mũi và họng	Xóa Sửa

6.3.7. Giao diện [xóa khoa bệnh]

The screenshot shows a software interface with a sidebar on the left containing icons for appointment management, information management, call history, statistics, and feedback. The main area is titled 'Danh Sách Khoa Bệnh' (List of Medical Specialties) and displays a table with three entries:

Mã Khoa Bệnh	Tên Khoa Bệnh	Mô Tả	Thao Tác
1	Nội khoa	Chuyên khám và điều trị các bệnh lý nội khoa mới.	Xóa Sửa
2	Nhi khoa	Khám và điều trị các bệnh về trẻ em	Xóa Sửa
3	Tai mũi họng	Chuyên về các bệnh tai&44; mũi và họng	Xóa Sửa

CHƯƠNG TRÌNH THỰC NGHIỆM

6.3.8. Giao diện [đánh giá phản hồi]

Mã Đánh Giá	Tên Bệnh Nhân	Tên Bác Sĩ	Thời Gian Hẹn	Nội Dung Đánh Giá	Ngày Đánh Giá	Số Sao	Thao Tác
3	Nguyễn Anh Huy	Phạm Minh Bác Sĩ	9/15/2023 2:00:00 PM	Dịch vụ rất tốt&44; bác sĩ nhiệt tình và chu...	9/20/2024 12:00:00 AM	★★★★★	Xóa
4		Phạm Minh Bác Sĩ	9/16/2023 9:00:00 AM	Bác sĩ tư vấn kỹ càng&44; nhưng thời gian...	9/21/2024 12:00:00 AM	★★★★★	Xóa
5	Lương Trung Đạt	Hoàng Hương Diễm	11/23/2024 4:40:33 AM	Dịch vụ rất tốt.	1/20/2024 12:00:00 AM	★★★★★	Xóa
6	Lương Trung Đạt	Phạm Minh Bác Sĩ	11/23/2024 11:38:40 PM	thời gian chờ hơi lâu.	9/21/2024 12:00:00 AM	★★★	Xóa

6.3.9. Giao diện [quản lý danh mục]

Mã Bài Viết	Tiêu Đề Bài Viết	Link Youtube	Ngày Đăng	Mã Loại Bài Viết	Lượt Xem	Thao Tác
1	SỨC KHOẺ TỔNG QUÁ	https://www.youtube.com/watch?v=s13Uo5VfQ9I	3/9/2025 12:00:00 AM	3	150	Xóa Sửa
13	Đồng hành cùng ba mẹ châm sóc và nâng niu những mầm xanh!	https://www.youtube.com/watch?v=s13Uo5VfQ9I	11/21/2024 12:00:00 AM	4	2	Xóa Sửa
20	GIỚI THIỆU KÊNH VOV – FM89MHz		11/25/2024 12:00:00 AM	1	0	Xóa Sửa
21	QUY CHÉ HOẠT ĐỘNG		11/25/2024 12:00:00 AM	1	0	Xóa Sửa

CHƯƠNG TRÌNH THỰC NGHIỆM

6.3.10. Giao diện [danh sách lịch khám]

Mã Lịch Khám	Tên Bệnh Nhân	Tên Bác Sĩ	Thời Gian Hẹn	Thời Gian BD	Thời Gian KT	Trạng Thái	Số tiền khám
1	Nguyễn Anh Huy	Phạm Minh Bác Sĩ	15/09/2023 2:00:00 CH	15/09/2023 2:15:00 CH	15/09/2023 2:45:00 CH	Đã hoàn thành	300000
2	asdfgh	Phạm Minh Bác Sĩ	16/09/2023 9:00:00 SA	16/09/2023 9:15:00 SA	16/09/2023 9:45:00 SA	Đã hoàn thành	250000
10	Lương Trung Đạt	Hoàng Hương Diễm	23/11/2024 4:40:33 SA			Đã hoàn thành	175000
12	Lương Trung Đạt	Phạm Minh Bác Sĩ	23/11/2024 11:38:40 CH			Đã hoàn thành	210000
13	Lương Trung Đạt	Hoàng Hương Diễm	24/11/2024 2:49:41 SA			Đã hủy	
15	Lương Trung Đạt	Hoàng Hương Diễm	24/11/2024 3:17:08 SA			Đã hoàn thành	175000
16	Lương Trung Đạt	Hoàng Hương Diễm	24/11/2024 3:19:36 SA			Đã hoàn thành	175000
17	Lương Trung Đạt	Hoàng Hương Diễm	24/11/2024 3:29:01 SA			Đã hoàn thành	250000
19	Lương Trung Đạt	Phạm Minh Bác Sĩ	24/11/2024 7:38:44 CH			Đã hoàn thành	300000

