PowerEnJoy

Software Engineering II

Requirements Analysis and Specification Document

Giovanni Scotti, Marco Trabucchi

Document version: 1

November 3, 2016

Contents

Section 1

Introduction

1.1   Purpose

The Requirement Analysis and Specification Document for the *PowerEnJoy* digital management system is intended to describe the system itself, the functional and non-functional requirements, its components as well as its constraints and the relationship with the real world and the users by providing several use cases and scenarios. Furthermore part of the documentation makes use of Alloy, a language for describing structures and a tool for exploring them, and gives a formal specification of some features of the system to be.

This document establishes some baselines for the project planning, its estimation and evaluation and it may be legally binding; it is mainly addressed to developers and programmers, who have to implement the requirements, testers, who have to determine whether the requirements have been met, project managers, who control the development process and - last but not least - users, who validate the system goals.

1.2   Scope

The product is a digital management system to support a car-sharing service that exclusively employs electric cars.

The system consists of a back-end server application that manages rental requests remotely and three front-end applications:

- A web-based application to provide the final user with a friendly interface to take advantage of the services of *PowerEnJoy*;

- An application that runs on the existing on-board computers provided on each vehicle, used to interact with the car itself, unlock it and access the GPS/sat-nav service;

- A mobile application that allows the user to easily access the service anywhere he/she needs to.

The system is mainly intended for one type of user: drivers, who should be allowed to register and access the system via username and password, in order to make the renting and payment processes easier and quicker to carry out. Moreover, the system aids the users by locating nearby available vehicles and keeps track of the distance driven, all while notifying them about the amount of money they are being charged. A defined Safe Area stands as a limit for the service; properly equipped charging spots are signaled by an on-board computer.

The system aims to motivate drivers to maintain a virtuous behavior providing discounts when it detects signs of responsible and ecologic actions.

Lastly, the system is also in charge to inform the maintenance division about out-of-order vehicles so that the technicians can work out any kind of issue.

1.3 Goals

The goals of *PowerEnJoy* are the following:

1. Let the user register to the service and login via the provided credentials;

2. Let the user manage his/her own profile;

3. Let the driver find the location of nearby available cars and reserve a chosen car up to an hour before picking it up;

4. Improve the efficiency of the service by assuring that no car stays reserved for more than an hour if not actually in use;

5. Allow the user to easily access the cars by unlocking them once the driver is in proximity;

6. Automatically manage payments in order to make the service quicker and more dynamic to use;

7. Allow every registered user who reserved a car to drive it for a ride;

8. Manage the security of the cars by locking them properly after every ride;

9. Incentivize responsible behaviors, providing discounts for the worthiest users and additional charges for bad users.

1.4 Definitions, Acronyms and Abbreviations

Active reservation: a reservation of a vehicle that is not older than an hour, meaning that the user that performed it can still claim and use the reserved vehicle.

API: Application Programming Interface. A set of tools, protocols and libraries for building software and applications.

Back-end application: any computer program that remains in the background and offers application logic and communication interfaces to work with the front-end counterpart. It does not involve any graphical user interface, but it can provide a data access layer.

Charging spot: a particular parking spot equipped with a power grid and plugs to recharge the cars.

CAN bus: the Controller Area Network is a vehicle bus standard designed to allow microcontrollers and other devices to communicate with each other without a host computer.

DBMS: Database Management System.

Driver: See **User**.

Front-end application: any application the users interact with directly. It provides the so called presentation layer.

GPS: Global Positioning System.

m: meter, *Interational System of Units (SI)* unit of measure.

Mobile broadband: it is the marketing term for wireless internet access delivered through mobile phone towers to any digital device using a portable modem.

OS: Operating System.

Parking spot: a generic place whitin the Safe Area where the driver can park the vehicles.

PC: Personal Computer.

PIN: Personal Identification Number.

RASD: Requirements Analysis and Specification Document.

Ride: the trip that involves the use of a *PowerEnJoy* electric car.

Safe Area: the predefined area where it is possible to start and end a ride; it marks the boundaries within which the user can park a vehicle and conclude the ride so that the system stops charging him/her.

System: The software system-to-be, in all of its entirety.

UML: Unified Modeling Language, the modeling language used in this document.

UMTS: Universal Mobile Telecommunications System.

User: Any person subscribed to the service who rents a car using *PowerEnJoy*.

Vehicle: Any of the electric cars provided by *PowerEnJoy*.

W3C: World Wide Web Consortium. It is the main international standards organization for the World Wide Web, founded and led by Tim Berners-Lee.

1.5   References

This document follows the guidelines provided by ISO/IEC/IEEE 29148:2011 [1] and IEEE 830-1998 [2] respectively related to the requirements engineering for systems and software products and the recommended practice for software requirements specifications.

Moreover it is strictly based on the specifications concerning the RASD assignment [3] for the Software Engineering II project, part of the course held by professors Luca Mottola and Elisabetta Di Nitto at the Politecnico di Milano, A.Y. 2016/17.

1.6   Overview

This document consists of three sections:

Section 1: Introduction. A general introduction and overview of the system-to-be purpose, scope and goals, along with some important information about this document.

Section 2: Overall description. It describes the general factors that affects the product and its requirements. The section provides a background for those requirements which are defined in detail in Section 3 and makes them easier to figure out.

Section 3: Specific Requirements. All the software requirements are specified to a level of detail which is sufficient to let the designers satisfy them. Both functional and non-functional requirements are mentioned.

At the end of the document are an **Appendix** and a **Bibliography**, providing additional information about the sections listed above.

Section 2

Overall Description

2.1 Product Perspective

2.1.1 User interfaces

The users have several ways to access the system: a web application can be executed on any personal computer while a mobile application provides flexibility, portability and can be used literally everywhere. Despite the fact that the hardware interfaces running the application are rather different, a unified and common user interface is provided. It should be user friendly and very intuitive to allow everyone to easily use it without any specific knowledge.

Moreover the users have to interact with the on-board computer installed on each electric vehicle, therefore it should offer an interface as straightforward as the one implemented by the web and mobile applications.

2.1.2 Hardware interfaces

The web application can be executed on any general purpose computer that complies with the minimum system requirements specified in subsection 2.4.2.

The mobile application has to exchange data with the GPS module located on any recent smart-phone. Moreover it has to access the mobile broadband in order to communicate with the main system server.

An on-board computer is already set up in each electric car and it talks to the vehicle control unit through the CAN bus and to the system server via the mobile broadband. All sensors and hardware components needed to support the functionalities of the system are already installed on the vehicles.

2.1.3 Software interfaces

The web based application must support the main browsers such as IE, Google Chrome and Mozilla Firefox.

The mobile application has to be compatible with iOS and Android. The server side of the application, that is the system back-end, stores data in a relational DBMS and runs on any web server supporting Java.

The on-board computers run a customized Linux-based kernel and provide adequate APIs for the developers to communicate easily with the existing vehicle systems. This is to make communication between the system-to-be and the existing software easier.

The back-end has mainly to deal with the rental service and data management, essentially aimed towards keeping track of transactions and customers information. Moreover it has the aim to communicate with the maintenance technicians sending information about out-of-service vehicles via simple automatic e-mail messages.

2.2 Product Functions

The system allows users to reserve available cars, to manage their requests and charge them for the rental.

