Car rental franchise app

Requirements Specification

Version 1.0

April 19, 2021

**Table of Contents**

[**CAR RENTAL FRANCHISE APP REQUIREMENTS SPECIFICATION 1**](#_heading=h.19c6y18)

[**VERSION 1.0 1**](#_heading=h.3tbugp1)

[**APRIL 19, 2021 1**](#_heading=h.28h4qwu)

[**1.**](#_heading=h.nmf14n) **EXECUTIVE SUMMARY 3**

[1.1](#_heading=h.37m2jsg) Project Overview 3

[1.2](#_heading=h.1mrcu09) Purpose and Scope of this Specification 3

[**2.**](#_heading=h.46r0co2) **PRODUCT/SERVICE DESCRIPTION 3**

[2.1](#_heading=h.2lwamvv) Product Context 3

[2.2](#_heading=h.111kx3o) User Characteristics 3

[2.3](#_heading=h.3l18frh) Assumptions 3

[2.4](#_heading=h.206ipza) Constraints 3

[2.5](#_heading=h.4k668n3) Dependencies 4

[**3.**](#_heading=h.2zbgiuw) **REQUIREMENTS 4**

[3.1](#_heading=h.1egqt2p) Functional Requirements 5

[3.2](#_heading=h.3ygebqi) Non-Functional Requirements 5

[*3.2.1*](#_heading=h.2dlolyb) *Product Requirements 5*

[**3.2.1.1**](#_heading=h.44sinio) **User Interface Requirements** 6

[**3.2.1.2**](#_heading=h.2jxsxqh) **Usability** 6

[**3.2.1.3**](#_heading=h.z337ya) **Efficiency** 6

[3.2.1.3.1](#_heading=h.3j2qqm3) Performance Requirements 6

[3.2.1.3.2](#_heading=h.1y810tw) Space Requirements 6

[**3.2.1.4**](#_heading=h.4i7ojhp) **Dependability** 6

[**3.2.1.5**](#_heading=h.2xcytpi) **Security** 7

[*3.2.2*](#_heading=h.sqyw64) *Organizational Requirements 7*

[**3.2.2.1**](#_heading=h.3whwml4) **Environmental Requirements** 7

[**3.2.2.2**](#_heading=h.2bn6wsx) **Operational Requirements** 7

[**3.2.2.3**](#_heading=h.qsh70q) **Development Requirements** 7

[*3.2.3*](#_heading=h.3cqmetx) *External Requirements 7*

[**3.2.3.1**](#_heading=h.1pxezwc) **Regulatory Requirements** 7

[**3.2.3.2**](#_heading=h.49x2ik5) **Ethical Requirements** 7

[**3.2.3.3**](#_heading=h.2p2csry) **Legislative Requirements** 7

[3.2.3.3.1](#_heading=h.147n2zr) Accounting Requirements 7

[3.2.3.3.2](#_heading=h.3o7alnk) Security Requirements 7

[3.3](#_heading=h.1rvwp1q) Domain Requirements 7

[**4.**](#_heading=h.4bvk7pj) **USER SCENARIOS/USE CASES 7**

# Executive Summary

## Project Overview

The objective of this project is to create a web-based software program that can control a franchise for car rentals. With the help of the software, the franchise will be able to keep track of every vehicle in their fleet, including information on the make, model, year of manufacture, color, type of transmission, cost per day, fuel type, number of doors and seats, and more. The franchise will also be able to handle reservations, including dates, hours, and places, as well as create client invoices and contracts.

The app will help workers of a car rental franchise coordinate their work and reduce the usage of paper-based documents. An agent can store all of his transactions in dedicated spaces within the application and can share them with other agents and with administrators. Moreover, the app is directly connected to the invoice system, so every transaction is kept track of, and a bill can be issued per customer’s choice. Another crucial feature for the agents is that they can now update all the vehicles that are being rented so that there is no contradiction between customers.

In addition, our software will include other features that will facilitate the usage of the customers. One of them is the payment system which will be easy to use and completely safe for any user. The system will also support a catalog from which customers can read details for each car such as price and other specifications and make their decision. It is of paramount importance that the application maintains a reasonable ratio between being user-friendly and being professional. That being said the menus for the customers should be as friendly and as simple as possible however the views of agents and especially administrators must contain their dosage of professionalism since it is likely that the data provided to them will be used to compile official financial and legal documentation.

Key importance when developing this app will be given to data security from the creation of a secure database to the encryption of sensitive information such as employees’ SSNs, etc. The users should at first be certain about their integrity and security when using the app and then enjoy the experience that the software offers.

Project timeline: The extent and complexity of the software's features will determine the project's timeline. For development and testing, a general estimate for a project like this may be between 6 and 12 months.

Development team: A project manager, software architects, UI/UX designers, frontend developers, backend developers, database developers, quality assurance engineers, and a technical writer will likely make up the development team for this project. For GPS tracking, marketing/advertising integration, and payment processing, the team could also have to collaborate with outside providers.

## Purpose and Scope of this Specification

The purpose of this specification is to provide an evaluative insight into the current state of product design. It will also serve as a document to keep track of all the activities and processes related to the planning and drafting of each element of the project. The target audience of this specification is the development team and stakeholders such as investors or managing parties.

Within the scope of this documentation lie all aspects of the planning process discussed in detail such as:

* Features and functionalities included in the product.
* An outline of the application’s processes and interface.
* User characteristics, constraints, dependencies and assumptions. (All discussed in part 2 of the specification)
* List of requirements alongside evaluation and ranking. (Discussed in part 3)
* Full requirements’ description based on the type and importance of each requirement. (Discussed in part 3)
* Detailed information for the flow of action and flow of data within the application.
* User scenarios and descriptions of how the user will be able to operate the system. (Use case diagrams and use case explanations discussed in the last part of the specification)

There are other aspects that fall outside of the scope including:

* Financial considerations including but not limited to: financial outcome calculations, auditing, cost of each feature, etc.
* Issues related to product liability such as terms and agreements. (Note that we do refer to the inclusion of terms and agreements as a legal requirement, but the content of such legislative documentation will not be part of the project specification)
* Anything related to product marketing

Note: All the items that are not within the boundaries of the scope are either already dealt with by other parties included in the project or are not a direct responsibility of the project development team.

# Product/Service Description

## Product Context

This software application intends to facilitate and improve the process of car renting and the interaction of the car rental franchise with the customers. There are many other apps with the same purpose in the current market, however, we as a team have designed this software with the goal of gathering all the available features that have proven to be useful and desirable by the user while also adding new functionalities that will distinguish our app from others and revolutionize the involvement of technology in the car rental domain.

The software has a user-friendly GUI which is easy to use and compatible with all devices in the market. The fairly simple design of the interface is certain to help workers adapt to using it and to attract more clients since it does not require much concentration to be used.

For the customers, it will provide interactive menus from which they can make their reservations quickly and safely. While for other users such as administrators and agents, the focus will be on the easiness of communication within the organization. The system will provide a grid of messages and notifications to keep them up to date and will provide commands that allow them to inform each other in real-time.

## User Characteristics

**Administrator:**

* Has the authority to manage everything that agents and drivers do.
* Has the authority to manage employees’ information.
* Can view data about total incomes and total costs.
* Needs access to data about all the cars, reservations, and customer information.
* Changes the prices of the cars and other specifications.
* Adds and removes cars from the catalog.

**Car Rental Agent:**

* Needs access to data about available cars and reservations.
* Can issue invoices and rental contracts.
* Needs access to GPS device data for tracking cars.
* Receives notifications about the day’s reservations and rented cars.
* Needs to check for fines and ensure customers pay them.
* Notifies other agents when a car becomes unavailable.

**Customer:**

* Needs to log in or sign up to make a reservation.
* Can see available cars and make reservations through the website.
* Needs to provide personal information, such as name, surname, birthday, ID, email, and phone number.
* Can pay through credit card.
* Is obliged to return the car in good condition and pay for any fines incurred.
* Notifies the agency for any problems he might have with the car  
  .
* Can inquire information about prices and cars specifications.
* May have different levels of subscription (regular, premium, or platinum) with different rental options and extra features

**Driver:**

* Needs access to reservation information.
* Can check off whether the car has been delivered or not.
* Is responsible for checking the car before and after delivering it to the customer and ensuring it is in good condition.
* Reports any damage or issue to the administrator.
* Reports about any investment that is needed for any of the cars.

## Assumptions

1. It is assumed that the staff and customers are familiar with the English or Albanian language.
2. The car rental software application assumes the availability of a stable and reliable internet connection for both the staff and customers to access the system.
3. It is assumed that the application will be available on popular mobile devices with the latest iOS and Android operating systems.
4. The application assumes that the staff and customers have basic knowledge of operating the application and troubleshooting any issues that may arise.
5. It is assumed that the car rental company will have a valid and up-to-date insurance policy for all their vehicles and customers, and that customers will be made aware of the insurance coverage and limitations.
6. It is assumed that the car rental company will have enough vehicles available to meet the demand of their customers during peak times.
7. It is assumed that the car rental company will have a customer support system in place to address any concerns or issues that customers may have while using the application or renting a vehicle.
8. The application assumes that the customers will return the rented car on time and in the same condition as when it was rented, and that any damages or additional charges will be promptly communicated and resolved.

## Constraints

The system will have the following constraints:

* Every manager, agent, driver, or customer must log in and/or register with a username and password.
* The software should be able to integrate with other systems such as financial ones so that the customers can pay online for the reservations.
* To make a reservation, the customer needs to create an account or log into an existing one.
* The software should be able to properly handle a huge number of customers, reservations, and other operations, especially during busy season.
* The software should be secure and protect the data of the customers, reservations, and payment history.
* It should provide backup in case of crashes and errors.

## Dependencies

1. In order to print the bill contract, the car rental agent must enter the information of the customer and the available car.
2. If a car is not available, the car rental agent cannot make a reservation for it. In order to make a rental reservation, the customer needs to log in, pick an available car and pay for the provided service.
3. If a chosen car is not currently available, the customer may choose another car for reservation.
4. In order to finalize the return of the vehicle, the customer must pay every fine placed on the vehicle.
5. Customers are dependent on their subscription so that they can pick a certain car model.
6. The customer cannot pick any car they want if they do not have a subscription on the service.
7. In order to receive a discount on the next rental, the customer must have made reservations at least 5 times.
8. A car cannot be available for reservations, if the driver has confirmed it is under use by another customer.
9. The car’s availability is dependent on the condition of the car post-rental, checked by the driver, in order to be available for another time.

# ***Requirements***

## Functional Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req#** | **Requirement** | **Comments** | **Priority** | **Date Rvwd** | **SME Reviewed / Approved** |
| R\_01 | The system should support a multi-user environment. | This is done to facilitate the permission based features. It is much easier to have users rather than check permissions on action each time the server receives a request. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_02 | The system should be supported by all browsers. | The system should not have limitations about the browser since clients are free to use any browser they want. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_03 | The system should be usable in tablets and mobile phones. | Obviously the goal of the project is facilization of the car rental business and the opportunity to use such an application from the comfort of your phone is a must | 1 | 8/5/23 | Aldrin Çifliku |
| R\_04 | The system should differentiate users’ permission and features based on their level | For the application to function logically as well as technically, we need to implement a system that is able to tell if the person who is using the app is allowed to do certain things. For example an agent is not allowed to fire employees so the system will not give this feature to an agent user | 1 | 8/5/23 | Aldrin Çifliku |
| R\_05 | The system should provide quick log-in information validation based on the most recent security regulations. | This is an obligation for any app that is trying to enter the market in 2023 | 2 | 8/5/23 | Aldrin Çifliku |
| R\_06 | The system should be able to carry out encrypted transactions. | This is mostly related to safety and to the banking process within the boundaries of legislation. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_07 | The system should allow connection to a dynamic database. | The interaction with the database should be done in real time instead of only when we open and close the app. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_08 | The system should keep track of financial statistics. | Since the system will be used to help the manager file for taxes but also run the business it is obligatory to gather some financial statistics so as to enable other functionalities required. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_09 | The system should be able to analyze data based on given quotas and limits. | This is because the system is required to have certain behaviors when the financial situation of the business is not optimal. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_10 | The system should be able to transfer data in almost real time. | This is a must because the application only achieves its goal if all the employees are connected properly. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_11 | The system should keep track of a user’s job status. | This is so that accounts of employees that are fired or suspended will not have the right to modify or enter new information. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_12 | The system should assign unique user id-s based on UUID technology. | This is to differentiate users in the database. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_13 | The system should calculate employees’ progress and estimate their productivity during a period. | This is related to the business end of the application. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_14 | The system should provide data about the cars that are being rented the most or the least and should update such data dynamically. | This is related to the business end of the application. | 2 | 8/5/23 | Aldrin Çifliku |
| R\_15 | The system should handle data incoming from geo tracking devices. | It is important to allow the agents to track the vehicles | 2 | 8/5/23 | Aldrin Çifliku |
| R\_16 | The system should allow users to upload documents, pdfs, photos, etc. | This allows user to upload files to their profile such as qualifications or photos and also allows agents to exchange invoices and contracts. | 2 | 8/5/23 | Aldrin Çifliku |
| R\_17 | The system should maintain only sufficient active information. | This is related to the data structures that the system should employ to guarantee efficient time of execution and sufficient information to the users. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_18 | The system should allow users to download and print information such as bills or contracts. | One of the functionalities of the application is to manage hard copies of important information such as invoices or transaction documents. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_19 | The system should be able to generate and save invoices. | This is important to keep track of the financial activity of the business. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_20 | The system should facilitate communication between coworkers. | Crucial because the system is supposed to save time. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_21 | The system should be able to handle different types of customers. | It is important since the business offers multiple levels of customers. | 2 | 8/5/23 | Aldrin Çifliku |
| R\_22 | The system should calculate incomes and return taxes automatically. | Related to the business end of the application. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_23 | The system should allow the registration of new cars or car models. | It’s important that the list should be updatable. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_24 | The system should allow users to modify cars’ information. | It is important that objects within the system are dynamic rather than static. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_25 | The system should be able to differentiate between users and display the same view as read only or as editable based on the user level. | This, for example, allows the admin to view and edit the car’s list but the agent can only pick one of the cars. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_26 | The system should be able to notify for abnormal situations. | Abnormal situations such as invalid inputs, failure, problems with the servers, etc. | 3 | 8/5/23 | Aldrin Çifliku |
| R\_27 | The system should allow users to check in and out of their shifts. | This is so that the hours of work are kept track of automatically. | 1 | 8/5/23 | Aldrin Çifliku |
| R\_28 | The system should notify users that are concerned for any important update, such as the change of the rental time for example. | Notifications should be enabled. | 2 | 8/5/23 | Aldrin Çifliku |
| R\_29 | The system should provide safe transactions for online payments. | Since customers can use the website to make reservations online, they should also be able to pay online for their reservations. This is why the system should make sure that the payment process is safe. | 1 | 8/5/23 | Fabiona Tafçiu |
| R\_30 | The system should support embedded systems. | The website will have a GPS system and Fine system to support the needs of the agency. This is why the system should be able to handle the embedded system. | 1 | 8/5/23 | Fabiona Tafçiu |
| R\_31 | The system should generate reports for incomes, number of reservations, and cars. | The admin may want to check how the agency is doing and the incomes to make a better business plan or get general information. | 2 | 8/5/23 | Fabiona Tafçiu |
| R\_32 | The admin can create, update or delete services or cars from the system. | This is done so the clients and agents are up to date with what the agency is offering. | 2 | 8/5/23 | Fabiona Tafçiu |
| R\_33 | The user has to log in to make a reservation. | If they do not log in, they can only view the services that are provided by the agency. | 1 | 8/5/23 | Fabiona Tafçiu |
| R\_34 | The admin can create, update or delete an employee's profile. | The admin can add agents or drivers in the system or modify existing ones. | 2 | 8/5/23 | Fabiona Tafçiu |
| R\_35 | The system should support online payment. | The system should be able to offer the customer the opportunity to pay via credit card through different banks. | 1 | 8/5/23 | Fabiona Tafçiu |
| R\_36 | The agent can add reservations or modify them. | They can change the reservation dates or other information or add new ones when getting the requirements by the clients. | 1 | 8/5/23 | Fabiona Tafçiu |
| R\_37 | The agency should provide a catalog with all their cars and other services. | This way, the people visiting the website can get a better understanding of the agency and know what services they are provided. | 2 | 8/5/23 | Fabiona Tafçiu |
| R\_38 | Customers can cancel the reservations three days before the date without paying a fine. | If a customer makes a reservation online, he cannot delete the reservation within the last three days, otherwise he will have to pay a percentage of the reservation payment. | 2 | 8/5/23 | Fabiona Tafçiu |
| R\_39 | Only the admin and agents have access to the reservation to modify or change them. | This is done to secure data and information so other users cannot interact with the reservations. | 1 | 13/05/23 | Fabiona Tafçiu |
| R\_40 | The online reservations should be confirmed by the agents within 24 hours. | After making the reservations online, the customer should get a confirmation by the agent if their order can be fulfilled. | 1 | 13/05/23 | Fabiona Tafçiu |
| R\_41 | Customers and agents can filter cars based on their characteristics. | When looking for a car to book, they can limit their choices by selecting different car characteristics such as transmission gear, petrol, year of production etc. | 3 | 13/05/23 | Fabiona Tafçiu |
| R\_42 | The agent should print and give the customer the signed contract. | When finishing the reservation, the customer should be provided a contract with all the legal responsibilities and rights of the rental service. | 2 | 13/02/23 | Fabiona Tafçiu |
| R\_43 | According to the number of reservations or the extent of the reservation, the customer can receive a discount. | If a customer has rented up to 5 cars with the agency, he is provided a 5% discount on the next reservation. If the car is reserved for more than 5 days, the discount is applied, too. | 3 | 13/05/23 | Fabiona Tafçiu |
| R\_44 | The customers can leave comments and reviews about the services offered from the agency. | In order to get feedback from the customers, the website provides a section for them to leave comments and ratings. | 4 | 13/05/23 | Fabiona Tafçiu |
| R\_45 | The website should provide the customers the price of every service. | When looking through the catalog of services, the customer should also have information about the price. | 3 | 13/05/23 | Fabiona Tafcu |
| R\_46 | The driver should be able to change the delivery status and returning status of the cars. | The driver will have a section to update if the car has been delivered to the customer or taken from them through a check box. | 1 | 13/05/23 | Fabiona Tafçiu |
| R\_47 | The system should provide customer support through phone, email, or chat. | In case the customer has problems during the rental period and requires assistance, the website should provide them with ways to communicate with the agents. | 2 | 13/05/23 | Fabiona Tafçiu |
| R\_48 | The agency should offer insurance and additional services. | Apart from standard services, the agency offers additional services such as insurance to the customer for extra payment. | 2 | 13/05/23 | Fabiona Tafçiu |

