Malnad College Of Engineering

Under the auspices of M.T.E.S ® (An Autonomous Institution Affiliated to VTU, Belgaum)

P.B. No. 21, Hassan 573 202, Karnataka

Full-Stack Development(23IS553)

TOPIC: *CAR Rentals Web Application*

Submitted By:

Name	USN
A Y Kaustubha Sai	4MC23IS002
Bhuvanesh A U	4MC23IS020
Likhith Arya D	4MC23IS056
Manthan Nayak	4MC23IS057

Under The Guidance of:

Mr. Krishna Swaroop A

(Assistant Professor, ISE)

Department of Information Science and Engineering

Malnad College of Engineering

Hassan - 573 202

TABLE OF CONTENTS

1. ABSTRACT

The increasing growth of the transportation and tourism sector has created a strong demand for efficient, reliable, and technology-driven car rental services. Traditional car rental operations often rely on manual processes such as maintaining physical registers, handling phone inquiries, and manually tracking vehicle availability. These outdated methods result in operational inefficiencies, customer dissatisfaction, booking conflicts, inconsistent record maintenance, and overall reduced service quality.

To address these challenges, this project implements a **Car Rental Web Application** built using Django, a robust and scalable backend framework. The system offers end-to-end automation of car rental processes, from user login/registration to browsing available cars, searching by brand or model, viewing detailed specifications, and completing a booking. Administrators can manage the entire fleet, monitor availability, update car records, oversee bookings, and ensure smooth workflow operations.

The system incorporates modern UI enhancements through **Bootstrap**, **CSS**, **AOS animations**, and a **blue neon-themed interface**, providing a smooth, interactive, and visually appealing user experience. Real-time updates, database-driven design, secure authentication, and modular architecture contribute to the system's efficiency. Ultimately, the application streamlines business operations, improves accuracy, and enhances user satisfaction.

2. INTRODUCTION

2.1 Background

With rising travel demands, car rental services have evolved into essential components of modern transportation infrastructure. Customers now expect instant access to rental information, online booking capability, transparency in pricing, and flexible selection options. Businesses that continue relying on manual methods struggle to meet such expectations.

Digital transformation in car rentals is not only beneficial but essential. A web-based system ensures that users can easily browse, compare, and book cars while enabling service providers to handle large-scale operations with precision.

2.2 Need for the System

Without an automated platform, a rental agency faces:

- Frequent calculation errors
- Slow response times
- No centralized customer database

- Poor vehicle tracking
- Increased chances of double-booking
- Difficulty maintaining service quality

The need for a **centralized, digital, real-time platform** becomes critical for maintaining accuracy, reducing workload, and improving customer satisfaction.

2.3 Problem Identification

Key problems identified:

1. **Complexity in managing large vehicle inventories**
2. **Difficulty in updating availability instantly**
3. **Miscommunication between staff and customers**
4. **Inability to handle multiple bookings efficiently**
5. **No integrated platform for booking history and customer information**

2.4 Proposed Solution

A web-based Car Rental System offers:

- Automated rental workflow
- Real-time availability checks
- Secure login for customers
- Centralized database
- Detailed car listings with images
- Accurate rental date validation
- Instant booking confirmation

The proposed Django-based application is designed to be scalable, user-friendly, and efficient.

3. OBJECTIVES OF THE PROJECT

The primary objectives include:

- To develop a structured and automated system for managing car rentals.
- To eliminate manual operations by transforming them into digital processes.

- To create a secure login/registration module using Django Authentication.
- To enable customers to browse, compare, and book cars with ease.
- To maintain a database of bookings, customers, and vehicles with minimal redundancy.
- To support administrators with user-friendly tools to manage the fleet.
- To develop a responsive, attractive, and intuitive interface using modern web technologies.
- To ensure fast system performance through optimized backend logic.

4. SCOPE OF THE PROJECT

The Car Rental Web Application serves:

- Customers seeking cars for short-term or long-term travel
- Car rental agencies with multiple vehicles
- Admins managing rental operations

In-Scope Features

- User Registration & Login
- Car Search & Filtering
- Real-time Availability
- Booking System
- Admin Dashboard (Django Admin)
- View Bookings

Out-of-Scope (Future Enhancements)

- Automatic GPS car tracking
- Offline booking support
- Mobile application integration

5. SYSTEM REQUIREMENTS

5.1 Software Requirements (Expanded)

- Operating System: Windows / Linux
- Python 3.10+
- Django 5.x
- SQLite (default) or MySQL (optional upgrade)
- HTML5, CSS3, Bootstrap 5, JavaScript
- AOS.js for animation
- VS Code / PyCharm
- Browser (Chrome, Firefox, Edge)

5.2 Hardware Requirements

- Minimum RAM: 4 GB
- Recommended RAM: 8 GB
- Processor: Intel i3 / Ryzen 3 or higher
- Disk Space: 10 GB for project & dependencies

6. SYSTEM ANALYSIS

6.1 Existing System

The existing manual system involves:

