VEHICLE PARKING - MAD 1

Author

Name: Chitrakshi

Roll number: 24f2004114

Email: [24f2004114@ds.study.iitm.ac.in](mailto:24f2004114@ds.study.iitm.ac.in)

About:

I am a Diploma level student at IIT Madras online degree . Also I am a second year student at IIIT Kalyani in the CSE department . My areas of interest are ML , AI, Data Science etc.

Description

**Problem Statement**

To design a smart and user-friendly web-based system to manage vehicle parking using a centralized platform, allowing admins to manage lots/spots and users to book and track parking in real-time.

**Approach**

* Defined relational database schema with Admin, User, Parking Lot, Spot, and Booking tables.
* Built a Flask-based web server with routes for user/admin actions.
* Used Jinja2 templating to pass data from backend to frontend.
* Applied Bootstrap for responsive design.
* Generated charts using Matplotlib for admin dashboards.
* Maintained login sessions and user authentication.

Technologies used

* Python 3 : Core language used to build the app
* Flask : Framework for routing and rendering templates
* Flask-SQLAlchemy: For interacting with the SQLite database
* SQLite: database
* HTML, Bootstrap : for UI and styling
* Matplotlib: Used to generate charts

DB Schema Design

The design uses five main tables:

1. Admin

**Column Type Description**

admin\_ID Integer (PK) auto incremented (but not really used)

username String(100) hardcoded but can be changed

password String(100) hardcoded but can be changed

This table is to authenticate admin . Username and password are initially hard coded but can be changed by visiting view/edit profile page

1. User

**Column Type Description**

user\_ID Integer (PK) unique, auto incremented

Username String(100) hardcoded but can be changed

password String(100) hardcoded but can be changed

fullname String(200) fullname of user

email\_ID String(200) user email address

address String(300) address

PinCode Integer Pincode

Username is unique so no duplication . Users are linked to the **Booking** table using a **one to many** relationship (user.booking)

1. Parkinglot

**Column Type Description**

Lot\_ID Integer (PK) unique, auto incremented

Prime\_Location\_Name String(200) unique name of the location

Price Float price for a spot

Maximum\_spots Integer capacity of the parking lot

address String(300) address of parking lot

PinCode Integer Pincode of location

Each parking lot has multiple spots . This is implemented through a **one to many relationship** with the **Spo**t table (parkinglot.spots).

1. Spot

**Column Type Description**

Spot\_ID Integer (PK) unique, auto incremented

Lot\_ID ForeignKey linked to Parkinglot.Lot\_ID

Status Boolean true if occupied, false if free

Each spot is associated with a parking lot.

