



POLITECNICO DI MILANO  
MSc COMPUTER SCIENCE AND ENGINEERING

SOFTWARE ENGINEERING 2  
ACADEMIC YEAR 2016-2017

---

## Design Document

*PowerEnJoy*

---

*Authors:*

Melloni Giulio 876279

Renzi Marco 878269

Testa Filippo 875456

*Reference Professor:*

MOTTOLA Luca

*Release Date: December 11<sup>th</sup>, 2016*

*Version 1.0*

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Purpose . . . . .	1
1.2	Scope . . . . .	1
1.3	Definitions, Acronyms and Abbreviations . . . . .	2
1.4	Reference Documents . . . . .	3
1.5	Document Structure . . . . .	3
<b>2</b>	<b>Architectural Design</b>	<b>4</b>
2.1	Overview . . . . .	4
2.2	Component View . . . . .	4
2.3	Deployment View . . . . .	4
2.4	Runtime View . . . . .	4
2.5	Component interfaces . . . . .	4
2.6	Selected Architectural Styles and Patterns . . . . .	4
2.7	Other Design Decisions . . . . .	4
2.7.1	ER Diagram . . . . .	4
<b>3</b>	<b>Algorithm Design</b>	<b>6</b>
<b>4</b>	<b>User Interface Design</b>	<b>7</b>
<b>5</b>	<b>Requirements Traceability</b>	<b>8</b>
<b>6</b>	<b>Effort Spent</b>	<b>9</b>
<b>7</b>	<b>References</b>	<b>10</b>
7.1	Tools . . . . .	10

# List of Figures

2.1 ER Diagram . . . . . 5

# 1 | Introduction

## 1.1 Purpose

Analyzing the system-to-be in a more detailed way, it is possible to better define the functionalities that the system should provide to the users and the employees.

The user has the possibility to register to get a personal account on *PowerEnJoy*, which is by the way mandatory to access to all the functionalities provided. A logged-in user can search for an available car within a certain range from his current position or from a given address and, in case, he can pick up the car. Finally, he has the possibility to reserve a chosen car (he can also delete the reservation subsequently if the car won't be used: in this way he is free from fees).

Moreover, to promote a more eco-friendly behaviour of the users, the system applies special discounts or extra charges on the cost of the rides according to some different conditions that will be described in the Non Functional Requirements section.

With regards to the company employees, the system will let them use special functions to retrieve specific information about the cars (e.g.: battery level, internal working status) and their positions. In this way, they can move cars from safe areas to charging areas and vice versa if needed.

## 1.2 Scope

The system-to-be will interact with three main world entities: *People*, *Physical platform* and *External software services*. *People* entity includes the clients and all those people who have a relation with the system, such as the employees. For what concerns the clients, the interaction with the system consists in both the functionalities that the system provides and the events that are triggered by the users. Log-in, reservations of cars, rides, notifications, parking are the main shared phenomena between the clients and the system.

Employees collaborate with the system in terms of managing the cars: the search of cars, the notification of their states, the parking are the core shared phenomena between the employees and the system.

*Physical platform* consists of the physical infrastructure of the world with which the system exchanges information. It includes the cars, the *Safe Areas* and *Charging Areas*. Unlocking, locking the cars and routes management are the main phenomena that are shared with the system.

*Safe Areas* and *Charging Areas* come into account when considering the search for the availability of parking lots or of power plugs.

Finally, *External software services* are the software tools that cooperate with the system-to-be. They can be considered external systems that already exist in the environment. The main ones are the payment system and GPS location system. The first one interacts with

*PowerEnJoy* in the reservation and in the payment phases while GPS system helps with all those actions that require information about position (search for a car, notification, parking, reservation).

## 1.3 Definitions, Acronyms and Abbreviations

**PowerEnJoy** is the name of the system that has to be developed.

**System** sometimes called also *system-to-be*, represents the application that will be described and implemented. In particular, its structure and implementation will be explained in the following documents. People that will use the car-sharing service will interact with it, via some interfaces, in order to complete some operations (e.g.: reservation and renting).

**Renting** it is the act of picking-up an available car and of starting to drive.

**Ride** the event of picking-up a car, driving through the city and parking it. Every Ride is associated to a single user and to a single car.

**Reservation** it is the action of booking an available car.

**Car** a car is an electrical vehicle that will be used by a registered user.

**Not Registered User** indicates a person who hasn't registered to the system yet; for this reason he can't access to any of the offered function. The only possible action that he can carry out is the registration to get a personal account.

**Registered User** interacts with the system to use the sharing service. He has an account (which contains personal information, driving license number and payment data) that must be used to access to the application in order to exploit all the functionalities.

**Employee** it's a person who works for the company, whose main task is to plug into the power grid those cars that haven't been plugged in by the users. He is also in charge of taking care of the status of the cars and of moving the vehicles from a safe area to a charging area and vice versa if needed.

**Safe Area** indicates a set of parking lots where the users have to leave the car at the end of the rent; the set of the Safe Areas is pre-defined by the system management. These areas are spread all over the city.

**Plug** defines the electrical component that physically connects the car to the power grid.

**Charging Area** is a special *Safe Area* that also provides a certain number of plugs that connect the cars to the power grid in order to recharge the battery.

**Registration** the procedure that an unregistered user has to perform to become a registered user. At the end, the unregistered user will have an account. To complete this operation three different types of data are required: personal information, driving license number and payment info.

**Search** this functionality lets the registered user search for available cars within a certain range from his/her current position or from a specified address.

