

**POLITECNICO**  
**MILANO 1863**

A.A. 2019/2020  
Computer Science and Engineering  
Software Engineering 2

# ***SafeStreets***

RASD  
Requirement Analisys and Specification  
Document

Bonatti Andrea      Buttironi Monica  
Caruso Marco Giuseppe

10/11/2019

Version: 1.0

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Purpose . . . . .	3
1.1.1	General purpose . . . . .	3
1.1.2	Document purpose . . . . .	4
1.2	Scope . . . . .	4
1.3	Definitions, Acronyms, Abbreviations . . . . .	5
1.3.1	Definitions . . . . .	5
1.3.2	Acronyms . . . . .	5
1.3.3	Abbreviations . . . . .	6
1.4	Revision history . . . . .	6
1.5	Reference documents . . . . .	6
1.6	Document structure . . . . .	6
<b>2</b>	<b>Overall description</b>	<b>8</b>
2.1	Product perspective . . . . .	8
2.1.1	Class diagram . . . . .	8
2.1.2	State diagrams . . . . .	9
2.1.3	World and Phenomena . . . . .	11
2.2	Product functions . . . . .	12
2.2.1	Report . . . . .	12
2.2.2	Improvement . . . . .	12
2.2.3	Statistics . . . . .	12
2.3	User characteristics . . . . .	12
2.4	Assumptions, dependencies and constraints . . . . .	13
<b>3</b>	<b>Specific requirements</b>	<b>14</b>
3.1	External interface requirements . . . . .	14
3.1.1	User interfaces . . . . .	14
3.1.2	Hardware interfaces . . . . .	24
3.1.3	Software interfaces . . . . .	24
3.1.4	Communication interfaces . . . . .	24
3.2	Functional requirements . . . . .	24
3.2.1	Scenarios . . . . .	24
3.2.2	Use case diagrams . . . . .	26
3.2.3	Use cases . . . . .	28
3.2.4	Sequence diagrams . . . . .	39
3.2.5	Mapping requirements . . . . .	47
3.2.6	Traceability matrix . . . . .	51

3.3	Performance requirements . . . . .	51
3.4	Design constraints . . . . .	51
3.4.1	Standard compliance . . . . .	51
3.4.2	Hardware limitations . . . . .	52
3.5	Software systems attributes . . . . .	52
3.5.1	Reliability . . . . .	52
3.5.2	Availability . . . . .	52
3.5.3	Security . . . . .	52
3.5.4	Maintainability . . . . .	52
3.5.5	Portability . . . . .	52
<b>4</b>	<b>Formal analysis using Alloy</b>	<b>53</b>
4.1	World model . . . . .	54
4.2	Generated world . . . . .	61
4.2.1	Generic world . . . . .	61
<b>5</b>	<b>Effort spent</b>	<b>63</b>
<b>6</b>	<b>References</b>	<b>64</b>

# 1. Introduction

## 1.1 Purpose

### 1.1.1 General purpose

SafeStreets is a crowd-sourced application that wants to provide users with the possibility to notify authorities when parking violations occur.

The application will allow users to send pictures of violations, including their date, time, and position. Then authorities will be able to check, validate and eventually use those data for issuing tickets.

To achieve this objective many goals have been identified, this vital aims will have to be met before the release of the product.

In short, the S2B will satisfy the following goals:

- [G1] The system must allow logged-in users to send a report of the violation
- [G2] The system must allow logged-in users to see their past reports
- [G3] The system must allow logged-in users to retrieve information about the position and types of valid reports
- [G4] The system must allow verified authorities to mine information about date, time, position and type of valid reports
- [G5] The system must allow verified authorities to retrieve statistics about valid reports
- [G6] The system must be able to cross the data retrieved from the municipality with its own, in order to identify unsafe areas and suggest possible interventions to the municipal employee
- [G7] The system must allow local officer to set the validity of a report sent by the user
- [G8] The system must ensure that the chain of custody of the information coming from the user to the municipality is never broken, and the information is never altered

W.r.t. [G6] an area is considered "unsafe" if and only if a minimum number of problems is met on that area.

### **1.1.2 Document purpose**

This document represents the Requirement Analysis and Specification Document (RASD). This document aims to completely describe the system in terms of functional and non-functional requirements, analyze the real needs of the users to model the system, show the constraints and the limit of the software and indicate the typical use cases that will occur after the release.

This document is addressed to the developers who have to implement the requirements and could be used as a contractual basis.

## **1.2 Scope**

This service is born from the idea that social responsibility on the street can be achieved with the help of everyday citizens. Such an objective is achievable by giving good-willed people the possibility to record parking violations that they spot on the street and making them visible to the authorities later. To this kind of people, regarded as unregistered users before their subscription to the service, the choice of signing up is given.

When an unregistered user signs up, he/she will become a registered user, able to log in whenever desired. A logged-in registered user, to employ the functionalities of the system, must have at least a mobile phone with a camera and a GPS localization system, otherwise, the product won't be available for use. With the minimum requirements satisfied the user will be able to compile and send reports of the parking violations and, if interested, search in a selected area for violations that will be shown as dots on an interactive map provided by the map service.

Each report will be composed by the type of the violation, i.e. "vehicle parked in a forbidden area", by a picture of the vehicle with its license plate highlighted, that will be later recognized and added to the report with an OCR service, by the date, hour and position where the picture has been taken. Registered users will also be able to see their past reports that, if the authority has already judged them as genuine, will be recognized as valid and highlighted with a green check. Invalid reports will be recognizable by a red cross and reports which evaluation is still pending will be represented by a yellow clock.

The authorities, embodied by the municipal employees and local officers, will be able to retrieve data from the system using any available device capable of connecting to the internet and running a browser. In particular, both the municipal employees and the local officers will be able to extract the reports sent by the users as one or more reports, choosing time, date, area or type of violation. If a group of reports has been chosen they will be showed in the same kind of way as a user, like dots on a map, but more information will be made available. Furthermore, both the municipal employees and the local officers will be able to retrieve statistics, derived from data collected by the system. The system data is collected with the registered users' reports, along with the ticket and accident information possessed by the municipality. If the authority possesses such data, it will be retrieved via a ticket service, that will

fetch date, time, position and violation type of the vehicle found committing the infraction, and a municipal accident service, that will recover information about road accidents, such as the date, time and the vehicles involved in the accident. Moreover the municipal employee, by selecting a position, will be able to search for possible improvements on such position, to see what kind of interventions should be made, and, if some of them have already been completed, he/she can change their status from "not done" to "done", a "done" improvement won't be shown again.

The last functionality is dedicated to the local officers: after having withdrawn a report, the local officer is able to check its validity, and eventually utilize its data to fine the vehicle that committed the violation.

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

- **Report:** Collection of Data that represents a Violation, in particular
  - Picture: a photo of the vehicle that has been found committing a violation
  - Date: the date when the picture has been taken
  - Time: the hour when the picture has been taken
  - Position: the place, formatted using GPS location, of the vehicle that has been found committing a violation.
- **Improvement:** a possible road intervention finalized to the development of the road and to achieve a safer environment

### 1.3.2 Acronyms

- UU = Unregistered User;
- RU = Registered User;
- ME = Municipal Employee;
- LO = Local Officer;
- S2B = Software to Be;
- TS = Ticket Service;
- MS = Map Service;
- MAS = Municipal Accident Service;
- OCRS = OCR Service;
- VT = Violation type;

For a precise description of RU, ME and LO see section 2.3

### 1.3.3 Abbreviations

- [Gn] : n-th goal.
- [Dn] : n-th domain assumption
- [Rn] : n-th functional requirement

## 1.4 Revision history

- Version 1.0
  - Initial release
- Version 1.1
  - Changed document appearance
  - Fixed names and coherence between chapters
  - Partial revision of use cases and scenarios
  - Fixed alloy model and description
  - Partial revision of the domain assumptions and requirements w.r.t. the authorities' credentials
  - Expansion of the security performance section
  - Correction of loops in sequence diagrams
  - Correction of useless requirements about issued tickets (old R20)
  - Correction of report validation, with valid report sent to the ticket service

## 1.5 Reference documents

- *Specification document:* "Mandatory Project Assignment AY 2019/2020"

## 1.6 Document structure

The RASD document is composed by six chapters, as outlined below:

**Chapter 1** describes the purpose of the system and the list of goals that the application has to reach. Moreover, it defines the scope, where the aim of the project is defined and the application domain with the shared phenomena are shown.

**Chapter 2** offers an overall description of the project. Here the actors, involved in the application's usage, are identified and the boundaries of the project are defined, listing all the necessary assumptions. Moreover, a class diagram is provided, to better understand the general structure of the project. Then some state diagrams are listed to make the evolution of the crucial objects and actors 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 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 primary 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** shows the effort which each member of the group has spent working on the project

**Chapter 6** contains eventual references used during the writing of the document

## 2. Overall description

## 2.1 Product perspective

The product will be developed from scratch and will be organized in two different macro-entities, one dedicated to the normal user, the other dedicated to the authorities, that may be MEs or LOs (for more information on the users see section 2.3).

The two parts will be differentiated to satisfy different needs of different parties that will employ SafeStreets. Their form, along with one of some other parts of the system, is represented with the following high-level class diagram.

### 2.1.1 Class diagram

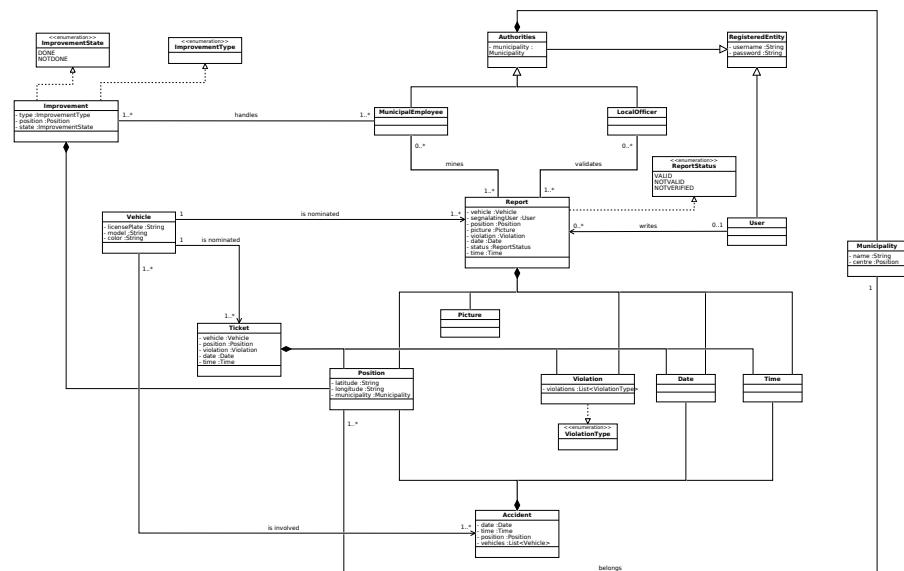


Figure 2.1: Class diagram

## 2.1.2 State diagrams

Some of the entities of the class diagram evolve, assuming different states, while the system is being used. The following diagrams show those states that will be found.

### 2.1.2.1 User state diagram

These diagrams are assumed by all users (authorities and normal users)

This diagram does not have an end state because, once a user has been correctly registered in the system, its presence in the system will ideally never disappear.

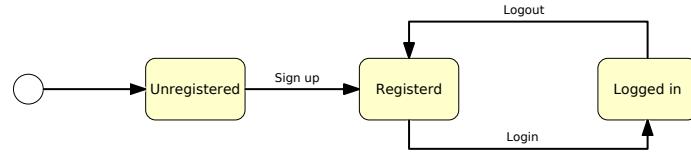


Figure 2.2: User state diagram

### 2.1.2.2 Report construction state diagram

This diagram contains all the states that the system will assume while a user creates a report. Not all of the transitions are caused by the user, in fact, in the "Impending GPS localization" state the system will independently retrieve the location, it is possible also to notice that in this state, unlike all the others, both the abort and go back transitions are missing. Thanks to the domain assumptions (see section 2.4) the system will never be stuck on the "Impending GPS localization" state, giving only to the user the choice of reaching one of the two final states.

