

Version: 1.0.0  
Release date: 25-10-2016

Riccardo Cattaneo 873647

Fabio Chiusano 874294

PowerEnjoy

Summary

Introduction 3

Description of the given problem 3

Goals 3

Domain properties 4

Glossary 4

Text assumptions 5

Constraints 5

Regulatory policies 5

Hardware limitations 6

Interfaces to other applications (system boundaries) 6

Parallel operation 6

Proposed system 7

Identifying stakeholders 7

Reference documents 8

Actors identifying 9

Requirements 10

Functional requirements 10

Non-functional requirements 11

Scenario identifying 12

Scenario 1 12

Scenario 2 12

Scenario 3 12

Scenario 4 12

UML models 13

Use case diagram 13

Use case description 14

Class diagram 17

Sequence diagrams 18

Log-In 18

Change Payment Info 19

Show Safe Areas with Power Grid 20

Activity diagrams 21

User makes a reservation 21

State diagrams 22

Mobile app states 22

Car on-board software states 23

Alloy modeling 24

Model 24

Alloy result 27

World generated 27

Future development 28

Used tools 29

Hours of work 30

Changelog 31

Class notes on this project 32

Goals 32

General 32

# Introduction

## Description of the given problem

We will project the digital management system for PowerEnJoy™, which is a car-sharing service that exclusively employs electric cars.

The system, first, has to provide normally provided by car-sharing services such as the possibility for a new user to register and log in, to find locations of nearby available cars and to reserve them.

It has also to guarantee that a user who has used the service pays a fee that should be as fair as possible.

In order to strengthen the ecological mission of PowerEnJoy™, the system aims to incentivize virtuous behaviours of the users by adapting the final bill for every ride.

For example, if there have been at least three people on the car, or if the car has been left charging at special parking areas, the system has to apply a discount. Instead, if the car has been left far from a charging station with a low battery level, it has to apply a charge on the bill.

## Goals

* Users can see and select an available car close to them, or close to a specified address, and reserve it for up to one hour before they pick it up;
* Users can get in a car only if they are near it and they reserved it;
* Users should pay proportionally to minutes they have used the car, and they should see in real time the amount of the bill;
* Users could register to the system and have their personal area;
* Virtuous behaviours by users should be incentivized.

In particular, the system could achieve that by charging user, on the last ride:

* + 10% less if they share their trip with at least other two passengers;
  + 20% less if the car is left with at least 50% of battery level;
  + 30% less if the car is left plugged in at special parking areas;
  + 30% more if the car is left at more than 3km from the nearest power grid station with less than 30% of battery level.

## 

## Domain properties

We suppose that these conditions hold in the analysed world:

* All the users have a device connected to the Internet with a GPS built in;
* All the electric cars have an on-board computer that allows execution of Java software;
* All the electric cars have a GPS to indicate their actual position, that cannot be turned off, and a sensor for every seat which detect the presence of a passenger;
* All the electric cars have Internet connection, that is always working and can’t be turned off;
* All the electric cars on-board computer can’t be turned off or sabotaged;
* GPS position is always accurate;
* All the cars can carry a maximum of 4 passengers;
* A car can be in only one zone at the same time and this is the actual real zone;
* A car can’t be used by multiple users simultaneously;
* In a special parking area with power grid stations there’s always space for a car to be plugged in;
* Users behave politely and have no intention of cheating;
* The payment information provided by the users are always valid;
* The company never reach the limit of requests per day for the external services.

## Glossary

* Management System: the management system of the car share service;
* User: a user is a person already registered in the system, so that has a profile, and sometimes is interested to reserve and use a car;
* Car: for “car”, “electric vehicle” or “electric car” we mean an electric car involved in PowerEnJoy™, ready to be reserved and used;
* Ride: usage of a car, by one user, that has to pay it with a bill. It starts when the user picks up a car after a reservation is made and it ends when the user leaves the car in a safe area;
* Reservation: is the ability of a user to reserve a car at most one hour prior to the pick up;
* Bill: compensation to be paid for a ride by the user;
* Guest: a guest is a person that probably for the first time accesses the system or that has not already signed up;
* Safe Area: area where a user can leave the car he’s renting;
* Special Safe Area or “Safe Area with power grid station”: Safe Area where the user can plug the car into the power grid station in order to get a discount on the ride;
* Battery level: how much in percentage the battery is charged;
* Passengers: people that are in a car during a ride. The user that drives the car is included in the passengers count.
* Search-on-map service: a service that provides the possibility to search locations and see distances on a digital map.
* Payment information: everything that the external payment handler needs in order to run its services.

