

```
Public void Lập trình hướng đối tượng {  
    Nhóm_10.quản lý bãi đỗ xe();  
}
```



## Public class Giới thiệu thành viên {

```
new Student("Phạm Văn Đồng","Fullstack");  
new Student("Phạm Đức Long","Thiết kế hệ thống, service, thuyết trình");  
new Student("Phan Đức Duy","Báo cáo, slide, entity, DTOs");  
new Student("Nguyễn Bá Hoàng","Usecase, Database, service");  
new Student("Phạm Quốc Minh","Frontend, ERD, Controller");  
}
```

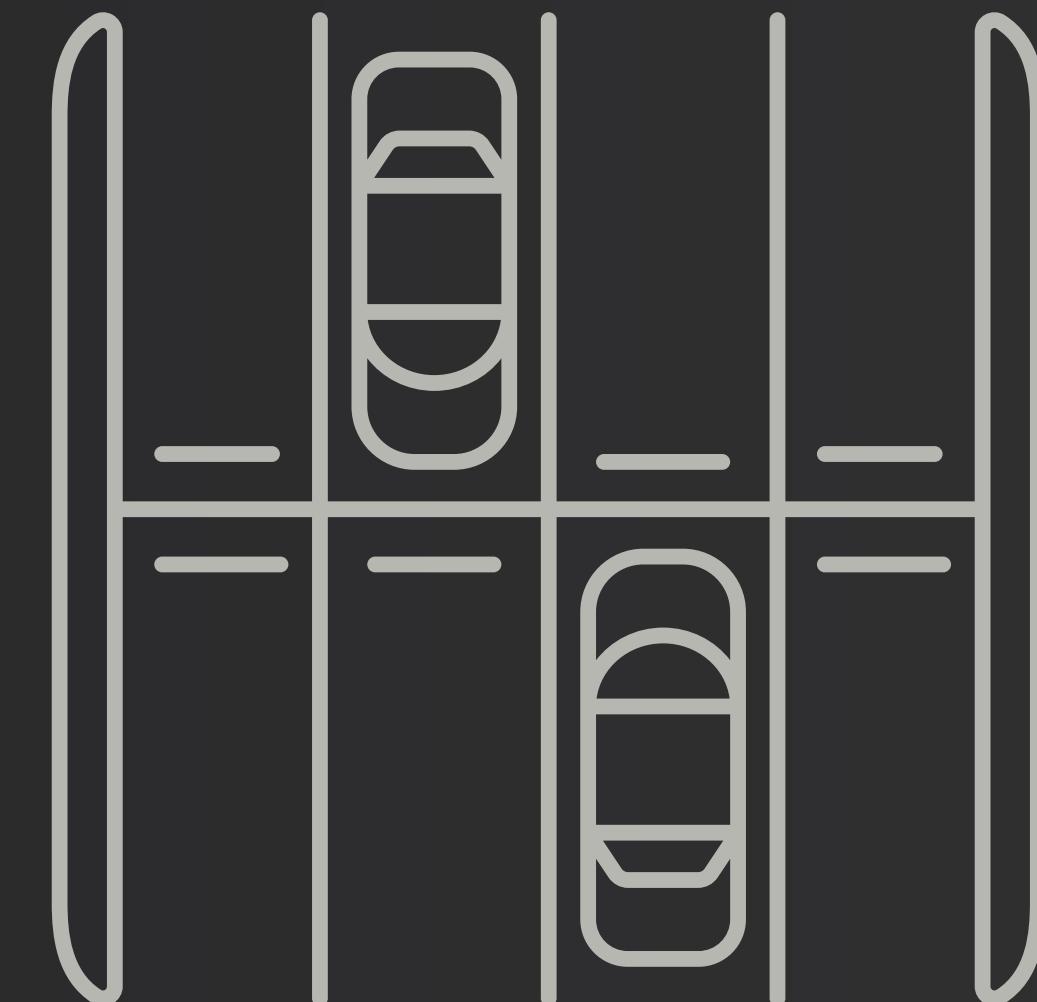
## Giới thiệu bài toán



Xây dựng một hệ thống quản lý thông tin bãi gửi xe cơ bản, sử dụng app cho phép người dùng có thể đăng ký gửi xe online

Gồm các chức năng cơ bản như:

- đăng ký thông tin người dùng
- đăng ký gửi xe
- quản lý xe
- quản lý vé xe
- ...



