Content

1. Introduction
    1. Purpose
    2. Scope
    3. Definition, acronyms, abbreviations
    4. Document structure
2. Architecture design
    1. Overview
    2. High level components and their interaction
    3. Component view
    4. Deploying view
    5. Run-time view
    6. Component interfaces
    7. Selected architectural styles and pattern
    8. Other design decisions
3. Algorithm design
4. User interface design
5. Requirements traceability
6. Reference
7. Hours of working

1. Introduction

1. Purpose

In this document, more technical details will be presented than the RASD about the PowerEnjoy system.

As we compeleted before in the RSAD, we have shown a general system what it looks like and how it works. This document aims to present how we implement the system specifically includes compnent view, Run-time view, deploying view, algorithm design,etc.

2. Scope

The project PowerEnjoy, which is a service based on mobile application(based on Android) and web application. The system allows user to reservate a electric car via mobile app and web app.

3. Definition, acronyms, abbreviations

- **RASD**: requirements analysis and specifications document.
- **DD**: design document.
- **API**: application programming interface; it is a common way to communicate with another system or service.
- **GUI**: **G**raphical **U**ser **I**nterface.

4. Document structure

2. Architecture design

1. Overview

2. High level components and their interaction

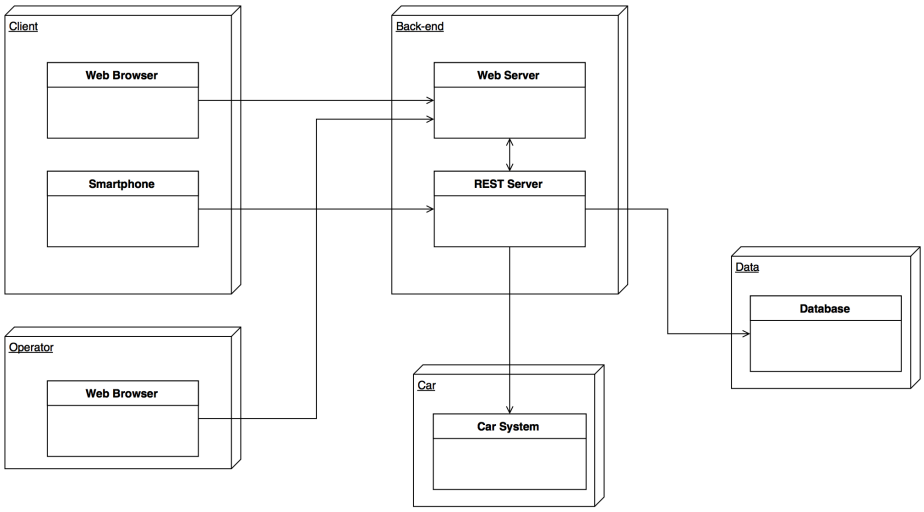

Figure 1: High Level Components

The figure above describes the high level components and their interaction. It is based on the application architecture presented in the RASD Document // TODO Add section //. The REST Server is the main component in our application as it is the central point between the GUI (Web and mobile), the database and the cars. It holds the business logic of our system.

Web browser

Any common web browser can be used by the client or the operator to access the application. The user and the operator have different access levels. The user can register, manage his account and reserve a car through the website. The operator, on the other hand, have more rights in the website. He can check the localization of all the cars despite their status. He can also do all CRUD operations on the cars and users. The operator can as well validate the users when he checks their driving license.

Smartphone

The mobile application is designed uniquely for the user. As for the website, it can be used to register, manage account and reserve a car. The mobile

application is also used to unlock the car a user reserved using the position of the device. At the end of a ride, the user can check the bill on the mobile application.

Web Server

The web server is the component that takes part of formatting and serving pages to the client/operator. No business logic is held in this component.