## Text assumptions

There are few things that we will assume about the specification document:

* If the user gets close to the car he rented and the car unlocks itself, then the car is also able to lock itself if the user distances himself.
* Power grid maintenance is always ok.
* The payment for a ride is carried out when the user quits the car.
* The payment for the reservation fee is carried out as soon as the reservation hour expires.
* Nothing is said about the payment handler, therefore we decided to use an external payment handler and we assume that it’s able to manage the cases where there’s something wrong with a payment (e.g. the user has no money).
* The discount for having at least two other passengers onto the car is applied only if those passengers are in the car before the engine ignites.
* It is said that “A user that reaches a reserved car must be able to tell the system she’s nearby” and we assume that GPS is not necessary. Therefore the user can use the system portal to explicitly say that she’s nearby, without satellite connections.

## Constraints

### Regulatory policies

The Management System must ask the users the permission to get their position and to manage sensible data (position). The user is not forces to accept them in order to use the service because everything can be done without satellite connections:

* He/she can search near available cars explicitly writing his/her current location.
* He/she can tell the system that he/she’s near the rented car in order to unlock it.

### Hardware limitations

3G/4G connections is required since the system must be usable on top of a platform built for mobile systems. GPS is not required since every operation can be made without it. Users must have enough storage space to install the PowerEnjoy app. The server must be able to run a web server application, for instance Apache Web Server.

|  |  |
| --- | --- |
| **System component** | **Hardware limitations** |
| Client mobile device | * 3G/4G connections * GPS connection (optional) * Enough memory for the app package |
| Server | * Internet connection * Should run an OS that allows Apache Web Server to run |
| Car on-board computer | * Internet connection * GPS connection * should run Java software |

### Interfaces to other applications (system boundaries)

The system relies on an external payment handler (e.g. Paypal) and on an external search-on-map service (e.g. Google Maps). In both cases, we must register our application to the service provider in order to take advantage of it. Sometimes it’s also possible to buy a pro licence for these services so that the cost for the service with a lot of requests if smaller. However this step can be made once the managers find it necessary as it doesn’t compromize the scalability of our system.

### Parallel operation

The server supports of course parallel operations from different clients . In the future it will be possible to have multiple redundant logic layers distributed on more than one server.

## Proposed system

We propose a web app that will give users a compfortable way to use our service.

The web app will be available to all the major mobile operating systems and will be developed in a way to communicate with the PowerEnjoy servers. We assume that at the moment the user base is small and therefore one server will be enough to serve everyone. However, due to the fact that the use base can become wider in a small amount of time, everything will be made in a way that implementing redundant arrays of them is easy. The database will stay on the same machine of the web server at the moment, but nothing will compromise the possibility to move it on another machine in the future. Of course the electric cars must be able to communicate with the server, so they must be provided with an Internet connection and an on-board computer that must be able to run Java sofware. However, the car is only an agent in our system and therefore all the business logic will lay on the server.

The proposed system can be summarized in an high-level way in the following image:



As for the communications, the server will expose a RESTful API to the mobile app and the electric car.

TODO: HERE WE SHOULD PUT A GRAPH WITH IMAGES FROM THE IMAGES FOLDER WHERE WE SHOW REQUESTS AND RESPONSES BETWEEN THE COMPONENTS.

## Identifying stakeholders

Trivial stakeholders are:

* The company that serves the PowerEnjoy service (managers, employers, …).
* The customers of PowerEnjoy.
* Possible creditors/suppliers and shareholders, for example:
  + The car company that provides the cars that runs the PowerEnjoy service.
  + The company that provides all the hardware needed for the system.

