POLITECNICO
MILANO 1863

Academic year 2019-2020
Software Engineering 2 Project

SafeStreets

RASD

Requirement Analysis and Specification Document

Version 1.2

Authors:
Marcer Andrea - 941276
Marchisciana Matteo - 945878
Motta Dennis - 940064

Professor:
Rossi Matteo

1. Introduction

1.1. Purpose

1.1.1. General purpose

The idea behind this product is to give to the citizen the possibility to report traffic violations he sees during the day, to the authorities. Normally, the citizen would stop, call the non emergency police number, give information about current location, type of violation, brand of the car, license plate number, ecc., spending a lot of time on the phone just to report a violation. Our purpose is to make all of this quicker and easier. With our application, all that the user must do is snap a few pictures of the car in violation including the license plate and send the report to our system. In turn, the system will send the report to the municipality that operates in the corresponding city. We believe that reducing the effort will lead to an increase in the number of reports and ticket issued, reducing overall traffic violations.
Furthermore, the product will have an additional function: we want to give the citizens and municipalities the opportunity to mine the information regarding our data and that of the municipalities. Both the citizens and the municipalities will be able to see and filter violations occurred on the map of the city.
Finally the system will be able to give the municipality some suggestions on how to improve the condition of the roads. These suggestions will be based on the number of similar violations reported in the proximity of a specific area. For example, if in a certain area a lot of cars park on the street because there are not enough parking lots the system will suggest to increase their number or redesign them in a more space efficient way.

1.1.2. Goals

- [G1] A citizen can report a violation.
- [G2] A violation report received by the system must have enough information to be valid, i.e. has at least one picture of the violation, exactly one GPS position, exactly one timestamp, exactly one type of violation and the license plate of the vehicle.
- [G3] Users and municipality can retrieve information about violations, accidents and issued tickets in a certain area, with different levels of visibility.
- [G4] Municipality will be able to retrieve suggestions for possible interventions in order to increase safety.
- [G5] Municipality receives enough information about the violation in order to issue a ticket.
- [G6] The integrity of the violation report is guaranteed.
- [G7] Municipality can visualize statistics about the violations in its territory and the effectiveness of the SafeStreets initiative.

It is important to notice that all the stakeholders' needs are portrayed in the goals, in fact every function expressed by the stakeholders in the Project Assignment can be fulfilled by some of the goals as shown in the table.

		Goals						
		G1	G2	G3	G4	G5	G6	G7
Functions expressed by the stakeholders in the Project Assignment	Basic service	X	X	X				
	Advanced function 1			X	X			
	Advanced function 2			X		X	X	X

1.2. Scope

The S2B will interact with both citizens and the municipality. To do so, it will offer to the first ones a mobile application, that can be easily downloaded from the major mobile app stores, and for the second ones a web graphical user interface.
The main feature offered to the citizens will be to report a violation directly from the application, reducing the usual time of reporting.
The municipality will have the role of checking the correctness of the violation reports submitted to the system and will have the possibility to retrieve possible suggestions in order to increase the safety of the streets under its control.

WORLD PHENOMENA	**SHARED PHENOMENA**	**MACHINE PHENOMENA**
A traffic violation occurs	A report of a violation is received	The system processes the information regarding a violation
A citizen sees a violation	A user signs up	A violation report is saved
A traffic ticket is issued	Information about issued tickets is retrieved from municipality	A query on data about violations, issued tickets or accidents is processed
	A query on data about violations, issued tickets or accidents is received	Information about accidents is crossed with information about violations to suggest possible interventions
	Information about accidents is retrieved from the municipality	

1.3. Definitions, acronyms, abbreviations

1.3.1. Definitions

- Entity: a “natural person”, or a “juridical person” in the case of the municipality.
- User: a citizen registered to the SafeStreets service.
- Municipality: a city or town that has corporate status and local government.
- Timestamp: a representation of date and time.
- Anonymized data: data that don't give any personal information, such as license plate, pictures and names.
- Valid violation report: a violation report composed by at least one picture, exactly one location, exactly one timestamp, exactly one type of violation and the license plate of the vehicle.
- Approved violation report: a valid violation report that may represent a correct violation report.
- Correct violation report: an approved violation report that the municipality evaluated as a traffic violation.
- Area: a district or a neighborhood of a country or city, especially one characterized by a particular feature or activity.
- GPS fix: a position derived from measuring in relation to GPS satellites.

1.3.2. Acronyms

- API: Application Programming Interface.
- GPS: Global Positioning System.
- UI: User Interface.
- S2B: Software To Be.
- GDPR: General Data Protection Regulation.
- OS: Operating System.

1.3.3. Abbreviations

- Gn: nth goal.
- Dn: nth domain assumption.
- Rn: nth requirement.

1.4. Revision history

- Version 1.0: Initial release.
- Version 1.1:
  - Corrected typo in a comment of alloy about municipality access to data;
  - Added a scenario and a use case for statistics;
  - Added goal 7, requirement 20 and 21;
  - Updated traceability matrix;
  - Updated effort spent.

- Version 1.2:
    - Updated table of contents;
    - Updated municipality uses cases;
    - Updated picture in section 2.1.

1.5. Reference documents

- Specification Document: “SafeStreets Mandatory Project Assignment.pdf”.
- ISO/IEC/IEEE 29148:2011: Recommended Practice for Software Requirements Specifications.
- Alloy documentation: http://alloy.lcs.mit.edu/alloy/documentation/quickguide/seq.html
- IEEE Software Requirements Specification Template from Dalhousie University: https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

1.6. Document structure

Chapter 1: Introduction
In this chapter we describe informally the idea behind the product, we define the goals to be reached along with the most important phenomena involving the S2B and finally we define some technical terms and acronyms we are going to use.

Chapter 2: Overall description
This chapter is focused on describing the product and its functionalities. In particular, we describe the context and origin of the product, we explain in detail the two main functions of the product, we identify the actors of our system and our domain assumptions. Furthermore, we provide some UML Diagrams and a flowchart, to better explain some characteristics of the system and to analyze how crucial entities of the S2B evolves.

Chapter 3: Specific requirements
In this chapter some real-life scenarios are analyzed in order to identify the use classes and the relative use cases. To better explain the interaction of such classes and the system some sequence and uses cases diagram are provided.
We then provide some insight about the possible User Interface through mockups and we illustrate the domain assumptions and constraints that are needed to satisfy each goal. Finally, we show some attributes and constraints of our system.

Chapter 4: Formal analysis using Alloy
We created a logical model of our system so that the most critical aspects of the system can be guaranteed.

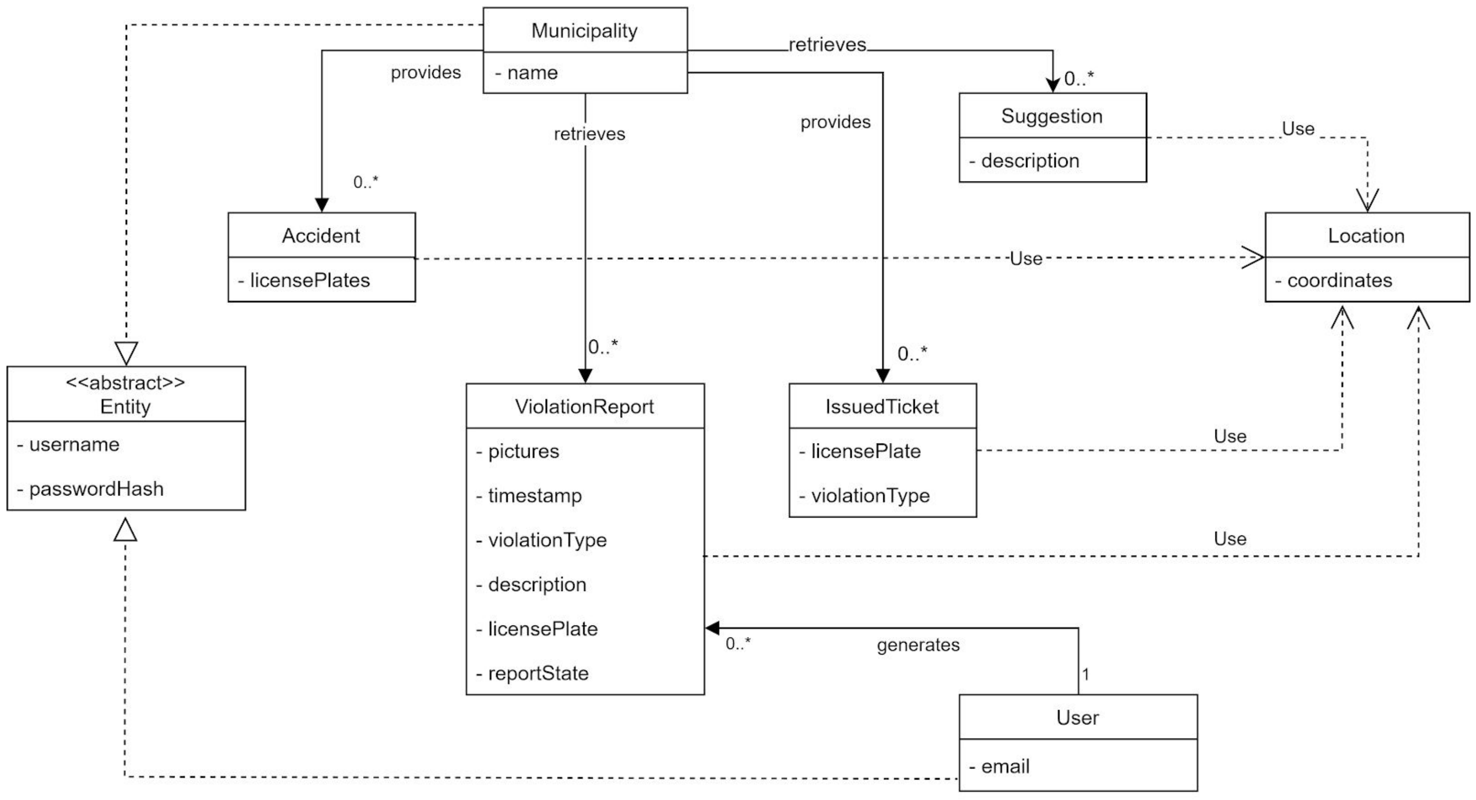
Chapter 5: Effort spent
Here we show the effort spent by each team member working on this document.

2. Overall description

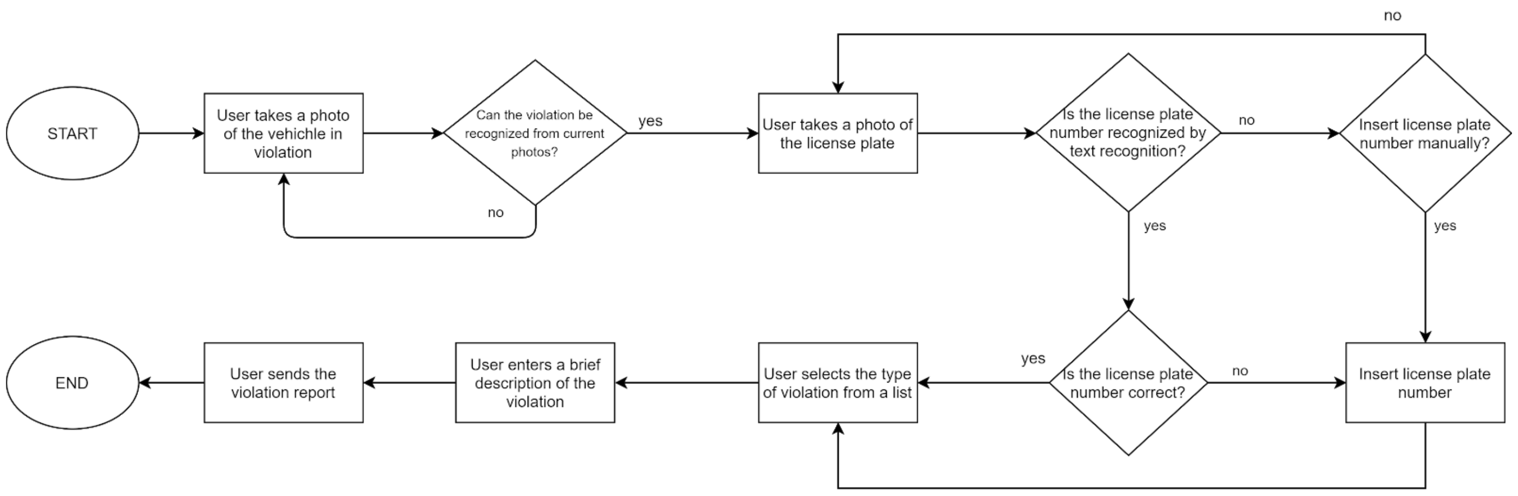
2.1. Product perspective

SafeStreets is a crowd-sourced application that relies on the contribution of all citizens to gain useful data that can be exploited to make the streets a safer place. Given the widespread use of mobile phones today, SafeStreets will be released as a mobile application. The average user will be a normal citizen, concerned for his well-being and that of others.

