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.
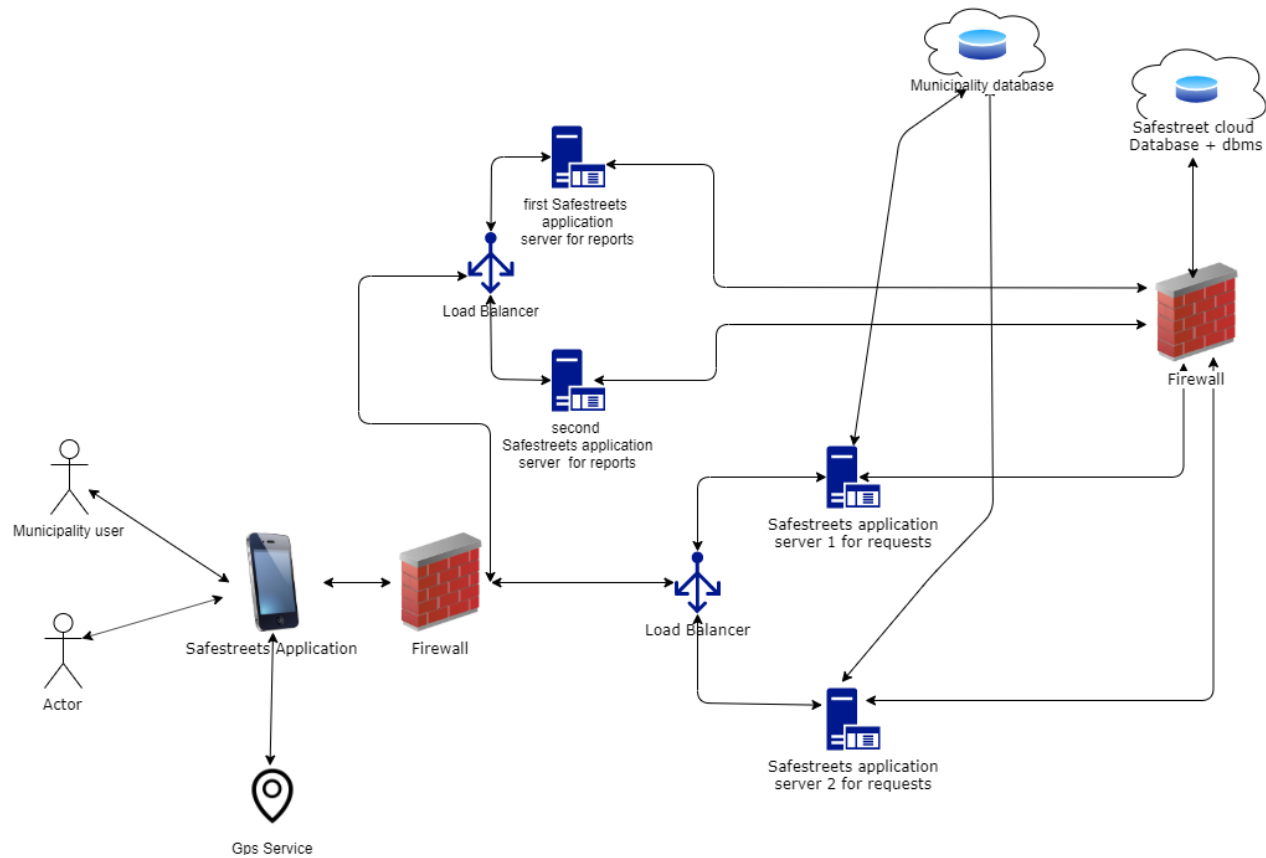

Figure 1: Architectural overview

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 an high level description of the software components that will be implemented in the specific hardware.
