Software Engineering 2

Design document

POLITECNICO
DI MILANO

Version 1.0

Alexsander H. Baldin 876273
Jesús Bermejo herrero 878667

Table of Contents

1. Introduction 3
    1.1. Purpose..... 3
    1.2. Scope..... 3
    1.3. Reference documents..... 3
    1.4. Document structure..... 4
2. Architectural design 5
    2.1. Overview..... 5
    2.2. High level components and their interaction..... 5
    2.3. Component view..... 6
    2.4. Deploying view..... 7
    2.5. Runtime view..... 8
    2.6. Component interfaces..... 10
    2.7. Selected architectural styles and patterns..... 10
3. Algorithm design 11
4. User interface design 12
5. Reference..... 12
    5.1. Used tools..... 12
6. Hours of work..... 13

1. Introduction

1.1 Purpose

The objective of this document is to detail the technical specifications of the system PowerEnJoy complementing the information already presented in the RASD. Since this is a more technical document, it is focused for the developers of the system, specifying certain elements like architecture, design patterns and runtime behavior.

1.2 Scope

The main target of the system is the people interested in renting an electric car, this clients will need to register in the system, inserting their personal information and the payment method, so the charges will be pay automatically.
All the interaction with the software will be done via mobile application, using the GPS position or typing manually the location in which the system will search for available car for use.
The system is supposed to work as any other car sharing softwre available, but using only eletric car spread all over the city, chatging the user for the time he used the car and providing some optimizations such as having discounts in case the car was used to transport more than three people and if the car is left with the battery above 50%.
The cars will have to communicate with the application as well, sending information from its sensor, like how much people are in the car, the status of the battery, and any other information necessary.
The user will be able to reserve an available car for up to one hour, and if he is not able to reach the car in this time, the car will be available for other users again, but if the client manages to get to the car, he will be able to unlock the car through the application and start using it for as long as he wants, charging him until he leaves the car in a safe area.

1.3 Reference Documents

- Specification Document: Assignments AA 2016-2017.pdf
- IEEE standard on requirement engineering.pdf
- Examples documents:
    - SAMPLE DESIGN DELIVERABLE DISCUSSED ON NOV. 2.pdf