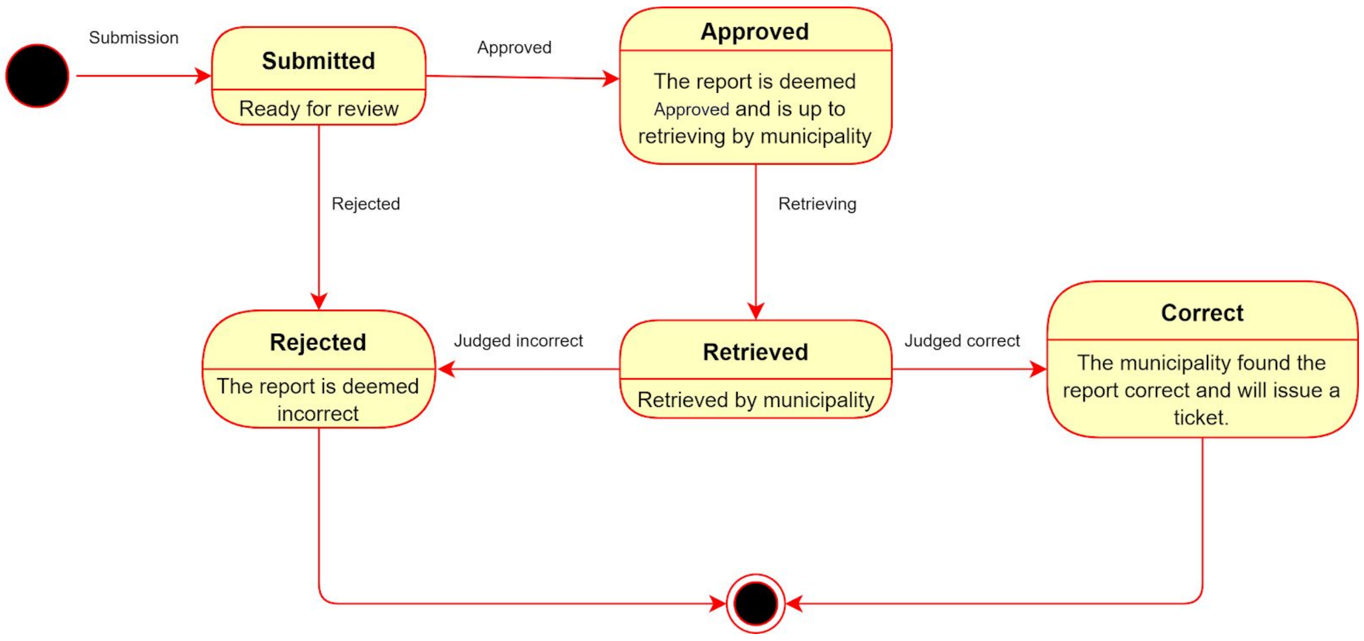
The following class diagram is a model of the application domain: as a high-level representation, it contains only a few essential attributes and does not include every class that will be necessary to the system.
The user can generate a violation report that can later be retrieved by the municipality in order to check its correctness. Furthermore, The municipality provides to the system information about accidents and its issued tickets. In this way the S2B have all the information it needs to elaborate different suggestions that can be retrieved by the municipality.



The process of generating a violation report is described in the following flowchart. The user can take one or more pictures of the violation. The application will be equipped with text recognition in order to recognize the license plate number; however, if the picture taken has low quality, is blurry or the recognition doesn't work, the user will have the possibility to type the license plate number manually.


START
User takes a photo of the vehichle in violation
Can the violation be recognized from current photos?
yes
no
User takes a photo of the license plate
Is the license plate number recognized by text recognition?
no
yes
Insert license plate number manually?
no
yes
Is the license plate number correct?
yes
no
Insert license plate number
User selects the type of violation from a list
User enters a brief description of the violation
User sends the violation report
END

During its existence, the violation report can assume many different states. This is because we would like to be transparent and show the user what are the consequences of his effort. The following is a UML State Diagram representing the different states that the violation report can assume.
A violation that is submitted to the system is a valid violation; that means it has all the mandatory fields are filled.
Then the system will do a preliminary scan of the reports. A report is deemed approved when it potentially represents a violation: reports with blurry pictures, photos of anything other that a violation or with incorrect information are rejected. Once the report is approved, it will be up for retrieving by the corresponding municipality. The municipality will then retrieve it and check if the approved violation report is correct; for example, it can check if the license plate really exists or if the license plate is registered to the same type of car in the picture. The municipality can then decide that the report does not represent a violation, thus rejecting it, or decide that, based on the report, a violation occurred and a ticket will be issued to the driver. In this case, the state of the report will be labeled as “correct”.


Submission
Submitted
Ready for review
Approved
Approved
The report is deemed Approved and is up to retrieving by municipality
Rejected
Retrieving
Rejected
The report is deemed incorrect
Judged incorrect
Retrieved
Retrieved by municipality
Judged correct
Correct
The municipality found the report correct and will issue a ticket.

2.2. Product functions

2.2.1. Violation reporting

The basic function for SafeStreets is Violation Reporting. A user that sees a traffic violation and wants to report it must fill all the mandatory information needed for the report. In particular, the user will need to take enough pictures of the incriminated vehicle to discern that it is indeed a traffic violation. A picture of the license plate is needed for text recognition to work but, as said above, if for any reason the text recognition fails the user will have the opportunity to type or correct the license plate number. A short description of the issue is welcomed, although not required. The system will then automatically collect the date, time, and GPS location of the user when the report is sent, so the user must have the GPS functionality enabled.
The report is then reviewed to make sure that it could effectively represent a violation. All the report that do not satisfy the requirements are discarded, the others will be available for the corresponding municipality to retrieve it.
The municipality can then retrieve the reports and evaluate them. It can decide to consider the report grave enough to issue a ticket to the driver; in this case the report is labeled as *correct*.

2.2.2. Data mining

The data about each violation sent through SafeStreets will be stored in an apposite data structure.
Since the municipalities offer services to retrieve information about accidents and tickets occurred in their area of operation, the intention is to combine these two sources of information.

The system will be able to cross and analyze this data to generate useful metainformation that will be used, for example, to search for potentially dangerous roads or to identify unsafe areas.
The level of visibility will be different for the two entities: users should only see aggregate data without personal information; municipalities, on the other hand, will be able to see license plates or even query for the history of violations of a certain specified license plate number.
Furthermore, the system will be able to provide useful statistics and give suggestions about possible intervention in dangerous road, making these features available for the municipality to retrieve.

2.3. User characteristics

We identify two actors of our system.

- User: a citizen that is registered to the SafeStreets application. The user can report traffic violations and visualize them on a map.
- Municipality: a city or town that has corporate status and local government. The municipality can review reports submitted by citizens, retrieve suggestions and visualize statistics about the violations in its territory and the effectiveness of the SafeStreets program

2.4. Assumptions, dependencies and constraints

2.4.1. Domain assumptions

- [D1] A citizen who wishes to report a violation has a mobile phone with the SafeStreets app installed.
- [D2] Municipality offers a service to retrieve information about accidents.
- [D3] Municipality offers a service to retrieve information about tickets.
- [D4] When a device is able to obtain a GPS fix, the location provided has an accuracy of at least 20 meters.
- [D5] The municipality checks if approved violation reports can actually represent a traffic violation.
- [D6] Data transferred through connections that use modern encryption protocols can not be manipulated.

The meaning of D5 is that the municipality will have to do all the possible verifications, using their databases, to be sure that the violation report actually represents a violation, for example they should check if car model associated with the license plate actually corresponds to the car model in the picture.

All these domain assumptions are believed to represent valid properties about the world that can be assumed true in any case.
In particular for:

- D1: it is trivial to believe that the assumption can be assumed in any case.
- D2 and D3: while there may be cases in which these assumptions may not hold, in those cases it would be practically impossible to reach some of the goals, so they are effectively needed to be assumed in order to reach the goals.
- D4: it is quite safe to assume that a GPS fix reaches an accuracy of at least 20 meters considering that usually a GPS fix has an accuracy of ≈ 5 meters in an open area[1].
- D5: it can be assumed since it's something that the municipality must do in order to issue ticket.
- D6: it is known that modern encryption protocols are proved to be safe against cyber attacks.

2.4.2. Dependencies

The SafeStreets application will be dependant on the Operating System hosting it, so it must follow all the requirements of it.
Also the system will have to use a mapping service to perform operations with locations and visualize them.

