Design Document

Abbo Giulio  
Accordi Gianmarco  
Bonetti Massiliamo

**1.Introduction**

**2.Architectural Design**

**A. Overview: High-­‐level components and their interaction**The highlevel architecture of the SafeStreets’ system is highlighted in the below Figure 1:

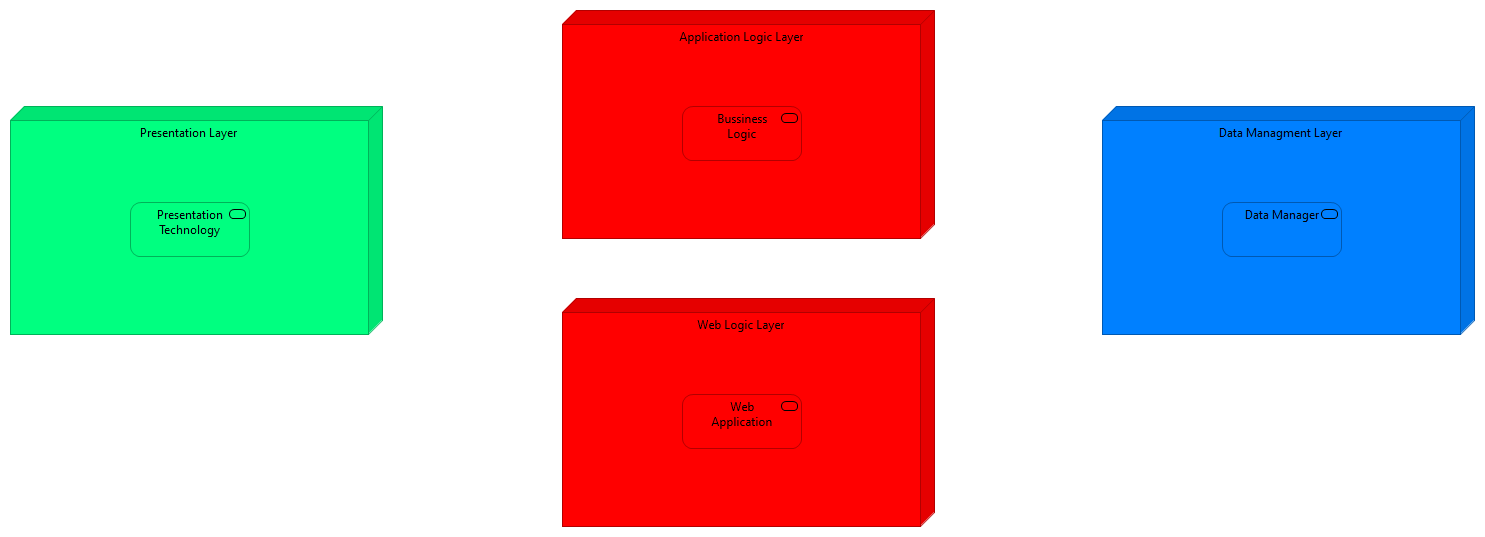


Figure 1

The defined architecture is a Four Tier architecture in which each components is briefly described below:

* *Presentation Layer* is the part that manage the visualization of the data and the possible interaction with the system in a human readability way, and also human friendly as much as possible, so this part knows how to show to the final user the results that comes from the Application or from the Web layer, also the user interact with this layer in order to interact with the system by using the application or the web access point through a browser;
* *Application Logic Layer* implements all the business logic of the Safestreets’ system, it receive all the request from the application on the users’ devices, and also the request coming from the web access, then it elaborates them, by retrieving all the information contained in the data manager, also it takes care of managing the data, by integrate them with the data provided by the municipality;
* *Data Management Layer* has the task of managing the physical allocation of the data, and to responds to: the queries that come from the application layer, and to store the data that the application layer wants to memorized.
* *Web Logic Layer* is used to respond to the web application request made by the user, but it doesn’t implement any type of business logic, so it’s only used to decouple the logic from the web visualization of the data, then all the requests made by the user through a web page are redirected to the application layer

The different levels of abstraction allow to manage different functionalities offered by the system on different machines, that need only to implement the required interface, allowing the final implementation to replicate the different machine if necessary, to scale in case of necessity and to be fault tolerant. The user isn’t an aware of the different distribution of the levels: their distribution must be as possible transparent; he only needs to communicate throw a graphic interface. The data used by the system are obtained directly through the users’ registration, but it also need to retrieve this data from the municipality data manager, so periodically the system will integrate the new data obtain by the municipality with the data present inside the data manager of the Safestreets’ System.

To allow the communication to system different from the SafeStreets’ system the architecture makes use of adapter, this allows all the other component inside the system to use the same sets of operation, but the implementation of the adapter will change during time based on the implementation of the third party it refers to.  
The different level of abstractions are decoupled as much as possible, this means that they communicate through well defined interfaces, that allows us to extend some layer if necessary or to change them, for example it we want to change the data manager is sufficient to change the component on the Application Level

