

Software Engineering 2

SafeStreets

Requirements Analysis and Specification Document

Fortina Valeria Maria
Galluccio Alessio

Sunday, November 10, 2019



Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 4 |
| 1.1 | Purpose | 4 |
| 1.1.1 | Goals | 5 |
| 1.2 | Scope | 6 |
| 1.2.1 | World Phenomena | 6 |
| 1.2.2 | Machine Phenomena | 6 |
| 1.2.3 | Shared Phenomena | 6 |
| 1.3 | Definitions, Acronyms, Abbreviations | 7 |
| 1.3.1 | Definitions | 7 |
| 1.3.2 | Acronyms | 8 |
| 1.3.3 | Abbreviations | 8 |
| 1.4 | Revision History | 8 |
| 1.5 | Reference Documents | 9 |
| 1.6 | Document Structure | 9 |
| 2 | Overall description | 11 |
| 2.1 | Product perspective | 11 |
| 2.1.1 | Class Diagram | 14 |
| 2.1.2 | State Diagram of a Report by the end user | 15 |
| 2.1.3 | State Diagram of a Report handled by an Authority user | 16 |
| 2.2 | Product functions | 17 |
| 2.3 | User characteristics | 18 |
| 2.3.1 | End Users | 18 |
| 2.3.2 | Authorities | 18 |
| 2.3.3 | System Managers | 18 |
| 2.4 | Assumptions, dependencies and constraints | 18 |
| 2.4.1 | Text assumptions | 19 |
| 2.4.2 | Domain assumption | 19 |
| 2.4.3 | Constraints | 19 |
| 3 | Specific Requirements | 20 |
| 3.1 | External Interface Requirements | 20 |
| 3.1.1 | User Interfaces | 20 |
| 3.1.2 | Hardware Interfaces | 24 |
| 3.1.3 | Software Interfaces | 24 |
| 3.1.4 | Communication Interfaces | 24 |
| 3.2 | Functional Requirements | 25 |
| 3.2.1 | Use Cases | 26 |
| 3.2.2 | Mapping on functional requirements | 43 |
| 3.2.3 | Traceability Matrix | 46 |
| 3.3 | Performance Requirements | 49 |
| 3.4 | Design Constraints | 49 |
| 3.4.1 | Standards Compliance | 49 |
| 3.4.2 | Hardware Limitations | 49 |

| | | |
|----------|--|-----------|
| 3.5 | Software System Attributes | 49 |
| 3.5.1 | Reliability | 49 |
| 3.5.2 | Availability | 50 |
| 3.5.3 | Security | 50 |
| 3.5.4 | Maintainability | 50 |
| 3.5.5 | Portability | 50 |
| 4 | Formal Analysis using alloy | 51 |
| 4.1 | Signatures | 51 |
| 4.2 | Facts | 52 |
| 4.3 | Predicates | 54 |
| 4.3.1 | No Ticket has no Offender | 54 |
| 4.3.2 | Ticket not already generated | 55 |
| 4.3.3 | No Report if no Vehicle in Photo | 56 |
| 4.4 | Proof of consistency | 57 |
| 5 | Effort Spent | 59 |
| 6 | References | 60 |

1 Introduction

The following Requirements Analysis and Specification Document (RASD) aims at providing a baseline for project planning and development with complete description of all functional and non-functional requirements of the system. In addition, other goals of this document are:

- analyzing the real needs of the target customers for modeling the system;
- providing a detailed description of the constraints and limits of the software itself and indicate the typical use cases that will occur after the release.

This document is addressed to different subjects and stakeholders:

- customers and users (chapter 1);
- developers (chapter 2,3);
- testers (chapter 3,4);
- project managers (chapter 1,2,3,4).

1.1 Purpose

SafeStreets is a new crowd-sourced application whose goal is to ensure better road safety and improve compliance with the traffic rules, especially those related to parking. To achieve its goals, SafeStreets is aimed at different types of users, first all ordinary citizens. In fact, ordinary citizens can access SafeStreets services via the mobile application and, once registered, make a significant contribution. The application makes it possible to report possible violations by taking a photo and sending it along with general information such as the type of violation, the location, the date of the event and the license plates of the vehicles. These reports are taken over by the second type of user, those who can be defined as an authority, in particular, police officers. Once the report is received, the authority can assess its validity and if it deems necessary to directly generate a ticket to those who have committed the violation. Finally, to keep the system efficient over time, SafeStreets allows its employees, system managers, to make changes to the system, such as adding or deleting streets or types of violations, based on changes in the law or urban planning. To improve the user experience SafeStreets has also introduced additional features depending on the user, for example searching for a registered vehicle license plate, or violations committed on a specific street or even checking SafeStreets' efficiency data.

1.1.1 Goals

- G.1 Allow each End User and Authority to be properly identified by the system;
- G.2 Allow End User to send pictures of the report , including their date, time and position;
- G.3 Allow End User to select report's type from a given list;
- G.4 Allow End User to send the license plate of the vehicle;
- G.5 Allow End User and Authority to see streets' number and types of violations;
- G.6 Allow Authority to see all the vehicles' information which have already committed a violation or search them by license plate;
- G.7 Allow Authority to generate traffic tickets for violations;
- G.8 Allow Authority and System Managers to see the efficiency of SafeStreets
 - G.8.1 Allow Authority and System Managers to see the number of the tickets on the total reports globally;
 - G.8.2 Allow Authority and System Managers to see the trend of tickets over time globally;
 - G.8.3 Allow Authority and System Managers to see the number of the tickets on the total reports per street;
 - G.8.4 Allow Authority and System Managers to see the trend of tickets over time per street;
- G.9 Allow a System Manager to do operations on the system for updating and maintenance;
 - G.9.1 Allow a System Manager to add a violation type to the system;
 - G.9.2 Allow a System Manager to add a street to the system;
 - G.9.3 Allow a System Manager to delete a violation type to the system;
 - G.9.4 Allow a System Manager to delete a street to the system;
- G.10 Allow a System Manager to do user management operations;
 - G.10.1 Allow a System Manager to add a new Authority;
 - G.10.2 Allow a System Manager to add a new System Manager;
 - G.10.3 Allow a System Manager to delete an Authority;
 - G.10.4 Allow a System Manager to delete a System Manager;

1.2 Scope

According to the World and Machine paradigm, we can distinguish the Machine, which is System to be developed, from the World, which is a portion of the real-world affected by the machine. This separation leads to a classification of the phenomena in three different types, depending on where they occur. In this context, the main relevant phenomena are organized as follows:

1.2.1 World Phenomena

Person commits a possible violation with a vehicle
User witnesses a possible violation
Traffic laws are modified
The urban planning of a city is modified
An authorized agent issues a fine without using SafeStreets

1.2.2 Machine Phenomena

Algorithm to detect the license plate of a vehicle in the highlighted part of an image
All operations performed to retrieve, store and update data
Synchronize data with the government database
System clock: the System runs its tasks according to its clock.
Algorithm to detect possible alteration of reports' data

1.2.3 Shared Phenomena

- **Controlled by the world and observed by the machine**

Updating of data in the government database
Registration/Login: a guest can sign up to the application or otherwise log in if he/she is already registered.
Collection of user's data by means of sensors (GPS position, date, photos)
Creation or cancellation of reports by users
Creation or cancellation of violation types by users
Authority User decides that a report is a violation
Addition or removal of a street by the System Manager
Addition or removal of a type of violations by the System Manager

Addition or removal of a System Manager by the System Manager

Addition or removal of an Authority by the System Manager

- **Controlled by the machine and observed by the world**

Generation of tickets

Updating of statistics and data

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- End User: a common citizen registered to the system.
- Authority (or Authority User): a registered police officer.
- Report: a possible transgression of the traffic code.
- Violation: a verified report, which can be punished with a ticket.
- Ticket: sanction that an authority decides has to be paid as punishment for a violation of the traffic code.
- Owner: owner of one or more vehicles.
- Offender: owner of a vehicle for which the violation has been recognized and a ticket can be issued to.
- App: the part of the system with which the user interacts.
- Violations' data: reported violations, with their date, time, position, picture and type.
- User's data: user personal information and login information.
- Registered vehicles: vehicles that have already committed a violation.
- Efficiency: SafeStreets' efficiency, is calculated on the number of tickets issued before and after SafeStreets' arrival and on the the number of tickets on the total reports.
- Global Efficiency: SafeStreets' efficiency calculated on the whole territory covered by the system.
- Street Efficiency: SafeStreets' efficiency calculated in a specific street.
- Statistics: union of Violations' data, Global Efficiency and Street efficiencies.
- Special User: an Authority user or a System Manager.

1.3.2 Acronyms

- DB: Database
- GPS: Global Positioning System
- RASD: Requirement Analysis and Specification Document
- UI: User Interface
- HTTPS : Hyper Text Transfer Protocol Secure
- HTTP : Hyper Text Transfer Protocol

1.3.3 Abbreviations

- G.n : n-th goal
- D.n: n-th domain assumption
- R.n : n-th functional requirement

1.4 Revision History

- An update to the sequence diagrams in which all SafeStreets' components have been eliminated.
- A change in requirements 24 and 25 to make the meaning more understandable.
- A change in requirements 3 to make the meaning more understandable.
- A new goal added: number 10 and 10.1, 10.2, 10.3, 10.4. In this way Authority and System Manager addition to the system is more clear.
- New shared phenomena "Addition or removal of a System Manager/Authority by the System Manager"
- New phenomena in Product perspective for adding/removal of System Manager and Authority
- New Functional Requirements in 2.2 Product functions for adding/removal of System Manager and Authority by System Manager.
- New line on 2.3 User characteristics in System Manager specifying adding/removal of System Manager and Authority by System Manager.
- Goal 10 added in mapping of functional requirements, with new requirements 26,27,28,29.
- New definition "Special User" added.
- Goal 10 added to the Traceability Matrix.

- Updated State diagrams
- Added "Remove Special User" and "Add Special User" in the Use Case diagram and in the Use cases

1.5 Reference Documents

- IEEE Std 830-1993 - IEEE Guide to Software Requirements Specifications.
- IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications.