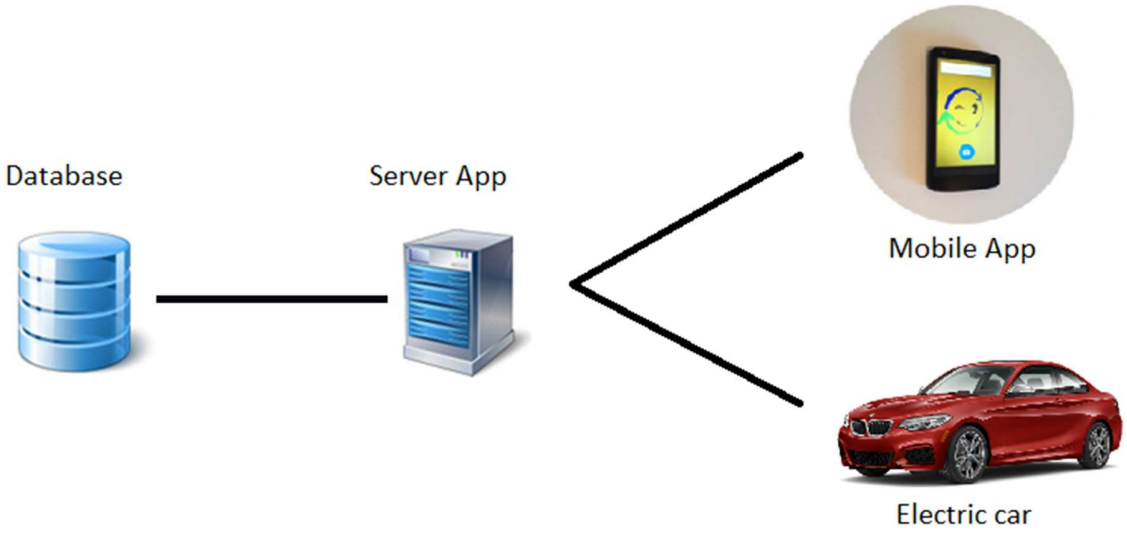
1.4 Document Structure

1. **Introduction**: the section id responsible for the presentation of the design document, showing its objectives and structure to implement the information that is available in the RASD.
2. **Architecture Design**: presents all the different views and interfaces that will be used in the project, so it is divided in various sub-sections:
    a. **Overview**: present a summary of the next sections eplaning how they are divided;
    b. **High level components and their interaction**: presents the main components and the interactions of the system;
    c. **Component view**: presents more details on the components of the system;
    d. **Deploying view**: presents the component tha will have to be deployed to make the application functional;
    e. **Runtime view**: presents the various tasks that the system will have to manage;
    f. **Component interfaces**: presents the interfaces between the components;
    g. **Architetural styles and patterns**: presents the patterns and styles chosen for the project;
3. **Algorithm Design**: presents some algorithm with pseudo code of the most critical parts of the system;
4. **User Interface design**: presents the mockups of the system showing the usage of the system;
5. **Requirements Traceability**: presents the connections between this document and the RASD;
6. **Effort Spent**: presents the time spent to develop this document;
7. **References**

2. Architecture design

2.1 Overview

The system PowerEnJoy will be divided in a simple three tier architecture, separated in GUI, application and database:



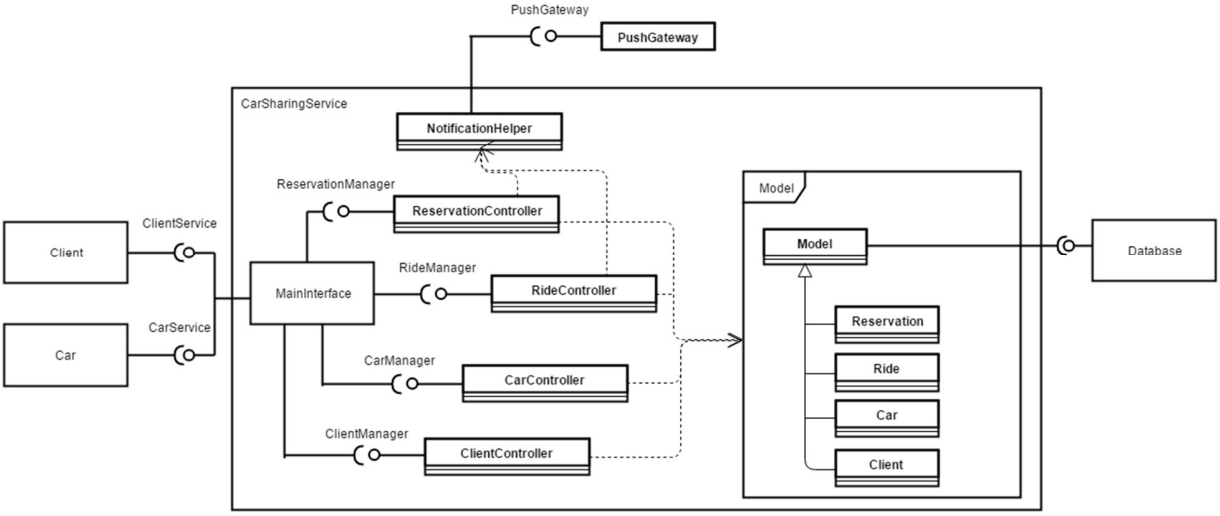
The application server will be responsible to receive the informations from the clients and from the cars, having one different interface for each, and storing this data in the database for further usage.
The car can be considered as a simple collection of sensors, and motors making certain actions, like unlock the door, that will have no logic and only sends and receives informations to the server.
The mobile application have a dynamic GUI that will be able to operate in any Android or Apple device, to englobe the biggest portion of the current market of smartphones.

