"resources/images/""logo_polimi".png

SafeStreets

Marco Premi (941388)

marco.premi@mail.polimi.it

Fabrizio Siciliano (939895)

fabrizio.siciliano@mail.polimi.it

Giuseppe Taddeo (928360)

giuseppe.taddeo@mail.polimi.it

Computer Science and Engineering

2019/2020

Software engineering 2

DD

Design Document

Version 1.0 - [Data da inserire]

Reference professor:

Matteo Giovanni Rossi
`matteo.rossi@polimi.it`

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to give more technical details than the RASD about SafeStreets system. The RASD presented a more abstract and general view of the system and of the functions is supposed to execute. Indeed, this document presents more details about the design, run-time processes, deployment and algorithm. It also provideds more information about implementation, integration and testing with a testing plan.
In particular, the document presents the following topics:

- Overview of the high-level architecture
- The main components, their interfaces and deployment
- The run-time behavior
- The desing patterns
- Requirements on architecture components
- Implementation plan
- Integration plan
- Testing plan

1.2 Scope

Here it's presented a review of the application scope, made referring to what has been stated in RASD document.
With SafeStreets users can notify the authorities when traffic violations occur, and in particular parking violations. Both user and authorities must register to the application and agree that SafeStreets stores the information provided, completing it with suitable meta-data. The whole system, because it tracks users information, must respect the standards defined for processing of sensitive information such as GDPR if it is used in Europe. The user sends the type

of the violation to the municipality and direct proofs of it (like a photograph). The system runs an algorithm to read the license plate and also asks the user to directly insert the license for a better recognition. Obviously, other information are required, like the name of the street when the violation has occurred, which can be retrieved from user's direct input or from the geographical position of the violation (using Google Maps API). Furthermore, the system, by cross referencing data from third party services, automatically can highlight the streets with the highest frequency of violations or the vehicles that commit the most violations. SafeStreets crosses information about the accidents that occur on the territory of the municipality with his own data to identify potentially unsafe areas and suggest possible interventions. Because municipality could generates traffic tickets from the information about violations SafeStreets should guarantee that information is never altered (if a manipulations occurs, the application should discard the information). Such features are made possible trough the use of one mobile application with two different UIs which are determined by the kind of customer that logs in (user or PO). The collected information are sent to a back-end and they all of those can be accessed by municipality employees in order to execute different actions (emit ticket, analyze unsafe areas, etc...).

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

User: it is identified as a civilian customer of the product. It will be the main source for the SafeStreets initiative to obtain information about traffic violations and therefore to be successful;

1.3.1.0.1 Third parties: those kind of organization/company that could provide services useful to SafeStreets;

1.3.1.0.2 Customer: it defines both authorithy users (police officers or municipality employees) and civilians;

1.3.1.0.3 Authority user: all of those customers who have a responsibility role in regard of the streets' safety and the SafeStreets initiative. Example of these category are: police officers, municipal employees, director and basically anyone in charge and able to issue fines and deal with road violations;

1.3.1.0.4 Ghiro: image manipulation detection software, used by authorithy users in order to detect any image manipulation and assess the veracity of the hard evidence connected to the traffic ticket

1.3.2 Acronyms

1.3.2.0.1 UI: User Interface

1.3.2.0.2 GDPR: General Data Protection Regulation

1.3.2.0.3 API: Application Programming Interface

1.3.2.0.4 GPS: Global Positioning System

1.3.2.0.5 PO: Police Officer

1.3.2.0.6 ME: Municipality Employee

1.3.3 Abbreviations

1.3.3.0.1 Gn: nth goal;

1.3.3.0.2 Dn: nth assumption;

1.3.3.0.3 Rn: nth requirement;

1.3.3.0.4 ID: identifier (Fiscal Code for Users, a municipality identifier for Authorithy Users)

1.4 Revision history

1.5 Reference Documents

1.6 Document Structure

Chapter 1 - Introduction

Chapter 1 is an introduction of the design document. It describes the purpose and the scope of the document and it highlights the differences with the RASD. It also shows some abbreviations, definitions and acronyms in order to provide a better understanding of the document to the reader.

Chapter 2 - Architectural design

Chapter 2 deals with the architectural design of the system and it's the core section of the document.
It provides an overview of the architecture and it contains the most relevant architecture views:

- Component view
- Deployment view
- Run-time view

It also shows the interaction of the interfaces and the selected architectural styles and patterns, with an explanations of each one of them.

Chapter 3 - User interface design

Chapter 3 specifies the user interface design and refers to the mock-ups already presented in the RASD.

Chapter 4 - Requirements traceability

Chapter 4 explains how the requirements defined in the RASD map to the design elements defined in this document.

Chapter 5 - Implementation, integration and test plan

Chapter 5 specifies the description and the order of implementation, integration and testing plan of the sub-components of the system.

Chapter 6 - Effort spent

Chapter 6 shows the effort spent by each member of the group working on this project.

Chapter 7 - References

Chapter 7 includes the reference documcuments.

2 | Architectural design

2.1 Overview

2.2 High-level architecture

2.3 Component view

2.4 Deployment view

2.5 Run-time view

2.5.1 Synchronization

2.5.2 Request data regarding a group of people

2.5.3 Request data regarding a particular user by providing his/her UUID

2.6 Component interface

2.7 Selected architectural styles and patterns

2.7.1 Design Patterns

Sensor Aggregation

Model View Controller (MVC)

Observer

2.8 Other design decisions

3 | User interface design

3.1 Interface mockups

3.2 UX Diagrams

3.2.1 Mobile application

3.2.2 Web application

4 Requirements traceability

5 | Implementation, integration and test plan

5.1 Implementation plan

5.2 Integration and testing

5.2.1 Entry criteria

5.2.2 Elements to be integrated

5.2.3 Integration testing strategy

5.2.4 Sequence of component/function integration

6 Effort spent

Marco Speziali

Chapter	Effort (hours)
Chapter 1 - Introduction	1
Chapter 2 - Architectural design	15
Chapter 3 - User interface design	3
Chapter 4 - Requirements traceability	3
Chapter 5 - Implementation, integration and test plan	3
Total (hours)	22

Qi Zhou

Chapter	Effort (hours)
Chapter 1 - Introduction	1
Chapter 2 - Architectural design	15
Chapter 3 - User interface design	3
Chapter 4 - Requirements traceability	3
Chapter 5 - Implementation, integration and test plan	3
Total (hours)	22