The users can:

- Register to the service;

- Login with their personal account;

- Edit and manage personal information;

- Delete their existing account;

- Locate nearby available-to-rent vehicles;

- Reserve one of said cars for a fixed amount of time;

- Unlock the reserved vehicle based on their mobile device GPS position;

- Alternatively, unlock the reserved car by inserting a vehicle-specific code (found on the windshield) into the application;

- Start the engine by inserting their PIN;

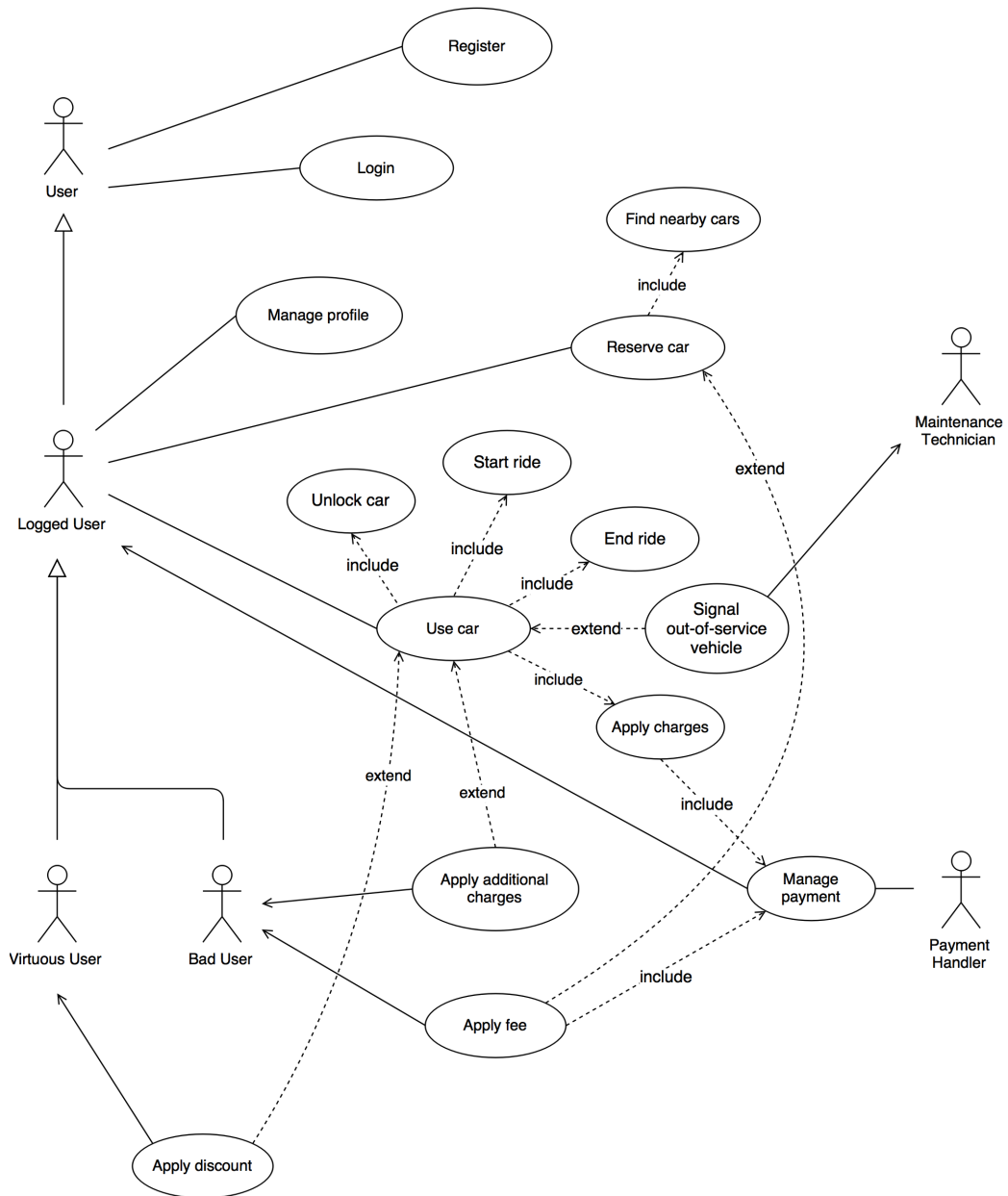- Perform payments for the used service via credit card or $PayPal^{TM}$.

Figure 2.1: The comprehensive use-case diagram of all the functionalities provided by the system

2.3 User Characteristics

Target of the service are *drivers* that wish to rent a car.

The base assumption is that every registered user is in possess of a driving

license which is valid in the country where he/she makes use of the service.

Moreover, the user must have the means of accessing the service from anywhere: this includes having the possibility to use a stable internet connection both from a mobile device and a stationary one such as a laptop or desktop PC.

Lastly, the user should provide the system a valid payment method upon the act of registration to the service.

2.4 Constraints

2.4.1 Regulatory policies

The system must be allowed by the user to collect, process and store personal data. Furthermore, the system is capable to delete all personal data upon user request and to keep track of each payment.

The user has the responsibility to use the system properly in order to comply with the local laws and policies.

2.4.2 Hardware limitations

The system should comply to these following minimum hardware requirements:

- Mobile application:
  - ▶ 3G UMTS connection at its maximum speed of 2 Mb/s
  - ▶ 50 MB of available space
  - ▶ 1 GB of RAM
  - ▶ GPS module

- Web application:
  - ▶ Internet connection at 7 Mb/s
  - ▶ 800x600 screen resolution

2.4.3 Parallel operation

The system must support parallel operations from different users and the DBMS relies heavily on concurrent transactions.

2.4.4 Reliability requirements

The system reliability, that is the probability to operate without a failure for a specific period of time, must be 99%.

2.4.5 Criticality of the application

Life-critical applications do not concern the system to be developed.

2.4.6 Safety and security consideration

The user must have a valid driving license in order to take advantage of the car sharing service. The license number is asked and stored by the system for security reason. The locations and the travels performed by the users must be tracked but kept private.

2.5 Assumptions and Dependencies

For the rest of the RASD, the following assumptions are given for granted:

- The GPS location of users and cars is always functioning and accurate, with an uncertainty of ± 1 m;

- All users can access a reliable and stable internet connection;

- The users' mobile devices feature a working GPS;

- All users are always charged the correct amount after the ride;

- The on-board computers always notify the correct charge to the driver during the ride;

- All cars unlock properly upon insertion of the code by the user who reserved them or in case he/she is detected to be in proximity;

- The equipment of the cars always gives a correct reading of the number of passengers, driver included, for the current ride;

- The number of passengers never changes for the duration of the ride;

- If a user reserves and uses a car he/she is the one who drives it and is responsible for the associated trip;

- The maintenance division always fix the reported out-of-order vehicles;

- The maintenance technicians process each request in no more than 1 hour;

2.6 Future Extensions

The system implementation will open the possibility of including new features. For example:

- The possibility of including more categories of vehicles, such as motorcycles and bycicles;

- The possibility of receiving active feedback from the users, mainly concerning car conditions;

- The possibility for everyone, registered and unregistered people, of notifying *PowerEnJoy* of incorrect or unsafe situations involving their vehicles;

- The implementation of a new front-end application to make the management of out-of-order vehicles much more easier.

Section 3

Specific Requirements

3.1 External Interface Requirements

3.1.1 User interface

The user interfaces must be strongly user-friendly, in order to provide an easy and intuitive way to access the system. To achieve this, the following constraints have to be satisfied by the web and mobile applications:

- The first page must always ask the user to login or register to the service;

- The main page must show a map with a search box in order to find all nearby available cars with respect to the specified address;

- The interface must offer the possibility to choose between a set of different languages;

- A toolbar must be visible in every screen except the first page;

- The toolbar must offer a quick and simple overview of the main system functionalities (reserve a car, show personal information area, log out...);

- The user interface must dynamically adapt to the screen size;

- In order to enhance the user experience, the web application and the mobile one must use the same graphical elements and layouts.

With respect to the on-board computer application, the restrictions are the following:

- After the car has been unlocked, the display of the on-board computer must turn on and show a screen thats allows the user to input his/her PIN in order to start the ride;

- Once the ride starts, the screen must clearly show the current charges;

- Similarly, during the ride, the screen must clearly show the nearby charging spots and visually signal the Safe Area boundaries;

- When the car is available, the screen must always be off.

Some mock-ups that give an overall idea of the interface structure of the applications are provided in the next section (3.2). In addition to that, several application-dependent constraints are provided:

- Web application

  All the web pages must abide by the W3C standards to ensure a long-term growth of the application and take the full advantage of HTML5 markup language and CSS.

- Mobile application

  Both the Android and iOS versions must follow the design guidelines provided by the respective platform manufacturers.

- Server application

  The server application must be easily configurable via a XML file and use the APIs provided by the payment handler to interface with its information system.

- On-board computers application

  The application mounted on on-board computers must use the APIs provided by the vehicle manufacturer to interface with the cars.

