# TÀI LIỆU PHÂN TÍCH YÊU CẦU HỆ THỐNG

## ỨNG DỤNG WEB ĐẶT VÉ XE KHÁCH TRỰC TUYẾN

## MỤC LỤC

1. [Khảo sát Nghiệp vụ](https://claude.ai/chat/438386db-74a2-4bea-9632-61c0f593bf39#1-kh%E1%BA%A3o-s%C3%A1t-nghi%E1%BB%87p-v%E1%BB%A5)
2. [Mô tả Bài toán](https://claude.ai/chat/438386db-74a2-4bea-9632-61c0f593bf39#2-m%C3%B4-t%E1%BA%A3-b%C3%A0i-to%C3%A1n)
3. [Xác định Yêu cầu](https://claude.ai/chat/438386db-74a2-4bea-9632-61c0f593bf39#3-x%C3%A1c-%C4%91%E1%BB%8Bnh-y%C3%AAu-c%E1%BA%A7u)
4. [Xác định Tác nhân và Use Case](https://claude.ai/chat/438386db-74a2-4bea-9632-61c0f593bf39#4-x%C3%A1c-%C4%91%E1%BB%8Bnh-t%C3%A1c-nh%C3%A2n-v%C3%A0-use-case)
5. [Đặc tả Use Case Chi tiết](https://claude.ai/chat/438386db-74a2-4bea-9632-61c0f593bf39#5-%C4%91%E1%BA%B7c-t%E1%BA%A3-use-case-chi-ti%E1%BA%BFt)

# 1. KHẢO SÁT NGHIỆP VỤ

## 1.1 Quy trình hiện tại (AS-IS)

### Quy trình đặt vé truyền thống:

Khách hàng → Đến bến xe/Quầy vé → Hỏi tuyến & giờ → Nhân viên check ghế trống

↓

Nhận vé giấy ← Trả tiền mặt ← Nhân viên ghi sổ/Excel ← Chọn ghế

Thời gian: 15-30 phút/lần  
 Điểm yếu:

* Phải đến trực tiếp bến xe
* Không biết trước ghế còn trống
* Dễ nhầm lẫn, vé trùng
* Khó quản lý, đối soát
* Không có thông tin khách hàng
* Vé giấy dễ làm giả, mất mát

### Quy trình quản lý của nhà xe:

Nhập lịch trình → Sổ/Excel → Bán vé → Ghi chép thủ công → Đối soát cuối ngày

Vấn đề:

* Nhiều sai sót do thủ công
* Khó theo dõi doanh thu real-time
* Không có dữ liệu phân tích
* Tốn nhân sự tại quầy
* Khó mở rộng kênh bán

## 1.2 Quy trình đề xuất (TO-BE)

### Quy trình đặt vé online:

Khách hàng → Web Browser → Tìm kiếm tuyến → Chọn chuyến → Chọn ghế

↓

Email/SMS ← Vé điện tử (QR) ← Thanh toán online ← Điền thông tin

Thời gian: 3-5 phút  
 Ưu điểm:

* Đặt vé mọi lúc, mọi nơi (24/7)
* Xem real-time ghế còn trống
* So sánh nhiều nhà xe, tìm kiếm nhiều tiện ích
* Thanh toán đa dạng, an toàn
* Vé điện tử với mã QR chống giả
* Lưu lịch sử đặt vé
* Thông báo tự động

### Quy trình quản lý của nhà xe:

Web Admin nhà xe → Tạo lịch trình → Hệ thống tự động → Dashboard real-time

↓

Khách đặt vé online

↓

Báo cáo tự động

Lợi ích:

* Tự động hóa hoàn toàn
* Báo cáo real-time
* Giảm nhân sự quầy vé
* Tăng doanh thu
* Dữ liệu khách hàng đầy đủ
* Phân tích và tối ưu

## 1.3 Phân tích stakeholder

Bên liên quan chính:

| Stakeholder | Vai trò | Nhu cầu |
| --- | --- | --- |
| Khách hàng | Người mua vé | Đặt vé nhanh, tiện, giá tốt |
| Nhà xe | Nhà cung cấp | Tăng doanh thu, giảm chi phí |
| Admin | Quản trị hệ thống | Quản lý hiệu quả, giám sát |
| Tài xế/Soát vé | Kiểm tra vé | Quét vé nhanh, chính xác |

Cơ hội:

* UX/UI ngày càng hiện đại hơn
* Công nghệ tốt hơn
* Hỗ trợ tốt hơn

# 2. MÔ TẢ BÀI TOÁN

## 2.1 Tổng quan

Hệ thống Đặt vé Xe khách Trực tuyến là một ứng dụng web cho phép khách hàng tìm kiếm, đặt vé, thanh toán và quản lý chuyến đi xe khách một cách dễ dàng và thuận tiện. Hệ thống kết nối khách hàng với các nhà xe, tạo nên một nền tảng minh bạch cho ngành vận tải hành khách đường bộ.

## 2.2 Mục tiêu

Mục tiêu chính:

1. Số hóa quy trình đặt vé: Chuyển từ đặt vé thủ công sang online hoàn toàn
2. Tăng trải nghiệm khách hàng: Đặt vé nhanh (< 5 phút), tiện lợi, minh bạch
3. Tăng hiệu quả cho nhà xe: Giảm chi phí vận hành, tăng doanh thu
4. Xây dựng nền tảng tin cậy: An toàn, bảo mật, ổn định với uptime

## 2.3 Phạm vi hệ thống

### Trong phạm vi:

Module chính:

1. Website responsive (desktop & mobile browser)
2. Tìm kiếm và đặt vé (search, select, book)
3. Thanh toán online (MoMo, VNPay, ZaloPay, ATM, Visa/Master)
4. Quản lý vé điện tử với mã QR
5. Web admin cho nhà xe (quản lý tuyến, xe, lịch trình, giá)
6. Web admin hệ thống (quản lý tổng thể)
7. Web soát vé (quét QR code để kiểm tra vé)
8. Thông báo (email, SMS)
9. Đánh giá và review
10. Báo cáo và thống kê

Tính năng bổ sung:

* Hủy/đổi vé online
* Chương trình khách hàng thân thiết
* Voucher và khuyến mãi
* Lịch sử đặt vé
* Hỗ trợ khách hàng (chat, hotline)

### Ngoài phạm vi:

* Mobile app native (iOS/Android)
* Hệ thống GPS tracking phần cứng
* Quản lý bảo trì xe
* Hệ thống kế toán chi tiết
* Booking khách sạn, tour

## 2.4 Chức năng chính

### A. Cho Khách hàng:

1. Tìm kiếm chuyến xe theo tuyến, ngày giờ
2. Xem thông tin chi tiết: giá, xe, tiện ích, đánh giá
3. Chọn ghế ngồi trên sơ đồ ghế real-time
4. Điền thông tin hành khách
5. Thanh toán online nhiều phương thức
6. Nhận vé điện tử có mã QR qua email/SMS
7. Quản lý vé đã đặt (xem, tải, hủy, đổi)
8. Đánh giá chuyến đi
9. Tích lũy điểm thưởng

### B. Cho Nhà xe:

1. Đăng ký và xác thực tài khoản
2. Quản lý thông tin nhà xe
3. Quản lý xe và sơ đồ ghế
4. Tạo và quản lý tuyến đường
5. Tạo lịch trình chuyến xe
6. Thiết lập giá vé và khuyến mãi
7. Xem danh sách booking real-time
8. Báo cáo doanh thu chi tiết
9. Quản lý nhân viên và tài xế
10. Trả lời đánh giá từ khách hàng

### C. Cho Tài xế/Soát vé:

1. Đăng nhập vào web quản lý chuyến
2. Xem danh sách hành khách của chuyến
3. Quét mã QR để xác thực vé
4. Đánh dấu hành khách đã lên xe
5. Báo cáo tình trạng chuyến xe

### D. Cho Admin hệ thống:

1. Quản lý người dùng (khách hàng, nhà xe)
2. Duyệt đăng ký nhà xe mới
3. Quản lý nội dung (banner, blog, FAQ)
4. Xử lý khiếu nại và hỗ trợ
5. Xem báo cáo tổng hợp
6. Cấu hình hệ thống
7. Quản lý voucher hệ thống

## 2.5 Công nghệ

Nền tảng: Web Application

Frontend:

* Framework: React 18
* UI Library: Tailwind CSS + Ant Design/Material-UI
* State Management: Redux/Zustand hoặc Pinia
* Build: Vite

Backend:

* Language: Node.js (Express)
* Database: MongoDb
* Cache: Redis 6
* Queue: Redis Queue

Infrastructure:

* Cloud: AWS/Azure/GCP
* Container: Docker
* Web Server: Nginx
* CDN: CloudFlare

Third-party:

* Payment: VNPay, MoMo, ZaloPay APIs
* SMS: VNPT SMS, Viettel SMS
* Email: SendGrid, AWS SES
* Maps: Google Maps API (nếu cần tracking)

# 3. XÁC ĐỊNH YÊU CẦU

## 3.1 Yêu cầu chức năng (Functional Requirements)

### FR-1: Quản lý Người dùng

FR-1.1: Đăng ký tài khoản

* Người dùng có thể đăng ký bằng email hoặc số điện thoại
* Yêu cầu xác thực qua email, SMS OTP
* Mật khẩu phải đủ mạnh (tối thiểu 8 ký tự)

FR-1.2: Đăng nhập

* Đăng nhập bằng email/SĐT + mật khẩu
* Hỗ trợ đăng nhập Google, Facebook OAuth
* Ghi nhớ đăng nhập (Remember me)

FR-1.3: Quản lý hồ sơ

* Xem và chỉnh sửa thông tin cá nhân
* Đổi mật khẩu
* Lưu danh sách hành khách thường đi

### FR-2: Tìm kiếm và Đặt vé

FR-2.1: Tìm kiếm chuyến xe

* Input: Điểm đi, điểm đến, ngày khởi hành
* Hiển thị danh sách chuyến xe với: nhà xe, giờ giữa 2 điểm di chuyển, giá, số ghế trống, đánh giá, các tiện ích hiện có(dưới dạng icon)
* Lọc theo: giá, giờ, nhà xe, loại xe
* Sắp xếp theo: giá, giờ, đánh giá

FR-2.2: Xem chi tiết chuyến

* Thông tin nhà xe (tên, logo, rating)
* Thông tin xe (loại xe, tiện ích)
* Lịch trình chi tiết (giờ đi, giờ đến, điểm dừng)
* Chính sách hủy/đổi vé
* Đánh giá từ khách hàng

FR-2.3: Chọn ghế

* Hiển thị sơ đồ ghế theo layout của xe
* Phân biệt: ghế trống (xám), đang chọn (vàng), đã đặt (xanh)
* Lock ghế tạm thời 15 phút khi khách đang đặt
* Cho phép chọn nhiều ghế cùng lúc

FR-2.4: Nhập thông tin

* Thông tin hành khách cho từng ghế: Họ tên, SĐT, CMND
* Chọn điểm đón và điểm trả chi tiết (nằm trong vùng điểm đi và điểm đến)
* Email nhận vé
* Ghi chú

FR-2.5: Áp dụng voucher

* Nhập mã voucher/giảm giá
* Validate và áp dụng tự động
* Hiển thị số tiền được giảm

### FR-3: Thanh toán

FR-3.1: Chọn phương thức thanh toán

* Ví điện tử: MoMo, ZaloPay, VNPay, ShopeePay
* Thẻ ATM nội địa
* Thẻ Visa/Mastercard/JCB
* Chuyển khoản ngân hàng
* Thanh toán khi lên xe (COD)

FR-3.2: Xử lý thanh toán

* Tích hợp với payment gateway
* Xác nhận thanh toán thành công/thất bại
* Tự động hoàn tiền nếu lỗi
* Lưu lịch sử giao dịch

FR-3.3: Hóa đơn điện tử

* Tạo hóa đơn tự động sau thanh toán
* Gửi hóa đơn qua email
* Cho phép tải hóa đơn (PDF)

### FR-4: Quản lý Vé

FR-4.1: Tạo vé điện tử

* Thông tin đầy đủ: mã vé, chuyến xe, ghế, hành khách, giá
* Mã QR chứa thông tin mã hóa
* Format PDF có thể in
* Gửi qua email

FR-4.2: Quản lý vé đã đặt

* Xem danh sách vé (sắp tới, đã đi, đã hủy)
* Xem chi tiết từng vé
* Tải lại vé khi cần
* Tìm kiếm vé theo mã, ngày, tuyến

FR-4.3: Hủy vé

* Kiểm tra chính sách hủy vé
* Xác nhận hủy vé
* Hoàn tiền tự động
* Gửi email xác nhận

FR-4.4: Đổi vé

* Tìm chuyến mới
* Tính chênh lệch giá + phí đổi
* Thanh toán/hoàn tiền chênh lệch
* Tạo vé mới, hủy vé cũ

### FR-5: Xác nhận Vé (Web Quản lý chuyến xe)

FR-5.1: Đăng nhập dành cho người quản lý chuyến xe

* Đăng nhập bằng mã số nhân viên và mật khẩu
* Xác nhận quyền được phân (tài xế chỉ được xem chuyến phân công còn người quản lý được chỉnh sửa và soát vé)
* Hiển thị chuyến xe được phân công
* Hiển thị thông tin chuyến xe: số hành khách, giờ, điểm đi điểm đến
* Cập nhật trạng thái chuyến xe: chưa bắt đầu, diễn ra, hoàn thành

FR-5.2: Xác thực vé

* Mở camera hoặc upload ảnh QR
* Tự động giải mã và xác thực
* Kiểm tra vé hợp lệ, đúng chuyến, chưa sử dụng, đã hủy, không hợp lệ
* Hiển thị thông tin: hành khách, ghế, điểm đón/trả
* Cảnh báo rõ ràng nếu vé không hợp lệ

FR-5.3: Đánh dấu đã sử dụng

* Xác nhận hành khách lên xe
* Đánh dấu vé đã sử dụng (không thể quét lại)
* Cập nhật danh sách hành khách

FR-5.4: Danh sách hành khách

* Xem tất cả hành khách của chuyến
* Phân biệt: đã lên xe / chưa lên xe
* Tìm kiếm theo tên, ghế
* Thống kê: đã lên/tổng số

### FR-6: Đánh giá và Review

FR-6.1: Đánh giá chuyến đi

* Đánh giá sau khi hoàn thành chuyến (1-5 sao)
* Đánh giá chi tiết: xe, tài xế và phụ xe, đúng giờ, dịch vụ
* Viết nhận xét (tùy chọn)

FR-6.2: Xem đánh giá

* Điểm trung bình của nhà xe
* Tổng số đánh giá
* Phân bố số sao
* Danh sách đánh giá (phân trang)
* Lọc và sắp xếp đánh giá

### FR-7: Quản trị Nhà xe

FR-7.1: Dashboard

* Tổng quan: doanh thu, số vé, tỷ lệ lấp đầy
* Biểu đồ doanh thu theo thời gian
* Chuyến xe sắp khởi hành
* Thông báo quan trọng

FR-7.2: Quản lý tuyến đường

* Thêm/sửa/xóa tuyến
* Thiết lập điểm đi, đến, điểm dừng
* Khoảng cách và thời gian dự kiến