6.3.11. Giao diện [quản lý thanh toán]

Tổng số tiền thanh toán
6.185.000 VND
Tổng cộng tất cả giao dịch trong danh sách

Mã Thanh Toán	Người Thanh Toán	Số Tiền	Thời Gian Thanh Toán	Lý Do Hoàn Phí	Ngày Hoàn Phí	Thao Tác
4	Nguyễn Anh Huy	200000 VND	20/09/2024 12:00:00 SA	Lỗi	22/07/2025 3:48:56 SA	Hoàn phí khám
9	Nguyễn Anh Huy	200000 VND	20/09/2024 12:00:00 SA			Hoàn phí khám
10	asdfgh	150000 VND	21/09/2024 12:00:00 SA			Hoàn phí khám
11	Lương Trung Đạt	250000 VND	02/03/2025 12:00:00 SA	Lỗi luôn	22/07/2025 4:26:16 SA	Hoàn phí khám
18	Lương Trung Đạt	175000 VND	22/07/2025 12:00:00 SA			Hoàn phí khám

CHƯƠNG TRÌNH THỰC NGHIỆM

6.3.12. Giao diện [quản lý thông báo]

The screenshot shows a left sidebar with a teal header 'Bacsit24' and a list of navigation items:

- Xác nhận đăng ký bác sĩ
- Quản lý hồ sơ y tế
- Quản lý danh mục
- Xem danh sách lịch khám
- Đánh giá và phản hồi của bệnh nhân
- Quản lý thanh toán
- Quản lý thông báo
- Thống kê

The main area has two sections:

- Gửi Thông Báo**: A form with fields for 'Tiêu Đề' (Subject), 'Nội Dung' (Content), 'Thời Gian Thông Báo' (Reporting Time), and 'Mã Người Dùng' (User ID). A dropdown shows '1-Nguyễn Anh Huy'. A green 'Gửi Thông Báo' button is at the bottom.
- Danh Sách Thông Báo Đã Gửi**: A table listing sent notifications. The columns are: Mã Thông Báo (Notification ID), Tiêu Đề (Subject), Nội Dung (Content), Thời Gian TB (Reporting Time), Mã Người Dùng (User ID), Tên Người Dùng (User Name), Trạng Thái Đọc (Read Status), and Thao Tác (Action). Two entries are shown:

Mã Thông Báo	Tiêu Đề	Nội Dung	Thời Gian TB	Mã Người Dùng	Tên Người Dùng	Trạng Thái Đọc	Thao Tác
21	4	test 4	3/30/2025 9:18:03 AM	3	Phạm Minh Bác Sĩ	False	Xóa
19	3	thầy chưa	3/30/2025 9:17:40 AM	3	Phạm Minh Bác Sĩ	False	Xóa

6.3.13. Giao diện [thống kê]



CHƯƠNG TRÌNH THỰC NGHIỆM

VOV
Bacsit24

- Xác nhận đăng ký bác sĩ
- Quản lý hồ sơ y tế
- Quản lý danh mục
- Xem danh sách lịch khám
- Đánh giá và phản hồi của bệnh nhân
- Quản lý thanh toán
- Quản lý thông báo
- Thông kê

