**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**ĐỒ ÁN THỰC HÀNH**

**MÔN HỌC: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**HỆ THỐNG QUẢN LÝ HỒ SƠ BỆNH ÁN**

**Giảng viên hướng dẫn: TS. Nguyễn Thành Sơn**

**MÃ MÔN HỌC:** **WESE331479\_22\_2\_02**

**Sinh viên thực hiện:**

1. Bùi Quốc Tĩnh 20110737

*Tp. Hồ Chí Minh, ngày 23 tháng 11 năm 2022*

**MỤC LỤC**

[LỜI CẢM ƠN 1](#_heading=h.26in1rg)

[I. Lý do chọn đề tài 2](#_heading=h.lnxbz9)

[II. Đặc tả đề tài 2](#_heading=h.35nkun2)

[2.1. Chức năng quản lý Bệnh nhân 3](#_heading=h.1ksv4uv)

[2.2. Chức năng quản lý Nhân viên 3](#_heading=h.44sinio)

[2.3. Chức năng quản lý Dịch vụ 3](#_heading=h.2jxsxqh)

[2.4. Chức năng quản lý Thuốc 4](#_heading=h.z337ya)

[2.5. Chức năng quản lý Thanh toán 4](#_heading=h.3j2qqm3)

[III. Thiết kế CSDL 5](#_heading=h.1y810tw)

[3.1. Thiết kế CSDL mức quan niệm 5](#_heading=h.4i7ojhp)

[3.2. Thiết kế mức Logic 5](#_heading=h.1ci93xb)

[3.2.1. Mô hình quan hệ 5](#_heading=h.3whwml4)

[3.2.2. Mô tả các bảng 6](#_heading=h.2bn6wsx)

[3.2.2.1. Bảng People 6](#_heading=h.qsh70q)

[3.2.2.2. Bảng Patients 8](#_heading=h.3as4poj)

[3.2.2.3. Bảng Relatives 8](#_heading=h.1pxezwc)

[3.2.2.4. Bảng Employees 9](#_heading=h.49x2ik5)

[3.2.2.5. Bảng Services 9](#_heading=h.2p2csry)

[3.2.2.6. Bảng UsingService 10](#_heading=h.147n2zr)

[3.2.2.7. Bảng Departments 10](#_heading=h.3o7alnk)

[3.2.2.8. Bảng Receipts 11](#_heading=h.23ckvvd)

[3.2.2.9. Bảng Pay 12](#_heading=h.ihv636)

[3.2.2.10. Bảng Prescriptions 13](#_heading=h.32hioqz)

[3.2.2.11. Bảng CreatedPrescription 13](#_heading=h.1hmsyys)

[3.2.2.12. Bảng Medicines 14](#_heading=h.41mghml)

[3.2.2.13. Bảng Prescription\_Medicine 15](#_heading=h.2grqrue)

[3.2.2.14. Bảng MedicineGroups 16](#_heading=h.vx1227)

[3.2.2.15. Bảng Examination 17](#_heading=h.3fwokq0)

[3.2.2.16. Account 18](#_heading=h.1v1yuxt)

[3.2.2.17. Role 19](#_heading=h.4f1mdlm)

[3.2.3. Biểu đồ quan hệ giữa các bảng (Database diagram) 20](#_heading=h.2u6wntf)

[3.3. Các ràng buộc toàn vẹn 20](#_heading=h.3tbugp1)

[3.4. Cài đặt CSDL và các ràng buộc 24](#_heading=h.28h4qwu)

[3.4.1. Cài đặt bảng People 24](#_heading=h.nmf14n)

[3.4.2. Cài đặt bảng Patients 25](#_heading=h.37m2jsg)

[3.4.3. Cài đặt bảng Relatives 26](#_heading=h.1mrcu09)

[3.4.5. Cài đặt bảng Employees 26](#_heading=h.46r0co2)

[3.4.6. Cài đặt bảng Services 26](#_heading=h.2lwamvv)

[3.4.7. Cài đặt bảng UsingService 27](#_heading=h.111kx3o)

[3.4.8. Cài đặt bảng Departments 28](#_heading=h.3l18frh)

[3.4.9. Cài đặt cập nhật bảng Employees 28](#_heading=h.206ipza)

[3.4.10. Cài đặt bảng Receipts 28](#_heading=h.4k668n3)

[3.4.11. Cài đặt bảng Pay 29](#_heading=h.2zbgiuw)

[3.4.12. Cài đặt bảng Prescriptions 29](#_heading=h.1egqt2p)

[3.4.13. Cài đặt bảng CreatedPrescription 30](#_heading=h.3ygebqi)

[3.4.14. Cài đặt bảng Medicines 30](#_heading=h.2dlolyb)

[3.4.15. Cài đặt bảng Prescription\_Medicine 31](#_heading=h.sqyw64)

[3.4.16. Cài đặt bảng MedicineGroups 31](#_heading=h.3cqmetx)

[3.4.17. Cài đặt cập nhật bảng Medicine 32](#_heading=h.1rvwp1q)

[3.4.18. Cài đặt cập nhật bảng Examination 32](#_heading=h.4bvk7pj)

[3.4.19. Cài đặt cập nhật bảng Account 33](#_heading=h.2r0uhxc)

[IV. Thiết lập chức năng 34](#_heading=h.1664s55)

[4.1. Kết nối CSDL 34](#_heading=h.3q5sasy)

[4.2. Chức năng Doctor 34](#_heading=h.25b2l0r)

[4.2.1. Chức năng trong form “frmExamine” 34](#_heading=h.kgcv8k)

[4.2.1.1. Chức năng hiển thị thông tin bệnh nhân chờ 34](#_heading=h.34g0dwd)

[4.2.1.2. Chức năng đếm số lượng bệnh nhân chờ 35](#_heading=h.1jlao46)

[4.2.1.3. Hiển thị danh sách bệnh nhân chờ cấp thuốc 36](#_heading=h.43ky6rz)

[4.2.1.4. Chức năng lấy tên phòng khoa bằng Id của doctor 37](#_heading=h.2iq8gzs)

[4.2.1.5. Chức năng lấy họ và tên doctor bằng id doctor 38](#_heading=h.xvir7l)

[4.2.1.6. Chức năng thêm thông tin vào bảng Examination đồng thời tạo đơn thuốc rỗng 39](#_heading=h.3hv69ve)

[4.2.2. Chức năng trong form “frmPrescription” 43](#_heading=h.1x0gk37)

[4.2.2.1. Chức năng hiển thị thông tin thuốc được lấy 43](#_heading=h.4h042r0)

[4.2.2.3. Chức năng thêm thuốc vào đơn thuốc 44](#_heading=h.2w5ecyt)

[4.2.2.4. Chức năng cập nhật thuốc vào đơn thuốc 45](#_heading=h.1baon6m)

[4.2.2.5. Chức năng xóa thuốc khởi đơn thuốc 46](#_heading=h.3vac5uf)

[4.2.2.6. Cập nhật thông tin đơn thuốc rỗng đã tạo 48](#_heading=h.2afmg28)

[4.3. Chức năng Nhân viên 50](#_heading=h.pkwqa1)

[4.3.1. Chức năng trong form “TiepNhanBenhNhan” 50](#_heading=h.39kk8xu)

[4.3.1.1. Chức năng xem danh sách bệnh nhân 50](#_heading=h.1opuj5n)

[4.3.1.2 Chức năng thêm bệnh nhân với chờ khám 50](#_heading=h.48pi1tg)

[4.3.2. Chức năng trong form “ThongtinBenhNhan” 51](#_heading=h.2nusc19)

[4.3.2.1. Chức năng xem danh sách thân nhân của bệnh nhân 51](#_heading=h.1302m92)

[4.3.2.2. Chức năng cập nhật thông tin bệnh nhân 52](#_heading=h.3mzq4wv)

[4.3.2.3. Chức năng xem lịch sử khám của bệnh nhân 54](#_heading=h.2250f4o)

[4.3.2.3. Chức năng thêm bệnh nhân 58](#_heading=h.haapch)

[4.3.3. Chức năng trong form “ThemThanNhan” 59](#_heading=h.319y80a)

[4.3.3.1. Chức năng thêm thân nhân đã tồn tại trong cơ sở dữ liệu 59](#_heading=h.1gf8i83)

[4.3.3.2. Chức năng thêm thân nhân chưa tồn tại trong cơ sở dữ liệu 59](#_heading=h.40ew0vw)

[4.3.4. Chức năng trong form “ThongTinDonKham”: 61](#_heading=h.2fk6b3p)

[4.3.5. Chức năng trong form “SuDungDichVu” 62](#_heading=h.upglbi)

[4.3.5.1. Chức năng chọn sử dụng dịch vụ 62](#_heading=h.3ep43zb)

[4.3.5.2. Chức năng bỏ sử dụng dịch vụ 64](#_heading=h.1tuee74)

[4.3.5.3. Chức năng thay đổi sử dụng dịch vụ: 64](#_heading=h.4du1wux)

[4.3.6. Chức năng trong form “ThanhToanSuDungDichVu” 65](#_heading=h.2szc72q)

[4.3.7. Chức năng trong form “ThanhToanDonThuoc”: 69](#_heading=h.184mhaj)

[4.4. Chức năng Admin 74](#_heading=h.3s49zyc)

[4.4.1. Chức năng trong form “frmDepartment” 74](#_heading=h.279ka65)

[4.4.1.1. Hiển thị thông tin phòng ban 74](#_heading=h.meukdy)

[4.4.1.2. Thêm phòng ban 75](#_heading=h.36ei31r)

[4.4.1.3. Chỉnh sửa thông tin phòng ban 77](#_heading=h.1ljsd9k)

[4.4.1.4. Xóa thông tin phòng ban 79](#_heading=h.45jfvxd)

[4.4.1.5. Tìm kiếm tên theo ký tự tên có trong tên phòng ban 79](#_heading=h.2koq656)

[4.4.2. Chức năng trong form “frmMedicine” 80](#_heading=h.zu0gcz)

[4.4.2.1. Hiển thị thông tin thuốc 80](#_heading=h.3jtnz0s)

[4.4.2.2. Thêm thông tin thuốc 81](#_heading=h.1yyy98l)

[4.4.2.3. Cập nhật thông tin thuốc 83](#_heading=h.4iylrwe)

[4.4.2.4. Xóa thông tin thuốc khỏi người dùng 85](#_heading=h.2y3w247)

[4.4.1.5. Tìm kiếm tên theo ký tự tên có trong tên thuốc 86](#_heading=h.1d96cc0)

[4.4.3. Chức năng trong form “frmMedicineDel” 87](#_heading=h.3x8tuzt)

[4.4.3.1. Hiển thị thông tin thuốc đã xóa bởi người dùng 87](#_heading=h.2ce457m)

[4.4.3.2. Khôi phục thông tin thuốc với người dùng 88](#_heading=h.rjefff)

[4.4.4. Chức năng trong form “frmMedicineGroup” 90](#_heading=h.3bj1y38)

[4.4.4.1. Hiển thị thông tin nhóm thuốc 90](#_heading=h.1qoc8b1)

[4.4.4.2. Thêm thông tin nhóm thuốc 91](#_heading=h.4anzqyu)

[4.4.4.3. Cập nhật thông tin nhóm thuốc 92](#_heading=h.2pta16n)

[4.4.4.4. Xóa thông tin đơn thuốc 94](#_heading=h.14ykbeg)

[4.4.5. Chức năng trong form “frmService” 95](#_heading=h.3oy7u29)

[4.4.5.1. Hiển thị thông tin dịch vụ 95](#_heading=h.243i4a2)

[4.4.5.2. Thêm thông tin dịch vụ 96](#_heading=h.j8sehv)

[4.4.5.3. Cập nhật thông tin dịch vụ 97](#_heading=h.338fx5o)

[4.4.5.4. Xóa thông tin dịch vụ 99](#_heading=h.1idq7dh)

[4.4.5.5. Tìm kiếm tên theo ký tự tên có trong tên thuốc 100](#_heading=h.42ddq1a)

[4.4.6. Chức năng trong form FrmDoctors 101](#_heading=h.2hio093)

[4.4.6.1. Lấy thông tin danh sách bác sĩ 101](#_heading=h.wnyagw)

[4.4.6.2. Thêm thông tin bệnh nhân 102](#_heading=h.3gnlt4p)

[4.4.6.3. Cập nhật thông tin bệnh nhân 106](#_heading=h.1vsw3ci)

[4.4.7. Chức năng trong form “FrmTatCaNhanVien” 109](#_heading=h.4fsjm0b)

[4.4.7.1. Lấy thông tin nhân viên 109](#_heading=h.2uxtw84)

[4.4.7.2. Thêm thông tin nhân viên 110](#_heading=h.1a346fx)

[4.4.7.3. Cập nhật thông tin nhân viên 114](#_heading=h.3u2rp3q)

[4.4.8. Chức năng trong form “FrmPatients” 118](#_heading=h.2981zbj)

[4.4.8.1. Lấy thông tin bệnh nhân chờ 118](#_heading=h.odc9jc)

[4.4.8.2. Thêm thông tin bệnh nhân 119](#_heading=h.38czs75)

[4.4.7.3. Cập nhật thông tin bệnh nhân 123](#_heading=h.1nia2ey)

[V. Tạo Role, User và Phân quyền 127](#_heading=h.47hxl2r)

[5.1. Tạo Role : 127](#_heading=h.2mn7vak)

[5.2. Phân quyền cho các Role: 127](#_heading=h.11si5id)

[5.2.1. Admin: 127](#_heading=h.3ls5o66)

[5.2.2. Doctor: 127](#_heading=h.20xfydz)

[5.2.2.1. Phân quyền mức View: 127](#_heading=h.4kx3h1s)

[5.2.2.2. Phân quyền trên View: 128](#_heading=h.302dr9l)

[5.2.2.3. Phân quyền trên Procedure: 128](#_heading=h.1f7o1he)

[5.2.3. Employee: 129](#_heading=h.3z7bk57)

[5.2.3.1. Phân quyền mức View: 129](#_heading=h.2eclud0)

[5.2.3.2. Phân quyền trên View: 129](#_heading=h.thw4kt)

[5.2.3.3. Phân quyền trên Procedure: 130](#_heading=h.3dhjn8m)

[5.3. Tạo User, cấp quyền truy cập (SQL): 130](#_heading=h.1smtxgf)

[VI. Thiết kế giao diện: 132](#_heading=h.4cmhg48)

[6.1. Form Đăng nhập: 132](#_heading=h.2rrrqc1)

[6.2. Các form trong role Admin: 133](#_heading=h.3qwpj7n)

[6.2.1. Form trang chủ Admin (Admin\_Home): 133](#_heading=h.261ztfg)

[6.2.2. Form quản lý Accounts (FrmAccount): 134](#_heading=h.356xmb2)

[6.2.3. Form quản lý Departments (frmDepartment): 135](#_heading=h.44bvf6o)

[6.2.4. Form quản lý Doctor (FrmDoctors): 136](#_heading=h.ymfzma)

[6.2.5. Form quản lý Medicines (frmMedicine - frmMedicineDel): 136](#_heading=h.3im3ia3)

[6.2.6. Form quản lý Medicine Group (frmMedicineGroup): 139](#_heading=h.2wwbldi)

[6.2.7. Form quản lý Services (frmService): 140](#_heading=h.3w19e94)

[6.2.8. Form quản lý Employee (Không tính Doctors - FrmTatCaNhanVien): 141](#_heading=h.qbtyoq)

[6.3. Các form trong role Doctor: 141](#_heading=h.1pgrrkc)

[6.3.1. Form quản lý Examinations (frmExamination): 141](#_heading=h.49gfa85)

[6.3.2. Form quản lý Prescriptions (frmPrescription): 142](#_heading=h.13qzunr)

[6.4. Các form trong role Employee(Staff): 143](#_heading=h.22vxnjd)

[6.4.1. Form trang chủ role Employee (TrangChu): 143](#_heading=h.i17xr6)

[6.4.2. Form quản lý UsingService (Sudungdichvu): 143](#_heading=h.1h65qms)

[6.4.3. Form Pay Prescription (ThanhToanDonThuoc): 144](#_heading=h.2gb3jie)

[6.4.4. Form Pay usingService (ThanhToanSuDungDichVu): 144](#_heading=h.3fg1ce0)

[6.4.5. Form quản lý Relatives (ThemThanNhan): 145](#_heading=h.4ekz59m)

[6.4.6. Form quản lý Patients (ThongTinBenhNhan): 145](#_heading=h.18vjpp8)

[6.4.7. Form thông tin Examinations (ThongTinDonKham): 146](#_heading=h.280hiku)

[6.4.8. Form tiếp nhận Patients (TiepNhanBenhNhan): 147](#_heading=h.375fbgg)

[VII. Kết luận: 148](#_heading=h.46ad4c2)

[TÀI LIỆU THAM KHẢO 149](#_heading=h.2lfnejv)

**DANH MỤC HÌNH ẢNH**

[Hình 3.1.1. Lược đồ ERD 5](#_heading=h.2xcytpi)

[Hình 3.2.3.1. Biểu đồ quan hệ giữa các bảng (Database diagram) 20](#_heading=h.19c6y18)

[Hình 6.1.1. Form đăng nhập 132](#_heading=h.16x20ju)

[Hình 6.2.1.1. Form trang chủ Admin (Admin\_Home) 133](#_heading=h.l7a3n9)

[Hình 6.2.2.1. Form quản lý Accounts (FrmAccount) 134](#_heading=h.1kc7wiv)

[Hình 6.2.3.1. Form quản lý Departments (frmDepartment) 135](#_heading=h.2jh5peh)

[Hình 6.2.5.1. Form frmMedicine 138](#_heading=h.1xrdshw)

[Hình 6.2.5.1. Form frmMedicineDel 138](#_heading=h.4hr1b5p)

[Hình 6.2.6.1. Form quản lý Medicine Group (frmMedicineGroup) 140](#_heading=h.1c1lvlb)

[Hình 6.2.7.1. Form quản lý Services (frmService) 141](#_heading=h.2b6jogx)

[Hình 6.2.8.1. Form quản lý Employee (Không tính Doctors - FrmTatCaNhanVien) 141](#_heading=h.3abhhcj)

[Hình 6.3.1.1. Form quản lý Examinations (frmExamination) 142](#_heading=h.2olpkfy)

[Hình 6.3.2.1. Form quản lý Prescriptions (frmPrescription) 142](#_heading=h.3nqndbk)

[Hình 6.4.1.1. Form quản lý Prescriptions (frmPrescription) 143](#_heading=h.320vgez)

[Hình 6.4.2.1. Form quản lý UsingService (Sudungdichvu) 143](#_heading=h.415t9al)

[Hình 6.4.3.1. Form Pay Prescription (ThanhToanDonThuoc) 144](#_heading=h.vgdtq7)

[Hình 6.4.4.1. Form Pay usingService (ThanhToanSuDungDichVu) 144](#_heading=h.1ulbmlt)

[Hình 6.4.5.1. Form quản lý Relatives (ThemThanNhan) 145](#_heading=h.2tq9fhf)

[Hình 6.4.6.1. Form quản lý Patients (ThongTinBenhNhan) 146](#_heading=h.3sv78d1)

[Hình 6.4.7.1. Form quản lý Patients (ThongTinBenhNhan) 146](#_heading=h.n5rssn)

[Hình 6.4.8.1. Form tiếp nhận Patients (TiepNhanBenhNhan) 147](#_heading=h.1maplo9)

# 

# LỜI CẢM ƠN

Để hoàn thành được đồ án này, trước hết nhóm chúng em xin gửi lời cảm ơn chân thành đến: Ban giám hiệu Trường Đại học Sư phạm Kỹ thuật thành phố Hồ Chí Minh. Trường đã tạo điều kiện cho sinh viên nói chung và nhóm em nói riêng có cơ hội tiếp cận môn học Hệ Quản Trị Cơ Sở Dữ Liệu một cách thuận lợi với hệ thống cơ sở vật chất phong phú, đa dạng về tài liệu để nghiên cứu.

Tiếp theo chúng em xin cảm ơn đến thầy Nguyễn Thành Sơn, người đã truyền đạt cho chúng em đầy đủ những kiến thức bổ ích trong suốt quá quá trình giảng dạy môn học này. Và chính thầy đã trực tiếp giúp đỡ chúng em trong suốt quá trình thực hiện đồ án này. Sự giúp đỡ nhiệt huyết của thầy đã giúp nhóm em giảm bớt rất nhiều khó khăn và hoàn thành việc nghiên cứu đề tài tốt hơn. Nhóm em cũng xin cảm ơn đến các bạn, các anh chị đi trước đã truyền đạt những kinh nghiệm quý báu cho chúng em tránh được những sai sót trong suốt quá trình viết tiểu luận cũng như củng cố được kiến thức.

Nhờ vào những bài học thực tế, thú vị của môn Hệ Quản Trị Cơ Sở Dữ Liệu, nhóm em đã tiếp thu được rất nhiều kiến thức bổ ích. Điều này sẽ giúp ích rất nhiều cho nhóm em nói riêng cũng như các bạn sinh viên nói chung có thể áp dụng những vào các dự án của mình sau này một thành thạo và giúp cho việc phát triển các dự án có thể trở nên đỡ vất vở hơn.

Đồ án này được nhóm em hoàn thành trong xuyên suốt học kỳ. Dù đã cố gắng tiếp thu những kiến thức mà thầy truyền đạt, tuy nhiên nhóm em không sao tránh khỏi những thiếu sót, mong sao thầy và các bạn có thể góp ý, chỉ dẫn để đồ án được hoàn thiện hơn.

Lời cuối cùng, nhóm em xin chúc thầy có nhiều sức khỏe và hạnh phúc trong cuộc sống. Chúng em xin chân thành cảm ơn.

# I. Lý do chọn đề tài

Ngày nay, ngành công nghệ thông tin đang phát triển vô cùng mạnh mẽ. Nó đã chứng tỏ được sức mạnh của mình trong mọi lĩnh vực của cuộc sống. Ngành tin học nước ta đã và đang khẳng định vai trò vị trí lớn của mình trong mọi lĩnh vực kinh tế, xã hội. Rất nhiều cơ quan, công ty, trường học, bệnh viện... đã đưa tin học vào ứng dụng. Trong đó việc quản lý thông tin là một ứng dụng cần thiết và mang lại nhiều lợi ích. Nó giúp cho các công ty. Xí nghiệp, trường học, bệnh viện,…tiết kiệm được thời gian, tiền bạc, nhân công và có độ chính xác rất cao.

Công nghệ thông tin là phương tiện trợ giúp đắc lực và có hiệu quả cao trong công tác quản lý nền hành chính nói chung và quản lý ngành y tế nói riêng. Tại các bệnh viện, trạm y tế, hồ sơ bệnh án là tài liệu quan trọng phải được giữ gìn, bảo quản tốt theo đúng quy định của pháp luật về lưu trữ. Vậy nên việc áp dụng công nghệ thông tin trong công tác quản lý bệnh viện, xây dựng các phần mềm quản lý kho hồ sơ bệnh án của tất cả các khoa, hỗ trợ tối ưu trong việc lưu trữ các bệnh án cẩn thận và tốt hơn, việc tra cứu thông tin, vị trí hồ sơ nhanh và chính xác, tránh mất nhiều thời gian. Góp phần thúc đẩy bệnh viện phát triển toàn diện, từng bước đáp ứng yêu cầu về khám chữa bệnh và chăm sóc sức khỏe cho nhân dân.

Xuất phát từ nhận thức trên cùng với những kiến thức đã học trong môn học Hệ quản trị cơ sở dữ liệu, nhóm chúng em xin được tìm hiểu và thực hiện đề tài “Hệ thống quản lý Hồ sơ bệnh án”.

# II. Đặc tả đề tài

Trong một phòng khám, mỗi thông tin của một **People** đều được thêm vào hệ thống quản lý, **People** sẽ được phân thành 3 loại bao gồm : **Thân nhân**, **Bệnh nhân** và **Nhân viên** bằng role cả 3 loại People này đều được thừa kế tất cả thuộc tính của People bao gồm Mã people, Tên people, Địa chỉ, Số điện thoại, Ngày sinh, Giới tính, Ngày tạo thông tin, Ngày thông tin được sửa đổi,... và mỗi loại đều có thêm những thuộc tính riêng biệt. Riêng Nhân viên sẽ được phân loại lần 2 bằng thuộc tính position bao gồm Bác sĩở từng phòng **Khoa** và Xnhư Y tá, điều dưỡng,.. Mỗi Nhân viên sẽ được cấp một **Tài khoản** để truy cập vào phần mềm quản lý**.** Phòng khám sẽ có những **Dịch vụ** khám để Bệnh nhân sử dụng và những loại **Thuốc** thích hợp cho Bệnh nhân sau khi bệnh nhân đã khám xong.

## 2.1. Chức năng quản lý Bệnh nhân

Một **Bệnh nhân** có thể có **Thân nhân** đi cùng, **Thân nhân** sẽ có Mã thân nhân và quan hệ đối với **Bệnh nhân. Bệnh nhân** có Mã bệnh nhân để phân biệt, có bảo hiểm y tế để xét đến các phúc lợi được hưởng, công việc, và lý do **Bệnh nhân** vào viện là gì.

