SOFTWARE REQUIREMENTS SPECIFICATION

**For**

**Car Rental System**

**Prepared by:-**

*Sabari Krishnan B R*

*Sambath Kumar K*

*Vasanth K*

*Yogeshwaran R*

# Introduction

## Purpose

The main objective of this document is to outline the requirements for the Car Rental System project. It details both functional and non-functional requirements as proposed by the client. The project aims to create a user-friendly environment for managing car rental details, facilitating easy circulation through computer systems, and generating various reports. The document also describes hardware and software interface requirements using ER diagrams and UML diagrams.

## Document Conventions

* + - Entire document should be justified.
    - Convention for Main title

Font face: Times New Roman Font style: Bold

Font Size: 14

* + - Convention for Sub title

Font face: Times New Roman Font style: Bold

Font Size: 12

* + - Convention for body

Font face: Times New Roman Font Size: 12

## Scope of Development Project

The Car Rental System project is designed to streamline the management of car rentals, providing a user-friendly interface for staff and customers. It includes features for booking, tracking, and reporting on car rentals. The system is adaptable for use in various situations and can be easily updated with new features. Java is chosen as the development language for its performance, platform independence, and development tools.

## Definitions, Acronyms and Abbreviations

JAVA -> platform independence SQL-> Structured query Language ER-> Entity Relationship

UML -> Unified Modeling Language

IDE-> Integrated Development Environment SRS-> Software Requirement Specification

## 1.5 References

**Books:**

Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices (ACM Press) by Michael Jackson

Software Requirements (Microsoft) Second Edition by Karl E. Wiegers

Software Engineering: A Practitioner’s Approach Fifth Edition By Roger S. Pressman

# Overall Descriptions

## Product Perspective

Use Case Diagram of Car Rental System

This high-level diagram illustrates the basic overview of the Car Rental System. Users can be staff or customers. The system provides search functionality based on various criteria such as car model or registration number. Staff can add/update car details, and users can request booking/renewal/return of cars based on certain criteria.

## Product Function

Entity Relationship Diagram of Car Rental System

The Car Rental System offers real-time information about available cars and user details. The project aims to minimize manual work, manage bookings, returns, calculate/manage fines, and generate various reports. Librarians (administrators) control user accounts and car information. Members can view their account information and request bookings.

## User Classes and Characteristics

The car rental system features distinct functionalities tailored for two primary user types: Admin and Members (customers). Admin users wield administrative privileges, encompassing the issuance of cars, viewing car categories, adding/updating car information, managing reports, and accessing all user accounts. The initial action involves the Admin logging into the system, leading to an administrative dashboard. From there, they can navigate through functionalities such as viewing car categories, updating car information, issuing cars, managing reports, and overseeing all user accounts. On the other hand, Members engage with the system for rental purposes, being able to view car categories, access a list of available cars, create and manage their accounts, view issued cars, request new cars, check booking history, and search for specific cars.

For Admin interactions, the use case unfolds with the login process, granting access to the administrative dashboard. The Admin then engages in tasks like viewing car categories, updating car information, issuing cars to members, managing reports, and accessing all user accounts, each action contingent on appropriate permissions and system prerequisites. Conversely, the Member's journey commences with creating an account, followed by logging in to access a personalized dashboard. Members can peruse available car categories, request new cars, view issued cars, inspect booking history, and search for specific cars, enhancing their overall experience within the car rental system.

In conclusion, this use case encapsulates the nuanced interactions within a car rental system, delineating the roles of Admin and Members, and portraying a comprehensive narrative that elucidates the system's functionalities and user experiences. The seamless integration of administrative and member-specific tasks ensures a robust and user-friendly environment, fostering efficient car rental management.

## Operating Environment

The product operates in the Windows environment and is compatible with major browsers (Microsoft Internet Explorer, Google Chrome, Mozilla Firefox). The hardware configuration includes a 40 GB hard disk, 15” color monitor, and a standard keyboard. The system requires an internet connection.

## Assumptions and Dependencies

The assumptions are:-

Error-free coding

User-friendly system

Information stored in an accessible database

Adequate storage capacity and quick database access

24/7 system availability

**Dependencies:**

Specific hardware and software for system operation

Development based on specified requirements

User (admin) understanding of the product

**2.6 Requirement**

**Software Configuration:**

Front End: Java (Supported by Sun Microsystems)

Back End: Microsoft SQL Server

Operating System: Windows NT, Windows 98, Windows XP

Language: Java Runtime Environment, NetBeans 7.0.1 (Front End)

Database: MS SQL Server (Back End)

**Hardware Configuration:**

Processor: Pentium(R) Dual-core CPU

Hard Disk: 40GB

RAM: 256 MB or more

**2.7 Data Requirement**

Inputs consist of queries to the database, and outputs include solutions and user details. The system handles queries like creating an account, selecting cars, and retrieving account details. Outputs include time, date, and currently booked cars.

# External Interface Requirement

**3.1 GUI**

The software provides a graphical interface for users and administrators. Key features include quick report viewing, stock verification, customizable user interface, and interaction with the user management module. The design is simple, follows standard templates, and includes login/logout modules.

**Login Interface:**

User registration for new accounts

Login with username and password

Error messages for incorrect login details

**Search:**

Customer can search for cars by entering the car model.