**RASD** is the acronym of *Requirements Analysis and Specification Document*

**DD** is the acronym of *Design Document*

## 1.4 Reference Documents

During the writing of this document, the following resources have been taken into account:

- Specification Document:
  - *Assignments+AA+2016-2017.pdf*
  - *RASD.pdf*
- Example document:
  - *Sample Design Deliverable Discussed on Nov. 2.pdf*
- Papers on *Green Move Project*

## 1.5 Document Structure

The first chapter recalls the purpose and the scope of the system that has to be developed and in particular of the current document, which focuses on the architecture of *PowerEnjoy*.

In the second chapter, which is the core of the *Design Document*, the overall architecture will be presented at various levels of abstraction. After that a brief explanation of the architectural styles and patterns used is provided.

The third chapter covers the algorithmic part of the system; some interesting algorithms that will be useful later on in the real implementation are described in pseudocode.

The fourth chapter contains the mockups of both the webapp user interfaces and the ad-hoc touch screen embedded into the cars. Along with them, there is a small description.

The fifth chapter reports the mapping of the requirements, identified in the *RASD*, with respect to the architecture's component.

The sixth and seventh chapters simply refers to the time effort spent in writing the *DD* and the used tools to do it.

## 2 | Architectural Design

### 2.1 Overview

### 2.2 Component View

### 2.3 Deployment View

### 2.4 Runtime View

### 2.5 Component interfaces

### 2.6 Selected Architectural Styles and Patterns

### 2.7 Other Design Decisions

#### 2.7.1 ER Diagram

An important issue in the Design Process is to select the suitable data that the system has to store in the database. This is crucial in the Design Phase because it affects the implementation of all the system functionalities, which rely on these data. The following diagram shows the pieces of information that the database server provides to the application.

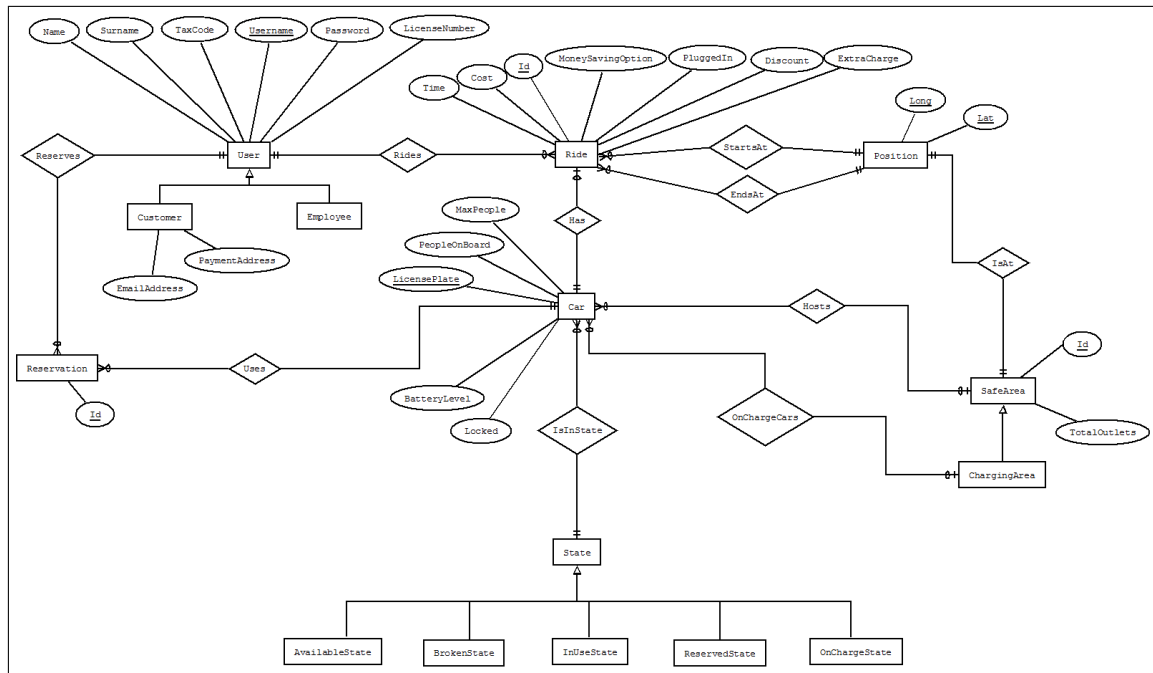


Figure 2.1: ER Diagram

The diagram suggests that the system has two types of *Users*: *Customer* and *Employee*. As already stated in the *RASD* this is done to distinguish the roles and permissions of the two types of actors.

The system has to store information about the *Ride* such as the time (in order to be able to compute the cost), possible discount or extra charge and the initial and final position so to keep track of the associated *Car*.

*Cars* are identified by their unique license plate and the system is always updated on their current state, which is one of the shown above.

The system also records the info of each *Reservation* in terms of the user and the corresponding car. This is clearly not enough to identify in a unique way a reservation, so the attribute *Id* is used as a key field for it.

Finally the system stores the data that are necessary to properly represent the *SafeAreas* and the *ChargingAreas*: their position, their id, information about the number of total outlets and the cars that are parked. This piece of information is crucial for some key functionalities of the system, such as the computation of the uniform distribution of cars in the city.



## 3 | Algorithm Design

## 4 | User Interface Design

## 5 | Requirements Traceability

## 6 | Effort Spent

## 7 | References

### 7.1 Tools

During the writing of this document, the following application tools have been used:

- Star UML, for creating all types of UML models
- TeXstudio, for writing the document in  $\text{\LaTeX}$
- Balsamiq, for creating the mockups of the user interface
- draw.io, for drawing the diagrams of the architecture