Design Document
Safestreets

Luca Massini   Daniele Nicolò

release date to be defined

Contents

1 Introduction

1.1 Purpose

This document is necessary to describe the architecture of the system from several points of view. An architecture can be expressed in terms of the system components and their interactions with each other.

An architecture can be thought as a starting point for possible future changes,according to the needs of the stakeholders.

Finally the requirements mentioned in the RASD document will be explained here from an architectural perspective, that can be seen with component, deployment and runtime views.

1.2 Scope

Safestreets is a service that allows private users to inform authorities about parking and traffic violations. A user must take a picture of the violation and describe it and the place where it occurred. An image recognition algorithm is run on the sent picture. A user has also the possibility to see the safety of the streets and the parking areas and to check the most reported streets and vehicles.

The system offers also the possibility to receive suggestions in order to improve the streets safety. This feature is available only for municipality accounts. These suggestions are generated by the system using an algorithm that retrieves the information from the users' reports and also from the data given by the municipality.

1.3 Definitions,Acronyms and Abbreviations

1.3.1 Definitions

TODO

1.3.2 Acronyms

TODO

1.3.3 Abbreviations

TODO

1.4 Revision History

TODO

1.5 Reference Documents

TODO

1.6 Document Structure

The following sections are structured as below:

- **Architectural Design:** In this section there will be the description of the system from the architectural point of view. This means to show the components and their interactions with each other and to explain the design patterns choices and the architectural styles.

- **User Interface Design:** In this section there is an explanation,in terms of user experience (UX), of the user interfaces already showed in the RASD mockups.

- **Requirements Traceability:** Here we describe how the requirements already explained in the RASD match with the design choices done in this document.

- **Implementation, Integration and Test plan:** Here there is the description of how we will implement all the components, how we will integrate them together and finally how we will test both the single components and also the integrated system.

- **Effort spent:** Here there is the division of the work hours of each member of the group and the description of the tasks completed and related time spent.

2 Architectural Design

2.1 Overview

In the figure below it is shown the architecture of the entire system. This description has to be intended only as an overview and because of this some details are not shown.
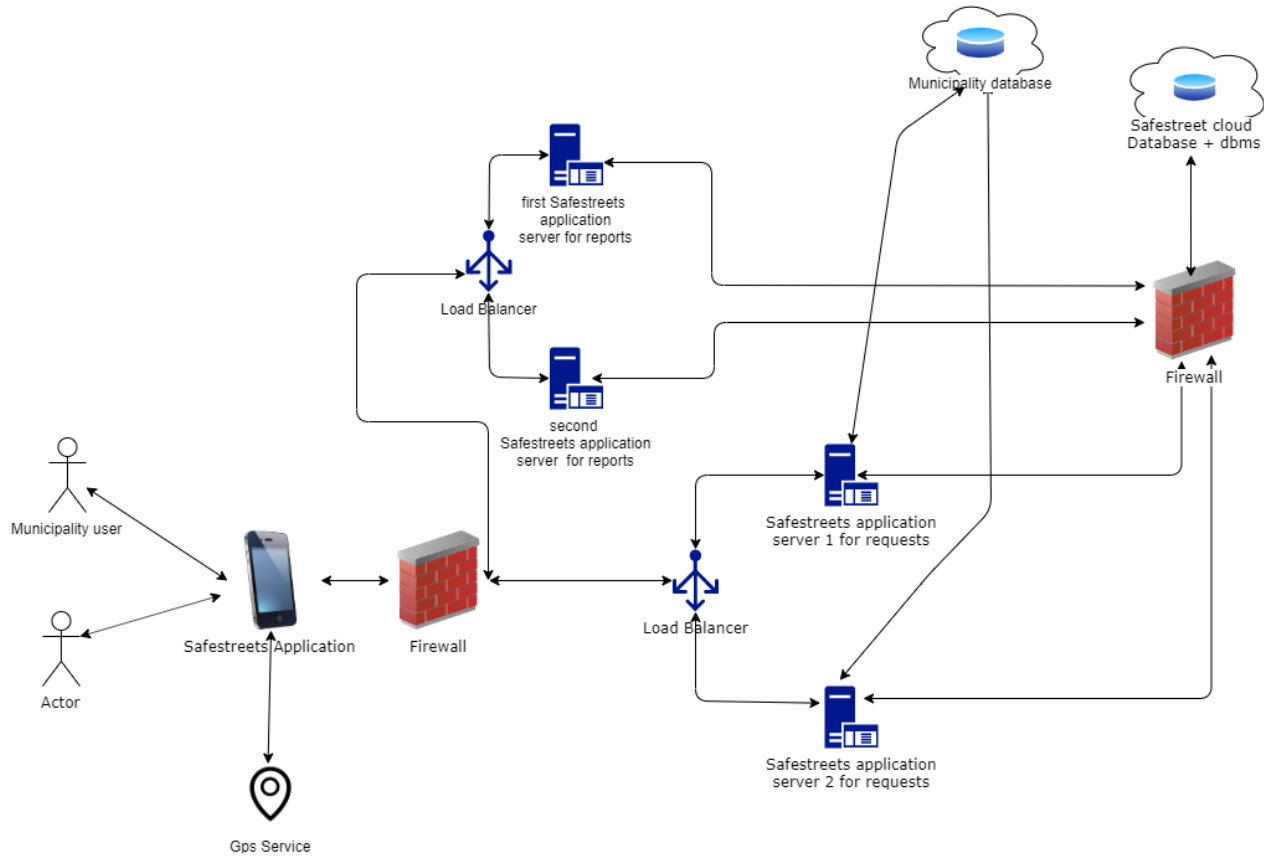

