PowerEnJoy

## Requirement Analysis and Specification Document

Cattaneo Davide Frontino Francesco El Hariry Matteo

Contents

1. ***Introduction***
   1. Purpose
   2. Goals
   3. Definitions, acronyms, abbreviations
   4. Reference documents
   5. Document overview
2. ***Overall description***
   1. Product perspective
   2. Product functions
   3. User characteristics
   4. Constraints
   5. Assumptions and dependencies
   6. Future possible implementation
3. ***Specific requirements***
   1. External interface requirements
      1. User interfaces
      2. Hardware interfaces
      3. Software interfaces
      4. Communication interfaces
   2. Functional requirements
   3. Non-functional requirements
   4. Performance requirements
   5. Scenarios
   6. UML Models
      1. Use cases
      2. Class diagram
      3. State diagrams
4. ***Overall description***
   1. Alloy
      1. Signatures
      2. Facts
      3. Assertions
      4. Predicates
      5. Alloy model
      6. Generated world
   2. Software and tools used
   3. Team work
5. ***Revision***
   1. Changed assumptions
   2. Modified functional requirements
   3. Modified scenarios
   4. Modified use cases
   5. Modified diagrams
6. ***Introduction***
   1. ***Purpose***

With this project we will design an electric-car sharing software system.

Car Sharing is a very cost-effective and useful service for anyone who needs a car occasionally. It allows people to use and pay for the car according to their personal use, without the hassle and costs of owning their own vehicle (parking, purchase costs, maintenance, insurance etc.).

The system that we will develop is meant for cities which are provided with an efficient amount of parking lots and a wide distribution of electric car-charging platforms throughout the urban areas.

The application must allow the users which are registered to perform several easy and effective operations. Once logged in, the user can find available cars around him/her or in specified locations of the city, and chose the one to reserve.

Afterwards the user, who needs to reach the car before a given time slot expiration, will be able to easily enter the vehicle and drive to his/her destination.

* 1. ***Goals***

To create a throughout solution for the problem of providing a good and effective service to our users the application must allow the following:

• Users must be able to register to the system by providing their credentials and payment information. They receive back a password that can be used to access the system.

• Registered users must be able to find the locations of available cars within a certain distance from their current location or from a specified address

• Among the available cars in a certain geographical region, users must be able to reserve a single car for up to one hour before they pick it up.

• If a car is not picked‐up within one hour from the reservation, the system tags the car as available again, and the reservation expires; the user pays a fee of 1 EUR

• A user that reaches a reserved car must be able to tell the system he’s nearby, so the system unlocks the car and the user may enter.

As soon as the engine ignites, the system starts charging the user for a given amount of money per minute; the user is notified of the current charges through a screen on the car.

• The system stops charging the user as soon as the car is parked in a safe area and the user exits the car; at this point, the system locks the car automatically.

• The set of safe areas for parking cars is pre‐defined by the management system.

• In order for the system to evaluate properly an eventual discount for the user’s ride, it has to know the parking location, the battery level, status (in charge or not) and if there were two passengers onboard.

* 1. ***Definitions, acronyms, abbreviations***

Here is a brief description of the most important actors and words used in our system:

* **User:** by user we mean a person already registered in the system, so that has a profile, uses the features provided by the system and perform actions accordingly. He can use all the functionalities described below (see Functional Requirements).
* **Guest:** a guest is a person that probably for the first time accesses the system or that hasn’t already signed up. Guest has less power in the system than a user, his functionalities are limited to access an introduction view and to register.
* **System:** is the application core. The software system which will perform all the operations and monitor interactions between users and cars.
* **Reservation:** the allocation of a car to a user, which starts when the booking request arrives and ends either when the expiration time ends or when the car is unlocked.
* **Car:** the vehicle used by the users, which contains different sensors and an embedded computer. It has seat sensors to detect if there are passengers, battery level and charging sensors. The computer of course has as main functionality to provide navigation through GPS system and to send all the relevant data to the main system server.
* **Ride:** conceptually is the use of the car, and it can be identified by the time duration of the user’s journey, from unlocking the vehicle until the final parking with the car locked.
* **Administrator:** the administrator of the system is the person allowed to manage eventual unexpected cases (like incidents and damaging situations).
  1. ***Reference documents***
  2. ***Document overview***

1. ***Overall description***
   1. ***Product perspective***

We propose a mobile application platform that will provide the public with the services described below.