1.6 Document Structure

This RASD is organized in six different chapters following a classical RASD structure:

- 1 **Introduction:** The first chapter starts with a general introduction to the document using a textual description of the product along with the main goals of the project. To follow there is an analysis of the phenomena based on the Jackson and Zave's Machine and World approach. Next, a section has been devoted to useful definitions, acronyms, and abbreviations used throughout the whole document. The chapter ends with a review history of the document.
- 2 **Overall Description:** In the second chapter, the shared phenomena identified in the previous chapter are analyzed in depth. The most important functional and non functional requirements, domain assumptions and constraints are defined. With the use of class and state diagrams, the system-to-be is further detailed. Moreover, the types of users are identified and their needs and interests are specified.
- 3 **Specific Requirements:** The third chapter concerns a more in-depth analysis of the system-to-be. Initially, it focuses on the analysis of interfaces, such as user's (provided by means of mock-ups), software, hardware, and communication. Following it shows the basic use cases, some example scenarios, along with sequence diagrams, and a complete mapping on requirements shown also in the form of a traceability matrix. Then the functional requirements that must be satisfied to provide a good grade of usability are reported. The chapter ends with a detailed look at the main design constraints that will guide the choices of the next design phase that will be part of the Design Document (DD) and the main software system attributes (reliability, availability, security, maintainability, portability).
- 4 **Formal Analysis using alloy:** In this section, using the formal language Alloy, some important properties of the system are proved, and consistent worlds generated by the reasoner are provided. Finally, the report of the reasoner is showed.

5 **Effort Spent:** In this chapter the working hours of the team members who worked on this document are shown.

2 Overall description

2.1 Product perspective

SafeStreets is a crowd-sourced application that aims to provide to the end users the possibility to notify the authorities of traffic violations that occur using photos. It guarantees to the authorities that the information is not manipulated, in order to use the data to generate traffic tickets. Finally, it gives a set of statistics to the user. End user can only see the number, the type and the frequency of violations per street. Authorities can access to the same data of the end users, plus all the information of each violation, all the information of registered vehicles, the issued tickets, and efficiency of SafeStreets application. System Managers can add and eliminate types of violations and streets, in order to follow changes of the world like change of laws or urban planning. System Managers can also access to efficiency of SafeStreets application.

- **End User sign up process (world controlled, machine observed):** the End User can submit their credentials. If they are already registered, he/she log in. Otherwise, he/she registers in the system using his/her fiscal code as id and choosing a username and a password. The application helps the End User with an interface in submitting the data and checks it. The credentials are saved in the system.
- **End User log in process (world controlled, machine observed):** the End Users submits his/her credentials. If they are correct, he/she can log in.
- **Authority log in process (world controlled, machine observed):** the Authority User can submit their credentials. If they are already registered, they will log in. Otherwise, they will register using an authorized code as id. Authority users don't need a sign up, because they get an authorized code and a password from a System manager.
- **System manager log in process (world controlled, machine observed):** the System Manager can log in using an authorized code as id. System managers don't need a sign up, because they receive an authorized code and a password from another System Manager.
- **Report by an end user (world controlled, machine observed):** the End User takes a picture with the app on the smartphone of the vehicle. The End User inserts the type of violation, adds the vehicles and highlights the license plates. The application adds the location and the time of the report in the data using the GPS system and clock of the smartphone. All the data is saved in the system.
- **Handling of a Report by an Authority User (world controlled, machine observed):** the Authority User selects a report and decides if a violation has been committed or not. If it is committed, he/she selects

the license plates to which give the Traffic ticket. If it acknowledges that there were no violation, the report is discarded.

- **Request of seeing a statistics by a User (world controlled, machine observed):** the User requests to access to statistics he/she can access to. The system retrieves the information and shows it to the User through a UI.
- **Adding a type of violation (world controlled, machine observed):** a System Manager can add a new type of violation, or restore an old one. The system will add this option in the possible one to be selected in the report of the end user. If the type of violation is restored, the system will show the old data and the new one of that type of violation after the next update of statistics.
- **Removing a type of violation (world controlled, machine observed):** a System Manager can remove a type of violation. The system will not permit the end users to select it in the next reports. The system will not delete it in the data, but it will not show it in the statistics.
- **Adding a street (world controlled, machine observed):** a System Manager can add a new street, or restore an old one. The system will update the statistics. If the street is restored, the system will show the old data and the new one of that street in the statistics after the next update of statistics.
- **Removing a street (world controlled, machine observed):** a System Manager can remove a street. The system will not delete it in the data, but it will not show it in the statistics.
- **Adding a System Manager (world controlled, machine observed):** a System Manager can add a new System Manager. The system will save the new data, that will be used for future login.
- **Removing a System Manager (world controlled, machine observed):** a System Manager can remove another System Manager. The system will delete the data. The data will no longer be used for logins.
- **Adding an Authority (world controlled, machine observed):** a System Manager can add a new Authority. The system will save the new data, that will be used for future login.
- **Removing an Authority (world controlled, machine observed):** a System Manager can remove another Authority. The system will delete the data. The data will no longer be used for logins.
- **Generation of tickets (machine controlled, world observed):** if a report is confirmed to be a violation by an authority, the system will generate a traffic tickets accordingly to the offenders selected by the Authority user.

- **Automatic update of data (machine controlled, world observed):**
the system updates the statistics, the vehicles' data and the violations every day, including the violations and vehicles saved in the Government database.

2.1.1 Class Diagram

The class diagram represents the main entities that the system should have to reach its goals. The most relevant attributes and relations between entities are showed. For simplicity, only two types of violations are showed. The System Managers can add, restore and remove them, following the traffic laws as needed.

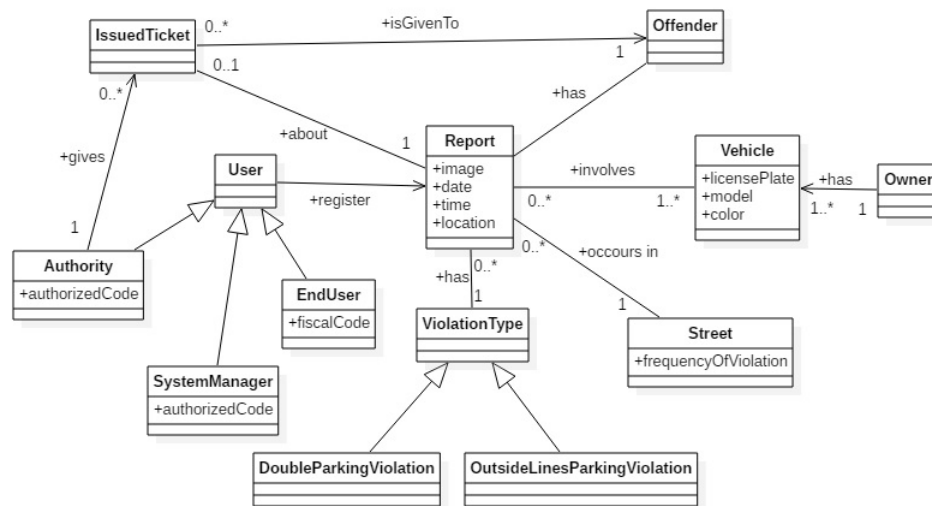


Figure 1: Class Diagram

2.1.2 State Diagram of a Report by the end user

This state diagram represents the states that the machine can reach when an End User wants to create a Report. For simplicity, we omitted the action "abort creation of the Report", but it can happen anywhere and the system will cancel this operation. The user must take a photo, add vehicles and select the license plates. The system saves automatically the time and the position, and it helps the user to compile all the data consistently. Then it handles the update of its data storage.

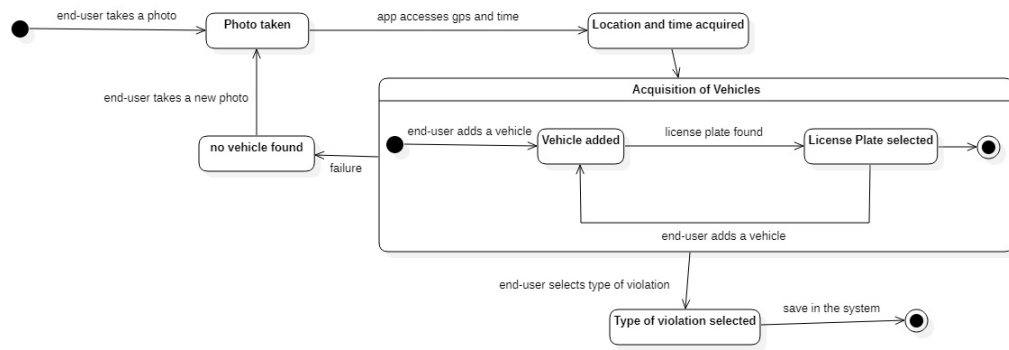


Figure 2: State Diagram: Report by End User

2.1.3 State Diagram of a Report handled by an Authority user

This state diagram represents the state that the machine reach when an Authority User selects a report. After the Authority User decides if the report is a violation or not, the system acts accordingly. If the report is confirmed to be a violation, the system generates a ticket for the offenders selected previously by the Authority User, otherwise it deletes the report.

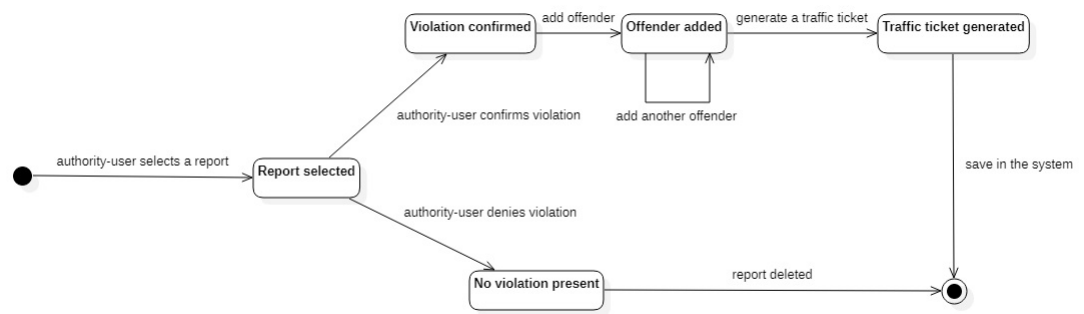


Figure 3: State Diagram: Handling of report by Authority

2.2 Product functions

Functional Requirements

- End users shall be able to:
 - Sign up
 - Login
 - Report a violation
 - See the number of violations per types happened in each street
- Authority users shall be able to:
 - Login
 - See all new reports (including all their data)
 - See all the violations per street
 - Delete reports which don't have a violation
 - Assign traffic tickets
 - See list of all registered vehicles and their data
 - Search a vehicle by license plate
 - See globally efficiency of SafeStreets
 - See street efficiency of SafeStreets
- System managers shall be able to:
 - Login
 - Add a new type of violation
 - Delete a type of violation
 - Add a new street
 - Delete a street
 - Add a System Manager
 - Add an Authority
 - Delete a System Manager
 - Delete an Authority
 - See globally efficiency of SafeStreets
 - See street efficiency of SafeStreets
- The system shall be able to:
 - Guarantee that the information of the reports is not altered
 - Collect data from the user's devices
 - Generate traffic tickets

Non-Functional Requirements

- The system must be available 24/7.
- Statistics must be updated at least once a day
- Data about vehicles must be updated at least once a day
- Data about streets must be updated at least once a day

2.3 User characteristics

There are three types of user, each one with their needs:

2.3.1 End Users

The end users need a user-friendly app for smartphones to take photos of violations and report them to the authorities. They also want to know how much a street is unsafe by seeing the number and type of traffic tickets generated per street. The number of traffic tickets should also include the ones that are not generated through SafeStreets, but in other ways.

