MACHINE LEARNING

A Web-Based Car Rental Management System for Efficient Vehicle Search, Availability and Booking

M. Venkatesh, K. Murari, M. Ganesh, M. Madhu, Dharmendra K. Roy

Department of Computer Science and Engineering, Hyderabad Institute of Technology and Management, Hyderabad, India

Published November 2026

Abstract

This paper presents the design and development of a web-based Car Rental Management System that simplifies vehicle discovery, availability checking, pricing review, and reservation management. The proposed system replaces fragmented manual processes with a centralized application that connects customers, vehicle records, and booking information. The prototype uses a responsive front end with HTML, CSS, JavaScript and Bootstrap, while PHP and MySQL can be used as the server and database layers. The system is organized into customer, vehicle, booking, payment, and administrative modules. A database-driven availability workflow reduces duplicate booking risks by checking vehicle status against requested dates before a reservation is stored. The paper describes the system architecture, workflow, database design, implementation approach, and expected evaluation measures. The design is intended to be simple enough for small rental operators while remaining extensible for online payments, notifications, analytics, and mobile support.

Keywords: Car rental system; vehicle reservation; web application; PHP; MySQL; availability management; online booking

1 Introduction

Car rental services are becoming more convenient with the help of online platforms, where customers can easily search for cars, compare details, and book vehicles from anywhere. However, traditional rental methods often rely on phone calls, paper records, and spreadsheets, which can make it difficult to manage car availability and process bookings quickly.

To overcome these challenges, the proposed Car Rental Management System provides a simple and user-friendly web application. Customers can view available cars, check vehicle details, choose their rental dates, and submit booking requests online. Administrators can manage car information and monitor customer reservations through a centralized system. In the future, the system can also be enhanced with online payment facilities, automatic notifications, and report generation to make car rental management more efficient.

The work is motivated by existing research on online vehicle reservation and rental-management applications. For example, Hemalatha et al. describe a scalable online car reservation system implemented using the MERN stack and discusses the automation of transactions, scheduling, and vehicle inventory [1]. The present project follows the same broad digital-service direction but focuses on a lightweight architecture suitable for an academic prototype and small rental business.

2 Problem statement

Manual car rental management can create several operational difficulties: customers may not know real-time vehicle availability; booking information may be stored in separate records; price and vehicle details may require manual communication; and administrators may spend time checking reservations individually. Duplicate or conflicting reservations can occur when availability is not synchronized.

The objective of this project is to develop a user-friendly web-based system that centralizes vehicle information and booking records. The system should allow a customer to select a vehicle and rental period, validate availability, and create a reservation while allowing an administrator to maintain the fleet and monitor bookings.

3 Literature survey

Digital vehicle-rental systems commonly focus on online reservation, inventory management, customer registration, and automation of business transactions. Hemalatha et al. [1] present a flexible vehicle sharing and leasing application using the MERN stack and identifies online scheduling and inventory management as important functions. Nobayashi et al. investigate vehicle-management methods using location data and discusses their application to rental-car operations [2]. These studies indicate that centralized digital records and vehicle information can support more efficient rental workflows.

However, many student and small-business implementations use complex technology stacks that may increase deployment and maintenance requirements. This project therefore emphasizes a modular and understandable architecture using standard web technologies. The proposed design can later be upgraded with APIs, GPS-based tracking, recommendation algorithms, payment gateways, and cloud deployment.

4 Proposed system

The proposed system consists of a customer interface, application logic, database layer, booking module, and administrative module. The customer interface provides the home page, vehicle listing, vehicle details, login/registration, booking form, and contact information. The application layer validates user input, calculates rental duration and estimated cost, and checks whether a vehicle is available for the requested dates.

The database stores customers, vehicles, bookings, payments, and administrative information. The booking process is designed around an availability check so that a reservation is accepted only when the selected vehicle is not already booked for an overlapping period. This creates a single source of truth for vehicle and reservation records.

5 System architecture

Figure 1 illustrates the major components of the proposed architecture. A customer interacts with the responsive web interface. Requests are processed by the application layer, which communicates with the MySQL database. The booking and availability component handles date-based reservation validation, while the administration component supports fleet and booking management.

