

# **IS 3920 – Individual Project on Business Solutions**

## **Software Requirements Specification**

**for**

**TravelNest**

**Prepared by 225083P Geeneth Punchihewa**

**Department of Interdisciplinary Studies**

**Faculty of Information Technology**

**University of Moratuwa**

**15/01/2026**

**Word Count: 1976**

## Student's Declaration

I hereby declare that this report is my original work and has not been submitted, in whole or in part, for any degree or diploma at any university or other institution of higher learning. All information derived from the work of others, whether published or unpublished, has been properly acknowledged in the text, and a complete list of references has been provided.

.....16/01/2026.....  
Date

.....  
Signature of Student

## Supervisors' declaration

I hereby declare that I have reviewed this project and find it adequate in both scope and quality.

1. Name of Supervisor: Prof. (Mrs.) Thanuja C. Sandanayake

Designation: Professor, Department of Interdisciplinary Studies, Faculty of Information Technology, University of Moratuwa

Date: 16/01/2026

Signature:

Any further comments:

2. Name of Supervisor: Mr. Nishith Pinnawala

Designation: Senior Software Engineer, Digital Mobility Solutions Lanka (Pvt) Ltd

Date: 16/01/2026

Signature:

Any further comments:

## Table of Contents

1 Introduction .....	7
1.1 Purpose .....	7
1.2 Document Conventions .....	7
1.3 Intended Audience and Reading Suggestions.....	7
1.4 Product Scope .....	7
1.4.1. Problem in brief .....	8
1.4.2. Aim and Objectives .....	8
1.5 References .....	8
2 Overall description.....	9
2.1 Product Perspective .....	10
2.2 Product Functions .....	10
2.3 User Classes and Characteristics.....	10
2.4 Operating Environment .....	10
2.5 Design and Implementation Constraints .....	11
2.6 User Documentation.....	11
2.7 Assumptions and Dependencies.....	11
3 External Interface Requirements .....	12
3.1 User Interfaces .....	12
3.2 Hardware Interfaces .....	12
3.3 Software Interfaces.....	12
3.4 Communication Interfaces.....	12
4 System Features.....	13
4.1 User Registration and Profile Management .....	13
4.2 Vehicle Listing and Management.....	13
4.3 Search and Quotation System.....	13
4.4 Booking and Payment Processing .....	13
4.5 Admin Verification and Oversight.....	14
5 Other Nonfunctional Requirements .....	14
5.1 Performance Requirements.....	14
5.2 Safety Requirements.....	14
5.3 Security Requirements.....	14

5.4 Software Quality Attributes .....	14
5.5 Business Rules .....	14
Appendix A – Glossary .....	15
Appendix B – Analysis Models .....	15
Use Case Diagram .....	15
Class Diagram.....	15
ER Diagram.....	15

## List of Figures

Figure 1: TravelNest System Architecture Diagram.....	9
Figure 2: TravelNest Use Case Diagram .....	16
Figure 3: TravelNest ER Diagram Part 1 .....	17
Figure 4: TravelNest ER Diagram Part 2 .....	18
Figure 5: TravelNest Class Diagram Part 1 .....	19
Figure 6: TravelNest Class Diagram Part 2 .....	20

## Revision History

Name	Date	Reason for Changes	Version

# 1 Introduction

## 1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to document the functional and non-functional requirements for the "TravelNest" Bus Rental Marketplace Platform. This document provides a detailed description of the system's capabilities, user interfaces, and constraints to guide the development, testing, and deployment phases.

## 1.2 Document Conventions

This document follows the IEEE 830-1998 standard for Software Requirements Specifications.

- **Bold text** is used for headings and key terms.
- *Italic text* is used for emphasis.
- Priorities for requirements are indicated as High, Medium, or Low.

## 1.3 Intended Audience and Reading Suggestions

- Project Supervisors/Mentors: To evaluate the scope and feasibility of the proposed solution.
- Developers: To understand the system architecture and implement features.
- Testers: To design test cases based on functional requirements.
- Stakeholders (Bus Owners/Customers): To understand the platform's capabilities.

## 1.4 Product Scope

The TravelNest platform encompasses the full lifecycle of a charter bus rental, serving as a digital intermediary between bus owners and private groups.

**The system boundaries include:**

- **Web-Based Marketplace:** A centralized platform allowing multiple bus owners to list vehicles and customers to compare options.
- **Automated Logistics:** Algorithms for distance calculation, route optimization (via Google Maps), and standardized pricing quotation.
- **Transaction Management:** Handling of booking requests, confirmations, and secure online payments.
- **Stakeholder Portals:** Dedicated interfaces for Customers, Bus Owners, and System Administrators.

