POLITECNICO
MILANO 1863

Computer Science and Engineering

A.A. 2019/2020

Software Engineering 2 Project:

"SAFE-STREET"

Requirements **A**nalysis and **S**pecification **D**ocument

December 15, 2019

Prof.Rossi Matteo Giovanni

Amirsalar Molaei

karim Zakaria Saloma

Erfan Rahnemoon

Contents

Figures

Tables

1 Introduction

1.1 Purpose

1.1.1 General Purpose

Nowadays, an ever-increasing number of cars and a shortage in the number of police officers caused the emergence of various traffic violations and accidents. Although two traditional solutions to solve these problems were rising the number of police officers and their equipment, due to the poor efficiency and inordinate cost, it is not feasible to continue this trend. This is where the power of technology can take the responsibility to help authorities to bring the order to the streets.

The only solution to assist authorities without expanding budgets is to participate in people with an intuitive and simple method. Hence, the *SafeStreets* app is proposed, which provides the possibility of reporting traffic violations and accidents by taking advantage of crowed-sourcing. Users can report violations by just taking pictures of infringement and license plate, then sending them.

1.1.2 Goals

The goals that the system aimed to achieve are presented as follows:

- [G-1] Users should be able to report traffic violations
- [G-2] Users should be able to access information regarding the safety of different areas.
- [G-3] Authorities should have access to the details of the traffic violations reported by the users.
- [G-4] Authorities should be provided with possible interventions to prevent violations.
- [G-5] Authorities should have access to refined data related to committed violations.
- [G-6] Users should be able to view reports that they have previously made.

1.2 Scope

The *SafeStreets* system shall be providing four main functions to various users; in this section, the system boundaries and scope used to define the limitations and different responsibilities of the system to be.

The first of the main functionalities is the enabling of users to report traffic violations. Regarding this, some phenomena are regarded as world phenomena not viewed by the system due to its limitations such as the fact that the system does not directly detect a violation. However, it can be accounted for by the system through a traffic report made by the users. Moreover, another functionality that has to do with the users is the publishing of collected data to be viewed by the users in a refined representation to help them consider the safety of various areas based on traffic violations. The data is also communicated to the authorities but with different levels of details.

The other two main functions have to do with the *SafeStreets* system providing services to government authorities. The domain limitations of the system affecting this interaction are also discussed in this section. Such as, the fact that the system is only able to make suggestions for preventive measures to the authorities based on the accident data that have been communicated. Meaning, that the system does not have any knowledge of accidents unless they are reported by the authorities and that the system can only suggest interventions and neither put them into place nor can detect them being applied. Moreover, a second function to the authorities would be the communication of traffic reports received from users to be later used by government officials to give out traffic tickets, the system responsibilities to support this process is to prevent the users from tampering with images *digitally* and to provide the collected reports to the authorities proactively. In other words, physical tampering with license plates to mislead authorities and the actual process of giving out tickets is not part of the application domain.

Below is a table summarizing and classifying the different phenomena that are related to the system functionalities. Main system functionalities:

→ ***F1:*** Reporting of violations

→ ***F2:*** Communication of collected data to users

→ ***F3:*** Suggestion of interventions

→ ***F4:*** Communication of reports for ticketing

Phenomena	Classification	Justification	Functionality
Physical tampering with license plate	World	Pure world phenomena since no measures are to be applied to detect nor prevent this phenomenon therefore it is unobserved by the system	F1, F4
Issuing of tickets	World	The actual issuing of the tickets is the responsibility of the authorities the system has no part in it and does not have access to the data regarding issued tickets	F4
Putting preventive measures for traffic violations into place	World	The application of preventive measure by the municipality is also a pure world phenomenon as the system has no means of knowing new measures by applied	F3
Traffic violations	World	The system does not directly observe or detect committed traffic violations if they are not reported by the user then the system cannot be held responsible for not having knowledge of them	F1, F4
Occurrence of accidents	World	Similarly, to traffic violations unless system acquires this kind of data through the authorities it has no way of detecting such phenomena	F3
Publishing of insights regarding the accumulated data	Shared	Performed by the machine observed by users and authorities in the world world	F2
Reporting traffic violation	Shared	Performed by users in the world observed by the machine	F1, F4
Publishing of accident data by the municipality	Shared	Performed by authorities in the world observed by the machine	F4
Suggesting interventions	Shared	Performed by system and communicated to authorities then if applied observed by the world	F3
Municipality report submission	Shared	Performed by system and communicated to authorities therefore it is observed by the world	F4

Table 1: World and machine phenomenas.

1.3 Definitions And Acronyms

1.3.1 Definitions

Latex Is a mark up language specially suited for scientific documents

Alloy Language and analyzer for software modeling

User Individual who wishes to exploit the system features

Unregistered User User that have not completed the registration and verification process

Registered User User that have completed the registration process and verified credentials

Activated User User that have granted access to phone camera and location

Authority Governmental body having jurisdiction

Municipality Single administrative division having corporate status and powers of self-government or jurisdiction

Violation Report A detailed description of a traffic violation

Safety Representation A graphical representation of safety on geographical basis

Refined Data Insights extracted through mining and aggregation of reports

1.3.2 Acronyms

S2B *Software To Be*

GPS *Global Positioning System*

UI *User Interface*

GDPR *General Data Protection Regulation*

UML *Unified Modeling Language*

RASD *Requirements Analysis and Specification Document*

OS *Operating System*

SMS *Short Message Service*

CI/CD *Continuous Integration/ Continuous Delivery*

GSF ID *Google Services Framework Identifier*

1.4 Revision history

Revision	Date	Author(s)	Description
1.0	10/11/2019	karim Zakaria Saloma, Amirsalar Molaei, Erfan Rahnemoon	First document issue
1.1	11/11/2019	Erfan Rahnemoon	Problem in CI/CD (automatic build from wrong HEAD)
1.2	15/12/2019	karim Zakaria Saloma	Final release version of Document

1.5 Reference Documents

References

[1] *Systems and software engineering — Life cycle processes — Requirements engineering IEEE 29148.* 2011.

[2] Alloy,
`https:alloytools.org`

[3] UML,
`https:omg.org/spec/UML`

[4] Specification Document. *SafeStreets Mandatory Project Assignment.*

1.6 Document Structure

The requirement analysis and specification document(RASD) is consisting of five chapters, an outline is presented as follows:

Chapter 1 is an introduction, which states the problems of traffic violations and also describes the previous unsuccessful methods that did not have tangible results. In addition to that, it illustrates the goals that the system aimed to reach and defines the scope of the system by pointing the world and shared phenomena.

Chapter 2 is about the perspective of the product along with details in actors, shared phenomena, assumptions, dependencies and constraints. On top of these for more elaboration class and state diagrams are used to demonstrate the general view of entities and behaviors of actors, respectively.

Chapter 3 contains information on interface requirements in terms of software, hardware and communication between them. Furthermore, user system interactions are represented in detail through use case diagram, sequence diagram and scenarios. Moreover, most significantly functional requirements with associated domain assumptions are declared, followed by design constraints and system attributes.

In **chapter 4**, Alloy which is a declarative specification language used to model behaviors and analyze various parts of the system.

Chapter 5 indicates the effort spent on each part of the document for each group member.

2 Overall Description

2.1 Product perspective

The *SafeStreets* system can be viewed as a standalone system in the sense that it does not depend on other external systems in order to properly function. However, in order to provide some external services to other entities it requires the provision communication interfaces to facilitate the flow data to and from the system. To be more specific, to interact with the municipality it is needed to have the appropriate interfaces as channels to retrieve data to be processed and to provide processed and aggregated data.

Apart from the aforementioned system interaction, *Safastreets* seldom depends on external services or systems. This is specially true for the main features of the system; i.e., the services provided to the civilian users. Specifically, the service provided in order for users to report traffic violations and the *View Safety* feature which could be seen to built upon the first feature; since, it is considered as a geographically oriented representation of report data.

Surely, the internal modules of the system are codependent since the data collected from user and possibly retrieved from the authorities flows internally through the system in many forms such as raw report details, insights built on report details and graphical representations. *Figure 1* provides a high level representation of the entities interacting with the system and represents the relationship between abstract entities, such as, *Report* and *Safety Representation*.

Furthermore, since the users are viewed as the main entity interacting with the *Safestreets* system it is rather important to clearly define the different states of the user and the state defining interactions with the system. First and foremost, it is worthy of mention that all user-system interactions are done through the user interfaces to be presented in the following sections. Moving onto the relevant topic of user states as seen by the system, any user can be classified into one of three categories; this classification is based on the different features accessible to the user. First, is the *unregistered users* who are unable to access any of the system features until registration; which brings us to the second category of user, the registered users which are able to access all system features apart from creating traffic reports; atleast, not until they allow the system to access their mobile phone camera and location services. Finally, fully activated users who have are able to fully exploit the *SafeStreets* features.

The first of the following state diagrams show a clear representation of the different user states and the user actions that cause a transition from one classification to another. As well as, a representation of the permission granting process that is a sort of a prerequisite for becoming an activated user. The second diagram describes the different states that the violation reports follow from being filled by the user till their communication to municipality to be used for ticketing.

