# SafeStreets

Marco Premi, Fabrizio Siciliano, Giuseppe Taddeo

October 24, 2019

## Contents

**1 Introduction** **3**
- 1.1 Purpose . . . 3
  - 1.1.1 General Purpose . . . 3
  - 1.1.2 Goals . . . 3
- 1.2 Scope . . . 4
- 1.3 Definitions, Acronyms, Abbreviations . . . 4
  - 1.3.1 Definitions . . . 4
  - 1.3.2 Acronym . . . 4
  - 1.3.3 Abbreviations . . . 5
- 1.4 Revision History . . . 5
- 1.5 Reference Documents . . . 5
- 1.6 Document Structure . . . 5

**2 Overall Description** **7**
- 2.1 Product Perspective . . . 7
- 2.2 Product Functions . . . 8
- 2.3 User characteristics . . . 8
- 2.4 Assumptions, dependencies and constraints . . . 8
  - 2.4.1 Assumptions . . . 8
  - 2.4.2 Dependencies . . . 8
  - 2.4.3 Constraints . . . 8

**3 Specific Requirements** **9**
- 3.1 External Interface Requirements . . . 9
  - 3.1.1 User Interfaces . . . 9
  - 3.1.2 Hardware Interfaces . . . 18
  - 3.1.3 Software Interfaces . . . 18
  - 3.1.4 Communication Interfaces . . . 18
- 3.2 Scenarios . . . 19
- 3.3 Functional requirements . . . 19
- 3.4 Use Cases . . . 19
  - 3.4.1 User use cases . . . 19
  - 3.4.2 Third party use cases . . . 19

3.5 Performance requirements . . . . . 19
3.6 Design Constraints . . . . . 19
3.6.1 Standars compliance . . . . . 19
3.6.2 Hardware Limitations . . . . . 19
3.6.3 Any Other Contraint . . . . . 19
3.7 Software System Attributes . . . . . 19
3.7.1 Reliability . . . . . 19
3.7.2 Availability . . . . . 19
3.7.3 Security . . . . . 19
3.7.4 Mantainability . . . . . 19
3.7.5 Portability . . . . . 19

**4 Formal Analysis using Alloy 20**

**5 Effort spent 21**

**6 References 21**

# 1 Introduction

## 1.1 Purpose

### 1.1.1 General Purpose

The purpose of this document is to correctly analyze all requirements, goals and actions needed in order to correctly develop SafeStreets.

SafeStreets is a crowd-sourced application that intends to provide users with the possibility to notify authorities when traffic violations occur (eg. traffic violations) and will help to maintain stability and order within the streets. A reporting system will also be available to police offices and municipality employees in order to allow them to analyze (and take actions accordingly) different areas of the city and assess which areas have the most violations committed in.

The core application will focus on storing useful traffic violation data provided by users, mainly with the help of input forms and hard evidence such as images. At any violation input, SafeStreets will also store useful metadata such as date and time the violation was retrieved, geolocate where it is and update a city wide map highlighting the areas where violations happen.

In addition to that, with the help of third parties (eg. municipality), SafeStreets will be able to retrieve the data given by such party and cross-reference them with its own data retrieved by users. By doing so, it will be possible to identify unsafe areas, assess which kind of problems happen more frequently and suggest possible intervention. It will also be possible for third parties (municipality and police officers) to automatically generate traffic tickets. This will be happening in order to cross-reference all available data in order to build statistics such as the most (or less) egregious offenders or the effectiveness of the SafeStreets initiative

### 1.1.2 Goals

Follows a list of all goals that will be reached with the SafeStreets initiative.

- G1: The system must allow all kind of users (both third parties and civilians) to correctly input (with hard evidence) traffic violations around the city;
- G2: The system must autonomously retrieve metadata from hard evidence useful to report and to all users;
- G3: The system must allow to create different clearance levels in order to offer different reporting systems to different kind of users;
- G4: The system will provide an efficient reporting system in order to highlight different violation categories through all parts of the city where the initiative is active;

