Vehicle Parking System - Project Report

Author

Harshit Jain
23F3001028
23f3001028@ds.study.iitm.ac.in
BSc Data Science student currently at Diploma level who also is also pursuing a dual degree in BCA. This project reflects my skills in full-stack development and interest in creating structured solutions.

Description

This project involves developing a web-based vehicle parking system with two types of users — admin and clients. Admins can manage lots and monitor parking activities, while users can book and release parking spots. The goal is to automate slot allocation and ensure a seamless user experience across all operations.

Technologies Used

- **Python + Flask**: Core web framework used for server-side logic and routing.
- **Flask-SQLAlchemy**: ORM (Object-Relational Mapping) for managing models and database communication.
- **SQLite**: Lightweight database used for local development.
- **Bootstrap 5**: For responsive, mobile-friendly and clean UI design.
- **Jinja2**: For dynamic HTML rendering.
- **Flask Blueprints**: For modular routing (admin, auth, client).
- **HTML5 + CSS3 + JS**: Base front-end technologies for form handling and flash behaviour.

These technologies were selected as per the mandatory requirement of the project.

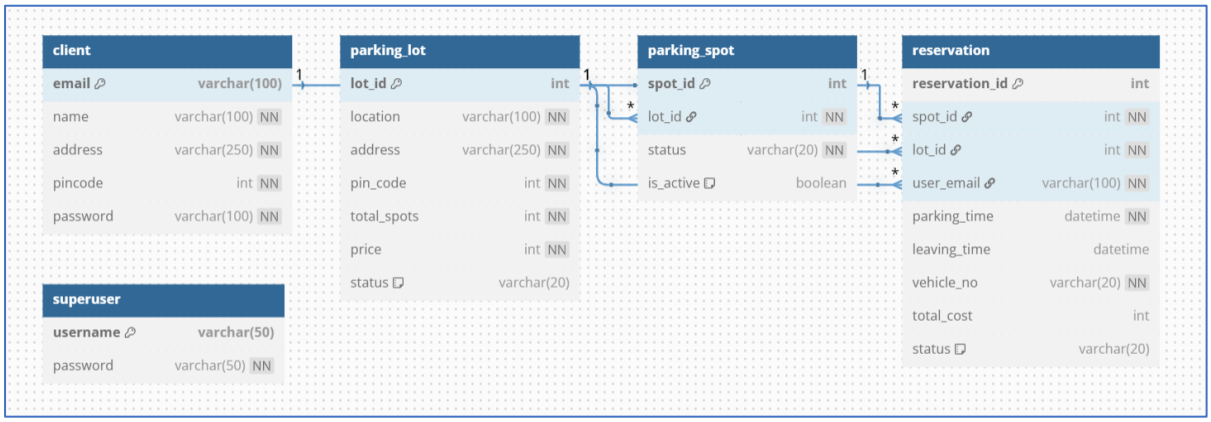
DB Schema Design

The app uses five main tables:

1. **client**:
    - email (PK), name, address, pincode, password
    - Used for user authentication and reservation tracking
2. **superuser**:
    - username (PK), password
    - Used for admin login
3. **parking_lot**:
    - lot_id (PK), location, address, pin_code, price, status
    - Represents a complete parking lot

4. **parking_spot**:
    - spot_id (PK), lot_id (FK), status, is_active
    - Individual spot associated with a lot
5. **reservation**:
    - reservation_id (PK), user_email (FK), lot_id (FK), spot_id (FK), vehicle_no, parking_time, leaving_time, total_cost, status
    - Stores each parking record

The schema is normalized with strong constraints and relationships for efficient data integrity.



Architecture and Features

- **Project Structure**:
    - app/routes/: Contains Flask Blueprints for admin, client, and auth (authorization).
    - app/models/: SQLAlchemy model definitions.
    - app/templates/: HTML templates categorized by role (admin/client/login).
    - app/static/: Images.
- **Implemented Features**:
    - **Admin**:
        - Create, view, and delete parking lots (only if empty)
        - Auto-generate parking spots
        - View all registered users
        - Summary dashboard
        - Searching across all models
    - **Client**:
        - View/update profile
        - View lots and book parking
        - Auto slot allocation

- View, release, and track past reservations
- Individual summary dashboard (analytics for each user)

- **UI/UX**:
  - Fully responsive layout using Bootstrap and CSS
  - Flash messages with auto-dismiss
  - Confirm prompts for delete and release actions
  - Elegant English-themed color scheme

Video

Link: https://drive.google.com/file/d/1gAcWDN1Z4VZHdt4fXoJ9C3tfsJQW0x0d/view?usp=sharing

