SafeStreets - RASD

Politecnico di Milano - A.Y. 2019/20120

Andrea Aspesi - andrea.aspesi@mail.polimi.it
Elia Battiston - elia.battiston@mail.polimi.it
Alessandro Carabelli - alessandro2.carabelli@mail.polimi.it

Contents

I Introduction 4

1 Purpose 4

2 Scope 4

2.1 Description of the given problem . . . 4
2.2 Current systems . . . 5
2.3 Goals . . . 5

3 Definitions, Acronyms, Abbreviations 6

3.1 Definitions . . . 6
3.2 Acronyms . . . 6
3.3 Abbreviations . . . 6

4 Revision history 6

5 Reference documents 6

6 Document structure 6

II Overall description 8

1 Product perspective 8

2 Product functions 9

2.1 Violation reporting . . . 9
2.2 Data visualization . . . 9
2.3 Data analysis and notification . . . 9
2.4 Automatic check generation . . . 10

3 User characteristics 10

3.1 Actors . . . 10

4 Assumptions, dependencies and constraints 10

4.1 Assumptions and dependencies . . . 10
4.2 Constraints . . . 12

III Specific requirements 13

1 External Interface Requirements 13

1.1 User interfaces . . . 13
1.2 Hardware interfaces . . . 16
1.3 Software interfaces . . . 16

2 Functional requirements 17

2.1 Visitor . . . 17
2.2 User . . . 17
2.3 Authority . . . 22
2.4 Administrator . . . 28
2.5 Requirements . . . 35

3 Performance requirements 38

4 Design constraints **38**
4.1 Standards compliance 38
4.2 Hardware limitations 38
4.3 Any other constraint 39

5 Software system attributes **39**
5.1 Availability and Reliability 39
5.2 Security 39
5.3 Maintainability 39
5.4 Portability 39

IV Formal analysis using Alloy **40**

1 Alloy **40**
1.1 Data Types 40
1.2 Data 41
1.3 Facts 42
1.4 Assertions 43
1.5 Dynamic model 43

2 Results **44**
2.1 Proof of Consistency 44
2.2 Generated World 44

V Effort spent **45**

Part I
Introduction

1 Purpose

This RASD is the Requirement Analysis and Specification Document of the SafeStreets project. Objectives of this document are to fully analyse and describe the customer prospective, represent the system in terms of its requirements, describe the solution adopted in terms of its use cases and show the constraints and the limit of this community tool. The addressees of this document are developers working on the implementation of the solution and the customer.

2 Scope

2.1 Description of the given problem

SafeStreets is a community service that aims to provide the users of the possibility to notify parking and traffic violations. It can be done providing proofs of the accident like pictures.
The goal is to raise awareness in drivers that driving and parking can make other's life harder. With a little effort, instead, it can be made easier.
The service will be served through a mobile application where users can report new violations and see the ones they reported before. Every attached picture comes with the metadata provided by the smartphone (like: time, position...) and with valuable information given by the user whom can describe the violation. More pictures of the same violation will be connected crossing the spacial and time information so that more viewing angles will be available.
To ensure the chain of custody, the backend infrastructure will run an AI algorithm to ensure the quality of the data given (e.g. no edited picture) before considering them valuable.
Authorities have access to all the data provided by the users, plus:

- ▷ Suggestions from the system on ways the municipality can improve the safety of the streets, directly from their dashboard. These are computed by crossing SafeStreets reports with the municipality accident and traffic tickets data;
- ▷ Automatic acquisition of reports to produce traffic tickets from the data collected;
- ▷ Maps of the less safe roads in the city;
- ▷ Vehicles that commit the most violations and other statistics for their municipality.

2.1.1 Phenomena table

Phenomenon	**Shared**	**Controller**
Occurrence of a violation	N	World
Occurrence of an accident	N	World
Emission of a traffic ticket	N	World
Detection of a violation by a citizen	N	World
Submission of a report	Y	Machine
Visualization of data about streets safety	Y	Machine
Visualization of a user's past violation reports	Y	Machine
Filtering and research of reports of an officer	Y	Machine
Request for safety suggestions from the municipality	Y	Machine
Saving of a report	N	Machine
Analysis of data to produce statistics and suggestions	N	Machine
Delivery of new reports to the municipality's systems	N	Machine
Importing of accidents and traffic tickets data from the municipality's systems	N	Machine

Table 1: SafeStreets phenomena

2.2 Current systems

Currently, there is no automatic system for sharing violations like the one proposed by SafeStreets. The citizen has to directly contact the authorities if it sees one.
On the other hand there are already multiple municipality systems that we can integrate with for automatic tickets creation and municipality information retrieval. We opted to fully cooperate with the municipality's API so that they can interact with data we provide.

2.3 Goals

- ▷ [G1] Allow citizens to register providing basic information that certifies their identities and become certified users;
- ▷ [G2] Allow authorities to register through a verification procedure;
- ▷ Enable certified users to:
  - [G3] report new traffic or parking violations;
  - [G4] see their own past reports;
  - [G5] access the map showing the most reported streets.
- ▷ Enable authorities to:
  - [G6] access all certified users services;
  - [G7] access all report data (including data about the user who created it);
  - [G8] acquire report data in their own systems to generate tickets;
  - [G9] access data analysis results with suggestions.
- ▷ Enable administrators to:
  - [G10] certify authorities;
  - [G11] block or remove users;
  - [G12] assign roles to users.

3 Definitions, Acronyms, Abbreviations

3.1 Definitions

- ▷ *Violation*: illegal behaviour executed by the Offender (3.1)
- ▷ *Violation report / Report:* information regarding a single violation, sent by a Registered User (3.1) through SafeStreets
- ▷ *Server:* SafeStreets server which elaborates the reports and communicates to municipality through custom APIs

3.2 Acronyms

- ▷ *GDPR:* General Data Protection Regulation
- ▷ *API:* Application Programming Interface
- ▷ *GPS:* Global Positioning System
- ▷ *UI:* User Interface
- ▷ *ID:* Identification Document
- ▷ SLA: Service Level Agreement

3.3 Abbreviations

- ▷ *[Gn]:* Goal n
- ▷ *[Dn]:* domain assumption n
- ▷ *[Rn]:* Requirement n

4 Revision history

- ▷ *Version 1.0:*
  Initial release
- ▷ *Version 1.0.1*
  Sequence diagram pictures fixes
- ▷ *Version 1.1*
  Changes to some requirements

5 Reference documents

- ▷ Specification document: “SafeStreets Mandatory Project Assignment”
- ▷ Alloy language reference

6 Document structure

This document consists of the following parts:

1. Brief description of this document, its nomenclature and the description of the project.
2. Overall description of the system to be realized and definition of involved actors, assumptions on the project’s definition and domain.

3. More detailed description of the structure of the system to be realised, including interfaces, use cases, functional and non functional requirements.
4. Analysis of the system's consistency using Alloy.
5. Reporting of the time spent by each team member while redacting the document.

Part II

Overall description

1 Product perspective

The product, built from scratch, will use the geolocation systems embedded in smartphones to acquire data about the position of violation, and the smartphone's camera for license plate and violation pictures. It won't be possible to manually upload photos and manually insert positions to avoid fake reports. All connections will be handled via HTTPS requests with TLS encryption, to ensure all the reports arrive from the sender to the server with no external manipulation and no external access by non-authorized personnel; this also assures GDPR compliance. The database and the server will be externally hosted; privacy will be granted by the maintainer of the servers.
The systems dialogues with the municipality services using its API to acquire information about occurred accidents (with the aim of information crossing analysis of potentially unsafe or dangerous areas to be notified to the municipality) and to send report information to the automatic ticketing system of the municipality.

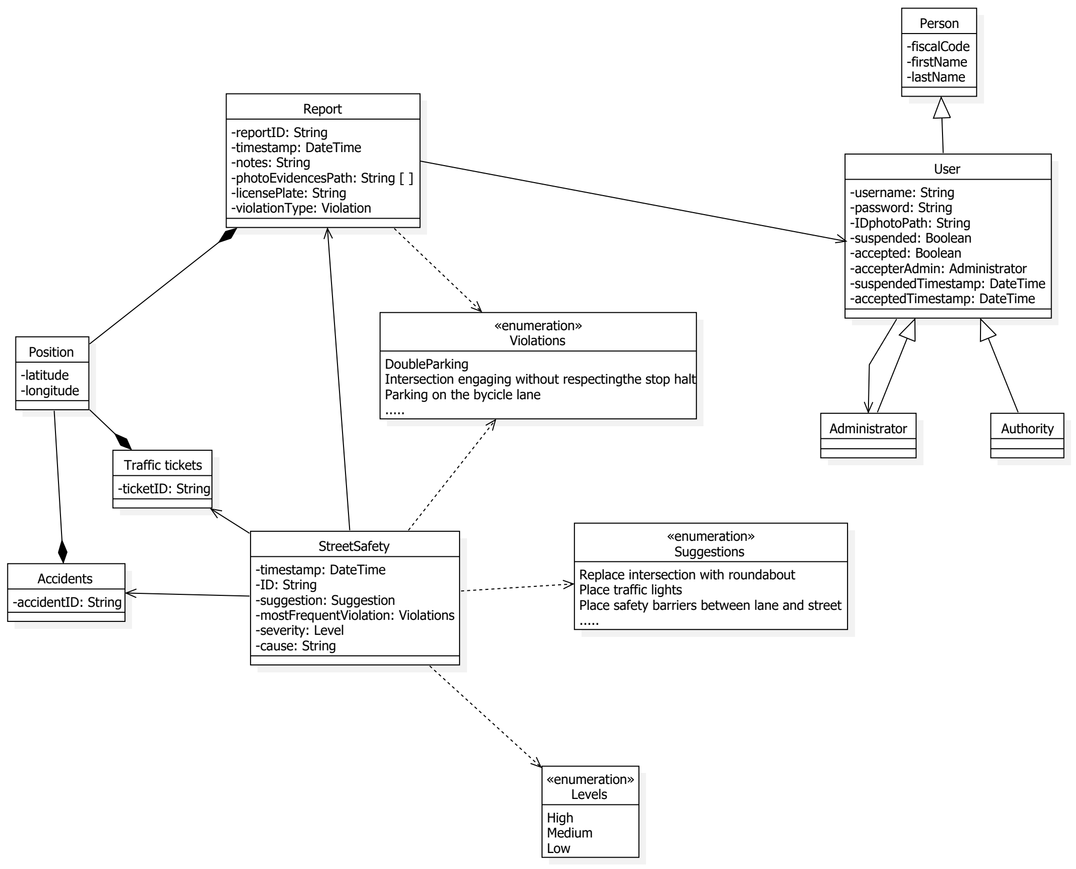


Figure 1: Class diagram

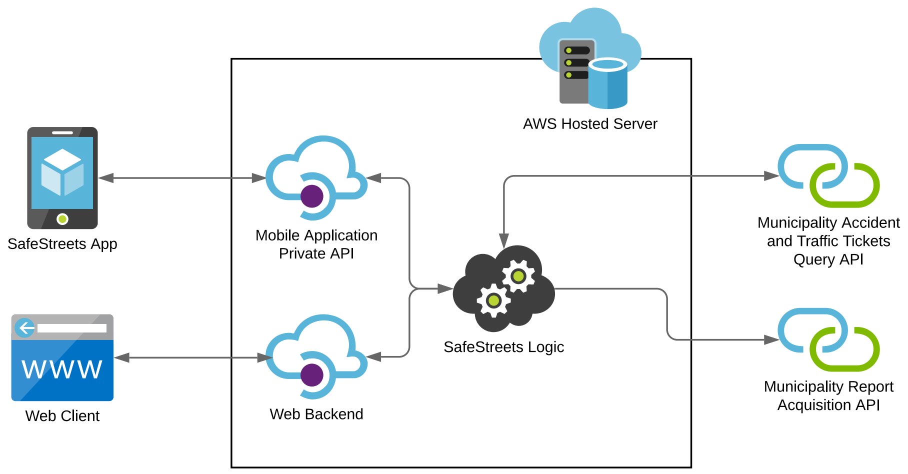

AWS Hosted Server
SafeStreets App
Mobile Application Private API
Municipality Accident and Traffic Tickets Query API
SafeStreets Logic
Web Client
Web Backend
Municipality Report Acquisition API

Figure 2: Proposed architecture

2 Product functions

To explicit all the possibilities that will and won't be offered by the system, in the following paragraphs all the available functions are listed and punctually analysed. The aim of the explanation is to line up an unambiguous description of what the system is expected to do and what users can or can't do, in addition to covering the goals listed in 2.3.

