Politecnico di Milano

SAFESTREETS - DD

POLITECNICO
MILANO 1863

Samuele Meta - Stiven Metaj

Supervisor: Matteo Rossi

Department of Computer Science and Engineering

December 4, 2019

Contents

1 Introduction

1.1 Purpose

The following Design Document (DD) is aimed to provide an overview of the *SafeStreets* application, explaining how to satisfy the project requirements declared in the RASD and stating the successive refinements made together with the Stakeholders according to their needs. The document is mainly intended to be used by developers teams as a guidance in the development process, by testing teams to write automated testing and to avoid structural degradation of the system in case of maintenance or future extension. Indeed, its purpose is to provide a functional description of the main architectural components, their interfaces and their interactions, along with the design patterns and algorithms to be implemented.

1.2 Scope

As explained in the RASD document, *SafeStreets* is a crowd-sourced mobile application that allows Citizens to notify Authorities about traffic violations, with particular emphasis on parking contraventions. Citizens are able to manage their reports, visualizing the whole history and modifying or removing them. On the other hand, Authorities can access a complete overview of the incoming reports and choose if to accept or reject them.

1.3 Definitions, Acronyms, Abbreviations

In this section follow definitions, acronyms (including their meaning) and abbreviations used in this document.

1.3.1 Definitions

- *Client*: a desktop computer or workstation that is capable of obtaining information and applications from a Server.
- *Server*: a computer or computer program which manages access to a centralized resource or service in a network.
- *Firewall*: a part of a computer system or network which is designed to block unauthorized access while permitting outward communication.
- *Port*: an endpoint of communication in an operating system.
- *Design Pattern*: reusable software solution to a commonly occurring problem within a given context of software design.

1.3.2 Acronyms

DBMS	DataBase Management System
HTTPS	Hyper Text Transfer Protocol over SSL
API	Application Programming Interface
REST	REpresentational State Transfer
MVC	Model View Controller
OS	Operative System
UI	User Interface
UX	User Experience
URL	Uniform Resource Locator
RASD	Requirements Analysis and Specification Document
SPA	Single Page Application
ERD	Entity-Relationship Diagram
SSL	Secure Sockets Layer
DDoS	Distributed Denial of Service

Table 1.1: *Acronyms*

1.3.3 Abbreviations

- [**R.n**]: n-th Requirement in the RASD document

1.4 Revision History

[TBD]

1.5 Reference Documents

- Specifications document "SafeStreets. Mandatory project assignment"
- SafeStreets RASD Document
- IEEE Standard 1016-2009: IEEE Standard on Software Design Descriptions
- UML documentation: `https://www.uml-diagrams.org`

1.6 Document Structure

The rest of the document is organized as follows:

- **Architectural Design**: details the System's architecture by defining the main components and the relationships between them as well as specifying the hardware needed for the System deployment. It will also be focused on design choices and architectural styles, patterns and paradigms.
- **User Interface Design**: provides further details on the UI defined in the RASD document through the use of UX modeling.
- **Requirements Traceability**: shows the relations between the requirements from the RASD and the design choices of the DD and how they are satisfied by the latter.
- **Implementation, Integration and Test Plan**: provides a roadmapping of the implementation and integration process of all components and explains how the integration will be tested.
- **Effort Spent**: describes how the work has been split between the members of the team and how long did the DD take to be completed.

2 Architectural Design

2.1 Overview

2.2 Component View

2.3 Deployment View

2.4 Runtime View

2.5 Component Interfaces

2.5.1 REST API

2.6 Selected Architectural Styles and Patterns

2.6.1 Multi-tier Architecture

2.6.2 Use of RESTful guidelines

2.7 Other Design Decisions

2.7.1 Thin Client

2.7.2 MVC

2.7.3 Firewalls

3 User Interface Design

3.1 User eXperience Diagrams

4 Requirements Traceability

5 Implementation, Integration and Test Plan

5.1 Overview

5.2 Component Integration

5.3 Something on Testing?

6 Effort Spent

The effort spent from each member of the team to build the DD can be summarized with the following tables: