Politecnico di Milano

A.A 2019/2020

Design Document

SafeStreets

Authors:

Armenante Valerio

Capaldo Marco

Di Salvo Dario

Contents

1 Introduction

1.1 Purpose

In the presented document is the Software Design Document where there is a general view of the SafeStreets platform presented by the RASD, with all the functions that the system has to realize and to give implementative and technical details. Specifically, this document wants to concentrate on the analysis with high level descriptions of the main aspects, describe the architectural styles and pattern, and generally establish the design standards for the development phase of components, run-time processes, deployment. All the informations that are going to be written in this document are also intended to behave as a guide to follow during the software developing process. The document will pursue the analysis following a rigorous scheme, to make a clear and complete description about these characteristics of the software. In particular, this document is intended for stakeholders, software engineers, and programmers and must be used as reference throughout the whole development of the system. The document is going to present an overview of the high level architecture, together with an explanation of the main components and the communication system between them through their interfaces; the description of the runtime behavior together with an overview of the deployment mechanism; the definition of the design patterns of the most crucial parts of both the mobile application and web application; the definition of the implementation, integration and testing plans.

1.2 Scope

As presented in the RASD Document, the SafeStreets software allow users to notify traffic violations to authorities. More specifically, the notification process consists in the acquisition of pictures, date, time, location and type of reported violation inserted by user. The aim of SafeStreets basic service is to retrieve and store information from inputted data, completing it with suitable metadata. In particular, when user sends pictures related to a traffic violations, SafeStreets is able to recognize license plate by means of an external algorithm. After that, SafeStreets dispatches the notification to the authority member, which is in an available status and is chosen according to the minimum distance between the notified traffic violation and authority member's position. In the end, the assigned authority member will opportunely manage the received request. In addiction, users and authorities member, through a Mobile Application interface presenting different level of visibility to different roles, will be able to mine information that has been received, for example by highlighting the areas with the highest frequency of violations. This is going to be realized designing some algorithms able to generate statistics on the previously stored data. A user who approaches the mobile application for the first time has to complete a registration process, as well as an authority which approaches the dedicated website for the first time. In particular, an authority inserts all their authority member in SafeStreets database through a website's form specifying their e-mail and unique code. In this way, each authority member have to do only an activation process, inserting their personal data and the assigned unique code. Thanks to the first advance function, the software is able to improve more the security of cities through a cooperation with municipality. First of all SafeStreets retrieves information from municipality through an external service, then information are categorized (accidents, violations, etc.) and some statistics are generated. Then crossing in-

formation between resultant elaboration and already available SafeStreets data, the goal can be reached. More specifically, if traffic violations cardinality, related to an area, exceed a specified threshold, SafeStreets identifies the area as unsafe and suggests possible interventions to municipality. Obviously, the threshold is properly defined according to the characteristics of the city. Then through the second advance function, SafeStreets can provide correct information, related to traffic violations, to municipality(in particular local police). In this way, municipality can easily and correctly emits traffic tickets to the offenders. In order to ensure the correctness of information, SafeStreets implements an algorithm that recognize if a manipulation is occurred on uploaded data by users. Moreover, Safestreets builds some statistics regarding traffic tickets data such as a ranking of the most offenders, most common violations and the effectiveness of SafeStreets initiative.

1.3 Definitions, Acronyms and Abbrevations

1.3.1 Acronyms

RASD - Requirements Analysisand Specifications Document

DB - Database

DBMS - Database Management System

API - Application Programming Interface

DD - Design Document

GUI - Graphical User Interface

MVC - Model View Controller is a designed pattern used for GUIs

JDBC - Java DataBase Connectivity

RMI - Remote Method Invocation

PC - Personal Computer

HTML - Hyper Text Markup Language

CSS - Cascading Style Sheets

AM - Authority Member

1.3.2 Abbreviations

Gn - n-Goal

Rn - n-functional Requirement

1.4 Revision History

TBU

1.5 Document Structure

Chapter 1 Gives an introduction of the design document defining the purpose and the scope of the document as well as the utility of the project, possible text conventions which might be present, and the framework of the document. It might include, as the RASD, some used acronyms and abbrevation list to provide a better understanding of the document to the reader.

Chapter 2 Deals with the architectural design of the application illustrating the main components of the system and the relationships between them. It wants to give an overview of the architecture together with providing an image of the most relevant architecture views: component view, class view, deployment view and runtime view. It shows, then, the inteeraction of the component interfaces. Here there might be presented also some of the defined design patterns together with some used architectural designs, and an explanation of each one of them together with the purpose of their usage.

Chapter 3 Refers to the Mock-Ups already presented in the RASD document.

Chapter 4 Explains how the requirements that have been defined in the RASD are mapped to the design elements that are described in this document in a way that an association between the decisions taken in the RASD with the ones taken in this DD is generated.

Chapter 5 Identifies the order in which it is planned to implement the

subcomponents of the system and the order in which it is planned to integrate such subcomponents and test the integration.

Chapter 6 Shows the effort spent by each group member while working on this project.

Chapter 7 Presents the reference documents.

2 Architectural Design

2.1 Overview: High-level components and their interaction

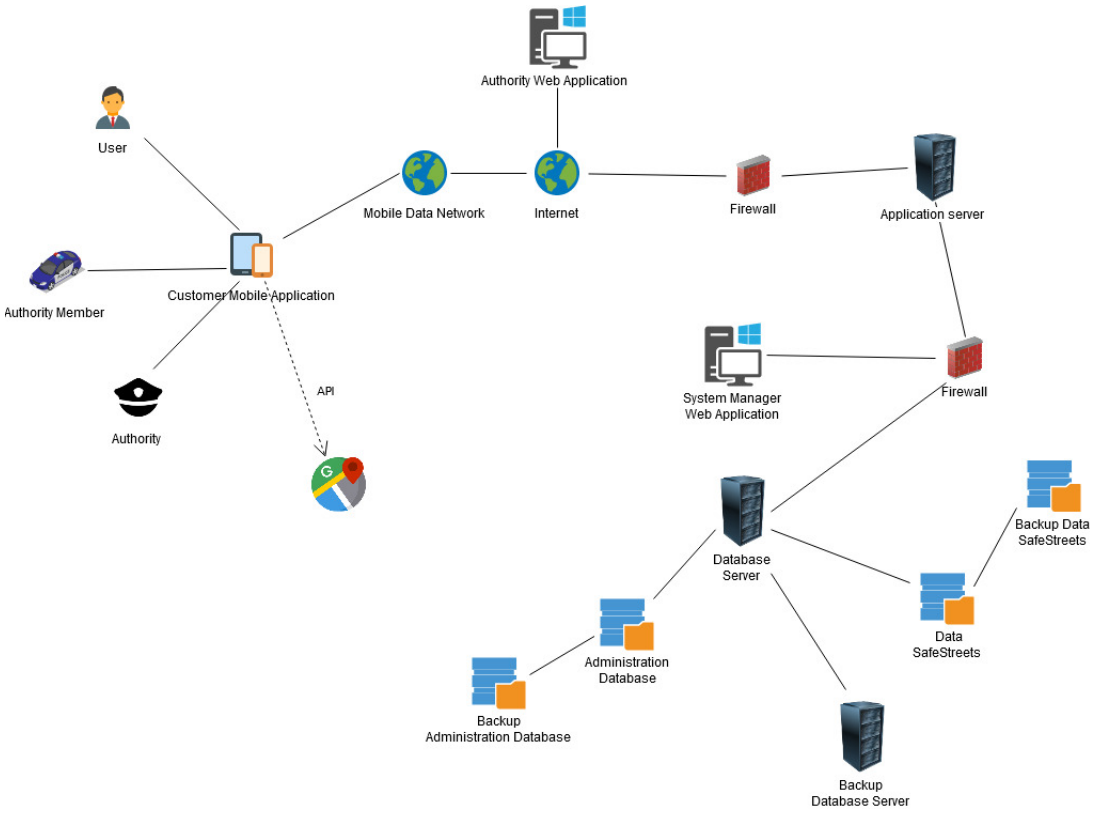

Figure 2.1: Physical architecture diagram.

