**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**NGUYỄN ANH TUẤN**

**PHÁT TRIỂN HỆ THỐNG BÃI ĐỖ Ô TÔ TÍCH HỢP CÁC KỸ THUẬT TRÍ TUỆ NHÂN TẠO**

**ĐỒ ÁN NGÀNH**

**NGÀNH KHOA HỌC MÁY TÍNH**

**TP. HỒ CHÍ MINH, 2024BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**NGUYỄN ANH TUẤN**

**PHÁT TRIỂN HỆ THỐNG BÃI ĐỖ Ô TÔ TÍCH HỢP CÁC KỸ THUẬT TRÍ TUỆ NHÂN TẠO**

**Mã số sinh viên: 2051012127**

**ĐỒ ÁN NGÀNH**

**NGÀNH KHOA HỌC MÁY TÍNH**

**Giảng viên hướng dẫn: TS. Trương Hoàng Vinh**

**TP. HỒ CHÍ MINH, 2024**

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn sâu sắc đến thầy Trương Hoàng Vinh - Giảng viên khoa Công Nghệ Thông Tin Trường Đại học Mở thành phố Hồ Chí Minh, đã nhiệt tình hỗ trợ, hướng dẫn và động viên em vượt qua khó khăn trong suốt quá trình thực hiện đồ án, để em có thể hoàn thành tốt đồ án của mình.

Em cũng xin chân thành cảm ơn các thầy cô giáo đang giảng dạy tại khoa Công Nghệ Thông Tin Trường Đại học Mở thành phố Hồ Chí Minh đã ủng hộ, cung cấp tài liệu và đưa ra những ý kiến đóng góp quý báu nhằm giúp em hoàn thiện đề tài hơn.

Kính chúc các thầy cô giáo khoa Công Nghệ Thông Tin Trường Đại học Mở thành phố Hồ Chí Minh thật nhiều sức khỏe, hạnh phúc và luôn vun đắp cho sự nghiệp trồng người.

Cuối cùng, Em xin dành lời cảm ơn chân thành tới bạn bè đã động viên, khuyến khích, giúp đỡ và tạo điều kiện để em hoàn thành tốt đề tài của mình.

***Xin chân thành cảm ơn!***

*Hồ Chí Minh, tháng 10 năm 2024*

**Nguyễn Anh Tuấn**

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

TÓM TẮT ĐỐ ÁN NGÀNH

**Phát triển hệ thống bãi đỗ ô tô tích hợp các kỹ thuật trí tuệ nhân tạo,** nhằm giải quyết những thách thức trong việc quản lý và vận hành bãi đỗ xe tại các khu vực đô thị đông đúc cần một giải pháp đỗ xe an toàn và tiện lợi. Hệ thống này sử dụng công nghệ nhận diện khuôn mặt và quản lý thông qua các ứng dụng web, cho phép người dùng đăng ký, đặt chỗ đỗ xe, và thanh toán trực tuyến một cách nhanh chóng, tiện lợi và an toàn.

Người dùng có thể dễ dàng đăng ký tài khoản với dữ liệu khuôn mặt của mình, đăng ký phương tiện, và thực hiện các giao dịch liên quan đến đặt chỗ hoặc đăng ký chỗ. Thanh toán trực tiếp trên hệ thống thông qua ví điện tử Momo. Hệ thống tích hợp tính năng nhận diện khuôn mặt và bóc tách hình ảnh xe ô tô giúp xác thực chủ phương tiện khi vào và ra khỏi bãi đỗ, nâng cao tính an ninh và giảm thiểu các rủi ro phát sinh.

Bên cạnh đó, hệ thống còn lưu trữ hình ảnh xe cộ tại thời điểm ra và vào bãi, nhằm đối chiếu, giải quyết các khiếu nại liên quan đến hư hại phương tiện. Quá trình phát triển được thực hiện bằng các công nghệ như Django, ReactJS, Face-api.js, Plate Recognizer cùng với tích hợp API thanh toán của MoMo.

ABSTRACT

**MỤC LỤC**

[LỜI CẢM ƠN 1](#_Toc179801702)

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 2](#_Toc179801703)

[TÓM TẮT ĐỐ ÁN NGÀNH 3](#_Toc179801704)

[ABSTRACT 3](#_Toc179801705)

[DANH MỤC TỪ VIẾT TẮT 7](#_Toc179801706)

[DANH MỤC HÌNH VẼ 8](#_Toc179801707)

[DANH MỤC BẢNG 10](#_Toc179801708)

[Chương 1. TỔNG QUAN ĐỀ TÀI 11](#_Toc179801709)

[1.1. Tổng quan 11](#_Toc179801710)

[1.2. Mục tiêu và phạm vi 11](#_Toc179801711)

[1.3. Phương pháp thực hiện 12](#_Toc179801712)

[1.3.1. Thiết kế hệ thống và lựa chọn công nghệ: 12](#_Toc179801713)

[1.3.2. Tích hợp các công nghệ nhận diện và thanh toán: 12](#_Toc179801714)

[1.3.3. Triển khai và kiểm thử: 12](#_Toc179801715)

[1.3.4. Đánh giá và hoàn thiện: 13](#_Toc179801716)

[1.4. Bố cục báo cáo 13](#_Toc179801717)

[1.4.1. Chương 1: Tổng quan đề tài 13](#_Toc179801718)

[1.4.2. Chương 2: Cơ sở lý thuyết 13](#_Toc179801719)

[1.4.3. Chương 3: Phát triển hệ thống bãi đỗ ô tô tích hợp các kỹ thuật trí tuệ nhân tạo 13](#_Toc179801720)

[1.4.4. Chương 4: Kết luận và hướng phát triển 14](#_Toc179801721)

[Chương 2. CƠ SỞ LÝ THUYẾT 15](#_Toc179801722)

[2.1. Django 15](#_Toc179801723)

[2.1.1. Tổng quan về Django 15](#_Toc179801724)

[2.1.2. Kiến trúc Django 15](#_Toc179801725)

[2.1.3. Sử dụng Django 16](#_Toc179801726)

[2.1.4. Django Rest Framework 18](#_Toc179801727)

[2.2. ReactJs 18](#_Toc179801728)

[2.2.1. Tổng quan về ReactJs 18](#_Toc179801729)

[2.2.2. Kiến trúc ReactJs 19](#_Toc179801730)

[2.2.3. Sử dụng ReactJs 20](#_Toc179801731)

[2.3. MySQL 21](#_Toc179801732)

[2.3.1. Tổng quan về MySQL 21](#_Toc179801733)

[2.3.2. Kiến trúc của MySQL 22](#_Toc179801734)

[2.4. Các công cụ hỗ trợ khác 23](#_Toc179801735)

[2.4.1. Postman 23](#_Toc179801736)

[2.4.2. Pythonanywhere 23](#_Toc179801737)

[2.4.3. Momo API 23](#_Toc179801738)

[2.4.4. Face-api.js 24](#_Toc179801739)

[2.4.5. Plate Recognizer 25](#_Toc179801740)

[2.4.6. SMTP Gmail 26](#_Toc179801741)

[2.4.7. OAuth2 26](#_Toc179801742)

[Chương 3. PHÁT TRIỂN HỆ THỐNG BÃI ĐỖ Ô TÔ TÍCH HỢP CÁC KỸ THUẬT TRÍ TUỆ NHÂN TẠO 28](#_Toc179801743)

[3.1. Giới thiệu đề tài 28](#_Toc179801744)

[3.2. Phân tích hệ thống 28](#_Toc179801745)

[3.2.1. Chức năng chung của toàn bộ User 28](#_Toc179801746)

[3.2.2. Vai trò của Nhân viên 29](#_Toc179801747)

[3.2.3. Quản trị viên quản trị toàn diện hệ thống 30](#_Toc179801748)

[3.2.4. Khách hàng tương tác với hệ thống 31](#_Toc179801749)

[3.3. Thiết kế hệ thống 40](#_Toc179801750)

[3.3.1. Sequence diagram 40](#_Toc179801751)

[3.3.2. Activity diagram 45](#_Toc179801752)

[3.4. Thiết kế dữ liệu 46](#_Toc179801753)

[3.4.1. Class diagram 46](#_Toc179801754)

[3.4.2. Chi tiết thiết kế cơ sở dữ liệu 48](#_Toc179801755)

[3.5. Kiến trúc hệ thống 50](#_Toc179801756)

[3.6. Sản phẩm 50](#_Toc179801757)

[3.6.1. Khách hàng đăng ký thành viên, và đăng nhập 50](#_Toc179801758)

[3.6.2. Xem trạng thái chỗ đỗ xe 53](#_Toc179801759)

[3.6.3. Khách hàng đặt chỗ 54](#_Toc179801760)

[3.6.4. Khách hàng đăng ký chỗ, gia hạn chỗ 54](#_Toc179801761)

[3.6.5. Khách hàng quản lý phương tiện 56](#_Toc179801762)

[3.6.6. Khách hàng quản lí thông tin cá nhân 56](#_Toc179801763)

[3.6.7. Nhân viên hỗ trợ khách hàng vào và ra bãi xe 56](#_Toc179801764)

[3.6.8. Khách hàng đánh giá bãi xe 58](#_Toc179801765)

[3.6.9. Admin tạo báo cáo 58](#_Toc179801766)

[Chương 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 60](#_Toc179801767)

[4.1. Kết luận 60](#_Toc179801768)

[4.1.1. Ưu điểm 60](#_Toc179801769)

[4.1.2. Khuyết điểm 60](#_Toc179801770)

[4.2. Hướng phát triển 61](#_Toc179801771)

[TÀI LIỆU THAM KHẢO 62](#_Toc179801772)

[PHỤ LỤC 63](#_Toc179801773)

DANH MỤC TỪ VIẾT TẮT

DANH MỤC HÌNH VẼ

[*Hình 3.1. Chức năng chung của User* 29](#_Toc179802851)

[*Hình 3.2. Vai trò của nhân viên* 30](#_Toc179802852)

[*Hình 3.3. Quản trị viên quản trị toàn diện hệ thống.* 31](#_Toc179802853)

[*Hình 3.4. Khách hàng đăng ký tài khoản* 32](#_Toc179802854)

[*Hình 3.5. Khách hàng quản lí thông tin cá nhân, thông tin phương tiện* 33](#_Toc179802855)

[*Hình 3.6. Khách hàng đặt chỗ, đăng ký mua chỗ* 36](#_Toc179802856)

[*Hình 3.7. Khách hàng vào và ra bãi đỗ xe* 38](#_Toc179802857)

[*Hình 3.8. Khách hàng đánh giá và khiếu nại* 40](#_Toc179802858)

[*Hình 3.9. Sequence diagram Khách hàng đặt chỗ* 40](#_Toc179802859)

[*Hình 3.10. Sequence diagram Khách hàng đăng ký chỗ* 41](#_Toc179802860)

[*Hình 3.11. Sequence diagram Khách hàng gia hạn chỗ* 42](#_Toc179802861)

[*Hình 3.12. Sequence diagram Khách hàng vào bãi đỗ xe* 43](#_Toc179802862)

[*Hình 3.13. Sequence diagram Khách hàng ra khỏi bãi đỗ xe* 44](#_Toc179802863)

[*Hình 3.14. Activity diagram khách hàng đặt chỗ* 45](#_Toc179802864)

[*Hình 3.15. Activity diagram khách hàng đăng ký chỗ đỗ xe* 45](#_Toc179802865)

[*Hình 3.16. Activity diagram khách hàng gia hạn chỗ đăng ký* 46](#_Toc179802866)

[*Hình 3.17. Class digram Hệ thống quản lí bãi đỗ xe* 47](#_Toc179802867)

[*Hình 3.18. Lược đồ ERD* 49](#_Toc179802868)

[*Hình 3.19. Kiến trúc hệ thống* 50](#_Toc179802869)

[*Hình 3.20. Giao diện đăng ký* 51](#_Toc179802870)

[*Hình 3.21. Giao diện đăng nhập* 51](#_Toc179802871)

[*Hình 3.22. Giao diện khách hàng đăng nhập* 52](#_Toc179802872)

[*Hình 3.23. Giao diện nhân viên đăng nhập* 52](#_Toc179802873)

[*Hình 3.24. Giao diện Admin đăng nhập* 53](#_Toc179802874)

[*Hình 3.25. Giao diện xem trạng thái chỗ đỗ xe* 53](#_Toc179802875)

[*Hình 3.26. Giao diện đặt chỗ* 54](#_Toc179802876)

[*Hình 3.27, Giao diện đăng ký chỗ* 55](#_Toc179802877)

[*Hình 3.28. Giao diện gia hạn chỗ* 55](#_Toc179802878)

[*Hình 3.29. Giao diện khách hàng quản lý phương tiện* 56](#_Toc179802879)

[*Hình 3.30. Giao diện khách hàng quản lí thông tin cá nhân* 56](#_Toc179802880)

[*Hình 3.31. Giao diện vào bãi đỗ xe.* 57](#_Toc179802881)

[*Hình 3.32. Giao diện ra khỏi bãi đỗ xe* 58](#_Toc179802882)

[*Hình 3.33. Giao diện đánh giá bãi xe* 58](#_Toc179802883)

[*Hình 3.34. Báo cáo đánh giá của người dùng* 59](#_Toc179802884)

[*Hình 3.35. Báo cáo doanh thu theo tháng* 59](#_Toc179802885)

DANH MỤC BẢNG

# Chương 1. TỔNG QUAN ĐỀ TÀI

## Tổng quan

**Hệ thống quản lý bãi đỗ xe tích hợp các kỹ thuật trí tuệ nhân tạo** là một giải pháp nhằm cải thiện và tối ưu hóa quy trình quản lý bãi đỗ xe thông qua việc áp dụng các công nghệ hiện đại. Đề tài tập trung vào việc phát triển một hệ thống tích hợp sử dụng các công nghệ web hiện đại như Django và ReactJS, cùng với các dịch vụ nhận diện hình ảnh và thanh toán trực tuyến, để cung cấp một giải pháp quản lý bãi đỗ xe hiệu quả và tiện lợi với người dùng.

Thiết kế hệ thống để đáp ứng các nhu cầu như quản lý và đặt chỗ cho các vị trí đỗ xe, theo dõi lịch sử ra vào bãi đỗ xe, và xử lý thanh toán trực tuyến thông qua API của Momo. Đồng thời, tích hợp các công nghệ nhận diện khuôn mặt và bóc tách thông tin biển số xe để tự động hóa quy trình ra vào bãi đỗ xe, nâng cao tính chính xác và bảo mật.

Với việc tích hợp các kỹ thuật trí tuệ nhân tạo thông minh như API nhận diện biển số của Plate Recognizer và thư viện dùng để nhận diện khuôn mặt face-api.js cho phép hệ thống hoạt động hiệu quả trong việc xác thực và theo dõi các phương tiện ra vào, hệ thống giúp tiết kiệm thời gian và giảm thiểu sai sót, mang lại sự tiện lợi cho người dùng và quản lý bãi đỗ xe.

## Mục tiêu và phạm vi

**Mục tiêu** của đề tài "Hệ thống quản lý bãi đỗ xe thông minh" là phát triển một giải pháp hiệu quả để quản lý bãi đỗ xe với sự tích hợp của các công nghệ hiện đại. Hệ thống nhắm đến việc cải thiện quy trình quản lý đỗ xe, giảm thiểu thời gian tìm chỗ đỗ và nâng cao trải nghiệm người dùng thông qua các tính năng thông minh như đặt chỗ trực tuyến, nhận diện biển số xe và xác thực khuôn mặt.

**Phạm vi** của đề tài tập trung vào việc giải quyết các nhu cầu của các bãi đỗ xe vừa và nhỏ, đáp ứng các yêu cầu quản lý cơ bản cho các bãi đỗ xe với quy mô trung bình, nơi số lượng chỗ đỗ không quá lớn và cần một giải pháp dễ triển khai và bảo trì.

## Phương pháp thực hiện

### ****Thiết kế hệ thống và lựa chọn công nghệ****:

* **Django** được sử dụng làm nền tảng phát triển ứng dụng web phía server, cung cấp một khung làm việc mạnh mẽ cho việc xây dựng các API thông qua Django Rest Framework. Django giúp đảm bảo việc quản lý dữ liệu bãi đỗ xe, người dùng, và lịch sử đỗ xe được thực hiện một cách hiệu quả và an toàn.
* **ReactJs** được chọn để phát triển giao diện người dùng phía client, cho phép xây dựng các trang web động với khả năng tương tác cao. ReactJs hỗ trợ xây dựng các thành phần giao diện người dùng dễ dàng và hiệu quả, cung cấp trải nghiệm người dùng mượt mà và nhanh chóng.

### ****Tích hợp các công nghệ nhận diện và thanh toán****:

* **Face-api.js:** thư viện này cho phép nhận diện khuôn mặt từ video và hình ảnh. Được sử dụng để thực hiện phân tích và xác thực khuôn mặt, giúp tự động hóa quy trình kiểm tra ra vào bãi đỗ xe.
* **API bóc tách biển số của Plate Recognizer** được tích hợp để nhận diện biển số xe từ hình ảnh hoặc video, hỗ trợ nhận dạng thông tin các phương tiện trong khi ra vào bãi đỗ xe. API này có khả năng nhận diện chính xác và nhanh chóng, rất phù hợp với yêu cầu của hệ thống.
* **API thanh toán Momo** được tích hợp dùng để xử lý các giao dịch thanh toán trực tuyến, cho phép người dùng thanh toán các loại hóa đơn về đỗ xe ô tô một cách nhanh chóng và an toàn.

### ****Triển khai và kiểm thử****:

* **Triển khai hệ thống** được thực hiện trên nền tảng **PythonAnywhere** để quản lý và triển khai ứng dụng web. Đây là một dịch vụ hosting cho phép dễ dàng triển khai các ứng dụng Python và Django.
* **Kiểm thử** hệ thống để đảm bảo mọi tính năng hoạt động đúng như mong đợi, bao gồm kiểm tra các API, các công nghệ được tích hợp, giao diện người dùng, các chức năng quản lý và tạo báo cáo bãi đỗ xe.

### ****Đánh giá và hoàn thiện****:

* **Đánh giá** và phân tích hiệu suất, nhằm đảm bảo rằng hệ thống đáp ứng được các yêu cầu và mục tiêu đã đề ra.
* **Hoàn thiện** các tính năng và khắc phục các vấn đề phát sinh trong quá trình kiểm thử để nâng cao chất lượng và hiệu quả của hệ thống.

## Bố cục báo cáo

### Chương 1: Tổng quan đề tài

Chương 1 cung cấp cách nhìn tổng quan về đề tài "Phát triển hệ thống bãi đỗ ô tô tích hợp các kỹ thuật trí tuệ nhân tạo." Phần này trình bày mục tiêu và phạm vi của đề tài, giải thích lý do lựa chọn đề tài và ý nghĩa của hệ thống trong bối cảnh hiện tại. Đề cập đến các vấn đề hiện tại trong quản lý bãi đỗ xe và các giải pháp tiềm năng mà hệ thống này có thể mang lại. Chương 1 cũng giới thiệu các yêu cầu cơ bản và chức năng của hệ thống, từ đó tạo nền tảng cho các phân tích chi tiết trong các chương tiếp theo.

### Chương 2: Cơ sở lý thuyết

Chương 2 tập trung vào việc trình bày các cơ sở lý thuyết liên quan đến các công nghệ và công cụ được sử dụng trong hệ thống. Phần này bao gồm tổng quan về các framework và library như Django, ReactJS, cở sở dữ liệu MySQL và các công nghệ khác như SMTP Gmail, Postman, PythonAnywhere, API thanh toán Momo, và các công nghệ nhận diện như face-api.js dùng để nhận diện khuôn mặt và Plate Recognizer dùng để quét biển số từ hình ảnh. Chương này cung cấp nền tảng lý thuyết cần thiết để hiểu rõ các kỹ thuật và công nghệ được áp dụng trong việc phát triển và triển khai hệ thống.

### Chương 3: Phát triển hệ thống bãi đỗ ô tô tích hợp các kỹ thuật trí tuệ nhân tạo

Chương 3 đi sâu vào chi tiết về thiết kế và triển khai hệ thống quản lý bãi đỗ xe tích hợp các kỹ thuật trí tuệ nhân tạo. Phần này mô tả cấu trúc hệ thống, làm rõ các chức năng cũng như cách hoạt động và lưu trữ dữ liệu bằng các lược đồ như Use Case, Class diagram, Sequence diagram, Activity diagram và ERD, đồng thời cũng cho thấy cách các kỹ thuật trí tuệ nhân tạo được tích hợp để đáp ứng các yêu cầu của đề tài. Chương này sẽ trình bày cách hệ thống sử dụng Django và ReactJS để xây dựng API và giao diện, tích hợp các công nghệ nhận diện khuôn mặt và biển số xe để tự động hóa quy trình, và tích hợp API thanh toán để xử lý giao dịch.

### Chương 4: Kết luận và hướng phát triển

Chương 4 tổng kết các kết quả đạt được từ việc triển khai hệ thống, đánh giá hiệu quả và độ tin cậy của giải pháp. Phần này sẽ nêu ra các thành công và hạn chế của hệ thống trong việc đáp ứng các mục tiêu đề ra. Đồng thời cũng nêu ra các ưu khuyết điểm, cùng với những điều đã học được, những điều còn sai sót trong suốt quá trình thực hiện đồ án. Chương này cũng sẽ đề xuất các hướng phát triển khả thi để cải tiến và mở rộng hệ thống như phát triển thêm các tính năng mới, tích hợp các công nghệ hiện đại, nâng cấp hiệu năng và khả năng mở rộng để phục vụ các bãi đỗ xe lớn hơn, cung cấp cái nhìn tổng quan về các bước tiếp theo và triển vọng tương lai của hệ thống.