**Categories View:**

Displays categories of cars

Allows customer to add/edit/delete categories

**Control Panel:**

Manages user accounts

Adds/edits/removes cars

Manages lending options

# System Features

# The system guarantees user authentication and validation through a rigorous registration process involving email verification and optional two-factor authentication. For administrator monitoring, a dedicated dashboard is established, offering real-time updates on user activities and account statuses, supported by a comprehensive notification system and detailed logging for accountability. Account status updates are efficiently managed through automated processes and predefined user roles, ensuring smooth transitions based on user activities or established criteria. To enforce accountability and restrict member access, the system employs strict access controls, data encryption, and audit trails, recording all interactions. Privacy settings allow users to customize their visibility preferences. Together, these features create a secure and transparent environment, safeguarding user accounts and ensuring a high level of system integrity.

# Other Non-functional Requirements

## Performance Requirement

The system places a premium on achieving fast and accurate performance, emphasizing efficiency in its operations. Robust error-handling mechanisms are integrated to proactively prevent information loss, ensuring the integrity of critical data. With a design that prioritizes scalability, the system exhibits the capability to seamlessly handle substantial volumes of data without succumbing to faults. This focus on both speed and accuracy, coupled with robust error management and scalable data handling, collectively contributes to a resilient and reliable system that can efficiently process large datasets without compromising performance or risking the loss of vital information.

* 1. **Safety Requirement**

In adherence to safety requirements, the system incorporates measures to safeguard against potential risks. Regular database backups are a crucial aspect of data management, providing a preventive strategy to mitigate the impact of data loss. By routinely creating and storing backup copies of the database, the system ensures data integrity and facilitates swift recovery in the event of unexpected incidents. Furthermore, the inclusion of Uninterruptible Power Supply (UPS) or inverters addresses vulnerabilities related to power supply failure. This proactive approach ensures uninterrupted system operation, protecting against data corruption and system downtime that may result from power fluctuations or outages. Together, these safety measures enhance the overall reliability and resilience of the system, contributing to a secure and stable operational environment.

## Security Requirement

The system prioritizes data protection through the use of a secured database, ensuring that sensitive information remains confidential. Normal users are granted read-only access, enhancing security by limiting their capabilities. Different user access constraints are implemented to tailor permissions based on roles, contributing to a fine-grained security model. Robust user authentication mechanisms are in place to prevent password hacking, and separate accounts are maintained for administrators and members, further fortifying system security.

## Requirement attributes

Underscore the need for multiple administrators with project modification rights, ensuring effective system management. Embracing an open-source project approach fosters transparency and community collaboration. The user-friendly database design facilitates efficient interactions, and easy download and installation processes contribute to a seamless user experience.

## Business Rules

Emphasize adherence to project cost and discount offers to ensure financial sustainability. The system is committed to avoiding illegal rules and protocols, aligning with ethical and legal standards.

## User Requirement

The user requirements for a car rental system are multifaceted, encompassing various aspects to ensure a seamless and satisfying experience for both customers and administrators. The system should facilitate easy user registration and authentication with secure mechanisms, offer a user-friendly interface for browsing and selecting available cars, and provide a straightforward booking process with transparent pricing and payment integration. Personalized user profiles, notification systems, and features for cancellation or modification contribute to a customer-centric approach. Additionally, a feedback and rating system, along with administrator features such as dashboards and reporting tools, enhance transparency and management capabilities. Mobile accessibility, robust security measures, and comprehensive support channels further contribute to the overall usability and reliability of the car rental system

# Other Requirements

## Data and Category Requirement

In a car rental system, user categories play a crucial role in tailoring the experience for various stakeholders. Common user categories include regular customers, administrators, and possibly employees managing the rental locations. Differentiated access rights based on user categories ensure that each user type has appropriate permissions. Regular customers may have access to browsing and booking functionalities, while administrators enjoy extended privileges to manage car listings, user accounts, and bookings. The system should implement robust authentication mechanisms to secure these varying access levels. Simultaneously, different car categories with coded data, such as economy, sedan, SUV, or luxury, facilitate efficient navigation and selection. These coded categories enable users to filter and search for specific types of vehicles, streamlining the booking process. This dual categorization approach, focusing on both user roles and car types, enhances the overall flexibility, security, and usability of the car rental system.

## Appendix

A: Admin, Abbreviation, Acronym, Assumptions; B: Books, Business rules; C: Class, Client, Conventions

## Glossary

The following are the list of conventions and acronyms used in this document and the project as well:

* + - Administrator: A login id representing a user with user administration privileges to the software
    - User: A general login id assigned to most users
    - Client: Intended users for the software
    - SQL: Structured Query Language; used to retrieve information from a database
    - SQL Server: A server used to store data in an organized format
    - Layer: Represents a section of the project
    - User Interface Layer: The section of the assignment referring to what the user interacts with directly
    - Application Logic Layer: The section of the assignment referring to the Web Server. This is where all computations are completed
    - Data Storage Layer: The section of the assignment referring to where all data is recorded
    - Use Case: A broad level diagram of the project showing a basic overview
    - Class diagram: It is a type of static structure diagram that describes the structure of a system by showing the system’s cases, their attributes, and the relationships between the classes
    - Interface: Something used to communicate across different mediums
    - Unique Key: Used to differentiate entries in a database

## Class Diagram

A class is an abstract, user-defined description of a type of data. It identifies the attributes of the data and the operations that can be performed on instances (i.e. objects) of the data. A class of data has a name, a set of attributes that describes its characteristics, and a set of operations that can be performed on the objects of that class. The classes’ structure and their relationships to each other frozen in time represent the static model.