FR-7.3: Quản lý xe

* Thêm/sửa/xóa xe
* Thông tin xe: biển số, loại xe, số chỗ
* Thiết lập sơ đồ ghế
* Tiện ích xe
* Trạng thái (hoạt động/bảo trì)

FR-7.4: Quản lý lịch trình

* Tạo chuyến xe (tuyến, xe, tài xế, người quản lý, giờ khởi hành)
* Thiết lập giá vé
* Sao chép lịch trình định kỳ
* Hủy/sửa chuyến xe

FR-7.5: Quản lý nhân viên

* Thêm nhân viên: Nhập thông tin cá nhân.
* Chỉnh sửa nhân viên
* Tìm kiếm và lọc: Tìm kiếm nhân viên theo tên, vai trò, mã nhân viên hoặc trạng thái.
* Phân quyền: Thiết lập quyền làm việc cho tài xế và người quản lý chuyến
* Theo dõi tình trạng làm việc: Hiển thị danh sách nhân viên đang hoạt động, tạm nghỉ, hoặc đã nghỉ việc.
* Liên kết với lịch trình: Cho phép xem lịch trình làm việc của từng tài xế hoặc phụ xe đã được phân công

FR-7.6: Quản lý giá và khuyến mãi

* Thiết lập bảng giá linh hoạt
* Tạo mã giảm giá (voucher)
* Thiết lập điều kiện áp dụng
* Thời gian hiệu lực

FR-7.6: Báo cáo

* Báo cáo doanh thu (ngày/tuần/tháng)
* Báo cáo tỷ lệ lấp đầy
* Báo cáo vé hủy
* Top tuyến đường
* Export Excel/PDF

### FR-8: Quản trị Hệ thống

FR-8.1: Quản lý người dùng

* Xem danh sách tất cả users
* Tìm kiếm, lọc users
* Khóa/mở khóa tài khoản
* Reset mật khẩu

FR-8.2: Quản lý nhà xe

* Duyệt đăng ký nhà xe mới
* Xác thực giấy tờ
* Phê duyệt/từ chối
* Tạm ngưng/khôi phục nhà xe

FR-8.3: Quản lý nội dung

* Upload và quản lý banner
* Thêm/sửa/xóa bài viết blog
* Quản lý FAQ
* SEO settings

FR-8.4: Xử lý khiếu nại

* Xem danh sách tickets
* Phân loại và ưu tiên
* Assign cho nhân viên
* Theo dõi và giải quyết
* Đóng ticket

FR-8.5: Báo cáo tổng hợp

* Dashboard tổng quan hệ thống
* Doanh thu toàn hệ thống
* Số lượng users, nhà xe, bookings
* Growth metrics
* Top performing routes/operators

### FR-9: Thông báo

FR-9.1: Email notifications

* Xác nhận đặt vé thành công
* Vé điện tử đính kèm
* Nhắc nhở trước giờ xuất bến
* Thông báo thay đổi lịch trình
* Khuyến mãi

FR-9.2: SMS notifications

* Mã OTP xác thực
* Link tải vé
* Nhắc nhở quan trọng
* Mã booking

### FR-10: Hỗ trợ và Tìm kiếm

FR-10.1: FAQ

* Danh sách câu hỏi thường gặp
* Phân danh mục
* Tìm kiếm trong FAQ

FR-10.2: Contact support

* Form liên hệ
* Chat online (nếu có)
* Hotline hiển thị rõ
* Email support

## 3.2 Yêu cầu phi chức năng (Non-Functional Requirements)

### NFR-1: Hiệu năng (Performance)

NFR-1.1: Thời gian tải trang chủ ≤ 2 giây  
 NFR-1.2: Thời gian tìm kiếm chuyến xe ≤ 3 giây  
 NFR-1.3: Xử lý thanh toán ≤ 5 giây  
 NFR-1.4: Hỗ trợ tối thiểu 5,000 người dùng đồng thời

### NFR-2: Tính khả dụng (Availability)

NFR-2.1: Uptime tối thiểu 99.9% (downtime ≤ 43 phút/tháng)  
 NFR-2.2: Hệ thống hoạt động 24/7/365  
 NFR-2.3: Recovery Time Objective (RTO) ≤ 1 giờ  
 NFR-2.4: Backup tự động hàng ngày

### NFR-3: Bảo mật (Security)

NFR-3.1: Mã hóa mật khẩu bằng bcrypt (cost factor ≥ 12)  
 NFR-3.2: HTTPS/TLS 1.3 cho mọi kết nối  
 NFR-3.3: Tuân thủ PCI-DSS cho thanh toán  
 NFR-3.4: Không lưu trữ thông tin thẻ tín dụng  
 NFR-3.5: Session timeout sau 30 phút không hoạt động  
 NFR-3.6: Rate limiting: 100 requests/phút/IP

### NFR-4: Khả năng sử dụng (Usability)

NFR-4.1: Quy trình đặt vé hoàn thành trong ≤ 5 phút  
 NFR-4.2: Không quá 5 bước để hoàn thành đặt vé  
 NFR-4.3: Giao diện responsive (desktop, tablet, mobile)  
 NFR-4.4: Hỗ trợ trình duyệt: Chrome, Firefox, Safari, Edg  
 NFR-4.5: Font size tối thiểu 14px trên mobile

### NFR-5: Khả năng bảo trì (Maintainability)

NFR-5.1: Code coverage ≥ 70%  
 NFR-5.2: Code tuân thủ coding standards  
 NFR-5.3: Tài liệu API đầy đủ  
 NFR-5.4: Logging đầy đủ cho debug

### NFR-6: Tương thích (Compatibility)

NFR-6.1: Responsive design cho màn hình từ 320px đến 4K  
 NFR-6.2: Progressive Web App capabilities  
 NFR-6.3: Tương thích với các payment gateway APIs

### NFR-7: Tuân thủ (Compliance)

NFR-7.1: Tuân thủ Luật Thương mại điện tử Việt Nam  
 NFR-7.2: Tuân thủ Luật Bảo vệ dữ liệu cá nhân  
 NFR-7.3: Tích hợp hóa đơn điện tử hợp pháp  
 NFR-7.4: Chính sách bảo mật và điều khoản sử dụng rõ ràng

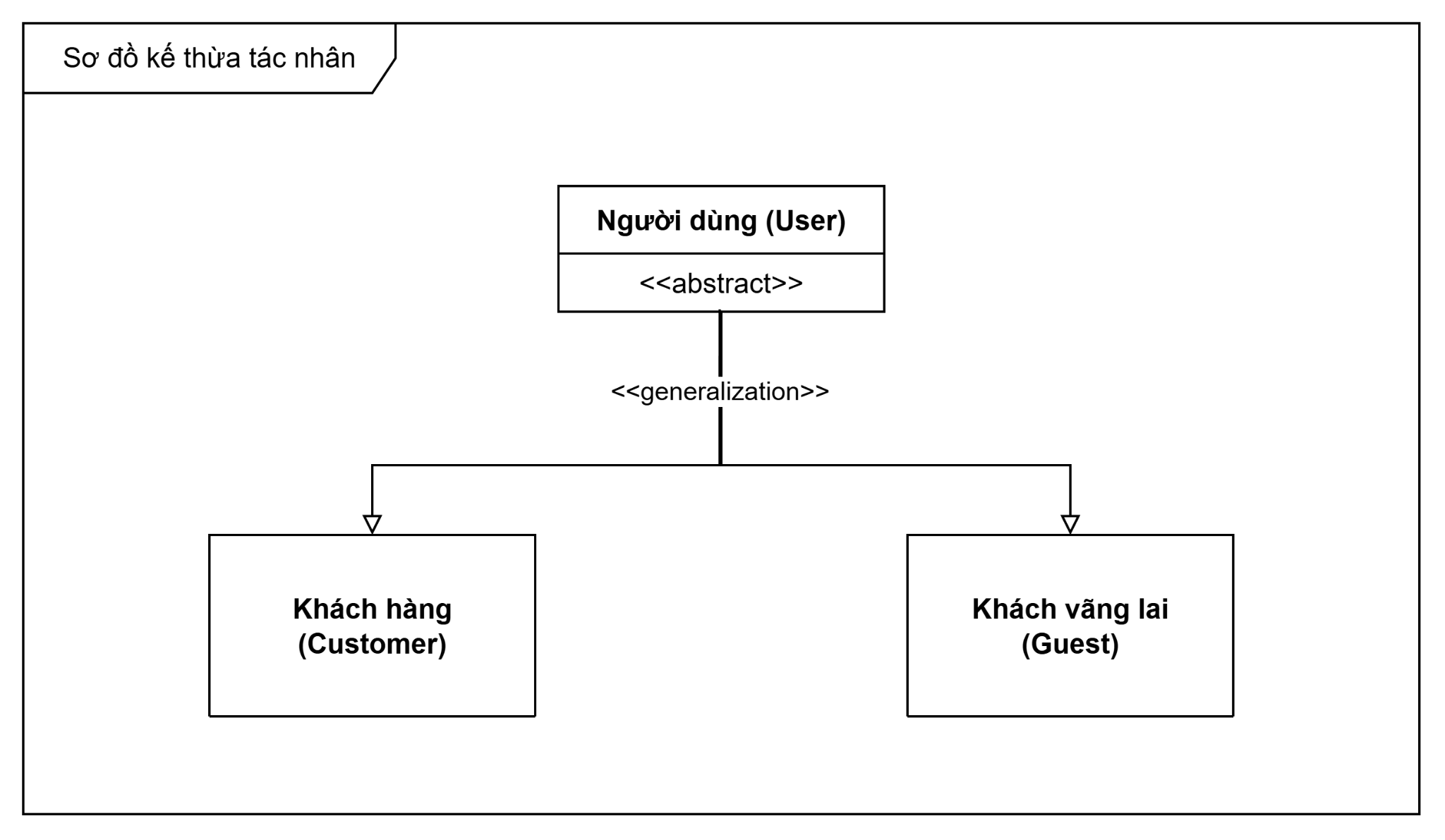
## 3.3 Ràng buộc kỹ thuật

Ngôn ngữ: Vietnamese và English  
 Múi giờ: GMT+7 (Vietnam)  
 Tiền tệ: VND (Vietnam Dong)  
 Payment: Tích hợp ít nhất 3 phương thức thanh toán  
SMS:  
 Email: Sử dụng dịch vụ email transactional  
 Hosting: Cloud-based (AWS/Azure/GCP)  
 Database: Relational database (Mongodb)

# 4. XÁC ĐỊNH TÁC NHÂN VÀ USE CASE

## 4.1 Xác định Tác nhân (Actors)

### 4.1.1 Cấu trúc kế thừa tác nhân

* 

### 4.1.2 Tác nhân cơ sở (Abstract Base Actor)

#### Người dùng (User) – *Tác nhân trừu tượng*

Mô tả:  
 Đại diện cho bất kỳ cá nhân nào sử dụng hệ thống để thực hiện thao tác đặt vé xe khách trực tuyến.

Chức năng chung:

* Tìm kiếm chuyến xe
* Xem thông tin chi tiết chuyến đi
* Đặt vé và chọn ghế ngồi
* Thanh toán trực tuyến
* Nhận vé điện tử có mã QR
* Quản lý vé đã đặt
* Thực hiện hủy hoặc đổi vé

Đặc điểm:

* Truy cập hệ thống thông qua trình duyệt web
* Nhận thông báo qua Email hoặc SMS
* Có thể đặt tối đa 6 ghế mỗi lần

### 4.1.3 Tác nhân chuyên biệt (Specialized Actors)

#### 1. Khách hàng (Customer) – *Kế thừa từ User*

Mô tả:  
 Người dùng đã đăng ký tài khoản và có thể đăng nhập để sử dụng đầy đủ các chức năng của hệ thống.

Kế thừa từ User:  
 Tất cả các chức năng cơ bản.

Chức năng mở rộng:

* Đăng ký và đăng nhập tài khoản
* Quản lý vé thông qua bảng điều khiển (Dashboard) cá nhân
* Tích lũy điểm thưởng sau mỗi chuyến đi
* Lưu danh sách hành khách thường đi (tối đa 5 người)
* Đánh giá và nhận xét chuyến đi
* Sử dụng voucher hoặc mã giảm giá thành viên
* Theo dõi lịch sử điểm thưởng

Phương thức xác thực: Email và Mật khẩu

Lợi ích:

* Không cần xác thực OTP cho mỗi lần đặt vé
* Được tích điểm và đổi quà
* Hưởng ưu đãi từ các voucher độc quyền
* Ưu tiên trong quá trình hỗ trợ khách hàng

#### 2. Khách vãng lai (Guest) – *Kế thừa từ User*

Mô tả:  
 Người dùng chưa đăng ký tài khoản, chỉ sử dụng hệ thống để đặt vé nhanh mà không cần đăng nhập.

Kế thừa từ User:  
 Tất cả các chức năng cơ bản.

Chức năng mở rộng:

* Tra cứu vé bằng Email hoặc Số điện thoại kết hợp OTP
* Quản lý vé thông qua mã truy cập tạm (UC-27)
* Lưu trữ lịch sử vé theo Email/SĐT trong 6 tháng
* Có thể nâng cấp lên tài khoản Customer bất kỳ lúc nào

Phương thức xác thực: Email hoặc SĐT + mã OTP (6 số, hiệu lực 5 phút)

Giới hạn:

* Không được tích điểm
* Không thể đánh giá chuyến đi
* Không lưu được danh sách hành khách thường đi
* Phiên làm việc (session) tự động hết hạn sau 30 phút
* 4.1.4 Các tác nhân khác

#### 3. Nhà xe (Bus Operator)

Mô tả:  
 Đơn vị cung cấp dịch vụ vận chuyển hành khách thông qua hệ thống.

Chức năng:

* Quản lý tuyến đường và lịch trình
* Quản lý phương tiện và sơ đồ ghế
* Tạo và cập nhật lịch trình chuyến xe
* Thiết lập giá vé và chương trình khuyến mãi
* Quản lý nhân sự (tài xế, quản lý chuyến)
* Xem và xuất báo cáo doanh thu

#### 4. Người quản lý chuyến (Trip Manager)

Mô tả:  
 Nhân viên nhà xe chịu trách nhiệm điều hành và kiểm soát các chuyến xe cụ thể.

Chức năng:

* Đăng nhập hệ thống quản lý chuyến
* Quét mã QR để xác thực vé
* Quản lý danh sách hành khách
* Cập nhật trạng thái chuyến xe

Quyền hạn: Có thể chỉnh sửa và soát vé.

#### 5. Tài xế (Driver)

Mô tả:  
 Người lái xe thuộc nhà xe, được phân công thực hiện các chuyến đi.

Chức năng:

* Đăng nhập hệ thống quản lý chuyến
* Xem thông tin chi tiết chuyến được phân công
* Xem danh sách hành khách trên chuyến
* Cập nhật trạng thái chuyến (chỉ xem, không chỉnh sửa)

Quyền hạn: Chỉ được quyền xem thông tin.

#### 6. Quản trị viên hệ thống (System Admin)

Mô tả:  
 Người chịu trách nhiệm quản lý và vận hành toàn bộ hệ thống.

Chức năng:

* Quản lý người dùng (Customer, Guest, Nhà xe)
* Duyệt đăng ký của các nhà xe mới
* Quản lý nội dung hiển thị (banner, blog, FAQ,...)
* Xử lý các khiếu nại của khách hàng
* Xem và thống kê báo cáo tổng hợp

Quyền hạn: Cao nhất trong hệ thống.