Figure 1: Architectural overview

In this overview we can immediately distinguish the architectural layers. Servers manage the logic one whereas databases manage the data layer and finally the client smartphone is where the presentation layer resides.

The architecture has 2 clusters. Every server contained in one cluster access the system databases whereas instead only the two databases dedicated to requests access the external one (which is the external database of the municipality). Servers which resides in different cluster don't communicate with each other to guarantee the service. Also servers of the same cluster don't communicate with each other obviously. The workload balancing between servers of the same cluster will have to work thanks to load balancers.

The client in order to receive the service will speak directly and only to the firewall placed at the entrance of the system. The first firewall (the one near the SafeStreets application in the diagram) is the element that will rout the user message to the right cluster based on the type of request. In this case that firewall has not only a security function (obviously the main function of a firewall is that) but it also behaves as a gateway which routes packets. All the clusters are placed inside the DMZ (between two firewalls) for security reasons. They must be accessible from outside the internal network. The second firewall divides clusters from the system database. The database will have to be accessible only from the clusters for security reasons and because of this fact they are outside of the DMZ.

The fact that we have two independent servers per cluster that offers the same services will cost more for sure but we want this system to be discreetly fault-tolerant and performing. A solution with only one server would have been too much dangerous in the terms expressed just above.

2.2 Component view

Here there is a description of the software components that will be implemented in the specific hardware.
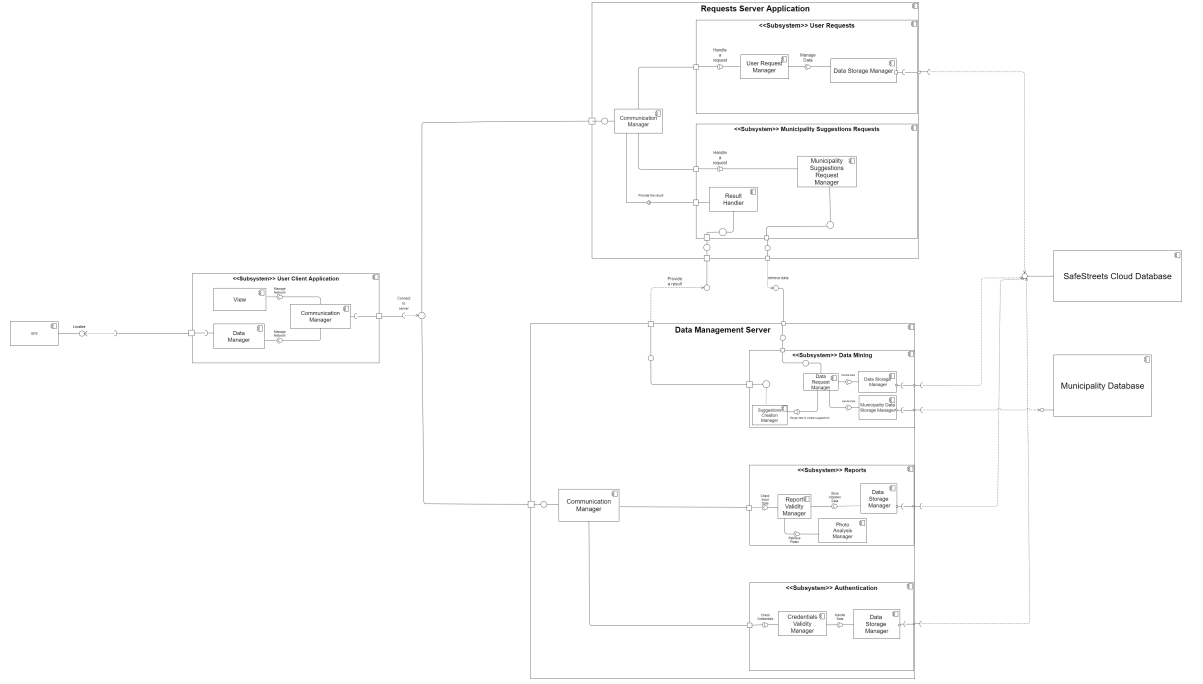
Figure 2: Component diagram

With this diagram our purpose is to show the internal architecture of SafeStreets. The main components are the user client application, the requests server application and the data management server. The external interfaces are SafeStreets cloud database, the municipality database and the GPS interface.

These main components contain smaller modular components that communicate with each other and with the external interfaces to complete different functions. Now we will describe the internal components and the external interfaces in detail.

User Client Application The User Client Application is situated in the user's smartphone. It contains the Data Manager, the View and the Communication Manager:

- **Data Manager**
  This component is responsible of taking the user's position in the world using the GPS service offered by his smartphone. Those data will be sent to the server in case he selects a service that uses the GPS localization.

- **View**
