BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC VIỆT NHẬT

XÂY DỰNG PHẦN MỀM
QUẢN LÝ GARA Ô TÔ

BÀI TẬP LỚN
MÔN HỌC: LẬP TRÌNH NÂNG CAO

Sinh viên thực hiện: Nguyễn Minh Hiếu

Mã sinh viên : 23110168

Giảng viên : Chu Thị Minh Huệ

HÀ NỘI - 2025

MỤC LỤC

I. YÊU CẦU

1. Yêu cầu người dùng

- Mục tiêu chung:

    + Hệ thống quản lý gara sửa chữa xe ô tô/mô tô giúp lưu trữ thông tin khách hàng, xe, phiếu sửa chữa, kỹ thuật viên, phụ tùng, hóa đơn và lịch bảo dưỡng.

    + Hỗ trợ quy trình nghiệp vụ từ tiếp nhận xe, chẩn đoán, duyệt báo giá, thực hiện sửa chữa, xuất hóa đơn và thanh toán.

- Các yêu cầu cụ thể của người dùng:

    + Quản lý thông tin khách hàng: tạo, sửa, xóa, tìm kiếm; lưu nhiều xe cho một khách hàng.

    + Quản lý thông tin xe: biển số, loại xe, đời xe, mô tả, km hiện tại.

    + Tạo phiếu tiếp nhận/phiếu sửa chữa (ServiceOrder): ghi nhận hạng mục sửa chữa, phụ tùng cần thay, nhân viên thực hiện, thời gian dự kiến, trạng thái (mới, đang làm, hoàn thành, hủy).

    + Quản lý kho phụ tùng: tồn kho, nhập xuất, theo dõi giá vốn, cảnh báo tồn.

    + Tạo báo giá cho khách (Estimate) và cho phép khách duyệt/không duyệt.

    + Quản lý hóa đơn và thanh toán (hỗ trợ nhiều phương thức: tiền mặt, chuyển khoản, thẻ).

    + Lịch hẹn/đặt lịch trước: cho phép đặt lịch cho khách.

    + Báo cáo: doanh thu theo ngày/tuần/tháng, tồn kho, lịch sử sửa chữa của xe, hiệu suất kỹ thuật viên.

    + Phân quyền: quản trị (admin), nhân viên tiếp nhận, kỹ thuật viên, thu ngân; thao tác giới hạn theo vai trò.

    + Giao diện web (hoặc desktop) thân thiện, phản hồi nhanh, có API REST để tích hợp.

- Yêu cầu phi chức năng:

+ Bảo mật: xác thực, phân quyền, mã hóa dữ liệu nhạy cảm.

+ Hiệu năng: phản hồi nhanh với lượng dữ liệu lớn.

+ Khả năng mở rộng: kiến trúc modul, tách dịch vụ nếu cần.

+ Sao lưu/khôi phục dữ liệu, logging, audit.

2. Phân tích các chức năng (Functional Requirements)

- Các chức năng CRUD chính (một số với chi tiết):

Khách hàng (Customer)

- Create: Thêm khách hàng mới (Tên, ĐT, Email, Địa chỉ, Ghi chú).

- Read: Danh sách, tìm kiếm theo tên/SDT, xem chi tiết, lịch sử xe/phiếu sửa.

- Update: Sửa thông tin khách hàng.

- Delete: Xoá (hoặc đánh dấu xóa mềm).

Xe (Vehicle)

- Create: Thêm xe cho khách (Biển số, Model, VIN, Năm SX, Km).

- Read: Tra cứu theo biển số/VIN, danh sách theo khách.

- Update/Delete: Sửa/xóa.

Phiếu sửa chữa / tiếp nhận (ServiceOrder)

- Create: Tạo phiếu, gồm danh sách hạng mục công việc, phụ tùng, nhân viên phụ trách.

- Read: Tra cứu theo số phiếu, trạng thái, xe, khách.

- Update: Cập nhật trạng thái, thêm công việc/phụ tùng.

- Delete: Hủy phiếu (ghi nhận lịch sử).

- Workflow: Tạo -> Chờ duyệt báo giá -> Đang thực hiện -> Hoàn thành -> Xuất hóa đơn -> Thanh toán.

Kho phụ tùng (Inventory / SparePart)

- CRUD phụ tùng: mã, tên, mô tả, số lượng tồn, giá nhập/giá bán.
- Nhập kho, xuất kho khi dùng trong phiếu sửa chữa.
- Cảnh báo tồn tối thiểu.

Kỹ thuật viên (Mechanic / User)

- Quản lý thông tin nhân viên, phân công phiếu.
- Theo dõi thời gian thực hiện, ghi nhật ký công việc.

Báo cáo

- Doanh thu theo thời gian.
- Tồn kho.
- Lịch sử sửa chữa theo xe/khách.
- Hiệu suất kỹ thuật viên (số phiếu, doanh thu).

