

AA 2019/2020

Computer Science and Engineering

Software Engineering 2 Project

SafeStreets

**RASD**

Requirement Analysis and Specification Document

version 1.0 --- 04/11/2019

|  |  |
| --- | --- |
| Authors: | Professor: |
| Aida Gasanova | Elisabetta Di Nitto |
| Alexandre Batistella Bellas |  |
| Ekaterina Efremova |  |

**Table of Contents**

[1 Introduction 3](bookmark://_Toc23811417#_Toc23811417)

[1.1 Purpose 3](bookmark://_Toc23811418#_Toc23811418)

[1.1.1 General purpose 3](bookmark://_Toc23811419#_Toc23811419)

[1.1.2 Goals 3](bookmark://_Toc23811420#_Toc23811420)

[1.2 Scope 4](bookmark://_Toc23811421#_Toc23811421)

[1.3 Definitions, acronyms and abbreviations 4](bookmark://_Toc23811422#_Toc23811422)

[1.3.1 Definitions 4](bookmark://_Toc23811423#_Toc23811423)

[1.3.2 Acronyms 4](bookmark://_Toc23811424#_Toc23811424)

[1.3.3 Abbreviations 4](bookmark://_Toc23811425#_Toc23811425)

[1.4 Revision history 4](bookmark://_Toc23811426#_Toc23811426)

[1.5 Reference documents 4](bookmark://_Toc23811427#_Toc23811427)

[1.6 Document structure 5](bookmark://_Toc23811428#_Toc23811428)

[2 Overall Description 6](bookmark://_Toc23811429#_Toc23811429)

[2.1 Product perspective 6](bookmark://_Toc23811430#_Toc23811430)

[2.2 Product functions 6](bookmark://_Toc23811431#_Toc23811431)

[2.3 User characteristics 6](bookmark://_Toc23811432#_Toc23811432)

[2.4 Assumptions, dependencies and constraints 6](bookmark://_Toc23811433#_Toc23811433)

[3 Specific Requirements 7](bookmark://_Toc23811434#_Toc23811434)

[3.1 External interface requirements 7](bookmark://_Toc23811435#_Toc23811435)

[3.1.1 User interfaces 7](bookmark://_Toc23811436#_Toc23811436)

[3.1.2 Hardware interfaces 7](bookmark://_Toc23811437#_Toc23811437)

[3.1.3 Software interfaces 7](bookmark://_Toc23811438#_Toc23811438)

[3.1.4 Communication interfaces 7](bookmark://_Toc23811439#_Toc23811439)

[3.2 Functional requirements 7](bookmark://_Toc23811440#_Toc23811440)

[3.3 Performance requirements 7](bookmark://_Toc23811441#_Toc23811441)

[3.4 Design constraints 7](bookmark://_Toc23811442#_Toc23811442)

[3.4.1 Standards compliance 7](bookmark://_Toc23811443#_Toc23811443)

[3.4.2 Hardware limitations 7](bookmark://_Toc23811444#_Toc23811444)

[3.4.3 Any other constraint 7](bookmark://_Toc23811445#_Toc23811445)

[3.5 Software system attributes 7](bookmark://_Toc23811446#_Toc23811446)

[3.5.1 Reliability 7](bookmark://_Toc23811447#_Toc23811447)

[3.5.2 Availability 7](bookmark://_Toc23811448#_Toc23811448)

[3.5.3 Security 7](bookmark://_Toc23811449#_Toc23811449)

[3.5.4 Maintainability 7](bookmark://_Toc23811450#_Toc23811450)

[3.5.5 Portability 7](bookmark://_Toc23811451#_Toc23811451)

[3.6 Other requirements 7](bookmark://_Toc23811452#_Toc23811452)

[4 Formal analysis with Alloy modeling 8](bookmark://_Toc23811453#_Toc23811453)

[5 Appendices 9](bookmark://_Toc23811454#_Toc23811454)

[5.1 Used tools 9](bookmark://_Toc23811455#_Toc23811455)

[5.2 Hours of effort spent 9](bookmark://_Toc23811456#_Toc23811456)

1. Introduction
   1. Purpose
      1. General purpose

This document represents the Requirement Analysis and Specification Document (RASD). The main purpose of this document is to fully describe the software product in order to help developers model it.

