CAR RENTAL SERVICES

Software Requirements Specification

20.10.2023

MANGALI SRINIVAS

REG:NO:12111428, ROLL:NO: RK21FA23

Prepared for
Continuous Assessment 3
Autumn 2023

Table of Contents

1. Introduction

The Car Rental Services Project is a comprehensive web application developed using JavaScript, jQuery, Bootstrap, CSS, and HTML. This Software Requirements Specification (SRS) document outlines the purpose, scope, and functionality of the project, as well as the requirements and specifications necessary to successfully design and implement the system. Car rental services have become an integral part of modern transportation solutions, offering customers the convenience of renting vehicles for various purposes such as personal travel, business trips, and special events. The Car Rental Services Project is designed to provide a user-friendly and efficient platform for both customers and car rental service providers.

1. Project Overview:
The Car Rental Service Project is an ambitious web-based application aimed at providing an efficient and user-friendly platform for both customers and administrators to facilitate car rentals. This Software Requirements Specification (SRS) document outlines the comprehensive system requirements and specifications for the development of this online car rental service.

2. Stakeholders:
The primary stakeholders in this project include:
Customers: Individuals seeking to rent vehicles for personal or business purposes.
Administrators: Staff responsible for managing the car rental fleet, customer bookings, and system configuration.
Drivers: Individuals designated to provide rental vehicles to customers and collect returns.

3. Project Objectives:
The key objectives of the Car Rental Service Project are as follows:
Customer Convenience: Provide an easy-to-use online platform for customers to search, book, and rent vehicles.
Fleet Management: Empower administrators to efficiently manage the fleet of vehicles, ensuring their availability, maintenance, and safety.
Transparent Pricing: Display clear pricing, terms, and conditions for rental services.
Security: Implement robust security measures to protect customer data and transactions.
User Support: Offer customer support and assistance to address inquiries, issues, and emergencies.

4. Technology Stack:
The project will be developed using the following technologies:
HTML: For structuring the web pages.
CSS: For styling and layout design.
JavaScript: To add interactivity and client-side functionalities.
jQuery: A JavaScript library for simplified DOM manipulation and event handling.
Bootstrap: A front-end framework for responsive and mobile-friendly design.

1.1 Purpose

The primary purpose of this project is to create an online portal that facilitates the seamless rental of vehicles, allowing users to browse available cars, make reservations, and manage their bookings. For service providers, the system streamlines the management of their vehicle fleet, bookings, and financial transactions.

1.2 Scope

The Car Rental Services Project encompasses the following key features:

User Registration and Authentication: Users can create accounts, log in, and manage their profiles, providing a personalized experience.

Vehicle Listings: The system displays a comprehensive list of available vehicles, complete with detailed information, images, and pricing.

Booking and Reservation: Users can search for vehicles, make reservations, and view booking history. Service providers can manage and confirm bookings.

Admin Dashboard: Administrators have access to a dedicated dashboard for managing users, vehicles, bookings, and other system settings.

Feedback and Support: Users can provide feedback, and customer support features are available.

1.3 Definitions, Acronyms, and Abbreviations

Definitions:

Car Rental Services Project: The web-based application developed using JavaScript, jQuery, Bootstrap, CSS, and HTML, designed to facilitate the rental of vehicles to customers and streamline vehicle management for service providers.

User: Any individual or entity accessing the Car Rental Services Project, including customers and service providers.

Administrator (Admin): A user with elevated privileges responsible for overseeing and managing the system, including user accounts, vehicle listings, and bookings.

Booking: A reservation made by a user to rent a vehicle for a specified period, including vehicle selection and payment.

Service Provider: A business or entity that provides vehicles for rental through the system, managing their fleet and bookings.

Feedback: Comments and ratings provided by users to express their satisfaction or concerns about the service.

Acronyms and Abbreviations:

SRS: Software Requirements Specification

JS: JavaScript

jQuery: JavaScript library

CSS: Cascading Style Sheets

HTML: HyperText Markup Language

UI: User Interface

UX: User Experience

DB: Database

API: Application Programming Interface

URL: Uniform Resource Locator

GUI: Graphical User Interface

HTTP: Hypertext Transfer Protocol

HTTPS: Hypertext Transfer Protocol Secure

CRM: Customer Relationship Management

QA: Quality Assurance

UI/UX: User Interface/User Experience

DBMS: Database Management System

OCR: Optical Character Recognition

CSV: Comma-Separated Values

SMTP: Simple Mail Transfer Protocol

PDF: Portable Document Format

These definitions, acronyms, and abbreviations are provided to clarify and standardize terminology and acronyms used throughout the Software Requirements Specification (SRS) document for the Car Rental Services Project. They are essential for ensuring clear communication and understanding among project stakeholders and developers.

1.4 References

1. W3Schools (https://www.w3schools.com/): W3Schools is an excellent resource for learning and referencing HTML, CSS, JavaScript, and jQuery. It provides tutorials, examples, and documentation for these technologies.

2. jQuery Documentation (https://api.jquery.com/): The official jQuery documentation is a crucial reference for understanding the usage of jQuery and its various functions.

3. Bootstrap Documentation (https://getbootstrap.com/docs/): Bootstrap's official documentation is essential for understanding the framework's components and responsive design features.

4. JavaScript MDN (https://developer.mozilla.org/en-US/docs/Web/JavaScript): MDN's JavaScript documentation is especially valuable for understanding JavaScript's core functionality and features.

5. CSS-Tricks (https://css-tricks.com/): CSS-Tricks is a helpful resource for CSS-related information, tutorials, and tips.

6. GitHub (https://github.com/): GitHub is a platform for version control and collaborative development. It can be referenced for source code repositories, libraries, and open-source projects that may have been used or contributed to during the project's development.

7. Online API Documentation: If you used third-party APIs for services like payment processing, mapping, or other integrations, include references to the respective API documentation. For example, Stripe API (https://stripe.com/docs), Google Maps JavaScript API (https://developers.google.com/maps/documentation/javascript/), etc.

1.5 Overview

This subsection of the Software Requirements Specification (SRS) document for the Car Rental Services Project provides an overview of what the rest of the SRS contains and explains how the document is organized.

Description of SRS Content:

The SRS for the Car Rental Services Project is a comprehensive document that defines the requirements and specifications for the development of a web-based application using JavaScript, jQuery, Bootstrap, CSS, and HTML. It serves as a detailed guide for all stakeholders, including developers, designers, project managers, and clients, to understand the project's scope, features, and functional and non-functional requirements.

The document contains the following key sections:

Functional Requirements: Detailed descriptions of the system's functional features, including user registration, vehicle listings, booking and reservation management, payment processing, admin dashboard, and feedback and support features.

Non-Functional Requirements: Descriptions of non-functional aspects such as performance, security, usability, and compliance requirements.

Use Cases: Specific use case scenarios that illustrate how different actors interact with the system, including users, administrators, and service providers.

Data Requirements: Information about data sources, storage, and processing, including databases and data integration.

System Architecture: An outline of the system's architecture, including components, modules, and their interactions.

Testing and Quality Assurance: Details regarding the testing process and quality assurance measures to ensure the system's reliability and functionality.

Project Timeline: An estimated timeline for development, testing, and deployment of the Car Rental Services Project.

Organization of the SRS:

The SRS document is organized in a logical and structured manner to make it easy for readers to access specific information and understand the project's requirements and specifications. It follows a standardized structure, including:

- A clear and informative Introduction section that sets the context for the document.
- Functional and non-functional requirements sections that provide a comprehensive understanding of the project's features and quality attributes.
- Use cases to illustrate real-world scenarios and interactions.
- Data requirements and system architecture sections for data and technical details.
- Testing and quality assurance information to ensure the project's reliability and robustness.
- A project timeline that outlines the expected development and deployment schedule.

This well-organized structure allows project stakeholders, including developers, designers, and clients, to quickly locate and comprehend the information they need to contribute to or oversee the project. The SRS for the Car Rental Services Project serves as a foundational document that will guide the successful development and implementation of the system, ensuring that it meets the specified requirements and objectives.

2. General Description

This section provides a comprehensive overview of the Car Rental Services Project, outlining its context, features, user characteristics, constraints, and dependencies.

2.1 Product Perspective

The Car Rental Services Project is a standalone web application that operates independently. It serves as a user-centric platform connecting customers and car rental service providers. There is no direct integration or dependency on external systems, but it may make use of third-party services for payment processing and mapping functionalities. The project's interfaces include user interfaces for customers, service providers, and administrators, as well as a database for data storage and retrieval.

2.2 Product Functions

The Car Rental Services Project includes the following key functions:

User Registration and Authentication: Users can register for accounts and authenticate themselves to access the system's features.

Vehicle Listings: The system displays a list of available vehicles with detailed information and images.

Booking and Reservation Management: Users can search for vehicles, make reservations, and view their booking history. Service providers can manage and confirm bookings.

Admin Dashboard: Administrators have access to a dashboard for managing users, vehicles, bookings, and system settings.

Feedback and Support: Users can provide feedback and access customer support features.

2.3 User Characteristics

The Car Rental Services Project caters to three primary user roles:

Customers: These are individuals or entities seeking to rent vehicles. They may have varying levels of technical proficiency and experience with online booking systems.

Service Providers: Car rental businesses that list their vehicles on the platform and manage bookings. They are expected to be familiar with the system's administrative functions.

Administrators: System administrators responsible for overseeing and managing the entire platform, including user accounts, vehicles, and bookings. They are expected to have a deeper understanding of the system's administrative features.

2.4 General Constraints

Browser Compatibility: The system will be compatible with major web browsers, but certain features may vary depending on the browser's capabilities.

Internet Connection: Users will require an internet connection to access the system and make reservations.

Security: The system will employ security measures to protect user data and payment information, but no system can guarantee complete security.

2.5 Assumptions and Dependencies

Third-Party Services: The project assumes that third-party services (e.g., payment gateways, mapping services) will be available and functional.

Data Integrity: It is assumed that the data stored within the system's database is accurate and up-to-date.

User Internet Access: Users are assumed to have access to the internet and devices capable of accessing the web application.

Web Development Technologies: The project assumes that the web development technologies (JavaScript, jQuery, Bootstrap, CSS, HTML) will be available and properly implemented.

These assumptions and dependencies serve as the foundation for the project's development and are essential considerations in planning, designing, and implementing the Car Rental Services Project.

3. Specific Requirements

This section outlines the specific requirements for the Car Rental Services Project, including external interface requirements.

3.1 External Interface Requirements

External interface requirements describe the interactions between the system and its external components, such as user interfaces, hardware, software, and communication interfaces.

3.1.1 User Interfaces

The Car Rental Services Project will provide user interfaces for the following primary actors:

Customers: The customer interface will allow users to:

- Register and authenticate.
- Search and view available vehicles.

- Make vehicle reservations.
- View and manage their booking history.
- Provide feedback and support requests.
- Service Providers: The service provider interface will allow businesses to:
- Register and authenticate.
- List their available vehicles with descriptions and images.
- Confirm or reject booking requests.
- Manage their vehicle fleet.

Administrators: The administrator interface will provide access to:

- User management features.
- Vehicle and booking management.
- System configuration and settings.

The user interfaces will be designed with a user-friendly and responsive approach, ensuring a seamless experience on various devices and screen sizes.

3.1.2 Hardware Interfaces

The Car Rental Services Project does not have direct hardware dependencies. However, the following hardware requirements are applicable:

Client Devices: Users will require internet-enabled client devices (e.g., computers, smartphones, tablets) to access the web application.

Servers: The project will be hosted on web servers capable of handling web traffic, databases for data storage, and server-side scripting support.

3.1.3 Software Interfaces

The project interacts with software components as follows:

Web Browsers: The system should be compatible with common web browsers, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari, ensuring a consistent user experience.

Database Management System (DBMS): The project will interface with a DBMS to store and retrieve data. The choice of DBMS will be specified during the system's implementation.

3.1.4 Communications Interfaces

Communication interfaces pertain to data exchange between the system and external entities:

Internet Connectivity: Users must have access to the internet to use the system.

APIs and Web Services: Interaction with third-party services, including payment gateways and mapping services, will involve the use of their provided APIs.