The high level architecture of the system is divided in three main tiers (as shown in figure 2.1):

- **Client-side tier**: this is the closest part of the system to the customers: authority member, authority and user's interfaces. It is composed by the customer mobile application, and the authority web application.

- **System core network tier**: identifies the central portion of the infrastructure, the "mind" of the system. It is composed by the Application Server, which is protected by Firewalls, and by the System Manager Web Application, which owns a direct interface to the Application Server and monitor all the functionality aspects of the system.

- **Storage tier**: identifies the storage tier of the infrastructure, where there are all the backups of the data of our system. It is composed by the Database Server, SafeStreets Data Database and Administration Database.

The System core network tier is composed by the the Application Server, the "mind" of the system, that is protected in an efficient and secure way thanks to firewalls placed before the communication with the internet network , the Database Server and the System Manager Web Application. The System Manager have to monitor the Application Server with the aim to manage events into the system, for example Request of Data Access like statistic's data or other type of data saved in the system itself, or the Registration Request of new customer and so on. Furthermore, the System Manager can access to the Storage tier through the Application Server and manage the data in the Database. The Customer Mobile Application is the User client front-end portion of the system, that owns an interface to the Mobile Data Network, through which the client can access to the Internet network and can use the services offer by the system. The Database Server receives and manages every incoming request, which regards one of the storage entities among SafeStreets Data Database and Administration Database (it holds Data about subscribed Authorithy, Authority Mem-

ber, User and other relevant administration-oriented data). The system is completely oriented towards data integrity and persistence thanks to the Backup Database Server.

2.2 Component view

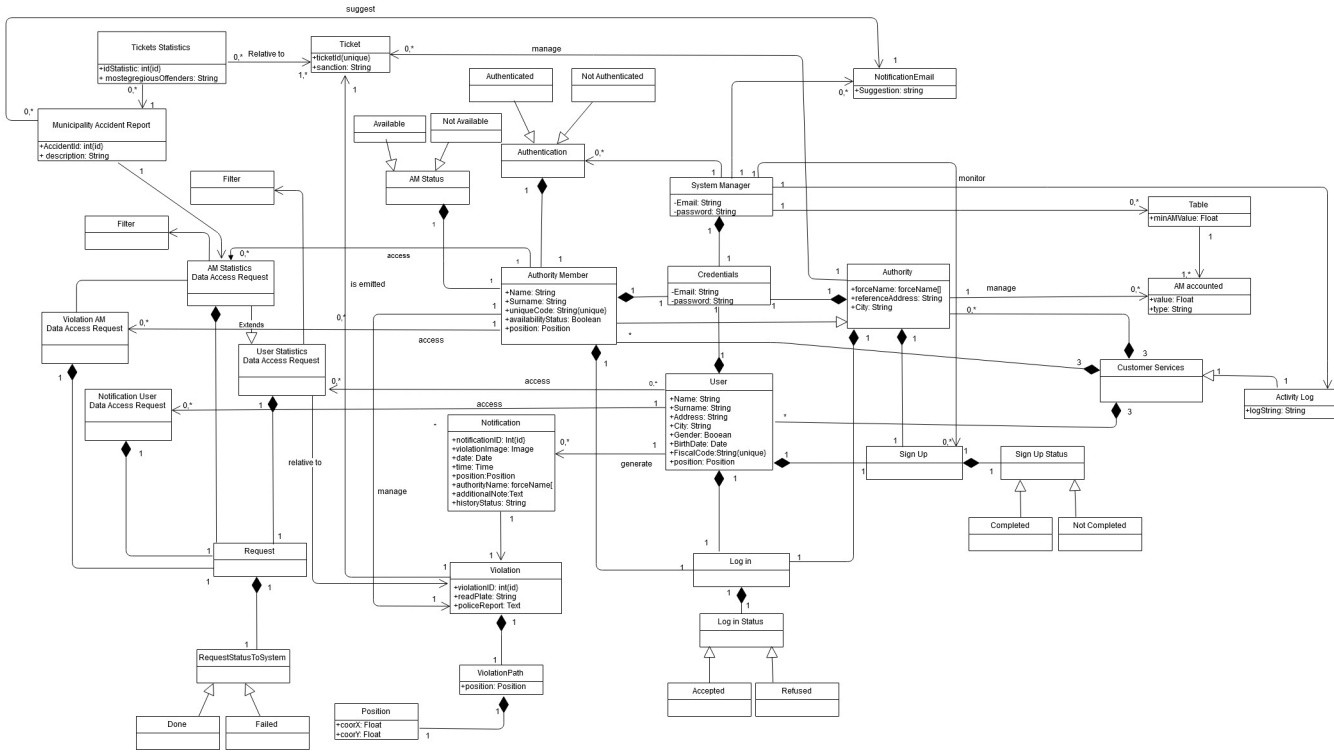

Figure 2.2: SafeStreets Class Diagram.

2.2.1 High Level Component Design

In the figure 2.3 there is presented an high level Component Diagram about interfaces provided by the Server Side of the application, Authority Web Services, System Manager Web Services and SafeStreets Mobile Services. The Authority Web Services and the System Manager Web Services interacts with the Web Application client side, while the SafeStreets Mobile Services interacts with the Mobile Application client side. An unique mobile application is provided to Customers while two different web application are intended for the System Manager and Authority.

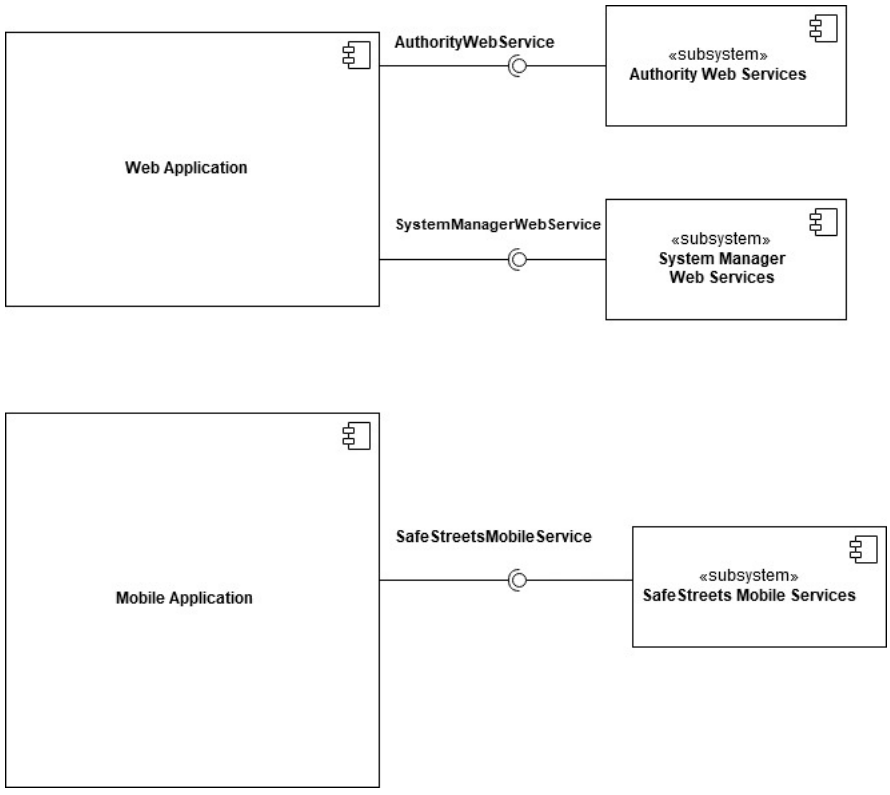

Figure 2.3: SafeStreets High Level Component Diagram.

Authority Web Services - provides the interface to permit Authority to perform Data Access Request and manage Data.

System Manager Web Services - provides the interfaces to System Manager to perform manintenance and updates tables, occurred Subscriptions and evaluating Data Access Requests.

SafeStreets Mobile Services - provides the interfaces that permit Customers to exploit all the SafeStreets services.

2.2.2 SafeStreets Mobile Services projection

SafeStreets Mobile Services subsystem represents the main core of the application from the components point of view. It is analyzed in detail describing each components with their respective functionalities. Moreover, it is composed by more components and two different interfaces that are exposed to the mobile Client: RetrieveData Service and Customer Activity Service. The components function contained in the SafeStreets Mobile services are described in the following:

- **Customer Control Activity**: it manages all the interactions and all the function calls between the subsystems in order to allow to interface the right component. Another function of this component is to handle the Login, Registration and Authentication processes for customers.

- **Profile Manager**: this component is used by Customer to manage its profile and personal data. For the User this component allows to modify informations like the E-Mail, City, Password and so on. It is important to underline for the Authority that is possible to control its employee's lists through the AM List Service interface. Furthermore, Authority Member has the possibility to set its availability status through Availability Status Service interface.

- **AM List Manager**: the scope of this component is limited to Authority view. AM List Manager component then offers all methods to add, remove or change its employee lists.
- **Notification Manager**: through this component User have the possibility to send traffic Notification, enclosing metadata through Data Service interface. On the other hand this component allows to AM to receive Notification.
- **Device Data Manager**: the main goal of this component is to retrieve data from Customer's Mobile device and to provide them to other components. Moreover the info about Customer's position is provided by Maps API.
- **Violation Manager**: this component allows AM to handle the Notification and confirm if it is a violation or not by adding more informations. The communication between this component and Notification Manager component is performed through the Violation Data Service interface.
- **Availability Status Manager**: this component permits an AM to change its availability status, in this way the dispatching performances of notification are improved. This happens cause when an AM set his availabiity status not available the system will not consider this AM for the handling of notification.
- **Ticket Manager**: if a violation is confirmed to need a ticket as fee, through Violation Ticket Service this component collect all the info necessary to perform ticket generation processes. Moreover this component has to communicate with Find Owner Plate which provides personal data of an egregious offender from vehichle's plate.

- **Statistics Data Manager**: this component shows statistics regarding emitted Tickets to Customers offering different level of visibility.

In order to pursuit their goal every components need to communicate with the DBMS.

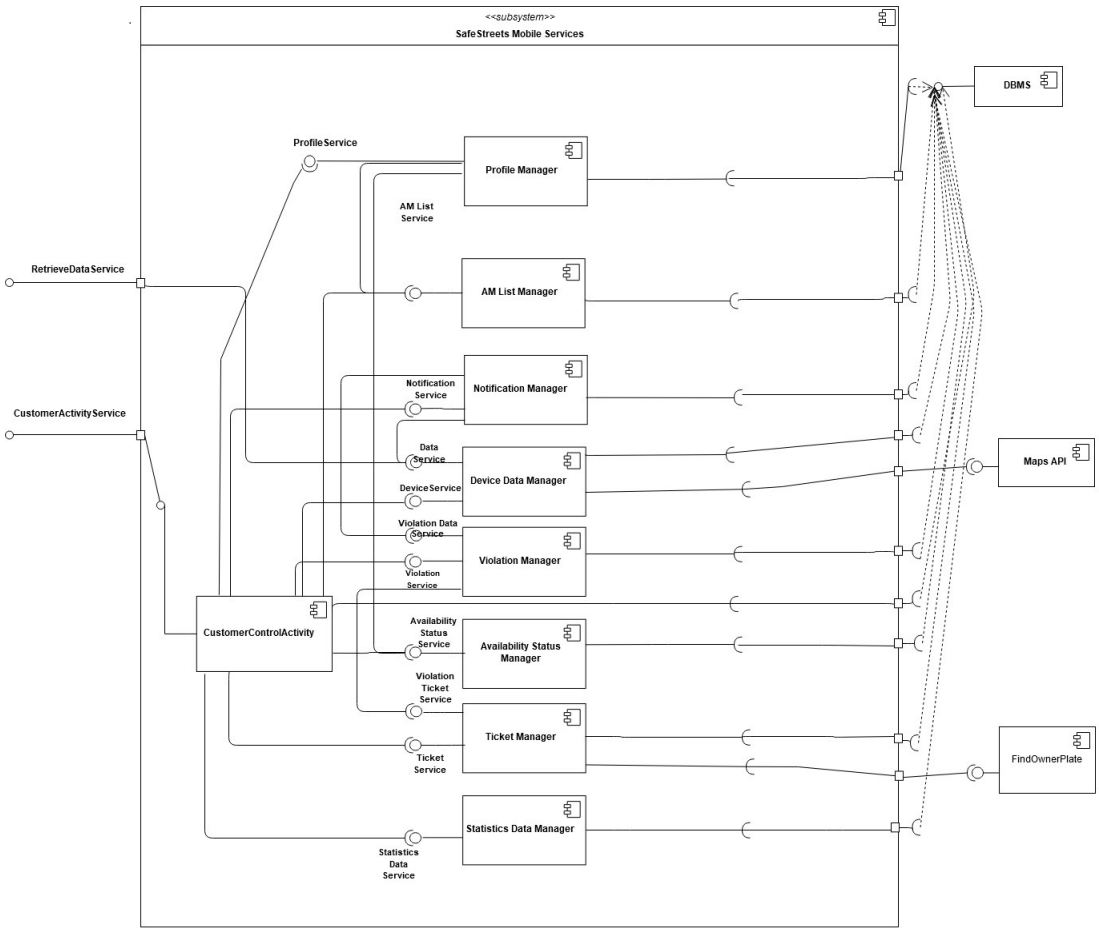

Figure 2.4: SafeStreets Mobile Services projection.

2.2.3 Authority Web Services projection

In the component diagram of the Authority Web Services subsystem are presented the principal components and interfaces. At the first sight there is the Authority Profile Manager that act as the main component and exposes the Authority Web Service interface to the Web Client in order to fullfill the purpose of the application exploiting the interfaces of the Data Request Manager, Ticket Manager and the AM List Manager.The Data Request Manager is intended to perform the Data Access Request, the Ticket Manager provides the feature for managing a ticket's emission and the AM List Manager component to organize and modify the table lists about the authority's employeees. Certainlty all the components of the subsystem needs to communicate with the DBMS.

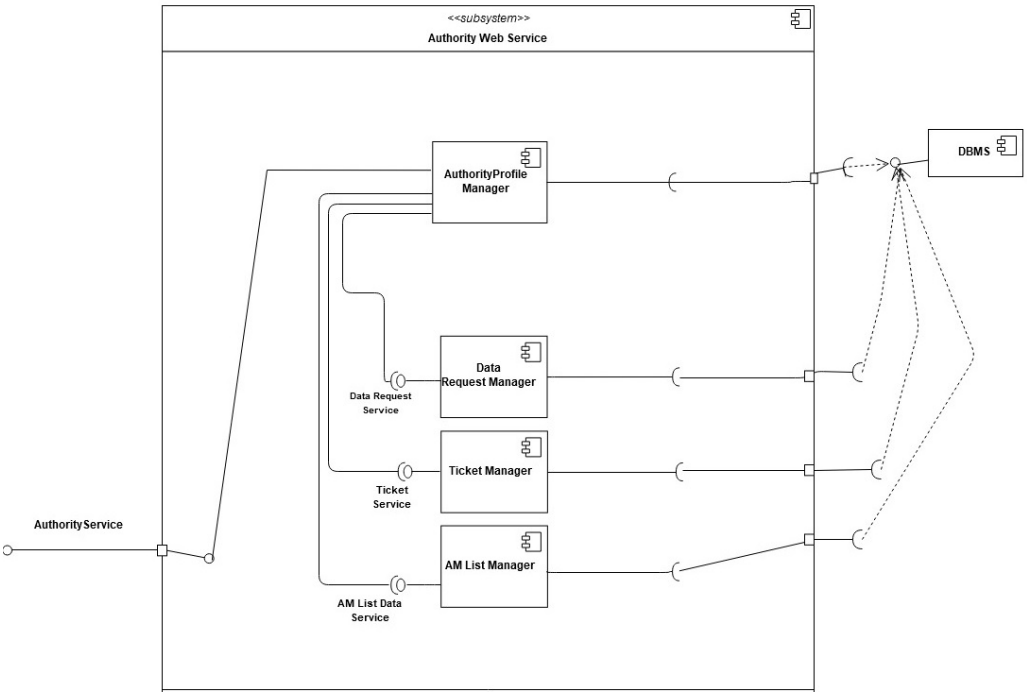

Figure 2.5: Authority Web Services projection.

2.2.4 System Manager Web Services projection

This subsystem is similar in the architectural pattern point of view as the Authority Web Services subsystem. The System Manager Activity Profile exposes the three main interfaces that enable System Manager to access the three corrispondent main components: System Data Manager, System Statistics Manager and System AM List Data Manager. Thanks to these components the System Manager can modify the AM lists, update the Statistics and manage also the activity as authorize the login, signup and authentication, control the validity of data and send through e-mail the suggestions to the Municipality. The System Manager can notify the Municipality about the improvements thanks to the infos gets by the statistics.

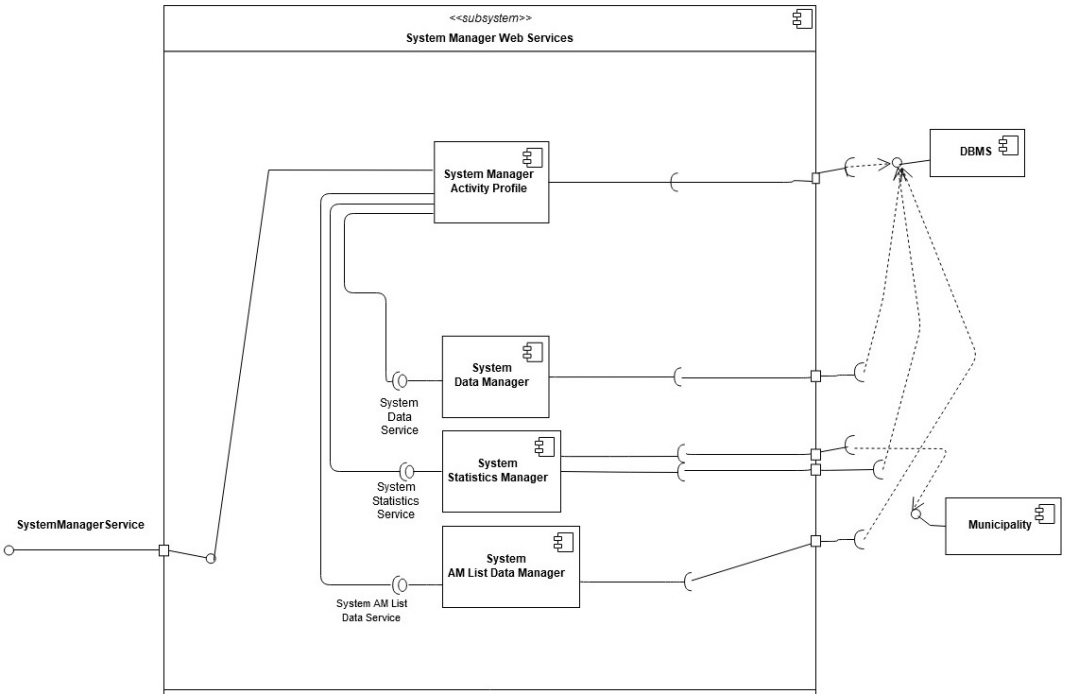

Figure 2.6: System Manager Web Services projection.

2.3 Deployment View

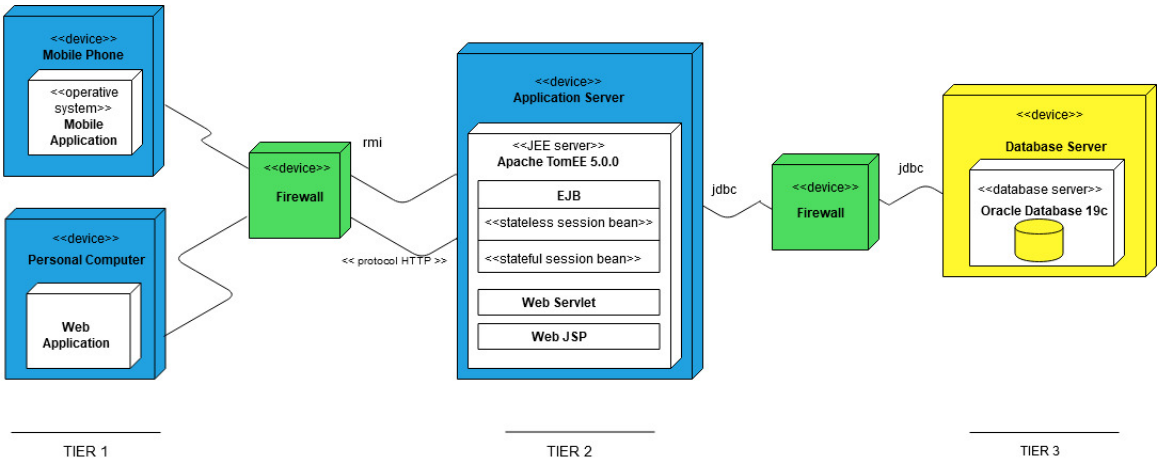

Figure 2.7: Deployment view.

There is a description of the main three tiers in Figure 2.7:

- The first tier is the *Client-side* one. In this tier there are the Mobile application and the Web Application. These two applications communicate directly to the Application Server using the RMI systems. The Mobile phone application is designed for the User, AM and Authority phones, while the Web Application is thought for Authority PC. The recommended implementation of the Mobile Application have to suits Android, iOS operative systems. For the Web Application development is recommended the HTML5,JavaScript and CSS.

- The second tier is the *System core network* tier. In this tier there is the Application Server, consisting in Apache TomEE 5.0.0, running on a Linux OS which act as a Web Server, script engine Web Servlet and Web JSP modules. It receives RMIs from the Mobile Application and HTTP messages from the Web Application.

- The third tier is the *Storage* one, which consists of a DB server, Oracle Database 19c, linked to each storage entity involved into the system. This server communicate with the Application Server through JDBC protocol.

2.4 Runtime View

In the following sequence diagrams there is the description about the interactions between the components and the requests handled by them during the execution of the main features. Is better to specify that this is a high-level description, so functions and their names may be modified or named in another way during the implementation process.

