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

December 9, 2019 Version 1.2

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.
- Thanks to the Computer Vision, recognize useful details such as car model or color, on images of violations submitted by users.
- 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 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**] Allowing users to notify officers when particular parking violations occur.
- [**G2**] Permitting both users and officers to learn which are the most unsafe areas in their jurisdiction.
- [**G3**] Suggesting possible interventions to potentially unsafe areas in the city.
- [**G4**] Allowing the authorities to generate automatic parking violation tickets from users' reports.
- [**G5**] Permitting both users and officers to learn which vehicles commit the most violations.
- [**G6**] Permitting customers to apprehend the possible effectiveness of these mechanisms.
- [**G7**] Allowing both users and officers to learn the analytics of the tickets given in their city.

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, C3, C4, C6	U1
G2	F2	C8, C3, C9	U6, U7
G3	F3	C3, C7, C8	U7
G4	F4	C2, C4, C6	U5
G5	F2, F5	C4, C6, C5	U9, U8
G6	F2, F5	C6, C8,C9	U10
G7	F2, F5	C2,C8	U10
-	F0	C11	U1, U2, U3

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.

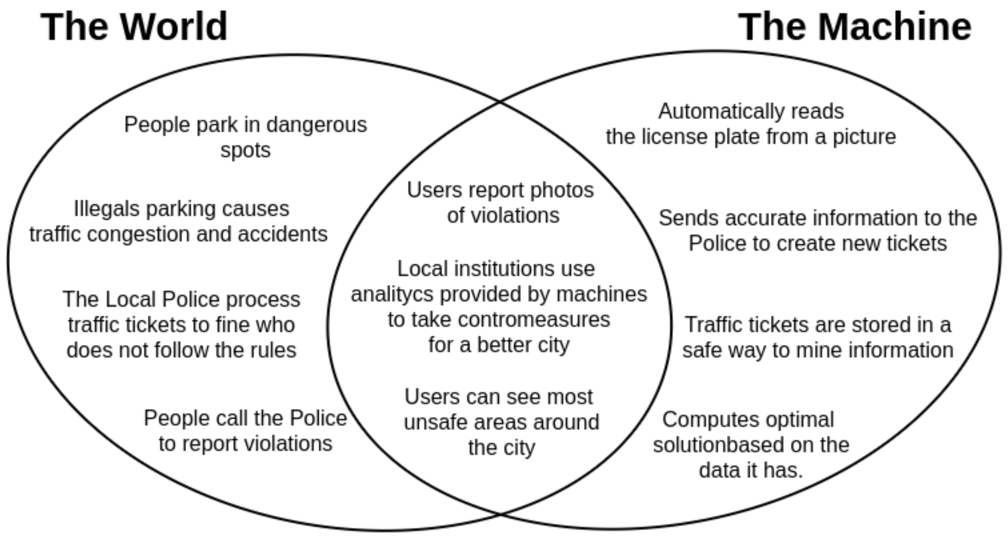

The World
The Machine
People park in dangerous spots
Illegals parking causes traffic congestion and accidents
The Local Police process traffic tickets to fine who does not follow the rules
People call the Police to report violations
Users report photos of violations
Local institutions use analitycs provided by machines to take contromeasures for a better city
Users can see most unsafe areas around the city
Automatically reads the license plate from a picture
Sends accurate information to the Police to create new tickets
Traffic tickets are stored in a safe way to mine information
Computes optimal solutionbased on the data it has.

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.
- *Computer Vision*: The Computer Vision includes methods for acquiring, processing, analyzing and understanding digital images, to extract high-dimensional data from the real world.
- *Report*: The data that a user has provided to the authorities that witness a violation by a vehicles.
- *Ticket*: Notice issued by a law enforcement official to a motorist or other road user, indicating that the user has violated traffic laws.
- *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.
- *Officer* or *Authority*: This actor represents a public officer that interacts with SafeStreets in some ways.
- *Customer*: either a Guest or an Authority

1.3.2 Acronyms

- *AI*: Artificial Intelligence
- *API*: Application Programming Interface
- *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 available to as many people as possible and to be able to run on 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 a system that will access SafeStreets data through its 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.

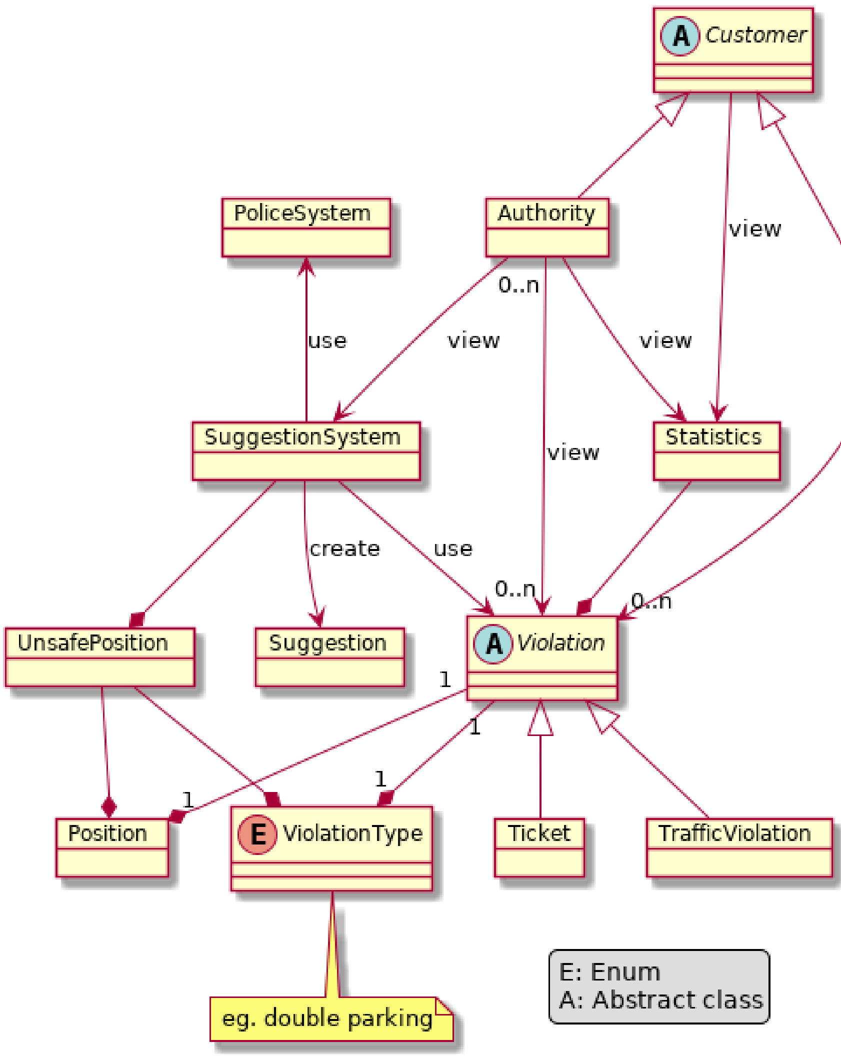
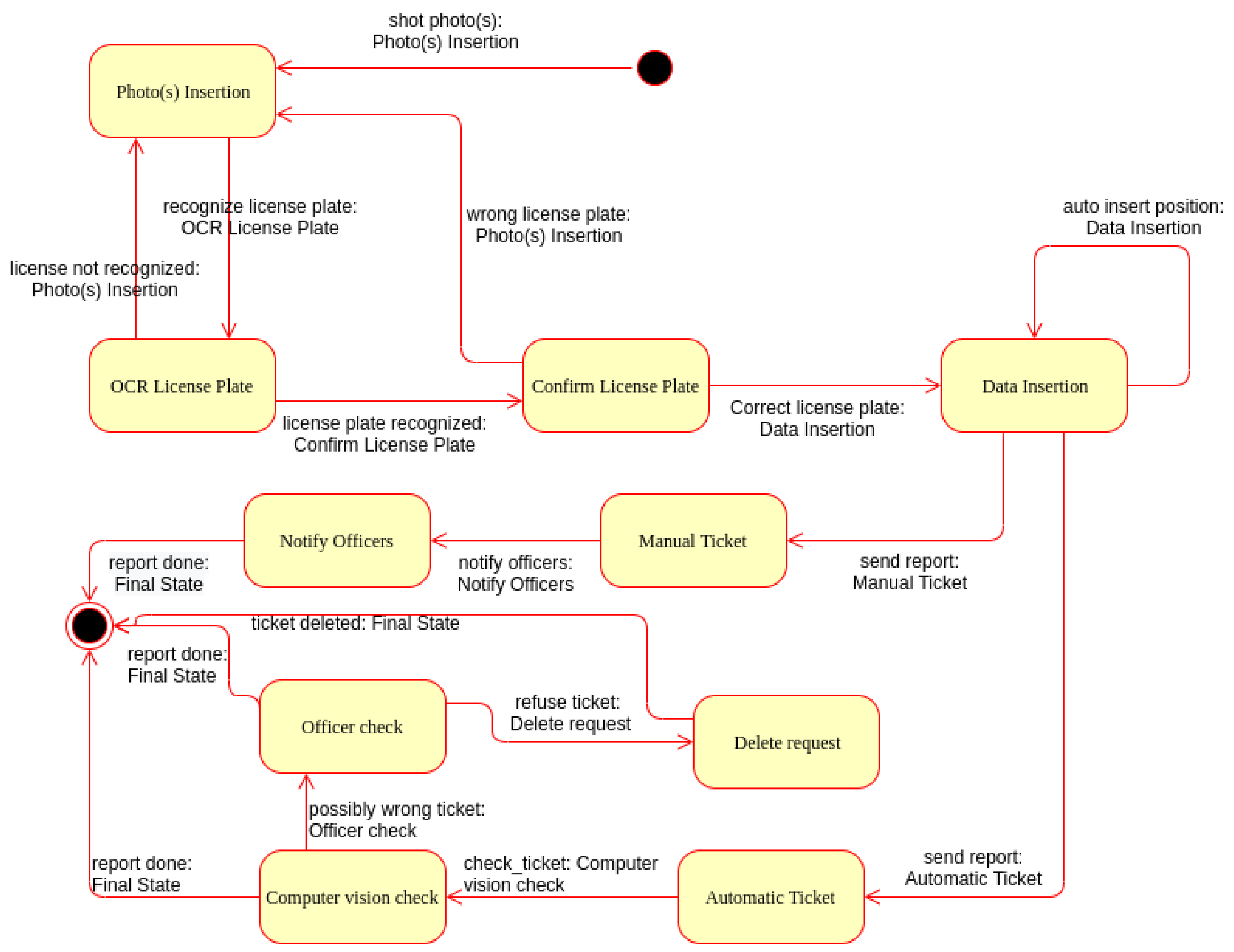


Figure 2: The class diagram of SafeStreets.

2.3 State Diagrams

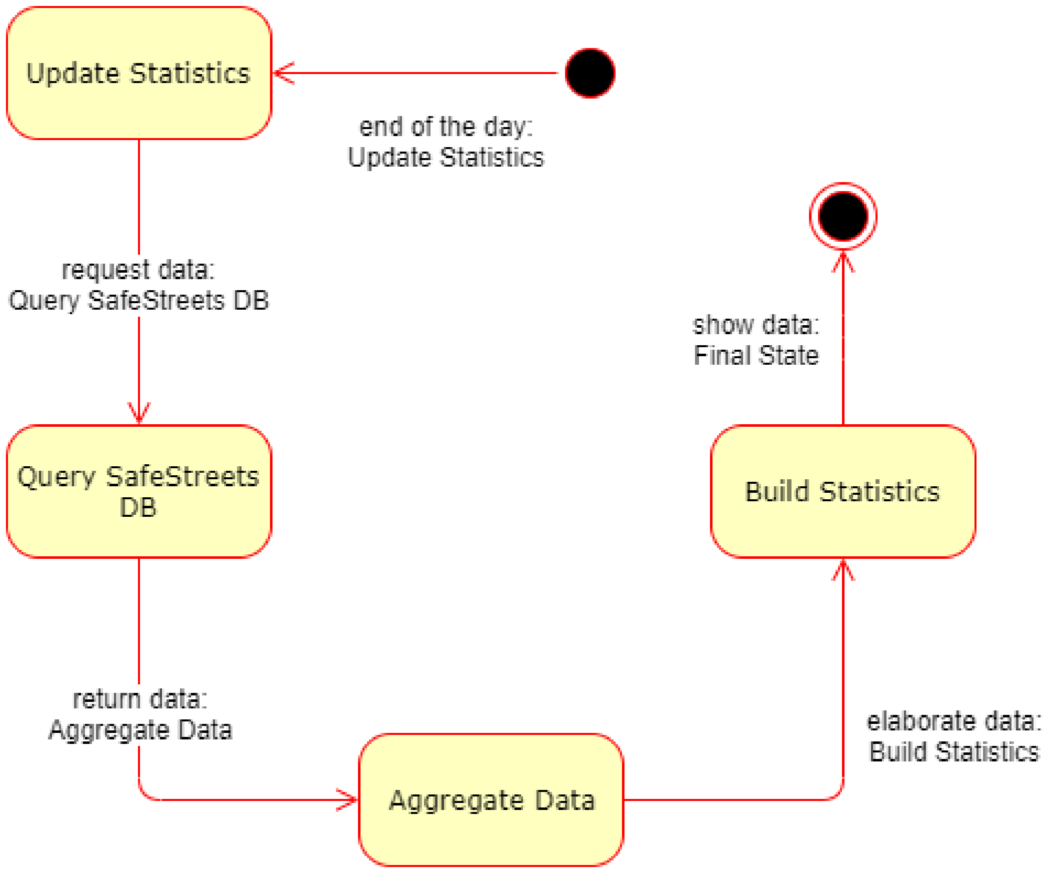
2.3.1 Creation of a Report

Whenever the user wants to send a report of a parking violations to the authorities he has to follow what is visually described in the diagram below. The user needs to insert a set of photos where a license plate is readable, and after confirming that the app has successfully read the license plate of the vehicles in the photos he should complete the form with the information required. At last he has to select if he wants to send an automatic or a manual ticket to the officers, after that the process is completed. If the automatic ticket option is selected, the AI and Computer Vision should identify a low degree of confidence.


