POLITECNICO
MILANO 1863

Requirement Analysis and Specification Document

PowerEnJoy

Tommaso Castagna, Lorenzo Casalino

November 13, 2016

Contents

1 Introduction

1.1 Purpose

The commissioner, namely the car sharing society, is a new company that is trying to deploy its own service to the citizen of a certain city.
Previous analysis marked that the best way for the commissioner to support their service, among other solutions, is by means of as tailored digital system.
Thus, the main purpose of the digital system is the support of the car sharing service through the implementation of classical car sharing functionalities and implementation of functionalities that encourage service's users towards the respect for other users and towards the environment.

1.2 Scope

The requested Digital Management System, abbreviated to System for sake of simplicity, is a software system developed with the aim to support the Car Sharing Service provided by the project's commissioner, briefly named Society.

The main funcionalities supported and provided by the System are those related to the access to the Service, such as registration and log in, and the use of the Service.

Users interested in the use of the service will have the possibility to go through a sign up process, where they will insert their credential informations, such as name, surname, driving license ID etc..., and payment information.
During this procedure the System generates and release to the user a unique random-generated code needed in order to use a reserved car.
After registration, the sign in process allow the registered user to access to the whole service's functionalities.

A registered and logged user has the possibility to search for an available vehicles around an area according to a position decided by the user itself, that can be a specific address inside the city whithin the service operates or the approximated position of the user.
The user can decide to reserve one of the vehicles around the position (s)he has specified for a certain amount of time within which he can reach it and use it.
The use of vehicle is preceded by an authentication step in which

the user has to prove to be near the vehicle and prove to be the user that reserved it.

The Service provides a series of parking areas distribuited all over the city's area, divided into two sets: safe parking areas and special parking areas.
Both of them are individuated by coordinates specifing their boundaries.
The System stores these informations in a stable memory.

Once in use, the System charges the user proportionally to the ride's time, showing the current ride's cost onto a digital display installed into the vehicle.
The charges can be modified according to some particular events, like overcome of city's boundaries.

Users driving a vehicle can terminate the reservation only if (s)he parks into one of the parking areas in the city.
The total ride's cost is calculate at reservation's end, taking in account the ride's cost calculated during the usage and other special conditions that are main part of the Service's usage politic.

The access to the Service, and relative functionalities, is permitted by means of a web application, providing a unique interface for every kind of device the user will use.

Further functionalities can be implemented or the actutal modified/eliminated in the future, hence the system will be designed to accomodate possibile future decisions.

1.3 Stakeholders

1.3.1 Service's customers:

"Every person that decides to sign up to the car-sharing service and and benefits of the advantages provided by the use of the car sharing service through the Digital Management System."

It is clearly a stakeholder because:

- Affects the project: his judgment on the system.
- Is affected by the outcome of the project: the experience that the system provides him.

- Is affected by the decision of the project: the digital management of the service and relative use of the service.

1.3.2 Car-sharing society:

"The society committing, supervisioning and paying the project."

1.3.3 Automotive society:

"The automotive society providing the cars fleet to the car-sharing society."

The Digital Management System is the core of the car-sharing service and is the system with which the cars will interact. Therefore, without the definition and design of the system, the fleet cannot be still provided.

The automotive society will provide the cars equiped with all the needed instrumentations to interact and integrate with the system.

The automotive society is affected by the projects because it should modify the current production branch for those specific cars in order to produce cars with the required hardwares.

1.4 Definitions and acronyms

1.4.1 Definitions

1. System: the system we are designing.
2. Services: set of all the funciontalities made available by the system.
3. User: a phisical person that is interacting with the system.
4. Registred user: a user that has completed the registration process.
5. Logged user: a user that is currently logged in the system
6. User ID: unique code generated during the registration process that is communicated to the user and used to check his identity when trying to unlock a reserved car.
7. Vehicle ID: unique identification number associated to every car owned by the society.

8. Driver: a user that is driving the car that he has reserved.
9. Passenger: any person who is in the car different by the driver.
10. Society: the society that is running the car sharing business and has commisioned us the design of the system.
11. Company: synonym for society.
12. Contract: it is the legal document that any user has to subscribe in order to register to the service that rules all the aspects of the service itself.
13. Central system: the set of all the hardware and software components installed in the car that monitors all the aspects of the car and allow the communication between the car and the system.
14. Stop option: is the possibiliy that the user have to leave the car without ending his reservation so that he can use it again later, paying the stop fee
15. Stop fee: is the fee that the user pays for activating the stop option.
16. Authentication device: is a device installed on each of the company's cars that allow a user to identify himself in order to unlock the car.
17. Unlocking timer: is the timer that starts when the user notifies the system that he wants to unlock his reserved car, at this point the system enables the authentication device on the corresponding car, and the user has to authenticate himself before the timer ends, otherwise the authentication device will be disabled.
18. Reservation time: is the maximum amount of time, fixed by the society, that a user has to pick up the car after his reservation has been accepted.
19. Forwarded reservation request: a request for the reservation of a car that has been sent by the user to the system.
20. Available car: a car is considered available if it is not reserved by a user and doesen't need maintenance.

21. Not available car: a car is considered not available if it is reserved by a user or needs maintenance.
22. Payment system: the payment method provided by the user during the registration.
23. Special conditions: particular situations, decided by the company, in which the user is eligible of a discount or an increase on the ride price.
24. Safe parking areas: areas of the city where users are allowed to park the rent car.
25. Special parking areas: special parking areas are safe parking areas provided with electical plugs that allow car charging.

1.4.2 Acronyms

1. RASD: Requirements Analysis and Specification Document.
2. DB: DataBase.
3. DBMS: DataBase Management System.
4. API: Application Programming Interface.
5. GPS: Global Positionig System.

1.5 References

1. "Requirement Engineering Part III", page 49-50, Di Nitto, Mottola.
2. "IEEE Standard For Requirement Engineering, ISO/IEC/IEEE 29148", par. 9.5
3. "Requirements Analysis Document", Florida State University, http://www.cs.fsu.edu/~lacher/courses/COP3331/rad.html
4. "Defining the Scope in IT Projects", ProjectPerfect http://www.projectperfect.com.au/info_define_the_scope.php
5. Assignment Document: Assignment1

1.6 Overview

This document is aimed to providing a high level description of the system we are developing.
In section 2 of this document we're focusing on functionalities and assumptions.
The third section is mainly dedicated to a deeper investigation of our system functionalities through the presentation of the different goal we want to meet and a UML modeling of the system and the main operations a user can perform.
In the last part of the decument is shown a possible Alloy formalization of the system to be.

2 Overall Description

2.1 Product perspective

The system to be is composed of four main components:

1. Main user interface available via web or mobile application
2. The central system insalled on every car
3. The backend infrastructure that manages the whole system

2.1.1 System interfaces

The system should present a set of APIs with relative guidelines in order to allow further development of the application in the future.

2.1.2 User interfaces

The web application should be implemented responsive, in order to scale al the contents in the web page to preserve all the funcionalities and to keep the application usable regardless of the screen size of the device used by the user, wether a desktop or a mobile device.
The mobile application should also be usable on different mobile divices such as phones and tablets keeping a consistent user iterface on different phones and tablets.

2.1.3 Software interfaces

The system must comunicate with an external web server in order to let the customers interact with the system through the web application.
It also must be interfaced with a suitable DBMS to store and retreive all the informations needed to the proper functioning of the

car sharing service.
The system relies on and external mapping service in order to determine the position of each car through the GPS data provided by the central system, and to show the user their position on a map so that he can select the desired vehicle.
The system must also communicate with the city motorization web service in order to check the validity of the driving license informations provided bu the user during the registration.
In order to manage the payments of the users who benefit from the service, the system must support the most used payments method.

2.2 Product functions

In the following is presented a brief summary of the main functionalities provided by the System.
The list includes functions regarding the use of the Service and the management of the Service itself, in the specific:

- User profile creation, management and cancelation.
- Reservation tracking.
- Car research, reservation and unlock.
- Management of car's availability.
- Management of users' behaviour differing from the Society's usage politic.
- Update parking areas informations.
- Update cars' information.
- Update cars' fleet.

2.3 User characteristic

Since the System will interact with a broad set of users, the general audience has a varied knowledge and competence with techonological device and mechanism. However, we assume that the audience has a minimum of ability in the interaction with mobile and desktop devices and web applications.
The technological background of Society's workes is not expected to be broader than the one of the service users, with the exception of administration's tools knowledge.

2.3.1 Actors identifying

1. Non-registered user: A user that does not have an associated profile on the system. The only operation allowed to him is the registration on the system in order to get access to the service's functionalities.
2. Registered user: A user that has an associated profile on the system but has not performed the logging into the service. He can only log in to access to service's features.
3. Non-logged user: A user that can be a registered user or a non-registered user. In both cases, it is not marked as logged user. The only operations allowed are logging in if he's a registered user, registration if he is a non registered user.
4. Logged user: A user that has an associated profile on the system and has successufully perfomed the logging in operation. He has access to all the system's functionalities, such as search for cars and reserve one.
5. Car's Central System: The hardware/software system installed into each car. It interfaces with all instrumentations installed into the car and comunicate directly with the digital system.
6. Motorization's Web Service: The public web service to which the digital system interfaces to check the validity of inserted driver licenses.
7. Client Service Operator: The operator instructed to interact with the clients when particular events, like an accident, occur.
8. Mantainance Operator: The operator instructed to proceed with the mantainance and recover of company's vehicle.

2.4 Constraints

2.4.1 Regulatory policies

Since the user of the System are going to provide their sensitive datas and position, the System is subject to the regulatory policies regarding the treatment of personal and sensitive datas. Further regulatory policies are applied regarding datas recorded by the web application used to access to the Service.

2.4.2 Hardware limitations

In general, we don't expect any particular constraint or limitation due to the hardware thank to the technology evolution. Anyways, even if the mobile field is technologically growing, not all the users are going to use last mobile device available in the market.
As a consequence, is necessary to introduce a minimun set of constraints in order to maximize the usability and efficiency of the System and the application used to interface to the service. We have individuated the following hardware constraints:

- The application must run smoothly on single core CPU, based on X86 architecture.
- The user device must support at least the 3G connectivity.
- The application must use a maximum amount of 64MB RAM and 128MB of second mass storage.
- Screen size of the device must be greater or equal to 480 x 320 px (height x width).

2.4.3 Interfaces to other applications

Payment functionalities, such as transaction execution, transaction protection, encryption etc... are provided by an external payment service to which the System interface.

Mapping functionalities are provided by an external Mapping service.

2.4.4 Parallel operation

Since the System will be deployed into an high concurrent environment, we expect that the System will have to manage several transaction requested at the same time by different end users (humans and machines).

2.5 Domain properties and assumptions

1. Only registred users can log into the system.
2. Each car is equiped with a dedicated GPS tracking systems.
3. Each car is equiped with a permanent internet connection module.

4. Cars' GPS system guarantee the current vehicle's position with a minimal bias.
5. The area's width for the cars research is predefined by the system.
6. A forwarded reservation request is confirmed if and only if the car speficied into the request is available.
7. Each car is equiped with an identification device that allows the user that has made the resevation to notify the system that he's near the car.
8. Each car is equiped with a central system that manages all the installed instrumentation and the communication with the system.
9. Each car is equiped with sensors for notice the presence of passengers on the seats.
10. A person can specify one and only one payment system for registration.
11. The payment process is delegated to an external service.
12. The city motorization has a public web service usable for verify the validity of the driver licenses.
13. The motor vehicle registry contains an updated list of valid driving licenses.
14. A person can have at most one account registered in the system.
15. Non registred user can use the system only for registering themselves to the sistem itself.
16. Registred users can only use the system if they are currently logged into it.
17. Only registred users are allowed to log into the system.
18. The society relies on an external company for cars' maintenance.
19. Each of the company's car is provided with a accident statement report form.

20. Each of the company's car has an accident insurance.
21. It is responsibility of the user to contact the competent authorities in case of accident.
22. It is responsibility of the user to fill the accident form and to send it to the company within the time limit stated in the terms and conditions accepted during the registration.

3 Specific Requirements

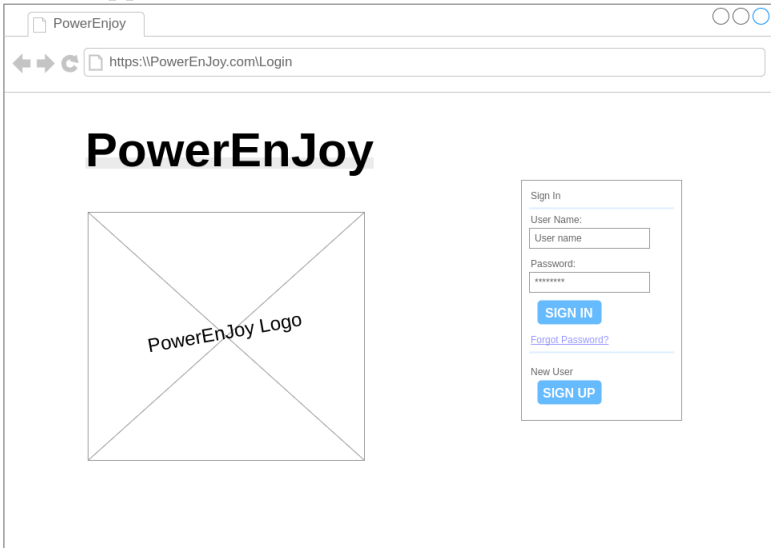
3.1 External Interface Requirements

3.1.1 User Interfaces

Following is a sequence of sketches representing possible implementations of the user interfaces.

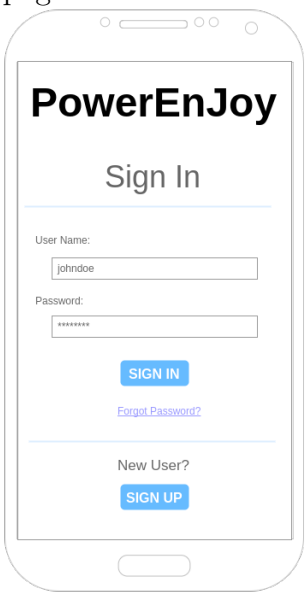
Sketch 1:

The sketch shows a possible implementation of the login page into the web application interface.



