POLITECNICO MILANO 1863

Requirements Analysis and Specifications Document

Software Engineering II Project - A.Y. 2019-2020

SafeStreets

Authors	ID Numbers
Andrea Furlan	944774
Cosimo Russo	945891
Giorgio Ughini	944710

November 3, 2019 Version 1.0

Contents

1 Introduction

1.1 Purpose

The main purpose of SafeStreets is to create a software that provides users the possibility to notify authorities when parking violations occur providing some useful features such as finding the most unsafe areas around them and proposing suggestions to the municipality. In addition, SafeStreets will enable the Local Police to generate traffic tickets from it and to cross all the information it owns with the data of the accidents happened.
Specifically, we want to realize a product which is able to:

- Retrieve pictures uploaded by users of parking violations with possible attached information such as license plate position in the image, type of violation and GPS meta-data.

- Automatically complete the data of a reported violation running a recognition algorithm able to read license plate text.

- Highlight to users the areas with the highest frequency of violations and information about vehicles that commit most violations.

- Automatically identify potentially unsafe areas crossing SafeStreets' information with accident data from the Local Police, possibly suggesting possible interventions.

- Send violations data to the Local Police to automatically create new traffic tickets if it can be proved that the chain of custody of the information coming from the users is never broken.

- Generate statistics related to ticket emissions to inform users about how effective SafeStreets is.

On the other hand, the purpose of this paper is to define in a detailed way all the functions and requirements of the application.
In doing this, we start focusing on a brief overview to characterize the product with relevance to its interaction with the world, then we will proceed deeply in analysing which functions are relevant and should be provided, and which requirements are needed to the stakeholders.

1.1.1 Goals

- **[G1]** Notifing officers when particular parking violations occur.
- **[G2]** Permitting both users and officers to learn which areas have the highest frequency of violations.
- **[G3]** Permitting both users and officers to learn which vehicles commit the most violations.
- **[G4]** Suggesting possible interventions to potentially unsafe areas.
- **[G5]** Allowing the local police to generate traffic tickets from SafeStreets data.
- **[G6]** Building and exhibiting statistics.

1.1.2 Traceability Matrix

Since goals, functions and constraints are related to each other a traceability matrix is provided in order to enlight the various relationships among them.

Goal ID	Functions ID	Constraints ID	UseCases ID
G1	F1	C1,C2,C3,C4,C6	-
G2	F2	C8,C11	-
G3	F2	C5,C10	-
G4	F2,F3	C9	-
G5	F4	C2,C7	-
G6	F5	C12	-

Table 1: Traceability Matrix

1.2 Scope

As our software needs to be compliant with different laws and as it needs to interact with the Local Police, initially, SafeStreets will have a restricted geographic domain coincident with the Italian city of Milan.
Indeed, in order to provide the most complete service, SafeStreets will require the access the Local Police web application to be able to process traffic tickets.
It goes without saying that to organize this kind of service in the most effective way we must experiment first this activity in a internationally-visible city, then applying that to anyone who will demand.

1.2.1 The world-machine phenomena

The first model of our system to be presented is the model "The world and the machine" by M. Jackson and P. Zave. This model highlights the division between phenomena that happen entirely either in the world or in the machine, and those that are shared between the two of them.

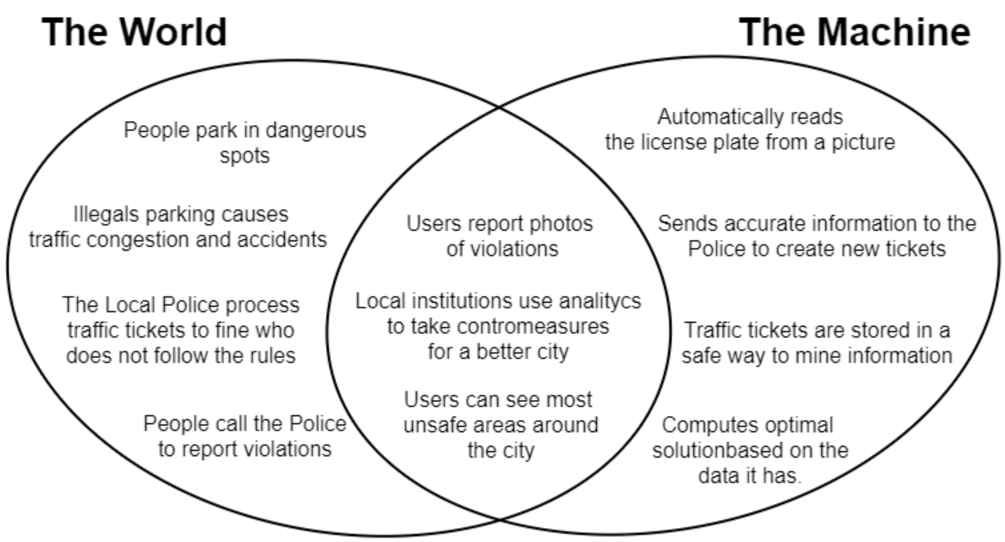

Figure 1: The world-machine phenomena chart.

1.3 Definition and Acronyms

1.3.1 Definitions

- *Data Mining*: The process of discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems.
- *Report*: The data that an user has provided to the authorities that witness a violation by a vehicles.
- *Violation*: The general infraction that a vehicle has done and that has been reported by a SafeStreets user.

- *Guest*: This actor plays the role of a person who is not registered and thus logged in.
- *User*: This actor refers to the condition of a normal person (not an officer) already signed up and logged.
- *Officer* or *Authority*: This actor represents a public officer that interacts with SafeStreets in some ways.

1.3.2 Acronyms

- *AI*: Artificial Intelligence
- *API*: Application Programming Interface
- *CMS*: Content Management System
- *IEEE*: Institute of Electrical and Electronic Engineers
- *GPS*: Global Positioning System
- *OCR*: Optical Character Recognition
- *TLS*: Transport Layer Security
- *UML*: Unified Modeling language

1.4 References

- The 2019-2020 Software Engineering 2 Project Assignment document
- The IEEE Standard for RASD
- The official Alloy documentation

1.5 Document Structure

This document is divided in four parts:

- **Introduction**: a description about the goals of SafeStreets and the context in which it will be implemented is provided. A subsection dedicated to the understaing of some acronyms and definitions is also present.
- **Overall Description**: gives an overall description of SafeStreets, focusing on the domain assumptions and the constraints of the application. This section also aims to provide a context to the whole project and to show its integration with the real world. It also shows the possible interactions between the world and the users of SafeStreets.
- **Specific Requirements**: the software requirements, explained in a sufficiently detailed manner to design a system that satisfies them, and the testers to test said requirements are provided.

  A detailed description of the possible interactions that can occur between the world and the system is also present, followed by a series of simulations and previews about the interactions mentioned above.
- **Formal Analysis using Alloy**: the requirements are expressed through the Alloy model, which, being it is a declarative specification language, makes it possible to define the functions, the constraints and the interactions of SafeStreets.

In the last part of the document a short note about the softwares used and the effort spent in producing this RASD by its authors is shown.

2 Overall description

2.1 Product perspective

The idea is to create an application to allow users to report parking violations without taking too much time from their daily life.
We want therefore to realize an extremely friendly user interface and a lightweight software in order to make SafeStreets affordable to as many people as possible and to be able to run by many devices.

Users will certainly be able to exploit the advanced functions of SafeStreets such as charts and analitycs. However, since those functions rely on data, the basic violations reporting function will be the core one.

Since a small downtime of SafeStreets is not going to cause damage to anyone, it will be tolerated. On the other hand, as our software is going to run some kind of OCR and AI recognition algorithm that will probably be expensive in terms of resources, it should be very dynamic to support different queries in a few seconds.
In addition, our software is going to process very specific data that could potentially lead someone to be fined, hence it should ensure that the chain of custody of a report is never broken and the images are never altered.

To upload a new picture on SafeStreets or to view charts about violations, a functional internet connection is required.

The officers appointed to check SafeStreets violations reports will use an external CMS developed by the municipality, that will access SafeStreets data through APIs, since a dedicated section for the authorities directly on the app could be exploited by malicious users.

Concerning the hardware, we intend to have a storage system which contains all the historical information about reports made by the users, most likely a database or a data warehouse. This system will allow both users and officers to query for aggregated and detailed information which require a huge amount of data to be processed.

2.2 Class Diagram

This diagram provides a high-level view of the application, showing its main actors and functions and their interactions.