- Manual booking entries
- Maintaining physical registers
- Phone and offline-based inquiries
- Spreadsheets for storing records

6.2 Drawbacks

- Errors in manual entries
- Time-consuming booking process
- Low efficiency in managing multiple customer requests

- No integrated availability status
- Poor customer experience

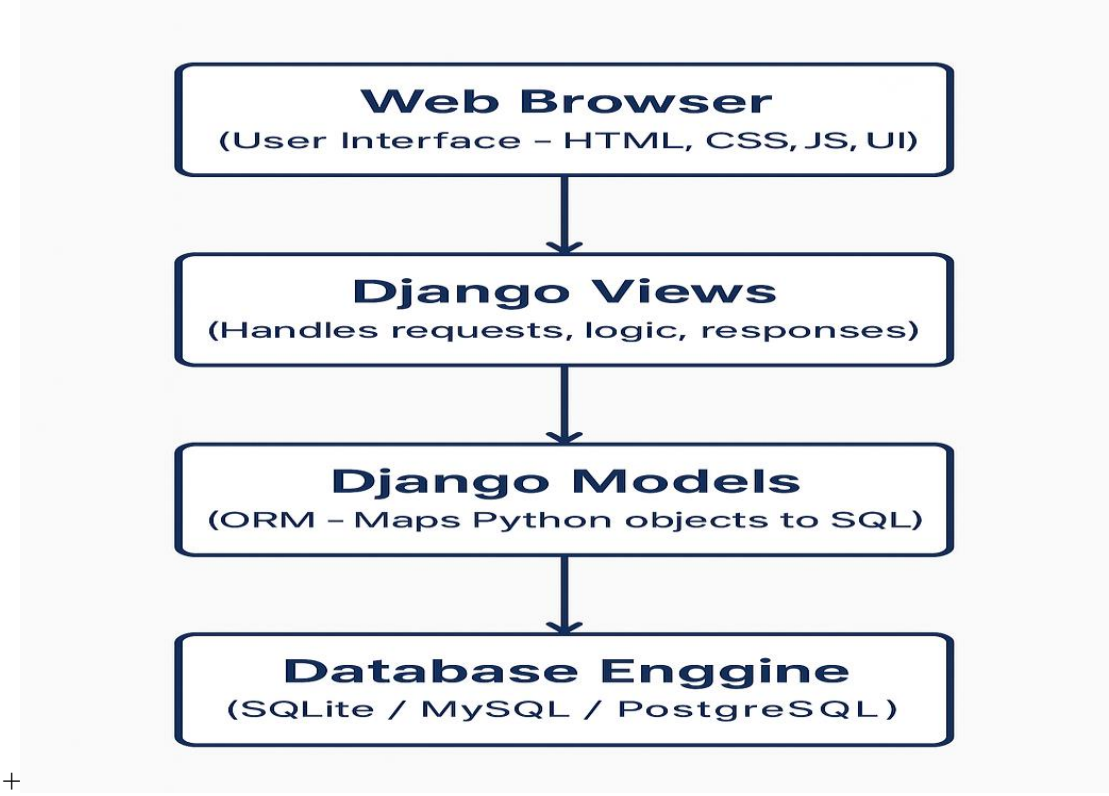
6.3 Proposed System

The proposed Django system provides:

- Real-time booking
- Online browsing
- Secure authentication
- Centralized database
- Scalable architecture
- Reduced operational cost

