Author

Yuvaraj Karunakaran
22f2001731
22f2001731@ds.study.iitm.ac.in
Passionate programmer, consistent worker, and someone eager to take on new challenges. An avid reader, football fanatic, and general sports lover.

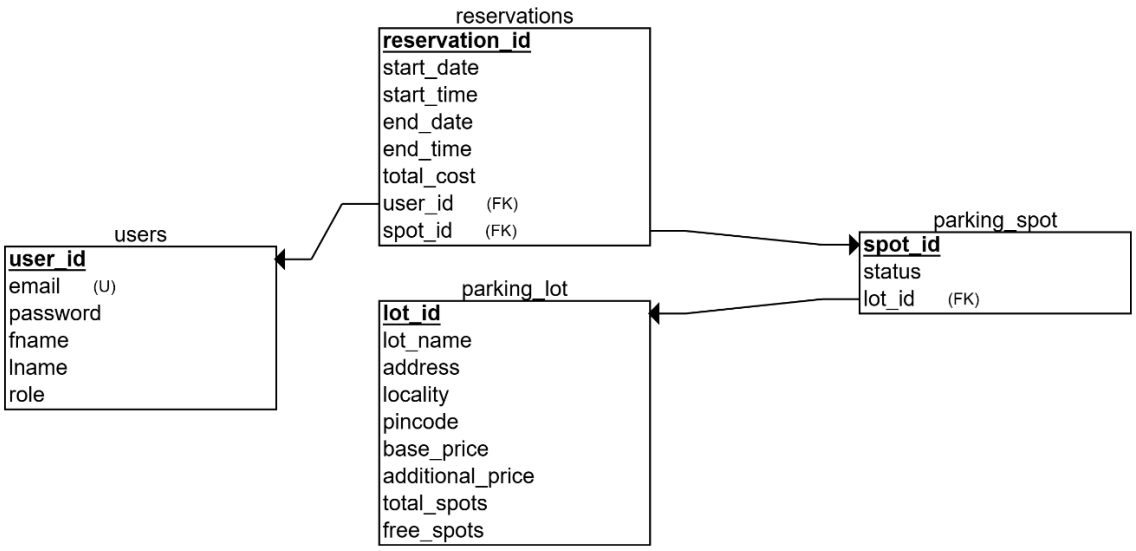
Description

The Vehicle Parking Application, as the name suggests, involves a comprehensive parking system specifically for four-wheelers. It has an admin user who manages parking lots, parking spots, and user reservations, and regular users who reserve available parking spots in any desired spot for their vehicle. Admins should have 100% access to all data that is dealt with here, whereas users will only have access to their own specific data.

Technologies Used

Technology	Purpose
Python Flask	Backend Logic, Mediator between frontend and DB
SQLite	Database
HTML	Frontend Content
CSS	Frontend Design
Jinja2	Template for Flask Responses
JavaScript	Dynamic Frontend
ChartJS	Summary Charts

DB Schema Design



users	Details of user, role is either ‘admin’ or ‘user’
parking_lot	Details of parking lot, including location, pricing, spot availability
parking_spot	Details of individual spot, which simple contains status: ‘A’ for available and ‘O’ for occupied, and reference to lot_id to indicate which parking_lot it belongs to
reservation	Stores details of reservation timings, total cost, and the user who made the reservation for the given spot

‘total_spots’ and ‘free_spots’ in parking_lot are maybe unnecessary attributes, as this information can be derived from the parking_spot table itself. However, for ease of access and potential efficiency gains, these additional redundant attributes are added to parking_lot.

Architecture

models – contains DB definition for Schema

1. user.py
2. parking_lot.py
3. parking_spot.py
4. reservation.py

controllers – backend logic for each component of the application

1. auth.py – registration, login, forgot password, logout
2. auth_utils.py – helper functions for auth
3. admin_dashboard.py – create, view, update, delete parking lots; view user details; view reservations
4. admin_utils.py – helper functions for admin_dashboard
5. user_dashboard.py – view parking lots; reserve, release parking spots, view current, all reservations
6. user_utils.py – helper functions for user_dashboard
7. user_admin_utils.py – helper functions for both user and admin dashboard

static/js – all JS files

1. filter.js – to filter and display parking lots by locality
2. side_panel.js – to toggle side panel
3. update.js – to display update input for parking_lot based on chosen update parameter
4. graph.js – summary chart (bar graph) for admin and user

static/styles – all CSS files

1. auth.css – for register, login, logout
2. home.css – for home page
3. style.css – general styles
4. user.css – for user and admin dashboard

static/json – all JSON files

1. lot_data.json – contains lot_name, locality, total and free spots of all parking lots, used in filter.js
2. reservation_data.json – contains reservation counts, grouped by locality

templates – all HTML files

1. admin/admin_dashboard.html – for admin dashboard
2. auth/home.html – for home page
3. auth/register.html – for register page
4. auth/login.html – for login page
5. auth/forgot_password.html – for forgot password page
6. auth/user_dashboard.html – for user dashboard

app.py – DB relations are initialised, and app is run here

Features

It is assumed that the organization which has developed and is managing this parking application operates only within the city of Chennai. As such, only parking lots in Chennai are created and managed.

General

1. Register,
2. Login
3. Reset Password - simply type user ID and email to get new password

Password is stored in DB only after hashing. Ensures better security.

Admin

1. Add a new Parking Lot with all necessary details.
2. View Parking Lot details – of any selected parking lot.
3. Update Parking Lot details – any specific detail can be updated. For updating total spots,
    a. if new value = 0
    b. if new value < no. of occupied spots presently

the update will not take place. Such edge conditions are gracefully handled.

4. Delete Parking Lot – only those lots which don't presently hold any reservations can be deleted. Other parking lots with current ongoing reservations won't even be shown in the delete section.
5. Tabular view of user details such as name, email, and current reservation details, which includes all the details specified in the schema.
6. Tabular view of entire reservation history of all parking lots.
7. Summary graph of Localities and their respective reservation counts, showcasing the importance of localities and which localities must be prioritised for introducing more parking lots.

User

1. View Parking Lot details, filter by locality.
2. Reserve a free spot in any parking lot.
3. View currently reserved spot details.
4. Release any currently reserved parking spot, total cost of parking gets computed.
5. View complete reservation history.
6. Summary graph of the most frequent parking lots used, grouped by locality.

Demo Video Link

22f2001731_MAD_1_Demo