## 1.2 Scope

With SafeStreets users can notify the authorities when traffic violations occur, and in particular parking violations. Both user and authorities must register to the application and agree that SafeStreets stores the information provided, completing it with suitable meta-data. The whole system, because it tracks users information, must respect the standards defined for processing of sensitive information such as GDPR if it is used in Europe. The user sends the type of the violation to the municipality and direct proofs of it (like a photograph). The system runs an algorithm to read the license plate and also asks the user to directly insert the license for a better recognition. Of course other information are required, like the name of the street when the violation has occurred, which can be retrieved from user's direct input or from the geographical position of the violation (using Google Maps API). Both users and authorities can highlight the streets with the highest frequency of violations or the vehicles that commit the most violations. SafeStreets crosses information about the accidents that occur on the territory of the municipality with his own data to identify potentially unsafe areas and suggest possible interventions. Because municipality could generates traffic tickets from the information about violations SafeStreets should guarantee that information is never altered (if a manipulations occurs, the application should discard the information). Such features are made possible trough the use of two mobile applications(one for the citizens and one for the officers on the field). The collected information are sent to a back-end. All the services can also be accessed through a specific web-site.

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

**User:** it is identified as a civilian customer of the product. It will be the main source for the SafeStreets initiative to obtain information about traffic violations and therefore be successful;

**Third parties:** those kind of organization/company that could provide services useful to SafeStreets and that will be able to retrieve data in order to improve the streets' safety;

**Customer:** it defines both third party SafeStreets users (police officers or municipality employees) and civilians;

**Ghiro:** image manipulation detection software, used by third party users in order to detect any image manipulation and assess the veracity of the hard evidence connected to the traffic ticket

### 1.3.2 Acronym

**UI:** User Interface

**GDPR:** General Data Protection Regulation

**API:** Application Programming Interface

**GPS:** Global Positioning System

### 1.3.3 Abbreviations

**Gn:** nth goal;

**Dn:** nth domain;

**Rn:** nth requirement;

## 1.4 Revision History

## 1.5 Reference Documents

## 1.6 Document Structure

**Chapter 1 - Introduction** Gives an introduction to the problem by describing the purpose of SafeStreets. It also shows the goals and the scope of the application.

**Chapter 2 - Overall Description** [H] Offers an overall description of the project. It identifies the actors involved in the application and lists all the assumptions in order to identify all the boundaries of the project. The product perspective includes details on the shared phenomena and the domain models. The class diagram describe the domain model used and the state diagrama analyzes:

- The process of collecting violations from users
- The process of sharing informations with the municipality

The majority of functions of the system are more precisely specified by taking in mind the goals of the system.

**Chapter 3 - Specific Requirements** Contains external interface requirements which are: user interfaces, hardware interfaces, software interfaces and communication interfaces. Few scenarios describing how the system acts in real world are listed here. Furthermore it provides the description of the functional requirements, through the use of use cases and sequence diagrams. The non-functional requirements are defined through performance requirements, design constraints and software system attributes.

**Chapter 4 - Formal analysis using Alloy** Includes the alloy model of some critical aspects with comments and documentation.

**Chapter 5 - Effort Spent** Shows the effort spent by each single group member while working on the RASD.

**Chapter 6 - References** Includes the documents we used as reference.

# 2 Overall Description

## 2.1 Product Perspective

SafeStreets is designed to be a completely new software applications. It uses some already proven services (Google Maps, PlateRecognizer APIs and Ghiro software) for its critical tasks. The software uses these services in order to double check whether both addresses and license plates are correctly standardized in order to be stored into the violations database and if the collected hard evidence have been somehow manipulated or corrupted.
The system is composed of two different mobile applications: one for the citizens that want to reports violations and one for the officers acting on the field. It also provides a web site for third party users which allows them to assess and analyze potential unsafe areas, thanks also to a powerful reporting system.
Taken into consideration that the municipality could generate traffic tickets from the input violations, the software will be critical when it comes to handling chain of custody. The latter is assured to never be broken by not allowing any kind of customer (user, PO or employee) to modify the reported violation. Supposedly, when some traffic violations might be erroneous or do not have any reason of existence, the inputting user can warn the responsible third party by attaching a warning explaining why it should not be taken into consideration. The systems also ensures the veracity of each violation and the hard evidence attached to it by running a image manipulation detection software (Ghiro, per instance). This process is used by third parties before the emission of each ticket to the corresponding offender.

A high-level class diagram can be found below, which provides a model of the application domain. The most important classes (not all of those which will be implemented once the software will be ready) are shown in order to define how the different components of SafeStreets will be communicating with each other. It is possible to identify two kinds of third party users: police officers and municipality employees. The first ones will be given access to both mobile application and web application; the latter will be provided access just to the above mentioned web application which will help these users assess the veracity of the hard evidence attached to the traffic violations and to further analyze unsafe areas around the municipality.