2.2 High level components and their interaction

The high level components in this projects are the central system, the client, the car and the database. The central will be a singleton, that can have only one instance, and will receive informations from the clients and from the cars and send the appropriate messages back to them, saving the necessary info in the database. The client will be the one starting the

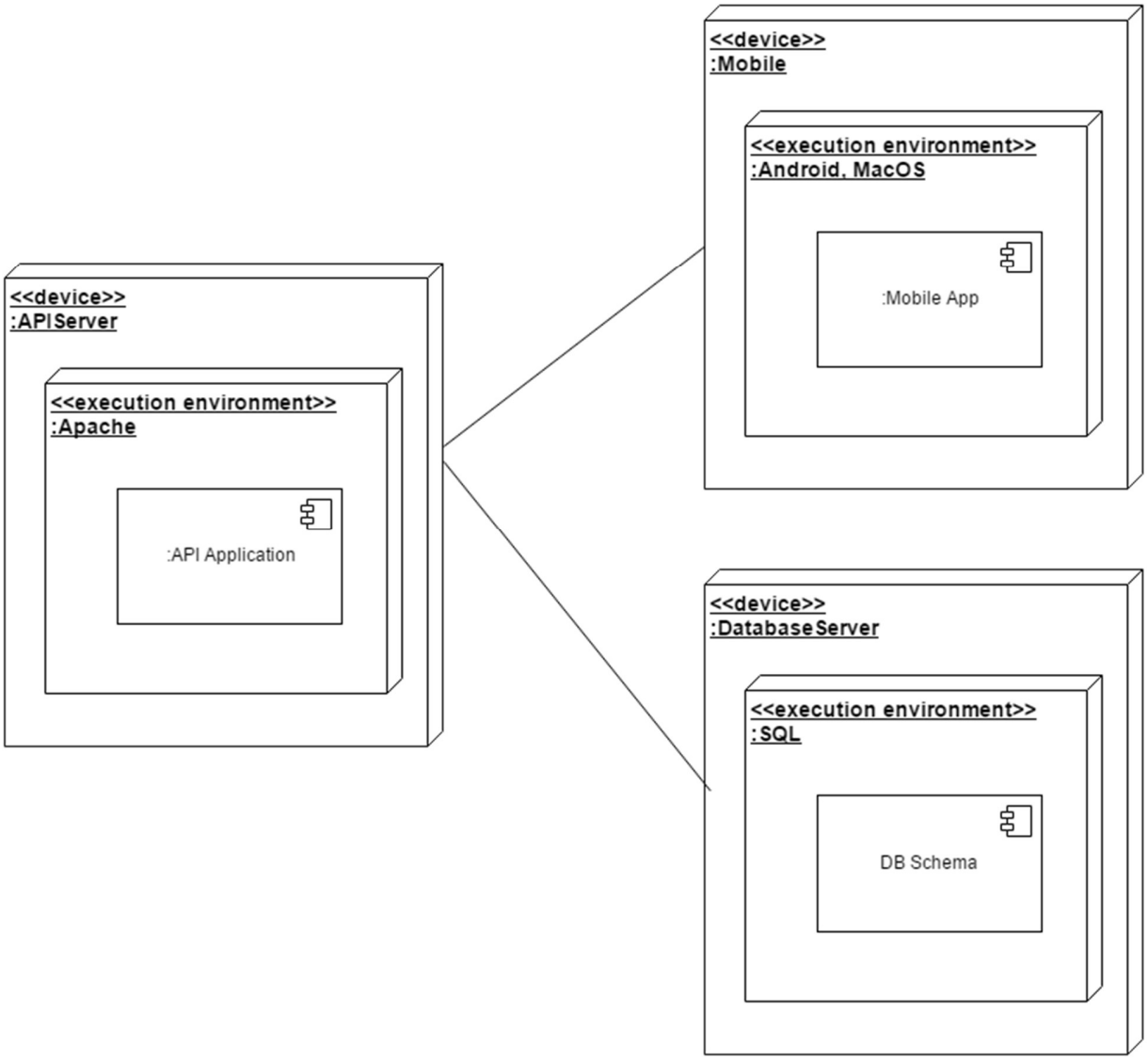
communication with the central through the login or the creation of an account. The messages sent from the client to the central will always be synchronous, so the central can tell if the client can advance on his tasks or not. The communication with the car is a lot different, when the client is able to reach the car in the reserved time, the central sends a message for the car to open its door and from this point on, the car will send messages to the central informing the amount of battery left, how many people are in the car and its location, until it is left in a safe area and the central is able to charge the client for the right amount, all this communication can be done assynchronously, until the end of the ride. There is no direct communication between the car and the client, every message has to pass by the central. And the last one is the communication between the central and the database, where the central sends synchronous messages to the database askd for data or inserting new registers.

