###### **TRƯỜNG ĐẠI HỌC PHENIKAA**

###### **KHOA CÔNG NGHỆ THÔNG TIN**

**-----&-----**

**A logo of a university

AI-generated content may be incorrect.**

###### **ĐỀ TÀI: Quản lý Showroo oto**

**Giảng viên hướng dẫn: TS.Vũ Quang Dũng**

Nhóm 1:

###### Nguyễn Trung Hiếu – 23013003

###### Nguyễn Trung Hiếu – 23011981

###### Vũ Hải Khoa – 23012825

###### Hoàng Việt Tiệp – 23016415

**Lớp học phần: Kĩ thuật phần mềm COUR01.**

###### Hà Nội, 06 - 2025.

MỤC LỤC

[LỜI MỞ ĐẦU 2](#_Toc201298913)

[PHẦN NỘI DUNG 3](#_Toc201298914)

[CHƯƠNG I: GIỚI THIỆU KỸ THUẬT/CÔNG NGHỆ 3](#_Toc201298915)

[1.1 Ngôn ngữ Java 3](#_Toc201298916)

[1.2 Java Swing 3](#_Toc201298917)

[CHƯƠNG II: GIỚI THIỆU BÀI TOÁN VÀ CÁC CHỨC NĂNG 4](#_Toc201298918)

[2.1 Bài toán 4](#_Toc201298919)

[2.2 Các chức năng dự định thực hiện 5](#_Toc201298920)

[2.2.1 Giao diện người dùng 5](#_Toc201298921)

[2.2.2 Giao diện quản trị dành cho quản lý showroom 5](#_Toc201298922)

[CHƯƠNG III: GIỚI THIỆU, TRIỂN KHAI VÀ PHÂN CÔNG NHIỆM VỤ 6](#_Toc201298923)

[3.1 Giới thiệu ứng dụng 6](#_Toc201298924)

[3.1.1 Giao diện đăng nhập quản trị viên (Admin) 6](#_Toc201298925)

[3.1.2 Giao diện tra cứu thông tin xe cho khách hàng 6](#_Toc201298926)

[3.1.3 Giao diện quản lý danh sách xe 7](#_Toc201298927)

[3.1.4 Giao diện quản lý khách hàng 7](#_Toc201298928)

[3.1.5 Giao diện quản lý hợp đồng bán xe 7](#_Toc201298929)

[3.1.6 Giao diện thống kê doanh số/bán hàng 8](#_Toc201298930)

[3.1.7 Kết nối cơ sở dữ liệu SQLiteStudio 8](#_Toc201298931)

[3.1.7.1 Cơ sở dữ liệu 8](#_Toc201298932)

[3.1.7.2 Các câu lệnh SQL Các thao tác dữ liệu được thực hiện thông qua các lớp DAO (Data Access Object) tương ứng, sử dụng JDBC API để tương tác với SQLite Studio. 8](#_Toc201298933)

[3.2 Triển khai ứng dụng 10](#_Toc201298934)

[3.2.1 Chức năng đăng nhập 10](#_Toc201298935)

[3.2.2 Chức năng thêm, sửa, xóa xe 10](#_Toc201298936)

[3.2.3 Chức năng bán xe, lập hợp đồng 10](#_Toc201298937)

[3.2.4 Chức năng quản lý khách hàng và thống kê 10](#_Toc201298938)

[3.3 Phân công nhiệm vụ thành viên nhóm 10](#_Toc201298939)

[4.Đặc tả và thiết kế 10](#_Toc201298940)

[4.1Use Case + Mô tả 10](#_Toc201298941)

[4.1.1. Đăng nhập hệ thống (Login to System) 10](#_Toc201298942)

[4.1.2. Quản lý danh mục xe (Manage Car Inventory) 11](#_Toc201298943)

[4.1.3. Quản lý khách hàng (Manage Customers) 12](#_Toc201298944)

[CHƯƠNG IV :KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 12](#_Toc201298945)

[1.Tóm tắt kết quả đạt được 12](#_Toc201298946)

[2.Hạn chế và thách thức 13](#_Toc201298947)

[3.Hướng phát triển trong tương lai 13](#_Toc201298948)

[LỜI KẾT 16](#_Toc201298949)

# LỜI MỞ ĐẦU

Ngành công nghiệp ô tô toàn cầu đang chứng kiến sự tăng trưởng mạnh mẽ và sự thay đổi không ngừng trong hành vi tiêu dùng. Điều này đặt ra yêu cầu ngày càng cao về khả năng quản lý linh hoạt, hiệu quả và chuyên nghiệp đối với các showroom ô tô. Từ việc quản lý kho xe, thông tin khách hàng, đến quy trình bán hàng và báo cáo doanh thu, mỗi khía cạnh đều đòi hỏi sự chính xác và kịp thời. Nếu không có một hệ thống quản lý bài bản, các showroom dễ dàng đối mặt với các vấn đề như thất thoát thông tin, khó khăn trong việc theo dõi tồn kho, sai sót trong hợp đồng, và mất thời gian trong việc tạo báo cáo thủ công. Những thách thức này không chỉ ảnh hưởng đến hiệu quả vận hành mà còn tác động trực tiếp đến doanh thu và khả năng cạnh tranh của showroom trên thị trường.

Nhận thức được những khó khăn đó, dự án "Phần mềm quản lý showroom ô tô" được khởi xướng nhằm xây dựng một giải pháp phần mềm toàn diện, đáp ứng nhu cầu quản lý chuyên nghiệp cho các showroom. Mục tiêu của dự án là cung cấp một công cụ mạnh mẽ, giúp tự động hóa các quy trình quản lý, giảm thiểu sai sót, tối ưu hóa nguồn lực và cải thiện trải nghiệm khách hàng. Phần mềm này hứa hẹn sẽ mang lại hiệu quả vượt trội trong việc quản lý xe, khách hàng, hợp đồng và doanh thu, góp phần nâng cao năng lực cạnh tranh và phát triển bền vững cho các showroom ô tô trong tương lai.

# PHẦN NỘI DUNG

# CHƯƠNG I: GIỚI THIỆU KỸ THUẬT/CÔNG NGHỆ

Để đảm bảo tính ổn định, bảo mật và khả năng mở rộng của hệ thống, chúng tôi đã lựa chọn các công nghệ nền tảng phổ biến và được cộng đồng hỗ trợ rộng rãi. Việc lựa chọn kỹ thuật phù hợp không chỉ giúp tăng tốc độ phát triển mà còn đảm bảo chất lượng và hiệu suất của ứng dụng trong dài hạn.

## 1.1 Ngôn ngữ Java

Java là một trong những ngôn ngữ lập trình phổ biến nhất thế giới, được biết đến với đặc tính "Write Once, Run Anywhere" (Viết một lần, chạy mọi nơi) nhờ vào máy ảo Java (JVM). Đối với dự án phần mềm quản lý showroom ô tô, Java được lựa chọn vì những lý do sau:

* **Tính đa nền tảng**: Mã Java có thể chạy trên nhiều hệ điều hành khác nhau như Windows, macOS, Linux mà không cần sửa đổi, giúp ứng dụng linh hoạt và dễ triển khai.
* **Hướng đối tượng**: Java là ngôn ngữ lập trình hướng đối tượng thuần túy, cho phép thiết kế các module code rõ ràng, dễ bảo trì và mở rộng. Điều này đặc biệt quan trọng đối với một hệ thống quản lý phức tạp với nhiều đối tượng (xe, khách hàng, hợp đồng).
* **Bảo mật**: Java có các tính năng bảo mật tích hợp, giúp bảo vệ dữ liệu và ứng dụng khỏi các mối đe dọa. Đây là yếu tố then chốt khi xử lý các thông tin nhạy cảm như dữ liệu khách hàng và hợp đồng.
* **Thư viện phong phú**: Java sở hữu một hệ sinh thái thư viện và framework khổng lồ, hỗ trợ hầu hết mọi tác vụ phát triển, từ kết nối cơ sở dữ liệu đến xây dựng giao diện người dùng. Điều này giúp đẩy nhanh quá trình phát triển và giảm thiểu công sức code thủ công.
* **Cộng đồng lớn và tài liệu đa dạng**: Với cộng đồng lập trình viên Java đông đảo, việc tìm kiếm hỗ trợ, giải pháp cho các vấn đề kỹ thuật và tài liệu học tập trở nên dễ dàng, đảm bảo dự án có thể được phát triển và duy trì hiệu quả.

