Towards We-Government: Collective and participative approaches for addressing local policy challenges
Grant Agreement number: 693514

Deliverable

D3.4

Second release of WeGovNow platform prototype

Project co-funded by the European Commission within H2020-EURO-2014-2015/H2020-EURO-6-2015		
Dissemination Level		
PU	Public	■
PP	Restricted to other programme participants (including the Commission Services	
RE	Restricted to a group specified by the consortium (including the Commission Services	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Author List

Organisation	Name	Contact Information
UniTo	Alessio Antonini	antonini@di.unito.it
UniTo	Liliana Ardissono	ardissono@di.unito.it
LiquidFeedback	Jan Behrens	jan.behrens@flexiguided.de
UniTo	Guido Boella	boella@di.unito.it
UniTo	Luigi Di Caro	dicaro@di.unito.it
Infalia	Elena Dougia	elena@infalia.com
UCL	Carles Boils Gisbert	c.gisbert@ucl.ac.uk
MfC	Louise Francis	l.francis@mappingforchange.org.uk
LiquidFeedback	Axel Kistner	axel.kistner@flexiguided.de
Empirica	Lutz Kubitschke	lutz.kubitschke@empirica.com
PoliTo	Luigi La Riccia	Luigi.lariccia@polito.it
UniTo	Maurizio Lucenteforte	lucenteforte@di.unito.it
UniTo	Lucia Lupi	lupi@di.unito.it
UniTo	Roberto Micalizio	micalizio@di.unito.it
Infalia	Alexandros Mokkas	mokkas@infalia.com
LiquidFeedback	Andreas Nitsche	andreas.nitsche@flexiguided.de
UniTo	Giovanna Petrone	petrone@di.unito.it
UniTo	Adriano Savoca	savoca@di.unito.it
UniTo	Claudio Schifanella	schi@di.unito.it
UniTo	Marino Segnan	segnan@di.unito.it
LiquidFeedback	Björn Swierczek	b.swierczek@flexiguided.de
Infalia	Ioannis Tsampoulatidis	itsam@infalia.com
PoliTo	Angioletta Voghera	angioletta.voghera@polito.it

Status, Abstract, Keywords, Statement of originality

Dissemination level:	Public
Deliverable No.	3.4
Leading Partner	UniTo
Participating Partners	LiquidFeedback, Infalia, UCL, Mapping for Change, Empirica
Contractual date of delivery:	31 Jul 2017
Actual date of delivery:	10 Aug 2017
Work Package:	WP3 – Agile Development of the WeGovNow platform
Type:	Report
Approval Status:	Final
Version:	1.0

Abstract

This document describes the second release of WeGovNow platform prototype, extending the description of the first prototype (deliverable D3.3)

Keywords

Prototype, Platform, Functionalities

History

Version	Date	Reason	Revised by
1.0	12.07.2017	Creation of the initial document structure	Alessio Antonini
	26.07.2017	AreaViewer draft, annotations of needed integrations	Alessio Antonini
	27.07.2017	InputMap V2	Alessio Antonini
	28.07.2017	Area calendar, TileServer	Alessio Antonini
	30.07.2017	Trusted Marketplace & ongoing development	Ioannis Tsampoulatidis
	01.08.2017	OnToMap ongoing development	Liliana Ardissono and Adriano Savoca
	3.08.2017	Consolidate contributions, revise executive summary, fix references, fix table of contents, update introduction to section 6, conclusions	Alessio Antonini
	4.08.2017	Fix annex, add annex 6 and 7, fix changelog, fix references	Alessio Antonini
	04.08.2017- 07.08.2017	Updates in 2, 4.2, 4.6 and 5	Andreas, Nitsche Axel Kistner
	07.08.2017	Update conclusions	Alessio Antonini
	08.08.2017	Update changelogs, update section 3, 2 and 6, fix exhibit references, finalisation	Alessio Antonini
	10.08.2017	Quality assurance and submission	Lutz Kubitschke

Table of Contents

Author List 2
Status, Abstract, Keywords, Statement of originality 3
History 4
Changelogs 6
Executive Summary 8
1. Introduction 12
2. Overview of the integration work achieved so far 15
3. Description of components integrated into the platform prototype (v2) 19
3.1. GeoKey & CommunityMap 19
3.2. FirstLife 20
3.3. ImproveMyCity 21
3.4. LiquidFeedback 23
4. Level of integration achieved in relation to existing and newly developed components 26
4.1. LandingPage 29
4.2. Authentication Server 33
4.3. GeoKey and Community Maps 35
4.4. FirstLife 40
4.5. ImproveMyCity 42
4.6. LiquidFeedback 46
5. Description of platform services 48
6. Ongoing development work 69
6.1. User Management 70
6.2. Accessibility Preferences 72
6.3. Trusted Marketplace 73
6.4. AreaCalendar 77
7. Conclusion and lesson learned 79
Annexes 80

Changelogs

The present document extends the report of the "First release of WeGovNow platform prototype", deliverable D3.3, updating the state of development of WeGovNow platform. The following list summarises the major changes:

Environments

1. Renaming of https://sandbox.wegovnow.eu > https://pt1.wegovnow.eu
2. New https://pt2.wegovnow.eu, 2nd prototype

New available and updated features

1. InputMap has been revised in style and features, considering integration requirements provided by all technical partners
2. AreaViewer first version is been released as part of the landing page, it enables to explore user activities on location-based entities reported in OntoMap logger
3. UserArea first version is been released as part of the landing page, it presents the most recent user's activities reported in OntoMap logger, providing direct access to the resources within their native component
4. TileServer, first release of vector tile server based on the integration between open data and OpenStreetMap entities
5. OnToMap (which previously supported the retrieval of all the data belonging to a concept from a specified data source) has been extended in order to enable applications to select subsets of data based on the specification of filters, e.g., temporal intervals. Moreover, it has been extended to offer information retrieval at different abstraction levels (e.g., all the instances of all the concepts subsumed by a single higher-level concept). Furthermore, Open Data from Southwark have been integrated in the Open Data Container.

Platform instance settings

1. Introduced Trusted Marketplace component in 2nd prototype installation
2. Missing GeoKey/CommunityMap in 2nd prototype installation

New ongoing development for the final prototype

1. Area calendar module in FirstLife
2. Asynchronous service provision in OnToMap Logger, integration of new Open Data

Technical reports

1. Extending WeGovNow – Adoption requirements

The releases of WeGovNow prototype platform are available at:

- 1st prototype (D3.3) https://pt1.wegovnow.eu

- 2nd prototype (D.4) https://pt2.wegovnow.eu

Executive Summary

The present document extends the report of the "First release of WeGovNow platform prototype", deliverable D3.3, updating the state of development of the WeGovNow platform (see Changelogs for further details). To facilitate the reading of the current document as a self-contained report information that has already been provided in the previous version is repeated to some extent. A change log has been added summarising the major changes achieved when compared with the previous version of the WeGovNow prototype.

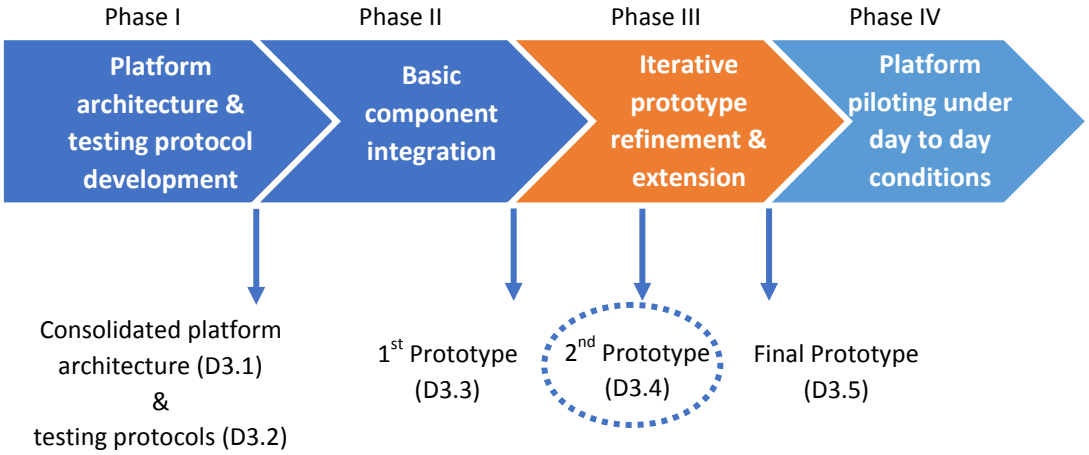
In a nutshell, WeGovNow strives for integrating a set of innovative software applications into a unified citizen engagement platform. In doing so the project aims at overcoming current limitations of existing digital tools for citizen reporting, e-participation, and communication between the citizen and the public administration. To this end, a number of civic engagement applications that have already existed prior to the project are to be integrated into a single online platform together with software components to be newly developed.

With a view to facilitating the utilisation of the WeGovNow platform within different local contexts, individual implementation instances are to be configurable in a modular manner. Also, the new WeGovNow engagement platform is to be principally extendable with further applications which potentially may be desired to be added in future implementation instances. Such a multi-faceted set of basic requirements poses a number of challenges for the design of the WeGovNow platform, in particular when it comes to

- the development cycle and design process which is to involve a variety of stakeholders at the three pilot cities participating in WeGovNow,
- and the achievement of a coherent user experiences across a diverse range of software components to be integrated into a single online platform, including existing ones and newly developed ones.

Against this background, a multi staged development approach has been adopted for the purposes of WeGovNow. As graphically summarised in the schema below, four main phases can be discerned, starting with a thorough consolidation of the initial platform architecture (D3.1) and the development of suitable testing protocols (D3.2) to be utilised throughout the subsequent development phases.

WeGovNow platform development phases.



This was followed by basic integration work (Phase II) mainly focussing on the seamless interoperation of those software components that have already existed prior to WeGovnow, e.g. in terms of a unified user management solution. The previous report (D3.3) presented a description of the first prototype of WeGovNow platform. The current report (D3.4) extends the previous report including new functionalities and ongoing development activities.

During the ongoing development phase (Phase III), the current prototype version will be further developed in an iterative manner within experimental settings, thereby involving a diverse range of stakeholders in three cities participating in the project. This will yield the final version of the fully tested platform which is then to be piloted under everyday conditions with large numbers of users (Phase IV).

The first release of the WeGovNow prototype platform is provided as a "working" environment for:

- Development purposes (https://dev.wegovnow.eu)
- Testing and presentation purposes (https://sandbox.wegovnow.eu)

To support the involvement of stakeholders at the three WeGovNow pilot sites during the next stage of platform development (Phase III) dedicated prototype implementation instances are provided to each of the three sites:

- San Donà di Piave (https://sandona.wegovnow.eu)
- Southwark (https://southwark.wegovnow.eu)
- Torino (https://torino.wegovnow.eu)

From a development angle, the WeGovNow platform prototype - as it currently stands - provides an environment of various internal services (core features) required to run a WeGovNow instance, including smooth interoperation of various

stand-alone web components (WeGovNow components) integrated into the overall platform.

A local WeGovNow implementation instance can be configured to use any combination of modular components, enabling cross-component features according to the combination of active components.

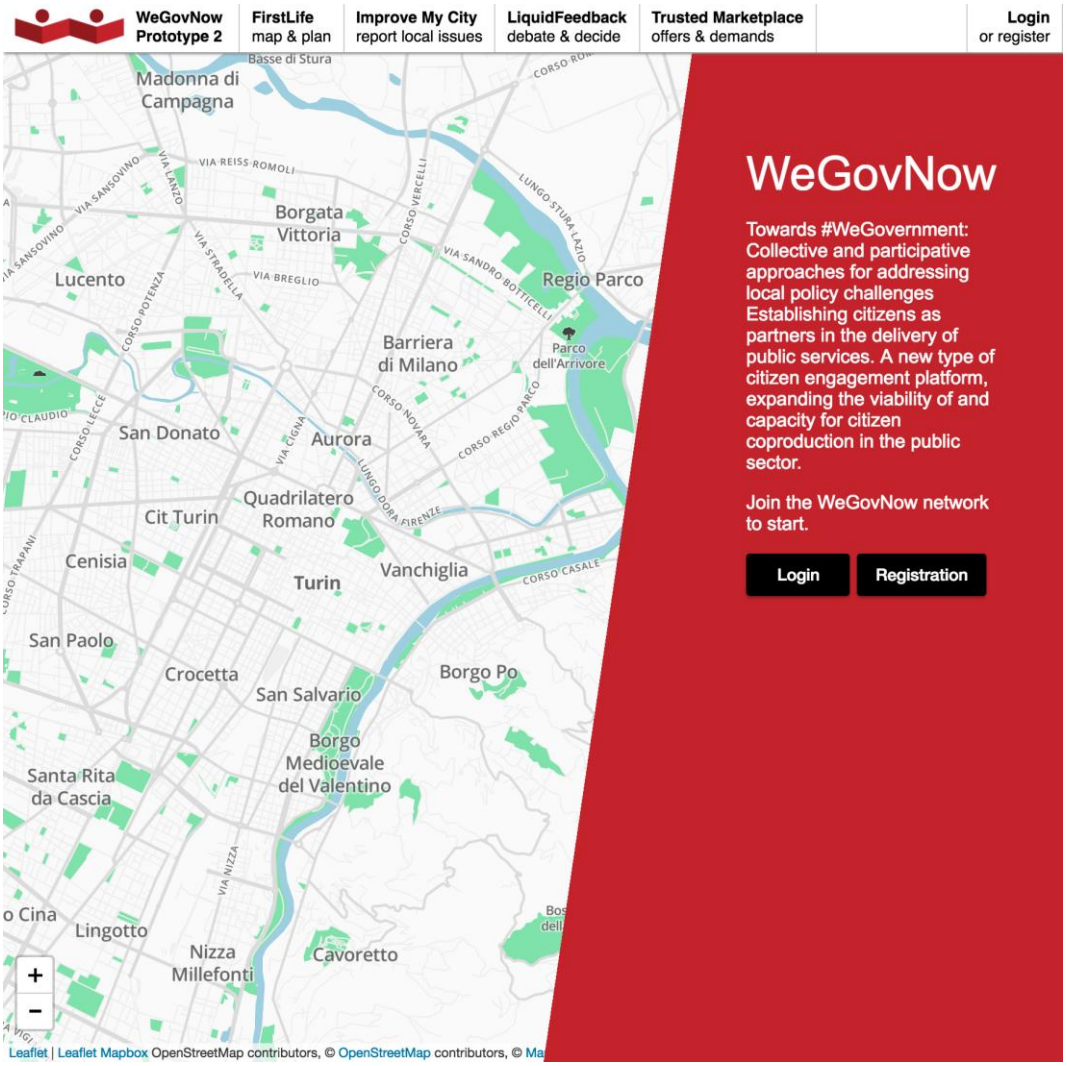
From the perspective of the end users, the current version of the prototype platform provides a number of functionalities through the integration of components that have existed prior to WeGovNow already, including:

- Map-based social networking functionalities realised by means of integrating FirstLife;
- Citizen reporting functionalities realised through the integration of ImproveMyCity;
- Opinion formation and voting functionalities realised through the integration of LiquidFeedback.
- Community mapping functionalities realised by means of integrating Community Map and GeoKey

When it comes to interface integration in particular, the approach adopted for the purposes of WeGovNow can be characterised as a gradual convergence toward a well-documented design framework (Material Design https://material.io/), a repository to collect common solutions to shared design issues, and a set of features supporting the propagation of user setups, customisations and elements among WeGovNow components. This approach provides a pragmatic solution to the need for providing a coherent user experience across the platform while at the same time taking into account that changing the interface design of existing components is much more complex and resource consuming (and thus costly) when compared with newly developed components. Also, the design approach adopted for the purpose of WeGovNow helps in keeping the platform as open as possible towards additional components that may be desired to be integrated after the ending of the project duration.

The WeGovNow platform is accessed by the users through a common web interface. The screen shot below shows the common user entry point to the current version of the prototype platform. It is expected that this version will undergo further changes according to the stakeholders' input to be received from various engagement activities to be conducted at the three pilot sites during the next development stage (Phase III).

Screen shot of the web interface of the v2 platform prototype



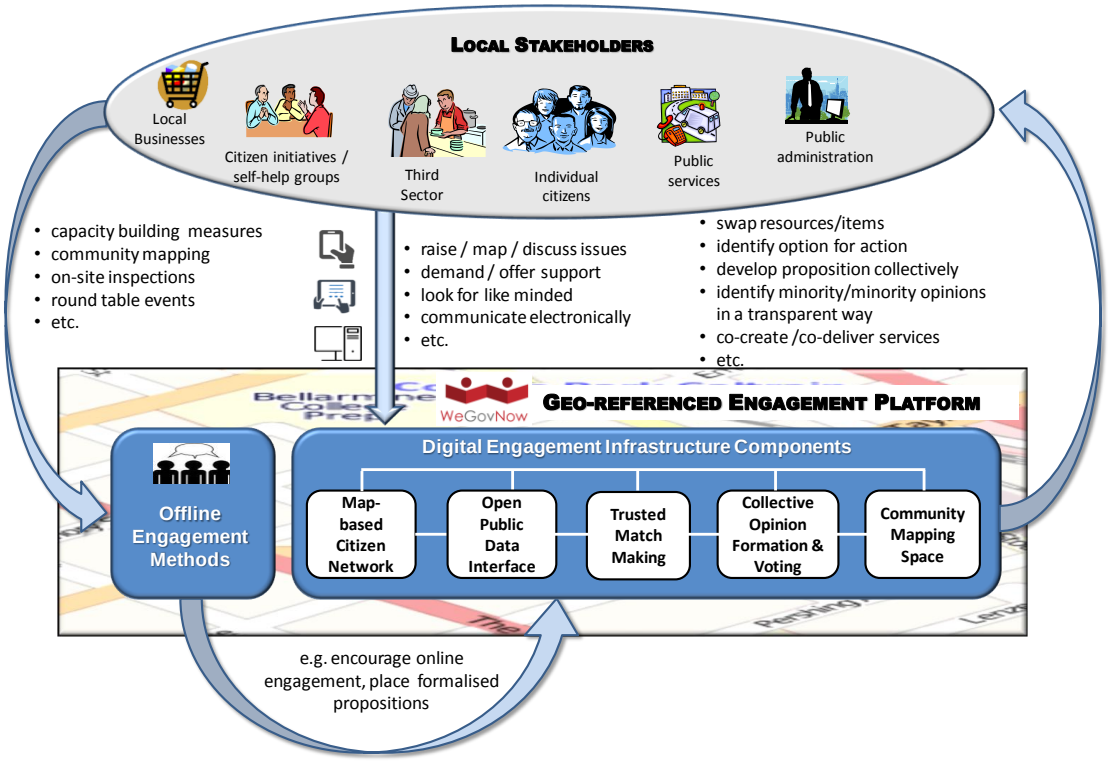
Beyond the functionalities provided to the end users that have been implemented so far, as mentioned earlier, a range of internal software services have been developed for the current release of the platform. These enable seamless interoperation of the different platform components and customisation of local platform implementation instances, e.g. in terms of Style Service, an Application Discovery Service, a Unified Authentication System, a Centralised User Profile, a Centralised Activity Logger and others. Finally, development work that is currently ongoing in relation to newly developed platform components to be implemented with the next release of the prototype platform is outlined in the current document.

1. Introduction

The present document extends the report of the "First release of WeGovNow platform prototype", deliverable D3.3, updating the state of development of WeGovNow platform (see Changelogs for further details).

In a nutshell, WeGovNow strives for integrating a set of innovative software applications into a unified citizen engagement platform. In doing so the project aims at overcoming current limitations of existing digital tools for citizen reporting, e-participation, and communication between the citizen and the government as graphically summarised in Exhibit 1. To this end, a number of civic engagement applications that have existed already prior to the project are to be integrated into a single online platform together with various software components to be newly developed.

Exhibit 1: Graphical overview of the WegovNow platform and its utilisation.



With a view to facilitating further utilisation of the WeGovNow platform within different local contexts, individual implementation instances are to be configurable in a modular manner. Also, the new WeGovNow engagement platform is to be principally extendable with further applications which potentially may be desired to be added in future implementation instances. Such a multi-faceted set of basic requirements poses a number of challenges for the design of the WeGovNow platform, in particular when it comes to

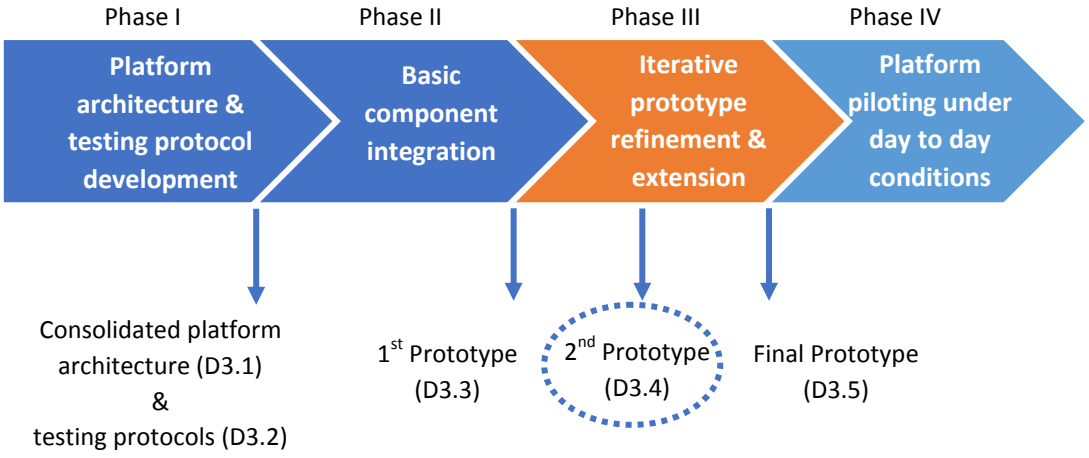
- the development cycle and design process which is to involve a variety of stakeholders at the three pilot cities participating in WeGovNow,

- and the achievement of a coherent user experiences across a diverse range of software components to be integrated into a single online platform, including those that have existed prior WeGovNow already and those that are newly being developed within the project.

Against this background, a multi staged development approach has been adopted for the purposes of WeGovNow. As graphically summarised in Exhibit 2, four main phases can be discerned, starting with a thorough consolidation of the initial platform architecture (D3.1) and the development of suitable testing protocols (D3.2) to be utilised throughout the subsequent development phases.

This was followed by basic integration work (Phase II) mainly focussing on the seamless interoperation of those software components that already existed prior to WeGovnow, e.g. in terms of a unified user management solution. The previous report (D3.3) is paper based description of the first prototype of WeGovNow platform. The current report (D3.4) extends the previous report including the new functionalities and ongoing development activities.

Exhibit 2: WeGovNow platform development phases.



During the ongoing development phase (Phase III), the current prototype version will be continued to be further developed in an iterative manner within experimental settings, thereby involving a diverse range of stakeholders in three cities participating in the project. This will yield the final version of the fully tested platform which is then to be piloted under everyday conditions with large numbers of users (Phase IV).

The following Chapter 2 of this report provides an overview of the integration work achieved so far. This is followed by a description of the functionalities provided by the current version of the WeGovNow platform prototype through the integration of existing software components into the overall platform (Chapter 3). Next, a

description of the level of integration achieved so far is provided when it comes to both, components that have existed prior WeGovNow already and those components that are specifically being developed for the purposes of WeGovNow (Chapter 4). Moreover, core platform features implemented as internal software services with the current prototype release are described (Chapter 5). Finally, ongoing development work feeding into the next version of the platform prototype is outlined (Chapter 6). To support a better understanding of some of the information provided throughout the main body of this report, supportive material is provided in the annex.

2. Overview of the integration work achieved so far

As mentioned above, the first release of the WeGovNow platform prototype (v1 prototype) largely resulted from the integration of existing software solutions into the overall platform according to an architectural model developed during the start-up phase of the project (D3.1), under the premise of building a seamless environment specifically for:

1) user authentication
2) component navigation

The first prototype enabled users to log in from any component and navigate through the platform, keeping a user authentication session via single-sign-on ("auto login") functionality implemented by each component.

From the perspective of integrating the various software components, the first prototype enabled to use the individual components' APIs by means of an UWUM OAuth 2.0 token, generated by users' authentication. Moreover, the first prototype enabled the implementation of functionalities within the WeGovNow ecosystem, based on the following patterns:

1. data integration (JSON/GeoJSON document exchange)
2. inclusion of dynamic html snippet
3. feature integration (HTML/CSS/JavaScript module integration)
4. integrated data based on the common logging system

From a development angle, the WeGovNow platform prototype as it currently stands provides an environment of services (core features) required to run a WeGovNow instance, including a set of stand-alone web components (WeGovNow components) integrated into the overall platform. A local WeGovNow implementation instance can be configured to use any combination of modular components, enabling cross-component features according to the combination of active components.

Within the context of WeGovNow, the integration challenge concerns multiple levels, i.e. integration of data, functionalities and interfaces. Each of the individual software applications that have been integrated into the overall platform so far has its inner coherence in terms of data, functionalities and interfaces. However, in terms of WeGovNow platform components they must be harmonised with a view to add value beyond their implementation as standalone solutions. The full harmonisation of all three layers - data, functionalities and interfaces - will finally be the result of the ongoing WeGovNow platform development work strand (Phase III). Development work will be continued to be informed by ongoing use case consolidation work (WP2) pursued at each of the three pilot cities, thereby involving a variety of stake holders that are envisaged to utilise the WeGovNow platform for their purposes (Exhibit 1).