2.3 Component view

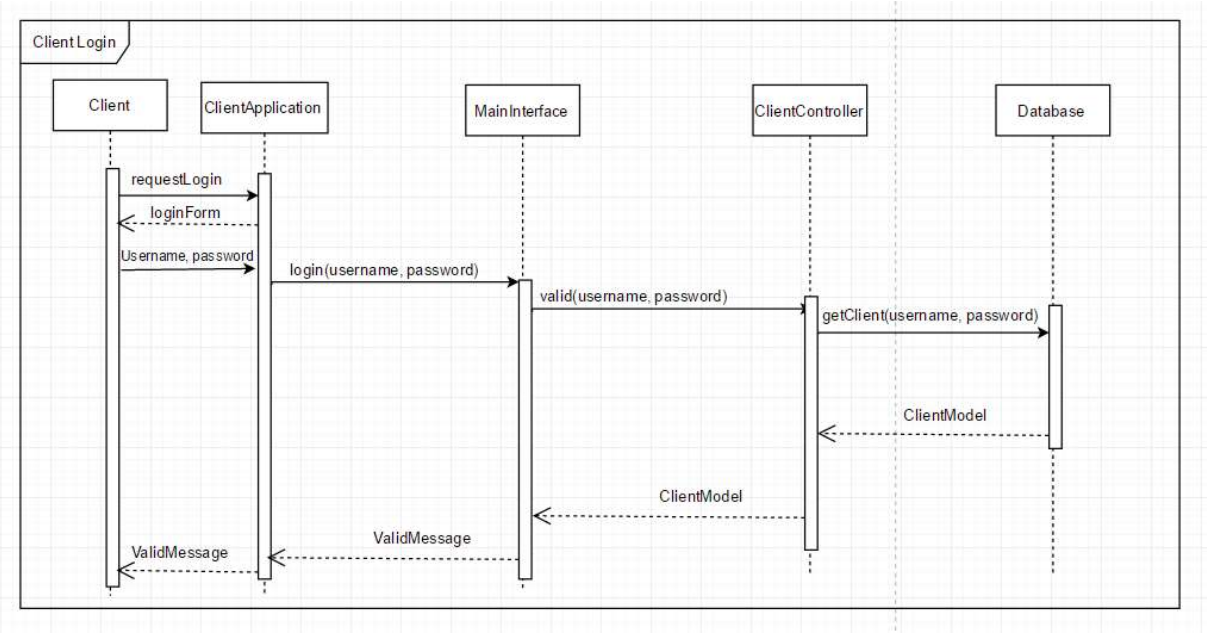


The external devices will be connect to the applications's main interface where the client will need to be authenticated. The car informations will also be receive through the router and the controller will be responsible for the management of this information and the creation of the model that will be stored in the database. The push notification will be used to send messages to the client informing the status of his reservetion and the situation of the ride.

