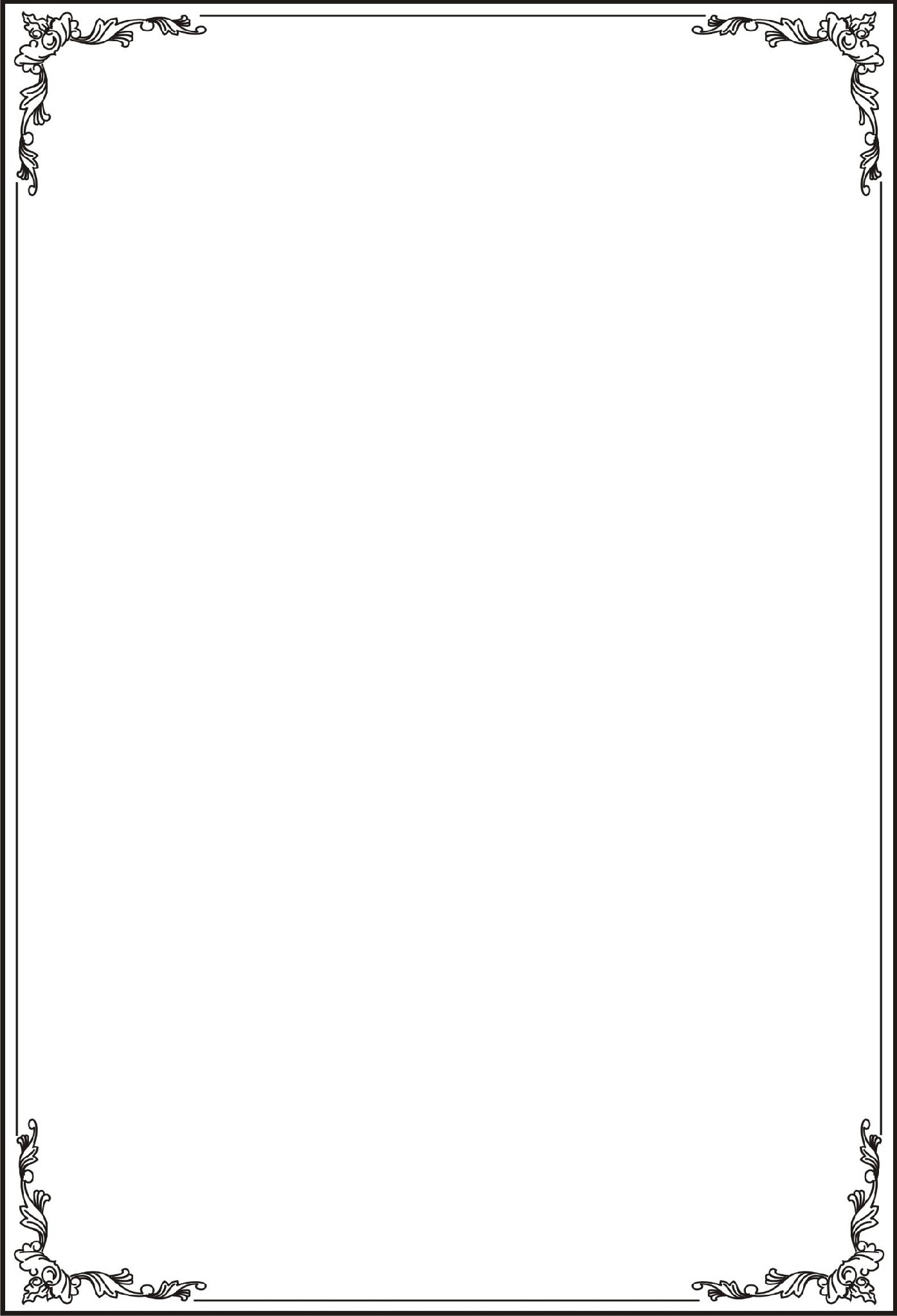
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH KHOA CÔNG NGHỆ THÔNG TIN**



**BỘ MÔN QUẢN TRỊ CƠ SỞ DỮ LIỆU**

-----🙞🙞🙞🙞🙞-----

### ĐỀ TÀI CUỐI KỲ MÔN HỆ QUẢN TRỊ CSDL

**TÊN ĐỀ TÀI: QUẢN LÝ DỊCH VỤ CHO THUÊ XE**

**GVHD:** *TS. Nguyễn Thành Sơn*

**Lớp HP:** *241DBMS330284*

**Nhóm thực hiện:** *Nhóm 11*

Phạm Tiến Anh – 22110282 Nguyễn Công Quý – 22110403 Nguyễn Đức Sang – 22110404 Lương Quang Thịnh – 22110428

**Học kỳ**: 1

**Năm học:** 2024-2025

Thành phố Hồ Chí Minh, tháng 9, năm 2024

DANH SÁCH SINH VIÊN NHÓM THỰC HIỆN HỌC KÌ 1 NĂM HỌC 2024 – 2025

Nhóm 11

Đề tài: Quản lý dịch vụ cho thuê xe

|  |  |
| --- | --- |
| 22110403 | Nguyễn Công Quý |
| 22110404 | Nguyễn Đức Sang |
| 22110282 | Phạm Tiến Anh |
| 22110428 | Lương Quang Thịnh |

### NHẬN XÉT CỦA GIÁO VIÊN

Tp. Hồ Chí Minh, ngày 26 tháng 9 năm 2024

Giảng viên chấm điểm

# MỤC LỤC

[CHƯƠNG 1: TỔNG QUAN VỀ HỆ THỐNG 6](#_bookmark0)

1. [Đặc tả đề tài 6](#_bookmark1)
   1. [Mô tả bài toán 6](#_bookmark2)
   2. [Mô tả dữ liệu 6](#_bookmark3)
   3. [Mô tả chức năng 9](#_bookmark4)
   4. [Về phần giao diện: 10](#_bookmark5)

[CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG 12](#_bookmark6)

1. [Thiết kế cơ sở dữ liệu mức quan niệm 12](#_bookmark7)
2. [Thiết kế cơ sở dữ liệu mức logic 14](#_bookmark8)
3. [Các ràng buộc cần có 15](#_bookmark9)
4. [Cài đặt SQL và các ràng buộc 19](#_bookmark10)
5. [Lược đồ quan hệ sau khi cài đặt SQL 24](#_bookmark11)
6. [Các view 25](#_bookmark12)
7. [Các trigger 26](#_bookmark13)

[CHƯƠNG 3: THIẾT KẾ CÁC CHỨC NĂNG 28](#_bookmark14)

1. [Kết nối cơ sở dữ liệu 28](#_bookmark15)
2. [Các chức năng chủ doanh nghiệp 28](#_bookmark16)
3. [Chức năng nhân viên 43](#_bookmark17)
4. [Chức năng của khách hàng 51](#_bookmark18)

[CHƯƠNG 4: PHÂN QUYỀN 62](#_bookmark19)

1. [Tạo Role và gán quyền 62](#_bookmark20)

[CHƯƠNG 5: THIẾT KẾ GIAO DIỆN 78](#_bookmark21)

1. [Môi trường cài đặt và công nghệ sử dụng 78](#_bookmark22)
2. [Phân chức năng chủ doanh nghiệp: 78](#_bookmark23)
3. [Các chức năng của nhân viên: 82](#_bookmark24)
4. [Các chức năng của khách hàng: 84](#_bookmark25)

### LỜI CẢM ƠN

Kính thưa thầy Nguyễn Thành Sơn,

Em xin bày tỏ lòng biết ơn sâu sắc của mình đến thầy bởi những sự hướng dẫn tận tình mà chúng em nhận được trong suốt quá trình học tập và hoàn thiện đồ án môn Hệ quản trị cơ sở dữ liệu này.

Trong khi thực hiện đồ án em gặp phải nhiều vấn đề phát sinh từ việc lên ý tưởng bài toán đến khi bắt tay phát triển và hoàn thiện. Nhờ sự chỉ dẫn tận tình của thầy mà em đã có thể hoàn thành đồ án hiệu quả và đúng đắn. Thầy đã dành thời gian và tâm huyết, giúp em trang bị những kiến thức về cơ sở dữ liệu và kỹ năng để phân công và phát triển đồ án một cách linh hoạt và hiệu quả. Hơn thế nữa, em sẽ có thể ứng dụng những kiến thức đã học này cho công việc và sự nghiệp sau này.

Một lần nữa, nhóm em xin chân thành cảm ơn thầy Nguyễn Thành Sơn vì sự giúp đỡ và hướng dẫn của thầy trong quá trình học tập của nhóm.

Trân trọng, Nguyễn Công Quý, Nguyễn Đức Sang, Phạm Tiến Anh,

Lương Quang Thịnh.

### LỜI NÓI ĐẦU

Xin chào thầy và các bạn,

Đồ án quản lý dịch vụ cho thuê xe là kết quả của nhóm em sau một thời gian học tập và thực hành môn học Hệ quản trị cơ sở dữ liệu. Đây là một đồ án đầy thử thách và hứa hẹn đem lại cho chúng ta những kiến thức và kinh nghiệm quý giá.

Trong quá trình hoàn thành đồ án, nhóm em đã đối mặt với rất nhiều khó khăn và thách thức. Tuy nhiên, với sự hướng dẫn và hỗ trợ tận tình của giáo viên hướng dẫn cùng với sự nỗ lực không ngừng của chúng em, đồ án đã được hoàn thành một cách tốt đẹp.

Trong bối cảnh xã hội hiện đại, việc tận dụng nguồn lực sẵn có, đặc biệt trong lĩnh vực phương tiện vận chuyển, ngày càng trở nên phổ biến. Tại Việt Nam, nhu cầu đi lại lớn và nguồn cung cấp phương tiện dồi dào, đặc biệt là xe máy - Việt Nam được coi là một quốc gia có số lượng xe máy nhiều nhất thế giới. Vì thế đề tài “Quản lý dịch vụ cho thuê xe” ra đời nhằm khai thác lợi thế này. Với mục đích là cho phép những khách hàng chưa có phương tiện nhưng có trình độ, bằng cấp về xe được phép thuê phương tiện thông qua những phương tiện có sẵn của công ty nhằm đáp ứng nhu cầu di chuyển linh hoạt của khách hàng và tạo giá trị kinh tế cho doanh nghiệp.

Đồ án quản lý dịch vụ cho thuê xe là một ứng dụng thực tiễn, nhằm mục đích giúp các doanh nghiệp cung cấp dịch vụ cho thuê xe có thể quản lý hoạt động của mình một cách dễ dàng và hiệu quả hơn. Bằng việc áp dụng những kiến thức về hệ quản trị cơ sở dữ liệu, chúng em đã xây dựng được một hệ thống quản lý hoàn chỉnh, bao gồm các chức năng cho khách hàng thuê xe, quản lý phương tiện, quản lý hợp đồng và các hoạt động khác.

Nhóm em hy vọng rằng đồ án của chúng em sẽ đem lại cho quý thầy cô và các bạn sinh viên những kiến thức và trải nghiệm thực tế hữu ích. Chúng em cũng mong rằng đồ án này sẽ truyền cảm hứng và khơi gợi những ý tưởng sáng tạo cho các bạn sinh viên khác trong việc ứng dụng hệ quản trị cơ sở dữ liệu vào thực tế.

Cảm ơn quý thầy cô và các bạn sinh viên đã theo dõi và đóng góp ý kiến cho đồ án của chúng em.

### CHƯƠNG 1: TỔNG QUAN VỀ HỆ THỐNG

### Đặc tả đề tài

### Mô tả bài toán

Hệ thống cho thuê xe giải quyết vấn đề bằng cách kết nối giữa doanh nghiệp và khách hàng do đó nó sẽ có những nhóm đối tượng sau:

Chủ doanh nghiệp: đây là đối tượng đại diện công ty cung cấp xe và thực hiện đăng tải xe lên hệ thống. Họ sẽ cung cấp về thông tin phương tiện và quản lý hợp đồng của khách hàng mượn xe trong một khoảng thời gian với mức phí phù hợp.

Nhân viên: đây là nhóm đối tượng thuộc sự quản lý của Chủ doanh nghiệp, nhân viên sẽ thực hiện công việc giao xe cho khách hàng ngay sau khi hợp đồng giữa doanh nghiệp và khách hàng được hoàn tất, nhận xe ngay sau khi thời hạn thuê của khách hàng đã kết thúc như đúng với hợp đồng.

Khách hàng: nhóm đối tượng này không sở hữu phương tiện cá nhân hoặc đang có nhu cầu thuê xe trong thời gian ngắn (có bằng cấp, trình độ về việc lái xe) với mục đích cá nhân như công việc, du lịch ngắn hạn, đi lại… Họ sẽ sử dụng hệ thống để tìm kiếm và thuê xe với mức giá phù hợp đã hiển thị trên hệ thống. Việc thuê xe sẽ được tiến hành qua ký kết hợp đồng với chủ doanh nghiệp về thời gian thuê, chi phí, chính sách thuê xe…

Mục tiêu của hệ thống là quản lý việc này một cách tự động và hiệu quả quy trình thuê xe từ khâu quản lý phương tiện, giao nhận xe, ký kết hợp đồng hay thanh toán, không chỉ giảm thiểu các lỗi thủ công mà còn tiện lợi cho việc theo dõi, phân tích dữ liệu qua đó giúp cho doanh nghiệp đạt được mục đích của mình và cung cấp cho khách hàng trải nghiệm dịch vụ minh bạch và an toàn.

### Mô tả dữ liệu

- **Mô tả các tập thực thể**

Hệ thống sẽ cung cấp dịch vụ thuê mượn đối với nhiều loại xe. Ở đây sẽ chia thành 2 loại xe chính đó là : xe máy, xe ô tô. Mỗi loại xe sẽ chỉ được sử dụng dịch vụ thuê mượn của loại xe đó, điều này giúp hệ thống dễ dàng chia giá thành cho hợp lý.

Quản lý dịch vụ thuê xe không thể thiếu dữ liệu về phương tiện. Trong hệ thống này, mỗi xe sẽ có các thông tin quan trọng như mã xe (biển số), hãng xe, tên xe, trạng thái, giá xe, hình ảnh, phí xăng. Mỗi phương tiện chỉ được cho thuê theo thời điểm xác định, tùy thuộc vào thời hạn hợp đồng mà người sở hữu xe đã ký kết với công ty, nếu xe đó đã được thuê thì nó sẽ tạm thời không xuất hiện trên hệ thống.

Ví dụ: ông A thuê xe X trong vòng 5 ngày, thì xe X sẽ tạm thời không hiển thị trên hệ thống trong 5 ngày thuê đó do vậy khách hàng khác thì cũng không thể thuê xe X. Sau khi hết 5 ngày của thời hạn hợp đồng khách hàng đã trả xe thì xe đó sẽ thay đổi trạng thái hiển thị trở lại trên hệ thống và cho phép khách hàng khác tiến hành thuê.

Khi khách hàng thuê xe sẽ tiến hành tạo hợp đồng, thông tin hợp đồng hợp đồng thuê xe bao gồm: Mã hợp đồng, nội dung, tiền cọc, ngày bắt đầu, ngày kết thúc, tổng tiền. Và sẽ có một mã khách hàng để cho biết rằng hợp đồng này là của khách hàng nào.

Trong hệ thống thuê xe, các hoạt động như giao xe, nhận xe hoặc các nhiệm vụ khác đều được quản lý thông qua các công việc cụ thể. Mỗi công việc sẽ có trạng thái công việc, ngày thực hiện công việc và sẽ được gán cho một nhân viên thực hiện. Công việc này cũng sẽ liên kết với hợp đồng và loại công việc để phân biệt rõ nhiệm vụ.

Mỗi hợp đồng sẽ cần phải được thanh toán, điều này sẽ dễ dàng cho việc tạo số liệu tổng kết doanh thu, thanh toán bao gồm : Mã thanh toán, số tiền, ngày thanh toán, phương thức thanh toán. Đồng thời sẽ kèm thêm mã hợp đồng để xác định chính xác cần thanh toán cho hợp đồng nào.

Sau khi hợp đồng được ký kết và hoàn tất, việc giao xe cho khách hàng sẽ được tiến hành. Thông tin của xe trước khi giao sẽ được nhân viên lưu lại bao gồm: Số km đi, số xăng đi và trạng thái của xe. Mỗi lần giao xe sẽ có một mã công việc để xác định nhiệm vụ đó được giao cho nhân viên nào.

Khi thời hạn hợp đồng kết thúc, nhân viên sẽ thực hiện việc nhận xe. Họ sẽ kiểm tra và ghi lại thông tin về xe như: Số km về, số xăng còn lại, và trạng thái của xe. Tương tự như giao xe, quá trình nhận xe cũng có một mã công việc để xác định rõ công việc chuyên trách này.

Trong quá trình sử dụng dịch vụ, có thể xảy ra các sự cố phát sinh. Mỗi sự cố sẽ được lưu trữ lại nhằm dễ dàng cho việc quản lý và xử lý, dữ liệu về sự cố sẽ bao gồm: Mã sự cố, tên sự cố, lý do, nội dung, tiền phạt. Đồng thời, hệ thống sẽ kèm thêm mã hợp đồng để xác định sự cố xảy ra trong hợp đồng nào.

Hệ thống cung cấp dịch vụ thuê nhiều loại xe khác nhau, từ xe máy đến ô tô. Mỗi loại xe sẽ có một mã loại và những thông tin như mô tả, tên loại, và số lượng xe thuộc loại đó. Điều này giúp hệ thống dễ dàng phân loại và quản lý các phương tiện cho thuê.

Mỗi loại xe sẽ có một mức giá thuê riêng biệt. Hệ thống sẽ lưu trữ thông tin về giá thuê của từng xe, bao gồm thời gian và giá thuê tương ứng. Điều này cho phép khách hàng dễ dàng chọn xe và thuê xe dựa trên giá cả phù hợp.

Về phần lưu trữ các thông tin của người dùng:

Tất cả người dùng trong hệ thống sẽ cần phải có tài khoản để đăng nhập và sử dụng dịch vụ. Tài khoản sẽ chứa các thông tin đăng nhập cơ bản như tên đăng nhập, mật khẩu và vai trò của người dùng (Chủ doanh nghiệp, Nhân viên, Khách hàng). Vai trò này sẽ giúp hệ thống xác định quyền hạn của mỗi người dùng khi truy cập vào hệ thống.Đó là những thông tin cơ bản để giúp xác định người dùng nhằm biết danh tính của Chủ doanh nghiệp và để quản lý nhân viên hay lưu trữ thông tin của khách hàng sử dụng dịch vụ.

Chủ doanh nghiệp là người có quyền quản lý toàn bộ hệ thống và có quyền tạo, chỉnh sửa các thông tin quan trọng liên quan đến doanh nghiệp và hoạt động của nó. Thông tin của chủ doanh nghiệp sẽ được lưu trữ trong hệ thống bao gồm họ tên, địa chỉ, CMND, giới tính và ngày sinh. Mỗi chủ doanh nghiệp cũng sẽ có một tài khoản để đăng nhập vào hệ thống và quản lý hoạt động của mình.

Nhân viên là những người đảm nhiệm các công việc như giao xe, nhận xe hoặc quản lý các tác vụ khác liên quan đến hoạt động thuê xe. Họ cũng có tài khoản đăng nhập vào hệ thống để thực hiện công việc. Thông tin của nhân viên được lưu trữ bao gồm họ tên, địa chỉ, giới tính, ngày sinh và CMND. Mỗi nhân viên sẽ được phân công các công việc cụ thể thông qua hệ thống.

Khách hàng là người sử dụng dịch vụ thuê xe của hệ thống. Mỗi khách hàng cũng có tài khoản để đăng nhập và thực hiện các thao tác như tạo hợp đồng, thanh toán, v.v. Hệ thống sẽ lưu trữ thông tin của khách hàng bao gồm họ tên, địa chỉ, giới tính, ngày sinh và CMND. Thông qua mã tài khoản, hệ thống sẽ xác định khách hàng sử dụng tài khoản nào.

### - Mô tả các mối quan hệ

Mỗi người dùng có một tài khoản. Mối quan hệ này là 1 – 1 có nghĩa là một người dùng chỉ có thể có một tài khoản và ngược lại

Người dùng chia làm 3 loại : Nhân Viên, Chủ Doanh Nghiệp, và Khách hàng

Một Nhân viên có thể đảm nhiệm nhiều công việc, nhưng một công việc chỉ do một nhân viên thực hiện

Một Công việc phải liên quan đến hành động Giao Xe hoặc Nhận Xe. Đây là hai loại công việc khác nhau

Mỗi Khách Hàng có thể ký nhiều hợp đồng, trong đó mỗi hợp đồng sẽ liên quan đến một hoặc nhiều xe, chứng tỏ việc khách hàng có thể thuê nhiều xe cùng một lúc

Mỗi Hợp Đồng có thể có một Thanh toán (quan hệ 1-N) nhưng mỗi thanh toán chỉ thuộc về một hợp đồng

Một Sự Cố có thể phát sinh trong quá trình thực hiện hợp đồng, mỗi hợp đồng có thể gặp nhiều sự cố

Mỗi Xe thuộc về Loại xe, nghĩa là một loại xe bao gồm nhiều xe

Xe có thể có nhiều mức giá thuê khác nhau tùy theo thời gian, nghĩa là một xe có thể nhiều giá xe

### Mô tả chức năng

**Có ba nhóm người dùng chính là: Chủ doanh nghiệp, nhân viên và khách hàng**

#### Chức năng chung:

*+ Đăng nhập:* Mỗi người dùng có tài khoản riêng để đăng nhập và truy cập các chức năng tương ứng với quyền hạn của mình.

*+ Quản lý thông tin cá nhân:* Người dùng có thể chỉnh sửa và cập nhật thông tin cá nhân của mình.

#### Chức năng của chủ doanh nghiệp:

*+ Quản lý hợp đồng:* Theo dõi và điều chỉnh các hợp đồng thuê xe, bao gồm trạng thái, chi tiết hợp đồng, và lịch sử giao dịch, xác nhận hợp đồng khi có yêu cầu từ phía khách hàng

*+ Quản lý nhân viên:* Thêm, sửa, xóa thông tin nhân viên; theo dõi hoạt động của từng nhân viên trong hệ thống.

*+ Quản lý xe:* Bổ sung hoặc xóa xe.

*+ Phân công công việc:* Giao việc cho từng nhân viên, phân chia công việc dựa trên hợp đồng và yêu cầu khách hàng.