## 1.2 Java Swing

Java Swing là một bộ công cụ phát triển giao diện người dùng đồ họa (GUI) được xây dựng trên nền tảng Java Foundation Classes (JFC). Mặc dù có các lựa chọn GUI hiện đại hơn, Swing vẫn là một lựa chọn mạnh mẽ và đáng tin cậy cho các ứng dụng desktop, đặc biệt là với dự án này:

* **Đa dạng thành phần GUI**: Swing cung cấp một bộ sưu tập phong phú các thành phần giao diện như nút, nhãn, hộp văn bản, bảng (JTable), cây (JTree), thanh cuộn, và các hộp thoại. Các thành phần này có thể được tùy chỉnh cao, cho phép tạo ra giao diện người dùng trực quan và hấp dẫn.
* **Tính độc lập nền tảng**: Tương tự như Java, Swing cũng có khả năng chạy trên nhiều hệ điều hành mà không phụ thuộc vào giao diện đồ họa gốc của hệ điều hành đó. Điều này đảm bảo rằng giao diện của phần mềm sẽ nhất quán trên mọi môi trường triển khai.
* **Mô hình MVC linh hoạt**: Swing cho phép triển khai kiến trúc Model-View-Controller (MVC) một cách dễ dàng, giúp tách biệt logic nghiệp vụ khỏi giao diện người dùng. Điều này cải thiện khả năng bảo trì và kiểm thử của ứng dụng.
* **Kiểm soát cao đối với giao diện**: Swing cung cấp cho nhà phát triển quyền kiểm soát chi tiết về cách các thành phần được vẽ và tương tác, cho phép tạo ra các giao diện phức tạp và đáp ứng các yêu cầu thiết kế cụ thể.
* **Đơn giản trong việc tích hợp với Java**: Là một phần của hệ sinh thái Java, Swing dễ dàng tích hợp với các thư viện Java khác, bao gồm cả các thư viện kết nối cơ sở dữ liệu như JDBC, giúp việc phát triển toàn diện ứng dụng trở nên thuận tiện.

# CHƯƠNG II: GIỚI THIỆU BÀI TOÁN VÀ CÁC CHỨC NĂNG

## 2.1 Bài toán

Quản lý một showroom ô tô là một công việc phức tạp, bao gồm nhiều nghiệp vụ khác nhau từ quản lý hàng tồn kho, thông tin khách hàng, quy trình bán hàng, đến việc theo dõi doanh thu và hiệu suất kinh doanh. Hiện tại, nhiều showroom vẫn đang sử dụng các phương pháp thủ công hoặc các công cụ không chuyên biệt (như bảng tính Excel) để quản lý các hoạt động này. Điều này dẫn đến một số vấn đề nghiêm trọng:

* **Mất mát và sai sót dữ liệu**: Việc nhập liệu thủ công hoặc lưu trữ phân tán dễ gây ra sai sót, trùng lặp thông tin và khó khăn trong việc đồng bộ hóa dữ liệu.
* **Khó khăn trong quản lý tồn kho**: Không có hệ thống theo dõi tồn kho tự động dẫn đến việc không nắm bắt được số lượng xe còn lại, gây khó khăn trong việc lên kế hoạch nhập hàng và tư vấn khách hàng.
* **Thiếu thông tin tổng quan về khách hàng**: Không có hồ sơ khách hàng tập trung, gây trở ngại cho việc chăm sóc khách hàng cá nhân hóa, theo dõi lịch sử mua hàng và xây dựng các chương trình khuyến mãi phù hợp.
* **Quy trình bán hàng không hiệu quả**: Việc lập hợp đồng, hóa đơn thủ công tốn thời gian, dễ sai sót và khó theo dõi tiến độ.
* **Khó khăn trong thống kê và báo cáo**: Việc tổng hợp dữ liệu doanh thu, số lượng xe bán ra để tạo báo cáo định kỳ mất rất nhiều thời gian và công sức, đồng thời dễ thiếu chính xác, ảnh hưởng đến việc ra quyết định kinh doanh.
* **Thiếu kiểm soát và bảo mật**: Dữ liệu quan trọng không được bảo vệ an toàn, dễ bị truy cập trái phép hoặc mất mát.

Mục tiêu của dự án là giải quyết các vấn đề trên bằng cách xây dựng một hệ thống phần mềm quản lý showroom ô tô toàn diện, tin cậy và dễ sử dụng.

## 2.2 Các chức năng dự định thực hiện

Hệ thống được thiết kế để hỗ trợ nhiều đối tượng người dùng với các vai trò và quyền hạn khác nhau, đảm bảo mỗi đối tượng có thể thực hiện công việc của mình một cách hiệu quả nhất.

## 2.2.1 Giao diện người dùng

Giao diện người dùng (UI) được thiết kế nhằm mang lại trải nghiệm trực quan, dễ hiểu và dễ thao tác cho tất cả các đối tượng. Các giao diện này sẽ đảm bảo tính nhất quán và rõ ràng, giúp người dùng mới dễ dàng làm quen với hệ thống. Các thành phần chính bao gồm:

* **Giao diện đăng nhập và đăng ký**: Cho phép người dùng đăng nhập vào hệ thống với tài khoản đã có hoặc đăng ký tài khoản mới (dành cho khách hàng hoặc nhân viên được cấp phép).
* **Giao diện tra cứu thông tin xe cho khách hàng**: Khách hàng có thể tìm kiếm, xem chi tiết các thông tin về xe như tên xe, giá, màu sắc, thông số kỹ thuật, tình trạng tồn kho mà không cần đăng nhập. Giao diện này sẽ cung cấp bộ lọc và chức năng tìm kiếm để khách hàng dễ dàng tìm thấy xe mong muốn.
* **Giao diện xem thông tin cá nhân**: Người dùng (khách hàng, nhân viên) có thể xem và cập nhật thông tin cá nhân của mình.
* **Giao diện xem lịch sử giao dịch (dành cho khách hàng)**: Khách hàng có thể xem lại các giao dịch mua xe đã thực hiện.

