

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
Trường công nghệ thông tin và truyền thông



BÁO CÁO BÀI TẬP LỚN OOP

Đề tài: QUẢN LÝ BÃI ĐỖ XE

Giảng viên hướng dẫn: Lê Đức Hậu

Nhóm thực hiện: 10

Lớp: 151964

Năm học 2024 – 2025

Contents

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI	1
I. Giới thiệu thành viên	3
II. Giới thiệu bài toán	3
1. Vấn đề thực tế	3
2. Giải pháp	3
III. Phân tích và thiết kế hệ thống	3
1. Đối tượng sử dụng	3
2. Biểu đồ User case	4
3. ERD	6
4. Class Diagram	7
IV. Triển khai hệ thống	10
1. Công nghệ sử dụng	10
2. Mô hình hệ thống	11
V. Chạy thử	11
VI. Đánh giá và kết luận	15

I. Giới thiệu thành viên

Thành Viên	Mã số sinh viên	Đóng góp
Phạm Văn Đồng	20225702	24%
Phạm Đức Long	20225737	24%
Phan Đức Duy	20225831	17.3%
Nguyễn Bá Hoàng	20225844	17.3%
Phạm Quốc Minh	20225743	17.3%

II. Giới thiệu bài toán

1. Vấn đề thực tế

Nhận thấy mô hình gửi xe truyền thống còn nhiều khuyết điểm như khó quản lý những xe nào đã ra vào bãi, số lượng xe đang ở trong bãi và tình trạng của bãi gửi xe. Cùng với đó là nhu cầu gửi xe ngày càng tăng cao do đô thị hóa và gia tăng dân số tại các đô thị lớn. Hệ quả là nhiều bãi đỗ xe rơi vào tình trạng mất cấp xe hay quá tải.

2. Giải pháp

Do đó bọn em đã xây dựng một hệ thống quản lý bãi đỗ xe nhằm giải quyết những khuyết điểm của mô hình truyền thống bằng cách số hóa quá trình quản lý như lưu trữ thông tin xe ra vào bãi (biển số, thời gian ra/vào) trong cơ sở dữ liệu, giúp dễ dàng theo dõi và kiểm tra. Cải thiện hiệu suất quản lý như tự động cập nhật số lượng xe có trong bãi, trình trạng chỗ trống và cảnh báo khi bãi quá tải. Tăng cường bảo mật như lưu trữ thông tin gửi xe và đối chiếu với mã định danh duy nhất nhằm làm giảm tình trạng mất cấp xe. Tất cả những điều trên nhằm mang lại sự tiện lợi và an toàn cho người gửi xe.