**The system boundaries exclude:**

- **Scheduled Passenger Services:** The platform does not handle individual seat ticketing for public transport routes (e.g., intercity bus tickets).
- **Physical Fleet Operations:** The system does not manage vehicle maintenance, driver scheduling, or fuel logistics; these remain the responsibility of the vehicle owner.

- **Native Mobile Applications:** The solution is scoped as a Progressive Web App (PWA) and does not include separate native Android or iOS app development in this phase.

#### 1.4.1. Problem in brief

The Sri Lankan group transportation market lacks a centralized, transparent marketplace for private bus charter services. Existing platforms focus on smaller vehicles (PickMe) or single-company fleets (Malkey). Customers face challenges with fragmented pricing, lack of vehicle comparison, and manual booking processes.

#### 1.4.2. Aim and Objectives

##### **Aim:**

To design and develop an island-wide, web-based bus rental marketplace platform that facilitates connections between bus owners and customers for group transportation services.

##### **Objectives:**

- Develop a responsive web application (PWA) for occasional user accessibility.
- Implement automated owner registration with document validation and permit compliance.
- Create advanced search/filtering capabilities with geographic mapping integration.
- Establish a standardized quotation system with all-inclusive pricing models.
- Integrate secure online payment processing and multilingual support (Sinhala/Tamil/English).

#### 1.5 References

- <https://pickme.lk/services/rentals/>
- <https://www.malkey.lk/vehicle-fleet/buses-and-vans.html>
- <https://www.rofi.lk/blog/toyota-coaster-manual-the-go-to-mini-bus-for-group-tours-in-sri-lanka/>
- IEEE Computer Society. (1998). *IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998)*. Institute of Electrical and Electronics Engineers.



## 2 Overall description

The TravelNest system follows a micro-services inspired architecture using a React/Next.js frontend and a Node.js/Express backend. Below is the [System Architecture diagram](#).

