Politecnico di Milano

A.A 2019/2020

Requirements Analysis and Specifications Document

SafeStreets

Authors:

Armenante Valerio

Capaldo Marco

Di Salvo Dario

Contents

1 Introduction

1.1 Purpose

The aim of this document is to give an overview of the requirements and specifications of the system to be developed. The goal of the document is to describe in detail all functional and non-functional requirements of the system, analysing the needs of the customer and explaining common use case scenarios. It will set a baseline for project planning and cost estimation, giving a detailed insight to all stakeholders which include the SafeStreets Investors and engineers (present and future) involved in development, testing and maintenance.

1.2 Scope

1.2.1 Description of the given problem

SafeStreets Basic Service

The given problem is to allow users to notify traffic violations to authorities. More specifically, the notification process consists in the acquisition of pictures, date, time, location and type of reported violation inserted by user. The aim of SafeStreets basic service is to retrieve and store information from inputted data, completing it with suitable metadata. In particular, when user sends pictures related to a traffic violations, SafeStreets is able to recognize license plate by means of an external algorithm. After that, SafeStreets dispatches the notification to the authority member, which is in an available status and is chosen according to the minimum distance between the notified traffic violation and author-

ity member's position. In the end, the assigned authority member will opportunely manage the received request. In addiction, users and authorities member, through a Mobile Application interface presenting different level of visibility to different roles, will be able to mine information that has been received, for example by highlighting the areas with the highest frequency of violations. This is going to be realized designing some algorithms able to generate statistics on the previously stored data. A user who approaches the mobile application for the first time has to complete a registration process, as well as an authority which approaches the dedicated website for the first time. In particular, an authority inserts all their authority member in SafeStreets database through a website's form specifying their e-mail and unique code. In this way, each authority member have to do only an activation process, inserting their personal data and the assigned unique code.

SafeStreets Advanced Function 1

The goal of this SafeStreets' first advanced function is to improve more the security of cities through a cooperation with municipality. First of all SafeStreets retrieves information from municipality through an external service, then information are categorized (accidents, violations, etc.) and some statistics are generated. Then crossing information between resultant elaboration and already available SafeStreets data, the goal can be reached. More specifically, if traffic violations cardinality, related to an area, exceed a specified threshold, SafeStreets identifies the area as unsafe and suggests possible interventions to municipality. Obviously, the threshold is properly defined according to the characteristics of the city.

SafeStreets Advanced Function 2

The goal of this SafeStreets' second advanced function is to provide correct information, related to traffic violations, to municipality(in particular local police). In this way, local police authority can easily and correctly emits traffic tickets to the offenders. In order to ensure the correctness of information, SafeStreets implements an algorithm that recognize if a manipulation is occurred on uploaded data by users. Moreover, Safestreets builds some statistics regarding traffic tickets data such as a ranking of the most offenders, most common violations and the effectiveness of SafeStreets initiative.

Phenomenon	Shared	Who controls it
User wants to contact an authority	NO	World
User reports a traffic violation	YES	World
The machine ensures the correctness of information contained in a traffic violation	NO	Machine
The machine rejects the traffic violation to the user	YES	Machine
User wants to know information related to safeness of his city	NO	World
The machine build statistics related to traffic violations received	NO	Machine
The machine suggests to municipality some improvements	YES	Machine
An authority member receives a traffic notification	YES	World
An authority member sets his status as available	YES	World
The machine dispatches a traffic notification to the nearest authority	YES	Machine
Local police generates traffic tickets	NO	World
The machine builds some statistics regarding traffic tickets	NO	Machine

Table 1.1: World and Machine table.

1.2.2 Goals

Here are summarized goals that must be achieved by SafeStreets:

[G1]: Users can be uniquely identified, thanks to the completion of the Registration Process.

[G2]: Authorities can be uniquely identified, thanks to the completion of Registration Process.

[G3]: Authority members can be uniquely identified, thanks to the completion of Authentication Process.

[G4]: Allows users to notify authorities when traffic violations occur.

[G5]: Allows authority members to receive the notifications about traffic violations in order to increase the local security.

[G6]: Allows end users to mine information on statistics built by the software itself.

[G7]: Allows authority members to mine information on statistics built by the software itself.

[G8]: Builds a cross information analysis between municipality's data and itself data in order to improve reliability of the service and suggests to municipality possible interventions.

[G9]: Allows municipality (in particular local police authority) to retrieve traffic violations in order to generate relative traffic tickets.

[G10]: Builds statistics using information related to emitted traffic tickets.

1.3 Definition, Acronyms and Abbreviations

1.3.1 Definition

Customer: It is used inside the document when a functionality is ref erred to End Users, Authority and Authority members.

Hardened Database Model: hardening refers to providing various means of protection in a computer system.

1.3.2 Acronyms

GPS – Global Positioning System

RASD – Requirement Analysis and Specification Document

SS - SafeStreets

DB - Database

AI - Artificial Intelligence

AM - Authority Member

1.3.3 Abbreviations

[Gn] – n-goal

[Dn] – n-domain assumption

[Rn] – n-functional requirement

1.4 Revision History

- Version 1.0:

  - First Release

- Version 1.1:

  - Some requirements have been modified in order to mantain the coherency with the Design Document.

  - Use case descriptors has been revisited.

1.5 Reference Document

- IEEE Std 830-1998: "IEEE Recommended Practice for Software Requirements Specifications"
- Project description: "Assignments AA 2019-2020.pdf"
- Alloy Language Reference : `https://alloytools.org/documentation.html`
- UML Language Reference : `https://www.utdallas.edu~chung/Fujitsu/UML_2.0/Rumbaugh--UML_2.0_Reference_CD.pdf`

1.6 Overview

1. **Introduction**: this section gives a general description of the software and its characteristics.
2. **Overall Descripion**: this section gives a general description of the software and its characteristics.

3. **3. Specific Requirements**: this section gives a deep insight about the system's main functionality, analyzing scenarios with their relative use-cases and includes requirements(functional and not). Event flows are model using sequence and state diagrams.

4. **Formal analysis using Alloy**: this section provides an Alloy model in order to give a detailed description of SS. Alloy is a declarative language used for specifying models of system and software.

2 Overall Description

2.1 Product Perspective

The whole system is going to be developed from scratch, according to the choices written in this document. The registration process is always costless. Moreover, access and utilization for End Users, Authority member and Authority is free. This important choice has been made because of SafeStreets aim is to improve the security related to traffic violations, not a reason for earning money.

2.1.1 External Functions

FindOwnerPlate is an external service that given in input a number plate then returns the owner of the vehicle. This service is well-integrated with SS and it is used to make faster traffic tickets process.

2.2 Product Functions

Starting from the assumption that all the goals written in the previous part will be offered as functionalities of our system, we're going to precisely list all the technical functions that the product will offer after its realization.

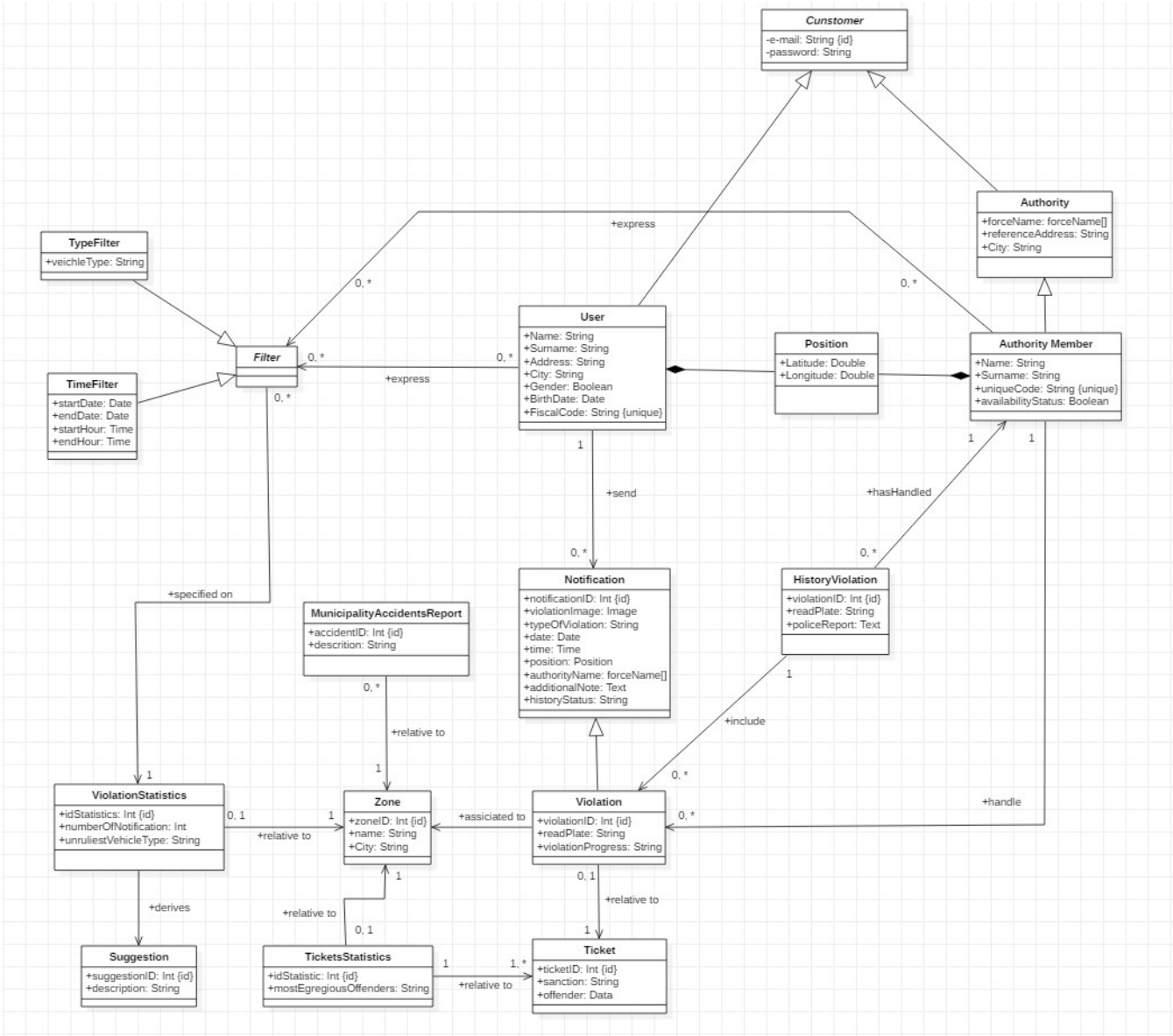

Figure 2.1: Class diagram.

2.2.1 Data storing management

A relational DB is made available as storage memory for the whole ecosystem, in which all personal and non personal information, together with Customers' gathered data, will be saved. It is mandatory to store data in a way that both security measures for sensitive information as well as fast and reliable access possibilities are guaranteed. To reach [G1] and [G2], it is necessary to exploit all the facilities provided by the Hardened Database model. The aim is to prevent data loss, leakage, or unauthorized access. At the registration time of each Customer, a new tuple will be created into the respectively table. Each Customer tuple will be uniquely linked to the Customer who own it. To reach that, a unique identifier will be generated for every Customer, to ensure that the associated data will be correctly extractable from the DB without ambiguities. The identifier assignment and the DB scanning mechanism will be chosen to be as fast as possible, since the system is going to be built for efficiently managing a huge amount of customers. Sequential DB scanning approaches as well as progressive identifiers assignment algorithms will be ignored during the choice. Finally, to ensure that any kind of information related to traffic violations are not altered, an encryption mechanism is put in place.