  This component is the view part of the MVC pattern. It takes data

provided by the controller situated in the Server and shows them to the user with its interface.

- **Communication Manager**
  The Communication Manager is responsible for the network. It manages all the communication from the client to the server and vice versa.

Requests Server Application   This component is situated on one of the two servers and contains a Communication Manager, the User Requests component and the Municipality Suggestions Requests one:

- **Communication Manager**
  It has the same function of the one situated on the client. It dispatches and receives message from the client application.

- **User Requests**
  This component manages all the requests coming from the private users and the municipality, except for the suggestions requests. It contains a User Requests Manager, that elaborates the requests and asks for the necessary data to the Data Storage Manager. The User Requests component contains also the Data Storage Manager that retrieves the requested data querying the SafeStreets external database.

- **Municipality Suggestions Requests**
  This component deals with the suggestions requests made by the municipality. It contains the Municipality Suggestions Requests Manager that receives the request from the Communication Manager and asks for the needed data to the Data Mining component, situated in the other server in the Data Management Server Application. When the suggestion is ready, the Data Mining component send it back to the Municipality Suggestions Request component, especially to the Result Handler, that dispatches it to the Communication Manager that sends it to the Client Application.

Data Management Server Application   This component is situated on the other server compared to the Requests Server Application. It contains the Data Mining Component, the Reports Component and the Authentication one:

- **Data Mining**
  This component is responsible for the retrieval of the information needed to provide suggestions to the municipality. It receives from the Municipality Suggestions Requests Component a request for suggestions. It contains the Data Requests Manager that handle the received request and asks for data to other two components, also situated in the Data Mining Component. They are the Data Storage Manager that provides the data about the violation and the streets that SafeStreets has in its cloud database, and the other is the Municipality Data Storage Manager that provides

the data, again about the violations and the streets, but now querying the Municipality external database. When all those data arrive to the Data Requests Manager, it dispatches them to the Suggestions Creation Manager, that runs a data mining algorithm provided by an Oracle API in order to merge the received data and retrieve information useful to provide suggestions to the Municipality. Finally the result is sent to the Result Handler, situated in the Municipality Suggestions Requests Component, that proceeds to give it to the User Client Application.

