

**BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI**



BÀI TẬP LỚN

**PHÁT TRIỂN ÚNG DỤNG CHO THIẾT BỊ DI ĐỘNG
ĐỀ TÀI: ÚNG DỤNG ĐẶT LỊCH KHÁM
BOOKINGCARE**

Giáo viên hướng dẫn: ThS. Kiều Tuấn Dũng

Sinh viên thực hiện:

STT	Mã sinh viên	Họ và tên	Lớp
1	2251061862	Hoàng Thị Phương	64CNTT1
2	2251061772	Đoàn Thị Ánh Hậu	64CNTT1
3	2251061771	Bùi Thị Hảo	64CNTT1
4	2251061764	Đoàn Thị Hà Giang	64CNTT1

Hà Nội, năm 2025

**BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI**



BÀI TẬP LỚN

**PHÁT TRIỂN ÚNG DỤNG CHO THIẾT BỊ DI ĐỘNG
ĐÈ TÀI: ÚNG DỤNG ĐẶT LỊCH KHÁM BOOKINGCARE**

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bằng Số	Bằng Chữ
1	2251061862	Hoàng Thị Phương	11/06/2004		
2	2251061772	Đoàn Thị Ánh Hậu	01/01/2004		
3	2251061771	Bùi Thị Hảo	20/11/2004		
4	2251061764	Đoàn Thị Hà Giang	02/08/2004		

CÁN BỘ CHẤM THI

Hà Nội, năm 2025

LỜI NÓI ĐẦU

Trong bối cảnh ngành y tế ngày càng phát triển và nhu cầu khám chữa bệnh của người dân ngày càng tăng, việc tìm kiếm một giải pháp công nghệ giúp tối ưu hóa quy trình đặt lịch khám bệnh trở nên vô cùng quan trọng. Tuy nhiên, việc đặt lịch khám bệnh truyền thống thường gặp phải nhiều bất cập như tồn thời gian, dễ gây nhầm lẫn và không thuận tiện cho người bệnh trong việc lựa chọn thời gian khám phù hợp.

Nhận thức được những vấn đề này, nhóm chúng tôi đã quyết định phát triển **Ứng dụng Đặt Lịch Khám - BOOKINGCARE**, một nền tảng giúp người dùng có thể dễ dàng tra cứu thông tin bác sĩ, các chuyên khoa và phòng khám, đồng thời đặt lịch khám trực tuyến một cách nhanh chóng và tiện lợi. Ứng dụng không chỉ giúp người bệnh tiết kiệm thời gian mà còn hỗ trợ các cơ sở y tế trong việc quản lý và tổ chức lịch khám hiệu quả, giảm thiểu tình trạng quá tải và sai sót trong quá trình quản lý lịch hẹn.

Dự án này được thực hiện trong khuôn khổ chương trình học của chúng tôi, nhằm ứng dụng các kiến thức đã học vào thực tế. Qua đó, chúng tôi mong muốn đóng góp một giải pháp công nghệ hữu ích, giúp nâng cao chất lượng dịch vụ y tế và mang lại trải nghiệm người dùng tốt hơn cho cả người bệnh và các cơ sở y tế.

Chúng tôi hy vọng rằng, **BOOKINGCARE** sẽ là một công cụ hữu ích, góp phần cải thiện quy trình đặt lịch khám, giảm thiểu thời gian chờ đợi và mang lại sự thuận tiện cho người bệnh trong việc tiếp cận dịch vụ chăm sóc sức khỏe.

MỤC LỤC

DANH MỤC HÌNH ẢNH	7
1. Thiết kế giao diện dành cho người dùng bệnh nhân	7
2. Thiết kế giao diện dành cho người dùng Bác sĩ	8
3. Thiết kế giao diện dành cho người dùng Cơ sở y tế.....	8
4. Thiết kế giao diện dành cho người dùng Admin.....	9
5. Màn hình giao diện Bệnh nhân.....	9
6. Màn hình giao diện Bác sĩ	10
7. Màn hình giao diện CSYT	10
8. Màn hình giao diện Admin.....	11
DANH MỤC BẢNG BIỂU.....	12
DANH MỤC CÁC TỪ VIẾT TẮT	12
Chương 1. TỔNG QUAN VỀ ĐỀ TÀI	13
1.1. Giới thiệu về đề tài.....	13
1.2. Mục tiêu của đề tài	13
1.3. Phạm vi của đề tài.....	14
a, Các chức năng chính.....	14
b, Yêu cầu phi chức năng	15
Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ	18
2.1. Kiến trúc hệ thống	18
2.1.1. Khái niệm kiến trúc hệ thống	18
2.1.2. Mô hình kiến trúc tổng thể	18
2.1.3. Các thành phần chính của hệ thống	19
2.1.4. Luồng hoạt động của hệ thống	19
2.1.5. Ưu điểm của kiến trúc hệ thống	20
2.2. Giới thiệu về Công nghệ phát triển.....	20

2.2.1. Ngôn ngữ lập trình.....	20
2.2.2. Nền tảng phát triển	20
2.2.3. Lý do lựa chọn công nghệ	20
Chương 3. XÂY DỰNG ỨNG DỤNG	22
3.1. Thiết kế Figma	22
3.1.1. Giao diện User (Bệnh nhân)	22
3.1.2 Giao diện Bác sĩ.....	28
3.1.3 Giao diện cơ sở y tế	33
3.1.4. Giao diện Admin	38
3.2. Thiết kế CSDL.....	41
3.2.1. Xác định các lớp	41
3.2.2. Mối quan hệ giữa các lớp	47
3.2.3. Biểu đồ lớp cho mô hình miền	47
3.2. Giao diện ứng dụng	49
3.2.1. Màn hình giao diện Người dùng (Bệnh nhân)	49
3.1.2. Màn hình giao diện Bác sĩ.....	55
3.1.3. Màn hình giao diện Cơ sở y tế	62
3.1.4. Màn hình giao diện Admin.....	68
3.4. Code minh họa các chức năng cốt lõi.....	72
3.4.1 Chức năng đăng ký	72
3.4.2 Chức năng đăng nhập , đăng xuất	79
3.4.3 Chức năng đặt lịch hẹn, xác nhận, hủy lịch hẹn	83
3.4.4 Chức năng tìm kiếm	87
3.4.5 Chức năng hiển thị các bác sĩ “Đang hoạt động”	93
3.4.6 Sửa thông tin cá nhân (Bệnh nhân, bác sĩ, Csyt).....	94
3.4.7 Chức năng tạo, xóa bài viết	101

3.4.8 Các chức năng thêm (chuyên khoa)	103
3.4.9 Các chức năng quản lý (bệnh nhân, csyt, bacsi, bài viết, chuyên khoa)	104
3.4.10 Tải ảnh lên Firebase Storage, copy ảnh vào bộ nhớ cache.....	110
3.4.11 Thông báo.....	114
3.4.12 File FirebaseHelper (hỗ trợ thêm sửa xóa trên firebase realtime).....	114
KẾT LUẬN	114
TÀI LIỆU THAM KHẢO.....	117

DANH MỤC HÌNH ẢNH

1. Thiết kế giao diện dành cho người dùng bệnh nhân

- 1.1. Giao diện Màn hình chính
- 1.2. Giao diện Đăng nhập
- 1.3. Giao diện Đăng ký
- 1.4. Giao diện tài khoản
- 1.5 Giao diện Hồ sơ y tế
- 1.6. Giao diện Khám chuyên khoa
- 1.7. Giao diện cơ sở y tế
- 1.8. Giao diện Thông tin bác sĩ
- 1.9. Giao diện Menu
- 1.10. Giao diện Chuyên khoa – Cơ xương khớp
- 1.26. Giao diện Thông tin Phiếu đặt lịch
- 1.27. Giao diện Thông tin lịch khám
- 1.28. Giao diện Điều khoản sử dụng
- 1.29. Giao diện Liên hệ hỗ trợ
- 1.30. Giao diện tìm kiếm
- 1.31. Giao diện Chuyên khoa – CSYT
- 1.32. Giao diện đặt lịch

2. Thiết kế giao diện dành cho người dùng Bác sĩ

- 1.32. Giao diện Thông tin các bác sĩ
- 1.33. Giao diện Trang chủ
- 1.34. Giao diện Thông tin chi tiết Cơ sở y tế
- 1.35. Giao diện xem lịch hẹn
- 1.36. Giao diện Chi tiết lịch hẹn
- 1.37. Giao diện Chi tiết lịch hẹn đã xác nhận
- 1.38. Giao diện Đăng bài viết
- 1.39. Giao diện Chọn ảnh/video
- 1.40. Giao diện Thông tin chi tiết Bác sĩ
- 1.41. Giao diện Danh sách bác sĩ
- 1.42. Giao diện Chuyên khoa khám
- 1.43. Giao diện Tìm kiếm
- 1.44. Giao diện sửa thông tin
- 1.45. Giao diện Trang cá nhân
- 1.46. Giao diện Menu

3. Thiết kế giao diện dành cho người dùng Cơ sở y tế

- 1.50. Giao diện Trang chủ
- 1.51. Giao diện Menu
- 1.52. Giao diện tìm kiếm
- 1.53. Giao diện Thông tin chi tiết bác sĩ
- 1.54. Giao diện Chuyên khoa khám
- 1.55. Giao diện Danh sách bác sĩ
- 1.56. Giao diện thông tin chi tiết CSYT
- 1.57. Giao diện sửa thông tin
- 1.59. Giao diện Thông tin CSYT
- 1.60. Giao diện Hiển thị danh sách chuyên khoa
- 1.61. Giao diện Thêm thông tin Chuyên khoa
- 1.62. Giao diện xem lịch hẹn
- 1.63. Giao diện thông tin đặt lịch
- 1.64. Giao diện Thông tin đặt lịch đã xác nhận

4. Thiết kế giao diện dành cho người dùng Admin

- 1.65. Giao diện Trang chủ
- 1.66. Giao diện Đăng nhập
- 1.67. Giao diện Hiển thị danh sách bác sĩ
- 1.68. Giao diện thông tin chi tiết bác sĩ
- 1.70. Giao diện Sửa thông tin cơ sở y tế
- 1.71. Giao diện Thêm thông tin chi tiết chuyên khoa
- 1.72. Giao diện Sửa thông tin chi tiết chuyên khoa
- 1.73. Giao diện Hiển thị danh sách chuyên khoa
- 1.74. Giao diện Hiển thị danh sách bác sĩ
- 1.75. Giao diện Hiển thị danh sách CSYT

5. Màn hình giao diện Bệnh nhân

- 2.1 Giao diện Trang chủ
- 2.2 Giao diện Đăng nhập
- 2.3. Giao diện Đăng ký
- 2.4. Giao diện tài khoản
- 2.5 Giao diện Hồ sơ y tế
- 2.6. Giao diện Khám chuyên khoa
- 2.7. Giao diện cơ sở y tế
- 2.8. Giao diện Thông tin bác sĩ
- 2.9. Giao diện Menu
- 2.10. Giao diện Chuyên khoa - Tiêu hóa
- 2.11. Giao diện Thông tin Phiếu đặt lịch với bác sĩ
- 2.12. Giao diện Thông tin đặt lịch với CSYT
- 2.13. Giao diện Thông tin lịch khám

2.14. Giao diện Điều khoản sử dụng

2.15. Giao diện Liên hệ hỗ trợ

2.16. Giao diện tìm kiếm

2.17. Giao diện Chuyên khoa - CSYT

2.18. Giao diện Đặt lịch

6. Màn hình giao diện Bác sĩ

1.32. Giao diện Thông tin các bác sĩ

1.33. Giao diện Trang chủ

1.34. Giao diện Thông tin chi tiết Cơ sở y tế

1.35. Giao diện xem lịch hẹn

1.36. Giao diện Chi tiết lịch hẹn

1.37. Giao diện Chi tiết lịch hẹn đã xác nhận

1.38. Giao diện Đăng bài viết

1.39. Giao diện Chọn ảnh/video

1.40. Giao diện Thông tin chi tiết Bác sĩ

1.41. Giao diện Danh sách bác sĩ

1.42. Giao diện Chuyên khoa khám

1.43. Giao diện Tìm kiếm

1.44. Giao diện sửa thông tin

1.45. Giao diện Trang cá nhân

1.46. Giao diện Menu

7. Màn hình giao diện CSYT

2.32. Giao diện Trang chủ

2.33. Giao diện Menu

2.34. Giao diện tìm kiếm bác sĩ

2.35. Giao diện Thông tin chi tiết bác sĩ

2.36. Giao diện Chuyên khoa khám

- 2.37. Giao diện Danh sách bác sĩ
- 2.38. Giao diện thông tin chi tiết CSYT
- 2.39. Giao diện sửa thông tin CSYT
- 2.40. Giao diện Thông tin chuyên khoa
- 2.41. Giao diện Thông tin CSYT
- 2.42. Giao diện Hiển thị danh sách chuyên khoa
- 2.43. Giao diện Thêm thông tin Chuyên khoa
- 2.44. Giao diện xem lịch hẹn
- 2.45. Giao diện thông tin đặt lịch

8. Màn hình giao diện Admin

- 2.46. Giao diện Trang chủ
- 2.47. Giao diện Đăng nhập
- 2.48. Giao diện Hiển thị danh sách bác sĩ
- 2.49. Giao diện thông tin chi tiết bác sĩ
- 2.50. Giao diện Sửa thông tin cơ sở y tế
- 2.51. Giao diện Thêm thông tin chi tiết chuyên khoa
- 2.52. Giao diện Sửa thông tin chi tiết chuyên khoa
- 2.53. Giao diện Hiển thị danh sách chuyên khoa
- 2.54. Giao diện Hiển thị danh sách bác sĩ
- 2.55. Giao diện Hiển thị danh sách CSYT
- 2.56. Giao diện quản lý bệnh nhân
- 2.57. Giao diện quản lý bài viết

DANH MỤC BẢNG BIỂU

Biểu đồ 1. Biểu đồ lớp

DANH MỤC CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	CSDL	Cơ sở dữ liệu
2	CSYT	Cơ sở y tế

Chương 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu về đề tài

Trong thời đại công nghệ số hiện nay, các dịch vụ y tế truyền thống đang dần được thay thế và nâng cao nhờ ứng dụng công nghệ thông tin. Việc đặt lịch khám bệnh, một trong những nhu cầu cơ bản và thiết yếu của người dân, hiện vẫn còn gặp phải nhiều khó khăn và bất tiện. Người bệnh thường phải trực tiếp gọi điện thoại hoặc đến bệnh viện để xếp lịch khám, điều này vừa mất thời gian, lại có thể dẫn đến sự cố nhầm lẫn hoặc trùng lịch.

Để giải quyết những vấn đề này, dự án **Ứng dụng Đặt Lịch Khám - BOOKINGCARE** được xây dựng với mục tiêu mang lại một giải pháp hiện đại, giúp người dùng có thể dễ dàng và nhanh chóng đặt lịch khám bệnh trực tuyến. Ứng dụng không chỉ giúp người bệnh tiết kiệm thời gian mà còn hỗ trợ các cơ sở y tế trong việc quản lý và tổ chức lịch khám hiệu quả hơn.

BOOKINGCARE cung cấp một nền tảng kết nối trực tiếp giữa người bệnh và các bác sĩ, cho phép người dùng tra cứu thông tin bác sĩ, chuyên khoa, phòng khám và các lịch hẹn. Thông qua ứng dụng, người bệnh có thể lựa chọn thời gian khám, nhận thông báo về lịch hẹn và các thay đổi liên quan, từ đó giúp quá trình khám chữa bệnh trở nên thuận tiện và hiệu quả hơn.

Dự án này không chỉ nhằm cải thiện chất lượng dịch vụ y tế mà còn là cơ hội để sinh viên áp dụng những kiến thức lý thuyết đã học vào thực tế, đồng thời khám phá và phát triển các kỹ năng lập trình, thiết kế giao diện người dùng và quản lý dữ liệu.

BOOKINGCARE được kỳ vọng sẽ là một công cụ hữu ích trong việc nâng cao trải nghiệm của người dùng và hỗ trợ các cơ sở y tế trong việc quản lý lịch khám.

1.2. Mục tiêu của đề tài

Mục tiêu của đề tài là nghiên cứu và phát triển một hệ thống ứng dụng di động hỗ trợ người dùng trong việc đặt lịch khám bệnh trực tuyến một cách nhanh chóng, thuận tiện và hiệu quả. Ứng dụng nhằm giải quyết các vấn đề tồn tại trong quy trình khám chữa bệnh truyền thống như: thời gian chờ đợi kéo dài, khó khăn trong việc tiếp cận thông tin bác sĩ, và thiếu chủ động trong việc sắp xếp lịch khám phù hợp với nhu cầu cá nhân.

Cụ thể, đề tài hướng đến việc đạt được các mục tiêu sau:

- Xây dựng hệ thống tra cứu thông tin y tế:** Cung cấp đầy đủ, chính xác và cập nhật thông tin về các cơ sở y tế, bác sĩ, chuyên khoa, lịch làm việc và chi phí khám chữa bệnh.
- Phát triển chức năng đặt lịch khám trực tuyến:** Cho phép người dùng lựa chọn bác sĩ, khung giờ phù hợp và thực hiện đặt lịch trực tiếp qua ứng dụng.
- Thiết kế hệ thống quản lý lịch hẹn cá nhân:** Hỗ trợ người dùng quản lý lịch sử khám bệnh, theo dõi các lịch hẹn sắp tới, đồng thời có khả năng hủy hoặc thay đổi lịch khi cần thiết.
- Tích hợp chức năng thông báo tự động:** Gửi thông báo nhắc lịch khám đến người dùng nhằm đảm bảo lịch hẹn được tuân thủ đúng thời gian.

Qua việc hiện thực hóa các mục tiêu trên, đề tài hướng đến việc góp phần thúc đẩy chuyển đổi số trong lĩnh vực y tế, đồng thời nâng cao hiệu quả phục vụ người bệnh trong bối cảnh phát triển công nghệ thông tin hiện nay.