2.2.2 Data sharing management

SafeStreets gives top priority to Customer's privacy. No data and traffic notification access will be granted to other Users. When a user sends a notification to an authority member, this one will share some limited personal data such as unique code, name and surname. All traffic violations can be accessed by authority members and SafeStreets gives the possibility to filter violations using specific criteria. Moreover, an authority member can access to a personal panel that shows past violations managed by him.

2.2.3 Traffic violations handler

After that an end user sends traffic violations using SafeStreets service, then they are forwarded to the nearest authority member using a specific handler. SafeStreets uses an external service (such as Google Maps) to calculate the time spent by each authority member, in that zone, to arrive in traffic violation position. Then SafeStreets handler selects the authority member that has the minimum arrival time and this is done because often minimum distance, inside city, does not imply faster reaction.

2.2.4 Municipality collaboration service

In order to ensure a correct collaboration with municipality, SafeStreets' system administrator communicates to an employee of municipality how information must be sent. After that a protocol is chosen then SafeStreets is able to store all information inside its DB and uses a specific algorithm to cross information. Using that analysis, SS identifies unsafe areas and suggests some pre-set improvements to municipality through an e-mail.

2.2.5 RecognizePlate

RecognizePlate is a well-built algorithm that permits to SafeStreets to recognize plate from pictures sent by end users. When a picture is received, then RecognizePlate uses advanced AI techniques to recognize plate in the image and returns to SafeStreets the number plate.

2.2.6 Traffic tickets handler

When a certified traffic violations are sent to an authority, belonging to local police, SafeStreets uses RecognizePlate and FindOwnerPlate, explained before, is able to provide all useful information to make traffic tickets. Ob-

viously, SafeStreets gives the possibility to discard a traffic violation to the assigned authority.

2.3 User characteristics

2.3.1 Actors

- **Visitor**: a person which has not yet completed the Registration Process. The only thing which it is able to do is to start the registration process.
- **User**: a person using SafeStreets services which has completed the Registration Process.
- **Authority**: an employee of an institution who represents a specific station of the institution itself.
- **Authority member**: a recognized authority member. It uses SafeStreets to receive notification related to traffic violations.
- **Municipality**: an employee of Municipality who have the task to manage the communication with SafeStreets.
- **System Administrator**: a person in charge of keeping the system up to date, checking Authority policies and behaviors.
- **Device**: a device used to transfer significant, and well-represented, world Data to the system such as pictures of traffic violations.

2.4 Assumptions, dependencies and contraints

2.4.1 Domain assumptions

[D1] - Authorities correctly insert address of reference station.

[D2] - Authorities must provide to its authority member the unique code.

[D3] - Authority members specify correctly their availability status.

[D4] - Each authority member must have an uniquely identifiable code.

[D5] - Devices used by end users are supposed to have a camera and an integrated and enabled GPS sensor.

[D6] - Sent position is assumed to be reliable and precise.

[D7] - System is supposed to be well integrated with reading plate algorithm that has been already designed and is correctly working.

[D8] - Each already uploaded notification of violation is every time correctly received and stored by the software system.

[D9] - The internet connection works properly without failure.

[D10] - Every time an authority member starts working, has to logs into the application and sets properly availability status.

[D11] - Authority members knows the local traffic laws and the related fines.

[D12] - Authority member that accept to provide an intervention must check the correctness of traffic violations notified and signals to Safe Streets.

[D13] - Municipality service is well integrated with SafeStreets.

[D14] – Municipality has an active mail system and it is periodically checked by its own employee.

[D15] – Municipality can fulfill the improvements suggested by the software.

[D16] – External service(FindOwnerPlate) is well integrated with SafeStreets that permits to retrieve personal data of the vehicle's owner.

3 Specific Requirements

3.1 Exsternal Interface Requirements

3.1.1 Customer Interfaces

Figure 3.1: SafeStreets home view.

Figure 3.2: SafeStreets Sign Up.

Figure 3.3: Authority registration.

Figure 3.4: AM authentication.

Figure 3.5: User Registration.

Figure 3.6: Login view.

Figure 3.7: User Menu view.

Figure 3.8: Authority Member menu.

Figure 3.9: Authority menu.

Figure 3.10: Notification's form user.

Figure 3.11: User notifications.

Figure 3.12: Statistics for user.

Figure 3.13:AM notifications.

Figure 3.14: Authority tickets.

Figure 3.15: Authority members' list.

Figure 3.16: AM summary stats.

3.1.2 Hardware and Communication Interfaces

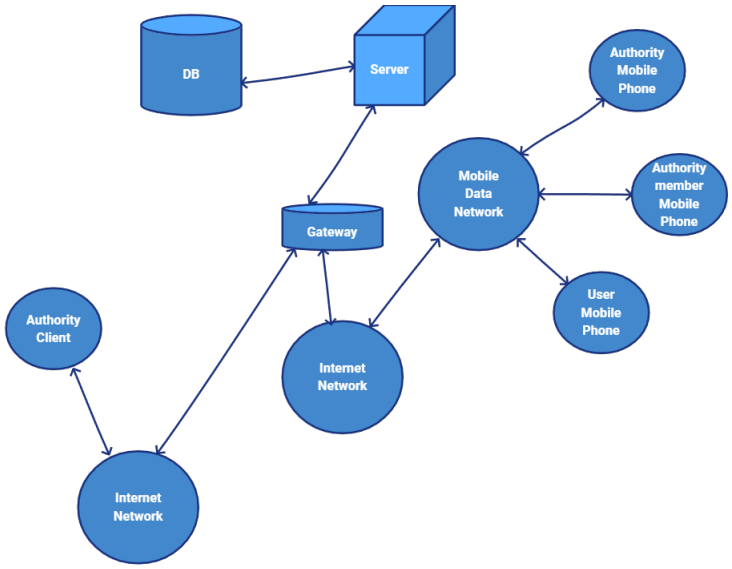

Figure 3.17: Communication architecture.

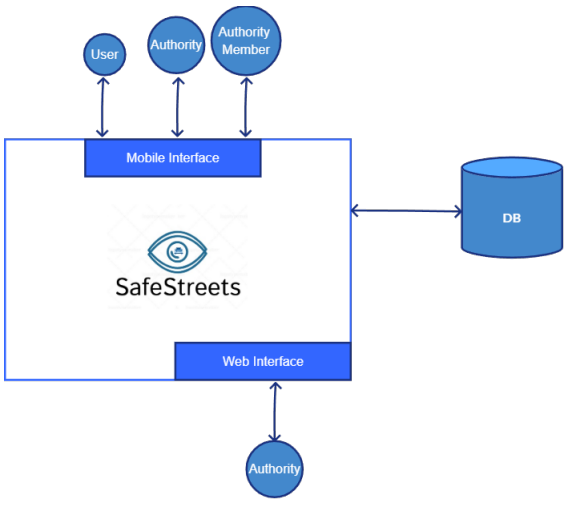

Figure 3.18: Application diagram.

3.2 Scenarios

3.2.1 Scenario 1

Robert has a repair workshop in the heart of the city. Every morning someone parks in front of his garage's workshop . So one of his employees advises him to install the SafeStreets application in his phone, so in this way he can notify to authorities the parking violation. Robert, after has downloaded the application, does the registration process and sends the traffic violation, filling the form and selecting the local police as authority. A local police authority member will arrive as soon as possible on the place, taking the relative judgment.

3.2.2 Scenario 2

Catherine booked a trip in a city thanks to her summer savings. On her journey, however, she realized that she needed help to understand the unsafe areas of the city. For this reason, Catherine asks in a police station whether or not there is a method of being able to have information on the streets that she have to avoid. The police reccomends her to download SafeStreets. After downloading and installing the application, Catherine proceeds to registration process as user. After that, she can interact with SafeStreets interface and access to the "Statistics" section, where she finds all the desired information about safeness in that city.

3.2.3 Scenario 3

During everyday life, authorities have difficulties due to the high number of violations in metropolis. Directors of the various authority station (for example police, local police, etc.) gathered with the relative municipality to find some improvement that could help them to better manage all road violations. Thanks to a survey, the municipality finds SafeStreets to better

manage the safeness of city streets. From this moment on, each authority joins to SafeStreets service, registering all its employees and each one of them will be able to log in to the application with a unique code provided to each of them. Finally, authority member can receive traffic violations by mean of notifications that are automatically managed by the system and showed to them through a panel.

3.2.4 Scenario 4

Local police station director noted that his employees have some difficulties in managing fines or traffic tickets in the best possible way. Sometimes, it may happen that fines are referred to wrong violations, or sometimes referred to the wrong car or other types of human errors. The local police director heard about the excellent impact that SafeStreets service have had in other cities, so he decided to register his station, allowing users to access it. After downloading and installing the application, and registering all employees via forms, then all employees of the local police station are able to take in a correctly way the information about a traffic violations and generate the fines/tickets.

3.2.5 Scenario 5

A municipality has to decide in which area of city organize a worldwide event that will be in 6 months. In order to accomplish this, the mayor contacts the authority, asking which are safe areas in according to collected data. The authorities are able to respond exactly to the request made by the municipality, since after registering on SafeStreets they can obtain information, thanks to the crossing of data analysis carried out by the application itself, on which areas are most secure in fast and exact way to the municipality. So now the municipality can decide thanks to given data where can manage the important event in the city.

3.3 Use cases diagrams

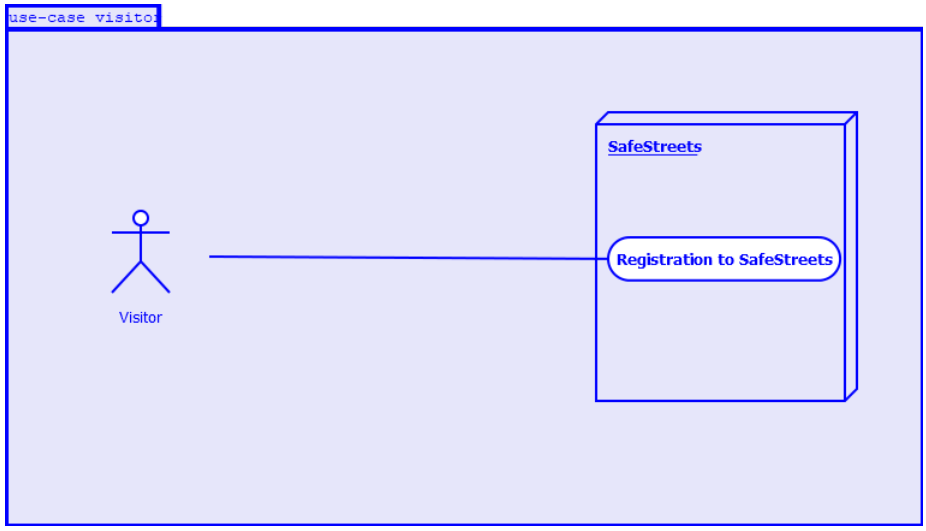

Figure 3.19: Visitor use case.

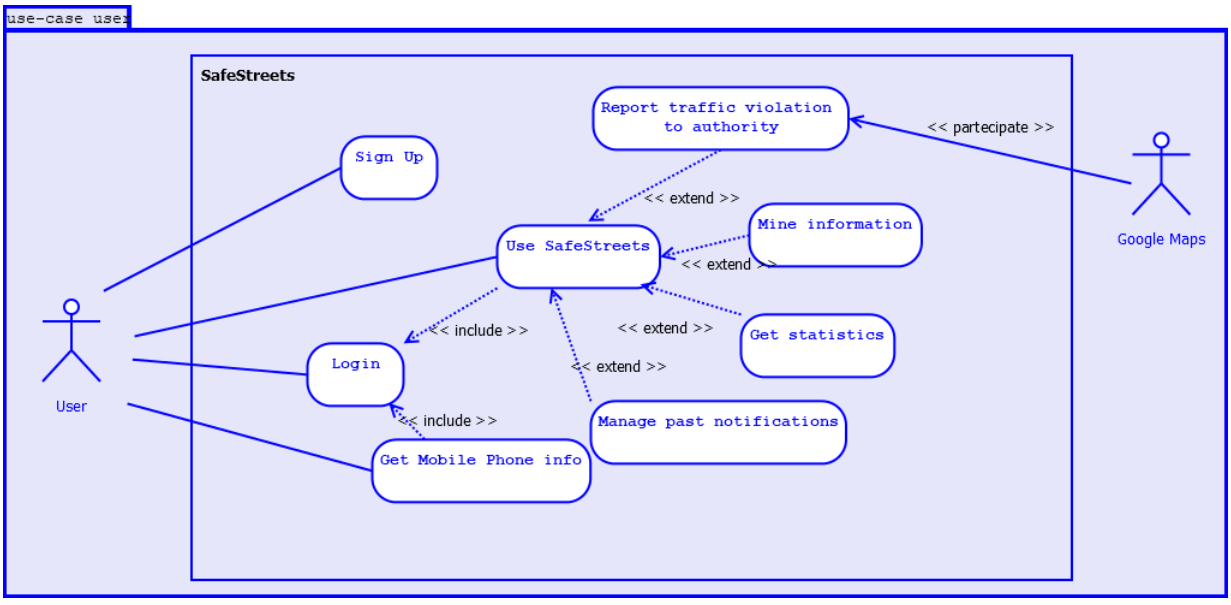

Figure 3.20: User use case.

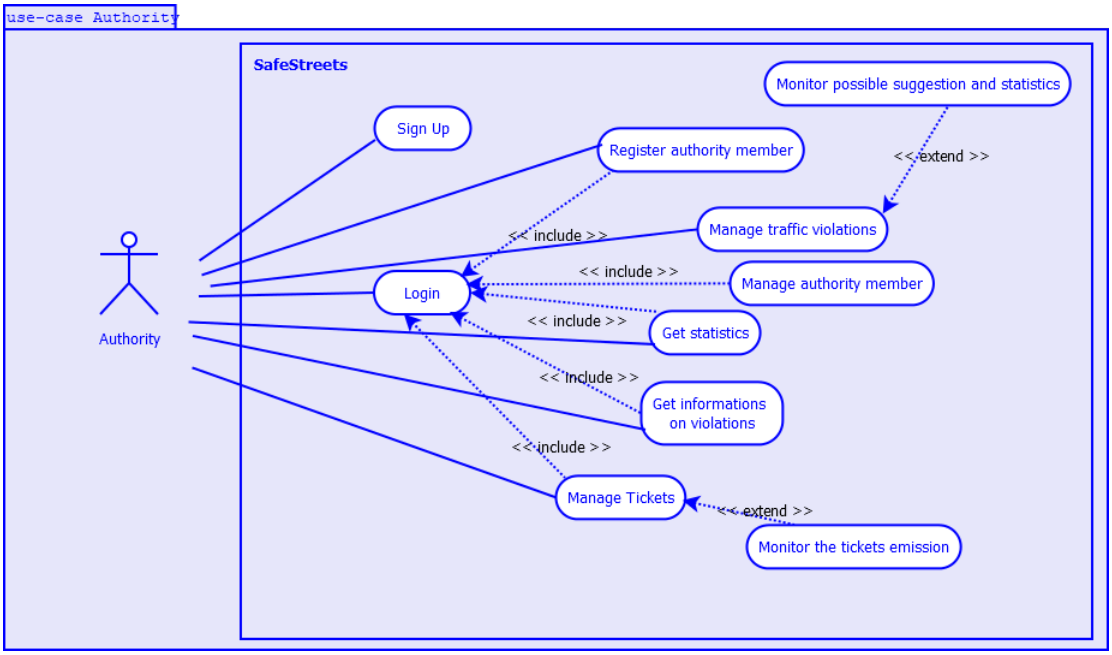

Figure 3.21: Authority use case.

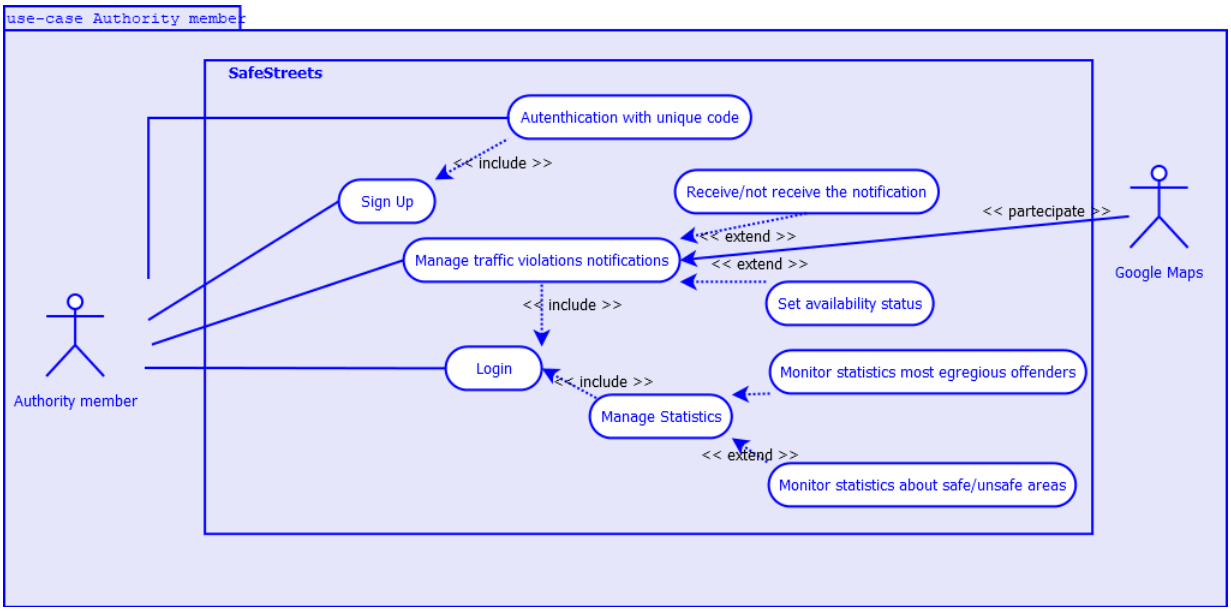

Figure 3.22: Authority Member use case.

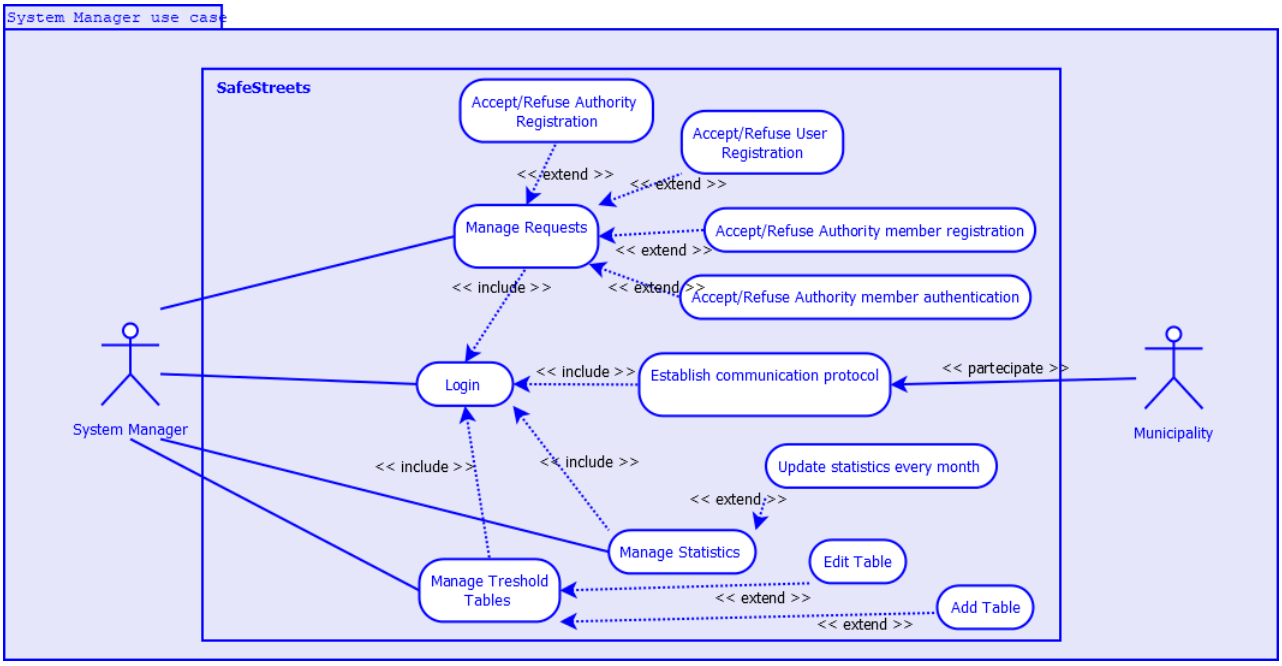

Figure 3.23: System Manager use case.

3.4 Functional Requirements

3.4.1 SafeStreets Functional Requirements

[G1]: Users can be uniquely identified, thanks to the completion of the Registration Process.

[R1] – Citizens must be able to begin the Registration Process into login page.

[R2] – During the Registration Process, Visitor have to be asked from the system for filling with his personal data (name, surname, address, gender, age, email, and fiscal code) a specific form.

[R3] – The system must reject the signup by a Visitor if the provided fiscal code is already associated to another existing account.

[R4] – The system must verify coherence between fiscal code inserted in the registration process form, and personal data provided by the user.

[R5] – The signup must include a completion process in order to verify the correctness of the user's registration and enable the user to access to the software.

[R6] – In order to complete Registration Process, the system will ask for an identity confirmation of the Visitor thanks to a code sent via SMS or e-email.

[G2]: Authorities can be uniquely identified, thanks to the completion of Registration Process.

[R7] – Each authority, belonging to a town in which SafeStreets' services are active, must be registered in order to lets its employee be notified by the users.

[R8] – After the Registration Process, Authority can manage the list of Authority Members adding or delete employees from the list.

[R9] – During the Registration Process, finalized by an Authority member, the system asks for the information about the physical information about the Authority, its formal force name (Police, Carabinieri, Local Police District), the address of reference station and a list of its employee (and for each of them provide name, surname, institutional email and an unique code).

[R10] – System when have the complete employees' list, create an account for each of them.

[R11] – The system must reject the registration by an Authority if the provided couple (formal name, address, city) is already associated to another existing one.

[D1] – Authorities correctly insert address of reference station.

[D2] – Authorities must provide to its employees the unique code.

[G3]: Authority members can be uniquely identified, thanks to the completion of Authentication Process.

[R12] – System must allow authority member to activate their own account providing the unique code that reference Authority Department has assigned to them.

[R13] – The sign up must include a completion process in order to verify the correctness of the user's registration did by the Authority, and enable the Authority Member to access to the software.

[R14] – Software allows authority member to specify their availability status.

[D3] – Authority members specify correctly their availability status.

[D4] – Each authority member must have an uniquely identifiable code.

[G4]: Allows users to notify Authority Members when traffic violations occur.

[R15] – Users must provide their credentials, into the form of login page, to access to their personal area.

[R16] – If credentials do not match with the stored ones, the system must reject the request of login prompting an error.

[R17] – The mobile application must provide a section where the users can fill a form and upload images about the occurred traffic violations. In order to better organize force displacement user specifies to which Authority notify the violation.

[R18] – The mobile application must provide a section where users can find all his past notifications.

[R19] – Allows end users to share the traffic violation's position.

[R20] – Data and time are directly taken from end users' device.

[D5] – Devices used by end users are supposed to have a camera and an integrated and enabled GPS sensor.

[D6] – Sent position is assumed to be reliable and precise.

[D7] – System is supposed to be well integrated with reading plate algorithm that has been already designed and is correctly working.

[D8] – Each already uploaded notification of violation is every time correctly received and stored by the software system.

[G5]: Allows authority member to receive the notifications about traffic violations in order to increase the local security.

[R21] – Authorities member must provide their credentials, into the form

of login page, to access to their personal area.

[R22] – If credentials do not match with the stored ones, the system must reject the request of login prompting an error.

[R23] – Software system dispatches the notification about the incident to the nearest authority member, for which user requested an intervention. If the first authority member notified is busy, the notification will be passed to the next authority member always close to the incident.

[R24] – Software permits to each authority member to specify their availability status.

[R25] – System must be able to recognize license plate from images.

[R26] – System must be able to recognize any possible kind of altered information contained in a traffic violation sent by a user.

[D9] – Authority members specify correctly its availability status.

[D10]– System is supposed to be well integrated with reading plate algorithm that has been already designed and is correctly working.

[D11]– The internet connection works properly without failure.

[D12] – Every time an authority member starts working, has to logs into the application and sets properly availability status.

[D13]– Authority members knows the local traffic laws and the related fines.

[D14]– Authority member that accept to provide an intervention must check the correctness of traffic violations notified and signals to Safe Streets.

[G6]: Allows end users to mine information on traffic violations that has been received and build some statistics.

[R22] – Users must provide their credentials, into the form of login page,

to access their personal view.

[R23] – Software system shows statistics related to unsafe areas thanks to the highest number of violations.

[R24] – System shows to the end users the statistics in a specific section of the software.

[R25] – Statistics must be updated each month.

[G7]: Allows authority members to mine information on traffic violations that has been received, and build some statistics.

[R26] – Software system show which kind of traffic violations occurs more frequently for each area.

[R27] – System shows the related statistics in a specific section of the software.

[R28] – Software system is able to show statistics related to unsafe areas thanks to the highest number of traffic violations in that zone.

[R29] – Statistics must be updated each month.

[R30] – Software system is able to show statistics related to vehicles that commit the most violations.

[R31] – Authority members must provide their credentials, into the form of login page, to access their personal view.

[G8]: Builds a cross information analysis between municipality's data and its self data to improve reliability of the service and suggest to municipality possible interventions.

[R32] – Software system must be able to retrieve information from municipality service and generate their relative statistics.

[R33] – SafeStreets provides an algorithm able to cross information which

derives from its own statistics and municipality's statistics.

[R34] – Permits to suggest to municipality how to improve the security.

[R35] – SafeStreets is able to communicate suggestion to the municipality through e-mail.

[D15] – Municipality service is well integrated with SafeStreets.

[D16] – Municipality has an active mail system and it is periodically checked by its own employee.

[D17] – Municipality can fulfill the improvements suggested by the software.

[G9]: Allows municipality (in particular local police) to retrieve traffic violations in order to generate relative traffic tickets.

[R36] – System has to avoid any possible kind of altered information contained in a traffic violation sent by a user.

[R37] – System must be able to recognize license plate from images.

[R38] – Provides personal data of the vehicle's owner that committed an infraction to authorities, retrieved by an external service(FindOwnerPlate).

[R39] – SafeStreets is able to send all informations related to traffic violations to the nearest local police station which consequently generates traffic tickets.

[R40] – SafeStreets stores position of all local police centers in the city where SafeStreets works.

[D18] – External service (FindOwnerPlate) is well integrated with SafeStreets that permits to retrieve personal data of the vehicle's owner.

[D19] – System is supposed to be well integrated with reading plate algorithm that has been already designed and is correctly working.

[D20] – The authority member knows the local traffic laws and the related fines.

[G10]: Builds statistics using information related to emitted traffic tickets.

[R41] – Provides personal data of the vehicle's owner, who committed an infraction, to authorities retrieved by an external service.

[R42] – SafeStreets is able to store all infractions sent to local police station and generate their relative statistics by mean of an algorithm.

[R43] – SafeStreets provides to local police a ranking of the most offenders in their relative area.

[R44] – SafeStreets provides to users the statistics concerning the improvement brought by SafeStreets initiative.

[D21] – External service (FindOwnerPlate) is well integrated with SafeStreets that permits to retrieve personal data of the vehicle's owner.

3.4.2 Use case descriptions

Actor	Visitors
Goal	G1
Input condition	There are no entry conditions.
Event flow	1. The Visitor, after downloaded the mobile application, comes to the view of the mobile app and clicks on the Sign In button to start Registration Process 2. The Visitor from a first panel choose button labelled "Citizens" 3. The Visitor fills all personal mandatory fields (name, surname, address, gender, birth date, e-mail, phone number and fiscal code). 4. The Visitor clicks the Confirmation button. 5. The system saves and check correctness of provided data. 6. The system asks for the user a favourite communication method (SMS or e-mail) 7. The system sends, accordingly to the user's choice, a confirmation link. 8. The Visitor clicks on the confirmation link to correctly enable his account.
Output condition	The Visitor successfully ends the registration process, becoming a User. Since now, he/she can log into the application with his/her e-mail and password.
Exception	1. The Visitor does not specify at least one mandatory field. 2. The Visitor inserts invalid information in at least one field (data-type not respected, no match between personal information and provided fiscal code, invalid e-mail provided). 3. The Visitor specifies an email which has already been associated with an existing account. 4. The Visitor specifies a fiscal code which has already been associated with an existing User. 5. All exceptions are handled notifying the issue to the Visitor and taking back to the Event Flow to the point 3.

Table 3.1: Registration Process – User (Citizen).

Actor	Visitors
Goal	G2
Input condition	There are no entry conditions.
Event flow	1. The Visitor, after downloaded the mobile application, comes to the view of the mobile app and clicks on the Sign In button to start Registration Process 2. The Visitor from a first panel choose button labelled "Authority District" 3. The Visitor fills mandatory fields: formal force name (Polizia, Carabinieri, Local Police District), reference address for that station, city name and an institutional mail address. 4. The Visitor upload a list of its employee (for which is needed to specify name, surname, institutional e-mail and a unique code) that match required file extension. 5. The Visitor clicks the Confirmation button. 6. The system saves and check correctness of provided data. 7. The system, in an automatic way, create an account for each Authority Member belonging to that District. 8. The system sends a mail containing a confirmation link and a randomly generated password. 9. The Visitor clicks on the confirmation link to correctly enable his account.
Output condition	The Visitor successfully ends the registration process, becoming an Authority. Since now, an employee assigned to this role for that District, can log into the application using institutional district's e-mail and password.
Exception	1. The Visitor does not specify at least one mandatory field. 2. The Visitor inserts invalid information in at least one field. 3. The Visitor specifies a triple (formal name, address, city) which has already been associated with an existing District. 4. The Visitor specifies an e-mail address already associated to an existing District. 5. The Visitor do not upload any attachment related to employees list. 6. The Visitor upload a file with not allowed extension. 7. Exceptions 1-4 are handled notifying the issue to the Visitor and taking back to the Event Flow to the point 3. 8. Exceptions 5-6 are handled notifying the issue to the Visitor and taking back to the Event Flow to the point 4.

Table 3.2: Registration Process – Authorities.

Actor	Authority Members
Goal	G3
Input condition	There are no entry conditions.
Event flow	1. The Visitor, after downloaded the mobile application, comes to the view of the mobile app and clicks on the Sign In button to start Registration Process 2. The Visitor from a first panel choose button labelled "Authority Member" 3. The Visitor after specifying information about District of belonging, insert the unique code provided to him/her. 4. The system retrieves personal data associated to that code and show them to the Visitor. 5. The Visitor check data correctness (in case something is wrong modifies identified fields) and then clicks the Confirmation button. 6. The system sends a mail containing a confirmation link and a randomly generated password. 7. The Visitor clicks on the confirmation link to correctly enable his account.
Output condition	The Visitor successfully ends the activating account process, becoming an Authority Member. Since now, he/she can log into the application with his/her institutional e-mail and password.
Exception	1. The Visitor inserts a wrong unique code, or one not associated to specified District of belonging 2. The Visitor whose choose to modify some field inserts invalid information in at least one of them (data type not respected, invalid e-mail provided). 3. Exception 1 is handled notifying the issue to the Visitor and taking back to the Event Flow to the point 3. 4. Exception 2 is handled notifying the issue to the Visitor and taking back to the Event Flow to the point 5.

Table 3.3: Activating Account Process – Authority Members.

Actor	Users, Authorities and Authority Members
Goal	G1, G2, G3
Input condition	The Customer in play has correctly performed the Sign In process, accordingly to the rules specified for its role in the application.
Event flow	1. The Customer comes to the view of the mobile app and clicks on the Log In button. 2. The Customer insert its credential: (institutional) e-mail and password. 3. The system check correctness of inserted data. 4. The system allows to the Customer to access to its personal profile.
Output condition	The Customer successfully log into the application and can now perform all possible action accordingly to the role of the Customer in the application.
Exception	1. The Customer does not specify at least one mandatory field. 2. The Customer inserts invalid information in at least one field. 3. All Exception are handled notifying the issue to the Customer and taking back to the Event Flow to the point 2.

Table 3.4: Login Process – All.

Actor	User
Goal	G4
Input condition	User has correctly performed the login operation.
Event flow	1. The User comes to the view of the mobile app and clicks on specific panel to start upload a violation's report. 2. Only first time, the User allows to SafeStreets usage of camera and GPS sensor. 3. The User fills a form about the occurred traffic violations. Required fields are type and image of violation, where violation occur and to which Authority notify the violation. 4. The User can optionally type additional notes to improve accuracy of report. 5. The User clicks the Confirmation button. 6. The system runs the Reading Plate algorithm and adds as metadata, the result of this elaboration. 7. The system saves and check correctness of provided data, recognizing possible altered information contained in the traffic violation's report. 8. The system performs some elaboration through designed algorithm in order to extract essential information from the report. 9. The system makes result of that elaboration part of the statistics. 10. The system displays an image that let the User know process is correctly ended.
Output condition	The User successfully notified Authorities for the violation. Since now, he/she can follow the progress of this and also past notifications handling into the specific section of SafeStreets' application.
Exception	1. The User does not specify at least one mandatory field. 2. The User inserts invalid information in at least one field. 3. The User do not allow requested authorization. 4. The system, after run Reading Plate algorithm, cannot find a clear match with provided image. 5. The system recognizes in the user report an attempt to data alteration. 6. All exceptions are handled notifying the issue to the User and taking back to the Event Flow to the point 3.

Table 3.5: Notify Authority – User.

Actor	Authority Member
Goal	G5
Input condition	- Authority Member has correctly performed the login operation. - Authority Member has set as "Available" his/her availability status. - Authority Member, whose is send the request of intervention, belongs to the specified by User Authority to notify, and is the closest, available member for handling the violation.
Event flow	1. The Authority Member is notified by SafeStreets about the violation. 2. The Authority Member start handling the request and goes to the place where violation occurs. 3. The system updates the history of that notification as "Taken in charge". 4. The system automatically set as "Unavailable" Authority Member status. 5. The Authority Member once there, verifies the truthfulness of the violation and apply proper sanctions accordingly with laws in force for that specific town. 6. The Authority Member fills on the application a police report. 7. The system performs some elaboration through designed algorithm in order to extract essential information from the report. 8. The system makes result of that elaboration part of the statistics. 9. The system updates the history of that notification as "Handled". 10. The system attaches to User's notification history the police report filled by Authority Member that has handled the request of intervention. 11. The system set again as "Available" Authority Member status.
Output condition	The Authority Member correctly handled the notified violation. Since now, he/she can check his/her own police report related to that violation into the specific section of SafeStreets' application.
Exception	1. The Authority Member once on the place of traffic offence isn't able to verify the truthfulness of the violation. 2. The Authority Member don't see notification of SafeStreets application. 3. Exception 1 is handled by the system, updating the history of that notification as "No violation observed", aborting the Event Flow from point 7 on. 4. Exception 2 is handled by the system, notifying again the Authority Member and taking back to the Event Flow to the point 2

Table 3.6: Authority Member handle violation report.

Actor	Authority Members
Goal	G7
Input condition	Authority Members has correctly performed the login operation.
Event flow	1. The Authority Members comes to the view of the mobile app and clicks on specific panel to visualize Statistics. 2. The system retrieves from storage Statistics, accordingly with level of visibility of this Customer and with his/her town of belonging. 3. The Authority Members visualize requested Statistics. 4. The Authority Members can add some filters to the search (at this time those include a more restrictive search on a specific Area, lists the Top 10 unsafe area of the city, the unruliest vehicle drivers). 5. The system retrieves updated Statistics that respects the constraints expressed. 6. The Authority Members visualize requested, updated Statistics.
Output condition	The Authority Members is able to visualize Statistics about unsafe area of its town.
Exception	1. The Authority Members, whose choose to express a filter, do not specify at least one mandatory field. 2. The Authority Members, whose choose to express a filter, inserts invalid information in at least one field. 3. The system in retrieving Statistics for the filters added by the Authority Members doesn't find any match in the storage. 4. All exceptions are handled notifying the issue to the Authority Members and taking back to the Event Flow to the point 1.

Table 3.8: Authority Members requests to visualize Statistics.

Actor	User
Goal	G4
Input condition	User has correctly performed the login operation.
Event flow	1. The User comes to the view of the mobile app and clicks on specific panel about his/her past Notification. 2. The system retrieves from storage Notification which sender is the User whose is performing the request. 3. The User visualize his/her past Notification list. 4. The User tap on an element of the list for which want to see more details. 5. The system retrieves from storage all details and attachment for that Notification. 6. The User visualize details about that Notification and if already handled can read the police report attached.
Output condition	The User is able to visualize progress of an his/her Past Notification.
Exception	1. The User hasn't already performed any notification to the Authorities. 2. Exception 1 is handled showing an empty list and aborting the Event Flow form the point 3 on.

Table 3.9: User requests to visualize progress of their past notifications.

Actor	Authority Members
Goal	G5
Input condition	Authority Members has correctly performed the login operation.
Event flow	1. The Authority Members comes to the view of the mobile app and clicks on specific panel about Notification that he/she has handled. 2. The system retrieves from storage those Notifications. 3. The Authority Members visualize his/her past Notification list. 4. The Authority Members tap on an element of the list for which want to see more details. 5. The system retrieves from storage all details and attachment for that Notification. 6. The Authority Member visualizes details about that Notification and can read police report he/she wrote.
Output condition	The Authority Member is able to visualize his/her Past handled Notification.
Exception	1. The Authority Member hasn't already handled any notification. 2. Exception 1 is handled showing an empty list and aborting the Event Flow form the point 3 on.

Table 3.10: Authority Members requests to visualize their past handled notification.

Actor	Municipality
Goal	G8
Input condition	Municipality offers a service that allows users to retrieve the information about the accidents that occur on the territory.
Event flow	1. Municipality sends its collected information to SafeStreets. 2. The system makes a cross information between its data and municipality's data. 3. The system analyses the generated data, and, for each critical zone, selects a pre-set suggestion. 4. SafeStreets sends selected suggestion to Municipality through the communication protocol.
Output condition	Municipality receives suggestions made by the System.
Exception	1. No communication protocol is found. 2. Municipality information retrieved by the system are not well-encoded. 3. The system does not find a suggestion for a critical zone. 4. Exception 2 is handled by the system, notifying to municipality the correct encoding of information. 5. Exception 3 is handled by the system, sending a notification to system administrator trying to add other pre-set suggestion.

Table 3.11: SafeStreets suggests same possible intervention to the Authority.

12. SafeStreets sends to Authority (local Police) certified violations for which is possible to generate traffic tickets

Actor	Authority (local Police)
Goal	G9
Input condition	Authority has correctly performed the Sign In process, accordingly to the rules specified for its role in the application. Authority Member, which has *ForceName = "Local Police"*, has certified that a notification sent by a user is effectively a violation.
Event flow	1. The system receives the intention expressed by AM to proceed to the creation of a ticket. 2. The system, taken the plate read in previous phases of the violation handling process, run *findOwnerPlate* external algorithm. 3. *findOwnerPlate* returns the data of the person whose committed the infraction. 4. The system updates the statistics regarding traffic tickets. 5. The system attaches all needed information and send a notification to let the Authority (which has the AM whose certified the violation in its employee list) compile the ticket. 6. Authority receives the notification and is ready to handle the ticket generation process.
Output condition	The Authority correctly receives SafeStreets notification and has all information in order to start ticket generation process.
Exception	1. *findOwnerPlate* is unable to find a match with the plate provided but the system is unable to perform any action to avoid this. 2. The Authority don't see notification of SafeStreets application. 3. Exception 1 is handled by the system, sending a form filled without offender's data. Could be the case of a compromised plate. 4. Exception 2 is handled by the system, notifying again the Authority and taking back to the Event Flow to the point 5

Table 3.12: Local police receives certified violations for which is possible to generate traffic tickets .

13. Authority (local Police) requests to visualize Statistics about Traffic Tickets

Actor	Authority (local Police)
Goal	G10
Input condition	Authority has correctly performed the login operation.
Event flow	1. Authority accesses to traffic tickets statistic by mean of main page. 2. The system provides traffic tickets statistic related to the belonging zone of authority member.
Output condition	Local police authority visualizes statistics about traffic tickets in its area.
Exception	1. No traffic tickets generated in the zone of action of the Authority. 2. Exception 1 is handled by the system, notifying that no traffic tickets are occurred.

Table 3.13: Local police visualizes statistics regarding traffic tickets.

3.4.3 Sequence Diagrams

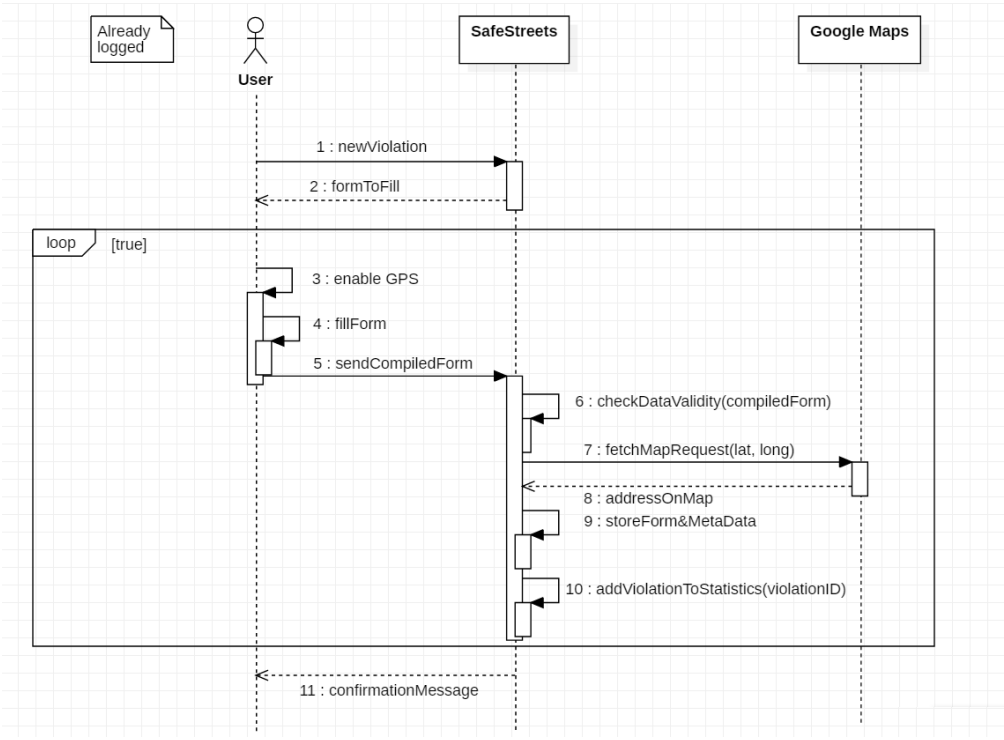

Figure: 3.24 User sends notification sequence diagram.

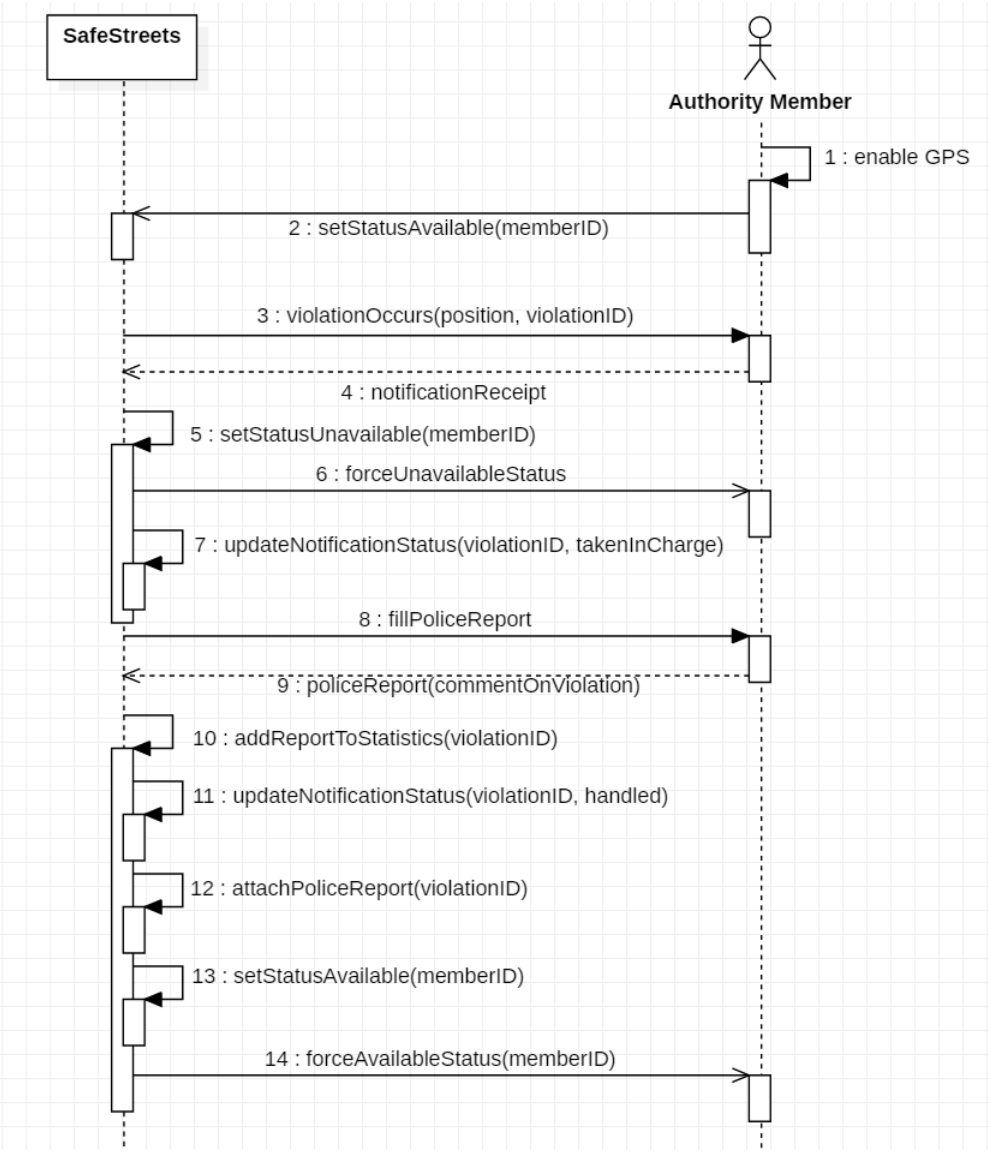

Figure: 3.25 AM receives notification sequence diagram.

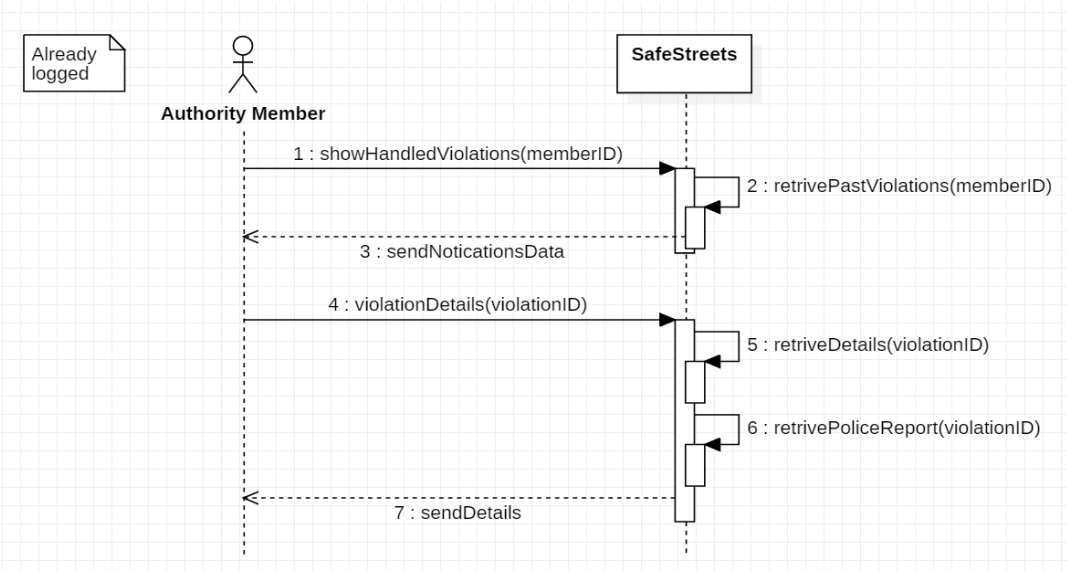

Figure: 3.26 AM visualizes past notifications sequence diagram.

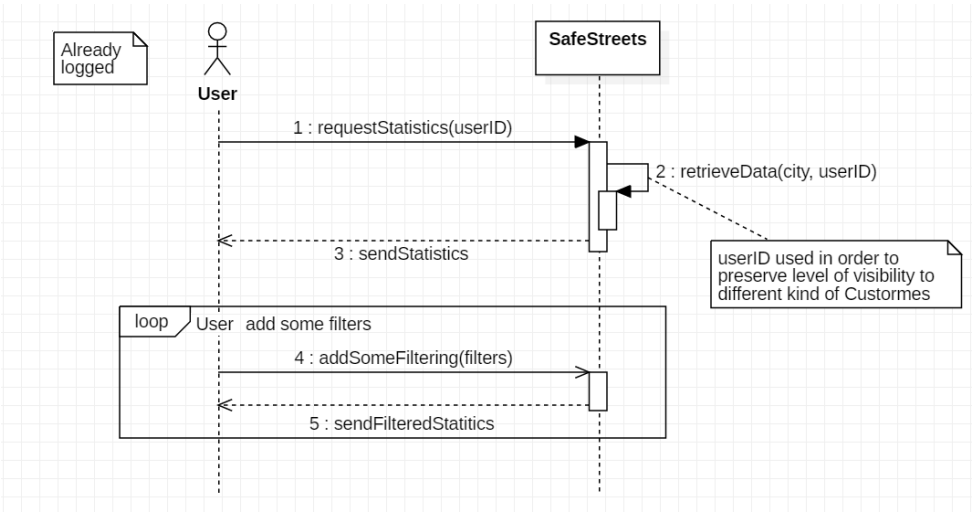

Figure: 3.27 User requests for statistics sequence diagram.

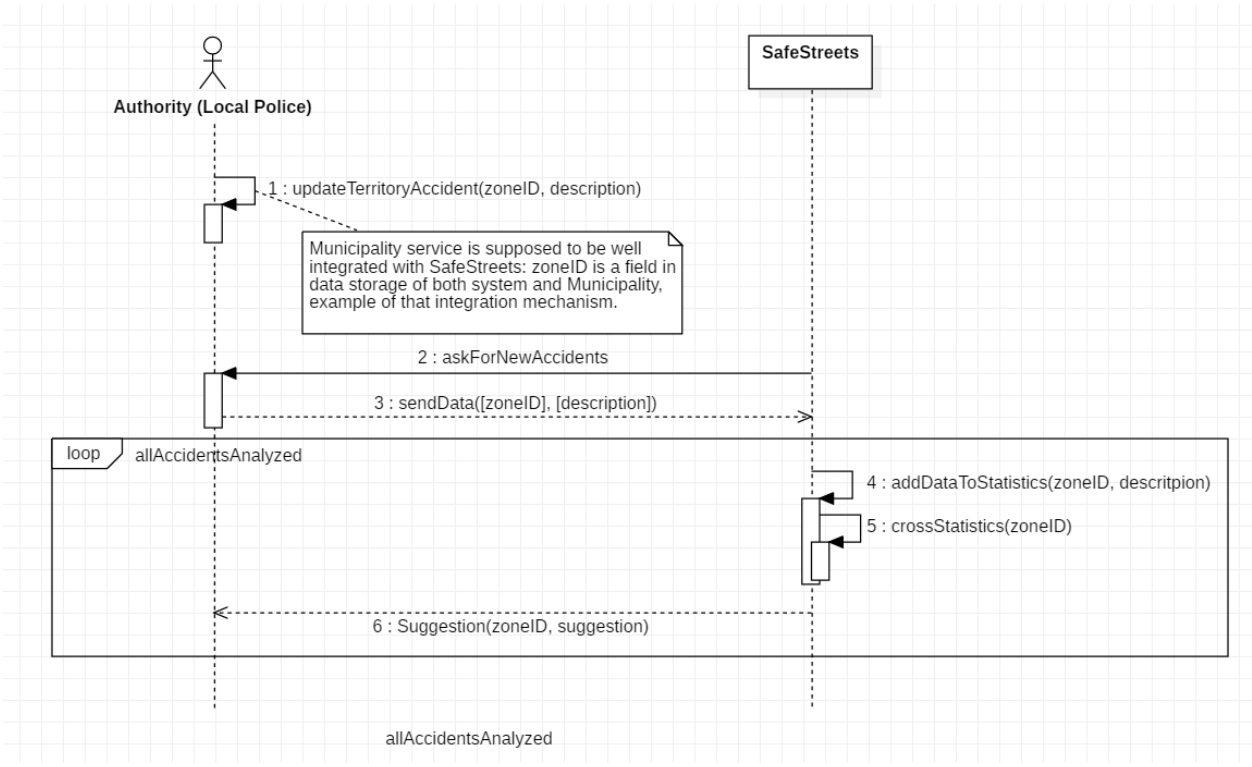

Figure: 3.28 System suggests intervention sequence diagram.

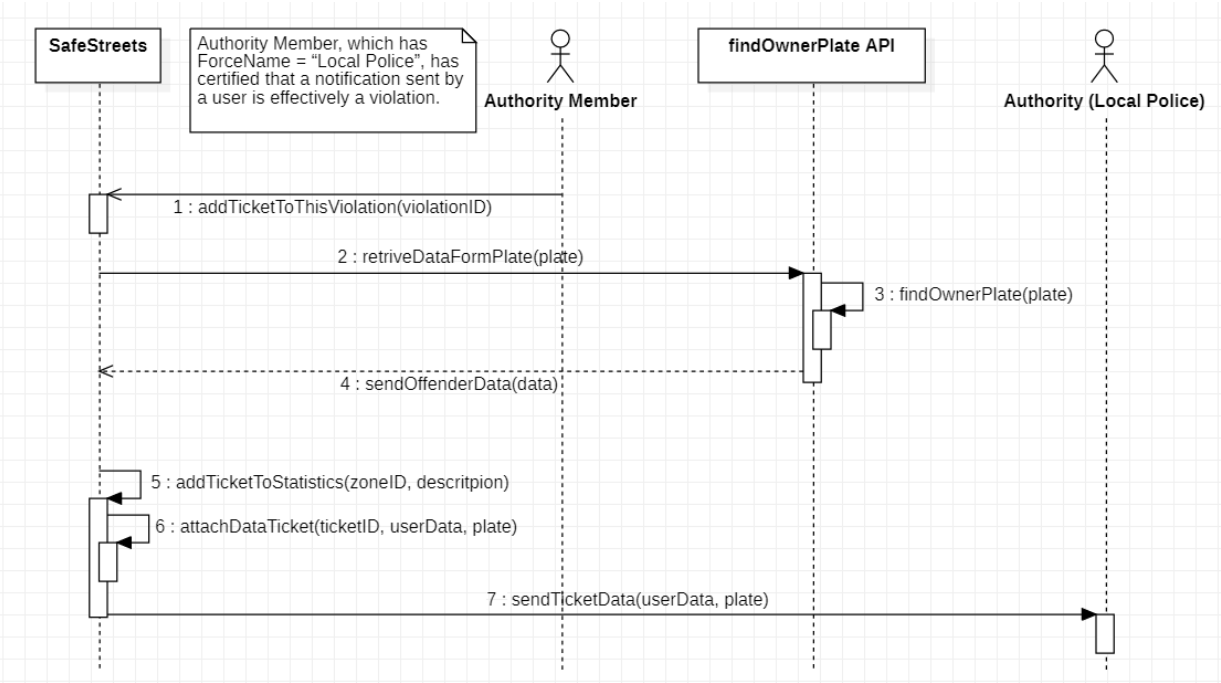

Figure: 3.29 Ticket generation process. sequence diagram.

3.5 Performance Requirements

[P1] – The system is designed to manage in an efficiently and effectively way the requests of a little-medium size city of about four hundred thousand population. In a first estimation, an average 75% of contemporary users activity is expected.

[P2] – The system is designed to guarantee a response time less than 10 seconds between an user notification about a traffic violation and the succesful receipt and dispatching occurred between the agents closer to the streets' violation.

[P3] – Starting from the System Manager, the system must guarantee the correcting update of every statics every month with a response time less than 30 seconds.

[P4] – Starting from the User notification to authorities, the system must guarantee a response time of less than 4 seconds for computing the accepting phase by the authorities member.

[P5] – The system must guarantee an updating time of picture and infos in the form fill by the User about less than 3 seconds from the time the User updated the parameters and files.

3.6 Design Constraints

3.6.1 Hardware limitations

- For the mobile app, an android 5+ or ios 10.1+ smartphone must be used.
- One among WI-FI, 3G, 4G, 4G+ connection should be available in order to garantee the lowest latency.

- A GPS sensor must be available on smartphone, and correctly face the SafeStreets data needs.

3.7 Software System Attribute

3.7.1 Reliability

The system must guarantee 24/7 service with a failure date of less than 0.1%, since its failure will almost certainly cause losing of information about traffic violations and disease in the use of the application.

3.7.2 Availability

Seeing the reliability constraints, the system is going to have almost a 100% availability rate for SafeStreets.

3.7.3 Security

An high security level is a mandatory constraint for a system like this. This constraint is implemented thanks to a hardened database technlogy to face with the data storing managament problem. Talking about data transmission mechanism, every communication between mobile or web interface and the server will be ciphered with AES protocols, and secured with HTTPS/SSL protocols.

3.7.4 Maintainability

A modular approach is one of the serveral key values which we had chosen to identify for the system. The system is designed to work with the aim of extending the available services (aSOS and T4R, for instance,are internally devised for being modular).

3.7.5 Portability

Becoming aware of the critically of the system and its application, it is obviously that no much effort will be given to the portability attribute, since the concepts of ”adaptability” and ”security and reliability” are inversely proportional.

4 Formal analysis using Alloy

4.1 Alloy model

```
module SafeStreets

//SIGNATURES

abstract sig Gender{}
one sig Male extends Gender{}
one sig Female extends Gender{}

abstract sig AvailabilityStatus{}
one sig Available extends AvailabilityStatus{}
one sig Unavailable extends AvailabilityStatus{}

abstract sig Customer {
email: one Email,
password: one Password,
}

sig FiscalCode{}
sig Email{}
sig Password{}
sig UniqueCode{}
sig City{}

sig User extends Customer {
name: one String,
surname: one String,
address: one String,
gender: one Gender,
age: one Int,
city: one City,
fiscalcode: one FiscalCode,
position: one Position,
notification: set Notification,
}{
#notification >= 0
age >= 18 and age <= 90
}

sig Position{
coorX: one Int,
coorY: one Int,
}

```

```
sig Notification {
notificationID: one Int,
typeOfViolation: one String,
date: one Int,
time: one Int,
position: one Position,
referenceAuthorityName: one String,
description: one String,
historyStatus: one String, //Used to take in care the progress in handling notification
user: one User,
}{
date >=0
time >=0
}

sig Authority extends Customer {
forcename: one String,
referenceaddress: one String,
city: one City,
authoritymember: set AuthorityMember,
}

sig AuthorityMember extends Customer {
name: one String,
surname: one String,
uniqueCode: one UniqueCode,
availabilityStatus: one AvailabilityStatus,
position: one Position,
authority: one Authority,
}

sig Violation extends Notification {
violationID: one Int,
readPlate: one String,
authorityMember: one AuthorityMember,
}

sig PastViolation extends Notification {
violationID: one Int,
readPlate: one String,
authorityMember: one AuthorityMember,
report: one String,
nameZone: one String,
```

```
}

sig UserStatistics{
violation: set PastViolation,
}

sig AuthorityStatistics{
violation: set PastViolation,
}

sig Suggestion{
userstatistics: set UserStatistics,
authoritystatistics: set AuthorityStatistics,
}

sig Ticket{
ticketID: one String,
violation: one PastViolation,
sanction: one String,
}

sig TicketStatistics{
ticketstat: set Ticket,
}

//CUSTOMER FACT
//*****************************************
fact numberOfCustomer{
    Authority + AuthorityMember + User = Customer
}

fact noEmailWithoutCustomer {
#Email=#Customer
}

fact noPassWithoutCustomer {
#Password=#Customer
}

fact customerEmailIsUnique{
no disjoint c1,c2: Customer | c1.email = c2.email or c1.password = c2.password
}

```

```
130
131  //USER FACT
132  //*****************************************
133  fact allCityAssociatedToUser{
134  all c:City| some u1:User | c in u1.city
135  }
136
137  fact userFiscalCodeIsUnique{
138  no disjoint u1,u2: User | u1.fiscalcode= u2.fiscalcode
139  }
140
141  fact noFiscalCodeWithoutUser{
142  all fc:FiscalCode| some u:User | fc in u.fiscalcode
143  }
144
145  fact userHasAGender {
146  #Gender=#User
147  }
148
149  fact noUserGender{
150  all g:Gender| some u:User | g in u.gender
151  }
152
153  fact allPositionAssociatedToUser{
154  all p:Position| some u1:User | p in u1.position
155  }
156
157
158  //NOTIFICATION FACT
159  //*****************************************
160  fact notificationIdIsUnique{
161  no disjoint n1, n2: Notification | n1.notificationID=n2.notificationID
162  }
163
164
165  //AUTHORITY FACT
166  //*****************************************
167  fact allCityAssociatedToAuthority{
168  all c:City| some a:Authority | c in a.city
169  }
170
171  // triple (city, address, force name) unique
172  fact uniqueAuthorityInformation {
```

```
no disjoint a1, a2: Authority | a1.city = a2.city and
                                a1.referenceaddress = a2.referenceaddress and
                                a1.forcename = a2.forcename
}

//each member linked to an authority must be in the set of members of that authority
fact memberBelongingToAuthorityMemberSet{
    all am:AuthorityMember, a:Authority | am in a.authoritymember => am.authority = a
}

//check if an authority member is inside the set of authority
assert checkingOnBelongingAuthority{
    some am:AuthorityMember, a:Authority | am.authority= a and am not in a.authoritymember
}


//AUTHORITY MEMBER FACT
//*****************************************

//Code are unique inside each authority
fact authorityCodeIsUnique{
no disjoint am1,am2: AuthorityMember | am1.uniqueCode = am2.uniqueCode
                                       and am1.authority = am2.authority
}

fact noUniqueCodeWithoutAuthority{
all uc:UniqueCode| some am:AuthorityMember | uc in am.uniqueCode
}

fact memberAvailability{
all av:AvailabilityStatus| some am:AuthorityMember | av in am.availabilityStatus
}

//VIOLATION FACT
//*****************************************

fact violatioIDisUnique {
no disjoint v1, v2: Violation | v1.violationID=v2.violationID
}

fact violationHandlerBySpecifiedAuthority{
all n:Notification, v:Violation | v.notificationID = n.notificationID
                                  => v.authorityMember.authority.forcename = n.referenceAuthorityName
```

```
}

//Violation is handled by authority member which has the same city of the user
fact eachViolationHasMemberAndUserSameCity {
all v: Violation | v.user.city=v.authorityMember.authority.city
}

//if an authority member is handling a violation then its status is set as unavailable

fact forcingUnavailableStatusToMember{
all am:AuthorityMember, v:Violation | v.authorityMember=am => am.availabilityStatus=Unavailable
}

fact MemberHandleOneViolationaTime{
no disjoint v1,v2:Violation | v1.authorityMember = v2.authorityMember
}


//PAST-VIOLATION FACT
//*****************************************

fact managingStatusToAuthorityMember {
all n: Notification, v: Violation, pv: PastViolation | (n.notificationID = v.notificationID
and n.notificationID != pv.notificationID)
=> v.authorityMember.availabilityStatus= Available
or( (n.notificationID = pv.notificationID and n.notificationID != v.notificationID)
or ( n.notificationID != v.notificationID and  n.notificationID != pv.notificationID))
=> v.authorityMember.availabilityStatus= Unavailable
}

//STATISTICS FACT
//*****************************************

fact userStatisticRelatedToPastViolation{
all s:UserStatistics | some pv:PastViolation | pv in s.violation
}

fact authorityStatisticRelatedToPastViolation{
all s:AuthorityStatistics | some pv:PastViolation | pv in s.violation
}

fact authorityStatisticRelatedToPastViolation{
all sugg:Suggestion | some as1:AuthorityStatistics | as1 in sugg.authoritystatistics
```

```
all sugg:Suggestion | some us1:UserStatistics | us1 in sugg.userstatistics
}


//TICKET FACT
//*****************************************

fact ticketIdIsUnique {
no disjoint t1, t2: Ticket | t1.ticketID=t2.ticketID
}

fact ticketRelatedToOneViolation{
all t:Ticket | one pv:PastViolation | pv in t.violation
}

//TICKET STATISTIC FACT
//*****************************************

fact ticketStatsRelatedToTicket{
all ts:TicketStatistics | some t:Ticket | t in ts.ticketstat
}

//PREDICATE

pred userSendNotification [u: User, a:Authority] {
all n:Notification | n.user = u and n.referenceAuthorityName = a.forcename
}


pred memberHandleNotification[u: User, a: Authority, am: AuthorityMember] {
all v: Violation | v.user = u and v.referenceAuthorityName = a.forcename and v.authorityMember = am
}


pred ticketGeneratedForViolation[u:User, a:Authority, am:AuthorityMember,v:Violation]{
all t:Ticket | v.user = u and v.referenceAuthorityName = a.forcename and v.authorityMember = am
        and t.violation= v
}


pred statisticsGenerate[u:User, a:Authority, am:AuthorityMember,t:Ticket]{
all ts:TicketStatistics | v.user = u and v.referenceAuthorityName = a.forcename and v.authorityMember = am
        and ts.ticket= t
```

```
302  }
303
304  pred show {
305  #User > 0
306  #AuthorityMember >0
307  #Authority > 0
308  #Notification > 0
309  #Violation > 0
310  #PastViolation > 0
311  #Ticket > 0
312  }
313
314  run userSendNotification for 6 but 8 Int, exactly 6 String
315  run memberHandleNotification for 6 but 8 Int, exactly 6 String
316  run ticketGeneratedForViolation for 6 but 8 Int, exactly 6 String
317  run statisticsGenerated for 6 but 8 Int, exactly 6 String
318  run show for 6 but 8 Int, exactly 6 String
```

4.2 World Generated

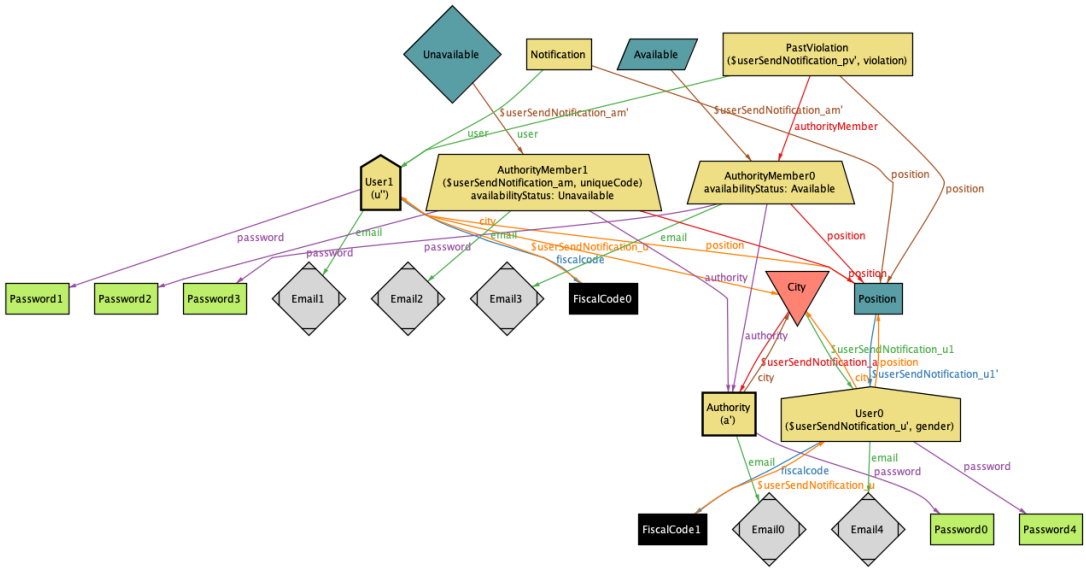

Figure: 4.1 World generated 1: User sends a notification.

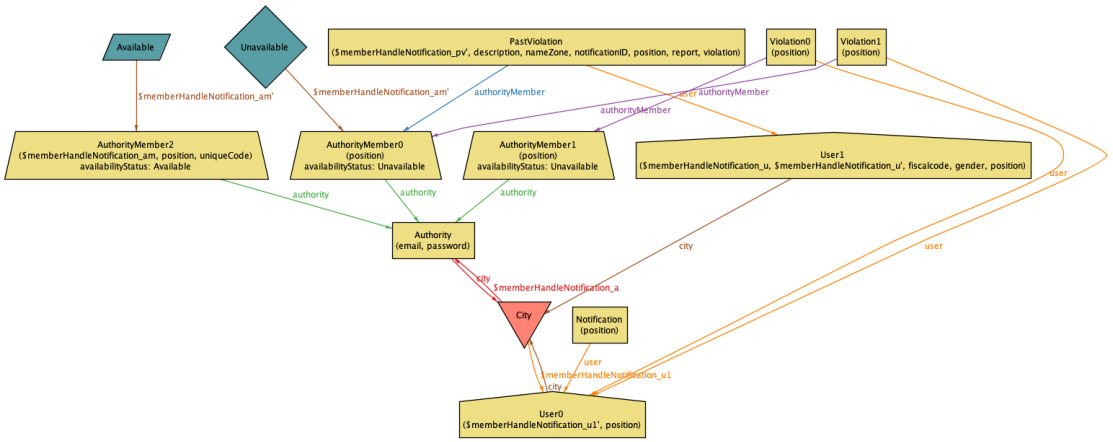

Figure: 4.2 World generated 2: AM handles a violation.

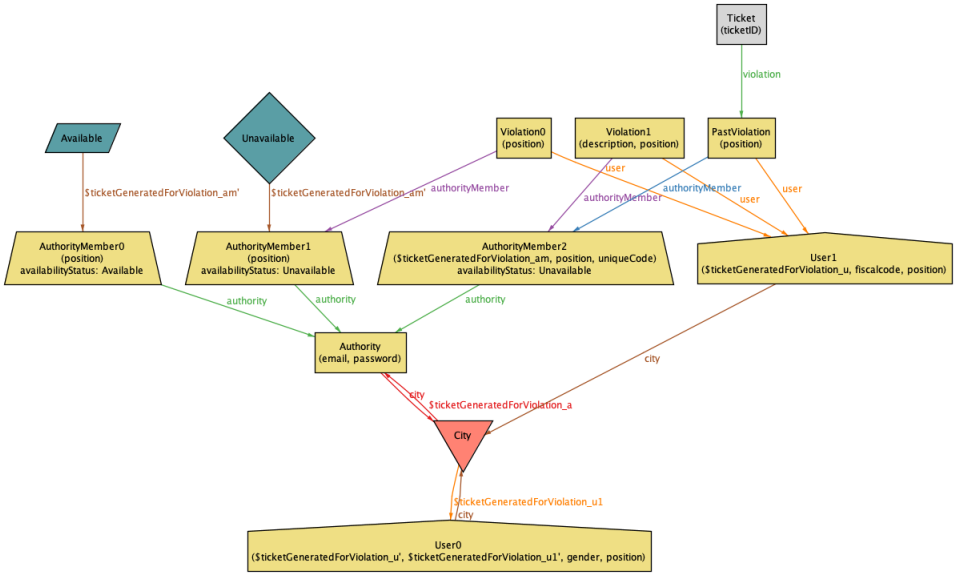

Figure: 4.3 World generated 3: Generation of ticket from violation.

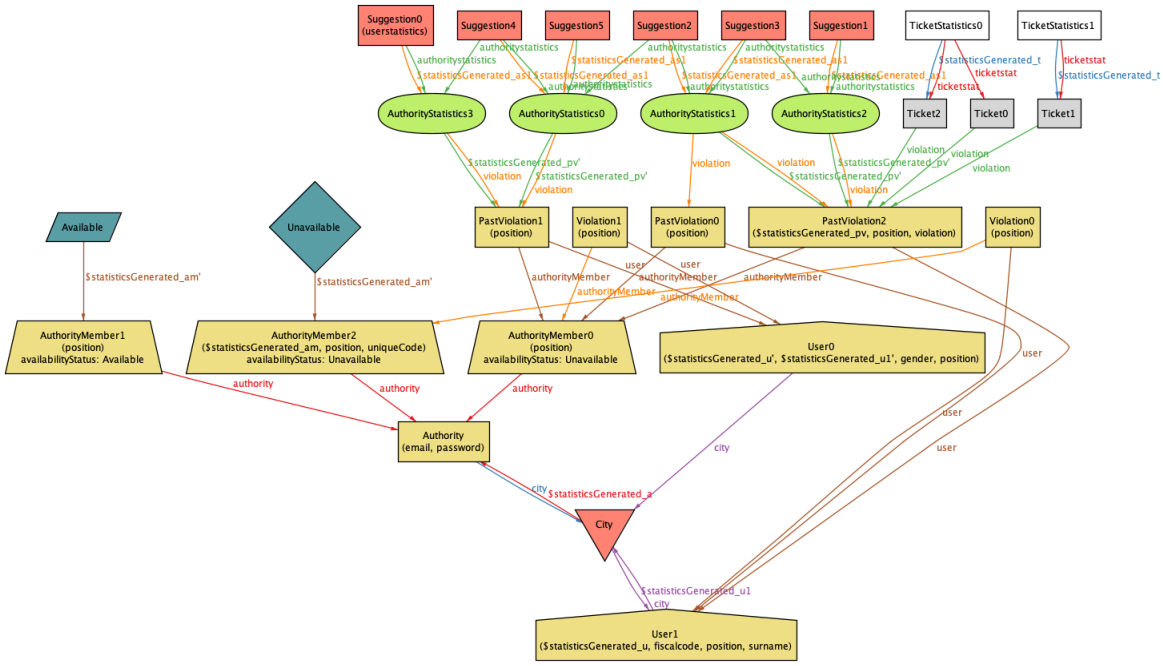

Figure: 4.4 World generated 4: Some statistics being generated in the World.

4.3 Alloy Results

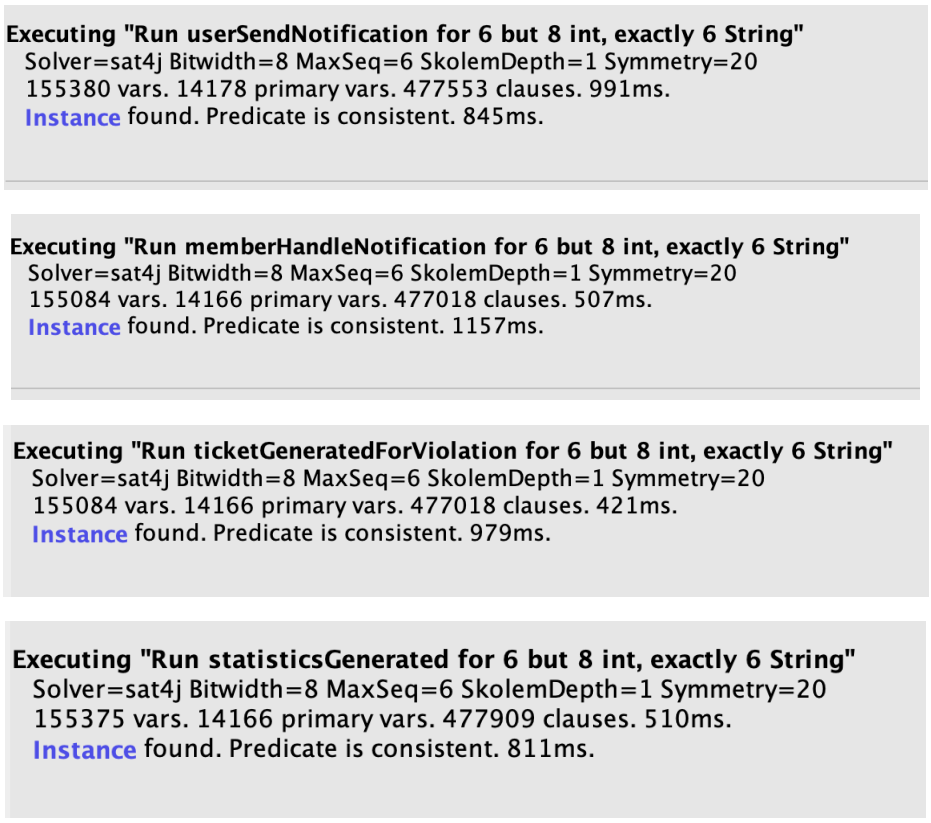

Figure: 4.5 Alloy Results.

Effort spent

Armenante Valerio	60 hours
Capaldo Marco	60 hours
Di Salvo Dario	60 hours