RV COLLEGE OF ENGINEERING®, BENGALURU-560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Car Showroom Management System

Mini - Project Report

Submitted by

HARDIK HIRAMAN PAWAR	***1RV21CS046***
HARSHIT DHOOT	***1RV21CS049***
KARAN SATHISH	***1RV21CS058***

in partial fulfillment for the requirement of 5th Semester

Introduction to Database Systems Laboratory Mini Project (21CS53)

Under the Guidance of
Dr. Pratiba .D

Department of CSE, RVCE,
Bengaluru - 560059

RV COLLEGE OF ENGINEERING®, BENGALURU - 560059
(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work titled **"Car Showroom Management System"** is carried out by **Hardik Hiraman Pawar (1RV21CS046) , Harshit Dhoot (1RV21CS049), Karan Sathish(1RV21CS058)** who are bonafide students of RV College of Engineering®, Bengaluru, in partial fulfillment of the curriculum requirement of 5th Semester Database Design Laboratory Mini Project during the academic year **2023-2024**. It is certified that all corrections/suggestions indicated for the internal Assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the academic requirements in all respect laboratory mini-project work prescribed by the institution.

Signature of Faculty In-charge

Head of the Department
Dept. of CSE, RVCE

<u>External Examination</u>

Name of Examiners	Signature with date
1	
2	

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project work. I would like to take this opportunity to thank them all.

I deeply express my sincere gratitude to my guide **Internal Guide, Designation,** Department of CSE, RVCE, Bengaluru, for his able guidance, regular source of encouragement and assistance throughout this project

I would like to thank Dr.Ramakanth Kumar P, Head of Department, Computer Science & Engineering, R.V.C.E, Bengaluru, for his valuable suggestions and expert advice.

First and foremost I would like to thank **Dr. Subramanya. K. N**, Principal, R.V.C.E, Bengaluru, for his moral support towards completing my project work.

I thank my Parents, and all the Faculty members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least, I would like to thank my peers and friends who provided me with valuable suggestions to improve my project.

Abstract

Traditional car showroom management systems often rely on manual processes and disconnected databases, leading to inefficiencies in data management, customer interaction, and payment processing. This project addresses these shortcomings by developing a web-based car showroom management system that utilizes a combination of Flask, MySQL, and NoSQL databases.

The system offers a comprehensive suite of functionalities, including car listing management, customer interaction tools, test drive booking, sales tracking, and employee incentive tracking. A user-friendly web interface and unique QR code generation for customer identification prioritize accessibility and streamline both customer and employee interactions. Additionally, the integration of two secure payment gateways, Stripe and a blockchain-based option, provides flexibility and caters to diverse customer preferences. A customer review system empowers customers to provide feedback, fostering a customer-centric approach.

The project successfully demonstrates the potential of technology to enhance car showroom management. The system streamlines operations, improves customer experience with a user-friendly interface and diverse payment options, and motivates employees through incentive tracking features. This foundation can be further developed to include advanced inventory management, in-depth financial reporting, and direct car service scheduling functionalities. By continuously improving upon the core functionalities and incorporating evolving technologies, this system has the potential to significantly improve efficiency, customer satisfaction, and employee engagement within the car showroom industry.

Table of Contents

List of Figures

GLOSSARY

SRS	:	Software Requirement Specification

Chapter 1
Introduction

The ever-evolving automotive industry demands efficient management systems that streamline operations and enhance customer experience. This project addresses this need by developing a web-based car showroom management system that utilizes a combination of Flask, MySQL, and NoSQL databases.

Project Objectives:

- **Streamline Showroom Operations:** Develop a comprehensive system to manage car listings, customers, test drive bookings, and sales data, facilitating efficient workflow within the showroom.
- **Enhance User Experience:** Design a user-friendly web interface that is accessible and simplifies interactions for both customers and showroom employees.
- **Offer Diverse Payment Options:** Integrate two secure payment gateways: Stripe for traditional transactions and a blockchain-based option for added security and cater to evolving customer preferences.
- **Empower Customers:** Implement features like unique QR code generation for identification and a review system to provide feedback on employee performance, fostering a customer-centric approach.
- **Motivate Employees:** Integrate functionalities for employees to manage sales data and track their associated incentives, promoting a results-oriented environment.

Project Scope:

This project focuses on developing a web-based application using Flask for the user interface and a combination of MySQL and NoSQL databases for data storage. The system will encompass functionalities relevant to car showroom management, customer interaction, employee performance tracking, and secure payment processing with two integrated gateways.

While the core functionalities cater to core showroom needs, the scope excludes functionalities like advanced inventory management, in-depth financial reporting, or direct car service scheduling. These functionalities can be considered for future enhancements based on evolving requirements.

This report will delve deeper into the system's architecture, functionalities, and implementation details, demonstrating how it achieves the outlined objectives within the defined scope. It will highlight the project's contribution to improved car showroom management, enhanced customer experience, and employee engagement.

Chapter 2
Software Requirement specification

A software requirements specification (SRS) is a description of a software system to be developed. The software requirements specification lays out functional and non-functional requirements, and it may include a set of use cases that describe user interactions that the software must provide to the user for perfect interaction.

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function (in a market-driven project, these roles may be played by the marketing and development divisions). Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules.

2.1 Hardware Requirements

1) QR Scanners:

High-resolution document scanners with QR capabilities

- Reliable High-speed Ethernet:
  For reliable database connectivity with MongoDB at all times

2.2 Software Requirements

1. Relational Database Management System (RDBMS): MySQL

Store structured data related to the relations for employees, customers, cars, department, sale, etc. maintaining relationships between entities while ensuring data integrity & consistency while retrieval & update operations on the database.

2. NoSQL Database: MongoDB

Generated ID card images for customers & employees and customer feedback will be stored the online non-relational database – MongoDB in the form of a document-oriented structure using JSON-like documents, ensuring high performance, scalability, and support for flexible querying and indexing.

3. QR Software: qrcode

The unique ID generated for each employee and customer will be embedded in the QR code present in the ID card image.

4. Web Development Framework: Flask

The main heavyweight of the Prototype, Flask (microframework in Python) will be used to create CRUD API Endpoints, handle user interactions and facilitate overall rendering of webpages and data transfer.

5. Frontend Framework: Tailwind CSS

Designed to style the user interface using utility classes for rapid development, ensuring a consistent and customizable look and feel throughout the various sections and components of the website.

2.3 Functional Requirements

1) Employee & Customer Management using QR ID cards

- Customers should be able to log in with a unique username and password and track their purchase history, test drives, appointments, etc.
- Employees will be able to see their performance, sales made for a particular car, appointments in which they must partake, etc.
- Perform CRUD (Create, Read, Update, Delete) operations on various relations with concurrency control.

2) Communication channels (email, SMS) integration

Reminders regarding date of sale, appointments & test drives will be pushed to the customers via Meta API to alert on WhatsApp and SMTP protocol to inform on the registered email-id.

3) Feedback System

- Allow customers to provide feedback on the system.

- Admins should be able to view and analyse feedback for continuous improvement.

4) Car Sale Records

- Record details of car sales transactions, including customer, employee, and car details.
- Retrieve and display sales records based on various criteria.

5) Web API Endpoints

- Implement CRUD API endpoints using Flask for all major entities (employees, customers, sales records).
- Secure API endpoints to prevent unauthorized access.

Chapter 3
Entity Relationship Diagram

The following figure 3.1 outlines the design of a database system for a car showroom. This system aims to effectively manage information about employees, customers, cars, departments, appointments, test drives, sales, reviews, and the relationships between them. By establishing a well-structured database, the showroom can streamline operations, gain valuable insights, and ultimately enhance customer experience.

This figure 3.1 details the entities within the system, their attributes, and the crucial relationships that connect them. These relationships define how data is linked and accessed, ensuring a cohesive and efficient information management system for the car showroom.