3.1.2 Hardware interface

The most relevant aspect to highlight, in addition to section 2.1.2, is the ability of the system-to-be to communicate with the car systems via the embedded computer located in the car itself that takes the advantage of the on-board computer application to be developed. The on-board computer can communicate with the mobile broadband thanks to a SIM and provides a GPS module exploitable by the car-side of the application.

3.1.3 Software interface

The server side of the application, that represents the back-end of the system-to-be, requires the following software products:

- Java EE 8 - `http://www.oracle.com/technetwork/java/javaee/overview/index.html`

- MySQL 5.7 - `http://dev.mysql.com/`

The embedded on-board computer located in each car provides a Linux OS that is capable to run C/C++ programs. Moreover the car-side of the application can use specific software libraries provided by the manufacturer to access the car systems the computer is connected with.

3.1.4 Communication interface

The clients communicate with the server via HTTPS protocol by using TCP and default port 443. Moreover a safe and stable connection is also established between the server and the payment handler whenever the user performs a payment. The systems talks to the maintenance technicians via SMTP protocol and simple auto-generated e-mail messages.

3.2 Functional Requirements

3.2.1 Register

Purpose

The user who wants to subscribe to the *PowerEnJoy* service can carry out the registration process through the mobile application or the web one. First of all the user has to fill a registration form with his/her personal information and accept the Terms and Conditions of use otherwise the procedure will be aborted.

As soon as the user submits the data, the system checks all the information and sends a confirmation e-mail to the specified e-mail address including the password to access the service and the personal PIN required to start the car engine.

Scenario 1

Marta would like to register to the *PowerEnJoy* rental service from her computer. She opens the browser and loads the registration page by clicking on

the *"sign up"* button located in the *PowerEnJoy* login page. She fills in the registration form and agrees the service Terms and Conditions. Then Marta submits everything to the system, that checks the personal data. Personal PIN and password are sent to Marta in a confirmation e-mail notifying the successful registration.

Scenario 2

Yuri cannot afford a car and therefore decides to take advantage of the *PowerEnJoy* service. He accesses the login page from his mobile phone and taps the *"sign up"* button since he is not a registered user. Then he fills in the form and accepts the Terms and Conditions of use.

Unfortunately he forgets to fill in his own tax code and the system does not allow him to carry out a successful registration notifying the forgetfulness. Afterwards Yuri completes the procedure by filling the whole form. The system communicates the successful registration and sends him a confirmation e-mail containing PIN and password.

Use-case

The use-case of the registration procedure is analyzed in Table 3.1. A more detailed use-case diagram about the registration procedure is shown in Figure 3.1.

Activity diagram

The activity diagram of the registration procedure is shown in Figure 3.2.

Functional requirements

1. The user must not enter an already registered e-mail address to perform the registration process;

2. The user must provide the following personal information:

   - e-mail address
   - name
   - surname
   - ID card number
   - tax code
   - license ID number

17

- address

- phone number

- credit/prepaid card or $PayPal^{TM}$ account credentials

3. There must not be another subscribed user with the same identity card number, tax code or license ID;

4. If the user does not agree the Terms and Conditions of use, the registration process is canceled;

5. The system must send an e-mail containing PIN and password to the e-mail address specified by the user when he/she clicks on *"Submit"* button and all the required information has been entered;

6. The system allows the user to exit the registration process at any time;

7. The system must allow the user to provide either a valid credit/prepaid card number or $PayPal^{TM}$ account credentials to perform the payments;

8. The PIN provided by the system in the confirmation e-mail is made by only digits and has a length of 6;

9. The password provided by the system in the confirmation e-mail is made by both digits and letters (capital and not) and has a length of 8.

Figure 3.1: Detailed use-case diagram of the registration process.

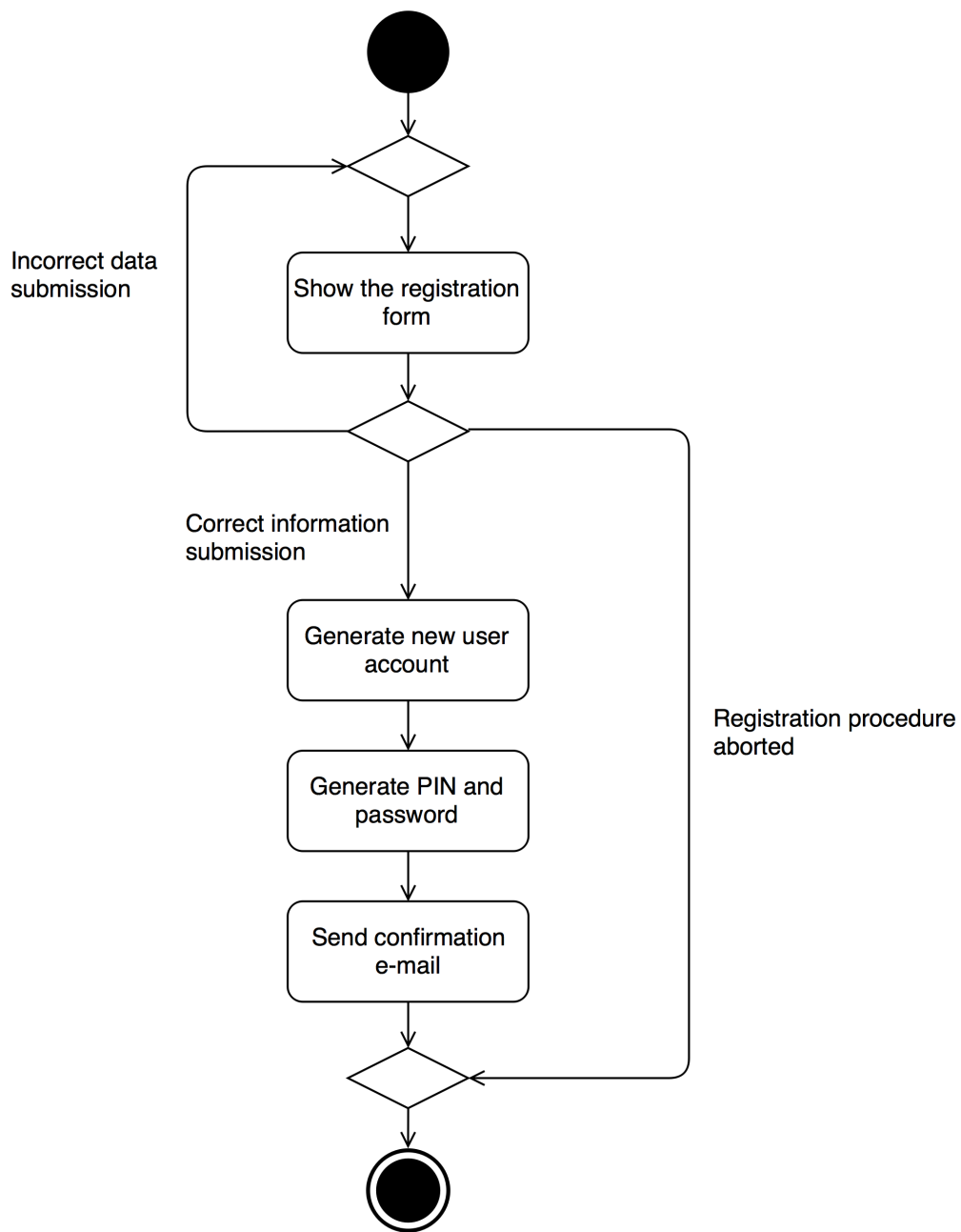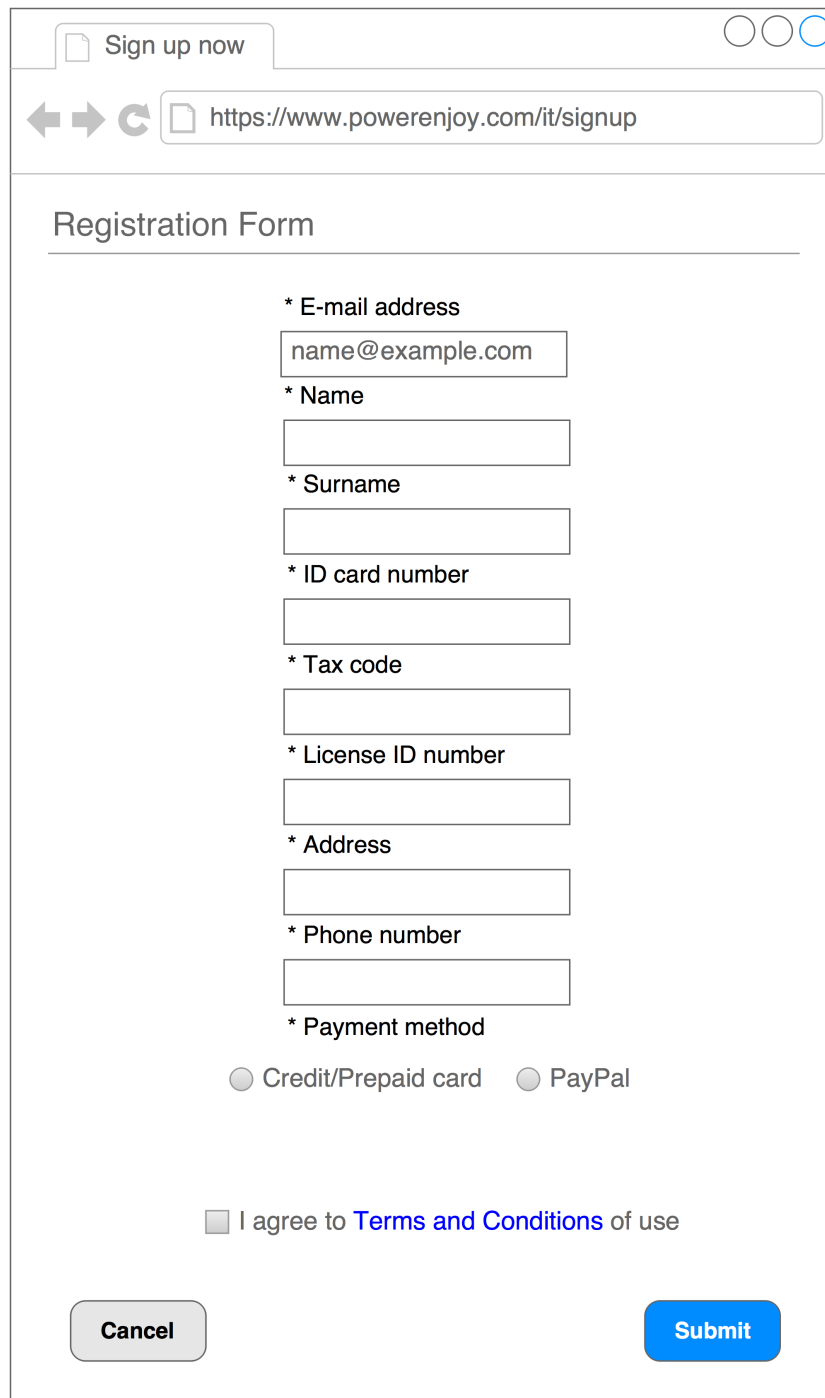|	
Actor	User
Goal	Goal 1
Input Condition	The user wants to subscribe to the service.
Event Flow	1. The user opens the *PowerEnJoy* home page or the mobile application and clicks or taps on the "Sign up" button; 2. The registration form is loaded and the user fills in all the fields with his/her personal information; 3. The user agrees the service Terms and Conditions by checking the corresponding box; 4. The user clicks on *"Submit"* button; 5. The system saves the data in its database and sends a confirmation e-mail containing PIN and password to the new registered user.
Output Condition	The system tells the user he/she is successfully subscribed and lets him/her to log in with the provided credentials.
Exception	When some exceptions occur, the system reloads the registration form and goes back to step 2 of Event Flow notifying the user about the error. This happens when the following requirements are not satisfied: 1, 2, 3. The registration process is aborted when the user decides not to carry it out completely or when requirement 4 generate an exception.