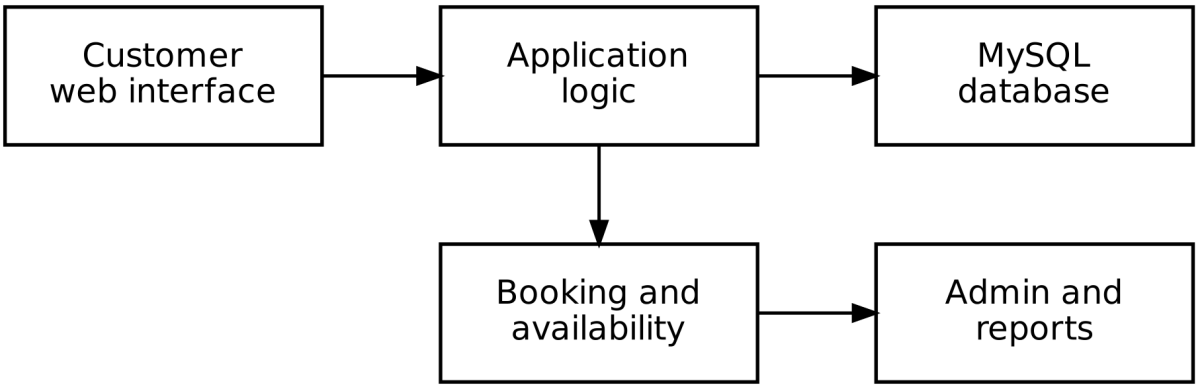


Figure 1. Proposed system architecture.

6 Modules

- **User Module:** Provides registration, login, profile information, vehicle browsing, and booking history.
- **Vehicle Module:** Stores vehicle name, category, model, registration information, rental price, status, and descriptive information. Example categories include sedan, SUV, and hatchback.
- **Search and Availability Module:** Accepts rental dates and filters vehicles based on their current status and existing bookings.
- **Booking Module:** Captures customer, vehicle, pickup date, return date, and booking status. It prevents an overlapping reservation from being stored when the vehicle is already reserved.
- **Payment Module:** Represents the payment stage of the workflow. The prototype can initially record payment status, while a real deployment can connect a compliant payment gateway.
- **Administration Module:** Allows authorized staff to add or update vehicles, review bookings, change availability status, and generate basic reports.

7 Methodology

The project follows an incremental software-development approach. First, requirements are identified from the activities performed by customers and rental administrators. Second, the user interface and database entities are designed. Third, vehicle search and booking functions are implemented. Fourth, validation and integration testing are performed. Finally, the interface is refined for usability and responsiveness.

The booking logic can be represented as: select vehicle → enter dates → validate dates → query existing bookings → determine availability → calculate estimated rental cost → save booking → show confirmation. Rental cost may be computed using a daily rate and the number of rental days, with additional charges incorporated when required by the business rules.

8 Database design

A relational database is appropriate because customers, vehicles, and bookings have well-defined relationships. The main entities are Customer, Vehicle, Booking, Payment, and Admin. A Customer can create multiple Bookings, while each Booking references one Vehicle. A Vehicle can therefore be associated with many bookings over time, provided their rental periods do not overlap.

A simplified relational structure is: Customer(Customer_ID, Name, Email, Phone, Password); Vehicle(Vehicle_ID, Model, Category, Rate, Status); Booking(Booking_ID, Customer_ID, Vehicle_ID, Pickup_Date, Return_Date, Total_Cost, Status); Payment(Payment_ID, Booking_ID,Amount,Payment_Status, Payment_Date). Primary and foreign keys maintain data consistency.

9 Implementation

The front end can be implemented using HTML5, CSS3, JavaScript and Bootstrap for responsive layouts. PHP can process form submissions and server-side validation, while MySQL can store persistent records. A simple MVC-like separation can be adopted so that interface code, business logic, and data access are easier to maintain.

The home page can present a hero section, vehicle cards, services, about information, contact details, and online-payment information. Each vehicle card can display an illustrative image, category, price, and a button to view details or book the vehicle. The interface should use clear labels and validation messages so that users understand the booking process.

10 Illustrative vehicle catalogue

The original illustrative vehicle images in Figure 2 represent the types of cars that can be displayed in the prototype. They are created for this project document and are not photographs of specific commercial vehicles.

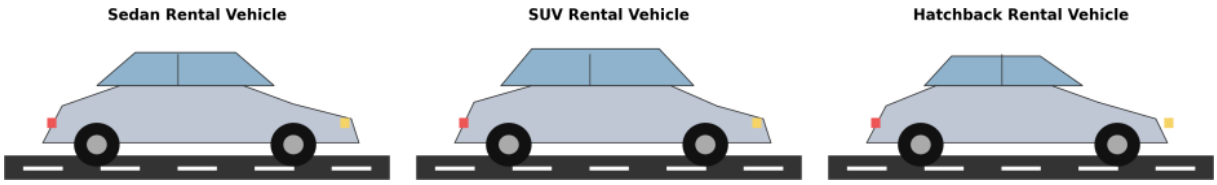


Figure 2. Sedan, SUV and hatchback illustrations used in the prototype vehicle cards.

