Vehicle Parking App - Project Overview and Guide

project name : WhereMyCar

This project is a web-based Vehicle Parking App designed to simplify the process of finding and booking parking spots. It provides a user-friendly platform for drivers to locate available spaces and for administrators to manage the parking lot efficiently. The system is built with Python and Flask, SQL alchemy.

About :

- **Name:** RITURAJ
- **Student ID / Roll Number:** 23f2004390
- Email ID: 23f2004390@ds.study.iitm.ac.in
- **Course:** MAD 1 projects (vehicle parking app)

Intro:

Video link :

https://drive.google.com/file/d/1H1DbyURpYwboSBKUN4sO499omZuvefg1/view?usp=sharing

Report doc :

https://docs.google.com/document/d/1PUYWtft_0fvC1K_1mHXMo3WjYaeEpmWs5McXNjpGw-o/edit?usp=sharing

Key Features

- **Two Main part:** The application provides separate views for users and administrators.
- **User Features:**
  - **Account Management:** Easy sign-up and login functionality.
  - **Parking Search:** View the list of available parking spots and book it.
  - **Booking System:** Book an available spot for a specific time.
  - **History:** View past and active parking spot bookings.
  - **Profile Management:** Update personal information and separate section for password.
- **Administrator Features:**

- **Secure Login:** using session, secure login for the management dashboard both for user and admin.
- **Lot Overview:** See the status of all parking spots at a glance (occupied vs free).
- **User Management:** View and manage all registered user accounts.
- **Booking Tracking:** Monitor all active and past bookings.
- **Manual Updates:** Manually change the status of any parking spot.

The Technology Behind the App

- **Backend (The Engine):**
  - Built with **Python** and the **Flask** framework. (Core functionality)
  - Handles all server-side logic, including user management, booking processing, and database communication.
- **Frontend (What You See):**
  - **HTML:** Provides the basic structure of the web pages.
  - **CSS:** Used for all styling, including colors, fonts, and layout.
  - **JavaScript:** Makes the pages interactive and dynamic.
- **Database:**
  - Uses **SQLite**.
  - Stores all user information, parking spot data, and booking records.

How to Get the App Running (RUN main.py)

- **Prerequisites:**
  - python
  - flask
  - jinja2
  - flask-SQL alchemy

    Start the Application:
    - In the same terminal window, run this command:
      python main.py

  - **Use the App in Your Browser:**
    - The terminal will show the server is running, usually at http://127.0.0.1:5000/.

      Or localhost:5000 (port: 5000).
    - Open this address in your web browser to see the application's landing page.

Database Schema

The database consists of four main tables to manage the application's data, including the concept of a Parking Lot.

- **User Table:** Stores information about registered users.
  - id: Unique ID for each user (Primary Key).
  - username: The user's unique name for logging in.
  - email: The user's unique email address.
  - password: The user's encrypted password.
  - is_admin: A flag to check if the user is an administrator (True/False).
- **Parking Lot Table:** Stores information about each parking facility.
  - id: Unique ID for each parking lot (Primary Key).
  - prime_location_name: short name or nickname
  - name: The name of the parking lot (e.g., "Main Street Garage").
  - address: The address or description of the lot's location.
- **Parking Spot Table:** Stores information about each individual spot within a lot.
  - id: Unique ID for each parking spot (Primary Key).
  - lot_id: Links to the Parking Lot this spot belongs to (Foreign Key).
  - spot_number: The specific number or name of the spot (e.g., "A1").
  - is_available: A flag to check if the spot is currently available (True/False).
- **Booking Table:** Links users to the parking spots they have booked.
  - id: Unique ID for each booking (Primary Key).
  - user_id: The ID of the user who made the booking (links to the User table).
  - parking_spot_id: The ID of the parking spot that was booked (links to the Parking Spot table).
  - start_time: The date and time when the booking begins.
  - end_time: The date and time when the booking ends.
  - is_active: A flag to check if the booking is currently active (True/False)

File Structure

The project is organized into several key folders and files:

```
/
|-- main.py              # Main Flask application file
|-- instance/
|   |-- site.db          # SQLite database file
|-- static/
|   |-- css/             # Stylesheets for the application
|   |   |-- admin.css
|   |   |-- landing.css
|   |   `-- user.css
```

```
|  |-- js/              # JavaScript files for interactivity
|  |  |-- dashboard.js
|  |  |-- landing.js
|  |  |-- user.js
|  |  `-- user_search.js
|  `-- images/          # Image assets used in the app
|     |-- aviable_spot.jpg
|     |-- bg_landing_page.jpg
|     |-- landing-page.png
|     |-- parking_ui.png
|     `-- phone_parking.jpg
|-- templates/          # HTML templates for different pages
|  |-- admin.html       # Admin dashboard page
|  |-- landing.html     # Main landing page
|  `-- user.html        # User dashboard page
|-- .python-version     # Specifies the Python version
`-- README.md           # Project documentation(full)
```

Quick Start

1. **Install Dependencies:** Open a terminal in the project folder and run:
   pip install Flask Flask-SQLAlchemy Flask-Login

2. **Run the App:** In the same terminal, run:
   python main.py

3. **Open in Browser:** Go to http://127.0.0.1:5000/ in your web browser.