From the perspective of the end users, WeGovNow platform can provide the following functionalities:

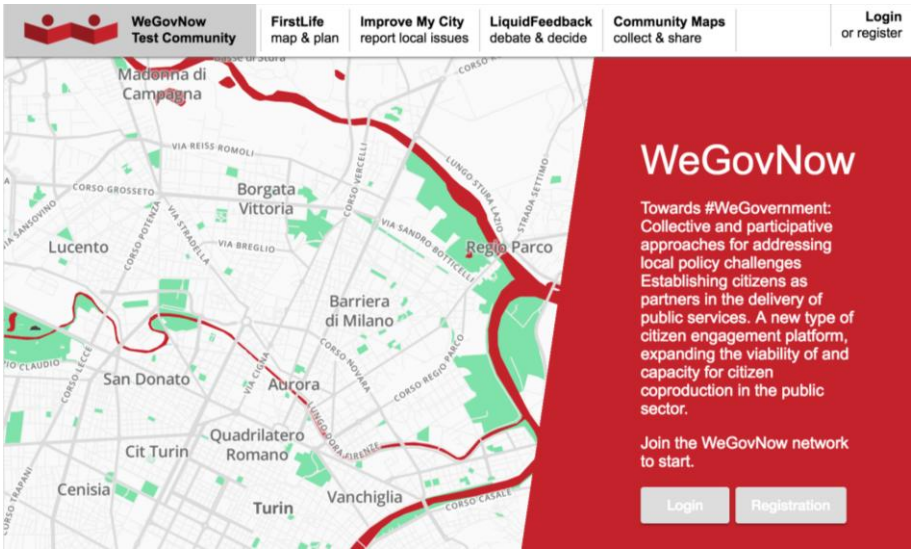
- Map-based social networking functionalities realised by means of integrating FirstLife. A documentation is available at: http://tutorials.firstlife.org/ (Italian)
- Citizen reporting functionalities realised by the integration of ImproveMyCity. A documentation is available at:
    - https://www.youtube.com/embed/CyHArXYCqnA?rel=0 (English)
    - http://www.improve-my-city.com/features (English)
    - https://wegovnow.improve-my-city.com/handbook (English)
- Opinion formation and voting functionalities realised by the integration of LiquidFeedback. A documentation is available at:
    - http://liquidfeedback.org/ (English)
    - http://www.interaktive-demokratie.org/files/downloads/LF-Information-Kit-EN.pdf (English)
    - http://principles.liquidfeedback.org/ (English)
    - http://dev.liquidfeedback.org/trac/lf/ (English)
- Search, profile management and users' activities
    - Trusted Marketplace, first prototype (English)

When it comes to interface integration in particular, the approach adopted for the purposes of WeGovNow can be characterised as a gradual convergence toward a well-documented design framework (Material Design https://material.io/), a repository to collect the solutions to shared design issues, and a set of features to support the propagation of user setups, customisations and elements among individual WeGovNow components.

Adopting a common design standard (Material Design) as a goal to be achieved by the end of the prototype development phase and continuous collaboration in addressing common issues can be seen as the operational approach towards software integration adopted for the purposes of WeGovNow. This approach is a pragmatic solution to the need of providing a coherent platform in a development context within which changing the design of existing components is costly, time consuming and not as straight forward as for new components. Approaching the interface integration as a path and not as a requisite also enables the WeGovNow platform to remain principally open towards developers and companies who may be interested in contributing further platform components in the future, thereby avoiding unreasonable demands such as to re-design existing software as mandatory condition.

At the same time, WeGovNow provides many services to support the integration of platform components, and the choice of Material Design was not at least made due to the availability of extensive documentations, examples and frameworks based material supporting to the development of material interfaces in the most popular web languages and frameworks.

Exhibit 3: Screen shot of the WeGovNow platform protytpe (v1) home page.



From the point of view of the end users, the current release of the platform prototype is accessed through a common web interface which will be described in more detail later in this document (Section 4.1). Exhibit 3 provides a screen shot of the common entry point to the WeGovNow platform prototype as it is currently available to the users. It is expected that this version will undergo further changes according to the stake holder input to be yielded by the various engagement activities conducted at the three pilot sites during the current development stage (Phase III).

The development environment (https://dev.wegovnow.eu) is meant to be constantly updated with the alfa version of the software, it is specifically required to test the communication between components and to develop integrations between components.

1st and 2nd prototypes versions correspond to two of three milestones of the project and development life cycle. The platform versions are frozen “as they are”, to keep track of the evolution of WeGovNow platform.

The first platform prototype provided a proof of concept of the integration of existing components. The second prototype tests the capability of run multiple instances of WeGovNow and to test the integration of a new component (Trusted Marketplace).

In the next months, to support the further involvement of stakeholders at the three pilot sites during the next stage of platform development (Phase III) dedicated platform instances are provided to each of the three trial sites.

1st prototype settings

The settings of the first prototype of WeGovNow platform (https://pt1.wegovnow.eu) includes all existing components (landing page in Exhibit 3):

- GeoKey/CommunityMap https://wegovnow-cm.geokey.org.uk/
- FirstLife https://pt1.wegovnow.firstlife.org/
- ImproveMyCity https://pt1-imc.infalia.com/
- LiquidFeedback https://wegovnow-pt1.liquidfeedback.com/

The entry point of the platform is the first prototype of LandingPage https://pt1.wegovnow.eu/.

2nd prototype settings

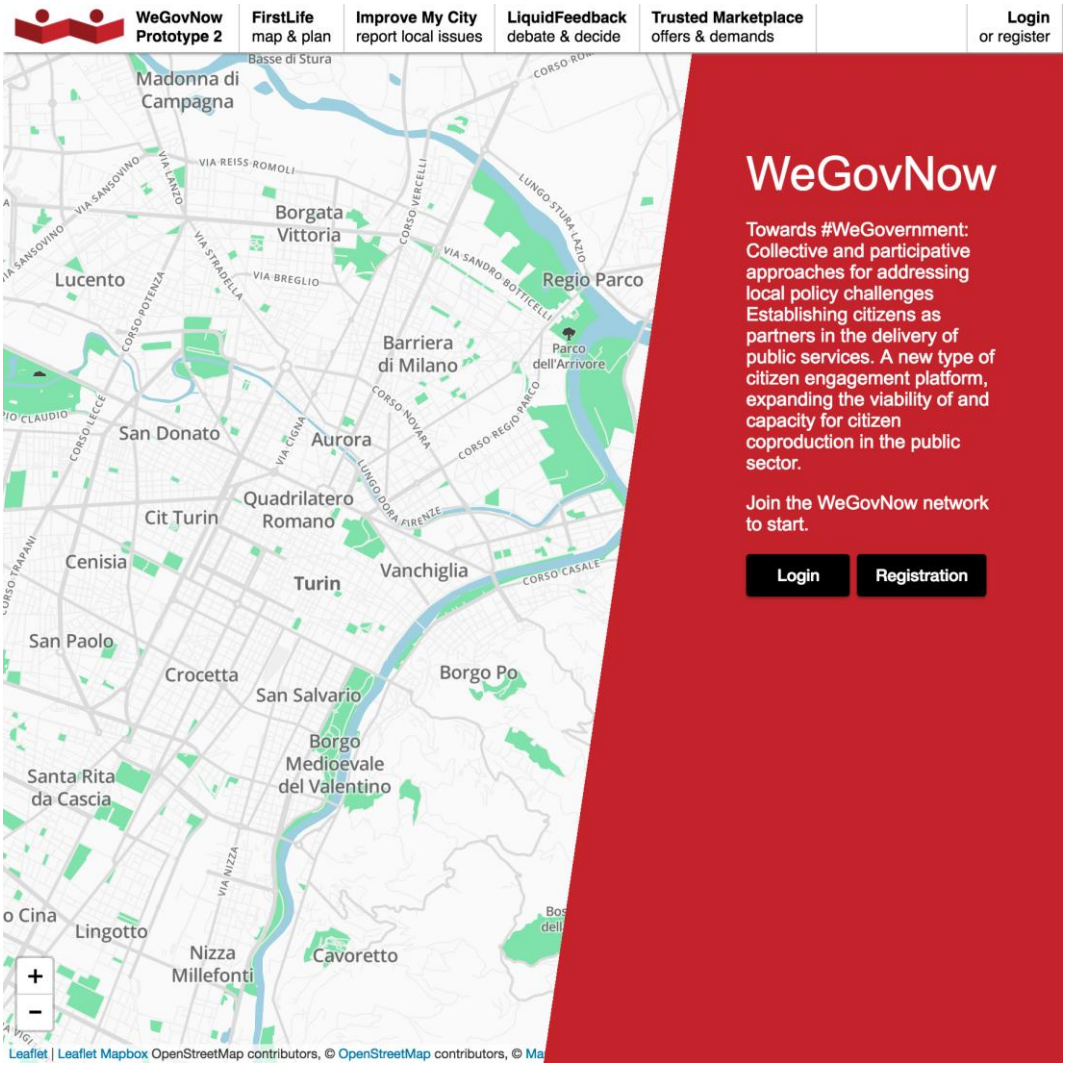
The installation of the second prototype (https://pt2.wegovnow.eu) of WeGovNow (landing page in Exhibit 4) introduces a new component:

- Trusted Marketplace 1st prototype version

The first version of Trusted Marketplace does not provide full functionalities yet but it is fully integrated in WeGovNow (see Section 4).

On the other hand, the second prototype does not include GeoKey/CommunityMap components, due to a technical problem preventing to run multiple instances of GeoKey that has emerged during the setup operation of the platform instance.

Exhibit 4. Screen shot of the landing page of 2nd prototype of WeGovNow platform.



3. Description of components integrated into the platform

WeGovNow is an environment integrating various stand-alone components. In general, the capabilities provided by the overall platform are related to the use of individual components in conjunction with new possibilities provided by the coexistence (integration) of those components. The following sections describe the features provided by components included in WeGovNow platform.

3.1. GeoKey & CommunityMap

On one hand, GeoKey provides local communities with a web-based infrastructure to collect, share and discuss local knowledge. You can use it to setup your own mapping project with your community and to collect, visualise and analyse data using the tools of your choice.

Key Features

- **Setup up your data structures:** You decide what data your community should collect by setting up categories and attributes.
- **Decide who can access, contribute and moderate data:** Use user groups and data groupings — a predefined subset of all data contributed to the project — to define, which users can access, contribute and moderate data.
- **Add photos, videos and audio files to your contributions:** Enrich your contributions by uploading photos and videos to each of your contributions.
- **Discuss:** Comments on each contribution allow you and your community to discuss observations you have made, offer suggestions and links to other web-based material.
- **Connect your app using our API:** Use our public http://geokey.org.uk/docs/ to build and connect applications for data collection, analysis and visualisation.

On the other hand, we have the web-map based application, CommunityMaps. This application allows users to collect new data, visualize the existing data, discuss and attach media to the contributions added to the projects previously created on GeoKey using the public REST API.

One real example could be that one municipality wants to do a survey to know where the inaccessible wheelchair local businesses are in the neighbourhood. The municipality will define the data structure on GeoKey (e.g. defining the name of business and type of business, selection criteria to assess the level of access etc.). Southwark will decide who can contribute to the project as well.

Once the municipality makes the project live, any citizen will be able to add contributions and comments to any of the contributions on the project. Indeed, users will be able to attach media files such as audio, video and pictures.

GeoKey has different extension which allow project administrators to export data. In this example, the municipality will be able to download the data in different formats (GeoJSON, KML or CSV), to upload data to existing projects, and additional functions which you can find on ExCiteS Group repository (https://github.com/ExCiteS/).

3.2. FirstLife

FirstLife is the newest software among the existing component of WeGovNow, and at current time is yet to be widely adopted outside pilots and projects. The maturity level of FirstLife is therefore inferior to others and it is only currently being finalised as service.

FirstLife addresses the need of a coordination/collaboration platform for urban activities and urban services. One of the strong aspects of FirstLife is to provide a working environment shared among the heterogeneous urban actors, such as

institutions, organisations, charities, privates, etc. Moreover, FirstLife supports the self-organisation of citizens and informal groups.

Mapping, organization and documentation of distributed events

FirstLife supports a set of users' actions on distributed entities, such as events.

Users can add events on the map, linking them through a set of tags allowing to search for all the elements connected to a specific topic (for example: the name of a festival used as tag provide the possibility to create automatically a tag-map) or grouping all the events in an "initiative" that is a relation among entities. Different users can add new events on the map and decide to link them to a specific initiative, or contribute in describing and sharing real-time updates about planned, on-going or past events by adding a post in the entity card.

Users can create groups and sub-groups to manage in a light way coordination activities among the organizers and other people involved in the events (supporters, public, places owners, etc.). News and posts can be shared within each group, but keeping the information public for all the platform users.

Users can add stories to each event or group, sharing their experience about a specific event of the initiative in general, building a live and collaborative report of the events.

Users can easily retrieve all contents related to a specific initiative by selecting a period of interest on the timeline in the main interface and using FirstLife as an interactive archive of city activities.

Create groups to organize joint activities in multiple places

Citizen organizations operating in one or more neighbourhood of the city can create groups to coordinate and document the activities related to a specific project, such as managing a few urban gardens, with their schedules of gardening activities.

Each group can contain a list of places with their cards, events, news and stories. Each member of the group can add new content to the group (events or news) and visualize the map of the group.

General users, not members of the group, can explore the group content and choose to follow the group and receive notification about updates and new-shared content.

3.3. ImproveMyCity

Direct citizen-government communication & collaboration

ImproveMyCity enables residents to directly report, to their public administration, everyday local issues about their neighbourhood such as; discarded trash bins, faulty

streetlights, and broken tiles on sidewalks, illegal advertising boards, but also ideas and suggestions.

The reported issues are automatically transmitted to the appropriate department in public administration to schedule their settlement more quickly and more efficient comparing to the tradition paperwork based procedures. Reporting is feasible through a friendly web based frontend that adopts a map-based visualisation, which makes reporting a user-friendly and intriguing process. The management and routing of incoming issues is performed through a backend administration infrastructure that serves as an integrated management system with easy to use interfaces.

Key features frontend

- Accurately positioned issues: By offering a map to facilitate citizens in determining the exact location of their issue with geocoding and reverse geocoding support
- Categorised on issue nature: By urging citizens to select one of the pre-specified categories reflecting the municipality departments and its internal organisation
- Easily submit new issues: By asking citizens to provide only the information necessary to locate and resolve the issue, such as title, description, location and category
- Photo enabled: By allowing to attach one or more images on the spot for describing the issue, easily with drag ‘n drop
- Commented and voted: By offering the mechanisms to post comments or vote for issues that have been submitted by other citizens
- Filtering and search: By allowing to see only your own issues, or issues of specific categories or specific status or a combination of them

Key features backend (concerns the Municipality employees and administrators)

- Browse effectively: Issues are presented on the city map, as an ordered list but also in a single-issue page displaying the full set of submitted details. Only assigned issues are visible according to the user group that each officer belongs
- Track pending issues: Issues are automatically routed not only to the appropriate department but also to the inbox of the responsible officer. Detailed rules defined who gets notification and on what action
- Provide direct feedback: Provide written feedback to the citizens giving non-standard explanations for each specific case with a very friendly user interface that guides officers to take action
- Distributed responsibilities: Assign one or more officers per category and/or user group and split the administration effort across the municipality departments

- Monitor progress: Resolve issues and inform citizens by email automatically or through a progress indication bar (Open - Acknowledged - Closed, etc.)
- Tailored to each authority/Municipality: Fully customize the system in terms of user rights, number and nature of categories, notification rules and localization settings. The hierarchy of the Municipality can be projected to the application no matter the complexity.

A typical example would be:

A citizen, on her way to work, realised that the pedestrian crossing lights next to the primary school are not synchronised correctly and more over they do not let enough time for children to cross the street safely. The citizen does not have time to go in person to the municipal police to submit an official complain, and she is afraid that this action will be very time consuming and very bureaucratic. Still she really needs to inform the officials about the issue because it is not an obvious problem and maybe no one else noticed that already. She realised the problem after many times of observing the children and the lights in that specific crossroad. Using ImproveMyCity to report the issue is very easy. She spots the road by typing the address and she fine tune the marker by zooming and drag to the correct place on map. She also uploads a photo of the pedestrian crossing and selects the Municipal Police category from the available ones. She submits the issue and immediately receives an informative email that her issue submitted successfully.

On the other hand, the Municipality employees who are set to get notifications when a new issue is submitted to their department, are aware of the new post. They read about it and unmoderated since it concerns a real problem and the status of the issue is set to acknowledged. The issue is now publicly available for other citizens to see thus promoting transparency. Moments later, the issue receives 15 upvotes and 2 comments by other citizens in the neighbourhood saying that they also are aware of the problem but they didn't know where to report about. It proved to be an issue that concerned a lot of people but left unreported for months. All these votes, eventually speed up the process of having the issue fixed in the next 2 days already. The status of the issue has changed in every step so that the citizens are aware of the progress. The problem is solved and anyone could see how fast the Municipality responded.

3.4. LiquidFeedback

LiquidFeedback allows for opinion formation on a given issue with a discussion (deliberation) phase before the actual voting.

Citizens can start initiatives (proposals) and seek support among their fellow citizens. Other citizens can suggest improvements or start alternative initiatives. Considering nobody has enough time and knowledge for every issue, votes can be dynamically

delegated by topic. Delegations are transitive and can be revoked at any time. Liquid Democracy is sometimes referred to as »Delegated« or »Proxy Voting«.

LiquidFeedback doesn't rely on a request commission and doesn't need a moderator. Instead, all participants gain equal rights in a scalable structured discussion process where it is ensured that minorities gain a fair share of representation and that even individuals may put up their proposals for discussion. The system is designed in such way that noisy minorities won't harm other minorities in the discussion process.

Predefined rules and timings ensure that plans on decision processes are made public in time. Decisions are made by recorded vote only, and all voting-relevant data in LiquidFeedback is made available to all participants in both human– and machine–readable form. This enables a transparent decision-making process and ensures that participants can verify the voting procedure.

LiquidFeedback doesn't ask predefined questions but encourages participants to suggest alternatives. A sophisticated voting system is facilitated to allow participants to express their opinions without necessity of tactical considerations. Its mathematical properties avoid vote–splitting and allow that similar proposals don't harm each other.

A typical example would be:

A citizen (initiator 1) suggests to allow BBQ in a given park and creates a new initiative in the dedicated subject area for this park. As there are no other initiatives on this matter, a new issue is automatically created. Other citizens support the idea. One citizen writes a suggestion and asks to consider the fire danger and to restrict the BBQ use to suitable areas. Some citizens share the concern and support the suggestion. Initiator 1 accepts the suggestions and uses the input map to mark areas in the park he thinks may be suitable.

Another citizen (initiator 2) likes the general idea but comes up with an alternative initiative. She suggests a picnic area with tables, benches and fixed fire places. She suggests a specific location in the park and marks it on the map. The alternative soon becomes as popular as the original initiative.

Another citizen suggests another location because that location would be closer to existing public restrooms. However, initiator 2, does not like the idea and therefore does not accept the suggestion. As a consequence, the citizen (now initiator 3) starts another alternative initiative and marks the proposed location on the map.

Over the next weeks all initiatives in this issue undergo a process of improvements by considering and eventually accepting suggestions. Citizens can support more than one initiative.

Only a few days before the foreseen discussion phase ends, a citizen (initiator 4) adds another alternative basically combining initiative 2 and 3 by suggesting picnic areas in both places.

Finally, all initiatives with the necessary support, e.g. 10 percent of the people interested in the issue, become voting options. The citizens can vote using a preferential order, starting with their favorite initiative. This way citizens can agree to a second choice without fearing this could harm their favorite. Likewise, citizens are not encouraged to tactical voting, e.g. exclusively voting in favor of the second choice just because they think their favorite has no chances.

LiquidFeedback's preferential voting (Clone Proof Schwartz Sequential Dropping) determines a collective preference.

4. Level of integration achieved in relation to existing and newly developed components

As mentioned earlier, the WeGovNow platform is to include a number of software components that have existed prior of WeGovNow already, augmented with various components that are to be specifically developed within the project. Exhibit 5 provides an overview of the components implemented so far. As can be seen from the table, the current prototype includes all already existing features and some new features developed so far. Each component provides a set of features to users, and to other software components to enable new integrated features. A subset of modules provides the core features of the platform (WGN Core).

The current status of the integration of individual components into the overall platform can be evaluated by the adoption of WeGovNow core features: the features required to let stand-alone components to be integrated in the platform:

1) UWUM integration, the adoption of the WeGovNow authentication server and the correct use of OAuth 2.0 protocol
2) Single-sign-on: "Auto login", check of user session and seamless login in the software components as the users switch from one component to another
3) APIs over UWUM token, the component APIs can be invoked using an UWUM token
4) NavigationBar service, the component is dynamically retrieving the NavigationBar as HTML or JSON from UWUM
5) AreaViewer integration, the map-based summary view is done using AreaViewer, which incorporates the information from the common logger
6) Log User's activities, the software component is integrated with the logger and is sending information about user's actions
7) InputMap integration, the software component collects user's location inputs using InputMap
8) User Profile Service, the software component uses stores and retrieves the common user information in UWUM
9) Material Design, the software component follows material design guidelines
10) Styling service, the software component retrieves the platform styles dynamically from UWUM
11) WCAG 2.0 AA, the software component is compliant with the guidelines about accessibility as prescribed by WCAG 2.0 level AA
12) Accessibility Setup, the software component retrieves dynamically the user accessibility setups from UWUM

Exhibit 5. WeGovNow software components, modules and features.

Software Components	Modules	WGN Core	Features	Pre WGN	1st	2nd
Community Map	Core		GeoKey data visualisation	X	X	X
			Contribution management	X	X	X
GeoKey	Core		Project management	X	X	X
			SocialMedia publishing		X	X
			Data import	X	X	X
FirstLife	**InputMap**	X	Map-based input		X	X
	TileServer	X	Vector Tile Server			X
	Core		Custom Maps	X	X	X
			Collaborative Initiatives		X	X
			Crowdmapping	X	X	X
			Area calendar			
ImproveMyCity	Core		Issue reporting	X	X	X
			Issue management	X	X	X
			Organisation management	X	X	X
LandingPage	**AreaViewer**	X	Map-based view		~	X
			Data export			
	User's Area		User's fast access			
LiquidFeedback	**UWUM**	X	Authentication		X	X
			Style		X	X
			Application discovery		~	X

			Dynamic client registration		X	X
			Navigation bar		X	X
			User profile & settings via API		~	X
	Core		Delegation management	X	X	X
			Collaborative proposals	X	X	X
			Voting	X	X	X
			Independence of "subject area memberships"		X	X
			Storage of geo data and geo-spatial indices and searches		X	X
OnToMap	**Logger**	X	Activity logger		X	X
	Core		Open data	X	X	X
			Semantic search	X	X	X
			Data integration			~
Trusted Marketplace	**User Management**	X	Accessibility preferences		~	X
			User profile management		~	X
	Core		Activity timeline			X
			Reputation mechanism			~
			Offer & Demands			X
			Data export			

The level of integration of the existing components is currently restricted to the core features released so far and to the relevance of the core features in the mechanisms and purposes of the software components (see Exhibit 6).

Some components introduced by the WeGovNow Consolidated System Architecture (D3.1), including the so called Trusted Marketplace, are still under development.

Further details about modules and components that are currently under development are provided in Chapter 6.

Exhibit 6. Level of integration of WeGovNow software components.

	LP	UWUM	PSW	GK	CM	FL	IMC	LF	TM*
UWUM integration	X	X		X	X	X	X	X	X
Single-sign-on/auto login	X	X		X	X	X	X	X	X
APIs over UWUM token	X	X		X		X	~	X	~
NavigationBar service	X	X			X	X	X	X	X
AreaViewer integration*	X	-	-	-	-				X
Log User's activities		~				X	~	~	X
InputMap integration	-	-	-	-	-	X		X	X
User Profile service		X							
Material Design	X	X			~	~	X	~	X
Styling service		X					~	X	~
WCAG 2.0 AA							~		~
Accessibility Setup*									X

(X): archived, (~): work in progress, (-) not applicable, (*) not released yet,
LP: LandingPage, UWUM: Unified WeGovNow User Management is provided by LF (includes Authentication Server, User and Application Management),
PWP: Profile and settings wizard is provided by TM, GK: GeoKey; CM: CommunityMap, FL: FirstLife, IMC: ImproveMyCity, LF: LiquidFeedback, TM: TrustedMarketplace

4.1. LandingPage

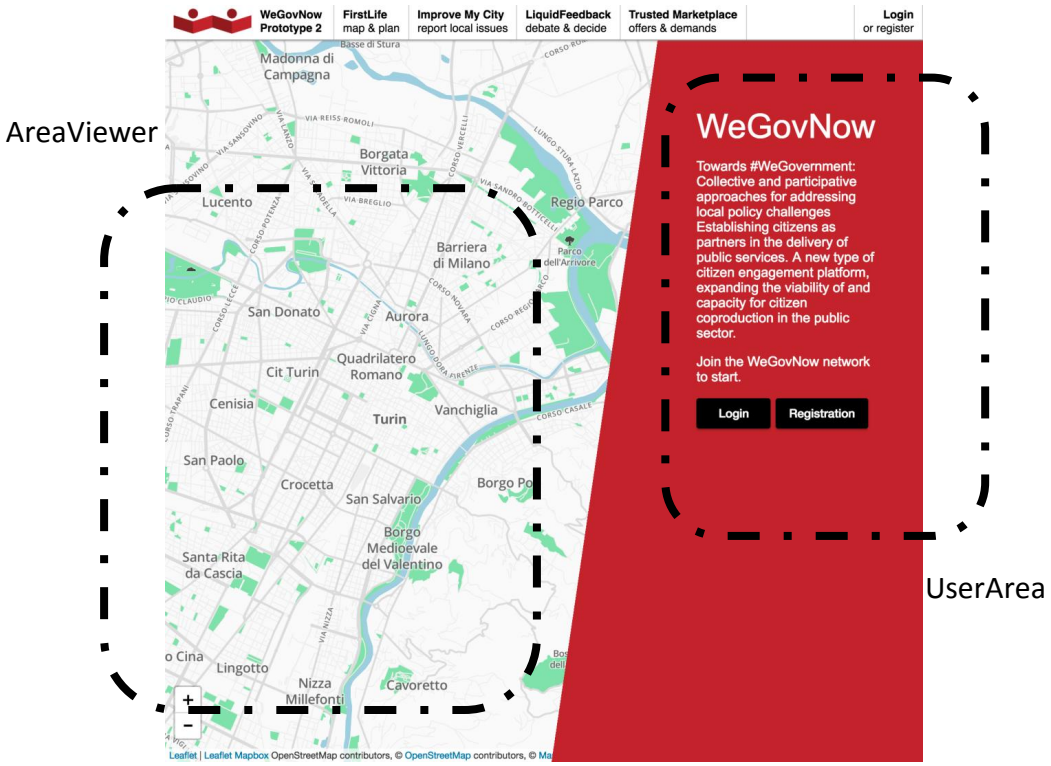
The LandingPage is the entry point of WeGovNow platforms. It has the role of presenting a summary of the status of WeGovNow platform. LandingPage is not an existing component, it was introduced during the development of the general architecture of WeGovNow (D3.1 Consolidated System Architecture).

The LandingPage has two main components (see Exhibit 7):

a) AreaViewer, a map-based view integrating information about the activities in the current instance of WeGovNow
b) (*to be implemented*) UserArea, a menu to collect direct links to user's last activities in the different WeGovNow components

Furthermore, the LandingPage will include the styling personalisations requested for each instance of WeGovNow.