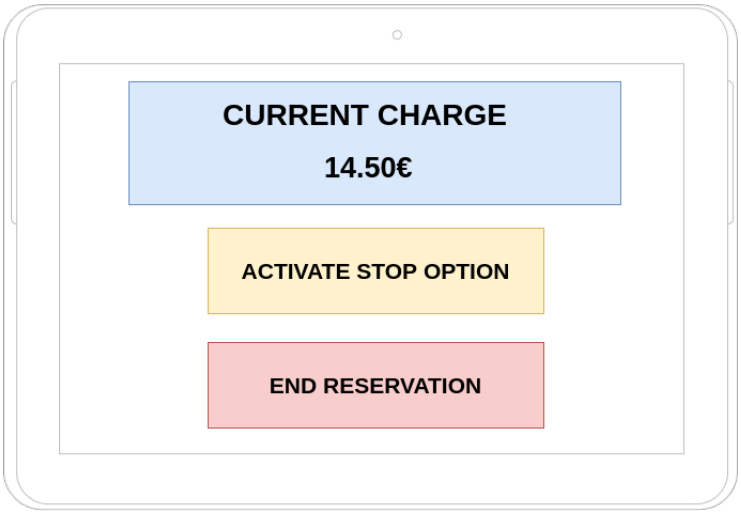
Sketch 2:

The following sketch shows a possible implementation of the login page into the user mobile application



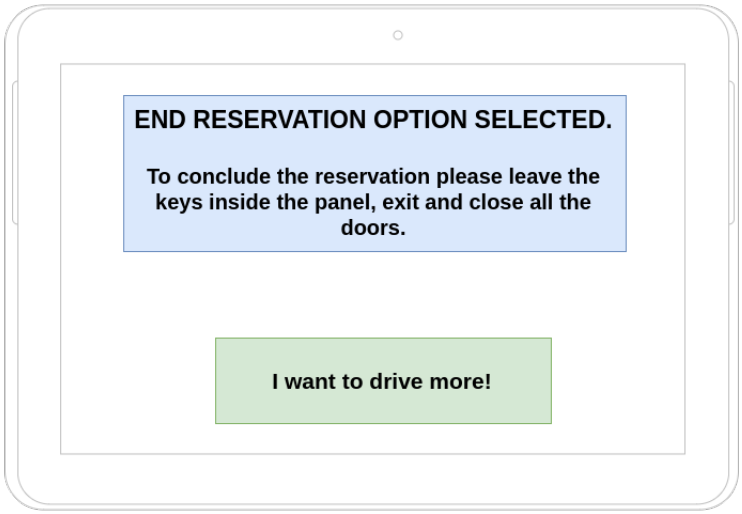
Sketch 3:

The sketch shows the possbile implementation of the main car's display interface.



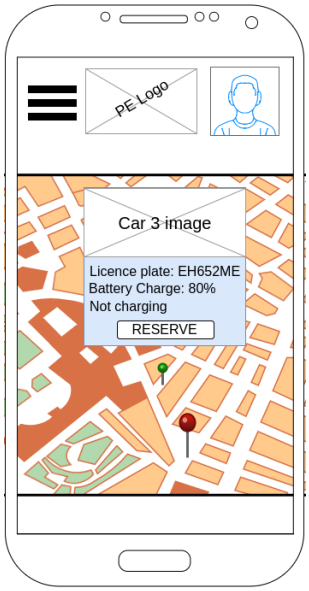
Sketch 4:

The sketch shows the possible implementation of the end reservation screen.



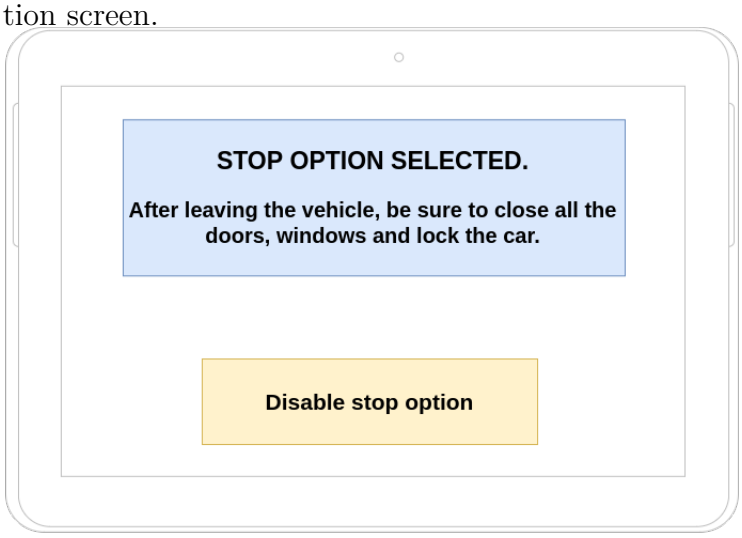
Sketch 5:

The sketch image shows a possible implementation for the car search on a mobile device



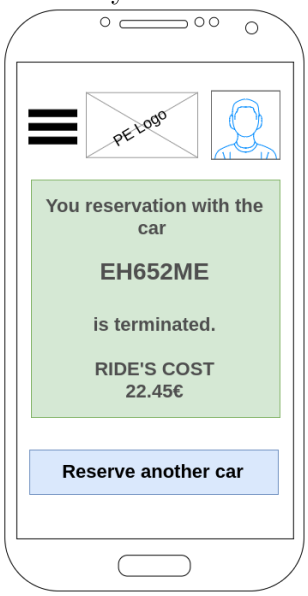
Sketch 6:

The sketch shows the possible implementation of the stop reservation screen.



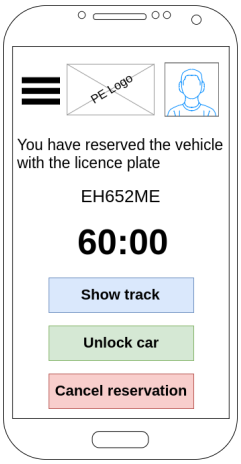
Sketch 7:

The sketch shows a possible implementation of the end reservation summary for the mobile devices.



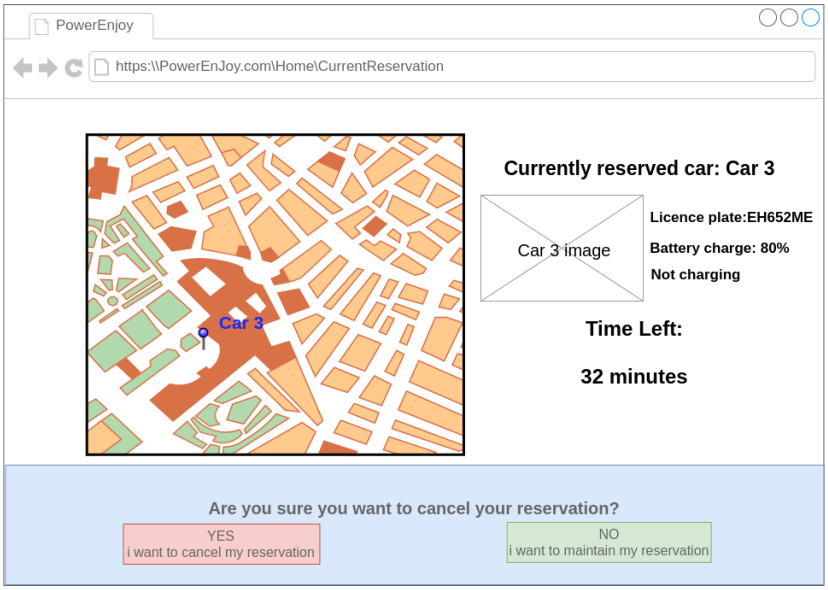
Sketch 8:

The sketch shows a possible implementation of the current reservation's summary for mobile devices.



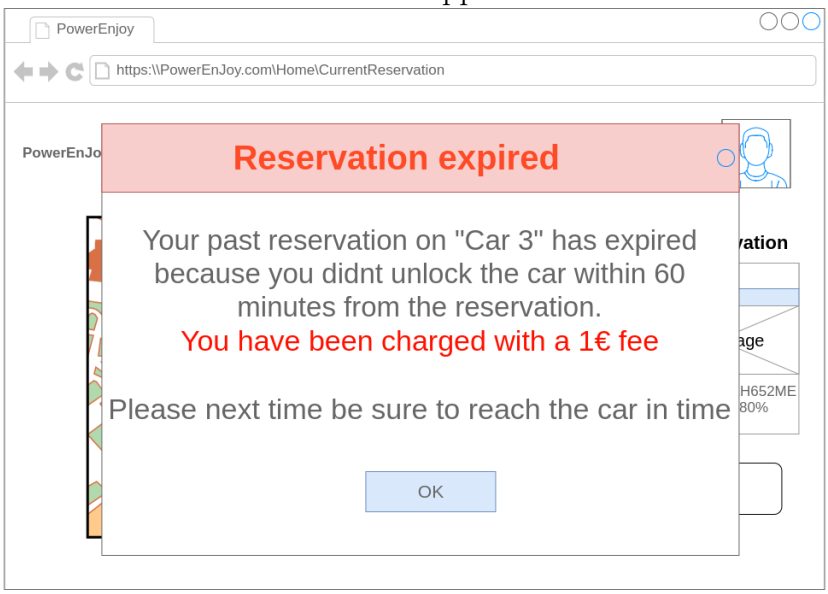
Sketch 9:

The sketch shows the possible implementation of the "cancel reservation" function on the web application.



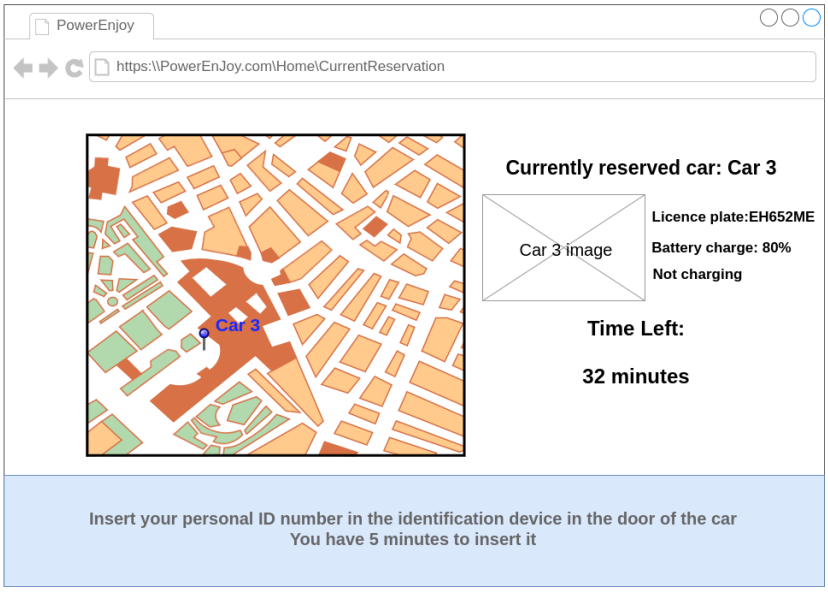
Sketch 10:

The sketch shows the possible implementation of the expired reservation notification for the web application.



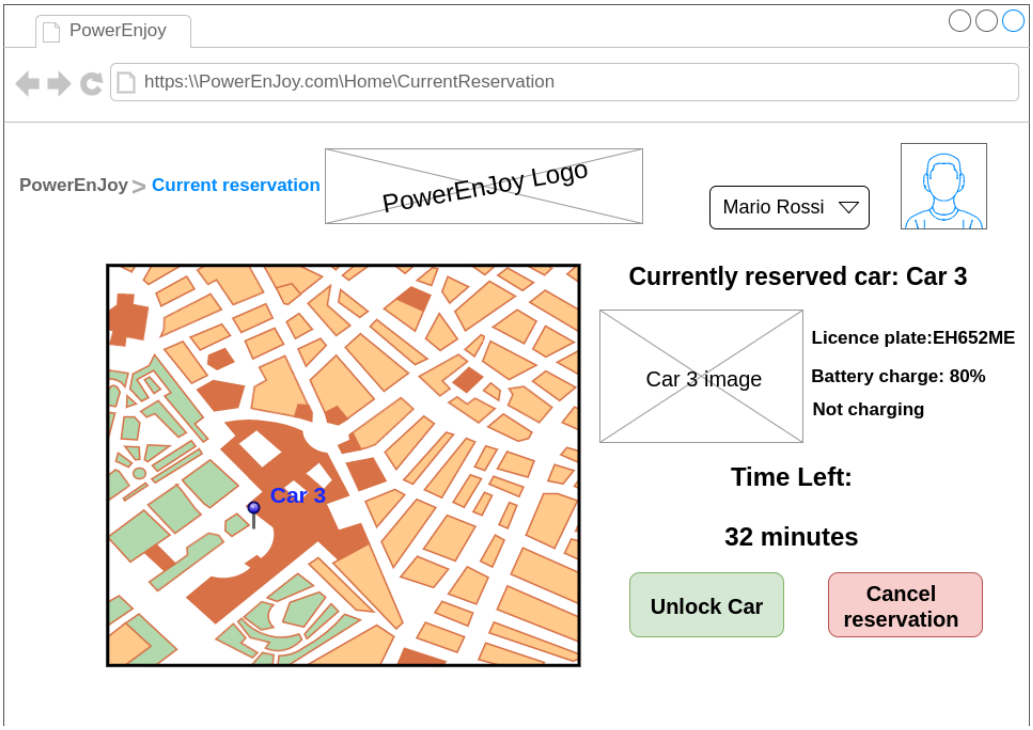
Sketch 11:

The sketch shows the possible implementation of the "unlock" functionality in the web application.



Sketch 12:

The sketch shows a possible implementation of the current reservation summary for the web application.



3.2 Funcitonal Requirements

3.2.1 Goal 1

Only logged users should access to the service provided by the society.

Requirements:

1. The system should implement a mechanism to separate the operations priviledges on the platform according to the user category.
2. Whenever an user is trying to access to the services, namely is trying to log in, (s)he has to pass through a verification step where the System verifies if (s)he has a registered profile associated.

3. The system should implement a multifactor mechanism to verify if an user that passed the account existence step is the real owner of the profile.

3.2.2 Goal 2

A logged user may look for available cars and reserve one.

