Project Report

Vehicle Parking App - V1

Name: Biplab Keshari Mohanty
Roll Number: 24f2008369
Student E-Mail: 24f2008369@ds.study.iitm.ac.in

Problem Statement

With the growing number of vehicles in urban areas, finding and managing parking spaces efficiently has become a significant challenge. The aim of this project is to build a smart and responsive vehicle parking system where users can easily book parking spots in real-time and administrators can manage parking lots, monitor status, and oversee transactions.

Approach/Methodology

To tackle the problem effectively, the project was divided into modules and developed in a structured manner:

1. **Authentication System**: Implemented using Flask Login and Flask sessions for secure login/logout functionality. Role-based access to distinguish between user and admin.
2. **Database Design**: Designed an ER model with tables for users, parking lots, parking spots, and transactions. Used SQLite for development with schema migration handled manually.
3. **Parking Lot and Spot Management**: Admins can add, update, and delete parking lots. Parking spots are automatically generated or adjusted based on the max capacity set by the admin.
4. **Booking and Release Flow**: Users can book available spots. Upon release, the system calculates cost and deducts it from the user's wallet.
5. **Wallet Integration**: Users can view wallet balance, add funds, and see a history of transactions. All operations are recorded in a transaction table with timestamps and descriptions.
6. **Google Maps Integration**: Admins can add location links during lot creation. Users can open directions to the lot via a button on the dashboard.
7. **Frontend Development**: Built using HTML, CSS, and Bootstrap for a clean UI. Custom flash popup messages are shown for feedback and errors.
8. **Security and User Experience:** Error pages like 404 are handled gracefully. Flash popup messages and session validation provide user-friendly interactions and prevent unauthorized access.

Frameworks and Libraries Used

1. Web Framework and Server-Side Logic

- **Flask**: Core web framework used to handle routing, requests, session management, and overall backend logic.
- **Jinja2**: Templating engine used to dynamically render HTML pages with data from the backend.
- **Werkzeug**: Internal Flask dependency used for secure routing and utility functions.

2. Database and Data Storage

- **SQLite3**: Lightweight embedded database for storing user, parking, wallet, and transaction data.
- **json**: Used to read and manage admin credentials from a JSON file securely.
- 3. Authentication and Security
- **bcrypt**: Used for hashing and verifying user passwords securely.
- **Flask-Login**: Manages user sessions and login state.
- **os and pathlib**: Used for secure file path resolution and image uploads.
- **secure_filename (from werkzeug.utils)**: Prevents directory traversal and filename injection when saving files.

4. Frontend and UI Styling

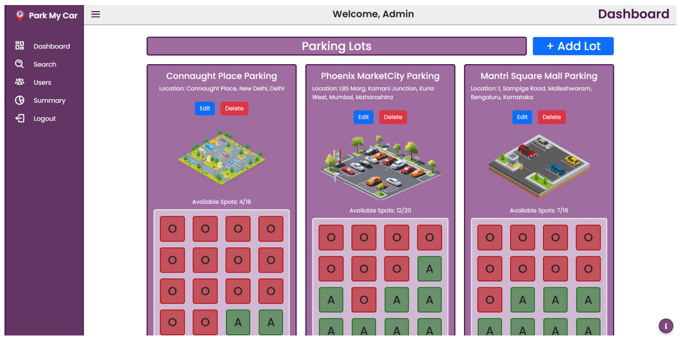
- **HTML5 & CSS3:** Structure and styling of all pages including dashboards, forms, and tables.
- **Bootstrap 5**: For responsive design and styling of cards, modals, buttons, alerts, and tables.
- **Custom CSS**: Added hover effects, alert message behavior, and button stylings to improve UI.

5. Client-Side Interactivity

JavaScript (vanilla): Used for showing/hiding flash messages, modals, and adding interactivity like redirect buttons.

Google Maps URLs: Used to show "Get Directions" links for each parking lot that open in a new tab.

Admin Dashboard

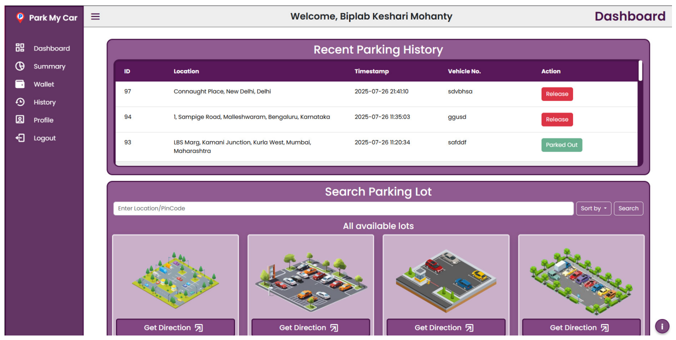


Connaught Place Parking
Location: Connaught Place, New Delhi, Delhi
Available Spots: 4/18

Phoenix MarketCity Parking
Location: LBS Marg, Kamani Junction, Kurla West, Mumbai, Maharashtra
Available Spots: 12/20

Mantri Square Mall Parking
Location: 1, Sampige Road, Malleshwaram, Bengaluru, Karnataka
Available Spots: 7/16