shot photo(s): Photo(s) Insertion
Photo(s) Insertion
recognize license plate: OCR License Plate
wrong license plate: Photo(s) Insertion
auto insert position: Data Insertion
license not recognized: Photo(s) Insertion
OCR License Plate
Confirm License Plate
Data Insertion
license plate recognized: Confirm License Plate
Correct license plate: Data Insertion
Notify Officers
Manual Ticket
report done: Final State
notify officers: Notify Officers
send report: Manual Ticket
ticket deleted: Final State
report done: Final State
refuse ticket: Delete request
Officer check
Delete request
possibly wrong ticket: Officer check
report done: Final State
check_ticket: Computer vision check
send report: Automatic Ticket
Computer vision check
Automatic Ticket

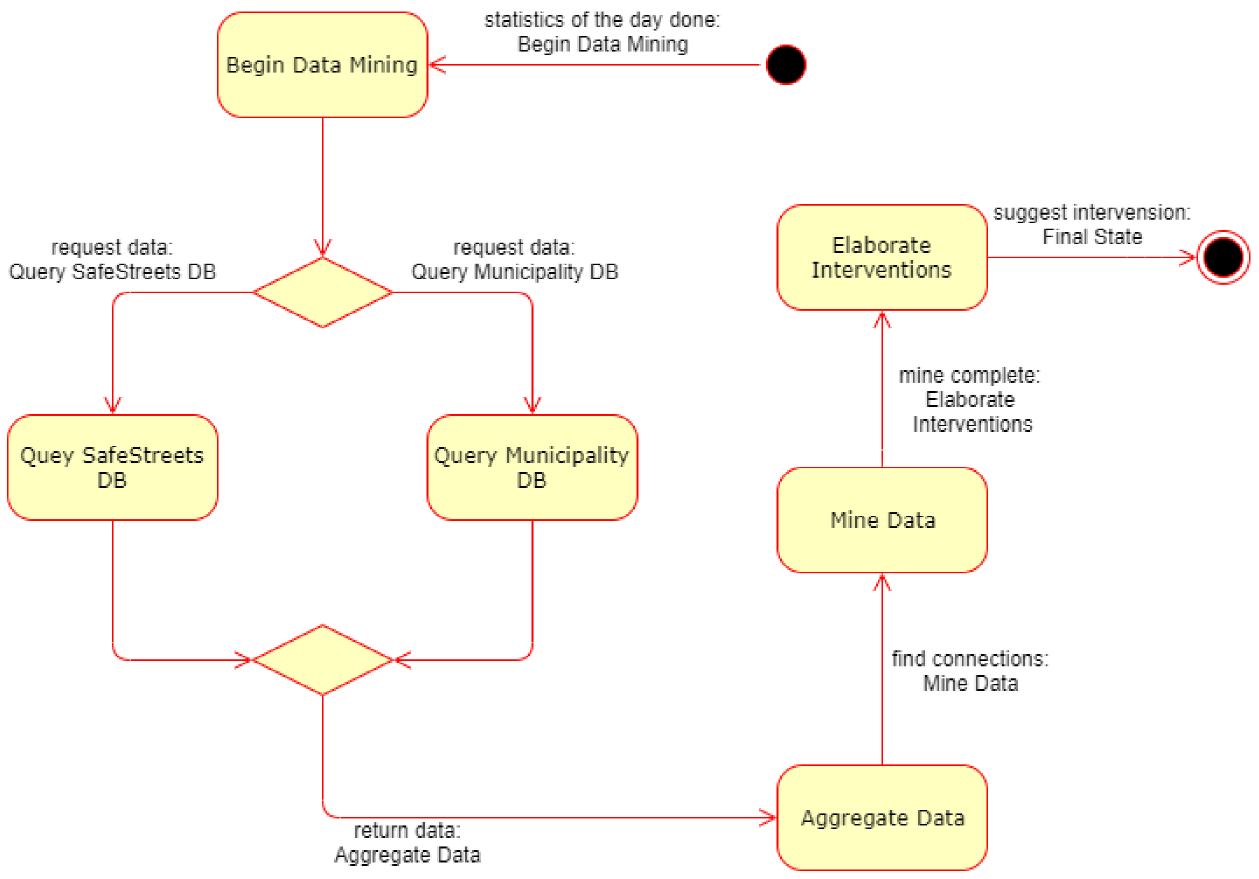
2.3.2 Statistics Construction

At the end of the day the statistics need to be updated in order to offer to the users and to the officers only recent, and therefore consistent, information. When the servers will probably have the lowest load, so in the middle of the night, SafeStreets will at first retrieve the new data that are now present in the SafeStreets database. Once the complex queries have returned the new information the data needs to be aggregate and elaborate in order to exhibit the updated statistics to the costumers of SafeStreets.



2.3.3 Data mining for Interventions

At the end of the day as soon as the statistics are completed, the process of suggesting possible interventions needs to be executed. To do so, mining data using both SafeStreets data and municipality information on the accidents occurred in the city is necessary. The flow is similar to the one for the statistics, since both needs to retrieve data, analyze it and produce some result with it. After the process of computing information to suggest interventions in the city the authorities must be warned in order to actually intervene.



2.4 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.

2.4.1 [F0] User data management

SafeStreets users must sign up the first time they intend to access any application functionality and further usages of the app will require a login to access all its functionalities. Of course, there will be the possibility to recover the password if a user does not remember it.

2.4.2 [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 licence plate will be automatically read by the software and prompted to the user.

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 description 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.

2.4.3 [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.

In particular, a user will be able to get information about:

- The app usage: how many users are using the app and when
- The effectiveness of the initiative by looking at trends
- The most popular traffic violations in a city by category
- The most active user of the month: the one who reported most violations

2.4.4 [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.

2.4.5 [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.

The tickets are first checked by a computer vision mechanism that takes several parameters (position, pictures, similar violations) to spot wrong violation and forward them to an officer for a double check. The officer can then accept the ticket or delete it.

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 stored on a server which will not allow any modifications to it, and will be used by the police systems to confirm the tickets.

2.4.6 [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 along with his license plate and his car model and color, and the change in the number of tickets in a certain place during time.

2.4.7 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, C3, C4, C6	U1
G2	F2	C8, C3, C9	U6, U7
G3	F3	C3, C7, C8	U7
G4	F4	C2, C4, C6	U5
G5	F2, F5	C4, C6, C5	U9, U8
G6	F2, F5	C6, C8,C9	U10
G7	F2, F5	C2,C8	U10
-	F0	C11	U1, U2, U3

Table 2: Traceability Matrix

2.5 User characteristics

The users of SafeStreets can be of any age and gender 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 of any gender but they need to be actual public officers enrolled in the Local Police. They should have a medium knowledge of SafeStreets software 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.

2.6 Assumptions and dependencies

2.6.1 Domain Assumptions

- Most users will input correct data when reporting a violation
- The image and picture meta-data aren't altered by the user who first submits the report
- The municipality will provide correct and complete information about the accidents
- The municipality services will not be offline for more than 5 minutes in a row during the uptime of SafeStreet
- The police system and the interfaces it provides guarantee an acceptable level of security

2.6.2 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. a 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 are 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.

2.6.3 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]** Giving multiple tickets to the same car for a single violation is prevented
- **[C7]** A suggestion for a possible intervention is destined to the authorities only.
- **[C8]** Statistics are generated using all and only the succesfully submitted reports.
- **[C9]** The issue of reports which location is outside the competence area of the local police is prevented
- **[C10]** The communication channel between the SafeStreets and the police system must be secure
- **[C11]** Two users cannot register with the same email
- **[C12]** The chain of custody of the information coming from the users cannot be broken

2.6.4 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, C3, C4, C6	U1
G2	F2	C8, C3, C9	U6, U7
G3	F3	C3, C7, C8	U7
G4	F4	C2, C4, C6	U5
G5	F2, F5	C4, C6, C5	U9, U8
G6	F2, F5	C6, C8,C9	U10
G7	F2, F5	C2,C8	U10
-	F0	C11	U1, U2, U3

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 provide 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.

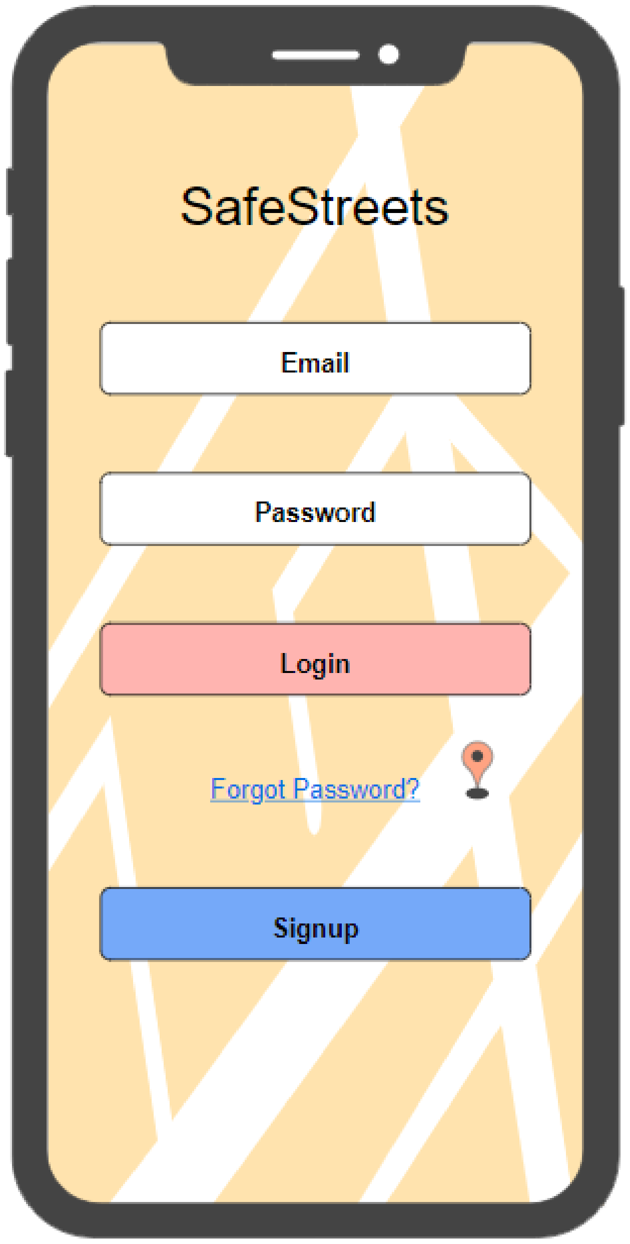


Figure 3: SafeStreets Login page.

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.

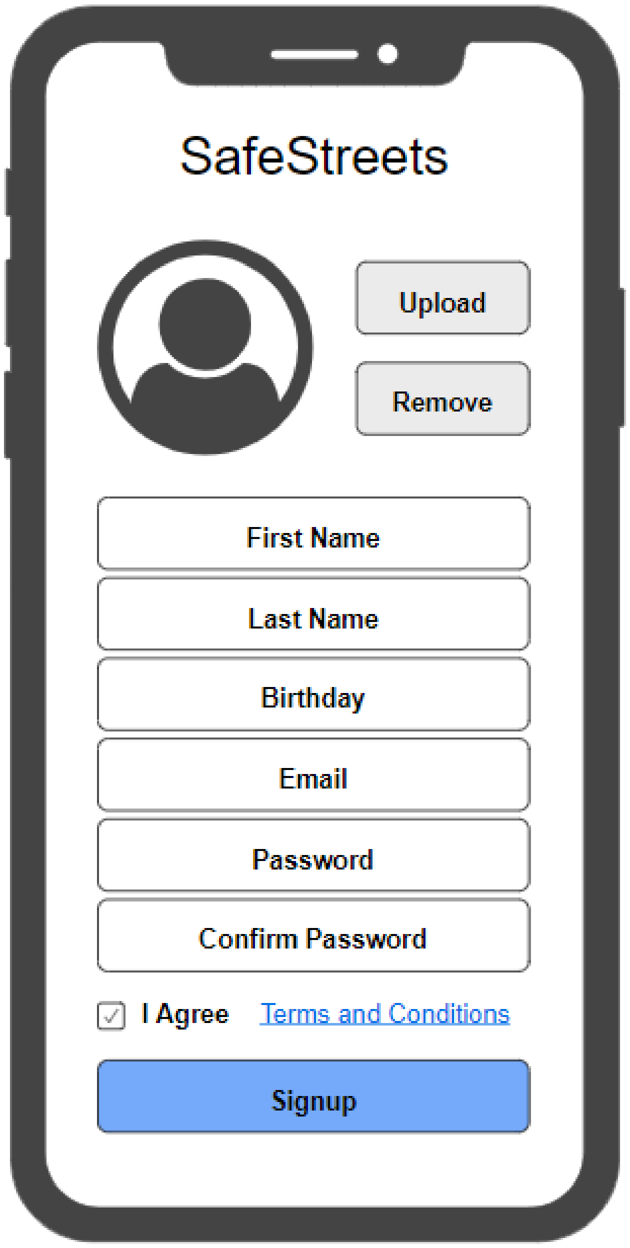


Figure 4: SafeStreets SignUp page.

