CITYkeys

Co-funded by the European Commission within
the H2020 Programme.
Grant Agreement no: 646440.
2015-02-01…2017-01-31 (24 months).

Deliverable 2.2 Specifications

Revision 0
Preparation date 2016-03-31 (m14)
Due date 2016-03-31 (m14)
Lead contractor............................. VTT

Authors:

Kalevi Piira.................................... VTT
Aapo Huovila .... VTT
Isabel Pinto Seppä........................ VTT
Miimu Airaksinen .... VTT
Henri Biström................................ VTT
Topi Penttinen............................... VTT
Mikko Tuomisto............................. VTT
Hans-Martin Neumann.................. AIT
Stefan Vielguth .... AIT
Eva Pangerl .... VIE

Dissemination level		
PU	Public	X
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)	

Deliverable administration

No & name	**D2.2 Specifications**				
Status	Final	**Due**	m14	**Date**	2016-03-31
Author(s)	Kalevi Piira, VTT, Aapo Huovila, VTT, Isabel Pinto Seppä, VTT, Miimu Airaksinen, VTT, Henri Biström, VTT, Topi Penttinen, VTT, Mikko Tuomisto, VTT, Hans-Martin Neumann, AIT, Stefan Vielguth, AIT, Eva Pangerl, VIE				
Description of the related task and the deliverable in the DoA	T2.2 Methodology for data collection and indicators calculation The task work will focus on specifying the methodology to collect the needed data and to calculate the performance measurement indicators. Consolidating WP1 and T2.1 results these task will define the algorithms for the KPIs calculation. The work will analyze existing systems and available open platforms in order to integrate these, and build on top of, these current available systems. The final result will be the specifications for the collection and calculation of the key performance indicators. D2.2 Specifications Specifications for the collection system and calculation methodologies for the performance measurements				

Planned resources PM	VTT	AIT	TNO	EUR	TAM	ROT	VIE	ZGZ	ZAG	Total
	4	2					0,25			
Comments										

V	Date	Authors	Description
0	2016-01-26	Kalevi Piira	1st draft contents
0	2016-02-05	Kalevi Piira	Draft distributed to partners
0	2016-03-08	Kalevi Piira	Consolidated version for WP leader review (all partner's contributions added)
0	2016-03-18	Kalevi Piira	Added chapter 2 (CITYkeys platform overview) and some fixes to Executive summary. Only limited number of review comments fixed yet.
0	2016-03-28	Kalevi Piira	Document updated by review comments. In addition new chapters added: 4.2.2, 4.2.3 and 4.2.4 (CITYkeys dataset specifications). Chapter 5.2 and 8 updated.
0	2016-03-29	Kalevi Piira	Final draft Chapter 6 (Example of the methodology application in a lighthouse project) updated.
1	2016-03-31	Aapo Huovila	Final version for submission

Table of contents

Executive summary 1

1. Introduction 2
1.1 Purpose and target group 2
1.2 Contributions of partners 2
1.3 Baseline 2
1.4 Relations to other activities 3

2. Existing platforms for smart cities 4
2.1 CKAN 4
2.2 Socrata 7
2.3 SCIS 8
2.4 FIWARE 9
2.5 CitySDK 12

3. CITYkeys platform overview 14

4. Methodology to collect indicator data 16
4.1 Main concept 16
4.2 Dataset readers 17
4.2.1 Open data sources in CITYkeys partner cities 17
4.2.2 General CITYkeys datasets 19
4.2.3 CITYkeys city datasets 21
4.2.4 CITYkeys project datasets 28
4.3 Manual input 35
4.3.1 Manual input form for project assessment 35
4.3.2 Manual input form for city assessment 42

5. Methodology and algorithms for calculating CITYkeys KPIs 46
5.1 Main concept 46
5.2 Methodology 46
5.3 Algorithms 48
5.3.1 Algorithms for CITYkeys project KPIs 48
5.3.2 Algorithms for CITYkeys city KPIs 52

6. Example of the methodology application in a lighthouse project 57

7. Conclusions 60
7.1 Summary of achievements 60
7.2 Relation to continued developments and lessons learnt 60

8. Acronyms and terms 62

9. References 65

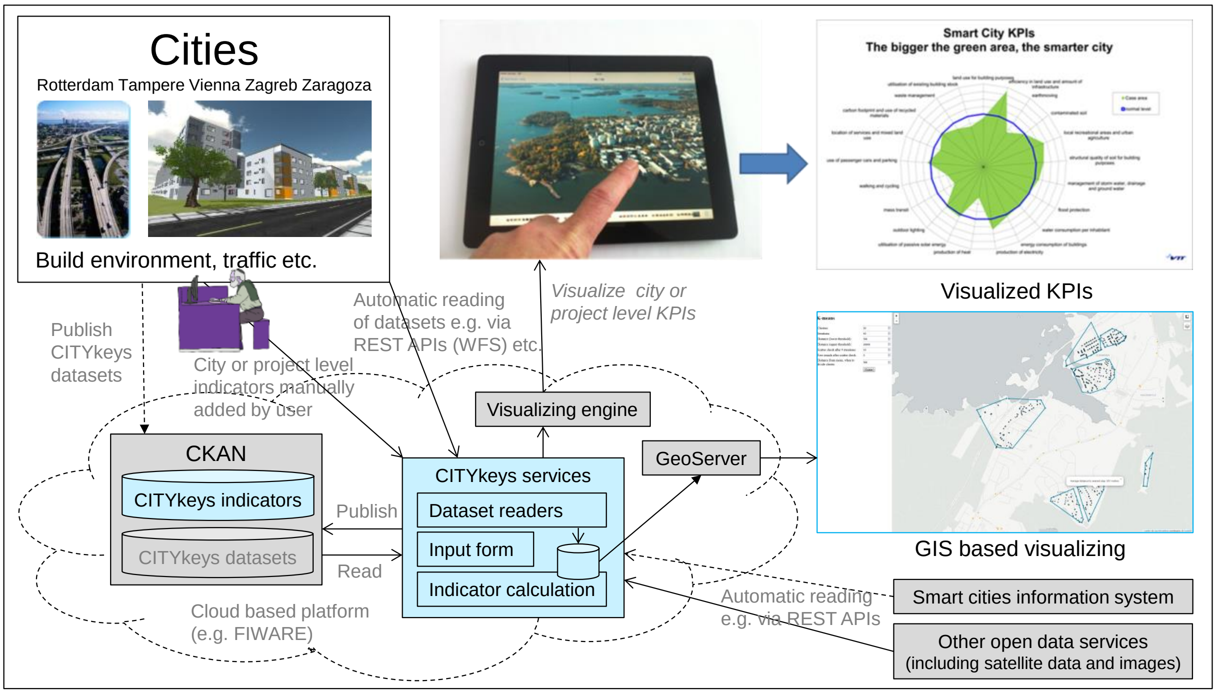
EXECUTIVE SUMMARY

This report specifies the data collection methods and calculation algorithms required by CITYkeys indicators. In addition, most relevant existing ICT systems and open platforms related to city data are presented and analysed.

The main results presented in this report can be summarised as follows:

- Specifications for the collection of the needed city and project level data utilising CITYkeys datasets and open data APIs. This includes technical specifications of CITYkeys datasets in the format required by the data collection system.
- Description of the methodologies to collect the needed city and project level indicator values manually via web page. This includes data input forms for general information on city and project assessment, assessment of quantitative and qualitative KPIs and a technical solution for implementing target values and indicator weighting coefficients.
- Definition of the calculation algorithms for the city and project level KPIs as defined in the CITYkeys framework[1]. This includes also a short description of the required main software components and instructions on how to use them (e.g. read datasets via CKAN REST APIs, calculate KPIs and store KPI values as open data).
- Analysis of the relevant existing city ICT systems and open platforms that are potential candidates to be used in CITYkeys prototype implementation (e.g. FIWARE, SCIS and CitySDK development kit as well as open city data publishing, sharing, finding, using and visualising platforms like CKAN and Socrata and the most common interactions / APIs that citizens have with their municipality like e.g. Open311, Linked Data and Tourism API).

Finally, an overview of the CITYkeys prototype platform is illustrated (see below).



[1] CITYkeys project deliverable 1.4 Smart city KPIs and related methodology – final. Available at http://citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D1-4-Smart-City-KPIs-WSWE-A7LN3E

1. INTRODUCTION

1.1 Purpose and target group

This document specifies both the methodology to collect the smart city data required by CITYkeys KPIs and the algorithms and related methodology to calculate those indicators. In addition this report describes the most important existing ICT systems and available open platforms which can be utilized when developing CITYkeys prototype platform for smart city and project assessment.

The main target group consists of technology providers, e.g. software developers, who are interested in developing CITYkeys related smart city solutions or utilising related open data. Furthermore the report targets, on general level, the city personnel who will input city or project level indicator values to CITYkeys system.

1.2 Contributions of partners

Partners' contributions are shown in the table below.

Section / Chapter	Contributor
1, 3, 4, 5.1, 5.2, 7	VTT Technical Research Centre of Finland (VTT)
2, 5.3, 6	AIT Austrian Institute of Technology GmbH (AIT)
6	City of Vienna (VIE)

1.3 Baseline

This report builds on results achieved in CITYkeys on indicator framework definition and related assessment methodologies (project deliverable 1.4 Smart city KPIs and related methodology – final[2]) and definition of needed data sets including data requirements and data sources (project deliverable 2.1 Definition of needed data sets[3]).

The main existing state of the art / available technologies and open data platforms for smart cities analysed in this task include:

- Open data publishing, sharing, finding, using and visualising platforms like CKAN and Socrata
- Platform ecosystems like FIWARE
- Service development kits for cities and developers like CitySDK and APIs like Open311 API, Linked Data API and Tourism API
- City specific APIs for city data

Related other (EU) projects:

- AIT (also partner in this task) is participating in the development of the Smart Cities Information System (SCIS), a project funded by EU under H2020 programme

---

[2] Available at http://citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D1-4-Smart-City-KPIs-WSWE-A7LN3E

[3] Available at http://www.citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D2-1-Definition-of-data-sets-WSWE-A7LN4E

- VTT has developed in the EU FP7 project EEPOS a unity game engine based 3D virtual model which has been utilised e.g. in visualising city and neighbourhood energy performance related KPIs[4]

1.4 Relations to other activities

The work described in this report is mainly based on the results of the CITYkeys Tasks 1.3 (Smart city KPIs and related methodology[5]) and 2.1 (definition of needed data sets[6]). The results presented in this report will be used as a starting point for Task 2.3 on the implementation of the CITYkeys prototype performance measurement system including user interface and visualisations. The prototype will be tested in Task 2.4.