2.1 Violation reporting

Users who notice a violation are granted the possibility to notify it to authorities using the SafeStreets app, via the compilation of a small form and attachment of pictures of violation and plate number. Data about time and position will be automatically taken from local time (acquired through the Internet connection) and the built-in GPS. As already said in 1 and fully explained in 4.2, devices which are not equipped with a camera, GPS sensor or a functioning Internet connection won't be able to notify violations.
Before permanently saving reports in SafeStreets database, the server verifies that inserted data is coherent with reality, verifying that the picture has not been altered, eventually converting the license plate picture into a plate number and verifying that this number corresponds to the license plate of a circulating car (this last check is made using the municipality API). In case of fake reporting or invalid plate number, the report is rejected and the user is notified of the rejection. In case of multiple reports sent by the same user and sequentially rejected, a notice will be sent to officers.
Registered violations will be notified to authorities.

2.2 Data visualization

The systems allows registered users to access analysed and anonymized data about the distribution of the reports over the selected area, presented in the form of a map with highlighted areas in graded colours, from less-reported areas to highly-reported areas (from green to red). Users have the possibility to zoom in and out in the map, having as upper limit the city in the whole and lower limit of detail a single street (it won't be possible to visualize details about the distribution of reports along a single street.
In addition to registered users, officers will have the possibility to visualize the distribution of the reports with no lower bound, in addition to the visualization of the full list of reports without anonymization and the list of the most reported cars.

2.3 Data analysis and notification

The SafeStreets server acquires data from the municipality through the granted APIs to be crossed with the data from the reports; the result will be automatically analysed from the system itself to verify possible unsafe or even

dangerous areas of the city (the level of dangerousness is directly connected to the category of violation of each report and the quantity of reports from that area).
The system will grant authorities access to the elaborated data and possible correcting manoeuvrers, such as the positioning of barriers to separate the vehicle part of the road from the pedestrian one, with suggestions given coherently with the type of notifications received.
Alerts will be automatically sent to authorities in case of new unsafe areas elaborated from the system or in case of reception of a large number of reports regarding the same vehicle and/or area (the number of reports over which an alert is sent is decided with the authorities and corrected with the starting of the service).

2.4 Automatic check generation

Data sent by users and verified by the server will be forwarded to the municipality via the provided API (always using HTTPS and TLS encryption) to be processed for the generation of tickets. The municipality system will then provide informations about issued and paid tickets, to be used by the server to build statistics about efficiency, effectiveness and precision of the SafeStreets product.

3 User characteristics

3.1 Actors

- ▷ *Visitor / Unregistered User:* a person who installed the SafeStreets application but never registered to the service.
  He can only view a brief explanation of the application's functions and proceed with the registration.
- ▷ *User / Registered User*: a person who executed the whole registration procedure, providing necessary information and confirming its registration by following the link in a confirmation email.
  He can log to his user account though the application to access all of the functionalities it provides to normal users without particular permissions. This means he can send a violation report and view its previous violations, view information on streets with the most reports. He can't view reports submitted by other users.
- ▷ *Administrator:* SafeStreets employee whose role is to manually manage user accounts (for example by assigning or removing roles and permissions).
  They can access a dedicated web backend to carry out these operations.
- ▷ *Officers / Authority:* police officer or other type of municipality employee involved with SafeStreets' activity. He can independently sign up for an account through the application, and request its Officer permissions to be granted by an Administrator in a second moment. Alternatively, the municipality can ask SafeStreets to make an Administrator create its account with relevant permissions already set.
  Besides being able to use every feature offered to Registered Users through the mobile application, he can also view reports sent by any user.
  Authorities can also access a web backend provided by SafeStreets to easily browse the whole reports dataset and get statistical data about it.
- ▷ *Municipality:* entity represented by the local authorities, which provides data through API endpoints and receives data form SafeStreets through different API endpoints.
- ▷ *Offender:* a person who is responsible for a violation, for example the person who parked their car in a forbidden area or exceeded the speed limit while driving. If no information about the specific person who caused the violation is present, the owner of the offending vehicle is considered to be the offender.
- ▷ *Car / Vehicle:* the offender's vehicle who caused the violation. The violation can occur both while it's parked (with or without physical presence of the offender) or while it's being driven by the offender.

4 Assumptions, dependencies and constraints

4.1 Assumptions and dependencies

Because of ambiguities in the specification of requirements, it's necessary to make some assumptions on the correct interpretation of the customer's needs.

4.1.1 Request assumptions

1. Users needs to be registered in order to send a violation report
2. Registration is carried out by providing identification data of the user. Since the report of violations can lead to the generation of a traffic ticket against the offender, we have to provide a strong identification of the person who filed the report:
   - ▷ First name
   - ▷ Last name
   - ▷ Fiscal code
   - ▷ ID photo
   - ▷ Email address
3. At registration, a SafeStreets account is created for the new user. He can log in it by providing his email or username, and the password he chose.
   SafeStreets should never have access to the plaintext password. It has to be hashed in a secure way on the client's machine to be consecutively sent to the server.
4. To assure confidentiality, any data sent between SafeStreets and the user on a public network have to be encrypted.
5. Every type of account in the system starts with basic permissions, on top of which an Administrator can assign roles such as "Officer" (3.1).
6. The municipality offers retrieval of accident information through a REST API. SafeStreets has to obtain permissions to access it and credentials to do so.
7. The municipality creates a REST API for traffic tickets generation from the information sent by SafeStreets.
8. Reports can only be sent with photos taken directly from the camera, and not by uploading an image, in order to avoid fake reports and to be sure temporal data is correct. This implies only devices with cameras can be used with SafeStreets.
9. Since the system has to automatically detect the user's location when he's filing a report. This implies only devices with GPS can be used with SafeStreets.
10. Because of assumptions 8 & 9, we assume the main component of the SafeStreets system only has to be developed as a mobile application.
    Remaining components such as the administrative console and functions reserved to municipalities can be implemented as web applications to maximize accessibility and compatibility across devices.

4.1.2 Domain assumptions

- ▷ [D1] User's identification data is correct and verified
- ▷ [D2] Location data in reports is precise enough to detect the name of the street where the violation took place and a rough estimation of the nearest civic number
- ▷ [D3] One or more of the images provided with the report are sufficient to recognize the offender's car and visually verify which violation took place
- ▷ [D4] It's always possible for authorities to identify the offender either by its appearance in a photo or by ownership of the vehicle through the license plate
- ▷ [D5] User's reports are good willed and do not contain fake information on purpose
- ▷ [D6] People who want to become SafeStreets users have an Italian fiscal code or can request one

4.2 Constraints

4.2.1 Regulatory policies

Since SafeStreets needs to identify users before letting them access its services, it needs to do so in compliance with current regulations on the matter (mainly GDPR). This means:

- ▷ The user has to be clearly informed about how its data will be used by SafeStreets, and allow this use before completing the registration process.
- ▷ SafeStreets has to provide users a way to review and eventually request the deletion of any data about him on the service's databases.

In addition, after registration, the user has to grand permission on the usage of its live position (acquired through his device's GPS) to complete metadata of his reports.
Contact information such as email addresses and telephone numbers will not be used for commercial purposes.

4.2.2 Hardware limitations

- ▷ Mobile application (4.1.1.10)
  - Android or iOS operating system
  - Internet connection (mobile data plan or Wi-Fi)
  - Camera
  - GPS
- ▷ Administrative and municipality web backends
  - Web browser supporting HTML5, CSS3 and JavaScript
  - Internet connection

Part III

Specific requirements

1 External Interface Requirements

1.1 User interfaces

1.1.1 Navigation in the mobile application

The application will use an intuitive *Master-Detail* structure: the navigation between functions will be achieved with a side menu, summoned by dragging from the left side of the screen or by tapping on the *burger menu* icon in pages from where it is available.

1.1.2 Mobile application mockups

The following designs represent the approximate layout of the information presented to users in the SafeStreets mobile application.

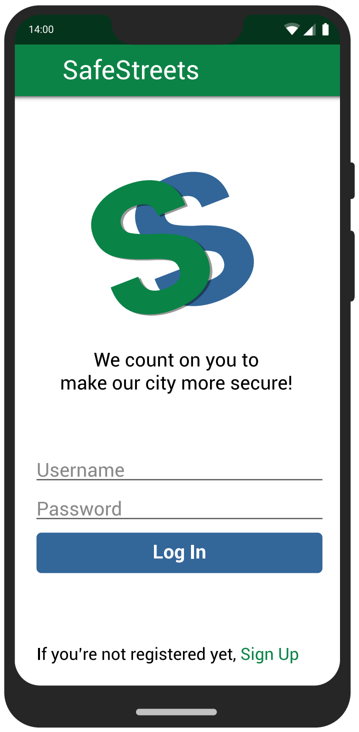

14:00
SafeStreets
We count on you to make our city more secure!
Username
Password
Log In
If you're not registered yet, Sign Up

Figure 3: Login page

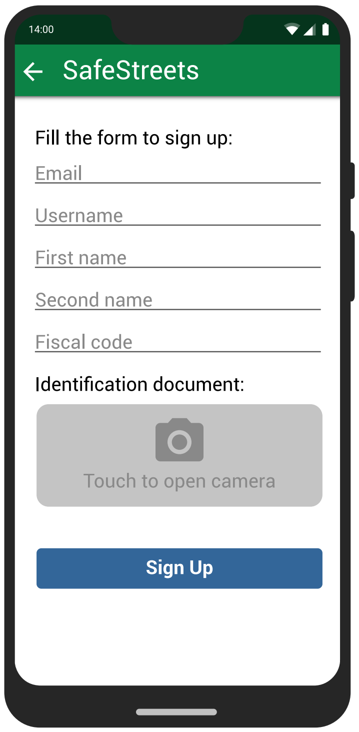


Figure 4: Sign up page

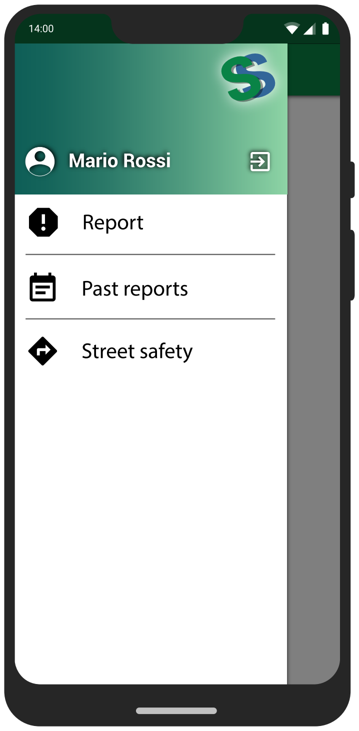


Figure 5: Detail page

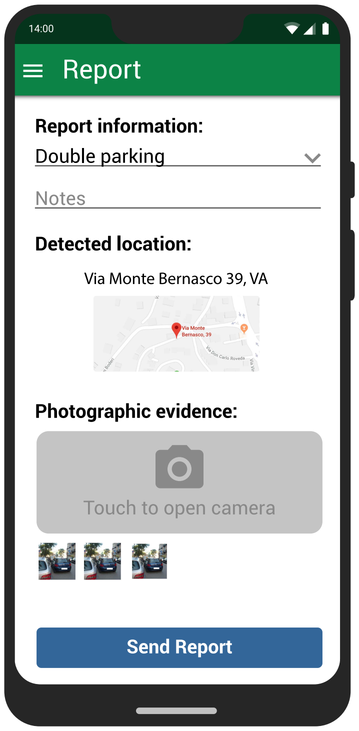
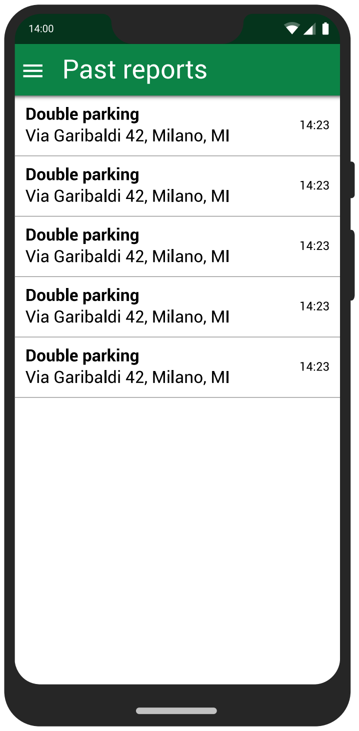