Requirements:

1. For each logged user searching a car, the system must comunicate them what cars are available, along with their position, battery status, conditions and estimated distance from user position.
2. The system should retrieve coordinates and status informations of each car.
3. The system must keep track of each available car for possible reservations.
4. When a logged user decides to look for a car, (s)he should have the possibility to choose around which geographical point inside the city search.
5. If no car is available in the area around the searching point, the system should notify the user.
6. A logged user may comunicate what car has decided to reserve.
7. For every forwarded reservation request the system verify if the chosen vehicle is still available.
8. If the car specified into the reservation request is not available anymore, the system notifies the user.
9. The system keeps track of the details of every confirmed reservations.
10. The user may be able to see what are the safe parking areas and special parking areas.

3.2.3 Goal 3

A user may unlock and use a car for which (s)he forwarded a reservation request.

Requirements:

1. The system must provide the user a mechanism to enable the unlock procedure on the car that the user has reserved.
2. The system must verify through the identification device on the car the identity of the user, and allow only the correct user to unlock the car.

3.2.4 Goal 4

The system should guarantee a correct management of the car's availability.

Requirements:

1. Whenever a reservation request is accepted, the system tags the reserved car as "not available".
2. Whenever the central system notifies the existence of a problem with one of the car's component, the system should mark the corresponding car as "not available".
3. If the user doesn't pick up the car within the reservation time his reservation is automatically ended by the system.
4. Whenever a reservation is ended by the system or by a user, the corresponding car is marked as "available" if it doesn't need mainenance, otherwise it's marked as "not available".

3.2.5 Goal 5

The system should calculate and deduct the total ride cost at ride's end from the payment system indicated by the user.

Requirements:

1. The system must allow the user to explicitly end his reservation.
2. The system must allow the user to leave the car without ending his reservation, activating the stop option, applying a specific fare.

3. If the user leaves the car without ending the reservation or without activating the stop option, the system waits a fixed amount of time before it locks the car and ends the reservation.
4. Whenever a reservation ends, the system updates the reservation details.
5. The system calculates the total ride cost according to the ride details and possible special conditions.
6. The system has to verify if the payment system has enough credit on.
   If not the system deduct the credit on the payment system and not allow to the user to reserve other vehicles until the user specifies a payment system with enough credit to cover the debit.
   Else the total sum is directly deducted.

3.2.6 Goal 6

The society may have the possibility to change the set of parking areas, both safe and special.

Requirements:

1. The system should store the coordinates and boundaries of each parking area, in case of special parking areas, also the number of plugs is stored.
2. The system must provide an interface through which society's responsibles may edit all the parking areas' datas, cancel and add areas.

3.2.7 Goal 7

The user may have the possibilty to create a profile, manage and delete it from the system.

Requirements:

1. The system stores primary credentials, payment system and driver license ID.
2. The system should verify the validity of the payment system.
   If the payment system is rejected, the registration procedure

is blocked until a new valid payment system is specified or the user abort the sign up process.

3. The system should verify the validity of the driver license ID. If the driver license ID is rejected, the registration procedure is blocked until a new valid driver license ID is specified or the user abort the sign up process.
4. The system generates and stores a unique user ID for every registered user after a successful registration.
5. The system should permit the association of a driving licence to at most one user at a time.
6. The system should provide to registered user an interface through which manage his profile.
7. The system should remove all datas related to an user that successfuly deleted his account.

3.2.8 Goal 8

A users may report problems regarding car's conditions and accidents during the use of cars.

Requirements:

1. The system must provide a short guided procedure to report status regarding the car's conditions.
2. The system records all conditions reports.
3. After the submission of the accident report form, the company will apply penalties to the user as stated in the contract and will take care of the car.

3.2.9 Goal 9

A registered user may have access to informations about his reservation history.

Requirements:

1. Whenever a car's status changes, the system records it along with the triggering event.
2. The system links status's changes to the registered user that triggered the change.

3.2.10 Goal 10

The system must be able to handle the case in which the user behavior differs from the service policies.

Subgoals:

1. The system must take action in case the user drives a rent car outside the the geographical boundaries of the service.

Requirements:

1. As a user overcomes the service's boundaries, the system notifies the user inviting to re enter the allowed usage area as soon as possible.
2. The system applies a extra charge to normal one for every minute spent outside the boundaries.
3. If the user does not enter into the boundaries within a fixed time, the system warns the law enforcement, giving detailed informations about the car and the user.

2. The system must take action in case the user doesn't close the car's door and leaves the car.

Requirements:

1. The system must prevent the user from closing his reservation if the doors are not correctly closed and notify him of the problem.
2. If a car is left open the system will notify an employee of the problem and the location of the car.

3. The system must take action in case the user doesn't park in a safe or special parking area and leaves the car.

Requirements:

1. If a user shuts off the engine outside of a safe or special parking area the system notifies him that he's going to be fined in accordance with the company policies if he ends his reservation.
2. If a car is left in an unauthorized area the system will notify an employee the problem and the location of the car.

3.3 Performance Requirements

As no information about the city, citizen habit regarding the car usage and presence of other car sharing services in this section the performance requirements expressed are made under some assumptios and raw analysis:

1. The city is supposed to be a medium metropolitan city like Rome or New York, hence we can expect that the system will works under a huge load of datas and transactions. For this analysis we expect to deploy the system into a city with a number of 5 milion of users.
2. To evaluate the worst scenario possible, we assume that in the city there's no other car sharing competitor, maximizing the expected number of users.
3. No information about the the citizen habits, number of private vehicles and is available. Thus, we assume that only half of the population has a car.
4. We assume that only one fifth of the population without a car subscribes to the service.
5. We assume that the 70% of the subscribers are monthly active.

Hence, in the light of the previous assumptions, we can state the following raw performance requirements:

- Support the registration of at least 500'000 registered users in total.
- Support at least a minimum number of 500'000 parallel transactions.
- Response time for search and booking transaction should be less than 1s.
- General transactions can have a maximum time of response of 1.5 seconds.

Again, we remark that these are raw requirements made on assumptions. Deeper analysis of the context should be performed.

3.4 Design Constraints

We don't have any specific design constraint also due to the fact that we are building our system from scratch.

3.5 Software System Attributes

3.5.1 Reliability:

The system is supposed to be used constantly by a high amount of users, possibly growing. Therefore, the system should be developed in order to give a high level of reliability, with main components failing no more than once a year.

3.5.2 Availability:

As the system is supposed to be used constantly, the architecture of the system should be designed to guarantee an availability of 99.99%, that is a downtime period of about 50 minute per year. Use of replicated components and fast recovery mechanisms are suggested.

3.5.3 Security:

Since the system is supposed to interact via internet with the users, several precautions should be taken in account to avoid denial of services attacks, disclosure of sensitive informations, system breaching, sniffing and hijacking attacks.

Regarding denial of services attacks and system breaching attempts, the system should be hosted by machines running reliable operating system, correctly updated and mantained.
Furthermore, stateful firewall or web application firewalls and correct firewall rules should be applied to protect the system from malicious requests and attempts to disrupt the service through flooding.

Correct database priviledges separation can prevent system breaching (such as SQL injections) and disclosure of sensitive informations.
Moreover, mechanism to avoid bruteforce attack should be implemented, such as temporaneuos block after too many attempts of access to a certain profile.
Multiple-factor authentication mechanism should be implemented as well to avoid access to the system by unauthorized users.
Credentials used to access to the system should be encrypted through modern mechanisms (such as hashing and salting) to avoid their disclosure in case of datas leaking.

Sniffing and hijacks can be almost prevented through the use of secure communication protocol, such as TSL, implementing communication encryption and parties identification and authentication and use of reliable operating systems.

The car's central system should be protected as well from attacks that can allow a malicious user to access the system or can provoke damages through it.

Backup copies of the databases should be performed regularly and stored on secure mass storages.

The servers, backup copies and other devices and system supporting the service should be hosted in a facility secure against natural and artificial threaths, like fire outbreak, earthquake, blackout, flooding and so on.

3.5.4 Portability

To maximize the portability of both the digital system and the applications available for the users, both them should be developed through the use of languages and frameworks platform independent by the underlying operating system where they will be hosted and executed.
Moreover, the interface between the car's central system and the digital system should be designed in order to be more abastract as possible, making possible the future modification of the central system or of the system itself, avoiding any impact of the interaction.

3.6 Scenarios

3.6.1 Accidents happen

It's a quite evening in the car sharing's call center.
Laura is enjoying her coffee when she receives a call.
On the other side of the cable is one service's user that has been involved in an accident, reporting that the booked car is not more usable.

As usual in these situations, Laura asks for the details of the accident, the exact location of the accident, the usercredentials and the car's identifier.
After getting it, she sends a create report accident request through her terminal, indicating all the informations received by the user.

The system receives the request and creates the report, recording the accident's details and creating an association between the report and the reservation.
An acknowledge is send back to Laura.

Laura receives the notification from the system.
Since the vehicle is not more usable, Laura has to manually end the reservation, so she selects the "end reservation" option, indicating the car identifier provided by the user.

As received the request, the system provides to end the reservation, updating the related details.

Laura receives a success notificatino from the system.
Now she can warn the user to wait the arrive of an operator instructed to recover the damaged vehicle.

3.6.2 Better To Unlock It

Andrew is moving towards the car he has reserved through the smartphone.
He sees the vehicle at no more than 50 meters away from his position, so he decides to enable the unlock procedure tapping the specific option over the application's interface displayed on his smartphone.
The application sends the enabling request to the system.

The system receives the requests and comunicates to the central system of the booked car to activate the authentication device installed on the outside of the car.
Once near the driver side, Andrew interacts with the authentication device, completing the authentication step and waits the unlock of the vehicle.
The car's central system sends the input to the system.

The system receives the authentication informations provided by Andrew and verifies if the informations authenticate Andrew himself.
After the success of the verification, the system sends an unlock signal to the car's central system and wait for an acknowledge.

The central system receiving the signal unlocks the doors and sends back to the system an acknowledge to notify the correct success of the operation.

As receiving the acknowledge, the system deletes the timer associated to the reservation and notifies Andrew of the operation'success through a notification sent to the application.
The application shows the notification to the Registered User.
Andrew may now enter and drive without any physical effort to home.

3.6.3 Icarus

Today is a really beautiful day and Charlie organised with her friends a picnic near Belle's Park.
All his friends live in the same district, in the outskirts of the city, so he decides to book a car and go to pick up them.

During the travel towards the district, Charlie mistakes and does not turn in the right street, thus entering into the highway.
He reaches the tollbooth and as he passes it, the central system of the car notifies the system of boundaries overcome.

The system sets a timer and comunicates to the user the application of a higher fare due to his error.
Moreover, the system notifies the user that if the car won't be back before the timer reaches the 0, the law enforcement will be called.

Charlie starts to sweat but does not lose control and keeps driving.
After a while he manage to reach the nearest tollbooth, enter in the city and re-enter in the highway, with direction the city of competence of the society

After entering into the city boundaries, the central system of the car notifies the system.
The system verifies that the timer is not over and records into the details of the event.

Charlie won't forget the next time to pay attention.

3.6.4 Log, search and reserve

After an intensive evening workout in a lost park located in the outskirts of the city, Andrew realizes that the public trasport has ended the service.
Andrew lives near the city center and would be a hell to walk for an hour or more after the training.
So he picks up from the pocket his smartphone and logs into the car sharing application.

To book a car, taps on the "Reserve" option displayed on the application's interface.
The application diplays to Andrew a form where to insert the position where start the research.
Since he is tired, he decides to look for car near his position tapping on the corresponding option on the interface.

A search request, containing the chosen position by Andrew, is forwarded to the System and Andrew waits the response containing car's locations and informations.

The system, once received the request, gathers informations about the position of each available car through their GPS system and selects only the ones near Andrew's position.
The System sends back to the application the position of the nearby cars along with informations regarding distance and battery's charge.

The application shows on Andrew's smartphone the nearby cars plotting them onto the application's interface.

After evaluating all the possible choices, Andrew decides to reserve a car near St.Petersburg street.
Hence, he taps over the "Reserve this vehicle" option, sending a "Reservation" request pointing out the selected car.

The System verifies that the selected car has not been reserved meanwhile Andrew was deciding and marks the chosen car as not available.
The system set up a timer and associates it to the reservation.
A success notification is sent back to Andrew, who now can go towards the reserved vehicle.

3.6.5 I Want To Delete Free

After have reserved a car throught the service, Mattia exits quickly from the building hosting his office.
While he's on the way to the vehicle, the CEO of his company calls him asking to present for tomorrow morning an important review about the RASD of the current project.
Sadly, Mattia has to say yes.

After closing the call, Mattia accesses to the service throught his portable device.
The device's display the service's interface, which shows the current reservation timer's value and the position of the vehicle plus some

options.
Mattia has to go back into the office to work all the night, so taps the "cancel reservation" options displayed, forwarding a "cancel reservation" request to the System.
The request points out the reservation identifier.

The System receives the request and marks the vehicle associated to the reservation as available again.
The reservation timer is deleted and every detail of the event is recorded.
The System returns back to Mattia a success notification.

Mattia slowly returns back to his office for a long night of work.

3.6.6 Time to park

Barbara is driving with a reserved car towards the happy hour meeting place to spent a good evening with her friends.
Once reached the nearest parking area and after parked the car, Barbara taps on the display the option to end the ride.
Barbara turn off the engine, leaves the car's keys into the car's panel and exits from the car.
Once out of the car and closed the door, the sensors inside the vehicle notice the absence of passengers and the presence of the keys inside the panel.
The car's central system notifies the system.

As the system gets the notification, it sends a lock signal to the car and waits for an acknowledge.
The central system receives the signal and lock the car's doors, notifying the system of the success of the operation.

The system receives the acknowledge and stores the details of the reservation, calculating the ride's price, and tags the car as available for further reservations.

While walking towards the club where her friend are waiting her, Barbara receives on the smartphone the details of the ride, along with the confirmation of the reservation's expiration.

3.6.7 Take a break

Alice just entered and started the car she has reserved when she notice to have forget the gift for her friend Eve.
The system has already started to charge Alice and giving up the

reservation means to loose time to look for another available car, and she has no guarantee that another vehicle is available nearby her position.