[4] Available at http://eepos-project.eu/wp-content/uploads/2016/01/EEPOS_D53public.pdf

[5] Available at http://citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D1-4-Smart-City-KPIs-WSWE-A7LN3E

[6] Available at http://www.citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D2-1-Definition-of-data-sets-WSWE-A7LN4E

2. EXISTING PLATFORMS FOR SMART CITIES

The most important smart city related open data publishing and visualizing platforms are CKAN and Socrata and more advanced relevant platforms include FIWARE, SCIS and CitySDK. These platforms are described in more detailed in this chapter. The most important open data sources relevant for CITYkeys partner cities are listed in CITYkeys document "Definition of data sets" (D2.1[7]).

2.1 CKAN

This section gives an overview of the open source system CKAN (Comprehensive Knowledge Architecture Network) which has the purpose to provide publishing functionality for various data sources within the context of Open Data paradigm. It summarizes the usage possibilities and technical background with respect to possible applications for the CITYkeys project avoiding to reproduce the CKAN documentation.

General description and usage:

CKAN is a free open source (licensed under Affero GNU GPL v3.0) based data portal platform for publishing, sharing, finding, using and visualising data. CKAN is developed by the Open Knowledge Foundation (http://ckan.org/). Its main objective is to make open data websites, i.e. content management systems for publicly available data. Basically the platform can be used by governments, research institutions and other organizations collecting large amounts of data. The published data can afterwards be searched, accessed (viewed) and downloaded in different ways.

The basic entity (object) to be published within a CKAN instance is called a dataset. A dataset is piece of data provided in a certain format. A dataset consists of two additional information categories containing metadata and the data itself. The metadata describes the data by means of e.g. title, publisher, license, etc. therefore describes the content of the latter. CKAN is able to process various formats of data provided.

The system provides an authentication and authorization mechanism which is needed in case of data publishing. Already published data can be accessed for viewing and downloading purpose by anyone without the need of login and authorization. For publishing purpose all datasets have to be proprietary of a certain organization. Within an organization several layers of role models can exists. In the default case, any data which is provided within an organization is only available to the users of the same organization. The publication to all users of the system – making it open – has to be enabled by a separate particular action (press button). Each organization has an administrator who can modify the role permissions accordingly. CKAN can be setup also as a wiki-like data hub, i.e. data can be uploaded and published without belonging to any organizations meaning that in this specific case no authorization and authentication is needed.

For each dataset the following information categories have to be provided to describe the dataset:

- Description – long description describing the content and purpose in detail
- Tags – search tags helping finding and linking the data

[7] Available at http://www.citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D2-1-Definition-of-data-sets-WSWE-A7LN4E

- License – usage restrictions regarding the published data
- Organization – when publishing the data, it is the organization that the provided data belongs to in the default case

Afterwards the type of data resource with regard to physical accessibility has to be provided for the dataset:

- Link to a file – URL description e.g. http://mysite.com/myresource.csv
- Link to an API
- Upload a file – physical representation of the dataset, i.e. file will be uploaded to CKAN and linked
- Additional non mandatory information can be provided
    - Name
    - Description
    - File format – e.g. CSV, XML, JSON, PDF, etc.

For each dataset at least one resource has to be provided, but there can be more resources added according to the need to describe the dataset.

In the final step information about the visibility of the provided dataset has to be added:

- Visibility – public or private, the latter means that only members of the provider organization can see the data; public means that any user can see the data.
- Author and e-mail – name and e-mail of the provider (person or organization).
- Maintainer and e-mail – second responsible person/organization regarding the dataset.

After going through the above mentioned steps the dataset can be accessed according to the visibility settings.

Organizations and members can be added and managed as well within the system. The related procedure is straightforward.

Datasets can be searched by simple search field and appropriate filter options (e.g. organization) and wildcard patterns. If datasets are tagged by geographical area, CKAN provides an extension making it possible to search for datasets by selecting a particular area on a map.

For each found dataset, information about activity streams and related items are provided. The activity stream shows the history of changes done to the dataset, related items shows links to e.g. web pages related to the dataset or other interesting sources.

Each user can register for the dataset for the purpose of being notified in case of changes/updates done to the dataset.

For more detailed information regarding the usage of the system see the related documentation [1].

<u>Technology stack:</u>

CKAN is built with Python on the backend and Javascript on the frontend, and uses the Pylons web framework and SQLAlchemy as its ORM. Its database engine is PostgreSQL and its search is powered by SOLR. It has a modular architecture that allows extensions to be developed to provide additional features such as harvesting or data upload. CKAN uses its internal model to store metadata about the different records, and presents it on a web interface that allows users to browse and search this metadata. It also offers a powerful API that allows third-party applications and services to be built around it [2].

<u>Data preview and visualization</u>

Each CKAN resource page can contain one or more visualizations of the resource data file or content. These so called resource views can show data based on e.g. tables, bars, charts, map, etc. The type of view can be modified by the publishing organization. Individual views of data resources can be embedded on external sites and linked to CKAN. The possible views that might match with the data provided are also linked to the type of resource (File Store or DataStore see next).

File- and DataStore extension

These are the basic possibilities to add datasets and related resources to the CKAN system. The FileStore extension provides the possibility to physically upload the resource file to the underlying file system of CKAN. In extension to the basic functionality of FileStore which provides basically on the upload and download of file resources, the DataStore extension provides more sophisticated options with regard to querying – e.g. filtering and updating of data, without the need of download/uploading data files – and accessing the resources of datasets. Therefore users can also access parts of the data without the need to download the whole data. It allows the automatic data previews of the contents and allows several views of data which can be configured within the resource section of datasets. The DataStore extension provides and ad hoc database for storage of structured data from CKAN resources.

Form Integration

CKAN allows the integration of its modification and creation forms for datasets into an external front-end. It provides a simple way to redirect forms back to the external front-end application after submitting the data.

Linked Data and RDF

CKAN datasets can be serialized to RDF files with help of DCAT extension. It provides also support for importing RDF based datasets into CKAN.

Application Programming Interfaces (APIs)

According to the CKAN documentation this section intends to describe the basic APIs which might be used by external applications which want to interact with CKAN. The basic paradigm follows the remote procedure call (RPC) style that exposes all CKAN's core features to API clients. In fact any of the web application's functions can be also called by external applications through this API.

Basically the Legacy APIs allows accessing various functionalities regarding CKAN. The basic components can be classified by the CKAN APIs

- Model
    - Resources – entities describing a resource, e.g. Dataset, Group, License, etc.
    - Methods – GET, POST, PUT methods for read, write, update resource entities properties
    - Formats – retrieve data in JSON, JSON-format string to be send to via method POST or PUT
- Search
    - Resources – search is provided for DataSet, Resource, Revision and Tag Counts
    - Methods – GET and POST methods are supported
    - Formats – Search string is defined in JSON format (DataSet and Revision)
- Util – provides various utility functions e.g. auto-completion

Every outcome of API method invocation is signalled via HTTP status codes. For further details regarding the API see the CKAN documentation [1].

Conclusion

CKAN might be used for publishing KPI data from cities in various types and formats. According to privacy and legal restrictions regarding the datasets, it depends on the cities to publish the data to the public or only to their own and shared organization/representatives and related persons. Due to the fact, that CKAN is pure Open Source Software and underlies the Affero GNU GPL v3.0 license model, each city could setup their own CKAN based OGD platform for publishing purpose without the risk of additional investment cost. In the case of the CITYkeys partner cities Rotterdam, Vienna and Zagreb this is already done, since their open data portals use CKAN[8]. Tampere is also considering publishing open data with CKAN.

2.2 Socrata

Socrata is a software-as-a-service platform that provides a cloud-based solution for open data publishing and visualization. All Socrata datasets are API-enabled and Socrata Open Data API (SODA) developers worldwide can use SODA to create apps, analyses, and complex visualizations atop any Socrata dataset. The SODA server has been open sourced and could be self-provisioned as well. The New York City open data site is a great example of a Socrata site: NYC Open Data [3]. Socrata and CKAN datasets are interoperable via federation and both concepts share the goal of advancing the open data movement **Error! Reference source not found.**.

Socrata Open Data is developed and owned by the private company Socrata [4].

Therefore basically the functions and purpose of both platforms i.e. Socrata and CKAN can be described the same way. The main distinction lies as mentioned in the operation and the investment costs of the solution where the first one is a cloud-based solution where investment and running costs are to be expected and the latter can be installed and operated by each city on themselves without extra cost.

Due to commercial solution and the closed source paradigm of Socrata, no detailed documentation about architecture and usage can be found so far. As mentioned above, the Socrata Open Data API (SODA) enables developers and users to feed in and read data from Socrata Open Data and the creation of applications on top [5].

Basically the following operations are supported for developers by the API (list is not exhaustive):

- Analyzing and visualizing data – Heat maps, Charts, Google Maps
- Statistical operations – Python (Pandas), R
- Software Development Kit Support (SDK) – .Net, PHP, Python, R, Ruby, Scala, Swift, Java, etc.

For data publishers it is important to clarify in advance of using Socrata, how the datasets are provided and how they should be imported into the system.

If datasets are expected to be provided only on a very infrequent basis (snapshot data), it is recommended to publish the data via the web interface. This seems very straightforward and can be compared to the CKAN way of publishing data (forms, users, roles, etc.) [6].

[8] https://open.wien.gv.at/site/open-data/

For frequently provided data it is recommended to use the Socrata DataSync method. It comprises of a Java application which supports the configuration and setup of automatic processes [7]. The application can process CSV or TSV files as input on a local machine or network drive and can be embedded into own ETL processes and applications. It will guarantee that new available data will be published automatically to the Socrata Open Data system. To be allowed to publish new data, a Socrata account with publisher or owner role of the dataset and an app token is needed.

For users of the FME software – ETL tool with GIS capabilities [8] – it is possible to link FME directly to the Socrata system. The company Safe© [9] has developed a writer for their SW making it possible to link ETL jobs directly to Socrata.