- **Reports**
  This component is responsible for the violation reports. It receives the reports from the Communication Manager and it contains a Report Validity Manager that checks that the violation data inserted by the user are correct. Then it has the Photo Analysis Manager that retrieves the license plate of the violator from the photograph using an algorithm provided by an external API that scans the photo. Finally there is a Data Storage Manager that takes all the information about the violation (license plate, type of violation, description, position, date and hour) and stores them in the SafeStreets database.

- **Authentication**
  The Authentication component manages all the operation about login, sign up and logout. It contains a Credentials Validity Manager that checks the user's input (both in login and sign up) and checks if he is logged in whenever he wants to logout. Then it contains also a Data Storage Manager that asks for the account data that has to be checked by the Credential Validity Manager and stores the user's account data in case of sign up.

External Interfaces Some of the components already described have to communicate with external interfaces in order to guarantee specific functionalities. These interfaces are:

- **SafeStreets cloud database**
  This component is shared with both Servers. In each of the two servers there is a component that is the only one that communicates with this database and it's the Data Storage Manager, that can both query to retrieve data and add new information into the database.

- **Municipality Database**
  This component communicates with the Data Management Server Application, in particular with the Municipality Data Storage Manager, that can only query to obtain information in order to create suggestions for the Municipality. This database is in read-only mode.

- **GPS**
  This external component is necessary for the user's automatic localization.

It communicates with the User Client Application, in particular with the Data Manager that is the one specialized in retrieving the position information (city and street) from the GPS service offered by the user's smartphone.
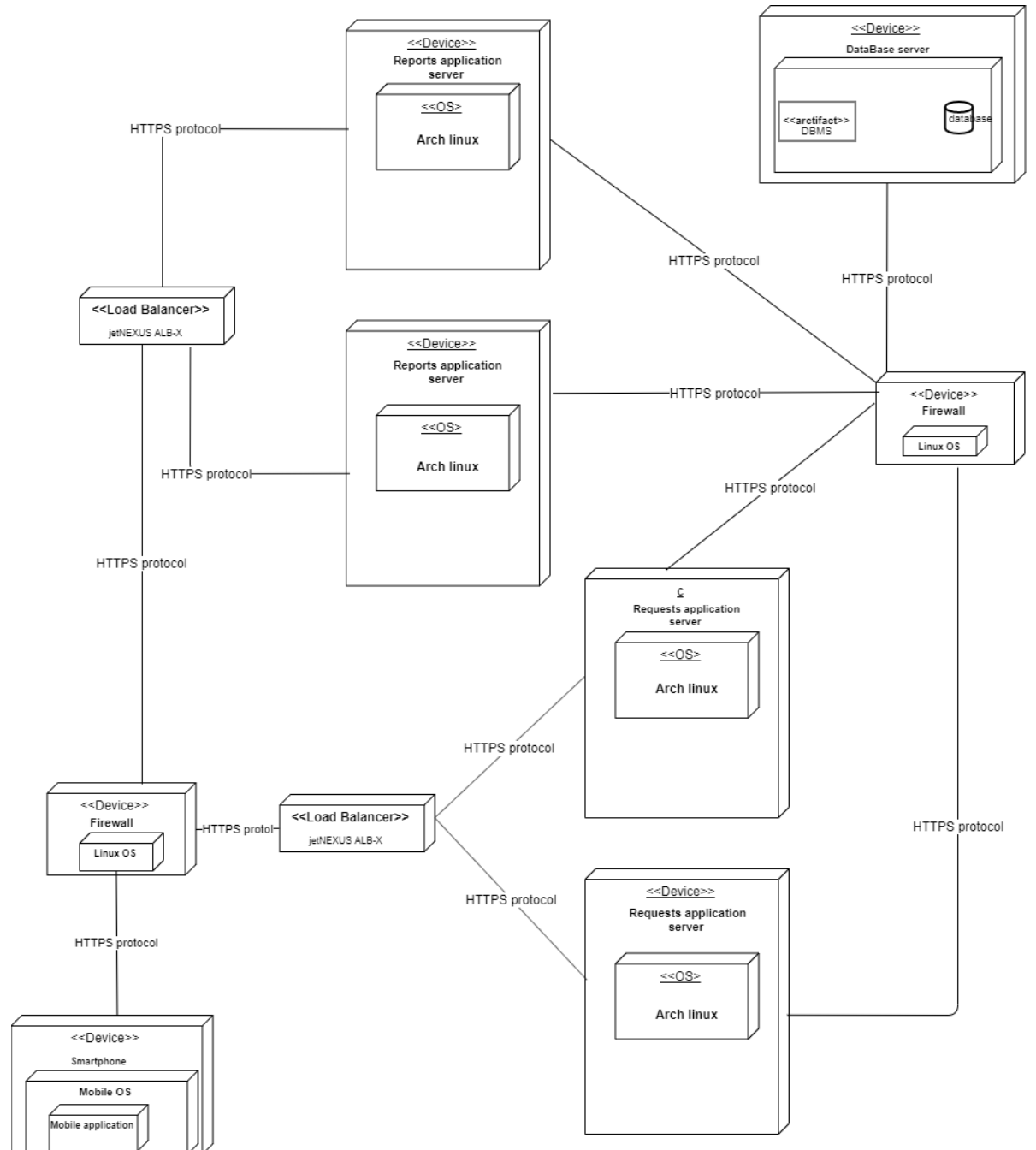
2.3 Deployment view