II. MÔ HÌNH THỰC THỂ (Entity Model)

1. Tổng quan cơ sở dữ liệu

- Hệ thống sử dụng SQL Server làm hệ quản trị cơ sở dữ liệu. Việc thiết kế, truy vấn và quản lý schema được thực hiện bằng Entity Framework Core (EF Core) theo phương pháp Code-First.
- Các bảng (Tables) sẽ được ánh xạ thành các lớp (Classes) tương ứng (mỗi lớp tương ứng một Entity). Quan hệ giữa các bảng (FK, 1–N, N–M) được cấu hình bằng Fluent API hoặc Data Annotations trong OnModelCreating của DbContext.
- Việc tạo/ thay đổi cấu trúc DB sẽ thực hiện bằng Migration (Create-Migration / Update-Database) để đảm bảo thay đổi schema được quản lý có phiên bản.
- Các thực thể được chia nhóm theo chức năng nghiệp vụ để dễ quản lý và mở rộng.

2. Danh sách các thực thể (Entities)

a. Nhóm quản trị hệ thống

- Role (tblRoles): Quản lý vai trò và quyền hạn hệ thống
  - RoleId (PK)
  - Name (tên vai trò: Admin, Reception, Mechanic, Cashier)
  - Description
- User (tblUsers): Lưu trữ thông tin tài khoản người dùng, liên kết Role
  - UserId (PK)
  - Username
  - PasswordHash
  - FullName
  - Phone
  - RoleId (FK -> tblRoles)
  - IsActive, CreatedAt, LastLogin

b. Nhóm khách hàng

- Customer (tblCustomers): Lưu trữ thông tin khách hàng
  - CustomerId (PK)
  - FullName
  - Phone
  - Address
  - Note
  - CreatedAt
- ContactHistory (tblCustomerContacts) (tùy chọn): Ghi nhận liên hệ / tương tác
  - ContactId (PK)

- CustomerId (FK)
- ContactType (Call /Visit)
- Content, ContactDate, UserId (người thực hiện)

c. Nhóm xe (Vehicle)

- Vehicle (tblVehicles): Thông tin xe của khách
  - VehicleId (PK)
  - CustomerId (FK -> tblCustomers)
  - PlateNumber (unique)
  - Model
  - VIN
  - Year
  - CurrentKm
  - Color, Note
  - CreatedAt
- VehicleServiceHistory (tblVehicleHistory) (view / log): Lưu lịch sử km, lần bảo dưỡng (có thể là DTO/Report)

d. Nhóm phiếu sửa chữa / tiếp nhận (ServiceOrder)

- ServiceOrder (tblServiceOrders): Phiếu tiếp nhận / sửa chữa
  - ServiceOrderId (PK)
  - OrderNumber (mã phiếu, có thể format)
  - VehicleId (FK)
  - CustomerId (FK)
  - MechanicId (FK -> tblUsers, nullable)
  - Status (New, Quoted, Approved, InProgress, Completed, Cancelled)
  - CreatedAt, ExpectedFinish, CompletedAt

  - Notes, TotalAmount (tổng dự tính)
- ServiceOrderItem (tblServiceOrderItems): Hạng mục công việc (công lao động)
  - ItemId (PK)
  - ServiceOrderId (FK)
  - Description
  - LaborHours
  - UnitPrice
  - Quantity
  - Amount
- ServiceOrderPart (tblServiceOrderParts): Phụ tùng sử dụng trong phiếu
  - PartLineId (PK)
  - ServiceOrderId (FK)
  - SparePartId (FK -> tblSpareParts)
  - Quantity
  - UnitPrice
  - Amount

e. Nhóm phụ tùng & kho (Inventory / Spare Parts)

- SparePart (tblSpareParts): Danh mục phụ tùng trong kho
  - SparePartId (PK)
  - Code (mã phụ tùng)
  - Name
  - Description
  - QuantityInStock
  - ReorderLevel (mức cảnh báo)

- CostPrice, SellPrice

- InventoryTransaction (tblInventoryTransactions): Ghi nhận nhập / xuất kho

- TransactionId (PK)

- SparePartId (FK)

- TransactionType (IN/OUT/ADJUSTMENT)

- Quantity

- ReferenceType (PO / ServiceOrder / Adjustment)

- ReferenceId (tham chiếu đến phiếu liên quan)

- TransactionDate

- Note

f. Nhóm hóa đơn & thanh toán (Invoice / Payment)

- Invoice (tblInvoices): Hóa đơn phát sinh sau khi hoàn thành/ hoặc thanh toán partial

- InvoiceId (PK)

- InvoiceNumber

- ServiceOrderId (FK)

- InvoiceDate

- SubTotal, VAT, Discount, Total

- Status (Unpaid, Paid, PartiallyPaid)

