University of the Punjab
Gujranwala Campus

Final Year Project Documentation

Project ID: BSIT-2011

STATEMENT OF SUBMISSION

This is to certify that following students have successfully completed the final project named as: **Car Rental Management System** at University of the Punjab, Gujranwala Campus, Gujranwala, to fulfill the partial requirement of the degree of **Bachelor in Information Technology**

Sr. No.	Student's Roll No.	Student's Name
1	BIT20004	Maaz Ahmad
2	BIT20037	Hassan Waris
3	BIT20047	Muhammad Talha Fiaz

Internal Supervisor
Dr. Ghulam Mustafa
PUGC.

Co-Supervisor
Mr. Burhan Sheikh
PUGC.

Project Coordinator
Prof. Muhammad Younas
PUGC.

Incharge IT Department
Dr. Naveed Ahmad Khan
PUGC.

Director General
Prof. Dr. Muhammad Mudasar Ghafoor
PUGC.

PROOFREADING CERTIFICATE

It is to certify that I have read the document thoroughly and circumspectly. I am convinced that the resultant project does not contain any spelling, punctuation or grammatical mistakes as such. All in all, I find this document well organized and I am in no doubt that its objectives have been successfully met.

Ms. Aadila Hussain
PUGC.

Acknowledgements

This project is the direct result of the teamwork. We sincerely thank the instructors and students who have shared their suggestion with us over last few weeks.
Many thanks go to **Dr. Ghulam Mustafa** our Project supervisor, whose management helped us to get this project produced in timely and efficient manner.
And finally, we thank **Prof. Dr. Salman Naseer and Prof. Muhammad Younas** who weren't directly involved in this project but who have stood by us the whole way, we appreciate your encouragement and support. We are also thankful to our friends and families whose silent support led us to complete our project.

Write to us

We welcome your response to this project. If there is anything you want to mention about the improvement of this project please let us know:

Maaz Ahmad	mzit3116@gmail.com
Muhammad Talha Fiaz	talhabsit@gmail.com
Hassan Waris	hassanwaris317@gmail.com

ABSTRACT

The Car Rental Management System (CRMS) is a comprehensive software solution tailored to streamline the operations of car rental businesses. It serves as a centralized platform that facilitates and automates various aspects of the rental process, catering to both customers and rental companies.

For customers, the CRMS offers an intuitive and user-friendly interface accessible through web browsers. Within the system, customers can easily browse through the available inventory of vehicles, view detailed specifications, and make reservations based on their preferences and rental needs. The platform provides real-time updates on vehicle availability and pricing, ensuring transparency and enabling customers to make informed decisions.

On the rental company's side, the CRMS provides a suite of tools for efficient fleet management. Rental administrators can maintain a comprehensive database of vehicles, including information on make, model, and availability. Moreover, the CRMS streamlines administrative tasks associated with rental operations. It simplifies processes such as invoicing, billing, and payment collection, reducing manual effort and minimizing errors. Additionally, integrated reporting functionalities provide rental companies with valuable insights into key performance metrics. These insights empower businesses to make data-driven decisions, identify areas for improvement, and optimize their operations for greater efficiency and profitability.

<u>For Our Beloved Parents and Teachers</u>

"Dedicated to our parents and teachers without whose wholehearted support, encouragement and guidance it would have been impossible for us to make this project"

TABLE OF CONTENTS

Project Overview Statement

<table>
<tr><td><b>Project Title</b></td><td>Car Rental Management System</td></tr>
<tr><td><b>Group Leader</b></td><td>Maaz Ahmad</td></tr>
<tr><td><b>Project Members</b></td><td><table>
<tr><th>Name</th><th>Roll #</th></tr>
<tr><td>Maaz Ahmad</td><td>BIT20004</td></tr>
<tr><td>Hassan Waris</td><td>BIT20037</td></tr>
<tr><td>Talha Fiaz</td><td>BIT20047</td></tr>
</table></td></tr>
<tr><td><b>Project Supervisor</b></td><td>Dr. Ghulam Mustafa</td></tr>
<tr><td><b>Project Goal</b></td><td>To develop a comprehensive Car Rental Management System that will enable users to rent vehicles easily and efficiently by creating a user-friendly and responsive website. This system will provide a convenient platform for users to browse, select, and rent vehicles of their choice while also offering administrators the tools to manage the rented cars and reservations effectively.</td></tr>
<tr><td><b>Main Objectives</b></td><td>1. System should be able to permit user to register/login into website.
2. System should be able to help customers for checking list of cars available for rent.
3. System should be able to help customers to search the desired car.
4. System should be able to allow user to reserve the car available for rent.
5. System should be able to help the admin to maintain the list of cars.
6. System should be able to help the admin to update statuses of cars and rentals.</td></tr>
<tr><td><b>Project Success Criteria</b></td><td>• Customer’s satisfaction
• All objectives should be achieved successfully
• To achieve the objectives of the project within time, budget, and available resources.</td></tr>
<tr><td><b>Platform</b></td><td>■ Web based   □ Distributed
□ Desktop based   □ Other</td></tr>
</table>

1. Project Planning and Scheduling

Introduction

This part of the deliverable is all about planning and scheduling of project. This deliverable must contain following artifacts:

a. Project Feasibility
b. Project Scope
c. Project Costing
d. Critical Path Method Analysis (CPM Analysis)
e. Gantt Chart
f. Introduction to Team Members
g. Tools and Technologies
h. Risk List

1.1 Project Feasibility Report

Feasibility means the extent to which appropriate data and information are readily available or can be obtained with available resources such as staff, expertise, time, and equipment. It is basically used as a measure of how practical or beneficial the development of a software system will be to you (or organization).

1.1.1 Technical Feasibility

The implementation of a web-based Car Rental Management System (CRMS) is technically feasible, leveraging modern web technologies to deliver a scalable, secure, and user-friendly solution. Utilizing a web-based architecture offers several advantages, including cross-platform compatibility, centralized management, and real-time access for users and administrators.

While integrating third-party services, such as payment gateways can enhance the functionality and feature set of the CRMS, it also introduces complexities and challenges that require careful consideration and planning. Implementing third-party services may involve compatibility issues, security concerns, optimization problems and reliability issues.

1.1.2 Operational Feasibility

The CRMS can be seamlessly integrated into the organization's existing business processes and workflows. Its features and functionalities are designed to streamline car rental operations, enhance efficiency, and improve service delivery, aligning with the organization's objectives and requirements. The web-based CRMS offers a user-friendly interface and intuitive navigation, facilitating easy adoption customers. Comprehensive user guides, and support resources can be implemented to ensure that users are equipped with the necessary knowledge and skills to effectively utilize the system.

1.1.3 Economic Feasibility

A web-based CRMS eliminates the need for significant upfront hardware investments, as the approach reduces capital expenditures, minimizes maintenance costs, and provides a predictable pricing model based on subscription fees or usage-based pricing, facilitating budgetary planning and financial management.

By offering enhanced services, personalized customer experiences, and innovative features, the CRMS attracts and retains customers, fosters loyalty, and differentiates the organization from competitors, ultimately driving revenue growth and market share expansion.

1.1.4 Schedule Feasibility

Developing the CRMS within the desired timeframe poses challenges and may exceed established project milestones or operational timelines. Factors contributing to this delay could include technical complexities, integration issues with existing systems, or unforeseen challenges during the development and testing phases. Therefore, careful evaluation, proactive planning, and effective project management are essential to address potential obstacles, optimize the implementation process

1.1.5 Information Feasibility

While the necessary information and resources are largely available for the successful development and operation of the CRMS, additional guidance, support, and oversight from supervisor may be needed to address specific challenges, complexities, or uncertainties encountered by team members. Supervisor play a crucial role in facilitating communication, clarifying expectations, providing training, and resolving issues to ensure that working members have the knowledge, tools, and confidence to effectively contribute to the CRMS's implementation and utilization. Supervisor can enhance the team's capabilities, address informational gaps, and promote the successful integration and optimization of the CRMS.

1.1.6 Legal Feasibility

There are no legal barriers or constraints that would prevent or prohibit the development, deployment, and operation of the CRMS within the applicable jurisdiction and regulatory framework governing the car rental industry. This assessment considers compliance with relevant laws, regulations, and standards related to data protection, privacy, consumer rights, intellectual property, and industry-specific requirements.

1.2 Project Scope

A Car Rental Management System (CRMS) is a comprehensive software solution designed to streamline and automate the operations of a car rental company. The scope of such a system encompasses various functional areas and features. Below is an outline of the project scope for a Car Rental Management System

System Overview:

The main objective is to develop a comprehensive Car Rental Management System that will enable users to rent vehicles easily and efficiently by creating a user-friendly and responsive website. This system will provide a centralized and convenient platform for users to browse, select, and rent vehicles of their choice while also offering administrators the tools to manage such as reservations, fleet management, customer management, billing, and reporting.

Key Functional Modules:

a. **Vehicle Management:**

- Add, edit, and delete vehicle details (model, category, number, etc.)
- Track vehicle availability and status (rented, available, under maintenance)
- Maintain vehicle maintenance and service records

b. **Reservation & Booking:**

- Online booking portal for customers
- Reservation management (view, modify, cancel)
- Availability calendar and real-time booking status

c. **Customer Management:**

- Customer registration and profile management
- Maintain a history of customer transactions

d. **Billing & Invoicing:**

- Generate invoices for rentals and reservations
- Integration with payment gateways for online payments
- Track payment status

e. **Reporting & Analytics:**

- Generate various reports (financial, operational, customer analytics)
- Dashboard for real-time insights and performance metrics

Integration:

- API integrations with third-party services for online payment processing

Security & Compliance:

- Data encryption and secure storage
- Regular backups and disaster recovery planning

User Experience:

- Intuitive user interface for both employees and customers

Budget & Resources:

- Estimate the budget requirements, including development costs

Quality Assurance & Testing:

- Implement testing protocols to ensure system reliability, security, and performance

1.3 Project Costing – Function Point Analysis

Cost estimation can be done by just one methodology: Function Point Analysis

Function Point Analysis can provide a mechanism to track and monitor scope creep. Function Point counts at the end of requirements; analysis, design, code, testing and implementation can be compared.

Information domain values are defined in the following manner:

- **Number of user inputs:**
  Each user input that provides distinct application-oriented data to the software.
- **Number of user outputs:**
  Each user output that provides application-oriented information to the user.
- **Number of user inquiries:**
  An on-line input that results in the generation of some immediate software response in the form of an on-line output.
- **Number of files:**
  Each logical master file (database objects)
- **Number of external interfaces:**
  All the machine-readable interfaces that are used to transmit information to another system.

Function Point Estimation

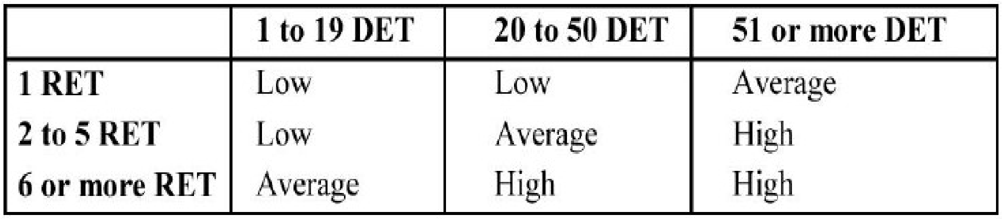
DET – Data Element Type is a unique user recognizable, non-repetitive field.

FTR – File Type Referenced is a file type referenced by a transaction. An FTR must also be either an Internal or External file.

RET – Record Element Type is a user recognizable sub group of data elements within an Internal or External File.

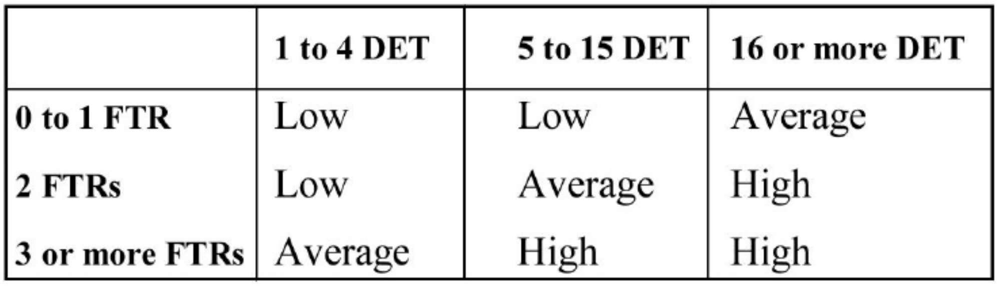
25 DETs, 10 RETs and 2 FTRs are required in the system.

Complexity Matrix used for ILF and EIF



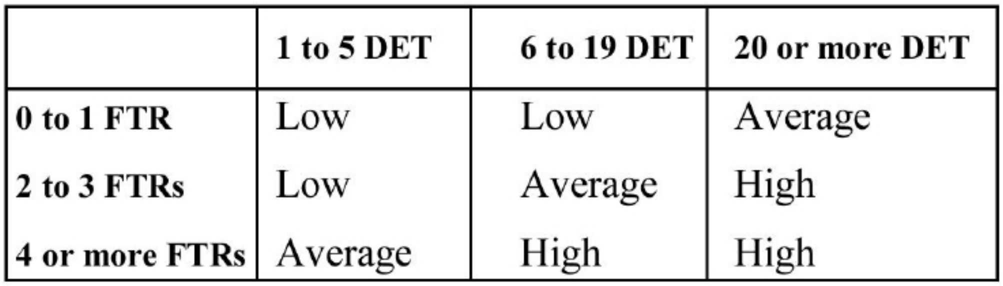
	1 to 19 DET	20 to 50 DET	51 or more DET
1 RET	Low	Low	Average
2 to 5 RET	Low	Average	High
6 or more RET	Average	High	High

Complexity Matrix used for EI



	1 to 4 DET	5 to 15 DET	16 or more DET
0 to 1 FTR	Low	Low	Average
2 FTRs	Low	Average	High
3 or more FTRs	Average	High	High

Complexity Matrix used for EO and EQ

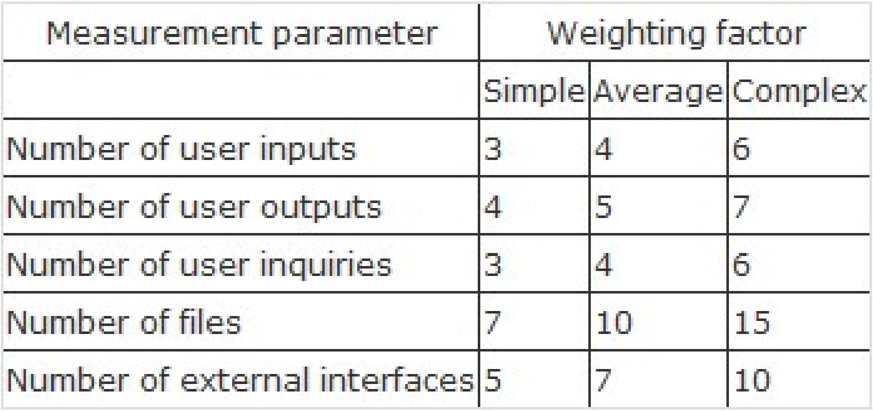


	1 to 5 DET	6 to 19 DET	20 or more DET
0 to 1 FTR	Low	Low	Average
2 to 3 FTRs	Low	Average	High
4 or more FTRs	Average	High	High

From above tables we can compute:

- Internal Logical Files (ILF): High Complexity
- External Interface Files (EIF): Low Complexity
- External Inputs (EI): Average Complexity
- External Outputs (EO): Average Complexity
- External Inquiries (EQ): Low Complexity

Based on complexity, the weighing factors for each parameter are given as:



Measurement parameter	Weighting factor		
	Simple	Average	Complex
Number of user inputs	3	4	6
Number of user outputs	4	5	7
Number of user inquiries	3	4	6
Number of files	7	10	15
Number of external interfaces	5	7	10

Function points are computed by completing the table shown in the figure below. Five information domain characteristics are determined and counts are provided in the appropriate table location.

CAR RENTAL MANAGEMENT SYSTEM						
Domain Characteristics	Count		Weighting Factor			Count
			Simple	Average	Complex	
Number Of User Inputs	15	**X**		**4**		60
Number Of User Outputs	12	**X**		**5**		60
Number Of User Enquiries	10	**X**	**3**			30
Number Of Files	10	**X**			**15**	150
Number Of External Interfaces	01	**X**	**5**			5
Count Total						305

FP est. = Count Total * [0.65 + 0.01 * (Fi)]

Where (Fi) is value adjustment factor (VAF) is based on 14 general system characteristics (GSC's) that rate the general functionality of the application being counted. Each characteristic has associated descriptions that help determine the degrees of influence of the characteristics. The degrees of influence range on a scale of zero to five, from no influence to strong influence.