## Non-Functional Requirements

### Product Requirements

#### **User Interface Requirements**

A car rental franchise application UI should be designed to provide a seamless and user-friendly experience for all user roles: Managers, car rental agents, customers, drivers.

Application should include a login screen with clearly marked username and password fields so that users can access their accounts securely.

Upon login, each user role should be greeted with a visually appealing and intuitive dashboard. The dashboard should present an overview of key information specific to that role. For example, an administrator's dashboard shows revenue and expense totals, while a car rental agent's dashboard shows notifications about bookings and available cars for the day.

A fleet management interface should allow easy addition, editing, and deletion of vehicle details such as model, year, color, transmission type, fuel type, doors, seats, and price per day. Filters and search options should be available so that users can find specific cars based on their criteria. A visual indicator should clearly indicate vehicle availability, and a notification should be displayed to the car rental company if the vehicle has already been rented.

A booking management interface should provide a clear calendar view or booking system for users to create and manage bookings. The interface should also include notifications and reminders for upcoming reservations and overdue returns.

For the customer role, the interface should be able to enter first name, last name, date of birth, card id, email his address, phone number to login or register, view available cars, and book. There should be an online credit card payment area and the customer should be able to choose the time, date and location of vehicle pickup. In addition, the customer should be informed through the interface of any fines during the rental period and be able to pay them upon returning the vehicle.

For the driver role, the interface should provide access to view the reservation without modifying the reservation. The interface should include a checkbox to indicate whether the car has been delivered and a checklist to ensure the car is in good condition before and after delivery. This includes returning advanced functions and checking the overall condition of the vehicle. Overall, the user interface should prioritize usability, clear navigation, and intuitive interactions to improve the user experience for each user role of the car rental franchise application.

#### **Usability**

-The software should be available in terrain for drivers to notify the other employees on real time.

-The software should be easy to use even for people without too much experience using smartphones or computers.

-The software’s design should be compatible and adaptable to all smart devices.

-The software should display clear messages when requiring action from the user.

-The software should have the same functionalities regardless of the preferences of the user such as language, screen mode, font size etc.

-The software must allow usage through keyboard shortcuts as well as through mouse and touchscreen

-The software should support most common file types so that the user will feel free to upload them regardless of their device. (Eg. Linux saves zip files with .zip while most commonly on windows we find .rar files)

-Navigation should be clear and consistent (Eg. User should be aware when switching menus through transitions. Eg. The screen size does not change if you switch menus)

#### **Efficiency**

##### Performance Requirements

Response Time: To guarantee that user interactions are immediately carried out, the software should have quick response times. To give users a seamless and responsive experience, target response times should be under a few seconds.

Processing Speed: The program should analyze input and make calculations quickly. To avoid delays or slowdowns, complex operations should be finished in a reasonable amount of time, such a few milliseconds.

Scalability: The program must be scalable to handle growing data and user volumes. It should be able to accommodate growth while maintaining high levels of performance, supporting numerous concurrent transactions (for example, 1000+ concurrent users).

System faults or breakdowns should be kept to a minimum thanks to the software's high level of dependability. High availability should be the goal, with an uptime goal of at least 99% or greater.

Network Performance: For data transfer and remote communication, the software should have efficient network performance. Low latency should be desired, with network response times around 100 milliseconds.

##### Space Requirements

Resource Use: To prevent excessive consumption, the software must use system resources wisely. To avoid slowdowns or performance problems, it should maintain low resource usage, such as CPU utilization below 70% and RAM usage below 80%.

Load Handling: The program should be able to handle anticipated loads and peak periods without noticeably degrading its performance. In order to achieve a maximum response time of 2 seconds even under heavy loads, it should enable concurrent user requests.

Data Storage and Retrieval: The program should effectively retrieve and store data, especially for huge datasets or databases. Database queries should be prepared for quick retrieval, with a target response time of 500 milliseconds or less.

Database Capacity: The software's database ought to be able to hold a sizable amount of data. It should be able to support a database with a minimum capacity of 500 GB, which can accommodate millions of records and transactions.

File Attachment Limit: Users should be able to attach files and documents to their records using the software. The application should allow users to upload files up to 20 megabytes (MB) in size, guaranteeing they may upload and store larger files if necessary.

Memory utilization: To achieve effective performance, the software should optimize memory utilization. To avoid excessive memory utilization and associated system slowdowns, it should try to maintain memory consumption below 500 megabytes (MB) or 10% of the available system memory.

#### **Dependability**

-The software should be reliable and available 24/7 with minimal downtime.

-It should have automated backup and recovery mechanisms to ensure data integrity and continuity.

-The software should be able to handle errors and exceptions gracefully without crashing or corrupting data.

-The software should have proper documentation and support to assist users in case of issues or questions.

**Availability**

-Any internet-connected device should be able to access the software.

-It must have enough server power to support numerous users at once.

-The application should work with widely used operating systems and browsers.

-To guard against illegal access and guarantee data privacy, it should have security measures in place.

-Should be available remotely

**Reliability**

Data integrity:

Applications must maintain data integrity by accurately storing and retrieving information about cars, reservations, users, and financial transactions.

Fault tolerance:

Systems must be resilient to failures and failures, ensuring that critical functionality remains available even in the event of hardware or software failure.

Backup and restore:

The application should regularly back up data to avoid data loss in the event of a system crash. It also needs mechanisms to recover and recover data quickly in the event of an unforeseen failure.

Error handling:

The application will skillfully handle errors by providing clear and user-friendly error messages, instructing users on how to troubleshoot or contact support if necessary. Protect:

Applications should implement strong security measures to protect user data, prevent unauthorized access, and secure financial transactions. This includes encryption of sensitive data, secure authentication, and compliance with applicable data protection regulations.

Ability of extension:

Applications must be designed to handle increasing user loads and increasing volumes of data without significant performance loss. It will be able to adjust resources as needed to meet demand.

Performance tracking:

Applications should include monitoring mechanisms to monitor system performance, identify bottlenecks, and ensure optimal response times. This monitoring should be proactive, allowing administrators to take the necessary steps to maintain high levels of performance.

User feedback and support:

Apps must allow users to provide feedback or report problems. In addition, effective support channels should be in place to address user concerns and resolve technical or functional issues in a timely manner.

Compliance:

Applications must comply with industry standards, regulations and best practices, ensure the confidentiality and security of user data, and comply with legal requirements for financial transactions and security.

**Monitoring**

-Performance: regularly check the system's response time, server load, and network traffic to ensure optimal performance.

-Security: regularly monitor for security vulnerabilities and implement security measures such as firewalls, encryption, and intrusion detection systems.

-Backups: regularly back up data to prevent data loss in case of system failure.

**Maintenance**

-Updates: regularly update the software with the latest features and security patches.

-User support: provide 24/7 user support with a help desk for reporting issues.

-Testing: regularly test the software through unit testing, integration testing, and system testing.

-Documentation: maintain up-to-date documentation to ensure ease of use.

#### **Security**

-The software should have strong encryption and authentication mechanisms to protect sensitive data, such as credit card information and employee records.

-It should have role-based access control to restrict user privileges and prevent unauthorized actions.

-The software should have a firewall and intrusion detection system to prevent cyber-attacks.

-The software should comply with relevant data protection regulations, such as

GDPR or CCPA.

### Organizational Requirements

#### **Environmental Requirements**

All the important documents must be part of our system’s database. This is because the system will deal with crucial activities such as payments, tax filing, financial management, and so on. Therefore, it is essential for the system to contain all the information for bills, contracts, employee’s deals and bonuses, etc. Needless to say all the information will be retained with the consent of the company’s directors.

In addition to documents the system might need to have information about the whereabouts of the offices and of cars. This is to ensure that the information presented in the web app is up to date and to enhance publicity.

The system does not contain any hardware so there is no other specification to the environment of usage. However, it must be assumed that the computers and smart devices where this software will be run will be up to date with the most recent technologies.

#### **Operational Requirements**

* User should log in before using
* The user creates an account or has his account created by the administrator before he can log in.
* The system should not be overloaded (for example users should not perform multiple transactions from the same account at the same moment).
* The system should be available 24/7
* Administrator should approve before changing essential information.
* When performing transactions of high importance the user should make sure that the connection is stable.
* When printing invoices and contracts the user should guarantee resources such as printer and paper.
* To use the tracking system the users should provide the GPS devices and connect them to the application.
* Administrator creates, modifies and deletes users.
* Administrator creates, modifies and deletes cars.
* Users should double check schedules before confirming them.
* The users should be careful with information such as SSN or legal names as they will be used to file for taxes and mistakes might happen.

#### **Development Requirements**

The application will be created using the JavaScript programming language and, more precisely, the React Native framework. Because React Native enables cross-platform development, the application may be made available for both iOS and Android devices. It offers a reliable and effective development environment for putting app functionality in place and creating user interfaces.

Database built with Firebase: React's backend-as-a-service platform Firebase will be used to build the application's database. Building dynamic and scalable mobile applications is suited for Firebase since it provides real-time data synchronization, authentication, and other crucial features. The cloud-based architecture and simplicity of integration offered by Firebase can be used to the app's advantage.

Incremental testing: At each level of development or increment, the system will be tested to make sure the objectives and specifications are met. This comprises performance testing to evaluate the app's speed, responsiveness, and resource utilization as well as functional testing to ensure that the features and functionalities function as intended. To find any problems or faults early in the development process, testing should be thorough and cover a variety of use scenarios.

Maintenance Check: To guarantee the application's continuous performance and stability, routine maintenance checks will be carried out. This entails keeping an eye out for potential performance snarls, security flaws, and software defects, and fixing them. To keep the app operating smoothly and effectively, maintenance checks may involve software upgrades, security patches, database optimization, and general system health evaluations.

**3.2.3 External Requirements**

The Car Rental software application and its services must be in full compliance with the regulatory, ethical, and legal requirements of The Republic of Albania.

The Car Rental application system should ensure that the vehicles offered for rental comply with the technical norms on vehicle safety and emissions set by the Albanian government. The software should also be designed to comply with the personal data protection regulations, ensuring that the personal data of customers is kept secure and confidential. Additionally, the system should be capable of generating accurate and complete financial reports. Finally, the application software should be designed to comply with the consumer protection regulations, ensuring that the rental agreement terms and conditions are clear and transparent for the customers.

**3.2.3.1 Regulatory Requirements:**

* Compulsory Insurance in the Transport Sector (no. 10076, dated 29.05.2009)
* Law for Merchants and Trade Companies (no. 9901, dated 17.04.2008)
* Law on Personal Data Protection (no. 9887, dated 10.03.2008)
* Law on Electronic Communications (no. 9918, dated 19.05.2008)
* Law For Licenses, Authorizations and Permits in The Republic of Albania (no. 10 081, dated 23.02.2009)
* Law for Consumer Protection (no. 9902, dated 17.04.2008)

**3.2.3.2 Ethical Requirements:**

* Albanian Constitution, Article 18
* Law on Personal Data Protection (no. 9887, dated 10.03.2008)

**3.2.3.3 Legislative Requirements:**

* Road Code of The Republic of Albania (no. 8378, dated 22.07.1998)
* Value Added Tax Law (no. 92/2014, dated 24.07.2014)
* Income Tax Law (no. 8438, dated 28.12.1998)

## Domain Requirements

Our application operates in the car rental industry. As such, it is fair to assume that it should contain several users and features typical to that field of concern. Note that since we have already elaborated on all the other requirements, this section appears to encompass the pre-mentioned information therefore, it might seem redundant. However, the purpose of this section is to clarify which of the requirements we have provided is dictated by the industry's dynamics and to allow the person who will use this requirements document to compare the specific requirements of this application with the requirements that apps of this field usually provide.

**• Domain requirements about Users:**

The car rental application should be designed to cater to the needs of both drivers and rental companies.

Drivers may be leisure or business travelers, or individuals who need a car for personal use.

Rental companies may be small or large businesses that rent out cars, trucks, or other vehicles.

Some practical examples of user types and their needs include:

- Drivers: Drivers need a user-friendly interface for searching and booking rental vehicles, as well as clear pricing and availability information. They may also value features such as GPS tracking, roadside assistance, and customer support.

- Rental companies: Rental companies need a dashboard or management portal for adding and updating vehicle inventory, setting pricing and availability rules, and tracking bookings and payments.

They may also require features such as data analytics and reporting to optimize their operations and improve customer satisfaction.

• **Common features:**

The car rental application should provide a range of features that cater to the needs of drivers and rental companies.

Some practical examples of features that could be included are:

- Vehicle search and booking: The application should allow drivers to search for vehicles based on location, dates, and other criteria, and make reservations through a simple and secure checkout process.

- Payment processing: The application should have a secure payment gateway that supports multiple payment methods and currencies, and provides real-time payment processing and invoicing.

- Customer reviews and ratings: The application should allow drivers to leave reviews and ratings for rental companies and vehicles, and use this feedback to improve the quality of the service.

- Car availability and pricing information: The application should provide up-to-date information on vehicle availability and pricing, as well as any discounts or promotions that may be available.

- GPS tracking for rental vehicles: The application may integrate with GPS tracking devices to provide real-time location information for rental vehicles, allowing drivers to track their rental car and providing rental companies with theft prevention and recovery capabilities.