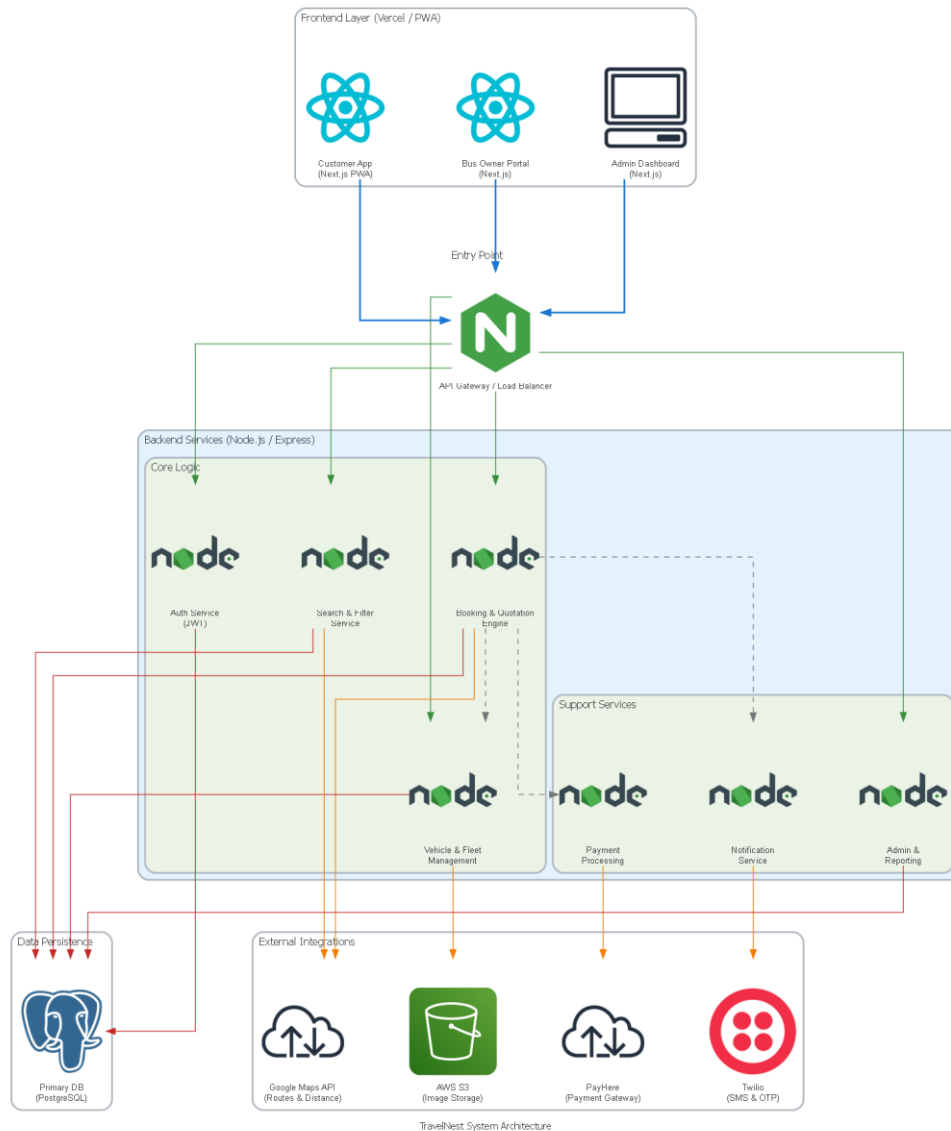


Figure 1: TravelNest System Architecture Diagram

- **Frontend:** Next.js PWA for Customers and Bus Owners; Admin Dashboard.
- **Backend:** Node.js/Express services for Auth, Booking, Vehicle Management.
- **Database:** PostgreSQL for structured data.
- **External Services:** Google Maps (Location), PayHere (Payments), Twilio (SMS), AWS S3 (Storage)

## 2.1 Product Perspective

TravelNest is a standalone web-based marketplace that fills the gap in the heavy vehicle rental sector. It operates as an intermediary platform connecting independent bus owners with customers. It integrates with external payment gateways (PayHere) and mapping services (Google Maps) to provide a seamless experience.

## 2.2 Product Functions

The major functions of the system include:

1. **Search & Discovery:** Search for buses based on location, date, and capacity.
2. **Quotation Generation:** Automated calculation of rental costs including fuel, driver, and tolls.
3. **Booking Management:** Real-time availability checking and reservation.
4. **Owner Dashboard:** Fleet management, availability setting, and booking acceptance.
5. **Payment Processing:** Secure advance payments and digital receipts.
6. **Admin Verification:** Validation of owner documents (insurance, permits).

## 2.3 User Classes and Characteristics

User Class	Characteristics	Responsibilities
Customer	Occasional users (schools, corporate, private groups). Tech literacy varies.	Search for buses, request quotes, make bookings, rate services.
Bus Owner	Individuals or transport companies owning one or more buses.	Register vehicles, set pricing/availability, manage bookings, upload verification docs.
System Admin	Technical administrators.	Verify owner documents, manage disputes, monitor platform health, generate reports.

## 2.4 Operating Environment

- Client Side: Modern web browsers (Chrome, Firefox, Safari, Edge) on Desktop and Mobile devices. PWA support for mobile installation.
- Server Side: Node.js runtime environment hosted on cloud infrastructure (Vercel/AWS).
- Database: PostgreSQL 16+.
- Network: Requires stable internet connection (4G/Fiber) for real-time booking and mapping.

## 2.5 Design and Implementation Constraints

- **Regulatory Compliance:** Must adhere to Sri Lankan transport regulations regarding charter services.
- **Language Support:** Interface must support English, Sinhala, and Tamil.
- **Payment Security:** Must comply with PayHere security standards.
- **Image Storage:** Limited by AWS S3 free/tier limits during development.

## 2.6 User Documentation

- **Online Help Center:** Integrated FAQs and "How-to" guides for owners and customers.
- **Contextual Tooltips:** Inline help within the application forms.

## 2.7 Assumptions and Dependencies

- **Assumptions:** Bus owners have access to smartphones/computers to manage listings.
- **Dependencies:** Reliability of Google Maps API for distance calculation and PayHere for transaction processing.

## 3 External Interface Requirements

### 3.1 User Interfaces

- Frontend Framework: React.js with Next.js.
- Styling: Tailwind CSS for responsive, mobile-first design.
- Layout:
  - Customer View: Clean, search-centric homepage with easy filtering.
  - Owner Portal: Dashboard style with tabular data for bookings and fleet.
- Multilingual: Language switcher (EN/SI/TA) prominently displayed.

### 3.2 Hardware Interfaces

The system is software-based and does not require specific custom hardware. It runs on standard commodity hardware (Smartphones, PCs, Tablets).

### 3.3 Software Interfaces

- **Operating System:** Platform independent (Web-based).
- **Database:** PostgreSQL for persistent data storage.
- **APIs:**
  - Google Maps API: For geocoding, distance matrix, and route visualization.
  - PayHere API: For processing credit/debit card transactions.
  - Twilio API: For SMS notifications and OTP.
  - AWS S3 API: For uploading and retrieving vehicle images.

### 3.4 Communication Interfaces

- **Protocol:** HTTP/HTTPS (Secure Sockets Layer) for all client-server communication.
- **Data Format:** JSON for API responses.
- **Real-time:** WebSockets (or polling) for instant booking notifications.

## 4 System Features

### 4.1 User Registration and Profile Management

- **Description:** Allows customers and bus owners to create accounts.
- **Priority:** High
- **Functional Requirements:**
  - The system shall allow users to register using Email or Mobile Number.
  - The system shall verify mobile numbers via OTP (Twilio).
  - The system shall allow Bus Owners to upload verification documents (NIC, Business Registration).
  - The system shall support password recovery.

### 4.2 Vehicle Listing and Management

- **Description:** Enables owners to list buses with detailed specifications.
- **Priority:** High
- **Functional Requirements:**
  - The system shall allow owners to add vehicles with attributes: Capacity (16-50+), AC/Non-AC, Amenities (TV, Sound System), and Photos.
  - The system shall validate uploaded images (format: JPG/PNG, size limit).
  - The system shall allow owners to set "Base Location" and "Operating Areas".
  - The system shall allow owners to update availability calendars (Block dates).

### 4.3 Search and Quotation System

- **Description:** Core engine for finding buses and calculating prices.
- **Priority:** Critical
- **Functional Requirements:**
  - The system shall accept itinerary details: Pickup, Destination, Intermediate Stops, Date/Time.
  - The system shall use Google Maps API to calculate total distance and estimated duration.
  - The system shall generate an all-inclusive quotation based on:  $(\text{Distance} * \text{Rate per km}) + \text{Driver Allowance} + \text{Agency Fee}$ .
  - The system shall allow filtering by Price, Rating, and Amenities.

### 4.4 Booking and Payment Processing

- **Description:** Handling the transaction workflow.
- **Priority:** Critical
- **Functional Requirements:**
  - The system shall allow customers to select a bus and proceed to "Book Now".
  - The system shall redirect to PayHere gateway for advance payment.
  - The system shall generate a digital booking confirmation/invoice upon success.
  - The system shall send SMS/Email notifications to both Owner and Customer.
  - The system shall update the vehicle availability calendar automatically.

## 4.5 Admin Verification and Oversight

- **Description:** Back-office functions for trust and safety.
- **Priority:** Medium
- **Functional Requirements:**
  - The system shall provide a dashboard for Admins to view "Pending" owner registrations.
  - The system shall allow Admins to "Approve" or "Reject" documents (Insurance, Route Permits).
  - The system shall allow Admins to suspend users for policy violations.
  - The system shall generate monthly revenue and usage reports.

# 5 Other Nonfunctional Requirements

## 5.1 Performance Requirements

- **Response Time:** System should load initial content within 2 seconds on 4G networks.
- **Throughput:** Support up to 500 concurrent users during peak seasons.
- **Scalability:** The architecture (Node.js/Cloud) allows horizontal scaling.

## 5.2 Safety Requirements

- **Data Backup:** Automated daily backups of the PostgreSQL database.
- **Failover:** Graceful error handling if external APIs (Google Maps) are down.

## 5.3 Security Requirements

- **Encryption:** All data in transit must be encrypted via TLS 1.2+.
- **Authentication:** JWT (JSON Web Tokens) for secure session management.
- **Data Privacy:** Passwords must be hashed (e.g., bcrypt) before storage. Personal data must be handled according to privacy laws.

## 5.4 Software Quality Attributes

- **Usability:** Mobile-first design ensures high usability for non-technical drivers/owners.
- **Reliability:** 99.9% uptime target for the hosting platform.
- **Maintainability:** Codebase structured using MVC pattern and modular React components.

## 5.5 Business Rules

- **Cancellation Policy:** Full refund if cancelled 2 weeks before trip; 50% refund if cancelled 1 week before trip.
- **Commission:** Platform retains a fixed percentage (e.g., 5-10%) of the booking fee.

## Appendix A – Glossary

- **PWA:** Progressive Web Application.
- **OTP:** One-Time Password.
- **SaaS:** Software as a Service.
- **Charter:** Hiring a vehicle for exclusive use by a group.
- **API:** Application Programming Interface.
- **JWT:** JSON Web Token.