# CƠ SỞ LÝ THUYẾT

## Django

### Tổng quan về Django

**Django** là một framework web mã nguồn mở, được viết bằng Python, được phát triển nhằm giúp việc xây dựng và duy trì các ứng dụng web trở nên nhanh chóng và hiệu quả. Django có hai nguyên tắc "Don't Repeat Yourself" (DRY) và “Keep it short and simple” (KISS), vì thế nên sẽ có nhiều tính năng đi kèm sẵn có như hệ thống quản lý người dùng, bảo mật, và cơ sở dữ liệu ORM (Object-Relational Mapping), giúp giảm thiểu công sức và tiết kiệm thời gian khi phát triển.

Ưu điểm:

* **Phát triển nhanh**: Django cung cấp nhiều tính năng sẵn có như admin panel, ORM, và routing, giúp lập trình viên giảm bớt thời gian xây dựng các chức năng cơ bản.
* **Bảo mật cao**: Django tự động xử lý nhiều vấn đề bảo mật phổ biến như SQL injection, cross-site scripting (XSS), và cross-site request forgery (CSRF).
* **Cộng đồng lớn**: Django có một cộng đồng phát triển mạnh mẽ, dễ dàng tìm kiếm tài liệu, hỗ trợ và nhiều ứng dụng mở rộng.
* **Tính module hóa**: Hệ thống ứng dụng module cho phép phát triển các ứng dụng web phức tạp nhưng vẫn có thể quản lý dễ dàng.

Nhược điểm:

* **Hơi nặng**: Django có nhiều tính năng sẵn có, đôi khi có thể làm hệ thống trở nên cồng kềnh và phức tạp hơn so với các framework nhẹ khác.
* **Không linh hoạt như các microframework**: Django tập trung vào việc cung cấp một cấu trúc mạnh mẽ và đầy đủ, nên đôi khi hạn chế sự linh động trong những dự án nhỏ hoặc các dự án cần tùy chỉnh cao.

### Kiến trúc Django

Django sử dụng kiến trúc **MVT (Model - View - Template)**, Kiến trúc MVT giúp tách biệt và thể hiện rõ ràng các phần khác nhau nhưng cũng có quan hệ mật thiết của ứng dụng (dữ liệu, logic, và giao diện), giúp dễ bảo trì và mở rộng. Mô hình này cũng hỗ trợ tốt cho việc phát triển ứng dụng lớn với nhiều thành viên làm việc cùng lúc.

* **Model**: Định nghĩa cấu trúc dữ liệu và cách thức giao tiếp với cơ sở dữ liệu. Django cung cấp một lớp ORM giúp lập trình viên làm việc với cơ sở dữ liệu một cách đơn giản mà không cần viết SQL thủ công.
* **View**: Xử lý logic của ứng dụng, kết nối Model với Template. View sẽ nhận các yêu cầu từ người dùng, thao tác với Model và trả lại giao diện thông qua Template.
* **Template**: Chịu trách nhiệm về giao diện người dùng. Templates là các tệp Html có thể được kết hợp với dữ liệu từ Model để hiển thị trên trình duyệt.

### Sử dụng Django

Để sử dụng Django, các bước cơ bản như sau:

B1: **Cài đặt Django**: Bằng cách sử dụng công cụ pip, cài đặt Django thông qua lệnh:

pip install django

B2: **Tạo dự án mới**: Sau khi cài đặt Django, khởi tạo một dự án mới bằng lệnh:

django-admin startproject myproject

B3: **Tạo ứng dụng**: Django chia dự án thành các app nhỏ để quản lý dễ hơn. Tạo một ứng dụng bằng lệnh:

python manage.py startapp myapp

B4: **Định nghĩa Models**: Models định nghĩa cấu trúc dữ liệu và tạo các bảng trong cơ sở dữ liệu. Ví dụ:

**Models:**

from django.db import models

class Vehicle(models.Model):

license\_plate = models.CharField(max\_length=20)

owner\_name = models.CharField(max\_length=100)

entry\_time = models.DateTimeField(auto\_now\_add=True)

exit\_time = models.DateTimeField(null=True, blank=True)

B5: **Xây dựng Views và Templates**: Views chứa logic ứng dụng và sẽ render Templates với dữ liệu từ Models. Ví dụ:

**View:**

from django.shortcuts import render

from .models import Vehicle

def vehicle\_list(request):

vehicles = Vehicle.objects.all()

return render(request, 'vehicle\_list.html', {'vehicles': vehicles})

**Templates:**

<!DOCTYPE html>

<html>

<head>

<title>Vehicle List</title>

</head>

<body>

<h1>List of Vehicles</h1>

<ul>

{% for vehicle in vehicles %}

<li>{{ vehicle.owner\_name }} - {{ vehicle.license\_plate }}</li>

{% endfor %}

</ul>

</body>

</html>

**Kết quả:** Trình duyệt sẽ hiển thị danh sách các phương tiện, với tên chủ sở hữu và biển số xe.

### Django Rest Framework

Django Rest Framework (DRF) là một framework được xây dựng trên Django để hỗ trợ phát triển các RESTful API. Django Rest Framework giúp việc xây dựng API trở nên dễ dàng và hiệu quả. Phù hợp cho nhiều quy mô dự án khác nhau.

Kiến trúc: DRF tách biệt giữa các tầng của ứng dụng: **View, Model, Serializer** và **Router**, tạo điều kiện cho việc xây dựng các endpoint API rõ ràng và dễ bảo trì.

* View: Tiếp nhận yêu cầu từ client, gọi model tương ứng và gửi phản hồi dưới dạng dữ liệu JSON hoặc các định dạng khác.
* Serializer: Chuyển đổi dữ liệu từ model Django thành định dạng JSON để tương thích với API, đồng thời xử lý việc nhận và xác thực dữ liệu từ client gửi về.
* Model: Quản lý và tương tác với cơ sở dữ liệu.
* **Router:** Tự động định tuyến các yêu cầu URL đến các view tương ứng trong API, giúp tổ chức cấu trúc API rõ ràng.

Ưu điểm:

* DRF tận dụng các cơ chế của Django, bao gồm mô hình ORM, bảo mật, và quản lý phiên người dùng.
* DRF có nhiều tính năng như: phân trang, xác thực, cấp quyền, lọc, và định dạng dữ liệu.
* DRF giúp nhanh chóng thiết kế và triển khai các API với cú pháp đơn giản và rõ ràng.
* DRF có khả năng tự động tạo ra tài liệu API.

Nhược điểm:

* DRF yêu cầu có kiến thức tốt về Django cũng như RESTful API.

## ReactJs

### Tổng quan về ReactJs

**ReactJs** là một thư viện JavaScript mã nguồn mở, được phát triển bởi Facebook, dùng để xây dựng giao diện người dùng (UI), đặc biệt là các ứng dụng web đơn trang (Single Page Application - SPA). ReactJS cho phép lập trình viên tạo các thành phần giao diện (components) có thể tái sử dụng và quản lý trạng thái của ứng dụng một cách hiệu quả. Điều này làm tăng tính linh hoạt và khả năng mở rộng của ứng dụng.

**Ưu điểm:**

* **Hiệu suất cao**: React sử dụng Virtual DOM (Document Object Model ảo), giúp tăng tốc độ cập nhật giao diện mà không cần phải thao tác trực tiếp với DOM thật.
* **Tái sử dụng component**: React được xây dựng dựa trên các thành phần (components), giúp dễ dàng tái sử dụng và duy trì code.
* **Dễ bảo trì**: Việc chia nhỏ giao diện thành các component riêng biệt giúp quản lý dễ dàng, đồng thời cải thiện khả năng bảo trì và mở rộng ứng dụng.
* **Hỗ trợ tốt cho SEO**: So với nhiều framework JavaScript khác, React có khả năng render từ phía server, giúp cải thiện SEO của các ứng dụng web.
* **Cộng đồng lớn**: React có một cộng đồng phát triển mạnh mẽ, với nhiều thư viện và công cụ hỗ trợ.

**Nhược điểm:**

* **Yêu cầu hiểu biết về JavaScript nâng cao**: Để làm việc tốt với React, lập trình viên cần nắm vững các khái niệm như ES6, JSX, và các hook.
* **Cấu trúc phức tạp hơn với các ứng dụng lớn**: Đối với các dự án lớn, việc quản lý trạng thái và các thành phần phức tạp có thể đòi hỏi thêm công cụ bổ trợ như Redux, gây khó khăn trong việc duy trì.

### Kiến trúc ReactJs

ReactJs dựa trên kiến trúc **component-based**. Mọi phần của giao diện đều được chia nhỏ thành các thành phần (components) độc lập. Kiến trúc này có những lợi ích lớn trong việc quản lý và tái sử dụng code, đồng thời cho phép xây dựng giao diện phức tạp và dễ bảo trì.

Các thành phần trong React có thể là:

* **Functional Components**: Là những thành phần đơn giản chỉ nhận vào props và trả về JSX (HTML trong JavaScript).
* **Class Components**: Trước khi React Hooks ra đời, class components được dùng để quản lý trạng thái và các phương thức vòng đời. Tuy nhiên, các thành phần class đang dần được thay thế bởi functional components với hooks.

Kiến trúc của React giải quyết vấn đề:

* **Tách biệt UI và logic**: Mỗi component có thể tự quản lý giao diện và logic riêng, giúp mã nguồn dễ bảo trì và tái sử dụng.
* **Virtual DOM**: Khi trạng thái (state) thay đổi, React sẽ so sánh DOM ảo với DOM thật và chỉ cập nhật những thay đổi cần thiết, giúp tăng hiệu suất đáng kể.

### Sử dụng ReactJs

Để sử dụng React, các bước cơ bản gồm:

B1: **Cài đặt React**: Bạn có thể khởi tạo một ứng dụng React bằng công cụ:

npx create-react-app my-app

cd my-app

npm start

B2: **Tạo component**: Mỗi giao diện trong React là một component. **Sử dụng JSX**: React sử dụng cú pháp JSX, cho phép viết HTML bên trong JavaScript. JSX sẽ được chuyển đổi thành các lệnh JavaScript tương ứng khi chạy. Ví dụ, tạo một component đơn giản hiển thị danh sách xe:

import React from 'react';

function CarList() {

const cars = ['Toyota', 'Honda', 'Ford'];

return (

<ul>

{cars.map((car, index) => <li key={index}>{car}</li>)}

</ul>

);

}

export default CarList;

**Quản lý trạng thái (state)**: React cung cấp cơ chế state để quản lý dữ liệu động. Ví dụ:

import React, { useState } from 'react';

function Counter() {

const [count, setCount] = useState(0);

return (

<div>

<p>Bạn đã bấm {count} lần</p>

<button onClick={() => setCount(count + 1)}>

Tăng số lần bấm

</button>

</div>

);

}

export default Counter;

**Hooks**: Từ React phiên bản 16.8, hooks được giới thiệu để quản lý state và các phương thức vòng đời trong functional components. Một số hook quan trọng như useState, useEffect.

## MySQL

### Tổng quan về MySQL

**MySQL** là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS - Relational Database Management System) mã nguồn mở, được phát triển và duy trì bởi Oracle Corporation. MySQL sử dụng ngôn ngữ SQL (Structured Query Language) để tương tác với cơ sở dữ liệu và là một trong những hệ quản trị cơ sở dữ liệu phổ biến nhất trên thế giới.

MySQL được sử dụng rộng rãi trong việc lưu trữ, truy xuất và quản lý dữ liệu cho các ứng dụng web, từ các website nhỏ đến các ứng dụng lớn như Facebook, Twitter và YouTube.

Ưu điểm:

* **Hiệu suất cao**: MySQL hoạt động tốt ngay cả với các ứng dụng có khối lượng dữ liệu lớn.
* **Dễ sử dụng**: MySQL có cú pháp SQL đơn giản và dễ học.
* **Mã nguồn mở**: MySQL miễn phí và có cộng đồng lớn hỗ trợ, cùng với các phiên bản thương mại cung cấp thêm nhiều tính năng.
* **Bảo mật tốt**: Hỗ trợ các tính năng bảo mật cao như mã hóa kết nối SSL, xác thực người dùng mạnh mẽ.

Nhược điểm:

* **Hạn chế với các tính năng phức tạp**: So với các hệ quản trị cơ sở dữ liệu thương mại khác như Oracle hoặc SQL Server, MySQL có thể thiếu một số tính năng nâng cao.
* **Giới hạn trong quy mô lớn**: Với các hệ thống yêu cầu tính năng phân tán hoặc khối lượng giao dịch rất lớn, MySQL có thể gặp khó khăn trong việc mở rộng.

### Kiến trúc của MySQL

MySQL có kiến trúc dựa trên mô hình client-server. Cấu trúc dữ liệu trong MySQL được tổ chức dưới dạng bảng, trong đó mỗi bảng gồm nhiều hàng (record) và cột (field). Dữ liệu trong các bảng có thể được liên kết với nhau thông qua các khóa ngoại (foreign keys), giúp tổ chức dữ liệu có tính liên quan, trong đó:

* **MySQL Server**: Đóng vai trò như máy chủ quản lý cơ sở dữ liệu, chịu trách nhiệm xử lý các yêu cầu từ client và thực hiện các tác vụ liên quan đến dữ liệu.
* **Client**: Là các ứng dụng hoặc công cụ giao tiếp với MySQL Server để gửi các câu truy vấn, cập nhật dữ liệu và nhận kết quả.

## Các công cụ hỗ trợ khác

### Postman

**Postman** là một công cụ phổ biến giúp lập trình viên và nhà phát triển API thực hiện, kiểm thử, và phát triển các API (Application Programming Interface) một cách hiệu quả. Postman hỗ trợ giao tiếp với nhiều loại API, bao gồm REST, SOAP, và GraphQL, và cho phép thực hiện các thao tác như gửi yêu cầu (request), nhận phản hồi (response), xác thực dữ liệu, và tạo tài liệu API.

### Pythonanywhere

**PythonAnywhere** là một nền tảng đám mây cung cấp môi trường phát triển và triển khai các ứng dụng web bằng Python. Được thiết kế đặc biệt cho Python, PythonAnywhere giúp các lập trình viên dễ dàng viết, chạy và triển khai các ứng dụng web mà không cần cài đặt phần mềm hoặc quản lý máy chủ. PythonAnywhere hỗ trợ nhiều framework web phổ biến như Django, Flask, và Bottle, đồng thời cung cấp một giao diện người dùng đồ họa để quản lý tệp tin, cơ sở dữ liệu và các công việc nền (background tasks).

### Momo API

Momo API là một giải pháp thanh toán trực tuyến được phát triển bởi Momo, một trong những ví điện tử lớn nhất Việt Nam. API này cho phép các hệ thống tích hợp chức năng thanh toán trực tuyến trực tiếp qua nền tảng của Momo, hỗ trợ đa dạng các phương thức thanh toán như quét mã QR, chuyển tiền qua số điện thoại, và thanh toán bằng tài khoản ngân hàng liên kết.

Kiến trúc: Momo API sử dụng kiến trúc RESTful, một trong những chuẩn kiến trúc phổ biến nhất cho các dịch vụ web. Hệ thống dựa trên giao thức HTTP(S) với các phương thức chính như POST và GET để gửi và nhận dữ liệu thanh toán. Điều này giúp đơn giản hóa quá trình tích hợp và tương thích tốt với nhiều nền tảng.

Ưu điểm:

* Tích hợp nhanh chóng và dễ dàng với các nền tảng web và mobile.
* Độ bảo mật cao nhờ các lớp mã hóa thông tin và chứng nhận PCI DSS (tiêu chuẩn bảo mật dữ liệu).
* Hỗ trợ nhiều phương thức thanh toán khác nhau, tạo sự tiện lợi cho người dùng.

Khuyết điểm:

* Hạn chế trong việc sử dụng ngoài lãnh thổ Việt Nam do hạ tầng chủ yếu hướng đến thị trường nội địa.
* Có thể gặp phải các vấn đề về tốc độ xử lý khi hệ thống có lượng giao dịch lớn.

Để sử dụng Momo API, cần thực hiện các bước:

1. Đăng ký tài khoản Momo Merchant và tạo ứng dụng.
2. Lấy thông tin API key, secret key, và endpoint từ Momo.
3. Gửi yêu cầu thanh toán với thông tin giao dịch (mã hóa) thông qua HTTP request.
4. Momo trả về kết quả giao dịch (success/failure) và mã xác nhận.
5. Hệ thống sử dụng kết quả trả về để xác nhận và hoàn thành quá trình thanh toán.

### Face-api.js

**Face-api.js** là một thư viện JavaScript mã nguồn mở được xây dựng trên nền tảng TensorFlow.js, giúp thực hiện các tác vụ liên quan đến nhận diện và xác thực khuôn mặt trực tiếp trên trình duyệt mà không cần cài đặt phần mềm đặc biệt. Thư viện này cho phép phát hiện khuôn mặt, nhận diện các điểm đặc trưng (face landmarks), phân loại biểu cảm, và đặc biệt là xác thực danh tính dựa trên khuôn mặt.

Ưu điểm:

* Chạy trực tiếp trên client (trình duyệt), giảm tải cho máy chủ.
* Không yêu cầu kết nối mạng hoặc xử lý server-side, tăng cường bảo mật cho dữ liệu người dùng.
* Dễ dàng tích hợp vào các ứng dụng web với các nền tảng như React, Vue, hoặc Angular.

Khuyết điểm:

* Hiệu suất có thể bị ảnh hưởng bởi sức mạnh của thiết bị (máy tính, điện thoại) người dùng.
* Không phù hợp cho những ứng dụng yêu cầu tốc độ và độ chính xác cao như nhận diện trong môi trường thiếu sáng hoặc đông người.

Cách sử dụng:

1. Tích hợp thư viện face-api.js vào trang web bằng cách nhúng qua CDN hoặc cài đặt qua npm.
2. Sử dụng camera để chụp hoặc truy xuất hình ảnh khuôn mặt người dùng.
3. Gọi hàm faceapi.detectAllFaces() để phát hiện khuôn mặt trong hình ảnh
4. Dùng các hàm phân tích khác để nhận diện các thuộc tính khuôn mặt hoặc so sánh với dữ liệu đã lưu trữ.

### Plate Recognizer

**Plate Recognizer** là một dịch vụ API nhận diện biển số xe (Automatic License Plate Recognition - ALPR) dựa trên công nghệ thị giác máy tính (computer vision) và machine learning. API này cho phép các ứng dụng nhận diện biển số xe từ hình ảnh hoặc video trong thời gian thực, hỗ trợ nhiều định dạng và quốc gia khác nhau. Plate Recognizer thường được sử dụng trong các hệ thống bãi đỗ xe, quản lý giao thông, an ninh, và tự động hóa công nghiệp.

Kiến trúc: PlateRecognizer sử dụng kiến trúc dựa trên mô hình OCR kết hợp với xử lý hình ảnh nâng cao. Các hình ảnh được gửi đến hệ thống sẽ trải qua quá trình phân tích và nhận diện các ký tự trên biển số xe. Kiến trúc này được thiết kế để đảm bảo tốc độ và độ chính xác cao, phục vụ cho các tình huống đòi hỏi xử lý thời gian thực như quản lý bãi đỗ xe.

Ưu điểm:

* Chính xác và nhanh chóng trong việc nhận diện biển số xe ở nhiều điều kiện ánh sáng khác nhau.
* Hỗ trợ nhiều quốc gia và định dạng biển số khác nhau.
* Cung cấp cả phiên bản cloud-based và on-premise, linh hoạt cho các hệ thống yêu cầu bảo mật cao.

Khuyết điểm:

* Chi phí sử dụng có thể khá cao đối với các hệ thống có số lượng lớn giao dịch.
* Hiệu suất có thể giảm trong các điều kiện ánh sáng yếu hoặc biển số xe bị che khuất một phần.

Cách sử dụng:

1. Tạo tài khoản và lấy API key từ Plate Recognizer.
2. Gửi hình ảnh chứa biển số xe đến endpoint của API bằng phương thức HTTP POST.
3. API sẽ trả về dữ liệu bao gồm số biển, quốc gia, và xác suất nhận diện.
4. Sử dụng kết quả này để xác định xe đã đăng ký trong hệ thống.

### SMTP Gmail

**SMTP (Simple Mail Transfer Protocol)** là giao thức tiêu chuẩn để gửi email qua Internet. **SMTP của Gmail** là dịch vụ cung cấp bởi Google, cho phép người dùng hoặc ứng dụng gửi email thông qua máy chủ Gmail bằng cách sử dụng tài khoản Gmail.

SMTP của Gmail giúp bạn dễ dàng tích hợp tính năng gửi email vào các ứng dụng hoặc hệ thống tự động như trang web, ứng dụng di động mà không cần thiết lập máy chủ email riêng.

### OAuth2

OAuth 2.0 là một giao thức mở cho phép các ứng dụng bên thứ ba truy cập vào tài nguyên của người dùng trên một dịch vụ khác (ví dụ như Google, Facebook) mà không cần tiết lộ thông tin đăng nhập của người dùng. Thay vì chia sẻ mật khẩu, hệ thống sử dụng **token** để ủy quyền quyền truy cập vào các tài nguyên.

Ưu điểm:

* Giảm thiểu rủi ro bảo mật do người dùng không cần cung cấp thông tin đăng nhập trực tiếp cho ứng dụng bên thứ ba.
* Linh hoạt, cho phép người dùng kiểm soát quyền truy cập theo từng mức độ khác nhau (chỉ cấp quyền một phần tài khoản).
* Hỗ trợ đa nền tảng và dễ dàng tích hợp với nhiều dịch vụ lớn như Google, Facebook, GitHub.

Khuyết điểm:

* Việc cấu hình OAuth 2.0 có thể phức tạp, đòi hỏi phải hiểu rõ cơ chế bảo mật và các luồng xác thực khác nhau.