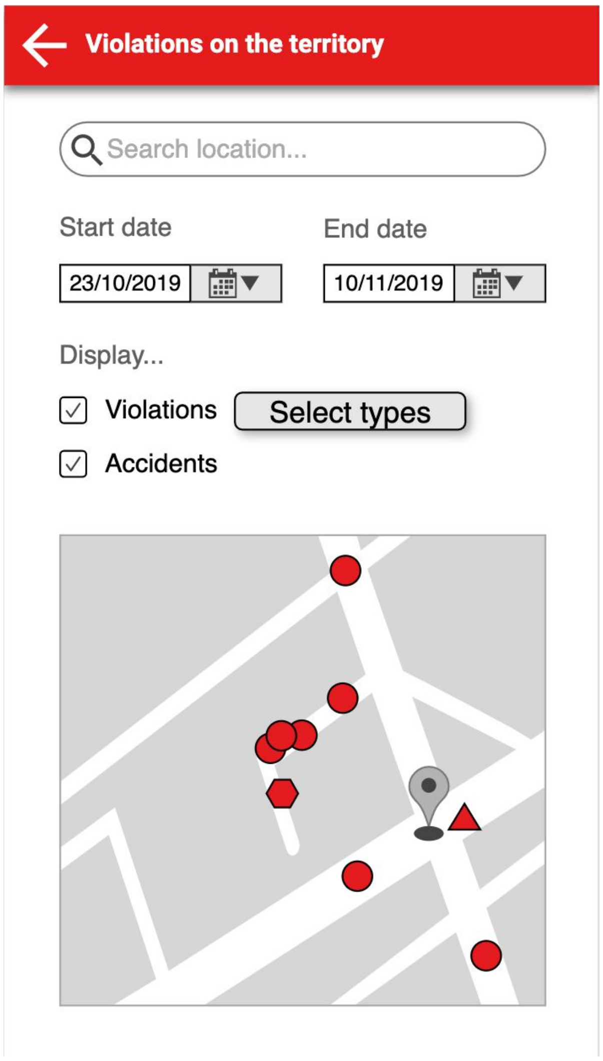
3. Specific requirements

3.1. External Interface Requirements

3.1.1. User Interfaces

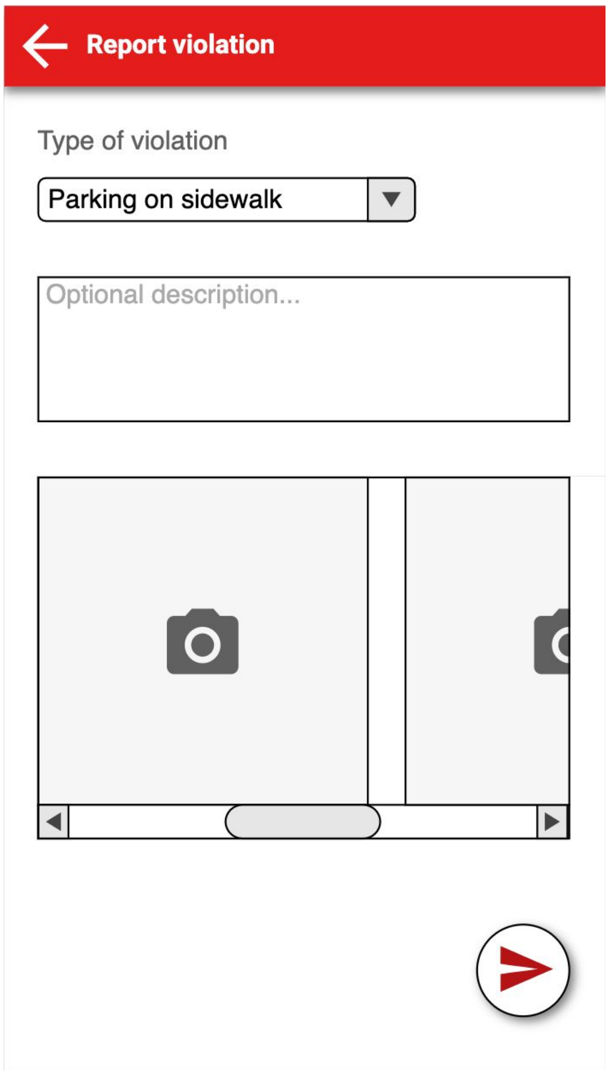
3.1.1.1. Visualization of violation reports on the application

The map in this mockup shows the violations and accidents in the system that correspond to the filters applied. The available filters for users are: location, date and type of violation.



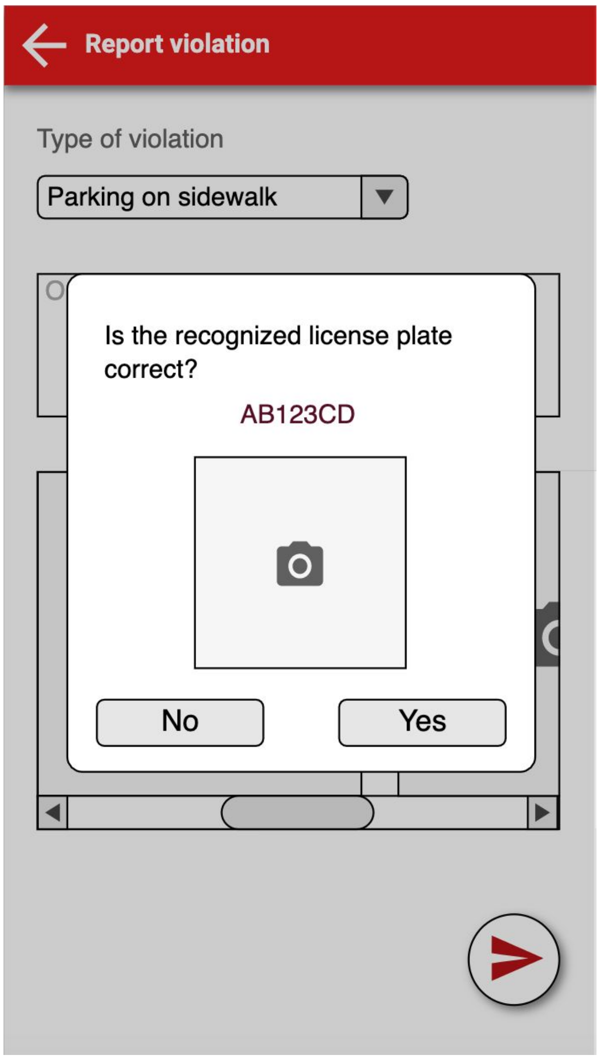
3.1.1.2. Creation of a violation report on the application

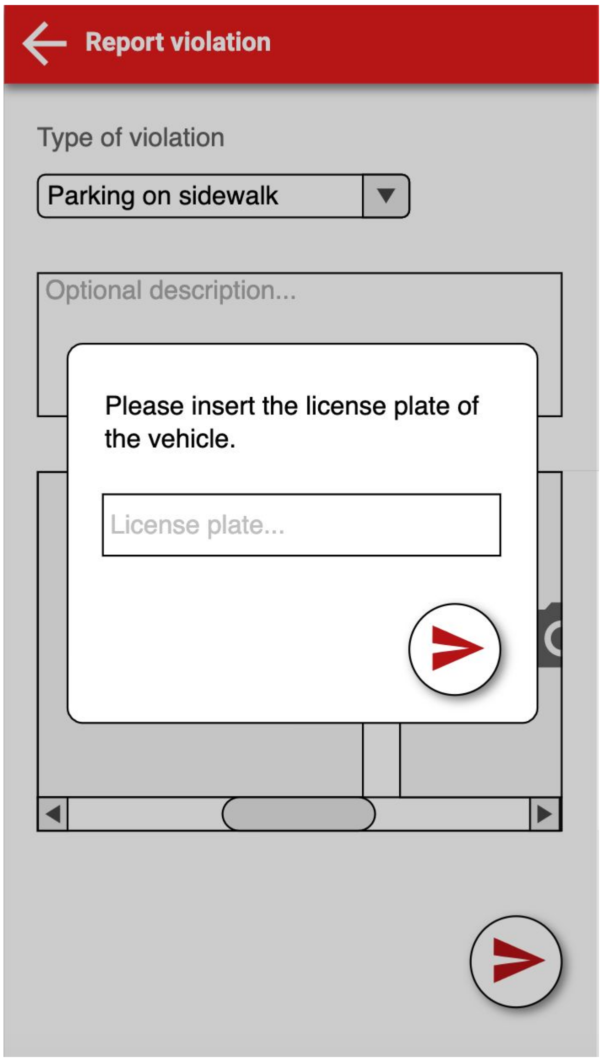
The following mockup represents what the users will see when reporting a violation. It's important to highlight some of the features present in this screen: the user can only take photos, choose the type of violation and add an optional description. The time, date and location are automatically collected by the application and can't be written by the user.



3.1.1.3. License plate recognition

When sending the violation report, the application will use a text recognition software to try to obtain the license plate of the vehicle. If the recognition fails to obtain a license plate or if the license obtained is not correct the user will have to insert it manually.





3.1.2. Hardware Interfaces

The system doesn't have any hardware interfaces.

3.1.3. Software Interfaces

The system will provide a graphical user interface that will be used by municipalities to retrieve violation reports and check whether or not they are correct. This solution has been adopted because it is easier to use, allowing the municipality to concentrate on more pressing matters. Nevertheless, the system will provide useful APIs, so the municipality will have the possibility to integrate our services into their already existing system.

The system will also use APIs provided by the municipalities to retrieve issued ticket and accidents occurred in the territory of the municipality.

3.1.4. Communication Interfaces

All the communications inside (server - application) and outside the system (system - municipality) will use standard TCP/IP network interfaces.

3.2. Functional Requirements

3.2.1. Citizen

3.2.1.1. Scenarios

Scenario 1: citizen registration

Phoebe’s mom is getting old and has difficulty walking on her own, so she needs to use a wheelchair in order to move around. Every saturday Phoebe takes her mom to the supermarket and helps her with the shopping, but most of the times the parking lot reserved for people with disabilities is occupied by cars that do not have the handicap badge. Phoebe argued with the supermarket owner, but he said that this type of problem does not bother him. One day, while she was searching for a new mobile game, Phoebe’s attention is drawn towards the suggested app section exactly on the SafeStreets app. She thinks that this application can help her with her problem. The registration is very easy and straightforward; she only needs to fill the mandatory fields: her name, her surname, her phone number, a username of her choice and a password. Now she is ready to report anyone that could prevent her to spend some time with her beloved mother.

3.2.1.2. Use cases

Use Case 1: Sign up

Name	Sign up
Actor	Citizen
Entry condition	The citizen opens the application.
Event flow	1. The citizen on the first screen of the application presses the "Sign in" button. 2. The citizen provides his email and password. 3. The citizen confirms the registration by pressing on the "Sign in" button. 4. The system saves the data of the new user.
Exit condition	The citizen has become a user, from now on he/she can log in to the system using the username and password provided during his/her sign in process.
Exceptions	1. The email is already associated with another account: the citizen is asked to insert another one. 2. Some information is not valid: the invalid fields are highlighted and it is suggested to the citizen to insert valid information. 3. The citizen does not fill all the mandatory fields: the missing fields are highlighted and it is suggested to the citizen to fill them.

3.2.2. User

3.2.2.1. Scenarios

Scenario 1: safe areas

Rachel and Ross are expecting a baby, but their current apartment is too small for them all, so they decided to buy a new one in the center of their town. Since they want the best for their child, they are not only searching for a comfortable apartment, but also for one situated in a safe area of the city. Fortunately they have the SafeStreets' app installed on their devices and can easily check where the safest areas are located. They can easily visualize a map that marks where each traffic violation has occurred; they can even filter by type of violation so they can look for the area where fewer accident occurred.

Scenario 2: reporting a violation, no bike parking lot

Joey cares about his health and about the environment, so he has decided to go to work by bike. Unfortunately today he turned the alarm clock off and fell asleep again. The sound of the traffic passing by his apartment wakes him up, it is 8 o'clock and his turn at the museum starts at 8:30. So he dresses up, skips his usual coffee and runs to take his bike. On his way to work he notice that the cycle lane is blocked by a parked yellow car. It is not the first time that this happens, but calling the police would take too much time. Fortunately Joey downloaded and signed up to SafeStreets. Now it is very easy for Joey to report the violation, all it takes is logging in, taking a photo of the car with the license plate well visible and fill the field that specifies the type of violation. Joey arrives in time to the museum and he knows that the owner of the car will receive a ticket and hopefully will learn the lesson.

3.2.2.2. Use cases

Use Case 2: Log in of a user

Name	Log in of a user
Actor	User
Entry condition	The user opens the application.
Event flow	1. The user presses the “Sign in” button 2. The user inserts his/her email and password in the corresponding fields. 3. The user presses the “Login” button. 4. The system logs the user in.
Exit condition	The user is now able to report a violation, visualize his/her own violation and the map.
Exceptions	1. The email is not associated with any account: the user is notified and he/she is asked to insert a valid email. 2. The password is incorrect: the user is notified and he/she is asked to insert the password again.

Use Case 3: Report violation

Name	Report violation
Actor	User
Entry condition	The user is logged in.
Event flow	1. The user chooses the option to report a new violation. 2. The user takes at least one photograph of the violation within the application. 3. The user writes the description. 4. The user selects the type of violation from a given list. 5. The user presses the “Send” button. 6. The system retrieves additional information about the report. 7. The system saves the report data. 8. The system retrieves and checks the consistency of the meta-information. 9. It is communicated to the user that the report has been received correctly.
Exit condition	The violation is saved in the system as “submitted” and it’s ready to be reviewed.
Exceptions	1. The application cannot retrieve the license plate from one of the photos: the application asks the user to insert it manually. 2. Some of the meta-information of the violation report are incorrect: the user is notified and is asked to correct them. 3. The application cannot access the camera: the user is notified about the problem and the submission of the violation is denied. 4. The application cannot access the GPS location: the user is notified about the problem and the submission of the violation is denied.