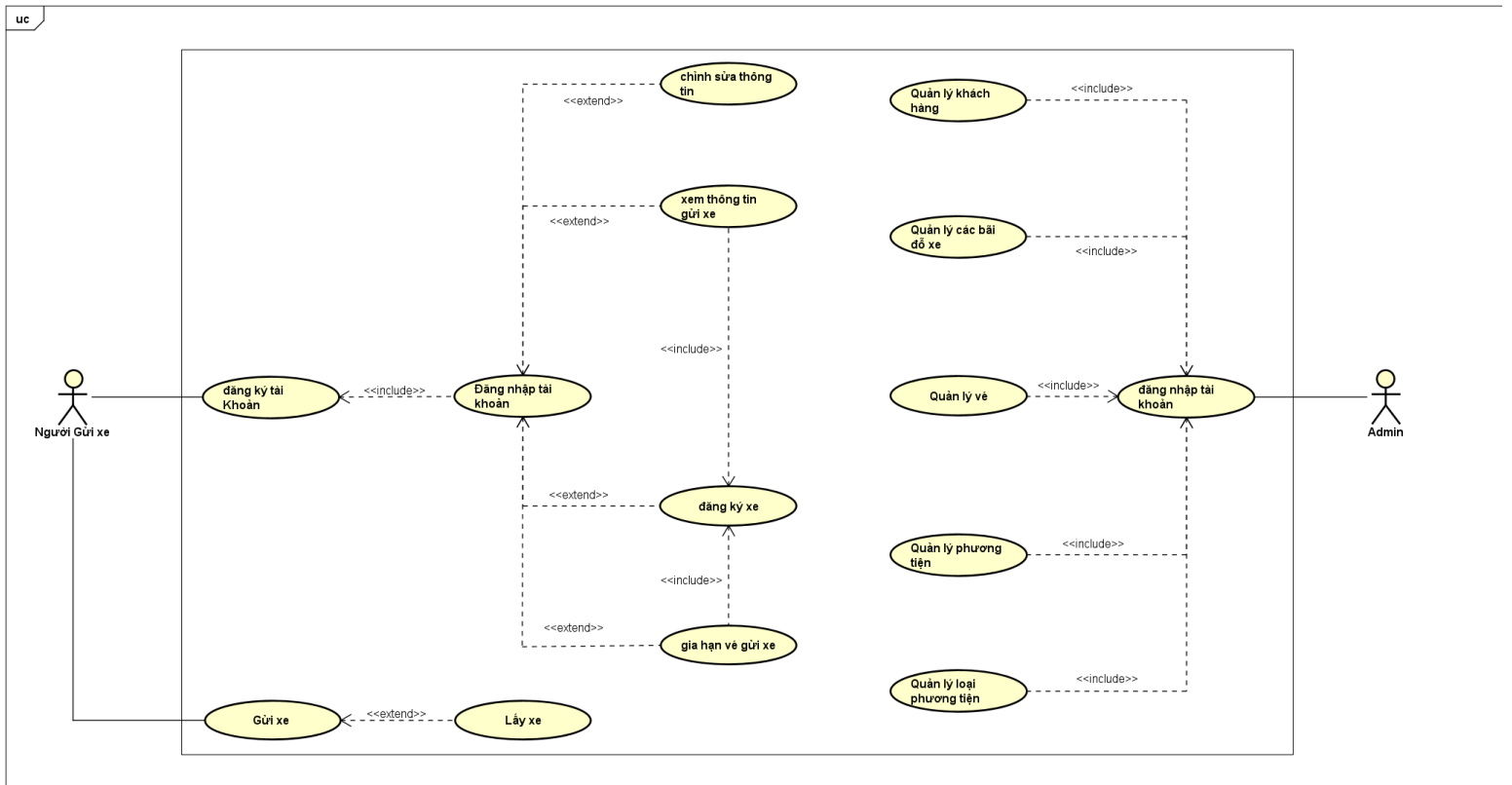
III. Phân tích và thiết kế hệ thống

1. Đối tượng sử dụng

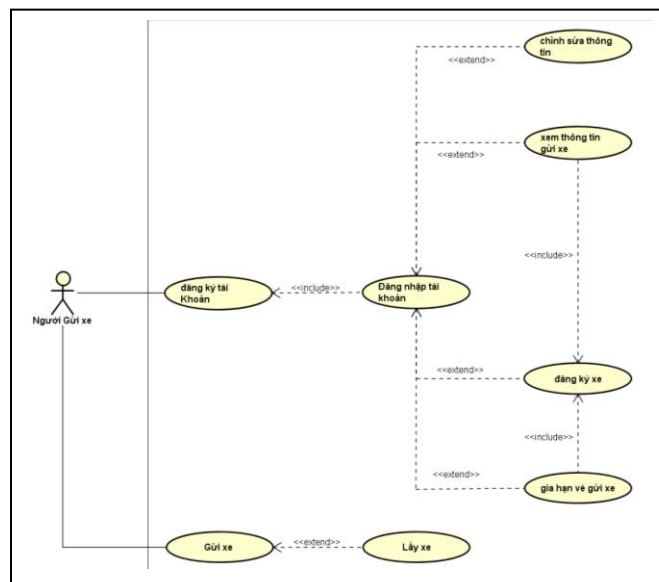
- Nhân viên, quản lý bãi gửi xe
- Người gửi xe

2. Biểu đồ User case

- biểu đồ User case

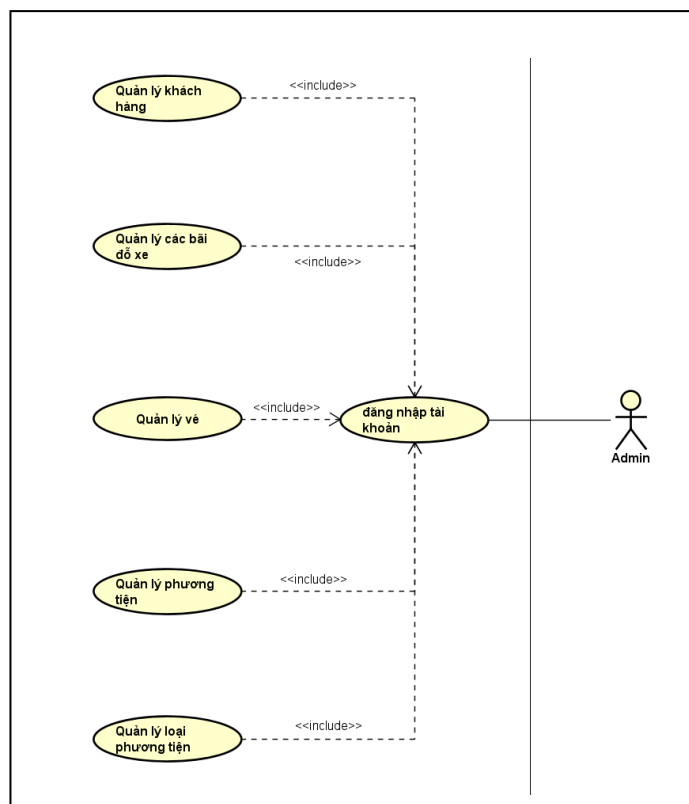


- Chức năng của người dùng



Chức năng của người dùng:

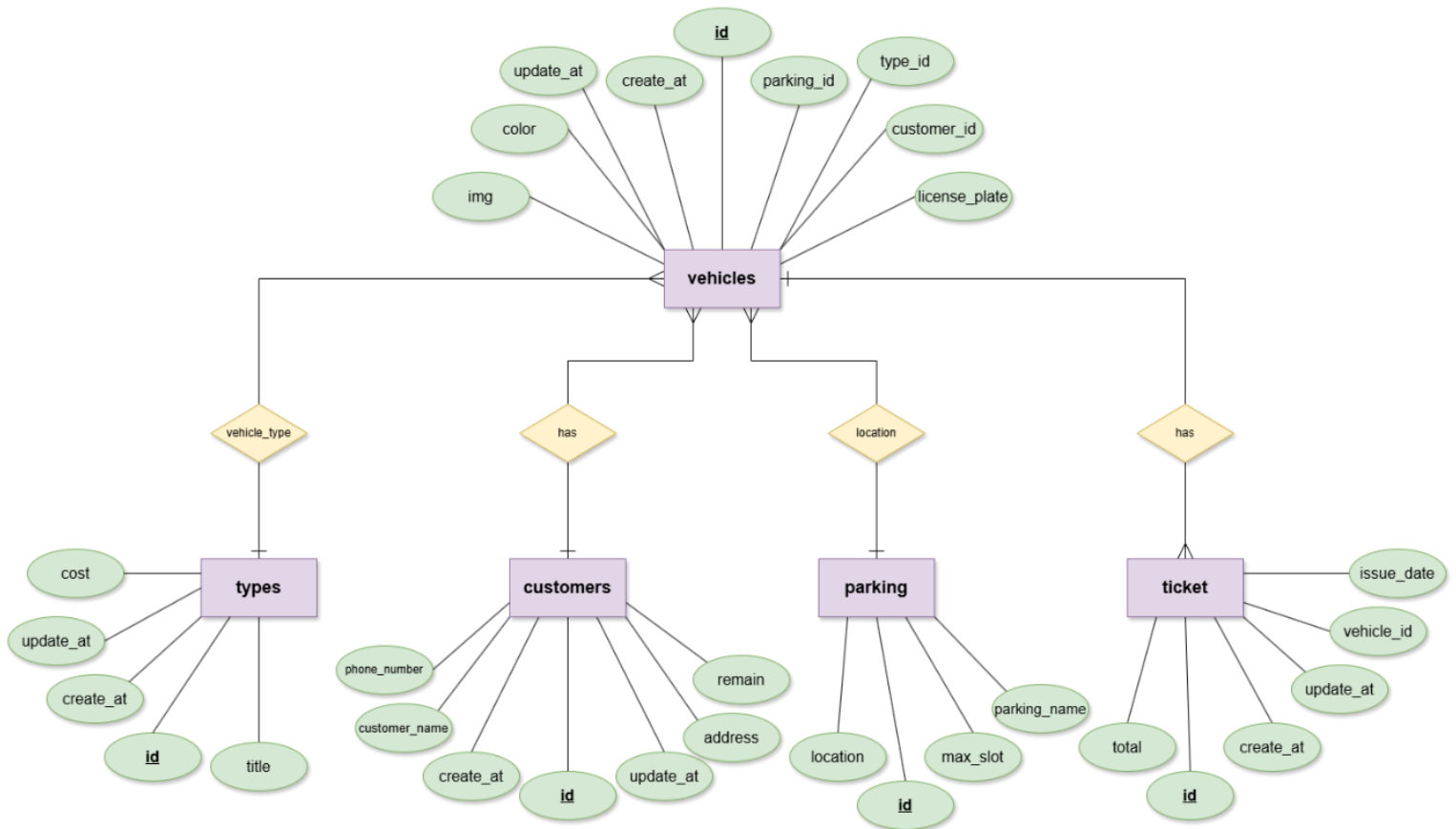
- Đăng ký tài khoản: người dùng cần nhập những thông tin cần thiết như tên, số điện thoại, địa chỉ để đăng ký tài khoản.
 - Chỉnh sửa thông tin: sau khi thực hiện đăng ký nếu như người dùng cảm thấy thông tin cũ không còn chính xác nữa thì họ có thể chọn thay đổi thông tin.
 - Đăng ký xe: người dùng cần nhập những thông tin cần thiết của xe như biển số xe, màu xe, loại xe để thực hiện đăng ký
 - Gửi xe và gia hạn: sau đó người dùng có thể thực hiện gửi xe
 - Xem thông tin gửi xe: sau khi đã gửi xe người dùng sẽ có thể xem những thông tin về xe mình như gửi ở đâu.
- Chức năng quản lý



Chức năng của quản lý:

- Về cơ bản thì mọi chức năng của người dùng quản lý đều có thể thực hiện
- Quản lý khách hàng: quản lý có thể thực hiện xem, thêm, sửa, xóa khách hàng
- Quản lý các bãi đỗ xe: cập nhật bãi đỗ xe
- Quản lý vé: thêm, sửa, xóa cập nhật các loại vé
- Quản lý phương tiện: xem, thêm, sửa, xóa các phương tiện đã đăng ký
- Quản lý loại phương tiện: cập nhật các loại phương tiện khác nhau như: oto, oto điện, xe máy, xe đạp,

3. ERD



The image displays a series of UML diagrams for a car rental system, organized into four main layers: Controller, Service, DTO, and Repository.

- Controller Layer:** Contains `CustomerCtrl`, `VehicleTypeCtrl`, `TicketCtrl`, `VehicleCtrl`, and `ParkingCtrl`. Each controller has a set of methods for handling requests, such as `getCustomerpage`, `getVehiclepage`, `addCustomer`, `addVehicle`, `addTicket`, and `addParking`.
- Service Layer:** Contains `CustomerService`, `VehicleService`, `TicketService`, `VehicleTypeService`, and `ParkingService`. These services implement the logic for the controllers, with methods like `getCustomerpageRequest`, `getVehiclepageRequest`, `addCustomer`, `addVehicle`, `addTicket`, and `addParking`.
- DTO Layer:** Contains `CustomerDTO`, `VehicleTypeDTO`, `TicketDTO`, `VehicleDTO`, and `ParkingDTO`. These data transfer objects represent the data structure for the services, with attributes like `customerName`, `vehicleName`, `ticketId`, `vehicleId`, and `parkingId`.
- Repository Layer:** Contains `TicketRepository`, `VehicleRepository`, `ParkingRepository`, `CustomerRepository`, and `VehicleTypeRepository`. These repositories manage the data storage and retrieval, with methods like `findTicket`, `findVehicle`, `findParking`, `findCustomer`, and `findVehicleType`.