2.4.1 User Send Notification

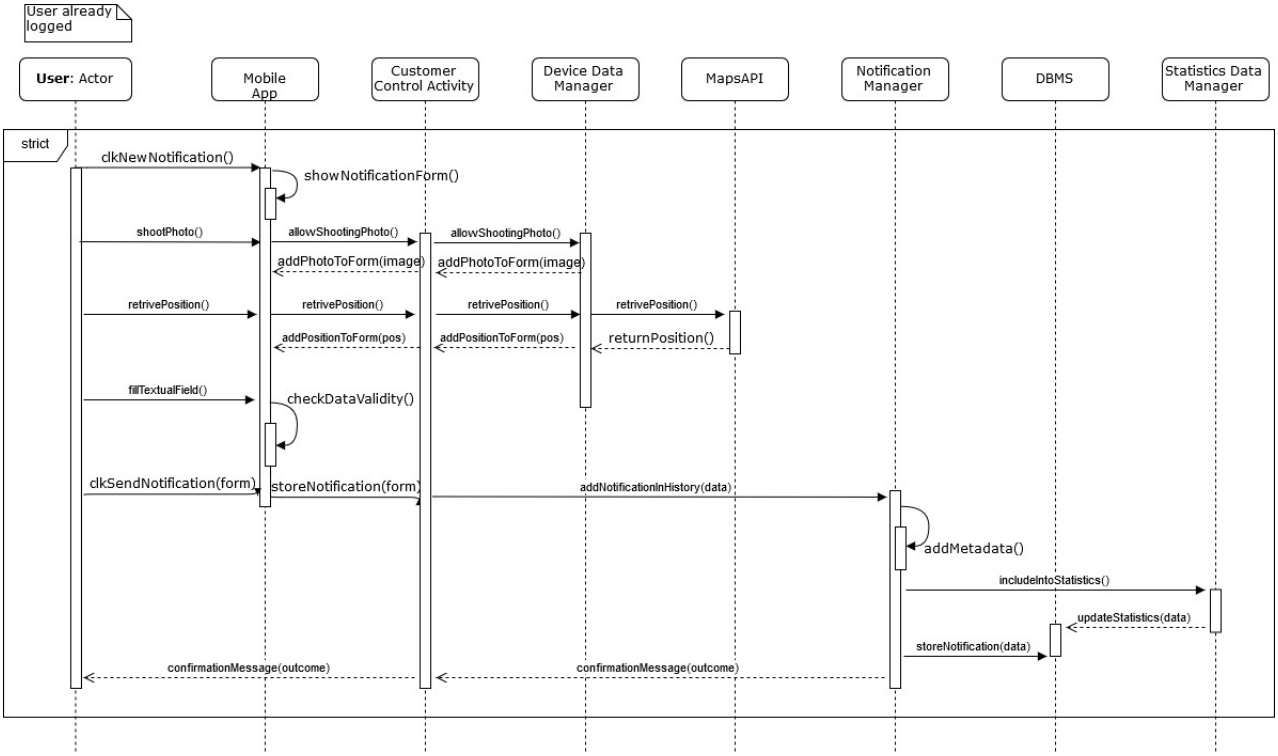

Figure 2.8: User send notification runtime view.

2.4.2 Authority Member receives Notification

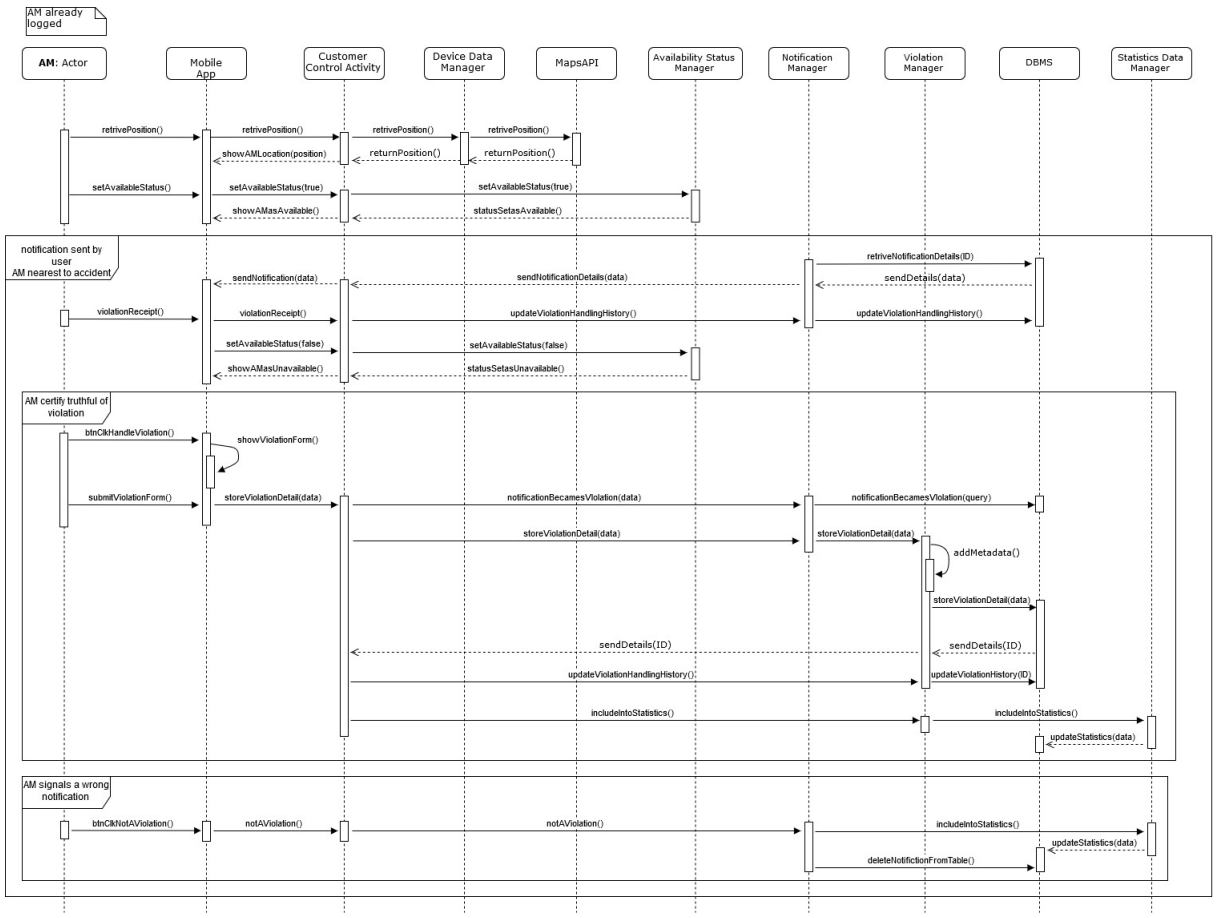

Figure 2.9: AM receives Notification runtime view.

2.4.3 Authority Member past Notification

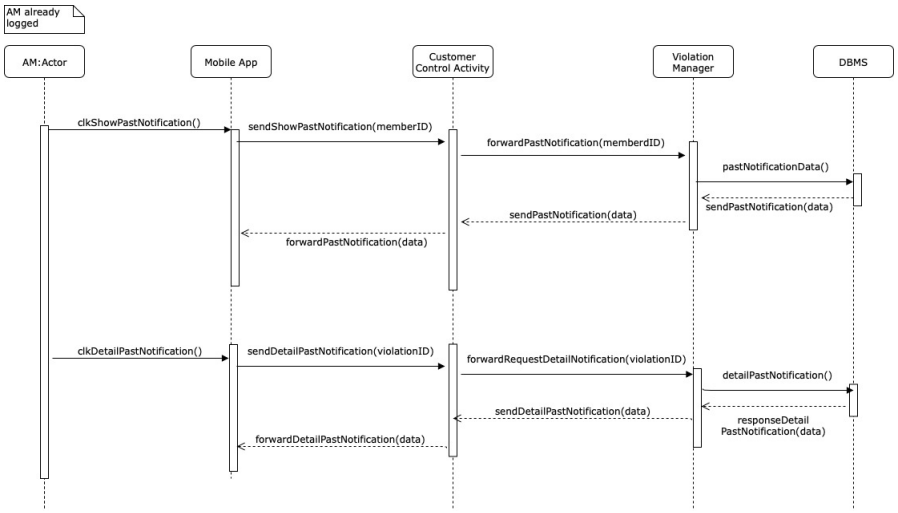

Figure 2.10: AM past Notification runtime view.