In extension to the above mentioned SODA API for developers, publisher have also the possibility to link their systems and applications more tightly to Socrate by the RESTful publisher API supporting the following functionalities [10]:

- Add, update, delete records within a Socrata dataset
- Maintain dataset metadata and privacy settings
- Create and import Socrata datasets

Conclusion

Similarly to CKAN, Socrata could be used for publishing KPI data from cities in various types and formats. The capabilities of the SODA API are powerful although the usage of CKAN web interface seems to be easier. The most critical issue is the be fact, that Socrata is a cloud-based solution developed by a private company, not following the Open Source paradigm and related license issues which might lead to extra cost (investment and running, TCO).

2.3 SCIS

The Smart Cities Information System (SCIS) is an ongoing H2020 project that brings together project developers, cities, institutions, industry and experts from across Europe to exchange data, experience and know-how and to collaborate on the creation of smart cities and an energy-efficient urban environment. Launched with support from the European Commission, SCIS encompasses data collected from ongoing and future projects under the CONCERTO initiative and Smart Cities and Communities calls under Horizon 2020. With a focus on smart cities, energy efficiency, transport and mobility, and ICT, SCIS showcases solutions in the fields of sustainable building and district development, renewable energy sources for cities, energy efficiency and low-carbon technology applications.[12]

CONCERTO is a European Commission initiative within the European Research Framework Programme (FP6 and FP7). Responding to the facts that buildings account for 40 % of total energy consumption in the Union, for 33% of CO2 emissions and that 70% of the EU's energy consumption and a similar share of GHG emission take place in cities, with a significant untapped potential for cost-effective energy savings, it aims to demonstrate that the energy-optimisation of districts and communities as a whole is more cost-effective than optimising each building individually, if all relevant stakeholders work together and integrate different energy-technologies in a smart way.[13]

The Smart Cities Information System:

- Collects valuable data and expertise from smart cities demonstration projects and sites and channels them into a comprehensive database to promote replication of projects;

- Presents a thematic overview of projects with a focus on technologies and expertise in fields such as energy-efficient buildings, districts and cities, sustainable energy, geothermal communities, sustainable urban planning, low-carbon cities and zero-energy neighbourhoods;
- Offers an outline of renewable energy sources and low-carbon technologies and examples of their use;
- Establishes best practice by analysing and visualising project results, enabling project developers and cities to learn and replicate;
- Identifies barriers and points out lessons learnt, with the aim of finding better solutions for technology implementation and replication, and policy development;
- Provides recommendations to policy makers on support and policy actions needed to address market gaps.

[12]

The idea in data collection is to read data from exiting non-proprietary open data APIs. The structure will be based on a public Application Program Interface (API) so that the external service will ensure a continuity of services regardless of internal structural changes or updates. This open API will be available to anyone requesting access and willing to consume the data for their own purposes, once approved by the appropriate services. The main purpose for this choice is to validate the data structure with the potential users as early as possible in development and avoid possible development deadlocks or non-required functionalities.

The data structure for the API will be based around the following entities:

- buildings;
- ESUs (Energy Supply Units);
- cities;
- countries;
- projects / communities

These entities will be each stored as proper documents in the CouchDB database. It will be possible to retrieve the full data of a single entity by using the required URL: entity_type/entity_id. Each entity will be made available as a JSON document. Each indicator will be calculated based on the relationship of the entity with the related entities (for example some indicators will be calculated based on the relationship of a building and its related city or country). Each indicator will be represented by a CouchDB View. Views are the primary tool used for querying and reporting on CouchDB databases. They are defined in JavaScript.

2.4 FIWARE

FIWARE is a middleware platform, driven by the European Commission, for the development and global deployment of applications for Future Internet. The API specification of FIWARE is open and royalty-free, where the involvement of users and developers is critical for this platform to become a standard and reusable solution. The objective of FIWARE is to facilitate a cost-effective creation and delivery of Future Internet applications and services in a variety of areas, including smart cities, sustainable transport, logistics, renewable energy, and environmental sustainability [14].

The FIWARE platform provides a rather simple but yet powerful set of APIs (Application Programming Interfaces) that ease the development of Smart Applications in multiple vertical sectors. The specifications of these APIs are public and royalty-free. Besides, an open source reference implementation of each of the FIWARE components is publicly available so that multiple FIWARE providers can emerge faster in the market with a low-cost proposition [15].

The FIWARE catalogue of libraries can be categorized by so called generic and domain specific enablers. These libraries contain reference implementations that allow developers to develop their own application based on functionalities like e.g. IoT applications, Big Data analysis apps and many more.

The generic catalogue can be categorized as follows:

- Data/Context
- Internet of Things (IoT) Service Enablement
- Advanced Web-based User Interface
- Security
- Interface to Networks and Devices (I2ND)
- Architecture of Applications/Services Ecosystem and Delivery Framework
- Cloud Hosting

The Data/Context catalogue contains services and functionalities regarding real time communication channels (messaging), stream processing (Kurento, multimedia capabilities), publish/subscribe information broker mechanisms, big data analysis capabilities (Cosmos) complex event processing and CKAN support.

The IoT Service Enablement catalogue implements a protocol adapter service enabling communication with devices based on Constrained Application Protocol (CoAP), a component for backend device management (IDAS), acting as interface between IoT gateways/devices and the data broker service capable of communicating based on several protocols, a IoT discovery service, the above mentioned IoT broker service and a real-time capable IoT data edge consolidation service. Basically the full application stack needs for connecting, discovering, managing IoT devices and storing their data are covered.

The Web-based User Interface component covers functionalities regarding maintenance of data in a generic, flexible and modular data mode, support for 2D/3D UI development, interface design tools and support for GIS based solutions. Moreover capabilities regarding virtual character definitions – creating, displaying and animating virtual characters – and synchronization mechanisms related to network-synchronized dynamic scene data models are supported.

The security catalogue of components implements mechanisms evaluating cyber security risks, implementations covering the aspects of trustworthy applications, support for backend authentication and authorization (PEP Proxy – Wilma), capabilities enabling identity management and authorization (user accounts and access to services and applications) as well as implementations regarding security monitoring (threat awareness).

The I2ND catalogue consists of services realizing a middleware for realizing communication and data exchange between different applications based on RPC and IDL specification mechanisms (Kiara Advanced Middleware). The Network Information and Control (OFNIC) component enables the abstraction and virtualization of network resources and functionalities. Its features are e.g. getting information about the network topology components and elements either real or virtual, therefore it enables the capabilities of smart routing if e.g. huge amount of data has to be transferred between sites of large network distance, finding the optimum route.

The Applications/Services and Data Delivery catalogue offers analytics capabilities based on a SpagoBI application stack (Open Source based Business Intelligence application framework). Revenues gained by implemented services can be distributed among the stakeholders – service providers and consumers – according to a flexible defined model – e.g. usage records – which can be implemented within this model. The Store – WStore component enables the selling of services to both consumers and developers of future internet applications. An end-user centred web application mashup platform aimes at allowing end users without programming skills to easily create web applications and dashboard/cockbids (application designer). A marketplace is provided bringing together offering and demand for making business. It allows registering business entities, publishing and retrieving and viewing offerings and demands. Finally a repository covering the aspects of service registration and associated media files for applications is offered.

Finally the Cloud Hosting catalogue offers services and components for virtualization and application management, e.g. Linux container services, HW deployment management capabilities (virtual servers and networks), basic management of cloud resources, support for users of cloud infrastructures and platforms to manage their services, monitoring capabilities reporting constantly the performance of the systems, automated deployment of applications and robust, scalable object storage functionalities based on OpenStack software stack.

Further details about this catalogue services can be looked up at [16].

The domain specific enabler catalogue covers the topics of

- Manufacturing
- Media
- eHealth
- Energy
- Agrifood

These enablers are different from the generic enabler catalogues described above. They are developed from 3rd party institutions not necessarily following the FIWARE developer guidelines but where made available to the FIWARE community within Future Internet PPP large scale trial project activities.

<u>License models:</u>

Figure 1. FIWARE General Enablers and associated license models [17]

The FIWARE platform ecosystem offers a huge variety of services and component with sound reference implementations regarding data management, applications, IoT ecosystems, security and operation of ICT infrastructures. It is possible to run own FIWARE instances. Regarding license options the situation might seem a little bit complicated, see *Figure* 1 for further details. It shows the above mentioned General Enabler catalogue and its associated license models. In general all catalogue items are published under Open Source initiative (OSI) approved Open Source licenses [18], but with different Open source licenses.

It turns out that the basic differences in the used license models lies in the often totally different philosophy regarding the freedom of commercialization. Some license are very demanding with regard to the copyleft paradigm, others are more vendor friendly and do not require to run the products under copyleft. Therefore if applications are developed under by means of FIWARE application stack (General Enabler catalogue items) much care must be put on the incorporated license models do not run into complicated legal frame conditions.

<u>Conclusion:</u>

The FIWARE application stack (middleware) offers a large variety of functionalities and contains a lot of reference implementations for further usage. For data management and analytics capabilities (CKAN integration) which might be interesting for CITYkeys tool implementation in the future, the capabilities of the framework should be further investigated. Especially in case of the license situation it is worth to follow further developments.

2.5 CitySDK

CitySDK is a service development kit for cities and developers, that tries to implement a generic application programming interfaces (APIs) across cities. The focus for the APIs lies on enabling rapid development of applications within the city context, which might be scaled and reused by citizens and developers. In the CitySDK project that started in January 2012, 8 cities across Europe have worked together to create some re-usable interfaces and processes.

During the development of the CitySDK the focus was set on participation, mobility and tourism as three of the most common interactions that citizens have with their municipality [19]. This resulted in three APIs:

- Open311 API [20]
- Linked Data API [21]
- Tourism API [22]

The Open311API interface offers services regarding the city's feedback systems. Citizens should have the possibility to report issues e.g. dumped waste, broken and damaged infrastructure items, to the system. Technologies which are used are based on XML and JSON formats, different media formats can be used when uploading media.

The Linked Data API offers unified and direct access to open transport, mobility and geo referenced data from government, commercial and crowd sources. Interfaces for other open source projects e.g. OpenTripPlanner, Analyst, CitySDK Open311 API and OpenStreetMap are currently implemented. The API is intended to make static and real-time data from several sources available. The format of the returned datasets can be JSON, JSON-LD, Geo-JSON and RDF/Turtle.

The Tourism API concentrates on creating location-based mobile services for tourists. It enables developers to access points of interest (POIs), routes and event information to people. Several cities in Europe have already implemented test instances based on that API (see link for further information).