Email Communication: The system may send automated emails for registration confirmation, booking details, and support responses. It will use standard email protocols for this purpose.

These external interface requirements ensure that the Car Rental Services Project seamlessly integrates with various components while providing a user-friendly and accessible experience to its users.

3.2 Functional Requirements

3.2.1 Functional Requirement 1: User Registration and Authentication

3.2.1.1 Introduction

The system shall allow users to create accounts and authenticate their identities to access the platform

3.2.1.2 Inputs

- User-provided registration details, including name, email, password.
- User credentials (email and password) for authentication.

3.2.1.3 Processing

- The system shall validate user registration details.
- User passwords shall be securely hashed and stored.
- Authentication shall compare provided credentials with stored credentials.

3.2.1.4 Outputs

- Upon successful registration, the system shall confirm the user's registration and provide a unique user ID.
- After successful authentication, the user shall gain access to their account and associated features.

3.2.1.5 Error Handling

- In case of registration errors (e.g., duplicate email), the system shall display appropriate error messages.
- Failed authentication attempts shall result in error messages without revealing specific authentication details.

3.2.2 Functional Requirement 2: Vehicle Search and Selection

3.2.2.1 Introduction
Users shall be able to search for vehicles based on criteria such as location, dates, vehicle type, and price.

3.2.2.2 Inputs
User-defined search criteria, including location, pickup and drop-off dates, vehicle type, and price range.

3.2.2.3 Processing
The system shall query the database to retrieve a list of available vehicles that match the specified criteria.

3.2.2.4 Outputs
The system shall display search results, including a list of vehicles meeting the user's criteria.
Users can select a vehicle to view detailed information and proceed with the booking.

3.2.2.5 Error Handling
- If the search criteria do not yield any matching vehicles, the system shall display a message such as "No vehicles matching your criteria were found."
- The user will be prompted to revise their search criteria, check their inputs, or consider alternate dates or locations.

3.2.3 Functional Requirement 3: Booking and Reservation Management

3.2.3.1 Introduction
Users shall be able to make vehicle reservations, including specifying pickup and drop-off dates.
3.2.3.2 Inputs
User-selected vehicle, reservation dates, and personal details.

3.2.3.3 Processing
- The system shall check vehicle availability for the specified dates.
- Upon reservation request, the system shall update the booking records and vehicle availability status.

3.2.3.4 Outputs
The system shall confirm the reservation and provide booking details.
Users can view their booking history and manage reservations.

3.2.3.5 Error Handling
- If the selected vehicle is not available for the specified dates, the system shall display an error message, such as "The selected vehicle is not available for the chosen dates. Please select alternative dates or choose a different vehicle."
- The user will be encouraged to adjust the booking dates or select an available vehicle.

3.2.4 Functional Requirement 4: Payment Processing

3.2.4.1 Introduction
The system shall facilitate secure online payments for reservations.

3.2.4.2 Inputs
User-provided payment information, including credit card details.

3.2.4.3 Processing
The system shall securely process payment transactions using a chosen payment gateway.

3.2.4.4 Outputs
Upon successful payment, the system shall generate a payment receipt and confirm the reservation.

3.2.4.5 Error Handling
- Implement real-time transaction verification with the payment gateway or processor.
- Handle communication errors with the gateway and provide users with clear error messages.

3.2.5 Functional Requirement 5: Administrative Dashboard

3.2.5.1 Introduction
Administrators shall have access to a dedicated dashboard to manage users, vehicles, and bookings.

3.2.5.2 Inputs
Administrator login credentials and selected management actions.

3.2.5.3 Processing
The system shall verify administrator credentials.
The administrative dashboard shall provide options to manage user accounts, vehicle listings, and bookings.

3.2.5.4 Outputs
Administrators shall be able to view and edit user accounts, manage vehicle listings, confirm bookings, and monitor system performance.

3.2.5.5 Error Handling
- Handle errors related to data retrieval from databases or external sources.
- Implement mechanisms to gracefully handle database connection failures or data source unavailability.

These functional requirements outline key features and interactions of the Car Rental Services Project. They represent the core functionalities necessary to provide a comprehensive car rental service through the web application.