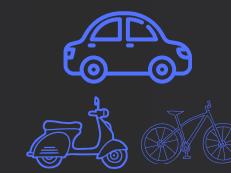
## Các thực thể chính



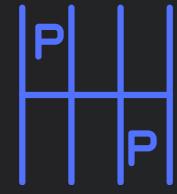
Khách hàng



Xe



Loại xe



Bãi đỗ

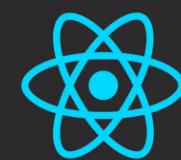


Vé

# Công nghệ sử dụng



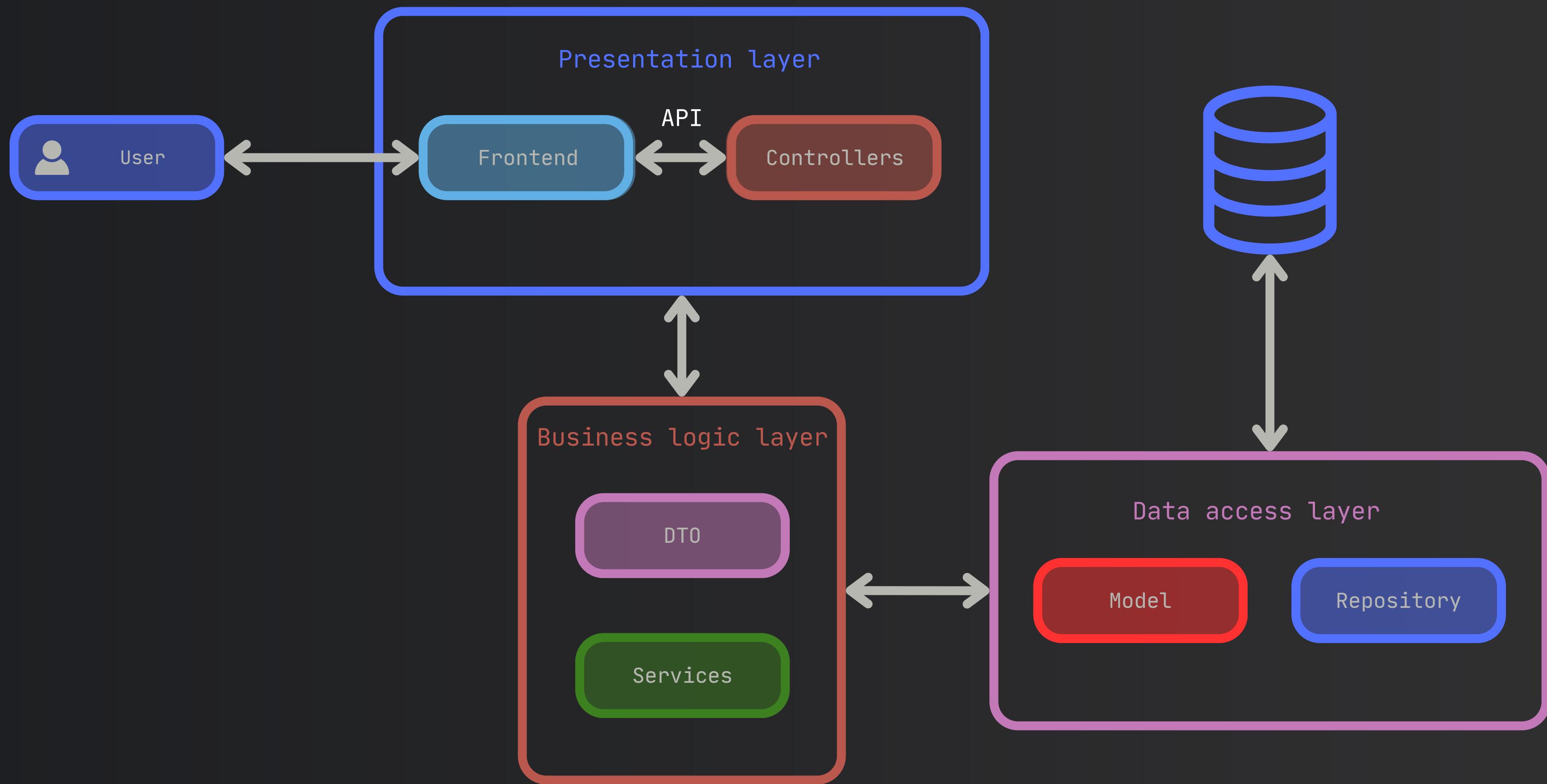
Backend: java spring boot