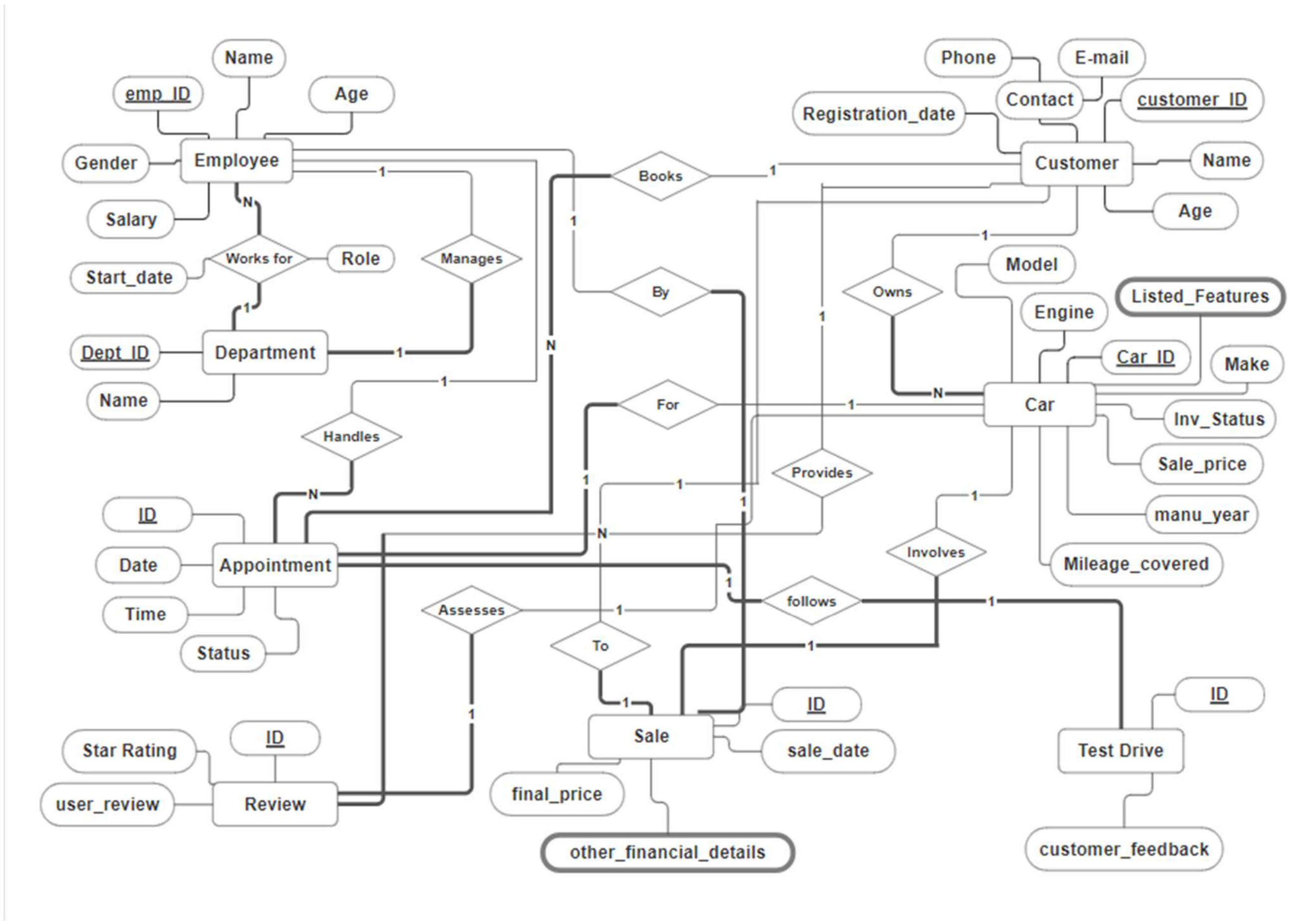


Fig 3.1

Employee

Each employee within the showroom can be identified by a **unique employee ID**. They have details like name, age, gender, and a fixed salary. Every employee is assigned to a **specific department** with a **designated role** (e.g., sales representative, manager) and started their employment on a particular date.

Customer

Every customer visiting the showroom is assigned a **unique customer ID**. They possess attributes like name, age, contact information (phone number, email), and the date they first registered with the showroom. Their purchase history (if any) is recorded, along with any raw data extracted from their **scanned ID** through **OCR** technology (stored in NoSQL).

Car

Each car available in the showroom has a **unique identifier**. Its details include make, model, year of manufacture, mileage covered, engine type, listed features, high-quality images and videos (stored in NoSQL), its current inventory status (e.g., available, sold), sale price, and a detailed maintenance history (stored in NoSQL).

Department

Each department within the showroom has a **unique identifier** and a descriptive name. Additionally, each department is assigned a specific manager using their employee ID.

Appointment

Every scheduled appointment between a customer and an employee for a specific car has a **unique identifier**. It's linked to both the customer and employee involved, along with the chosen car. Furthermore, it records the appointment date and time, and its current status (pending, confirmed, completed).

Test Drive

After an appointment, a test drive conducted by a customer, employee, and specific car can be registered with a **unique identifier**. It's linked to the associated appointment and captures any customer feedback provided about the test drive experience.

Sale

When a car is successfully sold, a detailed sale record is created with a **unique identifier**. It links the customer who purchased the car, the employee who facilitated the sale, and the specific car itself. The record documents the sale date, final price, chosen payment method, and any relevant finance details.

Review

Every customer can leave a review for a specific car they interacted with, assigned a **unique identifier**. The review links the customer and car involved, captures a star rating, and allows the customer to express their written thoughts or experiences in detail.

Relationships

One-to-One

- **WORKS_FOR**: One Employee belongs to one Department: employee.department_id references department.department_id
- **APPOINTMENT_FOR**: One Appointment is for one Car: appointment.car_id references car.car_id
- **FOLLOWS**: One Test Drive is linked to one Appointment: test_drive.appointment_id references appointment.appointment_id
- **INVOLVES**: One Sale is for one Car: sale.car_id references car.car_id
- **SALE_BY**: One Sale is made by one Employee: sale.employee_id references employee.employee_id
- **SALE_TO**: One Sale is made to one Customer: sale.customer_id references customer.customer_id

- **ASSESSES**: One Review is for one Car: review.car_id references car.car_id

One-to-Many

- **OWNS**: One Customer can have many Cars: customer.customer_id references car.customer_id (optional, tracks only cars purchased by the customer)
- **HANDLES**: One Employee can have many Appointments: employee.employee_id references appointment.employee_id
- **BOOKS**: One Customer can have many Appointments: customer.customer_id references appointment.customer_id
- **PROVIDES**: One Customer can write many Reviews: customer.customer_id references review.customer_id

Chapter 4

Detailed Design

Fig 4.1 & 4.2 presents a series of Data Flow Diagrams (DFDs) designed to illustrate the information flow within the showroom management system. These DFDs provide a visual representation of how data is captured, processed, and utilized within the system, ensuring efficient operations and a seamless customer experience.

The DFDs are structured into two key levels:

- **Level 0 DFD (Context Diagram):** This high-level overview depicts the entire showroom management system as a single process and identifies its interaction with external entities, such as customers and suppliers. It focuses on the overall data exchange between the system and the outside world.
- **Level 1 DFD:** This level delves deeper into the system, breaking down the single process from the Level 0 DFD into more granular sub-processes. It illustrates how these sub-processes work together to manage various aspects of the showroom, including customer interaction, car inventory, appointments, sales transactions, and customer reviews. It details the data flow between these sub-processes, ensuring a clear understanding of how data is utilized within the system.

By analyzing these DFDs, stakeholders can gain valuable insights into the data flow within the showroom management system. This analysis can be used to:

- **Identify potential bottlenecks:** Detect areas where data processing might be slow or inefficient.
- **Optimize internal processes:** Streamline data exchange between sub-processes for improved performance.
- **Enhance data security:** Understand how data is accessed and stored to implement appropriate security measures.
- **Improve information accessibility:** Ensure relevant data is readily available to the appropriate personnel for informed decision-making.

These DFDs provide a foundational understanding of the data flow within the showroom management system, paving the way for further system development, optimization, and ultimately, a well-functioning and efficient environment for both customers and showroom staff.

4.1 DFD Level 0

The Level 0 DFD(Fig 4.1), also known as the context diagram, provides a high-level overview of the entire showroom management system. It focuses on the system itself and its interaction with external entities.

- **Process:** Showroom Management
- **External Entities:**
  - Customers: Interact with the showroom to browse cars, schedule appointments, make purchases, and leave reviews.
  - Suppliers: Provide cars for the showroom to sell.
- **Data Flows:**
  - Inbound:
    - Customer inquiries (e.g., car details, appointment requests)
    - Car information from suppliers
  - Outbound:
    - Appointment confirmations
    - Sales information (car details, purchase details)
    - Reviews (written feedback)

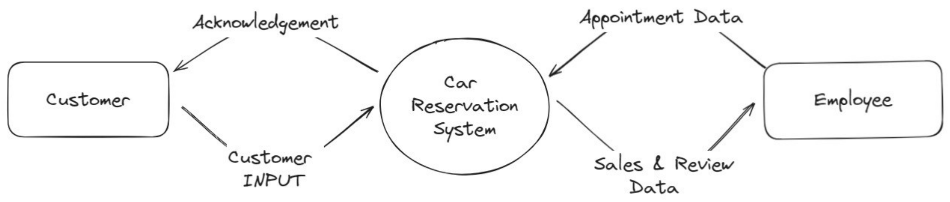


Fig 4.1

4.2 DFD Level 1

The Level 1 DFD breaks down the single process from the Level 0 DFD into more detailed sub-processes. It shows how the showroom management system achieves its overall function.

- **Process:** Showroom Management
    - Sub-processes:
        - **Customer Management:**
            - Handles customer registration and profile updates.
            - Tracks customer appointments and purchase history.
        - **Car Management:**
            - Tracks car inventory, details, and specifications.
            - Maintains car maintenance history.
            - Updates car status (available, sold, etc.)
        - **Appointment Management:**
            - Schedules appointments for customers to test drive cars.
            - Links customers, employees, and cars for appointments.
            - Tracks appointment status (pending, confirmed, completed).
        - **Sales Management:**
            - Processes car sales transactions.
            - Records sales details (date, price, payment method, finance details).
            - Links sales with customers, employees, and cars involved.
        - **Review Management:**
            - Captures customer reviews for cars.
            - Stores review details (star rating, written feedback).
            - Links reviews with customers and cars.
- **External Entities:** Same as Level 0 DFD (Customers, Suppliers)
- **Data Flows:**
    - Between Sub-processes: (examples)
        - Customer Management -> Appointment Management: Customer information for scheduling appointments.
        - Car Management -> Sales Management: Car details for sales transactions.

    - Appointment Management -> Sales Management: Appointment completion triggers sales recording.
    - Customer Management -> Sales Management: Customer information for sales records.
    - Customer Management -> Review Management: Customer information for linking reviews.
    - Car Management -> Review Management: Car information for linking reviews.
  - Between System and External Entities: (examples)
    - Customer inquiries -> Customer Management (Inbound)
    - Appointment confirmations -> Customers (Outbound)
    - Car information from Suppliers (Inbound) -> Car Management
    - Sales information -> Customers (Outbound) (e.g., invoices)
    - Reviews (written feedback) -> Customers (Outbound) (optional - confirmation or thank you)

This breakdown illustrates the internal functionalities of the showroom management system and how data flows between different sub-processes to achieve efficient customer service, car management, and sales operations. Remember, these are general examples, and the specific data flows and sub-processes can be further customized depending on the specific needs of the showroom.

