***Full Name**: Shreyavarshini Subramanian*
***Roll number**: 23f2005375*
***Student email**: 23f2005375@ds.study.iitm.ac.in*

Description:
-The project is a vehicle parking system where admins manage parking lots and slots, while users can search for nearby parking locations, book and release slots, and view summaries of their parking history and expenses.
-AI percentage used: 8% (CSS/Bootstrap).

Technologies Used:

1. Flask: Used to build the backend application which handles routing, form submission, authentication, and rendering Jinja templates.
2. SQLAlchemy: Used for defining data models and performing all database operations like queries, inserts, and updates.
3. Jinja2: Used to build HTML templates dynamically and render data based on user roles (admin/user).
4. SQLite: A lightweight relational database used to store user data, parking lot data, bookings, and history.

DB Schema Design:

1) User

1. u_id – Integer, Primary Key, Auto Increment
2. email – String, not null, unique
3. password – String, not null
4. name – String, not null
5. address – Text
6. pincode – Integer
7. reservations – Relationship with Reservation

2) Admin

1. id – Integer, Primary Key, Auto Increment
2. email – String, not null, unique
3. password – String, not null
4. name – String, not null

3) ParkingLot

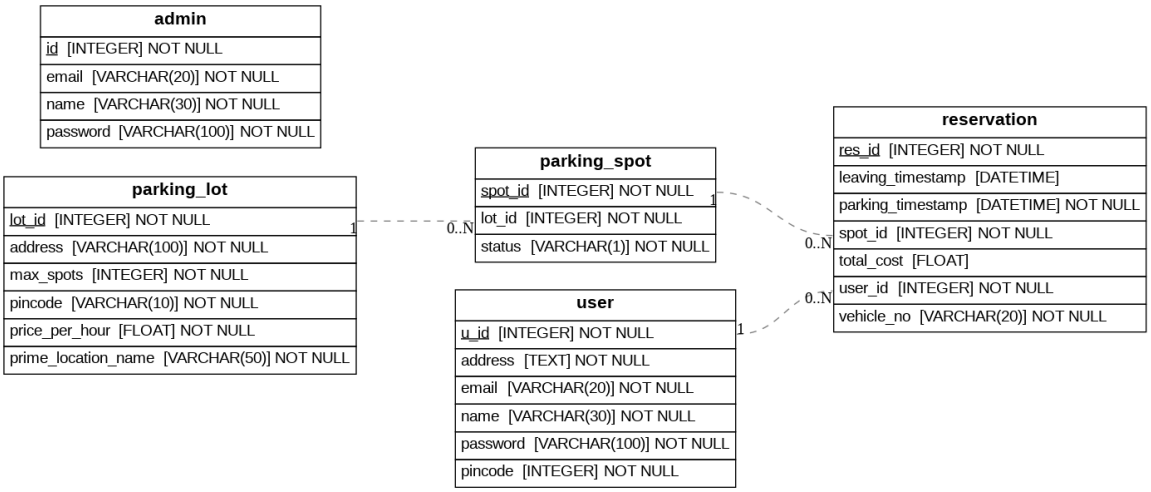
1. lot_id – Integer, Primary Key, Auto Increment
2. prime_location_name – String, not null
3. price_per_hour – Float, not null
4. address – String, not null
5. pincode – String, not null
6. max_spots – Integer, not null
7. spots – Relationship with ParkingSpot

4) ParkingSpot

1. spot_id – Integer, Primary Key, Auto Increment
2. lot_id – Foreign Key referencing ParkingLot
3. status – String ('A' for Available, 'O' for Occupied)
4. reservations – Relationship with Reservation

5) Reservation

1. res_id – Integer, Primary Key, Auto Increment
2. spot_id – Foreign Key referencing ParkingSpot
3. user_id – Foreign Key referencing User
4. parking_timestamp – DateTime
5. leaving_timestamp – DateTime (nullable)
6. total_cost – Float (nullable)
7. vehicle_no – String (nullable)



Each reservation references the user and parking spot used. Parking spots are tied to parking lots, forming a chain of traceability.

Architecture:

- The app.py contains the schema of the database, logic for admin/user verification and user registration, and controls the routes for admin and user views.
- The template folder contains the html files needed for frontend.
- The instance folder contains the database instance.

Features:

Admin

- Can add/edit/delete parking lots and spots.
- Can view summary of each parking lot: total spots, occupancy, and revenue.
- Can monitor real-time slot status via a dashboard.
- Can access analytics like total revenue and usage summary.

User

- Can register/login securely.
- Can search for nearby lots by location.
- Can book and release parking slots, with cost calculated based on parked time.
- Can view a summary of their bookings, total cost spent, and location usage.

Video link:

https://drive.google.com/file/d/13_IKz_-l9XW0n0fvC_CPCO2rDPb3XzIn/view?usp=sharing