<u>Conclusion</u>

The CitySDK can be compared to CKAN but with the restriction that it is much more narrowed in terms of domains, i.e. data, tourism and issue reporting. Nevertheless it should be taken into account if it comes to implementation of the CITYkeys platform in the future, especially with regards to linked-open-data issues (retrieving information from Open Data platforms).

3. CITYKEYS PLATFORM OVERVIEW

Based on the state-of-the-art analysis of the existing platforms for smart cities (chapter 2) the conclusions from the point of view of CITYkeys are as follows:

- CKAN is the world leading open source data publishing platform and can be installed also in cities' own servers. This makes it possible that cities can manage their own data (e.g. CITYkeys datasets and indicator values).
- CKAN has RESTful APIs. This makes it possible that external software (e.g. CITYkeys services) can publish indicator values to CKAN and read datasets from CKAN.
- Socrata is a software-as-a-service platform for open data publishing and visualization, developed by a private company, which might lead to extra cost (investment and running, TCO) and therefore not so interesting from CITYkeys point of view as CKAN. However Socrata has APIs which make it possible for CITYkeys to utilise related data if needed.
- FIWARE is a platform (driven by the European Commission) for the development and global deployment of applications for Future Internet. This platform supports CKAN and also GeoServer (an open source server for sharing geospatial data) and can be a potential future platform for running CITYkeys related applications.
- SCIS is an ongoing H2020 project that brings together project developers, cities, institutions, industry and experts from across Europe to exchange data, experience and know-how and to collaborate on the creation of smart cities and an energy-efficient urban environment. It does not have yet RESTful APIs with CITYkeys dataset (see chapters 4.2.2, 4.2.3 and 4.2.4) support. But it can be a potential data source also for CITYkeys if related APIs will be implemented.
- CitySDK service development kit for cities and developers is important, especially from citizen participation, mobility and tourism point of view as well as related Open311, Linked Data and Tourism APIs. These can be utilised when developing CITYkeys services.

Based on the previously presented analysis a conceptual overview of CITYkeys prototype platform's main components is shown in Figure 2.

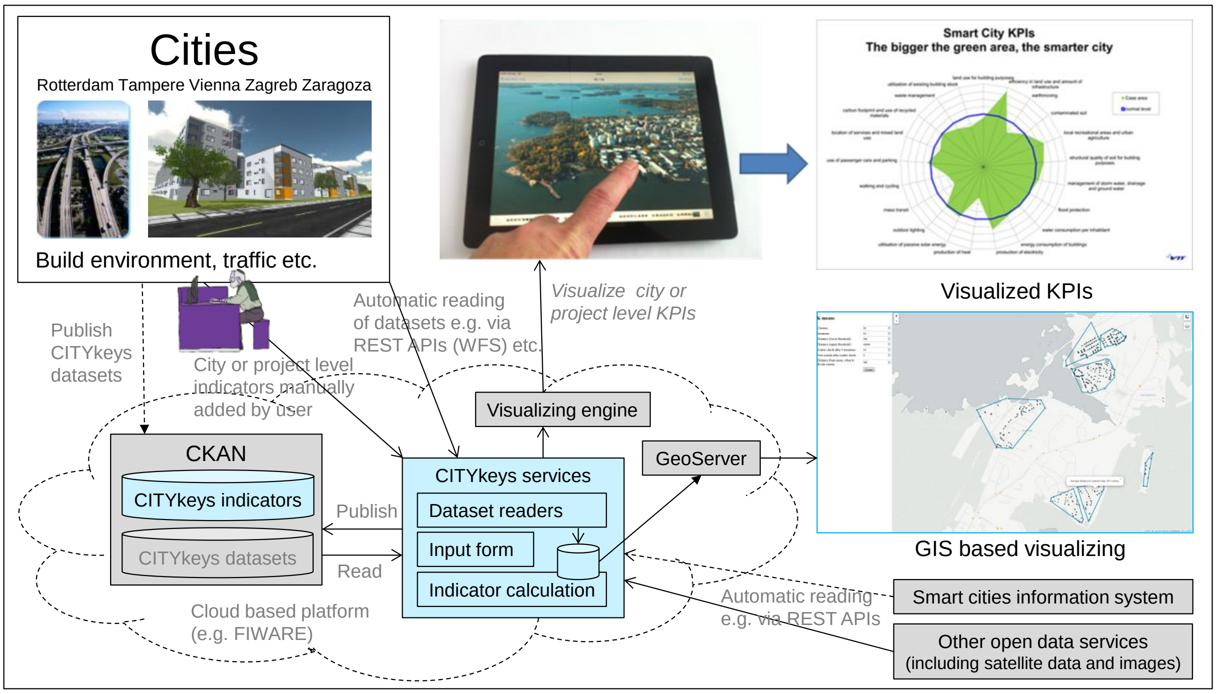


Figure 2. Overview of CITYkeys platform's main components

The first basic idea of the CITYkeys platform is to utilise existing open source software and open data APIs as much as possible. E.g. CKAN can be used for publishing and storing CITYkeys indicator values and optionally also CITYkeys datasets. In addition CKAN has APIs which CITYkeys services can utilise. This makes it possible that cities' indicator values will be stored to CITYkeys server or e.g. to cities' own CKAN.

The second basic idea of the CITYkeys platform is to utilise existing open data APIs as much as possible. This makes it possible to read datasets also from other city platforms if there are relevant APIs available. This issue is discussed in more detail in chapter 5.2.

The third basic idea of the CITYkeys platform is to use and develop software components which can be easily expanded and connected with other software and put the solution running in other platforms like FIWARE.

There is no current existing solution for CITYkeys city and project level indicator input or dataset reading and related indicator calculation. The target of the technical work package in the project (WP2) is to develop a practical approach that allows testing the feasibility of the KPIs framework implementation. Therefore this document presents a description of the optimal CITYkeys platform, possible to be implemented when all pre-conditions are in place, as well as, modules for system flexibility to adapt to different European cities' realities, including features for non-automatic dataset and KPIs value inputs. In the upcoming CITYkeys deliverable d2.3 a prototype will be presented with the objective of supporting the testing phase.

4. METHODOLOGY TO COLLECT INDICATOR DATA

4.1 Main concept

An overview of the methodologies to collect smart city and smart city project data is shown in Figure 3.

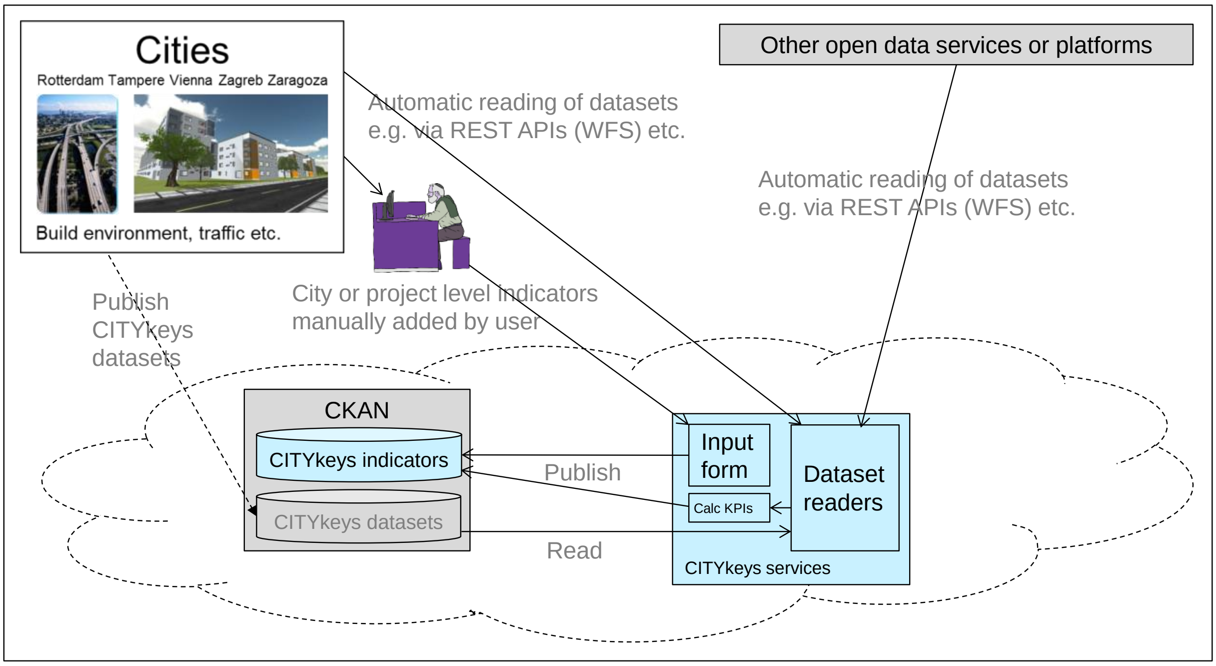


Figure 3. Overview of the methodologies to collect smart city data

The main idea in CITYkeys is to offer both automatic and manual data collection methods for city and project level indicator values.

For the manual mode, supported by web page interface (see chapter 4.3) the values are stored to existing open data portal (e.g. open source based CKAN). This makes it possible to utilise these indicator values through existing well known CKAN APIs by third party software developers as well as CITYkeys visualisation solutions (if authenticated).

On the automatic mode for CITYkeys data collection (see chapter 4.2) the datasets are read via existing APIs and related indicator values are calculated by CITYkeys algorithms (see chapter 5.2). Typically these data collection APIs are REST based web services (including WFS and WMS) which CITYkeys dataset readers could read automatically without user interaction. The challenge related to this option is the fact that, on average, only 15 % of datasets needed by CITYkeys city KPIs are available directly as open data in CITYkeys partner cities (i.e. Rotterdam, Tampere, Vienna, Zaragoza, Zagreb) as shown in the Citykeys report on datasets[9]. A big share of useful available datasets are in cities' internal databases and could be potentially read as well if good data management and collection practices are in place. They are often though scattered in different city departments, often not easily localisable and sometimes even not in easily machine readable formats. The aim of CITYkeys is to improve cities' related processes if possible, and at least to issue recommendations for improvements. With the current status in cities the option of creating

[9] Available at http://www.citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D2-1-Definition-of-data-sets-WSWE-A7LN4E

also other modules for non-automatic dataset input are, however, required. Cities can adapt the system with flexibility for different dataset realities. On the other hand this could be managed by publishing missing datasets as structured CITYkeys datasets (defined in chapters 4.2.2, 4.2.3 and 4.2.4) e.g. via CKAN or CitySDK APIs.