2.3.2 Authorities

The Authorities need a browser application that can be used on desktops and laptops which permits them to see the reports and, if they judge that there was a violation, to generate a traffic ticket for each offender they selected from the application. Otherwise, the report must be discarded. They should be able to search for a vehicle by its license plate, and see the number of violations per vehicle. They need to access to the value of efficiency of SafeStreets, which is calculated as trend of tickets over time and number of tickets generated on number of reports, globally and per street. They need a system which can guarantee that the pictures of the reports are never altered.

2.3.3 System Managers

The system managers should be able to add, restore or eliminate types of violations, in case of change of law. They need the possibility to add, restore or eliminate streets in case of urban modification. They also need to have access to the values of the efficiency of SafeStreets, the same of Authority Users. Finally they need the possibility to add or remove other System Managers and Authority Users.

2.4 Assumptions, dependencies and constraints

Some points of the customer specification document have different interpretation. In order to eliminate ambiguities, some assumptions have been done.

2.4.1 Text assumptions

- Reports must be controlled and approved by an authority to be considered violations.
- Traffic tickets are generated by the system after an authority confirmed the violation.
- Traffic laws and streets can change over time. System Managers are then needed to add and delete types of violations and streets.

2.4.2 Domain assumption

- D.1 An end user is always supposed to sign up with his own fiscal code.
- D.2 An authority user or a system manager is always supposed to log in with his own authorized code.
- D.3 An end user always log in with a smartphone.
- D.4 The smartphone of the end user has at least one photo-camera.
- D.5 The smartphone of the end user has a GPS system.
- D.6 The GPS of the smartphone of the end user always returns a correct location with a maximum error of 1 meter.
- D.7 An authority user or a system manager always login with a desktop.
- D.8 A smartphone is different from a desktop.
- D.9 There are enough authority users assigned to the application to manage all the reports of the day.
- D.10 Two or more streets (areas) can't overlap.
- D.11 The internet connection works properly without failure.
- D.12 Every alteration of the data of the reports can be identified by the security software
- D.13 Every license plate is registered in the DB of the Government
- D.14 The smartphone of the end user has a clock

2.4.3 Constraints

- The system must be able to communicate with the DB of the Government, since it has to retrieve the vehicle data and the data of the violations that haven't been reported through SafeStreets.
- Part of system must be deployed on a mobile device of the End User that can take photos and send data.

3 Specific Requirements

3.1 External Interface Requirements

SafeStreets can be used by three types of users: common individuals as end users, from which the system collects reports' data, authorities, such as police officers, which use the given data at their needs, and system managers which make changes to the system. While end users can interface themselves with SafeStreets through a mobile application, authorities and system managers have to use a web application to access the system's functionalities. Here is shown every kind of interface that the front-end offers or needs to interact with the users and the back-end.

3.1.1 User Interfaces

In this section the following mock-ups represent a basic idea of the user interfaces for the mobile application used by end users and for the web-based software used by authorities and system managers.

Sign Up

The app requests the end user to complete the sign up process by entering an username, a fiscal code and a password. There is also the option to log in if already registered.

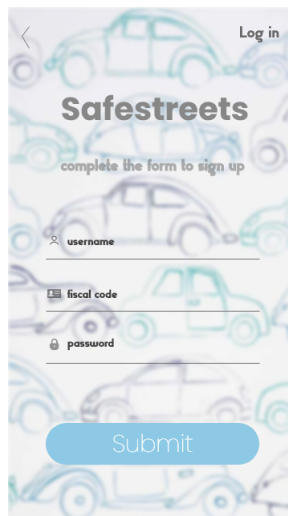
A mobile application mock-up for the 'Safestreets' sign-up screen. The background is light blue with a faint pattern of stylized cars. At the top left is a back arrow icon, and at the top right is a 'Log in' link. The app name 'Safestreets' is centered in a bold, dark font. Below it, the text 'complete the form to sign up' is displayed. There are three input fields: 'username' with a magnifying glass icon, 'fiscal code' with a document icon, and 'password' with a lock icon. A large blue 'Submit' button is at the bottom.

Figure 4: End user sign up

Log in

- End user: In the login page of the end user there are text fields for username and password in order to access the system. There is also the option to sign up if not already registered.

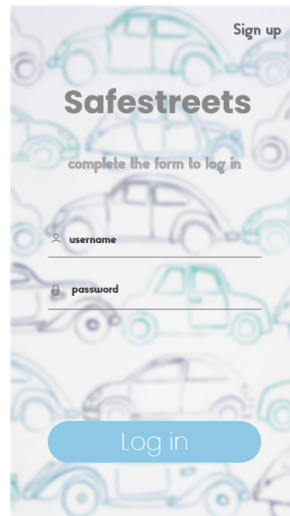


Figure 5: End user log in

- Authority and System Manager: in the login page there are text fields for the secret code and password in order to access the system.

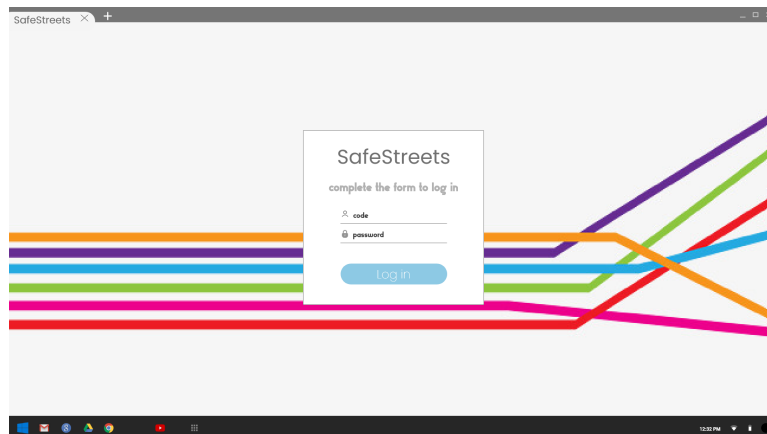


Figure 6: Authority and System Manager log in

Homepage

- End user: once registered, the end user will be welcomed by the homepage that show the option to create a new report. There is also the option to log out.

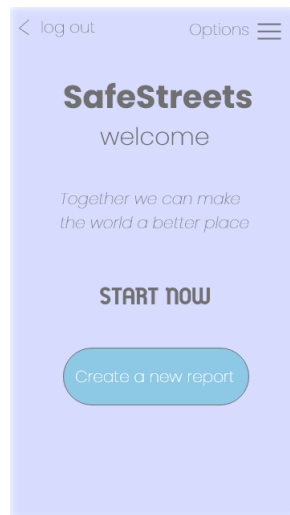


Figure 7: End user homepage

- Authority: after the access, the authority user will see all the new reports, with the possibility to generate tickets or disvehicled them. There is also the option to log out.

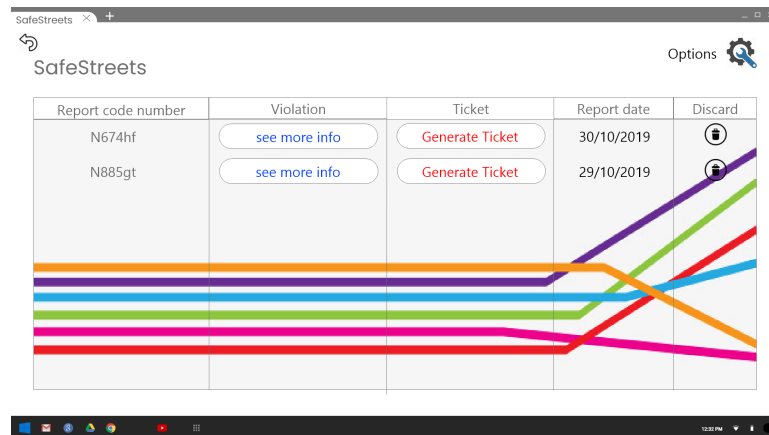


Figure 8: Authority homepage : tickets generator

Report creation

This is the area dedicated to the creation of a new report, the end user to generate a report must take a picture with the mobile phone, then add one or more vehicle. There is also the option to go back to the homepage.

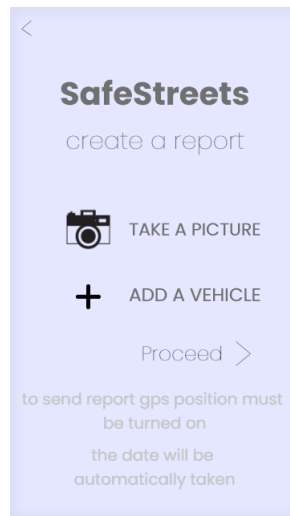


Figure 9: End user report creation

3.1.2 Hardware Interfaces

The main requirement for a end user is to have a mobile phone, since the since the application for end user is accessible only on this type of devices, another important requirement is to have an internet access to connect to the services and collect two type of data:

- **Position data**, which requires a GPS capable device.
- **Picture data**, which requires a camera on the device.
- **Time data**, which requires a clock on the device.

Regarding authorities and system managers, all they need is a desktop computer or a laptop with internet access.

3.1.3 Software Interfaces

The mobile app requires the Android or iOS as mobile operating system to run on a mobile phone.

Instead, the services provided for system managers and authority users require a web browser, for example Google Chrome, Internet Explorer or Mozilla Firefox.

The system's back-end requires a software to manage all the collected data stored in the system.

Our system must communicate with two external software interfaces: the Government database interface, to retrieve useful data, and the security software, to guarantee that the report's data is not altered.

3.1.4 Communication Interfaces

The mobile app requires the HTTPs protocol (port 443), to maintain security during the communication of data to the system's back-end. The same can be said for the communication between the system and authority users or system managers, as well as the communication between the system and the Government's Database.

3.2 Functional Requirements

After the use case diagram, the main use cases are listed, then two sequence diagrams shows some of their usage. The use cases can be divided in three major groups: the first one refers to the End Users, the second refers to Authority Users and System Managers who can access to more complicate functionalities. Finally the last use cases focus only on the System Managers and their jobs to the system maintenance. Login of the Authority User and System Manager is in common in the use case diagram because the method is identical: they use an authorized code and a password. The login of the End User is different, instead, because it uses an username and a password.