## 2.2.2 Giao diện quản trị dành cho quản lý showroom

Giao diện quản trị được thiết kế riêng cho người quản lý (Admin) và nhân viên showroom (Staff), cung cấp đầy đủ các công cụ để quản lý hoạt động kinh doanh:

* **Quản lý danh mục xe**: Cho phép thêm xe mới vào danh mục, cập nhật thông tin xe (giá, số lượng, mô tả), và xóa xe không còn kinh doanh. Chức năng tìm kiếm và lọc xe sẽ giúp việc quản lý trở nên dễ dàng hơn.
* **Quản lý khách hàng**: Cho phép thêm mới khách hàng, chỉnh sửa thông tin khách hàng, và xóa khách hàng khỏi hệ thống. Lịch sử giao dịch của khách hàng cũng sẽ được lưu trữ và truy cập tại đây.
* **Quản lý hợp đồng bán xe**: Hỗ trợ quy trình tạo, chỉnh sửa, xem và hủy bỏ các hợp đồng mua bán xe. Mỗi hợp đồng sẽ bao gồm thông tin xe, khách hàng, giá cả, phương thức thanh toán và các điều khoản liên quan.
* **Quản lý nhân sự (User Management)**: Cho phép quản lý tài khoản người dùng, bao gồm thêm nhân viên mới (staff), phân quyền (admin/staff), thay đổi mật khẩu và xóa tài khoản.
* **Thống kê doanh số/bán hàng**: Cung cấp các báo cáo và biểu đồ trực quan về doanh số bán hàng theo thời gian, số lượng xe bán ra theo loại, tổng doanh thu. Chức năng này giúp quản lý đưa ra các quyết định kinh doanh chiến lược.
* **Quản lý kho hàng**: Theo dõi số lượng xe tồn kho, cảnh báo khi số lượng xe thấp hơn ngưỡng quy định để quản lý có kế hoạch nhập hàng kịp thời.

# CHƯƠNG III: GIỚI THIỆU, TRIỂN KHAI VÀ PHÂN CÔNG NHIỆM VỤ

## 3.1 Giới thiệu ứng dụng

Ứng dụng quản lý showroom ô tô được thiết kế với kiến trúc phân lớp, tách biệt rõ ràng giữa giao diện người dùng (View), logic nghiệp vụ (Controller/Model), và lớp truy cập dữ liệu (DAO). Điều này giúp tăng cường khả năng bảo trì, mở rộng và kiểm thử hệ thống.

## 3.1.1 Giao diện đăng nhập quản trị viên (Admin)

* **Mô tả**: Đây là điểm khởi đầu của ứng dụng, cho phép người dùng nhập tên đăng nhập và mật khẩu để truy cập vào hệ thống.
* **Thiết kế**: Giao diện LoginView sẽ bao gồm các trường nhập liệu cho tên người dùng và mật khẩu, cùng với nút "Đăng nhập" và "Đăng ký".
* **Logic xử lý**: Khi người dùng nhấn nút "Đăng nhập", hệ thống sẽ gửi thông tin này đến UserDAO để xác thực. Nếu thông tin hợp lệ, dựa vào role (vai trò) của người dùng, hệ thống sẽ điều hướng đến giao diện AdminDashboardView (nếu là quản trị viên) hoặc MainDashboardView (nếu là nhân viên), hoặc CustomerDashboardView (dự kiến sẽ phát triển). Đối với trường hợp admin, AdminDashboardView sẽ cung cấp các tab để quản lý xe, khách hàng và người dùng.

## 3.1.2 Giao diện tra cứu thông tin xe cho khách hàng

* **Mô tả**: Giao diện này cung cấp cho khách hàng khả năng tìm kiếm và xem thông tin chi tiết về các loại xe có sẵn trong showroom. Khách hàng không cần phải đăng nhập để sử dụng chức năng này.
* **Thiết kế**: Giao diện này sẽ là một phần của MainDashboardView (hoặc một view công khai khác), hiển thị danh sách xe dưới dạng bảng (JTable). Các trường tìm kiếm theo tên, hãng, hoặc khoảng giá sẽ được cung cấp. Khi chọn một xe, thông tin chi tiết sẽ được hiển thị trong một cửa sổ pop-up hoặc một panel riêng.

## 3.1.3 Giao diện quản lý danh sách xe

* **Mô tả**: Đây là chức năng cốt lõi cho phép quản trị viên và nhân viên quản lý thông tin xe trong kho.
* **Thiết kế**: CarManagementPanel sẽ hiển thị danh sách xe trong một JTable. Các nút "Thêm", "Sửa", "Xóa" sẽ được đặt cạnh bảng để thực hiện các thao tác quản lý.
  + **Thêm xe**: Khi nhấn "Thêm", AddCarDialog sẽ xuất hiện, cho phép nhập các thông tin như car\_name, manufacturer, model, year, color, price, quantity, description. Dữ liệu sẽ được thêm vào cơ sở dữ liệu thông qua CarDAO.
  + **Sửa xe**: Khi chọn một dòng xe và nhấn "Sửa", EditCarDialog sẽ hiển thị thông tin hiện tại của xe đó để chỉnh sửa. Dữ liệu sẽ được cập nhật thông qua CarDAO.
  + **Xóa xe**: Khi chọn một dòng xe và nhấn "Xóa", một hộp thoại xác nhận sẽ hiện ra trước khi xóa xe khỏi cơ sở dữ liệu thông qua CarDAO.

## 3.1.4 Giao diện quản lý khách hàng

* **Mô tả**: Cho phép quản lý thông tin khách hàng một cách hiệu quả.
* **Thiết kế**: CustomerManagementPanel sẽ hiển thị danh sách khách hàng trong một JTable. Các nút "Thêm", "Sửa", "Xóa" sẽ được cung cấp.
  + **Thêm khách hàng**: AddEditCustomerDialog được sử dụng để nhập thông tin khách hàng mới (name, phone, email, address). Dữ liệu được lưu thông qua CustomerDAO.
  + **Sửa khách hàng**: Khi chọn một khách hàng và nhấn "Sửa", AddEditCustomerDialog sẽ được sử dụng để cập nhật thông tin.
  + **Xóa khách hàng**: Xóa thông tin khách hàng khỏi cơ sở dữ liệu thông qua CustomerDAO.

## 3.1.5 Giao diện quản lý hợp đồng bán xe

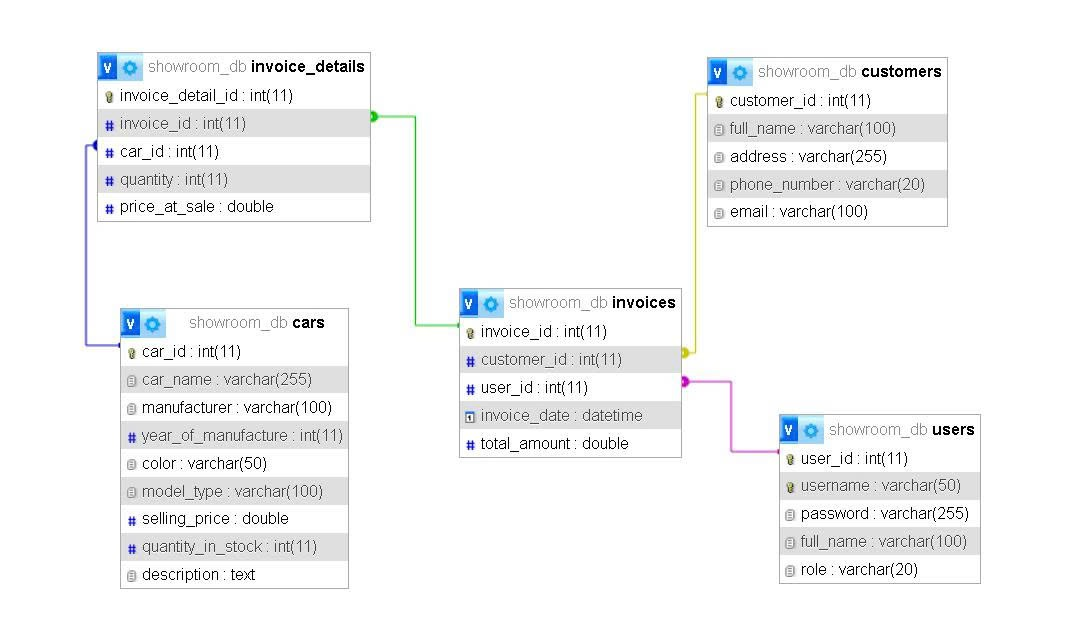
* **Mô tả**: Đây là chức năng quan trọng để ghi nhận và quản lý các giao dịch mua bán xe.
* **Thiết kế**: Một panel riêng sẽ được tạo để quản lý hóa đơn (Invoice). Chức năng này sẽ cho phép:
  + Tạo hóa đơn mới: Người dùng sẽ chọn xe, chọn khách hàng (hoặc thêm khách hàng mới ngay tại chỗ), nhập số lượng, phương thức thanh toán. Hệ thống sẽ tự động tính toán tổng tiền.
  + Xem danh sách hóa đơn: Hiển thị tất cả các hóa đơn đã tạo, bao gồm thông tin về xe, khách hàng, ngày bán, tổng tiền.
  + Xem chi tiết hóa đơn: Khi chọn một hóa đơn, hệ thống sẽ hiển thị chi tiết các mục đã bán trong hóa đơn đó.
* **Logic xử lý**: Chức năng này sẽ tương tác với InvoiceDAO để lưu trữ và truy xuất dữ liệu hóa đơn (Invoices) và chi tiết hóa đơn (InvoiceDetail). Quan trọng hơn, sau khi tạo hóa đơn thành công, số lượng xe trong kho (Car) phải được cập nhật giảm đi.