*+ Thống kê và đánh giá:* Tạo các báo cáo thống kê về hoạt động kinh doanh, đánh giá doanh số doanh thu của doanh nghiệp.

#### Chức năng của nhân viên:

+ *Giao xe*: Nhân viên thực hiện giao xe cho khách hàng.

+ *Nhận xe*: Nhân viên tiến hành nhận xe sau khi khách hàng thuê hoàn tất

+ *Xác nhận và cập nhật sự cố*: Sau khi khách hàng trả xe, nhân viên sẽ tiến hành kiểm tra xe nếu có sự cố liên quan đến xe và cập nhật sự cố đó lên hệ thống.

#### Chức năng của khách hàng:

*+ Đăng ký tài khoản:* Khách hàng tạo tài khoản cá nhân để dễ dàng quản lý dịch vụ thuê xe.

*+ Tạo hợp đồng:* Đăng ký thuê xe cho công ty hoặc cá nhân, với các tùy chọn và dịch vụ đa dạng theo nhu cầu.

*+ Quản lý hợp đồng:* Theo dõi hợp đồng đang thực hiện hay những hợp đồng đã hoàn thành từ trước, kiểm tra tiến độ và cập nhật thông tin liên quan, có thể hủy hợp đồng nếu hợp đồng đó chưa được xác nhận từ phía chủ doanh nghiệp.

*+ Thanh toán:* Thực hiện thanh toán cho các hợp đồng đã hoàn thành một cách nhanh chóng và tiện lợi.

*+ Tìm kiếm:* Tìm kiếm xe theo loại xe có sẵn trong cửa hàng

### Về phần giao diện:

**Phần chung:** Là những form mà tất cả người dùng đều thao tác

*Giao diện đăng nhập (LoginForm)* cho phép người dùng truy cập vào hệ thống với các thành phần như: Nhãn tên người dùng, Nhãn mật khẩu, Ô nhập tên người dùng, Ô nhập mật khẩu (mật khẩu được ẩn đi), Nút đăng nhập, và Liên kết quên mật khẩu. Các nhãn và ô nhập liệu được sắp xếp dọc ở trung tâm màn hình, với nút đăng nhập đặt ngay dưới ô nhập liệu, và liên kết chuyển đến trang quên mật khẩu nằm ở phía dưới cùng.

*Giao diện đăng ký tài khoản (RegisterForm)* cho phép người dùng tạo tài khoản mới, bao gồm: Nhãn họ và tên, Nhãn email, Nhãn tên người dùng, Ô nhập họ và tên, Ô nhập email, Ô nhập tên người dùng, Ô nhập mật khẩu, Ô nhập xác nhận mật khẩu và Nút đăng ký. Các nhãn và ô nhập liệu được sắp xếp dọc, và nút đăng ký được đặt ở dưới cùng. Trước khi cho phép đăng ký, hệ thống sẽ kiểm tra tính hợp lệ của thông tin.

**Phần giao diện riêng:** Bao gồm các phần giao diện con được quản lý trong một giao diện chính để tối ưu trải nghiệm.

*Giao diện quản trị viên (AdminForm)* cho phép quản lý các chức năng như: Quản lý nhân viên, Quản lý dịch vụ, Quản lý hợp đồng, và Thống kê hoạt động doanh nghiệp. Giao diện sử dụng Tab quản lý để tổ chức các chức năng trên một màn hình, bao gồm:

*+ Tab Quản lý nhân viên*: Hiển thị danh sách nhân viên qua Bảng thông tin nhân viên. Ngoài ra, còn có các nút để thực hiện thao tác như Nút thêm nhân viên, Nút chỉnh sửa nhân viên, Nút xóa nhân viên,... cùng với Ô tìm kiếm để tìm kiếm nhân viên.

+ *Tab Quản lý hợp đồng:* Hiển thị danh sách hợp đồng qua Bảng thông tin hợp đồng. Các nút thao tác gồm có Nút xác nhận hợp đồng và Nút hủy hợp đồng.

+ *Tab Quản lý dịch vụ:* Hiển thị danh sách dịch vụ qua Bảng thông tin dịch vụ. Các nút thao tác bao gồm Nút thêm dịch vụ, Nút chỉnh sửa dịch vụ, và Nút xóa dịch vụ.

*+ Tab Thống kê:* Hiển thị biểu đồ Thống kê doanh thu, kèm theo Danh sách lựa chọn loại báo cáo để chọn kiểu thống kê cần xem.

Mỗi tab hiển thị một chức năng quản lý riêng, với bảng thông tin và các nút tương tác như thêm, sửa, xóa.

*Giao diện khách hàng (CustomerForm)* cho phép người dùng tạo hợp đồng mới, thanh toán và quản lý các hợp đồng hiện có. Người dùng có thể tìm kiếm dịch vụ hoặc xe qua Ô tìm kiếm dịch vụ và Nút tìm kiếm dịch vụ. Các hợp đồng đã đăng ký sẽ được hiển thị trong Bảng thông tin hợp đồng của khách hàng, nơi người dùng có thể quản lý hợp đồng, hủy hợp đồng bằng Nút hủy hợp đồng, hoặc thanh toán hợp đồng qua Nút thanh toán hợp đồng, tạo nên giao diện trực quan và dễ sử dụng.

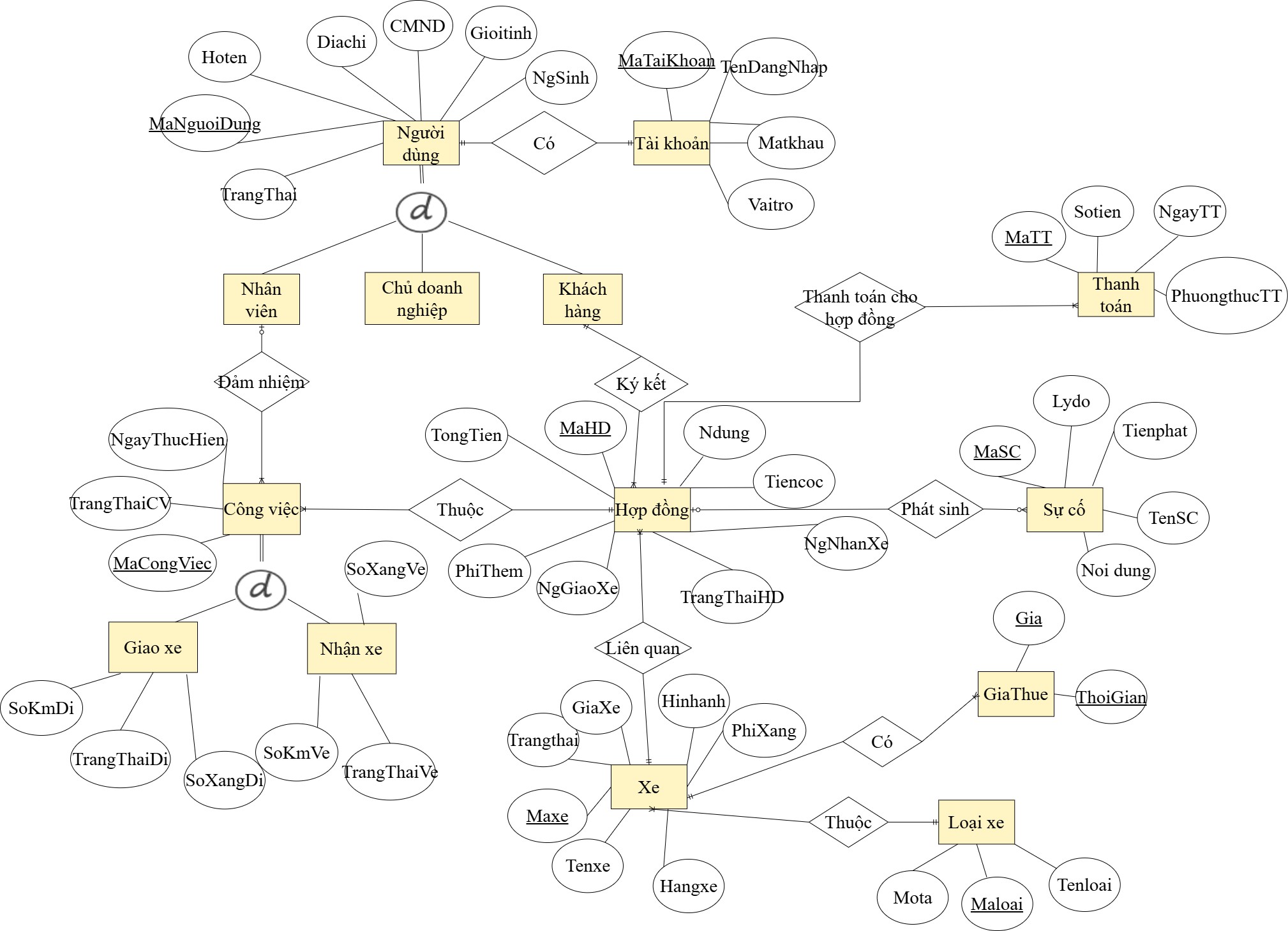
*Giao diện nhân viên (EmployeeForm)* bao gồm các chức năng như Giao xe, Nhận xe, và Quản lý sự cố. Form này hiển thị danh sách hợp đồng mà nhân viên được phân công qua Bảng hợp đồng đã phân công, cùng với các nút thao tác như Nút giao xe, Nút nhận xe sau khi khách trả xe, và Nút báo cáo sự cố liên quan đến xe.

Việc xây dựng một hệ thống quản lý dịch vụ thuê xe là một bước tiến quan trọng trong việc tự động hóa và tối ưu hóa quy trình quản lý dịch vụ. Hệ thống không chỉ giúp tiết kiệm thời gian và nguồn lực mà còn nâng cao chất lượng phục vụ, mang lại sự hài lòng cho khách hàng. Bằng việc tích hợp các chức năng quản lý thông tin và phân tích dữ liệu, doanh nghiệp có thể dễ dàng điều hành và phát triển hơn trong môi trường cạnh tranh khốc liệt hiện nay.

### CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

# Thiết kế cơ sở dữ liệu mức quan niệm

Từ mô tả về dữ liệu cần có ở phần mô tả của bài toán ta hình thành được sơ đồ thực thể kết hợp (ERD).



### A diagram of a company Description automatically generated with medium confidenceThiết kế cơ sở dữ liệu mức logic

TaiKhoan (MaTaiKhoan, username, password, LoaiNguoiDung)

NhanVien (MaNV, Hoten, Diachi, GioiTinh, NgSinh, CMND, MaTaiKhoan, TrangThai)

KhachHang (MaKH, Hoten, Diachi, GioiTinh, NgSinh, CMND, MaTaiKhoan, TrangThai)

ChuDoanhNghiep (MaCDN, Hoten, Diachi, GioiTinh, NgSinh, CMND, MaTaiKhoan, TrangThai)

HopDong (MaHD, NoiDung, TongTien, PhiThem, NgGiaoXe, NgNhanXe, TienCoc, MaKH, Maxe, TrangThaiHD)

CongViec (MaCongViec, TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKm, SoXang, TrangThaiXe, TrangThaiCV)

Xe (Maxe, Tenxe, Trangthai, Hinhanh, PhiXang, BienSoXe, HangXe, MaLoai)

LoaiXe (Maloai, Mota, Tenloai, Soluong)

GiaThue (MaXe, ThoiGian, GiaThue)

ThanhToan (MaTT, SoTien, NgayTT, PhuongThucTT, MaHD)

SuCo (MaSC, TenSC, LyDo, NoiDung, TienPhat, TrangThai, MaHD)

### Các ràng buộc cần có

|  |  |  |
| --- | --- | --- |
| **STT** | **BẢNG** | **RÀNG BUỘC** |
| 1 | **Khách hàng**  Hoten nvarchar (100), DiaChi nvarchar (255) , GioiTinh nvarchar (10), NgSinh Date,  CMND nvarchar (15), MaTaiKhoan int, TrangThai nvarchar(50) | Khóa chính MaKH  Khóa ngoại MaTaiKhoan đến bảng TaiKhoan  NOT NULL HoTen  NOT NULL DiaChi NOT NULL GioiTinh NOTNULL NgSinh  UNIQUE, NOTNULL CMND |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | CHECK TrangThai | |
| 2 | **Hợp đồng** | Khóa chính MaHD |  |
|  | MaHD int,  NoiDung nvarchar(255), | Khóa ngoại MaKH đến KhachHang | bảng |
|  | TongTien decimal(10, 2), | Khóa ngoại MaXe đến bảng Xe |  |
|  | PhiThem decimal(10, 2), | NOTNULL TongTien |  |
|  | NgayLap date, | NOTNULL NgayLap |  |
|  | NgayGiaoXe date, | NOTNULL NgBatDau |  |
|  | NgayNhanXe date, | NOTNULL NgKetThuc |  |
|  | MaKH int, | NOTNULL TrangThaiHD |  |
|  | MaXe int, | CHECK TrangThaiHD |  |
|  | TrangThaiHD nvarchar(50) |  |  |
| 3 | **Xe** | Khóa chính MaXe | |
|  | MaXe int,  TenXe nvarchar (100), | Khóa ngoại MaLoai tham chiếu bảng LoaiXe | |
|  | TrangThai nvarchar(50), | NOT NULL TenXe | |
|  | HinhAnh varbinary(MAX), | NOT NULL TrangThai  NOT NULL MaLoai | |
|  | PhiXang decimal(10, 2), | CHECK TrangThai | |
|  | HangXe nvarchar (50), |  | |
|  | MaLoai nvarchar(50) |  | |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 4 | **LoaiXe**  MaLoai int,  MoTa nvarchar (255),  TenLoai nvarchar (100), | Khóa chính MaLoai, NOT NULL TenLoai, |
| 5 | **TaiKhoan** MaTaiKhoan int username nvarchar (50)  password nvarchar (50)  LoaiNguoiDung nvarchar (50) | Khóa chính MaTaiKhoan  NOT NULL, UNIQUE username NOT NULL password  NOT NULL LoaiNguoiDung CHECK LoaiNguoiDung |
| 6 | **NhanVien**  MaNV int,  Hoten nvarchar (100),  DiaChi nvarchar (255),  GioiTinh nvarchar (10), NgSinh Date,  CMND nvarchar (15), MaTaiKhoan int, TrangThai nvarchar(50) | Khóa chính MaNV  Khóa ngoại MaTaiKhoan tham chiếu đến bảng TaiKhoan  NOT NULL Hoten NOT NULL GioiTinh NOT NULL DiaChi NOT NULL NgaySinh  NOT NULL, UNIQUE CMND  NOT NULL TrangThai CHECK TrangThai |

|  |  |  |
| --- | --- | --- |
| 7 | **CongViec**  MaCongViec int,  TrangThaiCV nvarchar (50),  NgayThucHien Date, MaNV int,  MaHD int,  LoaiCV nvarchar (50), SoKM int,  SoXang int,  TrangThai nvarchar(50) | Khóa chính MaCongViec  Khóa ngoại MaNV tham chiếu bảng NhanVien  Khóa ngoại MaHD tham chiếu bảng HopDong  NOT NULL TrangThaiCV NOT NULL TrangThai CHECK TrangThai |
| 8 | **ChuDoanhNghiep**  MaCDN int,  Hoten nvarchar (100),  DiaChi nvarchar (255),  GioiTinh nvarchar (10), NgSinh Date,  CMND nvarchar (50), MaTaiKhoan int, TrangThai nvarchar(50) | Khóa chính MaCDN  Khóa ngoại MaTaiKhoan tham chiếu qua bảng TaiKhoan  NOT NULL Hoten NOT NULL GioiTinh, NOT NULL DiaChi,  NOT NULL NgaySinh,  NOT NULL UNIQUE CMND  NOT NULL TrangThai |
| 9 | **SuCo**  MaSC int, | Khóa chính MaSC,  Khóa ngoại MaHD tham chiếu qua bảng HopDong |

|  |  |  |
| --- | --- | --- |
|  | TenSC nvarchar (100),  LyDo nvarchar (255),  NoiDung nvarchar (255),  TienPhat decimal(10, 2), TrangThai nvarchar(50), MaHD int | NOT NULL TenSC, NOT NULL LyDo,  NOT NULL NoiDung, NOT NULL TienPhat, NOT NULL TrangThai CHECK TrangThai |
| 10 | **ThanhToan**  MaTT int  SoTien decimal(10, 2) NgayTT Date  PhuongThucTT nvarchar (50)  MaHD int | Khóa chính MaTT, NOT NULL SoTien, NOT NULL NgayTT,  NOT NULL PhuongThucTT,  Khóa ngoại MaHD khóa ngoại tham chiếu qua bảng HopDong |
| 11 | **GiaThue**  ThoiGian int  GiaThue decimal(10, 2) | Khóa chính MaXe, ThoiGian,  Khóa ngoại MaXe tham chiếu qua bảng Xe  CHECK ThoiGian CHECK GiaThue >= 0 |

1. **Cài đặt SQL và các ràng buộc Bảng Tài Khoản**

CREATE TABLE TaiKhoan (

MaTaiKhoan INT PRIMARY KEY IDENTITY, username NVARCHAR(50) NOT NULL, password NVARCHAR(50) NOT NULL,

LoaiNguoiDung NVARCHAR(50) NOT NULL

);

CREATE TABLE TaiKhoan

ADD CONSTRAINT CK\_TaiKhoan\_LoaiNguoiDung CHECK (LoaiNguoiDung IN (N'Khách hàng', N'Chủ doanh nghiệp', N'Nhân Viên'));

**Bảng Khách Hàng**

CREATE TABLE KhachHang (

MaKH INT PRIMARY KEY IDENTITY, HoTen NVARCHAR(100) NOT NULL, DiaChi NVARCHAR(255) NOT NULL, GioiTinh NVARCHAR(10),

NgaySinh DATE,

CMND NVARCHAR(15) UNIQUE NOT NULL,

MaTaiKhoan INT,

TrangThai NVARCHAR(50) NOT NULL,

CONSTRAINT FK\_KhachHang\_TaiKhoan FOREIGN KEY (MaTaiKhoan) REFERENCES TaiKhoan(MaTaiKhoan) ON DELETE SET NULL

);

CREATE TABLE KhachHang

ADD CONSTRAINT CK\_KhachHang\_TrangThai CHECK (TrangThai IN (N'Hoạt động', N'Ngừng hoạt động'));

**Bảng Nhân Viên**

CREATE TABLE NhanVien (

MaNV INT PRIMARY KEY IDENTITY, HoTen NVARCHAR(100) NOT NULL, DiaChi NVARCHAR(255) NOT NULL, GioiTinh NVARCHAR(10),

NgaySinh DATE,

CMND NVARCHAR(15) UNIQUE NOT NULL,

MaTaiKhoan INT,

TrangThai NVARCHAR(50) NOT NULL,

CONSTRAINT FK\_NhanVien\_TaiKhoan FOREIGN KEY (MaTaiKhoan) REFERENCES TaiKhoan(MaTaiKhoan) ON DELETE SET NULL

);

CREATE TABLE NhanVien

ADD CONSTRAINT CK\_NhanVien\_TrangThai CHECK (TrangThai IN (N'Hoạt động', N'Ngừng hoạt động'));

**Bảng Chủ Doanh Nghiệp**

CREATE TABLE ChuDoanhNghiep (

MaCDN INT PRIMARY KEY IDENTITY, HoTen NVARCHAR(100) NOT NULL, DiaChi NVARCHAR(255) NOT NULL, GioiTinh NVARCHAR(10),

NgaySinh DATE,

CMND NVARCHAR(15) UNIQUE NOT NULL,

MaTaiKhoan INT,

TrangThai NVARCHAR(50) NOT NULL,

CONSTRAINT FK\_ChuDoanhNghiep\_TaiKhoan FOREIGN KEY (MaTaiKhoan) REFERENCES TaiKhoan(MaTaiKhoan)

);

CREATE TABLE ChuDoanhNghiep

ADD CONSTRAINT CK\_ChuDoanhNghiep\_TrangThai CHECK (TrangThai IN (N'Hoạt động', N'Ngừng hoạt động'));

**Bảng Loại Xe**

CREATE TABLE LoaiXe (

MaLoai INT PRIMARY KEY IDENTITY, TenLoai NVARCHAR(100) NOT NULL, MoTa NVARCHAR(255),

);

**Bảng Xe**

CREATE TABLE Xe (

MaXe INT PRIMARY KEY IDENTITY, TenXe NVARCHAR(100) NOT NULL, TrangThai NVARCHAR(50) NOT NULL, HinhAnh VARBINARY(MAX),

PhiXang DECIMAL(10, 2), HangXe NVARCHAR(50),

MaLoai NVARCHAR(50) NOT NULL,

CONSTRAINT FK\_Xe\_LoaiXe FOREIGN KEY (MaLoai) REFERENCES LoaiXe(MaLoai);

);

CREATE TABLE Xe

add CONSTRAINT CK\_Xe\_TrangThai CHECK (TrangThai IN (N'Tồn tại', N'Không tồn tại'));

**Bảng Giá Thuê**

CREATE TABLE GiaThue ( MaXe INT,

ThoiGian NVARCHAR(50) CHECK (ThoiGian IN (N'Giờ', N'Ngày',N'Tuần', N'Tháng', N'Năm')), -- Ràng buộc CHECK cho ThoiGian

GiaThue DECIMAL(10, 2) CHECK (GiaThue >= 0), -- Ràng buộc CHECK cho GiaThue PRIMARY KEY (MaXe, ThoiGian),

CONSTRAINT FK\_GiaThue\_Xe FOREIGN KEY (MaXe) REFERENCES Xe(MaXe)

);

**Bảng Hợp Đồng**

CREATE TABLE HopDong (

MaHD INT PRIMARY KEY IDENTITY,

NoiDung NVARCHAR(255),

TongTien DECIMAL(10, 2) NOT NULL, PhiThem DECIMAL(10, 2),

NgayLap DATE NOT NULL, NgayGiaoXe DATETIME NOT NULL,

NgayNhanXe DATETIME NOT NULL, MaKH INT,

MaXe INT,

TrangThaiHD NVARCHAR(50) NOT NULL,

CONSTRAINT FK\_HopDong\_KhachHang FOREIGN KEY (MaKH) REFERENCES KhachHang(MaKH), CONSTRAINT FK\_HopDong\_Xe FOREIGN KEY (MaXe) REFERENCES Xe(MaXe)

);

CREATE TABLE HopDong

ADD CONSTRAINT CK\_HopDong\_TrangThaiHD CHECK (TrangThaiHD IN (N'Chưa xác nhận', N'Đã hoàn thành',N'Đang xử lý', N'Đã hủy'));