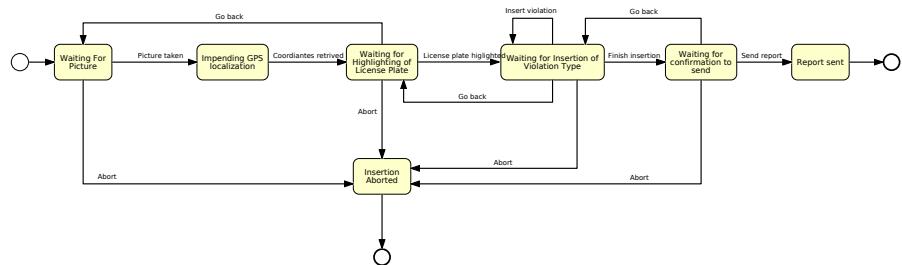


Figure 2.3: Report construction state diagram

### 2.1.2.3 Report state diagram

This diagram represents the state of a report from when it is received by the system to the evaluation of a LO. When a report arrives to the system, the

license plate, that has been already highlighted by the user, will be run through the OCRS, if the plate is regarded as illegible the report will be automatically regarded as not valid, otherwise the choice of the final state of the report will be taken by the LO.

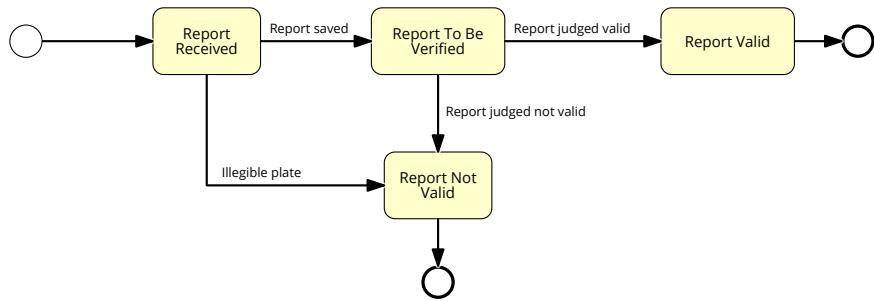


Figure 2.4: Report state diagram

#### 2.1.2.4 Improvement state diagram

This diagram contains the states of an improvement from when is framed to when it is completed. Is possible that some of the improvements will never be completed, remaining forever in the not done state, but if one is indeed finished, and set as done by a ME, it will be discarded and never shown again (only on the street where it was proposed)

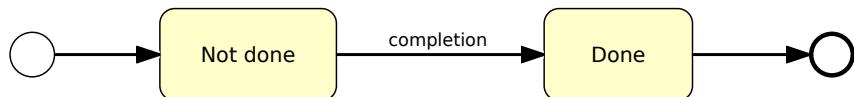


Figure 2.5: Improvement state diagram

### 2.1.3 World and Phenomena

Phenomenon	Shared	Who
UU inserts its credentials	Y	W
S2B checks UU credentials	N	M
RU/ME/LO logs in	Y	W
S2B checks RU/ME/LO credentials	N	M
RU wants to add a report	N	W
RU takes a picture of the violation	Y	W
RU choose the type of violations	Y	W
RU sends a report	Y	W
S2B registers a report	N	M
RU wants to request its reports	N	W
RU requests its reports	Y	W
RU asks the S2B for seeing reports by area	Y	W
S2B selects the reports for the RU	N	M
S2B provides RU with requested reports	N	M
ME/LO wants to mine reports	N	W
ME/LO chooses the way of mining reports	Y	W
ME/LO asks the S2B for the mining	Y	W
S2B selects the reports for the ME/LO	N	M
S2B provides ME/LO with requested reports	Y	M
ME/LO wants to retrieve statistics	N	W
ME/LO asks the S2B for statistics	Y	W
S2B crunches statistics for the ME/LO	N	M
S2B provides ME/LO with statistics	Y	M
ME wants go get the possible improvements	N	W
ME asks the S2B for possible improvements	Y	W
S2B finds possible improvements for the ME's municipality	N	M
S2B provides ME with the possible improvements	Y	M
ME marks an improvement as done or not done	Y	W
S2B changes the status of the improvement as done or not done	N	M
LO wants to validate one or more reports	N	W
LO asks the S2B for validating reports	Y	W
S2B provides LO with the reports the reports to be verified	Y	M
LO marks a report as valid or not valid	Y	W
S2B changes the status of the report as valid or not valid	N	M

Table 2.1: World and phenomena

## 2.2 Product functions

In this section the most important functions of the system are reported.

### 2.2.1 Report

The core of the system revolves around the management of reports. Reports, that are structures containing information about a violation, are created and then saved in the system with an "add" function. The "add" function builds the report, assembling the picture of the car that has committed a violation, with its license plate, date and time, GPS location, and violation type, sending it to SafeStreets at the end of the procedure.

When a report is received, the system checks if the license plate is readable, if not the report is discarded as "not valid", otherwise a set of possible operations becomes available. These operations are the "validate" and "mine" functions. The "validate" function shows every stored report, which status is set as "to be verified", with all the information listed above, and permits to change their status to "valid", if the report is considered legit, or "not valid", in the opposite case. The "mine" functions find the report type, of existing valid reports, searching in the system using type, date, time or area as parameters for the query.

### 2.2.2 Improvement

Using both data provided by the authorities and the data owned by SafeStreet is possible to identify the unsafe areas of a Municipality. Using the improvement function makes it possible to determine feasible solutions that can be used to improve the safety of such areas, i.e. add a barrier between the bike lane and the part of the road for motorized vehicles to prevent unsafe parking.

### 2.2.3 Statistics

With the information about issued tickets coming from the Ticket Service and SafeStreets' own data, is possible to build statistic about the violations and the perpetrators who cause them.

## 2.3 User characteristics

There are three kind of users that will employ this product:

**User:** the normal, everyday citizen that has subscribed to SafeStreets and recognized as RU. The RU is able to compile and send reports, see the chronology of his reports and search for violations in a selected area. Other functions will not be accessible by the user to protect the privacy of other people, not necessary other RU, and to avoid providing excessive useless data to RU.

**Municipal Employee:** the ME is someone hired from the municipality to which the task of accessing information, on behalf of the authorities, will be given. To such kinds of individuals, a unique username and password

will be provided, already linked to an account able to utilize all functions dedicated to the retrieval of data. A ME is unable to add or modify the status of any reports but is able to change the status of an improvement from "not done" to "done".

**Local Officer:** The LO is part of the law enforcement of the municipality. As the ME, the LO receives his own account information from the state authority that will permit him/her to utilize all the functions to retrieve data. Unlike the ME, the LO is able to modify the state of a report but is unable to do so with an improvement. A LO may, or may not, utilize the data stored by SafeStreet to write tickets for the cars that committed a violation

## 2.4 Assumptions, dependencies and constraints

To assure the correct formulations of the requirements, and avoid unforeseen events, a certain amount of the world phenomena is considered as follows.

- [D1] The number of possible violations is finite and is aligned to the current traffic rules
- [D2] The number of possible interventions is finite and there exists an already established correlation between violations and possible interventions
- [D3] When using the S2B, the user's device is always connected to internet
- [D4] When using the S2B, the user's device has a valid GPS signal
- [D5] The internet connection works properly without failure
- [D6] The user device has a camera and is able to take pictures
- [D7] The user does not fake his position
- [D8] Every location has one and only one municipality
- [D9] Car plates are unique
- [D10] Each authority has its own account, certified and authorized by a state authority, and linked with the municipality which he/she works for
- [D11] The official credentials of each municipal employee are different from those used for report violations (if he/she has registered as a normal user)
- [D12] The official credentials of each local officer are different from those used for report violations (if he/she has registered as a normal user)
- [D13] The state authority voids credentials of municipal employees or local officers at the end of their service
- [D14] When using the S2B, the authority's device is always connected to internet

# 3. Specific requirements

## 3.1 External interface requirements

### 3.1.1 User interfaces

The following mockups give an approximate idea of how the application's interfaces should appear.

