Software Requirements Specification

for

Car Rental Management System

Prepared by Group #1

Dept. of Computer Science & Engineering

University of Rajshahi

14 April 2023

Group #1 members:

- Ahnaf Shahrear Khan 1910576101
- Mst. Mhamuda Khatun 1912076104
- Toufiqul Islam 1910876107
- Md Nayem Molla 1910476108
- Tareq Munawer Siddiqui 1911176114
- Md. Forhan Shahriar Fahim 1910476120
- Ashiq Uddin Pranto 1911176136
- Md Mamun Miah 1910476148
- Sk. Solaiman Abdullah 1910976149
- Moshiur Rahman 1910476151
- Adrita Alam 1912276153
- Selim Reja 1810676112
- Md. Hasibul Alam 1810576148

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of the project is to make a Car Rental Management System that will automate the processes involved in managing rental cars including car listing, car inspections, reservation, booking management, rent agreements, rent collection, and maintenance requests.

This project is a prototype for the Car Rental Management System which is intended to benefit car owners and car renters. The project has been implemented under the guidance of a university professor.

1.2 Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents.

1.3 Document Abbreviations

This document uses the following conventions.

SRS	System Requirement Specification
CRMS	Car Rental Management System
DB	Database
DDB	Distributed Database
ER	Entity Relationship
API	Application Programming Interface

1.4 Project Scope

The scope of the Car Rental Management System (CRMS) involves the development and implementation of a software solution aimed at simplifying the car renting process and providing a user-friendly interface for both car owners and car renters. The system is designed to operate using a relational database, with car and car owner information provided by the owners, and car renter information provided by the renters. The system is managed by administrators who oversee its functionality and maintenance. The key components within the project scope include:

1. Car Listing:

- The CRMS enables car owners to list their vehicles for rent by providing relevant information such as car make, model, year, specifications, rental rates, and availability.
- Car owners can update and manage their car listings, including adjusting availability dates and pricing information.

2. Car Rental Management:

- The system provides car renters with a user-friendly interface to search and browse available rental cars based on their preferences, such as location, car type, rental duration, and rental rates.
- Car renters can view detailed information about each car listing, including photos, descriptions, rental terms, and any additional charges or requirements.
- Renters can make reservations or bookings for selected cars, specifying the rental duration and communicating with car owners through the system.

3. Rental Agreements and Documentation:

- The CRMS supports the generation and management of digital rental agreements between car owners and renters.
- Car owners can create standard rental agreements that outline terms and conditions, insurance coverage, fuel policy, and other important details.
- The system allows renters to review and electronically sign rental agreements, which are securely stored for future reference.

4. Rental Payment:

- The system facilitates rental payment transactions by providing online payment options for renters to submit rental fees securely.
- Car owners can track rental payments received and manage payment schedules and reminders.
- The CRMS generates reports and notifications to keep car owners informed about rental payments and outstanding balances.

5. Maintenance and Servicing:

- Car owners can provide information about their cars' maintenance history and schedule upcoming servicing tasks.
- Renters can report any issues or maintenance requirements through the system, allowing owners to address them promptly.
- The system facilitates communication between owners and renters regarding maintenance requests and updates on the resolution process.

6. Administrative Functions:

- Administrators have privileged access to the CRMS, allowing them to manage system settings, user accounts, and overall system configuration.
- They can review and moderate car listings, and rental agreements, and resolve any disputes or issues that may arise between owners and renters.
- The system provides administrative tools for monitoring and generating reports on various aspects, such as rental activity, financial transactions, and system usage statistics.

It's important to note that the CRMS prototype focuses on automating and optimizing the car rental process. The primary objective is to create a convenient and user-friendly platform for car owners and car renters, using a relational database to store and manage car and user information. The system is designed to ease the car renting system and enhance the experience for all parties involved within the specified scope.

1.5 References

Websites Links:

1. IEEE Recommended Practice for Software Requirements Specification (IEEE-STD-830- 1998). Available at https://personal.utdallas.edu/~chung/RE/IEEE830-1993.pdf
2. A study from the Geeks for Geeks website. https://www.geeksforgeeks.org/software-requirement-specification-srs-format/
1. A further study from JavaTpoint https://www.javatpoint.com/software-requirement-specifications
2. Sample SRS study from https://gephi.org/users/gephi_srs_document.pdf

2. Overall Description

The Car Rental Management System (CRMS) is a comprehensive platform that provides various functionalities and information related to car rentals. It serves as a centralized database for storing and retrieving data regarding cars, car owners, car renters, drivers, and payment details.

2.1 Product Perspective

The CRMS offers an extensive range of information, ensuring a seamless car rental experience for users. It encompasses the following key aspects:

1. **System Boundaries:**

   The Car Rental Management System (CRMS) is a standalone web-based and mobile application that operates within a specific scope defined by its functional and non-functional requirements. It provides car rental companies and customers with a centralized platform to efficiently manage car rentals and payments.

2. External Interfaces:

User Interfaces: The CRMS offers user interfaces for car rental companies and customers to interact with the system and perform their respective tasks. These interfaces should be intuitive, user-friendly, and accessible from different devices.

Integration with External Systems: The CRMS may need to integrate with external systems such as payment gateways, GPS tracking systems, or online booking platforms. Integration should enable seamless data exchange and enhance functionality.

3. Dependencies:

Data Sources: The CRMS relies on various data sources, including car inventory, customer information, and financial data. These data sources can be external systems or entered directly into the CRMS.

Third-Party Services: The CRMS may depend on third-party services, such as payment processors or insurance providers, to provide specific functionalities. Integration with these services should be considered during development.

4. System Relationships:

Car Rental Ecosystem: The CRMS interacts with multiple stakeholders involved in car rental operations, including car rental companies and customers. It serves as a central platform for communication, reservation management, and information sharing.

Reporting and Analytics: The CRMS may provide reporting and analytics features to generate insights on rental performance, financial metrics, and customer-related data. These reports help car rental companies make informed decisions.

5. System Updates and Maintenance:

Versioning and Upgrades: The CRMS may undergo version updates or upgrades to introduce new features, enhance security, or address software issues.

Support and Maintenance: Ongoing support and maintenance activities are required to ensure the CRMS's reliability, security, and compatibility with evolving technologies.

The CRMS provides the following information:

- **Car Information**

  Types of car: The system provides a detailed list of car types available for rental, including sedans, SUVs, hatchbacks, and more.
  Availability: Users can easily check the availability of cars for specific dates and times, ensuring convenience and flexibility in their rental choices.
  Rent information (pricing): The system displays the rental prices for each car type, allowing users to compare rates and make informed decisions.
  Other information: It provides essential details about each car, such as car papers, location, number of seats, photos or videos showcasing the vehicle's condition, size specifications, and additional features.

- **Car renters' information**

  Personal information: The system records the personal details of car renters, including their name, profession, and contact information (cell phone number, email address).

  Photo, driving license, and national ID card: Car renters are required to upload their photo, driving license, and national ID card, verifying their identity and eligibility to rent a car.

  Ratings and feedback: Car owners can rate and provide feedback on the car renters based on their behavior and adherence to rental

policies, enabling transparency and trust-building within the community.

- **Drivers' information**

  Personal information: The CRMS stores essential details of drivers, including their name and contact information (cell phone number, email address).