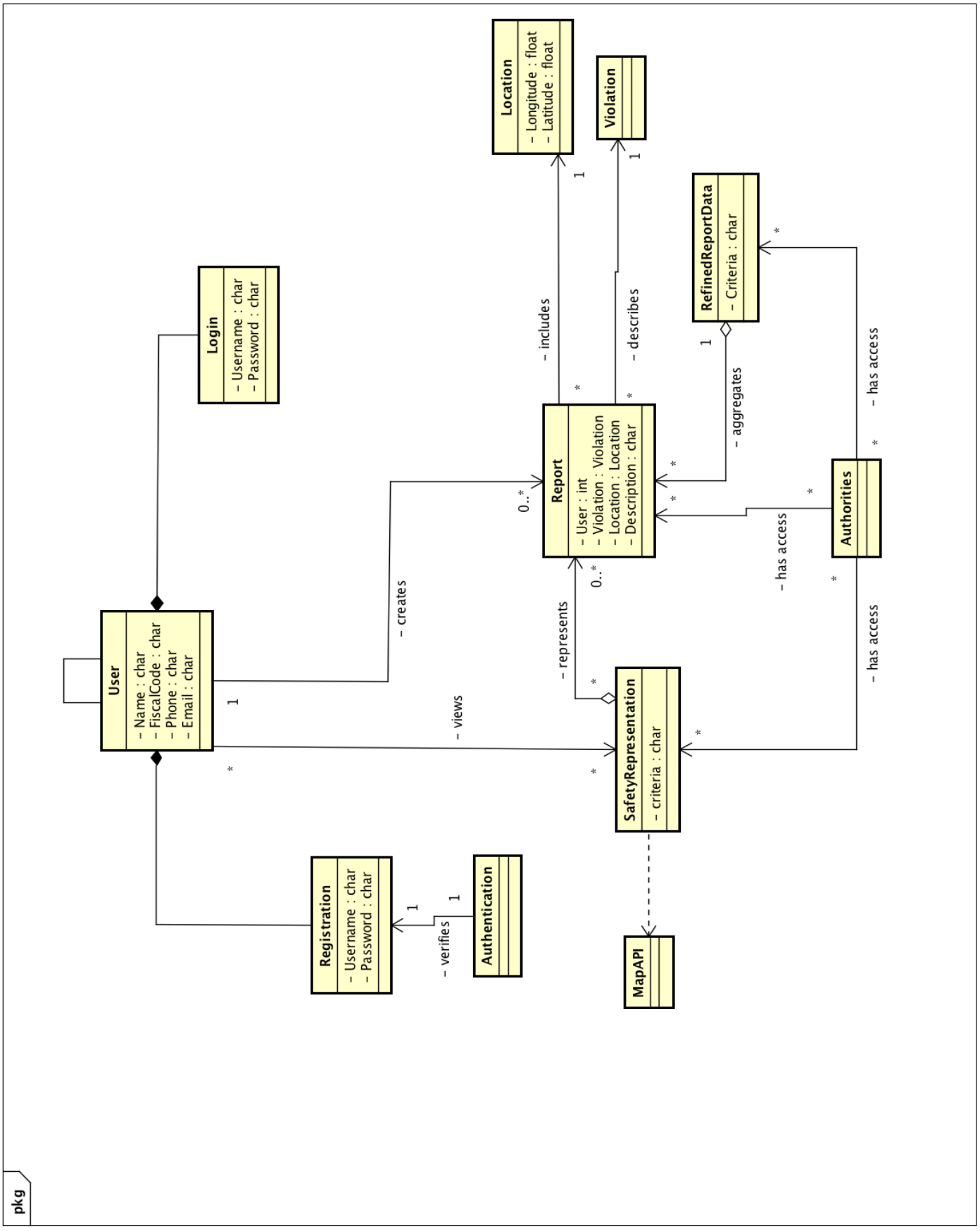


Figure 1: Entity Class Diagram

Figure 2: State Machine Diagram for User

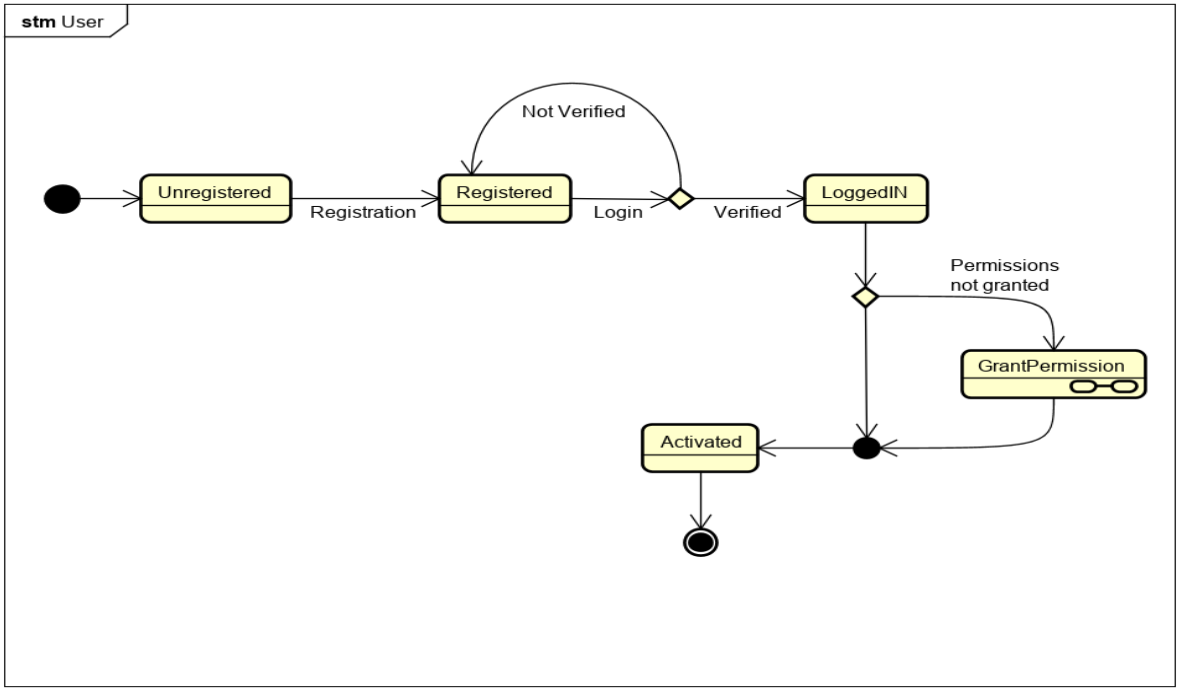


Figure 3: State Machine Diagram for Giving Permission

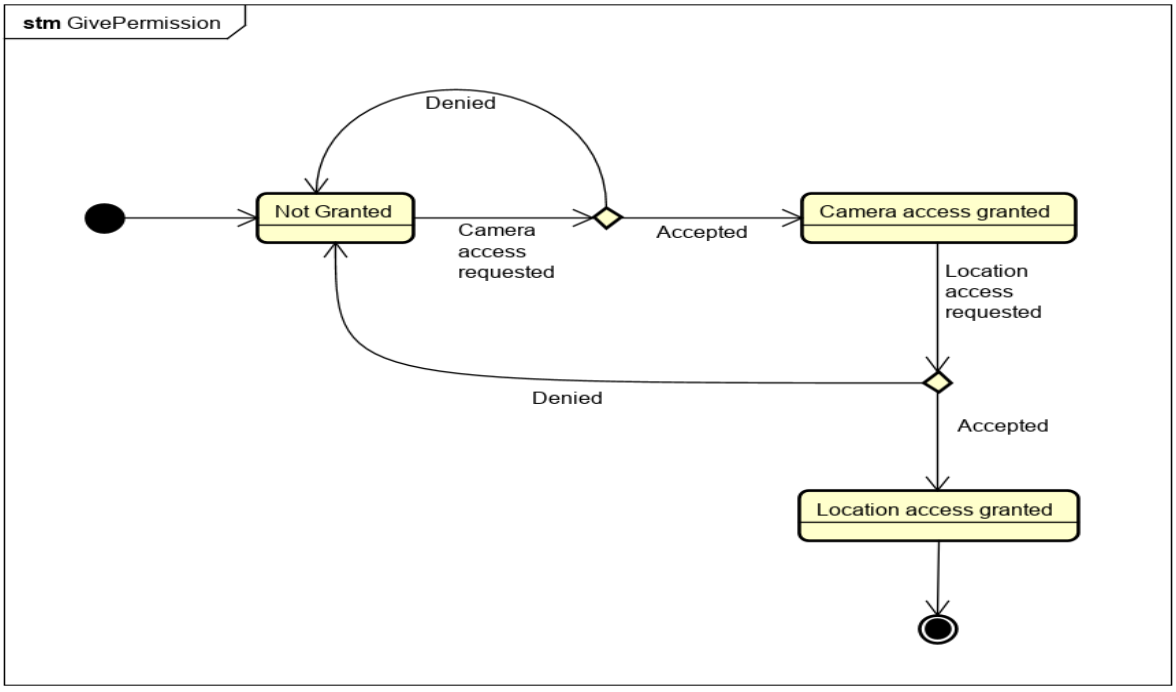
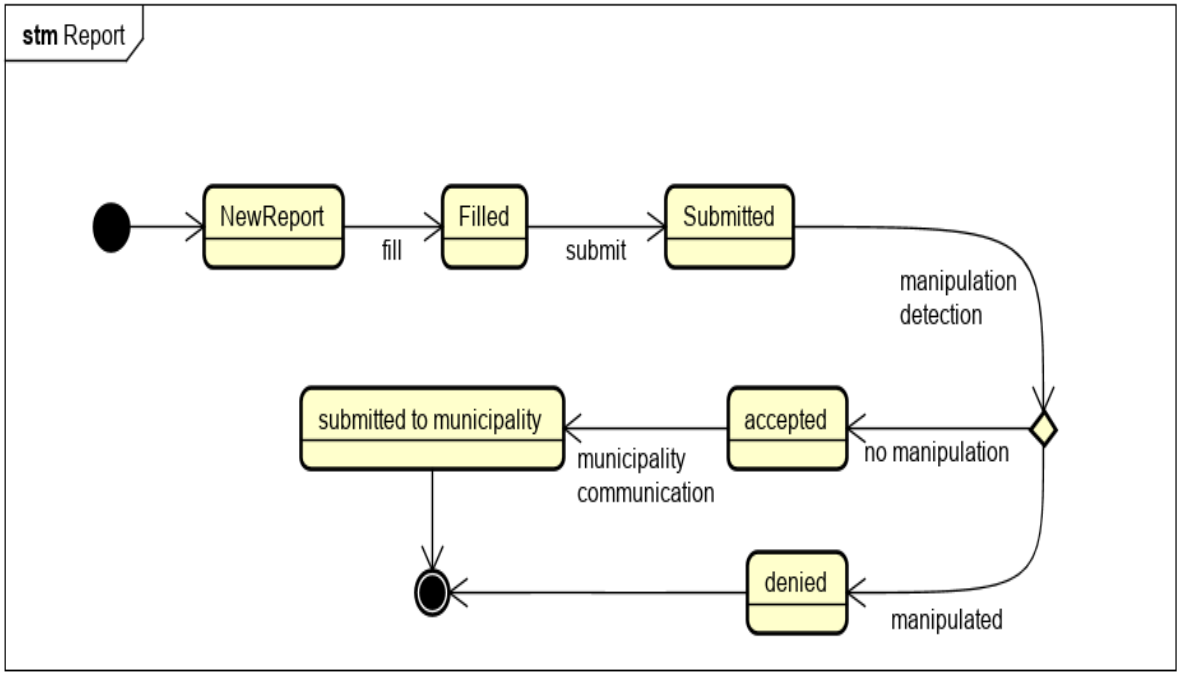


Figure 4: State Machine Diagram for Violation Report



2.2 Product functions

In the following subsection the various major functions of the *SafeStreets* system; which were briefly mentioned in the *Scope* section of the document. However, this section shall provide a more in depth description of said functions providing a broad yet clear overview of the system's functionality.

2.3 Violation Report

The *Violation Report* function is the motivating main function of the *SafeStreets* system. This function provided mainly to be used by the average or in other words civilian user entails the enabling of such users to formalize traffic violation reports and submit them to be later used by either the system internally or by an external benefactor; i.e., the municipality. The described functionality is achieved through a user interface; ideally, a mobile phone through which the accesses the *SafeStreets* interface and starts the process of reporting a violation. This process is briefly constituted by the imaging of the violation directly from the interface, the verification of the detected plate number and detected location and filling in some data regarding the details of the traffic violation, e.g., the violation type. Finally, the user submits to report which is saved and used by multiple other functions as will be discussed in the coming sections.

2.4 View Area Safety

This second function is also mainly used by civilians; although, the data provided to users in this section are also provided to the authorities although through a different interface as shall be seen later. The *View Area Safety* function is based on the previously mentioned *Violation Report* functionality. Which shall be clear as we describe the exact details of this function. In the *View Area Safety* function the user would specify a certain area in which he is interested in knowing the safety of. Then, the user is presented with a representation of the safety of the specified area. This is done formalized using the aggregated reports from various users of the system and graphically represented using the aid of map API.

2.5 Suggest Intervention

The following function along with the last function are provided to different class of users or to be more specific benefactors of the *SafeStreets* system. The described entity is the governing authorities in the different areas observed by the system through user reports. It is worth mentioning, that these two functionalities are achieved through a sort of mutual interaction between the *SafeStreets* system and the municipality. In which a municipality must first provide a means of communication or interaction, i.e., an interface in order for the system to provide these services. The *Suggest Intervention* service is the process by which the system suggests possible interventions to put into place by officials in certain areas; in order to decrease the number of violations which leads to accidents in such areas. To achieve this, the municipality must provide and interface to access the accident data in order to be coupled with the system's reports and also another interface to submit the suggestions.

2.6 Violation Report Communication

The last major function of the system is also one provided to the municipality. This service is simply the communication of the submitted traffic reports submitted by users while ensuring the secure transmission and ensuring that the data is not tampered with to a certain extent. This, can be done through the same interface which the system suggests possible interventions through. Moreover, the system could be provide more useful refined data to the municipality, such as, statistics on the different committed violations in various areas and most reported plate number.

2.7 User characteristics

The only actor of SafeStreet is the user, who can be explained in the following cases:

A user is a person who's aimed to facilitate authorities by reporting traffic violations in the city. The user registers to SafeStreet app, when any violation is observed, he/she takes pictures of the violating vehicle and along with the geographic coordinate sends it to the SafeStreet.

A user is someone who needs to obtain information about traffic safety in different areas of the city, the user can use SafeStreet in order to get this information from previously submitted reports, app displays information based on different areas of city using map API.

2.8 Assumptions, Dependencies and Constraints

In order to use SafeStreet, users are assumed to have a smartphone or tablet with capabilities of connecting to the internet, taking a picture and having a GPS sensor. More in details, the connection of internet via WiFi, 3G or 4G is needed to sent reports to the system, to recognize license plate and indicating occurred violations, users need to take pictures and send them to SafeStreet's server, finally to locate where the violation is occurred GPS sensor is used to get the coordinates of the violation.