* 4.1.5 Tác nhân phụ (Secondary Actors)

#### 7. Hệ thống Thanh toán (Payment Gateway)

Mô tả:  
 Dịch vụ bên thứ ba được tích hợp để xử lý thanh toán trực tuyến.

Ví dụ: VNPay, MoMo, ZaloPay, VietQR.

#### 8. Hệ thống Email/SMS

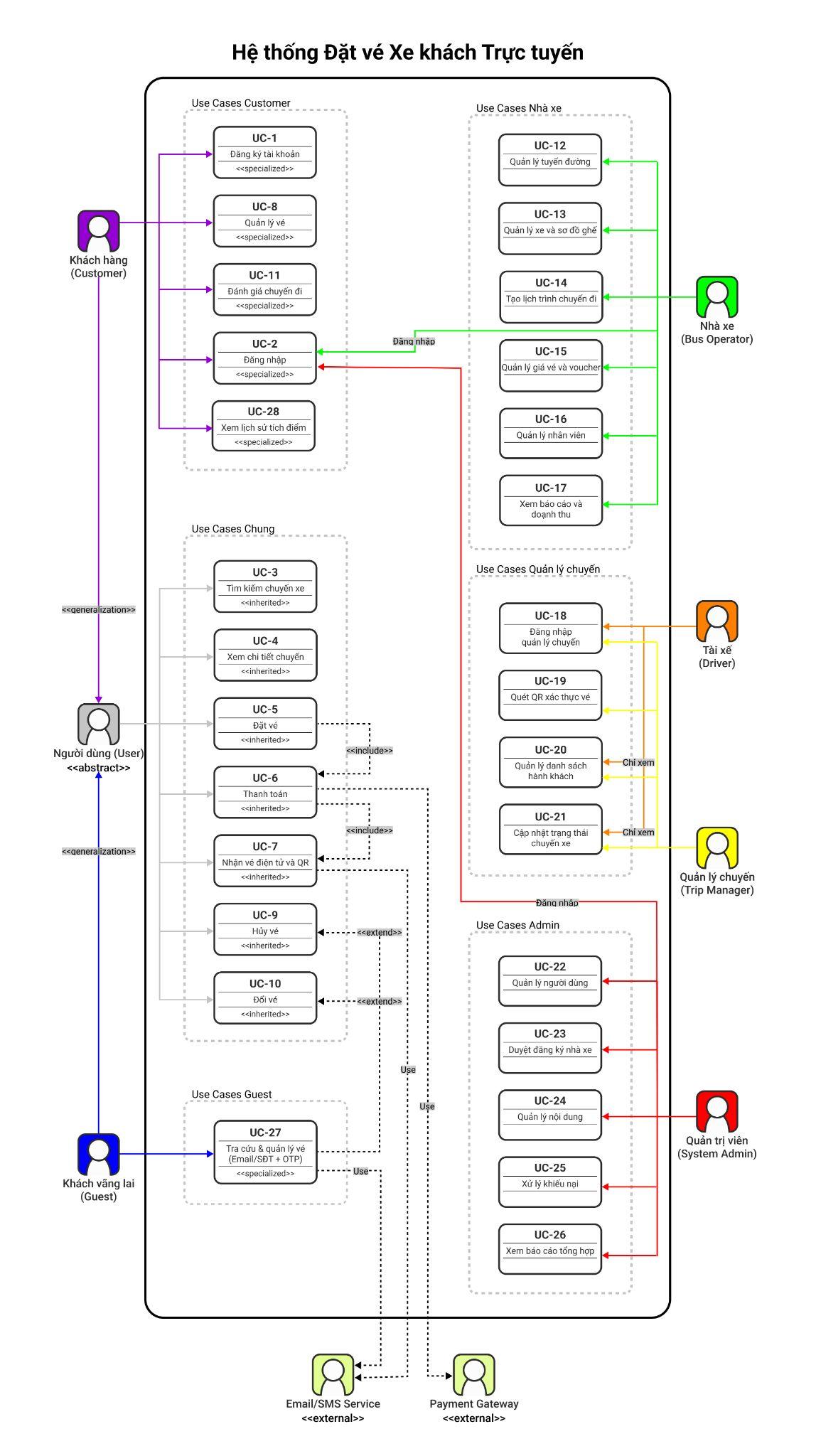
Mô tả:  
 Dịch vụ gửi thông báo tự động từ hệ thống đến người dùng.

Chức năng:

* Gửi mã OTP
* Gửi vé điện tử
* Gửi thông báo và xác nhận thanh toán

Ví dụ: SendGrid, AWS SES, VNPT SMS.

## 4.2 Sơ đồ Use Case Tổng quan



## 4.3 Danh sách Use Case theo Actor

### User (Abstract) - 7 use cases chung:

* UC-3: Tìm kiếm chuyến xe
* UC-4: Xem chi tiết chuyến
* UC-5: Đặt vé
* UC-6: Thanh toán
* UC-7: Nhận vé điện tử & QR
* UC-9: Hủy vé
* UC-10: Đổi vé

### Customer - 12 use cases (7 kế thừa + 5 riêng):

Kế thừa từ User: UC-3, UC-4, UC-5, UC-6, UC-7, UC-9, UC-10

Riêng:

* UC-1: Đăng ký tài khoản
* UC-2: Đăng nhập
* UC-8: Quản lý vé (Dashboard)
* UC-11: Đánh giá chuyến đi
* UC-28: Xem lịch sử tích điểm

### Guest - 8 use cases (7 kế thừa + 1 riêng):

Kế thừa từ User: UC-3, UC-4, UC-5, UC-6, UC-7, UC-9\*, UC-10\*

Riêng:

* UC-27: Tra cứu & quản lý vé (Email/SĐT + OTP)

\*UC-9, UC-10: Thực hiện qua UC-27, không qua Dashboard

### Nhà xe (Bus Operator) - 6 use cases:

* UC-12: Quản lý tuyến đường
* UC-13: Quản lý xe và sơ đồ ghế
* UC-14: Tạo lịch trình chuyến xe
* UC-15: Quản lý giá vé & voucher
* UC-16: Quản lý nhân viên
* UC-17: Xem báo cáo doanh thu

### Trip Manager - 4 use cases:

* UC-18: Đăng nhập quản lý chuyến
* UC-19: Quét QR xác thực vé
* UC-20: Quản lý danh sách hành khách
* UC-21: Cập nhật trạng thái chuyến xe

### Driver - 2 use cases (chỉ xem):

* UC-18: Đăng nhập quản lý chuyến (xem)
* UC-21: Cập nhật trạng thái chuyến xe (xem)

### System Admin - 5 use cases:

* UC-22: Quản lý người dùng
* UC-23: Duyệt đăng ký nhà xe
* UC-24: Quản lý nội dung
* UC-25: Xử lý khiếu nại
* UC-26: Xem báo cáo tổng hợp

Tổng cộng: 28 use cases

# 5. ĐẶC TẢ USE CASE CHI TIẾT

## 5.1. Nhóm Use Cases dành cho Customer

### UC-1: Đăng ký tài khoản

* **Tác nhân:** Customer (chưa đăng ký)
* **Điều kiện tiên quyết:** Người dùng chưa có tài khoản.
* **Luồng chính:**

1. Người dùng nhập email/SĐT, mật khẩu và họ tên.
2. Hệ thống gửi mã OTP qua email/SMS.
3. Người dùng nhập mã OTP để xác thực.
4. Hệ thống tạo tài khoản và gửi email chào mừng.

* **Luồng thay thế:**

2a. Email/SĐT đã tồn tại → Thông báo lỗi.

3a. OTP sai hoặc hết hạn → Yêu cầu gửi lại.

* **Kết quả:** Tài khoản được tạo và sẵn sàng đăng nhập.

### UC-2: Đăng nhập

* **Tác nhân:** Customer
* **Điều kiện tiên quyết:** Đã có tài khoản.
* **Luồng chính:**

1. Nhập email/SĐT và mật khẩu.
2. Hệ thống xác thực thông tin.
3. Tạo phiên đăng nhập và chuyển đến trang chủ.

* **Luồng thay thế:**

2a. Đăng nhập bằng Google/Facebook (OAuth).

2b. Sai mật khẩu → Thông báo lỗi (giới hạn 5 lần/15 phút).

2c. Quên mật khẩu → Gửi link reset qua email.

* **Kết quả:** Đăng nhập thành công, tạo session.

### UC-3: Tìm kiếm chuyến xe

* **Tác nhân:** User (Customer/Guest)
* **Điều kiện tiên quyết:** Không yêu cầu đăng nhập.
* **Luồng chính:**
  1. Nhập điểm đi, điểm đến, ngày khởi hành.
  2. Hệ thống truy vấn dữ liệu để tìm chuyến phù hợp.
  3. Hiển thị danh sách kết quả (nhà xe, giờ, giá, ghế trống, rating).
  4. Người dùng có thể áp dụng bộ lọc nếu cần.
* **Luồng thay thế:**

2a. Không có chuyến → Gợi ý các ngày gần nhất có chuyến.

* **Kết quả:** Hiển thị danh sách chuyến xe phù hợp.

### UC-4: Xem chi tiết chuyến xe

* **Tác nhân:** User (Customer/Guest)
* **Điều kiện tiên quyết:** Có danh sách chuyến xe.
* **Luồng chính:**
  1. Chọn một chuyến xe từ kết quả tìm kiếm.
  2. Hiển thị thông tin chi tiết: nhà xe, xe, lịch trình, tiện ích, chính sách.
  3. Hiển thị đánh giá từ khách hàng.
  4. Hiển thị sơ đồ ghế real-time.
* **Kết quả:** Người dùng xem được đầy đủ thông tin chuyến.

### UC-5: Đặt vé

* **Tác nhân:** User (Customer/Guest)
* **Điều kiện tiên quyết:** Đã chọn chuyến xe.
* **Luồng chính:**
  1. Chọn ghế trên sơ đồ (tối đa 6 ghế).
  2. Hệ thống khóa ghế tạm thời trong 15 phút.
  3. Nhập thông tin hành khách.
  4. Chọn điểm đón và điểm trả.
  5. Nhập email nhận vé và ghi chú.
  6. Nhập mã voucher (nếu có).
  7. Xác nhận và chuyển sang thanh toán.
* **Luồng thay thế:**

1a. Ghế đã được chọn trước đó → Cập nhật sơ đồ.

2a. Hết thời gian giữ ghế → Release và yêu cầu chọn lại.

6a. Voucher không hợp lệ → Thông báo lỗi.

* **Kết quả:** Booking tạm thời được tạo, chuyển sang thanh toán.

### UC-6: Thanh toán

* **Tác nhân:** User (Customer/Guest)
* **Điều kiện tiên quyết:** Đã hoàn thành bước đặt vé.
* **Luồng chính:**
  1. Chọn phương thức thanh toán.
  2. Chuyển đến Payment Gateway.
  3. Người dùng thực hiện thanh toán.
  4. Payment Gateway trả kết quả về hệ thống.
  5. Hệ thống cập nhật trạng thái booking.
* **Luồng thay thế:**

4a. Thanh toán thất bại → Giữ booking 10 phút cho phép thử lại.

4b. Thanh toán thành công → Tạo vé điện tử.

* **Kết quả:** Thanh toán hoàn tất hoặc chờ thử lại.

### UC-7: Nhận vé điện tử & mã QR

* **Tác nhân:** User (Customer/Guest)
* **Điều kiện tiên quyết:** Thanh toán thành công.
* **Luồng chính:**
  1. Hệ thống tạo vé điện tử (PDF) và mã QR.
  2. Gửi vé qua email.
  3. Gửi SMS chứa link tải vé và mã booking.
  4. Lưu vé vào lịch sử giao dịch.
* **Kết quả:** Người dùng nhận được vé điện tử.

### UC-8: Quản lý vé

* **Tác nhân:** Customer
* **Điều kiện tiên quyết:** Đã đăng nhập.
* **Luồng chính:**
  1. Truy cập trang quản lý vé.
  2. Xem danh sách vé (sắp tới/đã đi/đã hủy).
  3. Tìm kiếm và lọc theo mã, ngày, tuyến.
  4. Xem chi tiết, tải lại, hủy hoặc đổi vé.
* **Kết quả:** Người dùng quản lý được vé của mình.

### UC-9: Hủy vé

* **Tác nhân:** Customer/Guest
* **Điều kiện tiên quyết:** Vé hợp lệ và trong thời hạn hủy.
* **Luồng chính:**
  1. Chọn vé cần hủy.
  2. Hệ thống kiểm tra chính sách hủy.
  3. Hiển thị số tiền hoàn lại.
  4. Xác nhận hủy.
  5. Cập nhật trạng thái và hoàn tiền.
  6. Gửi email xác nhận.
* **Luồng thay thế:**

2a. Quá hạn → Từ chối.

* **Kết quả:** Vé được hủy và hoàn tiền.

### UC-10: Đổi vé

* **Tác nhân:** Customer/Guest
* **Điều kiện tiên quyết:** Vé hợp lệ và trong thời hạn đổi.
* **Luồng chính:**
  1. Chọn vé cần đổi.
  2. Tìm chuyến mới.
  3. Chọn ghế mới.
  4. Tính chênh lệch giá và phí đổi.
  5. Thanh toán/hoàn tiền chênh lệch.
  6. Hủy vé cũ và tạo vé mới.
  7. Gửi email xác nhận.
* **Luồng thay thế:**

2a. Không có chuyến phù hợp → Gợi ý hủy vé.

* **Kết quả:** Vé mới được tạo.

### UC-11: Đánh giá chuyến đi

* **Tác nhân:** Customer
* **Điều kiện tiên quyết:** Chuyến đi đã hoàn thành.
* **Luồng chính:**
  1. Hệ thống gửi email mời đánh giá.
  2. Người dùng đánh giá (1–5 sao).
  3. Viết nhận xét (tùy chọn).
  4. Gửi đánh giá.
  5. Hệ thống cộng điểm thưởng.
* **Kết quả:** Đánh giá được lưu.

### UC-28: Xem lịch sử tích điểm

* **Tác nhân:** Customer
* **Điều kiện tiên quyết:** Đã đăng nhập.
* **Luồng chính:**
  1. Truy cập trang điểm thưởng.
  2. Xem tổng điểm.
  3. Xem lịch sử tích/tiêu điểm.
  4. Xem các phần quà có thể đổi.
* **Kết quả:** Hiển thị lịch sử điểm thưởng.

## 5.2. Nhóm Use Cases dành cho Guest

### UC-27: Tra cứu & quản lý vé qua OTP

* **Tác nhân:** Guest
* **Điều kiện tiên quyết:** Đã đặt vé bằng email/SĐT.
* **Luồng chính:**
  1. Nhập email/SĐT.
  2. Hệ thống gửi OTP.
  3. Nhập OTP để xác thực.
  4. Hiển thị danh sách vé tương ứng.
  5. Cho phép xem, tải, hủy hoặc đổi vé.
* **Luồng thay thế:**

3a. OTP sai → Cho thử lại (tối đa 3 lần).

* **Kết quả:** Guest truy cập được các vé của mình.