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.

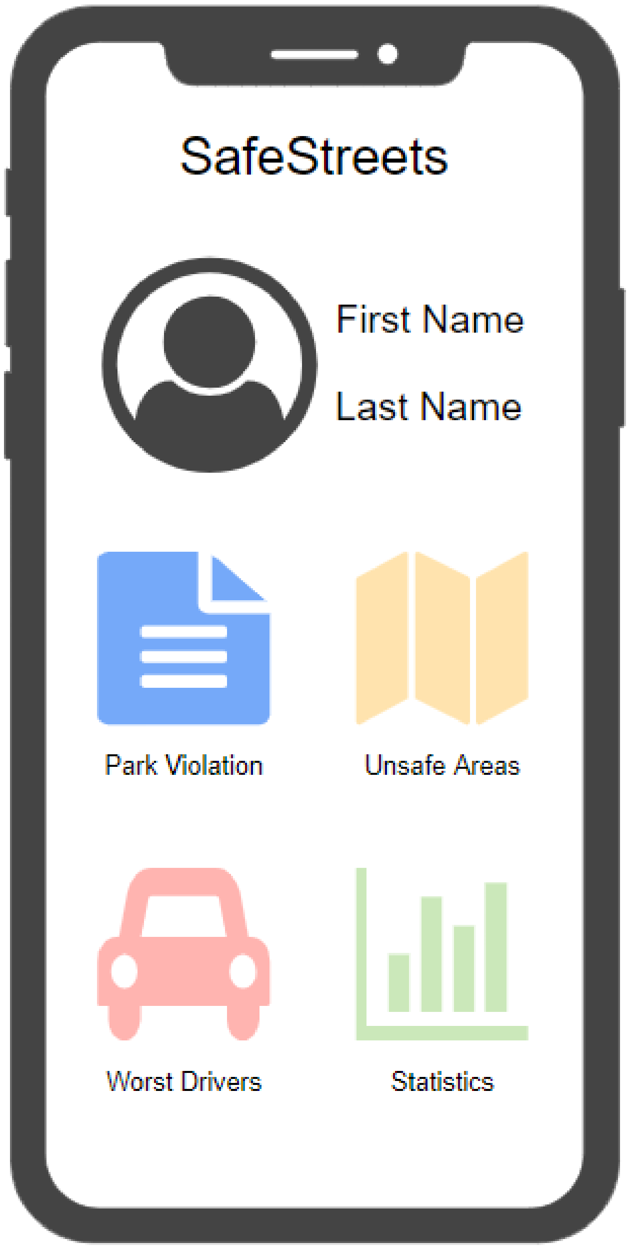


Figure 5: SafeStreets Home page.

Violation Page:
If the user wants to send to the officers a parking violation of some sorts a form to be filled is provided. The user can either send a report that will generate an automatic ticket or that will need to be checked by the officers.

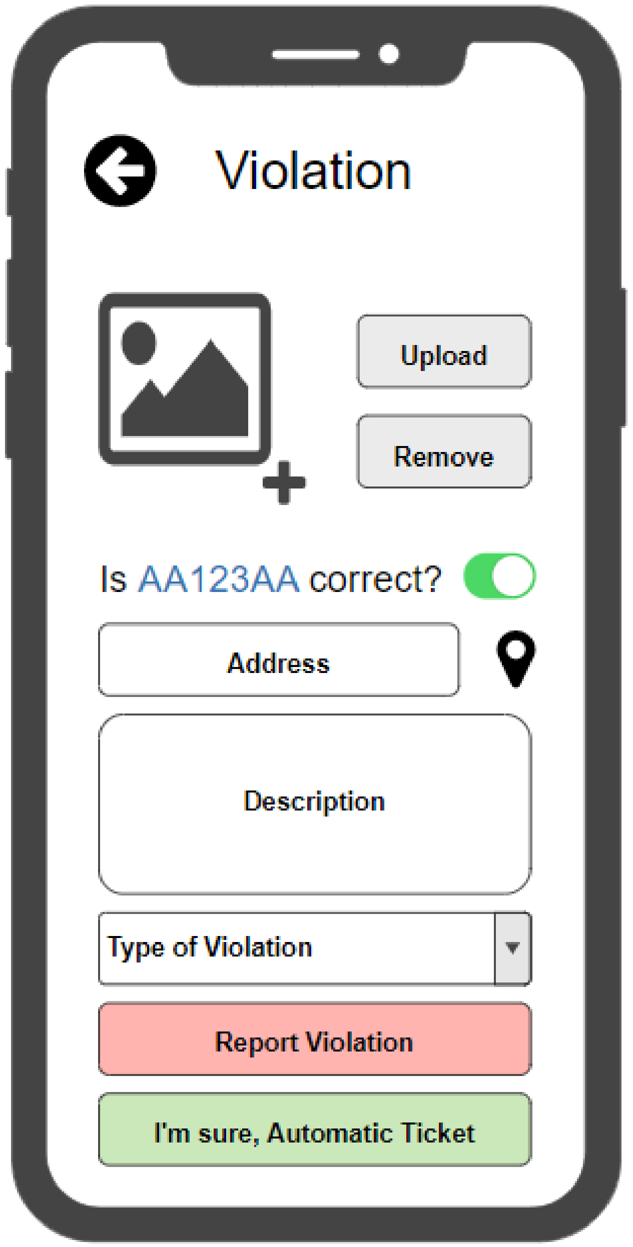


Figure 6: SafeStreets Violation page.

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.

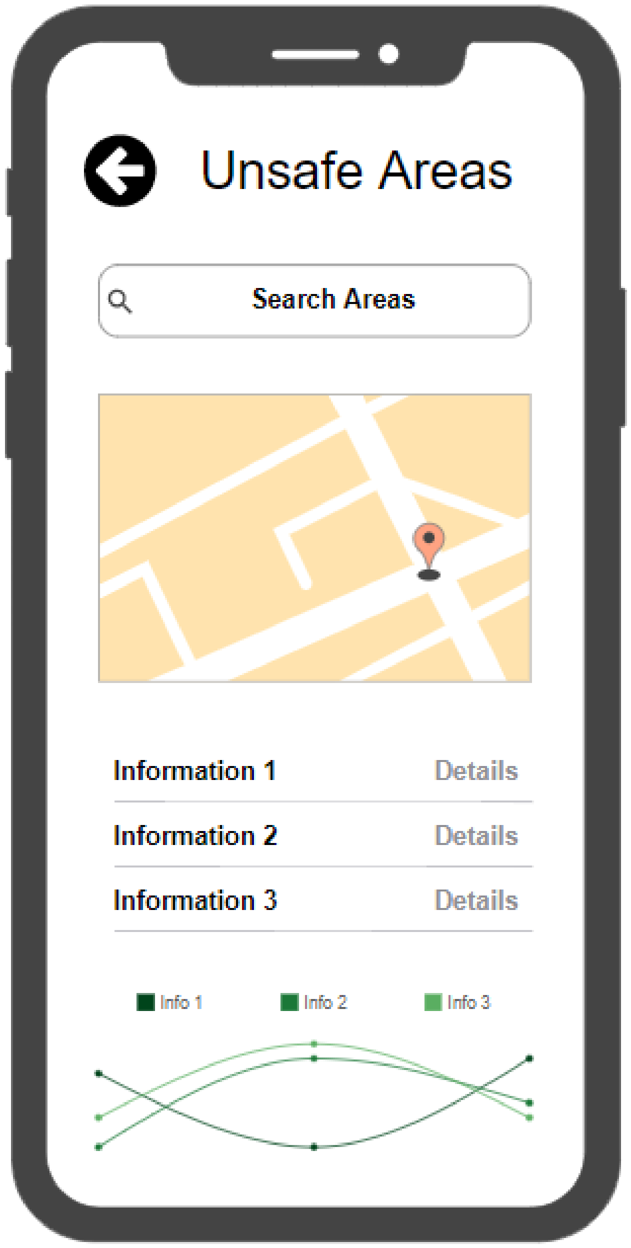


Figure 7: SafeStreets Unsafe Areas page.

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.

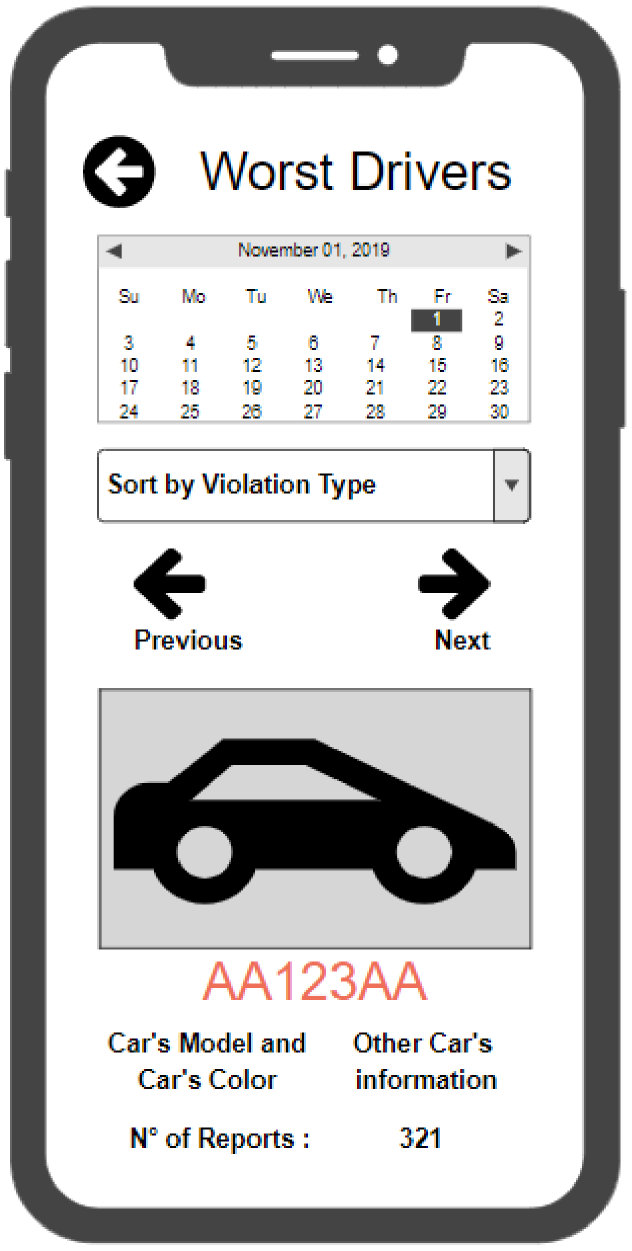


Figure 8: SafeStreets Worst Drivers page.

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

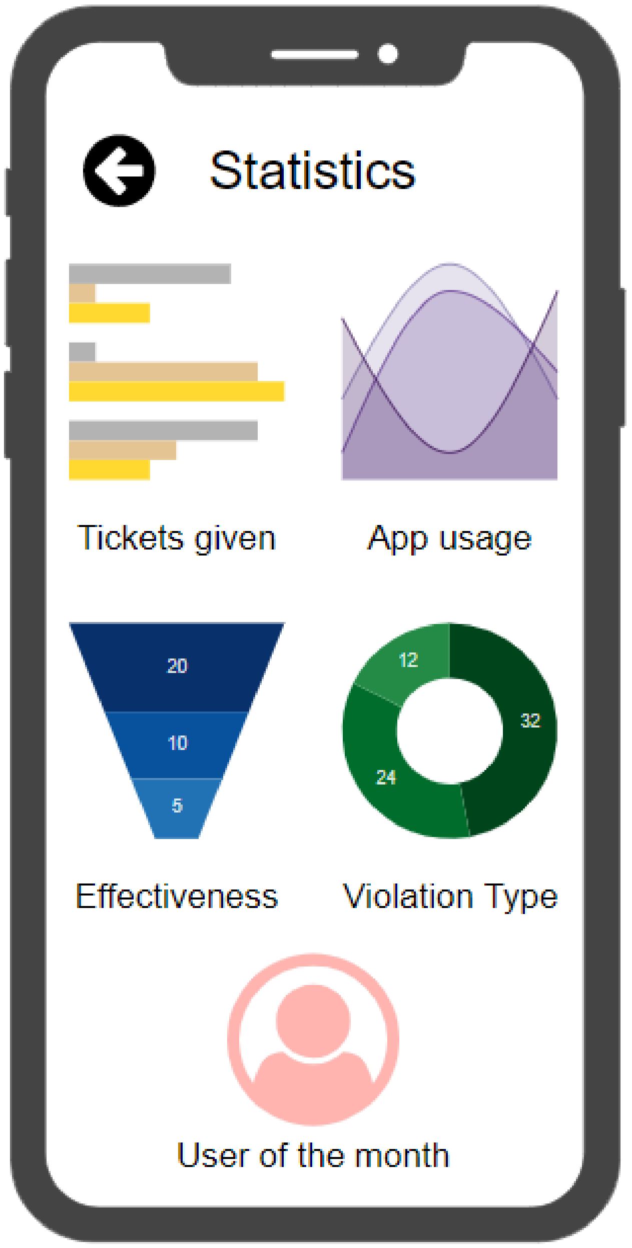


Figure 9: SafeStreets Statistics page.

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.