3.3 Non-Functional Requirements

Non-Functional Requirements (NFRs) are essential for defining the qualities and constraints of a software system. Below are some non-functional requirements for a car rental service project developed using HTML, CSS, JavaScript, jQuery, and Bootstrap as specified in the Software Requirements Specification (SRS) document:

3.3.1 Performance:

Response Time: The system should provide a responsive user experience with a page load time of less than 3 seconds for all web pages and transactions.

Scalability: The application should be able to handle a minimum of 100 concurrent users without significant performance degradation.

Resource Utilization: Optimize code and resources to ensure efficient operation and minimal server resource consumption.

3.3.2 Reliability:

Availability: The system should be available 24/7, with a target uptime of at least 99.9%.

Error Handling: Implement comprehensive error handling to prevent system crashes and provide meaningful error messages to users.

Data Integrity: Ensure data consistency and accuracy, and implement mechanisms for detecting and recovering from data corruption.

3.3.3 Availability:

Redundancy: Implement redundancy and failover mechanisms to ensure continuous service availability in case of server failures or other critical issues.

Maintenance Windows: Specify scheduled maintenance windows for updates and maintenance tasks to minimize disruption to users.

Backup and Recovery: Regularly back up data and establish a recovery plan in case of data loss or system failures.

3.3.4 Security:

Authentication and Authorization: Implement secure user authentication and authorization mechanisms to ensure that only authorized users can access and modify data.

Data Encryption: Ensure that sensitive user data, such as login credentials and payment information, is transmitted and stored securely through encryption.

Protection against Common Attacks: Protect the system against common web vulnerabilities like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).

User Privacy: Comply with data protection regulations, such as GDPR, and ensure user data privacy.

Logging and Auditing: Maintain logs of all security-related events for auditing and analysis purposes.

3.3.5 Maintainability:

Code Documentation: Provide comprehensive documentation for the codebase, including comments and inline documentation for developers.

Modularity: Design the system with a modular architecture to facilitate easier maintenance and updates.

Version Control: Use version control systems like Git to manage the source code and track changes.

Bug Tracking: Implement a bug tracking system to log and prioritize reported issues, enabling timely bug fixes.

Regular Updates: Plan for regular updates and improvements to the system to address emerging issues and enhance functionality.

3.3.6 Portability:

Cross-Browser Compatibility: Ensure that the web application works consistently across various web browsers, including Chrome, Firefox, Safari, and Edge.

Responsive Design: Develop a responsive design to make the application accessible and usable on different devices, such as desktops, tablets, and mobile phones.

Cross-Platform Compatibility: Make sure the application is compatible with different operating systems, such as Windows, macOS, and Linux.

Compliance with Web Standards: Adhere to web standards and best practices to ensure the project's portability and compatibility across various platforms and devices.
These non-functional requirements should serve as guidelines for the development of your car rental service project and help ensure that it meets the necessary performance, reliability, availability, security, maintainability, and portability criteria. Be sure to include these NFRs in your Software Requirements Specification (SRS) document for clear communication with the development team and stakeholders.

3.4 Design Constraints

3.4.1 Technology Stack:
The project shall be developed using HTML5, CSS3, JavaScript, jQuery, and the Bootstrap framework.
The system shall be compatible with the latest versions of major web browsers, including Chrome, Firefox, Safari, and Edge.

3.4.2 Responsive Design:
The user interface must be designed to be responsive, ensuring a consistent and user-friendly experience across various devices, including desktop computers, tablets, and mobile phones.

3.4.3 Database Management System:
The project shall use a specific relational database management system (e.g., MySQL, PostgreSQL) for data storage and retrieval.

3.4.4 Hosting Environment:
The application shall be hosted on a specified web hosting platform or infrastructure (e.g., AWS, Azure, or a specific hosting provider).

3.4.5 Third-Party APIs:
Integration with third-party services, such as payment gateways, mapping services, and vehicle tracking systems, may be required. The project shall adhere to the APIs and guidelines provided by these third-party services.

3.4.6 Legal and Regulatory Compliance:
The system must comply with all relevant legal and regulatory requirements, including data protection regulations (e.g., GDPR) and any specific industry standards.