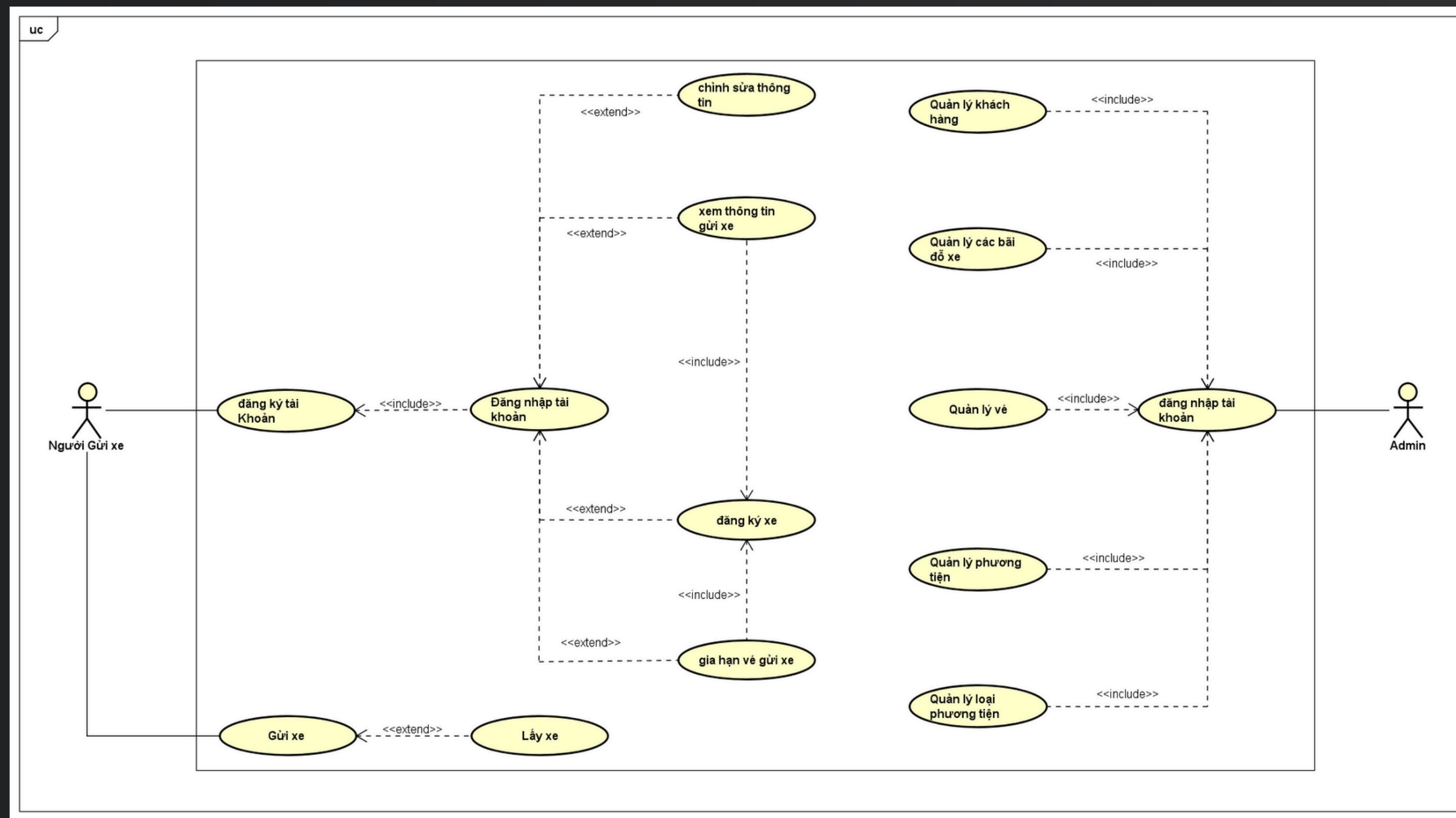
Frontend: ReactJS



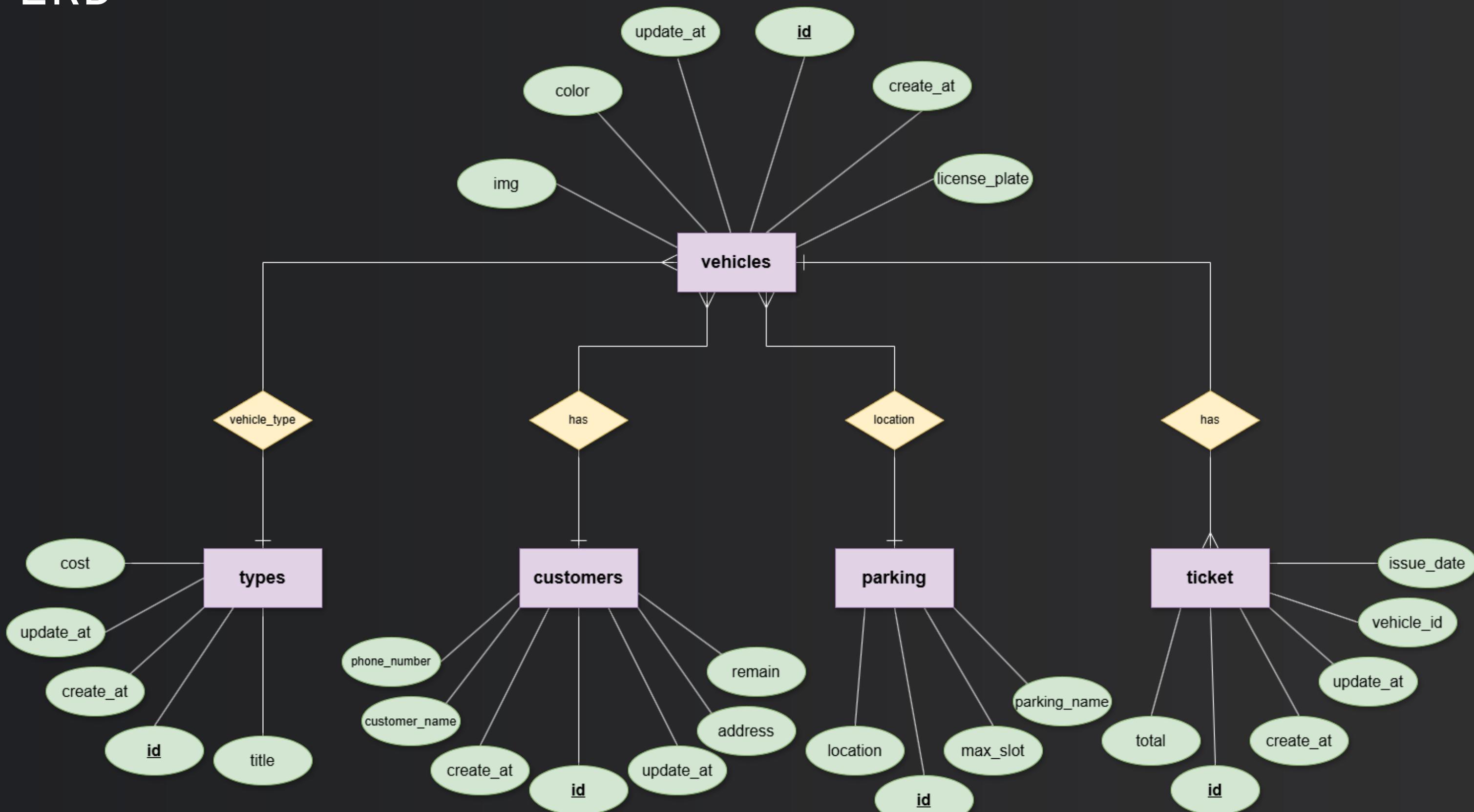
Database: SQL server