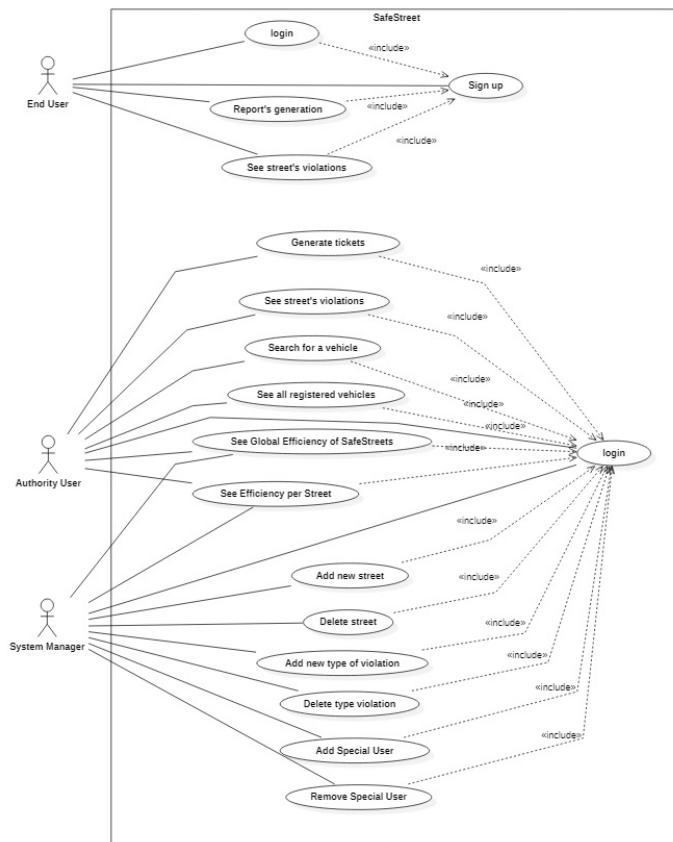


Figure 10: Use case diagram

3.2.1 Use Cases

End-User's perspective

The following tables describe the use cases from the perspective of a end user.

| | |
|------------------------|---|
| UC1.1 | Sign up (end user's perspective) |
| Name or ID | The guest wants to register himself/herself to the system using the app |
| Description | Guest |
| Actors | The guest has a fiscal code |
| Preconditions | |
| Flow of events | <ol style="list-style-type: none">1. The guest download and install the app2. The guest opens the app and clicks on the sign up button3. The system reacts showing a form with username, fiscal code, password fields4. The guest fills in the form5. The guest clicks on the sign up button6. The system validates data and registers the new user's data in the system |
| Post conditions | The guest is now a registered user (as end user) and their data are now registered in the system |
| Exceptions | Data inserted by the guest are incorrect, hence the flow of events restart from point 3. |

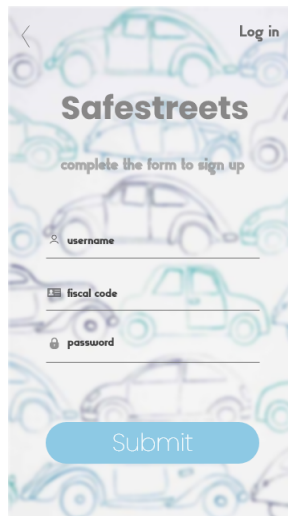


Figure 11: End user sign up

UC1.2

Name or ID

Description

Actors

Preconditions

Flow of events

Post conditions

Exceptions

Login (end user's perspective)

An already registered end user wants to access the system. In addition, he/she will also be able to perform all actions he/she, as an end user, is entitled to do.

End user

The end user has valid credentials

1. The end user opens the app and clicks on the log in button
2. The system reacts showing a form with username and password fields
3. The end user fills in the form
4. The end user click on the log in button
5. The system validates the end user's credentials

The end user is now logged in the system and can perform all the permitted actions

Data inserted by the end user are incorrect, hence the flow of events restart from point 2.

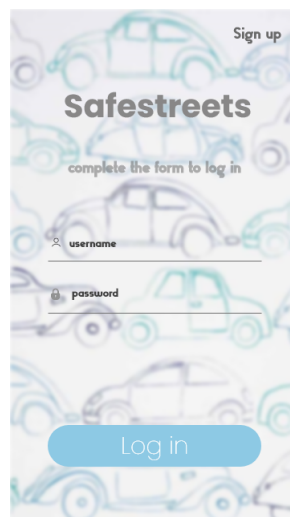


Figure 12: End user log in

UC1.3

Name or ID

Description

Actors

Preconditions

Flow of events

Report's generation (end user's perspective)

An end user wants to create a new report.

End user

The user is already logged

1. The end user clicks on the create report button
2. The system reacts showing a report homepage with "take a picture" button and a "add a vehicle" button
3. The end user clicks on the "take a picture" button
4. The system opens the mobile phone's camera
5. The end user takes a picture
6. The system accesses the position and the time
7. The end user go back on the report homepage
8. The end user clicks on the "add a vehicle" button
9. The system asks the end user to highlight the license plate in the pictures
10. The end user highlight the licence plate of the vehicle
11. The end user go back on the report homepage
12. The end user can add more than one vehicles that have committed the possible violation, flow of events restart from point 8
13. The end user clicks the proceed button and select a type of violation from a list
14. The end user clicks the send button and sends the report.

Post conditions

The user has now generated a report

Exceptions

1. The picture taken by the end user is invalid, hence the flow of events restart from point 4
2. The license plate is not selected correctly by the end user, hence the flow of events restart from point 9
3. The end user incorrectly clicks on the "add a vehicle" button when all the vehicles that committed the violation have already been added, the end user clicks on the "go back" button and hence the flow of events restart from point 11.

| | |
|-------------------|--|
| UC1.4 | |
| Name or ID | See street's violations (end user's perspective) |
| Description | An end user wants to see the number and types of violation in a specific street. |
| Actors | End user |
| Preconditions | The user is already logged |
| Flow of events | <ol style="list-style-type: none"> 1. The end user clicks on the "option" button in the app homepage 2. The app reacts showing a new button, "search street" 3. The end user clicks on the search button 4. The app reacts showing a form with city and address fields and a search button 5. The end user fills in the form 6. The end user clicks the search button 7. The system reacts showing the types of violations that occurred on that street in order of frequency without any other information |
| Post conditions | End user reaches the information about a specific street. |
| Exceptions | Data inserted by the end user are incorrect, hence the flow of events restart from point 4. |

Authority's and System Manager perspective

The following tables describe the use cases from the perspective of an Authority or System Manager.

UC2.1

Name or ID

Login (Authority and System Manager's perspective)

Description

An already registered Authority or System Manager user wants to access the system. In addition, he/she will also be able to perform all actions he/she, as an Authority or System Manager, is entitled to do.

Actors

Authority or System Manager

Preconditions

The user has valid credentials

Flow of events

1. The user opens the web application and clicks on the log in button
2. The system reacts showing a form with code and password fields
3. The user fills in the form
4. The user click on the log in button
5. The system validates validates user's credential

Post conditions

The user is now logged in the system and can perform all the permitted actions

Exceptions

Data inserted by the user are incorrect, hence the flow of events restart from point 2.

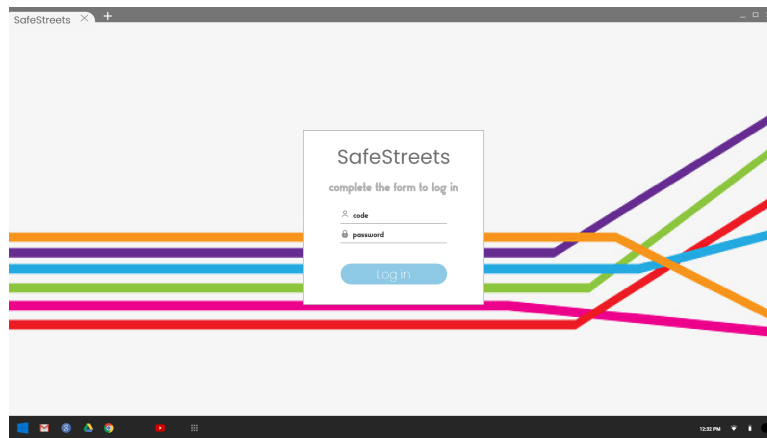


Figure 13: Authority and System Manager log in

UC2.2

Name or ID

Description

Actors

Preconditions

Flow of events

Post conditions

Exceptions

Generate tickets (Authority's perspective)

Authority wants to generate one or more tickets

Authority

Authority is already logged and the web application shows the homepage

1. The Authority clicks on the "see more info" button for the report he/she wants to analyze
2. The system reacts showing the license plate of the vehicle, a pictures, the time and date of the report, the location and the type of violation
3. The Authority clicks the "Generate Ticket" button
4. The system reacts showing a list of the license plates indicated in the report and a "send" button
5. The Authority selects all the license places for which she/he want to generate a ticket and press send.

Authority generates one or more tickets for the report, the report became a violation.

The Authority decides that the report is invalid, clicks on the discard button and the report is cancelled.

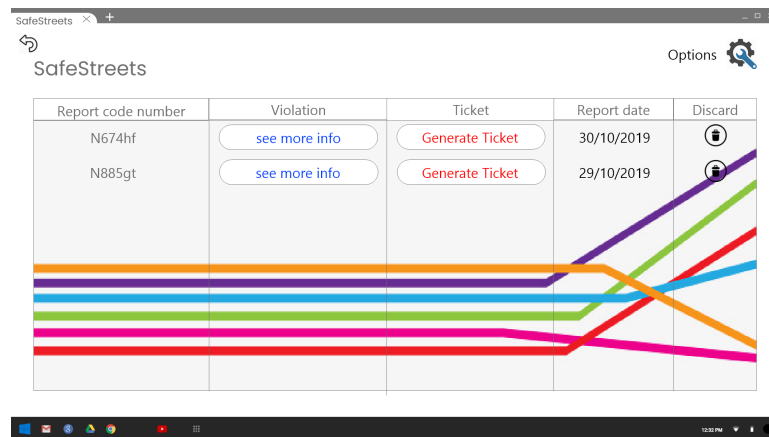


Figure 14: Tickets generation

| | |
|-------------------|--|
| UC2.3 | |
| Name or ID | See street's violations (Authority's perspective) |
| Description | Authority wants to see the number and type of violations in a specific street. |
| Actors | Authority |
| Preconditions | The Authority is already logged and the system shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The Authority clicks on the "Options" button 2. The system reacts showing a new web page 3. The Authority clicks on the "search street" button 4. The system reacts showing a form with city and address fields and a search button 5. The Authority fills in the form 6. The Authority clicks the search button 7. The system reacts showing the types of violations that occurred on that street in order of frequency and the list of license plates for each type of violations |
| Post conditions | Authority reaches the information about a specific street and license plates of the vehicles involved in the violations. |
| Exceptions | Data inserted by the Authority are incorrect, hence the flow of events restart from point 4. |
| UC2.4 | |
| Name or ID | Search for a vehicle (Authority's perspective) |
| Description | Authority wants to see all the vehicle information knowing the license plate. |
| Actors | Authority |
| Preconditions | The Authority is already logged |
| Flow of events | <ol style="list-style-type: none"> 1. The Authority clicks on the "Options" button 2. The system reacts showing a new web page 3. The Authority clicks on the "search vehicle" button 4. The system reacts showing a form with a licence plate fields and a search button 5. The Authority fills in the form 6. The Authority clicks the search button 7. The system reacts showing the vehicle searched, with the information (time, date, position, picture, type and offender) on the violations committed |
| Post conditions | Authority reaches the information about a specific vehicle . |
| Exceptions | Data inserted by the Authority are incorrect, hence the flow of events restart from point 4. |