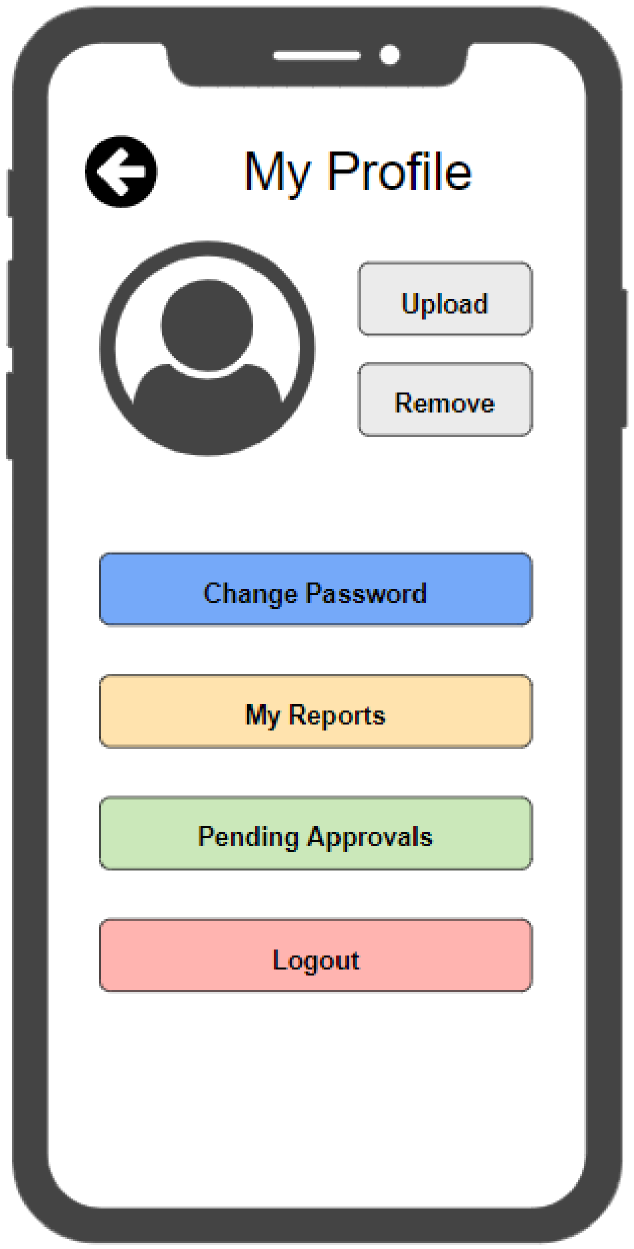


Figure 10: SafeStreets User Profile page.

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

	Creation of a report
Actors	Vigilio Ferri: a retired helper of people in needs and a rude man
Flow of events	Vigilio is an active 70 years old that likes long walks and sitting on the bench in the park near his home. It's 11 A.M. and as every day Vigilio goes for his morning walk. He likes to do a lap around his neighborhood, stopping and the newsstand down the street for his daily newspaper. Walking by, he notices a huge BMW that after doing 60 km/h near the neighbor elementary school brakes vigorously and stops exactly on a park reserved for the disabled. Thinking it was an emergency and given his past as a care giver he runs to the car only to find out that is just a 30 years old man that crosses the street and goes to the barber, where he was probably late for the appointment. Annoyed by that man behavior, but most importantly knowing that her friend who is on a wheelchair lives right near that park and could need it, he takes out his smartphone, opens SafeStreets and without hesitating he proceeds to report the violation that he had just witnessed. Working with people in needs his all life he is sure that since the BMW does not have a handicap placard exposed, the ticket for the rude man is guaranteed, so he sends a report with an automatic ticket. He waits a few second and... All Done! No officers needs to come to check the violation or no authorities has to be sure that he violation occurred, the tickets is already signed and a consistent fine awaits the young man. In just over a minute Vigilio can now continue his walk knowing that he did something good for the neighborhood and for the people in need he cares about a lot.

Table 5: Scenario 2

3.2.3 Third 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 6: Scenario 3

3.2.4 Fourth 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 7: Scenario 4

3.2.5 Fifth 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 8: Scenario 5

3.2.6 Sixth 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 9: Scenario 6

3.2.7 Seventh scenario

	Analytics consulting by users
Actors	Alberto Legno: A frantic motorist and a new SafeStreets user
Flow of events	Alberto lives and works in the suburbs, however he owns a restaurant in the city for which he has to attend 3 days a week. His local is placed in a historic area of the city where most of the parking lots present are reserved to the residents of that area, so when he does not find a valid spot, he usually proceeds to just park his motorbike where he shouldn't. Since the introduction of SafeStreets, Alberto is taking more fines than ever, still he thinks that is better to take some fines rather than having to search hours for a valid parking spot. Alberto is aware that he is very often fined by SafeStreets users,so he has recently downloaded the application on his mobile phone. One day Alberto, in his free time, began to consult the statistics of SafeStreets. He was shocked when he realized that in the section dedicated to the worst drivers of his restaurant area there was his car model with his license plate on it, reported so many times by the users to be considered a record. He stopped for a while to think at it and he realized that it is really rude to illegally occupy the positions of those who, by right, would need them! Impersonating himself in one of the inhabitants of that area in fact, Alberto realized how difficult would have been to find parking near his home if a lot of people would occupy the residents parking lots like he does. From that day on, Alberto stopped to illegally occupying the parking spaces reserved for residents and purchased a quarterly pass for a private parking near his restaurant.

Table 10: Scenario 7

3.2.8 Eighth scenario

	Analytics consulting by users
Actors	Oronzo Argento: a cutting-edge mayor and a man passing by
Flow of events	Oronzo is a 50 years old guy with a passion for electronics and new technologies in general. He is the mayor of a city where SafeStreets is not yet implemented. He is very much aware of the modern ways with which authorities can enforce the law and having in mind his city has a modern one he is generally prone to implement these new ways to keep citizens safe in his jurisdiction. Oronzo's sister lives in a huge metropolis (where SafeStreets is running for some months) and he goes to her place every Sunday to have lunch and to discuss about their lives. One Sunday, after having parked his car in front of his sister's house, he notices a person on the sidewalk who is taking a series of photos of a car parked in front of a vehicles exit. Curious, Oronzo asks to the man if there were some problems, and the man replies that he was just sending a report on SafeStreets so that the police could fine the car owner. Perplexed, since it was the first time Oronzo eared about SafeStreets, he does a rapid search on the application store of his phone and proceeds to download the app. After seeing how the application works a specific chart in the statistics section lights up his eyes: the effectiveness of SafeStreets is amazing! The next day back to work he has already made up his mind: SafeStreets will land on Oronzo's city's streets as soon as possible.

Table 11: Scenario 8

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

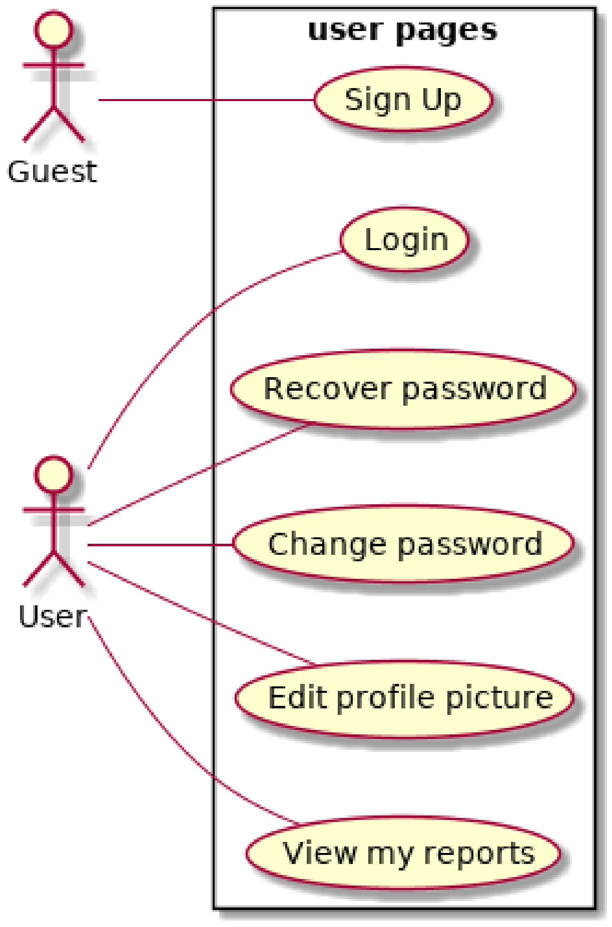


Figure 11: Use cases relative to the user registration and authentication

1. **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 mandatory field empty, he was already registered or the username was already taken: an error message is displayed and the user is asked to fill the form again.

Table 12: Use Case table - Sign up

Diagrams

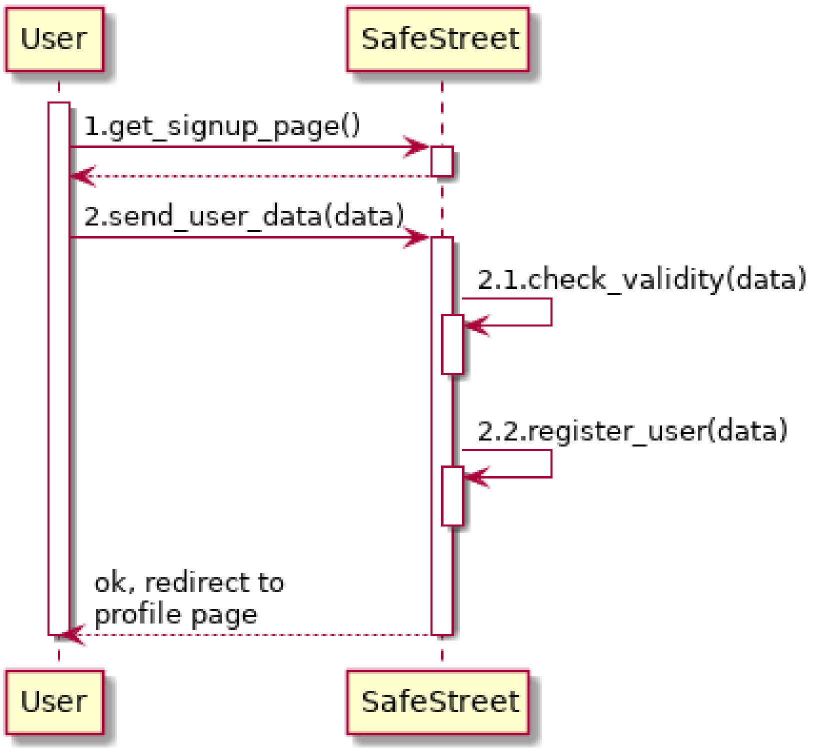


Figure 12: Use case diagram of the signup process

2. **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 13: Login Use Case table

Diagrams

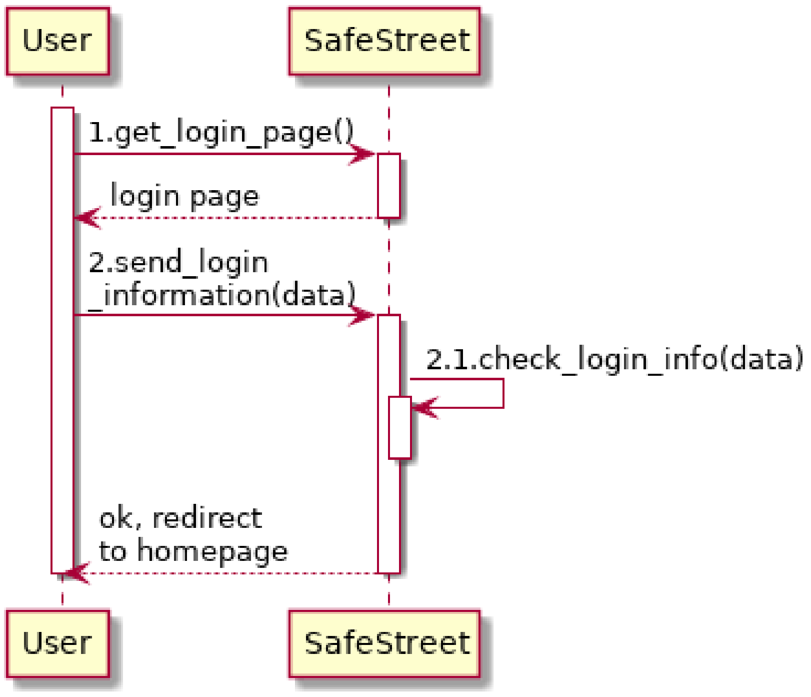


Figure 13: Use case diagram of the login process

3. **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 14: Recover password Use Case table