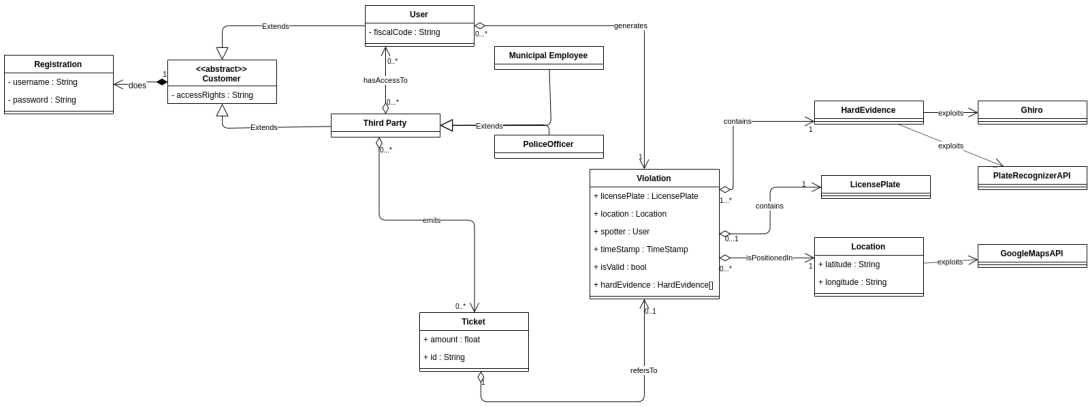


Figure 1: Class diagram

Each customer will be provided with a clearance level defined by a string

## 2.2 Product Functions

In the following section the most important product functions of the system are reported.