This document describes SafeStreets application, which can help the prevention of traffic violations, and in particular parking violations, by sending the information obtained by ordinary pedestrians who are users of this application to the authorities. Both sides, users and authorities, can use this data for useful purposes, for example, see areas that have a high frequency of violations, or even the vehicles that commit the most violations. Besides this, the application can cross its own data with external data from municipally (if available) to identify potentially unsafe areas, and suggest possible interventions. Lastly, the application can provide information to a municipally system that emits traffic tickets to people that committed violations, since the information came from the application is with guaranteed integrity.

* + 1. Goals
  + G1: The system must allow users to report traffic violations, and in particular parking violations;
  + G2: The system must identify the license plate automatically from the picture sent by a user, storing the information, beside all provided by the user, in a local database;
  + G3: The application must allow users and authorities to mine the information stored in the system;
  + G4: The application must cross its information with the information about the accidents that occur on the territory of the city;
  + G5: The system must suggest possible interventions to potentially unsafe areas;
  + G6: The system must offer to users and authorities truthful information about the registered violations;
  + G7: The application must build statistics from the issued tickets;

The goals G4 and G5 can only be considered if the municipality offers a service that allows information retrieval about the accidents that occur on the territory of the municipality.

* 1. Scope

The SafeStreets service is a crowd-sourced application offered to common users and authorities that want to follow the violations occurred on the municipality territory.



Figure 1 – SafeStreets as a sharing information system

* 1. Definitions, acronyms and abbreviations
     1. Definitions
* **User:** the “normal” customer of the application that send the information about the violations to authorities or extract the information that have been received (to use it for useful purposes);
  + **Authorities:** the customer of the application that receive the information about violations that have been received from “normal” customers;
  + **Customer:** general SafeStreets customer;
  + **Municipality:** a town or district that has local government;
  + **Violation:** general traffic violation, and in particular parking violation;
    1. Acronyms
* **API:** Application Programming Interface
* **GPS:** Global Positioning System
* **UI:** User Interface
* **AI:** Artificial Intelligence
* **RASD:** Requirement Analysis and Specification Document
  + 1. Abbreviations
* **Gn:** n-th goal
* **Dn:** n-th domain assumption
* **Rn:** n-th functional requirement
  1. Revision history
  2. Reference documents
  + Specification document: “Mandatory Project Assignment AY 2018-2019”
  + IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications
  + UML diagrams: https://www.uml-diagrams.org/
  + Alloy doc: <http://alloy.lcs.mit.edu/alloy/documentation/quickguide/seq.html>
  1. Document structure

The RASD document is composed by five chapters, as outlined below.

**Chapter 1** is an introduction: it describes the purpose of the system informally and also by making use of the list of goals which the application has to reach. Moreover, it defines the scope, where the aim of the project is defined in greater detail and the application domain and the most important shared phenomena are shown.