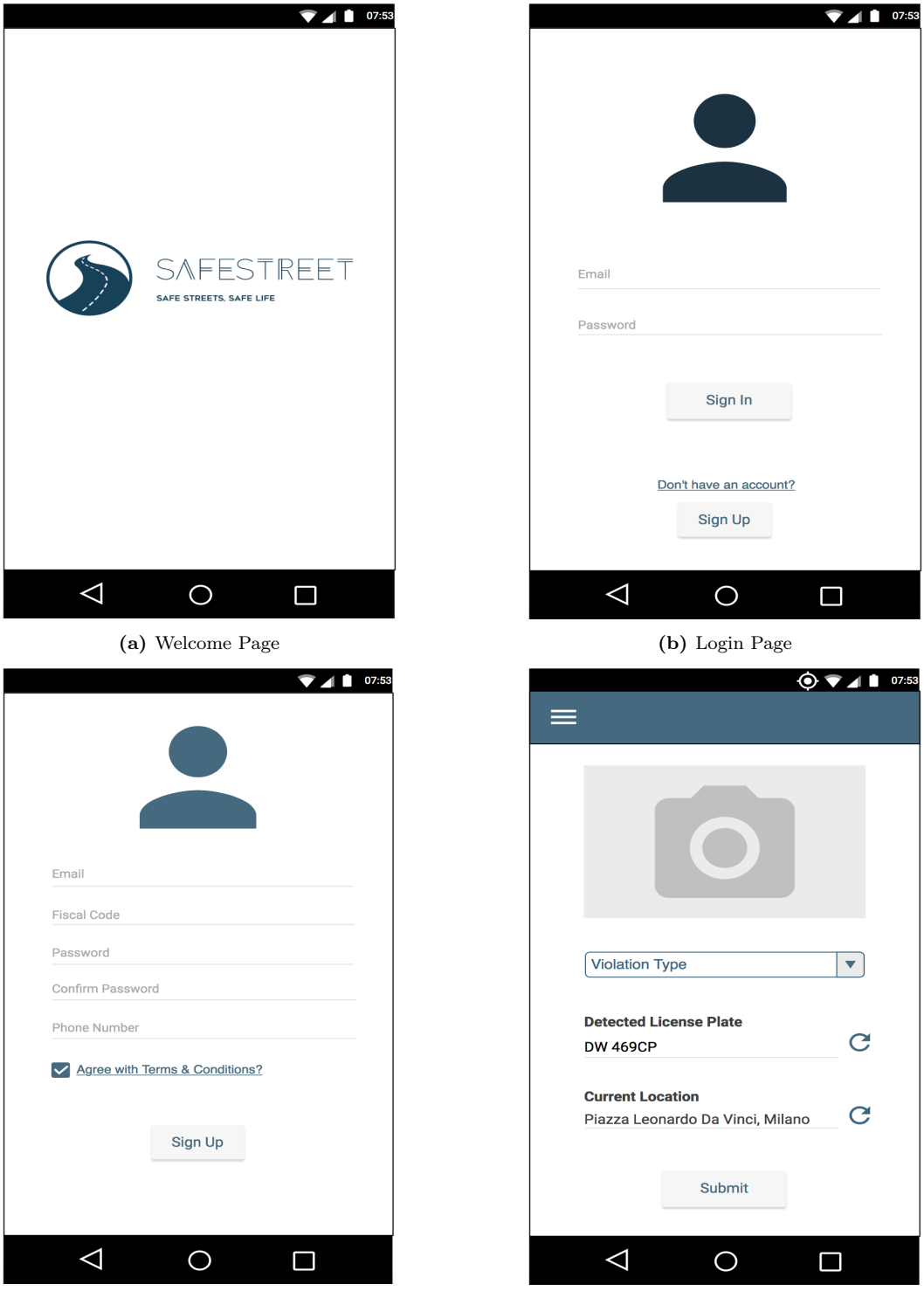
This system is aimed to provide service throughout Italy, the privacy of the users is significant for the system, Thereby it observes General Data Protection Regulation(GDPR) legislated by the European Union.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

In this section, Android mobile application mockups are represented which provide information about interfaces that the user will interact with. Figure 4 demonstrates feasible login, registration and report violation interfaces for the user, in the report violation page user can submit his/her report by taking pictures and specifying related information about the occurred violation. Taking closer look at figure 5, it can be understood that the user is able to use view safety functionality in order to get proper information about safety of different areas of city and also he/she has the possibility of modifying his/her profile information.



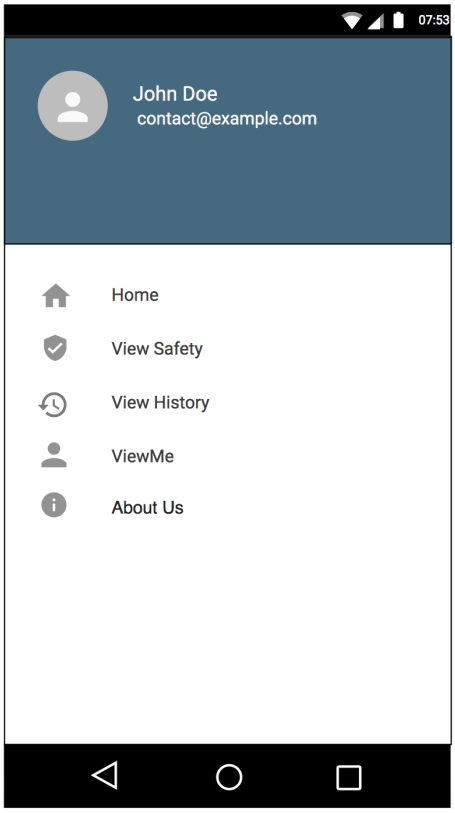
(a) Welcome Page

(b) Login Page

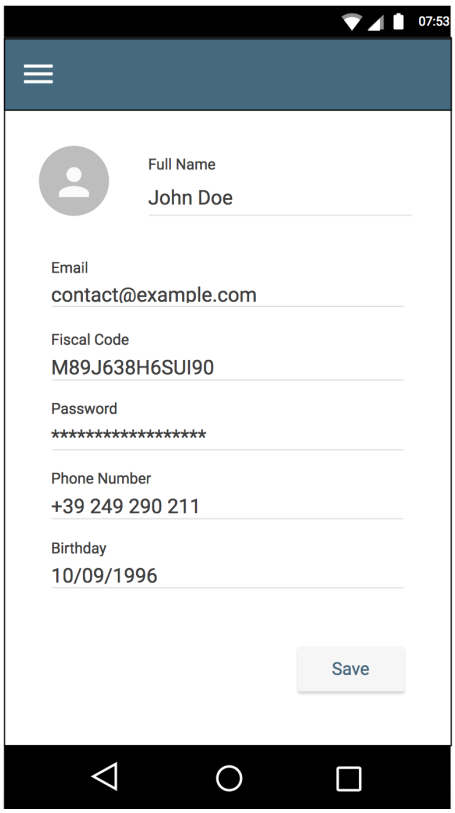
(c) Registration Page

(d) Violation Report Page

Figure 5: User Interface of the Android App



(a) Main Menu of the App



(b) ViewMe Page

(c) view Safety Page

Figure 6: User Interface of the Android App

3.1.2 Hardware Interfaces

The only external hardware interface worthy of mention in this section, is the user-owned mobile devices used in order to use the system features. Other than this, there are no other hardware interfaces neither provided by the system for external use, nor offered by an external entity and used by the system.

3.1.3 Software Interfaces

This system does not provide any software interfaces to external systems to access any of the system's various resources. It does however,offer user interfaces for the users to make use of the system features as previously stated. Moreover, some external interfaces are exploited by the system to retrieve and communicate data; which will be discussed in the communication interfaces section.

3.1.4 Communication Interfaces

As previously mentioned, the system uses the interfaces provided by the municipality to perform some of the main functions. The interfaces accessed by the system are as follows; firstly, an interface provided by the authorities for the retrieval of accident data. Said interface is used in order to couple the accident data with *SafeStreets* report data in order to decide upon appropriate interventions in order to decrease violation leading to accidents. To communicate the interventions along with other data to the authorities; specifically, details of aggregated reports and refined report data inferring some insights the system uses a second interface offered by the municipality for the submission of such data. Other than the municipality's interfaces all communication interfaces used by the system are internal interfaces used by the diverse system sub-modules in order to interact. Finally, it is noteworthy to mention that if the system is not provided the interfaces by the municipality it is impossible to communicate any of the previously mentioned data, since it is assumed that the system shall exploit the authority provided communication service to proactively provide this data.

3.2 Functional Requirements

3.2.1 Scenarios

Scenario 1

Lorenzo is a student who is living in Rome. During the summer, he decides to go on a trip to Milan. Due to the fact that Milan is a really crowded city and number of accidents and traffic violations have skyrocketed in the recent decade, he is skeptical to drive his personal car to enjoy the eye-catching scenery along the way or book a flight. His friend suggests to use the SafeStreets app to check the safety of the area he will be staying in Milan in order to decide whether it would be safe to leave his car behind and move on foot there. Therefore he goes on his phone app store and installs the app. Then he fills up all the required fields in the registration form. Then he stays for the verification of his account by SMS and Email. After Lorenzo's account is verified, he goes to login in the app, so he entered his credentials for login and logs into the app so he is redirected to the homepage. He gets a message from the app asking permission to access his mobile phone camera; he accepts. He then gets a second message to give permission to the app to access his current location; he accepts this as well. Then Lorenzo opens the View Safety section and searches for the area around his hotel.

Scenario 2

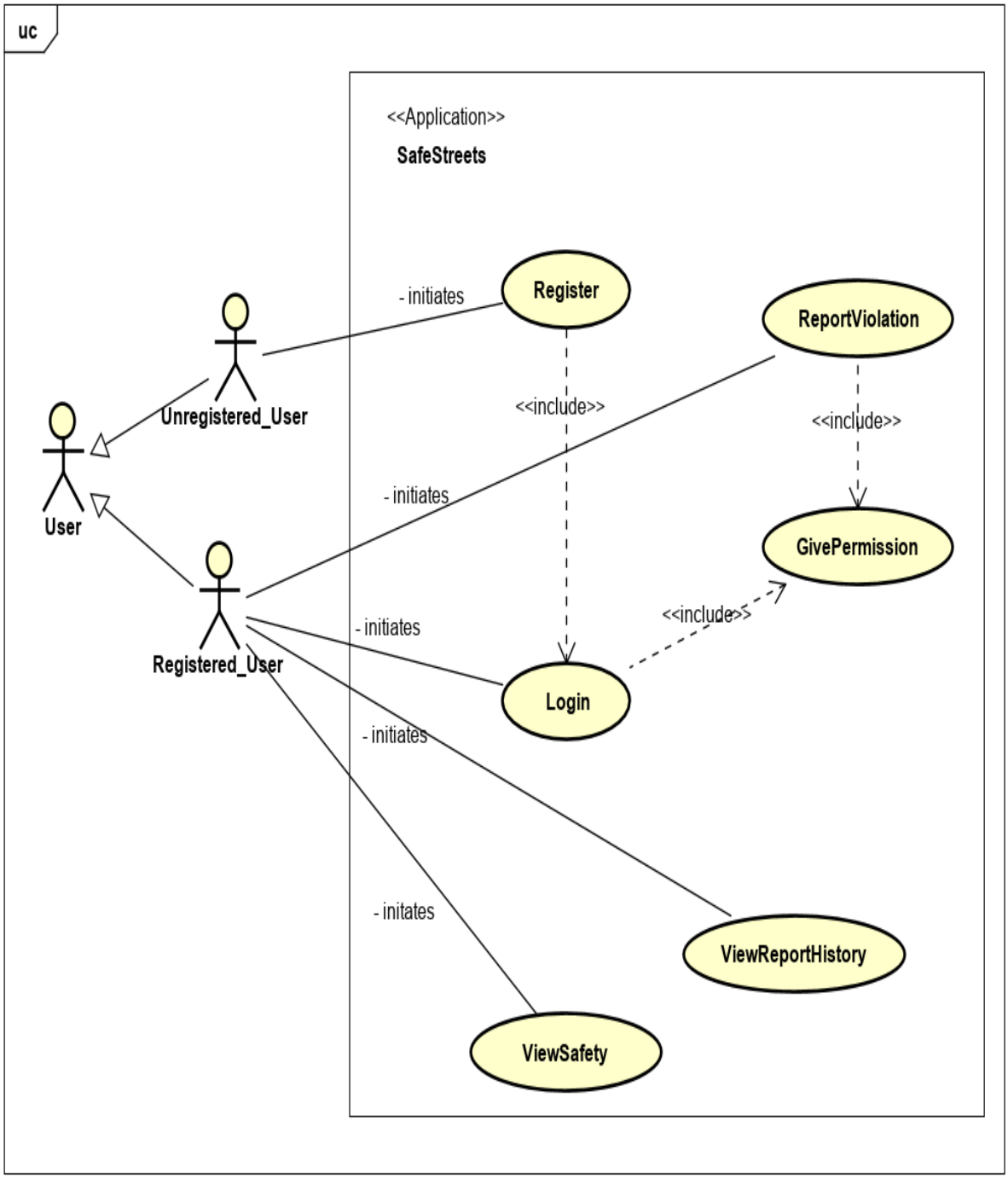
Paul is walking on the sidewalk near his home when he notices in the distance a car parked on the sidewalk obstructing his way. He decides to report this since it poses a threat to the safety of people walking by. Paul opens the *SafeStreets* app on his mobile phone to make the report. He takes a photo of the violating vehicle showing the license plate number, The plate number is detected and shown in text on the app. He indicates the type of violation . The location of the violation is detected using Paul's current location. After submitting the report, he gets the feeling that this is not the first time he reports this car for parking of the sidewalk near his house; therefore, he opens the Report History section of the app and searches for his reports from the last month to check if he was right.