| | |
|----------------------------|--|
| UC2.5 | |
| Name or ID | See all registered vehicles (Authority's perspective) |
| Description | Authority wants to see all the vehicles registered in the system. |
| Actors | Authority |
| Preconditions | The Authority is already logged, the system shows the homepage, the vehicles registered have already committed at least one violation |
| Flow of events | <ol style="list-style-type: none"> 1. The Authority clicks on the "Options" button 2. The system reacts showing a new web page 3. The Authority clicks on the "show all vehicle" button 4. The system reacts showing a list of all vehicles in the system in license plates' alphabetical order with a "info" button for each vehicle 5. The Authority clicks the info buttons of the vehicles he/she wants to analyze 6. The system reacts showing the vehicles information |
| Post conditions | Authority reaches the information about all the vehicles. |
| Exceptions | There are no registered vehicles yet, the system shows the web application homepage. |
| Efficiency analysis | |
| UC2.6 | |
| Name or ID | See Global Efficiency of SafeStreets (Authority and System Manager's perspective) |
| Description | The user wants to see the global efficiency of SafeStreets. |
| Actors | Authority, System Manager |
| Preconditions | the user is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The user clicks on the "Options" button 2. The system reacts showing a new web page 3. The user clicks on the "Efficiency" button 4. The system reacts showing the a list of type of efficiency to analyze 5. The user clicks the Global Efficiency button 6. The system reacts showing global efficiency charts the user see the Global Efficiency. |
| Post conditions | . |
| Exceptions | . |

| | |
|-------------------|--|
| UC2.7 | |
| Name or ID | See Efficiency per Street (Authority and System Manager's perspective) |
| Description | Authority or System Manager wants to see the efficiency of SafeStreets in a specific street. |
| Actors | Authority, System Manager |
| Preconditions | the user is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The user clicks on the "Options" button 2. The system reacts showing a new web page 1. The user clicks on the "Efficiency" button 2. The system reacts showing the a list of type of efficiency to analyze 3. The user clicks the Street's Efficiency button 4. The system reacts showing a form with city and address fields 5. The user fills the fields and presses the search button 6. The system shows the efficiency of SafeStreets in the selected street. |
| Post conditions | the user see the efficiency of SafeStreets in a specific street. |
| Exceptions | Data inserted by the user are incorrect, hence the flow of events restart from point 4. |

System Manager perspective - Maintenance System

The following tables describe the use cases from the perspective of System Manager in the maintenance of the system.

| | |
|-------------------|---|
| UC3.1 | |
| Name or ID | Add new street (System Manager's perspective) |
| Description | System Manager wants to add a new street in the system. |
| Actors | System Manager |
| Preconditions | the System Manager is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none">1. The System Manager clicks on the "Add a street" button2. The system reacts showing a form with a street and city fields and a add button3. The System Manager fills the form and clicks add4. The system validates data and registers the new street data or restore the data in the system |
| Post conditions | A new street is added or restored in SafeStreets. |
| Exceptions | <p>Data inserted by the user are incorrect, hence the flow of events restart from point 2.</p> <p>The street is already registered in the system, hence the flow of events restart from point 2.</p> |

| | |
|-------------------|--|
| UC3.2 | |
| Name or ID | Delete street (System Manager's perspective) |
| Description | System Manager wants to delete a street from the system. |
| Actors | System Manager |
| Preconditions | The System Manager is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The System Manager clicks on the "Delete a street" button 2. The system reacts showing a form with a street and city fields and a delete button 3. The System Manager fills the form and clicks the delete button 4. The system validates data and remove the street's data from the system for future reports |
| Post conditions | The selected street is cancelled from SafeStreets, past violation data is maintained, future reports will not have that position available. |
| Exceptions | Data inserted by the user are incorrect, hence the flow of events restart from point 2. |

| | |
|-------------------|--|
| UC3.3 | |
| Name or ID | Add new type of violation (System Manager's perspective) |
| Description | System Manager wants to add a new type of violation in the system. |
| Actors | System Manager |
| Preconditions | The System Manager is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The System Manager clicks on the "Add a type" button 2. The system reacts showing a form with type fields and a add button 3. The System Manager fills the form and clicks add 4. The system validates data and registers the new type or restored in the system |
| Post conditions | A new type of violation is added or restored in SafeStreets. |
| Exceptions | The type is already registered in the system, hence the flow of events restart from point 2. |

| | |
|-------------------|---|
| UC3.4 | |
| Name or ID | Delete violation type (System Manager's perspective) |
| Description | System Manager wants to delete a violation type from the system. |
| Actors | System Manager |
| Preconditions | The System Manager is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The System Manager clicks on the "Delete a type" button 2. The system reacts showing a list of all the violation types 3. The System Manager clicks the type he/she wants to delete 4. The system deletes the type of violation |
| Post conditions | The type of violation is cancelled from SafeStreets, future reports will not have that type of violation of the available options, past violation data is maintained |
| Exceptions | . |

| | |
|-------------------|--|
| UC3.5 | Add Special User (System Manager's perspective) |
| Name or ID | |
| Description | System Manager wants to add a new special user in the system |
| Actors | System Manager |
| Preconditions | The System Manager is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The System Manager clicks on the "Add a Special User" button 2. The system reacts showing a list of all types of Special Users (Authority user and System Manager) 3. The System Manager clicks the type of user he/she wants to add 4. The system reacts showing a form where to insert the password of the new user 5. The System Manager types the password of the new user in the form 6. The system registers a new Special User of the type selected with the password inserted, and shows the identifier code of the new user to the System Manager |
| Post conditions | A new special user is added in the system |
| Exceptions | Password inserted by the System Manager is empty or not acceptable. The flow of events restarts from point 4. |

| | |
|-------------------|---|
| UC3.6 | Remove Special User (System Manager's perspective) |
| Name or ID | |
| Description | System Manager wants to remove a special user in the system |
| Actors | System Manager |
| Preconditions | The System Manager is already logged and the web application shows the homepage |
| Flow of events | <ol style="list-style-type: none"> 1. The System Manager clicks on the "Remove Special User" button 2. The system reacts showing a form where to insert the code of the user to be removed 3. The System Manager types the identifier code of the user to be removed in the form 4. The system removes the user from the system |
| Exceptions | <p>Identifier code inserted by the System Manager is his/her own. The flow of events restarts from point 2.</p> <p>No user has the identifier code inserted by the System Manager. The flow of events restarts from point 2.</p> |

Sequence Diagram

The first sequence diagram focuses on the use case 1.2 "Login" from the End User's perspective. On the other hand, the second sequence diagram focuses on the use case 3.1 and the System Manager's side. In this scenario the System Manager asks to add a new street, the system is consequently responsible to verify if the street is not already active in the system. If there is a match, the street is already activated for reports usage and the system shows to the System Manager the form to fill again, instead if there isn't a match the new street can be added or restored and the System Manager is informed of the success.