---

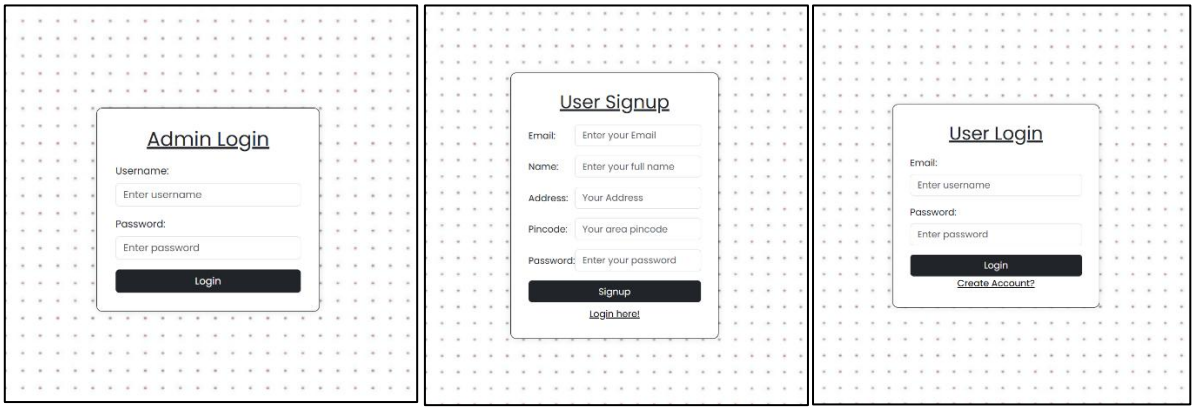
AI Usage Declaration

I have used around 20-25% AI-generated content was used during development for complex problem solving, understanding and learning few concepts, and debugging — but not misused in any manner, just for gaining knowledge purpose and limited to guidance and brainstorming. All code and logic were implemented by me from scratch in a hands-on manner, ensuring originality and full understanding.