Figure 6: Report (default master page)



Double parking
Via Garibaldi 42, Milano, MI 14:23

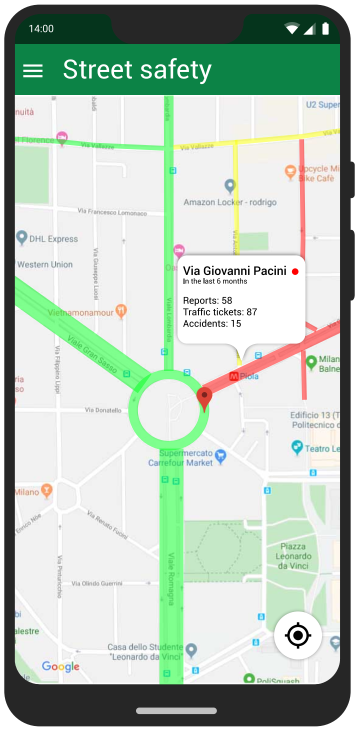
Double parking
Via Garibaldi 42, Milano, MI 14:23

Double parking
Via Garibaldi 42, Milano, MI 14:23

Double parking
Via Garibaldi 42, Milano, MI 14:23

Double parking
Via Garibaldi 42, Milano, MI 14:23

Figure 7: Past reports



Via Giovanni Pacini
In the last 6 months
Reports: 58
Traffic tickets: 87
Accidents: 15

Figure 8: Street safety

1.1.3 Navigation in the web backend

The backend interface will be very basic, by showing relevant data through tables. Administrators and authorities can choose different functionalities through the left sidebar.

1.1.4 Web backend mockups

The following designs represent the approximate layout of the information presented to administrators and authorities in the SafeStreets web backend.

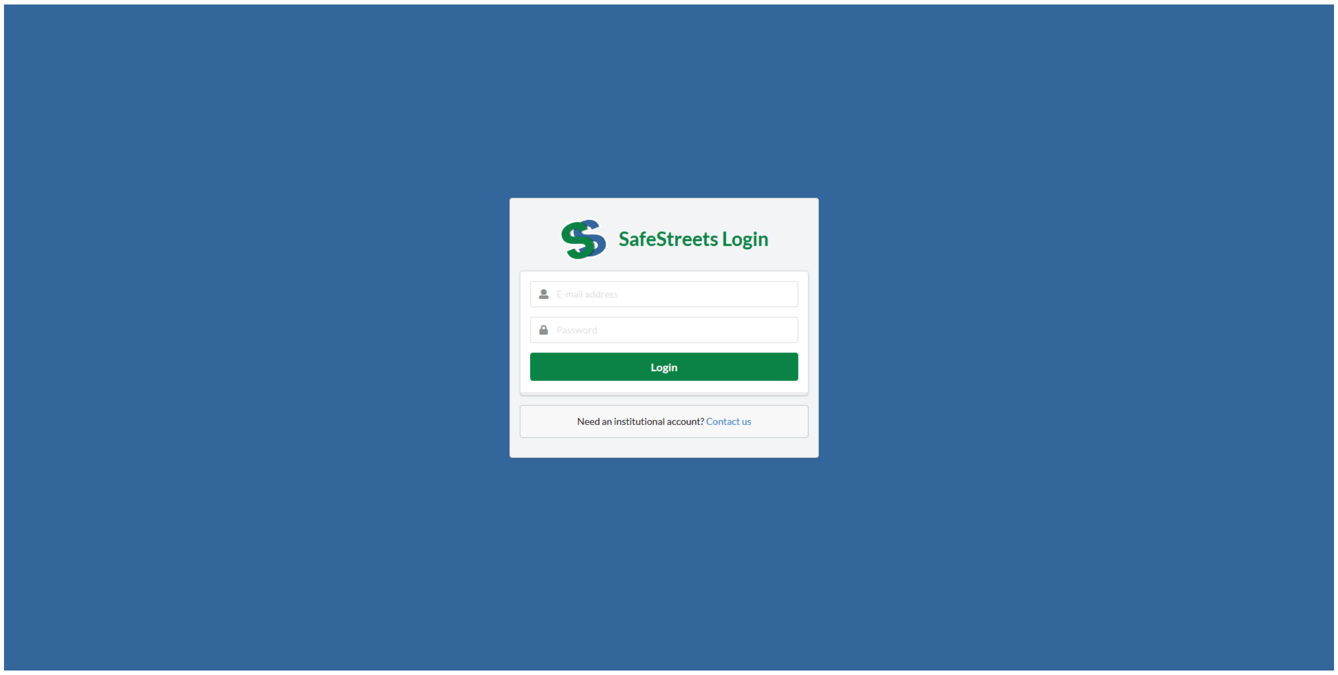
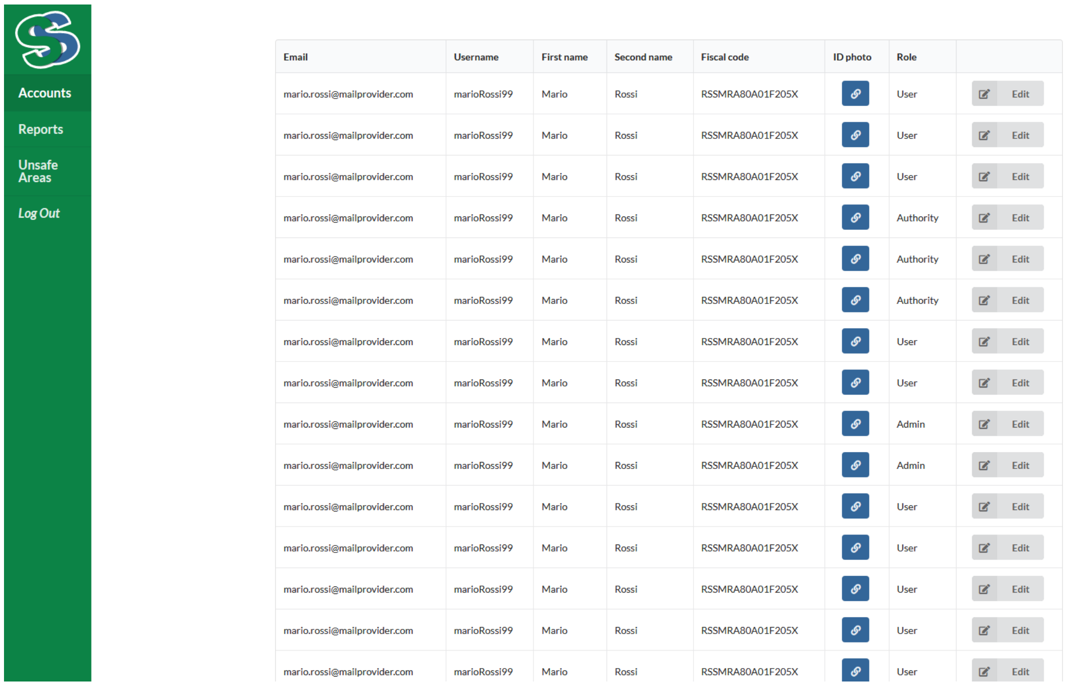
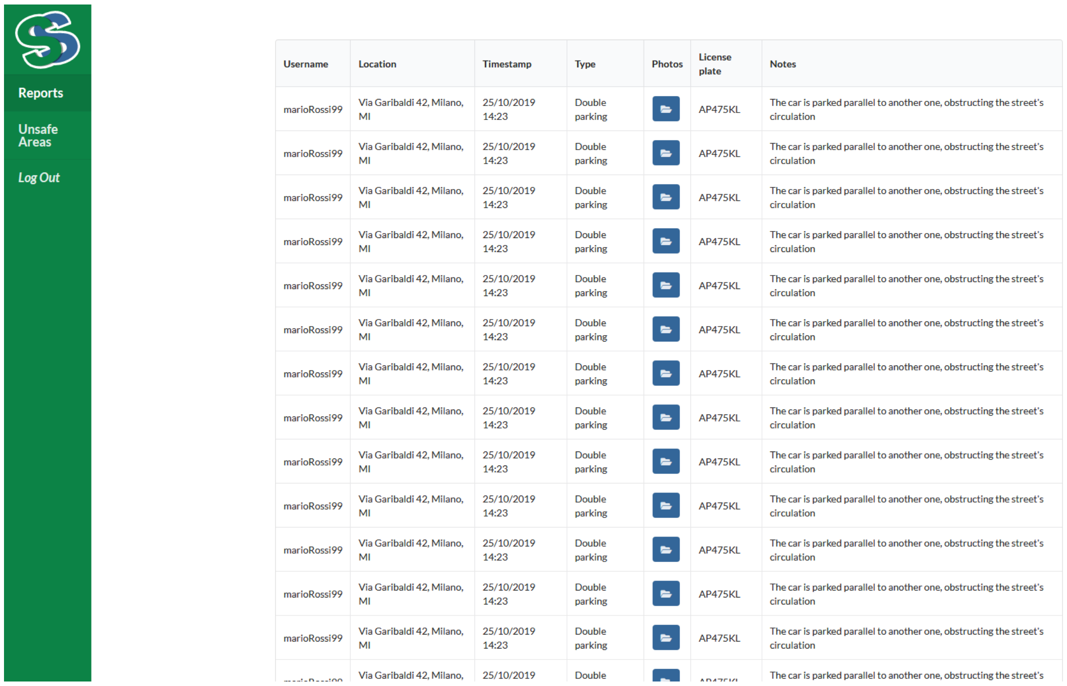


Figure 9: Login page



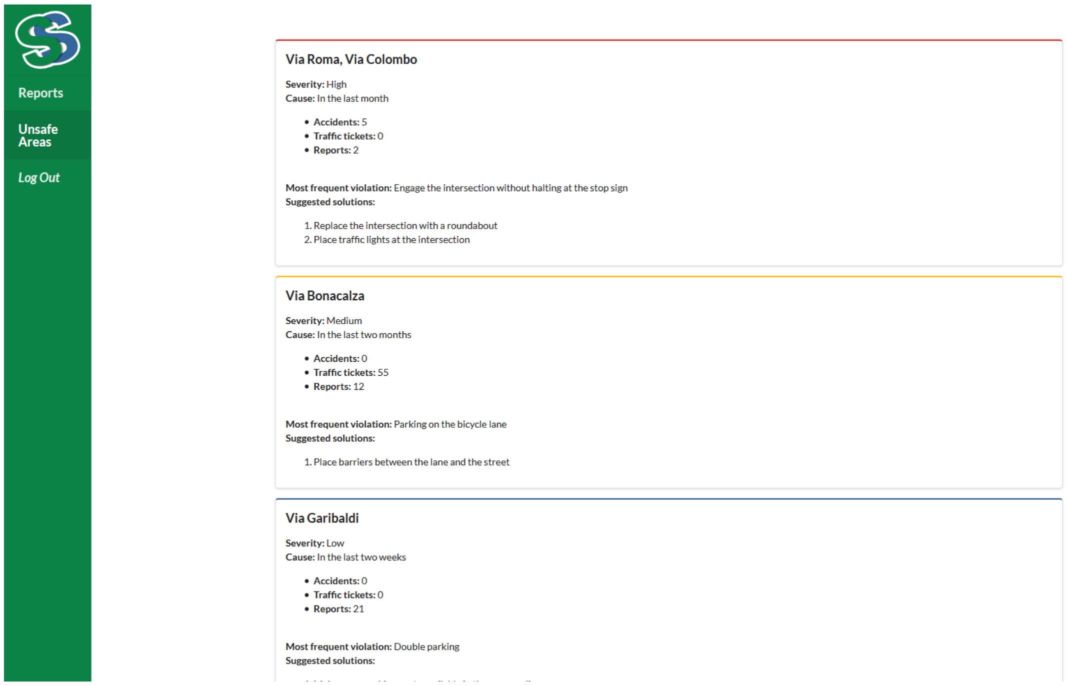
Email	Username	First name	Second name	Fiscal code	ID photo	Role	
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	Authority	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	Authority	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	Authority	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	Admin	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	Admin	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit
mario.rossi@mailprovider.com	marioRossi99	Mario	Rossi	RSSMRA80A01F205X	🔗	User	Edit