4.2 Dataset readers

Dataset readers are defined here as server side software which has as main function to collect automatically datasets needed for indicator calculation via available open data APIs (typically REST APIs). Dataset readers could collect structured CITYkeys datasets as well as potentially also other datasets as follows:

- Read CITYkeys datasets (see chapters 4.2.2, 4.2.3 and 4.2.4) published via CitySDK APIs or open data platform like CKAN
- Optionally read other structured datasets from available open (see Table 1) and non-open data sources for CITYkeys partner cities including CitySDK, CKAN and existing smart city platforms (e.g. smart city information system)
- Optionally read non-open data from cities' databases and other non-open data sources
- Optionally read other data like pictures from other open data sources (including e.g. open satellite related data and images services, e.g. Landsat 8)

Dataset readers related dataset queries will utilise Linked Data[10] technologies e.g. by interlinking the needed CITYkeys dataset sources (and optionally also other needed open and non-open dataset sources if needed ontologies and related dataset attribute level mappings between CITYkeys datasets and other datasets are known) e.g. via RESTful APIs.

4.2.1 Open data sources in CITYkeys partner cities

The available open data sources for CITYkeys partner cities were mapped in report D2.1 and are summarized in Table 1.

Table 1. Available open data sources for CITYkeys partner cities

Name of the source	**URL**	**Description**	**Provider**	**Relevant city**
Rotterdam opendatastore	http://rotterdamopendata.nl	Open data provided by city of Rotterdam	Rotterdam	Rotterdam
KNMI Data Centre	https://data.knmi.nl/portal/KNMI-DataCentre.html#openData=true	Meteorological open data	Royal Netherlands Meteorological Institute	Rotterdam
CBS Open data StatLine	http://opendata.cbs.nl/dataportaal/portal.html	General statistical datasets	Statistics Netherlands	Rotterdam
Netherlands	https://data.overheid.nl/data/dataset	Data from over	Dutch	Rotter

[10] More info available at http://linkeddata.org/

open government data portal		150 government organisations	Ministry of Interior and Kingdom Relations	dam
Open data van RVO.nl	http://www.rvo.nl/open-data-van-rvonl	Enterprise related datasets	Netherlands Enterprise Agency	Rotterdam
RDW Open data	https://opendata.rdw.nl/	Vehicle registration data	RDW	Rotterdam
Tampere open data	http://www.tampere.fi/tampereen-kaupunki/tietoa-tampereesta/avoin-data.html	List of open datasets provided by city of Tampere	Tampere	Tampere
Tampere GIS open data portal	http://kartat.tampere.fi	List of GIS based open datasets	Tampere	Tampere
Statistics Finland open data	https://www.stat.fi/org/avoindata/index.html	General statistical datasets	Statistics Finland	Tampere
SOTKAnet statistics and indicator database	https://www.sotkanet.fi/	Statistics and indicators about health and welfare	National Institute for Health and Welfare	Tampere
NLS open data	https://tiedostopalvelu.maanmittauslaitos.fi/tp/kartta/	GIS datasets	National Land Survey of Finland	Tampere
FMI open data	https://en.ilmatieteenlaitos.fi/open-data	Meteorological open data	Finnish Meteorological Institute	Tampere
Trafi open data	http://www.trafi.fi/en/services/open_data	Vehicle registration data	Finnish Transport Safety Agency	Tampere
Finnish open data portal	https://www.avoindata.fi/en	Finnish open data portal	Ministry of Finance	Tampere
Vienna open data	https://open.wien.gv.at/site/open-data/	Open data provided by city of Vienna	Vienna	Vienna
Statistics Austria open data	http://data.statistik.gv.at/web/	General statistical datasets	Staistik Austria	Vienna

Austrian open government data portal	https://www.data.gv.at/	Open government data	Cooperation OGD Österreich	Vienna
Zagreb open data	http://data.zagreb.hr/	Open data provided by city of Zagreb	Zagreb	Zagreb
Croatian open government data	http://data.gov.hr/	Open government data		Zagreb
Zaragoza open data	https://www.zaragoza.es/ciudad/risp/	Open data provided by city of Zaragoza	Zaragoza	Zaragoza
Spanish open government data portal	http://datos.gob.es/catalogo	Open government data		Zaragoza
AEMET open data	http://www.aemet.es/es/datos_abiertos/catalogo	Meteorological open data	AEMET	Zaragoza
European Data Portal	http://www.europeandataportal.eu/	Connects data portal across Europe		Europe
European Union Open Data Portal	https://open-data.europa.eu/en/data	Data produced by EU institutions and other bodies	European Union	Europe
EUROSTAT	http://ec.europa.eu/eurostat	European statistics	EUROSTAT	Europe

4.2.2 General CITYkeys datasets

A general overview of datasets required by CITY keys KPIs was given in the project report D2.1 Definition of needed datasets. Those datasets are, however, further specified in this report's sections 4.2.2-4.2.4 based on the needs of the CITY keys data collection system under development.

The general CITYkeys datasets include background information needed to calculate some CITYkeys indicator values. Some of this information is already published based on e.g. cities' own dataset definitions. Typically the existing datasets include much more information (e.g. datasets for bus stops have many attributes describing bus stop characteristics) and those datasets can be used instead of these CITYkeysGeneralDatasets if available.

For example the "CITYkeysGeneralDatasets_PopulationInBuildingsWithCoordinates" dataset is needed when calculating "CITYkeysCityPeopleHealth" dataset related attribute value "populationWithAccessToBasicHealthCareServicesWithin500m" or when calculating

"CITYkeysCityPeopleAccessToOtherServices" dataset related attribute value "numberOfInhabitantsWithAPublicAmenityWithin500m" described in chapter 4.2.3.

CITYkeys general datasets can be published e.g. in CSV, XML or JSON format based on the following definitions:

CITYkeysGeneralDatasets_PopulationInBuildingsWithCoordinates

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- buildingId (unit: -, data type: integer)
- buildingName (unit: -, data type: string)
- buildingAddress (unit: -, data type: string)
- buildingType (unit: -, data type: string);
- populationInApartmentBuilding (unit: -, data type: integer)
- buildingCoordinates (unit: -, data type: coordinates)

CITYkeysGeneralDatasets_BasicInfrastructureWithCoordinates

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- infraId (unit: -, data type: integer)
- infraName (unit: -, data type: string)
- infraAddress (unit: -, data type: string)
- infraType (unit: -, data type: string)
    - e.g. community centres, sports grounds, restrooms, drinking fountains
- infraCoordinates (unit: -, data type: coordinates)

CITYkeysGeneralDatasets_PublicTransportStopsWithCoordinates

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- publicTransportStopID (unit: -, data type: integer)
- publicTransportStopsName (unit: -, data type: string)
- publicTransportStopAddress (unit: -, data type: string)
- PublicTransportStopType (unit: -, data type: string);
- publicTransportStopCoordinates (unit: -, data type: coordinates)

CITYkeysGeneralDatasets_HourlyEnergyConsumption

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- energyConsumptionUnitId (unit: -, data type: integer)
- energyConsumptionUnitName (unit: -, data type: string)
- energyConsumptionUnitAddress (unit: -, data type: string)
- energyConsumptionUnitType (unit: -, data type: string)
- energyConsumptionUnitCoordinates (unit: -, data type: coordinates)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)

- hourlyEnergyConsumption_kWh (unit: kWh, data type: real)

CITYkeysGeneralDatasets_HourlyEnergyProduction

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- energyProductionUnitId (unit: -, data type: integer)
- energyProductionUnitName (unit: -, data type: string)
- energyProductionUnitAddress (unit: -, data type: string)
- energyProductionUnitType (unit: -, data type: string)
- energyProductionUnitCoordinates (unit: -, data type: coordinates)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- hourlyEnergyProduction_kWh (unit: kWh, data type: real)

CITYkeysGeneralDatasets_HourlyEnergyStored

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- energyStorageId (unit: -, data type: integer)
- energyStorageName (unit: -, data type: string)
- energyStorageAddress (unit: -, data type: string)
- energyStorageType (unit: -, data type: string)
- energyStorageCoordinates (unit: -, data type: coordinates)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- hourlyEnergyStored_kWh (unit: kWh, data type: real)

4.2.3 CITYkeys city datasets

CITYkeys city dataset specifications are based on the data needed in the CITYkeys indicator calculation algorithms described in chapter 5.3. City dataset can be published e.g. in CSV, XML or JSON format based on the following definitions:

CITYkeysCityPeopleHealth

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- populationWithAccessToBasicHealthCareServicesWithin500m (unit: -, data type: integer)

CITYkeysCityPeopleSafety

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- totalNumberOfAllCrimesReported (unit: -, data type: integer)
- numberOfFatalitiesRelatedToTransportationOfAnyKind (unit: -, data type: real)

CITYkeysCityPeopleAccessToOtherServices

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)

- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- numberOfInhabitantsWithATransportationStopWithin500m (unit: -, data type: integer)
- totalNumberOfVehiclesAvailableForSharing (unit: -, data type: integer)
- totalKilometersOfBicyclePathsAndLanes_km (unit: km, data type: real)
- totalLengthOfStreetsExcludingMotorways_km (unit: km, data type: real)
- numberOfInhabitantsWithAPublicAmenityWithin500m (unit: -, data type: integer)
- numberOfInhabitantsWithAtLeastSixCommercialAmenitiesWithin500m (unit: -, data type: integer)
- numberOfFixedBroadbandSubscriptions (unit: -, data type: integer)
- sumOfWifiNodesCoverage_m2 (unit: m^2, data type: real)
- totalCityUrbanSurface_m2 (unit: m^2, data type: real)

CITYkeysCityPeopleEducation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- numberOfSchoolsWithEnvironmentalEducationPrograms (unit: -, data type: integer)
- totalNumberOfSchools (unit: -, data type: integer)
- numberOfPeopleReached (unit: -, data type: integer)
- numberOfPeopleInTargetGroup (unit: -, data type: integer)

