# Index

[Index 1](#_Toc465528770)

[1 Introduction 3](#_Toc465528771)

[1.1 Purpose 3](#_Toc465528772)

[1.2 Scope 3](#_Toc465528773)

[1.3 Goals (france) 5](#_Toc465528774)

[1.4 Actors (france) 5](#_Toc465528775)

[1.5 Definitions and abbreviations (france) 5](#_Toc465528776)

[1.5.1 Definition 5](#_Toc465528777)

[1.5.2 Abbreviations 5](#_Toc465528778)

[1.6 Stakeholders 5](#_Toc465528779)

[1.7 Reference documents 5](#_Toc465528780)

[1.8 Overview 6](#_Toc465528781)

[2 Overall description 6](#_Toc465528782)

[2.1 Product perspective 6](#_Toc465528783)

[2.2 Product functions 7](#_Toc465528784)

[2.2.1 Use Case Diagrams 8](#_Toc465528785)

[2.3 User characteristics 10](#_Toc465528786)

[2.4 Constraints 10](#_Toc465528787)

[2.4.1 Regulations 10](#_Toc465528788)

[2.4.2 Reliability requirements 10](#_Toc465528789)

[2.4.3 Hardware limitation 10](#_Toc465528790)

[2.4.4 Criticality 10](#_Toc465528791)

[2.4.5 Performance requirements 10](#_Toc465528792)

[2.4.6 Security 10](#_Toc465528793)

[2.5 Assumptions and Dependencies 10](#_Toc465528794)

[3 Specific Requirements 11](#_Toc465528795)

[3.1 External Interface Requirements 11](#_Toc465528796)

[3.1.1 User Interfaces 11](#_Toc465528797)

[3.1.2 Hardware Interfaces 11](#_Toc465528798)

[3.1.3 Software Interfaces 11](#_Toc465528799)

[3.1.4 Communication Interfaces 11](#_Toc465528800)

[3.2 Functional Requirements 11](#_Toc465528801)

[3.2.1 Sign Up 14](#_Toc465528802)

[3.2.2 Login 15](#_Toc465528803)

[3.2.3 User Account Management 16](#_Toc465528804)

[3.2.4 Research Car 17](#_Toc465528805)

[3.2.5 Select car 18](#_Toc465528806)

[3.2.6 Reserve car 18](#_Toc465528807)

[3.2.7 Delete reservation 19](#_Toc465528808)

[3.2.8 Unlock the car 20](#_Toc465528809)

[3.2.9 View car status 20](#_Toc465528810)

[3.2.10 View charge during the trip 21](#_Toc465528811)

[3.2.11 Enable “Money saving” option 22](#_Toc465528812)

[3.2.12 Plug-in the car in order to get the discount 23](#_Toc465528813)

# Introduction

## Purpose

This document represents the Requirements Analysis and Specification Document (RASD). Its aim is to capture all the functional and non-functional requirements that the system-to-be must respect, in order to satisfy the stakeholders goals, under certain domain properties. This document also contains Use Case Diagrams, Sequence Diagrams and Class Diagrams that can be useful to better understand how the system is organized. Further, this document is a valid basis for system testing, verification and validation and has also a contractual value.

## Scope

The aim of our project is to develop a digital management system for a car-sharing service that exclusively employs electric cars. The system should provide the functionality normally provided by car-sharing services. Users must be able to register to the system by providing their credentials and payment information, then they receive back a password that can be used to access the system. Registered users must be able to find the locations of available cars within a certain distance from their current location or from a specified address. The system provides also the possibility to reserve a single car, but with some constraint: for example, if a car is not picked up, the user must pay a fee. On the other hands, if a user reaches a reserved car, he must be able to tell the system he’s nearby his reserved car, so the car will be unlocked and the user can enter and start his rent.

Car-sharing system initiate the charging of money as soon as the engine ignites, and the system starts charging the user for a given amount of money per minute. Indeed, the user is notified of the current charges through a screen on the car. The system stops charging the user as soon as the car is parked in a safe area and the user exits the car.