**B. Component View**The Safestreets’ System is composed by a component structure defined in the below Figure 2:

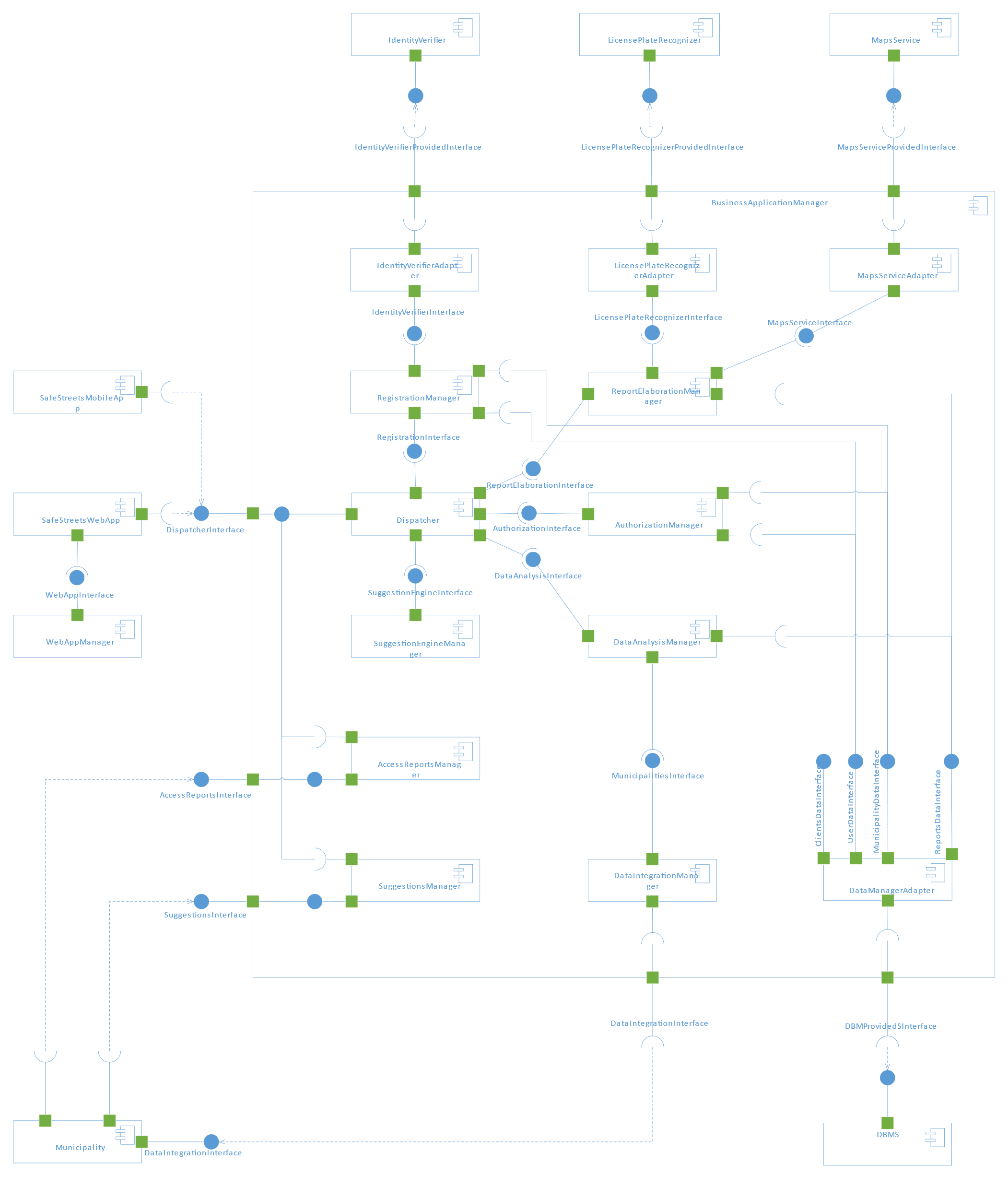


Figure 2

The figure illustrated how the components are linked between them: the figure highlight ,as already state in the high level architecture, that the module communicates among them through well-defined interface, so the figure defined which components expose an interface and which other components are interested in the same interface.  
In this chapter we will defined into details all the components that characterize out system.

The following list define the sets of tasks carried out by each single component:

* ***SafeStreets’ Mobile App*** is the component that works on the client’s device, in fact the user interact with the system also trough the App graphical interface that is similar the one developed for the web page, this component allow the user to see how to interact with the Safestreets’ System, this means that this componentsends to the *Business* *Application Manager* component all the requests made by the user, that will be dispatched by the *Dispatcher* component(in a transparent way to the user), the component also display to the user all the data that come from the *Business* *Application Manager*;
* ***Maps Service*** this component provides to the system a geographical meaning to geographical space the system works with, the *Maps Service* is not implemented in the system, but it rely on a third parties that offers this service(like for example any GIS), this service needs to provide an interface that allows out system to retrieve the address of the violations, and also to get an overlay map that we can modify in order to provide a better graphical representation to the user, and in general it helps the system to have a better understanding of the geographical space;
* ***SafeStreets’ Web App*** is another method that the user can use in order to interact with the SafeStreets’ system, it is a similar style to the Mobile App, but in this case it can be used to any kind of internet browser, this in fact will contact the *Web App Manager*  that will provide the required page, and not directly with the *Business* *Application Manager*, this component will be implemented during the development of the system*;*
* ***Web App Manager*** takes care of all the requests coming from the *Web App* of the system, this means that it has to provide all the page requested and due to the fact that it has no understanding of the business rules, he simply forward the request to the *Business* *Application Manager, and when it responds the Web App Manager* will sends back to the User’s *Web App* a web page with the results, it’s implementation will be done during the development of the system;
* ***Municipality Manager*** is the component managed by the Municipality, if the Municipality offers to us the possibility to access to their violations repositories, in order to offers a better analysis of the data, through the query done to the SafeStreets’ system, so the implemented system will rely to this component that needs to implement a standard interface defined by the system in order to correctly communicate with it;
* ***License Plate Recognizer*** also this component will not be implemented by the system, but the system will rely on a third party that offers this service, that carries out the task of recognize the plate from the report’s photos that have been sent by the user, as already state in the RASD the response that we get from this component will be trusted by the system, so it will not be subject to other verification;