2.4.4 Authority Ticket generation process

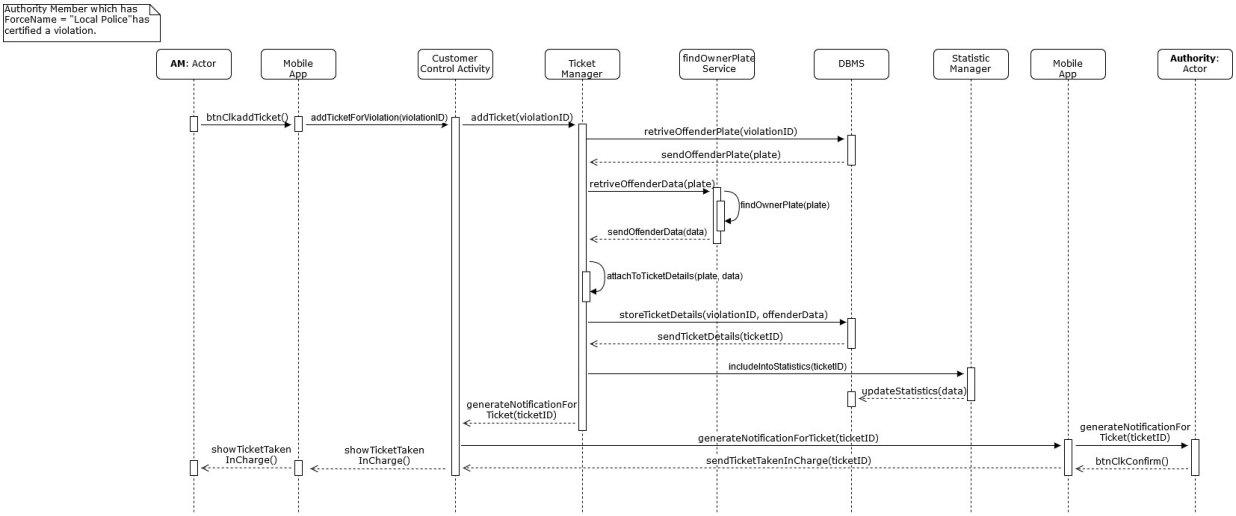

Figure 2.11: Authority Ticket generation process runtime view.

2.4.5 User accesses Statistics

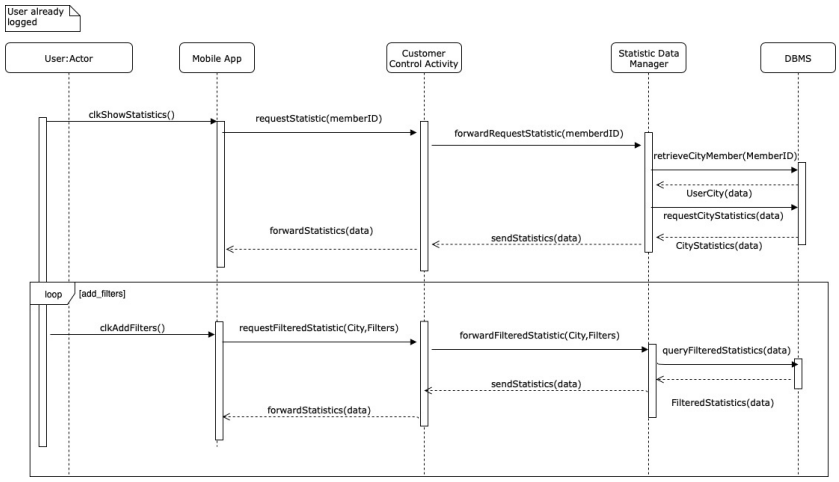

Figure 2.12: User accesses Statistics process runtime view.

2.4.6 System sends Suggestions

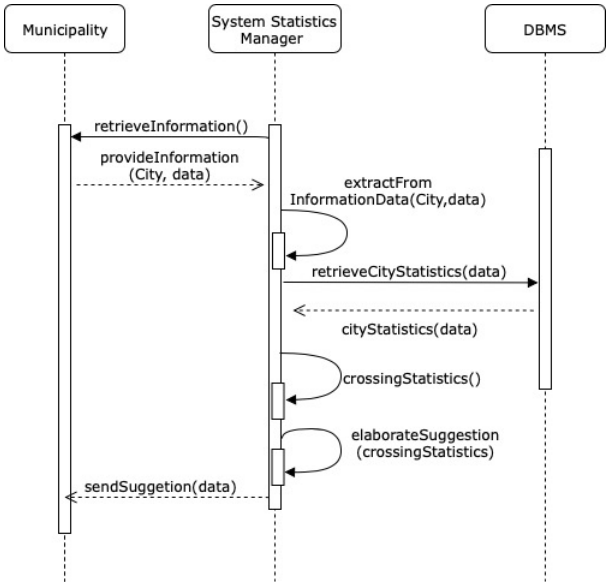

Figure 2.13: System sends Suggestions runtime view.

2.5 Component Interfaces

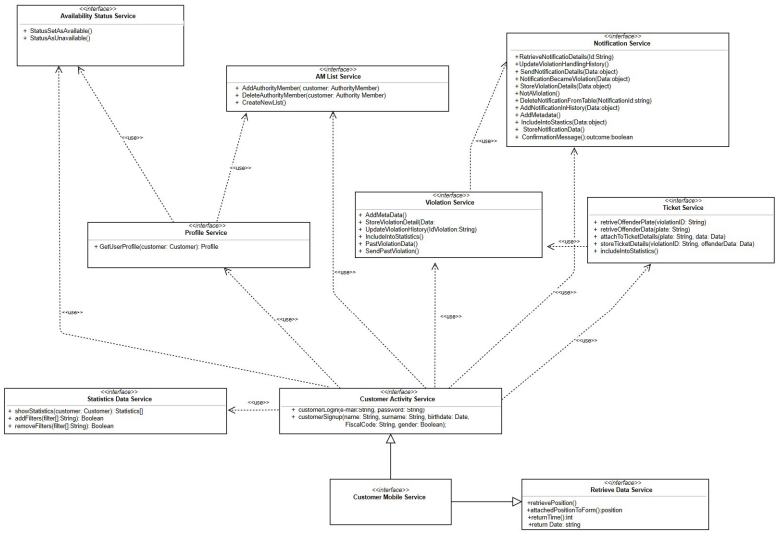

Figure 2.14: Component Interfaces Customer Mobile Service.

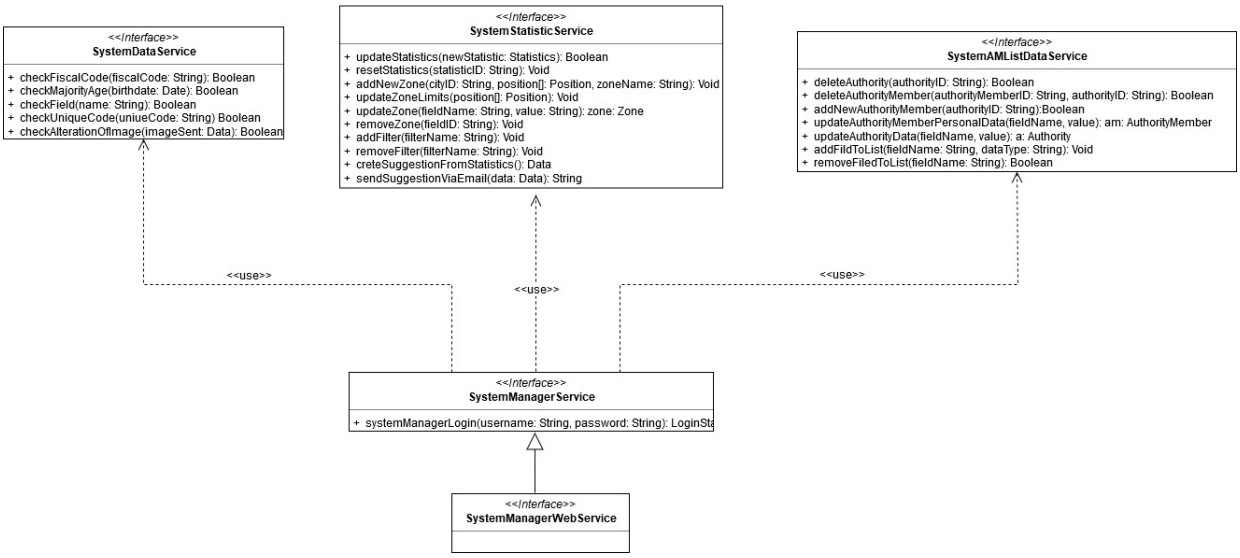

Figure 2.15: Component Interfaces System Manager Web Service.