## Appendix B – Analysis Models

Use Case Diagram

Class Diagram

ER Diagram

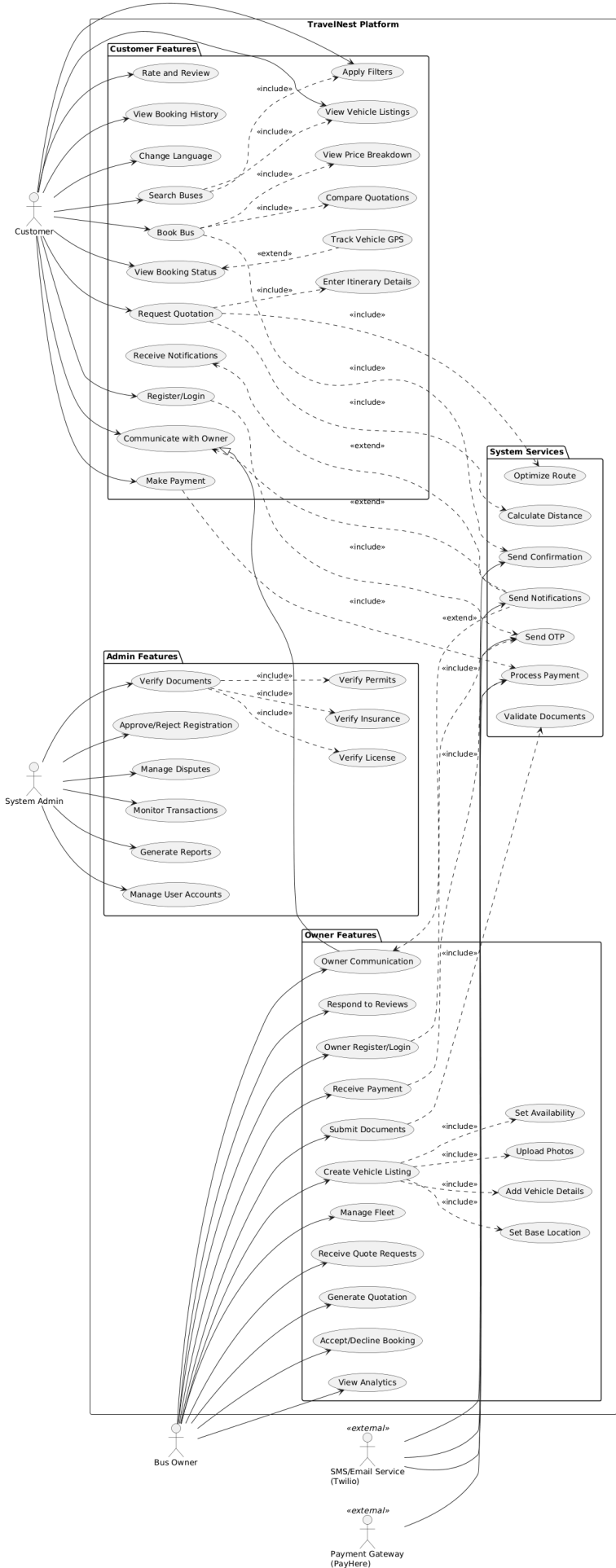


Figure 2: TravelNest Use Case Diagram



commission rules		
rule_id $\phi$	Integer	
rule_name	varchar(100) NN	
commission_type	enum(percentage, fixed, tiered) NN	
commission_value	decimal(10,2) NN	
applicable_to	enum(all, vehicle_type, owner_type, booking_amount)	
min_booking_amount	decimal(10,2)	
max_booking_amount	decimal(10,2)	
is_active	boolean	
effective_from	date	
effective_to	date	
created_at	timestamp	
updated_at	timestamp	

amenities		
amenity_id $\phi$	Integer	
amenity_name	varchar(100) NN	
amenity_icon	varchar(100)	
description	text	
is_active	boolean	
created_at	timestamp	

users		
user_id $\phi$	Integer	
email	varchar(255) NN	
phone	varchar(20) NN	
password_hash	varchar(255) NN	
user_type	enum(customer, owner, admin) NN	
first_name	varchar(100) NN	
last_name	varchar(100) NN	
profile_picture	varchar(255)	
preferred_language	enum(en, si, ta)	
is_verified	boolean	
is_active	boolean	
created_at	timestamp	
updated_at	timestamp	
last_login	timestamp	

otp verifications		
otp_id $\phi$	Integer	
user_id $\phi$	Integer	
otp_code	varchar(6) NN	
otp_type	enum(registration, login, password_reset, phone_verification) NN	
phone_number	varchar(20)	
email	varchar(255)	
is_verified	boolean	
expires_at	timestamp NN	
created_at	timestamp	

customers		
customer_id $\phi$	Integer	
user_id $\phi$	Integer	
organization_name	varchar(255)	
organization_type	enum(school, religious, corporate, event_planner, individual) NN	
address_line1	varchar(255)	
address_line2	varchar(255)	
city	varchar(100)	
district	varchar(100)	
postal_code	varchar(10)	
created_at	timestamp	
updated_at	timestamp	

bus owners		
owner_id $\phi$	Integer	
user_id $\phi$	Integer	
business_name	varchar(255) NN	
business_type	enum(company, individual) NN	
business_registration_number	varchar(100)	
tax_identification_number	varchar(100)	
address_line1	varchar(255) NN	
address_line2	varchar(255)	
city	varchar(100) NN	
district	varchar(100) NN	
postal_code	varchar(10)	
base_location_lat	decimal(10,8)	
base_location_lng	decimal(11,8)	
verification_status	enum(pending, approved, rejected, suspended)	
verification_notes	text	
rating_average	decimal(3,2)	
total_bookings	Integer	
created_at	timestamp	
updated_at	timestamp	