- Payment (tblPayments): Ghi nhận giao dịch thanh toán

- PaymentId (PK)

- InvoiceId (FK)

- PaymentDate

- Amount

- PaymentMethod (Cash, BankTransfer, Card)

- Reference, Note

g. Nhóm lịch hẹn (Appointment)

- Appointment (tblAppointments): Đặt lịch nhận xe
  - AppointmentId (PK)
  - CustomerId (FK)
  - VehicleId (FK)
  - ScheduledDate, TimeSlot
  - Status (Scheduled, Confirmed, Cancelled, Completed)
  - Notes, CreatedBy (UserId)

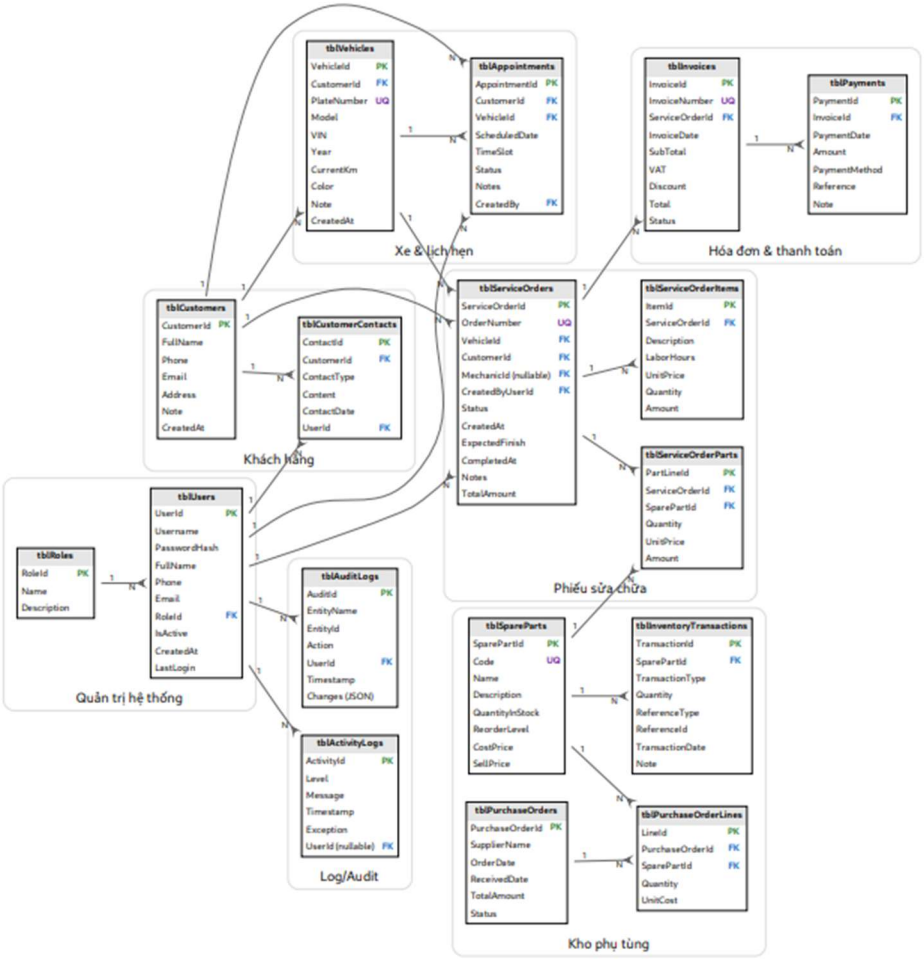
h. Nhóm báo cáo / log / audit

- AuditLog (tblAuditLogs): Ghi hành vi thay đổi quan trọng
  - AuditId (PK)
  - EntityName, EntityId
  - Action (Create/Update/Delete)
  - UserId, Timestamp
  - Changes (JSON)
- ActivityLog (tblActivityLogs): Log hệ thống / lỗi

i. Các lớp DTO (Data Transfer Objects) và ViewModel

- Các DTO phục vụ Presentation và API, không map 1-1 với bảng:
  - CustomerDto, VehicleDto, ServiceOrderDto, ServiceOrderDetailDto, SparePartDto, InventoryDto, InvoiceDto, PaymentDto
- Các lớp Statistics / Report DTO:
  - RevenueByDateDto, StockAlertDto, MechanicPerformanceDto

- Lưu ý: DTO chỉ dùng để truyền dữ liệu giữa Presentation <-> Service; không thay thế Entity trong DbContext.