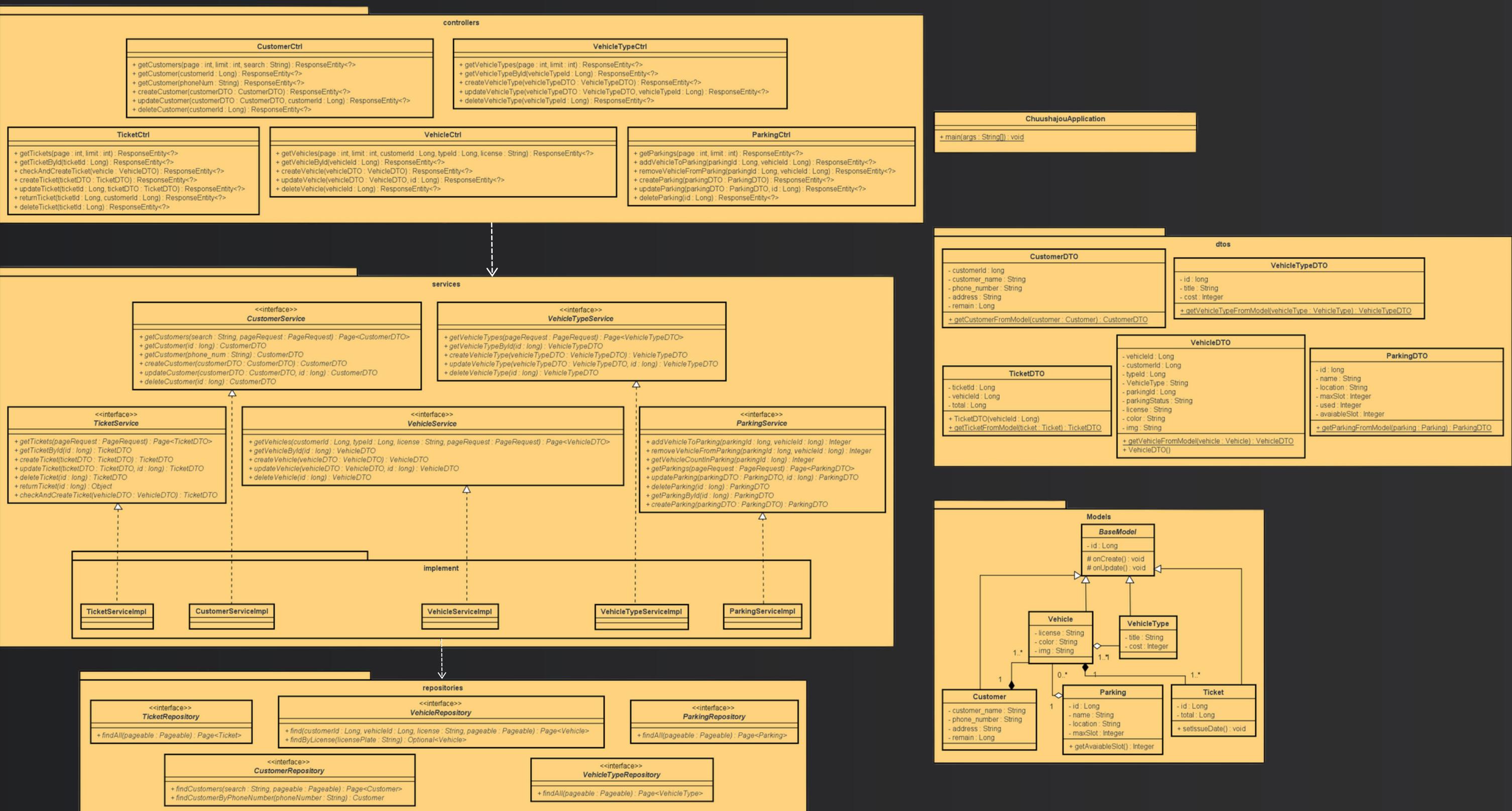
# Usecase



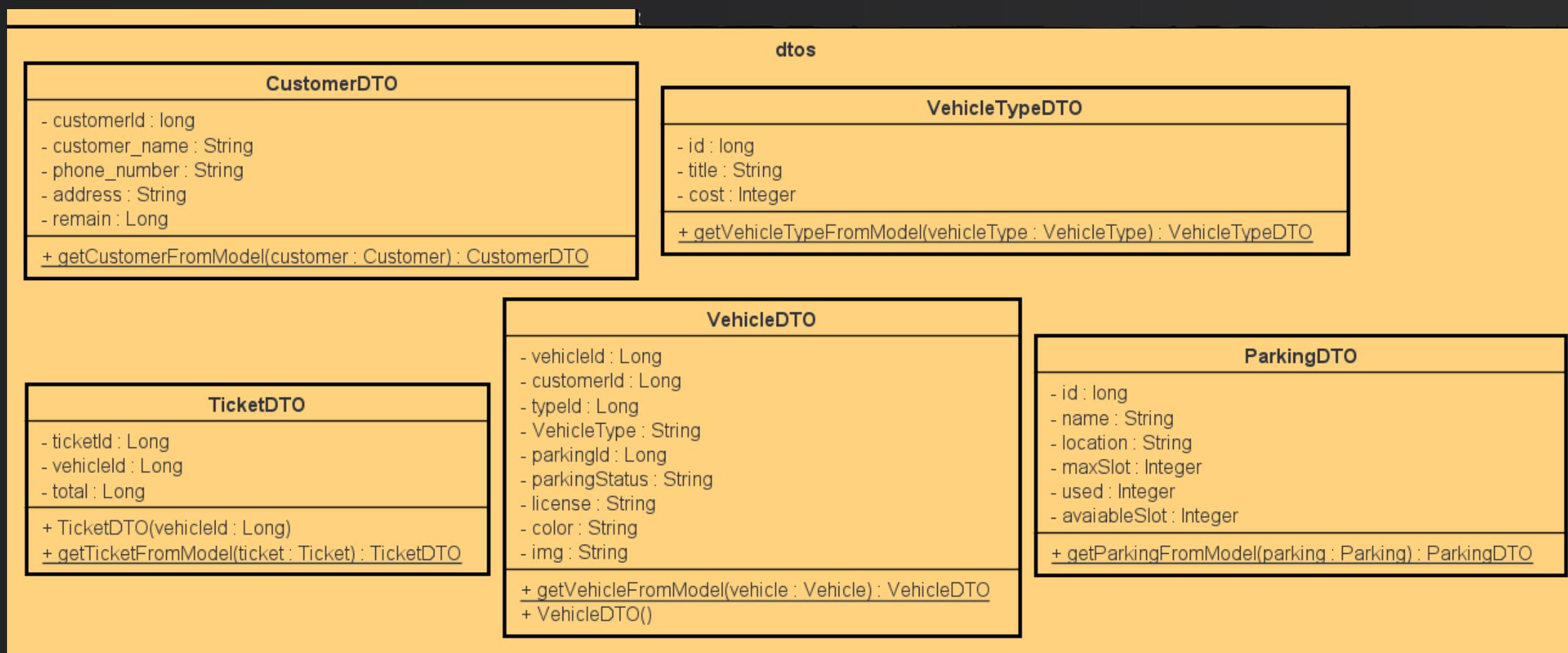
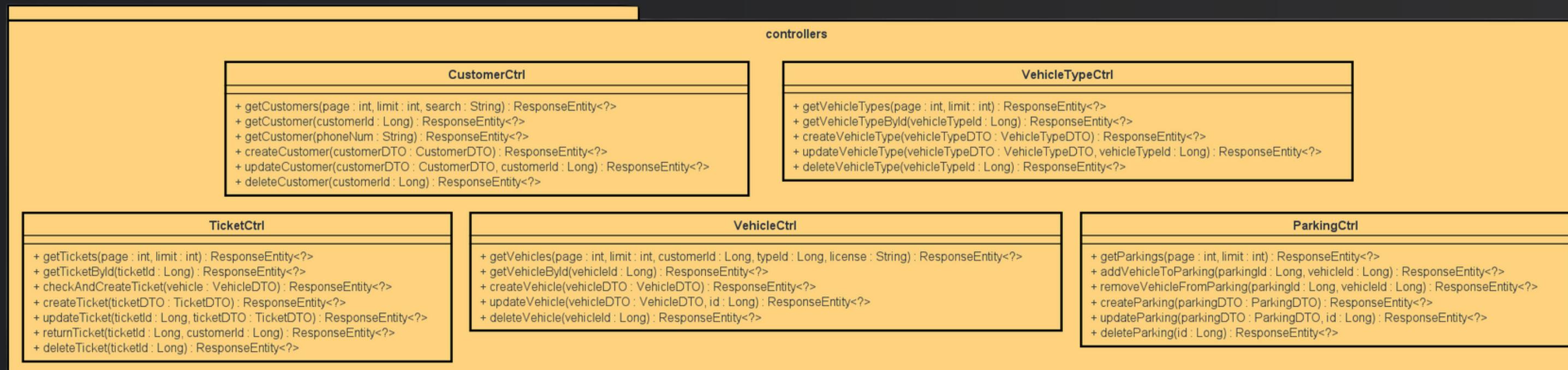
# ERD