Hence, Alice decides to select the stop option from the car's integrated display, in order to keep her reservation.
The system is notified of Alice's choice and applies a different charge to the reservation, registering the event into the reservation's details.

Alice exits the car and locks it with the locking system activable through the car's keys, being sure that no one would steal it while she is away from the vehicle.
She runs towards his appartment to recover the forgotten gift.

When she is back, Alice unlocks the car using the keys and enter inside the car.
She disables the stop option and start to drive.

The system receives the notification and applies the usual fare on the reservation and records the event's details.

Appendix A Use Cases

Use case 1

Use case name:
DeleteReservation

Partecipating Actors:
Registered User

Entry Condition:
The Registered User selects the "delete reservation" option through his device.

Flow of Events:

1. The Registered User forward a "delete reservation" request to the System.
2. The System receives the request, tagging the reserved vehicle as "available".
3. The System deletes the timer associated to the reservation.

4. The reservation's details are recorded.
5. The System sends a notification success to the Registered User.

Exit Condition:
The Registered User is notified by the System of the success of the operation.

Exceptions:

- The communication between the Registered User and the System goes down. The Registered User is immediately notifiedof the event by the service's application interface. The use case ends.
- The Registered User aborts the use case. The use case ends.

Use case 2

Use case name:
EndReservation

Partecipating Actors:
Registered User
Car's Central System

Entry Conditition:
The Registered User selects the end reservation option through the car's display.

Flow of Events:

1. The Car's Central System sends to the System the request along with the geographical coordinates of the used car and the car's identifier.
2. The System verifies if the geographical coordinates are inside into one of the safe areas distributed over the city. After positive result, the System notifies the Registered User the possibility to leave the car and waits for the acknowledge by the Car's Central System.
3. The Registered User turn off the engine and lights, leaving the car's keys inside the car's panel. The Registered User exits from the vehicle.

4. The Car's Central System notifies the System of the absence of passengers inside the car.
5. The System receives the notification and verifies if both the doors are closed the car's keys are inserted into the panel.
6. The Car's Central System performs the control and notifies the System.
7. The System receives the notification and deducts the ride's cost opening the ExecutePayment use case. The System notifies the Registered User of the success.

Exit Conditions:
The Registered User receives the success notification by the System.

Exceptions:

- The communication between the Car's Central System and the System goes down. The Registered User is immediately notifed. The procedure is aborted. The use case ends.
- The Registered User aborts the process. The use case ends.
- The Registered User exits without leaving the keys inside the panel. The use case is aborted.
- The Registered User does not close the car's door when exits. The Registered User is immediately notified. The use case ends.
- The System cannot detect the position of the vehicle. The Registered User is notified. The use case terminates.

Use case 3

Use case name:
MakeReservation

Participating Actors:
Registered user
Car's central system

Entry Condition:
The Registered user select the "Reserve" option on the device's interface.

Flow of Events:

1. The Registered User specifies the geographical position. The Registered User sends the "Search" request to the System.
2. The System retrieves informations and positions of the availables' car near the position specified by the Registered User. The System sends these informations to the Registered User.
3. The Registered gets the informations and position plotted on his/her device's display. The Registered User sends a "Reserve" request along with the car's identifier.
4. The System creates a Reservation, recording the details of the reservation. The System creates a Reservation Timer and associates it to the reservation. The System notifies the Registered User of the operation's success.

Exit Condition:
The Registered user is notified of the success of the reservation.

Exception:

- If no vehicle is available at all, the System notifies the Registered User sending an Error Notification to the Registered User.
- If no vehicle is available in the area around the position specified by the Registered User, the System notifies the Registered User sending him an Error Notification. The use case ends.
- The Registered User is notified immediately if the connection between his/her device and the System goes down. The use case ends.
- The car selected by the Registered User gets reserved by somebody else. The System notifies the Registered User sending him/her an Error Notification The use case is terminated.
- The Registered User aborts the procedure. The use case ends.

Use case 4

Use case name:
OverwatchBoundaries

Partecipating Actors:
Registered User
Car's Central System

Entry Condition:
The Registered user crosses the city's boundaries with the reserved vehicle.

Flow of Events:

1. The Car's Central System detects the overcome of city's limits and notifies the System of the event.
2. The System applies a new fare to the reservation and record the event into the reservation's details. The System creates a timer within the vehicle has to re-enter inside the city's boundaries. The timer is associated to the reservation. The System notifies the vehicle's driver of the event sending a warning message to the car and displaying the remaining time.
3. The Registered User re-enter inside the city's limits.
4. The Car's Central System notifies the System.
5. The System apply the usual fare to the reservation, recording the event into the the reservation's details. The System deletes the timer.

Exit Condition:
The System notifies the Registered User sending him a notification.

Exception:

- The timer ends before the vehicle re-enter into the city's limits. The CallEmergency use case is open. The use case ends.
- The comunication between the Car's Central System and the System goes down. The Car's Central System notifies the Register User. The use case terminates.

Use case 5

Use case name:
PerformeStop

Partecipating Actors:
Registered User
Car's Central System

Entry Condition:
The Registered User selects the stop option from car's display.

Flow of Events:

1. The Car's Central System sends the request to the System.
2. The System recevices the request and changes the current fare applied to the reservation.
3. The event is recorded into the reservation's details.
4. The Registered User disables the stop option through the car or through his device.
5. The System edit the current fare to the usual one and notify the Registered User.

Exit Conditions:
The Registered User receives the notification.

Exceptions:

- The System cannot detect the position of the car. The Registered User is immediately notified. The use case ends.
- The connection between the System and the Car's Central System goes down. The Car's Central System notifies immediately the Registered User of the event through the car's display. The use case terminates.

Alternative flow:
4a. The Registered User drives the car while the option is active.

Use case 6

Use case name:
UnlockCar

Partecipating Actors:
Registered user
Car's Central System

Entry Condition:
The Registered User request the authentication device's activation.

Flow of Events:

1. The System receives the request and comunicates to the Car's Central System to activate the Authentication Device.
2. The Car's Central System activates the authentication device and set up a timer.
3. The Registered User interacts with the authentication device.
4. The car's central system sends to the system the input inserted by the Registered User.
5. The System verifies the validity of the authentication input and sends to the the Car's Central System a notification.
6. The Car's Central System unlocks the car's door and activate all the components. The authentication device is turned off and the timer associated is deleted. The Car's Central System notifies the system of the operation's success.
7. The System gets the acknowledge from the Car's Central System and notifies the Registered user.

Exit Condition:
The Registered User receives the notification of the success of the operation.

Exceptions:

- If the comunication between the Registered User and the System goes down, the Registered user is immediately notified. The use case terminates.

- If the comunication between the System and Car's Central System goes down, the Registered User is immediately notified. The use case ends.
- If the verification of the authentication input returns a negative result, the Registered User is notified. The use case ends.
- If the timer associated to the authentication device reachs the zero, the Registered User gets notified. The use case ends.

Use case 7

Use case name:
ReportAccident

Partecipating Actors:
Operator

Entry Condition:
The Operator creates a "report accident" request through his terminal

Flow of Events:

1. The Operator specifies the accident's details, car's identifier and accident's location inside the request. The Operator sends the request.
2. The System receives the request and create the Report, inserting all the informations received by the request. The System associates the Report to the related Reservation. The System notifies the Operator.
3. The Operator receives the notification and terminates the reservation selecting the "end reservation" option, specifing the car's identifier. The Operator sends the request.
4. The System receives the request and updates the reservation status to "expired". The System notifies the Operator.

Exit Condition:
The Operators receives the notification.

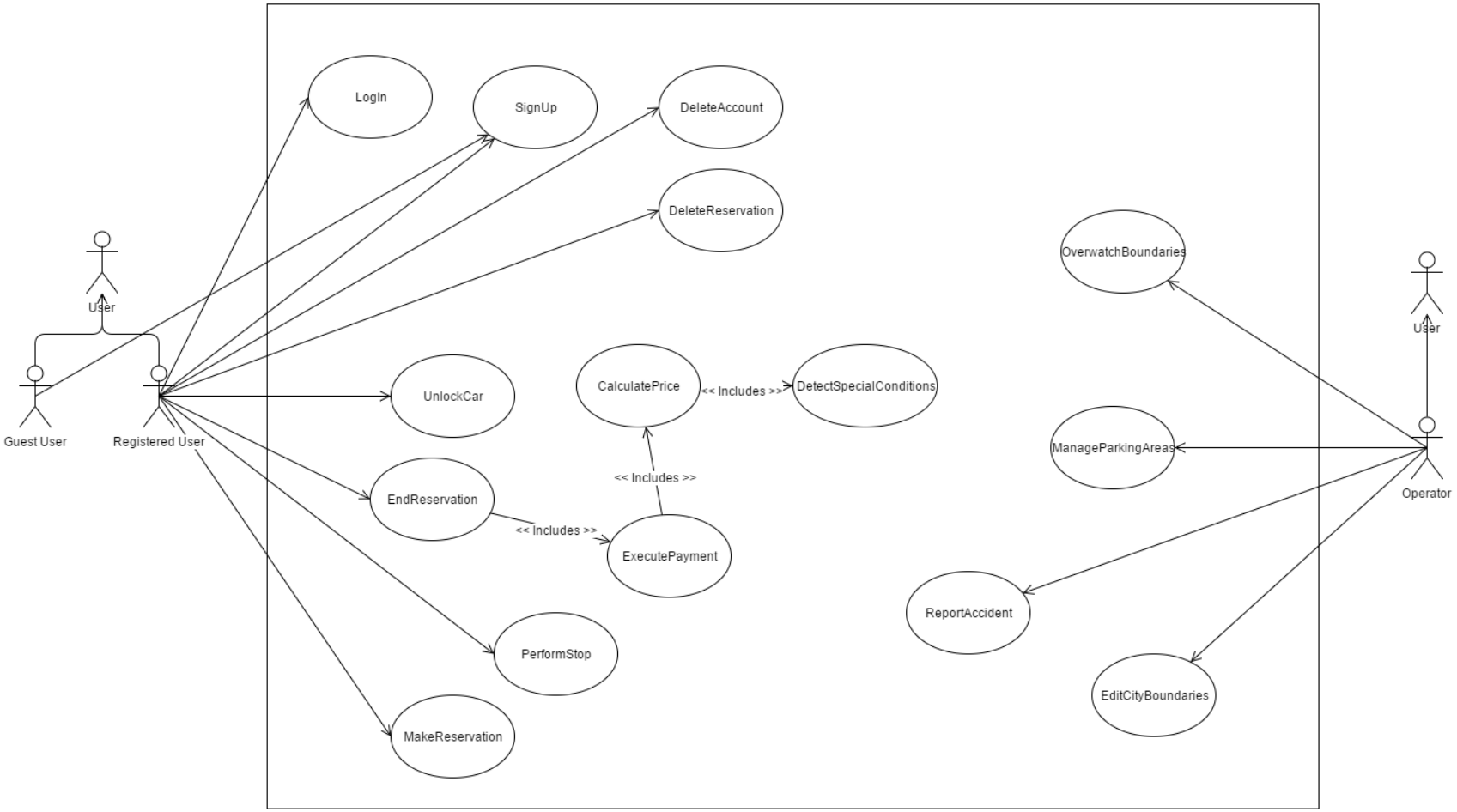
Exceptions:

- The comunication between the terminal and the System is interrupted. The Operator is notified of the event. The use case terminates
- The car's identifier specified does not exists. The Operator is notified by a notification error. The use case ends.
- The reservation associated to the vehicle is already expired. The Operator is warned through a notification error. The use case terminates.
- No reservation is associated to the specified car's identifier. The System warns the Operator through a notification error. The use case ends.
- If the timer associated to the authentication device reachs the zero, the Registered User gets notified. The use case ends.

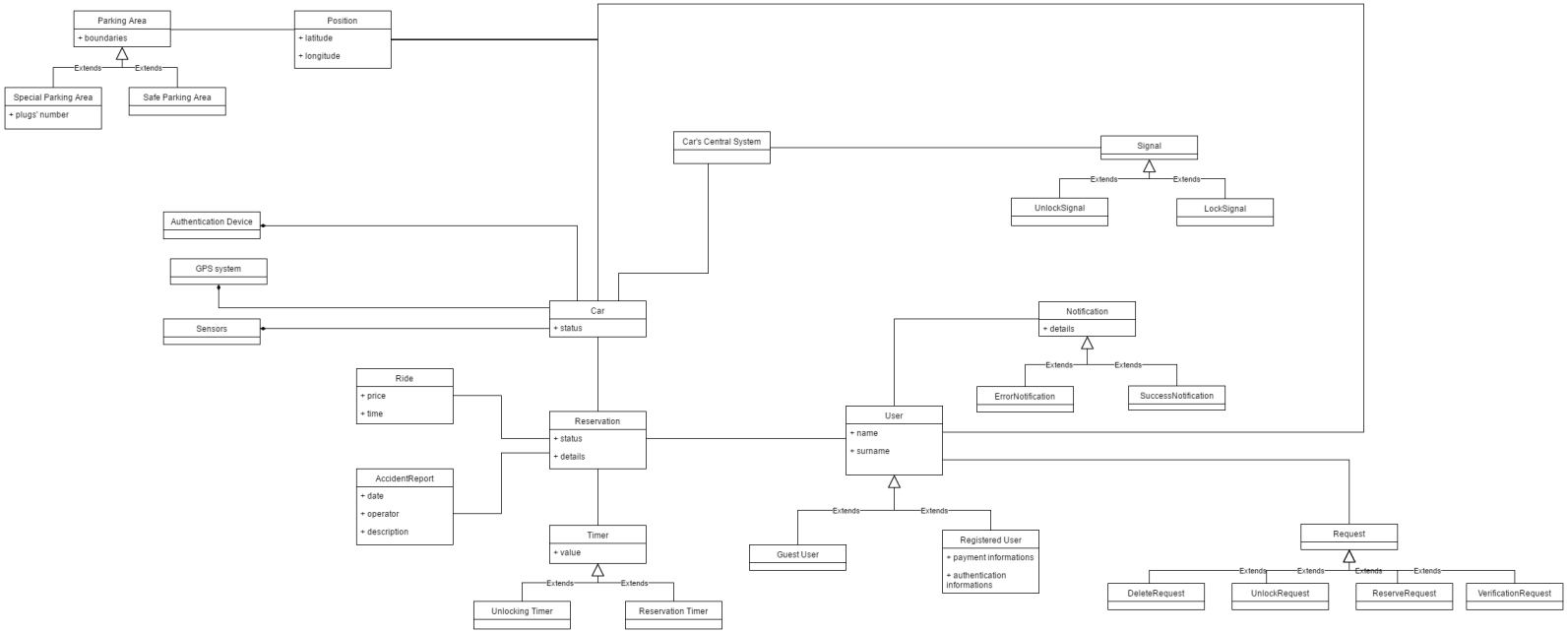
Appendix B UML diagrams

B.1 Use case diagram

Here we include a diagram which contains all the use cases that are relevant for this system.