Table 3.1: Register use-case

Figure 3.2: Activity diagram of the registration process from the system point of view.

Figure 3.3: Mockup of the sign up webpage.

3.2.2 Login

Purpose

The main purpose of the login feature is to grant the access of the *PowerEnJoy* service to any registered user. The system requires a valid e-mail address and password to login without generating any error.

In addition to that, the login screen offers the chance to recover a forgotten password. The user clicks on *"forgot password?"*, a new one is sent to his e-mail address immediately and he/she can carry out the login procedure with the new password provided by the system.

Scenario 1

Mike would like to rent a car for having a ride in a beautiful sunny day. In order to do that he needs to access the system by means of his credentials. He opens the *PowerEnJoy* home page and enters his e-mail address and his password. Then Mike clicks on *"login"* and, due to the fact that everything is correct, he obtains the access as a logged user.

Scenario 2

Bill wants to take the advantage of the *PowerEnJoy* service. He opens the home page and he is asked for his e-mail address and password. He is aware of his e-mail address, but he does not remember the password. After some time spent trying to recall his personal password, Bill inputs his e-mail address and decides to click on *"forgot password?"*. Right away the system sends a new password to the specified e-mail address. Bill can now enter the system using the new password.

Use-case

The login use-case is analyzed in Table 3.2.

Activity diagram

The login activity diagram is shown in Figure 3.4.

Sequence diagram

The sequence diagram of the login procedure is shown in Figure 3.5.

Functional requirements

1. The user must be already registered to the system in order to perform a successful login;

2. The user must be aware of his e-mail address and password to successfully obtain the system access;

3. The password provided by the user must correspond to the specified e-mail address;

4. Wrong credentials prevent the user to access the system;

5. The system sends a new password to the specified e-mail address if and only if the specified e-mail address is valid, registered to the system and the user clicks on *"forgot password?"*;

6. After requesting a new password, the system must allow the user login with the new provided password;

7. After three times the entered password is wrong, the system allows a new attempt 30 seconds later;

8. The system does not allow a registered user with a suspended account due to pending payments to log in.

Actor	User
Goal	Goal 1
Input Condition	The user is already registered to the system and wants to login.
Event Flow	1. The user opens the *PowerEnJoy* home page or the mobile application and the system shows the login page; 2. The user enters his/her e-mail address and password; 3. The user clicks on *"login"* button.
Output Condition	The system lets the user to log in if the provided credentials are valid and loads his/her personal home page.
Exception	If the e-mail address provided by the user has never been registered to the system or if the password does not correspond to the entered e-mail, the system notifies the user with an error message. If the user enters wrong credentials three times in a row, the system will prevent any attempt for the following 30 seconds informing the user.

Table 3.2: Login use-case

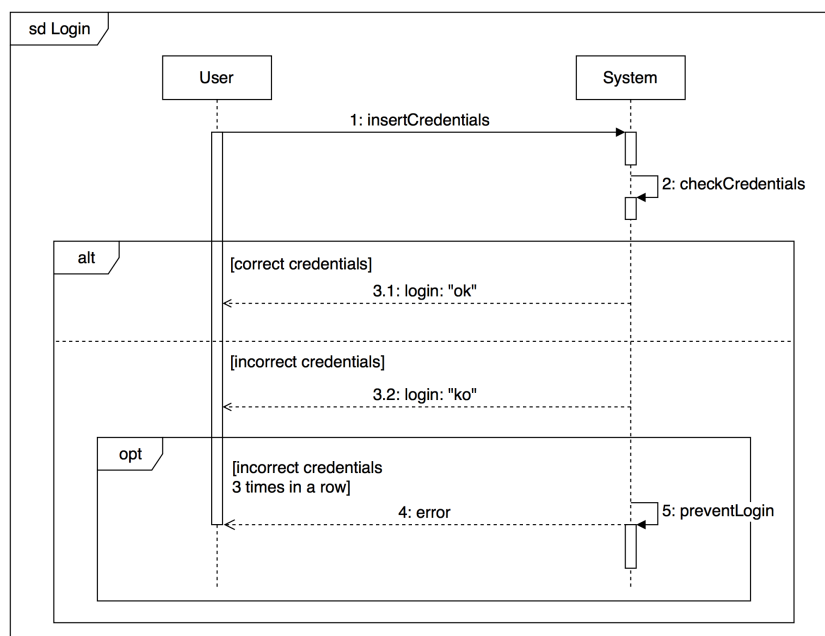**Figure 3.4:** Activity diagram of the login process from the system point of view.