1.3. Phạm vi của đề tài

a, Các chức năng chính

Ứng dụng được xây dựng theo mô hình phân quyền, với ba nhóm người dùng chính: **Bệnh nhân**, **Bác sĩ**, và **Quản trị viên (Admin)**. Mỗi nhóm có các chức năng riêng biệt nhằm đảm bảo quá trình đặt lịch và quản lý khám bệnh diễn ra hiệu quả. Mục tiêu của ứng dụng hướng đến các chức năng chính sau:

*Đối với bệnh nhân

- Đăng ký/Đăng nhập tài khoản qua Firebase Authentication.
- Cập nhật thông tin cá nhân (họ tên, giới tính, năm sinh, địa chỉ...).
- Đặt lịch khám:
 - Chọn chuyên khoa, bác sĩ, ngày và giờ khám.
 - Nhận xác nhận đặt lịch và thông báo nhắc hẹn.
- Quản lý lịch hẹn:
 - Xem danh sách lịch khám đã đặt.

- Hủy hoặc thay đổi lịch nếu cần (trước thời gian hẹn).
- Xem thông tin bác sĩ: Hồ sơ, chuyên môn, lịch làm việc.
- Xem lịch sử khám bệnh: Theo dõi quá trình khám chữa bệnh.
- Nhận thông báo từ bác sĩ/phòng khám (qua Firebase Cloud Messaging).

*Đối với Bác sĩ

- Đăng nhập hệ thống bằng tài khoản do admin cấp hoặc xác thực Firebase.
- Xem danh sách lịch khám của mình theo ngày/giờ.
- Cập nhật trạng thái khám (chờ khám, đã khám, hủy).
- Xem hồ sơ bệnh nhân trước khi khám.
- Nhắn tin hoặc gửi ghi chú cho bệnh nhân (nếu có hỗ trợ chức năng tư vấn).
- Xem và cập nhật lịch sử khám của từng bệnh nhân (nếu được phân quyền).
- Cập nhật thông tin cá nhân và lịch làm việc của mình.

*Đối với Admin

- Quản lý tài khoản người dùng: Xem danh sách bệnh nhân, bác sĩ, tạo hoặc xóa tài khoản bác sĩ.
- Quản lý chuyên khoa: Thêm, sửa, xóa danh sách chuyên khoa.
- Quản lý thông tin bác sĩ: Hồ sơ, lịch làm việc, chuyên môn.
- Duyệt và kiểm soát lịch đặt khám (nếu cần).
- Quản lý cơ sở dữ liệu và bảo mật: Theo dõi và kiểm tra thông tin trên Firebase.
- Thống kê, báo cáo: Số lượt khám, lịch đã đặt, bác sĩ được chọn nhiều,...

b, Yêu cầu phi chức năng

Đề tài tập trung nghiên cứu và xây dựng một ứng dụng đặt lịch khám bệnh trên nền tảng thiết bị di động (Android), tích hợp với cơ sở dữ liệu để quản lý thông tin người dùng, bác sĩ, cơ sở y tế và lịch hẹn. Vì vậy, yêu cầu phi chức năng cụ thể như sau:

- **Dễ sử dụng:** Giao diện của ứng dụng cần được thiết kế thân thiện, dễ hiểu và dễ sử dụng cho mọi đối tượng người dùng.
- **Hiệu năng:** Ứng dụng cần hoạt động nhanh chóng, mượt mà và hiệu quả, ngay cả

khi có nhiều công việc.

- **Độ tin cậy:** Dữ liệu cần được lưu trữ an toàn và bảo mật, đảm bảo không bị mất mát hoặc xâm nhập trái phép

1.4 Phân chia nhiệm vụ

STT	Họ và tên	Nhiệm vụ
1	Hoàng Thị Phương	<ul style="list-style-type: none"> - Thiết kế cơ sở dữ liệu, - Làm giao diện bệnh nhân và admin - Code các chức năng: các chức năng của trang chủ, tìm kiếm, đăng ký, hồ sơ y tế, thanh nav_bar, Các chức năng phía admin(Quản lý bác sĩ, Quản lý bệnh nhân, Quản lý csyt, Quản lý chuyên khoa, Trạng thái tài khoản, Thêm chuyên khoa, Thông báo, tải ảnh lên firebase))
2	Bùi Thị Hảo	<ul style="list-style-type: none"> - Làm figma admin - Làm giao diện bác sĩ (trang chi tiết, trang sử thông tin, trang tạo bài viết, trang thông tin bác sĩ - Code các chức năng của bác sĩ: Sửa thông tin bác sĩ, Tạo bài viết, xóa bài viết, trang thông tin chi tiết bác sĩ, xác nhận lịch hẹn
3	Đoàn Thị Ánh Hậu	<ul style="list-style-type: none"> - Tham gia sửa figma - Làm giao diện tài khoản csyt: Sửa cơ sở y tế, Quản lý chuyên khoa khám, trang thêm chuyên khoa khám - Code chức năng của csyt: Sửa thông tin csyt, Thêm chuyên khoa khám csyt, xóa chuyên khoa khám, xác nhận đặt lịch khám, đăng nhập, đăng xuất
4	Đoàn Thị Hà Giang	<ul style="list-style-type: none"> - Làm figma (Giao diện user, bác sĩ, csyt) - Làm giao diện: Chi tiết lịch hẹn, Danh sách bác sĩ, Khám chuyên khoa - Code chức năng: Tạo lịch hẹn, trang thông tin chi tiết lịch hẹn, list hiển thị lịch hẹn, hủy lịch hẹn), chia sẻ ứng dụng, liên hệ, điều khoản .

Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ

2.1. Kiến trúc hệ thống

2.1.1. Khái niệm kiến trúc hệ thống

Kiến trúc hệ thống là bản thiết kế tổng thể thể hiện cấu trúc và sự tổ chức của hệ thống phần mềm. Nó bao gồm các thành phần chính (modules), các lớp (layers), dữ liệu, dịch vụ và mối quan hệ tương tác giữa chúng nhằm đảm bảo hệ thống hoạt động hiệu quả, ổn định và có khả năng mở rộng trong tương lai.

Trong đề tài này, kiến trúc hệ thống ứng dụng **Booking Care** được xây dựng dựa trên mô hình **3 lớp (Three-tier Architecture)** kết hợp với các dịch vụ của **Firebase** để tối ưu hiệu suất và đơn giản hóa triển khai.

2.1.2. Mô hình kiến trúc tổng thể

Hệ thống được chia thành ba lớp chính:

a. Lớp trình bày (Presentation Layer)

- Là giao diện người dùng (UI) của ứng dụng di động, được phát triển trên nền tảng **Android** sử dụng ngôn ngữ Java.
- Đóng vai trò tiếp nhận thao tác từ người dùng và hiển thị dữ liệu truy xuất từ hệ thống.
- Giao tiếp với backend thông qua các API hoặc Firebase SDK.

b. Lớp xử lý nghiệp vụ (Business Logic Layer)

- Đảm nhận việc xử lý các quy trình nghiệp vụ như: đặt lịch, kiểm tra lịch trống, xác thực người dùng, xử lý thông báo.
- Sử dụng **Firebase Cloud Functions** để thực hiện các tác vụ backend không cần máy chủ riêng (serverless).
- Logic được viết bằng **JavaScript/TypeScript** và chạy trên nền tảng **Google Cloud Functions**.

c. Lớp dữ liệu (Data Layer)

- Lưu trữ toàn bộ dữ liệu của hệ thống: thông tin người dùng, bác sĩ, lịch hẹn, phản

hồi...

- Sử dụng **Firebase Realtime Database** hoặc **Firestore** để quản lý dữ liệu theo thời gian thực.
- **Firebase Storage** được sử dụng để lưu trữ hình ảnh (hồ sơ y tế, ảnh đại diện bác sĩ, tài liệu đính kèm...).

2.1.3. Các thành phần chính của hệ thống

Thành phần	Công nghệ sử dụng	Mô tả chức năng
Ứng dụng Android (Client)	Java	Giao diện người dùng, tương tác và gửi yêu cầu đến server
Firebase Authentication	Firebase	Quản lý đăng ký
Firebase Realtime Database / Firestore	Firebase	Lưu trữ dữ liệu người dùng, bác sĩ, lịch hẹn, phản hồi, phân quyền người dùng, đăng nhập
Firebase Cloud Messaging (FCM)	Firebase	Gửi thông báo nhắc lịch đến người dùng
Firebase Storage	Firebase	Lưu trữ tệp tin như ảnh đại diện, hình ảnh bệnh lý, giấy tờ đính kèm

2.1.4. Luồng hoạt động của hệ thống

1. Người dùng mở ứng dụng và đăng nhập bằng tài khoản đã đăng ký (sử dụng Firebase Authentication).
2. Ứng dụng gửi yêu cầu truy xuất danh sách bác sĩ hoặc phòng khám.
3. Firebase Realtime Database trả về dữ liệu phù hợp theo truy vấn.
4. Người dùng chọn bác sĩ, ngày giờ, và gửi yêu cầu đặt lịch.
5. Firebase Cloud Function kiểm tra lịch trống, xử lý logic và lưu thông tin lịch hẹn vào cơ sở dữ liệu.
6. Sau khi đặt lịch thành công, hệ thống sử dụng Firebase Cloud Messaging để gửi thông báo xác nhận, đồng thời thiết lập nhắc lịch tự động.
7. Người dùng có thể xem, chỉnh sửa hoặc hủy lịch hẹn thông qua ứng dụng.

- Các ảnh đại diện, giấy tờ y tế (nếu có) sẽ được tải lên và lưu trữ tại **Firebase Storage**.

2.1.5. Ưu điểm của kiến trúc hệ thống

- Hiệu suất cao:** Do sử dụng Firebase với khả năng mở rộng tự động và xử lý theo thời gian thực.
- Chi phí thấp:** Firebase cung cấp gói miễn phí và chỉ tính phí khi sử dụng vượt mức.
- Không cần máy chủ riêng:** Dễ triển khai và bảo trì nhờ vào mô hình **serverless**.
- Khả năng mở rộng linh hoạt:** Có thể tích hợp thêm các tính năng như chatbot tư vấn, thanh toán trực tuyến, video call...
- Bảo mật dữ liệu:** Firebase cung cấp cơ chế phân quyền, xác thực và mã hóa dữ liệu.

2.2. Giới thiệu về Công nghệ phát triển

2.2.1. Ngôn ngữ lập trình

- Java:** Là ngôn ngữ chính được sử dụng để phát triển ứng dụng Android. Java có tính phổ biến cao, cộng đồng hỗ trợ lớn, cú pháp rõ ràng và được hỗ trợ mạnh mẽ bởi Android SDK.
- JavaScript (hoặc TypeScript):** Được sử dụng để viết các hàm backend trên Firebase Cloud Functions, phục vụ các tác vụ như xử lý lịch hẹn, gửi thông báo, xác thực nâng cao.

2.2.2. Nền tảng phát triển

- Android Studio:** Là công cụ phát triển chính thức cho ứng dụng Android, hỗ trợ đầy đủ Java, tích hợp các công cụ kiểm tra, mô phỏng và quản lý tài nguyên dễ dàng.
- Firebase Console:** Giao diện web giúp cấu hình và theo dõi các dịch vụ như Authentication, Database, Storage, Cloud Functions... thuận tiện trong quá trình phát triển và triển khai.

2.2.3. Lý do lựa chọn công nghệ

- Java** là ngôn ngữ truyền thống, được Android hỗ trợ mạnh mẽ, dễ tìm tài liệu và dễ dàng bảo trì.
- Firebase** giúp đơn giản hóa quá trình phát triển nhờ tích hợp nhiều dịch vụ trong cùng một nền tảng: từ xác thực, lưu trữ, xử lý nghiệp vụ đến gửi thông báo.

- Không cần xây dựng server riêng, tiết kiệm thời gian và chi phí.
- Phù hợp với mô hình MVP (Minimum Viable Product) – nhanh chóng có sản phẩm mẫu để thử nghiệm và cải tiến.

Chương 3. XÂY DỰNG ỨNG DỤNG

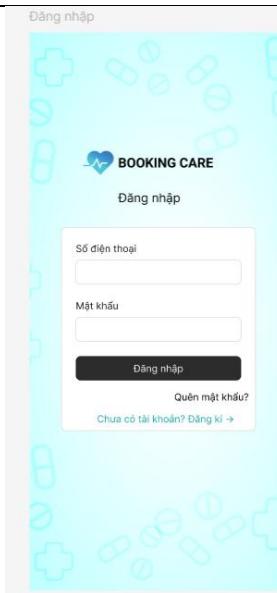
3.1. Thiết kế Figma

3.1.1. Giao diện User (Bệnh nhân)

Hình 1.1 Giao diện Màn hình chính



Hình 1.2 Giao diện Đăng nhập

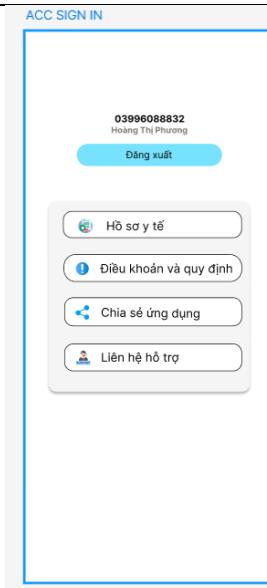


Hình 1.3. Giao diện Đăng kí



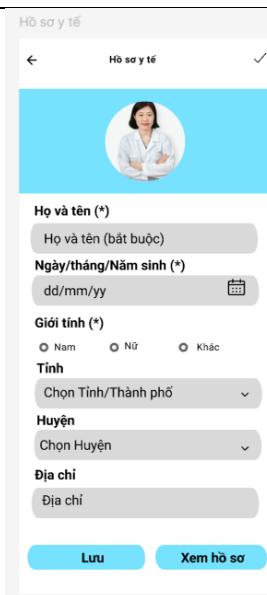
The registration screen for Booking Care. It features a light blue background with white medical icons (pill bottles, syringes, etc.). At the top right is the text "Đăng kí tài khoản". Below it is the "BOOKING CARE" logo with a heart icon. The main form is titled "Đăng kí" and contains three input fields: "Số điện thoại", "Mật khẩu", and "Nhập lại mật khẩu". Each field has a placeholder text and a red asterisk indicating it is required. Below these fields are two buttons: "Đăng nhập" (blue) and "Đăng kí" (black). There is also a small link "Đang nhập?".

1.4. Giao diện tài khoản



The account sign-in screen. It shows a blue header bar with the text "ACC SIGN IN". Below it is a user profile card with the phone number "03996088832" and the name "Hoàng Thị Phương". A blue button labeled "Đăng xuất" (Logout) is located below the profile. The main content area contains four rounded rectangular buttons with icons: "Hồ sơ y tế" (Medical Record), "Điều khoản và quy định" (Terms and Conditions), "Chia sẻ ứng dụng" (Share App), and "Liên hệ hỗ trợ" (Contact Support).

1.5 Giao diện Hồ sơ y tế

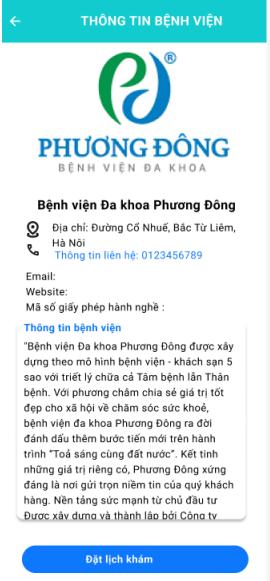


The medical record creation screen. It has a light gray header with the text "Hồ sơ y tế" and navigation arrows. The main form starts with a circular profile picture of a woman. Below it is a section for "Họ và tên (*)" with a placeholder "Họ và tên (bắt buộc)". Next is a date input field "Ngày/tháng/Năm sinh (*)" with a calendar icon. Following that are gender options "Giới tính (*)": "Nam", "Nữ", and "Khác". Then there are dropdown menus for "Tỉnh" (Province) and "Huyện" (District). Below that is a text input field "Địa chỉ" (Address). At the bottom are two buttons: "Lưu" (Save) and "Xem hồ sơ" (View Record).

1.6. Giao diện Khám chuyên khoa



1.7. Giao diện cơ sở y tế



1.8. Giao diện Thông tin bác sĩ



1.9. Giao diện Menu



1.10. Giao diện Chuyên khoa - Cơ xương khớp



1.26. Giao diện Thông tin Phiếu đặt lịch



1.27. Giao diện Thông tin lịch khám

Thông tin lịch khám



1.28. Giao diện Điều khoản sử dụng

Điều khoản sử dụng

← ĐIỀU KHOẢN SỬ DỤNG

ĐIỀU KHOẢN VÀ ĐIỀU KIỆN SỬ DỤNG

GIỚI THIỆU

Chúng tôi, Công ty Cổ phần Công nghệ BookingCare, đơn vị sở hữu và vận hành "Nền tảng Y tế Chăm sóc sức khỏe toàn diện BookingCare" bao gồm hệ thống website và các ứng dụng di động. BookingCare cung cấp nền tảng công nghệ để bệnh nhân thuận tiện trong việc đặt lịch dịch vụ y tế với bác sĩ và cơ sở y tế. Việc truy cập hoặc sử dụng dịch vụ của BookingCare, bạn hoàn toàn đồng ý theo các điều khoản, điều kiện dưới đây.

Chúng tôi duy trì quyền thay đổi hoặc điều chỉnh bất kỳ điều khoản và điều kiện nào bất cứ khi nào. Mọi sửa đổi nếu có sẽ có hiệu lực ngay lập tức sau khi đăng tải trên hệ thống trong này.