1.	Data communications	(3)
2.	Distributed data processing	(1)
3.	Performance	(3)
4.	Heavily used configuration	(2)
5.	Transaction rate	(3)
6.	On-Line data entry	(4)
7.	End-user efficiency	(2)
8.	On-Line update	(5)
9.	Complex processing	(2)
10.	Reusability	(2)
11.	Installation ease	(3)
12.	Backup and Recovery	(0)
13.	Multiple sites	(4)
14.	Facilitate change	(1)

So, FP est. = 305 * [0.65 + 0.01 * (3+1+3+2+3+4+2+5+2+3+2+0+4+1)] = 305

Given that:

Average Productivity = 25 FP/person-month

Labor Cost = Rs. 4,500/month

So,

Cost per FP = Labor Cost / Average Productivity

= Rs. 180 per FP

Project Cost = FP est. * Cost per FP = Rs. 54,900

Estimated Effort = FP est. / Average Productivity

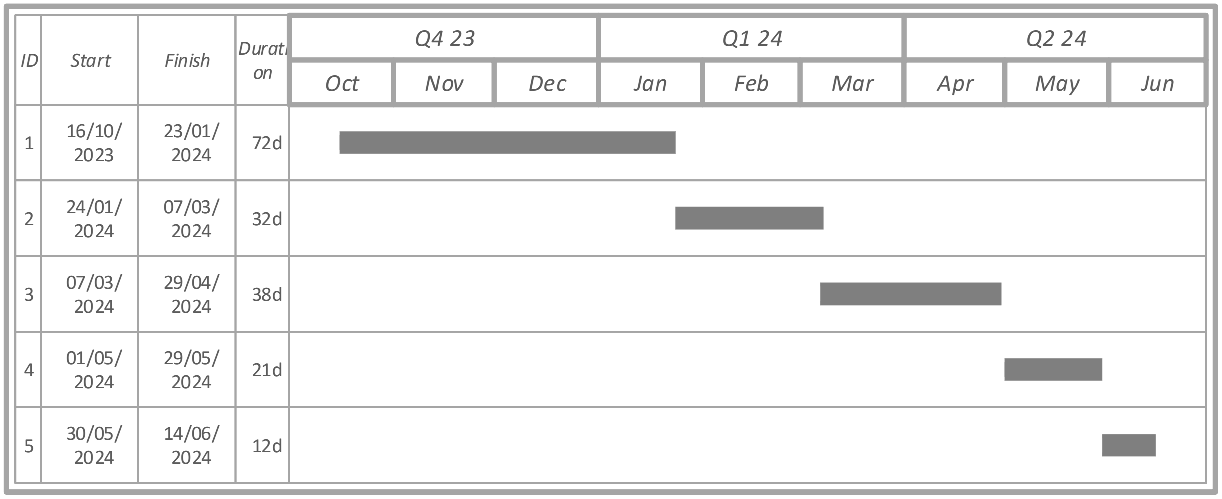
= 12.26 person-month

1.4 Gantt chart

The project's progress is represented on something like a Gantt chart. It connects with the customer and provides the project's anticipated completion date, determining the resources required, and planning the sequence in which tasks will be completed.

Tasks ID:

1. Requirement Analysis and Planning
2. Frontend Development
3. Backend Development
4. System Testing
5. Deployment and Installation



ID	Start	Finish	Durati on
1	16/10/ 2023	23/01/ 2024	72d
2	24/01/ 2024	07/03/ 2024	32d
3	07/03/ 2024	29/04/ 2024	38d
4	01/05/ 2024	29/05/ 2024	21d
5	30/05/ 2024	14/06/ 2024	12d

Gantt Chart of the Project

1.5 CPM - Critical Path Method

CPM provides the following benefits:

- Provides a graphical view of the project.
- Predicts the time required to complete the project.
- Shows which activities are critical to maintaining the schedule and which are not.

CPM models the activities and events of a project as a network. Activities are depicted as nodes on the network and events that signify the beginning or ending of activities are depicted as arcs or lines between the nodes.

Determining the following six parameters for each activity which can identify the critical path:

- **ES:** Earliest Start Time: The earliest time at which the activity can start given that its precedent activities must be completed first.
  ES (K)= max [EF(J) : J is an immediate predecessor of K]

- **EF:** Earliest Finish Time: Equal to the earliest start time for the activity plus the time required to complete the activity.
  EF (K)= ES (K) + Dur (K)

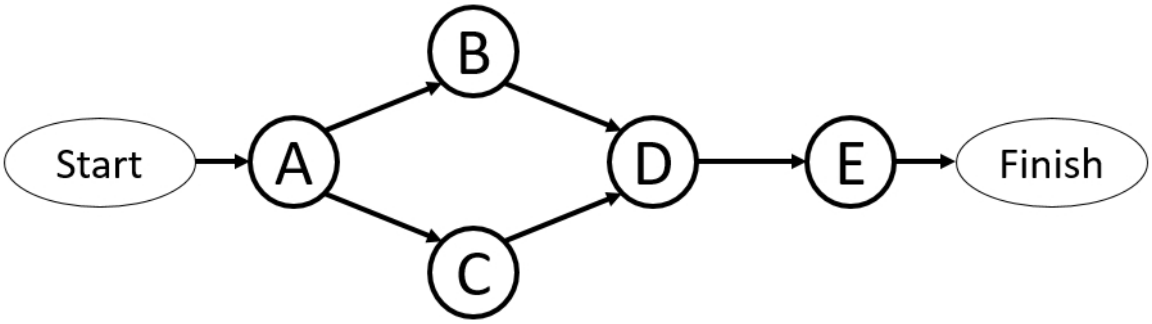
- **LF:** Latest Finish Time: The latest time at which the activity can be completed without delaying the project.
  LF (K)= min [LS(J) : J is a successor of K]

- **LS:** Latest Start Time: Equal to the latest finish time minus the time required to complete the activity.
  LS (K)= LF(K) – Dur (K)

- **TS:** Total Slack: The time that the completion of an activity can be delayed without delaying the end of the project
  TS (K)= LS(K) – ES(K)

Activity	Description	Immediate Predecessors	Duration (weeks)
A	Requirement Analysis and Planning	-	10
B	Frontend Development	A	4
C	Backend Development	A	5
D	System Testing	B,C	3
E	Deployment and Installation	D	2

Car Rental Management System (ID: BSIT-2011)



Network Diagram

Activity	Time requirement	Early Start (ES)	Early Finish (EF)	Late Start (LS)	Late Finish (LF)	Total Slack (TS)
A	10	0	10	0	10	0
B	4	10	14	11	15	1
C	5	10	15	10	15	0
D	3	15	18	15	18	0
E	2	18	20	18	20	0

The critical path is the longest-duration path through the network. The significance of the critical path is that the activities that lie on it cannot be delayed without delaying the project

The critical path is:

A -> C -> E -> F

1.6 Introduction to Team Members and their Skillsets

The following team members will work on the project.

BIT20004 - Maaz Ahmad: Project Leader & Backend Developer

Maaz Ahmad is a dynamic and experienced leader who excels in guiding teams and driving projects to success. As the Project Leader and Backend Developer, Maaz plays a pivotal role in orchestrating the development process, fostering collaboration, and ensuring the delivery of a high-quality solution.

- **Leadership & Vision:** Maaz possesses a strategic mindset and a clear vision for the project, setting the direction and priorities to align with the business objectives and stakeholder expectations. With strong communication skills and a collaborative approach, Maaz engages with team members and supervisors to foster a shared understanding of goals, requirements, and milestones.

- **Technical Expertise:** Maaz has a strong focus on building robust and scalable server-side applications. Maaz is well-equipped to navigate technical challenges, design robust solutions, and optimize system performance. His deep understanding of backend development principles, data architecture, and system integration ensures that the project's technical foundation is solid and scalable.

BIT20047 - Talha Fiaz: Web Designer

Talha Fiaz is a creative and experienced Web Designer with a passion for crafting visually appealing and user-friendly interfaces. With a background in graphic design and a keen eye for aesthetics, Talha excels in translating concepts into engaging web designs and adept at creating responsive and intuitive designs that enhance the user experience. With a strong understanding of design principles, usability, and accessibility standards, Talha collaborates closely with the development team to ensure seamless integration of design elements into functional web applications.

BIT20037 - Hassan Waris: Database Administrator

Hassan Waris is a dedicated and knowledgeable Database Administrator with expertise in managing and optimizing database systems. With a solid foundation in database design, implementation, and maintenance, Hassan ensures the reliability, performance, and security of the project's data infrastructure. Proficient in SQL databases, Hassan has a deep understanding of data modeling, indexing, and backup strategies. Additionally, as an Integration Specialist, Hassan excels in handling third-party applications, including payment gateways, seamlessly integrating these services into existing systems. By implementing best practices in database administration and integration, Hassan plays a critical role in ensuring availability, scalability, and enhanced functionality, thereby supporting the project's objectives and requirements.

1.7 Tools and Technology Used

1.7.1 HTML

HTML (Hypertext Markup Language) is the foundational language of the web, responsible for structuring web pages by defining elements like headings, paragraphs, links, and images using tags and attributes. HTML provides the essential scaffolding for web content, enabling browsers to render and display information in a structured manner.

1.7.2 CSS

CSS (Cascading Style Sheets) complements HTML by controlling the presentation and styling of web pages. It allows developers to define the visual aspects of HTML elements, such as colors, fonts, layout, and responsiveness. CSS uses selectors to target specific elements and applies rules to dictate their appearance, ensuring a consistent and aesthetically pleasing user experience. Bootstrap is a CSS library that builds on the capabilities of CSS by providing a comprehensive framework that includes a responsive grid system, pre-designed UI components, and JavaScript plugins.

1.7.3 JavaScript

JavaScript (JS) is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow clientside scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. jQuery is a fast, small, and feature-rich JavaScript library that simplifies HTML document traversal, manipulation, event handling, and animation. DataTables is a powerful jQuery plugin that adds advanced interaction controls to HTML tables, including features like pagination, searching, and sorting, making it easier to manage and display large datasets dynamically on web pages.

1.7.4 PHP

PHP is a powerful server-side scripting language widely used in web development. It operates on the web server to process data and generate dynamic content before delivering it to the client's browser. PHP seamlessly integrates with HTML, allowing developers to embed PHP code within web pages to create interactive elements, handle user input through forms, interact with databases, and perform a wide range of server-side tasks.

1.7.5 phpMyAdmin

phpMyAdmin is a popular web-based tool for managing MySQL databases. It provides a user-friendly interface that allows users to interact with their MySQL databases through a web browser. It offers a range of features including table creation, data manipulation, query execution, and database export/import functionalities. phpMyAdmin is widely used for its reliability and scalability, making it a preferred choice for web applications. It is often used in conjunction with PHP to create data-driven web solutions, allowing developers to define tables, relationships, and queries to organize and manipulate data, enabling dynamic content generation and comprehensive database management.

1.7.6 XAMPP:

XAMPP, which stands for Cross-Platform (X), Apache (A), MySQL (M), PHP (P), and Perl (P), is a free and open-source web server solution stack. It provides a convenient platform for local web development and testing on Windows. XAMPP includes Apache as the web server, MySQL as the database server, PHP as the scripting language, and Perl as a programming language. This stack facilitates the development of dynamic websites and web applications in a local environment before deploying them to a production server. XAMPP's ease of installation and configuration makes it a popular choice for developers working with PHP-based projects.

1.7.7 Visual Studio Code

Visual Studio Code is a free and open-source source-code editor developed by Microsoft. It supports various programming languages, including PHP, HTML, CSS, and JavaScript. VS Code provides a lightweight yet powerful environment for developers, featuring built-in syntax highlighting, IntelliSense code completion, debugging capabilities, and an extensive extension marketplace. Its user-friendly interface and cross-platform compatibility make it a popular choice for web developers seeking a versatile and customizable code editor.

1.7.8 JazzCash Sandbox Merchant

The JazzCash Sandbox for merchants is a simulated testing environment that allows businesses to integrate and trial JazzCash payment solutions without real financial transactions. It provides API integration, enables testing of various transaction scenarios like payments and refunds, ensures secure and compliant processes, and offers detailed error handling support. This sandbox environment helps developers refine their systems risk-free and includes comprehensive documentation and support for smooth implementation.

1.8 Risk List

The Risk List is designed to capture the perceived risks to the success of the project. It identifies the events that could lead to a significant negative outcome. It serves as a focal point for project activities and is the basis around which iterations are organized

1.8.1. The Project may require more time to complete.

- **Complexity of Requirements:** The project's scope, technical requirements, or integration needs may be more complex than initially anticipated. As the team delves deeper into the project, they may uncover additional features, dependencies, or challenges that extend the timeline.

- **Resource Constraints:** Limited availability of resources, such as skilled personnel, specialized equipment, or third-party services, could impact the project's progress. If the necessary resources are not allocated or are not readily available, it may result in delays.
- **Unforeseen Challenges:** Unexpected issues, such as technical setbacks, regulatory compliance requirements, or changes in business priorities, could arise during the project lifecycle. Addressing these unforeseen challenges may require additional time and effort to navigate and resolve effectively.
- **Stakeholder Expectations:** Managing and aligning stakeholder expectations, feedback, and approvals throughout the project can influence the timeline. Delays in decision-making, scope changes, or revisions to requirements may contribute to extending the project duration.

1.8.2. Implementing Online Payment Processing may be difficult.

- **Technical Challenges:** Integrating online payment processing involves complex technical considerations, such as security protocols, encryption methods, and compliance with payment industry standards. Ensuring seamless and secure transactions, handling different payment methods can present technical hurdles.
- **Regulatory Compliance:** Implementing online payment processing requires adherence to regulatory requirements and financial regulations. Ensuring compliance with laws related to data protection, consumer rights, and financial transactions adds complexity to the implementation process.
- **Security Concerns:** Safeguarding sensitive financial data and protecting against fraud, data breaches, or cyber-attacks is paramount in online payment processing. Implementing robust security measures, authentication mechanisms, and fraud detection systems is crucial but can be challenging and resource-intensive.
- **User Experience:** Balancing security requirements with a seamless and user-friendly payment experience is essential. Designing intuitive interfaces, optimizing transaction flows, and providing adequate support and guidance to users can be complex to achieve while maintaining security and compliance.
- **Integration & Compatibility:** Integrating online payment processing with existing systems, platforms, or third-party services within the project ecosystem may present integration challenges. Ensuring compatibility, data synchronization, and seamless communication between systems can be difficult to achieve and may require specialized expertise.

2. Requirement Engineering

Introduction

Requirements engineering process provides the appropriate mechanism for understanding what the customer wants, analyzing need, assessing feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification and managing the requirements as they are transformed into an operational system. The task of capturing, structuring, and accurately representing the user's requirements so that they can be correctly embodied in systems which meet those requirements.

- Requirements elicitation
- Requirements analysis and negotiation
- Requirements specification
- System modeling
- Requirements validation
- Requirements management.

Here, requirements specification is to be discussed. Requirements specification would lead to the steps discussed below.

2.1 System Overview

The Car Rental Management System (CRMS) serves as a centralized and comprehensive solution designed to efficiently manage various aspects of car rental operations. This system encompasses functionalities such as Vehicle Management, Reservation & Booking, Customer Management, Billing & Invoicing. These components collectively contribute to a seamless and organized car rental process, optimizing fleet management, customer satisfaction, and financial operations within the car rental industry.

2.2 Summary Requirements

The purposed system must fulfill following requirements as follow:

2.2.1. Vehicle Management:

Vehicle Management in a Car Rental Management System involves the systematic administration of the fleet. This includes functionalities such as adding new vehicles to the system with detailed specifications, updating existing vehicle information, monitoring the availability and status of each vehicle, scheduling regular maintenance, and tracking important details like mileage and condition. Efficient Vehicle Management ensures that the fleet is well-maintained, organized, and ready for customer reservations, contributing to the overall operational effectiveness of the car rental business.

2.2.2. Reservation & Booking:

The Reservation & Booking component facilitates the customer journey in reserving and booking vehicles seamlessly. Customers can check the availability of vehicles for specific dates, choose preferred vehicle types, and make reservations. The system should provide confirmation details, including pickup locations and return instructions. Reservation & Booking streamlines the booking process, minimizes conflicts over vehicle availability, and enhances the overall customer experience, fostering customer satisfaction and loyalty.

2.2.3. Customer Management:

Customer Management focuses on handling customer-related information to personalize services and streamline interactions. It includes creating and updating customer profiles with details such as contact information, preferences, and historical rental data. Effective Customer Management ensures personalized customer experiences, targeted marketing efforts, and loyalty programs. It also aids in understanding customer behavior, allowing the business to tailor services to meet specific customer needs and preferences.

2.2.4. Billing & Invoicing:

Billing & Invoicing is a crucial component that handles the financial aspects of the car rental process. It involves generating accurate invoices based on the customer's rental activity, calculating charges for rental periods and additional services, applying discounts or promotions, and managing payment transactions. This component ensures transparent and efficient financial transactions, contributing to the financial health of the car rental business and maintaining positive customer relationships.

2.2.5. Reporting & Analytics:

Reporting & Analytics provides tools for generating insights and analyzing data within the CRMS. It includes features for creating various reports, monitoring system performance, and making data-driven decisions to optimize operations. Robust reporting capabilities empower management to track key performance indicators, identify areas for improvement, and make informed strategic decisions to enhance overall efficiency and profitability in the car rental business.

2.2.6. Inventory Management:

Inventory Management oversees the complete lifecycle of vehicles within the fleet. It involves tracking the status, location, and condition of each vehicle, managing stock levels, handling routine maintenance, and ensuring that vehicles are available and ready for rental. Efficient Inventory Management minimizes downtime, maximizes vehicle utilization, and contributes to a well-maintained and organized fleet, which is essential for delivering high-quality service to customers.

2.2.7. Integration:

Integration focuses on connecting the CRMS with external systems or services to enhance its functionality. This may include integrating with payment gateways for secure transactions. Integration ensures that the system remains adaptable, leverages external resources, and provides a seamless and feature-rich experience for both customers and internal users.