Figure 3.5: Sequence diagram of the login process.

Figure 3.6: Mockup of the login webpage.

Figure 3.7: Mockup of the mobile version of the login screen.

3.2.3 Manage profile

Purpose

Both the mobile application and the web one provide the chance to any registered and logged user to view and update the personal profile information. The system is able to provide also information about the reservations and rides history as well as the respective payments.

Moreover the logged user is allowed to delete his/her account if the service is no longer needed.

Scenario 1

Anna has already carried out the registration process, but she does not like the password the system provided her because it is hard to remember. Therefore she decides to log in from her mobile phone and access the personal information area through the designated toolbar.

She taps *"edit profile"* and the system allows her to edit profile information. Anna enters the old password in the designated field as well as the new one, that must be entered twice to avoid unwanted errors. Then Anna selects *"confirm"* and the system notifies her about the successful update.

Scenario 2

Flavio has recently renewed his identity card. In order to take the advantage of the *PowerEnJoy* service, he must update his profile information. Flavio opens the browser, reaches the *PowerEnJoy* home page and authenticates himself carrying out the login procedure.

Then he access his personal information area and clicks *"edit profile"*. He fills in the gap corresponding to the ID card number and, as soon as Flavio is ready, he clicks on the *"confirm"* button and the system confirms the ID card number has been successfully updated.

Scenario 3

Diana has carried out the subscription to *PowerEnJoy* providing her credit card number. She creates a $PayPal^{TM}$ account and wants to use it as payment method whenever she rents a car. In order to do that, she logs in from her browser and accesses her personal information area.

Diana selects *"edit profile"* and changes the payment method from credit card to $PayPal^{TM}$. The system immediately asks for her $PayPal^{TM}$ credentials which will be used to perform payments automatically. She provides the information and confirms the change. The system informs her that the payment method has been successfully modified.

Scenario 4

Emma wants to delete her *PowerEnJoy* account because she does not need the service anymore. She logs in from her mobile phone and enters her personal information area. She selects *"edit profile"*, taps *"delete account"* and confirms her choice twice. The system removes immediately all Emma's information from the database. From now on she needs a new account in order to access the system again.

Use-case

The overall profile management use-case diagram is shown in Figure 3.8.
The use-case of view profile procedure is analyzed in Table 3.3.

The use-case of modify profile procedure is analyzed in Table 3.4.
The use-case of delete profile procedure is analyzed in Table 3.5.

Functional requirements

1. The system allows logged user to view his/her profile. When the user enters his/her personal information area, the system must display:

   - all the user's personal information that has been entered during the registration process;

   - the rides and reservations history;

   - the payments history.

2. The system allows logged user to modify his/her personal information as needed:

   - the system allows to change any information except e-mail address and personal PIN when the user selects *"edit profile"*;

   - the system accepts passwords with a maximum length of 8;

   - the system asks the user to enter the old password and two times the new one whenever he/she needs to change it;

   - the system allows the user to change the password if and only if the old one has been entered correctly;

   - the system asks for the new password twice when the users wants to change it;

   - the system must not allow the user to change the password if the new one has not been entered twice;

   - the system makes the change permanent as soon as the user clicks the *"confirm"* button;

   - the system allows the user to abort the profile editing at any time.

3. The system allows the user to delete his/her account:

   - the system asks for a confirmation twice whenever the user wants to delete his/her account;

   - the system completely destroys the user's personal information and removes his/her account from the database.
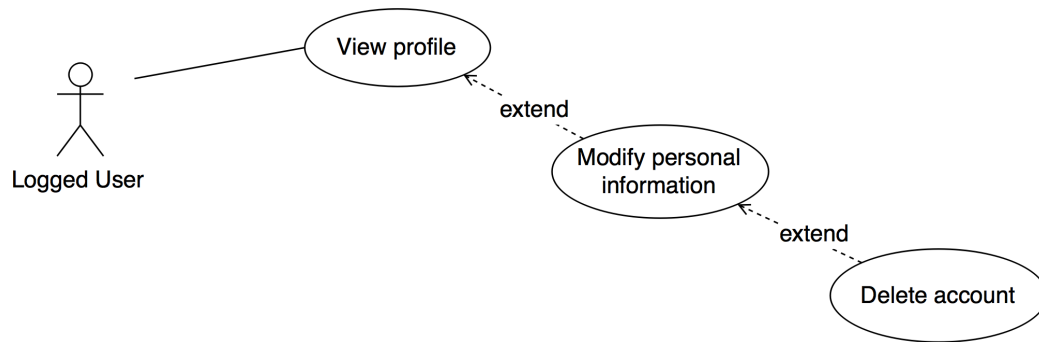
Figure 3.8: Profile management use-case diagram.

Actor	Logged user
Goal	Goal 2
Input Condition	The user is already authenticated and wants his/her profile information to be displayed.
Event Flow	1. The user enters his/her personal information area. 2. The system loads the page.
Output Condition	The system lets the user check his/her profile, the reservations and rides history as well as the completed payments.
Exception	None.

Table 3.3: Profile visualization use-case.

Actor	Logged user
Goal	Goal 2
Input Condition	The user is already authenticated and wants to edit his/her profile information.
Event Flow	1. The user enters his/her personal information area; 2. The user selects *"edit profile"*; 3. The system loads the edit profile page; 4. The user enters the new information; 5. The user confirms the desired changes.
Output Condition	The system updates the user's personal information.
Exception	Every time the user does not satisfy one of the requirements listed as number 2, the system simply ignores the modification and reloads the *"edit profile"* page.

Table 3.4: Modify profile use-case.

Actor	Logged user
Goal	Goal 2
Input Condition	The user is already authenticated and wants to delete completely his/her profile.
Event Flow	1. The user enters his/her personal information area; 2. The user selects *"edit profile"*; 3. The system loads the edit profile page; 4. The user selects *"delete account"*; 5. The system asks to confirm and the user confirms; 6. The system asks again to confirm and the user agrees;
Output Condition	The system deletes the user's account.
Exception	If the user decides not to confirm twice, the deletion process is aborted and the *"edit profile"* page is reloaded.

Table 3.5: Delete profile use-case.

3.2.4 Reserve Car

Purpose

The reserve car feature is a key service of *PowerEnJoy*. Its main purpose is that of allowing users to choose a nearby car from the ones marked as available on the map, hence signaling their intention of reserving it for an hour. The service can be accessed from the home page of both the web and mobile applications, clicking on the corresponding icon on the toolbar.

The reservation screen shows the user only available cars, based on the information provided by the server. The user can either search for an available car using the mobile application or the web application, indicating an address around which he wishes to search for a vehicle. If the user is logged in through the mobile application, he/she can also choose to use his/her GPS

position - as provided by the mobile device - instead of a specific address.

Only one car should be rented at a time, so the system will prevent the user from trying to reserve several cars simultaneously.

Scenario 1

Alice is logged in to the *PowerEnJoy* service with her laptop, because she wants to find a car to go meet a friend downtown. To do so, she clicks on the reservation icon and opens the city map. She inputs her address, to indicate that she wants a car not far from her house. The system proceeds to mark the vehicles within the desired location. Alice finds a car that she thinks is reasonably close and clicks on it. The system prompts her asking: *"Do you really want to reserve this car?"*. She clicks on the *"Confirm"* button, hence forwarding the request of the specific car to the system.

Scenario 2

Bob is in town and has just finished shopping. He wants to go back home, but does not want to catch a crowded subway train during the rush hour. For this reason he has logged in to the *PowerEnJoy* application and taps on the reservation icon. The system opens a map that shows the GPS position - provided by Bob's smartphone - and several icons marking nearby available cars. He chooses a vehicle just around the corner and properly confirms his intention of reserving the car for some time; by doing so, his request is forwarded to the server. A few minutes later, he meets a friend that offers him a ride, and decides to accept. So, he logs in to his account again and opens the reservation function. He taps on the *"Release Reservation"* button and the system processes his request by marking the car as available again.