The diagrams illustrate the relationships between these components, including dependencies, inheritance, and associations. For example, the `Vehicle` class in the Model layer is associated with `VehicleType` and `Parking` classes, and the `VehicleRepository` is associated with the `Vehicle` class.

- Package:
 - Controllers : thực hiện giao tiếp với Frontend thông qua API
 - DTOs: chứa các lớp DTO dùng để truyền tải dữ liệu giữa tầng Presentation và Business Logic
 - Services: chứa các lớp thực hiện logic của chương trình
 - models: chứa các lớp dùng để nhận dữ liệu từ database đến tầng Data access
 - repositories: chứa những interface dùng để trao đổi dữ liệu với data access
- Controllers:
 - **Class CustomerCtrl:**
 - **getCustomers():** Dùng để lấy danh sách khách hàng với phân trang và có thể tìm kiếm theo tên hoặc thông tin khác.
 - **getCustomer(Long customerId):** Dùng để lấy thông tin chi tiết của một khách hàng theo ID.

- **getCustomer(String phoneNum):** Dùng để lấy thông tin khách hàng dựa trên số điện thoại.
- **createCustomer(CustomerDTO customerDTO):** Dùng để tạo mới một khách hàng.
- **updateCustomer(CustomerDTO customerDTO, Long customerId):** Dùng để cập nhật thông tin khách hàng theo ID.
- **deleteCustomer(Long customerId):** Dùng để xóa khách hàng theo ID.
- **Class ParkingCtrl**
 - **getParkings():** Dùng để lấy danh sách bãi đỗ xe với phân trang.
 - **addVehicleToParking(Long parkingId, Long vehicleId):** Dùng để thêm một xe vào bãi đỗ, cập nhật số lượng chỗ trống còn lại.
 - **removeVehicleFromParking(Long parkingId, Long vehicleId):** Dùng để xóa xe khỏi bãi đỗ, cập nhật số lượng chỗ trống còn lại.
 - **createParking(ParkingDTO parkingDTO):** Dùng để tạo mới một bãi đỗ xe.
 - **updateParking(ParkingDTO parkingDTO, Long id):** Dùng để cập nhật thông tin bãi đỗ xe theo ID.
 - **deleteParking(Long id):** Dùng để xóa một bãi đỗ xe theo ID.
- **Class TicketCtrl**
 - **getTickets():** Dùng để lấy danh sách vé với phân trang.
 - **getTicketById(Long ticketId):** Dùng để lấy thông tin chi tiết của vé theo ID.
 - **checkAndCreateTicket(VehicleDTO vehicle):** Dùng để kiểm tra và tạo mới vé cho phương tiện.
 - **createTicket(TicketDTO ticketDTO):** Dùng để tạo mới một vé.
 - **updateTicket(Long ticketId, TicketDTO ticketDTO):** Dùng để cập nhật thông tin vé theo ID.
 - **returnTicket(Long ticketId, Long customerId):** Dùng để trả vé và cập nhật thông tin.
 - **deleteTicket(Long ticketId):** Dùng để xóa vé theo ID.
- **Class VehicleCtrl**
 - **getVehicles():** Dùng để lấy danh sách phương tiện với phân trang và các bộ lọc theo customer_id, type_id, và license.
 - **getVehicleById(Long vehicleId):** Dùng để lấy thông tin chi tiết của phương tiện theo ID.
 - **createVehicle(VehicleDTO vehicleDTO):** Dùng để tạo mới một phương tiện.
 - **updateVehicle(Long id, VehicleDTO vehicleDTO):** Dùng để cập nhật thông tin phương tiện theo ID.
 - **deleteVehicle(Long vehicleId):** Dùng để xóa phương tiện theo ID.
- **Class VehicleCtrl**
 - **getVehicleTypes():** Dùng để lấy danh sách tất cả các loại phương tiện.

- **getVehicleTypeById(Long vehicleTypeId)**: Dùng để lấy thông tin chi tiết của loại phương tiện theo ID.
- **createVehicleType(VehicleTypeDTO vehicleTypeDTO)**: Dùng để tạo mới một loại phương tiện.
- **updateVehicleType(Long vehicleTypeId, VehicleTypeDTO vehicleTypeDTO)**: Dùng để cập nhật thông tin loại phương tiện theo ID.
- **deleteVehicleType(Long vehicleTypeId)**: Dùng để xóa loại phương tiện theo ID.