Use Case 4: Filter violations on the map

Name	Filter violations on the map
Actor	User
Entry condition	The user is logged in.
Event flow	1. The user chooses to visualize the map. 2. The application shows the map of the city near him/her retrieving information from the GPS. 3. The user chooses how to filter the violations.
Exit condition	The user can navigate the map and visualize the aggregated information.
Exceptions	1. The GPS location cannot be retrieved: the default location is used.

3.2.3. Police officer

3.2.3.1. Scenarios

Scenario 1: suggestions for the municipality

Chandler and his friends every Friday meet up in their favorite pub in town. Chandler for convenience has bought an electric motorcycle, but the parking lots dedicated for his vehicles are always occupied by some cars. While discussing about this problem with his friends one of them suggests to Chandler to download SafeStreets and report the car ifever it would happen again. Chandler does as suggested by his friends and the next week starts reporting every violation. The police officer that checks the suggestions given by SafeStreets notice the high number of reports done by Chandler and decides that an intervention is required. Only a few weeks after Chandler arrives at the pub and with great surprise he notices that no car is parked on the motorcycle parking lots thanks to some barriers. He is so happy that offers a round of beers to all of his friends.

Scenario 2: critical areas

As a new directive, all policemen in charge of issuing tickets in the city have to look up for the areas where recently there have been a higher number of reports. So, before going out for his turn, Bob logs in the web interface offered by SafeStreets and can easily see which are the most critical areas, i.e. areas with a high number of reports. In this way he is able to increase the efficiency of his work.

Scenario 3: violation acceptance

Richard works at the municipality. He used to work at the issued ticket archive, where he needed to sort every ticket by hand verifying that the license plate corresponded to the correct car model using a very old interface with the local database. Now, thanks to the web interface of SafeStreets, he does not need to sort anything by hand anymore. Richard simply logs in to the web interface which loads and visualizes on the screen the list of all the approved violation reports that need to be confirmed. The only thing Richard needs to do is verify that the license plate is valid and it matches with the car's model and then press the "Confirm" or "Reject" button.

Scenario 4: statistics

Mr. Heckles is the eldest police officer in the municipality. He doesn't like changes and he was firmilly against the introduction of SafeStreets in his town. Nevertheless the major decided that SafeStreets would help reduce violations and agreed to register his town to the service. Now Mr. Heckles sits behind a desk confirming or rejecting violation reports, but every time he logs in he can see the statistics and after some months he can clearly see that the number of reports and issued tickets has decreased. Afterall it wasn't a bad idea.

3.2.3.2. Use cases

Use Case 5: Log in for a police officer

Name	Log in for a police officer
Actor	Police officer
Entry condition	The police officer connects to the web interface.
Event flow	1. The police officer inserts his/her username and password. 2. The police officer presses the "Login" button. 3. The system logs the user in.
Exit condition	The police officer can confirm violations, visualize the map of violations or check for suggestions.
Exceptions	1. The username is not associated with any account: the police officer is notified and he/she is asked to insert a valid email. 2. The password is incorrect: the police officer is notified and he/she is asked to insert the password again.

Use Case 6: Verify reports

Name	Verify reports
Actor	Police officer
Entry condition	The police officer is logged in.
Event flow	1. The police officer chooses the option to verify the violations. 2. The system shows on the screen a list of approved violation reports. 3. The police officer chooses a violation report to verify. 4. The system shows in detail all the information of the violation report. 5. The police officer checks the correctness of the violation type, the license plate and the car model. 6. The police officer presses the “Confirm” or “Reject” button accordingly to what he/she has found.
Exit condition	The report is labeled accordingly to what the police officer has chosen.
Exceptions	There are no exceptions.

Use Case 7: Receive suggestions

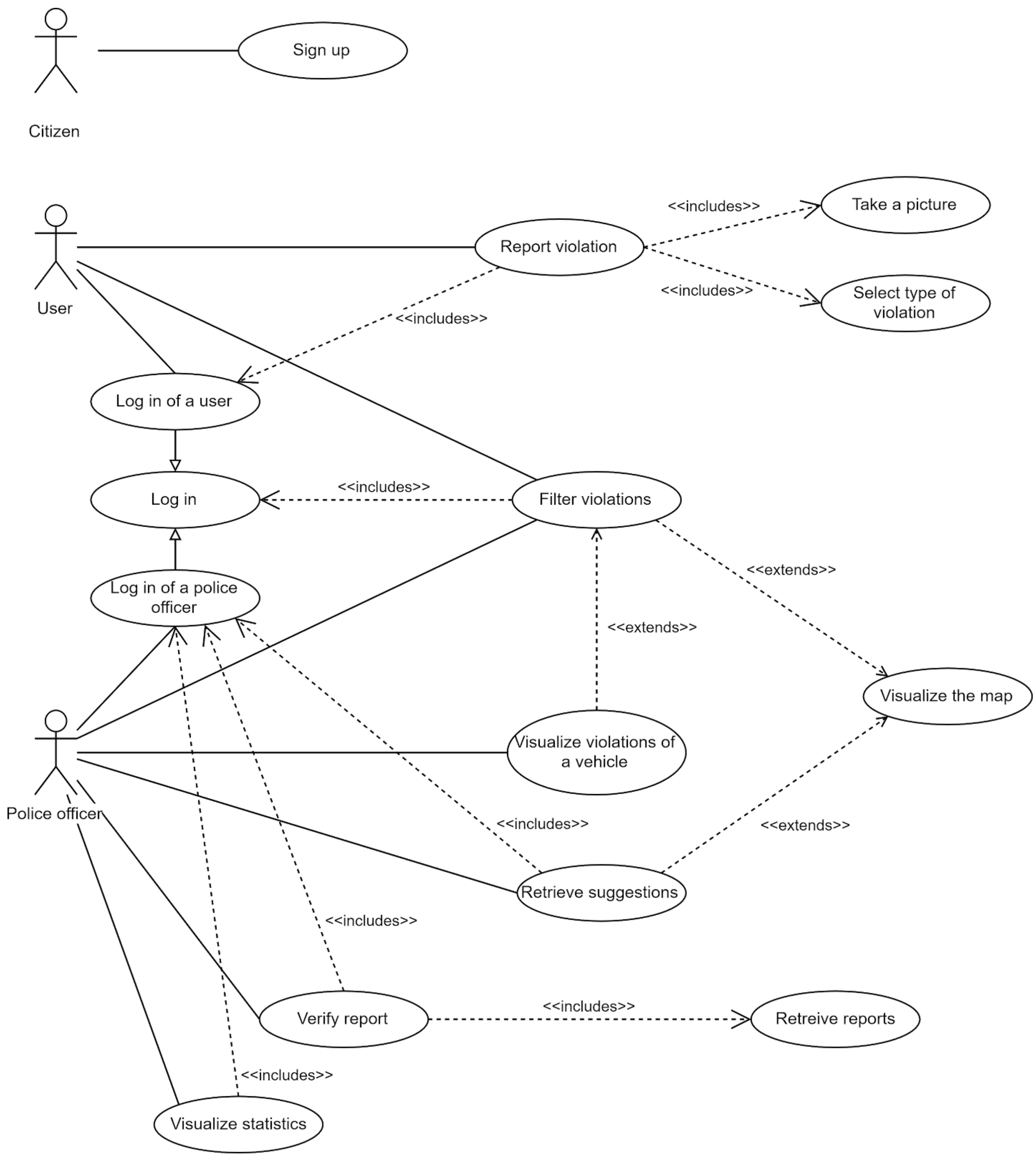
Name	Receive suggestions
Actor	Police officer
Entry condition	The police officer is logged in.
Event flow	1. The police officer chooses to visualize the suggestions. 2. The system visualizes the map of the corresponding municipality. 3. The system provides the information about the violations and the corresponding suggestions.
Exit condition	The police officer can explore the map and search for more suggestions.
Exceptions	There are no exceptions.

Use Case 8: Filter violations on the map

Name	Filter violations on the map
Actor	Police officer
Entry condition	The police officer is logged in.
Event flow	1. The police officer chooses to visualize the map. 2. The system shows the map of the city controlled by the municipality visualizing reports. 3. The police officer inserts the license plate of the vehicle. 4. The system visualizes all the reports concerning the specified license plate.
Exit condition	The police officer can visualize all the information regarding the violations and accidents concerning the specified license plate.
Exceptions	There are no exceptions.

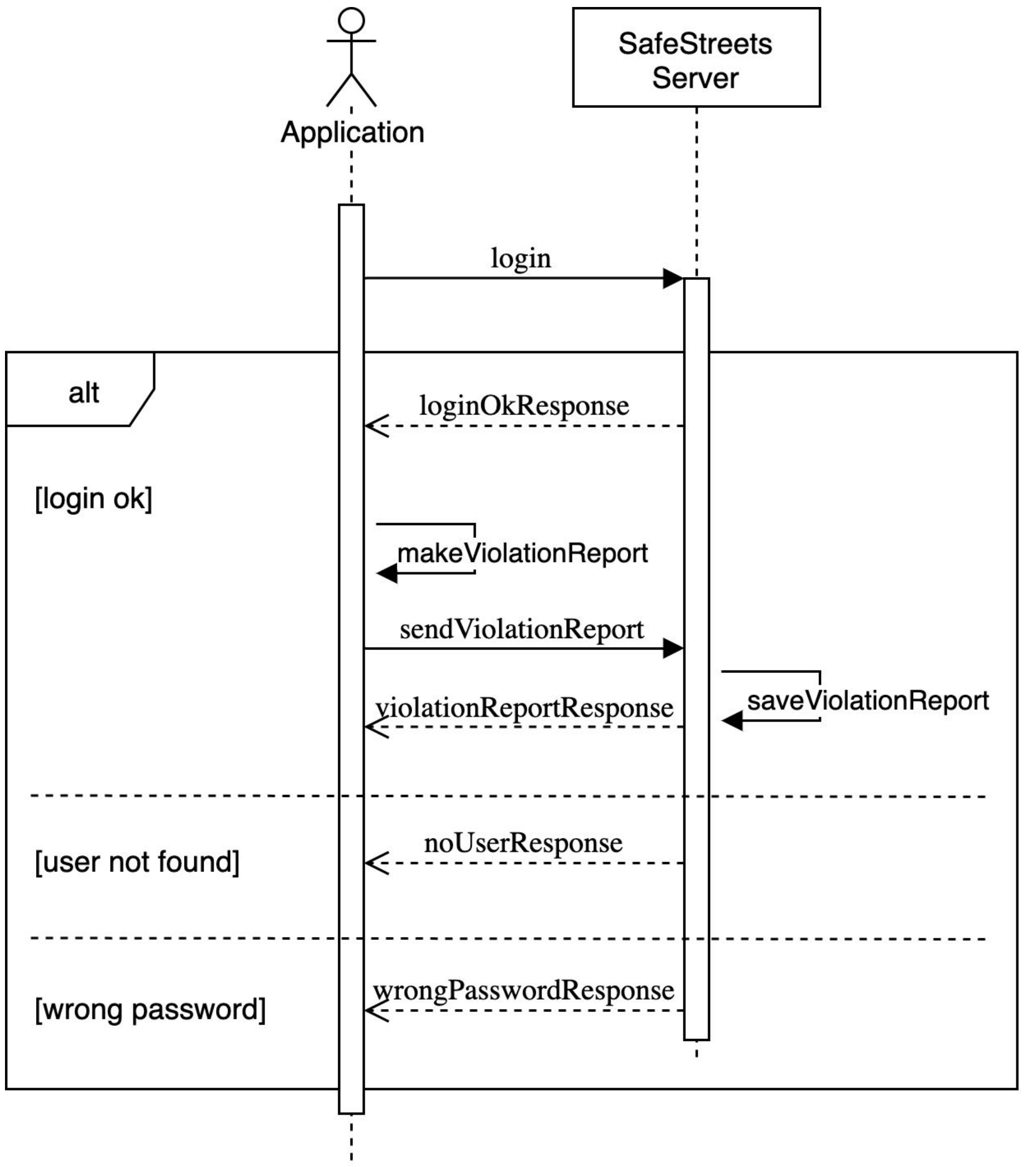
Use Case 9: Visualize statistics

Name	Visualize statistics
Actor	Police officer
Entry condition	The police officer is logged in.
Event flow	1. The police officer chooses to visualize the statistics. 2. The system retrieves the statistics and visualizes them.
Exit condition	The police can see the statistics.
Exceptions	There are no exceptions.