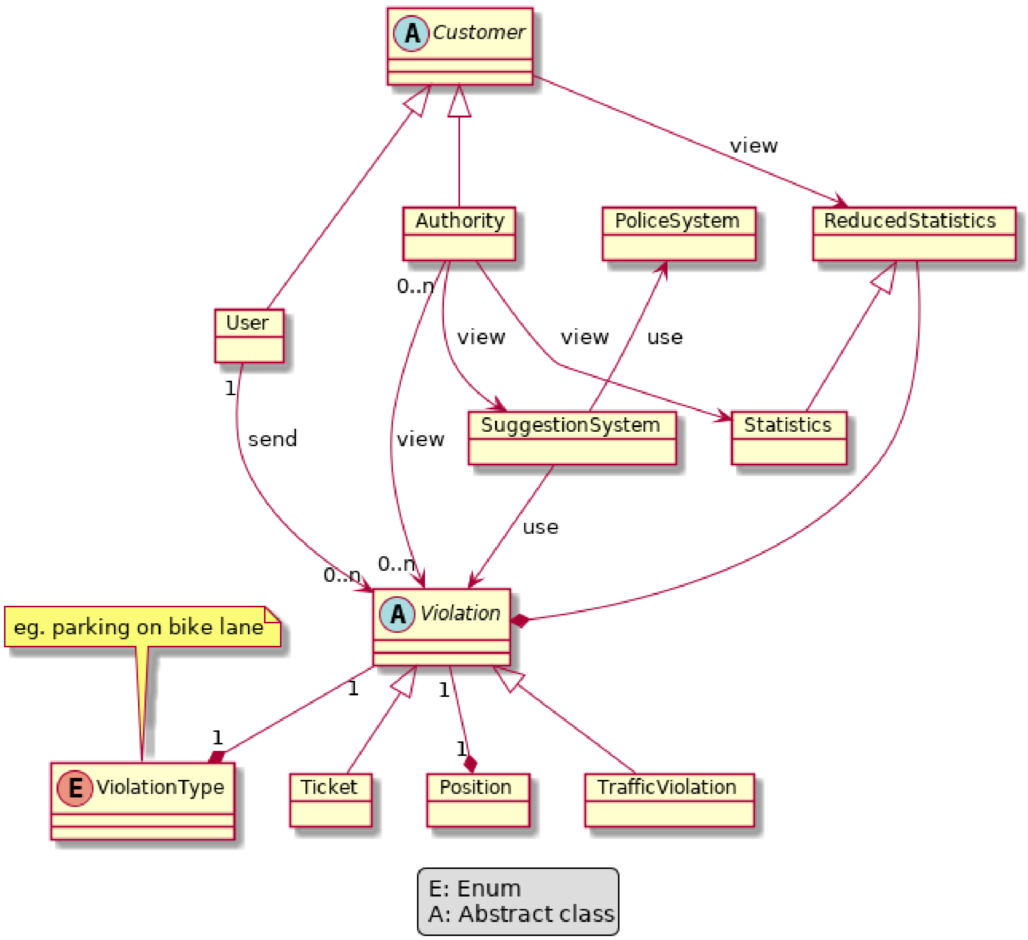

Figure 2: The class diagram of SafeStreets.

2.3 Product functionalities

This section provides an abstract of the main functions of the application. To be able to use any of the given functionalities, the user must first register and then login to the application by providing a valid email and a password.

2.3.1 [F1] Notification of Violations

The base function of the application is the possibility to signal a traffic violation.

The user must send one or more picture(s) of the car in which both the violation and the license plate are clearly visible.

The application will try to automatically get the user position using its GPS system, and will notify the user in case of failure so that it can enter it manually.

The users will then send the following information to the system:

- The pictures selected by the user
- The position of the user
- The current date and time
- The type of violation (to be picked from a pre-defined list)
- An optional comment inserted by the user

The information sent by the user will be stored on persistent storage on the server and the officers will be able to see it on their clients.

2.3.2 [F2] Data Mining

The system will allow both users and officers to extract statistics about violations in the various areas/streets of the cities in the system, for example a user can find the areas in which most reports occurred in the last 3 months.

Data mining must take into account the privacy of the users. To guarantee an acceptable level of privacy, different roles are given to the users and the officers.

In particular, a user will only be able to see statistics provided by aggregated data, never he will see the absolute numbers but only percentages.

Officers will, instead, have the finest granularity: they will see all the information enriched by the actual number of violations and can drill down to the specific licence plates which committed the violations. They will also have more filters available with respect to the users, for example the possibility to see which cars committed most violations in a given period.

2.3.3 [F3] Request for interventions

The system will get information from the local police systems about incidents, including the location, the licence plates of the cars involved, and the infractions committed.

By crossing the data about the incidents with the segnalations from its users, SafeStreets will be able to find unsafe areas and also to suppose a reason for it and make suggestions. For example, if a road has many cyclists hit and many signals of cars parked on the bike lane, it can suggest to add a barrier between the parking lane and the road. The correlations between the infractions found on the police system and the ones on the SafeStreet system, along with the possible solutions, will be done automatically and can be revised by hand by a officer. An artificial intelligence can then help to calibrate when the system should launch a warning, training on the approval/rejection of the previous signals.

The officers responsible for handling these recommendations will see on their clients all the data about the signal, including the number of incidents and signals, and can decide to discard it or approve it, thus keeping it in the system for future reference. All the approved signals will be reachable by the officers, once they have been resolved they can be deleted from the list but will remain in an archive available for the AI.

2.3.4 [F4] Automatic Tickets

The user will be indirectly able to give tickets when he spots a violation. This works by adding to the functionality "Notification of Violations" the possibility to confirm that he is sure that he is signalling an actual violation and wants to give a ticket. In this case the system reads the licence plate and uses the police system to automatically issue a ticket.

Security is a key aspect of this functionality. The system must be sure that the signal has not been modified. To achieve this, all messages are encrypted with TLS and an hash of every picture is included in the message.

2.3.5 [F5] Statistics

The system will also be able to build statistics using the violation and ticket systems. These data will represent the amount of tickets given in a certain street/area in a given period, the most egregious offenders, and the change in the number of tickets in a certain place during time. Just like in the Data Mining section, users will not be able to obtain information on single people, they will only see aggregated data and percentages instead of total numbers. The officers will instead have the possibility to see the actual numbers and to get information about specific license plates.

2.3.6 Traceability Matrix

Since goals, functions and constraints are related to each other a traceability matrix is provided in order to enlight the various relationships among them.

Goal ID	Functions ID	Constraints ID	UseCases ID
G1	F1	C1,C2,C3,C4,C6	-
G2	F2	C8,C11	-
G3	F2	C5,C10	-
G4	F2,F3	C9	-
G5	F4	C2,C7	-
G6	F5	C12	-

Table 2: Traceability Matrix

2.4 User characteristics

The users of SafeStreets can be both males or females of any age with no particular limitations. Of course, said users should have at least a basic knowledge of smartphones and electronic devices in general, especially on how to make photos and videos. Users should also have an e-mail address, primarily used to register and authenticate themselves.
Officers instead can also be both males or females but they need to be actual public officers enrolled in the Local Police. They should have a medium knowledge of networks, softwares and hardwares as well as a fully understanding of how traffic tickets laws can be applied, and they should be capable of using municipality tools and softwares fluently.

2.5 Assumptions and dependencies

2.5.1 Domain Assumptions

- A user should input only correct data when reporting a violation, for example the license plate position on the picture(s) shot should be correct
- The image and picture meta-data aren't altered by the user who first submit the report
- The municipality will provide provide correct and complete information about the accidents
- The municipality services are supposed to be functional during the uptime of SafeStreets.

2.5.2 General Assumptions

- A user cannot access any of the functionality of the application if he does not register first. The registration approach will be similar to the one used by social media: the user will provide his first and last name, hit birthday, an email, a password and a profile picture. All these data will be automatically approved during the registration phase, and the user will be able to use the SafeStreets functions right away. An user can be banned at any time if it uses the application in a non-compliant way, for example if he sends a fake violation report.
- Since users could be accountable for what they do with the app (they could be banned), we suppose that they will use SafeStreets in the most responsible way. For this reason, all tickets that are inserted correctly will be automatically approved.

2.5.3 Constraints

- **[C1]** It is impossible to send a report without attaching a picture, a description and selecting a reason that represents the violation.
- **[C2]** It is impossible to cancel an already sent ticket.
- **[C3]** It is prevented to attach a non-valid address to a report.
- **[C4]** It is prevented to send a report with a picture that contains non-readable license plate.
- **[C5]** Searching the license plate of a person in the worst driver section is forbidden.
- **[C6]** A report sent by an user must be taken in consideration by an officer in a reasonable time.
- **[C7]** An automatic ticket should be double checked by an officer before sending it to the violator.
- **[C8]** In the unsafe areas section, only the areas judged as unsafe can be searched.
- **[C9]** A suggestion for a possible intervention is destined to the authorities.
- **[C10]** The worst drivers are grouped by violation type.
- **[C11]** An unsafe areas is determined as such if it has received a consistent amount of certified violations in the last month.
- **[C12]** Statistics are generated using both accepted and denied violation reports.

2.5.4 Traceability Matrix

Since goals, functions and constraints are related to each other a traceability matrix is provided in order to enlight the various relationships among them.

Goal ID	Functions ID	Constraints ID	UseCases ID
G1	F1	C1,C2,C3,C4,C6	-
G2	F2	C8,C11	-
G3	F2	C5,C10	-
G4	F2,F3	C9	-
G5	F4	C2,C7	-
G6	F5	C12	-

Table 3: Traceability Matrix

3 Specific requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Concerning the user interface requirements, a series of mockups that represent how the application layout should be is provided. Changes that won't alter SafeStreets functioning can be applied during the implementation.
The sketches of the main pages of SafeStreets are designed with the aim to make the application as user friendly as possible. Still, since the following mockups are more concepts than rules, the design could take a more technical path.