SỬ DỤNG BOOKINGCARE

Thông tin người cung cấp dịch vụ "Khám chữa bệnh"

1.29. Giao diện Liên hệ hỗ trợ

Liên hệ hỗ trợ

← LIÊN HỆ HỖ TRỢ

Nền tảng Đặt khám BookingCare

ĐT: 02473.612.468

Email: support@bookingcare.vn

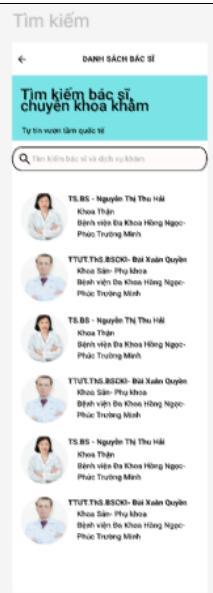
Trụ sở:

Công ty Cổ phần Công nghệ BookingCare

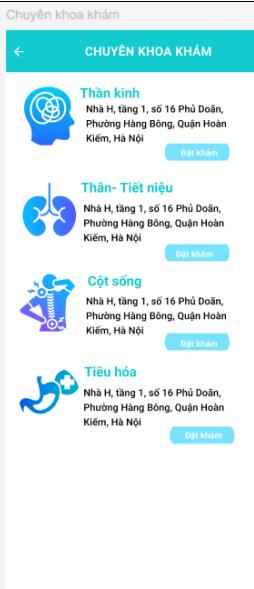
ĐKKD số: 0106790291, Số 101-DT Hà Nội cấp ngày: 16/03/2015

Địa chỉ: Lô 04/D21, Khu đô thị mới Cầu Giấy, Phường Dịch Vọng Hậu, Quận Cầu Giấy, Thành phố Hà Nội, Việt Nam

1.30. Giao diện tìm kiếm



1.31. Giao diện Chuyên khoa - CSYT

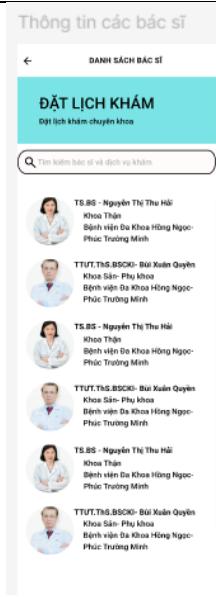


1.32. Giao diện đặt lịch



3.1.2 Giao diện Bác sĩ

1.32. Giao diện Thông tin các bác sĩ



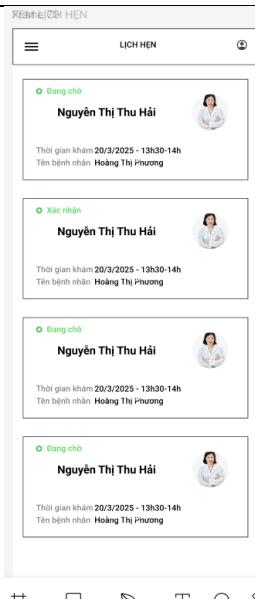
1.33. Giao diện Trang chủ



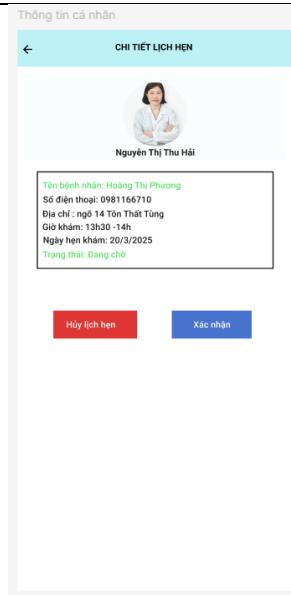
1.34. Giao diện Thông tin chi tiết Cơ sở y tế



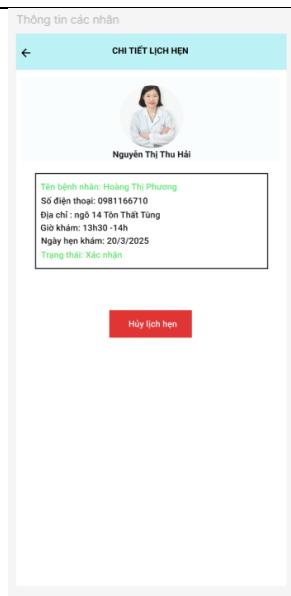
1.35. Giao diện xem lịch hẹn



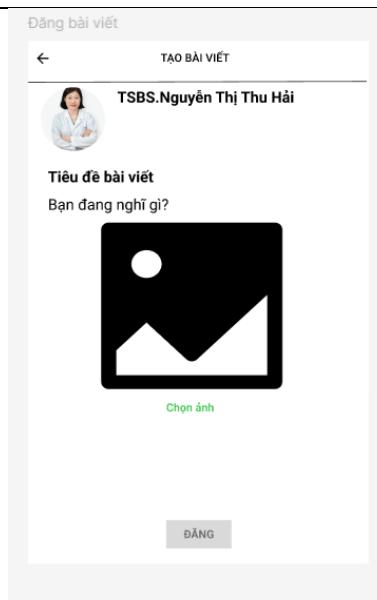
1.36. Giao diện Chi tiết lịch hẹn



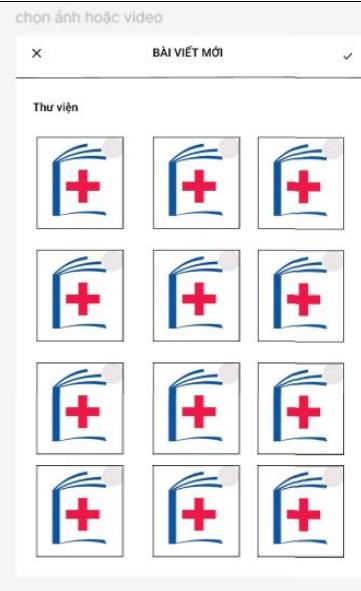
1.37. Giao diện Chi tiết lịch hẹn đã xác nhận



1.38. Giao diện Đăng bài viết



1.39. Giao diện Chọn ảnh/video



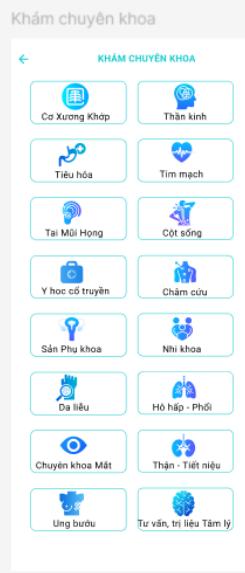
1.40. Giao diện Thông tin chi tiết Bác sĩ



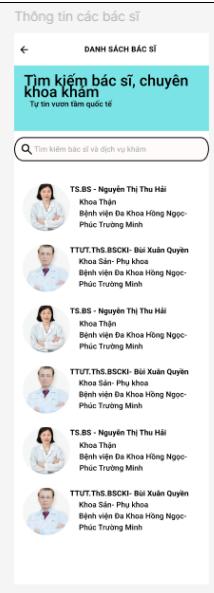
1.41. Giao diện Danh sách bác sĩ



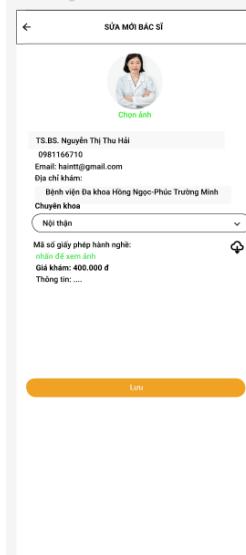
1.42. Giao diện Chuyên khoa khám



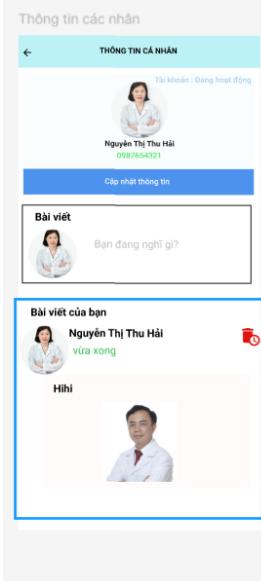
1.43. Giao diện Tìm kiếm



1.44. Giao diện sửa thông tin



1.45. Giao diện Trang cá nhân



1.46. Giao diện Menu



3.1.3 Giao diện cơ sở y tế

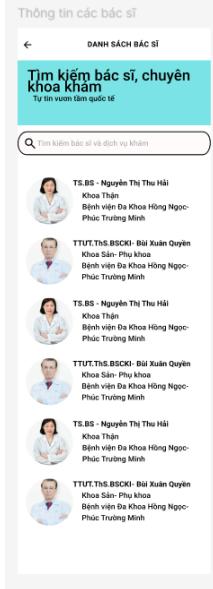
1.50. Giao diện Trang chủ



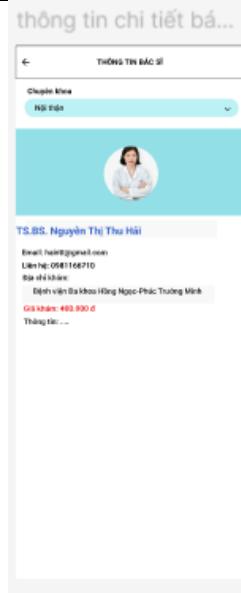
1.51. Giao diện Menu



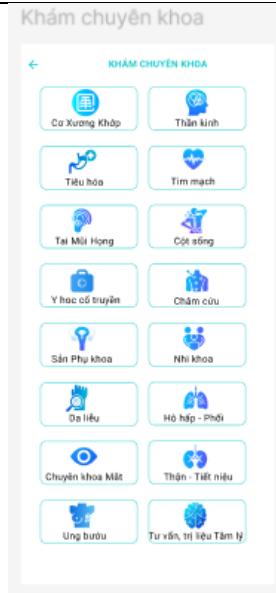
1.52. Giao diện tìm kiếm



1.53. Giao diện Thông tin chi tiết bác sĩ



1.54. Giao diện Chuyên khoa khám



1.55. Giao diện Danh sách bác sĩ



1.56. Giao diện thông tin chi tiết CSYT

The screenshot shows a mobile application interface for a hospital. At the top, it says "thông tin chi tiết cơ sở y tế". Below that is a logo of a green circle with a white cross. The text "BỆNH VIỆN HỮU NGHỊ VIỆT ĐỨC" is displayed. It includes the address "Số 16 Phù Đổng, Hàng Bông, Hoàn Kiếm", phone number "0972866834", email "benhvienvietduc.info@gmail.com", website "https://benhvienvietduc.org", and license number "GPKD01234567890". A section titled "Thông tin bệnh viện:" lists working hours from 2pm to 7pm, 7am to 12pm, and 1pm to 3pm. It also describes the hospital as one of five in the city, with a history of over 100 years, specialized in various medical fields, and having many foreign experts. It mentions that the hospital has a high success rate in treatment and surgery. Another section discusses the qualifications of the medical staff.

1.57. Giao diện sửa thông tin

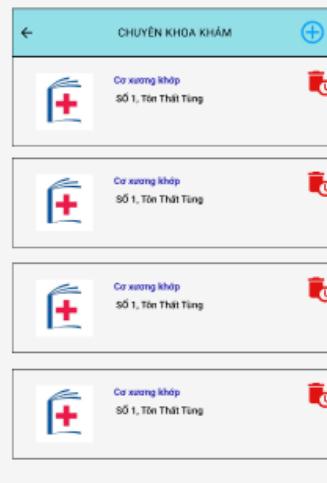
This screenshot shows the same mobile application interface as above, but in edit mode. The title bar says "sửa thông tin chi tiết chuyên...". The rest of the content is identical to the previous screenshot, including the hospital's name, address, contact information, and detailed descriptions of its services and staff.

1.59. Giao diện Thông tin CSYT

This screenshot shows the main information screen for the hospital. It features a large illustration of the hospital building at the top. Below the building, there are two buttons: "Thông tin cơ sở y tế" and "Quản lý chuyên khoa khám".

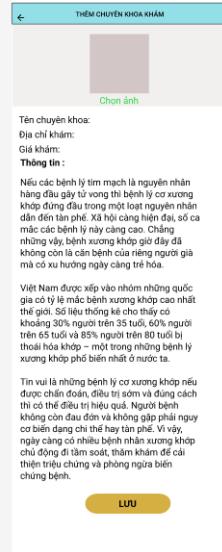
1.60. Giao diện Hiển thị danh sách chuyên khoa

Hiển thị danh sách chuyên k...



1.61. Giao diện Thêm thông tin Chuyên khoa

thêm thông tin chi tiết chuyên ...



1.62. Giao diện xem lịch hẹn

XEM LỊCH HẸN



1.63. Giao diện thông tin đặt lịch

Thông tin các nhân

← CHI TIẾT LỊCH HẸN



Thân kinh

Tên bệnh nhân: Hoàng Thị Phương
Số điện thoại: 0981166710
Địa chỉ: ngõ 14 Tôn Thất Tùng
Giờ khám: 13h30 - 14h
Ngày hẹn khám: 20/3/2025
Trạng thái: Đang chờ

Hủy lịch hẹn

Xác nhận

1.64. Giao diện Thông tin đặt lịch đã xác nhận

Thông tin các nhân

← CHI TIẾT LỊCH HẸN



Thân kinh

Tên bệnh nhân: Hoàng Thị Phương
Số điện thoại: 0981166710
Địa chỉ: ngõ 14 Tôn Thất Tùng
Giờ khám: 13h30 - 14h
Ngày hẹn khám: 20/3/2025
Trạng thái: Xác nhận

Hủy lịch hẹn

3.1.4. Giao diện Admin

1.65. Giao diện Trang chủ

BOOKING CARE ADMIN



QUẢN LÝ BÁC SĨ



QUẢN LÝ BỆNH NHÂN



QUẢN LÝ CƠ SỞ Y TẾ

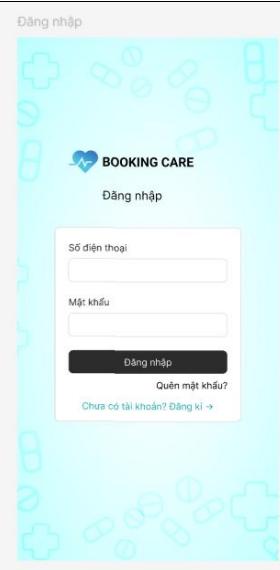


QUẢN LÝ CHUYÊN KHOA

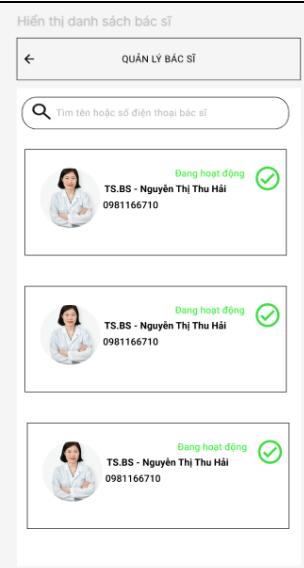


QUẢN LÝ BÀI VIẾT

1.66. Giao diện Đăng nhập



1.67. Giao diện Hiển thị danh sách bác sĩ



1.68. Giao diện thông tin chi tiết bác sĩ



1.70. Giao diện Sửa thông tin cơ sở y tế

sửa thông tin cơ sở y tế

← THÔNG TIN CƠ SỞ Y TẾ



Tên cơ sở y tế (*)
Bệnh viện Hữu nghị Việt - Đức

Thông tin liên hệ : (*)
0981166710

Địa chỉ : (*)
 Nhập đầy đủ thông tin (bắt buộc)

Email : (*)
 Nhập đầy đủ thông tin (bắt buộc)

Mã số giấy phép kinh doanh : (*)
GPKD01234567890

Website: (*)
 Nhập đầy đủ thông tin (bắt buộc)

Thông tin cơ sở y tế (*)
 Nhập đầy đủ thông tin (bắt buộc)

LƯU

1.71. Giao diện Thêm thông tin chi tiết chuyên khoa

thêm thông tin chi tiết chuyên khoa

← THÊM KHOA KHOA



Chọn ảnh

Tên chuyên khoa:

Thông tin :
Nếu các bệnh lý tim mạch là nguyên nhân hàng đầu gây tử vong thì bệnh lý xương khớp là nguyên nhân đứng thứ hai. Người bệnh không còn đau đớn và không gặp phải nguy cơ biến dạng chí thể hay tàn phế. Vì vậy, ngày càng có nhiều bệnh nhân xương khớp cần được điều trị và phẫu thuật để cải thiện triệu chứng và phòng ngừa biến chứng bệnh.

LƯU

1.72. Giao diện Sửa thông tin chi tiết chuyên khoa

sửa thông tin chi tiết chuyên khoa

← SỬA THÔNG TIN CHUYÊN KHOA



Chọn ảnh

CHUYÊN KHOA XƯƠNG KHỚP

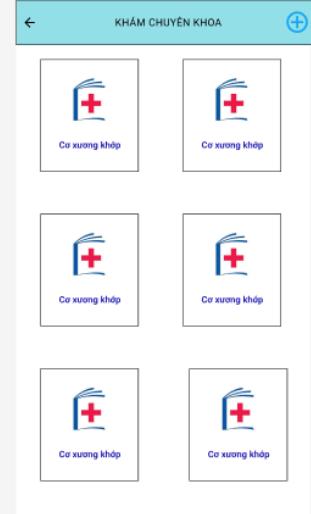
Thông tin :
Nếu các bệnh lý tim mạch là nguyên nhân hàng đầu gây tử vong thì bệnh lý xương khớp là nguyên nhân đứng thứ hai. Người bệnh không còn đau đớn và không gặp phải nguy cơ biến dạng chí thể hay tàn phế. Vì vậy, ngày càng có nhiều bệnh nhân xương khớp cần được điều trị và phẫu thuật để cải thiện triệu chứng và phòng ngừa biến chứng bệnh.