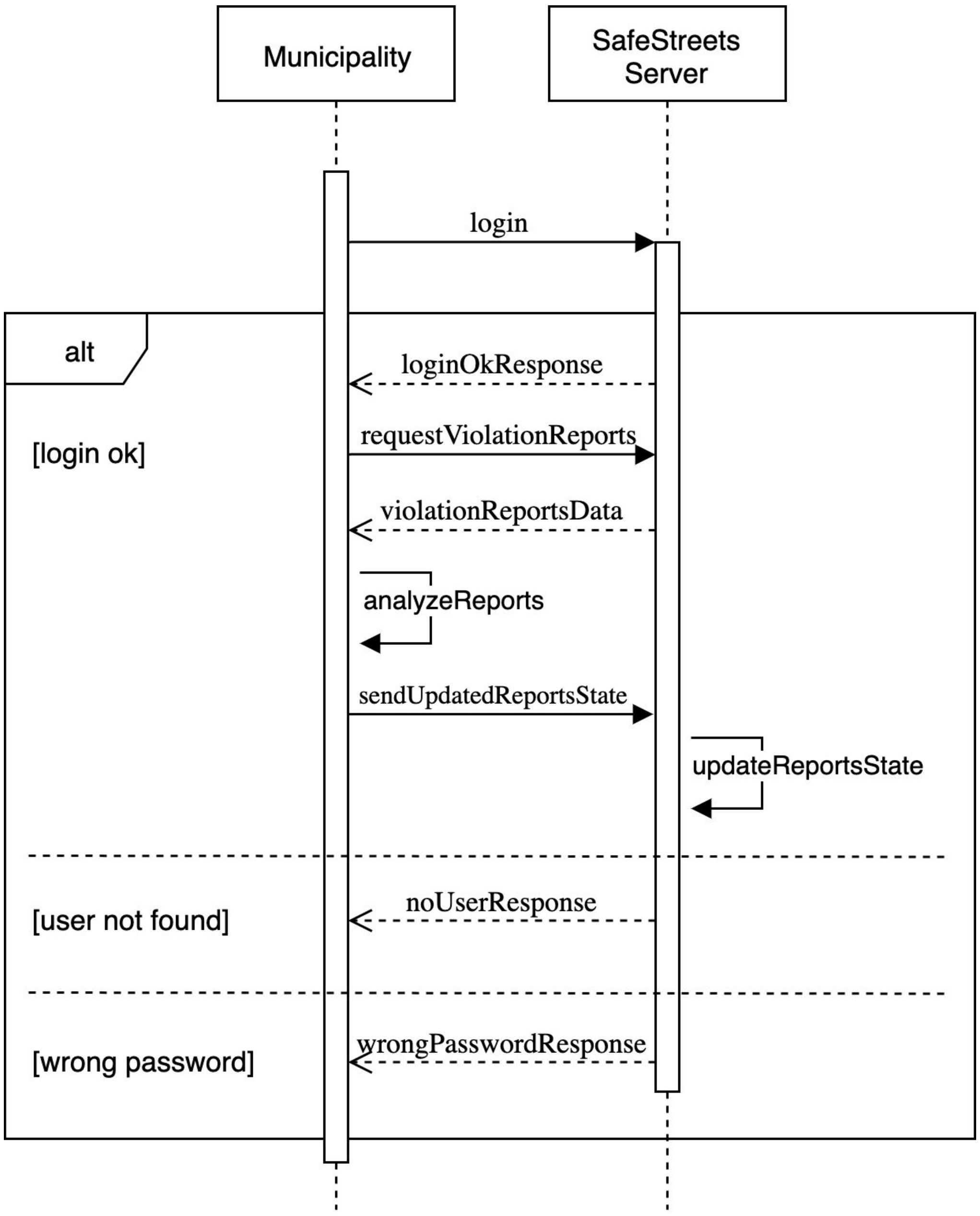

Citizen
Sign up
User
Report violation
<<includes>>
Take a picture
<<includes>>
Select type of violation
<<includes>>
Log in of a user
Log in
<<includes>>
Filter violations
Log in of a police officer
<<extends>>
<<extends>>
Visualize the map
Visualize violations of a vehicle
Police officer
<<includes>>
<<extends>>
Retrieve suggestions
<<includes>>
Verify report
<<includes>>
Retreive reports
<<includes>>
Visualize statistics

3.2.3. Sequence diagrams

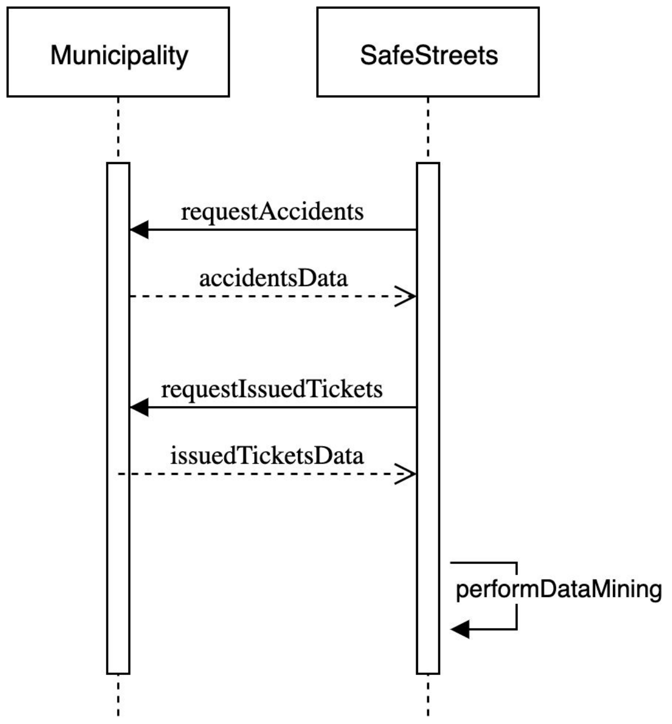
3.2.3.1. User login and violation reporting



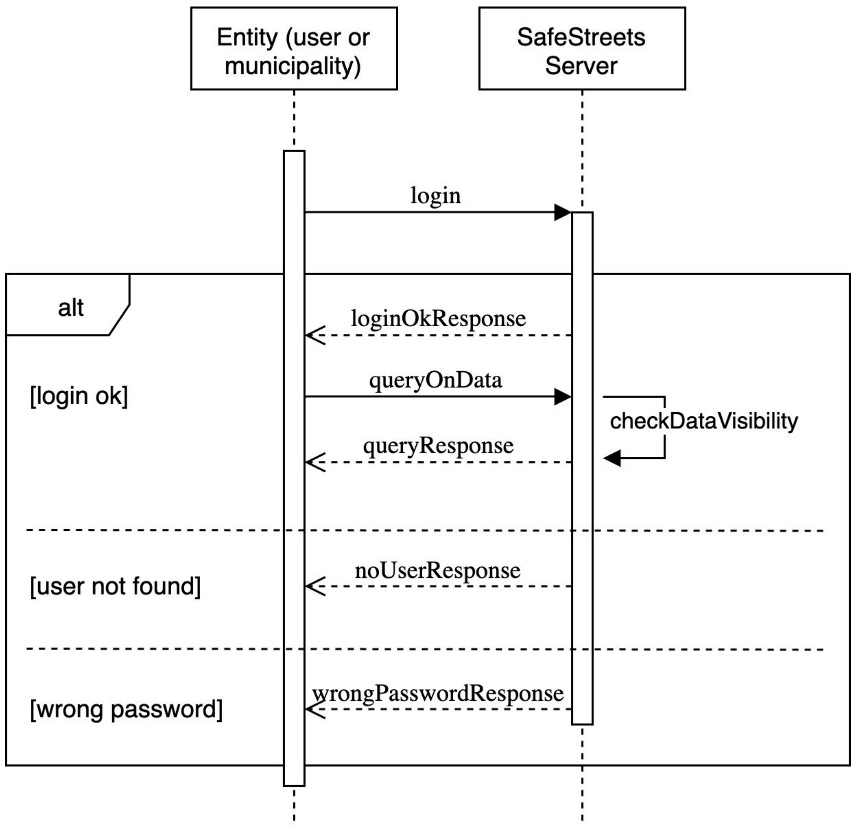
3.2.3.2. Municipality login and reports retrieving



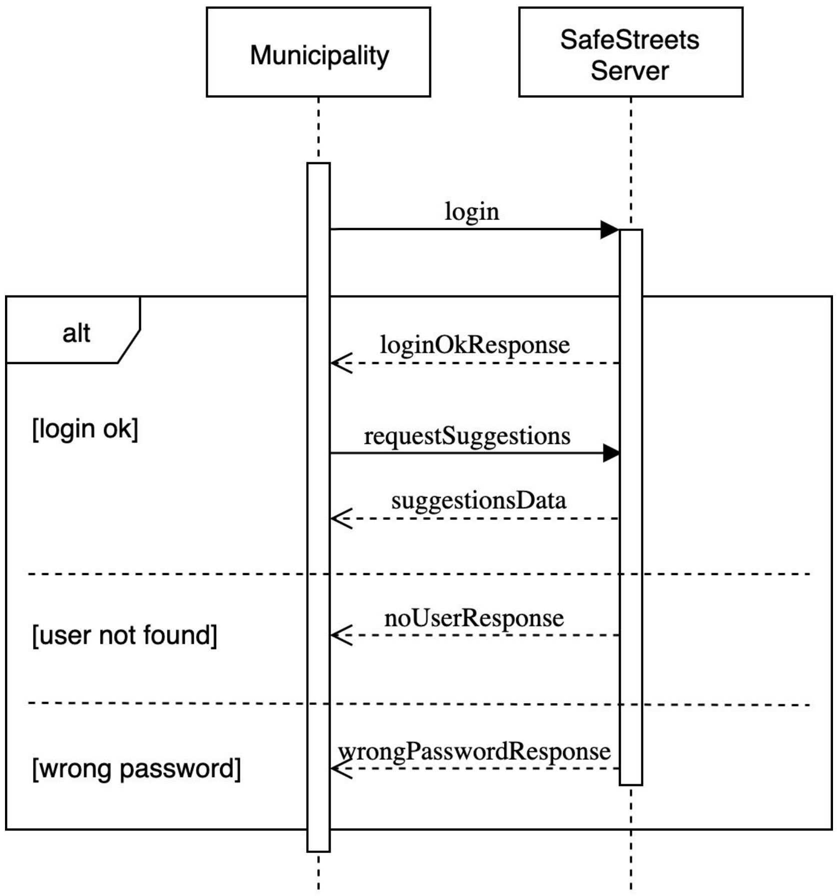
3.2.3.3. Data mining: municipality's data retrieving



3.2.3.4. Data mining: visualization



3.2.3.5. Data mining: suggestions



3.2.4. Requirements

The objective of this paragraph is to show the completeness of the specified requirements.
To reach this objective three steps must be performed:

- Goals must adequately capture all the stakeholders' needs. This has been proved in the section "1.1.2. Goals".
- Domain assumption must be valid properties about the world. This has been proved in section "2.4.1. Domain assumptions".
- Requirements must ensure satisfaction of the goals given the context of the domain assumptions, or written in logic terms: Requirements, Domain Assumptions ⊨ Goals. This is what is going to be proved here by showing in an informal manner which goal is satisfied by certain requirements and domain assumptions.