## 2.3 User characteristics

## 2.4 Assumptions, dependencies and constraints

### 2.4.1 Assumptions

### 2.4.2 Dependencies

### 2.4.3 Constraints

# 3 Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

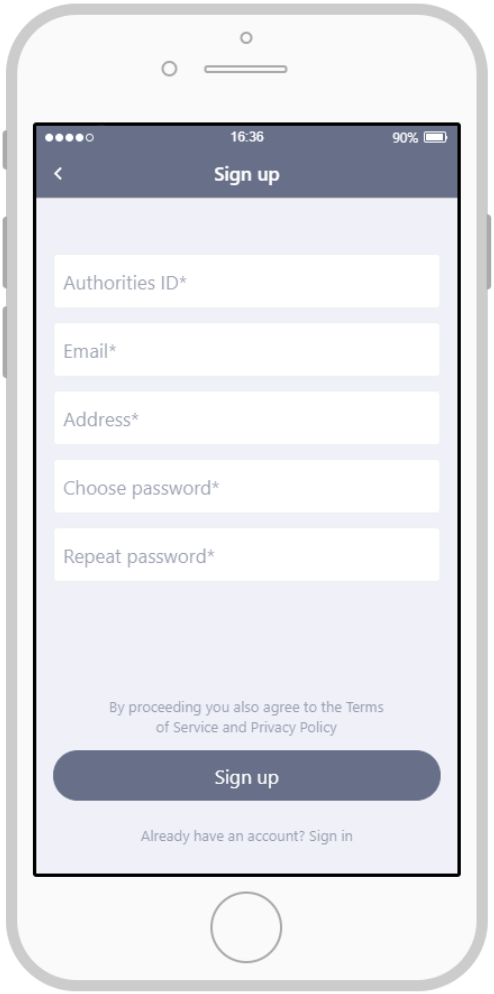


Figure 2: SignUp authorities

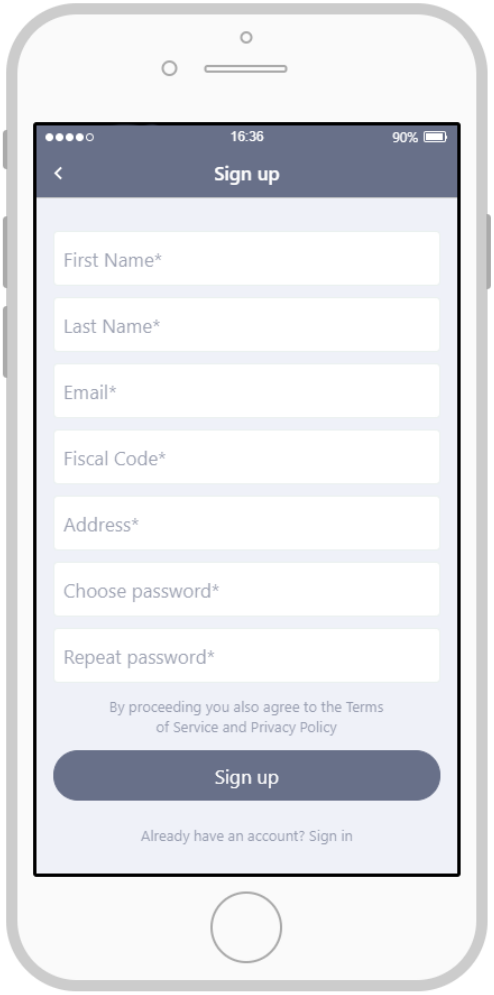


Figure 3: SignUp User

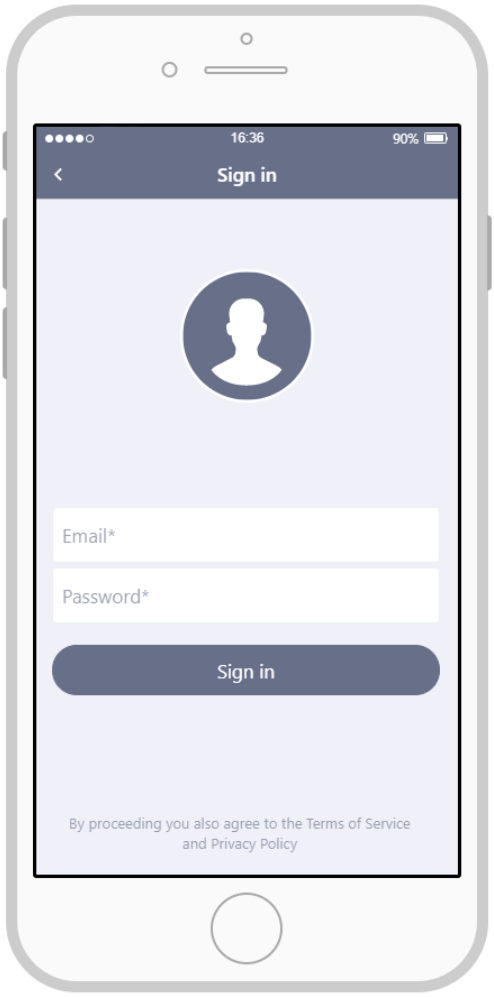


Figure 4: SignIn

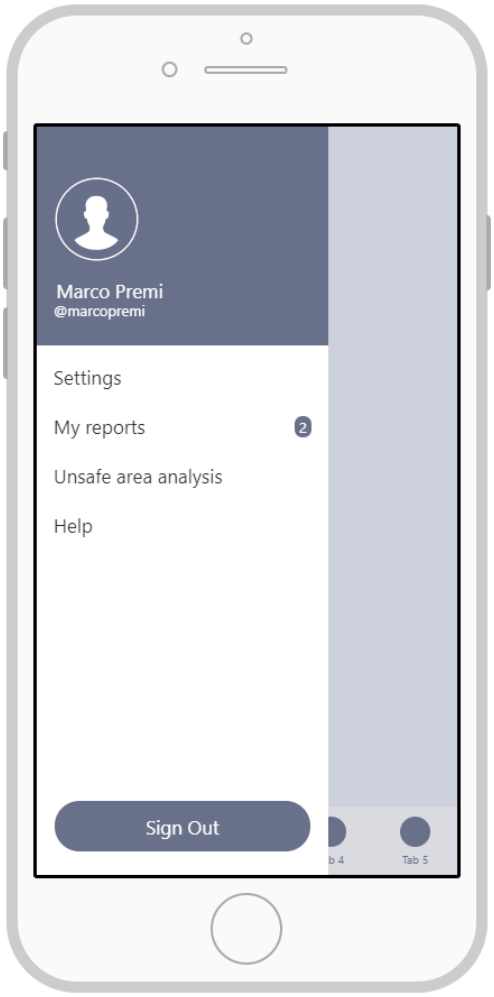


Figure 5: Menu

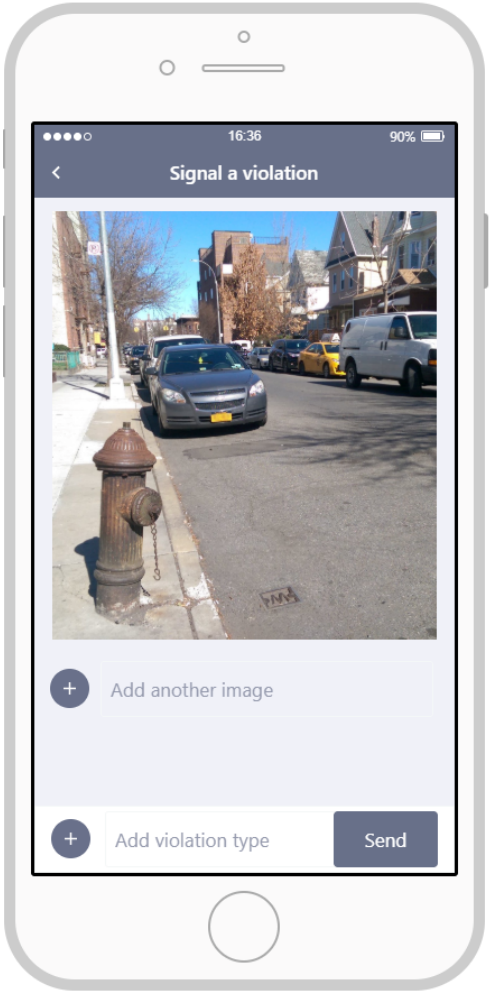


Figure 6: SignIn

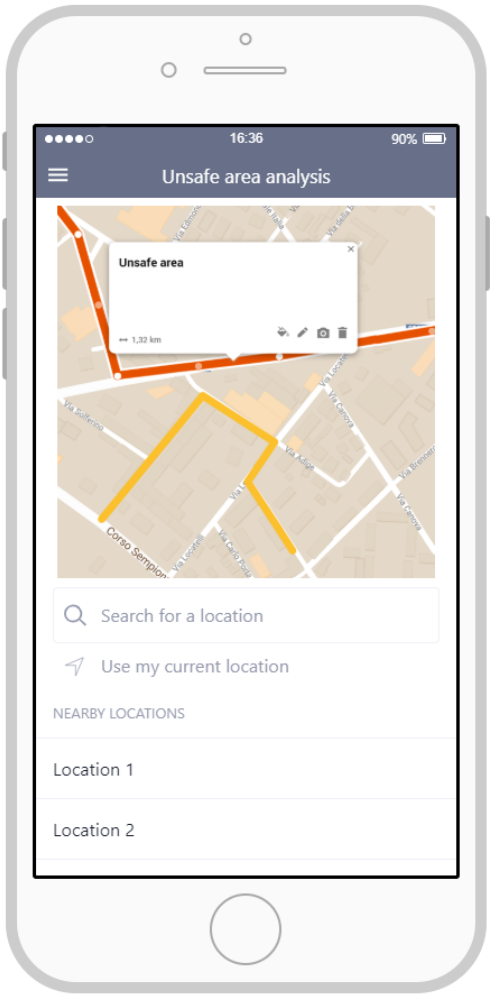


Figure 7: SignIn

Figure 8: SignUp

Figure 9: SignIn

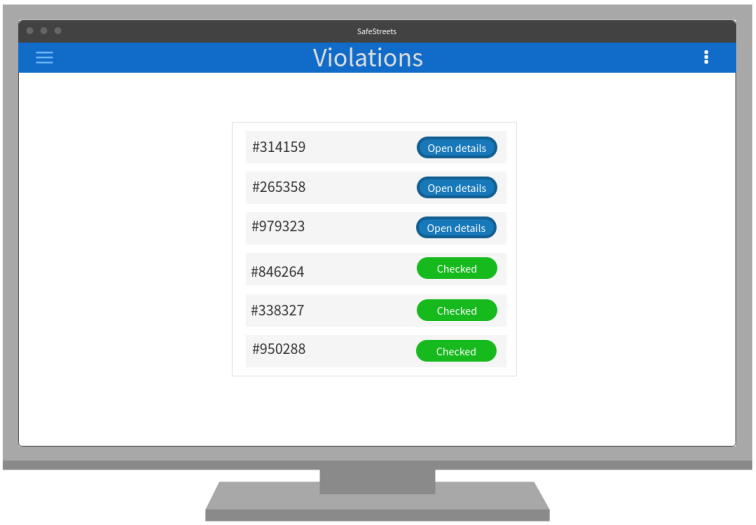


Figure 10: Violations

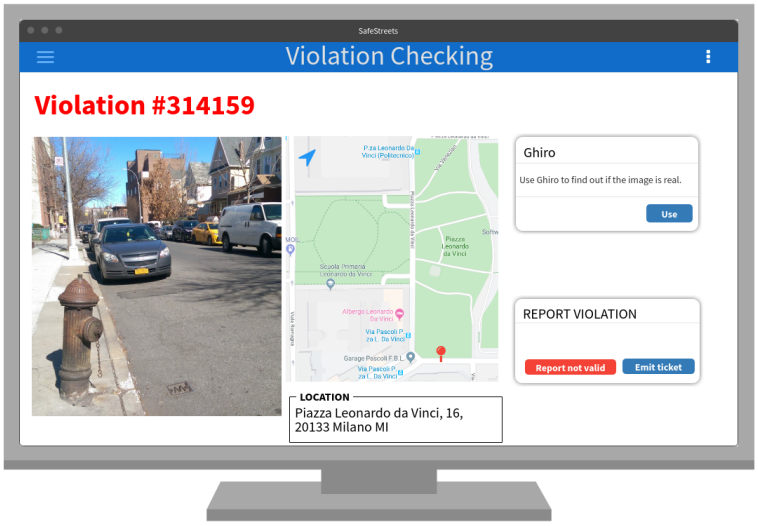


Figure 11: Violations checking