**Bảng Thanh Toán**

CREATE TABLE ThanhToan (

MaTT INT PRIMARY KEY IDENTITY, SoTien DECIMAL(10, 2) NOT NULL, NgayTT DATE NOT NULL,

PhuongThucTT NVARCHAR(50) NOT NULL, MaHD INT,

CONSTRAINT FK\_ThanhToan\_HopDong FOREIGN KEY (MaHD) REFERENCES HopDong(MaHD)

);

**Bảng Sự Cố**

CREATE TABLE SuCo (

MaSC INT PRIMARY KEY IDENTITY, TenSC NVARCHAR(100) NOT NULL, LyDo NVARCHAR(255),

NoiDung NVARCHAR(255), TienPhat DECIMAL(10, 2),

TrangThai NVARCHAR(50) NOT NULL, MaHD INT,

CONSTRAINT FK\_SuCo\_HopDong FOREIGN KEY (MaHD) REFERENCES HopDong(MaHD)

);

CREATE TABLE SuCo

add CONSTRAINT CK\_SuCo\_TrangThai CHECK (TrangThai IN (N'Đang xử lý', N'Đã giải quyết'));

**Bảng Công Việc**

CREATE TABLE CongViec (

MaCongViec INT PRIMARY KEY IDENTITY, TrangThaiCV NVARCHAR(50) NOT NULL,

NgayThucHien DATE, MaNV INT,

MaHD INT,

LoaiCV NVARCHAR(50), SoKM INT,

SoXang INT,

TrangThai NVARCHAR(50) NOT NULL,

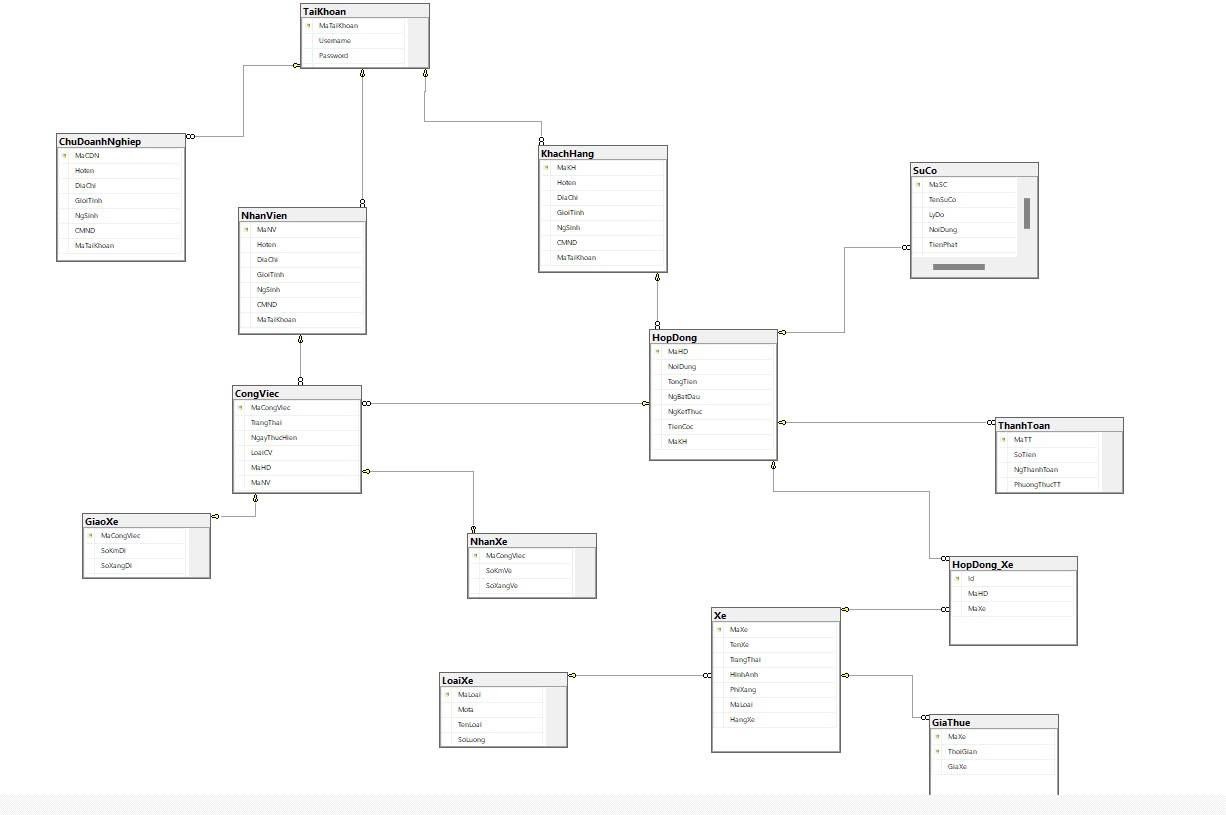
CONSTRAINT FK\_CongViec\_NhanVien FOREIGN KEY (MaNV) REFERENCES NhanVien(MaNV), CONSTRAINT FK\_CongViec\_HopDong FOREIGN KEY (MaHD) REFERENCES HopDong(MaHD)

);

CREATE TABLE CongViec

ADD CONSTRAINT CK\_CongViec\_LoaiCV CHECK (LoaiCV IN (N'Giao Xe', N'Nhận xe'));

### Lược đồ quan hệ sau khi cài đặt SQL

****

### Các view

1. **View xem các Công việc của nhân viên**

create view vw\_CongViec as select \* from CongViec

1. **View xem chi tiết Hợp đồng**

create view vwChiTietHopDong as select \* from HopDong

1. **View xem khách hàng**

create view vw\_KhachHang as select \* from KhachHang

1. **View xem nhân viên**

create view vw\_NhanVien as select \* from NhanVien

1. **View xem thanh toán**

create view vw\_ThanhToan as select \* from ThanhToan

1. **View xem chi tiết xe**

CREATE VIEW vwChiTietXe AS SELECT

Xe.MaXe, Xe.TenXe, Xe.TrangThai, Xe.HinhAnh, Xe.PhiXang, Xe.HangXe, Xe.BienSoXe, LoaiXe.MaLoai, LoaiXe.TenLoai, LoaiXe.MoTa, GiaThue.ThoiGian, GiaThue.GiaThue

FROM Xe

INNER JOIN LoaiXe ON Xe.MaLoai = LoaiXe.MaLoai INNER JOIN GiaThue ON GiaThue.MaXe = Xe.MaXe;

### Các trigger

1. **Trigger cho HopDong để cập nhật trạng thái khi xóa**

CREATE TRIGGER trg\_HopDong\_Delete ON HopDong

INSTEAD OF DELETE AS

BEGIN

UPDATE HopDong

SET TrangThaiHD = N'Đã hủy'

WHERE MaHD IN (SELECT MaHD FROM deleted);

END;

1. **Trigger kiểm tra trạng thái HopDong khi thêm thanh toán (Nếu hợp đồng không ở trạng thái "Đang xử lý" không cho phép thêm thanh toán mới)**

CREATE TRIGGER trg\_ThanhToan\_Insert\_CheckHopDong ON ThanhToan

AFTER INSERT AS

BEGIN

IF EXISTS (

SELECT 1

FROM HopDong hd

JOIN inserted i ON hd.MaHD = i.MaHD

WHERE hd.TrangThaiHD <> N'Đang xử lý'

) BEGIN

RAISERROR('Không thể thêm thanh toán mới cho hợp đồng.', 16, 1); ROLLBACK;

END

END;

1. **Trigger xóa xe chuyển trạng thái sang “Không tồn tại”**

ALTER TRIGGER [dbo].[trg\_Xe\_Delete] ON [dbo].[Xe]

INSTEAD OF DELETE AS

BEGIN

UPDATE Xe

SET TrangThai = N'Không tồn tại'

WHERE MaXe IN (SELECT MaXe FROM deleted);

END;

1. **Khách hàng ở trạng thái ngừng hoạt động thì không thể thêm hợp đồng**

CREATE TRIGGER trg\_HopDong\_Insert\_CheckKhachHang ON HopDong

AFTER INSERT AS

BEGIN

IF EXISTS ( SELECT 1

FROM KhachHang k

JOIN inserted i ON k.MaKH = i.MaKH WHERE k.TrangThai = N'Ngừng hoạt động'

) BEGIN

RAISERROR('Khách hàng đã ngừng hoạt động, không thể tạo hợp đồng mới.', 16, 1); ROLLBACK;

END

END;

### CHƯƠNG 3: THIẾT KẾ CÁC CHỨC NĂNG

### Kết nối cơ sở dữ liệu

private static SqlConnection connection;

public static SqlConnection Connection { get => connection; set => connection = value; }

public static void SetMyDB(string username, string passWord)

{

Connection = new SqlConnection($@"Data Source =Tieanh19\SQLEXflRESS; Initial Catalog = DBMSCuoiKy; UserId={username.Trim()}; flassword={passWord.Trim ()};");

}

private static void OpenConnection()

{

if (Connection.State == ConnectionState.Closed)

{

Connection.Open();

}

}

private static void CloseConnection()

{

if (Connection.State == ConnectionState.Open)

{

Connection.Close();

}

}

### Các chức năng chủ doanh nghiệp

* 1. **Xóa xe**

CREATE PROCEDURE sp\_DeleteXe

@MaXe INT -- Khóa chính để xác định xe cần xóa

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

-- Kiểm tra xem xe có tồn tại không

IF EXISTS (SELECT 1 FROM Xe WHERE MaXe = @MaXe and Xe.TrangThai = N'Tồn

tại')

BEGIN

-- Xóa giá thuê liên quan đến xe DELETE FROM GiaThue

WHERE MaXe = @MaXe;

-- Xóa xe DELETE FROM Xe

WHERE MaXe = @MaXe;

-- Cam kết giao dịch COMMIT TRANSACTION;

PRINT N'Xe đã được xóa thành công.';

END ELSE BEGIN

RAISERROR(N'Xe không tồn tại', 16, 1); ROLLBACK TRANSACTION;

END END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

RAISERROR(N'Xóa xe thất bại', 16, 1); END CATCH

END;

**Xử lý bên C#:**



public bool xoaXe(int maXe)

{

SqlCommand cmd = new SqlCommand("sp\_DeleteXe");

cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@MaXe", maXe);

return MyDB.Execute onQuery(cmd);

}

* 1. **Xác nhận hợp đồng**

ALTER PROCEDURE [dbo].[sp\_XacNhanHopDong] @MaHD nvarchar (50),

@TrangThaiCV nvarchar (50) = N'Chưa hoàn thành', @SoKM int = 0,

@SoXang int = 0, @TrangThai nvarchar(50),

@NgayThucHienGiaoXe Datetime, @MaNVGiaoXe int,

@NgayThucHienNhanXe Datetime, @MaNVNhanXe int

AS BEGIN

SET NOCOUNT ON;

-- Gán Ngày lập bằng GETDATE() tại đây DECLARE @MaXe int

select @MaXe = MaXe FROM HopDong WHERE MaHD = @MaHD; DECLARE @IsConflict BIT;

DECLARE @TrangThaiHD NVARCHAR(50);

SELECT @TrangThaiHD = TrangThaiHD FROM HopDong WHERE MaHD = @MaHD;

IF @TrangThaiHD <> N'Chưa xác nhận' BEGIN

RAISERROR(N'Hợp đồng đã được xác nhận, không thể thực hiện lại !', 16,

1);

RETURN;

END

-- Kiểm tra thời gian hợp lệ

IF @NgayThucHienGiaoXe >= @NgayThucHienNhanXe BEGIN

RAISERROR(N'Ngày giao xe phải trước ngày nhận xe.', 16, 1); RETURN;

END

-- Kiểm tra xung đột thời gian với hàm fn\_CheckHopDong

SET @IsConflict = dbo.fn\_CheckHopDong(@MaXe, @NgayThucHienGiaoXe, @NgayThucHienNhanXe);

IF @IsConflict = 1 BEGIN

-- Nếu có xung đột, ném ra lỗi

RAISERROR(N'Hợp đồng này xung đột với một hợp đồng khác.', 16, 1); RETURN; -- Thoát khỏi thủ tục

END

BEGIN TRY

BEGIN TRANSACTION;

INSERT INTO dbo.CongViec (TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKM, TrangThai, SoXang)

VALUES (@TrangThaiCV, @NgayThucHienGiaoXe, @MaNVGiaoXe, @MaHD, N'Giao xe', @SoKM, @TrangThai, @SoXang);

INSERT INTO dbo.CongViec (TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKM, TrangThai, SoXang)

VALUES (@TrangThaiCV, @NgayThucHienNhanXe, @MaNVNhanXe, @MaHD, N'Nhận xe', @SoKM, @TrangThai, @SoXang);

UPDATE HopDong Set TrangThaiHD = N'Đang xử lý' Where HopDong.MaHD =

@MaHD

COMMIT TRANSACTION; END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

-- Lấy thông báo lỗi và ném lại DECLARE @ErrorMessage NVARCHAR(4000); SET @ErrorMessage = ERROR\_MESSAGE(); RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END;

**Xử lý bên C#:**



**public string xac hanHopDong (HopDong hopDong)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_Xac hanHopDong"); cmd.CommandType = CommandType.Storedflrocedure;**

***// Add parameters***

**cmd.flarameters.AddWithValue("@MaHD", hopDong.MaHopDong); cmd.flarameters.AddWithValue("@TrangThai", "**Đ**ang x**ử

**l**ý**"); *// Example value***



**cmd.flarameters.AddWithValue("@ gayThucHienGiaoXe", hopDong.GiaoXe. gayThucHien);**

**cmd.flarameters.AddWithValue("@Ma VGiaoXe", hopDong.GiaoXe. hanVien.Ma V);**

**cmd.flarameters.AddWithValue("@ gayThucHien hanXe", hopDong. hanXe. gayThucHien);**

**cmd.flarameters.AddWithValue("@Ma V hanXe", hopDong. hanXe. hanVien.Ma V);**

**return MyDB.Execute onQueryReturnString(cmd);**

**}**

* 1. **Thêm nhân viên**

ALTER PROCEDURE [dbo].[sp\_AddNhanVien] @HoTen NVARCHAR(100),

@DiaChi NVARCHAR(255),

@GioiTinh NVARCHAR(10), @NgaySinh DATE,

@CMND NVARCHAR(15),

@Username NVARCHAR(50), @Password NVARCHAR(50), @LoaiNguoiDung NVARCHAR(50),

@TrangThaiNhanVien NVARCHAR(50) = N'Hoạt động' -- Giá trị mặc định

AS BEGIN

SET NOCOUNT ON;

-- Kiểm tra xem username đã tồn tại chưa

IF EXISTS (SELECT 1 FROM TaiKhoan WHERE username = @Username) BEGIN

RAISERROR(N'Username đã tồn tại', 16, 1); RETURN;

END

IF EXISTS (SELECT 1 FROM NhanVien WHERE CMND = @CMND) BEGIN

RAISERROR(N'CMND đã tồn tại', 16, 1); RETURN;

END

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

DECLARE @MaTaiKhoan INT;

-- Thêm tài khoản mới

INSERT INTO TaiKhoan (username, password, LoaiNguoiDung) VALUES (@Username, @Password, @LoaiNguoiDung);

-- Lấy ID tài khoản vừa thêm

SET @MaTaiKhoan = SCOPE\_IDENTITY();

-- Thêm nhân viên mới

INSERT INTO NhanVien (HoTen, DiaChi, GioiTinh, NgaySinh, CMND, MaTaiKhoan, TrangThai)

VALUES (@HoTen, @DiaChi, @GioiTinh, @NgaySinh, @CMND, @MaTaiKhoan, @TrangThaiNhanVien);

-- Cam kết giao dịch COMMIT TRANSACTION;

END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu xảy ra lỗi ROLLBACK TRANSACTION;

-- Lấy thông tin lỗi chi tiết từ hệ thống

DECLARE @ErrorMessage NVARCHAR(4000) = ERROR\_MESSAGE();

-- Tùy chỉnh thông báo lỗi RAISERROR(@ErrorMessage,16,1);

END CATCH

END;

**Xử lý bên C#:**



**public bool Them hanVien( hanVien nv)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_Add hanVien"); cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@HoTen", nv.HoTen); cmd.flarameters.AddWithValue("@ gaySinh", nv. gaySinh); cmd.flarameters.AddWithValue("@GioiTinh", nv.GioiTinh); cmd.flarameters.AddWithValue("@DiaChi", nv.DiaChi); cmd.flarameters.AddWithValue("@CM D", nv.CM D); cmd.flarameters.AddWithValue("@Username",**

**nv.TaiKhoan.Username);**

**cmd.flarameters.AddWithValue("@flassword", nv.TaiKhoan.flassword);**

**cmd.flarameters.AddWithValue("@Loai guoiDung", nv.TaiKhoan.Loai guoiDung);**

**cmd.flarameters.AddWithValue("@TrangThai hanVien", nv.TrangThai);**

**return MyDB.Execute onQuery(cmd);**

**}**

* 1. **Xem danh sách nhân viên**

create FUNCTION fn\_GetDanhSachNhanVien(@TrangThai NVARCHAR(50)) RETURNS TABLE

AS RETURN (

SELECT \*

FROM NhanVien

WHERE TrangThai = @TrangThai

)

**Xử lý bên C#:**



**private List < hanVien> get hanVienByTrangThai (string trangThai)**

**{**

**string queryStr = $"select \* from fn\_GetDanhSach hanVien ( '{trangThai}')";**

**DataTable dt = MyDB.GetDataTable(queryStr); return hanVienMapper.DataTableTo hanVien(dt);**

**}**

* 1. **Tìm kiếm nhân viên theo mã**

CREATE FUNCTION fn\_getNhanVienByMaNV (

@MaNV INT

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vw\_NhanVien WHERE MaNv = @MaNV

);

**Xử lý bên C#:**



**public hanVien get hanVienByMa V (int ma V)**

**{**

**string queryStr = $"select \* from fn\_get hanVienByMa V({ma V})";**

**DataTable dt = MyDB.GetDataTable(queryStr); List< hanVien> nhanViens =**

**hanVienMapper.DataTableTo hanVien(dt); if (nhanViens.Count == 0)**

**return null;**

**else return nhanViens[0];**

**}**

* 1. **Xem thanh toán**

CREATE FUNCTION fn\_XemThanhToan (

@MaHD INT = NULL, -- Mã hợp đồng, có thể NULL để xem toàn bộ @NgayBatDau DATE = NULL, -- Ngày bắt đầu, có thể NULL

@NgayKetThuc DATE = NULL -- Ngày kết thúc, có thể NULL

)

RETURNS TABLE AS

RETURN

(

-- Truy vấn thông tin thanh toán

SELECT MaTT, SoTien, NgayTT, PhuongThucTT, MaHD FROM ThanhToan

WHERE

(@MaHD IS NULL OR MaHD = @MaHD) -- Kiểm tra nếu @MaHD là NULL thì bỏ qua điều kiện này

AND

(@NgayBatDau IS NULL OR NgayTT >= @NgayBatDau) -- Kiểm tra nếu @NgayBatDau là NULL thì bỏ qua điều kiện này

AND

(@NgayKetThuc IS NULL OR NgayTT <= @NgayKetThuc) -- Kiểm tra nếu @NgayKetThuc là NULL thì bỏ qua điều kiện này

);

**Xử lý bên C#:**



public DataTable bieuDo(DateTime ngBdau, DateTime ngKthuc)

{

DataTable dt = new DataTable();

string query = "SELECT MaTT, SoTien, gayTT, flhuongThucTT, MaHD " + "FROM dbo.fn\_XemThanhToan( ULL, @ gayBatDau,

@ gayKetThuc);";

using (SqlCommand command = new SqlCommand(query, MyDB.Connection))

{

command.flarameters.AddWithValue("@ gayBatDau", ngBdau); command.flarameters.AddWithValue("@ gayKetThuc", ngKthuc);

SqlDataAdapter adapter = new SqlDataAdapter(command); adapter.Fill(dt);

}

return dt;

}

* 1. **Thêm xe**

CREATE PROCEDURE [dbo].[sp\_AddXe] @TenXe NVARCHAR(100),

@TrangThai NVARCHAR(50) = N'Tồn tại', @HinhAnh varbinary(MAX),

@PhiXang DECIMAL(10, 2),

@GiaNgay DECIMAL(10, 2), -- Giá thuê theo ngày @GiaTuan DECIMAL(10, 2), -- Giá thuê theo tuần @GiaThang DECIMAL(10, 2), -- Giá thuê theo tháng @HangXe NVARCHAR(50),

@MaLoaiXe INT, -- Loại xe là khóa ngoại @BienSoXe nvarchar(50)

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch

##### BEGIN TRY

BEGIN TRANSACTION;

-- Kiểm tra nếu hợp đồng đã có thanh toán

IF EXISTS (SELECT 1 FROM Xe WHERE BienSoXe = @BienSoXe) BEGIN

RAISERROR(N'Biển số xe đã tồn tại', 16, 1); RETURN;

##### END

-- Thêm xe mới

INSERT INTO Xe (TenXe, TrangThai, HinhAnh, PhiXang, HangXe, MaLoai, BienSoXe)

VALUES (@TenXe, @TrangThai, @HinhAnh, @PhiXang, @HangXe, @MaLoaiXe,@BienSoXe);

-- Lấy MaXe vừa được thêm DECLARE @MaXe INT;

SET @MaXe = SCOPE\_IDENTITY();

-- Thêm giá thuê cho xe theo 3 mốc thời gian INSERT INTO GiaThue (MaXe, ThoiGian, GiaThue) VALUES

(@MaXe, N'Ngày', @GiaNgay), (@MaXe, N'Tuần', @GiaTuan), (@MaXe, N'Tháng', @GiaThang);

-- Cam kết giao dịch COMMIT TRANSACTION;

PRINT N'Xe và giá thuê đã được thêm thành công.'; END TRY

##### BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Khai báo biến để lưu thông tin lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

-- Lấy thông tin lỗi SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

##### END;

-- Ném lại lỗi với thông tin chi tiết RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState);

END CATCH

### Xử lý bên C#:

**public bool themXe(Xe xe)**



**{**

**SqlCommand cmd = new SqlCommand("sp\_AddXe");**

**cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@TenXe", xe.TenXe); cmd.flarameters.AddWithValue("@TrangThai", "T**ồ**n t**ạ**i"); cmd.flarameters.Add("@HinhAnh", SqlDbType.VarBinary).Value**