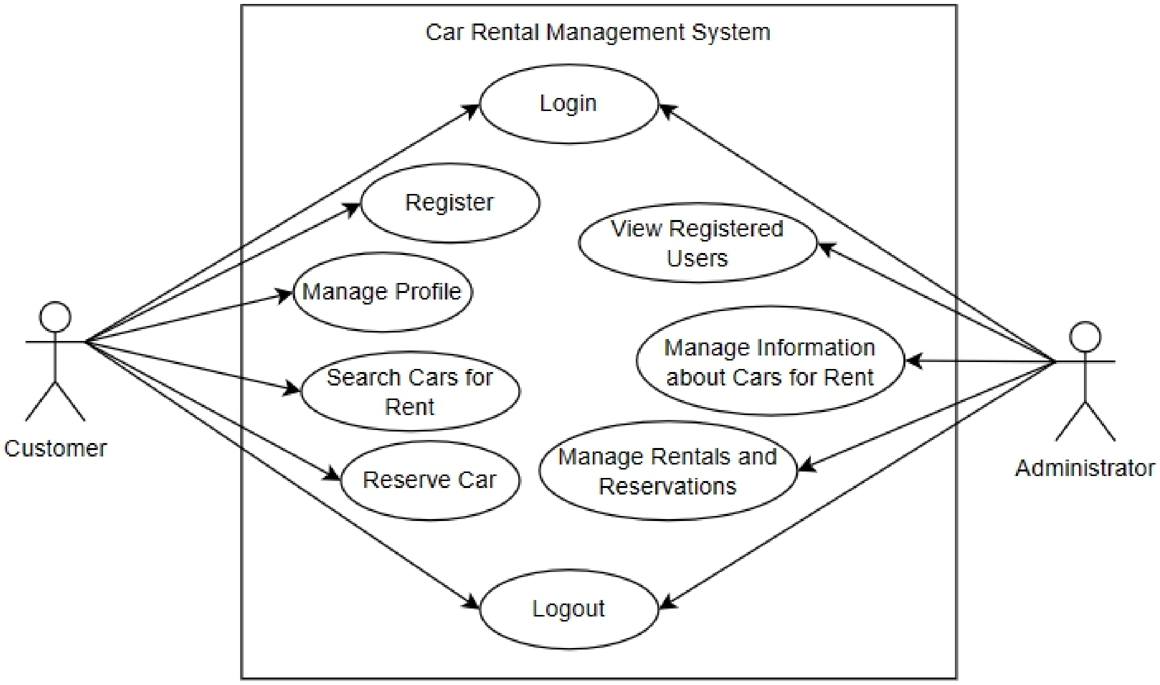
2.3 Identifying External Entities or Actors

On the basis of the System Overview and Summary Requirements, one might identify the following external entities from Car Rental Management System.

- Customer
- Administrator/Employee

2.4 Use Case Diagram

Use cases diagram capture who (actor) does what (interaction) with the system, for what purpose (goal), without dealing with system internals. A complete set of use cases specifies all the different ways to use the system, and therefore defines all behavior required of the system, bounding the scope of the system.



Use Case Diagram of Car Rental Management System

2.5 Capture "shall" Statements and the external entities

Para #	External Entity	Initial Requirements
1.0	customer	The customer shall be able to create an account with a username, contact number, unique email and password.
2.0	customer	The customer shall have the ability to update and manage their profile information, including contact details and username.
2.0		The system shall implement encryption technique to protect user data, transactions, and system integrity.
2.0	administrator	The admin shall have the ability to manage user accounts, including registration, authentication, and profile management.
1.0	customer	A customer shall login to the system and can change his password
1.0		The system shall maintain high availability, ensuring uptime and reliability for users and administrators
1.0	customer	The user shall be able to search for available vehicles based on location, and car category.
1.0	administrator	The admin shall be able to view and manage customer bookings, reservations, and transactions
1.0	customer	The user shall have the ability to reserve and book a vehicle online.
3.0	customer	Customer shall be able to make payment; either through cash or through a credit card
3.0		System shall generate invoice and confirmation receipt
2.0		The system shall integrate seamlessly with third-party services, applications, and platforms
1.0	administrator	The admin shall access various reports and analytics, including operational performance, customer insights, and inventory analysis.
1.0	administrator	The admin shall maintain an updated inventory of available vehicles, managing details, availability, and status.
1.0	administrator	The admin shall have the ability to add, edit, or remove vehicles from the system.

2.6 Allocate Requirements

Para #	Initial Requirements	Use Case Name
1.0	The customer shall be able to create an account with a username, contact number, unique email and password.	UC_Registration_Request
2.0	The customer shall have the ability to update and manage their profile information, including contact details and username.	UC_Update_Information UC_Edit_Profile
2.0	The system shall implement encryption technique to protect user data, transactions, and system integrity.	UC_Encryption
2.0	The admin shall have the ability to manage user accounts, including registration, authentication, and profile management.	UC_View_Customer_Details UC_Manage_Customer_Details
1.0	A customer shall login to the system and can change his password	UC_Login UC_Change_Password
1.0	The user shall be able to search for available vehicles based on location and car category.	UC_Search_Vehicles
1.0	The admin shall be able to view and manage customer bookings, reservations, and transactions	UC_View_Transactions
1.0	The user shall have the ability to reserve and book a vehicle online.	UC_Apply_Reservation
3.0	Customer shall be able to make payment; either through cash or through a credit card	UC_Process_Payment
3.0	System shall generate invoice and confirmation receipt	UC_Invoice_Generation
2.0	The system shall integrate seamlessly with third-party services, applications, and platforms	UC_Connect_Gateway
1.0	The admin shall access various reports and analytics, including operational performance, customer insights, and inventory analysis.	UC_View_Action
1.0	The admin shall maintain an updated inventory of available vehicles, managing details, availability, and status.	UC_Update _Status
1.0	The admin shall have the ability to add, edit, or remove vehicles from the system.	UC_Add_Vehicle UC_Edit_Vehicle UC_Remove_Vehicle

2.7 Prioritize Requirements

Para #	Priority	Initial Requirements	Use Case ID	Use Case Name
1.0	High	The customer shall be able to create an account with a username, contact number, unique email and password.	UC-3	UC_Registration_Request
2.0	High	The customer shall have the ability to update and manage their profile information, including contact details and username.	UC-4 UC-5	UC_Update_Information UC_Edit_Profile
1.0	High	A customer shall be able to log out of the system.	UC-2	UC_Logout
2.0	High	The admin shall have the ability to manage user accounts, including registration, authentication, and profile management.	UC-12 UC-13	UC_View_Customer_Details UC_Manage_Customer_Details
1.0	High	A customer shall login to the system and can change his password.	UC-1 UC-6	UC_Login UC_Change_Password
1.0	High	The user shall be able to search for available vehicles based on location, and car category.	UC-7	UC_Search_Vehicles
1.0	High	The admin shall be able to view and manage customer bookings, reservations, and transactions	UC-15	UC_View_Transactions
1.0	High	The user shall have the ability to reserve and book a vehicle online.	UC-8	UC_Apply_Reservation
3.0	Low	Customer shall be able to make payment; either through cash or through a credit card	UC-10	UC_Process_Payment
3.0	Low	System shall generate invoice and confirmation receipt	UC-11	UC_Invoice_Generation
2.0	Low	The system shall integrate seamlessly with third-party services, applications, and platforms	UC-9	UC_Connect_Gateway
1.0	Medium	The admin shall access various reports and analytics, including operational performance, customer insights, and inventory analysis.	UC-14	UC_View_Action
1.0	High	The admin shall maintain an updated inventory of available vehicles, managing details, availability, and status.	UC-16	UC_Update _Status
1.0	High	The admin shall have the ability to add, edit, or remove vehicles from the system.	UC-17 UC-18 UC-19	UC_Add_Vehicle UC_Edit_Vehicle UC_Remove_Vehicle

2.8 Requirement Traceability Matrix

Para #	Rank	Initial Requirements	Build	Use Case Name	Category
1.0	High	The customer shall be able to create an account with a username, contact number, unique email and password.	B1	UC_Registration_Request	Business
2.0	High	The customer shall have the ability to update and manage their profile information, including contact details and username.	B1	UC_Update_Information UC_Edit_Profile	Business
2.0	Medium	The system shall implement encryption technique to protect user data, transactions, and system integrity.	B1	UC_Encryption	Business
2.0	High	The admin shall have the ability to manage user accounts, including registration, authentication, and profile management.	B1	UC_View_Customer_Details UC_Manage_Customer_Details	Business
1.0	High	A customer shall login to the system and can change his password	B1	UC_Login UC_Change_Password	Business
1.0	High	The user shall be able to search for available vehicles based on location, and car category.	B1	UC_Search_Vehicles	Business
1.0	High	The admin shall be able to view and manage customer bookings, reservations, and transactions.	B1	UC_View_Transactions	Business
1.0	High	The user shall have the ability to reserve and book a vehicle online.	B1	UC_Apply_Reservation	Business
3.0	Low	Customer shall be able to make payment; either through cash or through a credit card	B1	UC_Process_Payment	Business
3.0	Low	System shall generate invoice and confirmation receipt	B1	UC_Invoice_Generation	Business
2.0	Low	The system shall integrate seamlessly with third-party services, applications, and platforms	B1	UC_Connect_Gateway	Business

1.0	Medium	The admin shall access various reports and analytics, including operational performance, customer insights, and inventory analysis.	B1	UC_View_Action	Business
1.0	High	The admin shall maintain an updated inventory of available vehicles, managing details, availability, and status.	B1	UC_Update _Status	Business
1.0	High	The admin shall have the ability to add, edit, or remove vehicles from the system.	B1	UC_Add_Vehicle UC_Edit_Vehicle UC_Remove_Vehicle	Business

3. System Design

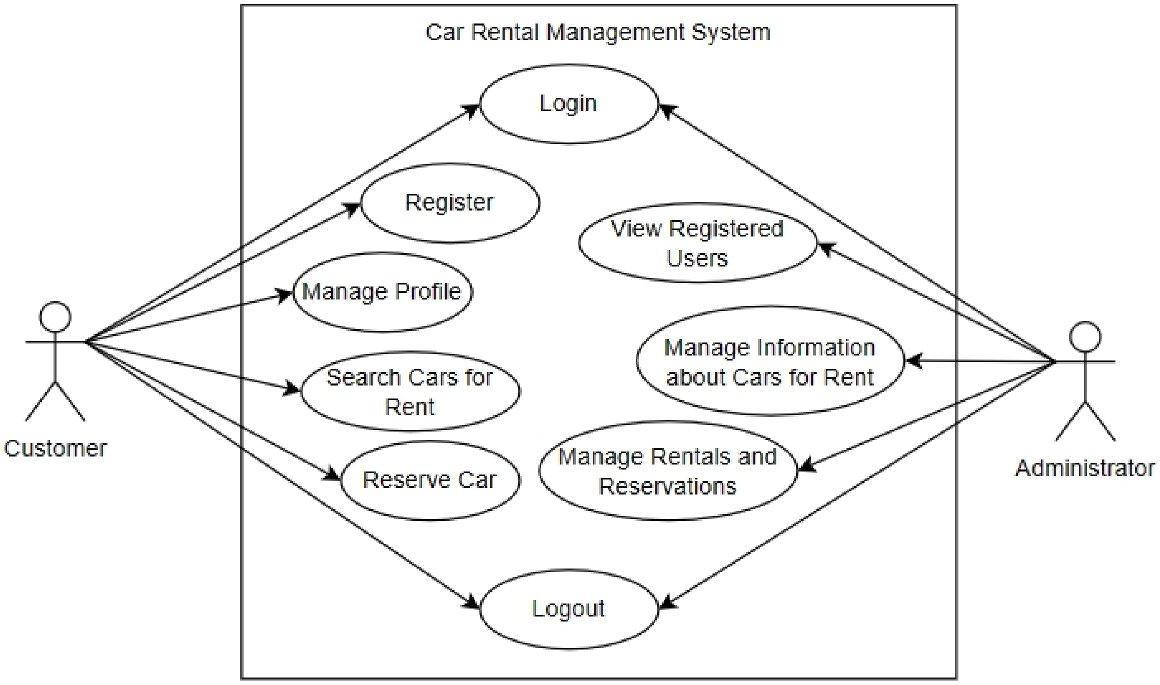
Introduction

This deliverable is all about the use case modeling and software design. In the previous deliverable, analysis of the system is completed. So we understand the current situation of the problem domain. Now we are ready to strive for a solution for the problem domain Following artifacts must be included in this deliverable.

- Use Case Diagram
- Use Case Description
- Sequence Diagram
- Collaboration Diagram
- Class Diagram
- Relational Data Model

3.1 Use Case Diagram

Use cases diagram capture who (actor) does what (interaction) with the system, for what purpose (goal), without dealing with system internals. A complete set of use cases specifies all the different ways to use the system, and therefore defines all behavior required of the system, bounding the scope of the system.



Use Case Diagram of Car Rental Management System

3.2 Use Case Description

While technically not part of UML, use case documents are closely related to UML use cases. A use case document is text that captures the detailed functionality of a use case.

Use case description typically contains the following parts:

- **Use Case Name**: The name or title of the specific use case, describing what functionality or feature is being addressed.
- **Brief Description**: A concise overview of the use case, outlining its purpose and objectives.
- **Actors**: The actors or users involved in the use case, describing who interacts with the system to achieve the specified goals.
- **Preconditions**: Conditions that must be true or events that must have occurred before the use case can be initiated.
- **Basic Flow**: The primary sequence of steps or interactions that occur within the use case to achieve its goal.
- **Alternative Flows**: Optional or alternative paths within the use case, describing deviations from the basic flow based on certain conditions or scenarios.
- **Postconditions**: The expected state of the system after the successful completion of the use case.
- **Exceptions**: Error conditions or exceptional scenarios that may occur during the execution of the use case and how they are handled.

Description of all use cases are written down.

UC-1: UC_Login

Actor	Customer, Administrator
Description	The user accesses the system to log in using their credentials.
Precondition	The user must have a registered account.
Postcondition	The user gains access to their account and can perform various actions within the system.
Basic Flow	1. The system presents a login interface. 2. The user enters their username and password. 3. The system verifies the credentials. 4. If the credentials are valid, the system grants access to the user's account.
Alternate Flow	If the credentials are invalid, the system prompts the user to re-enter their credentials or reset their password.

UC-2: UC_Logout

Actor	Customer, Administrator
Description	The user logs out of their account.
Precondition	The user must be logged in to their account.
Postcondition	The user is logged out of their account and no longer has access to restricted functionalities.
Basic Flow	1. The user navigates to the logout option within the system. 2. The system confirms the user's intent to logout. 3. The system logs out the user and redirects them to the login page.

UC-3: UC_Registration_Request

Actor	Customer
Description	A new customer creates an account in the system.
Precondition	The customer must not already have an existing account.
Postcondition	The customer successfully registers and gains access to their account.
Basic Flow	1. The customer accesses the registration page. 2. The customer fills out the required registration form, including personal information and desired login credentials. 3. The system validates the entered information, creates the account and notifies the customer of successful registration.
Alternate Flow	If the entered information is incomplete or the email already exists, the system prompts the customer to correct the errors.

UC-4: UC_Update_Information

Actor	Customer
Description	The customer can view and update their profile information.
Precondition	The customer must be logged into their account.
Postcondition	The customer's profile information is updated as desired.
Basic Flow	1. The customer navigates to the profile management section. 2. The system displays the customer's current profile information. 3. The customer selects the option to edit their profile. 4. The system presents a form with editable fields. 5. The customer makes the desired changes and submits the form such as changing password. 6. The system updates the customer's profile with the new information.

UC-5: UC_Edit_Profile

Actor	Customer
Description	The customer can view and update their profile information.
Precondition	The customer must be logged into their account.
Postcondition	The customer's profile information is updated as desired.
Basic Flow	1. The customer navigates to the profile management section. 2. The system displays the customer's current profile information. 3. The customer selects the option to edit their profile. 4. The system presents a form with editable fields. 5. The customer makes the desired changes and submits the form such as changing password. 6. The system updates the customer's profile with the new information.

UC-6: UC_Change_Password

Actor	Customer
Description	The customer can view and update their profile information.
Precondition	The customer must be logged into their account.
Postcondition	The customer's profile information is updated as desired.
Basic Flow	1. The customer navigates to the profile management section. 2. The system displays the customer's current profile information. 3. The customer selects the option to edit their profile. 4. The system presents a form with editable fields. 5. The customer makes the desired changes and submits the form such as changing password. 6. The system updates the customer's profile with the new information.

UC-7: UC_Search_Vehicles

Actor	Customer
Description	The customer searches for available cars for rent based on specified criteria.
Precondition	The customer must be logged into their account.
Postcondition	The customer then receives a list of available cars matching their search criteria.
Basic Flow	1. The customer navigates to the car rental search page. 2. The customer specifies search criteria such as location, and car type. 3. The system retrieves and displays a list of available cars matching the criteria.
Alternate Flow	If no cars are available matching the specified criteria, the system informs the customer and suggests refining the search parameters.

UC-8: UC_Apply_Reservation

Actor	Customer
Description	The customer reserves a specific car for rent.
Precondition	The customer must be logged into their account and must have selected a car from the search results.
Postcondition	The customer successfully reserves the selected car for the specified dates.
Basic Flow	1. The customer selects a car from the list of available options. 2. The customer specifies the rental dates and any additional preferences. 3. If the car is available, the system processes the payment, confirms the reservation and provides invoice to the customer.

UC-9: UC_Connect_Gateway

Actor	Customer
Description	The customer reserves a specific car for rent.
Precondition	The customer must be logged into their account and must have selected a car from the search results.
Postcondition	The customer successfully reserves the selected car for the specified dates.
Basic Flow	1. The customer selects a car from the list of available options. 2. The customer specifies the rental dates and any additional preferences. 3. If the car is available, the system processes the payment, confirms the reservation and provides invoice to the customer.

UC-10: UC_Process_Payment

Actor	Customer
Description	The customer reserves a specific car for rent.
Precondition	The customer must be logged into their account and must have selected a car from the search results.
Postcondition	The customer successfully reserves the selected car for the specified dates.
Basic Flow	1. The customer selects a car from the list of available options. 2. The customer specifies the rental dates and any additional preferences. 3. If the car is available, the system processes the payment, confirms the reservation and provides invoice to the customer.