**Bệnh nhân** sau khi cung cấp những thông tin, thì sẽ đóng tiền **Hóa đơn**, sau đó sẽ được **Bác sĩ** khám, sử dụng những **Dịch vụ** và được cung cấp Toa thuốc thích hợp.

## 2.2. Chức năng quản lý Nhân viên

**X** tiến hành lấy thông tin của Bệnh nhân**,** hướng dẫn Bệnh nhân các bước hoàn tất thủ tục đóng tiền, chỉ dẫn Bệnh nhân đến **Khoa** phù hợp với lý do vào viện của Bệnh nhân để khám. Mỗi **Khoa** đều có Mã khoa để phân biệt và Tên khoa, một Khoa có thể có một hoặc nhiều Bác sĩ. Bác sĩ tiếp nhận **Bệnh nhân** sẽ Khám, lưu trữ những thông tin về Chiều cao, Cân nặng, Mạch đập, Triệu chứng, Nhịp tim, Nhiệt độ và Ngày khám,... mỗi lần Khám sẽ có Mã khám để phân biệt. Sau đó thực hiện chuẩn đoán bệnh tình, Bác sĩ sẽ cho Bệnh nhân tiến hành sử dụng những **Dịch vụ** của phòng khám để đưa ra kết quả chuẩn xác nhất và kê Toa thuốc thích hợp.

Mỗi **Nhân viên** đều có quyền thêm, tìm kiếm, xem, sửa thông tin của **Bệnh nhân**, **Nhân viên** có quyền và chức vụ của Admin sẽ có thêm quyền xóa thông tin.

Mỗi **Nhân viên** đều có quyền thêm, tìm kiếm, xem, sửa, xóa thông tin của **Thuốc, Dịch vụ** theo đúng thực tế.

Admin sẽ có quyền thêm, sửa, xóa **Tài khoản, Role** của **Nhân viên.** Mỗi **Tài khoản** sẽ cần có Id và mật khẩu để đăng nhập, cùng với đó là Mã phân quyền role để phân quyền quản lý phần mềm. Mỗi **Role** sẽ bao gồm Mã và tên **Role.**

## 2.3. Chức năng quản lý Dịch vụ

Phòng khám sẽ có danh sách những **Dịch vụ** như Xét nghiệm máu, Xét nghiệm nước tiểu,… để **Bệnh nhân** sử dụng sau khi được **Bác sĩ** chẩn đoán. Mỗi **Dịch vụ** sẽ có Mã dịch vụ, Tên dịch vụ và Giá tiền khi sử dụng dịch vụ đó. Bệnh nhân **Sử dụng dịch vụ** sẽ được lưu trữ thông tin về **Dịch vụ,** Số lượng đã sử dụng và Ngày sử dụng dịch vụ đó.

## 2.4. Chức năng quản lý Thuốc

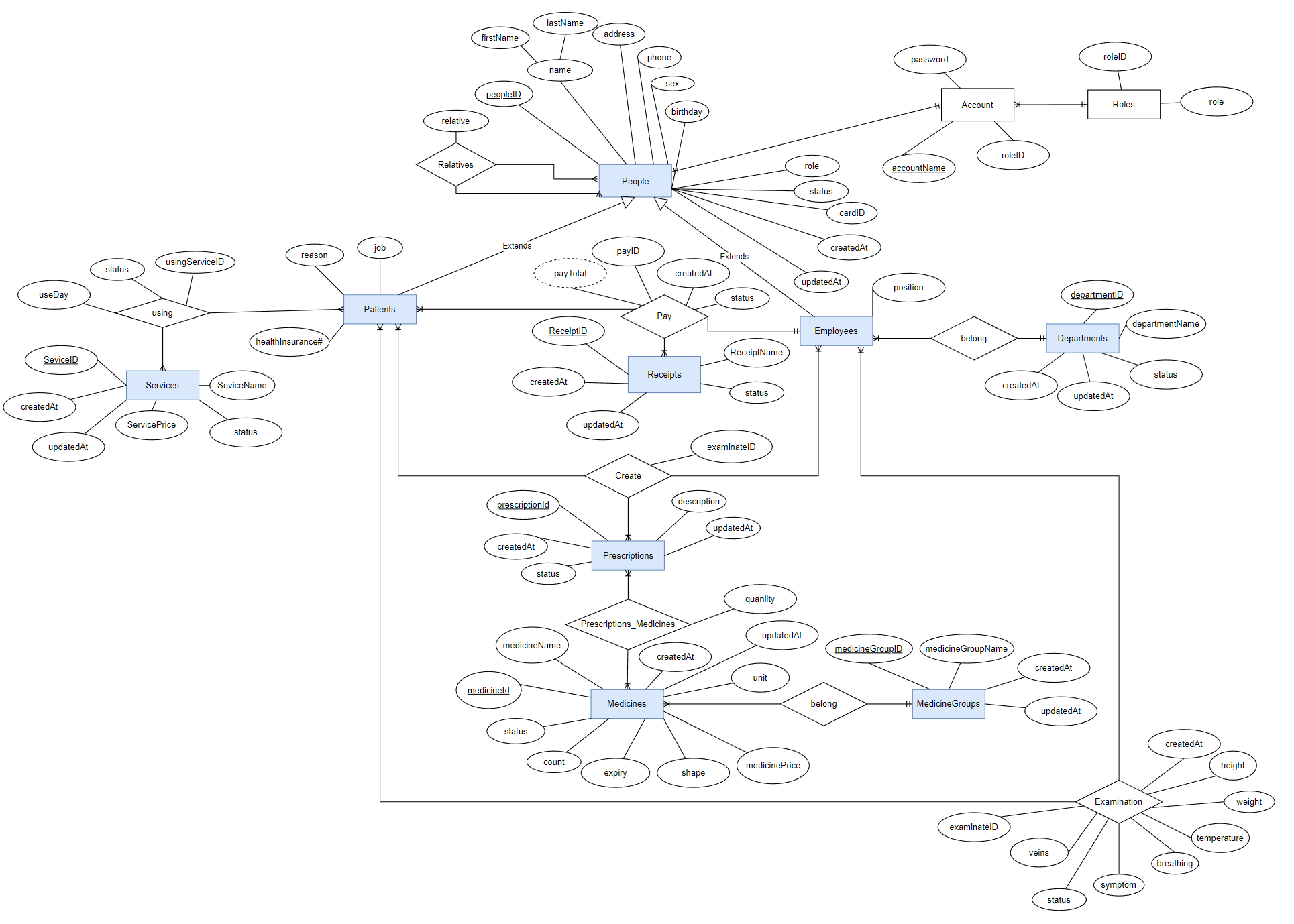
Sau khi đã thực hiện xong hết các bước khám bệnh và sử dụng Dịch vụ, thì **Bác sĩ** sẽ dựa vào đó để đưa ra kết quả cuối cùng cho **Bệnh nhân**, sau đó sẽ kê **Toa thuốc** cho bệnh nhân. Toa thuốcsẽ có Mã toa thuốc, Mô tả toa thuốc, Ngày kê toa và mỗi Toa thuốc có thể có nhiều loại **Thuốc**. Những loại thuốc trong toa này đều có trong kho **Thuốc** của phòng khám, mỗi một loại **Thuốc** đều có Mã thuốc, Tên thuốc, Số lượng, Hạn sử dụng, Đơn vị thuốc, Ngày nhập thuốc và Ngày chỉnh sửa (Chỉnh sửa giá cả,…).

## 2.5. Chức năng quản lý Thanh toán

Tổng số tiền mà **Bệnh nhân** đã sử dụng bao gồm tiền trong **Hóa đơn,** tiền **Dịch vụ** đã sử dụng và tiền **Thuốc**.

# III. Thiết kế CSDL

## 3.1. Thiết kế CSDL mức quan niệm

Hình 3.1.1. Lược đồ ERD

## 3.2. Thiết kế mức Logic

### 3.2.1. Mô hình quan hệ

**People**(peopleID, firstName, lastName, sex, birthDay, birthDay, address, phone, cardID, role, status, createdAt, updatedAt)

**Patients**(patientID, patientJob, healthInsurance, reason)

**Relatives**(relativeID, patientID)

**Employees**(employeeID, position, departmentID)

**Services**(serviceID, serviceName, servicePrice, status, createdAt, updatedAt)

**UsingService**(usingServiceID, patientID, serviceID, quantity, status, createdAt)

**Departments**(departmentID, departmentName, status, createdAt, updatedAt)

**Receipts**(receiptID, receiptName, status, createdAt, updatedAt)

**Pay**(payID, patientID, employeeID, receiptID, payTotal, status, createdAt)

**Prescriptions**(prescriptionID, descriptionPrescription, createdAt], updatedAt, status)

**CreatedPrescription**(\_id, patientID, employeeID, prescriptionID)

**Medicines**(medicineID, medicineName, unit, medicinePrice, status, expiry, count, shape, createdAt, updatedAt, medicineGroupName)

**Prescription\_Medicine**(prescriptionID, medicineName, quality)

**MedicineGroups**(medicineGroupID, medicineGroupName, createdAt, updatedAt)

**Examination**(examinateID, patientID, employeeID, height, weight, temperature, breathing, symptom, veins, bloodPressure, preliminaryDiagnosis, finalDiagnosis, treatmentDirection, status, createdAt)

**Account**(accountName, password, role)

**Role**(ID, role)

### 3.2.2. Mô tả các bảng

#### 3.2.2.1. Bảng People

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | peopleID | ID của người  + primary key(khóa chính)  + default: XX0000  + tự sinh  + maxLength 20 | string |
| 2 | firstName | Họ của Người  + not null  + maxLength 32 | string |
| 3 | lastName | Tên của người  + not null  + maxLength 32 | string |
| 4 | sex | Giới tính  + not null  + maxLength 1  + enum['M', 'F', 'O'] | char |
| 5 | birthDay | Ngày sinh  + not null | date |
| 6 | address | Địa chỉ  + not null  + maxLength 510 | string |
| 7 | phone | Số điện thoại  + not null  + maxLength 15 | string |
| 8 | cardID | Số CMND  + not null  + unique | string |
| 9 | role | Vai trò của người đó  + thân nhân = 0; bệnh nhân = 1; nhân viên = 2  + not null | integer |
| 10 | status | Trạng thái  + not null  + default: 1 | integer |
| 11 | createdAt | Thời gian khởi tạo People  + auto-generated | datetime |
| 12 | updatedAt | Thời gian cập nhật People  + auto-updated | datetime |

#### 3.2.2.2. Bảng Patients

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | patientID | ID của bệnh nhân  + primary key(khóa chính)  + default: XX0000  + tự sinh  + maxLength 20  + ref: People | string |
| 2 | patientJob | Công việc của bệnh nhân  + maxLength 255 | string |
| 3 | healthInsurance | Số bảo hiểm ý tế  + maxLength 20 | string |
| 4 | reason | Lý do vào viện  + maxLength 500 | string |

#### 3.2.2.3. Bảng Relatives

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | relativeID | ID của mối quan hệ  + primary key(khóa chính)  + maxLength 20  + ref: People | string |
| 2 | patientID | ID của bệnh nhân  + primary key(khóa chính)  + maxLength 20  + ref: People | string |

#### 3.2.2.4. Bảng Employees

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | employeeID | ID của nhân viên  + primary key(khóa chính)  + maxLength 20  + ref: People | string |
| 2 | position | Chức vụ của nhân viên  + maxLength 25 | string |
| 3 | departmentID | ID phòng ban  + maxLength 20  + ref: Departments | string |

#### 3.2.2.5. Bảng Services

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | serviceID | ID dịch vụ  + primary key(khóa chính)  + default: XX0000  + maxLength 20 | string |
| 2 | serviceName | Tên của dịch vụ  + maxLength 255  + not null  + unique | string |
| 3 | servicePrice | Đơn giá của dịch vụ | integer |
| 4 | status | Trạng thái  + not null  + default: 1 | integer |
| 5 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |
| 6 | updatedAt | Thời gian cập nhật  + auto-updated | datetime |

#### 3.2.2.6. Bảng UsingService

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | usingServiceID | ID của sử dụng dịch vụ  + primary key(khóa chính)  + maxLength 20  + Tự động tăng | integer |
| 2 | patientID | ID của bệnh nhân  + primary key(khóa chính)  + maxLength 20  + ref: Patients | string |
| 3 | serviceID | ID của dịch vụ  + primary key(khóa chính)  + maxLength 20  + ref: Services | string |
| 4 | useday | Ngày sử dụng dịch vụ | date |
| 5 | quantity | Số lượng các dịch vụ sử dụng | integer |
| 6 | status | Trạng thái  + not null  + default 1 | integer |

#### 3.2.2.7. Bảng Departments

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | departmentID | ID  + primary key(khóa chính)  + auto-generated (được tạo tự động)  + default : XXX0000  + maxLength 20 | string |
| 2 | departmentName | Tên phòng ban  + maxLength 255  + not null  + unique | string |
| 3 | status | Trạng thái  + default: 1  + not null | integer |
| 4 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |
| 5 | updatedAt | Thời gian cập nhật People  + auto-updated | datetime |

#### 3.2.2.8. Bảng Receipts

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | receiptID | ID  + Khóa chính  + Default 'XXX0000'  + Maxlength = 20 | string |
| 2 | receiptName | Tên hóa đơn  + Maxlength = 225 | string |
| 3 | status | Trạng thái hóa đơn  + Default = 1  + NOT NULL | interger |
| 4 | createdAt | Thời gian khởi tạo People  + auto-generated | datetime |
| 5 | updatedAt | Thời gian cập nhật People  + auto-updated | datetime |

#### 3.2.2.9. Bảng Pay

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | payID | ID thanh toán  + Khóa chính  + Tự động tăng | integer |
| 2 | patientID | ID bệnh nhân  + Khóa chính  + Maxlength = 20  + Ref: Patients | string |
| 3 | employeeID | ID nhân viên  + Khóa chính  + Maxlength = 20  + Ref: Employees | string |
| 4 | receiptID | ID hóa đơn  + Khóa chính  + Maxlength = 20  + Ref: Receipts | string |
| 5 | payTotal | Tổng số tiền trả | integer |
| 6 | status | Trạng thái hóa đơn  + Default = 1  + NOT NULL | integer |
| 7 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |

#### 3.2.2.10. Bảng Prescriptions

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | prescriptionId | ID của đơn thuốc  + Khóa chính  + NOT NULL  + Maxlength = 20  + Tự sinh  + DEFAULT 'XX0000' | string |
| 2 | descriptionPrescription | Mô tả đơn thuốc | string |
| 3 | status | Trạng thái của đơn thuốc  + Default = 1  + NOT NULL | integer |
| 4 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |
| 5 | updatedAt | Thời gian cập nhật  + auto-updated | datetime |

#### 3.2.2.11. Bảng CreatedPrescription

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | \_id | ID cấp thuốc  + Khóa chính  + Tự động tăng | integer |
| 2 | patientID | ID bệnh nhân  + Khóa chính  + Maxlength = 20  + Ref: Patients | string |
| 3 | employeeID | ID nhân viên  + Khóa chính  + Maxlength = 20  + Ref: Employees | string |
| 4 | prescriptionID | ID đơn thuốc  + Khóa chính  + Maxlength = 20  + Ref: Prescriptions | string |

#### 3.2.2.12. Bảng Medicines

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | medicineID | ID thuốc  + Khóa chính  + Maxlength = 20  + Tự sinh  + DEFAULT 'XX0000' | string |
| 2 | medicineName | Tên thuốc  + NOT NULL  + UNIQUE  + Maxlength = 225 | string |
| 3 | unit | Đơn vị đo của thuốc  + Maxlength = 225 | string |
| 4 | medicinePrice | Giá của thuốc trên từng đơn vị đo  + min = 1 | integer |
| 5 | status | Trạng thái của thuốc  + Default = 1  + NOT NULL | integer |
| 6 | expiry | Số ngày sử dụng thuốc (tức hạn sử dụng)  + NOT NULL | integer |
| 7 | count | Số lượng thuốc còn trong kho  + NOT NULL  + min = 0 | integer |
| 8 | shape | Hình dạng thuốc  + NOT NULL  + Maxlength = 100 | string |
| 9 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |
| 10 | updatedAt | Thời gian cập nhật  + auto-updated | datetime |
| 11 | medicineGroupName | Tên nhóm thuốc  + Ref: MedicineGroups  + Maxlength = 225 | string |

#### 3.2.2.13. Bảng Prescription\_Medicine

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | prescriptionID | ID của đơn thuốc  + Khóa chính  + Maxlength = 20  + Ref: Prescriptions | string |
| 2 | medicineName | Tên thuốc  + Khóa chính  + Maxlength = 225  + Ref: Medicines | string |
| 3 | quality | Số lượng thuốc  + NOT NULL | integer |

#### 3.2.2.14. Bảng MedicineGroups

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | medicineGroupID | ID của nhóm thuốc  + Khóa chính  + Maxlength = 20  + Tự động  +DEFAULT 'XXX0000' | string |
| 2 | medicineGroupName | Tên nhóm thuốc  + Maxlength = 225  + NOT NLL  + UNIQUE | string |
| 3 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |
| 4 | updatedAt | Thời gian cập nhật  + auto-updated | datetime |

#### 3.2.2.15. Bảng Examination

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | examinateID | ID giấy khám  + Khóa chính  + Maxlength = 20  + Tự sinh  + DEFAULT 'XX0000' | string |
| 2 | patientID | ID bệnh nhân  + Khóa chính  + Maxlength = 20  + Ref: Patients | string |
| 3 | employeeID | ID bác sĩ khám  + Khóa chính  + Maxlength = 20  + Ref: Employees | string |
| 4 | height | Chiều cao người khám | float |
| 5 | weight | Cân nặng người khám | float |
| 6 | temperature | Thân nhiệt người khám | float |
| 7 | breathing | Mạch người khám | integer |
| 8 | symptom | Triệu chứng của người khám  + Maxlength = 510 | string |
| 9 | veins | Mạch người khám | integer |
| 10 | bloodPressure | Huyết áp người khám | integer |
| 11 | preliminaryDiagnosis | Chẩn đoán sơ bộ  + Maxlength = 510 | string |
| 12 | finalDiagnosis | Chẩn đoán sau cùng  + Maxlength = 510 | string |
| 13 | treatmentDirection | Hướng điều trị  + Maxlength = 510 | string |
| 14 | status | Trạng thái của tờ khám  + Default = 1  + NOT NULL | integer |
| 15 | createdAt | Thời gian khởi tạo  + auto-generated | datetime |
|  |  |  |  |

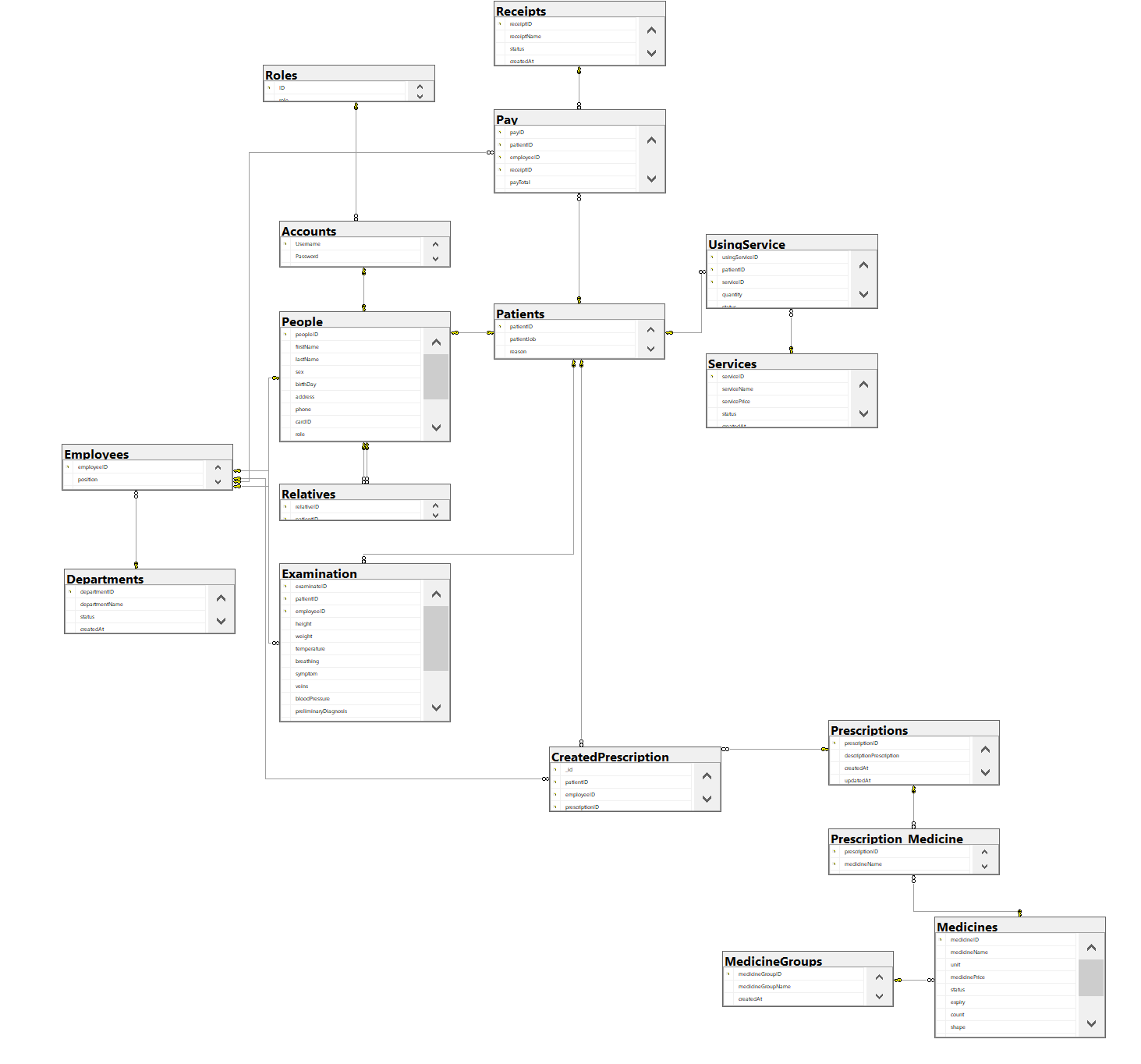
#### 3.2.2.16. Account

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | accountName | Tên tài khoản  + primary key(khóa chính)  + maxLength 20  + Ref: People | string |
| 2 | password | Mật khẩu  + maxLength 20  + not null | string |
| 3 | roleID | Vai trò của người đó  + eum[admin, doctor, employee]  + maxLength 20  + not null  + Ref: Role | string |

#### 3.2.2.17. Role

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên field** | **Mô tả** | **Kiểu dữ liệu** |
| 1 | ID | ID role  + primary key(khóa chính)  + Tăng tự động | int |
| 2 | role | Mật khẩu  + maxLength 50  + not null  + enum[admin, doctor, emploee] | string |

### 3.2.3. Biểu đồ quan hệ giữa các bảng (Database diagram)



Hình 3.2.3.1. Biểu đồ quan hệ giữa các bảng (Database diagram)