## 5.3. Nhóm Use Cases dành cho Bus Operator

### UC-12: Quản lý tuyến đường

* **Tác nhân:** Bus Operator
* **Điều kiện tiên quyết:** Đăng nhập với quyền nhà xe.
* **Luồng chính:**
  1. Thêm/sửa/xóa tuyến.
  2. Thiết lập điểm đi, điểm đến và điểm dừng.
  3. Nhập thông tin khoảng cách, thời gian dự kiến.
  4. Lưu tuyến đường.
* **Kết quả:** Tuyến đường được quản lý.

### UC-13: Quản lý xe & sơ đồ ghế

* **Tác nhân:** Bus Operator
* **Điều kiện tiên quyết:** Đăng nhập.
* **Luồng chính:**
  1. Thêm/sửa/xóa xe.
  2. Thiết lập sơ đồ ghế.
  3. Thêm tiện ích xe.
  4. Cập nhật trạng thái xe.
* **Kết quả:** Xe và layout ghế được cấu hình.

### UC-14: Tạo lịch trình chuyến xe

* **Tác nhân:** Bus Operator
* **Điều kiện tiên quyết:** Có tuyến và xe.
* **Luồng chính:**
  1. Chọn tuyến.
  2. Chọn xe.
  3. Chọn tài xế và quản lý chuyến.
  4. Nhập giờ đi – giờ đến dự kiến.
  5. Thiết lập giá vé.
  6. Chọn lịch lặp lại (nếu có).
  7. Lưu lịch trình.
* **Luồng thay thế:**

3a. Tài xế/xe xung đột lịch → Cảnh báo.

* **Kết quả:** Tạo lịch trình thành công.

### UC-15: Quản lý giá vé & voucher

* **Tác nhân:** Bus Operator
* **Luồng chính:**
  1. Thiết lập bảng giá.
  2. Tạo voucher.
  3. Kích hoạt/vô hiệu hóa.
  4. Xem báo cáo sử dụng.
* **Kết quả:** Giá vé và voucher được quản trị.

### UC-16: Quản lý nhân viên

* **Tác nhân:** Bus Operator
* **Luồng chính:**
  1. Thêm nhân viên.
  2. Phân quyền.
  3. Xem lịch trình làm việc.
  4. Cập nhật trạng thái.
  5. Tìm kiếm/lọc.
* **Kết quả:** Quản lý toàn bộ nhân sự của nhà xe.

### UC-17: Xem báo cáo doanh thu

* **Tác nhân:** Bus Operator
* **Luồng chính:**
  1. Chọn khoảng thời gian.
  2. Hệ thống hiển thị các chỉ số:
     + Tổng doanh thu
     + Số vé bán
     + Tỷ lệ lấp đầy
     + Top tuyến
     + Số vé hủy
  3. Xuất báo cáo PDF/Excel.
* **Kết quả:** Báo cáo doanh thu chi tiết.

## 5.4. Nhóm Use Cases dành cho Trip Manager và Driver

### UC-18: Đăng nhập quản lý chuyến

* **Tác nhân:** Trip Manager, Driver
* **Luồng chính:**
  1. Nhập mã nhân viên và mật khẩu.
  2. Xác thực thông tin và quyền.
  3. Hiển thị các chuyến được phân công.
  4. Xem thông tin chuyến.
* **Kết quả:** Truy cập vào danh sách chuyến phân công.

### UC-19: Quét QR xác thực vé

* **Tác nhân:** Trip Manager
* **Điều kiện tiên quyết:** Đã đăng nhập và được phân công chuyến.
* **Luồng chính:**
  1. Mở camera hoặc upload ảnh QR.
  2. Quét mã QR trên vé.
  3. Hệ thống giải mã và xác thực.
  4. Kiểm tra tính hợp lệ.
  5. Hiển thị thông tin hành khách.
  6. Xác nhận khách lên xe.
* **Luồng thay thế:**

4a. Vé không hợp lệ.

4b. Vé đã sử dụng.

* **Kết quả:** Vé được xác thực thành công.

### UC-20: Quản lý danh sách hành khách

* **Tác nhân:** Trip Manager
* **Luồng chính:**
  1. Xem danh sách hành khách.
  2. Theo dõi người đã lên/chưa lên xe.
  3. Tìm kiếm hành khách.
  4. Đánh dấu đã lên xe.
  5. Xem tổng số đã lên.
* **Kết quả:** Danh sách hành khách được quản lý.

### UC-21: Cập nhật trạng thái chuyến

* **Tác nhân:** Trip Manager (cập nhật), Driver (xem).
* **Luồng chính:**
  1. Xem trạng thái chuyến.
  2. Cập nhật trạng thái.
  3. Hệ thống gửi thông báo cho hành khách.
* **Kết quả:** Trạng thái chuyến được cập nhật.

## 5.5. Nhóm Use Cases dành cho System Admin

### UC-22: Quản lý người dùng

* **Tác nhân:** System Admin
* **Luồng chính:**
  1. Xem danh sách người dùng.
  2. Tìm kiếm/lọc.
  3. Xem chi tiết.
  4. Khóa/mở khóa tài khoản.
  5. Reset mật khẩu.
* **Kết quả:** Người dùng được quản trị.

### UC-23: Duyệt đăng ký nhà xe

* **Tác nhân:** System Admin
* **Luồng chính:**
  1. Xem danh sách yêu cầu đăng ký.
  2. Kiểm tra giấy tờ.
  3. Phê duyệt hoặc từ chối.
  4. Gửi thông báo kết quả.
* **Kết quả:** Nhà xe được duyệt hoặc từ chối.

### UC-24: Quản lý nội dung hệ thống

* **Tác nhân:** System Admin
* **Luồng chính:**
  1. Upload quản lý banner.
  2. Thêm/sửa/xóa bài viết blog.
  3. Quản lý FAQ.
  4. Cấu hình SEO.
* **Kết quả:** Nội dung hệ thống được cập nhật.

### UC-25: Xử lý khiếu nại

* **Tác nhân:** System Admin
* **Luồng chính:**
  1. Xem danh sách tickets.
  2. Phân loại và ưu tiên.
  3. Giao ticket cho nhân viên.
  4. Theo dõi tiến độ.
  5. Đóng ticket.
  6. Gửi thông báo cho khách hàng.
* **Kết quả:** Khiếu nại được xử lý đầy đủ.

### UC-26: Xem báo cáo tổng hợp hệ thống

* **Tác nhân:** System Admin
* **Luồng chính:**
  1. Truy cập dashboard.
  2. Xem các chỉ số tổng hợp.
  3. Chọn khoảng thời gian tùy chỉnh.
  4. Xuất báo cáo.
* **Kết quả:** Hiển thị báo cáo tổng quan toàn hệ thống.

# 6. Xác Định Lớp và Quan Hệ

## **6.1.** **Xác Định Các Lớp**

## 6.1.1 Nhóm Lớp Người Dùng (User Domain)

### 1. User Model

* **Thuộc tính:**
  + \_id: ObjectId (PK - auto generated by MongoDB)
  + email: String (required, unique, lowercase)
  + phone: String (required, unique)
  + password: String (required, minlength: 8, hashed with bcrypt, select: false)
  + fullName: String (required)
  + dateOfBirth: Date (optional)
  + gender: Enum ('male', 'female', 'other')
  + avatar: String (URL)
  + role: Enum ('customer', 'admin') - default: 'customer'
  + **OAuth:**
    - googleId: String
    - facebookId: String
  + **Verification:**
    - isEmailVerified: Boolean (default: false)
    - isPhoneVerified: Boolean (default: false)
    - emailVerificationToken: String
    - phoneVerificationOTP: String
    - otpExpires: Date
  + **Security:**
    - passwordResetToken: String
    - passwordResetExpires: Date
    - lastLogin: Date
  + **Preferences:**
    - savedPassengers: Array of embedded documents
      * fullName: String
      * phone: String
      * idCard: String
  + **Loyalty:**
    - loyaltyTier: Enum ('bronze', 'silver', 'gold', 'platinum') - default: 'bronze'
    - totalPoints: Number (default: 0)
  + **Status:**
    - isActive: Boolean (default: true)
    - isBlocked: Boolean (default: false)
  + createdAt: DateTime (auto - timestamps)
  + updatedAt: DateTime (auto - timestamps)
* **Indexes:**
  + email: 1
  + phone: 1
  + googleId: 1
* **Phương thức:**
  + Pre-save hook: Hash password với bcrypt (salt rounds: 12)
  + comparePassword(candidatePassword): Promise
  + toJSON(): Object (ẩn các field nhạy cảm)
* **Ghi chú:**
  + Trong thiết kế ban đầu có tách Customer và Guest, nhưng trong code hiện tại User model xử lý cả hai roles thông qua field role và logic xác thực

### 2. BusOperator Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + companyName: String (required, unique)
  + email: String (required, unique, lowercase)
  + phone: String (required)
  + password: String (required, minlength: 8, hashed, select: false)
  + **Business Info:**
    - businessLicense: String (required)
    - taxCode: String (required)
    - logo: String (URL)
    - description: String
    - website: String
  + **Address (embedded object):**
    - street: String
    - ward: String
    - district: String
    - city: String
    - country: String (default: 'Vietnam')
  + **Bank Info (embedded object):**
    - bankName: String
    - accountNumber: String
    - accountHolder: String
  + **Approval:**
    - verificationStatus: Enum ('pending', 'approved', 'rejected') - default: 'pending'
    - verifiedAt: Date
    - verifiedBy: ObjectId (ref: 'User')
    - rejectionReason: String
  + **Rating:**
    - averageRating: Number (default: 0, min: 0, max: 5)
    - totalReviews: Number (default: 0)
  + **Statistics:**
    - totalTrips: Number (default: 0)
    - totalRevenue: Number (default: 0)
  + **Commission:**
    - commissionRate: Number (default: 5, min: 0, max: 100) // Percentage
  + **Status:**
    - isActive: Boolean (default: true)
    - isSuspended: Boolean (default: false)
    - suspensionReason: String
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + companyName: 1
  + email: 1
  + verificationStatus: 1
* **Phương thức:**
  + Pre-save hook: Hash password với bcrypt (salt rounds: 12)
  + comparePassword(candidatePassword): Promise
  + toJSON(): Object (ẩn password)

## 6.1.2 Nhóm Lớp Tuyến Đường và Lịch Trình (Route & Schedule Domain)

### 3. Route Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + operatorId: ObjectId (ref: 'BusOperator', required)
  + routeName: String (required)
  + routeCode: String (required, unique, uppercase)
  + **Origin (embedded locationSchema):**
    - city: String (required)
    - province: String (required)
    - station: String
    - address: String
    - coordinates: { lat: Number, lng: Number }
  + **Destination (embedded locationSchema):**
    - city: String (required)
    - province: String (required)
    - station: String
    - address: String
    - coordinates: { lat: Number, lng: Number }
  + **Pickup & Dropoff Points (embedded pointSchema):**
    - pickupPoints: Array of { name: String, address: String, coordinates: {lat, lng} }
    - dropoffPoints: Array of { name: String, address: String, coordinates: {lat, lng} }
  + **Route Details:**
    - distance: Number (required, min: 0) // km
    - estimatedDuration: Number (required, min: 0) // minutes
  + **Status:**
    - isActive: Boolean (default: true)
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + routeCode: 1
  + operatorId: 1
  + 'origin.city': 1, 'destination.city': 1
* **Ghi chú:**
  + Location và Point được implement dưới dạng embedded schema, không phải separate collection

### 4. Bus Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + operatorId: ObjectId (ref: 'BusOperator', required)
  + busNumber: String (required, unique, uppercase) // License plate
  + busType: Enum ('limousine', 'sleeper', 'seater', 'double\_decker') - required
  + **Seat Configuration (embedded seatLayout):**
    - seatLayout.floors: Number (required, min: 1, max: 2)
    - seatLayout.rows: Number (required)
    - seatLayout.columns: Number (required)
    - seatLayout.layout: Array of Array of String (required) // 2D array representing seat map
  + totalSeats: Number (required, min: 1)
  + **Amenities:**
    - amenities: Array of Enum ('wifi', 'ac', 'toilet', 'water', 'blanket', 'tv', 'usb\_charger', 'reading\_light')
  + **Images:**
    - images: Array of String (URLs)
  + **Status:**
    - isActive: Boolean (default: true)
    - maintenanceStatus: Enum ('good', 'maintenance', 'repair') - default: 'good'
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + operatorId: 1
  + busNumber: 1
* **Ghi chú:**
  + SeatLayout được implement dưới dạng embedded object, không phải separate collection
  + Seat configuration được lưu trữ dưới dạng 2D array trong field layout

### 5. Trip Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + operatorId: ObjectId (ref: 'BusOperator', required)
  + routeId: ObjectId (ref: 'Route', required)
  + busId: ObjectId (ref: 'Bus', required)
  + tripCode: String (required, unique, uppercase)
  + **Schedule:**
    - departureTime: Date (required)
    - arrivalTime: Date (required)
  + **Pricing:**
    - basePrice: Number (required, min: 0)
  + **Seat Availability:**
    - availableSeats: Number (required)
    - occupiedSeats: Array of String // Seat numbers
    - lockedSeats: Array of embedded objects
      * seatNumber: String
      * lockedUntil: Date
      * sessionId: String
  + **Staff Assignment:**
    - driver: ObjectId (ref: 'Staff') ⚠️ *Staff model chưa tạo*
    - tripManager: ObjectId (ref: 'Staff') ⚠️ *Staff model chưa tạo*
  + **Status:**
    - status: Enum ('scheduled', 'boarding', 'in\_progress', 'completed', 'cancelled') - default: 'scheduled'
    - cancellationReason: String
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + tripCode: 1
  + operatorId: 1, departureTime: 1
  + routeId: 1, departureTime: 1
  + departureTime: 1, status: 1
* **Phương thức:**
  + isSeatAvailable(seatNumber): Boolean
  + lockSeats(seatNumbers, sessionId, durationMinutes = 15): Promise
  + releaseLocks(sessionId): Promise
  + occupySeats(seatNumbers): Promise
* **Ghi chú:**
  + SeatLock được implement dưới dạng embedded array, không phải separate collection
  + Staff model chưa được tạo, cần implement trước khi deploy production

## 6.1.3 Nhóm Lớp Đặt Vé (Booking Domain)

### 6. Booking Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + bookingCode: String (required, unique, uppercase) // Format: BKYYYYMMDD####
  + customerId: ObjectId (ref: 'User', required)
  + tripId: ObjectId (ref: 'Trip', required)
  + operatorId: ObjectId (ref: 'BusOperator', required)
  + **Seats (array of embedded objects):**
    - seats: Array of
      * seatNumber: String (required)
      * passenger: embedded object
        + fullName: String (required)
        + phone: String (required)
        + idCard: String
  + **Pickup & Dropoff (embedded objects):**
    - pickupPoint: { name: String, address: String, coordinates: {lat, lng} }
    - dropoffPoint: { name: String, address: String, coordinates: {lat, lng} }
  + **Pricing:**
    - subtotal: Number (required, min: 0)
    - discount: Number (default: 0, min: 0)
    - totalAmount: Number (required, min: 0)
  + **Voucher:**
    - voucherId: ObjectId (ref: 'Voucher') ⚠️ *Voucher model chưa tạo*
    - voucherCode: String
  + **Contact:**
    - contactEmail: String (required)
    - contactPhone: String (required)
    - notes: String
  + **Status:**
    - status: Enum ('pending', 'confirmed', 'cancelled', 'completed') - default: 'pending'
  + **Cancellation:**
    - cancellationReason: String
    - cancelledAt: Date
    - refundAmount: Number (default: 0)
    - refundStatus: Enum ('pending', 'processed', 'failed')
  + **Check-in:**
    - checkedInSeats: Array of String
    - checkedInAt: Date
    - checkedInBy: ObjectId (ref: 'Staff') ⚠️ *Staff model chưa tạo*
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + bookingCode: 1
  + customerId: 1
  + tripId: 1
  + status: 1
  + createdAt: -1
* **Static Methods:**
  + generateBookingCode(): Promise // Generates unique booking code
* **Ghi chú:**
  + Passenger được implement dưới dạng embedded object trong seats array, không phải separate collection
  + Voucher model chưa được tạo