3.1.2 Software Interfaces

Considering the domain of our application, we decided to integrate in our project some software components to create an easier-to-use product.
In order to provide users the ability to automatically insert location info into their violation pictures meta-data, we chose to adopt a location provider service. That supplier should expose an API to retrieve the user exact address by his or her GPS position coordinates.
In addition, APIs provided by the municipality are required to retrieve real time information about the accidents that occurred on the territory of the municipality to allow information mining and suggestion proposing.
Finally, the Local Police is required to provide an API to retrieve information about violation coming from SafeStreets users to create and process traffic ticket from it.

Login Page:
Since logging into SafeStreets is mandatory, this is the first page that the user will face when the app does not recognizes him. Of course, cookies or sessions could avoid to force the user to login every time.

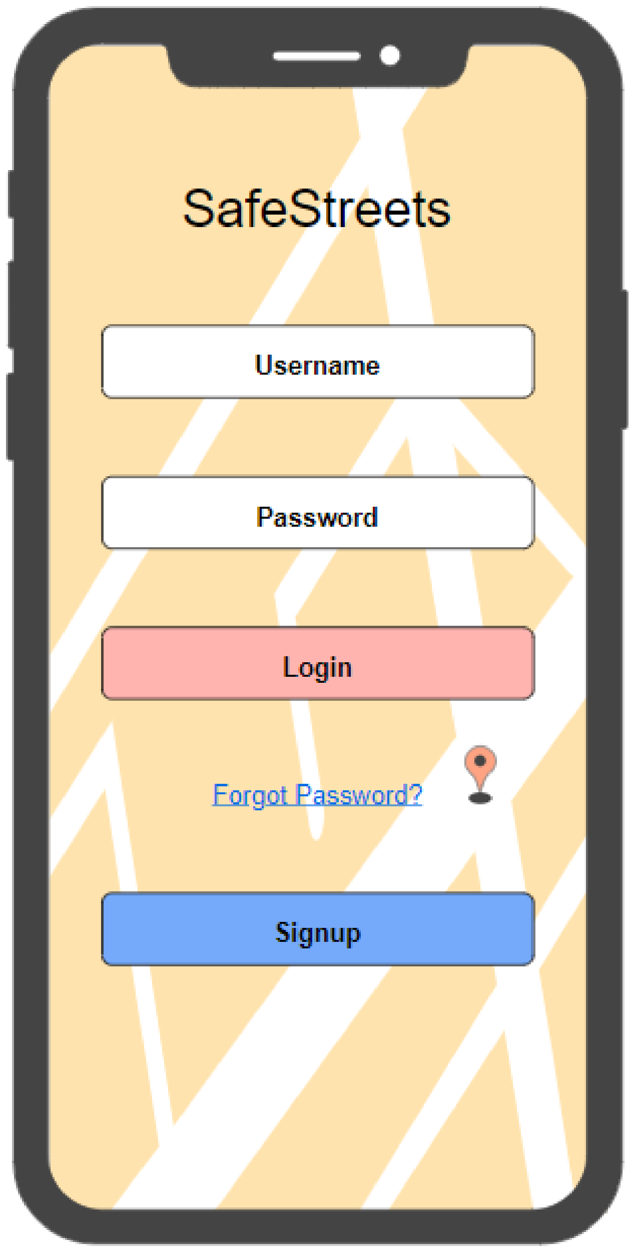

Figure 3: SafeStreets Login page.

SignUp Page:
Since logging into SafeStreets is mandatory, if the user does not have an account he needs to make one. Every field except the photo should be mandatory.

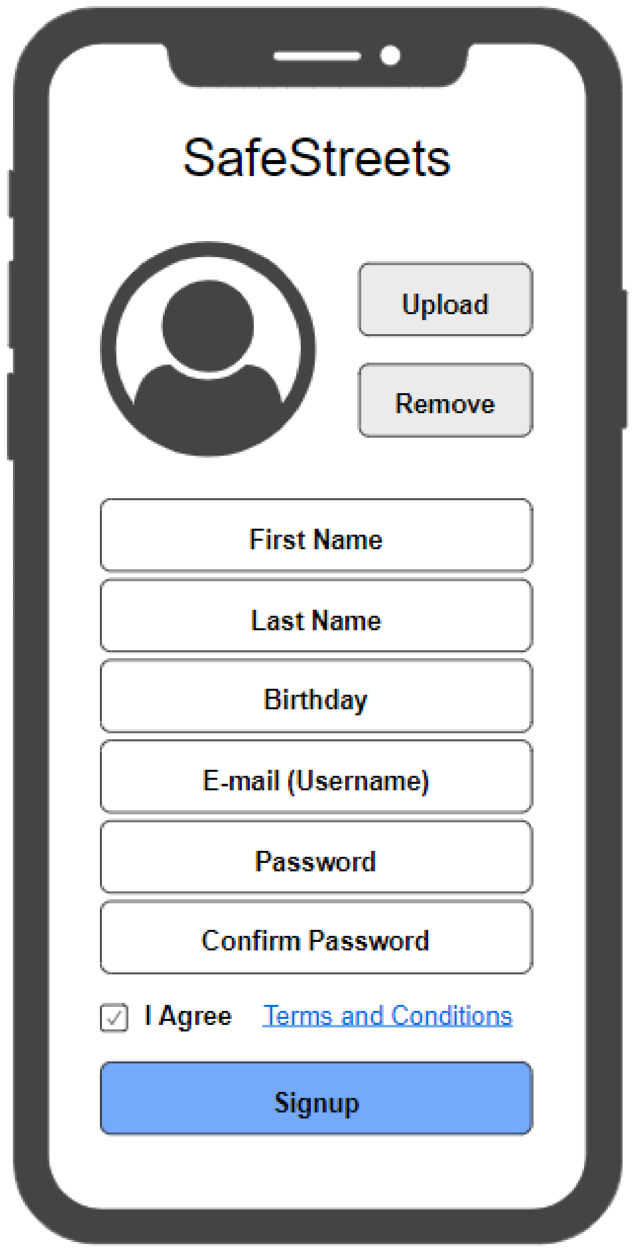

Figure 4: SafeStreets SignUp page.

Home Page:
When the user logs into SafeStreets successfully, an home page should be provided with the options for the user to use every functionality of the application.

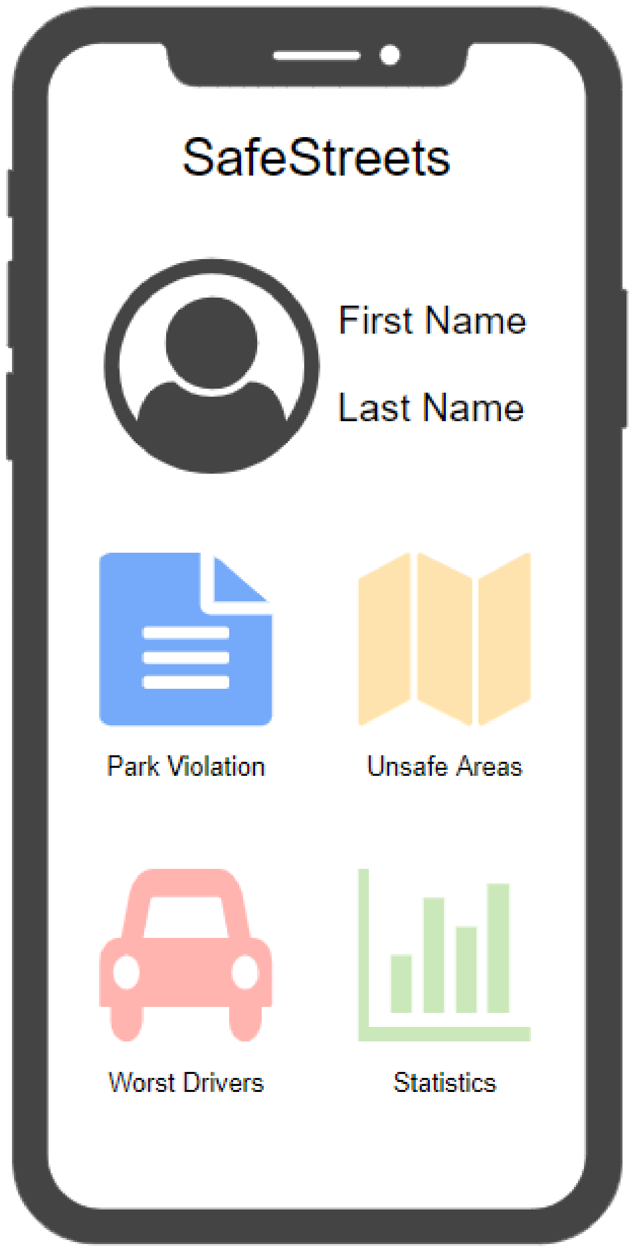

Figure 5: SafeStreets Home page.

Violation Page:
If the user wants to send to the officers a parking violation of some sorts a form to be filled is provided. Every field should be mandatory.