## 3.1.6 Giao diện thống kê doanh số/bán hàng

* **Mô tả**: Cung cấp cái nhìn tổng quan về hiệu suất kinh doanh của showroom.
* **Thiết kế**: Một panel chuyên dụng sẽ hiển thị các biểu đồ (ví dụ: biểu đồ cột, biểu đồ tròn) và bảng số liệu tóm tắt. Các lựa chọn lọc theo thời gian (tháng, quý, năm) hoặc theo loại xe sẽ được cung cấp.
* **Dữ liệu**: Dữ liệu cho phần này sẽ được lấy từ bảng Invoices và InvoiceDetail thông qua các truy vấn tổng hợp từ InvoiceDAO.

## 3.1.7 Kết nối cơ sở dữ liệu SQLiteStudio

## 3.1.7.1 Cơ sở dữ liệu



* **Hệ quản trị cơ sở dữ liệu**: MySQL được chọn vì tính phổ biến, hiệu suất cao, và khả năng tương thích tốt với Java JDBC.
* **Tên cơ sở dữ liệu**: showroom\_db
* **Thông tin kết nối**:
  + URL = "jdbc:sqlite:showroom\_db.db";
  + Class.forName("org.sqlite.JDBC"); // Trình điều khiển cho SQLite
* **Cấu trúc bảng (Schema)**:
  + **Users**: Lưu trữ thông tin người dùng hệ thống (admin, staff, customer). Các trường: user\_id (PK), username, password, role.
  + **Cars**: Lưu trữ thông tin chi tiết về các mẫu xe. Các trường: car\_id (PK), car\_name, manufacturer, model, year, color, price, quantity, description.
  + **Customers**: Lưu trữ thông tin khách hàng. Các trường: customer\_id (PK), name, phone, email, address.
  + **Invoices**: Lưu trữ thông tin về các hóa đơn bán hàng. Các trường: invoice\_id (PK), customer\_id (FK), user\_id (FK - người tạo hóa đơn), invoice\_date, total\_amount.
  + **InvoiceDetail**: Lưu trữ chi tiết các xe trong mỗi hóa đơn. Các trường: invoice\_detail\_id (PK), invoice\_id (FK), car\_id (FK), quantity, price\_at\_sale.
* **Lớp kết nối**: DatabaseConnection.java chịu trách nhiệm quản lý kết nối đến cơ sở dữ liệu, cung cấp các phương thức để mở và đóng kết nối.

## 3.1.7.2 Các câu lệnh SQL Các thao tác dữ liệu được thực hiện thông qua các lớp DAO (Data Access Object) tương ứng, sử dụng JDBC API để tương tác với SQLite Studio.

* **CarDAO.java**:
  + getAllCars(): SELECT \* FROM Cars
  + addCar(Car car): INSERT INTO Cars (car\_name, manufacturer, model, year, color, price, quantity, description) VALUES (?, ?, ?, ?, ?, ?, ?, ?)
  + updateCar(Car car): UPDATE Cars SET car\_name=?, manufacturer=?, model=?, year=?, color=?, price=?, quantity=?, description=? WHERE car\_id=?
  + deleteCar(int carId): DELETE FROM Cars WHERE car\_id=?
  + getCarById(int carId): SELECT \* FROM Cars WHERE car\_id=?
  + updateCarQuantity(int carId, int newQuantity): UPDATE Cars SET quantity=? WHERE car\_id=?
* **CustomerDAO.java**:
  + getAllCustomers(): SELECT \* FROM customers
  + addCustomer(Customers customer): INSERT INTO customers (name, phone, email, address) VALUES (?, ?, ?, ?)
  + updateCustomer(Customers customer): UPDATE customers SET name=?, phone=?, email=?, address=? WHERE customer\_id=?
  + deleteCustomer(int customerId): DELETE FROM customers WHERE customer\_id=?
* **UserDAO.java**:
  + checkLogin(String username, String password): SELECT \* FROM Users WHERE username=? AND password=?
  + registerUser(User user): INSERT INTO Users (username, password, role) VALUES (?, ?, ?)
  + isUsernameExists(String username): SELECT COUNT(\*) FROM Users WHERE username=?
  + getAllUsers(): SELECT \* FROM Users
  + deleteUser(int userId): DELETE FROM Users WHERE user\_id=?
  + updateUserRole(int userId, String newRole): UPDATE Users SET role=? WHERE user\_id=?
* **InvoiceDAO.java**:
  + createInvoice(Invoices invoice, List<InvoiceDetail> details): Thực hiện giao dịch (transaction) bao gồm:
    1. INSERT INTO Invoices (customer\_id, user\_id, invoice\_date, total\_amount) VALUES (?, ?, ?, ?) và lấy invoice\_id vừa tạo.
    2. Với mỗi InvoiceDetail: INSERT INTO InvoiceDetail (invoice\_id, car\_id, quantity, price\_at\_sale) VALUES (?, ?, ?, ?)
    3. Cập nhật số lượng xe tồn kho trong bảng Cars cho từng xe được bán.

## 3.2 Triển khai ứng dụng

Quá trình triển khai ứng dụng được thực hiện theo từng chức năng cụ thể, đảm bảo mỗi module hoạt động độc lập và hiệu quả trước khi tích hợp vào hệ thống tổng thể.

## 3.2.1 Chức năng đăng nhập

* LoginView.java là giao diện chính cho phép người dùng nhập thông tin đăng nhập.
* Khi người dùng nhấn nút "Đăng nhập", phương thức checkLogin() của UserDAO sẽ được gọi để xác thực thông tin với cơ sở dữ liệu.
* Nếu đăng nhập thành công, tùy thuộc vào vai trò (admin, staff, customer) được trả về từ cơ sở dữ liệu, ứng dụng sẽ chuyển hướng đến AdminDashboardView hoặc MainDashboardView (hoặc các view tương ứng khác).
* Giao diện RegisterView.java cho phép người dùng mới đăng ký tài khoản.