UC-11: UC_Invoice_Generation

Actor	Customer
Description	The customer reserves a specific car for rent.
Precondition	The customer must be logged into their account and must have selected a car from the search results.
Postcondition	The customer successfully reserves the selected car for the specified dates.
Basic Flow	1. The customer selects a car from the list of available options. 2. The customer specifies the rental dates and any additional preferences. 3. If the car is available, the system processes the payment, confirms the reservation and provides invoice to the customer.

UC-12: UC_View_Customer_Details

Actor	Administrator
Description	The administrator views a list of registered users in the system.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has accessed the list of registered users.
Basic Flow	1. The administrator navigates to the section for managing registered users. 2. The system retrieves and displays a list of all registered users. 3. The administrator can view details such as usernames, contact information, etc.

UC-13: UC_Manage_Customer_Details

Actor	Administrator
Description	The administrator manages a list of registered users in the system.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has manages the list of registered users.
Basic Flow	4. The administrator navigates to the section for managing registered users. 5. The system retrieves and displays a list of all registered users. 6. The administrator can manages customer details such as usernames, contact information, etc.

UC-14: UC_View_Action

Actor	Administrator
Description	The administrator manages rentals and reservations made by customers.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has successfully managed rentals and reservations.
Basic Flow	1. The administrator logs in to the system. 2. The system displays a list of current and upcoming reservations on the main section. 3. The administrator can view details such as customer information, rental dates, and car details.

UC-15: UC_View_Transactions

Actor	Administrator
Description	The administrator view details of transactions made by customers.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has successfully managed transactions.
Basic Flow	1. The administrator navigates to the section for managing rentals and reservations. 2. The system displays a list of current and upcoming reservations. 3. The administrator can view details such as customer information, rental dates, and car details. 4. The administrator can view transaction details of each reservation.

UC-16: UC_Update_Status

Actor	Administrator
Description	The administrator manages rentals and reservations made by customers.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has successfully managed rentals and reservations.
Basic Flow	1. The administrator navigates to the section for managing rentals and reservations. 2. The system displays a list of current and upcoming reservations. 3. The administrator can view details such as customer information, rental dates, and car details. 4. The administrator can update reservation statuses as needed.

UC-17: UC_Add_Vehicle

Actor	Administrator
Description	The administrator manages information related to cars for rent.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has successfully managed information about cars for rent.
Basic Flow	1. The administrator navigates to the section for managing car rental information. 2. The system displays a list of available cars along with their details such as make, model, year, and availability status. 3. The administrator can add new cars to the system by entering relevant details. 4. The administrator can update existing car information, such as availability status, rental rates, or additional features. 5. The administrator can remove cars from the system if they are no longer available for rent.
Alternate Flow	If there are no cars available for rent, the administrator may need to coordinate with car suppliers or update the system accordingly.

UC-18: UC_Update_Vehicle

Actor	Administrator
Description	The administrator manages information related to cars for rent.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has successfully managed information about cars for rent.
Basic Flow	1. The administrator navigates to the section for managing car rental information. 2. The system displays a list of available cars along with their details such as make, model, year, and availability status. 3. The administrator can add new cars to the system by entering relevant details. 4. The administrator can update existing car information, such as availability status, rental rates, or additional features. 5. The administrator can remove cars from the system if they are no longer available for rent.
Alternate Flow	If there are no cars available for rent, the administrator may need to coordinate with car suppliers or update the system accordingly.

UC-19: UC_Delete_Vehicle

Actor	Administrator
Description	The administrator manages information related to cars for rent.
Precondition	The administrator must be logged into their administrative account.
Postcondition	The administrator has successfully managed information about cars for rent.
Basic Flow	1. The administrator navigates to the section for managing car rental information. 2. The system displays a list of available cars along with their details such as make, model, year, and availability status. 3. The administrator can add new cars to the system by entering relevant details. 4. The administrator can update existing car information, such as availability status, rental rates, or additional features. 5. The administrator can remove cars from the system if they are no longer available for rent.
Alternate Flow	If there are no cars available for rent, the administrator may need to coordinate with car suppliers or update the system accordingly.

3.3 Sequence Diagram

A Sequence diagram depicts the sequence of actions that occur in a system. The invocation of methods in each object, and the order in which the invocation occurs is captured in a Sequence diagram. This makes the Sequence diagram a very useful tool to easily represent the dynamic behavior of a system.

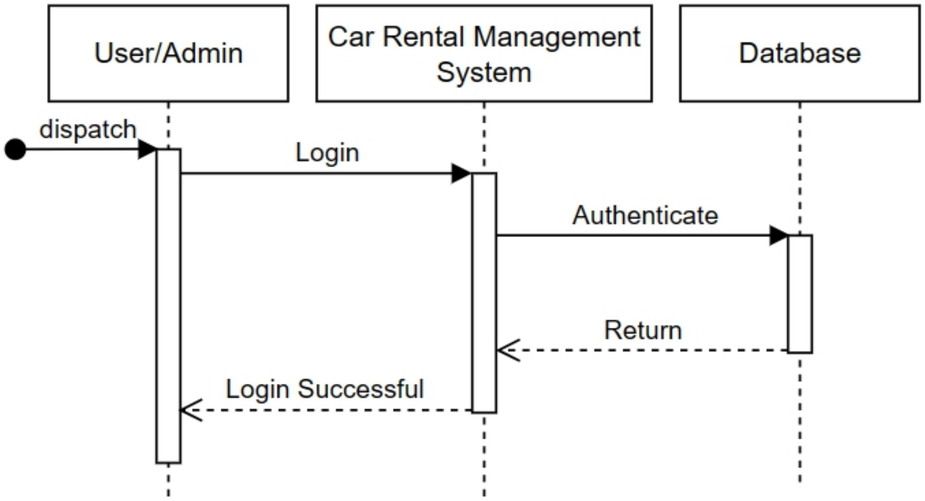
A Sequence diagram is two-dimensional in nature. On the horizontal axis, it shows the life of the object that it represents, while on the vertical axis, it shows the sequence of the creation or invocation of these objects. Because it uses class name and object name references, the Sequence diagram is very useful in elaborating and detailing the dynamic design and the sequence and origin of invocation of objects. Hence, the Sequence diagram is one of the most widely used dynamic diagrams in UML.

Sequence Diagram consists of following main elements.

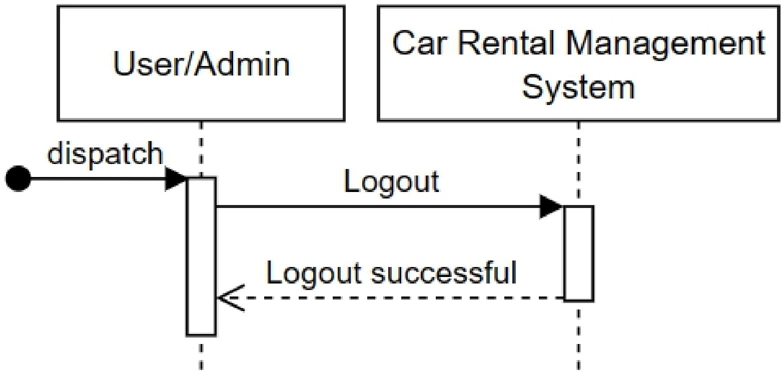
- **Objects/Participants**: Represented as boxes at the top of the diagram, indicating the various objects or participants involved in the sequence of interactions.
- **Lifelines**: Vertical lines extending from each object/participant, representing the lifespan of each object during the sequence.
- **Messages**: Arrows between lifelines, depicting the messages passed between objects. These messages show the flow of data between the participants.
- **Activation/Execution Occurrences**: Optional markers on lifelines, illustrating when an object is active or executing a particular part of the sequence.
- **Control Flows**: The order and flow of messages and interactions between objects, showing the sequence of events in the system.

Sequence diagrams of all use cases are written down.