# User Scenarios/Use Cases

* 1. **Use case diagrams**

Diagram

Description automatically generated

|  |  |
| --- | --- |
| Use Case (UC\_1.1): | *Log In* |
| Summary | The agents need to log in the system to later use it. |
| Actors | Primary Actor: agent |
| Preconditions | The agent should open the website |
| Description of the Main Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: The agent is logged in successfully.* |
| Description of the Alternative Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: The agent cannot log in.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | The agent should be logged in |

|  |  |
| --- | --- |
| Use Case (UC\_1.2): | *View available cars* |
| Summary | *The agent can view the available cars in the system.* |
| Actors | *Primary actor: Agent* |
| Preconditions | *Agent is logged in the system.* |
| Description of the Main Sequence | * *Step 1: After logging in, the agent clicks the button to view the cars.* * *Step 2: They put in the time period they want to look for.* * *Step 3: The system shows the cars available for that time period.* |
| Description of the Alternative Sequence | * *Step 1: After logging in, the agent clicks the button to view the cars.* * *Step 2: They put in the time period they want to look for.* * *Step 3: There are no available cars for that time period.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The page must load within 2 seconds.* |
| Postconditions | *The agent has access to the available cars.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.3): | *Makes reservations* |
| Summary | *The agent should be able to make reservations for the customers.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *The agent should have a customer asking for a reservation.* |
| Description of the Main Sequence | * *Step 1: The agent gets the information from the customer.* * *Step 2: They open the reservation section on the website.* * *Step 3: They choose the car and the time period.* * *Step 4: They fill in the required information.* * *Step 5: They save the reservation.* |
| Description of the Alternative Sequence | * *Step 1: The agent gets the information from the customer.* * *Step 2: They open the reservation section on the website.* * *Step 3: They choose the car and the time period.* * *Step 4: The car is not available, and the agent gets a notification and does not continue the reservation.* |
| Nonfunctional requirements | *The system should be able to maintain a huge number of reservations.*  *The reservation should be saved within 3 seconds.* |
| Postconditions | *The reservation is saved on the system.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.4): | *Change the status of reservation to paid/unpaid* |
| Summary | *After making the reservation, the agent should change the status to paid when it gets paid or leave it unpaid otherwise.* |
| Actors | *Primary actor: Agent* |
| Preconditions | *There should be a reservation, so that the agent can change its status.* |
| Description of the Main Sequence | * *Step 1: The agent gets the payment from the customer.* * *Step 2: The agent changes the status to paid.* |
| Description of the Alternative Sequence | * *Step 1: If the customer decides to extend the rental period, the reservation status should become unpaid, until the extension is paid for.* |
| Nonfunctional requirements | *The system should save the changes within 3 seconds.* |
| Postconditions | *The reservation is showed in the system as paid.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.5): | *Create and print bills* |
| Summary | *The agent should create the bill for a certain reservation and print it for the customer.* |
| Dependency | *This use case depends on making reservation use case and changing the status of reservation to paid or unpaid. In order to create a bill, you should first have a reservation and the reservation should be paid or should be to be paid. This is why the use case is included in those use cases.* |
| Actors | *Primary actor: Agent* |
| Preconditions | *There should be a reservation and the customer should be paying for the reservation.* |
| Description of the Main Sequence | * *Step 1: The agent clicks the button that creates a bill after making a reservation.* * *Step 2: The website itself calculates the amount to be paid and creates the bill in the programmed format.* * *Step 3: The agent prints the bill and hand it to the customer.* |
| Description of the Alternative Sequence | * *The agent clicks the button that creates a bill after making a reservation.* * *Step 2: The website cannot create the bill because of a certain problem.* |
| Nonfunctional requirements | *The system should be accurate in order to create a correct bill with the correct amount.*  *The system should show the amount in euros and leke.* |
| Postconditions | *The bill is created and printed out. The customer has a detailed bill for the services.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.6): | *Print and sign the contract* |
| Summary | *After making the reservation, the system provides a contract with every legal issue regarding renting the car. The agent should print it, sign it and give the customer to sign it. There should be two copies of the contract, one for the agency and the other for the customer.* |
| Dependency | *This use case depends on making reservation use case. In order to print the contract, you should first have a reservation. This is why it is included in making reservation use case.* |
| Actors | *Primary Actor: Agent, Customer* |
| Preconditions | *There should be a reservation.* |
| Description of the Main Sequence | * *Step 1: The agent clicks the button to print the contract.* * *Step 2: The website creates the contract based on the programmed format.* * *Step 3: The contract is printed and signed by both sides.* |
| Description of the Alternative Sequence | * *Step 1: The agent clicks the button to print the contract.* * *Step 2: The website cannot create the contract.* |
| Non functional requirements | *Narrative description of nonfunctional requirements, such as performance and security requirements.* |
| Postconditions | *The contract is created and printed out. The customer has a detailed legal information about the services provided.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.7): | *Check the location of the cars through GPS* |
| Summary | *In order to always have knowledge about the cars and make sure that the customers are not misusing them, the agents should be able to check the location of the cars through GPS installed in every car.* |
| Dependency | *To make this use case work, the website should have embedded the software provided by the GPS company to check the car location.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *The cars should have GPS installed in them.* |
| Description of the Main Sequence | * *Step 1: The agent has access to the embedded system by clicking the appropriate button.* * *Step 2: The agent searches for the desired car and checks the location.* |
| Description of the Alternative Sequence | * *Step 1: The agent has access to the embedded system by clicking the appropriate button.* * *Step 2: The GPS is off, or the GPS system is down.* |
| Nonfunctional requirements | *It should be available 24 hours.* |
| Postconditions | *The agent is provided with the information about the car location.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.8): | *Filter the cars according to different characteristics* |
| Summary | *When searching for a car, the agents should be able to filter the cars using car model, year of production, color, price etc. according to their needs.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *The system should keep information about the car details to be able to filter cars according to them.* |
| Description of the Main Sequence | * *Step 1: The agent clicks the filter button and chooses the characteristic they want to find.* * *Step 2: Navigate through the generated responses.* |
| Description of the Alternative Sequence | * *Step 1: The agent clicks the filter button and chooses the characteristic they want to find.* * *Step 2: There is no car with those characteristics.* |
| Nonfunctional requirements | *Easily navigable.* |
| Postconditions | *The agent is provided with the cars list that have the required characteristic.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.9): | *Access customer’s information* |
| Summary | *The agent may need to access customer’s information such as phone number that are obtained during the reservation.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *The customer must have done a reservation in order to access their information.* |
| Description of the Main Sequence | * *Step 1: The agent clicks the details of the reservation.* * *Step 2: It accesses the information given to them during the filling of the reservation form.* |
| Description of the Alternative Sequence | * *The reservation is deleted so there is no other way to access the information.* |
| Nonfunctional requirements | *If the reservation is cancelled or deleted, the system should delete the customer’s information.* |
| Postconditions | *The agent is provided with the needed information.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.10): | *Delete a reservation* |
| Summary | *If the customer cancels the reservation or for some reasons, it is impossible to provide the service required, the agent should be able to delete the reservation.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *In order to delete the reservation, it should firstly exist and there should be a reason to delete it.* |
| Description of the Main Sequence | * *Step 1: The agent clicks the details of the reservation.* * *Step 2: By clicking the delete button at the end, the reservation is deleted.* |
| Description of the Alternative Sequence | * *Step 1: The agent clicks the details of the reservation.* * *Step 2: The reservation cannot be deleted.* |
| Nonfunctional requirements | *The deletion of the reservation should not affect the other reservations.* |
| Postconditions | *The reservation is deleted from the system.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.11): | *Check for fines* |
| Summary | *During the rental periods, it Is possible that the customers can get a fine for breaking the law. This is why the agent should check for fines and get the money from the customer when he delivers the car at the end of the rental period.* |
| Dependency | *To make this use case work, the website should have embedded software to check the fines provided by the state.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *There should be cars rented for that period.* |
| Description of the Main Sequence | * *Step 1: The agent clicks the button that gives them access to check the fines.* * *Step 2: The agent should search by the license plate.* * *Step 3: They should keep a note of the cars with fines.* |
| Description of the Alternative Sequence | * *Step 1: The agent clicks the button that gives them access to check the fines.* * *Step 2: The agent should search by the license plate.* * *Step 3: No car has fines, or the embedded system is down.* |
| Nonfunctional requirements | *The software should be up to date and available 24 hours.* |
| Postconditions | *The agent has information about the fines and the customers pay for the fines.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.12): | *Log Out* |
| Summary | *The agent at the end of the shift logs out of the system.* |
| Actors | *Primary Actor: Agent* |
| Preconditions | *The agent should be logged in the system.* |
| Description of the Main Sequence | * *Step 1: The agent goes to their profile and logs out successfully by clicking the log out button.* |
| Description of the Alternative Sequence | * *Step 1: The agent fails to log out.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.* |
| Postconditions | *The agent is logged out of the website.* |

Diagram

Description automatically generated

|  |  |
| --- | --- |
| Use Case (UC\_2.1): | *Log In /Sign in* |
| Summary | The customers need to log in or sign in the system to later use it. |
| Actors | *Primary Actor: Customer* |
| Preconditions | The customer should open the website. |
| Description of the Main Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password/ other required fields.* * *Step 3: Press the log in/sign in button.* * *Step 4: The agent is logged in successfully.* |
| Description of the Alternative Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: The customer could not log in.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | *The customer should be logged in/ signed in.* |

|  |  |
| --- | --- |
| Use Case (UC\_2.2): | *Make reservation* |
| Summary | *The customer should be able to make reservations through the website.* |
| Dependency | *This optional section describes whether the UC depends on other UCs.* |
| Actors | *Primary Actor: Customer* |
| Preconditions | *The customer should have chosen the car.* |
| Description of the Main Sequence | * *Step 1: They open the reservation section on the website.* * *Step 2: They choose the car and the time period.* * *Step 3: They fill in the required information.* * *Step 4: They save the reservation.* |
| Description of the Alternative Sequence | * *Step 1: They open the reservation section on the website.* * *Step 2: They choose the car and the time period.* * *Step 3: The car is not available and the customer does not continue the reservation.* |
| Nonfunctional requirements | *The system should be able to maintain a huge number of reservations.*  *The reservation should be saved within 3 seconds.* |
| Postconditions | *The reservation is saved on the system.* |

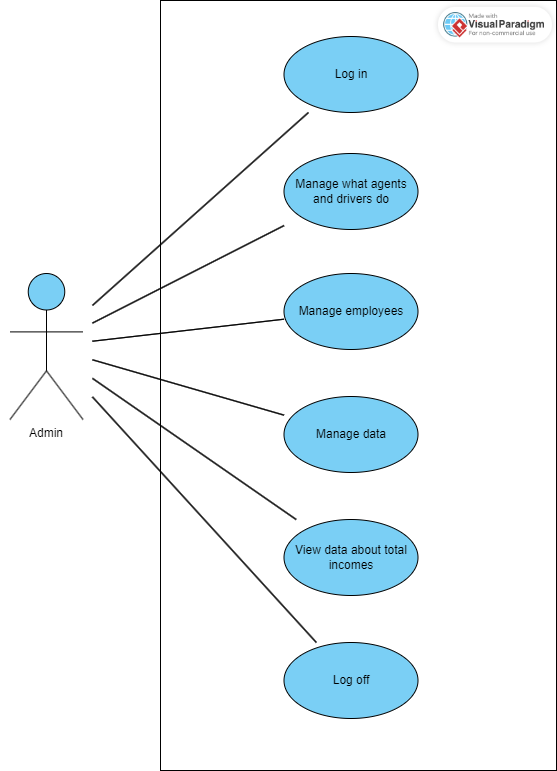
|  |  |
| --- | --- |
| Use Case (UC\_2.3): | *View available cars and services* |
| Summary | *The customer can view the available cars and services in the system.* |
| Actors | *Primary actor: Customer* |
| Preconditions | *Cutomer is logged in the system.* |
| Description of the Main Sequence | * *Step 1: After logging in, the customer clicks the button to view the cars.* * *Step 2: They put in the time period they want to look for.* * *Step 3: The system shows the cars available for that time period.* |
| Description of the Alternative Sequence | * *Step 1: After logging in, the customer clicks the button to view the cars.* * *Step 2: They put in the time period they want to look for.* * *Step 3: There are no available cars for that time period.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The page must load within 2 seconds.* |
| Postconditions | *The customer has access to the available cars and services.* |

|  |  |
| --- | --- |
| Use Case (UC\_2.4): | *Pay for the reservation* |
| Summary | *After making the reservation, the customer should pay for the reservation online using credit card.* |
| Dependency | *To make this use case work, the website should support different online payment ways.* |
| Actors | *Primary Actor: Customer* |
| Preconditions | *The customer should have made a reservation.* |
| Description of the Main Sequence | * *Step 1: The customer makes the reservation.* * *Step 2: The customer clicks the payment way they want to choose.* * *Step 3: The customer follows the step to process the payment.* |
| Description of the Alternative Sequence | * *Step 1: The customer makes the reservation.* * *Step 2: The customer clicks the payment way they want to choose.* * *Step 3: The customer follows the step to process the payment, but it gets cancelled or it is not supported.* |
| Nonfunctional requirements | *The payment ways should be secure for both the agency and the customer.* |
| Postconditions | *The reservation is paid.* |

|  |  |
| --- | --- |
| Use Case (UC\_2.5): | *Cancel reservations* |
| Summary | *The customer may need to cancel the reservation for various reasons.* |
| Actors | *Primary Actor: Customer* |
| Preconditions | *In order to delete the reservation, it should firstly exist and there should be a reason to delete it.* |
| Description of the Main Sequence | * *Step 1: The customer clicks the details of the reservation.* * *Step 2: By clicking the delete button at the end, the reservation is deleted.* |
| Description of the Alternative Sequence | * *Step 1: The customer clicks the details of the reservation.* * *Step 2: The reservation cannot be deleted.* |
| Nonfunctional requirements | *The deletion of the reservation should not affect the other reservations.* |
| Postconditions | *The reservation is deleted from the system.* |

|  |  |
| --- | --- |
| Use Case (UC\_2.6): | *Filter the cars according to different characteristics* |
| Summary | *When searching for a car, the customer should be able to filter the cars using car model, year of production, color, price etc. according to their needs.* |
| Actors | *Primary Actor: Customer* |
| Preconditions | *The system should keep information about the car details to be able to filter cars according to them.* |
| Description of the Main Sequence | * *Step 1: The customer clicks the filter button and chooses the characteristic they want to find.* * *Step 2: Navigate through the generated responses.* |
| Description of the Alternative Sequence | * *Step 1: The customer clicks the filter button and chooses the characteristic they want to find.* * *Step 2: There is no car with those characteristics.* |
| Nonfunctional requirements | *Easily navigable.* |
| Postconditions | *The customer is provided with the cars list that have the required characteristic.* |

|  |  |
| --- | --- |
| Use Case (UC\_1.7): | *Log Out* |
| Summary | *The customer at the end of the reservation logs out of the system.* |
| Actors | *Primary Actor: Customer* |
| Preconditions | *The customer should be logged in the system.* |
| Description of the Main Sequence | * *Step 1: The customer goes to their profile and logs out successfully by clicking the log out button.* |
| Description of the Alternative Sequence | * *Step 1: The customer fails to log out.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.* |
| Postconditions | *The customer is logged out of the website.* |