3.2.2 UseCase Diagram

The following usecase diagram show the different usecases to be described in the following subsections and the relations and interactions among them and with the system users. The users, shown as actors initiate the different usecases some of which initiate other usecases such as the register usecase which initiates the login.

Use Case Diagram

Figure 7: Use Case Diagram



3.2.2.1 Registration

3.2.3 General Functions

Use Case

Name	Register and Verify account
Actor	Unregistered User
Entry conditions	User installed the App successfully on the proper device (which has GPS and Camera, also appropriate OS) with access to the Internet and enough storage space.
Events flow	1. User opens the App and sees the welcome pages 2. by skipping the welcome pages, the user will see the login and registration page then he/she could move to the registration part 3. In the registration part, the user will fill up the mandatory fields and if the user wants could also fill optional fields; next by checking the box related to "terms of service" the data will be sent to the system. 4. The user should wait for the SMS from the system which contains a code in it and enters the code; in the next section, she/he should verify the email address used for registration. Eventually, the registration process will be finished after email verification.
Exit conditions	The User will see the successful registration page and will be driven to the Login page.
Exceptions	• Not enough storage space: showing a warning to the user which there is not enough space to could cache needed files. • No Internet connection: Showing an Error message which says to continue the process the Internet connection is necessary. • Duplicate username: Showing an Error message to the user to change the username or if he/sheforgot the password it could be recovered by linking to the related part of the App. • Improper username or ignoring the constraints in the form:showing an Error message to the user before he/she could check the "terms of service" box. • Not receiving the SMS of verification: showing Warning after the timer of this section finished which tells the user to continue the process the new message should be requested. If the number of requests becomes more than 10 then display an Error message to the user to begin the registration process from the beginning and drivethe user to the Welcome pages. • Not receiving the email of verfication: display warning which email is not verified and if there is no email in inbox or spam box, request for the new verification email. If the number of requests becomes more than 15 then display an Error message to the user to begin the registration process from the beginning and drive the user to the Welcome pages. • Not checking the "terms of service" box: display Warning which accepting the terms of service is vital for registration. • Any problem from the system: display an Error and apologies from the user and ask for another try in future • Any software bug in the application: display an Error message and ask from the user to send the log of the crash to the system and close the App

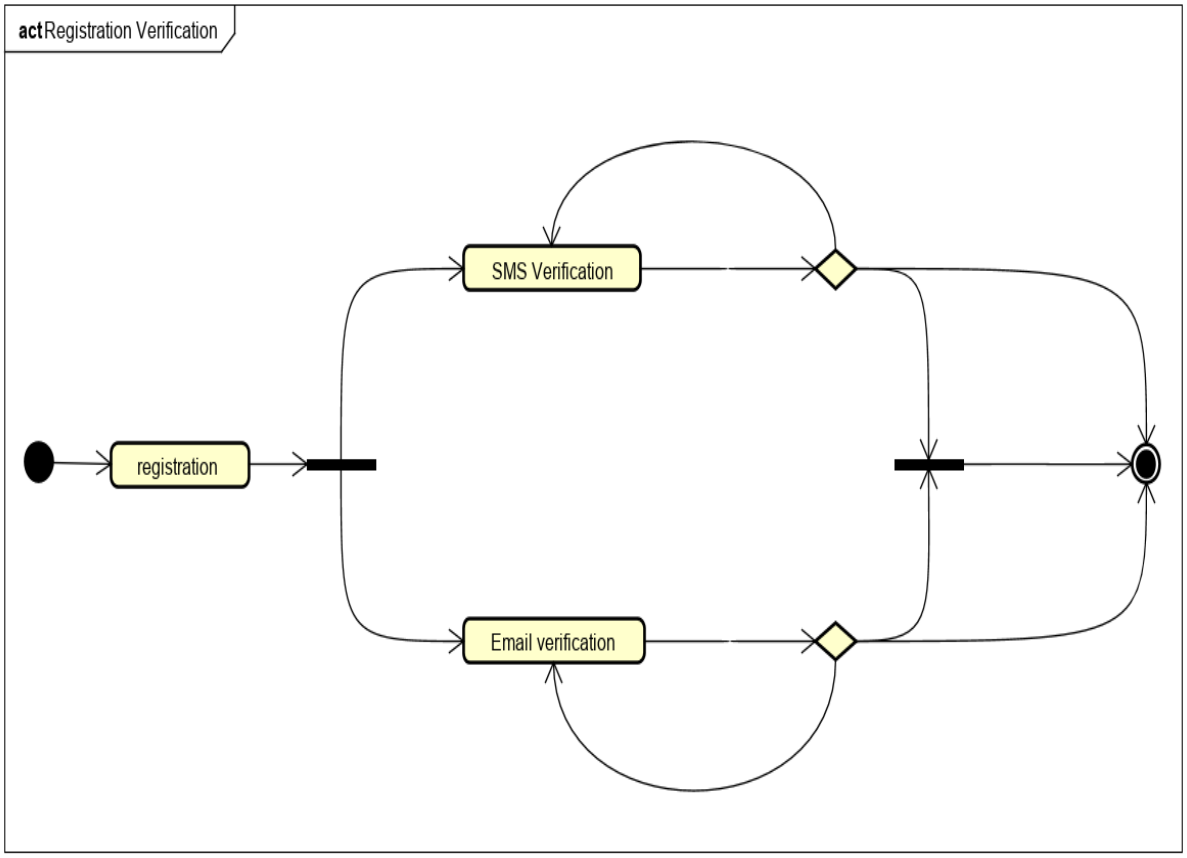
Table 3: Registeration and Verifiction of account usecase

The following diagrams, describe the registration process in terms of activity and runtime diagrams. The process is constituted by the filling and submission of the registration form, after which, the user receives

email and SMS verification. If confirmed, the registration process is complete.

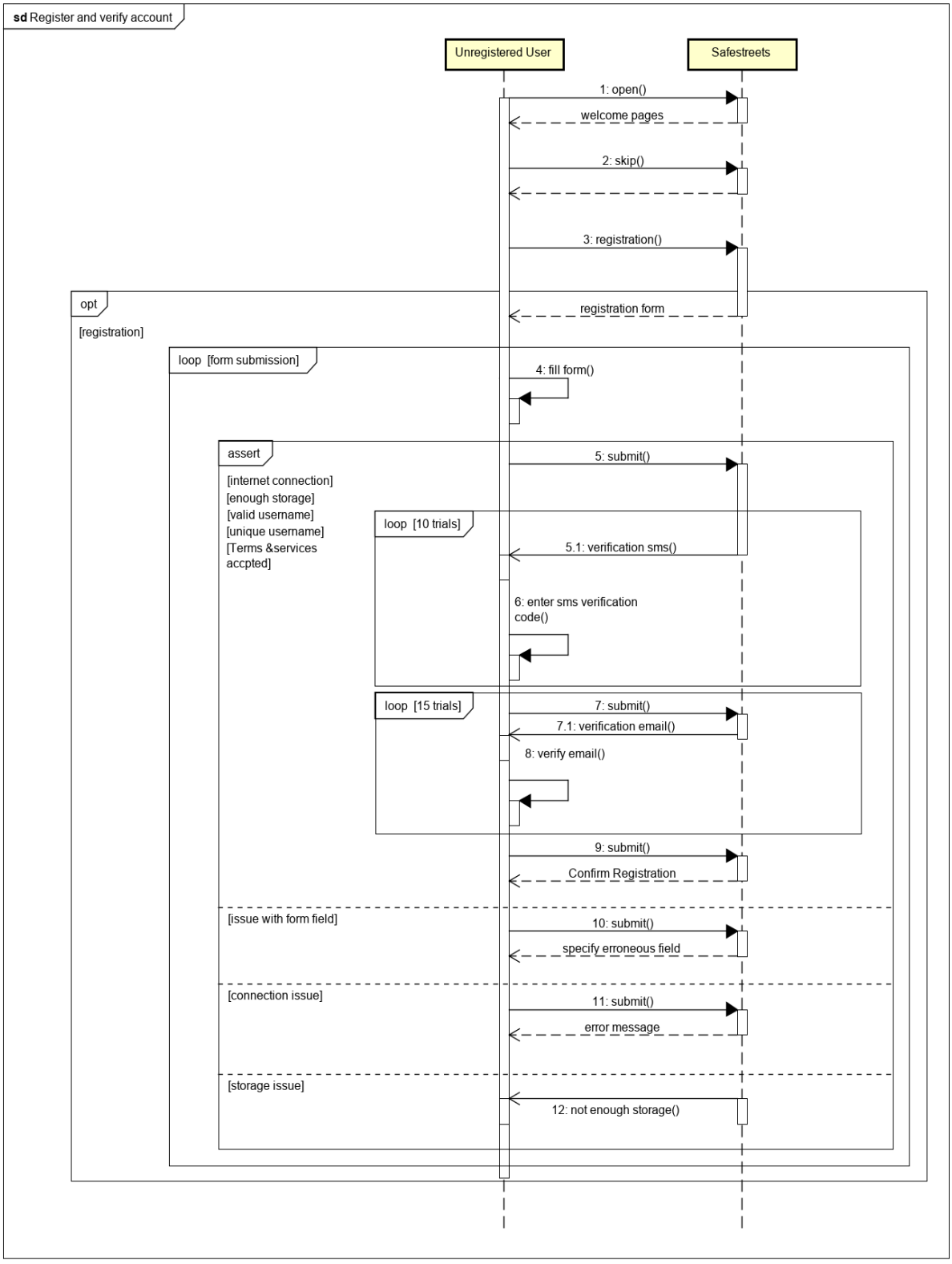
Activity Diagram

Figure 8: Activity Diagram for Registration and Verification



Sequence Diagram

Figure 9: Sequence Diagram for Registration and Verification



3.2.3.1 Login

Use Case

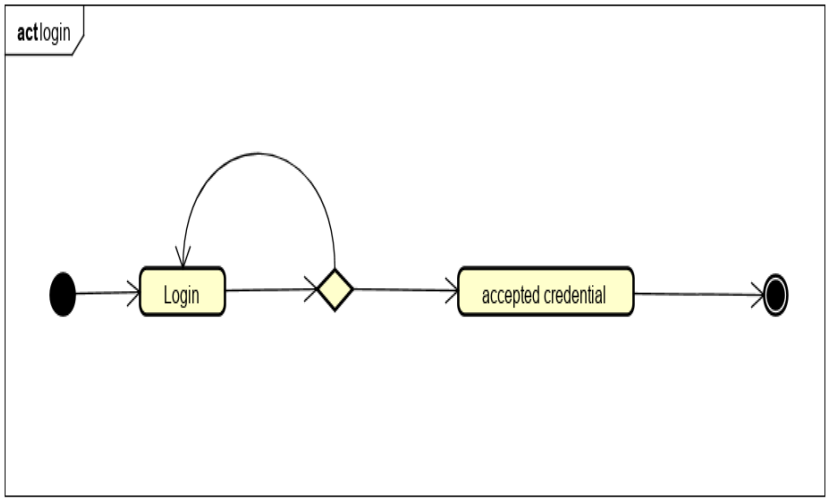
Name	Login
Actor	Registered User
Entry conditions	The User should have a verified account in the system and installed App in a proper device (which has GPS and Camera, also appropriate OS) with access to the Internet and enough storage space.
Events flow	The user will open the App and enters the credentials for login and if she/he wants to speed up the reporting violation and not logging in to the app every time, the user will check the "keep me logged in" box. If the credentials are valid the user will be driven to the reporting violation page.
Exit conditions	The user successfully login to the application.
Exceptions	• Absence of user name: display a warning page in which the entered username does not exist and gives an option to the user to be driven to the registration section. • Incorrect Credential: show an Error message to the user in which the entered credential is wrong. • Not enough storage space: showing a warning to the user which there is not enough space to could cache needed files. • No Internet connection: Showing an Error message which says to continue the process the Internet connection is necessary. • Any problem from the system: display an Error and apologies from the user and ask for another try in future • Any software bug in the application: display an Error message and ask from the user to send the log of the crash to the system and close the app

Table 4: Usecase for Login

The login process is shown in the following figures. The user starts by submitting their login credentials. The system then checks them, and if verified the user is redirected to the homepage of the application. If however, there are issues with the credentials the user is notified and allowed to try again with a maximum of 20 trials.

