Project Report

Author

Name: Aman Kumar

Roll No.: 24f1002107

Email: 24f1002107@ds.study.iitm.ac.in

Introduction:

Myself Aman Kumar, a Diploma Level Student. Pursuing this degree as standalone.

Description:

Objective was to build a Vehicle Parking Management Application focusing on Flask as the backend framework for the web app. In this web app, I have implemented an Admin Dashboard from where the Admin can access and manage all the Parking based info and create new parking lots.

Tech Stack Used:

- **Backend**: Python Flask
- **Database**: SQLite with SQLAlchemy ORM
- **Frontend**: HTML5, CSS3, Bootstrap 5
- **Authentication**: Werkzeug password hashing
- **Session Management**: Flask sessions

Features:

For Users

- **User Registration & Authentication**: Secure user registration and login system
- **Parking Lot Browsing**: View all available parking lots with real-time availability
- **Spot Booking**: Book available parking spots with vehicle number
- **Parking History**: View complete parking history with duration and costs
- **Spot Release**: Release parking spots and get cost calculation
- **User Dashboard**: Personalized dashboard showing current and historical reservations

For Administrators

- **Admin Authentication**: Secure admin login system
- **Parking Lot Management**:
    - Add new parking lots with custom details
    - Edit existing parking lots (location, address, price, spot count)
    - Delete parking lots (when no spots are occupied)
    - View detailed parking lot information
- **User Management**: Monitor all users and their parking activities
- **Parking Records**: Complete history of all parking transactions
- **Advanced Search**: Search across users, spots, lots, and reservations
- **Real-time Statistics**: Occupancy rates and parking lot performance

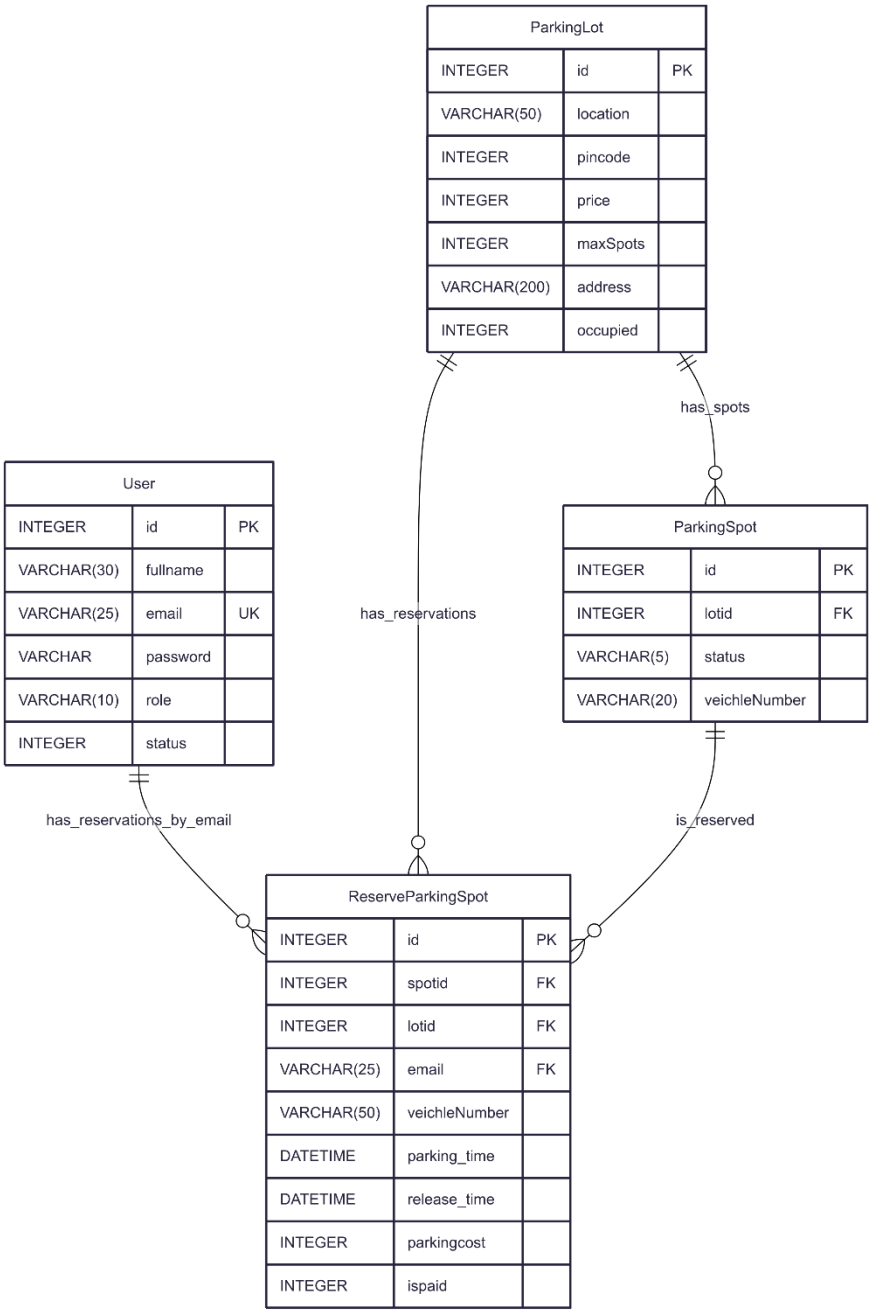
Security Features:

- **Password Hashing**: All passwords are hashed using Werkzeug
- **Session Management**: Secure session handling
- **Role-based Access**: Separate user and admin interfaces
- **Input Validation**: Form validation and sanitization
- **SQL Injection Protection**: SQLAlchemy ORM prevents SQL injection

Project Structure:

```
vehicle_parking_24f1002107/
├── app.py                          # Main Flask application
├── database.py                     # Database configuration
├── init_db.py                      # Database initialization script
├── requirements.txt                # Python dependencies
├── README.md                      # Project documentation
├── controllers/
│   ├── controllers.py             # Route handlers and business logic
│   └── forms.py                   # Flask-WTF form definitions
├── models/
│   └── models.py                  # SQLAlchemy database models
├── templates/                      # HTML templates
│   ├── index.html                 # Base template
│   ├── home.html                  # Home page
│   ├── login.html                 # User login
│   ├── register.html              # User registration
│   ├── admin_login.html           # Admin login
│   ├── user_dashboard.html        # User dashboard
│   ├── admin_dashboard.html       # Admin dashboard
│   ├── parking_lots.html          # Parking lots listing
│   ├── add_parking_lot.html       # Add parking lot form
│   ├── edit_parking_lot.html      # Edit parking lot form
│   └── ...                        # Other template files
└── instance/
    └── parking.db                 # SQLite database file
```

Database Schema:



Video Link:

https://drive.google.com/file/d/1v4YzPpUxOCoDlpw1xfpxt0KiXqJh7fU7/view?usp=sharing

Declaration: I have taken help from AI in styling and layout of my webapp. Used **8% AI** as used in CSS/Bootstrap.