Application Server

This server holds all the application's business logic. It is the core of the application. It interfaces with other components through a REST API. The mobile application communicates directly with this server to operate. The web browser in the other hand do not have a direct access to this server. It is the duty of the web server to communicate get date from the REST Server, format it and the follow it to the web browser.

Database

The database is used to persist data generated and used by the application. For security and integrity reasons, the database is only connected the REST Server. Other components are indirectly connected through the latter.

Car Component

It is the abstraction of the on-board computer. It take care of gathering all car variables (battery levels, location, passenger, locking system…) and sending them to the application server that uses them for operations.

3.Component view

- Notification Helper : Manage notifications, noticing the user that they are already close to the car.
- Ride Controller : manage rides,
- Reservation Controller : manage reservation,
- Bill Controller : manage payment method and bills,
- Economic Controller : manage money saving request,
- Car Controller : manage the status and availability of cars,
- Router : route the request to related controller,
- Clients : mobile application based on Android and web application (browser),
- User Controller : manage user, access log in or sign in request.

4. Deploying view

5. Run-time view

Sequence diagrams for - Login process W

In this sequence diagram it can be shown that users have to input their login information to the App when they want to use the system. The login request is sent with these information to the system as parameter. First these information will be sent to the UserController which will check these in the database. If users' information(username) is found in the database and the password matches the username then the UserController returns login_success message to the Mobile application so that user can login into the system. Otherwise, the system shows error messages.

- Reservation process RA

The user starts the process, in the mobile application, by searching for cars close to him or around an address. The mobile application sends a request to the Car Controller (through the router). The location is passed as a parameter in the call. The location comes from the GPS module of the smartphone or is manually entered by the user. The Car Controller calls the Localization Controller to get all the cars close to the location. Upon return, the list is filtered and then displayed to the user, if it is not empty. The user can choose a car to reserve, it is then added as a reservation and marked as unavailable. To conclude the operation, a success message is displayed to the user.