#### 3.1.1.1 Unregistered User



Figure 3.1: UU SignUp mockup

### 3.1.1.2 Registered User



Figure 3.2: User login mockup

### Get my reports, all reports

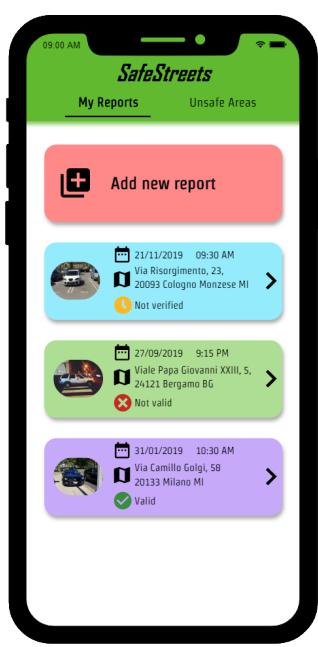


Figure 3.3: Get my reports (all) mockup

### Get my reports, information of a single report

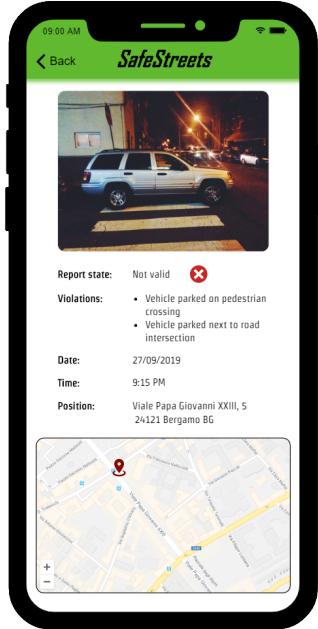


Figure 3.4: Get my reports (detail) mockup

### Get unsafe areas

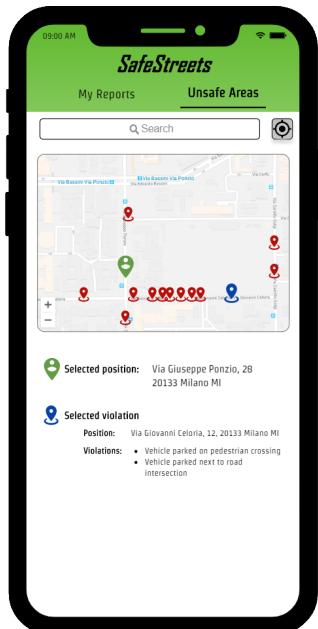


Figure 3.5: Get unsafe areas mockup

**Add reports, taking picture from camera**



Figure 3.6: Add report (take picture) mockup

**Add report, confirmation of taken picture**



Figure 3.7: Add report (confirm picture) mockup

### Add report, highlighting of the vehicle's plate



Figure 3.8: Add report (highlight plate) mockup

### Add report, choosing of the violations' type

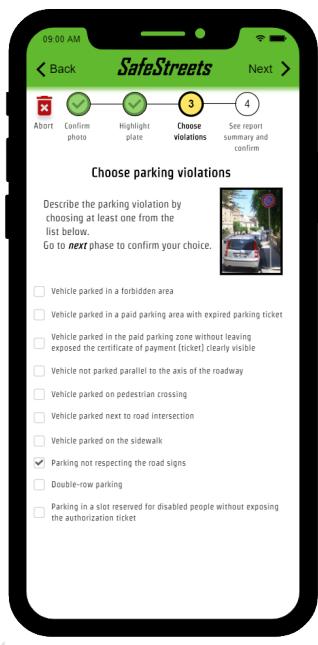


Figure 3.9: Get unsafe areas mockup

### Add report, confirmation

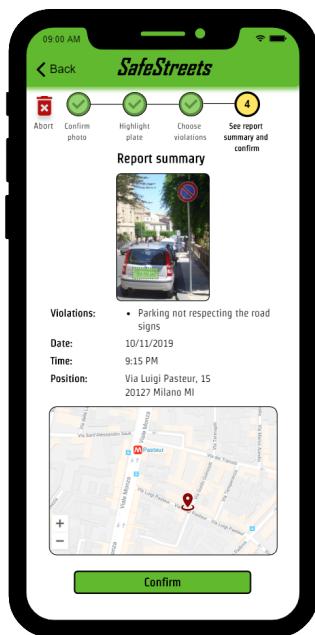


Figure 3.10: Add report (confirm) mockup

#### 3.1.1.3 Authority

##### Authority login



Figure 3.11: Authority login mockup

### 3.1.1.4 Municipal Employee

Home page



Figure 3.12: Municipal Employee home page mockup

Get improvements

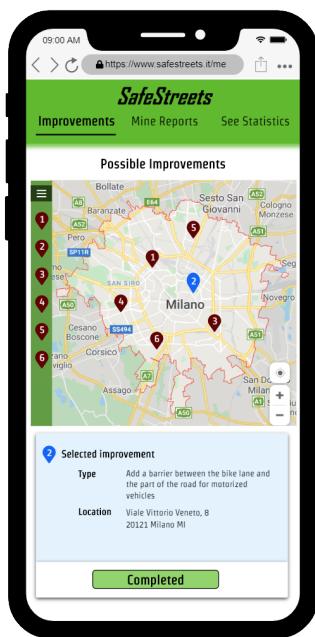


Figure 3.13: Get improvements mockup

### Mine reports by\*

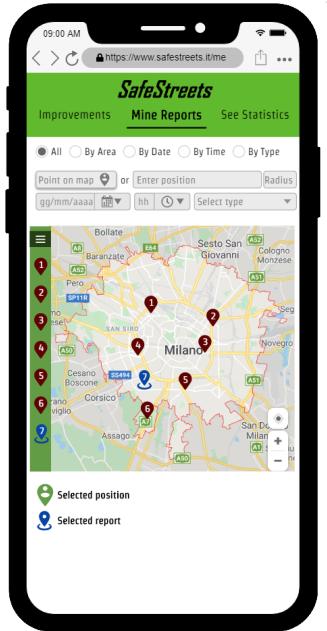


Figure 3.14: Municipal Employee mine reports by\* mockup

### See statistics



Figure 3.15: See statistics mockup

### 3.1.1.5 Local Officer

#### Home page



Figure 3.16: Local Officer home page mockup

#### Validate reports, compact view

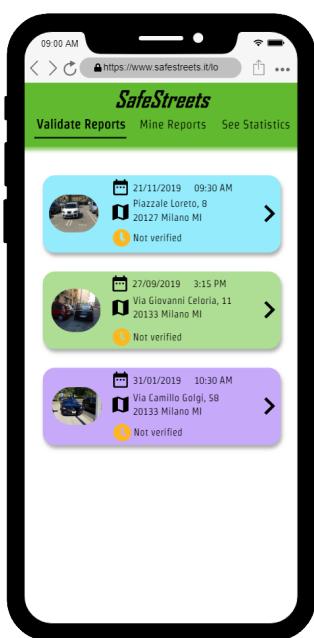


Figure 3.17: Validate reports (all) mockup

### Validate report, expanded view

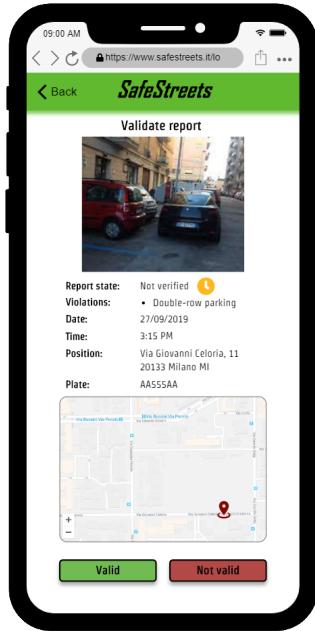


Figure 3.18: Validate report (detail) mockup

### Mine reports by\*



Figure 3.19: Local Officer mine reports by\* mockup

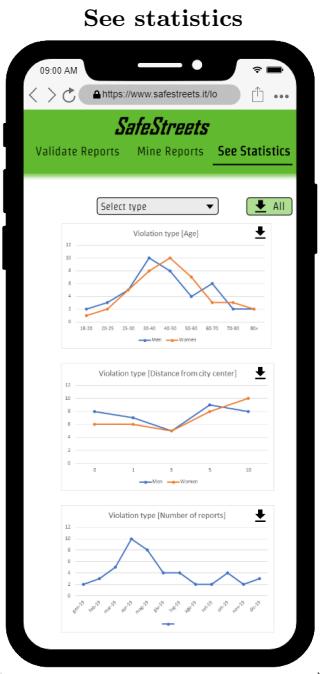


Figure 3.20: Local Officer see statistics mockup

### 3.1.2 Hardware interfaces

The system has no hardware interface.

### 3.1.3 Software interfaces

The system does not offer any APIs to external applications for its usage.

### 3.1.4 Communication interfaces

The system does not offer any APIs to external applications for communication.

## 3.2 Functional requirements

### 3.2.1 Scenarios

#### 3.2.1.1 Scenario 1

Cat has a disabled child. Tired from everybody that leaves their car in the disable's parking lot, Cat decides to download and install the SafeStreets app. After subscribing to the service, she starts to take pictures of all the perpetrator's cars. Some time after, Cat notices that her reports are getting validated and along with that the number of times that she finds cars in the disabled parking lot starts to diminish, that may be because the local police intervened and ticketed them.

### **3.2.1.2 Scenario 2**

Ned is looking for a house in the city, he has always lived in the countryside and he's scared of the possibility to live in areas with difficulty in finding car parks. He opens SafeStreets and using "Get unsafe areas" and, by selecting areas around his favourite house on sale, he can find what happened there and he can avoid the worst areas, looking for a house with confidence.

### **3.2.1.3 Scenario 3**

Jon has a reputation for not being a trustworthy guy. When he decides to use SafeStreets, Jon starts to report every car that he does not like, hoping that the owners of those car will receive a ticket. Luckily for the owners, those report won't incur in any fine because the local police is able to validate the received reports and does not issue the cars in order, ignoring Jon's report.

### **3.2.1.4 Scenario 4**

Martha works as a municipal employee. The mayor asks her to make a list of all possible investment to develop the streets of the city, in order to increase street security. Luckily for Martha, SafeStreets with its function "Get Improvements", highlights the most unsafe areas and suggests possible interventions. Reading these recommendations, Martha realizes that some of them have already been done so she diligently set them as completed. Martha can now finish her assignment quickly.

### **3.2.1.5 Scenario 5**

The mayor, to promote his image before the next elections, wants to publish a report with the statistics regarding public security. With SafeStreets his secretary can easily retrieve such information and prepare a detailed report with all the statistics about how public street security has improved during the mayor's mandate.

### **3.2.1.6 Scenario 6**

A group of citizens is concerned with the street security of their neighborhood but they don't have any real data that can be used to support their concerns. They go to the municipality to ask for more information and possibly an intervention of the authorities. The municipality can quickly search for data regarding their area and forward it to higher authorities that will hopefully do something.

### **3.2.1.7 Scenario 7**

A local officer in patrol is tasked to give fines during his work shift but usually, this isn't such an easy task. Normally he would spend a lot of time searching the usual street but, using SafeStreets, he can now retrieve reports about positions of parking violations using "Mine Reports" function. With this information, the local officer starts the search. Luckily for him, he finds a lot of violations in the streets specified by SafeStreet, so he can proceed and charge the owners of those cars for their sins (violations).

### 3.2.2 Use case diagrams

#### 3.2.2.1 Unregistered User

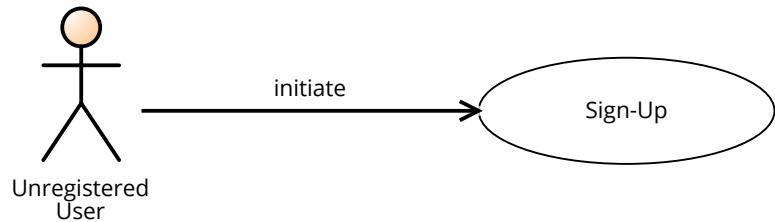


Figure 3.21: Unregisterd User use case diagram

#### 3.2.2.2 Registered User

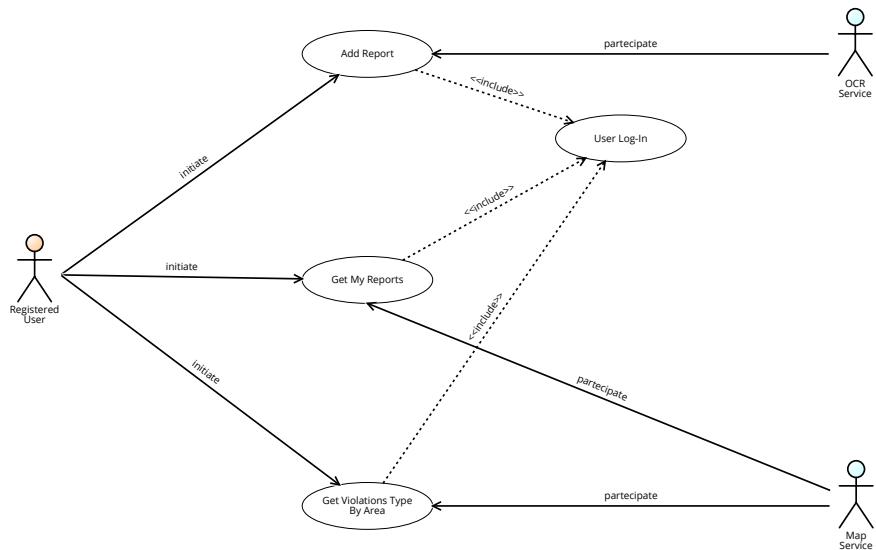


Figure 3.22: Registered User use case diagram

### 3.2.2.3 Municipal Employee

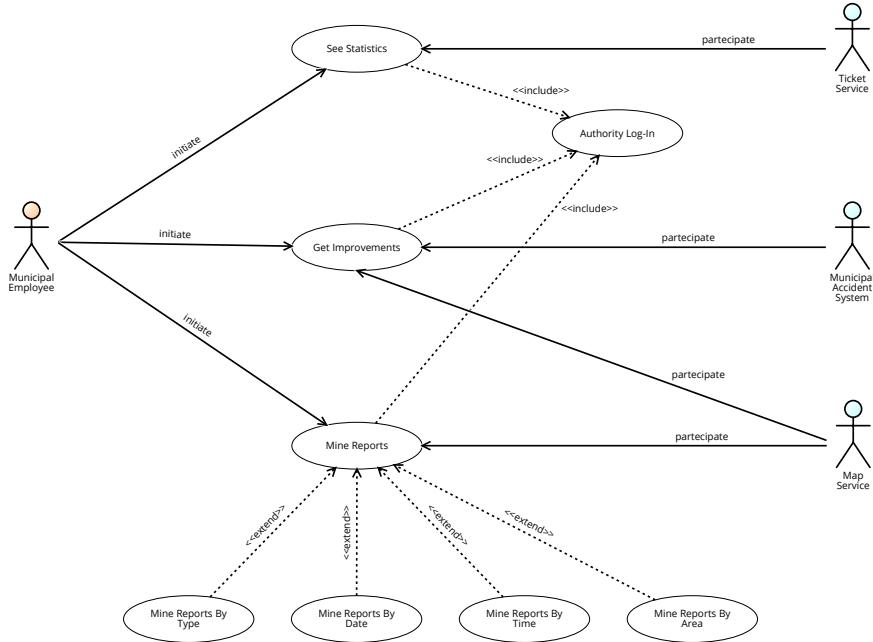


Figure 3.23: Municipal Employee use case diagram

### 3.2.2.4 Local Officer

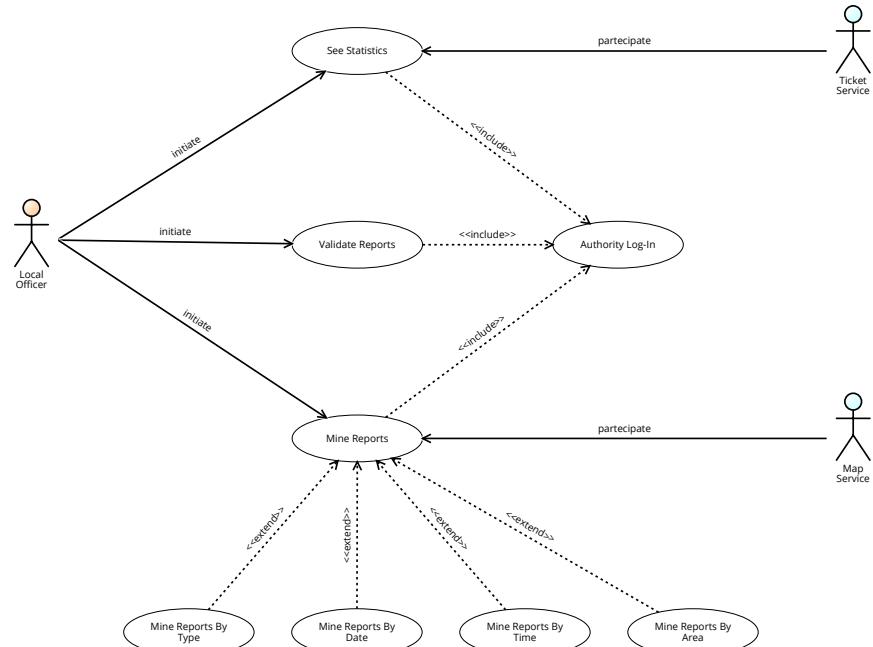


Figure 3.24: Local Officer use case diagram

### 3.2.3 Use cases

#### 3.2.3.1 Sign-Up

Name	Sign-Up
<b>Actors</b>	Unregistered User
<b>Entry Conditions</b>	The UU is already on the login page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The UU clicks on the "Sign-Up" field to start the registration process</li> <li>2. The UU provides his/her email address</li> <li>3. The UU provides a username</li> <li>4. The UU provides a password and writes it again for confirmation</li> <li>5. The UU clicks on the confirmation button</li> <li>6. The system saves the data</li> </ol>
<b>Exit Conditions</b>	The UU becomes now a RU. From now on he/she can log into the application and use SafeStreets service.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. The UU provides an already used email address</li> <li>2. The UU provides an already used username</li> <li>3. The two passwords provided by the UU are not corresponding</li> </ol> <p>In the case 1 an error message is displayed, saying "Credentials already in use, please log in", and the UU is taken back to the login page.</p> <p>In the case 2 an error message is displayed, saying "Username already in use, please choose a different one", and the UU is taken back to the point 2.</p> <p>In the case 3 an error message is displayed, saying "Email already in use, please log in", and the UU is taken back to the login page.</p>