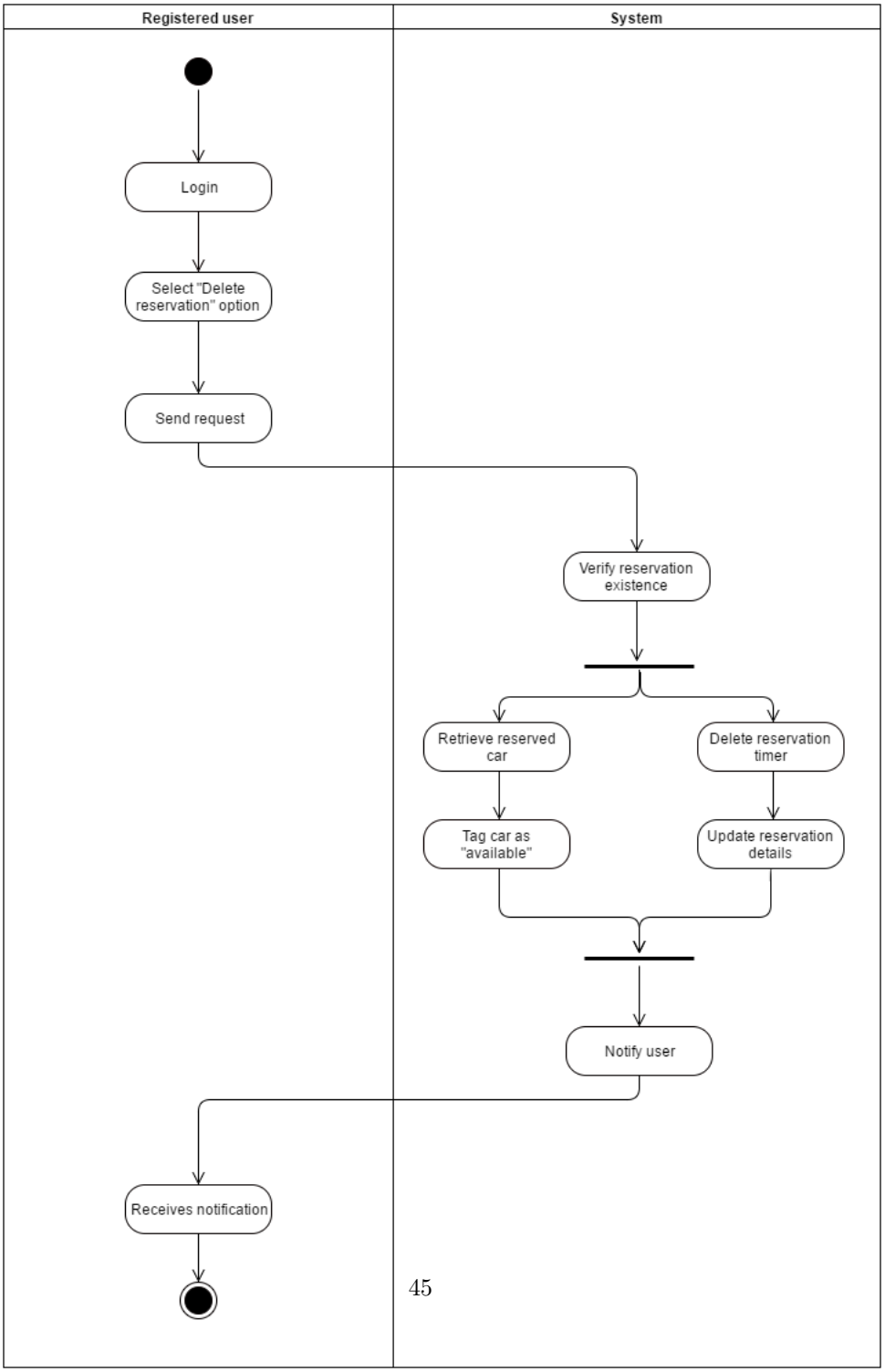

LogIn
SignUp
DeleteAccount
DeleteReservation
User
Guest User
Registered User
UnlockCar
CalculatePrice
<< Includes >>
DetectSpecialConditions
EndReservation
<< Includes >>
<< Includes >>
ExecutePayment
PerformStop
MakeReservation
OverwatchBoundaries
User
ManageParkingAreas
Operator
ReportAccident
EditCityBoundaries

B.2 Class diagram

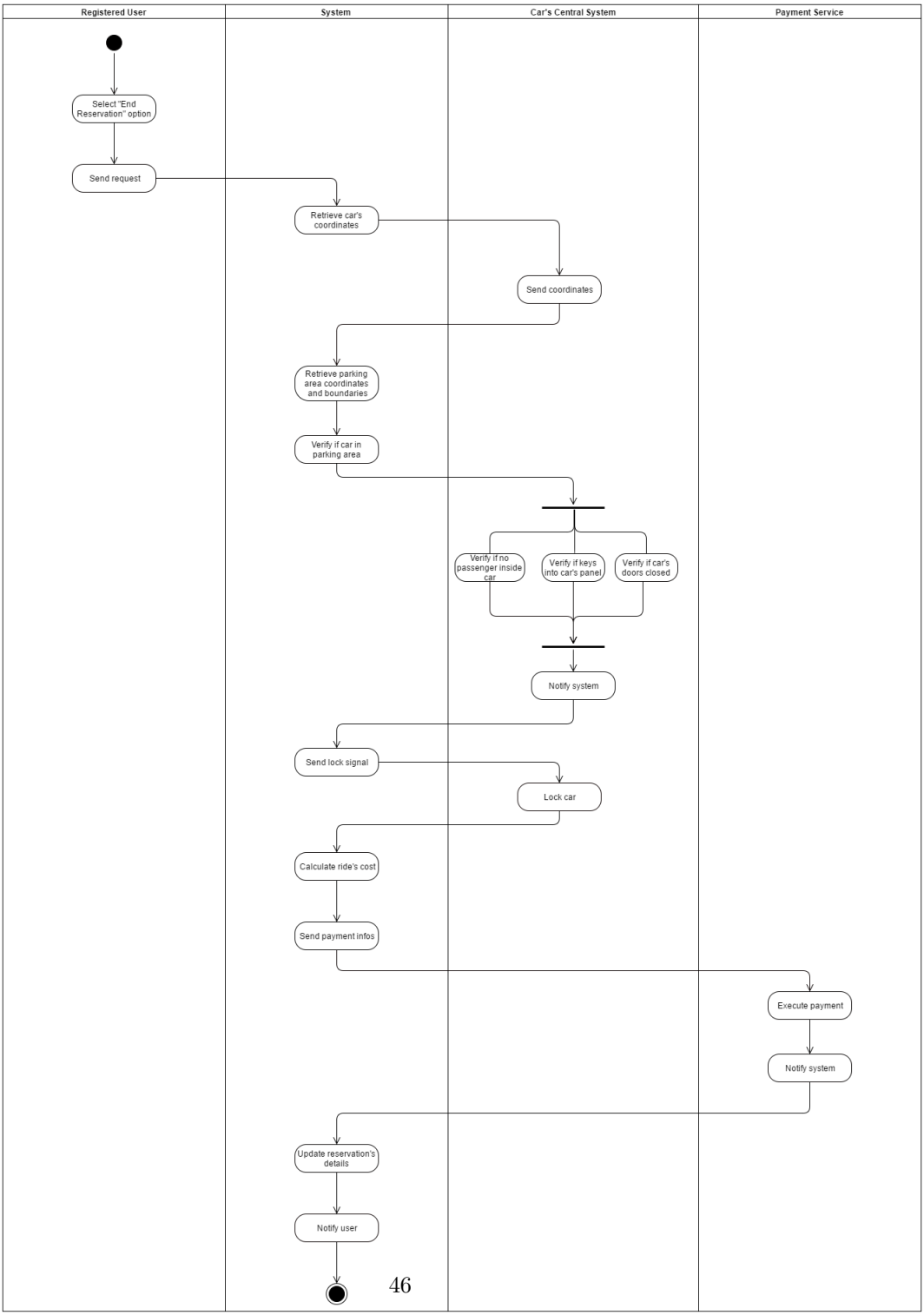

Parking Area
+ boundaries
Position
+ latitude
+ longitude
Extends
Extends
Special Parking Area
+ plugs' number
Safe Parking Area
Car's Central System
Signal
Extends
Extends
UnlockSignal
LockSignal
Authentication Device
GPS system
Sensors
Car
+ status
Notification
+ details
Extends
Extends
ErrorNotification
SuccessNotification
Ride
+ price
+ time
Reservation
+ status
+ details
User
+ name
+ surname
AccidentReport
+ date
+ operator
+ description
Timer
+ value
Extends
Extends
Guest User
Registered User
+ payment informations
+ authentication informations
Request
Extends
Extends
Extends
Extends
DeleteRequest
UnlockRequest
ReserveRequest
VerificationRequest
Unlocking Timer
Reservation Timer

