Author

NANDURI SAI BHASKAR

23f3002374

23f3002374@ds.study.iitm.ac.in

A B. Tech Computer Science Engineering student at Gayatri Vidya Parishad College of Engineering (Autonomous), alongside I'm doing IIT Madras BS Degree and currently pursuing Diploma.

Description

To develop a web-based 4-Wheeler parking management system. It supports both admin and user logins, and allows users to reserve, view, and release parking spots. Admin can add lots and view overall summaries with plots of bookings and revenue, while users can see their parking amount spent. The pricing is calculated per hour.

Technologies used

- Flask: Web Application framework, used for running application
- Flask-SQLAlchemy: It's ORM for database, where we can connect to database and retrieve information by querying
- Jinja2: For templating, provides more flexibility to the html document.
- SQLite: Lightweight database.
- Matplotlib: For plotting different graphs.
- OS, io: For environment and file operations.

DB Schema Design

- **User: For storing the user details in unique way**
    - username (String, Primary Key)
    - fullname (String, Not Null)
    - email (String, Unique, Not Null)
    - password (String, Not Null)
    - address (String, Not Null)
    - pincode (String, Not Null)
- **ParkingLot: For storing where the lot is located, and also for identifying each lot individually**
    - lot_id (Integer, Primary Key, Autoincrement)
    - primelocation (String, Not Null)
    - price (String, Not Null)
    - address (String, Not Null)
    - pincode (String, Not Null)
    - max_spots (Integer, Not Null)
- **ParkingSpot: For allocating each spot in a lot as an individual identity and to check whether the spot is Available/Occupied**
    - id (Integer, Primary Key, Autoincrement)
    - lot_id (Foreign Key → ParkingLot)
    - status (Boolean, Not Null, Default = False)

- **Booking: To handle bookings of the user and for calculation of final fare/prize based in start time & end time of parking**
    - id (Integer, Primary Key)
    - username (Foreign Key → User)
    - lot_id (Foreign Key → ParkingLot)
    - spot_id (Foreign Key → ParkingSpot)
    - starttime (DateTime, Default = current UTC time)
    - endtime (DateTime, Nullable)
    - vehicle_number (String, Not Null)
    - prize (Float, Nullable)

Architecture and Features

- Architecture details:
    1. The 'app.py' is the entry point where the flask app running and calling controllers is possible
    2. Templates folder is used to serve html files it contains files like login.html, signup.html, admin.css, user.css ……
    3. Controllers folder is used to maintain logic and give functionality for a specific option opted it contains files like login.py, add_lots.py, edit_lot.py ….
    4. Static folder is used to create CSS files for styling our current html files
    5. Models folder is used to store our database schemas it has files like user.py, parking_lot.py, parking_spot.py & book.py and Instance folder is automatically created when successful execution of app.py where our database file is stored.
- Features:
    1. After running 'app.py' first of all the all database schemas are created and loaded.
    2. The flask app opens login page if the user type is 'admin' he can directly login to the portal so he'll navigate to admin dashboard, if the user is not an admin, he'll login with his account if he doesn't have an account, he can navigate to create account page and fill the signup form.
    3. Here all form validations are processed and also in login & signup page all the information validation like valid/invalid credentials are displayed.
    4. If Admin he can Add lots and number of slots in a lot and he can edit the lot and able to delete the lot if none of the users are occupied the spots in the particular lot
    5. Admin can see the registered users details, summaries.
    6. The common feature for admin and users is edit profile where he can edit profile details like address, pincode, name etc.
    7. Now coming to user, he can see the available parking lots there is a search functionality added so he can search for parking lot based on pincode or location
    8. After fetching the lot he'll able to reserve a spot in a lot and spot is automatically allocated by the server in first-come-first-serve basis.
    9. Then he can see the details of allocated spot, location, booking date and time etc., after sometime if user wants to vacate the parking spot he must click on release so that the final fare/prize is displayed and then user confirm the release and pay amount displayed to admin.
    10. Finally, he can see the insights in summary section and he can logout from the portal Similarly, after successfully performing operations admin can logout.

Video: https://drive.google.com/file/d/1QrjZr8_dqkzAJFpuyKZzhSN06trPpQ21/view?usp=sharing