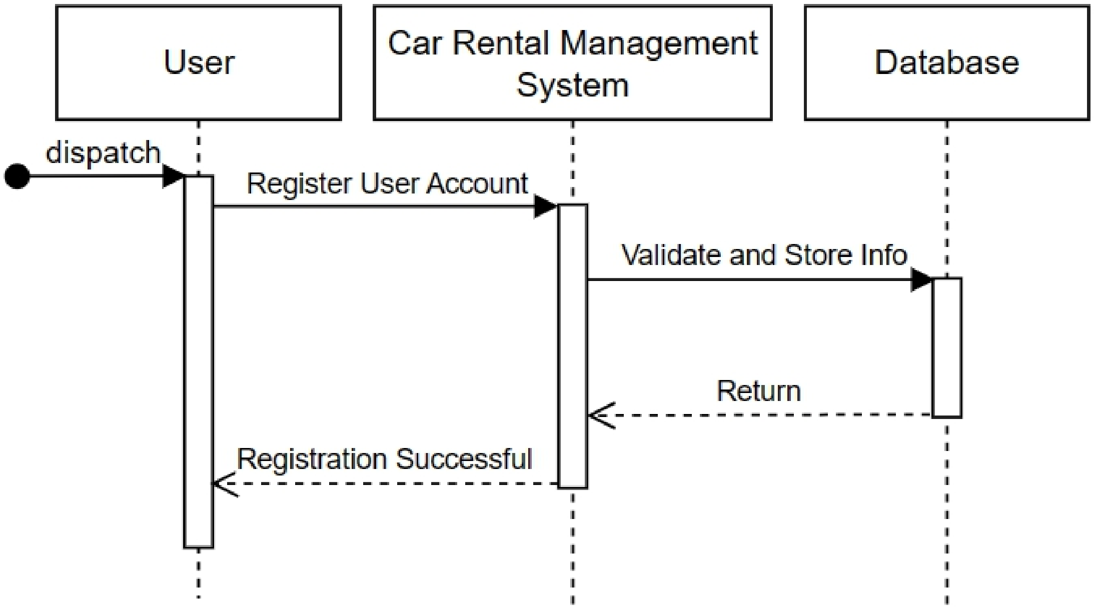
UC-1: UC_Login



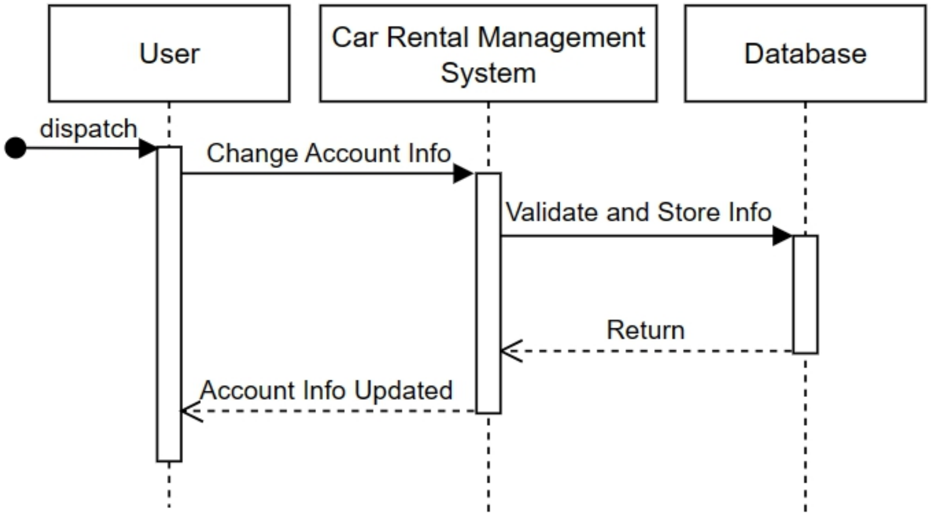
UC-2: UC_Logout



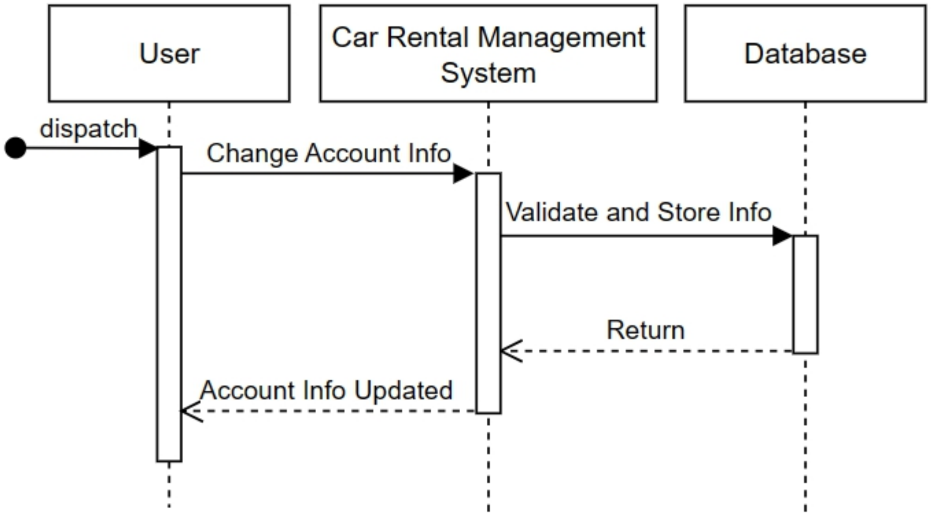
UC-3: UC_Registration_Request



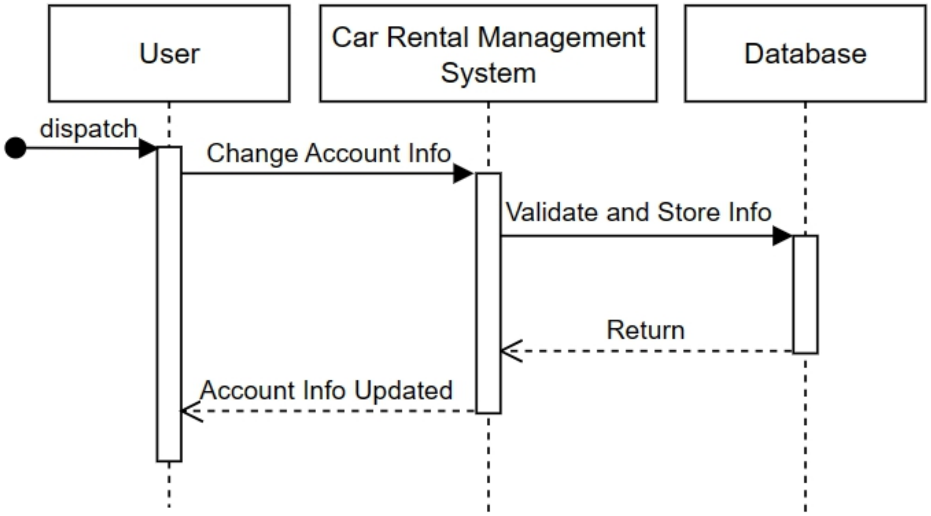
UC-4: UC_Update_Information



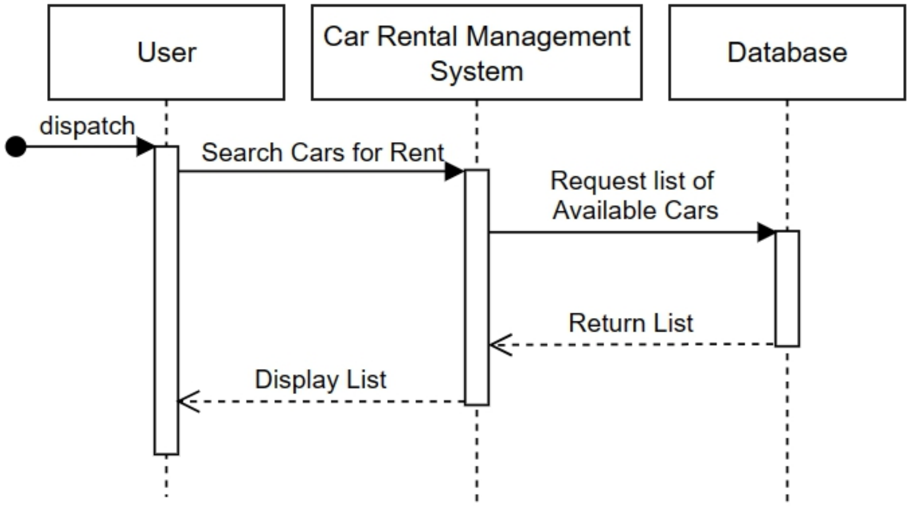
UC-5: UC_Edit_Profile



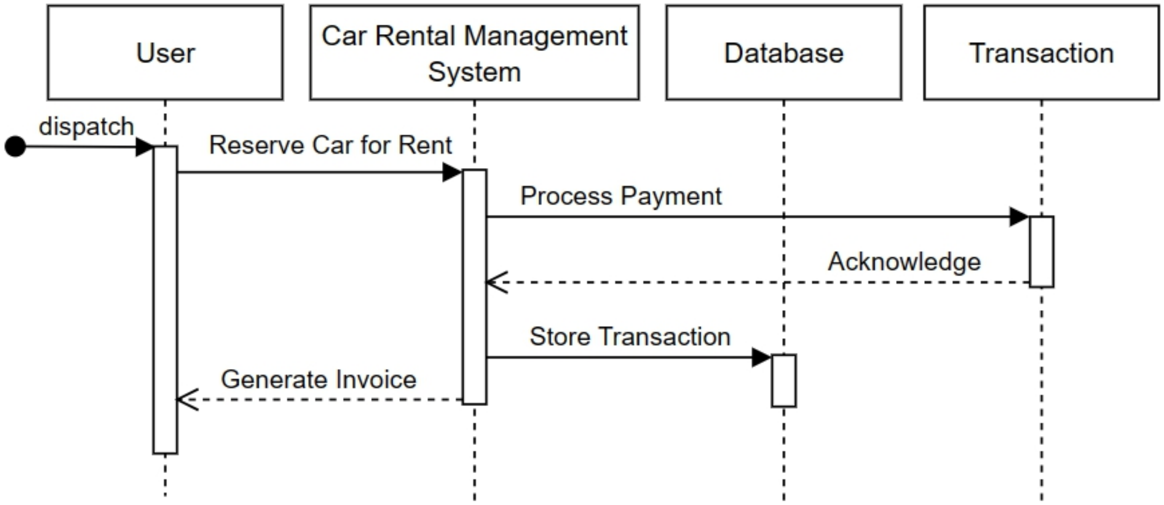
UC-6: UC_Change_Password



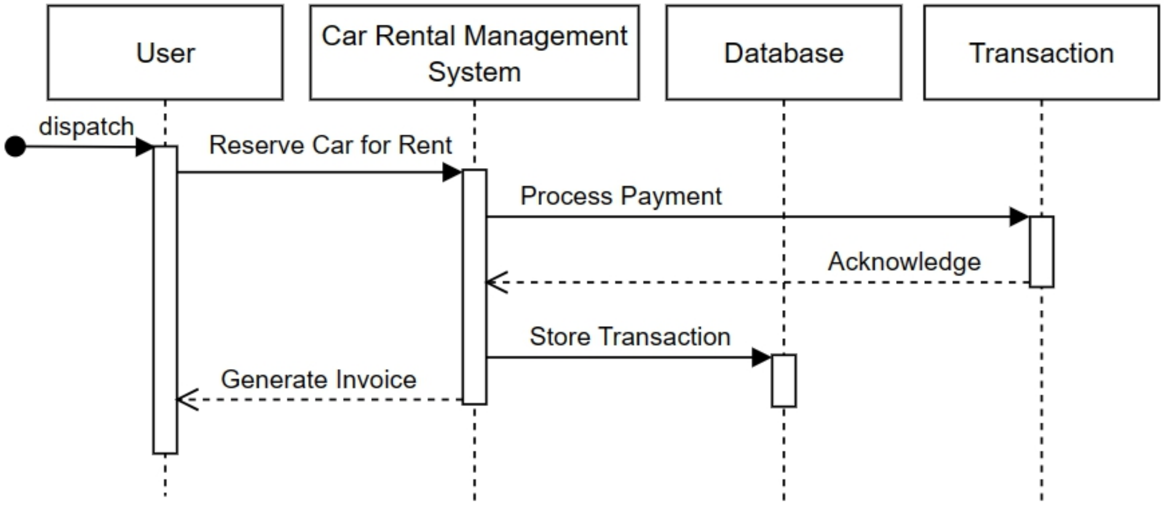
UC-7: UC_Search_Vehicles



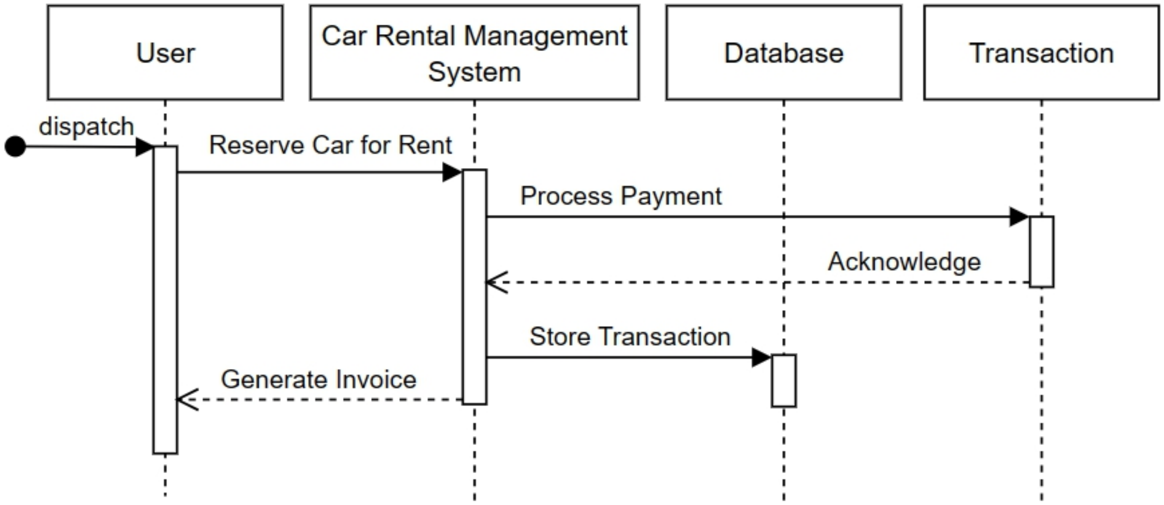
UC-8: UC_Apply_Reservation



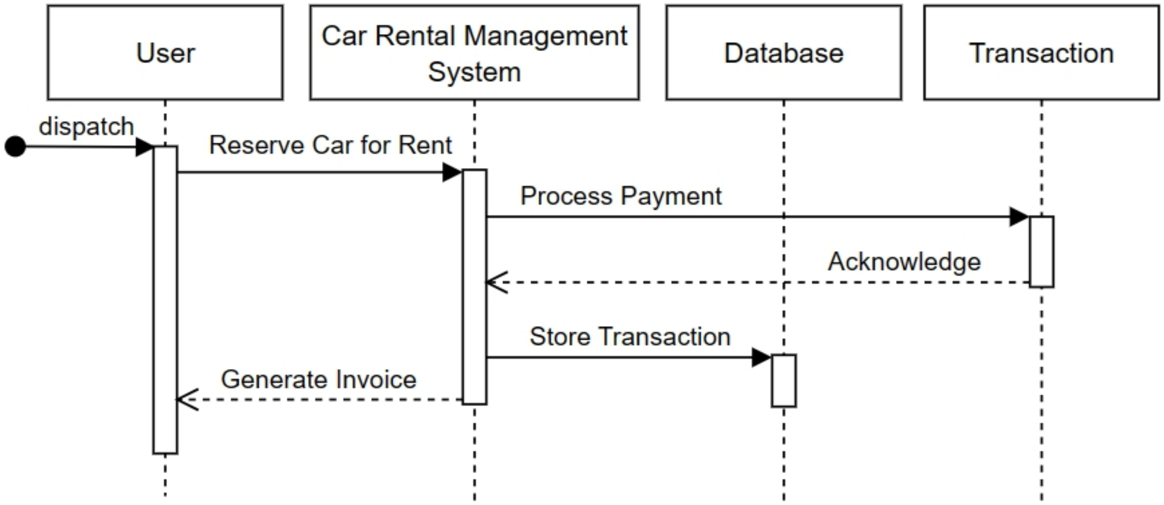
UC-9: UC_Connect_Gateway



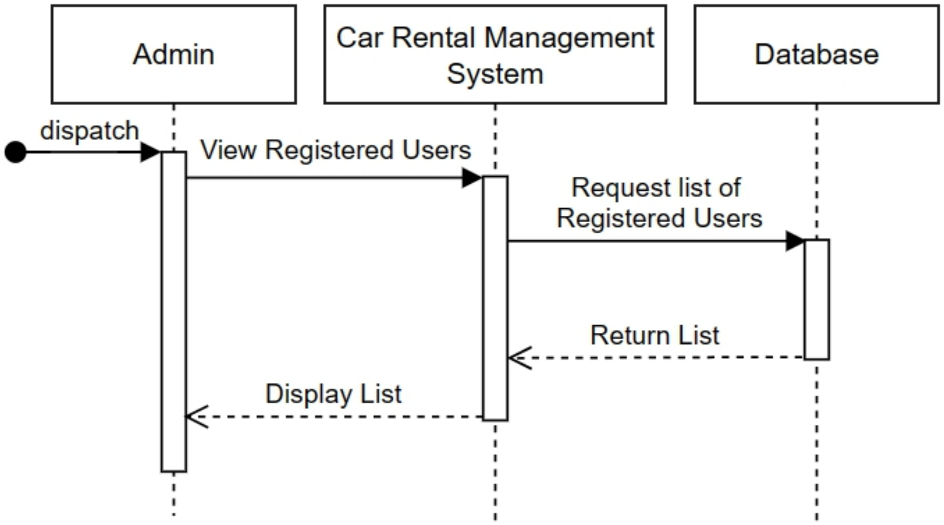
UC-10: UC_Process_Payment



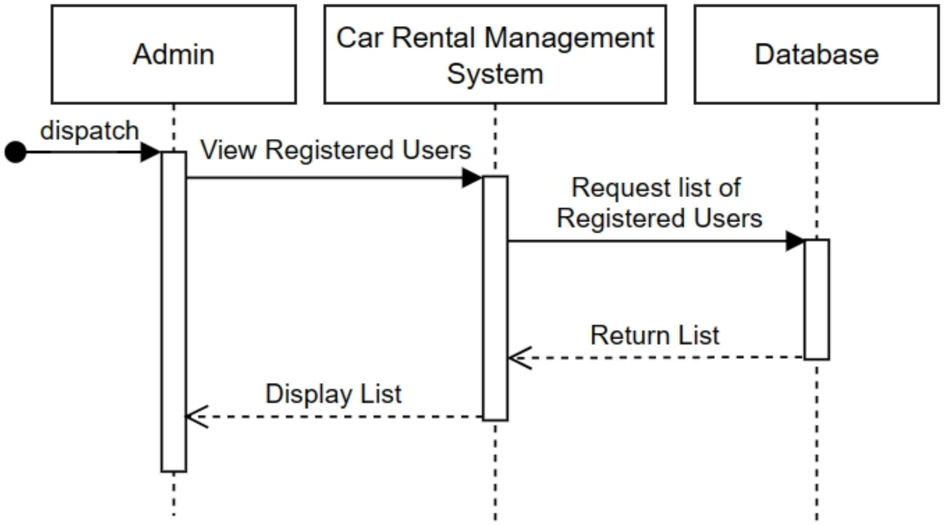
UC-11: UC_Invoice_Generation



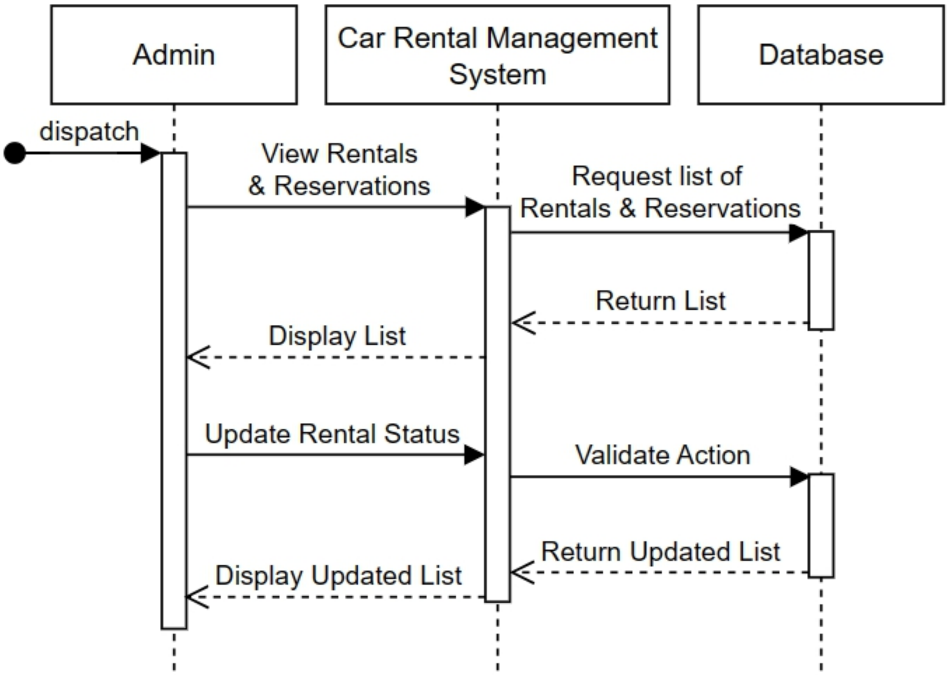
UC-12: UC_View_Customer_Details



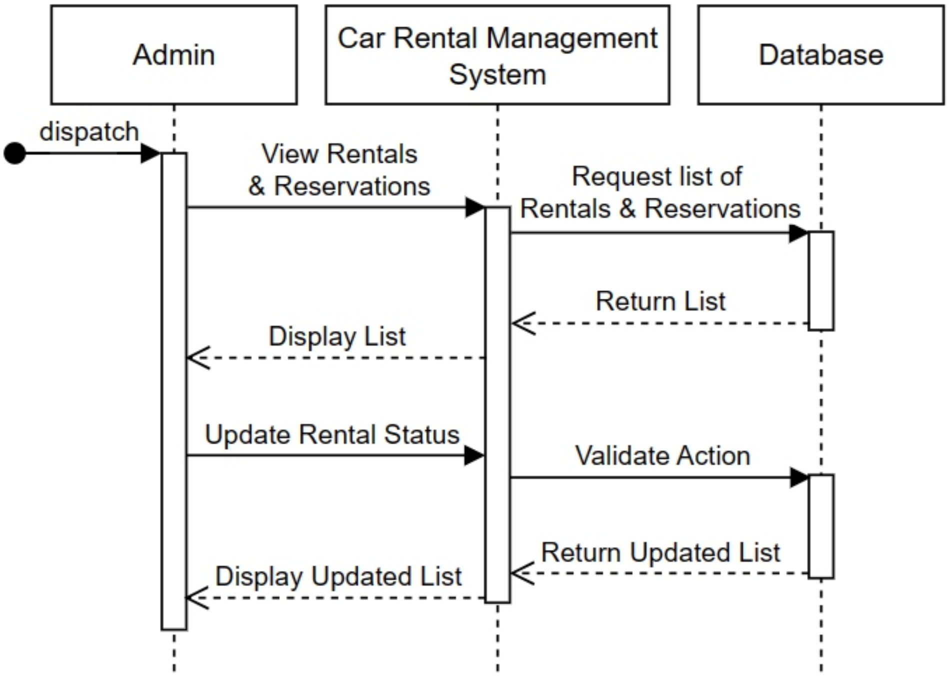
UC-13: UC_Manage_Customer_Details



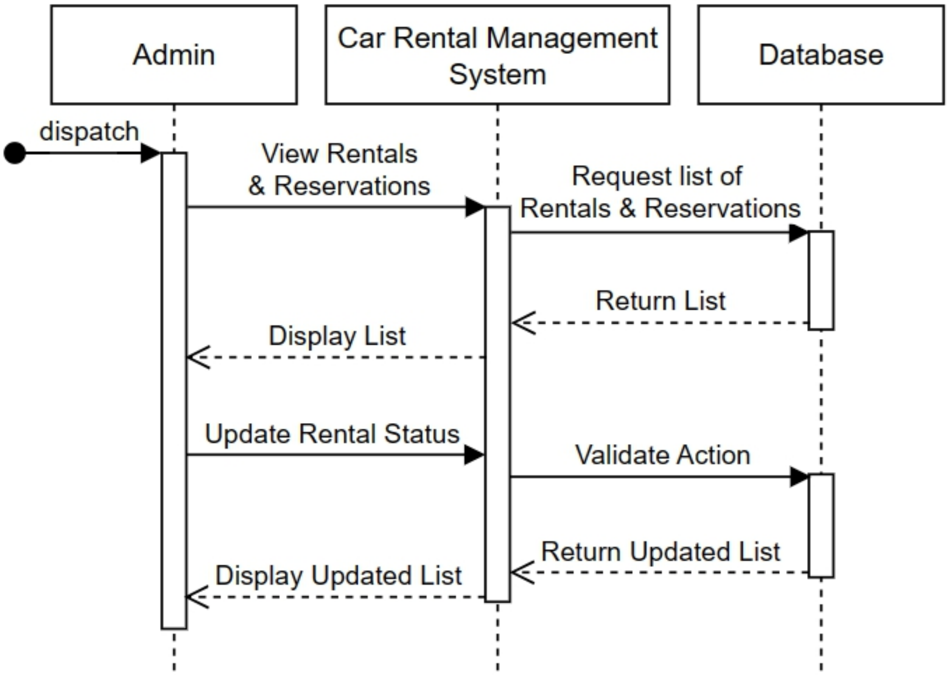
UC-14: UC_View_Action



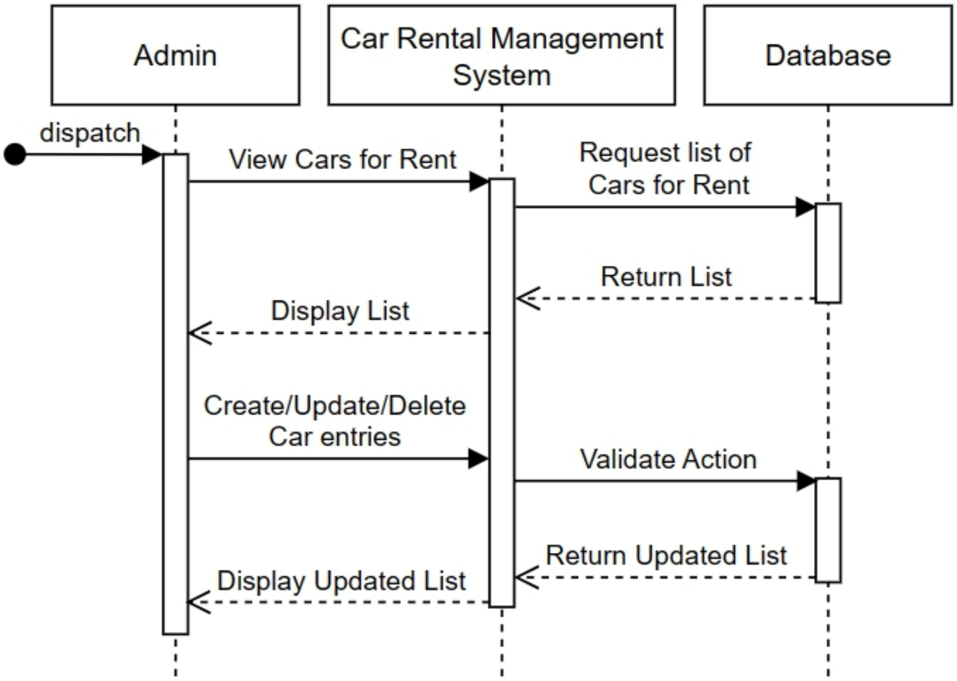
UC-15: UC_View_Transactions



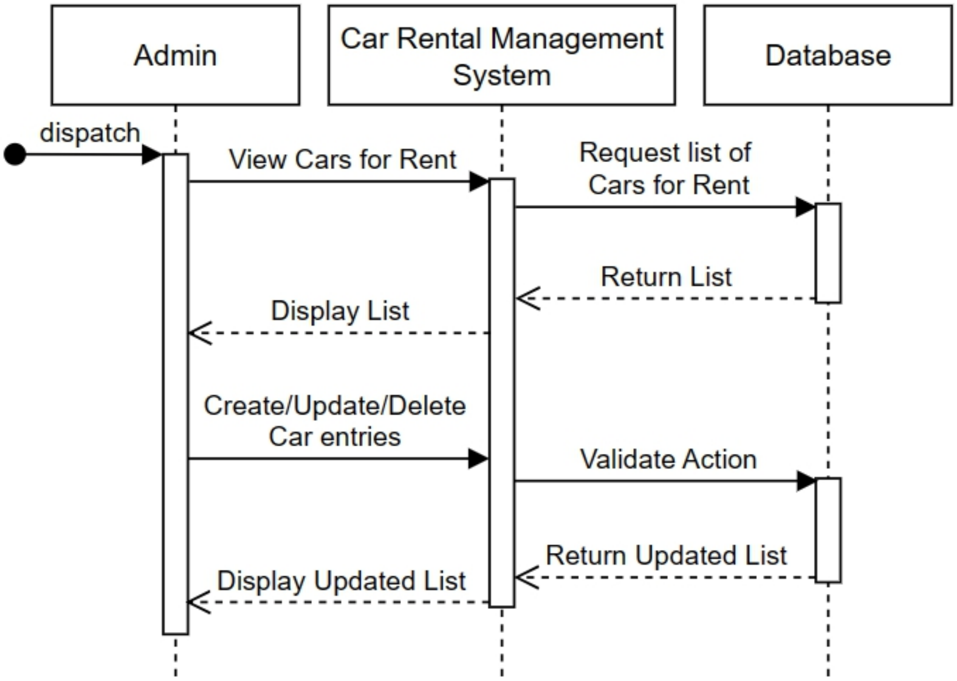
UC-16: UC_Update_Status



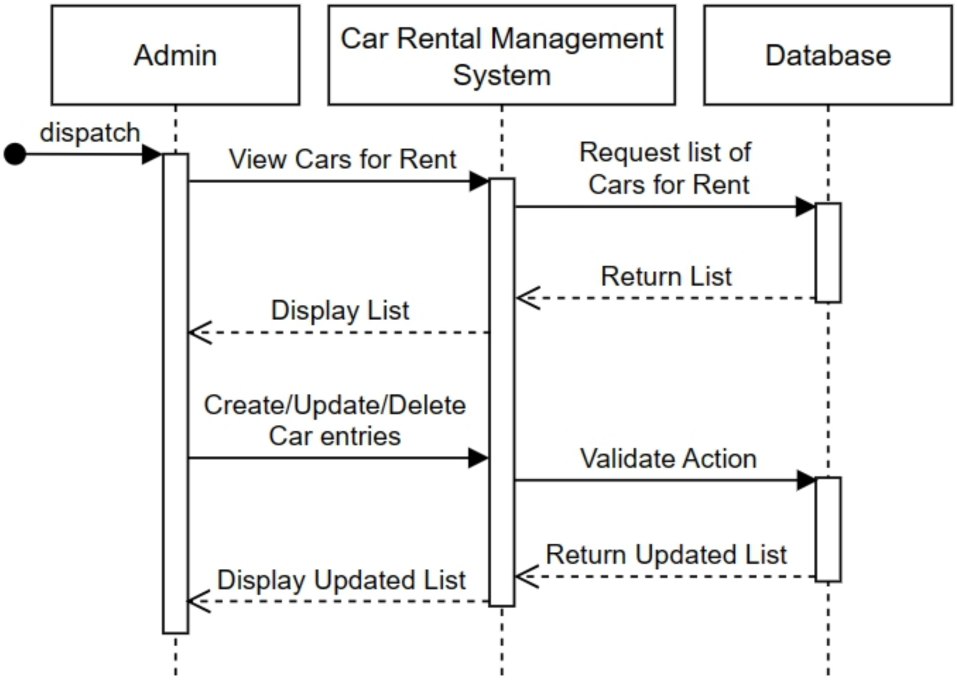
UC-17: UC_Add_Vehicle



UC-18: UC_Update_Vehicle



UC-19: UC_Delete_Vehicle



3.4 Collaboration Diagram

A collaboration diagram describes a pattern of interaction among objects; it shows the objects participating in the interaction by their links to each other and the messages that they send to each other. Collaboration diagrams are used to show how objects interact to perform the behavior of a particular use case, or a part of a use case.

Along with sequence diagrams, collaborations are used by designers to define and clarify the roles of the objects that perform a particular flow of events of a use case. They are the primary source of information used to determining class responsibilities and interfaces.