Scenario 3

Carla is in need of a car after work, and decides to use a *PowerEnJoy* one. She is looking at the reservation screen and decides to reserve a vehicle by selecting it. Since she is not sure that she will be able to reach the vehicle easily, she tries to reserve a second one after a few minutes. At that point, the system shows a visual recap of her current reservation. Carla tries to access the city map, but the system prevents her to do so, unless she cancels her current reservation. So she renounces to the second reservation and the system does not change anything about the active reservation status.

Use-case

A more detailed use-case diagram is provided in order to model the possibility of releasing an existing reservation. The detail of the use-case diagram is shown in Figure 3.9.

The use-case for the standard procedure needed to request a car is analyzed in Table 3.6. The use-case for the procedure needed to release an existing reservation is analyzed in Table 3.7.

Sequence diagram

The diagram for the standard behavior of the system in case of a car reservation is shown in Figure 3.10.

Functional Requirements

1. Only addresses within a distance of 2 km from the Safe Area boundaries must be considered valid;

2. If a user inputs a non-valid address, the system must notify him/her with an error message saying that he/she is too far from the Safe Area to locate a car nearby;

3. The system must only show cars as available if they are within the Safe Area;

4. The system must provide the position of all nearby vehicles in a radius of 2 km precisely for every address that was considered valid;

5. The system must mark as available only cars that are not already reserved or out-of-service;

6. The system must check that only available vehicle can be rented;

7. An available car can be rented by only one registered user at a time;

8. If a user tries to reserve an car, but that car has been reserved by another user during the visualization of the available vehicles, the system must notify it with an error message;

9. The system must allow to reserve cars only users that do not have an active reservation, that is to say a reservation not older than an hour;

10. If a user has an active reservation, the system must show as first screen - upon the selection of the reservation function - the recap of said active reservation, including time of expiration and car location;

11. If a user has an active reservation, the system must allow him/her to renounce to it by clicking on the *"Release reservation"* button and confirming his/her intentions;

12. The system must always notify the user of a successful reservation/recess from a reservation, immediately after the action is saved in memory by the server;

13. The system must allow the user to visualize the available cars without renting any, by canceling the current reservation procedure.
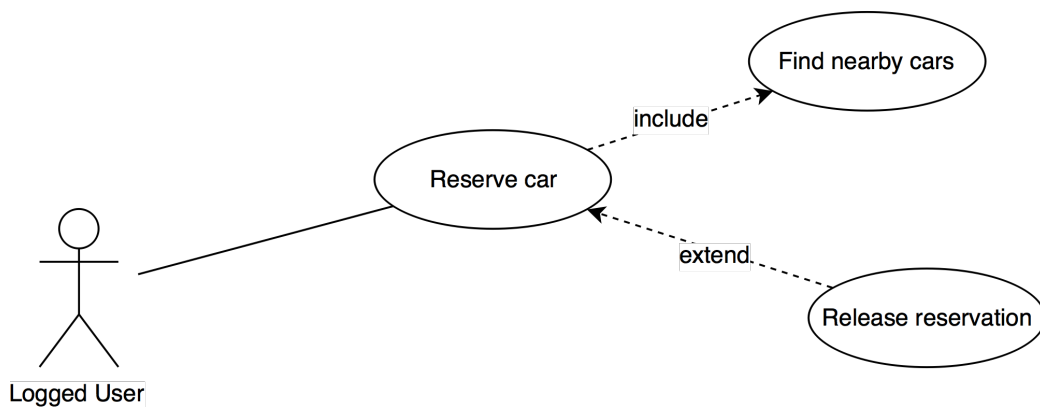

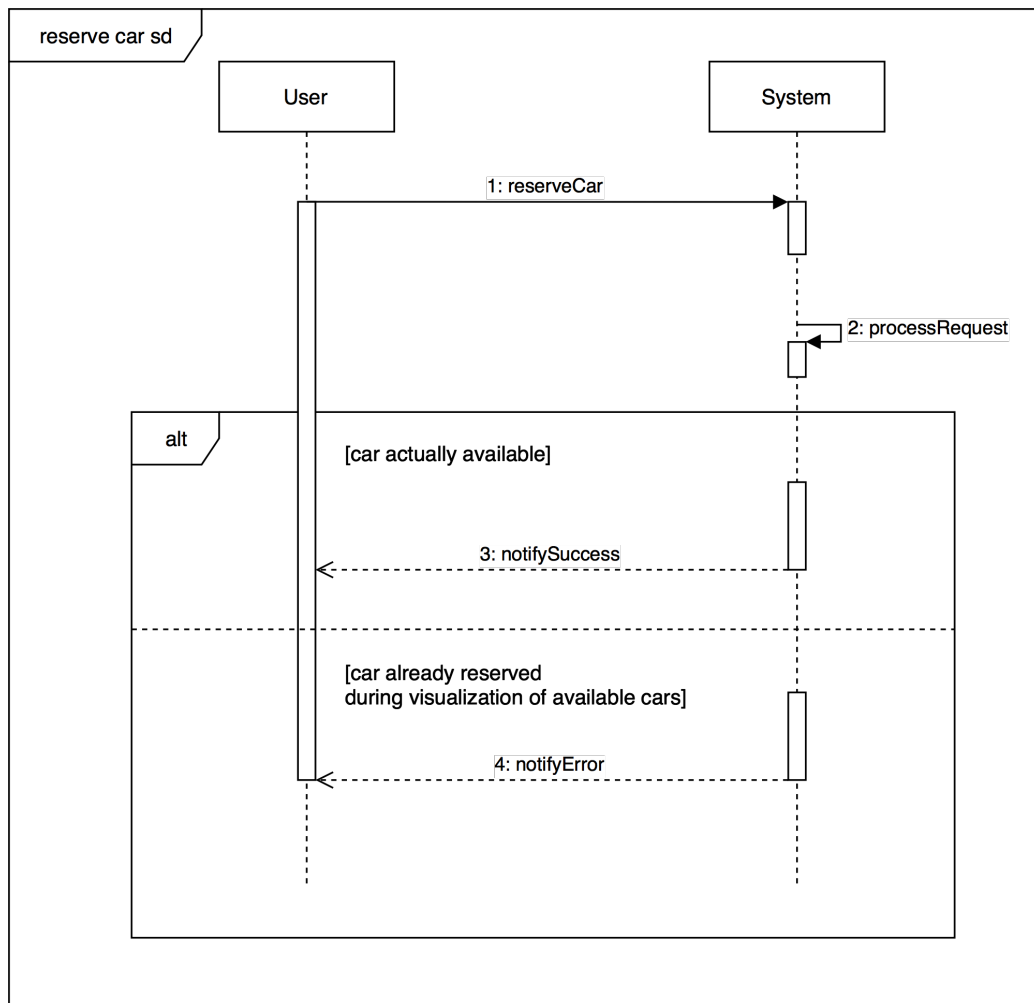**Figure 3.9:** Extension of the use case diagram that includes the "Release reservation" feature

Figure 3.10: Sequence diagram that describes the event flow of a standard reservation request

Actor	User
Goal	Goal 3
Input Condition	The user is logged in and has no active reservations
Event Flow	1. In the main page for the *PowerEnJoy* website or mobile application, the user clicks or taps on the *"Reserve car"* button; 2. The user inputs the desired location: specific address or current GPS location; 3. The system loads a map of the Safe Area, marking every available vehicle near the desired location; 4. The user clicks or taps on the car he/she wants to rent; 5. The user clicks or taps on the *"Confirm"* button when asked if he actually wants to rent the car; 6. The system saves the registration in its database and logs it; 7. The system confirms the reservation to the user, reminding him/her of the fact that the reservation will expire in an hour of time with a small fee if the car is not actually used.
Output Condition	The system notifies the user that the reservation was successful and marks the chosen car as reserved and not available.
Exception	If the user tries to reserve more than one car, the system notifies him/her that this action is forbidden. If the user changes his/her mind and decides not to reserve a car, the event flow stops at the point in which the map was shown. No car will be reserved in this eventuality.