Figure 3: Deployment diagram

This diagram is very similar to the one in the overview. It express some aspects which have been already explained and some others are explained only here. In fact we can see all the operating systems for each server and also the protocol used in the communication between the devices. Here you can find a brief description of what the single components do:

- **Firewall:** Firewalls are well known for sure by the majority of the people but we think that it is important to describe their operation the same. Firewalls are placed between the internal network and the external ones.They are used to filter the incoming/outcoming packet traffic. Through a set of rules they decide if a packet can pass or (unique alternative) be blocked. This can be very useful to guarantee a better system safety.

- **Load Balancer:** Load Balancers are used to improve the workloads across multiple computing resources. Thanks to load balancer the load can be balanced in an optimal way.This can make the system more performing and can also improve the fault tolerance of the system because of redundancy.

- **Servers:** In this architecture we can find two clusters, each one containing 2 application servers. The servers are specialized with no overlap of functionalities. One is specialized in the managing of the reports by the private users. It must update the database with the new incoming data only after having done all the checks about the validity and correctness of the report.
The other server deals with the managing of the requests by the private users or municipalities. It is in charge of check the correctness and validity of the request,and then it must run all the data mining algorithms necessary to retrieve the result and finally send the results to the right private user.

- **Databases:** The databases are cloud databases managed thanks to an API that will be provided by the provider of the service. We have only one cloud database used to store all the Safestreets data. Another cloud database is the external one of the municipality. Safestreets has only a special view of the part of the municipality database concerning the accidents. The access to the external database will have to be possible thanks to an API provided by the municipality. This API will have to follow a standard provided by us.

2.4 Runtime View

2.5 Component interfaces

2.6 Selected architectural styles and patterns

We have selected an 5 tiers architecture. In this architecture the presentation layer is present only in the smartphone of the user. This decision is due to the fact that we want the mobile application to be light and runnable with discrete performance in a lot of phones. We an application that limits as little as possible the hardware in a phone in terms of performance. The system data are stored in a database whereas the systems uses also another database which is the one of the municipality. The logic is divided in two specialized clusters. One, as already explained before, is specialized in the request and the other one is specialized in the receiving of the reports and them managing. The load is spread thanks to load balancers in the