notifications		
notification_id $\phi$	Integer	
user_id $\phi$	Integer	
notification_type	enum(booking, payment, quotation, review, system, promotional) NN	
title	varchar(255) NN	
message	text NN	
related_entity_type	enum(booking, quotation, payment, review, vehicle)	
related_entity_id	Integer	
channel	enum(in_app, email, sms, push) NN	
is_read	boolean	
read_at	timestamp	
sent_at	timestamp	
delivery_status	enum(pending, sent, delivered, failed)	
created_at	timestamp	

system settings		
setting_id $\phi$	Integer	
setting_key	varchar(100) NN	
setting_value	text	
setting_type	enum(string, number, boolean, json)	
description	text	
is_public	boolean	
updated_by $\phi$	Integer	
created_at	timestamp	
updated_at	timestamp	

audit logs		
log_id $\phi$	Integer	
user_id $\phi$	Integer	
action_type	enum(create, update, delete, login, logout, payment, booking) NN	
entity_type	varchar(50)	
entity_id	Integer	
old_values	json	
new_values	json	
ip_address	varchar(45)	
user_agent	varchar(500)	
created_at	timestamp	

analytics events		
event_id $\phi$	Integer	
user_id $\phi$	Integer	
event_name	varchar(100) NN	
event_category	varchar(50)	
event_data	json	
session_id	varchar(100)	
page_url	varchar(500)	
referrer_url	varchar(500)	
device_type	enum(mobile, tablet, desktop)	
browser	varchar(50)	
ip_address	varchar(45)	
created_at	timestamp	

owner documents		
document_id $\phi$	Integer	
owner_id $\phi$	Integer	
document_type	enum(license, insurance, permit, registration, tax, other) NN	
document_name	varchar(255) NN	
document_url	varchar(500) NN	
document_number	varchar(100)	
issue_date	date	
expiry_date	date	
verification_status	enum(pending, verified, rejected, expired)	
verified_by $\phi$	Integer	
verified_at	timestamp	
verification_notes	text	
uploaded_at	timestamp	

vehicles		
vehicle_id $\phi$	Integer	
owner_id $\phi$	Integer	
vehicle_registration_number	varchar(30) NN	
vehicle_type	enum(mini_bus, standard_bus, luxury_bus, semi_luxury_bus) NN	
make	varchar(100) NN	
model	varchar(100) NN	
year	Integer NN	
seating_capacity	Integer NN	
color	varchar(50)	
ac_type	enum(ac, non_ac, both) NN	
vehicle_condition	enum(excellent, good, fair)	
description	text	
base_price_per_km	decimal(10,2)	
base_price_per_day	decimal(10,2)	
driver_allowance_per_day	decimal(10,2)	
is_active	boolean	
is_verified	boolean	
gps_enabled	boolean	
created_at	timestamp	
updated_at	timestamp	

quotation requests		
request_id $\phi$	Integer	
customer_id $\phi$	Integer	
pickup_location	varchar(255) NN	
pickup_lat	decimal(10,8)	
pickup_lng	decimal(11,8)	
destination_location	varchar(255) NN	
destination_lat	decimal(10,8)	
destination_lng	decimal(11,8)	
start_date	date NN	
end_date	date NN	
start_time	time	
estimated_distance	decimal(10,2)	
estimated_duration	Integer	
passenger_count	Integer NN	
vehicle_type_preference	enum(mini_bus, standard_bus, luxury_bus, semi_luxury_bus, any)	
ac_preference	enum(ac, non_ac, any)	
special_requirements	text	
status	enum(pending, quoted, expired, cancelled)	
created_at	timestamp	
updated_at	timestamp	

vehicle photos		
photo_id $\phi$	Integer	
vehicle_id $\phi$	Integer	
photo_url	varchar(500) NN	
photo_type	enum(exterior, interior, seats, dashboard, other) NN	
is_primary	boolean	
display_order	Integer	
uploaded_at	timestamp	

vehicle availability		
availability_id $\phi$	Integer	
vehicle_id $\phi$	Integer	
start_date	date NN	
end_date	date NN	
is_available	boolean	
notes	text	
created_at	timestamp	
updated_at	timestamp	

vehicle amenities		
vehicle_amenity_id $\phi$	Integer	
vehicle_id $\phi$	Integer	
amenity_id $\phi$	Integer	

itinerary stops		
stop_id $\phi$	Integer	
request_id $\phi$	Integer	
stop_order	Integer NN	
location_name	varchar(255) NN	
location_lat	decimal(10,8)	
location_lng	decimal(11,8)	
stop_duration	Integer	
notes	text	