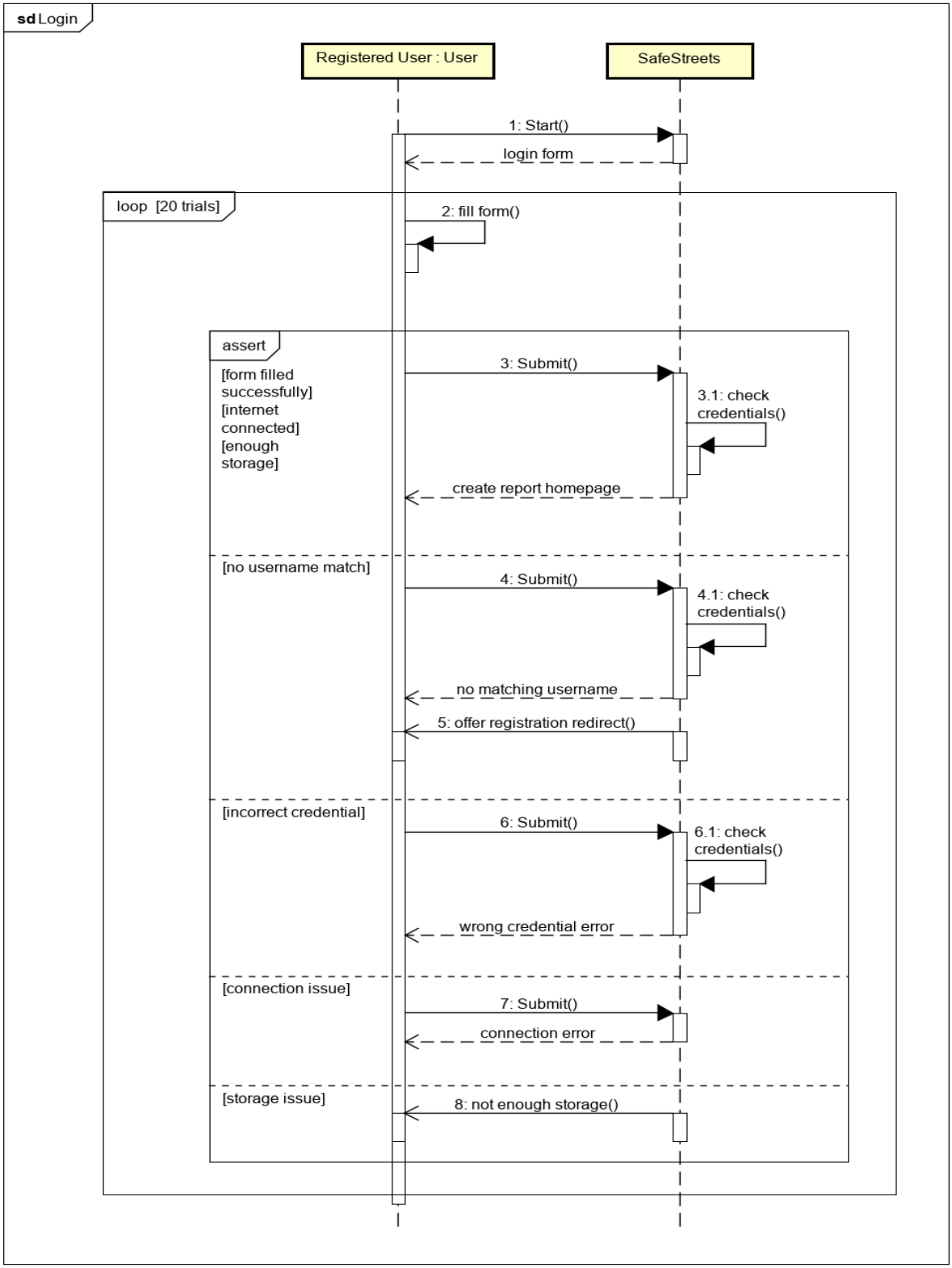
Activity Diagram

Figure 10: Activity Diagram for Login



Sequence Diagram

Figure 11: Sequence Diagram for Login



3.2.3.2 Giving Permission

Use Case

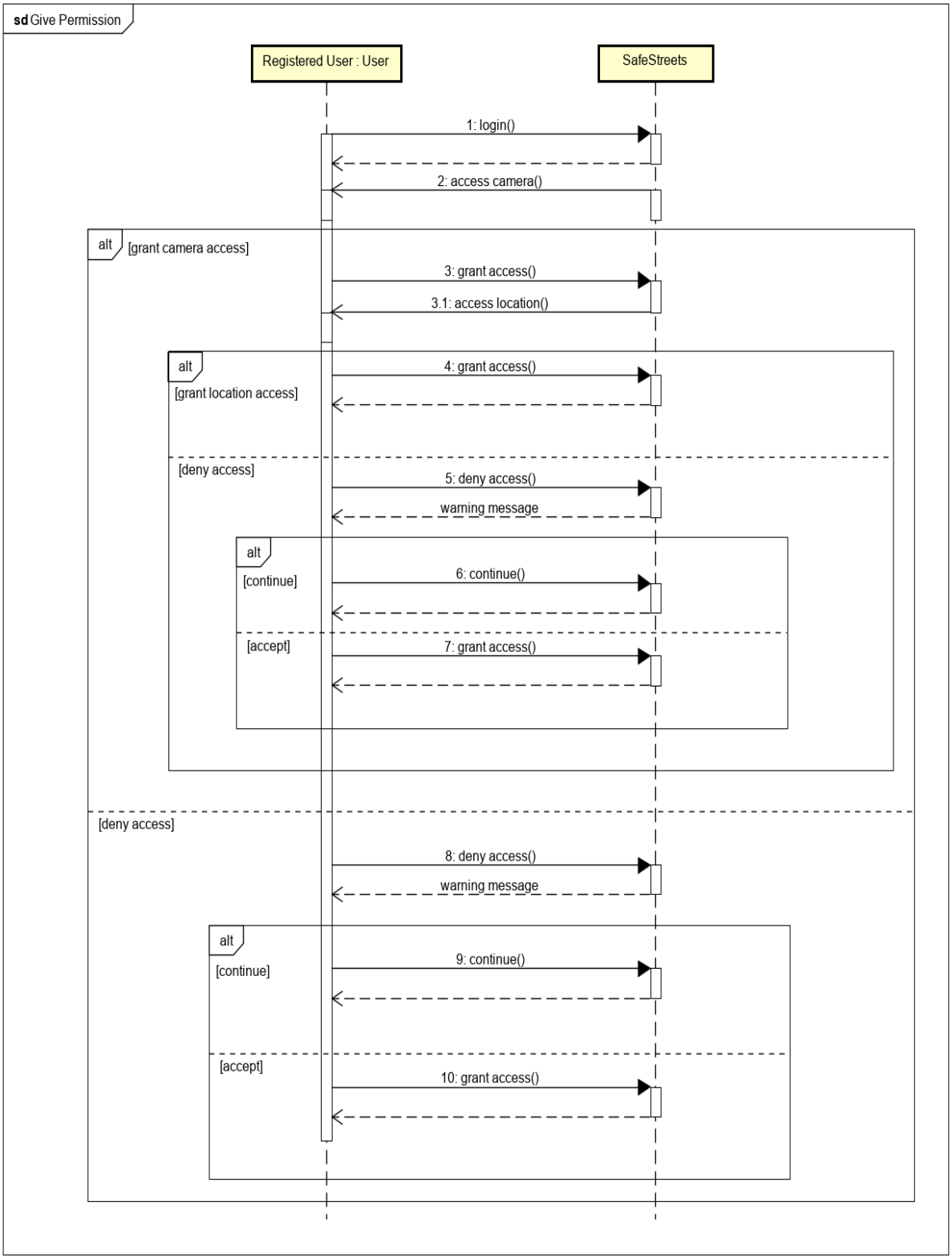
Name	Giving Permission
Actors	Registered User
Entry Condition	The user logged in to the app for the first time
Event Flow	1. After logging a dialog message asking for access to phone camera appears 2. The user clicks on the 'accept' button 3. Dialog message asking for access to location data appears 4. The user clicks on the 'accept' button
Exit Condition	The user grants access to both camera and location data
Exceptions	• The user denies access to either camera or location data: A message appears stating that if access is not granted no trafficviolation reports can be made and asking the user if they want to continue anyway If the user continues, they will not be able to make violation reports If the user grants access, all functions of the system are available to be exploited by the user

Table 5: Usecase for Giving Permission

After first login, the user is asked to give permission for the application to access the device camera and location services. If the user denies, the they are shown a warning message that they will be unable to make report submission until they accept access.

Sequence Diagram

Figure 12: Sequence Diagram for Giving Permission



3.2.3.3 View Report History

Use Case

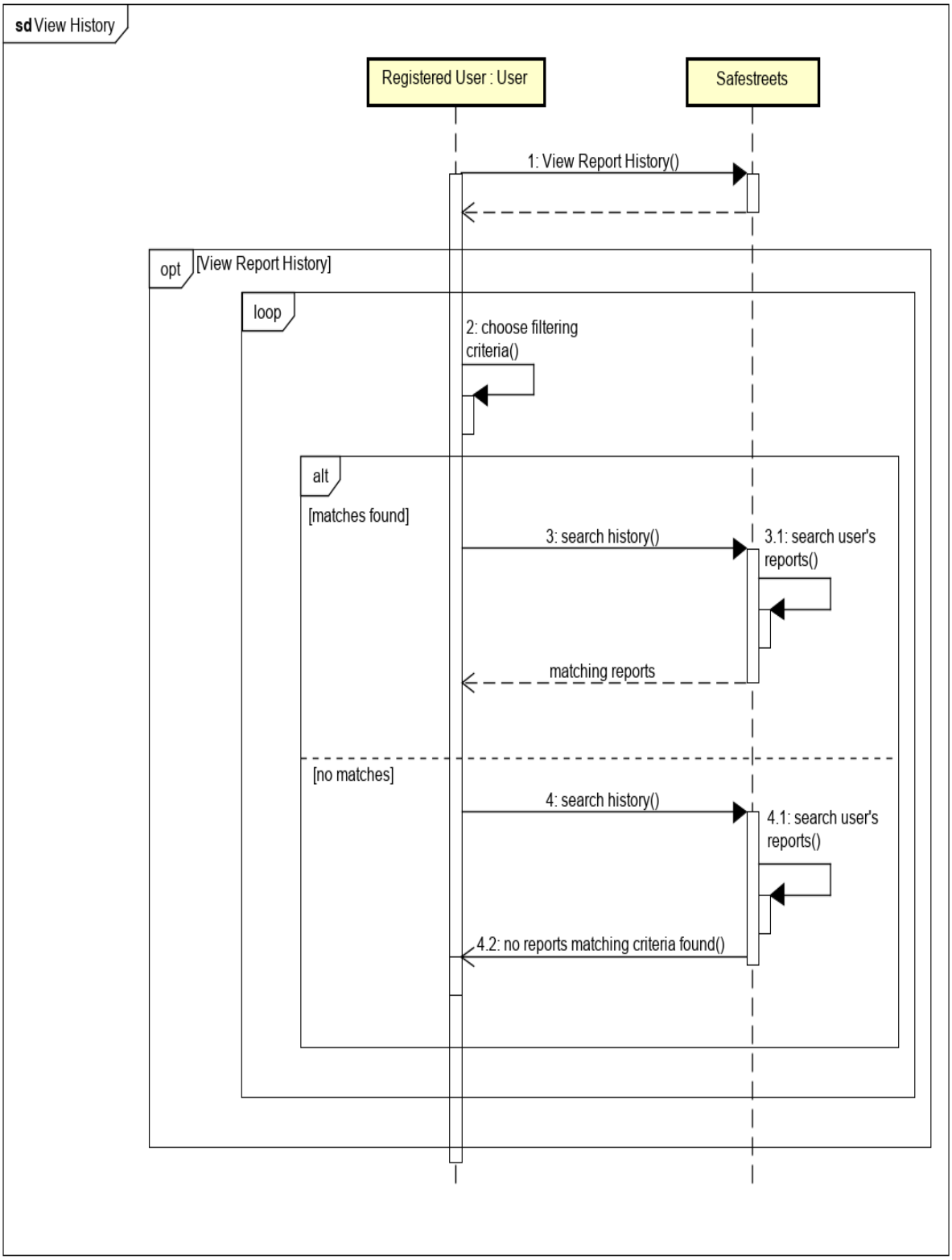
Name	View Report History
Actor	Registered User
Entry conditions	Signed in user wishes to see previously submitted report
Events flow	1. From the homepage the user opens the view history section 2. The user chooses from a group of filters to apply to history 3. The user is presented with a list of previously submitted reports that match filters
Exit conditions	The user is presented with report history
Exceptions	• No reports match applied filters: A message is presented stating that there are no matching reports in history

Table 6: Usecase for View Report History

The last of the general functions represented in the following figure, is the Report History functionality. Where, the user chooses the filtering criteria and they are presented with the violation reports previously submitted by them that match the search criteria.