Figure 10: Accounts control page, only usable by administrators



Username	Location	Timestamp	Type	Photos	License plate	Notes
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
marioRossi99	Via Garibaldi 42, Milano, MI	25/10/2019 14:23	Double parking	📁	AP475KL	The car is parked parallel to another one, obstructing the street's circulation
	Via Garibaldi 42, Milano,	25/10/2019	Double	📁		The car is parked parallel to another one, obstructing the street's

Figure 11: Reports control page, usable by both administrators and authorities



Via Roma, Via Colombo

Severity: High
Cause: In the last month

- **Accidents:** 5
- **Traffic tickets:** 0
- **Reports:** 2

Most frequent violation: Engage the intersection without halting at the stop sign
Suggested solutions:

1. Replace the intersection with a roundabout
2. Place traffic lights at the intersection

Via Bonacalza

Severity: Medium
Cause: In the last two months

- **Accidents:** 0
- **Traffic tickets:** 55
- **Reports:** 12

Most frequent violation: Parking on the bicycle lane
Suggested solutions:

1. Place barriers between the lane and the street

Via Garibaldi

Severity: Low
Cause: In the last two weeks

- **Accidents:** 0
- **Traffic tickets:** 0
- **Reports:** 21

Most frequent violation: Double parking
Suggested solutions:

Figure 12: Unsafe area warnings and suggestions, usable by both administrators and authorities

1.2 Hardware interfaces

The system does not provide hardware interfaces.

1.3 Software interfaces

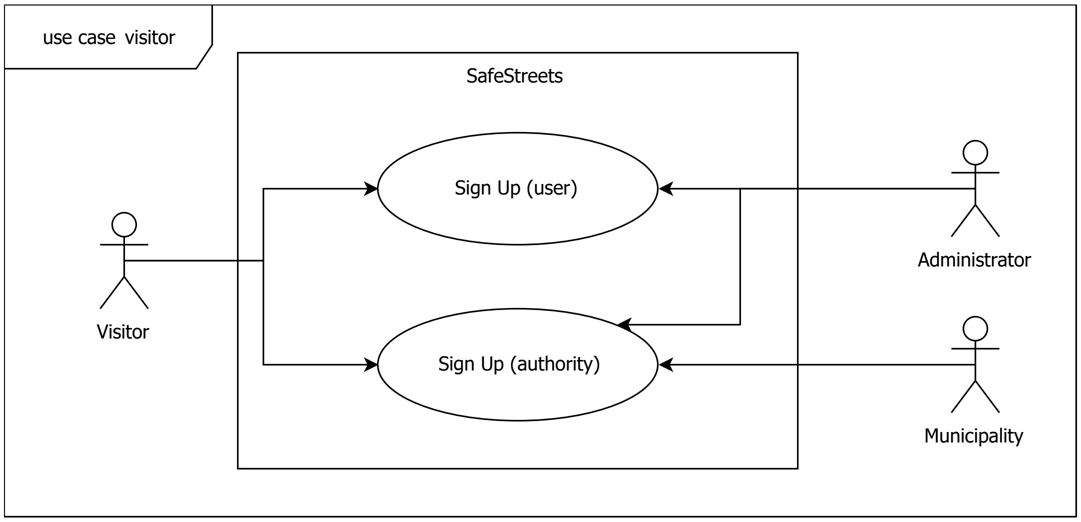
No software interface for other applications is planned for this release of SafeStreets.

Interaction with the municipality's system will be handled completely by authorities with the creation of REST APIs to which SafeStreets will perform request to obtain or provide data, as explained in (10.1.1.6 and 7).
In other words, SafeStreets will always act as a client in the communication between the systems.

2 Functional requirements

2.1 Visitor

For visitors, only the use case diagram of the sign-up activity is included; the activity for each category of actors is described in the sections below.



2.2 User

2.2.1 *Scenarios*

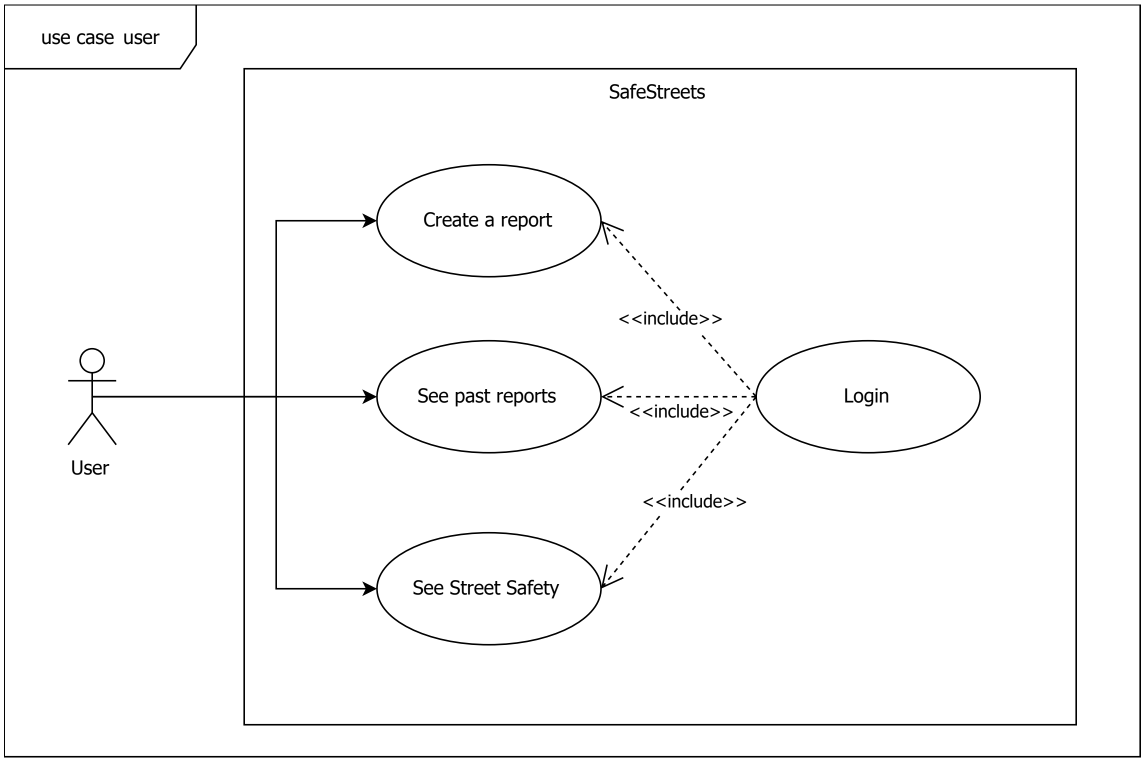
▷ *Scenario 1*

Gianluca lives in a crowded street in Milan and he often finds double parked neighbours' cars near the garage when he goes out to work. Every time this happens, Gianluca risks to score the car while driving it out. Gianluca registered to SafeStreets so that, from now on, every time he finds a double parked car he will send a report and the car owners will be checked, discouraging double parking in the area and allowing Gianluca to save money from the auto body mechanic.

▷ *Scenario 2*

Mario is a registered SafeStreets user and lives in a farmstead near Voghera. He usually drives to Milan with his wife and his two little children to visit the city and wants to know in which areas he can safely park near the location he chose for the trip. Thanks to SafeStreets, Mario can check out which streets have the highest reporting for double parking, red light offences and unsafe driving, so that he can avoid those areas and arrive to its destination without worries about his family's safety.

2.2.2 Use case diagram



2.2.3 Use cases

Name	**Sign up**
Actor	Visitor
Entry conditions	The visitor installed the application on his device and opens it.
Events flow	1. The user chooses the Sign Up option. 2. The user fills in all the fields. 3. The user provides a photo of his identification document. 4. The user hits the Sign Up button. 5. The system saves the data. 6. The administrator validates user's data and accepts the registration.
Exit conditions	The visitor becomes a registered user and can access SafeStreets functionalities.
Exceptions	▷ Some or all of the fields are not filled / no photo has been provided: the system prompts the user a notification and asks for a data filling. ▷ The user is already registered: the app notifies the user to contact an administrator to retrieve the password. ▷ The administrator refuses the registration: the system sends an e-mail to the visitor to notify the refusal and eventual motivations.

Name	**Login**
Actor	Registered user
Entry conditions	▷ The user installed the application on his device and opens it ▷ The user has already completed the “Sign up” activity
Events flow	1. The user fills the Username and Password fields with his credentials. 2. The user hits the Login button.
Exit conditions	Login is completed, the user is prompted to the main activity page.
Exceptions	▷ The user insert wrong credentials: the system shows a notification and suggests to contact an administrator to retrieve the username and/or password. ▷ The user is not registered: for security reasons the system prompts the same alert for the Wrong credentials exception.

Name	**Create a report**
Actor	Registered user
Entry conditions	The user has already completed the “Login” activity.
Events flow	1. The user selects the “Report” option in the application menu. 2. The user fills in all the fields. 3. The user make at least one photo of the violation. 4. The user hits the “Send Report” button. 5. The system saves the report and the attached photo(s) in the database.
Exit conditions	The report is sent to the system and will be taken care of by the first free Officer.
Exceptions	▷ Some or all of the fields are not filled / no photo has been provided: the system prompts the user a notification and asks for a data filling.

Name	**See past reports**
Actor	Registered user
Entry conditions	The user has already completed the “Login” activity.
Events flow	1. The user selects the “Past reports” in the application menu. 2. The system retrieves the data about user’s past sent reports.
Exit conditions	The application shows a list of the user’s past sent reports.
Exceptions	▷ The user has no past reports: the application shows an empty page with “No past reports found” notification in the middle.

