Project Report: QuickPark – Vehicle Parking Web Application

Submitted by: Anirudhha U

Roll No: 24F2005215

Course: Modern Application Development Project I

Project: Vehicle Parking App - V1

Date: 31-07-2025

**1. Introduction**

* QuickPark is a web-based application for managing vehicle parking facilities.
* Developed using Python Flask for backend logic and SQLite3 as the database.
* It provides role-based access for Admins and Users.
* Focuses on streamlined booking, releasing, and tracking of parking spaces.

**2. Technologies Used**

* Backend: Python Flask framework
* Frontend: HTML5, CSS3, Jinja2 templates
* Database: SQLite3
* Session Management: Flask built-in session
* Styling: Custom CSS (consistent layout for Admin and User views)

**3. Database Schema**

**a. Users Table**

* Stores credentials and profile information.
* Fields:
  + id (INTEGER, PRIMARY KEY, AUTOINCREMENT)
  + username (TEXT, UNIQUE)
  + password (TEXT)
  + full\_name (TEXT)
  + address (TEXT)
  + pin\_code (TEXT)
  + is\_admin (INTEGER, 0 for user, 1 for admin)

**b. Parking\_Details Table**

* Represents each individual parking spot.
* Fields:
  + id (INTEGER, PRIMARY KEY, AUTOINCREMENT)
  + place\_name (TEXT)
  + total\_spots (INTEGER)
  + available\_spots (INTEGER)
  + occupied\_spots (INTEGER)
  + cost\_per\_hour (REAL)

**c. Parking\_History Table**

* Records every parking session by users.
* Fields:
  + id (INTEGER, PRIMARY KEY, AUTOINCREMENT)
  + user\_id (INTEGER, FOREIGN KEY → users.id)
  + spot\_id (INTEGER, FOREIGN KEY → Parking\_Details.id)
  + park\_time (DATETIME)
  + release\_time (DATETIME)
  + amount (REAL)
  + status (TEXT: "Parked In" / "Parked Out")

**3. Core Features Implemented**

* User Authentication:
  + Signup and Login functionality.
  + Differentiates between Admin and User roles.
  + Session-based access control.
* Admin Dashboard:
  + View, Add, Edit, and Delete parking lots.
  + Summary of total lots, users, and spots (occupied/available).
  + Access to all user data and search functionalities.
* Parking Spot Booking:
  + Users can view and book available parking spots.
  + Booking details are stored in the Parking\_History table.
* Spot Release & Billing:
  + Users can release a spot, and the system calculates the duration and cost.
  + A minimum billing of 5 minutes is applied, charged per hour rate.
* Parking History:
  + Users can see their complete booking and release history.
  + Admins can view all users' parking activity logs.
* Search Functionality (Admin):
  + Admins can search users by:
    - User ID
    - Full Name
    - Pincode
    - Spot ID
    - Place Name
    - Parking Status (Parked In / Out)
  + Uses SQL joins across multiple tables to show combined results.
* Summary Pages:
  + Admins see system-wide statistics.
  + Users see their own booking stats and status.

**4. Recommended Features Included**

* Backend Validation:
  + Checks for missing fields, invalid credentials, duplicate users, etc.
* Frontend Validation:
  + All form fields use HTML5 validation (required, type, etc.).
* Flash Messages:
  + Used throughout the application for success and error feedback.
* Minimum Time Billing Logic:
  + Ensures fair cost calculation by setting a minimum parking duration.
* Modular HTML Templates:
  + Consistent header, navigation, and layout structure for both roles.
* Search Optimizations:
  + Dynamic search logic with flexible filters and meaningful results
  + Drop down box used in the search tab.
* Attractive Styling:
  + User-friendly, color-coded interfaces.
  + Uniform layout using reusable CSS components.

**5. Conclusion**

* QuickPark offers a complete solution for managing parking operations.
* It covers all essential functionalities including user management, spot allocation, billing, and activity tracking.
* Built with scalability in mind — potential for adding API endpoints, charting tools, or mobile support in the future.
* All core and several recommended features have been implemented making the system both practical ,professional and user friendly.