# PHÁT TRIỂN HỆ THỐNG BÃI ĐỖ Ô TÔ TÍCH HỢP CÁC KỸ THUẬT TRÍ TUỆ NHÂN TẠO

## Giới thiệu đề tài

Trong bối cảnh các đô thị hiện đại ngày càng đông đúc, việc tìm kiếm chỗ đỗ xe phù hợp đã trở thành một bài toán nan giải đối với nhiều người. Với nhu cầu về sự tiện lợi và hiệu quả, việc phát triển hệ thống bãi đỗ ô tô thông minh nhằm đáp ứng yêu cầu của người dùng và giúp tối ưu hóa việc quản lý bãi đỗ. Đề tài này tập trung vào việc xây dựng hệ thống bãi đỗ tích hợp các kỹ thuật nhận dạng thông minh, nhằm mang đến trải nghiệm đỗ xe tự động, an toàn và tiết kiệm thời gian. Sử dụng các giải pháp công nghệ như nhận diện khuôn mặt (face-api.js), phân tích biển số xe (plate recognizer), hệ thống sẽ tự động hóa quy trình ra vào bãi, hỗ trợ quản lý chỗ đỗ và khả năng phục vụ khách hàng một cách toàn diện.

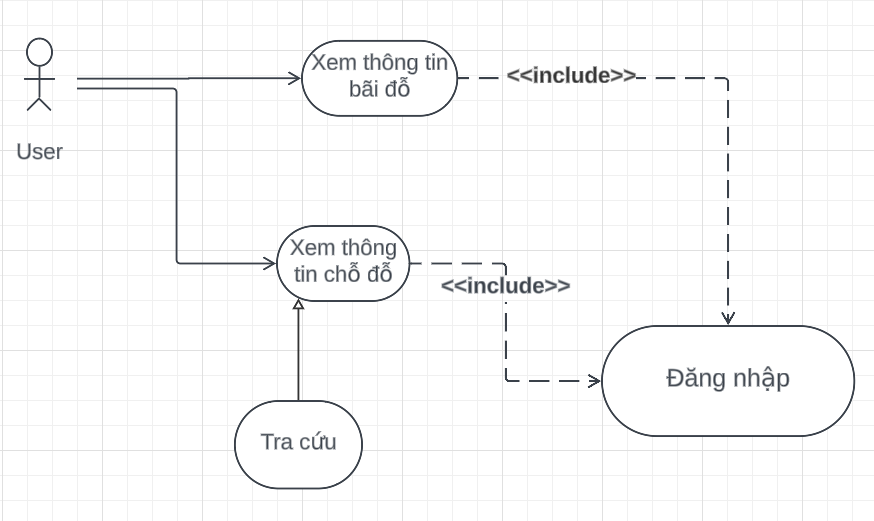
## Phân tích hệ thống

Tổng quát lược đồ use case:

* **Khách hàng**: Tương tác chủ yếu với hệ thống để quản lý thông tin cá nhân, thông tin xe, đặt chỗ, mua chỗ theo tháng hoặc quý, khiếu nại và đánh giá.
* **Nhân viên**: Báo cáo chỗ đỗ cần bảo trì, theo dõi tình trạng chỗ đỗ và hỗ trợ khách hàng vào bãi đúng chỗ và ra khỏi bãi.
* **Admin**: Quản lý toàn diện hệ thống, bao gồm bãi đỗ xe, người dùng, đăng ký chỗ, đặt chỗ, hóa đơn thanh toán và khiếu nại, xác nhận chỗ cần bảo trì, tạo báo cáo và phân tích về doanh thu và trải nghiệm người dùng.

### Chức năng chung của toàn bộ User

* Mô tả: User có thể xem thông tin bãi đỗ xe và chỗ đỗ xe.
* Actor chính: User.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng đăng nhập vào hệ thống bằng tài khoản vẫn còn hoạt động.
* Luồng hoạt động:
  1. Truy cập thông tin bãi đỗ xe.
  2. Người dùng chọn bãi đỗ xe.
  3. Hệ thống trả về thông tin chỗ đỗ của bãi đỗ đã chọn.
* Luồng thay thế: Ở bước 2, nếu người dùng không chọn bãi đỗ hệ thống sẽ không trả về chỗ đỗ.



*Hình 3.1. Chức năng chung của User*

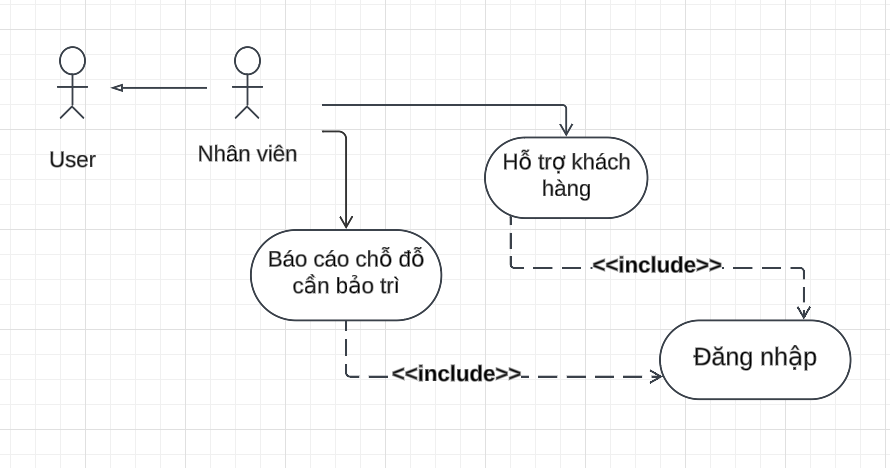
### Vai trò của Nhân viên

***Hỗ trợ khách hàng:***

* Mô tả: Nhân viên được xem kết quả lịch sử ra vào bãi của khách hàng để hướng dẫn khách hàng vào chỗ đỗ xe.
* Actor chính: Nhân viên.
* Actor phụ: không có.
* Tiền điều kiện: Đăng nhập tài khoản còn hoạt động với quyền nhân viên.
* Luồng hoạt động:
  1. Khách hàng cung cấp thông tin vào bãi đỗ xe.
  2. Nhân viên xác nhận thông tin và biết chỗ đỗ xe.
  3. Hướng dẫn khách hàng vào chỗ đỗ xe.
* Luồng thay thế: Ở bước 1, nếu khách hàng cung cấp thông tin không đúng nhân viên sẽ không cho phép vào bãi đỗ xe.

***Báo cáo chỗ đỗ cần bảo trì:***

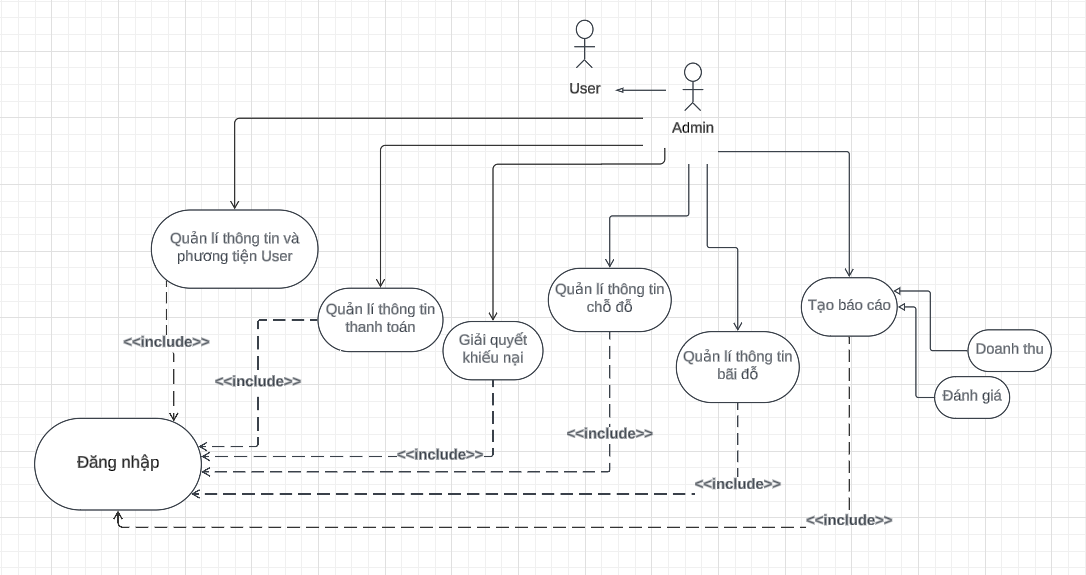
* Mô tả: Nhân viên khảo sát hiện trạng bãi đỗ và báo cáo các vị trí cần bảo trì cho quản trị viên
* Actor chính: Nhân viên
* Actor phụ: không có
* Tiền điều kiện: Đăng nhập tài khoản còn hoạt động với quyền nhân viên.
* Luồng hoạt động:
  1. Nhân viên khảo sát và nắm danh sách các chỗ đỗ cần bảo trì.
  2. Nhân viên gửi thông tin các chỗ đỗ đến Quản trị viên.
  3. Quản trị viên xác nhận thông tin và thay đổi trạng thái chỗ đỗ.
* Luồng thay thế: Ở bước 3, nếu Nhân viên gửi vị trí chỗ đỗ không hợp lệ hệ thống sẽ báo lỗi.



*Hình 3.2. Vai trò của nhân viên*

### Quản trị viên quản trị toàn diện hệ thống

* Mô tả: Admin có thể xem, thêm, sửa, xóa tất cả các thông tin trong hệ thống, cập nhật trạng thái chỗ đỗ, tạo báo cáo doanh thu và trải nghiệm người dùng, giải quyết và cập nhật trạng thái khiếu nại.
* Actor chính: Admin.
* Actor phụ: không có.
* Tiền điều kiện: đăng nhập Adminsite bằng tài khoản superuser.
* Luồng hoạt động:
  1. Thực hiện các thao tác quản trị hoặc tạo báo cáo theo ý muốn.
  2. Hệ thống thực hiện và trả về kết quả.

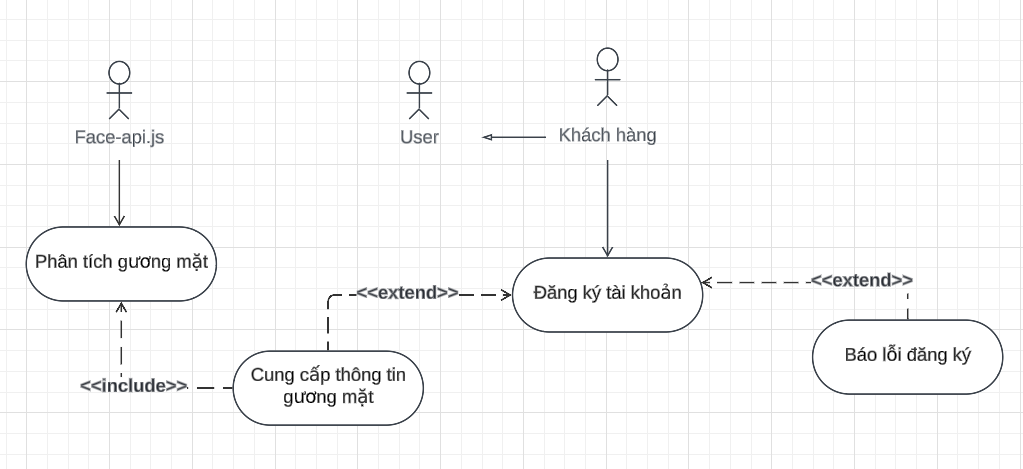


*Hình 3.3. Quản trị viên quản trị toàn diện hệ thống.*

### Khách hàng tương tác với hệ thống

#### Khách hàng đăng ký tài khoản

* Mô tả: Khách hàng được phép đăng ký tài khoản, với tùy chọn có cung cấp thông tin gương mặt hay không
* Actor chính: Khách hàng
* Actor phụ: Face-api.js
* Tiền điều kiện: không có
* Luồng hoạt động:
  1. Khách hàng nhập username và password, xác nhận password.
  2. Nhập thêm các thông tin cá nhân như email, số điện thoại, ngày sinh, họ và tên.
  3. Chụp ảnh và phân tích gương mặt.
  4. Đồng ý với các điều khoản dịch vụ và chính sách của bãi xe.
  5. Chọn đăng ký.
* Luồng thay thế:
  + Ở bước 1, nếu xác nhận mật khẩu không chính xác sẽ báo lỗi.
  + Ở bước 2, nếu định dạng các thông tin không đúng hoặc bỏ trống sẽ báo lỗi.
  + Ở bước 3, nếu không phát hiện được khuôn mặt sẽ báo phân tích không thành công, phải chọn đồng ý bỏ qua phân tích gương mặt nếu muốn tiếp tục đăng ký.
  + Ở bước 4, nếu không đồng ý với chính sách sẽ không thể click đăng ký.
* Luồng ngoại lệ: Nếu username, email, số điện thoại, hoặc dữ liệu phân tích khuôn mặt bị trùng sẽ đăng ký không thành công.



*Hình 3.4. Khách hàng đăng ký tài khoản*

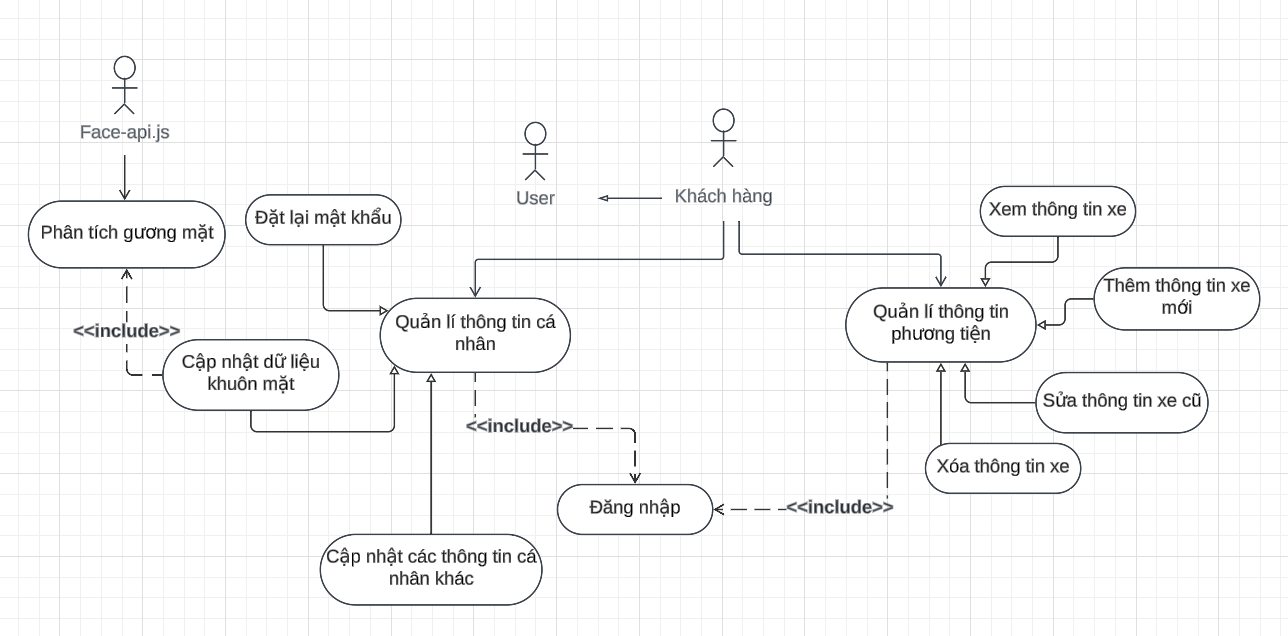
#### Khách hàng quản lý thông tin cá nhân, thông tin phương tiện

***Quản lí thông tin phương tiện:***

* Mô tả: Khách hàng được phép xem, thêm, sửa, xóa thông tin phương tiện thuộc về mình.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điệu kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Luồng hoạt động:
  1. Truy cập giao diện quản lí xe.
  2. Thực hiện các thao tác quản lí theo ý muốn.
  3. Hệ thống thực hiện và trả về kết quả.
* Luồng thay thế: Ở bước 2, nếu thông tin xe không đúng định dạng hệ thống sẽ báo lỗi.
* Luồng ngoại lệ: Khách hàng sẽ không có quyền quản trị phương tiện không thuộc sở hữu của mình.

***Quản lí thông tin cá nhân:***

* Mô tả: Khách hàng được phép xem và cập nhật thông tin cá nhân đã đăng ký, như đặt lại mật khẩu, cập nhật dữ liệu không mặt, cập nhật các thông tin cá nhân khác như email, số điện thoại, họ tên và ngày sinh.
* Actor chính: Khách hàng.
* Actor phụ: Face-api.js.
* Tiền điều kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Luồng hoạt động:
  1. Truy cập đến trang thông tin cá nhân.
  2. Thực hiện các thay đổi theo mong muốn.
  3. Hệ thống thực hiện và trả về kết quả.
* Luồng thay thế: Ở bước 2, nếu định dạng các thông tin như mật khẩu, email, số điện thoại, ngày sinh, dữ liệu khuôn mặt không đúng hệ thống sẽ báo lỗi.
* Luồng ngoại lệ: Khách hàng sẽ không có quyền quản trị thông tin cá nhân của người khác.



*Hình 3.5. Khách hàng quản lí thông tin cá nhân, thông tin phương tiện*

#### Khách hàng đặt chỗ, đăng ký mua chỗ, gia hạn chỗ

***Đặt chỗ:***

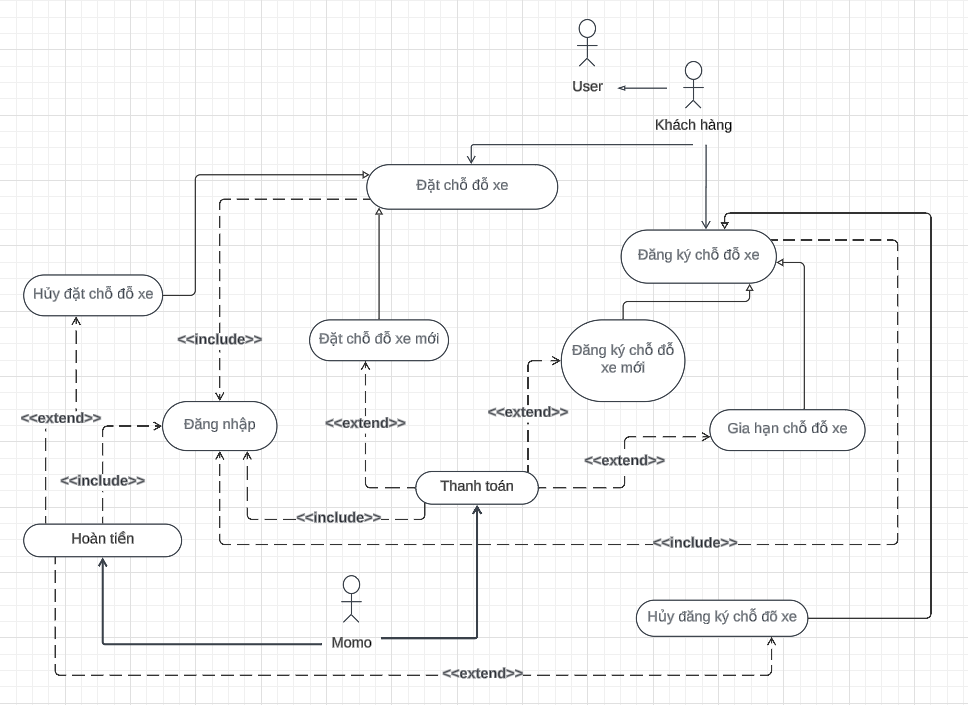
* Mô tả: Khách hàng được phép đặt chỗ trước khi vào bãi xe.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Hậu điều kiện: gửi email thông tin đặt chỗ bao gồm mã đặt chỗ, chỗ đỗ cho khách hàng.
* Luồng hoạt động:
  1. Xem thông tin chỗ đỗ xe khả dụng.
  2. Chọn chỗ để tiến hành đặt chỗ.
  3. Chọn giờ vào và ra bãi xe.
  4. Tiến hành thanh toán.
  5. Trả về kết quả đặt chỗ thành công.
* Luồng thay thế:
  + Ở bước 2, nếu chọn chỗ có trạng thái không khả dụng sẽ không thể đặt chỗ thành công.
  + Ở bước 3, đặt trước tối đa 5 giờ trước giờ vào bãi.
  + Ở bước 4, khách hàng có thể từ chối thanh toán, hoặc yêu cầu hoàn tiền trong vòng 2 giờ.
* Luồng ngoại lệ: Khách hàng không thể đặt chỗ với phương tiện chưa đăng ký dưới quyền của mình.

***Đăng ký chỗ:***

* Mô tả: Khách hàng được phép đăng ký mua một chỗ đỗ xe trong vòng 1 tháng hoặc 1 quý.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Hậu điều kiện: gửi email thông tin đăng ký chỗ bao gồm mã đặt chỗ, chỗ đỗ cho khách hàng.
* Luồng hoạt động:
  1. Xem thông tin các chỗ đỗ xe khả dụng.
  2. Chọn chỗ đỗ để tiến hành đăng ký chỗ.
  3. Chọn gói tháng hoặc gói quý.
  4. Tiến hành thanh toán.
  5. Trả về kết quả đăng ký chỗ thành công.
* Luồng thay thế:
  + Ở bước 2, nếu chọn chỗ có trạng thái không khả dụng sẽ không thể đăng ký chỗ thành công.
  + Ở bước 4, khách hàng có thể từ chối thanh toán, hoặc yêu cầu hoàn tiền trong vòng 2 giờ.

