Project Report

Student Details:

Name**: Aditya Kumar Chaubey**

Roll No: **23f2005214**

Email: 23f2005214@ds.study.ac.in

Course: BS in Data Science and Application

Description:

This project is a web-based vehicle parking management system that allows users to book parking spots and view their history, while admins can manage lots, users, and spots. The goal is to create a modern, user-friendly, and database-driven solution.

Project Details:

Project Title: **Vehicle Parking App**

Question Statement: Design and implement a web-based vehicle parking management system that allows users to book parking spots, view history, and enables admins to manage lots, users, and spots. The solution should be modern, user-friendly, and database-driven.

Approach: I started by analyzing the requirements and breaking down the features for users and admins. I designed the database schema, set up the Flask backend, and created responsive dashboards for both user and admin roles. I used SQLAlchemy for ORM and Bootstrap for UI. I iteratively improved the UI/UX and fixed bugs as they arose.

Technologies Used

- Flask (web framework)

- Flask-SQLAlchemy (ORM for database)

- Bootstrap (UI framework)

- Font Awesome (icons)

- Werkzeug (security)

- SQLite (database)

```
Parking_management_23f2005214/
│
├── app.py                  # Main Flask application (routes, models, logic)
├── main.py                 # (Optional) Alternate entry point or legacy code
├── requirements.txt        # Python dependencies
│
├── instance/
│   └── parking_app.db      # SQLite database file (created at runtime)
│
├── static/
│   └── css/
│       └── style.css       # Custom CSS styles
│   └── ...                 # Other static assets (images, JS, etc.)
│
├── templates/
│   ├── base.html               # Base template (navbar, layout, etc.)
│   ├── login.html              # Login page
│   ├── register.html           # Registration page
│   ├── user_dashboard.html     # User dashboard
│   ├── admin_dashboard.html    # Admin dashboard
│   ├── book_parking.html       # Parking booking page
│   ├── create_parking_lot.html # Admin: create lot
│   ├── edit_parking_lot.html   # Admin: edit lot
│   ├── admin_parking_lots.html # Admin: lots list
│   ├── admin_parking_spots.html # Admin: spots list
│   ├── admin_users.html        # Admin: users list
│   └── ...                     # Other templates
│
└── .gitignore              # Files/folders to ignore in git (e.g., parking_app.db)
```

Purpose: Flask provides a lightweight web framework, SQLAlchemy simplifies database operations, Bootstrap ensures responsive design, and SQLite offers a simple, file-based database for development.

```
Features
- User registration and login
- Book parking spots
- Release parking spots
- View booking history
- Dashboard with stat cards: Current Parking, Total Bookings, Total Spent (₹),
Recent Parking Lot Used
- Admin dashboard: manage lots, spots, users, and view stats
- API endpoints for parking lots and spots
- Responsive design with Bootstrap and Montserrat font
```

DB Schema Design

User: id (PK), username (unique), password_hash, is_admin, created_at

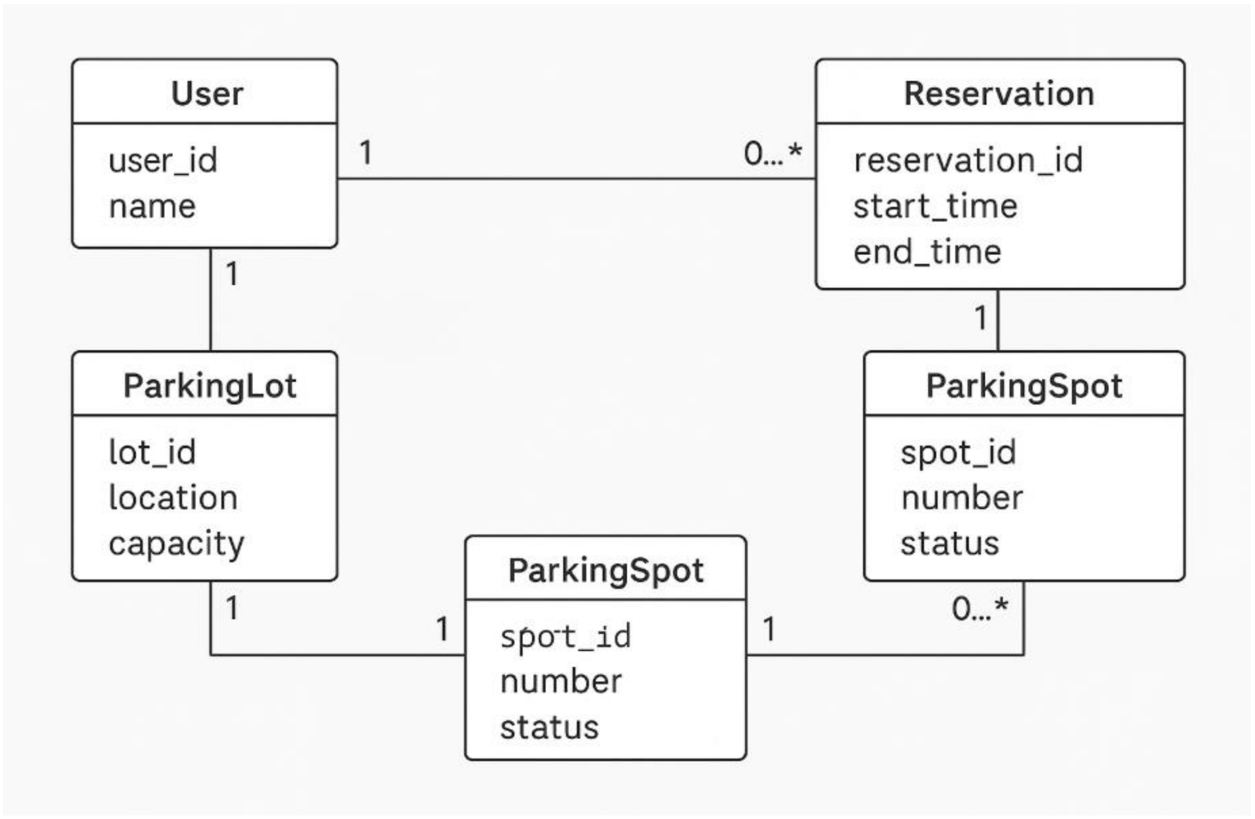
ParkingLot: id (PK), prime_location_name, price, address, pincode, maximum_number_of_spots,

created_at

ParkingSpot: id (PK), lot_id (FK), status, created_at

Reservation: id (PK), spot_id (FK), user_id (FK), parking_timestamp, leaving_timestamp, parking_cost, is_active

ER DIAGRAM:



Constraints: Unique usernames, foreign key relationships, default values for timestamps and status. The schema is designed for normalization, scalability, and clear relationships between entities.

API Design

APIs are created for parking lot and spot data retrieval. Endpoints return JSON data for integration with other services. Implementation uses Flask routes and SQLAlchemy queries.

Architecture and Features

The project is organized with controllers (routes) in app.py, templates in the templates/ folder, and static assets in static/. Features include user registration/login, booking/releasing parking spots, viewing history, and admin management of lots, spots, and users. Additional features: dashboard stats, search, and responsive UI.

Drive Link of Presentation Video:

https://drive.google.com/file/d/10nSpl0MN_odFkLAYryemHXJcGXMcATFB/view?usp=sharing