Hình 1: Sơ đồ quan hệ thực thể

III. THIẾT KẾ CÁC THÀNH PHẦN PHẦN MỀM (Kiến trúc 3 lớp)

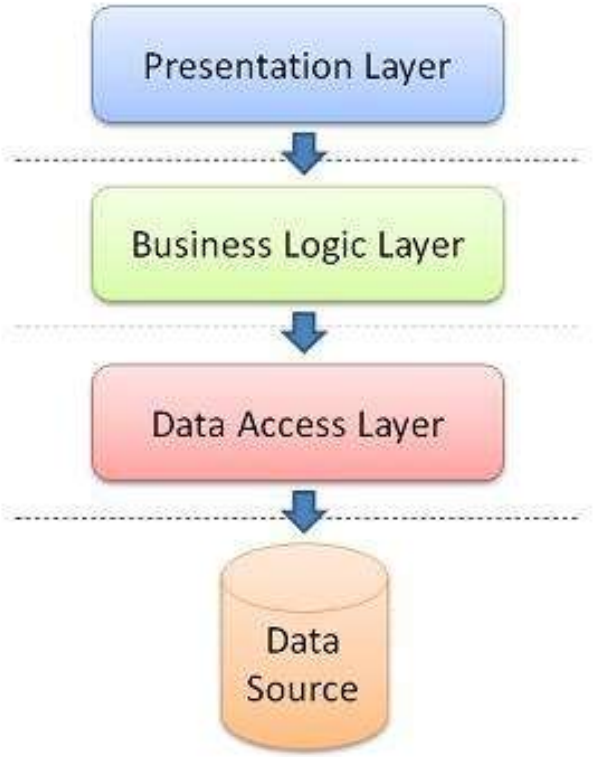
1. Kiến trúc tổng thể.

Hệ thống được xây dựng theo mô hình Kiến trúc 3 lớp (3 – Layer Architechture) kết hợp với mô hình Entity Data Model và các tầng được tách thành các components riêng biệt. Việc thiết kế theo mô hình này giúp phân chia mã

nguồn rõ ràng, dễ bảo trì, dễ mở rộng và tách biệt giữa giao diện người dùng và logic xử lý.

Mô hình phần mềm bao gồm các tầng:

- Presentation Layer (GUI): Tầng giao diện người dùng (WinForms).
- Business Logic Layer (BLL): Tầng xử lý nghiệp vụ.
- Data Access Layer (DAL): Tầng truy cập dữ liệu (Sử dụng công nghệ Entity Framework Core).
- Data Transfer Object (DTO/Entities): Tầng đối tượng vận chuyển dữ liệu giữa các tầng.



Hình . Mô hình kiến trúc 3 lớp

2. Tổ chức Dự án (Solution Structure)

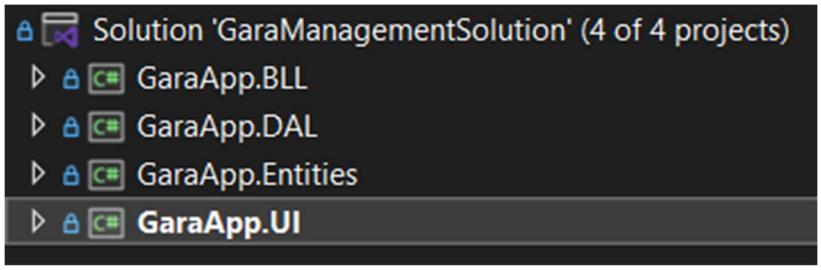
Clinic Management Project trong Visual Studio được tổ chức thành 4 project thành phần riêng biệt trong cùng một solution nhằm đảm bảo tính đóng gói:

- GUI (ClinicManagement): Project chính để chạy chương trình, chứa các forms giao diện người dùng (frmLogin, frmMain,...).

- BLL (Business Logic): Chứa các file Services (ServiceOrderService, InventoryService, InvoiceService,…) thực hiện các logic nghiệp vụ.

- DAL (Data Access): Chứa GarageDbContext, cấu hình entity (IEntityTypeConfiguration), repository, migration (EF Core migrations), seed data, script T-SQL nếu cần.

- Entities: Chứa các lớp thực thể(Customer, Vehicle, ServiceOrder,...)