Exhibit 7. LandingPage includes the NavigationBar, AreaViewer (the map on the left) and UserArea to collect references to user's activities.



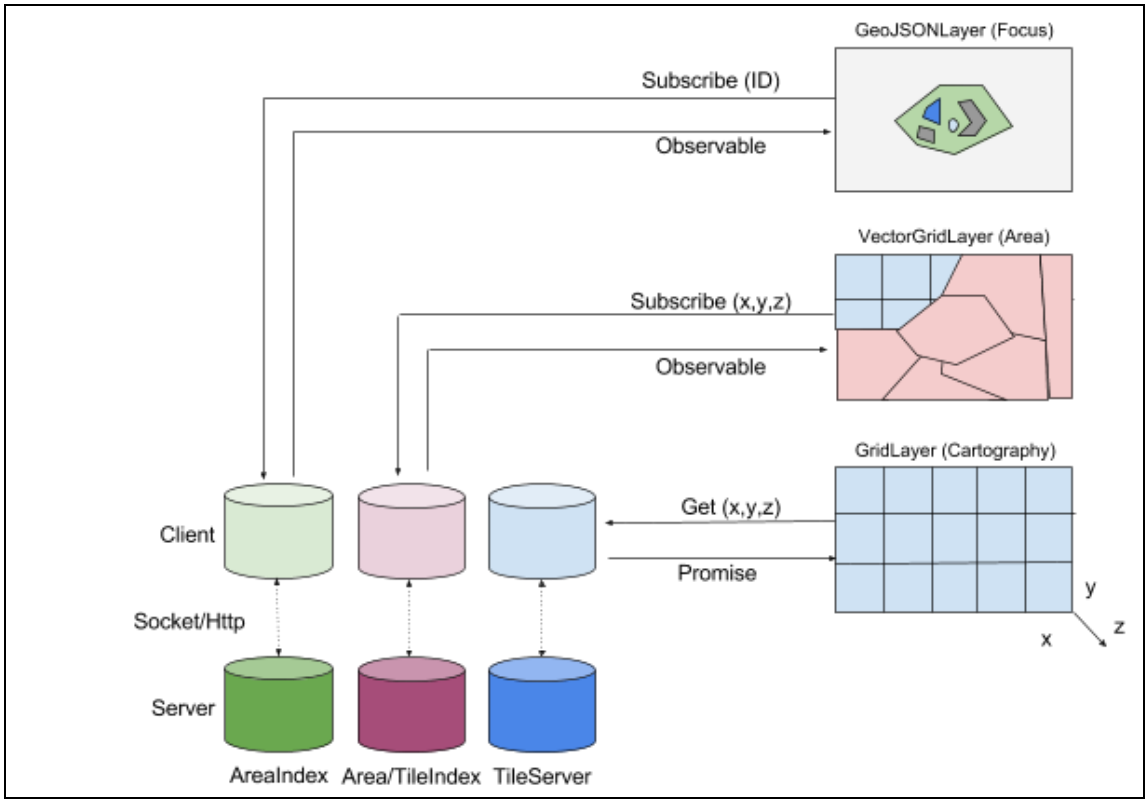
The prototype of LandingPage is implemented in Angular 2 release candidate 5 and Leaflet 1 release candidate. This version is currently being ported to Angular 2 and Leaflet stable versions.

AreaViewer (V1)

The AreaViewer is a web map based module providing a view of WeGovNow aggregated data based on OnToMap. It works in combination with the InputMap component, exploiting the explicit relations between application data of WeGovNow components and OnToMap entities.

The purposes of the AreaViewer are enhancing the overall look and feel of WeGovNow platform and providing a coherent visualisation of the status of WeGovNow instances across components. AreaViewer presents information from the overall WeGovNow platform extracted from users' activities. The information provides will be a summary represented as area-based clusters (aggregations based on geographical entities), see Exhibit 8.

Exhibit 8. AreaViewer is composed by a static cartographic layer, a dynamic geographical layer presenting aggregated data in area-based clusters and a focus layer presenting the details of area-based clusters.

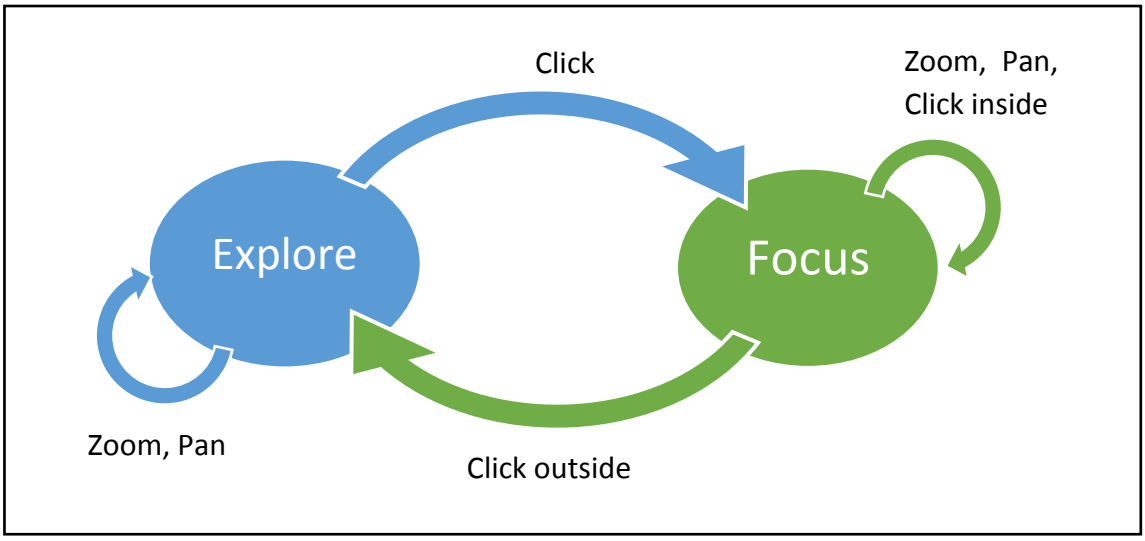


The AreaViewer is a component of the LandingPage that can be included in any WeGovNow component trough iFrame (embed) and be controlled via url. Currently, the following parameters are available:

1) **C**, hash encoding on the initialization latitude, longitude (map centre) and zoom level (latitutde:longitude:zoom) of the map.
2) **Date**, ISO format of the day, month, year of the retrieved data (default current day)
3) **Contrast**, selection of the low/high contrast style
4) **Lang,** interface language

AreaViewer has two main status: “explore” and “focus” (Exhibit 9). In explore status it is possible to pan and zoom the map, changing the view port, clicking will cause to switch to focus status. The focus will highlight the clicked area, hiding all point of intereset (POIs) not related to the specific area. In focus state, a click inside the focus area will cause to focus on deeply on a new area, a click outside the focus area will cause to go back to explore mode, restoring the previous viewport.

Exhibit 9. AreaViewer status automata: explore mode will change to focus on a specific area by click, it goes back to explore mode by click outside.



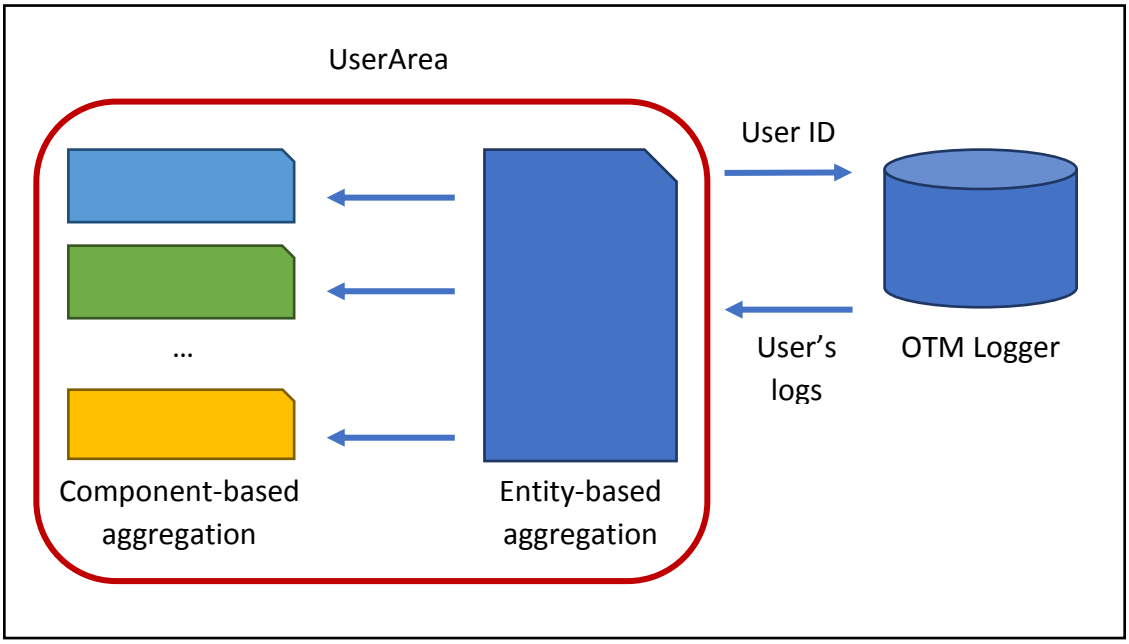
AreaViewer is based on Leafletjs technologies, a famous open source JavaScript library for web maps. It is developed in JavaScript ECMA6 and released under MIT license. AreaViewer source code and documentation can be found in the project repository (https://gitlab.di.unito.it/firstlife/areaviewer).

UserArea (V1)

UserArea's purpose is to provide a direct access to latest entities which a logged in user interacted with. It retrieves logs of user's activities from OntoMap logger to build a list of entities structured by their origin component (Exhibit 10).

The entity list will provide a direct reference (URI) to the origin component, highlighting if new updates are available.

Exhibit 10. OntoMap (OTM) logs of user's activities are first clustered by entity and then organised by the source of activities.



UserArea is developed in Angular 4.x, as module of LandingPage.

Combining AreaViewer and UserArea

The output of the AreaViewer focus is the current area of interest of the user. A message exchange from AreaViewer to UserArea, UserArea change state from user wall to an area-based view: retrieving logs related to the focus area, UserArea presents chronological list of latest activities in the area, providing fast access to the resources via deep link, to the source components.

4.2. Authentication Server

New users can register either by using an email address, existing social media credentials or local identity providers. The registration is valid for all components of a WeGovNow installation. In Exhibit 11 and Exhibit 12, login user interfaces.

The Authentication Server is provided by LiquidFeedback. The implementation shall also support user login using social media ID services. This allows participants to use already known credentials to access a WeGovNow platform - no need for another password. It is planned that Google ID and Facebook Login will be supported out of the box. Other ID providers could be added using an external login interface. Nevertheless, using social media ID services does not replace an appropriate accreditation process of the participants to ensure that no person can use multiple accounts to increase their voting weight (adherence to "one man – one vote").

For certain rights, e.g. voting privileges, a validation of an existing account may be necessary. A user can request his or her account to be validated. To ensure a proper accreditation process, the validation process for a given installation must be defined by the municipality or organization in charge of the WeGovNow installation.

Exhibit 11. UWUM login form.

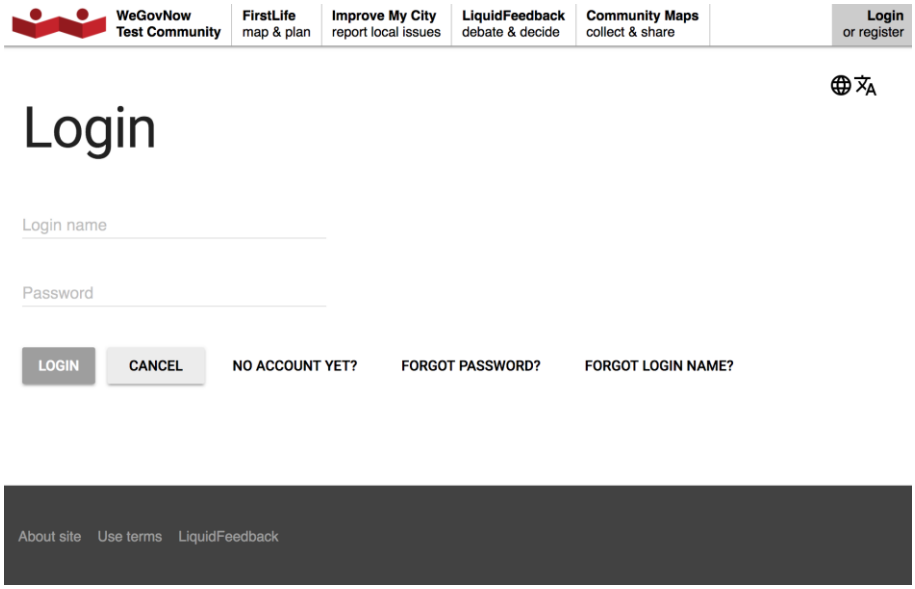
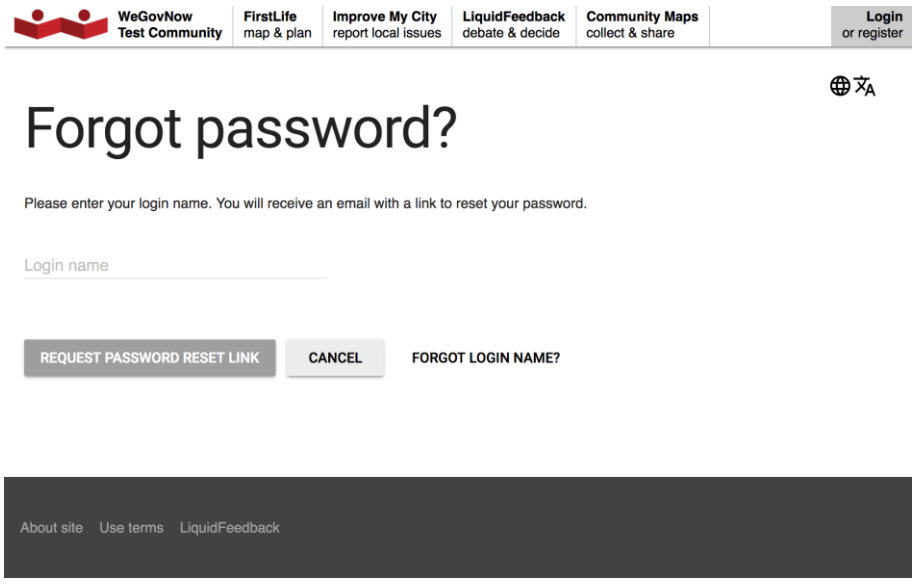


Exhibit 12. UWUM recovery password form.



The Authentication Server provides single-sign-on functionality for all other applications. The already implemented full-fledged OAuth 2.0 server implementation allows to share participant authorization information with other components of a participation solution, e.g. mapping or issue reporting components. Furthermore, the unified user management allows sharing of profile data and user settings across

different components of a participation solution. This allows a seamless integration of all components into a homogeneous platform. Participants can access all connected components without the need for multiple account registrations or multiple logins on different platforms. In turn, other applications can rely on LiquidFeedback as an identity provider, including a check whether an internet user has voting privileges in a given setup.

A full description of the implementation can be found in LiquidFeedsback's Work report on Unified WeGovNow User Management (UWUM) development.

4.3. GeoKey and Community Maps

GeoKey provides a database-driven backend storage, together with a custom API that allows two main tasks namely interaction with data (data creation, editing, deleting) and the creation of projects which group data together. The latter is accessed via a web-based project management interface. In addition to this functionality, an API is also provided for user management, which is again enabled via a web-based frontend. The GeoKey architecture allows the data store to be accessed by any frontend application from the WeGovNow suite (web or mobile based) which can be customised making use of the provided APIs.

A flexible and stylish participatory mapping frontend, Community Maps can visualise data, compare information, and encourage conversation about the places which matter. Designed using the latest web development technologies, Community Maps offers a fast, reliable and intuitive interface. The display is clear, professional, and engaging for all screen types.

The Community Maps tools have been developed to make use of the GeoKey public REST API and separation between GeoKey and Community Maps allows to have one user interface for project management (more technical) and another for data collection/visualisation (intuitive and easy to use). This method hides complexity of the technology behind the minimalistic and modern approach for end-user interaction. All information within Community Maps is stored on GeoKey, where the API enables storage and retrieval of e data via secure SSL connection.

GeoKey is a web based platform for participatory mapping. GeoKey is the connecting point between data collection on the one hand and data utilisation through the analysis and visualization on the other hand.

GeoKey

GeoKey is a web based platform for participatory mapping. GeoKey is the connecting point between data collection on the one hand and data utilisation through the analysis and visualization on the other hand.

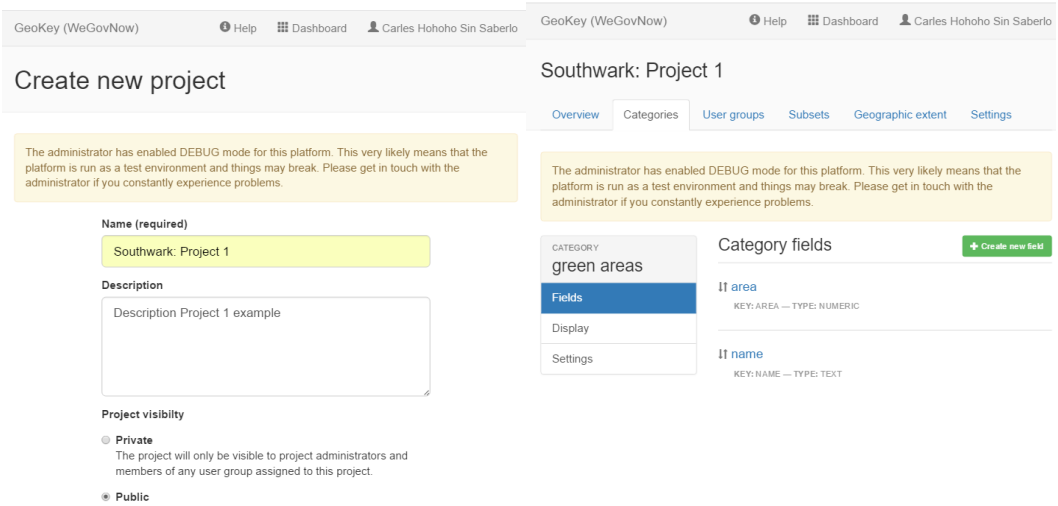
GeoKey allows project administrators (in WGN this relates primarily to municipalities) to generate their own projects and define the data on different categories. The type of data will differ depending the category and the type of data (fields) should be predefined by the admins. The existing type of fields in GeoKey are:

- Text
- Numeric
- Date / Date & time / Time
- Selectbox
- Multiple select box

Thus, the following stages are required:

- Administrators create a project in GeoKey (see Exhibit 13). Once created, administrators can edit the project settings (see Exhibit 14)
- Within the project, administrators create a new collection that will appear on the map. They give this collection a name and (optionally) a description. You can think of this as the name being the equivalent of the table name in a database and the description a metadata description of what the table contains (see Exhibit 15)
- The administrators then give a structure ("columns") to the table – this can be ANYTHING they like – driven by what the users need. So any combination of column types (date, number, lookup, multiple lookup, text and so forth) can be used.

Exhibit 13. Project creation and custom fields.



Each project thus has its own set of categories (corresponding to layers on a GIS map). The important thing is that the categories and the information they contain are defined by owners of the project (i.e. the users of the system) and created by the administrators to contain the information that the specific group of users needs for their project. The variety of data that can be collected is illustrated by going to our live site: https://communitymaps.org.uk/welcome

This approach gives WGN users - whether municipalities, other organisations or local groups - the flexibility to create their own projects and decide what needs to be mapped as a group, allowing them to focus on local interests and priorities.

Exhibit 14. Project geographic extent and settings.

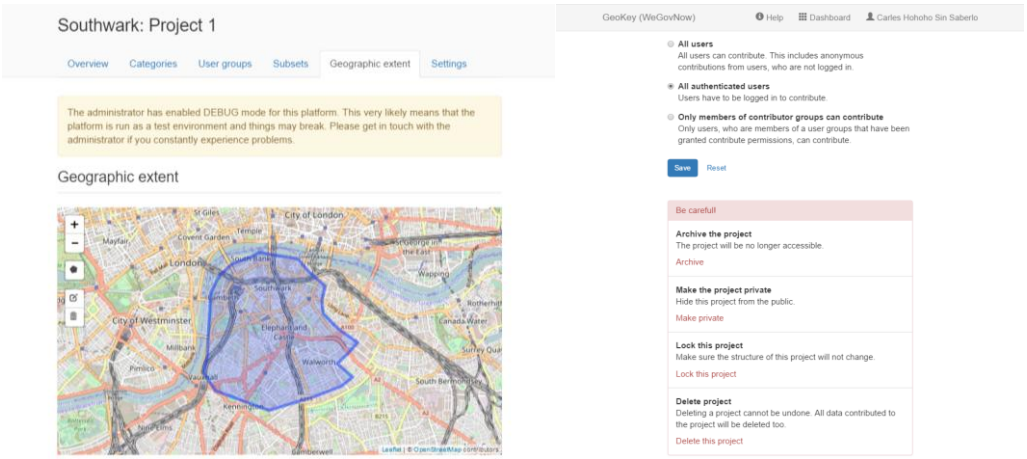
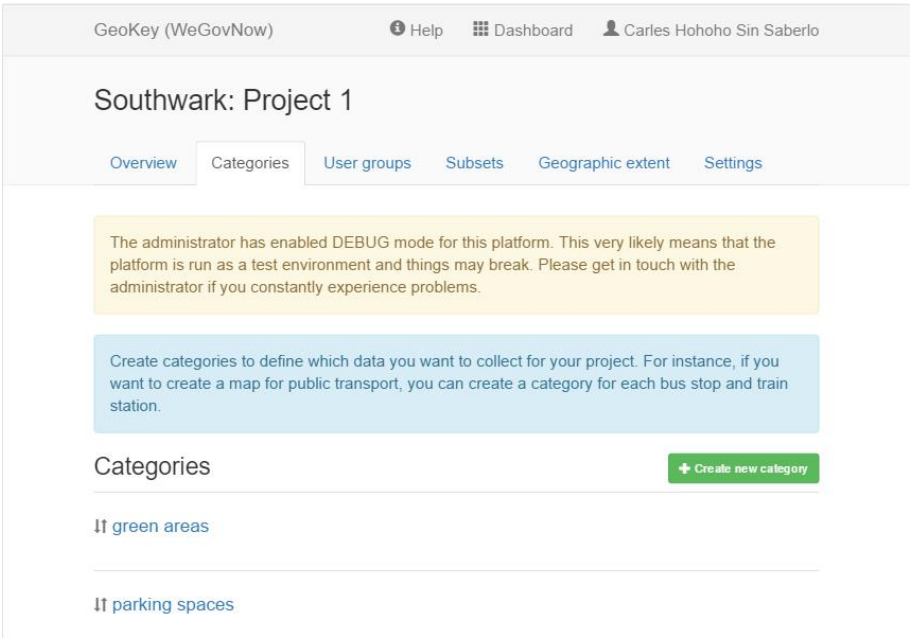


Exhibit 15. GeoKey list of custom categories.



Community Maps

Once the project is live, users add data into the structure provided, using the Community Maps web mapping interface (see Exhibit 16 and Exhibit 17). In Community Maps, users can add new data points (see Exhibit 18 and Exhibit 19), edit existing data or delete data. They can also upload media and add comments to the data points.

There is the option of an administrator being able to moderate the data on the Community Maps site (via GeoKey) if required by the project. Further instructions on how to contribute to a project can be found here: http://help.communitymaps.org.uk/en/add-new-contribution.html

Exhibit 16. List of projects on CM.

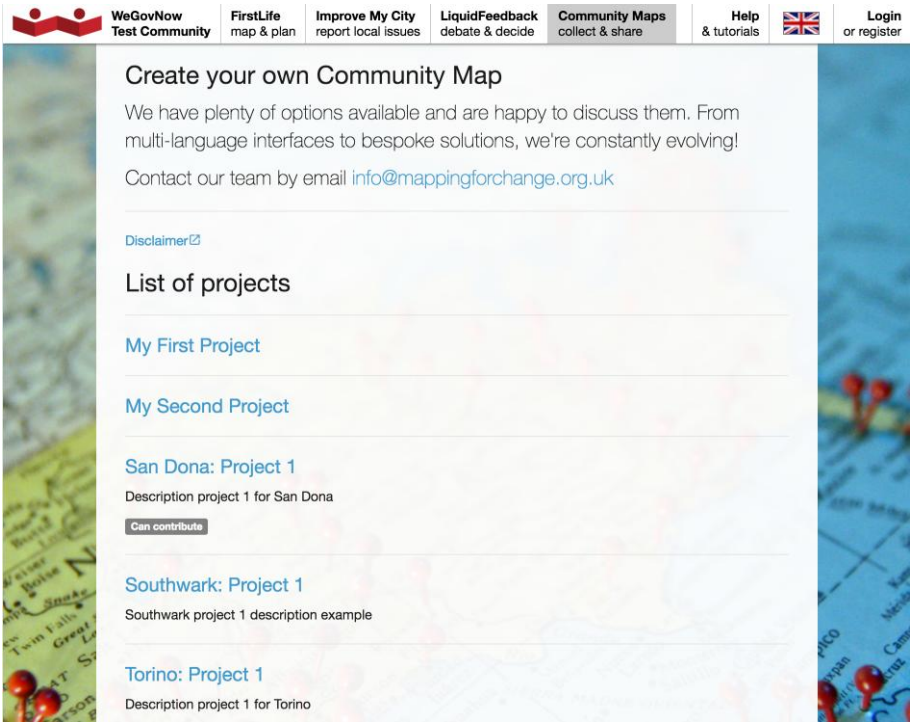


Exhibit 17. Display existing contributions for one project on CM.

