1. Introduction

1.1 Purpose

The purpose of the **Vehical_Entry_System** is to digitize and automate the management of vehicles entering and exiting a specific premise (e.g., corporate office, residential complex, or parking lot). This system replaces manual logbooks, enhances security, and provides real-time data on vehicle occupancy.

1.2 Scope

- **In Scope:**
    - Digital logging of vehicle entry and exit times.
    - Capturing driver and vehicle details (License Plate, Phone Number, Name).
    - Role-based access (Admin vs. Security Guard).
    - Historical reporting and search functionality.
- **Out of Scope (Phase 1):**
    - Automatic License Plate Recognition (ALPR) via camera integration (Future Scope).
    - Payment gateway integration for parking fees.

---

2. User Personas

Role	Description	Key Responsibilities
Administrator	Facility Manager or System Owner	Manage users, view global analytics, export reports, configure system settings.
Security Guard	Gatekeeper / Operator	Log vehicle entries, validate exits, search for active vehicles.

---

3. Functional Requirements

3.1 Authentication & Authorization

- **FR-01:** System must allow users to log in using a secure username and password.
- **FR-02:** System must distinguish between Admin and Guard privileges.

3.2 Vehicle Entry (Check-In)

- **FR-03:** The "Add Entry" interface must capture the following fields:
    - Vehicle Number (License Plate)
    - Vehicle Type (2-wheeler, 4-wheeler, Truck)
    - Driver Name
    - Contact Number
    - Purpose of Visit / Flat Number
- **FR-04:** System must automatically timestamp the "Time In" upon submission.
- **FR-05:** System must generate a unique "Token ID" or "Pass ID" for the visit.

3.3 Vehicle Exit (Check-Out)

- **FR-06:** Guards must be able to search for a vehicle by License Plate or Token ID.
- **FR-07:** System must allow the guard to mark a vehicle as "Exited."
- **FR-08:** System must automatically timestamp the "Time Out" and calculate the "Total Duration."

3.4 Dashboard & Reporting

- **FR-09:** Dashboard must display "Currently Parked Vehicles" count.
- **FR-10:** Admin must be able to filter records by date range and export data (CSV/Excel).

---

4. Non-Functional Requirements

- **NFR-01 Performance:** The system should load the dashboard in under 2 seconds.
- **NFR-02 Reliability:** The database must ensure zero data loss during concurrent entry/exit logging.
- **NFR-03 Scalability:** The database schema should support up to 100,000 historical records without significant query lag.
- **NFR-04 Responsiveness:** The UI must be mobile-responsive (tablet/phone friendly) for guards using handheld devices.

---

5. System Architecture

5.1 Technology Stack (Recommended)

- **Frontend:** HTML5, CSS3, Bootstrap (for responsiveness), JavaScript.
- **Backend:** Python (Django/Flask) or Node.js (Express).
- **Database:** MySQL or PostgreSQL (Relational Data).

5.2 Database Schema (Conceptual)

Table: Users

- user_id (PK)
- username
- password_hash
- role (Admin/Guard)

Table: Vehicle_Logs

- log_id (PK)
- vehicle_number (VARCHAR)
- driver_name (VARCHAR)
- entry_time (DATETIME)
- exit_time (DATETIME, Nullable)
- status (ENUM: 'Parked', 'Exited')

---

6. UI/UX Flow

6.1 Entry Flow

1. **Dashboard** $\rightarrow$ Click **"New Entry"** Button.
2. **Form Modal** opens.
3. Guard inputs **Vehicle No** and **Driver details**.
4. Click **Submit**.
5. Success Toast appears; Dashboard count increments by +1.

6.2 Exit Flow

1. **Dashboard** $\rightarrow$ Search **Vehicle No**.
2. System displays active record.
3. Click **"Mark Exit"**.

4. System confirms Time Out.
5. Record moves from "Active" list to "History."

---

7. Future Roadmap (Phase 2)

- **ALPR Integration:** Use OpenCV to read license plates automatically from a video feed.
- **SMS Notifications:** Send entry/exit alerts to the resident being visited.
- **QR Code Passes:** Pre-approve visitors via QR codes sent to their phones.