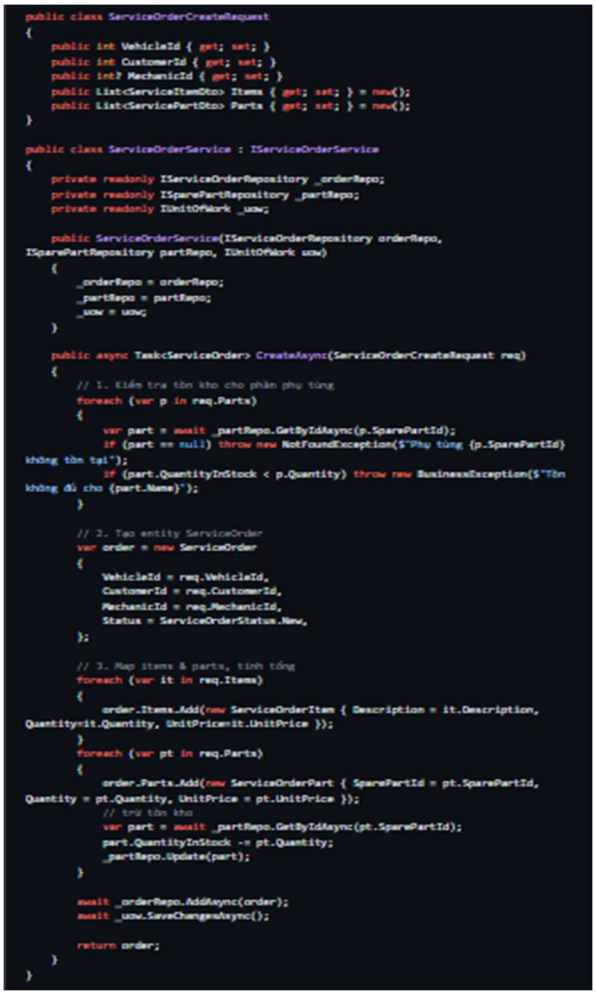
Hình 3. Cấu trúc Solution

3. Ví dụ minh họa một chức năng (Tạo phiếu sửa chữa) — từ tầng dưới lên trên

a) DAL: Repository method (ví dụ dùng Generic Repository)

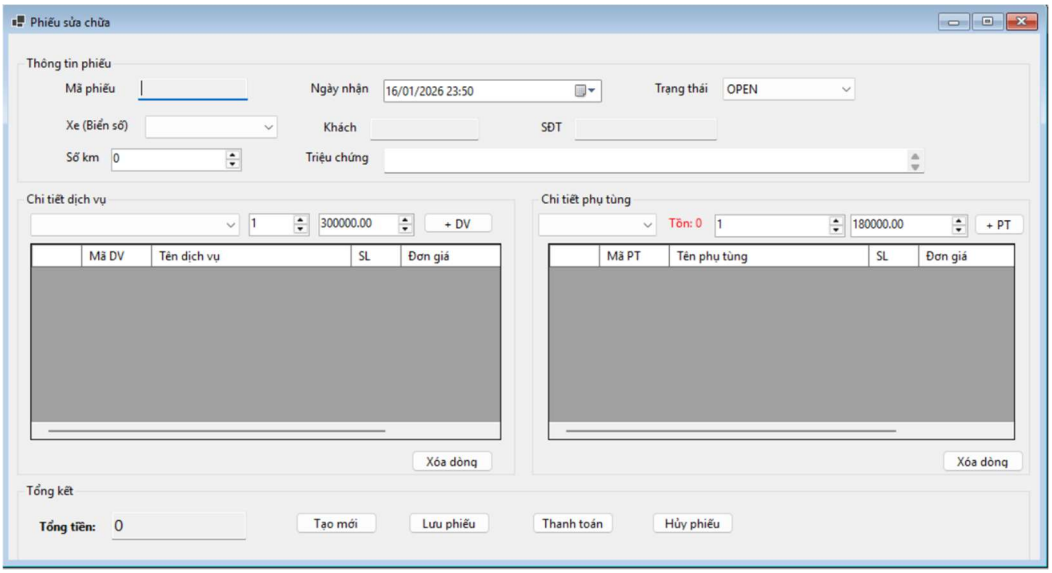
```
// IGenericRepository.cs
public interface IGenericRepository<T> where T : class
{
    Task<T?> GetByIdAsync(int id);
    IQueryable<T> Query();
    Task AddAsync(T entity);
    void Update(T entity);
    void Remove(T entity);
}

// ServiceOrderRepository.cs (nếu cần method đặc thù)
public interface IServiceOrderRepository : IGenericRepository<ServiceOrder>
{
    Task<ServiceOrder?> GetWithDetailsAsync(int id);
}
```