  Photo, driving license, and national ID card: Drivers are required to provide their photo, driving license, and national ID card for authentication purposes.

  Ratings and feedback: Both car owners and renters can rate and provide feedback on drivers based on their professionalism, driving skills, and punctuality, helping users make informed decisions when hiring a driver.

- **Payment information**

  Payment details: The system securely stores payment information, including the amount, date-time of the transaction, payment method used, and relevant information about the recipients and senders involved.

  Due information: Users can access details about upcoming payment dues, ensuring timely payments and avoiding any inconvenience.

  Payment history: The CRMS maintains a comprehensive record of past transactions, allowing users to review their payment history and track their financial interactions.

By providing a comprehensive set of features and information, the CRMS simplifies the car rental process, enhances transparency, and promotes trust and efficiency within the car rental community.

2.2 Product Functions

The Car Rental Management System (CRMS) offers a range of features that contribute to its seamless functionality. These features enhance the user experience and streamline the car rental process. Here are the expanded details for each system feature:

User Registration and Authentication:

- User registration: The CRMS provides a user-friendly registration process, allowing individuals to create their accounts by providing necessary information such as name, email address, and contact details.
- Account authentication: To ensure the security and authenticity of users, the system incorporates authentication mechanisms such as email verification or mobile number verification.

Car Listing and Search:

- Car listing: Car owners can easily add their vehicles to the CRMS by providing detailed information about the car, including make, model, year, mileage, and any additional features. They can also upload high-quality images or videos to showcase the car's condition.
- Car search: Renters can search for available cars based on their preferred location, rental dates, car type, and other specific requirements. The system then presents a list of relevant cars with detailed information and pricing.

Booking and Reservation Management:

- Booking process: Renters can initiate the booking process by selecting a car and providing their desired rental dates. The system validates the availability of the chosen car and confirms the booking if the car is available.
- Reservation management: The CRMS allows users to view, modify, or cancel their existing reservations. It provides a clear overview of upcoming and past reservations, ensuring efficient management of rental schedules.

Ratings and Reviews:

- Rating system: The system enables users to rate their experience with car owners, renters, and drivers. This feedback contributes to the overall reputation and trustworthiness of each party within the community.
- Review system: Users can provide detailed reviews sharing their experiences and insights regarding specific cars, owners, renters, or drivers. These reviews help other users make informed decisions and maintain a high level of service quality.

Payment and Invoicing:

- Secure payment gateway: The CRMS integrates with reliable and secure payment gateways, allowing users to make payments using various methods such as credit/debit cards, digital wallets, or bank transfers.
- Automated invoicing: The system generates detailed invoices for each transaction, providing users with a clear breakdown of rental costs, additional charges (if any), and payment receipts.

Notifications and Reminders:

- Real-time notifications: Users receive notifications about booking confirmations, payment reminders, changes in reservation status, and other important updates via email, SMS, or in-app notifications.
- Reminders: The system sends reminders to users regarding upcoming reservations, due payments, or any other relevant information to ensure a smooth rental experience.

Customer Support and Assistance:

- Helpdesk and support: The CRMS offers a dedicated customer support system to address any queries, concerns, or technical issues faced by users. This support can be provided through email, live chat, or a helpline.
- Knowledgebase and FAQs: The system provides a comprehensive knowledge base and frequently asked questions section, offering users self-help resources and answers to common inquiries.

These system features work harmoniously to create a user-friendly and efficient car rental platform. They facilitate smooth communication, transparent transactions, and a high level of convenience for both car owners and renters, ensuring a positive experience within the CRMS ecosystem.

2.3 User Classes and Characteristics

The Car Rental Management System (CRMS) caters to different user roles, including car owners, car renters, and administrators. Each user category has specific functionalities and management capabilities within the system. Here are the details for each user role:

The Car owners should be able to do the following functions:

- **Create and manage accounts:** Car owners can create their accounts by providing necessary personal information and creating a username and password. They can manage their account settings, including updating contact information and profile details.
- **Login with username and password:** Car owners can securely access their accounts by entering their registered username and password.
- **Add car, manage, and update car information:** Car owners can list their cars by providing detailed information such as make, model, year, mileage, and features. They can update the car details whenever required, including availability, pricing, and uploading images or videos.
- **Manage payments:** Car owners can track and manage the payment transactions associated with their rented cars. They can view payment history, monitor pending payments, and receive notifications about successful payments.
- **Send notifications to car renters:** Car owners can communicate with car renters through the CRMS, sending notifications regarding booking confirmations, changes in availability, or any other relevant updates.

- **Give ratings and feedback:** Car owners have the ability to rate and provide feedback on the car renters based on their experience, helping build a reliable and trustworthy community.

Car Renters should have the following functions:

- **Create and manage an account:** Car renters can register with the CRMS by providing personal information and creating a username and password. They can manage their account settings, update contact details, and customize their preferences.
- **Login with username and password:** Car renters can securely log in to their accounts using their registered username and password.
- **Search and view cars:** Car renters can search for available cars based on their preferred location, rental dates, and specific requirements. They can view detailed car listings, including information about the car, pricing, availability, and images.
- **Car booking and car rental agreements:** Car renters can initiate the booking process by selecting a car and providing their desired rental dates. They can review and agree to the rental terms and conditions before confirming the booking.
- **Make payment and view payment history:** Car renters can make secure payments for their bookings using various payment methods supported by the CRMS. They can view their payment history and access receipts for their transactions.
- **Booking cancellation:** In case of any changes or cancellations to their bookings, car renters can utilize the system's functionality to cancel their reservations according to the specified cancellation policy.
- **Give ratings and feedback:** Car renters have the ability to rate and provide feedback on the car owners and the rental experience, helping other users make informed decisions.

The Administrator should have the following management functions:

- **Create and manage accounts:** Administrators can create accounts for other administrators and manage their permissions and access levels.
- **Login with username and password:** Administrators can securely log in to the system using their authorized username and password.

- **Update privacy policy and terms & conditions:** Administrators have the authority to update and maintain the privacy policy and terms & conditions of the CRMS, ensuring compliance with legal and regulatory requirements.
- **Manage user accounts:** Administrators can manage user accounts within the system, including suspending or temporarily blocking user accounts in case of policy violations or other administrative needs.

By assigning specific management functions to each user category, the CRMS ensures smooth operations and efficient control over car listings, bookings, payments, notifications, and user interactions. This approach streamlines the user experience and facilitates effective management of the car rental ecosystem.

2.4 Operating Environment

The Car Rental Management System (CRMS) operates in a client-server system, utilizing various operating systems, databases, and tools. Here are the details for each component of the operating environment:

- Client-Server system
- Operating Systems:
    - Android
    - iOS
    - MacOS
    - Mac OS X
    - Windows XP
    - Windows 7
    - Windows 10
    - Windows 11
- Database:
    - Structured Query Language(SQL)
    - MariaDB for Relational Database
- Languages and Frameworks:
    - Flutter and Dart for Front-end
    - Nodejs for Back-end

2.5 Design and Implementation Constraints

The development and implementation of the Car Rental Management System (CRMS) are subject to various constraints and considerations. Here are the expanded details for each design and implementation constraint:

1. **Technology Stack**: The CRMS will be developed using specific technologies and frameworks, including Node.js for server-side development, MariaDB for the database management system, and Flutter with Dart for mobile application development. These technology choices provide a robust and efficient foundation for the system's functionality and ensure compatibility across different platforms.