## 3.3. Các ràng buộc toàn vẹn

|  |  |
| --- | --- |
| **Quan hệ** | **Các ràng buộc** |
| People | - Ràng buộc NOT NULL: firstName, firstName, sex, birthDay, address, phone, role, status  - Ràng buộc UNIQUE: cardID  - Ràng buộc DEFAULT: peopleID = 'XX0000', status = 1  - Ràng buộc CHECK: sex = 'M' OR sex = 'F' OR sex = 'O', role BETWEEN 0 AND 2, birthDay < GETDATE(), phone NOT LIKE '%[^0-9]%'  - Ràng buộc PRIMARY KEY: peopleID |
| Patients | - Ràng buộc DEFAULT: patientID = 'XX0000'  - Ràng buộc PRIMARY KEY: patientID  - Ràng buộc POREIGN KEY: patientID Ref People |
| Relatives | - Ràng buộc PRIMARY KEY: relativeID, patientID  - Ràng buộc POREIGN KEY: relativeID Ref People, patientID Ref People |
| Employees | - Ràng buộc CHECK: position = N'Bác sĩ' OR position = N'Nhân viên'  - Ràng buộc PRIMARY KEY: employeeID  - Ràng buộc POREIGN KEY: employeeID Ref People, departmentID Ref Departments |
| Services | - Ràng buộc NOT NULL: serviceName, status  - Ràng buộc DEFAULT: serviceID = 'XX0000', status = 1  - Ràng buộc CHECK: servicePrice > 0  - Ràng buộc PRIMARY KEY: serviceID |
| UsingService | - Ràng buộc NOT NULL: usingServiceID, status  - Ràng buộc UNIQUE: usingServiceID  - Ràng buộc DEFAULT: status = 1  - Ràng buộc CHECK: quantity > 0  - Ràng buộc PRIMARY KEY: usingServiceID, patientID, serviceID  - Ràng buộc POREIGN KEY: patientID Ref Patients, serviceID Ref Services |
| Departments | - Ràng buộc NOT NULL: departmentName, status  - Ràng buộc UNIQUE: departmentName, status  - Ràng buộc DEFAULT: departmentID = 'XXX0000', status = 1  - Ràng buộc PRIMARY KEY: departmentID |
| Receipts | - Ràng buộc NOT NULL: status  - Ràng buộc DEFAULT: receiptID = 'XXX0000', status = 1  - Ràng buộc PRIMARY KEY: receiptID |
| Pay | - Ràng buộc NOT NULL: payID, patientID, employeeID, receiptID, status  - Ràng buộc UNIQUE: payID  - Ràng buộc DEFAULT: status = 1  - Ràng buộc PRIMARY KEY: payID, patientID, employeeID, receiptID  - Ràng buộc POREIGN KEY: patientID Ref Patients, employeeID Ref Employees, receiptID Ref Receipts |
| Prescriptions | - Ràng buộc NOT NULL: status  - Ràng buộc DEFAULT: prescriptionID = 'XX0000', status = 1  - Ràng buộc PRIMARY KEY: prescriptionID |
| CreatedPrescription | - Ràng buộc NOT NULL: \_id, patientID, employeeID, prescriptionID  - Ràng buộc UNIQUE: \_id  - Ràng buộc DEFAULT: status = 1  - Ràng buộc PRIMARY KEY: \_id, patientID, employeeID, prescriptionID  - Ràng buộc POREIGN KEY: patientID Ref Patients, employeeID Ref Employees, prescriptionID Ref Prescriptions |
| Medicines | - Ràng buộc NOT NULL: medicineName, status, expiry, count, shape  - Ràng buộc UNIQUE: medicineName  - Ràng buộc DEFAULT: medicineID = 'XX0000', status = 1  - Ràng buộc CHECK: count >= 0, medicinePrice > 0  - Ràng buộc PRIMARY KEY: medicineID  - Ràng buộc POREIGN KEY: medicineGroupName Ref medicineGroupName |
| Prescription\_Medicine | - Ràng buộc NOT NULL: prescriptionID, medicineName, quality  - Ràng buộc PRIMARY KEY: prescriptionID, medicineName  - Ràng buộc POREIGN KEY: prescriptionID Ref Prescriptions, medicineName Ref Medicines |
| MedicineGroups | - Ràng buộc NOT NULL: medicineGroupName  - Ràng buộc UNIQUE: medicineGroupName  - Ràng buộc DEFAULT: medicineGroupID = 'XXX0000'  - Ràng buộc PRIMARY KEY: medicineGroupID |
| Examination | - Ràng buộc NOT NULL: status  - Ràng buộc UNIQUE: examinateID  - Ràng buộc DEFAULT: examinateID = 'XX0000', status = 1  - Ràng buộc CHECK: height > 0, weight > 0, temperature > 0, breathing > 0, veins > 0  - Ràng buộc PRIMARY KEY: patientID, employeeID, examinateID  - Ràng buộc POREIGN KEY: patientID Ref Patients, employeeID Ref Employees |
| Account | - Ràng buộc NOT NULL: password, roleID  - Ràng buộc CHECK: role = 'admin' OR [role] = 'doctor' OR role = 'employee'  - Ràng buộc PRIMARY KEY: Username  - Ràng buộc POREIGN KEY: Username Ref People, roleID Ref Role |
| Role | - Ràng buộc NOT NULL: role  - Ràng buộc CHECK: role = 'admin' OR [role] = 'doctor' OR role = 'employee'  - Ràng buộc PRIMARY KEY: ID |

## 3.4. Cài đặt CSDL và các ràng buộc

### 3.4.1. Cài đặt bảng People

|  |
| --- |
| CREATE TABLE [People]  (  [peopleID] VARCHAR(20) DEFAULT 'XX0000',  [firstName] NVARCHAR(32) NOT NULL,  [lastName] NVARCHAR(32) NOT NULL,  [sex] CHAR(1) NOT NULL,  [birthDay] DATE NOT NULL,  [address] NVARCHAR(510) NOT NULL,  [phone] VARCHAR(15) NOT NULL,  [cardID] VARCHAR(15) NULL UNIQUE,  [role] TINYINT NOT NULL,  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  [updatedAt] DATETIME,  CONSTRAINT [peopleKey]  PRIMARY KEY ([peopleID]),  CONSTRAINT [sexCheck] CHECK ([sex] = 'M'  OR [sex] = 'F'  OR [sex] = 'O'  ),  -- thân nhân = 0; bệnh nhân = 1; nhân viên = 2  CONSTRAINT [role] CHECK ([role]  BETWEEN 0 AND 2  ),  CONSTRAINT [birthdayCheck] CHECK ([birthDay] < GETDATE()),  CONSTRAINT [chk\_phone] CHECK ([phone] NOT LIKE '%[^0-9]%'),  );  GO |

### 3.4.2. Cài đặt bảng Patients

|  |
| --- |
| CREATE TABLE [Patients]  (  [patientID] VARCHAR(20) DEFAULT 'XX0000', --Khóa chính tương ứng lớp cha  [patientJob] NVARCHAR(255),  [healthInsurance#] VARCHAR(20),  [reason] NVARCHAR(500),  CONSTRAINT [patientKey]  PRIMARY KEY ([patientID]),  CONSTRAINT [people\_patient]  FOREIGN KEY ([patientID])  REFERENCES [dbo].[People] ([peopleID]) ON UPDATE CASCADE  );  GO |

### 3.4.3. Cài đặt bảng Relatives

|  |
| --- |
| CREATE TABLE [Relatives]  (  [relativeID] VARCHAR(20),  [patientID] VARCHAR(20),  CONSTRAINT [relativeKey]  PRIMARY KEY (  [relativeID],  [patientID]  ),  CONSTRAINT [twoPeople] CHECK ([patientID] <> [relativeID]),  CONSTRAINT [existPerson1]  FOREIGN KEY ([patientID])  REFERENCES [dbo].[People] ([peopleID]),  CONSTRAINT [existPerson2]  FOREIGN KEY ([relativeID])  REFERENCES [dbo].[People] ([peopleID])  );  GO |

### 3.4.5. Cài đặt bảng Employees

|  |
| --- |
| CREATE TABLE [Employees]  (  [employeeID] VARCHAR(20), --Khóa chính tương ứng lớp cha  [position] NVARCHAR(25),  CONSTRAINT [employeeKey]  PRIMARY KEY ([employeeID]),  CONSTRAINT positionCheck CHECK (position = N'Bác sĩ' OR position = N'Nhân viên'),  CONSTRAINT [people\_employee]  FOREIGN KEY ([employeeID])  REFERENCES [dbo].[People] ([peopleID]) ON UPDATE CASCADE  );  GO |

### 3.4.6. Cài đặt bảng Services

|  |
| --- |
| CREATE TABLE [Services]  (  [serviceID] VARCHAR(20) DEFAULT 'XX0000',  [serviceName] NVARCHAR(255) NOT NULL UNIQUE,  [servicePrice] INT,  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  [updatedAt] DATETIME,  CONSTRAINT [sPrice] CHECK ([servicePrice] > 0),  CONSTRAINT [serviceKey]  PRIMARY KEY ([serviceID])  );  GO |

### 3.4.7. Cài đặt bảng UsingService

|  |
| --- |
| CREATE TABLE [UsingService]  (  [usingServiceID] INT IDENTITY(1,1) NOT NULL UNIQUE,  [patientID] VARCHAR(20),  [serviceID] VARCHAR(20),  [quantity] TINYINT,  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  CONSTRAINT [usingPKEY]  PRIMARY KEY (  [usingServiceID],  [patientID],  [serviceID]  ),  CONSTRAINT [pUsingFKEY]  FOREIGN KEY ([patientID])  REFERENCES [dbo].[Patients] ([patientID]) ON UPDATE CASCADE,  CONSTRAINT [sUsingFKEY]  FOREIGN KEY ([serviceID])  REFERENCES [dbo].[Services] ([serviceID]) ON UPDATE CASCADE,  CONSTRAINT [using#] CHECK ([quantity] > 0)  );  GO |

### 3.4.8. Cài đặt bảng Departments

|  |
| --- |
| CREATE TABLE [Departments]  (  [departmentID] VARCHAR(20) DEFAULT 'XXX0000',  [departmentName] NVARCHAR(255) NOT NULL UNIQUE,  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  [updatedAt] DATETIME,  CONSTRAINT [departmentKey]  PRIMARY KEY ([departmentID])  );  GO |

### 3.4.9. Cài đặt cập nhật bảng Employees

|  |
| --- |
| -- relationship belong  ALTER TABLE [dbo].[Employees] ADD [departmentID] VARCHAR(20) NULL;  GO  ALTER TABLE [dbo].[Employees]  ADD CONSTRAINT [belongDepartment]  FOREIGN KEY ([departmentID])  REFERENCES [dbo].[Departments] ([departmentID]) ON DELETE SET NULL ON UPDATE CASCADE;  GO |

### 3.4.10. Cài đặt bảng Receipts

|  |
| --- |
| CREATE TABLE [Receipts]  (  [receiptID] VARCHAR(20) DEFAULT 'XXX0000',  [receiptName] NVARCHAR(255),  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  [updatedAt] DATETIME,  CONSTRAINT [receiptKey]  PRIMARY KEY ([receiptID])  );  GO |

### 3.4.11. Cài đặt bảng Pay

|  |
| --- |
| CREATE TABLE [Pay]  (  [payID] INT IDENTITY(1,1) NOT NULL UNIQUE,  [patientID] VARCHAR(20) NOT NULL,  [employeeID] VARCHAR(20) NOT NULL,  [receiptID] VARCHAR(20) NOT NULL,  [payTotal] INT,  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  CONSTRAINT [payKey]  PRIMARY KEY (  [payID],  [patientID],  [employeeID],  [receiptID]  ),  CONSTRAINT [patientPay]  FOREIGN KEY ([patientID])  REFERENCES [dbo].[Patients] ([patientID]),  CONSTRAINT [employeePay]  FOREIGN KEY ([employeeID])  REFERENCES [dbo].[Employees] ([employeeID]),  CONSTRAINT [receiptPay]  FOREIGN KEY ([receiptID])  REFERENCES [dbo].[Receipts] ([receiptID])  );  GO |

### 3.4.12. Cài đặt bảng Prescriptions

|  |
| --- |
| CREATE TABLE [Prescriptions]  (  [prescriptionID] VARCHAR(20) DEFAULT 'XX0000',  [descriptionPrescription] NVARCHAR(500),  [createdAt] DATETIME,  [updatedAt] DATETIME,  [status] TINYINT NOT NULL DEFAULT 1,  CONSTRAINT [prescriptionKey]  PRIMARY KEY ([prescriptionId])  );  GO |

### 3.4.13. Cài đặt bảng CreatedPrescription

|  |
| --- |
| CREATE TABLE [CreatedPrescription]  (  [\_id] INT IDENTITY(1, 1) NOT NULL UNIQUE,  [patientID] VARCHAR(20) NOT NULL,  [employeeID] VARCHAR(20) NOT NULL,  [prescriptionID] VARCHAR(20) NOT NULL,  CONSTRAINT [CreatedPrescriptionKey]  PRIMARY KEY([\_id],  [patientID],  [employeeID],  [prescriptionID]),  CONSTRAINT [CreatedPrescription\_Patient\_FK] FOREIGN KEY (patientID) REFERENCES Patients (patientID),  CONSTRAINT [CreatedPrescription\_Employee\_FK] FOREIGN KEY (employeeID) REFERENCES Employees (employeeID),  CONSTRAINT [CreatedPrescription\_Prescription\_FK] FOREIGN KEY (prescriptionID) REFERENCES Prescriptions (prescriptionID)  );  GO |

### 3.4.14. Cài đặt bảng Medicines

|  |
| --- |
| CREATE TABLE [Medicines]  (  [medicineID] VARCHAR(20) DEFAULT 'XX0000',  [medicineName] NVARCHAR(255) NOT NULL UNIQUE,  [unit] NVARCHAR(255),  [medicinePrice] INT,  [status] TINYINT NOT NULL DEFAULT 1,  expiry INT NOT NULL, --số ngày sử dụng  count INT NOT NULL CHECK(count >= 0), -- số tồn trong kho  shape NVARCHAR(100) NOT NULL,  [createdAt] DATETIME,  [updatedAt] DATETIME,  CONSTRAINT [dPrice] CHECK ([medicinePrice] > 0),  CONSTRAINT [drugKey]  PRIMARY KEY ([medicineID])  );  GO |

### 3.4.15. Cài đặt bảng Prescription\_Medicine

|  |
| --- |
| CREATE TABLE Prescription\_Medicine  (  [prescriptionID] VARCHAR(20) NOT NULL,  [medicineName] NVARCHAR(255) NOT NULL,  [quanlity] INT NOT NULL,  CONSTRAINT [Prescription\_Medicine\_Key]  PRIMARY KEY(  [prescriptionID],  [medicineName]),  CONSTRAINT [Prescription\_Medicine\_FK] FOREIGN KEY (prescriptionID) REFERENCES Prescriptions(prescriptionID),  CONSTRAINT [Medicine\_Prescription\_FK] FOREIGN KEY (medicineName) REFERENCES Medicines(medicineName),  );  GO |

### 3.4.16. Cài đặt bảng MedicineGroups

|  |
| --- |
| CREATE TABLE MedicineGroups  (  [medicineGroupID] VARCHAR(20) DEFAULT 'XXX0000',  [medicineGroupName] NVARCHAR(255) NOT NULL UNIQUE,  [createdAt] DATETIME,  [updatedAt] DATETIME,  CONSTRAINT [medicineGroupKey]  PRIMARY KEY ([medicineGroupID])  );  GO |

### 3.4.17. Cài đặt cập nhật bảng Medicine

|  |
| --- |
| ALTER TABLE [Medicines] ADD [medicineGroupName] NVARCHAR(255);  GO  ALTER TABLE [Medicines] ADD CONSTRAINT [Medicine\_MedicineGroup\_FK] FOREIGN KEY (medicineGroupName) REFERENCES MedicineGroups(medicineGroupName)  ON DELETE SET NULL ON UPDATE CASCADE;  GO |

### 3.4.18. Cài đặt cập nhật bảng Examination

|  |
| --- |
| CREATE TABLE [Examination]  (  [examinateID] VARCHAR(20) UNIQUE DEFAULT 'XX0000',  [patientID] VARCHAR(20),  [employeeID] VARCHAR(20),  [height] FLOAT,  [weight] FLOAT,  [temperature] FLOAT,  [breathing] INT,  [symptom] NVARCHAR(510),  [veins] INT,  [bloodPressure] INT,  [preliminaryDiagnosis] NVARCHAR(510),  [finalDiagnosis] NVARCHAR(510),  [treatmentDirection] NVARCHAR(510),  [status] TINYINT NOT NULL DEFAULT 1,  [createdAt] DATETIME,  CONSTRAINT [examKey]  PRIMARY KEY (  [patientID],  [employeeID] ,  [examinateID]  ),  CONSTRAINT [patientExam]  FOREIGN KEY ([patientID])  REFERENCES [dbo].[Patients] ([patientID]),  CONSTRAINT [employeeExam]  FOREIGN KEY ([employeeID])  REFERENCES [dbo].[Employees] ([employeeID]),  CONSTRAINT [examID]  UNIQUE ([examinateID]),  CONSTRAINT [gtZero] CHECK ([height] > 0  AND [weight] > 0  AND [temperature] > 0  AND [breathing] > 0  AND [veins] > 0  )  );  GO |

### 3.4.19. Cài đặt cập nhật bảng Account

|  |
| --- |
| CREATE TABLE [Account]  (  [accountName] VARCHAR(20) primary key,  [password] VARCHAR(20) NOT NULL,  [status] TINYINT NOT NULL default 1,  [role] VARCHAR(20) NOT NULL,  CONSTRAINT [roleCheck] CHECK ([role] = 'admin'  OR [role] = 'doctor'  OR [role] = 'employee'  ),  );  GO |

# IV. Thiết lập chức năng

## 4.1. Kết nối CSDL

|  |
| --- |
| <connectionStrings>  <add *name*="QLHSBAEntities" *connectionString*="metadata=res://\*/Model.Model.csdl|res://\*/Model.Model.ssdl|res://\*/Model.Model.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=DESKTOP-CEL3U5G\SQLEXPRESS;initial catalog=QLHSBA;user id=sa;password=123456;MultipleActiveResultSets=True;App=EntityFramework&quot;" *providerName*="System.Data.EntityClient" />  </connectionStrings> |

*(Chuỗi kết nối trong file config)*

## 4.2. Chức năng Doctor

### 4.2.1. Chức năng trong form “frmExamine”

#### 4.2.1.1. Chức năng hiển thị thông tin bệnh nhân chờ

- View “View\_Patients\_Wait”:

|  |
| --- |
| CREATE VIEW [View\_Patients\_Wait]  AS  SELECT [P].[peopleID] as [patientID]  , [P].[firstName] as [patientFirstName]  , [P].[lastName] as [patientLastname]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [P2].[patientJob]  , [P2].[healthInsurance]  , [P2].[reason]  , [P].[createdAt]  , [P].[updatedAt]  FROM ([dbo].[People] as [P]  INNER JOIN [dbo].[Patients] as [P2]  ON ([P].[peopleID] = [P2].[patientID]))  INNER JOIN [dbo].[Examination] as [EX]  ON ([P2].[patientID] = [EX].[patientID])  WHERE [EX].[status] = 1;  GO |

- Stored procedure “FindAllPatientWait”:

|  |
| --- |
| --Begin: PROCEDURE FindAllPatientWait  CREATE PROCEDURE FindAllPatientWait  AS  BEGIN  SELECT \* FROM View\_Patients\_Wait  END  GO  --END: PROCEDURE FindAllPatientWait |

#### 4.2.1.2. Chức năng đếm số lượng bệnh nhân chờ

- View “View\_Patients\_Wait”:

|  |
| --- |
| CREATE VIEW [View\_Patients\_Wait]  AS  SELECT [P].[peopleID] as [patientID]  , [P].[firstName] as [patientFirstName]  , [P].[lastName] as [patientLastname]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [P2].[patientJob]  , [P2].[healthInsurance]  , [P2].[reason]  , [P].[createdAt]  , [P].[updatedAt]  FROM ([dbo].[People] as [P]  INNER JOIN [dbo].[Patients] as [P2]  ON ([P].[peopleID] = [P2].[patientID]))  INNER JOIN [dbo].[Examination] as [EX]  ON ([P2].[patientID] = [EX].[patientID])  WHERE [EX].[status] = 1;  GO |

- Stored procedure “spCountPatientsWait”:

|  |
| --- |
| CREATE PROCEDURE spCountPatientsWait  AS  BEGIN  SELECT COUNT(\*) FROM View\_Patients\_Wait;  END  GO |

#### 4.2.1.3. Hiển thị danh sách bệnh nhân chờ cấp thuốc

- View View\_Wait\_DrugSupply:

|  |
| --- |
| CREATE VIEW [View\_Wait\_DrugSupply]  AS  SELECT CPre.prescriptionID  , CPre.examinateID  , [P].[peopleID] as [patientID]  , [P].[firstName] as [patientFirstName]  , [P].[lastName] as [patientLastname]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [P2].[patientJob]  , [P2].[healthInsurance]  , [P2].[reason]  , [P].[createdAt]  , [P].[updatedAt]  FROM (([dbo].[People] as [P]  INNER JOIN [dbo].[Patients] as [P2]  ON ([P].[peopleID] = [P2].[patientID]))  INNER JOIN (dbo.CreatedPrescription as [CPre] INNER JOIN dbo.Prescriptions as [Pre] ON (CPre.prescriptionID = Pre.prescriptionID))  ON (P2.patientID = CPre.patientID))  WHERE Pre.[status] = 1;  GO |

- Stored procedure “spWaitDrugSupply”:

|  |
| --- |
| CREATE PROCEDURE spWaitDrugSupply  AS  BEGIN  SELECT \* FROM dbo.[View\_Wait\_DrugSupply]  END  GO |

#### 4.2.1.4. Chức năng lấy tên phòng khoa bằng Id của doctor

- View “View\_Doctor”:

|  |
| --- |
| CREATE VIEW [View\_Doctor]  AS  SELECT [P].[peopleID] as [doctorID]  , [P].[firstName] as [doctorFirstname]  , [P].[lastName] as [doctorLastName]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [D].[departmentName]  , [P].[createdAt]  , [P].[updatedAt]  FROM([dbo].[People] as [P]  INNER JOIN [dbo].[Employees] as [E]  ON [P].[peopleID] = [E].[employeeID])  JOIN [dbo].[Departments] as [D]  ON ([E].[departmentID] = [D].[departmentID])  WHERE [E].[position] = N'Bác sĩ';  GO |

- Stored procedure “FindDepartmentNameByDoctorId”:

|  |
| --- |
| --Begin: PROCEDURE FindDepartmentNameByDoctorId  CREATE PROCEDURE FindDepartmentNameByDoctorId(@doctorID NVARCHAR(25))  AS  BEGIN  SELECT departmentName FROM View\_Doctor WHERE doctorID = @doctorID;  END  GO  --End: PROCEDURE FindDepartmentNameByDoctorId |

#### 4.2.1.5. Chức năng lấy họ và tên doctor bằng id doctor

- Function “func\_FullName”:

|  |
| --- |
| -- BEGIN: func\_FullName  CREATE FUNCTION func\_FullName(@peopleID VARCHAR(20))  RETURNS NVARCHAR(100)  AS  BEGIN  DECLARE @firstName NVARCHAR(32)  DECLARE @lastName NVARCHAR(32)  DECLARE @fullName NVARCHAR(100)  SELECT @firstName = firstName FROM People WHERE peopleID = @peopleID  SELECT @lastName = lastName FROM People WHERE peopleID = @peopleID  SET @fullName = @lastName + ' ' + @firstName  RETURN @fullName  END  GO  -- END: func\_FullName |

- Stored procedure “FindFullNameDoctorByDoctorId”:

|  |
| --- |
| CREATE PROCEDURE FindFullNameDoctorByDoctorId(@doctorID NVARCHAR(25))  AS  BEGIN  SELECT dbo.func\_FullName(@doctorID);  END  GO |

#### 4.2.1.6. Chức năng thêm thông tin vào bảng Examination đồng thời tạo đơn thuốc rỗng

- Stored procedure “spUpdateExamination”:

|  |
| --- |
| CREATE PROCEDURE spUpdateExamination(@patientID VARCHAR(20),  @employeeID VARCHAR(20),  @height FLOAT,  @weight FLOAT,  @temperature FLOAT,  @breathing INT,  @symptom NVARCHAR(510),  @veins INT,  @bloodPressure INT,  @preliminaryDiagnosis NVARCHAR(510),  @finalDiagnosis NVARCHAR(510),  @treatmentDirection NVARCHAR(510),  @examinateID VARCHAR(20))  AS  BEGIN  BEGIN TRANSACTION Tran\_UpdateExamination  BEGIN TRY  UPDATE dbo.Examination SET patientID = @patientID, employeeID = @employeeID, height = @height,  weight = @weight, temperature = @temperature, breathing = @breathing, symptom = @symptom, veins = @veins, bloodPressure = @bloodPressure,  preliminaryDiagnosis = @preliminaryDiagnosis, finalDiagnosis = @finalDiagnosis, treatmentDirection = treatmentDirection WHERE examinateID = @examinateID  INSERT INTO dbo.Prescriptions (descriptionPrescription) VALUES (NULL)  DECLARE @prescriptionID VARCHAR(20)  SET @prescriptionID = dbo.func\_GetPrescriptionID\_New()  INSERT INTO dbo.CreatedPrescription (patientID, employeeID, prescriptionID, examinateID)  VALUES (@patientID, @employeeID, @prescriptionID, @examinateID)  COMMIT TRANSACTION Tran\_UpdateExamination  END TRY  BEGIN CATCH  PRINT('Thêm không thành công')  -- SELECT @ErrorNumber = ERROR\_NUMBER()  -- SELECT ERROR\_NUMBER() AS ErrorNumber,  -- ERROR\_MESSAGE() AS ErrorMessage;  ROLLBACK TRANSACTION Tran\_UpdateExamination  END CATCH  END  GO |

- Trigger “trg\_Inserted\_Examination”:

|  |
| --- |
| -- Begin: trigger Inserted in Examination  CREATE TRIGGER trg\_Inserted\_Examination ON dbo.Examination  FOR INSERT  AS  DECLARE @createdAt DATETIME  DECLARE @examinateID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @examinateID = Inserted.examinateID, @createdAt = Inserted.createdAt  FROM Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.Examination SET createdAt = @createdAt WHERE examinateID = @examinateID  SET @examinateID = dbo.func\_Auto\_examinateID()  UPDATE [Examination] SET examinateID = @examinateID WHERE examinateID = 'XX0000'  END  END  GO  -- End: trigger Inserted in Examination |

- Function “func\_Auto\_examinateID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_examinateID  CREATE FUNCTION func\_Auto\_examinateID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'EX'  END  SELECT @max = COUNT(examinateID) FROM [Examination]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT examinateID FROM [Examination] WHERE examinateID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_examinateID |

- Function “func\_GetPrescriptionID\_New”:

|  |
| --- |
| CREATE FUNCTION func\_GetPrescriptionID\_New()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @ID VARCHAR(20)  SELECT @ID = prescriptionID FROM Prescriptions WHERE createdAt = (SELECT MAX(createdAt) FROM Prescriptions)  RETURN @ID  END  GO |