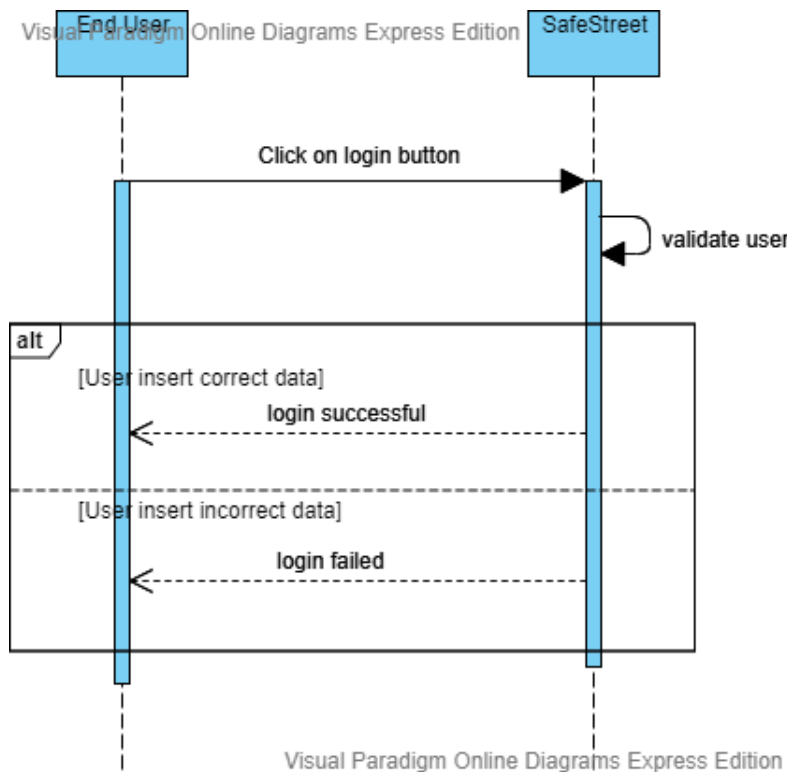


Figure 15: Sequence Diagram Login End User

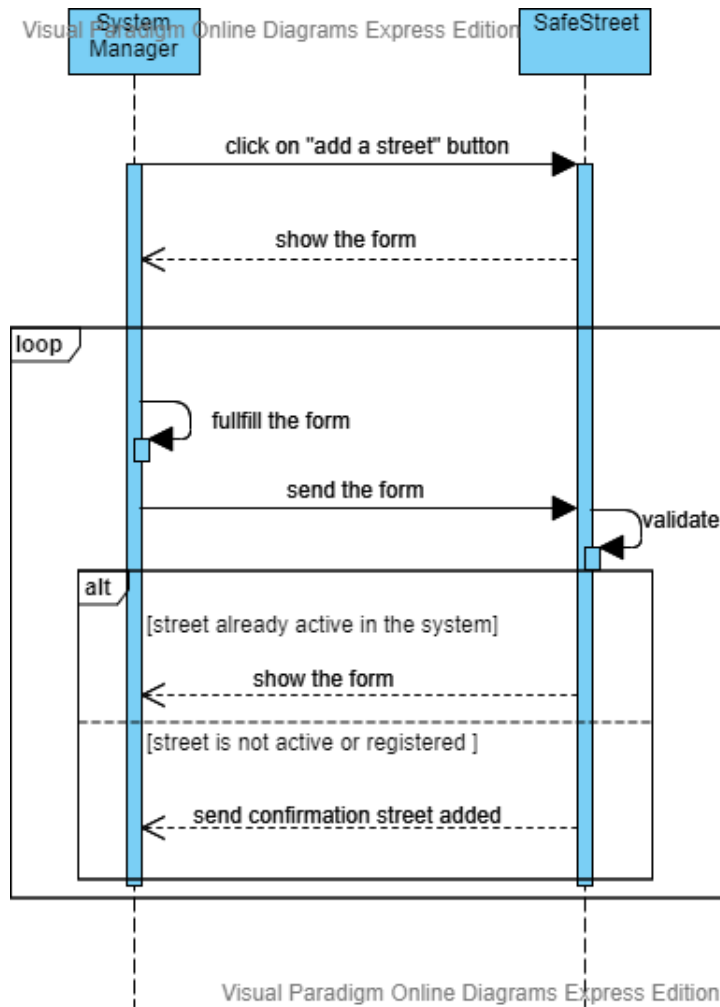


Figure 16: Sequence Diagram Add A Street

Scenarios

Scenario 1 - A bad way to start your day

A September morning Dylan woke up earlier for work: it was an important day for him he had a decisive meeting with a branch company who could have changed the lot of his promotion. He had just chose the perfect tie when, looking down from his widow, he noticed a vehicle parked right in front of his garage. Dylan quickly ran downstairs on the road and immediately realized that the vehicle blocked the driveway. As his vehicle was blocked and he had to take the bus to reach his meeting, Dylan couldn't wait for the police to arrive this is

why he used the SafeStreets app taking a picture of the possible violation and generating a report.

Scenario 2 - A perfect vacation

Marina had been daydreaming of her holidays for months, wondering where her, her husband and their two kids could have gone during the summer. As she searched through various hotels Marina started to think about their location. Since they travelled by vehicle she needed to find streets nearby the hotel where to park and where she knew her kids could cross and go to the beach with out her constantly worrying about them. That's when Marina thought about using the SafeStreets app, checking the safety of the streets nearby her hotel by going through the type and number of violations happening in a particular street.

Scenario 3 - Agent Danvers' quest

Agent Danvers had been for multiple weeks on a case of a mysterious vehicle which nearly every morning parked in front of a flower shop driveway, blocking the shop's business van. Since the old owner of the shop couldn't gave a proper description and the vehicle always vanished before Agent Danvers could arrive, she hadn't been able to generate a ticket. Only later, thanks to SafeStreets app, Agent Danvers finally received a picture of the infraction. Wrongly the old shop owner had highlighted both the vehicles' license plates, but through SafeStreets web application, Agent Danvers was able to generate a ticket only for the blocking vehicle, cancelling instead the report of the van, which was in a regular parking.

Scenario 4 - A new pedestrian path Mrs Swinson, one of SafeStreets' system managers recently received a communication that Shorebrook's new major, following the new urbanization policy of the town, decided to create a new pedestrian path and limited traffic zone, Downtown Street. Through SafeStreets web application Mrs Swinson added Downtown Street in the system, for future reports.

3.2.2 Mapping on functional requirements

For each goal, a list of requirements has also been itemised to provide an anticipation of what will be formally mapped in the traceability matrix by adding also the use cases.

[G.1] Allow each End User and Authority to be properly identified by the system

- R.1 The system must provide a sign up mechanism to end users. During the registration the System will ask to provide credentials.
- R.2 The System must check in the sign up process if the Guest credentials are valid: username not already taken by another registered User, fiscal code in the right format and password with a minimum length for the end user.
- R.3 The system must be able to keep track of each user's data.
- R.4 The system must provide a login mechanism to users.
- D.1 An end user is always supposed to sign up with his/her own fiscal code.
- D.2 An authority user or a system manager is always supposed to log in with his own authorized code.
- D.7 An authority user or a system manager always login with a desktop.
- D.8 A smartphone is different from a desktop.
- D.11 The internet connection works properly without failure.

[G.2] Allow End User to send pictures of the report , including their date, time and position

- R.5 The system must provide the end user with a way of sending a report.
- R.6 The system must be able to collect data from the user's devices.
- R.7 The system must be able to guarantee that the information of the reports is not altered.
- D.4 The smartphone of the end user has at least one photo-camera.
- D.5 The smartphone of the end user has a GPS system.
- D.6 The GPS of the smartphone of the end user always returns a correct location with a maximum error of 1 meter.
- D.10 Two or more streets (areas) can't overlap.
- D.11 The internet connection works properly without failure.

D.12 Every alteration of the data of the reports can be identified by the security software

D.14 The smartphone of the end user has a clock

[G.3] Allow End User to select report's type from a given list

R.8 The system must provide the end user with a way of selecting the report's type

D.11 The internet connection works properly without failure.

[G.4] Allow End User to send the license plate of the vehicle

R.9 The system must provide the end user with a way of sending the license plate of a vehicle.

D.4 The smartphone of the end user has at least one photo-camera.

D.11 The internet connection works properly without failure.

D.12 Every alteration of the data of the reports can be identified by the security software

D.13 Every license plate is registered in the DB of the Government

[G.5] Allow End User and Authority to see streets' number and types of violations

R.10 The system must be able to publish new data

R.11 The system must provide the user with a way of searching a street's information.

D.10 Two or more streets (areas) can't overlap

D.11 The internet connection works properly without failure

D.13 Every license plate is registered in the DB of the Government

[G.6] Allow Authority to see all the vehicles' information which have already committed a violation or search them by license plate

R.10 The system must be able to publish new data

R.12 The system must provide the Authority with a way of searching a vehicle's information.

R.13 The system must provide the Authority with a way of seeing all the vehicles' information.

D.11 The internet connection works properly without failure

D.13 Every license plate is registered in the DB of the Government

[G.7] Allow Authority to generate traffic tickets for violations

- R.10 The system must be able to publish new data
- R.14 The system must provide the Authority with a way of seeing a report's information.
- R.15 The system must provide the Authority with a way of delete a report.
- R.16 The system must provide the Authority with a way of selecting one or multiple license plate.
- R.17 The system must provide the Authority with a way of generate traffic tickets.
- D.9 There are enough authority users assigned to the application to manage all the reports of the day.
- D.11 The internet connection works properly without failure.
- D.12 Every alteration of the data of the reports can be identified by the security software.

[G.8] Allow Authority and System Managers to see the efficiency of SafeStreets

- R.10 The system must be able to publish new data.
- R.18 The system must provide the Authority and System Manager with a way of see Global Efficiency.
- R.19 The system must provide the Authority and System Manager with a way of see Street Efficiency.
- D.11 The internet connection works properly without failure.

[G.9] Allow a System Manager to do operations on the system for updating and maintenance

- R.20 The system must provide System Manager with a way of add a new street.
- R.21 The system must provide System Manager with a way of add a new violation type.
- R.22 The system must provide System Manager with a way of delete a street.
- R.23 The system must provide System Manager with a way of delete a violation type.
- R.10 The system must be able to publish new data.
- R.24 The system must be able to restore violations' type data.

R.25 The system must be able to restore streets' data .

D.11 The internet connection works properly without failure.

[G.10] Allow a System Manager to do user management operations

R.26 The system must provide System Manager with a way of add a new System Manager.

R.27 The system must provide System Manager with a way of add a new Authority.

R.28 The system must provide System Manager with a way of remove a System Manager.

R.29 The system must provide System Manager with a way of remove an Authority.

D.2 An authority user or a system manager is always supposed to log in with his own authorized code.

D.11 The internet connection works properly without failure.

3.2.3 Traceability Matrix

| Goal ID | Requirement ID | Domain ID | Use Case ID |
|---------|----------------------|----------------|-----------------|
| 1 | 1,2,3,4 | 1,2,7,8,11 | 1.1,1.2,2.1 |
| 2 | 5,6,7,14 | 4,5,6,10,11,12 | 1.3 |
| 3 | 8 | 11 | 1.3 |
| 4 | 9 | 4,11,12 | 1.3 |
| 5 | 10,11 | 10,11,13 | 1.4,2.3 |
| 6 | 10,12,13 | 11,13 | 2.4, 2.5 |
| 7 | 10,14,15,7,17 | 9,11,12 | 2.2 |
| 8 | 10,18,19 | 11 | 2.6,2.7 |
| 9 | 20,21,22,23,10,24,25 | 11 | 3.1,3.2,3.3,3.4 |
| 10 | 26,27,28,29 | 2,11 | 3.5,3.6 |

Goals

G.1 Allow each End User and Authority to be properly identified by the system;

G.2 Allow End User to send pictures of the report , including their date, time and position;

G.3 Allow End User to select report's type from a given list;

G.4 Allow End User to send the license plate of the vehicle;

G.5 Allow End User and Authority to see streets' number and types of violations;

- G.6 Allow Authority to see all the vehicles' information which have already committed a violation or search them by license plate;
- G.7 Allow Authority to generate traffic tickets for violations;
- G.8 Allow Authority and System Managers to see the efficiency of SafeStreets;
 - G.8.1 Allow Authority and System Managers to see the number of the tickets on the total reports globally;
 - G.8.2 Allow Authority and System Managers to see the trend of tickets over time globally;
 - G.8.3 Allow Authority and System Managers to see the number of the tickets on the total reports per street;
 - G.8.4 Allow Authority and System Managers to see the trend of tickets over time per street;
- G.9 Allow a System Manager to do operations on the system for updating and maintenance;
 - G.9.1 Allow a System Manager to add a violation type to the system;
 - G.9.2 Allow a System Manager to add a street to the system.
 - G.9.3 Allow a System Manager to delete a violation type to the system;
 - G.9.4 Allow a System Manager to delete a street to the system.

Use Cases

- UC1.1 : Sign up (end user's perspective);
- UC1.2 : Login (end user's perspective);
- UC1.3 : Report's generation (end user's perspective);
- UC1.4 : See street's violations (end user's perspective);
- UC2.1 : Login (Authority and System Manager's perspective);
- UC2.2 : Generate tickets (Authority's perspective);
- UC2.3 : See street's violations (Authority's perspective);
- UC2.4 : Search for a vehicle (Authority's perspective);
- UC2.5 : See all registered vehicles (Authority's perspective);
- UC2.6 : See Global Efficiency of SafeStreets (Authority and System Manager's perspective);
- UC2.7 : See Efficiency per Street (Authority and System Manager's perspective);
- UC3.1 : Add new street (System Manager's perspective);

- UC3.2 : Delete street (System Manager's perspective);
- UC3.3 : Add new type of violation (System Manager's perspective);
- UC3.4 : Delete violation type (System Manager's perspective).
- UC3.5 : Add Special User (System Manager's perspective).
- UC3.6 : Remove Special User (System Manager's perspective).