### 7. Ticket Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + ticketCode: String (required, unique, uppercase) // Format: TKYYYYMMDD####
  + bookingId: ObjectId (ref: 'Booking', required)
  + customerId: ObjectId (ref: 'User', required)
  + tripId: ObjectId (ref: 'Trip', required)
  + **Seat & Passenger Info (embedded):**
    - seatNumber: String (required)
    - passenger: embedded object
      * fullName: String (required)
      * phone: String (required)
      * idCard: String
  + **QR Code:**
    - qrCode: String (required) // Base64 image
    - qrData: String (required) // Encrypted data for verification
  + **Ticket PDF:**
    - ticketPDF: String // URL to PDF file
  + **Validation:**
    - isValid: Boolean (default: true)
    - isUsed: Boolean (default: false)
    - usedAt: Date
    - validatedBy: ObjectId (ref: 'Staff') ⚠️ *Staff model chưa tạo*
  + **Trip Details (denormalized for quick access):**
    - tripDetails: embedded object
      * routeName: String
      * origin: String
      * destination: String
      * departureTime: Date
      * busNumber: String
      * operatorName: String
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + ticketCode: 1
  + bookingId: 1
  + qrData: 1
  + customerId: 1
* **Static Methods:**
  + generateTicketCode(): Promise
* **Ghi chú:**
  + Trip details được denormalize để tránh join khi hiển thị vé
  + QR code data được mã hóa để bảo mật

## 6.1.4 Nhóm Lớp Thanh Toán (Payment Domain)

### 8. Payment Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + transactionId: String (required, unique) // Format: TXN{timestamp}{random}
  + bookingId: ObjectId (ref: 'Booking', required)
  + customerId: ObjectId (ref: 'User', required)
  + **Amount:**
    - amount: Number (required, min: 0)
    - currency: String (default: 'VND')
  + **Payment Method:**
    - paymentMethod: Enum ('momo', 'vnpay', 'zalopay', 'shopeepay', 'atm', 'visa', 'mastercard', 'cod') - required
  + **Gateway Info:**
    - gatewayTransactionId: String
    - gatewayResponse: Mixed (object chứa response từ payment gateway)
  + **Status:**
    - status: Enum ('pending', 'success', 'failed', 'refunded') - default: 'pending'
  + **Refund:**
    - refundAmount: Number (default: 0)
    - refundedAt: Date
    - refundReason: String
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + transactionId: 1
  + bookingId: 1
  + customerId: 1
  + status: 1
  + createdAt: -1
* **Static Methods:**
  + generateTransactionId(): Promise

#### 

### 9. Staff Model

* **Thuộc tính:**
  + \_id: ObjectId (PK - auto generated by MongoDB)
  + operatorId: ObjectId (ref: 'BusOperator', required)
  + employeeCode: String (required, unique, uppercase)
  + fullName: String (required)
  + phone: String (required)
  + email: String (lowercase)
  + dateOfBirth: Date
  + avatar: String (URL)
  + address: String
  + **Role & Status:**
    - role: Enum ('driver', 'tripManager', 'ticketChecker') - required
    - status: Enum ('active', 'onLeave', 'resigned') - default: 'active'
  + **Authentication:**
    - password: String (required, minlength: 8, hashed with bcrypt, select: false)
    - lastLogin: Date
  + **Driver Specific:**
    - licenseNumber: String // Giấy phép lái xe
    - licenseExpiry: Date
    - experienceYears: Number (min: 0)
  + **Statistics:**
    - totalTripsCompleted: Number (default: 0)
    - averageRating: Number (default: 0, min: 0, max: 5)
    - totalRatings: Number (default: 0)
  + createdAt: DateTime (auto - timestamps)
  + updatedAt: DateTime (auto - timestamps)
* **Indexes:**
  + employeeCode: 1
  + operatorId: 1, status: 1
  + role: 1
  + email: 1
* **Phương thức:**
  + Pre-save hook: Hash password với bcrypt (salt rounds: 12)
  + comparePassword(candidatePassword): Promise
  + toJSON(): Object (ẩn password và các field nhạy cảm)
  + Instance method: isAvailable(date): Promise - Kiểm tra staff có rảnh vào ngày cụ thể
  + Static method: findAvailableStaff(operatorId, role, date): Promise - Tìm nhân viên rảnh
* **Ghi chú:**
  + Model này được reference bởi Trip (driver, tripManager), Booking (checkedInBy), và Ticket (validatedBy)
  + Cần validate licenseExpiry cho role 'driver'

### 10. Voucher Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + code: String (required, unique, uppercase, minlength: 4, maxlength: 20)
  + title: String (required) // Tên voucher hiển thị
  + description: String
  + **Discount Configuration:**
    - discountType: Enum ('percentage', 'fixed') - required
    - discountValue: Number (required, min: 0)
    - maxDiscount: Number (min: 0) // Giảm tối đa (cho percentage)
    - minOrderValue: Number (default: 0, min: 0) // Giá trị đơn hàng tối thiểu
  + **Validity:**
    - validFrom: Date (required)
    - validTo: Date (required)
  + **Usage Limits:**
    - usageLimit: Number (min: 1) // null = unlimited
    - usedCount: Number (default: 0, min: 0)
    - usageLimitPerUser: Number (default: 1, min: 1)
  + **Scope:**
    - operatorId: ObjectId (ref: 'BusOperator') // null = system-wide voucher
    - applicableRoutes: Array of ObjectId (ref: 'Route') // Áp dụng cho tuyến cụ thể
    - applicableBusTypes: Array of Enum ('limousine', 'sleeper', 'seater', 'double\_decker')
  + **Status:**
    - isActive: Boolean (default: true)
    - isPublic: Boolean (default: true) // false = voucher riêng tư (qua email)
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + code: 1
  + operatorId: 1, isActive: 1
  + validFrom: 1, validTo: 1
  + isActive: 1, validFrom: 1, validTo: 1
* **Phương thức:**
  + Instance method: isValid(currentDate = new Date()): Boolean - Kiểm tra voucher còn hiệu lực
  + Instance method: canBeUsed(userId, orderValue, routeId, busType): Promise - Kiểm tra có thể sử dụng
  + Instance method: calculateDiscount(orderValue): Number - Tính số tiền giảm
  + Instance method: incrementUsage(): Promise - Tăng số lần sử dụng
  + Static method: findValidVouchers(operatorId, routeId, busType): Promise - Tìm voucher khả dụng
  + Pre-save hook: Validate validTo > validFrom
* **Ghi chú:**
  + Được reference bởi Booking model
  + Cần track usage per user trong separate collection nếu cần chi tiết

#### 

### 11. Review Model

* **Thuộc tính:**
  + \_id: ObjectId (PK)
  + bookingId: ObjectId (ref: 'Booking', required, unique) // Một booking chỉ có 1 review
  + customerId: ObjectId (ref: 'User', required)
  + operatorId: ObjectId (ref: 'BusOperator', required)
  + tripId: ObjectId (ref: 'Trip')
  + **Ratings:**
    - overallRating: Number (required, min: 1, max: 5)
    - busRating: Number (min: 1, max: 5)
    - driverRating: Number (min: 1, max: 5)
    - punctualityRating: Number (min: 1, max: 5)
    - serviceRating: Number (min: 1, max: 5)
  + **Review Content:**
    - comment: String (maxlength: 1000)
    - images: Array of String (URLs) // Ảnh đánh giá
  + **Status:**
    - isApproved: Boolean (default: false) // Admin duyệt
    - isHidden: Boolean (default: false)
    - approvedBy: ObjectId (ref: 'User')
    - approvedAt: Date
  + **Response:**
    - operatorResponse: String // Phản hồi từ nhà xe
    - respondedAt: Date
  + **Helpful:**
    - helpfulCount: Number (default: 0) // Số người thấy hữu ích
  + createdAt: DateTime (auto)
  + updatedAt: DateTime (auto)
* **Indexes:**
  + bookingId: 1
  + customerId: 1
  + operatorId: 1, isApproved: 1
  + tripId: 1
  + overallRating: 1
  + createdAt: -1
* **Phương thức:**
  + Instance method: approve(adminId): Promise - Duyệt đánh giá
  + Instance method: hide(): Promise - Ẩn đánh giá
  + Instance method: addOperatorResponse(response): Promise - Thêm phản hồi từ nhà xe
  + Static method: getAverageRatings(operatorId): Promise - Tính rating trung bình
  + Post-save hook: Cập nhật averageRating và totalReviews của BusOperator
  + **Ghi chú:**
    - Chỉ cho phép review sau khi chuyến xe completed
    - Cần validate booking đã hoàn thành trước khi tạo review

### 12. Notification Model

* + - **Thuộc tính:**
      * \_id: ObjectId (PK)
      * userId: ObjectId (ref: 'User', required)
      * **Notification Type:**
        + type: Enum ('email', 'sms', 'push', 'in\_app') - required
        + category: Enum ('booking', 'payment', 'promotion', 'system', 'reminder') - required
      * **Content:**
        + title: String (required, maxlength: 200)
        + content: String (required, maxlength: 2000)
        + imageUrl: String
        + actionUrl: String // Deep link hoặc URL
      * **Delivery:**
        + status: Enum ('pending', 'sent', 'failed', 'read') - default: 'pending'
        + sentAt: Date
        + readAt: Date
        + failureReason: String
      * **Priority:**
        + priority: Enum ('low', 'normal', 'high', 'urgent') - default: 'normal'
      * **Metadata:**
        + metadata: Mixed // Object chứa thông tin bổ sung (bookingId, tripId, etc.)
      * **Scheduling:**
        + scheduledFor: Date // Gửi vào thời điểm cụ thể
        + expiresAt: Date
      * createdAt: DateTime (auto)
      * updatedAt: DateTime (auto)
    - **Indexes:**
      * userId: 1, status: 1
      * userId: 1, readAt: 1
      * type: 1, status: 1
      * scheduledFor: 1, status: 1
      * createdAt: -1
    - **Phương thức:**
      * Instance method: markAsSent(): Promise - Đánh dấu đã gửi
      * Instance method: markAsRead(): Promise - Đánh dấu đã đọc
      * Instance method: markAsFailed(reason): Promise - Đánh dấu thất bại
      * Static method: createBookingNotification(userId, bookingId, type): Promise
      * Static method: getPendingNotifications(limit = 100): Promise - Lấy notifications chưa gửi
      * Static method: getUnreadCount(userId): Promise - Đếm thông báo chưa đọc
      * Static method: markAllAsRead(userId): Promise - Đánh dấu tất cả đã đọc
    - **Ghi chú:**
      * Làm việc với notificationService.js để gửi thông báo
      * Có thể dùng cron job để gửi scheduled notifications

### 13. Content Model

* + - **Thuộc tính:**
      * \_id: ObjectId (PK)
      * **Content Type:**
        + type: Enum ('banner', 'blog', 'faq', 'policy', 'announcement', 'promotion') - required
        + slug: String (required, unique, lowercase) // URL-friendly
      * **Content:**
        + title: String (required, maxlength: 200)
        + subtitle: String (maxlength: 300)
        + body: String (required) // HTML hoặc Markdown
        + excerpt: String (maxlength: 500) // Mô tả ngắn
        + imageUrl: String
        + thumbnailUrl: String
        + tags: Array of String
      * **SEO:**
        + metaTitle: String
        + metaDescription: String
        + metaKeywords: Array of String
      * **Display:**
        + isActive: Boolean (default: true)
        + order: Number (default: 0) // Thứ tự hiển thị
        + featured: Boolean (default: false) // Nổi bật
      * **Author:**
        + createdBy: ObjectId (ref: 'User', required)
        + updatedBy: ObjectId (ref: 'User')
      * **Analytics:**
        + viewCount: Number (default: 0)
        + likeCount: Number (default: 0)
      * **Scheduling:**
        + publishedAt: Date
        + expiresAt: Date
      * createdAt: DateTime (auto)
      * updatedAt: DateTime (auto)
    - **Indexes:**
      * slug: 1
      * type: 1, isActive: 1
      * type: 1, order: 1
      * featured: 1, isActive: 1
      * createdBy: 1
      * publishedAt: -1
    - **Phương thức:**
      * Instance method: incrementView(): Promise - Tăng lượt xem
      * Instance method: incrementLike(): Promise - Tăng lượt thích
      * Instance method: publish(): Promise - Xuất bản content
      * Instance method: unpublish(): Promise - Gỡ xuất bản
      * Static method: getFeaturedContent(type, limit = 5): Promise - Lấy content nổi bật
      * Static method: getContentByType(type, options): Promise - Lấy theo loại
      * Pre-save hook: Generate slug từ title nếu chưa có
      * Pre-save hook: Set publishedAt nếu chuyển isActive từ false -> true
    - **Ghi chú:**
      * Dùng cho CMS (Content Management System)
      * Body có thể chứa HTML hoặc Markdown tùy implementation

### 14. SupportTicket Model

* + - **Thuộc tính:**
      * \_id: ObjectId (PK)
      * ticketNumber: String (required, unique, uppercase) // Format: TK{YYYYMMDD}{####}
      * userId: ObjectId (ref: 'User', required)
      * **Ticket Info:**
        + subject: String (required, maxlength: 200)
        + description: String (required, maxlength: 2000)
        + category: Enum ('booking', 'payment', 'refund', 'technical', 'complaint', 'other') - required
        + relatedBookingId: ObjectId (ref: 'Booking')
        + relatedPaymentId: ObjectId (ref: 'Payment')
      * **Priority & Status:**
        + priority: Enum ('low', 'medium', 'high', 'urgent') - default: 'medium'
        + status: Enum ('open', 'assigned', 'inProgress', 'resolved', 'closed', 'cancelled') - default: 'open'
      * **Assignment:**
        + assignedTo: ObjectId (ref: 'User') // Admin/Support user
        + assignedAt: Date
      * **Messages (embedded array):**
        + messages: Array of embedded objects

sender: ObjectId (ref: 'User', required)

senderType: Enum ('customer', 'admin')