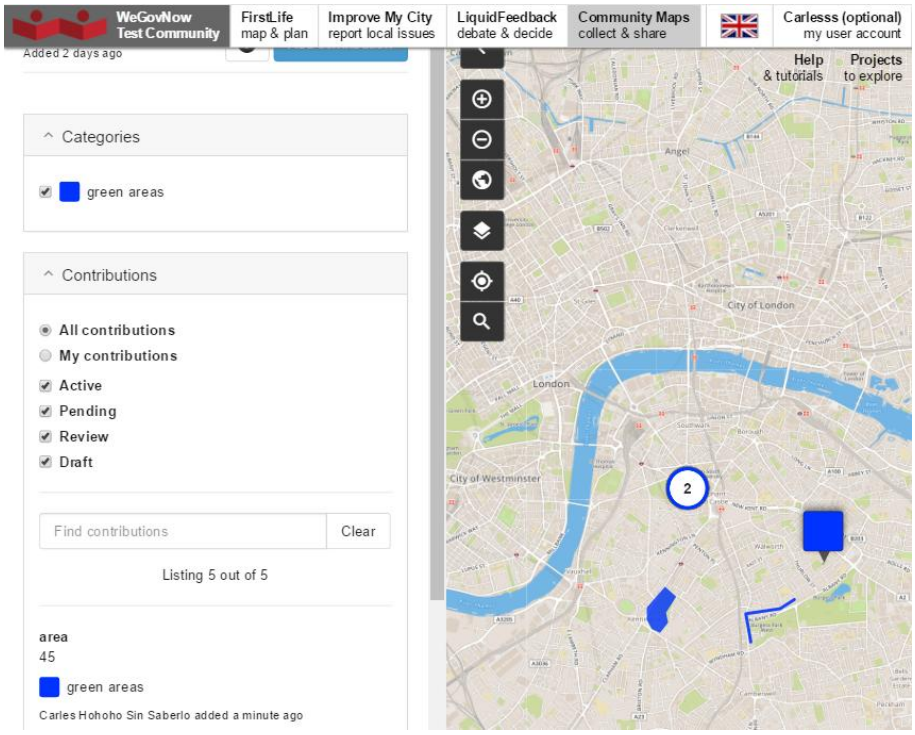


Exhibit 18. Add new contribution on CM. Geometry type definition, and categories.

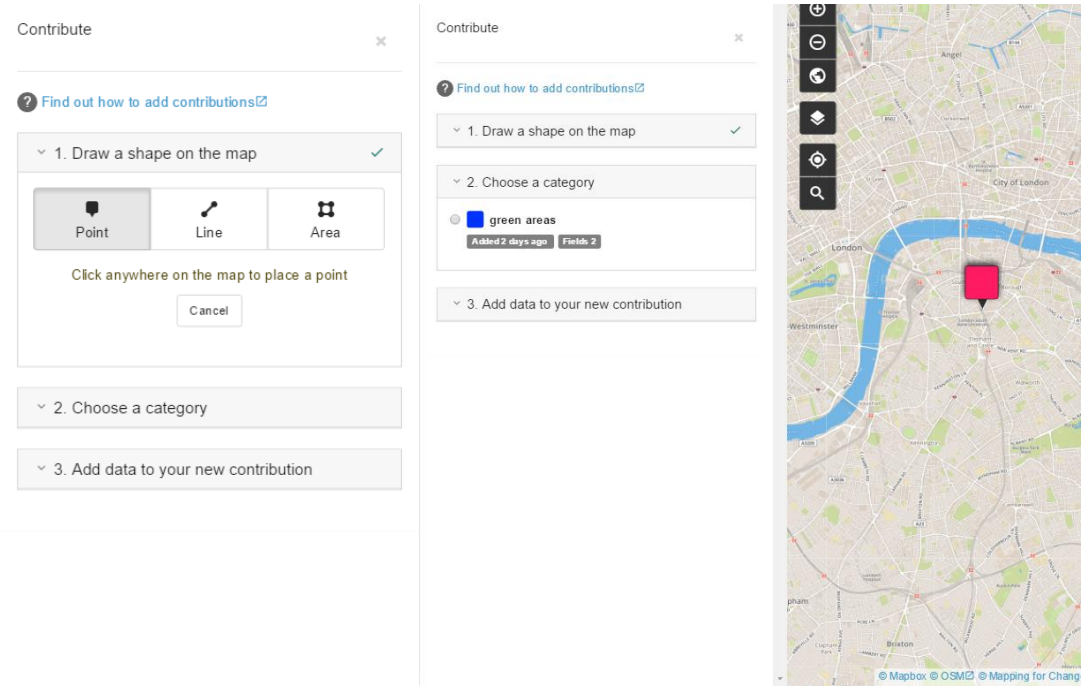
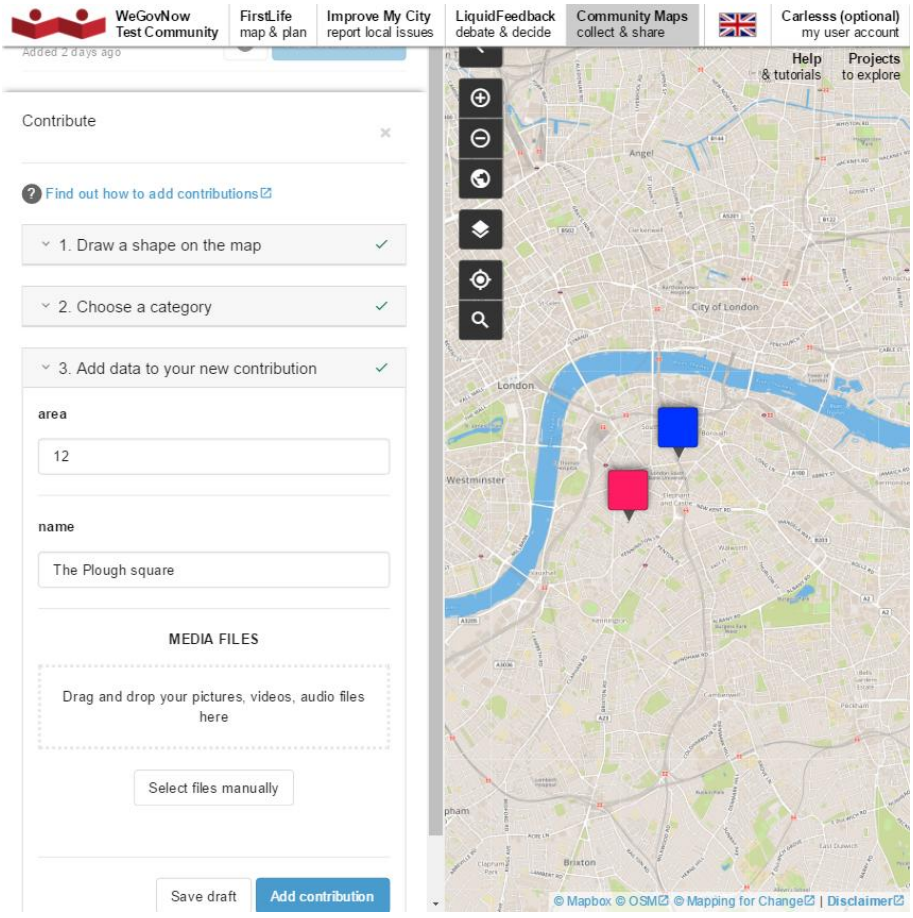


Exhibit 19. Add new contribution on CM. Fill the fields for the chosen category.



4.4. FirstLife

FirstLife is a map-based solution (see Exhibit 20) enabling users to interact with five types of entities: places, events, news, stories, and groups.

Each entity is an aggregation of geo-referenced contents related to a specific geographical unit or area (see Exhibit 21), at multiple scale: building, city block, neighbourhood, district, city, etc.

Exhibit 20. FirstLife map-based view: pie charts representing clusters of markers.

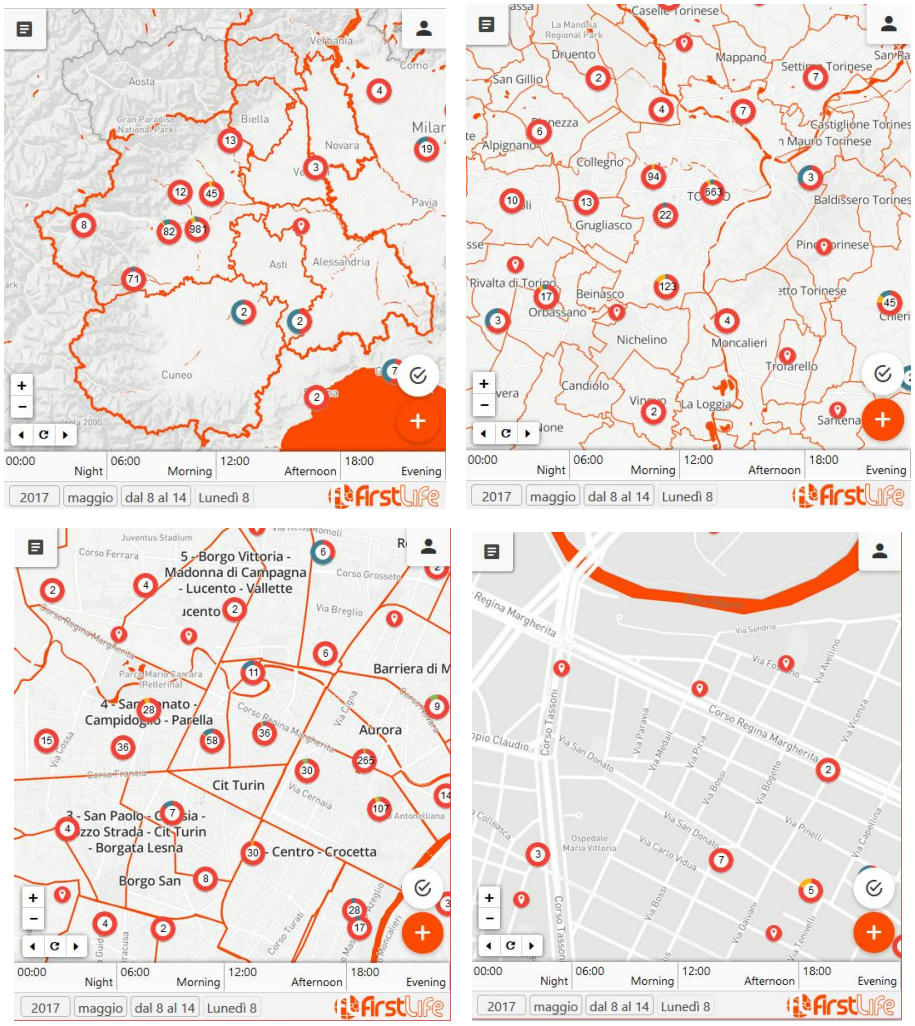
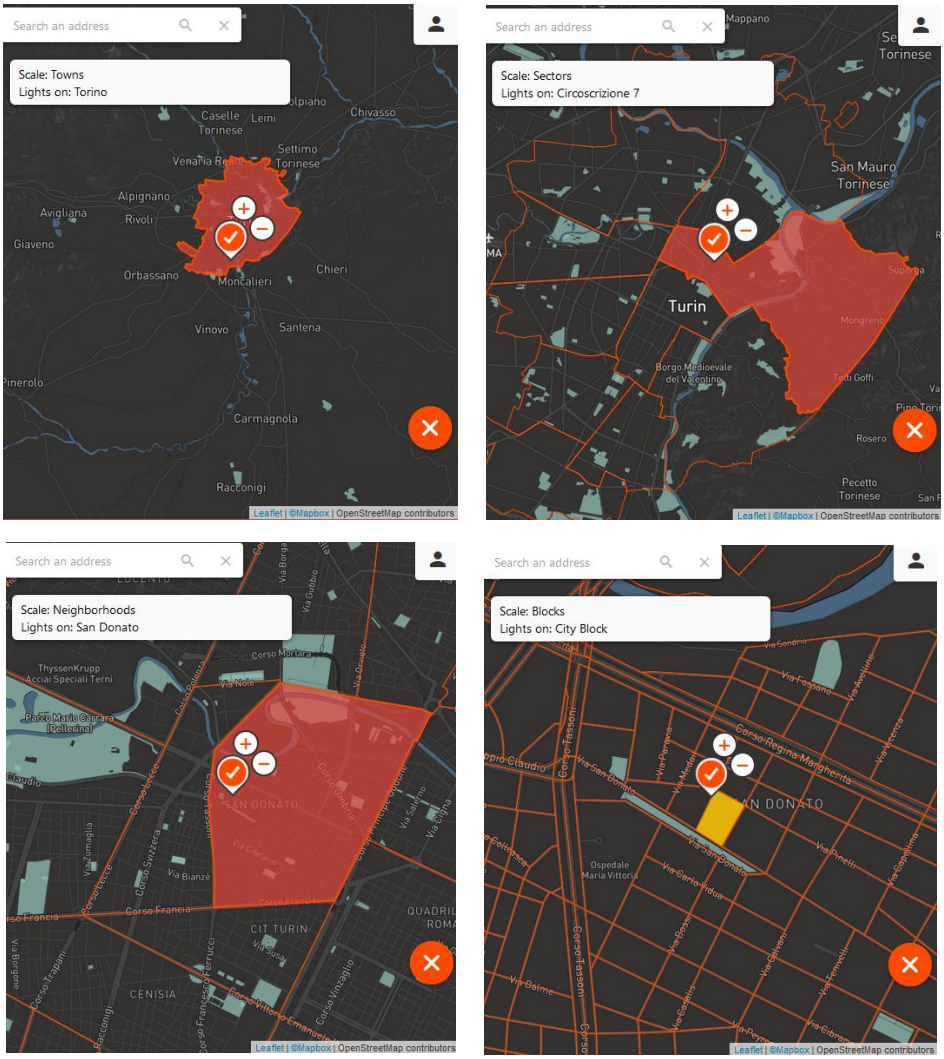


Exhibit 21. FirstLife uses InputMap to collect location input from users.

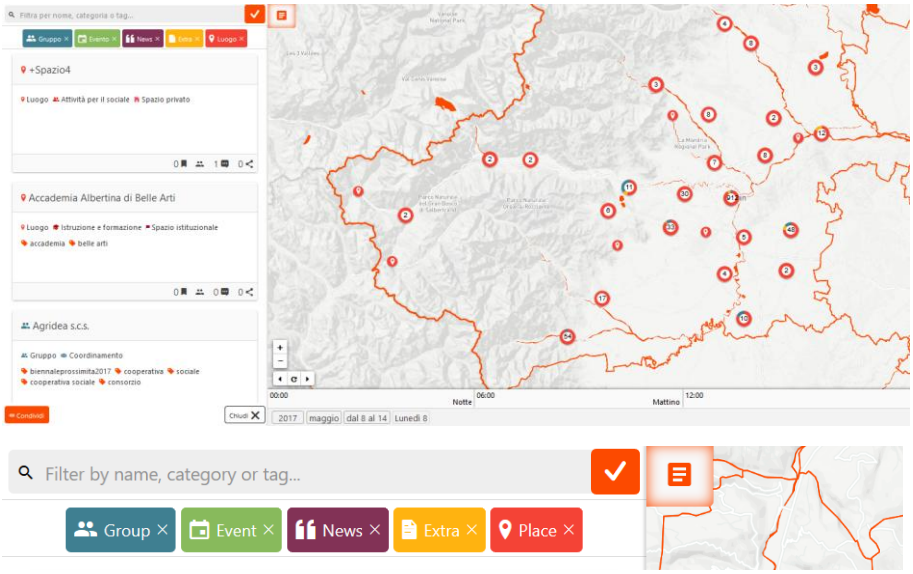


Users can:

- Explore existing contents by opening the entity cards clicking on the markers or selecting the element of interest in the wall
- Contribute to existing contents, by adding a new post or sub-entities to an existing card
- Add new entities on the map.

The integration of FirstLife in the WeGovNow prototype platform is implemented in terms of Angular 1.X release candidate 5 and Leaflet 1 release candidate. This version is currently being ported to Angular 2 and Leaflet stable versions.

Exhibit 22. FirstLife provides a map-based view and wall to explore crowdsourced content. FirstLife implement a multi-dimension filtering system: time, tag, entity type and categories.



4.5. ImproveMyCity

ImproveMyCity can stand both, with or without a map. However, using a map-based UI to display the issues gives a more intuitive user experience to the end users and a better overview when browsing issues. Users can easily select at any time - if they prefer - a list or card based representation of the issues by clicking the appropriate button. The information on both views is the same. But card-display focuses mostly on photos. In any case, depending of personal preference different mores are supported. Different views are displayed by Exhibit 23: on the left is the list view and on the right is the card view.

Each issue contains further details and more importantly, the citizens are able to see the timeline of each issue. When it is submitted (and by whom according to the settings), who (either responsible employee or Department name), when and what action has been taken as shown in Exhibit 24. Submitting a new issue by registered users is a very quick and easy procedure, as displayed in the form in Exhibit 25.

It should be noted that users are able to edit their own issues ONLY if the status of their issue has not yet changed by the administrators. This allows a time window to correct typos, add new photos, etc. The business logic (e.g. when an issue is editable, when it becomes public, whether commenting is allowed or when comments are kept privately between issuers and Municipality, who gets notified and when and many other actions and rules) is set graphically in the backend / administration side.

Exhibit 23. Browsing on existing issues. Different views, same content and information.

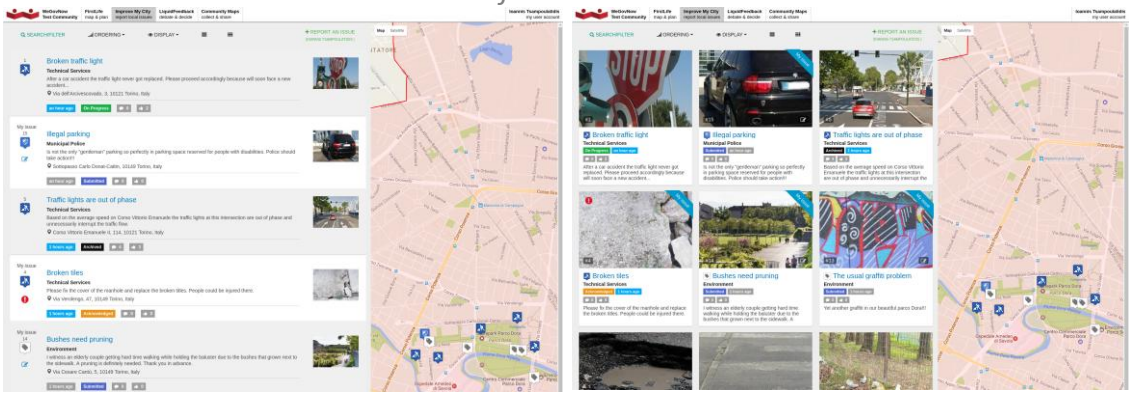


Exhibit 24. Issue details (showing also commenting and voting functionality) and detailed timeline that promotes transparency and direct citizens – Municipality interaction.

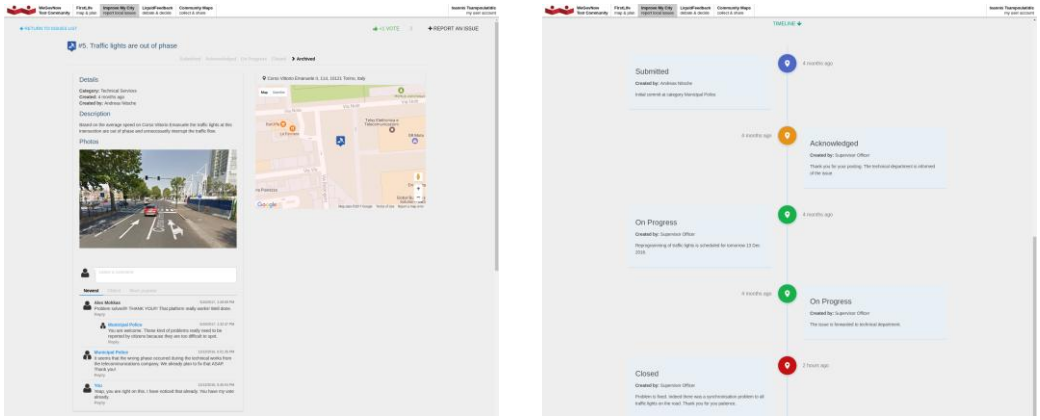
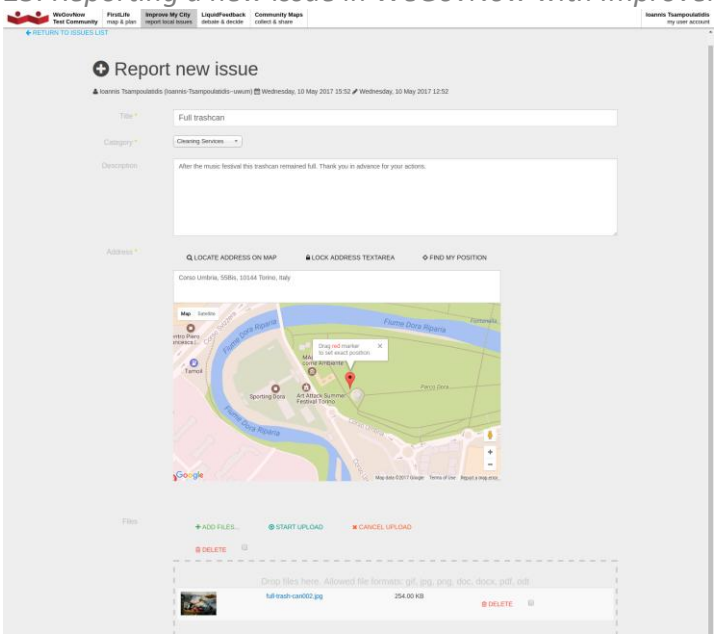


Exhibit 25. Reporting a new issue in WeGovNow with ImproveMyCity.



ImproveMyCity is not a monolithic application but a highly modular application that can expand or reduce the complexity and features on demand. More specifically, it is composed of:

- Core component
- Map module
- Filtering module
- Search plugin
- Notifications plugin
- Categories extra fields plugin
- Reporting expansion
- Workflows expansion

Each and every module/plugin has its own settings and preferences but they operate and communicate with each other flawlessly. The following exhibit (Exhibit 26) shows some of the settings that are set graphically by the administrator.

A municipality's organisational diagram is easily imported no matter how complex it is. For each department, office and generally any hierarchical level, it is feasible to define permissions. Each employee belongs to one or more groups/departments and inherits its permissions. Each department (e.g. Technical Department) can have one or more corresponding categories (roads, parks, etc.) and its assigned officers get notified automatically according to the rules.

Exhibit 26. ImproveMyCity is highly modular and parametric to cover different Municipality needs according to their workflow and complexity.

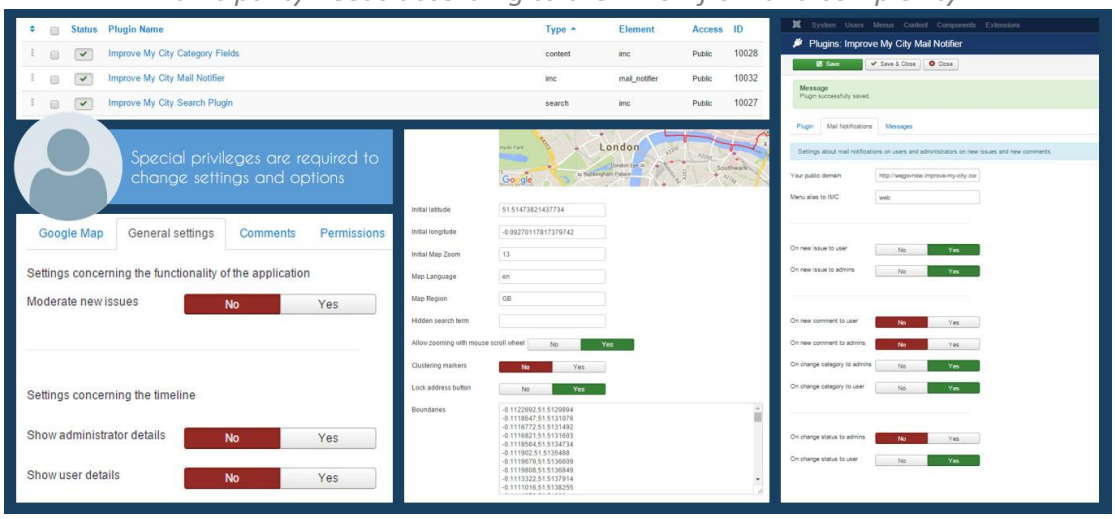
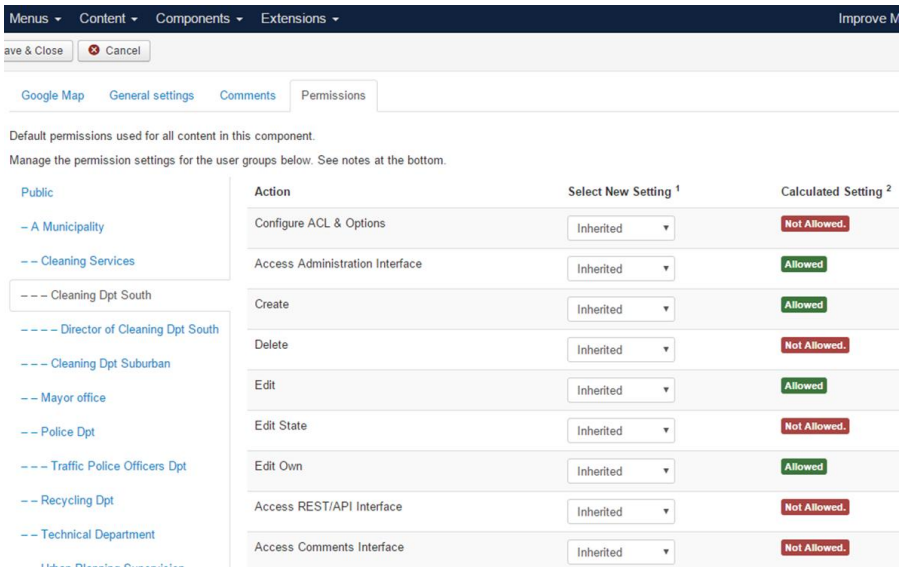
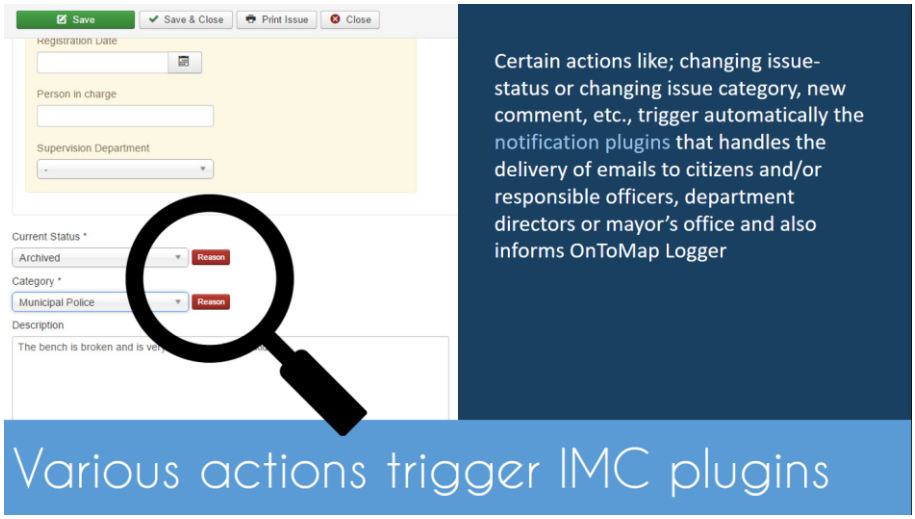


Exhibit 27. ImproveMyCity permissions according to the Municipality Hierarchy. "Who does what, who sees what".