3.3 Performance Requirements

To provide a good grade of usability the following requirements must be satisfied:

- Statistics must be updated and published once every at maximum 24 hours.
- New reports must be published once every at maximum 24 hours.
- 98 percentage of users' requests shall be processed in less than 10 s.
- There is no limit on the total number of registered users.
- The system must support at least 10000 connected end user at once.
- The system must support at least 1000 simultaneously active authority users at once.

3.4 Design Constraints

3.4.1 Standards Compliance

The GPS-related information are given in form of Latitude and Longitude degrees.

The Time information is given in form of seconds, minutes, hour, day, month and year with the timezone.

3.4.2 Hardware Limitations

The system has to run under the following worst-case conditions:

- To run on a mobile phone the app requires at least:
 - 3G connection, at 2 Mb/s;
 - 20 MB of space;
 - GPS;
 - Clock;
 - 2 megapixel camera's resolution;
 - 2 GB of RAM for Android devices and 1 GB for iOS
- Web application:
 - 1 Mb/s Internet connection
 - 800x600 resolution

3.5 Software System Attributes

3.5.1 Reliability

The System should guarantees a 99 percentage of reliability: its components must have a failure rate that guarantees this goal.

3.5.2 Availability

Since the system wants to be a valid substitute to the classic method of tickets giving, the system must guarantee an availability of 98 percentage, with a downtime of 10 seconds.

3.5.3 Security

At least the following guidelines must be followed:

- All the communications between server and clients must be protected by strong encryption using the SSL protocol.
- All the communications between the server and the government server must be protected by strong encryption using the SSL protocol.
- All attempts of establishing an unsafe communication channel (e.g. plain HTTP) with the server must be refused.
- Users' passwords must not be stored in plain text in the system: instead, they must be hashed and salted.
- End Users' username must not be stored in plain text in the system: instead, they must be hashed and salted.
- Authorities' and System Managers' code must not be stored in plain text in the system: instead, they must be hashed and salted.
- The data of every report must be controlled by the security software to guarantee that no alteration has been done.
- If a report is altered, it must be discarded.

3.5.4 Maintainability

The code should be well documented using JavaDoc in order to enable other developers to easily understand and edit it. Moreover it is advisable for the developer to use other common and robust design patterns for boosting the maintainability in case of future issues.

3.5.5 Portability

The web application must support the current versions of Internet Explorer, Microsoft Edge, Mozilla Firefox, Google Chrome, Safari, Opera. The mobile application must be supported by the last 2 major versions of Android and iOS.

4 Formal Analysis using alloy

The alloy model that is presented is built to prove that the model proposed is consistent and to control that some important properties are satisfied. In particular, we selected three critical properties and we generated the relative worlds:

- **No Ticket has no Offender:** A traffic ticket can't be generated if it isn't given to someone (an Authority User must always specify to which persons he/she gives a traffic ticket) and that someone must be an offender in a report
- **Ticket not already generated:** Even if this application is meant to generate tickets, there can be a state in the machine where there are reports which hasn't still generated a ticket, because an authority user hasn't still viewed them. So, we must control that there can be a state where there are reports (so offenders), but still no tickets.
- **No Report if no Vehicle in Photo:** An End User can't generate a report if there are no vehicles in the photo he/she has taken.

The Alloy code is divided in signatures, facts and Predicates. For each predicate, the world generated is shown. At the end, it can be viewed the report of the proof of the reasoner. For simplicity, the timestamps are saved as Int. This simplification doesn't compromise the proof, however.

4.1 Signatures

```
// Signature for a generic String code
sig StringCode{}

// Every User of the system
abstract sig User {
    password: one StringCode,
}

// End Users of the application
sig EndUser extends User{
    issuedReport: set Report,
    id: one FiscalCode
}

// Authority Users
sig AuthorityUser extends User{
    id: one AuthorizedCode
}

// System Managers
sig SystemManager extends User{}

// Signature for a generic Identifier
abstract sig ID{
    code: one StringCode
}

// Fiscal Code used by End Users
sig FiscalCode extends ID{}
```

```

// Authorized Code used by Authority Users and System Managers
sig AuthorizedCode extends ID{}

// Location registered by the GPS of the smartphone of a End User
sig Location{
    latitude: one Int,
    longitude: one Int
}

// Photo added to a Report
sig Photo{
    report: one Report,
    containsVehicles: set Vehicle
}

//Time registered when the End User makes the Photo for the Report. For
    ↪ simplicity, we use Int for the timestamp
sig Time{
    timestamp: one Int
}{
    timestamp ≥ 0
}

// License Plate of a vehicle
sig LicensePlate{
    vehicle: one Vehicle
}

// Owner of a Vehicle
sig Owner{
    vehicle: some Vehicle
}

sig Vehicle{
    licensePlate: one LicensePlate,
    owner: one Owner
}

//Report made by the End User and sent to the system
sig Report{
    location: one Location,
    photo: one Photo,
    vehicles: some Vehicle,
    offender: some Owner,
    ticket: lone TicketList,
    time: one Time
}

// Set of Tickets generated for a Report
sig TicketList{
    // generated by only one Authority user
    givenBy: one AuthorityUser,
    // but given to one or more Owners
    givenTo: some Owner
}

```

4.2 Facts

```

// No different users have the same ID
fact uniqueID{
    no disj user1, user2: EndUser | user1.id = user2.id
    no disj user1, user2: AuthorityUser + SystemManager | user1.id =
        ↪ user2.id
    no disj id1, id2: ID | id1.code = id2.code
}

```

```

// No different Vehicles have the same License Plate
// No different License Plates have the same Vehicle
// If a vehicle has a License Plate, the License Plate is registered with
    ↪ that Vehicle
fact uniqueLicensePlate{
    no disj v1, v2: Vehicle | v1.licensePlate = v2.licensePlate
    no disj l1, l2: LicensePlate | l1.vehicle = l2.vehicle
    all v1 : Vehicle, l1: LicensePlate | ((l1 = v1.licensePlate) ≤> (l1.
        ↪ vehicle = v1))
}

// If a Report has a Photo, that Photo has that report as Report
// All the vehicles of the Report are the same of the pPhoto
fact consistencyPhotoAndReport{
    all p1 : Photo, r1: Report | ((p1 = r1.photo) ≤> (r1 = p1.report))
    all p1: Photo, r1: Report | (p1 = r1.photo) implies (p1.
        ↪ containsVehicles = r1.vehicles)
}

// Different Owners can't have the same Vehicle
fact uniqueOwner{
    no disj o1, o2: Owner | o1.vehicle = o2.vehicle
}

// Different End Users can't generate the same Report
fact uniqueAuthorReport{
    no disj e1, e2: EndUser | e1.issuedReport = e2.issuedReport
}

// Ticket must be given to an Offender of a Report
fact consistencyTicketReport{
    all t1: TicketList | (some r1: Report | (r1.ticket in t1) implies (t1
        ↪ .givenTo in r1.offender))
}

// Two Reports can't have the same TicketList
fact twoReportsCantHaveSameTicket{
    no disj r1,r2 : Report | r1.ticket = r2.ticket
}

// There can't be Time, Location or Photo without a Report
fact allTimeLocationPhotoHaveReport{
    all t1: Time | (some r1: Report | r1.time = t1)
    all l1: Location | (some r1: Report | r1.location = l1)
    all p1: Photo | (some r1: Report | r1.photo = p1)
}

```

4.3 Predicates

4.3.1 No Ticket has no Offender

```
// No Ticket must be generated if they are not given to someone
pred noTicketHasNoOffender{
  no t1: TicketList | #(t1.givenTo) = 0
  #TicketList > 0
  #givenTo > 0
}
run noTicketHasNoOffender for 2
```

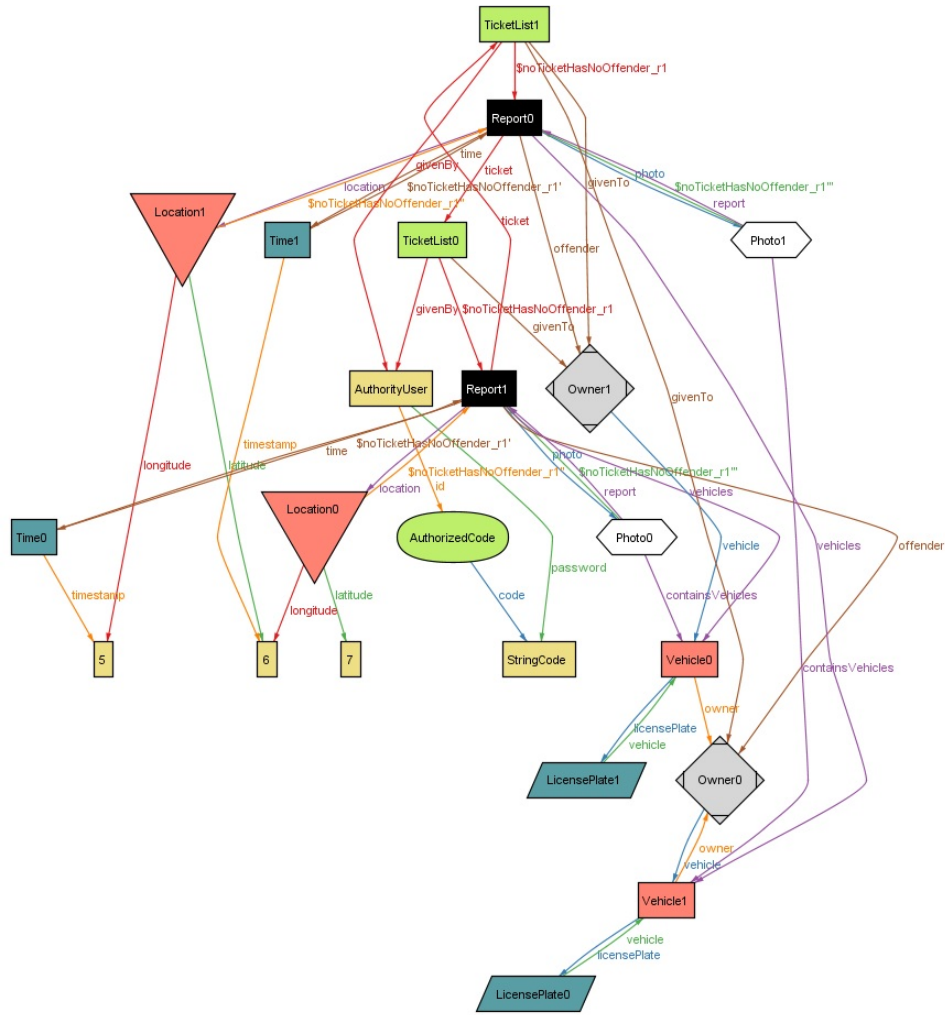


Figure 17: No Ticket Has No Offender

4.3.2 Ticket not already generated

```
// There can be a situation where there are Reports, but the Authority users
// → haven't still generated the Tickets
pred ticketsNotAlreadyGenerated{
  #TicketList = 0
  #offender > 0
}
run ticketsNotAlreadyGenerated for 3
```