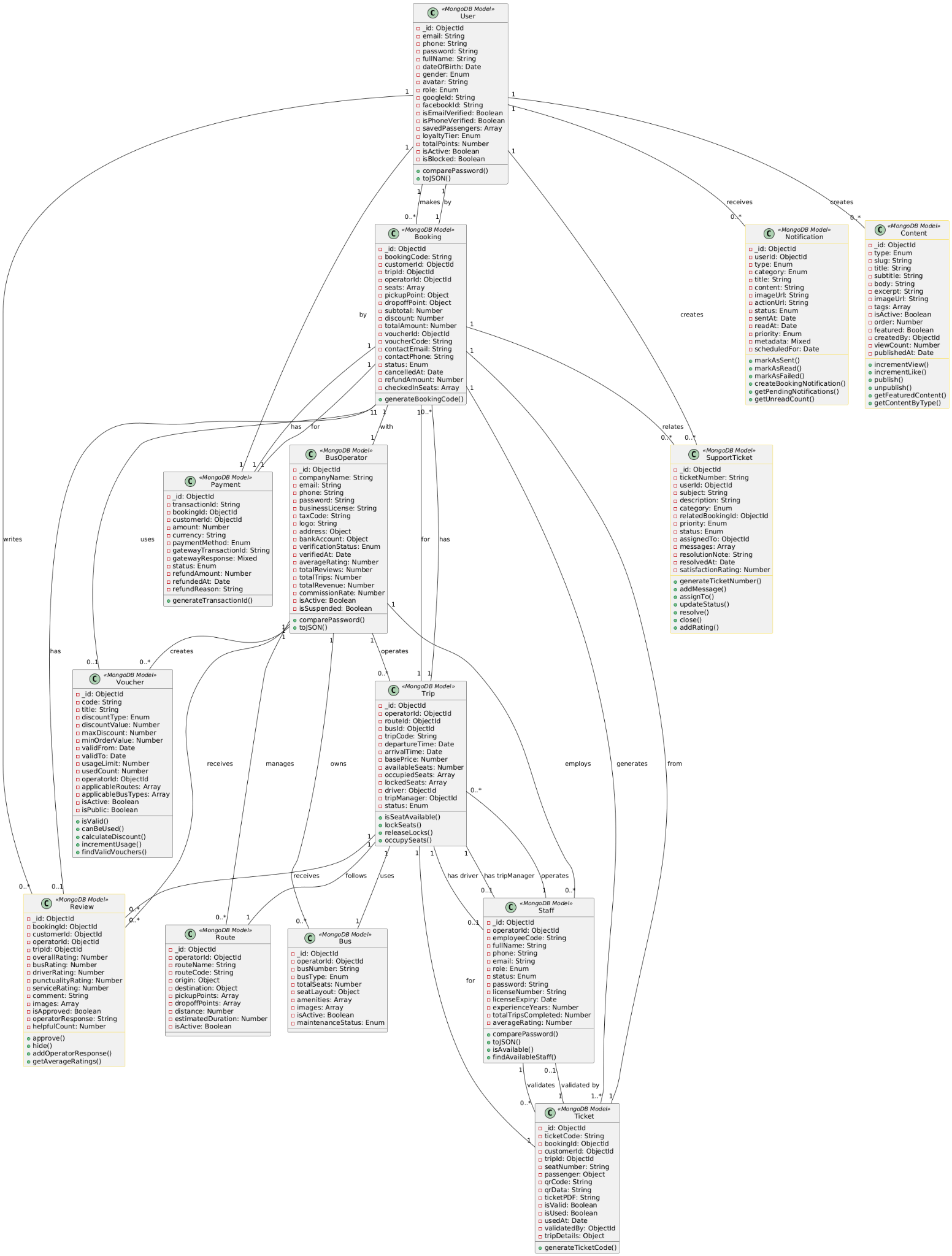
message: String (required)

attachments: Array of String (URLs)

createdAt: Date (default: Date.now)

* + - * **Resolution:**
        + resolutionNote: String
        + resolvedAt: Date
        + resolvedBy: ObjectId (ref: 'User')
        + closedAt: Date
      * **Rating:**
        + satisfactionRating: Number (min: 1, max: 5) // Đánh giá sau khi giải quyết
        + feedbackComment: String
      * createdAt: DateTime (auto)
      * updatedAt: DateTime (auto)
    - **Indexes:**
      * ticketNumber: 1
      * userId: 1, status: 1
      * assignedTo: 1, status: 1
      * category: 1, status: 1
      * status: 1, priority: 1
      * createdAt: -1
    - **Phương thức:**
      * Static method: generateTicketNumber(): Promise - Tạo mã ticket
      * Instance method: addMessage(senderId, senderType, message, attachments): Promise - Thêm tin nhắn
      * Instance method: assignTo(adminId): Promise - Gán cho admin
      * Instance method: updateStatus(newStatus): Promise - Cập nhật trạng thái
      * Instance method: resolve(adminId, note): Promise - Đánh dấu đã giải quyết
      * Instance method: close(): Promise - Đóng ticket
      * Instance method: addRating(rating, comment): Promise - Thêm đánh giá
      * Static method: getTicketsByStatus(status, options): Promise
      * Static method: getMyTickets(userId, options): Promise
      * Static method: getAssignedTickets(adminId, options): Promise
      * Pre-save hook: Tự động set assignedAt khi có assignedTo
      * Post-save hook: Gửi notification khi có message mới
    - **Ghi chú:**
      * Messages được lưu dưới dạng embedded array
      * Cần real-time updates cho chat (có thể dùng [Socket.io](http://socket.io))

## 6.2 Class Diagram

.

## 6.3 Quan Hệ Giữa Các Lớp

### 2.5.1 Quan hệ Reference (ObjectId References)

Trong MongoDB, các models được liên kết thông qua ObjectId references:

User Model:

* User ← Booking.customerId (1:N) - Một user có nhiều bookings
* User ← Payment.customerId (1:N) - Một user có nhiều payments
* User ← Ticket.customerId (1:N) - Một user có nhiều tickets
* User ← Review.customerId (1:N) - Một user có nhiều reviews
* User ← Notification.userId (1:N) - Một user có nhiều notifications
* User ← Content.createdBy (1:N) - Một user tạo nhiều contents
* User ← SupportTicket.userId (1:N) - Một user có nhiều tickets

BusOperator Model:

* BusOperator ← Route.operatorId (1:N) - Một nhà xe quản lý nhiều tuyến
* BusOperator ← Bus.operatorId (1:N) - Một nhà xe sở hữu nhiều xe
* BusOperator ← Trip.operatorId (1:N) - Một nhà xe vận hành nhiều chuyến
* BusOperator ← Booking.operatorId (1:N) - Một nhà xe nhận nhiều bookings
* BusOperator ← Staff.operatorId (1:N) - Một nhà xe có nhiều nhân viên
* BusOperator ← Voucher.operatorId (1:N) - Một nhà xe tạo nhiều vouchers
* BusOperator ← Review.operatorId (1:N) - Một nhà xe nhận nhiều reviews

Route Model:

* Route ← Trip.routeId (1:N) - Một tuyến có nhiều chuyến
* Route ← Voucher.applicableRoutes (N:M) - Voucher áp dụng cho nhiều tuyến

Bus Model:

* Bus ← Trip.busId (1:N) - Một xe thực hiện nhiều chuyến

Trip Model:

* Trip ← Booking.tripId (1:N) - Một chuyến có nhiều bookings
* Trip ← Ticket.tripId (1:N) - Một chuyến có nhiều tickets
* Trip → Staff (driver) (N:1) - Nhiều chuyến có thể có cùng tài xế
* Trip → Staff (tripManager) (N:1) - Nhiều chuyến có thể có cùng quản lý
* Trip ← Review.tripId (1:N) - Một chuyến có nhiều reviews

Booking Model:

* Booking ← Ticket.bookingId (1:N) - Một booking tạo nhiều tickets
* Booking ← Payment.bookingId (1:1) - Một booking có một payment
* Booking → Voucher.voucherId (N:1) - Nhiều bookings có thể dùng cùng voucher
* Booking → Staff.checkedInBy (N:1) - Nhiều bookings được check-in bởi cùng staff
* Booking ← Review.bookingId (1:1) - Một booking có một review
* Booking ← SupportTicket.relatedBookingId (1:N) - Một booking có thể có nhiều tickets

Ticket Model:

* Ticket → Staff.validatedBy (N:1) - Nhiều tickets được validate bởi cùng staff

Staff Model:

* Staff ← Trip.driver (1:N) - Một staff có thể lái nhiều chuyến
* Staff ← Trip.tripManager (1:N) - Một staff có thể quản lý nhiều chuyến
* Staff ← Booking.checkedInBy (1:N) - Một staff check-in nhiều bookings
* Staff ← Ticket.validatedBy (1:N) - Một staff validate nhiều tickets

### 2.5.2 Quan hệ Embedded (Nested Documents)

Các documents được nhúng trực tiếp trong parent document (không phải reference):

User Model:

* savedPassengers: Array of embedded { fullName, phone, idCard }

BusOperator Model:

* address: Embedded { street, ward, district, city, country }
* bankAccount: Embedded { bankName, accountNumber, accountHolder }

Route Model:

* origin: Embedded { city, province, station, address, coordinates }
* destination: Embedded { city, province, station, address, coordinates }
* pickupPoints: Array of embedded { name, address, coordinates }
* dropoffPoints: Array of embedded { name, address, coordinates }

Bus Model:

* seatLayout: Embedded { floors, rows, columns, layout }

Trip Model:

* occupiedSeats: Array of String
* lockedSeats: Array of embedded { seatNumber, lockedUntil, sessionId }

Booking Model:

* seats: Array of embedded { seatNumber, passenger: { fullName, phone, idCard } }
* pickupPoint: Embedded { name, address, coordinates }
* dropoffPoint: Embedded { name, address, coordinates }

Ticket Model:

* passenger: Embedded { fullName, phone, idCard }
* tripDetails: Embedded { routeName, origin, destination, departureTime, busNumber, operatorName }

Payment Model:

* gatewayResponse: Mixed (object chứa raw response từ payment gateway)

SupportTicket Model:

* messages: Array of embedded { sender, senderType, message, attachments, createdAt }

### 2.5.3 Đặc Điểm Thiết Kế MongoDB

1. Denormalization (Phi chuẩn hóa):

* Ticket.tripDetails chứa thông tin denormalized từ Trip, Route, Bus - Tránh join khi hiển thị vé
* Booking.operatorId duplicate từ Trip - Optimize query bookings by operator

2. Embedded vs Referenced:

* Embedded: Dữ liệu không cần query riêng (passenger, seatLayout, lockedSeats, messages)
* Referenced: Dữ liệu cần query độc lập hoặc được share (User, Trip, Bus, Route, Staff, Voucher)

3. Array Fields:

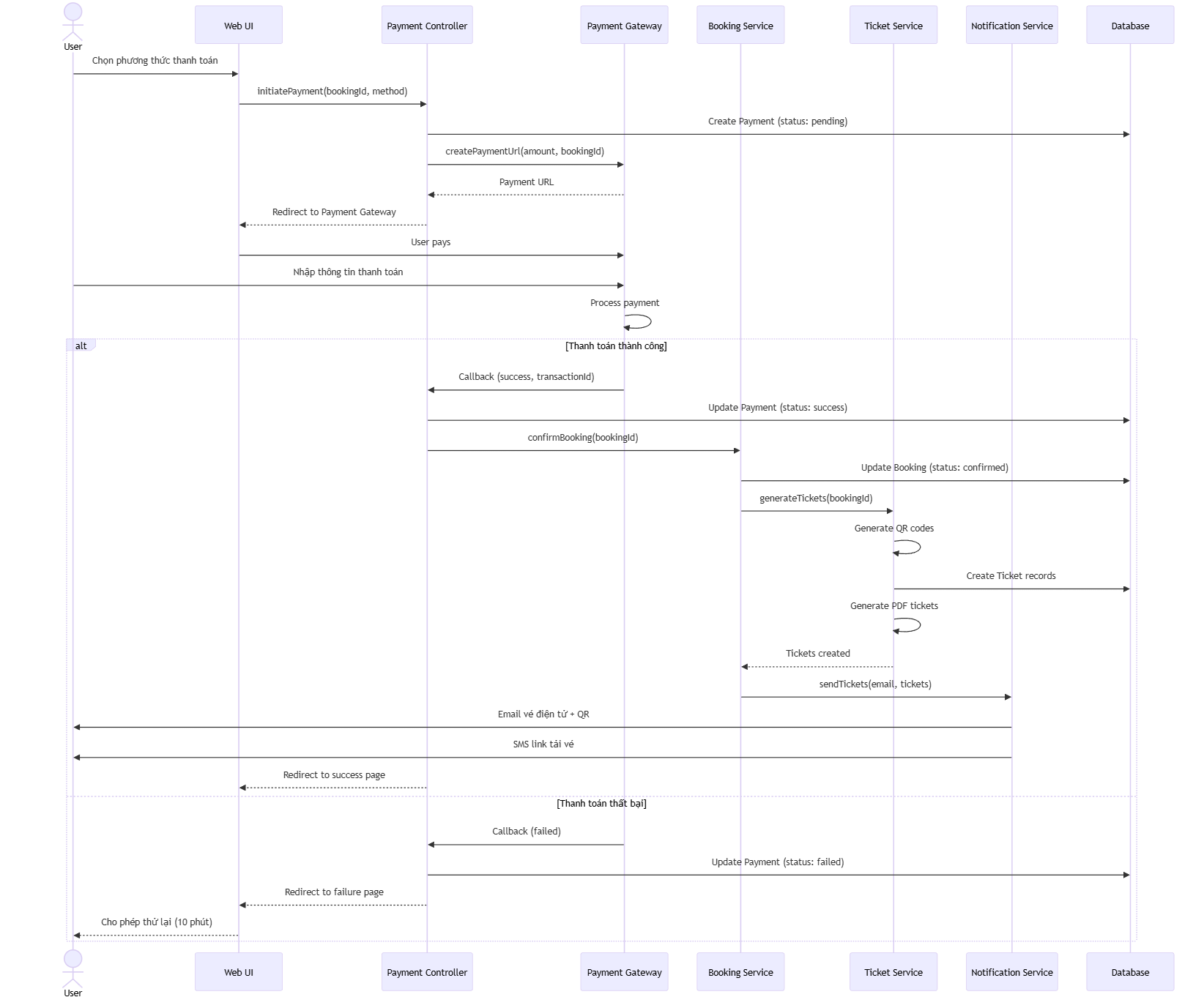
* occupiedSeats, lockedSeats, seats, pickupPoints, messages, applicableRoutes - Cho phép query linh hoạt
* Indexes được tạo trên các array fields quan trọng