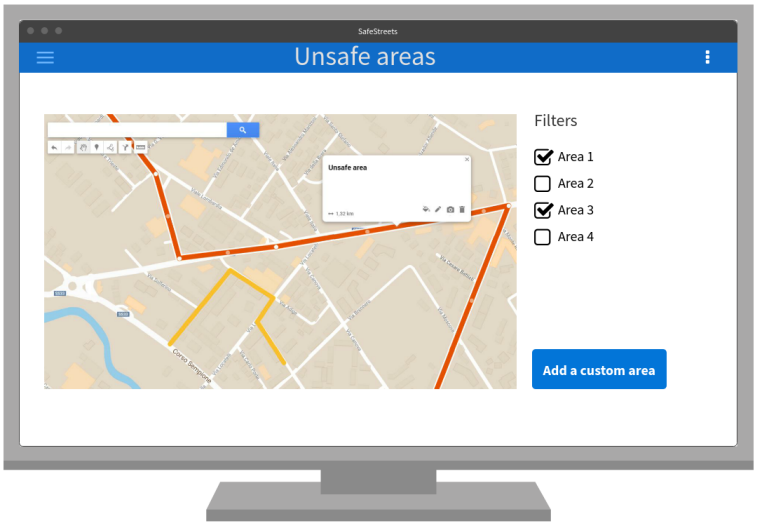


Figure 12: Unsafe Areas

### 3.1.2 Hardware Interfaces

The system has no hardware interfaces.

### 3.1.3 Software Interfaces

The system doesn't provide any API to external applications.
However some softwares part of SafeStreets is developed by other companies.

- Google Maps API: for localization and creation of unsafe areas.
- Plate Recognizer API: for License Plate recognition.
- Ghiro: a digital image forensics tool used for find out if the image in the violation report is real.

### 3.1.4 Communication Interfaces

The system only uses HTTP(more precisely HTTPS) as communication service.
HTTP/HTTPS is used for:

- User and third parties registration.
- Sending violations both to SafeStreets and to authorities.
- Using Google Maps API, Plate Recognizer API and Ghiro

## 3.2 Scenarios

## 3.3 Functional requirements

## 3.4 Use Cases

### 3.4.1 User use cases

### 3.4.2 Third party use cases

## 3.5 Performance requirements

## 3.6 Design Constraints

### 3.6.1 Standars compliance

### 3.6.2 Hardware Limitations

### 3.6.3 Any Other Contraint

## 3.7 Software System Attributes

### 3.7.1 Reliability

### 3.7.2 Availability

### 3.7.3 Security

### 3.7.4 Mantainability

### 3.7.5 Portability

# 4 Formal Analysis using Alloy

# 5 Effort spent

| Description of the task | MP | FS | GT |
|---|---|---|---|
| Introduction | 2.5 | 2 | 0 |
| Overall Description | 1.5 | 3 | 0 |
| Specific requirements | 0 | 0 | 0 |
| Formal analysis using Alloy | 0 | 2 | 0 |

# 6 References

**Plate Recognizer:** https://app.platerecognizer.com