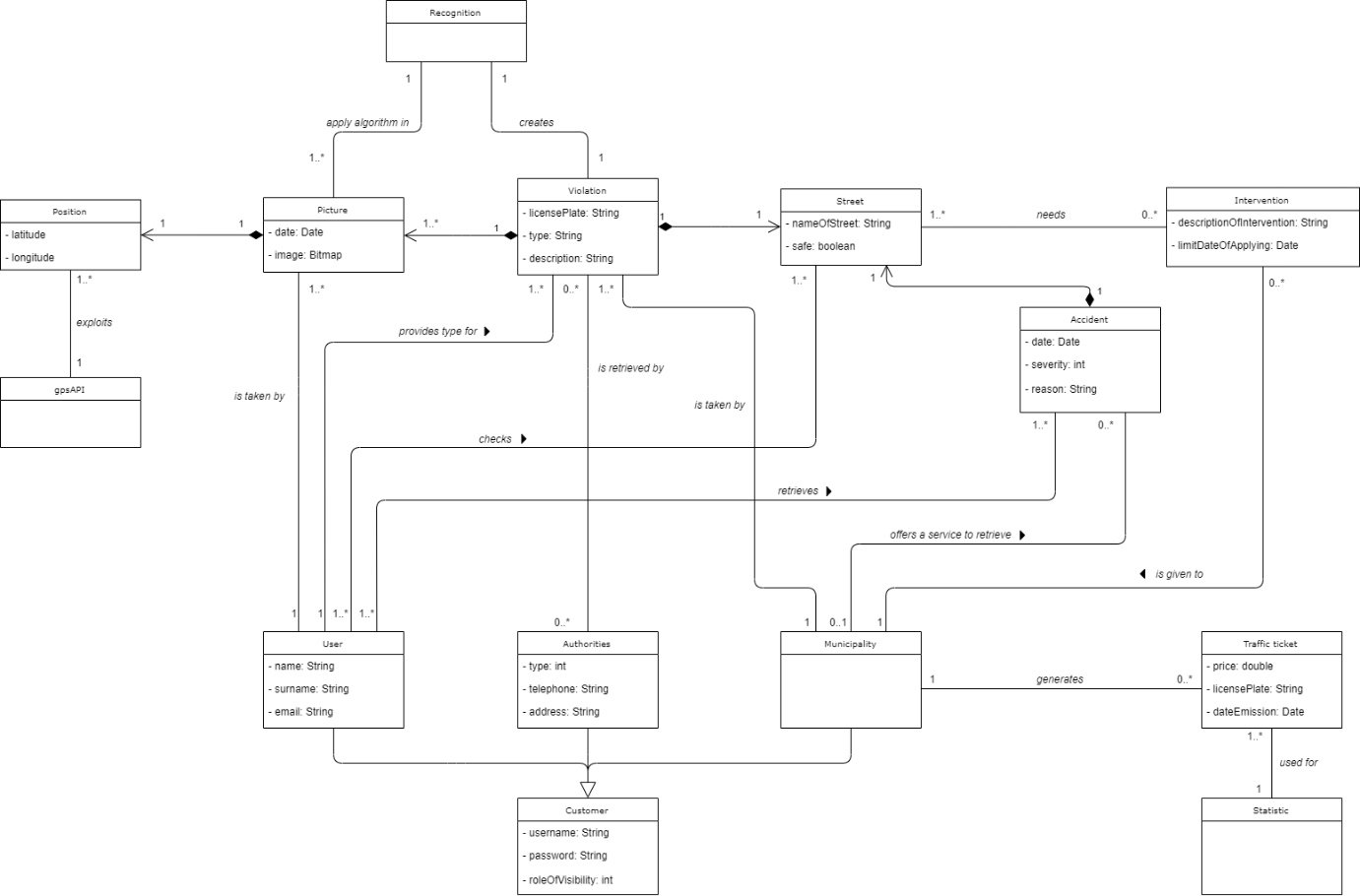
**Chapter 2** offers an overall description of the project. Here the actors involved in the application’s usage lifecycle are identified and the boundaries of the project are defined, listing all the necessary assumptions. Furthermore, a class diagram is provided, aid to better understanding the general structure of the project, with all the related entities. Then some state diagrams are listed to make 10 the evolution of the crucial objects clear. Finally, the functions offered by the system are here more clearly specified, with respect to the previously listed goals.

**Chapter 3** represents the body of the document. It contains the interface requirements, which are: user interfaces, hardware interfaces and software interfaces. It then lists some scenarios to show how the system acts in real world situations, followed by the description of the functional requirements, using use cases and sequence diagrams. All the requirements necessary in order to reach the goals are given, linked with the related domain assumptions. Lastly, the non-functional requirements are defined through performance requirements, design constraints and software system attributes.

**Chapter 4** contains the Alloy model of some critical aspects with all the related comments and documentation in order to show how the project has been modeled and represented through the language.

**Chapter 5** contains the list of the tools used and shows the effort which each member of the group spent working on the project.

1. Overall Description
   1. Product perspective



**Figure 1.** *Class diagram of the software project*

* 1. Product functions
  2. User characteristics
  3. Assumptions, dependencies and constraints

1. Specific Requirements
   1. External interface requirements
      1. User interfaces
      2. Hardware interfaces
      3. Software interfaces
      4. Communication interfaces
   2. Functional requirements
   3. Performance requirements
   4. Design constraints
      1. Standards compliance
      2. Hardware limitations
      3. Any other constraint
   5. Software system attributes
      1. Reliability
      2. Availability
      3. Security
      4. Maintainability
      5. Portability
   6. Other requirements
2. Formal analysis with Alloy modeling
3. Appendices
   1. Used tools

The tools used for the development of this document were those ones listed below.

* Microsoft Office Word 2016
* GitHub
* Alloy Analyser 4.2
  1. Hours of effort spent

The hours spent by the group are listed below, differentiating for each participant.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Hours spent** | | |
| **Aida Gasanova** | **Alexandre**  **Batistella Bellas** | **Ekaterine Efremova** |
| Introduction | 1.5 | 1.5 | 0 |
| Product perspective |  |  |  |
| Product functions |  |  |  |
| Domain assumptions |  |  |  |
| External interface requirements |  |  |  |
| Functional requirements |  |  |  |
| Non-functional requirements |  |  |  |
| Formal analysis using Alloy |  |  |  |