Diagrams

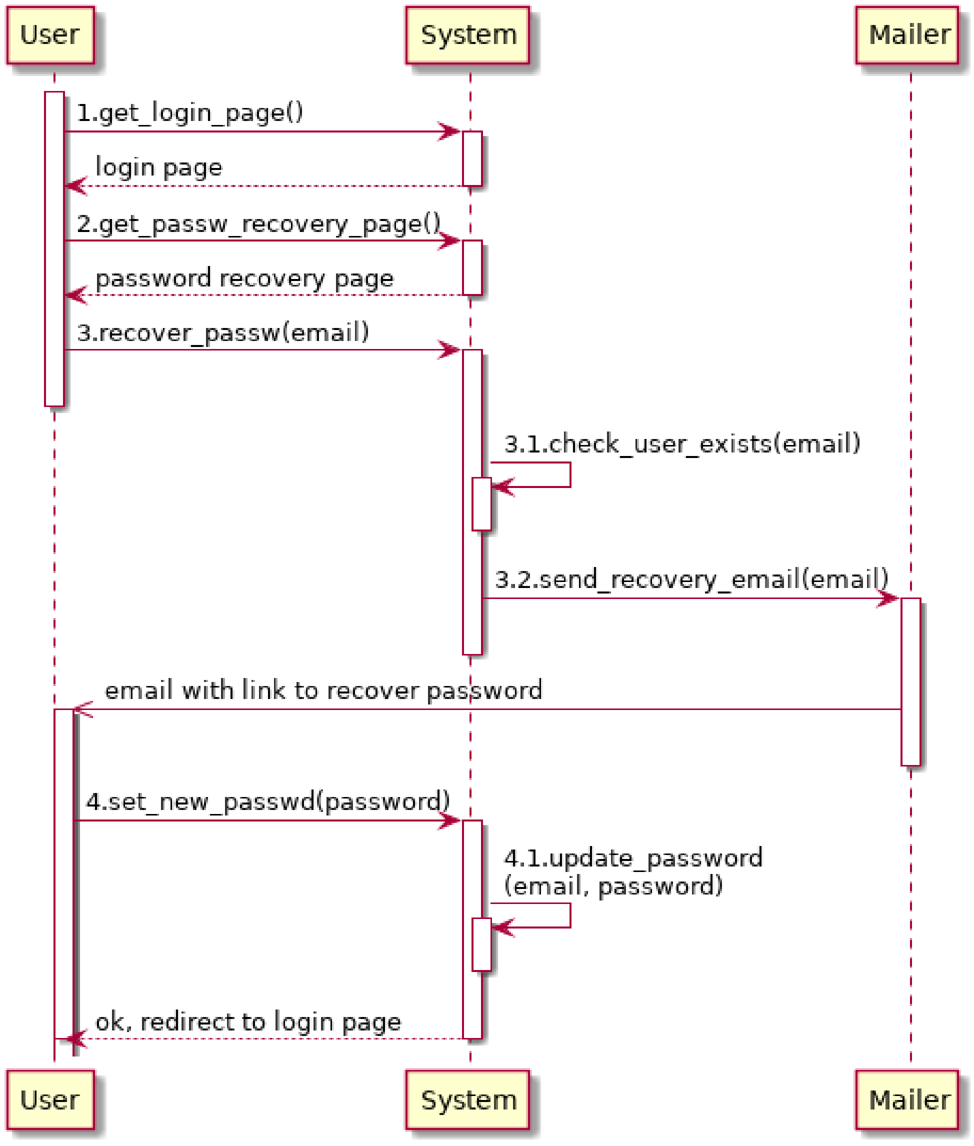

User
System
Mailer
1.get_login_page()
login page
2.get_passw_recovery_page()
password recovery page
3.recover_passw(email)
3.1.check_user_exists(email)
3.2.send_recovery_email(email)
email with link to recover password
4.set_new_passwd(password)
4.1.update_password (email, password)
ok, redirect to login page
User
System
Mailer

Figure 14: Use case diagram of the password recovery process

3.3.2 Report and Information Management use cases

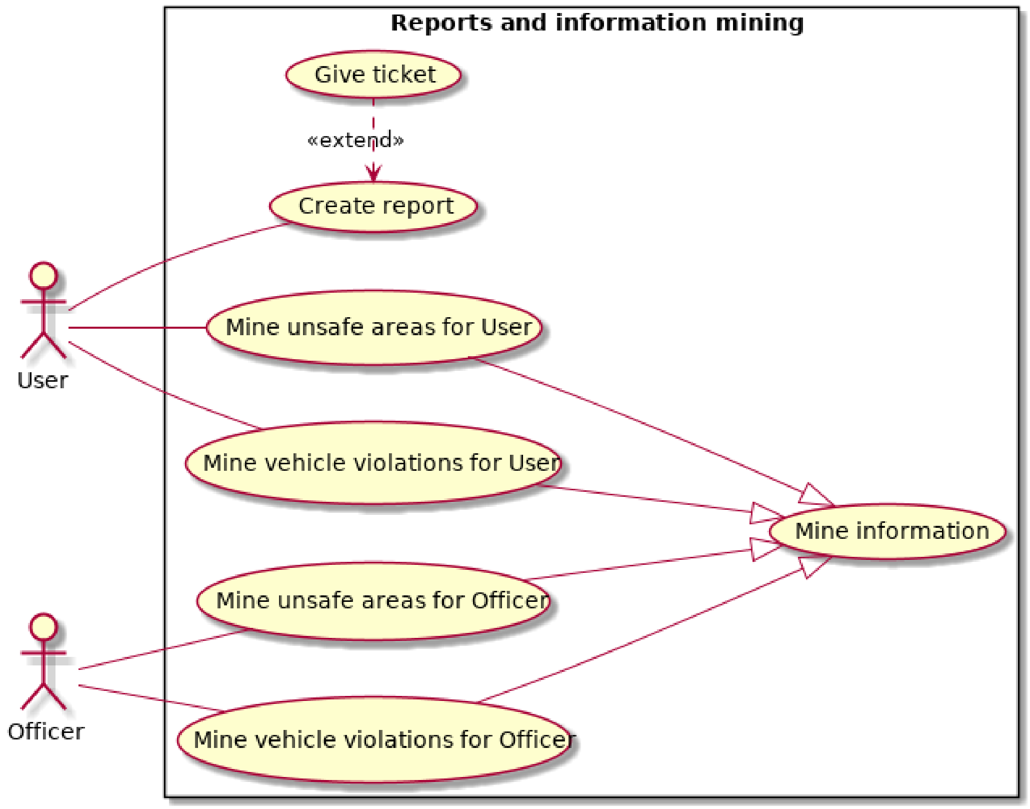

Reports and information mining
Give ticket
«extend»
Create report
User
Mine unsafe areas for User
Mine vehicle violations for User
Mine information
Mine unsafe areas for Officer
Officer
Mine vehicle violations for Officer

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

4. **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 input form to create a 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 does not click on the "Verified Violation" checkbox because he's not sure of it being a violation or not. • 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 and redirects him to the main page.
Exit conditions	The new report with the user-inserted data is created into the SafeStreets system.
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. • The violation results in a duplicate or the license plate is not readable: the system put this report in a revision queue and block the process.

Table 15: Report Creation Use Case table

Diagrams

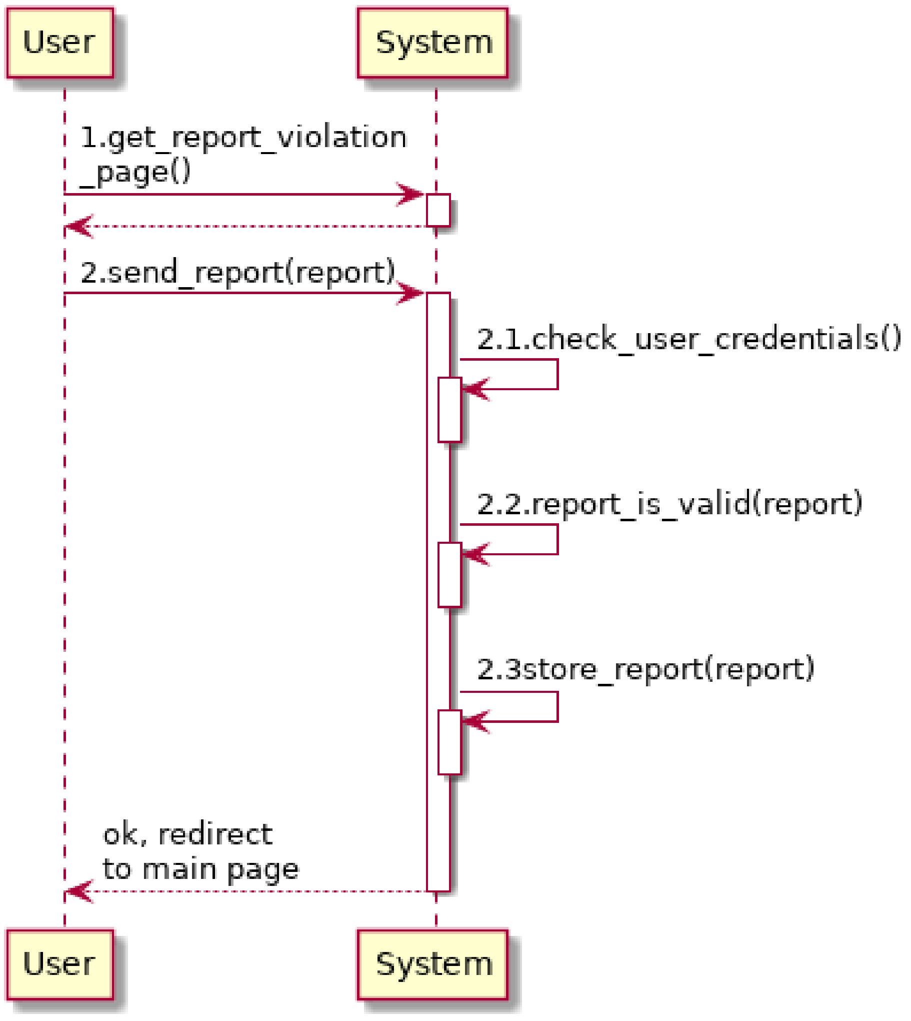


Figure 16: Use case diagram of the report creation process

5. **Automatic Traffic Ticket generation**

Name	**Automatic Traffic Ticket generation**
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 input form to create a 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 ”Verified Violation” checkbox because he is sure of it being a violation because it's evident or any other motivation. • 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 and redirects him to the main page.
Exit conditions	The new report is created into the SafeStreets system and passed to the Local Police service.
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. • The violation results in a duplicate or the license plate is not readable: the system put this report in a revision queue and block the process.

Table 16: Automatic Traffic Ticket generation Use Case table

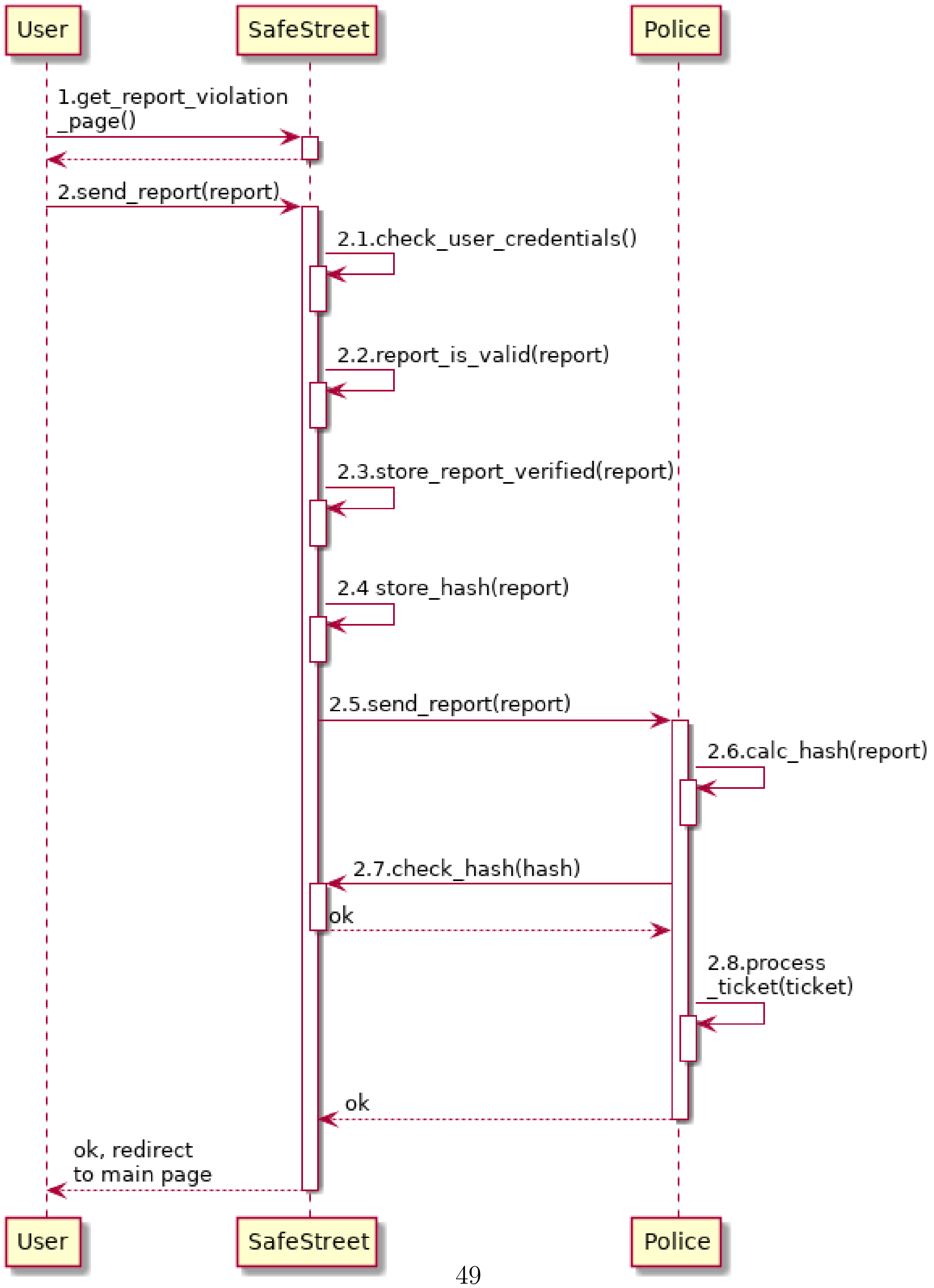

User
SafeStreet
Police
1.get_report_violation _page()
2.send_report(report)
2.1.check_user_credentials()
2.2.report_is_valid(report)
2.3.store_report_verified(report)
2.4 store_hash(report)
2.5.send_report(report)
2.6.calc_hash(report)
2.7.check_hash(hash)
ok
2.8.process _ticket(ticket)
ok
ok, redirect to main page
User
SafeStreet
Police

