

# Vehicle Parking Management System

Project Documentation - Version 1.0

## Author Information

**Name:** Samar Nathani

**Roll Number:** 24f2006661

**Email:** 24f2006661@ds.study.iitm.ac.in

**Institution:** Indian Institute of Technology Madras

This project represents my understanding and implementation of modern web development principles, combining database management with intuitive user interface design to solve real-world parking management challenges.

## Project Description

The Vehicle Parking Management System is a comprehensive web-based solution designed to streamline parking operations in urban environments. This application addresses the growing need for efficient parking space management by providing both administrative control and user-friendly booking interfaces.

The system enables administrators to manage multiple parking lots, monitor occupancy rates, and generate detailed revenue reports, while allowing users to search for available spaces, make reservations, and track their parking history. The application focuses on real-time data management and provides visual analytics to support decision-making processes.

Built with scalability in mind, this solution can be easily adapted for various parking scenarios, from shopping centers to residential complexes, making it a versatile tool for modern parking management needs.

## Technologies Used

### Backend Framework

**Flask** - Lightweight Python web framework for rapid development and deployment

### Database

**SQLite3** - Embedded database with direct SQL queries for better DBMS understanding

### Data Visualization

**Matplotlib** - Python plotting library for generating charts and graphs

### Frontend Styling

**Bootstrap & CSS** - Responsive design framework enhanced with custom styling

### Session Management

**Flask Sessions** - Secure user authentication and role-based access control

### File Operations

**OS Module** - File system operations for chart generation and database management

## Technology Choices Rationale

I chose **SQLite3 with direct SQL queries** over SQLAlchemy because it provided better learning opportunities for database management concepts learned in DBMS coursework. The query-based approach offered more control and transparency in database operations.

For visualization, **matplotlib with Agg backend** was implemented to generate static charts saved in the static folder, ensuring compatibility with web deployment without requiring GUI dependencies.

Bootstrap and CSS styling were enhanced with AI assistance to create professional-looking interfaces, acknowledging that frontend design isn't my strongest skill but ensuring user experience wasn't compromised.

## Database Schema Design

The database schema follows relational design principles with four main entities ensuring data integrity through foreign key relationships and appropriate constraints.

### Core Tables Structure

#### USERS Table

Stores user credentials and profile information with role-based access control distinguishing between admin and regular users.

#### Parking\_lot Table

Contains parking facility information including location details, pricing, and capacity management.

#### Parking\_spot Table

Individual parking spaces linked to lots with availability status tracking (Available/Occupied).

#### Reserve\_parking\_spot Table

Reservation records with timestamps, cost calculations, and vehicle information for complete transaction tracking.

### Design Rationale

The schema separates parking lots from individual spots to support dynamic capacity management. This design allows administrators to modify lot sizes while maintaining referential integrity. The reservation table captures both active bookings and historical data for comprehensive reporting.


## Architecture and Features

### MVC Architecture Implementation


The application follows the Model-View-Controller pattern with clear separation of concerns. The **app.py** file serves as the main controller handling all route definitions and business logic, while **database.py** represents the model layer for database operations. HTML templates in the templates folder constitute the view layer, enhanced with Bootstrap styling stored in the static directory.

Controllers are organized by user roles with dedicated route groups for admin and user functionalities. This separation ensures security through session-based authentication and provides intuitive navigation paths for different user types.


### Key Features Implementation

**Authentication System**


Secure login/logout functionality with session management and role-based access control

**Parking Lot Management**


Complete CRUD operations for parking facilities with dynamic spot allocation

**Smart Search**


Location and pincode-based search functionality for finding available parking spaces

**Visual Analytics**

Dynamic chart generation using matplotlib for occupancy rates and revenue analysis

**Cost Calculation**

Automated billing based on parking duration with hourly rate calculations

**Responsive Design**

Mobile-friendly interface using Bootstrap framework for cross-platform compatibility

### Advanced Features

**Real-time Availability Tracking:** The system maintains live updates of parking spot availability, preventing double bookings and providing accurate occupancy information.

**Historical Data Management:** Complete transaction history with detailed reporting capabilities for both administrative oversight and user reference.

**Dynamic Chart Generation:** Charts are generated on-demand and cached in the static folder, with automatic cleanup to prevent storage bloat while ensuring up-to-date visualizations.

## Project Demonstration

Watch the complete walkthrough of the Vehicle Parking Management System showcasing all features and functionalities.

 [View Demo Video](#)

The video demonstrates both admin and user interfaces, including parking lot management, booking processes, and analytical reporting features.