Sequence Diagram

Figure 13: Sequence Diagram for View Report History



3.2.4 Violation Report

Use Case

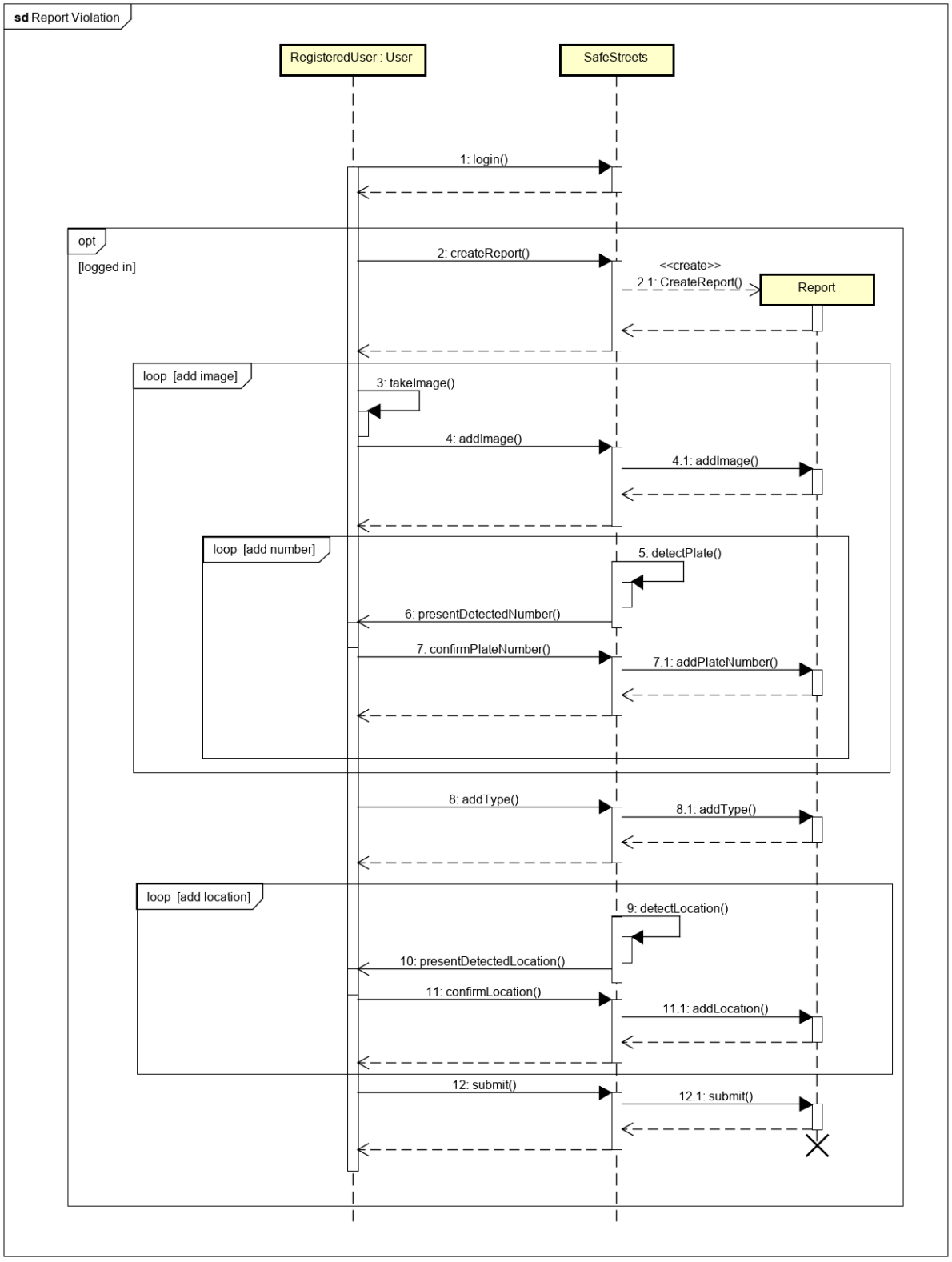
Name	ReportViolation
Actors	Registered User
Entry Condition	The user witnessed a traffic violation which he wants to report
Event Flow	1. In the homepage, the user takes a photo of the violating vehicle 2. The user verifies that the image is clear and the license number is visible 3. The user clicks on the "Continue" button to go on with the processes 4. The user is presented with the page for completing the details of the report 5. The detected plate number is shown 6. The user chooses the type of the violation he is reporting from the available choices 7. The detected user location is shown 8. The user verifies that all the data in the report are right 9. The user clicks the "submit" button
Exit Condition	The user report is submitted to the server and stored in user report history
Exceptions	• The user has not logged in before: The user is redirected to the "GivePermission" functionality • The user has denied access in their first login: The user redirected to the "GivePermission" functionality • The image is not clear or the license plate number is not visible in it: The user clicks the "take another" button to take another photo • The type of violation is not in the available choices: The user chooses the "other" option and specifies the type of violation by typing it in • The detected data fields (plate number and location) is not accurate: The user clicks on the "redetect" button associated to that field The erroneous field is redetected
Special Req	• {NFR_1} The detection of plate number must be done in less than 1 second • {NFR_2} The detected plate number is accurate 99% of the time

Table 7: Usecase for Report Violation

The sequence of steps the the user performs in order to complete the violation report process are represented in the following diagram. The user starts by taking images of the report from which the plate number is detected. After which, the type of the violation selected and the location is detected using the user's location. Finally, the report is submitted.

Sequence Diagram

Figure 14: Sequence Diagram for Report Violation



3.2.5 View Safety

Use Case

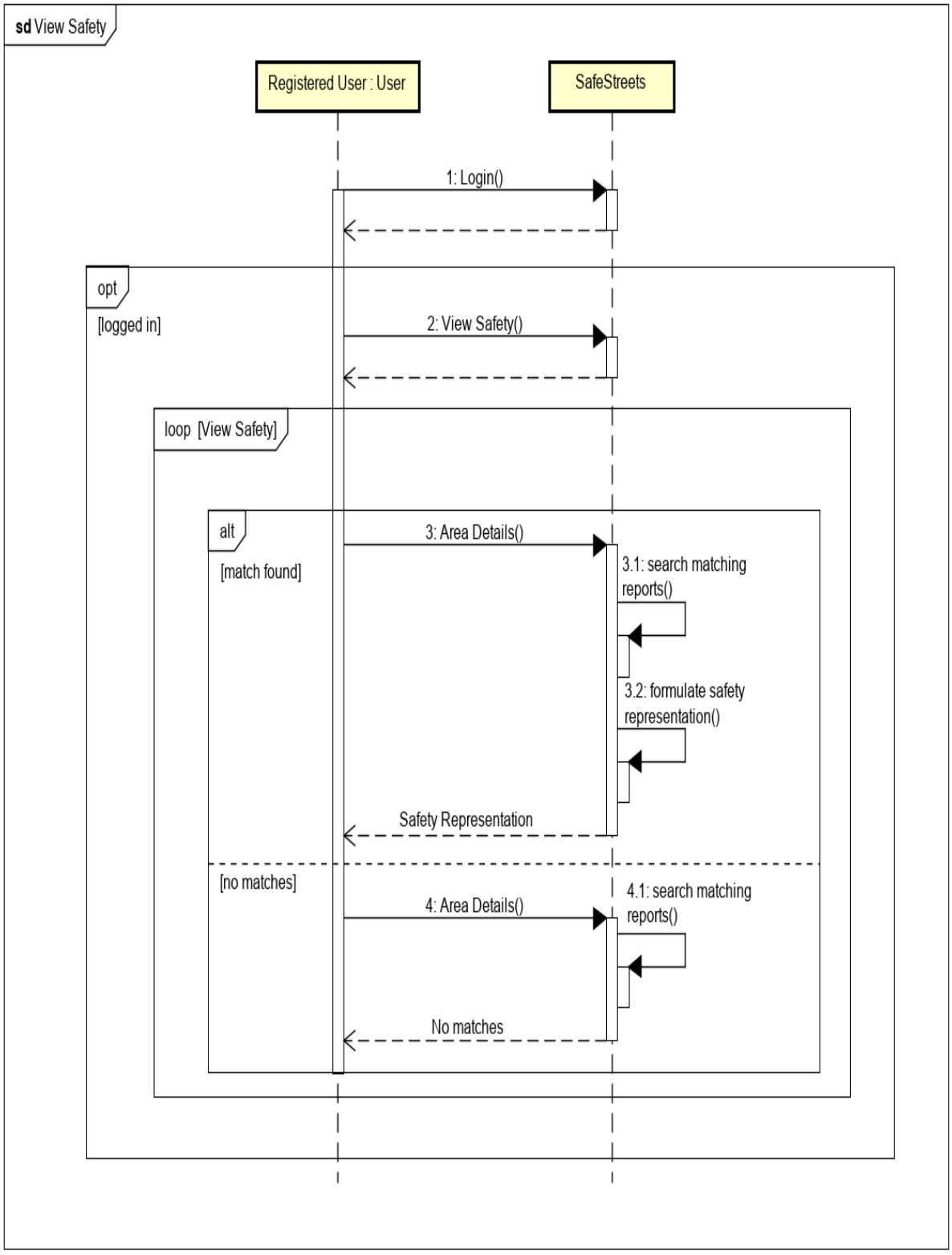
Name	View Safety
Actor	Registered User
Entry conditions	The user wants to check the safety of a certain area that he is interested in
Events flow	1. In the homepage the user opens the view safety section of the app 2. The user specifies details of the area he is interested in 3. The user submits search criteria and chooses area from a list of matches 4. The user is represented with a geographical representation of the area's safety based on number of violations and accidents occurring in the area
Exit conditions	The user is represented with area safety
Exceptions	• The specified area does not yield any search results: A message stating there are no results matching entered data is shown

Table 8: Usecase for View Safety

The following figure describes the view safety process. In which, the user is able to see a representation of the safety of various areas based on their search. The user enters the area details and if there are any reports in that area a geographical representation of the area is shown.

Sequence Diagram

Figure 15: Sequence Diagram for View Safety



3.2.6 Requirements

In the following section the functional requirements of the system will be discussed. The requirements shall be grouped together by the various main functions of the system that they are related to.