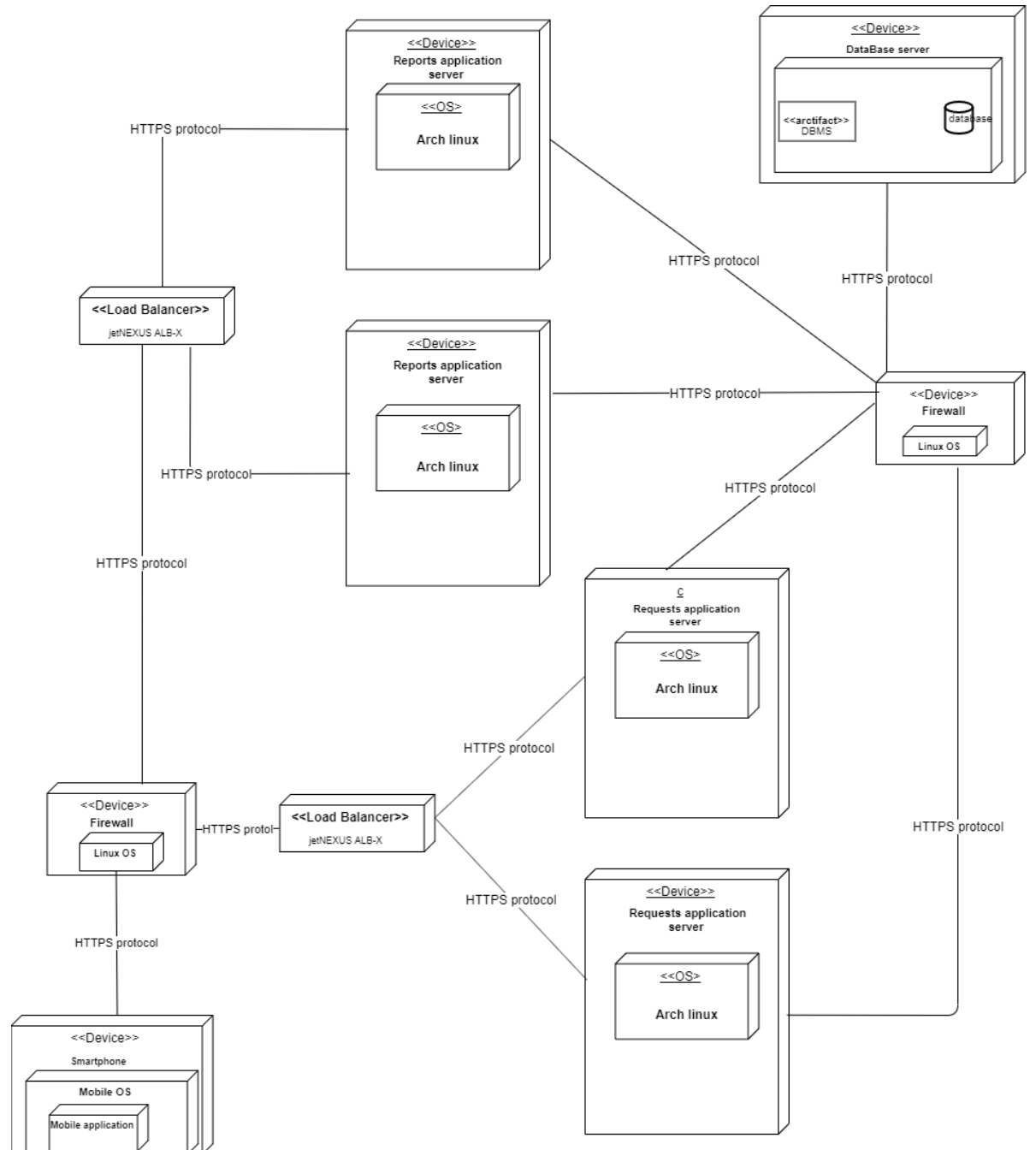
2.3 Deployment view

Figure 2: 