CITYkeysCityPeopleQualityOfHousingAndTheBuiltEnvironment

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- groundFloorSpaceUsedCommerciallyOrPublically_m2 (unit: m^2, data type: real)
- totalGroundFloorSpace_m2 (unit: m^2, data type: real)
- outdoorPublicRecreationSpace_m2 (unit: m^2, data type: real)
- totalPopulation (unit: -, data type: integer)
- totalGreenAreaInTheCity_hectare (unit: 10000 m^2, data type: real)

CITYkeysCityPeopleQualityOfHousingAndTheBuiltEnvironment_DwellingUnits

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- theTotalNumberOfDwellingUnitsInAllCategories (unit: -, data type: integer)
- totalNumberOfDetachedResidentialLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDetachedResidentialSmallNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDuplexOrTownhouseLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDuplexOrTownhouseSmallNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithNoElevatorLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithNoElevatorMedium70ToNotOver116m2 (unit: -, data type: integer)

- totalNumberOfDwellingUnitInMultiunitBuildingWithNoElevatorSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator4StoriesOrFewerLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator4StoriesOrFewerMedium70ToNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator4StoriesOrFewerSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator5to8StoriesLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator5to8StoriesMedium70ToNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator5to8StoriesSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator9StoriesOrMoreLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator9StoriesOrMoreMedium70ToNotOver7116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator9StoriesOrMoreSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfLiveWorkSpaceLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfLiveWorkSpaceSmallNotOver116m2 (unit: -, data type: integer)
- totalNumberOfAccessoryDwellingUnitLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfAccessoryDwellingUnitSmallNotOver116m2 (unit: -, data type: integer)

CITYkeysCityPlanetEnergyAndMitigation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- totalAnnualUseOfFinalEnergyWithinACity_MWh (unit: MWh, data type: real)
- totalAnnualConsumptionOfEnergyFromRenewableSources_MWh (unit: MWh, data type: real)
- totalAnnualEnergyConsumption_MWh (unit: MWh, data type: real)
- totalAmountOfDirectCO2EmissionsGeneratedOverACalendarYearByAllActivities_tons (unit: equivalent carbon dioxide units, data type: real)

CITYkeysCityPlanetOtherResources

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- annualTotalWeightOfDirectMaterialInput_tons (unit: 1000 kg, data type: real)
- annualTotalWeightOfMaterialExports_tons (unit: 1000 kg, data type: real)
- citysTotalWaterConsumption_litresPerDay (unit: Litres/day, data type: real)
- housesWithGreyAndRainWaterReuseCapability (unit: -, data type: integer)

- totalNumberOfHouses (unit: -, data type: integer)
- volumeOfWaterAbstractionInTheGeographicallyRelevantArea_m3 (unit: m^3, data type: real)
- volumeOfLongTermFreshwaterResourcesInTheGeographicallyRelevantArea_m3 (unit: m^3, data type: real)
- volumeOfWaterSupplied_m3 (unit: m^3, data type: real)
- volumeOfWaterBilled_m3 (unit: m^3, data type: real)
- volumeOfWaterSupplied_m3 (unit: m^3, data type: real)
- overallAreaOfTheCity_km2 (unit: km^2, data type: real)
- foodProducedIn100kmRadius_tons (unit: 1000 kg, data type: real)
- totalFoodDemandWithinCity_tons (unit: 1000 kg, data type: real)
- brownfieldAreaRedevelopedInTheLastYear_km2 (unit: km^2, data type: real)
- totalBrownfieldAreaInTheCity_km2 (unit: km^2, data type: real)

CITYkeysCityPlanetClimateResilience_TimeSeriesOfAirTemperature

- id (unit: -, data type: integer)
- measurementPointId (unit: -, data type: integer)
- measurementPointName (unit: -, data type: string)
- measurementPointCoordinates (unit: -, data type: coordinates)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- airTemperature_C (unit: °C, data type: real)

CITYkeysCityPlanetEnvironment

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- annualNO2Emissions_g (unit: g, data type: real)
- annualPM2.5Emissions_g (unit: g, data type: real)
- inhabitantsExposedToNoiseOver55dBaAtNightTime (unit: -, data type: integer)
- totalAmountOfTheCitysSolidWasteThatIsRecycled_tons (unit: 1000 kg, data type: real)
- totalAmountOfSolidWasteProducedInTheCity_tons (unit: 1000 kg, data type: real)
- annualAmountOfGenereratedMunicipalSolidWaste_tonsPerYear (unit: 1000 kg/year, data type: real)
- capita (unit: -, data type: integer)

CITYkeysCityPlanetEnvironment_TimeSeriesForAirQualityIndex

- id (unit: -, data type: integer)
- measurementPointId (unit: -, data type: integer)
- measurementPointName (unit: -, data type: string)
- measurementPointCoordinates (unit: -, data type: coordinates)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- NO2_µgPerm3 (unit: µg/m3, data type: real)
- PM10_µgPerm3 (unit: µg/m3, data type: real)
- PM10daily_µgPerm3 (unit: µg/m3, data type: real)

- Ozone_µgPerm3 (unit: µg/m3, data type: real)
- SO2_ µgPerm3 (unit: µg/m3, data type: real)
- Benzene_ µgPerm3 (unit: µg/m3, data type: real)

CITYkeysCityPlanetEcosystem

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- waterArea_km2 (unit: km2, data type: real)
- greenSpaceArea_km2 (unit: km2, data type: real)
- totalLandArea_km2 (unit: km2, data type: real)
- totalSpeciesWithinTheCity_vascularPlants (unit: -, data type: integer)
- newSpeciesWithinTheCity_vascularPlants (unit: -, data type: integer)
- locallyExtinctSpeciesWithinTheCity_vascularPlants (unit: -, data type: integer)
- totalSpeciesWithinTheCity_birds (unit: -, data type: integer)
- newSpeciesWithinTheCity_birds (unit: -, data type: integer)
- locallyExtinctSpeciesWithinTheCity_birds (unit: -, data type: integer)
- totalSpeciesWithinTheCity_butterflies (unit: -, data type: integer)
- newSpeciesWithinTheCity_butterflies (unit: -, data type: integer)
- locallyExtinctSpeciesWithinTheCity_butterflies (unit: -, data type: integer)
- additionalTaxonomicGroup1_name (unit: -, data type: string)
- totalSpeciesWithinAdditionalTaxonomicGroup1_value (unit: -, data type: integer)
- newSpeciesWithinAdditionalTaxonomicGroup1_value (unit: -, data type: integer)
- locallyExtinctSpeciesWithinAdditionalTaxonomicGroup1_value (unit: -, data type: integer)
- additionalTaxonomicGroups2_name (unit: -, data type: string)
- totalSpeciesWithinAdditionalTaxonomicGroups2_value (unit: -, data type: integer)
- newSpeciesWithinAdditionalTaxonomicGroups2_value (unit: -, data type: integer)
- locallyExtinctSpeciesWithinAdditionalTaxonomicGroups2_value (unit: -, data type: integer)

CITYkeysCityProsperityEmployment

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- workingAgeResidentsNotInPaidEmploymentOrSelfEmploymentButAvailableForWorkAndSeekingWork (unit: -, data type: integer)
- totalLabourForce (unit: -, data type: integer)
- totalNumberOfEnemployedYouth (unit: -, data type: integer)
- youthLabourForce (unit: -, data type: integer)

CITYkeysCityProsperityEquity

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)

- numberOfPeopleLivingInAffordableHousing (unit: -, data type: integer)

CITYkeysCityProsperityEquity_ListOfHouseholds

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- household_id (unit: -, data type: integer)
- modelledFuelConsumption_kWh (unit: kWh, data type: real)
- price (unit: €/kWh, data type: real)
- income_Eur (unit: €, data type: real)

CITYkeysCityProsperityGreenEconomy

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- numberOfCompaniesWithISO140001Certificate (unit: -, data type: integer)
- totalNumberOfCompaniesInTheCity (unit: -, data type: integer)
- millionEurAnnualProcurementUsingEnvironmentalCriteria_MEur (unit: M€, data type: real)
- millionEurTotalAnnualProcurementOfTheCityAdministration (unit: M€, data type: real)
- numberOfGreenJobs (unit: -, data type: integer)
- totalNumberOfJobs (unit: -, data type: integer)

CITYkeysCityProsperityEconomicPerformance

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- grossDomesticProductOnTheLevelOfTheCityPerCapita_Eur (unit: €/capita, data type: real)
- numberOfNewCompaniesRegistered (unit: -, data type: integer)
- totalPopulation (unit: -, data type: integer)
- medianDisposableAnnualHouseholdIncome_Eur (unit: €/household, data type: real)

CITYkeysCityProsperityInnovation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- peopleWorkingInCreativeIndustries (unit: -, data type: integer)
- totalWorkforce (unit: -, data type: integer)
- innovationHubsInTheCity (unit: -, data type: integer)
- totalPopulation (unit: -, data type: integer)
- totalExpenditureOnR&D (unit: -, data type: integer)
- cityGDP (unit: -, data type: integer)
- numberOfOpenGovernmentDatasets (unit: -, data type: integer)

CITYkeysCityProsperityInnovation_listOfDatasetStars

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- dataset_id (unit: -, data type: integer)
- datasetStars (unit: -, data type: integer)

CITYkeysCityProsperityAttractivenessAndCompetitiveness

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- travelTimesInPeakHours_h (unit: h, data type: integer)
- travelTimesDuringNonCongestedPeriods_h (unit: h, data type: integer)
- tripsMadeAnnuallyInTheCityWithPublicTransport (unit: -, data type: integer)
- totalPopulation (unit: -, data type: integer)
- moveIns (unit: -, data type: integer)
- moveOuts (unit: -, data type: integer)
- populationUnder14 (unit: -, data type: integer)
- populationOver65 (unit: -, data type: integer)
- populationFrom15to64 (unit: -, data type: integer)
- internationalEvents (unit: -, data type: integer)
- touristNightsPerYear (unit: -, data type: integer)

CITYkeysCityGovernanceCoCreation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- participationProcessesPerYear (unit: -, data type: integer)
- totalProjectsExecuted (unit: -, data type: integer)

CITYkeysCityGovernanceCoCreation_listOfCitizensParticipationLevelInProjects

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- project_id (unit: -, data type: integer)
- citizensParticipationLevel (unit: -, data type: integer between 1-5)

CITYkeysCityCommunityEngagement

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- timeStamp_year (unit: year, data type: integer)
- numberOfPeopleWhoVotedInLastMunicipalElections (unit: -, data type: integer)
- totalPopulationEligibleToVote (unit: -, data type: integer)

CITYkeysCityMultilevelGovernance

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)

- timeStamp_year (unit: year, data type: integer)
- totalPopulation (unit: -, data type: integer)
- totalAnnualExpendituresByTheMunicipalityForATransitionTowardsASmartCity (unit: -, data type: integer)

4.2.4 CITYkeys project datasets

CITYkeys project dataset specifications are based on the data needed in the CITYkeys indicator calculation algorithms described in chapter 5.3. Many of these calculation algorithms utilise data values before and after the project. In addition there is also need to calculate related indicator values during the project. This is solved in these CITYkeys dataset definitions by adding attributes "timePeriodStartTime_date", "timeStamp_date" and "timePeriodEndTime_date" so that "timeStamp_date" refers to the time during the project. E.g. if "timeStamp_date" is equal to "timePeriodStartTime_date" the values are before the project and if "timeStamp_date" is equal to "timePeriodEndTime_date" then the values are after the project. And if "timeStamp_date" is between "timePeriodStartTime_date" and "timePeriodEndTime_date" the dataset values are during the project.