**= DB ull.Value;**

**cmd.flarameters.AddWithValue("@flhiXang", xe.flhiXang); cmd.flarameters.AddWithValue("@Gia gay",**

**xe.GiaThue[0].Giathue);**

**cmd.flarameters.AddWithValue("@GiaTuan", xe.GiaThue[1].Giathue);**

**cmd.flarameters.AddWithValue("@GiaThang", xe.GiaThue[2].Giathue);**

**cmd.flarameters.AddWithValue("@HangXe", xe.HangXe); cmd.flarameters.AddWithValue("@MaLoaiXe",**

**xe.LoaiXe.MaLoai);**

**cmd.flarameters.AddWithValue("@BienSoXe", xe.BienSoXe);**

**return MyDB.Execute onQuery(cmd);**

**}**

* 1. **Tìm thanh toán thông qua mã hợp đồng**

CREATE FUNCTION fn\_GetThanhToanByMaHD(@MaHD int) RETURNS TABLE

AS RETURN (

select \* from vw\_ThanhToan where MaHD = @MaHD

);

**Xử lý bên C#:**

**public ThanhToan getThanhToanByMaHD (int maHD)**

**{**

**string queryStr = $"select \* from fn\_GetThanhToanByMaHD({maHD})";**

**DataTable dt = MyDB.GetDataTable(queryStr); List<ThanhToan> thanhtoanLst =**

**ThanhToanMapper.ConvertToThanhToanList(dt); if (thanhtoanLst.Count == 0)**

**return null;**

**else return thanhtoanLst[0];**

**}**

* 1. **Tìm kiếm hợp đồng thông qua mã hợp đồng**

CREATE FUNCTION fn\_GetHopDongByMaHD ( @MaHD int

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vwChiTietHopDong WHERE MaHD = @MaHD

);

**Xử lý bên C#:**

**public HopDong getHopDongByMaHD(int maHD)**

**{**

**string queryStr = $"select \* from fn\_GetHopDongByMaHD({maHD})";**

**DataTable dt = MyDB.GetDataTable(queryStr); List <HopDong> hopDonglst**

**= HopDongMapper.ConvertDataTableToList(dt);**

**if (hopDonglst.Count == 0) return null;**

**else return hopDonglst[0];**

**}**

* 1. **Tìm kiếm hợp đồng thông qua trạng thái**

CREATE FUNCTION fn\_getHopDongByTrangThai( @trangThai NVARCHAR(50)

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vwChiTietHopDong

WHERE vwChiTietHopDong.TrangThaiHD = @trangThai

);

**Xử lý bên C#:**



**private List<HopDong> getHopDong(string trangThaiHd)**

**{**

**string queryStr = $"select \* from fn\_getHopDongByTrangThai ( '{trangThaiHd}')";**

**DataTable dt = MyDB.GetDataTable(queryStr); return HopDongMapper.ConvertDataTableToList(dt);**

**}**

* 1. **Hàm so sánh biểu đồ(Thống kê doanh thu)**

CREATE FUNCTION fn\_CalculateDifference ( @Year1 INT,

@Year2 INT

)

RETURNS TABLE AS

RETURN (

WITH MonthlyTotals AS ( SELECT

MONTH(NgayTT) AS Month,

SUM(CASE WHEN YEAR(NgayTT) = @Year1 THEN SoTien ELSE 0 END) AS

TotalYear1,

SUM(CASE WHEN YEAR(NgayTT) = @Year2 THEN SoTien ELSE 0 END) AS

TotalYear2

FROM

ThanhToan GROUP BY

MONTH(NgayTT)

) SELECT

Month, TotalYear1, TotalYear2,

ABS(TotalYear2 - TotalYear1) AS Difference, -- Sử dụng hàm ABS để lấy giá trị tuyệt đối

CASE

WHEN TotalYear1 = 0 THEN NULL -- Tránh chia cho 0

ELSE (CAST((TotalYear2 - TotalYear1) AS FLOAT) / TotalYear1) \* 100 END AS PercentageDifference

FROM

MonthlyTotals

);

**Xử lý bên C#: thông qua 3 hàm**



**public DataTable bieuDo(DateTime ngBdau, DateTime ngKthuc)**

**{**

**DataTable dt = new DataTable();**

**string query = "SELECT MaTT, SoTien, gayTT, flhuongThucTT, MaHD " + "FROM dbo.fn\_XemThanhToan( ULL, @ gayBatDau, @ gayKetThuc);";**

**using (SqlCommand command = new SqlCommand(query, MyDB.Connection))**

**{**

**command.flarameters.AddWithValue("@ gayBatDau", ngBdau); command.flarameters.AddWithValue("@ gayKetThuc",ngKthuc);**

**SqlDataAdapter adapter = new SqlDataAdapter(command); adapter.Fill(dt);**

**}**

**return dt;**

**}**

**public DataTable soSanhBieuDo(int year1, int year2)**

**{**

**DataTable dt = new DataTable();**

**string query = "SELECT \* FROM fn\_CalculateDifference(@Year1,**

**@Year2)";**

**using (SqlCommand command = new SqlCommand(query, MyDB.Connection))**

**{**

**command.flarameters.AddWithValue("@Year1", year1); command.flarameters.AddWithValue("@Year2", year2);**

***// T****ạ****o SqlDataAdapter*** *để đ****i****ề****n d****ữ* ***li****ệ****u v****à****o DataTable***

**using (SqlDataAdapter adapter = new SqlDataAdapter(command))**

**{**

**adapter.Fill(dt);**

**}**

**}**

**return dt;**

**}**

* 1. **Xem thanh toán theo tháng**

CREATE PROCEDURE [dbo].[sp\_XemThanhToanTheoThang] @Year INT -- Năm bạn muốn xem

AS BEGIN

-- Truy vấn để lấy số tiền của từng tháng trong năm SELECT

MONTH(NgayTT) AS Thang, -- Lấy số tháng từ cột NgayTT SUM(SoTien) AS TongTien -- Tổng tiền cho mỗi tháng

FROM

ThanhToan WHERE

YEAR(NgayTT) = @Year -- Điều kiện lọc theo năm GROUP BY

MONTH(NgayTT) -- Nhóm theo tháng ORDER BY

Thang; -- Sắp xếp theo tháng

END GO

**Xử lý bên C#:**



**public DataTable bieuDoTheo am(int year)**

**{**

**DataTable dt = new DataTable();**

**string query = "EXEC sp\_XemThanhToanTheoThang @Year;";**

**using (SqlCommand command = new SqlCommand(query, MyDB.Connection))**

**{**

**command.flarameters.AddWithValue("@Year", year);**

**SqlDataAdapter adapter = new SqlDataAdapter(command); adapter.Fill(dt);**

**}**

**return dt;**

**}**

* 1. **Hàm tìm kiếm khách hàng theo mã khách hàng**

CREATE FUNCTION fn\_getKhachHangByMaKH (

@MaKH INT

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vw\_KhachHang WHERE MaKH = @MaKH

);

**Xử lý bên C#:**

**public KhachHang getKhachHangByMaKH(int maKH)**

**{**

**string queryStr = $"select \* from fn\_getKhachHangByMaKH**

**({maKH})";**

**DataTable dt = MyDB.GetDataTable(queryStr);**

**return KhachHangMapper.ConvertDataTableToKhachHang(dt);**

**}**

* 1. **Hàm tìm kiếm khách hàng theo tên**

CREATE FUNCTION fn\_TimKiemKhachHangTheoTen(@Ten NVARCHAR(100))

RETURNS @DanhSachKhachHang TABLE (

ID INT,

HoTen NVARCHAR(100), DiaChi NVARCHAR(255), GioiTinh NVARCHAR(10),

NgaySinh DATE, CMND NVARCHAR(15)

) AS

BEGIN

INSERT INTO @DanhSachKhachHang (ID, HoTen, DiaChi, GioiTinh, NgaySinh, CMND) SELECT MaKH, HoTen, DiaChi, GioiTinh, NgaySinh, CMND

FROM KhachHang

WHERE HoTen LIKE '%' + @Ten + '%'; -- Tìm kiếm tên theo từ khóa

RETURN;

END;

**Xử lý bên C#:**

**public KhachHang getKhachHangByTenKH(String tenKhachHang) { string queryStr = $"select \* from**

**fn\_TimKiemKhachHangTheoTen ('{tenKhachHang}')"; DataTable dt = MyDB.GetDataTable(queryStr);**

**return KhachHangMapper.ConvertDataTableToKhachHang(dt);**

**}**

* 1. **Lấy danh sách khách hàng**

CREATE FUNCTION fn\_GetDanhSachKhachHang(@TrangThai NVARCHAR(50)) RETURNS TABLE

AS RETURN (

select \* from KhachHang

where TrangThai = @TrangThai

);

select \* from fn\_GetDanhSachKhachHang (N'Ngừng Hoạt Động')

**Xử lý bên C#:**



**public List <KhachHang> getKhachHang()**

**{**

**string queryStr = $"select \* from fn\_GetDanhSachKhachHang ( 'Ho**ạ**t** độ**ng')";**

**DataTable dt = MyDB.GetDataTable(queryStr);**

**return KhachHangMapper.ConvertDataTableToKhachHangLst(dt);**

**}**

* 1. **Hàm tính tiền thuê xe**

CREATE FUNCTION fn\_TinhTienThue ( @MaXe INT,

@SoThoiGianThue INT, -- Số lượng thời gian thuê, ví dụ 1 năm, 2 năm @ThoiGianThue nvarchar (50)

)

RETURNS DECIMAL(10, 2) AS

BEGIN

DECLARE @TongTien DECIMAL(10, 2); DECLARE @GiaThue DECIMAL(10, 2);

SELECT @GiaThue = GiaThue FROM GiaThue WHERE MaXe = @MaXe AND ThoiGian = @ThoiGianThue; -- Lấy giá theo ngày

SET @TongTien = (@SoThoiGianThue \* @GiaThue);

RETURN @TongTien; -- Trả về tổng tiền

END; GO

**Xử lý bên C#:**



**public double tinhTienThueXe (int maXe,int soThoiGianThue, string thoiGian)**

**{**

**string query = $"SELECT dbo.fn\_TinhTienThue({maXe},**

**{soThoiGianThue}, '{thoiGian}')";**

**return MyDB.ExecuteScalarDouble(query);**

**}**

* 1. **Xóa nhân viên**

CREATE PROCEDURE [dbo].[sp\_XoaNhanVien] @MaNV INT, -- Mã nhân viên cần xóa

@MaNVThayThe INT = NULL -- Mã nhân viên thay thế (trường hợp có công việc chưa hoàn thành)

AS BEGIN

-- Bắt đầu transaction BEGIN TRANSACTION;

-- Kiểm tra xem nhân viên có công việc chưa hoàn thành không IF EXISTS (

SELECT 1

FROM CongViec

WHERE MaNV = @MaNV AND TrangThaiCV = N'Chưa hoàn thành'

) BEGIN

-- Nếu @MaNVThayThe không được cung cấp hoặc không hợp lệ, thì yêu cầu nhập

lại

IF @MaNVThayThe IS NULL OR NOT EXISTS (SELECT 1

FROM

NhanVien

WHERE MaNV = @MaNVThayThe)

BEGIN

RAISERROR ('Nhân viên có công việc chưa hoàn thành. Vui lòng chọn nhân viên khác thay thế.',16,1);

ROLLBACK TRANSACTION; RETURN;

END

-- Cập nhật công việc của nhân viên hiện tại thành nhân viên thay thế

UPDATE CongViec

SET MaNV = @MaNVThayThe

WHERE MaNV = @MaNV AND TrangThaiCV = N'Chưa hoàn thành';

END

DECLARE @MaTaiKhoan INT;

-- Lấy MaTaiKhoan của nhân viên SELECT @MaTaiKhoan = MaTaiKhoan FROM NhanVien

WHERE MaNV = @MaNV;

IF @MaTaiKhoan IS NOT NULL BEGIN

-- Xóa nhân viên DELETE FROM NhanVien WHERE MaNV = @MaNV;

DELETE FROM TaiKhoan WHERE MaTaiKhoan = @MaTaiKhoan;

COMMIT TRANSACTION;

PRINT 'Xóa nhân viên thành công.';

END ELSE BEGIN

RAISERROR(N'Nhân viên không tồn tại', 16, 1); ROLLBACK TRANSACTION;

END

-- Hoàn tất transaction

END;

**Xử lý bên C#:**



**public string Xoa hanVien(int ma V)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_Xoa hanVien"); cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@Ma V", ma V); cmd.flarameters.AddWithValue("@Ma VThayThe", nvThayThe ==**

**null o (object)DB ULL.Value : nvThayThe.Ma V );**

**return MyDB.Execute onQueryReturnString(cmd);**

**}**

### Chức năng nhân viên

* 1. **Tìm kiếm công việc thông qua trạng thái**

CREATE FUNCTION fn\_GetCongViecByTrangThai ( @TrangThai NVARCHAR(50)

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vw\_CongViec

WHERE TrangThaiCV = @TrangThai

);

**Xử lý bên C#:**



**public List<CongViec> getCongViecByTrangThai(string trangThai)**

**{**

**string queryStr = $"SELECT \* FROM dbo.fn\_GetCongViecByTrangThai( '{trangThai}')";**

**DataTable dt = MyDB.GetDataTable(queryStr); List<CongViec> congViecs =**

**CongViecMapper.ConvertToCongViecList(dt); return congViecs;**

**}**

* 1. **Xác nhận công việc hoàn thành**

CREATE PROCEDURE dbo.sp\_XacNhanCongViecHoanThanh ( @MaCongViec INT, -- Job ID

@SoKM INT, -- Distance traveled (in kilometers) @SoXang INT, -- Fuel used (in liters)

@TrangThai nvarchar(50),

@PhiThem DECIMAL(10, 2) -- Extra fee to be added to the contract

) AS

BEGIN

DECLARE @MaHD INT; -- Variable to store Contract ID

DECLARE @DeliveryStatus NVARCHAR(50); -- Variable to check delivery status

BEGIN TRY

BEGIN TRANSACTION; -- Start the transaction

-- Get the contract ID (MaHD) for the given job SELECT

@MaHD = MaHD FROM CongViec

WHERE MaCongViec = @MaCongViec;

-- Check if the job type is 'Nhận Xe'

IF EXISTS (SELECT 1 FROM CongViec WHERE MaCongViec = @MaCongViec AND LoaiCV

= N'Nhận Xe')

BEGIN

-- Check if the associated delivery job is completed SELECT

@DeliveryStatus = TrangThaiCV FROM CongViec

WHERE MaHD = @MaHD AND LoaiCV = N'Giao Xe';

IF @DeliveryStatus IS NULL OR @DeliveryStatus <> N'Đã hoàn thành' BEGIN

-- Raise an error if the delivery job is not completed RAISERROR('Công việc giao xe cho khách hàng vẫn chưa hoàn thành,

không thể xác nhận công việc này !.', 16, 1);

ROLLBACK TRANSACTION; -- Rollback the transaction

RETURN; -- Stop execution of the procedure

END

-- Update the contract with the extra fee UPDATE HopDong

SET

TongTien = TongTien + @PhiThem, -- Assuming there's a TotalFees column to store fees

PhiThem = @PhiThem -- Assuming there's a column for the extra fee WHERE

MaHD = @MaHD;

END

-- Update the job to mark it as completed UPDATE CongViec

SET

SoKM = @SoKM,

SoXang = @SoXang,

TrangThai = @TrangThai, TrangThaiCV = N'Đã hoàn thành'

WHERE

MaCongViec = @MaCongViec;

-- Update distance traveled

-- Update fuel used

-- Update job status to "Completed"

-- Where the job ID matches

COMMIT TRANSACTION; -- Commit the transaction if all is successful

END TRY BEGIN CATCH

ROLLBACK TRANSACTION; -- Rollback the transaction on error DECLARE @ErrorMessage NVARCHAR(4000);

DECLARE @ErrorSeverity INT; DECLARE @ErrorState INT;

SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); -- Raise the error END CATCH

END;

**Xử lý bên C#:**



**public string xac hanCongViec (CongViec congViec, double phiThem)**

**{**

**SqlCommand command = new SqlCommand("sp\_Xac hanCongViecHoanThanh");**

**command.CommandType = CommandType.Storedflrocedure;**

***// Thêm các tham s****ố* ***v****à****o command***

**command.flarameters.AddWithValue("@MaCongViec", congViec.MaCongViec);**

**command.flarameters.AddWithValue("@SoKM", congViec.MaCongViec);**



**command.flarameters.AddWithValue("@SoXang", congViec.SoXang);**

**command.flarameters.AddWithValue("@flhiThem", phiThem); command.flarameters.AddWithValue("@TrangThai",**

**congViec.TrangThai);**

**return MyDB.Execute onQueryReturnString(command);**

**}**

* 1. **Tính toán chi phí thêm**

CREATE FUNCTION dbo.fn\_TinhToanPhiThem ( @MaHD INT, -- Contract ID @SoXangKhiNhan INT

)

RETURNS DECIMAL(10, 2) AS

BEGIN

DECLARE @PhiThem DECIMAL(10, 2) = 0;

DECLARE @SoXangBanDau INT; DECLARE @PhiXang DECIMAL(10, 2); DECLARE @MaXe INT;

-- Get initial fuel level from the contract with LoaiCV = 'Giao Xe' SELECT

@SoXangBanDau = CongViec.SoXang FROM CongViec

WHERE CongViec.MaHD = @MaHD AND CongViec.LoaiCV = N'Giao Xe'; print (@SoXangBanDau)

-- Get the vehicle ID (MaXe) from the contract SELECT

@MaXe = HopDong.MaXe FROM HopDong

WHERE HopDong.MaHD = @MaHD;

-- Get the fuel cost (PhiXang) from the vehicle SELECT

@PhiXang = Xe.PhiXang FROM Xe

WHERE Xe.MaXe = @MaXe; print (@PhiXang)

-- Calculate extra fee if the fuel level on return exceeds the initial fuel level

IF (@SoXangKhiNhan > @SoXangBanDau) BEGIN

-- Calculate extra fuel fee

SET @PhiThem = (@SoXangKhiNhan - @SoXangBanDau) \* @PhiXang;

END

RETURN @PhiThem;

END;

**Xử lý bên C#:**



**public double tinhflhiThem(int soXangKhi han, int maHD)**

**{**

**string query = $"SELECT dbo.fn\_TinhToanflhiThem({maHD},{soXangKhi han})";**

**return MyDB.ExecuteScalarDouble(query);**

**}**

* 1. **tìm kiếm công việc thông qua mã nhân viên**

CREATE FUNCTION dbo.fn\_GetCongViecByMaNV (

@MaNV int,

@TrangThaiCV nvarchar (50),

@LoaiCV nvarchar (50)

)

RETURNS TABLE AS

RETURN (

-- Select the row from CongViec where MaCongViec matches @MaCV

SELECT MaCongViec, TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKM, TrangThai, SoXang

FROM vw\_CongViec

WHERE MaNV = @MaNV and TrangThaiCV = @TrangThaiCV and LoaiCV = @LoaiCV

);

**Xử lý bên C#:**



**public List<CongViec> getCongViecByMa V(int ma V, string trangThaiCV, string loaiCV)**

**{**

**string queryStr = $"SELECT \* FROM dbo.fn\_GetCongViecByMa V({ma V}, '{trangThaiCV}', '{loaiCV}')";**

**DataTable dt = MyDB.GetDataTable(queryStr); List<CongViec> congViecs =**

**CongViecMapper.ConvertToCongViecList(dt); return congViecs;**

**}**

* 1. **Tìm kiếm công việc thông qua trạng thái**

CREATE FUNCTION fn\_GetCongViecByTrangThai ( @TrangThai NVARCHAR(50)

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vw\_CongViec

WHERE TrangThaiCV = @TrangThai

);

**Xử lý bên C#:**



**public List<CongViec> getCongViecByTrangThai(string trangThai)**

**{**

**string queryStr = $"SELECT \* FROM dbo.fn\_GetCongViecByTrangThai( '{trangThai}')";**

**DataTable dt = MyDB.GetDataTable(queryStr); List<CongViec> congViecs =**

**CongViecMapper.ConvertToCongViecList(dt); return congViecs;**

**}**

* 1. **Tìm kiếm công việc qua mã công việc**

CREATE FUNCTION dbo.fn\_GetCongViecByMaCV (

@MaCV INT -- Input parameter to specify the job's ID

)

RETURNS TABLE AS

RETURN (

-- Select the row from CongViec where MaCongViec matches @MaCV

SELECT MaCongViec, TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKM, TrangThai, SoXang

FROM CongViec

WHERE MaCongViec = @MaCV

);

**Xử lý bên C#:**

**public CongViec getCongViecByMaCV(int maCV)**

**{**

**string queryStr = $"SELECT \* FROM dbo.fn\_GetCongViecByMaCV({maCV})";**

**DataTable dt = MyDB.GetDataTable(queryStr); List <CongViec> congViecs =**

**CongViecMapper.ConvertToCongViecList(dt); if (congViecs.Count == 0 )**

**return null; return congViecs[0];**

**}**

* 1. **Cập nhật nhân viên**

ALTER PROCEDURE [dbo].[sp\_UpdateNhanVien]

@MaNV INT, -- Khóa chính để xác định nhân viên @HoTen NVARCHAR(100) = NULL,

@DiaChi NVARCHAR(255) = NULL, @GioiTinh NVARCHAR(10) = NULL,

@NgaySinh DATE = NULL, @CMND NVARCHAR(15) = NULL,

@MaTaiKhoan INT = NULL,

@TrangThai NVARCHAR(50) = NULL

AS BEGIN

SET NOCOUNT ON;

BEGIN TRY

-- Bắt đầu giao dịch BEGIN TRANSACTION;

-- Kiểm tra xem MaNV có tồn tại trong bảng NhanVien không IF NOT EXISTS (SELECT 1 FROM NhanVien WHERE MaNV = @MaNV) BEGIN

RAISERROR('Không tìm thấy nhân viên với MaNV: %d', 16, 1, @MaNV); ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật thông tin nhân viên UPDATE NhanVien

SET

HoTen = COALESCE(@HoTen, HoTen), -- Nếu @HoTen là NULL, giữ nguyên giá

trị hiện tại

DiaChi = COALESCE(@DiaChi, DiaChi), GioiTinh = COALESCE(@GioiTinh, GioiTinh), NgaySinh = COALESCE(@NgaySinh, NgaySinh), CMND = COALESCE(@CMND, CMND),