B.3 Activity diagram

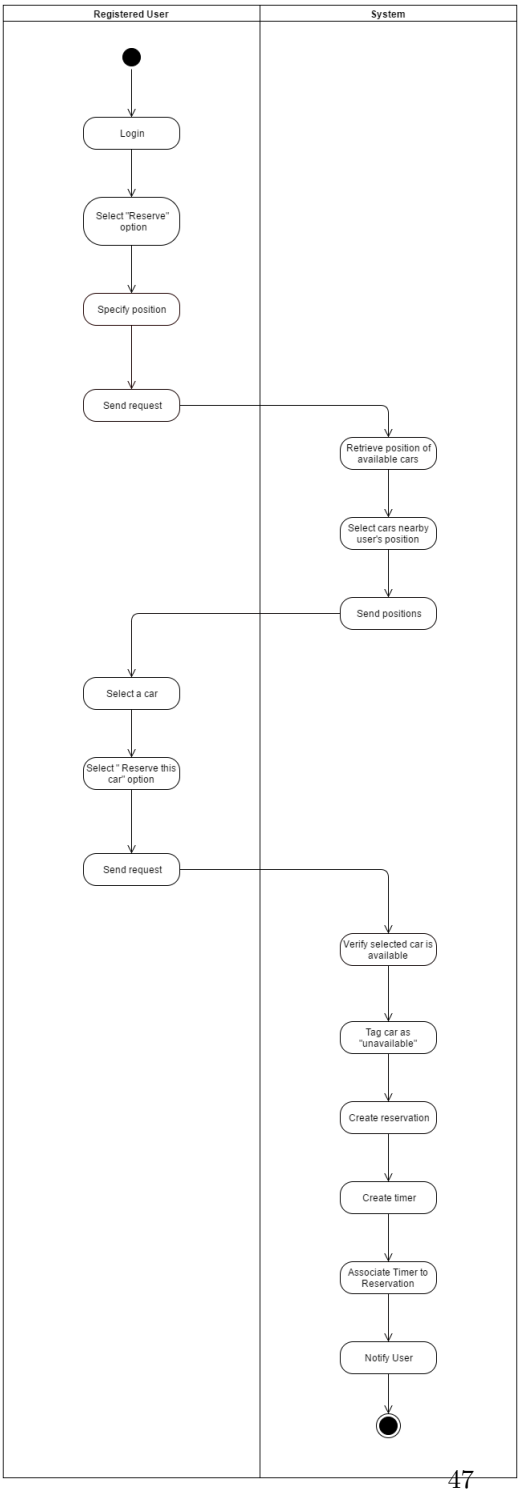
B.3.1 Delete a reservation



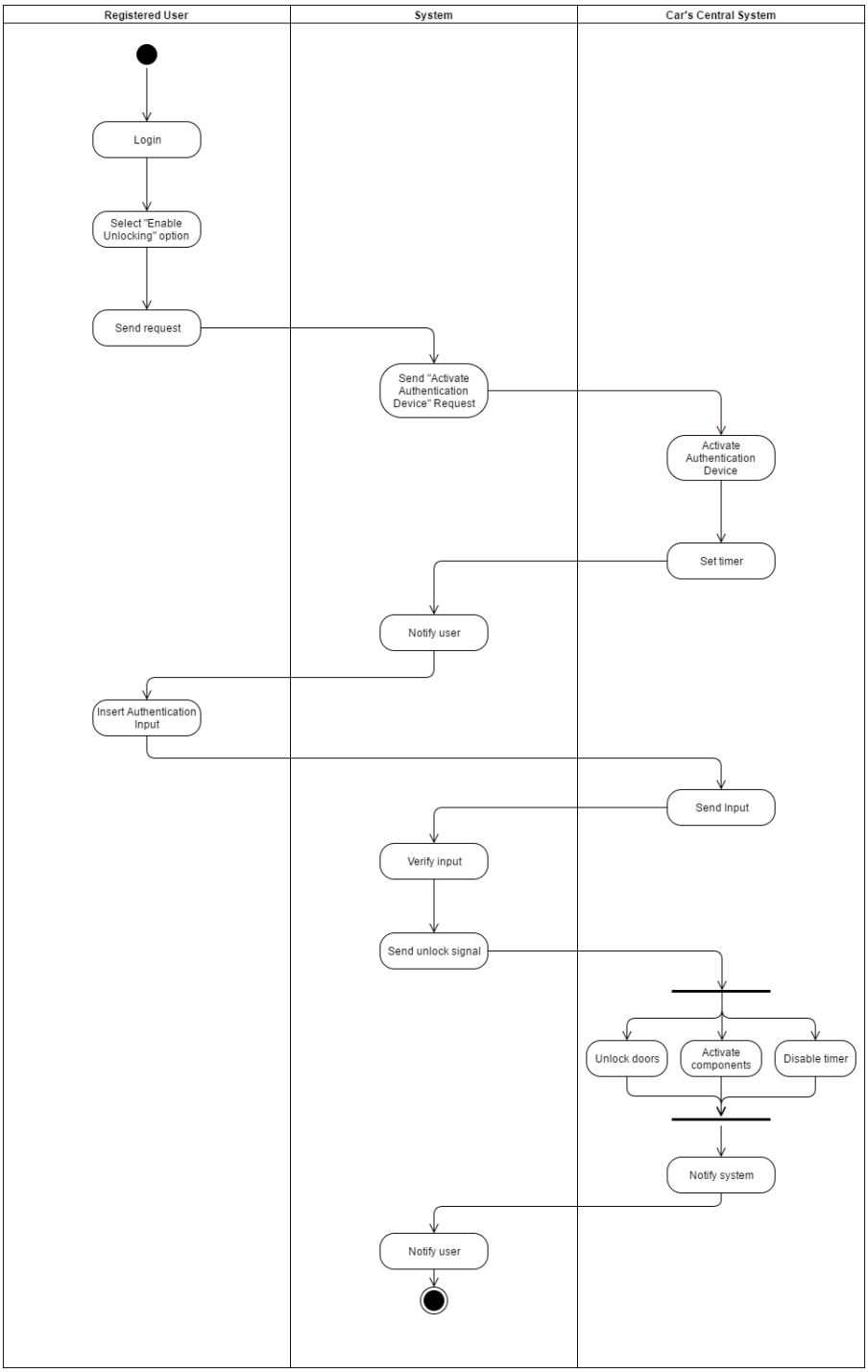
B.3.2 End a reservation



B.3.3 Make a reservation

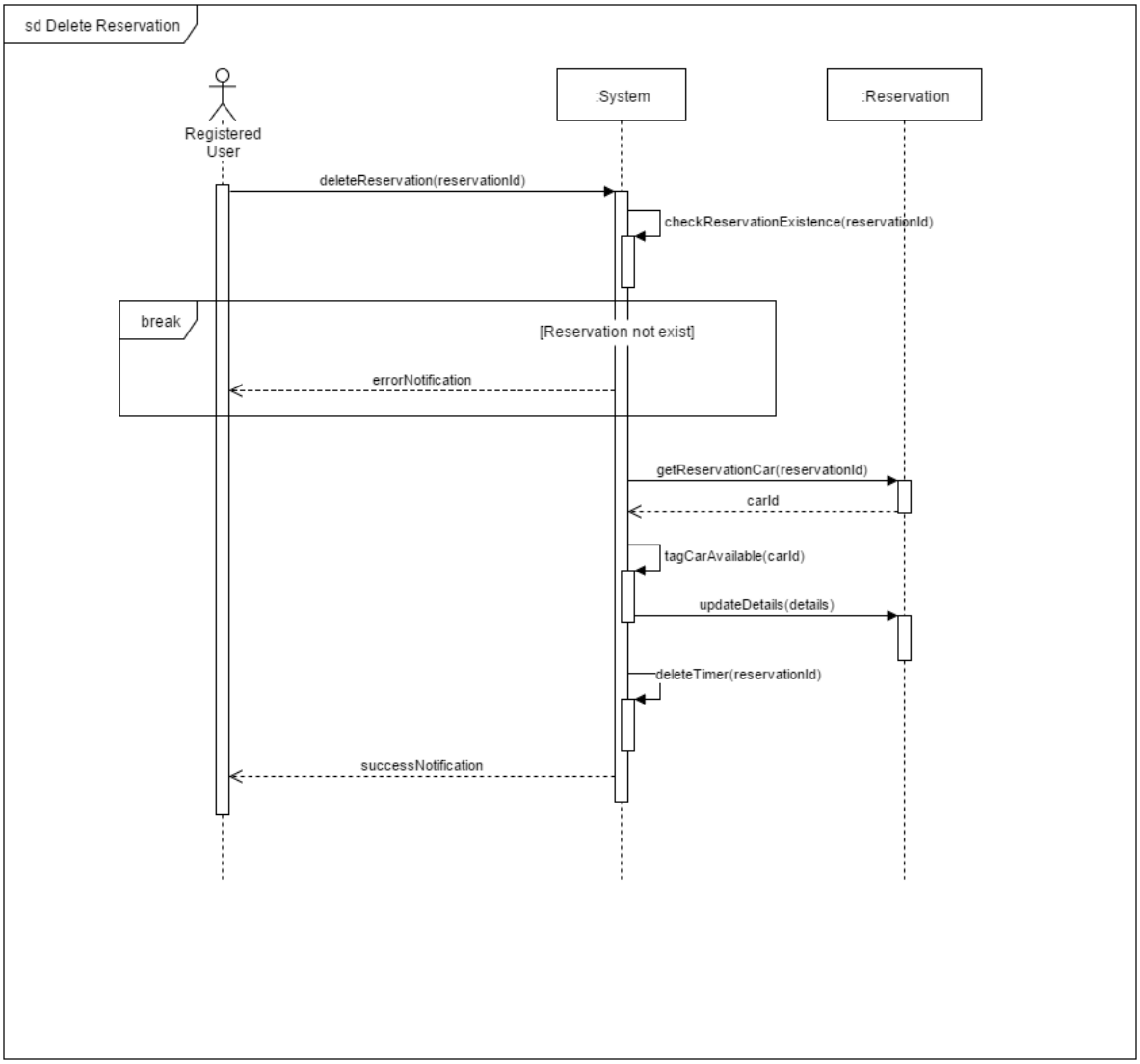


B.3.4 Unlock a car

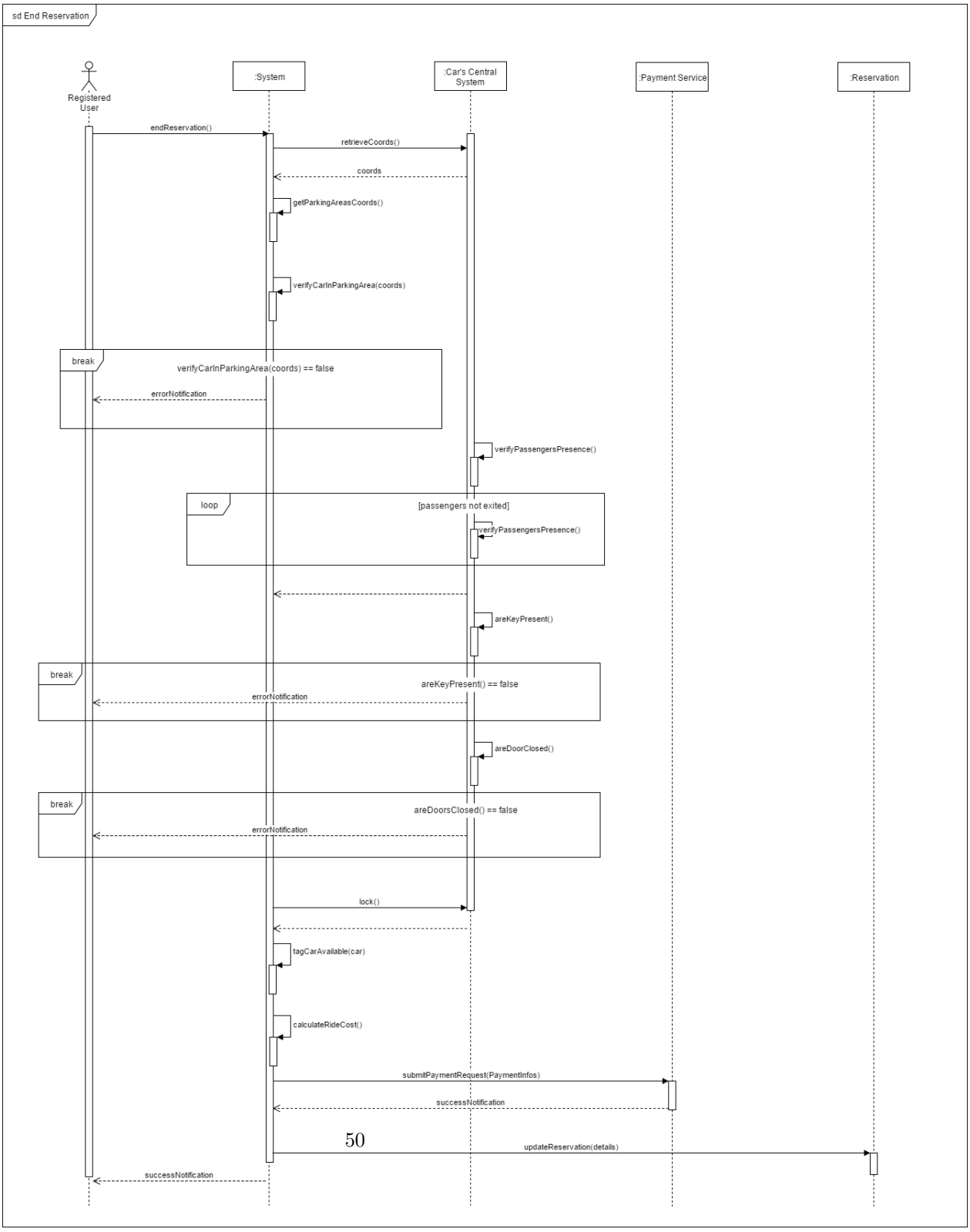


B.4 Sequence diagram

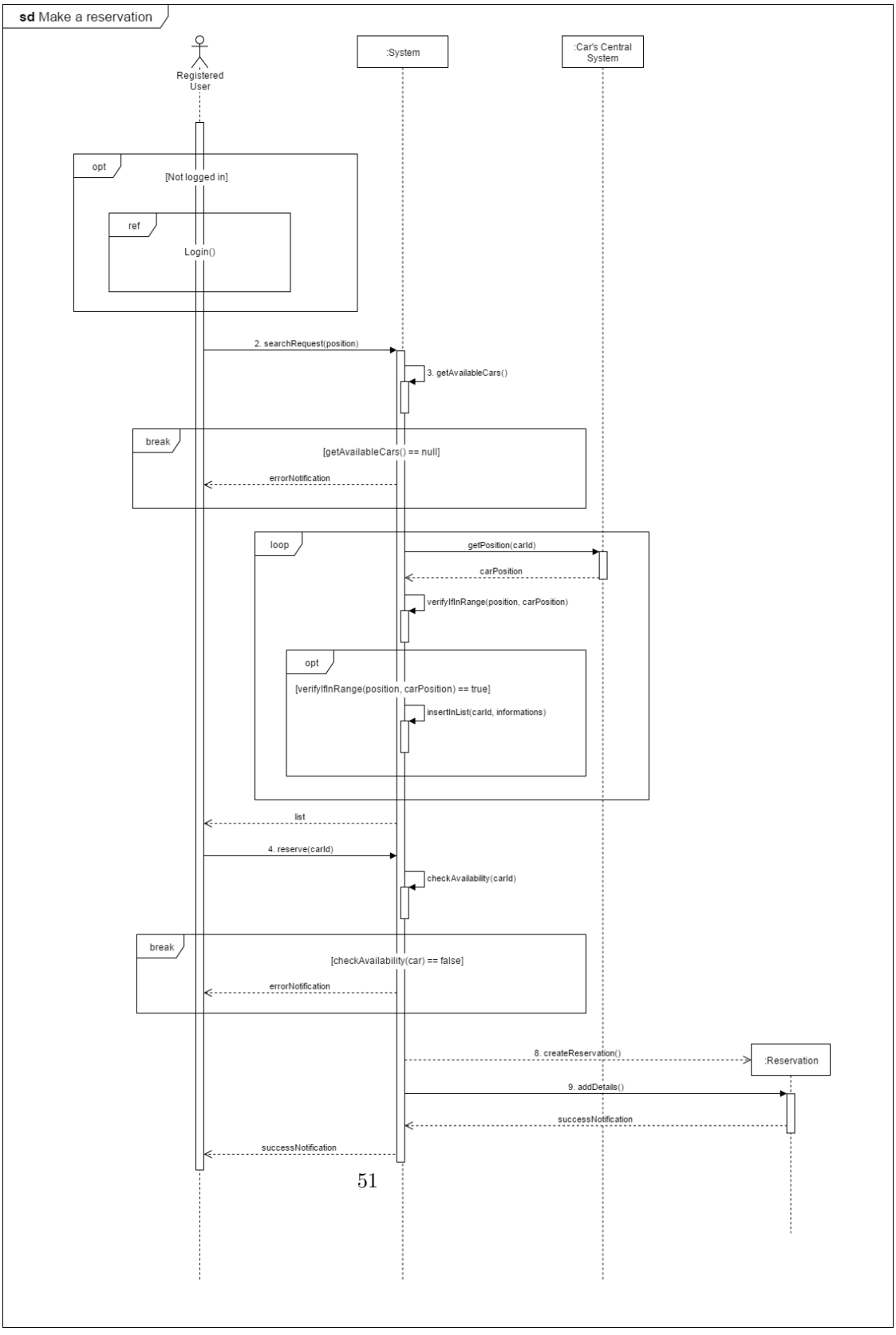
B.4.1 Delete a reservation



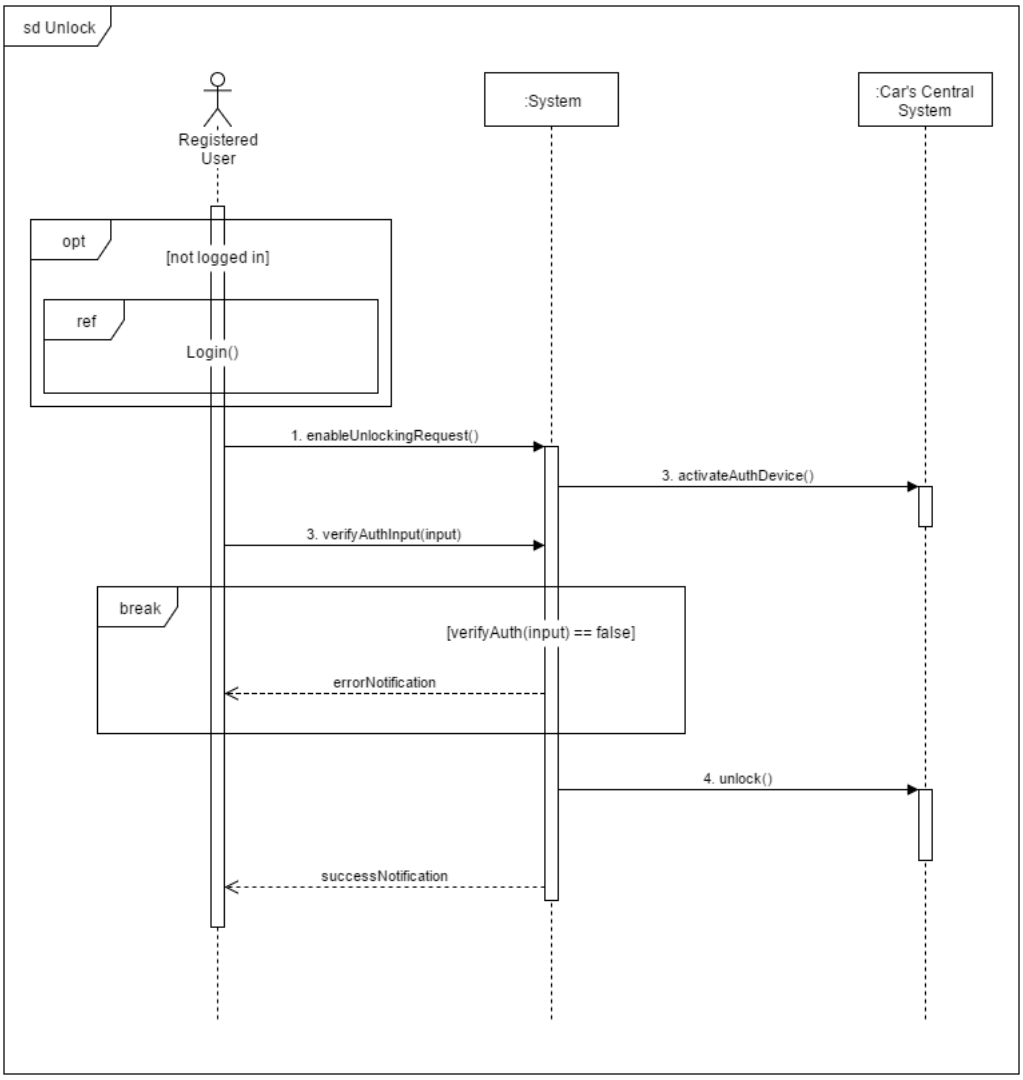
B.4.2 End a reservation



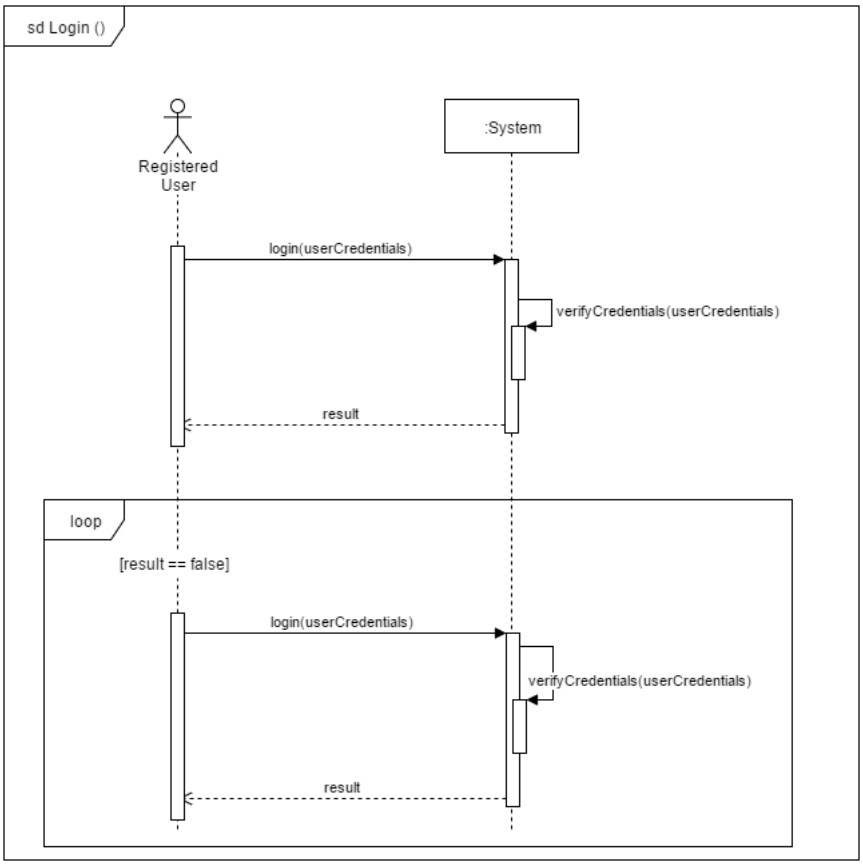
B.4.3 Make a reservation



B.4.4 Unlock a car

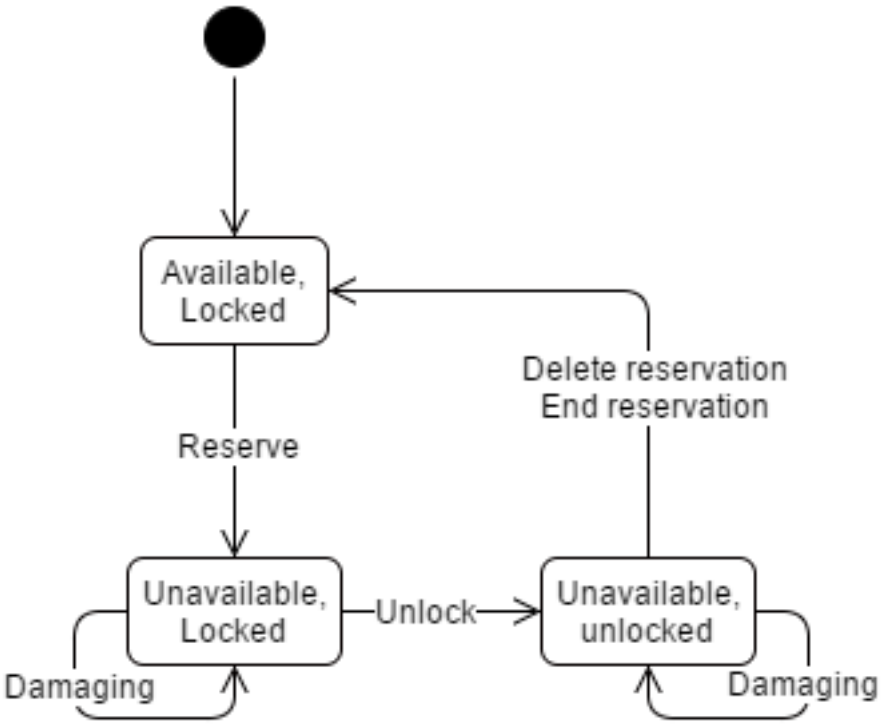


B.4.5 Login

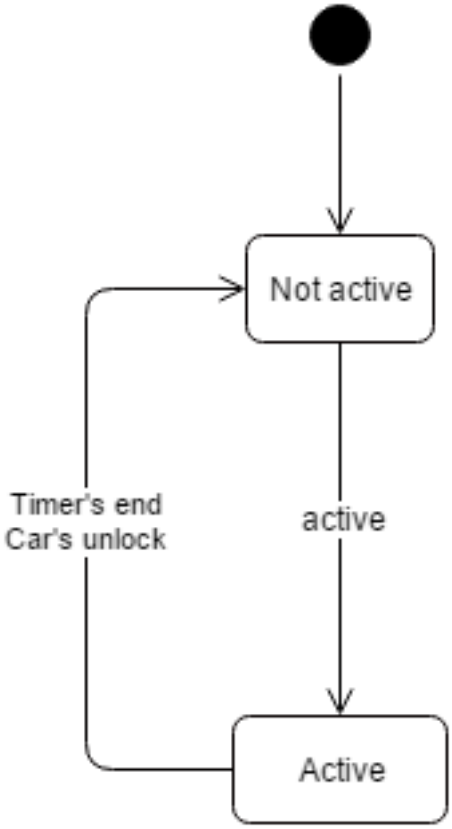

sd Login ()
Registered User
:System
login(userCredentials)
verifyCredentials(userCredentials)
result
loop
[result == false]
login(userCredentials)
verifyCredentials(userCredentials)
result

B.5 State chart diagram

B.5.1 Car state



B.5.2 Auhentication device state



Appendix C Alloy

The following code present a possible implementation of the system we are developing, additional functionalities can be implemented to reflect all the aspects of the real system

```
open util/boolean

//COMPANY
one sig Company { owns: some Car, registredUsers: some
User }

fact onlyCompanysCars{ all c: Car | c in Company.owns
}
```

```
//RESERVATIONS
abstract sig Reservation {
rid: one ReservationID, reservingUser: one User, reservedCar: one Car
}

sig ReservationID {}
{
ReservationID in Reservation.rid
}

sig PendingReservation extends Reservation {
reservationTimer: Int
}

sig ActiveReservation extends Reservation {}

sig EndedReservation extends Reservation {
endingReason: one EndingReason,
discountsApplied: some Discount,
penaltiesApplied: some Penalty,
finalCost: Int
}
{ finalCost > 0 }

fact uniqueReservationIDinSameReservationType {
all disj r1, r2: ActiveReservation | r1.rid != r2.rid
all disj r1, r2: PendingReservation | r1.rid != r2.rid
all disj r1, r2: EndedReservation | r1.rid != r2.rid
}

fact reservationTimerIsAnHour {
all pr: PendingReservation | pr.reservationTimer > Int[0]
and
pr.reservationTimer <= Int[60] }

fact uniqueReservedCar {
all disj r1, r2: (PendingReservation + ActiveReserva-
```

```
tion) |
r1.reservedCar & r2.reservedCar = none
}

fact uniqueReservingUser {
all disj r1, r2: (PendingReservation + ActiveReserva-
tion) |
r1.reservingUser & r2.reservingUser = none
}

fact pendingReservationCarsAreParked {
all c: Car | c in PendingReservation.reservedCar =>
c.isParked != none
}

abstract sig EndingReason {}

lone sig EndedByUser extends EndingReason {}

lone sig ReservationTimerEnded extends EndingReason
{}

lone sig EndedBySystem extends EndingReason {}

abstract sig Discount {}

lone sig NoDiscounts extends Discount {}