Administrator employees in ImproveMyCity are facing a very simplified User Interface without the hassles and complexities. They see only the issues that concern them and the only actions that really take is to change a dropdown menu and type a response. All others (notifications, logs, updates, etc.) are handled automatically by the application.

Exhibit 28. ImproveMyCity triggers most actions automatically reducing the effort of employees to the bare minimum.

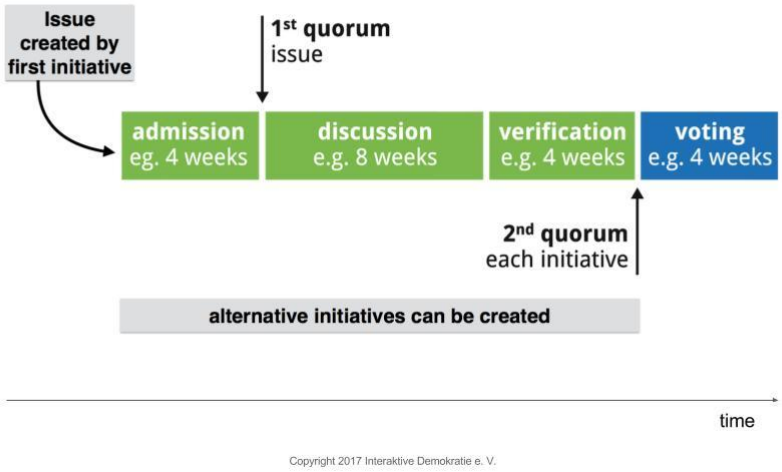


For a detailed explanation of the backend/administrator side of ImproveMyCity, visit https://wegovnow.improve-my-city.com/handbook/administration/general.html

4.6. LiquidFeedback

The overall function of LiquidFeedback within WeGovNow is opinion formation consisting of a deliberation process and a voting phase to determine a collective preference (Exhibit 29).

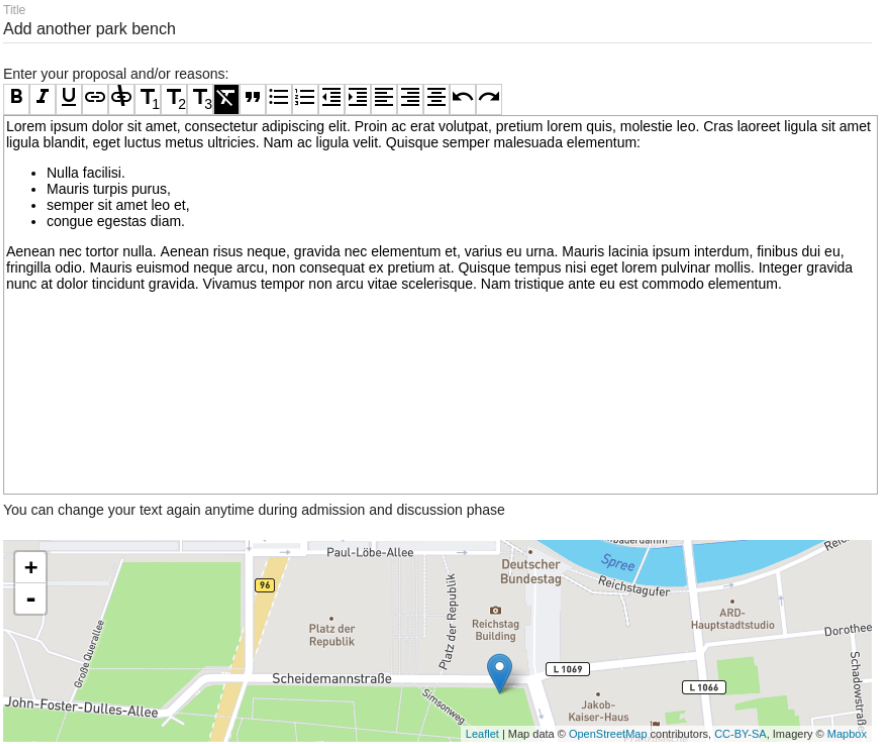
Exhibit 29. The LiquidFeedback proposition development process.



Citizens can start initiatives (proposals) and seek support among their fellow citizens using the input map provided by FirstLife (partner UniTo) and the new WYSIWYG editor (rich text editor) of LiquidFeedback Exhibit 30. Other citizens can suggest improvements or start alternative initiatives.

Predefined rules and timings ensure that plans on decision processes are made public in time. Decisions are made by recorded vote only, and all voting-relevant data in LiquidFeedback is made available to all participants in both human– and machine–readable form. This enables a transparent decision making process and ensures that participants can verify the voting procedure.

Exhibit 30. Creating a New Initiative in LiquidFeedback with WYSIWYG Editor and WeGovNow InputMap.



In the context of WeGovNow, LiquidFeedback already added geospatial functionality and published a core update as a prerequisite for the integration (see “Work report on pgLatLon in Appendix 6.5, an alternative to PostGIS”and “Second work report on extending the LiquidFeedback Core” in the appendix). The adoption of material design is in progress.

For prototype 2, LiquidFeedback focused on finalizing the integration features and check the feasibility. The input map was integrated, a new rich text editor was added, and LiquidFeedback's UWUM server was extended with an application discovery endpoint providing access to the list of both static and dynamically registered applications. The finalization of the frontend will be the central aspect of prototype 3.

5. Description of platform services

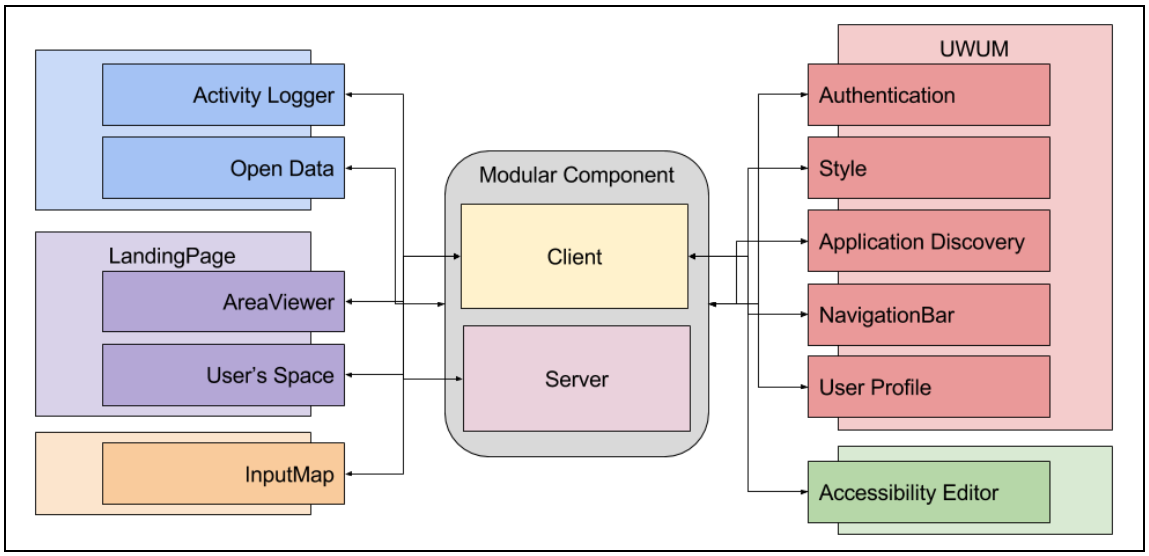
The core features of WeGovNow are meant to support the integration between components. A set of core features is provided by an extension of stand-alone components, while other core features required the definition of new modules. In this regard, as indicated in Exhibit 5 and Exhibit 6, the first prototype of WeGovNow does not include some new core features which are currently being developed.

The WeGovNow core thus provides features essential for realising the seamless integration of individual WeGovNow components:

- **Unified Authentication System:** provides functionalities concerning user registration, authentication and authorization, as well as single sign-on.
- **Application Discovery Service:** is an API service providing the list and details of currently available components in a WeGovNow instance.
- **Style Service:** is an API service providing WeGovNow style sheets dynamically to the components, it is used to retrieve the instance customization such as colours, fonts, etc.
- **NavigationBar:** is an API service providing the description or the HTML source of WeGovNow navigation bar, including the button tabs to the current available components and the reference to the user profile.
- **Centralised User Profile:** provides a user profile data store accessible to all registered WeGovNow components to store, retrieve and share user profile information, it consents to keep user information updated across the platform.
- **Accessibility Settings:** is a wizard to collect and edit cross-platform user's profile setups about accessibility (see section 6.2).
- **Centralised Activity Logger:** provides centralised data logging within the WeGovNow platform and data integration aimed at integrating the knowledge about users and about geographical data shared in the platform.
- **Linked Open Data and Crowdsourced Data endpoint:** is a semantic endpoint to retrieve the linked open data generated from the open data of the municipalities and the user activities within a WeGovNow instance.
- **InputMap:** is an embeddable web map to collect spatial input (point based references) and references to existing entities in OntoMap.
- **AreaViewer:** is an embeddable web map to visualise summary information extracted from OntoMap (see section 4.1).
- **TileServer:** a vector tile server providing geographical entities from official open data and OpenStreetMap, in protobuf format (PBF).

As mentioned earlier, the core features are provided by existing and new components specifically developed for WeGovNow.

Exhibit 31: WeGovNow core features provide the environment to integrate modular components accordingly to the specific needs of each WeGovNow instance.



Core features are used by modular components (see *Exhibit 31*) to:

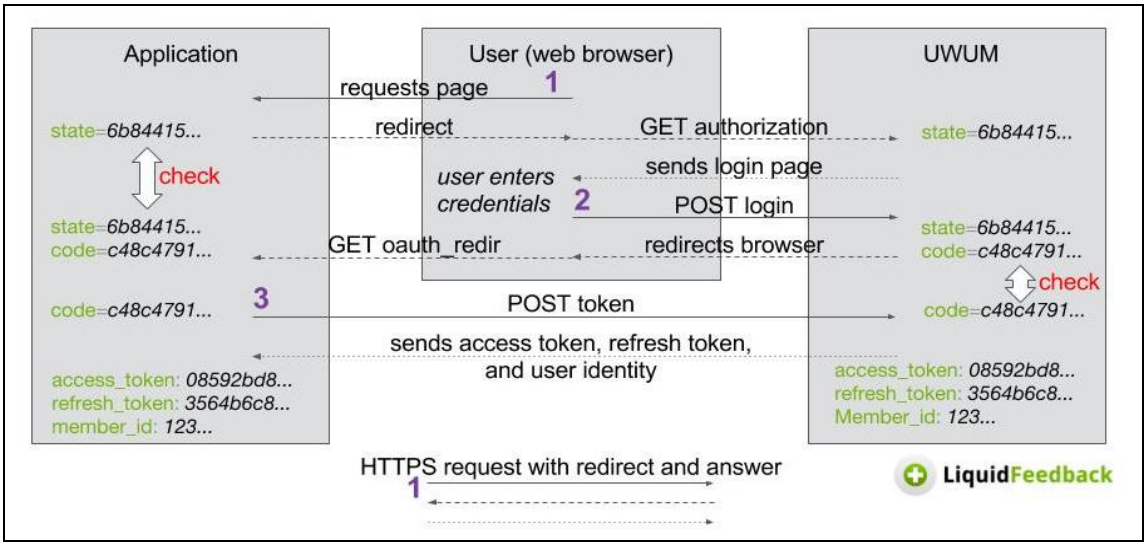
- Establish connections with other WeGovNow components
- Provide cross-component features
- Unify the appearance to the rest of a WeGovNow instance
- Synchronise components

As any other modular component, core features themselves are optional as far as their non-essential functionalities are concerned: if a core feature is not fully enabled in a specific instance of WeGovNow, only its critical functionalities will be available. In the following subsections, the WeGovNow core is described in more detail.

Authentication Server

For reasons of interoperability and security, WeGovNow aims to create an implementation that is fully compliant with the OAuth 2.0 Authorization Framework as described in RFC 6749, but extended in such way that it allows for secure user authentication following the OAuth 2.0 Authorization Code flow (see Exhibit 32). Special security considerations were taken into account, for instance client identity verification (through X.509 certificates) to repel authorisation code substitution attacks. For further security considerations refer to Consolidated System Architecture D3.1, section 2 and section 5.

Exhibit 32: OAuth 2.0 Authorization Code flow.



RFC 6749 defines several roles ("authorisation server", "client", and "resource server"). The UWUM component as implemented by LiquidFeedback takes the role of the "authorisation server". Other WeGovNow components will take the role of "clients" but may also act as "resource server" for other components. This allows other components to interact with each other, while UWUM is responsible for user authentication and authorisation.

To register a new client, it is required to address the UWUM integration checklist (see annex 2, Integration checklist). UWUM provides two methods of client registration:

- registering clients through the municipality (or their technical administration) or an organisation running an installation of WeGovNow,
- registration of any other ("dynamic") client on a per-user basis by each user who wishes to use that client to access WeGovNow (machine accessibility).

Manual client registration by the municipality is only suitable for those clients that are known at the time of deployment of a WeGovNow instance.

Currently, it is possible to use UWUM with an invite code, in the future it shall be possible to create non-verified accounts using an email registration, Google, Facebook or OAuth 2.0 compliant authentication server. Non-verified accounts must not have any voting right or be counted in any other quantification until they are verified.

Application Discovery Service

The Application Discovery Service provides information about the available software modules in a WeGovNow instance, as defined in the configuration of the platform. The Application Discovery Service role is to enable the dynamic configuration of

active WeGovNow software components: to enable cross-components features accordingly to the availability of other services, to provide navigation active components can "discover" other enabled components.

New components can be included in a WGN instance and in WGN environment in general following the Integration Checklist (see Annex 2).

The Application Discovery Service is available via a REST API endpoint for authenticated clients (Exhibit 33). For technical details and future development of the Application Discovery Service see Annex 2 to D3.1 WeGovNow Consolidated Architecture.

An application discovery endpoint has been added for prototype 2.

Exhibit 33. Application discovery.

https://wegovnow-pt2.liquidfeedback.com/api/1/application


```
// 20170805164739
// https://wegovnow-pt2.liquidfeedback.com/api/1/application

{
  "result": [
    {
      "name": "FirstLife",
      "type": "system",
      "url": "https://pt2.wegovnow.firstlife.org/"
    },
    {
      "name": "Trusted Marketplace",
      "type": "system",
      "url": "https://pt2-tmp.infalia.com/"
    },
    {
      "name": "Landing Page",
      "type": "system",
      "url": "https://pt2.wegovnow.eu/"
    },
    {
      "name": "Improve My City",
      "type": "system",
      "url": "https://pt2-imc.infalia.com/"
    }
  ]
}
```


Considering the extent of the existing WeGovNow applications, using application discovery might create more effort for every component owner than possible benefit. However, the existing UWUM endpoint for application discovery returns a

list of applications along with their registered base URLs and provides a method for application discovery in WeGovNow systems with a growing number of applications.

Dynamically registered applications (dynamic clients for OAuth 2.0) are fully supported (registration with X.509 TLS client certificate authentication or DNS TXT records, see section 2.4.2 of the UWUM Work Report from December 12, 2016). These dynamically registered applications are included in the list of registered applications returned by the application discovery endpoint for the currently logged-in user. Additional information on provided protocols and services may either be retrieved through a yet-to-be-defined well-known URL constructed from each application's base URL, included in future versions of the UWUM database, and/or collected by other applications such as OnToMap.

Style Service

An instance of WeGovNow includes style customization, for instance colours of a municipality, fonts, icons, etc. The style service was introduced as part of UWUM (Unified WeGovNow User Management) specifically to provide the style customization of the current instance of WeGovNow.

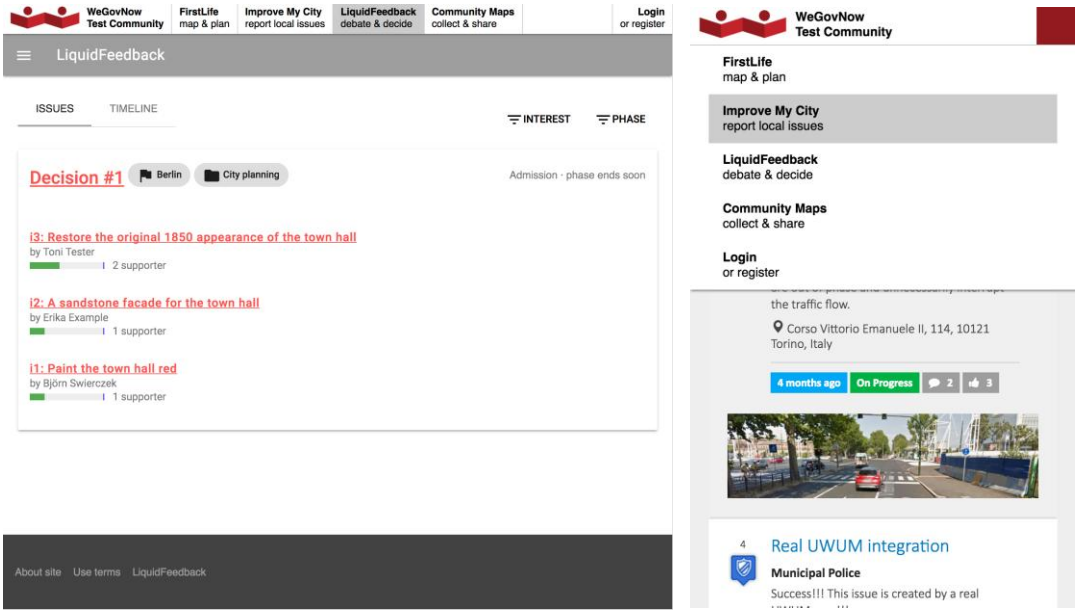
The Style Service already provides a material design colour theme. As of now only LiquidFeedback honours the colour theme. It planned that all other components also honour the colour theme. Future versions may also include fond and other style information.

For technical details and further development of the Style Service see Annex 2 to D3.1 WeGovNow Consolidated System Architecture.

NavigationBar

To integrate all WeGovNow applications in such way that they look and feel like a single application, all WeGovNow applications share a common WeGovNow navigation bar (see Exhibit 34).

Exhibit 34. NavigationBar is dynamically retrieved from UWUM with the current setup of the platform instance. There is also a responsive version depicted on the right

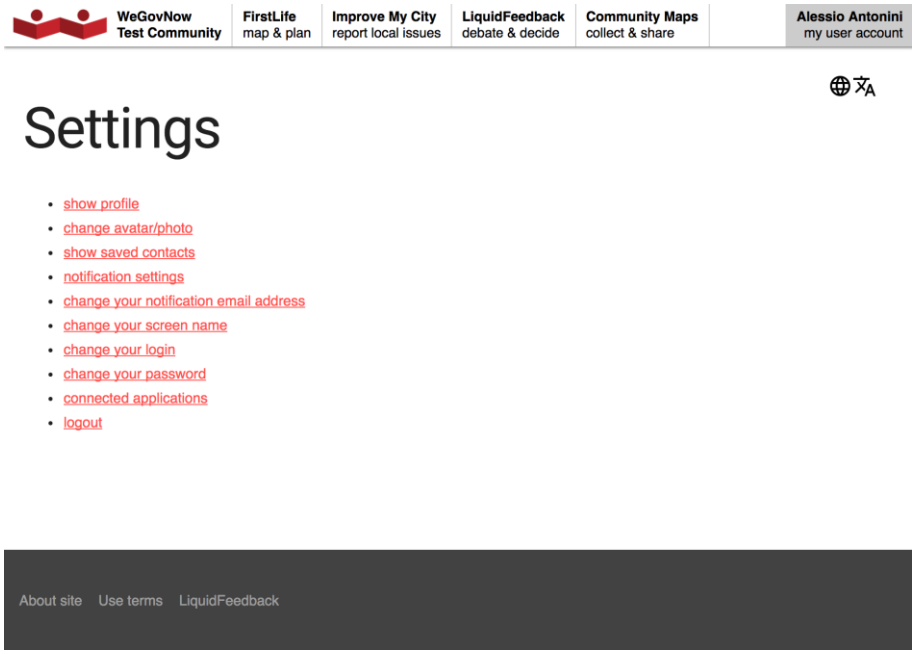


The navigation endpoint of the UWUM server returns this navigation bar to be included by each WeGovNow application. This way, modifications to the navigation bar can be made at a central place without the need to change every single application. Either a login button or the user name with a link to a user page (where logout is possible) is included in the navigation bar, depending on whether an access token is provided when calling the endpoint.

Centralised User Profile

The WeGovNow platform will provide a centralised user profile repository to its components where user profile information of common use among the platform are collected. The centralised user profile will be an extension of user profile settings of UWUM component. Currently, the user settings in UWUM enable users to manage their personal data, authentication details and notification settings. The user settings are accessible only through LiquidFeedback user settings (see Exhibit 35), TrustedMarketplace component will provide an extended interface to the user settings (see section 6.1).

Exhibit 35. UWUM user settings.



The centralised user profile has the following main functions:

- Single synchronisation point for common user information (such as firstname, lastname, displayname, email, etc.) avoiding the multiple request of user profile data;
- Propagating the updates of users' profile across WeGovNow platform (Exhibit 31) avoiding inconsistent information;
- Increasing the integration of WeGovNow components providing a mechanism to share user's information platform wide.

The user profile data fields and their type will be configurable per installation. These data fields can be of standard types such as text, number, image, location or complex JSON data fields. Standard type fields will be automatically editable by UWUM's built in editor but can also be updated using the API. Complex JSON data fields need to be updated using the API by the corresponding component(s) using such fields.

The management of the user profile will be extended in TrustedMarketplace including the reference to the information about the user stored in UWUM, the reference to user settings in each application, the accessibility settings (see next section), and global setups about the management of user information in WeGovNow. Further details in section 6.1, section 6.3 and section 6.2.

Linked Open Data and Crowdsourced endpoint

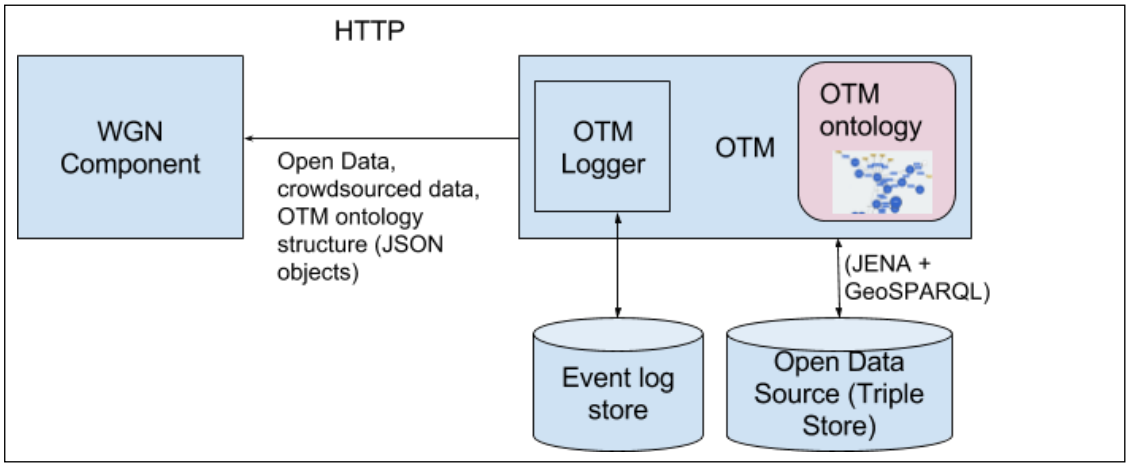
The OnToMap endpoint supports data integration within the WeGovNow platform. The idea is that of having a single point of access to the geographical information available in an instance of the platform, in a unified format and representation language that enables the applications to retrieve and present shared data regardless of their origin. Specifically:

1. OnToMap supports the integration of heterogeneous Open Data sources, which are stored into the platform in a Linked Data format supporting the semantic exploration of information (see below). Up to now, we have acquired a portion of the Open Data from Torino, San Dona' Di Piave and London Southwark, concerning services available in the towns, transportation systems, facilities, and the like. In the next future, further Open Data from the three cities will be added; e.g., aggregated statistical data concerning the economic tissue in San Donà di Piave. Moreover, in a different project (MIMOSA - MultIModal Ontolgy-driven query system for the heterogeneous data of a SmArtcity, "Progetto di Ateneo Torino_call2014_L2_157", 2015-17) we are working at the integration in OnToMap of data from OpenStreetMap[1]. We will make this feature available to the WeGovNow platform in order to complement the data provided by the cities (e.g., in the cases in which the Open Data is partial).
2. OnToMap supports the integration of data shared within the platform by the WeGovNow applications, which can publish the users' activities concerning the creation, revision, etc., of data items through the OnToMap Logger. As this type of information is based on the observation of events concerning user activities, it is stored in a log as a set of events mapped to the common terminology provided by OnToMap. This aspect is described in the section about the OnToMap Logger.

The integration of heterogeneous data and their representation in a common format is based on a semantic knowledge representation layer that reconciles data representational and conceptual heterogeneity.

[1] https://www.openstreetmap.org

Exhibit 36. OnToMap architecture.



The OnToMap Ontology (see Exhibit 36) is a knowledge representation layer that makes it possible to:

- Integrate heterogeneous Open Data and manage it as Linked Data. As reported in http://linkeddata.org/, "Linked Data" is about using the Web to connect related data that wasn't previously linked, or using the Web to lower the barriers to linking data currently linked using other methods."
- Provide a dictionary, shared in the platform, for mapping the concepts used in the domain conceptualizations adopted by the WeGovNow applications to a unified terminology for cross-application data sharing.
- Describe semantic relations among information items to express spatial relations, different levels of abstraction in the description of entities and thematic relations. This approach enables a semantic and geospatial exploration of the information space.

