Modern Application Development 2 – Project Report

# Vehicle Parking App

Submitted By:

Name: Aditya Goyal
Roll number: 23f1002424
E-mail: 23f1002424@ds.study.iitm.ac.in

IITM Online BS Degree Program

Indian Institute of Technology, Madras, Chennai

Tamil Nadu, India, 600036

## 1. Project Description

The Vehicle Parking App is a full-stack web application designed to allow users to search, book, and manage parking spots while giving administrators control over slot management and data insights.

AI/LLM Used:
Used ~6% for boilerplate code suggestions (e.g., Flask-JWT setup, Vue component structure) and 10% for error fixes and naming suggestions. No generative code was used for core logic.

## 2. Technologies Used

.
**Backend:** Flask, Flask-Cors, JWT-Extended, SQLAlchemy, Migrate, Login
**Frontend:** Vue.js, Bootstrap, Chart.js
**Database:** SQLite
**Other Tools:**

- Celery + Redis for background jobs
- Axios for frontend-backend communication

**Why These?**
Fast prototyping (Flask), reactive UI (Vue), minimal overhead (SQLite), and support for background tasks (Celery) made them ideal.

## 3. Database Design

.
**Tables:**

- users(id, username, email, password_hash)
- parking_lots(id, name, location)
- parking_spots(id, lot_id, spot_number, status)
- reservations(id, user_id, spot_id, start_time, end_time, status)

**Relations:**

- One user → Many reservations
- One lot → Many spots
- One spot → One active reservation

This normalized schema enables scalability, analytics, and clean admin control.

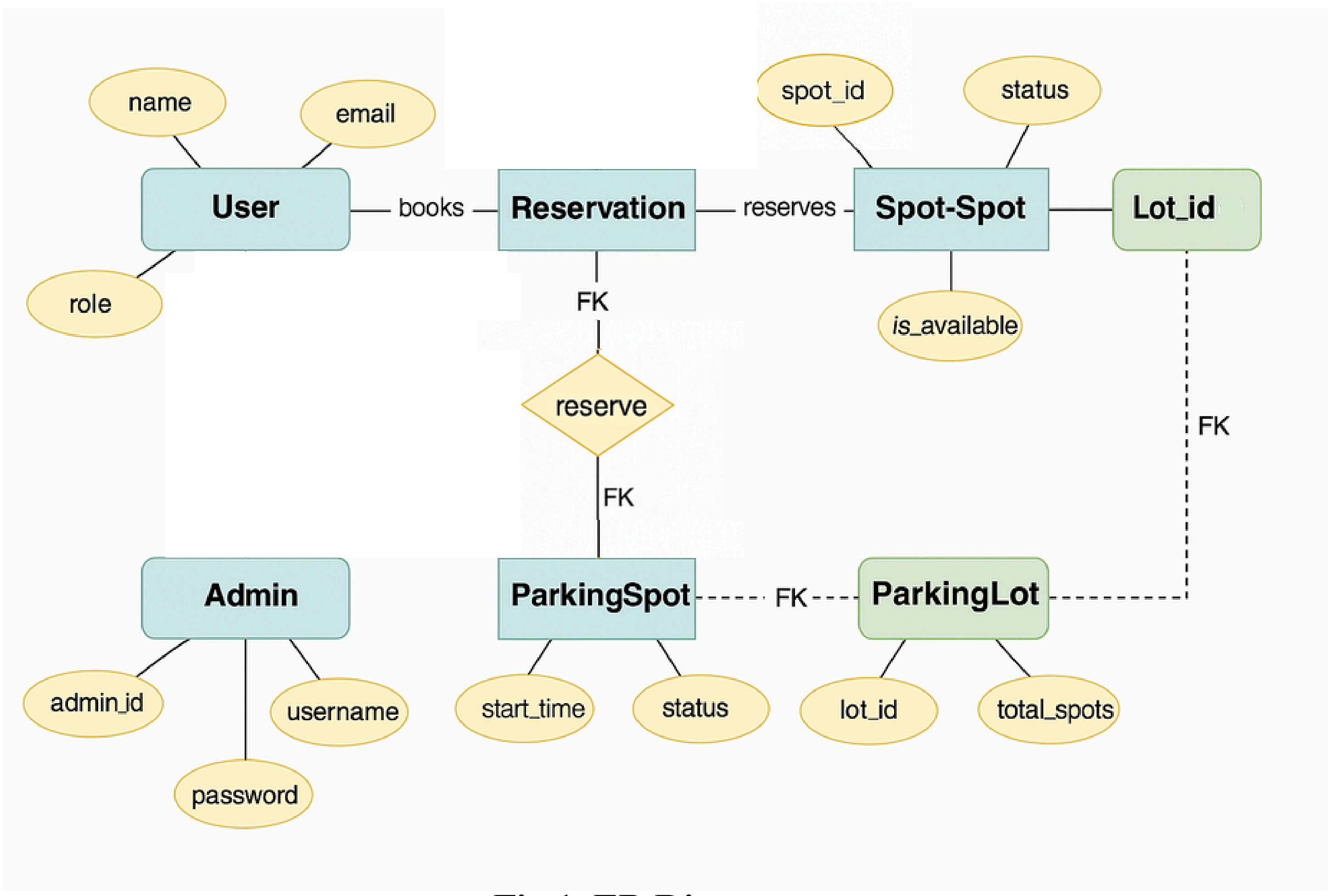

**Fig.1: ER Diagram**

## 4. API Overview

**Auth APIs:**

- POST /register, POST /login

**User APIs:**

- POST /reserve, GET /reservations, DELETE /cancel/<id>

**Admin APIs:**

- GET/POST /lots, POST /spots, PATCH /spots/<id>/status

**Analytics APIs:**

- /stats/occupancy, /stats/daily

APIs follow standard REST structure with JWT-based authentication.

## 5. Features

**Core Features:**

- Login/Signup (JWT)
- Lot/Spot management
- Real-time booking
- Reservation tracking
- Admin panel

**Add-ons:**

- Reservation analytics (Chart.js)
- CSV export
- Celery tasks for expired reservations

## 6. Architecture

- **Backend:** Modular Flask structure with routes, services, models, and Celery config
- **Frontend:** Vue components with Bootstrap UI and Axios for API
- **Minimal Jinja**: Vue handles almost all UI rendering
- **Database:** SQLite

## 7. Video Demo

https://drive.google.com/drive/folders/1whJGJf34SKXWyoSX1YIqLme0-dvyEEf0?usp=sharing