quotations		
quotation_id $\phi$	Integer	
request_id $\phi$	Integer	
owner_id $\phi$	Integer	
vehicle_id $\phi$	Integer	
vehicle_rental_cost	decimal(10,2) NN	
driver_cost	decimal(10,2) NN	
fuel_cost	decimal(10,2) NN	
toll_charges	decimal(10,2)	
permit_fees	decimal(10,2)	
other_charges	decimal(10,2)	
total_cost	decimal(10,2) NN	
tax_amount	decimal(10,2)	
grand_total	decimal(10,2) NN	
cost_breakdown	json	
validity_period	Integer	
notes	text	
status	enum(sent, viewed, accepted, rejected, expired)	
created_at	timestamp	
expires_at	timestamp NN	

bookings		
booking_id $\phi$	Integer	
booking_reference	varchar(50) NN	
quotation_id $\phi$	Integer	
customer_id $\phi$	Integer	
owner_id $\phi$	Integer	
vehicle_id $\phi$	Integer	
pickup_location	varchar(255) NN	
pickup_lat	decimal(10,8)	
pickup_lng	decimal(11,8)	
destination_location	varchar(255) NN	
destination_lat	decimal(10,8)	
destination_lng	decimal(11,8)	
start_date	date NN	
end_date	date NN	
start_time	time	
passenger_count	Integer NN	
total_amount	decimal(10,2) NN	
booking_status	enum(confirmed, in_progress, completed, cancelled, disputed)	
payment_status	enum(pending, partial_paid, refunded)	
cancellation_reason	text	
cancelled_by $\phi$	Integer	
cancelled_at	timestamp	
completed_at	timestamp	
created_at	timestamp	
updated_at	timestamp	