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:
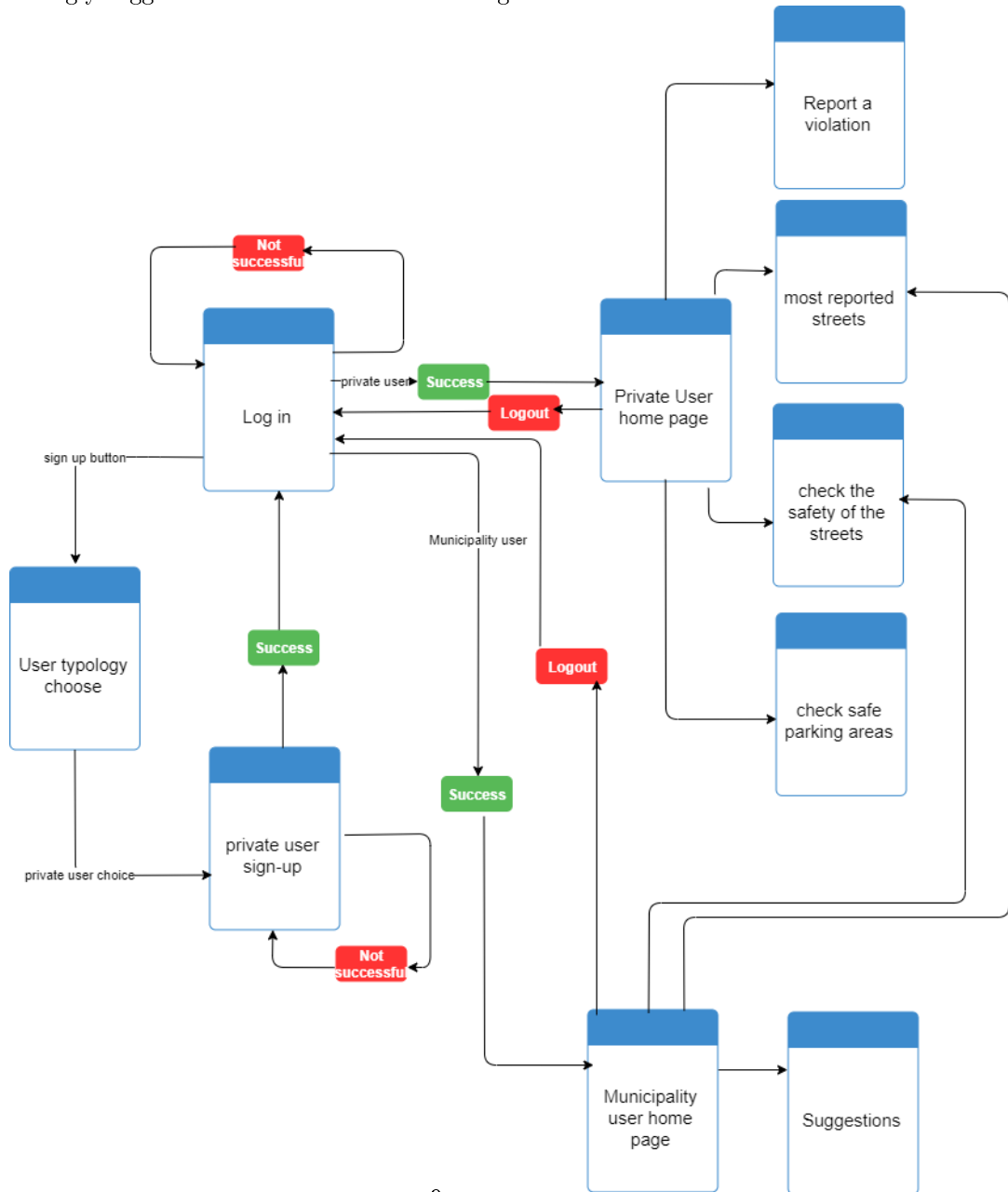
Figure 3: UX diagram