***Gia hạn chỗ:***

* Mô tả: Khách hàng được phép gia hạn những chỗ đã mua vẫn còn thời hạn sử dụng.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điệu kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Hậu điều kiện: gửi email thông tin gia hạn chỗ bao gồm mã đặt chỗ, chỗ đỗ cho khách hàng.
* Luồng hoạt động:
  1. Truy cập giao diện gia hạn chỗ.
  2. Chọn chỗ muốn gia hạn.
  3. Chọn gói tháng hoặc quý.
  4. Tiến hành thanh toán.
  5. Trả về kết quả gia hạn chỗ thành công.
* Luồng thay thế:
  + Ở bước 2, nếu chọn chỗ có trạng thái đã hết hạn sẽ không thể gia hạn chỗ thành công.
  + Ở bước 4, khách hàng có thể từ chối thanh toán, hoặc yêu cầu hoàn tiền trong vòng 2 giờ.
* Luồng ngoại lệ: Khách hàng không thể gian hạn chỗ không do mình đăng ký.



*Hình 3.6. Khách hàng đặt chỗ, đăng ký mua chỗ*

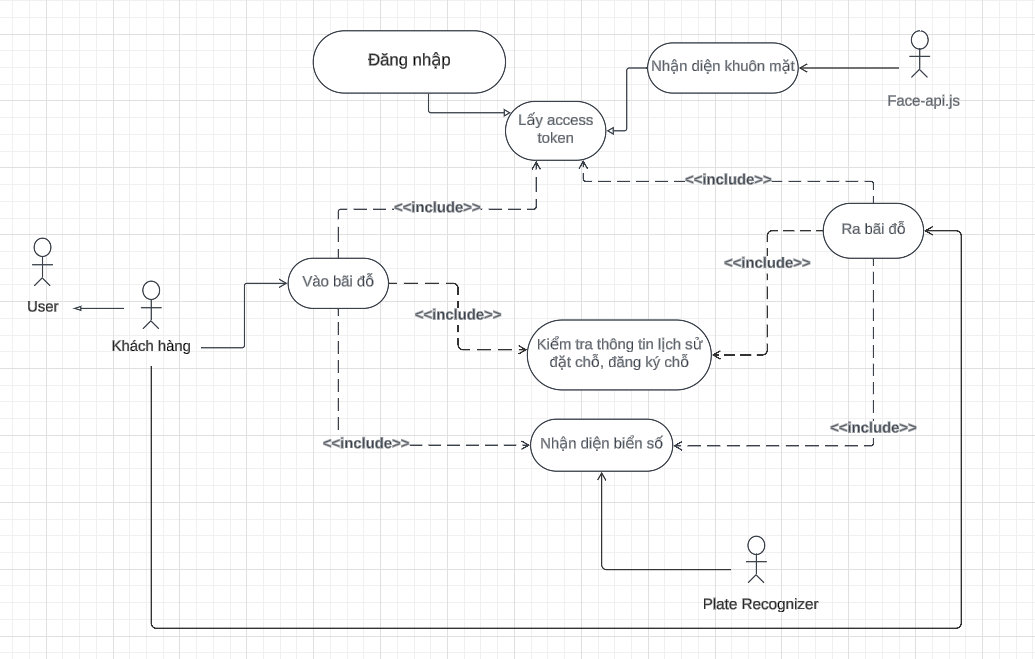
#### Khách hàng vào và ra bãi đỗ xe

***Quá trình vào bãi đỗ xe****:*

* Mô tả: Khách hàng thêm lịch sử đỗ xe khi vào bãi đỗ.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng cung cấp một access\_token chỉ có quyền thêm lịch sử vào bãi và hình ảnh xe.
* Hậu điều kiện: gửi email xe đã vào bãi cho khách hàng bao gồm thông tin chỗ đỗ, thời gian vào, thời gian ra dự kiến.
* Luồng hoạt động:
  1. Khách hàng cung cấp một access\_token chỉ có quyền thêm lịch sử vào hệ thống thông qua nhận diện khuôn mặt hoặc lấy từ ứng dụng web.
  2. Khách hàng cung cấp hình ảnh xe để phân tích so sánh biển số, kiểm tra xe có thuộc người dùng hay không, và lưu trữ để giải quyết khiếu nại (nếu cần).
  3. So sánh thông tin người dùng và xe nhận được xem người dùng đăng ký chỗ hay đặt chỗ.
  4. Hiển thị lịch sử vào bãi bao gồm mã chỗ đỗ xe để khách hàng có thể vào bãi xe.
* Luồng thay thế:
  + Ở bước 1, nếu không cung cấp được access\_token sẽ không thể vào bãi xe.
  + Ở bước 2, nếu thông tin xe không khớp với các xe mà khách hàng đã đăng ký thì không cho phép vào bãi xe.
  + Ở bước 3, nếu không tìm thấy thông tin đăng ký chỗ hay đặt chỗ thì không cho phép vào bãi xe.

***Quá trình ra khỏi bãi đỗ xe****:*

* Mô tả: Khách hàng cập nhật lịch sử đỗ xe khi ra khỏi bãi đỗ.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng cung cấp một access\_token chỉ có quyền cập nhật lịch sử đỗ xe và hình ảnh xe.
* Hậu điều kiện: gửi email xe đã ra khỏi bãi cho khách hàng bao gồm thông tin chỗ đỗ, thời gian ra, vé phạt muộn giờ ra (nếu có).
* Luồng hoạt động:
  1. Khách hàng cung cấp một access\_token chỉ có quyền cập nhật lịch sử hệ thống thông qua nhận diện khuôn mặt hoặc lấy từ ứng dụng web.
  2. Khách hàng cung cấp hình ảnh xe để phân tích so sánh biển số, kiểm tra xe có thuộc người dùng hay không, và lưu trữ để giải quyết khiếu nại (nếu cần).
  3. So sánh thông tin người dùng và xe nhận được với lịch sử vào bãi xe.
  4. Hiển thị thông báo đã ra khỏi bãi xe thành công.
* Luồng thay thế:
  + Ở bước 1, nếu không cung cấp được access\_token sẽ không thể cập nhật lịch sử ra khỏi bãi xe.
  + Ở bước 2, nếu thông tin xe không khớp với các xe mà khách hàng đã đăng ký thì không cho phép vào bãi xe.
  + Ở bước 3, nếu không tìm thấy lịch sử vào phù hợp thì sẽ thông báo không thể ra khỏi bãi.



*Hình 3.7. Khách hàng vào và ra bãi đỗ xe*

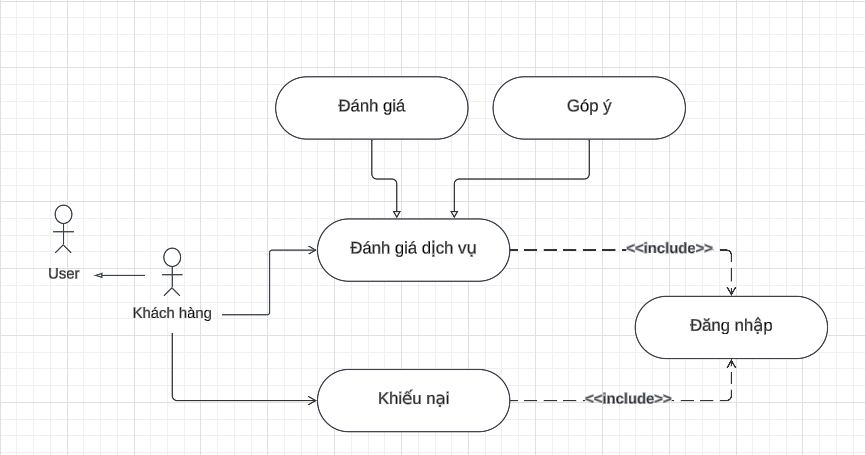
#### Khách hàng đánh giá dịch vụ, khiếu nại

***Khách hàng đánh giá dịch vụ:***

* Mô tả: Khách hàng được đánh giá chất lượng dịch vụ của bãi xe.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Luồng hoạt động:
  1. Truy cập giao diện đánh giá bãi đỗ xe.
  2. Đánh giá theo thang điểm từ 1 đến 5 sao và để lại bình luận.
  3. Trả về kết quả đã gửi đánh giá thành công
* Luồng thay thế: Ở bước 2, nếu khách hàng không đánh giá sẽ không trả về kết quả gì.

***Khách hàng khiếu nại:***

* Mô tả: Khách hàng có thể gửi khiếu nại của mình trong quá trình đậu xe tại bãi đến Admin.
* Actor chính: Khách hàng.
* Actor phụ: không có.
* Tiền điều kiện: Người dùng đăng nhập bằng tài khoản có quyền khách hàng vẫn còn hoạt động.
* Luồng hoạt động:
  1. Truy cập giao diện khiếu nại.
  2. Gửi khiếu nại đến Admin thông qua mã của lịch sử ra vào bãi xe.
  3. Trả về thời gian giải quyết khiếu nại dự kiến và thông tin liên lạc với Quản trị viên.
* Luồng thay thế: Ở bước 2, nếu khách hàng gửi lịch sử không có lưu trữ ảnh lúc vào và ra bãi xe thì không thể khiếu nại.
* Luồng ngoại lệ: nếu gửi khiếu nại với thông tin lịch sử đậu xe tại bãi thuộc người dùng khác thì sẽ không được giải quyết.



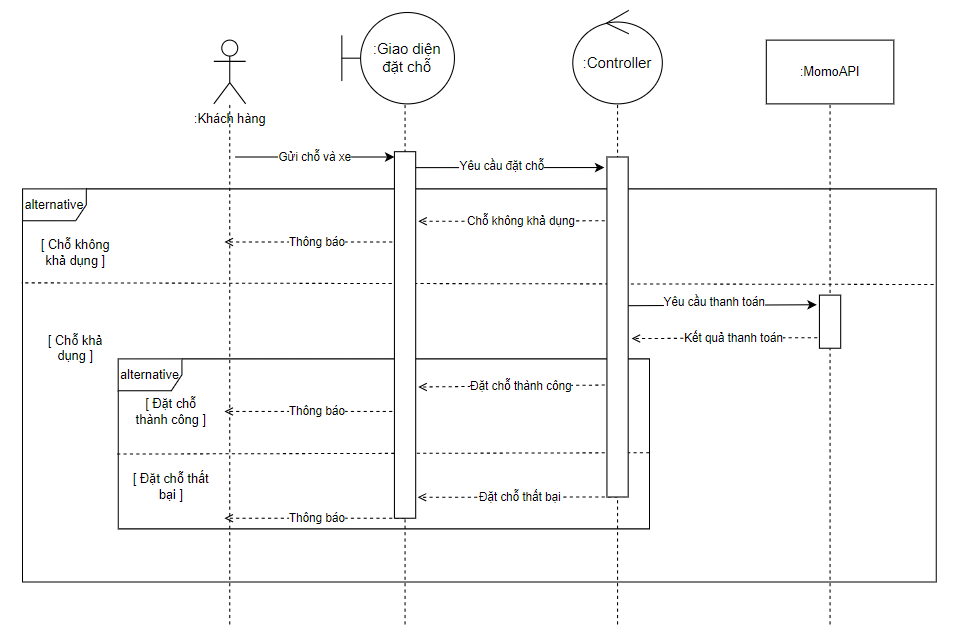
*Hình 3.8. Khách hàng đánh giá và khiếu nại*

## Thiết kế hệ thống

### Sequence diagram

#### Khách hàng đặt chỗ đỗ xe

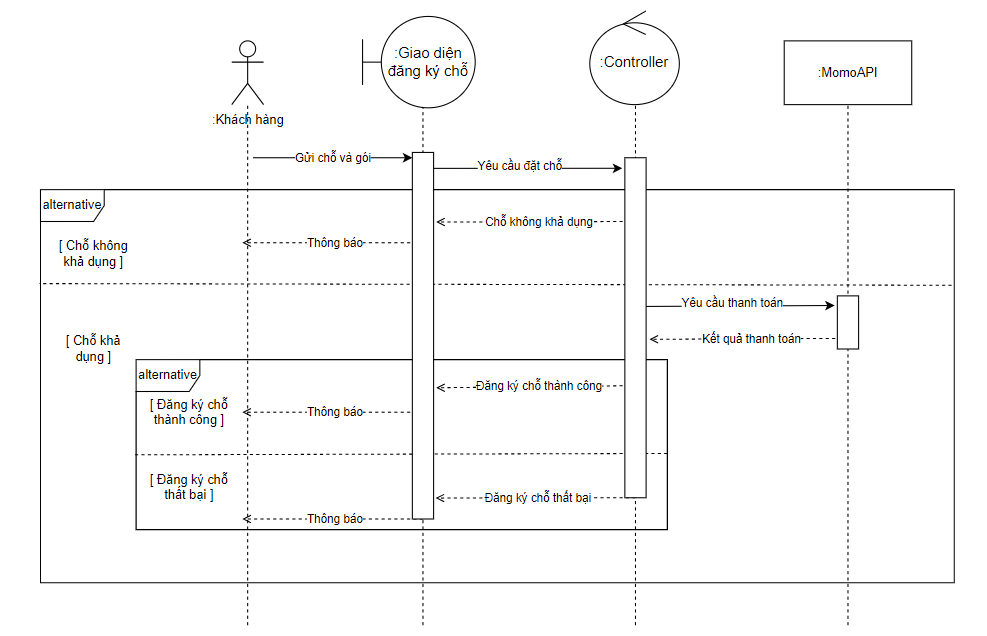
Khách hàng gửi thông tin chỗ và xe cần đặt, trả về kết quả đặt xe thất bại nếu chỗ không khả dụng, yêu cầu thanh toán nếu chỗ khả dụng, nhận kết quả thanh toán thành công thì ghi nhận đặt chỗ thành công và thông báo, ngược lại nếu thanh toán không thành công thì đặt chỗ không thành công.



*Hình 3.9. Sequence diagram Khách hàng đặt chỗ*

#### Khách hàng đăng ký chỗ đỗ xe

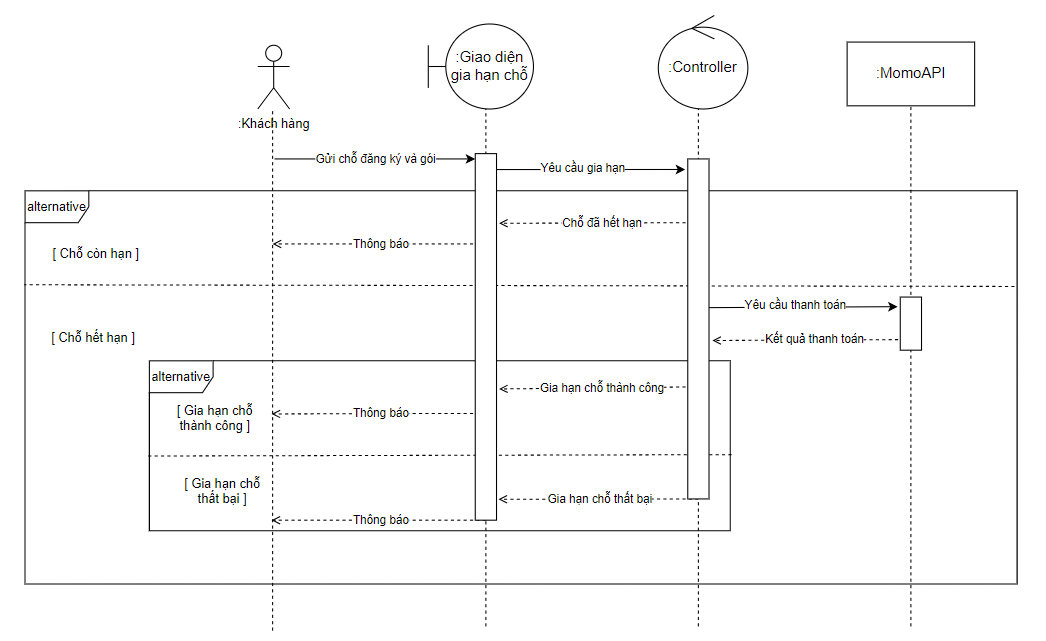
Khách hàng gửi thông tin chỗ và gói muốn đăng ký, trả về kết quả đăng ký chỗ thất bại nếu chỗ không khả dụng, yêu cầu thanh toán nếu chỗ khả dụng, nhận kết quả thanh toán thành công thì ghi nhận đăng ký chỗ thành công và thông báo, ngược lại nếu thanh toán không thành công thì đăng ký chỗ không thành công.



*Hình 3.10. Sequence diagram Khách hàng đăng ký chỗ*

#### Khách hàng gia hạn đăng ký

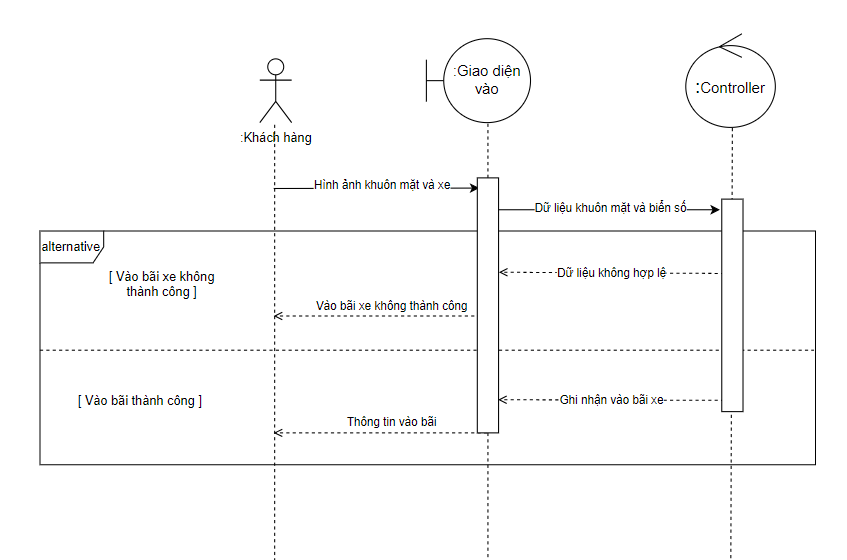
Khách hàng gửi thông tin chỗ đã đăng ký và gói muốn gia hạn, thông báo nếu chỗ đã hết hạn, ngược lại nếu chỗ còn hạn đăng ký thì yêu cầu thanh toán, thông báo gia hạn chỗ thành công nếu kết quả gia hạn chỗ thành công, ngược lại thông báo gia hạn chỗ thất bại nếu không thanh toán thành công.



*Hình 3.11. Sequence diagram Khách hàng gia hạn chỗ*

#### Khách hàng vào bãi đỗ xe

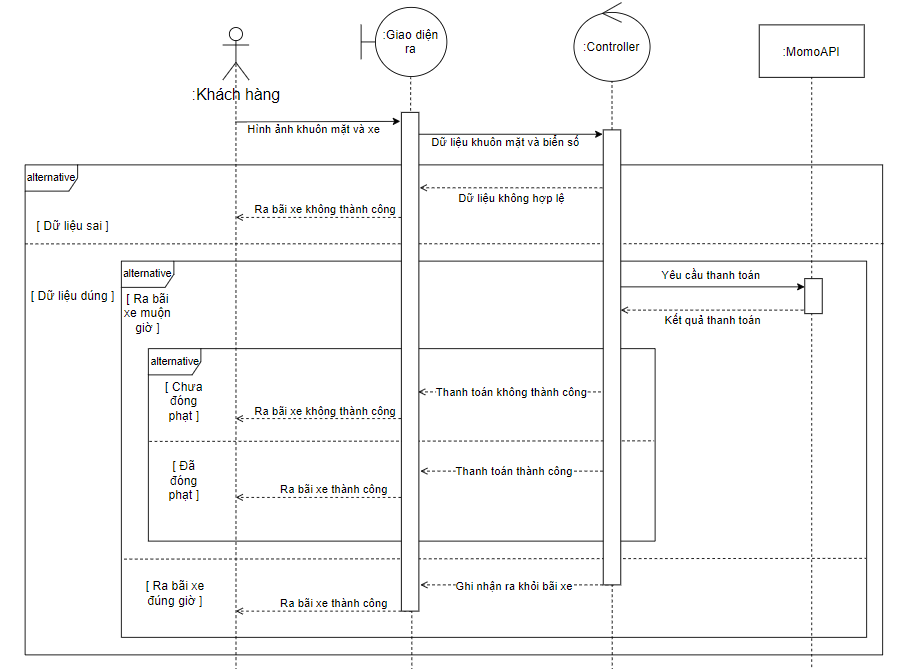
Khách hàng cung cấp hình ảnh khuôn mặt và xe khi vào bãi xe, giao diện ở client sẽ phân tích dữ liệu khuôn mặt và gọi API của Plate Recognizer để quét biển số, controller sẽ kiểm tra dữ liệu khuôn mặt và cấp access\_token đại diện cho User, tiến hành kiểm tra xe có thuộc User không, nếu không cấp được access\_token hoặc xe chưa đăng ký sẽ thông báo dữ liệu khuôn mặt và biển số không hợp lệ. Nếu dữ liệu hợp lệ tiến hành kiểm tra đăng ký chỗ và đặt chỗ nếu có thì vào bãi xe thành công, ngược lại nếu không thì không cho phép vào bãi.