Name	**See Street Safety**
Actor	Registered user
Entry conditions	The user has already completed the “Login” activity.
Events flow	1. The user selects the “Street safety” in the application menu. 2. The system generates a map highlighting the streets with safety issues, with different colours based on the safeness of the street. 3. The user clicks over a highlighted street. 4. The application shows a popup with all the statistics and details of the clicked street.
Exit conditions	The application shows the map and allows the user to interact with it.
Exceptions	\

2.2.4 Sequence diagrams

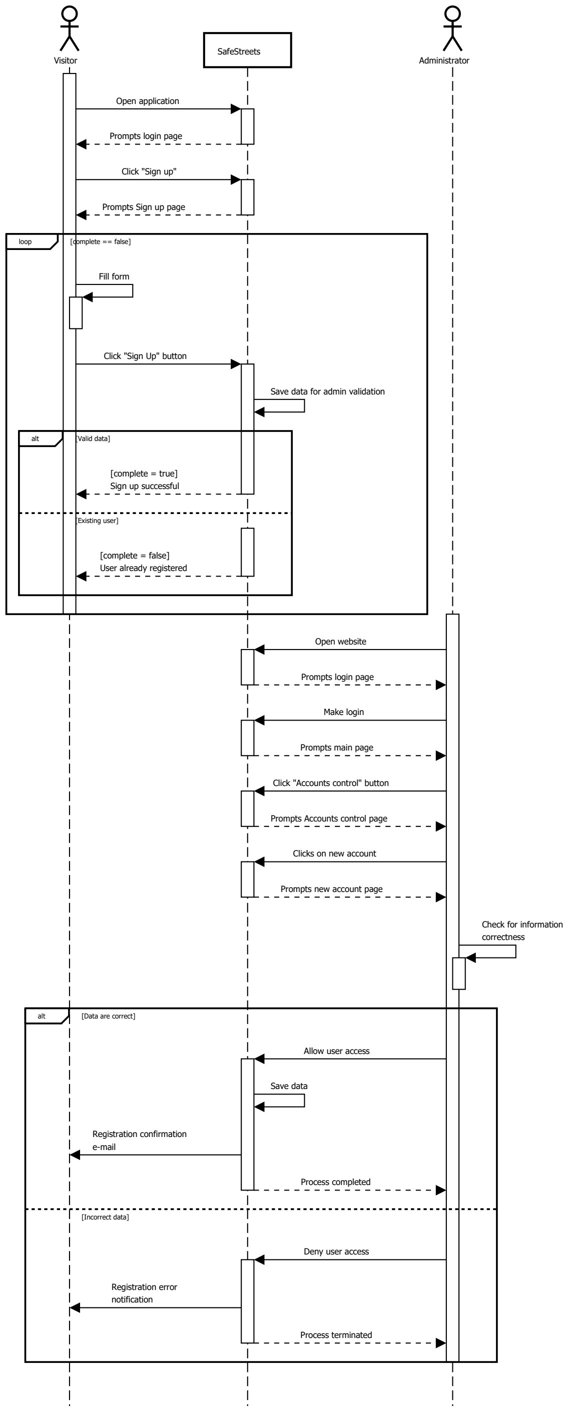


Figure 13: Sign up (user)

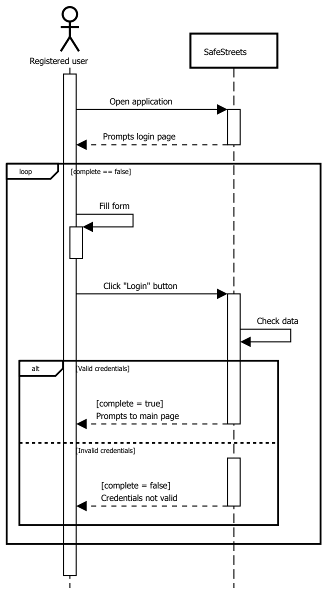


Figure 14: Login (application)

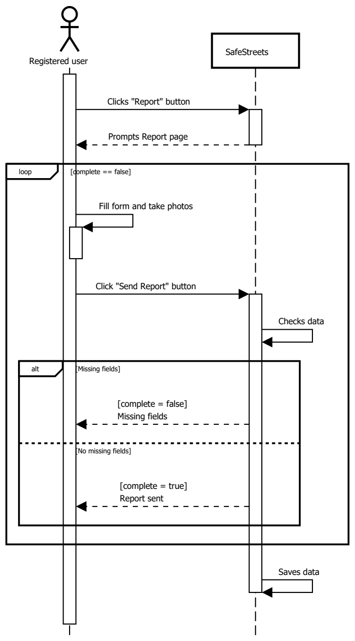


Figure 15: Create report

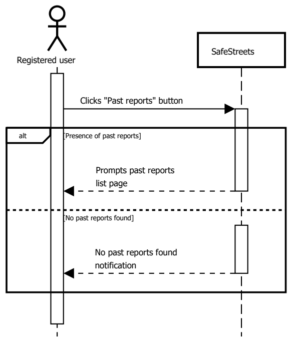


Figure 16: See past reports

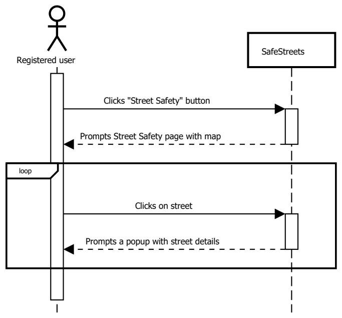

Registered user
SafeStreets
Clicks "Street Safety" button
Prompts Street Safety page with map
loop
Clicks on street
Prompts a popup with street details

Figure 17: See Street Safety

2.3 Authority

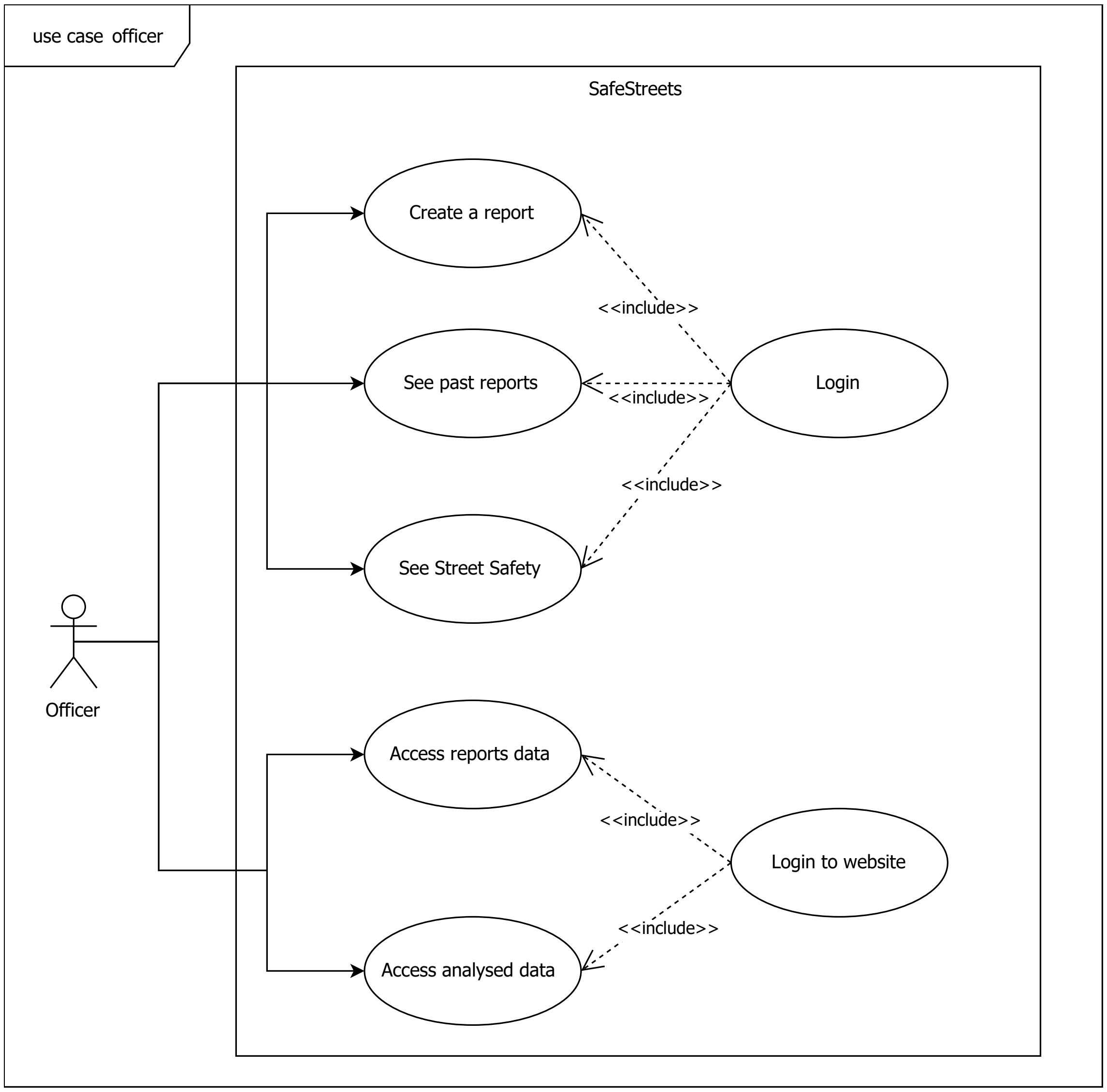
2.3.1 Scenarios

- ▷ *Scenario 3*
  Milan municipality has decided to adopt SafeStreets to increase street safety and encourage active citizenship. Traffic policemen's chief officer registers over the SafeStreets platform and requests an authority access. After receiving the accesses, he decides to start monitoring reports received through the system, noticing that lots of people signed up and started sending reports; moreover almost the totality of the reports are well-made and correctly categorised. The chief is very pleased of this system and congratulates with the municipality for the great choice.

- ▷ *Scenario 4*

After two months of continuous use of the platform, the chief officer decides to analyse the data retrieved by the platform and to verify if the suggestions elaborated by the system effectively answer real necessities. He decides to investigate a small area in the “college town” area of the city, already known to the municipality for the large number of bike-car accidents. The chief already presented a suggestion to the municipality to build a bike-only lane on the road to limit the number of accidents, but the street is full of parking spots and barriers can’t be built. After focusing on the area, he remains astonished after having read suggestions for the area: the system, in addition to noticing the large number of accidents reported, showed a very high number of reports for red light offences at the crossroad at the end of the street, with a suggestion of substituting it with a roundabout. The system says the roundabout will also reduce the number of accidents, because cars won’t be in hurry to cross the intersection, reducing the average speed.

2.3.2 Use case diagram



2.3.3 Use cases

Name	**Sign up**
Actor	Authority, not already registered
Entry conditions	The authority installed the application on his device and opens it / the authority opens SafeStreets website from a computer browser.
Events flow	1. The authority chooses the Sign Up option. 2. The authority fills in all the fields. 3. The authority provides a photo of his identification document. 4. The authority hits the Sign Up button. 5. The system saves the data. 6. The administrator validates user's data and accepts the registration. 7. The user notifies the administrator he is an authority (providing coherent documentation) and asks for authority access. 8. The administrator validates authority's data and upgrades the account.
Exit conditions	The authority has access to the system with specific permits.
Exceptions	▷ Some or all of the fields are not filled / no photo has been provided: the system prompts the user a notification and asks for a data filling. ▷ The user is already registered: the app notifies the user to contact an administrator to retrieve the password. ▷ The administrator refuses the registration (for basic or authority access): the system sends an e-mail to the user to notify the refusal and eventual motivations.

Name	**Login to website**
Actor	Registered user
Entry conditions	▷ The officer opens SafeStreets website within a browser. ▷ The officer has already completed the "Sign up" activity
Events flow	1. The user fills the E-mail address and Password fields with his credentials. 2. The user hits the Login button.
Exit conditions	Login is completed, the user is prompted to the main activity website page.
Exceptions	▷ The user insert wrong credentials: the system shows a notification and suggests to contact an administrator to retrieve the e-mail address and/or password. ▷ The user is not registered: for security reasons the system prompts the same alert for the Wrong credentials exception.

The *Login, Create a report, See past reports* and *See Street safety* activities are not reported, as equal to the User ones.

Name	**Access reports data**
Actor	Officer
Entry conditions	The officer has already completed the "Login to website" activity.
Events flow	1. The officer selects the "Consult reports" in the website menu. 2. The system retrieves the data about sent reports.
Exit conditions	The website shows all reports made by users and allows the officer to filter data and to download report photos.
Exceptions	▷ There are no reports: the application shows an empty page with "No reports found" notification in the middle.

