Software Engineering 2: PowerEnJoy

Requirements Analysis and Specification Document (RASD)

Version 1.0

Politecnico di Milano, A.A. 2016/2017

Agosti Isabella, 874835
Cattivelli Carolina, 879359

November 13, 2016

Contents

1 INTRODUCTION

1.1 Description of the given problem

We are to develop a digital management system for a car-sharing service called *PowerEnJoy*, that exclusively employs electric cars.
The users will be able to interact with the system through a web or mobile application. The system will thus keep track of the users' position monitoring their GPS signal (if active), so as to be able to show them the locations of the available cars within a certain distance. In alternative, users will also be able to find all the available cars in a certain distance from an address of their choice.
In order to benefit from the service, users need to register to the system by providing their credentials and payment information. Then they receive back a password via email that can be used to access the system.

1.2 Goals

Here are our application's goals:

- [G1] Allow users to register to the system filling out a form.
- [G2] Allow registered users to access the system by entering their username and password.
- [G3] Allow registered users to manage their personal information.
- [G4] Allow registered users to find the locations of all available cars within a certain distance from their current location or from a specified address.
- [G5] Allow registered users to reserve a single car among the available ones in a certain geographical region for up to one hour before they pick it up.
- [G6] Allow registered users to tell the system they are nearby when they reach a car they previously reserved.
- [G7] Allow registered users to see on a screen the amount of money they are being charged for while they are driving.
- [G8] Allow registered users to see on a screen a map showing all the safe areas they can park in.
- [G9] Allow registered users to see on a screen the discount percentage (if any) applied on their bill once the ride has ended.
- [G10] Allow registered users to cancel a reservation paying a fee of 1 EUR.
- [G11] Allow registered users to benefit from a discount percentage in certain cases.
- [G12] Allow registered users to report an issue when they realize a car they reserved is somehow broken.

1.3 Glossary

- **User**: a person that interacts with the PowerEnJoy mobile or web application to register into the system.
- **Registered user**: a person already registered into the system that interacts with the PowerEnJoy mobile or web application in various ways. A registered user is identified by the following information:
  - Name.
  - Surname.
  - Email.
  - Username.
  - Password.
  - Card number.
  - Card code.
  - Driver license code.
  - Address.
- **Car-sharing service**: model of car rental where people rent cars for short periods of time, often by the hour.
- **Electric car**: automobile that is propelled by one or more electric motors, using electrical energy stored in rechargeable batteries. It is identified by:
  - Location.
  - State: reserved/available, blocked/unlocked, ignited/off, plugged/unplugged.
  - Screen.
  - Weight sensors.
  - Doors sensors: to define if they are open or closed.
  - Battery level.
- **Registration**: the act or process of filling out an online form providing credentials and payment information.
- **Log-in**: process by which a user gains access to the system by identifying and authenticating himself/herself. The user credentials are some form of "username" and a matching "password".

- **Reservation**: arrangement through which a registered user holds a car for his use at a later time. It is identified by:
  - Begin time.
  - End time.
  - Corresponding reserved car.
  - User who made the reservation.
- **Available car**: car that has not been chosen in any user reservation.
- **Reserved car**: car that has been chosen by a user in a reservation.
- **Location/address**: position of a car or a user, identified by longitude and latitude.
- **Geographical region**: circular region whose center is the user's position and whose radius is calculated as a certain distance from the center. It shows all the available cars and their position on the map.
- **Safe area**: area whose position is predefined by the management system. Safe areas are the only ones in which a user is allowed to park a car.
- **Special safe area**: special type of safe area where a car can be recharged.
- **Ride**: it starts when the user ignites the car engine and ends when the car is parked in a safe area (or special safe area) and the user exits the car. It is identified by:
  - Begin time.
  - End time.
  - Used car.
  - User that is driving the car.
  - Total price the user has to pay.
- **Discount percentage**: discount applied on the user's last ride only in certain circumstances.
- **Low battery**: the car's battery level is considered "low" when less than 20%.

- **Discount1**: a 10% discount is applied on the ride if the user takes at least two other passengers onto the car.
- **Discount2**: a 20% discount is applied on the ride if a car is left with no more than 50% of the battery empty.
- **Discount3**: a 30% discount is applied on the ride if a car is left at a special parking area where it can be recharged and the user takes care of plugging it into the power grid.
- **Overcharge**: a 30% overcharge is applied on the ride if a car is left at more than 3 KM from the nearest power grid station or with more than 80% of the battery empty.

1.4 Domain assumptions

We suppose that these properties hold in the analyzed world:

- All GPSs always give the right position.
- Addresses entered by the users always exist.
- Cars GPS cannot be switched off.
- All PowerEnJoy cars are registered into the system.
- Cars plates are unique.
- All car doors are closed during the ride.
- Users own a smartphone with Internet connection.
- Users can associate to their profile only prepaid and credit cards.
- The information provided by the user at the moment of his/her registration is valid.
- If a car is no longer associated with the PowerEnJoy service its information is deleted from the database.
- All PowerEnJoy cars are in order.
- The system, once a ride ends, is able to identify if discounts or overcharges are to be applied on the total price.
- The system charges a registered user for a 1EUR fee if he/she cancels a reservation.
- The system charges a registered user for thirty cents for each minute he/her spends driving the PowerEnJoy car.
- The system charges a user for a 1EUR fee if he/she does not pick-up the car within one hour from the reservation.
- PowerEnJoy cars are real.
- If a registered user leaves the car in a non-safe area, he/she continues paying for the reservation until the system notifies him/her via email after ten minutes. The car is no longer reserved and the registered user must pay a fee for abandoning the car.

- There isn't any already existent PowerEnJoy system.
- The car monitors are touch-screen.
- If a car is parked in a special safe area, it is always plugged into the power grid.

1.5 Constraints

1.5.1 Regulatory policies

Upon registration the user must consent to the processing of his/her personal data (name, surname, position...). The company is so committed to handle each user's sensitive data in accordance with his/her privacy.

1.5.2 Hardware limitations

- Mobile application:
  - User:
    - 3G connection.
    - Space for app package.
  - Registered user:
    - 3G connection.
    - Space for app package.
    - GPS, in order to benefit from the option "Current location".
- Web application:
  - Flash Player 11.2 or later.
  - Modern browser with AJAX support.
  - Ethernet or Wi-Fi connection.

1.5.3 Interfaces with other applications

- Interface with email providers such as Gmail, Hotmail, Libero, Yahoo and so on.
- One interface for each platform (iOS, Android, Windows Phone...).

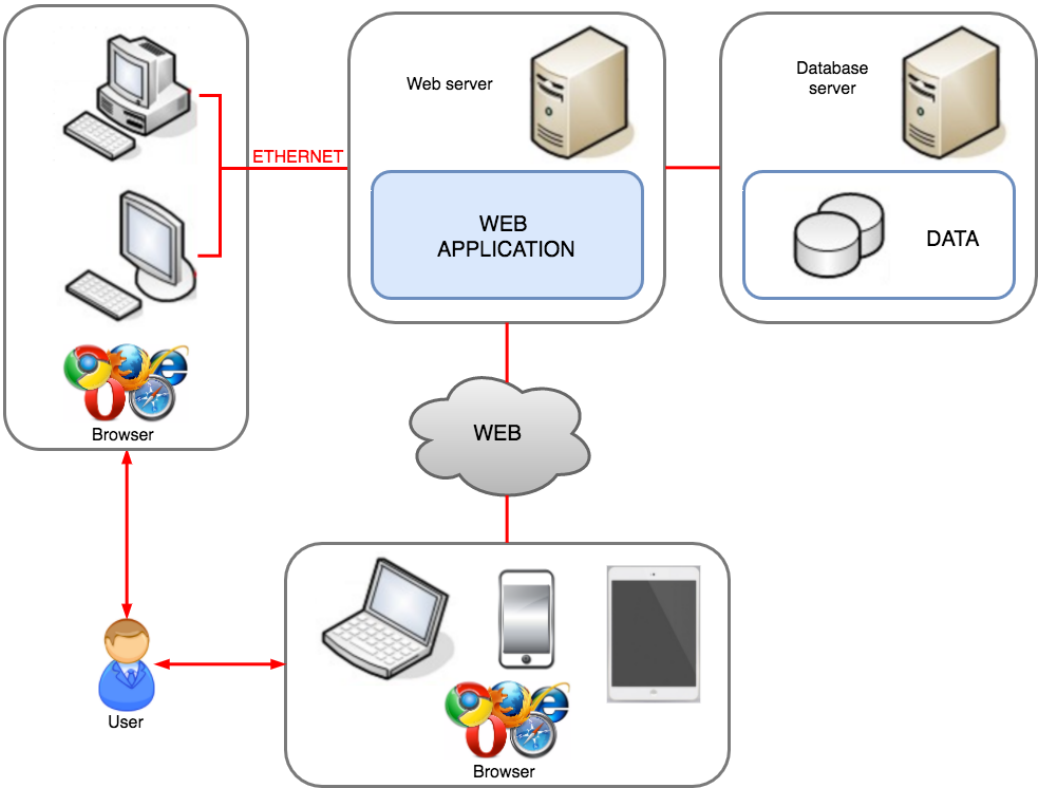
1.5.4 Parallel operation

The application supports parallel operations from different users (registered or not).

1.6 Proposed system

The system will be presented both as a web and a mobile application with a client-server architecture, based on the MVC pattern. It will not be integrated with an already existing system.

The server will generate web pages and a database system will be used to store user's information. On the other side, clients will interact with the application (mobile or web) through a Graphical User Interface.



1.7 Identifying stakeholders

- Primary:
  - **Users** (both registered and not registered), since they are the ones that benefit the most from using the application and the car-sharing service.
- Secondary:
  - **Other passengers**, since they benefit from the service via someone else.
- Tertiary:
  - **Electric energy providers**, since they are affected by the service's introduction, but they do not directly use it.
  - **Electric cars retailers**, since they are affected by the service's introduction, but they do not directly use it.
  - **The city of Milan**, since it is influenced by the service's adoption.

1.8 Reference documents

- Specifications document: Assignments AA 2016-2017.pdf
- RASD sample from Oct. 20 lecture.pdf

2 ACTORS

The actors in our system are:

- **User**: he/she can use the application only to register to the system by filling out a form.
- **Registered user**: he/she already owns a password (that he/she received by email at the moment of the registration) and can log into the system with it. After doing this, he/she can perform various actions:
  - View his/her profile and manage his/her personal information.
  - View current promotions.
  - Reserve a car inserting a specific address or letting the system track his/her position with the GPS.
  - Cancel a reservation.
  - Report an issue after unlocking the car.

3 REQUIREMENTS

3.1 Functional requirements

Assuming that the domain assumptions listed in paragraph [1.4] hold, and in order to fulfill the goals listed in paragraph [1.2], the following requirements can be derived.

Requirements are grouped under each goal from which they are derived.

- [G1] Allow users to register to the system by filling out a form.
  - [R1] The system will provide a registration form.
  - [R2] The system will allow not registered users to access only the homepage and the registration page.
  - [R3] Upon registration, the system will check the uniqueness of the chosen username.
- [G2] Allow registered users to access the system by entering their username and password.
  - [R1] Registered users must have previously filled out the registration form.
  - [R2] Registered users must have previously received a password via email from the system.
  - [R3] The system will provide a login functionality.
  - [R4] The system will verify the existence of the username-password combination in the database.
- [G3] Allow registered users to manage their personal information.
  - [R1] Registered users must be logged into the system.
  - [R2] The system will provide a personal page for each registered user to manage.
  - [R3] The system will allow registered users to save the personal information they update.
- [G4] Allow registered users to find the locations of all available cars within a certain distance from their current location or from a specified address.
  - [R1] Registered users must be logged into the system.

  - [R2] Registered user must have chosen the functionality "make a reservation".
  - [R3] Registered user must have provided a valid address (within the city of Milan), corresponding either to their current location or to a specified address of their choice.
  - [R4] The system will provide a map showing the location of all the available cars.

- [G5] Allow registered users to reserve a single car among the available ones in a certain geographical region for up to one hour before they pick it up.
  - [R1] Registered users must be logged into the system.
  - [R2] Registered users must have chosen the functionality "Make a reservation".
  - [R3] Registered users must have selected a single car among the available ones showed on the map.
  - [R4] The system must provide a timer that will be activated at the moment of the reservation.

- [G6] Allow registered users to tell the system they are nearby when they reach a car they previously reserved.
  - [R1] Registered users must be logged into the system.
  - [R2] Registered users must have selected a single car among the available ones showed on the map.
  - [R3] The system will provide a mobile functionality that allows the users to unlock the reserved car once they are nearby.
  - [R4] The system will unlock the car once the user is nearby.

- [G7] Allow registered users to see on a screen the amount of money they are being charged for while they are driving.
  - [R1] Each car must have a screen.
  - [R2] The system will provide a functionality that updates the amount of money the user will be charged of every minute that passes.

- [G8] Allow registered users to see on a screen a map showing all the safe areas they can park in.

  - [R1] Each car must have a screen.
  - [R2] The car screen will provide an interactive and updated map showing all the safe areas.
  - [R3] The system will provide a functionality that detects the safe areas.

- [G9] Allow registered users to see on a screen the discount/overcharge percentage (if any) applied on their bill once the ride has ended.
  - [R1] Each car must have a screen.
  - [R2] The system will be able to realize when a user parks into a safe area.
  - [R3] The system will provide a functionality that calculates the ride price after applying the discount percentage/overcharge.

- [G10] Allow registered users to cancel a reservation paying a fee of 1 EUR.
  - [R1] Registered users must be logged into the system.
  - [R2] Registered users must have previously reserved a car and not yet unlocked it.
  - [R3] The system will provide a functionality that allows registered users to cancel their current reservation.

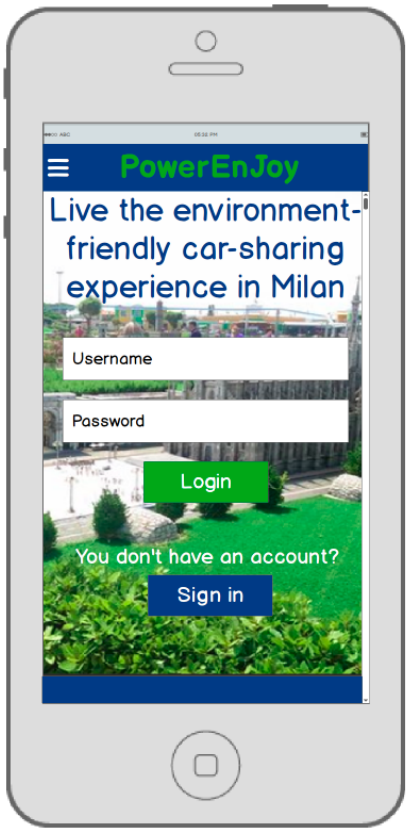
- [G11] Allow registered users to benefit from a discount/pay an overcharge percentage in certain cases.
  - [R1] The system will provide a functionality that calculates the ride price after applying the discount/overcharge percentage.
  - [R2] The system will recognize the cases in which the user is supposed to benefit from a discount/pay an overcharge percentage.
  - [R3] Registered user must have parked in a safe area and turned off the car ignition.