3.2.4.1. Satisfying goal 1

[G1] A citizen can report a violation.

- [D1] A citizen who wishes to report a violation has a mobile phone with the SafeStreets app installed.
- [R1] A citizen not yet registered must be able to sign up and become a user.
- [R2] The application must allow users to authenticate through log in.

- [R3] The application must allow users to report a violation.

3.2.4.2. Satisfying goal 2

[G2] A violation report received by the system must have enough information to be valid, i.e. has at least one picture of the violation, exactly one GPS position, exactly one timestamp, exactly one type of violation and the license plate of the vehicle.

- [R4] The application must allow reporting of violations only from devices equipped with a GPS receiver which are in the conditions to obtain a GPS fix.
- [R5] The application must allow reporting of violations only from devices equipped with a camera.
- [R6] The application must allow reporting of violations only from devices with an active internet connection.
- [R7] A user has the possibility to specify the type of the reported violation choosing from a list.
- [R8] The application creates a violation report with at least one picture, exactly one timestamp, exactly one location, exactly one type of violation and the license plate of the vehicle.

3.2.4.3. Satisfying goal 3

[G3] Users and municipality can retrieve information about violations, accidents and issued tickets in a certain area, with different levels of visibility.

- [D2] Municipality offers a service to retrieve information about accidents.
- [D3] Municipality offers a service to retrieve information about tickets.
- [R9] The system saves all the information regarding the violations reported by users.
- [R10] The system must be able to retrieve data regarding issued tickets and accidents from the municipality.
- [R11] The system must offer an interface to retrieve information about violations, accidents and issued tickets.
- [R12] The system must show to users only anonymized data.
- [R13] The system must analyze valid violations report and approve which of them may represent a correct violation.

3.2.4.4. Satisfying goal 4

[G4] Municipality will be able to retrieve suggestions for possible interventions in order to increase safety.

- [D2] Municipality offers a service to retrieve information about accidents.
- [D3] Municipality offers a service to retrieve information about tickets.
- [R9] The system saves all the information regarding the violations reported by users.
- [R10] The system must be able to retrieve data regarding issued tickets and accidents from the municipality.

- [R14] The system must be able to elaborate data about violations, accidents, issued tickets and generate useful suggestions about possible interventions.
- [R15] The system must offer an interface to the municipality for retrieving useful suggestions about possible interventions.

3.2.4.5. Satisfying goal 5

[G5] Municipality receives enough information about the violation in order to issue a ticket.

- [D4] When a device is able to obtain a GPS fix, the location provided has an accuracy of at least 20 meters.
- [D5] The municipality checks if approved violation reports can actually represent a traffic violation (for example they could check if the license plate actually corresponds to the car model in the picture).
- [R8] The application creates a violation report with at least one picture, exactly one timestamp, exactly one location, exactly one type of violation and the license plate of the vehicle.
- [R13] The system must analyze valid violations report and approve which of them may represent a correct violation.
- [R16] The system must offer a service to the municipality for retrieving approved violations report.

3.2.4.6. Satisfying goal 6

[G6] The integrity of the violation report is guaranteed.

- [D6] Data transferred through connections that use modern encryption protocols can not be manipulated.
- [R17] The application will allow using pictures in a violation report only if the picture was taken by the application itself, preventing it to be manipulated on the device.
- [R18] All connections used by the system use modern encryption protocols.
- [R19] Data saved in the server can not be manipulated.

3.2.4.7. Satisfying goal 7

[G7] Municipality can visualize statistics about the violations in its territory and the effectiveness of the SafeStreets initiative.

- [D2] Municipality offers a service to retrieve information about accidents.
- [D3] Municipality offers a service to retrieve information about tickets.
- [R9] The system saves all the information regarding the violations reported by users.
- [R10] The system must be able to retrieve data regarding issued tickets and accidents from the municipality.

- [R20] The system must be able to elaborate data about violations, accidents, issued tickets and generate useful statistics for each municipality.
- [R21] The system must offer an interface to the municipality for retrieving statistics.

3.2.5. Traceability matrix

	Use cases									Goals						
	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	G1	G2	G3	G4	G5	G6	G7
R1	x									x						
R2		x								x						
R3			x							x						
R4			x								x					
R5			x								x					
R6			x								x					
R7			x								x					
R8			x			x					x			x		
R9			x	x				x	x			x	x			x
R10				x		x	x	x	x			x	x			x
R11				x	x			x				x				
R12				x								x				
R13				x		x		x				x		x		
R14							x						x			
R15					x		x						x			
R16						x								x		
R17			x												x	
R18			x												x	
R19			x			x									x	
R20									x							x
R21					x				x							x

3.3. Performance Requirements

The system must have a scalable performance since its load is expected to undergo a lot of variations during the day. An expected real time load during peak times (6:00-10:00 and 16:00-20:00) could be approximated to < 2% of citizens living in the municipality, for every municipality. During the night the load is expected to decrease drastically.

3.4. Design Constraints

3.4.1. Standards compliance

To guarantee compatibility to other systems and to ensure that the best practices are followed the system and the organization will have to use specific standards:

- ISO 6709[2] - “Standard representation of geographic point location by coordinates”: for the format of the GPS location.
- ISO/IEC 27001[3] - “Information security management”: provides best practices for managing sensitive company information so that it remains secure.
- ISO/IEC 27701[4] - “Security techniques”: provides best practices for implementing, maintaining and continually improving a Privacy Information Management System.

The system must also be compliant to the General Data Protection Regulation (GDPR), and by complying with the ISO/IEC 27001 and ISO/IEC 27701 standards the system will meet the privacy and information security requirements of the aforementioned regulation[5].

3.4.2. Hardware limitations

Some hardware requirements need to be met in order to fully take advantage of the functionalities offered by SafeStreets:

3.4.2.1. Data mining - hardware limitations

The “Data mining” functionality needs an active internet connection with a reasonable data bandwidth.

3.4.2.2. Violation reporting - hardware limitations

There are stricter hardware constraints needed to be met in order to make a violation report. A device used to perform such functionality must have:

- An active internet connection with a reasonable data bandwidth.
- A GPS receiver.
- A camera.

These constraints are needed to create a valid violation report.

3.4.3. Any other constraint

The system will be subject to other constraints due to the presence of the communication between the system itself and the municipalities; in order to prevent an overload of the municipality's server, the system must make a reasonable amount of requests to the interfaces.
Also the SafeStreets application must follow all the requirements of the Operating Systems hosting it.

3.5. Software System Attributes

3.5.1. Reliability

To avoid faults and to reach a high reliability the S2B will be built using defensive programming techniques.
A defensive approach has been already started here, at the requirement level, by making domain assumptions that are very plausible and can actually be assumed all the time.

3.5.2. Availability

The system is not safety critical so it is possible to tolerate not so high levels of availability. A 3-nines availability (≈9 hours of downtime every year) can be enough.

3.5.3. Security

A main concern of the system is to always ensure that the information provided to the municipality, which will be used to issue tickets, cannot be manipulated in any way. No one should be able to use the system to create fraudulent tickets.
Another still important concern is to always assure that the system cannot be compromised in any way that could alter its behavior, for example when reporting data about violations or when suggesting possible interventions to the municipality.
Furthermore to be complied to the GDPR, user credentials and data need to be protected from external access with appropriate security measures.

3.5.4. Maintainability

The software must be written in a way that would make it easy to maintain and update; to achieve this objective components and interfaces must be specified at the most abstract level possible and appropriate design patterns must be used.

3.5.5. Portability

Portability is a main concern for the system since the application must be available to different mobile operating system. Also it is wise to consider that other OSs can be introduced in the next few years so the application and the system in its entirety must be ready for the case.

4. Formal analysis using alloy

In this section we analyze some critical aspects of our system using the Alloy modeling language. In particular, we identified and analyzed the following aspects:

- Goal 2: A violation report received by the system must have enough information to be valid: the report, once created, will have at least one picture, the location, timestamp, type of the violation and license plate.
- Goal 6: The integrity of the violation report is guaranteed. This means the report, since its creation, will never be in a state in which its integrity can be compromised.
- Two different entities cannot have the same username.
- Privacy must be respected for queries; queries from the municipality should be able to see personal information like license plates and pictures, whether users should not.

For the two goals that we analyzed, it has been necessary to include our requirements and domain assumptions as facts; for this reason we included R4, R5, R6, R8, R17, R18, R19, D6.