Table 3.1: Sign up use case

### 3.2.3.2 Registered User login

<b>Name</b>	<b>Registered User login</b>
<b>Actors</b>	Registered User
<b>Entry Conditions</b>	The RU is already on the login page
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The RU provides his/her username or email.</li> <li>2. The RU provides his/her password.</li> <li>3. The RU clicks on the confirmation button</li> <li>4. The system redirects the RU to the corresponding home page.</li> </ol>
<b>Exit Conditions</b>	The RU is successfully redirected to the corresponding home page.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. The RU provides an incorrect username, email or password. In the case 1 an error message is displayed, saying "Wrong Credentials", and the RU is taken back to the point 1.</li> </ol>

Table 3.2: Registered User login use case

### 3.2.3.3 Add report

<b>Name</b>	<b>Add report</b>
<b>Actors</b>	Registered User, OCR Service, Map Service
<b>Entry Conditions</b>	The RU has logged in and is in the home page, which is the "Get My Reports" page
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The RU clicks on the "Add Report" button.</li> <li>2. The RU takes a photo of the vehicle through his/her device's camera and confirms the picture</li> <li>3. The RU highlights the plate of the reported vehicle and goes to the next phase.</li> <li>4. The RU adds one or more type of violations and confirms.</li> <li>5. The RU sees the complete report and confirms.</li> <li>6. The system receives the report and stores it, with the eventual plate recognized thanks to the OCRS.</li> </ol>
<b>Exit Conditions</b>	The RU successfully uploads a new report into the system. Then he/she is taken back to the home page.
<b>Exceptions</b>	There are no exceptions under the given domain assumptions.

Table 3.3: Add report use case

### 3.2.3.4 Get my reports

Name	Get my reports
<b>Actors</b>	Registered User, Map Service
<b>Entry Conditions</b>	The RU has logged in and is in the home page, which is "Get My Reports" page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The system automatically provides the RU with the reports he/she has sent, sorted in chronological order, starting from the newest one, preceded by the "Add Report" button.</li> <li>2. Eventually the RU is able to open a report, see all its information, with the position provided by the MS, and see its status.</li> </ol>
<b>Exit Conditions</b>	The RU successfully gets his/her reports and is able to navigate through them.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. The RU has never uploaded any report The case 1 is handled by showing only the "Add Report" button.</li> </ol>

Table 3.4: Get my reports use case

### 3.2.3.5 Get unsafe areas

Name	Get unsafe areas
<b>Actors</b>	Registered User, Map Service
<b>Entry Conditions</b>	The RU has logged in and is in the home page, which is the "Get My Reports" page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The RU has to swipe to the "Unsafe areas" page.</li> <li>2. Once in the correct page, the RU has to provide an address or use his/her current GPS location</li> <li>3. A map with the violations, shown as points, is displayed. The map is provided by the MS</li> </ol>
<b>Exit Conditions</b>	The RU now is able to browse on the map between violations and see their type.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. The RU provides an invalid address In the case 1 an error message is displayed, saying "Invalid address", and the RU is taken back to the point 2.</li> </ol>

Table 3.5: Get unsafe areas use case

### 3.2.3.6 Authority login

Name	Authority login
<b>Actors</b>	Municipal Employee, Local Officer
<b>Entry Conditions</b>	The ME/LO is already on the login page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO provides his/her username.</li> <li>2. The ME/LO provides his/her password.</li> <li>3. The ME/LO chooses his/her type: "Municipal Employee" or "Local Officer".</li> <li>4. The ME/LO clicks on the confirmation button.</li> <li>5. The system redirects the ME/LO to the corresponding home page.</li> </ol>
<b>Exit Conditions</b>	The ME/LO is successfully redirected to the home page.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. The ME/LO provides an incorrect username or password.</li> </ol> <p>In the case 1 an error message is displayed, saying "Invalid credentials", and the ME/LO is taken back to the point 1.</p>

Table 3.6: Authority login use case

### 3.2.3.7 See statistics

Name	See statistics
<b>Actors</b>	Municipal Employee, Local Officer, Ticket Service
<b>Entry Conditions</b>	The ME/LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO clicks on the "See Statistics".</li> <li>2. The system creates statistics using stored valid reports and issued tickets from Ticket Service.</li> <li>3. The system sends the statistics to the ME/LO.</li> <li>4. The ME/LO chooses the type of statistics to retrieve.</li> <li>5. The ME/LO can see and download the received statistics.</li> </ol>
<b>Exit Conditions</b>	The ME/LO successfully receives the statistics provided by the system.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. The system has no data to build statistics on.</li> </ol> <p>In the case 1 an error message is displayed, saying "There is not enough data to create statistics", and the ME is taken back to the home page.</p>

Table 3.7: See statistics use case

### 3.2.3.8 Get improvements

Name	Get improvements
<b>Actors</b>	Municipal Employee, Municipal Accident Service, Map Service
<b>Entry Conditions</b>	The ME has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME clicks on the "Get Improvements" function.</li> <li>2. The system cross its information with the MAS and finds all the possible improvements in the ME's municipality.</li> <li>3. The system sends the information to the ME.</li> <li>4. The possible improvements are shown in a descendant list on the left, starting from the most urgent one. A map with the possible improvements, shown as points, is displayed on the right, thanks to the MS.</li> <li>5. The ME can click on a improvements and see it highlighted on the map and see its detail below.</li> </ol>
<b>Exit Conditions</b>	The ME can browse through all the possible improvements in his/her municipality.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no possible improvements. In the case 1 an error message is displayed, saying "There is not enough data to generate improvements", and the ME/LO is taken back to the home page.</li> </ol>

Table 3.8: Get improvements use case

### 3.2.3.9 Mine reports by all

Name	Mine reports by all
<b>Actors</b>	Municipal Employee, Local Officer, Map Service
<b>Entry Conditions</b>	The ME/LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO clicks on the "Mine Reports" function.</li> <li>2. The ME/LO chooses the "Mine By All" option.</li> <li>3. The system provides the ME/LO with all the violations in his/her municipality.</li> <li>4. Only the violation types are shown, in a list on the left, starting from the newest one. A map with violation types, shown as points, is displayed on the right, thanks to the MS.</li> </ol>
<b>Exit Conditions</b>	The ME/LO is able to search through the violations and see where and when they happened.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no reports for the ME/LO municipality.</li> </ol> <p>In the case 1 an error message is displayed, saying "There are no Violations to be displayed", and the ME/LO is taken back to the home page.</p>

Table 3.9: Mine reports by all use case

### 3.2.3.10 Mine reports by type

Name	Mine reports by type
<b>Actors</b>	Municipal Employee, Local Officer, Map Service
<b>Entry Conditions</b>	The ME/LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO clicks on the "Mine Reports" function.</li> <li>2. The ME/LO chooses the "Mine By Type" option.</li> <li>3. The ME/LO decides one or more types of violations to retrieve.</li> <li>4. The system provides the ME/LO with all the violations in his/her municipality, with at least one of the given types.</li> <li>5. Only the violation types are shown, in a list on the left, starting from the newest one. A map with violation types, shown as points, is displayed on the right, thanks to the MS.</li> </ol>
<b>Exit Conditions</b>	The ME/LO is able to search through the violations and see where and when they happened.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no reports with at least one of the given violation type.</li> </ol> <p>In the case 1 an error message is displayed, saying "There are no Violations to be displayed", and the ME/LO is taken back to the home page.</p>

Table 3.10: Mine reports by type use case

### 3.2.3.11 Mine reports by date

Name	Mine reports by date
<b>Actors</b>	Municipal Employee, Local Officer, Map Service
<b>Entry Conditions</b>	The ME/LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO clicks on the "Mine Reports" function.</li> <li>2. The ME/LO chooses the "Mine By Date" option.</li> <li>3. The ME/LO decides the specific date of the violations to retrieve.</li> <li>4. The system provides the ME/LO with all the violations in his/her municipality that were issued in the specific day.</li> <li>5. Only the violation types are shown, in a list on the left. A map with violation types, shown as points, is displayed on the right, thanks to the MS.</li> </ol>
<b>Exit Conditions</b>	The ME/LO is able to search through the violations and see where they happened.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no reports issued in the specific date. In the case 1 an error message is displayed, saying "There are no Violations to be displayed", and the ME/LO is taken back to the home page.</li> </ol>

Table 3.11: Mine reports by date use case

### 3.2.3.12 Mine reports by time

Name	Mine reports by time
<b>Actors</b>	Municipal Employee, Local Officer, Map Service
<b>Entry Conditions</b>	The ME/LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO clicks on the "Mine Reports" function.</li> <li>2. The ME/LO chooses the "Mine By Time" option.</li> <li>3. The ME/LO decides the time (expressed in hours, i.e. 8 pm) for the violations he/she wants to retrieve.</li> <li>4. The system provides the ME/LO with all the violations in his/her municipality that were issued at the specific time.</li> <li>5. Only the violation types are shown, in a list on the left. A map with violation types, shown as points, is displayed on the right, thanks to the MS.</li> </ol>
<b>Exit Conditions</b>	The ME/LO is able to search through the violations and see where they happened.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no reports issued at the specific time. In the case 1 an error message is displayed, saying "There are no Violations to be displayed", and the ME/LO is taken back to the home page.</li> </ol>

Table 3.12: Mine reports by time use case

### 3.2.3.13 Mine reports by area

Name	Mine reports by area
<b>Actors</b>	Municipal Employee, Local Officer, Map Service
<b>Entry Conditions</b>	The ME/LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The ME/LO clicks on the "Mine Reports" function.</li> <li>2. The ME/LO chooses the "Mine By Area" option.</li> <li>3. The ME/LO decides the address, or selects a point on a map provided by the MS, of the violations to retrieve and select the radius of the search.</li> <li>4. The system provides the ME/LO with all the violations in his/her municipality that were issued within the specified radius.</li> <li>5. Only the violation types are shown, in a list on the left, starting from the newest one. A map with violation types, shown as points, is displayed on the right, thanks to the MS.</li> </ol>
<b>Exit Conditions</b>	The ME/LO is able to search through the violations and see when they happened.
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no reports issued in the specific area. In the case 1 an error message is displayed, saying "There are no Violations to be displayed", and the ME/LO is taken back to the home page.</li> </ol>