Project dataset can be published e.g. in CSV, XML or JSON format based on the following definitions:

CITYkeysProjectPeopleHealth

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- waitingTimeBeforeProjec_h (unit: h, data type: real)
- waitingTime_h (unit: h, data type: real)

CITYkeysProjectPeopleSafety

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- transportationFatalities (unit: -, data type: integer)
- crimes (unit: -, data type: integer)

CITYkeysProjectPeopleAccessToOtherServices

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)

- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- kmsCyclingRoads_km (unit: km, data type: real)

CITYkeysProjectPeopleDiversityAndSocialCohesion

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- numberOfPeopleReached (unit: -, data type: integer)
- totalNumberOfPeopleConsideredAsTheTotalTargetGroupOfTheProject (unit: -, data type: integer)

CITYkeysProjectPeopleQualityOfHousingAndTheBuiltEnvironment

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- extraGroundFloorSpaceUsedCommercially_m2 (unit: m^2, data type: real)
- publicallyCreatedByTheProjectIn_m2 (unit: m^2, data type: real)
- currentTotalGroundFloorSpaceIn_m2 (unit: m^2, data type: real)
- urbanPublicOutdoorRecreationSpaceWithin500m_m2 (unit: m^2, data type: real)
- greenSpaceWithin500m_m2 (unit: m^2, data type: real)

CITYkeysProjectPeopleQualityOfHousingAndTheBuiltEnvironment_DwellingUnits

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- theTotalNumberOfDwellingUnitsInAllCategories (unit: -, data type: integer)
- totalNumberOfDetachedResidentialLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDetachedResidentialSmallNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDuplexOrTownhouseLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDuplexOrTownhouseSmallNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithNoElevatorLargeOver116m2 (unit: -, data type: integer)

- totalNumberOfDwellingUnitInMultiunitBuildingWithNoElevatorMedium70ToNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithNoElevatorSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator4StoriesOrFewerLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator4StoriesOrFewerMedium70ToNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator4StoriesOrFewerSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator5to8StoriesLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator5to8StoriesMedium70ToNotOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator5to8StoriesSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator9StoriesOrMoreLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator9StoriesOrMoreMedium70ToNotOver7116m2 (unit: -, data type: integer)
- totalNumberOfDwellingUnitInMultiunitBuildingWithElevator9StoriesOrMoreSmallNotOver70m2 (unit: -, data type: integer)
- totalNumberOfLiveWorkSpaceLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfLiveWorkSpaceSmallNotOver116m2 (unit: -, data type: integer)
- totalNumberOfAccessoryDwellingUnitLargeOver116m2 (unit: -, data type: integer)
- totalNumberOfAccessoryDwellingUnitSmallNotOver116m2 (unit: -, data type: integer)

CITYkeysProjectPlanetEnergyAndMitigation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- energyConsumption_kWh (unit: kWh, data type: real)
- lifeCycleEnergyUse_kWh (unit: kWh, data type: real)
- referenceScenario_kWh (unit: kWh, data type: real)
- energyProduction_kWh (unit: kWh, data type: real)
- CO2_Emissions_tons (unit: 1000 kg, data type: real)
- CO2_LifecycleEmissions_tons (unit: 1000 kg, data type: real)

CITYkeysProjectPlanetEnergyAndMitigation_HourlyMHD_Electricity

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)

- projectName (unit: -, data type: string)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- maximumHourlyStorageDishargeRate_Electricity_kW (unit: kW, data type: real)
- maximumHourlyEnergyLoad_Electricity_kW (unit: kW, data type: real)
- maximumHourlyEnergyGeneration_Electricity_kW (unit: kW, data type: real)

CITYkeysProjectPlanetEnergyAndMitigation_HourlyMHD_Heat

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- maximumHourlyStorageDishargeRate_Heat_kW (unit: kW, data type: real)
- maximumHourlyEnergyLoad_Heat_kW (unit: kW, data type: real)
- maximumHourlyEnergyGeneration_Heat_kW (unit: kW, data type: real)

CITYkeysProjectPlanetEnergyAndMitigation_HourlyMHD_Cooling

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timeStamp_h (unit: h, data type: date and time by EN ISO 8601)
- maximumHourlyStorageDishargeRate_Cooling_kW (unit: kW, data type: real)
- maximumHourlyEnergyLoad_Cooling_kW (unit: kW, data type: real)
- maximumHourlyEnergyGeneration_Cooling_kW (unit: kW, data type: real)

CITYkeysProjectPlanetMaterials

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- finalMaterialConsumption_tons (unit: 1000 kg, data type: real)
- baselineMaterialConsumption_tons (unit: 1000 kg, data type: real)
- recycledAndReusedMaterialsUsedByTheProject_tons (unit: 1000 kg, data type: real)
- totalMaterialConsumptionByTheProject_tons (unit: 1000 kg, data type: real)
- renewableMaterialsUsedByTheProject_tons (unit: 1000 kg, data type: real)
- materialsUsedByTheProjectThatCanBeRecycledAfterUsed_tons (unit: 1000 kg, data type: real)
- totalMaterialsUsedByTheProject_tons (unit: 1000 kg, data type: real)

CITYkeysProjectPlanetWater

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)

- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- decreaseInVolumeOfTheWaterUsedDueTheProject_m3 (unit: m^3, data type: real)
- volumeOfTotalWaterConsumptionOfTheCity_m3 (unit: m^3, data type: real)
- volumeOfRainAndGreyWaterReusedOnSite_m3 (unit: m^3, data type: real)
- totalVolumeOfWaterUsedOnSite_m3 (unit: m^3, data type: real)
- increasedVolumeOfTheWaterUsedFromLocalResources_m3 (unit: m^3, data type: real)
- volumeOfTotalWaterConsumptionOfTheCity_m3 (unit: m^3, data type: real)

CITYkeysProjectPlanetLand

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- projectArea_ha (unit: 10000 m^2, data type: real)
- inhabitants (unit: -, data type: integer)
- workPlaces (unit: -, data type: integer)
- extraFoodProducedIn100kmRadiusBecauseOfTheProject_tons (unit: 1000 kg, data type: real)
- totalFoodDemandWithinTheProjectBoundariesWithin100kmRadius_tons (unit: 1000 kg, data type: real)

CITYkeysProjectPlanetPollutionAndWaste

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- NO2_Emissions_tons (unit: 1000 kg, data type: real)
- PM2.5_Emissions_tons (unit: 1000 kg, data type: real)
- dB_Level (unit: dB, data type: real)
- solidWaste_tonsPerTimeperiod (unit: 1000 kg, data type: real)

CITYkeysProjectPlanetEcosystem

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)

- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- blueAndGreenSpace_m2 (unit: m^2, data type: real)

CITYkeysProjectProsperityEmployment

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- useOfLocalWorkforceInProject_Eur (unit: Eur, data type: real)
- totalUseOfWorkforceInProject_Eur (unit: Eur, data type: real)
- jobsCreatedByTheProject (unit: -, data type: integer)

CITYkeysProjectProsperityEquity

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- grossHouseholdIncome_EurPerYear (unit: Eur, data type: real)
- energyCosts_EurPerYear (unit: Eur, data type: real)
- fixedHousingCosts_EurPerYear (unit: Eur, data type: real)

CITYkeysProjectProsperityGreenEconomy

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- numberOfCompaniesWith_ISO140001_Certificate (unit: -, data type: integer)
- totalCompaniesInvolved (unit: -, data type: integer)
- annualCostsOfTheProjectSpentIn_CO2_EmissionReduction_Eur (unit: Eur, data type: real)
- annual_CO2_EmissionSavingsAchieved_Eur (unit: Eur, data type: real)

CITYkeysProjectProsperityEconomicPerformance

- id (unit: -, data type: integer)

- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- totalDirectAnnualCostsForEndUsers_Eur (unit: Eur, data type: real)
- amountToBeInvested_Eur (unit: Eur, data type: real)
- estimatedAnnualNetCashFlow_Eur (unit: Eur, data type: real)
- subsidiesReceived_Eur (unit: Eur, data type: real)
- totalInvestmentsOrCosts_Eur (unit: Eur, data type: real)
- initialInvestment_Eur (unit: Eur, data type: real)
- CashInflowInTime_Eur (unit: Eur, data type: real)
- CashOutflowInTime_Eur (unit: Eur, data type: real)
- discountRate (unit: -, data type: integer)
- referenceStudyPeriod_Years (unit: Year, data type: real)

CITYkeysProjectProsperityInnovation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- numberOfStartUpsResultingFromTheProject (unit: -, data type: integer)

CITYkeysProjectProsperityCompetitivenessAndAttractiveness

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)
- travelTimesInPeakHours_h (unit: h, data type: real)

CITYkeysProjectGovernanceCoCreation

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timePeriodStartTime_date (unit: -, data type: date by EN ISO 8601)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- timePeriodEndTime_date (unit: -, data type: date by EN ISO 8601)

- activelyEngagedUsersInTheProject (unit: -, data type: integer)
- totalPopulationOfTheCity (unit: -, data type: integer)

CITYkeysProjectPropagationAspectsOfSuccess

- id (unit: -, data type: integer)
- cityName (unit: -, data type: string)
- projectId (unit: -, data type: integer)
- projectName (unit: -, data type: string)
- timeStamp_date (unit: -, data type: date by EN ISO 8601)
- numberOfVisitorsToTheProjectSite (unit: -, data type: integer)

4.3 Manual input

Since currently only on average 15% of CITYkeys datasets are available via existing open data APIs in CITYkeys partner cities, manual input is selected as the main method to collect city and project indicator values in the project testing phase. The rest of the datasets available in the partner cities are mostly not collected and stored in automatically readable formats.