2. **Centralized Database**: To ensure data integrity and consistency, the CRMS will implement a centralized database management system. This approach allows for seamless data storage, retrieval, and management, facilitating efficient communication between different components of the system.

3. **Development Timeline**: There is a specific time constraint for the development of the CRMS, with a target completion date of 12 April 2023. This timeline serves as a milestone for the development team to ensure timely delivery of the system.

4. **Resource Requirements**: The development of the CRMS may require certain resources such as servers, software tools, and hardware. Adequate server infrastructure must be in place to host the system, and software tools should be accessible to the development team. Additionally, hardware requirements should be considered to ensure optimal performance and compatibility.

5. **Security and Data Protection**: The CRMS must prioritize the security and protection of sensitive data, such as tenant and car owner information. It should implement robust security measures, including encryption of sensitive data, secure authentication mechanisms, and adherence to industry best practices to prevent unauthorized access and data breaches.

6. **Scalability**: The system should be designed to handle a large number of users and properties. It should be scalable to accommodate growth and increasing demands without compromising performance. This can be achieved through efficient database design, optimization of server resources, and implementation of scalable architecture patterns.

7. **Compliance with Laws and Regulations**: The CRMS should comply with relevant laws and regulations related to rental management, data privacy, and user protection. It should adhere to local, national, and international regulations to ensure legal and ethical operation within the jurisdictions it serves.

By considering these design and implementation constraints, the development team can create a secure, scalable, and compliant Car Rental Management System that meets the specified timeline and resource requirements.

2.6 Assumptions and Dependencies

The successful implementation and usage of the Car Rental Management System (CRMS) rely on certain assumptions and dependencies. Here are the expanded details for the assumptions and dependencies:

1. **Internet Access**: It is assumed that users of the CRMS will have reliable access to the Internet. As the system operates online, users need to connect to the internet to access the CRMS website or mobile application. Continuous internet connectivity is necessary for seamless interaction with the system, including browsing available cars, making bookings, and managing accounts.

2. **User Technical Proficiency**: The CRMS assumes that users will possess a basic level of technical proficiency and familiarity with using web-based applications or mobile applications. Users should be comfortable navigating through interfaces, filling out forms, and following instructions provided by the system. While the CRMS may

provide user-friendly interfaces, some level of digital literacy is expected from the users.

3. **Hardware and Software Compatibility**: The proper functioning of the CRMS depends on the compatibility of users' devices with the required hardware and software. Users should have devices (computers, smartphones, or tablets) that meet the minimum system requirements for accessing the CRMS. Additionally, users must have compatible operating systems (such as Windows, Android, iOS, or MacOS) and up-to-date web browsers or mobile application versions to ensure optimal performance.

4. **Availability of External Services**: The CRMS may depend on external services or APIs (Application Programming Interfaces) for certain functionalities. This includes services for payment processing, map integration, and notifications. The system assumes the availability and proper functioning of these external services to deliver the intended features seamlessly. Any disruptions or limitations in these external services may impact the overall performance and user experience of the CRMS.

5. **User Data Accuracy and Integrity**: The CRMS assumes that the information provided by users, such as personal details, payment information, and rental agreements, is accurate and valid. Users are responsible for providing correct and up-to-date information when registering, making bookings, or managing their accounts. The system relies on the integrity and accuracy of user data to ensure smooth operations and accurate processing of transactions.

While these assumptions and dependencies form the basis for the CRMS design and functionality, efforts should be made to provide clear instructions, user-friendly interfaces, and robust error handling mechanisms within the system. User support and assistance channels can also be established to address any technical difficulties or challenges that users may encounter during their interaction with the CRMS.

3. Functional Requirements

3.1 System Features

- **Car Listings**

  The CRMS shall allow car owners to create and manage car listings. Each listing shall include the following details:

  1. **Car information:** Car model, type, size, no of seats, registration papers, and availability.
  2. **Rental price:** The rental price for the car.
  3. **Description:** Color, AC/Non-AC, condition, mileage.
  4. **Insurance information:** Details about the car's insurance policy such as insurance company name, policy number, and expiration date.

- **Car Owner Info**

  The CRMS shall allow car owners to create and manage a profile. The profile shall include the following details:

  1. **Personal information:** Name, address, phone number, and any other relevant contact information.
  2. **Cars:** Information about cars and pricing.

- **Car Renters Info**

  The CRMS shall allow car renters to create and manage a profile. The profile shall include the following details:

  1. **Personal information:** Name, address, phone number, and any other relevant contact information.
  2. **Rental preferences:** Location, rental price range, car type, number of seats, and any other relevant search criteria.

- **Rent Management**

  The CRMS may allow car owners to create and manage rents. Each rent must include the following details:

  1. **Rent start and end time:** The start and end time for the rent.
  2. **Rent amount:** The rental amount for the car.

3. **Security deposit amount:** The amount of the security deposit required for the car.
4. **Renters information:** The name and contact information of the renter on the rent.
5. **Rent status:** The status of the rent, such as active or terminated.

- **Payment Processing**

  The CRMS shall allow car renters to make rental payments through the system, and car owners to track payments and issue invoices. The system shall include the following features:

  1. **Payment processing:** The CRMS shall accept rental payments from car renters via credit card, debit card, mobile banking or bank transfer.
  2. **Payment tracking:** The CRMS shall track all rental payments made by the car renter and display the payment history for each rent.
  3. **Invoicing:** Property owners shall be able to generate and send invoices for rental payments due from car renters.