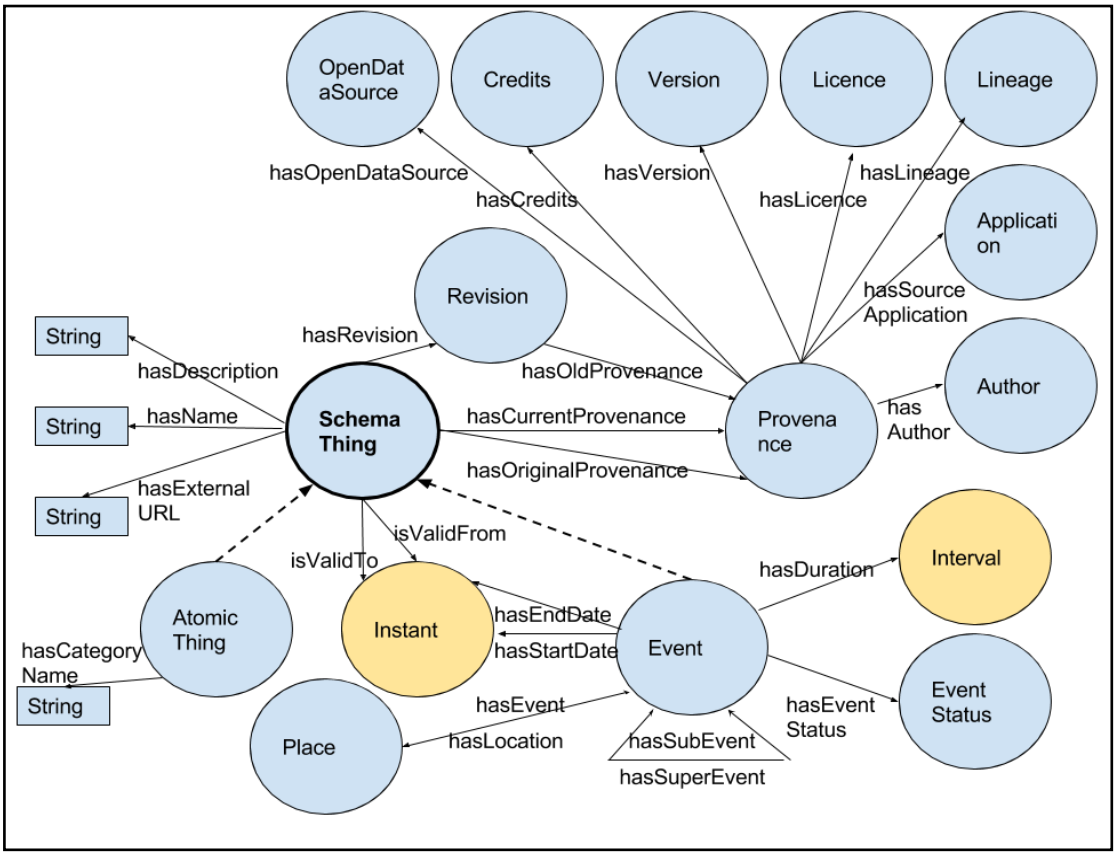
The OnToMap Ontology is defined using the OWL Web Ontology Language https://www.w3.org/OWL/.

The latest version of the OnToMap Ontology can be downloaded in OWL format at the following URL: http://ontomap.eu/ontology/. The ontology includes about 100 concepts.

The following Exhibit 37 shows a portion of the current version of the OnToMap ontology, focusing on concept "SchemaThing", the root of the taxonomy concerning geographical information.

"SchemaThing" specifies the main data structure for geographical information items. It is related to concept "Revision", in order to comply with the possibility of having multiple versions of data, modified at different times by different users, and to concept "Provenance" in order to support the retrieval of the provenance of the current version of data. In turn, each revision of a geographical data item is related to its own provenance, so that the revision history of data can be reconstructed.

Exhibit 37. Portion of the OnToMap Ontology.



The OnToMap Ontology supports a more or less tightly coupled data integration, in order to allow both the mapping of stable domain conceptualizations, based on a fixed set of data categories (e.g., FirstLife's one), as well as the integration of dynamic conceptualizations, similar to tag-based systems. "SchemaThing" has the following sub concepts:

- "AtomicThing", which represents geographical information based on dynamic data categories, such as those used in GeoKey/CommunityMaps. It has a "hasCategoryName" property, which can be used to associate information items to categories, modelled as tags. This approach supports a free type of data tagging, which does not require consistency checks when adding new categories to a domain conceptualization.
- Concepts for the representation of geographic information based on stable domain conceptualizations, which can be mapped to OnToMap Ontology:
    - "Event", which represents events, associated to their temporal extension (see the relations with concept "Instant", which models temporal points), and to the "Places" in which they are held.
    - "Place", which represents all the geographical items having a geometry representing their geographical extension.

Regarding this work, POLITO has collected, analysed and prepared for acquisition in OnToMap the Open Data provided by the cities of Torino, San Dona' di Piave, and Southwark. UNITO has carried out the design and development of the Software (OWL ontology, OnToMap component with Logger).

Representation and storage of Open Data

The Open Data integrated in OnToMap is mapped to the ontology concepts and it is represented as Linked Data, as RDF (Resource Description Framework[2]) triples. This makes it possible to exploit standard tools for browsing the ontology and for retrieving geographical information items.

As shown in Exhibit 36, OnToMap stores Linked Data in the Open Data Store: this is a triple store, i.e., a specialized tool for managing and querying RDF datasets. The current implementation of OnToMap exploits the Parliament triple store[3], which supports GeoSPARQL queries[4].

OnToMap exploits the Apache Jena ontology API[5] to perform GeoSPARQL queries on ontology concepts. The result of a query is a set of RDF triples representing geographical information items. This set of triples is translated to an external GeoJSON format to make data directly usable by WeGovNow applications for map-based presentation.

The mappings between the data categories defined in the Open Data Sources and the ontology concepts are aimed at translating external data items to sets of RDF triples to be loaded in OnToMap Open Data Store. As Open Data sources typically use standard data representations, open source mapping tools can be found for doing this work without defining ad-hoc software. For instance, for converting shape files to RDF data we use GeoTriples (https://github.com/LinkedEOData/GeoTriples)

Data retrieval API

The interaction between OnToMap and the other WeGovNow applications, regarding data retrieval and semantic data representation, is managed through the OnToMap Logger API specified at https://ontomap.eu/, which returns results as a JSON object.

The API can be used to navigate the common data structure defined by the OnToMap Ontology and to retrieve the Open Data provided by the OnToMap Open Data Store, as well as the data crowdsourced by WeGovNow applications, in the unified terminology and format provided by the OnToMap Ontology. Specifically:

2 https://www.w3.org/RDF/
3 http://parliament.semwebcentral.org/
4 http://www.opengeospatial.org/standards/geosparql
5 https://jena.apache.org/documentation/ontology/

- The API supports the exploration of the concepts and relations stored in the OnToMap Ontology. It makes it possible to retrieve the root concept of the ontology, the list of concepts, the concepts related to a specific concept C, and so forth.
- The API can also be used to retrieve information about the instances of a concept C, which can be expressed either in the terminology of the WeGovNow application (in which case OnToMap translates C to the corresponding concept of the OnToMap Ontology), or directly in the OnToMap Ontology format. Depending on the filters used in the query, the retrieved instances can belong to the Open Data and/or to the data crowdsourced by the WeGovNow applications. The API supports filtering instances by geographical area and/or temporal validity, and/or data source (e.g., Open Data belonging to a certain source, crowdsourced data). The geographical information items returned by the API are, by default, the most recent versions of the data available to OnToMap, and they are represented as GeoJSON "Features"[6].

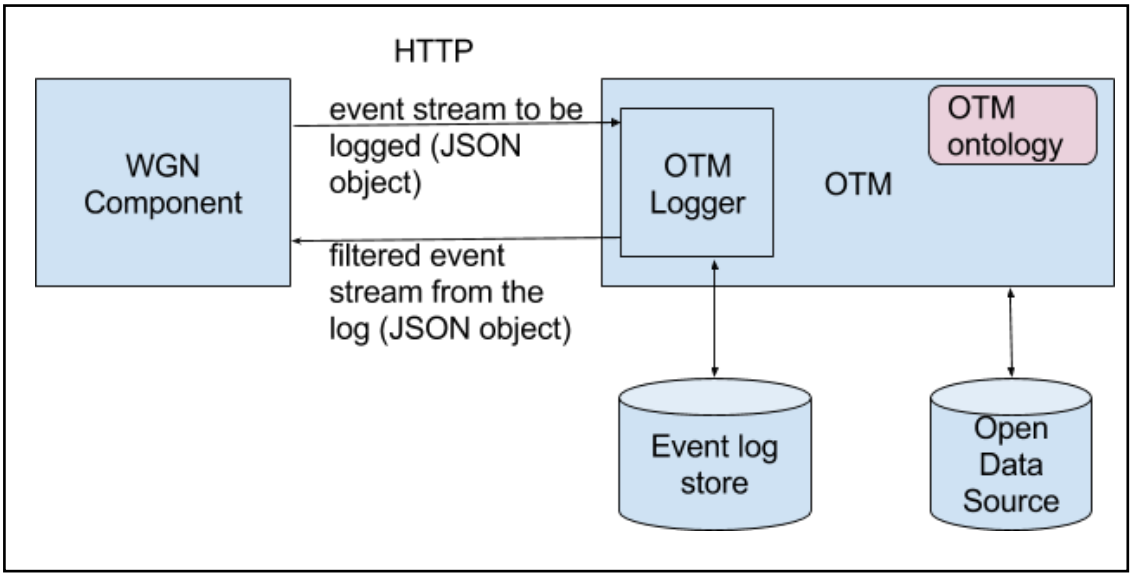
The data retrieval API is fully developed and partially tested with a subset of the WeGovNow applications.

User's activity logger

UNITO has developed the OnToMap Logger. The OnToMap Logger is a centralised collector of the history of actions performed by WeGovNow users by interacting with the WeGovNow applications. The goal is that of achieving a unified perspective on user behaviour and of providing a unified view on the data shared in the WeGovNow platform, including the Open Data managed by OnToMap, in order to enable WeGovNow applications to retrieve information collected about geographical objects, initiatives, issues, and so forth. The OnToMap Logger enables WeGovNow applications to push streams of events to be logged, and to retrieve filtered log information; e.g., the activities performed by a certain user in all the WeGovNow applications on a certain date.

[6] See http://geojson.org/geojson-spec.html#feature-objects.

Exhibit 38. Interaction between a WeGovNow (WGN) component and the OnToMap(OTM) Logger.



For user activity collection, the OnToMap Logger offers an API that allows WeGovNow applications to push user activity events, represented as JSON objects; see Exhibit 38. Events can be pushed synchronously (one by one, as soon as the user actions happen), or asynchronously (pushing a list of events to be stored). The action descriptions specify user activities on data items; e.g., creating, updating, removing, commenting some data objects.

The Logger APIs for logging and for data retrieval are fully developed. Currently, we are carrying out experiments with a subset of the WeGovNow applications to test the interaction between applications in the overall WeGovNow platform.

Only authorized WeGovNow applications must be able to use the Logger APIs for storing and retrieving data and events. For enforcing this requirement, the OnToMap logger makes use of X.509 client certificates, which are used for authorization and authentication of WeGovNow applications. In order to connect to the OnToMap Logger, every application must forward a Certificate Signing Request to LiquidFeedback that will result in a client certificate signed by LiquidFeedback that will have to be included in every request made to the OnToMap Logger APIs.

The OnToMap Logger APIs allow WeGovNow applications to retrieve lists of user action events, sorted from the most recent to the least recent. It is possible to use and combine different filters, to generate list of events which correspond to the specified filter(s); e.g., to retrieve all actions of a specific user in a specific timeframe. It is possible to filter events by the user performing the action, application in which the action was performed, user activity type, interval of time in which the action was performed.

Since, as previously described, the OnToMap Logger is also used to provide a unified view on the data shared among the WeGovNow applications by users; it is possible to filter the event data by some properties of the shared data items associated with the events. This results in the Data Retrieval APIs described in the previous section. Moreover, in addition to the previously cited filters, it is possible to filter the event data by geographical area in which the shared data items are located, the concept of which the shared data items are instances and the unique identifier associated with every shared data item.

The OnToMap Logger documentation can be found at the following URL: https://ontomap.eu/. This documentation reports the specification of the APIs, and examples of usage with sample results.

In the following we report an example showing the invocation of the Logger API with a very small set of sample results (in GeoJSON format): the query concerns the instances of Park, with their descriptions and geometries, as collected by a WeGovNow app denoted as App1. Obviously, more complex queries could be performed, e.g., to specify the bounding box for retrieving instances, and other parameters. See Appendix «Logger Endpoint»

InputMap V2

To enhance integration between WeGovNow components, the consolidated system architecture (deliverable D3.1) introduced a unified location map-based module to input spatial coordinates within WeGovNow components (Exhibit 39), improving the overall look and feel by adopting a common solution platform wide, the overall integration link components' entities to a common geographical data source.

InputMap is a web map module connecting users input on a map (point-based reference) with entities of the geographical data source provided by an ad hoc tile server integrating official open data and OpenStreetMap entities of the project areas (Exhibit 40). InputMap can be used by all components in replacement of common map modules used to collect point-based locations (latitude and longitude), without any specific technical requirements. The second version of InputMap introduces new parameters and features, and a general restyling based on Material Design.

Exhibit 39. InputMap overlaps cartography with the geographical entities available in WeGovNow. User's input is enhanced with references to the selected entity.

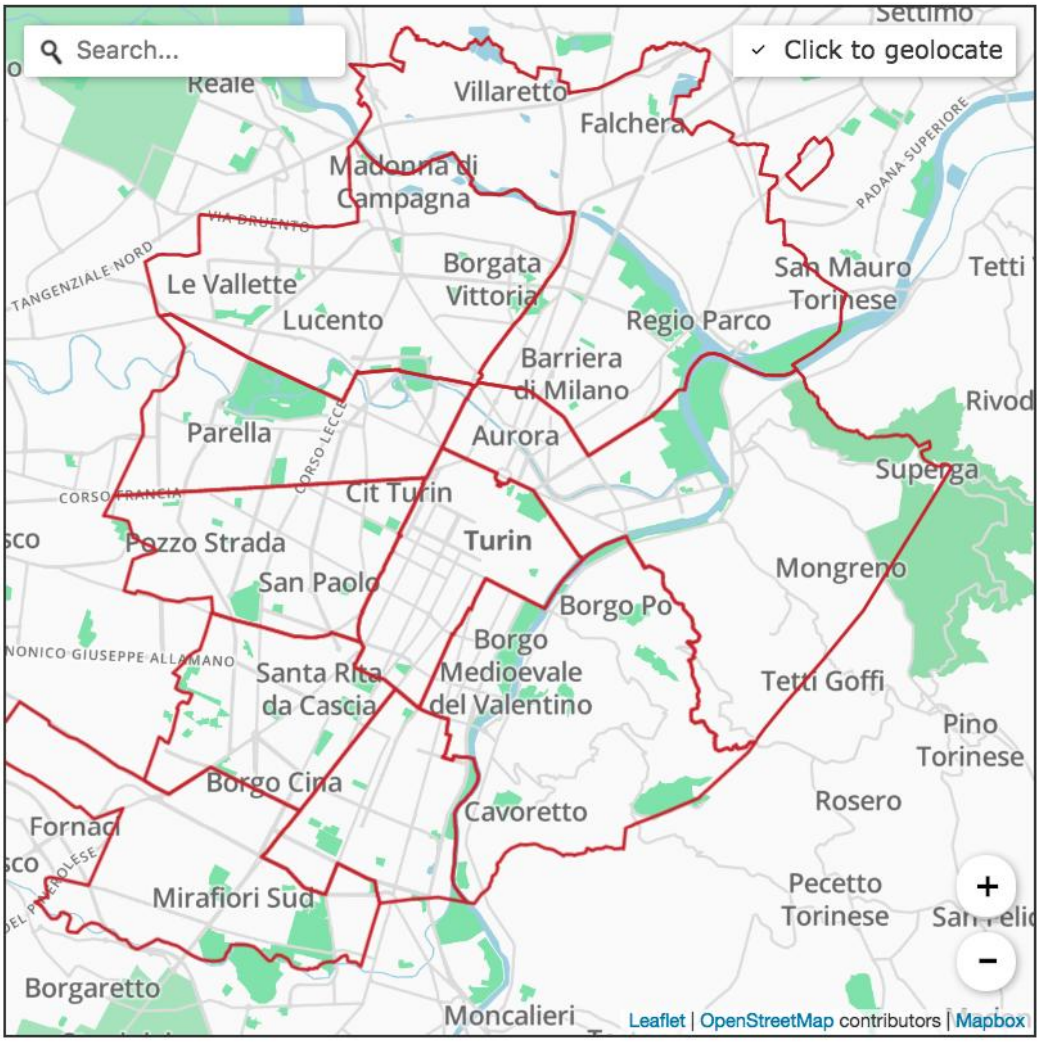
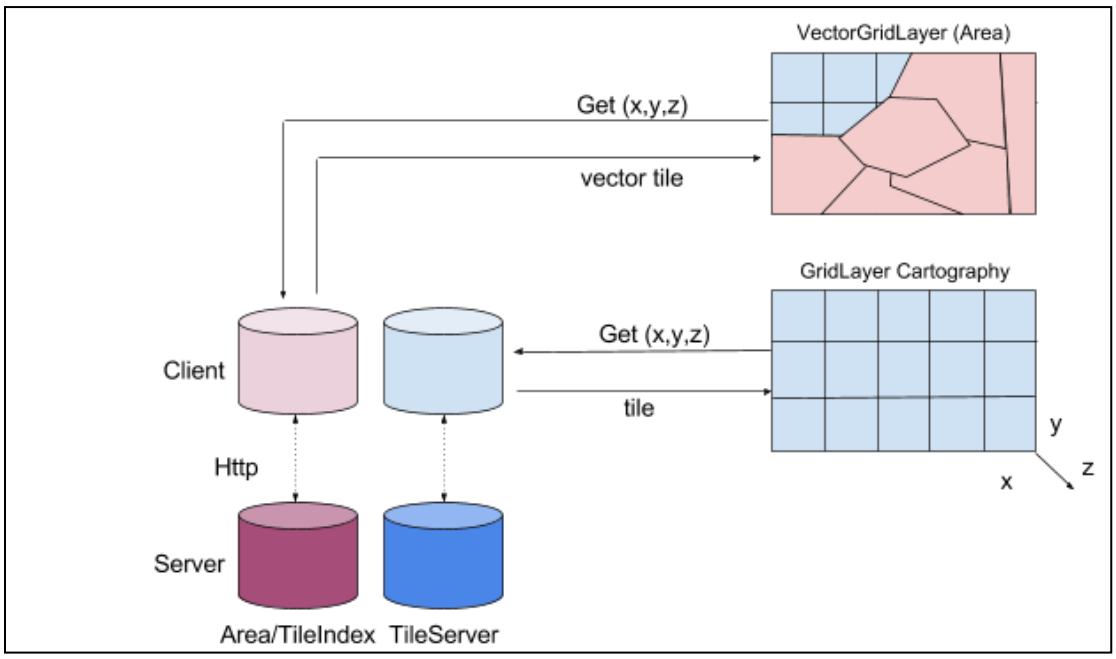
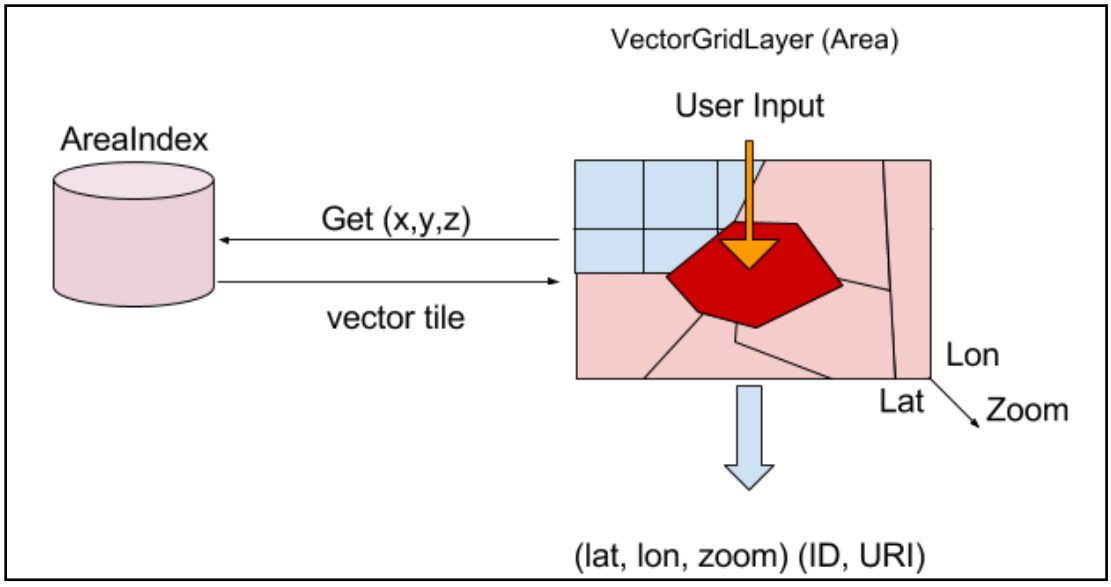


Exhibit 40. InputMap uses a cartographic layer and a vector layer to the interactive layer.The interative layer is a GeoJSON data layer provided by a specific tile server integrating municipalities open data and OpenStreetMap entities.



InputMap is based on Leaflet and Leaflet VectorGrid plugin, to render the data provided by a tile server specifically developed to provide the scale-wise the geographical data available in WeGovNow. InputMap collects user's location input (click on the map, see Exhibit 41) and sends information to the "hosting" application, the component embedding InputMap.

Exhibit 41. InputMap concept: user's input is combined with geographical information retrived from WeGovNow tile server (AreaIndex).



The output of InputMap are (https://gitlab.dev.di.unito.it/firstlife/inputmap):

- Latitude and Longitude
- Zoom level, current zoom of the map
- AreaId, identifier of the selected area (if applicable)
- TileId, hash encode of tile
- OSM Id, identifier of the geographical entity (if applicable)
- Name of the geographical entity
- Type of the geographical entity
- **Address** provided by the reverse geocoding service of OpenStreetMap Nominatim.
- **Display name** of the address provided by the reverse geocoding

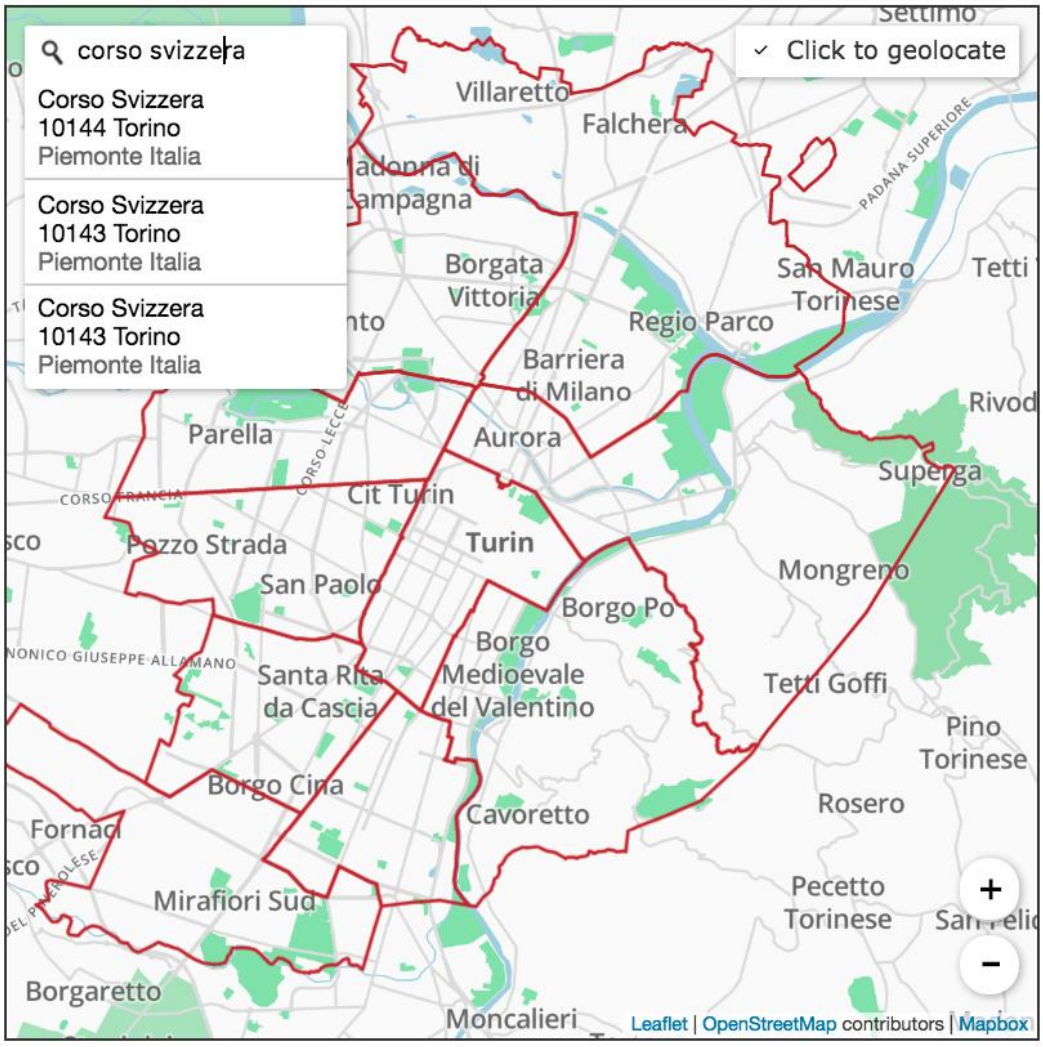
InputMap can be used even when there are not geographical sources available. The extra information about zoom level in combination with latitude and longitude can be used to infer the reference later on, as new geographical sources are available. Tile id is specifically meant to provide a reference to an area, to support the area-based aggregation provided by AreaViewer.

InputMap can be included as embed within any web application or websites as replacement of other web maps. Trough InputMap, users can interact with the geographical entities within WeGovNow collected from OpenStreetMap or Open Data, and select a direct reference between them and the hosting application contents.

V2 of InputMap introduces a geocoding service connected to OpenStreetMap Nominatim web service. The geocoding service is used to support address-based geolocation via search bar, and to enrich the point-based (input on map) location with the nearest address (reverse geocoding).