The set of safe areas for parking cars is predefined by the management system, so we are able to contact a database in order to catch some information about the current position of the car, and then the system can decide if it is parked in a safe area.

Although, the system must be able to define certain user’s behaviour with the car-sharing services and apply some discount (or charging) in consequence of determinate action.

Users will be able to use a mobile application for use the car-sharing services and register himself during the first rent, and to register himself with a web application created to improve the comfort of registration.

## Goals (france)

## Actors (france)

## Definitions and abbreviations (france)

### Definition

### Abbreviations

## Stakeholders

The stakeholders of our project are:

* **Users**: they are involved in our project because they will use our system, in particular they will have the possibility to choose between the web application or the mobile application.
* **Car-sharing service**: our system will offer a web application and a mobile application, developed to manage in a better way all the car-sharing system provided by related company.
* **System administrators**: they are the managers of our system.
* **Testers**: they should check if our system respects all the requirements that have been identified.

## Reference documents

* International standard IEEE Systems and software engineering: Life cycle processes — Requirements engineering (ISO/IEC/ IEEE 29148).
* AA 2016-2017 Software Engineering 2 — Project goal, schedule, and rules.

## Overview

Our document is organized in four main parts:

* **Introduction**: in this section, we give an overview of the scope and goals of our system-to-be. We also identify the main actors that will be involved in our system and give the basic definitions of some words we will often use in this document.
* **Overall description**: in this part, we try to focus our attention on constraints and assumptions concerning our system-to-be and the world around it. This section also considers some possible future implementations that could be added to our system.
* **Specific requirements**: this section is the body of our document. All the specific requirements that our system need are described here and they are associated with different kinds of diagrams, in order to create a model of the real system.
* **Bibliography**: in this part, we specify the documents or books we have referred to.

# Overall description

## Product perspective

The product is a web based application that interacts with a back-end server that handles data and stores them into a DBMS. The client application is supposed to be developed in both web and mobile form, but they include some difference. There are some main features included:

* Multi-platform: the back-end system can be used to develop the mobile and web applications.
* User account: the system allows the user to create an account in the system and provide features of updating and viewing profiles.
* Number of users being supported by the system: though the number is precisely not mentioned, the system is able to support a large number of online users at a time.
* Car reservation: the system allows the reservation of a car, locking the car until the user reach the car, or otherwise until the reservation is cancelled.
* “Money saving” option: the system can help users with some useful advice, in order to obtaining discounts thanks to good behaviour. If the user enable this option, the system asks for a specific destination and calculate some option for leave the car in a “safe area” plugged into the electric grid.

## Product functions

The main function of the systems is to allow users to reserve an electric car, provided by our car-sharing service. The user can also manage his/her reservation, deleting or confirming it. System encourages users’ environmental friendly behaviour providing them discount in case of trip with at least 2 passengers. It also provides special deals if the users take care of plugging-in the car on the electric grid because he/she contribute to the wellness of the system.

* Users:
  + Sign up into the system (simo)
  + Log into the system (simo)
  + Manage account (simo)
  + Research cars (gian)
  + Select car (gian)
  + Reserve car (gian)
  + Delete reservation (gian)
  + Unlock the car in order to use it (gian)
  + View car status (france)
  + View charge during the trip (france)
  + Enable “Money saving” option (france)
  + Plug-in the car in order to get the discount (france)
  + Visualize “trip review” (simo)
  + Conclude the rent and pay (simo)

### Use Case Diagrams







## User characteristics

We expect that our user’s application will be installed by users who are looking for an easy way to research or reserve a car, with the formula of car-sharing services. The system provide the possibility to reserve a car, check their reservations whenever they want, spot some car roundabout through user’s position, check the discount with “money saving” option and make more comfortable the payment of the service. The system is developed in order to make the user comfortable with the use of the mobile application, with the exception of the registration, where the user can use also web application, that is more conformable because are requested some personal data, that could be difficult to insert with the keyboard of mobile phones.