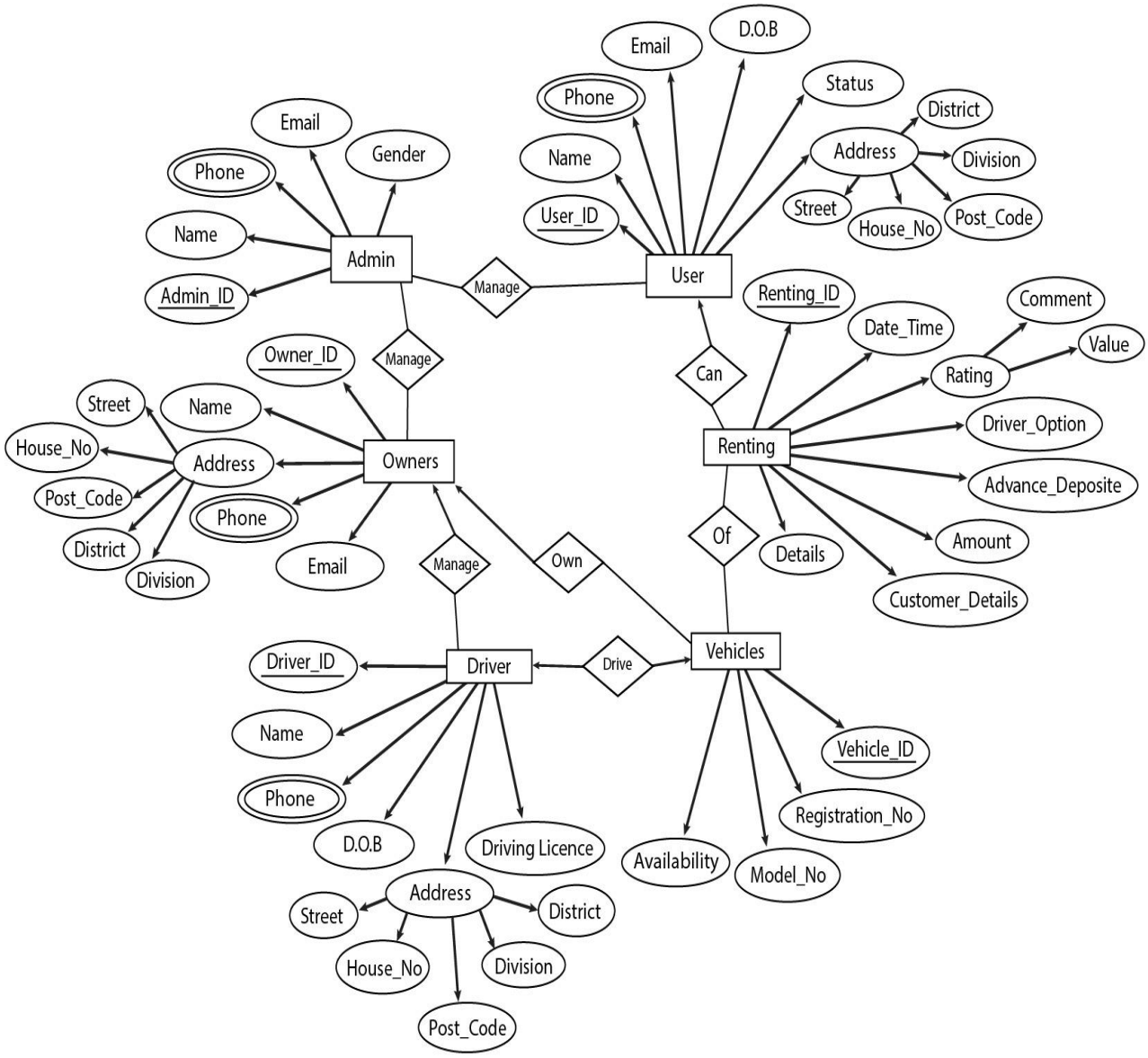
- **Reporting**

  The CRMS shall provide reporting features for car owners to track rental income, expenses, and rental rates. The system shall include the following reports:

  1. **Rental income report:** A report that shows the total rental income earned by the car owner over a specified period.
  2. **Expense report:** A report that shows the total expenses incurred by the car owner for a specified period, including maintenance, repairs, and other costs. It also includes a report of the car renter’s expenses.

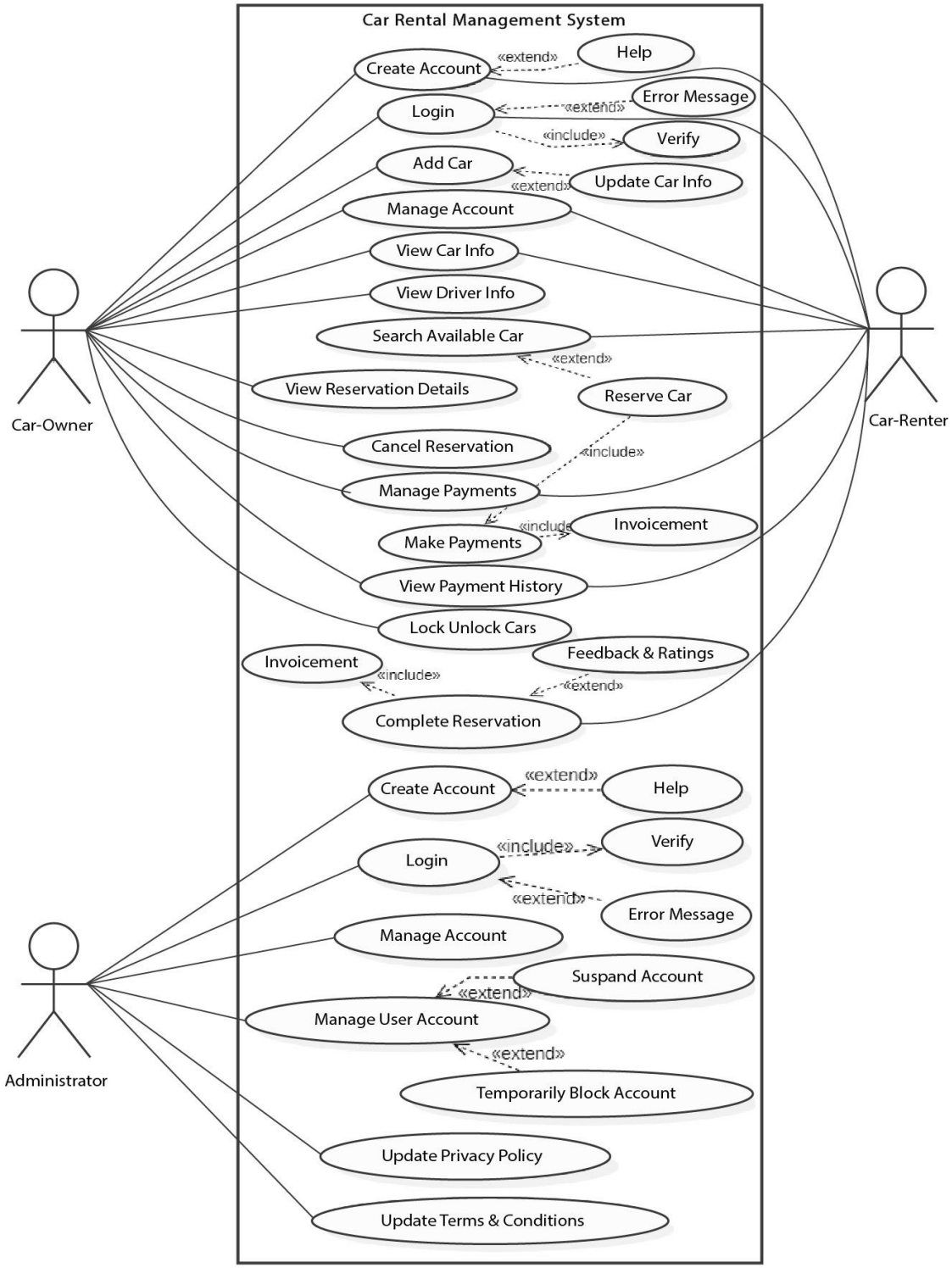
3.2 ER Diagram:

The major features of the Car Rental Management System as shown in the below Entity-Relationship model (ER model)