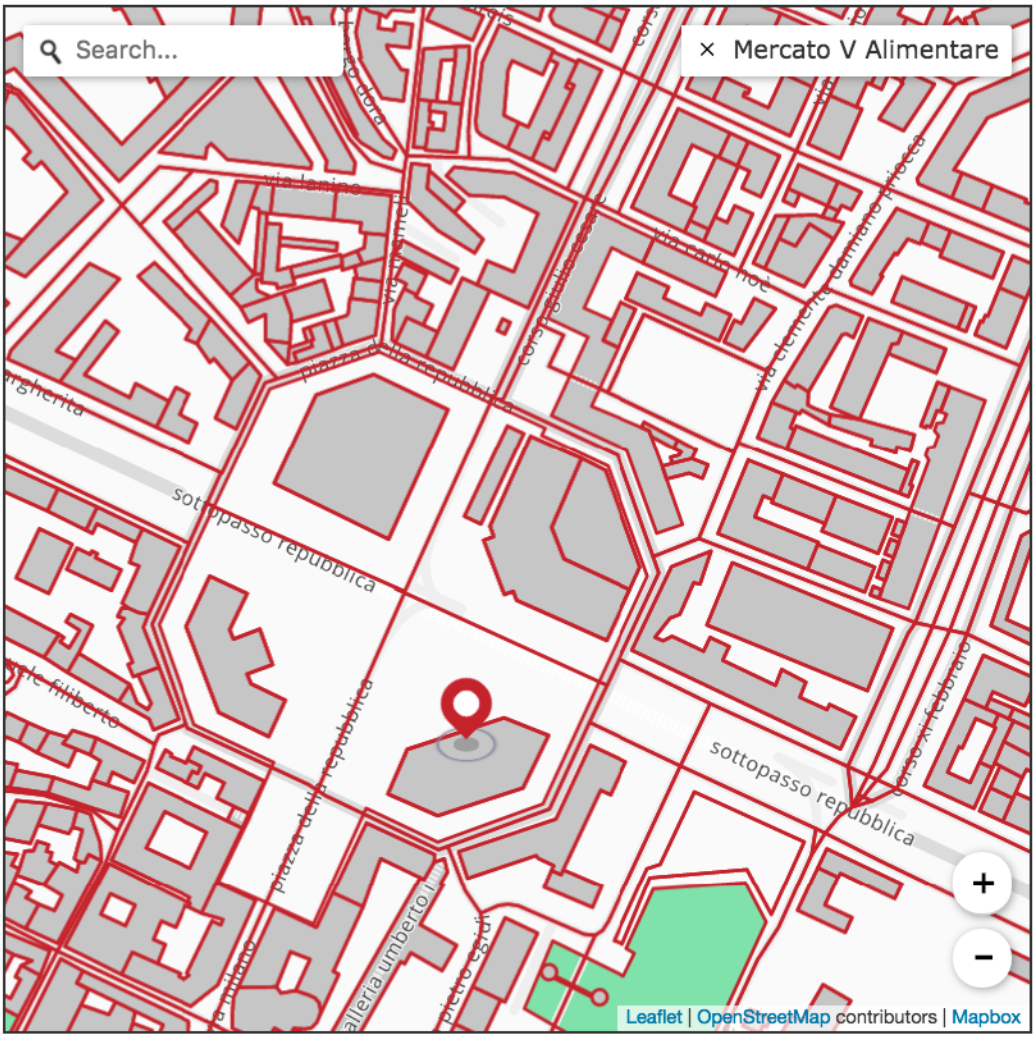
The search bar enables users to search location by address, by click on results users can locate the map (Exhibit 42).

Exhibit 42. InputMap V2 introduce a search bar (top left corner) enabling geolocation by address. This service is based on OpenStreetMap Nominatim search API.



Relying on Nominatim APIs, V2 InputMap introduces the use of reverse geocoding of cordinates, to enrich user's input retrieving the most "close" address to their input on map. The result of reverse geocoding is included in the overall result, and could be used in case the TileServer does not contain information regarding specific coordinates. The result is a resilient system which prioritise the information contained in official open data, but in case non-covered areas or scales, it is able to rely on OpenStreetMap crowd data.

Exhibit 43. InputMap enable users to input different entities according to the selected zoom. E.g. it makes possible to refer explicitly to city district or building.



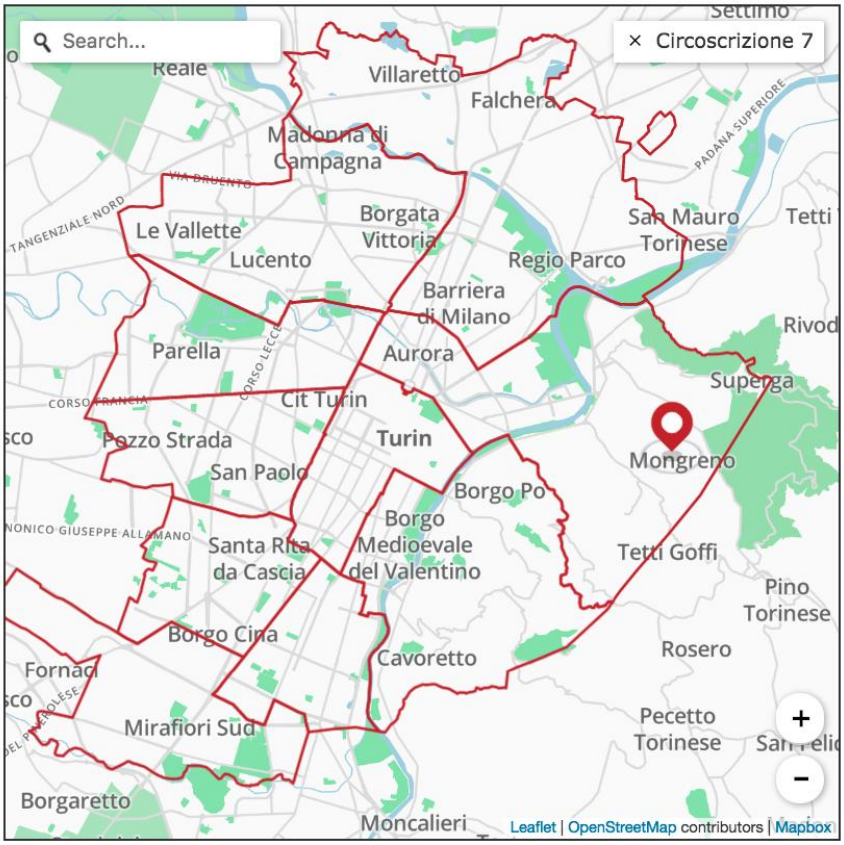
InputMap V2 introduced a label box introducing a tooltip text about how to use inputmap and the current selected location. The language of the tooltip is dynamically chose considering the agent language (browser), or it could be passed as parameter "lang" invoking InputMap. The label box consents to cancel the current selection by clicking "x" button (Exhibit 43). The cancel action sends a "null" message to the hosting application.

To support integration of InputMap, a test page is available at https://inputmap.firstlife.org/test. The test page shows InputMap result messages sent to hosting applications triggered by users' interactions (Exhibit 44).

Exhibit 44. InputMap test page shows InputMap messages corresponding to user's interactions with InputMap.

Test InputMap

InputMap in embed in an iframe tag



Returned object to the hosting window

Area Info

Name: Circoscrizione 7
Area Id: 19286
OSM Id: -2615151
Type: administrative

Input Info

Latitude: 45.0652657154241
Longitude:7.748794555664063
Zoom: 12

InputMap Info

Src: InputMap
Tileid: 2136:1472:12
Tile: [2136,1472,12]

Nominatim

Display Name: Madonna del Pilone, Circoscrizione 7, Torino, TO, Piemonte, 10132, Italia
Address:
{"neighbourhood":"Madonna del Pilone","suburb":"Circoscrizione 7","city":"Torino","county":"TO","state":"Piemonte","postcode":"10132","country":"Italia","country_code":"it"}

TileServer

InputMap and AreaViewer require to access to a common source of geographical data to render an interactive layer on a web map. In order to provide a fast and standard access to web map components, a tile server has been developed and deployed within WeGovNow platform (TileServer).

TileServer provides geographical entities in vector tile format 2.1 (https://www.mapbox.com/vector-tiles/specification/), a open source format widely used to provide vector tiles. TileServer provides other features to support the interconnection between WeGovNow components' entities and the common geographical datasource included in TileServer. Specifically, TileServer provides the following endpoints:

- **/areas/:id** returns the detail of a specific area
- **/area/:z/:lon/:lat** returns the ID of the related area
- **/areas/content/:id** returns a feature collection of contained areas
- **/tile/:z/:x/:y** returns a vector tile in PBF format[7]

Currently, TileServer datasource include the official open data and OpenStreetMap layers of the municipality of Turin.

The end point of TileServer can be found at https://tiles.firstlife.org/, source code and documentation can be found at the project repository https://gitlab.di.unito.it/firstlife/tileserver/ . TileServer is an ExpressJS application, using libraries and tools developed by Mapbox and Leaflet communities.

[7] https://developers.google.com/protocol-buffers/docs/encoding

6. Ongoing development work

During the definition of The WeGovNow Consolidated System Architecture (deliverable D3.1), the technical teams of WeGovNow identified a set of requirements to support the integration of WeGovNow components.

The engagement activities of stakeholders at the trial sites and the requirement elicitation process brought us to the definition of a set of general needs, preliminary requirements, daily-based use cases, and expectations based on the stakeholders' intuition of WeGovNow technologies, summarised in deliverable D2.3, Annex II. The collected inputs had been assessed by technical teams, under the light of the scope of the project, of the availability of time and resources, of the existing components and their workflows, resulting in a list of indications for further development of WeGovNow platform.

The assessment (see Annex 6 "Extending WeGovNow platform"), was supported by interviews with the goal of:

1. extend service scenarios from a specific application example to a general pattern of use
2. mapping the fragmented daily use cases in steps of full service processes, involving multiple actors in wide-range of time
3. collecting the missing information about the context and work procedures, important for the adoption of the platform
4. extracting the underlying workflows from the service patterns
5. clustering similar workflows in common we-government processes
6. mapping workflows' steps with components, to highlight missing features and required extensions
7. validation of workflows and corresponding platform functionalities with a subset of local municipalities (up to now with Turin and San Donà di Piave)

The result of the assessment lead to clusters of requirements (see Annex 6 "Extending WeGovNow platform"):

- Transversal requirements for the adoption of WeGovNow platform by local municipalities,
- Functional requirements to implement the applicative scenarios through WeGovNow components

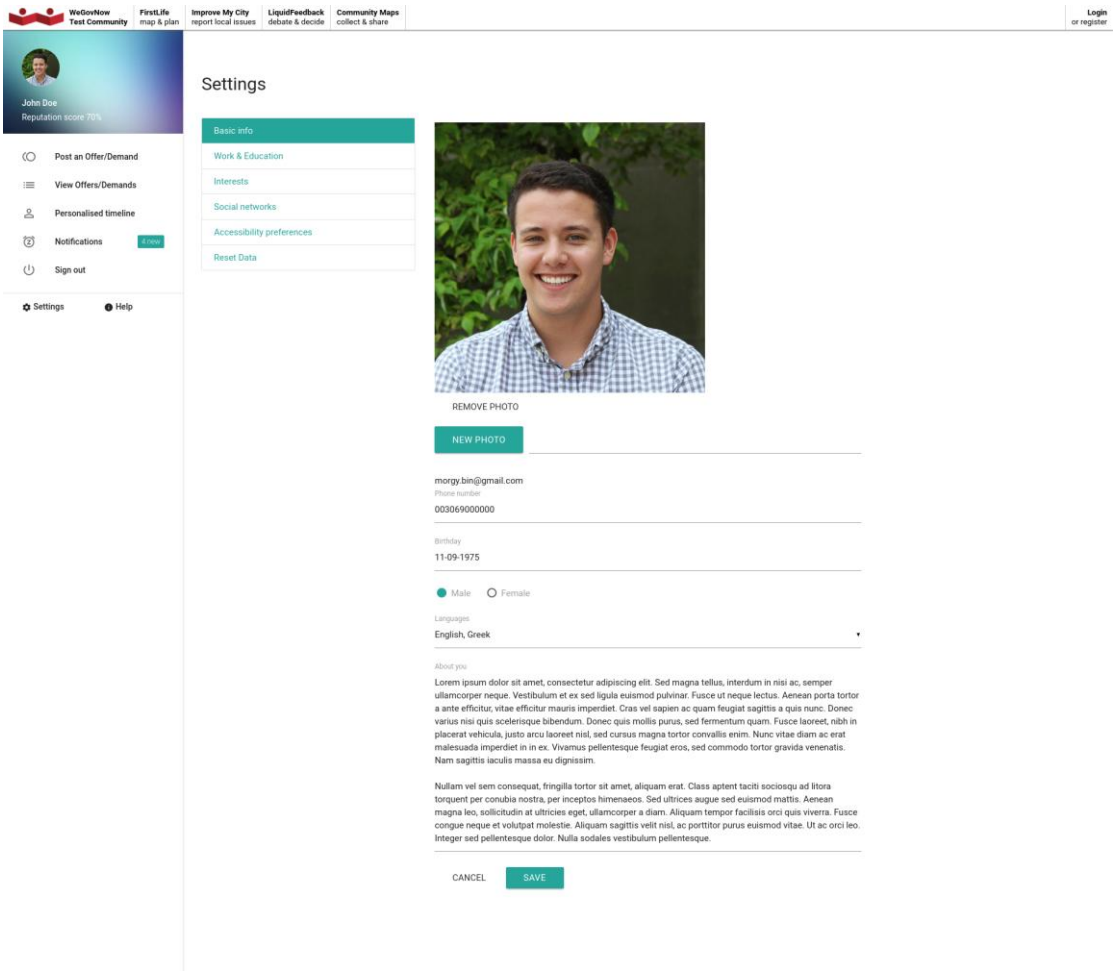
Those requirements will lead the next steps of development of the WeGovNow platform, toward the final prototype and the first release of the platform, which will be pilot in the local sites.

In the following subsections, further information is provided on those modules and components which are currently under development. These are expected to be released before the final version of WeGovNow platform.

6.1. User Management

User Management is a component introduced as extension of Trusted Marketplace (TM) to provide a single entry-point to user's preferences and data across WeGovNow platform. User Management will extend the profile settings in UWUM (see Exhibit 45), including the possibility to edit all information stored in UWUM about the user by all WeGovNow components.

Exhibit 45. The User Profile Management page, extending the UWUM profile settings.



User Management, besides basic info, it includes other sections such as:

- Work & Education (see Exhibit 46)
- Personal interests (see Exhibit 47)
- Social network accounts (see Exhibit 48)
- Accessibility preferences (see section 6.2)

and the possibility to reset user data on demand

It should be noted that under the hood, the system provides with more than 60000 predefined values which are selected automatically while the user types. This helps a

lot to identify (match) for example people with similar interests. Of course, users are able to type free text.

Exhibit 46. Section of user's "work and education".

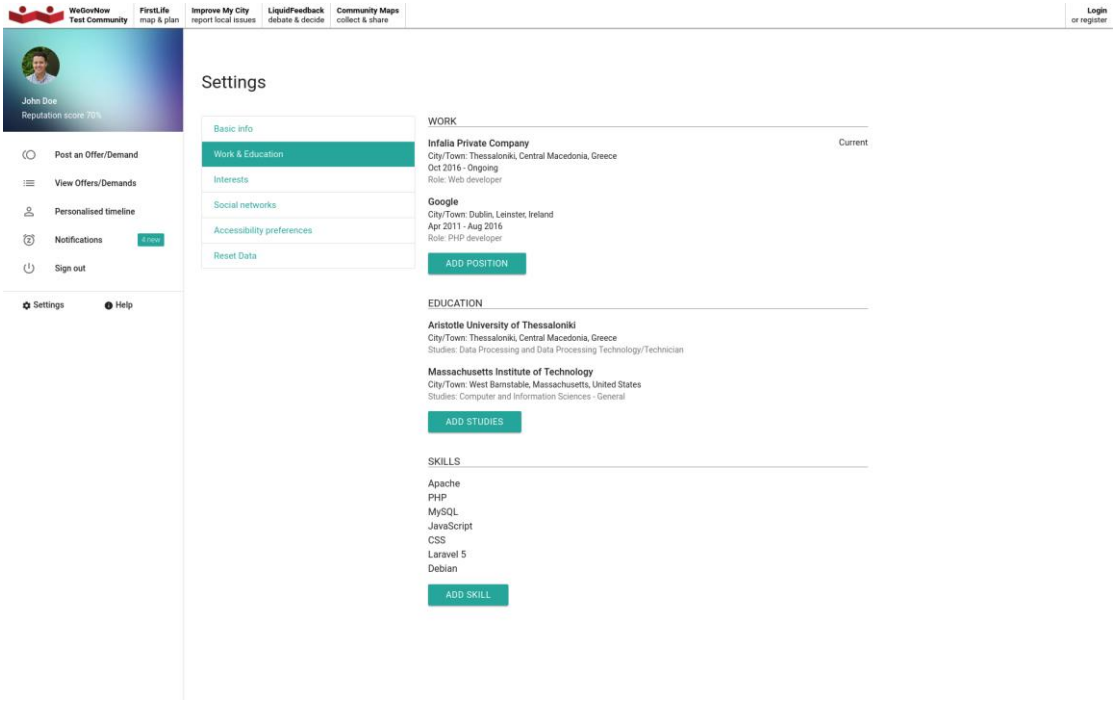
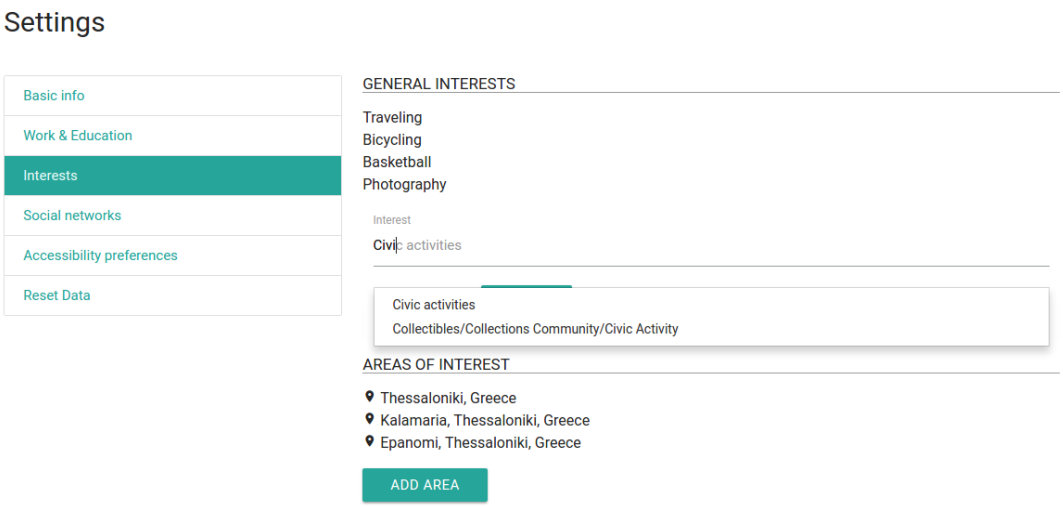


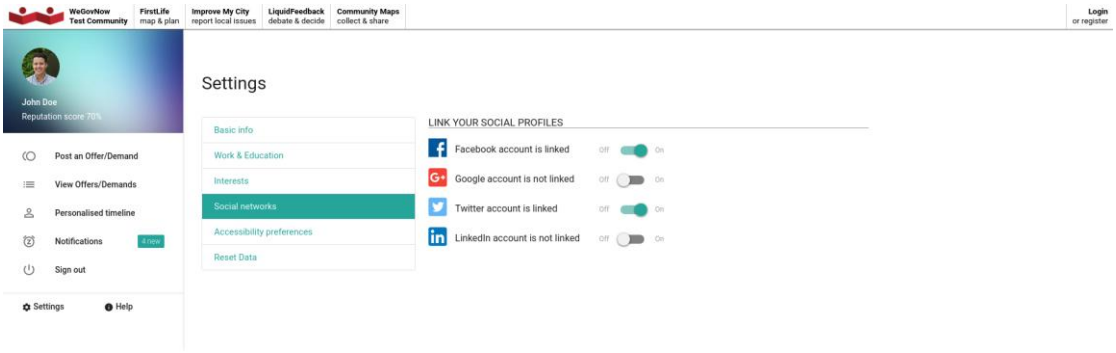
Exhibit 47. Section of user's interests.



Social Networks account

Through the User Management as an extension of Trusted Marketplace (TM), users are able to link to their user profile various social networks as depicted in Exhibit 48.

Exhibit 48. Link social accounts.



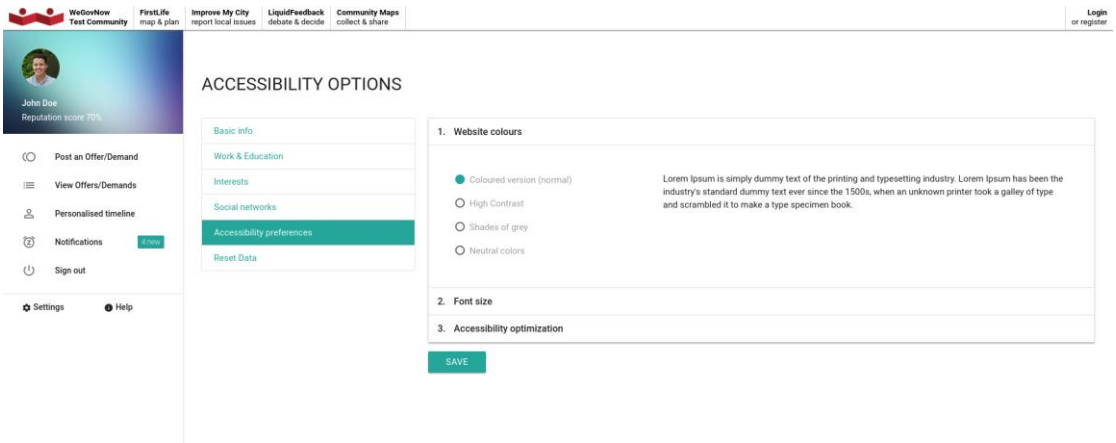
The purpose of linking social accounts is to collect most information about the users and thus enhance the match-making procedure. At any time, users are able to unlink these accounts.

6.2. Accessibility Preferences

The accessibility preferences will help users to setup WeGovNow platform accordingly to their specific needs, the information collected by the wizard will be available to all components to dynamically setup the correct view for the user (see Exhibit 49).

Basic support for accessibility is extended from specific features of components to the overall WeGovNow platform. Specifically, users' preferences about text size, contrast, languages are collected once and shared among WeGovNow components through UWUM's user settings.

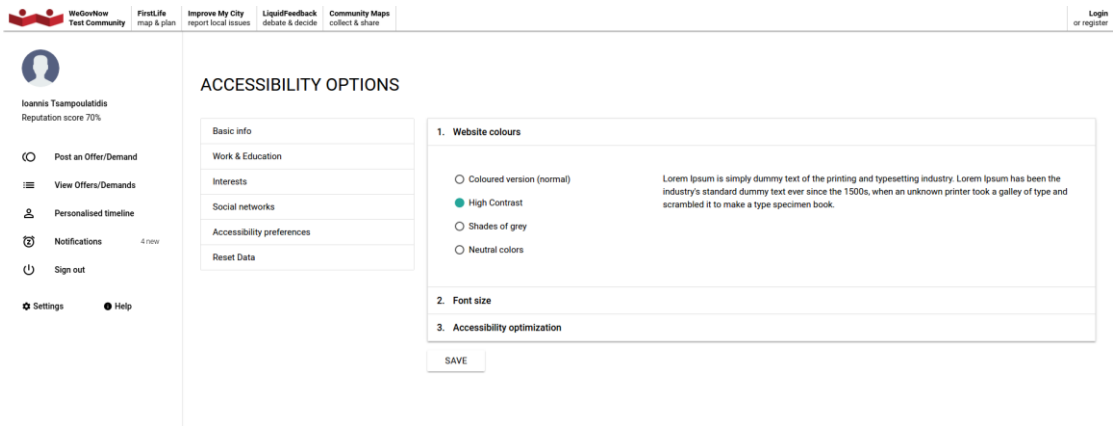
Exhibit 49. The accessibility settings will help users to setup WeGovNow platform accordingly to their specific needs, the information collected by the wizard will be available to all components to dynamically setup the correct view for the user.



User's preferences are collected through a wizard interface meant to guide users in defining their best condition of use. The update of preferences is propagated via UWUM, editing the user settings.

The following exhibit (Exhibit 50) depicts the updated UI after user selected "high contrast" through the accessibility preferences.

Exhibit 50. Selecting "high contrast" in the accessibility preferences to help visual impaired users



At this stage, accessibility preferences are applied only on Trusted MarketPlace but the functionality is expected to adopted to other components through UWUM key-value pairs storage mechanism (not yet implemented).