- Trigger “trg\_Inserted\_Updated\_Prescription”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in Prescriptions  CREATE TRIGGER trg\_Inserted\_Updated\_Prescription ON dbo.Prescriptions  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @prescriptionID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @prescriptionID = Inserted.prescriptionID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.Prescriptions SET createdAt = @createdAt WHERE prescriptionID = @prescriptionID  SET @prescriptionID = dbo.func\_Auto\_prescriptionID()  UPDATE [Prescriptions] SET prescriptionID = @prescriptionID WHERE prescriptionID = 'XX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.Prescriptions SET updatedAt = @updatedAt WHERE prescriptionID = @prescriptionID  END  END  GO  -- End: trigger Inserted and Updated in Prescriptions |

- Function “dbo.func\_Auto\_prescriptionID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_prescriptionID  CREATE FUNCTION func\_Auto\_prescriptionID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'DT'  END  SELECT @max = COUNT(prescriptionID) FROM [Prescriptions]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT prescriptionID FROM [Prescriptions] WHERE prescriptionID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_prescriptionID |

### 4.2.2. Chức năng trong form “frmPrescription”

#### 4.2.2.1. Chức năng hiển thị thông tin thuốc được lấy

- View “View\_CreatedPrescription\_New”:

|  |
| --- |
| CREATE VIEW [View\_CreatedPrescription\_New]  AS  SELECT P.prescriptionID, M.medicineID, M.medicineName, M.medicineGroupName, M.shape, M.unit, M.medicinePrice, M.expiry, M.count, P.quality  FROM dbo.Medicines AS [M]  INNER JOIN dbo.Prescription\_Medicine AS [P] ON M.medicineName = P.medicineName  GO |

- Stored procedure “spFindAllPrescription\_Medicine”:

|  |
| --- |
| CREATE PROCEDURE spFindAllPrescription\_Medicine(@prescriptionID VARCHAR(20))  AS  BEGIN  SELECT \* FROM [View\_CreatedPrescription\_New] WHERE prescriptionID = @prescriptionID  END  GO |

#### 4.2.2.3. Chức năng thêm thuốc vào đơn thuốc

- Stored procedure “spInsertPrescription\_Medicine”:

|  |
| --- |
| CREATE PROCEDURE spInsertPrescription\_Medicine(@prescriptionID VARCHAR(20), @medicineName NVARCHAR(255), @quanlity INT)  AS  BEGIN  INSERT INTO dbo.Prescription\_Medicine (prescriptionID, medicineName, quality)  VALUES (@prescriptionID, @medicineName, @quanlity)  END  GO |

- Trigger “trg\_Inserted\_Prescription\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Inserted\_Prescription\_Medicine ON dbo.Prescription\_Medicine  FOR INSERT  AS  DECLARE @prescriptionID VARCHAR(20)  DECLARE @medicineName NVARCHAR(255)  DECLARE @quanlity INT  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @prescriptionID = Inserted.prescriptionID, @medicineName = Inserted.medicineName, @quanlity = Inserted.quanlity FROM Inserted  DECLARE @count INT  SELECT @count = count FROM dbo.Medicines WHERE medicineName = @medicineName  SET @count = @count - @quanlity      UPDATE dbo.Medicines SET [count] = @count WHERE medicineName = @medicineName  UPDATE dbo.Prescriptions SET [status] = 2 WHERE prescriptionID = @prescriptionID  END  GO |

#### 4.2.2.4. Chức năng cập nhật thuốc vào đơn thuốc

- Stored procedure “spUpdatePrescription\_Medicine”:

|  |
| --- |
| CREATE PROCEDURE spUpdatePrescription\_Medicine(@prescriptionID VARCHAR(20), @medicineName NVARCHAR(255), @quanlity INT)  AS  BEGIN  UPDATE dbo.Prescription\_Medicine SET quanlity = @quanlity WHERE prescriptionID = @prescriptionID AND medicineName = @medicineName  END  GO |

- Trigger “trg\_Updated\_Prescription\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Updated\_Prescription\_Medicine ON dbo.Prescription\_Medicine  FOR UPDATE  AS  DECLARE @prescriptionID VARCHAR(20)  DECLARE @medicineName NVARCHAR(255)  DECLARE @quanlity INT  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @prescriptionID = deleted.prescriptionID, @medicineName = deleted.medicineName, @quanlity = deleted.quality FROM deleted  DECLARE @count INT  SET @count = dbo.func\_UpdateQuanlityMedicine(@prescriptionID, @medicineName, @quanlity)    UPDATE dbo.Medicines SET [count] = @count WHERE medicineName = @medicineName  END  GO |

- Function “func\_UpdateQuanlityMedicine”:

|  |
| --- |
| CREATE FUNCTION func\_UpdateQuanlityMedicine(@prescriptionID VARCHAR(20), @medicineName NVARCHAR(255), @quanlity INT)  RETURNS INT  BEGIN  DECLARE @count INT  DECLARE @quanlityNew INT  SELECT @count = count FROM dbo.Medicines WHERE medicineName = @medicineName  SELECT @quanlityNew = quality FROM dbo.Prescription\_Medicine WHERE medicineName = @medicineName AND prescriptionID = @prescriptionID    SET @count = @count + @quanlity - @quanlityNew  RETURN @count  END  GO |

#### 4.2.2.5. Chức năng xóa thuốc khởi đơn thuốc

- Stored procedure “spDeletePrescription\_Medicine”:

|  |
| --- |
| CREATE PROCEDURE spDeletePrescription\_Medicine(@prescriptionID VARCHAR(20), @medicineName NVARCHAR(255))  AS  BEGIN  BEGIN TRANSACTION Tran\_DeletePrescription\_Medicine  BEGIN TRY  DECLARE @quanlity INT  SELECT @quanlity = quality FROM dbo.Prescription\_Medicine WHERE medicineName = @medicineName AND prescriptionID = @prescriptionID  DECLARE @count INT  SELECT @count = count FROM dbo.Medicines WHERE medicineName = @medicineName  SET @count = @count + @quanlity  UPDATE dbo.Medicines SET [count] = @count WHERE medicineName = @medicineName  DELETE dbo.Prescription\_Medicine WHERE prescriptionID = @prescriptionID AND medicineName = @medicineName  COMMIT TRANSACTION Tran\_DeletePrescription\_Medicine  END TRY  BEGIN CATCH  PRINT('Xóa không thành công')  -- SELECT @ErrorNumber = ERROR\_NUMBER()  -- SELECT ERROR\_NUMBER() AS ErrorNumber,  -- ERROR\_MESSAGE() AS ErrorMessage;  ROLLBACK TRANSACTION Tran\_DeletePrescription\_Medicine  END CATCH  END  GO |

- Trigger “trg\_Deleted\_Prescription\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Deleted\_Prescription\_Medicine ON dbo.Prescription\_Medicine  FOR DELETE  AS  DECLARE @prescriptionID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @prescriptionID = deleted.prescriptionID FROM deleted  DECLARE @count INT  SELECT @count = COUNT(\*) FROM Prescription\_Medicine WHERE prescriptionID = @prescriptionID  IF (@count = 0)  BEGIN  UPDATE dbo.Prescriptions SET [status] = 1 WHERE prescriptionID = @prescriptionID  END  END  GO |

#### 4.2.2.6. Cập nhật thông tin đơn thuốc rỗng đã tạo

- Stored procedure “spUpdatePrescription”:

|  |
| --- |
| CREATE PROCEDURE spUpdatePrescription(@prescriptionID VARCHAR(20), @descriptionPrescription NVARCHAR(500))  AS  BEGIN  UPDATE dbo.Prescriptions SET descriptionPrescription = @descriptionPrescription WHERE prescriptionID = @prescriptionID  END  GO |

- Trigger “trg\_Inserted\_Updated\_Prescription”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in Prescriptions  CREATE TRIGGER trg\_Inserted\_Updated\_Prescription ON dbo.Prescriptions  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @prescriptionID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @prescriptionID = Inserted.prescriptionID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.Prescriptions SET createdAt = @createdAt WHERE prescriptionID = @prescriptionID  SET @prescriptionID = dbo.func\_Auto\_prescriptionID()  UPDATE [Prescriptions] SET prescriptionID = @prescriptionID WHERE prescriptionID = 'XX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.Prescriptions SET updatedAt = @updatedAt WHERE prescriptionID = @prescriptionID  END  END  GO  -- End: trigger Inserted and Updated in Prescriptions |

- Function “func\_Auto\_prescriptionID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_prescriptionID  CREATE FUNCTION func\_Auto\_prescriptionID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'DT'  END  SELECT @max = COUNT(prescriptionID) FROM [Prescriptions]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT prescriptionID FROM [Prescriptions] WHERE prescriptionID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_prescriptionID |

## 4.3. Chức năng Nhân viên

### 4.3.1. Chức năng trong form “TiepNhanBenhNhan”

#### 4.3.1.1. Chức năng xem danh sách bệnh nhân

- View “View\_Patient”:

|  |
| --- |
| --================== View 'View\_Patients' để xem các thông tin cơ bản của bệnh nhân  CREATE VIEW [dbo].[View\_Patients] AS SELECT [P].[peopleID] as [patientID]  , [P].[firstName] as [patientFirstName]  , [P].[lastName] as [patientLastname]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [P2].[patientJob]  , [P2].[healthInsurance]  , [P2].[reason]  , [P].[createdAt]  , [P].[updatedAt]  FROM [dbo].[People] as [P]  INNER JOIN [dbo].[Patients] as [P2]  ON ([P].[peopleID] = [P2].[patientID]) |

#### 4.3.1.2 Chức năng thêm bệnh nhân với chờ khám

Để cho bệnh nhân vào chờ khám thì ta tạo mới 1 record của bảng Examination với các trường thông tin null và trường ‘status’ bằng 1 để đánh dấu là đang chờ.

- Store Procedure “proc\_createNewNullExamination”:

|  |
| --- |
| --============================= procedure 'proc\_createNewNullExamination'  CREATE PROCEDURE [dbo].[proc\_createNewNullExamination]  (  @patientID VARCHAR(20),  @employeeID VARCHAR(20)  )  AS  BEGIN  -- khởi tạo đơn khám null tất cả các thuộc tính,  -- chỉ có ID bệnh nhân và ngày tạo  -- ID nhân viên ở đây lấy mặc định là  -- của nhân viên lễ tân đăng nhập và tạo Exam  INSERT INTO Examination  (patientID, employeeID, status)  VALUES  (@patientID, @employeeID, 1)  END |

- Khi thêm mới 1 Examination thì Trigger “trg\_Inserted\_Examination” sẽ thực thi:

|  |
| --- |
| --============================= procedure 'proc\_createNewNullExamination'  CREATE PROCEDURE [dbo].[proc\_createNewNullExamination]  (  @patientID VARCHAR(20),  @employeeID VARCHAR(20)  )  AS  BEGIN  -- khởi tạo đơn khám null tất cả các thuộc tính,  -- chỉ có ID bệnh nhân và ngày tạo  -- ID nhân viên ở đây lấy mặc định là  -- của nhân viên lễ tân đăng nhập và tạo Exam  INSERT INTO Examination  (patientID, employeeID, status)  VALUES  (@patientID, @employeeID, 1)  END |

## 4.3.2. Chức năng trong form “ThongtinBenhNhan”

#### 4.3.2.1. Chức năng xem danh sách thân nhân của bệnh nhân

**-** Function “func\_getRelatives”:

|  |
| --- |
| --=============================== FUNCTION 'func\_getRelatives'  /\*  FUNCTION lấy danh sách các thân nhân của 1 bệnh nhân    tham số truyền vào: VARCHAR(20) : id bệnh nhân cần xem  trả về: TABLE : danh sách các id, họ tên của thân nhân mà bệnh nhân ấy có  \*/  CREATE FUNCTION [dbo].[func\_getRelatives]  (  @patientID VARCHAR(20)  )  RETURNS @a TABLE  (  relativeID VARCHAR(20),  firstName NVARCHAR(32),  lastName NVARCHAR(32)  )  AS  BEGIN  INSERT INTO @a  SELECT R.relativeID, P.firstName, P.lastName FROM  (Relatives as R INNER JOIN People AS P ON (R.relativeID = P.peopleID))  INNER JOIN View\_Patients as Pa ON (R.patientID = Pa.patientID)  WHERE (R.patientID = @patientID)  RETURN  END |

#### 4.3.2.2. Chức năng cập nhật thông tin bệnh nhân

- Thực thi bằng câu lệnh C#:

|  |
| --- |
| public void Update(View\_Patients newInfo)  {  // Update people  Person person = db.People.Single(p => p.peopleID.Equals(newInfo.patientID));  person.firstName = newInfo.patientFirstName;  person.lastName = newInfo.patientLastname;  person.sex = newInfo.sex;  person.birthDay = newInfo.birthDay;  person.address = newInfo.address;  person.phone = newInfo.phone;  person.cardID = newInfo.cardID;  db.People.AddOrUpdate(person);  db.SaveChanges();    // Update patient  Patient patient = db.Patients.Single(p => p.patientID.Equals(newInfo.patientID));  patient.patientJob = newInfo.patientJob;  patient.healthInsurance = newInfo.healthInsurance;  patient.reason = newInfo.reason;  db.Patients.AddOrUpdate(patient);  db.SaveChanges();  } |

- Khi cập nhật thông tin, Trigger “trg\_Inserted\_Updated\_People” được thực thi:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER [dbo].[trg\_Inserted\_Updated\_People] ON [dbo].[People]  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN    SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID    -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'    -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END |

#### 4.3.2.3. Chức năng xem lịch sử khám của bệnh nhân

- View “View\_Examination":

|  |
| --- |
| --==================== View 'View\_Examination' để xem các thông tin bên trong 1 đơn khám  -- ================== chỉ lấy những đơn khám chưa bị xóa (kể cả những đơn khám đang đợi)  CREATE VIEW [dbo].[View\_Examination] AS SELECT  Ex.examinateID,  Ex.patientID,  P1.firstName as tenBenhNhan,  P1.lastName as hoBenhNhan,  Ex.employeeID,  P2.firstName as tenBacSi,  P2.lastName as hoBacSi,  Ex.[height],  Ex.[weight],  Ex.[temperature],  Ex.[breathing],  Ex.[symptom],  Ex.[veins],  Ex.[bloodPressure],  Ex.[preliminaryDiagnosis],  Ex.[finalDiagnosis],  Ex.[treatmentDirection],  Ex.[createdAt],  Ex.status  FROM ((Examination as Ex INNER JOIN  (Patients as Pa  INNER JOIN  People AS P1  ON (Pa.patientID = P1.peopleID))  ON (Ex.patientID = Pa.patientID))  INNER JOIN  (Employees as Em  INNER JOIN  People as P2  ON (Em.employeeID = P2.peopleID))  ON (Ex.employeeID = Em.employeeID))  WHERE  Ex.status != 0 |

**-** Function “func\_getExamHistory”:

|  |
| --- |
| --=============================== FUNCTION 'func\_getExamHistory'  /\*  FUNCTION lấy lịch sử các lần khám trước của bệnh nhân  trả về các thông tin cơ bản của những lần khám đó    tham số đầu vào: VARCHAR(20) : id bệnh nhân cần xem  trả về: TABLE multi value với thông tin các lần khám trước  \*/  CREATE FUNCTION [dbo].[func\_getExamHistory]  (  @patientID VARCHAR(20)  )  RETURNS @a TABLE  (  examinateID VARCHAR(20),  patientID VARCHAR(20),  tenBenhNhan NVARCHAR(32),  hoBenhNhan NVARCHAR(32),  employeeID VARCHAR(20),  tenBacSi NVARCHAR(32),  hoBacSi NVARCHAR(32),  height FLOAT,  weight FLOAT,  temperature FLOAT,  breathing INT,  symptom NVARCHAR(510),  veins INT,  bloodPressure INT,  preliminaryDiagnosis NVARCHAR(510),  finalDiagnosis NVARCHAR(510),  treatmentDirection NVARCHAR(510),  createdAt DATETIME,  status INT  )  AS  BEGIN  INSERT INTO @a  SELECT  \*  FROM View\_Examination as VE  WHERE VE.patientID = @patientID AND VE.status = 2  -- status = 2 nghĩa là đã khám rồi  ORDER BY VE.createdAt DESC  RETURN  END |

- Function “func\_getExamWithID”:

|  |
| --- |
| --=============================== FUNCTION 'func\_getExamWithID'  /\*  FUNCTION xem thông tin các đơn khám thông qua ID    tham số truyền vào: VARCHAR(20) : id đơn khám cần tìm  trả về : TABLE : thông tin của đơn khám cần tìm  \*/  CREATE FUNCTION [dbo].[func\_getExamWithID]  (  @examID VARCHAR(20)  )  RETURNS @a TABLE  (  examinateID VARCHAR(20),  patientID VARCHAR(20),  tenBenhNhan NVARCHAR(32),  hoBenhNhan NVARCHAR(32),  employeeID VARCHAR(20),  tenBacSi NVARCHAR(32),  hoBacSi NVARCHAR(32),  height FLOAT,  weight FLOAT,  temperature FLOAT,  breathing INT,  symptom NVARCHAR(510),  veins INT,  bloodPressure INT,  preliminaryDiagnosis NVARCHAR(510),  finalDiagnosis NVARCHAR(510),  treatmentDirection NVARCHAR(510),  createdAt DATETIME,  status INT  )  AS  BEGIN  INSERT INTO @a  SELECT  \*  FROM View\_Examination as VE  WHERE VE.examinateID = @examID  RETURN  END |

#### 4.3.2.3. Chức năng thêm bệnh nhân

- Câu lệnh thực thi bằng C#:

|  |
| --- |
| public void Create(Person person, Patient patient)  {  db.InsertPatient(person.firstName,  person.lastName,  person.sex,  person.birthDay,  person.address,  person.phone,  person.cardID,  patient.patientJob,  patient.healthInsurance,  patient.reason);  db.SaveChanges();  } |

- Khi cập nhật thông tin, Trigger ‘trg\_Inserted\_Updated\_People’ được thực thi (trong mục 4.3.2.2)

## 4.3.3. Chức năng trong form “ThemThanNhan”

### 4.3.3.1. Chức năng thêm thân nhân đã tồn tại trong cơ sở dữ liệu

Khi nhân viên nhân thấy thân nhân của bệnh nhân đã có thông tin trong cơ sở dữ liệu của phòng khám, nhân viên sẽ chọn thông tin của người ấy và thêm vào thông tin thân nhân của bệnh nhân.

Stored Procedure ‘InsertRelative\_Exist’:

Tham số đầu vào:

- @patientID: ID của bệnh nhân được chọn thêm thân nhân

- @relativeID: ID của người được chọn.

|  |
| --- |
| CREATE PROCEDURE [dbo].[InsertRelative\_Exist] (@patientID VARCHAR(20), @relativeID VARCHAR(20))  AS  BEGIN  INSERT INTO [dbo].[Relatives]  ([relativeID],[patientID])  VALUES  (@relativeID, @patientID)  END |

#### 4.3.3.2. Chức năng thêm thân nhân chưa tồn tại trong cơ sở dữ liệu

Khi nhân viên nhân thấy thân nhân của bệnh nhân chưa có thông tin trong cơ sở dữ liệu, nhân viên sẽ tạo thêm 1 người mới trong cơ sở dữ liệu và sau đó gán thông tin của người đó vào làm thân nhân của bệnh nhân đang được thêm.

Stored Procedure “InsertRelative”: Tham số đầu vào:

- @patientID: ID của bệnh nhân được chọn thêm thân nhân

- Các thông tin cơ bản của người thân

|  |
| --- |
| --Begin: PROCEDURE InsertRelative  CREATE PROCEDURE [dbo].[InsertRelative] (@relativeFirstName nvarchar(32),  @relativeLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @patientID VARCHAR(20))  AS  BEGIN  BEGIN TRANSACTION Tran\_InsertRelative  BEGIN TRY  DECLARE @role TINYINT  SET @role = 0  INSERT INTO dbo.People (firstName, lastName, sex, birthDay, address, phone, cardID, role)  VALUES (@relativeFirstName, @relativeLastname, @sex, @birthDay, @address, @phone, @cardID, @role)      DECLARE @relativeID VARCHAR(20)  SELECT @relativeID = MAX(peopleID) FROM dbo.People WHERE role = @role    INSERT INTO dbo.Relatives (relativeID, patientID)  VALUES(@relativeID, @patientID)    COMMIT TRANSACTION Tran\_InsertRelative  END TRY  BEGIN CATCH  PRINT('Thêm không thành công')  ROLLBACK TRANSACTION Tran\_InsertRelative  END CATCH  END |

- Khi thân nhân mới được thêm vào, Trigger ‘trg\_Inserted\_Updated\_People’ được thực thi (trong mục 4.3.2.2)

### 4.3.4. Chức năng trong form “ThongTinDonKham”:

Chức năng xem danh sách đơn thuốc của đơn khám.

- View “View\_ListMedicine\_Precesciption”

|  |
| --- |
| CREATE VIEW [dbo].[View\_ListMedicine\_Precesciption] AS SELECT  CP.prescriptionID,  Cp.examinateID,  patientID,  employeeID,  medicineID,  M.medicineName,  PM.quanlity,  M.medicinePrice,  PM.quanlity \* M.medicinePrice as totalPrice,  P.createdAt,  P.status  FROM  CreatedPrescription as CP  INNER JOIN  (Prescription\_Medicine as PM INNER JOIN Prescriptions AS P ON (PM.prescriptionID = P.prescriptionID)  INNER JOIN  Medicines as M  ON (PM.medicineName = M.medicineName))  ON (CP.prescriptionID = CP.prescriptionID) |

**-** Function ‘func\_getListMedicion\_withExamID’:

|  |
| --- |
| /\*  Functione 'func\_getListMedicion\_withExamID'  Lấy danh sách các thuốc trong đơn khám    tham số truyền vào : VARCHAR(20) : ID đơn khám  trả về : danh sách các đơn thuốc, danh sách thuốc  lấy cả tên thuốc, tên bệnh nhân và giá tiền theo số lượng  \*/  CREATE FUNCTION [dbo].[func\_getListMedicion\_withExamID]  (  @examID VARCHAR(20)  )  RETURNS @a TABLE  (  prescriptionID VARCHAR(20),  examinateID VARCHAR(20),  patientID VARCHAR(20),  employeeID VARCHAR(20),  medicineID VARCHAR(20),  medicineName NVARCHAR(255),  quanlity INT,  medicinePrice INT,  totalPrice INT,  createdAt DATE,  status tinyint  )  AS  BEGIN  INSERT INTO @a  SELECT \*  FROM View\_ListMedicine\_Precesciption  WHERE (examinateID = @examID)  RETURN  END |

### 4.3.5. Chức năng trong form “SuDungDichVu”

#### 4.3.5.1. Chức năng chọn sử dụng dịch vụ

- Procedure “proc\_InsertUsingService”:

|  |
| --- |
| --================================== procedure 'proc\_InsertUsingService'  /\*  procedure tạo 1 using service  với đầu vào là  ID bệnh nhân,  ID dịch vụ chọn sử dụng  và số lượng dịch vụ đó  với status bằng 1 có nghĩa là usingService đó chưa bị xóa và chưa được thanh toán    \*/  CREATE PROCEDURE [dbo].[proc\_InsertUsingService]  (  @patientID VARCHAR(20),  @serviceID VARCHAR(20),  @quantity tinyint  )  AS  BEGIN  INSERT INTO UsingService  (patientID, serviceID, quantity, status)  VALUES  (@patientID, @serviceID, @quantity, 1)  END |

- Khi 1 UsingService được thêm mới thì Trigger “UsingService\_CreatedAt” được thực thi.

|  |
| --- |
| --Begin: trigger createdAt in UsingService  CREATE TRIGGER [dbo].[UsingService\_CreatedAt] ON [dbo].[UsingService]  FOR INSERT  AS  DECLARE @createdAt DATETIME  DECLARE @usingServiceID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN    SELECT @usingServiceID = Inserted.usingServiceID, @createdAt = Inserted.createdAt  FROM Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.UsingService SET createdAt = @createdAt WHERE usingServiceID = @usingServiceID  END  END |

#### 4.3.5.2. Chức năng bỏ sử dụng dịch vụ

**-** Thực thi câu lệnh trong C#:

|  |
| --- |
| public void removeUsingService(int usID)  {  UsingService us = db.UsingServices.Where(u => u.usingServiceID == usID).FirstOrDefault();  db.UsingServices.Remove(us);  db.SaveChanges();  } |

#### 4.3.5.3. Chức năng thay đổi sử dụng dịch vụ:

**-** Procedure “proc\_UpdateUsingService”:

|  |
| --- |
| --================================== procedure 'proc\_UpdateUsingService'  /\*  PROCEDURE cập nhật thông tin của usingService với ID là usingServiceID truyền vào  được cập nhật mới các thông tin như  id bệnh nhân sử dụng  id dịch vụ sử dụng  và số lượng của dịch vụ được chọn  \*/  CREATE PROCEDURE [dbo].[proc\_UpdateUsingService]  (  @usingServiceID VARCHAR(20),  @patientID VARCHAR(20),  @serviceID VARCHAR(20),  @quantity tinyint  )  AS  BEGIN  UPDATE UsingService  SET  patientID = @patientID,  serviceID = @serviceID,  quantity = @quantity  WHERE  usingServiceID = @usingServiceID  END |