****

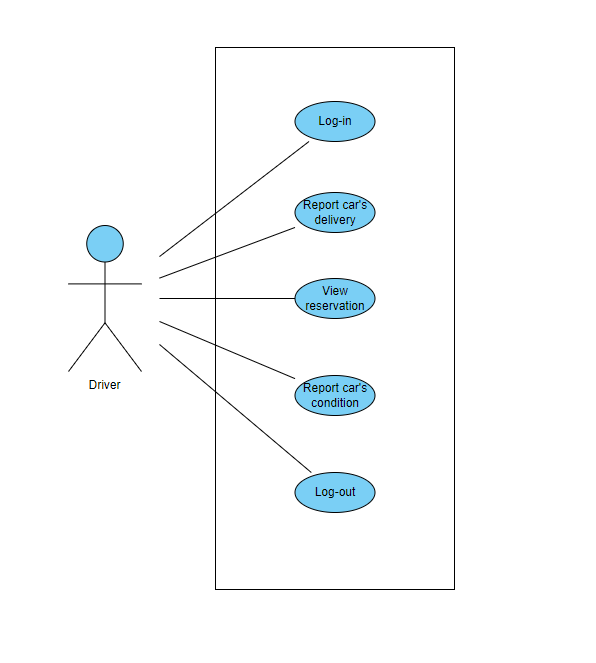
|  |  |
| --- | --- |
| Use Case (UC\_3.1): | *Log In* |
| Summary | The admin needs to log in the system to later use it. |
| Actors | Primary Actor: Admin |
| Preconditions | The admin should open the website |
| Description of the Main Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: Authentication* * *Step 5: The admin is logged in successfully.* |
| Description of the Alternative Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: Authentication* * *Step 5: The admin cannot log in.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | The admin should be logged in |

|  |  |
| --- | --- |
| Use Case (UC\_3.2): | *Manage what drivers and agents do* |
| Summary | *The admin manages all actions of drivers and agents.* |
| Actors | *Primary Actor: admin*  *Secondary Actor: driver, agent* |
| Preconditions | *Driver and agent should do their job.* |
| Description of the Main Sequence | * *Step 1: Admin checks employee list.* * *Step 2: Picks employee.* * *Step 3: Views his/her statistics.* * *Step 4: (Optional) Adds goals.* |
| Description of the Alternative Sequence | * *No alternative sequence* |
| Non-functional requirements | *All actors needs to be connected to the internet* |
| Postconditions | *The admin manages agents and drivers.* |

|  |  |
| --- | --- |
| Use Case (UC\_3.3): | *Manage employees* |
| Summary | The admin can register, modify, delete employees, and he can keep data about them |
| Actors | Primary Actor: admin  Secondary Actor: employees |
| Preconditions | The admin should open the website. |
| Description of the Main Sequence | *● Step 1: Admin checks employee list.*  *● Step 2: Picks employee.*  *● Step 3: Modifies his/her information.* |
| Description of the Alternative Sequence | *● Step 1: Admin checks employee list.*  *● Step 2: Picks employee.*  *● Step 3: Deletes employee.* |
| Description of the Alternative Sequence no 2 | *● Step 1: Admin checks employee list.*  *● Step 2: Makes register request.*  *● Step 3: Registers employee* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | The admin can modify the data of employees |

|  |  |
| --- | --- |
| Use Case (UC\_3.4): | *View data about total incomes* |
| Summary | The admin can view analytical data about his/her business. |
| Actors | Primary Actor: admin |
| Preconditions | The admin should open the website. |
| Description of the Main Sequence | *● Step 1: Admin opens the financial statistics menu.*  *● Step 2: Chooses specific filters.*  *● Step 3: Views data, keeps notes and sets new goals.* |
| Description of the Alternative Sequence | *● No alternative sequence* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | Administrator has analyzed the financial data |

|  |  |
| --- | --- |
| Use Case (UC\_3.5): | *Log Out* |
| Summary | *The admin at the end of the shift logs out of the system.* |
| Actors | *Primary Actor: Admin* |
| Preconditions | *The admin should be logged in the system.* |
| Description of the Main Sequence | * *Step 1: The admin goes to their profile and logs out successfully by clicking the log out button.* |
| Description of the Alternative Sequence | * *Step 1: The admin saves and terminates any tasks that might be still going on.* * *Step 2: The admin goes to their profile and logs out successfully by clicking the log out button.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.* |
| Postconditions | *The admin is logged out of the website.* |

****

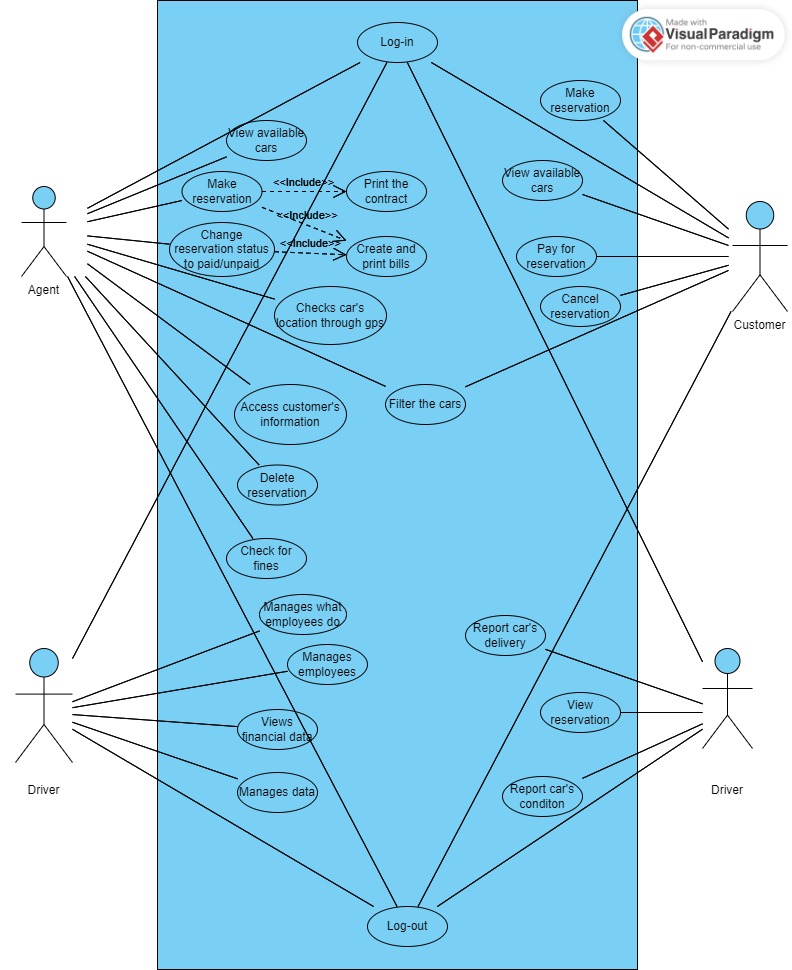
|  |  |
| --- | --- |
| Use Case (UC\_4.1): | *Log In* |
| Summary | The driver needs to log in the system to later use it. |
| Actors | Primary Actor: Driver |
| Preconditions | The Driver should open the website |
| Description of the Main Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: Authentication* * *Step 5: The driver is logged in successfully.* |
| Description of the Alternative Sequence | * *Step 1: Enter the username.* * *Step 2: Enter the password.* * *Step 3: Press the log in button.* * *Step 4: Authentication* * *Step 5: The driver cannot log in.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | The driver should be logged in |

|  |  |
| --- | --- |
| Use Case (UC\_4.2): | *View reservation* |
| Summary | The driver can view reservations in order to make sure he delivers on time and in the correct place |
| Actors | Primary Actor: driver |
| Preconditions | The driver should be logged in the system. |
| Description of the Main Sequence | *● Step 1: Driver opens the reservations menu*  *● Step 2: Chooses the reservation using reservation details.*  *● Step 3: Gets reservation information that he/she needs* |
| Description of the Alternative Sequence | *● No alternative sequence* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | Driver has retrieved the information about time and place of delivery. |

|  |  |
| --- | --- |
| Use Case (UC\_4.3): | *Report cars delivery* |
| Summary | The driver should report about any car he has delivered to the customers |
| Actors | Primary Actor: driver |
| Preconditions | The driver should be logged in |
| Description of the Main Sequence | *● Step 1: Driver opens the reservations menu.*  *● Step 2: Finds the reservation using reservation’s details.*  *● Step 3: Marks the reservation as delivered.* |
| Description of the Alternative Sequence | *● No alternative sequence* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | Driver has reported a successful delivery. |

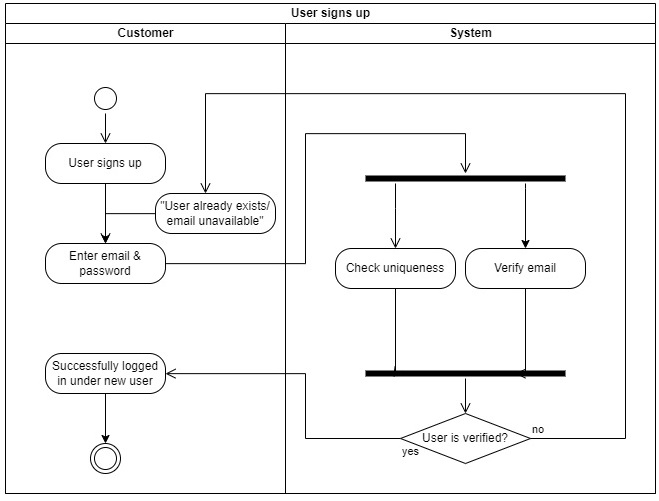
|  |  |
| --- | --- |
| Use Case (UC\_4.4): | *Report the car’s condition* |
| Summary | The driver should report any damage that has happened to the car after the client returns it |
| Actors | Primary Actor: driver |
| Preconditions | The driver should be logged in. |
| Description of the Main Sequence | *● Step 1: Driver opens the cars menu*  *● Step 2: Finds the specific car by its license number*  *● Step 3: Makes a condition report request.*  *● Step 4: Writes any information regarding the car’s condition.* |
| Description of the Alternative Sequence | *● No alternative sequence* |
| Nonfunctional requirements | *The device needs to be connected to the internet.*  *The system must be secure from unauthorized access.* |
| Postconditions | Driver has reported the conditions of the car |

|  |  |
| --- | --- |
| Use Case (UC\_4.5): | *Log Out* |
| Summary | *The driver at the end of the shift logs out of the system.* |
| Actors | *Primary Actor: Driver* |
| Preconditions | *The Driver should be logged in the system.* |
| Description of the Main Sequence | * *Step 1: The driver goes to their profile and logs out successfully by clicking the log out button.* |
| Description of the Alternative Sequence | * *Step 1: The driver saves and terminates any tasks that might be still going on.* * *Step 2: The driver goes to their profile and logs out successfully by clicking the log out button.* |
| Nonfunctional requirements | *The device needs to be connected to the internet.* |
| Postconditions | *The driver is logged out of the website.* |

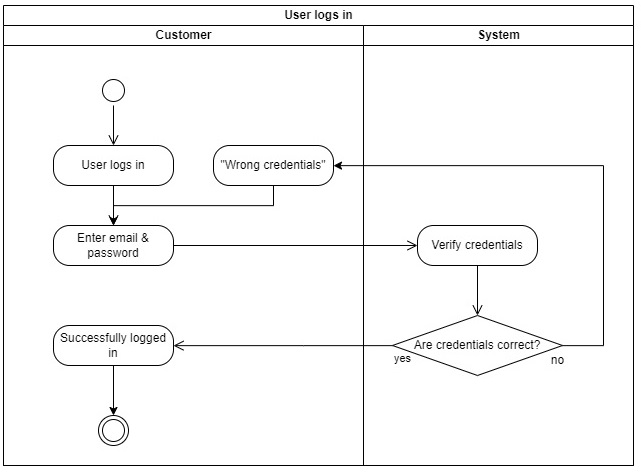
****

* 1. **Activity diagrams**

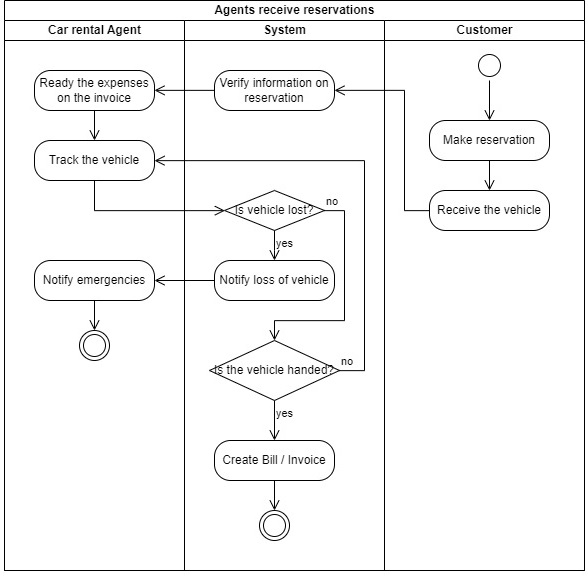
AC\_1: User signs up



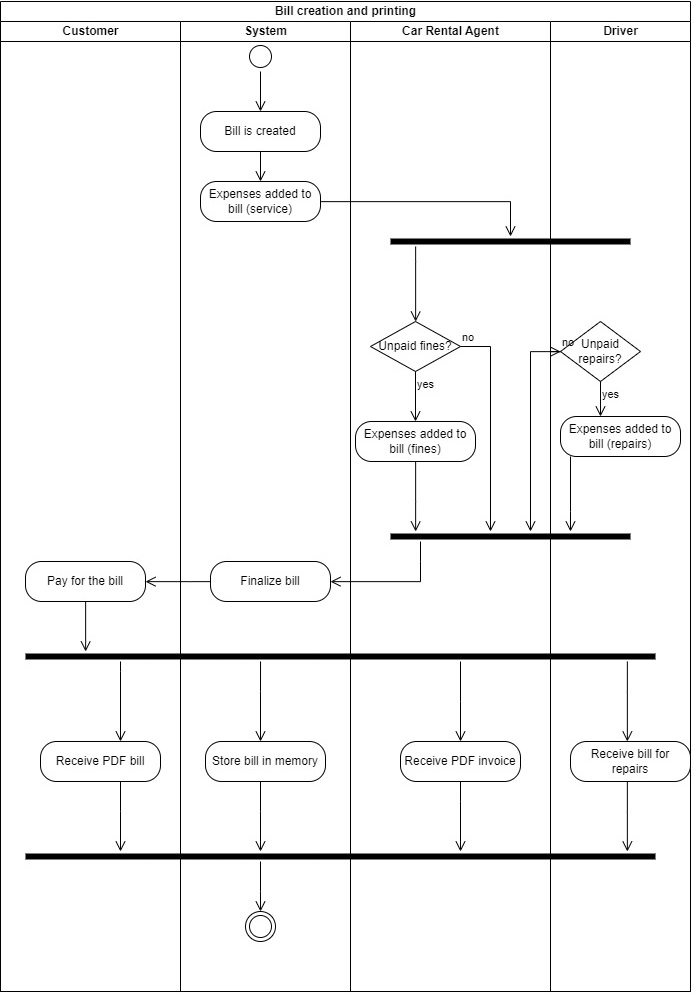
AC\_2: User logs in:



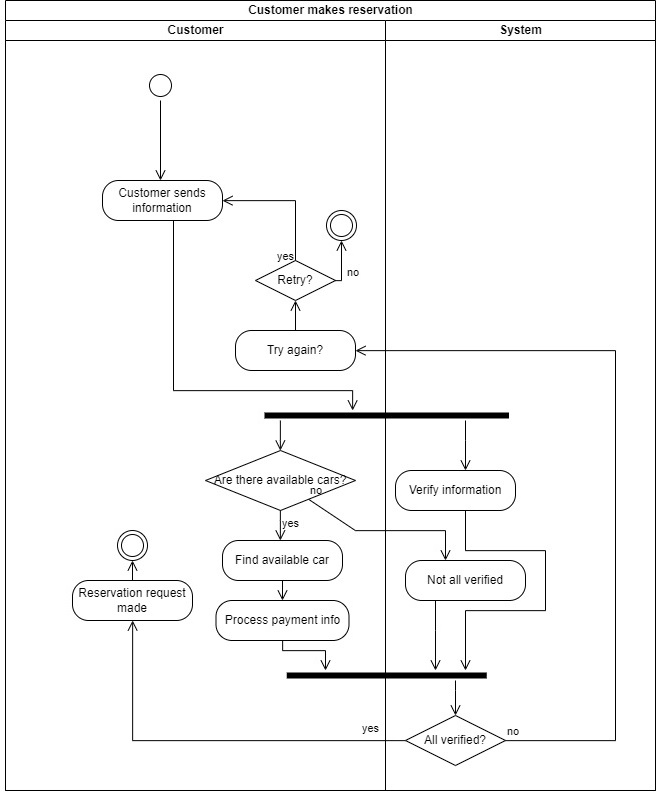
AC\_3: Agents receive reservations:



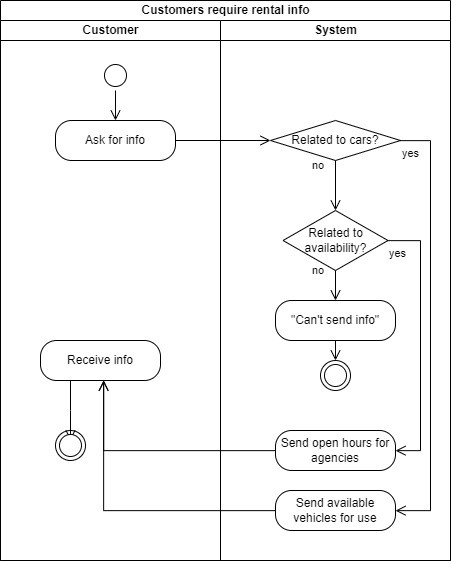
AC\_4: Bill is created:



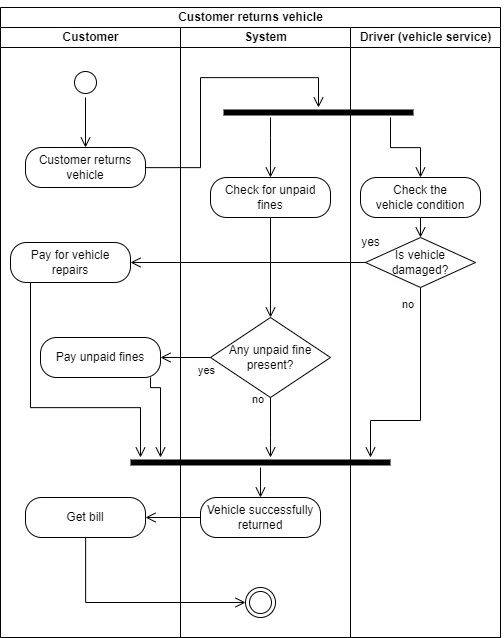
AC\_5: Customer makes reservation:



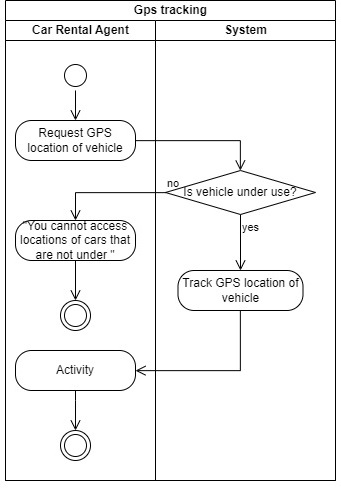
AC\_6 Customer requires rental information:



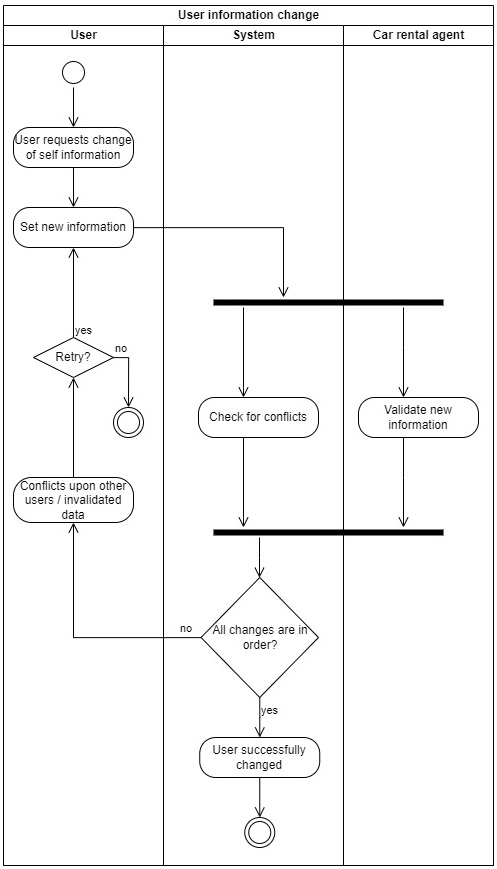
AC\_7: Customer returns vehicle



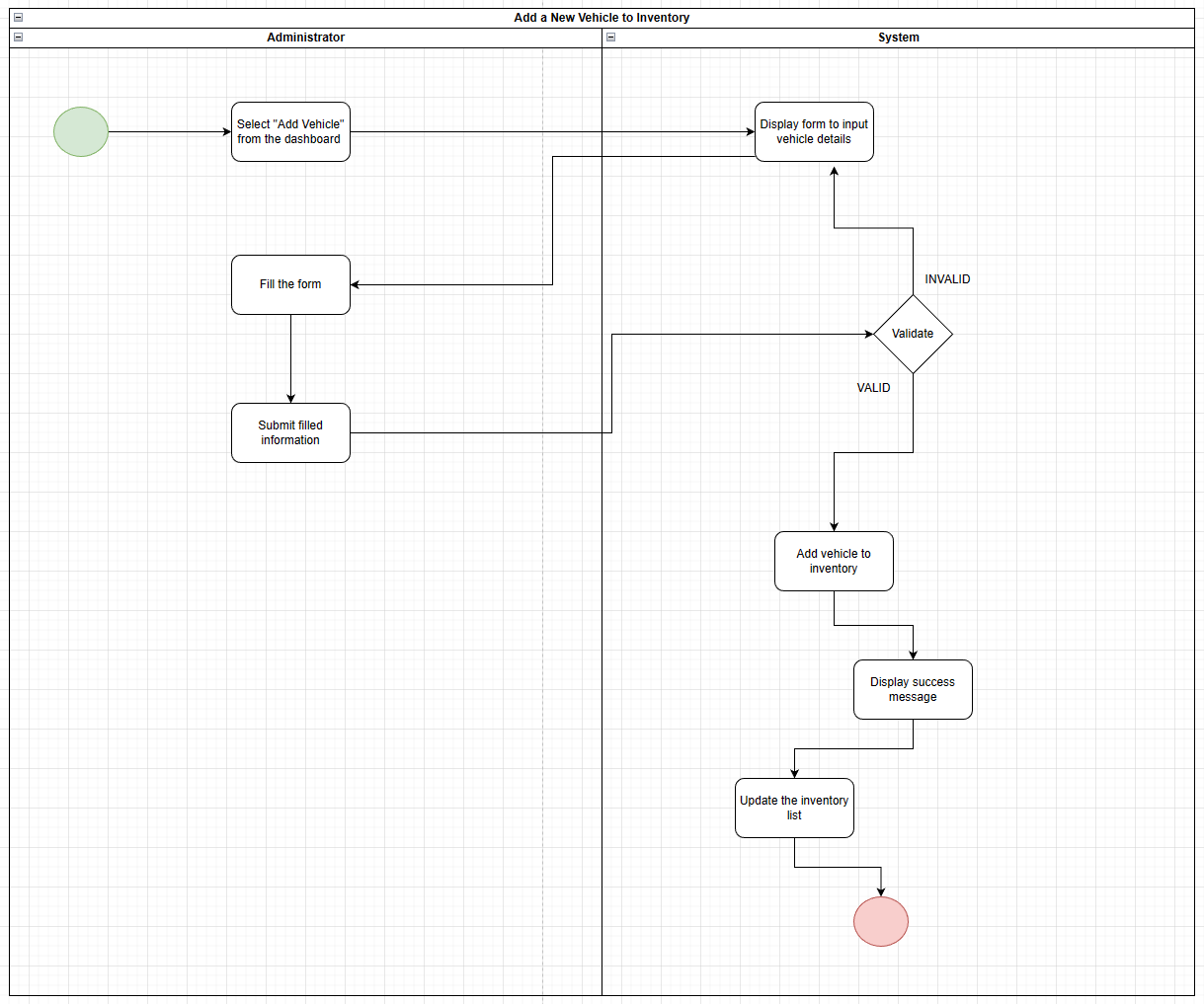
AC\_8: GPS-Tracking



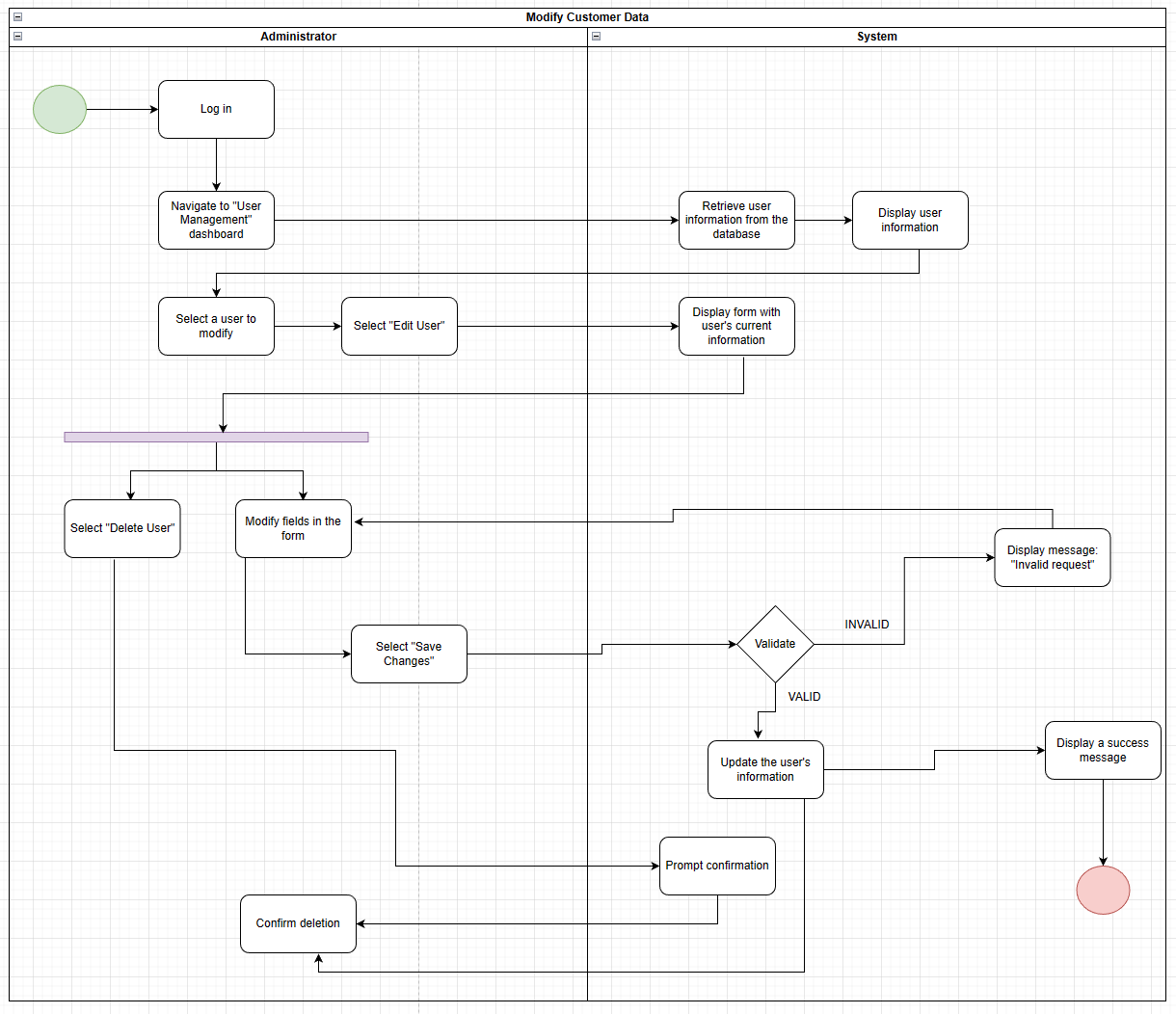
AC\_9: User changes profile information



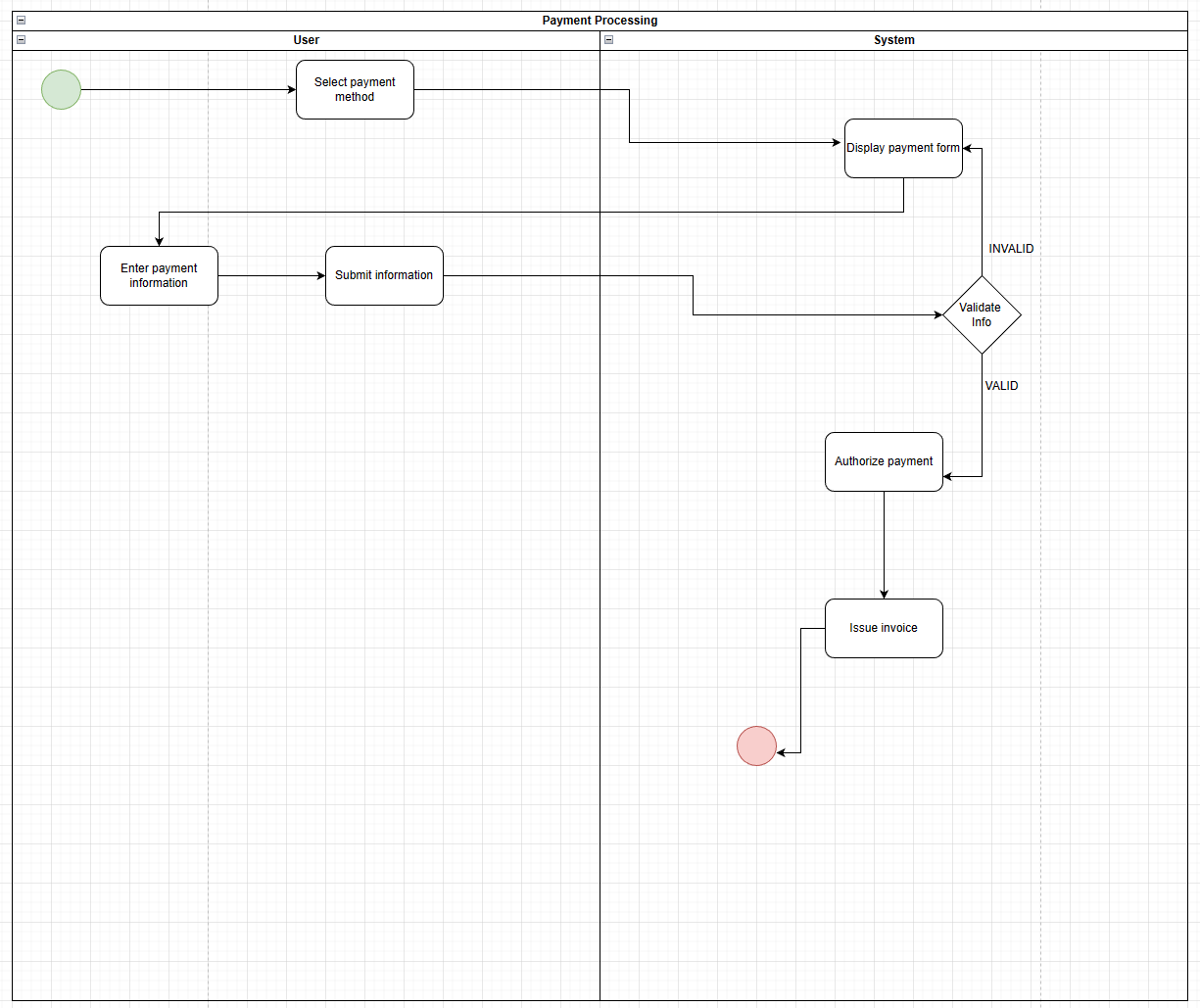
AC\_10: Admin adds new vehicle to inventory