MaTaiKhoan = COALESCE(@MaTaiKhoan, MaTaiKhoan), TrangThai = COALESCE(@TrangThai, TrangThai)

WHERE MaNV = @MaNV;

-- Cam kết giao dịch nếu không có lỗi COMMIT TRANSACTION;

END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Lấy thông tin lỗi và ném lại lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); END CATCH

END;

**Xử lý bên C#:**



**public bool update hanVien( hanVien nv)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_Update hanVien"); cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@Ma V", nv.Ma V); cmd.flarameters.AddWithValue("@HoTen", nv.HoTen); cmd.flarameters.AddWithValue("@ gaySinh", nv. gaySinh); cmd.flarameters.AddWithValue("@GioiTinh", nv.GioiTinh);**



**}**

**cmd.flarameters.AddWithValue("@DiaChi", nv.DiaChi); cmd.flarameters.AddWithValue("@CM D", nv.CM D); cmd.flarameters.AddWithValue("@TrangThai", nv.TrangThai);**

**return MyDB.Execute onQuery(cmd);**

* 1. **Thêm sự cố**

CREATE PROCEDURE sp\_ThemSuCo @TenSC NVARCHAR(255), @LyDo NVARCHAR(255), @NoiDung NVARCHAR(255),

@TienPhat DECIMAL(18, 2), @TrangThai NVARCHAR(50), @MaHD INT

AS BEGIN

-- Insert the new incident into the table

INSERT INTO SuCo (TenSC, LyDo, NoiDung, TienPhat, TrangThai, MaHD) VALUES (@TenSC, @LyDo, @NoiDung, @TienPhat, @TrangThai, @MaHD);

-- Optionally, return the newly created incident ID SELECT SCOPE\_IDENTITY() AS NewIncidentID;

END;

**Xử lý bên C#:**

|  |  |  |
| --- | --- | --- |
| public string themSuCo (SuCo suCo)  {  SqlCommand cmd = new SqlCommand("sp\_ThemSuCo"); cmd.CommandType = CommandType.Storedflrocedure;  // Thêm các tham số vÃo stored procedure cmd.flarameters.AddWithValue("@TenSC", suCo.TenSC); cmd.flarameters.AddWithValue("@LyDo", suCo.LyDO); cmd.flarameters.AddWithValue("@oiDung", suCo.oiDung); cmd.flarameters.AddWithValue("@Tienflhat", suCo.Tienflhat); cmd.flarameters.AddWithValue("@MaHD", suCo.HopDong.MaHopDong); cmd.flarameters.AddWithValue("@TrangThai", suCo.TrangThai);  return MyDB.ExecuteonQueryReturnString(cmd); | | |
|  | } |  |

* 1. **Tìm kiếm sự cố thông qua mã hợp đồng**

CREATE FUNCTION fn\_getSuCoByMaHD (@MaHD INT) RETURNS TABLE

AS RETURN (

select \* from vw\_SuCo as sc where sc.MaHD = @MaHD

);

**Xử lý bên C#:**

public List <SuCo> getSuCobyMaHD (int maHD)

{

string queryStr = $"select \* from fn\_getSuCoByMaHD({maHD})"; DataTable dt = MyDB.GetDataTable(queryStr);

Return SuCoMapper.ConvertDataTableToList(dt);

}

## Chức năng của khách hàng

* 1. **Tìm kiếm hợp đồng thông qua khách hàng**

CREATE FUNCTION dbo.fn\_getHopDongByMaKH (

@MaKH NVARCHAR(50),

@TrangThai NVARCHAR (50)

)

RETURNS TABLE AS

RETURN (

SELECT \*

FROM vwChiTietHopDong

WHERE MaKH = @MaKH and TrangThaiHD = @TrangThai

);

**Xử lý bên C#:**



**public List<HopDong> getHopDongbyMaKH(int maKH,string trangThaiHd)**

**{**

**string queryStr = $"select \* from fn\_getHopDongByMaKH ({maKH}, '{trangThaiHd}')";**

**DataTable dt = MyDB.GetDataTable(queryStr); return HopDongMapper.ConvertDataTableToList(dt);**

**}**

* 1. **Hủy hợp đồng**

CREATE PROCEDURE sp\_HuyHopDong (

@MaHD INT -- Mã hợp đồng

) AS

BEGIN

-- Kiểm tra xem hợp đồng có tồn tại và trạng thái là "Chưa xác nhận" không IF EXISTS (SELECT 1 FROM HopDong WHERE MaHD = @MaHD AND TrangThaiHD = N'Chưa

xác nhận')

BEGIN

-- Nếu hợp đồng có trạng thái "Chưa xác nhận", thì cập nhật trạng thái mới

UPDATE HopDong

SET TrangThaiHD = N'Đã hủy' WHERE MaHD = @MaHD;

PRINT 'Hủy hợp đồng thành công !';

END ELSE BEGIN

-- Nếu trạng thái không phải là "Chưa xác nhận", không cho phép cập nhật RAISERROR(N'Hợp đồng này đã được xác nhận và xử lý, không thể hủy.', 16,

1);

END

END;

**Xử lý bên C#:**



**public string huyHopDong (int maHD)**

**{**

**SqlCommand command = new SqlCommand("sp\_HuyHopDong"); command.CommandType = CommandType.Storedflrocedure;**

***// Thêm các tham s****ố* ***v****à****o stored procedure***

**command.flarameters.Add(new Sqlflarameter("@MaHD", SqlDbType.Int)).Value = maHD;**

**return MyDB.Execute onQueryReturnString(command);**

**}**

* 1. **Thanh toán hợp đồng**

CREATE PROCEDURE sp\_ThanhToanHopDong

@MaHD INT, -- Mã hợp đồng @SoTien DECIMAL(10, 2), -- Số tiền thanh toán @NgayTT DATE, -- Ngày thanh toán

@PhuongThucTT NVARCHAR(50) -- Phương thức thanh toán

AS BEGIN

BEGIN TRY

BEGIN TRANSACTION;

INSERT INTO ThanhToan (SoTien, NgayTT, PhuongThucTT, MaHD) VALUES (@SoTien, @NgayTT, @PhuongThucTT, @MaHD);

-- Cập nhật trạng thái hợp đồng thành 'Đã hoàn thành' UPDATE HopDong

SET TrangThaiHD = N'Đã hoàn thành' WHERE MaHD = @MaHD;

-- Commit giao dịch COMMIT TRANSACTION;

END TRY BEGIN CATCH

ROLLBACK TRANSACTION;

-- Thông báo lỗi

DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

SELECT @ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR (@ErrorMessage, @ErrorSeverity, @ErrorState); END CATCH

END;

**Xử lý bên C#:**



**public string thanhToanHopDong (ThanhToan thanhToan)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_ThanhToanHopDong"); cmd.CommandType = CommandType.Storedflrocedure;**

**cmd.flarameters.AddWithValue("@MaHD", thanhToan.MaHD); cmd.flarameters.AddWithValue("@SoTien", thanhToan.SoTien); cmd.flarameters.AddWithValue("@ gayTT", thanhToan. gayTT); cmd.flarameters.AddWithValue("@flhuongThucTT",**

**thanhToan.flhuongThucTT);**

**return MyDB.Execute onQueryReturnString(cmd);**

**}**

* 1. **Kiểm tra trạng thái hợp đồng**

ALTER FUNCTION [dbo].[fn\_CheckHopDong] ( @MaXe INT,

@NgayGiaoXe DATETIME, @NgayNhanXe DATETIME

)

RETURNS BIT AS

BEGIN

DECLARE @Conflict BIT = 0;

-- Kiểm tra xem hợp đồng mới có xung đột thời gian với bất kỳ hợp đồng nào khác không

IF EXISTS (

SELECT 1

FROM HopDong

WHERE not ( @NgayNhanXe < HopDong.NgayGiaoXe or @NgayGiaoXe > HopDong.NgayNhanXe )

and HopDong.MaXe = @MaXe and HopDong.TrangThaiHD = N'Đang xử

lý'

) BEGIN

SET @Conflict = 1; -- Có xung đột

END

RETURN @Conflict; -- Trả về 1 nếu có xung đột, ngược lại 0

END;

* 1. **Thêm hợp đồng của khách hàng**

CREATE PROCEDURE sp\_AddHopDong @NoiDung NVARCHAR(255),

@TongTien DECIMAL(10, 2),

@PhiThem DECIMAL(10, 2),

@NgayGiaoXe DATETIME, @NgayNhanXe DATETIME, @MaKH INT,

@MaXe INT,

@TrangThaiHD NVARCHAR(50) = N'Chưa xác nhận' -- Giá trị mặc định

##### AS BEGIN

SET NOCOUNT ON;

-- Gán Ngày lập bằng GETDATE() tại đây DECLARE @NgayLap DATETIME = GETDATE();

DECLARE @IsConflict BIT;

-- Kiểm tra xung đột thời gian với hàm fn\_CheckHopDong

SET @IsConflict = dbo.fn\_CheckHopDong(@MaXe, @NgayGiaoXe, @NgayNhanXe);

IF @IsConflict = 1 BEGIN

-- Nếu có xung đột, ném ra lỗi

RAISERROR(N'Hợp đồng này xung đột với một hợp đồng khác.', 16, 1); RETURN; -- Thoát khỏi thủ tục

END

BEGIN TRY

BEGIN TRANSACTION;

INSERT INTO HopDong (NoiDung, TongTien, PhiThem, NgayLap, NgayGiaoXe, NgayNhanXe, MaKH, MaXe, TrangThaiHD)

VALUES (@NoiDung, @TongTien, @PhiThem, @NgayLap, @NgayGiaoXe, @NgayNhanXe, @MaKH, @MaXe, @TrangThaiHD);

COMMIT TRANSACTION; END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

END; GO

-- Lấy thông báo lỗi và ném lại DECLARE @ErrorMessage NVARCHAR(4000); SET @ErrorMessage = ERROR\_MESSAGE(); RAISERROR(@ErrorMessage, 16, 1);

END CATCH

**Xử lý bên C#:**



**public string taoHopDong (HopDong hopDong)**

**{**

**SqlCommand command = new SqlCommand("sp\_AddHopDong"); command.CommandType = CommandType.Storedflrocedure;**

***// Thêm các tham s****ố* ***v****à****o stored procedure***

**command.flarameters.AddWithValue("@ oiDung", hopDong. oidung);**

**command.flarameters.AddWithValue("@TongTien", hopDong.TongTien);**

**command.flarameters.AddWithValue("@flhiThem", hopDong.flhiThem);**

**command.flarameters.AddWithValue("@ gayGiaoXe", hopDong. gayGiaoXe);**

**command.flarameters.AddWithValue("@ gay hanXe", hopDong. gay hanXe);**

**command.flarameters.AddWithValue("@MaKH", hopDong.KhachHang.MaKH);**

**command.flarameters.AddWithValue("@MaXe", hopDong.Xe.MaXe); return MyDB.Execute onQueryReturnString (command);**

**}**

* 1. **Thêm khách hàng**

CREATE PROCEDURE sp\_AddKhachHang @HoTen NVARCHAR(100), @DiaChi NVARCHAR(255),

@GioiTinh NVARCHAR(10), @NgaySinh DATE,

@CMND NVARCHAR(15),

@Username NVARCHAR(50), @Password NVARCHAR(50), @LoaiNguoiDung NVARCHAR(50),

@TrangThaiKhachHang NVARCHAR(50) = N'Hoạt động' -- Giá trị mặc định

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

DECLARE @MaTaiKhoan INT;

-- Thêm tài khoản mới

INSERT INTO TaiKhoan (username, password, LoaiNguoiDung) VALUES (@Username, @Password, @LoaiNguoiDung);

-- Lấy ID tài khoản vừa thêm

SET @MaTaiKhoan = SCOPE\_IDENTITY();

-- Thêm khách hàng mới

INSERT INTO KhachHang (HoTen, DiaChi, GioiTinh, NgaySinh, CMND, MaTaiKhoan, TrangThai)

VALUES (@HoTen, @DiaChi, @GioiTinh, @NgaySinh, @CMND, @MaTaiKhoan, @TrangThaiKhachHang);

-- Cam kết giao dịch COMMIT TRANSACTION;

END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Khai báo biến để lưu thông tin lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

-- Lấy thông tin lỗi SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

-- Ném lại lỗi với thông tin chi tiết RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState);

END CATCH

END; GO

**Xử lý bên C#:**



**public bool themKhachHang(KhachHang khachHang)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_AddKhachHang"); cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@HoTen", khachHang.HoTen); cmd.flarameters.AddWithValue("@DiaChi", khachHang.DiaChi); cmd.flarameters.AddWithValue("@GioiTinh",**

**khachHang.GioiTinh);**

**cmd.flarameters.AddWithValue("@ gaySinh", khachHang. gaySinh);**

**cmd.flarameters.AddWithValue("@CM D", khachHang.CM D); cmd.flarameters.AddWithValue("@Username",**

**khachHang.TaiKhoan.Username);**

**cmd.flarameters.AddWithValue("@flassword", khachHang.TaiKhoan.flassword);**

**cmd.flarameters.AddWithValue("@Loai guoiDung", khachHang.TaiKhoan.Loai guoiDung);**

**cmd.flarameters.AddWithValue("@TrangThaiKhachHang", khachHang.TrangThai);**

**return MyDB.Execute onQuery(cmd);**

**}**

* 1. **Cập nhật khách hàng**

CREATE PROCEDURE sp\_UpdateKhachHang

@MaKH INT, -- Khóa chính để xác định khách hàng @HoTen NVARCHAR(100) = NULL,

@DiaChi NVARCHAR(255) = NULL, @GioiTinh NVARCHAR(10) = NULL,

@NgaySinh DATE = NULL, @CMND NVARCHAR(15) = NULL,

@Username NVARCHAR(50) = NULL, @Password NVARCHAR(50) = NULL,

@LoaiNguoiDung NVARCHAR(50) = NULL, @TrangThaiKhachHang NVARCHAR(50) = NULL

AS BEGIN

SET NOCOUNT ON;

BEGIN TRY

-- Bắt đầu giao dịch BEGIN TRANSACTION;

-- Kiểm tra xem MaKH có tồn tại trong bảng KhachHang không IF NOT EXISTS (SELECT 1 FROM KhachHang WHERE MaKH = @MaKH) BEGIN

RAISERROR('Không tìm thấy khách hàng với MaKH: %d', 16, 1, @MaKH); ROLLBACK TRANSACTION;

##### RETURN;

END

-- Cập nhật thông tin khách hàng UPDATE KhachHang

##### SET

trị hiện tại

HoTen = COALESCE(@HoTen, HoTen), -- Nếu @HoTen là NULL, giữ nguyên giá

DiaChi = COALESCE(@DiaChi, DiaChi), GioiTinh = COALESCE(@GioiTinh, GioiTinh), NgaySinh = COALESCE(@NgaySinh, NgaySinh), CMND = COALESCE(@CMND, CMND),

TrangThai = COALESCE(@TrangThaiKhachHang, TrangThai)

WHERE MaKH = @MaKH;

NULL

-- Cập nhật thông tin tài khoản nếu có Username và Password

IF @Username IS NOT NULL OR @Password IS NOT NULL OR @LoaiNguoiDung IS NOT

##### BEGIN

UPDATE TaiKhoan SET

username = COALESCE(@Username, username), -- Nếu @Username là NULL, giữ nguyên giá trị hiện tại

password = COALESCE(@Password, password), LoaiNguoiDung = COALESCE(@LoaiNguoiDung, LoaiNguoiDung)

WHERE MaTaiKhoan = (SELECT MaTaiKhoan FROM KhachHang WHERE MaKH =

@MaKH);

END

-- Cam kết giao dịch nếu không có lỗi COMMIT TRANSACTION;

END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Lấy thông tin lỗi và ném lại lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); END CATCH

END; GO

**Xử lý bên C#:**



**public bool updateKhachHang(KhachHang khachHang)**

**{**

**SqlCommand cmd = new SqlCommand("sp\_updateKhachHang"); cmd.CommandType = CommandType.Storedflrocedure; cmd.flarameters.AddWithValue("@MaKH", khachHang.MaKH); cmd.flarameters.AddWithValue("@HoTen", khachHang.HoTen); cmd.flarameters.AddWithValue("@DiaChi", khachHang.DiaChi); cmd.flarameters.AddWithValue("@GioiTinh",**

**khachHang.GioiTinh);**

**cmd.flarameters.AddWithValue("@ gaySinh", khachHang. gaySinh);**

**cmd.flarameters.AddWithValue("@CM D", khachHang.CM D); cmd.flarameters.AddWithValue("@Username",**

**khachHang.TaiKhoan.Username);**

**cmd.flarameters.AddWithValue("@flassword", khachHang.TaiKhoan.flassword);**

**cmd.flarameters.AddWithValue("@Loai guoiDung", khachHang.TaiKhoan.Loai guoiDung);**

**cmd.flarameters.AddWithValue("@TrangThaiKhachHang", khachHang.TrangThai);**

**return MyDB.Execute onQuery(cmd);**

**}**

* 1. **Thanh toán sự cố**

CREATE PROCEDURE sp\_ThanhToanSuCo @MaSC INT -- Incident ID

AS BEGIN

-- Update the incident record, marking it as resolved UPDATE SuCo

SET TrangThai = 'Đã giải quyết' WHERE MaSC = @MaSC;

END;

**Xử lý bên C#:**

public string thanhToanSuCo (int maSC)

{

SqlCommand cmd = new SqlCommand("sp\_ThanhToanSuCo"); cmd.CommandType = CommandType.StoredProcedure; cmd.Parameters.AddWithValue("@MaSC", maSC);

return MyDB.ExecuteNonQueryReturnString(cmd);

}

* 1. **Tìm tài khoản thông qua mã khách hàng**

CREATE FUNCTION fn\_getTaiKhoanByMaTaiKhoan(@MaTaiKhoan int) RETURNS TABLE

AS RETURN (

select \* from TaiKhoan

where MaTaiKhoan = @MaTaiKhoan

);

**Xử lý bên C#:**

**public TaiKhoan getTaiKhoanByMaTaiKhoan (int maTaiKhoan)**

**{**

**string queryStr = $"select \* from fn\_getTaiKhoanByMaTaiKhoan({maTaiKhoan})";**

**DataTable dt = MyDB.GetDataTable(queryStr);**

**return TaiKhoanMapper.ConvertDataTableToTaiKhoan(dt);**

**}**

* 1. **Tìm kiếm xe thông qua thời gian, trạng thái, tên loại**

CREATE FUNCTION fn\_LayChiTietXeByThoiGian\_TrangThai\_TenLoai (

@TrangThai NVARCHAR(50) = NULL, -- Tham số trạng thái, có thể NULL @ThoiGian NVARCHAR(50) = NULL, -- Tham số thời gian thuê, có thể NULL

@TenLoai NVARCHAR(50) = NULL

)

RETURNS TABLE AS

RETURN (

SELECT

\*

FROM vwChiTietXe WHERE

(@TrangThai IS NULL OR TrangThai = @TrangThai) -- Điều kiện kiểm tra trạng

thái

AND (@ThoiGian IS NULL OR ThoiGian = @ThoiGian) -- Điều kiện kiểm tra thời

gian

AND (@TenLoai IS NULL OR TenLoai = @TenLoai)

);

**Xử lý bên C#:**



**public List<Xe> getXeThueByTrangThai\_ThoiGian\_TenLoai(string trangThai, string thoigian, string tenLoai )**

**{**

**string query = $"select \* from fn\_LayChiTietXeByThoiGian\_TrangThai\_TenLoai(@trangThai, @thoigian, @tenLoai)";**

**SqlCommand cmd = new SqlCommand(query); cmd.flarameters.AddWithValue("@trangThai",**

**(object)trangThai oo DB ull.Value);**

**cmd.flarameters.AddWithValue("@thoigian", (object)thoigian oo DB ull.Value);**

**cmd.flarameters.AddWithValue("@tenLoai", (object)tenLoai oo DB ull.Value);**

**DataTable dt = MyDB.GetDataTable(cmd); List<Xe> list = new List<Xe>(); foreach (DataRow dr in dt.Rows) {**

**Xe xe = XeMapper.ConvertDataTableToXe(dr);**

**list.Add(xe);**

**}**

**return list;**

**}**

* 1. **Tìm kiếm xe thông qua mã xe**

CREATE FUNCTION fn\_getXe (@maXe INT) RETURNS TABLE

AS RETURN (

SELECT

\*

FROM vwChiTietXe WHERE MaXe = @maXe

);

**Xử lý bên C#:**

**public Xe getXebyId (int maXe)**

**{**

**string queryStr = $"select \* from fn\_getXe ({maXe})"; DataTable dt = MyDB.GetDataTable(queryStr);**

**return XeMapper.ConvertDataTableToXe(dt);**

**}**

### CHƯƠNG 4: PHÂN QUYỀN

### Tạo Role và gán quyền

* + Hệ thống sẽ bao gồm 3 Role được định nghĩa trong bảng Account bao gồm: Chủ doanh nghiệp, nhân viên và khách hàng.
  + Tương ứng sẽ tạo 3 Role trong SQL Server và thực hiện phân quyền cho các Role này: sysadmin (role mà hệ thống hỗ trợ), NhanVien và KhachHang.
  + Tạo Role và gán quyền cho role NhanVien trên các func, proc:

GRANT EXECUTE ON [dbo].[sp\_XacNhanCongViecHoanThanh] TO NhanVien GRANT EXECUTE ON [dbo].[sp\_ThemSuCo] TO NhanVien

GRANT EXECUTE ON fn\_CheckChuoiRong TO NhanVien GRANT EXECUTE ON fn\_CheckLaIntOrDecimal TO NhanVien GRANT EXECUTE ON sp\_UpdateNhanVien TO NhanVien

GRANT SELECT ON dbo.fn\_getNhanVienByUsername TO NhanVien; GRANT SELECT ON fn\_GetHopDongByMaHD TO NhanVien;

GRANT SELECT ON fn\_GetCongViecByMaNV TO NhanVien GRANT SELECT ON fn\_getSuCoByMaHD TO NhanVien GRANT SELECT ON fn\_GetHopDongByMaHD TO NhanVien GRANT SELECT ON fn\_getXe TO NhanVien

GRANT SELECT ON fn\_getKhachHangByMaKH TO NhanVien

* + Tạo Role và gán quyền cho role KhachHang trên các func, proc:

GRANT EXECUTE ON [dbo].[sp\_ThanhToanHopDong] TO KhachHang GRANT EXECUTE ON [dbo].[sp\_UpdateKhachHang] TO KhachHang GRANT EXECUTE ON [dbo].[sp\_ThanhToanSuCo] TO KhachHang GRANT EXECUTE ON [dbo].[fn\_TinhTienThue] TO KhachHang GRANT EXECUTE ON fn\_CheckChuoiRong TO KhachHang

GRANT EXECUTE ON sp\_AddHopDong TO KhachHang GRANT EXECUTE ON sp\_HuyHopDong TO KhachHang GRANT EXECUTE ON sp\_HuyHopDong TO KhachHang

GRANT EXECUTE ON fn\_CheckLaIntOrDecimal TO KhachHang GRANT SELECT ON fn\_getKhachHangByUsername TO KhachHang

GRANT SELECT ON fn\_LayChiTietXeByThoiGian\_TrangThai\_TenLoai TO KhachHang GRANT SELECT ON GetRentalPriceByCarId TO KhachHang

GRANT SELECT ON fn\_getHopDongByMaKH TO KhachHang GRANT SELECT ON fn\_getXe TO KhachHang

GRANT SELECT ON fn\_GetCongViecByMaHD TO KhachHang GRANT SELECT ON fn\_getNhanVienByMaNV TO KhachHang GRANT SELECT ON fn\_getSuCoByMaHD TO KhachHang

* + Gán quyền cho các bảng:

-- Gán quyền vào các bảng cho 3 ROLE

GRANT SELECT, UPDATE ON dbo.TaiKhoan TO NhanVien; GRANT SELECT, INSERT, UPDATE ON dbo.SuCo TO NhanVien;

GRANT SELECT, UPDATE ON dbo.HopDong TO NhanVien; GRANT SELECT ON dbo.CongViec TO NhanVien;

GRANT SELECT, UPDATE ON dbo.TaiKhoan TO KhachHang;

GRANT SELECT,INSERT, UPDATE ON dbo.KhachHang TO KhachHang; GRANT SELECT ON dbo.Xe TO KhachHang;

GRANT SELECT, INSERT ON dbo.ThanhToan TO KhachHang; GRANT SELECT ON dbo.Xe TO KhachHang;

* + Trigger tạo tài khoản:

ALTER TRIGGER [dbo].[trg\_CreateSQLAccount] ON [dbo].[TaiKhoan]

AFTER INSERT AS

DECLARE @username nvarchar(50), @password nvarchar(50), @loainguoidung nvarchar(50);

SELECT @username = nl.username, @password = nl.password, @loainguoidung = nl.LoaiNguoiDung

FROM inserted nl;

BEGIN

DECLARE @sqlString nvarchar(MAX), @macv nvarchar(10);

-- Tạo login

SET @sqlString = 'CREATE LOGIN [' + @username + '] WITH PASSWORD = ''' + @password + ''', DEFAULT\_DATABASE = [DBMSCuoiKy], CHECK\_EXPIRATION = OFF, CHECK\_POLICY = OFF';

EXEC (@sqlString);

-- Tạo user

SET @sqlString = 'CREATE USER [' + @username + '] FOR LOGIN [' + @username +

']';

EXEC (@sqlString);

-- Gán vai trò cho user

IF (@loainguoidung = N'Nhân viên') BEGIN

SET @sqlString = 'ALTER ROLE [NhanVien] ADD MEMBER [' + @username + ']';

END

ELSE IF (@loainguoidung = N'Khách hàng') BEGIN

SET @sqlString = 'ALTER ROLE [KhachHang] ADD MEMBER [' + @username + ']';

END

EXEC (@sqlString);

END;

* + Trigger xóa tài khoản:

ALTER TRIGGER [dbo].[trg\_DeleteSQLAccount] ON [dbo].[TaiKhoan]

AFTER DELETE AS

BEGIN

SET NOCOUNT ON;

DECLARE @username nvarchar(50); DECLARE @SessionID INT;

DECLARE @sql nvarchar(MAX);

-- Lấy tên đăng nhập (username) từ bảng TaiKhoan bị xóa SELECT @username = deleted.username

FROM deleted;

-- Kiểm tra nếu user đang đăng nhập và nếu có thì kill session đó SELECT @SessionID = session\_id

FROM sys.dm\_exec\_sessions WHERE login\_name = @username;

IF @SessionID IS NOT NULL BEGIN

SET @sql = 'KILL ' + CONVERT(nvarchar(20), @SessionID); EXEC (@sql);

END

-- Xóa USER và LOGIN trong SQL Server BEGIN TRY

SET @sql = 'DROP USER [' + @username + ']'; EXEC (@sql);

SET @sql = 'DROP LOGIN [' + @username + ']'; EXEC (@sql);

END TRY BEGIN CATCH

-- Xử lý lỗi nếu việc xóa user hoặc login thất bại DECLARE @err NVARCHAR(MAX);

SELECT @err = N'Lỗi khi xóa tài khoản: ' + ERROR\_MESSAGE(); RAISERROR(@err, 16, 1);

END CATCH

END;

* + Func đăng nhập:

ALTER FUNCTION [dbo].[fn\_LoginAndGetRole] (

@username NVARCHAR(50), @password NVARCHAR(50)

)

RETURNS NVARCHAR(50) -- Trả về tên vai trò hoặc thông báo lỗi AS

BEGIN

DECLARE @role NVARCHAR(50);

DECLARE @isValid BIT;

-- Kiểm tra username và password có hợp lệ không SELECT @isValid = CASE

WHEN EXISTS (

SELECT 1

FROM TaiKhoan

WHERE username = @username

AND TaiKhoan.password = @password

)

THEN 1

ELSE 0 END;

IF @isValid = 1 BEGIN

-- Nếu đăng nhập hợp lệ, lấy vai trò của user SELECT @role = dp.name

FROM sys.database\_role\_members AS drm

JOIN sys.database\_principals AS dp ON drm.role\_principal\_id = dp.principal\_id

JOIN sys.database\_principals AS dp2 ON drm.member\_principal\_id = dp2.principal\_id

WHERE dp2.name = @username;

-- Nếu user không thuộc vai trò nào thì trả về thông báo không có vai trò IF @role IS NULL

BEGIN

SET @role = N'User này không có role nào';

END

END

ELSE BEGIN

-- Nếu đăng nhập không hợp lệ, trả về thông báo đăng nhập thất bại SET @role = N'Đăng nhập thất bại';

END

RETURN @role;

END;

* + Xử lý đăng nhập phía C#:

public string login(string username, string password)

{

MyDB.SetMyDB("phamtienanh", "1");

string sqlQuery = "SELECT dbo.fn\_LoginAndGetRole(@username, @password)"; SqlCommand cmd = new SqlCommand(sqlQuery); cmd.flarameters.AddWithValue("@username", username); cmd.flarameters.AddWithValue("@password", password);

cmd.CommandType = CommandType.Text;

return MyDB.ExecuteScalar(cmd);

}



private void btnLogin\_Click(object sender, EventArgs e)

{

string username = txtUsername.Text; string password = txtflassword.Text;

string role = dAOLogin.login(username, password); if (role != "")

{

MyDB.SetMyDB(username, password); if (role == "Đăng nhập thất bại")

{

lblErr.Text = "Sai tÃi khoản hoặc mật khẩu !";

}

else

{

this.Hide();

if (role == "ChuDoanh ghiep")

{

new MainFrm().ShowDialog();

}

else if (role == "KhachHang")

{

var customer = new DAOKhachHang().getKhachHangByUsername(username);

new CustomerMainFrm(customer).ShowDialog();

}

else if (role == " hanVien")

{

var employee = new

DAO hanVien().get hanVienByUsername(username);

new EmployeeMainFrm(employee).ShowDialog();

}

txtflassword.Text = ""; txtUsername.Text = ""; lblErr.Text = ""; this.Show();

}

}

else

{

// Đăng nhập thất bại, hiển thị thông báo lỗi từ database lblErr.Text = "Có lỗi hệ thống";

}

}

### CHƯƠNG 5: TRANSACTION

- Transaction với xác nhận hợp đồng

CREATE PROCEDURE sp\_XacNhanHopDong @MaHD nvarchar (50),

@TrangThaiCV nvarchar (50) = N'Chưa hoàn thành', @SoKM int = 0,

@SoXang int = 0, @TrangThai nvarchar(50),

@NgayThucHienGiaoXe Datetime, @MaNVGiaoXe int,

@NgayThucHienNhanXe Datetime, @MaNVNhanXe int

##### AS BEGIN

SET NOCOUNT ON;

-- Gán Ngày lập bằng GETDATE() tại đây DECLARE @MaXe int

select @MaXe = MaXe FROM HopDong WHERE MaHD = @MaHD; DECLARE @IsConflict BIT;

-- Kiểm tra xung đột thời gian với hàm fn\_CheckHopDong

SET @IsConflict = dbo.fn\_CheckHopDong(@MaXe, @NgayThucHienGiaoXe, @NgayThucHienNhanXe);

IF @IsConflict = 1 BEGIN

-- Nếu có xung đột, ném ra lỗi

RAISERROR(N'Hợp đồng này xung đột với một hợp đồng khác.', 16, 1); RETURN; -- Thoát khỏi thủ tục

END

BEGIN TRY

BEGIN TRANSACTION;

INSERT INTO dbo.CongViec (TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKM, TrangThai, SoXang)

VALUES (@TrangThaiCV, @NgayThucHienGiaoXe, @MaNVGiaoXe, @MaHD, N'Giao xe', @SoKM, @TrangThai, @SoXang);

INSERT INTO dbo.CongViec (TrangThaiCV, NgayThucHien, MaNV, MaHD, LoaiCV, SoKM, TrangThai, SoXang)

VALUES (@TrangThaiCV, @NgayThucHienNhanXe, @MaNVNhanXe, @MaHD, N'Nhận xe', @SoKM, @TrangThai, @SoXang);

UPDATE HopDong Set TrangThaiHD = N'Đang xử lý' Where HopDong.MaHD =

@MaHD

COMMIT TRANSACTION; END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

-- Lấy thông báo lỗi và ném lại DECLARE @ErrorMessage NVARCHAR(4000); SET @ErrorMessage = ERROR\_MESSAGE(); RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END; GO

- Transaction xóa khách hàng

CREATE PROCEDURE sp\_DeleteKhachHang @MaKH INT

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

DECLARE @MaTaiKhoan INT;

-- Lấy MaTaiKhoan của khách hàng SELECT @MaTaiKhoan = MaTaiKhoan FROM KhachHang

WHERE MaKH = @MaKH;

-- Kiểm tra xem khách hàng có tồn tại không IF @MaTaiKhoan IS NOT NULL

BEGIN

-- Xóa tài khoản tương ứng DELETE FROM TaiKhoan

WHERE MaTaiKhoan = @MaTaiKhoan;

-- Xóa khách hàng DELETE FROM KhachHang WHERE MaKH = @MaKH;

-- Cam kết giao dịch COMMIT TRANSACTION;

END ELSE BEGIN

RAISERROR(N'Khách hàng không tồn tại', 16, 1); ROLLBACK TRANSACTION;

END END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

RAISERROR(N'Xóa khách hàng thất bại', 16, 1); END CATCH

END; GO

- Transaction cập nhật khách hàng

CREATE PROCEDURE sp\_UpdateKhachHang

@MaKH INT, -- Khóa chính để xác định khách hàng @HoTen NVARCHAR(100) = NULL,

@DiaChi NVARCHAR(255) = NULL, @GioiTinh NVARCHAR(10) = NULL,

@NgaySinh DATE = NULL, @CMND NVARCHAR(15) = NULL,

@Username NVARCHAR(50) = NULL, @Password NVARCHAR(50) = NULL,

@LoaiNguoiDung NVARCHAR(50) = NULL, @TrangThaiKhachHang NVARCHAR(50) = NULL

AS BEGIN

SET NOCOUNT ON;

BEGIN TRY

-- Bắt đầu giao dịch BEGIN TRANSACTION;

-- Kiểm tra xem MaKH có tồn tại trong bảng KhachHang không IF NOT EXISTS (SELECT 1 FROM KhachHang WHERE MaKH = @MaKH) BEGIN

RAISERROR('Không tìm thấy khách hàng với MaKH: %d', 16, 1, @MaKH); ROLLBACK TRANSACTION;

##### RETURN;

END

-- Cập nhật thông tin khách hàng UPDATE KhachHang

##### SET

trị hiện tại

HoTen = COALESCE(@HoTen, HoTen), -- Nếu @HoTen là NULL, giữ nguyên giá

DiaChi = COALESCE(@DiaChi, DiaChi), GioiTinh = COALESCE(@GioiTinh, GioiTinh), NgaySinh = COALESCE(@NgaySinh, NgaySinh), CMND = COALESCE(@CMND, CMND),

TrangThai = COALESCE(@TrangThaiKhachHang, TrangThai)

WHERE MaKH = @MaKH;

NULL

-- Cập nhật thông tin tài khoản nếu có Username và Password

IF @Username IS NOT NULL OR @Password IS NOT NULL OR @LoaiNguoiDung IS NOT

##### BEGIN

UPDATE TaiKhoan SET

username = COALESCE(@Username, username), -- Nếu @Username là NULL, giữ nguyên giá trị hiện tại

password = COALESCE(@Password, password), LoaiNguoiDung = COALESCE(@LoaiNguoiDung, LoaiNguoiDung)

WHERE MaTaiKhoan = (SELECT MaTaiKhoan FROM KhachHang WHERE MaKH =

@MaKH);

##### END

-- Cam kết giao dịch nếu không có lỗi COMMIT TRANSACTION;

##### END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Lấy thông tin lỗi và ném lại lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); END CATCH

END; GO

- Transaction thêm khách hàng

CREATE PROCEDURE sp\_AddKhachHang @HoTen NVARCHAR(100), @DiaChi NVARCHAR(255),

@GioiTinh NVARCHAR(10), @NgaySinh DATE,

@CMND NVARCHAR(15),

@Username NVARCHAR(50), @Password NVARCHAR(50), @LoaiNguoiDung NVARCHAR(50),

@TrangThaiKhachHang NVARCHAR(50) = N'Hoạt động' -- Giá trị mặc định

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

DECLARE @MaTaiKhoan INT;

-- Thêm tài khoản mới

INSERT INTO TaiKhoan (username, password, LoaiNguoiDung) VALUES (@Username, @Password, @LoaiNguoiDung);

-- Lấy ID tài khoản vừa thêm

SET @MaTaiKhoan = SCOPE\_IDENTITY();

-- Thêm khách hàng mới

INSERT INTO KhachHang (HoTen, DiaChi, GioiTinh, NgaySinh, CMND, MaTaiKhoan, TrangThai)

VALUES (@HoTen, @DiaChi, @GioiTinh, @NgaySinh, @CMND, @MaTaiKhoan, @TrangThaiKhachHang);

-- Cam kết giao dịch COMMIT TRANSACTION;

END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Khai báo biến để lưu thông tin lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

-- Lấy thông tin lỗi SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

-- Ném lại lỗi với thông tin chi tiết RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState);

END CATCH

END; GO

- Transaction thêm hợp đồng

CREATE PROCEDURE sp\_AddHopDong @NoiDung NVARCHAR(255),

@TongTien DECIMAL(10, 2),

@PhiThem DECIMAL(10, 2),

@NgayGiaoXe DATETIME, @NgayNhanXe DATETIME, @MaKH INT,

@MaXe INT,

@TrangThaiHD NVARCHAR(50) = N'Chưa xác nhận' -- Giá trị mặc định

AS BEGIN

SET NOCOUNT ON;

-- Gán Ngày lập bằng GETDATE() tại đây DECLARE @NgayLap DATETIME = GETDATE();

DECLARE @IsConflict BIT;

-- Kiểm tra xung đột thời gian với hàm fn\_CheckHopDong

SET @IsConflict = dbo.fn\_CheckHopDong(@MaXe, @NgayGiaoXe, @NgayNhanXe);

IF @IsConflict = 1 BEGIN

-- Nếu có xung đột, ném ra lỗi

RAISERROR(N'Hợp đồng này xung đột với một hợp đồng khác.', 16, 1); RETURN; -- Thoát khỏi thủ tục

END

BEGIN TRY

BEGIN TRANSACTION;

INSERT INTO HopDong (NoiDung, TongTien, PhiThem, NgayLap, NgayGiaoXe, NgayNhanXe, MaKH, MaXe, TrangThaiHD)

VALUES (@NoiDung, @TongTien, @PhiThem, @NgayLap, @NgayGiaoXe, @NgayNhanXe, @MaKH, @MaXe, @TrangThaiHD);

COMMIT TRANSACTION; END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

-- Lấy thông báo lỗi và ném lại DECLARE @ErrorMessage NVARCHAR(4000); SET @ErrorMessage = ERROR\_MESSAGE(); RAISERROR(@ErrorMessage, 16, 1);

END CATCH

END; GO

- Transaction về cập nhật nhân viên

CREATE PROCEDURE sp\_UpdateNhanVien

@MaNV INT, -- Khóa chính để xác định nhân viên @HoTen NVARCHAR(100) = NULL,

@DiaChi NVARCHAR(255) = NULL,

@GioiTinh NVARCHAR(10) = NULL,

@NgaySinh DATE = NULL, @CMND NVARCHAR(15) = NULL,

@MaTaiKhoan INT = NULL, @TrangThai NVARCHAR(50) = NULL

AS BEGIN

SET NOCOUNT ON;

BEGIN TRY

-- Bắt đầu giao dịch BEGIN TRANSACTION;

-- Kiểm tra xem MaNV có tồn tại trong bảng NhanVien không IF NOT EXISTS (SELECT 1 FROM NhanVien WHERE MaNV = @MaNV) BEGIN

RAISERROR('Không tìm thấy nhân viên với MaNV: %d', 16, 1, @MaNV); ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật thông tin nhân viên UPDATE NhanVien

SET

HoTen = COALESCE(@HoTen, HoTen), -- Nếu @HoTen là NULL, giữ nguyên giá

trị hiện tại

DiaChi = COALESCE(@DiaChi, DiaChi), GioiTinh = COALESCE(@GioiTinh, GioiTinh), NgaySinh = COALESCE(@NgaySinh, NgaySinh), CMND = COALESCE(@CMND, CMND),

MaTaiKhoan = COALESCE(@MaTaiKhoan, MaTaiKhoan), TrangThai = COALESCE(@TrangThai, TrangThai)

WHERE MaNV = @MaNV;

-- Cam kết giao dịch nếu không có lỗi COMMIT TRANSACTION;

END TRY BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Lấy thông tin lỗi và ném lại lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); END CATCH

END;

- Transaction xác nhận công việc hoàn thành

CREATE PROCEDURE dbo.sp\_XacNhanCongViecHoanThanh ( @MaCongViec INT, -- Job ID

@SoKM INT, -- Distance traveled (in kilometers) @SoXang INT, -- Fuel used (in liters)

@TrangThai nvarchar(50),

@PhiThem DECIMAL(10, 2) -- Extra fee to be added to the contract

##### ) AS

BEGIN

DECLARE @MaHD INT; -- Variable to store Contract ID

DECLARE @DeliveryStatus NVARCHAR(50); -- Variable to check delivery status

##### BEGIN TRY

BEGIN TRANSACTION; -- Start the transaction

-- Get the contract ID (MaHD) for the given job SELECT

@MaHD = MaHD FROM CongViec

WHERE MaCongViec = @MaCongViec;

-- Check if the job type is 'Nhận Xe'

IF EXISTS (SELECT 1 FROM CongViec WHERE MaCongViec = @MaCongViec AND LoaiCV

= N'Nhận Xe')

##### BEGIN

-- Check if the associated delivery job is completed SELECT

@DeliveryStatus = TrangThaiCV FROM CongViec

WHERE MaHD = @MaHD AND LoaiCV = N'Giao Xe';

IF @DeliveryStatus IS NULL OR @DeliveryStatus <> N'Đã hoàn thành' BEGIN

-- Raise an error if the delivery job is not completed RAISERROR('Công việc giao xe cho khách hàng vẫn chưa hoàn thành,

không thể xác nhận công việc này !.', 16, 1);

ROLLBACK TRANSACTION; -- Rollback the transaction RETURN; -- Stop execution of the procedure

##### END

-- Update the contract with the extra fee UPDATE HopDong

##### SET

TongTien = TongTien + @PhiThem, -- Assuming there's a TotalFees column to store fees

PhiThem = @PhiThem -- Assuming there's a column for the extra fee WHERE

MaHD = @MaHD;

##### END

-- Update the job to mark it as completed UPDATE CongViec

##### SET

SoKM = @SoKM, -- Update distance traveled

SoXang = @SoXang, -- Update fuel used TrangThai = @TrangThai,

TrangThaiCV = N'Đã hoàn thành' -- Update job status to "Completed" WHERE

MaCongViec = @MaCongViec; -- Where the job ID matches

COMMIT TRANSACTION; -- Commit the transaction if all is successful END TRY

##### BEGIN CATCH

ROLLBACK TRANSACTION; -- Rollback the transaction on error DECLARE @ErrorMessage NVARCHAR(4000);

DECLARE @ErrorSeverity INT; DECLARE @ErrorState INT;

##### SELECT

@ErrorMessage = ERROR\_MESSAGE(),

@ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); -- Raise the error END CATCH

END;

- Transaction thêm xe

CREATE PROCEDURE [dbo].[sp\_AddXe] @TenXe NVARCHAR(100),

@TrangThai NVARCHAR(50) = N'Tồn tại', @HinhAnh varbinary(MAX),

@PhiXang DECIMAL(10, 2),

@GiaNgay DECIMAL(10, 2), -- Giá thuê theo ngày @GiaTuan DECIMAL(10, 2), -- Giá thuê theo tuần @GiaThang DECIMAL(10, 2), -- Giá thuê theo tháng @HangXe NVARCHAR(50),

@MaLoaiXe INT, -- Loại xe là khóa ngoại @BienSoXe nvarchar(50)

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

-- Kiểm tra nếu hợp đồng đã có thanh toán

IF EXISTS (SELECT 1 FROM Xe WHERE BienSoXe = @BienSoXe) BEGIN

RAISERROR(N'Biển số xe đã tồn tại', 16, 1); RETURN;

END

-- Thêm xe mới

INSERT INTO Xe (TenXe, TrangThai, HinhAnh, PhiXang, HangXe, MaLoai, BienSoXe)

VALUES (@TenXe, @TrangThai, @HinhAnh, @PhiXang, @HangXe, @MaLoaiXe,@BienSoXe);