### 4.3.6. Chức năng trong form “ThanhToanSuDungDichVu”

* View “View\_UsingService”

|  |
| --- |
| --=========================================== view 'View\_UsingService' để lấy các thông tin cơ bản của khách hàng sử dụng dịch vụ  --=========================================== có cả tính tiền tổng theo số lượng  CREATE VIEW [dbo].[View\_UsingService] AS SELECT  US.usingServiceID,  US.patientID,  S.serviceID,  S.serviceName,  S.servicePrice,  US.quantity,  US.status,  S.servicePrice \* US.quantity AS totalPay,  US.createdAt  FROM  (UsingService as US  INNER JOIN  Services AS S  ON (US.serviceID = S.serviceID)) |

- Function “func\_getListService\_notYet”:

|  |
| --- |
| --=============================== FUNCTION 'func\_getListService\_notYet'  /\*  FUNCTION lấy thông tin các dịch vụ bệnh nhân đã sử dụng mà chưa thanh toán  để thanh toán  tham số đầu vào: VARCHAR(20) : id bệnh nhân cần xem  trả về: TABLE thông tin danh sách các dịch vụ bệnh nhân đã sử dụng mà chưa thanh toán  \*/  CREATE FUNCTION [dbo].[func\_getListService\_notYet](@patientID VARCHAR(20))  RETURNS @usingServiceList TABLE  (  usingServiceID INT,  patientID VARCHAR(20),  serviceID VARCHAR(20),  serviceName NVARCHAR(255),  unitPrice int,  quantity TINYINT,  status TINYINT,  payTotal int,  createdAt DATETIME  )  AS  BEGIN  INSERT INTO @usingServiceList  SELECT \*  FROM  View\_UsingService AS VUS  WHERE  VUS.status = 1 AND VUS.patientID = @patientID  RETURN  END |

- Function “func\_getTotalPay\_usingService”

|  |
| --- |
| --=============================== FUNCTION 'func\_getTotalPay\_usingService'  /\*  FUNCTION lấy tổng giá tiền của các dịch vụ  mà bệnh nhân chọn sử dụng mà chưa thanh toán    tham số đầu vào: VARCHAR(20) : ID bệnh nhân  trả về: INT  \*/  CREATE FUNCTION [dbo].[func\_getTotalPay\_usingService] (@patientID VARCHAR(20))  RETURNS INT  AS  BEGIN  DECLARE @totalPay INT    SELECT @totalPay = SUM(payTotal)  FROM func\_getListService\_notYet(@patientID)    RETURN @totalPay  END |

- Procedure “proc\_InsertReceipt”:

|  |
| --- |
| --=============================== PROCEDURE 'proc\_InsertReceipt'  /\*  Tạo hóa đơn với tên tạo từ loại hóa đơn và id bệnh nhân  \*/  CREATE PROCEDURE [dbo].[proc\_InsertReceipt]  (  @type VARCHAR(50),  @patientID VARCHAR(20),  @employeeID VARCHAR(20)  )  AS  BEGIN  INSERT INTO Receipts (receiptName, status)  VALUES (@type + '\_' + @patientID + '\_' + @employeeID, 1)  END  --==================================================== |

* Procedure “proc\_InsertPay”

|  |
| --- |
| --====================== PROCEDURE 'proc\_InsertPay'  /\*  Nhập pay mới  \*/  CREATE PROCEDURE [dbo].[proc\_InsertPay]  (  @patientID VARCHAR(20),  @employeeID VARCHAR(20),  @receiptID VARCHAR(20),  @payTotal INT  )  AS  BEGIN  INSERT INTO Pay  (patientID, employeeID, receiptID, payTotal, status)  VALUES  (@patientID, @employeeID, @receiptID, @payTotal, 1)  END  --==================================================== |

Procedure “proc\_Pay\_service”:

|  |
| --- |
| -- ====================================== PROCEDURE 'proc\_Pay\_service'  --================= Thanh toán dịch vụ đã sử dụng  /\*  Tạo hóa đơn Pay và Receipt  Chuyển trạng thái status của các UsingService mà người dùng sử dụng,  chưa thanh toán thành 3 (đã thanh toán)  \*/  --=================  /\*  tham số đầu vào:  id bệnh nhân : VARCHAR(20)  id nhân viên thanh toán : VARCHAR(20)  \*/  CREATE PROCEDURE [dbo].[proc\_Pay\_service]  (  @patientID VARCHAR(20),  @employeeID VARCHAR(20)  )  AS  BEGIN  --========== Tạo Receipts mới  EXEC proc\_InsertReceipt 'Service', @patientID, @employeeID    DECLARE @receiptID VARCHAR(20)  SELECT @receiptID = MAX(receiptID) FROM Receipts    --=========== Tạo Pay mới  DECLARE @totayPay INT  SELECT @totayPay = [dbo].[func\_getTotalPay\_usingService](@patientID)    EXEC proc\_InsertPay @patientID, @employeeID, @receiptID, @totayPay    --======== Cập nhật status của các UsingService của bệnh nhân  ---- mà chưa thanh toán thành 3  DECLARE @usingServiceID INT,  @Count\_US INT    SELECT @Count\_US = COUNT(usingServiceID)  FROM [dbo].[func\_getListService\_notYet](@patientID)    WHILE @Count\_US > 0  BEGIN    SELECT @usingServiceID = MIN(usingServiceID) FROM [dbo].[func\_getListService\_notYet](@patientID)    UPDATE UsingService  SET status = 3  WHERE usingServiceID = @usingServiceID    SELECT @Count\_US = COUNT(usingServiceID) FROM [dbo].[func\_getListService\_notYet](@patientID)    END  END  --==================================================== |

### 4.3.7. Chức năng trong form “ThanhToanDonThuoc”:

* Function “View\_ListMedicine\_Precesciption”:

|  |
| --- |
| /\*  View 'View\_ListMedicine\_Precesciption' lấy danh sách các đơn thuốc, danh sách thuốc  lấy cả tên thuốc, tên bệnh nhân và giá tiền theo số lượng  \*/  CREATE VIEW [dbo].[View\_ListMedicine\_Precesciption]  AS  SELECT  CP.prescriptionID,  Cp.examinateID,  patientID,  employeeID,  medicineID,  M.medicineName,  PM.quanlity,  M.medicinePrice,  PM.quanlity \* M.medicinePrice as totalPrice,  P.createdAt,  P.status  FROM  CreatedPrescription as CP  INNER JOIN  (Prescription\_Medicine as PM INNER JOIN Prescriptions AS P ON (PM.prescriptionID = P.prescriptionID)  INNER JOIN  Medicines as M  ON (PM.medicineName = M.medicineName))  ON (CP.prescriptionID = CP.prescriptionID)  --=============================================================== |

* Function “func\_getListMedicion\_withExamID”:

|  |
| --- |
| /\*  Functione 'func\_getListMedicion\_withExamID'  Lấy danh sách các thuốc trong đơn khám    tham số truyền vào : VARCHAR(20) : ID đơn khám  trả về : danh sách các đơn thuốc, danh sách thuốc  lấy cả tên thuốc, tên bệnh nhân và giá tiền theo số lượng  \*/  CREATE FUNCTION [dbo].[func\_getListMedicion\_withExamID]  (  @examID VARCHAR(20)  )  RETURNS @a TABLE  (  prescriptionID VARCHAR(20),  examinateID VARCHAR(20),  patientID VARCHAR(20),  employeeID VARCHAR(20),  medicineID VARCHAR(20),  medicineName NVARCHAR(255),  quanlity INT,  medicinePrice INT,  totalPrice INT,  createdAt DATE,  status tinyint  )  AS  BEGIN  INSERT INTO @a  SELECT \*  FROM View\_ListMedicine\_Precesciption  WHERE (examinateID = @examID)  RETURN  END |

* Function “func\_getTotalPay\_ListMedicine\_withExamID”:

|  |
| --- |
| /\*  FUNCTION 'func\_getTotalPay\_ListMedicine\_withExamID'  Lấy tổng giá tiền của đơn thuốc trong đơn khám    Tham số truyền vào : VARCHAR(20) : ID đơn khám  trả về : integer  \*/    CREATE FUNCTION [dbo].[func\_getTotalPay\_ListMedicine\_withExamID]  (  @examID VARCHAR(20)  )  RETURNS INT  AS  BEGIN  DECLARE @totayPay INT  SELECT @totayPay = SUM(totalPrice)  FROM [dbo].[func\_getListMedicion\_withExamID](@examID) AS fGLM    RETURN @totayPay  END |

* Function “func\_getPrecription\_notYet”:

|  |
| --- |
| /\*  FUNCTION 'func\_getPrecription\_notYet'  Lấy danh sách các đơn thuốc của bệnh nhân chưa thanh toán    tham số truyền vào : id bệnh nhân : VARCHAR(20)  \*/  CREATE FUNCTION [dbo].[func\_getPrecription\_notYet]  (  @patientID VARCHAR(20)  )  RETURNS @a TABLE  (  examinateID VARCHAR(20),  HoBacSi NVARCHAR(32),  TenBacSi NVARCHAR(32),  createdAt DATE  )  AS  BEGIN  INSERT INTO @a  SELECT DISTINCT  VE.examinateID,  VE.hoBacSi,  VE.tenBacSi,  VE.createdAt  FROM  View\_ListMedicine\_Precesciption as VLP  INNER JOIN  View\_Examination as VE  ON (VLP.examinateID = VE.examinateID)  WHERE  (VE.patientID = @patientID)  AND  (VLP.status = 2)  RETURN  END |

* Function “proc\_Pay\_Medicine”:

|  |
| --- |
| -- ====================================== PROCEDURE 'proc\_Pay\_Medicine'  --================= Thanh toán đơn khám  /\*  Tạo hóa đơn Pay và Receipt  Chuyển trạng thái status của các Prescriptions của bệnh nhân,  chưa thanh toán thành 3 (đã thanh toán)  \*/  --=================  /\*  tham số đầu vào:  id bệnh nhân : VARCHAR(20)  id nhân viên thanh toán : VARCHAR(20)  id đơn khám  \*/  CREATE PROCEDURE [dbo].[proc\_Pay\_Medicine]  (  @patientID VARCHAR(20),  @employeeID VARCHAR(20),  @examinationID VARCHAR(20)  )  AS  BEGIN  --========== Tạo Receipts mới  EXEC proc\_InsertReceipt 'Medicine', @patientID, @employeeID    DECLARE @receiptID VARCHAR(20)  SELECT @receiptID = MAX(receiptID) FROM Receipts    --=========== Tạo Pay mới  DECLARE @totayPay INT  SELECT @totayPay = [dbo].[func\_getTotalPay\_ListMedicine\_withExamID](@examinationID)    EXEC proc\_InsertPay @patientID, @employeeID, @receiptID, @totayPay    --======== Cập nhật status của các Prescriptions  ---- mà chưa thanh toán thành 3 (đã thanh toán)  DECLARE @prescriptionID VARCHAR(20)    SELECT @prescriptionID = MIN(prescriptionID)  FROM [dbo].[func\_getListMedicion\_withExamID](@examinationID)    UPDATE Prescriptions  SET  status = 3  WHERE  prescriptionID = @prescriptionID  END |

## 4.4. Chức năng Admin

### 4.4.1. Chức năng trong form “frmDepartment”

#### 4.4.1.1. Hiển thị thông tin phòng ban

- View “View\_Department\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_Department\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(departmentID, 3) AS ConvertID FROM Departments;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllDepartment1”:

|  |
| --- |
| CREATE PROCEDURE spFindAllDepartment1  AS  BEGIN  SELECT departmentID, departmentName, createdAt, updatedAt FROM [View\_Department\_ConvertID] ORDER BY ConvertID ASC;  END  GO |

#### 4.4.1.2. Thêm phòng ban

- Stored procedure “spInsertDepartment”:

|  |
| --- |
| CREATE PROCEDURE spInsertDepartment(@departmentName NVARCHAR(255))  AS  BEGIN  INSERT INTO dbo.Departments (departmentName) VALUES (@departmentName)  END  GO |

- Trigger “trg\_Inserted\_Updated\_Department”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in Departments  CREATE TRIGGER trg\_Inserted\_Updated\_Department ON dbo.Departments  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @departmentID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @departmentID = Inserted.departmentID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.Departments SET createdAt = @createdAt WHERE departmentID = @departmentID  -- Cập nhật ID tăng tự động  SET @departmentID = dbo.func\_Auto\_departmentID()  UPDATE [Departments] SET departmentID = @departmentID WHERE departmentID = 'XXX0000'  END  -- Updated  ELSE  BEGIN  -- Cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.Departments SET updatedAt = @updatedAt WHERE departmentID = @departmentID  END  END  GO  -- End: trigger Inserted and Updated in Departments |

- Function “func\_Auto\_departmentID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_departmentID  CREATE FUNCTION func\_Auto\_departmentID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(3)  BEGIN  SET @object = 'Dep'  END  SELECT @max = COUNT(departmentID) FROM [Departments]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT departmentID FROM [Departments] WHERE departmentID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_departmentID |

#### 4.4.1.3. Chỉnh sửa thông tin phòng ban

- Stored procedure “spUpdateDepartment”:

|  |
| --- |
| CREATE PROCEDURE spUpdateDepartment(@departmentName NVARCHAR(255), @departmentID VARCHAR(20))  AS  BEGIN  UPDATE dbo.Departments SET departmentName = @departmentName WHERE departmentID = @departmentID  END  GO |

- Trigger “trg\_Inserted\_Updated\_Department”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in Departments  CREATE TRIGGER trg\_Inserted\_Updated\_Department ON dbo.Departments  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @departmentID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @departmentID = Inserted.departmentID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.Departments SET createdAt = @createdAt WHERE departmentID = @departmentID  -- Cập nhật ID tăng tự động  SET @departmentID = dbo.func\_Auto\_departmentID()  UPDATE [Departments] SET departmentID = @departmentID WHERE departmentID = 'XXX0000'  END  -- Updated  ELSE  BEGIN  -- Cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.Departments SET updatedAt = @updatedAt WHERE departmentID = @departmentID  END  END  GO  -- End: trigger Inserted and Updated in Departments |

- Function “func\_Auto\_departmentID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_departmentID  CREATE FUNCTION func\_Auto\_departmentID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(3)  BEGIN  SET @object = 'Dep'  END  SELECT @max = COUNT(departmentID) FROM [Departments]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT departmentID FROM [Departments] WHERE departmentID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_departmentID |

#### 4.4.1.4. Xóa thông tin phòng ban

- Stored procedure “spDeleteDepartment”:

|  |
| --- |
| CREATE PROCEDURE spDeleteDepartment(@departmentID VARCHAR(20))  AS  BEGIN  DELETE dbo.Departments WHERE departmentID = @departmentID  END  GO |

#### 4.4.1.5. Tìm kiếm tên theo ký tự tên có trong tên phòng ban

- View “View\_Department\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_Department\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(departmentID, 3) AS ConvertID FROM Departments;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllDepartment1BydepartmentNameLike”:

|  |
| --- |
| CREATE PROCEDURE spFindAllDepartment1BydepartmentNameLike (@departmentName NVARCHAR(255))  AS  BEGIN  SELECT departmentID, departmentName, createdAt, updatedAt FROM [View\_Department\_ConvertID] WHERE departmentName LIKE '%' + @departmentName + '%' ORDER BY ConvertID ASC;  END  GO |

### 4.4.2. Chức năng trong form “frmMedicine”

#### 4.4.2.1. Hiển thị thông tin thuốc

- View “View\_Medicine\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_Medicine\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(medicineID, 2) AS ConvertID FROM Medicines;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllMedicine1”:

|  |
| --- |
| CREATE PROCEDURE spFindAllMedicine1  AS  BEGIN  SELECT medicineID, medicineName, unit, medicinePrice, expiry, count, shape, medicineGroupName, createdAt, updatedAt FROM dbo.[View\_Medicine\_ConvertID] WHERE [status] = 1 ORDER BY ConvertID ASC;  END  GO |

#### 4.4.2.2. Thêm thông tin thuốc

- Stored procedure “spInsertMedicine”:

|  |
| --- |
| CREATE PROCEDURE spInsertMedicine(@medicineName NVARCHAR(255),  @unit NVARCHAR(255),  @medicinePrice INT,  @expiry INT,  @count INT,  @shape NVARCHAR(100),  @medicineGroupName NVARCHAR(255))  AS  BEGIN  INSERT INTO dbo.Medicines(medicineName, unit, medicinePrice, expiry, count, shape, medicineGroupName)  VALUES (@medicineName, @unit, @medicinePrice, @expiry, @count, @shape, @medicineGroupName)  END  GO |

- Trigger “trg\_Inserted\_Updated\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Inserted\_Updated\_Medicine ON dbo.Medicines  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @medicineID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @medicineID = Inserted.medicineID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.Medicines SET createdAt = @createdAt WHERE medicineID = @medicineID  SET @medicineID = dbo.func\_Auto\_medicineID();  UPDATE [Medicines] SET medicineID = @medicineID WHERE medicineID = 'XX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.Medicines SET updatedAt = @updatedAt WHERE medicineID = @medicineID  END  END  GO |

- Function “func\_Auto\_medicineID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_medicineID  CREATE FUNCTION func\_Auto\_medicineID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'MC'  END  SELECT @max = COUNT(medicineID) FROM [Medicines]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT medicineID FROM [Medicines] WHERE medicineID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_medicineID |

#### 4.4.2.3. Cập nhật thông tin thuốc

- Stored procedure “spUpdateMedicine”:

|  |
| --- |
| CREATE PROCEDURE spUpdateMedicine(@medicineName NVARCHAR(255),  @unit NVARCHAR(255),  @medicinePrice INT,  @expiry INT,  @count INT,  @shape NVARCHAR(100),  @medicineGroupName NVARCHAR(255),  @medicineID VARCHAR(20))  AS  BEGIN  UPDATE dbo.Medicines SET medicineName = @medicineID, unit = @unit, medicinePrice = @medicinePrice, expiry = @expiry, count = @count, shape = @shape, medicineGroupName = @medicineGroupName  WHERE medicineID = @medicineID;  END  GO |

- Trigger “trg\_Inserted\_Updated\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Inserted\_Updated\_Medicine ON dbo.Medicines  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @medicineID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @medicineID = Inserted.medicineID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.Medicines SET createdAt = @createdAt WHERE medicineID = @medicineID  SET @medicineID = dbo.func\_Auto\_medicineID();  UPDATE [Medicines] SET medicineID = @medicineID WHERE medicineID = 'XX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.Medicines SET updatedAt = @updatedAt WHERE medicineID = @medicineID  END  END  GO |

- Function “func\_Auto\_medicineID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_medicineID  CREATE FUNCTION func\_Auto\_medicineID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'MC'  END  SELECT @max = COUNT(medicineID) FROM [Medicines]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT medicineID FROM [Medicines] WHERE medicineID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_medicineID |

#### 4.4.2.4. Xóa thông tin thuốc khỏi người dùng

- Stored procedure “spDeleteMedicine1”:

|  |
| --- |
| CREATE PROCEDURE spDeleteMedicine1(@medicineID VARCHAR(20))  AS  BEGIN  UPDATE Medicines SET [status] = 0 WHERE medicineID = @medicineID;  END  GO |

- Trigger “ trg\_Deleted\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Deleted\_Medicine ON dbo.Medicines  FOR DELETE  AS  DECLARE @updatedAt DATETIME  DECLARE @medicineID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN    SELECT @medicineID = Deleted.medicineID, @updatedAt = Deleted.updatedAt  FROM Deleted  DECLARE @status TINYINT  SELECT @status = status FROM dbo.Medicines WHERE medicineID = @medicineID  IF (@status > 0)  BEGIN  RaisError ('Khong the > 10', 16, 1)  RollBack Tran  END  END  GO |

#### 4.4.1.5. Tìm kiếm tên theo ký tự tên có trong tên thuốc

- View “ spFindMedicineBymedicineNameLike”:

|  |
| --- |
| CREATE PROCEDURE spFindMedicineBymedicineNameLike (@medicineName NVARCHAR(255))  AS  BEGIN  SELECT medicineID, medicineName, unit, medicinePrice, expiry, count, shape, medicineGroupName, createdAt, updatedAt FROM dbo.[View\_Medicine\_ConvertID] WHERE medicineName LIKE '%' + @medicineName + '%' AND [status] = 1 ORDER BY ConvertID ASC;  END  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllMedicine1”:

|  |
| --- |
| CREATE PROCEDURE spFindAllMedicine1  AS  BEGIN  SELECT medicineID, medicineName, unit, medicinePrice, expiry, count, shape, medicineGroupName, createdAt, updatedAt FROM dbo.[View\_Medicine\_ConvertID] WHERE [status] = 1 ORDER BY ConvertID ASC;  END  GO |

### 4.4.3. Chức năng trong form “frmMedicineDel”

#### 4.4.3.1. Hiển thị thông tin thuốc đã xóa bởi người dùng

- View “View\_Medicine\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_Medicine\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(medicineID, 2) AS ConvertID FROM Medicines;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllMedicine0”:

|  |
| --- |
| CREATE PROCEDURE spFindAllMedicine0  AS  BEGIN  SELECT medicineID, medicineName, unit, medicinePrice, expiry, count, shape, medicineGroupName, createdAt, updatedAt FROM dbo.[View\_Medicine\_ConvertID] WHERE [status] = 0 ORDER BY ConvertID ASC;  END  GO |

#### 4.4.3.2. Khôi phục thông tin thuốc với người dùng

- Stored procedure “spRestoreMedicine”:

|  |
| --- |
| CREATE PROCEDURE spRestoreMedicine(@medicineID VARCHAR(20))  AS  BEGIN  UPDATE Medicines SET [status] = 1 WHERE medicineID = @medicineID;  END  GO |

- Trigger “trg\_Inserted\_Updated\_Medicine”:

|  |
| --- |
| CREATE TRIGGER trg\_Inserted\_Updated\_Medicine ON dbo.Medicines  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @medicineID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @medicineID = Inserted.medicineID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.Medicines SET createdAt = @createdAt WHERE medicineID = @medicineID  SET @medicineID = dbo.func\_Auto\_medicineID();  UPDATE [Medicines] SET medicineID = @medicineID WHERE medicineID = 'XX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.Medicines SET updatedAt = @updatedAt WHERE medicineID = @medicineID  END  END  GO |

- Function “func\_Auto\_medicineID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_medicineID  CREATE FUNCTION func\_Auto\_medicineID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'MC'  END  SELECT @max = COUNT(medicineID) FROM [Medicines]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT medicineID FROM [Medicines] WHERE medicineID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_medicineID |

### 4.4.4. Chức năng trong form “frmMedicineGroup”

#### 4.4.4.1. Hiển thị thông tin nhóm thuốc

- View “View\_MedicineGroup\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_MedicineGroup\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(medicineGroupID, 3) AS ConvertID FROM MedicineGroups;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “ spFindAllMedicineGroup”:

|  |
| --- |
| CREATE PROCEDURE spFindAllMedicineGroup  AS  BEGIN  SELECT medicineGroupID, medicineGroupName, createdAt, updatedAt FROM dbo.[View\_MedicineGroup\_ConvertID] ORDER BY ConvertID ASC;  END  GO |

#### 4.4.4.2. Thêm thông tin nhóm thuốc

- Stored procedure “spInsertMedicineGroup”:

|  |
| --- |
| CREATE PROCEDURE spInsertMedicineGroup(@medicineGroupName NVARCHAR(255))  AS  BEGIN  INSERT INTO dbo.MedicineGroups(medicineGroupName) VALUES (@medicineGroupName);  END  GO |

- Trigger “trg\_Inserted\_Updated\_MedicineGroup”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in MedicineGroups  CREATE TRIGGER trg\_Inserted\_Updated\_MedicineGroup ON dbo.MedicineGroups  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @medicineGroupID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @medicineGroupID = Inserted.medicineGroupID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.MedicineGroups SET createdAt = @createdAt WHERE medicineGroupID = @medicineGroupID  SET @medicineGroupID = dbo.func\_Auto\_medicineGroupID();  UPDATE [MedicineGroups] SET medicineGroupID = @medicineGroupID WHERE medicineGroupID = 'XXX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.MedicineGroups SET updatedAt = @updatedAt WHERE medicineGroupID = @medicineGroupID  END  END  GO  -- End: trigger Inserted and Updated in MedicineGroups |

- Function “func\_Auto\_medicineGroupID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_medicineGroupID  CREATE FUNCTION func\_Auto\_medicineGroupID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(3)  SET @object = 'MCG'  SELECT @max = COUNT(medicineGroupID) FROM [MedicineGroups]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT medicineGroupID FROM [MedicineGroups] WHERE medicineGroupID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_medicineGroupID |

#### 4.4.4.3. Cập nhật thông tin nhóm thuốc

- Stored procedure “spUpdateMedicineGroup”:

|  |
| --- |
| CREATE PROCEDURE spUpdateMedicineGroup(@medicineGroupID VARCHAR(20), @medicineGroupName NVARCHAR(255))  AS  BEGIN  UPDATE dbo.MedicineGroups SET medicineGroupName = @medicineGroupName WHERE medicineGroupID = @medicineGroupID;  END  GO |