b) BLL: Service xử lý nghiệp vụ tạo phiếu



```
public class ServiceOrderCreateRequest
{
    public int VehicleId { get; set; }
    public int CustomerId { get; set; }
    public int? MechanicId { get; set; }
    public List<ServiceItemDto> Items { get; set; } = new();
    public List<ServicePartDto> Parts { get; set; } = new();
}

public class ServiceOrderService : IServiceOrderService
{
    private readonly IServiceOrderRepository _orderRepo;
    private readonly ISparePartRepository _partRepo;
    private readonly IUnitOfWork _uow;

    public ServiceOrderService(IServiceOrderRepository orderRepo,
ISparePartRepository partRepo, IUnitOfWork uow)
    {
        _orderRepo = orderRepo;
        _partRepo = partRepo;
        _uow = uow;
    }

    public async Task<ServiceOrder> CreateAsync(ServiceOrderCreateRequest req)
    {
        // 1. Kiểm tra tồn kho cho phần phụ tùng
        foreach (var p in req.Parts)
        {
            var part = await _partRepo.GetByIdAsync(p.SparePartId);
            if (part == null) throw new NotFoundException($"Phụ tùng {p.SparePartId}
không tồn tại");
            if (part.QuantityInStock < p.Quantity) throw new BusinessException($"Tồn
không đủ cho {part.Name}");
        }

        // 2. Tạo entity ServiceOrder
        var order = new ServiceOrder
        {
            VehicleId = req.VehicleId,
            CustomerId = req.CustomerId,
            MechanicId = req.MechanicId,
            Status = ServiceOrderStatus.New,
        };

        // 3. Map items & parts, tính tổng
        foreach (var it in req.Items)
        {
            order.Items.Add(new ServiceOrderItem { Description = it.Description,
Quantity=it.Quantity, UnitPrice=it.UnitPrice });
        }
        foreach (var pt in req.Parts)
        {
            order.Parts.Add(new ServiceOrderPart { SparePartId = pt.SparePartId,
Quantity = pt.Quantity, UnitPrice = pt.UnitPrice });
            // trừ tồn kho
            var part = await _partRepo.GetByIdAsync(pt.SparePartId);
            part.QuantityInStock -= pt.Quantity;
            _partRepo.Update(part);
        }

        await _orderRepo.AddAsync(order);
        await _uow.SaveChangesAsync();

        return order;
    }
}
```

c) UI



IV. CHƯƠNG TRÌNH PHẦN MỀM

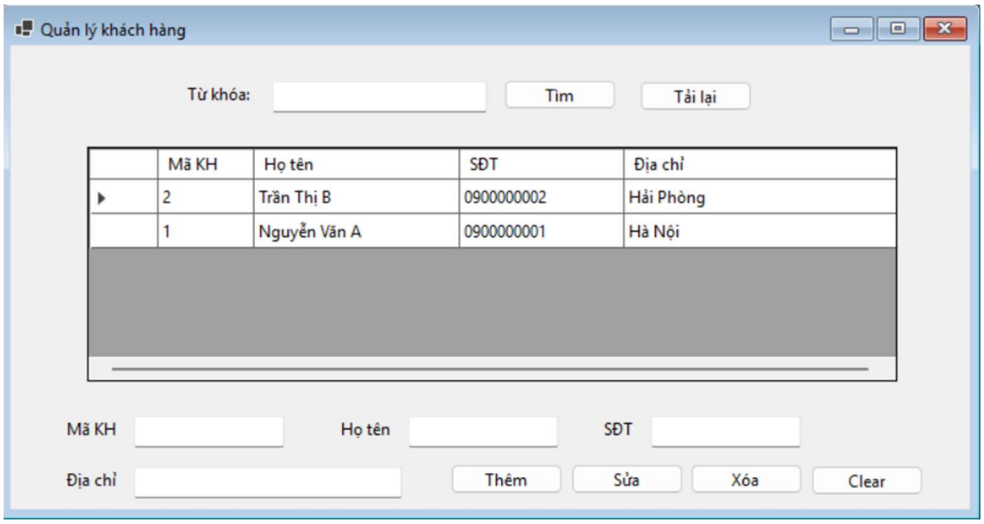
1. Màn hình đăng nhập

- Mô tả: Giao diện khởi động yêu cầu người dùng nhập tài khoản (tên đăng nhập) và mật khẩu. Hệ thống thực hiện xác thực, sau khi đăng nhập thành công sẽ tải thông tin người dùng (vai trò) và hiển thị các chức năng tương ứng.

Hình 6. Giao diện màn hình đăng nhập

2) Khách hàng (Customer Management)

- Mục đích: Quản lý thông tin khách hàng và lịch sử liên quan.
- Thành phần chính:
    + Bảng danh sách khách hàng, form thêm/sửa chi tiết.
    + Tìm kiếm/auto-complete theo tên/SDT.
    + Nút Thêm, Sửa, Xoá
- Chức năng:
    + CRUD khách hàng (Create, Read, Update, Delete / soft-delete).



Hình 7. Giao diện màn hình quản lý khách hàng

3) Xe (Vehicle)