4.1. Alloy code

```
//------------ ENTITIES ------------
abstract sig Entity {
        username: one Username
}
sig User extends Entity {}
sig Municipality extends Entity {}

sig Username{}

//------------ DEVICES ------------
sig Device {
        hasGPS: one Bool,
        hasInternet: one Bool,
        hasCamera: one Bool
}

//------------ VIOLATION REPORTS ------------
sig ViolationReport {
        pictures: set Picture,
        location: lone Location,
        timestamp: lone Timestamp,
        typeOfViolation: lone TypeOfViolation,
        licensePlate: lone LicensePlate,
        state: one ViolationReportLocation,
        createdBy: one Device,
	canBeAltered: one Bool
}

sig Picture {}
sig Location {}
sig Timestamp {}
sig TypeOfViolation {}
sig LicensePlate {}

//------------ QUERIES ------------
sig ViolationReportsQuery {
	askingEntity: one Entity,
	violationReportsData: set ViolationReportDataForQuery
}

abstract sig ViolationReportDataForQuery {
	pictures: set Picture,
	location: lone Location,
	timestamp: lone Timestamp,
	typeOfViolation: lone TypeOfViolation,
	licensePlate: lone LicensePlate,
}

sig ViolationReportDataForQueryWithPrivacy extends ViolationReportDataForQuery {
} {
	#pictures = 0
	#licensePlate = 0
}

sig ViolationReportDataForQueryWithoutPrivacy extends
ViolationReportDataForQuery {
} {
	#pictures >= 1
	#licensePlate = 1
}

//------------ ENUMS ------------
abstract sig Bool {}
one sig TRUE extends Bool {}
one sig FALSE extends Bool {}

abstract sig ViolationReportLocation {}
one sig ON_DEVICE extends ViolationReportLocation {}
abstract sig ON_NETWORK extends ViolationReportLocation {}
one sig ON_NETWORK_ENCRYPTED extends ON_NETWORK {}
one sig ON_NETWORK_NOT_ENCRYPTED extends ON_NETWORK {}
one sig ON_SERVER extends ViolationReportLocation {}

// ########## GENERAL FACTS ##########

/* There could not exist two reports in the same state that share the same
picture.
```

```
 *A picture is related to only one real violation report.
*/
fact uniqueReport{
	all r1, r2: ViolationReport | (r1 != r2 and r1.state = r2.state) implies
		#(r1.pictures & r2.pictures) = 0
}

/*There could not exist any data that is not associated with a report
*/
fact allDataAreReferencedByAReport{
	all p:Picture | some r:ViolationReport | p in r.pictures
	all li:LicensePlate | some r:ViolationReport | li = r.licensePlate
	all t:Timestamp | some r:ViolationReport | t = r.timestamp
	all lo:Location | some r:ViolationReport | lo = r.location
	all tv:TypeOfViolation | some r:ViolationReport | tv = r.typeOfViolation
}

fact networkModelling{
	// Ever report that is in the server has been created by a device.
		Furthermore it is granted that there exists a corresponding report
		in the network
	all vS : ViolationReport | vS.state = ON_SERVER implies
		(some vD : ViolationReport | vD.state = ON_DEVICE and
			sendReportToServerFromDevice[vD, vS])
	// Ever report that is in the network has been created by a device
	all vN : ViolationReport | vN.state in ON_NETWORK implies
		(some vD : ViolationReport | vD.state = ON_DEVICE and
			sendReportToNetwork[vD, vN])
	// Every report created by a device has at most one corresponding report
		in the network
	all vD : ViolationReport | vD.state = ON_DEVICE implies
		(lone vN : ViolationReport | vN.state in ON_NETWORK and
			sendReportToNetwork[vD, vN])
}

/* Users have access only to anonymized data
*/
fact useAnonymizedDataWhenUser {
	all query : ViolationReportsQuery | query.askingEntity in User =>
		query.violationReportsData in
		ViolationReportDataForQueryWithPrivacy
}

/* Municipality has access also to not anonymized data
*/
fact useNotAnonymizedDataWhenMunicipality {
	all query : ViolationReportsQuery | query.askingEntity in Municipality =>
		query.violationReportsData in
		ViolationReportDataForQueryWithoutPrivacy
}

/* There could not exist data not referenced by a query
```

```
*/
fact allDataAreReferencedByAQuery{
	all data : ViolationReportDataForQuery | some v:ViolationReportsQuery |
		v.violationReportsData = data
}

/* All queried data is located on the server
*/
fact dataQueriedOnServer{
	all q:ViolationReportDataForQuery |(
		(#q.pictures > 0 implies some v : ViolationReport | v.state =
			ON_SERVER and #(q.pictures & v.pictures) > 0) and
		(#q.location > 0 implies some v : ViolationReport | v.state =
			ON_SERVER and q.location = v.location) and
		(#q.timestamp > 0 implies some v : ViolationReport | v.state =
			ON_SERVER and q.timestamp = v.timestamp) and
		(#q.typeOfViolation > 0 implies some v : ViolationReport | v.state
			= ON_SERVER and q.typeOfViolation = v.typeOfViolation) and
		(#q.licensePlate > 0 implies some v : ViolationReport | v.state =
			ON_SERVER and q.licensePlate = v.licensePlate))
}

/* The username of every entity must be unique
*/
fact uniqueUsername {
	no disj e1, e2 : Entity | e1.username = e2.username
}

/* There could not exist a username not referenced by an entity
*/
fact allUsernamesAreAssociatedWithAnEtity{
	all u:Username | some e : Entity | e.username = u
}


// ########## NETWORK PREDICATES ##########

/* Represent the act of sending a violation report through the network.
* @param vDevice the representation of the violation report on the device.
* @param vNetwork the representation of the violation report in the network.
*/
pred sendReportToNetwork [vDevice : ViolationReport, vNetwork : ViolationReport]
{
	vDevice.state = ON_DEVICE
	vNetwork.state in ON_NETWORK
	vDevice.pictures = vNetwork.pictures
	vDevice.location = vNetwork.location
	vDevice.timestamp = vNetwork.timestamp
	vDevice.typeOfViolation = vNetwork.typeOfViolation
	vDevice.licensePlate = vNetwork.licensePlate
	vDevice.createdBy = vNetwork.createdBy
}
```

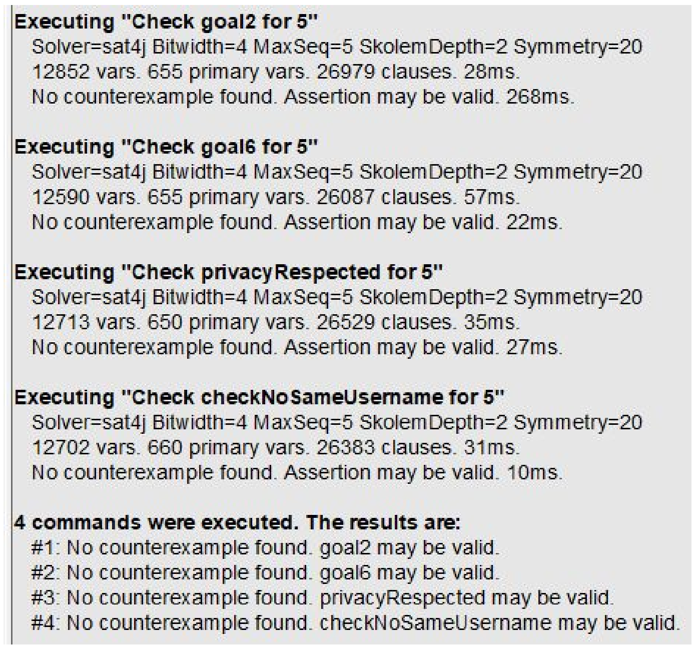
```
/* Represent the act of receiving a violation report from the network.
* @param vNetwork the representation of the violation report in the network.
* @param vServer the representation of the violation report on the server.
*/
pred receiveReportFromNetwork [vNetwork : ViolationReport, vServer :
ViolationReport] {
	vNetwork.state in ON_NETWORK
	vServer.state = ON_SERVER
	vNetwork.pictures = vServer.pictures
	vNetwork.location = vServer.location
	vNetwork.timestamp = vServer.timestamp
	vNetwork.typeOfViolation = vServer.typeOfViolation
	vNetwork.licensePlate = vServer.licensePlate
	vNetwork.createdBy = vServer.createdBy
}

/* Represent the act of sending a violation report to the server from the
device.
* This basically says that there exists a violation report sent in the network
such that:
* it has the same content of the violation report sent by the device,
* and the same content of the violation report received by the server.
* @param vDevice the representation of the violation report on the device.
* @param vServer the representation of the violation report on the server.
*/
pred sendReportToServerFromDevice [vDevice: ViolationReport, vServer :
ViolationReport] {
	vDevice.state = ON_DEVICE
	vServer.state = ON_SERVER
	some vNetwork : ViolationReport | vNetwork.state in ON_NETWORK and
		sendReportToNetwork[vDevice, vNetwork] and
		receiveReportFromNetwork[vNetwork, vServer]
}


// ########## GOAL 2 ##########

/*[R4] The application must allow reporting of violations only from devices
equipped
 *	with a GPS receiver which are in the conditions to obtain a GPS fix.
*/
fact requirement4 {
	#{v:ViolationReport | v.state = ON_DEVICE and v.createdBy.hasGPS = FALSE}
		= 0
}

/*[R5] The application must allow reporting of violations only from devices
equipped
 *	with a camera.
*/
fact requirement5 {
       #{v:ViolationReport | v.state = ON_DEVICE and v.createdBy.hasCamera =
              FALSE} = 0
}

 /*[R6] The application must allow reporting of violations only from devices
with an active
 *     internet connection. This requirement basically says that the violation
report
 *     created by the device will be correctly received by the server.
*/
fact requirement6 {
       #{v:ViolationReport | v.state = ON_DEVICE and v.createdBy.hasInternet =
              FALSE } = 0
}

 /*[R7] A user has the possibility to specify the type of the reported violation
choosing from a list.
 *[R8] The application creates a violation report with at least one picture,
exactly one timestamp,
 *     exactly one location, exactly one type of violation and the license plate
of the vehicle.
*/
fact requirement8 {
       //A valid report is created only if the device has the GPS, a camera and
              internet
       all v : ViolationReport | (v.state = ON_DEVICE and
                                  v.createdBy.hasGPS = TRUE and
                                  v.createdBy.hasInternet = TRUE and
                                  v.createdBy.hasCamera = TRUE)
                            implies
       (#v.pictures >= 1 and
       #v.location = 1 and
       #v.timestamp = 1 and
       #v.typeOfViolation = 1 and
       #v.licensePlate = 1)
}

/*[G2] A violation report received by the system must have enough information to
be valid,
 * i.e. has at least one picture of the violation, exactly one GPS position,
exactly one
 * timestamp, exactly one type of violation and the license plate of the
vehicle.
*/
assert goal2 {
       all v : ViolationReport | v.state = ON_SERVER implies
              (#v.pictures >= 1 and
              #v.location = 1 and
              #v.timestamp = 1 and
              #v.typeOfViolation = 1 and
              #v.licensePlate = 1)
}
```

```
check goal2 for 5


// ########## GOAL6 ##########

/*[D6] Data transferred through connections that use modern encryption protocols
 *cannot be manipulated.
*/
fact domainAssumption6 {
	all v : ViolationReport | v.state = ON_NETWORK_ENCRYPTED implies
		v.canBeAltered = FALSE
	all v : ViolationReport | v.state = ON_NETWORK_NOT_ENCRYPTED implies
		v.canBeAltered = TRUE
}

/*[R17] The application will allow using pictures in a violation report only if
the picture
 * was taken by the application itself, preventing it to be manipulated on the
device.
*/
fact requirement17 {
	all v : ViolationReport | v.state = ON_DEVICE implies v.canBeAltered =
		FALSE
}

/*[R18] All connections used by the system use modern encryption protocols.
*/
fact requirement18 {
	all v : ViolationReport | v.state != ON_NETWORK_NOT_ENCRYPTED
}

/*[R19] Data saved in the server can not be manipulated.
*/
fact requirement19 {
	all v : ViolationReport | v.state = ON_SERVER implies v.canBeAltered =
		FALSE
}

/*[G6] The integrity of the violation report is guaranteed.
*/
assert goal6 {
	all v : ViolationReport | v.canBeAltered = FALSE
}
check goal6 for 5


// ########## PRIVACY MUST BE RESPECTED FOR QUERIES ##########
assert privacyRespected {
	all query : ViolationReportsQuery | all data :
		ViolationReportDataForQuery | (data in query.violationReportsData
		and query.askingEntity in User) => (#data.pictures = 0 and
		#data.licensePlate = 0)
```

```
        all query : ViolationReportsQuery | all data :
                ViolationReportDataForQuery | (data in query.violationReportsData
                and query.askingEntity in Municipality) => (#data.pictures >= 1
                and #data.licensePlate = 1)
}
check privacyRespected for 5


// ########## NO ENTITIES WITH SAME USERNAME ##########
assert checkNoSameUsername {
        all e1 : Entity | all e2 : Entity | e1 != e2 => e1.username != e2.
                username
}
check checkNoSameUsername for 5
```

4.2. Assertions

The Alloy analyzer provided these results: showing that it wasn't able to find any counterexamples for the four assertions.