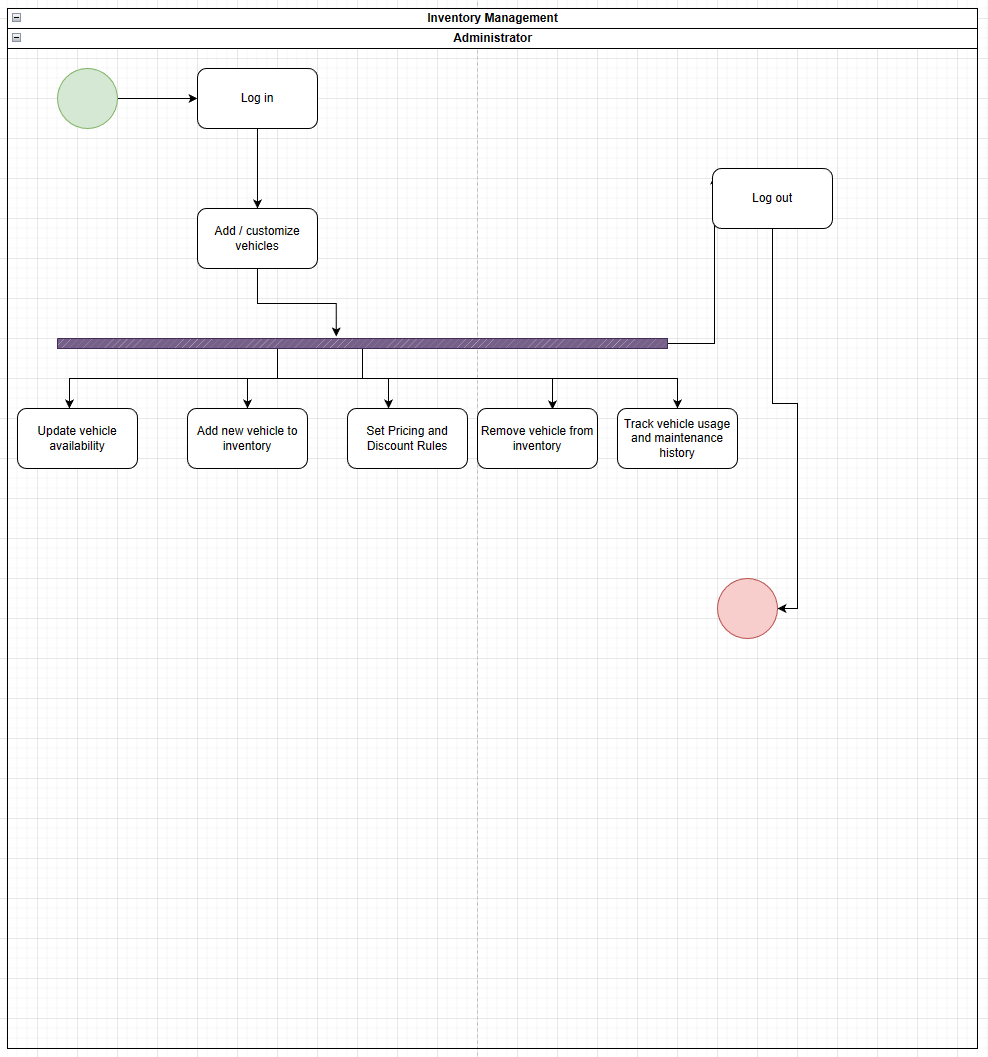
AC\_11: Admin modifies user’s data



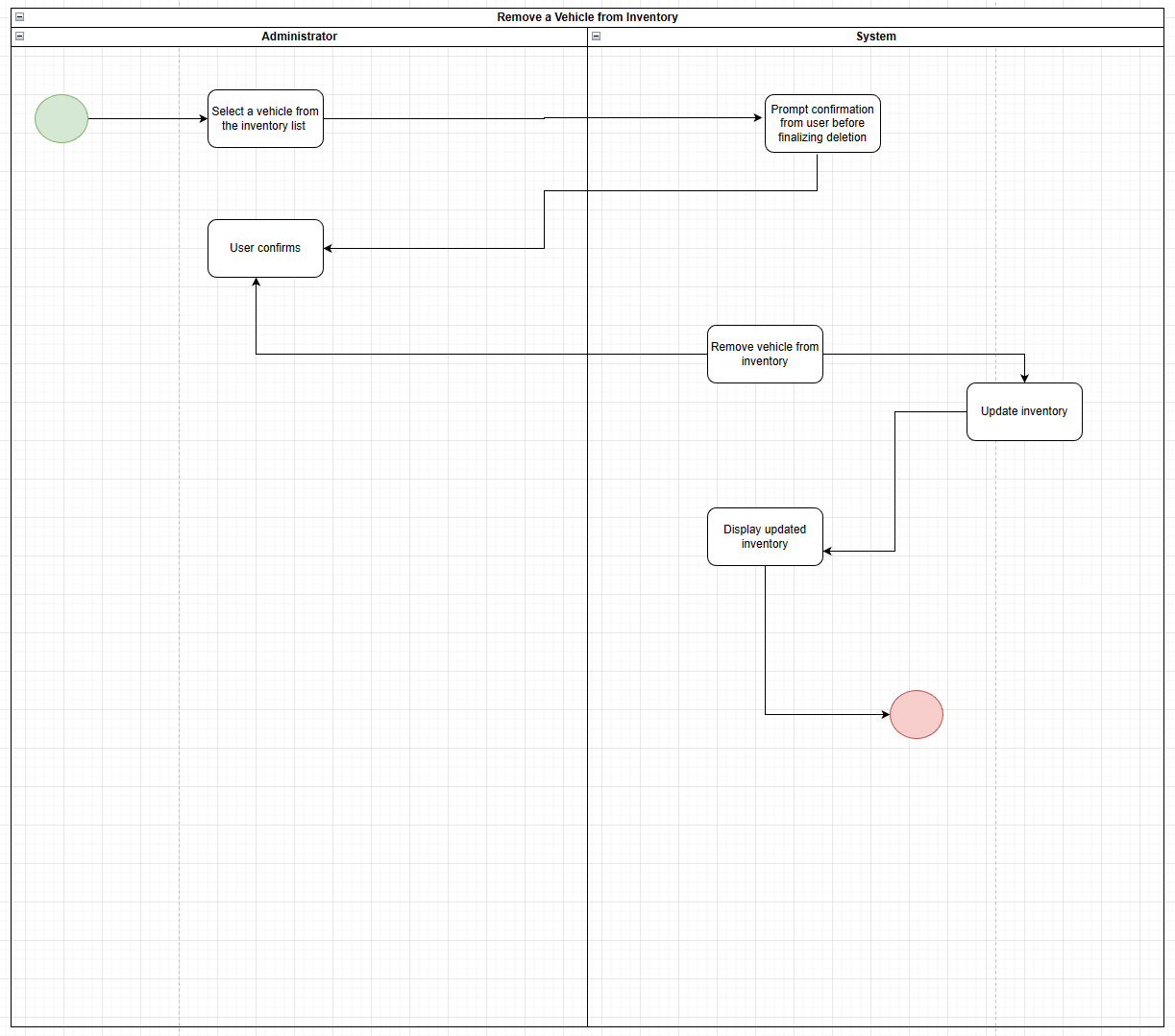
AC\_12: Payment processing



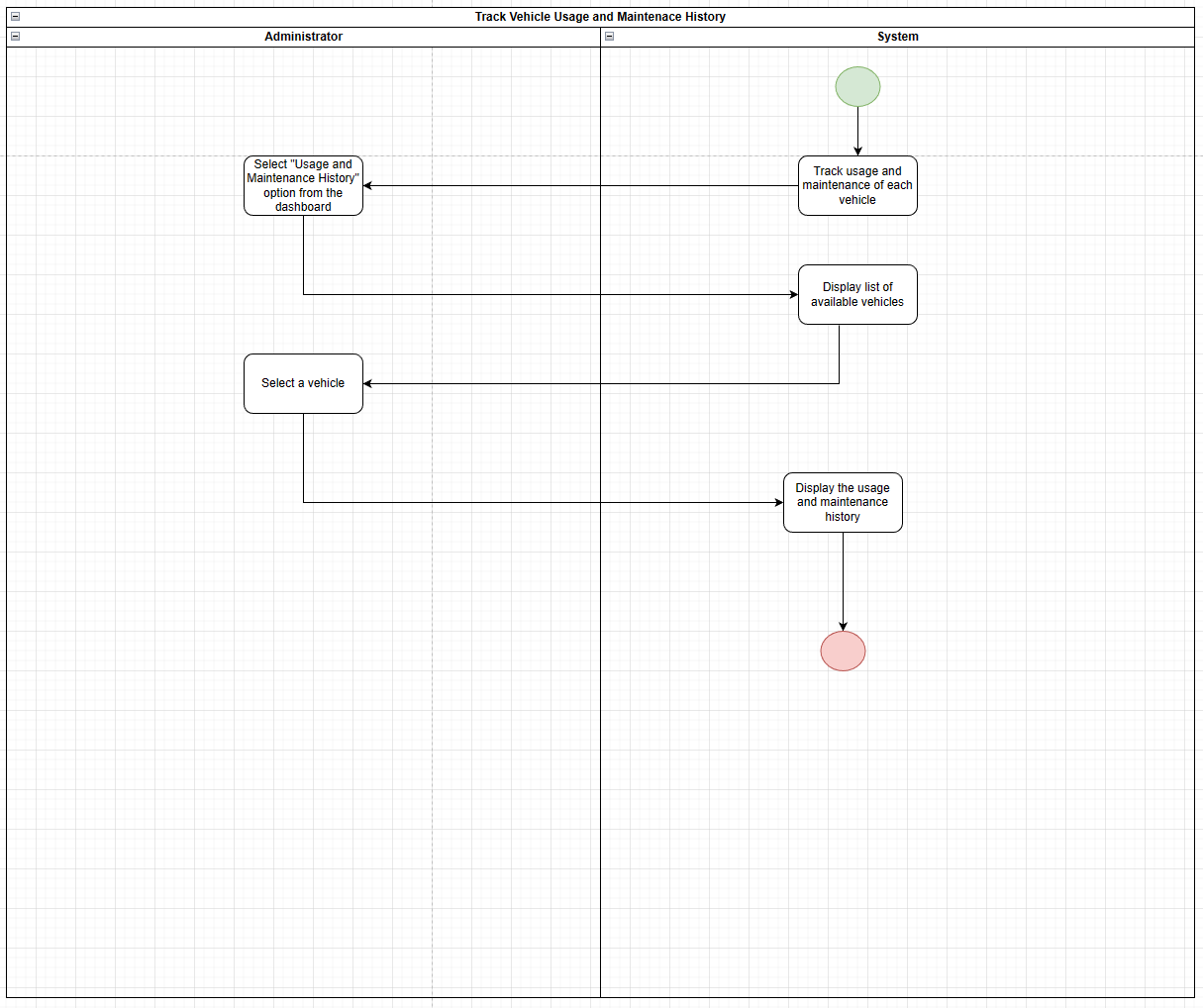
AC\_13: Administrator manages inventory



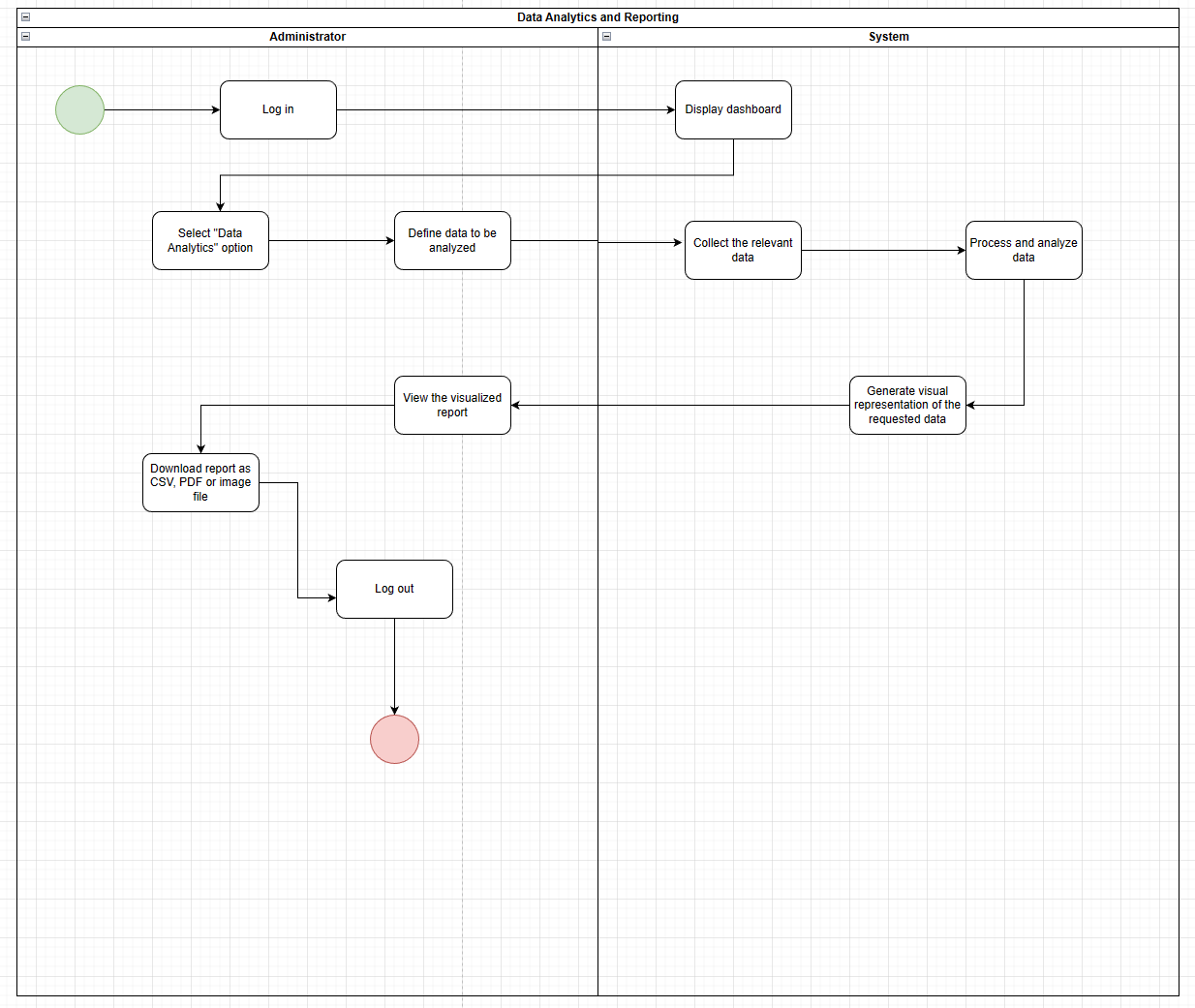
AC\_14: Administrator removes a vehicle from inventory