There are a lot of entities that incentivize the use of electric vehicles, since they are less harmful to the environment:

* The government.
* The city in which the service is active (the society).

External service providers and suppliers are stakeholders too. They are:

* Paypal.
* Google Maps.

And Eventually we are stakeholders too, as software developers/engineers.

|  |  |
| --- | --- |
| **Stakeholder** | **Needs** |
| PowerEnjoy | Provide the car sharing ecological service in an efficient, usable, reliable way to customers. |
| Customers | Use the service usefully. |
| Possible creditors and shareholders | Get profit. |
| Government and society | More sustainable and ecological viability. |
| External service providers | Get profit. |
| Us | Get profit. |

## Reference documents

* Specification Document: Assignments AA 2016-2017.pdf
* IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications
* Example documents:
  + RASD sample from Oct. 20 lecture.pdf

# Actors identifying

The main actor of PowerEnJoy™ system is:

* User: a person that has already registered and so has provided his personal information and payment method.

There is also another possible actor:

* Guest: a person that has not registered and can only perform basic functionalities such as looking for where safe areas are.

# Requirements

Assuming that the domain properties stipulated hold, and, in order to fulﬁll the goals listed before, the following requirements can be derived.

## Functional requirements

* [G1] *Users could see and select an available car close to him, or close to a specified address, and reserve it for up to one hour before they pick it up:*
* The system has to detect if a car is parked in a Safe Area with fully charged battery;
* The system has to detect car position and display it on a map;
* The system has to be able to identify the location of a user through his/her GPS, if he/she gives the consent;
* The system has to provide a list of available cars close to a given address;
* The system has to give the possibility to reserve a car at most by one user at a time;
* The system has to remove the reservation for a car after one hour if the user has not picked it up;
* The system has to apply a fee of 1€ if the reservation has expired;
* [G2] *Users could get in a car only if they are beside it and they reserved it:*
* The system has to be able to identify the location of the user and of the car;
* The system has to unlock the car if the position of the user is really close to the one of the car;
* [G3] *Users should pay proportionally to minutes they have used the car, and they should see in real time the amount of the bill:*
* The system has to reset trip information when a user get on the car
* The system has to be able to understand when the car engine ignites;
* The system has to start charging the user when the car engine ignites;
* The system has to display the current charge;
* The system has to identify when a car is parked in a safe area;
* The system has to identify when there is no one sit in the driver’s seat;
* The system has to stop charging the user when the car is parked in a safe area and there is no one sat in the driver’s seat;
* [G4] *Users could register to the system and have their personal area:*
* The system has to provide log-in functionalities to the users;
* The system has to provide sing-up form to users:
  + The system has to check that there are not two users with the same username;
  + The system has to store the password and personal information of every user;
  + The system has to provide the possibility to enter a payment method;
  + The system has to check if the payment method provided by the user is valid and usable;
* The system has to provide the possibility to change personal information or payment methods even after the registration;
* [G5] *Virtuous behaviours by users should be incentivized:*
  + The system has to apply a discount of 10% on the final bill if there were at least three passengers on the last ride:
  + The system has to identify and store how many passengers there were on the car in the last ride;
* The system has to apply a discount of 20% on the final bill if the car is left with at least 50% of battery level:
  + The system has to be able to identify the battery level of the car;
* The system has to apply a discount of 30% on the final bill if the car is left plugged-in in a Special Safe Area:
  + The system has to identify if the car is plugged-in;
* The system has to apply an extra-charge of 30% on the final bill if the car is left at least 3Km from the nearest Special Safe Area and the battery level is less than 30%:
  + The system has to be able to calculate the distance between the actual position of the car and the nearest Special Safe Area

## Non-functional requirements

* The system has to be interoperable with the payment-handler FaccardoPal™ in order to provide effectively payments function;
* The system has to be interoperable with the search-on-map service Google Maps;
* The system has to be available 24 hours per day, 7 days per week, the same as the time required to develop effectively this fucking document;
* The system has to be available at least as an Android app;
* Modified data about availability of cars in a database have to be updated for all users accessing it within 2 seconds.
* Users’ passwords have to be encrypted using SHA256 algorithm.