Name	**Access analysed data**
Actor	Officer, municipality
Entry conditions	The officer has already completed the "Login to website" activity.
Events flow	1. The officer selects the "Consult analysed data" in the website menu. 2. The system retrieves the analysed data, organized for urgency.
Exit conditions	The website shows the analysed data, with acquired problems and elaborated suggestions.
Exceptions	\

2.3.4 Sequence diagrams

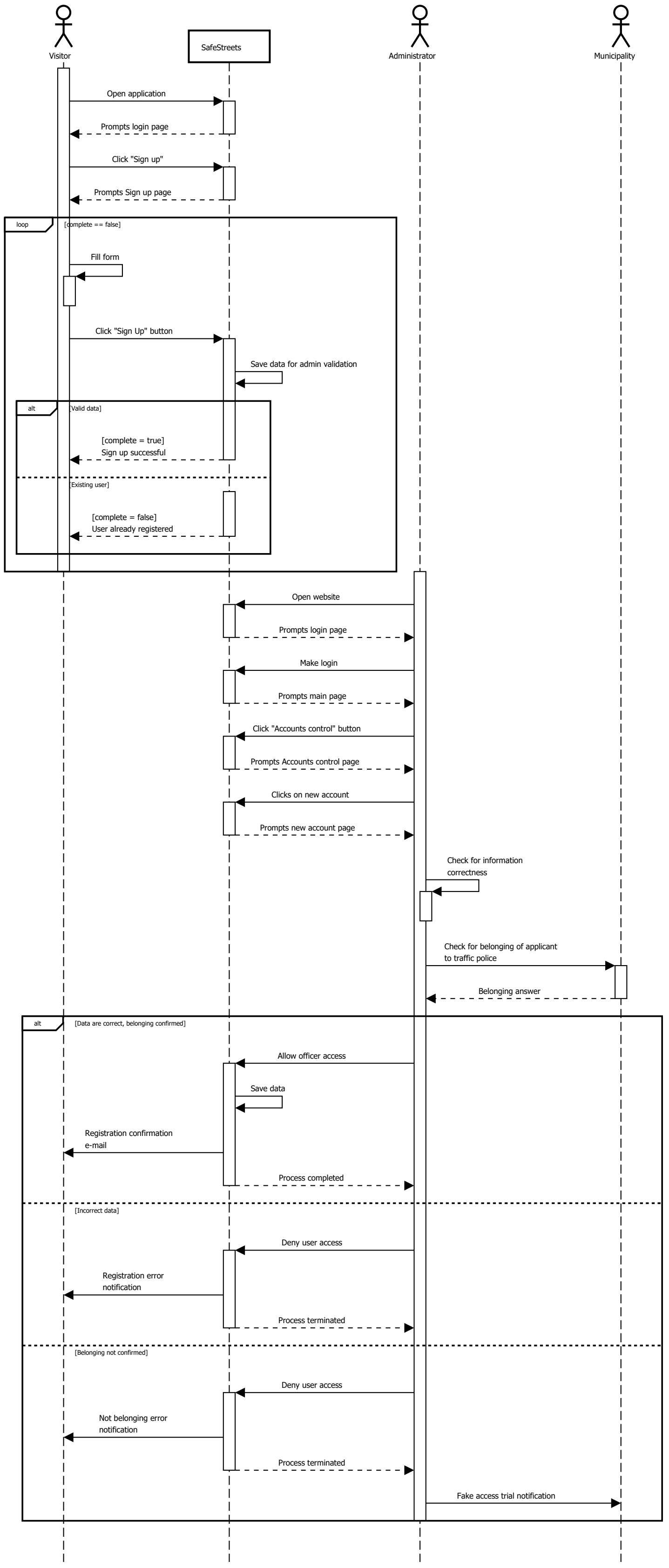


Figure 18: Sign up (officer)

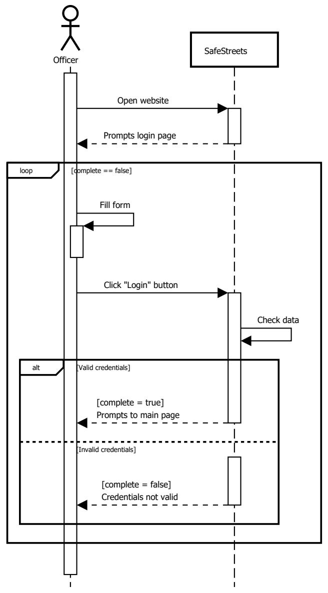


Figure 19: Login (website)

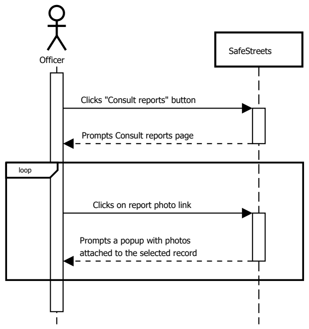

Officer
SafeStreets
Clicks "Consult reports" button
Prompts Consult reports page
loop
Clicks on report photo link
Prompts a popup with photos
attached to the selected record

Figure 20: Access reports data

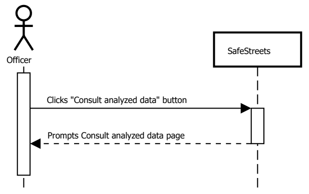

Officer
SafeStreets
Clicks "Consult analyzed data" button
Prompts Consult analyzed data page

Figure 21: Access analyzed data

2.4 Administrator

2.4.1 Scenarios

- *Scenario 5*
  Luigi is the system administrator of SafeStreets. He receives a notification for registration validation by a new user. After logging to the website, he opens the new user request and, after verifying that the provided fiscal code matches with the one reported on the document in the photo, grants the new user access to the platform.

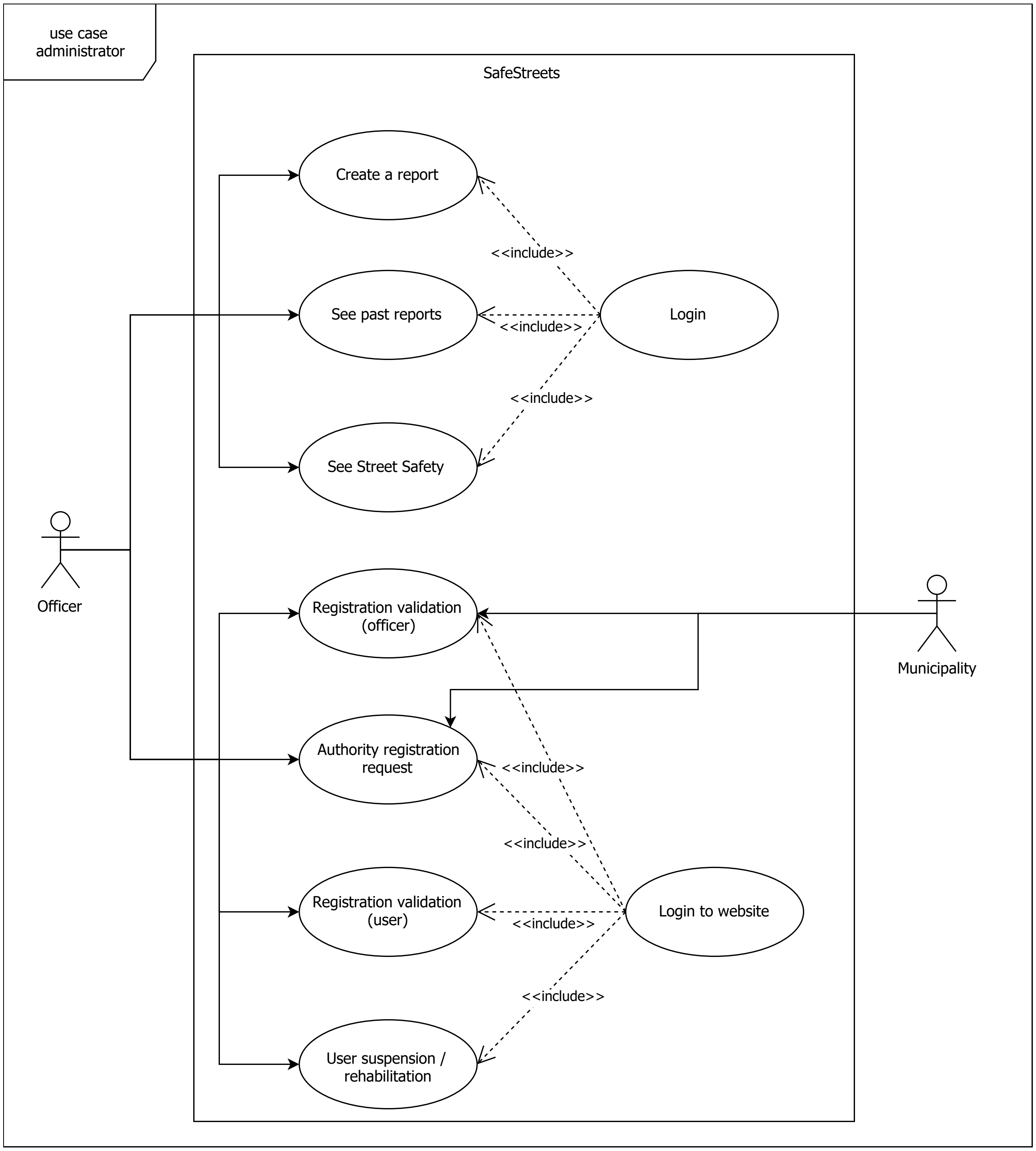
- *Scenario 6*
  Luigi receives a notification from the municipality about a user who continuously sends fake reports about double parking, with the request of deleting the user from the system. Luigi, to be sure not to re-grant access

to the user in case the latter tries to sign up again, suspends the user instead of deleting the account. This way he won't be able to access its account. Moreover, he won't be able to sign up again since his fiscal code is already registered.

▷ *Scenario 7*
Luigi receives an officer activation request from a new user, who declared to be an agent of the local traffic police. After receiving a confirmation of its identity from the municipality, Luigi grants him the Officer access.

▷ *Scenario 8*
SafeStreets administration receives a formal request from the municipality to activate an account for each agent of the traffic police, attaching the complete list of data needed for the registration to the request. The administration forwards the request to Luigi, who creates the requested accounts and sends an e-mail to each agent with temporary access credentials.

2.4.2 Use case diagram



2.4.3 Use cases

The *Create a report, See past reports* and *See Street safety* activities are not reported, as equal to the User ones. The *Login to website* activity is not reported, as equal to the Authority one.
There is no *Sign up* activity for Administrators because such accounts are created with the system initial instantiation directly by SafeStreets.

Name	**Registration validation (basic user)**
Actor	Administrator
Entry conditions	The administrator has already completed the “Login to website” activity.
Events flow	1. The administrator selects the “Accounts control” in the website menu. 2. The administrator selects the user to validate. 3. The administrator checks for information correctness (compared to the provided identification document photo). 4. The administrator validates the user.
Exit conditions	The user obtains the base access.
Exceptions	▷ No registered users: the website shows an empty page with “No users found” notification in the middle. ▷ User data not correct: the administrator refuses the validation clicking on “Refuse” button.

Name	**Registration validation (authority access request)**
Actors	Administrator, municipality
Entry conditions	The administrator has already completed the “Login to website” activity.
Events flow	1. The administrator selects the “Accounts control” in the website menu. 2. The administrator selects the user to validate and notices the Officer access request. 3. The administrator checks for information correctness (compared to the provided identification document photo). 4. The administrator verifies the identity of the user and the effective access permission asking to the municipality. 5. The administrator validates the user.
Exit conditions	The user obtains the base access.
Exceptions	▷ No registered users: the website shows an empty page with “No users found” notification in the middle. ▷ User data not correct: the administrator refuses the validation clicking on “Refuse” button. ▷ User not identified from municipality: the administrator refuses the validation and notices the user to the police office for fake declaration clicking on “Refuse and notice” button.

Name	**Authority registration request**
Actor	Administration, municipality
Entry conditions	▷ The administrator has already completed the “Login to website” activity. ▷ The administrator receives a formal request from the municipality requesting the activation of one or more accounts
Events flow	1. The administrator selects the “Accounts control” in the website menu. 2. For each account to be created, the administrator clicks on “New Officer account” button, fills in all the fields and finally hits the “Save” button. 3. The system sends an e-mail for each new account with the first access credentials.
Exit conditions	Requested accounts have been created and new officers can login.
Exceptions	▷ Missing data for one or more accounts: the administrator asks the municipality for missing data.