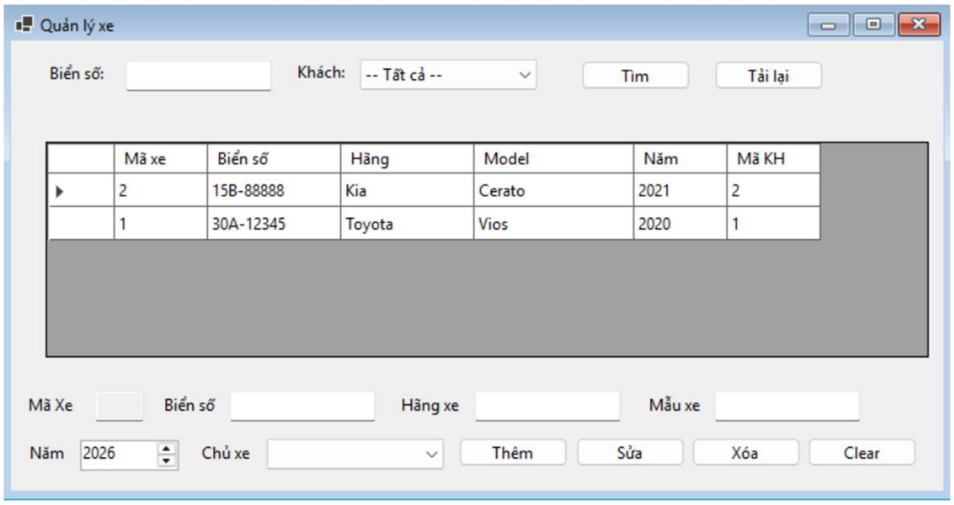
- Mục đích: Lưu trữ và tra cứu thông tin xe liên kết khách hàng.
- Thành phần chính:
    + Form thông tin xe (Biển số, VIN, Model, Năm, Km hiện tại).

+ Bảng danh sách xe, nút Thêm/Sửa/Xoá.

- Chức năng:

+ CRUD xe; thêm xe khi tạo khách hàng mới.

+ Tra cứu nhanh theo biển số/VIN.



Hình 8. Giao diện màn hình quản lý xe

4) Phiếu sửa chữa

- Mục đích: Tiếp nhận công việc, lập danh sách dịch vụ & phụ tùng dự kiến, theo dõi tiến trình.

- Thành phần chính:

+ Chọn Khách hàng/Vehicle; OrderNumber tự sinh.

+ Danh sách service items (mã dịch vụ, mô tả, giờ/đơn giá).

+ Danh sách parts (chọn SparePart, qty, đơn giá).

+ Chọn Mechanic, ngày dự kiến, trạng thái.

+ Nút Lưu, Lưu & Tạo báo giá, Gửi báo giá, Hủy.

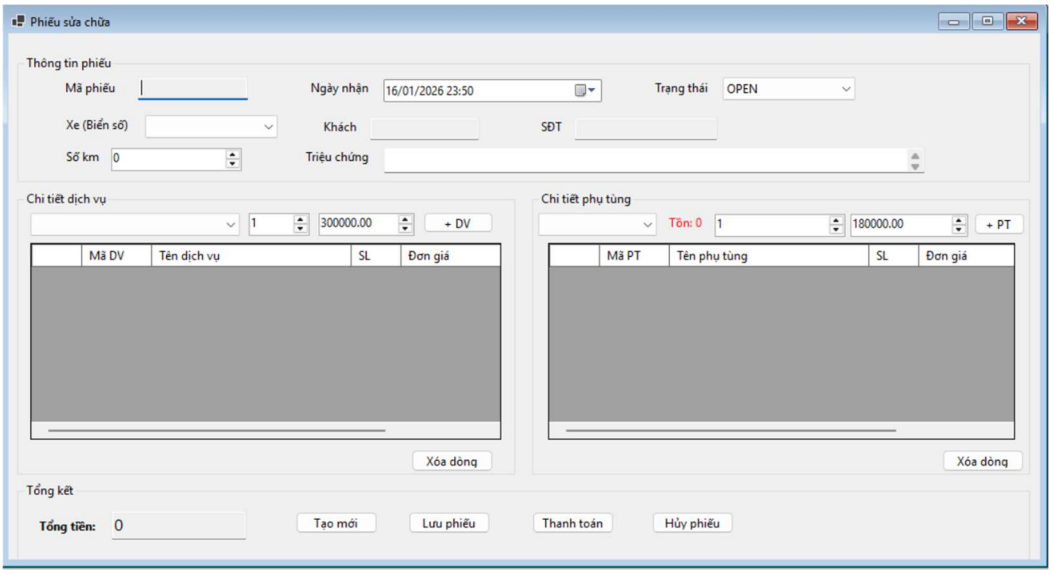
- Chức năng:

+ Tạo phiếu mới (sinh mã, lưu trạng thái New).

+ Thêm/xóa/sửa service item & part line.

+ Kiểm tra tồn kho trước khi xác nhận sử dụng phụ tùng.

+ Khi chuyển trạng thái thành Completed: tạo bản ghi sẵn sàng cho Invoice.



Hình 9. Giao diện phiếu sửa chữa

5) Kho phụ tùng (Inventory)

- Mục đích: Quản lý tồn kho, giá, nhập/xuất, điều chỉnh.

- Thành phần chính:

  + Bảng danh sách phụ tùng (Code, Name, Tồn kho, Giá nhập, Giá bán, ReorderLevel).