Figure 3: TravelNest ER Diagram Part 1



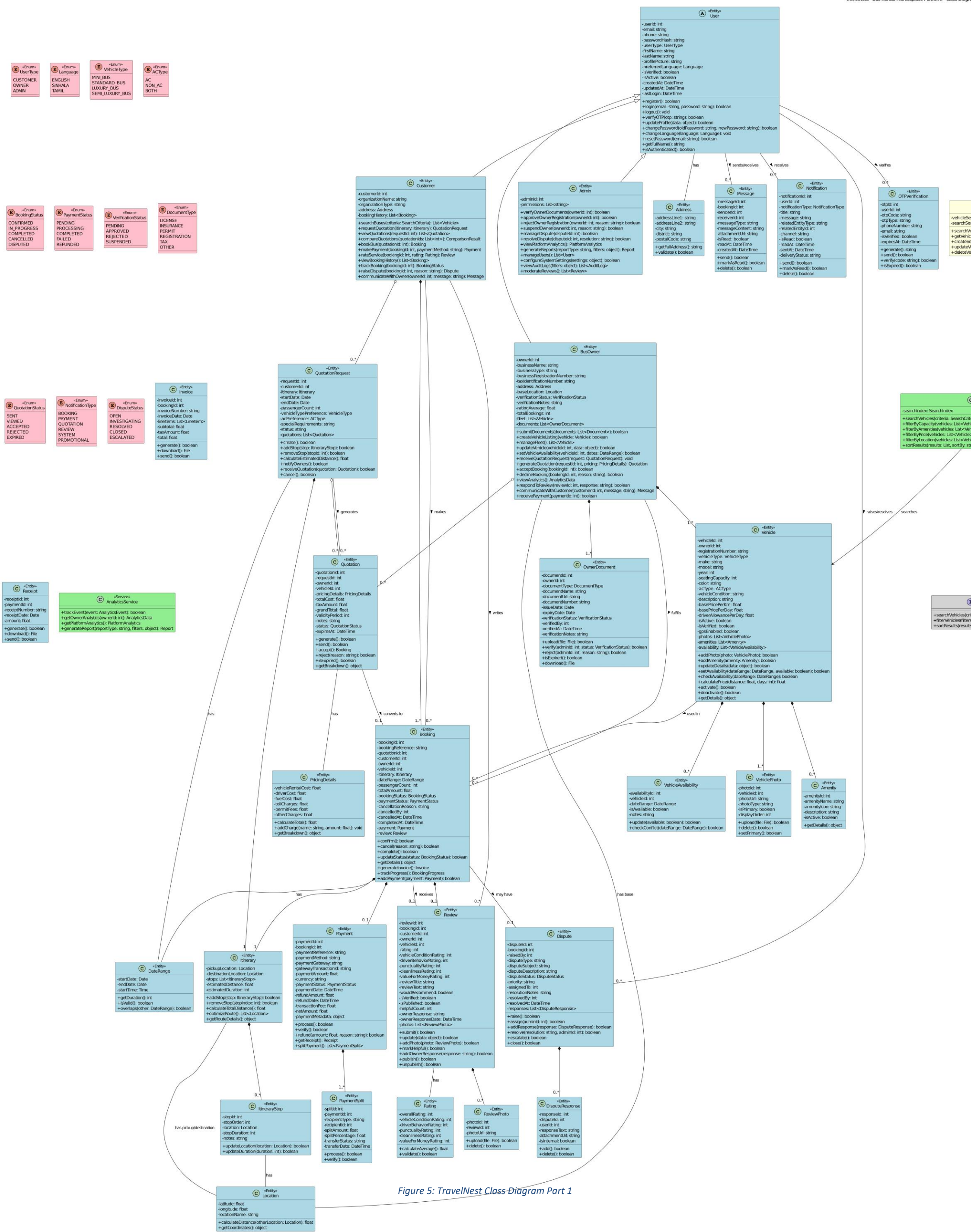


Figure 5: TravelNest Class Diagram Part 1



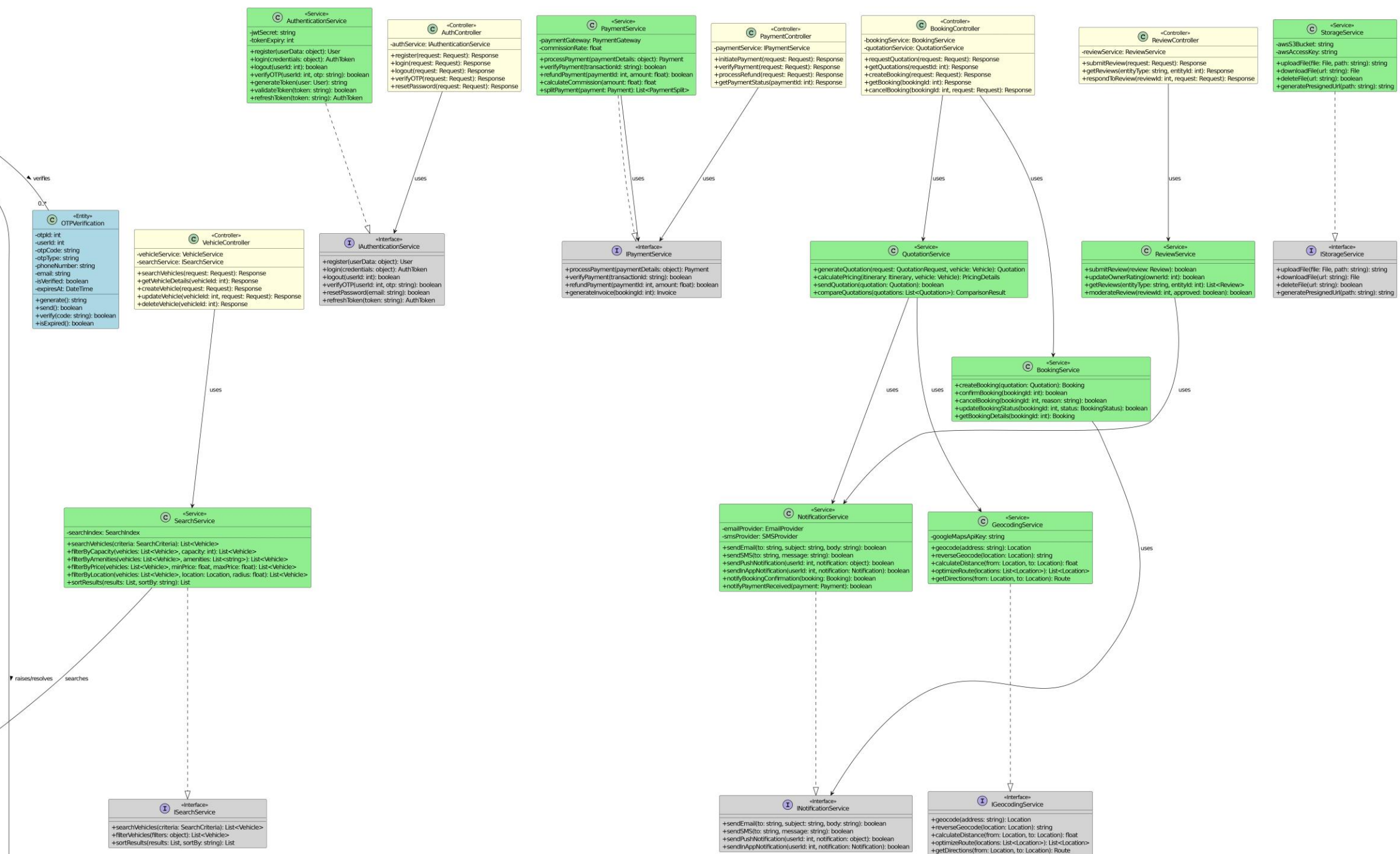


Figure 6: TravelNest Class Diagram Part 2