lone sig MoreThan2Passengers extends Discount {}

lone sig MoreThan50PercentBatteryLeft extends Discount
{}

lone sig CarPlugged extends Discount {}

abstract sig Penalty {}
```

```
lone sig LessThan20PercentBatteryOrFarFromPlugMoreThan3
extends Penalty {}

lone sig ServiceBoundsCrossed extends Penalty {}

lone sig NonAuthorizedParking extends Penalty {}

lone sig NoPenalties extends Penalty {}

fact endedReservationsConsistency {
all er: EndedReservation |
er.endingReason in ReservationTimerEnded =>
(er.finalCost = Int[0] and NoDiscounts in er.discountsApplied
and
NoPenalties in er.penaltiesApplied)
}

fact penaltiesConsistency {
all er: EndedReservation |
(NoDiscounts in er.discountsApplied => #er.discountsApplied
= 1) and
(NoPenalties in er.penaltiesApplied => #er.penaltiesApplied
= 1) }

sig VehicleID {} {
VehicleID in Car.vid
}

abstract sig PlugDistance {}

lone sig PlugDistanceMoreThan3 extends PlugDistance
{}

lone sig PlugDistanceLessThan3 extends PlugDistance
{}
```

```
fact plugDistanceConsistency {
all c: Car |
(c.isParked != none and c.isParked in PlugParkingSpot)
=>
c.plugDistance in PlugDistanceLessThan3

}

sig Position { latitude: Int, longitude: Int }

//CARS
sig Car {
batteryPercentage: Int,
status: one CarStatus,
isAvailable: Bool,
isParked: lone ParkingSpot,
isCharging: Bool,
carLocked: Bool,
plugDistance: one PlugDistance,
vid: one VehicleID,
carPosition: one Position,
passengers: Int
}

abstract sig CarStatus {}

lone sig CarParked extends CarStatus {}

lone sig CarStopped extends CarStatus {}

lone sig CarDriven extends CarStatus {}

fact carStatusContraints {
all c: Car |
( (c.status in CarParked =>
(c.isParked != none and c.isParked in ParkingSpot and
isTrue[c.carLocked])) and
(c.status in CarStopped =>
```

```
(c.isParked = none and c in ActiveReservation.reservedCar
and
isTrue[c.carLocked])) and
(c.status in CarDriven =>
(c.isParked = none and c in ActiveReservation.reservedCar
and
!isTrue[c.carLocked])) ) }

fact chargingConstraint {
all c: Car | isTrue[c.isCharging] =>
(c.isParked != none and c.isParked in PlugParkingSpot)
}

fact passengersConstraint {
all c: Car | c.passengers >= 0
c.passengers <= 4 and
(c not in ActiveReservation.reservedCar => c.passengers
= 0) }

fact uniqueVehicleID { all disj c1, c2: Car | c1.vid !=
c2.vid }

fact availabilityConstraints {
all c: Car | isTrue[c.isAvailable] =>
(c.isParked != none and c.batteryPercentage > Int[20]
and
c not in (ActiveReservation.reservedCar + PendingReser-
vation.reservedCar)) }

fact batteryPercentageIsPercentual {
all c: Car | (c.batteryPercentage > Int[0] and
c.batteryPercentage < Int[100]) }

//USERS
sig User {
uid: one UserID,
pi: one PaymentInformations
} { User in Company.registredUsers }
```