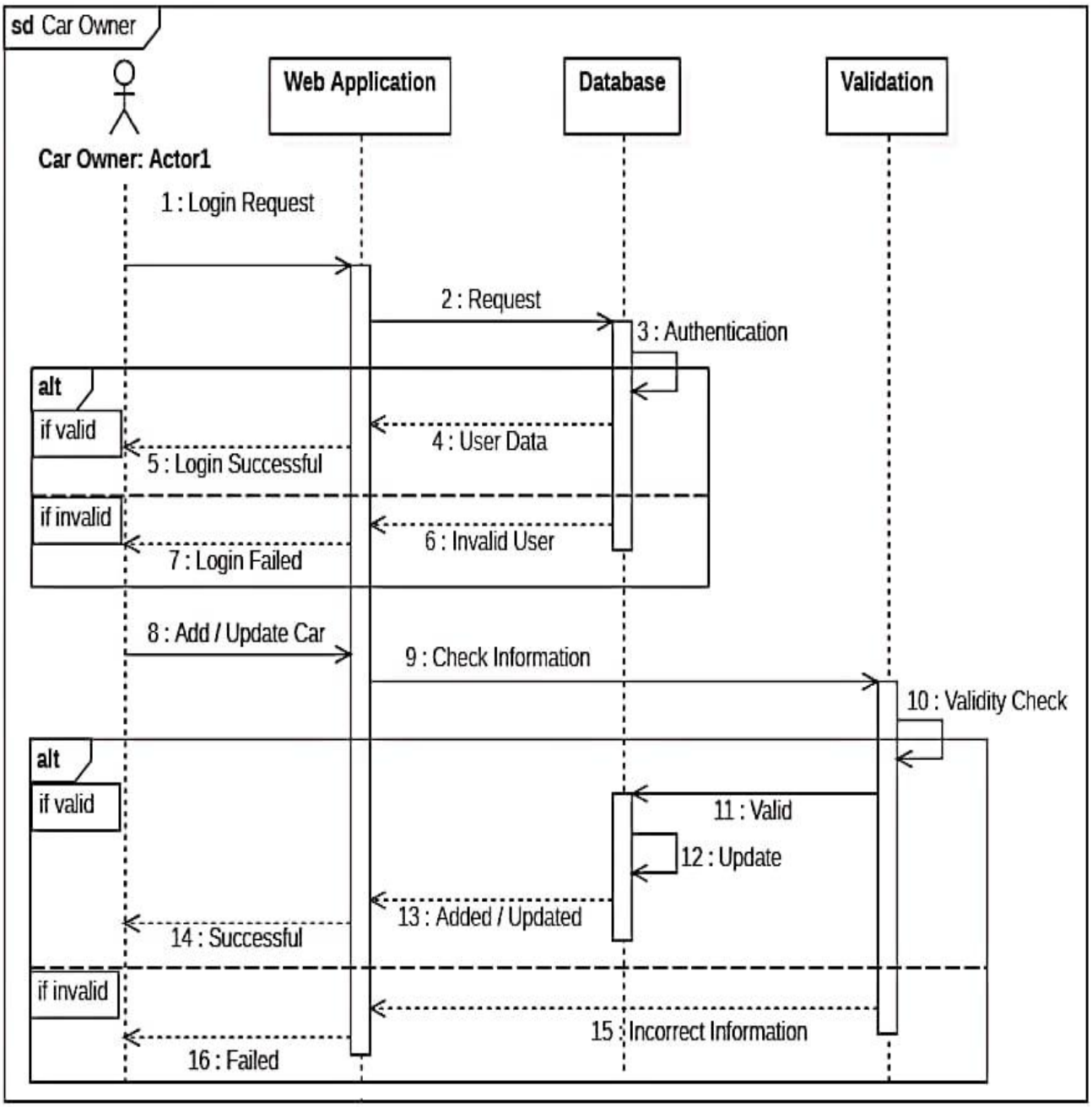
Entity-Relationship(ER) Diagram

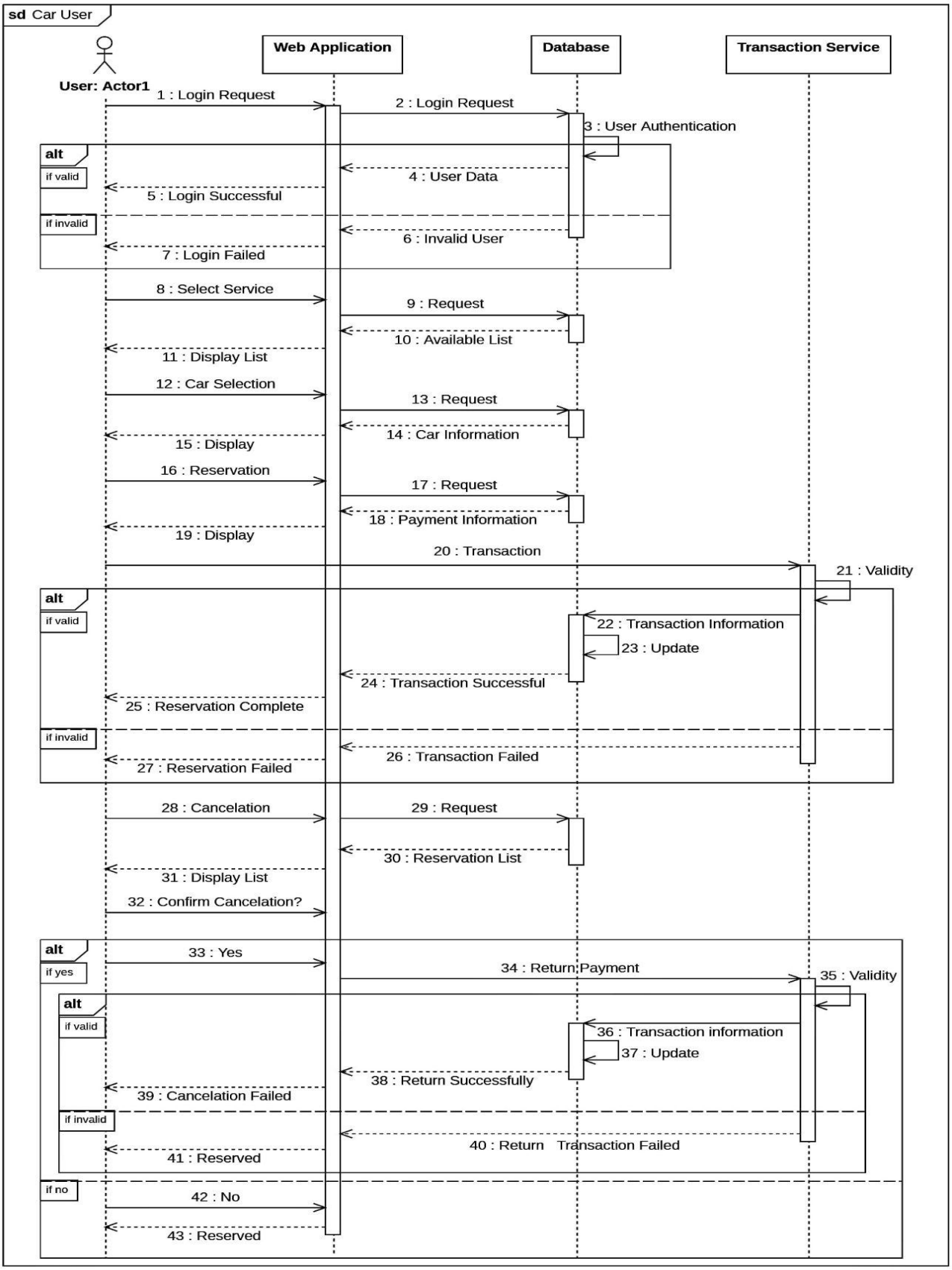
3.3 Use Case Diagram:

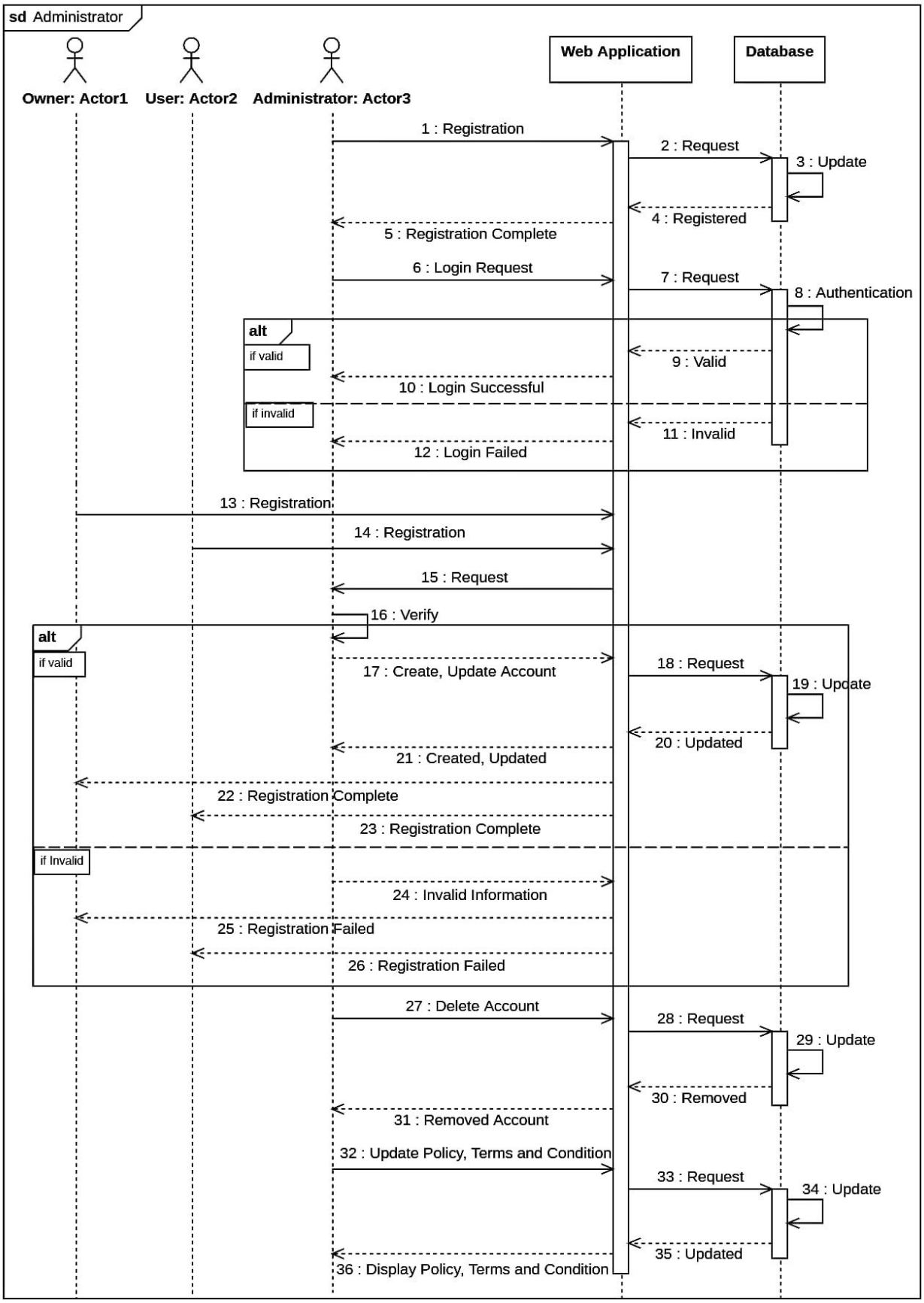


Use Case Diagram

3.4 Sequence Diagram:




sd Car User
User: Actor1
Web Application
Database
Transaction Service
1 : Login Request
2 : Login Request
3 : User Authentication
alt
if valid
4 : User Data
5 : Login Successful
if invalid
6 : Invalid User
7 : Login Failed
8 : Select Service
9 : Request
10 : Available List
11 : Display List
12 : Car Selection
13 : Request
14 : Car Information
15 : Display
16 : Reservation
17 : Request
18 : Payment Information
19 : Display
20 : Transaction
21 : Validity
alt
if valid
22 : Transaction Information
23 : Update
24 : Transaction Successful
25 : Reservation Complete
if invalid
26 : Transaction Failed
27 : Reservation Failed
28 : Cancelation
29 : Request
30 : Reservation List
31 : Display List
32 : Confirm Cancelation?
alt
if yes
33 : Yes
34 : Return Payment
35 : Validity
alt
if valid
36 : Transaction information
37 : Update
38 : Return Successfully
39 : Cancelation Failed
if invalid
40 : Return Transaction Failed
41 : Reserved
if no
42 : No
43 : Reserved


sd Administrator
Owner: Actor1
User: Actor2
Administrator: Actor3
Web Application
Database
1 : Registration
2 : Request
3 : Update
4 : Registered
5 : Registration Complete
6 : Login Request
7 : Request
8 : Authentication
alt
if valid
9 : Valid
10 : Login Successful
if invalid
11 : Invalid
12 : Login Failed
13 : Registration
14 : Registration
15 : Request
16 : Verify
alt
if valid
17 : Create, Update Account
18 : Request
19 : Update
20 : Updated
21 : Created, Updated
22 : Registration Complete
23 : Registration Complete
if Invalid
24 : Invalid Information
25 : Registration Failed
26 : Registration Failed
27 : Delete Account
28 : Request
29 : Update
30 : Removed
31 : Removed Account
32 : Update Policy, Terms and Condition
33 : Request
34 : Update
35 : Updated
36 : Display Policy, Terms and Condition

3.5 Use Case and Data Flow Description

3.5.1 Registration

Use Case ID: USC001

Description: This use case describes the process of registration.

Actor: All kinds of users.

Precondition:

- The actor must have a phone number and email address.
- The actor must input the required information for registration.

Post Condition:

- The system will save the registration information to the database.
- The actor will receive a confirmation email.
- The actor will activate the account by clicking the link in the email.
- The actor will log in to the system.

Normal Flows:

- **Actor:** The actor will input the required information for the registration and click on the "Register" button.
- **System:** The system will validate the registration information and save it to the database. The system will also send a confirmation email to the actor.
- **Actor:** The actor will receive the confirmation email and click on the link in the email to activate their account.
- **System:** The system will verify the activation link and show a success message to the actor. The system will also redirect the actor to the login page.

3.5.2 Login

Use Case ID: USC002

Actor: All kinds of users.

Description: This use case describes the process of login.

Precondition:

- The actor must have the user id and password.

Post Condition:

- The system will check the user id and password from the system database.
- If the user id and password are correct, the system will log in the user to the system according to their user type and show their profile page.
- If the user id and password are incorrect, the system will show an error message.

Normal Flows:

- **Actor:** The actor will enter the user id and password to the system and click on the "Login" button.
- **System:** The system will check the user id and password from the database system. If the user id and password are correct, the system will allow the user to log in and show their home page. If the user id and password are incorrect, the system will show an error message.

3.5.3 View Personal profile

Use Case ID: USC003.5

Description: This use case describes the process of viewing a personal profile.

