Project Documentation

Author: Ansul Mishra
Roll No.: 24f1001211
Email: 24f1001211@ds.study.iitm.ac.in
Education: 3rd year B.Tech CSE (AI & ML), SRMIST KTR

Project Title: Parking App V1

A web-based solution for managing parking operations. It allows lot owners to manage spaces and users to find, book, and pay for spots. Built with Flask (modular MVC structure) and SQLite, the project emphasizes secure authentication and analytics-ready design.

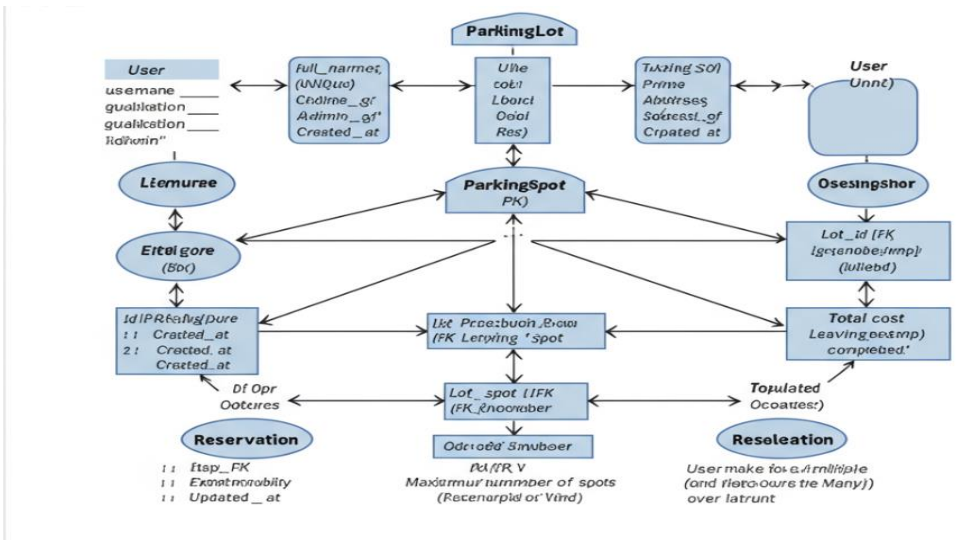
Technologies Used

Backend: Flask, Flask-SQLAlchemy, SQLAlchemy, Flask-Login, Werkzeug
Database: SQLite
Frontend: Bootstrap 5, Font Awesome, Jinja2
Chosen for: scalability, security, simplicity in rapid development

AI/LLM Usage

LLMs supported ~45–50% of the project:
- Frontend (HTML/JS): 28%
- UI styling: 8%
- Extras (API integration, edge cases): 9%
- Backend/DB design: minor support

Database Schema (via Flask-SQLAlchemy)



Entities & Key Fields:
- users: id (PK), username (UQ), email (UQ), password, role, balance, vehicle info

- parking_lots: id (PK), location, price, layout, features
- parking_spots: id (PK), lot_id (FK), coordinates, status (A/O/M)
- reservations: id (PK), user_id (FK), spot_id (FK), time, cost, vehicle, status
- payments: id (PK), user_id (FK), reservation_id (FK), amount, date, method
- transactions: id (PK), user_id (FK), amount, type, description, reference_id (UQ)
- system_stats: date (UQ), revenue, reservations, occupancy

Relationships:
- Users → Reservations, Payments, Transactions
- ParkingLots → ParkingSpots
- ParkingSpots → Reservations (1 active)
- Reservation → Payment

Design Goals: Modular, normalized, secure, supports real-time queries and analytics.

API Design (Flask Routes)

Auth: /login, /register, /logout
User: /user/dashboard, /profile, /wallet, POST /book/<lot_id>, POST /release/<reservation_id>, /api/wallet/balance
Admin: /admin/dashboard, /analytics, CRUD routes for lots, /api/lot/<lot_id>/layout

Architecture & Key Features

Architecture (MVC):
- app.py: Flask setup & controller loading
- models/models.py: ORM schema
- controllers/: auth, user, admin
- templates/: HTML (Jinja2)
- static/: CSS
- utils.py: helper functions

User Features:
- Register/login/logout
- Profile edit & password change
- Wallet view, add/withdraw funds
- Book/release parking spots
- View history

Admin Features:
- Dashboard: user/lots/revenue overview
- CRUD for parking lots & layout
- View analytics: revenue, occupancy

Video Demo

View here:
https://drive.google.com/file/d/1_7MMqy59Qza8ewNzLwFcw3YtWdixuKvI/view?usp=sharing