-- Lấy MaXe vừa được thêm DECLARE @MaXe INT;

SET @MaXe = SCOPE\_IDENTITY();

-- Thêm giá thuê cho xe theo 3 mốc thời gian INSERT INTO GiaThue (MaXe, ThoiGian, GiaThue) VALUES

(@MaXe, N'Ngày', @GiaNgay), (@MaXe, N'Tuần', @GiaTuan), (@MaXe, N'Tháng', @GiaThang);

-- Cam kết giao dịch COMMIT TRANSACTION;

PRINT N'Xe và giá thuê đã được thêm thành công.'; END TRY

BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Khai báo biến để lưu thông tin lỗi

DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

-- Lấy thông tin lỗi SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

-- Ném lại lỗi với thông tin chi tiết RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState);

END CATCH

END;

- Transaction thêm nhân viên

CREATE PROCEDURE sp\_AddNhanVien @HoTen NVARCHAR(100), @DiaChi NVARCHAR(255),

@GioiTinh NVARCHAR(10), @NgaySinh DATE,

@CMND NVARCHAR(15),

@Username NVARCHAR(50), @Password NVARCHAR(50), @LoaiNguoiDung NVARCHAR(50),

@TrangThaiNhanVien NVARCHAR(50) = N'Hoạt động' -- Giá trị mặc định

AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

BEGIN TRANSACTION;

DECLARE @MaTaiKhoan INT;

-- Thêm tài khoản mới

INSERT INTO TaiKhoan (username, password, LoaiNguoiDung) VALUES (@Username, @Password, @LoaiNguoiDung);

-- Lấy ID tài khoản vừa thêm

SET @MaTaiKhoan = SCOPE\_IDENTITY();

-- Thêm nhân viên mới

INSERT INTO NhanVien (HoTen, DiaChi, GioiTinh, NgaySinh, CMND, MaTaiKhoan, TrangThai)

VALUES (@HoTen, @DiaChi, @GioiTinh, @NgaySinh, @CMND, @MaTaiKhoan, @TrangThaiNhanVien);

-- Cam kết giao dịch COMMIT TRANSACTION;

END TRY BEGIN CATCH

ROLLBACK TRANSACTION;

RAISERROR(N'Thêm nhân viên thất bại', 16, 1); END CATCH

END; GO

### Transaction thêm xe

CREATE PROCEDURE [dbo].[sp\_AddXe] @TenXe NVARCHAR(100),

@TrangThai NVARCHAR(50) = N'Tồn tại', @HinhAnh varbinary(MAX),

@PhiXang DECIMAL(10, 2),

@GiaNgay DECIMAL(10, 2), -- Giá thuê theo ngày @GiaTuan DECIMAL(10, 2), -- Giá thuê theo tuần @GiaThang DECIMAL(10, 2), -- Giá thuê theo tháng @HangXe NVARCHAR(50),

@MaLoaiXe INT, -- Loại xe là khóa ngoại @BienSoXe nvarchar(50)

##### AS BEGIN

SET NOCOUNT ON;

-- Bắt đầu giao dịch BEGIN TRY

##### BEGIN TRANSACTION;

-- Kiểm tra nếu hợp đồng đã có thanh toán

IF EXISTS (SELECT 1 FROM Xe WHERE BienSoXe = @BienSoXe) BEGIN

RAISERROR(N'Biển số xe đã tồn tại', 16, 1); RETURN;

##### END

-- Thêm xe mới

INSERT INTO Xe (TenXe, TrangThai, HinhAnh, PhiXang, HangXe, MaLoai, BienSoXe)

VALUES (@TenXe, @TrangThai, @HinhAnh, @PhiXang, @HangXe, @MaLoaiXe,@BienSoXe);

-- Lấy MaXe vừa được thêm DECLARE @MaXe INT;

SET @MaXe = SCOPE\_IDENTITY();

-- Thêm giá thuê cho xe theo 3 mốc thời gian INSERT INTO GiaThue (MaXe, ThoiGian, GiaThue) VALUES

(@MaXe, N'Ngày', @GiaNgay), (@MaXe, N'Tuần', @GiaTuan), (@MaXe, N'Tháng', @GiaThang);

-- Cam kết giao dịch COMMIT TRANSACTION;

PRINT N'Xe và giá thuê đã được thêm thành công.'; END TRY

##### BEGIN CATCH

-- Hoàn tác giao dịch nếu có lỗi ROLLBACK TRANSACTION;

-- Khai báo biến để lưu thông tin lỗi DECLARE @ErrorMessage NVARCHAR(4000); DECLARE @ErrorSeverity INT;

DECLARE @ErrorState INT;

-- Lấy thông tin lỗi SELECT

@ErrorMessage = ERROR\_MESSAGE(), @ErrorSeverity = ERROR\_SEVERITY(), @ErrorState = ERROR\_STATE();

-- Ném lại lỗi với thông tin chi tiết

RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState); END CATCH

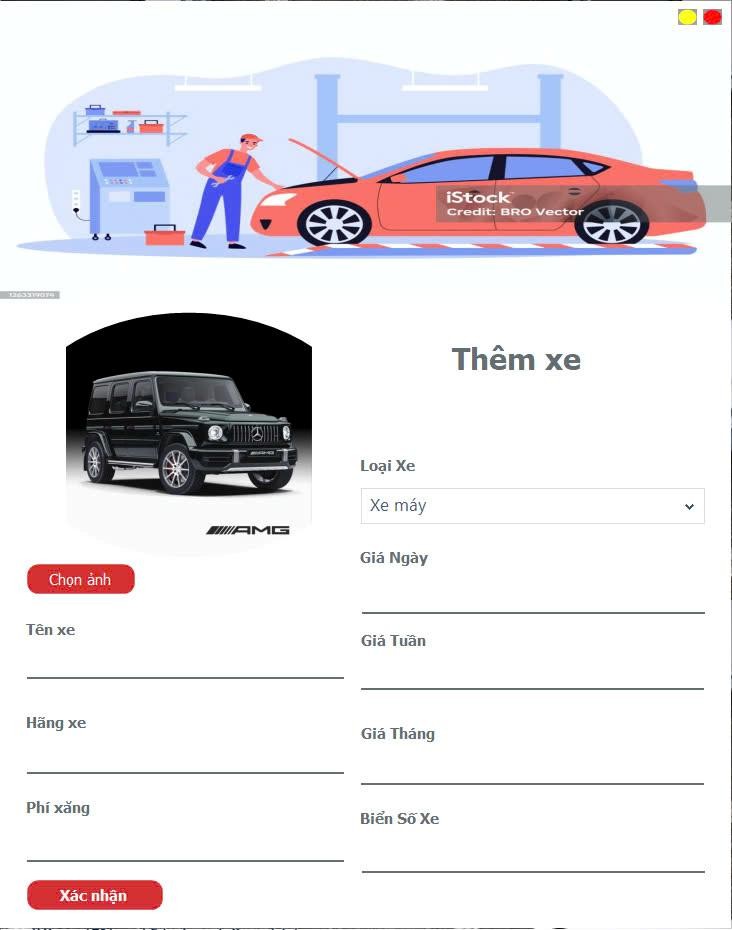
END;

### CHƯƠNG 6: THIẾT KẾ GIAO DIỆN

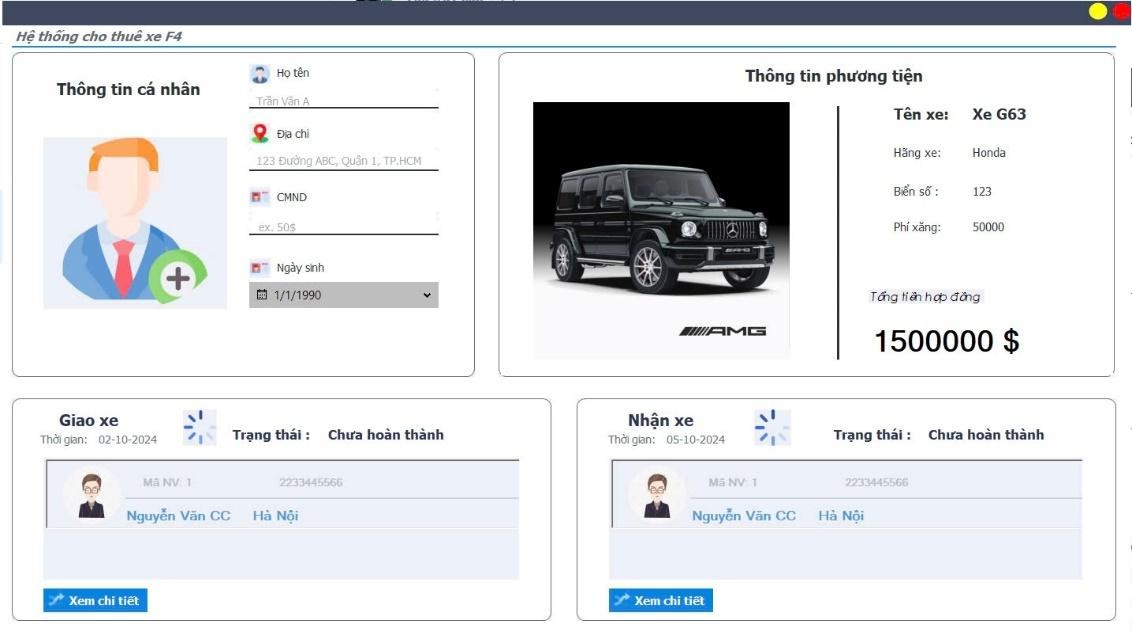
### Môi trường cài đặt và công nghệ sử dụng

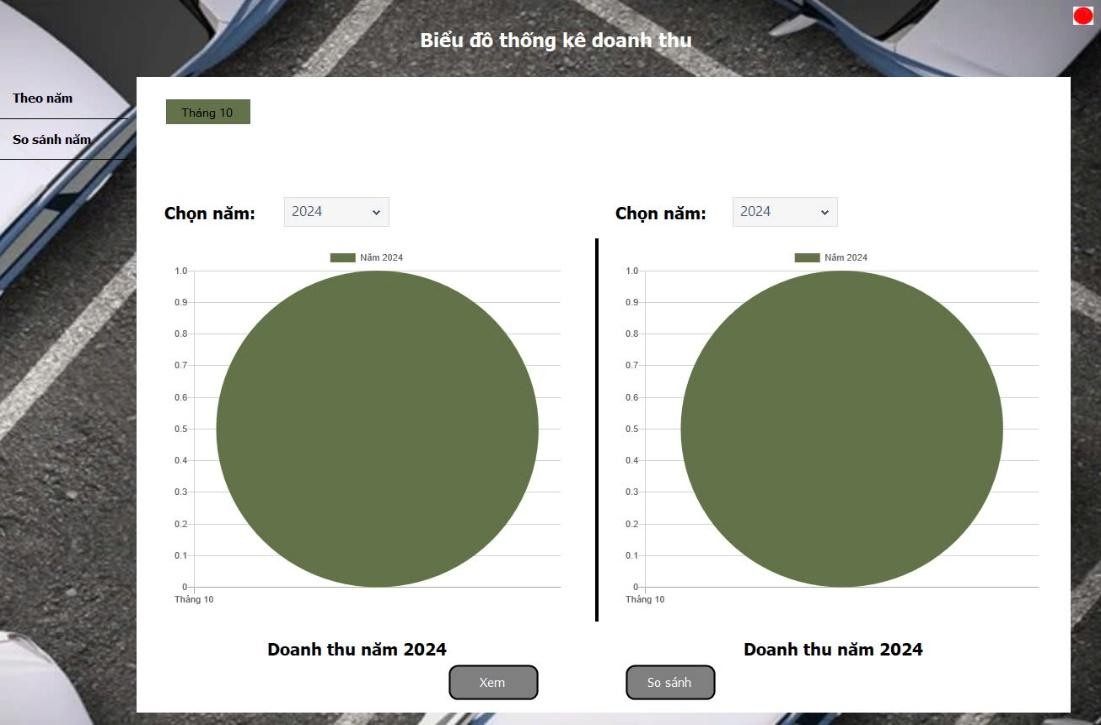
* Hệ cơ sở dữ liệu Microsoft SQL Server Management Studio 20.
* Chương trình Winform được viết bằng Microsoft Visual Studio 2022 - trên nền tảng ASP.NET Web Application (.NET Framework) C#.

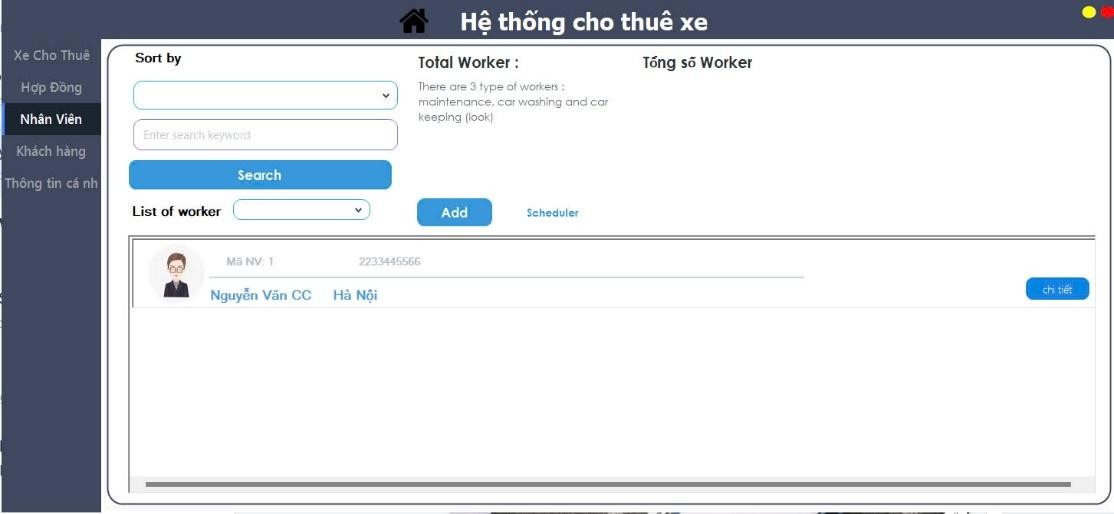
### Phân chức năng chủ doanh nghiệp:

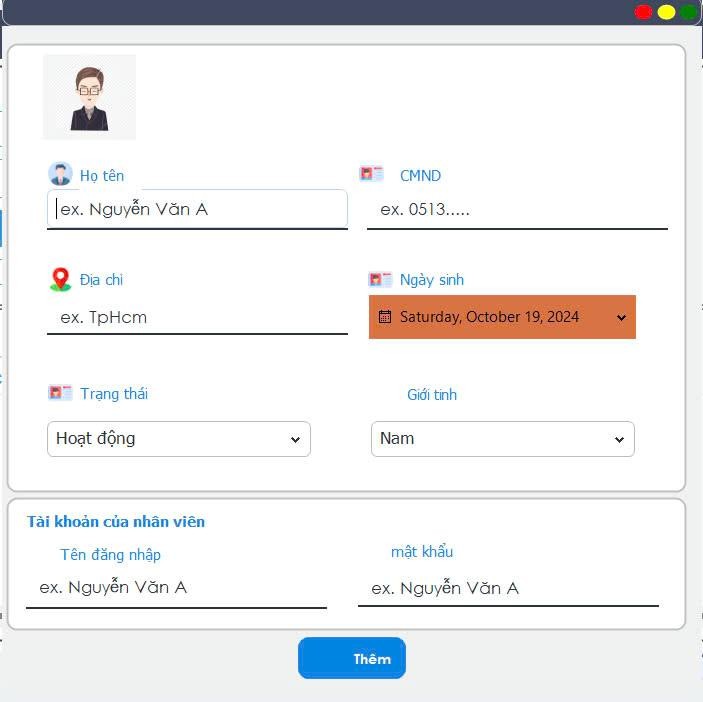
* 1. **Form của chủ doanh nghiệp:**
  2. **Quản lý xe:** thêm xe, sửa thông tin xe:
  3. **Quản lý hợp đồng:**

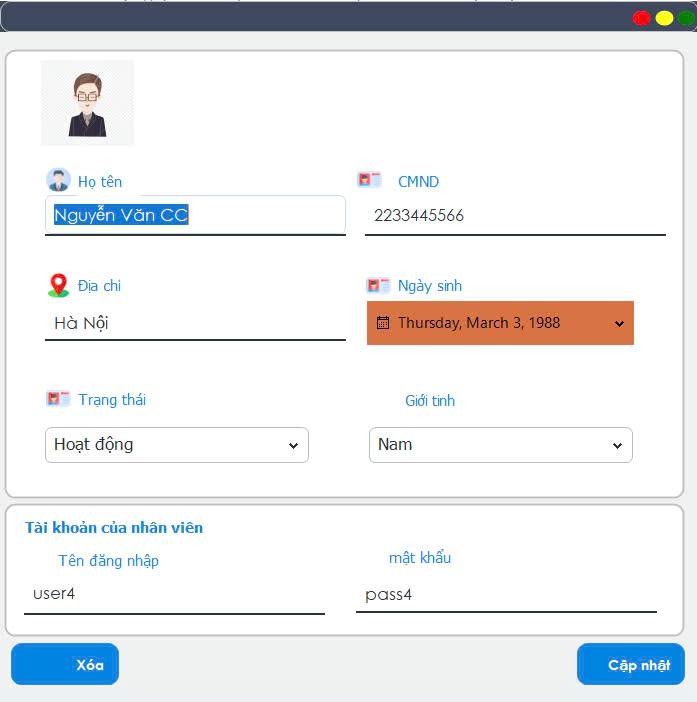


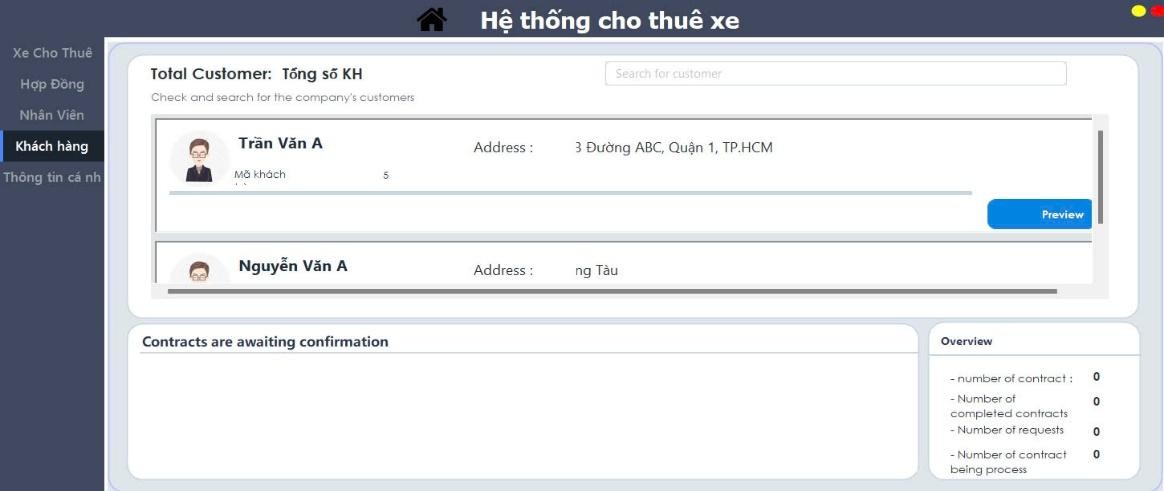
* 1. **Xác nhận hợp đồng từ khách hàng và phân công cho nhân viên:**
  2. **Xem chi tiết hợp đồng:**
  3. **Thống kê doanh thu theo tháng và theo từng năm:**

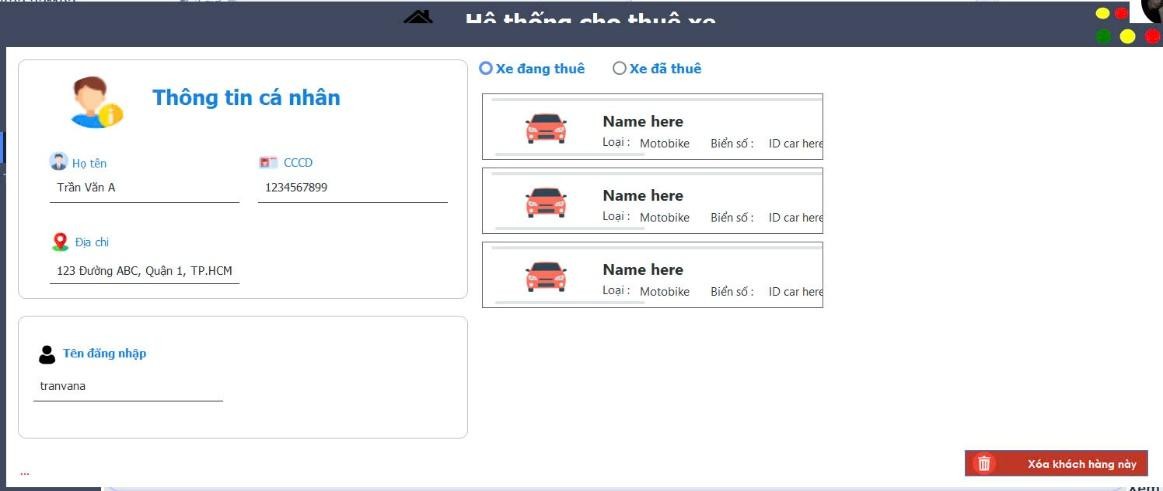


* 1. **Quản lý nhân viên:**
  2. **Thêm nhân viên:**



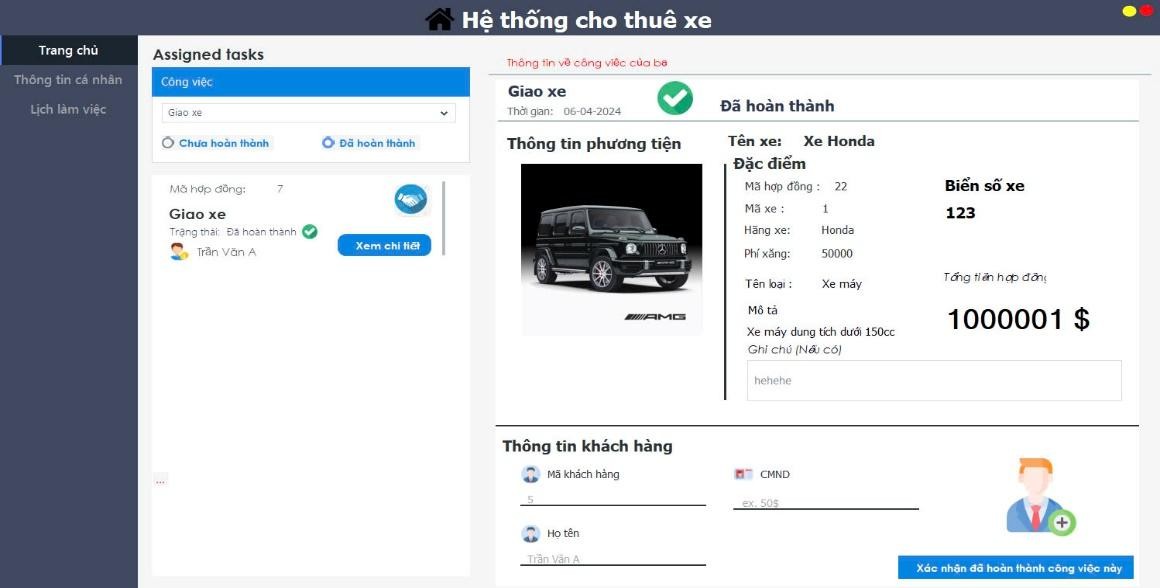
* 1. **Chỉnh sửa nhân viên:**
  2. **Xem khách hàng:**