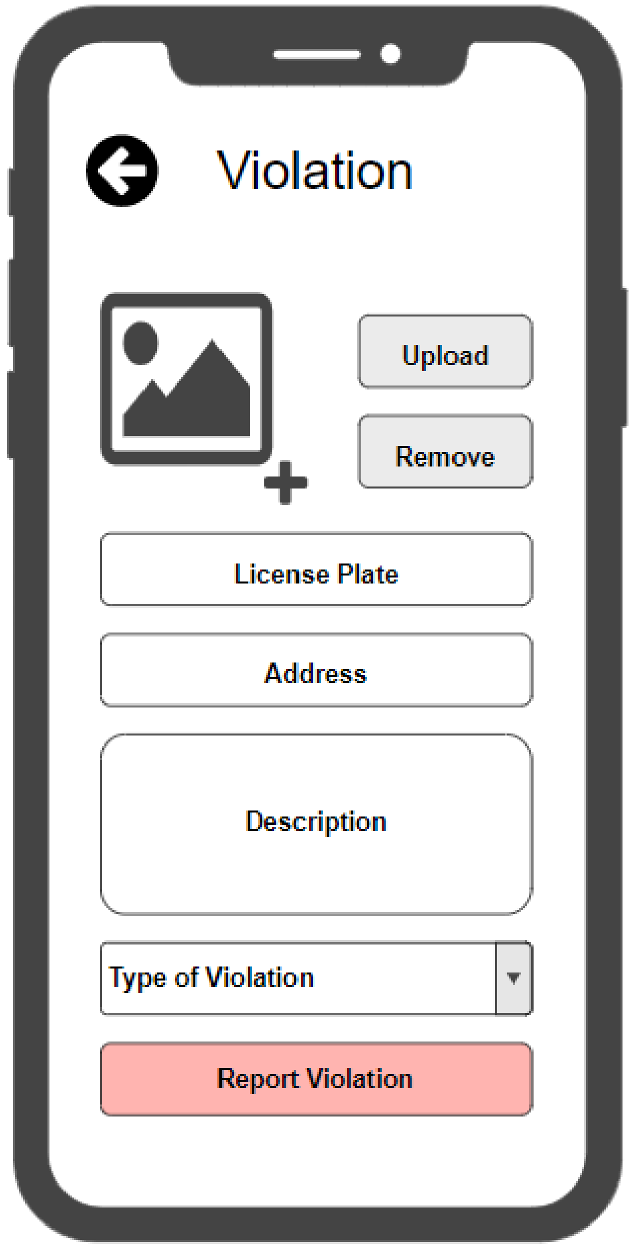

Figure 6: SafeStreets Violation page.

Unsafe Areas Page:
If the user wants to see which areas are the most subject to violations, an interface to search among all areas should be implemented.

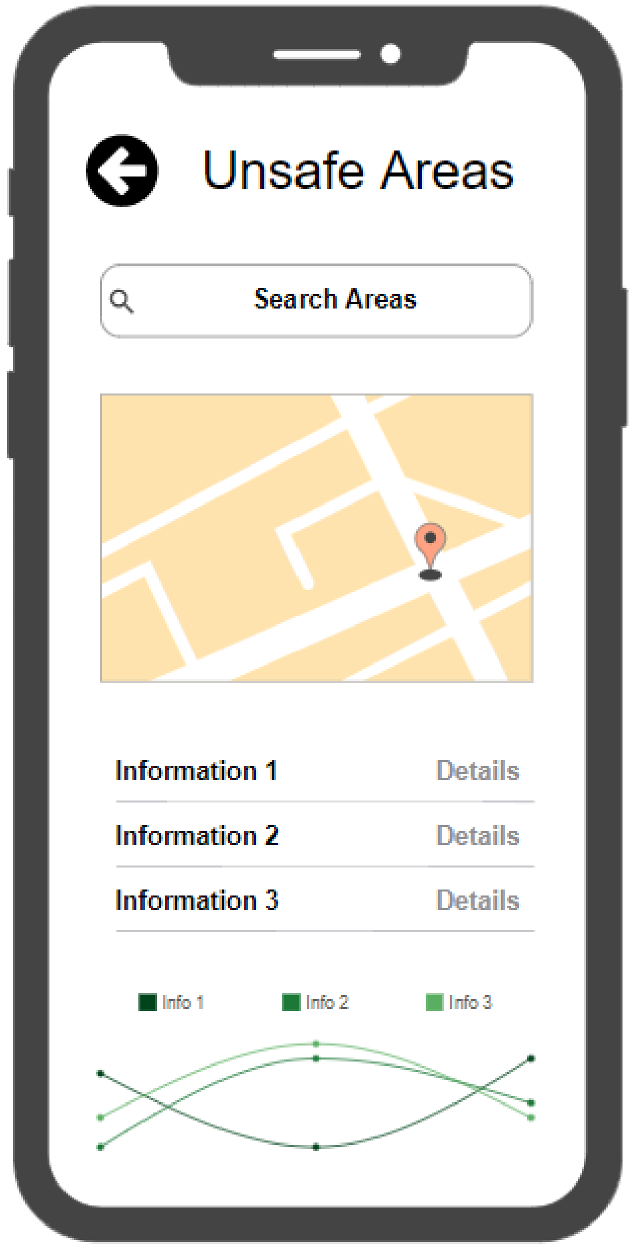

Figure 7: SafeStreets Unsafe Areas page.

Worst Drivers Page:
If the user wants to see which vehicles tends to not follow the city rules, an interface that shows this needs to be present in the application. A license plate search is better to be avoided.

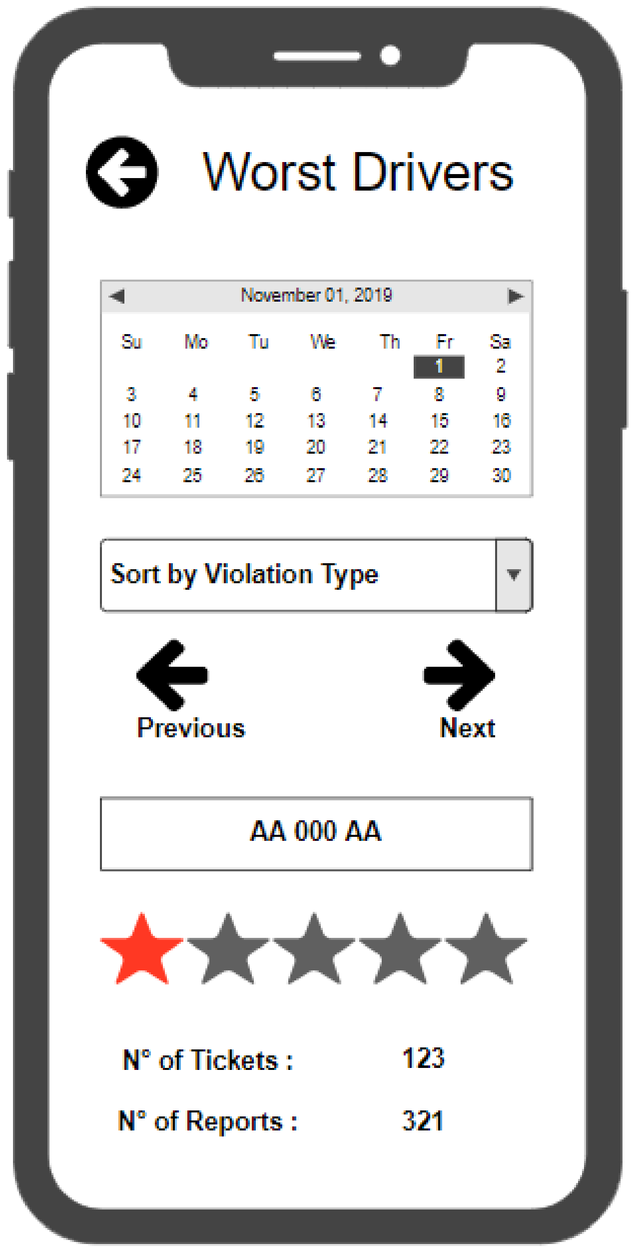

Figure 8: SafeStreets Worst Drivers page.

Statistics Page:
A complete page that exhibits all of SafeStreets data in a complete and detailed manner, that also enlightens SafeStreets effectiveness.