# Scenario identifying

## Scenario 1

Nick and his three best friends want to go out at night, but public transport is not serviceable at those hours. They do not want to spend a large amount of money, therefore they decide to take advantage of PowerEnjoy service and its discount.  
Nick decides to plan the trip in order to achieve the maximum discount possible, that is by leaving the car in the Safe Area closest to the pub they want to go.

He opens the PowerEnjoy mobile app about one hour before going out, makes a reservation for the car and finds out the best place where to leave the car.

He and his friends can enjoy the night without spend too much.

## Scenario 2

Riccardo and Fabio are two univerity students. Riccardo is excited about PowerEnjoy since he tried it for the first time and talks about it to his friend Fabio after the lesson. Fabio, who has always been a fond of environment-friendly companies, can’t wait to try PowerEnjoy electric cars and immediatly downloads the app of the service. He decides to go home with an electric car, so he registers to the system with his credentials and payment information and makes a reservation for a car in the nearest possible point. Fabio reaches the car and drives it to the closest park to his home. The next day, Riccardo tells Fabio that he should have left the car in one of the special parking of PowerEnjoy with power grid stations so that he would have had a reduction on the fee. Fabio feels sad.

## Scenario 3

Agnese has just finished shopping. She bought a lot of things and she doesn’t have the strength to bring home all her seven full shopping bags by hands. Agnese tried to call home to see if someone was there to help her but nobody answered. Therefore, she decides to use her favorite car-sharing app: PowerEnjoy. She takes her phone and looks for a car near her position. Unfortunately, the nearest car is farther than her house. Next time Agnese will buy less things, or will go shopping by car.

## Scenario 4

Daniel has to meet his friend Adriana in one hour. Since Daniel has no cars, he makes a reservation for a PowerEnjoy electric car. He finishes seeing some episodes of his favourite telefilm and reaches the car he rented. However, the car doesn’t open because Daniel saw to many episodes and more than an hour has passed. So he makes another reservation for a car that fortunately is available in that parking and takes it. The system charges him an extra fee.

# UML models

## Use case diagram

## Macintosh HD:Users:fabiochiusano:Desktop:magistrale primo anno:Software engineering 2:project:PowerEnjoy:UML:UserCase:usecase_image.jpg

## Use case description

|  |  |
| --- | --- |
| Name | Log-in |
| Actors | Guest |
| Entry conditions | Guest has previously signed up to the system. |
| Flow of events | * The Guest opens the application of PowerEnJoy™ on his/her device; * The system shows him the login page; * The Guest enters his e-mail address and password in the input form provided; * The Guest clicks the button “log in”. * The system shows the user’s Personal Page. |
| Exit conditions | Guest gets a confirmation message and is redirected to his/her personal page. He/she is now acknowledged by the system as a User. |
| Exceptions | The username and password provided by the Guest are not correct.  The system notifies him/her that he/she has made an error and allows the guest to input his/her username and password again. |