- Services

- **Class CustomerServiceImpl**
 - **getCustomers(String search, PageRequest pageRequest)**: Lấy danh sách khách hàng với khả năng tìm kiếm và phân trang.
 - **getCustomer(long id)**: Lấy thông tin khách hàng theo ID.
 - **getCustomer(String phone_num)**: Lấy thông tin khách hàng theo số điện thoại.
 - **createCustomer(CustomerDTO customerDTO)**: Tạo mới một khách hàng.
 - **updateCustomer(CustomerDTO customerDTO, long id)**: Cập nhật thông tin khách hàng theo ID.
 - **deleteCustomer(long id)**: Xóa khách hàng theo ID.
- **Class ParkingServiceImpl**
 - **addVehicleToParking(long parkingId, long vehicleId)**: Thêm xe vào bãi đỗ.
 - **removeVehicleFromParking(long parkingId, long vehicleId)**: Xóa xe khỏi bãi đỗ.
 - **getVehicleCountInParking(long parkingId)**: Lấy số lượng xe hiện có trong bãi đỗ.
 - **getParkings(PageRequest pageRequest)**: Lấy danh sách các bãi đỗ với phân trang.
 - **updateParking(ParkingDTO parkingDTO, long id)**: Cập nhật thông tin bãi đỗ.
 - **deleteParking(long id)**: Xóa bãi đỗ.
 - **getParkingById(long id)**: Lấy thông tin bãi đỗ theo ID.
 - **createParking(ParkingDTO parkingDTO)**: Tạo mới bãi đỗ.
- **Class TicketServiceImpl**
 - **getTickets(PageRequest pageRequest)**: Lấy danh sách vé với phân trang
 - **getTicketById(long id)**: Lấy thông tin vé theo ID
 - **createTicket(TicketDTO ticketDTO)**: Tạo một vé mới từ thông tin trong TicketDTO
 - **updateTicket(TicketDTO ticketDTO, long id)**: Cập nhật thông tin vé theo ID
 - **deleteTicket(long id)**: Xóa vé theo ID

- **returnTicket(long id):** Thực hiện trả vé, kiểm tra xem khách hàng có đủ tiền trả không
- **checkAndCreateTicket(VehicleDTO vehicleDTO):** Kiểm tra và tạo vé cho xe, nếu xe chưa có trong hệ thống sẽ tạo mới
- **Class VehicleServiceImpl**
 - **getVehicles(Long customerId, Long typeId, String license, PageRequest pageRequest):** Lấy danh sách xe với các bộ lọc (khách hàng, loại xe, biển số) và phân trang.
 - **getVehicleById(long id):** Lấy thông tin xe theo ID.
 - **createVehicle(VehicleDTO vehicleDTO):** Tạo mới một xe từ thông tin trong VehicleDTO.
 - **updateVehicle(VehicleDTO vehicleDTO, long id):** Cập nhật thông tin xe theo ID.
 - **deleteVehicle(long id):** Xóa xe theo ID.
- **Class VehicleTypeServiceImpl**
 - **getVehicleTypes():** Lấy danh sách tất cả các loại xe.
 - **getVehicleTypeById(long id):** Lấy thông tin loại xe theo ID.
 - **createVehicleType(VehicleTypeDTO vehicleTypeDTO):** Tạo mới một loại xe từ thông tin trong VehicleTypeDTO.
 - **updateVehicleType(VehicleTypeDTO vehicleTypeDTO, long id):** Cập nhật thông tin loại xe theo ID.
 - **deleteVehicleType(long id):** Xóa loại xe theo ID.
- các class còn lại DTO, Model hay interface Repositories nhìn chung chỉ thực hiện 1 chức năng đặc thù đã nói ở trên.

IV. Triển khai hệ thống

1. Công nghệ sử dụng

Hệ thống được phát triển theo 2 công nghệ phổ biến nhất hiện nay là java spring boots và reactJS và hệ quản trị cơ sở dữ liệu SQL Server. Trong đó:

Java Spring Boot là một framework mã nguồn mở được phát triển dựa trên nền tảng Spring, giúp xây dựng các ứng dụng Java dễ dàng và nhanh chóng. Spring Boot tự động cấu hình ứng dụng, giảm thiểu sự cần thiết phải cấu hình thủ công, giúp người phát triển tập trung vào việc xây dựng logic nghiệp vụ. Nó hỗ trợ phát triển các ứng dụng backend mạnh mẽ, dễ mở rộng, và tương thích với các công nghệ như RESTful API, microservices, và cơ sở dữ liệu SQL/NoSQL.

ReactJS là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook, dùng để xây dựng giao diện người dùng (UI) cho các ứng dụng web. ReactJS cho phép tạo ra các component tái sử dụng, giúp tăng cường tính linh hoạt và khả năng mở rộng của ứng dụng. Nó sử dụng kỹ thuật "Virtual DOM" để cải thiện hiệu suất và giảm thiểu các lần render không cần thiết, mang lại trải nghiệm người dùng mượt mà và hiệu quả.

SQL Server là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) của Microsoft, hỗ trợ ngôn ngữ SQL để lưu trữ, quản lý, và truy xuất dữ liệu hiệu quả. Với khả năng xử lý truy vấn phức tạp, bảo mật cao.

2. Mô hình hệ thống

Dự án sử dụng 2 mô hình là MVC (Model -View-Controller) kết hợp với mô hình 3 lớp (3 layer). Cụ thể về các thành phần trong mô hình như sau:

- Lớp Presentation: lớp này chứa frontend và controller và có nhiệm vụ chính là giao tiếp với người dùng rồi gửi dữ liệu cho lớp Business Logic và ngược lại.
- Lớp Business Logic: lớp này chứa phần services nơi thực hiện toàn bộ logic của chương trình, xử lý dữ liệu và truyền xuống cho lớp Data access và ngược lại là kiểm tra các ràng buộc, tính toàn vẹn, hợp lệ, yêu cầu nghiệp vụ trước khi gửi trả về cho lớp presentation. Dữ liệu ở lớp này và lớp presentation ở dạng DTO.
- Lớp Data access layer: lớp này chứa các entity và repositories thực hiện chức năng giao tiếp với hệ quản trị CSDL (tìm kiếm, thêm, sửa, xóa dữ liệu).