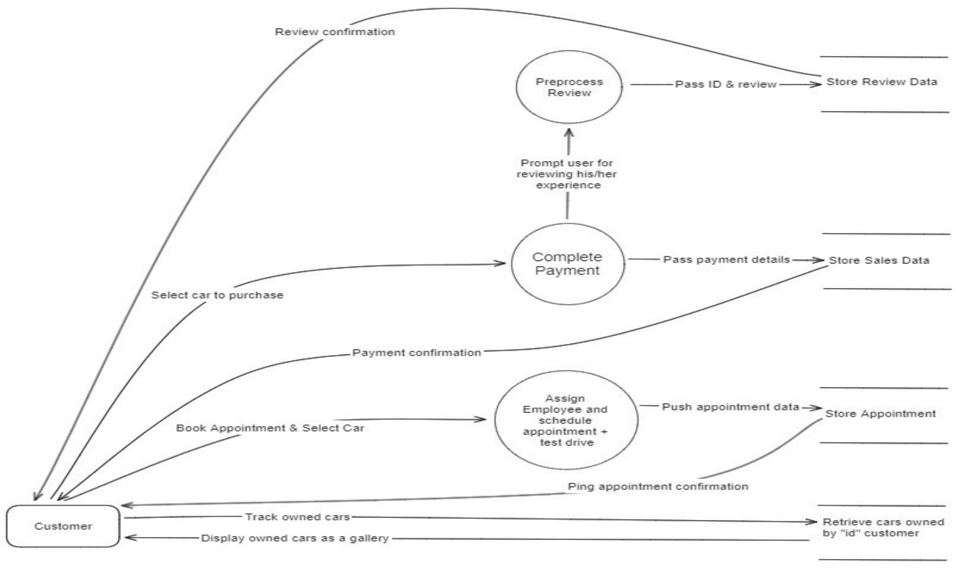


Fig 4.2

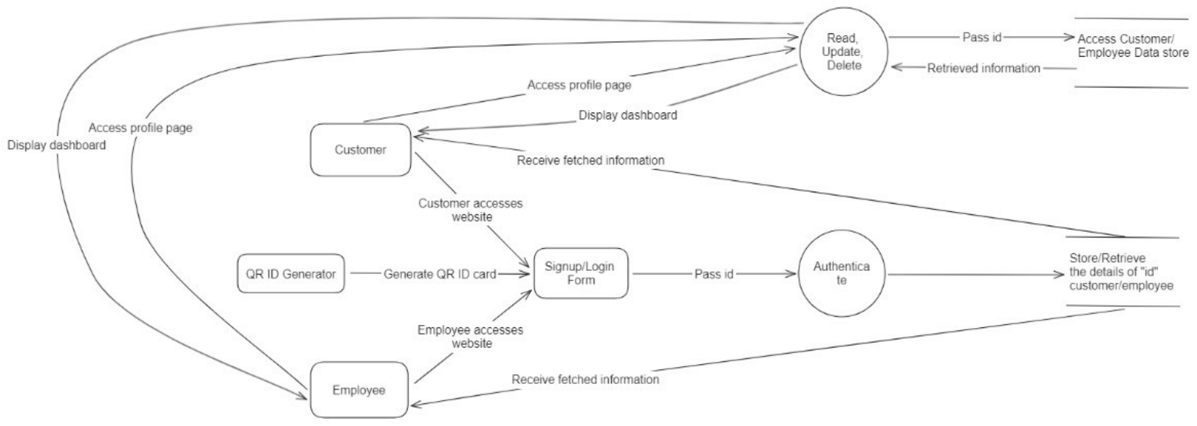

Read, Update, Delete
Pass id
Access Customer/ Employee Data store
Retrieved information
Access profile page
Display dashboard
Customer
Display dashboard
Access profile page
Receive fetched information
Customer accesses website
QR ID Generator
Generate QR ID card
Signup/Login Form
Pass id
Authentica te
Store/Retrieve the details of "id" customer/employee
Employee accesses website
Employee
Receive fetched information

Fig 4.3

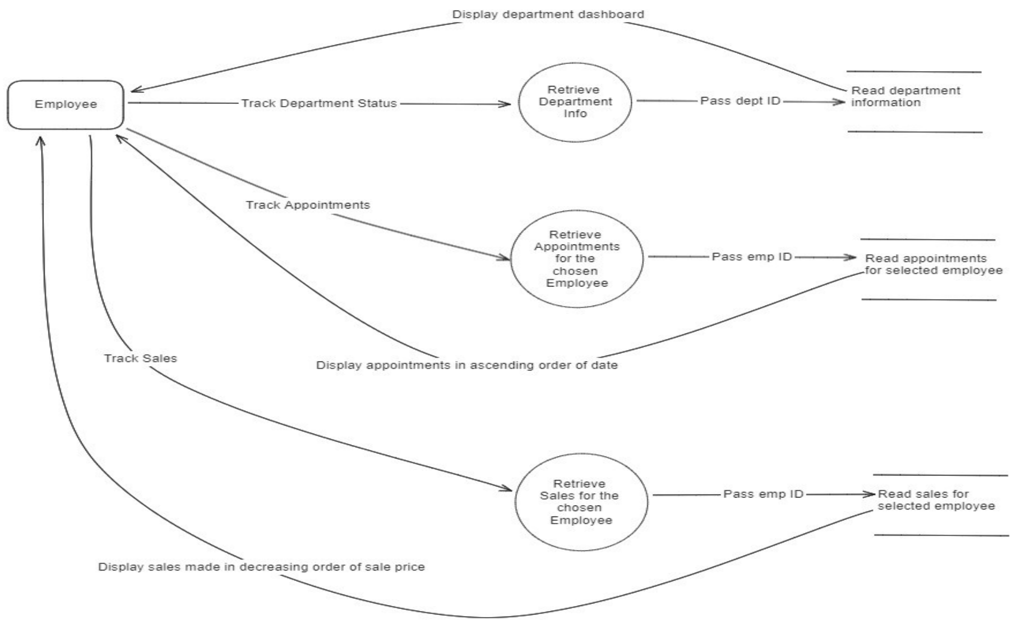

Display department dashboard
Employee
Track Department Status
Retrieve Department Info
Pass dept ID
Read department information
Track Appointments
Retrieve Appointments for the chosen Employee
Pass emp ID
Read appointments for selected employee
Display appointments in ascending order of date
Track Sales
Retrieve Sales for the chosen Employee
Pass emp ID
Read sales for selected employee
Display sales made in decreasing order of sale price

Fig 4.4

Chapter 5
Relational schema

This relational schema(Fig 5.1) outlines the structure of a database designed to manage a car showroom. It uses multiple tables to represent various entities within the system and their relationships.

Tables:

1. **Employee (emp_ID, Name, Age, Gender, Salary, works_for_dept_id):**
   - Stores information about employees, including their unique ID, name, age, gender, salary, and the department they work for (referenced by the dept_id foreign key).
   - **Primary Key (PK):** emp_ID (ensures each employee has a unique identifier).
   - **Foreign Key (FK):** works_for_dept_id references the dept_ID in the Department table, establishing a one-to-one relationship between an employee and their department.
2. **Department (dept_ID, Name, manager_emp_id):**
   - Stores information about departments within the showroom, including their unique ID, name, and the employee who manages the department (referenced by the manager_emp_id foreign key).
   - **Primary Key (PK):** dept_ID (ensures each department has a unique identifier).
   - **Foreign Key (FK):** manager_emp_id references the emp_ID in the Employee table, establishing a one-to-one relationship between a department and its manager.
3. **Appointment (app_ID, Date, Time, Status, handling_emp_id, booking_cust_id, appointment_for_car_id, test_drive_ID, customer_feedback):**
   - Stores information about customer appointments, including their unique ID, date, time, appointment status (pending, confirmed, completed), the employee handling the appointment (referenced by handling_emp_id foreign key), the customer who booked the appointment (referenced by booking_cust_id foreign key), the car the appointment is for (referenced by appointment_for_car_id foreign key), an optional test drive ID, and any customer feedback provided.
   - **Primary Key (PK):** app_ID (ensures each appointment has a unique identifier).
   - **Foreign Keys (FK):**
     - handling_emp_id references emp_ID in the Employee table (one employee can handle many appointments).

        - booking_cust_id references customer_ID in the Customer table (one customer can book many appointments).
        - appointment_for_car_id references car_ID in the Car table (one car can be involved in many appointments).