Table 3.13: Mine reports by area use case

### 3.2.3.14 Validate reports

Name	Validate reports
<b>Actors</b>	Local Officer, Ticket Service
<b>Entry Conditions</b>	The LO has logged in and is in the home page.
<b>Event Flow</b>	<ol style="list-style-type: none"> <li>1. The LO clicks on the "Validate Report" function.</li> <li>2. The system provides the LO with all the reports of his/her municipality still not verified.</li> <li>3. The reports are shown starting from the newest one.</li> <li>4. The LO clicks on one report and is able to validate or invalidate it.</li> <li>5. Eventually the LO can use the report data to create a new ticket, so valid tickets are directly sent to the TS.</li> </ol>
<b>Exit Conditions</b>	The LO successfully set the status of report not validated
<b>Exceptions</b>	<ol style="list-style-type: none"> <li>1. There are no reports in LO's municipality. In the case 1 an error message is displayed, saying "There are no Reports to be displayed", and the LO is taken back to the home page.</li> </ol>

Table 3.14: Validate reports use case

### 3.2.4 Sequence diagrams

#### 3.2.4.1 Unregistered User

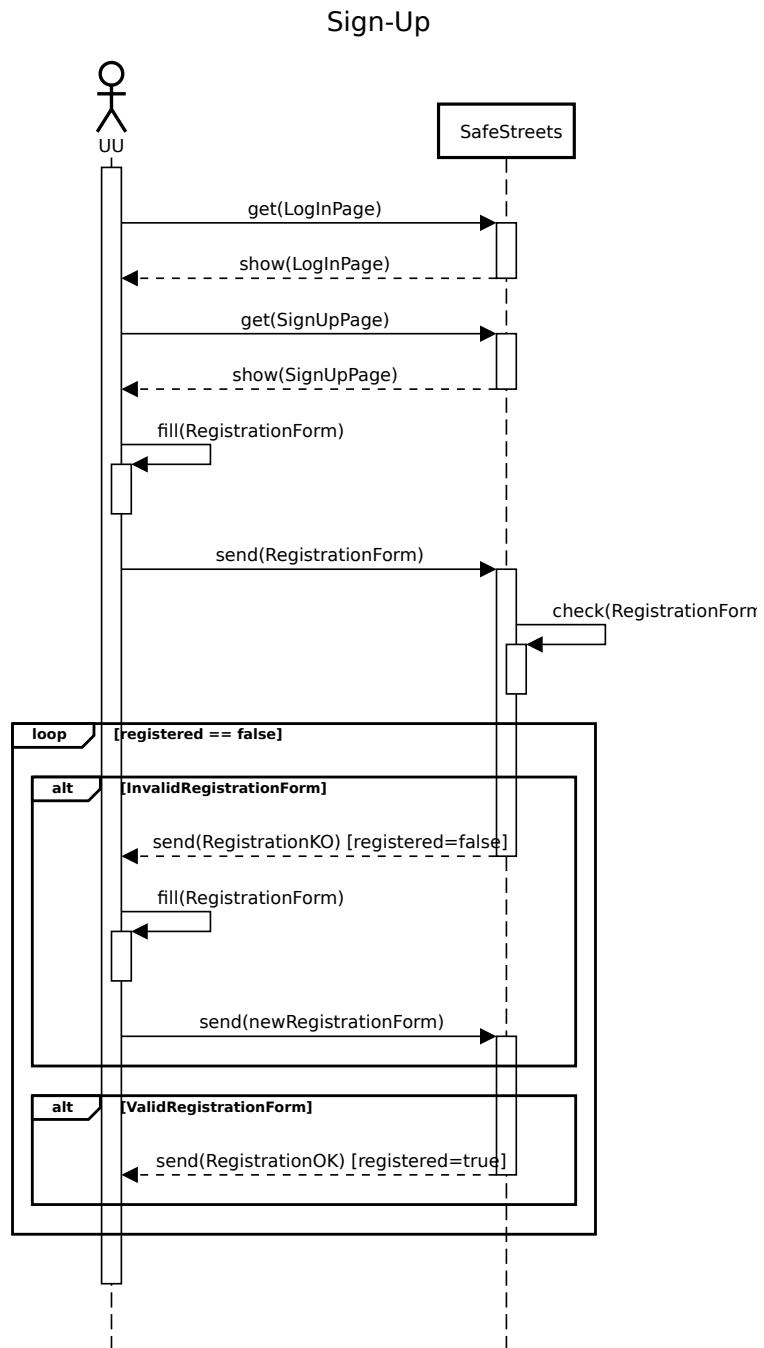


Figure 3.25: UU SignUp sequence diagram

### 3.2.4.2 Registered User

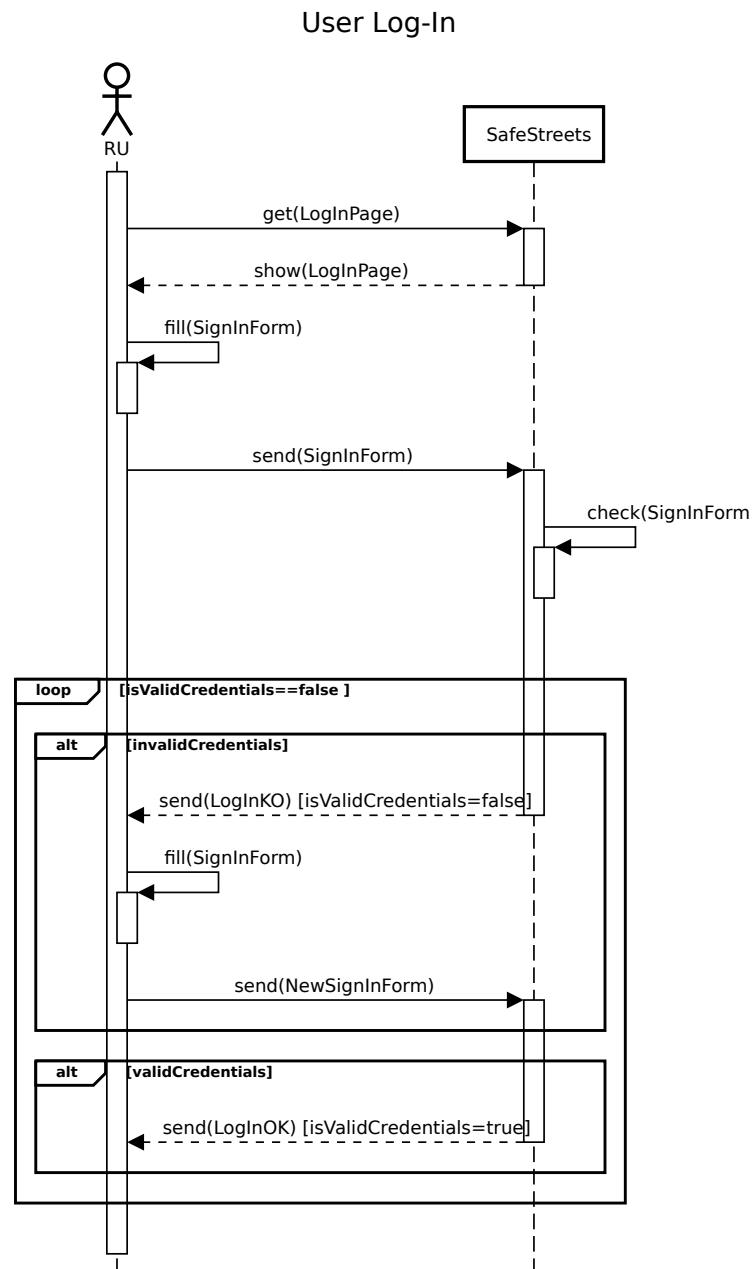


Figure 3.26: RU LogIn sequence diagram

### Get My Reports

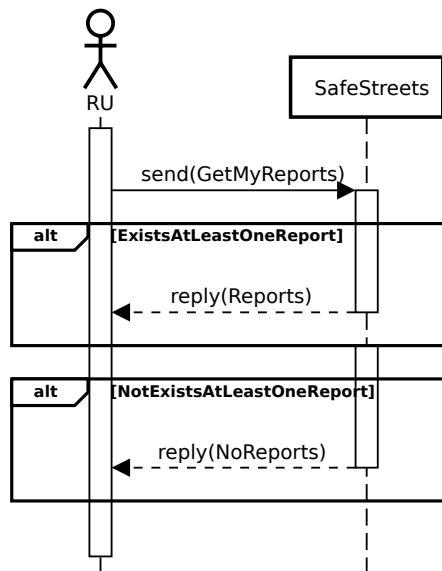


Figure 3.27: RU Get my reports sequence diagram

### Get Violations Type By Area

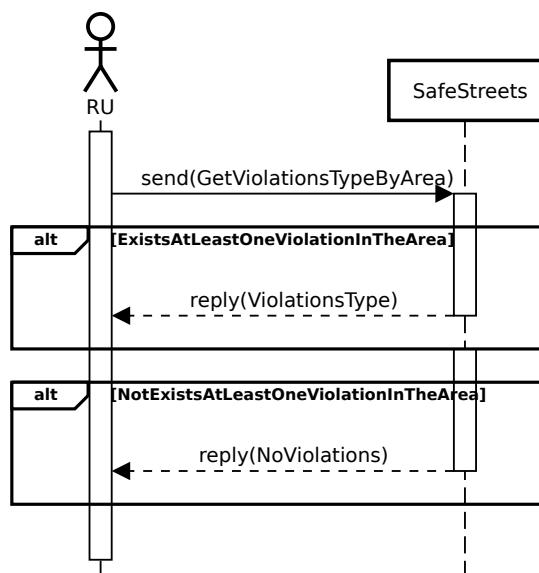


Figure 3.28: RU Get unsafe areas sequence diagram

### Add Report

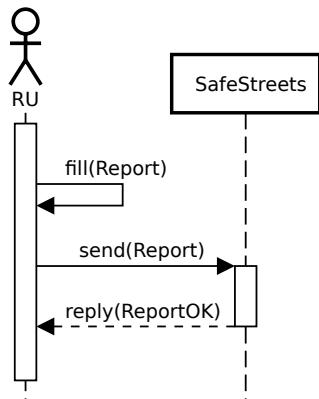


Figure 3.29: RU Add report sequence diagram

### 3.2.4.3 Authority

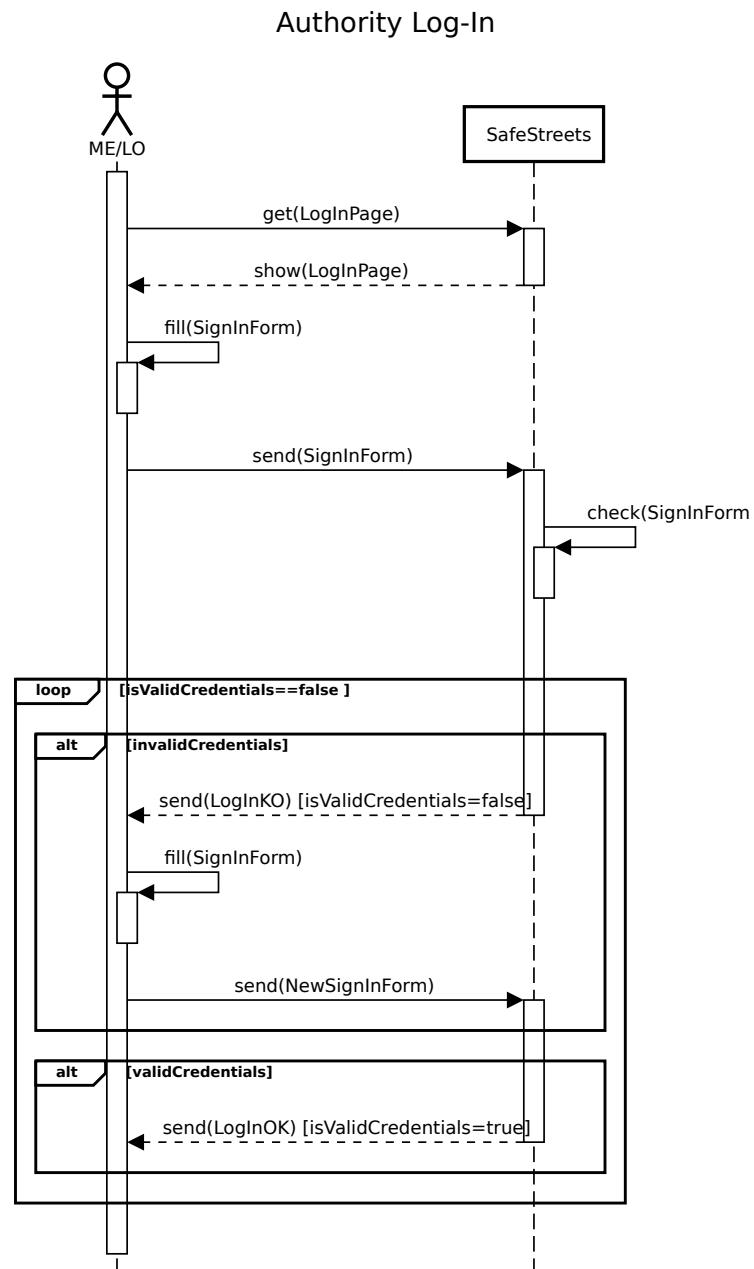


Figure 3.30: Authority LogIn sequence diagram

### Mine Reports By \*

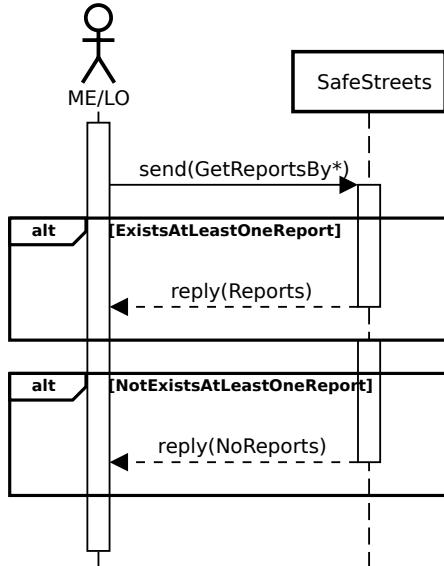


Figure 3.31: Authority Mine reports by\* sequence diagram

### See Statistics

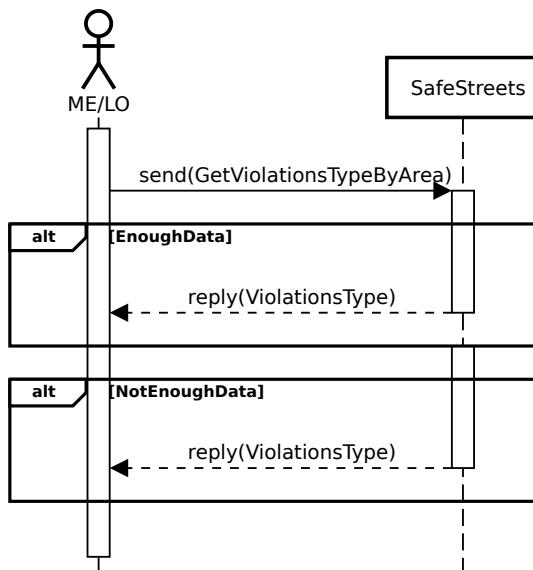


Figure 3.32: Authority See statistics sequence diagram

### 3.2.4.4 Municipal Employee

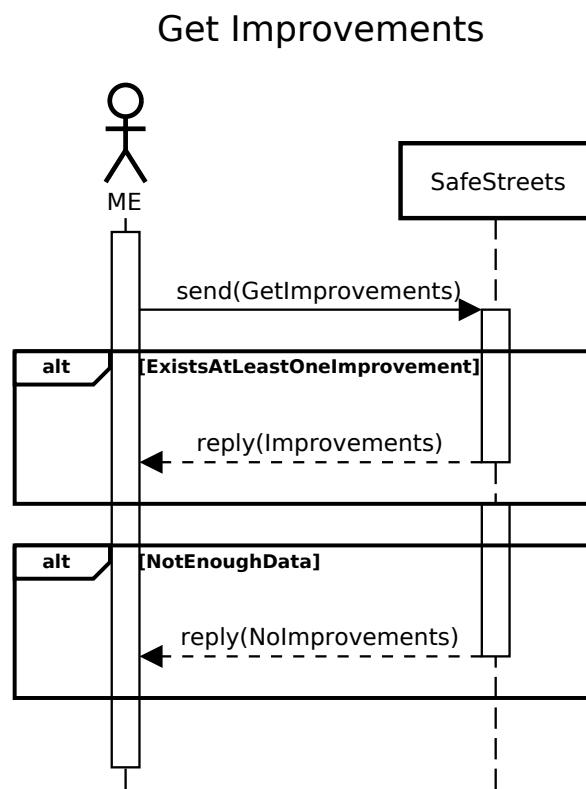


Figure 3.33: Municipal Employee Get improvements sequence diagram

### 3.2.4.5 Local Officer

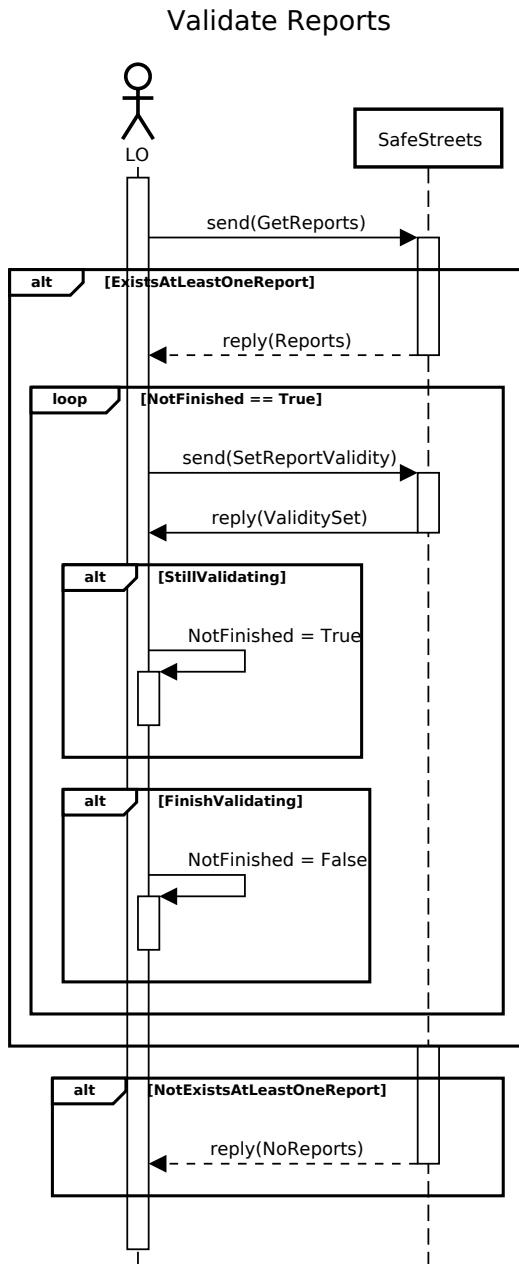


Figure 3.34: Local Officer Validate reports sequence diagram

### 3.2.5 Mapping requirements

In this section we show that the requirements ensure the satisfaction of the goals in the context of the domain assumptions: the list of requirements and domain assumptions under each goal have this purpose.

- [G1] The system must allow logged-in users to send a report of the violation
  - [D1] The number of possible violations is finite and is aligned to the current traffic rules
  - [D3] When using the S2B, the user's device is always connected to internet
  - [D4] When using the S2B, the user's device has a valid GPS signal
  - [D5] The internet connection works properly without failure
  - [D6] The user device has a camera and is able to take pictures
  - [D7] The user does not fake his position
  - [R1] A user must be able to sign up to the system with a unique personal username and password
  - [R2] The system must allow only registered users to login with their username and password
  - [R4] A user, municipal employee or local officer must be uniquely identified by his/her username
  - [R5] When composing the report, the system must be able to access the user's device camera and GPS sensor
  - [R6] When composing the report, a user must take a picture from the device's camera and highlight the license plate
  - [R7] When composing the report, time and date get automatically retrieved from the internet
  - [R8] When composing the report, position gets automatically retrieved from the device's GPS
  - [R9] When composing the report, a user can choose at least one type of violation
  - [R10] When composing the report, a user can't add the same violation type two times in the same report
  - [R11] When composing the report, a user can revert each phase of the creation of the report at any time, before sending it
  - [R12] When composing the report, a user can abort the creation of the report at any time, before sending it
  - [R13] Once a report has been sent, it can't be aborted or reverted
  - [R14] When receiving a report, the system must store it, recognize the car plate, if possible, and marked as unchecked

- [G2] The System must allow logged-in users to see their past reports
  - [D3] When using the S2B, the user's device is always connected to internet
  - [D5] The internet connection works properly without failure
  - [R1] A user must be able to sign up to the system with a unique personal username and password
  - [R2] The system must allow only registered users to login with their username and password
  - [R4] A users, municipal employee or local officer must be uniquely identified by his/her username
  - [R15] When a user asks for his/her reports, the system must provide the saved reports sent by that user
- [G3] The system must allow logged-in users to retrieve information about the position and types of valid reports
  - [D3] When using the S2B, the user's device is always connected to internet
  - [D4] When using the S2B, the user's device has a valid GPS signal
  - [D5] The internet connection works properly without failure
  - [D7] The user does not fake his position
  - [R1] A user must be able to sign up to the system with a unique personal username and password
  - [R2] The system must allow only registered users to login with their username and password
  - [R4] A users, municipal employee or local officer must be uniquely identified by his/her username
  - [R16] When getting the valid reports by area, a user can choose a position, or automatically get his/her from the GPS
  - [R17] When getting the valid reports by area, the system must provide all the valid reports near the position given by the user and display their violation type through the MS
- [G4] The system must allow verified authorities to mine information about date, time, position and type of valid reports
  - [D5] The internet connection works properly without failure
  - [D8] Every location has one and only one municipality