Thông tin :
Nếu các bệnh lý tim mạch là nguyên nhân hàng đầu gây tử vong thì bệnh lý xương khớp là nguyên nhân đứng thứ hai. Người bệnh không còn đau đớn và không gặp phải nguy cơ biến dạng chí thể hay tàn phế. Vì vậy, ngày càng có nhiều bệnh nhân xương khớp cần được điều trị và phẫu thuật để cải thiện triệu chứng và phòng ngừa biến chứng bệnh.

LƯU

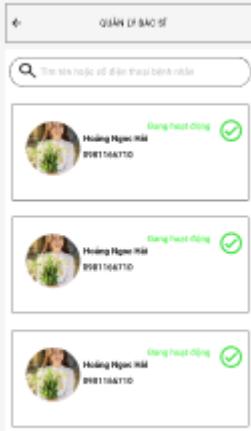
1.73. Giao diện Hiển thị danh sách chuyên khoa

Hiển thị danh sách chuyên khoa



1.74. Giao diện Hiển thị danh sách bác sĩ

Hiển thị danh sá...



1.75. Giao diện Hiển thị danh sách CSYT

Hiển thị danh sách cơ sở y tế



3.2. Thiết kế CSDL

3.2.1. Xác định các lớp

Phát triển ứng dụng cho thiết bị di động

* Lớp accout (các tài khoản người dùng đã đăng kí):

- Thuộc tính:

- String id : mã người dùng
- String name: tên người dùng
- String phone: số điện thoại của người dùng
- String pass: mật khẩu tài khoản của người dùng
- String as : vai trò của người dùng
- String token: mã xác thực người dùng
- String status: Trang thái người dùng

- Phương thức:

- accout
- getToken, setToken
- getId, setId
- getStatus, setStatus
- getName, setName
- getPhone, setPhone
- getPass, setPass
- getAs, setAs

* Lớp Bacsi (các tài khoản người dùng có vai trò là bác sĩ):

- Thuộc tính:

- String id : mã bác sĩ
- String name: tên bác sĩ
- String chuyenkhoa: tên chuyên khoa
- String imggiayphep: ảnh chứng chỉ hành nghề của bác sĩ
- String diachi: địa chỉ làm việc của bác sĩ
- String img: ảnh bác sĩ
- String thongtin: thông tin bác sĩ
- String Giakham: giá khám
- String sogiayphephanhnghe: mã số giấy phép hành nghề của bác sĩ
- String email: email bác sĩ

- String sdt: số điện thoại bác sĩ
- Phương thức:
 - Bacsi
 - getImggiayphep, setImggiayphep
 - getSogiayphephanhnghe, setSogiayphephanhnghe
 - getEmail, setEmail
 - getSdt, setSdt
 - getId, setId
 - getName, setName
 - getChuyenkhoa, setChuyenkhoa
 - getDiachi, setDiachi
 - getImg, setImg
 - getThongtin, setThongtin
 - getGiaKham, setGiaKham

* Lớp benhnhan (các tài khoản người dùng có vai trò là bệnh nhân):

- Thuộc tính:
 - String id : mã bệnh nhân
 - String ten: tên bệnh nhân
 - String sodienthoai: Số điện thoại bệnh nhân
 - String diachi: Địa chỉ bệnh nhân
 - String gioitinh: Giới tính bệnh nhân
 - String ngaysinh: Ngày sinh bệnh nhân
 - String benhlynen: Tiền sử bệnh nền của bệnh nhân
 - String img: ảnh bệnh nhân
- Phương thức:
 - Benhnhan
 - getImg, setImg
 - getId, setId
 - getTen, setTen
 - getSodienthoai, setSodienthoai

- getDiachi, setDiachi
- getGioitinh, setGioitinh
- getNgaysinh, setNgaysinh
- getBenhlynen, setBenhlynen

* Lớp Cosoyte(các tài khoản người dùng có vai trò là cơ sở y tế):

- Thuộc tính:
 - String id : mã cơ sở y tế
 - String name: tên cơ sở y tế
 - String img : ảnh logo của cơ sở y tế
 - String diachi: Địa chỉ cơ sở y tế
 - String masogiayphep : mã số giấy phép hoạt động của cơ sở y tế
 - String email: email của cơ sở y tế
 - String sdt: số điện thoại của cơ sở y tế
 - String website: trang web của cơ sở y tế
 - String thongtin: thông tin cơ sở y tế
- Phương thức:
 - cosoyte
 - getWebsite, setWebsite
 - getMasogiayphep, setMasogiayphep
 - getEmail, setEmail
 - getSdt, setSdt
 - getName, setName
 - getId, setId
 - getImg, setImg
 - getDiachi, setDiachi
 - getThongtin, setThongtin

* Lớp lichhen (lịch hẹn của bác sĩ và bệnh nhân):

- Thuộc tính:
 - String id : mã bài viết

- String idbenhnhan: mã bệnh nhân
 - String idnguoien: mã bác sĩ
 - String ngayhenkham: Ngày hẹn khám của bệnh nhân
 - String khunggiokham: khung giờ khám của bệnh nhân
 - String trangthai: trạng thái xác nhận của lịch hẹn
 - String namebenhnhan: Tên bệnh nhân
 - String sdtbenhnhan: số điện thoại của bệnh nhân
 - String diachibenhnhan: Địa chỉ bệnh nhân
 - String avatar_nganh: ảnh của bác sĩ
 - String name_hen: tên bác sĩ
- Phương thức:
 - Lichhen
 - getNamebs, setNamebs
 - getAvatarbs, setAvatarbs
 - getSdtbenhnhan, setSdtbenhnhan
 - getDiachibenhnhan, setDiachibenhnhan
 - getId, setId
 - getNamebenhnhan, setNamebenhnhan
 - getIdbenhnhan, setIdbenhnhan
 - getIdbacsi, setIdbacsi
 - getNgayhenkham, setNgayhenkham
 - getKhunggiokham, setKhunggioikham
 - getTrangthai, setTrangthai

* Lớp Baiviet (Bài viết chia sẻ của bác sĩ):

- Thuộc tính:
 - String id : mã bài viết
 - String idusers : mã users có vai trò là bác sĩ
 - String titile: tiêu đề bài viết
 - String content: nội dung bài viết
 - String timestamp: thời gian đăng

- String img : ảnh đại diện của bác sĩ
- Phương thức:
 - Baiviet
 - getImg, setImg
 - getTitile, setTitile
 - getId, setId
 - getIduser, setIduser
 - getContent, setContent
 - getTimestamp, setTimestamp

* Lớp chuyenkhoa (chuyên khoa khám bệnh của các bác sĩ):

- Thuộc tính:
 - String id : mã chuyên khoa
 - String tenchuyenkhoa: tên chuyên khoa khám bệnh
 - String img: ảnh biểu tượng cho các chuyên khoa khám bệnh
 - String thongtin : Thông tin chuyên khoa khám bệnh
- Phương thức:
 - chuyenkhoa
 - getId, setId
 - getTenchuyenkhoa, setTenchuyenkhoa
 - getImg, setImg
 - getThongtin, setThongtin

* Lớp chuyenkhoacsyt (các chuyên khoa khám bệnh của cơ sở y tế):

- Thuộc tính:
 - String id : mã chuyên khoa
 - String sdtcsyt: số điện thoại của cơ sở y tế
 - String tenkhoa: tên chuyên khoa
 - String giakham: giá khám của chuyên khoa
 - String img: ảnh của chuyên khoa
 - String thongtin: Thông tin của chuyên khoa
- Phương thức:

- Chuyenkhoacsyt
- GetSdtcsyt, setSdtcsyt
- GetId, setId
- GetTenkhoa, setTenkhoa
- GetDiachi, setDiachi
- GetGiakham, setGiakham
- GetImg, setImg
- GetThongtin, setThongtin

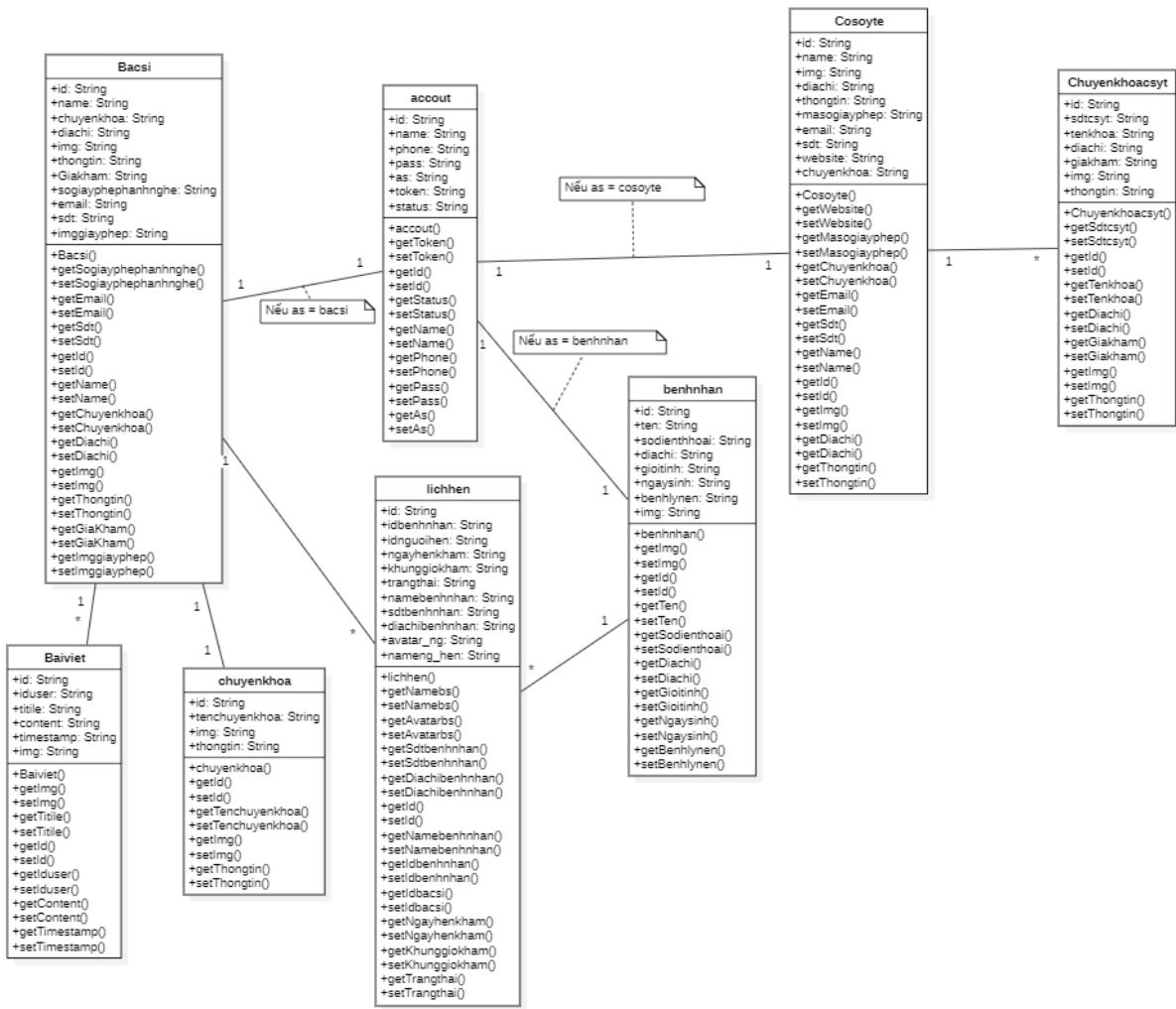
3.2.2. Mối quan hệ giữa các lớp

- **accout -Bacsı:** Nếu tài khoản người dùng đóng vai trò là bác sĩ thì một tài khoản đăng kí chỉ có 1 tài khoản bác sĩ. Đây là mối quan hệ **Một - Một**.
- **accout -benhnhan:** Nếu tài khoản người dùng đóng vai trò là bệnh nhân thì một tài khoản đăng kí chỉ có 1 tài khoản bệnh nhân . Đây là mối quan hệ **Một - Một**.
- **accout -Cosoyte:** Nếu tài khoản người dùng đóng vai trò là cơ sở y tế thì một tài khoản đăng kí chỉ có 1 tài khoản cơ sở y tế. Đây là mối quan hệ **Một - Một**.
- **Bacsı - chuyenkhoa :** Mỗi một bác sĩ chỉ có một chuyên khoa khám bệnh. Đây là mối quan hệ **Một - Một**.
- **Bacsı - Baiviet:** có liên kết **Một - Nhiều.** Một bác sĩ có thể đăng nhiều bài viết.
- **Bacsı - lichhen:** có liên kết **Một - Nhiều.** Một bác sĩ có thể có nhiều lịch hẹn.
- **benhnhan - lichhen:** có liên kết **Một - Nhiều.** Một bệnh nhân có thể đặt nhiều lịch hẹn.
- **Cosoyte – Chuyenkhoacsyt:** có liên kết **Một - Nhiều.** Một cơ sở y tế có thể thêm nhiều chuyên khoa khám bệnh cơ sở y tế.

3.2.3. Biểu đồ lớp cho mô hình miền

- Tập trung vào các thực thể nghiệp vụ và mối quan hệ giữa chúng.
- Thường không chứa các lớp liên quan đến giao diện người dùng, cơ sở dữ liệu, hay các chi tiết kỹ thuật khác.
- Trong ứng dụng "Đặt lịch khám BOOKINGCARE", biểu đồ lớp cho mô hình miền chứa các thực thể như:
 - Accout: các thuộc tính bao gồm id, name, phone, pass, as, status, token. Thực hiện chức năng phân quyền người dùng là bệnh nhân, bác sĩ, cơ sở y tế , admin.

- Bacsi:các thuộc tính bao gồm id, name, chuyenkhoa, diachi, img, imggiayphep, thongtin, GiaKham, sogenayphephanhnghe, email, sdt. Thực hiện chức sửa thông tin bác sĩ, tạo bài viết, xóa bài viết, hiển thị trang thông tin chi tiết bác sĩ, xác nhận lịch hẹn
- Benhnhan: các thuộc tính bao gồm:id, ten, sodienthoai, diachi, gioitinh, ngaysinh, benhlynen, img. Thực hiện các chức năng sửa thông tin bệnh nhân, đặt lịch khám bệnh.
- Baiviet: các thuộc tính bao gồm: id, iduser, titile, content, timestamp, img.
- chuyenkhoa: các thuộc tính bao gồm: id, tenchuyenkhoa, img, thongtin.
- Chuyenkhoacsyt: các thuộc tính bao gồm: id, sdtcsyt, tenkhoa, diachi, giakham, img, thongtin.
- Cosoyte: các thuộc tính bao gồm: id, name, img, diachi, thongtin, masogiayphep, email, sdt, website, chuyenkhoa. Thực hiện các chức năng chỉnh sửa thông tin cơ sở y tế, thêm các chuyên khoa khám bệnh, xác nhận lịch hẹn.
- lichhen: các thuộc tính bao gồm: id, idbenhnhan, idnguoien, ngayhenkham, khunggiokham, trangthai, namebenhnhan, sdtbenhnhan, avatar_ng, nameng_hen. Thực hiện chức năng chính là hiển thị thông tin chi tiết lịch hẹn giữa bác sĩ và bệnh nhân.

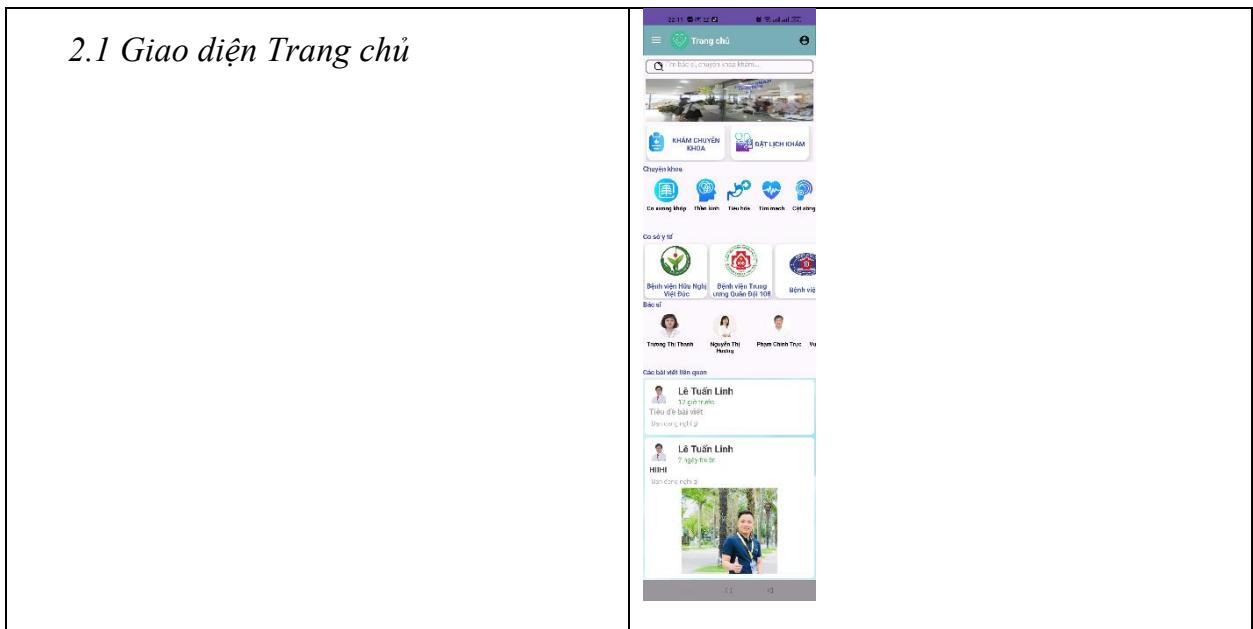


Biểu đồ 1. Biểu đồ lớp

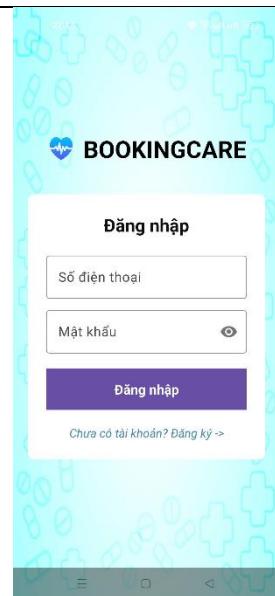
3.2. Giao diện ứng dụng

3.2.1. Màn hình giao diện Người dùng (Bệnh nhân)

2.1 Giao diện Trang chủ



2.2 Giao diện Đăng nhập



2.3. Giao diện Đăng ký



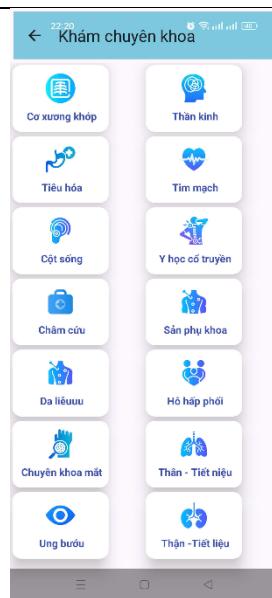
2.4. Giao diện tài khoản



2.5 Giao diện Hồ sơ y tế

The screenshot shows a mobile application interface for managing a patient's medical record. At the top, there is a placeholder for a profile picture with the text "Chọn ảnh" (Select photo). Below it, there are several input fields: "Họ và Tên (*)" (Last name and first name) containing "phuong", "Số điện thoại (*)" (Phone number) containing "0399608832", "Địa chỉ" (Address), "Giới tính" (Gender) with radio buttons for "Nữ" (Female) and "Nam" (Male) where "Nữ" is selected, "Ngày sinh" (Date of birth) with a date input field showing "dd/mm/yy", and "Bệnh lý nền" (Pre-existing conditions). A blue "Lưu" (Save) button is located at the bottom right. The top bar includes icons for signal strength, battery level, and connectivity.