4. **Review (review_ID, Star_Rating, User_Review, review_cust_id, assessed_car_id):**
    - Stores customer reviews for cars, including their unique ID, star rating, written review text, the customer who wrote the review (referenced by review_cust_id foreign key), and the car being reviewed (referenced by assessed_car_id foreign key).
    - **Primary Key (PK):** review_ID (ensures each review has a unique identifier).
    - **Foreign Keys (FK):**
        - review_cust_id references customer_ID in the Customer table (one customer can write many reviews).
        - assessed_car_id references car_ID in the Car table (one car can have many reviews).
5. **Sale (sale_ID, sale_date, final_price, payment_method, sale_to_cust_id, sale_by_emp_id, sale_involved_car_id):**
    - Stores information about car sales, including their unique ID, sale date, final price, payment method, the customer who purchased the car (referenced by sale_to_cust_id foreign key), the employee who facilitated the sale (referenced by sale_by_emp_id foreign key), and the car that was sold (referenced by sale_involved_car_id foreign key).
    - **Primary Key (PK):** sale_ID (ensures each sale has a unique identifier).
    - **Foreign Keys (FK):**
        - sale_to_cust_id references customer_ID in the Customer table (one customer can be involved in many sales).
        - sale_by_emp_id references emp_ID in the Employee table (one employee can make many sales).
        - sale_involved_car_id references car_ID in

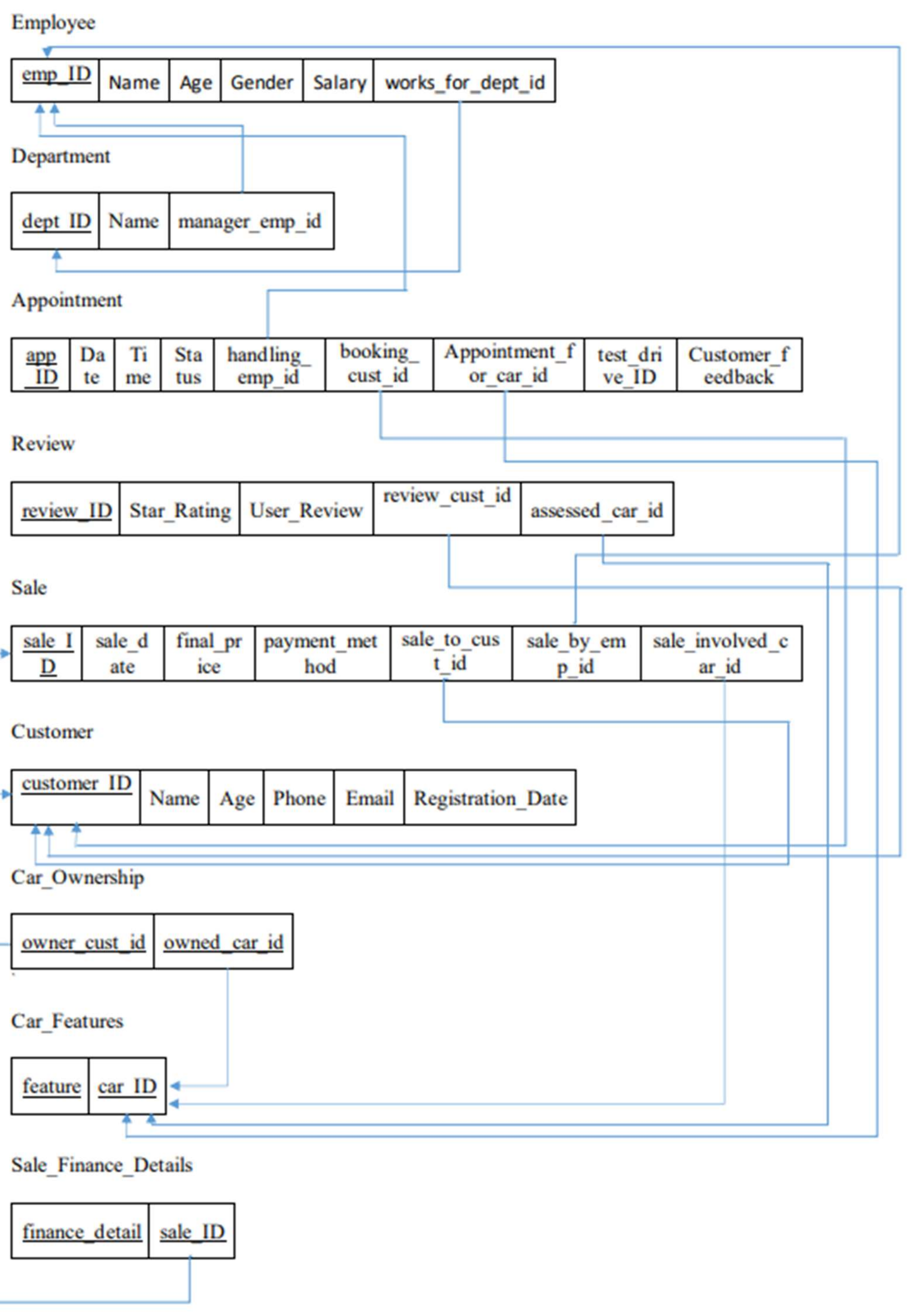

Employee
emp_ID
Name
Age
Gender
Salary
works_for_dept_id
Department
dept_ID
Name
manager_emp_id
Appointment
app_ID
Date
Time
Status
handling_emp_id
booking_cust_id
Appointment_for_car_id
test_drive_ID
Customer_feedback
Review
review_ID
Star_Rating
User_Review
review_cust_id
assessed_car_id
Sale
sale_ID
sale_date
final_price
payment_method
sale_to_cust_id
sale_by_emp_id
sale_involved_car_id
Customer
customer_ID
Name
Age
Phone
Email
Registration_Date
Car_Ownership
owner_cust_id
owned_car_id
Car_Features
feature
car_ID
Sale_Finance_Details
finance_detail
sale_ID

Fig 5.1

Normalization

The provided schema appears to be in 1NF (First Normal Form) already, as all attributes within each table contain single values and there are no repeating groups. However, we can further analyze it for higher normal forms (2NF and 3NF) to identify potential redundancies and improve data integrity.

1. First Normal Form (1NF):

The schema satisfies 1NF because:

- Each cell in the table contains a single atomic value (no lists or multiple values).
- There are no repeating groups of attributes within a table.

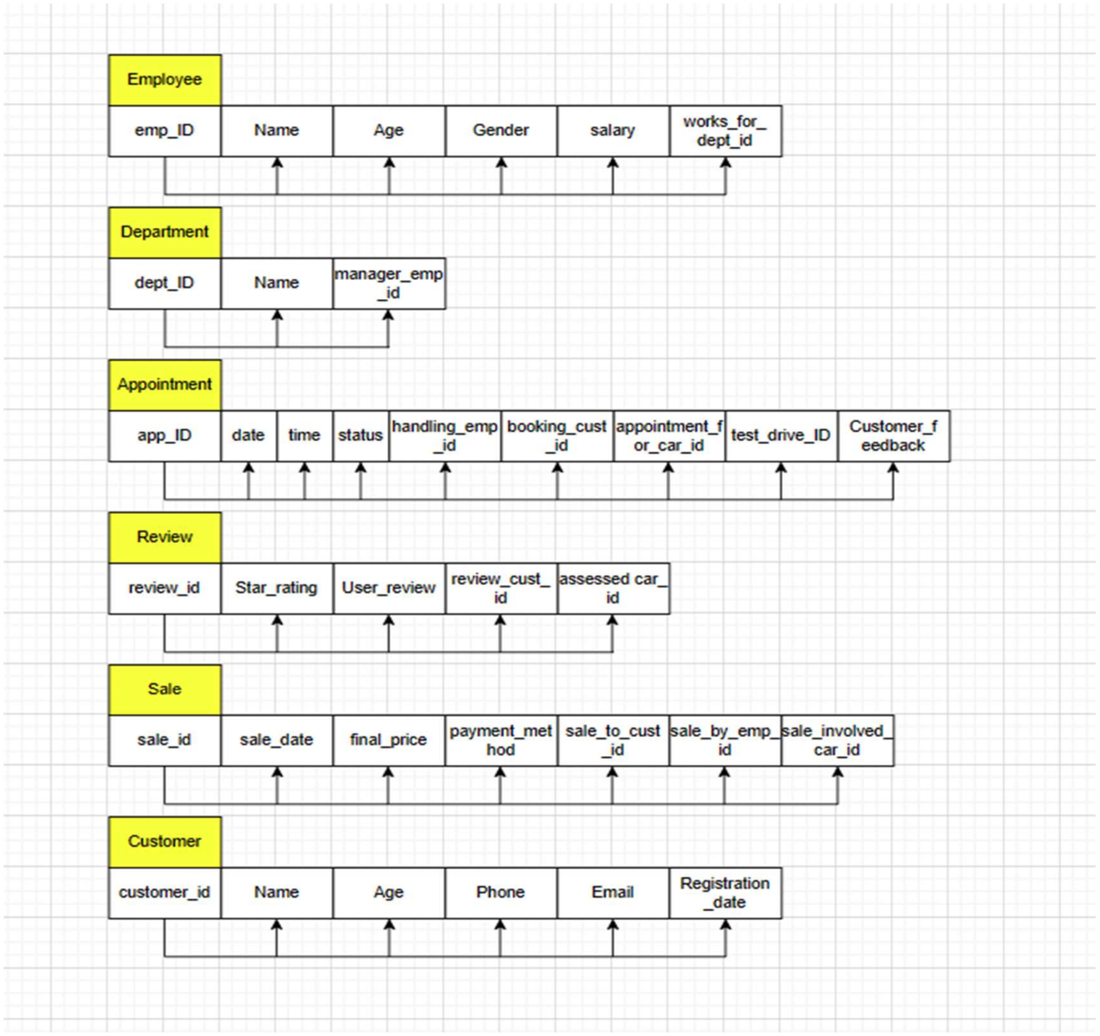


Fig 5.2

2. Second Normal Form (2NF):

The schema might have partial dependencies that violate 2NF in some tables. Let's analyze each table:

- **Employee:** This table is already in 2NF. All non-key attributes (Name, Age, Gender, Salary) are fully dependent on the primary key (emp_ID).
- **Department:** This table is also in 2NF. The non-key attribute (Name) is fully dependent on the primary key (dept_ID).
- **Appointment:** This table might violate 2NF. The attributes handling_emp_id, booking_cust_id, and appointment_for_car_id are all dependent on the primary key (app_ID) but might also be dependent on each other (an employee can't handle an appointment for a car that a different customer booked). This is a partial dependency.
- **Review:** This table is in 2NF. The non-key attributes (Star_Rating and User_Review) are fully dependent on the primary key (review_ID).
- **Sale:** This table might violate 2NF. Similar to the Appointment table, attributes sale_to_cust_id, sale_by_emp_id, and sale_involved_car_id are dependent on the primary key (sale_ID) but might also be dependent on each other (a customer can't buy a car from an employee who didn't sell it). This is a partial dependency.
- **Customer (optional table):** This table is in 2NF. All attributes are dependent on the primary key (customer_ID).
- **Car:** This table is in 2NF. All attributes are dependent on the primary key (car_ID).