## 3.2.2 Chức năng thêm, sửa, xóa xe

* Chức năng này được quản lý chủ yếu trong CarManagementPanel.java, là một phần của AdminDashboardView hoặc MainDashboardView (dành cho nhân viên).
* **Thêm xe**: AddCarDialog.java là một cửa sổ thoại (dialog) riêng biệt để nhập dữ liệu xe mới. Sau khi nhập, dữ liệu được chuyển đến CarDAO.addCar() để lưu vào cơ sở dữ liệu.
* **Sửa xe**: EditCarDialog.java cũng là một cửa sổ thoại tương tự AddCarDialog, nhưng được khởi tạo với dữ liệu của xe cần chỉnh sửa. Dữ liệu sau khi sửa sẽ được gửi đến CarDAO.updateCar().
* **Xóa xe**: Khi người dùng chọn một xe trên bảng và nhấn nút xóa, một hộp thoại xác nhận sẽ xuất hiện. Nếu người dùng đồng ý, CarDAO.deleteCar() sẽ được gọi để xóa bản ghi khỏi cơ sở dữ liệu.

## 3.2.3 Chức năng bán xe, lập hợp đồng

* Chức năng này sẽ được phát triển trong một panel riêng biệt (chưa có trong các file đã cung cấp nhưng được đề cập trong báo cáo).
* Quy trình sẽ bao gồm việc chọn xe từ danh sách, chọn khách hàng hiện có hoặc thêm khách hàng mới.
* Hệ thống sẽ tạo một đối tượng Invoices và một hoặc nhiều đối tượng InvoiceDetail.
* Phương thức createInvoice() của InvoiceDAO sẽ thực hiện giao dịch lưu hóa đơn và chi tiết hóa đơn, đồng thời cập nhật số lượng tồn kho của xe trong bảng Cars thông qua CarDAO.updateCarQuantity(). Việc sử dụng giao dịch đảm bảo tính toàn vẹn của dữ liệu (nếu có lỗi xảy ra trong quá trình lưu, tất cả các thay đổi sẽ được rollback).

## 3.2.4 Chức năng quản lý khách hàng và thống kê

* **Quản lý khách hàng**: Được thực hiện trong CustomerManagementPanel.java. Các thao tác thêm, sửa, xóa khách hàng sử dụng AddEditCustomerDialog.java và tương tác với CustomerDAO.java.
* **Thống kê**: Một panel riêng sẽ được phát triển để hiển thị các báo cáo thống kê doanh số. Dữ liệu sẽ được truy vấn từ InvoiceDAO và hiển thị dưới dạng bảng hoặc biểu đồ bằng các thư viện đồ họa Java (ví dụ: JFreeChart, nếu được tích hợp).

## 3.3 Phân công nhiệm vụ thành viên nhóm

Sự phân công nhiệm vụ rõ ràng là yếu tố then chốt để đảm bảo tiến độ và chất lượng dự án. Mỗi thành viên chịu trách nhiệm về một hoặc nhiều module, đồng thời có sự phối hợp chặt chẽ để đảm bảo tính đồng bộ của hệ thống.

* **Nguyễn Trung Hiếu (23013003)**:
  + Chịu trách nhiệm chính trong phát triển các chức năng nghiệp vụ cốt lõi, bao gồm logic xử lý cho các module quản lý xe và quản lý hóa đơn.
  + Thiết kế và triển khai các lớp Model và Controller cho các tính năng chính.
  + Đảm bảo tính toàn vẹn dữ liệu trong các giao dịch.
  + Hỗ trợ tích hợp các module.
* **Vũ Hải Khoa (23012825)**:
  + Tập trung vào thiết kế và triển khai giao diện người dùng (UI) cho toàn bộ ứng dụng, sử dụng Java Swing.
  + Đảm bảo giao diện trực quan, thân thiện và dễ sử dụng.
  + Xử lý các sự kiện tương tác trên giao diện người dùng (ActionListener, MouseListener...).
  + Thiết kế các dialog như AddCarDialog, EditCarDialog, AddEditCustomerDialog.
* **Nguyễn Trung Hiếu (23011981)**:
  + Đảm nhiệm phần kết nối cơ sở dữ liệu và phát triển các lớp Data Access Object (DAO) cho tất cả các bảng (Car, Customer, User, Invoice, InvoiceDetail).
  + Viết các câu lệnh SQL để truy vấn, thêm, sửa, xóa dữ liệu.
  + Đảm bảo hiệu suất và bảo mật của các thao tác cơ sở dữ liệu.
  + Xử lý các ngoại lệ liên quan đến kết nối và thao tác CSDL.
* **Hoàng Việt Tiệp (23016415)**:
  + Hỗ trợ triển khai các chức năng đã được thiết kế.
  + Thực hiện kiểm thử đơn vị (Unit Test) và kiểm thử tích hợp (Integration Test) cho các module.
  + Viết tài liệu kỹ thuật và hướng dẫn sử dụng cho các chức năng được giao.
  + Đảm bảo tính ổn định và khắc phục lỗi phát sinh trong quá trình phát triển.

# 4.Đặc tả và thiết kế

## 4.1Use Case + Mô tả

Phần mềm Quản lý Showroom Ô tô được thiết kế để phục vụ nhiều đối tượng người dùng với các vai trò và quyền hạn khác nhau, nhằm đảm bảo mỗi tác nhân có thể thực hiện công việc của mình một cách hiệu quả nhất. Dưới đây là các Use Case chính của hệ thống, bao gồm mô tả chi tiết từng Use Case.

## 4.1.1. Đăng nhập hệ thống (Login to System)

* **Mô tả**: Cho phép tất cả các loại người dùng (Quản trị viên, Nhân viên, Khách hàng) truy cập vào hệ thống bằng cách cung cấp tên đăng nhập và mật khẩu của họ. Hệ thống sẽ xác thực thông tin và điều hướng người dùng đến giao diện phù hợp với vai trò của họ.
* **Tác nhân**: Người dùng (Admin, Staff, Customer).
* **Điều kiện tiên quyết**: Người dùng đã có tài khoản hợp lệ trong hệ thống.
* **Luồng sự kiện chính**:
  1. Người dùng mở ứng dụng và truy cập màn hình đăng nhập.
  2. Người dùng nhập tên đăng nhập và mật khẩu vào các trường tương ứng.
  3. Người dùng nhấn nút "Đăng nhập".
  4. Hệ thống gửi thông tin đăng nhập đến lớp UserDAO để xác thực.
  5. UserDAO kiểm tra thông tin trong cơ sở dữ liệu (Users table).
  6. Nếu thông tin hợp lệ:
     + Hệ thống xác định vai trò của người dùng (admin, staff, customer).
     + Nếu là admin, hiển thị AdminDashboardView.
     + Nếu là staff, hiển thị MainDashboardView (có thể được xem là giao diện chính cho nhân viên).
     + Nếu là customer, hiển thị CustomerView.
     + Hiển thị thông báo "Đăng nhập thành công!".
     + Đóng cửa sổ đăng nhập.
  7. Nếu thông tin không hợp lệ:
     + Hệ thống hiển thị thông báo lỗi "Sai tên đăng nhập hoặc mật khẩu".
     + Người dùng có thể thử lại.
* **Luồng sự kiện thay thế**:
  1. **Quên mật khẩu**: (Chưa triển khai) Người dùng có thể nhấp vào "Quên mật khẩu" để bắt đầu quy trình khôi phục.
  2. **Đăng ký tài khoản mới**: Người dùng có thể nhấp vào "Chưa có tài khoản? Đăng ký." để chuyển đến RegisterView.