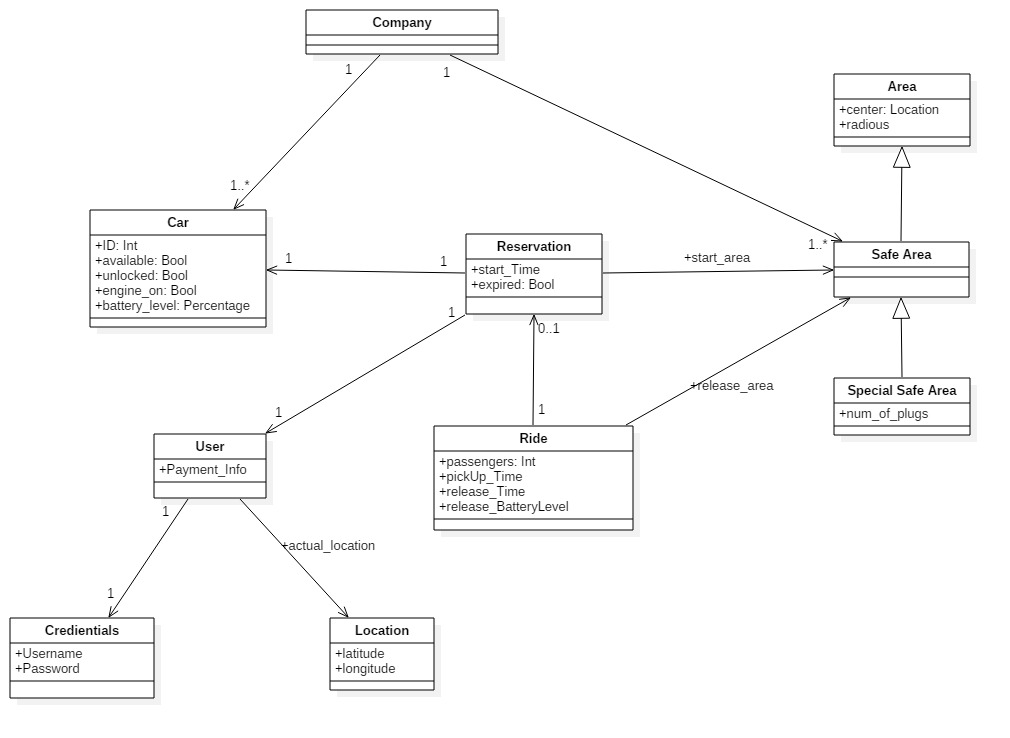
|  |  |
| --- | --- |
| Name | User changes payment info |
| Actors | User |
| Entry conditions | User has already signed in to the system. |
| Flow of events | * The user clicks on “Change default Payment” button; * The system redirects the User to a form where he/she has to provide all the information about the new payment method; * The User sets Payment Data and clicks on Submit button; * The systems checks through the Payment Handler module if information provided are valid; |
| Exit conditions | When Payment Handler confirms the Payment Method provided, the system shows a confirmation message and redirects the User to his Personal Page. |
| Exceptions | The Payment Handler rejects the Payment Method provided.  The system shows an error message and redirects the User to his Personal Page. |

|  |  |
| --- | --- |
| Name | Show close cars on map through GPS |
| Actors | Client |
| Entry conditions | User has already signed in to the system. |
| Flow of events | * User clicks on “Show Map” button; * The system shows an updated map on the center of the page; * User clicks on “Find me by GPS” * The system acquires user’s GPS location and shows it by adding a colored dot on the map; * User puts a tick on “Show close available cars”; * The system adds on the map in the right position a little car image for every available car within a default range of distance between location of cars and location of the user; * User moves a slider in order to adjust this distance; * The system adjusts objects on the map according to user’s choice; |
| Exit conditions | There are no exit conditions. |
| Exceptions | There are no cars or Safe Areas within the above distance.  The system displays a popup message to inform the user that PowerEnJoy™ service is not present in the area. |

|  |  |
| --- | --- |
| Name | Show close Safe Areas with Power Grid on map |
| Actors | Client |
| Entry conditions | User has already signed in to the system. |
| Flow of events | * User clicks on “Show Map” button; * The system shows an updated map on the center of the page; * User fills the address field with a desired address; * The system shows a colored dot on the map in the provided address; * User puts a tick on “Show close Safe Areas with Power Grid”; * The system adds on the map in the right position an identifier for every available car within a default range of distance between location of cars and location of the user; |
| Exit conditions | There are no exit conditions. |
| Exceptions | There are no cars or Safe Areas within the above distance.  The system displays a popup message to inform the user that PowerEnJoy™ service is not present in the area. |