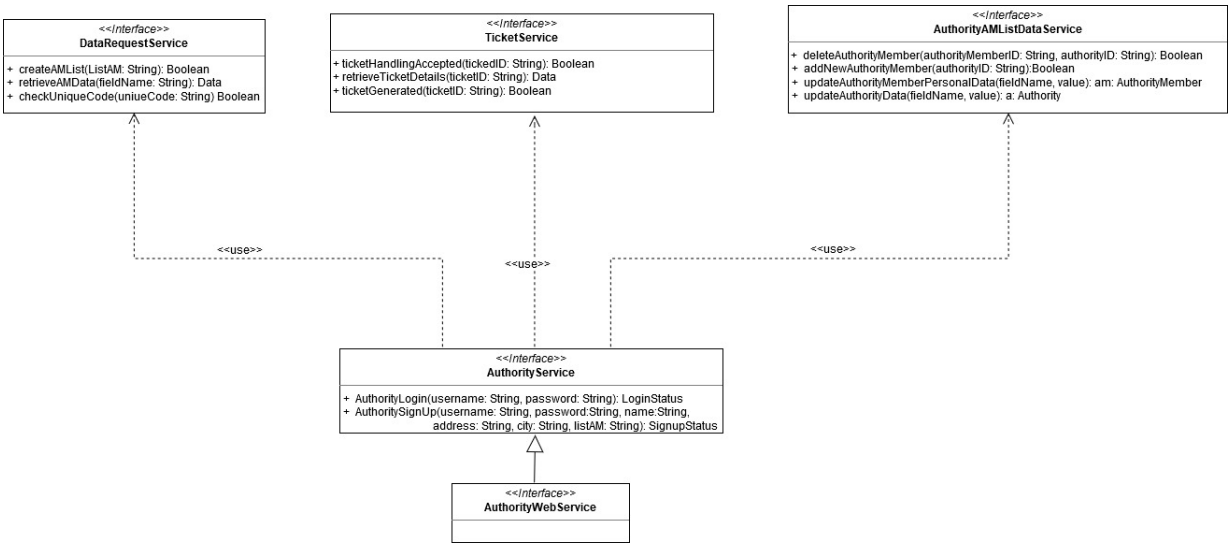

Figure 2.16: Component Interfaces Authority Web Service.

2.6 Selected architectural styles and patterns

2.6.1 Overall architecture

For the platform SafeStreets the architectural choice would be a Three Tier Architecture. In the three tier there are the different type of levels. At the highest level there is the presentation tier, which is demanded to show services functionalities and informations for Users, AMs and Authorities, through the mobile application, Authority can access through the dedicated web application too. Just in this tier User, Authority and AM can directly acess, through web and mobile app GUIs.The application tier is enclose by the presentation tier, which acts as an intermediator between the application and the data tiers. The data tier as specified in the RASD is designed to including functionalities of the Hardened Database model, with data persistence and data exposing mechanisms. The data tier also provides to the application tier an interface in order to allow the full management methods on data. In choosing a three tiers architecture there are pros and cons. The cons are like the high cost to develop this type of architecture, but in the pros there are more important aspects like the best scalability and maintainability aspects.

2.7 Design Patterns

- **Power and resources consumption constraints**

  The application should be optimized to minimize its power consumpation and memory usage during the processes. Then in every design decision should take into account the fact that mobile devices offers limited CPU, memory, storage capacity and battery durability.

- **Model View Controller(MVC)**

  User interface components are tied with data store components, to agree with MVC design pattern. This pattern is choice since mobile applications' core operation is to retrieve data from a data store and up-date the user interface with the newly requested information based on the user inputs, together with the fact that MVC is one of the most common and effective ways to avoid a dangerous level of coupling between the various parts of the whole system.

- **Improve of the reuse and maintainability improvements thanks to layered architecture**

  The concept of tiers is used to improve maintainability, for the mobile application. Also the web application is inclded into the application tier,exposing its view to make Authority able to interact with the system.

- **Facade pattern** Unified interface to a set of interfaces in the subsystem. Facade defines a high level interface that makes the subsystem easier to use.

- **Chain of responsibility pattern** Chain of Responsibility is a behavioral design pattern that lets you pass requests along a chain of han-

dlers. Upon receiving a request, each handler decides either to process the request or to pass it to the next handler in the chain.

- **Mediator pattern** The mediator pattern defines an object that encapsulates how a set of objects interact. This pattern is considered to be a behavioral pattern due to the way it can alter the program's running behavior.
- **Front Controller pattern** The pattern relates to the design of Web Applications. It provides a centralized entry point for handling requests.
- **DAO pattern** Low level separate data accessing API or operation from high level business services.
- **DTO pattern** Separation defined between the transferred data to the client with the persistent data in the DB.

3 User Interface Design

The Mobile Application Mock-Ups of the Customer Interface are showed in the SafeStreets RASD document [Section 3.1.1]

4 Requirements Tracebility

While making the design choices presented in this document, the aim is to accomplish all the goals and requirements that have been previously specified in RASD. Below are listed the design components to which requirements and goals are mapped:

[G1]: Users can be uniquely identified, thanks to the completion of the Registration Process.

[R1]-[R2]-[R3]-[R4]-[R5] User Registration: CustomerControlActivity

[R6] Confirmation on data: Profile Manager

[G2]: Authorities can be uniquely identified, thanks to the completion of Registration Process.

[R7]-[R9]-[R11] Authority Registration: Customer Control Activity

[R8]-[R10] Managing Authority Member: Customer Control Activity, Profile Manager, AM List Manager

[G3]: Authority members can be uniquely identified, thanks to the completion of Authentication Process.

[R12]-[R13] Authority member authentication: CustomerControlActivity, Profile Manager

[R14] Authority member availability status: Profile Manager, Availability Status Manager

[G4]: Allows users to notify Authority Members when traffic violations occur.

[R15]-[R16] Users login: Customer Control Activity

[R17] Notification to Authority Member: Notification Manager, Device Data Manager

[R18] Past Notification: Notification Manager

[R19] Sharing position's User: Notification Manager, Device Data Manager

[R20] Data and time from user's device: Device Data Manager

[G5]: Allows authority member to receive the notifications about traffic violations in order to increase the local security.

[R21]-[R22] Authority member login: Customer Control Activity

[R23]-[R24] Authority member manage notifications: Availability Status Manager, Notification Manager, Device Data Manager, Violation Manager

[R25]-[R26] Recognize license plate: Notification Manager,Violation Manager

[G6]: Allows end users to mine information on traffic violations that has been received and build some statistics.

[R27] User login: Customer Control Activity

[R28] Statistics request: Statistics Data Manager

[R29]-[R30] Statistics update: System Statistics Manager

[G7]: Allows authority members to mine information on traffic violations that has been received, and build some statistics.

[R31] – Authority member statistics: Statistics Data Manager

[R32]-[R33] – Stastics related to unsafe areas: Statistics Data Manager

[R34] Statistics update: System Statistics Manager r

[R35] Statistics related to vehicles: Statistics Data Manager

[R36] Authority member login: System Manager Service, Customer Control Activity

[G8]: Builds a cross information analysis between municipality's data and its self data to improve reliability of the service and suggest to municipality possible interventions.

[R37]-[R38]-[R39]-[R40] Statistics generation: System Manager Service, System Statistics Manager

[G9]: Allows municipality (in particular local police) to retrieve traffic violations in order to generate relative traffic tickets.