Unlike a sequence diagram, a collaboration diagram shows the relationships among the objects. Sequence diagrams and collaboration diagrams express similar information, but show it in different ways.

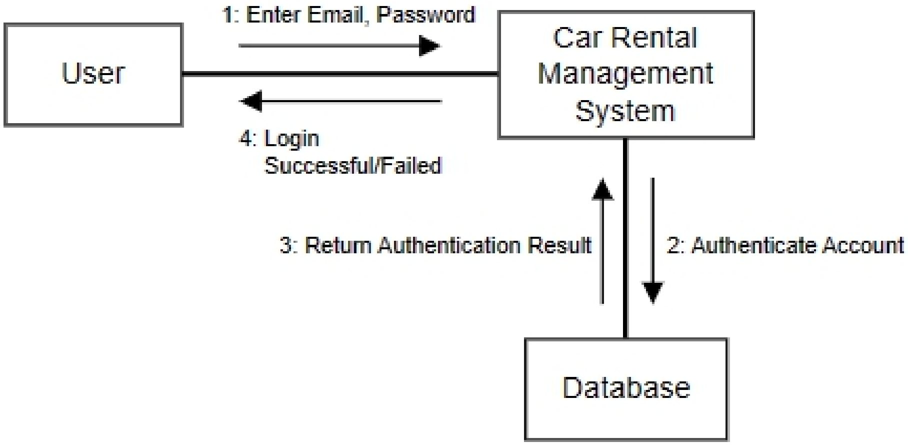
Because of the format of the collaboration diagram, they tend to better suited for analysis activities. Specifically, they tend to be better suited to depicting simpler interactions of smaller numbers of objects.

Collaboration Diagram consists of following main elements.

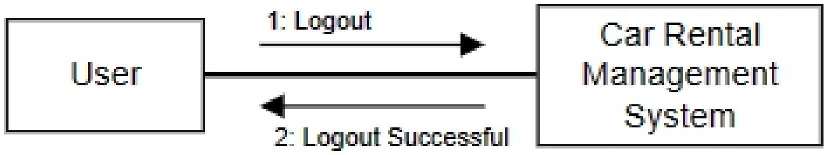
- **Objects/Participants**: Represented as boxes or rectangles, indicating the various objects or participants involved in the collaboration.
- **Links/Connections**: Arrows or lines between objects, depicting the relationships or interactions between them.
- **Roles**: Labels on the links or connections, indicating the role or purpose of the interaction between the objects.
- **Messages**: Information passed between objects, typically shown on the links or connections, representing the flow of communication or data.

Collaboration diagrams of all use cases are written down.

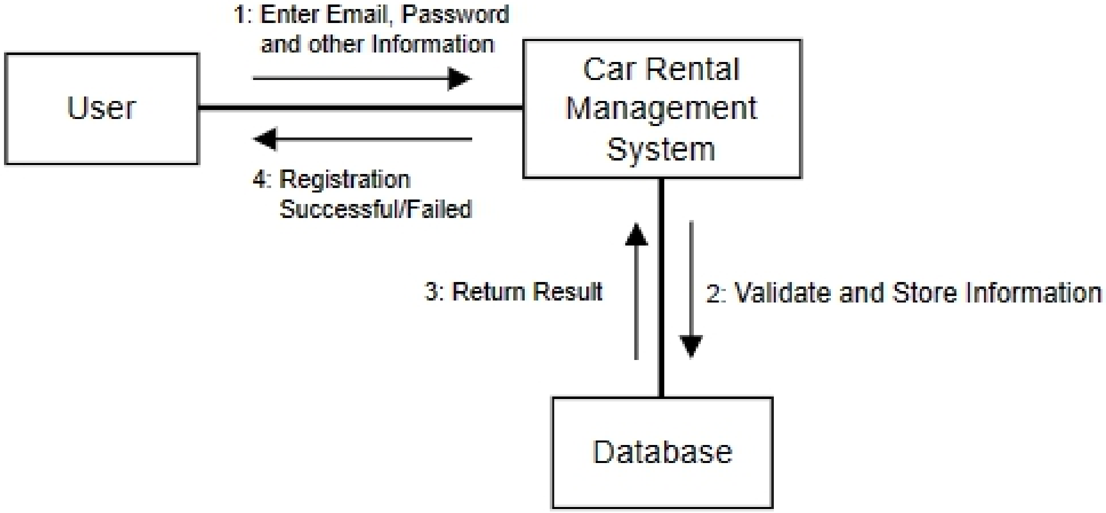
UC-1: UC_Login


1: Enter Email, Password
User
Car Rental Management System
4: Login Successful/Failed
3: Return Authentication Result
2: Authenticate Account
Database

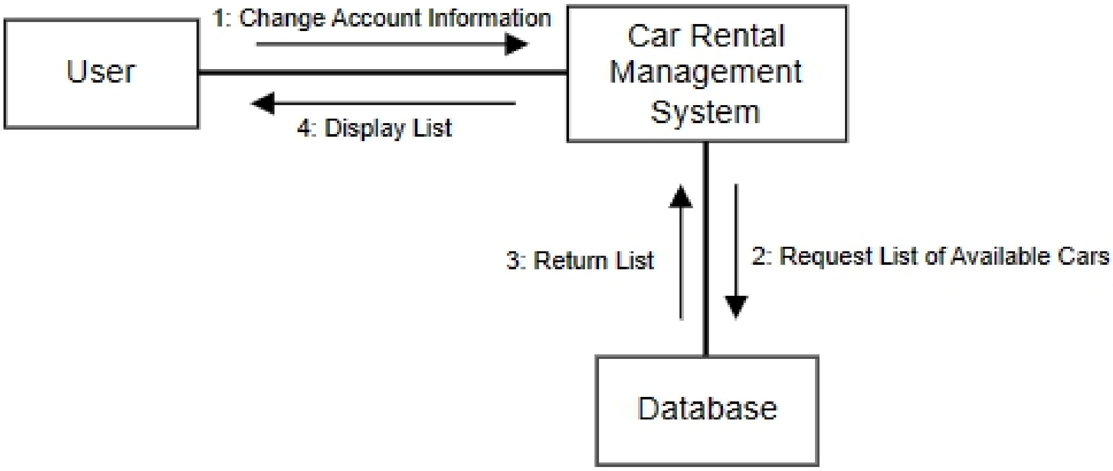
UC-2: UC_Logout



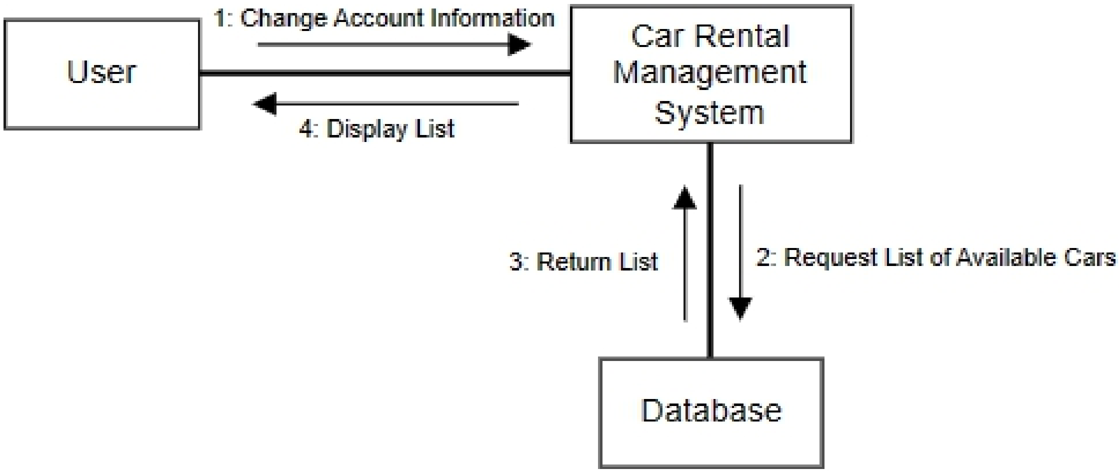
UC-3: UC_Registration_Request


User
1: Enter Email, Password and other Information
4: Registration Successful/Failed
Car Rental Management System
3: Return Result
2: Validate and Store Information
Database

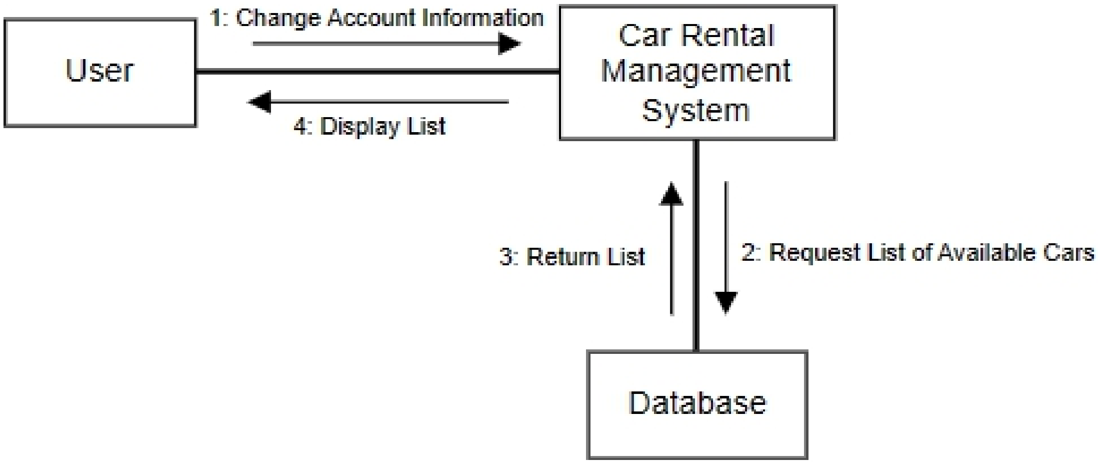
UC-4: UC_Update_Information


User
1: Change Account Information
4: Display List
Car Rental Management System
3: Return List
2: Request List of Available Cars
Database

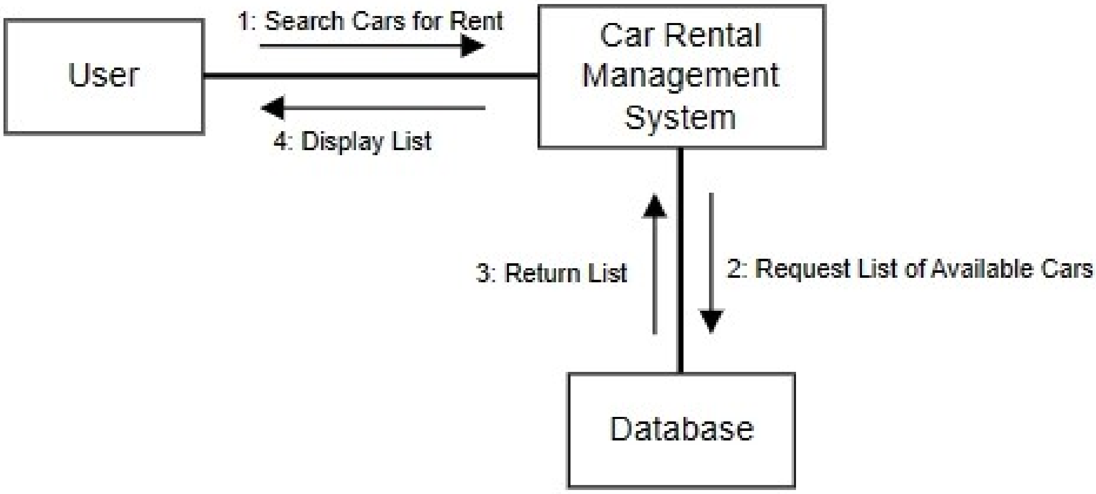
UC-5: UC_Edit_Profile


User
1: Change Account Information
4: Display List
Car Rental Management System
3: Return List
2: Request List of Available Cars
Database

