Author

BADDILA MOHAN SAI
23f1002534
23f1002534@ds.study.iitm.ac.in
I am curious about Web development and Software Development. This project gave me hands-on experience in building an application.

Description

This Project is a multi-user vehicle parking management system. It allows users to reserve and release parking spots, and administrators can add, edit, and delete parking lots, as well as view data related to parking lots.
AI/LLM Used percentage : 15-20% in assistive coding and debugging. I have used it as a guide during for learning

Technologies used

- Flask: Web framework to handle backend routes and templates
- Flask-SQLAlchemy: ORM to manage models and interact with SQLite database
- SQLite: It is the database which is used to store users, parking lots, parking spots and Reservations
- HTML/CSS: Used for frontend development
- Jinja2: Used for templating
- Chart.js: Used for visualisation

DB Schema Design

1. User
- id: Integer, Primary Key
- username: String(50), Unique, Not Null
- email: String(60), Unique, Not Null
- password: String(100), Not Null

2. ParkingLot
- id: Integer, Primary Key
- prime_location_name: String(100), Not Null
- price: Integer, Not Null
- address: String(200), Not Null
- pincode: String(8), Not Null
- maximum_number_of_spots: Integer, Not Null

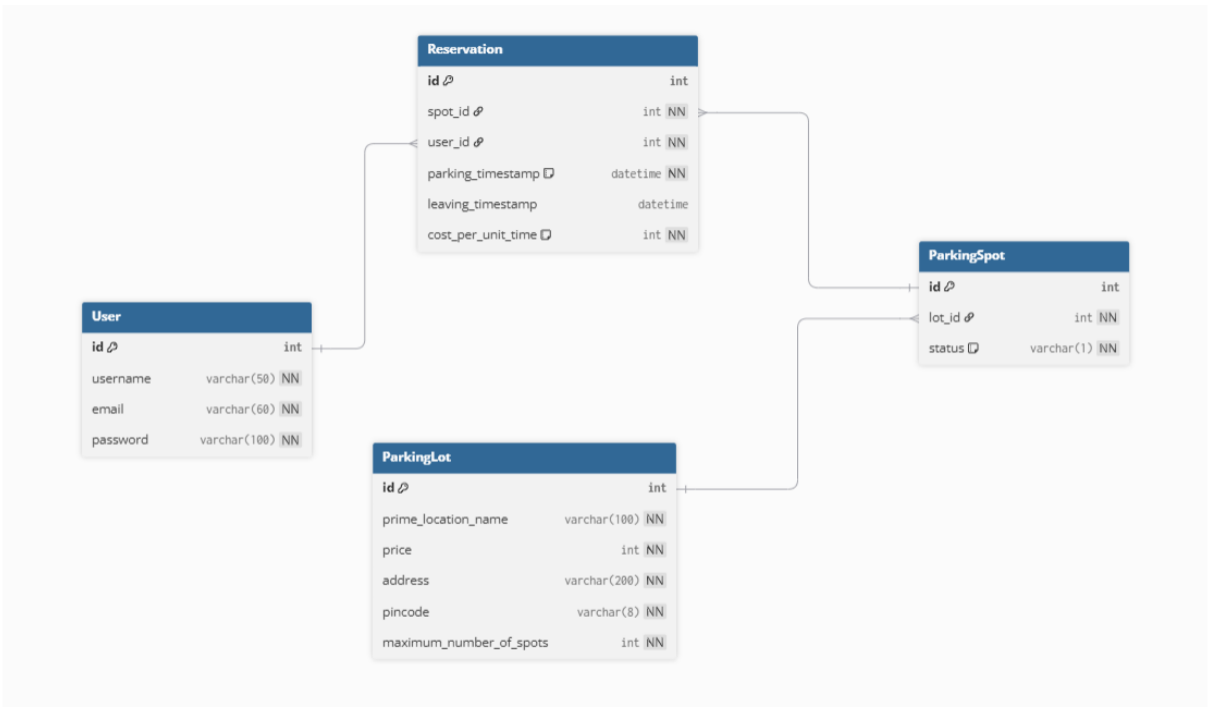
3. ParkingSpot

- id: Integer, Primary Key
- lot_id: Integer, Foreign Key → ParkingLot.id, Not Null
- status: String(1), Not Null, Default 'A'

4. Reservation

- id: Integer, Primary Key
- spot_id: Integer, Foreign Key → ParkingSpot.id, Not Null
- user_id: Integer, Foreign Key → User.id, Not Null
- parking_timestamp: DateTime, Default = datetime.now(timezone.utc)
- leaving_timestamp: DateTime, Nullable (null if ongoing)
- cost_per_unit_time: Integer, Default = 10, Not Null

Below is the ER diagram



API Design

RESTful routes were implemented using Flask.
Routes for authentication: login, registration, logout.
Routes for admin: Add, edit, delete, and view parking lots; view users; view parking status; and view charts.
Routes for user: Reserve spot, release spot, view reservation summary, and view charts.

Architecture and Features

All routes and logic are handled inside the “app.py” file. All the templates for the project reside in “templates/” folder while the “static/” folder contains the CSS files and images in “styles/” and “images/” folders respectively. The data models User, ParkingLot, ParkingSpot, and Reservation are defined in “models.py” . The“vehicle.db” file which contains the data is present in the “instance/” folder.

The project implements all core features required for a Vehicle Parking App. Users can register, log in, and reserve a parking spot automatically from the first available spot. Once done, users can also release the spot, and the app calculates the total time parked and the cost. Users can view all their past reservations. On the admin side, the admin can log in and manage everything: add, edit, view, or delete parking lots, view the list of users, and check the details of occupied spots. The admin can also see chart showing the most used parking lots and total reservations per lot .Only Admin can access admin features and only user can access user features. All features are handled using Flask routes, Jinja templates, and the database is managed using SQLAlchemy.

Video

https://drive.google.com/file/d/1b_tdlro1bACBFgFLA0zgrzojf6-_I7Fv/view?usp=drive_link