  - [D10] Each authority has its own account, certified and authorized by a state authority, and linked with the municipality which he/she works for
  - [D11] The official credentials of each municipal employee are different from those used for report violations (if he/she has registered as a normal user)

- [D12] The official credentials of each local officer are different from those used for report violations (if he/she has registered as a normal user)
  - [D13] The state authority voids credentials of municipal employees or local officers at the end of their service
  - [D14] When using the S2B, the authority's device is always connected to internet
  - [R3] The system must allow only municipal authorities to login with their username and password
  - [R4] A users, municipal employee or local officer must be uniquely identified by his/her username
  - [R18] When mining the information, a municipal employee or a local officer can access only to violations type and date and time of reports occurred in his/her municipality
  - [R19] When mining the information, a municipal employee or a local officer can filter reports by area, date, time or type of violation
- [G5] The system must allow verified authorities to retrieve statistics about valid reports
  - [D8] Every location has one and only one municipality
  - [D10] Each authority has its own account, certified and authorized by a state authority, and linked with the municipality which he/she works for
  - [D11] The official credentials of each municipal employee are different from those used for report violations (if he/she has registered as a normal user)
  - [D12] The official credentials of each local officer are different from those used for report violations (if he/she has registered as a normal user)
  - [D13] The state authority voids credentials of municipal employees or local officers at the end of their service
  - [D14] When using the S2B, the authority's device is always connected to internet
  - [R3] The system must allow only municipal authorities to login with their username and password
  - [R4] A users, municipal employee or local officer must be uniquely identified by his/her username
  - [R21] When retrieving statistics, a municipal employee or a local officer can access only to reports of violations that occurred in his/her municipality
  - [R22] The system must be able to calculate statistics from the reports of violations and issued tickets of the municipal employee or local officer's municipality

- [G6] The system must be able to cross the data retrieved from the municipality with its own, in order to identify unsafe areas and suggest possible interventions to the municipal employee
  - [D5] The internet connection works properly without failure
  - [D8] Every location has one and only one municipality
  - [D10] Each authority has its own account, certified and authorized by a state authority, and linked with the municipality which he/she works for
  - [D11] The official credentials of each municipal employee are different from those used for report violations (if he/she has registered as a normal user)
  - [D13] The state authority voids credentials of municipal employees or local officers at the end of their service
  - [D14] When using the S2B, the authority's device is always connected to internet
  - [D2] The number of possible interventions is finite and there exists an already established correlation between violations and possible interventions
  - [R3] The system must allow only municipal authorities to login with their username and password
  - [R4] A user, municipal employee or local officer must be uniquely identified by his/her username
  - [R23] When getting improvements, a municipal employee can access only to data of reports occurred in his/her municipality
  - [R24] The system must be able to retrieve information about accidents from the MAS (municipal accident system)
  - [R25] The system must be able to identify the possible unsafe areas of the municipal employee or local officer's municipality from the reports of violations and data coming the MAS
- [G7] The system must allow local officer to set the validity of a report sent by the user
  - [D5] The internet connection works properly without failure
  - [D8] Every location has one and only one municipality
  - [D10] Each authority has its own account, certified and authorized by a state authority, and linked with the municipality which he/she works for
  - [D12] The official credentials of each local officer are different from those used for report violations (if he/she has registered as a normal user)
  - [D13] The state authority voids credentials of municipal employees or local officers at the end of their service
  - [D14] When using the S2B, the authority's device is always connected to internet

- [R3] The system must allow only municipal authorities to login with their username and password
- [R4] A user, municipal employee or local officer must be uniquely identified by his/her username
- [R20] A local officer must be able to mark a report as valid or not valid
- [G8] The system must ensure that the chain of custody of the information coming from the user to the municipality is never broken, and the information is never altered
  - [D5] The internet connection works properly without failure

### 3.2.6 Traceability matrix

The following table keeps track of the relation between Use Cases and Requirements

Use cases	Requirements
Sign-Up	[R1]
Registered User login	[R2], [R4]
Add Report	[R5], [R6], [R7], [R8], [R9], [R10], [R11], [R12], [R13], [R14]
Get My Reports	[R15]
Get Violations Type By Area	[R16], [R17]
Authority login	[R3], [R4]
Get Improvements	[R23], [R24], [R25]
See Statistics	[R21], [R22]
Mine Reports By Type	[R18], [R19]
Mine Reports By Date	[R18], [R19]
Mine Reports By Time	[R18], [R19]
Mine Reports By Area	[R18], [R19]
Validate Reports	[R20]

Table 3.15: Traceability matrix

## 3.3 Performance requirements

The system does not have any particular performance requirements. Obviously, it will have to be able to handle multiple operations from multiple clients (users and authorities) at the same time.