- Trigger “trg\_Inserted\_Updated\_MedicineGroup”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in MedicineGroups  CREATE TRIGGER trg\_Inserted\_Updated\_MedicineGroup ON dbo.MedicineGroups  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @medicineGroupID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @medicineGroupID = Inserted.medicineGroupID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  SET @createdAt = GETDATE()  UPDATE dbo.MedicineGroups SET createdAt = @createdAt WHERE medicineGroupID = @medicineGroupID  SET @medicineGroupID = dbo.func\_Auto\_medicineGroupID();  UPDATE [MedicineGroups] SET medicineGroupID = @medicineGroupID WHERE medicineGroupID = 'XXX0000'  END  -- Updated  ELSE  BEGIN  SET @updatedAt = GETDATE()  UPDATE dbo.MedicineGroups SET updatedAt = @updatedAt WHERE medicineGroupID = @medicineGroupID  END  END  GO  -- End: trigger Inserted and Updated in MedicineGroups |

- Function “func\_Auto\_medicineGroupID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_medicineGroupID  CREATE FUNCTION func\_Auto\_medicineGroupID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(3)  SET @object = 'MCG'  SELECT @max = COUNT(medicineGroupID) FROM [MedicineGroups]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT medicineGroupID FROM [MedicineGroups] WHERE medicineGroupID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_medicineGroupID |

#### 4.4.4.4. Xóa thông tin đơn thuốc

- Stored procedure “ spDeleteMedicineGroup”:

|  |
| --- |
| CREATE PROCEDURE spDeleteMedicineGroup(@medicineGroupID VARCHAR(20))  AS  BEGIN  DELETE dbo.MedicineGroups WHERE medicineGroupID = @medicineGroupID;  END  GO |

### 4.4.5. Chức năng trong form “frmService”

#### 4.4.5.1. Hiển thị thông tin dịch vụ

- View “View\_Service\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_Service\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(serviceID, 2) AS ConvertID FROM Services;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllService”:

|  |
| --- |
| CREATE PROCEDURE spFindAllService  AS  BEGIN  SELECT serviceID, serviceName, servicePrice, createdAt, updatedAt FROM dbo.[View\_Service\_ConvertID] ORDER BY ConvertID ASC  END  GO |

#### 4.4.5.2. Thêm thông tin dịch vụ

- Stored procedure “spInsertService”:

|  |
| --- |
| CREATE PROCEDURE spInsertService(@serviceName NVARCHAR(255), @servicePrice INT)  AS  BEGIN  INSERT INTO dbo.Services (serviceName, servicePrice) VALUES (@serviceName, @servicePrice)  END  GO |

- Trigger “ trg\_Inserted\_Updated\_Service”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in Services  CREATE TRIGGER trg\_Inserted\_Updated\_Service ON dbo.Services  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @serviceID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @serviceID = Inserted.serviceID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động tạo ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.Services SET createdAt = @createdAt WHERE serviceID = @serviceID  -- Tự động tạo ID  SET @serviceID = dbo.func\_Auto\_serviceID()  UPDATE [Services] SET serviceID = @serviceID WHERE serviceID = 'XX0000'  END  -- Updated  ELSE  BEGIN  -- Tự động tạo ngày cập nhật  SET @updatedAt = GETDATE()  UPDATE dbo.Services SET updatedAt = @updatedAt WHERE serviceID = @serviceID  END  END  GO  -- End: trigger Inserted and Updated in Services |

- Function “func\_Auto\_serviceID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_serviceID  CREATE FUNCTION func\_Auto\_serviceID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'SV'  END  SELECT @max = COUNT(serviceID) FROM [Services]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT serviceID FROM [Services] WHERE serviceID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_serviceID |

#### 4.4.5.3. Cập nhật thông tin dịch vụ

- Stored procedure “spUpdateService”:

|  |
| --- |
| CREATE PROCEDURE spUpdateService(@serviceName NVARCHAR(255), @servicePrice INT, @serviceID VARCHAR(20))  AS  BEGIN  UPDATE dbo.Services SET serviceName = @serviceName, servicePrice = @servicePrice WHERE serviceID = @serviceID  END  GO |

- Trigger “ trg\_Inserted\_Updated\_Service”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in Services  CREATE TRIGGER trg\_Inserted\_Updated\_Service ON dbo.Services  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @serviceID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @serviceID = Inserted.serviceID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động tạo ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.Services SET createdAt = @createdAt WHERE serviceID = @serviceID  -- Tự động tạo ID  SET @serviceID = dbo.func\_Auto\_serviceID()  UPDATE [Services] SET serviceID = @serviceID WHERE serviceID = 'XX0000'  END  -- Updated  ELSE  BEGIN  -- Tự động tạo ngày cập nhật  SET @updatedAt = GETDATE()  UPDATE dbo.Services SET updatedAt = @updatedAt WHERE serviceID = @serviceID  END  END  GO  -- End: trigger Inserted and Updated in Services |

- Function “func\_Auto\_serviceID”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_serviceID  CREATE FUNCTION func\_Auto\_serviceID()  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  BEGIN  SET @object = 'SV'  END  SELECT @max = COUNT(serviceID) FROM [Services]  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT serviceID FROM [Services] WHERE serviceID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_serviceID |

#### 4.4.5.4. Xóa thông tin dịch vụ

- Stored procedure “spDeleteService”:

|  |
| --- |
| CREATE PROCEDURE spDeleteService(@serviceID VARCHAR(20))  AS  BEGIN  DELETE dbo.Services WHERE serviceID = @serviceID  END  GO |

#### 4.4.5.5. Tìm kiếm tên theo ký tự tên có trong tên thuốc

- View “View\_Service\_ConvertID”:

|  |
| --- |
| CREATE VIEW [View\_Service\_ConvertID]  AS  SELECT \*, dbo.func\_ConvertID(serviceID, 2) AS ConvertID FROM Services;  GO |

- Function “ func\_ConvertID”:

|  |
| --- |
| CREATE FUNCTION func\_ConvertID(@ID VARCHAR(20), @lenObject INT)  RETURNS INT  AS  BEGIN  DECLARE @Res INT  SELECT @ID = RIGHT(@ID, LEN(@ID) - @lenObject)  SET @Res = CONVERT(INT, @ID)  RETURN @Res  END  GO |

- Stored procedure “spFindAllServiceByServiceNameLike”:

|  |
| --- |
| CREATE PROCEDURE spFindAllServiceByServiceNameLike (@serviceName NVARCHAR(255))  AS  BEGIN  SELECT serviceID, serviceName, servicePrice, createdAt, updatedAt FROM dbo.[View\_Service\_ConvertID] WHERE serviceName LIKE '%' + @serviceName + '%' ORDER BY ConvertID ASC  END  GO |

### 4.4.6. Chức năng trong form FrmDoctors

#### 4.4.6.1. Lấy thông tin danh sách bác sĩ

- Code trong C#:

|  |
| --- |
| public DataTable getBacSi()  {  var bss = from bs in bacsi.View\_Doctor select bs;  DataTable dt = new DataTable();  dt.Columns.Add("Mã Bác sĩ");  dt.Columns.Add("Họ");  dt.Columns.Add("Tên");  dt.Columns.Add("Giới tính");  dt.Columns.Add("Ngày sinh");  dt.Columns.Add("Địa chỉ");  dt.Columns.Add("Số điện thoại");  dt.Columns.Add("CMND/CCCD");  dt.Columns.Add("Khoa");  dt.Columns.Add("Ngày tạo");  dt.Columns.Add("Ngày chỉnh sửa");  foreach (var i in bss)  {  dt.Rows.Add(i.doctorID, i.doctorLastName, i.doctorFirstname, i.sex, i.birthDay, i.address, i.phone, i.cardID, i.departmentName, i.createdAt, i.updatedAt);  }  return dt;  } |

- View “View\_Doctor”:

|  |
| --- |
| CREATE VIEW [View\_Doctor]  AS  SELECT [P].[peopleID] as [doctorID]  , [P].[firstName] as [doctorFirstname]  , [P].[lastName] as [doctorLastName]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [D].[departmentName]  , [P].[createdAt]  , [P].[updatedAt]  FROM([dbo].[People] as [P]  INNER JOIN [dbo].[Employees] as [E]  ON [P].[peopleID] = [E].[employeeID])  JOIN [dbo].[Departments] as [D]  ON ([E].[departmentID] = [D].[departmentID])  WHERE [E].[position] = N'Bác sĩ';  GO |

#### 4.4.6.2. Thêm thông tin bệnh nhân

- Stored procedure “InsertDoctor”:

|  |
| --- |
| --Begin: PROCEDURE InsertDoctor  CREATE PROCEDURE InsertDoctor (@employeeFirstName nvarchar(32),  @employeeLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @departmentID VARCHAR(25))  AS  BEGIN  BEGIN TRANSACTION Tran\_InsertDoctor  BEGIN TRY  DECLARE @role TINYINT  SET @role = 2  INSERT INTO dbo.People (firstName, lastName, sex, birthDay, address, phone, cardID, role)  VALUES (@employeeFirstName, @employeeLastname, @sex, @birthDay, @address, @phone, @cardID, @role)  DECLARE @employeeID VARCHAR(20)  SELECT @employeeID = MAX(peopleID) FROM dbo.People WHERE role = @role  DECLARE @position NVARCHAR(25)  SET @position = N'Bác sĩ'  UPDATE dbo.Employees SET position = @position, departmentID = @departmentID WHERE employeeID = @employeeID  COMMIT TRANSACTION Tran\_InsertDoctor  END TRY  BEGIN CATCH  PRINT('Thêm không thành công')  ROLLBACK TRANSACTION Tran\_InsertDoctor  END CATCH  END  GO  --End: PROCEDURE InsertDoctor |

- Trigger “trg\_Inserted\_Updated\_People”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER trg\_Inserted\_Updated\_People ON dbo.People  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID  -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'  -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END  GO  -- End: trigger Inserted and Updated in People |

- Function “func\_Auto\_PeopleId”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_PeopleId  CREATE FUNCTION func\_Auto\_PeopleId(@role TINYINT)  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  IF @role = 0  BEGIN  SET @object = 'TN'  END  ELSE  BEGIN  IF @role = 1  BEGIN  SET @object = 'BN'  END  ELSE  BEGIN  SET @object = 'NV'  END  END  SELECT @max = COUNT(role) FROM dbo.People WHERE role = @role  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT peopleID FROM dbo.People WHERE peopleID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_PeopleId |

#### 4.4.6.3. Cập nhật thông tin bệnh nhân

- Stored procedure “UpdateDoctor\_Staff ”:

|  |
| --- |
| --Begin: PROCEDURE UpdateDoctor\_Staff  CREATE PROCEDURE UpdateDoctor\_Staff (@employeeFirstName nvarchar(32),  @employeeLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @departmentID VARCHAR(25),  @peopleID VARCHAR(20))  AS  BEGIN  BEGIN TRANSACTION Tran\_UpdateDoctor\_Staff  BEGIN TRY  UPDATE dbo.People SET firstName = @employeeFirstName,  lastName = @employeeLastname,  sex = @sex,  birthDay = @birthDay,  address = @address,  phone = @phone,  cardID = @cardID  WHERE peopleID = @peopleID  UPDATE dbo.Employees SET departmentID = @departmentID WHERE employeeID = @peopleID  COMMIT TRANSACTION Tran\_UpdateDoctor\_Staff  END TRY  BEGIN CATCH  PRINT('Thêm không thành công')  ROLLBACK TRANSACTION Tran\_UpdateDoctor\_Staff  END CATCH  END  GO  --End: PROCEDURE UpdateDoctor\_Staff |

- Trigger “trg\_Inserted\_Updated\_People”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER trg\_Inserted\_Updated\_People ON dbo.People  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID  -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'  -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END  GO  -- End: trigger Inserted and Updated in People |

- Function “func\_Auto\_PeopleId”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_PeopleId  CREATE FUNCTION func\_Auto\_PeopleId(@role TINYINT)  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  IF @role = 0  BEGIN  SET @object = 'TN'  END  ELSE  BEGIN  IF @role = 1  BEGIN  SET @object = 'BN'  END  ELSE  BEGIN  SET @object = 'NV'  END  END  SELECT @max = COUNT(role) FROM dbo.People WHERE role = @role  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT peopleID FROM dbo.People WHERE peopleID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_PeopleId |

### 4.4.7. Chức năng trong form “FrmTatCaNhanVien”

#### 4.4.7.1. Lấy thông tin nhân viên

- Code trong C#:

|  |
| --- |
| public DataTable getNhanvien()  {  var nvs = from nv in nhanvien.View\_Staff select nv;  DataTable dt = new DataTable();  dt.Columns.Add("Mã nhân viên");  dt.Columns.Add("Last Name");  dt.Columns.Add("First Name");  dt.Columns.Add("Giới tính");  dt.Columns.Add("Ngày sinh");  dt.Columns.Add("Địa chỉ");  dt.Columns.Add("Số điện thoại");  dt.Columns.Add("CMND/CCCD");  dt.Columns.Add("Khoa");  dt.Columns.Add("ID khoa");  dt.Columns.Add("Ngày tạo");  dt.Columns.Add("Ngày cập nhật");  foreach (var i in nvs)  {  dt.Rows.Add(i.doctorID, i.doctorLastName, i.doctorFirstname, i.sex, i.birthDay,  i.address, i.phone, i.cardID, i.departmentName, i.departmentID, i.createdAt, i.updatedAt);  }  return dt;  } |

- View “View\_Staff”:

|  |
| --- |
| CREATE VIEW [View\_Staff]  AS  SELECT [P].[peopleID] as [doctorID]  , [P].[firstName] as [doctorFirstname]  , [P].[lastName] as [doctorLastName]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [D].[departmentName]  , [D].[departmentID]  , [P].[createdAt]  , [P].[updatedAt]  FROM([dbo].[People] as [P]  INNER JOIN [dbo].[Employees] as [E]  ON [P].[peopleID] = [E].[employeeID])  JOIN [dbo].[Departments] as [D]  ON ([E].[departmentID] = [D].[departmentID])  WHERE [E].[position] = N'Nhân viên';  GO |

#### 4.4.7.2. Thêm thông tin nhân viên

- Stored procedure “InsertStaff”:

|  |
| --- |
| --Begin: PROCEDURE InsertStaff  CREATE PROCEDURE InsertStaff (@employeeFirstName nvarchar(32),  @employeeLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @departmentID VARCHAR(25))  AS  BEGIN  BEGIN TRANSACTION Tran\_InsertStaff  BEGIN TRY  DECLARE @role TINYINT  SET @role = 2  INSERT INTO dbo.People (firstName, lastName, sex, birthDay, address, phone, cardID, role)  VALUES (@employeeFirstName, @employeeLastname, @sex, @birthDay, @address, @phone, @cardID, @role)  DECLARE @employeeID VARCHAR(20)  SELECT @employeeID = MAX(peopleID) FROM dbo.People WHERE role = @role  DECLARE @position NVARCHAR(25)  SET @position = N'Nhân viên'  UPDATE dbo.Employees SET position = @position, departmentID = @departmentID WHERE employeeID = @employeeID  COMMIT TRANSACTION Tran\_InsertDoctor  END TRY  BEGIN CATCH  PRINT('Thêm không thành công')  ROLLBACK TRANSACTION Tran\_InsertStaff  END CATCH  END  GO  --End: PROCEDURE InsertStaff |

- Trigger “trg\_Inserted\_Updated\_People”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER trg\_Inserted\_Updated\_People ON dbo.People  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID  -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'  -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END  GO  -- End: trigger Inserted and Updated in People |

- Function “func\_Auto\_PeopleId”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_PeopleId  CREATE FUNCTION func\_Auto\_PeopleId(@role TINYINT)  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  IF @role = 0  BEGIN  SET @object = 'TN'  END  ELSE  BEGIN  IF @role = 1  BEGIN  SET @object = 'BN'  END  ELSE  BEGIN  SET @object = 'NV'  END  END  SELECT @max = COUNT(role) FROM dbo.People WHERE role = @role  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT peopleID FROM dbo.People WHERE peopleID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_PeopleId |

#### 4.4.7.3. Cập nhật thông tin nhân viên

- Stored procedure “UpdateDoctor\_Staff ”:

|  |
| --- |
| --Begin: PROCEDURE UpdateDoctor\_Staff  CREATE PROCEDURE UpdateDoctor\_Staff (@employeeFirstName nvarchar(32),  @employeeLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @departmentID VARCHAR(25),  @peopleID VARCHAR(20))  AS  BEGIN  BEGIN TRANSACTION Tran\_UpdateDoctor\_Staff  BEGIN TRY  UPDATE dbo.People SET firstName = @employeeFirstName,  lastName = @employeeLastname,  sex = @sex,  birthDay = @birthDay,  address = @address,  phone = @phone,  cardID = @cardID  WHERE peopleID = @peopleID  UPDATE dbo.Employees SET departmentID = @departmentID WHERE employeeID = @peopleID  COMMIT TRANSACTION Tran\_UpdateDoctor\_Staff  END TRY  BEGIN CATCH  PRINT('Thêm không thành công')  ROLLBACK TRANSACTION Tran\_UpdateDoctor\_Staff  END CATCH  END  GO  --End: PROCEDURE UpdateDoctor\_Staff |

- Trigger “trg\_Inserted\_Updated\_People”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER trg\_Inserted\_Updated\_People ON dbo.People  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID  -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'  -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END  GO  -- End: trigger Inserted and Updated in People |

- Function “func\_Auto\_PeopleId”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_PeopleId  CREATE FUNCTION func\_Auto\_PeopleId(@role TINYINT)  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  IF @role = 0  BEGIN  SET @object = 'TN'  END  ELSE  BEGIN  IF @role = 1  BEGIN  SET @object = 'BN'  END  ELSE  BEGIN  SET @object = 'NV'  END  END  SELECT @max = COUNT(role) FROM dbo.People WHERE role = @role  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT peopleID FROM dbo.People WHERE peopleID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_PeopleId |

### 4.4.8. Chức năng trong form “FrmPatients”

#### 4.4.8.1. Lấy thông tin bệnh nhân chờ

Code trong C#:

|  |
| --- |
| public List<View\_Patients\_Wait> getThongTinBenhNhan()  {  List<View\_Patients\_Wait> info\_BenhNhan = new List<View\_Patients\_Wait>();  //kết nố db  using (QLHSBAEntities db = new QLHSBAEntities())  {  //Lấy ra tất cả các column trong bảng Info\_patient  var tb\_infoBN = from bn in db.View\_Patients\_Wait  select bn;  foreach (var i in tb\_infoBN)  {  View\_Patients\_Wait bn = new View\_Patients\_Wait();  bn.patientID = i.patientID;  bn.patientFirstName = i.patientFirstName;  bn.patientLastname = i.patientLastname;  bn.sex = i.sex;  bn.birthDay = i.birthDay;  bn.address = i.address;  bn.phone = i.phone;  bn.cardID = i.cardID;  bn.patientJob = i.patientJob;  bn.healthInsurance = i.healthInsurance;  bn.reason = i.reason;  bn.createdAt = i.createdAt;  bn.updatedAt = i.updatedAt;  info\_BenhNhan.Add(bn);  }  }  return info\_BenhNhan;  } |

- View “View\_Patients\_Wait”:

|  |
| --- |
| CREATE VIEW [View\_Patients\_Wait]  AS  SELECT [EX].[examinateID]  , [P].[peopleID] as [patientID]  , [P].[firstName] as [patientFirstName]  , [P].[lastName] as [patientLastname]  , [P].[sex]  , [P].[birthDay]  , [P].[address]  , [P].[phone]  , [P].[cardID]  , [P2].[patientJob]  , [P2].[healthInsurance]  , [P2].[reason]  , [P].[createdAt]  , [P].[updatedAt]  FROM ([dbo].[People] as [P]  INNER JOIN [dbo].[Patients] as [P2]  ON ([P].[peopleID] = [P2].[patientID]))  INNER JOIN [dbo].[Examination] as [EX]  ON ([P2].[patientID] = [EX].[patientID])  WHERE [EX].[status] = 1;  GO |

#### 4.4.8.2. Thêm thông tin bệnh nhân

- Stored procedure “InsertPatient”:

|  |
| --- |
| --Begin: PROCEDURE InsertPatient  CREATE PROCEDURE InsertPatient (@patientFirstName nvarchar(32),  @patientLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @patientJob nvarchar(255),  @healthInsurance varchar(20),  @reason nvarchar(500))  AS  BEGIN  BEGIN TRANSACTION Tran\_InsertPatient  BEGIN TRY  DECLARE @role TINYINT  SET @role = 1  INSERT INTO dbo.People (firstName, lastName, sex, birthDay, address, phone, cardID, role)  VALUES (@patientFirstName, @patientLastname, @sex, @birthDay, @address, @phone, @cardID, @role)  DECLARE @patientID VARCHAR(20)  SELECT @patientID = MAX(peopleID) FROM dbo.People WHERE role = @role    UPDATE dbo.Patients SET patientJob = @patientJob, healthInsurance = @healthInsurance, reason = @reason  WHERE patientID = @patientID  COMMIT TRANSACTION Tran\_InsertPatient  END TRY  BEGIN CATCH  PRINT('Thêm không thành công!')  COMMIT TRANSACTION Tran\_InsertPatient  END CATCH  END  GO  --End: PROCEDURE InsertPatient |

- Trigger “trg\_Inserted\_Updated\_People”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER trg\_Inserted\_Updated\_People ON dbo.People  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID  -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'  -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END  GO  -- End: trigger Inserted and Updated in People |

- Function “func\_Auto\_PeopleId”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_PeopleId  CREATE FUNCTION func\_Auto\_PeopleId(@role TINYINT)  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  IF @role = 0  BEGIN  SET @object = 'TN'  END  ELSE  BEGIN  IF @role = 1  BEGIN  SET @object = 'BN'  END  ELSE  BEGIN  SET @object = 'NV'  END  END  SELECT @max = COUNT(role) FROM dbo.People WHERE role = @role  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT peopleID FROM dbo.People WHERE peopleID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_PeopleId |

#### 4.4.7.3. Cập nhật thông tin bệnh nhân

- Stored procedure “ UpdatePatient ”:

|  |
| --- |
| --Begin: PROCEDURE UpdatePatient  CREATE PROCEDURE UpdatePatient (@patientFirstName nvarchar(32),  @patientLastname nvarchar(32),  @sex char(1),  @birthDay DATE,  @address nvarchar(510),  @phone varchar(15),  @cardID varchar(15),  @patientJob nvarchar(255),  @healthInsurance varchar(20),  @reason nvarchar(500),  @peopleID VARCHAR(20))  AS  BEGIN  BEGIN TRANSACTION Tran\_UpdatePatient  BEGIN TRY  UPDATE dbo.People SET firstName = @patientFirstName,  lastName = @patientLastname,  sex = @sex,  birthDay = @birthDay,  address = @address,  phone = @phone,  cardID = @cardID  WHERE peopleID = @peopleID  UPDATE dbo.Patients SET patientJob = @patientJob,  healthInsurance = @healthInsurance,  reason = @reason  WHERE patientID = @peopleID  COMMIT TRANSACTION Tran\_UpdatePatient  END TRY  BEGIN CATCH  PRINT('Cập nhật không thành công')  ROLLBACK TRANSACTION Tran\_UpdatePatient  END CATCH  END  GO  --End: PROCEDURE UpdatePatient |

- Trigger “trg\_Inserted\_Updated\_People”:

|  |
| --- |
| -- Begin: trigger Inserted and Updated in People  CREATE TRIGGER trg\_Inserted\_Updated\_People ON dbo.People  FOR INSERT, UPDATE  AS  DECLARE @createdAt DATETIME  DECLARE @updatedAt DATETIME  DECLARE @role TINYINT  DECLARE @peopleID VARCHAR(20)  BEGIN  IF TRIGGER\_NESTLEVEL() > 1  RETURN  SELECT @peopleID = Inserted.peopleID, @createdAt = Inserted.createdAt, @updatedAt = Inserted.updatedAt  FROM Inserted  -- Inserted  IF (@createdAt IS NULL)  BEGIN  -- Tự động cập nhật ngày tạo  SET @createdAt = GETDATE()  UPDATE dbo.People SET createdAt = @createdAt WHERE peopleID = @peopleID  -- Cập nhật Id tăng tự động  SELECT @role = Inserted.role FROM Inserted  SET @peopleId = dbo.func\_Auto\_PeopleId(@role)  UPDATE dbo.People SET peopleID = @peopleId WHERE peopleID = 'XX0000'  -- Kế thừa id qua Patient  IF (@role = 1)  BEGIN  INSERT INTO dbo.Patients (patientID)  VALUES (@peopleId)  END  -- Kế thừa id qua Employees  ELSE  BEGIN  IF (@role = 2)  BEGIN  INSERT INTO dbo.Employees (employeeID)  VALUES (@peopleId)  END  END  END  -- Updated  ELSE  BEGIN  -- Tự động cập nhật ngày update  SET @updatedAt = GETDATE()  UPDATE dbo.People SET updatedAt = @updatedAt WHERE peopleID = @peopleID  END  END  GO  -- End: trigger Inserted and Updated in People |

- Function “func\_Auto\_PeopleId”:

|  |
| --- |
| -- BEGIN: FUNCTION func\_Auto\_PeopleId  CREATE FUNCTION func\_Auto\_PeopleId(@role TINYINT)  RETURNS VARCHAR(20)  AS  BEGIN  DECLARE @id\_next VARCHAR(20)  DECLARE @max INT  DECLARE @object VARCHAR(2)  IF @role = 0  BEGIN  SET @object = 'TN'  END  ELSE  BEGIN  IF @role = 1  BEGIN  SET @object = 'BN'  END  ELSE  BEGIN  SET @object = 'NV'  END  END  SELECT @max = COUNT(role) FROM dbo.People WHERE role = @role  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  -- Kiểm tra id đã tồn tại chưa  WHILE(EXISTS(SELECT peopleID FROM dbo.People WHERE peopleID = @id\_next))  BEGIN  SET @max = @max + 1  SET @id\_next = @object + RIGHT('0' + CAST(@max AS VARCHAR(17)), 18)  END  RETURN @id\_next  END  GO  -- END: FUNCTION func\_Auto\_PeopleId |

# V. Tạo Role, User và Phân quyền

## 5.1. Tạo Role :

|  |
| --- |
| ------TẠO ROLE--------  create role admin  create role doctor  create role employee |

## 5.2. Phân quyền cho các Role:

### 5.2.1. Admin:

|  |
| --- |
| ------ADMIN CÓ TOÀN BỘ QUYỀN TRÊN DBS---------  GRANT ALTER, VIEW DEFINITION, EXECUTE TO admin |

### 5.2.2. Doctor:

#### 5.2.2.1. Phân quyền mức View:

|  |
| --- |
| --------PHÂN QUYỀN MỨC VIEW CHO DOCTOR------------------------  GRANT SELECT on dbo.Accounts to doctor  GRANT SELECT on dbo.Departments to doctor  GRANT SELECT on dbo.Employees to doctor  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Examination to doctor  GRANT SELECT on dbo.MedicineGroups to doctor  GRANT SELECT on dbo.Medicines to doctor  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Patients to doctor  GRANT SELECT on dbo.Pay to doctor  GRANT SELECT on dbo.Prescription\_Medicine to doctor  GRANT SELECT on dbo.Prescriptions to doctor  GRANT SELECT on dbo.Receipts to doctor  GRANT SELECT on dbo.Relatives to doctor  GRANT SELECT on dbo.Roles to doctor  GRANT SELECT on dbo.Services to doctor  GRANT SELECT on dbo.UsingService to doctor |

#### 5.2.2.2. Phân quyền trên View:

|  |
| --- |
| --Phân quyền trên view  GRANT SELECT on View\_CreatedPrescription\_New to doctor  GRANT SELECT on View\_Doctor to doctor  GRANT SELECT on View\_Patients to doctor  GRANT ALTER, SELECT on View\_Patients\_Wait to doctor  GRANT ALTER, SELECT on View\_Staff to doctor  GRANT ALTER, SELECT on View\_Examination to doctor  GRANT SELECT on View\_Department\_ConvertID to doctor  GRANT ALTER, SELECT on View\_ListMedicine\_Precesciption to doctor  GRANT ALTER, SELECT on View\_Medicine\_ConvertID to doctor  GRANT ALTER, SELECT on View\_MedicineGroup\_ConvertID to doctor  GRANT ALTER, SELECT on View\_Service\_ConvertID to doctor  GRANT ALTER, SELECT on View\_UsingService to doctor  GRANT SELECT on View\_Wait\_DrugSupply to doctor |

#### 5.2.2.3. Phân quyền trên Procedure:

|  |
| --- |
| --Phân quyền trên procedure  GRANT EXECUTE ON proc\_InsertUsingService TO doctor  GRANT EXECUTE ON proc\_UpdateUsingService TO doctor  GRANT EXECUTE ON spDeletePrescription\_Medicine TO doctor  GRANT EXECUTE ON spInsertPrescription\_Medicine TO doctor  GRANT EXECUTE ON spUpdateExamination TO doctor  GRANT EXECUTE ON UpdatePatient TO doctor  GRANT EXECUTE ON proc\_createNewNullExamination TO doctor |

### 5.2.3. Employee:

#### 5.2.3.1. Phân quyền mức View:

|  |
| --- |
| ------------------PHÂN QUYỀN MỨC VIEW CHO EMPLOYEE------------------------  GRANT SELECT on dbo.Accounts to employee  GRANT SELECT on dbo.Departments to employee  GRANT SELECT on dbo.Employees to employee  GRANT SELECT on dbo.Examination to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.MedicineGroups to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Medicines to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Patients to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Pay to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Prescription\_Medicine to employee  GRANT SELECT on dbo.Prescriptions to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Receipts to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.Relatives to employee  GRANT SELECT on dbo.Roles to employee  GRANT SELECT on dbo.Services to employee  GRANT SELECT, UPDATE, INSERT, DELETE on dbo.UsingService to employee |

#### 5.2.3.2. Phân quyền trên View:

|  |
| --- |
| -- Phân quyền cho Role Employee trên view  GRANT SELECT on View\_CreatedPrescription\_New to employee  GRANT SELECT on View\_Doctor to employee  GRANT SELECT on View\_Patients to employee  GRANT ALTER, SELECT on View\_Patients\_Wait to employee  GRANT ALTER, SELECT on View\_Staff to employee  GRANT ALTER, SELECT on View\_Examination to employee  GRANT SELECT on View\_Department\_ConvertID to doctor  GRANT ALTER, SELECT on View\_ListMedicine\_Precesciption to employee  GRANT ALTER, SELECT on View\_Medicine\_ConvertID to employee  GRANT ALTER, SELECT on View\_MedicineGroup\_ConvertID to employee  GRANT ALTER, SELECT on View\_Service\_ConvertID to employee  GRANT ALTER, SELECT on View\_UsingService to employee  GRANT SELECT on View\_Wait\_DrugSupply to employee |

#### 5.2.3.3. Phân quyền trên Procedure:

|  |
| --- |
| --Phân quyền role Employee trên procedure  GRANT EXECUTE on FindAllDoctor to employee  GRANT EXECUTE on FindAllStaff to employee  GRANT EXECUTE on FindDepartmentNameByDoctorId to employee  GRANT EXECUTE on FindFullNameDoctorByDoctorId to employee  GRANT EXECUTE on InsertPatient to employee  GRANT EXECUTE on InsertRelative to employee  GRANT EXECUTE on InsertRelative\_Exist to employee  GRANT EXECUTE on proc\_InsertPay to employee  GRANT EXECUTE on proc\_InsertReceipt to employee  GRANT EXECUTE on proc\_InsertUsingService to employee  GRANT EXECUTE on proc\_Pay\_Medicine to employee  GRANT EXECUTE on proc\_Pay\_service to employee  GRANT EXECUTE on proc\_UpdateUsingService to employee  GRANT EXECUTE on spDeletePrescription\_Medicine to employee  GRANT EXECUTE on spFindAllDepartment1 to employee  GRANT EXECUTE on spFindAllDepartment1ByDepartmentNameLike to employee  GRANT EXECUTE on spFindAllMedicine0 to employee  GRANT EXECUTE on spFindAllMedicine1byMedicineName to employee  GRANT EXECUTE on spFindAllMedicineGroup to employee  GRANT EXECUTE on spFindAllPrescription\_Medicine to employee  GRANT EXECUTE on spFindAllService to employee  GRANT EXECUTE on spFindAllServiceByServiceNameLike to employee  GRANT EXECUTE on spFindMedicineBymedicineNameLike to employee  GRANT EXECUTE on spRestoreMedicine to employee  GRANT EXECUTE on spWaitDrugSupply to employee  GRANT EXECUTE on UpdatePatient to employee  GRANT EXECUTE on UpdateRelative to employee |

## 5.3. Tạo User, cấp quyền truy cập (SQL):

Chức năng tạo User và phân quyền truy cập trên SQL sẽ được gọi chung với chức năng tạo User ở phía Application. Tương tự cho chức năng Xóa User(SQL) và xóa User(Application).

|  |
| --- |
| ------------------PROCEDURE THÊM ACCOUNT (VÀO SQL SERVER)-----------  create procedure addAccountServer2 @username varchar(20), @pass varchar(100), @roleid int  as begin  declare @login nvarchar(4000)  set @login = N'CREATE LOGIN ' + QUOTENAME(@username) + ' WITH PASSWORD = ' +  QUOTENAME(@pass, '''') + ', default\_database = ' + QUOTENAME('QLHSBA')  exec(@login)  declare @user nvarchar(4000)  set @user = N'CREATE USER ' + QUOTENAME(@username) + ' FOR LOGIN ' +  QUOTENAME(@username)  exec(@user)  begin  if(@roleid = 1) --admin  begin  exec sp\_addrolemember 'admin', @username  end  if(@roleid = 2) --doctor  begin  exec sp\_addrolemember 'doctor', @username  end  if(@roleid = 3) --employee  begin  exec sp\_addrolemember 'employee', @username  end  end  end  go |

# VI. Thiết kế giao diện:

## 6.1. Form Đăng nhập:



Hình 6.1.1. Form đăng nhập

* Khi khởi động form frmLogin sẽ được truy cập đầu tiên. Người dùng nhập thông tin đăng nhập: username và password lần lượt vào hai textbox. Button Đăng nhập truyền thông tin được nhập vào đến các hàm bên dưới để xử lý.

## 6.2. Các form trong role Admin:

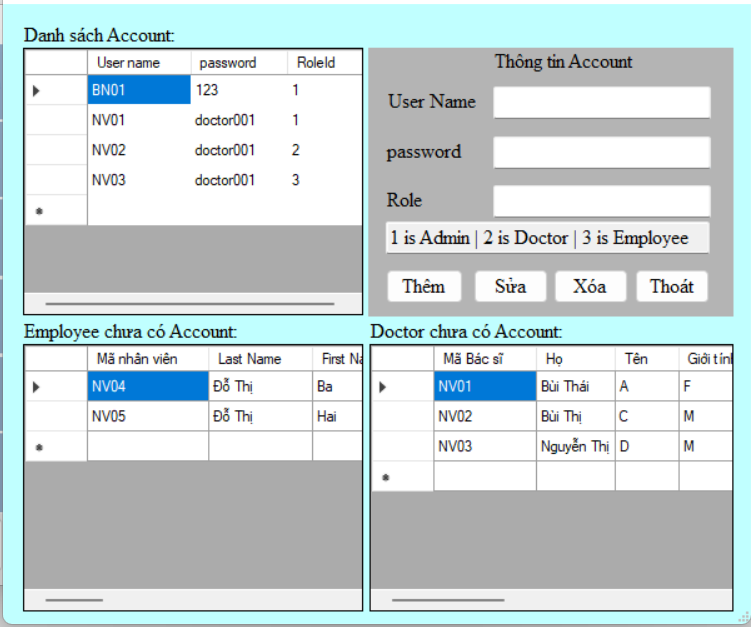
### 6.2.1. Form trang chủ Admin (Admin\_Home):



Hình 6.2.1.1. Form trang chủ Admin (Admin\_Home)

* Sau khi button Đăng nhập truyền thông tin, các hàm sẽ xử lý xem thông tin đăng nhập (username và password) có chính xác không, nếu chính xác thì sẽ kiểm tra role của account đó, role bằng 1 sẽ chuyển đến form Admin\_Home. Form này bao gồm tất cả những button chuyển đến các bảng mà role Admin quản lý. Button thoát sẽ đăng xuất khỏi phần mềm quản lý.

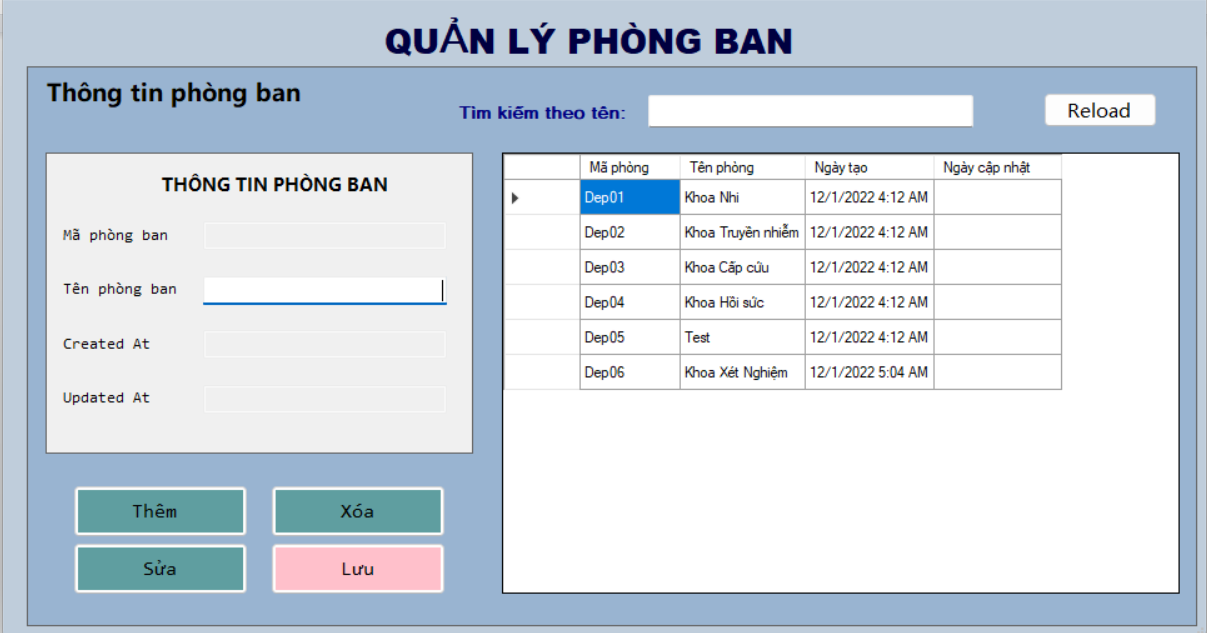
### 6.2.2. Form quản lý Accounts (FrmAccount):



Hình 6.2.2.1. Form quản lý Accounts (FrmAccount)

* Sau khi chọn button Quản lý tài khoản, sẽ được chuyển sang form quản lý Accounts và thông tin của các Account, cũng như những Employee, Doctor chưa có account sẽ được hiển thị trong DatagridView. Cùng với đó là các button để xử lý Procedure được viết trong SQL như:
* Button Thêm : Thực hiện thêm 1 Account, có những thông tin bắt buộc là Password và Role của Account đó, với điều kiện: Username của Account sẽ là ID của Employee và Doctor (Bắt buộc phải có employee hoặc Doctor thì mới có tài khoản).
* Button Sửa : Click đúp vào Account (trên Datagridview) muốn chỉnh sửa, có quyền chỉnh sửa password cũng như role của Account đó.
* Button Xóa : Thực hiện xóa 1 Account (chọn như button Sửa).
* Button Thoát : Thoát khỏi Form quản lý Account.

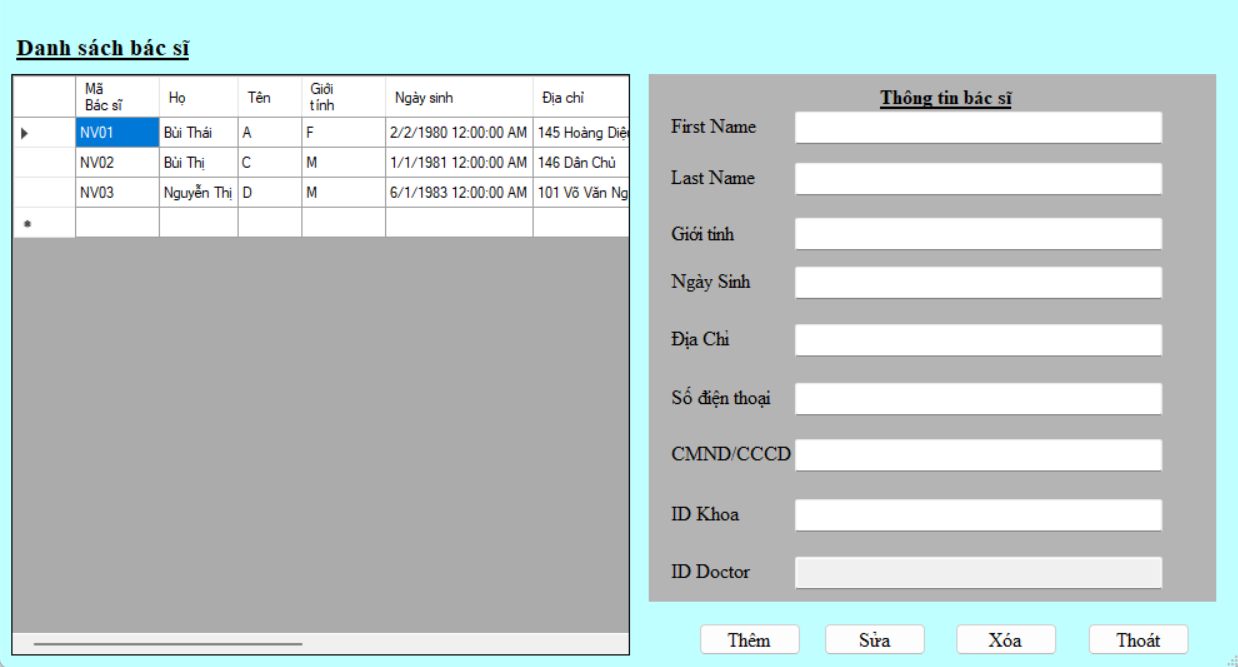
### 6.2.3. Form quản lý Departments (frmDepartment):



Hình 6.2.3.1. Form quản lý Departments (frmDepartment)

* Chọn button Quản lý phòng bản từ Admin\_Home sẽ chuyển sang form quản lý phòng ban, hiển thị danh sách tất cả phòng ban có trong database trên Datagridview. Có chức năng tìm kiếm phòng ban theo tên, nhập tên phòng ban vào txtTimKiem thì thông tin phòng ban sẽ được load ở DatagridView. Và các button:
* Button Thêm : Thực hiện thêm 1 phòng ban, chỉ cần nhập tên phòng ban vào txtName, sau đó nhấp button thêm thì phòng ban sẽ được thêm vào Database.
* Button Sửa : Thực hiện sửa 1 phòng ban có sẵn được chọn từ Datagridview, chỉ có thể sửa tên.
* Button Xóa : Thực hiện xóa 1 phòng ban có sẵn được chọn từ DatagridView.
* Button Lưu : Lưu dữ liệu vừa : Thêm, Sửa, Xóa vào Database.
* Button Reload : Load lại dữ liệu sau khi tìm kiếm.

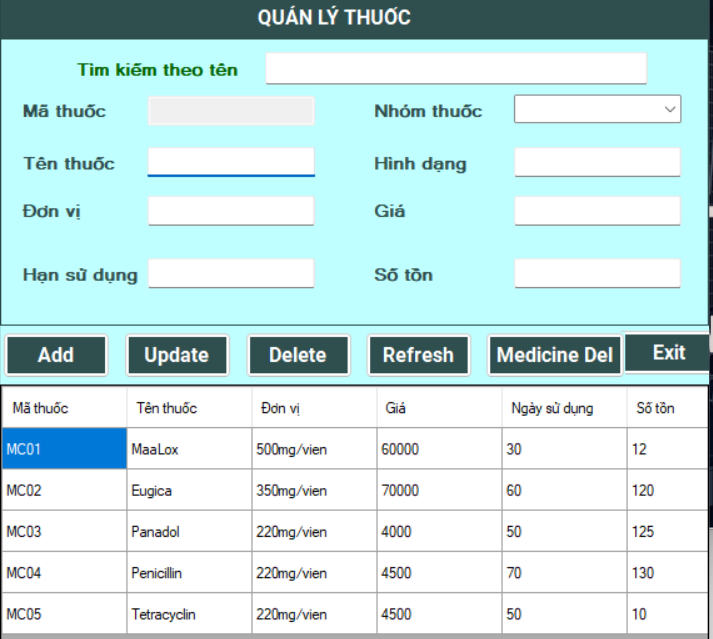
### 6.2.4. Form quản lý Doctor (FrmDoctors):



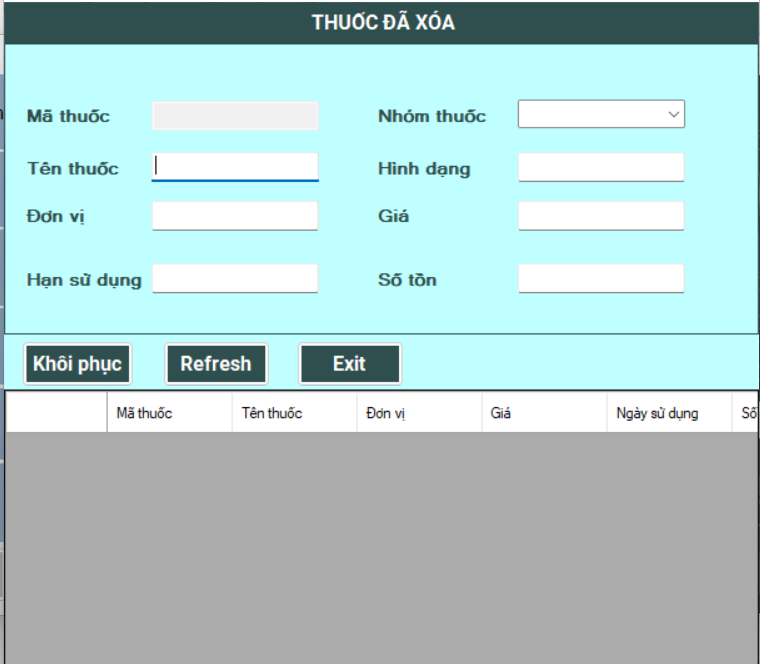
*Hình 6.2.4.1. Form quản lý Doctor (FrmDoctors)*

* Chọn button Quản lý Bác sĩ ở Admin\_home sẽ chuyển đến form quản lý Bác sĩ, hiển thị danh sách thông tin tất cả các Bác sĩ có trong Database. Và các button:
* Button Thêm : Thực hiện thêm 1 bác sĩ với các trường textbox bắt buộc như First Name, Last Name, ……..
* Button Sửa : Chỉnh sửa thông tin của 1 Bác sĩ được chọn từ DatagridView, không bao gồm cả ID Doctor.
* Button Xóa : Thực hiện xóa 1 Doctor theo ID Doctor được chọn từ DataGridView.
* Button Thoát : Thoát khỏi form quản lý Bác sĩ.