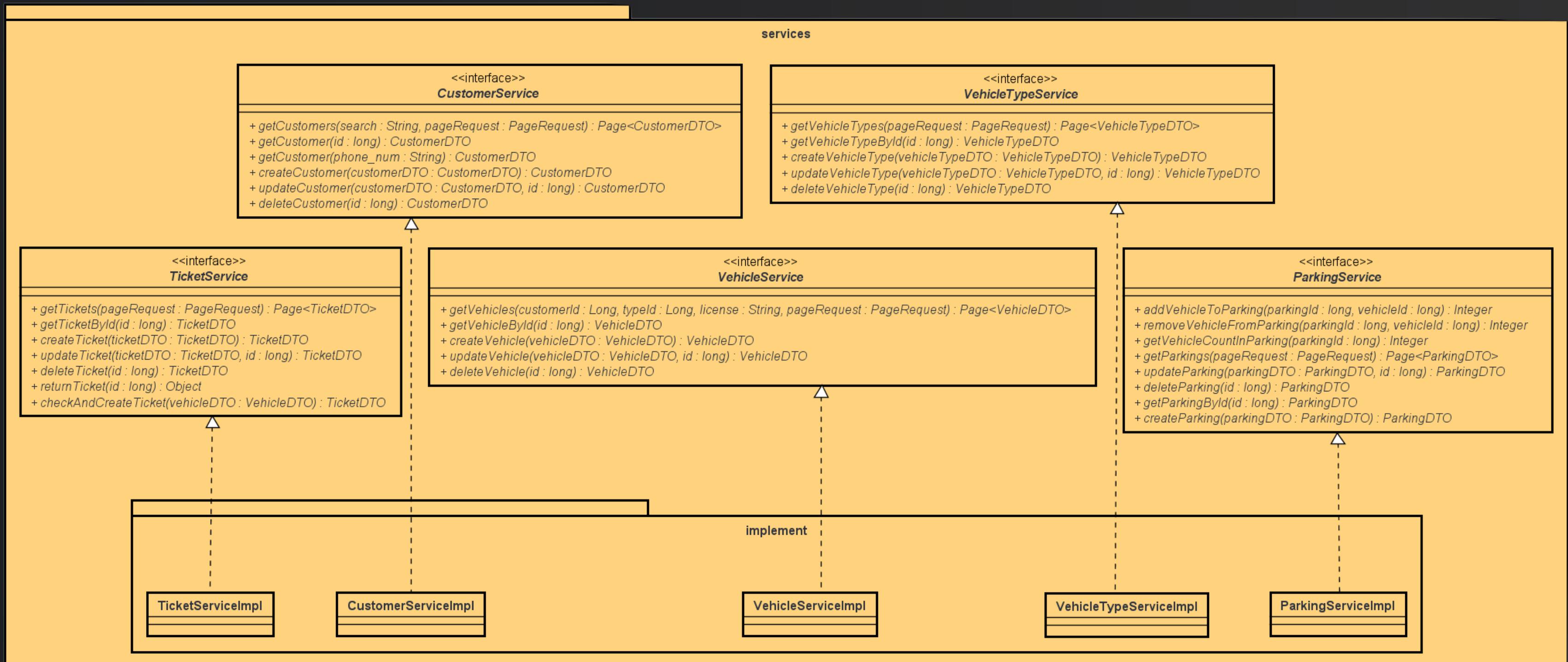
# Class Diagram



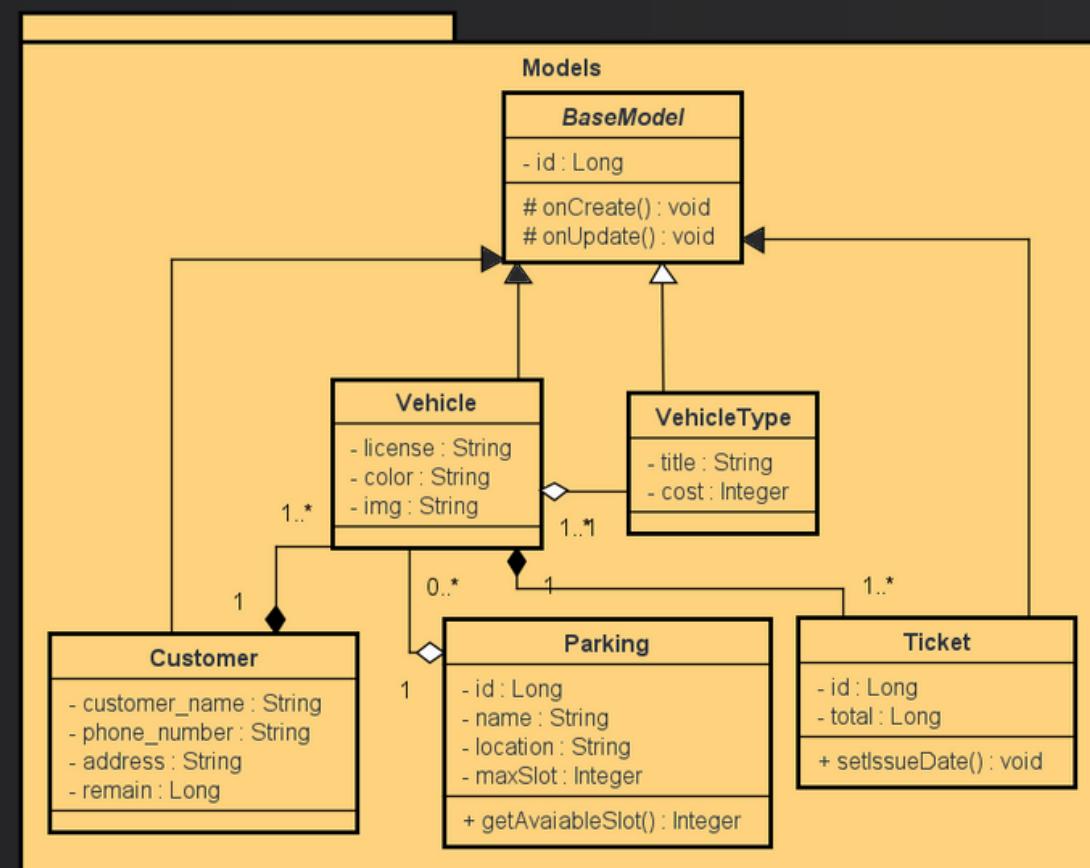