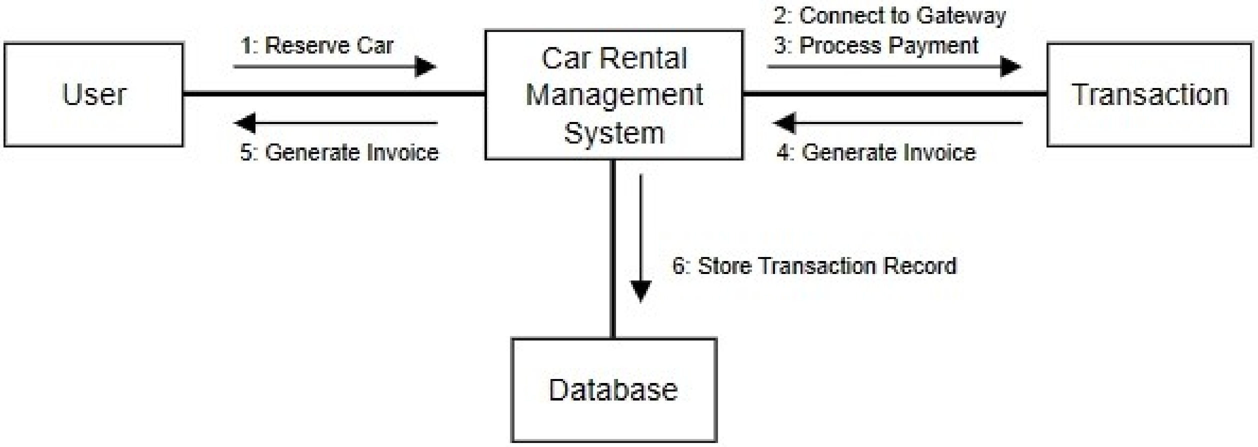
UC-6: UC_Change_Password


User
1: Change Account Information
4: Display List
Car Rental Management System
3: Return List
2: Request List of Available Cars
Database

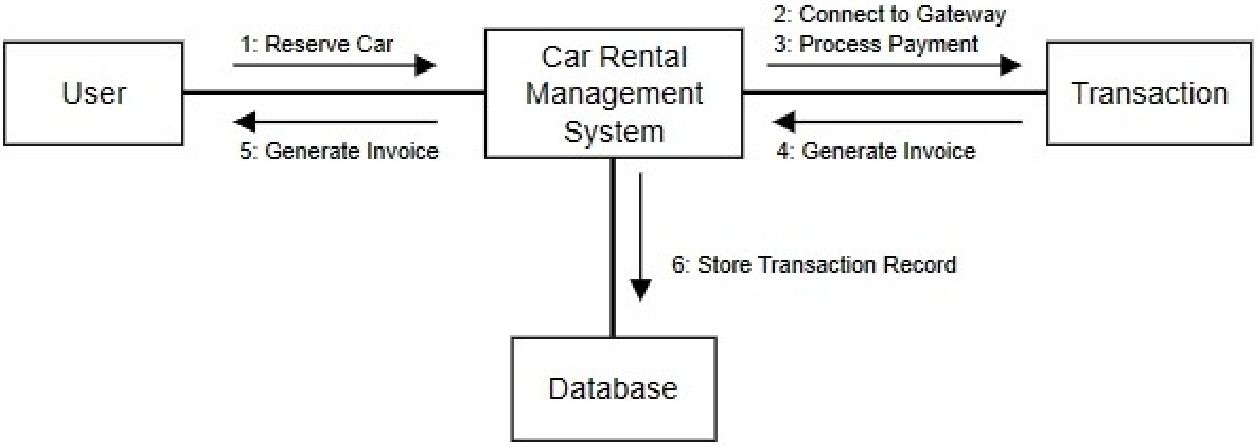
UC-7: UC_Search_Vehicles



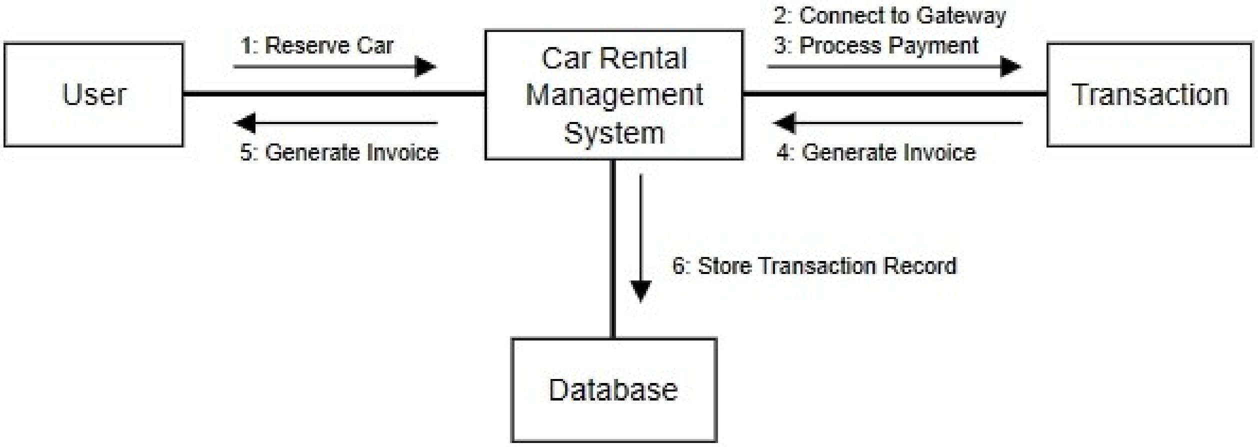
UC-8: UC_Apply_Reservation



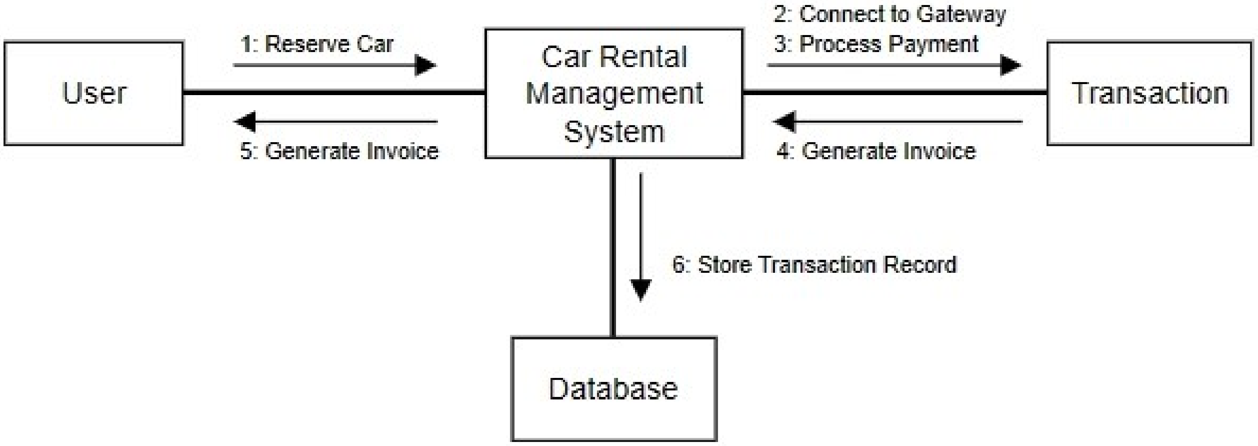
UC-9: UC_Connect_Gateway



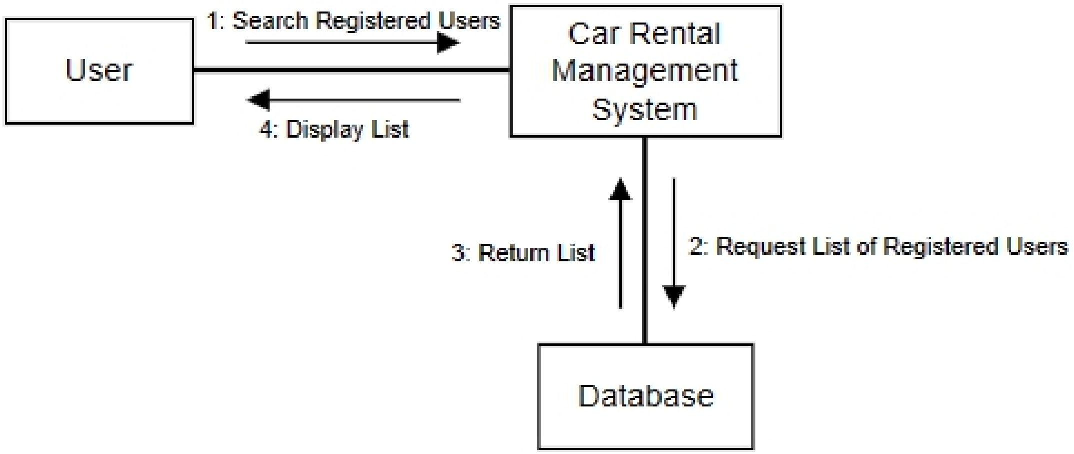
UC-10: UC_Process_Payment



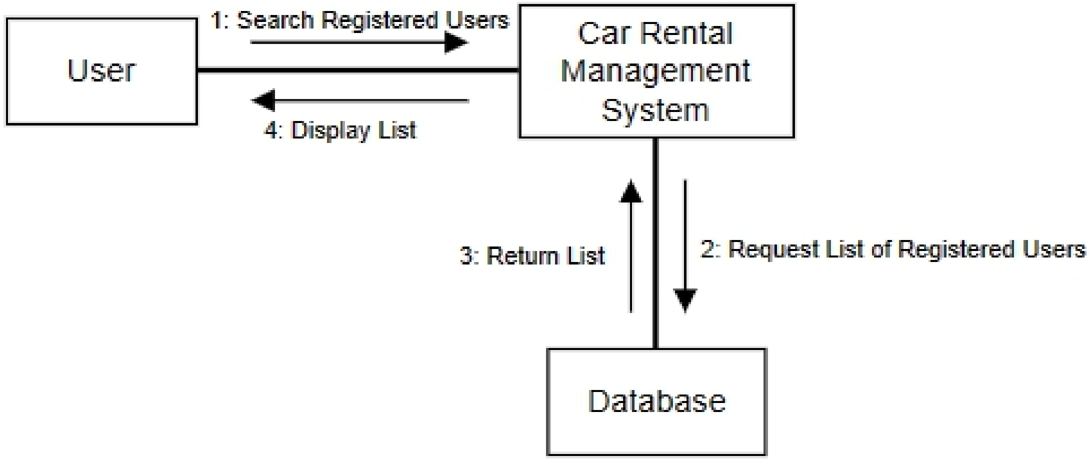
UC-11: UC_Invoice_Generation



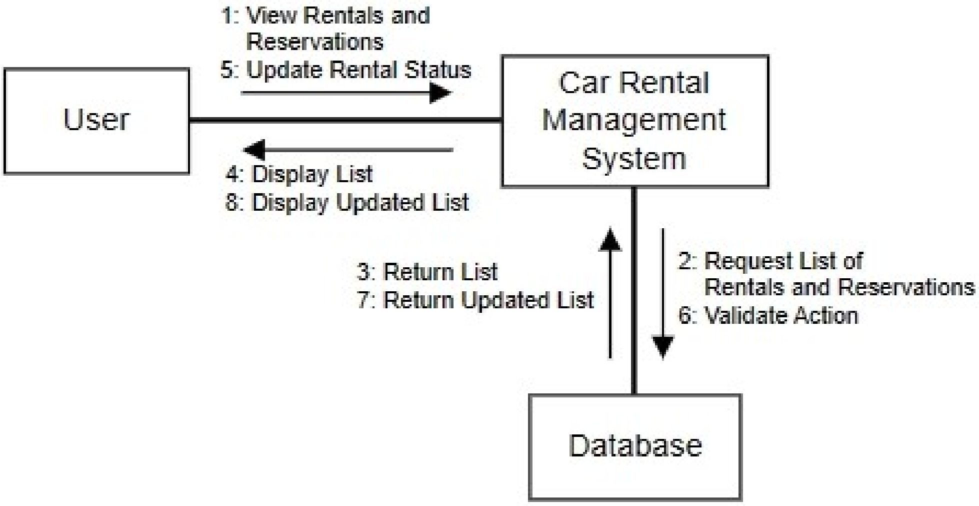
UC-12: UC_View_Customer_Details


User
1: Search Registered Users
4: Display List
Car Rental Management System
3: Return List
2: Request List of Registered Users
Database

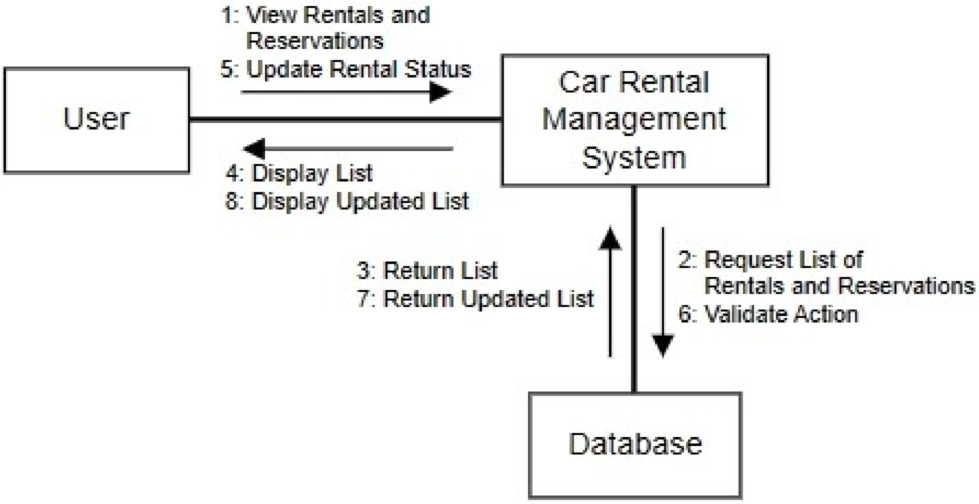
UC-13: UC_Manage_Customer_Details


User
1: Search Registered Users
4: Display List
Car Rental Management System
3: Return List
2: Request List of Registered Users
Database