3.5 Other Requirements

3.5.1 User Documentation:
Provide comprehensive user documentation, including user manuals and guides, to assist users in navigating and using the system effectively.

3.5.2 Training:
The development team shall provide training sessions for the system's administrators and support staff on how to operate and maintain the system.

3.5.3 Maintenance and Support:
A maintenance and support plan should be in place, including response times for bug fixes, updates, and ongoing technical support for end-users.

3.5.4 Data Backup and Recovery:
Regular data backups shall be performed to ensure data integrity, and a recovery plan must be established in case of data loss or system failures.

3.5.5 Performance Testing:
Comprehensive performance testing must be conducted to ensure the system meets the specified performance requirements and can handle expected user loads.

3.5.6 Testing and Quality Assurance:
A thorough testing and quality assurance process, including unit testing, integration testing, and user acceptance testing, must be carried out to ensure the reliability and quality of the system.

3.5.7 Change Management:
A change management process should be in place to document and manage changes to the system, including version control and change tracking.

3.5.8 Accessibility:
The system must adhere to accessibility standards (e.g., WCAG) to ensure that it is usable by individuals with disabilities.

3.5.9 Localization and Internationalization:
The system should be designed to support multiple languages and regions, allowing for localization and internationalization as required.

These design constraints and other requirements will provide a comprehensive framework for the development, deployment, and maintenance of the car rental service project. Tailor these requirements to your specific project's needs and include them in your SRS document to ensure a common understanding among stakeholders and the development team.

4. Analysis Models

Use Case Diagram:
Illustrate the interactions between different actors (e.g., customers, admin, drivers) and the car rental system.
Use tools like draw.io or Lucidchart to create the diagram.

Sequence Diagram:
Show the sequence of events and interactions between objects or components in the system when a user makes a booking or performs other actions.
Represent interactions between the front-end (HTML, CSS, JavaScript) and back-end components.

Activity Diagram:
Detail the workflow and activities performed by the system and users when, for example, reserving a car, updating a reservation, or canceling a booking.

Class Diagram:
Model the key classes, their attributes, and relationships in the system, including classes for vehicles, customers, reservations, and transactions.

Entity-Relationship Diagram (ERD):
Illustrate the database structure by defining entities and their relationships, helping to design the database schema for storing information about vehicles, customers, transactions, etc.

State Diagram:
Describe the different states that reservations, vehicles, and users can be in (e.g., available, booked, rented) and transitions between these states.

Mockups and Wireframes:
Create visual representations of the user interface using HTML, CSS, and Bootstrap, showing the layout, design, and user interactions.
Use tools like Adobe XD, Sketch, Figma, or simple HTML/CSS mockups.

Data Flow Diagram (DFD):
Depict the flow of data and information throughout the system, from user input to database storage and retrieval.

Gantt Chart or Project Timeline:
Present a project schedule or timeline to indicate when different features or modules will be developed and integrated.

Architecture Diagram:
Show the high-level architecture of the system, illustrating how the front-end (HTML, CSS, JavaScript, jQuery) interacts with the back-end components and external services.

Navigation Flowchart:
Define the structure and navigation paths for the web application, outlining how users can move from one section to another.

Test Cases and Scenarios:
Outline test cases to verify the functionality of the system, including both unit and integration testing for the HTML, CSS, JavaScript, jQuery, and Bootstrap components.

