POLITECNICO
MILANO 1863

Davide Perugini, Antonio Pipita, Stefano Panzeri

SafeStreets

Requirement Analysis and Specification Document
Software Engineering 2 Project

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Davide Perugini, Antonio Pipita, Stefano Panzeri
Version:	1.1
Date:	15-December-2019
Download page:	https://github.com/fafrullo2/PanzeriPeruginiPipita
Copyright:	

Contents

Table of Contents **3**

List of Figures **5**

1 Introduction **6**
1.1 Purpose 6
1.1.1 Goals 6
1.2 Scope 7
1.3 Definitions, acronyms and 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 8
1.6 Document structure 9

2 Overall Description **10**
2.1 Product Perspective 10
2.1.1 Class Diagram 11
2.1.2 Statecharts 12
2.2 Product Functions 13
2.3 User Characteristics 14
2.4 Assumptions, dependencies and constraints 14

3 Specific Requirements **15**
3.1 External interface requirements 15
3.1.1 User Intefaces 15
3.1.2 Hardware Interfaces 20
3.1.3 Software Interfaces 20
3.1.4 Communication Interfaces 20
3.2 Functional Requirements 21
3.2.1 Scenarios 23
3.2.2 Use Case Description 24
3.2.3 Use Case Diagram 28
3.2.4 Sequence Diagrams 30
3.2.5 Mapping on Requirements 33
3.3 Performance Requirements 34
3.3.1 Response time 34
3.3.2 Workload management 34
3.4 Design Constraints 34
3.4.1 Standard Compliance 34
3.4.2 Hardware Limitations 35
3.5 Software System Attributes 35
3.5.1 Reliability 35
3.5.2 Avaibility 35
3.5.3 Security 35
3.5.4 Mantainability 35
3.5.5 Portability 36

4 Formal Analysis Using Alloy 37
4.1 Introduction 37
4.2 Signatures 38
4.3 Facts 39
4.4 Assertions and world 40
4.5 Results 41
4.5.1 Checks on assertions 41
4.5.2 World generated 42

5 Effort Spent 43

List of Figures

1 Class diagram . . . 11
2 Sending a report . . . 12
3 System suggesting possible interventions . . . 12
4 Local police officer taking care of reports . . . 13
5 Mockup of the login page . . . 15
6 Mockup of the form for submitting a report . . . 16
7 Mockup of the camera interface . . . 17
8 Mockup of Unsafe areas page . . . 17
9 Mockup of part of the police client . . . 18
10 Mockup of the municipal authority dedicated webpage . . . 19
11 Regular user . . . 28
12 Municipal Authority . . . 28
13 Policeman . . . 29
14 Violation Report submission . . . 30
15 Police intervention . . . 31
16 Suggested Intervertion retrival . . . 32
17 Alloy model signatures . . . 38
18 Alloy model facts . . . 39
19 Alloy model assertions and world declaration . . . 40
20 Results of the checks on the assertions . . . 41
21 Example of the world described by the model . . . 42

1 Introduction

1.1 Purpose

The purpose of this project is to study the requirements and provide a specification about SafeStreets, a crowd-sourced application that permits users to notify authorities about traffic violations.
This document represents the requirements analysis and specification document of SafeStreets, where purposes, goals, requirements and assumptions of the applications will be defined to provide a support for the stakeholders.

SafeStreets allows users to send pictures of traffic violations with every useful metadata about it (date, time, position, type of violation, etc. . .) to authorities.
In addition, the application provides users and authorities with tools to mine the data collected, for example highlighting the streets where parking violations happen frequently, or the most unsafe areas.
SafeStreets also has the possibility to cross its own data with information about accidents coming from the municipality (if the municipality offers this information as an open service) to indentify unsafe areas with more precision and suggest possible interventions.

Finally, if the local police offers a service that takes data and pictures from SafeStreets to generate traffic tickets, the application must ensure the veracity of the information and use data about issued tickets to build statistics (effectiveness of the service, most dangerous vehicles etc. . .).

1.1.1 Goals

- (G1) Allow users to report traffic violations.
- (G2) Allow users to include pictures in violation reports.
- (G3) The system must be able to retrieve geographical position and time of a violation and add them to report.
- (G4) The system must recognise (from pictures) license plates.
- (G5) The system must store user made reports.
- (G6) Allow users and authorities to mine information collected by the application.
- (GA1.1) The system must be able to cross the data collected with information about the accidents coming from the municipality.
- (GA1.2) The system must be able to suggest possible interventions to decrease the risk of violations in unsafe areas.
- (GA2.1) Allow the local police to retrieve data about violations to generate traffic tickets.

- (GA2.2) The system must be able to ensure the veracity of the information sent by the users.
- (GA2.3) The system must track wether the police has taken care of a certain violation.
- (GA2.4) A traffic ticket must involve the same license plate that the report refers to.
- (GA2.5) The system must be able to build statistics from the information about issued tickets.

1.2 Scope

In a metropolitan city like Milan, one of the biggest problems is the overflowing stream of vehicles in the streets that comes and goes from universities, stations, work etc...
This is the perfect context in which an application like SafeStreets should be depolyed.
The application is thought for a world in which most of the people always brings along a smart device like a smartphone, able to take photographs and with a stable connection to the internet.

SafeStreets will allow every person of a city to collaborate to make streets safer and help police and municipality to identify areas where violations happen more often.

Data collected by the service can be later mined by both users and authorities; this can be useful for users, in order to avoid dangerous streets or really messy car parks, and for authorities in order to strengthen controls in the most unsafe areas or to plan possible interventions.
In a world where everyone owns a smart device, it is really easy to modify images so, to avoid fake data sent by the users, SafeStreets will also implement several countermeasures to check the veracity of every piece of information.

1.3 Definitions, acronyms and abbreviations

1.3.1 Definitions

- User: any costumer of the application.
- Regolar User: the customer of the application that exploits the service to send pictures and informations about traffic violations.
- Municipality: the government of a city; it can mine data from SafeStreets to obtain information about traffic violations and statistics.
- Police officer or Policeman: police officer employed by the municipality.
- Authorities: comprehend the municipality and the local police.
- Institutions: municipalities or other acknowledged entities allowed to access SafeStreets data.
- Traffic ticket: a fee issued by the local police to people that own a vehicle that committed traffic violation.
- Traffic violation: occurs when drivers violate laws that regulate vehicle traffic on streets or parking.
- Municipal Authorities: employees of the municipality

1.3.2 Acronyms

- RASD: Requirement Analysis and Specification Document
- SS: SafeStreet
- GPS: Global Positioning System
- API: Application Programming Interface
- OS: Opertating Systems
- DB: DataBase

1.3.3 Abbreviations

- (Gn): n-Goal
- (G1.n): n-Goal for advanced function 1
- (G2.n): n-Goal for advanced function 2
- (Dn): n-Domain assumption
- (Rn): n-Requirement

1.4 Revision history

- Version 1.0 - 10/11/2019
- Version 1.1 - 15/12/2019
  - improved readability
  - fixed workload management (3.3.2)
  - updated alloy notes
  - fixed section 2.2
  - domain assumption removal (2.4)
  - corrections to scenario 7 (3.2.1)
  - fixed use case descriptions 2 and 3 (3.2.2)
  - fixed UseCaseDiagram-Policeman
  - fixed user sending report sequence diagram

1.5 Reference documents

- Specification document *SafeStreets Mandatory Project Assignment*
- Slides from the course *Software Engineering 2* by professor Matteo Rossi

1.6 Document structure

- Chapter 1: an introduction to SafeStreets; it describes the purpose and the goals that the application aims to reach. It defines also the scope of the application, that includes the analysis of the world and of the shared phenomena.
- Chapter 2: provides an overall description of the system functionalities. It contains the various charts and diagram describing the domain, the most important requirements, the needs of the users, the various stakeholders and various constraints and assumptions.
- Chapter 3: provide a more specific study about the requirements of the applications describing the various interfaces needed (user, software and hardware), functional requirements with associated use cases and use case/sequence diagrams, performance requirements, design constraints and various software attributes (reliability, availability etc. . .).
- Chapter 4: provides a formal analysis of the application using Alloy. Here will be shown the alloy model of the most critical aspects, various comments to show how the project has been modelled and the world obtained by running the model itself.
- Chapter 5: information about the effort spent by every member of the team on the project.

2 Overall Description

2.1 Product Perspective

The aim of SafeStreets is to increase the security around the streets of the cities, also simplifying the interventions of the authorities.
In order to make this possible, the application must include some functionalities that helps determining whether a report is real or fake.
To monitor the position of the user sending the violation report, SafeStreets will exploit device's available GPS sensor and will also include a detailed map where users can discover the most unsafe areas or streets.
To allow users to take pictures of the violation, the application will use the camera nowadays inserted in every smartphone.
SafeStreets will also need a stable connection to the internet in order to send violation reports or to mine data from the database.

In order to differentiate functionalities for the various types of users, the application will be accessible by three different clients, one for each kind of user (Policeman, Municipal Authority and Regular User).
Each type of user has its feasible actions: RegularUser profiles will send reports and view data, Policeman profiles will act on reports and MunicipalAuthority profiles will analyzed the crowdsourced data.
MunicipalAuthority profiles will access the services via a webpage, also used to register all of the authorities profiles (MunicipalAuthority and Policeman).
Depending on the kind authority and the city he/she works for, SafeStreets offers different levels of data visibility.

2.1.1 Class Diagram

The high-level class diagram provides a model of the application domain, containing the most important classes that will be necessary to build the system.

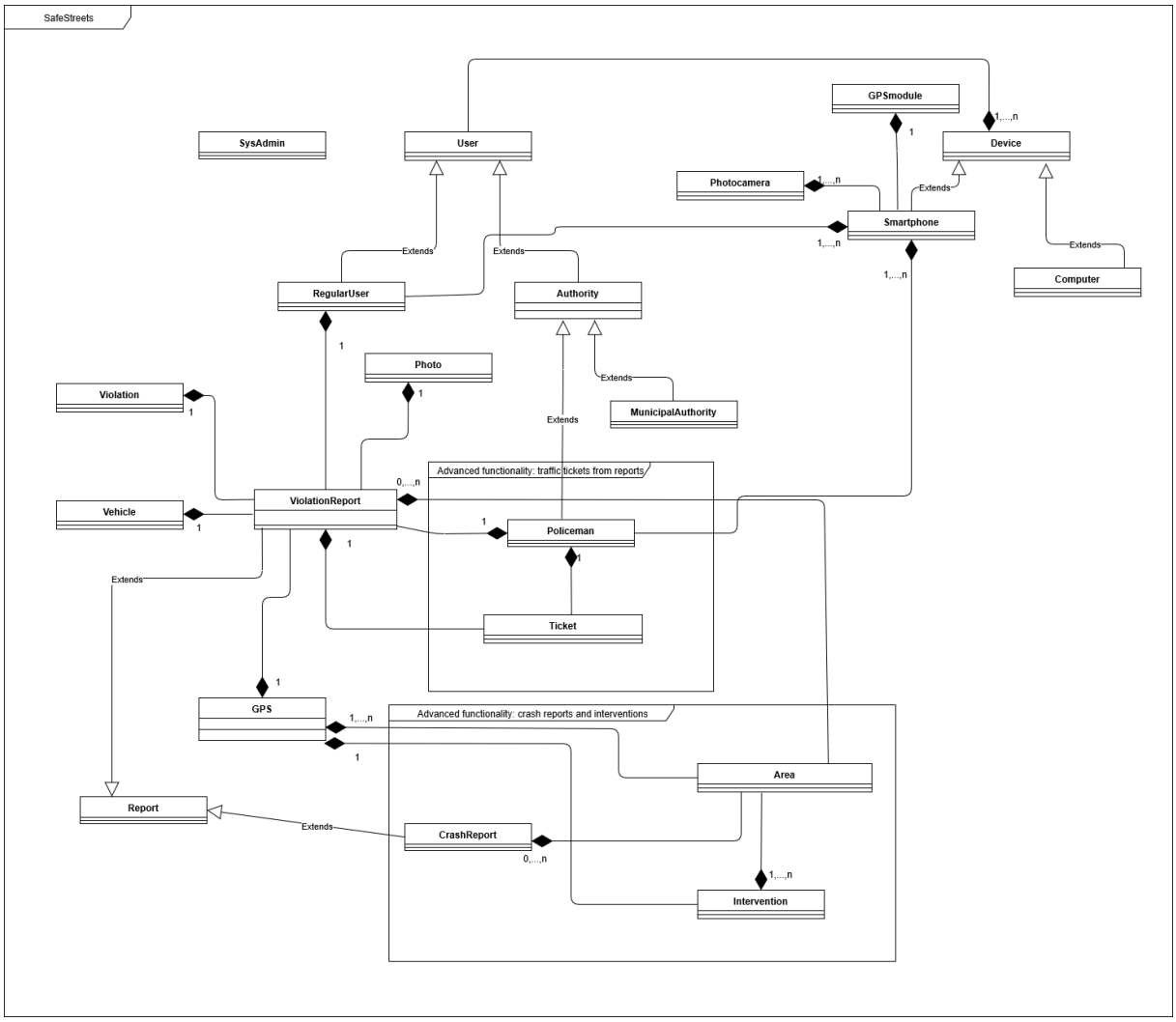

Figure 1: Class diagram

2.1.2 Statecharts

Now we will analyse the most critical aspects of the application workflow, modelling their behaviours using state diagrams.

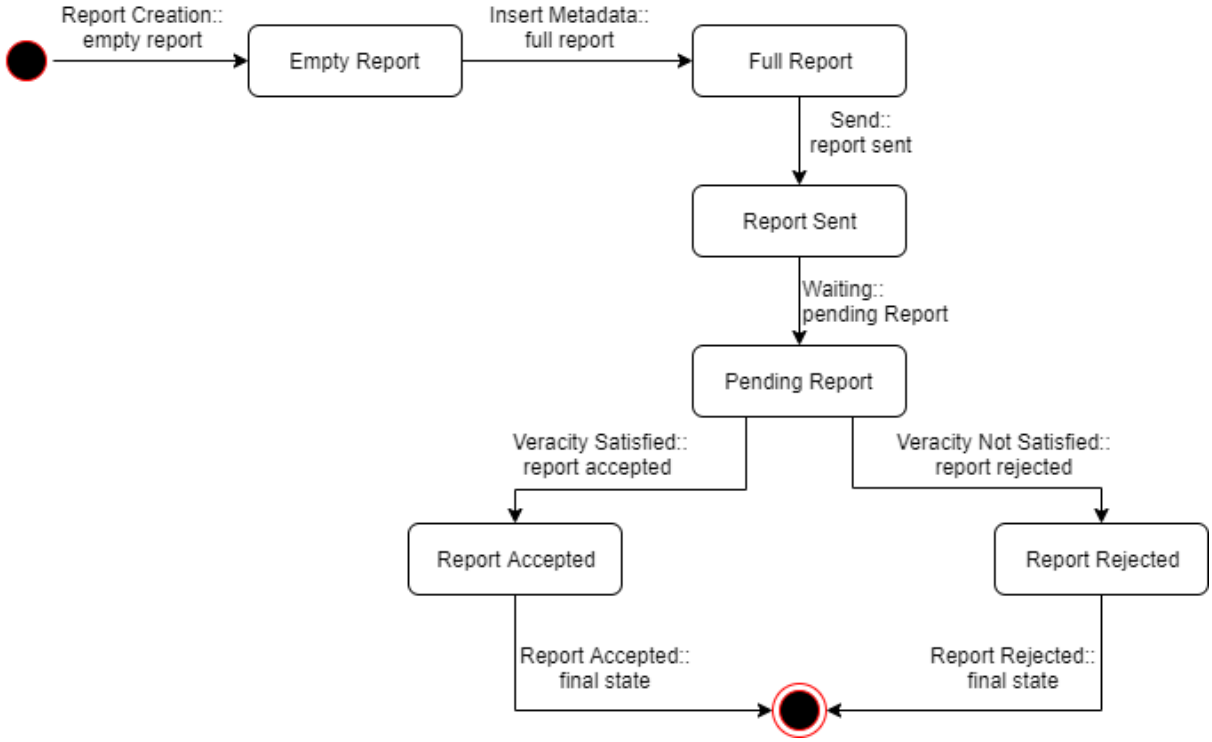

Figure 2: Sending a report

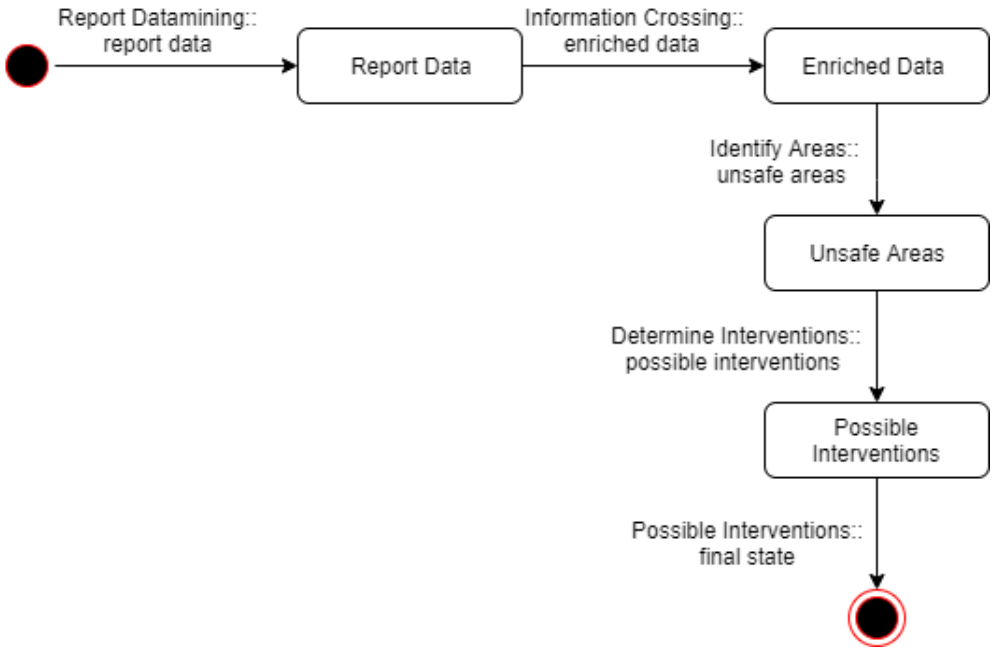

Figure 3: System suggesting possible interventions

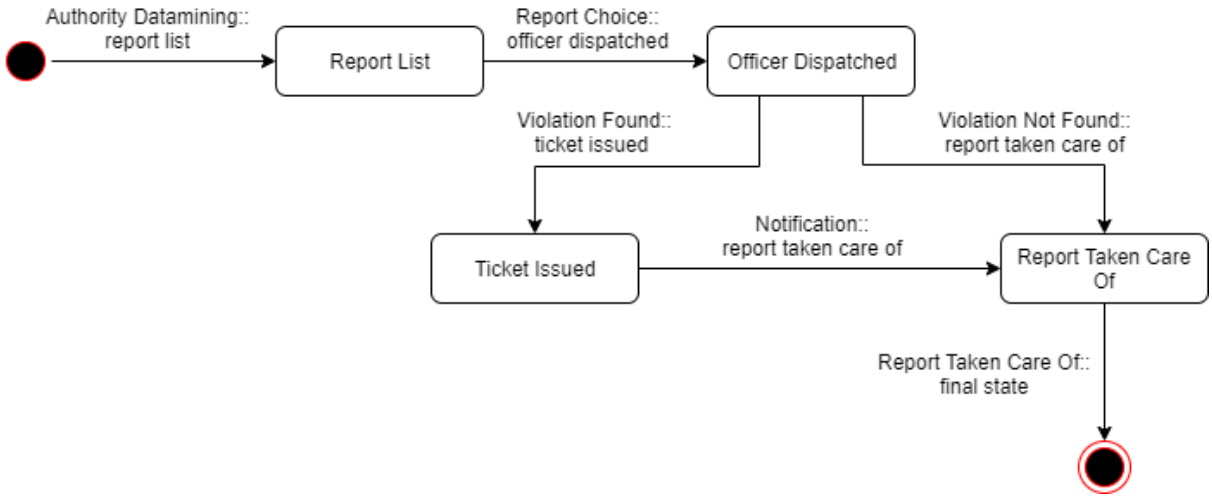

Figure 4: Local police officer taking care of reports

2.2 Product Functions

In this section we analyse the main functions of SafeStreets:

- Data Management:

  The users must give the application the permission to access their data and device's functionalities, such as GPS position and Camera.
  They are also asked to insert personal data during the account registration phase.
  If these requirements are met, SafeStreets allows authenticated users to send violation reports including geographical position and pictures of the violation.
  All this data is then used to check the veracity of a report and to help the authorities to identify the violator.
  On the other hand the authorities must insert authentication data in order to access the information stored by SafeStreets.
  The system must be also able to use stored data about tickets issued by the local police to build statistics.

- Report management:

  One of the main features of SafeStreets is the possibility for users to send reports about the traffic violations that occurs around them.
  The system must be able to automatically decide whether a report must be discarded or kept in memory.
  The reasons for reports to be discarded are basically three: the user signaling the report is not recognized; the report has been tampered with (or not appropriate); the plate number recognized by the system does not match the one inserted by the user.

- User management:

  The system includes two different clients for the login of the different type of users.
  One is used by basic users that can send reports and discover the unsafe areas of the city, the other is used by police officers and include exclusive functions, such as the possibility to signal that a report is under analisys.
  Finally SS will include a webpage with a dedicated area for registered authorities, with many ad-

vanced funcitons available such as reports data, maps of violations and suggested interventions.

- Possible intervention suggestions:

  SafeStreets must be able to automatically suggest possible interventions to avoid traffic violations in the most unsafe areas.
  The suggestions will be more accurate the more reports are sent from the same area.
  For example, if the applications receives a lot of reports about cars parked on the bike lane, there might be need of a barrier between the street and the lane.

- Information crossing:

  The system must be able to cross its data with informations about car accidents coming from the municipality.
  This functionality can be useful to identify the most dangerous areas or to accurately suggest interventions.

2.3 User Characteristics

Here is a list of the actors of the application:

- Regular Users: Every authenticated user that is able to send violation reports and mine the application data to know the most unsafe areas.
- Policemen: Users with a police service number able to access data about reports sent by the basic users.
- Municipal Authoritites: authorised users from the municipality of a city able to mine more detailed data such as the possible interventions suggested by SafeStreets.

2.4 Assumptions, dependencies and constraints

- (D1) Users' devices are connected to the internet
- (D2) Users' devices are equipped with a working camera
- (D3) GPS signal is always available on users' devices
- (D4) Users that want to access authorities' data must be acknowledged institutions
- (D5) Car's license plates are unique
- (D6) Every license plate is related to one and only one vehicle

3 Specific Requirements

3.1 External interface requirements

3.1.1 User Intefaces

SS will feature two different mobille clients: one available to ordinary users and another one dedicated to police officers. Each client will feature a dedicated graphical user interface.
In particular the user interface of the police officers client will offer graphical support for the additional exclusive functions.
Finally SS will also provide a webpage which will offer access to advanced feature to acknowledge institution.
The following mockups give an idea of the appearance of the application on release.

Figure 5: Mockup of the login page

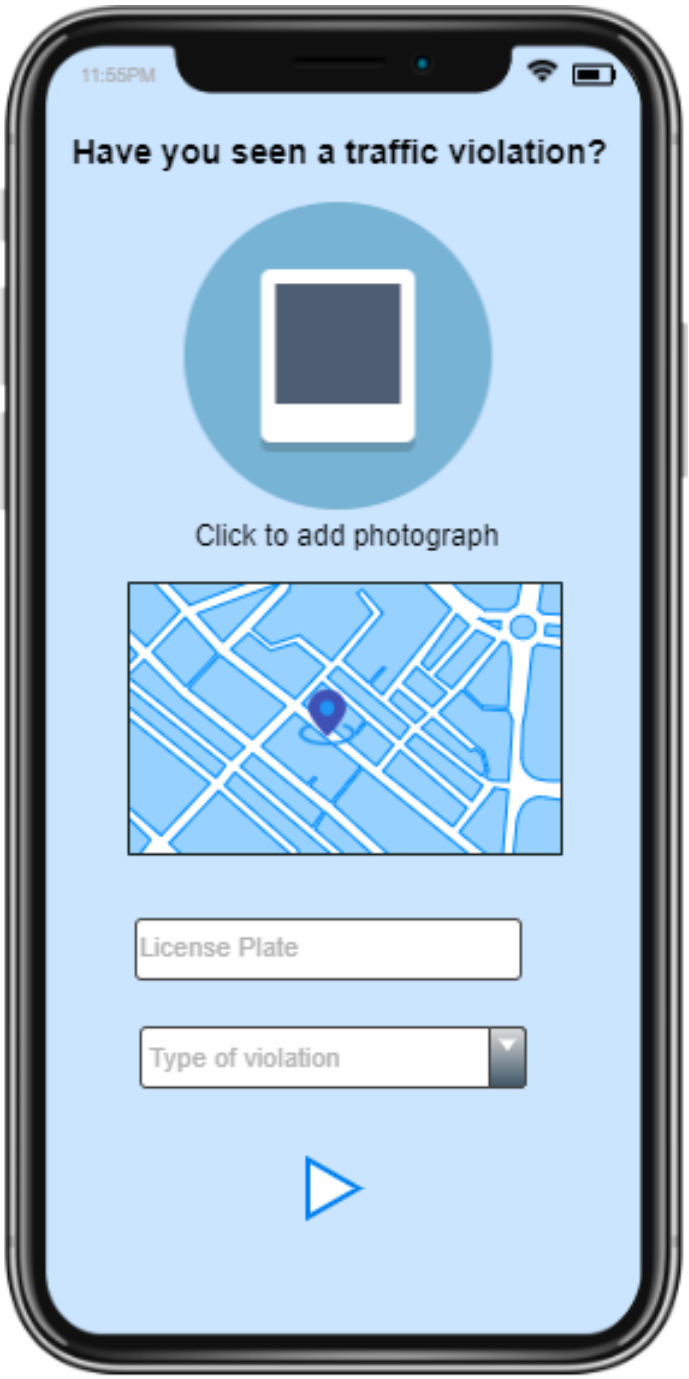

Figure 6: Mockup of the form for submitting a report

Figure 7: Mockup of the camera interface

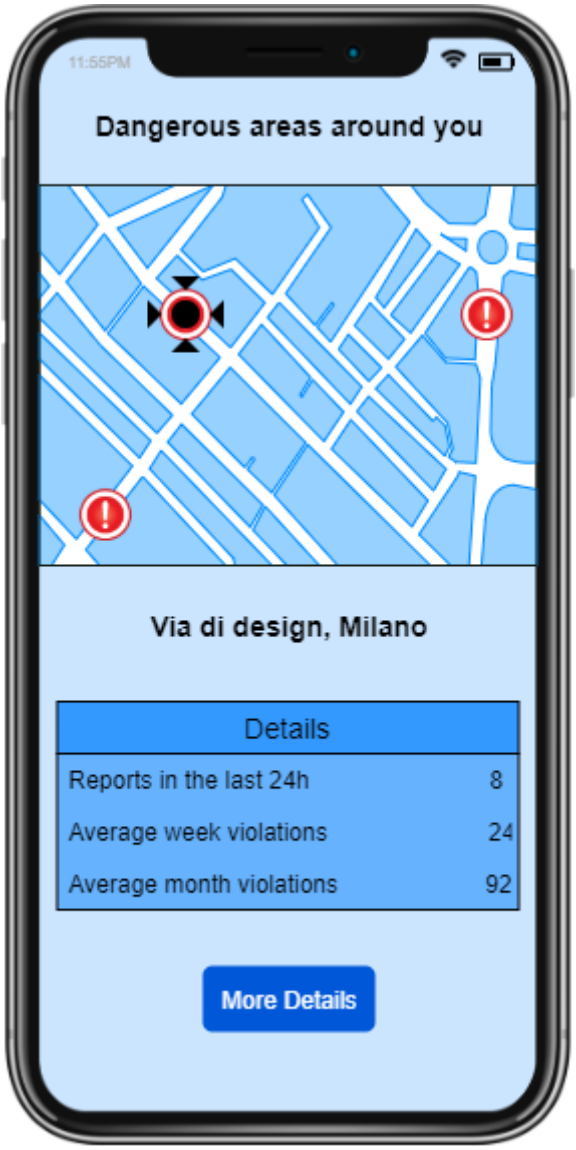

Figure 8: Mockup of Unsafe areas page

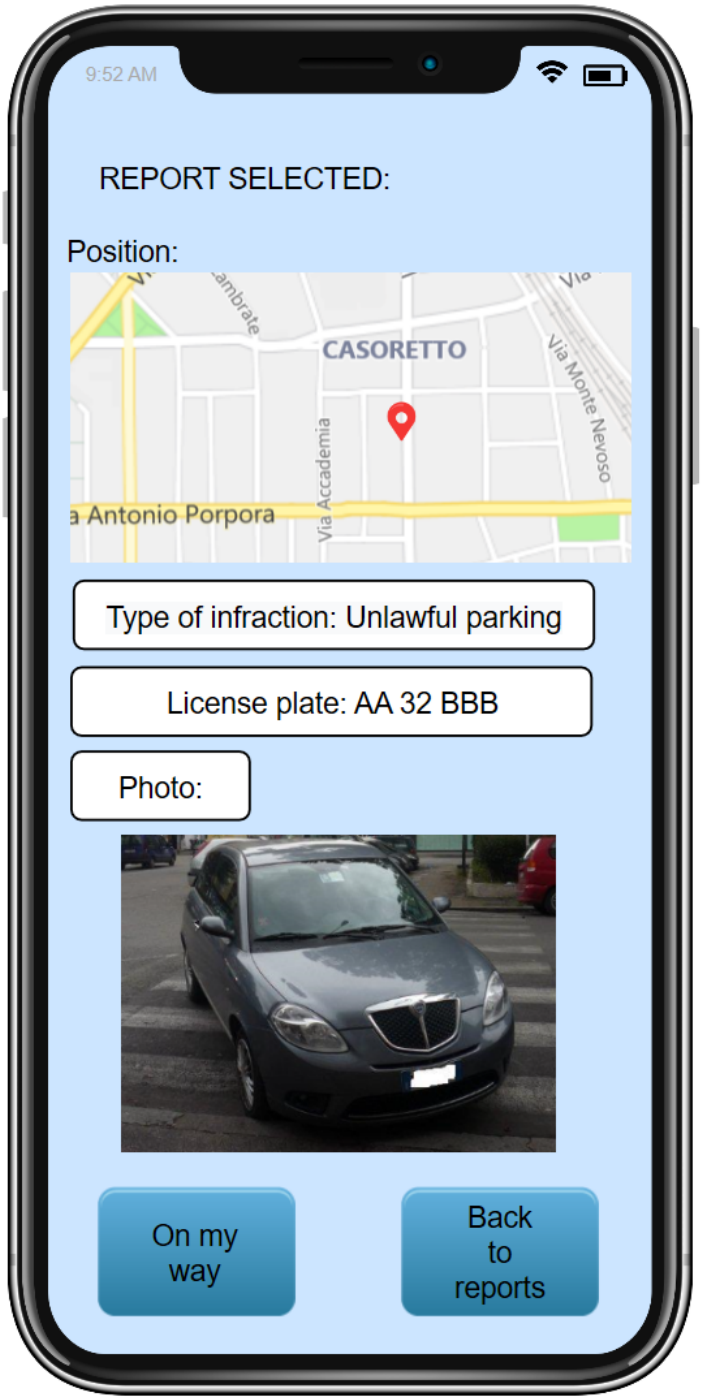

Figure 9: Mockup of part of the police client

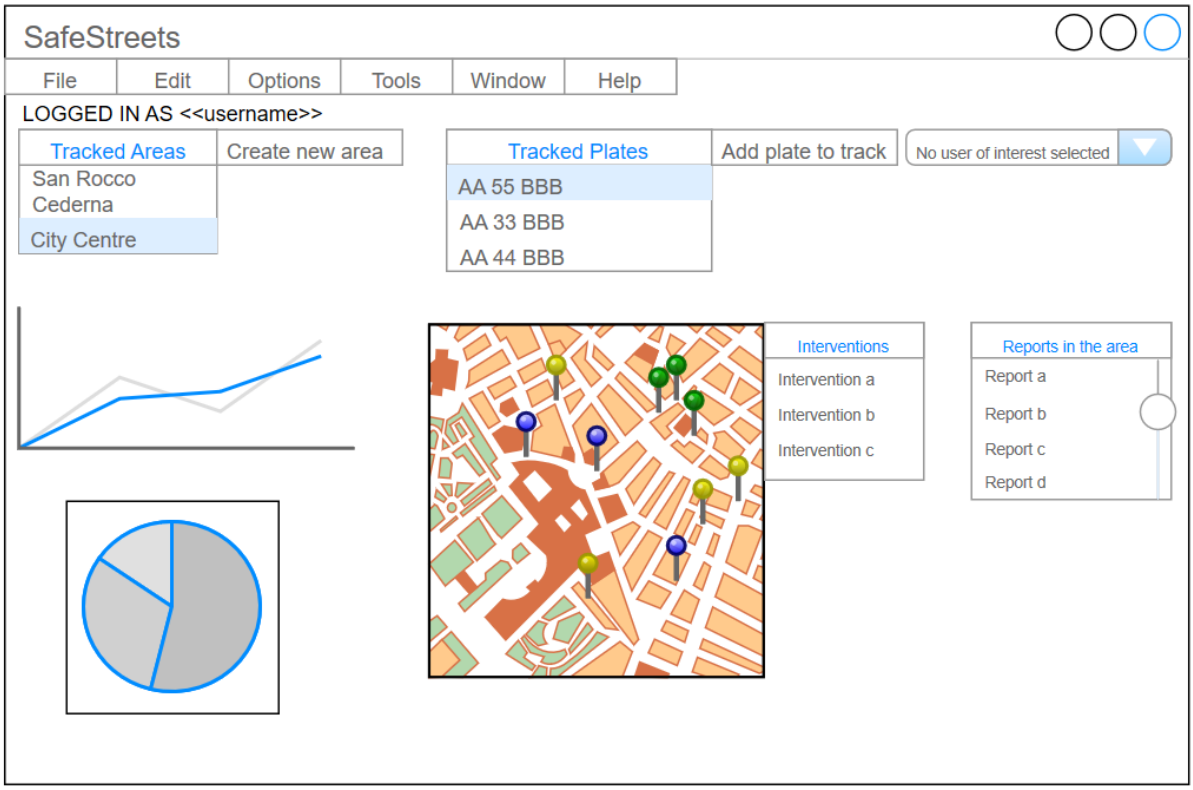

Figure 10: Mockup of the municipal authority dedicated webpage

3.1.2 Hardware Interfaces

Hardware interfaces are mainly required for the acquisition of the data composing a violation report. In particular these represent the necessary interfaces for SS service:

- Smartphone Sensors:
  The access is provided by the APIs and drivers offered by the mobile OS. Those APIs grant the possibility to easly exploit camera and GPS, crucial sensors inside SS user client.
- Internet Connection:
  Internet access is vital for the applications since it is entirely based on the comunication between user client and a remote server. So the user client is designed to be installed on a smartphone, assuming that every device will feature an internet module accessible through OS APIs.
  Internet connection is also exploited to retrieve timestamps of each violation and to increase the GPS location accurancy.

3.1.3 Software Interfaces

Software interfaces required by SS user client are represented by the APIs offered by the mobile OS for the management of the aformentioned hardware modules. In particular, access permission and OS level APIs are required for:

- Camera
- GPS
- Storage
- Internet Connection

Additionally SS client will exploit the APIs of a third party Street Map Service in order to provide to the users feaure like violations statistic per area, maps of violations and so on.
Looking at the server endpoint of SS service, the development of an high performance DataBase is reguarded as a main concern for the application scope. Moreover, the distributed architecture of the application makes the connection management central: interfaces for distributed software development will play a primary role.
Finally considering the involvement of third parties for the development of advanced feature of this application, is worth noting that the interfaces with external services will be defined in collaboration with representatives of the third parties involved.

3.1.4 Communication Interfaces

The unique (but fundamental) communication interface required for the scope of this application is the internet connection, as pointed out in the previous sections. The distributed nature of SS service makes this communication between endpoints strictly necessary. Thus, the system should provide a device agnostic connection with adeguated performance.
In order to provide access to third parties for stored data (mainly for mining purposes), dedicated APIs will be developed and maintained.

3.2 Functional Requirements

G1 - **Allow users to signal traffic violations**

- (R1) SS user client must provide a registration interface.
- (R2) SS must assign a unique ID to each registered user.
- (R3) SS user client must provide a violation report function.
- (R4) SS should verify each report is sent from an existent ID.

G2 - **Allow users to include pictures in violation reports**

- (R5) Application must have permission and access to the device camera.
- (R6) Pictures should be taken through camera inside the application to ensure protection against manipulations.

G3 - **The system must be able to retrieve the geographical position of a violation and add it to the report**

- (R7) Application must have permission and ability to access to GPS service offered by the smartphone device.

G4 - **The system must recognise (from pictures) license plates**

- (R8) The system should implement an image recognition algorithm.
- (R9) The system should ask the user to manually insert the license plate number of the offender car pictured in the provided photo for data integrity purposes.

G5 - **The system must store user made reports**

- (R10) SS must have a consistent database.
- (R11) Every report received has to be saved in SS database.
- (R12) SS must ensure that data cannot be tampered with.

G6 - **Allow users and authorities to mine information collected by the application**

- (R13) SS should provide an authorities dedicated client.
- (R14) SS authority client should provide a registration function.
- (R15) SS should provide the APIs and the tools to allow mining information by 3rd party.
- (R16) SS should implement an automated tool for data analysis and display.

GA 1.1 - **The system must be able to cross the data collected with information about the accidents coming from the municipality**

- (R17) The involved municipality should provide incidents data of his metropolitan area.

- (R18) Data access interfaces should be provided by municipality or defined by SS designers in collaboration with people from the municipality.
- (R19) The system should have already collected a suitable amount of data.

GA 1.2 - **The system must be able to suggest possible interventions to decrease the risk of violations in unsafe areas**

- (R16) SS should implement an automated tool for data analysis and display.

GA 2.1 - **Allow the local police to retrieve data about violation to generate traffic tickets**

- (R20) SS should provide a dedicated client to police officers.

GA 2.2 - **The system must be able to ensure the veracity of the information sent by the users**

- (R2) SS must assign a unique ID to each registered user.
- (R3) SS should verify each report is sent from an existent ID.
- (R21) SS should include anti-flooding protection mechanism (e.g. timer for each ID).
- (R6) Pictures should be taken through camera inside the application to ensure protection against manipulations.
- (R22) Report forward to DB should exploit a secure and encrypted channel.

GA 2.3 - **The system must track whether the police has taken care of a certain violation**

- (R23) SS policeman client should offer a function to signal that a traffic ticket (related to a previous report) has been generated.
- (R24) SS stores in its DB traffic ticket emission alongside the corresponding report.
- (R25) SS tracks whether a report has been taken care of.

GA 2.4 - **A traffic ticket must involve the same license plate that the report refer to.**

- (R26) The system must ensure that the traffic ticket emission signalling (policemen client) is available only for taken care reports.
- (R27) SS will ask the officer the license plate number and will accept to signal ticket emission only if the inserted number match the one contained in the report.

GA 2.5 - **The system must be able to build statistics from the information about issued tickets**

- (R16) SS should implement an automated tool for data analysis and display.

3.2.1 Scenarios

1. **Bad Parking**: Bob is a commuter and every morning needs to reach the train station by car: the daily struggle to park his car in the surroundings is even worsened by parking violations. One morning, coming across another car occupying two parking slots, Bob decides to start contributing to SafeStreet service and its violation map: he downloads and installs SS app, accepts to grant Camera and GPS authorizations to it and fills a brief form for the registration. Once logged in, he uses the dedicated report feature. He takes a picture of the violation, inserts manually the license plate number and, after the application completes the verification of the inserted data, he sends the violation report, which also includes the GPS location and the current time.

2. **An anxious mother**: every day Laura lets his 12 years old son walk to school by himself. One day she hears about SS, a new app providing information about traffic violation in her city, and thinks that it can be useful to check if her son daily walking route is reasonably safe or not. Using SS violation map, which provides a street level highlight of violation distribution, she finds out that his son usually walks in potentially unsafe streets, with cars parked on sidewalks forcing pedestrians to walk in the middle of the street, and suggests him a new route based on the map provided by SS.

3. **Smart Municipality**: Milan municipality is looking for a large data set regarding traffic monitoring for his metropolitan area. The administration aims to mine useful information from it in order to identify critical areas and plan interventions. They found out SS and its violation DB, so they subscribe to the service as a public institution filling a required form and gain access to the needed data. With high privilege data access granted to public authorities, SS provides them a suitable amount of data for the administration goal.

4. **Traffic Monitoring Service**: Gotham City municipality has just introduced a new traffic plan in one of his main districts and wishes to track precisely the evolution of traffic violations. SS should do the trick: Gotham municipality registers to SS service as public authority and now can have privileged access to the data collected by the application. However, Gotham City has no infrastructure dedicated to large data set mining. No worries: the impact of the new traffic plan can be evaluated exploiting built-in SS violation map and traffic tickets emission trends.

5. **SafeStreet for Safer Cities**: the city of Monza provides a dedicated service offering data about incidents on its territory. SS has exploited its advanced features to cross its own information about traffic in Monza with the data provided, identifying possible unsafe areas and suggesting possible interventions. The administration of the city knows that managing the traffic plan of a vast city is complex and find that SS automated analysis could really lower effort and costs of the operations. Thus, the municipality decides to register to SS services and to exploit its traffic monitoring and intervention suggestion tool.

6. **Traffic Tickets**: NotSoSafe City is struggling trying to guarantee a decent level of violation control in his vast metropolitan area, considering his personnel available is limited. The municipality comes to know that SS provides an efficient violation signalling service and a dedicated police officer client that even tracks ticket emissions. After registration as a public institution, the municipality issues to every policeman the installation of SS client on corporate (or personal) phone. Policemen can now take advantage of the real-time violation notification service of SS app for quick interventions if needed. Moreover, SS violation map offers the municipality a strong tool to simplify territory control: officer patrol activity can be directed to the most unsafe areas reported by the application.

7. **Policeman Intervention**: Anna is a police officer working for Rome municipality. The city administration has recently decided to exploit SS service in order to optimize traffic ticket emission. Anna is patrolling district one and desires to efficiently carry on the task. Anna opens SS client and taps on the dedicated section: a map with pending reports is visualized. Anna decides to take care of a report: she taps on the report and selects "On my way" option. The system marks that an officer has been dispatched for the picked report. Once on place, Anna verifies that the violation reported has truly occurred and issues a traffic ticket. She finally marks the report as" taken care" within SS app and gets back to her patrolling activity.

3.2.2 Use Case Description

1. **User Registration**

Actors	User
Goals	G1
Entry Conditions	User opens the mobile app for the first time
Event Flow	a) The user accepts to grant SS permission to access camera and GPS. b) SS user interface shows a menu with Log In and Sign Up options. c) The user taps on Sign Up opening the registration form. d) The user fills at least the mandatory fields with required personal information. e) The user accepts SS privacy policy by ticking a box. f) The user confirms the registration by cliking on a link sent to the inserted e-mail account.
Output Conditions	The user completes the registration and can access SS features.
Exceptions	a) The user does not accept required permissions. In this case the app will close. b) The user is already registered. An error message is displayed asking to reinsert the data is displayed. c) Mandatory fields are left empty or filled with invalid entries. An error message is displayed asking to reinsert the data is displayed. d) The user does not accept privacy policy. An error message is displayed asking to accept the policy is displayed. e) Verification link expires before user confirms registration. In this case the account is deleted from SS system.

2. **Institution Registration**

Actors	Institution/Authority, SysAdmin
Goals	G6, GA2.1
Entry Conditions	No entry condition
Event Flow	a) The institution connects to SS website. b) In the website institution dedicated area, the authority finds a mandatory registration form. c) The institution fills all the mandatory fields and accepts privacy policy. d) A SS SysAdmin takes care of the registration request, verifying the data and creating a corresponding Municipal Authority profile.
Output Conditions	Registration successfull: the institution can access advanced SS features.
Exceptions	a) The institution is already registered. b) Mandatory fields are incomplete. c) The institution does not accept Privacy policy. d) SS SyAdmin raises doubts regarding submitted data. All exceptions raise an error asking to complete the unfulfilled operations.

3. **Violation Signaling**

Actors	User
Goals	G1, G2, G3, G4, G5
Entry Conditions	User is registered, logged in the mobile app and wants to signal a violation
Event Flow	a) The user opens SS app and taps on the "Signal a Violation" option. b) The user selects one of the possible violations presented in graphic menu. c) The app asks the user to attach a photo of the violation, picturing the vehicle involved with license plate visible, and to manually insert license plate number of the vehicle. d) The user takes a photo of the violation within SS app and insert license plate number of the vehicle. e) The system verifies if the license plate pictured and the manually inserted one match and readies the violation report. f) The app shows a menu displaying a " Send Report" and a "Cancel" options. g) The user taps on "Send Report" and the report is sent to SS systems.
Output Conditions	The report is stored in SS database.
Exceptions	a) License plate number recognized by SS and the user inserted one don't match. An error message is raised and SS asks the user to retake the photo.

4. **Providing Useful Information to Users**

Actors	User
Goals	G6, GA2.5
Entry Conditions	User is registered, logged in the mobile app and wants to consult violation information
Event Flow	a) The user opens SS app and taps on the "Statistics" option. b) The user selects the prefered function in the menu displayed (e.g. violations maps, traffic tickets statistics). Note: functions availability may vary depending on the service offered by the city municipality.
Output Conditions	The user acess to the requested service.
Exceptions	a) The user requests a service not available in his city. An error message explaining the reasons of the function absence is displayed.

5. **Providing Data to Mine to Istitutions**

Actors	Institution/Authority
Goals	G6
Entry Conditions	Institution is registered to SS service and desires to access to data it collects
Event Flow	a) The institution specifies a proper query for the data required. b) The system verifies if the query author is a registered institution.
Output Conditions	Requested data is provided to the institution.
Exceptions	a) The query is not well formatted. An error is reported and the transaction is aborted. b) The query author is not recognized. An error is reported and the request is not fullfilled.

6. **Providing to Institution Intervention Suggestions**

Actors	Institution/Authority
Goals	GA1.1, GA1.2
Entry Conditions	Institution is registetred to SS service and desires to access to "Suggested Intervention" advanced feature
Event Flow	a) The institution logs in the SS website b) The institution navigates to the "Suggested Intervention" sections c) In the interface displayed the institution searches the areas of interest for possible intervention suggestions
Output Conditions	The institution can access to intervention suggestions for the selected area
Exceptions	a) The institution fails to login in the website page. An error message asking to reinsert credential is displayed. b) In the selected area there are no suggested intervention. An error message explaining the absense of possible intervention is reported.

7. **Police Officer Takes Care of a Report**

Actors	Police Officer
Goals	GA2.1, GA2.3
Entry Conditions	The police officers finds in the SS client dedicated section an available violation report and decides to take care of it
Event Flow	a) The police officer selects a violation report among the ones displayed on the map. b) The app shows the report and two option: "On my Way", to take over the violaton report; "Back to Reports", to come back to violations map. c) The police officer selects "On my way" option. d) The police officer reaches the location indicated in the violation reports. e) The police officer verifies if the signaled violation has truly occurred and in this case emits a traffic ticket. f) The police officer concludes intervention specifying its outcome in SS app, through the options displayed: "Ticket emitted" or "Violation not verified".
Output Conditions	In case of verified violation, a ticket emission is stored with the corresponding violation.
Exceptions	a) A colleague takes over the violation before the policeman selects "On my way" option. An error message stating that the violation is already under analisys is displayed.

3.2.3 Use Case Diagram

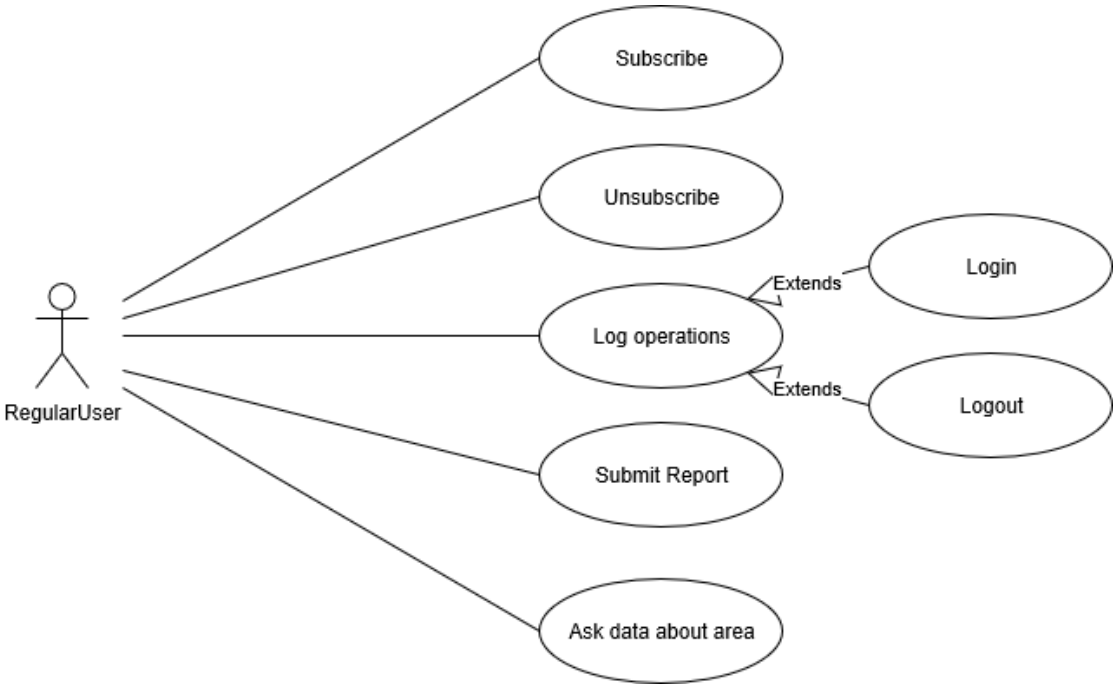

Figure 11: Regular user

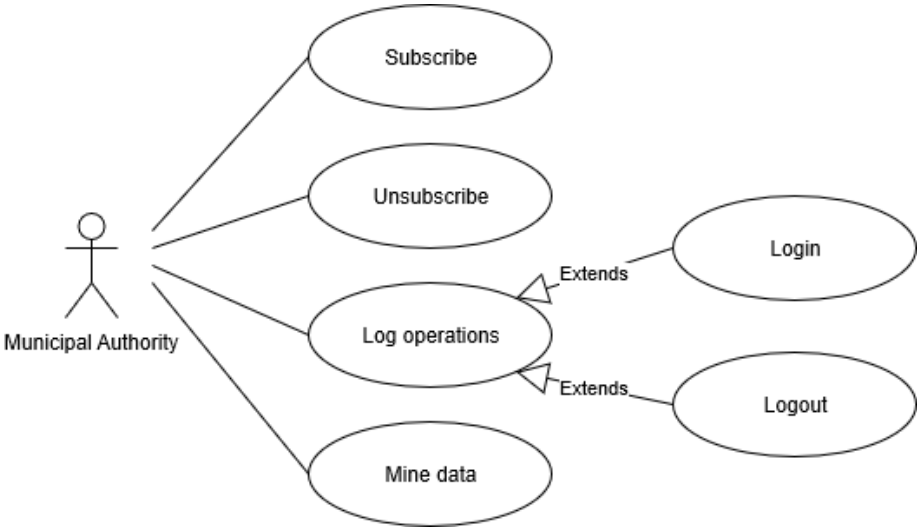

Figure 12: Municipal Authority

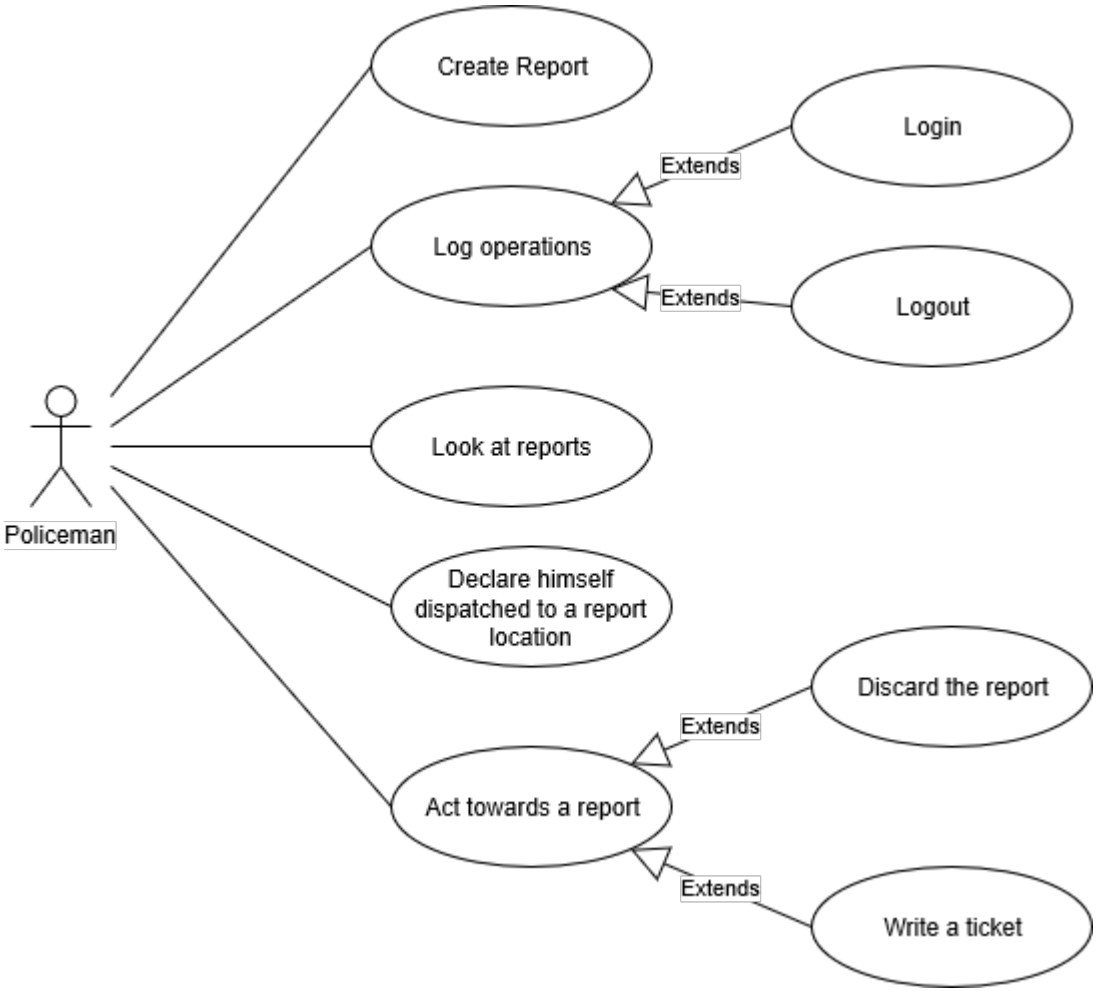

Figure 13: Policeman

3.2.4 Sequence Diagrams

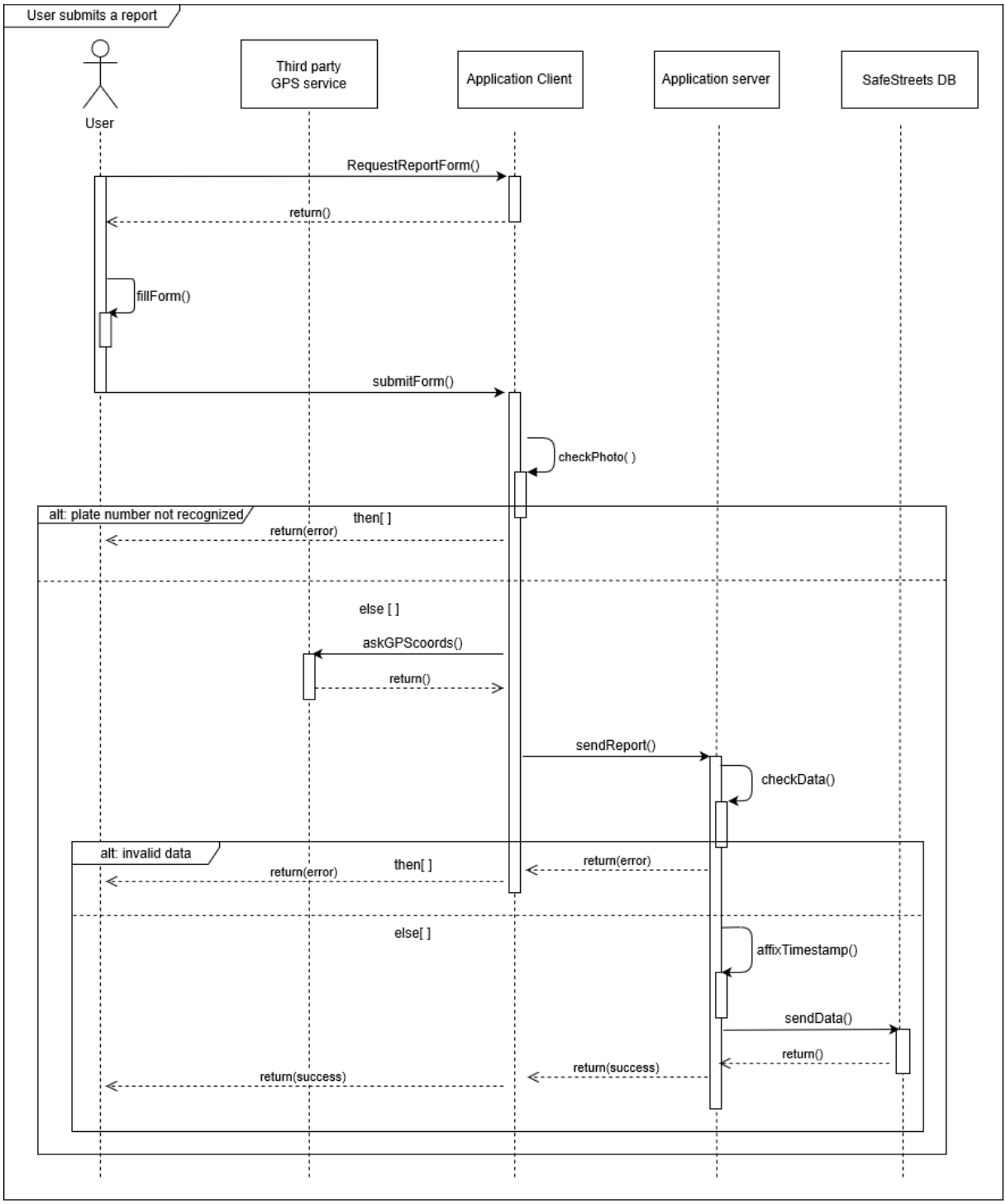

Figure 14: Violation Report submission

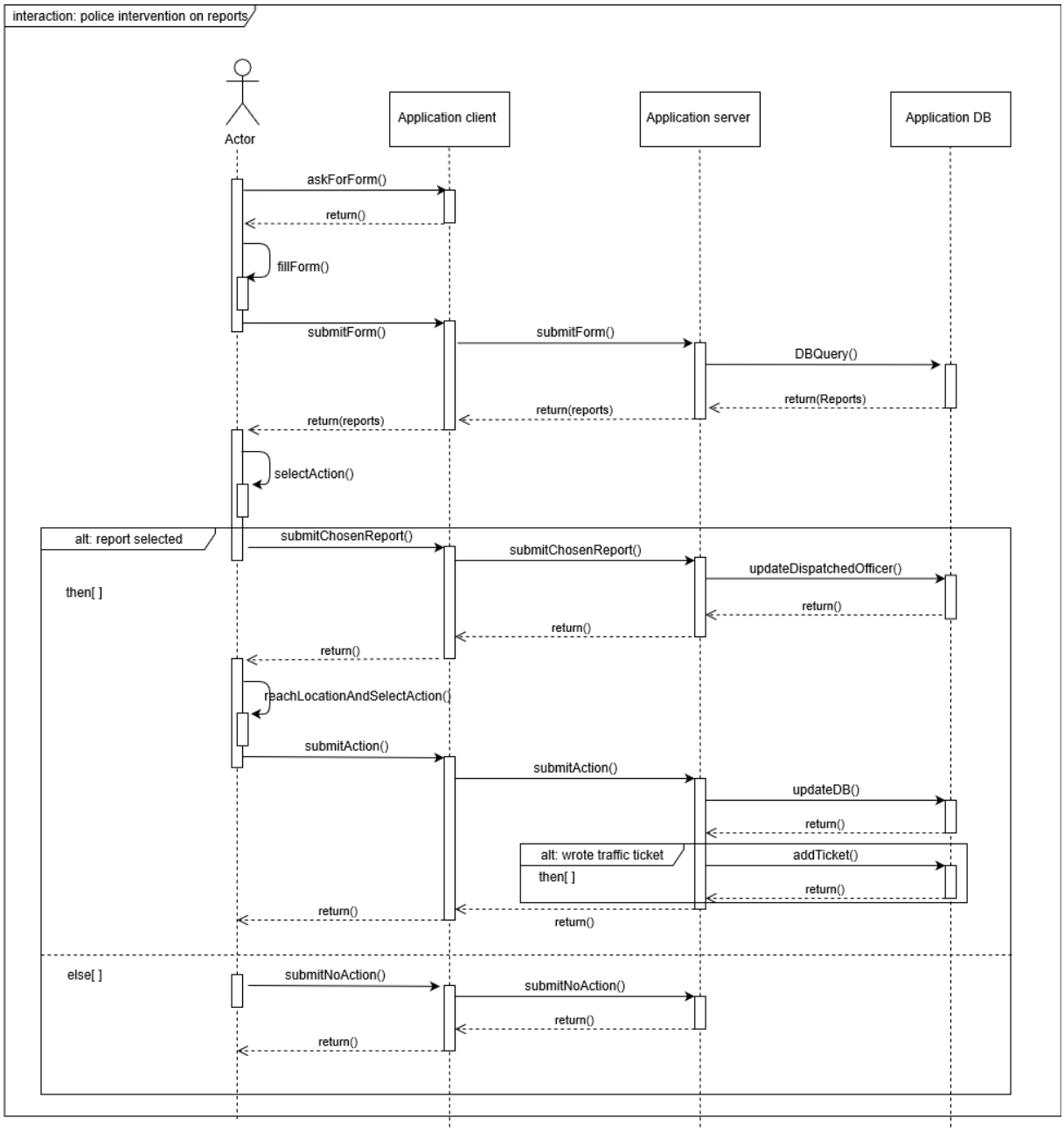

Figure 15: Police intervention

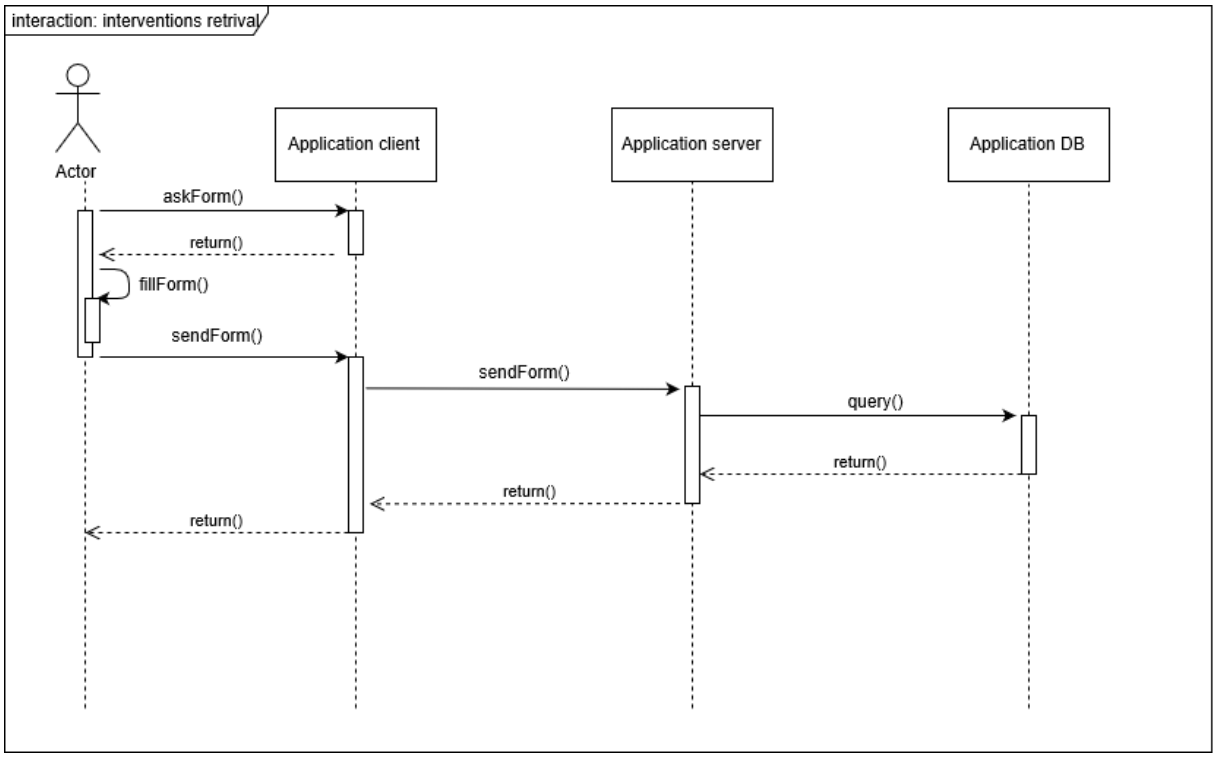

Figure 16: Suggested Intervertion retrival

3.2.5 Mapping on Requirements

Here we present the traceability matrix, showing relations among goals, requirements and use case of the application. For each relation, relevant notes can be included in the Comments column.

Row ID	Goals	Requirements	Use Cases	Comments
r1	G1	R1	1	
r2		R2	1	
r3		R3	3	
r4		R4	3	
r5	G2	R5	3	
r6		R6	3	
r7	G3	R7	3	
r8	G4	R8	3	
r9		R9	3	
r10	G5	R10	3	
r11		R11	3	
r12		R12	3	
r13	G6	R13	2	
r14		R14	2	
r15		R15	5	
r16		R16	5	
r17	GA1.1	R17	6	
r18		R18	6	
r19		R19	6	
r20	GA1.2	R16	6	
r21	GA2.1	R20	7	
r22	GA2.2	R2	1	
r23		R3	1	
r24		R21	3	
r25		R6	3	
r26		R22	3	
r27	GA2.3	R23	7	
r28		R24	7	
r29		R25	7	
r30	GA2.4	R26	7	
r31		R27	7	
r32	GA2.5	R21	5	

3.3 Performance Requirements

3.3.1 Response time

Server side SS is a data intensive application. Big volumes of data will be written and read at the same time. Given the nature of the application itself, fast responses are essential in order to make the policemen do their job properly. On the data analytics side, instead, the responses can be slower.

List of the response times:

- report forwarded to application DB: 500 ms, medium priority
- response to policeman query on DB: 500ms, medium priority
- responses to policemen actions insertion in DB: 200 ms, high priority
- responses to regular users queries on data: 500 ms, low priority
- responses to municipal authorities data-mining actions: 1 min, low priority

3.3.2 Workload management

SS will need to be able to sustain a heavy workload of database transaction: there will be a lot of simultaneous read and write operations. The workload required will differ depending on the differnt types of users.

- **Regular Users:** the number of data requests/posts will vary greatly depending on the number of users and the size of the city immplementing SS. As a safety measure, supposing to implement SS in a city roughly se same size as Milano, the number of expected requests or posts could exceed 25'000 daily, considering both reads and writes
- **Authorities:** given the relative small number of municipal and police employees, a good extimate in a city the size of Milano could be around 5000 requests or posts daily on average, insignificant when compared to the other users' requests/posts

In conclusion, given that the number of post or requests won't deviate much form the average, the system's scalability will not be a big concern.

3.4 Design Constraints

3.4.1 Standard Compliance

SS mobile client should respond to the following requirements:

- The application saves the current state when it is put in backgound or when the smartphone enter standby mode; it restores all the saved states when is called to the foreground.
- The application works flawlessly in an SD card, provided space is available.
- The application can be used either on landscape or portrait orientations.

The system to be delivered must comply with EU's GDPR (General Data Protection Regulation) and will follow any of its variations in the time to come.

3.4.2 Hardware Limitations

SS will have really different hardware limitations client side and server side:

- **Client side**:
  - Mobile Client: the only limitation is a working smartphone featuring a working camera, a GPS location module and an internet connection. Moreover, the smartphone will need enough memory space for app installation.
  - WebPage: working device with internet connection.
- **Server side** - SS will need to save big amounts of data, thus needing huge amounts of reliable storage. Server dimensioning is central for the scope of the application.

3.5 Software System Attributes

3.5.1 Reliability

Given the data-mining orientation of the application, a huge amount of data must be gathered in order to ensure that the statistic analysis on the aformentioned data is actually meaningful. Therefore, the system must be reliable in delivering un-compromised data to the database and reliably store them.
It is also necessary for the system to accurately update dispatched officers and what reports have been taken care of, in order to avoid an erroneous usage of municipal police manpower and time, whose are usually pretty low to begin with.

3.5.2 Avaibility

Once again, in order to gather enough information to produce meaningful statistical analysis, the system must collect as much data as possible. In order for this to happen the system must have a high availability, in the order of .9999. Moreover, an high availability is required for a fast(close to real-time) violation signaling to police officers client.

3.5.3 Security

Given that SS deals with a high volume of sensible data, such as pinpointing a user to a certain place at a certain time or having a list of various tickets emitted by the police, it is of the utmost importance for it to have a robust security apparatus. As a starter, all comunications between client and server must be encrypted (an asymmetric encryption mechanism with hash check should be enough); the same goes for the stored data. Other measures to be taken in order to avoid data leaks are the usual ones: don't use obvious names for tables and columns, filter all the user input, use adeguate csp to avoid code injections, use anti CSRF tokens and so on.
Please note that a leak could have disastrous consequences both on reputation and economic level, therefore security's importance cannot be understated and has to be considered one of the top priorities.

3.5.4 Mantainability

The greatest challenge in the maintanability field lays in the correct application of the "divide et impera" principle: if the application's modules have been defined and developed correctly then mantaining each module will be feasible with little cost in time and effort.

3.5.5 Portability

Portability is a relatively easy problem to solve. As a matter of fact, given that the users will access SS via a smartphone client, it will be enough to develop clients for Andorid and IOS and keep it updated to keep up with new os version releases.

4 Formal Analysis Using Alloy

This section contains the project analysis done using Alloy.

4.1 Introduction

The alloy model is focused on the main entities and rules of SS:

- User-made reports
- Main actors
- Rules regarding user-made reports
- Rules regarding police intervention
- Rules regarding data mining rights

While, on the other hand, cross-database interactions have not been modeled and the relation between area and violation has not been modeled in its entirety, as the rules about how to assign a violation to an area are missing.
For semplicity of representation the various violation types, two child signatures have been introduced. Note that, in reality, violation types are more than 2 and will be represented in a different way.
The alloy representation is also missing the SysAdmin, GPSmodule and Device (and child elements) representation.