## Constraints

### Regulations

### Reliability requirements

### Hardware limitation

### Criticality

### Performance requirements

### Security

## Assumptions and Dependencies

# Specific Requirements

## External Interface Requirements

### User Interfaces

### Hardware Interfaces

### Software Interfaces

### Communication Interfaces

## Functional Requirements

















### Sign Up

Guests can subscribe to the system via mobile or web application. In both cases they have to fill a form with their personal data and authorize the recipient to use and process their personal details. If the guest accepts the conditions then he can complete the registration, otherwise it is cancelled. Once a guest send the registration, the system checks the consistency of the data inserted by the guest and if everything is correct a mail and a message containing two codes are sent to the new user. Then the guest must insert in a new page the two codes he received and perform a submission, if the data he has submitted are right then the system complete the registration of the user.

|  |  |
| --- | --- |
| Actor | Guest |
| Goal |  |
| Input Condition | The guest isn’t registered to the website |
| Event Flow | 1. The guest enters the website;    1. The guest enters the application; 2. The guest clicks on the “SIGN UP” button; 3. The guest fills in the form where he has to write:    1. Name    2. Surname    3. Email    4. Username    5. Password    6. Date of birth;    7. Telephone number    8. Address    9. Picture    10. **Driving license data**:        1. date of achivement        2. expire date        3. number        4. category    11. **Credit card data**        1. number        2. expire date        3. CVV 4. The guest accept the conditions of the system; 5. The guest clicks the “DONE” button; 6. The system send an email to the user and a message on the mobile phone with two code in order to check the validity; 7. User insert in the application the codes and submit the request; 8. The system shows the home page of the user logged in; |
| Exit Condition | Registration is successfully done. |
| Exception | An exception can be caused if the username the guest inserts already exists or if some field that are not optional aren’t filled. Another possibility of failure happen if the user doesn’t accept the conditions provided by the system. There are also exceptions If the user doesn’t insert the rights codes that were sent to the user and in case the information related to the credit card and driving license are wrong. |

### Login

Users must be logged in the system in order to use the functionalities of the application. During the login process users must insert the username and the password. If the data they insert are right, then the users are redirected to their account page. In the 'user login' page, guests can be redirected to the 'registration' page.

|  |  |
| --- | --- |
| Actor | User |
| Goal |  |
| Entry Conditions | User has successfully signed up to the system |
| Event Flow | 1. The user enters the application. 2. The user fills in the text fields in the home page with username and password. 3. The user tick the option in order to be logged even if he exit from the application.    1. The user doesn’t tick the above option because he is not interest in this feature 4. The user clicks on the “LOG IN” button. |
| Exit Condition | The system shows the user his personal page. |
| Exception | The password and/or username inserted by the user are wrong. The System shows an error message to the user. |

### User Account Management

After the login process, users are redirected to their main page. Here they have the possibility to choose different actions to perform. For example, they can reset their password, change personal information, such the credit card associated to their account or update the driver license. Users can also delete their account.

|  |  |
| --- | --- |
| Actors | Logged User |
| Goal |  |
| Input Condition | The user is logged into the system |
| Event Flow | 1. The user select the personal a 2. The system shows at the users the modification form 3. The user modifies the chosen fields 4. The user submit the modifications 5. The system update the user’s info |
| Output Condition | The user’s profile is modified |
| Exception | The user doesn’t submit the modifications; the new modifications are invalid. In both cases the new modifications are not applied |

### Research Car

When a user runs the application, he should be able to navigate the map in order to find the best alternatives for his/her choice. He should be able to set his current position with GPS or insert one manually. Map should allow user to zoom in and out to make the research easier and faster.

|  |  |
| --- | --- |
| Actor | User |
| Goal | Goal n° # |
| Input condition | User starts mobile application |
| Event flow | 1. When the application is started, a map immediately appears on the screen. 2. Cars are displayed in the map as icon. 3. Icons of available cars are green colored, while icons of reserved but not picked-up cars are red. 4. Picked-up cars are not shown in the map 5. User can navigate the map looking for the nearest available car. |
| Output condition | User selects the car he/she would like to reserve, so he/she is redirected to reservation page. |
| Exception | There are no available cars in the area. |