Tổng người dùng
21

Tổng bác sĩ
13

Tổng doanh thu
6.185.000 VNĐ

Tổng lịch hẹn hoàn thành
25

Tổng thu nhập (30%)
1.855.500 VNĐ

Biểu đồ doanh thu & lịch hẹn hoàn thành theo tháng

Năm: 2025 Biểu đồ: Cột (Bar)

Tháng	Lịch hoàn thành	Doanh thu VNĐ
1	0	0
2	3	250.000
3	10	2.500.000
4	1	50.000
7	3	250.000

CHƯƠNG 7: KẾT LUẬN

7.1. Kết quả đạt được và chưa đạt được

7.1.1. Kết quả đạt được

Trong đồ án phần mềm về dự án đặt lịch khám VOV BÁC SĨ chúng tôi đã đạt được các kết quả sau:

- Xây dựng lại giao diện của vovbacsi24.com gần giống với vinmec.com
- Đăng nhập/ đăng ký người dùng bằng OTP của firebase
- Đăng nhập/ đăng ký bác sĩ
- Hiển thị danh sách và xem chi tiết các thông tin trong cơ sở dữ liệu
- Thêm xóa sửa các thông tin trong
- Tìm kiếm và lọc thông tin bác sĩ
- Sử dụng CK editor để thêm các bài viết để trông đẹp mắt hơn
- Đặt lịch khám
- Cập nhật các trang thái lịch khám

7.1.2. Kết quả chưa đạt được

Trong đồ án phần mềm về dự án đặt lịch khám VOV BÁC SĨ còn có nhiều khuyết điểm vẫn chưa giải quyết được:

- Ràng buộc dữ liệu trong các form nhập chưa được thực hiện đầy đủ, dẫn đến nguy cơ nhập thiếu hoặc sai định dạng.
- Chức năng tuy nhiều nhưng một số còn sơ sài, chưa có xử lý kiểm tra nghiệp vụ đầy đủ.
- Giao diện dành cho bác sĩ và quản trị viên còn đơn giản, thiếu tính thẩm mỹ và trải nghiệm người dùng chưa tối ưu

7.2. Hướng phát triển mở rộng ứng dụng trong tương lai

Trong tương lai, nhóm mong muốn tiếp tục cải thiện và mở rộng hệ thống theo các hướng sau:

- **Tăng cường bảo mật** hệ thống: bổ sung cơ chế phân quyền chi tiết giữa người dùng, bác sĩ và quản trị viên; áp dụng các kỹ thuật bảo vệ thông tin như mã hóa, CSRF/XSS protection.

- **Tối ưu ràng buộc dữ liệu và kiểm tra đầu vào:** đảm bảo dữ liệu hợp lệ ngay từ giao diện nhập, kết hợp với xác thực phía server để hạn chế lỗi nghiệp vụ.
- **Nâng cấp giao diện người dùng:** sử dụng các thư viện UI/UX hiện đại hơn như Tailwind CSS hoặc React component library để làm cho hệ thống thân thiện và trực quan hơn, đặc biệt là với bác sĩ và quản trị viên.
- **Triển khai chức năng thanh toán trực tuyến:** tích hợp cổng thanh toán (VNPay, Momo, v.v.) để người dùng có thể thanh toán phí khám bệnh nhanh chóng và tiện lợi.
- **Xây dựng hệ thống phản hồi và đánh giá:** để người dùng có thể đánh giá bác sĩ sau khi khám, từ đó tạo động lực cải thiện chất lượng dịch vụ.
- **Tối ưu hiệu năng và khả năng mở rộng hệ thống:** sử dụng cache, tách các dịch vụ (microservice) nếu cần để đảm bảo hệ thống hoạt động ổn định khi có lượng truy cập lớn.

Triển khai hệ thống trên nền tảng cloud (Azure, AWS) để đảm bảo khả năng