2.6. Giao diện Khám chuyên khoa



2.7. Giao diện cơ sở y tế

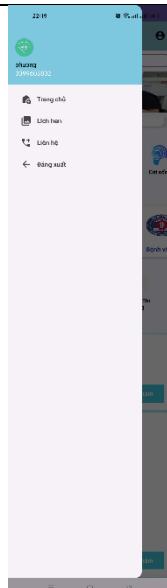


2.8. Giao diện Thông tin bác sĩ



Bật lịch ngày

2.9. Giao diện Menu



2.10. Giao diện Chuyên khoa - Tiêu hóa



2.11. Giao diện Thông tin Phiếu đặt lịch với bác sĩ

2.12. Giao diện Thông tin đặt lịch với CSYT

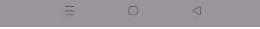
2.13. Giao diện Thông tin lịch khám



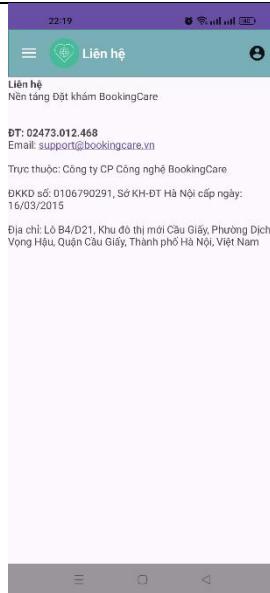
2.14. Giao diện Điều khoản sử dụng

ĐIỀU KHOẢN VÀ ĐIỀU KIỆN SỬ DỤNG GIỚI THIỆU Chúng tôi, Công ty CP Công nghệ BookingCare, đơn vị sở hữu và vận hành "Nền tảng Y tế Chăm sóc sức khỏe toàn diện BookingCare" bao gồm hệ thống website và các ứng dụng di động. BookingCare cung cấp nền tảng công nghệ để bệnh nhân thuận tiện trong việc đặt lịch dịch vụ y tế với bác sĩ và cơ sở y tế. Bằng việc truy cập hoặc sử dụng dịch vụ của BookingCare, bạn hoàn toàn đồng ý theo các điều khoản, điều kiện dưới đây.

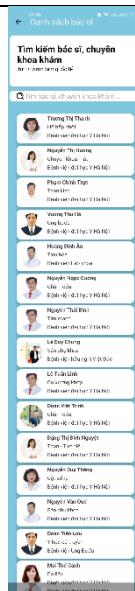
Chúng tôi duy trì quyền thay đổi hoặc điều chỉnh bất kỳ điều khoản và điều kiện nào dưới đây. Mọi sửa đổi nếu có sẽ có hiệu lực ngay lập tức sau khi đăng tải trên hệ thống trang này. **SỬ DỤNG BOOKINGCARE** Thông tin người cung cấp dịch vụ "Khám chữa bệnh" Hệ thống BookingCare đăng tải thông tin và lịch khám của bác sĩ, dịch vụ y tế và cơ sở y tế. Các thông tin này được cung cấp bởi chính "Người cung cấp dịch vụ Khám chữa bệnh" và các nguồn thông tin tin cậy khác do chúng tôi lựa chọn biên tập. Chúng tôi cố gắng tìm hiểu và lựa chọn thông tin chính xác để đăng tải trên hệ thống. Tuy nhiên, chúng tôi không đủ điều kiện xác minh sự chính xác tuyệt đối của thông tin đã đăng tải. **Dịch vụ đặt lịch khám trực tuyến** BookingCare cung cấp nền tảng công nghệ, phương tiện để kết nối bệnh nhân và bác sĩ, cơ sở y tế, qua đó cung cấp dịch vụ đặt lịch khám trực tuyến. Bệnh nhân lựa chọn bác sĩ, dịch vụ hoặc cơ sở y tế phù hợp trên hệ thống BookingCare để đặt lịch khám. BookingCare không phải là người cung cấp dịch vụ y tế và cũng không đại diện cho bất kỳ "Người cung cấp dịch vụ khám chữa bệnh" nào. Vai trò duy nhất của chúng tôi là



2.15. Giao diện Liên hệ hỗ trợ



2.16. Giao diện tìm kiếm



The screenshot shows the 'Chuyên khoa khám' (Specialty Consultation) section of the CSYT app. At the top, there is a back arrow icon and the text 'Chuyên khoa khám' followed by a QR code and a search icon. Below this, there are four categories with icons and details:

- Thần kinh**: Nhà H, tầng 1, số 16 Phù Đổan, Phường Hàng Bông, Quận Hoàn Kiếm, Hà Nội. A blue 'Đặt khám' button is present.
- Thân - Tiết niệu**: Nhà H, tầng 1, cổng số 7, số 16 Phù Đổan, Hàng Bông, Hoàn Kiếm, Hà Nội. A blue 'Đặt khám' button is present.
- Cột sống**: Nhà C4, Tầng 1, số 16-18 Phù Đổan, Hàng Bông, Hoàn Kiếm, Hà Nội. A blue 'Đặt khám' button is present.
- Tiêu hóa**: Nhà H, tầng 1, số 16 Phù Đổan, Hàng Bông, Hoàn Kiếm, Hà Nội. A blue 'Đặt khám' button is present.

At the bottom of the screen, there is a navigation bar with three icons: a menu icon (three horizontal lines), a square icon, and a left arrow icon.

3.1.2. Màn hình giao diện Bác sĩ

2.19. Giao diện Thông tin các bác sĩ

The screenshot shows a mobile application interface for a doctor's profile. At the top, there is a header bar with icons for battery, signal, and time (22:56). The title "Thông tin bác sĩ" is displayed. Below the header, the text "Chuyên khoa:" is followed by a search bar containing the text "Hô hấp phổi". A portrait photo of a female doctor, Trương Thị Thanh, is shown. Below the photo, her name "Trương Thị Thanh" is displayed in blue, followed by her details: Mã số giấy phép: TTT123456789, Email: thanhtt@gmail.com, Liên hệ: 0869999888. The text "Địa chỉ khám" is followed by "Bệnh viện đại học Y Hà Nội". The price "Giá khám: 230.000 đ" is listed. A detailed description of her qualifications follows:

Bác sĩ chuyên khoa cấp II, Bác sĩ nội trú. Trương Thị Thanh Bác sĩ Trung tâm CBHA& CTĐQ - BV ĐH Y Hà Nội. Tóm tắt sơ lược về quá trình đào tạo, quá trình công tác, các thành tích và kinh nghiệm nghiên cứu khoa học nổi bật: 2007-2013: Sinh viên hệ bác sĩ đa khoa Trường Đại học Y Hà Nội. 2013-2016: Học Bác sĩ nội trú chuyên ngành chẩn đoán hình ảnh. 2016- đến nay: Bác sĩ TT Chẩn đoán hình ảnh và CTĐQ - BV Đại học Y Hà Nội 2020-2022: Bác sĩ Chuyên khoa II tại Trường Đại học Y Hà Nội. Tháng 6 đến tháng 12 năm 2018: đi học chẩn đoán hình ảnh tại đại học Ajou Hàn Quốc. Chuyên môn sâu - Đốt sống cao tần (RFA) điều trị nhân tuyến giáp dưới hướng dẫn siêu âm. - Điều trị suy tĩnh mạch chi dưới bằng phương pháp Laser, tiêm xơ. - Hút u vú cổ họng trợ áp lực âm (VABB) - Chọc hút tế bào, sinh thiết các khối u (u vú, u giáp, u gan...) Các công trình khoa học đã công bố (Thống kê số lượng bài đăng trên các tạp chí khoa học trong nước và quốc tế, số báo cáo, tham gia các Hội nghị khoa học trong nước và quốc tế). Số bài báo khoa học được đăng

2.20. Giao diện Trang chủ

The screenshot shows the main page of the mobile application. At the top, there is a header bar with icons for battery, signal, and time (22:56). The title "Trang chủ" is displayed. Below the header, there is a search bar with the placeholder "Tìm kiếm...". A banner image of a doctor is visible. The main content area includes sections for "Bệnh viện" (with icons for KHÁM CHUYÊN KHOA and PHẪU KHẨU KHÁM), "Chuyên khoa" (with icons for Cơ xương khớp, Thần kinh, Tim mạch, Cố sưng), "Các cơ sở Y Tế" (with icons for Bệnh viện Hữu Nghị, Võ Đức, Bệnh viện Trung ương Quân đội 108, Bệnh viện...), and "Bác sĩ" (with icons for Trương Thị Thanh, Nguyễn Thị Hương, Phan Chính Tạo, Vũ...). There are also sections for "Các bài viết liên quan" (with posts by Lê Tuấn Linh) and "Hình ảnh" (with a thumbnail image of a person).

2.21. Giao diện Thông tin chi tiết Cơ sở y tế



2.22. Giao diện xem lịch hẹn



2.23. Giao diện Chi tiết lịch hẹn



2.24. Giao diện Đăng bài viết



2.25. Giao diện Thông tin chi tiết Bác sĩ

← 22:56 Thông tin bác sĩ 71

Chuyên khoa:

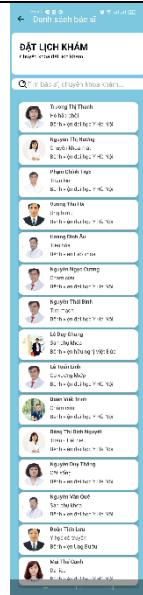
Hô hấp phổi

Trương Thị Thanh
Mã số giấy phép: TTT123456789
Email: thanhtt@gmail.com
Liên hệ: 0869999888

Địa chỉ khám
Bệnh viện đại học Y Hà Nội
Giá khám: 230.000 đ

Bác sĩ chuyên khoa cấp II, Bác sĩ nội trú. Trương Thị Thanh Bác sĩ Trung tâm CĐHA& CTĐQ – BV ĐH Y Hà Nội Tóm tắt sơ lược về quá trình đào tạo, quá trình công tác, các thành tích và kinh nghiệm nghiên cứu khoa học nổi bật: 2007-2013: Sinh viên hệ bác sĩ đa khoa Trường Đại học Y Hà Nội 2013-2016: Học Bác sĩ nội trú chuyên ngành chẩn đoán hình ảnh. 2016- đến nay: Bác sĩ TT Chẩn đoán hình ảnh và CTĐQ - BV Đại học Y Hà Nội 2020- 2022: Bác sĩ Chuyên khoa II tại Trường Đại học Y Hà Nội Tháng 6 đến tháng 12 năm 2018: đi học chẩn đoán hình ảnh tại đại học Ajou Hàn Quốc Chuyên môn sâu - Đốt sóng cao tần (RFA) điều trị nhân tuyến giáp dưới hướng dẫn siêu âm. - Điều trị suy tĩnh mạch chi dưới bằng phương pháp Laser, tiêm xơ - Hút u vú có hỗ trợ áp lực âm (VABB) - Chọc hút tế bào, sinh thiết các khối u (u vú, u giáp, u gan...) Các công trình khoa học đã công bố (Thống kê số lượng bài đăng trên các tạp chí khoa học trong nước và quốc tế, số báo cáo tham gia các Hội nghị khoa học trong nước và quốc tế). Số bài báo khoa học được đăng

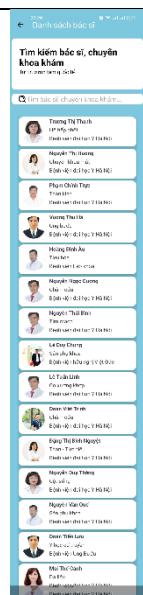
2.26. Giao diện Danh sách bác sĩ



2.27. Giao diện Chuyên khoa khám



2.28. Giao diện Tìm kiếm



2.29. Giao diện sửa thông tin bác sĩ

22:55 22/09/2023

Sửa mới Bác Sĩ

Chọn ảnh

0987654321

linhlt@gmail.com

Bệnh viện đại học Y Hà Nội

Cơ xương khớp

LTL1234567890

nhấn để xem ảnh

300.000 đ

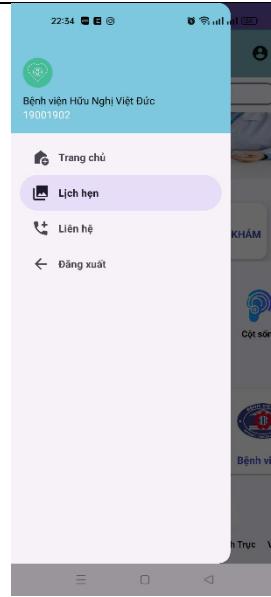
Tiến sĩ. Bác sĩ nội trú Lê Tuấn Linh
Giám đốc trung tâm CDHA và can thiệp điện quang – BV DH Y Hà Nội. Giảng viên chính, Phó chủ nhiệm bộ môn CDHA ĐHYHN. Giới thiệu Là bác sĩ chuyên ngành chẩn đoán hình ảnh, được đào tạo liên tục tại trường Đại học Y Hà Nội từ bậc Đại học, đến BSNT và Tiến sĩ. Hiện là giảng viên cơ hữu của bộ môn CDHA trường Đại học Y Hà Nội. Là một trong những bác sĩ đầu tiên được phân công về công tác kiêm nhiệm tại bệnh viện Đại học Y Hà Nội từ ngày đầu tiên thành lập đến nay, góp phần xây dựng khoa CDHA mà nay là Trung tâm Chẩn đoán hình ảnh và Can thiệp điện quang- BV ĐHYHN trở thành một đơn vị lớn mạnh trong khám chữa bệnh, đào tạo và nghiên cứu khoa học. Ngoài tham gia công tác tại trường đại học Y Hà Nội còn tham gia hỗ trợ đào tạo, hỗ trợ chuyên môn cho nhiều đơn vị y tế trong cả nước. Slogan: Innovation are Rad! Quá trình đào tạo 1995-2001. Sinh viên hệ đa khoa, trường ĐHYHN 2002-2006. BSNT chuyên ngành CDHA, ĐHYHN 2004-2005. Thực tập BSNT chuyên ngành CDHA tại Pháp 2011-2019. Tiến sĩ CDHA tại Trường Đại học Y Hà Nội Quá trình công tác 01/2007 đến nay Giảng viên bộ môn CDHA Chẩn đoán hình ảnh Trường ĐH YHN 08/2007 đến nay Bác sĩ khoa CDHA Khoa Chẩn đoán hình ảnh - BVĐHYHN Số 1A, Tôn Thất Tùng, Đồng Da, HN 08/2011 đến 03/2017 Phó khoa CDHA Khoa Chẩn đoán hình ảnh - BVĐHYHN Số 1A, Tôn Thất Tùng, Đồng Da, HN 03/2017 đến 12/2017 Phụ trách khoa CDHA Khoa Chẩn đoán hình ảnh - BVĐHYHN Số 1A, Tôn Thất Tùng, Đồng Da, HN 12/2017 đến 12/2020 Trưởng khoa CDHA Khoa Chẩn đoán hình ảnh - BVĐHYHN Số 1A, Tôn Thất Tùng, Đồng Da, HN 11/2019 đến nay Phó trưởng Bộ môn Bộ môn Chẩn đoán hình ảnh- Trường ĐHY Hà Nội Số 1A, Tôn Thất Tùng, Đồng Da, HN 12/2020 đến nay Giám đốc Trung tâm Trung tâm Chẩn đoán hình ảnh và Can thiệp điện quang -BVĐHYHN Số 1A, Tôn Thất Tùng, Đồng Da, HN 03/2021 đến nay Trưởng phòng khám Phòng khám đa khoa BV ĐHYHN cơ sở Quận Cầu Giấy Trương Công Giai, Cầu Giấy, HN Chuyên môn, kỹ thuật chuyên sâu Chẩn đoán hình ảnh da khoa Chẩn đoán hình ảnh hệ tiết niệu, Ứng dụng Công nghệ thông tin trong Chẩn đoán hình ảnh. Ứng dụng AI trong chẩn đoán hình ảnh Sách, sách chuyên khảo, giáo trình Thư kí biên soạn cuốn Bài giảng CDHA dành cho hệ bác sĩ đa khoa, nhà xuất bản y học, năm 2009. Các công trình khoa học đã công bố Là tác giả, đồng tác giả hơn 70 bài báo khoa học trong và ngoài nước, tham gia gần 10 công trình nghiên cứu các cấp, tham gia biên soạn nhiều sách phục vụ trong đào tạo cho đại học và sau đại học của trường Đại học Y Hà Nội. Đề tài KH&CN các cấp đã chủ trì hoặc tham gia Chủ trì 01 đề tài cấp cơ sở nghiệm thu 2016. Tham gia 02 đề tài cấp bộ. Tham gia 02 đề tài cấp nhà nước Thành viên tổ chức: Phó trưởng chi hội Chẩn đoán hình ảnh ổ bụng - Hội Điện quang và y học hạt nhân Việt Nam Ngôn ngữ Tiếng Việt, Tiếng Anh, Tiếng Pháp. Khen thưởng Bằng khen của tỉnh ủy Hà Giang Bằng khen của Trung ương Đoàn, Bộ y tế, UBDN tỉnh Lào Cai