6.3. Trusted Marketplace

The Trusted Marketplace is being designed and implemented from scratch according to the requirements and needs of the pilot use cases. Besides the major role of Trusted Marketplace that is the match-making of users-to-events and users-to-users and the handling of personalised notifications (based on the match-making suggestions), Trusted Marketplace also incorporates features that are not available by the other core components. These features include:

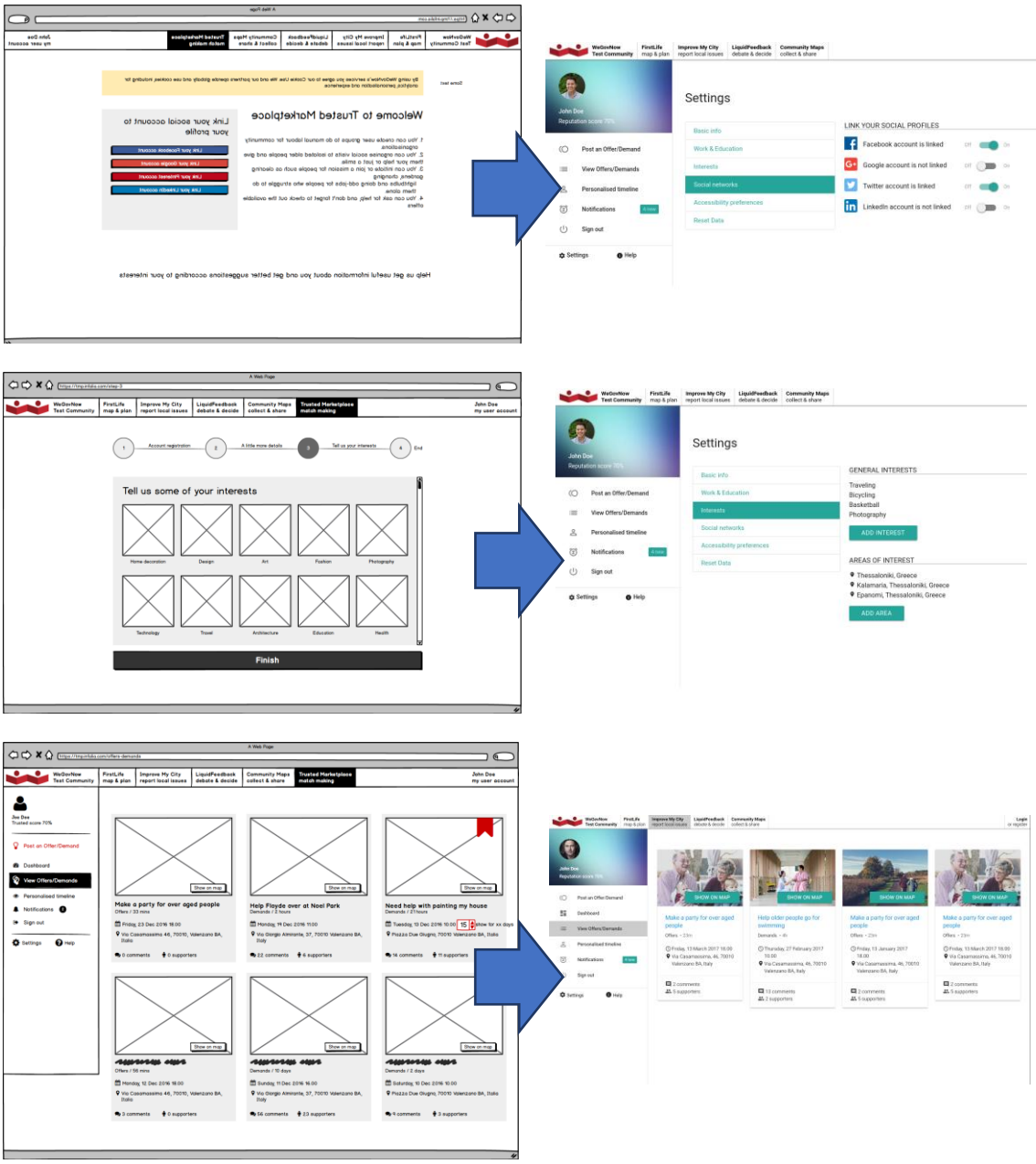
- Enhanced user profile management (see sections 6.1)
- Mechanism to handle demands and offers with a built-in reputation system.
- Graphical representation of the personalised timeline by aggregating OnToMap Logger user actions/activities in a friendly interface.
- Mechanism to define, globally in the overall WeGovNow platform, personalised accessibility preferences to promote the 'Design For All' principles and guidelines. (see section 6.2)
- A friendly dashboard for easy overview of all the above.

The Trusted Marketplace engine receives input from three different sources.

1. Indirectly from any other core component (including the “Offers & Demands Organiser, part of Trusted Marketplace)
2. Directly from the users through the “Dashboard” and their personal profile, preferences and interests that are explicitly declared by them.
3. By social networks, on user’s consent, through the “Social Media Linker & Collector”.

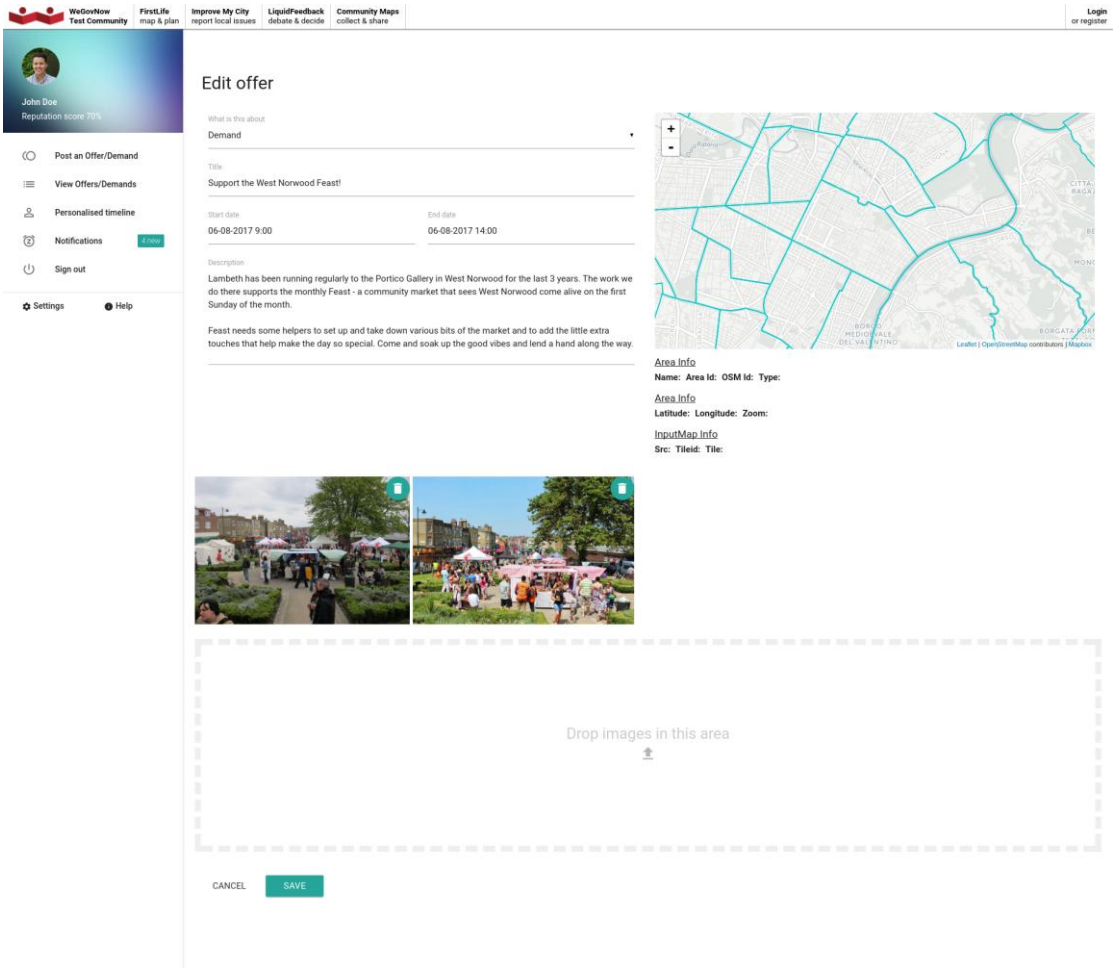
In order to show the current development status, in the following Exhibit 51 are comparisons between indicative mockups of the Trusted Marketplace as they presented in D3.1 (M12) and the actual current implementation.

Exhibit 51. Trusted Marketplace. Mock-ups vs Actual implementation.



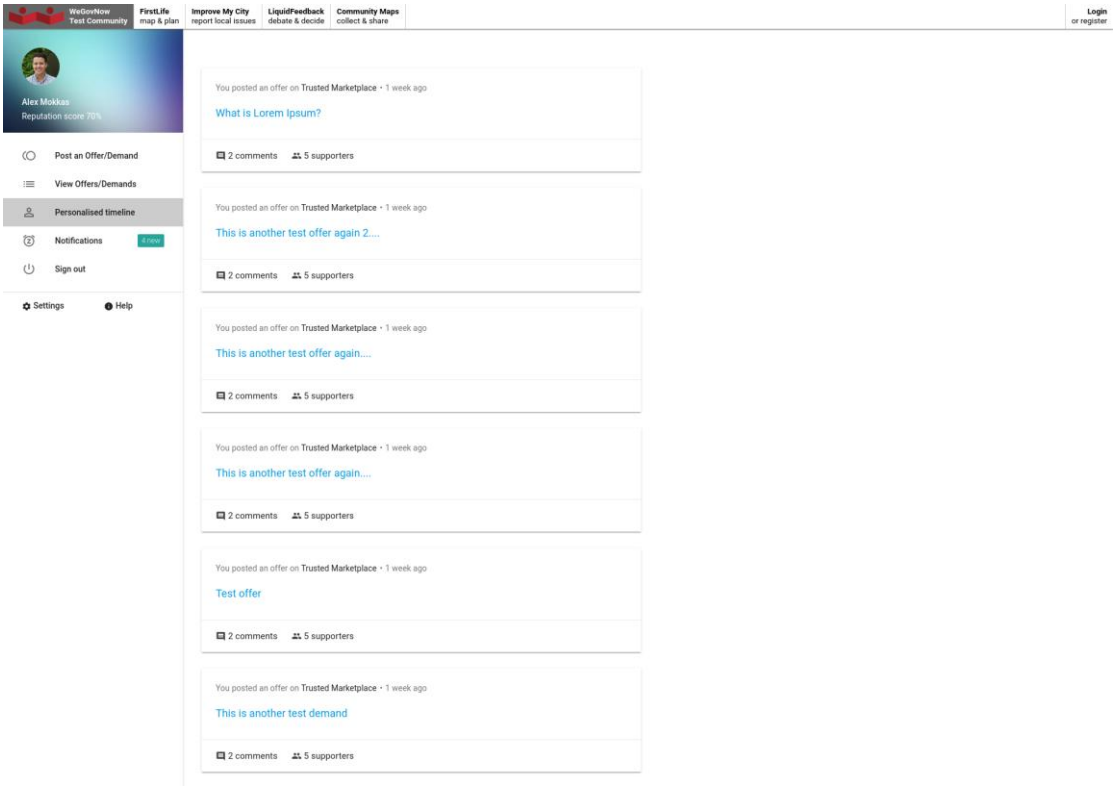
The integration of Trusted Marketplace with InputMap on "Offer/Demands", is depicted in the following Exhibit 52.

Exhibit 52. Trusted Marketplace integrates InputMap already



In addition, Trusted MarketPlace is linked with OnToMap. The following Exhibit (Exhibit 53) displays the personalized timeline of user actions as pulled from OnToMap. Currently, user actions of TM are pushed to OnToMap, and aggregated actions from every component are displayed inside TrustedMarket (pull actions from OnToMap)

Exhibit 53. Personalised Timeline: Trusted Marketplace integrates with OnToMap



6.4. AreaCalendar

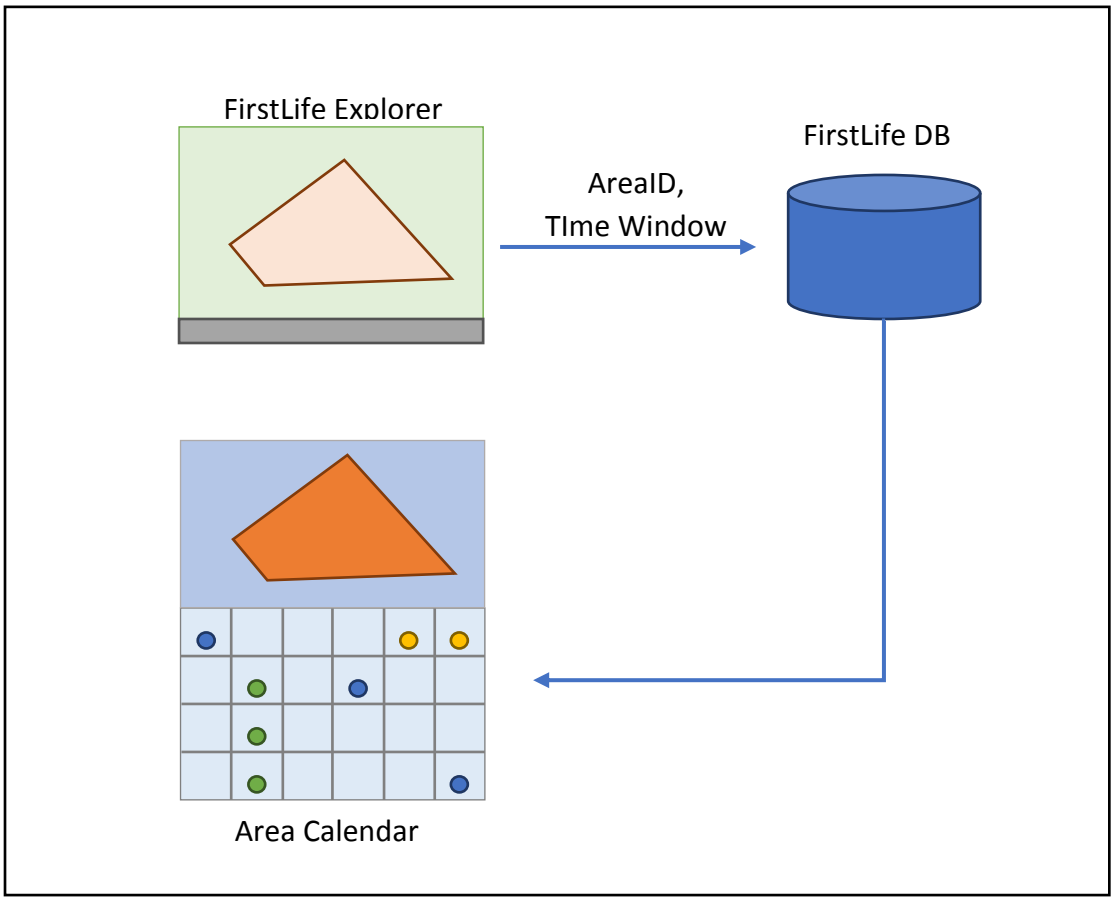
To address a common use case reported by the requirement elicitation process, FirstLife will be extended with a calendar module and periodicity functionality for recurrent events. Area calendar will provide a global calendar of the area, shared by all actors and groups acting on a specific area (Exhibit 54). Area calendar will be a tool to coordinate the scheduling, search and document public activities, accessible via FirstLife.

Technically, area calendar will be a web module of FirstLife querying FirstLife APIs in search of temporal events of a specific area, in a given time window defined by users. To represent a wider range of urban activities, FirstLife is currently being extended to support periodicity expression, such as:

- Every Friday between 15th of July and end of October
- Every 12 AM between Monday and Friday, Thursday excluded
- The first week of the month

FirstLife temporal extension is extensively described in “Temporal indexing of urban entities”, Annex 7.

Exhibit 54. Concept of AreaCalendar: user will select an area of interest from FirstLife, then go to AreaCalendar to explore the local activities on a calendar view.



7. Conclusion and lesson learned

The road to a production-ready version of WeGovNow platform includes three major milestones in terms of three prototype versions of the platform. In general, the purpose of releasing three prototypes is to consolidate, check and test the improvements made by the different development teams involved in WeGovNow project. Specifically, each prototype addresses one phase of the life-cycle of the development process.

The scope of the 1st prototype of WeGovNow platform was to be a proof of concept of the integration of the existing components of WeGovNow, in the technical framework of the platform (as described in "Consolidated System Architecture" D3.1).

The current 2nd prototype of WeGovNow will be useful to stress new modules and features we designed and developed within WeGovNow project (new components such as the Landing Page and Trusted Marketplace).

Finally, the 3rd and final prototype will include the bilateral integrations between existing and new components, and it will be used for testing focused on the integration of the platform.

Setting up and running multiple instances of WeGovNow platform is part of the technical testing of the platform itself. So far, we managed to solve various minor issues related to the extension of WeGovNow components to support the general architecture, such as CORS errors[8], miss use of https certificates and the like.

In particular, in order to run multiple instances of WeGovNow platform, each development team had to develop a methodology to provide:

- Data isolation
- Instance-based security settings
- Instance-based endpoints
- Version consistence

Currently, we are addressing one issue related to the setup of multiple instances of GeoKey (GeoKey could only set up one instance) which made it impossible to include GeoKey/CommunityMap software in the prototype 2 instance. This specific example demonstrates the value of the decision made in the project proposal of providing multiple prototypes of WeGovNow platform, keeping them up and running in parallel months before the piloting of the platform.

In terms of software development, those issues are not critical in general: the only real risk is to find them out too late to address them properly.

[8] https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS

Annexes

List of Annexes:

1. Logger Endpoint
2. UWUM integration
3. LiquidFeedback work report
4. UWUM work report
5. PgLatLon work report
6. Extending WeGovNow platform
7. Temporal indexing of Urban Entities

Annex 1

Logger endpoint

1. Logger Endpoint

Query example[9], GET
https://api.ontomap.eu/api/v1/instances/Park?descriptions=true&geometries=true&applications=app1.wegovnow.eu

Result


```
{ "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "geometry": {"type": "Point", "coordinates": [45.5, 7]},
      "id": "https://app1.wegovnow.eu/Parks/1",
      "application": " app1.wegovnow.eu ",
      "properties": {
        "external_url": " https://app1.wegovnow.eu/Parks/1",
        "hasType": "Park",
        "hasAddress": "Address for Park 1",
        "label":  "Parco 1"
      }
    },
    {
      "type": "Feature",
      "geometry": {"type": "Point", "coordinates": [45, 7.5]},
      "id": "https://app1.wegovnow.eu/Parks/5",
      "application": "app1.wegovnow.eu",
      "properties": {
        "external_url": "https://app1.wegovnow.eu/Parks/5",
        "hasType": "Park",
        "hasAddress": "Address for Park 5",
        "label": "Park 5"
      }
    }]}
```


[9]It should be noticed that the example query represents the kind of interaction between applications but its execution from a browser is going to fail because, in order to succeed, the invoking application must provide a client certificate signed by LiquidFeedback. Notice also that the invoking application, and the requested concept, are fictitious.

Annex 2

UWUM Integration

2. UWUM Integration

UWUM API endpoints

The UWUM endpoints for OAuth2 and the integration framework are available at the following URLs:

https://wegovnow-pt2.liquidfeedback.com/api/1/authorization

https://wegovnow-pt2.liquidfeedback.com/api/1/token

https://wegovnow-pt2.liquidfeedback.com/api/1/validate

https://wegovnow-pt2.liquidfeedback.com/api/1/navigation

https://wegovnow-pt2.liquidfeedback.com/api/1/style

https://wegovnow-pt2.liquidfeedback.com/api/1/application (formerly "client")

UWUM scopes

Currently the following access scopes can be granted through UWUM (based on the access levels of LiquidFeedback):

- “authentication” authenticate the current user by reading its
  - static ID (id)
  - current screen name (name)
- “identification” identify the current user by reading its unique ident string (identification). Automatically implies scope "authentication"
- “notify_email” read the notification email address of the current user (notify_email)
- “read_contents” read any user generated content (w/o authorship, ratings and votes)
- “read_authors” read the author names of user generated content (author's static ID and screen name)
- “read_ratings” read rating scores by other users
- “read_identities” read the identities of other users (identification)
- “read_profiles” read the profiles of other users (e.g. phone number, self-description)
- “post” post new content
- “rate” rate user generated content (e.g. thumbs up/down, "+1", support an initiative, rate a suggestion)
- “vote” finally vote for/against user generated content in a decision (i.e. vote on an issue)
- “profile” read profile data of current user (e.g. phone number, self-description, ...)

- “settings” read current user's settings (e.g. notification settings, display contrast, ...)
- “update_name” modify user's screen name (name)
- “update_notify_email” modify user's notification email address (notify_email)
- “update_profile” modify profile data (e.g. phone number, self-description, ...)
- “update_settings” modify user settings (e.g. notification settings, display contrast, ...)

Any of these scopes can be suffixed with "_detached" to request the scope for usage without the need for the user to be logged in. This should only be used when it is really needed.

X.509 certificate for client identification

To create a trustworthy relationship between applications using UWUM and the central UWUM component, we will use X.509 certificates. Therefor, any official WeGovNow client is required to provide a valid X.509 certificate with each request made to the central UWUM service. For this purpose we kindly ask all technical partners to provide X.509 certificate signing requests to be signed by the UWUM Certificate Authority.

For more information on X.509 certificates and signing requests, please refer to the documentation of your preferred TLS software such as LibreSSL or OpenSSL.

Integration Checklist

We will support all technical partners during UWUM integration. We defined a number of steps we like to take together with every technical partner. In the following list, the term "client application" refers to the application to be integrated with UWUM:

1. **Availability of application via IPv4**. The client application is available via a defined URL using IPv4.

2. **Availability of application via IPv6.** The client application is also available using IPv6.

3. **Serving via HTTPS.** The client application service is encrypted via HTTPS.

4. **Publicly trusted X.509 certificate for end users**. The client application server provides a publicly trusted X.509 certificate.

5. **OAuth2 redirection endpoint defined**. The URL of the OAuth2 redirection endpoint of the client application has been determined and submitted to LiquidFeedback (FlexiGuided GmbH).

6. **Certificate signing request for UWUM**. A private key for accessing the UWUM API has been created. A corresponding certificate signing request (CSR) has been submitted to LiquidFeedback (FlexiGuided GmbH).

7. **Certificate signed by UWUM Certificate Authority.** A signed certificate for the client application has been sent back to the technical partner.

8. **Successful X.509 secured connection**. The client application has successfully established a secured connection with the UWUM server, e.g. using LiquidFeedback's /info API endpoint.

9. **Authorization endpoint access**. The client application can redirect an end user to the UWUM authorization endpoint.

10. **Authorization endpoint error response handling**. The client application is capable of receiving authorization errors through its OAuth2 endpoint.

11. **Authorization endpoint error display**. The client application is able to display authorization error messages (see 10) to the end user.

12. **Successful authorization request and user identification**. The client application made a successful authorization request, received a UWUM access token, and determined the end user ID.

13. **Using access token for API calls to other components.** The client application has successfully performed a LiquidFeedback API call (e.g. to the /info API endpoint) using a previously obtained UWUM access token.

14. **Accepting access token from other components**. The client application (acting as resource server) provides at least one API call which accepts a UWUM access token for authorization.

15. **Access token verification**. The client application (acting as resource server) is capable of verifying the validity and scope of a UWUM access token passed from another component (see 14).

16. **Access token verification errors**. The client application (acting as resource server) is capable of handling error responses during validation of UWUM tokens (see 15).

17. **Accepting access tokens as "Authorization" header**. In conformance with RFC 6750 (Bearer Token Usage), the client application (acting as resource server) accepts UWUM access tokens through the HTTP request header field "Authorization".

18. **Cross-origin resource sharing (CORS).** The client application (acting as resource server) allows cross-origin resource sharing (CORS). See also https://www.w3.org/TR/cors/.

19. **HTTP Strict Transport Security (HSTS)**. The client application ensures secure access by using HTTP Strict Transport Security (HSTS) according to RFC 6797.

20. **Cross-application navigation**. The UWUM navigation bar has been successfully integrated into the client application.

Usage of scopes / screen name

When acting as WeGovNow UWUM client without data exchange with other WeGovNow applications, you will only need to request the "authentication" or the "identification" scope to be able to identify the user. These scopes allow to retrieve some user related information (i.e. numerical ID, identification string, screen name) to identify the user. When using the "authentication" or the "identification" scope, the response of the /api/1/token endpoint can optionally include an additional data structure providing member information. To request this optional "member" data structure, you need to set the parameter "include_member" to 1 or "true". Using the "identification" scope with the parameter "include_member" set to true, the response to the /api/1/token endpoint could look like as follows:


```
{
   "access_token": "UFYPzKrz7JHIKATI",
   "expires_in": 3600,
   "refresh_token": "5QM8OL7AbdabXusG",
   "token_type": "bearer",
   "member_id": 123,
   "member": {
      "id": 123,
      "name": "Johnny",
      "identification": "John Doe"
   }
}
```


The field "id" of the "member" object contains the static numerical ID of the user (equal to "member_id", i.e. redundant), the field "name" contains the screen name chosen by the user, the field "identification" contains the identification string set by the authority which identified the user as unique and eligible to use the WeGovNow application. In future, there may be more fields according to the upcoming specification of the /api/1/member endpoint of LiquidFeedback.

The parameter "include_member" can also be used at the /api/1/validate and the /api/1/info endpoints.

When acting as WeGovNow application using user related data or services of other WeGovNow applications, you will need to request the appropriate scopes from UWUM for the types of actions you want to perform with other WeGovNow applications (e.g. if you want to post new content to other applications, request the scope "post"; if you want to rate content in other applications request the scope "rate"; ...)

When acting as WeGovNow resource server (i.e. when offering user related data or services to other WeGovNow applications) you need to check (via the /api/1/validate endpoint) the scopes of the access token you received from the requesting application (e.g. if another application tries to post content for a user, check for scope "post"; if another application tries to rate content, check for the scope "rate").

Handling of updated user related data / user's email addresses

When a WeGovNow application wants to send notification emails to users, it is not adequate to retrieve the email address only once from UWUM as the notification email can be changed by the user at any time. Such a change needs to be reflected by all applications using this email address. Therefore, an application needs to retrieve the current notification email address *directly* before using it, in fact again before every usage.

For that purpose, the newly introduced API endpoint GET /api/1/notify_email can be used (using an access token with the "notify_email" scope). To be able to retrieve the email address while the user is not currently logged in, it will be neccessary to request the "notify_email_detached" scope when identifying the user and to store the received refresh token permanently. The suffix "_detached" requests a scope for detached usage, i.e. for usage even after the user logs out. Please note, when exchanging a refresh token for an access token after the user has been logged out, you must explicitly request the "*_detached" scope(s) you need, e.g. "notify_email_detached" using the scope parameter of the /api/1/token endpoint.

Similar situations can occur related to other member properties stored in one application but used in another one, e.g. the screen name. But these seem not to be as critical as to avoid using an outdated email address. Such properties could be cached for a limited time before retrieving them again from the application storing this property.

Sustainability, unregistered third-party clients and the future

Following these rules, even a complete new (non-registered, third-party) application can easily make use of the WeGovNow infrastructure. The application can request certain scopes from UWUM (which can be granted or declined by the user) and use the appropriate services of all other WeGovNow applications. Using the upcoming application and service discovery, this is also possible vice versa.

Scopes vs. User Privileges

NB: The scope does NOT grant a privilege to a user, it just means an application can trigger an action within the scope *if* the user is authorized to perform the action. Example voting: an application needs the scope "vote" to cast a vote on behalf of the user but casting a vote will only work if the user has the necessary voting privileges. You can think of this as a matrix of scopes and user privileges or (alternatively) as a logical AND conjunction. Scopes control that an application does not misuse user privileges: while the trusted WeGovNow applications can request certain scopes without user interaction, a non-trusted third-party application/client would trigger a request for a confirmation by the user "Do you want to allow application X to cast votes on your behalf? [yes, one time / yes, permanently]" (compare to permissions for third party Twitter/Facebook clients and Android permissions).