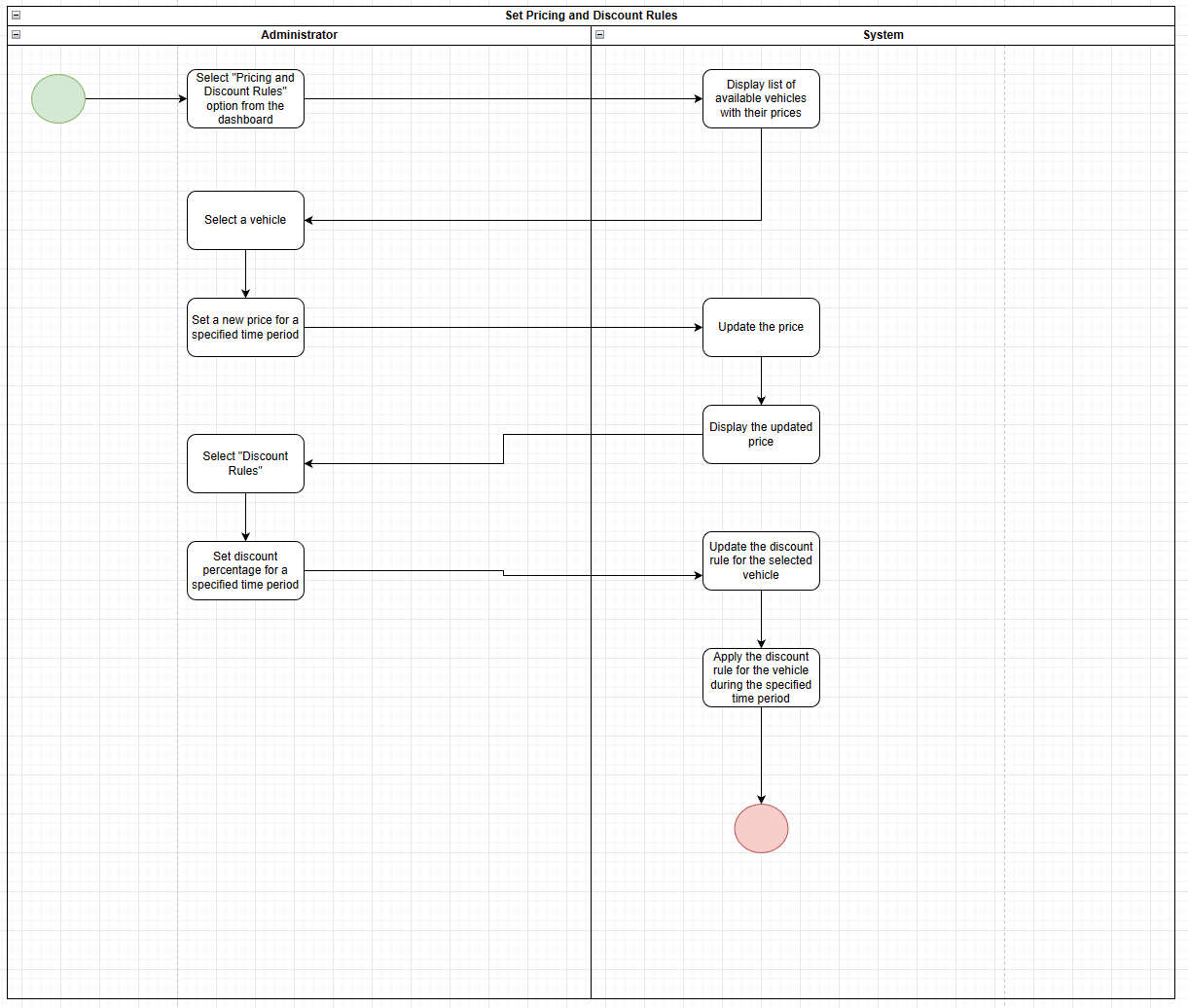
AC\_15: Tracking vehicle usage and maintenance history



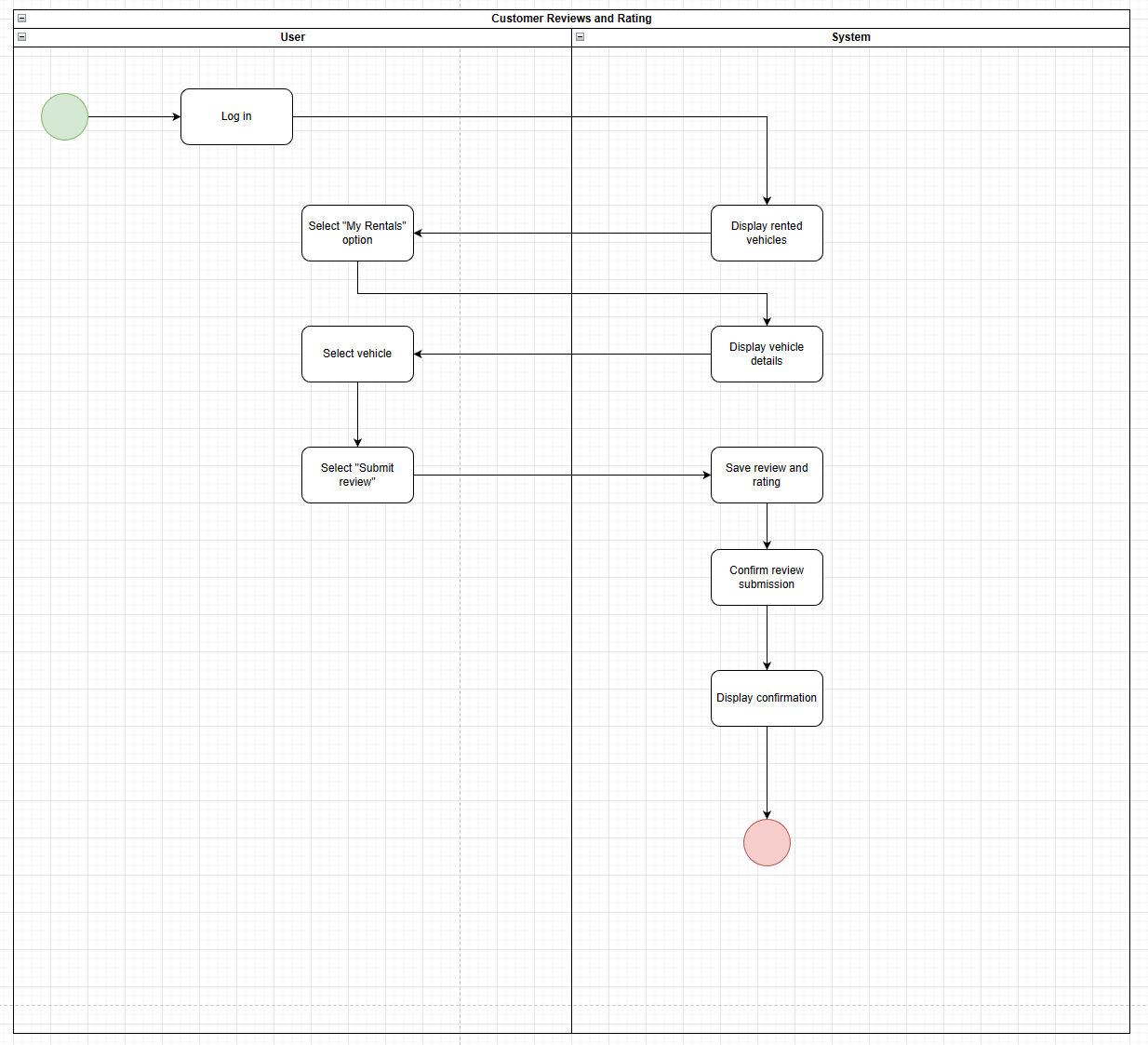
AC\_16: Administrator receives analytic data and reports



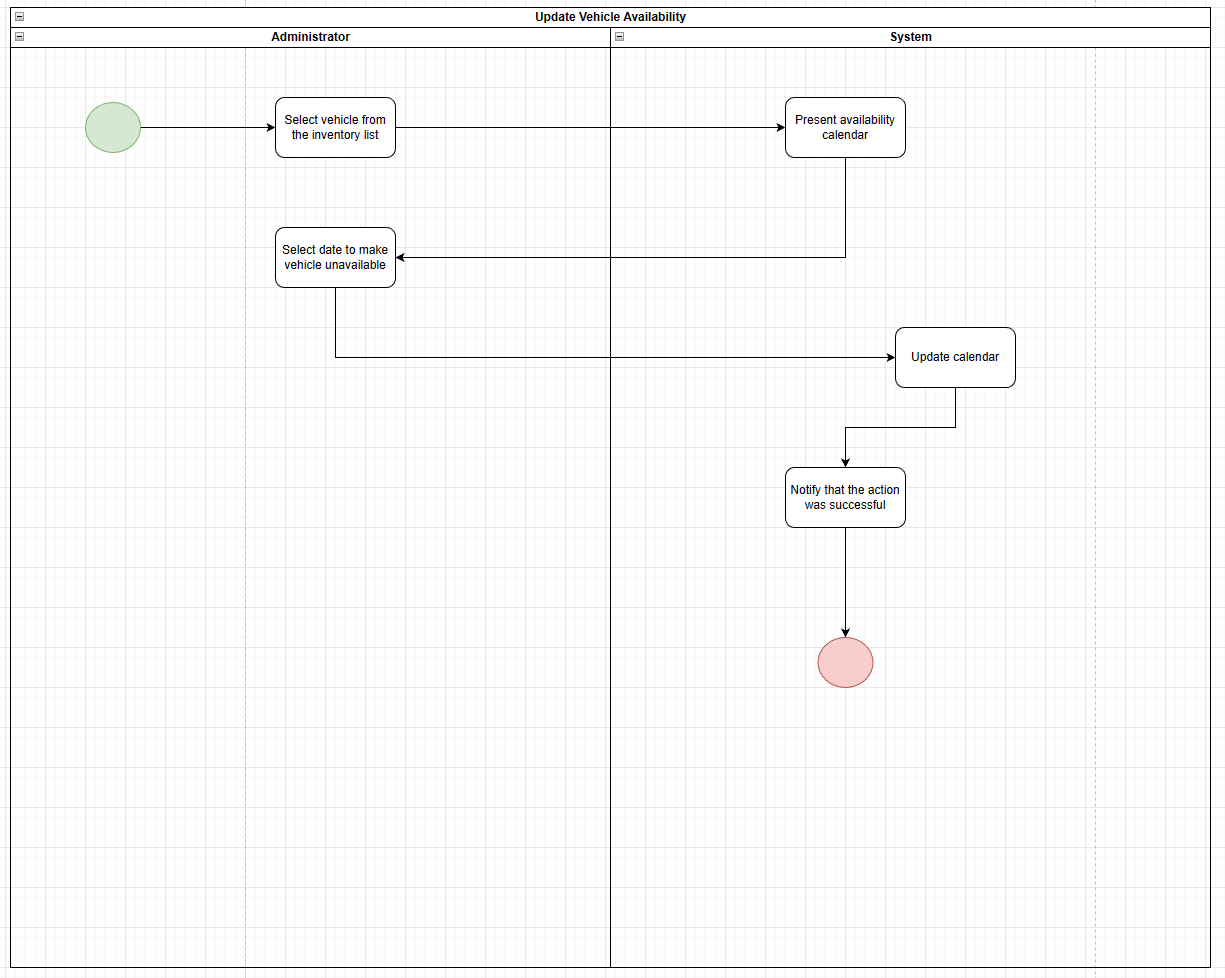
AC\_17: Setting prices and discount rules



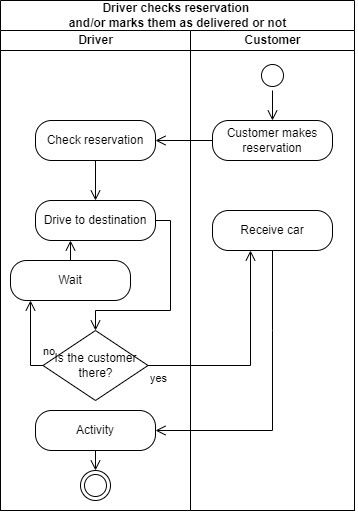
AC\_18: Customer reviews and rates the service

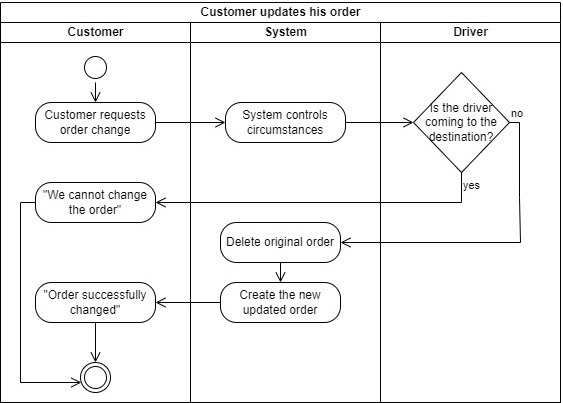


AC\_19: Update car’s availability

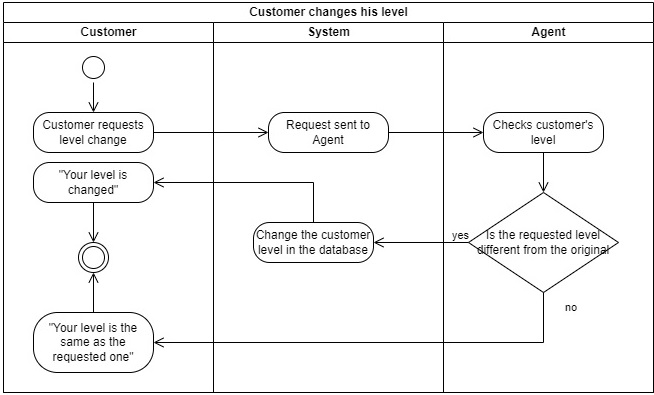


AC\_20: Driver checks reservation and/or marks them as delivered or not

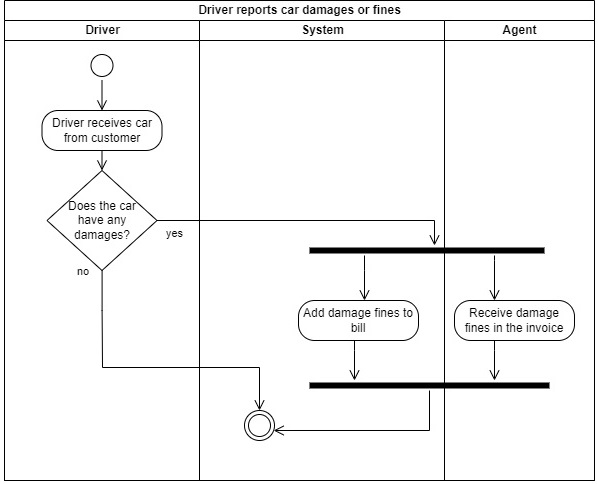


AC\_21: Customer updates his order

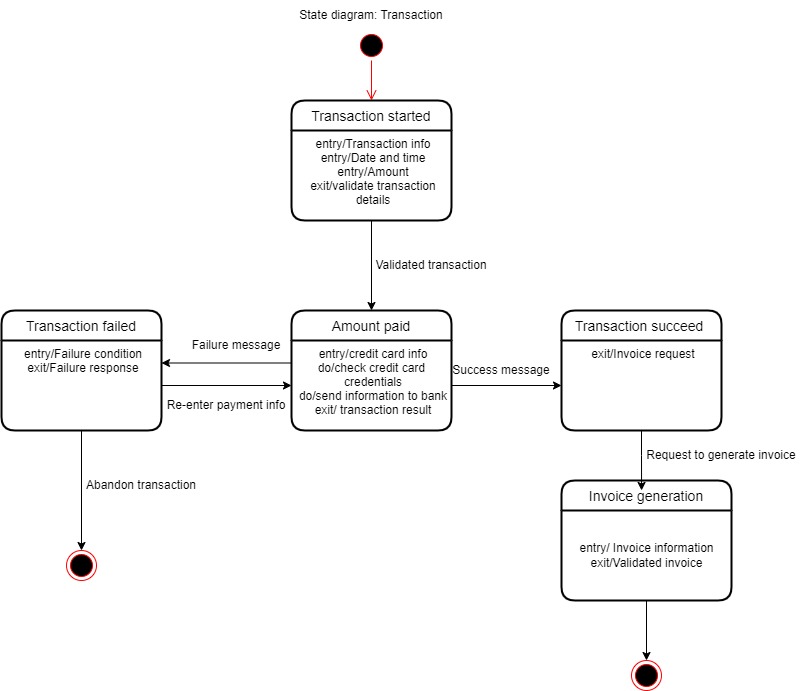
AC\_22: Customer changes his level



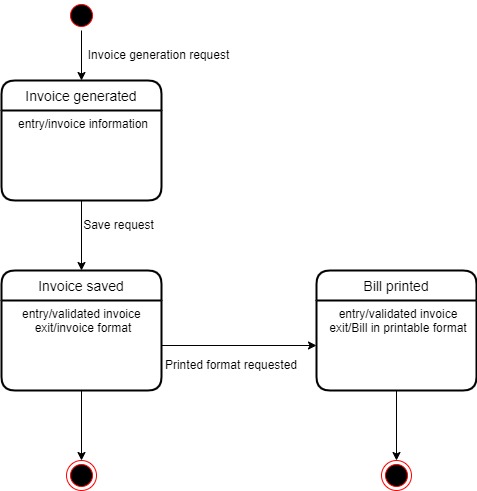
AC\_23: Driver reports car damages or fines