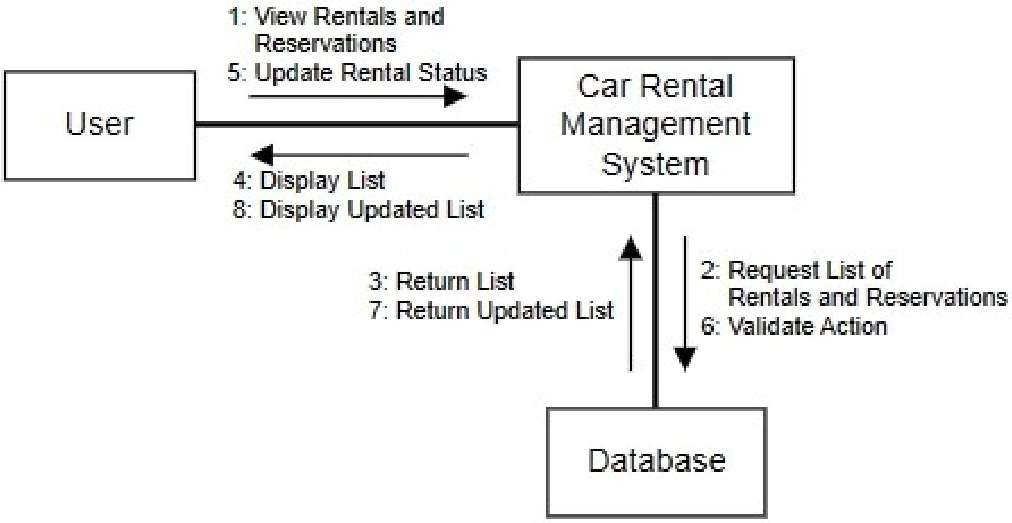
UC-14: UC_View_Action



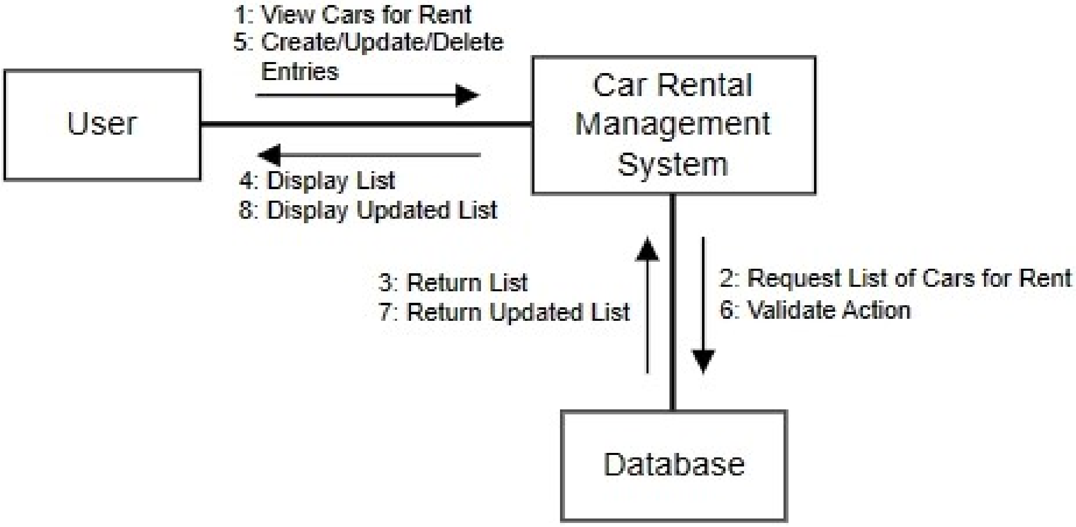
UC-15: UC_View_Transactions



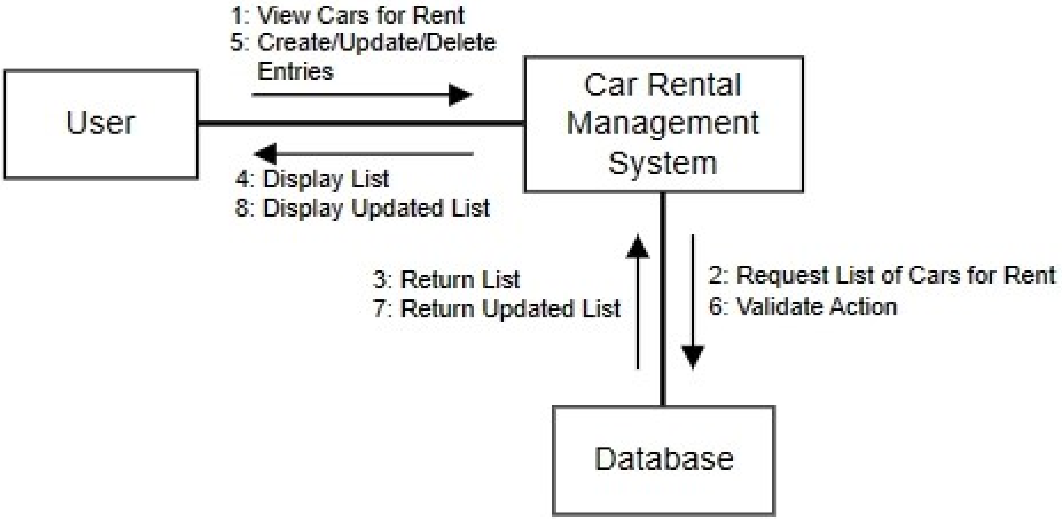
UC-16: UC_Update_Status



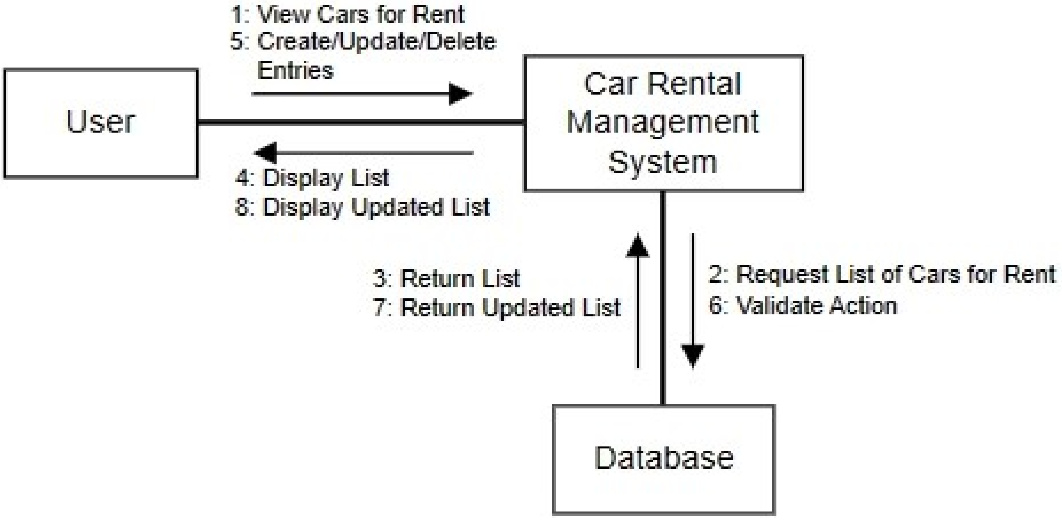
UC-17: UC_Add_Vehicle



UC-18: UC_Update_Vehicle



UC-19: UC_Delete_Vehicle



3.5 Class Diagram

The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages

Here are the key elements of a class diagram:

1. **Class**: A class is represented as a rectangle with three compartments. The top compartment contains the name of the class, the middle compartment lists the attributes or properties of the class, and the bottom compartment lists the methods or behaviors of the class.

2. **Attributes**: Attributes represent the data or properties associated with a class. They are listed in the middle compartment of the class rectangle. Each attribute typically includes a name and a data type.

3. **Methods**: Methods represent the operations or behaviors that can be performed by objects of the class. They are listed in the bottom compartment of the class rectangle. Each method typically includes a name, parameters (if any), and a return type.

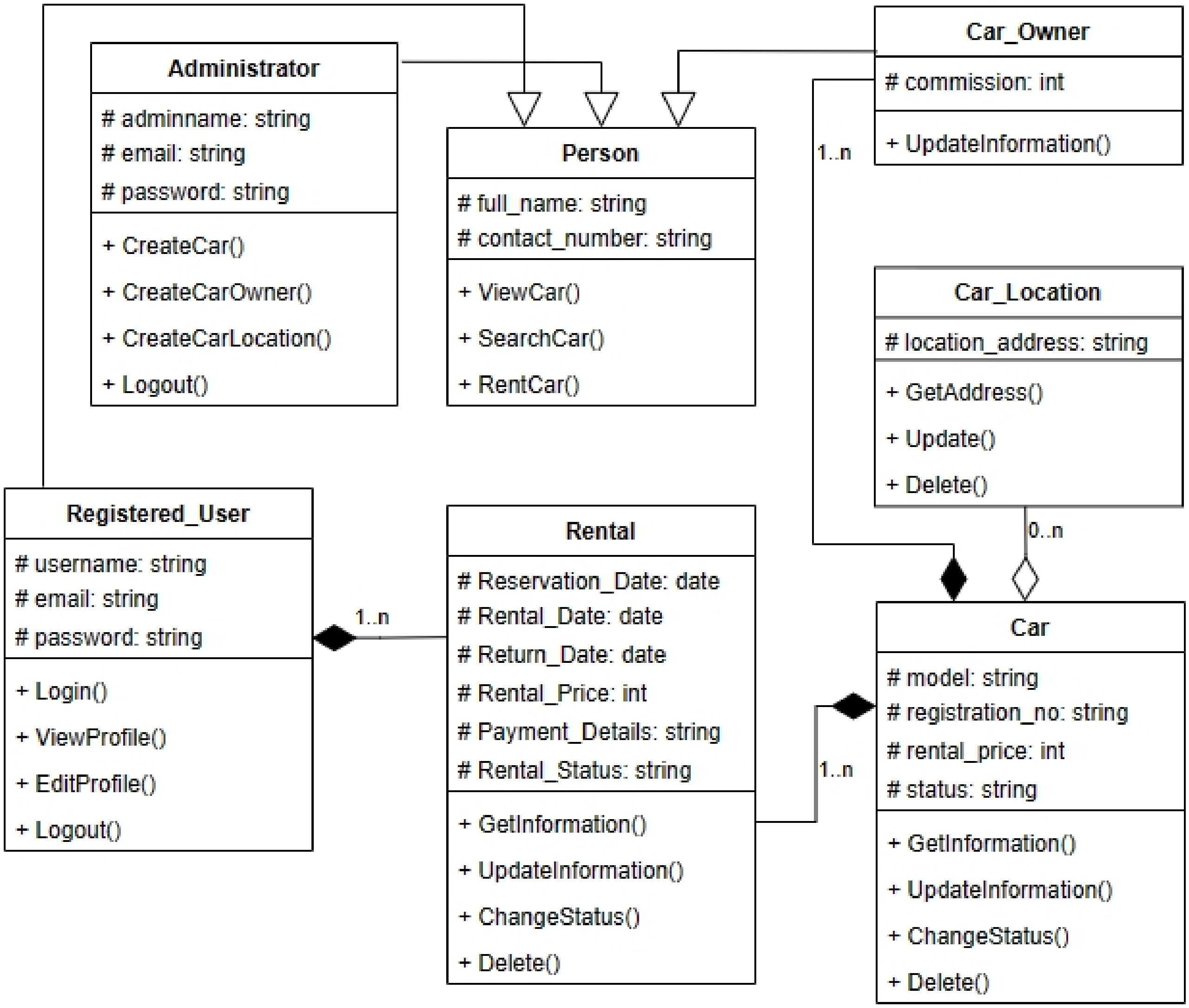
4. **Association**: Association represents the relationships between classes. It indicates that objects of one class are connected to objects of another class. Associations are represented by lines connecting the classes, with optional arrows indicating the direction of the relationship.

5. **Multiplicity**: Multiplicity specifies the number of instances of one class that can be associated with a single instance of another class. It is indicated near the ends of the association lines using numbers, asterisks (*), or other symbols.

6. **Inheritance (Generalization)**: Inheritance represents an "is-a" relationship between classes, where one class (subclass or child class) inherits the attributes and methods of another class (superclass or parent class). It is represented by a solid line with a hollow arrowhead pointing from the subclass to the superclass.

7. **Interface**: An interface defines a contract for classes that implement it, specifying a set of methods that the implementing classes must provide. In a class diagram, interfaces are represented similarly to classes but with the stereotype "<<interface>>" above the class name.

8. **Dependency**: Dependency represents a relationship where one class depends on another class, typically because it uses or relies on it in some way. It is represented by a dashed line with an arrow pointing from the dependent class to the independent class.



Administrator

adminname: string
email: string
password: string

+ CreateCar()
+ CreateCarOwner()
+ CreateCarLocation()
+ Logout()

Person

full_name: string
contact_number: string

+ ViewCar()
+ SearchCar()
+ RentCar()

Car_Owner

commission: int

+ UpdateInformation()

Car_Location

location_address: string

+ GetAddress()
+ Update()
+ Delete()

Registered_User

username: string
email: string
password: string

+ Login()
+ ViewProfile()
+ EditProfile()
+ Logout()

Rental

Reservation_Date: date
Rental_Date: date
Return_Date: date
Rental_Price: int
Payment_Details: string
Rental_Status: string

+ GetInformation()
+ UpdateInformation()
+ ChangeStatus()
+ Delete()

Car

model: string
registration_no: string
rental_price: int
status: string

+ GetInformation()
+ UpdateInformation()
+ ChangeStatus()
+ Delete()

Class Diagram of Car Rental Management System

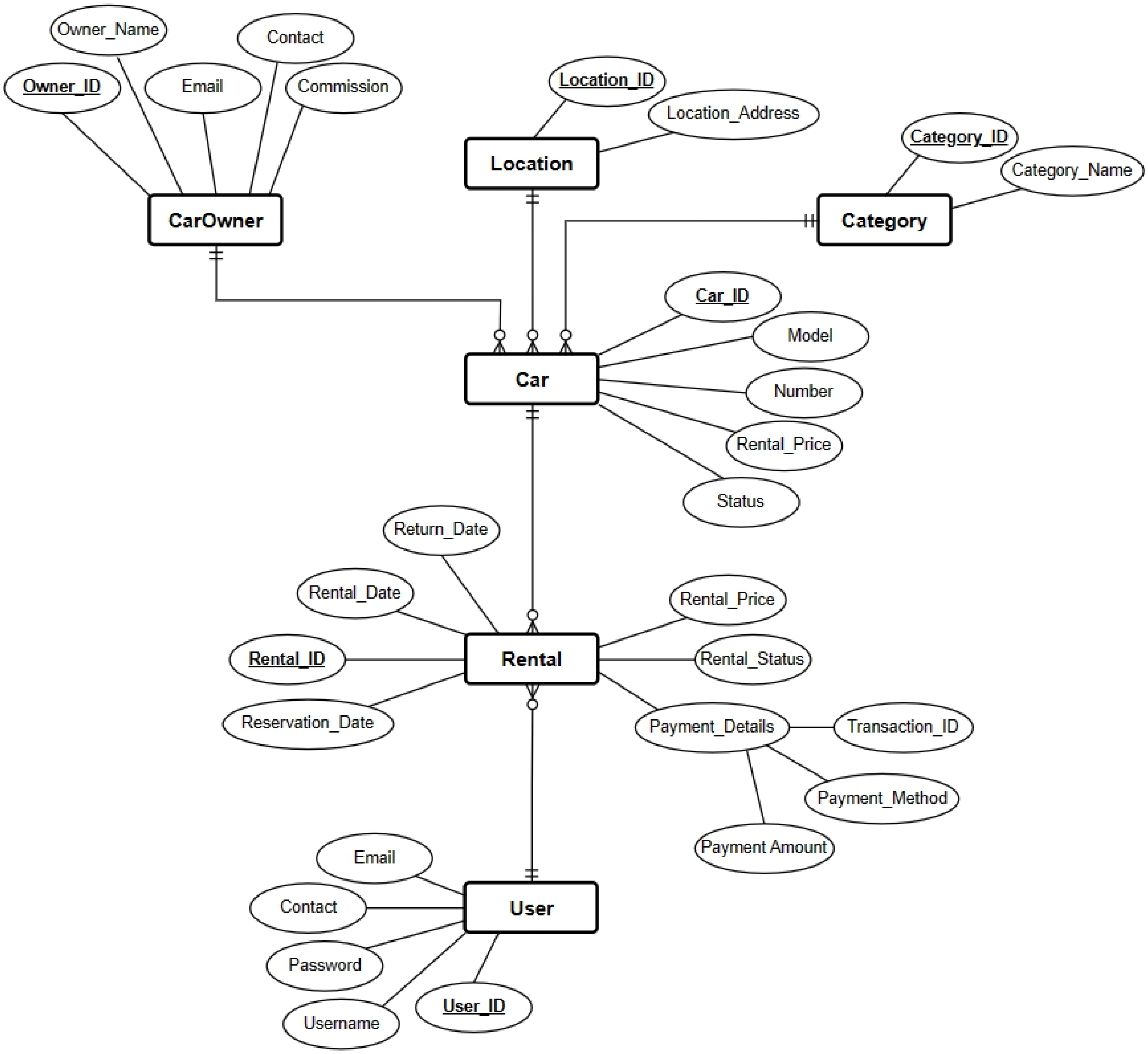
3.6 Relational Data Model

The data model is a subset of the implementation model, which describes the logical and physical representation of persistent data in the system. The relational model is composed of entities and relations. An entity may be a physical table or a logical projection of several tables also known as a view.

A relational data model diagram visually represents the structure and relationships within a relational database. It typically consists of tables, each representing an entity, with rows depicting individual records and columns representing attributes. The diagram illustrates the connections between tables through foreign key relationships, highlighting how data elements relate to each other. Primary keys are indicated to denote unique identifiers within each table, facilitating data integrity and ensuring accurate data retrieval and manipulation.

Designing Relational Data Model undergoes following steps.

1. **Identify Entities**: Identify and list the main entities or objects of interest in the system. Entities represent real-world objects or concepts that will be stored in the database (e.g., Customer, Order, Product).
2. **Find Relationships**: Determine how entities are related to each other. Relationships define the associations and interactions between entities (e.g., a Customer places an Order, a Product belongs to a Category).
3. **Draw Rough ERD**: Create a rough sketch of the ERD, illustrating entities and relationships visually using basic symbols (rectangles for entities, lines for relationships).
4. **Find Cardinalities**: Determine the cardinality and optionality of relationships (e.g., one-to-one, one-to-many, many-to-many) based on the business rules and requirements.
5. **Draw Key-Based ERD**: Refine the ERD by adding primary keys and foreign keys to represent the unique identifiers and relationships between entities more clearly.
6. **Identify Attributes of each Entity**: List the attributes (properties or characteristics) associated with each entity. Attributes describe the data that needs to be stored for each entity (e.g., Customer entity may have attributes like CustomerID, Name, Email).
7. **Draw Fully-Attributed ERD**: Finalize the ERD by incorporating all identified attributes within the entities. This complete ERD provides a detailed overview of the database schema, including entities, relationships, keys, and attributes.



Entity-Relationship Diagram of Car Rental Management System

- **Car Owner and Car:**

  Relationship: One-to-Many
  A Car Owner can own one or multiple cars. Each car is owned by one Car Owner.

- **Car and Location:**

  Relationship: Many-to-One
  A Car is located at one specific Location. However, multiple cars can be located at the same Location.

- **Car and Category:**

  Relationship: Many-to-One
  A Car belongs to one Category. However, multiple cars can belong to the same Category.

- **Car and Rentals:**

  Relationship: One-to-Many
  A Car can be rented multiple times (or once) for different rental periods. Each rental record corresponds to one specific Car. This indicates that one car can have multiple rental instances associated with it over time.

- **User and Rentals:**

  Relationship: One-to-Many
  A User can make multiple rental transactions (or reservations). Each rental is initiated by one User. This relationship signifies that one user can have multiple rental records associated with their account.

CONCLUSION

The Car Rental Management System (CRMS) represents a tool for modernizing and optimizing car rental businesses in today's competitive landscape. Throughout this journey, we've explored the multifaceted benefits and functionalities that the CRMS brings to both customers and rental companies alike.

For customers, the CRMS offers a seamless and convenient rental experience, empowering them to browse, select, and manage their bookings with ease. The intuitive interface and real-time updates ensure transparency and confidence in their rental decisions. With features such as reservation management and timely notifications, the CRMS enhances customer satisfaction and fosters loyalty by putting their needs at the forefront of the rental process.

On the rental company's side, the CRMS revolutionizes fleet management by providing a centralized platform for inventory tracking, maintenance scheduling, and pricing management. Streamlined administrative tasks such as invoicing, billing, and reporting enable businesses to operate more efficiently and profitably.

The CRMS continues to evolve and adapt to the changing needs and demands of the car rental industry. With ongoing advancements in technology and feedback from users, the system will remain at the forefront of innovation, driving greater efficiency, profitability, and customer satisfaction for rental companies worldwide.