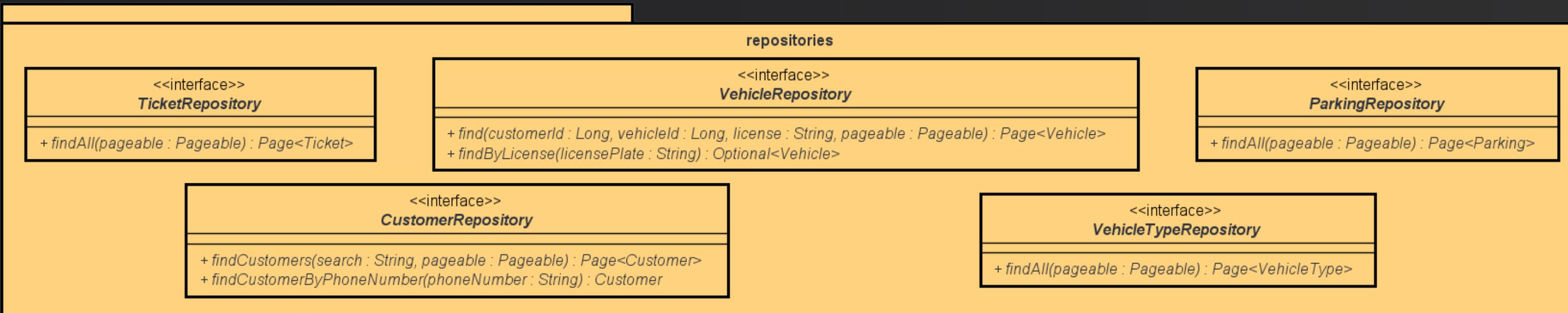
# Controllers & DTOs