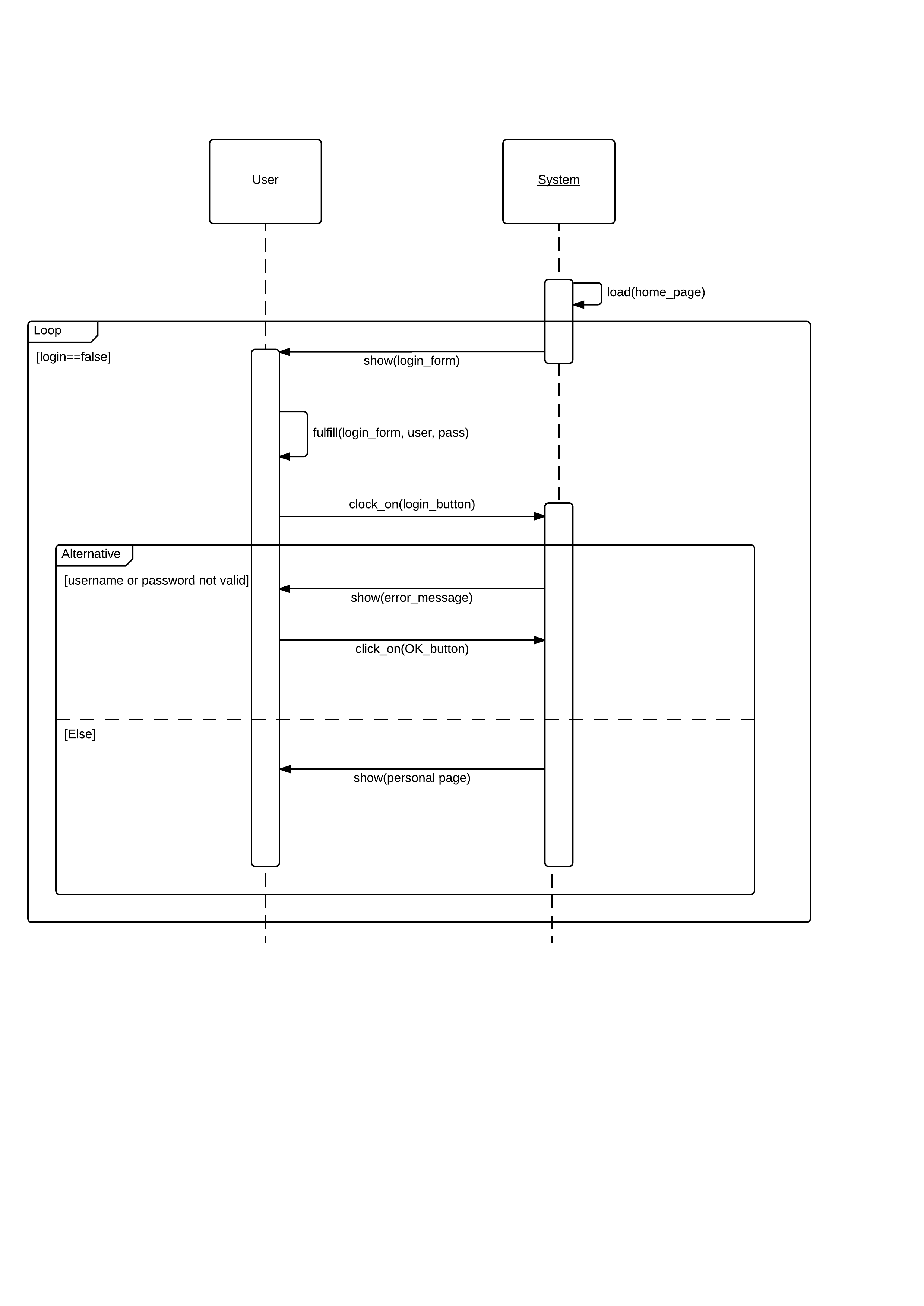
|  |  |
| --- | --- |
| Name | Make a reservation |
| Actors | Client |
| Entry conditions | User has already signed in to the system and is viewing available cars that may be of interest to him. |
| Flow of events | * User clicks on the car that best meets his/her needs; * The system shows a popup in which there are also information about the car; * User clicks on “Reserve this car”; * The system updates information of the car and marks it as reserved; * The system shows an information message about remaining time to pick-up the car. |
| Exit conditions | The User clicks OK, the system redirects him/her to the Personal Area. |
| Exceptions | Another user has reserved the car shortly before the user did. The system shows a message and redirects him/her to the map, displaying currently available cars. |