Lưu

2.30. Giao diện Trang cá nhân



2.31. Giao diện Menu

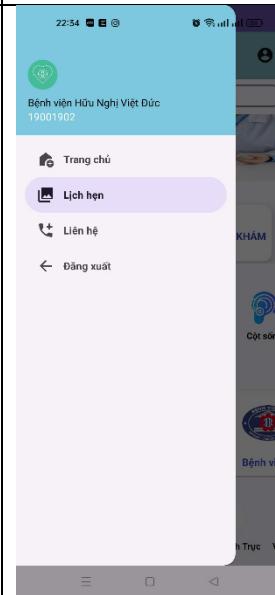


3.1.3. Màn hình giao diện Cơ sở y tế

2.32. Giao diện Trang chủ



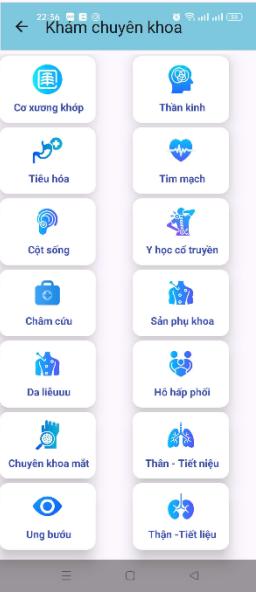
2.33. Giao diện Menu



2.34. Giao diện tìm kiếm bác sĩ



2.35. Giao diện Thông tin chi tiết bác sĩ



2.36. Giao diện Chuyên khoa khám



2.38. Giao diện thông tin chi tiết CSYT



2.39. Giao diện sửa thông tin CSYT



2.40. Giao diện Thông tin chuyên khoa



Chọn ảnh

Y học cổ truyền

Nếu các bệnh lý tim mạch là nguyên nhân hàng đầu gây tử vong thì bệnh lý cơ xương khớp đứng đầu trong một loạt nguyên nhân dẫn đến tàn phế. Xã hội càng hiện đại, số ca mắc các bệnh lý này càng cao. Chẳng những vậy, bệnh xương khớp giờ đây đã không còn là căn bệnh của riêng người già mà có xu hướng ngày càng trẻ hóa. Việt Nam được xếp vào nhóm những quốc gia có tỷ lệ mắc bệnh xương khớp cao nhất thế giới. Số liệu thống kê cho thấy có khoảng 30% người trên 35 tuổi, 60% người trên 65 tuổi và 85% người trên 80 tuổi bị thoái hóa khớp – một trong những bệnh lý xương khớp phổ biến nhất ở người ta. Tuy nhiên, những bệnh lý cơ xương khớp nếu được chẩn đoán, điều trị sớm và đúng cách thì có thể điều trị hiệu quả. Người bệnh không còn đau đớn và không gặp phải nguy cơ biến dạng chi thể hay tàn phế. Vì vậy, ngày càng có nhiều bệnh nhân xương khớp chủ động đi tầm soát, thăm khám để cải thiện triệu chứng và phòng ngừa biến chứng bệnh. Khoa Cơ xương khớp Bệnh viện Đa khoa Tâm Anh được thành lập để đáp ứng nhu cầu khám, tầm soát và điều trị bệnh cơ xương khớp đang ngày càng gia tăng ở Việt Nam. Quy tụ đội ngũ chuyên gia có chuyên môn cao, giàu kinh nghiệm thực tiễn, mỗi y bác sĩ của khoa Cơ cơ xương khớp BVĐK Tâm Anh không ngừng học tập và nghiên cứu, áp dụng các phương pháp chẩn đoán và điều trị tiên tiến vào từng bệnh nhân nhằm tối ưu hóa quá trình khám chữa bệnh. Bên cạnh đó, Khoa luôn chú trọng cải tiến chất lượng dịch vụ nhằm mang đến sự hài lòng tuyệt đối cho người bệnh.

Lưu

Thông tin CSYT

Bệnh viện Hữu Nghị Việt Đức

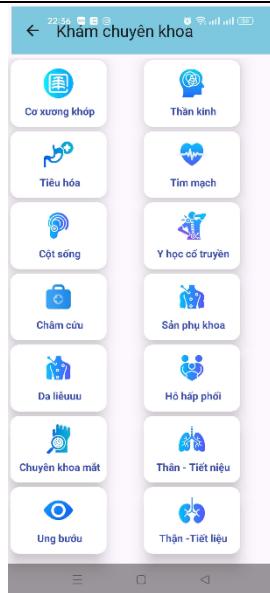


Thông tin cơ sở y tế

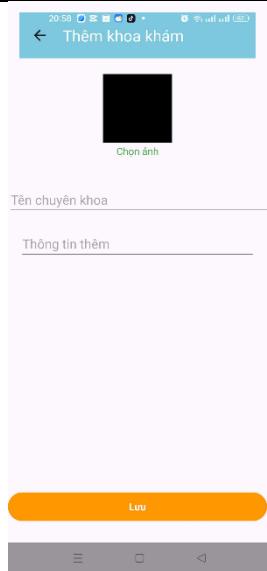
Quản lý chuyên khoa khám

⋮ ⌂ ⌁

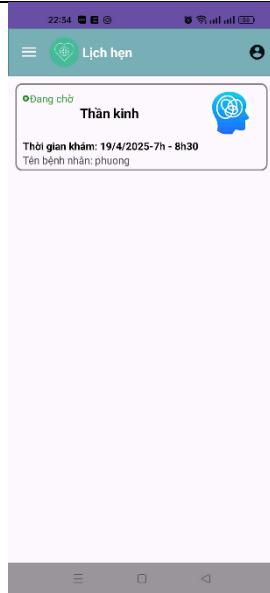
2.42. Giao diện Hiển thị danh sách chuyên khoa



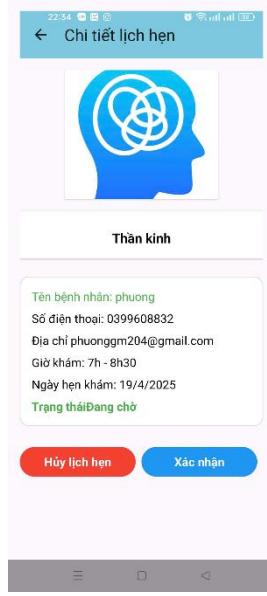
2.43. Giao diện Thêm thông tin Chuyên khoa



2.44. Giao diện xem lịch hẹn



2.45. Giao diện thông tin đặt lịch



3.1.4. Màn hình giao diện Admin

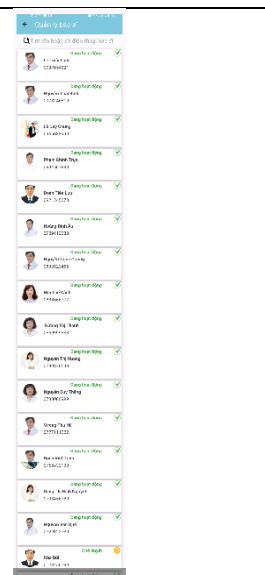
2.46. Giao diện Trang chủ



2.47. Giao diện Đăng nhập



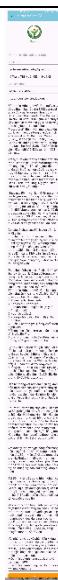
2.48. Giao diện Hiển thị danh sách bác sĩ



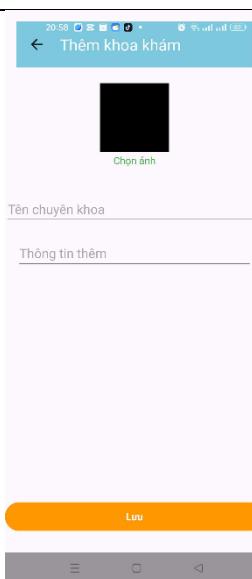
2.49. Giao diện thông tin chi tiết bác sĩ



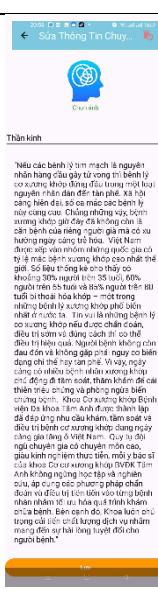
2.50. Giao diện Sửa thông tin cơ sở y tế



2.51. Giao diện Thêm thông tin chi tiết chuyên khoa

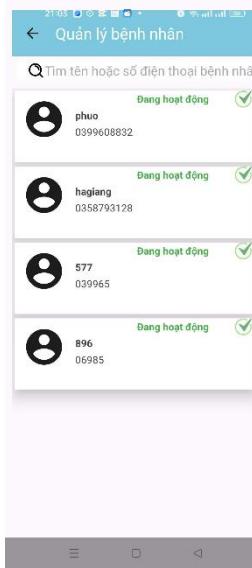


2.52. Giao diện Sửa thông tin chi tiết chuyên khoa



<p>2.53. Giao diện Hiển thị danh sách chuyên khoa</p>	
<p>2.54. Giao diện Hiển thị danh sách bác sĩ</p>	
<p>2.55. Giao diện Hiển thị danh sách CSYT</p>	

2.56. Giao diện quản lý bệnh nhân



2.57. Giao diện quản lý bài viết



3.4. Code minh họa các chức năng cốt lõi

3.4.1 Chức năng đăng ký

```
btnsigup.setOnClickListener(e->{
    String sdt = edtsdtsignup.getText().toString().trim();
    if(sdt.length() != 10 || !sdt.startsWith("0")){
        edtsdtsignup.setError("Vui lòng nhập số điện thoại hợp lệ");
        edtsdtsignup.requestFocus();
    }
})
```

```

        }
        sendOtp(sdt);
    //    signuprealtime();
    });

tologin.setOnClickListener(e->startActivity(new Intent(SignUpActitvity.this,
LoginActivity.class)));
}

private void sendOtp(String sdt) {
    PhoneAuthProvider.getInstance().verifyPhoneNumber(
        "+84" + sdt.substring(1),
        60,
        TimeUnit.SECONDS,
        this,
        new PhoneAuthProvider.OnVerificationStateChangedCallbacks() {
            @Override
            public void onVerificationCompleted(@NonNull PhoneAuthCredential
phoneAuthCredential) {
                // Tự động xác minh (nếu được) không cần nhập OTP
                signInWithPhoneCredential(phoneAuthCredential);
            }
        }

        @Override
        public void onVerificationFailed(@NonNull FirebaseException e) {
            Toast.makeText(SignUpActitvity.this, "Xác minh thất bại: " +
e.getMessage(), Toast.LENGTH_LONG).show();
            Log.e("OTP", "Verification failed", e);
        }
    }

    @Override

```

```
public void onCodeSent(@NonNull String verificationId,
    @NonNull PhoneAuthProvider.ForceResendingToken token)
{
    super.onCodeSent(verificationId, token);
    storedVerificationId = verificationId;
    resendingToken = token;
    showOtpDialog(verificationId); //  Gọi dialog nhập OTP ở đây
}
});
```

```
private void signInWithPhoneCredential(PhoneAuthCredential credential) {
    mAuth.signInWithCredential(credential).addOnCompleteListener(this, task -> {
        if (task.isSuccessful()) {
            // Đúng mã OTP → thực hiện lưu dữ liệu người dùng
            signuprealtime(); // hoặc hàm bạn dùng để lưu tài khoản
        } else {
            Toast.makeText(this,
                "Xác minh OTP thất bại", Toast.LENGTH_SHORT).show();
        }
    });
}
```

```
private void showOtpDialog(String verificationId){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Nhập mã OTP");

    final EditText input = new EditText(this);
    input.setInputType(InputType.TYPE_CLASS_NUMBER);
    input.setHint("Mã OTP gồm 6 chữ số");
```

```
        input.setPadding(50, 40, 50, 10);
        builder.setView(input);

        builder.setPositiveButton("Xác minh", (dialog, which) -> {
            String otp = input.getText().toString().trim();
            if (!otp.isEmpty()) {
                PhoneAuthCredential credential =
                    PhoneAuthProvider.get Credential(verificationId, otp);
                signInWithPhoneCredential(credential);
            } else {
                Toast.makeText(this, "Vui lòng nhập mã OTP",
                    Toast.LENGTH_SHORT).show();
            }
        });

        builder.setNegativeButton("Hủy", (dialog, which) -> dialog.dismiss());
    }

    builder.show();
}

private void signuprealtime() {
    reference = FirebaseDatabase.getInstance().getReference("Users");

    String name = edtnamesignup.getText().toString().trim();
    String sdt = edtsdtsignup.getText().toString().trim();
    String pass = edtpasssignup.getText().toString().trim();
    String as = "user";
    String status = "Đang hoạt động";
    String banla = spinrole.getSelectedItem().toString();
}
```

```
if (banla.equals("bệnh nhân")) {  
    as = "user";  
    status = "Đang hoạt động";  
  
} else if(banla.equals("bác sĩ")) {  
    as = "bacsi";  
    status = "Chờ duyệt";  
  
} else {  
    as = "csyt";  
    status = "Đang hoạt động";  
  
}  
  
// Kiểm tra dữ liệu trước khi đăng ký
```

```
if (name.isEmpty()) {  
    edtnamesignup.setError("Please enter name");  
    return;  
}  
  
if (sdt.isEmpty()) {  
    edtsdtsignup.setError("Please enter phone number");  
    return;  
}  
  
if (pass.isEmpty()) {  
    edtpasssignup.setError("Please enter password");  
    return;  
}  
  
if (pass.length() < 6) {  
    edtpasssignup.setError("Password must be at least 6 characters");  
    return;  
}
```

```

String token = "";

account acc = new account(name, sdt, pass, as, status, token);

FirebaseHelper.addAccount(acc, new FirebaseCallBack<account>(){

    @Override

    public void onSuccess(account data) {

        // Them 1 bac si hoac benh nhan dựa trên role của account (them len fire
        base)

        if(acc.getAs().equals("user")){
            benhnhan bn = new benhnhan(name, sdt, "", "", "", "", "", "");
            FirebaseHelper.addbenhnhan(bn, new FirebaseCallBack(){

                @Override

                public void onSuccess(Object data) {

                }

                @Override

                public void onFailure(String message) {

                }

            });

        } else if (acc.getAs().equals("bacsi")) {
            Bacsi bs = new Bacsi( name, "", "", "", "", "", "", "", sdt);
            FirebaseHelper.addBacsi(bs, new FirebaseCallBack() {

                @Override

                public void onSuccess(Object data) {

                }

            }

            @Override

            public void onFailure(String message) {

            }

        }

    }

});
```

```
    } else{
        Cosoyte csyt = new Cosoyte(name, "", "", "", "", "", "", sdt, "");
        FirebaseHelper.insertcosoyte(csyt, new FirebaseCallBack() {
            @Override
            public void onSuccess(Object data) {
                }

                @Override
                public void onFailure(String message) {
                    }

                    });

                }

                Toast.makeText(SignUpActivity.this, "Đăng ký thành công",
                Toast.LENGTH_SHORT).show();
                startActivity(new Intent(SignUpActivity.this, LoginActivity.class));
                finish();
            }

            @Override
            public void onFailure(String message) {
                Toast.makeText(SignUpActivity.this, "Đăng ký thất bại",
                Toast.LENGTH_SHORT).show();
            }

        });
    }
}
```

3.4.2 Chức năng đăng nhập , đăng xuất

// Chức năng đăng nhập:

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    EdgeToEdge.enable(this);  
    setContentView(R.layout.activity_login);  
    anhxa();  
    auth = FirebaseAuth.getInstance();  
    edtpasslogin.setOnEditorActionListener((v, actionId, event) -> {  
        hideKeyboard();  
        return false;  
    });  
    btnlogin.setOnClickListener(e->login());  
    tosigup.setOnClickListener(e->startActivity(new Intent(LoginActivity.this,  
    SignUpActitvity.class)));
```

// cap nhap mat khau moi len realtime neu co

}

```
private void checkemail(EditText email){  
    if(email.getText().toString().trim().isEmpty()){  
        email.setError("Vui lòng nhập số điện thoại");  
        email.requestFocus();  
        return ;  
    }  
    if(!Patterns.EMAIL_ADDRESS.matcher(email.getText().toString().trim()).matches()){
```

```
        email.setError("Vui lòng nhập đúng định dạng emial");
        email.requestFocus();
        return ;
    }

auth.sendPasswordResetEmail(email.getText().toString().trim()).addOnCompleteListener
(task->{
    if(task.isSuccessful()){
        Toast.makeText(LoginActivity.this, "Đã gửi email thành công",
        Toast.LENGTH_SHORT).show();
    }
});

private void login() {
    String sdt = edtsdtlogin.getText().toString().trim();
    String pass = edtpasslogin.getText().toString().trim();

    if(!validate(sdt, pass)) return ;

    reference = FirebaseDatabase.getInstance().getReference("Users");
    Query checkuser = reference.orderByChild("phone").equalTo(sdt);

    checkuser.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            if(snapshot.exists()){
                for (DataSnapshot ds: snapshot.getChildren()){
                    String passfromdb = ds.child("pass").getValue(String.class);

```