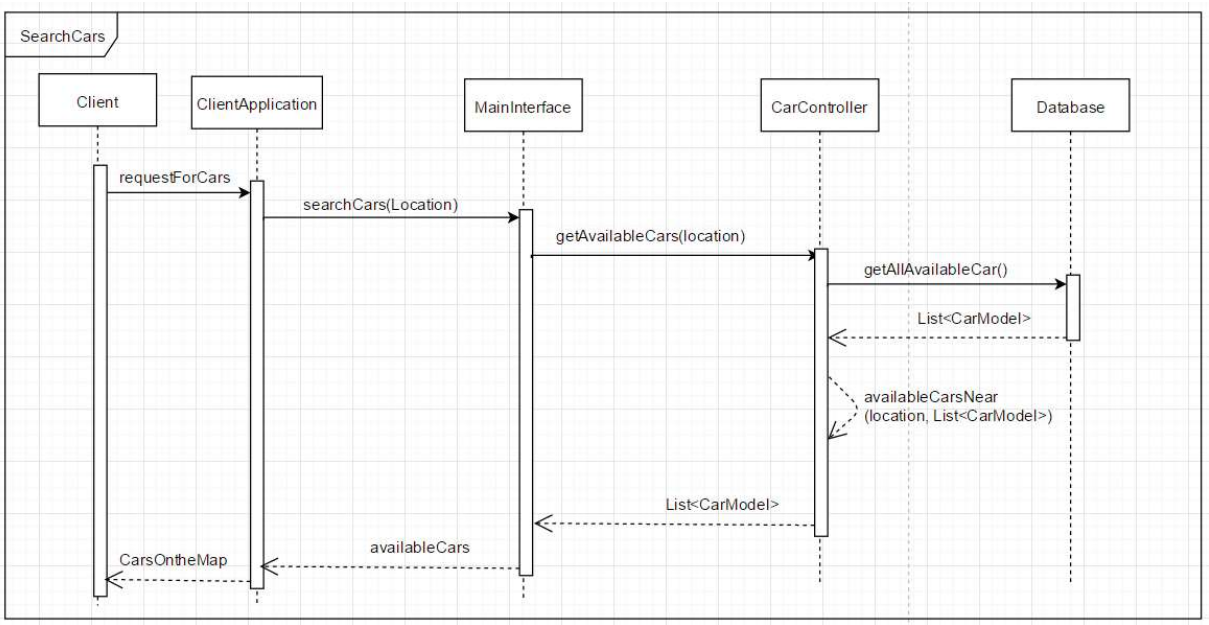
2.4 Deploying view



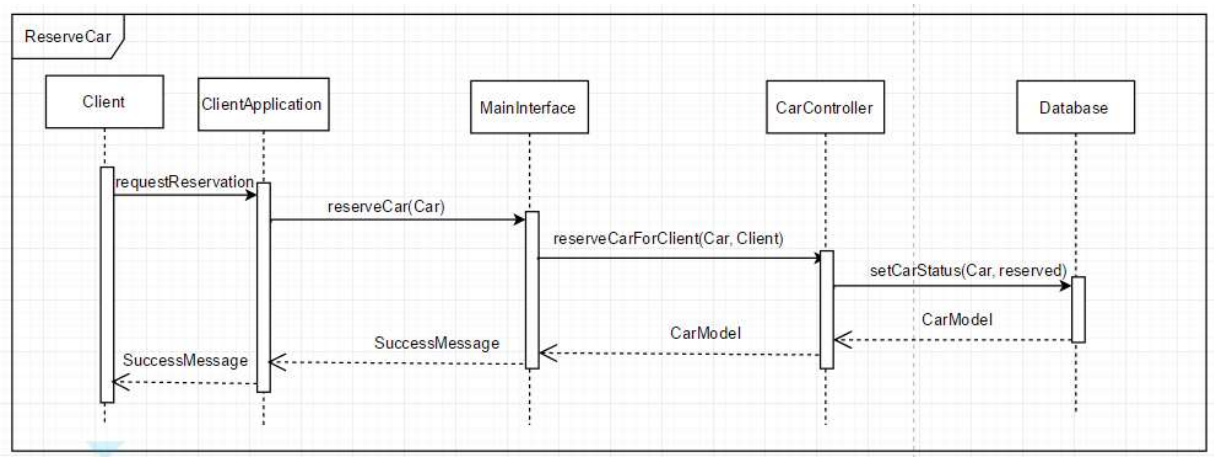
2.5 Runtime View



The sequence diagrams above describes the login of a client, where it is possible to see that once the data has been inputted, the mainInterface receives this info and sends it to the ClientController that checks if there is a valid client with those credentials in the DataBase, returning a Valid Message to the client and letting him enter the system.

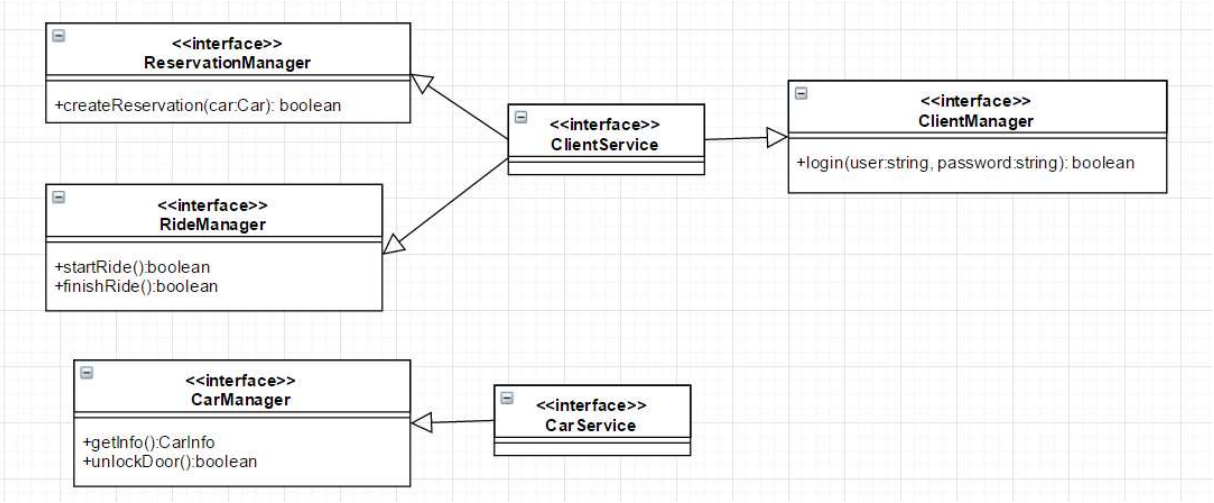


The sequence diagram above shows the search for available cars near a location passed through the mobile application via GPS or inputted by the user. The mainInterface sends the request for the carController that access the database to get the available cars and them selects the ones nears the location to shows them in the Client's map.



This sequence represents the process to reserve a selected car by the client, sending the request to change the status of the car to reserved in the Database.

2.6 Component interfaces



2.7 Selected architectural styles and patterns

The application will be developed using a cross platform, making it possible to be used in android and iphones, and using the MVC (Model-View-Controller) pattern that makes it easier to work with databases and organize the code.
The communication between the app and the server will consist of a simple client-server architecture, while the communication with the car won't have any logic on the cars end, only reading its sensor and sending comands like unlock the door.
The database used by the application will be an relational database to access any information of the entities involved in the project.

