Author

Deepanshu Singh
23F3000048
23f3000048@ds.study.iitm.ac.in

I am a Diploma-level student in the IIT Madras BS in Data Science program, passionate about backend development and system design. I enjoy building real-world, impactful web apps using Python and modern JavaScript frameworks.

Description

This project aims to build a Smart Vehicle Parking System for both users and admins, where users can book parking spots, and admins can manage lots, spots, and track history.

AI/LLM Used
ChatGPT was used primarily for debugging suggestions, code structuring tips, and UI improvements to some extend also used in backend and charts part.
Which make roughly 20% use of LLMs according to Guideline Documents.

Technologies Used

Flask: Core backend framework

Flask-RESTful: For structuring APIs cleanly

Flask-JWT-Extended: For implementing secure authentication and role-based access

Flask-Caching: To improve performance by caching frequent API

responses **Flask-CORS**: To handle cross-origin requests from Vue.js

frontend **SQLAlchemy**: ORM for DB modeling and querying

Vue.js: Frontend framework for building dynamic user/admin dashboards

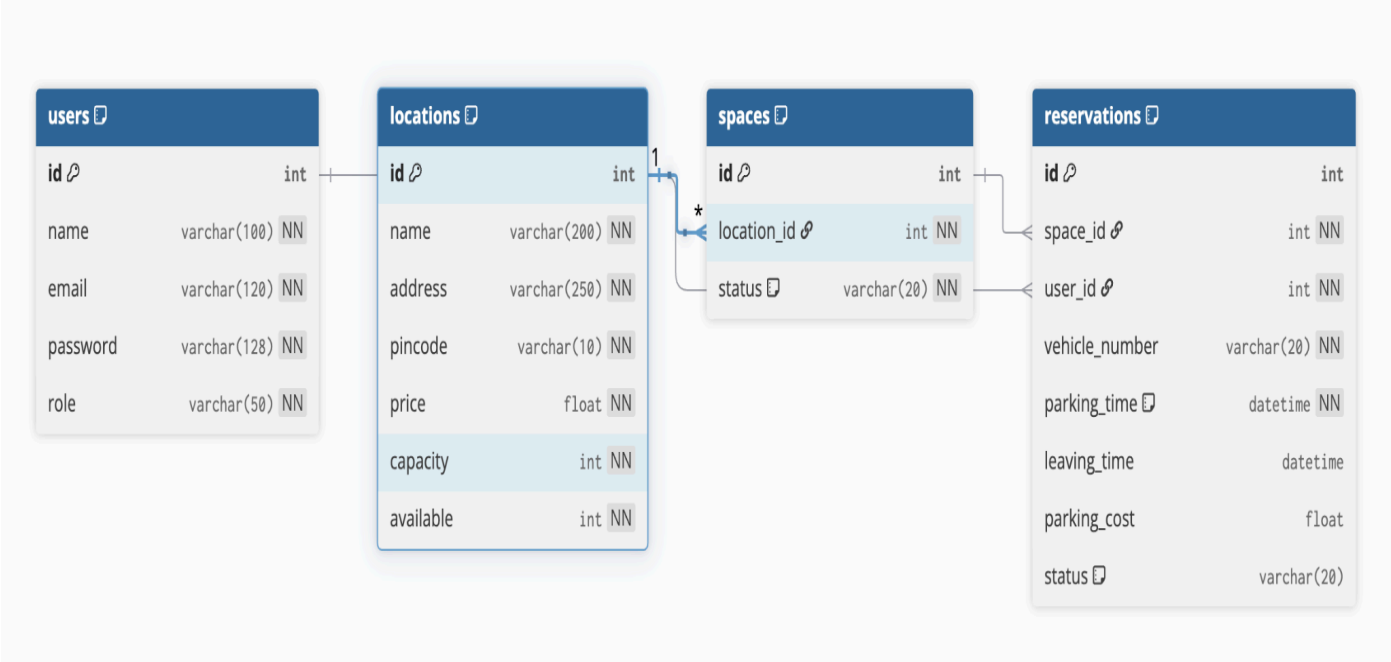
Bootstrap: For responsive and aesthetic UI

Celery + Redis: For background tasks like scheduled reminders

Purpose:

Each of these technologies was selected for clean API design, modular development, secure access, and scalable deployment. Vue + Bootstrap allowed for a lightweight but functional frontend interface.

DB Schema Design



API Design

APIs were created for the following modules:

- **Authentication**: Login, Signup, Role-based access (JWT)
- **User**: Book a spot, view parking history
- **Admin**: Create, update, delete parking lots and spots
- **Shared**: Summary data, charts, available spots
- **Asynchronous Tasks**: For reminders/reporting

All APIs are RESTful and implemented using Flask-RESTful.

Architecture and Features

The project follows a modular architecture with main.py as the entry point. API logic resides in the resources/ folder, models in models/, and helper functions in utils/. The frontend is built with Vue.js in the frontend/ folder, while templates/static files are used for rendering when needed.

Features Implemented

Default features include JWT-based auth, role-based dashboards, lot/spot management by admins, booking and history for users, and dynamic parking availability and cost tracking.

Additional features include admin dashboards with summary charts (Chart.js), API caching for performance (Flask-Caching), and optional background jobs for email reports/reminders via Celery and Redis.

Video

MAD2_Project.mov