*Hình 3.12. Sequence diagram Khách hàng vào bãi đỗ xe*

#### Khách hàng ra khỏi bãi đỗ xe

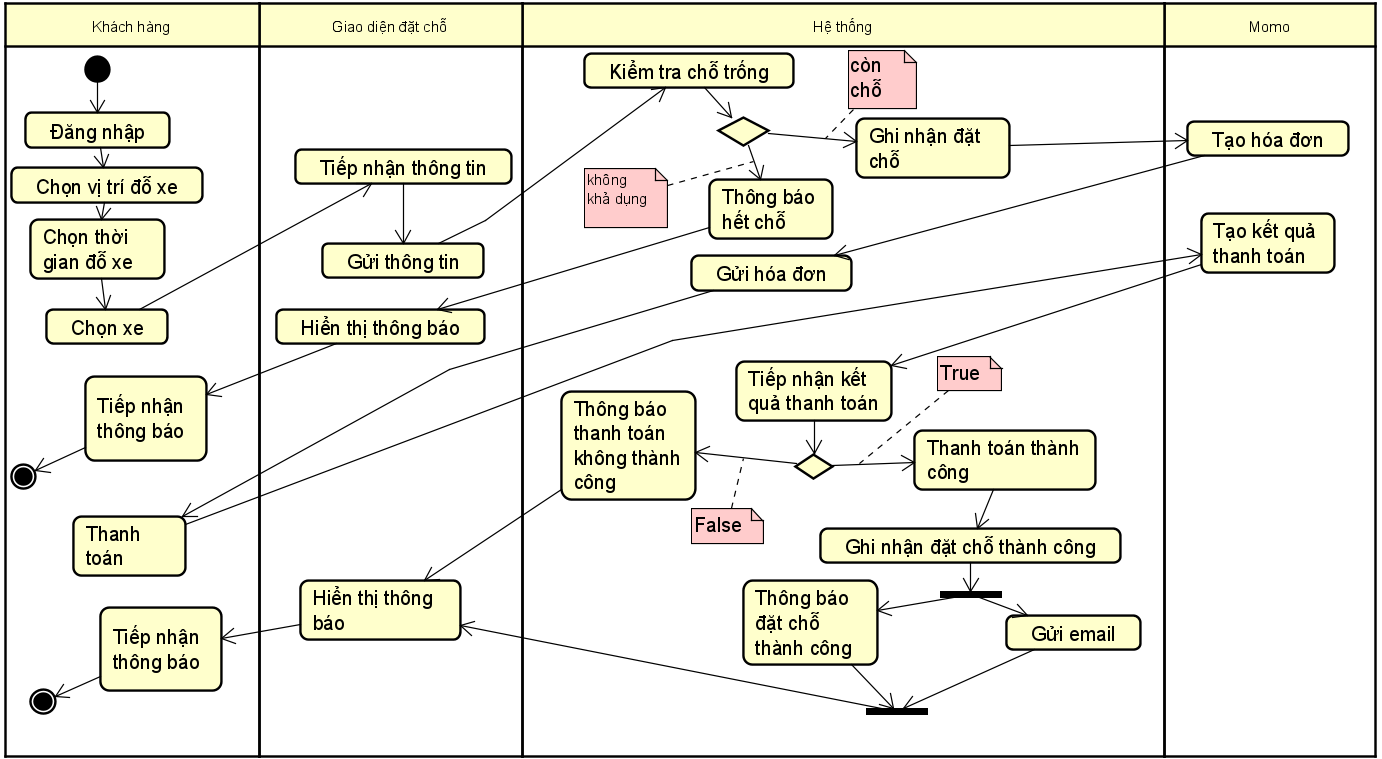
Khách hàng cung cấp hình ảnh khuôn mặt và xe khi ra bãi xe, giao diện ở client sẽ phân tích dữ liệu khuôn mặt và gọi API của Plate Recognizer để quét biển số, controller sẽ kiểm tra dữ liệu khuôn mặt và cấp access\_token đại diện cho User, tiến hành kiểm tra xe có thuộc User không, nếu không cấp được access\_token hoặc xe chưa đăng ký sẽ thông báo dữ liệu khuôn mặt và biển số không hợp lệ. Nếu dữ liệu hợp lệ, kiểm ra xe có ra muộn giờ theo giờ đã Booking hoặc ra muộn do hết hạn đăng ký không, nếu có thì yêu cầu thanh toán và sau khi thanh toán thành công sẽ được ra khỏi bãi, nếu không muộn thì hệ thống ghi nhận đã ra khỏi bãi thành công.



*Hình 3.13. Sequence diagram Khách hàng ra khỏi bãi đỗ xe*

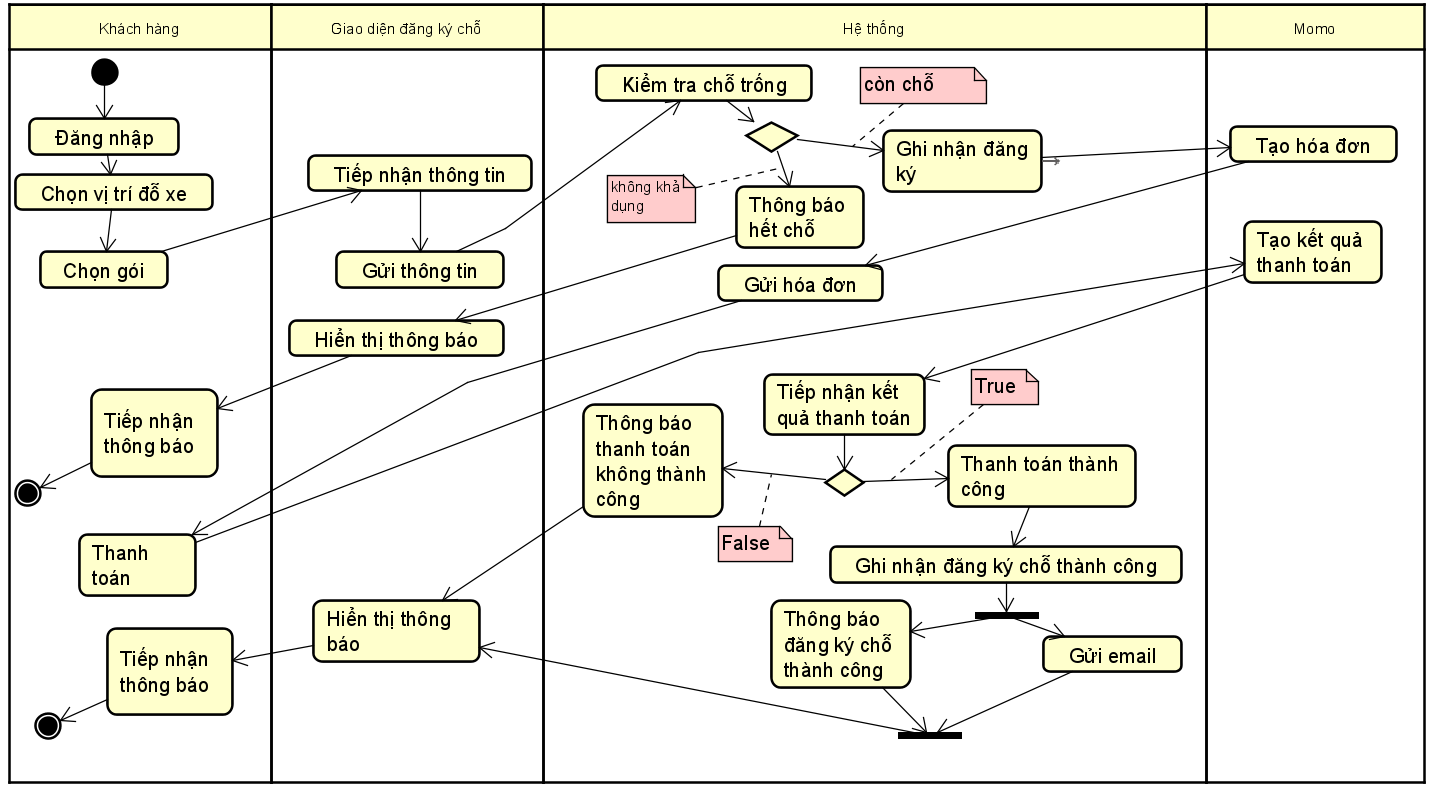
### Activity diagram

#### Khách hàng đặt chỗ đỗ xe



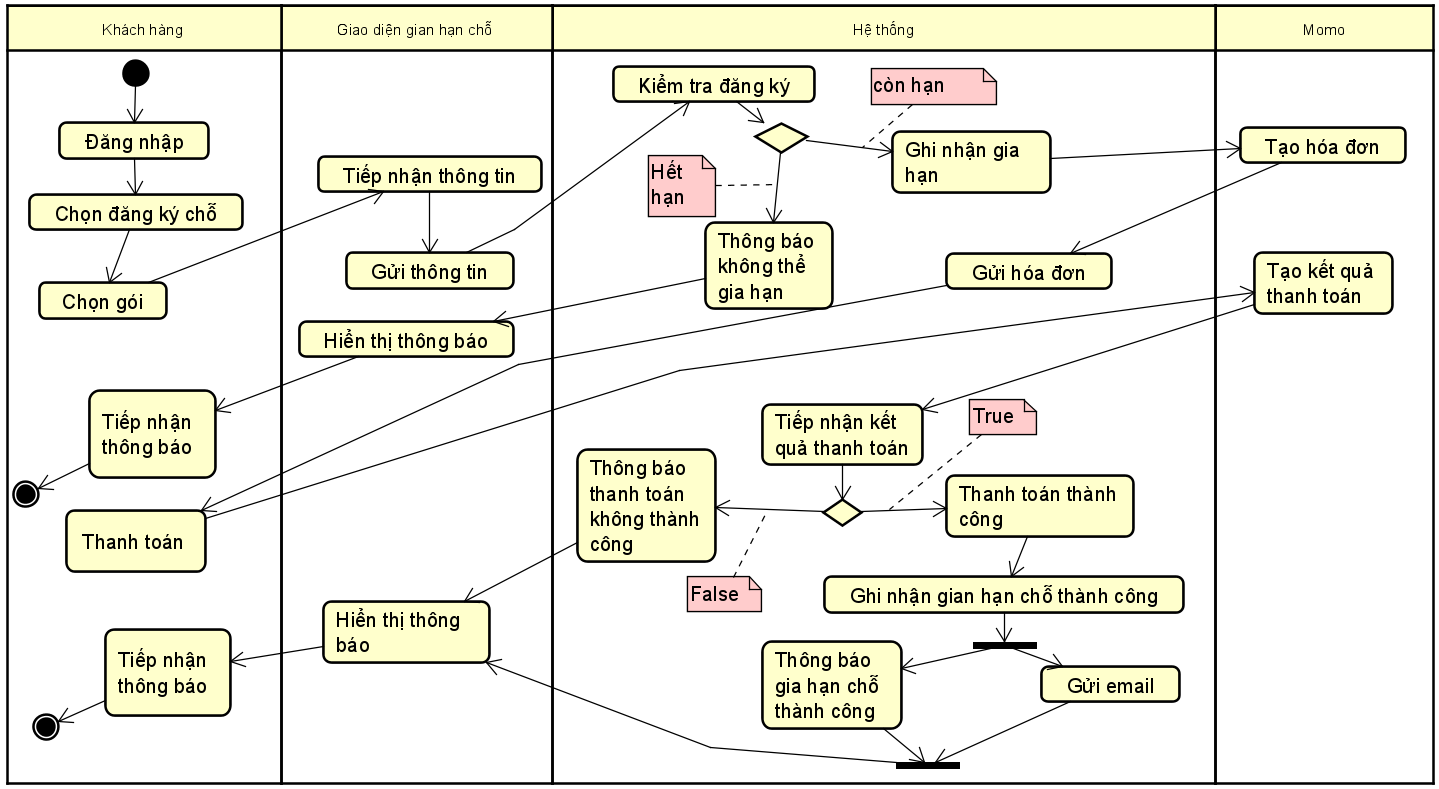
*Hình 3.14. Activity diagram khách hàng đặt chỗ*

#### Khách hàng đăng ký chỗ đỗ xe



*Hình 3.15. Activity diagram khách hàng đăng ký chỗ đỗ xe*

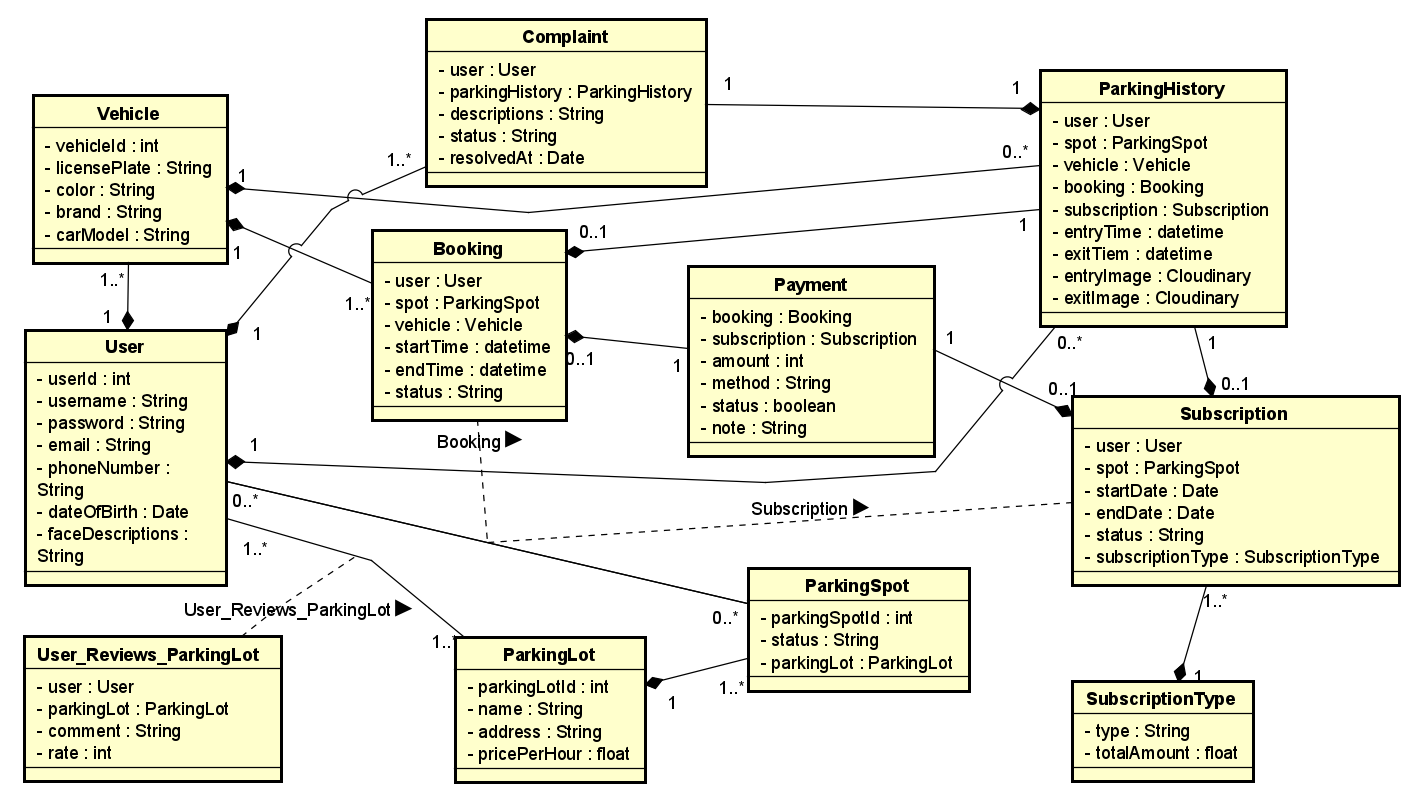
#### Khách hàng gia hạn chỗ đăng ký



*Hình 3.16. Activity diagram khách hàng gia hạn chỗ đăng ký*

## Thiết kế dữ liệu

### Class diagram

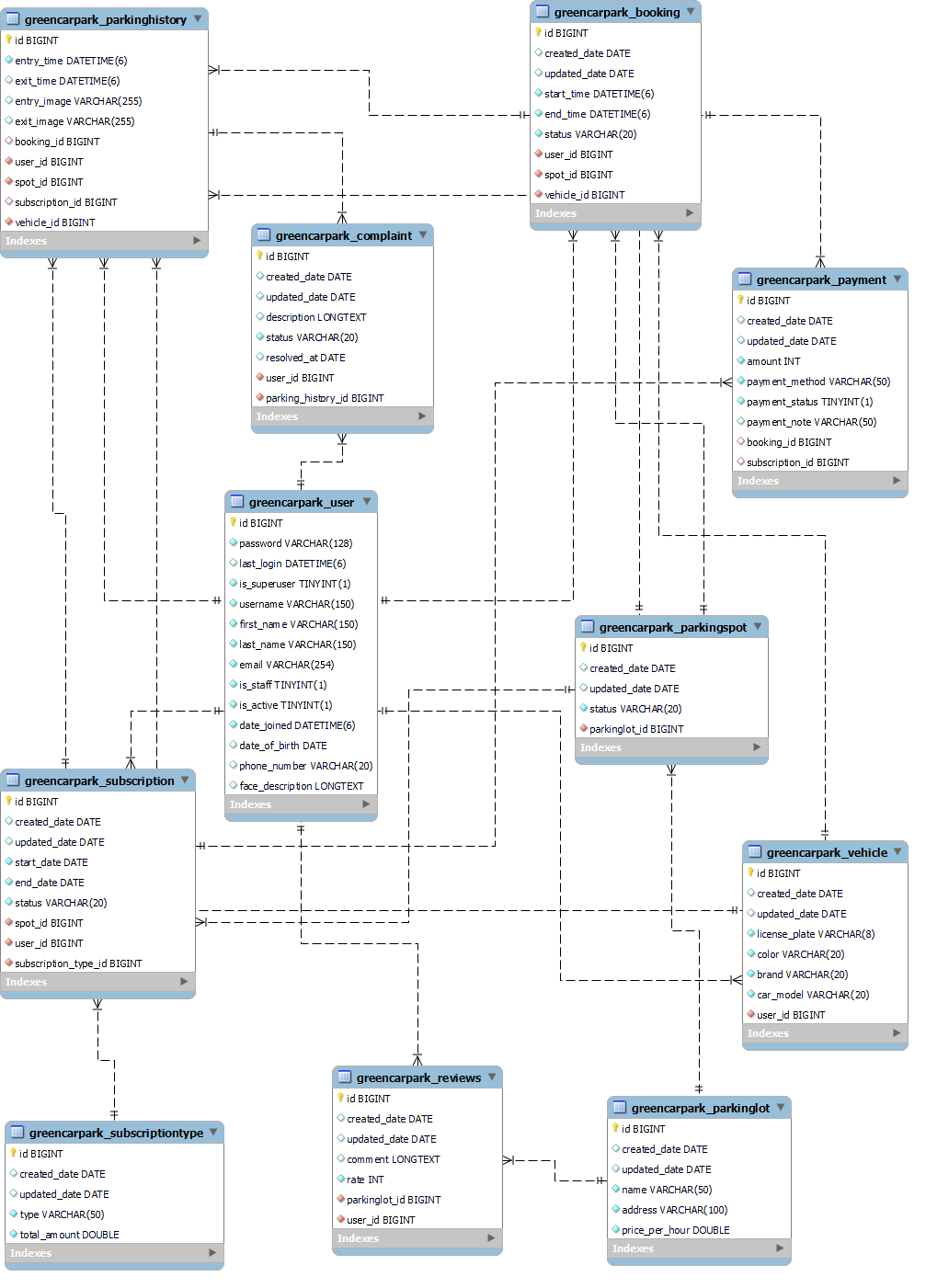
*Hình 3.17. Class digram Hệ thống quản lí bãi đỗ xe*

Phân tích, giải thích các mối quan hệ thiết lập:

* **Lớp User**: kế thừa AbstractUser của django.contrib.auth.models sử dụng phân quyền có sẵn của django, thêm vào một số thuộc tính như dateOfBirth, faceDescriptions, phoneNumber.
* **Lớp Vehicle**: lưu trữ thông tin phương tiện của User, một User có thể đăng ký nhiều phương tiện và mỗi phương tiện chỉ thuộc một User, nếu User bị xóa các phương tiện thuộc User đó cũng sẽ bị xóa theo.
* **Lớp ParkingLot**: lưu trữ thông tin bãi đỗ xe.
* **Lớp Reviews**: là lớp trung gian giữa mối quan hệ many to many của User và ParkingLot, mỗi User có thể đánh giá nhiều bãi xe và mỗi bãi xe có thể được nhiều User đánh giá, thông tin đánh giá gồm rate và comment.
* **Lớp ParkingSpot**: lưu trữ thông tin chỗ đỗ xe, mỗi ParkingSpot thuộc một ParkingLot và ParkingLot thì có nhiều ParkingSpot, nếu xóa ParkingLot sẽ xóa tất cả các ParkingSpot thuộc nó, không thể xóa các ParkingSpot đang trong trạng thái in\_use hoặc reserved.
* **Lớp Booking**: lưu trữ thông tin đặt chỗ, là lớp trung gian giữa mối quan hệ many to many của User và ParkingSpot, một User có thể đặt nhiều chỗ và mỗi chỗ có thể được nhiều User đặt, ngoài việc cung cấp giờ bắt đầu để vào bãi và giờ kết thúc để ra khỏi bãi, còn cần phải cung cấp Vehicle thuộc User đặt chỗ, nếu Vehicle bị xóa thì Booking cũng sẽ bị xóa theo.
* **Lớp SubscriptionType**: lưu trữ thông tin các gói đăng ký mua chỗ, bao gồm tên gói và tổng tiền, nếu SubscriptionType bị xóa, các Subscription thuộc nó sẽ bị xóa theo.
* **Lớp Subscription**: lưu trữ thông tin đăng ký chỗ đỗ, là lớp trung gian giữa mối quan hệ many to many của User và ParkingSpot, một User có thể đăng ký nhiều chỗ và mỗi chỗ có thể được nhiều User đăng ký, khi đăng ký chỗ cần cung cấp thông tin SubscriptionType.
* **Lớp Payment**: lưu trữ thông tin thanh toán, cần có thông tin của Subscription hoặc Booking để tạo Payment, Booking và Subscription tồn tại không đồng thời trong Payment, một trong hai sẽ có giá trị null, nếu xóa Booking hoặc Subscription thì Payment cũng sẽ bị xóa theo.
* **Lớp ParkingHistory**: lưu trữ lịch sử vào và ra của bãi xe, Vehicle thuộc User có thể ra vào bãi nhiều lần và mỗi lịch sử chỉ lưu một Vehicle và một User, đồng thời lưu trữ User sử dụng Booking hay Subscription để vào bãi, một trong hai giá trị là Booking hoặc Subscription sẽ có giá trị null, nếu xóa User, Vehicle, Subscription, Booking thuộc ParkingHistory thì ParkingHistory cũng sẽ bị xóa theo.
* **Lớp Complaint**: lữu trữ thông tin khiếu nại của khách hàng, một Complaint khiếu nại từ một ParkingHistory và của User nào đang khiếu nại, nếu ParkingHistory hoặc User bị xóa thì Complaint cũng sẽ bị xóa.