11 Workflow

Figure 3 shows the complete customer booking workflow. The most important control point is the availability check, which occurs before the booking record is inserted. If the requested vehicle is unavailable, the system should return alternative available vehicles or request different dates.

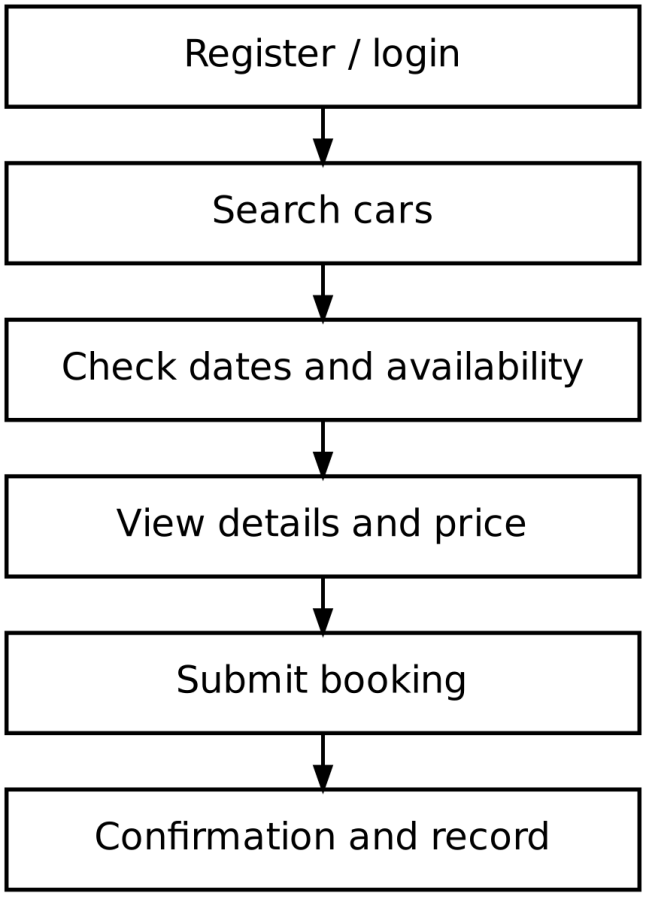


Figure 3. Customer booking workflow.

12 Expected results and evaluation

The prototype can be evaluated using functional and usability measures. Functional testing should verify registration, vehicle search, date validation, availability checking, booking creation, booking retrieval, and administrative updates. Database tests should verify that customer and vehicle relationships remain consistent.

A practical evaluation can record page response time, booking completion time, number of successful test bookings, number of rejected overlapping bookings, and user feedback on navigation. Because this is an academic prototype, the results should be reported from actual experiments rather than assumed performance values.

13 Challenges and solutions

One challenge is preventing duplicate reservations. The solution is to perform server-side overlap checks and use database constraints or transactions where appropriate. A second challenge is keeping the interface simple for first-time users; consistent navigation, readable vehicle cards, and short forms can address this. A third challenge is protecting customer data. Passwords should be securely hashed, server-side validation should be used, and database credentials should never be exposed in client-side code.

14 Future enhancements

Future versions can integrate secure online payment gateways, email/SMS notifications, GPS-based vehicle tracking, digital document verification, recommendation systems, customer reviews, dynamic pricing, mobile applications, and cloud deployment. Analytics can also be introduced to identify vehicle utilization and booking trends. These enhancements should be added only after the core reservation workflow is stable and secure.

15 Conclusion

This paper presented a web-based Car Rental Management System for centralizing vehicle information, availability checking, booking, and administration. The proposed architecture is modular and suitable for a student project or small rental operation. By replacing fragmented manual procedures with a database-driven workflow, the system can make vehicle search and reservation more organized. The prototype also provides a foundation for future extensions such as payment integration, notifications, analytics, and location-aware services.

References

[1] Hemalatha P, Dhavavarshini M, Dhivya N. Development of flexible, sharing and leasing a car using MERN stack. In: *2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE)*; 2022; Bangalore, India. p. 1–5. https://doi.org/10.1109/ICATIECE56365.2022.10047214

[2] Nobayashi D, Niwa Y, Tsukamoto K, Ikenaga T. Development of vehicle management system using location data collected by 920MHz LoRa. In: *2020 International Conference on Computational Science and Computational Intelligence (CSCI)*; 2020; Las Vegas, NV, USA. p. 208–209. https://doi.org/10.1109/CSCI51800.2020.00042