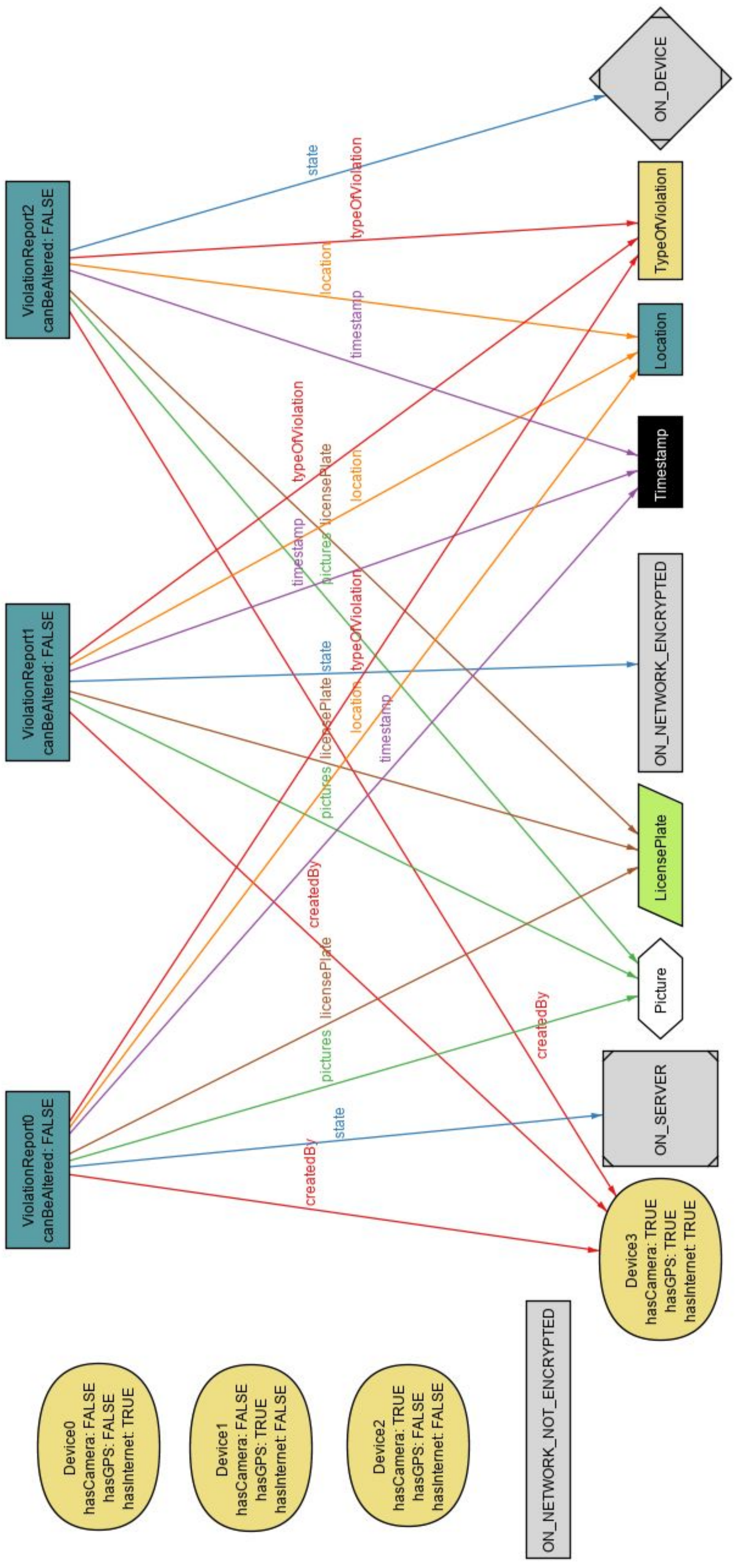
```
Executing "Check goal2 for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=2 Symmetry=20
   12852 vars. 655 primary vars. 26979 clauses. 28ms.
   No counterexample found. Assertion may be valid. 268ms.

Executing "Check goal6 for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=2 Symmetry=20
   12590 vars. 655 primary vars. 26087 clauses. 57ms.
   No counterexample found. Assertion may be valid. 22ms.

Executing "Check privacyRespected for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=2 Symmetry=20
   12713 vars. 650 primary vars. 26529 clauses. 35ms.
   No counterexample found. Assertion may be valid. 27ms.

Executing "Check checkNoSameUsername for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=2 Symmetry=20
   12702 vars. 660 primary vars. 26383 clauses. 31ms.
   No counterexample found. Assertion may be valid. 10ms.

4 commands were executed. The results are:
   #1: No counterexample found. goal2 may be valid.
   #2: No counterexample found. goal6 may be valid.
   #3: No counterexample found. privacyRespected may be valid.
   #4: No counterexample found. checkNoSameUsername may be valid.
```

4.3. Worlds

The following worlds have been chosen to show some important aspects modelled in Alloy of the S2B.

4.3.1. First world

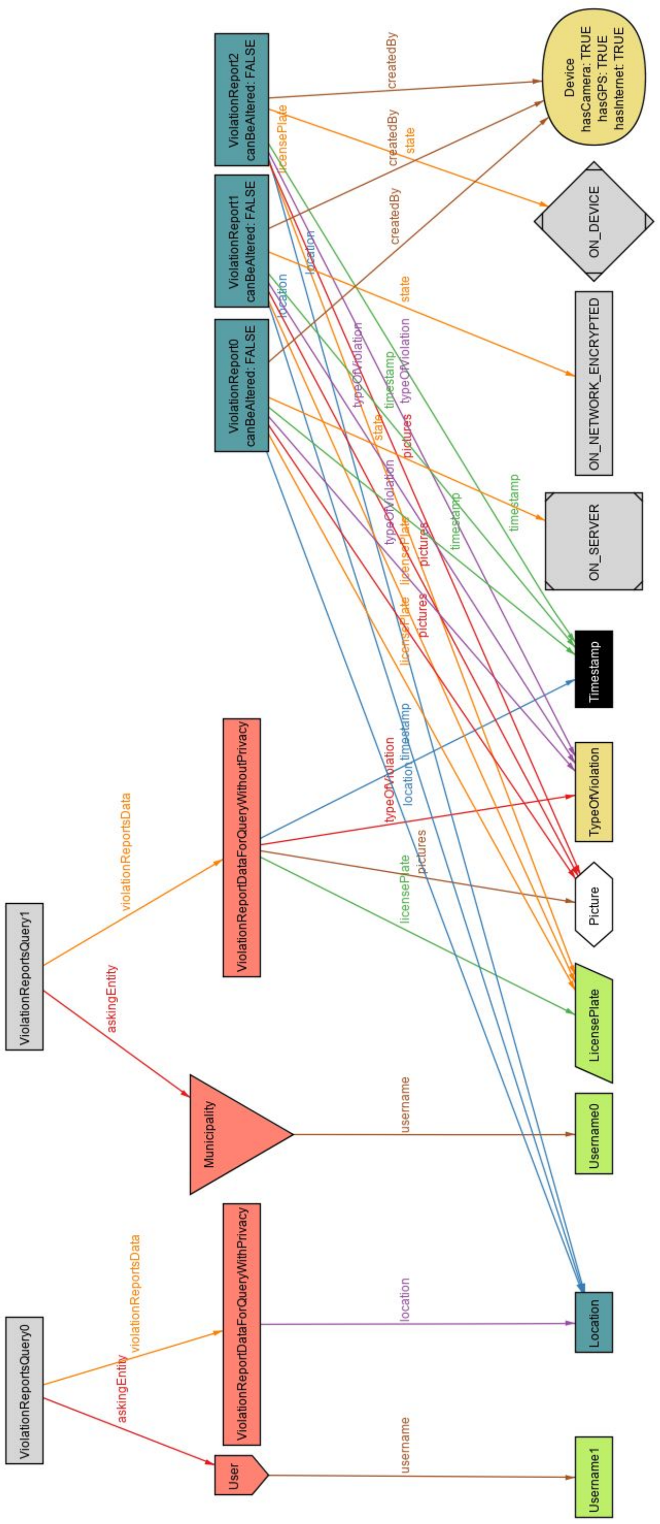
In the first world it is possible to see that all the violation reports present in the server are created by devices with an active internet connection, equipped with a camera and a GPS sensor. It's also possible to see the process in which the violation report go through: it first gets created on the device (*ON_DEVICE*), then goes through the network (*ON_NETWORK_ENCRYPTED*) and finally it reaches the server (*ON_SERVER*).


ViolationReport2
canBeAltered: FALSE
ViolationReport1
canBeAltered: FALSE
ViolationReport0
canBeAltered: FALSE
Device0
hasCamera: FALSE
hasGPS: FALSE
hasInternet: TRUE
Device1
hasCamera: FALSE
hasGPS: TRUE
hasInternet: FALSE
Device2
hasCamera: TRUE
hasGPS: FALSE
hasInternet: FALSE
Device3
hasCamera: TRUE
hasGPS: TRUE
hasInternet: TRUE
ON_NETWORK_NOT_ENCRYPTED
ON_DEVICE
TypeOfViolation
Location
Timestamp
ON_NETWORK_ENCRYPTED
LicensePlate
Picture
ON_SERVER
state
typeOfViolation
location
timestamp
typeOfViolation
licensePlate
location
timestamp
pictures
state
typeOfViolation
licensePlate
location
timestamp
pictures
createdBy
licensePlate
createdBy
pictures
state
createdBy

4.3.2. Second world

In the second world there is only one device that creates one violation report that goes through the process seen before. The data of that violation report is then used for answering a query made by a user and a query made by a municipality. The data provided for the query respects the privacy constraint imposed: the user can't see license plates and pictures while the municipality can.

Furthermore it's also possible to see the uniqueness of the usernames.



5. Effort spent

Matteo Marchisciana		
Date	Task(s)	Time
23/10/2019	World and machine phenomena, goals, domain assumptions and requirements	7 hr
26/10/2019	Overall description	5 hr 15 min
30/10/2019	Overall description	20 min
31/10/2019	Overall description	30 min
02/11/2019	Overall description & Alloy	1 hr
03/11/2019	Overall description and review	1 hr 30 min
04/11/2019	General purpose	20 min
06/11/2019	UML & General purpose	2 hr
09/11/2019	Alloy, document structure and review	3 hr 20 min
10/11/2019	Document structure and review	1 hr 45 min
Total:		23 hr

Andrea Marcer		
Date	Task(s)	Time
23/10/2019	World and machine phenomena, goals, domain assumptions and requirements	7 hr
26/10/2019	Scenarios	1hr
30/10/2019	Use cases	45min
31/10/2019	Use cases	1hr 45min
02/11/2019	Use cases	1hr 15min
03/11/2019	Use cases	2hr 15min

04/11/2019	Overall review	2hr 45min
05/11/2019	Use cases	30min
06/11/2019	Overall review	30min
07/11/2019	UML	1 hr 30 min
08/11/2019	Use cases	2hr
09/11/2019	Alloy	4hr 30min
09/11/2019	Overall review	2hr 15 min
10/11/2019	Alloy	3hr
10/11/2019	Alloy	2hr 30 min
10/11/2019	Overall review	1hr
04/12/2019	Use cases	30 min
12/12/2019	General overview	45 min
Total:		35 hr 15 min

Dennis Motta		
Date	Task(s)	Time
23/10/2019	World and machine phenomena, goals, domain assumptions and requirements	4 hr 0 min
26/10/2019	Goals, domain assumptions and requirements	2 hr 45 min
27/10/2019	Performance requirements, design constraints and Software System Attributes	2 hr 30 min
28/10/2019	Software System Attributes	1 hr 0 min
29/10/2019	Goals, domain assumptions and requirements	2 hr 30 min
31/10/2019	Overall description	0 hr 15 min

02/11/2019	Alloy, Software System Attributes, design constraints, dependencies, External Interface Requirements	2 hr 30 min
03/11/2019	Alloy, overall description	3 hr 0 min
04/11/2019	Software interfaces, overall description	0 hr 30 min
05/11/2019	Sequence diagrams	2 hr 0 min
06/11/2019	UML, Sequence diagrams	2 hr 0 min
08/11/2019	Mockups	1 hr 0 min
09/11/2019	Mockups, title page, Alloy	5 hr 30 min
10/11/2019	Alloy, overall review	4 hr 30 min
Total:		34 hr 0 min

6. References

1. “a GPS fix has an accuracy of ≈ 5 m radius in an open area”. https://www.gps.gov/systems/gps/performance/accuracy/
2. “ISO 6709”. https://www.iso.org/standard/39242.html
3. “ISO/IEC 27001”. https://www.iso.org/standard/54534.html
4. “ISO/IEC 27701”. https://www.iso.org/standard/71670.html
5. “ISO 27001 and the GDPR”. https://www.itgovernance.co.uk/gdpr-and-iso-27001