### 6.2.5. Form quản lý Medicines (frmMedicine - frmMedicineDel):

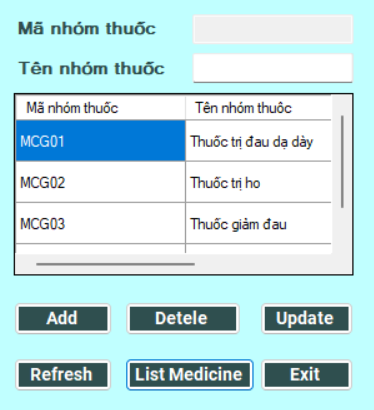


Hình 6.2.5.1. Form frmMedicine



Hình 6.2.5.1. Form frmMedicineDel

### 6.2.6. Form quản lý Medicine Group (frmMedicineGroup):



Hình 6.2.6.1. Form quản lý Medicine Group (frmMedicineGroup)

### 6.2.7. Form quản lý Services (frmService):



Hình 6.2.7.1. Form quản lý Services (frmService)

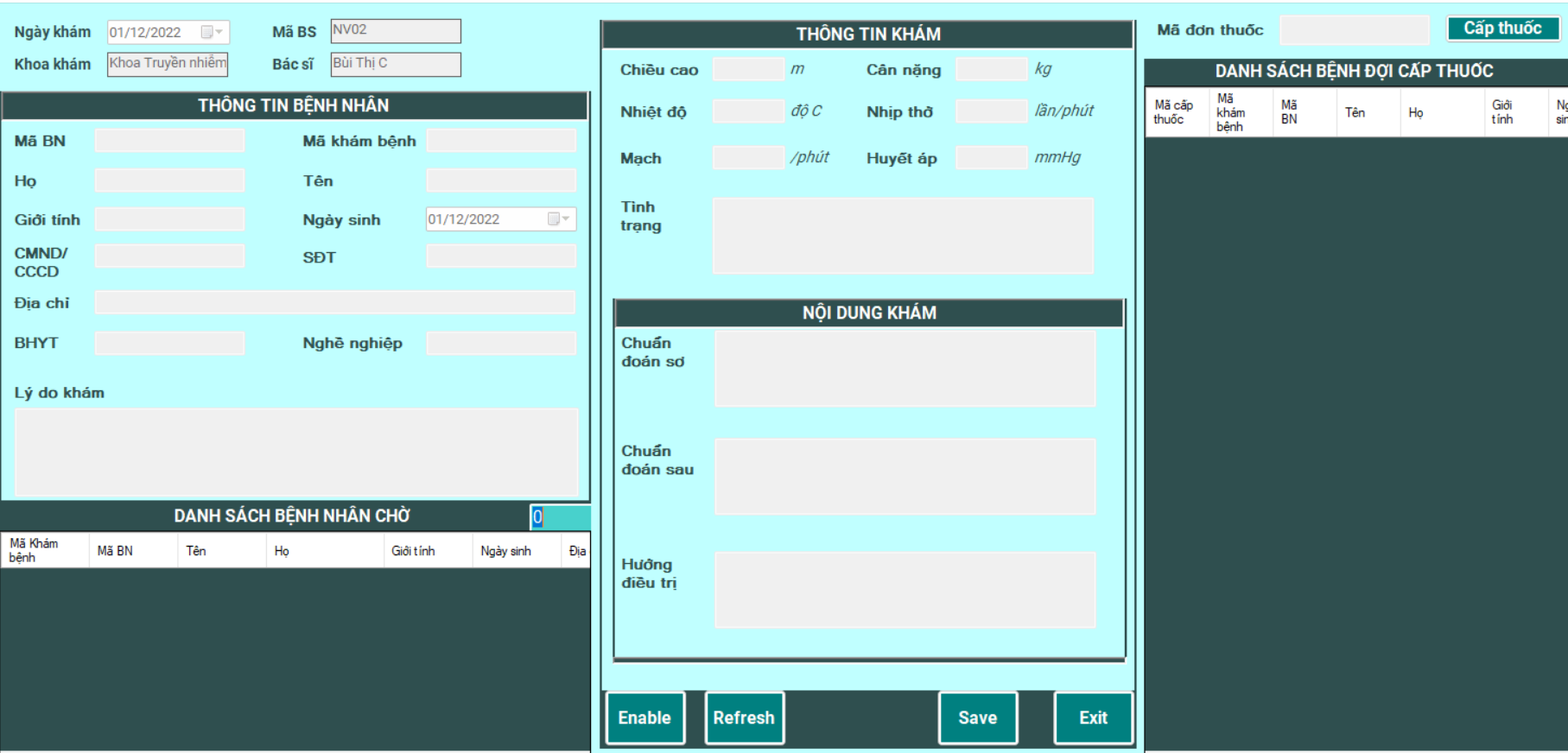
### 6.2.8. Form quản lý Employee (Không tính Doctors - FrmTatCaNhanVien):



Hình 6.2.8.1. Form quản lý Employee (Không tính Doctors - FrmTatCaNhanVien)

## 6.3. Các form trong role Doctor:

### 6.3.1. Form quản lý Examinations (frmExamination):



Hình 6.3.1.1. Form quản lý Examinations (frmExamination)

### 6.3.2. Form quản lý Prescriptions (frmPrescription):



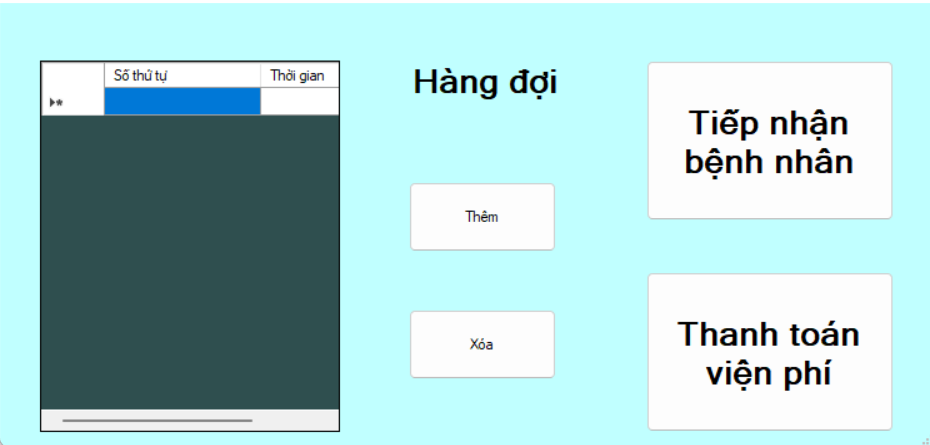
Hình 6.3.2.1. Form quản lý Prescriptions (frmPrescription)

Form Prescriptions giúp quản lý việc tạo đơn thuốc cho bệnh nhân:

* Để tạo đơn thuốc

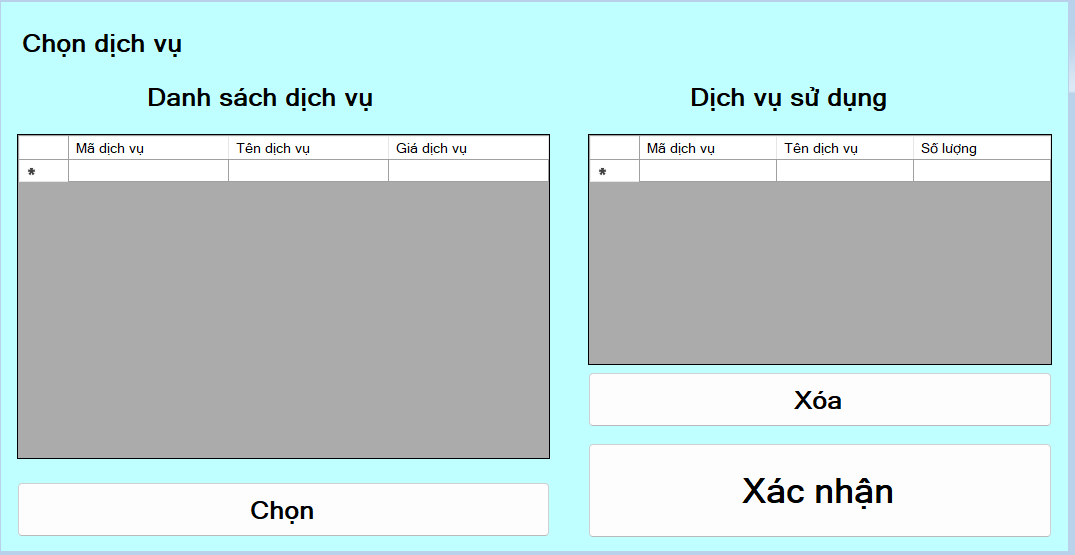
## 6.4. Các form trong role Employee(Staff):

### 6.4.1. Form trang chủ role Employee (TrangChu):



Hình 6.4.1.1. Form quản lý Prescriptions (frmPrescription)

### 6.4.2. Form quản lý UsingService (Sudungdichvu):



Hình 6.4.2.1. Form quản lý UsingService (Sudungdichvu)

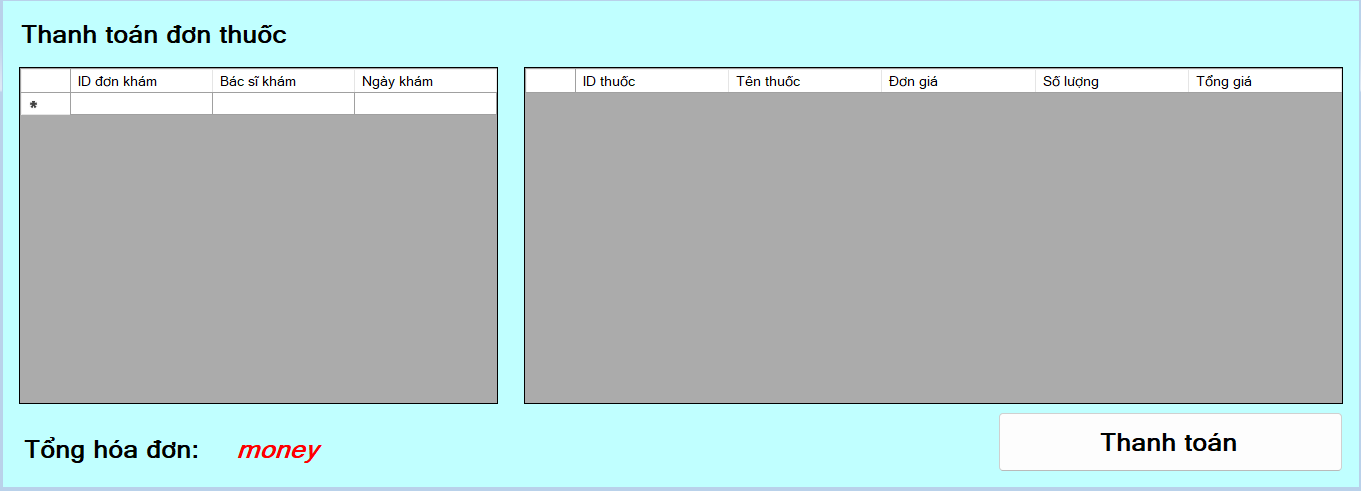
Form UsingService hiển thị các thông tin:

* Danh sách các dịch vụ của bệnh viện
* Danh sách các dịch vụ bệnh nhân sử dụng

Button Xóa: chọn 1 dịch vụ muốn xóa(không sử dụng nữa) → Xóa → Xác nhận

Button Xác nhận: kiểm tra chắc chắn rằng bạn muốn xóa dịch vụ.

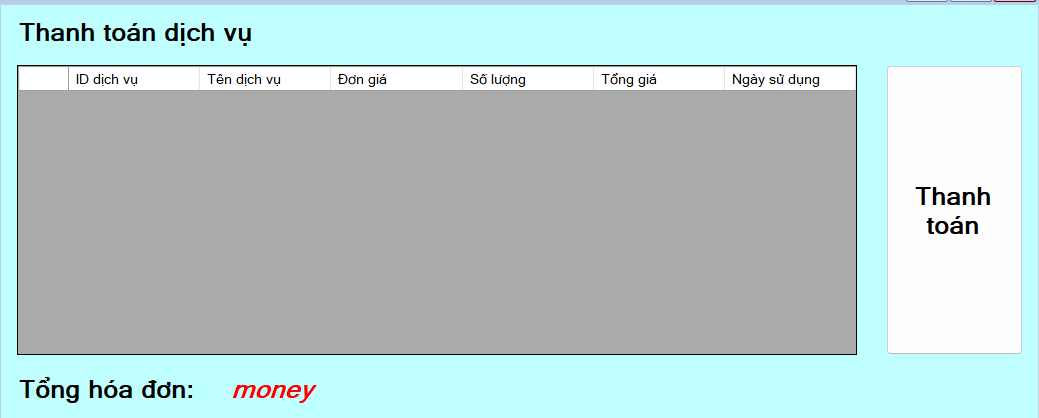
### 6.4.3. Form Pay Prescription (ThanhToanDonThuoc):



Hình 6.4.3.1. Form Pay Prescription (ThanhToanDonThuoc)

Form hiển thị các thông tin về đơn khám:tên bác sĩ, ngày khám bệnh, thông tin của đơn thuốc. Khi click “Thanh toán”, form hiển thị tổng số tiền của bệnh nhân.

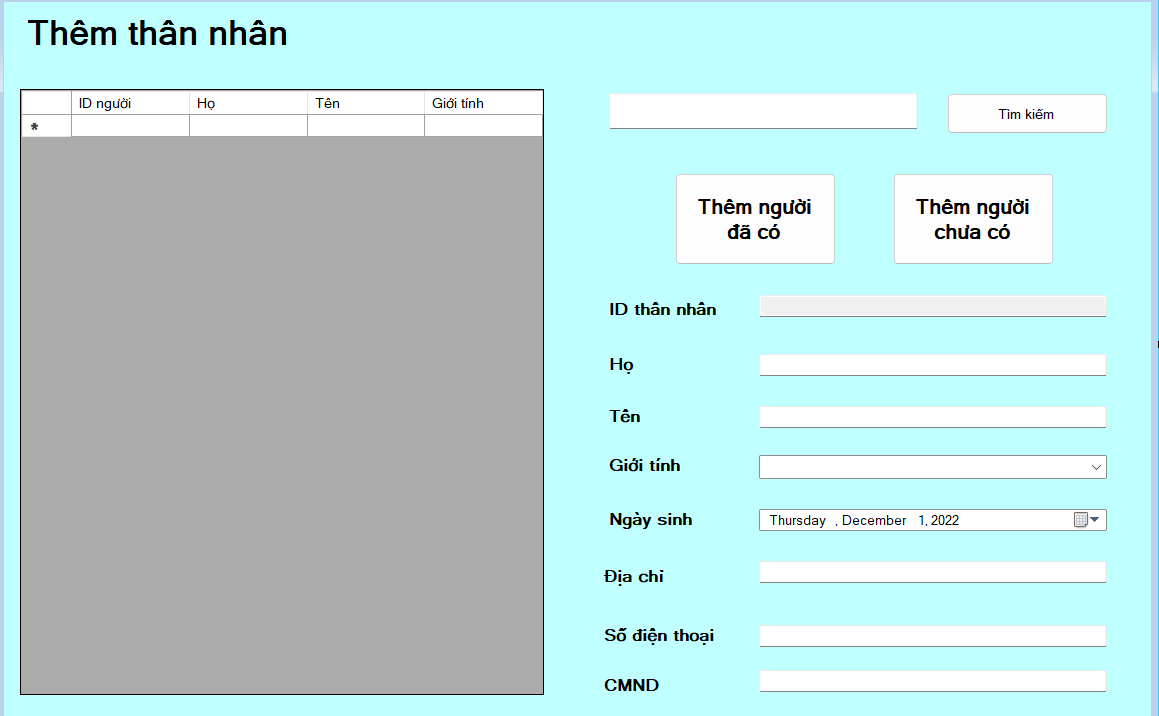
### 6.4.4. Form Pay usingService (ThanhToanSuDungDichVu):



Hình 6.4.4.1. Form Pay usingService (ThanhToanSuDungDichVu)

Form hiển thị các thông tin của dịch vụ mà bệnh nhân đã sử dụng. Khi click “Thanh toán”, hiển thị tổng số tiền của bệnh nhân phải trả khi sử dụng các dịch vụ.

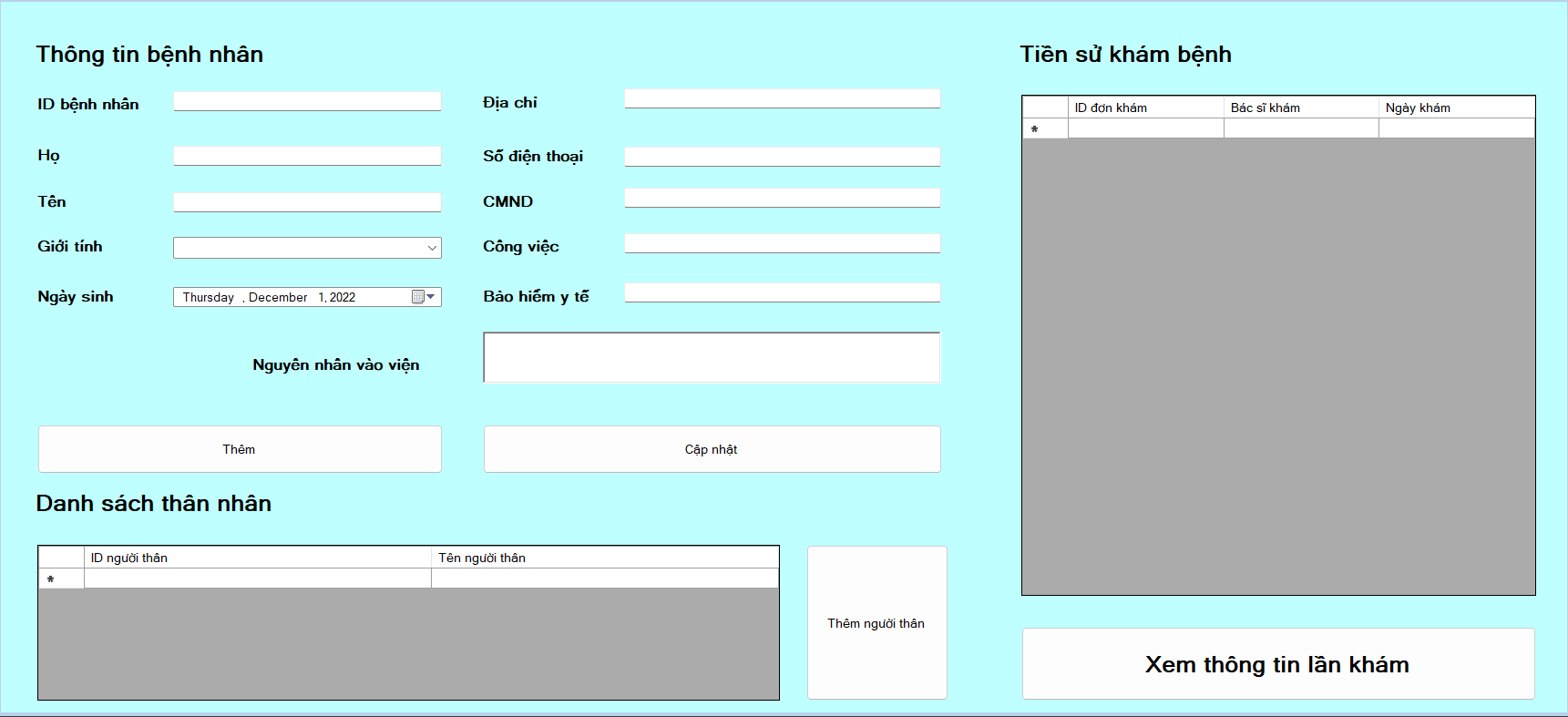
### 6.4.5. Form quản lý Relatives (ThemThanNhan):



Hình 6.4.5.1. Form quản lý Relatives (ThemThanNhan)

Form Thêm thân nhân hiển thị các thông tin sau: Các thân nhân đã có của bệnh nhân

### 6.4.6. Form quản lý Patients (ThongTinBenhNhan):



Hình 6.4.6.1. Form quản lý Patients (ThongTinBenhNhan)

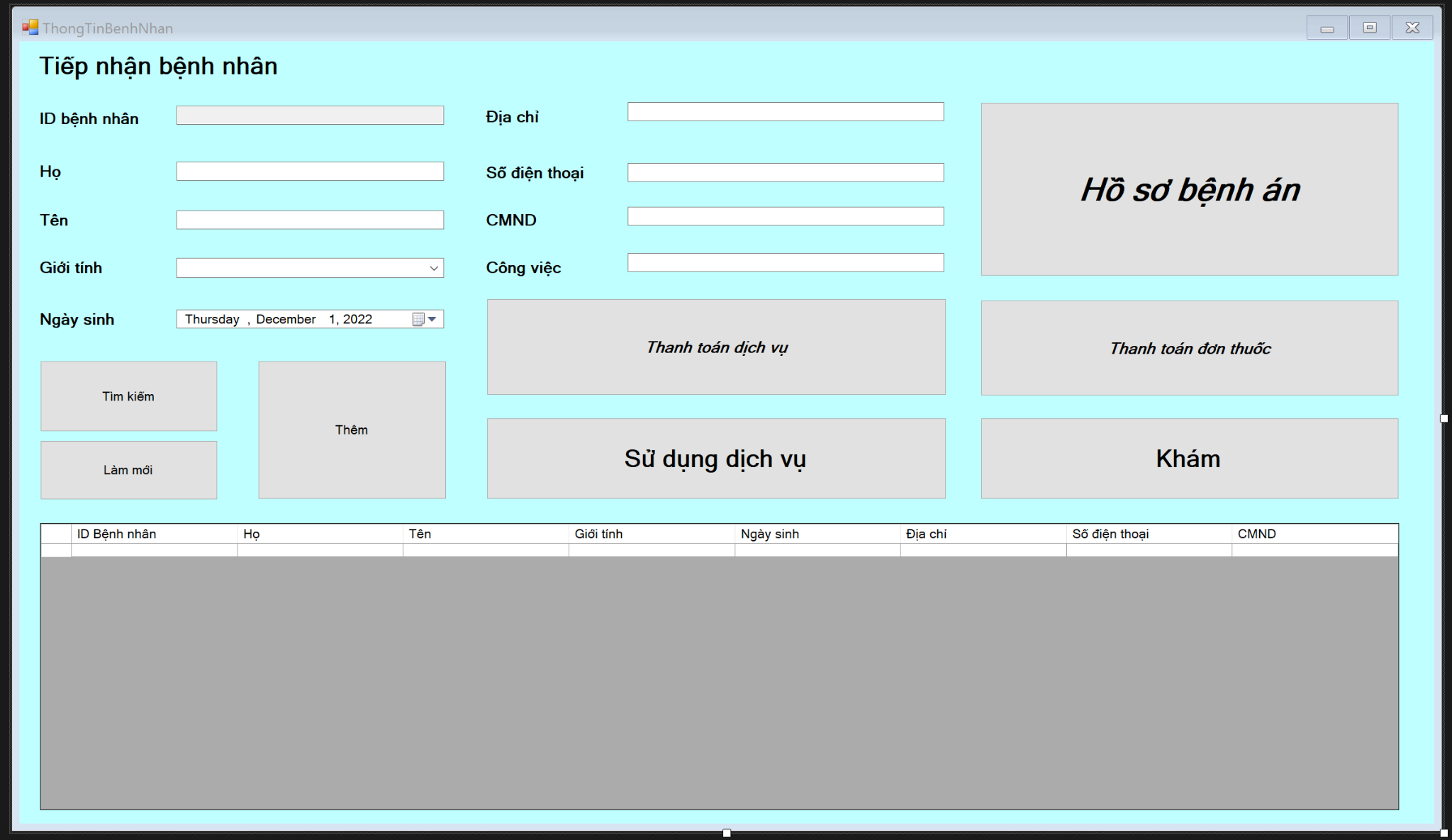
### 6.4.7. Form thông tin Examinations (ThongTinDonKham):



Hình 6.4.7.1. Form quản lý Patients (ThongTinBenhNhan)

Form ThongTinDonKham hiển thị thông tin của đơn khám: ID đơn khám, ID bệnh nhân, họ tên bệnh nhân, Họ tên bác sĩ,.... và các thông tin về tình hình sức khỏe của bệnh nhân. Đồng thời, hiển thị thông tin đơn thuốc của bệnh nhân đó.

### 6.4.8. Form tiếp nhận Patients (TiepNhanBenhNhan):



Hình 6.4.8.1. Form tiếp nhận Patients (TiepNhanBenhNhan)

# VII. Kết luận:

Phần mềm quản lý hồ sơ bệnh án với các tính năng cần thiết đáp ứng được nhu cầu sử dụng của các phòng khám và giúp cho việc quản lý hoạt động của phòng được dễ dàng hơn.

Về cơ bản thì phần mềm quản lý được các thông tin về Bệnh nhân, thân nhân, nhân viên, bác sĩ, phòng ban, dịch vụ, thuốc, quá trình khám và thanh toán. Phần mềm hỗ trợ các thao tác thêm, sửa, xoá các thông tin liên quan đến các đối tượng trên và thống kê theo dạng bảng một cách trực quan cho người sử dụng. Cơ sở dữ liệu lưu giữ thông tin của hệ thống sử dụng các view, functions, procedures, triggers, transactions để đảm bảo cho dữ liệu luôn được độc lập, rõ ràng, nhất quán và bền vững. Bên cạnh những ưu điểm trên thì hệ thống còn hạn chế về những chức năng như phân quyền, bảo mật.

Qua đồ án này nhóm em đã rút ra được nhiều kinh nghiệm khi làm một dự án liên quan đến thực tiễn, có được khả năng phân tích và đánh giá bài toán. Và thông qua được đồ án này thì giúp nhóm em hiểu rõ hơn về công cụ mà mình đã lựa chọn và cũng như phần lý thuyết, phương pháp tìm hiểu về tính đặc thù của các bài toán. Sử dụng tốt các chức năng mà SQL Server đã tạo ra.

Trong quá trình tìm hiểu và làm bài, chúng em đã cố gắng hoàn thiện bài làm và hoàn thiện mình hơn về mặt kiến thức. Nhưng cũng không thể tránh khỏi những thiếu sót mong thầy cô và các bạn đóng góp ý kiến để chúng em có thể hoàn thiện tốt hơn.

# 

# TÀI LIỆU THAM KHẢO

Trong quá trình thực hiện đồ án, nhóm 15 đã tham khảo những tài liệu sau:

1. Slide và video bài giảng các chương 1 - 2 - 3 - 4 - 5 của thầy Nguyễn Thanh Sơn trên trang Utex.
2. List video hướng dẫn Entity Framework Database First - Mô hình Three layer - Gọi Function, Store Procedure, View <https://www.youtube.com/watch?v=O4am6MmhFVU&t=336s>
3. Tài liệu hướng dẫn SQL <https://www.w3schools.com/sql/default.asp>,
4. Tài liệu hướng dẫn Function <https://learn.microsoft.com/en-us/ef/ef6/modeling/designer/advanced/tvfs?redirectedfrom=MSDN> .
5. Tham khảo đồ án [https://lucid.app/lucidchart/55ec3d6e-e76c-4251-a56d-acdb9883a487/edit?viewport\_loc=-341%2C-132%2C2679%2C1331%2C0\_0&invitationId=inv\_8e6cf5c0-da8f-43de-8743-e12d88bc1315#](https://lucid.app/lucidchart/55ec3d6e-e76c-4251-a56d-acdb9883a487/edit?viewport_loc=-341%2C-132%2C2679%2C1331%2C0_0&invitationId=inv_8e6cf5c0-da8f-43de-8743-e12d88bc1315)
6. Trang web hỏi đáp tìm hiểu thông tin,<https://stackoverflow.com/>