Actor: Car owner, Car renter

Precondition:

- The actor must be an authorized user and have to log in.
- The actor must have permission to view his personal profile.

Postcondition:

- The system must show the personal profile information.

Normal flow:

- **Actor:** At first actor will press the ‘profile’ button.
- **System:** Then the system will show the personal profile information, such as name, email, phone number, address, rating, reviews, etc.

3.5.4 View Car Info

Use Case ID: USC004

Description: This use case describes the process of viewing car information.

Actor: Car owner, Car renter

Precondition:

- The actor must be an authorized user and have to log in.
- The actor must have permission to view car information.

Postcondition:

- The system must show the car information, such as model, type, color, price, location, availability, rating, reviews, etc.

Normal flow:

- **Actor:** At First actor will press the ‘car info’ button on the car list page or the reservation details page.
- **System:** Then the system will show the car information.

3.5.5 View Driver Info

Use Case ID: USC003.5

Description: This use case describes the process of viewing driver information.

Actor: Car owner, Car renter

Precondition:

- The actor must be an authorized user and have to log in.
- The actor must have permission to view driver information.

Postcondition:

- The system must show the driver's information, such as name, license number, phone number, rating, reviews, etc.

Normal flow:

- **Actor:** At first actor will press the ‘driver info’ button on the car list page or the reservation details page.
- **System:** Then the system will show the driver's information.

3.5.6 Make Payment

Use Case ID: USC006

Description: This use case describes the process of making a payment.

Actor: Car renter

Precondition:

- The actor must be a car renter and have to log in.
- The actor must have reserved a car and confirmed the reservation.
- The actor must have a valid payment method.

Postcondition:

- The system must deduct the payment amount from the actor’s account or card.
- The system must send an invoice to the car owner and the car renter.
- The system must update the payment status of the reservation.

Normal flow:

- **Actor:** At first the actor will press the ‘pay’ button on the reservation details page.
- **System:** Then the system will show the payment amount and ask the actor to choose a payment method, such as credit card, debit card, mobile banking, etc.
- **Actor:** The actor will select a payment method and enter the required information, such as card number, expiry date, CVV, etc.
- **System:** The system will verify the payment information and process the payment. If the payment is successful, the system will show a confirmation message and send an invoice to the car owner and the

car renter via email. If the payment fails, the system will show an error message and ask the actor to try again or choose another payment method.

3.5.7 Search for Available Car

Use Case ID: USC007

Description: This use case describes the process of searching for cars.

Actor: Car Renter, Car Owner

Precondition:

- The actor must be an authorized user and have to log in (except for an unauthorized renter).
- The actor must have permission to search cars (except for unauthorized renters).
- The actor must categorize the car by selecting a size, type, price, location, or color.

Post Condition:

- The system must show the car information.

Normal Flows:

- **Actor:** The actor will input the car information (such as name and type) and click on the "Search" button.
- **System:** The system will show the car details that match the search criteria.

3.5.8 Add Car

Use Case ID: USC008

Description: This use case describes the process of adding a car to the system.

Actor: Car Owner

Precondition:

- The actor must be a car owner and have to log in.
- The actor must have permission to add a car to the system.
- The actor must have the necessary information about the car, such as model, type, color, price, location, etc.

Post Condition:

- The system will save the car information to the database.
- The system will display the car information on the car owner's profile page.

Normal Flows:

- **Actor:** The actor will click on the "Add Car" button on their profile page.
- **System:** The system will show a car form with fields for entering the car information.
- **Actor:** The actor will fill in the car form with the required information and click on the "Submit" button.
- **System:** The system will validate the car information and save it to the database. The system will also show a confirmation message to the actor.
- **System:** The system will display the car information on the car owner's profile page. The system will also make the car available for searching and reservation by other actors.

3.5.9 Payment History

Use Case ID: USC009

Description: This use case describes the process of seeing payment history.

Actor: Car Renter, Car Owner

Precondition:

- The actor must have to log in.
- The actor must have permission to see history.

Post Condition:

- The system must show the payment history information.

Normal Flows:

- **Actor:** The actor will click on the "History" button on their profile page.
- **System:** The system will show the payment history information of the actor.

3.5.10 Reserve a Car

Use Case ID: USC010

Description: This use case describes the process of reservation.

Actor: Car Renter

Precondition:

- The actor must be a car renter and have to log in.
- The car must be available.
- The actor must select a car and make payment for the reservation.

Post Condition:

- The system will save the reservation information to the database.
- The system will send an invoice to the car owner.
- The system will show a confirmation message to the customer. The system also provides a cancel reservation method.

Normal Flows:

- Actor: The actor will select a car and click on the "Reserve" button.
- System: The system will show the payment gateway page.
- **Actor:** The actor will perform all the payment-related tasks and click on the "Confirm" button.

- **System:** The system will validate the payment information and save it to the database. The system will also send an invoice to the car owner and a confirmation message to the customer.

3.5.11 Cancel Reservation

Use Case ID: USC011

Description: This use case describes the flows of cancellation of a reservation.

Actors: Car Renter, Car Owner.

Precondition:

- Actors must be authorized users and have to log in to the system.
- Actors must have permission to cancel a reservation.
- Actors must have a valid reservation id for cancellation.

Post Condition:

- The system will update the information in the database and show a confirmation message. The system will also send a cancel reservation message to the other actor.

Normal Flows:

- **Actor:** The actor will input the car name and corresponding reservation id for cancellation and click on the “Cancel” button.
- **System:** The system will check the reservation id and update the information in the database. The system will also show a confirmation message to the actor and send a cancel reservation message to the other actor.

3.5.12 Feedback and Ratings

Use Case ID: USC012

Description: This use case describes the process of giving and receiving feedback and ratings.

Actor: Car Renter

Precondition:

- The actor must be a car renter and have to log in.
- The actor must have to complete a reservation.
- The actor must have permission to give and receive feedback and ratings.

Post Condition:

- The system will save the feedback and ratings information to the database.
- The system will display the feedback and ratings information on the profile page of the car owner.

Normal Flows:

- **Actor:** The actor will select a completed reservation transaction and click on the "Give Feedback" button.
- **System:** The system will show a feedback form with a rating scale and a comment box.
- **Actor:** The actor will rate the other actor on a scale of 1 to 3.5 stars and write a comment about their experience.
- **System:** The system will validate the feedback and ratings information and save it to the database. The system will also show a confirmation message to the actor.
- **System:** The system will display the feedback and ratings information on the profile page of the actors. The system will also calculate the average rating of each actor based on their feedback history.

3.5.13 Lock and Unlock Cars

Use Case ID: USC013.5

Description: This use case describes the process of locking and unlocking cars.

Actor: Car Owner

Precondition:

- The actor must be a car owner and have to log in.
- The actor must have permission to lock and unlock cars.
- The actor must have access to the car database.

Post Condition:

- The system will update the car status in the database and show a confirmation message.

Normal Flows:

- **Actor:** The actor will select a car from the car database and click on the "Lock" or "Unlock" button.
- **System:** The system will check the car status and perform the action. The system will also update the car status in the database and show a confirmation message to the actor.

3.5.14 Manage User Account

Use Case ID: USC014

Description: This use case describes the process of managing user accounts.

Actor: Administrator

Precondition:

- The actor must be an administrator and have to log in.
- The actor must have permission to manage user accounts.
- The actor must have access to the user database.