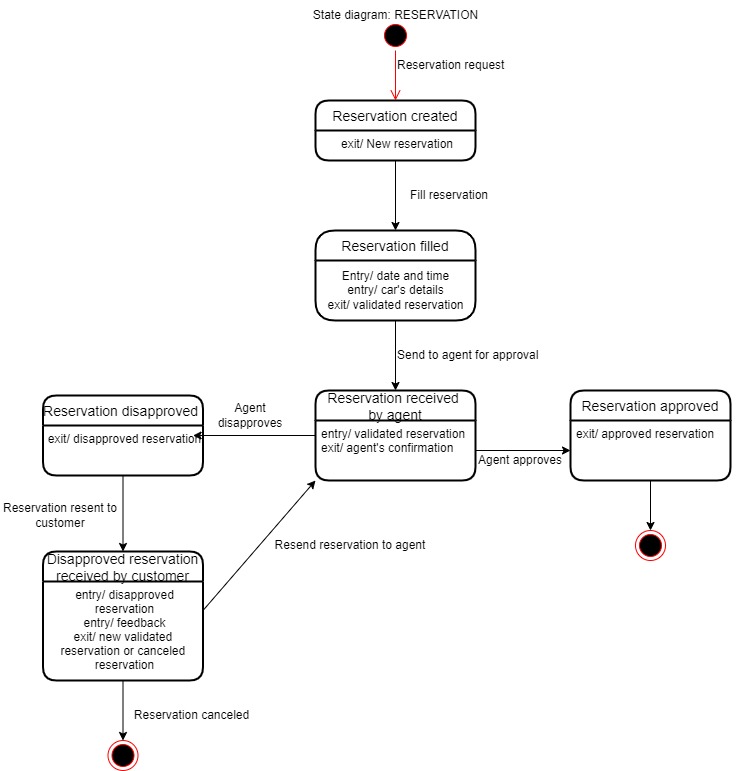


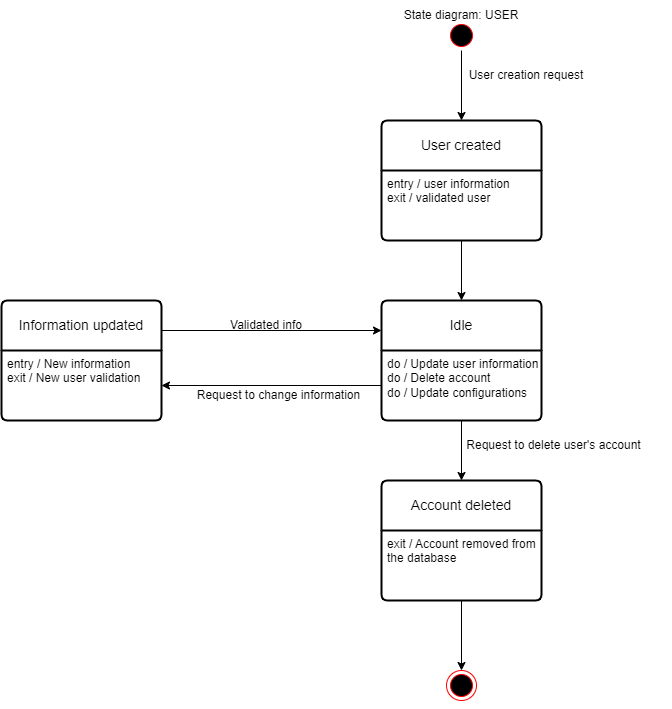
* 1. **State diagrams**

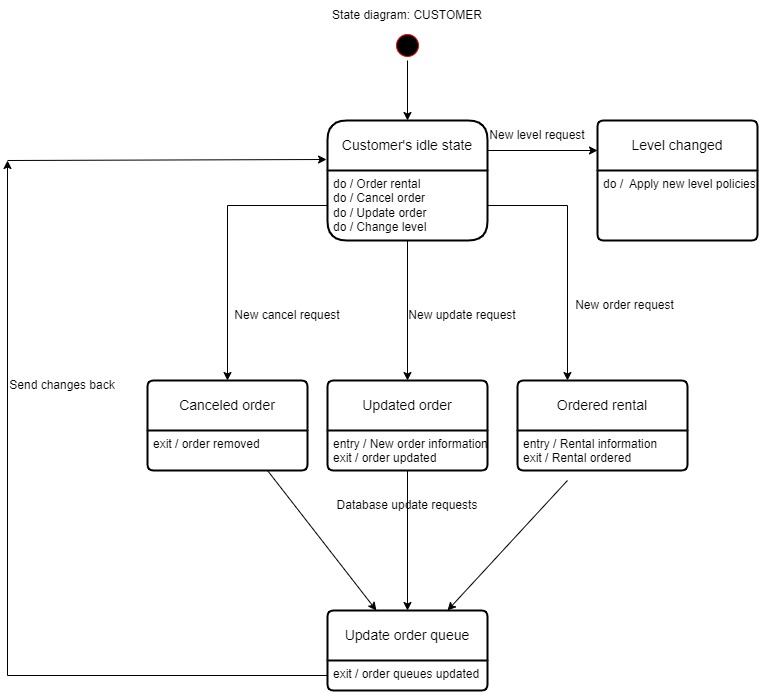
****

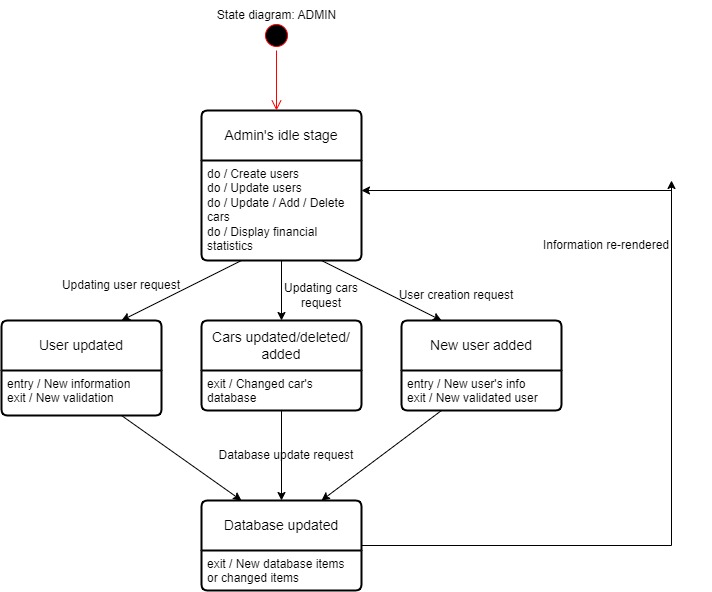
Invoice state diagram

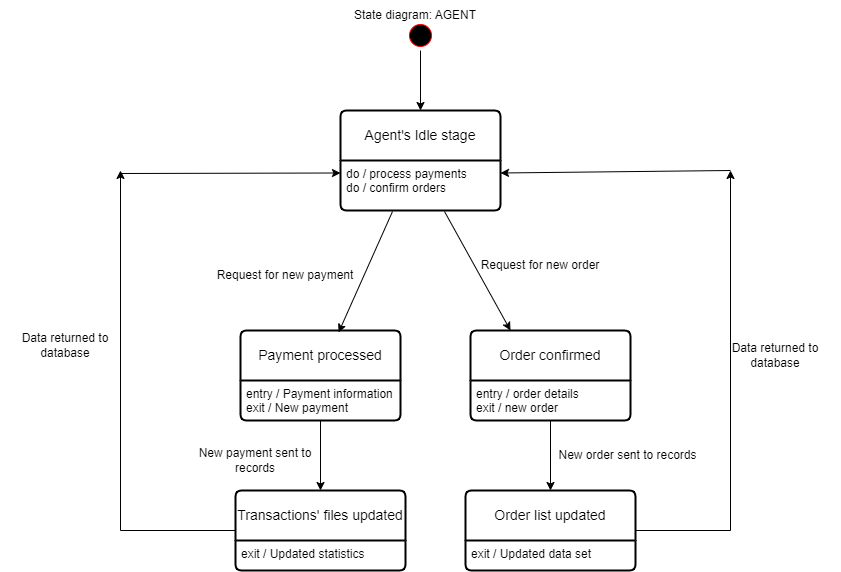
****

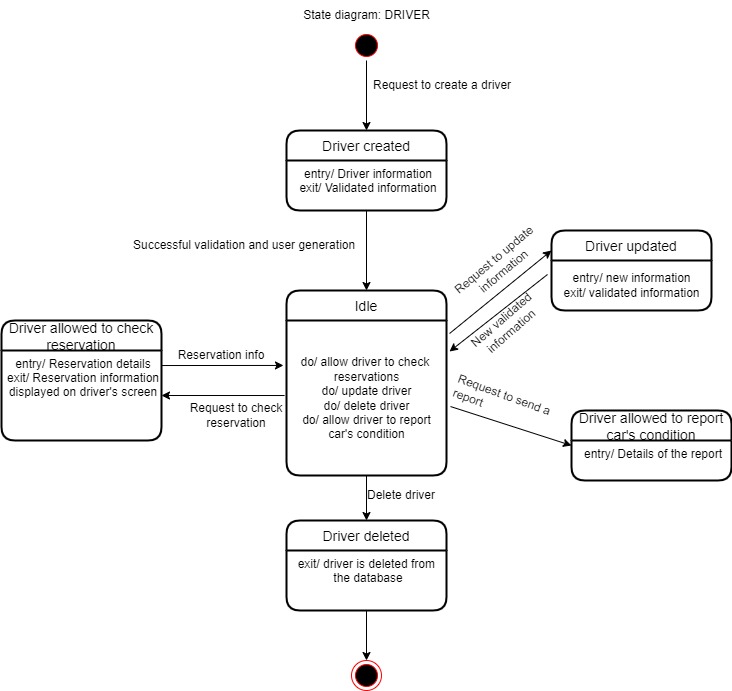


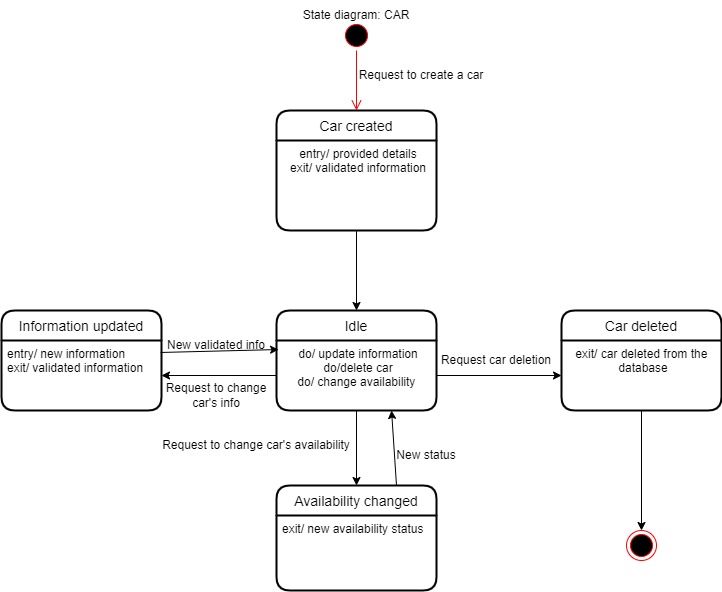












APPENDIX

The appendixes are not always considered part of the actual Requirements Specification and are not always necessary. They may include

* Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
* Supporting or background information that can help the readers of the Requirements Specification;
* A description of the problems to be solved by the system;
* Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the Requirements Specification should explicitly state whether or not the appendixes are to be considered part of the requirements.

1. **Definitions, Acronyms, and Abbreviations**

Define all terms, acronyms, and abbreviations used in this document.

1. **References**

List all the documents and other materials referenced in this document.

1. **Requirements Traceability Matrix**

The following trace matrix examples show one possible use of naming standards for deliverables (FunctionalArea-DocType-NN). The number has no other meaning than to keep the documents unique. For example, the Bargaining Unit Assignment Process Flow would be BUA-PF-01.

For example (1):

| **Business Requirement** | **Area** | **Deliverables** | **Status** |
| --- | --- | --- | --- |
| BR\_LR\_01  The system should validate the relationship between Bargaining Unit/Location and Job Class.---Comments: Business Process = "Assigning a Bargaining Unit to an Appointment" (Priority 1) | BUA | BUA-CD-01  Assign BU Conceptual Design | Accepted |
| BUA-PF-01  Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BUA-PF-01  Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BR\_LR\_09  The system should provide the capability for the Labor Relations Office to maintain the job class/union relationship.---Comments: Business Process = "Maintenance" (Priority 1) | BUA | BUA-CD-01  Assign BU Conceptual Design | Accepted |
| BUA-PF-02  BU Assignment Rules Maint Process Flow Diagram | ReadyForReview |

For example (2):

| **BizReqID** | **Pri** | **Major Area** | **DevTstItems DelivID** | **Deliv Name** | **Status** |
| --- | --- | --- | --- | --- | --- |
| BR\_LR\_01 | 1 | BUA | BUA-CD-01 | Assign BU Conceptual Design | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-DS-02 | Bargaining Unit Assignment DB Modification Description | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-PF-01 | Derive Bargaining Unit-Process Flow Diagram | Accepted |
| BR\_LR\_01 | 1 | BUA | BUA-UCD-01 | BU Assign LR UseCase Diagram | ReadyForReview |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-001 | BU Assignment by PC UseCase - Add Appointment and Derive UBU | Reviewed |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-002 | BU Assignment by PC UseCase - Add Appointment (UBU Not Found) | Reviewed |
| BR\_LR\_01 | 1 | BUA | BUA-UCT-006 | BU Assignment by PC UseCase - Modify Appointment (Removed UBU) | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-CD-01 | Assign BU Conceptual Design | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-DS-02 | Bargaining Unit Assignment DB Modification Description | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-PF-02 | BU Assignment Rules Maint Process Flow Diagram | Accepted |
| BR\_LR\_09 | 1 | BUA | BUA-UCD-03 | BU Assign Rules Maint UseCase Diagram | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-045 | BU Assignment Rules Maint: Successfully Add New Assignment Rule | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-051 | BU Assignment Rules MaintUseCase: Modify Rule | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-053 | BU Assignment Rules MaintUseCase - Review Assignment Rules | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UCT-057 | BU Assignment Rules MaintUseCase: Inactivate Last Rule for a BU | Reviewed |
| BR\_LR\_09 | 1 | BUA | BUA-UI-02 | BU AssignRules Maint UI Mockups | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-021 | BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Success | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-027 | BU Assignment Rules Maint TestCase: Modify Rule - Success | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-035 | BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Error Condition | ReadyForReview |
| BR\_LR\_09 | 1 | BUA | BUA-TC-049 | BU Assignment Rules Maint TestCase: Modify Rule - Error Condition | ReadyForReview |

For example (3):

| **BizReqID** | **CD01** | **CD02** | **CD03** | **CD04** | **UI01** | **UI02** | **UCT01** | **UCT02** | **UCT03** | **TC01** | **TC02** | **TC03** | **TC04** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BR\_LR\_01 |  |  | X |  | X |  | X |  |  | X |  | X |  |
| BR\_LR\_09 | X |  |  | X |  | X |  |  | X |  | X |  | X |
| BR\_LR\_10 | X |  |  | X |  |  |  |  | X |  | X |  |  |
| BR\_LR\_11 |  | X |  |  |  |  |  |  |  |  |  |  |  |

1. **Organizing the Requirements**

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

**By System Mode**

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

**By User Class**

Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

**By Objects**

Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

**By Feature**

A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

**By Stimulus**

Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

**By Response**

Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

**By Functional Hierarchy**

When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

**Additional Comments**

Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.