Table 3.6: Reserve car use-case

Actor	User
Goal	Goal 3
Input Condition	The user is logged in and has an active reservation not older than an hour
Event Flow	1. In the main page for the *PowerEnJoy* website or mobile application, the user clicks or taps on the *"Reserve car"* button; 2. Since the user already reserved a car, he is shown the location and time of his current reservation; 3. The user clicks or taps on the *"Release reservation"* button; 4. The system asks confirmation from the user; 5. The user confirms his/her choice of receding from his/her existing reservation; 6. The system modifies the state of the formerly reserved car marking it as available again and notifies the user of the fact that his/her reservation was successfully cancelled.
Output Condition	The system notifies the user that the reservation was successfully cancelled and marks the chosen car as available again. The user can reserve another car, since he/she has no more active reservations
Exception	If the user changes his/her mind and wants to keep the current reservation, the event flow stops at the point in which the current reservation was shown. The reservation will be kept unchanged in this eventuality.

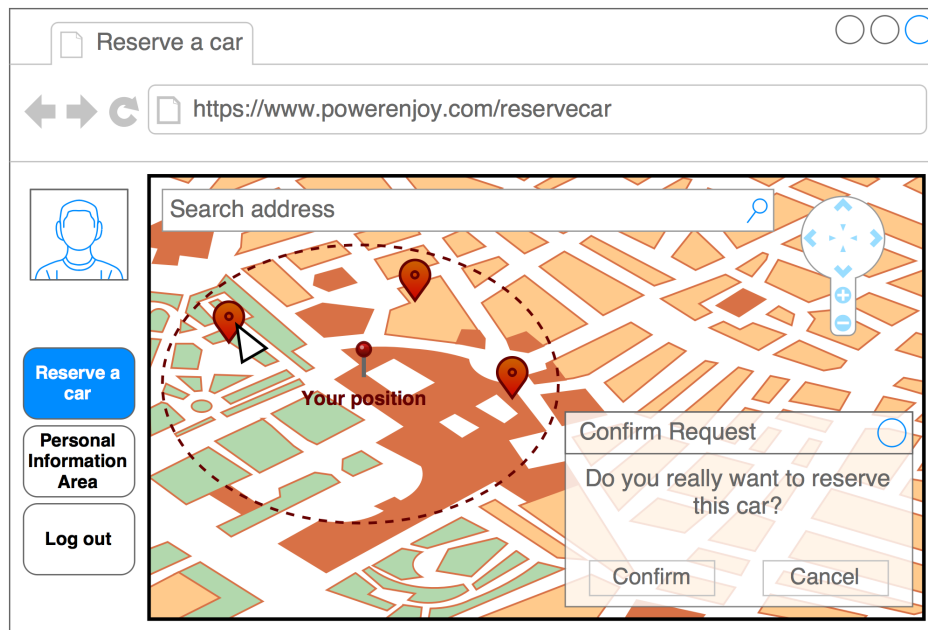**Table 3.7:** Reserve car use-case for the option of releasing a reservation

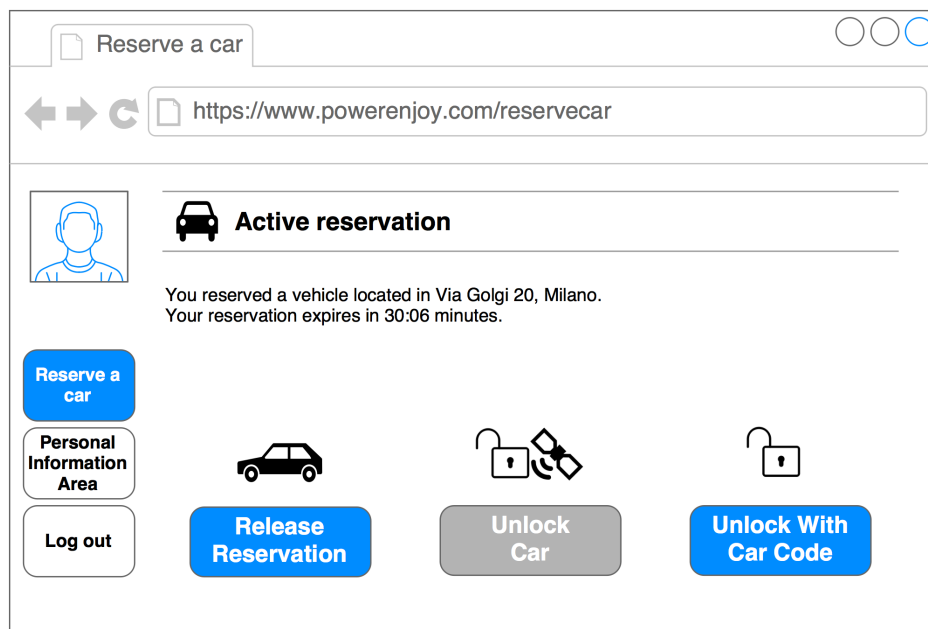**Figure 3.11:** Mockup of the reservation web page

Figure 3.12: Mockup of the reservation web page with an active reservation pending.

3.2.5 Use Car

Purpose

The use car feature is the key feature of the whole system. For the sake of completeness, this requirement is described by detailing each of its sub-parts, as seen in Figure 2.1. The detailed sub-parts are described in the subsections immediately after this one, namely: section 3.2.6, section 3.2.7.

Its purpose is that of allowing users to open, get on and drive the actual vehicle that they rented via the dedicated service. This feature requires the interaction of the user with both his/her mobile device and the on-board computer application.

The system must unlock a car upon being provided with the PIN from the user that reserved it. The on-board computer application must keep track of every important detail of user rides, such as the current charges, the nearby free charging spots and the boundaries of the Safe Area, all while notifying correspondingly the driver.

The on-board computer application will detect the beginning and end of the ride, counting the charges accordingly. Also, the system will be able to detect wheter the car is inside the Safe Area or not: if this is not the case, the user will be notified and the charges will keep increasing.

3.2.6 Unlock Car

Purpose

This sub-part of the "Use Car" functionality focuses on the sequence of sub-functionalities involved in the car unlocking process.

The purpose of the functionality is that of allowing the user to unlock the car he/she reserved based on his/her actual proximity to the vehicle - given the GPS of a mobile device is available - or upon the insertion of a code, found on the windshield of the cars, into the mobile app.

Scenario 1

Damian reserved a *PowerEnJoy* car and is walking towards the vehicle location. When he reaches the car, he takes out his smartphone and opens the *"Reserve Car"* function of the mobile app. The system shows the current reservation: since Damian is close enough to the car and the GPS is active on his mobile device, the *"Unlock car"* button has been activated. He taps on the button and the car unlocks, enabling him to enter it.

Scenario 2

Elsa stayed out later than usual to study for her exams, and decided to rent a *PowerEnJoy* car with her laptop. Her smartphone has low battery, so she decided to turn the GPS off to make the battery last longer. Once she gets to the car, she tries to unlock it by making the system detect her proximity. Since the button does not activate, she remembers that the GPS is off and decides to input the car code into the mobile app; she then looks at the code on the car windshield, opens the right function by tapping on the *"Unlock with car code"* button and inputs the 8-digit code. The system processes the request and, having verified that the user who inserted the code is the same who reserved the car, properly unlock the vehicle and lets Elsa enter it.

Use-case

A detail of the use-case is provided in Table 3.8

Functional requirements

1. Upon every try, the system must check that the user trying to unlock the car is the same that reserved it;

2. The system must guarantee to recognize the proximity of the user if he/she is within a 2 m radius from the car and has the GPS active on a mobile device;

3. The system must always guarantee the possibility for the user to unlock the vehicle he/she reserved by inserting the code on the windshield instead of using the GPS proximity detection system;

4. Upon every code insertion, the system must check that the code is the one associated with the car that the user who inserted the code reserved;

5. The code on the windshield is unique for each car and it has a length of 8 digits.

|	
Actor	User
Goal	Goal 5
Input Condition	The user reserved a car and is in its proximity OR The user reserved a car inserted the car code into the mobile application
Event Flow	1. The user who reserved the vehicle approaches it and opens the mobile application; 2. The user opens the *"Reserve car"* function and the system provides the sum-up of his/her active reservation; 3. The user either: has the GPS active and taps onto the *"Unlock car"* button OR reads and inserts the code from the car windshield into the application, tapping the *"Unlock with car code"* button; 4. The system processes the request, by verifying that the user who tries to unlock the car is the one who reserved it; 5. The system unlocks the vehicle.
Output Condition	The car is unlocked and the user can access it.
Exception	If the user wants to unlock the car but he/she is further than 2 m away from it, the *"Unlock car"* button will not be active. If the user inputs a wrong code when trying to unlock the car, the system will notify that the code is wrong and will not unlock the car, sending the user back to the reservation page on the mobile application.