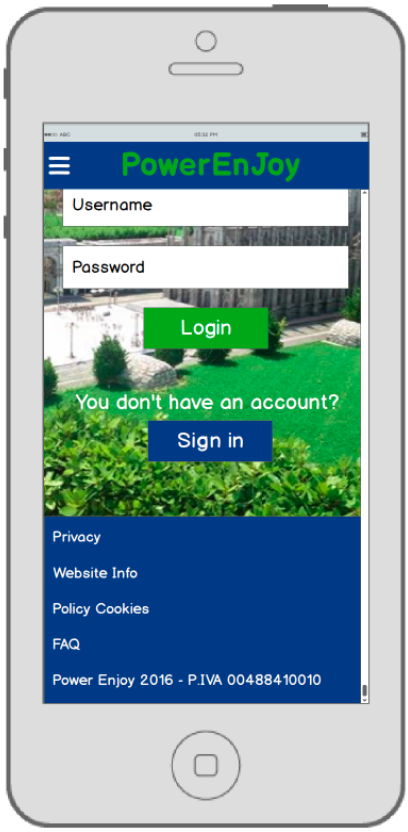
- [G12] Allow registered users to report an issue when they realize a car they reserved is somehow broken.
  - [R1] Registered users must be logged into the system.
  - [R2] Registered users must have previously reserved a car.
  - [R3] Registered users must have unlocked the car.
  - [R4] The system will provide a mobile functionality that allows registered users to report an issue.

3.2 Nonfunctional requirements

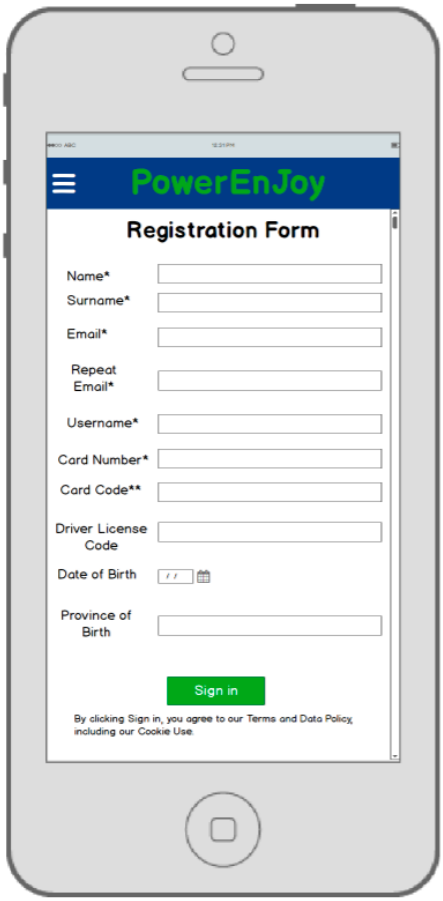
3.2.1 Mobile interface

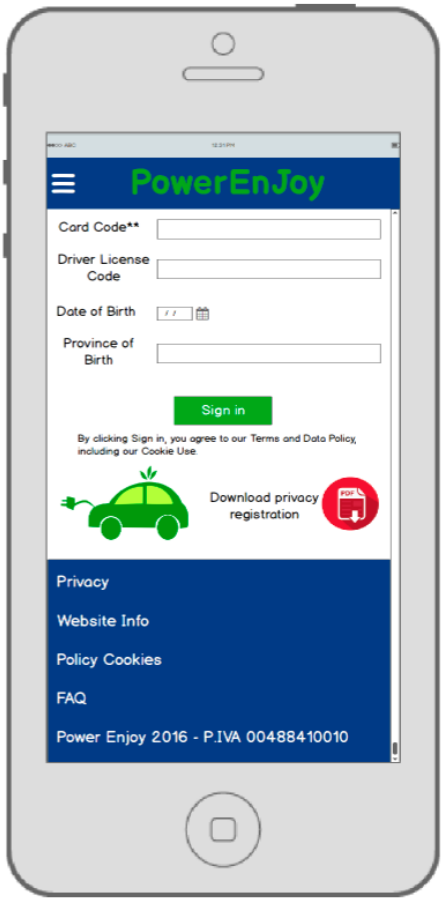
- Homepage:



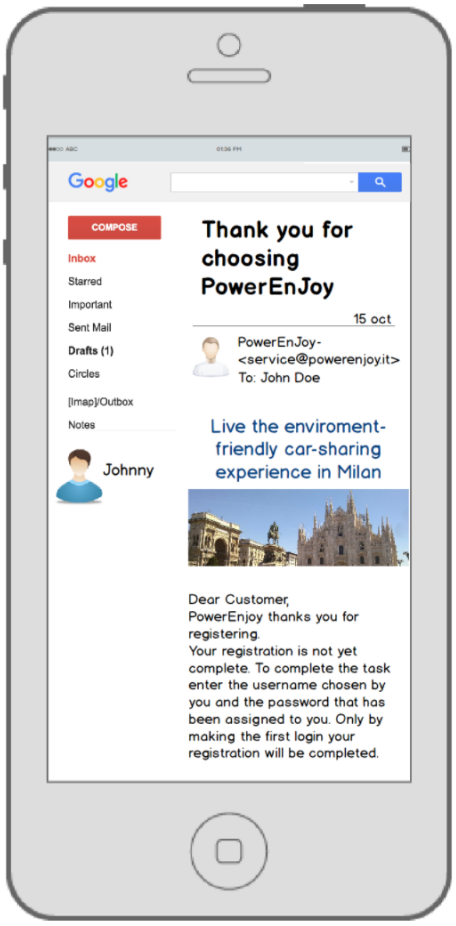


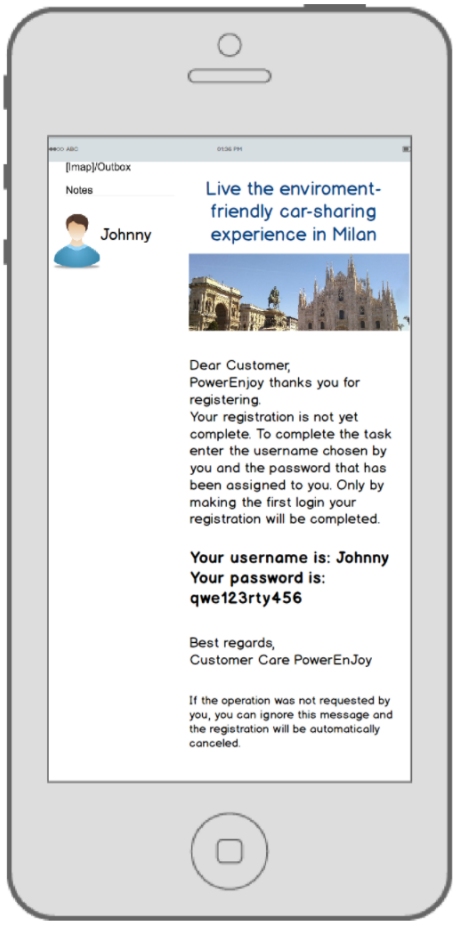
- Registration:



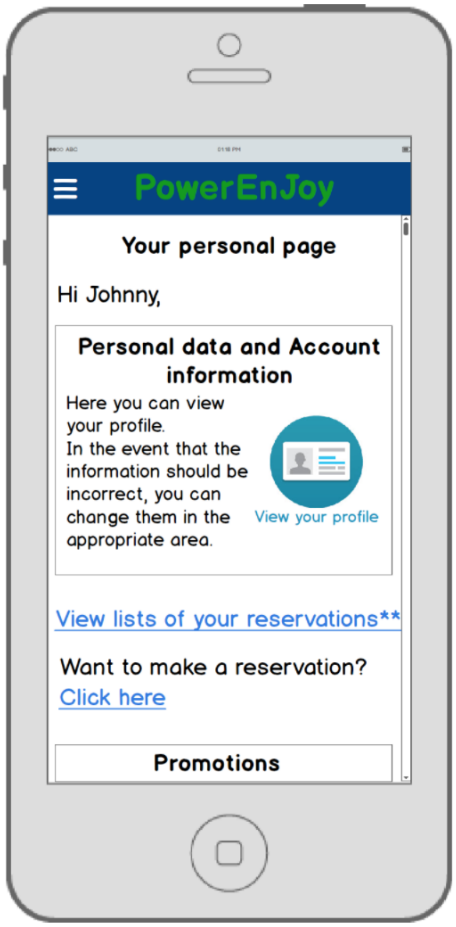


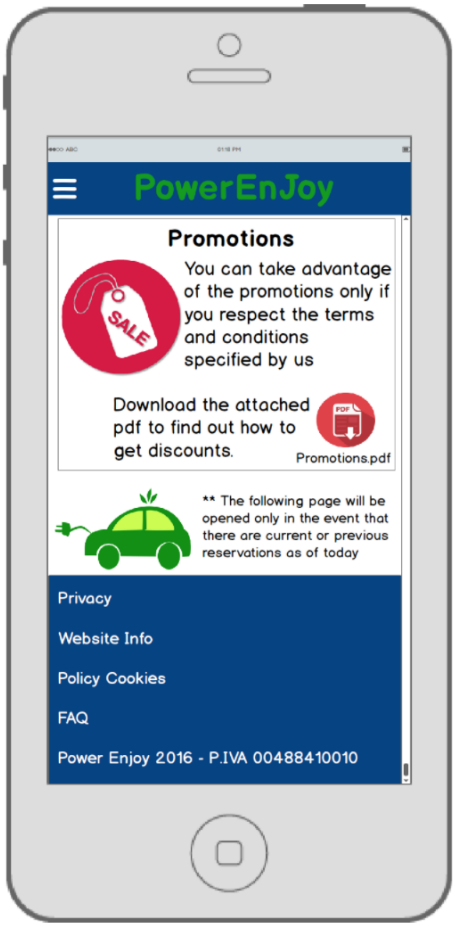
- Receiving the password:



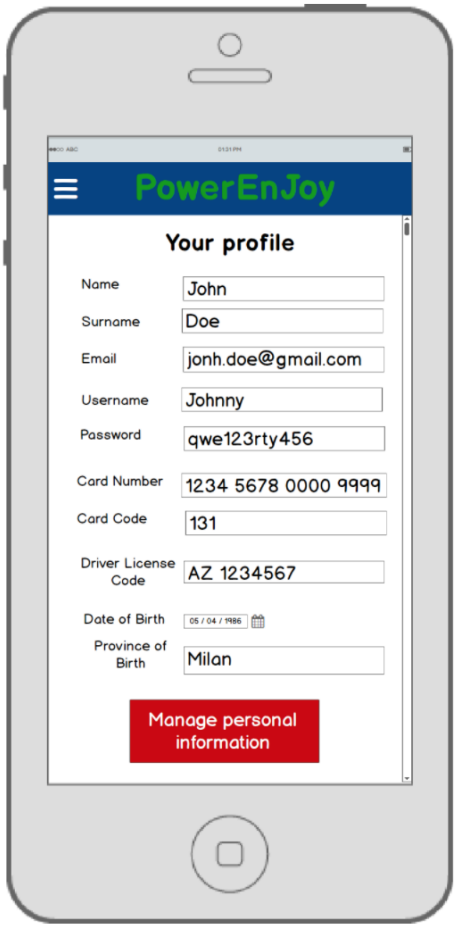


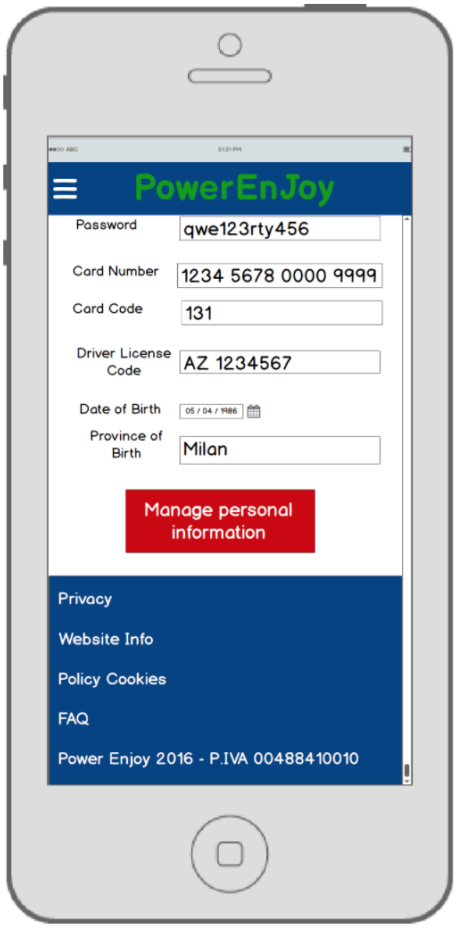
- Personal page:



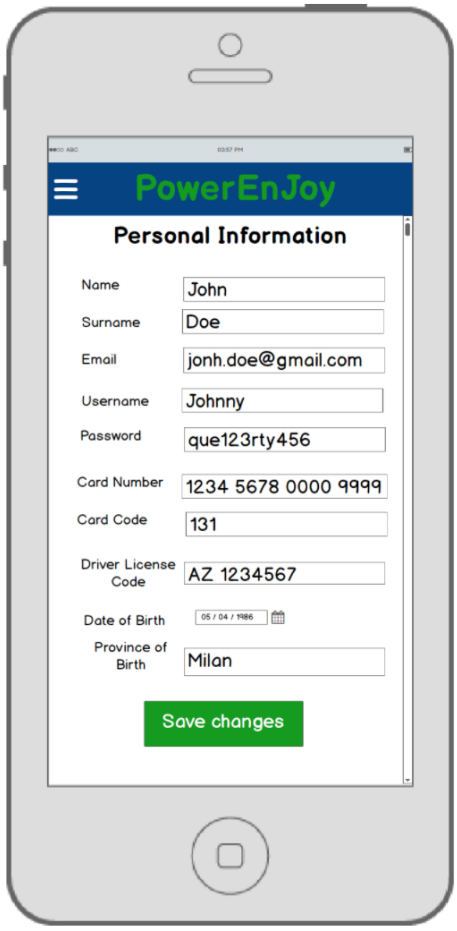


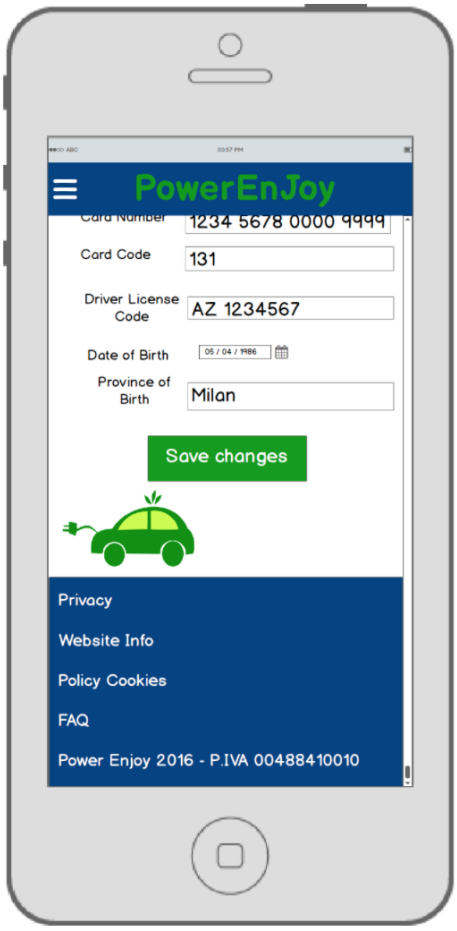
- View profile:



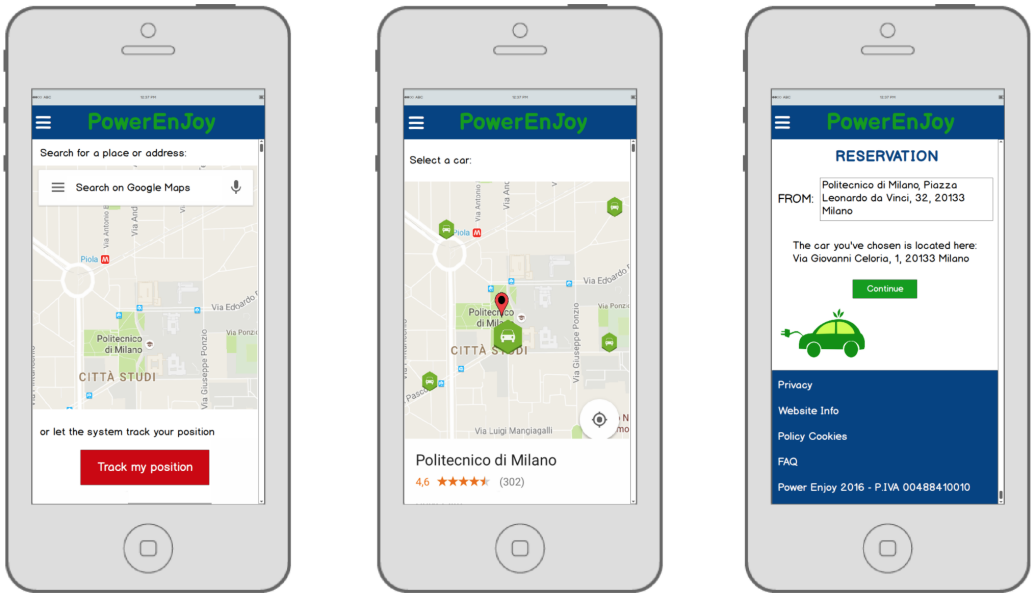


- Managing personal information:

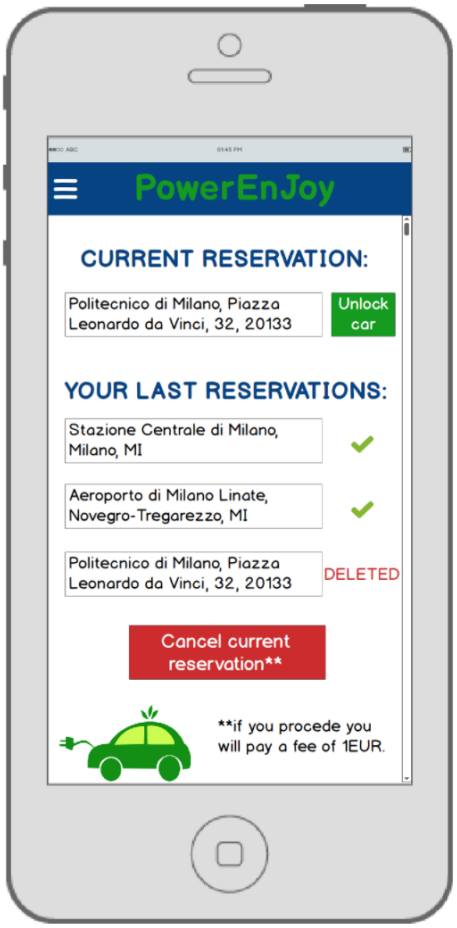


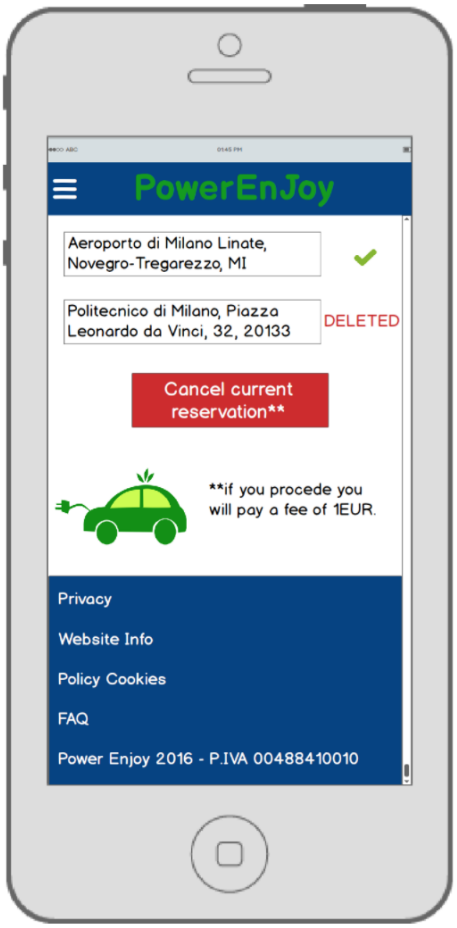


- Making a reservation:

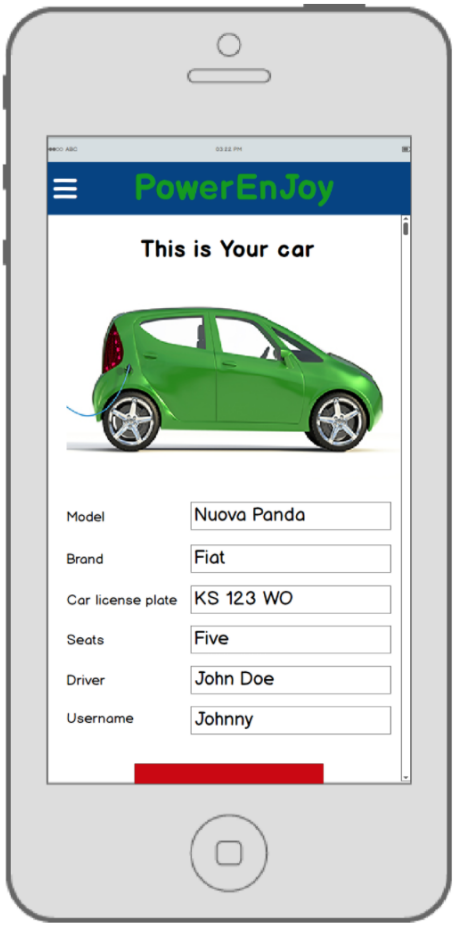


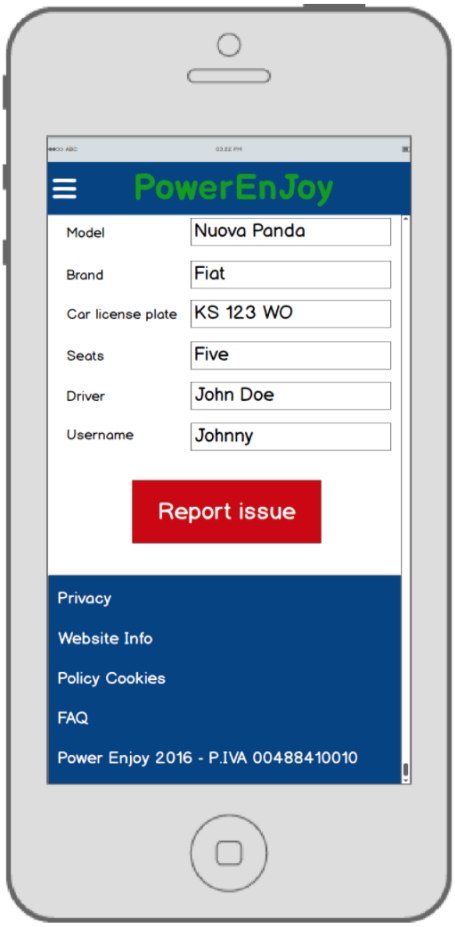
- Deleting a reservation:





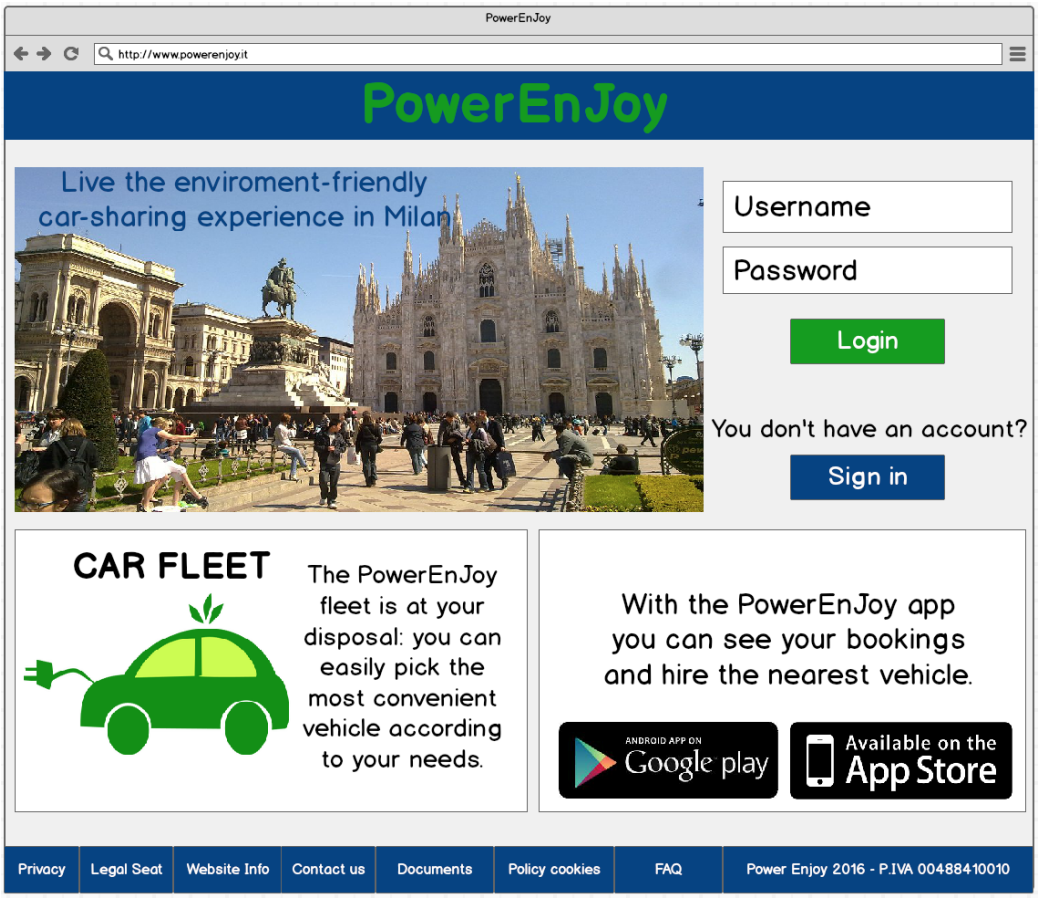
- Reporting an issue:



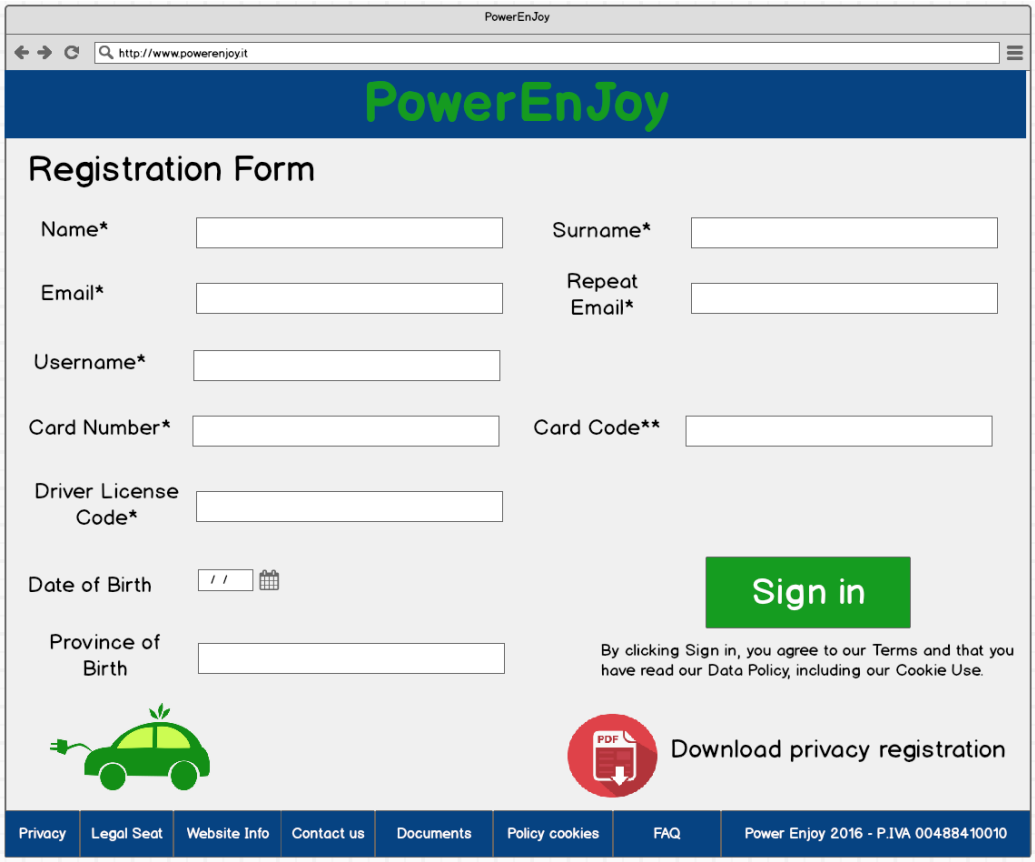


3.2.2 Web interface

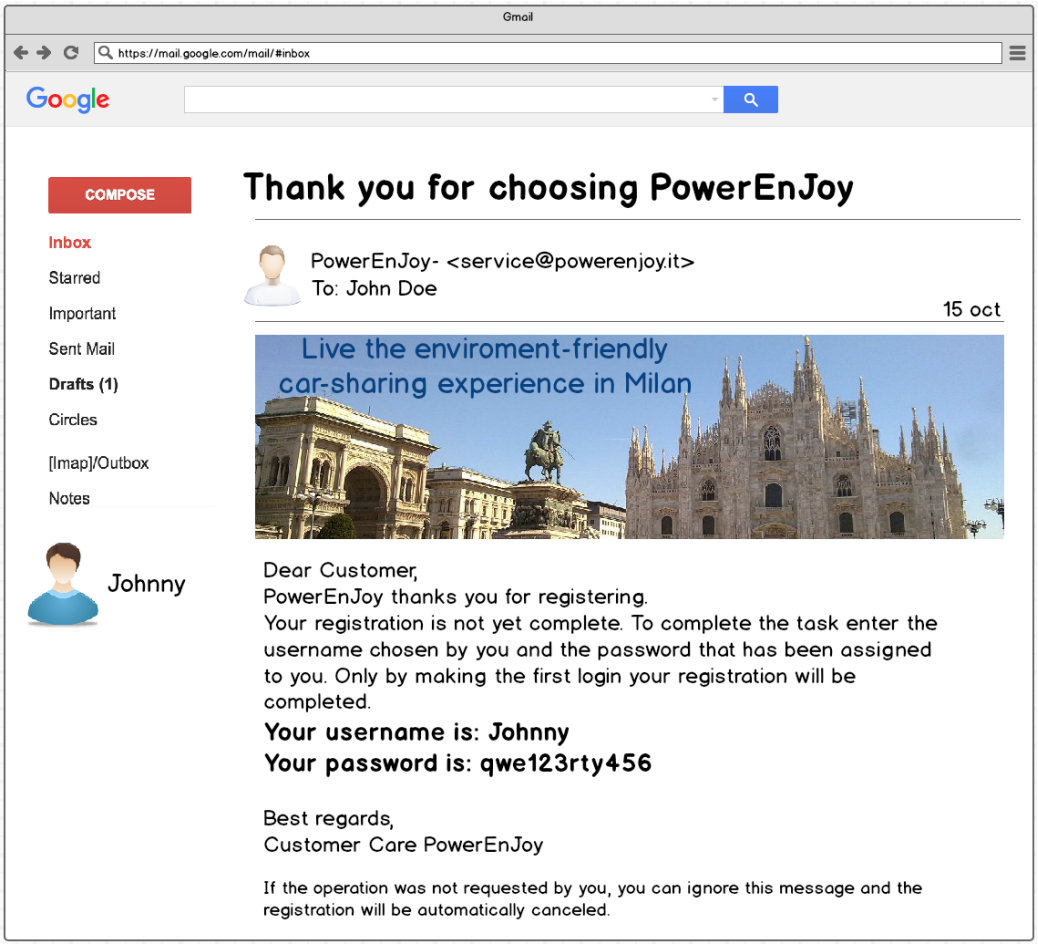
- Homepage:



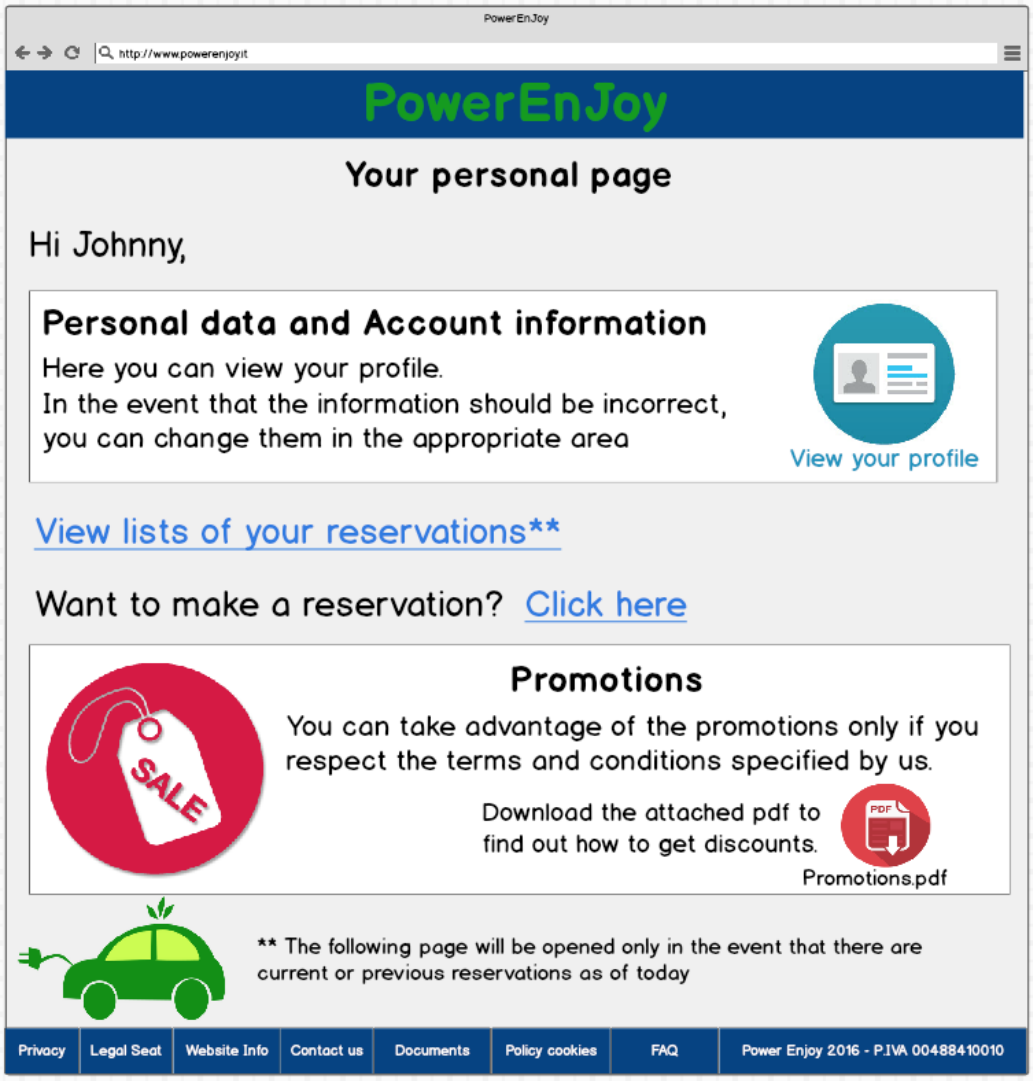
- Registration:



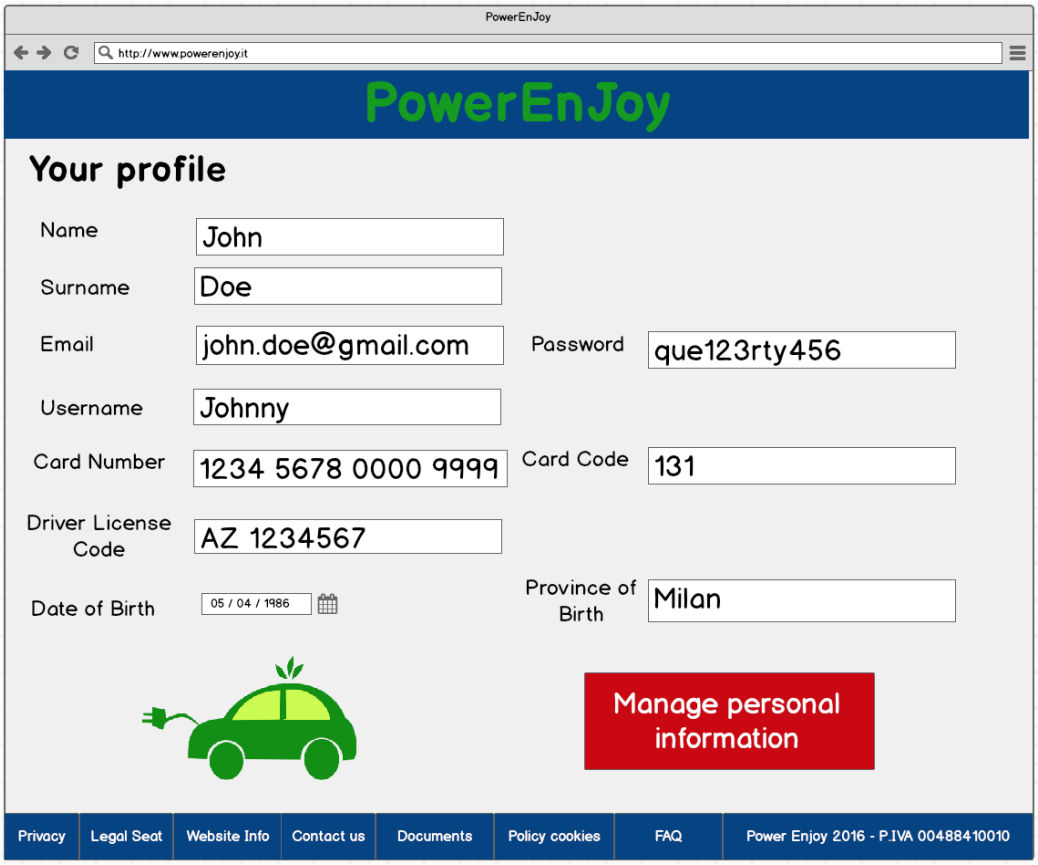
- Receiving the password:



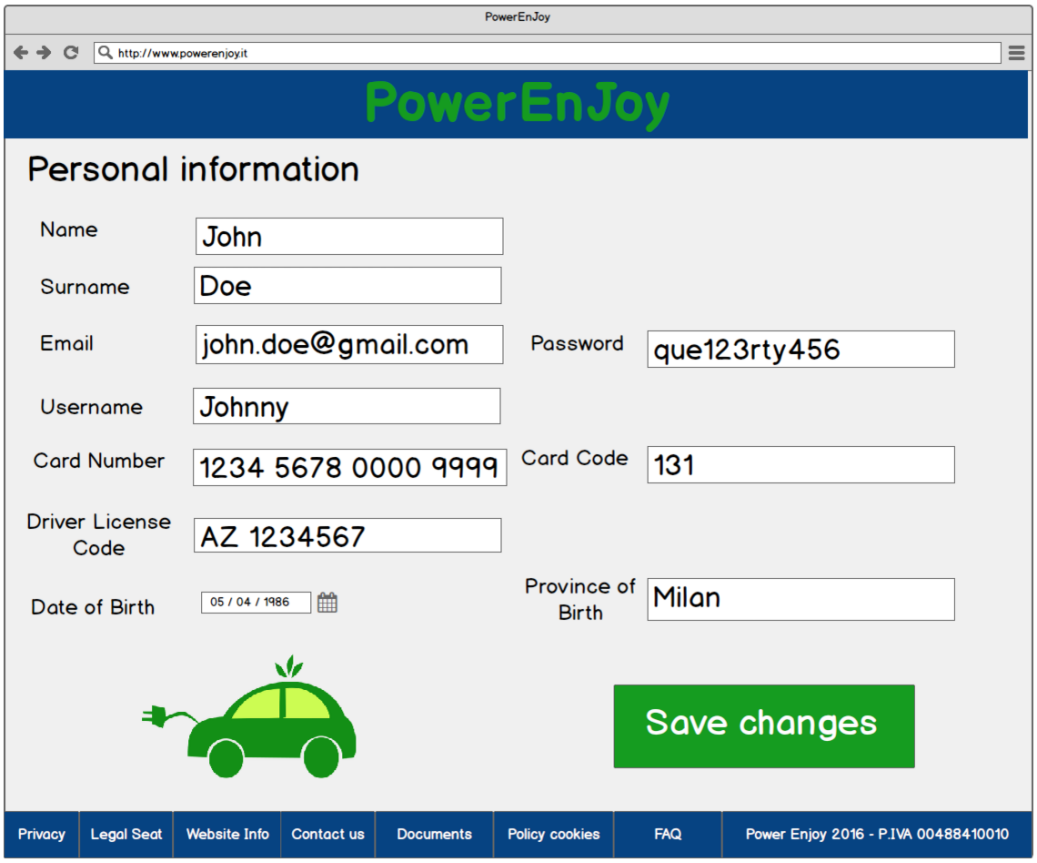
- Personal page:



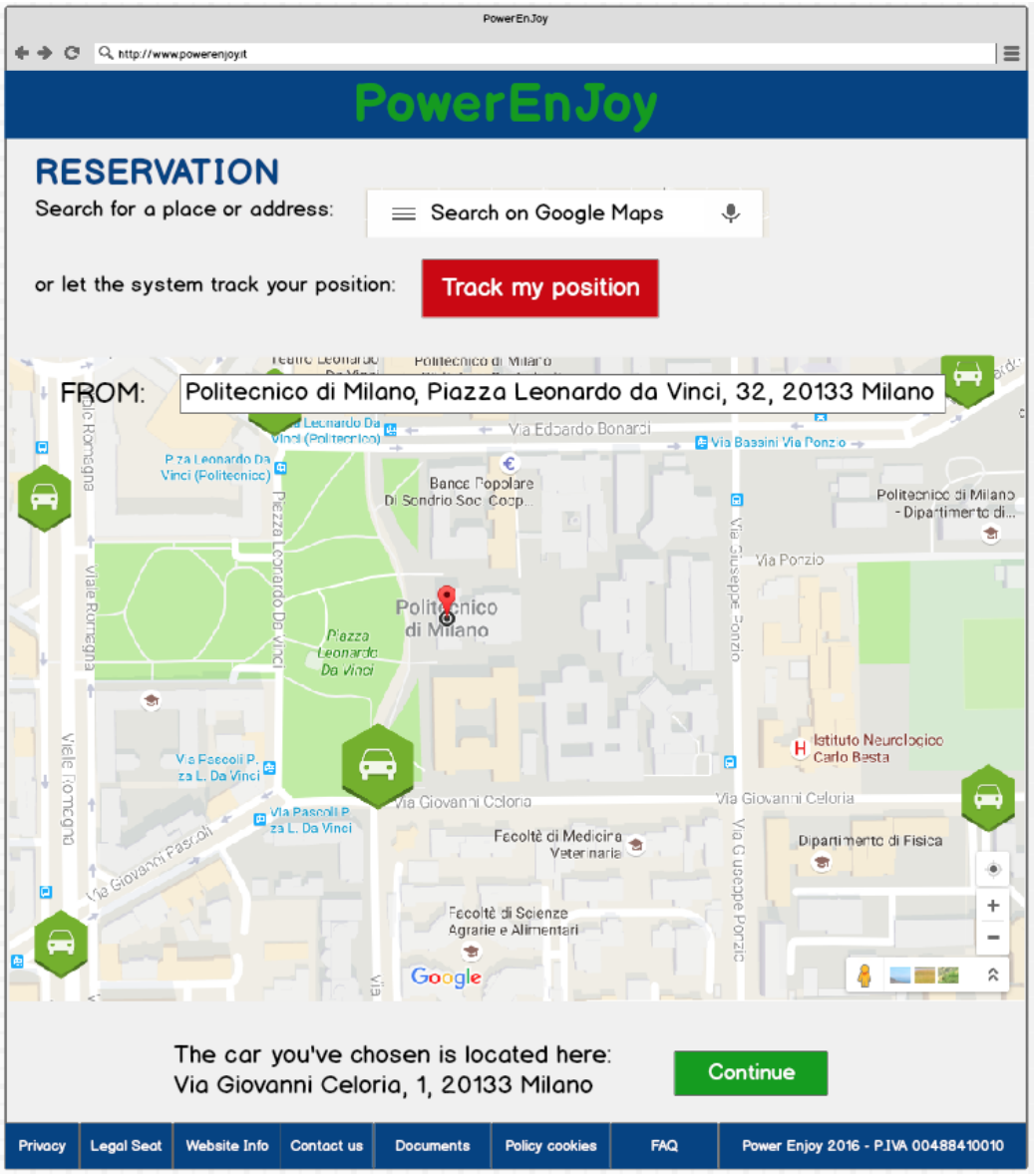
- View profile:



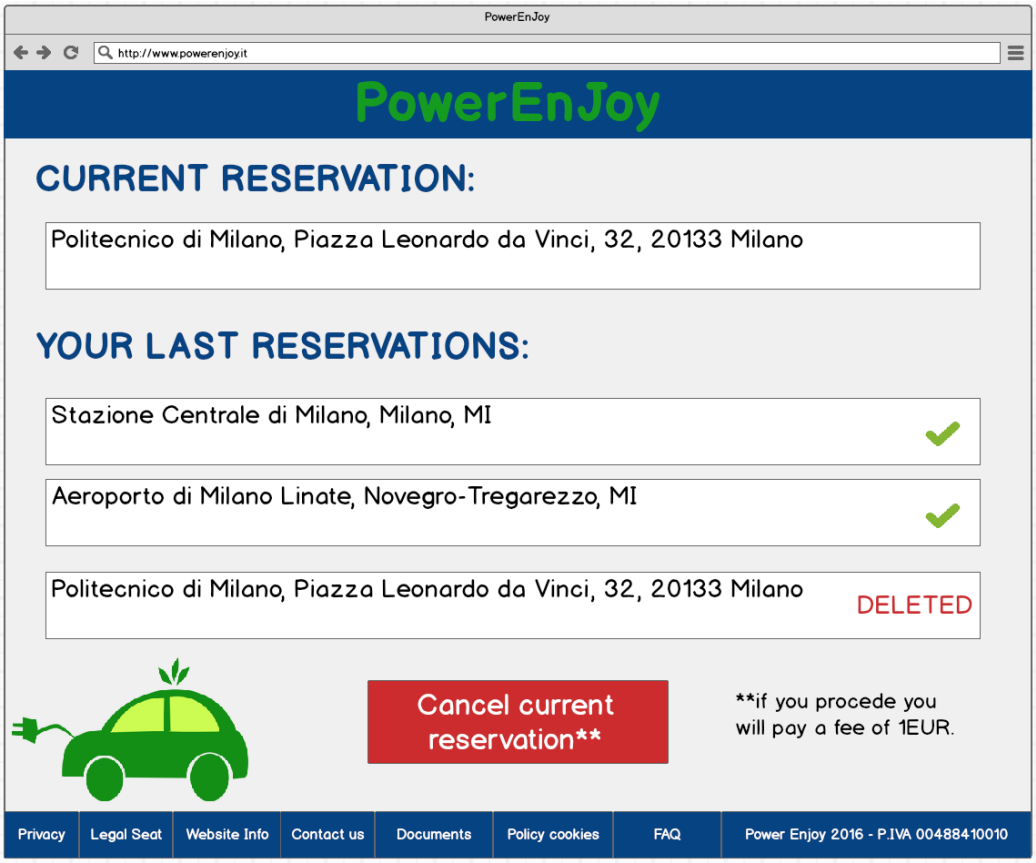
- Managing personal information:



- Reservation:



- Deleting a reservation:



3.2.3 Car

- Car screen asking for destination:

- Car screen with map and information about the ride:

- Car screen after the destination is reached:

3.2.4 Architectural considerations

We will use the following technologies:

- **MySQL**, for the storage of all the information related to both the PowerEnJoy application and the users.
- **Mapping service**, to keep track of the position of both cars and registered users (only the ones who allow it).
- **PHP**, to build the back-end of the application.
- **JavaScript, HTML, CSS** and **Bootstrap**, to create a responsive and well-designed website.
- **PhoneGap**, to create a responsive mobile application.
- Modern browser with JavaScript and AJAX support.
- **Java** for Android and iOS apps, using original SDK.
- Internet/Ethernet connection for data communication.
- **SMTP (Simple Mail Transfer Protocol)**, to transfer emails from a server to another with a point-to-point connection.

4 SCENARIOS

Here is the description of the possible scenarios.

4.1 Scenario 1

After having lunch with his friends, Johnny decides to not take the subway in order to get home. Instead he wants to benefit from the PowerEnJoy service he registered to the week before.

He logs into the system through the mobile application he previously downloaded on his smartphone and, providing his credentials, he is able to find the location of the available cars near the restaurant.

Since none of the cars is near enough, he closes the application without making any reservation.

4.2 Scenario 2

Tonight Johnny has to attend a family dinner but, since he is late, he tells his sister, who was supposed to pick him up, to go ahead without him.

Just before leaving his house, he realizes it is a public holiday and therefore there are no public transports available. In order to reach his family's house, he then decides to benefit from the PowerEnJoy service.

He logs into the system website using his laptop and, providing his credentials, he manages to find an available car not far from his house. He reserves the car and, doing so, he is able to attend the family dinner only ten minutes late.

4.3 Scenario 3

After having dinner together, a group of friends decides to go see a movie. Johnny suggests to use the PowerEnJoy service even though none of them has ever registered to it.

Having the smartphone with the best Internet connection, Jack offers to register. After downloading the mobile application, he fills out the registration form providing the required credentials and payment information.

After receiving the password via email, he logs in for the first time and he is able to make a reservation. Once the destination is reached, Jack discovers with pleasure that he can benefit from a 10% discount since he took two other passengers onto the car with him.

Pleased with the service, the group decides to use it also to get back home.

4.4 Scenario 4

Laura, a British environmentalist tourist, decided to visit his friend Johnny in Milan.

One night she tells him she would love to visit the city, so Johnny suggests her to register into the PowerEnJoy system he already used.

Since Laura is a bit reluctant, Johnny tries to convince her by showing her the service website and pointing out that all cars are electric. He also shows her where to keep track of the promotions and how to manage her personal information.

Persuaded by her friend, Laura decides to register into the system the day after.

4.5 Scenario 5

After spending the afternoon at his university, Johnny decides to go visit his girlfriend whose house is forty minutes from him.

He then chooses to use the PowerEnJoy service, aware that there are lots of safe areas near his girlfriend house.

After logging into the system and reserving a car, he decides to bring her a gift and remembers that there is a florist next to his university. Once the flowers are chosen, he realizes that more than an hour has passed since his reservation. He takes his smartphone out of his pocket and realizes he received an email, saying that he has been charged with a 1EUR fee for not showing up in time and that the car he reserved is no more available.

Aware of his mistake, he decides to not overthink it and starts searching for another available car. Once he finds it, he gets there immediately and starts driving to reach his girlfriend's house.

4.6 Scenario 6

After deciding to invite his friends over for lunch, Jack realizes there is nothing in the fridge and hurries to the grocery store. After leaving his house, he notices a PowerEnJoy car right in front of him and he chooses to use it instead of taking the bus.

Once the car is reserved, Jack unlocks it and proceeds to the grocery store. After reaching his destination, he sadly realizes there are no free safe areas nearby and decides to leave the car in a generic parking lot, sure that it does not change anything.

Once he is finished with the grocery shopping, he wants to reserve another PowerEnJoy car and, logging into the website, he finds out that his previous

reservation has not yet ended.

Finally Jack decides to head back home with that same car and sadly realizes he could have avoided to pay that much money.

4.7 Scenario 7

After a long day at work, Dave realizes it is raining outside and does not want to walk back home. He then decides to benefit from the PowerEnJoy service, that he has been using for the last month and really appreciate.

Sitting in the hall, Dave starts looking for a car on the mobile application and finally gets the results: there is a car parked a few blocks away and he decides to reserve it.

After getting close enough to the car, he unlocks it and, getting in, realizes with surprise that the car monitor is broken. After spending a few minutes contemplating his options, Dave remembers that the service offers the possibility to report an issue and decides to go on with it.

He then exits the car and decides to walk back home, for the first time in his life disappointed by the service.

4.8 Scenario 8

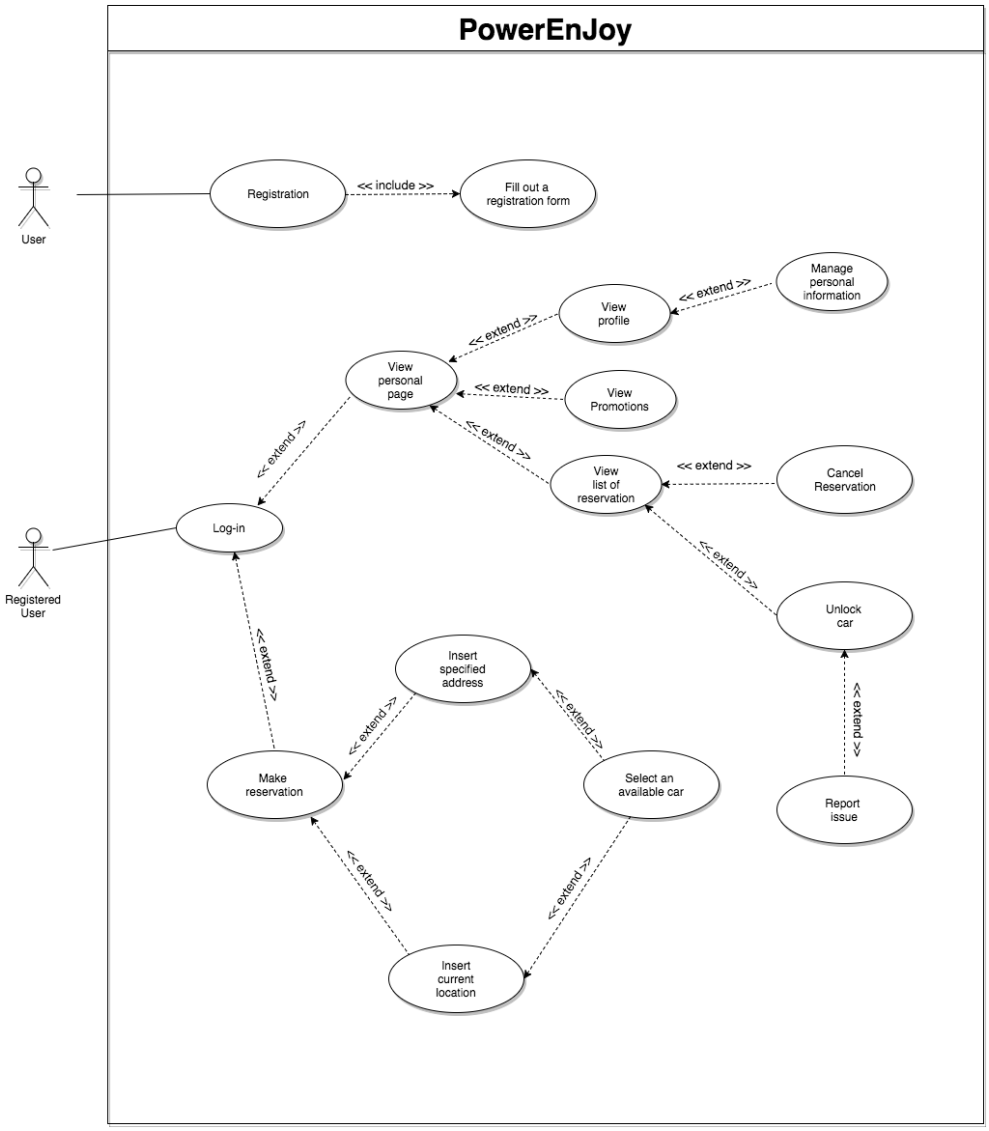
Every morning Alex goes to the university by bike in order to avoid air pollution. Class after class, it is finally time to head back home, but when Alex reaches the courtyard where she parked her bicycle, she realizes it is no longer there. It was stolen!!

Upset by the fact, she decides to use the PowerEnJoy service to reach the nearest police station and report the theft. After logging into the system, she reserves a car not far from her and, after unlocking it, starts her ride.

Once the police station is reached, Alex parks the car in a special safe area right in front of it. She then plugs the car into the power grid and happily sees on the monitor that a 30% discount has been applied on the ride's total price.

5 UML MODELS

5.1 Use case diagram



5.2 Use case description

In this paragraph all our use cases are described.