## 3.4 Design constraints

### 3.4.1 Standard compliance

The system adopts the current traffic rules in order to provide all the possible VTs.

### **3.4.2 Hardware limitations**

The system presents hardware requirements only on the user's side. As a matter of fact, the user is required to have a smartphone with a camera and internet connection (wifi or mobile). Authorities need at least a device capable of connecting to the net.

## **3.5 Software systems attributes**

### **3.5.1 Reliability**

In order to provide reliability, the system must be resilient to faults. The solution is to replicate the system's server. In particular, it will be fault-tolerant against Byzantine faults (faults where a disconnected system has unforeseeable behaviors), so the number of replicas of the system's server must be, at least,  $(3 * \text{number of failing replicas}) + 1$ , with the number of failing replicas decided during the design and implementation.

### **3.5.2 Availability**

In order to provide availability, as mentioned in the reliability section, the system's server must be replicated. In this way, it is possible to obtain a 24/7 service. Obviously, little deviations from these requirements will be accepted.

### **3.5.3 Security**

In order to provide security, both end users and authorities' data will be always transferred through encrypted channels. The encryption will be achieved through the usage of public and private keys, each message will be encoded first with the private key of the sender, secondly the public key of the receiver will be used. Moreover every report, sent by the user, is provided with a digital signature, which will make the server able to realize if unauthorized data manipulation has been performed during the data transfer. Thanks to these techniques the purpose of maintaining the chain of custody from the user up to the authorities will be fulfilled.

### **3.5.4 Maintainability**

In order to provide maintainability, the development of the system has to be done so that will be easy and cheap to fix and modify it in the future. In order to achieve these properties, appropriate design patterns will be used. More of this in the design document.

### **3.5.5 Portability**

In order to provide portability, the system will be available as a downloadable app for the user and as a web service for the authorities.

## 4. Formal analysis using Alloy

In this section, the system is described using an Alloy model. The model contains all entities involved in the system and their attributes, following the schema given by the class diagram. Some relations between entities are highlighted and expressed as static constraints:

- It cannot happen that two different registered entities have the same user-name or that an email address is associated with more than one account
- Each position must belong to one and only one municipality and if a position is the center of a municipality it must belong to that municipality
- It cannot happen that a user, a vehicle, a ticket or an accident are registered in different positions at the same time
- If a valid report or an accident occurred on a position, there must be at least one improvement suggestion for that position, but there must not be suggestion without having problems on a position
- If the plate of the vehicle in a report is not recognized by the OCRS, the report must be set as "Not valid", but there can be reports set as "Not valid" with a valid license plate recognized
- Each municipality can access only to reports occurred on its territory
- Each RU can access only to his/her reports

Then, some other necessary structural constraints are expressed and properly commented in the Alloy formal notation. These are verified analyzing the worlds generated running the model. It is also highlighted how the addition of a report involves the municipality, the user, the vehicle and the position.

## 4.1 World model

```

----- SIGNATURE -----
2   sig Position {
4       latitude: one Int,
5       longitude: one Int,
6       municipality: one Municipality
7   }
8
9   sig Municipality {
10      name: one String,
11      center: one Position,
12      reports: set Report,
13      accidents: set Accident,
14      tickets: set Ticket
15   }
16
17   sig Picture {}
18
19   -- It represents each violation type that can be associated to a report
20   abstract sig ViolationType {}

22   -- It represents the set of violations associated to a report
23   -- In each report there must be at least one type of violation
24   sig Violation {
25      violations: some ViolationType
26   }

28   sig Date {}

30   sig Time {}

32   sig Vehicle {
33      licensePlate: one String
34   }

36   abstract sig RegisteredEntity {
37      username: one String,
38      password: one String
39   }
40
41   abstract sig Authority extends RegisteredEntity {
42      municipality: one Municipality
43   }
44
45   sig User extends RegisteredEntity {
46   {
47      email: one String,
48      reports: set Report
49   }
50
51   sig LocalOfficer extends Authority {}

52
53   sig MunicipalEmployee extends Authority {}

56   abstract sig ReportStatus {}
57   one sig Valid extends ReportStatus {}
58   one sig NotValid extends ReportStatus {}
59   one sig NotVerified extends ReportStatus {}

60
61   sig Report {
62      vehicle: lone Vehicle,
63      segnalatingUser: one User,
64      position: one Position,
65      picture: one Picture,
66      violation: one Violation,
67      date: one Date,
68      time: one Time,
69      status: one ReportStatus
70   }

```

```

72   abstract sig ImprovementState {}
73   one sig NotDone extends ImprovementState {}
74   one sig Done extends ImprovementState {}

76   abstract sig ImprovementType {}

78   sig Improvement {
        type: one ImprovementType,
80        position: one Position,
81        state: one ImprovementState
82    }

84   sig Accident {
        position: one Position,
85        date: one Date,
86        time: one Time,
87        vehicles: some Vehicle
88    }

89   abstract sig Gender {}

90   one sig Male extends Gender {}
91   one sig Female extends Gender {}

92   sig Ticket {
        vehicle: one Vehicle,
93        violation: one Violation,
94        position: one Position,
95        date: one Date,
96        time: one Time,
97        offenderGender: one Gender,
98        offenderAge: one Int,
99        report: lone Report
100    }

101   {
        -- Offender must be adult (age >= 18), but here used scaled value for
        -- simplicity
102   offenderAge ≥ 3
        -- if a ticket is related to a report, this must be valid
103   report.status = Valid
104 }

105   {
        -- Offender must be adult (age >= 18), but here used scaled value for
        -- simplicity
106   offenderAge ≥ 3
        -- if a ticket is related to a report, this must be valid
107   report.status = Valid
108 }

109   -----
110   ----- FACT -----
111   ----

112   -- Each username is unique
113   fact UniqueUsername {
114     no disj u1,u2: RegisteredEntity | u1.username = u2.username
115   }

116   -- Each vehicle is unique and is associated to a unique license plate
117   fact UniqueVehiclePlate {
118     no disj v1,v2: Vehicle | v1.licensePlate = v2.licensePlate
119   }

120   -- Each municipality name is unique
121   fact UniqueMunicipalityName {
122     no disj m1, m2: Municipality | m1.name = m2.name
123   }

124   -- Email used for registration must not be associated to multiple users
125   fact UniqueEmail {
126     no disj u1, u2: User | u1.email = u2.email
127   }

128   -- There must not be positions with same coordinates belonging to different
129   -- municipalities
130   fact SamePositionSameMunicipality {
131     no disj p1, p2: Position |
132       p1.latitude = p2.latitude ∧
133       p1.longitude = p2.longitude
134 }

```

```

144                     p1.longitude = p2.longitude ∧
145                     p1.municipality ≠ p1.municipality
146     }

148     -- The position associated to a municipality center must belong to such
149     --   ↪ municipality
150     fact CenterInMunicipality
151     {
152         all m: Municipality      | m.center.municipality = m
153     }

154     -- Each picture must belong exactly to one report
155     fact AllPictureBelongToOneReport
156     {
157         all p: Picture | one r: Report | r.picture = p
158     }

160     -- A vehicle can not be reported simultaneously in different reports with
161     --   ↪ different positions,
162     -- but it can be reported in the same position, for example from different
163     --   ↪ users
164     fact NoVehicleUbiquityInReport
165     {
166         no disj r1, r2: Report |
167             r1.date = r2.date ∧
168             r1.time = r2.time ∧
169             r1.vehicle = r2.vehicle ∧
170             r1.position ≠ r2.position
171     }

172     -- A user can not make different reports simultaneously
173     fact OneReportPerTimeForUser
174     {
175         no disj r1, r2: Report |
176             r1.segnalatingUser = r2.segnalatingUser ∧
177             r1.date = r2.date ∧
178             r1.time = r2.time
179     }

180     -- A vehicle can not be involved in an accident and simultaneously be
181     --   ↪ reported in a different position
182     fact NoVehicleUbiquityBetweenReportAccident
183     {
184         no r: Report, a: Accident |
185             r.vehicle in a.vehicles ∧
186             r.date = a.date ∧
187             r.time = a.time ∧
188             r.position ≠ a.position
189     }

190     -- A vehicle can be involved in at most one accident per time
191     fact NoVehicleUbiquityInAccident
192     {
193         no disj a1, a2: Accident |
194             a1.date = a2.date ∧
195             a1.time = a2.time ∧
196             a1.position ≠ a2.position ∧
197             a1.vehicles & a2.vehicles ≠ none
198     }

200     -- There must not be multiple accidents involving the same vehicles (or a
201     --   ↪ part of them)
202     -- in the same position and at the same time and date
203     fact UniqueAccident
204     {
205         no disj a1, a2: Accident |
206             a1.date = a2.date ∧
207             a1.time = a2.time ∧
208             --   ↪ a1.position = a2.position &&
209             (a1.vehicles = a2.vehicles ∨ a1.vehicles & a2.vehicles ≠ none
210             --   ↪ )
211     }

212     -- There must not be duplicated suggestion for improvements on same position

```

```

212  fact UniqueImprovementForEachPosition
213  {
214      no disj i1, i2: Improvement |
215          i1.position = i2.position ∧
216          i1.type = i2.type
217  }
218
219  -- A suggested improvement for a position must be generated only if there
220  -- occurred an accident or a valid report
221  fact NoImprovementWithoutProblem
222  {
223      no i: Improvement |
224          ((no a: Accident | i.position = a.position) ∧
225           (no r: Report | r.position = i.position ∧ r.status = Valid))
226
227  -- If on one position there occurred at least a minimum number of accidents
228  -- ↪ or valid reports,
229  -- there must be a suggested improvement for that position
230  -- For simplicity, the minimum number of "problems" occurred on a position in
231  -- ↪ order to have a suggested improvement is set to one
232  fact NoProblematicPositionWithoutImprovement
233  {
234      no p: Position |
235          (((some a: Accident | a.position = p) ∨
236           (some r: Report | r.position = p ∧ r.status = Valid)) ∧
237           no i: Improvement | i.position = p)
238
239  -- If the OCRs did not recognize a valid license plate, the report is not
240  -- ↪ valid but is still stored
241  -- Some "Not Valid" reports can be associated to a vehicle: it might happen
242  -- ↪ that a local officer
243  -- marked them as "Not Valid" but the vehicle's license plate had been
244  -- ↪ recognized by the OCRs
245  fact NotValidReportsStored
246  {
247      all r: Report |
248          r.vehicle = none implies r.status = NotValid
249
250  -- If the OCRs recognized a valid license plate, the related vehicle must be
251  -- ↪ associated to the report
252  fact ValidReportsRelatedToAVehicle
253  {
254      all r: Report |
255          r.status ≠ NotValid implies r.vehicle ≠ none
256
257  -- All strings must be associated to a username, to a municipality name or to
258  -- ↪ a license plate
259  fact NoRandomString
260  {
261      all s: String |
262          ((one e: RegisteredEntity | e.username = s) ∨
263           (one m: Municipality | m.name = s) ∨
264           (one v: Vehicle | v.licensePlate = s))
265
266  -- If a ticket is related to a valid report, they must be consistent
267  fact TicketReportConsistency
268  {
269      all t: Ticket |
270          (t.report ≠ none implies
271              (t.position = t.report.position ∧
272               t.vehicle = t.report.vehicle ∧
273               t.date = t.report.date ∧
274               t.time = t.report.time ∧
275               t.violation.violations & t.report.violation.
276               ↪ violations ≠ none))
277
278  -- There must be a unique issued ticket for a vehicle on a specific date and
279  -- ↪ time