Figure 17: Use case diagram of the traffic ticket generation process

6. **Information Mining by Users (Unsafe Area)**

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 "Unsafe Areas" button and is redirected to the page with the general SafeStreets statistics about various areas in the city. • The user clicks on the selection menu to choose which area he wants to see, possibly entering its name or an address in that area. • The user could analyze the data that will be shown with a practical map and the textual information. • 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 (these 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 17: Data Mining Use Case table

Diagrams

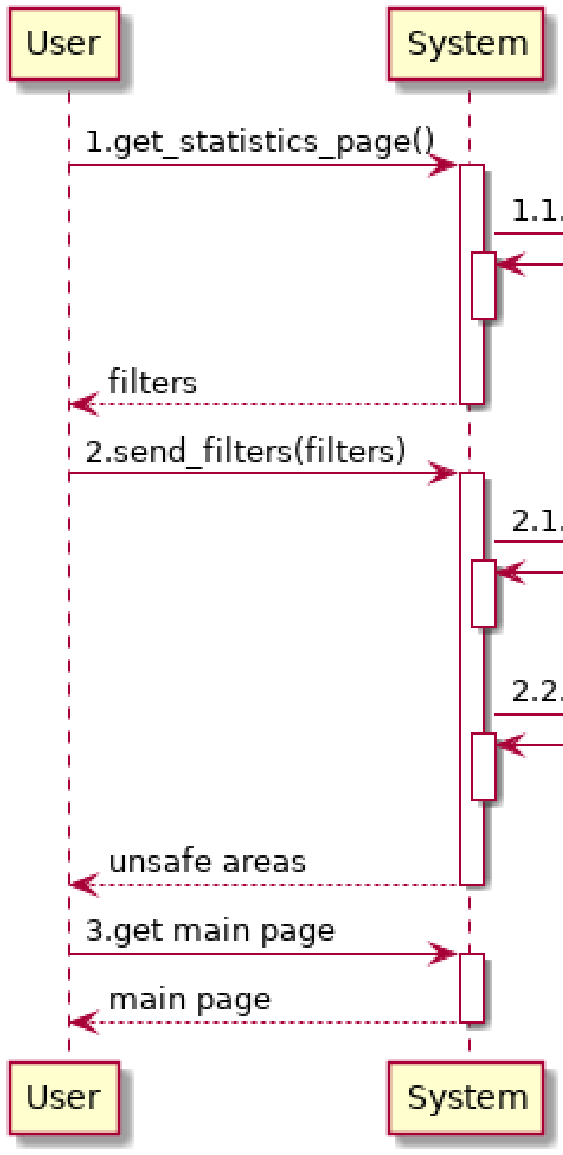


Figure 18: Use case diagram of the information mining process (unsafe area) for a user

7. **Information Mining by Officers (Unsafe Area)**

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 "Unsafe Areas" button and is redirected to the page with the general SafeStreets statistics about various areas in the officer municipality. • The officer clicks on the selection menu to switch between statistics about areas related to his municipality, possibly exporting it in CSV. • The officer could analyze the data that will be shown in textual and graphical form. • 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	The selected area is actually not present on SafeStreets or nonexistent: an error is displayed and the officer is asked to modify the inserted location accordingly.

Table 18: Data Mining for Officers Use Case table

Diagrams

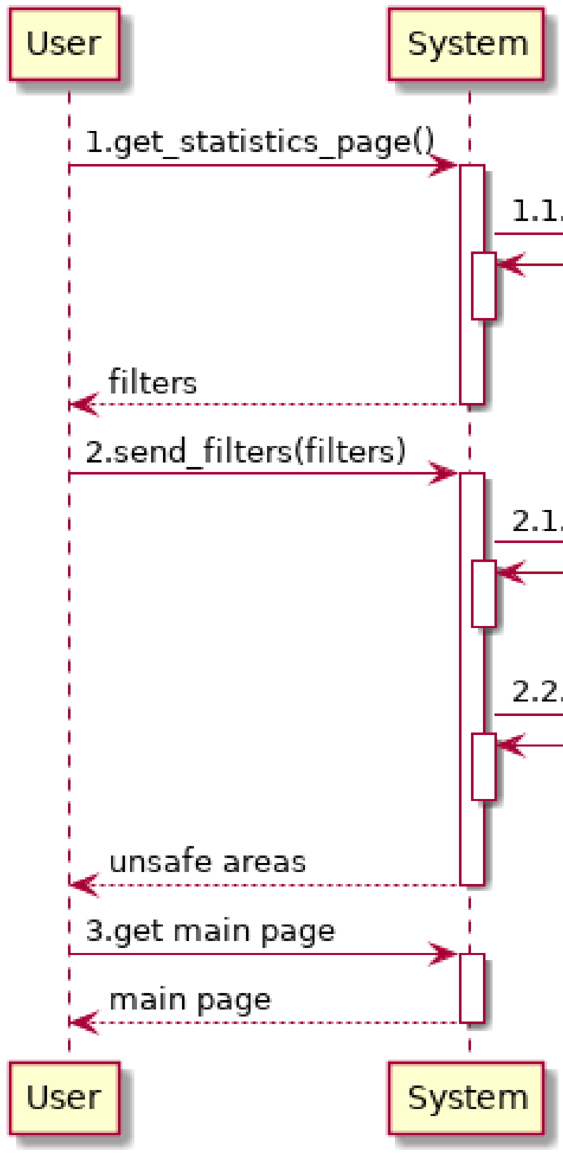


Figure 19: Use case diagram of the information mining process (unsafe area) for an officer

8. **Information Mining by Users (Vehicle Violations)**

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 "Worst Drivers" button and is redirected to the page with the general SafeStreets statistics about vehicle infractions in the city grouped by license plate. • The user clicks on the calendar menu to choose which time frame he's interested in, possibly entering its more information such as the violation type. • The user could analyze the data that will be shown in order of number of violations. • 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 (these kind of information is not sensitive).
Exceptions	The inserted time frame is invalid: an error is displayed and he is asked to modify the desired time frame.

Table 19: Data Mining for Vehicle Use Case table

Diagrams

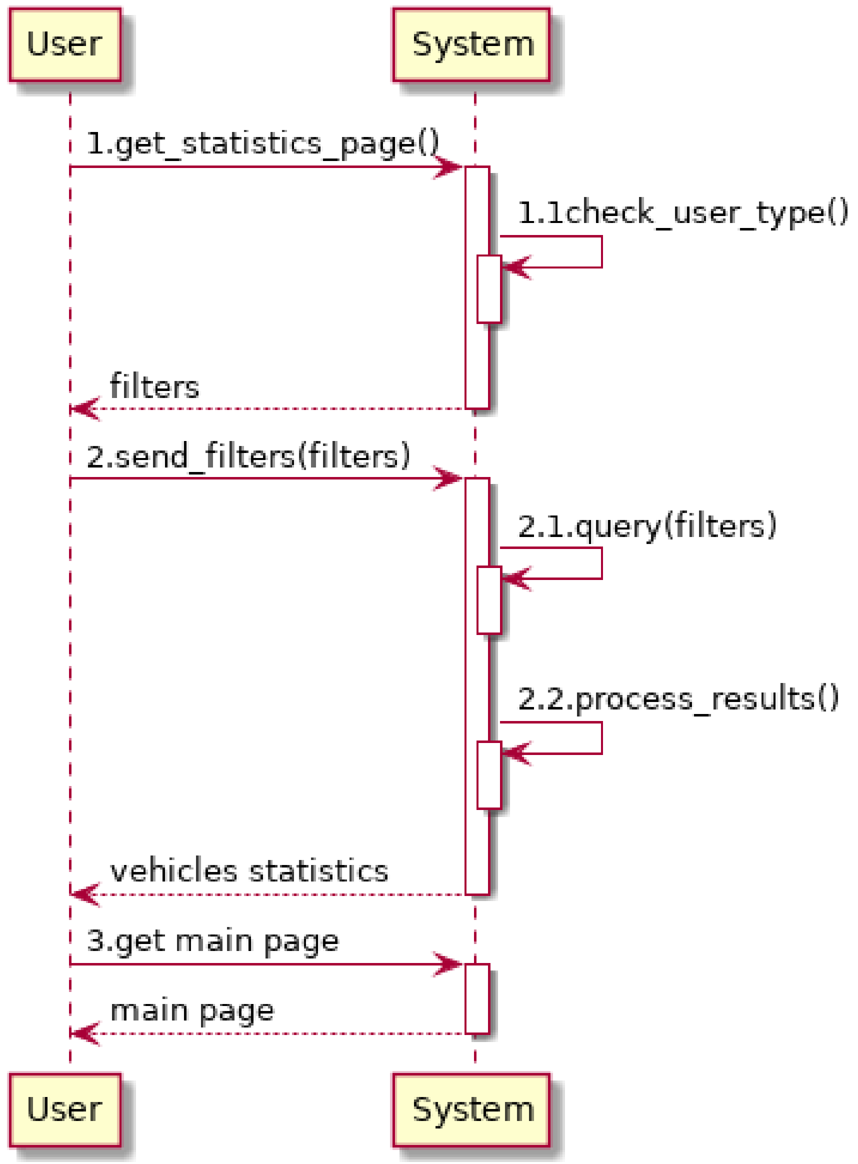


Figure 20: Use case diagram of the information mining process (vehicle violations) for an officer

9. **Information Mining by Officers (Vehicle Violations)**

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 "Worst Drivers" button and is redirected to the page with the general SafeStreets statistics about vehicle infractions in the officer municipality grouped by license plate. • The officer clicks on the calendar menu to choose which time frame he's interested in, possibly entering its more information such as the violation type with the possibility to export it in CSV. • The officer could analyze the data that will be shown in both textual and graphical form. • 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	The inserted time frame is invalid: an error is displayed and the officer is asked to modify the desired time frame.

Table 20: Data Mining for Vehicle by Officers Use Case table

Diagrams

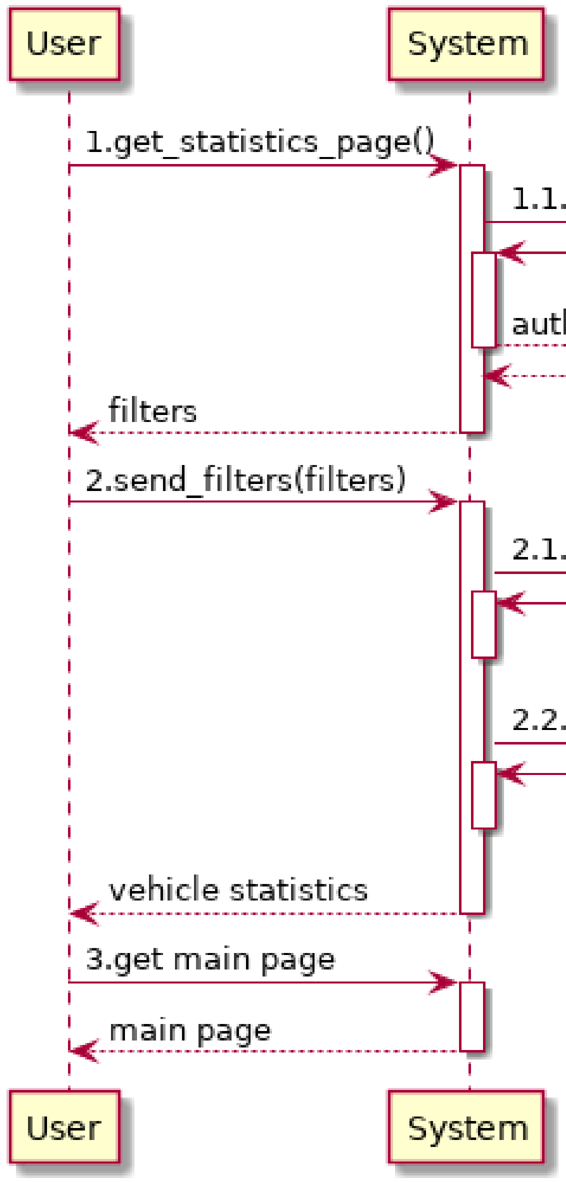


Figure 21: Use case diagram of the information mining process (vehicle violations) for an officer

3.3.3 Statistics use cases

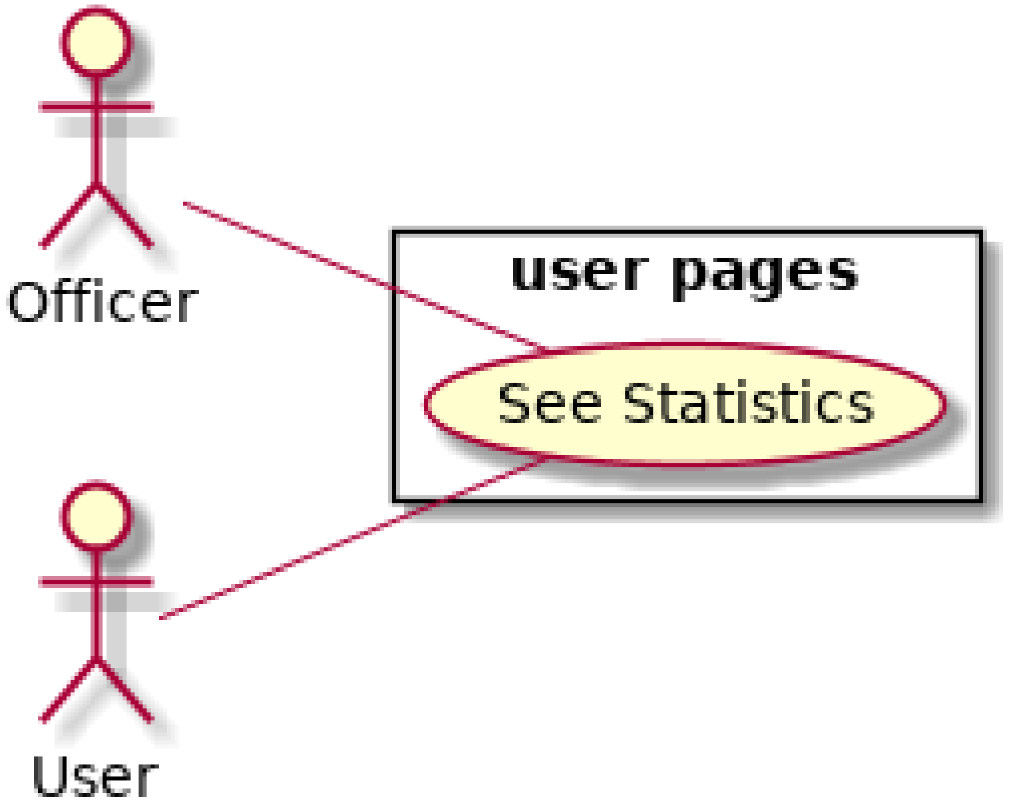


Figure 22: Simple use case demonstrating the functionalities of SafeStreets application to provide statistics of its usage.