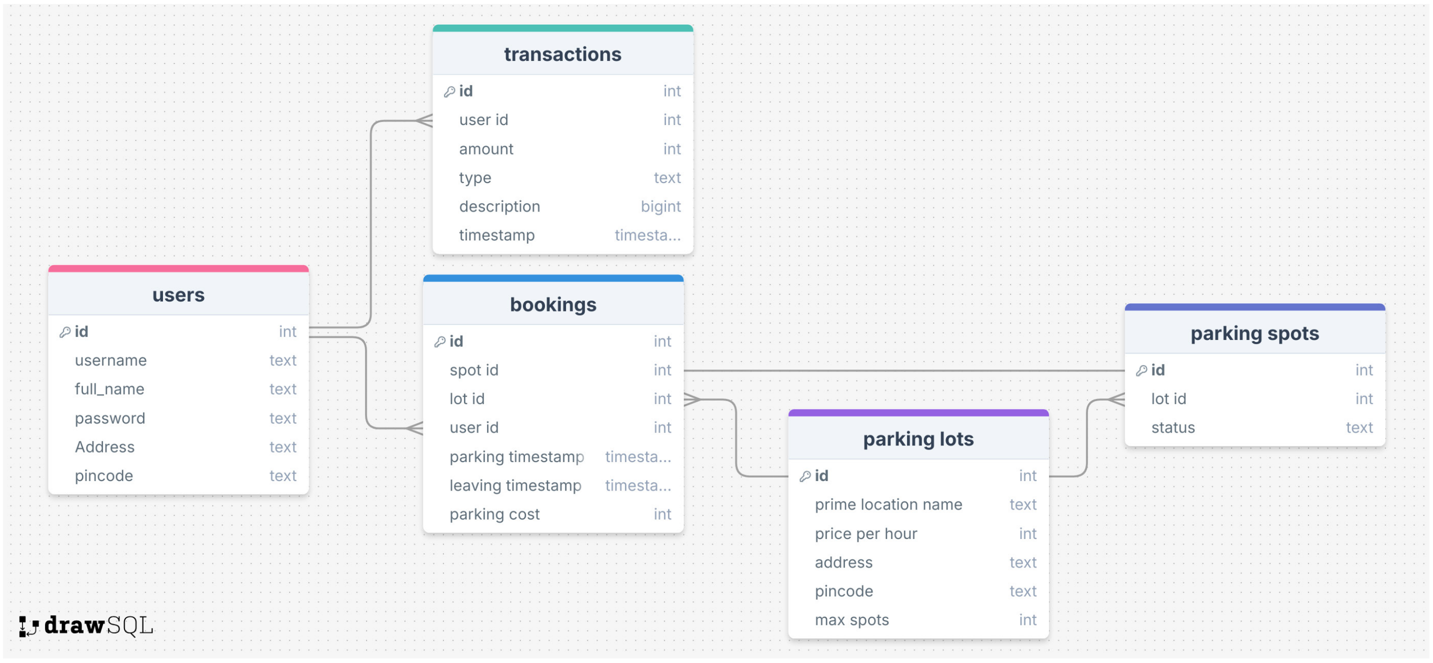
User Dashboard



Recent Parking History

ID	Location	Timestamp	Vehicle No.	Action
97	Connaught Place, New Delhi, Delhi	2025-07-26 21:41:10	sdvbhsa	Release
94	1, Sampige Road, Malleshwaram, Bengaluru, Karnataka	2025-07-26 11:35:03	ggusd	Release
93	LBS Marg, Kamani Junction, Kurla West, Mumbai, Maharashtra	2025-07-26 11:20:34	safddf	Parked Out

Entity-Relationship Diagram



Features Implemented

Core Features:

1. **User Registration and Login System:** Users can securely register and log in to access parking functionalities.
2. **Admin Login System:** Admin credentials are verified from a JSON file to allow only authorized access.
3. **Admin Dashboard**: Admins can add, edit, or delete parking lots, view all users, and manage parking operations.
4. **User Dashboard:** Users can view available parking lots, book and release parking spots, and see current booking status.
5. **Dynamic Parking Spot Allocation:** When a user books, the system automatically allocates the first available parking spot in the selected lot.
6. **Database-Driven Design:** Uses SQLite for persistent storage of users, lots, bookings, transactions, and more.
7. **Session Management:** Ensures login sessions for users and admins are securely maintained and cleared on logout.
8. **Summary Charts and Statistics:**

User Summary Charts

1. **Bookings Per Parking Lot**
2. **Parking Duration Categories**
3. **Daily Spend Trend**

Admin Summary Charts

1. **Income by Parking Lot**
2. **Daily Revenue Trend**
3. **Bookings Per Day**

Extra Features:

1. **Wallet Integration:** Each user has a wallet showing account balance, with automatic deduction on spot release and transaction history.
2. **Sort by Price Feature:** Parking lots can be sorted by price per hour to help users make cost-effective choices.
3. **Slot Grid Representation:** Each lot page includes a color-coded grid: red for occupied, blue for available, and green for user's own slot, giving a clear visual of the lot.
4. **Image Upload and Display:** Admins can upload custom images for parking lots, which are shown on the lot info page for users.
5. **Google Maps Link:** Each lot can have a Google Maps direction link; users can click "Get Directions" to open the location in a new tab.
6. **Password Hashing with Bcrypt:** User passwords are hashed with bcrypt before being stored in the database to ensure security.
7. **Flash Popup Messages:** All actions like login, booking, errors, and updates are followed by clear flash popups to notify users or admins

API Resource Endpoints:

1. api/fetch_first_available_spot<int:lot_id>
Method: GET
Description: Fetches the first available parking spot from the selected lot and assigns it to the user.
Input:
lot_id – ID of the parking lot
Output:
JSON response with spot ID if available, or error message if not

2. api/get_spot_details/<int:spot_id>
Method: GET
Description: Returns details of a particular parking spot by its ID.
Input:
spot_id – Path parameter specifying the ID of the spot
Output:
JSON response with spot status, lot info, and booking details

Video:

https://drive.google.com/file/d/1AiDf8UNUZPsZdJp0wFGnEedjbDxmYMi9/view?usp=sharing