General User Functions:

- [R-1] Users should be allowed to register to services provided by the system
- [R-2] Users should provide unique identification to the such as fiscal code during registration
- [R-3] Registered users should be allowed to login
- [R-4] Each registered user should have a unique username used for logging in chosen at registration time

User Violation Reporting:

- [R-5] System should enable registered users to report traffic violations
- [R-6] When reporting a violation, users should be able to take an image of the violating vehicle's license plate
- [R-7] When reporting a violation, users should be able to fill in the details of the reported violation such as the type of the violation
- [R-8] The system should be able to detect the current user location when reporting a violation
- [R-9] The system should extract the plate numbers from the image taken by the user

User View Area Safety:

- [R-10] The received reports must be stored by the system to be used by other services
- [R-11] Registered users should be allowed to view a representation of the safety of selected areas possibly with the help of a map API
- [R-12] The system should implement a means to measure the safety of various areas based on reported violations in said areas
- [R-13] Incoming reports should be integrated and used to update the safety of areas
- [R-14] If accident reports are provided by authorities the system should take that data into account when calculating the safety of a certain area

User Report History:

- [R-15] The system should present on demand to the users a record of all the reports previously submitted by them

Municipality Interaction:

- [R-16] The system should keep records regarding the submission dates of violations to the municipality
- [R-17] The system should aggregate the data regarding reported traffic violations since the last submission to the municipality
- [R-18] The aggregated traffic violation data should be converted to a form acceptable by the municipality interface
- [R-19] The system should periodically submit the new traffic violation data to the municipality interface

- [R-20] The system should be able to store submitted reports coming from users with proper metadata
- [R-21] The system should be able to export reported violations in form of specific file
- [R-22] The system should be able to filter data based on desired requests from authorities
- [R-23] The logging functionality has to be implemented in the system
- [R-24] The system should extract insights from the users' traffic violation reports such as the most frequent types of violations in certain areas
- [R-25] The system should be able to decide on appropriate interventions to minimize frequent traffic violations in the various areas
- [R-26] The system should formalize the interventions to be suggested in a form acceptable by the municipality interface
- [R-27] The system should submit the interventions regarding the areas with high frequency of violations to the municipality interface
- [R-28] The system has to be able to mine reports to find insights based on types of violation and different areas
- [R-29] The system should analyze the data from the reports to produce statistics such as the most frequent plate numbers that commit violations
- [R-30] The system should formalize the refined data and submit them to the municipality interface periodically

3.2.7 Traceability matrix

ROW ID	GOAL ID	REQ ID	DOM ID	UC ID	COMMENTS
1	[G-1]	[R-1]-[R-9]	[D1]-[D7]	Report Violation	
2	[G-2]	[R-10]-[R-14]	[D2],[D9]	View Safety	
3	[G-3]	[R-16]-[R-23]	[D6]-[D8]		Since authorities are not direct users of our system no use cases are provided for goals related to interaction
4	[G-4]	[R-24]-[R-27]	[D5],[D6],[D8]		
5	[G-5]	[R-28]-[R-30]	[D3]-[D8]		
6	[G-6]	[R-15]	[D4]	View Report History	

Table 9: Traceability matrix

3.3 Performance Requirements

- {NFR_1} The detection of plate number must be done in less than 1 second
- {NFR_2} The detected plate number is accurate 99% of the time

3.4 Design Constraints

3.4.1 Standards compliance

There are no standards constraints regarding the *S2B* design and implementation, apart from, the fact that the user interface must be implemented as an Android app in order to achieve the mobility required to report violations dynamically. Therefore, the decisions of implementation tools and languages used shall be left to the implementation team to apply the most effective methods given required functionalities.

3.4.2 Hardware limitations

As previously mentioned, some hardware constraints apply to the user owned devices that shall run the *SafeStreets* app. The following constraints apply in order to achieve full access to system features. Firstly, all devices must have a fully functioning camera capable of capturing images of previously stated quality; i.e., 3 megapixels. Moreover, the GPS system of such device must be functioning in order to locate users at time of violation report creation. Lastly, the devices must have enough free storage space to install the *Safestreets* app.

3.4.3 Software limitations

In order for the end users to be able to install the *SafeStreets* mobile app, their smartphones need to have **atleast** Android OS version 5 and a valid *GSF ID* key.

3.4.4 Any other constraint

The system must conform to general standard constraints such as not violating *Terms of Service* agreement provided to the user. Particularly, the constraints regarding users' sensitive data when communicating collected data to the authorities. Moreover, as previously discussed in details the users of the app must give their consent in order for the system to have access to the cameras and location when creating a report.

3.5 Software System Attributes

3.5.1 Reliability

For the system in hand, *reliability* could be viewed as a matter of providing an acceptable quality of service level. This could be achieved through numerous means, among which are duplication of the most critical system modules, using predictive maintenance to decrease the probability of failure and the consistent monitoring of the system, to name but a few.

3.5.2 Availability

As the case was for the *reliability* aspect of the system; *availability* is more of a general requirement to provide an acceptable quality of system; rather than, it being a critical requirement of the system. Therefore, it is sufficient for the system to have *3-nines availability*; i.e., be available *99.9%* of the time. This would provide balance between providing an acceptable quality of service and the increased system complexity needed to have higher *availability*.

3.5.3 Maintainability

A system constituted of multiple smaller subsystems, such as, the *SafeStreets* system should developed with the aspect of *maintainability* in mind. A modular approach for the different services and the interaction with external interfaces would be the most suitable approach for a system with such characteristics. Employing such an approach would make the process of diagnosing failures and fixing them much less complex. Therefore, increasing the measure of *maintainability* of the system.

3.5.4 Security

Unlike some of the previously mention system attributes, *security* is an extremely application critical aspect for this system. In addition, to the general security needs which most systems should take into consideration; in example, regarding the user personal data storage. It is imperative, to also take into consideration data security during transmission and during generation; i.e., the creation of the violation reports. This need stems from the fact that the generated and later communicated reports should be as reliable as possible. Since, they may be used by authorities to give out tickets. To achieve this goal, the encryption of data as well as the application of constraints on the report creation report is a must.

4 Formal Analysis Using Alloy

In this section, the formal analysis and modeling of the system that was performed using Alloy shall be represented. The analysis has been performed with a focus on the various data flows of the system. Since, the core abstract goal of the system is the aggregation and communication of data to different parties; such as the communication of accident reports to authorities, or the communication of insights such as area safety and other statistics to both users and authorities; the data flows were our main concern in this formal analysis. As such, the bidirectional data flow between the system and the authorities has been modeled, as well as, the generation of data by the users in the form of violation reports all the while ensuring the consistency constraints; such as the fact that no report may be created by more than one user. Moreover, this analysis models other critical aspects ensuring the valid operation of the system such as the registration process and the fact that no two users may have a duplicate username or fiscal code or that all users have performed a registration.

```
---------------- Register Sig---------------------
sig FiscalCode{}
sig Username{}
sig Password{}
sig PhoneNumber{}
sig Name{}
sig Surename{}
sig Birthday{}
sig Gender{}

sig Registration{
        fiscalCode: one FiscalCode,
        username: one Username,
        password: one Password,
        phoneNumber: one PhoneNumber
        }
-----------------------------------------
sig Authority{}
---------------- Report Sig----------------------
sig Picture{}
sig PlateNumber{}
sig Time{
timestamp: one Int
        }
sig TypeOfViolation{} -- for simplicity we do not consider string in it
sig Location{
latitude: one Int,
longitude: one Int
        }{latitude >= -3 and latitude <= 3 and longitude >= -6 and
           longitude <= 6 }

sig Report{
        user: one User,
        picture: one Picture,
        plateNumber: one PlateNumber,
        time: one Time,
        typeOfViolation: one TypeOfViolation,
        location: one Location
        }
-----------------------------------------
---------------- Data-Channel  Sig----------------------
```

```
abstract sig Data{}
abstract sig DataFlow{}
abstract sig Channel{}

sig Statistics extends Data{}
sig Interventions extends Data{}
sig AccidentReport extends Data{}

sig AuthorityToSystem extends DataFlow{}
sig SystemToAuthority extends DataFlow{}


sig ChannelOne extends Channel{
        dataflow: one AuthorityToSystem,
        data: some AccidentReport,
        owner: one Authority
        }

sig ChannelTwo extends Channel{
        dataflow: one SystemToAuthority,
        data1: some Report,
        data2: some Statistics,
        data3: some Interventions,
        owner: one Authority
        }
--------------- Request  Sig--------------------
abstract sig Request{}

sig RequestForProfile extends Request{
        user: one User
        }

sig RequestForSafeArea extends Request{
        location: one Location
        }
-----------------------------------------
abstract sig Customer{
  registration: one Registration
        }

sig User extends Customer{
        name: one Name,
        surename: one Surename,
        birthday: lone Birthday,
        gender: lone Gender
        }


 -----------------------------------------
--------------- User Facts --------------------

--All Name have to be associated to a User
fact NameUserConnection{
        all n: Name | some u: User | n in u.name
}
```

```
--All Surename have to be associated to a User
fact SurenameUserConnection{
        all s: Surename | some u: User | s in u.surename
}
--All Birthday have to be associated to a User
fact BirthdayUserConnection{
        all b: Birthday | some u: User | b in u.birthday
}
--All Gender have to be associated to a User
fact GenderUserConnection{
        all g: Gender | some u: User | g in u.gender
}
 ------------------------------------------
---------------- Registration Facts ---------------------

--All FiscalCodes have to be associated to a Registration
fact FiscalCodeRegistrationConnection{
        all fc: FiscalCode | some r: Registration | fc in r.fiscalCode
}

--All FiscalCodes have to be associated to a User
fact FiscalCodeUserConnection{
        all fc: FiscalCode | some u: User | fc in
           u.registration.fiscalCode
}

--All PhoneNumber have to be associated to a Registration
fact PhoneNumberRegistrationConnection{
        all p: PhoneNumber | some r: Registration | p in r.phoneNumber
}

--All PhoneNumber have to be associated to a User
fact PhoneNumberUserConnection{
        all p: PhoneNumber | some u: User | p in
           u.registration.phoneNumber
}

--All Usernames have to be associated to a Registration
fact UsernameRegistrationConnection{
        all u: Username | some r: Registration | u in r.username  }

--All Passwords have to be associated to a Registration
fact PasswordRegistrationConnection{
  all p: Password | some r: Registration | p in r.password
}


--Every Customer has a unique username
fact NoSameUsername {
  no disj c1,c2: Customer | c1.registration.username =
     c2.registration.username  }

--Every User has a unique FiscalCode
fact NoSameFiscalCode {
  no disj r1,r2 : Registration | r1.fiscalCode = r2.fiscalCode
```

```
}


 ------------------------------------------
---------------- Report Facts --------------------

--All Dates have to be associated to a report
fact DateReportConnection{
  all t: Time | some re: Report | t in re.time
}
--Picutre have to be associated to a report
fact PictureReportConnection{
  all p: Picture | some re: Report | p in re.picture
}
--PlateNumber have to be associated to a report
fact PlateNumberReportConnection{
  all pl: PlateNumber | some re: Report | pl in re.plateNumber
}
--ViolationType have to be associated to a report
fact ViolationReportConnection{
  all v: TypeOfViolation | some re: Report | v in re.typeOfViolation
}

--Location have to be associated to a report
fact LocationReportConnection{
  all v: Location | some re: Report | v in re.location
}
--All Reports have to be associated to a User
fact ReportUserConnection {
  all re: Report | some u: User | u in re.user
}

 ------------------------------------------
---------------- Request Facts --------------------
--All Profile request have to be associated to a user
fact PrequestConnection{
  all pr: RequestForProfile | some u: User | (pr.user = u)
}

--All SafeArea request have to be associated to a location
fact SArequestConnection{
  all rsa: RequestForSafeArea | some l: Location | (rsa.location = l)
}

 ------------------------------------------
---------------- Data-Channel Facts --------------------
fact AccidentReportConnection{
  all a: AccidentReport | some ch: ChannelOne | a in ch.data
}
fact InterventionsConnection{
  all i: Interventions | some ch: ChannelTwo |i in ch.data3
}
fact DataFlowChannelOneConnection{
  all a: AuthorityToSystem | some ch: ChannelOne | a in ch.dataflow
}
```

```
fact DataFlowChannelTwoConnection{
  all a: SystemToAuthority | some ch: ChannelTwo | a in ch.dataflow
}
 -----------------------------------------
fact RegistrationCustomerConnection {
  all r: Registration | some c: Customer | r in c.registration
}

-----------------------Assertions---------------------

assert AccidentReportSource {
        all acr: AccidentReport| some at: Authority | some ch1:
           ChannelOne| acr in ch1.data and at in ch1.owner
}
check AccidentReportSource for 5

assert InterventionsCommunication{
        all i:Interventions | some at:Authority | some ch2: ChannelTwo
           | i in ch2.data3 and at in ch2.owner
}
check InterventionsCommunication for 5

---------------- Predicates ---------------------
pred world1{
        #Report = 2
        #User = 1


        (some disj re1, re2: Report | some disj u1: User | u1 in
           re1.user and u1 in re2.user)

}
run world1 for 10 but 0 Interventions, 0 AccidentReport, 0 Statistics,
   0 Authority


pred world2{
        #Report = 1
        #User = 1
        #Authority = 1
        #ChannelOne = 1
        #ChannelTwo = 1
        #Interventions = 1
        #AccidentReport = 3
        #Statistics = 1

}
run world2 for 5 but 0 RequestForSafeArea, 0 RequestForProfile



pred world3{
        #Report = 1
        #User = 2
```

```
}
run world3 for 3 but 0 Authority, 0 Interventions, 0 AccidentReport, 0
    Statistics
```

```
Executing "Check AccidentReportSource for 5"
  Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
  17225 vars. 1550 primary vars. 27266 clauses. 205ms.
  No counterexample found. Assertion may be valid. 118ms.
```

Figure 16: Result of Accident Report Source Assertion

```
Executing "Check InterventionsCommunication for 5"
  Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
  17499 vars. 1575 primary vars. 27542 clauses. 286ms.
  No counterexample found. Assertion may be valid. 198ms.
```

Figure 17: Result of Intervention Communication Assertion

In the above figures, the results of the assertions on accident reports and interventions are shown; these assertions were performed to ensure, first, the fact that accident reports come from authorities since the system is not able to produce with accident reports without such communication from the municipality through the agreed-upon data channels and to ensure that interventions are communicated to the authorities, this is due to the fact that the only motivation for the formulation of interventions by the system is to be communicated to the authorities.

