Vehicle Parking App - V1

Project Report

Author

Nanda Kumar B
23f3001984
23f3001984@ds.study.iitm.ac.in
About Me:
I am a currently a diploma student who is pursing this course and simultaneously doing the B.E Computer Science & Engineering with interest in ML & Data Science. Also likes to solve the real-life problems and finding solutions to it.

Description

To develop a **Flask-based Vehicle Parking Web Application** that enables users to seamlessly find, book, and manage parking spots in real-time, And also which has admin access who can monitor the parking lots, monitor spot usage, manage user data, and analyze revenue through an interactive dashboard.

Technologies Used

Backend: Flask, Flask-Login, Flask-WTF, SQLAlchemy
Frontend: Bootstrap 5.3, JavaScript, Chart.js
Database: SQLite (lightweight, in-file DB for easy testing)

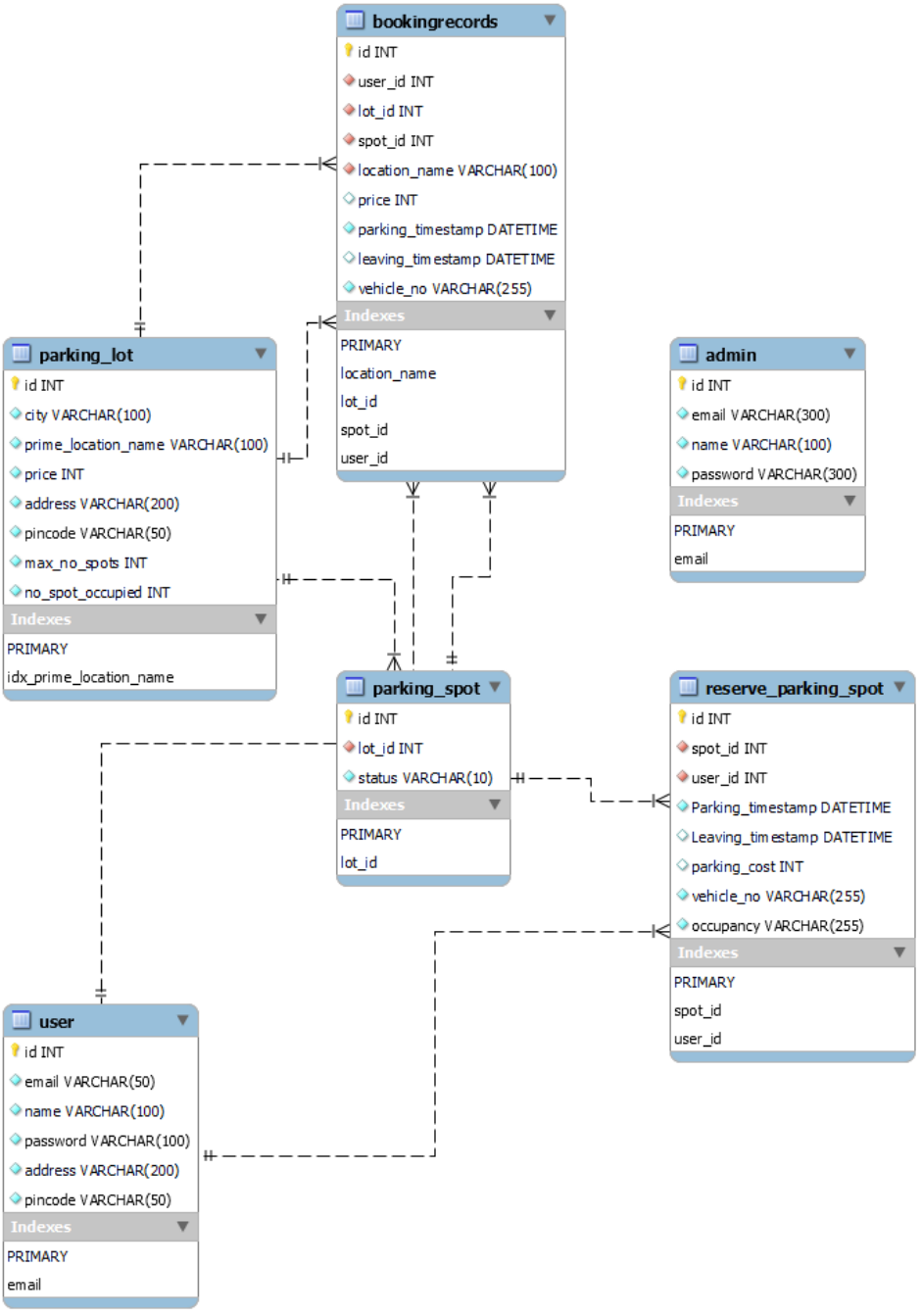
DB Schema Design

- **users:**
  id, email, name, password (which is hashed using the bcrypt library), address, pincode
- **admin:**
  id, email, name, password (which is hashed using the bcrypt library)
- **parking_lots:**
  id, city, prime_location_name, price, address, pincode, max_spots, no_spots_occupied

- **parking_spots:**
  id, lot_id (FK), status
- **reserver_parking_spot:**
  id, spot_id (FK), user_id (FK), parking_timestamp, leaving_timestamp, parking_cost, vehicle_number, occupancy
- **bookingrecords:**
  id, user_id (FK), lot_id (FK), spot_id (FK), location_name (FK), parking_timestamp, leaving_timestamp, price, vehicle_number

---

Design Rationale:
Clear role separation, referential integrity via foreign keys, and cascading deletes ensure smooth data flow and minimal redundancy.



Architecture & Features

Structure:

```
parking_app_23f3001984/
├── app.py
├── controllers/
│   ├── appFunctions.py
│   ├── forms.py
│   └── routes.py
├── models/
│   ├── __init__.py
│   └── model.py
├── static/
│   ├── Style.css
│   ├── Summary.js
│   ├── UserSummary.js
│   └── images/
│       ├── 37313853.png
│       └── Vehicle-parking.jpg
└── templates/
    ├── AddParkingLot.html
    ├── AddingParkingSpot.html
    ├── AdminDashboard.html
    ├── AdminSearch.html
    ├── AdminSummary.html
    ├── EditParkingLot.html
    ├── EditParkingSpot.html
    ├── UserBookingLot.html
    ├── UserSummary.html
    ├── Userdashboard.html
    ├── ViewUsers.html
    ├── base.html
    ├── editprofile.html
    ├── login.html
    ├── register.html
    └── release_spot.html
```

Note on Use of AI

AI tools (LLMs) were used during development for debugging, error resolution, and implementation brainstorming.

Features Overview:

- Auth via sessions and Backend verifications
- Secure sessions, role-based access
- Chart.js analytics for admins and users
- Alerts for various invalid requests, loading states

Key Functionality

User Dashboard:

- View the previously booked slots and option to release it & book slots
- Real-time availability of the available slots in the summary portal status.
- History, usage analytics, and profile prompts

Admin Panel:

- Add/edit/delete lots & spots
- View all users and reservation data
- Spot status: green (free), red (occupied)
- Analytics: revenue by each user, and usage of all the available parking spots.
- Search implementation for users and parking lots.

Video

https://drive.google.com/file/d/1kucAA-xwCi1ZosWIkNNCEnYkBvXvfxHw/view?usp=sharing