1. Booking

**Column Type Description**

Booking\_ID Integer (PK) unique, auto incremented

user\_ID ForeignKey links to User.user\_ID

spot\_ID ForeignKey links to Spot.Spot\_ID

vehicle\_number String(20) vehicle number

booking\_time DateTime time when the booking was made

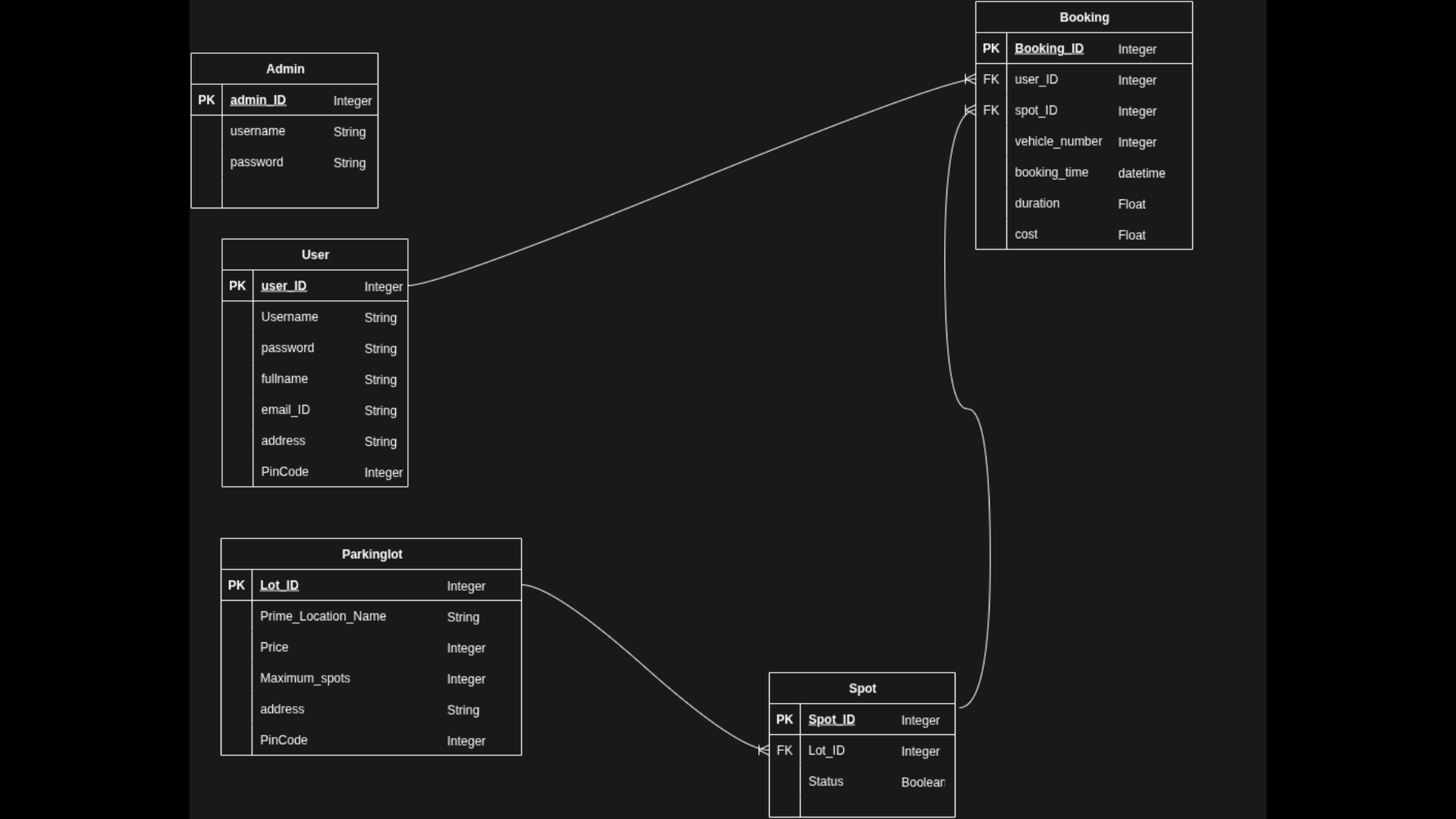
duration Float Duration in hours

cost Float total cost calculated

Each booking linked a user to the spot

Foreign keys and one to many relationships ensure scalable data linkage across tables.The spot table separates availability from the Parking Lot , allowing better slot-wise tracking. Keeping a Booking history table to show recent parking history to the user

ER Diagram



Architecture and Features

Structure

* [app.py](http://app.py) : Main app file to configure app
* Controllers: contains logic of user and admin actions
* Models: SQLAlchemy models for user, parking,booking etc
* Templates: HTML files
* Static: CSS and images used in frontend

Features Implemented

* Admin login and Dashboard
* User login and Dashboard
* Book and release parking spots
* Display parking availability
* Admin and user summary with graphs using Matplotlib
* Create and delete parking lot by admin

**Video link:**

https://drive.google.com/file/d/1Y6sUOCmdCv8wq5l5WZixro2GDeFz69qq/view?usp=sharing