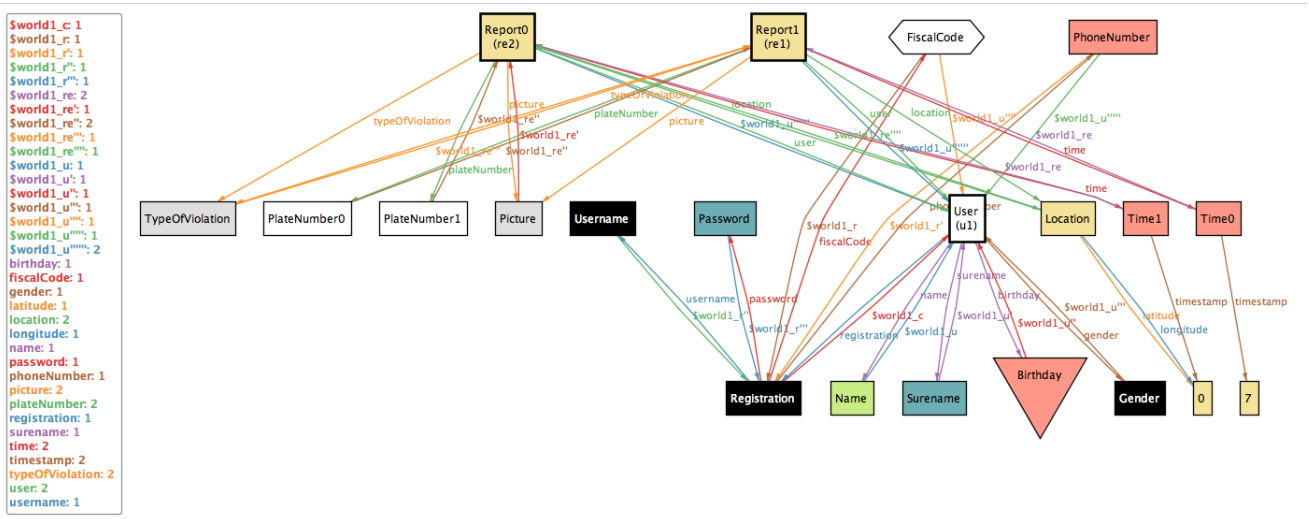


Figure 18: World One generated by Alloy Analyzer 4.2

The above world created by the Alloy Analyzer shows the overall consistency of the model focusing specifically on the users and the reports; in the sense that, all critical aspects of the system are modeled in a valid way. In example, all users have one and only one registration with a unique username and fiscal code, all reports are created by a unique user and that all reports have a location, time, plate number and an image of the violation in them.

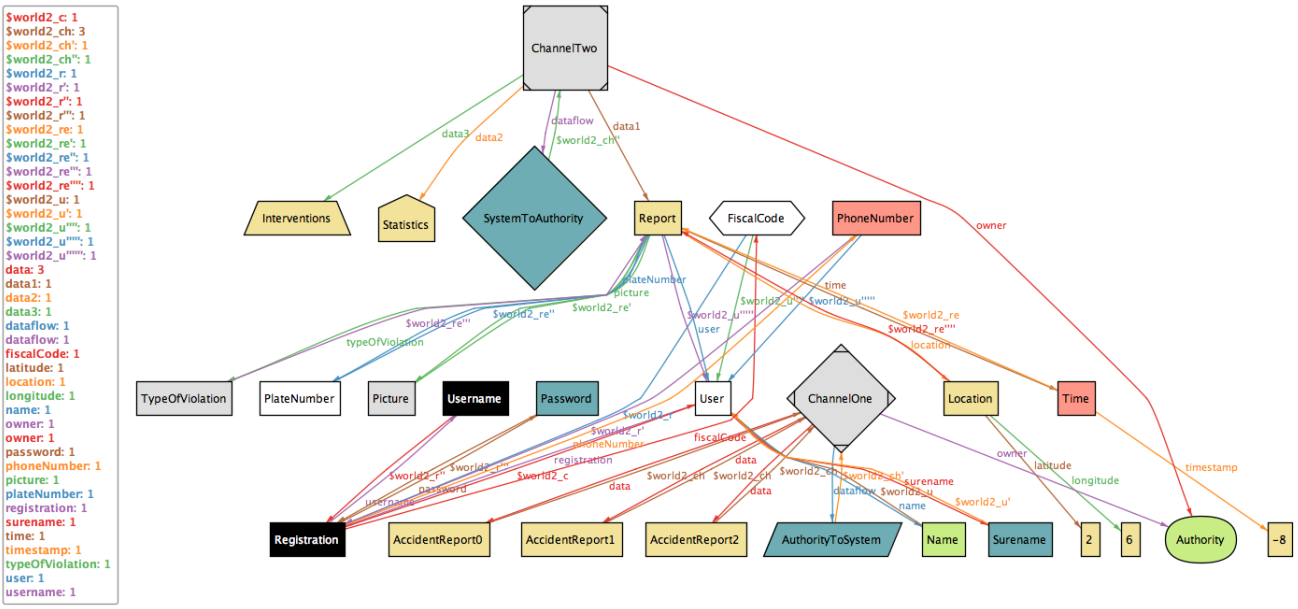


Figure 19: World Two generated by Alloy Analyzer 4.2

World two shown in the above figure focuses on the consistency of the authority communication; modeling, the communication channels, their data flows and the authority that owns them. As well as, modeling the flow of accident reports from the authorities to the system. It can be seen that all accident reports reach the system through a data channel which is owned by a unique authority as it has been checked in the ReportSource assertion and that interventions are linked to a data channel from the system to the authority.

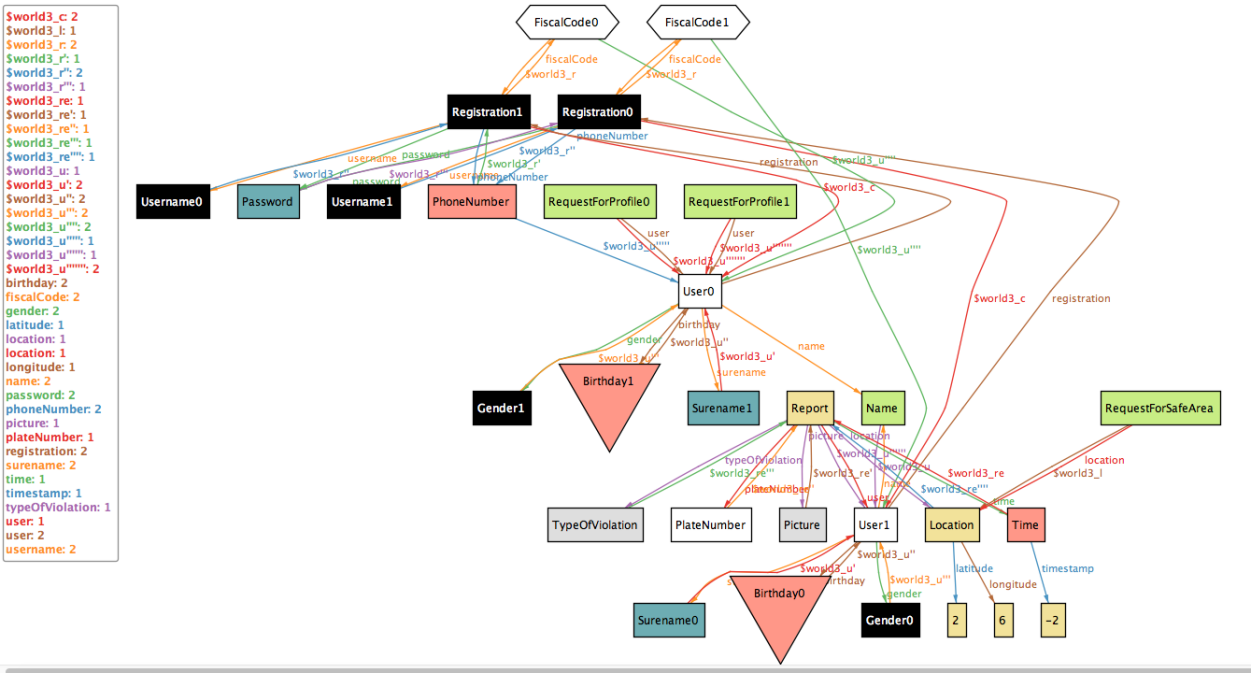


Figure 20: World Three generated by Alloy Analyzer 4.2

The last of the worlds generated in the figure above aims at the consistency of the general functions modeled in the analysis. Specifically, the area safety and profile requests. As seen in the figure, all profile

requests must be performed by a single unique user; as well as, the fact that a request to view the safety of a certain area must specify the location in order to be used to find matching reports.

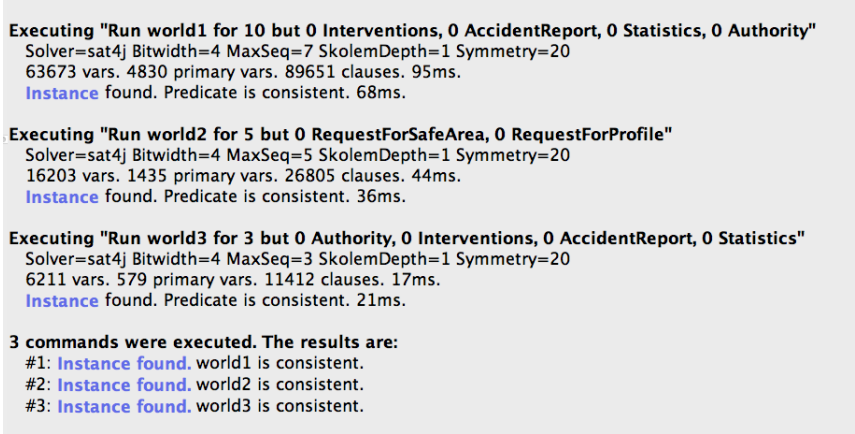


Figure 21: Result of Worlds generated by Alloy Analyzer 4.2

5 Effort Spent

Discription of the Task	Hours
karim Zakaria Saloma	
Introduction	5
Product perspective	7
Product functions	2
Domain assumptions	3
External interface requirements	1
Functional requirements	10
Non-functional requirements	3
Formal analysis using Alloy	5
Diagrams	11
Latex	3
Amirsalar Molaei	
Introduction	4
Product perspective	0
Product functions	2
Domain assumptions	3
External interface requirements	2
Functional requirements	4
Non-functional requirements	1
Formal analysis using Alloy	10
Diagrams	1
Latex	2
Erfan Rahnemoon	
Introduction	4
Product perspective	0
Product functions	2
Domain assumptions	3
External interface requirements	0
Functional requirements	4
Non-functional requirements	1
Formal analysis using Alloy	5
Diagrams	1
Latex	11

Table 10: Effort Spent by Each Team Member.