Users will be able to reserve the car which better suites their location preference and enter it as soon as they approach the door leaning the card on the door lock device. Once the car is unlocked the system start charging the user with the halt-rate until he starts the engine (then the normal rate starts). The rate will keep changing according to the engine mode (on with normal rate/off with halt rate) until the user reaches his destination and exit the vehicle (car lock enabled).

User may also interact with the system by handling his own profile with different provided operations (top up wallet, manage coupons, send gifts etc.).

The administrator will be given a special module to manage specific and special situations.

Server will generate different user experiences and consequentially different pages basing on the actor that’s using the platform.

* 1. ***Domain properties***

We suppose that the following conditions hold in the analyzed world.

* Once a user unlocks the reserved car (s)he uses it according to the functionalities provided.
* The GPS always gives the right position.
* The GPS of the car cannot be switched off.
* The system will consider only functioning cars, the once with any problem will be left to the maintenance.
* Payments are always considered to be executed successfully.
* User money is always considered to be available due to the use a credit cards only.
* Battery conditions are managed by the system that guarantees that available cars always have a fully charged battery
* Discount calculation is always correctly done basing the calculus on date transmitted directly by the car.
* Cars are always able to transmit data.
* When cars which were under maintenance are available again or there are new ones to be considered into the car-sharing system, they will be manually inserted among the available once accordingly with their location and details by the administrator. Analogously when a car is facing any problem which make her not usable, the system admin tags her as unavailable him from the information system.
* Users are able to reserve just one car at a time
* Available cars are always fully charged
  1. ***Assumptions***

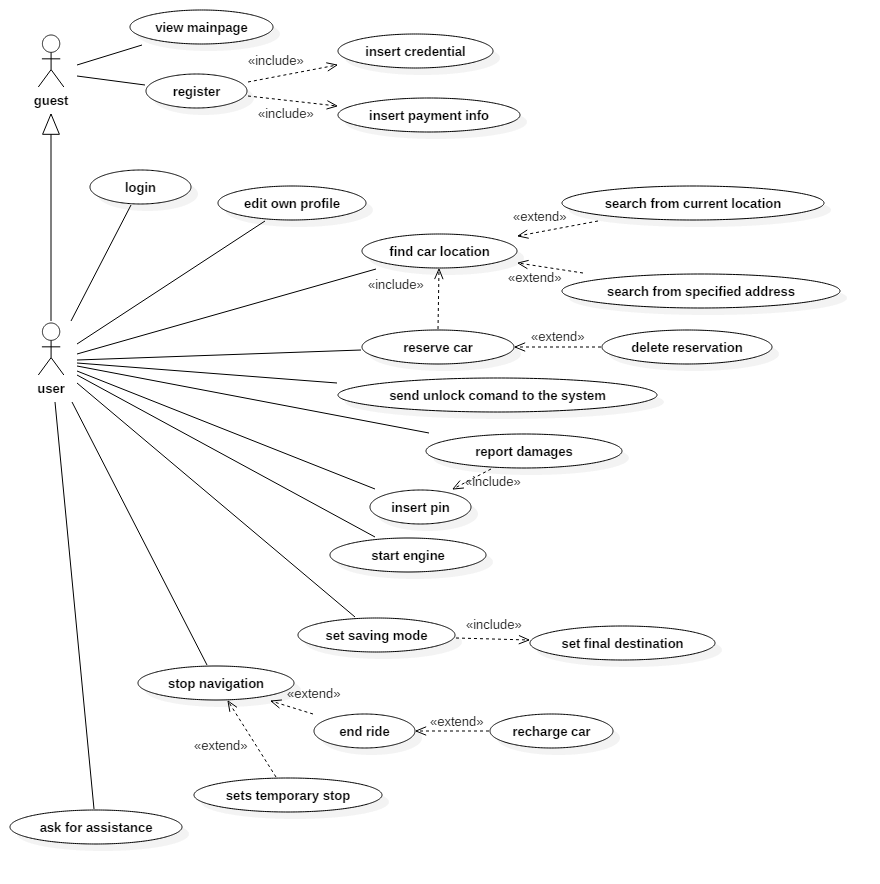
There below we specify how we assume the world works in ambiguous situations:

* The car is unlocked when the user inserts the correct PIN in the lock device, which is able to match the user profile with the one who reserved the car.
* During the period between the car opening and the car locking, in which the car engine is not running, the user will be charged with a halting rate, which will be smaller than the normal charge but useful in avoiding people from occupying the vehicle for free without using it.
* When dealing with safe parking evaluation, we assume that the onboard computer can identify each time if the parking has a match among the safe ones on its list.
* The information needed to estimate the discount for each ride, will be available to the system for each car. The system can get the battery level/status, car location and passengers number through sensors mounted in it. Then, knowing those parameters, it will calculate the eventual discount for the user.
* The discount will be applied to the user’s account only once the ride is finished and the car locked. This because the system needs to know whether the car after the usage has been plugged to the power grid or not and the exact location where it has been left.
* If the user has not put in charge the car in 40 seconds after the selection of the option “end ride and recharge”, the system is discarding that option, locking the car and applying the “end ride” action only.
* Fair vehicle distribution is not depending on PowerEnJoy and is left to users. PowerEnJoy just take care of having all available cars fully charged
  1. ***Possible future implementations***
* The service could be extended to disable people by adding special vehicles to the company’s set of cars
* The service could be extended to bicycles
* The service could allow the exchange of vehicles between cities in which PowerEnJoy operates

1. ***Specific requirements***
   1. ***External interface requirements***
      1. ***User interfaces***
      2. ***Hardware interfaces***
      3. ***Software interfaces***
      4. ***Communications interfaces***
   2. ***Functional requirements***
   3. ***Nonfunctional requirements***
   4. ***Performance requirements***
   5. ***Scenarios***
   6. ***UML models***

Here below are provided the major UML diagrams that allow a concrete comprehension of how the system is going to operate.

* + 1. ***Use cases***



* + - 1. View main page

|  |  |
| --- | --- |
| *Actors* | Guest |
| *Goal* | Allow the visitor to collect information about PowerEnJoy and its service |
| *Input condition* | - |
| *Event flow* | 1. The guest accesses PowerEnJoy web site 2. The guest visits web pages and collects information about the service 3. The guest decides whether to register or leave the web site |
| *Output condition* | - |
| *Exceptions* | - |

* + - 1. Register

|  |  |
| --- | --- |
| *Actors* | Guest |
| *Goal* | Allow the visitor to register to PowerEnJoy by submitting a form containing his/her personal data and payment information |
| *Input condition* | - |
| *Event flow* | 1. The guest clicks “Register” in the web page 2. The guest fills and submits his/her personal data and credit card number to the system 3. The system verifies user’s data and releases a password to access PowerEnJoy infrastructures |
| *Output condition* | Guest is now a user and the system has provided him with a password to access the service |
| *Exceptions* | * User data are not valid * User credit card is not valid |

* + - 1. Login

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to fill his credentials and access the system |
| *Input condition* | - |
| *Event flow* | 1. The user clicks “Login” in the web page 2. The user fills in his/her credentials 3. The user presses the “Login” button |
| *Output condition* | The user is logged in |
| *Exceptions* | * Username is invalid * Password is invalid |

* + - 1. Edit own profile

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to edit his profile data. |
| *Input condition* | The user is logged in |
| *Event flow* | 1. The user choses “Edit profile” from his/her account options 2. The user modifies his/her personal data and presses “Confirm” 3. New data are processed by the system and, if correct, accepted |
| *Output condition* | User’s data have been modified |
| *Exceptions* | * Incorrect data inserted |

* + - 1. Find car location

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to find a car location |
| *Input condition* | The user is logged in |
| *Event flow* | 1. The user accesses the PowerEnJoy app 2. The user selects “Find a car” among all possible actions 3. The user submits the system his/her location. Location may be specified by using user’s current position or by entering a valid address 4. The system looks for cars in the nearby of the specified positions having the status set as “AVAILABLE” or “RECHARGING” and a battery level of at least 15%. |
| *Output condition* | User is provided with a map showing every usable car in the nearby of the specified position. Every found car is ensured to be available. |
| *Exceptions* | * Location entered doesn’t exist * Impossible to detect user’s position |