2.7 Other design decisions

3 User interface design

Here there will be shown the user interfaces with the help of UX diagrams. In particular there is shown the flow that the user will follow to navigate inside the application.Some aspects concerning the user interfaces design have been already treated in the "Requirement Analysis and Specification Document",if the reader wants to understand in a better and more detailed way, he/she is strongly suggested to refer also to it. UX diagrams:
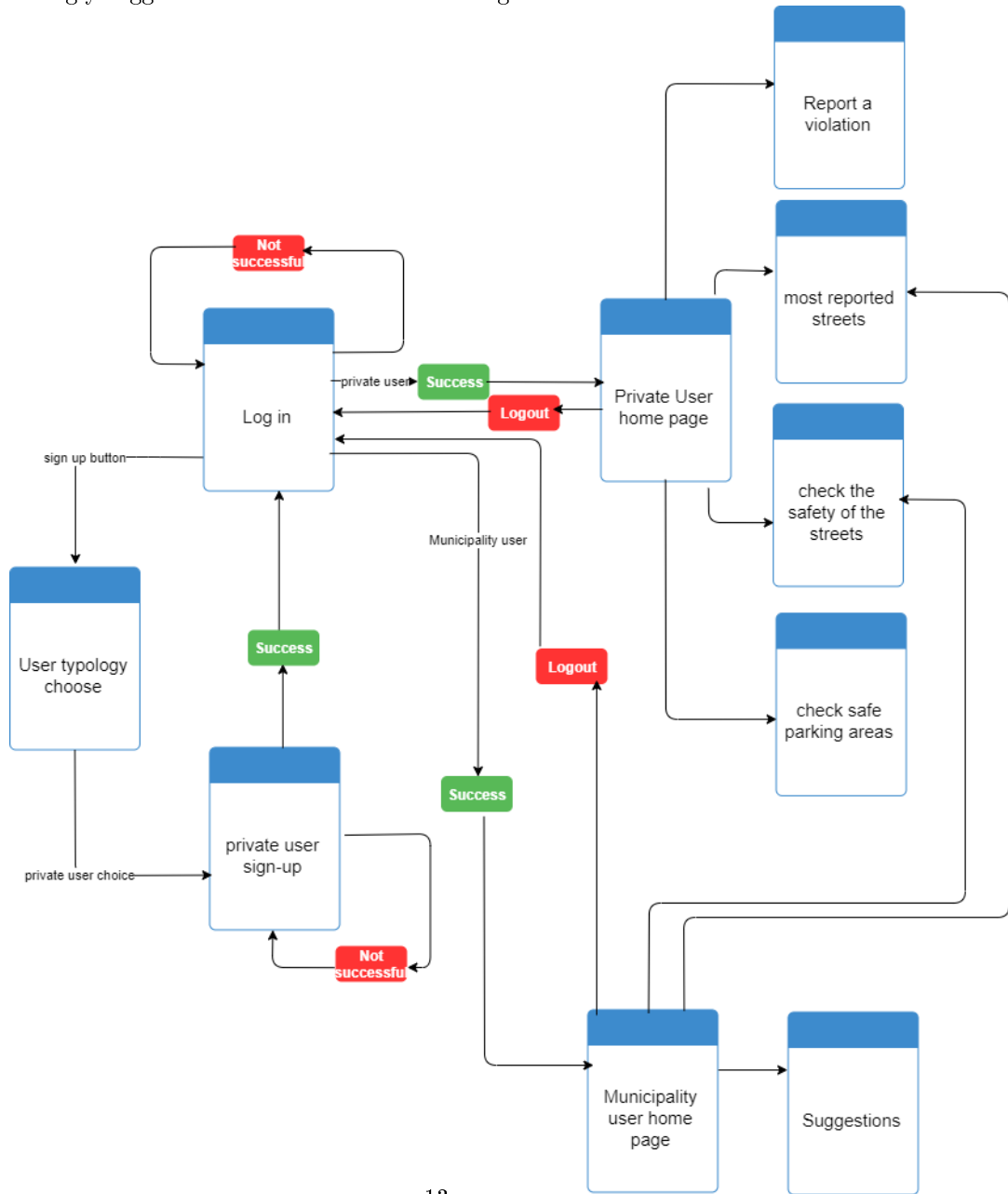
Figure 4: UX diagram