## Author

Name: - Suraj Kumar

Roll no: - 24dp1000038

Email: - 24dp1000038@ds.study.iitm.ac.in

## Description

**WheelSpot** is multi-user web application built to manage 4-wheeler parking lots efficiently. Designed for both admin and general users, this system allows real-time parking lot management, automated spot reservations. It features separate admin and user roles, with admin managing parking infrastructure and users reserving spots. The system includes scheduled jobs for reminders/reports.

## Technologies used

1. **Flask** – Backend API development.
2. **SQLite** – Lightweight relational database for storing users, parkingLots, bookings.
3. **Vue.js** – Frontend framework (via CDN)
4. **Bootstrap** – Enhances UI/UX with responsive and modern design.
5. **Redis** – Caching and Celery task queue backend
6. **Celery** – Asynchronous task processing
7. **HTML, CSS & JS** – Structures and styles the frontend of the application.

## Architecture and Features

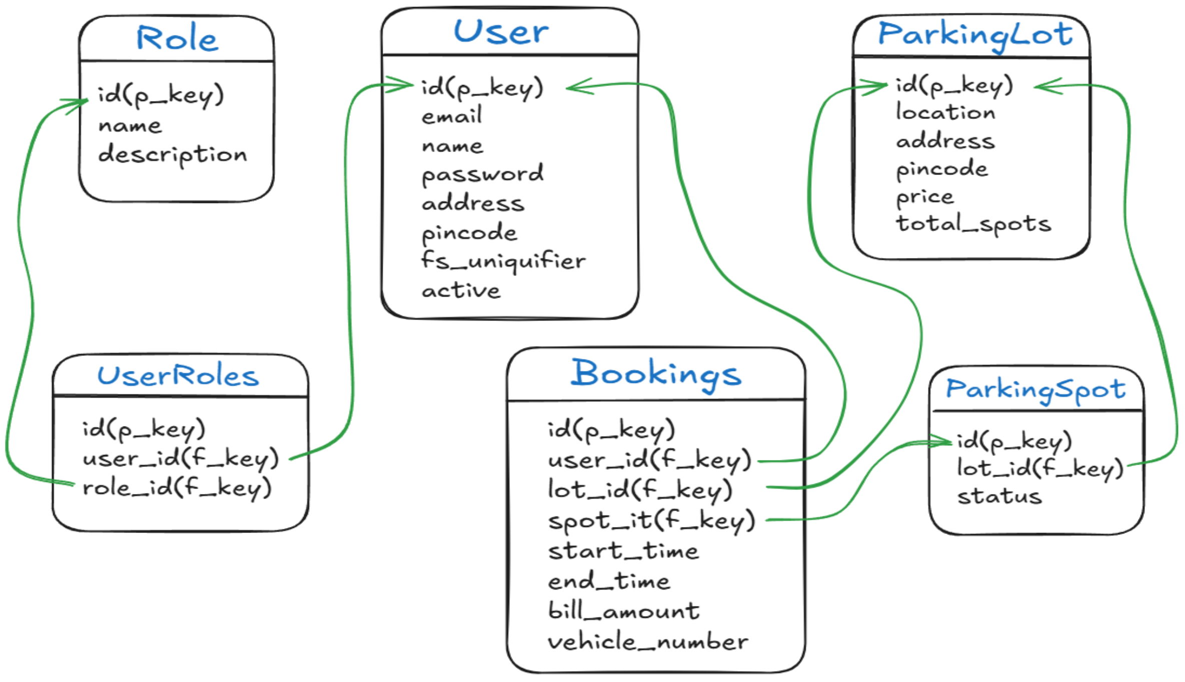
### *Project Organization:*

- app.py : The main entry point for the Flask application, initializing routes, configurations, and database connections.
- backend/model: DB models and relationships
- backend/routes: API endpoints and business logic
- templates/: Base (CDN approach)
- static/components: Vue.js components and assets
- gitignore – Specifies files to be excluded from version control.
- requirements.txt – Lists required dependencies for setting up the project environment.
- README.md – Provides documentation for installation, usage.

### *Implemented Features:*

- Role-based authentication system
- Admin parking lot management
- User spot reservation flow
- Daily reminder , Monthly PDF reports & CSV export functionality

## ER Diagram of Database



## API Design

- POST /api/login - login for both user and admin
- POST /api/register - register for users
- GET /api/admin/lots - List all parking lots
- POST /api/admin/lots - Create new parking lot
- GET /api/admin/lots/int:lot_id - Get parking lot details
- POST /api/admin/lots/int:lot_id - Update parking lot
- DELETE /api/admin/lots/int:lot_id - Delete parking lot
- GET /api/admin/spots - List all parking spots
- GET /api/admin/spots/int:spot_id - Get/Delete spot (DELETE if unoccupied)
- GET /api/admin/users - List all users
- GET /api/admin/search - Search users/lots/spots
- GET /api/admin/summary - Get admin dashboard summary
- GET /api/user/parking/lot/view - View available parking lots
- POST /api/user/book/spot/<lotId> - Book the spot
- GET /api/user/spot/release/preview/<spot_id> - release spot preview
- POST /api/user/spot/release/<spot_id> - release spot
- GET /api/user/summary - user summary

## Video Demo:

https://drive.google.com/file/d/11UOYzsJgQg_nEliBr8r6tpPsBRe1Ru-q/view?usp=drivesdk