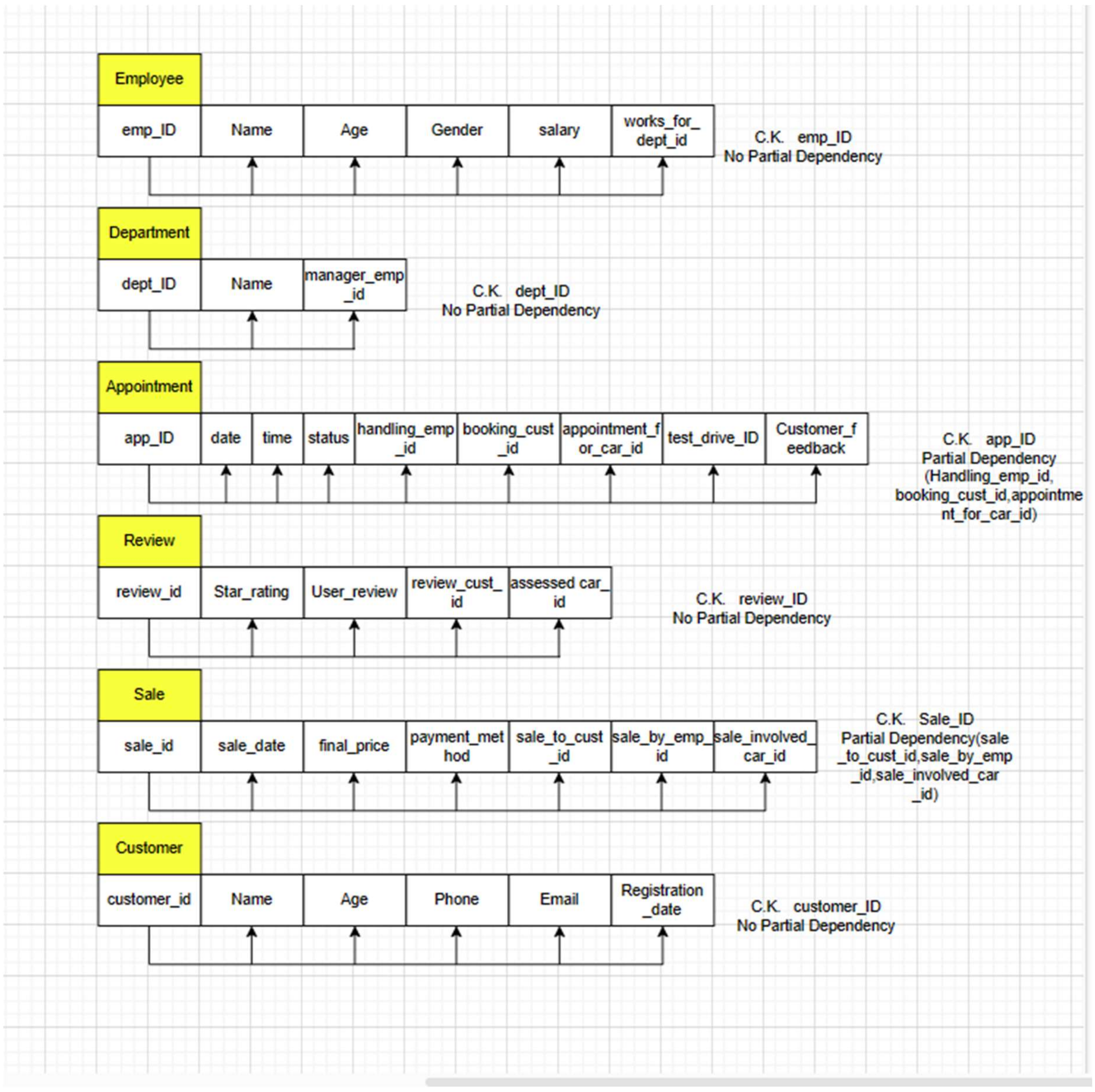


Fig 5.3

After Reducing to 2NF:

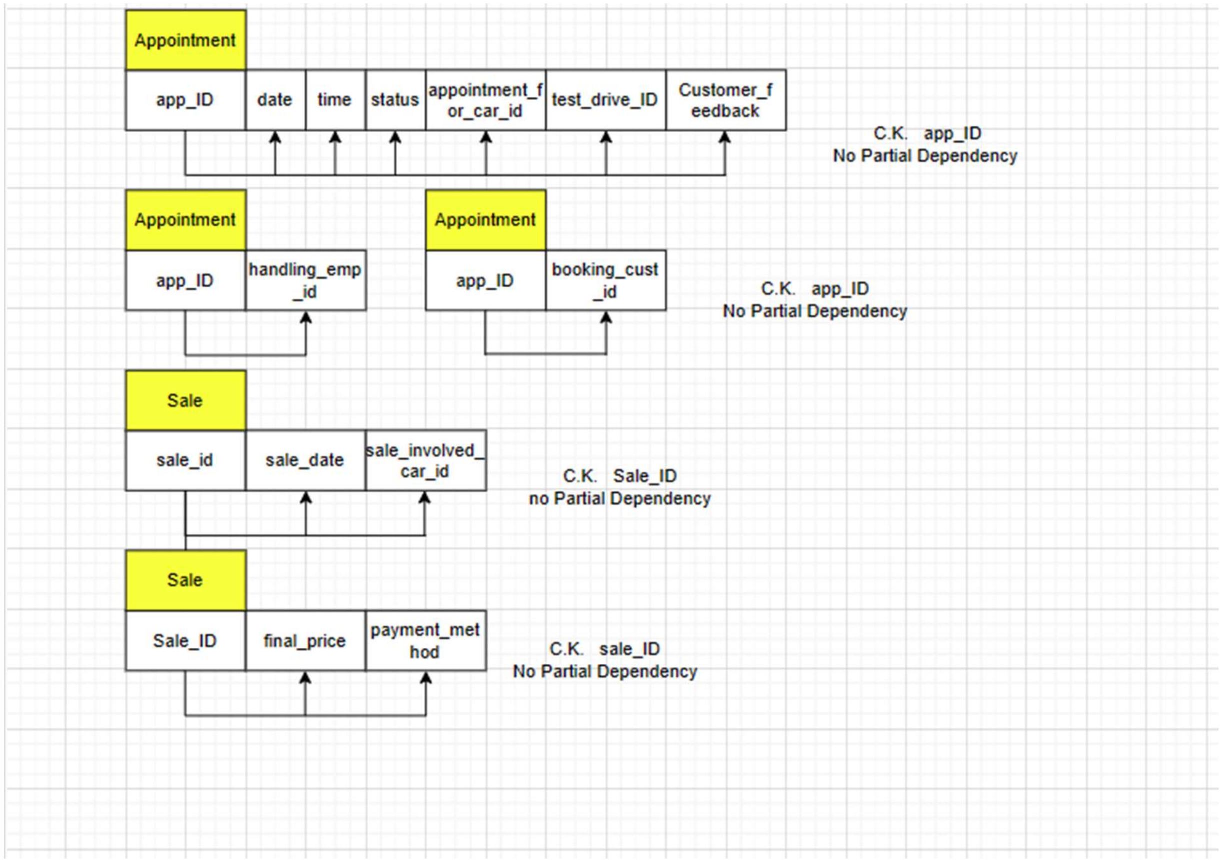


Fig 5.4

3. Third Normal Form (3NF):

Even if we address the partial dependencies in the Appointment and Sale tables to achieve 2NF, the schema might still violate 3NF. This is because some non-key attributes might be dependent on other non-key attributes that are not part of the primary key.

For example, in the Sale table, the payment method might be dependent on the final price (not part of the primary key). This is a transitive dependency that violates 3NF.

Recommendations:

To achieve higher normal forms, we can decompose the tables with partial dependencies:

- **Appointment:** We can create separate tables for Employee_Appointment (with emp_ID and app_ID as primary key) and Customer_Appointment (with customer_ID and app_ID as primary key) to eliminate the partial dependency between employee, customer, and appointment.

- **Sale:** We can create a separate table for Sale_Details (with sale_ID as foreign key and attributes like final_price, payment_method) to eliminate the transitive dependency between final price and payment method.

By decomposing these tables, we can achieve a higher level of data normalization, reducing redundancy and improving data integrity within the showroom management system schema.

Chapter 6

NOSQL

This section details the utilization of a NoSQL database, specifically MongoDB, to address customer identification and media management requirements within the car showroom management system website project. The following will explore the rationale behind this selection, along with the implementation specifics for both customer QR code storage and car image and video retrieval.

Reason for choosing MongoDB:

- **Schema Flexibility:** Images, videos, and QR codes are all binary data types that can vary greatly in size. MySQL, a relational database, enforces a strict schema which can be cumbersome for storing these elements. MongoDB, a NoSQL database, offers schema flexibility allowing you to store data with varying structures without needing to predefine the schema. This makes it easier to accommodate future changes in data formats for these media files.

- **Scalability:** A car showroom website could potentially handle a large volume of images and videos. MongoDB's horizontal scaling capabilities allow you to easily add more servers to handle the growing data size. While MySQL can also be scaled, it can become more complex to manage as the data grows.

- **Performance for Specific Queries:** Retrieving specific customer data associated with a QR code might be a more frequent operation than complex queries on the media itself. NoSQL databases like MongoDB can excel at these targeted queries due to their indexing mechanisms.

- **Storing Additional Data:** MongoDB allows us to embed this metadata within the same document as the media file, simplifying data retrieval and manipulation.\

System Architecture for NoSQL (MongoDB) in Car Showroom Management System:

This section outlines the system architecture for utilizing MongoDB within the car showroom management system website project.