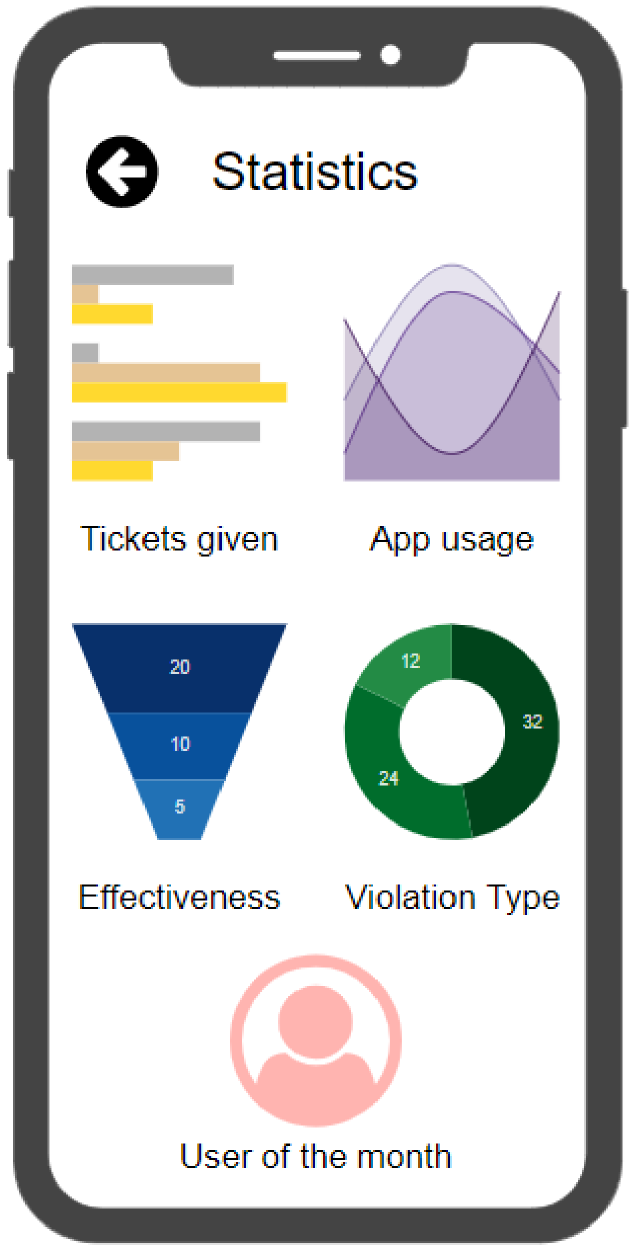

Figure 9: SafeStreets Statistics page.

User Profile Page:
If the user wants to change its profile picture or his password, if he wants to check his reports or to logout, he should be able to do all these things.

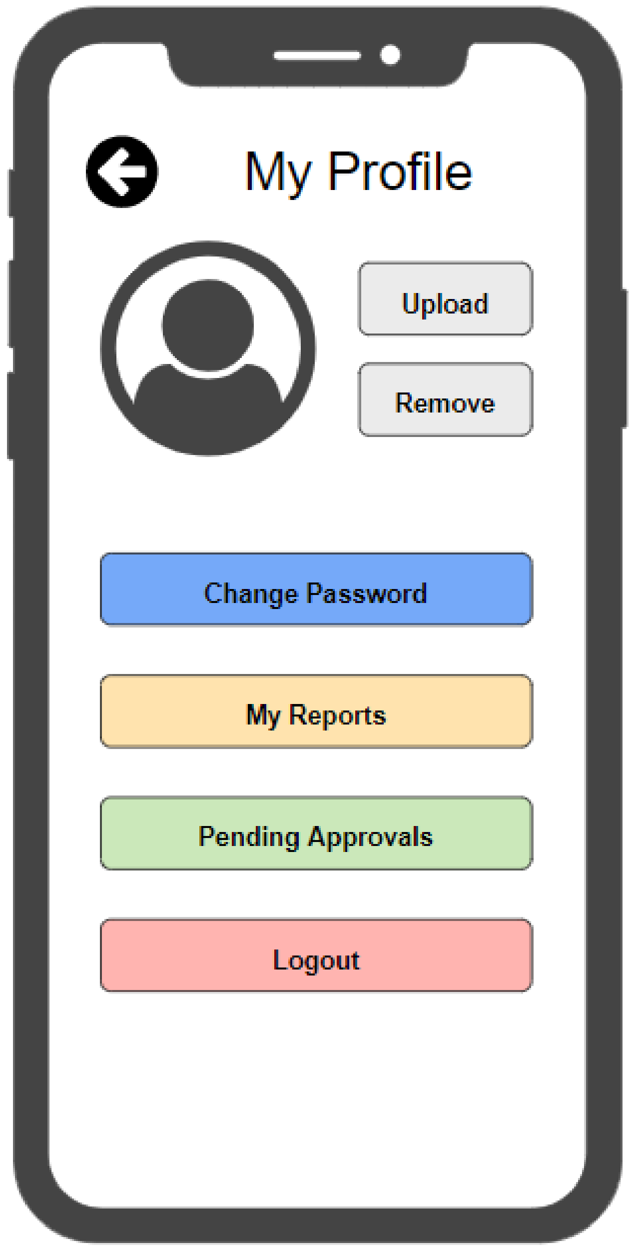

Figure 10: SafeStreets User Profile page.

3.2 Scenarios

3.2.1 First scenario

	Creation of a report
Actors	Mario Rossi and Luca Neri: habitual users Maurizio Verdi: a car owner
Flow of events	Maurizio is driving his car to reach his girlfriend house. Once arrived, he leaves his car double parked because he does not want to spend too much time looking for a parking spot. Mario is - as usual - taking his dog out before dinner and is headed through the park situated five minute by food. While walking, he notices that Maurizio's car is double parked and is slowing down all the cars running in the street. His dog is making more than an effort to go to the park and so Mario wouldn't have much time to call the Police to report this violation. Meanwhile, Luca is - as usual - getting back home by car. He lives outside the city and he thinks the train is not trustworthy so he chose the car. But getting out of the city at 7PM is always very slow because of the many dribbling the cars should do to avoid other cars parked on the lane and that said, he is late for the dinner with his wife. Mario, still in a hurry, takes out his mobile phone and opens SafeStreets app. From the homepage he creates a new report and the process did not take much time: all he has to do is shoot a photo and select the double park infraction from the menu. So, after a few seconds, he was able to continue his walk with his dog toward the park. That day, Luca ended up arriving late at home and eating alone his cold meal. As soon as Maurizio receives the ticket from the Local Police he realizes that, now that the world has SafeStreets, it's worth the time to seek a legal parking to avoid being fined again. As days go by, more car owner realize the same thing and after a while Luca is finally able to come back home with less pain after his long days of work.

Table 4: Scenario 1

3.2.2 Second scenario

	Data mining and suggestion proposal
Actors	Giovanni Ciano: head officer of the Local Police
Flow of events	As situations like double parking keeps happening in the same area as in Table 4, many car owner are getting more polite and respectful, but it is still not enough. Indeed, one day, Giovanni receives a letter from the mayor asking him to find a way to help solving the city mobility problem in that area. It is a difficult problem to handle by his own and he does not have enough data and suggestions from his officers because it seems that they cannot find a pattern. Giovanni is desperate but then he has an idea: he opens his SafeStreets control panel and sees that there are suggestions not examined yet. The software has detected that the same double parking violation keeps occurring and it therefore alerts Giovanni to proceed with the automatic detection of parking infractions. The automatic detection of such infraction is a mechanism, adopted by the Local Police, that consists in putting a special video camera on a Police car and then driving along the streets to automatically fine all car owners that are not compliant with the local laws. Giovanni schedules it for the next Friday: two officers will take care of it. After that raid, fifteen different parking violations were discovered all in the same street! Without SafeStreets, Giovanni wouldn't have ever imagined that such a disease was taking place in that street, but thanks to the suggestion he received, now car owners will certainly respect the law, avoiding slowing down the traffic in that street.

Table 5: Scenario 2

3.2.3 Third scenario

	Data mining and suggestion proposal
Actors	Matteo Neri: officer of the Local Police
Flow of events	Summer is a great season, there are less workers and students in the city and many of them use bikes or go to work by foot. Every autumn, instead, people in the city begin to leave their bike at home and head to work by car. Matteo is perfectly aware of this situation because, as usual, on October his head officer sends him and his colleagues to a few intensive sessions of heavy fining to car parked in the wrong way. Matteo thinks he could do more for the city than fining people all day, and is always bored by it. Ever since when SafeStreets has been introduced, citizens and normal users have become impressive reporter of violations and this behaviour keeps growing. Having realized this, now Matteo and his colleagues can be assigned to more meaningful tasks, for example they could handle the traffic near the schools when it starts or ends. Matteo is now more happy of working and he thinks he's contributing to his city in a better way, improving the kids and their parents quality of life.

Table 6: Scenario 3

3.2.4 Fourth scenario

	Analytics consulting by users
Actors	Tobia Viola: a 18 years old SafeStreets user
Flow of events	Mobile phones are now in everyday life for many of us and Tobia is a great example. Tobia does not like to watch TV after dinner and when he is alone at home he normally spends his time on Instagram, Facebook and the various applications he has installed on his iPhone. Sometimes, he even likes to open SafeStreets to consult the statistics of parking violations in his city. He wouldn't ever imagined that so many violations are occurring near his area so he decides to become an active reporter in the hope of lower that number as days go by. Consulting the analytics, he discovered that when SafeStreets was launched there were an average of 5 reported violations each day in his area. Four months has passed from that moment and now the statistics he's seeing highlights that there are only an average of 0.9 violations each day! Tobia thinks this is due to SafeStreets users and he is so proud of his work, because he reported some different vehicles in the last weeks. Tobia is seeing analytics but then he stops for a while: he sees his avatar in the "user of the month" section. He has reported 40 parking violations just this month! Tobia can't be more excited of this and he realizes that he would like to become a police officer in the future. He is now finishing college and then he'll probably attend the Police Academy.