7. SYSTEM DESIGN

7.1 System Architecture Diagram

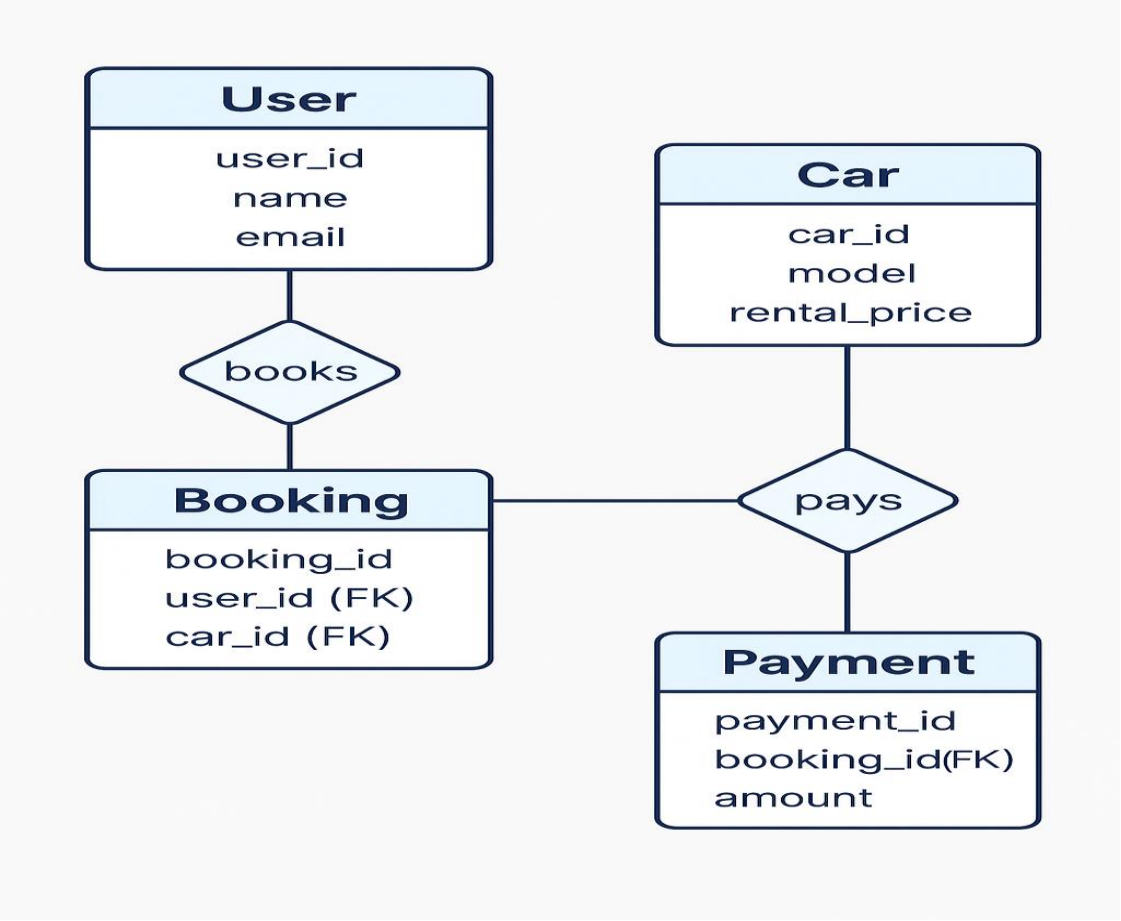


Explanation

- The browser sends requests through URLs.
- Views process logic and interact with models.

- Models communicate with the database through Django ORM.
- The final processed response is rendered via templates.

7.2 ER DIAGRAM



Explanation

- **CarBrand → Car:** One brand contains many cars
- **User → Booking:** One user can make multiple bookings
- **Car → Booking:** One car can have multiple bookings at different times

7.3 Data Flow Explanation

User Flow

1. User enters website
2. Views homepage with car listings
3. Searches by brand/model

4. Views car details
5. Logs in/registers
6. Books a car
7. Booking stored in database
8. Confirmation shown

Admin Flow

1. Logs into admin panel
2. Adds car brands and new cars
3. Monitors bookings
4. Updates availability

8. DATABASE DESIGN

8.1 Schema Explanation

The system uses SQL tables to store:

- User data
- Car brands
- Car information
- Bookings

Data is normalized to avoid redundancy.

8.2 Table-Level Description (Elaborated)

User Table

Stores customer details, login credentials.

CarBrand Table

Stores brand names like:

- Toyota
- Hyundai
- BMW

Car Table

Stores car details and images.

Booking Table

Maintains complete booking history.

9. MODULE DESCRIPTION

9.1 Authentication Module

- User registration
- Login
- Session handling
- Secure password storage

9.2 Car Management Module

- Add/edit/delete cars
- Upload car images
- Set availability

9.3 Car Search Module

- Keyword search
- Filters using Django ORM
- Instant result display

9.4 Booking Module

- Validate dates
- Prevent overlapping booking
- User-specific booking history

9.5 UI Module

- Blue neon theme
- AOS animations
- Mobile responsiveness

10. IMPLEMENTATION

Implementation includes:

- Setting up Django project
- Creating models for car, brand, booking
- Registering models in admin
- Creating views for listing, filtering, booking
- Designing HTML templates
- Adding CSS theme
- Writing logic for booking validation
- Using Django URL router
- Implementing authentication
- Testing and improving UI

11. TESTING

Testing includes:

- Unit testing model validations
- Checking authentication flow
- Checking booking overlap prevention
- UI responsiveness testing
- Browser compatibility testing
- Admin panel testing

12. RESULTS

- The system successfully automates car rentals
- Provides fast searching and browsing
- Booking confirmation shown instantly
- No manual errors in availability tracking

- Better user experience with neon UI
- Admin can manage the fleet efficiently

13. CONCLUSION

The development of the Car Rental Web Application showcases how digital transformation can significantly enhance service quality and operational workflow. The system provides an efficient and scalable solution for rental businesses, enabling real-time access, reducing workload, and ensuring secure booking. With Django's robust structure and Bootstrap's modern UI capabilities, the application provides both functionality and aesthetic appeal.

14. FUTURE ENHANCEMENTS

- Integrated payment gateways
- Push notifications
- GPS-based real-time tracking
- Driver hiring module
- Dynamic pricing based on season/demand
- Mobile app for customers

15. REFERENCES

- Django Documentation
- Bootstrap Documentation
- Python Official Docs
- W3Schools
- MDN Web Docs