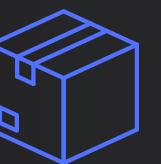
# Services



# Repositories & Models



# Các tính chất của OOP



Tính đóng gói



Tính kế thừa



Tính đa hình



Tính trừu tượng



# Tính đóng gói

```
● ● ●
@Entity 11 usages ± Dng +2 *
@Table(name = "vehicle_types")
public class VehicleType extends BaseModel {
    @Column(name = "title") 2 usages
    private String title;
    @Column(name = "cost") 2 usages
    private Integer cost;
    @OneToMany(mappedBy = "type", cascade = CascadeType.ALL)
    private List<Vehicle> vehicles;

    public List<Vehicle> getVehicles() { no usages new *
        return vehicles;
    }

    public void setVehicles(List<Vehicle> vehicles) { no usag
        this.vehicles = vehicles;
    }

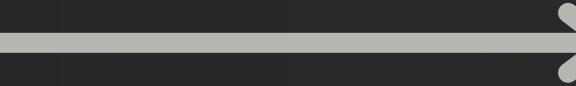
    public String getTitle() { 2 usages new *
        return title;
    }

    public void setTitle(String title) { 2 usages new *
        this.title = title;
    }
}
```



# Tính kế thừa

```
● ● ●  
@Getter 4 usages 4 inheritors ± Dng  
@Setter  
@MappedSuperclass  
public abstract class BaseModel {  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private Long id;  
  
    @Column(name = "created_at")  
    private LocalDateTime createdAt;  
  
    @Column(name = "updated_at")  
    private LocalDateTime updatedAt;  
  
    @PrePersist ± Dng  
    protected void onCreate() {  
        this.createdAt = LocalDateTime.now();  
        this.updatedAt = LocalDateTime.now();  
    }  
  
    @PreUpdate ± Dng  
    protected void onUpdate() {  
        this.updatedAt = LocalDateTime.now();  
    }  
}
```



```
● ● ●  
@Entity 17 usages ± Dng +1  
@Getter  
@Setter  
@Table(name = "customers")  
public class Customer extends BaseModel {  
    @Column(name="customer_name")  
    private String customer_name;  
  
    @Column(name="phone_number", unique = true)  
    private String phone_number;  
  
    @Column(name = "address")  
    private String address;  
  
    @OneToMany(mappedBy = "customer", cascade = CascadeType.ALL)  
    private List<Vehicle> vehicles;  
  
    @Column(name = "remain")  
    private Long remain;  
}
```



# Tính đa hình

```
@Service 4 usages 1 implementation ✎ Dng
public interface CustomerService {
    Page<CustomerDTO> getCustomers(String search, PageRequest pageRequest);

    CustomerDTO getCustomer(Long id); 1 usage 1 implementation ✎ Dng

    CustomerDTO getCustomer(String phone_num); 1 usage 1 implementation ✎ Dng

    CustomerDTO createCustomer(CustomerDTO customerDTO); 1 usage 1 implementation ✎ Dng

    CustomerDTO updateCustomer(CustomerDTO customerDTO, Long id); 1 usage 1 implementation ✎ Dng

    CustomerDTO deleteCustomer(Long id); 1 usage 1 implementation ✎ Dng
}
```



# Tính trừu tượng

```
@Service 4 usages 1 implementation ✎ Dng
public interface CustomerService {
    Page<CustomerDTO> getCustomers(String search, PageRequest pageRequest);

    CustomerDTO getCustomer(Long id); 1 usage 1 implementation ✎ Dng

    CustomerDTO getCustomer(String phone_num); 1 usage 1 implementation ✎ Dng

    CustomerDTO createCustomer(CustomerDTO customerDTO); 1 usage 1 implementation ✎ Dng

    CustomerDTO updateCustomer(CustomerDTO customerDTO, Long id); 1 usage 1 implementation ✎ Dng

    CustomerDTO deleteCustomer(Long id); 1 usage 1 implementation ✎ Dng
}
```

**THANKS FOR LISTENING**