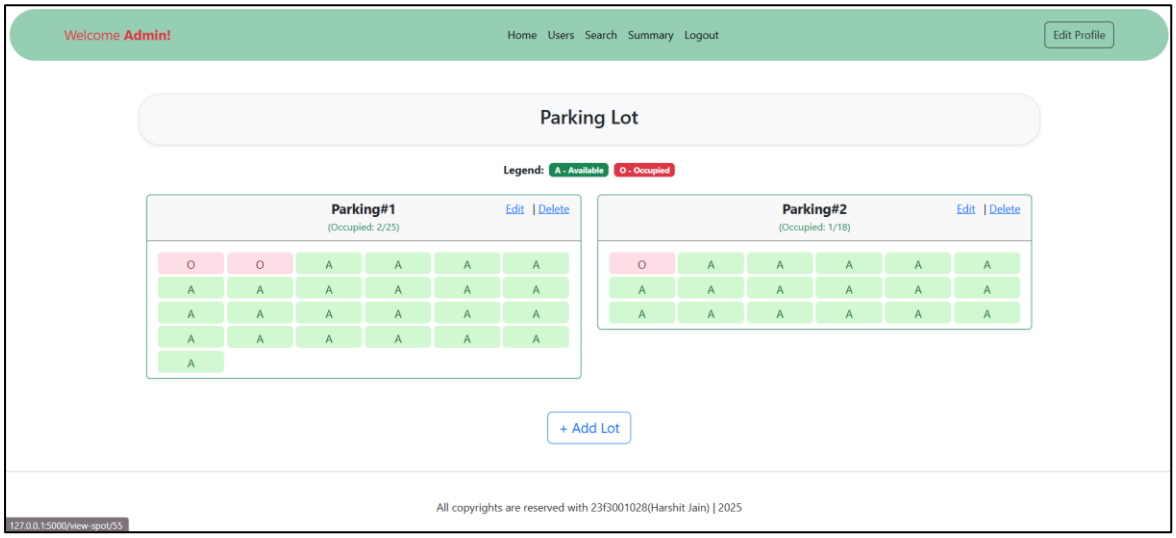
THANKING YOU

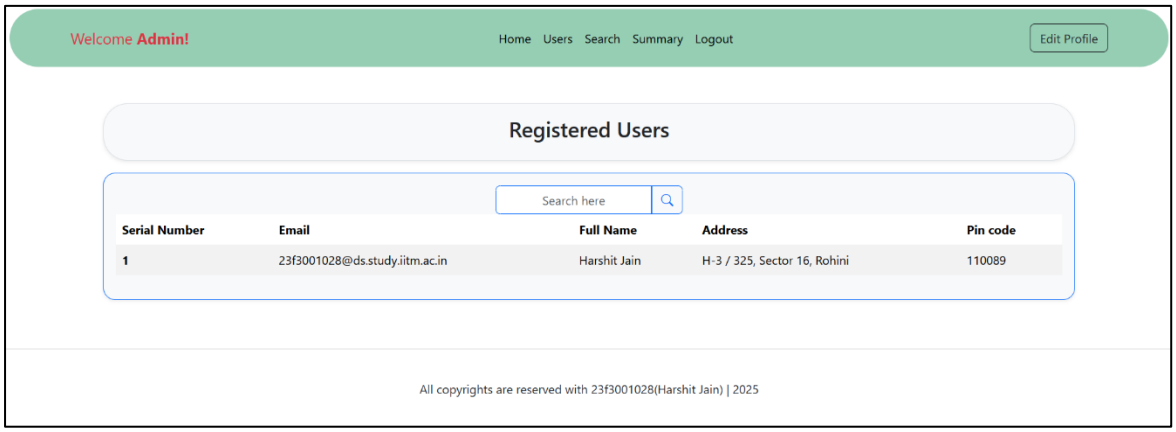
Screen Shots of the project

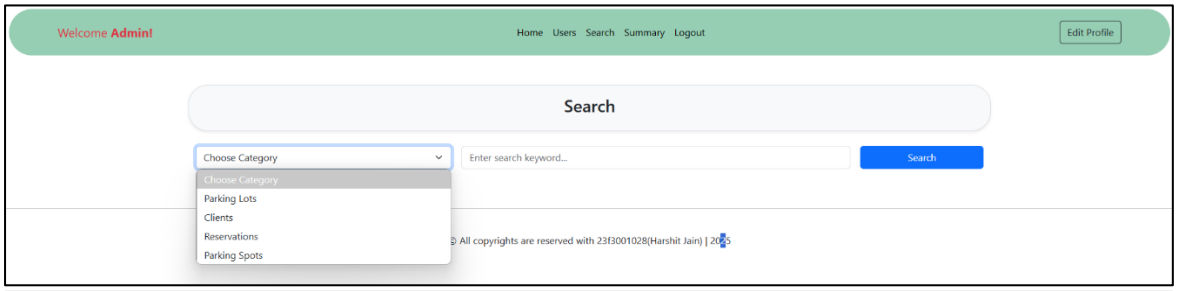
LOGIN PAGES

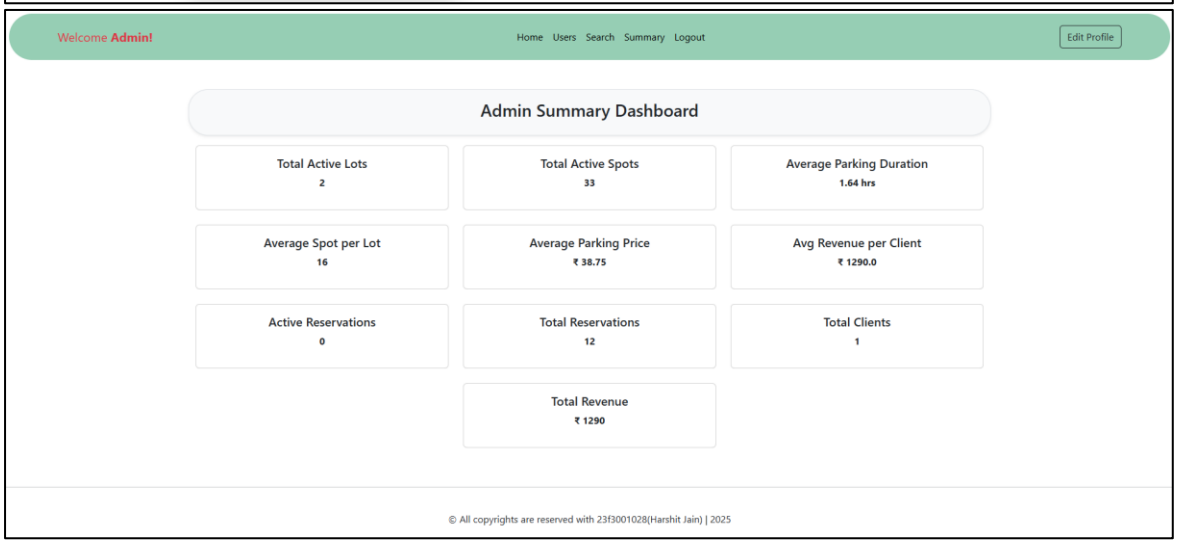


ADMIN DASHBOARD









IIT Madras
BSc Degree

USER DASHBOARD