## 4.1.2. Quản lý danh mục xe (Manage Car Inventory)

* **Mô tả**: Cho phép Quản trị viên và Nhân viên thêm mới, cập nhật thông tin, và xóa các mẫu xe khỏi danh mục của showroom.
* **Tác nhân**: Admin, Staff.
* **Điều kiện tiên quyết**: Người dùng đã đăng nhập vào hệ thống với vai trò Admin hoặc Staff.
* **Luồng sự kiện chính (Thêm xe)**:
  1. Người dùng truy cập CarManagementPanel (hoặc AdminDashboardView/MainDashboardView).
  2. Người dùng nhấn nút "Thêm xe".
  3. AddCarDialog xuất hiện.
  4. Người dùng nhập thông tin chi tiết về xe (tên xe, hãng sản xuất, năm, màu sắc, kiểu dáng, giá bán, số lượng, mô tả).
  5. Người dùng nhấn nút "Lưu".
  6. Hệ thống thực hiện kiểm tra dữ liệu đầu vào (ví dụ: tên xe không trống, giá bán/số lượng là số dương).
  7. Nếu dữ liệu hợp lệ, tạo đối tượng Car.
  8. CarDAO được gọi để thêm xe vào cơ sở dữ liệu (Cars table).
  9. Hiển thị thông báo "Thêm xe mới thành công!".
  10. AddCarDialog đóng lại và danh sách xe trên bảng được làm mới.
* **Luồng sự kiện chính (Sửa xe)**:
  1. Người dùng chọn một chiếc xe từ bảng hiển thị.
  2. Người dùng nhấn nút "Sửa xe".
  3. Hệ thống lấy car\_id của xe đã chọn.
  4. CarDAO.getCarById() được gọi để truy xuất thông tin chi tiết của xe.
  5. EditCarDialog xuất hiện với thông tin hiện tại của xe được điền sẵn.
  6. Người dùng chỉnh sửa thông tin cần thiết.
  7. Người dùng nhấn nút "Lưu".
  8. Hệ thống thực hiện kiểm tra dữ liệu đầu vào.
  9. Tạo đối tượng Car với các thông tin đã cập nhật và id của xe.
  10. CarDAO.updateCar() được gọi để cập nhật thông tin xe trong cơ sở dữ liệu.
  11. Hiển thị thông báo "Cập nhật xe thành công!".
  12. EditCarDialog đóng lại và danh sách xe trên bảng được làm mới.
* **Luồng sự kiện chính (Xóa xe)**:
  1. Người dùng chọn một chiếc xe từ bảng hiển thị.
  2. Người dùng nhấn nút "Xóa xe".
  3. Một hộp thoại xác nhận hiện ra hỏi người dùng có chắc chắn muốn xóa hay không.
  4. Nếu người dùng xác nhận "Có":
     + Hệ thống lấy car\_id của xe đã chọn.
     + CarDAO.deleteCar() được gọi để xóa xe khỏi cơ sở dữ liệu.
     + Hiển thị thông báo "Đã xóa xe thành công!".
     + Danh sách xe trên bảng được làm mới.
  5. Nếu người dùng xác nhận "Không" hoặc đóng hộp thoại:
     + Thao tác xóa bị hủy bỏ.
* **Luồng sự kiện thay thế**:
  1. **Lỗi nhập liệu**: Nếu dữ liệu nhập không hợp lệ, hiển thị thông báo lỗi cụ thể (ví dụ: "Năm sản xuất phải là số!").
  2. **Lỗi CSDL**: Nếu có lỗi xảy ra trong quá trình tương tác với CSDL, hiển thị thông báo lỗi hệ thống.

## 4.1.3. Quản lý khách hàng (Manage Customers)

* **Mô tả**: Cho phép Quản trị viên và Nhân viên thêm, sửa, xóa thông tin khách hàng.
* **Tác nhân**: Admin, Staff.
* **Điều kiện tiên quyết**: Người dùng đã đăng nhập vào hệ thống với vai trò Admin hoặc Staff.
* **Luồng sự kiện chính (Thêm khách hàng)**:
  1. Người dùng truy cập CustomerManagementPanel.
  2. Người dùng nhấn nút "Thêm Khách Hàng".
  3. AddEditCustomerDialog xuất hiện với tiêu đề "Thêm khách hàng mới".
  4. Người dùng nhập thông tin khách hàng (họ tên, địa chỉ, số điện thoại, email).
  5. Người dùng nhấn nút "Lưu".
  6. Hệ thống kiểm tra dữ liệu đầu vào (họ tên và SĐT là bắt buộc).
  7. Nếu dữ liệu hợp lệ, tạo đối tượng Customers.
  8. CustomerDAO.addCustomer() được gọi để lưu khách hàng vào CSDL.
  9. Hiển thị thông báo "Thêm khách hàng mới thành công!".
  10. Dialog đóng và danh sách khách hàng trên bảng được làm mới.
* **Luồng sự kiện chính (Sửa khách hàng)**:
  1. Người dùng chọn một khách hàng từ bảng hiển thị.
  2. Người dùng nhấn nút "Sửa Thông Tin".
  3. Hệ thống lấy customer\_id của khách hàng đã chọn.
  4. Tạo đối tượng Customers tạm thời từ dữ liệu trên bảng (trong thực tế nên gọi DAO để lấy đầy đủ).
  5. AddEditCustomerDialog xuất hiện với tiêu đề "Sửa thông tin khách hàng" và thông tin hiện tại được điền sẵn.
  6. Người dùng chỉnh sửa thông tin cần thiết.
  7. Người dùng nhấn nút "Lưu".
  8. Hệ thống kiểm tra dữ liệu đầu vào.
  9. Cập nhật đối tượng Customers đang chỉnh sửa.
  10. CustomerDAO.updateCustomer() được gọi để cập nhật thông tin khách hàng trong CSDL.
  11. Hiển thị thông báo "Cập nhật thông tin thành công!".
  12. Dialog đóng và danh sách khách hàng trên bảng được làm mới.
* **Luồng sự kiện chính (Xóa khách hàng)**:
  1. Người dùng chọn một khách hàng từ bảng hiển thị.
  2. Người dùng nhấn nút "Xóa Khách Hàng".
  3. Một hộp thoại xác nhận hiện ra.
  4. Nếu người dùng xác nhận "Có":
     + Hệ thống lấy customer\_id của khách hàng đã chọn.
     + CustomerDAO.deleteCustomer() được gọi để xóa khách hàng khỏi CSDL.
     + Hiển thị thông báo "Xóa khách hàng thành công.".
     + Danh sách khách hàng trên bảng được làm mới.
  5. Nếu người dùng xác nhận "Không" hoặc đóng hộp thoại:
     + Thao tác xóa bị hủy bỏ.