## Class diagram



## Sequence diagrams

### Log-In



### Change Payment Info

### Show Safe Areas with Power Grid

## Activity diagrams

### User makes a reservation



## State diagrams

### Mobile app states



### Car on-board software states



# Alloy modeling

## Model

open util/boolean

sig Company {

cars: set Car,

safe\_areas: set Safe\_Area

}

sig Car {

id: Int,

available: Bool,

location: Location,

unlocked: Bool,

engine\_on: Bool,

battery\_level: Percentage

}{

id>0

}

sig Percentage {

level: Int

} {

level >= 0,

level <= 100

}

sig User {

credential: Credential,

password: Password,

payment\_info: Payment\_Info,

location: Location

}

sig Credential {}

sig Password {}

sig Payment\_Info {}

sig Location {

latitude: Int,

longitude: Int

} {

latitude >= -90

latitude <= 90

longitude >= -180

longitude <= 180

}

sig Reservation {

user: User,

car: Car,

start\_area: Safe\_Area,

start\_time: Time,

expired: Bool

}

sig Time {

year: Int,

day: Int,

hours: Int,

minutes: Int

} {

year >= current\_year

year <= 2030 // For some reasons we limit the maximum time interval between the reservation timestamp and its start\_time.

day >= 0

day < 365

hours >= 0

hours < 24

minutes >= 0

minutes < 60

}

sig Ride {

reservation: Reservation,

passengers: Int, // User is included in passengers.

pickup\_time: Time,

release\_time: lone Time,

release\_battery\_level: lone Percentage,

release\_area: lone Safe\_Area

} {

passengers >= 1

passengers <= 4 // We assume that all cars are “small” and equally capient.

}

// General area (e.g. area whose center is the user location, used to find the closest cars).

sig Area {

center: Location,

radius: Int

} {

radius > 0

}

// Area where I can park the car.

sig Safe\_Area extends Area {}

// Area where I can park the car and charge it.

sig Special\_Safe\_Area extends Safe\_Area {}

fact noUsersWithSameCredentials {

all u1, u2: User | (u1 != u2) => u1.credential != u2.credential

}

fact noCarsWithSameIds {

all c1, c2: Car | (c1 != c2) => c1.id != c2.id

}

/\*

TODO:

FACTS:

- A user can have only one reservation each hour.

- A user can be on a ride at a time.

- Company.cars = Car.

- Company.safe\_areas = Safe\_Area.

DON’T WANT TO THINK NOW:

- car.available => the car has a reservation for at most one hour ago or there aren’t any rides whose release\_time is not set yet.

- car.location is in a Safe\_Area in which the last ride with such car finished.

- user.location is near car.location and there’s a reservation with them => car.unlocked = true.

- user.location is not near car.location and there’s a reservation with them and car.unlocked == true => car.unlocked = false. // We want to lock the car if the user goes away because he reminds that he left the gas open!

- car.engine\_on => ride.charge ++.

- after an hour from the reservation.start\_time there are no rides with such reservation => reservation.expired == true and reservation.charge += fee.

- ride.pickup\_time > ride.reservation.start\_time.

- ride.release\_time is empty or ride.release\_time[0] > ride.pickup\_time.

\*/

## Alloy result

## World generated

# Future development

There are a lot of possible improvements in the system to be:

* Accident management.
* Push notifications or SMS for reservation timer alerts.
* Add a web site.

# Used tools

# Hours of work

# Changelog

# Class notes on this project

## Goals

* Should we specify the rewards for virtuous behaviours? General goal: encourage the user to behave well. Rewards and penalties can be seen as subgoals. Therefore it’s NOT necessary to write all rewards and penalties.

## General

* Operators that charge the cars, move the cars (only for point ‘e’).
* Credit card payments done with an external service (that’s a boundary of our system). External services are always system boundaries and external actors in the use cases.
* We must manage all cases that are not in the domain properties/text assumptions.
* We should decide between managing accidents or not (I hope not).
* The payment is always successful => Domain property.