Vehicle Parking App – Project Report

Modern Application Development I – May 2025

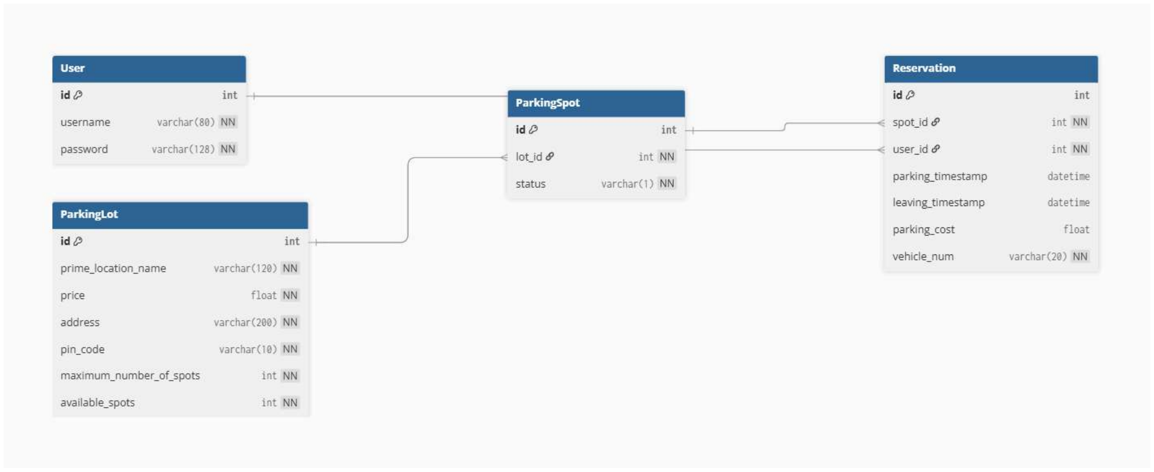
1. **Student Details**
    - Name: Ankit Kumar
    - Roll Number: 23f3004032
    - Email: 23f3004032@ds.study.iitm.ac.in
    - About Me: I'm Ankit, a diploma level student and Math-I TA Since last two terms. Outside academics, I enjoy video editing—and thanks to this project, I've started getting hooked on coding too.

2. **Project Details**
    - Title: Vehicle Parking App
    - Problem Statement: A multi-user web application to manage 4-wheeler parking lots, spots, and reservations, with two roles (Admin and User).
    - Approach: I followed a modular, frontend-first approach—starting with responsive UI using HTML, Jinja2, and Bootstrap, then building backend features one by one. Each functionality like registration, lot management, and booking was tested as implemented. Finally, I enhanced UX by integrating a Gemini 2.0 Flash AI chatbot for real-time parking assistance.

3. **Frameworks & Libraries Used**
    - Flask - backend
    - Jinja2 + Bootstrap - frontend & responsive UI
    - SQLite/SQLAlchemy - database & ORM
    - Matplotlib-server-side charts
    - Python-GenAI/Gemini (AI assistant).

4. **Database Schema Design**

   Below is the ER diagram of our SQLite schema.



5. **API Resource Endpoints**

POST Endpoints
`/` (Sign-in)
`/signup` (Register)
`/reserve/<lot_id>/<username>` (Reserve)
`/leave/<id>/<username>` (Release Spot)
`/chatbot` (Chatbot Query)
`/admin/manage-lots/create` (Create Lot)
`/admin/edit_lot/<lot_id>` (Edit Lot)
`/admin/delete_lot/<lot_id>` (Delete Lot)

GET Endpoints
`/` (Render Sign-in)
`/signup` (Render Sign-up)
`/dashboard/<username>` (User Dashboard)
`/profile/<username>` (User Profile)
`/logout` (Logout)
`/lots/<username>` (View User Lots)
`/reserve/<lot_id>/<username>` (Show Available)
`/bookings/<username>` (User Bookings)
`/leave/<id>/<username>` (Leave Spot - State Change)
`/user/<username>/chart` (Parking Time Chart)
`/chatbot` (Render Chatbot)
`/admin/dashboard` (Admin Dashboard)
`/admin/lots` (Admin View All Lots)
`/admin/status` (Admin Lot Status)
`/admin/status/<lot_id>` (Admin Individual Lot Status)
`/admin/users` (Admin User List)
`/admin/charts` (Admin Analytics)

6. Architecture and Features

The Vehicle Parking App V1, built with Flask, follows a modular architecture for clear separation of concerns. Key components include:

- **`app.py`**: Manages all user-facing functionalities (login, signup, dashboard, bookings, chatbot).
- **`admin.py`**: Handles administrative tasks via Flask Blueprints (parking lot management, user oversight, analytics).
- **`models.py`**: Defines database models using SQLAlchemy ORM (User, ParkingLot, ParkingSpot, Reservation).
- **`templates/`**: Contains HTML files, separated for user and admin views, using Jinja2.
- **`static/`**: Stores static assets like charts and styles.
- **`instance/`**: Holds the `parking.db` SQLite database.
- **`requirements.txt`**: Lists all necessary Python packages.

Core Features Implemented:

1. **User Authentication:** Secure login/registration for users and admins, with role-based access control.
2. **Admin Dashboard:** Enables creating, editing, and deleting parking lots (with auto-generated spots), viewing real-time lot and spot availability, managing users, and accessing visual analytics.
3. **User Dashboard:** Allows browsing lots, reserving the first available spot, leaving spots (with automated cost calculation), and tracking personal booking history.
4. **Reservation System:** Features auto-allocation of spots, timestamp-based billing, and real-time updates of spot availability.
5. **Analytics:** Provides users with total parked time charts, and admins with app-wide metrics like revenue, bookings, and live occupancy status.
6. **AI Chatbot Assistant:** Integrates a Gemini 2.0 Flash-powered chatbot to answer user queries about the app.

7. Presentation Video

Link to demonstration