* + - 1. Reserve car

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to reserve a car for 1h time |
| *Input condition* | The user is logged in and has already queried the system to find a car |
| *Event flow* | 1. The user selects one of the cars proposed by the system 2. The user presses the button “Reserve” 3. The car status is changed to “BOOKED” |
| *Output condition* | The selected car is reserved for up to 1h and the user receives a PIN to access the car during that time |
| *Exceptions* | * The user is trying to reserve more cars for the same period |

* + - 1. Delete reservation

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to cancel a reservation |
| *Input condition* | User’s reservation has not expired |
| *Event flow* | 1. The user accesses the PowerEnJoy app 2. The user selects “Delete reservation” among all possible actions 3. The user presses the button “Delete” |
| *Output condition* | User’s last reservation is deleted |
| *Exceptions* | - |

* + - 1. Send unlock command to the system

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to unlock a reserved car |
| *Input condition* | The car has been reserved and the user took less than 1h to use it |
| *Event flow* | 1. The user reaches the car 2. The user unlocks the car using PowerEnJoy by pressing the button “Unlock car” 3. The user has access to the car 4. The system starts charging money to the user |
| *Output condition* | The car is unlocked |
| *Exceptions* | - |

* + - 1. Insert PIN

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to unlock a make usable the car |
| *Input condition* | The car has been unlocked using the PowerEnJoy app |
| *Event flow* | 1. The user is asked to evaluate the car and report damages 2. The user enters the PIN |
| *Output condition* | The car is unlocked and ready for a ride |
| *Exceptions* | * Invalid PIN inserted |

* + - 1. Set saving mode

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to get tips for saving money by applying virtuous behaviors |
| *Input condition* | The car is “ONROAD” |
| *Event flow* | 1. The user presses the button “Saving mode” 2. The user inserts his/her final destination 3. The system tells the user which could be useful tips to save money at the end of the ride. The user is not bound to that tips and doesn’t necessarily follow them. |
| *Output condition* | - |
| *Exceptions* | - |

* + - 1. Start engine

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow to user to start the engine and begin his/her ride |
| *Input condition* | The car is unlocked |
| *Event flow* | 1. The user turns the key of the car 2. The system changes the car status from “PARKING” to “ONROAD” |
| *Output condition* | The car is unlocked and the ride has started |
| Exceptions | * Mechanical fault |

* + - 1. Stop navigation

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow to user to stop his/her navigation for a temporary of definitive stop |
| *Input condition* | The car is “ONROAD” |
| *Event flow* | 1. The user chooses whether to park the car or to end his/her ride by clicking the appropriate button shown on the car screen 2. **A)** If the user chooses to do a temporary stop, the car status is changed to “PARKING” **B)** If the user chooses to end the ride, the status is changed to “AVAILABLE” **C)** If the user chooses to end the ride and recharge the car (s)he has 40 seconds, after the engine is turned off, to plug it in a recharge station. When a car is recharging the status is set to “BATTERYCHARGE”. If the user takes more than 40 seconds to plug in the car, the recharge option is discarded and the status is set to “AVAILABLE” 3. In cases **B** and **C**, a payment request is submitted to the system 4. The car is locked by the system as soon as the user: - gets off (cases 2 and 3) - takes more than 40 seconds to plug in the car in the recharge station - plugs in the car in the recharge station |
| *Output condition* | The car is parked, recharging or available |
| *Exceptions* | - |

* + - 1. Ask for assistance

|  |  |
| --- | --- |
| *Actors* | User |
| *Goal* | Allow the user to receive assistance in case of problems or technical issues |
| *Input condition* | - |
| *Event flow* | 1. The user presses the button “Ask for assistance” on the car screen 2. The user is show some FAQ 3. If no FAQ is useful, the user can be put in contact with a PowerEnJoy operator |
| *Output condition* | - |
| Exceptions | - |

* + 1. ***Class diagram***
    2. ***State diagrams***

1. ***Overall description***
   1. ***Alloy***
      1. ***Signatures***
      2. ***Facts***
      3. ***Assertions***
      4. ***Predicates***
      5. ***Alloy model***
      6. ***Generated world***
   2. ***Software and tools used***
   3. ***Team work***