### Chi tiết thiết kế cơ sở dữ liệu

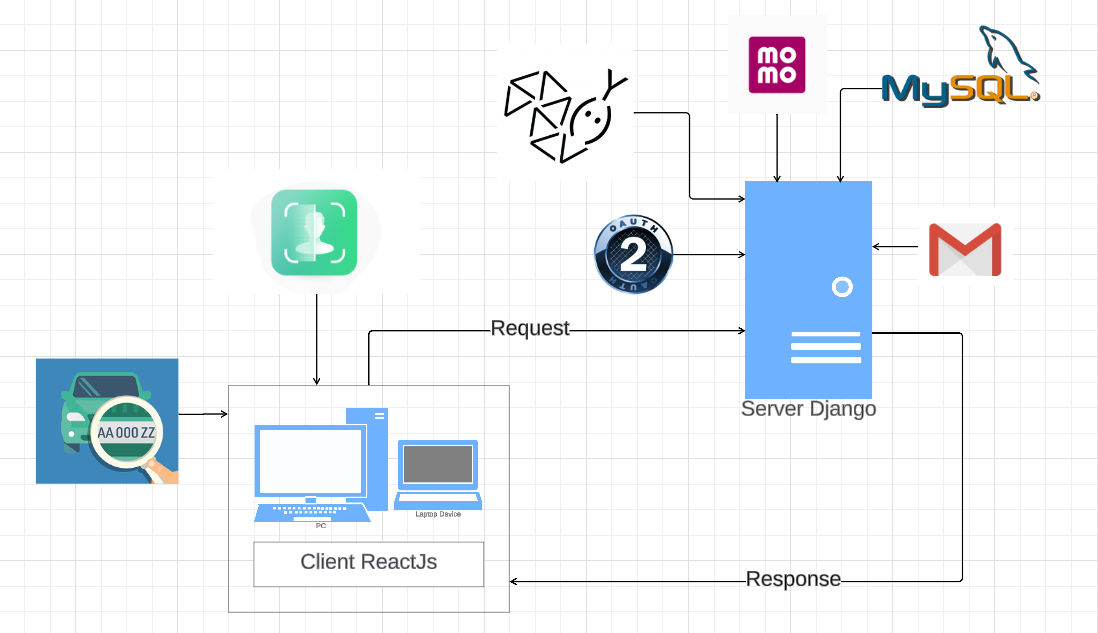
Lược đồ ERD:



*Hình 3.18. Lược đồ ERD*

## Kiến trúc hệ thống

Client – Server: Khách hàng đăng ký và cung cấp dữ liệu khuôn mặt, sau đó đăng nhập bằng username và password, thêm thông tin xe. Tiến hành đặt chỗ hoặc đăng ký chỗ còn trống. Hệ thống ghi nhận vào và ra bãi đỗ xe bằng nhận dạng khuôn mặt và quét biển số xe. Đánh giá dịch vụ sau khi sử dụng.

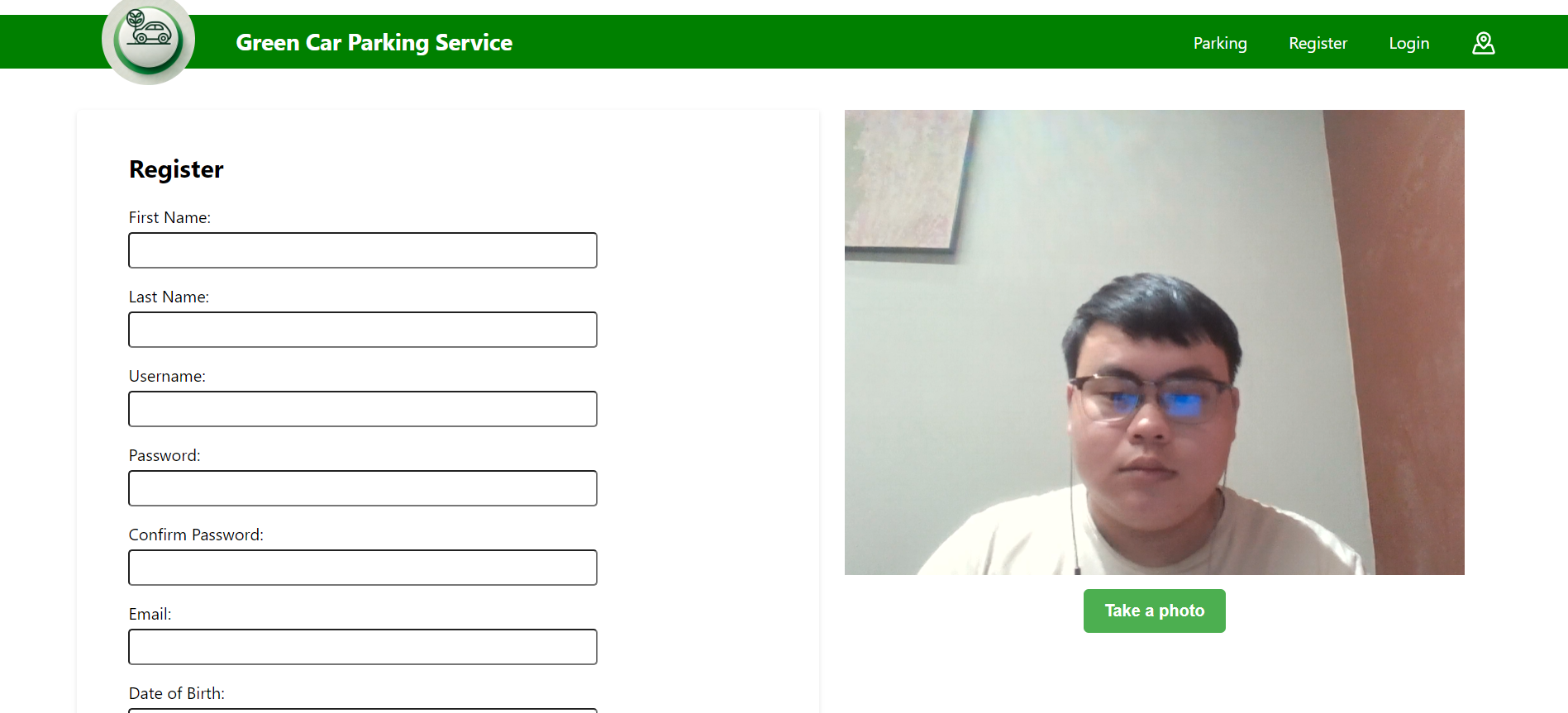


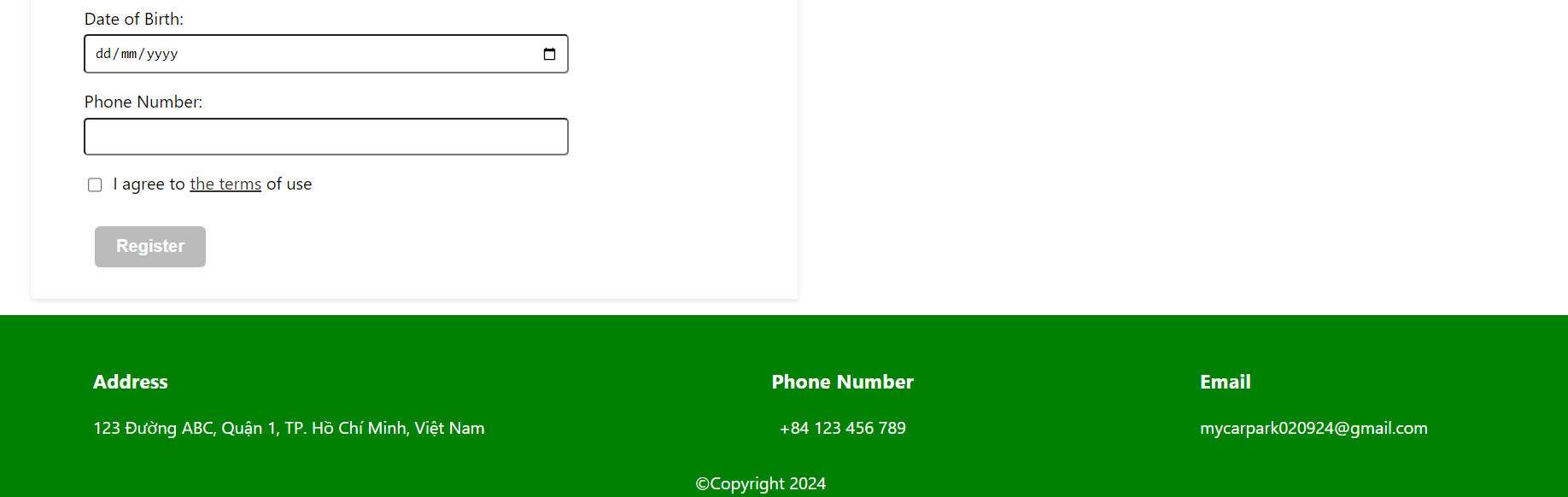
*Hình 3.19. Kiến trúc hệ thống*

## Sản phẩm

### Khách hàng đăng ký thành viên, và đăng nhập

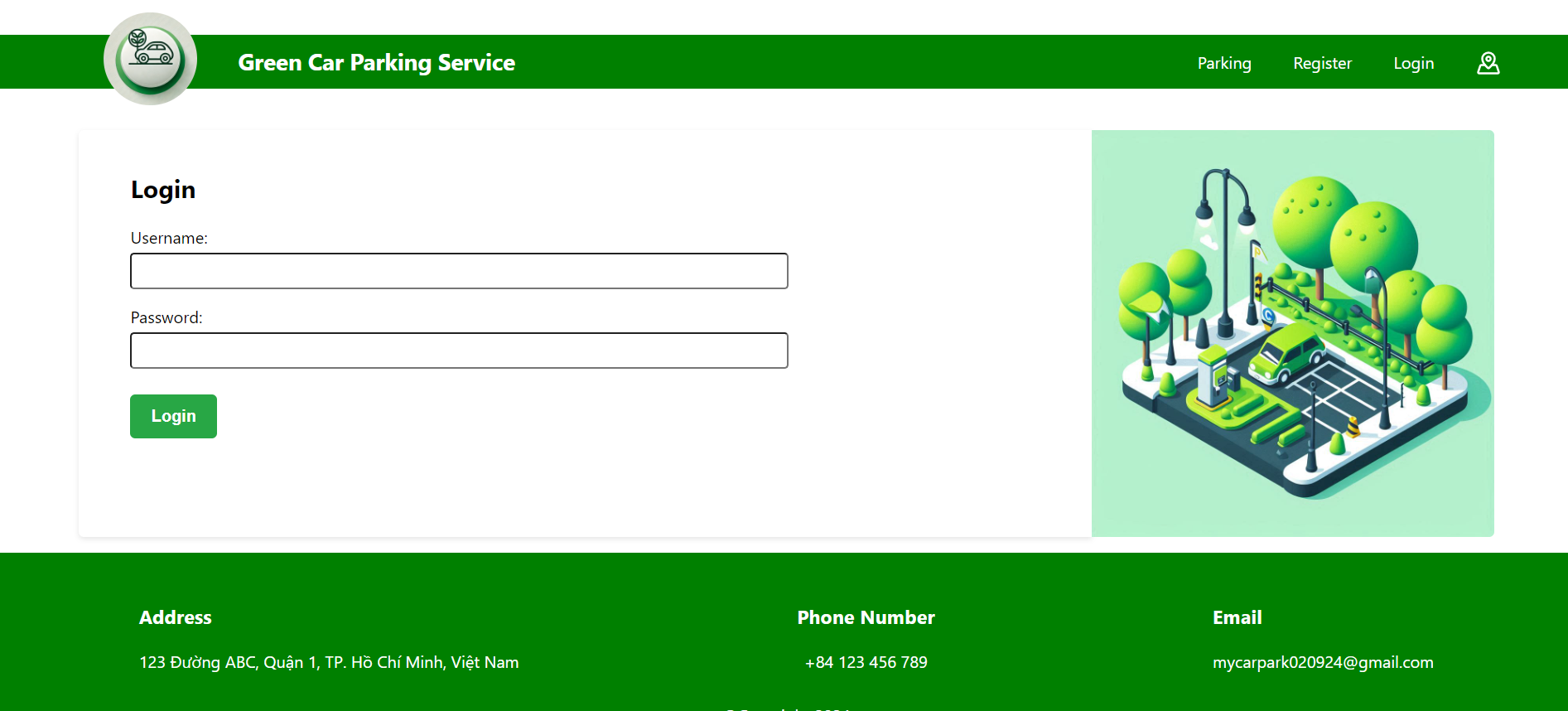
***Đăng ký:***





*Hình 3.20. Giao diện đăng ký*

***Đăng nhập:***



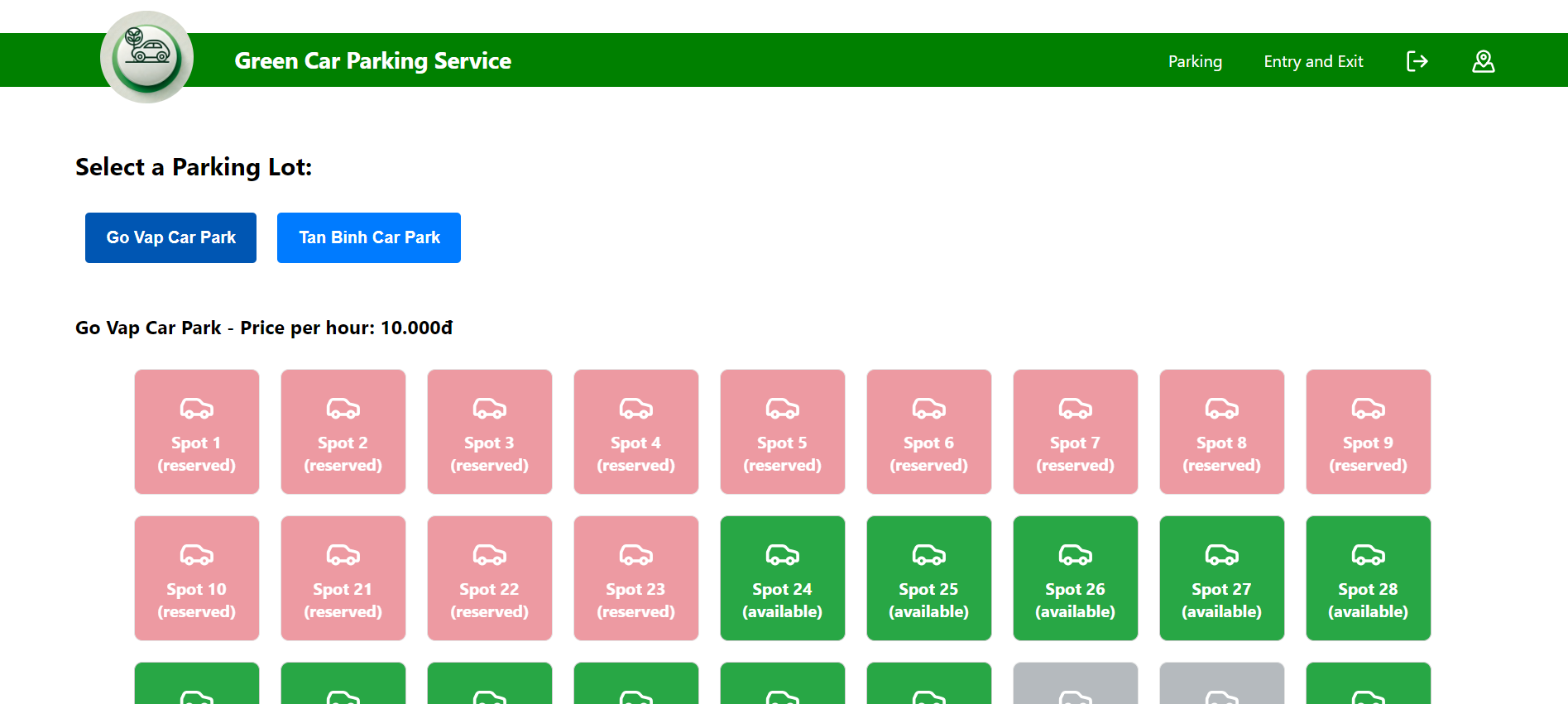
*Hình 3.21. Giao diện đăng nhập*

***Khách hàng đăng nhập:***



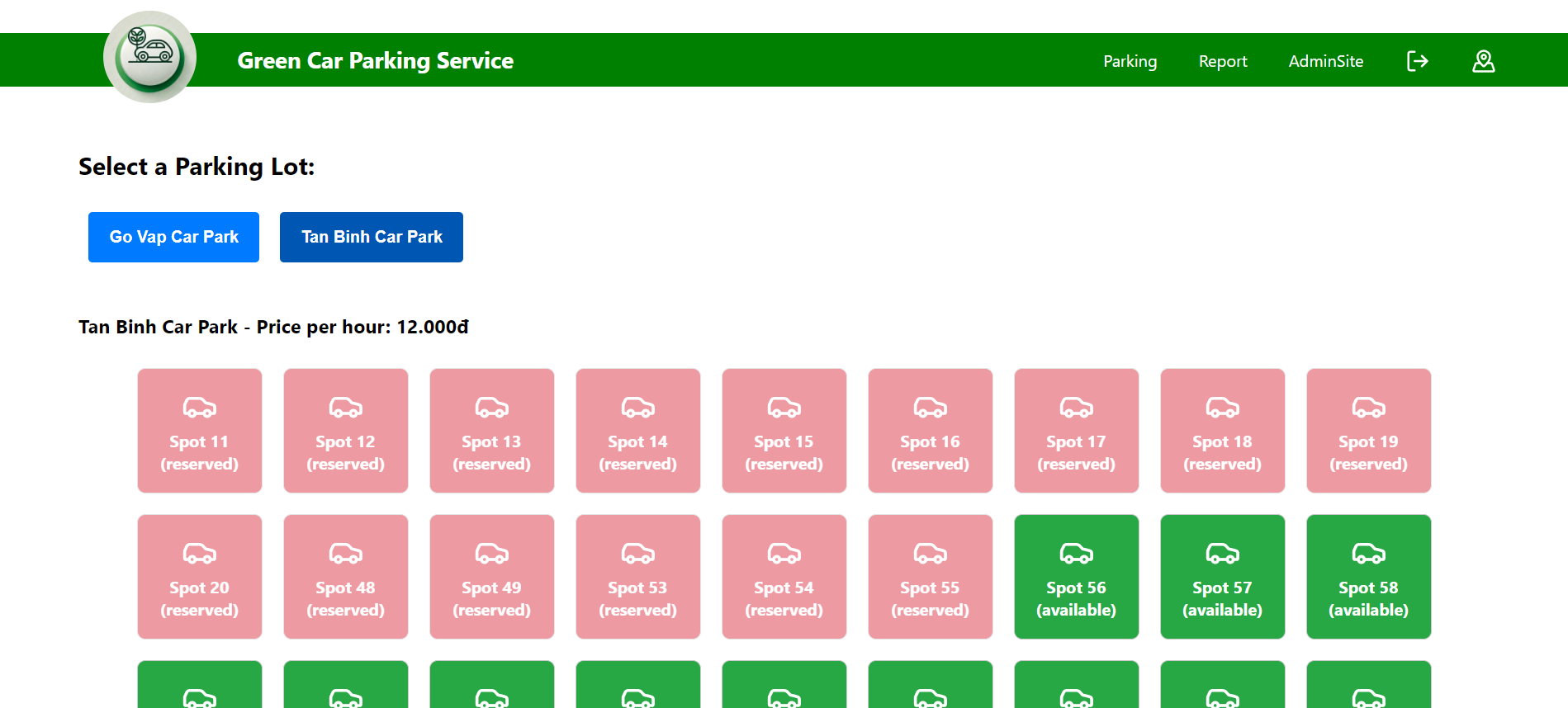
*Hình 3.22. Giao diện khách hàng đăng nhập*