4.1 Data Flow Diagrams (DFD)

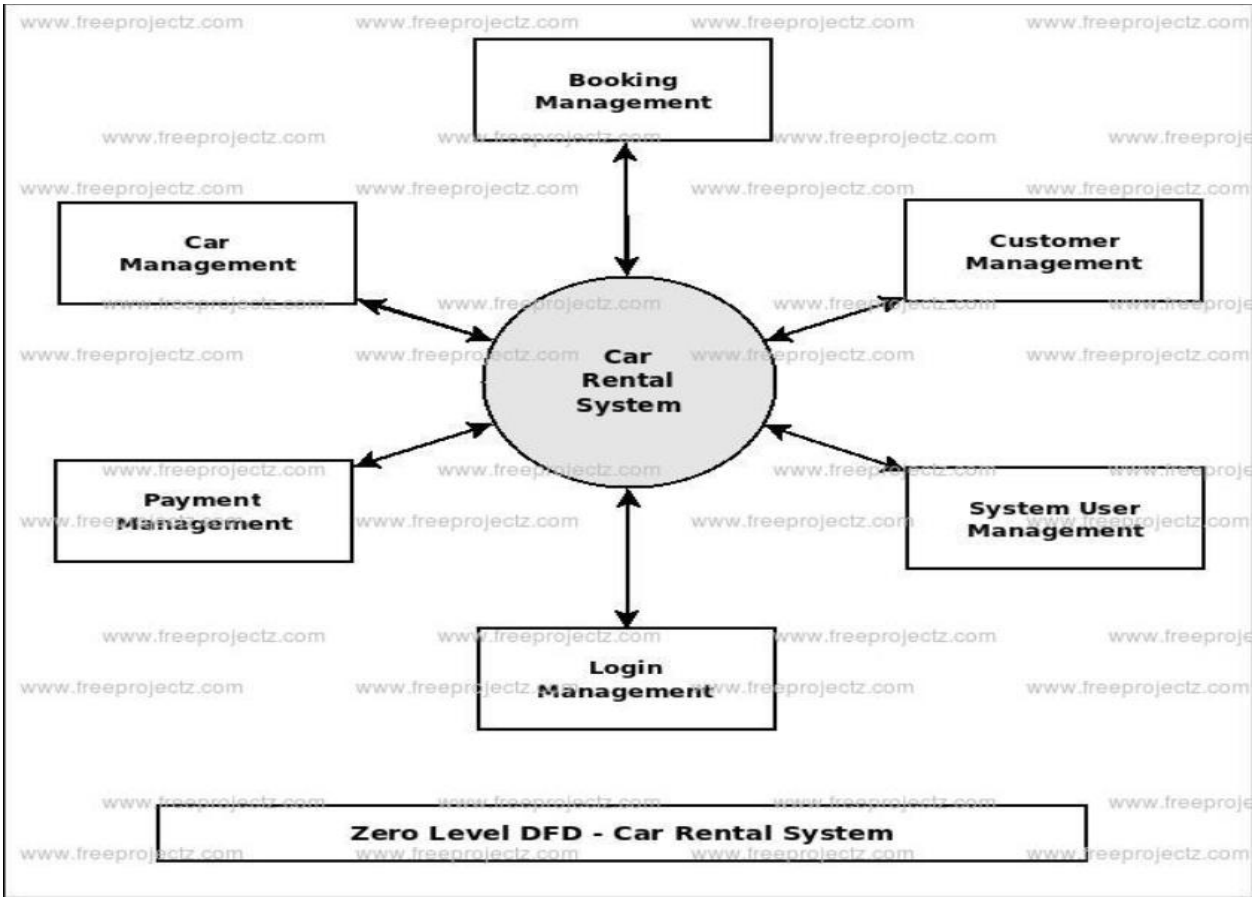
Car Rental System Data flow diagram is often used as a preliminary step to create an overview of the Car Rental without going into great detail, which can later be elaborated.it normally consists of overall application dataflow and processes of the Car Rental process. It contains all of the user flow and their entities such all the flow of Car, Customer, Booking, Payment, Insurance, Rent, Bill. All of the below diagrams has been used for the visualization of data processing and structured design of the Car Rental process and working flow.

Zero Level Data Flow Diagram(0 Level DFD) Of Car Rental System :

This is the Zero Level DFD of Car Rental System, where we have eloborated the high level process of Car Rental. It's a basic overview of the whole Car Rental System or process being analyzed or modeled. It's designed to be an at-a-glance view of Insurance,Rent and Bill showing the system as a single high-level process, with its relationship to external entities of Car,Customer and Booking. It should be easily understood by a wide audience, including Car,Booking and Insurance In zero leve DFD of Car Rental System, we have described the high level flow of the Car Rental system.

High Level Entities and proccess flow of Car Rental System:

- Managing all the Car
- Managing all the Customer
- Managing all the Booking
- Managing all the Payment
- Managing all the Insurance
- Managing all the Rent
- Managing all the Bill



First Level Data Flow Diagram(1st Level DFD) Of Car Rental System :