**4.1.4. Quản lý nhân sự (User Management)**

* **Mô tả**: Dành cho Quản trị viên để thêm tài khoản nhân viên mới, xóa người dùng và cập nhật vai trò của người dùng hiện có (chỉ cho Staff/Admin).
* **Tác nhân**: Admin.
* **Điều kiện tiên quyết**: Người dùng đã đăng nhập với vai trò Admin.
* **Luồng sự kiện chính (Thêm nhân viên)**:
  1. Admin nhấn nút "Quản lí người dùng" trên AdminDashboardView.
  2. Một JFrame mới với UserManagementPanel được hiển thị.
  3. Admin nhấn nút "Thêm Nhân Viên".
  4. CreateStaffDialog xuất hiện.
  5. Admin nhập tên đăng nhập, mật khẩu và họ tên cho nhân viên mới.
  6. Admin nhấn nút "Lưu".
  7. Hệ thống kiểm tra dữ liệu (không trống, tên đăng nhập không trùng).
  8. Nếu hợp lệ, tạo đối tượng User với vai trò "staff".
  9. UserDAO.registerUser() được gọi để thêm tài khoản vào CSDL.
  10. Hiển thị thông báo "Tạo tài khoản nhân viên thành công!".
  11. CreateStaffDialog đóng và danh sách người dùng trên bảng được làm mới.
* **Luồng sự kiện chính (Xóa người dùng)**:
  1. Admin chọn một người dùng từ bảng hiển thị trong UserManagementPanel.
  2. Admin nhấn nút "Xoá người dùng".
  3. Hệ thống kiểm tra xem có phải tài khoản admin gốc không (không được xóa).
  4. Một hộp thoại xác nhận hiện ra.
  5. Nếu Admin xác nhận "Có":
     + Hệ thống lấy user\_id của người dùng đã chọn.
     + UserDAO.deleteUser() được gọi để xóa người dùng khỏi CSDL.
     + Hiển thị thông báo "Xóa người dùng thành công!".
     + Danh sách người dùng trên bảng được làm mới.
* **Luồng sự kiện chính (Cập nhật vai trò)**:
  1. Admin chọn một người dùng từ bảng hiển thị.
  2. Admin chọn vai trò mới từ ComboBox (cmbRole) (ví dụ: "staff", "admin").
  3. Admin nhấn nút "Cập nhật vai trò".
  4. Hệ thống kiểm tra (không được thay đổi vai trò admin gốc).
  5. Lấy user\_id và vai trò mới.
  6. UserDAO.updateUserRole() được gọi để cập nhật vai trò trong CSDL.
  7. Hiển thị thông báo "Cập nhật vai trò thành công!".
  8. Danh sách người dùng trên bảng được làm mới.

**4.1.5. Tra cứu thông tin xe cho khách hàng (View Car Information)**

* **Mô tả**: Cho phép khách hàng tìm kiếm và xem thông tin chi tiết về các mẫu xe có sẵn trong showroom.
* **Tác nhân**: Khách hàng (có thể chưa đăng nhập).
* **Điều kiện tiên quyết**: Không có.
* **Luồng sự kiện chính**:
  1. Khách hàng truy cập CustomerView (hoặc sau khi đăng nhập với vai trò customer).
  2. Hệ thống hiển thị danh sách các xe đang có trong kho (quantity\_in\_stock > 0) trong một JTable.
  3. Khách hàng có thể sử dụng trường tìm kiếm (txtSearch) để lọc xe theo tên, hãng, v.v..
  4. Khách hàng chọn một chiếc xe từ danh sách.
  5. Khách hàng nhấn nút "Xem chi tiết".
  6. Hệ thống lấy car\_id của xe đã chọn và gọi CarDAO.getCarById().
  7. Một hộp thoại JOptionPane hiển thị tất cả các thông tin chi tiết của xe (tên, hãng, năm, màu, kiểu dáng, giá, số lượng tồn, mô tả).

**4.1.6. Thống kê doanh số/bán hàng (Sales/Revenue Statistics)**

* **Mô tả**: Cung cấp các báo cáo và biểu đồ trực quan về doanh số bán hàng, số lượng xe bán ra, và tổng doanh thu theo thời gian.
* **Tác nhân**: Admin, Staff.
* **Điều kiện tiên quyết**: Người dùng đã đăng nhập với vai trò Admin hoặc Staff.
* **Luồng sự kiện chính**:
  1. Người dùng (Admin/Staff) nhấn nút "Thống kê" trên Dashboard.
  2. ThongKe dialog xuất hiện.
  3. Dialog hiển thị tổng doanh thu và tổng số hóa đơn bằng cách gọi InvoiceDAO.getTotalRevenue() và InvoiceDAO.getTotalInvoices().
  4. Dialog hiển thị bảng doanh thu theo tháng bằng cách gọi InvoiceDAO.getRevenueByMonth().
  5. Người dùng xem các số liệu thống kê.
  6. Người dùng nhấn nút "Đóng" để thoát.

**4.1.7. Chức năng bán xe, lập hợp đồng (Sell Car, Create Invoice)**

* **Mô tả**: Hỗ trợ quy trình tạo, ghi nhận các giao dịch mua bán xe và lập hóa đơn. Đồng thời, chức năng này sẽ cập nhật số lượng xe trong kho.
* **Tác nhân**: Admin, Staff.
* **Điều kiện tiên quyết**: Người dùng đã đăng nhập với vai trò Admin hoặc Staff.
* **Luồng sự kiện chính**:
  1. Người dùng truy cập giao diện bán xe (chưa có panel cụ thể trong các file hiện tại, nhưng được đề cập trong báo cáo).
  2. Người dùng chọn xe từ danh sách xe có sẵn.
  3. Người dùng chọn khách hàng hiện có hoặc thêm khách hàng mới (nếu khách hàng chưa có trong hệ thống).
  4. Người dùng nhập số lượng xe muốn bán.
  5. Hệ thống tự động tính toán tổng tiền dựa trên giá xe và số lượng.
  6. Người dùng xác nhận giao dịch.
  7. Hệ thống tạo một đối tượng Invoices và các đối tượng InvoiceDetail tương ứng.
  8. InvoiceDAO.createInvoice() được gọi, thực hiện một giao dịch CSDL để:
     + Chèn thông tin hóa đơn mới vào bảng Invoices.
     + Chèn chi tiết từng xe đã bán vào bảng InvoiceDetail (sử dụng invoice\_id vừa tạo).
     + Cập nhật giảm số lượng tồn kho của từng xe đã bán trong bảng Cars thông qua CarDAO.updateCarQuantity().
  9. Hiển thị thông báo "Tạo hóa đơn thành công!" hoặc tương tự.
* **Luồng sự kiện thay thế**:
  1. **Không đủ số lượng tồn kho**: Nếu số lượng xe yêu cầu vượt quá số lượng tồn kho, hệ thống sẽ hiển thị cảnh báo và không cho phép tạo hóa đơn.
  2. **Lỗi giao dịch**: Nếu có bất kỳ lỗi nào xảy ra trong quá trình tạo hóa đơn (ví dụ: mất kết nối CSDL), toàn bộ giao dịch sẽ được rollback để đảm bảo tính toàn vẹn dữ liệu.

# CHƯƠNG IV :KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 1.Tóm tắt kết quả đạt được

Dự án "Phần mềm quản lý showroom ô tô" đã hoàn thành việc xây dựng một hệ thống cơ bản nhưng đầy đủ các chức năng cần thiết để quản lý hiệu quả hoạt động của một showroom ô tô. Cụ thể, chúng tôi đã đạt được các kết quả sau:

* **Hệ thống đăng nhập và phân quyền**: Ứng dụng đã xây dựng thành công giao diện đăng nhập cho phép người dùng đăng nhập với các vai trò khác nhau (Admin, Staff). Chức năng đăng ký tài khoản cũng được triển khai để mở rộng khả năng quản lý người dùng.
* **Quản lý danh mục xe hiệu quả**: Phần mềm cho phép quản lý thông tin xe một cách linh hoạt, bao gồm các chức năng thêm, sửa, và xóa thông tin xe, giúp showroom dễ dàng cập nhật danh mục sản phẩm.
* **Quản lý thông tin khách hàng**: Hệ thống đã triển khai chức năng quản lý khách hàng, cho phép thêm, sửa, xóa thông tin khách hàng, tạo cơ sở dữ liệu khách hàng tập trung để hỗ trợ các hoạt động bán hàng và chăm sóc khách hàng sau này.
* **Xây dựng cơ sở dữ liệu mạnh mẽ**: Sử dụng MySQL làm hệ quản trị cơ sở dữ liệu, chúng tôi đã thiết kế và triển khai các bảng dữ liệu cho xe, khách hàng, người dùng, hóa đơn và chi tiết hóa đơn. Điều này đảm bảo tính nhất quán và toàn vẹn của dữ liệu trong suốt quá trình hoạt động.
* **Kết nối CSDL ổn định**: Lớp DatabaseConnection đã được xây dựng để quản lý kết nối an toàn và hiệu quả đến cơ sở dữ liệu MySQL.
* **Áp dụng công nghệ phù hợp**: Dự án đã sử dụng ngôn ngữ lập trình Java và bộ công cụ Java Swing để phát triển ứng dụng desktop, mang lại tính ổn định, bảo mật và khả năng tương thích đa nền tảng.