```
if(passfromdb!=null && passfromdb.equals(pass)){
    //lấy role
    String asdb = ds.child("as").getValue(String.class);
    String status = ds.child("status").getValue(String.class);

    if(status.equals("Tạm khóa")){
        Toast.makeText(LoginActivity.this, "Tài khoản đã bị khóa",
LENGTH_SHORT).show();

    } else{
        if( asdb.equals("admin")){
            Toast.makeText(LoginActivity.this, "Iam admin",
LENGTH_SHORT).show();

            Intent intent = new Intent(LoginActivity.this, AdminActivity.class);
            intent.putExtra("role", "admin");
            startActivity(intent);
            finish();
        } else {
            String name = ds.child("name").getValue(String.class);
            String phone = ds.child("phone").getValue(String.class);

            Intent intent = new Intent(LoginActivity.this, UserActivity.class);
            intent.putExtra("iduser", sdt);
            intent.putExtra("name", name);
            intent.putExtra("phone", phone);
            intent.putExtra("role", asdb);
            startActivity(intent);
            finish();
        }
    }
}
```

```
        }
    }
} else{
    Toast.makeText(LoginActivity.this, "Tài khoản không tồn tại",
LENGTH_SHORT).show();
}
```

```
@Override
public void onCancelled(@NonNull DatabaseError error) {
}
});
```

```
}
```

```
private void hideKeyboard() {
    InputMethodManager imm = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);
    if (imm != null) {
        imm.hideSoftInputFromWindow(edtpasslogin.getWindowToken(), 0);
    }
}
```

```
private boolean validate (String sdt, String pass){
```

```
    if(sdt.isEmpty()){
        edtsdtlogin.setError("Vui lòng nhập số điện thoại");
        edtsdtlogin.requestFocus();
        return false;
    }
    if (pass.isEmpty()){
        edtpasslogin.setError("Vui lòng nhập mật khẩu");
```

```

        edtpasslogin.requestFocus();
        return false;
    }
    return true;
}

// Chức năng đăng xuất
btndangxuat.setOnClickListener(v -> {
    UserActivity.iduser = null;
    Intent intent = new Intent(ThongtinUser.this, LoginActivity.class);
    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
    Intent.FLAG_ACTIVITY_CLEAR_TASK);
    startActivity(intent);
    finish();
});

```

3.4.3 Chức năng đặt lịch hẹn, xác nhận, hủy lịch hẹn

// Đặt lịch hẹn

```
btndatlichkham.setOnClickListener(e->{
```

```

    String iduser = UserActivity.iduser;
    if(!validate()) return;
    String sdt = "";
    String img = "";
    String name = "";
    if(bacsi != null){
        sdt = bacsi.getSdt();
        img = bacsi.getImg();
        name = bacsi.getName();
    }
}
```

```
    } else {
        sdt = ckhoakham.getSdtcsyt();
        img = ckhoakham.getImg();
        name = ckhoakham.getTenkhoa();
    }
}
```

```
lichhen lh = new lichhen(
    iduser,
    sdt,
    edtngay.getText().toString(),
    selectedText, "Đang chờ",
    edthoten.getText().toString(),
    edtsdt.getText().toString(),
    edtdiachi.getText().toString(),
    img,
    name
);
```

```
FirebaseHelper.addlichhen(lh, new FirebaseCallBack() {
```

```
    @Override
    public void onSuccess(Object data) {
```

```
}
```

```
    @Override
    public void onFailed(String message) {
    }
});
```

```
if(bacsi!= null )  
    FirebaseHelper.getaccbyid(bacsi.getSdt(), new FirebaseCallBack<accout>() {  
        @Override  
        public void onSuccess(accout data) {  
            String token = data.getToken();  
            FCMHelper.sendFCM(token, "Bạn có lịch hẹn mới", "Hãy kiểm tra ứng dụng!")  
                .addOnSuccessListener(response -> {  
                    Log.d("FCM", "Success: " + response);  
                })  
                .addOnFailureListener(e -> {  
                    Log.e("FCM", "Error", e);  
                });  
        }  
  
        @Override  
        public void onFailed(String message) {  
            }  
        });  
        finish();  
    });  
  
    // Hủy và xác nhận lịch hẹn  
    bttnutHuyLichHen.setOnClickListener(v -> {  
        FirebaseHelper.deletelichhen(lh.getId());  
        FirebaseHelper.getaccbyid(lh.getIdbenhnhan(), new FirebaseCallBack<accout>() {  
            @Override  
            public void onSuccess(accout data) {
```

```

        String token = data.getToken();
        FCMHelper.sendNotification(lichhenDetail.this, token, "bacsi: " +
        lh.getNamebs(), "đã hủy lịchhen");

    }

    @Override
    public void onFailed(String message) {

    }
);

finish();
});

if(UserActivity.roleuser.equals("user") || lh.getTrangthai().equals("Xác nhận")){
    btnxacnhan.setVisibility(View.GONE);
}

btntaxacnhan.setOnClickListener(v -> {
    FirebaseHelper.updateTrangThai(lh.getId(), "Xác nhận");

    // get token by idbenhnhan
    FirebaseHelper.getaccbyid(lh.getIdbenhnhan(), new FirebaseCallBack<accout>() {
        @Override
        public void onSuccess(accout data) {
            String token = data.getToken();

            FCMHelper.sendFCM(token, "Bạn có lịch hẹn mới", "Hãy kiểm tra ứng dụng!")
                .addOnSuccessListener(response -> {
                    Log.d("FCM", "Success: " + response);
                })
        }
    });
}

```

```
.addOnFailureListener(e -> {
    Log.e("FCM", "Error", e);
});

}

@Override
public void onFailed(String message) {

}

});
```

3.4.4 Chức năng tìm kiếm

// Tìm kiếm bác sĩ theo chuyên khoa, tên

```
search_bar1.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
        String query = search_bar1.getText().toString().trim();
        filterList(query);
    }

    @Override
    public void afterTextChanged(Editable s) {

    }
});
```

```
}
```

```
private void filterList(String query) {
    ArrayList<accout> listacc = new ArrayList<>();
    FirebaseHelper.getaccoutbyStatusAndRoleinh("Đang hoạt động", "bacsi", new
    FirebaseCallBack<ArrayList<accout>>() {
        @Override
        public void onSuccess(ArrayList<accout> listAcc) {
            listacc.clear();
            listacc.addAll(listAcc);

            // Bước 2 sẽ làm trong này
            Log.d("TAG", "Lấy được acc: " + listAcc.size());

            listbacsi.clear();
            int[] count = {0};

            Log.d("title", title + "");

            if(title.equals("ĐẶT LỊCH KHÁM") || title.equals("Tìm kiếm bác sĩ, chuyên khoa
            khám")){
                listbacsi.clear();
                for (accout acc : listAcc) {
                    FirebaseHelper.getBacsiBySdt(acc.getPhone(), new
                    FirebaseCallBack<Bacsi>() {
```

```

@Override
public void onSuccess(Bacsi data) {
    listbacsi.add(data);
    count[0]++;
    if (count[0] == listAcc.size()) {
        applyFilter(query); // Gọi lọc ban đầu không có query
    }
}

@Override
public void onFailed(String message) {
    count[0]++;
    if (count[0] == listAcc.size()) {
        applyFilter(query); // Gọi lọc nếu acc cuối cùng bị fail
    }
}

else {
    listbacsi.clear();
    for (accout acc : listAcc) {
        Log.d("TAG", "acc phone: " + acc.getPhone());
        FirebaseHelper.getBacsiByChuyenkhoaAndSdt(title, acc.getPhone(), new
FirebaseCallBack<Bacsi>() {
            @Override
            public void onSuccess(Bacsi bacsi) {
                if (bacsi != null) listbacsi.add(bacsi);
                count[0]++;
                if (count[0] == listAcc.size()) {

```

```

        applyFilter(query); // Gọi lọc ban đầu không có query
    }
}

@Override
public void onFailed(String message) {
    count[0]++;
    if (count[0] == listAcc.size()) {
        applyFilter(query); // Gọi lọc nếu acc cuối cùng bị fail
    }
}
});

}

}

@Override
public void onFailed(String message) {
    Log.e("TAG", "Không lấy được acc: " + message);
}
});

}

private void applyFilter(String query) {
    filteredList.clear();
    Log.d("size list bacsi", "" + listbacsi.size());
    if (query.isEmpty()) {
        filteredList.addAll(listbacsi);
    }
}

```

```

} else {
    for (Bacsi b : listbacsi) {
        if (b.getName().toLowerCase().contains(query.toLowerCase()) ||
            b.getChuyenkhoa().toLowerCase().contains(query.toLowerCase())) {
            filteredList.add(b);
        }
    }
}

adapter.updateList(filteredList); // Cập nhật hiển thị danh sách
}

// Tìm kiếm theo tên cơ sở y tế
edtsearch.addTextChangedListener(new TextWatcher() {

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {

        // listbacsi.clear();
        String query = edtsearch.getText().toString().trim();
        // filterList(query);

        if (manager != null && manager.equals("quanlybacsi")) {

            filterList(query, "bacsi");
        } else if (manager != null && manager.equals("quanlybenhhan")) {
            filterList(query, "user");
        } else if (manager != null && manager.equals("quanlycsyt")) {
            filterListcsyt(query);
        }
        else if(manager != null && manager.equals("quanlybv")){
    }
}

```

```

// get bai viet by id bacsi

}

}

private void filterListcsyt(String query) {
ArrayList<Cosoyte> newlist = new ArrayList<>();
listcsyte.clear();

FirebaseHelper.getcosoyte(new FirebaseCallBack<ArrayList<Cosoyte>>() {
    @Override
    public void onSuccess(ArrayList<Cosoyte> data) {
        listcsyte.clear();
        listcsyte.addAll(data);

        if( query.isEmpty()){
            newlist.addAll(listcsyte);
        }
        else{
            for (Cosoyte csyt : listcsyte) {
                if (csyt.getName().toLowerCase().contains(query.toLowerCase())) {
                    newlist.add(csyt);
                }
            }
            listcsyte.clear();
            listcsyte.addAll(newlist);
            adaptercsyt.notifyDataSetChanged();
        }
    }
}

```

```
    @Override  
    public void onFailed(String message) {  
  
    }  
});  
}  
}
```

3.4.5 Chức năng hiển thị các bác sĩ “Đang hoạt động”

// Chỉ các bác sĩ được admin duyệt thành đang hoạt động mới được hiển thị lên sàn để bệnh nhân đặt hẹn

```
adapterDoctors.setItemClickListener(bacsi->{
```

```
    String itemName = bacsi.getName();  
    Intent intent;  
    intent = new Intent(getActivity(), Bacsi_details.class);  
    intent.putExtra("bacsi", bacsi);  
    intent.putExtra("title", itemName);  
    intent.putExtra("anh", bacsi.getImg());  
  
    startActivity(intent);  
});
```

```
FirebaseHelper.getAccountByStatusAndRole( "Đang hoạt động", "bacsi", new  
FirebaseCallBack<ArrayList<accout>>()
```

```
    {  
        @Override  
        public void onSuccess(ArrayList<accout> data) {  
            listaccout.clear();  
            listaccout.addAll(data);  
        }  
    }
```

```

// get listbac bysdt
listDoctors.clear();
for (accout accout : listaccout) {
    FirebaseHelper.getBacsibySdt(accout.getPhone(), new
FirebaseCallBack<Bacsi>(){
    @Override
    public void onSuccess(Bacsi data) {
        listDoctors.add(data);
        adapterDoctors.notifyDataSetChanged();

    }
    @Override
    public void onFailure(String message) {
        Toast.makeText(getApplicationContext(), message, Toast.LENGTH_SHORT).show();
    }
});

}
}

@Override
public void onFailure(String message) {
    Toast.makeText(getApplicationContext(), message, Toast.LENGTH_SHORT).show();
}
});

```

3.4.6 Sửa thông tin cá nhân (Bệnh nhân, bác sĩ, Csyt)

// hồ sơ y tế

```

// get benhnhan by iduser
FirebaseHelper.getbenhnhanBySdt(iduser, new FirebaseCallBack<benhnhan>() {
    @Override
    public void onSuccess(benhnhan data) {

```

```

        edthotencsyt.setText(data.getTen());
        edtsdthsyt.setText(data.getSodienthoai());
        edtdiachiahsyt.setText(data.getDiachi());
        edtngaysinhhsyt.setText(data.getNgaysinh());
        edtbenhlyne.setText(data.getBenhlynen());

        avatar = data.getImg();
        if (avatar != null && !avatar.isEmpty()) {
            Glide.with(imghsyt.getContext())
                .load(Uri.parse(avatar)) // Chuyển String thành Uri
                .circleCrop()
                .into(imghsyt);
        } else{
            imghsyt.setImageResource(R.drawable.baseline_account_circle_24);
        }

        if(data.getGioitinh()!= null && data.getGioitinh().equals("Nam")){
            rdgioitinh.check(R.id.rdbtnnam);
        }else{
            rdgioitinh.check(R.id.rdbtnnu);
        }
    }

    @Override
    public void onFailure(String message) {
    });

```

```
imghsyt.setOnClickListener(view -> {
    // chon anh tu dien thoai
    Intent intent1 = new Intent(Intent.ACTION_PICK);
    intent1.setType("image/*");
    startActivityForResult(intent1, 1);

});
```

```
btnluuhhsyt.setOnClickListener(e->{
    String ten = edthotencsyt.getText().toString();
    String sdt = edtsdthsyt.getText().toString();

    String diachi = edtdiachiahsyt.getText().toString();
    String ngaysinh = edtngaysinhhsyt.getText().toString();
    String benhlynen = edtbenhlyne.getText().toString();
    // get gioi tinh
    int selectedId = rdgioitinh.getCheckedRadioButtonId();
    String gioitinh = "";
    if (selectedId == R.id.rdbtnnam) {
        gioitinh = "Nam";
    } else if (selectedId == R.id.rdbtnnu) {
        gioitinh = "Nữ";
    }

    if (imageUri != null) {
```

```
        avatar = imageUri.toString();
    }

    benhnhan bn = new benhnhan(ten, sdt, diachi, gioitinh, ngaysinh, benhlynen,
avatar);

    FirebaseHelper.updatebenhnha(bn, new FirebaseCallBack() {

        @Override
        public void onSuccess(Object data) {

        }

        @Override
        public void onFailure(String message) {

        }
    );
    finish();
});

}

private void anhx() {
    tbhsyt = findViewById(R.id.tbhsyt);
    imghsyt = findViewById(R.id.imghsyt);
    edthotencsyt = findViewById(R.id.edthotencsyt);
    edtsdthsyt = findViewById(R.id.edtsdthsyt);
    edtdiachiahsyt = findViewById(R.id.edtdiachiahsyt);
    edtngaysinhhsyt = findViewById(R.id.edtngaysinhhsyt);
    edtbenhlyne = findViewById(R.id.edtbenhlyne);
    btnluuhsyt = findViewById(R.id.btnluuhsyt);
    rdgioitinh = findViewById(R.id.rdgioitinh);
```

```

int selectedId = rggioitinh.getCheckedRadioButtonId();
//
edtngaysinhhsyt.setOnClickListener(e->showDatePickerDialog());
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 1 && resultCode == RESULT_OK && data != null) {
        imageUri = data.getData();
        imageUri = Xuly.copyImageToInternalStorage(this, imageUri); // Lấy URI của ảnh
        String uniquename = "image_" + System.currentTimeMillis() + ".jpg";
        Xuly.uploadImageToFirebaseStorage(this, imageUri, uniquename, download->{ });
    }

    Glide.with(this)
        .load(Uri.parse(imageUri.toString())) // Chuyển String thành Uri
        .placeholder(R.drawable.baseline_account_circle_24) // Ảnh mặc định nếu
        đang load
        .error(R.drawable.baseline_account_circle_24) // Ảnh mặc định nếu load thất
        bại
        .into(imghsyt);
    }
}

private void showDatePickerDialog() {
    final Calendar calendar = Calendar.getInstance();
    int year = calendar.get(Calendar.YEAR);
    int month = calendar.get(Calendar.MONTH);
    int day = calendar.get(Calendar.DAY_OF_MONTH);
}

```

```

// Hiển thị DatePickerDialog
DatePickerDialog datePickerDialog = new DatePickerDialog(this, (view, selectedYear,
selectedMonth, selectedDay) -> {
    String selectedDate = selectedDay + "/" + (selectedMonth + 1) + "/" + selectedYear;
    edtngaysinhhsyt.setText(selectedDate); // Hiển thị ngày chọn trong EditText
}, year, month, day);

Calendar minDateCalendar = Calendar.getInstance();
minDateCalendar.set(Calendar.HOUR_OF_DAY, 0);
minDateCalendar.set(Calendar.MINUTE, 0);
minDateCalendar.set(Calendar.SECOND, 0);
minDateCalendar.set(Calendar.MILLISECOND, 0);

datePickerDialog.getDatePicker().setMaxDate(minDateCalendar.getTimeInMillis());

datePickerDialog.show();
}

```

```

// Sửa thông tin bác sĩ

btnsua4csyt.setOnClickListener(e->{
    bacsi.setName(edtten4sua.getText().toString());
    bacsi.setSdt(edtsdtsua4.getText().toString());
    bacsi.setEmail(edtemailsua4.getText().toString());
    bacsi.setDiachi(edtdiachisua.getText().toString());
    bacsi.setSogiayphephanhnghe(edtmasogiayphepsua.getText().toString());
    bacsi.setGiaKham(edtgiakhamsua.getText().toString());
    bacsi.setThongtin(edtthongtin4sua.getText().toString());
    bacsi.setChuyenkhoa(spinchuyenkhoacsytsua.getSelectedItem().toString());
    if (imageUri != null) {

```

Phát triển ứng dụng cho thiết bị di động