Post Condition:

- The system will update the user account information in the database and show a confirmation message.

Normal Flows:

- **Actor:** The actor will select a user account from the user database and click on the "Manage" button.
- **System:** The system will show the user account details and the available actions, such as suspend, block, activate, or delete.
- **Actor:** The actor will choose an action and click the "Confirm" button.
- **System:** The system will perform the action and update the user account information in the database. The system will also show a confirmation message to the actor and send a notification to the affected user.

3.5.15 Update Privacy Policy

Use Case ID: USC013.5

Description: This use case describes the process of updating the privacy policy of the car rental system.

Actor: Administrator

Precondition:

- The actor must be an administrator and have to log in.
- The actor must have permission to update the privacy policy.

Postcondition:

- The system must save the updated privacy policy to the database.
- The system must notify the users about the changes in the privacy policy.

Normal flow:

- **Actor:** At first actor will press the 'privacy policy' button on the admin dashboard page.
- **System:** Then the system will show the current privacy policy and allow the actor to edit it.
- **Actor:** The actor will make the necessary changes to the privacy policy and press the 'save' button.

- **System:** The system will validate the changes and save them to the database. The system will also send an email notification to all the users about the updated privacy policy.

4. External Interface Requirements

4.1 User Interfaces

The CRMS will have a web-based and application-based user interface that is minimal and user-friendly. The interface shall be accessible on desktop and mobile devices.

4.2 Hardware Interfaces

The CRMS will require an internet connection, a device (computer /smartphone), and a web browser or app to access the system.

4.3 Software Interfaces

Software used	Description
Operating system	We have used the most popular operating systems i.e. Windows, Android, iOS, and MacOS which are easily reachable by our user base.
MariaDB	To save information about the car, car owner, car renter, and admins.
Flutter and Dart	To implement the project in an android and web platform we have chosen this language and framework for its more interactive and cross-platform support.
Nodejs	To implement the backend of this project we have chosen Nodejs to create restAPI.

4.4 Communications Interfaces

This project will support all types of web browsers and devices (computers and smartphones).

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The steps involved to perform the implementation of the database are listed below

A. ER Diagram

The ER Diagram constitutes a technique for representing the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalize relation and finally obtain a relation database.

B. Normalization

The basic objective of normalization is to reduce redundancy which means that information is to be stored only once. Storing information several times leads to a wastage of storage space and an increase in the total size of the data stored.

Normalization is the process of breaking down a table into smaller tables. So that each table deals with a single theme. There are three different kinds of modifications of anomalies and formulating the first, second, and third normal forms is considered sufficient for most practical purposes. It should be considered only after a thorough analysis and complete understanding of its implications.

5.2 Safety Requirements

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the

operations of committed transactions from the backed up log, up to the time of failure.

5.3 Security Requirements

Security systems need database storage just like many other applications.

5.4 Technical Constraints

The HRMS shall be developed using Node Js for the back end, Flutter and Dart for the front end, and MariaDB for the database. The system shall be hosted on Amazon Web Services (AWS) using a Linux-based server.

5.5 Business Constraints

The HRMS shall comply with all applicable laws and regulations related to rental property management and payment processing. The system shall be compatible with major web browsers, including Google Chrome, Mozilla Firefox, Safari, etc, and popular Android and iOS versions.

5.6 Software Quality Attributes

- **Availability:** The HRMS should be available 24/7.
- **Correctness:** The system must provide accurate search results.
- **Maintainability:** The administrators should maintain correct schedules for updating the system.
- **Usability:** The system should satisfy a maximum number of customers' needs.
- **Scalability:** The HRMS shall be scalable to handle an increasing number of properties, tenants, and property owners.

6. Glossary

- **Use Case Diagram:** A use case diagram is a way to summarize details of a system and the users within that system.
- **Sequence Diagram:** A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction.

- **ER Diagram:** ER diagram stands for an Entity-Relationship diagram. It is a high-level data model. This model is used to define the data elements and relationships for a specified system.
- **Operating System:** An operating system (OS) is system software that manages computer hardware and software resources, and provides common services for computer programs.
- **SQL:** SQL(Structured Query Language) is a standard language for accessing and manipulating databases.
- **Relational database:** In a relational database, data is organized into tables, which consist of rows and columns. Each table represents a specific entity or concept, and each row in the table represents a unique record or instance of that entity. The columns, also known as attributes, define the different properties or characteristics of the entity.
- **Distributed Database:** A distributed database is a database that consists of two or more files located on different sites either on the same network or on entirely different networks.
- **MariaDB:** MariaDB is an open-source, community-developed relational database management system (RDBMS) that emerged as a fork of the MySQL database system. The developers aimed to create a drop-in replacement for MySQL that was compatible with its syntax and API while providing additional features, improved performance, and better community-driven development.
- **Front-end:** The layer above the back end is the front end and it includes all software or hardware that is part of a user interface.
- **Back-end:** The back end refers to parts of a computer application or a program's code that allow it to operate and that cannot be accessed by a user.
- **API:** API is the acronym for application programming interface — a software intermediary that allows two applications to talk to each other. APIs are an accessible way to extract and share data within and across organizations.
- **Invoice:** An invoice is a document that maintains a record of a transaction between a buyer and seller, such as a paper receipt from a store or an online record.
- **Node.js:** Node.js is an open-source, server-side JavaScript runtime environment that allows developers to run JavaScript code outside of a web browser. It enables the execution of JavaScript on the server,

providing a powerful and scalable platform for building network applications.

- **Web Applications:** Web applications, also known as web apps, are software programs or applications that run on web servers and are accessed through web browsers over the internet. Unlike traditional desktop applications that are installed on a user's computer, web applications are accessed and utilized entirely within a web browser.
- **Flutter & Dart:** Dart is the programming language used to develop applications with Flutter. Dart is an object-oriented language with a strong static type system that offers features such as a just-in-time (JIT) compiler during development and an ahead-of-time (AOT) compiler for production deployments. Dart is designed to be easy to learn and productive for developers, providing features like garbage collection, async/await syntax for handling asynchronous operations, and support for creating reusable libraries.
- **Privacy policy:** A privacy policy is a legal document or statement that explains how an organization or website collects, uses, stores, and protects the personal information of individuals who interact with their services. It outlines the practices and procedures implemented by the organization to ensure the privacy and security of user data.
- **Terms and Conditions:** Terms and conditions, also known as terms of service or terms of use, are a set of legally binding rules and agreements that govern the relationship between an organization or website and its users or customers. These terms outline the rights, obligations, and responsibilities of both parties when using the organization's services or accessing its website**.**

Contributions:

1910576101	Overall Description & ER Diagram
1912076104	Glossary & Use Case Diagram
1910876107	External Interface Req & ER Diagram
1910476108	Non-Functional Req & Sequence Diagram
1911176114	Non-Functional Req & Sequence Diagram,
1910476120	Functional Req & Use Case Diagram
1911176136	External Interface Req & ER Diagram
1910676148	Introduction & ER Diagram
1910976149	Overall Description & Sequence Diagram
1910476151	Functional Req & Use Case Diagram
1912276153	Introduction & Sequence Diagram
1810676112	Glossary & Use Case Diagram
1810576148	Functional Req & Sequence Diagram