  + Form Nhập kho (PO), Xuất kho (manual), Điều chỉnh tồn.

  + Modal/Tab lịch sử giao dịch (InventoryTransaction).

- Chức năng:

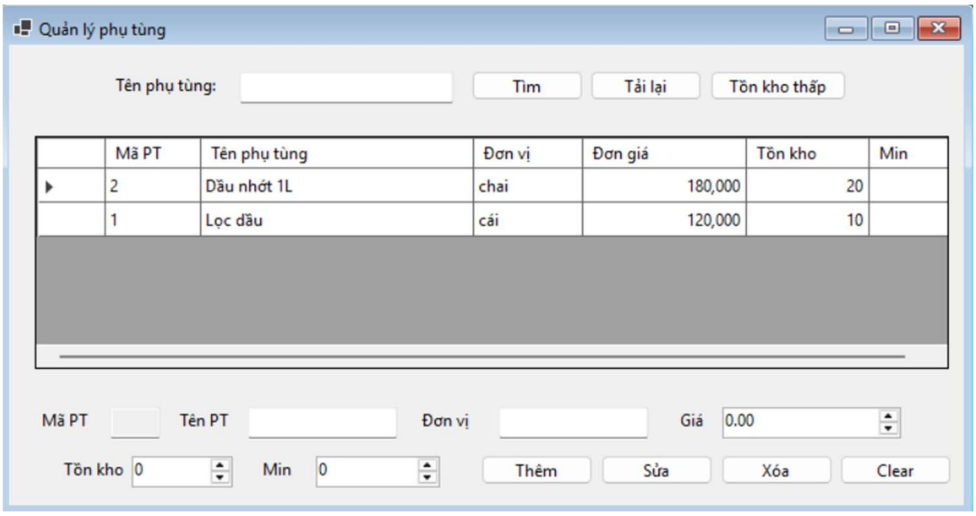
  + CRUD phụ tùng.

  + Tạo phiếu nhập kho (PO), nhận hàng và cập nhật tồn.

  + Xuất kho tự động khi xác nhận part usage trên ServiceOrder.

  + Điều chỉnh tồn (inventory adjustment) có lý do.

  + Cảnh báo & báo cáo phụ tùng dưới reorder level.



6) Dịch vụ

- Mục đích: Quản lý các loại dịch vụ tiêu chuẩn để tái sử dụng khi tạo phiếu.

- Thành phần chính:

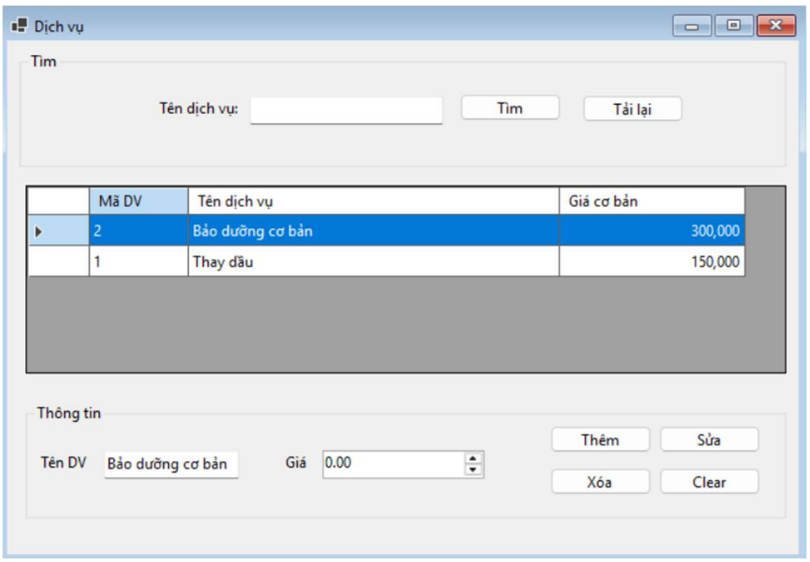
    + Form Thêm/Sửa dịch vụ (Code, Tên, Mô tả, Đơn giá, Thời gian chuẩn).

    + Bảng danh sách dịch vụ, tìm kiếm, import/export.

- Chức năng:

    + CRUD dịch vụ.

    + Gắn nhóm dịch vụ, nhãn, thẻ.



7) Báo cáo

- Mục đích: Trích xuất báo cáo doanh thu, tồn kho, hiệu suất nhân viên.

- Thành phần chính:

    + Bộ lọc: khoảng thời gian, nhân viên, loại dịch vụ.

    + Bảng/biểu đồ: Doanh thu theo ngày/tuần/tháng, Top phụ tùng tiêu thụ, Số phiếu theo trạng thái.

- Chức năng:

    + Tính toán tổng hợp, drill-down tới phiếu/hóa đơn.

    + Lưu báo cáo tuỳ chỉnh.