***Nhân viên đăng nhập:***



*Hình 3.23. Giao diện nhân viên đăng nhập*

***Admin đăng nhập:***



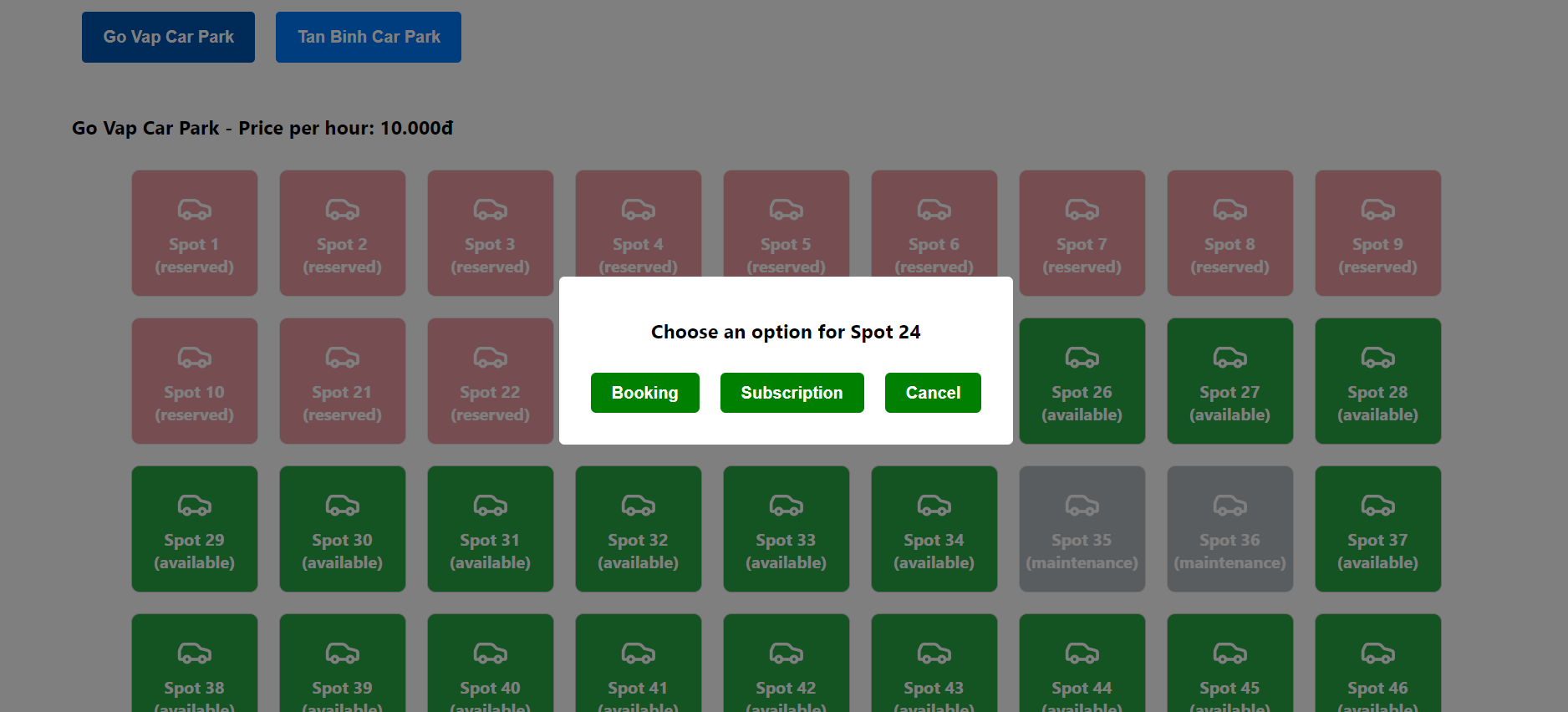
*Hình 3.24. Giao diện Admin đăng nhập*

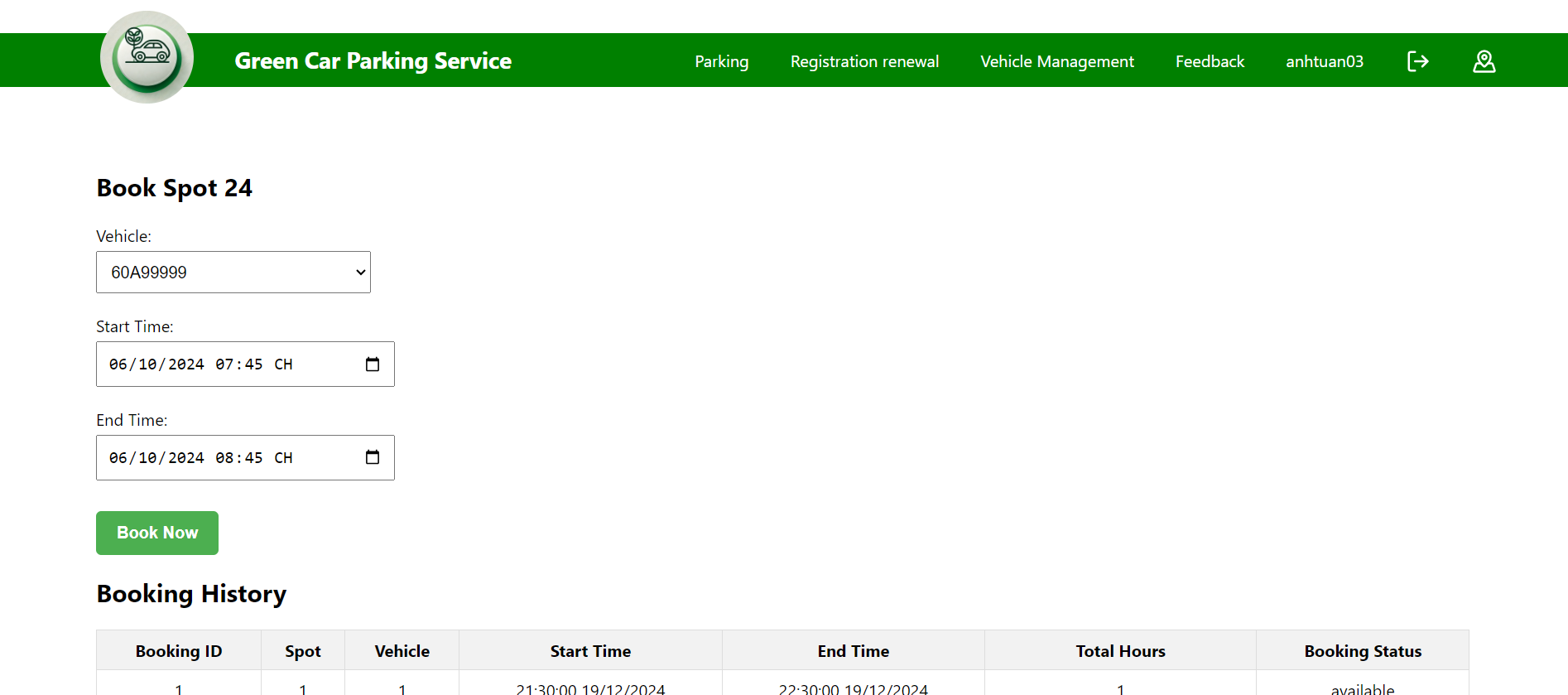
### Xem trạng thái chỗ đỗ xe



*Hình 3.25. Giao diện xem trạng thái chỗ đỗ xe*

### Khách hàng đặt chỗ

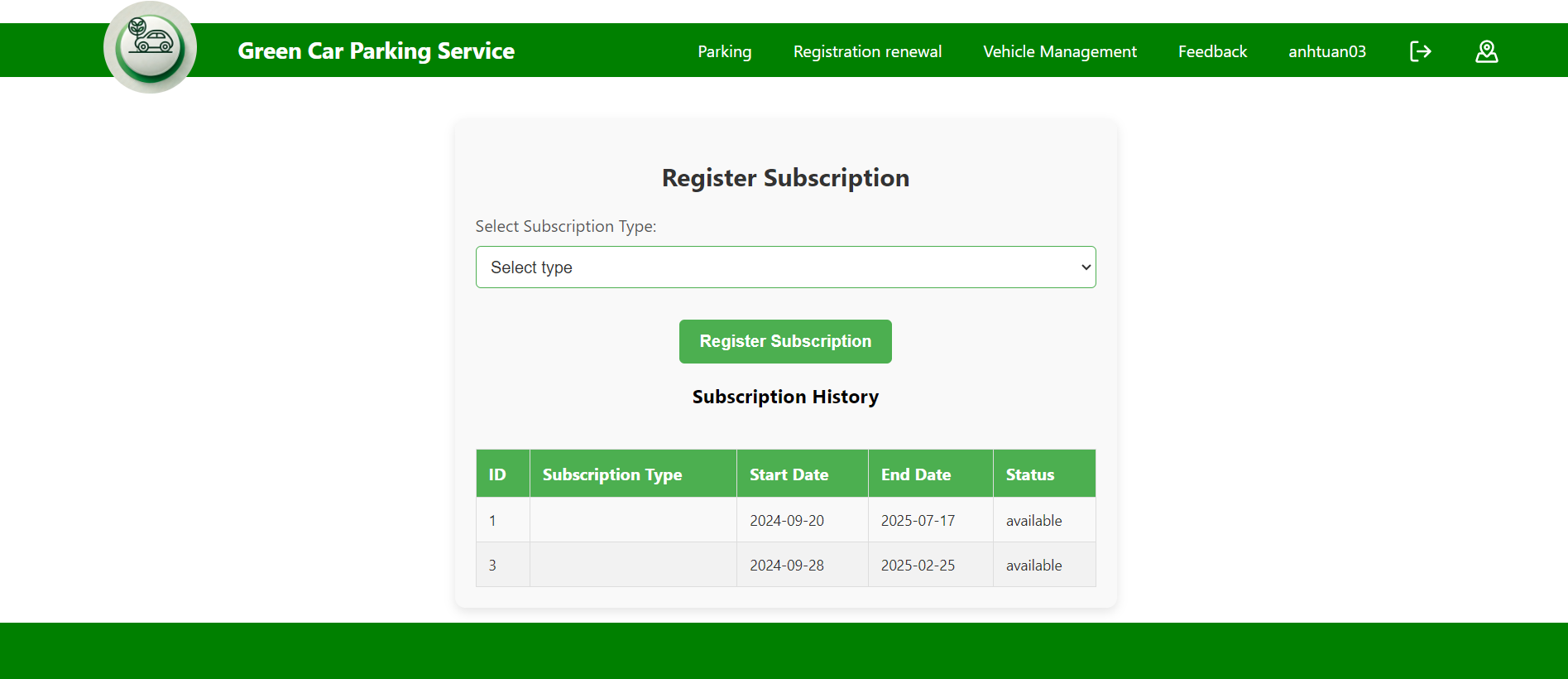




*Hình 3.26. Giao diện đặt chỗ*

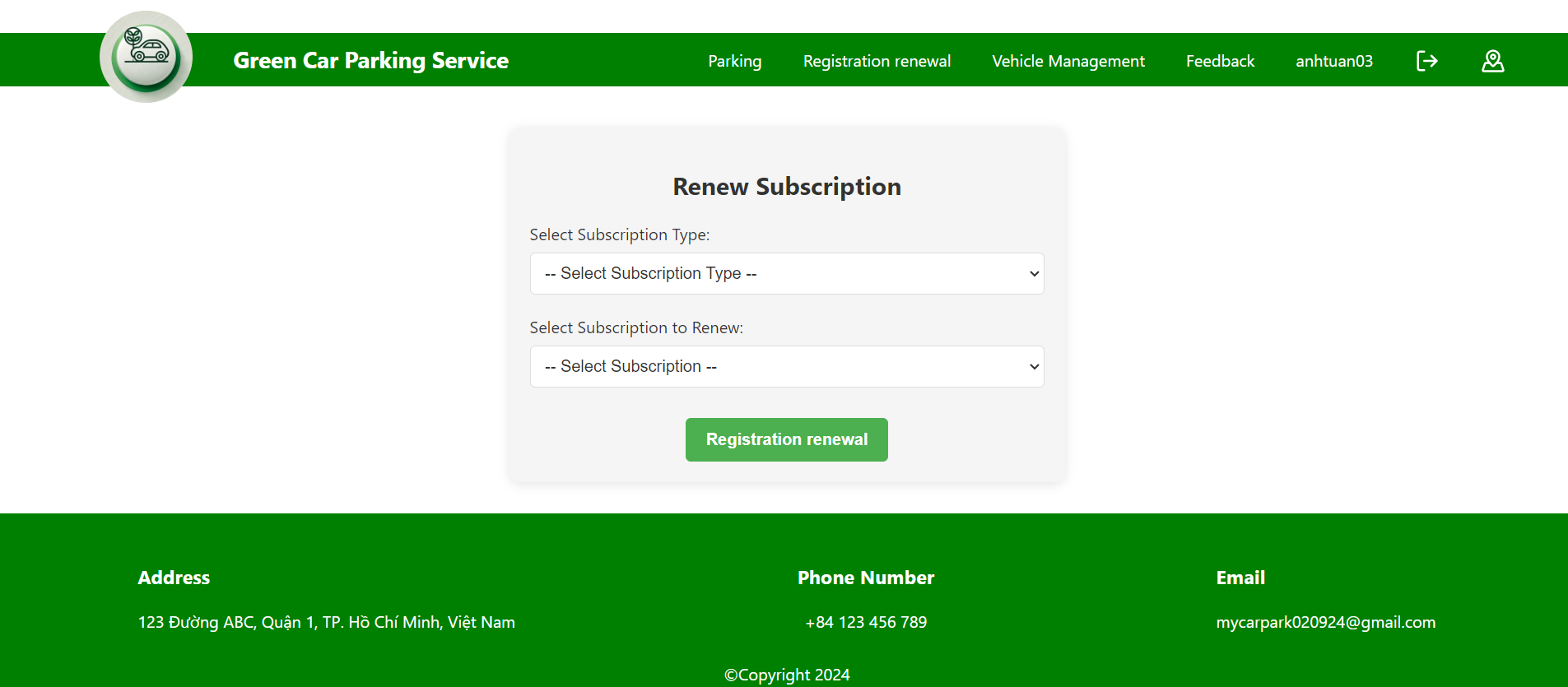
### Khách hàng đăng ký chỗ, gia hạn chỗ

***Đăng ký chỗ:***



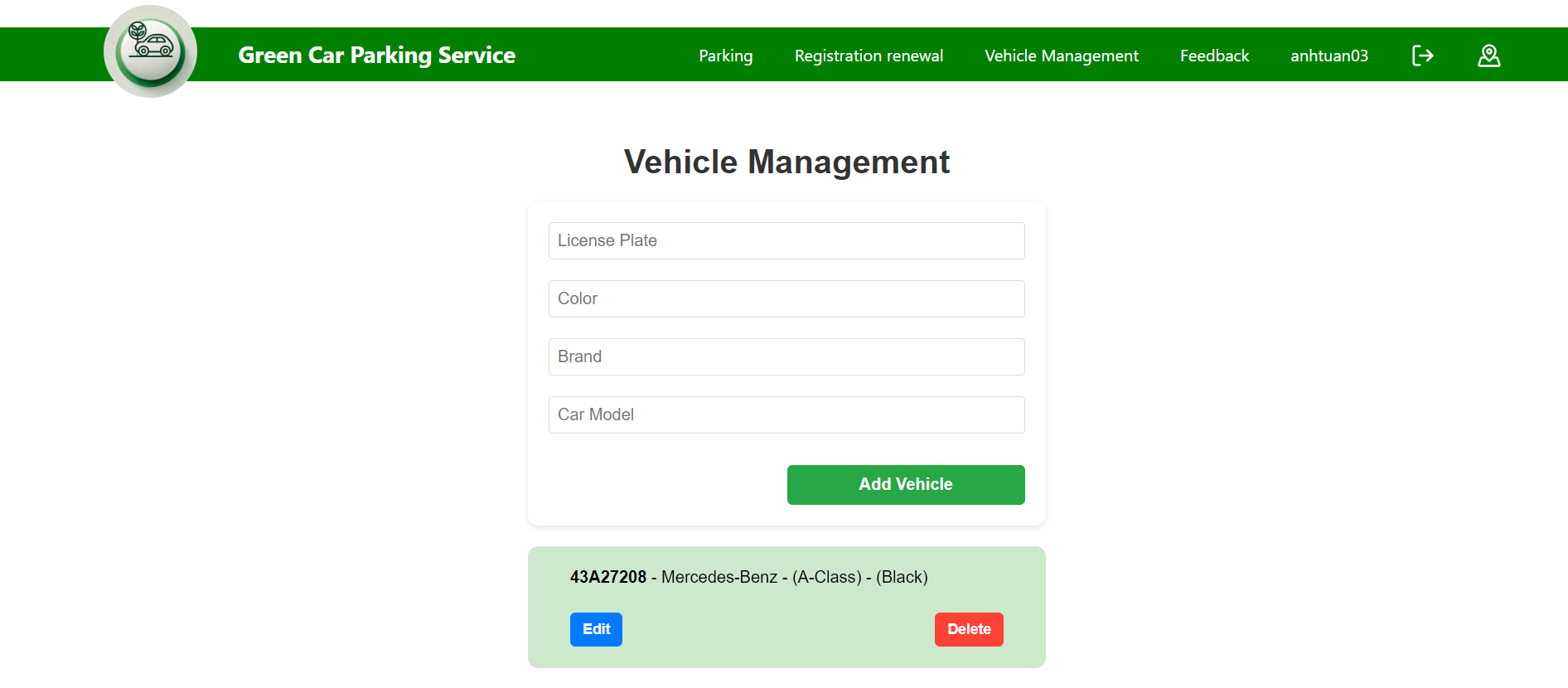
*Hình 3.27, Giao diện đăng ký chỗ*

***Gia hạn chỗ:***



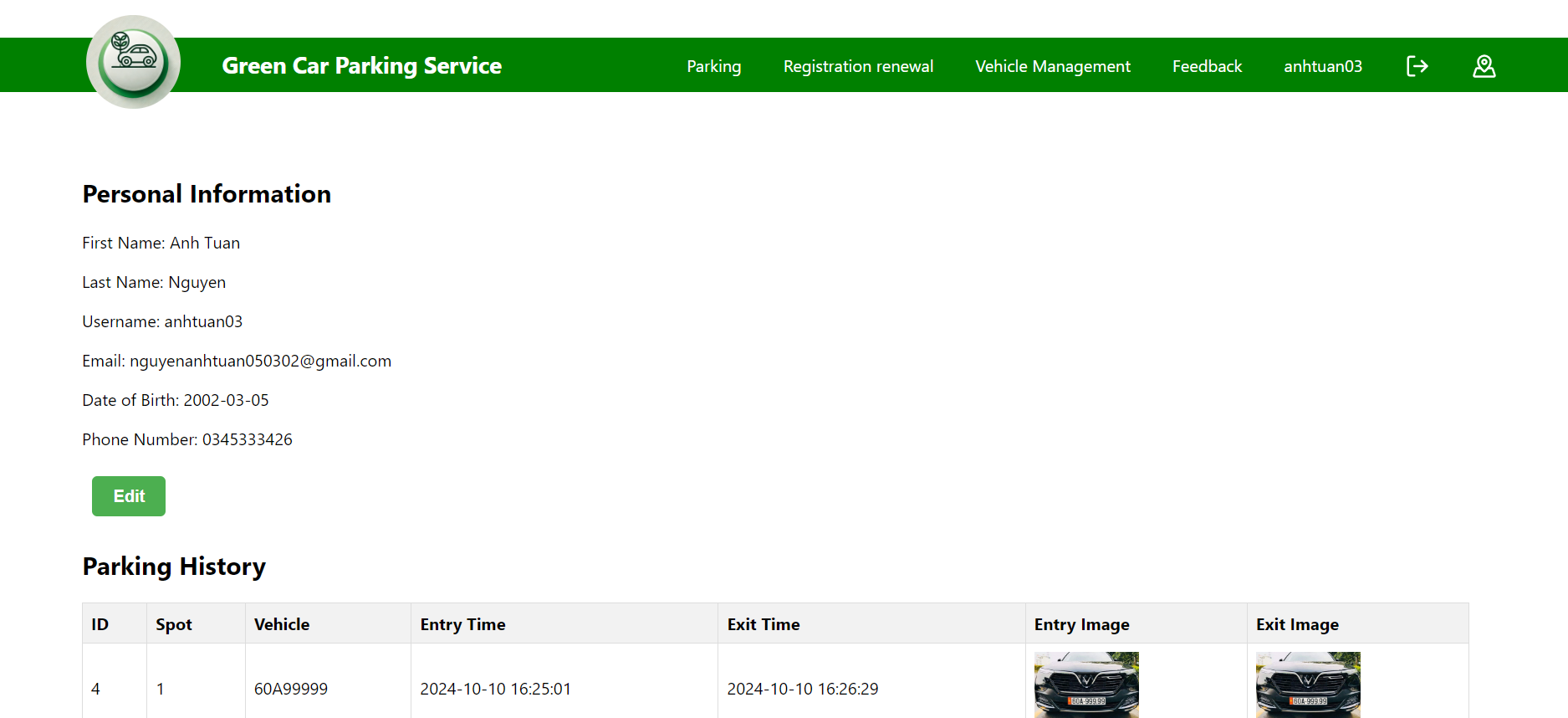
*Hình 3.28. Giao diện gia hạn chỗ*

### Khách hàng quản lý phương tiện



*Hình 3.29. Giao diện khách hàng quản lý phương tiện*

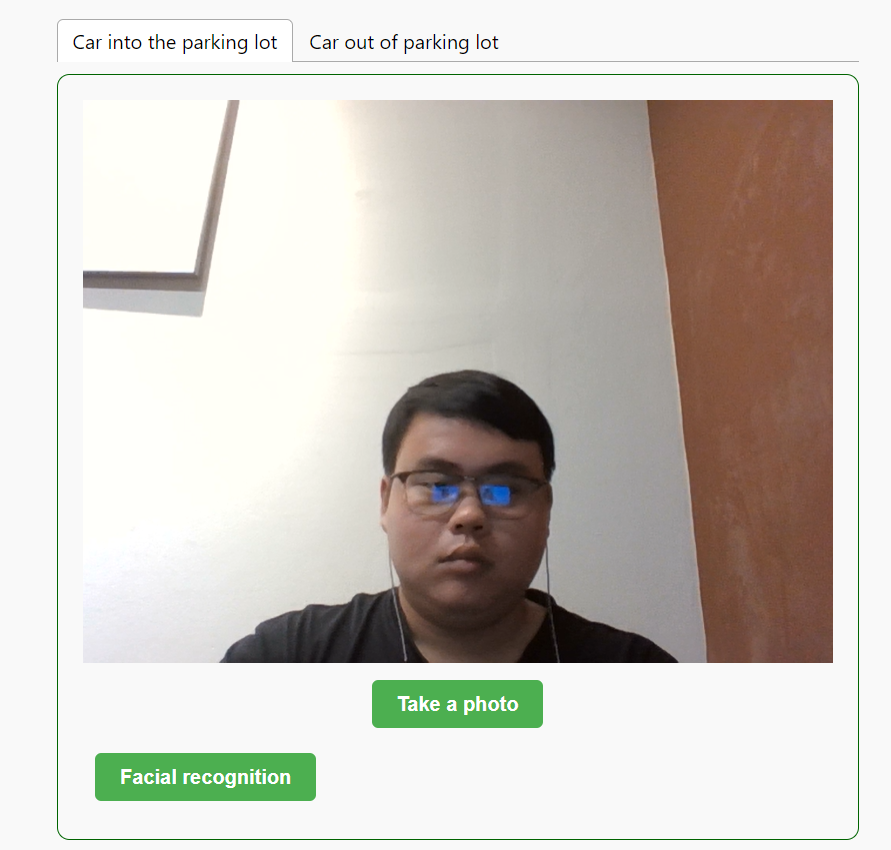
### Khách hàng quản lí thông tin cá nhân