- User registration by filling out a form [Sequence Diagram]:

Name	User: registration, fill out a registration form
Actors	Generic user
Entry conditions	There are no entry conditions.
Flow of Events	1. The user downloads the mobile application or accesses the service's website. 2. The user clicks on the "Sign in" button. 3. The user enters his credentials and payment information. 4. The user clicks again on the "Sign in" button. 5. The user receives a password by email.
Exit conditions	The user makes his first login with the assigned password and the username he chose.
Exceptions	The username is already taken or the user did not fill in all the mandatory fields.

- Registered user login [Sequence Diagram]:

Name	Registered user: login
Actors	Registered user
Entry conditions	The registered user has previously registered into the system.
Flow of Events	1. The registered user accesses the service's website or opens the mobile application. 2. The registered user enters his credentials into the corresponding fields. 3. The registered user clicks on the "Login" button.
Exit conditions	The registered user is able to see his personal page.
Exceptions	The username-password combination does not exist into the database.

- Registered user views his profile [Sequence Diagram]:

Name	Registered user: view profile
Actors	Registered user
Entry conditions	The registered user has previously logged into the system.
Flow of Events	1. The registered user accesses his profile by clicking on the "View your profile" button.
Exit conditions	The registered user is able to see his profile page.
Exceptions	Wrong button clicked.

- Registered user manages his personal information [Sequence Diagram]:

Name	Registered user: manage personal information
Actors	Registered user
Entry conditions	The registered user has previously logged into the system.
Flow of Events	1. The registered user accesses his profile by clicking on the "View your profile" button. 2. The registered user can modify his personal information by clicking on the "Manage personal information" button. 3. The registered user makes some changes (or none). 4. The registered user clicks on the "Save changes" button.
Exit conditions	The registered user's personal information is updated.
Exceptions	Wrong data entered.

- Registered user cancels his current reservation [Sequence Diagram]:

Name	Registered user: cancel reservation
Actors	Registered user
Entry conditions	The registered user has previously logged into the system.
Flow of Events	1. The registered user clicks on the "View the list of your reservations" button. 2. The registered user visualizes the page showing all his reservations. 3. The registered user clicks on the "Cancel current reservation" button.
Exit conditions	The registered user's current reservation no longer exists and is added to the past reservations' list labeled as "Deleted".
Exceptions	The registered user has no current or past reservations.

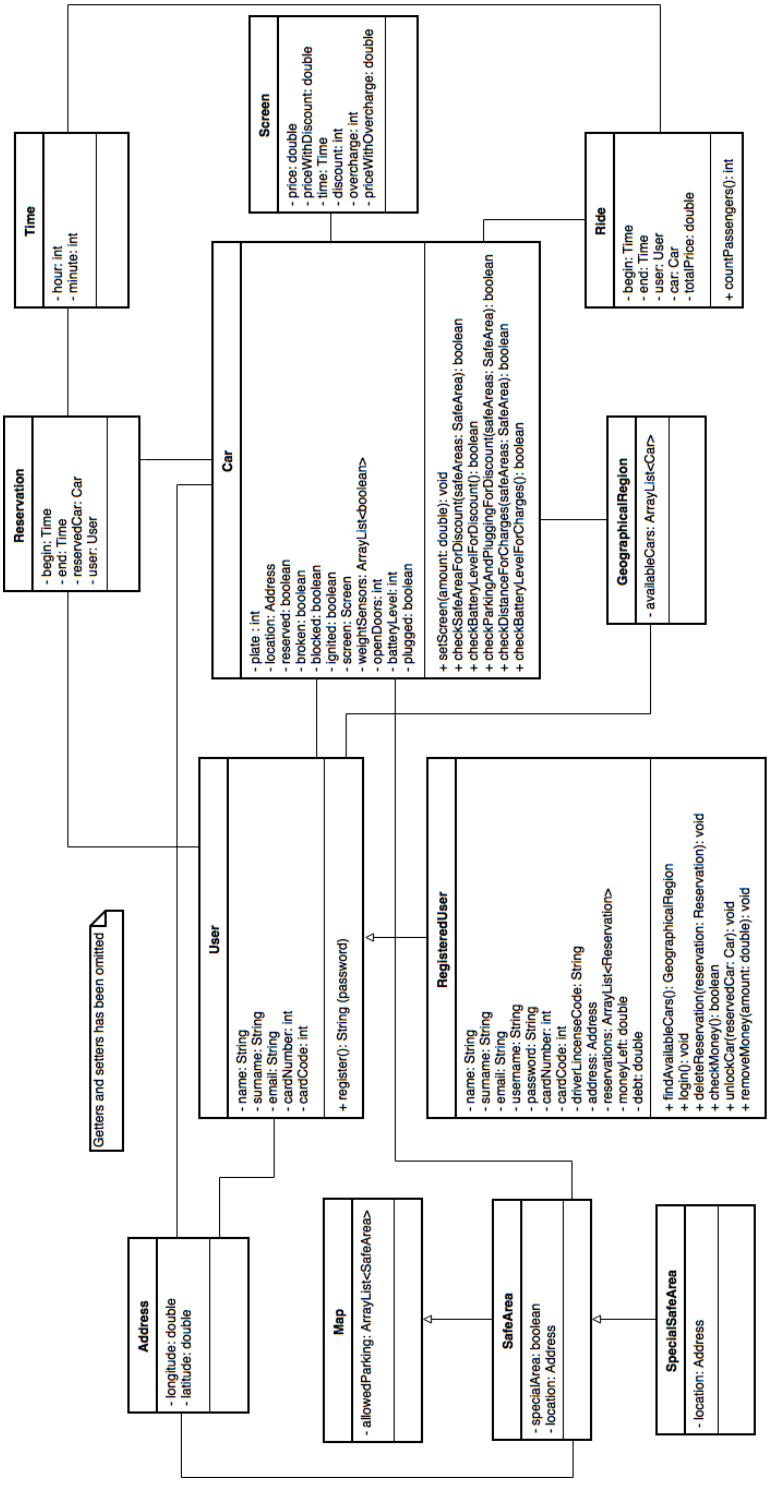
- Registered user reports an issue after unlocking the car [Sequence Diagram]:

Name	Registered user: unlock car, report issue
Actors	Registered user
Entry conditions	The registered user has previously logged into the system, made a reservation and selected a car.
Flow of Events	1. The registered user visualizes the list of his previous and current reservations from his personal page. 2. The registered user clicks on the "Unlock car" button next to his current reservation. 3. The registered user visualizes the car's information. 4. The registered user checks the car for possible issues. 5. The registered user clicks on the "Report issue" button.
Exit conditions	The registered user's current reservation is deleted.
Exceptions	The car presents no issues.

- Registered user makes a reservation by selecting a car [Sequence Diagram]:

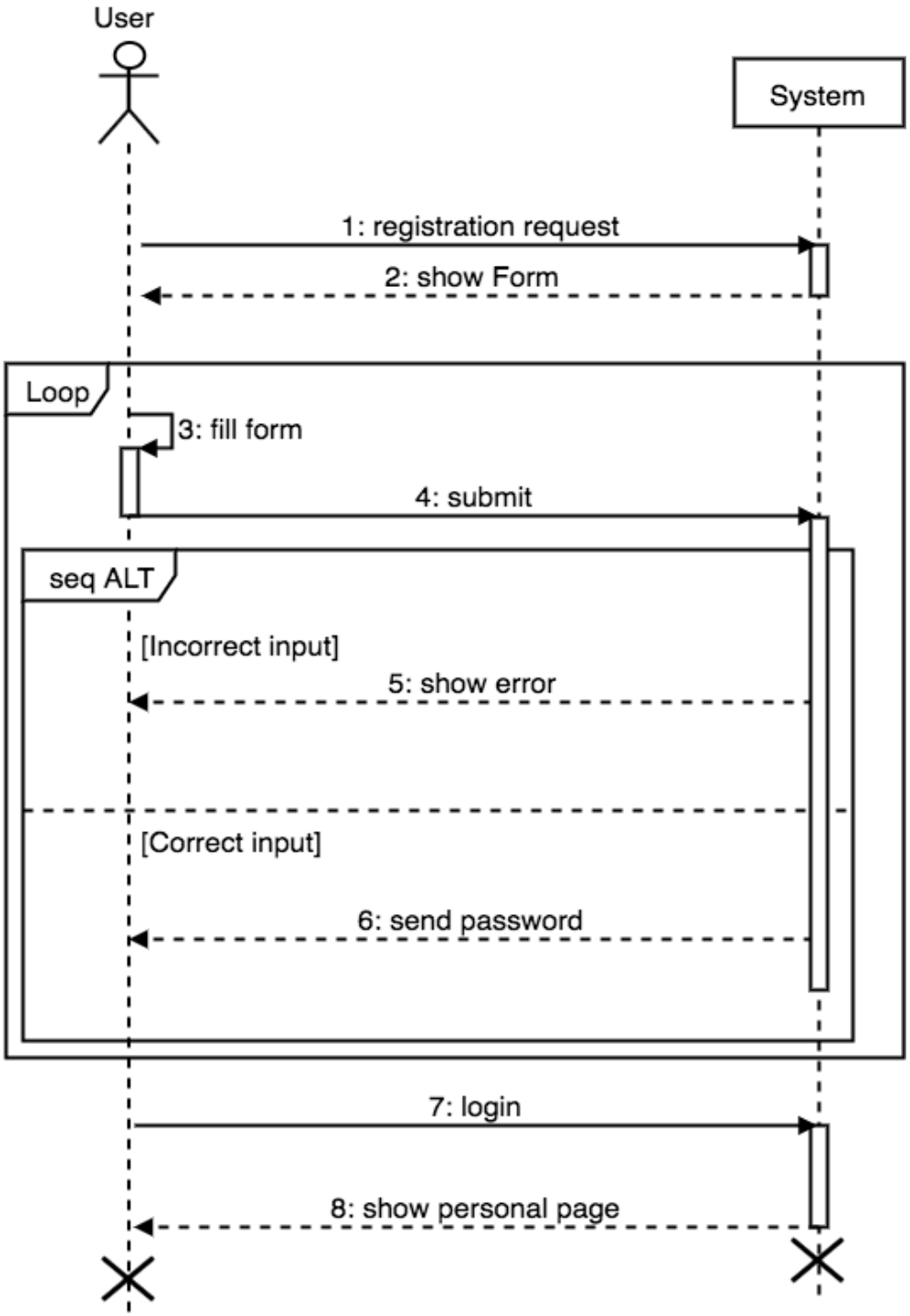
Name	Registered user: make reservation, insert specific address/current location, select available car
Actors	Registered user
Entry conditions	The registered user has previously logged into the system.
Flow of Events	1. The registered user clicks on the "Click here" button from his personal page. 2. The registered user enters a valid address or lets the system track his position by clicking on the "Track my position" button. 3. The registered user visualizes all the available cars on a map. 4. The registered user chooses one of the available cars shown on the map. 5. The registered user clicks on the "Continue" button to reserve the car.
Exit conditions	The registered user is able to see the page with all his previous and current reservations.
Exceptions	A non valid address is entered. None of the available cars is selected by the registered user. The registered user's GPS is not active although he has chosen the "Track my position" option.

5.3 Class diagram

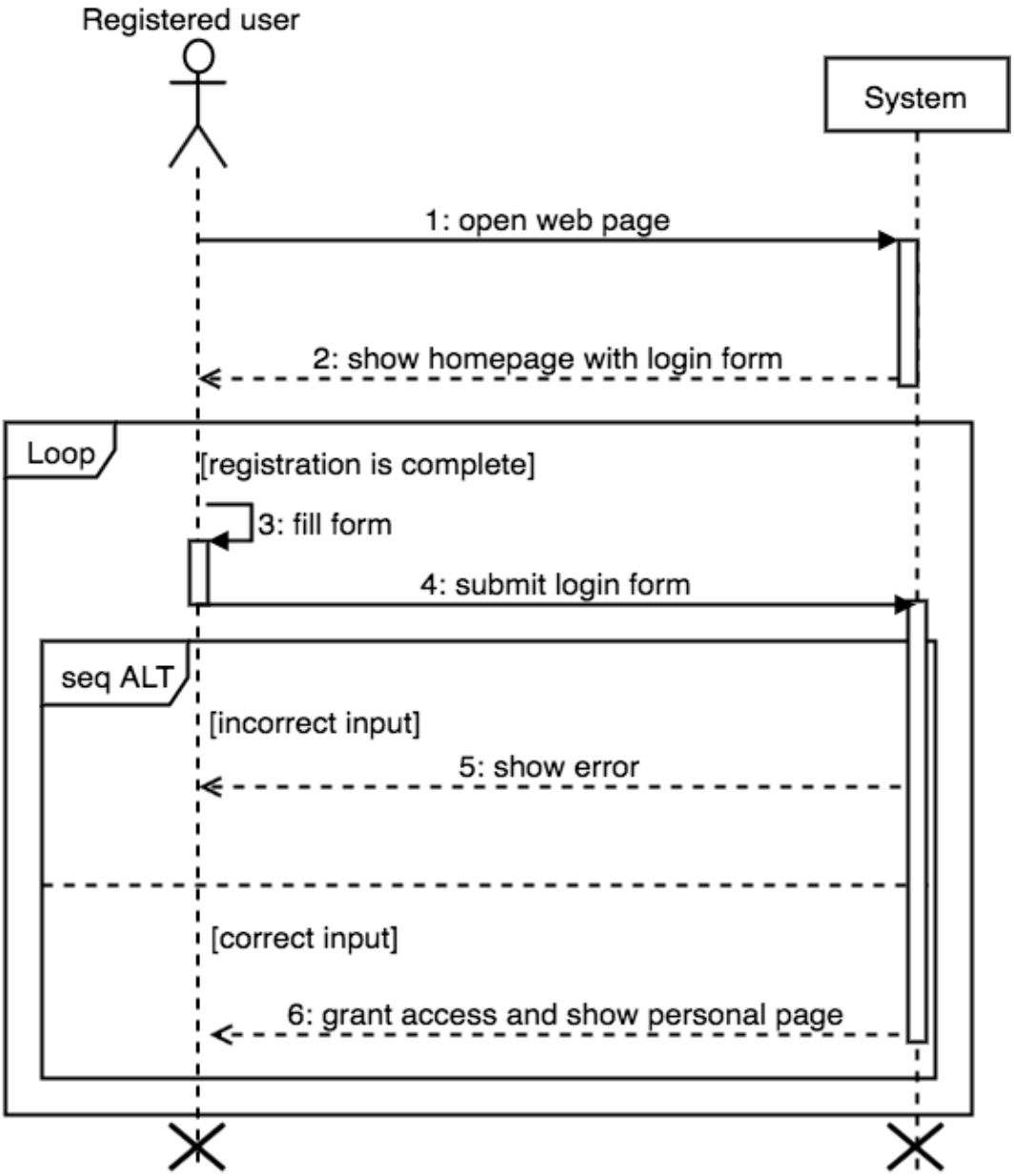


5.4 Sequence diagrams

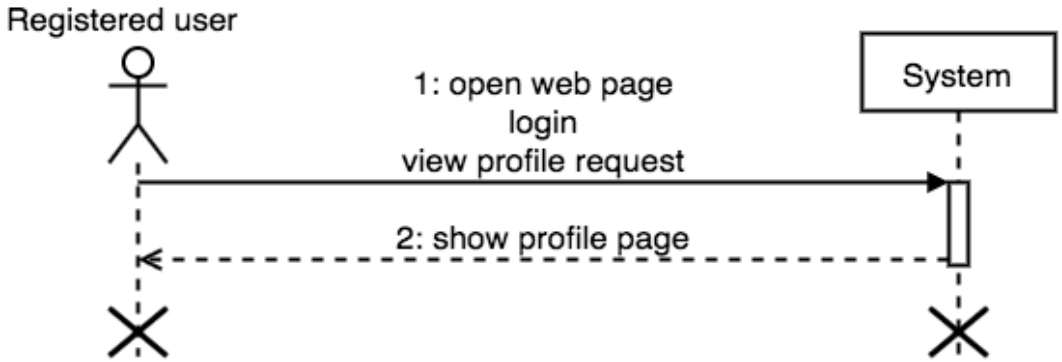
- User's registration:



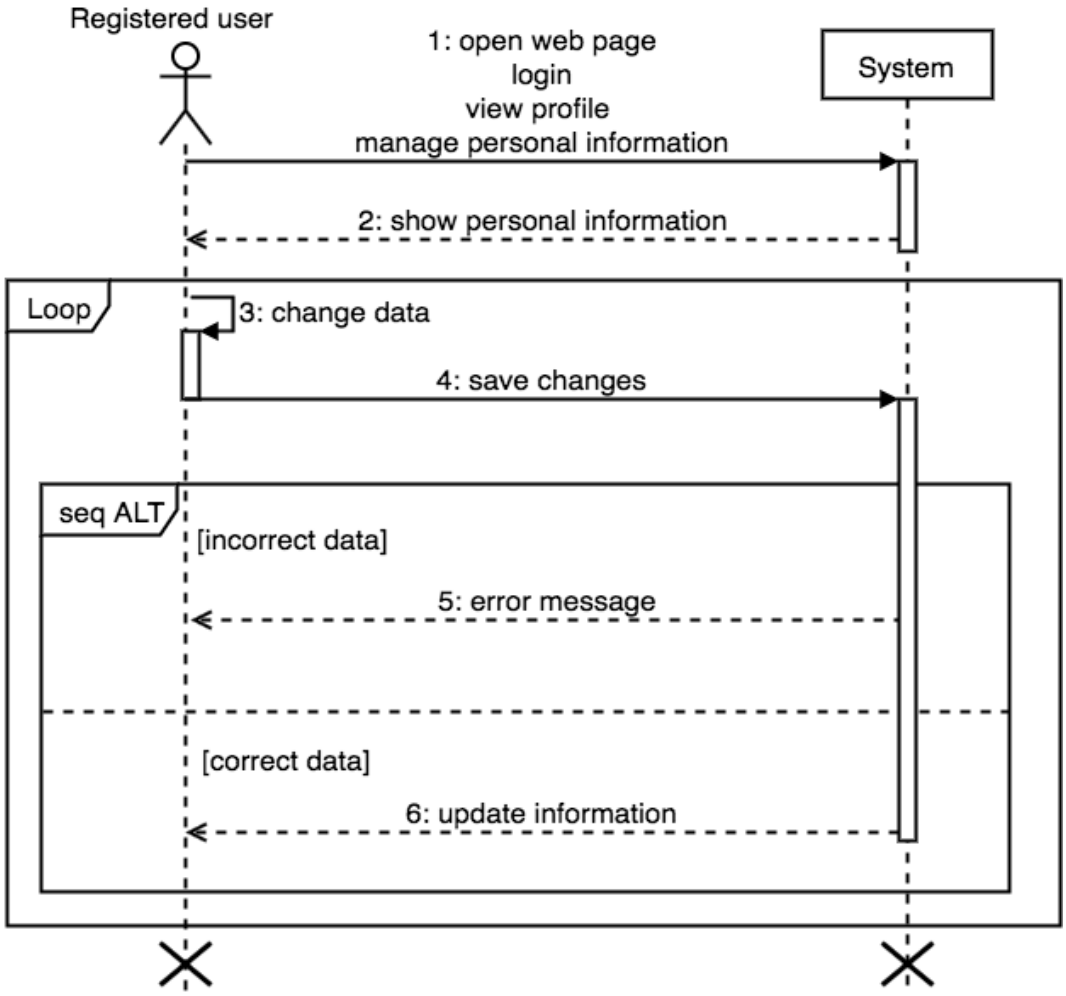
- Registered user login:



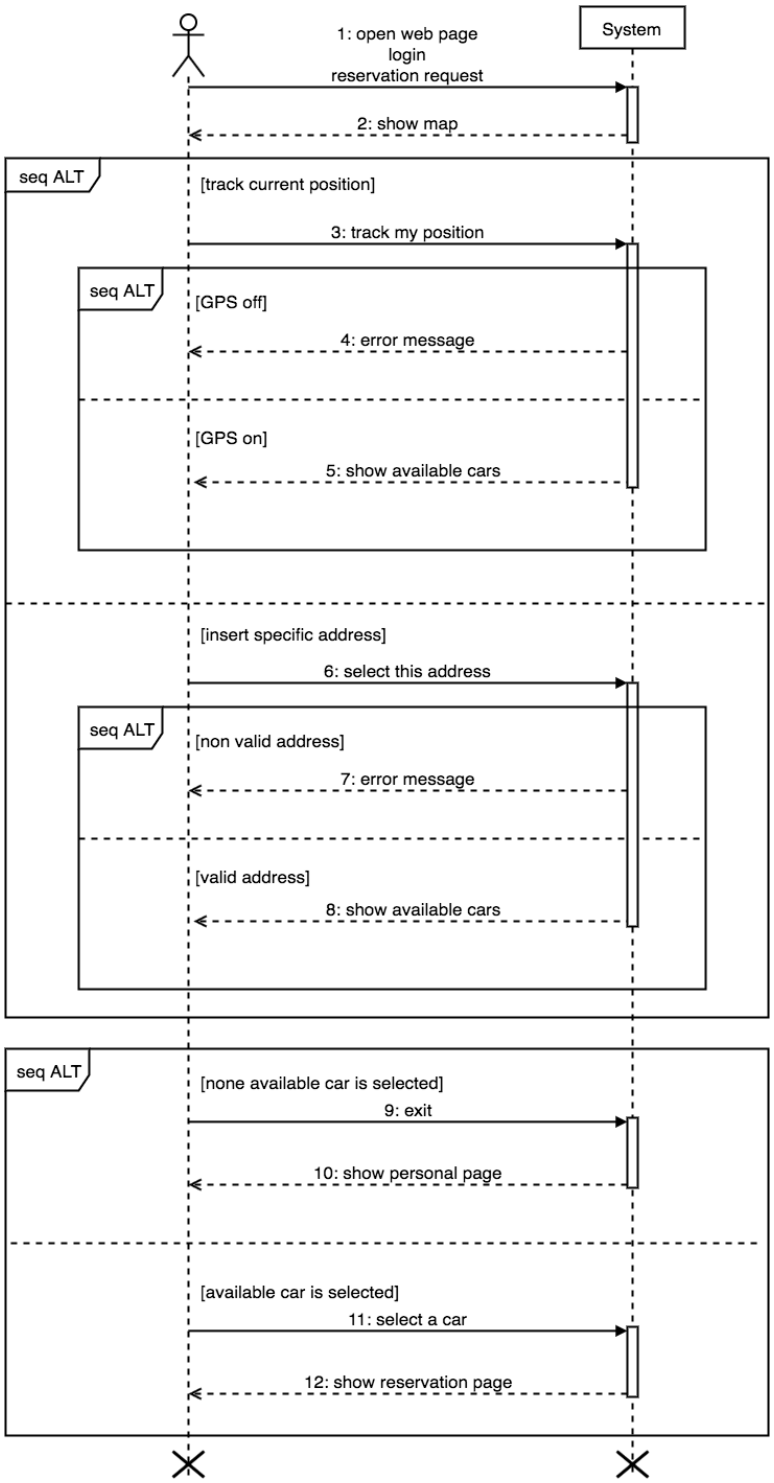
- Registered user views his profile:



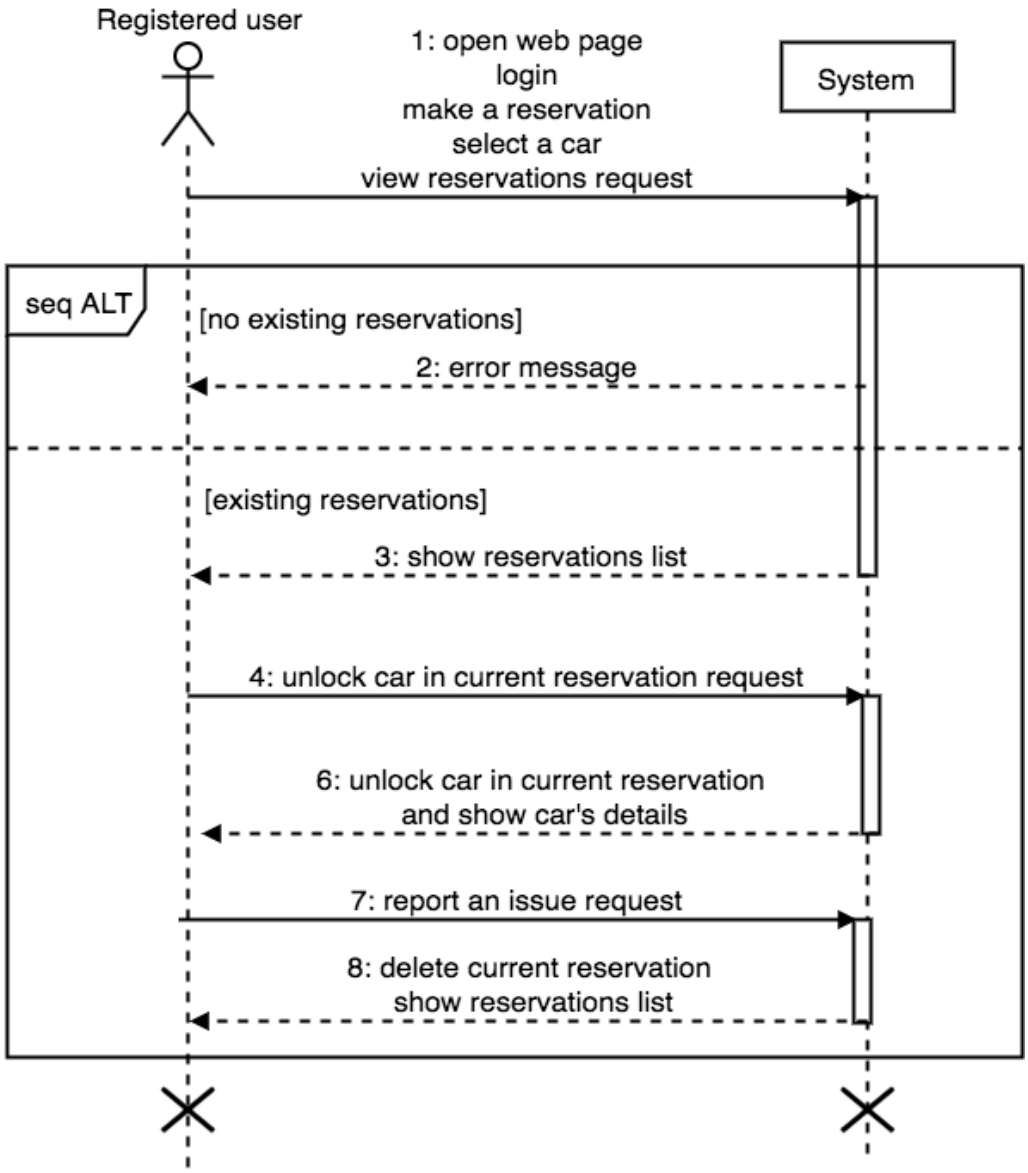
- Registered user manages his personal information:



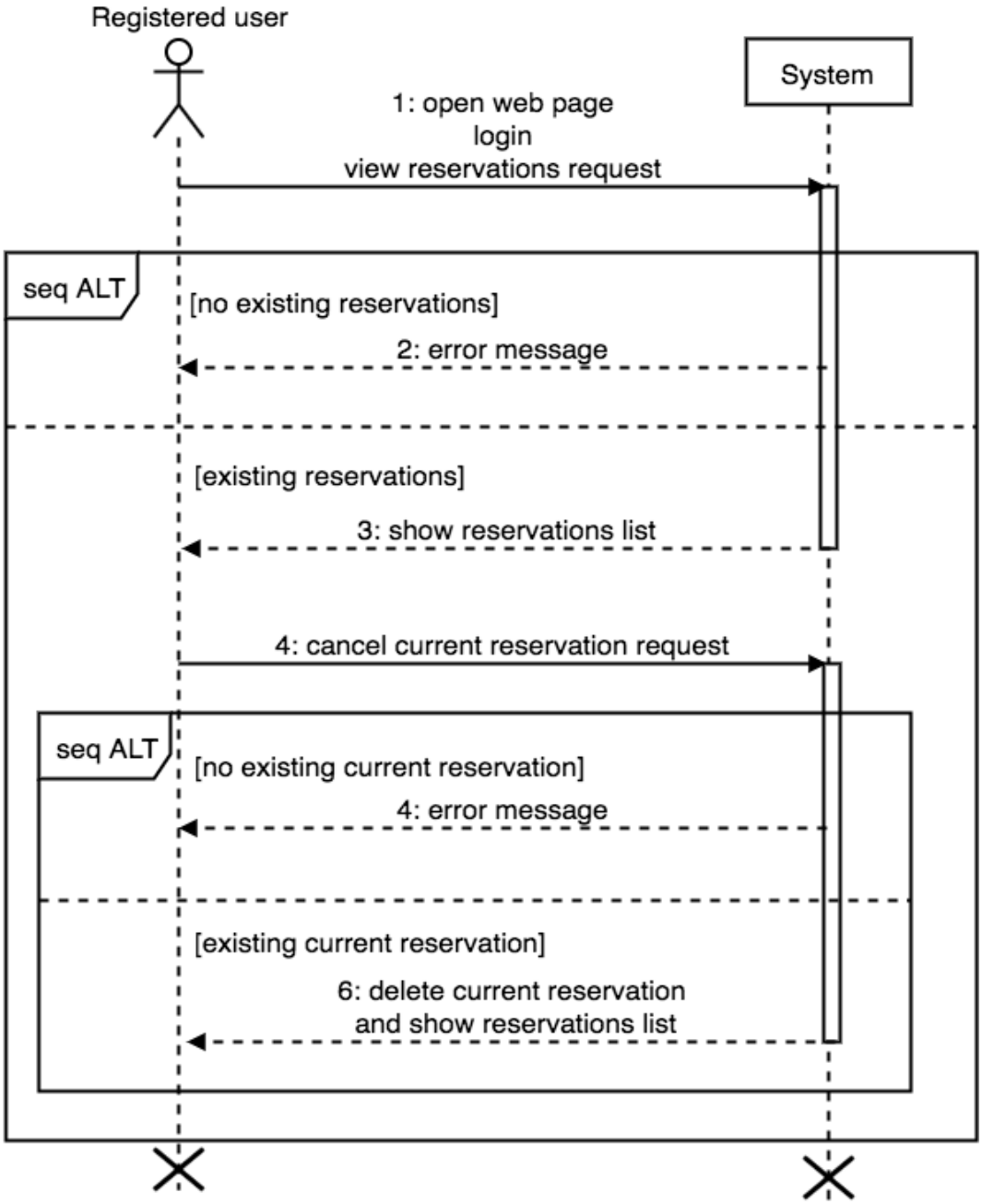
- Registered user makes a reservation by selecting a car:



- Registered user reports an issue after unlocking the car:

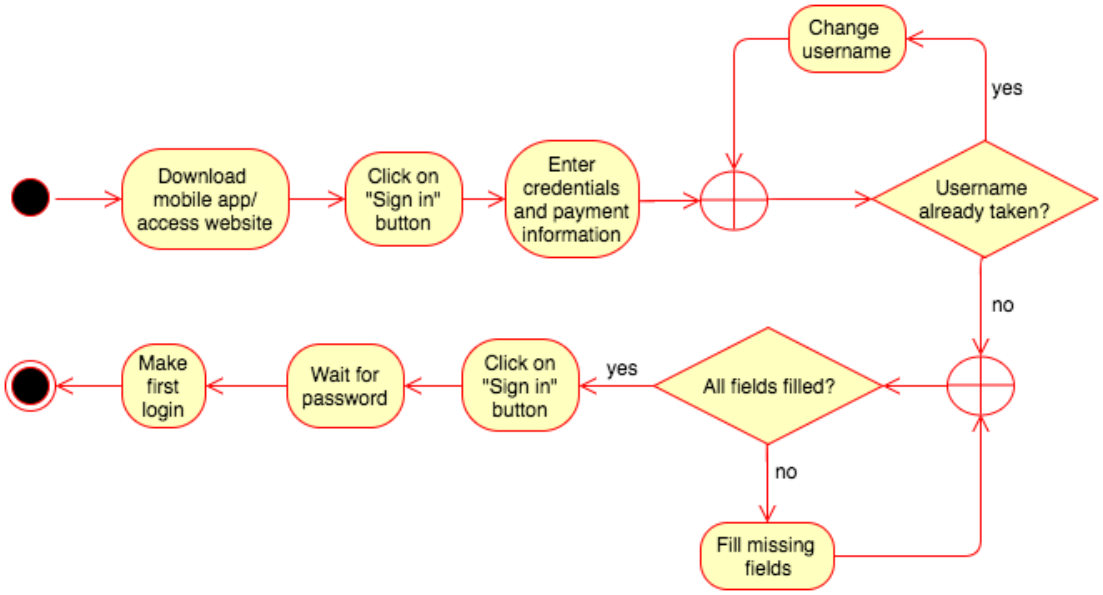


- Registered user cancels his current reservation:



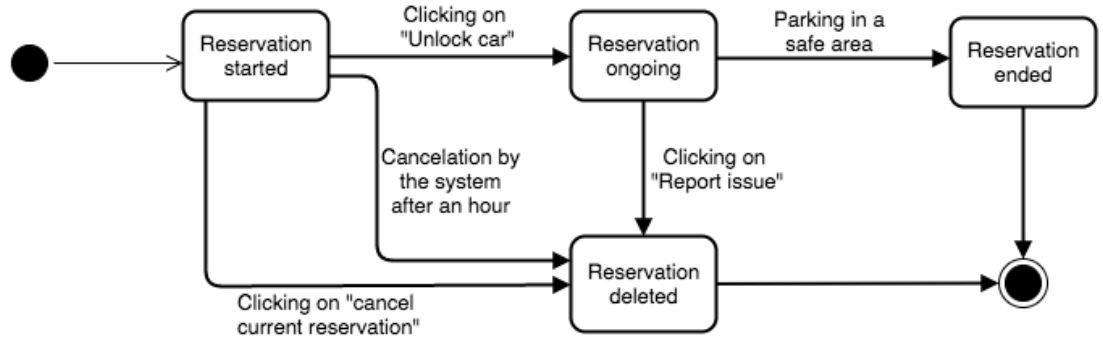
5.5 Activity diagram

User's registration by filling out a form:



5.6 State diagram

Reservation:



6 ALLOY

6.1 Model

```
open util/boolean

sig Username {}

sig Address {
longitude: Int, //should be double
latitude: Int //should be double
} {
longitude > 0
latitude > 0
}

fact differentAddresses {
all a1, a2: Address | (a1 != a2) <=> (a1.latitude  != a2.latitude) || (a1.longitude != a2.longitude)
}

sig Time {
hour: Int,
minute: Int
} {
hour >= 0
minute >= 0
}

abstract sig Discount{
totalDiscount: Int
}
{
totalDiscount > 0
}

one sig Discount1 extends Discount{}

one sig Discount2 extends Discount{}

one sig Discount3 extends Discount{}

fact CarsAndReservations {
#(Reservation) = #(Car.reserved)
}
```

```
sig Reservation {
reservedCar: one Car,
user: one RegisteredUser
}

fact uniqueCarReservation{
no disj r1,r2: Reservation | (r1.reservedCar = r2.reservedCar)
}

fact uniqueUserReservation{
no disj r1,r2: Reservation | (r1.user = r2.user)
}

fact carNotReserved{
all  c: Car | no r1: Reservation | c.reserved = False && r1.reservedCar = c
}

fact pluggedCarsInSafeArea2 {
all sa: SafeArea, c: sa.cars.elems | c.plugged = True <=> sa.specialArea = True
}

sig SafeArea {
specialArea: one Bool,
cars: seq Car
}

fact noDuplicatesInSafeArea {
all sa: SafeArea | not sa.cars.hasDups
}

fact differentCarSafeArea{
all c:Car | no disj sa1,sa2: SafeArea | (c in sa1.cars.elems && c in sa2.cars.elems)
}
```

```
sig Car {
plate: Int,
location: one Address,
reserved: one Bool,
broken: one Bool,
ignited: one Bool,
weightSensors: Int,
plugged: one Bool,
screen: one Screen,
lowBattery: one Bool
} {
plate>0
weightSensors > 0
weightSensors <= 5
}

fact allScreensAreOwned{
Car.screen =Screen
}

fact uniqueCar{
no disj c1,c2:Car | (c1.plate = c2.plate)
}

fact carAddress {
no disj c1,c2 : Car | (c1.location = c2.location)
}

fact carScreen {
no disj c1,c2 : Car | (c1.screen = c2.screen)
}

fact carBroken {
all c: Car | (c.broken = True) => (c.reserved = False)
}

fact carIgnited {
all c: Car | (c.ignited = True) => (c.reserved = True &&  c.broken = False)
}
```

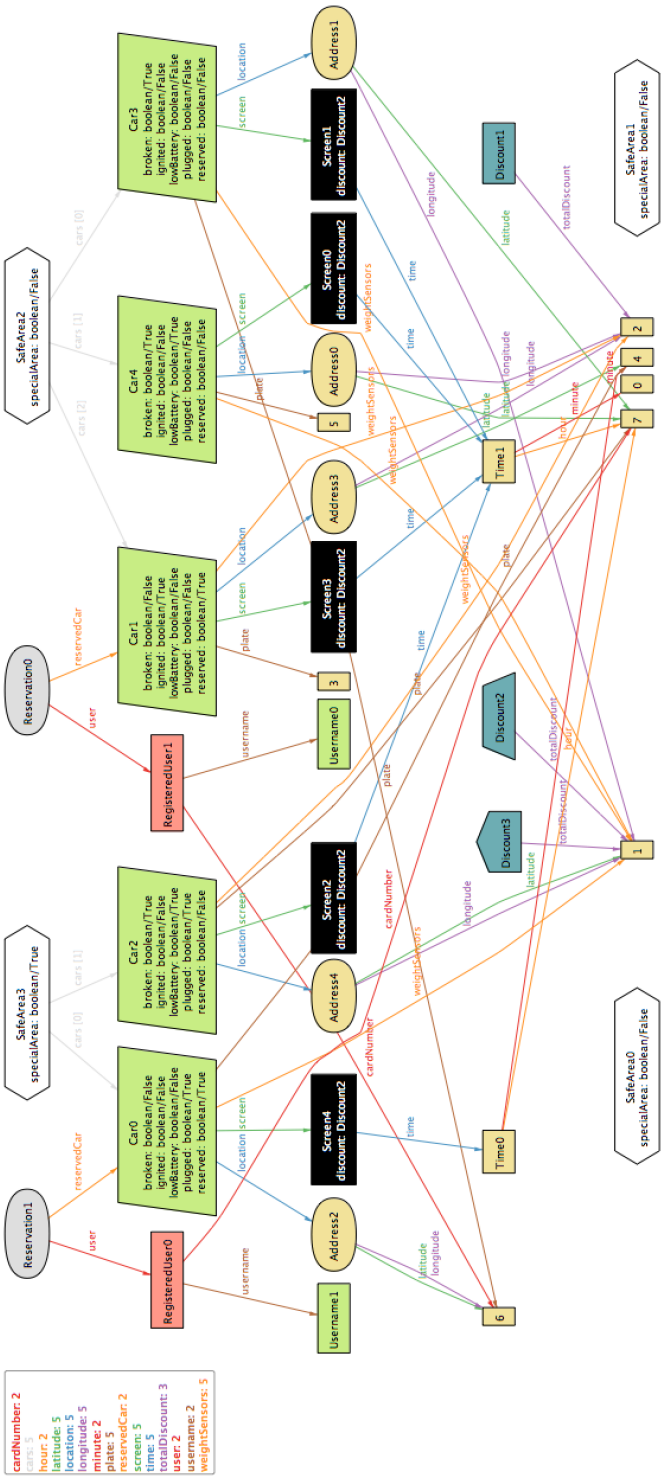
```
fact carPlugged{
all c: Car | (c.plugged = True) => (c.ignited = False)
}

fact carLowBattery {
all c: Car | (c.lowBattery = True) => (c.reserved  =False)
}

sig Screen {
time: one Time,
discount: Discount
}

fact discountPassengers {
all c:Car, s: Screen |  (c.weightSensors >= 3) => s.discount = Discount1
}

fact discountBattery {
all c:Car, s: Screen |  (c.lowBattery = False) => s.discount = Discount2
}

sig RegisteredUser{
cardNumber: Int,
username: one Username
} {
cardNumber > 0
}

fact allUsernamesAreOwned{
RegisteredUser.username=Username
}
fact uniqueUsername {
no disj u1, u2: RegisteredUser | (u1.username =u2.username)
}

fact uniqueCardNumber{
no disj u1, u2: RegisteredUser | (u1.cardNumber = u2.cardNumber)
}
```

```
assert reservationAndRegisteredUser{
all r:Reservation |r.user in RegisteredUser
}
check reservationAndRegisteredUser

assert numberCarReservedAndReservation{
all c:Car, r: Reservation | c.reserved = False => c!=r.reservedCar
}
check numberCarReservedAndReservation

assert notSpecialSafeAreaDoesNotHavePluggedCar {
all sa: SafeArea, c: Car | sa.specialArea = False && c.plugged = True => c not in sa.cars.elems
}
check notSpecialSafeAreaDoesNotHavePluggedCar

pred show(){
#SafeArea=4
#Car=5
}
run show for 6
```

6.2 Alloy result

```
4 commands were executed. The results are:
  #1: No counterexample found. reservationAndRegisteredUser may be valid.
  #2: No counterexample found. numberCarReservedAndReservation may be valid.
  #3: No counterexample found. notSpecialSafeAreaDoesNotHavePluggedCar may be valid.
  #4: Instance found. show is consistent.
```

6.3 World generated



7 FUTURE DEVELOPMENT

Expanding the PowerEnJoy service to other cities in Italy, starting from Rome; this requires the introduction of new maps, created by adding the missing addresses.

8 USED TOOLS

The tools we used to create this document are:

- **draw.io**
  To create the UML models.
  https://www.draw.io
- **GitHub** and **GitHub Desktop**
  To collaborate with the team and keep track of the changes in the document.
  https://github.com and https://desktop.github.com
- **Balsamiq**
  To design the mockups.
  https://balsamiq.com
- **TeXstudio**
  LaTeX editor we used to write this document.
  http://www.texstudio.org
- **BasicTeX**
  Distribution of the LaTeX system.
  http://www.tug.org/mactex/morepackages.html
- **Alloy Analyzer 4.2**
  To build strong and consistent models.
  http://alloy.mit.edu/alloy/

9 HOURS OF WORK

9.1 Agosti Isabella

- 20/10/16: 3h
- 27/10/16: 2h
- 29/10/16: 3h
- 30/10/2016: 3h
- 02/11/2016: 2h
- 03/11/2016: 3h
- 04/11/2016: 3h
- 05/11/2016: 8h
- 06/11/2016: 2h
- 07/11/2016: 3h
- 10/11/2016: 3h
- 11/11/2016: 5h
- 12/11/2016: 8h
- 13/11/2016: 4h

9.2 Cattivelli Carolina

- 20/10/16: 3h
- 27/10/16: 2h
- 29/10/16: 3h
- 30/10/2016: 3h
- 02/11/2016: 2h
- 03/11/2016: 3h
- 04/11/2016: 3h
- 05/11/2016: 8h
- 06/11/2016: 2h
- 07/11/2016: 3h
- 10/11/2016: 3h
- 11/11/2016: 5h
- 12/11/2016: 8h
- 13/11/2016: 4h