### Select car

To find out the best solution, user should be able to select a car to check out its status, including charge and position. By selecting car, user should also be able to reserve the car.

|  |  |
| --- | --- |
| Actor | User |
| Goal | Goal n° # |
| Input condition | User finds an available car in the map |
| Event flow | 1. User clicks on the interested icon. 2. User can reserve the car while check its status, that includes its charge, current position and address. |
| Output condition | User decides whether going on through reservation with the selected car or going back to the map to look for another car. |
| Exception | There are no available cars in the area. |

### Reserve car

As soon as user selected a car, he/she should be able to reserve it.

|  |  |
| --- | --- |
|  |  |
| Actor | User |
| Goal | Goal n° # |
| Input condition | User selected an available car. |
| Event flow | 1. Since user double-checked car’s status, he/she is ready to reserve it. 2. User can easily reserve the car by clicking the corresponding button. (i.e. “RESERVE CAR”) |
| Output condition | Application informs the user about the outcome of reservation. |
| Exception | The car is temporarily unavailable. |

### Delete reservation

Since a user reserved a car, he should have the possibility to decline the reservation. In this situation, application shows the user through the map the way to reach the car, and gives him/her to options: first one is to unlock the car, when he/she is nearby; the second one is to decline reservation: this option can be selected from any user’s position, since closeness is not requested.

|  |  |
| --- | --- |
| Actor | User |
| Goal | Goal n° # |
| Input condition | User reserved a car. |
| Event flow | 1. Application shows to the user status of reservation. 2. User can decline his/her reservation clicking “DECLINE RESERVATION” button. |
| Output condition | User is redirected to the map. |
| Exception | The car is temporarily unavailable. |

### Unlock the car

Since a user reserved a car, he should be able to unlock the car, when he/she reaches it. The car’s system should be able to detect user’s position: if he/she is near enough, it should be possible to unlock the car by clicking on the related button on mobile application.

|  |  |
| --- | --- |
| Actor | User |
| Goal | Goal n° # |
| Input condition | User reserved a car. |
| Event flow | 1. Application shows to the user status of reservation. 2. User can unlock the car clicking “UNLOCK CAR” button. This function works if and only if the user is sufficiently near the car. |
| Output condition | User can enter the car and start the ride.  ????????????????????? |
| Exception | User is not sufficiently near the car. In that situation the system can’t detect user closeness and unlock the car. |

### View car status

A user logged into the system, can research a car throughout his current position or writing in a specific position. After the elaboration of the system, some cars are displayed in a certain area (if they are available). The user can view the status of that car through the mobile application: specifically, he can check the status of the battery or if the car is reserved or not.

|  |  |
| --- | --- |
| Actor | Logged User |
| Goal |  |
| Input Condition | The user must be logged in and he should have researched a car in a certain area (through position or specific address) |
| Event Flow | 1. User inserts his credential on the main page of application 2. Credential are right, so log in is successful 3. User search a car, using his position or a specific address 4. The system display car in the selected zone, if there are some 5. The user select a specific car |
| Output condition | The system provides some useful information about the car, like level of battery and if the car is reserved or not |
| Exception | There aren’t cars in selected area.  The user inserts wrong credentials. |

### View charge during the trip

A user logged into the system, after starts his car-sharing travel, can find on the screen of the car the amount of money he will pay for the trip, instantly. So, the system update the on real time the costs of the service, also calculating the possible discount of the travel for good behaviour.

|  |  |
| --- | --- |
| Actor | Logged User that have rent a car |
| Goal |  |
| Input Condition | The user must be logged. After the search of the car, he should have picked up a car for a travel, so the costs of the service must be different to zero. |
| Event Flow | 1. User inserts his credential on the main page of application 2. Credential are right, so log in is successful 3. User search a car, using his position or a specific address 4. The system display car in the selected zone, if there are some 5. The user select a specific car 6. The user reach the car 7. Car-sharing services start, and the user can begin his travel |
| Output condition | The system provides some information about the cost of the service and about the travel, in real time |
| Exception | There aren’t cars in selected area.  The user inserts wrong credentials. |