4.2 Signatures

```
sig UsrId{}
abstract sig User{
    usrId: one UsrId
}
sig RegularUser extends User{
    trackedAreas: set Area
}
abstract sig Authority extends User{}
sig Policeman extends Authority{
    consideredReports: set Report
}
sig MunicipalAuthority extends Authority{
    trackedAreas: set Area,
    trackedUsers: set RegularUser,
    trackedPlates: set Plate
}
sig Position{}
sig Time{}
abstract sig ViolationType{}
sig ExpiredParking extends ViolationType{}
sig UnlawfulParking extends ViolationType{}
sig Plate{}
sig Photo{}
sig Intervention{}
sig Violation{
    violationType: one ViolationType,
    time: one Time,
    position: one Position,
    photo: one Photo,
    recognisedPlate: one Plate,//plate from photo
    writtenPlate: one Plate
}
sig Report{
    violation: one Violation,
    author: one User,
    dispatchedOfficer: lone Policeman,
    officerWhoTookAction: lone Policeman
}
sig Ticket{
    report: one Report,
    policeOfficer: one Policeman,
    recipient: one Plate
}
sig Area{
    reports: set Report,
    interventions: set Intervention
}
```

Figure 17: Alloy model signatures

4.3 Facts

```
49
50  //facts
51  fact noSameOrDoubleID{
52      no disj u1, u2: User |u1.usrId=u2.usrId
53  }
54
55
56  fact needSamePlate{
57      all v: Violation| v.recognisedPlate=v.writtenPlate
58  }
59
60  fact noDoublePhoto{
61      no disj v1, v2: Violation| v1.photo=v2.photo
62  }
63
64  //In the application there should be a margin of tolerance in regards of the GPS location as well as in
65  //regards of the time of the report,but in order to build and analyze the model in a simplier way these
66  //margins of tolerance have been ignored
67  fact multipleReportsForOneInfraction{
68      all disj r1, r2: Report| r1.violation.time=r2.violation.time and
69      r1.violation.violationType=r2.violation.violationType and r1.violation.position=r2.violation.position
70       and r1.violation.writtenPlate=r2.violation.writtenPlate and r1.violation.recognisedPlate=r2.violation.recognisedPlate
71       iff (r1.dispatchedOfficer=r2.dispatchedOfficer and r1.officerWhoTookAction=r2.officerWhoTookAction
72       and r1.author!=r2.author)
73  }
74
75  fact platesOnlyFromReports{
76      all p: Plate| p in Report.violation.recognisedPlate
77  }
78
79  fact photosOnlyFromReports{
80      all p: Photo| p in Report.violation.photo
81  }
82
83  fact noViolationWithoutReport{
84      all v: Violation| v in Report.violation
85  }
86
87  fact allPsitionsFromReports{
88      all p: Position| p in Report.violation.position
89  }
90
91  fact onlyConsiderReportsUndispatchedFor{
92      all p: Policeman| all r: Report| r in p.consideredReports iff #r.dispatchedOfficer=0
93  }
94
95  fact sameOfficer{
96      all r: Report| #r.officerWhoTookAction=1 iff ( #r.dispatchedOfficer=1 and
97      #r.dispatchedOfficer=#r.officerWhoTookAction)
98  }
99
100 fact ticketAuthor{
101     all t: Ticket| t.policeOfficer=t.report.officerWhoTookAction
102 }
103
104 fact noDoubleTicket{
105     no disj t1, t2: Ticket| t1.report=t2.report
106 }
107
108 fact noMunicipalAuthorityOnTheStreets{
109     no m: MunicipalAuthority| m in Report.author
110 }
111
112 fact reportFromOfficer{
113     all r:Report| r.author in Policeman iff r.dispatchedOfficer=r.author and r.officerWhoTookAction=r.author
114 }
115
116 fact rightPersonBilled{
117     all t: Ticket| t.recipient=t.report.violation.recognisedPlate
118 }
119
120 fact interventionsInArea{
121     all a: Area| #a.interventions>0 implies #a.reports>0
122 }
123
124 fact baseCityArea{
125     all r:Report| r in Area.reports
126 }
127
128 fact allInterventionsAboutSomeArea{
129     all i: Intervention| i in Area.interventions
130 }
131
132 fact onlyVehiclesInTheDB{
133     all m: MunicipalAuthority| m.trackedPlates in Violation.recognisedPlate
134 }
135
136
```

Figure 18: Alloy model facts

4.4 Assertions and world

```
137
138  //G1
139  assert allReportsHaveOnePhoto{
140      all r: Report| #r.violation.photo=1
141  }
142  //G2
143  assert allowDataMining{
144      all ru: RegularUser| all ma: MunicipalAuthority|
145      #ru.trackedAreas>=0 and #ma.trackedAreas>=0 and #ma.trackedUsers>=0 and #ma.trackedPlates>=0
146  }
147  //G3
148  assert plateRecognition{
149      all r: Report| r.violation.writtenPlate=r.violation.recognisedPlate
150  }
151  //G4
152  assert locationPinpointing{
153      all r: Report| #r.violation.position=1
154  }
155  //GA1.2
156  assert possibleInterventions{
157      all a: Area| #a.interventions>=0
158  }
159  //GA2.1
160  assert policemanWork{
161      all p: Policeman| #p.consideredReports>=0
162  }
163  //GA2.3
164  assert takenCareOf{
165      all r: Report| (r in Ticket.report iff #r.officerWhoTookAction=1) and #r.officerWhoTookAction=1
166      implies #r.dispatchedOfficer=1
167  }
168  //GA2.4
169  assert noSuchTicket{
170      no t: Ticket| t.recipient!=t.report.violation.recognisedPlate
171  }
172
173
     Execute
174  check allReportsHaveOnePhoto for 6
     Execute
175  check allowDataMining for 6
     Execute
176  check plateRecognition for 6
     Execute
177  check locationPinpointing for 6
     Execute
178  check possibleInterventions for 6
     Execute
179  check policemanWork for 6
     Execute
180  check takenCareOf for 6
     Execute
181  check noSuchTicket for 6
182
183  pred world{
184      #Plate>1
185      #Report>=2
186      #Ticket>=1
187      #Policeman>=1
188      #Violation>=2
189      #User>2
190      #ExpiredParking=1
191      #UnlawfulParking=1
192      #MunicipalAuthority>0
193      one r:Report| one a: Area| #a.interventions>0 and  r.author not in Policeman and #r.dispatchedOfficer=1
194  }
195
196
     Execute
197  run world for 6
```

Figure 19: Alloy model assertions and world declaration

4.5 Results

4.5.1 Checks on assertions

Figure 20: Results of the checks on the assertions

4.5.2 World generated

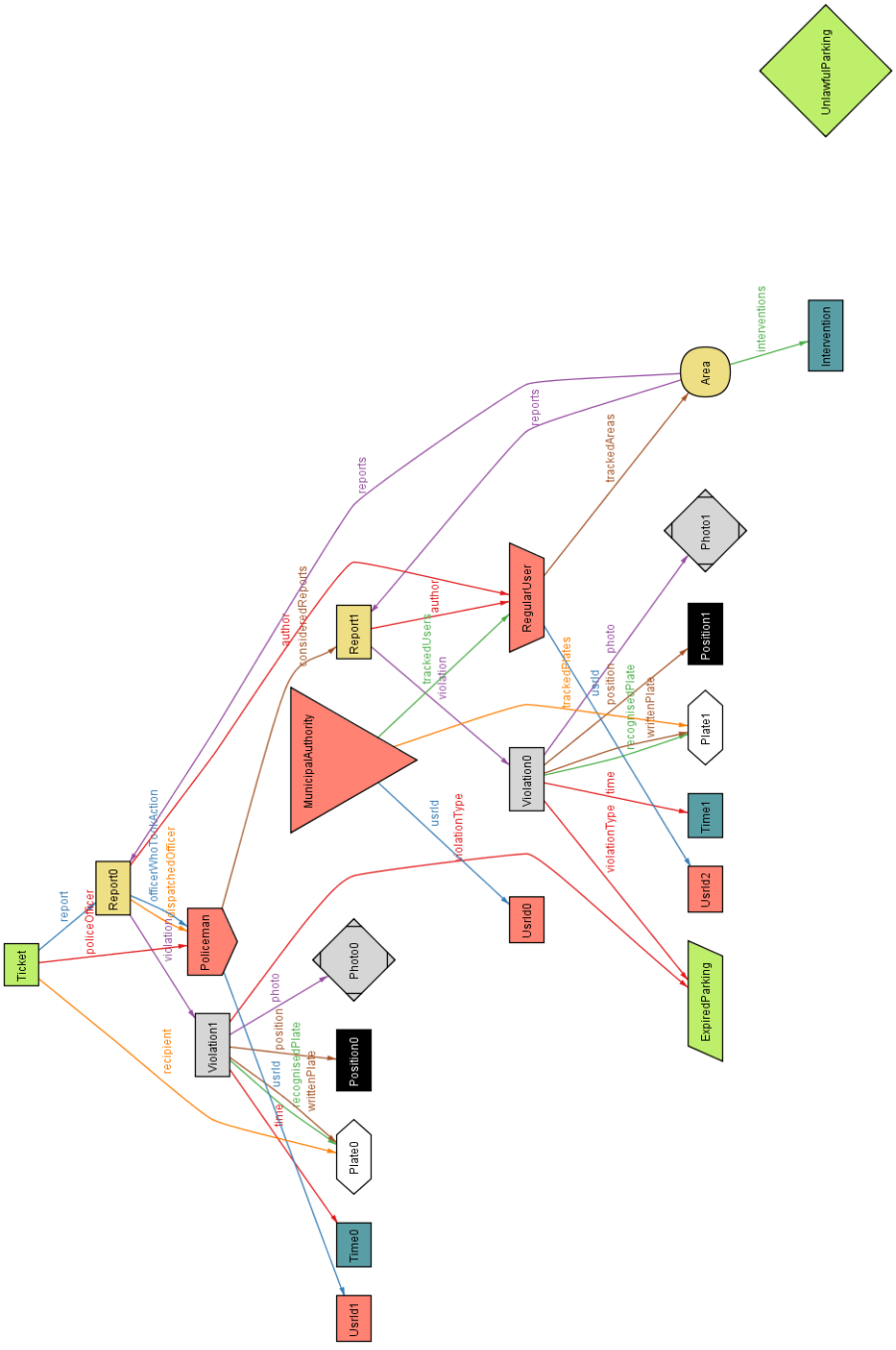

Figure 21: Example of the world described by the model

5 Effort Spent

- **Antonio Pipita**

Section	Hours
Purpose, Scope	2
Definitions, Document Structure	1
Product Perspective and Functions	1
Assumptions, dependecies and constraints	1.5
External Interface Requirements	1
Functional Requirements	4
Performance Requiremtnts	2
Design Constrains, Software System Attributes	2.5
Alloy model	30
Total	45

- **Davide Perugini**

Section	Hours
Purpose, Scope	5
Definitions, Document Structure	3
Product Perspective and Functions	10
Assumptions, dependecies and constraints	5
External Interface Requirements	2
Functional Requirements	2
Performance Requiremtnts	2
Design Constrains, Software System Attributes	2
Alloy model	4
Total	35

- **Stefano Panzeri**

Section	Hours
Purpose, Scope	3
Definitions, Document Structure	1
Product Perspective and Functions	2
Assumptions, dependecies and constraints	3
External Interface Requirements	4
Functional Requirements	16
Performance Requiremtnts	2
Design Constrains, Software System Attributes	3
Alloy model	4
Total	38