10. **Statistics and analytics consulting**

Name	**Statistics and analytics consulting**
Actors	Both Users or Officers
Entry conditions	The actor is logged in and he is in the main page.
Flow of events	• The actor clicks on the ”Statistics” button and is redirected to the page with alle the general SafeStreets statistics such as Tickets given, App usages, Effectiveness, Violation Types... • The user clicks on the statistic he’s interested in. • The user could analyze the analytics provided and can compare different analytics to acquire knowledge about the SafeStreets project. • After he finishes, he presses the Home button. • The actor is then redirected to the main page.
Exit conditions	The actor-requested analytics is displayed to the user (these kind of information is not sensitive).
Exceptions	None.

Table 21: Statistics consulting Use Case table

Diagrams

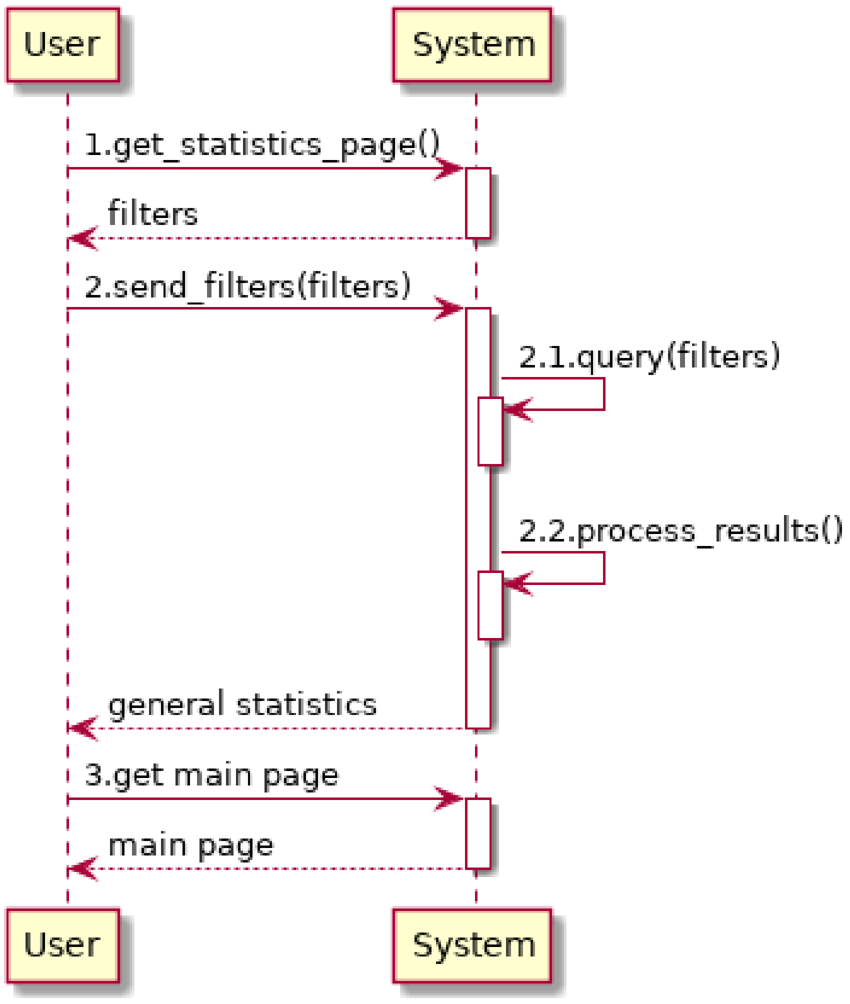


Figure 23: Use case diagram for an actor examining statistics

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, 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 to the user to use the application, indicatively they should take 5 seconds at most.

3.4.2 Reliability

As the system stores sensible data, it must be ensured that it is highly reliable and fault tolerant. For example, the central server, which contains all the information, should be duplicated.
The running processes (which provide the automatic ticket functionality) should be trustworthy, so if something goes wrong with an automatic ticket, that report that was generating the ticket should be put in a queue waiting for a manual analysis. In this way the consistency of information is ensured.

Any other technique can be adopted to ensure the required reliability.

3.4.3 Availability

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 downtime could be acceptable.
So, as SafeStreet does not need to be up to ensure critical aspects, but still some violations could have only a short period of time to be checked, it will have to guarantee a 99.9% availability (43 minutes of downtime every month).

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.
A public key encryption mechanism could be put in place to crypt reports when received by SafeStreets and to decrypt them only when the Police server has received them. As an alternative, the reports could be digitally signed with a trusted key of SafeStreets property.
Users also give their personal data when registering, and their privacy must be guaranteed. Techniques such as data encryption could be used to archive this.

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, the system must handle 1 million registered users and a peak of 7.000 concurrent requests.

3.4.6 Maintainability

An easy to fix, modify or update code is preferable because, according to the circumstances actions on it could be done. That action should be done without too much pain and with a cheap cost. Appropriate design patterns will be used, as it will be better explained in a further document.

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 Simple User Interface

The user interface has to be as simple and intuitive as possible, the application should allow an average user to set up an account and start using the application understanding its functionality in no more than a dozen minutes.

3.4.9 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 {} {
    //each email belongs to 1 customer only
    one u: Customer | this = u.email
}
sig Password {} {
    //A password must belong to a customer, but
       different customers can have the same pwd
    some u: Customer | this = u.password
}

//either a user or an officer
abstract sig Customer{
    email: one Email,
    password: one Password
}

sig User extends Customer {
    reports: set Violation
}
sig Officer extends Customer{
    handledViolations: set Violation
} {
    //officers handle only violations without
       ticket
    ∀ v : handledViolations | v.ticket = none
}

```

```
//The police system of a city
sig LocalPolice{
    officers: set Officer,
    violations: set Violation,
    accidents: set Accident,
    positions: set Position,
    unsafePositions: set UnsafePosition,
    confirmedTickets: set Violation
}


sig LicensePlate {} {
    //∀ licensePlates must belong to a picture
    some p: Picture | this = p.licensePlate
}

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

//An accident registered in the police system
sig Accident {
    position: one Position,
    accidentType: one AccidentType
} {
    //∀ accidents must belong to one and only one
       local police
    one p: LocalPolice | this in p.accidents
}

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

//The core of the suggestions given by the system
sig UnsafeReason {
    accType: one AccidentType,
    violType: one ViolationType,
    suggestion: one Suggestion
}

sig UnsafePosition{
    reasons: set UnsafeReason,
    position: one Position
}

sig Suggestion {} {
    //every suggestion belongs to an unsafe
       position
    some u: UnsafeReason | this = u.suggestion
}

abstract sig AbstrHashes {
    hashes: Violation -> Hash
}

//The hashes calculated by the SafeStreet System
one sig Hashes extends AbstrHashes {}

//The hashes calculated by the police system
one sig PoliceHashes extends AbstrHashes {}

sig Hash{} {
    //A hash must belong to either a system hash
       or a police hash
    some h: AbstrHashes |
        this in Violation.(h.hashes)
}

//a violation must have exactly one license plate
fact {
    ∀ v: Violation | #v.pictures.licensePlate = 1
}

/* An UnsafePosition must belong to a position iff
    its violation and accident
happened at least 2 times in that location */
//NB: 2 is a re∀y low number used to make the
    model work, it is not the actual number
fact {
    ∀ u: UnsafePosition | some p: Position, vt:
       ViolationType, at: AccidentType, r:
       UnsafeReason |
        p = u.position and r in u.reasons and vt =
           r.violType and at = r.accType and
        #{v: Violation | v.position = p and vt = v
          .violationType} > 1 and
        #{a: Accident | a.position = p and at = a.
           accidentType} > 1
}

//2 unsafePositions cannot have the same reason
   and position
fact {
    no disj up1, up2 : UnsafePosition | some ur:
       UnsafeReason, p: Position |
        ur in up1.reasons and ur in up2.reasons
        and p = up1.position and p = up2.position
}

//2 unsafeReason cannot have the same accident
   type and violation type
fact {
    no disj ur1, ur2: UnsafeReason | ur1.accType =
        ur2.accType and
        ur1.violType = ur2.violType
}

//∀ positions, accidents, violations and officers
   must belong to 1 and only 1 localPolice
fact {
    ∀ p: Position | one lp: LocalPolice | p in lp.
       positions
    ∀ o: Officer | one lp: LocalPolice | o in lp.
       officers
    ∀ v: Violation | one lp: LocalPolice | v in lp
       .violations
    ∀ a: Accident | one lp: LocalPolice | a in lp.
       accidents
}

//a violation belongs to a local police iff its
   position is one of the local police positions
fact {
    ∀ v: Violation | ∀ lp: LocalPolice |
        v.position in lp.positions ≤> v in lp.
           violations
}

//a violation can be handled by an officer only
fact {
    ∀ v: Violation | no disj o1,o2: Officer |
        v in o1.handledViolations and v in o2.
           handledViolations
}

//an unsafePositions must belong to the
   LocalPolice that has its position
fact{
    ∀ up: UnsafePosition | ∀ lp: LocalPolice |
        (up in lp.unsafePositions) ≤> (up.position
            in lp.positions)
}

/* a ticket is confirmed iff its hash on the
   police system matches the one on
the SafeStreet System */
fact {
    ∀ v: Violation | v in LocalPolice.
       confirmedTickets iff
        v.(Hashes.hashes) = v.(PoliceHashes.hashes
           )
}

//A violation can be in confirmedTickets only if
   it has a ticket
fact {
    no v: Violation | v in LocalPolice.
       confirmedTickets and v.ticket = none
}

//∀ violations have their hash on SafeStreet
   System
```

```
fact {
    ∀ v: Violation | one h: Hash | v->h in Hashes.
        hashes
}

//a violation cannot have more than 1 hash on the
   police system
fact {
    ∀ v: Violation | #v.(PoliceHashes.hashes) ≤ 1
}

//Violations without tickets do not have an hash
fact {
    ∀ v: Violation | v.ticket = none implies v ∉
        PoliceHashes.hashes.Hash
}

//2 violations cannot have the same hash
fact {
    no disj v1,v2 : Violation | some h: Hash |
        v1->h in Hashes.hashes and v2->h in Hashes
            .hashes
}

//for a localPolice to confirm a ticket it must be
     in its violations
fact {
    ∀ lp: LocalPolice | ∀ v: Violation | v in lp.
        confirmedTickets implies v in lp.violations
}

/******************ASSERTIONS******************/
//2 localPolice cannot have the same violation
check {
    no disj lp1, lp2: LocalPolice |
        some v: Violation | v in lp1.violations
            and v in lp2.violations
} for 5
```

```

//a unsafePosition cannot belong to 2 different
   localPolice
check {
    ∀ up: UnsafePosition | one lp: LocalPolice |
        up in lp.unsafePositions
} for 5

//a violation cannot appear 2 times in the hashes
   system
check{
    ∀ v: Violation | #v.(Hashes.hashes) = 1
} for 5

//if a LocalPolice has a confirmedTicket it must
   have at its position
check {
    ∀ lp: LocalPolice | ∀ v: Violation | v in lp.
        confirmedTickets implies v.position in lp.
        positions
} for 5