Table 7: Scenario 4

3.2.5 Fifth scenario

	Analytics consulting by users
Actors	Ernesto Paradiso: an active runner and a SafeStreets user
Flow of events	Life in the city, as we all know, is frenetic. Precisely for this reason, Ernesto every day after work takes his bicycle and rides it around the city's streets for 8 km. Ernesto, however, lately feels unsafe because of those who park on the bike lane, forcing him to dodge all the cars and sometimes to do so he must leave the bike lane and is forced to ride the bike on street. One day after returning home, he tells his wife how the many cars parked irregularly today forced him to stay on the road for much longer than usual and how dangerous it was. His wife, worried for him, advises Ernesto to consult the SafeStreets statistics to find a safer route. Ernesto, consulting the statistics, discovers that the road he travels with his bike is classified as "unsafe area" and looking at the analytics he discovers that he could ride the same amount of km simply by going along the parallel road, on which not many violations are reported. From the following day Ernesto changed his route and felt more secure with his new path around the city (safe) streets.

Table 8: Scenario 5

3.3 Functional Requirements

This section contains all the use cases initially described with the use case UML models, then the most important Use Cases have their own table which provides further details such as: involved actors, entry conditions, flow of events, exit conditions and exceptional conditions.

3.3.1 User Page use cases

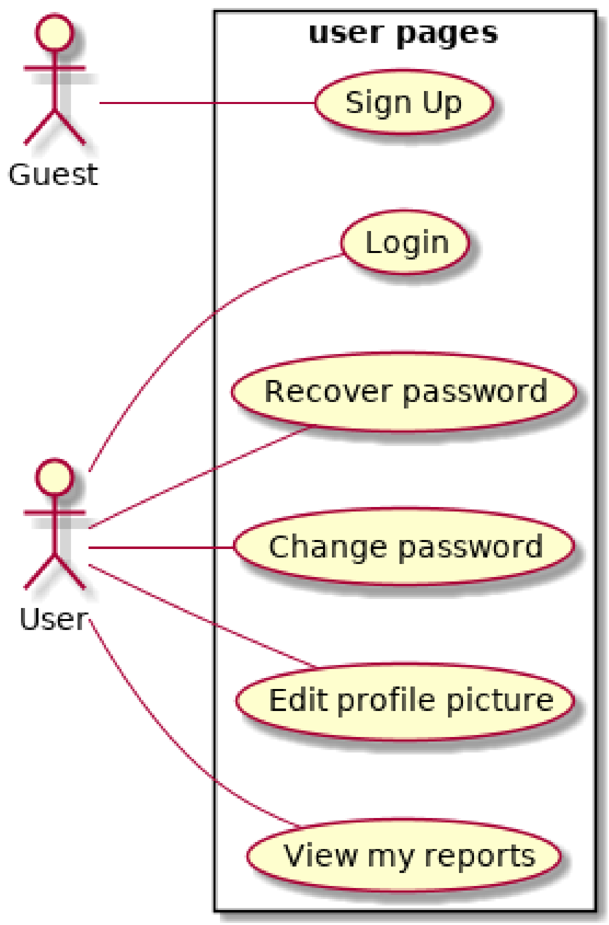

Figure 11: Use cases relative to the user registration and authentication

3.3.2 Sign up

Name	**Sign up**
Actors	Guest
Entry conditions	None
Flow of events	• The guest reaches the registration page containing the relative form • The guest fills up the form and clicks on "Sign up" to complete the process • The system redirects the user to his profile page.
Exit conditions	The guest has successfully registered into SafeStreets.
Exceptions	The guest left an empty field or typed something invalid: an error message is displayed and the user is asked to fill the form again.

Table 9: Use Case table - Sign up

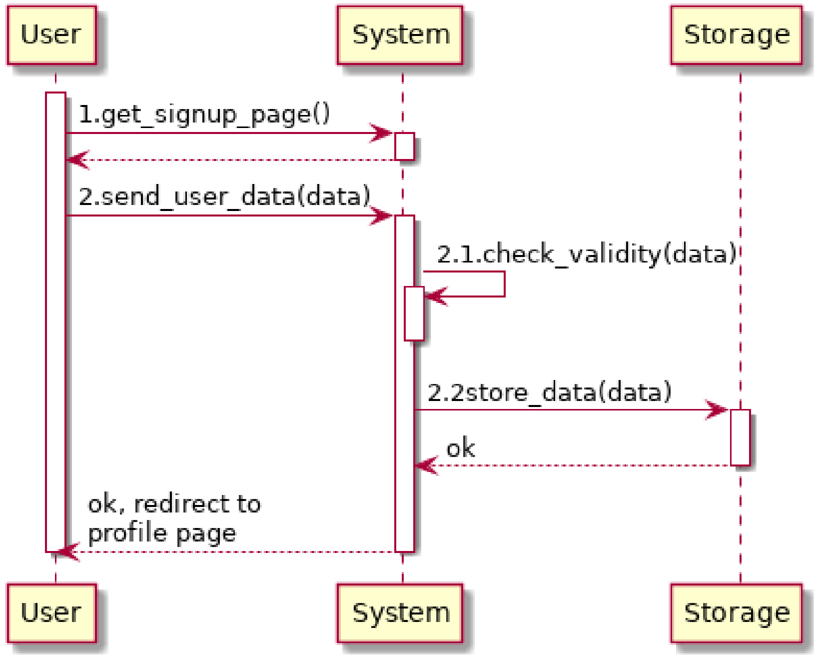

Figure 12: Use case diagram of the signup process

3.3.3 Login

Name	**Login**
Actors	User
Entry conditions	The user had already registered in the past.
Flow of events	• The user reaches the login page containing the relative form • The user types the username and password in the login form and clicks on the "Login" button. • The system redirects the user to the application homepage.
Exit conditions	The user has access to the application functionalities.
Exceptions	Username and password didn't correspond or the username didn't exist: an error message is displayed and the user is asked to fill the login form again.

Table 10: Login Use Case table

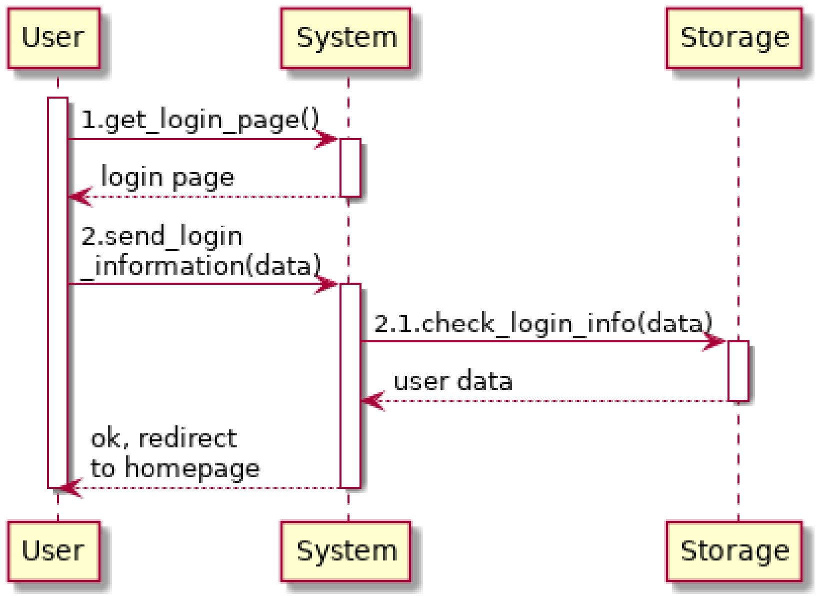

Figure 13: Use case diagram of the login process

3.3.4 Password Recovery

Name	**Recover Password**
Actors	User
Entry conditions	The user had already registered in the past.
Flow of events	• The user reaches the login page containing the relative form • The user does not remember his password so he clicks on "Password recovery" button and is redirected to the password recovery page. • The user inserts his email and clicks on "reset password". • The system sends an email to the user with a link and instruction to reset the password. • The user chooses and types a new password and confirms. • The system redirects the user to the login page.
Exit conditions	The user requested an email to reset his password
Exceptions	The inserted email does not match any user in the database, it is displayed an error message and the user is asked to retype a valid email.