Components:

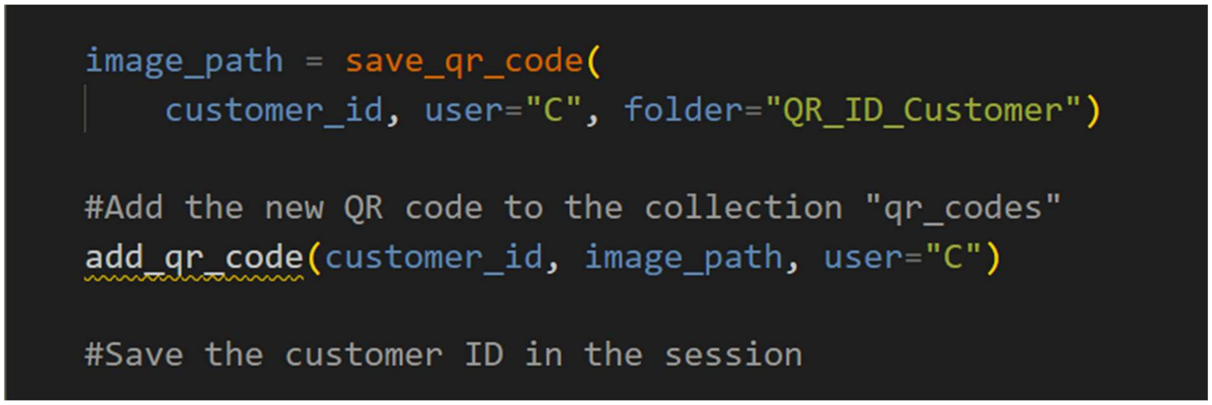
1. **MySQL Database:** This relational database remains the primary data store for core functionalities like customer information (excluding QR codes) and car listings. It manages user accounts, login credentials, and structured car details.

2. **MongoDB Database:** This NoSQL database acts as the secondary data store, specifically designed to handle:

   - **Customer QR Codes:** A dedicated collection within MongoDB stores unique QR codes generated for each customer upon signup. Each document likely includes the customer ID (reference to MySQL) and the associated QR code image data.

   - **Car Media:** Another collection stores car images and videos. Documents within this collection might include the car ID (reference to MySQL), image/video data, and potentially additional metadata like file names, formats, and sizes.

3. **Web Application:** This is the user interface that interacts with both databases. During customer signup, the application generates a QR code using a customer ID retrieved from MySQL. This QR code data is then stored in the designated MongoDB collection.

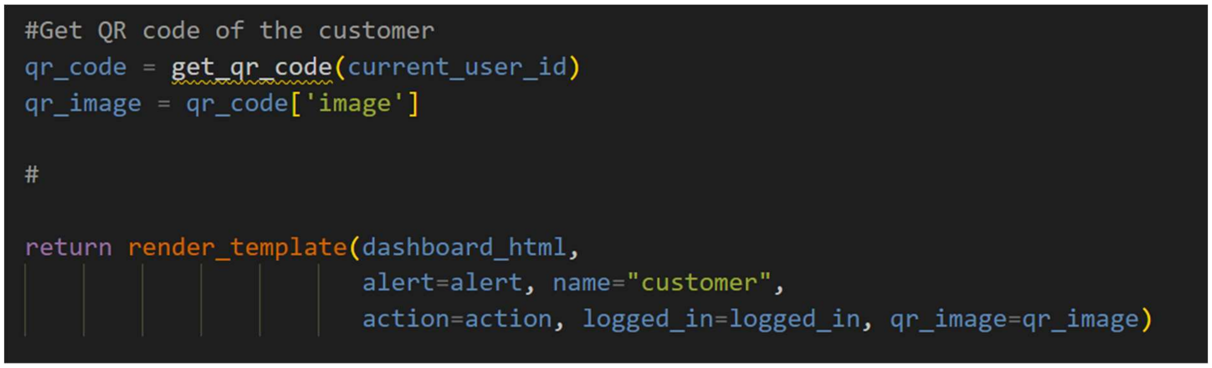


```
image_path = save_qr_code(
    customer_id, user="C", folder="QR_ID_Customer")

#Add the new QR code to the collection "qr_codes"
add_qr_code(customer_id, image_path, user="C")

#Save the customer ID in the session
```

Fig 6.1



```
#Get QR code of the customer
qr_code = get_qr_code(current_user_id)
qr_image = qr_code['image']

#

return render_template(dashboard_html,
                       alert=alert, name="customer",
                       action=action, logged_in=logged_in, qr_image=qr_image)
```

Fig 6.2

Similarly, when a car is added to the system, the web application stores references (car ID) in MySQL and potentially additional details like descriptions. When displaying a specific car's details page, the application retrieves images and video data from MongoDB using the car ID as an index.

Chapter 7

Conclusion & Future Enhancement

Project Conclusion: A Step Forward in Modern Car Showroom Management

This project successfully developed a web-based car showroom management system that leverages a combination of Flask, MySQL, and NoSQL databases. The system offers comprehensive functionalities for managing car listings, customers, test drives, sales, and employee incentives. Prioritizing user experience, the accessible web interface and unique QR code generation for customer identification streamline interactions. Additionally, the integration of two secure payment gateways, Stripe and a blockchain option, provides flexibility and caters to diverse customer preferences.

The implemented review system empowers customers to provide feedback, fostering a customer-centric approach. Furthermore, employee functionalities for managing sales data and tracking incentives promote a results-oriented work environment.

Key Achievements:

- Streamlined showroom operations through efficient data management.
- Enhanced customer experience with a user-friendly interface and diverse payment options.
- Empowered customers through feedback mechanisms.
- Motivated employees with incentive tracking features.
- Incorporated a crypto based payment gateway with gif-based transaction storage to gamify the payment experience on 5irechain and SHM.
- Incorporated a concept of downloadable payment receipts.
- Incorporated a QR code-based customer authentication system.

Future Enhancements:

This project serves as a foundation for a modern car showroom management system. Future enhancements could include:

- Integrating advanced inventory management functionalities.

- Developing in-depth financial reporting capabilities.
- Expanding the system to include direct car service scheduling.
- Expanding the payment options to incorporate more mainstream cryptocurrencies such as bitcoin, Ethereum and worldcoin.
- Including a admin login to provide complete control of the database from the website.

By continuously improving upon the core functionalities and incorporating evolving technologies, this system has the potential to significantly improve efficiency, customer satisfaction, and employee engagement within the car showroom industry.

References

[1] Flask, "Flask Documentation," 3.0.x, [Online].
Available: https://flask.palletsprojects.com/en/3.0.x/.

[2] phpMyAdmin, "phpMyAdmin Documentation," [Online].
Available: https://www.phpmyadmin.net/docs/.

[3] MongoDB, "MongoDB Documentation," [Online].
Available: https://www.mongodb.com/docs/.

[4] PyPI, "qrcode Documentation," [Online].
Available: https://pypi.org/project/qrcode/.

[5] Solidity, "Solidity Documentation," v0.8.24, [Online].
Available: https://docs.soliditylang.org/en/v0.8.24/.

[6] 5ire, "5ire Documentation," [Online].
Available: https://www.5ire.org/documentation.

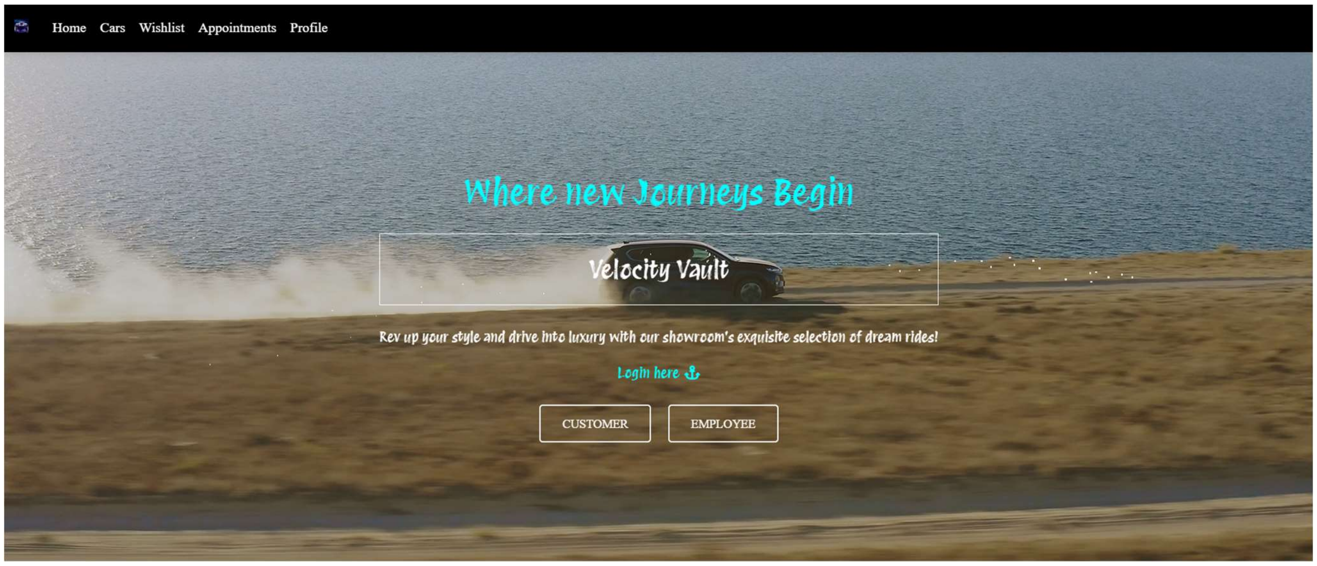
[7] "Car Showroom Management System," [Online].
Available: https://www.scribd.com/document/337853350/car-Showroom-Management-System-doc.