## 2.Hạn chế và thách thức

Trong quá trình phát triển dự án, nhóm đã gặp phải một số hạn chế và thách thức cần được xem xét cho các phiên bản tiếp theo:

* **Giao diện người dùng đơn giản**: Mặc dù Java Swing cung cấp các thành phần GUI cơ bản, giao diện hiện tại của ứng dụng vẫn còn khá đơn giản và chưa thực sự tối ưu về mặt trải nghiệm người dùng (UX) so với các ứng dụng web hoặc ứng dụng desktop hiện đại hơn. Việc tùy chỉnh sâu hơn để tạo ra một UI/UX thu hút và tiện lợi hơn đòi hỏi nhiều công sức.
* **Chức năng báo cáo và thống kê cơ bản**: Phần thống kê doanh số/bán hàng hiện tại chủ yếu tập trung vào việc hiển thị dữ liệu thô. Các công cụ phân tích nâng cao, biểu đồ động, và khả năng tạo báo cáo tùy chỉnh còn hạn chế.
* **Thiếu module quản lý hợp đồng hoàn chỉnh**: Mặc dù đã đề cập đến giao diện quản lý hợp đồng bán xe, chức năng này cần được phát triển chi tiết hơn với các luồng nghiệp vụ cụ thể như quản lý trạng thái hợp đồng, duyệt hợp đồng, in ấn hợp đồng, v.v.
* **Chưa tích hợp đầy đủ module quản lý nhân sự**: Chức năng quản lý người dùng hiện tại chủ yếu tập trung vào tài khoản admin và staff. Một module quản lý nhân viên chi tiết hơn với các thông tin cá nhân, lương, hiệu suất làm việc chưa được triển khai.
* **Chưa có cơ chế xử lý lỗi và log chi tiết**: Hệ thống còn thiếu các cơ chế ghi log lỗi và xử lý ngoại lệ một cách toàn diện, điều này có thể gây khó khăn trong việc debug và bảo trì khi ứng dụng gặp sự cố trong môi trường thực tế.
* **Không có tính năng sao lưu và phục hồi dữ liệu**: Đây là một chức năng quan trọng để bảo vệ dữ liệu, nhưng hiện tại chưa được tích hợp vào phần mềm.

## 3.Hướng phát triển trong tương lai

Để nâng cao giá trị và tính ứng dụng của phần mềm, nhóm đề xuất các hướng phát triển chính trong tương lai:

1. **Tích hợp module quản lý hợp đồng chi tiết**:
   * Phát triển giao diện và logic cho phép tạo, sửa, xem, và quản lý trạng thái của từng hợp đồng bán xe.
   * Hỗ trợ tạo hợp đồng mẫu và in ấn trực tiếp từ phần mềm.
   * Tích hợp các phương thức thanh toán linh hoạt (tiền mặt, chuyển khoản, trả góp).
2. **Nâng cao chức năng báo cáo và phân tích**:
   * Sử dụng thư viện đồ họa (ví dụ: JFreeChart) để tạo các biểu đồ trực quan (biểu đồ cột, biểu đồ tròn, biểu đồ đường) về doanh số bán hàng theo ngày/tháng/năm, loại xe, hãng xe, v.v..
   * Cung cấp các báo cáo tùy chỉnh, cho phép người dùng chọn các tiêu chí để tạo báo cáo riêng.
   * Phát triển các chỉ số hiệu suất chính (KPIs) và bảng điều khiển (dashboard) để quản lý có cái nhìn tổng quan nhanh chóng.
3. **Mở rộng module quản lý khách hàng (CRM)**:
   * Thêm tính năng quản lý lịch sử tương tác với khách hàng, các cuộc hẹn, yêu cầu hỗ trợ.
   * Triển khai các chương trình khách hàng thân thiết, quản lý điểm thưởng.
   * Tự động gửi thông báo (email/SMS) về các chương trình khuyến mãi, lịch bảo dưỡng xe.
4. **Phát triển module quản lý nhân sự**:
   * Tạo giao diện và chức năng quản lý thông tin chi tiết của từng nhân viên (lương, vị trí, lịch làm việc, hiệu suất).
   * Phân quyền chi tiết hơn cho từng vai trò người dùng trong hệ thống (ví dụ: nhân viên bán hàng chỉ có quyền xem tồn kho và tạo hóa đơn, không có quyền xóa xe).
5. **Tích hợp hệ thống thông báo và cảnh báo**:
   * Cảnh báo tự động khi số lượng xe tồn kho đạt đến ngưỡng tối thiểu.
   * Thông báo về các hợp đồng sắp đến hạn thanh toán hoặc các sự kiện quan trọng khác.
6. **Cải tiến bảo mật và sao lưu dữ liệu**:
   * Triển khai các biện pháp bảo mật nâng cao (ví dụ: mã hóa mật khẩu, kiểm tra phiên đăng nhập).
   * Xây dựng chức năng sao lưu và phục hồi cơ sở dữ liệu định kỳ để đảm bảo an toàn dữ liệu.
7. **Xem xét mở rộng sang ứng dụng web/di động**:
   * Nghiên cứu khả năng phát triển phiên bản ứng dụng web hoặc ứng dụng di động để tăng tính tiện lợi và khả năng truy cập từ xa cho người quản lý và khách hàng.
   * Điều này có thể đòi hỏi chuyển đổi sang các framework khác như Spring Boot (cho backend) và React/Angular/Vue (cho frontend web) hoặc React Native/Flutter (cho di động).

Với sự tiếp tục phát triển và cải tiến, "Phần mềm quản lý showroom ô tô" hứa hẹn sẽ trở thành một công cụ mạnh mẽ và không thể thiếu, giúp các showroom hoạt động hiệu quả hơn, nâng cao chất lượng dịch vụ và tăng cường lợi thế cạnh tranh trên thị trường.

# LỜI KẾT

Dự án "Phần mềm quản lý showroom ô tô" là một bước tiến quan trọng trong việc hiện đại hóa và tối ưu hóa các quy trình vận hành của một showroom. Với việc áp dụng các công nghệ tiên tiến như Java và Java Swing, cùng với hệ quản trị cơ sở dữ liệu MySQL, chúng tôi đã thành công trong việc xây dựng một hệ thống có khả năng quản lý toàn diện các hoạt động từ nhập xuất xe, thông tin khách hàng, quy trình bán hàng, cho đến việc thống kê doanh số. Các chức năng như quản lý danh sách xe , quản lý khách hàng , quản lý hợp đồng bán xe , và thống kê doanh số đã được thiết kế và triển khai nhằm đáp ứng các yêu cầu nghiệp vụ thực tế, giúp giảm thiểu sai sót, tiết kiệm thời gian và nâng cao hiệu quả làm việc.