3. Algorithm design

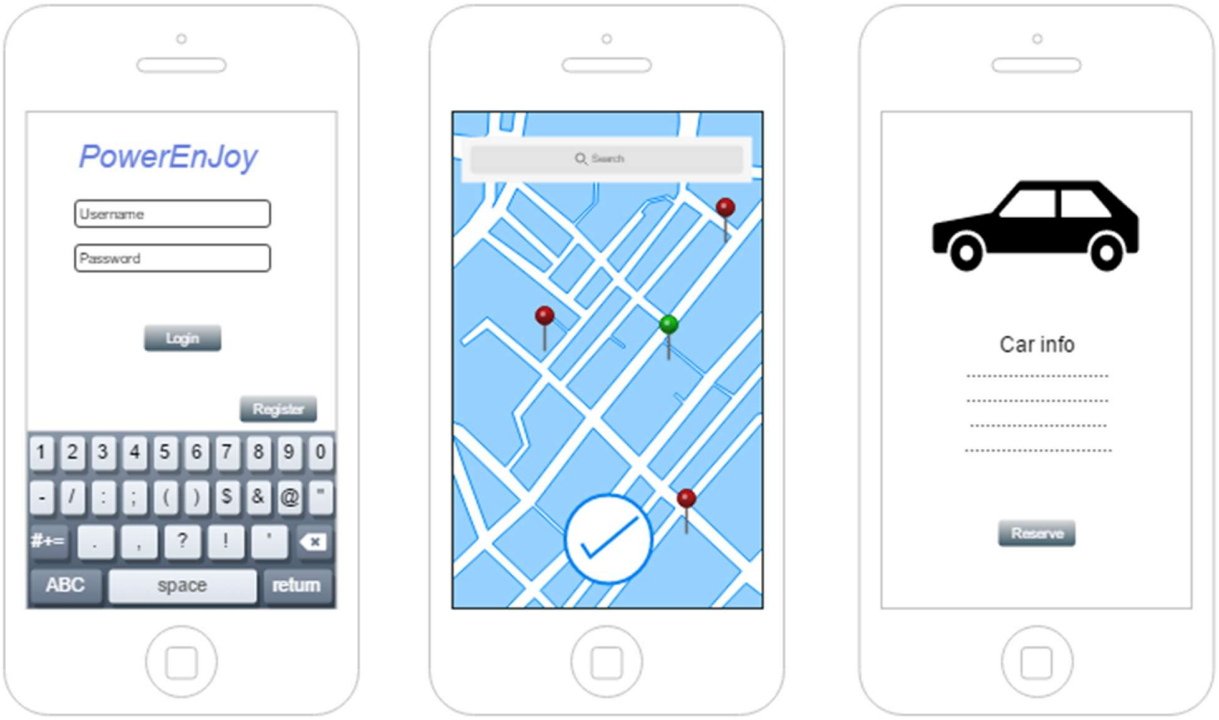
In this section, some code will be shown to represent the main algorithms used in the project.

Fare calculation

```
1   double PRICE_MINUTE = 0.5;
2
3   double rideFare(double rideTime, double battery, int nPeople) {
4       double finalPrice = rideTime * PRICE_MINUTE;
5       if (nPeople > 2)
6           finalPrice = finalPrice * 0.9;
7       if (battery > 0.5)
8           finalPrice = finalPrice * 0.8;
9       return finalPrice;
10  }
```

4. User Interface design

Mockups



The first screen represents the start screen of the mobile application, where it's possible to log in to the system or register a new account.

The second one is the map with the available cars nearby and the option to go to the car's page to see the details.

The last is the car's page, a picture of the car available in that location will be visible and other relevant information about it.

5. References

The following tools were used:

- Text editor Google Drive.
- Draw.io
- Github

6. Hour of work

Alexsander H. Baldin

3/12/16 → 2h
4/12/16 → 4h
8/12/16 → 4h
9/12/16 → 6h
10/12/16 → 8h
11/12/16 → 8h