[8] Hardvan, "DBMS-Lab-EL," GitHub. [Online].
Available: https://github.com/Hardvan/DBMS-Lab-EL.

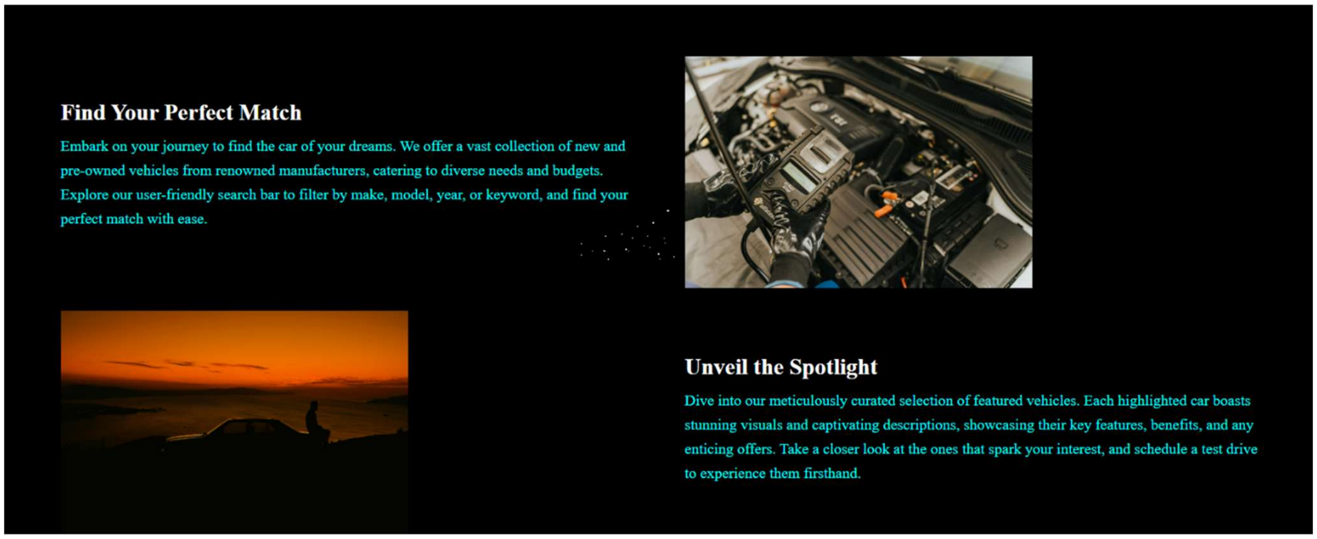
Appendix

Screenshots with descriptions

Dashboard Section: This section represents the landing page of the website. The user can choose to either sign-up/login as a customer or employee and the navbar changes accordingly.



The lower half of the dashboard gives a description of our vision and highlights.



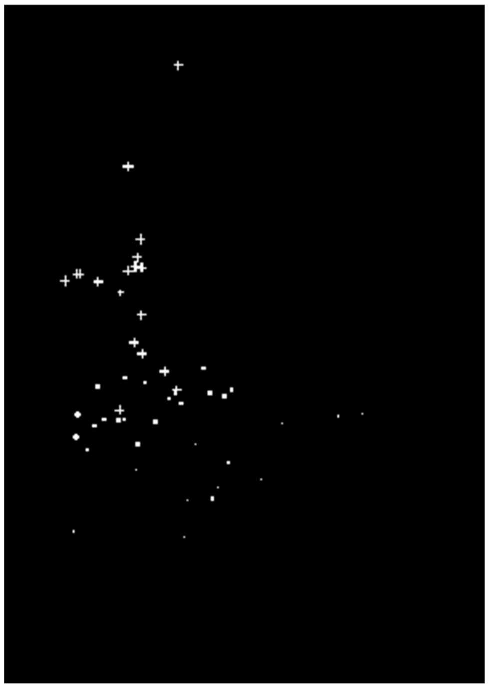
Find Your Perfect Match

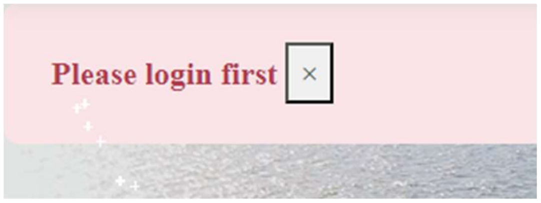
Embark on your journey to find the car of your dreams. We offer a vast collection of new and pre-owned vehicles from renowned manufacturers, catering to diverse needs and budgets. Explore our user-friendly search bar to filter by make, model, year, or keyword, and find your perfect match with ease.

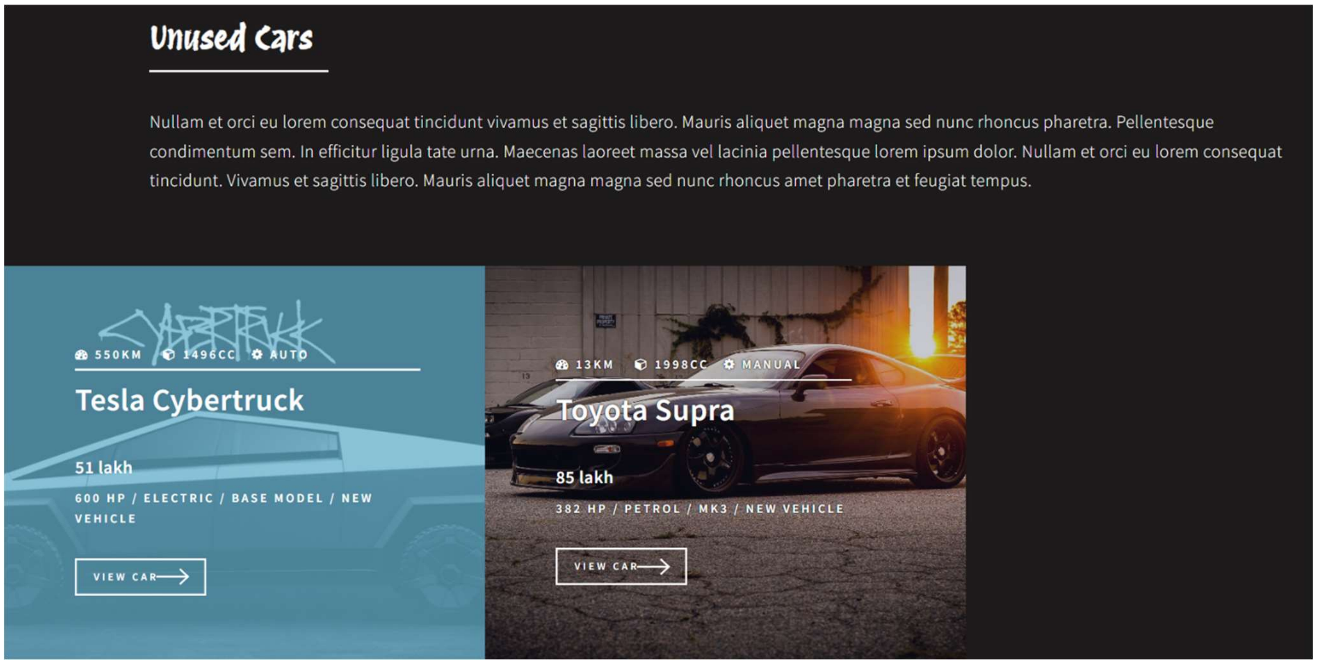
Unveil the Spotlight

Dive into our meticulously curated selection of featured vehicles. Each highlighted car boasts stunning visuals and captivating descriptions, showcasing their key features, benefits, and any enticing offers. Take a closer look at the ones that spark your interest, and schedule a test drive to experience them firsthand.

Innovative Frontend Features: The website uses a JavaScript based mouse graphic feature that drops glittery stars whenever the mouse is moved providing an aesthetic appearance and fun experience to users. In addition, login checks are implemented in the backend which display alerts whenever an unauthorized access is attempted.




Please login first ×


Unused Cars

Nullam et orci eu lorem consequat tincidunt vivamus et sagittis libero. Mauris aliquet magna magna sed nunc rhoncus pharetra. Pellentesque condimentum sem. In efficitur ligula tate urna. Maecenas laoreet massa vel lacinia pellentesque lorem ipsum dolor. Nullam et orci eu lorem consequat tincidunt. Vivamus et sagittis libero. Mauris aliquet magna magna sed nunc rhoncus amet pharetra et feugiat tempus.