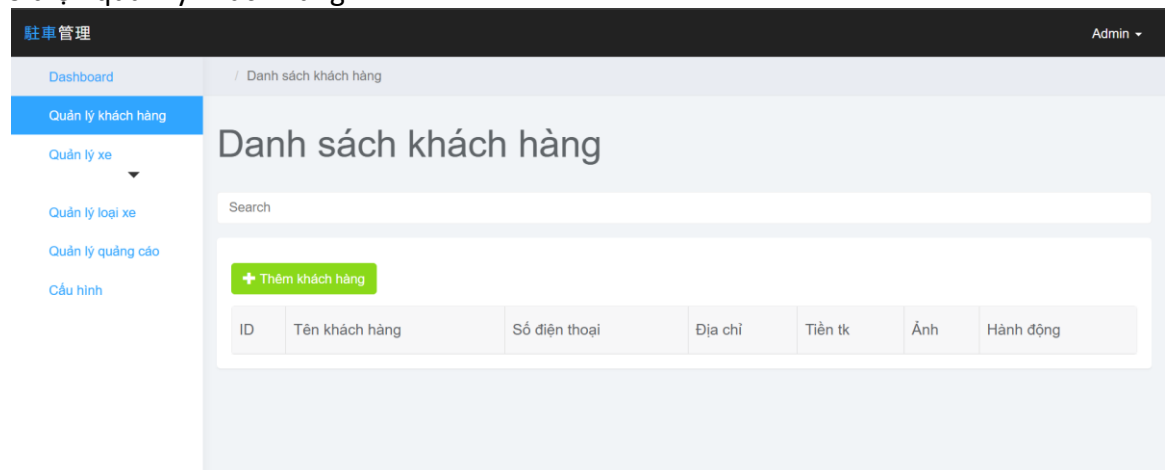
Trong dự án bọn em đã chia các thành phần vào các Package như sau:

- Controllers: thực hiện việc trao dữ liệu dưới dạng API với frontend và chuyển nó sang DTO rồi đẩy dữ liệu đó cho services xử lý
- dto: lưu trữ những lớp DTO như customerDTO, vehicleDTO,.... mục đích của các DTO là để giảm bớt lượng info không cần thiết phải chuyển đi, và cũng tăng cường độ bảo mật.
- Services: chứa những lớp thực hiện các tác vụ xử lý logic, kiểm tra ràng buộc, tính toàn vẹn của dữ liệu.
- Models: chứa các lớp entity, mỗi lớp này tương ứng với một bảng trong cơ sở dữ liệu và mỗi đối tượng sẽ tương ứng với một bản ghi.
- Repositories: chứa các interface thực hiện việc trao đổi dữ liệu với cơ sở dữ liệu

V. Chạy thử

1. Thêm mới một khách hàng

- Giao diện quản lý khách hàng



- ấn vào thêm khách hàng

Thêm khách hàng

Tên khách hàng:

Số điện thoại:

Địa chỉ:

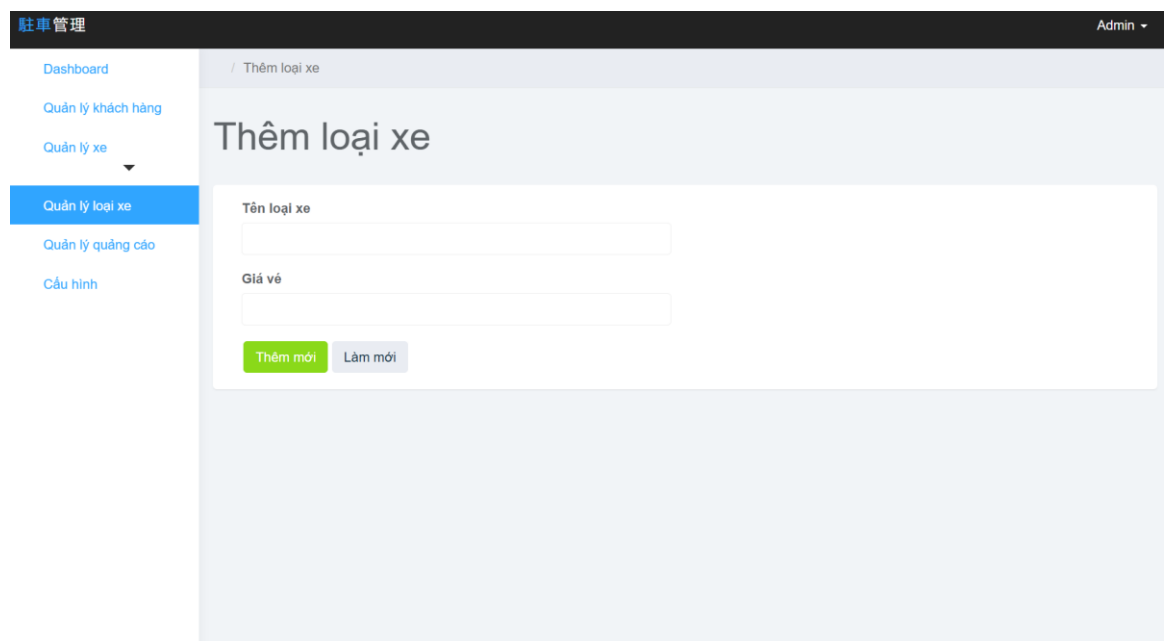
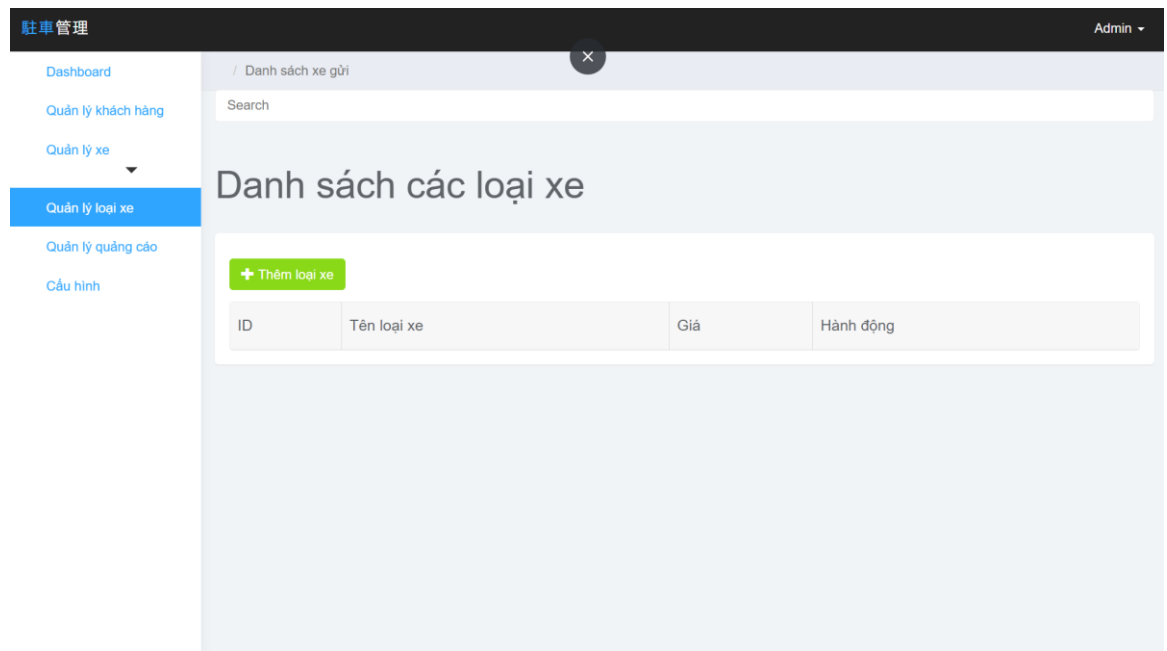
Số tiền:

Ảnh khách hàng: Không có tệp nào được chọn

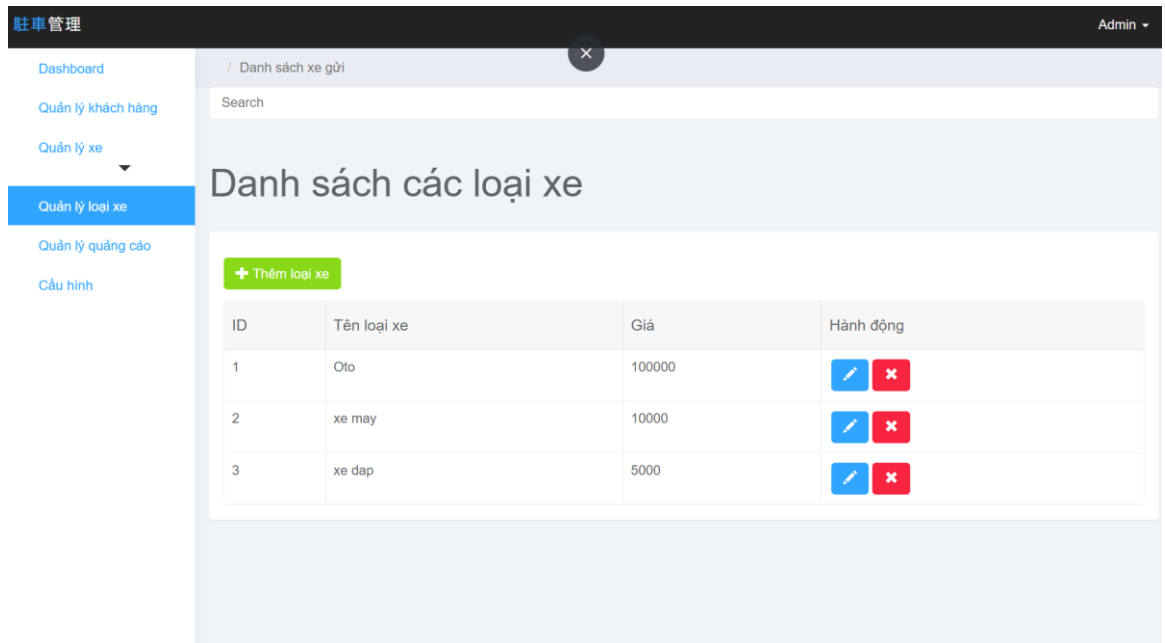
- Tại đây nhập những thông tin cần thiết
- danh sách khách hàng đã thêm