Table 3.8: Unlock car use-case
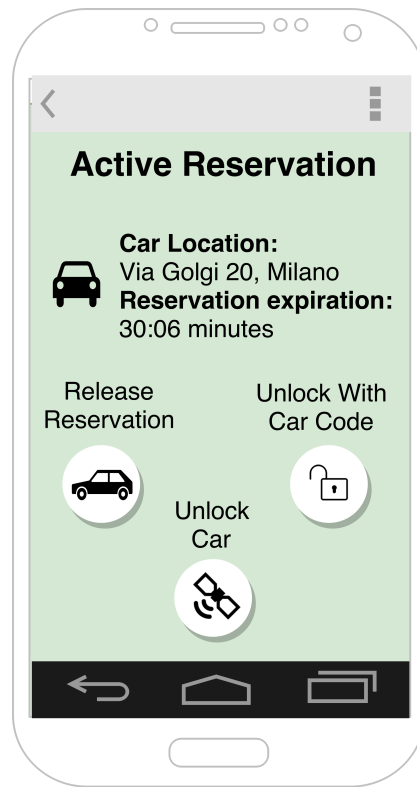
Figure 3.13: Mockup of the mobile unlock feature accessible from the reservation screen.

3.2.7 Start Ride

Purpose

This sub-part of the "Use Car" functionality focuses on the sub-functionalities concerned with the interaction between the user and the on-board application in order to allow the user to turn the engine on.

The system will interact with the user prompting him/her after the car has been unlocked and asking to input his/her PIN. This process will grant access to the car systems, hence allowing the user to start the ride.

Right after the engine has been ignited, the on-board application will also start charging the user, all while showing him/her the current charges on the screen, along with the sat-nav that is also going to signal nearby charging stations and the boundaries of the Safe Area.

Scenario 1

Francis just unlocked the car he reserved using the *PowerEnJoy* service. He sits on the driver's side and closes the door. The screen of the on-board computer is turned on and the system prompts Francis to insert his PIN. He inserts the 6-digit code correctly; the message *"You can now turn the engine on. Once you do so, the system will start charging you based on the duration of the ride."* appears on the screen. Francis turns the engine on and the system shows him his current charges; he then proceeds to drive the car, starting his ride.

Scenario 2

Graham mounted on a *PowerEnJoy* vehicle he just reserved and unlocked. When he inputs the PIN after being prompted by the system, he mistypes one of the 6 digits. The system then shows an error message saying: *"The inserted code is either wrong or does not correspond to the person who reserved this car. Please, retry or use the correct PIN."*. Graham then inserts his code again; this time it is correct, and the system allows him to drive the car.

Use-case

A detail of the use-case is provided in Table 3.9

Functional Requirements

1. The system must check that the code inserted by the user is correct, which is to say that it corresponds to the right user;

2. The system must start charging the user only after he/she turned the engine on;

3. The system must always notify the user about his/her current charges;

Actor	User
Goal	Goal 7
Input Condition	The user unlocked the car
Event Flow	1. The user inputs his/her PIN code on the on-board screen; 2. The system analyzes the code and verifies that the code is correct; 3. The system opens the possibility for the user to ignite the engine and start his/her ride.
Output Condition	The ride started, the system started charging the user
Exception	If the user inputs a wrong PIN, the system will notify it and report an error message; in this case, the car still can not be turned on.

Table 3.9: Start ride use-case

3.2.8 Pay Fee

Purpose

The system must prevent any user's unfair behavior. Whenever the user decides to reserve a car for a ride, he/she has a limited amount of time to reach the car and use it. This is reasonable because during this time nobody can rent that precise car. If the user does not come and use the reserved car, he/she must be penalized and fined by the system.

Moreover, in this case, the previously reserved car becomes available again and the system notify the bad user about the fine he/she has just received. Then the system talks to the payment handler which automatically charges the user.

Scenario 1

Negan is dancing at the disco with his friends when he realizes that it's very late. He opens the *PowerEnJoy* application on his smartphone and checks for an available and nearby car to rent to go home. He finds one and reserve it.

Suddenly he notices a very beautiful girl and decides to speak to her in order to know her better. The clock is ticking and he forgets about his previous reservation. After more than one hour, Negan checks his mobile phone and notices the system has notified him about the expired reservation charging a fine.

Use-case

The pay fee use-case is shown in Table 3.10.

Functional requirements

1. The system must not allow a car to be reserved for more than 1 hour;

2. The system must notify the user about the expired reservation;

3. The system automatically charges the user in case of reservation expiration;

4. If the reserved car is not picked-up by the user within 1 hour from the reservation, the system tags the car available again;

Actor	Bad user, payment handler
Goal	Goal 4
Input Condition	The user has reserved a car and he/she does not pick it up within 1 hour from the reservation
Event Flow	1. The system establish a connection to the payment handler; 2. The system asks the payment handler to charge the user a fee of 1 euro;
Output Condition	The system notifies the user about the fine and tags the reserved car available again.
Exception	None.

Table 3.10: Pay fee use-case

3.3 Performance Requirements

The following performance requirements must be satisfied by the system-to-be in order to guarantee the best user experience:

- The system must not have an upper-bound as far as the total number of registered users is concerned;

- The system must handle at least 1000 active reservations at the same time;

- The system must simultaneously support at least 1000 logged users who want to take the advantage of the service performing any kind of operation;

- 90% of car unlock requests shall be processed in less than 1 s;

- 100% of car unlock requests shall be processed in less than 2 s;

- 100% of any other kind of requests shall be processed in less than 5 s.

  Since the performance is essentially bound by the internet connection, the above timing constraints are to be intended from when the system receives the request to the dispatch of the reply.

3.4 Software System Attributes

Reliability

The reliability of the system depends on several factors. First of all it is related to the reliability of the servers that run the back-end application. In addition to that, the reliability of the on-board computers must be also taken into account. Therefore the probability of failure of the devices mentioned above heavily affects the overall system reliability, that can be evaluated as *Reliability = 1 − Probability of failure.*

Availability

The system must offer an availability of 99%. The remaining 1% shall take into account the time spent for ordinary maintenance sessions.

Security

- All the connections established between users and server must use the HTTPS protocol;

- All the communications between the server and the payment handler shall be encrypted;

- The users' passwords are stored in the database using a proper hashing mechanism;

- The system keeps track of all the login attempts storing the corresponding IP addresses.

Maintainability

The code of the whole software-to-be-developed must be thoroughly documented in order to let future developers easily know how the system works and how it has been designed. A version control system must be used to manage and organize all the different code revisions.

Portability

The server-side of the application shall be written in Java. Therefore it can be executed on any platform that supports the Java Virtual Machine and runs JEE. The web application must support the latest version of the main browsers like IE, Google Chrome and Firefox. The mobile application must be developed for both Android and iOS architectures. The car-side of the application is designed to run on the existing on-board computers.

Appendix A

Appendix

A.1 Software and tools used

- LaTeX, used as typesetting system to build this document.

- draw.io - `https://www.draw.io` - used to draw diagrams and mock-ups.

- GitHub - `https://github.com` - used to manage the different versions of the document and to make the distributed work much easier.

- GitHub Desktop, the GitHub official application that offers a seamless way to contribute to projects.

A.2 Hours of work

Bibliography

[1] ISO/IEC/IEEE 29148:2011 *Systems and software engineering - Life cycle processes - Requirements engineering*

[2] IEEE 830:1998 *Recommended Practice for Software Requirements Specifications*

[3] AA 2016/2017 Software Engineering 2 - *Project goal, schedule and rules*