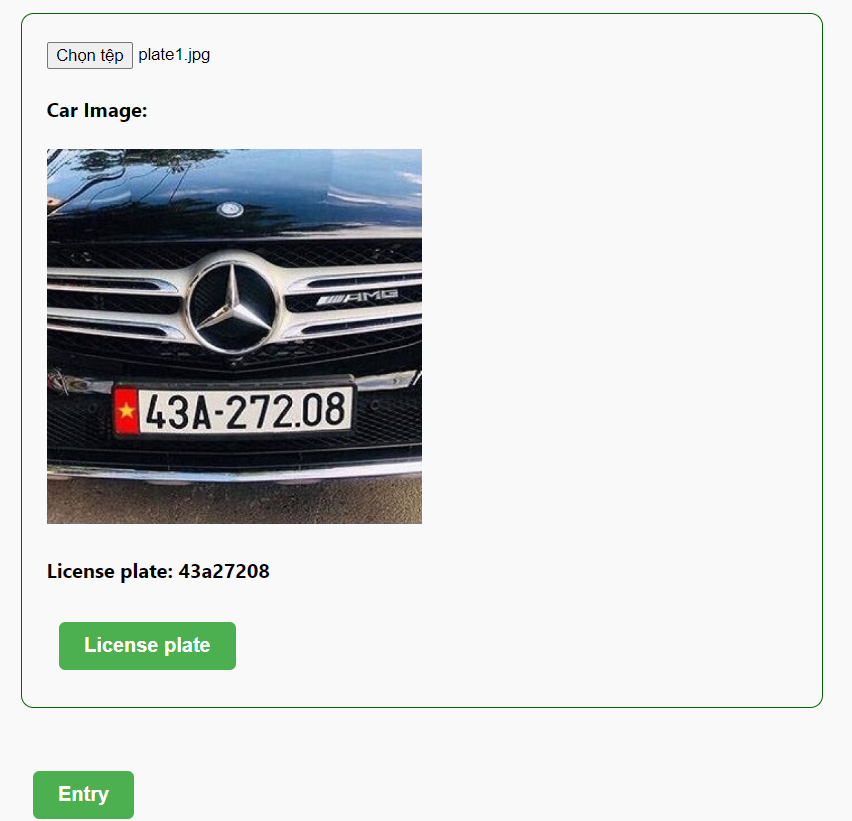


*Hình 3.30. Giao diện khách hàng quản lí thông tin cá nhân*

### Nhân viên hỗ trợ khách hàng vào và ra bãi xe

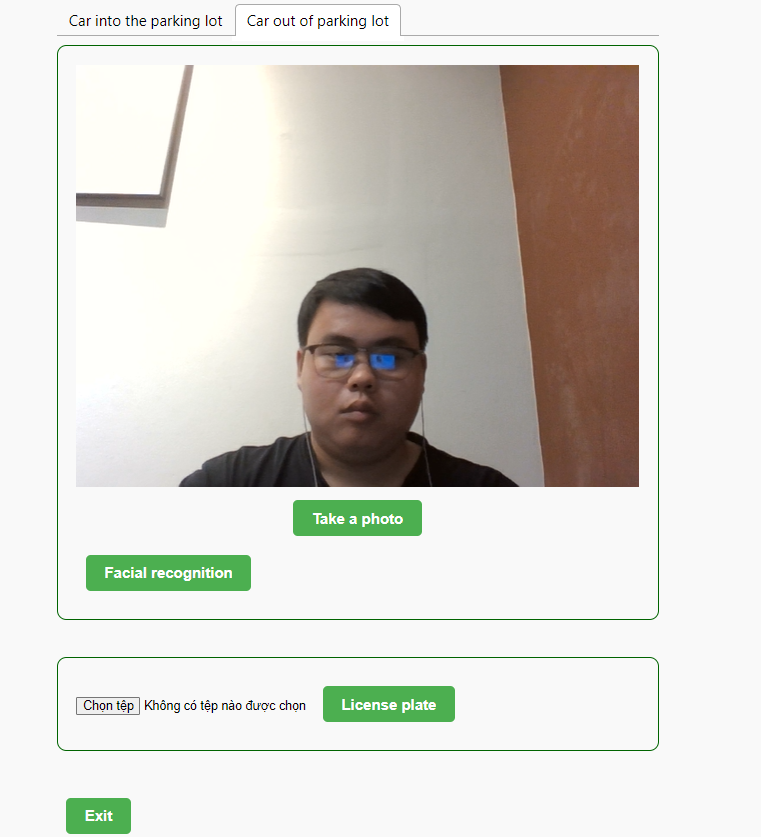
***Vào bãi:***





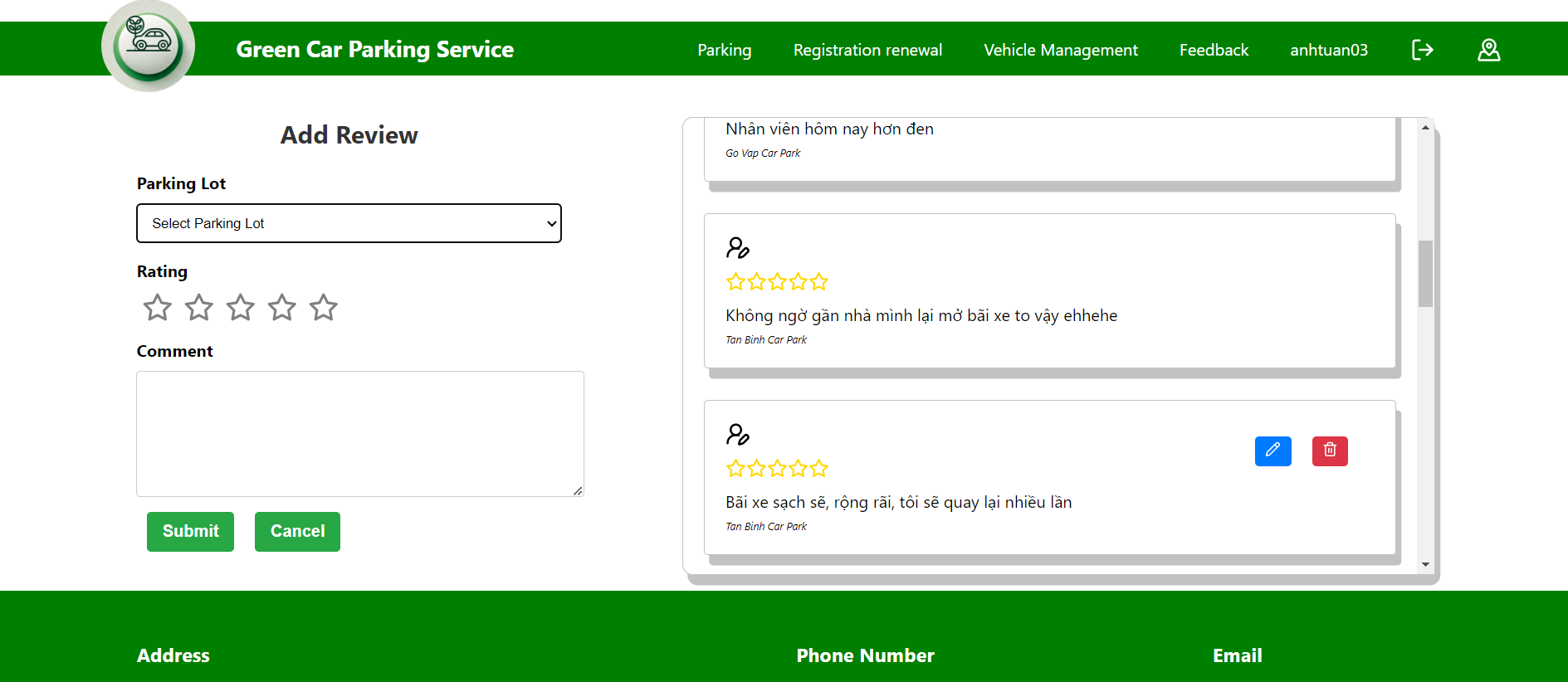
*Hình 3.31. Giao diện vào bãi đỗ xe.*

***Ra bãi:***



*Hình 3.32. Giao diện ra khỏi bãi đỗ xe*

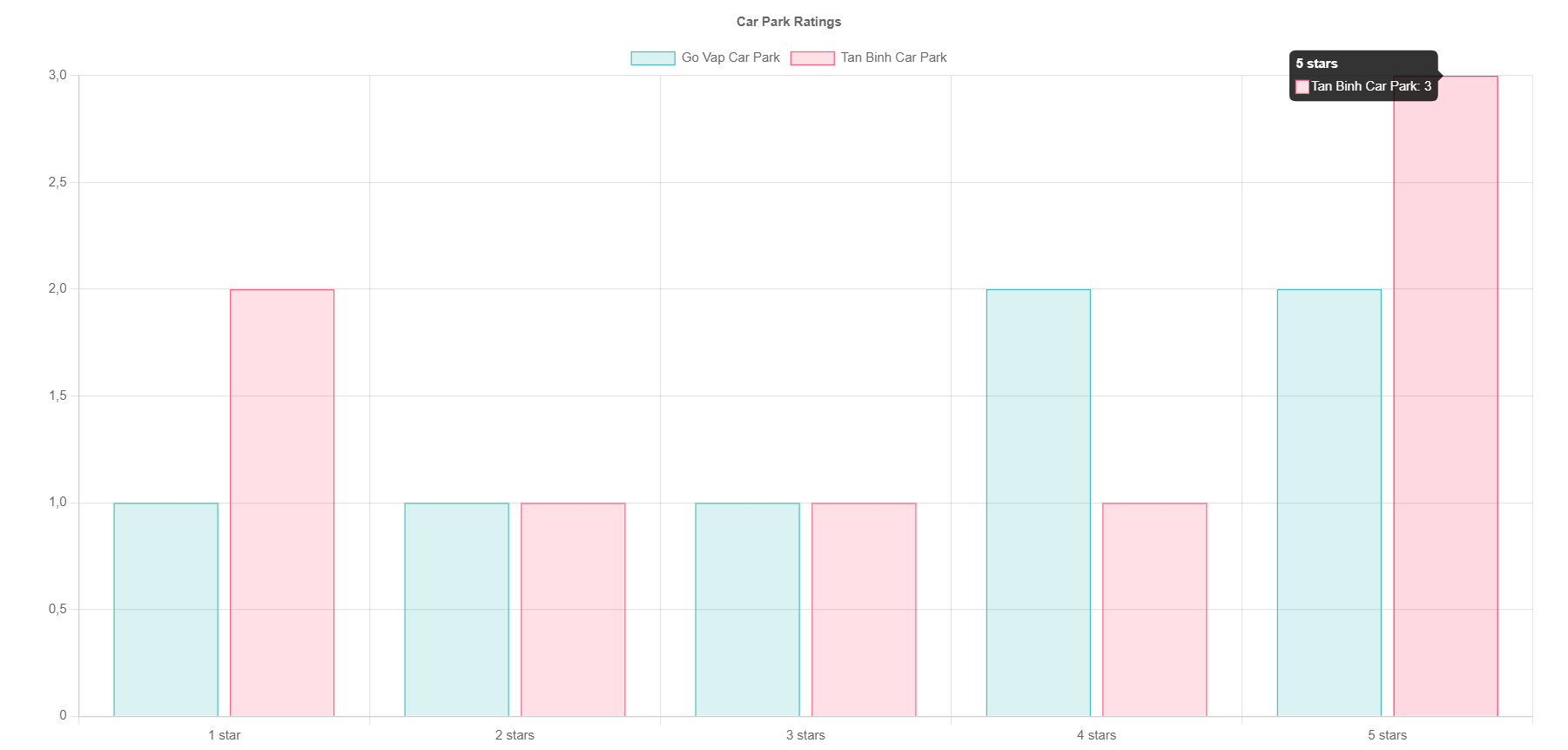
### Khách hàng đánh giá bãi xe



*Hình 3.33. Giao diện đánh giá bãi xe*

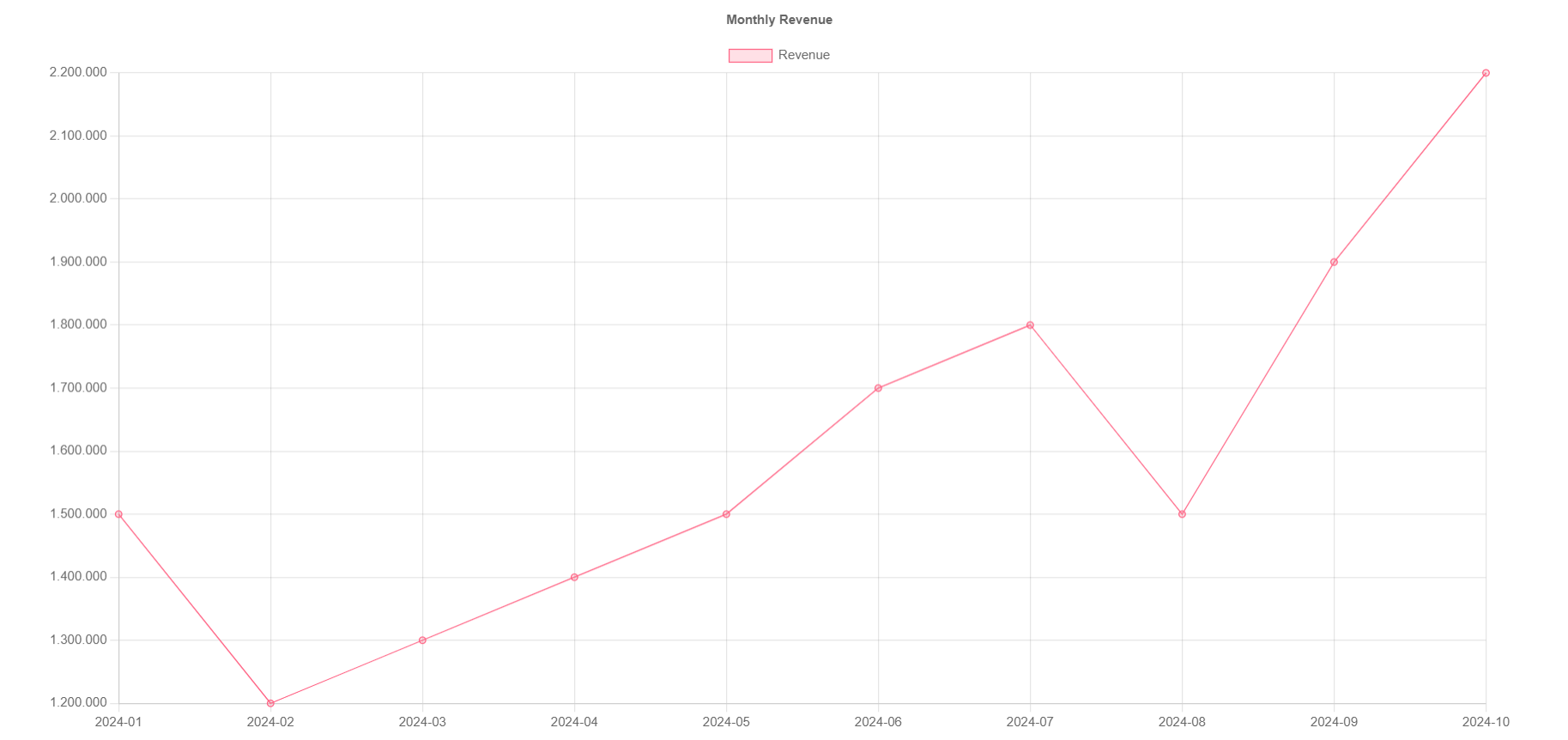
### Admin tạo báo cáo

***Báo cáo đánh giá:***



*Hình 3.34. Báo cáo đánh giá của người dùng*

***Báo cáo doanh thu theo tháng:***



*Hình 3.35. Báo cáo doanh thu theo tháng*

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Phát triển hệ thống bãi đỗ ô tô tích hợp các kỹ thuật trí tuệ nhân tạo nhằm cung cấp một giải pháp đỗ xe và quản lý bãi đỗ xe ô tô. Ngoài việc đã thành công trong việc tự động hóa quy trình ra vào bãi đỗ xe, tăng cường tính bảo mật, an toàn và tiện lợi hơn cho người dùng, hệ thống vẫn còn những khiếm khuyết về chức năng, hiệu năng và chưa đa dạng phương pháp xác thực thông tin.

### Ưu điểm

* Tự động hóa quy trình ra vào bãi đỗ xe thành công thông qua việc tích hợp được hai công nghệ trí tuệ nhân tạo là Face-api.js để nhân diện khuôn mặt và Plate Recognizer để bóc tách thông tin xe.
* Tăng cường an ninh, chỉ những xe và người được hệ thống xác thực mới có thể ra vào bãi đỗ xe, giảm thiểu tình trạng gian lận, bảo vệ tốt hơn xe của khách hàng.
* Nâng cao trải nghiệm, người dùng dễ dàng theo dõi trạng thái chỗ đỗ và lịch sử ra vào bãi, quản lí các phương tiện và thông tin cá nhân, đánh giá và góp ý dịch vụ bãi đỗ xe nếu muốn.
* Dữ liệu trong các báo cáo hữu ích trong việc cải thiện dịch vụ và theo dõi hoạt động kinh doanh của bãi đỗ xe ô tô.

### Khuyết điểm

* Phụ thuộc vào chất lượng hình ảnh, nếu hình ảnh chụp không rõ khuôn mặt và xe hoặc hình chụp trong các điều kiện ánh sáng không thuận lợi thì rất khó để nhận diện chính xác.
* Chưa đa dạng các giải pháp ra vào bãi đỗ xe, nếu người dùng không muốn cung cấp dữ liệu khuôn mặt hoặc không muốn lộ mặt khi ra vào bãi đỗ xe thì không còn cách khác để xác thực người dùng.
* Chưa phát triển thành công chức năng khiếu nại và giải quyết khiếu nại, khi có khiếu nại về mất mát, hư tổn xảy ra hệ thống chưa đưa ra được giải pháp tối ưu làm hài lòng khách hàng.
* Chưa phát triển chức năng nhân viên báo cáo và quản trị viên xác nhận các chỗ đỗ xe cần bảo trì.

## Hướng phát triển

* Tích hợp thêm các công nghệ trí tuệ nhân tạo khác nhằm dự đoán du cầu đỗ xe theo thời gian và xu hướng, từ đó chủ động trong việc quảng cáo, điều chỉnh giá và phân bổ nhân viên hợp lý hơn.
* Cải thiện khả năng nhận diện, sử dụng các công nghệ nhận diện hiện đại và tiên tiến hơn, nhằm tăng cường bảo mật và hoạt động tốt hơn trong môi trường ánh sáng cho ra hình ảnh không tốt.
* Phát triển việc tích hợp với các hệ thống giao thông thông minh nhằm cung cấp dữ liệu điều hướng và điều chỉnh lưu lượng giao thông xung quanh các khu vực bãi đỗ xe.
* Phát triển chức năng nhân viên thông báo và quản trị viên xác nhận các chỗ đỗ cần bảo trì.
* Phát triển chức năng khiếu nại của khách hàng khi xảy ra mất mát, hư tổn tại bãi đỗ xe, hệ thống ghi nhận quản trị viên giải quyết khiếu nại và tạo các báo cáo về khiếu nại.

TÀI LIỆU THAM KHẢO

[1] Django Software Foundation, “Django overview”, 2024. [Trực tuyến]. Địa chỉ: [Django overview](https://www.djangoproject.com/start/overview/). [Truy cập 13/10/2024].

[2] Django Software Foundation, “Documentation version: 5.1”, 2024. [Trực tuyến]. Địa chỉ: [Writing your first Django app](https://docs.djangoproject.com/en/5.1/intro/tutorial01/). [Truy cập 13/10/2024].

[3] Django REST framework, “Django REST framework”, 2024. [Trực tuyến]. Địa chỉ: [Django REST framework](https://www.django-rest-framework.org/). [Truy cập 13/10/2024].

[4] Meta Platforms, “Tutorial: Intro to React”, 2024. [Trực tuyến]. Địa chỉ: [Tutorial: Intro to React](https://legacy.reactjs.org/tutorial/tutorial.html). [Truy cập 13/10/2024].

[5] Refsnes Data, “React Tutorial”, 2024. [Trực tuyến]. Địa chỉ: [React Tutorial](https://www.w3schools.com/react/). [Truy cập 13/10/2024].

[6] Vincent Mühler, “face-api.js”, 2024. [Trực tuyến]. Địa chỉ: [face-api.js](https://justadudewhohacks.github.io/face-api.js/docs/index.html). [Truy cập 13/10/2024].

[7] ParkPow, “Plate Recognizer Snapshot”, 2024. [Trực tuyến]. Địa chỉ: [Plate Recognizer Snapshot](https://platerecognizer.com/snapshot/). [Truy cập 13/10/2024].

PHỤ LỤC