Requirement Analysis and Specification Document for PowerEnJoy

Enrico Migliorini, Alessandro Paglialonga, Simone Perriello

November 6, 2016

Contents

1 Introduction

1.1 Purpose

This document presents the requirements of the *PowerEnJoy* system, aimed at powering a car-sharing service. Said requirements will be presented using both natural language and diagrams.

1.2 Intended Audience

This document is addressed to all the stakeholders in the *PowerEnJoy* project. This includes, but is not limited to, the CEO, the end users, the development committee, product designers and engineers, quality assurance and marketing.

1.3 Product Scope

The *PowerEnJoy* is a partially automated electric car-sharing service. The system keeps track of users and Cars, addresses users to available cars, locks cars when not in use, and charges the users for use and abuse of the Cars. The system also needs to keep track of the battery level of the Cars and dispatch personnel to connect low-on-battery Cars to the power grid.

It is important to notice that this document only describes the requirements for the software dealing with the cars, not the hardware on which the software will run or the management system.

1.4 Definitions, Acronyms and Abbreviations

1.4.1 Business terms glossary

[OBSOLETE: REFER TO GLOSSARY.TXT, THIS WILL BE UPDATED LATER]

1.4.1.1 Car-sharing A *Car-sharing* service allows *Users* to rent *Cars* for a limited amount of time, being charged a *Fee* according to time and possibly applying a *Discount* or an *Increase*.

1.4.1.2 Database A structure that holds informations linked logically according to relationships. For instance, a *Database* could hold records of every registered *User*, every available *Car* and every time a *User* rented a *Car*.

1.4.1.3 Discount A reduction in the *Fee* to be paid because of good behaviour on the part of the *User*, e.g. leaving the *Car* plugged or bringing it back with a mostly-full battery. The actions that constitute good behaviour are determined ad detailed further in the document. [ADD A REFERENCE WHEN YOU MAKE THE SECTION]

1.4.1.4 Fee The amount of money that the *User* will be charged for his usage of the *Car-sharing* service.

1.4.1.5 Increase An increase in the *Fee* to be paid because of improper behavious on the part of the *User*, e.g. bringing the *Car* back with a mostly-empty battery.

1.4.1.6 Management System An external system that allows administrative access to the internal *Database*.

1.4.1.7 System The automated software structure this document is about. It tracks *Users* and *Cars* and deals with all the details needed for *Car-sharing*, from GPS mapping to charging *Users* with *Fees*.

1.4.1.8 User A person registered on the *System*, who will use the *Cars* for a *Fee*.

1.4.1.9 Car An electric car owned by the *Car-sharing* service, rented to the *User* and tracked by the *System*.

1.4.2 Document specific terms

1.4.2.1 Alloy A descriptive language that allows to describe a set of structures through constraints.

1.4.2.2 DBMS Data Base Management System. A software interface allowing to interact with the *Database*.

1.4.2.3 RASD Requirements Analysis and Specification Document. This document, describing the *System* to be developed.

1.4.2.4 UC Use Case. A description of interaction between *Users* and *System*.

1.4.2.5 UML Unified Modeling Language. A language for modeling Object-Oriented software systems.

2 General Description

2.1 General Description

The PowerEnjoy system is created in order to provide a unified platform to be used for reserving and renting electrical Cars.

Access to the full System is reserved to the registered Users, although it is also possible for a Visitor to register themselves, becoming a User.

Once registered, a User will have easy access to the functionalities of PowerEnJoy, allowing them to see the locations of nearby Cars, reserve and rent them and enable money-saving options to become eligible for Discounts.

2.2 Product Perspective

The main System will be housed on a central elaborator, called server. This server will be responsible for all necessary operations, including communicating with the internal Database, keeping track of all the Cars and interacting with the Banking System in order to charge Fees to the Users. The user will access the System through a mobile app (client), connected to the main System via Internet, and interacting with the Smartphone's GPS System. Information exchange between the client and the server will be based on the HTTP protocol.

2.3 Product Functions

Here are explained the main needs of the PowerEnJow system.

Need	**Priority**	**Function**
Registration	?	Allows a Visitor to register their credentials, becoming a User
Login	?	Lets a User access the system
Find Nearby Cars	?	Shows Available Cars in a location
Reservation	?	Allows a user to set a nearby Car as Reserved
Proximity Unlock	?	Unlocks the car when the User who Reserved it is close

Need	Priority	Function
Computation	?	Computes the Fee to be paid, including Discounts and Increases
Payment	?	Charges the user for the computed Fee
Detect Parking	?	Detects whether a Car has been left in a Parking Area
Detect Charging	?	Detects whether a Car is Plugged to a Charging Area
Administrative Access	?	Allows an administrator to modify the information in the Database or parametres of the System

2.4 User Classes and Characteristics

Name	Description	Responsibility
Visitor	A person who needs to undergo Registration to become a User	Performs a Registration
User	Someone who is registered in the system and can access its functionalities	Can find, reserve and rent Cars
Administrator	A specialized worker for the Company	Can modify the Database and the application's various parameters

2.5 Operating Environment

Users and Visitors will access the System through a smartphone application. The smartphones used to access the application will need to have a sufficiently advanced hardware in order to run it, and the application will require both an Internet connection and a GPS signal. The mobile application will, however, only be required in order to run the Find Nearby Cars, Reservation and Proximity Unlock functions.

The Cars will also need to be modified by adding a device capable of getting GPS signal and remotely communicate with the central System.

The central System will need a suitably powerful elaborator to run on.

2.6 Design and Implementation Constraints

The system will employ the HTTP protocol in order to have the client software, both the software application and the one installed on the Cars, communicate with the central System. As mentioned above, a network connection will be necessary. In order to maximize availability, it is advised to use the best available hardware.

User ID and password will be required in order to access the System. In order to register as a User, a Visitor needs to undergo Registration.

2.7 Assumptions and Dependencies

[INTEGRATE FROM THE LIST]

3 External Interface Requirements

The interfaces the system should interact with.

3.1 User Interface

The User Interface is provided by the client smartphone application, and allows the user to perform all of their actions.

3.2 Hardware Interface

The client application must be developed so as to have access to both the phone's network connection and its GPS locator.

The central server must be provided with one or more sufficiently advanced computers that may run the server-side application, and allow it access to a high-speed network connection.

The Cars will be fitted with the device mentioned in 2.5, allowing access to various informations, such as battery status, location, and eventual presence of Passengers.

3.3 Software Interface

The System does not need to interface with external software. Database management can be performed via the Administrative Access function.

3.4 Communication Interface

As mentioned above, the System heavily uses Internet communications protocols, mainly the HTTP protocol. HTTP requests to and from the server will be mostly carried by mobile network connections.

4 Functional Requirements

The functionality for the various users.

4.1 Use cases specification

4.2 Use Case Diagram

4.3 Alloy representation of requirements

5 Non-functional Requirements

Additional requirements that may be added to improve on the program.

5.1 Performance Requirements

5.2 Safety and Security Requirements

5.3 Software Quality Attributes

5.3.1 Availability

5.3.2 Reliability

5.4 Business Rules