Vehicle Parking App - V1

Project Report

Author

Name: Tamanna Chopra

Roll Number: 23f2004245

Email: 23f2004245@ds.study.iitm.ac.in

About Me

I am a second-year Data Science student at IIT Madras with a strong interest in building impactful tech products at the intersection of full-stack development and AI. Through this Vehicle Parking App project, I addressed a common urban challenge efficiently managing parking spaces. The project allowed me to apply end-to-end development skills, from designing a scalable database schema to implementing role-based dashboards for users and admins.

Description

A multi-user parking management system for 4-wheeler vehicle parking built using Flask and SQLite. The app allows Admins to manage parking lots and spots, while registered Users can reserve and release parking spaces.

Technologies Used

Backend: Flask, Flask-Login, Flask-WTF, SQLAlchemy, Werkzeug

Frontend: Bootstrap 5.3, HTML, CSS, Jinja2

Database: SQLite

DB Schema Design

1. Users

The Users table stores information for all registered users, including both regular users and admins. Each record contains:

- uid: A unique identifier for each user.
- uname: The username used for login (must be unique).
- upass: A hashed password for secure authentication.

- name: The full name of the user.
- created_on: Timestamp indicating when the user account was created.
- isAdmin: A boolean flag that determines whether the user has admin privileges.

Admins are not stored in a separate table. They are identified within the Users table using the isAdmin flag.

2. Parking Lots

The Parking Lots table holds metadata about each parking lot managed by the application. It includes:

- lot_id: A unique identifier for each parking lot.
- lot_title: The name/title of the lot.
- lot_location: The physical address or area name of the lot.
- lot_zip: The pincode or ZIP code for the lot's location.
- rate_per_hr: Hourly parking rate applicable to the lot.
- total_slots: Total number of parking spots available in the lot.

3. Parking Spots

This table represents individual parking spots within each parking lot.

- slot_id: Unique identifier for each parking spot.
- parent_id: Foreign key that links to ParkingLots.lot_id, indicating which lot the spot belongs to.
- slot_status: Indicates the current status of the spot (e.g., Available, Booked, Unavailable).

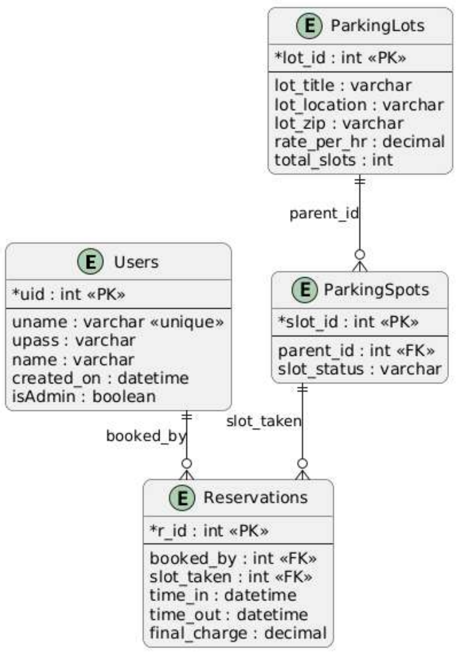
4. Reservations

The Reservations table logs all user bookings and associated details.

- r_id: Unique reservation ID.
- booked_by: Foreign key referencing Users.uid, indicating which user made the booking.
- slot_taken: Foreign key referencing ParkingSpots.slot_id, showing the spot reserved.
- time_in: Timestamp when the booking started.
- time_out: Timestamp when the spot was released or the booking ended.
- final_charge: The calculated charge based on duration and lot pricing.

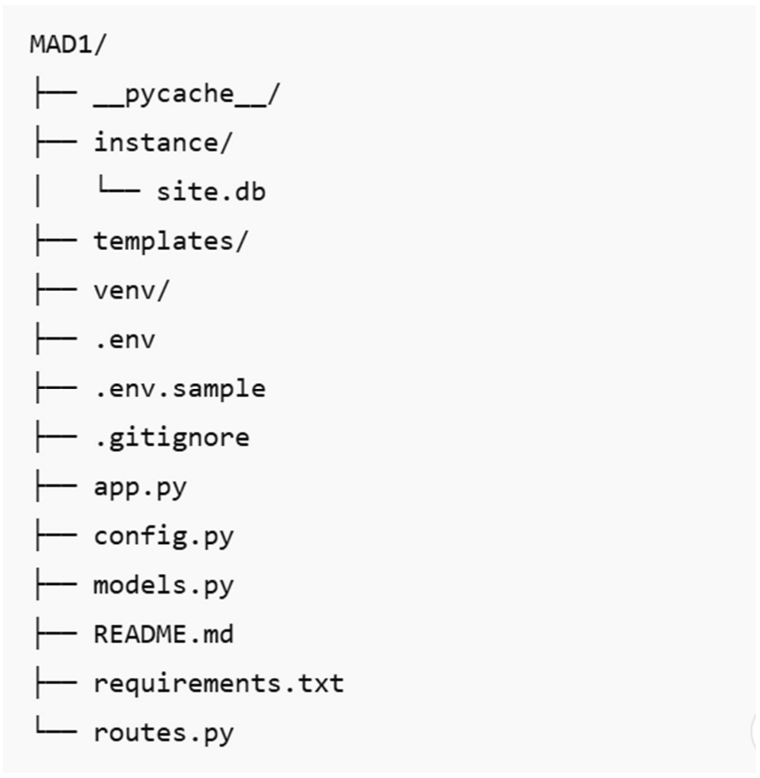
Design Rationale

The design keeps user roles separate, uses foreign keys to link data correctly, and supports automatic deletes for related records. This helps the app work smoothly, keeps data clean, and avoids repeating the same information.



Architecture & Features

Structure



```
MAD1/
├── __pycache__/
├── instance/
│   └── site.db
├── templates/
├── venv/
├── .env
├── .env.sample
├── .gitignore
├── app.py
├── config.py
├── models.py
├── README.md
├── requirements.txt
└── routes.py
```

Features Overview

- User Registration & Login: Secure sign-up and login functionality.
- Real-time Slot Availability: View available parking spots in various parking lots.
- Slot Booking: Book a parking spot instantly based on availability.
- Booking History: View past and current reservations with details.
- Slot Release: Option to release a spot when leaving earlier than expected.
- Cost Calculation: Automatic parking charge computation based on duration.

Key Functionality

Admin Dashboard

- Predefined admin user (no registration required)
- Create and manage parking lots
- Auto-create parking spots based on lot capacity
- View spot status (Occupied/Available)
- Delete parking lots (only if all spots are empty)
- View all registered users
-

User Dashboard

- Register/Login functionality
- View all available parking lots
- Auto-assign first available parking spot
- Release the spot when vehicle leaves
- See timestamps for parking in/out
- View personal parking history summary

Note on Use of AI

AI tools (LLMs) were leveraged during development to better understand Flask’s routing and templating system, integrate Bootstrap for responsive UI design, and assist with front-end implementation and styling decisions.

Video

Video Link: https://www.youtube.com/watch?v=DUU-1te1Q-k