Table 11: Recover password Use Case table

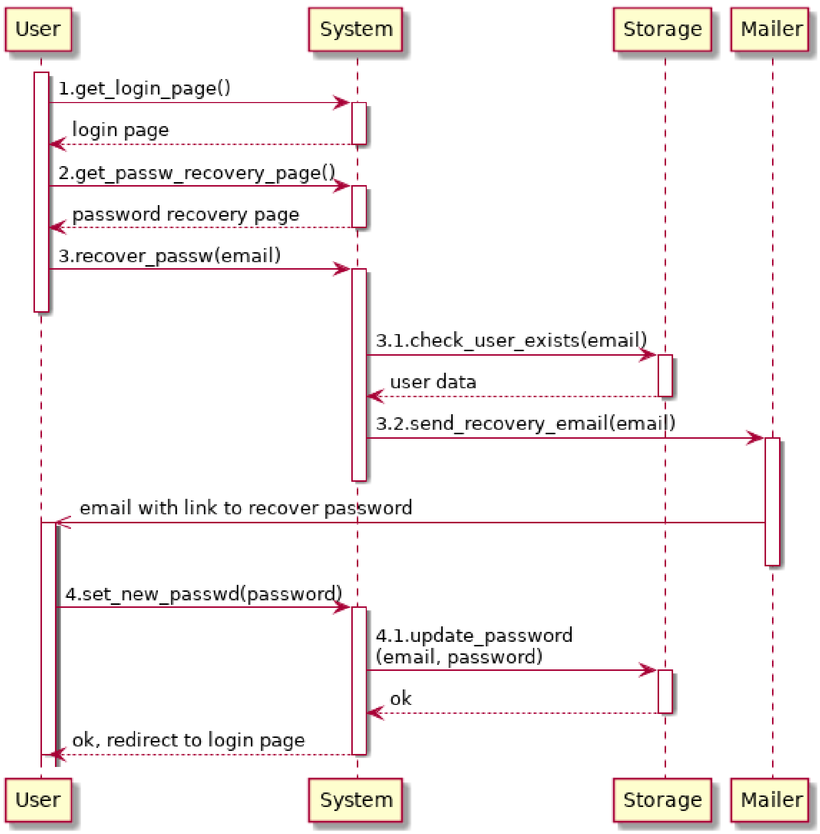

Figure 14: Use case diagram of the password recovery process

3.3.5 Report and Information Management use cases

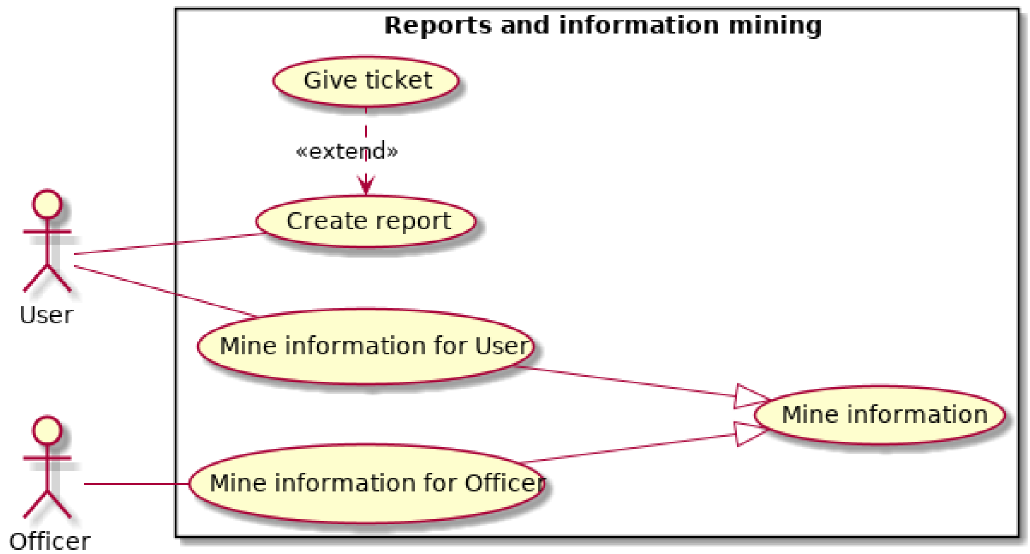

Figure 15: Main use cases showing the functionalities of SafeStreets application relative to the user report management and information mining.

3.3.6 Report Creation

Name	**Creation of a new report**
Actors	User
Entry conditions	The user is logged in and is in the main page.
Flow of events	• The user clicks on "Report Violation" button and is redirected to the page with the input form to create a new report. • The user fills up the form with the violation information (type of violation, pictures of it, address, description, license plate position, date and time, etc...). Eventually some fields such as date and time will be auto-completed. • The user clicks on the "Send Report" button after a quick check of all the field he typed. • The system shows a confirmation message to the user. • The user is redirected to the main page.
Exit conditions	The new report with the user-inserted data is created into the SafeStreets system. In the case of the violation resulting in a duplicate (possibly by another user) of in case of the license plate to be not readable, the system put this report in a revision queue and block the process.
Exceptions	The information inserted is wrong (non-existent address, date and time in the future) or some information is missing: a corresponding error is displayed and the user is asked to modify the inserted information accordingly.

Table 12: Report Creation Use Case table

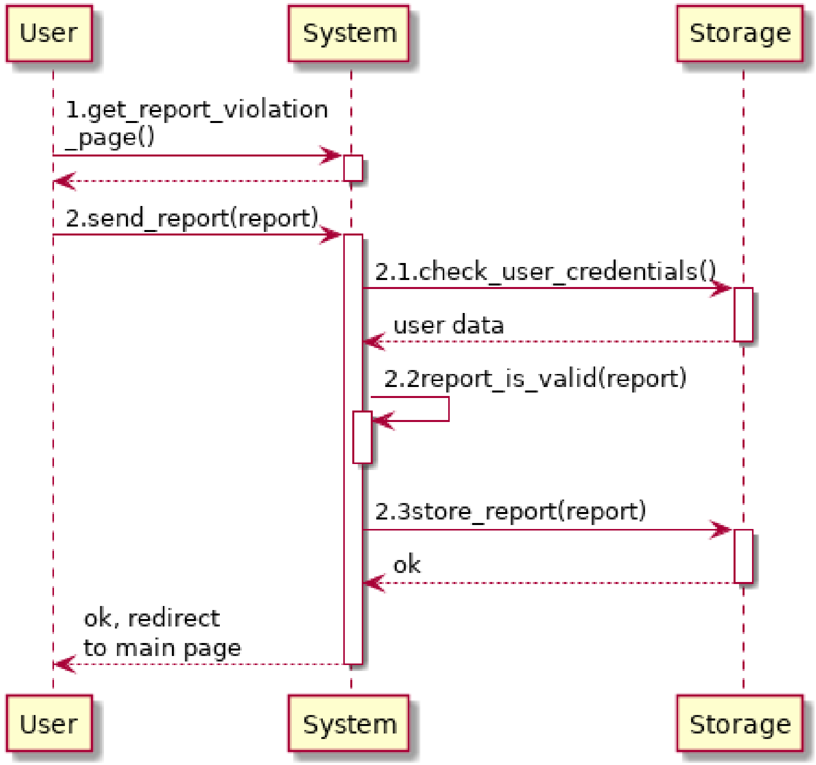

Figure 16: Use case diagram of the report creation process

3.3.7 Information Mining by Users

Name	**Information Mining by users**
Actors	User
Entry conditions	The user is logged in and is in the main page.
Flow of events	• The user clicks on the "Show statistics" button and is redirected to the page with the general SafeStreets statistics such as number of usages, cities availability, etc. • The user clicks on the selection menu to choose which statistics he wants to see, possibly entering more data about which area he wants to focus on (most dangerous areas, vehicles with highest frequency of violations, etc). • The user could analyze the data that will be shown and after he finishes, he presses the Home button. • The user is then redirected to the main page.
Exit conditions	The user-requested analytics is displayed to the user at the level of aggregation he has chosen (that kind of information is not sensitive).
Exceptions	The selected area is actually not present on SafeStreets or nonexistent: an error is displayed and he is asked to modify the inserted location accordingly.