```

```

276  fact UniqueTicket
277  {
278      no disj t1, t2: Ticket |
279          t1.vehicle = t2.vehicle ∧
280          t1.date = t2.date ∧
281          t1.time = t2.time
282  }
283
284  -- A vehicle can not be issued in a ticket and simultaneously be reported in
285  -- ↪ a different position
286  fact NoVehicleUbiquityBetweenReportTicket
287  {
288      no r: Report, t: Ticket |
289          r.vehicle in t.vehicle ∧
290          r.date = t.date ∧
291          r.time = t.time ∧
292          r.position ≠ t.position
293  }
294
295  -- A vehicle can not be involved in an accident and simultaneously be issued
296  -- ↪ in a different position
297  fact NoVehicleUbiquityBetweenTicketAccident
298  {
299      no t: Ticket, a: Accident |
300          t.vehicle in a.vehicles ∧
301          t.date = a.date ∧
302          t.time = a.time ∧
303          t.position ≠ a.position
304  }
305
306  -- Each municipality must have only reports occurred in a position belonging
307  -- ↪ to that municipality
308  fact MunicipalityOwnsOnlyItsReports
309  {
310      all m: Municipality |
311          (m.reports ≠ none implies
312              (all r: m.reports | r.position.municipality = m))
313  }
314
315  -- All reports must be associated to a municipality
316  fact AllReportsRelatedToMunicipality
317  {
318      all r: Report |
319          (one m: Municipality | r in m.reports)
320  }
321
322  -- Each user must have only his/her reports
323  fact UserOwnsOnlyItsReports
324  {
325      all u: User |
326          (u.reports ≠ none implies
327              (all r: u.reports | r.segnalatingUser = u))
328  }
329
330  -- All reports must be associated to a the segnalating user
331  fact AllReportsRelatedToUser
332  {
333      all r: Report |
334          (one u: User | r in u.reports)
335  }
336
337  -- Each municipality must have only accidents occurred in a position
338  -- ↪ belonging to that municipality
339  fact MunicipalityOwnsOnlyItsAccidents
340  {
341      all m: Municipality |
342          (m.accidents ≠ none implies
343              (all a: m.accidents | a.position.municipality = m))
344  }
345
346  -- All accidents must be associated to a municipality
347  fact AllAccidentsRelatedToMunicipality
348  {
349      all a: Accident |

```

```

346             (one m: Municipality | a in m.accidents)
348         }
349         -- Each municipality must have only accidents occurred in a position
350         --   ↪ belonging to that municipality
350     fact MunicipalityOwnsOnlyItsTickets
351     {
352         all m: Municipality |
353             (m.tickets ≠ none implies
354                 (all t: m.tickets | t.position.municipality = m))
355     }
356     -- All tickets must be associated to a municipality
357     fact AllAccidentsRelatedToMunicipality
358     {
359         all t: Ticket |
360             (one m: Municipality | t in m.tickets)
361     }
362     ----- ASSERTION -----
363     -- There are no municipalities with centers in the same position
364     assert NoOverlapMunicipalityCenter
365     {
366         no disj m1, m2: Municipality | m1.center = m2.center
367     }
368     check NoOverlapMunicipalityCenter
369     -- There are no reports reported by authority account
370     assert NoAuthorityReport
371     {
372         no r: Report, a: Authority | r.segnalatingUser.username = a.username
373     }
374     check NoAuthorityReport
375     ----- PREDICATE -----
376     pred addReport [m, m': Municipality, u, u': User, r: Report]
377     {
378         //precondition
379         r.segnalatingUser = u
380         r.position.municipality = m
381         //postcondition
382         u'.reports = u.reports + r
383         m'.reports = m.reports + r
384     }
385     -- Generic world
386     pred GenericWorld
387     {}
388
389     -- World to highlight municipality
390     pred MunicipalWorld
391     {
392         #Municipality = 3
393     }
394
395     -- World to highlight reports
396     pred UserWorld
397     {
398         #User = 4
399         #Report = 5
400         #Position = 3
401         one r: Report | r.status = Valid
402         one r: Report | r.status = NotValid
403     }
404
405     run GenericWorld for 3 but exactly 3 String

```

```
420  run MunicipalWorld for 5 but exactly 9 String, 3 Municipality
      run UserWorld for 6 but exactly 10 String, 0 Accident
      run addReport for 3 but exactly 5 String
```

```
Executing "Check NoAuthorityReport"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
6469 vars. 462 primary vars. 11594 clauses. 219ms.
No counterexample found. Assertion may be valid. 16ms.
```

```
Executing "Check NoOverlapMunicipalityCenter"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
6514 vars. 462 primary vars. 11738 clauses. 85ms.
No counterexample found. Assertion may be valid. 0ms.
```

Figure 4.1: Alloy assert execution

```
Executing "Run GenericWorld for 3 but exactly 3 String"
Solver=sat4j Bitwidth=4 MaxSeq=3 SkolemDepth=1 Symmetry=20
6939 vars. 501 primary vars. 12531 clauses. 69ms.
Instance found. Predicate is consistent. 100ms.
```

```
Executing "Run MunicipalWorld for 5 but exactly 9 string, 3 Municipality"
Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
19921 vars. 1175 primary vars. 37718 clauses. 285ms.
Instance found. Predicate is consistent. 285ms.
```

```
Executing "Run UserWorld for 6 but exactly 10 string, 0 Accident"
Solver=sat4j Bitwidth=4 MaxSeq=6 SkolemDepth=1 Symmetry=20
23752 vars. 1512 primary vars. 47471 clauses. 116ms.
Instance found. Predicate is consistent. 323ms.
```

```
Executing "Run addReport for 3 but exactly 5 string"
Solver=sat4j Bitwidth=4 MaxSeq=3 SkolemDepth=1 Symmetry=20
7340 vars. 546 primary vars. 13502 clauses. 47ms.
Instance found. Predicate is consistent. 46ms.
```

Figure 4.2: Alloy worlds execution

## 4.2 Generated world

### 4.2.1 Generic world

This is an example of a generic world, with entities and their relations.

Each position belongs to only one municipality (which center position belongs to it too) and they don't share reports.

A user can send reports to different municipalities and can access only to all his/her past reports.

Authorities are not related to reports.

Improvements are generated if and only if a problem occurred in a certain position.

There are no ubiquity problems.

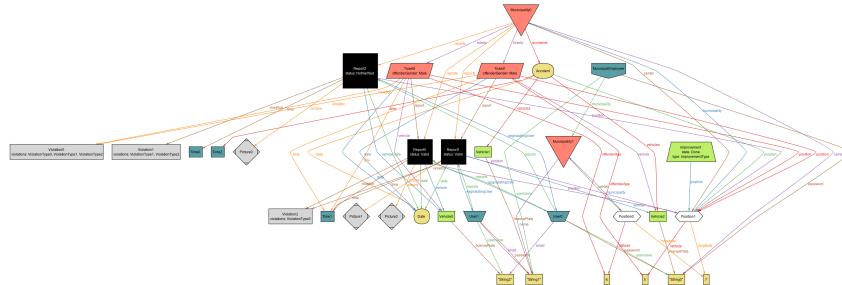


Figure 4.3: Alloy generic world

#### 4.2.1.1 Municipal world

This world highlights the associations occurring into a municipality, such as received reports, issued tickets, occurred accidents and suggested improvements. It also shows how positions are related to municipalities and what occurred on them.

Each municipality can access only to reports that occurred on its territory. It can also get improvements only in positions belonging to its territory.

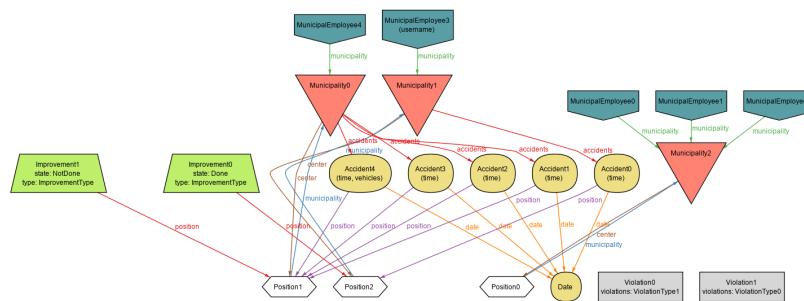


Figure 4.4: Alloy municipal world

#### 4.2.1.2 User world

This world highlights the associations between users and reports.

Each user can access only to his/her past reports, whatever the municipality on which each one occurred.

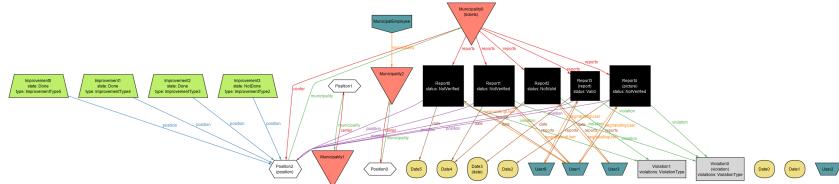


Figure 4.5: Alloy user world

#### 4.2.1.3 Add report

It shows how an "add report" event occurred, being related to user and municipality, and also to the vehicle and the position, eventually making possible the generation of a possible improvement if the report is valid.

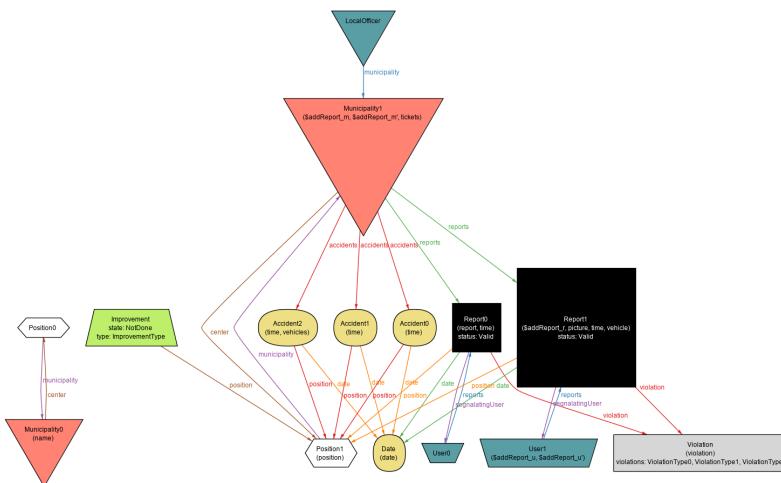


Figure 4.6: Alloy add report

## 5. Effort spent

<b>Student</b>	<b>Time spent</b>	<b>Total (h)</b>
Bonatti Andrea	Part 1: 12 h Part 2: 15 h Part 3: 5 h Part 4: 3 h	35
Buttironi Monica	Part 1: 3 h Part 2: 5 h Part 3: 13 h Part 4: 17 h	38
Caruso Marco Giuseppe	Part 1: 5 h Part 2: 7 h Part 3: 20 h Part 4: 3 h	35
<b>Total</b>	Part 1: 20 h Part 2: 27 h Part 3: 38 h Part 4: 23 h	108

Table 5.1: Effort spent

## 6. References

- L<sup>A</sup>T<sub>E</sub>X for document typesetting
- Github (<https://github.com>) for version control
- Alloy (<http://alloy.mit.edu/alloy/>) for model verification
- Signavio (<http://www.signavio.com/>) for use case diagrams, class diagrams and state diagrams
- Moqups (<https://moqups.com/>) for mockups
- UML Sequence Diagram Online Tool (<https://sequencediagram.org>) for sequence diagrams