```
        bacsi.setImg(imageUri.toString());
    }

FirebaseHelper.updateBacsi(bacsi, new FirebaseCallBack() {
    @Override
    public void onSuccess(Object data) {

    }

    @Override
    public void onFailure(String message) {

    });
}

finish();
});

}

// Sửa thông tin cơ sở y tế

btncsytsua.setOnClickListener(e->{
    String ten = edttencsytsua.getText().toString();
    String sdt = edtsdtcsytsua.getText().toString();
    String email = edtemailaddcsytsua.getText().toString();
    String diachi = edtdiachicsytsua.getText().toString();
    String chuyenkhoa = edtchuyenkhoacsytsua.getText().toString();
    String masogiayphep = edtmasogiayphepcsytsua.getText().toString();
    String website = edtwebsitecsytsua.getText().toString();
    String thongtin = edtthongtin4csytsua.getText().toString();

    csyt.setName(ten);

```

```

csyt.setSdt(sdt);
csyt.setEmail(email);
csyt.setDiachi(diachi);
csyt.setChuyenkhoa(chuyenkhoa);
csyt.setMasogiayphep(masogiayphep);
csyt.setWebsite(website);
csyt.setThongtin(thongtin);

FirebaseHelper.updatcosoyte(csyt, new FirebaseCallBack() {

    @Override
    public void onSuccess(Object data) {

    }

    @Override
    public void onFailed(String message) {

    }
});

finish();
});

}

```

3.4.7 Chức năng tạo, xóa bài viết

// tạo bài viết

```

btndangtai.setOnClickListener(view -> {
    String content = edtbaibnvt.getText().toString();

```

```
// timestamp = thoi gian hien tai
String timestamp = String.valueOf(System.currentTimeMillis());
baiviet.setContent(content);
baiviet.setTimestamp(timestamp);
baiviet.setIdUser(UserActivity.iduser);
baiviet.setTitle(edtTitleBv.getText().toString());
```

```
FirebaseHelper.addBaiviet(baiviet, new FirebaseCallBack() {
```

```
    @Override
```

```
    public void onSuccess(Object data) {
```

```
        finish();
```

```
}
```

```
    @Override
```

```
    public void onFailure(String message) {
```

```
}
```

```
});
```

```
});
```

```
// xóa bài viết
```

```
    @Override
```

```
    public void onItemClick(Baiviet baiviet) {
```

```
        FirebaseHelper.deleteBaiviet(baiviet.getId());
```

```
}
```

3.4.8 Các chức năng thêm (chuyên khoa)

```
btnaddchuyenkhoa.setOnClickListener(e->{
    String tenchuyenkhoa = edttenaddchuyenkhoa.getText().toString();
    String thongtin = edtthongtinaddchuyenkhoa.getText().toString();
    if(!validate1()){
        return;
    }
    ck.setTenchuyenkhoa( tenchuyenkhoa);
    ck.setThongtin(thongtin);

    FirebaseHelper.addchuyenkhoa(ck, new FirebaseCallBack() {
        @Override
        public void onSuccess(Object data) {
            }
        @Override
        public void onFailed(String message) {
            }
    });
    finish();
});
```

3.4.9 Các chức năng quản lý (bệnh nhân, csyt, bacsı, bài viết, chuyên khoa)

```
manager = intent.getStringExtra("manager");
```

```
if (manager != null && manager.equals("quanlybacsi")) {
```

```
    toolbar.setTitle("Quản lý bác sĩ");
```

```
    listacc = new ArrayList<>();
```

```
    adapteracc = new adapterAccout(listacc);
```

```
    rcvlisbsad.setAdapter(adapteracc);
```

```
    rcvlisbsad.setLayoutManager(new LinearLayoutManager(this));
```

```
    adapteracc.setOnItemSelectedListener(this);
```

```
    edtsearch.setHint("Tìm tên hoặc số điện thoại bác sĩ");
```

```
// get list acc by role = bac si
```

```
FirebaseHelper.getaccoutbyrole("bacsi", new FirebaseCallBack<ArrayList<accout>>()
```

```
{
```

```
    @Override
```

```
    public void onSuccess(ArrayList<accout> data) {
```

```
        listacc.clear();
```

```
        listacc.addAll(data);
```

```
        adapteracc.notifyDataSetChanged();
```

```
}
```

```
    @Override
```

```
    public void onFailed(String message) {
```

```
}
```

```
});
```

```
} else if(manager != null && manager.equals("quanlybenhnhan")){
```

```
toolbar.setTitle("Quản lý bệnh nhân");
listacc = new ArrayList<>();
adapteracc = new adapterAccout(listacc);
rvlisbsad.setAdapter(adapteracc);
rvlisbsad.setLayoutManager(new LinearLayoutManager(this));
adapteracc.setOnListener(this);
edtsearch.setHint("Tìm tên hoặc số điện thoại bệnh nhân");

FirebaseHelper.getAccountByRole("user", new FirebaseCallBack<ArrayList<accout>>()
{
    @Override
    public void onSuccess(ArrayList<accout> data) {
        listacc.clear();
        listacc.addAll(data);
        adapteracc.notifyDataSetChanged();

    }

    @Override
    public void onFailed(String message) {

    }
});

// size list

} else if (manager != null && manager.equals("quanlycsyt")) {
    toolbar.setTitle("QUẢN LÝ CSYT");
    listcsyte = new ArrayList<>();

    adaptercsyt = new adaptercosoyte(listcsyte);
```

```
rvvlisbsad.setAdapter(adaptercsyt);
rvvlisbsad.setLayoutManager(new GridLayoutManager(this, 2));
adaptercsyt.setOnItemClickListener(csyt->{
    Intent intent1 = new Intent(adDanhsachBs.this, ChitietCSYT.class);
    intent1.putExtra("cosoyte", csyt);
    startActivity(intent1);
});
```

```
FirebaseHelper.getcosoyte(new FirebaseCallBack<ArrayList<Cosoyte>>() {
```

```
    @Override
    public void onSuccess(ArrayList<Cosoyte> data) {
        listcsyte.clear();
        listcsyte.addAll(data);
        adaptercsyt.notifyDataSetChanged();
    }
}
```

```
    @Override
    public void onFailed(String message) {
```

```
    }
});
```

```
edtsearch.setHint("Tìm cơ sở y tế");
} else if(manager != null && manager.equals("quanlybv")){
    edtsearch.setHint("Tìm tên bác sĩ");
    toolbar.setTitle("Quản lý bài viết");
    listbaiviet = new ArrayList<>();
    adapter = new adapterBaiviet(listbaiviet);
    rvvlisbsad.setAdapter(adapter);
    rvvlisbsad.setLayoutManager(new LinearLayoutManager(this));
```

```
FirebaseHelper.getAllbaiviet(new FirebaseCallBack<ArrayList<Baiviet>>() {  
    @Override  
    public void onSuccess(ArrayList<Baiviet> data) {  
        listbaiviet.clear();  
        listbaiviet.addAll(data);  
        adapter.notifyDataSetChanged();  
    }  
  
    @Override  
    public void onFailed(String message) {  
    }  
});  
adapter.setOnItemClickListener(new adapterBaiviet.setItemClick() {  
    @Override  
    public void onItemClick(Baiviet baiviet) {  
    }  
  
    @Override  
    public void onImageavatarClick(Baiviet baiviet) {  
    }  
  
    @Override  
    public void onItemDatkhambClick(Baiviet baiviet) {  
    }  
  
    @Override  
    public void onItemDeleteClick(Baiviet baiviet) {  
    }  
}
```

```

        FirebaseHelper.deletebaiviet(baiviet.getId());

    }

});

}

public void onStatusLongClick(int position) {
    accout acc = listacc.get(position);
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Trạng thái tài khoản")
        .setItems(new String[]{"Đang hoạt động", "Chờ duyệt", "Tạm khóa"}, (dialog,
which) -> {
        String newStatus;
        if (which == 0) {
            newStatus = "Đang hoạt động";
        } else if (which == 1) {
            newStatus = "Chờ duyệt";
        } else {
            newStatus = "Tạm khóa";
        }

        // Cập nhật trạng thái
        acc.setStatus(newStatus);

        FCMHelper.sendNotification(adDanhsachBs.this, acc.getToken(), "Trạng thái
tài khoản đã ", newStatus);

        adapteracc.notifyDataSetChanged(); // Cập nhật lại RecyclerView

        // update lên firebase
    });
}

```

```
        reference = FirebaseDatabase.getInstance().getReference("Users");
        reference.child(acc.getPhone()).child("status").setValue(newStatus);

    })
.show();
}
```

@Override

```
public void onItemClick(int position) {
    // get bacsi by sdt
    if (manager != null && manager.equals("quanlybacsi")) {
        String sdt = listacc.get(position).getPhone();

        // Lấy bác sĩ theo số điện thoại để chuyển đến trang sửa
        FirebaseHelper.getBacsiBySdt(sdt, new FirebaseCallBack<Bacsi>() {
            @Override
            public void onSuccess(Bacsi data) {
                // Lưu đối tượng bacsi đã lấy được từ Firebase

                if(data == null){
                    Toast.makeText(adDanh sachBs.this, "Không tìm thấy bác sĩ",
                    Toast.LENGTH_SHORT).show();
                    return;
                }
                Intent intent = new Intent(adDanh sachBs.this, Bacsi_details.class);
                intent.putExtra("bacsi", data); // Truyền đối tượng bacsi vào Intent
            }
        });
    }
}
```

```

        startActivity(intent);
    }

    @Override
    public void onFailure(String message) {
        // Xử lý lỗi khi không lấy được bác sĩ từ Firebase
        Log.e("Firebase", "Lỗi: " + message);
    }
});

}

else if(manager != null && manager.equals("quanlybenhnhan")){
    Toast.makeText(this, "Bạn không có quyền xem tài khoản này",
    Toast.LENGTH_SHORT).show();
}

}

```

3.4.10 Tải ảnh lên Firebase Storage, copy ảnh vào bộ nhớ cache

```

import android.content.Context;
import android.net.Uri;
import android.widget.Toast;

import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import java.io.File;

```

```
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

public class Xuly {

    public static Uri copyImageToInternalStorage(Context context, Uri uri) {
        try {
            InputStream inputStream = context.getContentResolver().openInputStream(uri);
            if (inputStream == null) {
                Toast.makeText(context, "Không thể mở InputStream",
                    Toast.LENGTH_SHORT).show();
                return null;
            }

            // Tạo một file với tên duy nhất để tránh ghi đè
            File imageDir = new File(context.getFilesDir(), "images");
            if (!imageDir.exists()) {
                imageDir.mkdir();
            }

            File tempFile = new File(imageDir, "image_" + System.currentTimeMillis() +
                ".jpg");

            OutputStream outputStream = new FileOutputStream(tempFile);
            byte[] buffer = new byte[1024];
            int length;
            while ((length = inputStream.read(buffer)) > 0) {
                outputStream.write(buffer, 0, length);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

```
        }

        outputStream.close();
        inputStream.close();

        Uri fileUri = Uri.fromFile(tempFile);

        return fileUri;
    } catch (IOException e) {
        e.printStackTrace();
        Toast.makeText(context, "Lỗi khi sao chép ảnh",
Toast.LENGTH_SHORT).show();
        return null;
    }
}

public static void uploadImageToFirebaseStorage(Context context, Uri imageUri,
String imageName, OnSuccessListener<Uri> onSuccessListener) {
    StorageReference storageRef =
    FirebaseStorage.getInstance().getReference("images/" + imageName);

    storageRef.putFile(imageUri)
        .addOnSuccessListener(taskSnapshot -> {
            storageRef.getDownloadUrl().addOnSuccessListener(onSuccessListener);
        })
        .addOnFailureListener(e -> {
            Toast.makeText(context, "Lỗi khi tải ảnh lên Firebase",
Toast.LENGTH_SHORT).show();
        });
}

public static String getRelativeTime(String timestampStr) {
    if (timestampStr == null || timestampStr.isEmpty()) {
```

```

        return "Không rõ thời gian"; // Hoặc để trống tùy bạn
    }

try {
    long timestamp = Long.parseLong(timestampStr);
    long currentTime = System.currentTimeMillis();
    long diffMillis = currentTime - timestamp;

    long seconds = diffMillis / 1000;
    long minutes = seconds / 60;
    long hours = minutes / 60;
    long days = hours / 24;

    if (seconds < 60) {
        return "Vừa xong";
    } else if (minutes < 60) {
        return minutes + " phút trước";
    } else if (hours < 24) {
        return hours + " giờ trước";
    } else if (days < 7) {
        return days + " ngày trước";
    } else {
        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy",
        Locale.getDefault());
        return sdf.format(new Date(timestamp));
    }
}

} catch (NumberFormatException e) {
    e.printStackTrace();
    return "Thời gian không hợp lệ";
}
}

```

}

3.4.11 Thông báo

3.4.12 File FirebaseHelper (hỗ trợ thêm sửa xóa trên firebase realtime)

// 1 số đoạn code minh họa

KẾT LUẬN

1. Kết quả đạt được

Sau quá trình nghiên cứu và phát triển, đề tài đã đạt được các kết quả chính sau:

- **Xây dựng thành công ứng dụng Android Booking Care** sử dụng ngôn ngữ lập trình Java, giúp người dùng có thể đăng ký, đăng nhập và đặt lịch khám một cách tiện lợi.
- **Tích hợp đầy đủ các chức năng cần thiết**: quản lý người dùng, thông tin bác sĩ, đặt lịch, xem lịch sử khám, quản lý hồ sơ bệnh án.
- **Ứng dụng hoạt động ổn định**, dữ liệu được đồng bộ thời gian thực nhờ Firebase Realtime Database hoặc Firestore.
- **Bảo mật dữ liệu người dùng tốt** thông qua xác thực Firebase Authentication và kiểm soát quyền truy cập.
- **Sử dụng Firebase Cloud Messaging để gửi thông báo nhắc lịch khám**, giúp tăng tỷ lệ đến khám đúng hẹn của người dùng.
- **Triển khai thành công mô hình kiến trúc MVC**, tách biệt rõ ràng giữa giao diện, dữ liệu và xử lý logic, giúp dễ dàng bảo trì và nâng cấp sau này.
- **Ứng dụng đã được kiểm thử trên nhiều thiết bị Android phổ biến**, đảm bảo khả năng tương thích và trải nghiệm người dùng tốt.

2. Nhược điểm

Mặc dù ứng dụng đã hoàn thiện các chức năng cơ bản, nhưng đề tài vẫn còn tồn tại một số hạn chế như sau:

- **Chưa hỗ trợ đa nền tảng:** Ứng dụng hiện chỉ hoạt động trên thiết bị Android, chưa có phiên bản iOS hoặc web.
- **Chưa tích hợp thanh toán trực tuyến:** Người dùng chỉ mới đặt lịch khám, chưa thể thanh toán trực tiếp qua ví điện tử hoặc thẻ ngân hàng.
- **Chưa có hệ thống quản lý nâng cao cho bác sĩ và phòng khám** như thống kê doanh thu, số lượt khám, đánh giá hiệu quả khám chữa bệnh.
- **Tính năng tìm kiếm và lọc bác sĩ, chuyên khoa còn đơn giản**, chưa có gợi ý thông minh hoặc xếp hạng theo đánh giá.
- **Chưa tích hợp chatbot hỗ trợ tư vấn trước khám**, khiến trải nghiệm người dùng chưa thật sự toàn diện.
- **Giao diện còn đơn giản**, chủ yếu tập trung vào chức năng, chưa tối ưu hóa hoàn toàn về mặt thẩm mỹ và trải nghiệm người dùng (UX/UI).

3. Hướng phát triển

Để hoàn thiện và đáp ứng tốt hơn nhu cầu thực tiễn, ứng dụng Booking Care sẽ được mở rộng với các định hướng sau:

- **Phát triển phiên bản đa nền tảng:** Xây dựng thêm phiên bản cho hệ điều hành iOS và nền tảng web, giúp người dùng có nhiều lựa chọn khi sử dụng dịch vụ.
- **Tích hợp cổng thanh toán trực tuyến:** Cho phép người dùng thanh toán trước chi phí khám bệnh qua ví điện tử (Momo, ZaloPay...) hoặc thẻ ngân hàng, giúp quy trình khám chữa bệnh diễn ra nhanh chóng, minh bạch.
- **Tăng cường tương tác với người dùng:** Phát triển tính năng chatbot hỗ trợ tư vấn sức khỏe ban đầu hoặc hướng dẫn đặt lịch, giúp người dùng mới dễ làm quen và sử dụng app.
- **Cải thiện tính năng tìm kiếm và lọc thông tin:** Áp dụng trí tuệ nhân tạo (AI) để gợi ý bác sĩ phù hợp theo triệu chứng, lịch sử khám hoặc chuyên khoa.
- **Bổ sung hệ thống đánh giá, phản hồi sau khám:** Giúp người dùng đưa ra quyết định lựa chọn bác sĩ/phòng khám tốt hơn, đồng thời nâng cao chất lượng dịch vụ.
- **Xây dựng dashboard quản trị nâng cao cho cơ sở y tế:** Hỗ trợ quản lý lịch hẹn, thống kê lượt khám, hiệu suất bác sĩ, giúp tối ưu vận hành.

TÀI LIỆU THAM KHẢO

- [1] David Flanagan, JavaScript: The Definitive Guide, 7th Edition, O'Reilly Media, 2020.
- [2] Adam Freeman, “Pro jQuery”, Apress, 2018.
- [3] Benjamin Jakobus, “Mastering Bootstrap 5”, Packt Publishing, 2018.