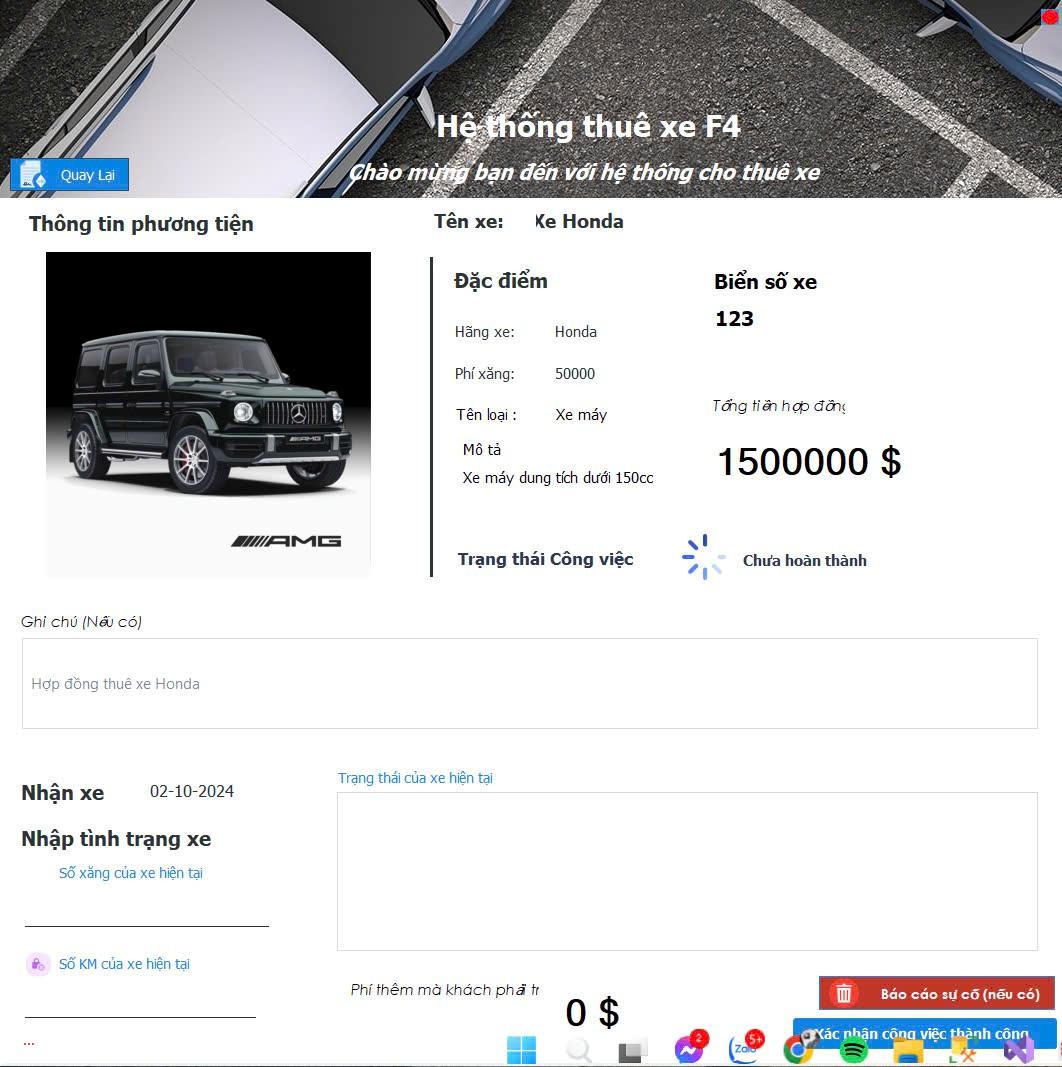


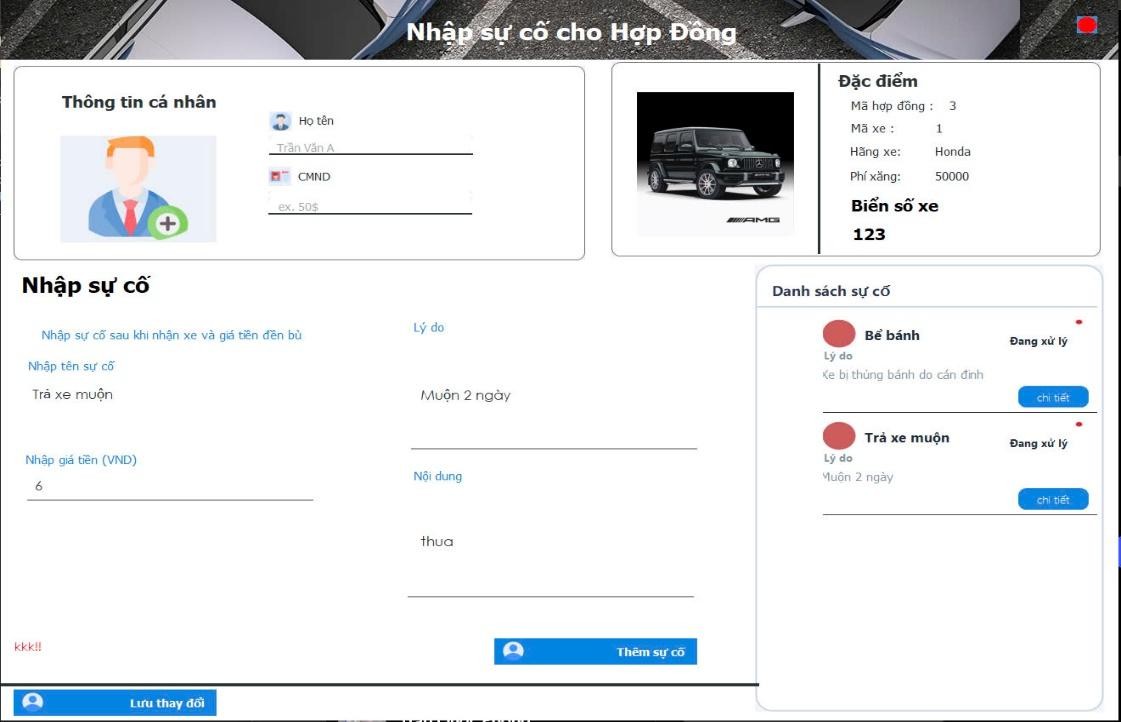
* 1. **Xem chi tiết khách hàng:** Có thể xóa khỏi hệ thống
  2. **Xem thông tin của bản thân:**

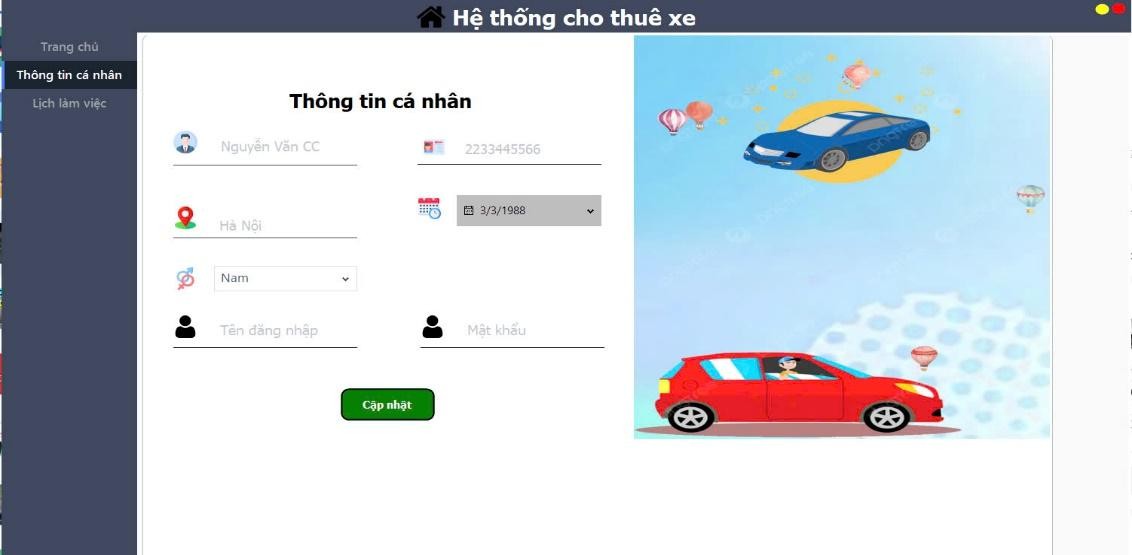
### Các chức năng của nhân viên:

* 1. **Xem công việc được giao:**

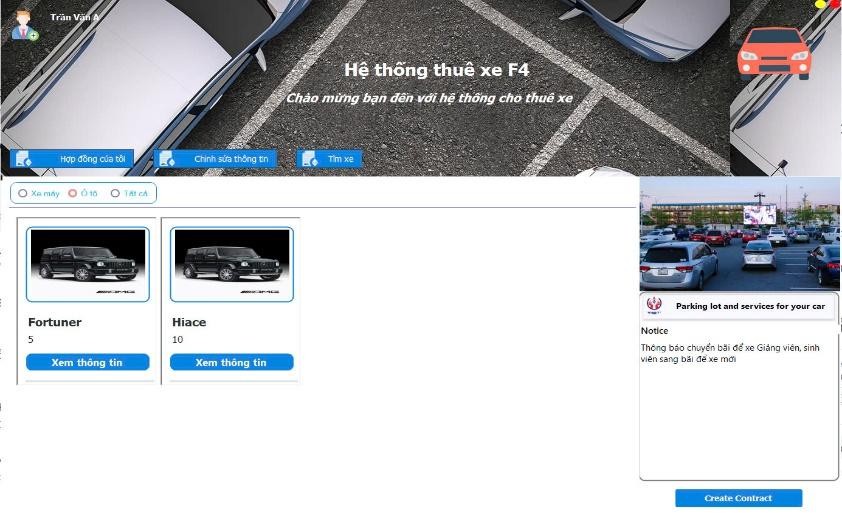


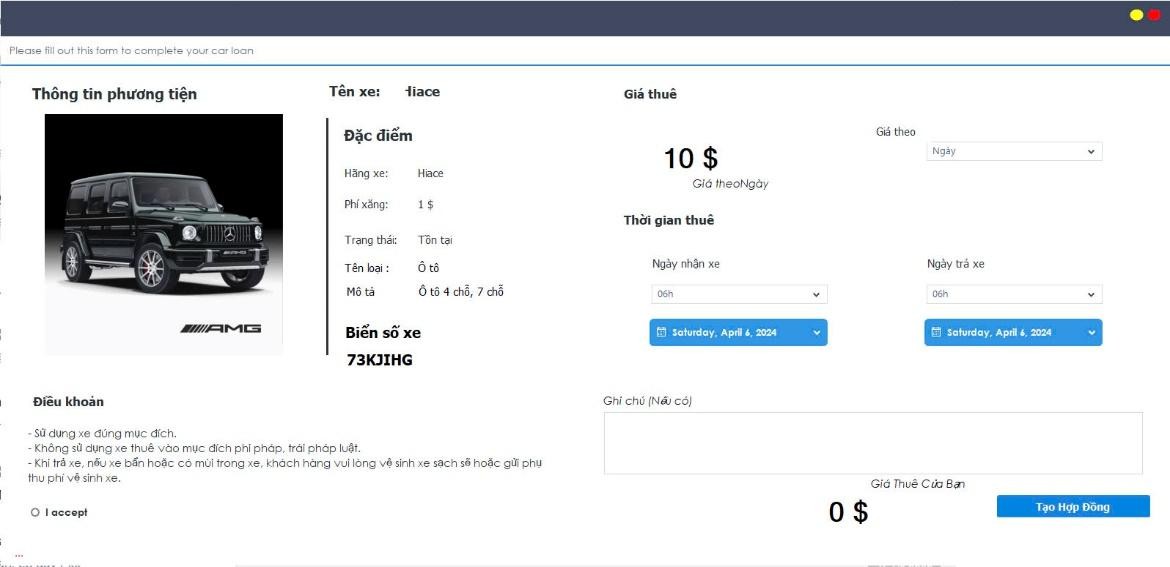
* 1. **Xác nhận công việc:**
  2. **Các sự cố có thể xảy ra:** Do nhân viên nhập



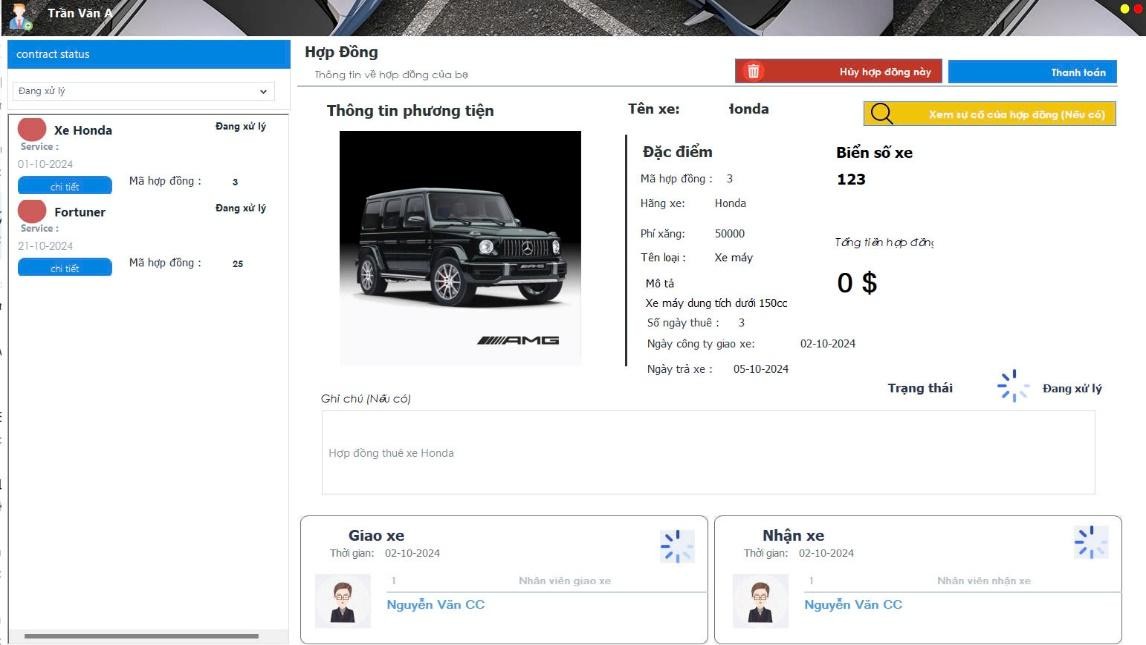
* 1. **Chỉnh sửa thông tin cá nhân:**

### Các chức năng của khách hàng:

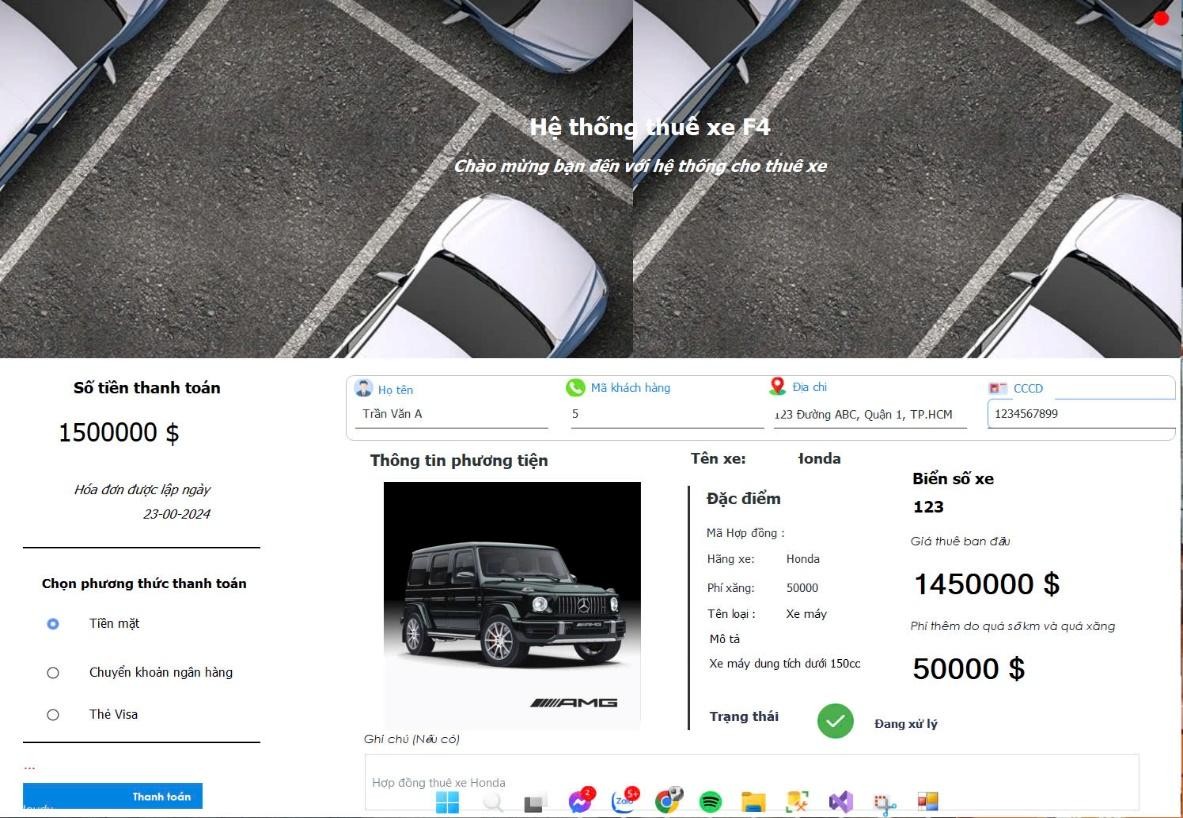
* 1. **Tìm xe muốn thuê:**
  2. **Tạo hợp đồng:**

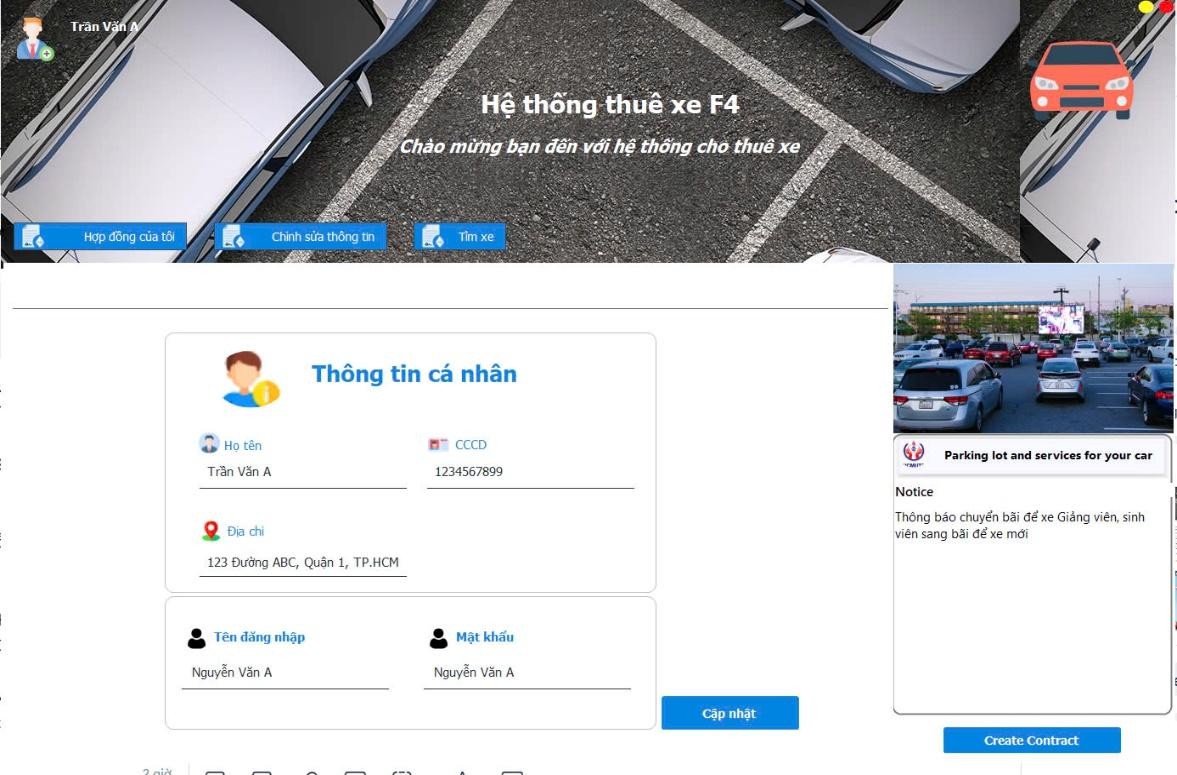


* 1. **Xem danh sách hợp đồng theo trạng thái:** Có thể hủy khi hợp đồng chưa được xác nhận từ phía chủ doanh nghiệp:



* 1. **Thanh toán hợp đồng:**



* 1. **Chỉnh sửa thông tin cá nhân:**