Name	**User suspension**
Actor	Administrator, authority
Entry conditions	▷ The administrator has already completed the “Login to website” activity. ▷ The administrator receives a suspension/cancellation request from the authority
Events flow	1. The administrator selects the “Accounts control” in the website menu. 2. The administrator selects the user to be suspended. 3. The administrator clicks on “Suspend” button and confirms the displayed popup.
Exit conditions	The selected user cannot access the service as long as his account is suspended.
Exceptions	\

Name	**User rehabilitation**
Actor	Administrator, authority
Entry conditions	▹ The administrator has already completed the “Login to website” activity. ▹ The administrator receives a rehabilitation request from the authority
Events flow	1. The administrator selects the “Accounts control” in the website menu. 2. The administrator selects the user to be rehabilitated. 3. The administrator clicks on “Rehabilitate” button and confirms the displayed popup.
Exit conditions	The selected user can access the service as it could before the suspension.
Exceptions	\

2.4.4 Sequence diagrams

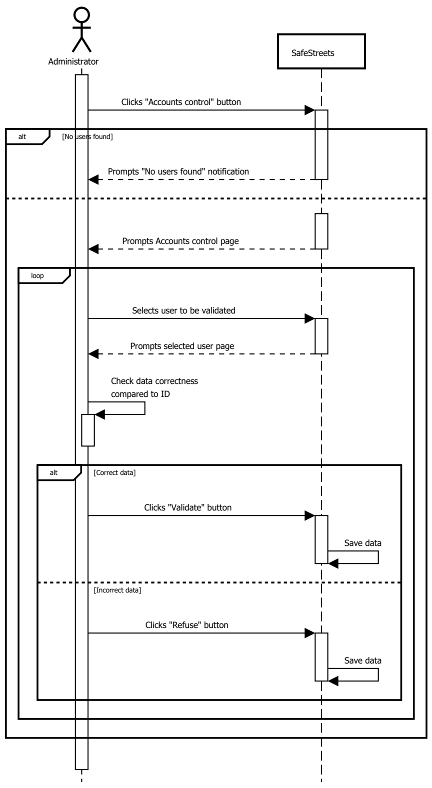

Administrator
SafeStreets
Clicks "Accounts control" button
alt
[No users found]
Prompts "No users found" notification
Prompts Accounts control page
loop
Selects user to be validated
Prompts selected user page
Check data correctness compared to ID
alt
[Correct data]
Clicks "Validate" button
Save data
[Incorrect data]
Clicks "Refuse" button
Save data

Figure 22: Registration validation (user)

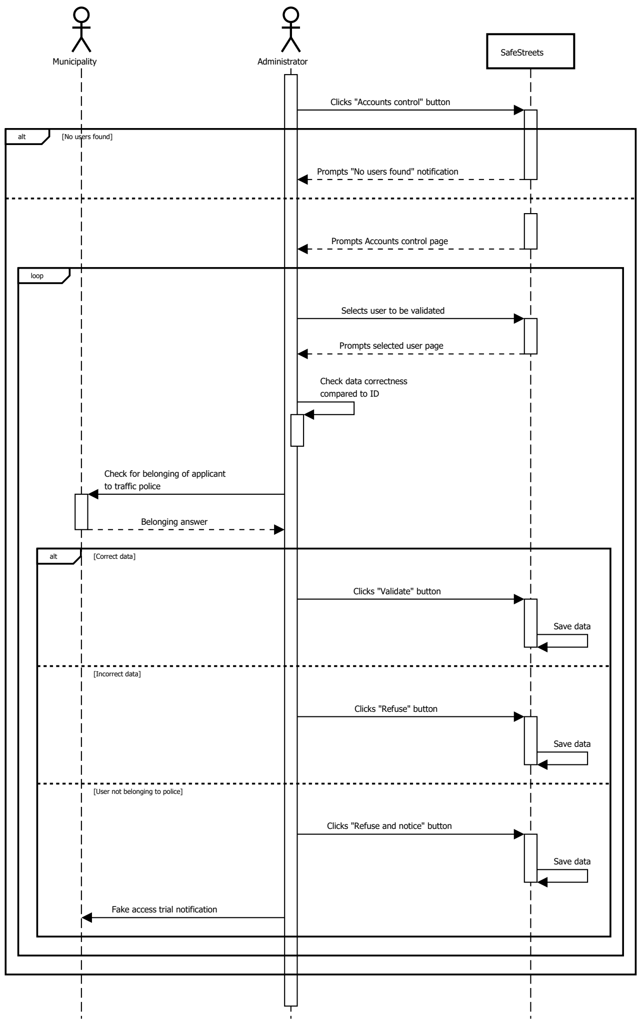

Municipality
Administrator
SafeStreets
Clicks "Accounts control" button
alt
[No users found]
Prompts "No users found" notification
Prompts Accounts control page
loop
Selects user to be validated
Prompts selected user page
Check data correctness
compared to ID
Check for belonging of applicant
to traffic police
Belonging answer
alt
[Correct data]
Clicks "Validate" button
Save data
[Incorrect data]
Clicks "Refuse" button
Save data
[User not belonging to police]
Clicks "Refuse and notice" button
Save data
Fake access trial notification

Figure 23: Registration validation (officer)

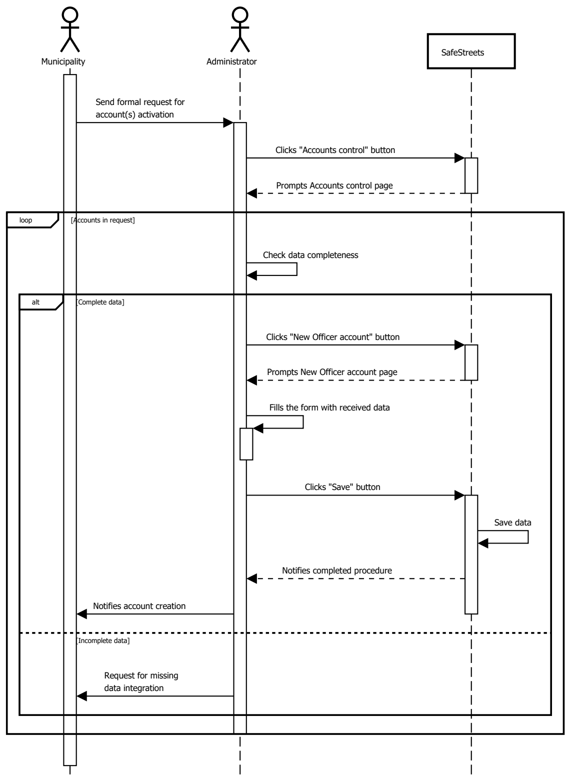

Municipality
Administrator
SafeStreets
Send formal request for account(s) activation
Clicks "Accounts control" button
Prompts Accounts control page
loop
[Accounts in request]
Check data completeness
alt
[Complete data]
Clicks "New Officer account" button
Prompts New Officer account page
Fills the form with received data
Clicks "Save" button
Save data
Notifies completed procedure
Notifies account creation
[Incomplete data]
Request for missing data integration

Figure 24: Authority registration request

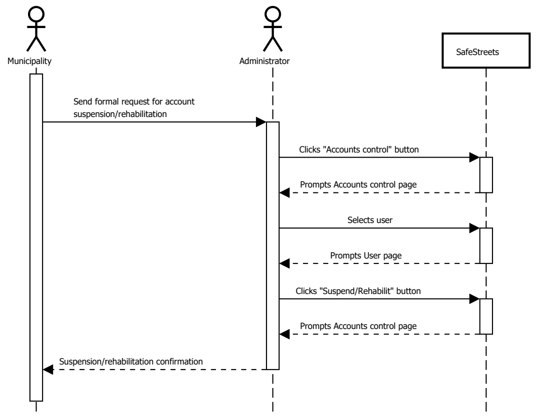

Municipality
Administrator
SafeStreets
Send formal request for account suspension/rehabilitation
Clicks "Accounts control" button
Prompts Accounts control page
Selects user
Prompts User page
Clicks "Suspend/Rehabilit" button
Prompts Accounts control page
Suspension/rehabilitation confirmation

Figure 25: User suspension / rehabilitation

2.5 Requirements

2.5.1 [G1] Allow citizens to register providing basic information that certifies their identities and become certified users

▷ [R1] Every citizen that wants to register has a valid identification document and a smartphone (with GPS and camera systems).

- ▷ [R2] Every citizen that wants to register owns a valid e-mail address.

- ▷ [D1] User's identification data is correct and verified.

2.5.2 [G2] Allow authorities to register through a verification procedure

- ▷ [R1] Every citizen that wants to register has a valid identification document and a smartphone (with GPS and camera systems).
- ▷ [R2] Every citizen that wants to register owns a valid e-mail address.
- ▷ [R3] The municipality allows SafeStreets administrator to verify the belonging of the requesting officer to the traffic police (via an automated system or sending an official request).

- ▷ [D1] User's identification data is correct and verified.

2.5.3 [G3] Enable certified users to report new traffic or parking violations

- ▷ [R4] The user has his smartphone with him, is able to login in the application and is not suspended.
- ▷ [R5] The user's smartphone has Internet connection enabled and functioning.
- ▷ [R6] The user's smartphone has GPS enabled and functioning.
- ▷ [R7] The user is able to take at least one picture of the violation to correctly choose the type of violation he is reporting.

- ▷ [D2] Location data in reports is precise enough to detect the name of the street where the violation took place and a rough estimation of the nearest civic number.
- ▷ [D3] One or more of the images provided with the report are sufficient to recognize the offender's car and visually verify which violation took place.
- ▷ [D5] User's reports are good willed and do not contain fake information on purpose.

2.5.4 [G4] Enable certified users to see their own past reports

- ▷ [R4] The user has his smartphone with him, is able to login in the application and is not suspended.
- ▷ [R5] The user's smartphone has Internet connection enabled and functioning.
- ▷ [R8] The user has already made at least one report.

2.5.5 [G5] Enable certified users to access the map showing the most reported streets

- ▷ [R4] The user has his smartphone with him, is able to login in the application and is not suspended.
- ▷ [R5] The user's smartphone has Internet connection enabled and functioning.

2.5.6 [G6] Enable authorities to access all certified users services

- ▷ [R9] The officer access has the same base functions of the basic user access, plus specific ones.

2.5.7 [G7] Enable authorities to access to all report data (including data about the user who created it)

- ▷ [R10] The officer has a computer with Internet access and is able to login to the system.

2.5.8 [G8] Enable authorities to acquire report data in their own systems to generate tickets

- ▷ [R11] The municipality and/or the authorities use the provided APIs.

- ▷ [D2] Location data in reports is precise enough to detect the name of the street where the violation took place and a rough estimation of the nearest civic number
- ▷ [D3] One or more of the images provided with the report are sufficient to recognize the offender's car and visually verify which violation took place
- ▷ [D4] It's always possible for authorities to identify the offender either by its appearance in a photo or by ownership of the vehicle through the license plate

2.5.9 [G9] Enable authorities to access to data analysis results with suggestions

- ▷ [R10] The officer has a computer with Internet access and is able to login to the system.

2.5.10 [G10] Enable administrators to certify authorities

- ▷ [R3] The municipality allows SafeStreets administrator to verify the belonging of the requesting officer to the traffic police (via an automated system or sending an official request).
- ▷ [R12] The administrator has a computer with Internet access and is able to login to the system.
- ▷ [R13] The administrator has the possibility to use an official communication channel with the municipality.

2.5.11 [G11] Enable administrators to block or remove users

- ▷ [R12] The administrator has a computer with Internet access and is able to login to the system.
- ▷ [R14] The administrator has received a block/removal request from the authorities.

2.5.12 [G12] Enable administrators to assign roles to users

- ▷ [R12] The administrator has a computer with Internet access and is able to login to the system.
- ▷ [R13] The administrator has the possibility to use an official communication channel with the municipality.
- ▷ [R15] The administrator has received a sign-up request from the authorities and/or from the municipality.