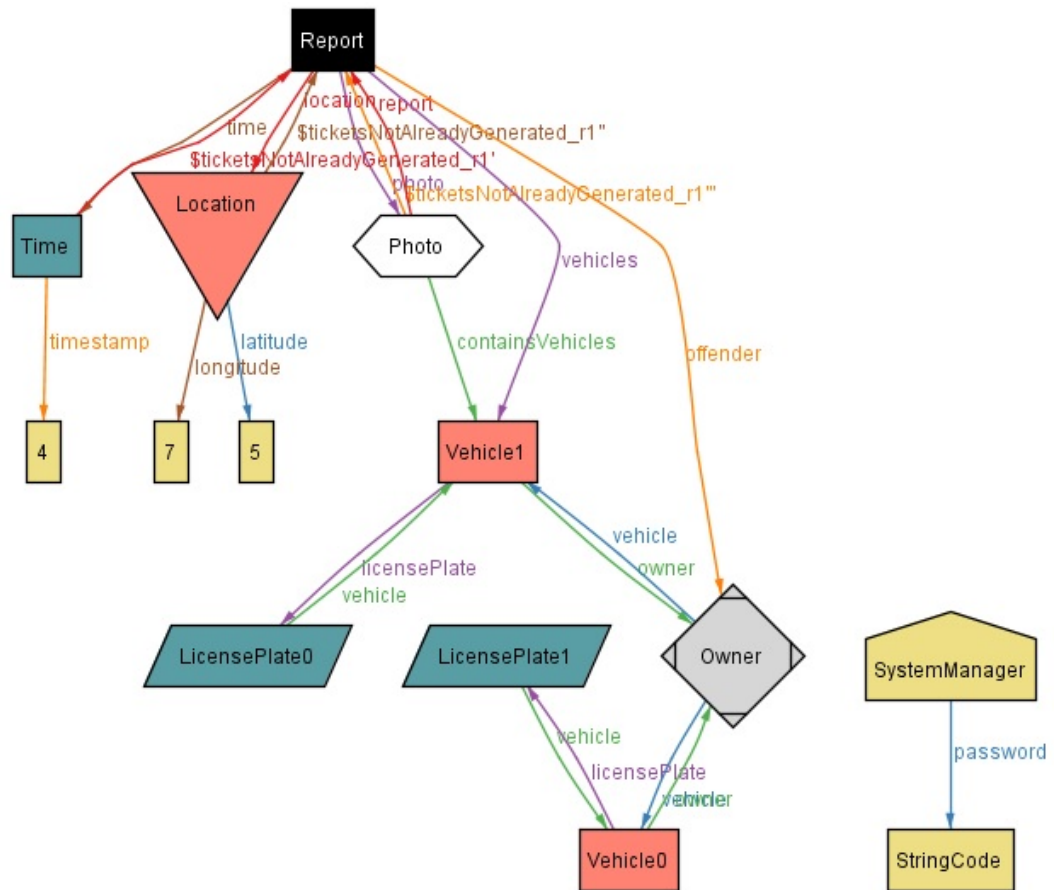


Figure 18: Ticket Not Already Generated

4.3.3 No Report if no Vehicle in Photo

```
// If no Vehicles are specified by the End User in the Photo, a Report can't
    ↳ be made
pred noReportIfNoVehicleInPhoto{
    #(Photo.containsVehicles) = 0 implies #Report = 0
}
run noReportIfNoVehicleInPhoto for 2
```

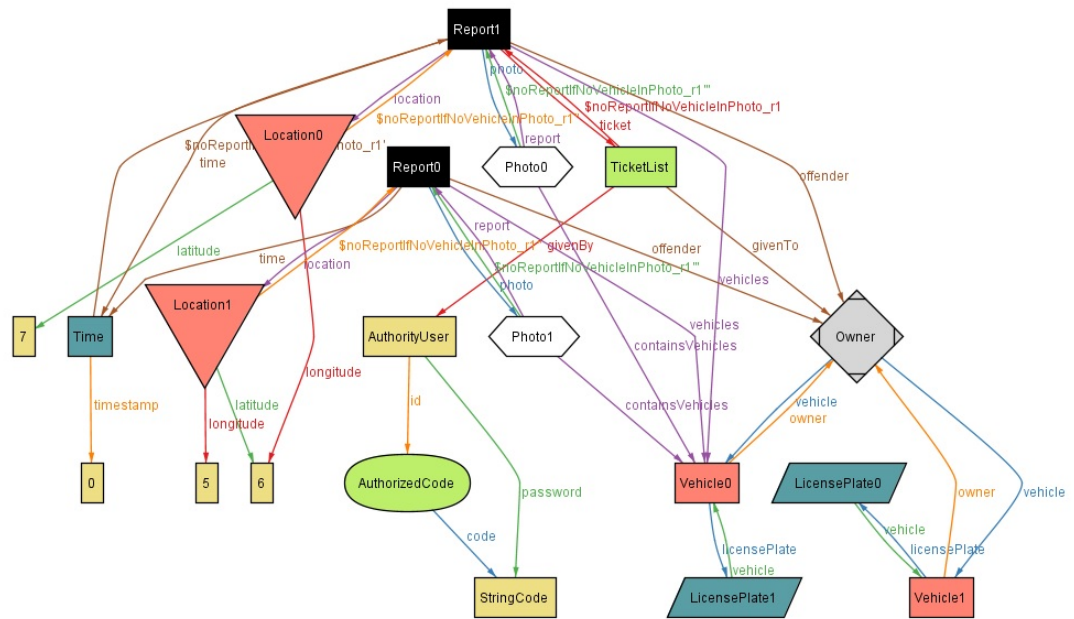


Figure 19: No Report If No Vehicle In Photo

4.4 Proof of consistency

This is the report of the reasoner about the three predicates.

```
Executing "Run noTicketHasNoOffender for 2"
  Solver=sat4j Bitwidth=4 MaxSeq=2 SkolemDepth=1 Symmetry=20
  1980 vars. 216 primary vars. 3796 clauses. 21ms.
  Instance found. Predicate is consistent. 60ms.

Executing "Run ticketsNotAlreadyGenerated for 3"
  Solver=sat4j Bitwidth=4 MaxSeq=3 SkolemDepth=1 Symmetry=20
  3994 vars. 393 primary vars. 7636 clauses. 36ms.
  Instance found. Predicate is consistent. 42ms.

Executing "Run noReportIfNoVehicleInPhoto for 4"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  6647 vars. 616 primary vars. 12536 clauses. 37ms.
  Instance found. Predicate is consistent. 32ms.
```

Figure 20: Proof of consistency

To confirm the proof of **noTicketHasNoOffender** and **noReportIfNoVehicleInPhoto**, we tried to execute two counterexamples, **someTicketHasNoOffender** and **someReportIfNoVehicleInPhoto**. No instance was found and world couldn't be generated. This confirms the proof of the two original predicates.

```
pred someTicketHasNoOffender{
  some t1: TicketList | #(t1.givenTo) = 0
  #TicketList > 0
  #givenTo > 0
}
run someTicketHasNoOffender for 2

pred someReportIfNoVehicleInPhoto{
  #(Photo.containsVehicles) = 0
  #Report = 1
}
run someReportIfNoVehicleInPhoto for 4
```

```
Executing "Run someTicketHasNoOffender for 2"
  Solver=sat4j Bitwidth=4 MaxSeq=2 SkolemDepth=1 Symmetry=20
  1998 vars. 218 primary vars. 3833 clauses. 22ms.
  No instance found. Predicate may be inconsistent. 5ms.

Executing "Run someReportIfNoVehicleInPhoto for 4"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  6679 vars. 616 primary vars. 12645 clauses. 31ms.
  No instance found. Predicate may be inconsistent. 9ms.
```

Figure 21: Counterexamples report

5 Effort Spent

Valeria Maria Fortina

| Date | Section | Hours |
|-------|--|-----------|
| 19/10 | Initial Brainstorming of the project | 1 |
| 20/10 | Purpose, Scope, Definitions, Acronyms, Abbreviations, First Part of Reference Documents and Document Structure | 2 |
| 22/10 | Rough Goals definition | 1.5 |
| 27/10 | Goals | 1.5 |
| 28/10 | User, Hardware, Software and Communication Interfaces, Goals revision, UI User with AdobeXD | 3 |
| 29/10 | User UI with AdobeXD | 3 |
| 30/10 | Part Use cases, Performance Requirements, Design Constraints, Software System Attributes | 4 |
| 2/11 | Functional Requirements: use cases and scenarios | 2 |
| 4/11 | Functional Requirements: use cases, User UI, mapping functional requirements | 6 |
| 5/11 | Functional Requirements scenarios, use cases, mapping functional requirements, sequence diagrams | 6.5 |
| 6/11 | New version of Purpose and Scope MW Phenomena | 3 |
| 7/11 | Corrections | 5 |
| 9/11 | Document Structure of chapter 3, corrections, complete revision | 5.5 |
| | | Total: 44 |

Alessio Galluccio

| Date | Section | Hours |
|-------|---|-------------|
| 19/10 | Initial Brainstorming of the project | 1 |
| 20/10 | Class diagram in UML, Introduction of part 2 | 4 |
| 27/10 | Goal definitions, Requirements definitions, Product perspective, User characteristics, Domain assumptions | 6 |
| 28/10 | State diagrams | 1.5 |
| 30/10 | Writing of Alloy code | 3 |
| 1/11 | Writing of Alloy code | 1 |
| 4/11 | Writing of Alloy code, Debugging, Corrections of the text | 6.5 |
| 5/11 | Alloy worlds, Alloy images, Writing part 4, Updating of Class and State diagrams | 3.5 |
| 6/11 | Use Case diagram, Corrections | 2.5 |
| 7/11 | Corrections of the text | 6 |
| 8/11 | Alloy corrections, Counterexamples in Alloy | 2 |
| 9/11 | Corrections, Complete revision | 4.5 |
| | | Total: 41.5 |

6 References