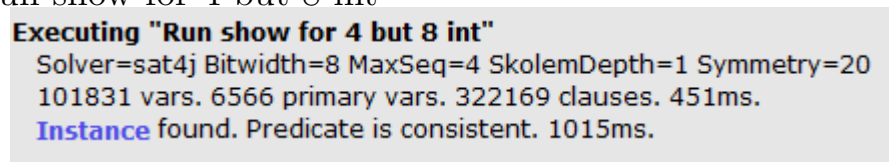
```
sig PaymentInformations {} { PaymentInformations in
User.pi }

sig UserID {} { UserID in User.uid }

fact uniquePaymentInformation { all disj u1, u2: User |
u1.pi != u2.pi }

fact uniqueUserID { all disj u1, u2: User | u1.uid !=
u2.uid }

//PARKING
sig Plug {} { Plug in PlugParkingSpot.plug }

sig ParkingSpotID {} { ParkingSpotID in ParkingSpot.psid
}

abstract sig ParkingSpot { psid: one ParkingSpotID }

fact uniqueParkedCar{
all disj c1, c2: Car | c1.isParked != none =>
(c1.isParked not in c2.isParked) }

fact uniqueParkingSpotID {
all disj ps1, ps2: ParkingSpot| ps1.psid != ps2.psid
}

sig SafeParkingSpot extends ParkingSpot {} {
SafeParkingSpot in SafeParkingArea.safeParkingSpots
}

sig PlugParkingSpot extends ParkingSpot { plug: one
Plug }
{
PlugParkingSpot in SpecialParkingArea.plugParkingSpots
}
```

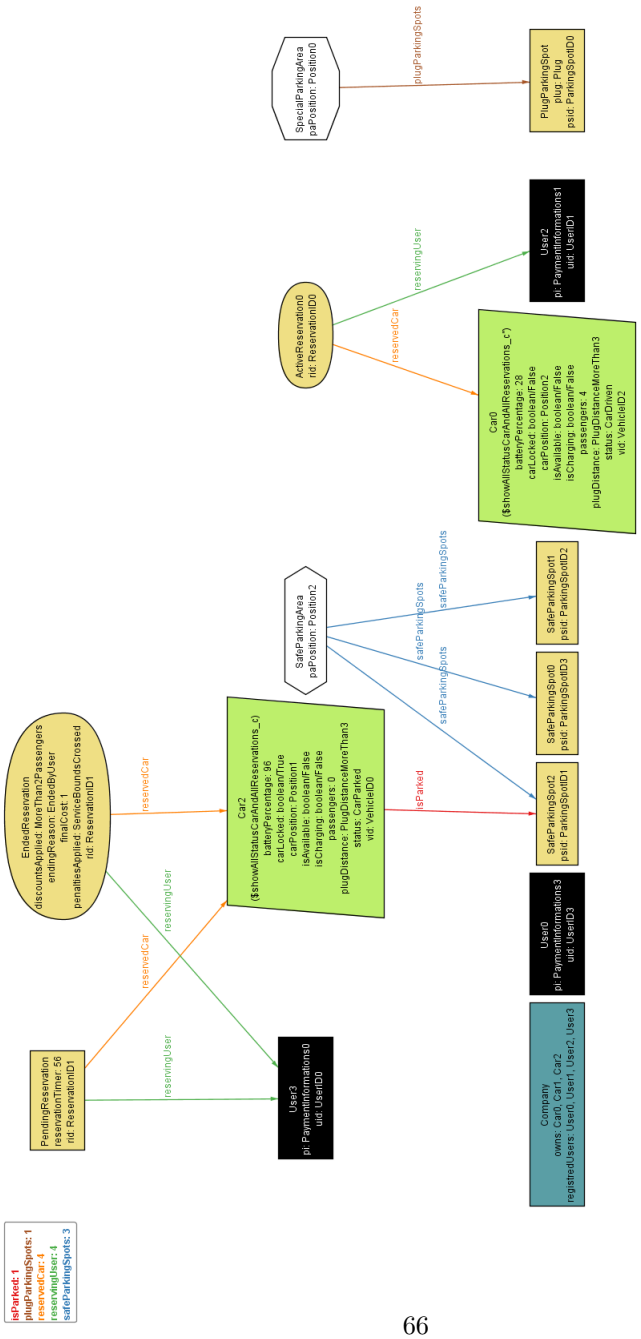
```
abstract sig ParkingArea { paPosition: one Position }

sig SafeParkingArea extends ParkingArea{
safeParkingSpots: some SafeParkingSpot
}

fact disjointSafeParkingSpots {
all disj safepa1, safepa2 : SafeParkingArea |
safepa1.safeParkingSpots & safepa2.safeParkingSpots =
none }

sig SpecialParkingArea extends ParkingArea{
plugParkingSpots: some PlugParkingSpot
}

fact disjointPlugParkingSpots {
all disj specialpa1, specialpa2 : SpecialParkingArea |
specialpa1.plugParkingSpots & specialpa2.plugParkingSpots
= none }

fact disjointParkingAreas {
all disj pa1, pa2: ParkingArea |
pa1.paPosition & pa2.paPosition = none
}

fact uniquePlugs {
all disj pps1, pps2: PlugParkingSpot |
let p1 = pps1.plug, p2 = pps2.plug |
(p1 & p2 = none)
}

fact ParkingSpotsConstraints {
#ParkingSpot >= #Car #PlugParkingSpot > 0
}

fact AlwaysCars { #Car > 0 }
```

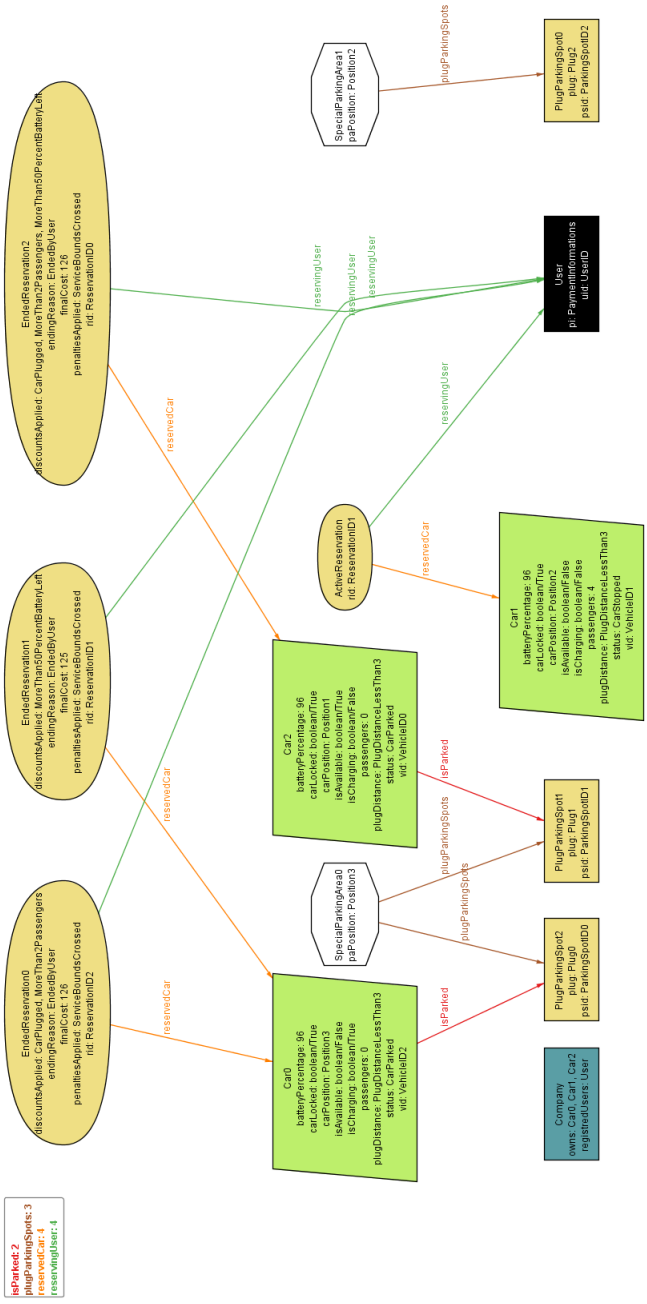
```
pred userEndsReservation [c, c': Car, er: EndedReservation] {
c in (ActiveReservation.reservedCar + PendingReservation.reservedCar)
c.isParked != none
c' not in (ActiveReservation.reservedCar + PendingReservation.reservedCar)
er.reservedCar = c'
er.endingReason = EndedByUser
(c.plugDistance = PlugDistanceMoreThan3) <=>
(LessThan20PercentBatteryOrFarFromPlugMoreThan3 in er.penaltiesApplied)
(c.batteryPercentage > Int[50]) <=>
(MoreThan50PercentBatteryLeft in er.discountsApplied)
(c.passengers > Int[1]) <=>
(MoreThan2Passengers in er.discountsApplied)
(isTrue[c.isCharging]) <=>
(CarPlugged in er.discountsApplied)

   let r = (reservedCar.c & (ActiveReservation + PendingReservation)) |
er.reservingUser = r.reservingUser }
//run userEndsReservation for 8 int

pred carReservation [c, c': Car, u: User, pr: PendingReservation] {
isTrue[c.isAvailable]
!isTrue[c'.isAvailable]
c' in pr.reservedCar
all r: (PendingReservation + ActiveReservation) | r != pr =>
u not in r.reservingUser
u in pr.reservingUser
pr.reservationTimer = Int[60]
}
//run carReservation for 8 int
```

```
pred showCarReservation[c, c': Car, u: User, pr: Pend-
ingReservation] {
carReservation[c, c', u, pr]
#Car = 2 }
//run showCarReservation for 8 int

assert reservationConsistency {
no c: Car | c in ActiveReservation.reservedCar and
c in PendingReservation.reservedCar
no u: User | u in ActiveReservation.reservingUser and
u in PendingReservation.reservingUser
}
//check reservationConsistency

assert DrivenMeansUnlocked{
no c: Car | c.status in CarDriven and isTrue[c.carLocked]
}
//check DrivenMeansUnlocked

pred showChargingCar {
some c: Car | isTrue[c.isCharging]
}
//run showChargingCar for 8 int

pred showAllStatusCar {
some c: Car | c.status in CarParked
some c: Car | c.status in CarStopped
some c: Car | c.status in CarDriven
}
//run showAllStatusCar for 8 int

pred showAvailableCar {
some c: Car | isTrue[c.isAvailable]
}
//run showAvailableCar for 8 int
```

```
pred showAllStatusCarAndAllReservations {
some c: Car | c.status in CarParked
some c: Car | c.status in CarStopped
some c: Car | c.status in CarDriven
#EndedReservation > 0
#ActiveReservation > 0
#PendingReservation > 0 }
//run showAllStatusCarAndAllReservations for 4 but 8
int

pred show {}

run show for 4 but 8 int
```




SpecialParkingArea
paPosition: Position0
plugParkingSpots
PlugParkingSpot
plug: Plug
psid: ParkingSpotID0
User2
pi: PaymentInformations1
uid: UserID1
ActiveReservation0
rid: ReservationID0
reservingUser
reservedCar
Car0
($showAllStatusCarAndAllReservations_c*)
batteryPercentage: 28
carLocked: booleanFalse
carPosition: Position2
isAvailable: booleanFalse
isCharging: booleanFalse
passengers: 4
plugDistance: PlugDistanceMoreThan3
status: Charging
vid: VehicleID2
SafeParkingArea
paPosition: Position2
safeParkingSpots
SafeParkingSpot1
psid: ParkingSpotID2
SafeParkingSpot0
psid: ParkingSpotID3
SafeParkingSpot2
psid: ParkingSpotID1
EndedReservation
discountsApplied: MoreThan2Passengers
endingReason: EndedByUser
finalCost: 1
penalitesApplied: ServiceBoundsCrossed
rid: ReservationID1
reservedCar
reservingUser
Car2
($showAllStatusCarAndAllReservations_c)
batteryPercentage: 96
carLocked: booleanTrue
carPosition: Position1
isAvailable: booleanFalse
isCharging: booleanFalse
passengers: 0
plugDistance: PlugDistanceMoreThan3
status: Completed
vid: VehicleID0
isParked
User0
pi: PaymentInformations3
uid: UserID3
PendingReservation
reservationTimer: 56
rid: ReservationID1
reservedCar
reservingUser
User3
pi: PaymentInformations0
uid: UserID0
Company
owns: Car0, Car1, Car2
registredUsers: User0, User1, User2, User3
isParked: 1
plugParkingSpots: 1
reservedCar: 4
reservingUser: 4
safeParkingSpots: 3


isParked: 2
plugParkingSpots: 3
reservedCar: 4
reservingUser: 4
EndedReservation0
discountsApplied: CarPlugged, MoreThan2Passengers
endingReason: EndedByUser
finalCost: 125
penalitiesApplied: ExceedBoundsCrossed
rid: ReservationID2
EndedReservation1
discountsApplied: MoreThan50PercentBatteryLeft
endingReason: EndedByUser
finalCost: 125
penalitiesApplied: ExceedBoundsCrossed
rid: ReservationID1
EndedReservation2
discountsApplied: CarPlugged, MoreThan2Passengers, MoreThan50PercentBatteryLeft
endingReason: EndedByUser
finalCost: 125
penalitiesApplied: ExceedBoundsCrossed
rid: ReservationID0
Car0
batteryPercentage: 96
carLocked: booleanTrue
carPosition: Position3
isAvailable: booleanFalse
isCharging: booleanTrue
passengers: 0
plugDistance: PlugDistanceLessThan3
status: CarParked
vid: VehicleID2
Car2
batteryPercentage: 96
carLocked: booleanTrue
carPosition: Position1
isAvailable: booleanTrue
isCharging: booleanFalse
passengers: 0
plugDistance: PlugDistanceLessThan3
status: CarParked
vid: VehicleID0
Car1
batteryPercentage: 96
carLocked: booleanTrue
carPosition: Position2
isAvailable: booleanFalse
isCharging: booleanFalse
passengers: 4
plugDistance: PlugDistanceLessThan3
status: CarStopped
vid: VehicleID1
ActiveReservation
rid: ReservationID1
SpecialParkingArea0
paPosition: Position3
SpecialParkingArea1
paPosition: Position2
PlugParkingSpot0
plug: Plug2
psid: ParkingSpotID2
PlugParkingSpot1
plug: Plug1
psid: ParkingSpotID1
PlugParkingSpot2
plug: Plug0
psid: ParkingSpotID0
User
pi: PaymentInformations
uid: UserID
Company
owns: Car0, Car1, Car2
registeredUsers: User
reservedCar
reservingUser
isParked
plugParkingSpots

Appendix D Hours of work

To build this document we spent about 50 hours per person