Table 13: Data Mining Use Case table

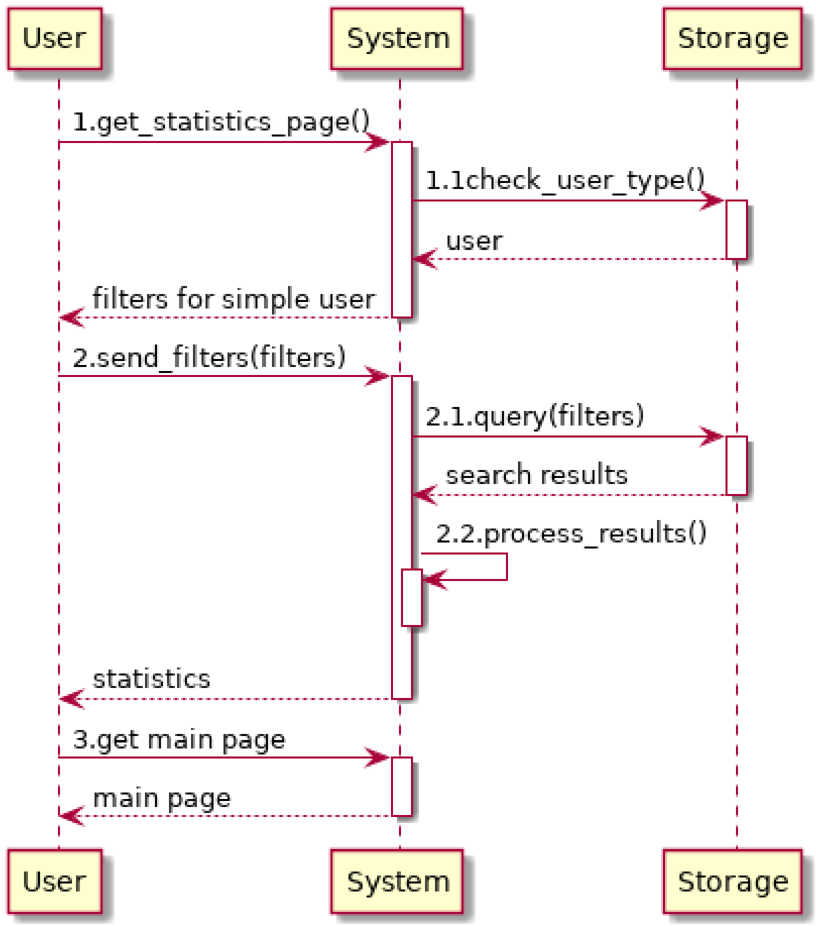

Figure 17: Use case diagram of the information mining process for a user

3.3.8 Information Mining by Officers

Name	**Information Mining by officers**
Actors	Officer
Entry conditions	The officer is logged in and is in the main page of his software.
Flow of events	• The officer clicks on the "Statistics" section button and is redirected to a page with the general SafeStreets statistics (such as number of usages, last use date, etc) in the officer municipality. • The officer clicks on the selection menu to switch between statistics related to his municipality, possibly exporting it in CSV (most dangerous areas, vehicles with highest frequency of violations, etc). • The officer could analyze the data that will be shown and after he finishes, he presses the Home button. • The officer is then redirected to the main page.
Exit conditions	The officer-requested analytics is displayed to the officer for the municipality land.
Exceptions	None.

Table 14: Data Mining for Officers Use Case table

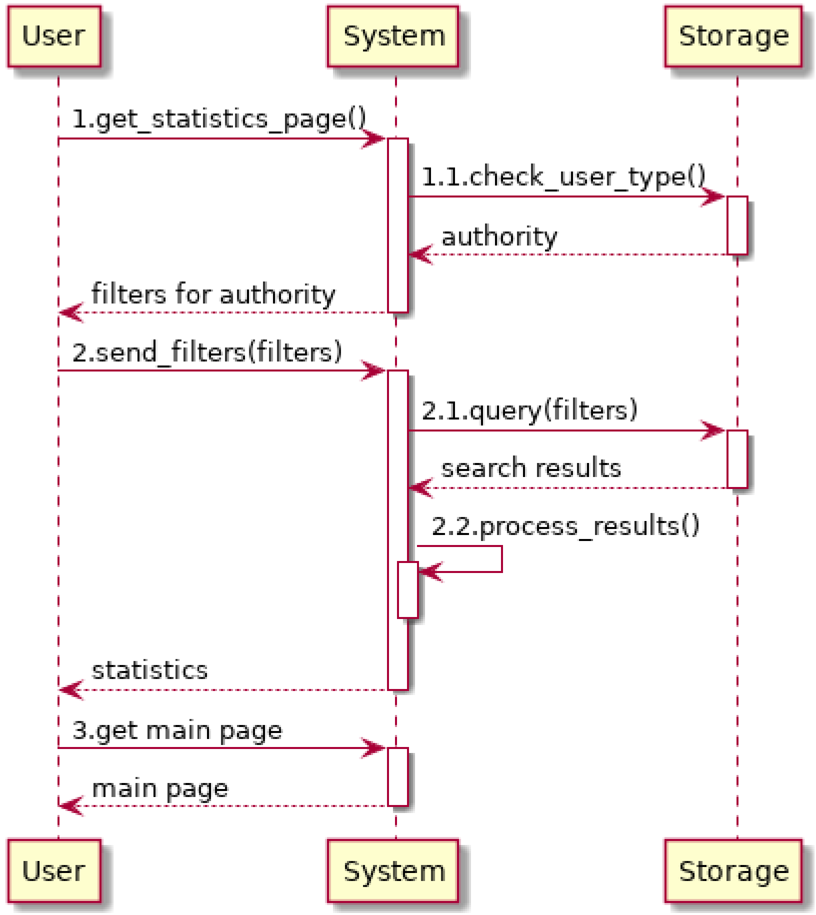

Figure 18: Use case diagram of the information mining process for an officer

3.4 Non functional requirements

3.4.1 Performance

The application should be able to deliver the violations (including the automatic tickets) in an acceptable time, which means at most 15 seconds with a 4G connection, but should average to about 5 seconds. This is not a strict requirement, but it improves a lot the user experience and the possibility that he does not decide to give up with the upload.

The data mining functionality is not as trivial as the delivery of the violations, but keeping in mind the effectiveness of the user experience it should still be able to deliver results with an average waiting time of 5-10 seconds, with an upper bound of 20 seconds.

The request for interventions functionality does not have any time constraints, it should just end. It can be run at any time and its results will be stored on persistent storage, without the need to be used immediately. This functionality does however consume a lot of computer resources, since it must cross a large quantity of data from different sources and it will also use an AI. For this reason, it will have to run either when the load on the system is low or on a completely separate hardware.

All the other operations that require an internet connection (login, logout...) should be fast enough to not become frustrating for the user to use the application, indicatively they should take 5 seconds at most.

3.4.2 Reliability

The system has no reasons to be highly available except for providing a great user experience. So, since this is not a critical application, short period of down could be acceptable.

3.4.3 Availability

Like said before SafeStreet does not need to be up to ensure critical aspects, but since some violations could have only a short period of time to be checked, it will have to guarantee a 99.99% availability.

3.4.4 Security

Since users will be able to give tickets automatically, security is a key aspect of the application. The information contained in the report of a violation must never be altered, and the login information must be kept safe to prevent hackers from giving false tickets using stolen identities.
Users also give their personal data when registering, and their privacy must be guaranteed.

3.4.5 Scalability

This application will be a pioneer in this field, so it is difficult to foresee how many users will use it and how. A high degree of scalability is therefore required. To handle the first city, Milan, the system must handle 1 million registered users and a peak of 10.000 concurrent requests.

3.4.6 Maintainability

Since the application requires a high degree of scalability, a code easy to fix, modify or update is preferable, because actions on it will probably take place.

3.4.7 Portability

The application has to be compatible with the majority of devices present on the market in the last years, in order to increase the people eligible to become users of SafeStreets.

3.4.8 Accuracy

The maximum error that the GPS system should commit for the application to accept a position is 15 meters, to let an eventual officer near the violation find the car.

4 Formal Analysis using Alloy

This section employs the Alloy formal notation to provide a description of the domain, the properties and the constraint of the model and to prove that they can be satisfied by the system.

4.1 Code

```
/******DOMAIN ASSUMPTIONS******/
sig Email {} {one u: User | this = u.email}
sig Password {} {some u: User | this = u.password}

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

sig User extends Customer {
    reports: set Violation
}
sig Officer extends Customer{}

//∀ licensePlates must belong to a picture
sig LicensePlate {} { some p: Picture | this = p.
   licensePlate }
// ∀ tickets, pictures and positions must belong
   to a violation
sig Ticket {} { one v: Violation | this = v.ticket
    }
sig Position {} { some v: Violation | this = v.
   position }
sig Picture {
    licensePlate: lone LicensePlate
} { one v: Violation | this in v.pictures }

//a violation type can belong to no violation
```

```
sig ViolationType{}

sig Violation {
    violationType: one ViolationType,
    position: one Position,
    ticket: lone Ticket,
    pictures: set Picture
} {
    //a violation must belong to only one user
    one u: User | this in u.reports
}

sig Accident {
    position: one Position,
    accidentType: one AccidentType
}
sig AccidentType {} {
    //an accidentType must belong to an accident
    some a: Accident | this in a.accidentType
}

//2 users cannot have same email
fact {
    no disj c1, c2: Customer | c1.email = c2.email
}

//a violation must have exactly one license plate
fact {
    //at least one
    ∀ v: Violation | some p:Picture | p in v.
       pictures and #(p.licensePlate)=1
    //at most one
    ∀ v: Violation | no disj p1, p2: Picture |
         (p1 in v.pictures) and (p2 in v.pictures)
           and (p1.licensePlate ≠ p2.licensePlate)
}

/****************ASSERTIONS****************/
```

```

//a report cannot belong to 2 users
check {
    no disj u1, u2: User | some v: Violation | v
       in u1.reports and v in u2.reports
} for 5

pred show{
    #User = 2
    ∀ u: User | #u.reports > 0
    #Violation > 0
    ∀ v: Violation | #(v.pictures) > 1
    ∀ p: Picture | #p.licensePlate = 1
}
run show for 5
```

5 Softwares Used and Effort Spent

5.1 Softwares Used

Task	Software Used
Edit and compile LATEX code	TeXstudio and VisualStudioCode
UML modelling	PlantUML
Run and compile Alloy	Alloy 4.2
Mockup	Moqups

Table 15: Softwares used

5.2 Effort Spent

Student	Hours spent
Andrea Furlan	-
Cosimo Russo	-
Giorgio Ughini	-

Table 16: Effort spent