2.5.13 Traceability Matrix

Requirement	***Goal(s)***	***Use case(s)***
R1	G1, G2	Sign up (user / authority)
R2	G1, G2	Sign up (user / authority)
R3	G2	Sign up (authority)
R4	G3, G4, G5	Login, Create a report, See past reports, See Street Safety
R5	G3, G4, G5	Create a report, See past reports, See Street Safety
R6	G3	Create a report
R7	G3	Create a report
R8	G4	See past reports
R9	G6	Login
R10	G7, G9	Login to website, Access reports data, Access analysed data
R11	G8	*No use case provided - usage outside the system*
R12	G10, G11, G12	Login to website
R13	G10, G12	Registration validation (officer), Authority registration request
R14	G11	User suspension, User rehabilitation
R15	G12	Registration validation (officer)

3 Performance requirements

The application should be light and fluid for a good user experience on all kind of supported device. The background data retrieval such as GPS loading, image upload or post download, has to take place in advance (e.g. update GPS data while taking the picture, not after; preload the next #x posts; ecc.).
In addition:

- ▷ There are no limits to the number of simultaneously connected users;
- ▷ Latency must be stable for different amount of users load and under a reasonable time;
- ▷ The app won't make blocking requests such as for the upload of a picture, these kind of requests needs to run in background;
- ▷ The AI algorithm responsible for the extraction of data from pictures has to run on devices only if dedicated hardware is provided; otherwise it will be run on SafeStreet's servers.

4 Design constraints

4.1 Standards compliance

SafeStreets will be developed keeping a strict division between what the backend and what the frontend will do. RESTful APIs will be implemented to allow reliable and secure HTTPS connections between the servers and the applications.
The iOS application will be compliant with the App Store Guidelines, as of the date of this document, described here: `https://developer.apple.com/app-store/review/guidelines/`.
The Android application will be compliant with the Play Store Guidelines, as of the date of this document, described here: `https://play.google.com/about/developer-content-policy/`.

4.2 Hardware limitations

The mobile application needs access to: geolocation services, camera and an internet connection. No other hardware limitations should be in act.

4.3 Any other constraint

iOS requires at least the OS version named iOS 9.1.
Android requires at least the OS version named Android 6.0 Marshmallow.

5 Software system attributes

5.1 Availability and Reliability

▷ SafeStreets should provide a guarantee of 99+% uptime for its services. This can be guaranteed by using managed cloud services, such as the ones provided by Amazon AWS. The provider masks and manages hardware faults with virtualization and replication for high reliability.
The web backend will benefit of the same guarantees since it will be hosted on the same services the system's logic is.

▷ The mobile applications will always be available on the PlayStore (Android) and on the AppStore (iOS) simultaneously. Its functionalities rely on the availability of the private APIs used to communicate with the server.

▷ The reliability of the service is also granted by the automatic testing executed on every update of the infrastructure (both frontend and backend).

5.2 Security

Several aspects of the system require a very high level of security both in data storage and transmission over various channels.

▷ Since the system will need to manage very sensitive data (user data, traffic tickets and accidents in the municipality), the best solution is to use cloud storage provided by a reputable provider. This assures the provider will keep up with the latest security patches used for the involved technologies.

▷ Every transfer of data (between applications and the private API, between external APIs such as the municipality ones) will be end-to-end encrypted with state of the art protocols such as HTTPS with TLS ≥ 1.2.
By certifying SafeStreets' and the municipality's public keys, the system will assure authenticity and integrity of the data on top of making it virtually impossible to understand in case of interception.

5.3 Maintainability

The software infrastructure will be developed using consolidated and well documented technologies:

▷ The main interaction of the on an Apache web server, with MySQL as a database server.

▷ The Xamarin Framework will be used for the development of the mobile application in order to be able to create both the Android and the iOS version with a single codebase.

▷ The web backend will be developed using a mature web framework such as Semantic UI.

5.4 Portability

▷ The frontend, built using Xamarin Forms needs to be written only once for both the platforms. This will decrease the time spent on the mobile application side up to 35%. Xamarin Forms also supports build for Windows, Mac and Tizen; platforms that we can target if requested.

▷ Building private RESTful APIs for the Server-to-Application communication with PHP allows the deployment of small components with great speed (compared to solutions based on Java Tomcat, for example) and the possibility to choose the underlying operating system (compared to solutions based on ASPX with .NET, only available on Windows Server).

Part IV
Formal analysis using Alloy

1 Alloy

1.1 Data Types

```
open util/integer
open util/boolean

--Representing time as integer allows us to make comparisons
sig Time{
    --The unix time is a format for representing time as the number of seconds passed
    --from the 1st of January 1970 and is widely used format for storing time data in
    --databases
    unixTime: one Int
}
{
        unixTime >= 0
}

one sig Never extends Time {}
{
        unixTime = 0
}

sig Username {}

--Avoid spawning useless instances not associated with any entity
fact validUsername{
        all u: Username |
        one us: User | us.username = u
}

sig Email {}

--Avoid spawning useless instances not associated with any entity
fact validEmail{
        all e: Email |
        one us: User | us.email = e
}

sig FiscalCode {}

--Avoid spawning useless instances not associated with any entity
fact validFiscalCode{
        all fc: FiscalCode |
        one us: User | us.fiscalCode = fc
}

sig Municipality {}

--Avoid spawning useless instances not associated with any entity
fact validMuniciplity{
        all m: Municipality |
                (one a: Authority | a.officeCity = m)
                or
                (one r: Report | r.city = m)
}

sig LicensPlate {}

--Avoid spawning useless instances not associated with any entity
fact validLicensePlate{
        all lp: LicensPlate |
        one r:Report | r.licensePlate = lp
}

sig Coord{
        latitude: one Int,
        longitude: one Int
}
```

1.2 Data

```
abstract sig Person{
    fiscalCode: one FiscalCode
}

sig User extends Person{
        username: one Username,
        email: one Email,
        created: one Time,
        blocked: one Bool,
        blockedDate: one Time,
        accepted: one Bool,
        accepterAdmin: lone Administrator
}{
    --Can't be blocked before being created
    (blocked = True) implies (blockedDate.unixTime > created.unixTime)
        created != Never
        (blockedDate = Never) iff (blocked = False)
        (accepted = True) implies (#accepterAdmin = 1)
}

sig Authority extends User{
        officeCity: one Municipality
}

sig Administrator extends User{}
{
        accepted = True
        accepterAdmin = System
}

one sig DummyFiscalCode extends FiscalCode{}

one sig System extends Administrator {}
{
        fiscalCode = DummyFiscalCode
}
```

```
sig Report{
        reportId: one Int,
        reporter: one User,
        date: one Time,
        position: one Coord,
        licensePlate: one LicensPlate,
        city: one Municipality,
        acknowledgedBy: one Authority
}
{
        date != Never
        reporter != System
}
```

1.3 Facts

```
fact uniqueUsername{
        no disj u1, u2: User | u1.username = u2.username
}

fact uniqueUserEmail{
        no disj u1, u2: User | u1.email = u2.email
}

fact uniqueFiscalCode{
    no disj u1, u2: User | u1.fiscalCode = u2.fiscalCode
}

fact uniqueReportId{
        no disj r1, r2:Report | r1.reportId = r2.reportId
}

--No multiple reports for a single accident by the same user
fact uniqueUserReport{
        no disj r1, r2: Report |
                r1.licensePlate = r2.licensePlate and
                r1.reporter = r2.reporter and
                r1.date = r2.date
}

--A user can't have reports sent after he was blocked
fact noReportWhenBlocked{
        no r: Report, u: User |
                u.blocked = True and
                r.date.unixTime >= u.blockedDate.unixTime
}

--A user has been confirmed by an administrator to be able to report
fact noReportIfNotAccepted{
        no r: Report, u: User |
                u.accepted = False and
                r.reporter = u
}

--The report has to be sent after the user's registration
```

```
fact noReportBeforeRegistration{
        no r: Report, u: User |
                r.date.unixTime < u.created.unixTime and
                r.reporter = u
}

--no multiple reports for a single accident by the same user
fact uniqueUserReport{
        no disjoint r1, r2:Report |
                r1.licensePlate = r2.licensePlate and
                r1.reporter = r2.reporter and
                r1.date = r2.date
}
```

1.4 Assertions

```
--Multiple users can report a single accident
assert multipleUsersReports{
        some disj r1, r2: Report |
                r1.licensePlate = r2.licensePlate and
                r1.reporter != r2.reporter and
                r1.date = r2.date
}

assert someReportsHaveBeenAcknowledged{
        some r:Report, a:Authority |
                r.acknowledgedBy = a
}
```

1.5 Dynamic model

```
pred show
{
        #User = 2
        #Report > 1
}

pred isUserAllowedToReport[u: User]
{
    (u.blocked = False) and (u.accepted = True)
}

--A normal user reports an Accidents
pred reportAnAccident[r:Report, u:User]{
        --precondition
        isUserAllowedToReport[u]

        r.reporter = u
        --The user was not an Authority nor an Admin
        all a: Authority, b: Administrator |
                a != u and
                b != u
}

pred multipleReportsSameVehicle[ r1,r2:Report]{
        r1.licensePlate = r2.licensePlate
}
```

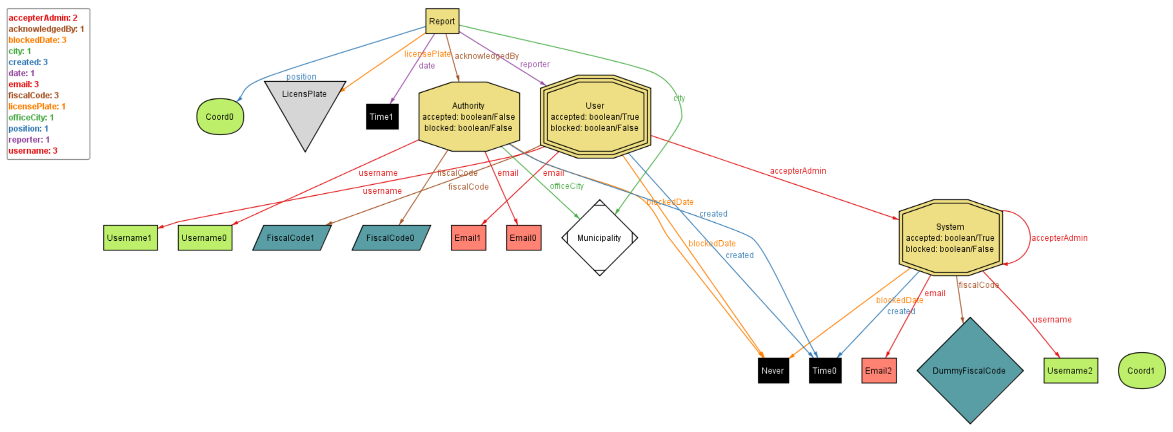
2 Results

```
run show for 2
run isUserAllowedToReport for 4
run reportAnAccident for 10
run multipleReportsSameVehicle for 10
```

2.1 Proof of Consistency

```
Executing "Run show for 2"
Solver=sat4j Bitwidth=4 MaxSeq=2 SkolemDepth=1 Symmetry=20
2782 vars. 201 primary vars. 7603 clauses. 469ms.
Instance found. Predicate is consistent. 154ms.

Executing "Run isUserAllowedToReport for 4"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
8377 vars. 515 primary vars. 21356 clauses. 185ms.
Instance found. Predicate is consistent. 151ms.

Executing "Run reportAnAccident for 10"
Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
41886 vars. 2085 primary vars. 104885 clauses. 444ms.
Instance found. Predicate is consistent. 783ms.

Executing "Run multipleReportsSameVehicle for 10"
Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
41715 vars. 2085 primary vars. 104841 clauses. 316ms.
Instance found. Predicate is consistent. 332ms.

4 commands were executed. The results are:
#1: Instance found. show is consistent.
#2: Instance found. isUserAllowedToReport is consistent.
#3: Instance found. reportAnAccident is consistent.
#4: Instance found. multipleReportsSameVehicle is consistent.
```

2.2 Generated World



Part V
Effort spent

Part	Hours
1. Introduction	3.5
2. Overall description	0
3. Specific requirements	3
4. Formal analysis using Alloy	9

Table 2: Andrea Aspesi's effort spent

Part	Hours
1. Introduction	1.50
2. Overall description	4.5
3. Specific requirements	8
4. Formal analysis using Alloy	1.5

Table 3: Elia Battiston's effort spent

Part	Hours
1. Introduction	2
2. Overall description	3
3. Specific requirements	10
4. Formal analysis using Alloy	0.5

Table 4: Alessandro Carabelli's effort spent