/*****************PREDICATES*****************/

pred world1{
    #LocalPolice = 2
    ∀ lp: LocalPolice | #lp.violations>0
    #LicensePlate > 1
    #User = 2
    #Officer = 1
    ∀ u: User | u.reports ≠ none
}

pred world2{
    #Suggestion = 2
    ∀ u: UnsafeReason | some up: UnsafePosition |
         u in up.reasons
    #confirmedTickets = 0
}

pred world3 {
    #confirmedTickets > 0
    #Ticket > #confirmedTickets
    #Violation > #Ticket
    #LocalPolice = 2
    ∀ lp: LocalPolice | lp.violations ≠ none
    #Accident = 0
    #AccidentType = 0
}

run world1 for 5
run world2 for 5
run world3 for 5
```

4.2 Assertions

All the assertions have been tested and no counterexample was found. They have been tested with a cardinality of 5, and in our model it is enough to be highly positive that they will hold for any value.

4.3 Worlds Generated

4.3.1 World 1: Violations

The first world presents the most important base functions of the model: it has 2 users and an officer, all registered with different emails. The users report Violations, which have a position, a set of images and optionally a ticket. Each local police receives a violation if its position is among the positions of the police itelf, which form a perimeter around the city.

In each violation there is always at least an image which shows a licence plate that has been read by the system. All the violations also have a violationType and the police system also exposes accidents along with their type.

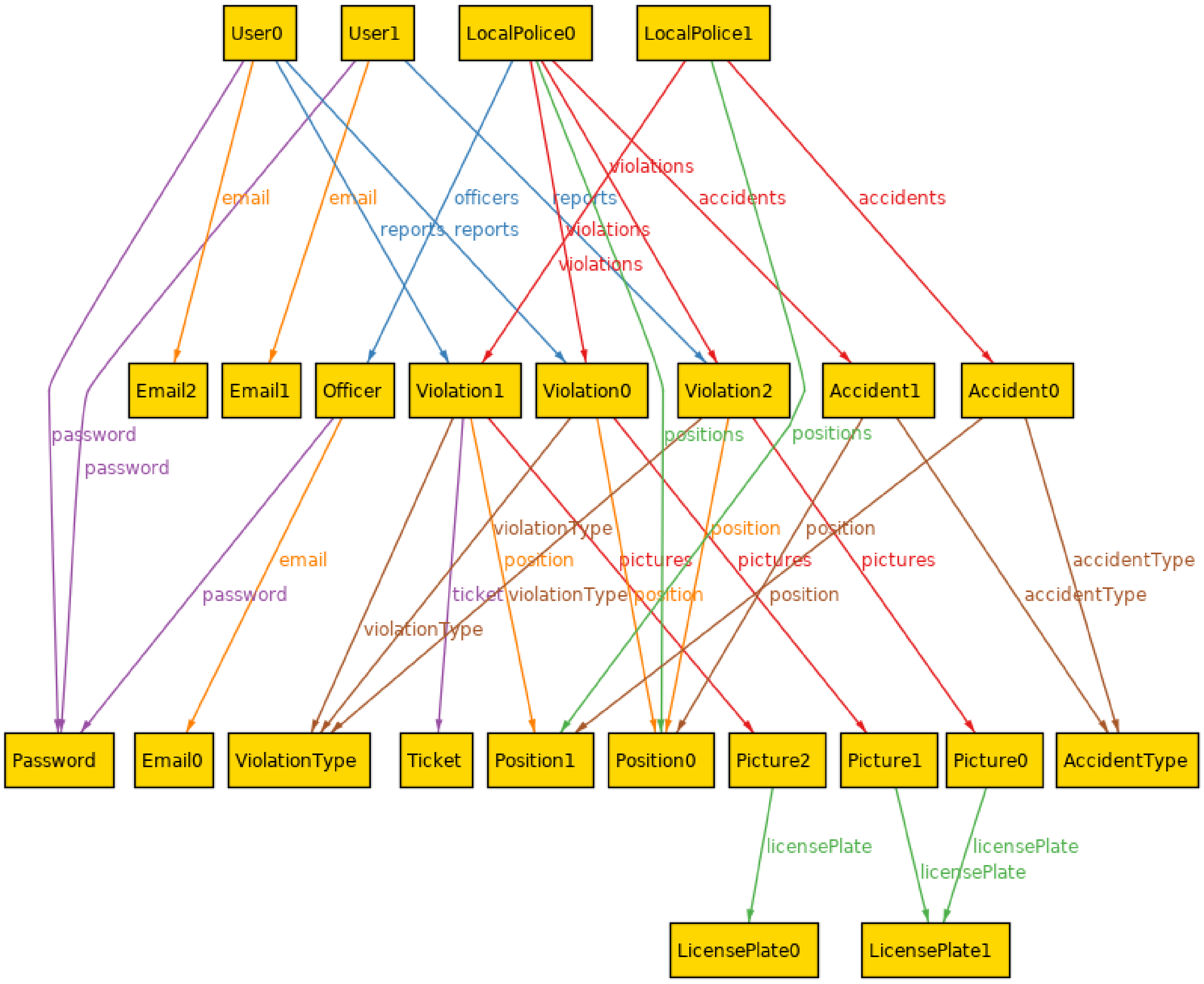


Figure 24: World 1 generation in Alloy

4.3.2 World 2: Suggestions

The second world shows a complete representation of the unsafe areas: each local police system gets signalled by SafeStreet a series of unsafe areas, which are connected to a position and are always in the police system handling that position. Each unsafePosition is connected to an unsafeReason, which is itself made by a violationType an accidentType and a suggestion. This is because the unsafeReasons are inserted by hand by operators using accident and violation categories.

The system detects an unsafe area if there are enough accidents and violations of a certain type in a position, and suggests a possible solution.

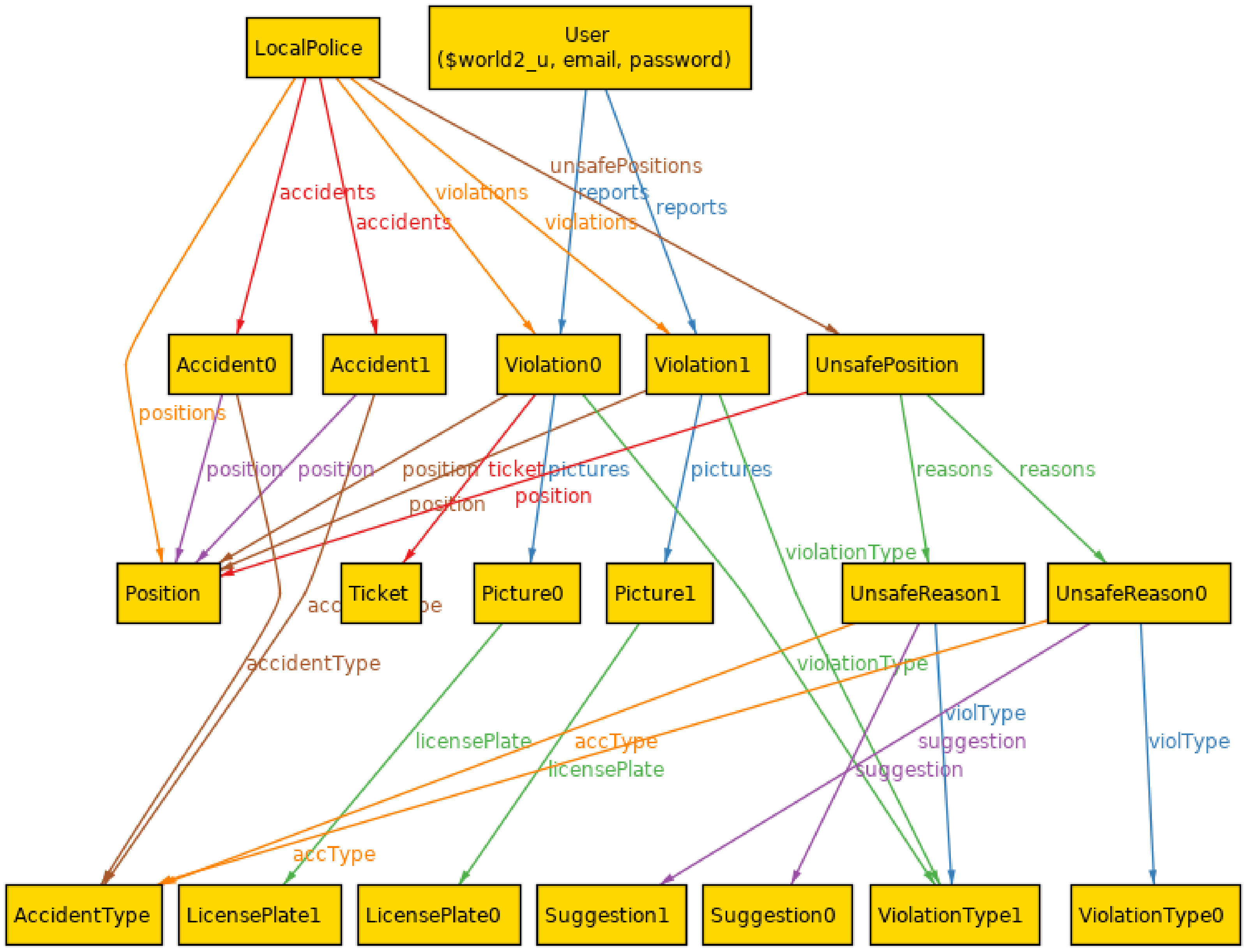


Figure 25: World 2 generation in Alloy

4.3.3 World 3: Alteration of information

This world represents the way in which SafeStreet is sure that from when a violation reaches its systems it is not altered until it reaches the police systems.

Each Violation is associated to a unique Hash in the Hashes system of SafeStreet, and a ticket is created only if the hash calculated on the police system matches the one on SafeStreet.

In the picture we can see that Violations 1 and 3 are confirmedViolations: this means they have the same hashes in the Hashes and PoliceHashes sig. Also, it is important to notice that the user who issued those violations requested a ticket

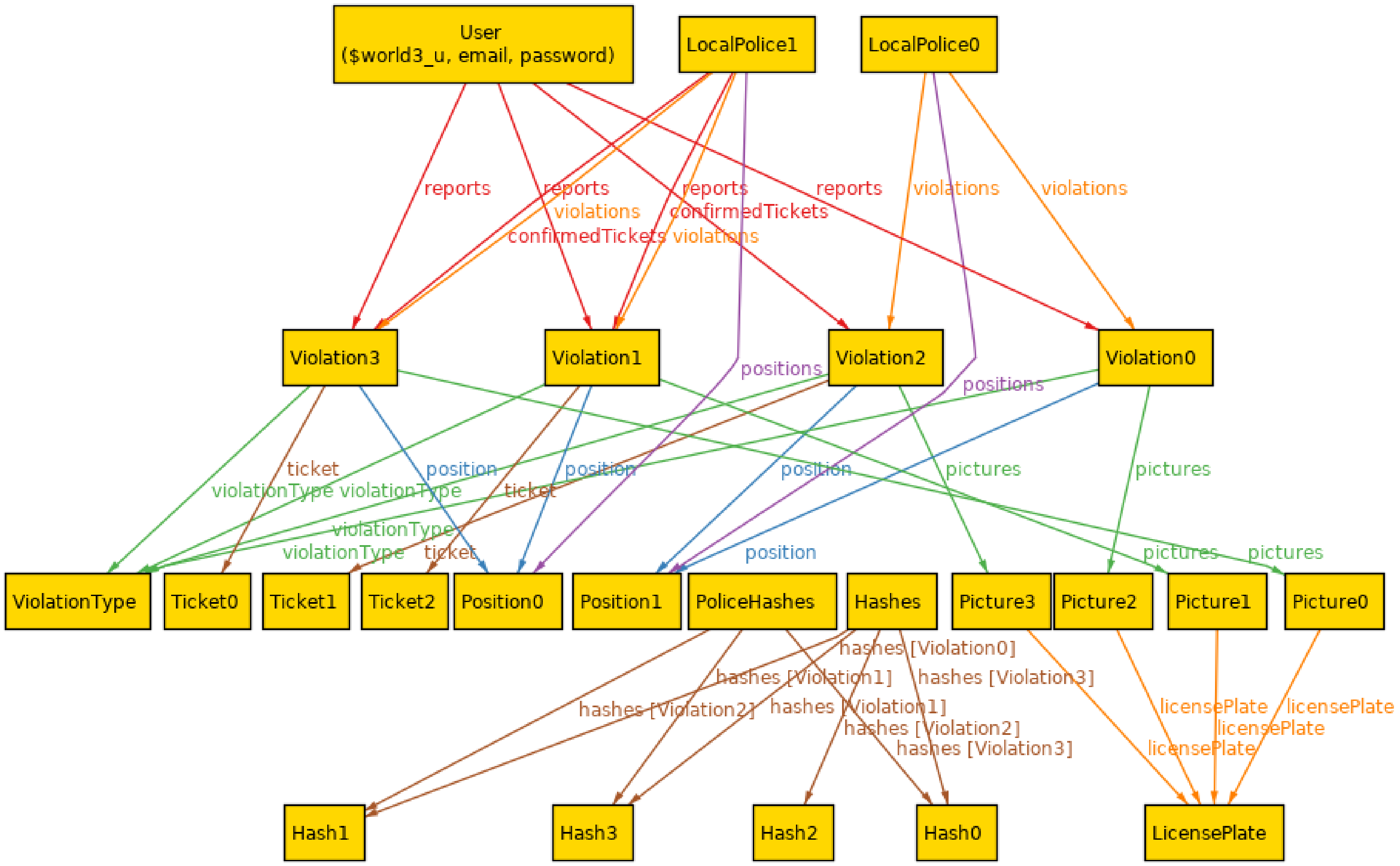


Figure 26: World 3 generation in Alloy

5 Softwares Used, Effort Spent and History

Software Used

Task	Software Used
Edit and compile LATEX code	TeXstudio and VisualStudioCode
UML modelling	PlantUML
Images and charts	draw.io
Run and compile Alloy	Alloy 4.2
Mockup	Moqups

Table 22: Softwares used

Effort Spent

Student	Hours spent
Andrea Furlan	close to 48
Cosimo Russo	close to 45
Giorgio Ughini	close to 49

Table 23: Effort spent

5.1 Revision history

- **v1.1 on 17/11/2019:** Added Computer Vision to double check automatic tickets and to mine vehicle information and minor fixes after a check with Luca Crippa - IBM.
- **v1.1 on 23/11/2019:** Modified diagrams and descriptions of 2.3.2 and 2.3.3 in order to be coherent with the microservices structure used in the Design Document.
- **v1.2 on 9/12/2019:** Modify sequence diagrams removing storage system to remove the concept of components