ID	Tên khách hàng	Số điện thoại	Địa chỉ	Tiền tk	Ảnh	Hành động
1	Phạm Đức Long	0111222333	Thanh Xuân, Hà Nội	1000000		<input type="button" value="Sửa"/> <input type="button" value="Xóa"/> <input type="button" value="Xem danh sách xe"/> <input type="button" value="Xem danh sách vé"/>
2	Phạm Đức Duy	0	Hà Đông, Hà Nội	1000000		<input type="button" value="Sửa"/> <input type="button" value="Xóa"/> <input type="button" value="Xem danh sách xe"/> <input type="button" value="Xem danh sách vé"/>

- Tại danh sách khách hàng ta còn những tùy chọn như sửa, xóa, xem danh sách xe, xem danh sách vé đối với từng khách hàng
- 2. Thêm loại xe mới
- Giao diện thêm loại xe

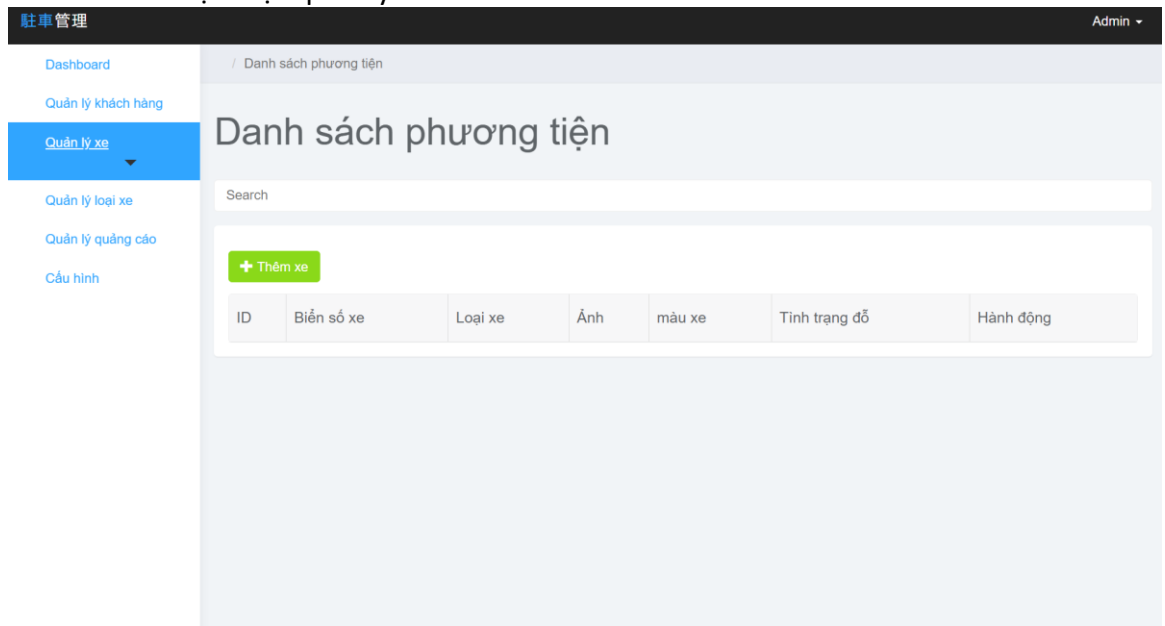


- Danh sách sau khi tạo



3. Thêm xe mới

- Thêm xe mới tại hoặc quản lý xe



- Giao diện thêm mới

Admin

Dashboard

Quản lý khách hàng

Quản lý xe

Quản lý loại xe

Quản lý quảng cáo

Cấu hình

Biển số xe

25A9837

Chủ xe (Điện số điện thoại)

0111222333

Tên chủ xe

Pham Duc Long

Loại xe

Oto

Thêm mới


Làm mới

Ảnh xe

Chọn tệp

Không có tệp nào được chọn

Ảnh khách hàng



- Sau khi thêm xe

Admin

Dashboard

Quản lý khách hàng

Quản lý xe

Quản lý loại xe


Quản lý quảng cáo

Cấu hình

Danh sách phương tiện

Search

Thêm xe

ID	Biển số xe	Loại xe	Ảnh	màu xe	Tình trạng đỗ	Hành động
1	25A9837	xe máy			Xe không đỗ	<div></div> <div></div>

VI. Đánh giá và kết luận

Nhìn chung hệ thống bãi đỗ xe không chỉ giải quyết được những khuyết điểm của mô hình gửi xe truyền thống mà còn tận dụng những công nghệ hiện đại để tối ưu hóa hiệu quả quản lý.

Với việc sử dụng Java Spring boot, React JS, SQL server, hệ thống được thiết kế để đảm bảo hiệu suất cao, dễ dàng xây dựng và mở rộng trong tương lai, đồng thời đáp ứng được những yêu cầu phức tạp.

Bên cạnh đó hệ thống tuân theo mô hình kết hợp giữa MVC và mô hình 3 lớp nhằm giúp cho mã nguồn trở nên dễ dàng bảo trì và đảm bảo rõ ràng trong việc phân chia nhiệm vụ giữa các lớp. đảm bảo tuân thủ nguyên tắc lập trình hướng đối tượng (OOP)

Các tính chất của OOP như tính trừu tượng, tính đóng gói, tính kế thừa và đa hình nhằm giúp mã nguồn dễ dàng mở rộng và tái sử dụng, đồng thời giảm thiểu lỗi phát sinh khi phát triển chức năng mới.