[R41]-[R42] System avoid data alteration: System Manager Service, System Data Manager

[R43]-[R44]-[R45] Managing ticket: Authority Service, Ticket Manager, Device Data Manager

[G10]: Builds statistics using information related to emitted traffic tickets.

[R46] Retrieve personal data owners vehicle: Ticket Manager

[R47]-[R48]-[R49] Generates statistics: System Data Manager, Statistics Data Manager

5 Implementation,Integration and Test plan

In the Implementation, Integration and Test plan section will be adopted a bottom-up approach, this allow to develop few components at a time and then to integrate them. With this procedure an easily coherent work can be done by testers and implementers. For this reason the components have to be implemented and tested with the same order. All components belonging to the same subsystem will be implemented, integrated and tested focusing only on SafeStreets Mobile Services. It is important to underline that external system components need not to be implemented and tested because they are considered reliable.

Below is reported a table containing a lists of the main features available for the Customer, respectively with the importance and implementation issues for each of them.

Feature	Importance for the customer	Difficulty of implementation
SignUp and Login	*Low*	*Low*
Visualize Profile Details	*Medium*	*Low*
Add AM as employee to Authority District	*Medium*	*Low*
Visualize Statistics about past violation and specify some filters	*High*	*High*
Notify authorities	*High*	*High*
Allow AM handle notification	*High*	*High*
Generate Tickets for violation	*High*	*Medium*

Table 5.1: Table of features.

Complying with the Table 5.1 from this point on will propose a mapping between features and components of SafeStreets Mobile Services. For each features, the order in which components are presented has to be intended as the same of the order of implementation and test phases:

- **Notify Authorities**: for this feature it is necessary to implement and produce unit-test on 'Device Data Manager', 'Notification Manager' and 'Statistics Data Manager'.
- **Allow AM handle notification**: in order to provide this feature the required components are: 'Availability Status Manager', 'Notification Manager', 'Violation Manager', 'Statistics Data Manager' and 'Device Data Manager'.
- **Generate Tickets for violation**: this is another important feature of the system and to exploit it, is necessary to test and integrate the following components,'Ticket Manager' and 'Statistic Data Manager'.
- **Visualize Statistics about past violation**: for this feature there is the need to implement, test and integrate 'Statistic Data Manager' component .
- **SignUp and Login**: this feature represents an entry condition of the system but obviously they are not core features and they are very simple, so to implement and integrated function only 'Customer Control Activity' service is needed.
- **Visualize profile details**: to guarantee this feature the 'Customer Control Activity' and 'Profile Manager' must be developed. In order to implement, test and integrate this feature it is necessary to have developed the SignUp/Registration feature.

- **Add AM as employee to Authority District**: this feature is provided thanks to the 'Customer Cotrol Activity', 'Profile Manager' and 'AM List Manager'.

Finally, the 'Customer Control Activity' component has to be implemented, unit-tested, and then integrated with all other components of the application. In the program testing phase is important to underline that to find bugs must be proceed in a parallel with the implementation. Unit testing has to be performed on the individual components and, as soon as the first versions of two components that have to be integrated are implemented, the integration is performed and tested, integration is performed in an incremental way to facilitate bug tracking.

5.1 Test Strategy

Before beginning to test anything, System Manager services must be implemented, because through this service is possible to build statistics and other important features of the application. After that it is assumed at least the 80% of the entire implementation of the application and in particular:

- 100% of the external services and interfaces with their API (Maps, Find Owner Plate...)
- 80% of the service regarding the notification and violation manager with offer the main functionalities.

5.2 Integration Phase

In the following diagrams are shown which components will go through the process of integration to be more clear. The arrows start from the component which uses the other one.

Figure 5.1: Notification Manager Integration.

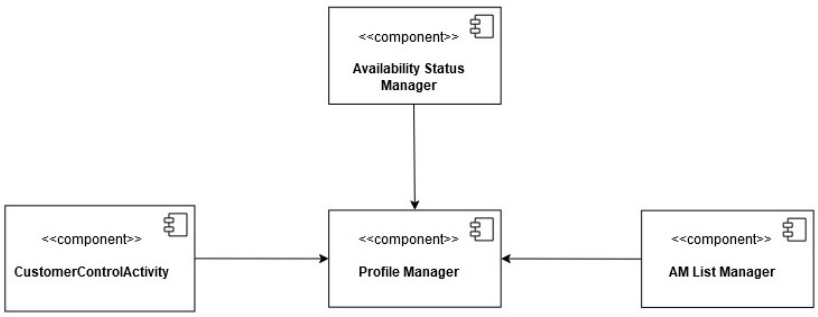

Figure 5.2: Profile Manager Integration.

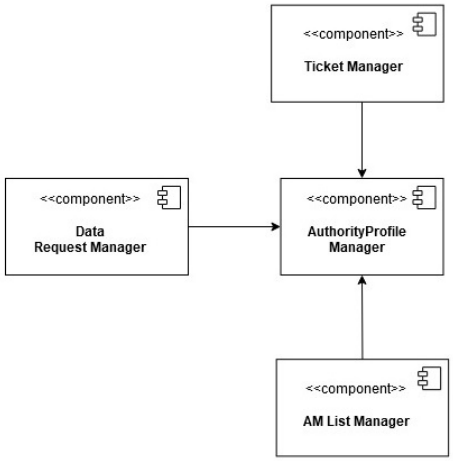

Figure 5.3: Authority Manager Integration.

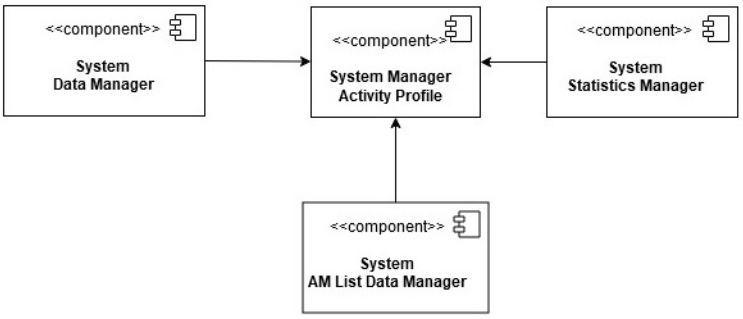

Figure 5.4: System Manager Integration.

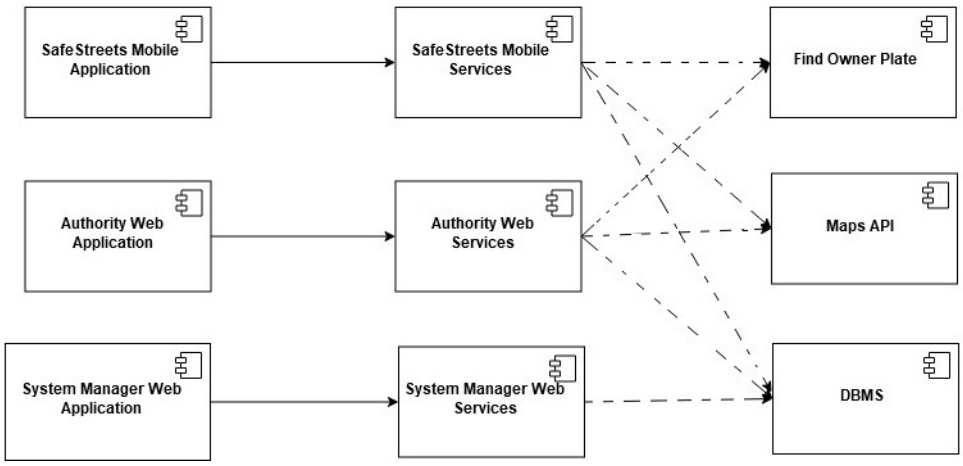

Figure 5.5: Subsystem Integration.

6 Effort Spent

Armenante Valerio	60 hours
Capaldo Marco	60 hours
Di Salvo Dario	60 hours

Table 6.1: Effort spent.

7 Reference Documents

This Design Document is directly linked to the SafeStreets RASD document.