- Billing process RA

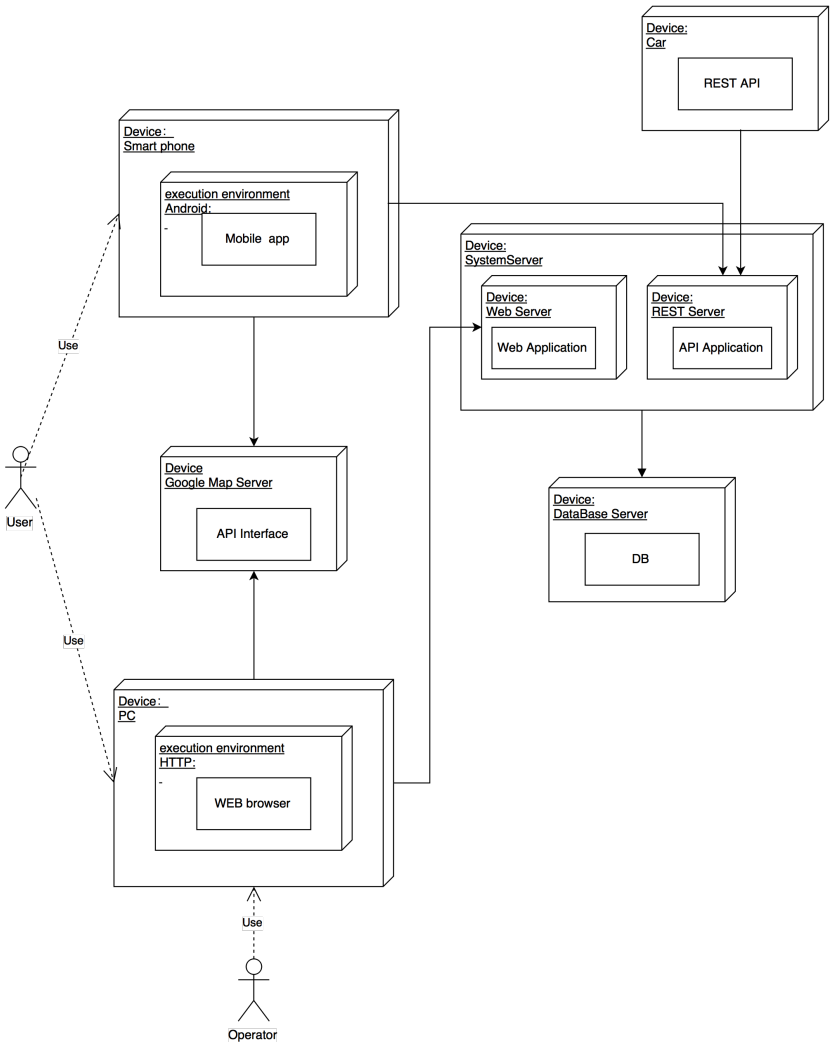

Figure 2: Deploying view

The billing process is started at the end of a ride. The ride controller signals the bill controller that a ride has ended. The reservation relative to that ride is passed as a parameter as it contains many variables that are used in calculating the bill.

- Check-in car process J
- Check-out car process J
- Money saving process W

In this sequence diagram it can be seen that if users ask for MoneySaving option they will be asked to input a destination by the system. The request is then sent with the filled information as parameter to the system.

7. Selected architectural styles and pattern

Application architecture

As stated in the RASD Document, we will be using the 3-tier client-server architecture. The presentation tier is composed of the mobile application and the website. The application layer is composed of two parts. The REST Server that exposes the REST API and holds the business logic. It can be consumed by the web server or the mobile application. In addition, it is a security barrier between the client and the database as it prevents direct accesses to the database by the user. The other component of the application layer is the web server. The web server takes care of formatting the data in webpages and communicating with the web browser. The last tier is the data tier which is, in this case, composed of only one database that takes care of persisting the data of the whole application. The fact that the business logic is held at the level of our servers, the client-side of the application is kept as light as possible. Therefore, users can quickly access the application by installing it on their device or browsing the website. It also prevents the direct access to the database from the GUI which increases security.

Servers application

Model-View-Controller patter is used in both the web server and the application server.

3. Algorithm Design

1. Billing process

After the user finishes a ride in a PowerEnjoy car, the application has to calculate the amount that should be charged to the user. The amount is calculated is the multiplication of the time (in minutes) spent in the car and the price per minute. After calculating this amount, discounts/penalties may be applied:

- *10% discount* if the driver had other passengers with him.
- *20% discount* if he left the car with at least 50% battery.
- *30% discount* if he parked the car in a charging station and plugged it.
- *30% penalty* if he left the car 3km away from charging stations or with less than 20% battery.

If many discounts or penalties should be applied, they cumulate and are applied on the total ride amount. For example, a client had more than two passengers with him and left the car charging in a station he will benefit of 40% discount (10% + 30%).

```
int PRICE_PER_MIN = config.getPricePerMin();

public Bill caculate_Bill(Reservation r)
{
  Ride ride = r.ride;
  Car  car  = r.car;

  float BaseFee = ride.duration * PRICE_PER_MIN;
  float discount = 0;
  // More than 2 passengers discount
  if (car.passengers >= 2) discount += 0.1 ; // 10% Discount
  // More than  50% battery left
  if (car.battery >= 0.5 ) discount += 0.2; // 20% Discount
  // User recharged car
  if (car.isCharging()) discount += 0.3; // 30% Discount
  // If car is left 3KM away from charging OR 20% battery
  boolean isFar = LocalisationController.isCarFar();

  if (LocalisationController.isCarFar(car) || car.battery <= 0.2)
              discount -= 0.3;

   float ChargedAmount = BaseFee * (1 - discount );

   return new Bill(new Date (), ChargedAmount);
}
```

Hours Worked

Reda Aissaoui

- 23/11/2016 4h
- 24/11/2016 4h
- 01/12/2016 4h
- 05/12/2016 1h