550KM 1496CC AUTO

Tesla Cybertruck

51 lakh

600 HP / ELECTRIC / BASE MODEL / NEW VEHICLE

VIEW CAR

13KM 1998CC MANUAL

Toyota Supra

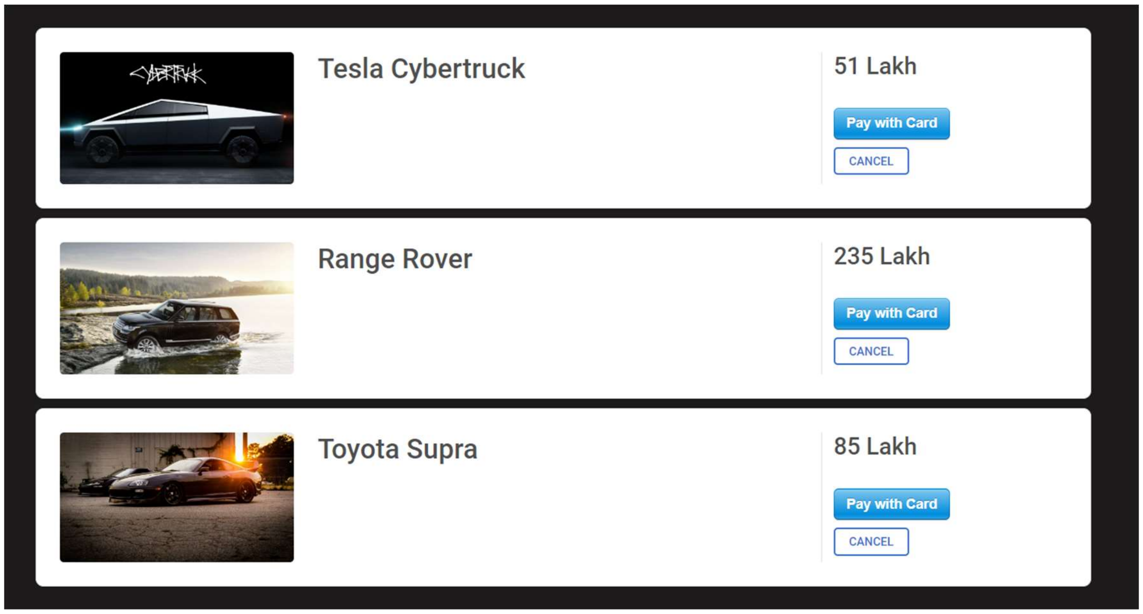
85 lakh

382 HP / PETROL / MK3 / NEW VEHICLE

VIEW CAR

Cars Section: This section contains well designed card representations of the cars available in the inventory in the form shown below. More information about a specific car can be accessed by clicking the view car button.

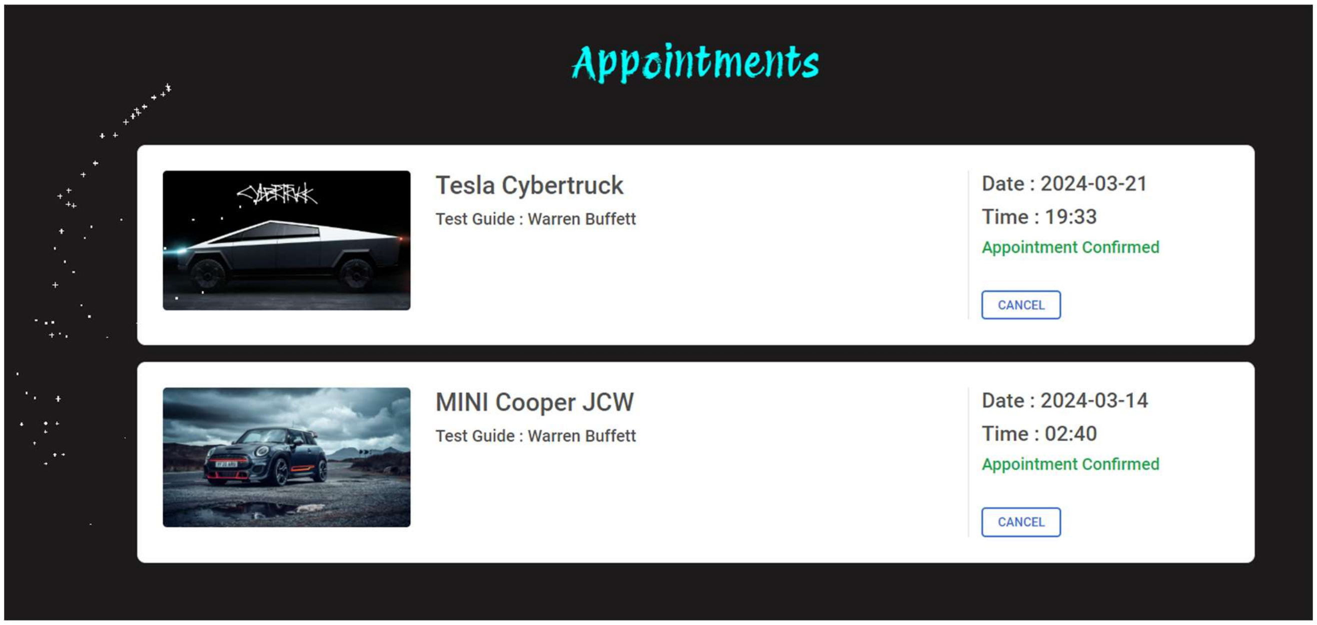
Wishlist Section: If a user likes a car, he/she can add it to the Wishlist section. The user can later access his Wishlist from the navbar to choose to buy/cancel the car.


Tesla Cybertruck
51 Lakh

Range Rover
235 Lakh

Toyota Supra
85 Lakh

Car Details Section: This section displays the full details of the chosen car from the engine type, mileage, fuel type and so on. It also allows a customer to book an appointment at a chosen date.Once

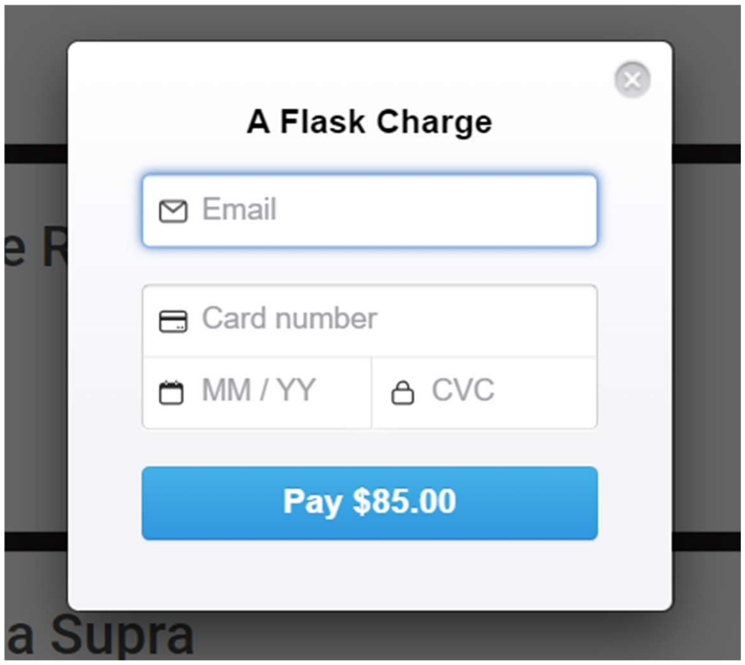

Appointments

Tesla Cybertruck
Test Guide : Warren Buffett
Date : 2024-03-21
Time : 19:33
Appointment Confirmed

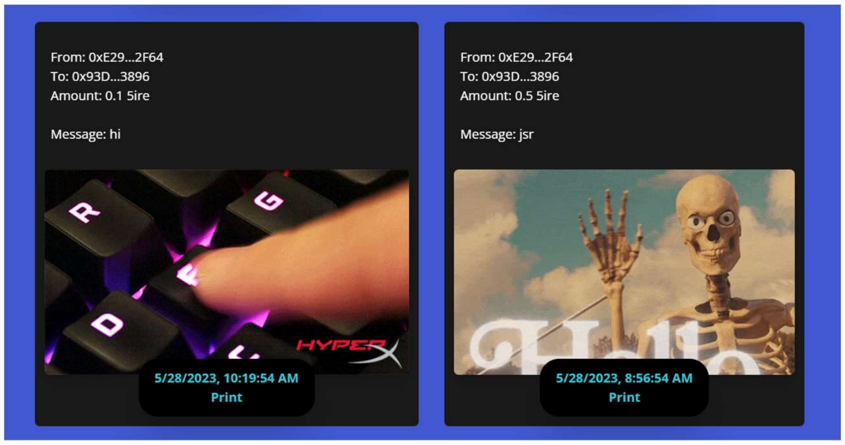
MINI Cooper JCW
Test Guide : Warren Buffett
Date : 2024-03-14
Time : 02:40
Appointment Confirmed

an appointment is booked, it can be viewed by the customer in the appointments section with a employee being allotted to the customer based on our randomizing function in the backend.

Payment by Stripe: Stripe API is used to accept card payments for any product on the website.



Payment in crypto: A gamified crypto payment gateway with immutable receipt generation for each transaction is integrated for an innovative feature in the project.


From: 0xE29...2F64
To: 0x93D...3896
Amount: 0.1 5ire

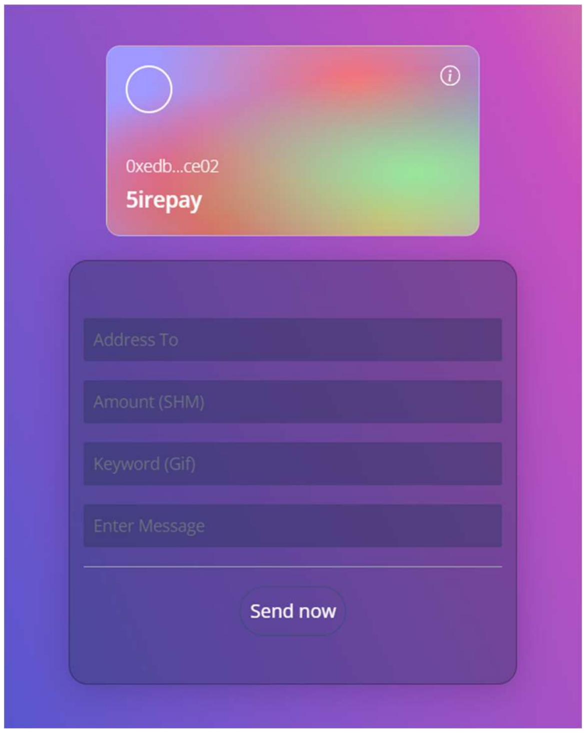
Message: hi

5/28/2023, 10:19:54 AM
Print

From: 0xE29...2F64
To: 0x93D...3896
Amount: 0.5 5ire

Message: jsr

5/28/2023, 8:56:54 AM
Print


0xedb...ce02
5irepay
Address To
Amount (SHM)
Keyword (Gif)
Enter Message
Send now