# 7. SƠ ĐỒ LUỒNG HOẠT ĐỘNG

## 7.1 Sequence Diagrams (Sơ Đồ Tuần Tự)

### 7.1.1 UC-5: Đặt Vé

### 7.1.2 UC-6: Thanh Toán



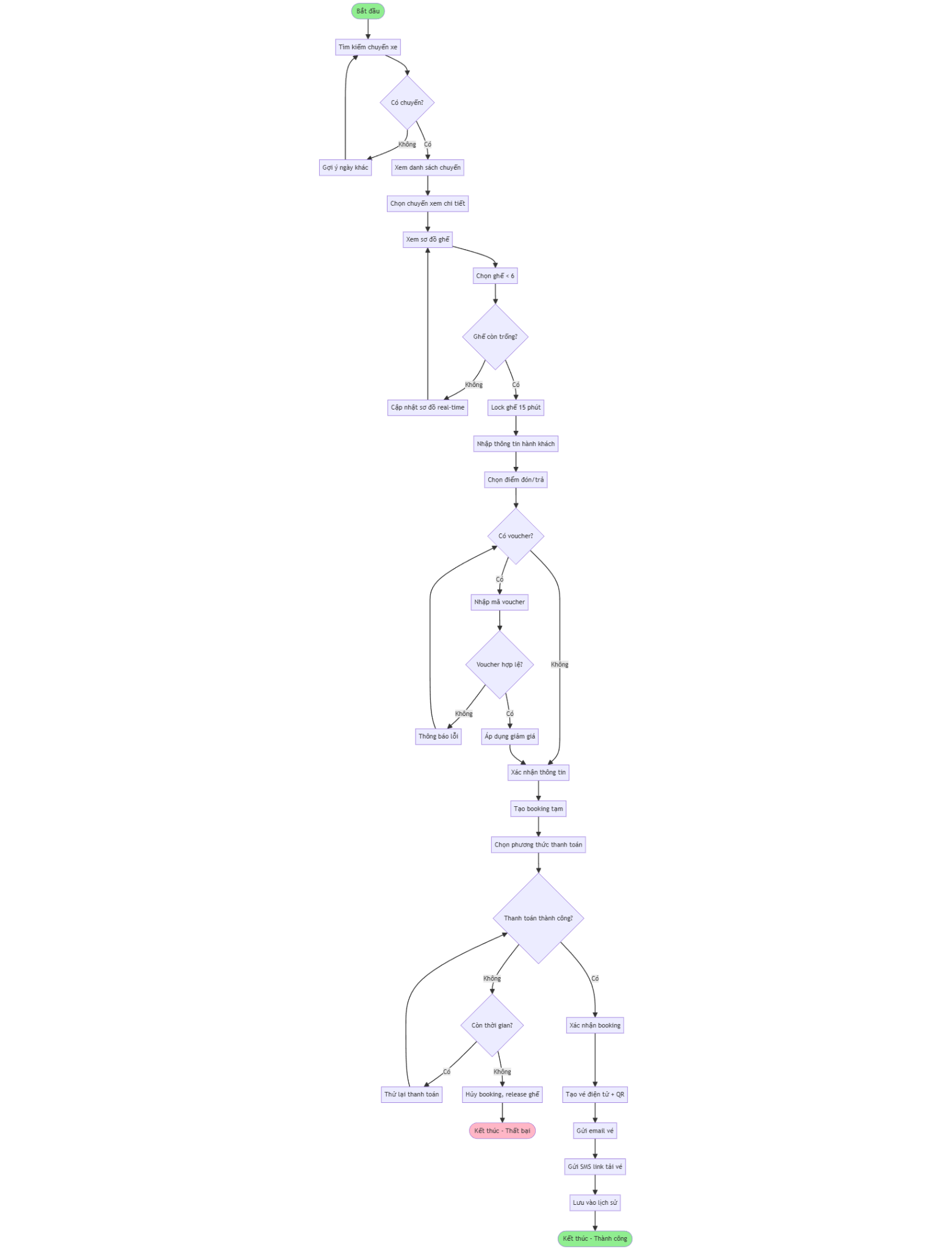
### 7.1.3 UC-19: Quét QR Xác Thực Vé

### 7.1.4 UC-14: Tạo Lịch Trình Chuyến Xe

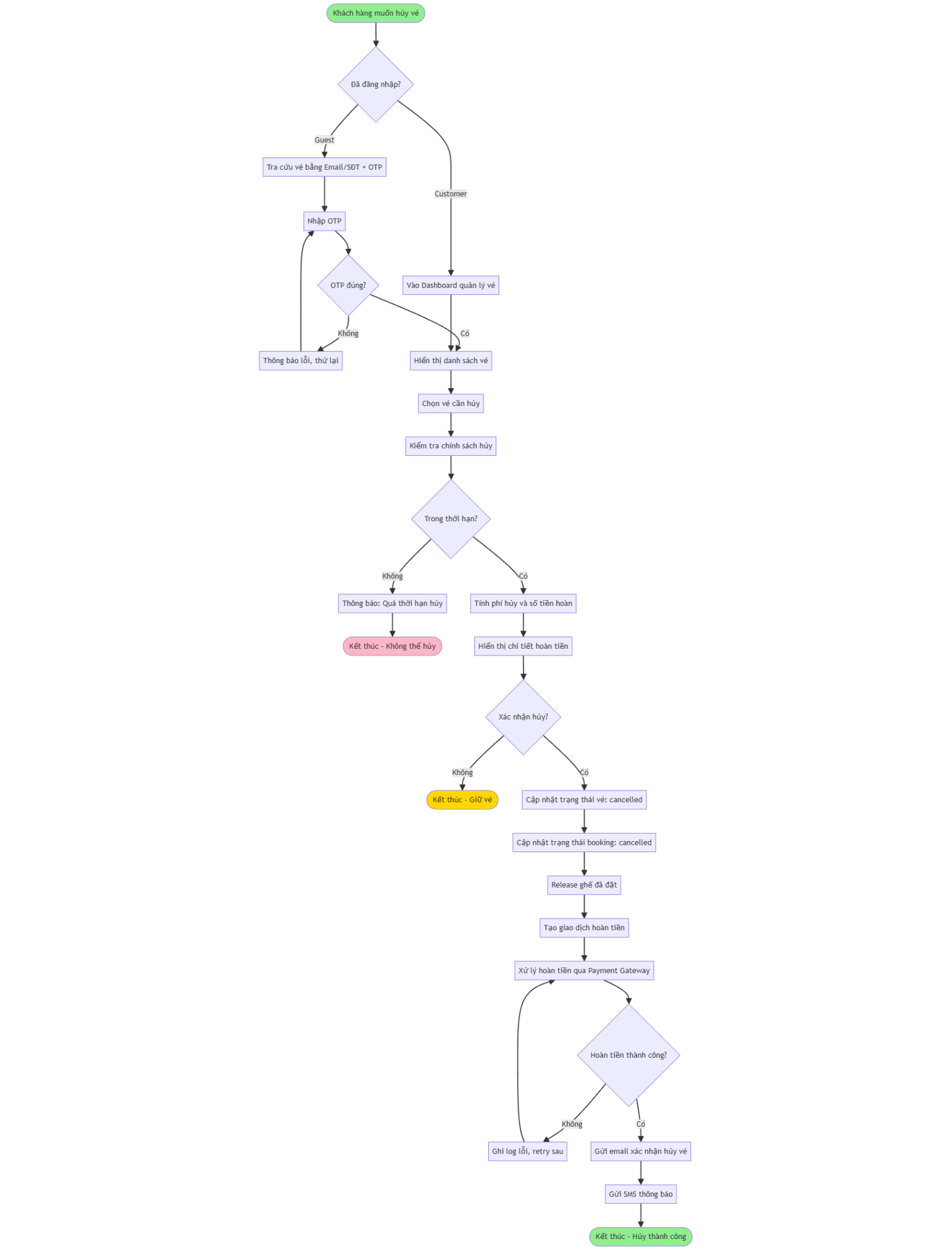
## 

## 7.2 Activity Diagrams (Sơ Đồ Hoạt Động)

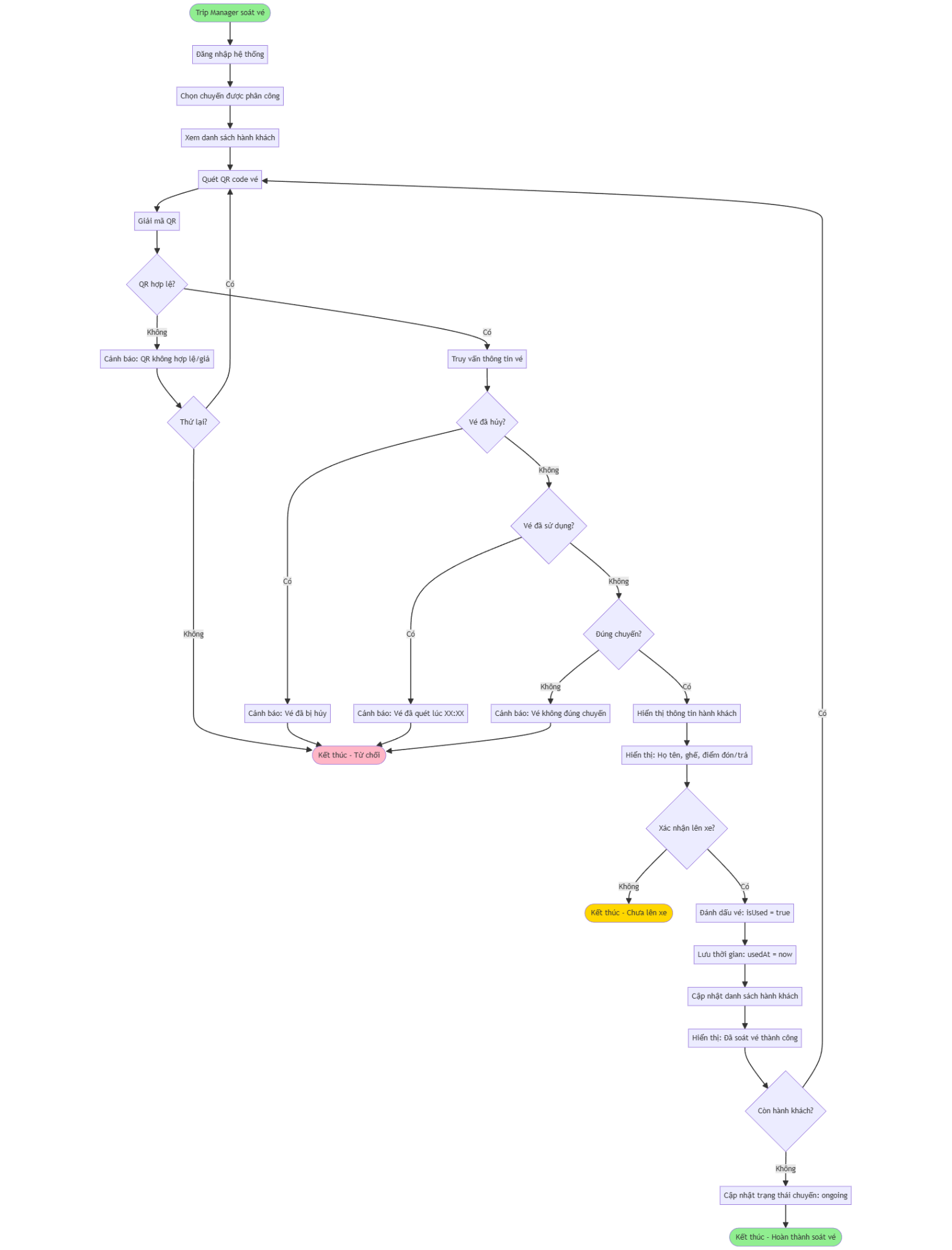
### 7.2.1 Quy Trình Đặt Vé Hoàn Chỉnh (End-to-End)