### Enable “Money saving” option

The system provides the possibility of some discount if the user has some attention for the environment. For example, it provides a discount if at the end of the rent, the user plugging in the car. This discount tempts the user, and make more probably that the next user will find a full-battery car nearby. If a lot of user do this, the car-sharing environment work well. So, when a logged user start to research a car, he can enable the “money saving” option that enable some functionality: the user can input his final destination and the system provides information about the station where to leave the car to get a discount.

|  |  |
| --- | --- |
| Actor | Logged User |
| Goal |  |
| Input Condition | The user must be logged in and he should have researched a car in a certain area (through position or specific address) |
| Event Flow | 1. User inserts his credential on the main page of application 2. Credential are right, so log in is successful 3. User search a car, using his position or a specific address 4. User can enable “money saving” option 5. The system asks for a final destination |
| Output condition | The system provides information about the station where to leave the car to get a discount. |
| Exception | The user inserts wrong credentials.  The final destination is far away from station where to leave the car. |

### Plug-in the car in order to get the discount

At the end of the travel, the user has some possibility to conclude his rent: he can reach a safe area, where it’s possible to plug-in the car and get a discount on the service thanks to good environmental behaviour, or alternatively he can interrupt the services without any precaution. The system give some advice to the driver, in order to increase the possibility that the cars are parked in a safe area and plugged in. This method can help the environment of the service because if a lot of car are plugged in, there are more possibility that some users will find an available car when it is requested.

|  |  |
| --- | --- |
| Actor | Logged User that have rent a car |
| Goal |  |
| Input Condition | The user must be logged. After the search of the car, he should have picked up a car for a travel, and after some time it should stop the rent. |
| Event Flow | 1. User inserts his credential on the main page of application 2. Credential are right, so log in is successful 3. User search a car, using his position or a specific address 4. The system display car in the selected zone, if there are some 5. The user select a specific car 6. The user reach the car 7. Car-sharing services start, and the user can begin his travel 8. User want to stop the rent |
| Output condition | The system provides some useful information about the safe areas thereabouts, and the system provide the possibility for the user to stop the service and recognise if the car is plugged in. After some steps, the user will obtain a discount for his good behaviour. |
| Exception | There aren’t cars in selected area.  The user inserts wrong credentials.  There isn’t safe area nearby the user’s destination. |

### Visualize “trip review”

At the end of the trip, when the user select the stop button in order to stop the travel, on the screen of the car and of the smartphone appear the “trip review” associated to the last ride. This provides information related to length of the trip in terms of minutes, the different discounts that are applied on the cost of the travel and the overall cost.

|  |  |
| --- | --- |
| Actor | Logged User that have rent a car |
| Goal |  |
| Input Condition | The user must have rent a car and started the trip. |
| Event Flow | 1. During the trip with the car, user can press the stop button in order to stop the trip; 2. The system at this point shows the “trip review” associated to the last ride with the car. |
| Output condition | The system provides information related to the trip, so the user get information related to the last ride and the discount that he had achieved. |
| Exception | Exception occurs if the user press the button in areas that are not safe, in this case he cannot stop the ride because he must park in safe areas |

### Conclude the rent and pay

The user, after reviewing the trip information, if there is possibility can plug the car in order to get another discount. Although this possibility the user after the trip review must conclude the rent with the related button and exit the car so he can finalize the payment and the rent. At this point the car become available again for the other users of the system.

|  |  |
| --- | --- |
| Actor | Logged User that have stop the trip |
| Goal |  |
| Input Condition | The user must have pressed the stop button. |
| Event Flow | 1. The user conclude the rent with the related button in order to pay for the trip and exit the car. |
| Output condition | The bill associated to the trip is paid by the user, and the car associated to that user becomes free again. So it can be rented by other users of the system. |
| Exception | If there weren’t exception after the stop button then there cannot be exception at this point, except for problem related to the payment from the credit card of the user. |