Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

This document defines the functional and non-functional requirements for a peer-to-peer vehicle renting platform operating in Egypt.
The system enables vehicle owners (individuals and companies) to list vehicles for rent and allows renters to search, book, and rent vehicles securely.

This SRS serves as the reference document for design, development, testing, and deployment.

1.2 Product Vision

Build a production-ready peer-to-peer vehicle rental platform (similar to Turo) tailored for the Egyptian market.

The system should:

- Enable individuals and companies to rent out vehicles
- Provide secure booking and payment
- Allow negotiation between renter and owner
- Maintain trust via ratings and reviews
- Support future AI-based querying using RAG (e.g., policies, car availability, pricing)

1.3 Scope

The system will:

- Allow user registration and authentication
- Support multiple user roles
- Enable vehicle listing and management
- Support rental booking and approval workflow
- Simulate wallet-based payments (MVP)
- Provide rating and feedback system
- Include admin moderation tools

Future scope:

- Real payment integration
- Real-time chat
- RAG-based intelligent assistant
- Mobile application

2. Overall Description

2.1 User Types

1. **Renter**
    a. Registers account
    b. Searches vehicles
    c. Sends rental request
    d. Pays via wallet
    e. Leaves feedback
2. **Owner (Individual)**
    a. Lists vehicles
    b. Sets availability and pricing
    c. Approves/rejects rental requests
    d. Receives payment
    e. Leaves feedback
3. **Company Owner**
    a. Similar to Owner
    b. May manage multiple vehicles under one company profile
4. **Admin**
    a. Manages categories
    b. Verifies owners
    c. Suspends users or vehicles
    d. Moderates feedback

Note:
A user may act as both renter and owner.

2.2 Core Workflow

Rental Workflow

1. User registers/logs in
2. User browses vehicle categories
3. User selects a vehicle
4. User selects rental period
5. System checks availability
6. User submits rental request
7. Owner reviews request
8. Owner approves or rejects
9. If approved → User pays via wallet
10. Rental becomes active
11. Rental completes after end date
12. Both parties leave feedback

3. Functional Requirements

3.1 User Management

FR-1: The system shall allow users to register using email and password.
FR-2: The system shall allow users to log in and log out securely.
FR-3: The system shall support role-based access control (Renter, Owner, Admin).
FR-4: The system shall allow a user to become an owner.

3.2 Vehicle Management

FR-5: The system shall allow owners to add a vehicle.
FR-6: Each vehicle must have:

- VIN (unique, 17 characters)
- Brand
- Model
- Year
- Category
- Location
- Base price per day
- Minimum rental price

FR-7: Owners shall upload photos for vehicles.
FR-8: Owners shall define availability periods.
FR-9: Owners shall activate/deactivate vehicles.

3.3 Search & Browsing

FR-10: Users shall browse vehicles by category.
FR-11: Users shall filter vehicles by:

- Location
- Price range
- Date availability

- Brand

3.4 Rental Management

FR-12: Users shall request vehicle rental for specific dates.
FR-13: The system shall prevent double booking.
FR-14: Owners shall approve or reject rental requests.
FR-15: Approved rentals shall require payment before activation.
FR-16: Rental status shall be tracked:

- Pending
- Approved
- Paid
- Active
- Completed
- Cancelled

3.5 Wallet System (MVP Payment)

FR-17: Users shall have a wallet balance.
FR-18: Users shall add funds to their wallet (simulation).
FR-19: Upon rental payment:

- Renter wallet is debited
- Owner wallet is credited
- Transaction record is created

3.6 Feedback System

FR-20: After rental completion, both parties shall leave a rating (1–5).
FR-21: Feedback must be tied to a completed rental.

3.7 Admin Controls

FR-22: Admin shall manage vehicle categories.
FR-23: Admin shall suspend users.
FR-24: Admin shall remove inappropriate vehicles or feedback.
FR-25: Admin shall verify owner accounts.

3.8 Future AI Integration (RAG)

FR-26: The system shall store structured data to support future AI querying.
FR-27: The system shall allow integration of a Retrieval-Augmented Generation module for policy and vehicle data queries.

4. Non-Functional Requirements

4.1 Performance

NFR-1: System shall respond to API requests within 2 seconds under normal load.
NFR-2: Search operations must be optimized for scalability.

4.2 Security

NFR-3: All endpoints must require authentication except registration/login.
NFR-4: Passwords must be encrypted.
NFR-5: Role-based authorization must be enforced.
NFR-6: Prevent double booking through transactional integrity.

4.3 Scalability

NFR-7: System shall support 10,000 concurrent users.
NFR-8: System architecture must allow horizontal scaling.

4.4 Data Integrity

NFR-9: VIN must be unique.
NFR-10: A vehicle must not have overlapping active rentals.
NFR-11: Wallet transactions must be atomic.

4.5 Maintainability

NFR-12: Codebase shall follow clean architecture principles.
NFR-13: Business logic shall be separated from controllers.
NFR-14: The system shall support modular extension (e.g., payment gateway, AI).