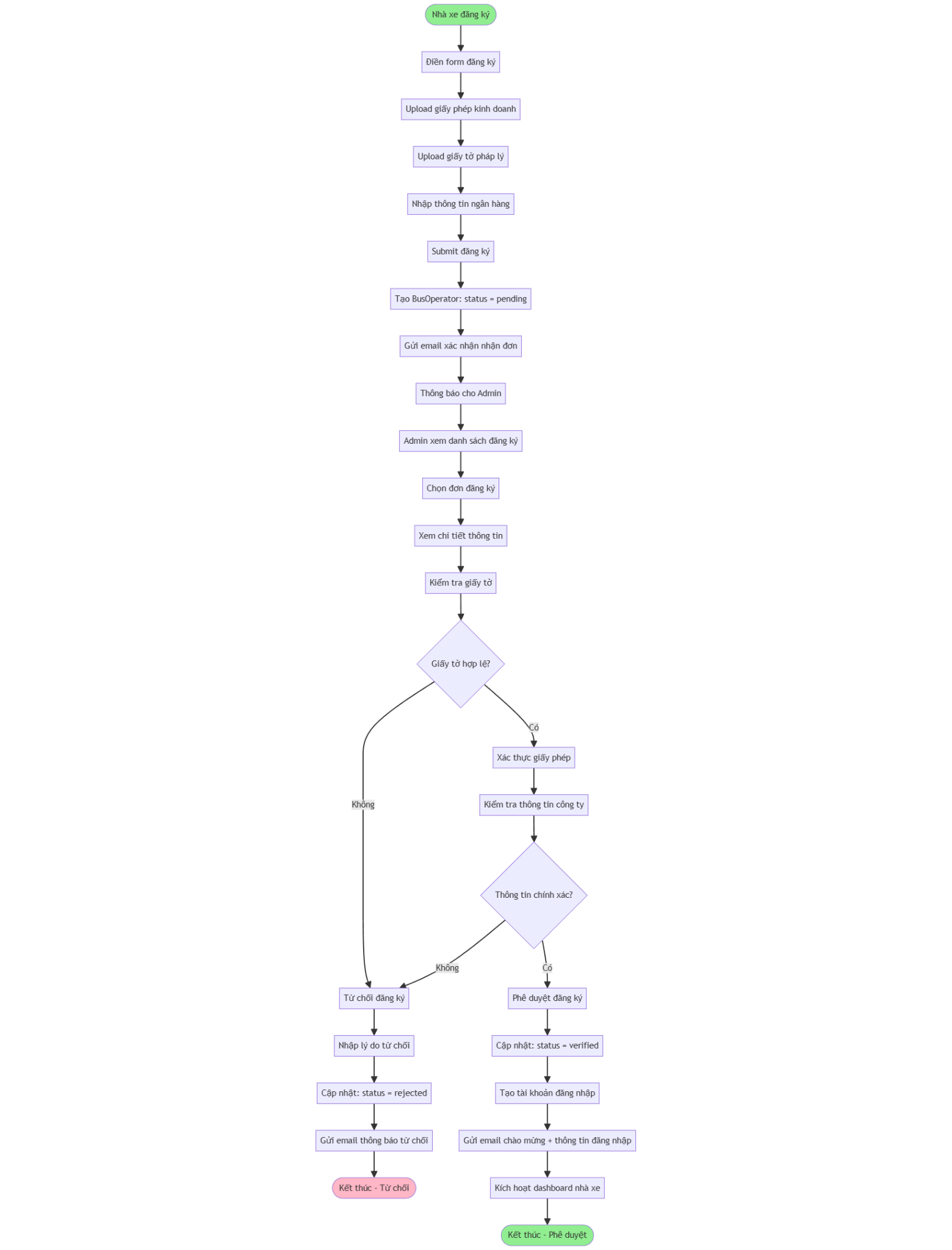
### 7.2.2 Quy Trình Hủy Vé



### 7.2.3 Quy Trình Soát Vé

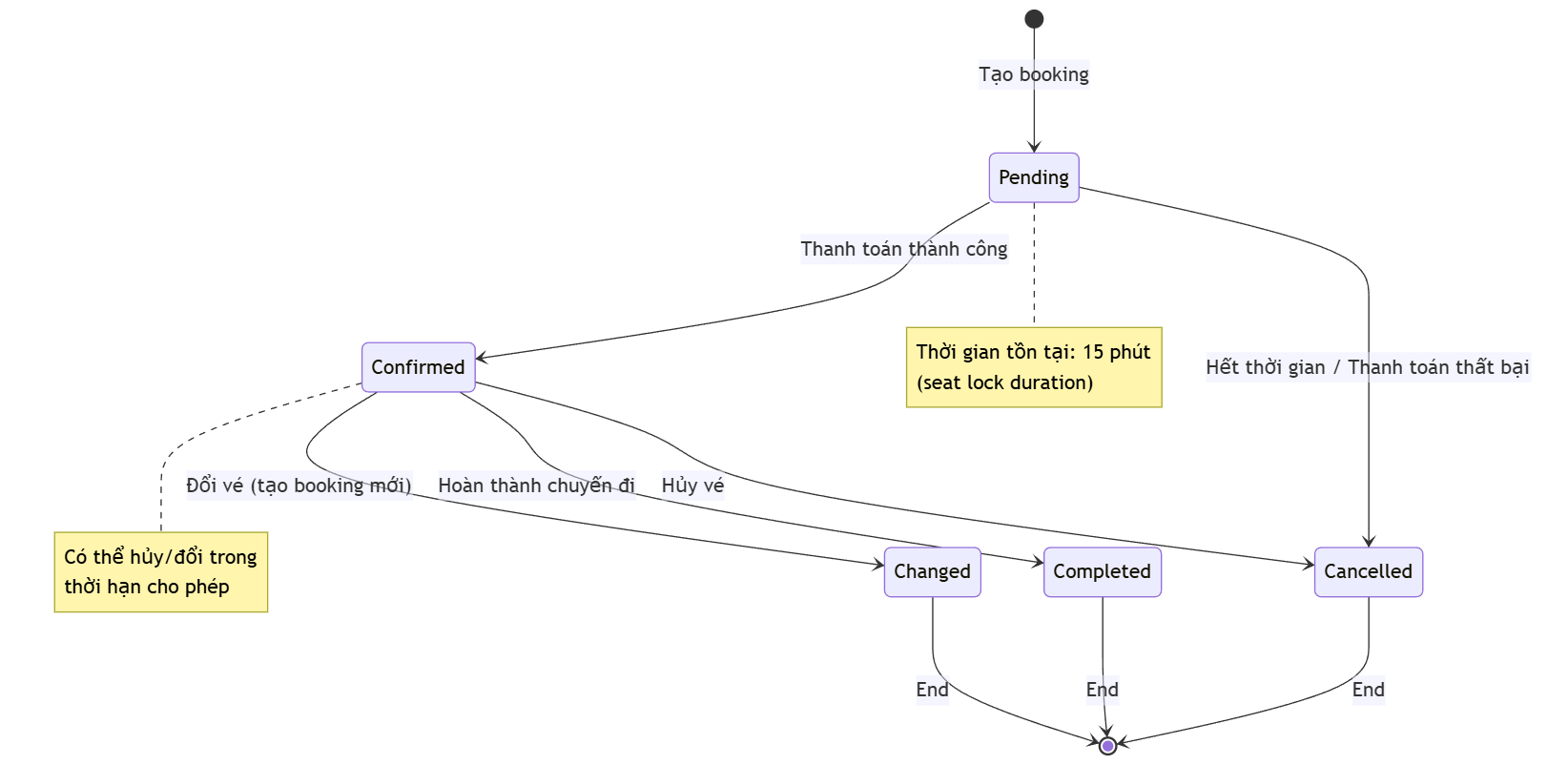


### 7.2.4 Quy Trình Duyệt Nhà Xe Mới

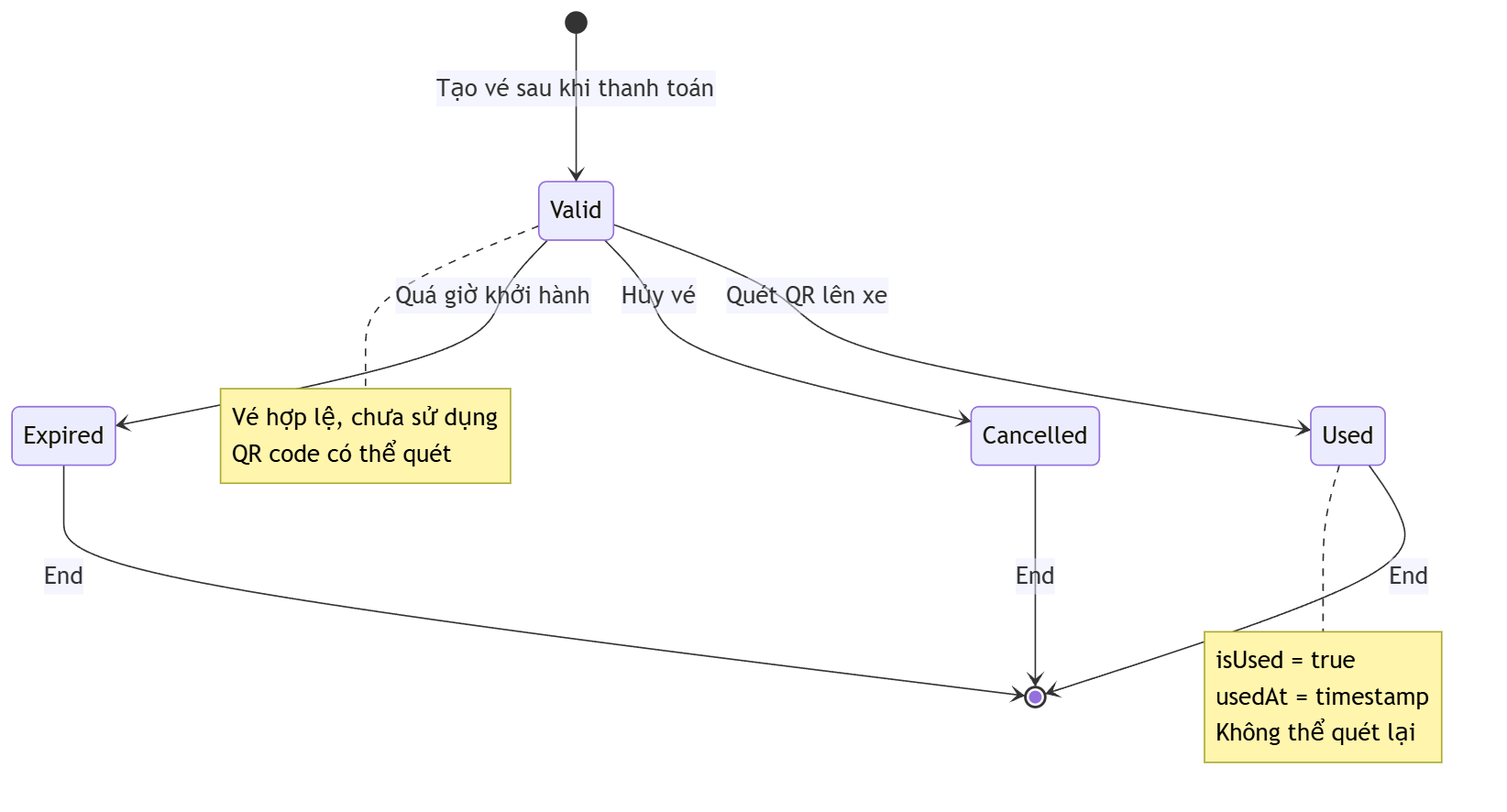


## 7.3 State Diagrams (Sơ Đồ Trạng Thái)

### 7.3.1 Trạng Thái Booking



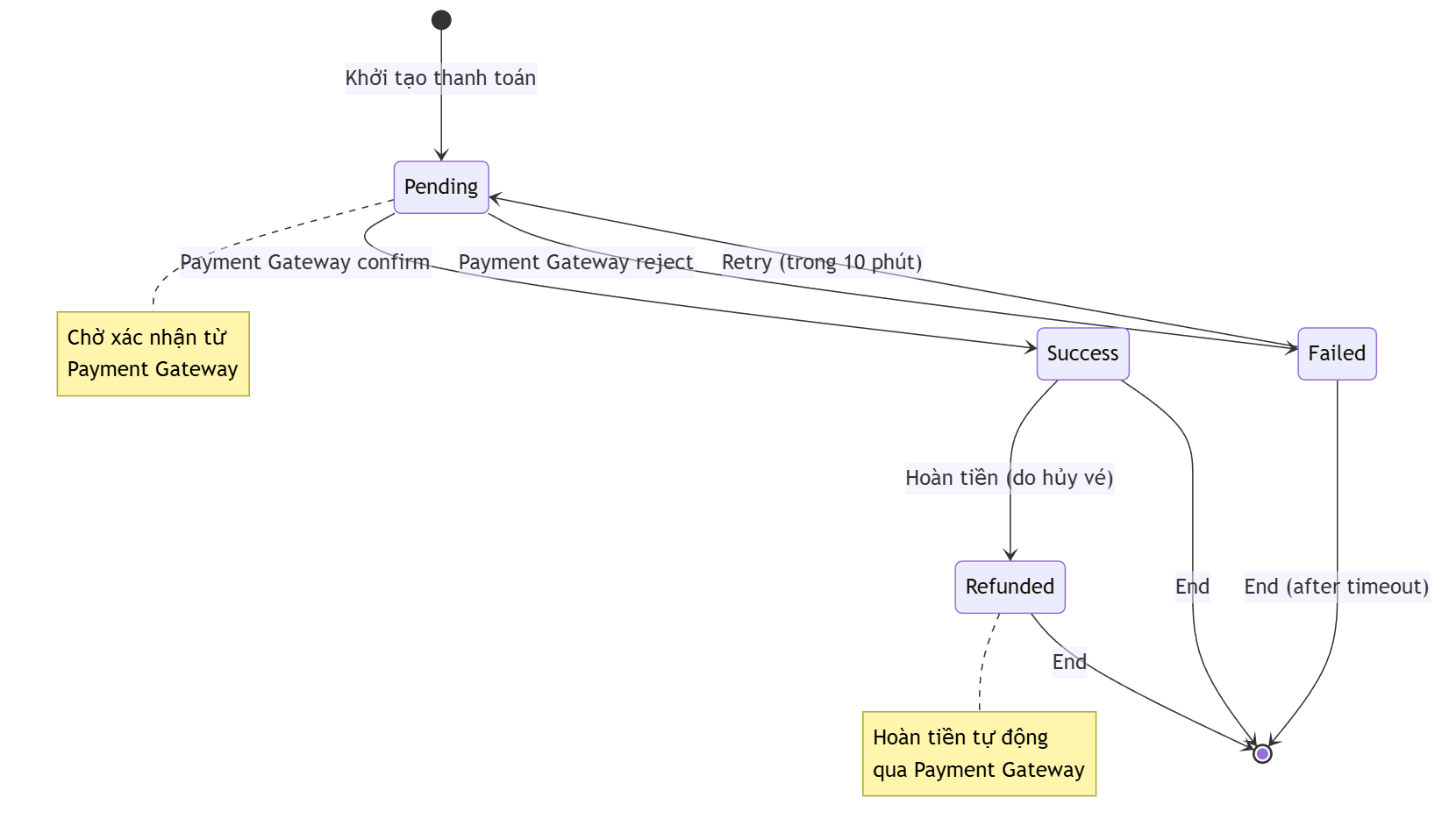
### 7.3.2 Trạng Thái Ticket



### 7.3.3 Trạng Thái Trip



### 7.3.4 Trạng Thái Payment

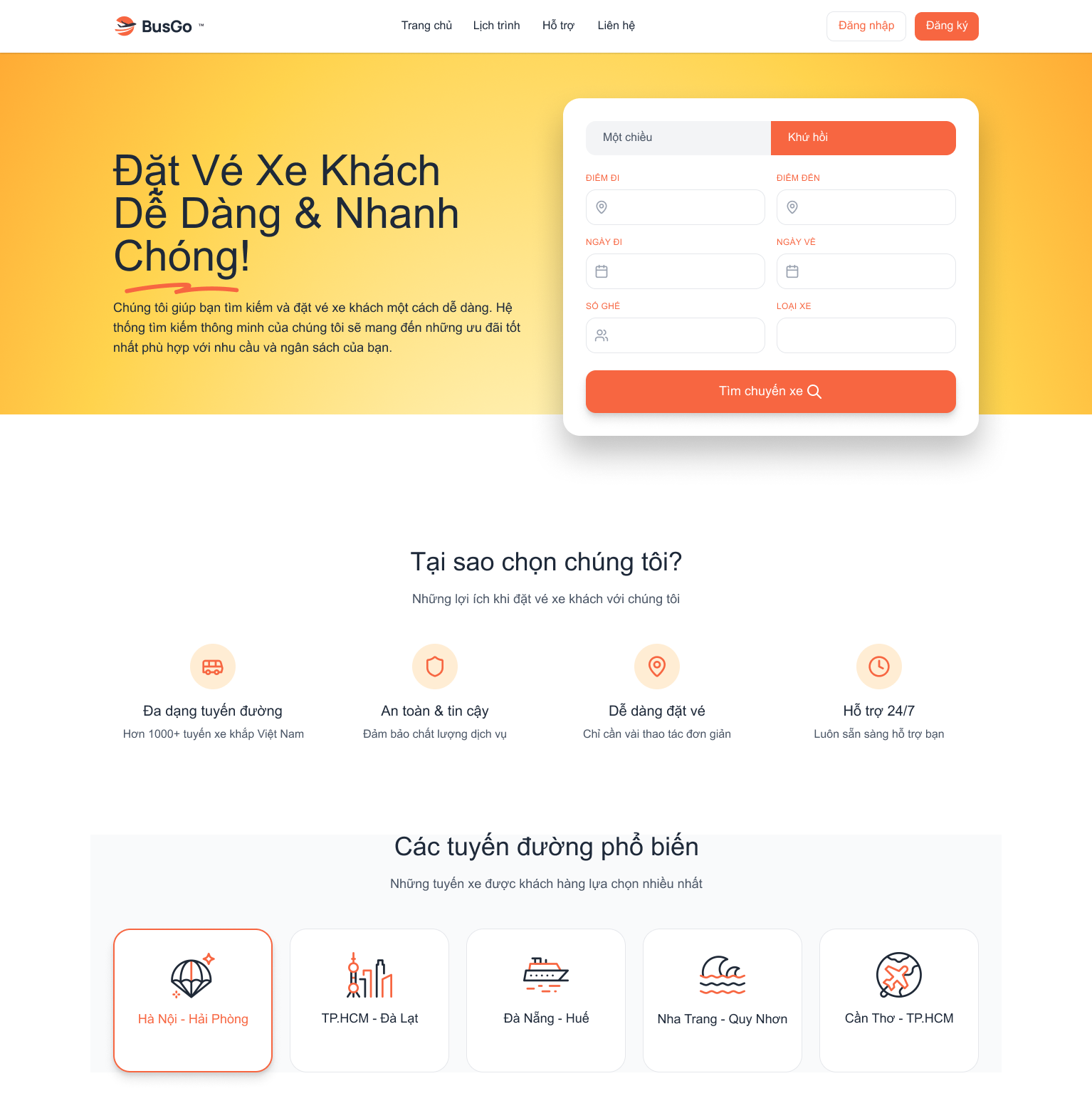


## 8. Lược Đồ Quan Hệ ERD

## 

# 9. Màn hình giao diện

## 9.1 Màn hình giao diện chính



## 9.2 Màn hình giao diện chọn chuyến đi