Manual input can be implemented in several different ways (e.g. by web page or uploading pre-completed indicator document like excel sheet to the open data publishing portal). In this chapter indicator collection via web page (input form) is specified.

CITYkeys input form has the function to read manually input, i.e. end user given, CITYkeys indicator values and save the values in selected open data platform (e.g. CKAN) as open or private data. Input form shall include the following features:

- Possibility to input all CITYkeys city indicators
- Possibility to input all CITYkeys project indicators
- All indicators do not need to be input at the same time and not all the indicators need to be assessed
- Both quantitative and qualitative indicators shall be supported
- Indicator shall include related time stamp
- Indicators inserted via web page shall be automatically saved in database (e.g. by publishing indicator values in open data platform via related API)
- Indicators can be searched according to sector of relevance

In addition, it is important to input general information on the assessed project or city evaluation (e.g. project boundaries) and specify the data sources used or the reference situation against which the project assessment is compared. CITYkeys framework will also offer the possibility for adding target values for numerical indicators or weights to compare the mutual importance of indicators. Those will be further discussed in tasks T2.3 and/or T3.3.

In summary, the manual indicator assessment web form is structured in three parts separately for project and city level assessments: 1) general information, 2) indicator value input, 3) target values and weights (potential content to be further discussed in T2.3/T3.3).

4.3.1 Manual input form for project assessment

This section presents through examples the three main parts of the manual input form for project assessment: 1) general information, 2) indicator value input, 3) target values and weights.

4.3.1.1 General information input form

Table 2 presents the structure of the first part of the project assessment web form "General information".

Table 2. General information for project assessment

City	
Country	
Project name	
Start and end date of the project	
Project size in Euros of investment	
Project phase (please select)	Planning / Implementation / Completed
Project's boundaries	
Please describe what is included/excluded in the project scope	
Area included in the project	
Who are the target users of the project results and how many are they?	
Project owner and other stakeholders involved in the project	
Project description	
Technologies or innovative methods used in the project that make it smart	
Expected main project outputs (e.g. new or improved services, products or knowledge)	
Relevant sectors (please select)	Natural environment / Built environment / Energy / Transport / ICT / Water and waste management / Health / Safety / Education / Innovation / Governance and community involvement / Economy
Description of the data collection process and methods	
Name of assessor(s) and position	
When was the assessment made?	
Data sources used	
Over which time period is the data?	
Comments	

4.3.1.2 Indicator value input form

There are three types of project KPIs in the CITYkeys framework that require slightly different assessment:

1) Quantitative KPIs which require an absolute numerical value as output. An example of such KPIs is "Net Present Value" which is calculated in [€].

2) Quantitative KPIs which require a relative numerical value as output. Those are typically calculated as % of change after the project compared to the situation before the project or a reference situation. An example of such KPIs is "Reduction in annual final energy consumption" which is calculated with the following formula: (Energy consumption after the project) / (Energy consumption before the project)]*100% - 100%. The value of this type of KPI is simply indicator in [%] of change due to the project. However, for the sake of transparency of the evaluation, the reference value against which the assessment is made, i.e. in this example "energy consumption before the project", should be indicated as well by the user. Alternatively it can be also automatically derived from the results read by the platform if related time-stamped energy consumption values are available and the general information provides clearly the project and assessment stages. In addition from CITYkeys platform point of view all manual and automatically read values like "energy consumption after the project" (marked as "energyConsumption_kWh" and time stamped with "timeStamp_date" in dataset "CITYkeysProjectPlanetEnergyAndMitigation", see chapter 4.2.4) are time-stamped values which makes it possible to study related KPIs also in different phase of the project (see also chapter 4.2.2).
3) Qualitative KPIs which require an integer between 1-5 as output. These indicators are evaluated through a qualitative assessment, by the user, based on a five level Likert scale described qualitatively. The system will display the description for the 5 values giving the user a "multiple-choice" kind of interface.

The indicator value input templates are slightly different in these three different cases and are presented though examples in Tables 3-5. These will be integrated in a web-based user-friendly interface for input.

Table 3. Indicator value input form for category 1 KPIs, example of KPI "Net Present Value"

Indicator name	Net Present Value
Relevant sectors	Economy
Status of the indicator assessment (select one option)	Assessed ☐ Will be assessed later ☐ Indicator not applicable ☐ Indicator not relevant ☐ Needed data not available ☐
Possible further explanation why the indicator isn't assessed	[*text to be added by the user*]
Definition	The Net Present Value (NPV) of the project calculated over the lifetime is a measure of financial project performance. If the NPV is positive, the benefits exceed the costs, and the project is worth pursuing.
Assessment method	Calculation with a formula using data on project's financial benefits and costs.
Calculation formula	$NPV = I_0 + \sum_{t=1}^{T} \frac{E_t - A_t}{(1+i)^t}$

		Input parameters: I_0 = Initial investment in t_0 [€] E_t = Cash inflow in t [€] A_t = Cash outflow in t [€] i = discount rate T = Reference study period [years]
Indicator value	*[input number by the user]*	EURO
Performance level (optional, requires definition of target values)		5. Excellent 4. Good 3. Average 2. Poor 1. Not acceptable
Link to complete indicator description		**Net Present Value** (full KPI description from CITYkeys D1.4 Appendix 1)
Comments Please explain the data sources used and add any information that is relevant for the interpretation of the assessment result		[*text to be added by the user*]

Table 4. Indicator value input form for category 2 KPIs, example of KPI "Increased efficiency of resource consumption"

Indicator name	Increased efficiency of resource consumption
Relevant sectors	Water and waste management
Status of the indicator assessment (select one option)	Assessed ☐ Will be assessed later ☐ Indicator not applicable ☐ Indicator not relevant ☐ Needed data not available ☐
Possible further explanation why the indicator isn't assessed	[*text to be added by the user*]
Definition	Reduction in material consumption in tonnes of the project compared to a baseline material consumption.
Assessment method	Calculation with a formula comparing project's material consumption to a baseline situation.

Calculation formula	[(final material consumption - baseline material consumption) / (baseline material consumption)] * 100%	
Indicator value	[*input number by the user*]	tonnes
Baseline material consumption used in the calculation	[*input number by the user*]	tonnes
Performance level (optional, requires definition of target values)	5. Excellent 4. Good 3. Average 2. Poor 1. Not acceptable	
Link to complete indicator description	**Increased efficiency of resource consumption** (full KPI description from CITYkeys D1.4 Appendix 1)	
Comments Please explain the data sources used and add any information that is relevant for the interpretation of the assessment result	[*text to be added by the user*]	

Table 5. Indicator value input form for category 3 KPIs, example of KPI "Improved cybersecurity"

Indicator name	Improved cybersecurity
Relevant sectors	ICT; Safety
Status of the indicator assessment (select one option)	Assessed ☐ Will be assessed later ☐ Indicator not applicable ☐ Indicator not relevant ☐ Needed data not available ☐
Possible further explanation why the indicator isn't assessed	[*text to be added by the user*]
Definition	The extent to which the project ensures cybersecurity
Assessment method	Qualitative evaluation on a five level Likert scale
Performance level (select)	5. Very high: A risk assessment on cybersecurity has been made for the project and there is a contingency plan for it. Risks on cyber security are low. The project uses only information systems with security

	assessment approvals (certified and accredited prior to deployment). 4. High: A risk assessment on cybersecurity has been made for the project and there is a contingency plan for it. Risks on cyber security are low. 3. Moderate: A risk assessment on cybersecurity has been made for the project and there is a contingency plan for it. 2. Low: A risk assessment on cybersecurity has been made for the project but there is either no contingency plan or high risks remain present. 1. Not at all: Cybersecurity hasn't received any attention in the project planning, even though the project involves the use of ICT.
Link to complete indicator description	**Improved cybersecurity** (full KPI description from CITYkeys D1.4 Appendix 1)
Comments Please explain the data sources used and add any information that is relevant for the interpretation of the assessment result	[*text to be added by the user*]

4.3.1.3 Target value and weight input form

The target values for scoring quantitative KPIs will be further discussed within tasks T2.3 and/or T3.3. The technical solution makes it possible to add them at whatever stage if decided to be used as a core component of the testing tool.

Target value input

For the numerical indicators in KPI categories 1-2 benchmarks can be added to set target levels and monitor their achievement on the following five level scale:

5. Excellent
4. Good
3. Average
2. Poor
1. Not acceptable

Having all the indicators in a uniform five level scale is useful because it makes the indicators comparable and thus gives the possibility for visualisations of the overall assessment results that are not possible otherwise.

Once that is done, the numerical outputs of these KPIs are automatically translated into an integer between 1 and 5.

Weight input

Setting weights for indicators makes it possible to indicate the mutual importance of different indicators by "putting more weight" on some indicators compared to others. It is useful for the interpretation of the overall result of an assessment that reflects the priorities of the user, especially when the overall result is illustrated in one figure. However, setting weights that would be used by all is often a source of controversy since the priorities on certain topics (indicators) differ between stakeholders and users. Therefore, the basic solution of the CITYkeys assessment tool is to use equal weights for all KPIs. The possibility to set other weights is however given for the user.

Two methods for setting weights are supported in CITYkeys:

1. Setting weights as percentages for all KPIs so that they sum to 100%.
2. Setting weights for the KPIs by using any numbers indicating the relative importance of the KPIs so that bigger number means more important and smaller number less important. For example the user can set number "1" weight for an indicator that has average importance, "3" weight for an indicator that is three times more important and "0,6" weight for an indicator that is 40% less important than the indicator that has a weight of one. At the end, these weights are automatically normalised to percentages that sum to 100%.

Another intuitive way to set weights is to set weights at different levels of the framework, i.e. themes, sub-themes and KPIs. Cities can use this method to define the weights to be inputted. Setting weights at different levels of the CITYkeys KPI framework would mean that weights are set at three levels:

- Themes (People, Planet, Prosperity, Governance, Propagation). A weight (number between 0-1) can be set for each theme so that their sum is 1. They indicate the mutual importance between the themes.
- Sub-themes. Under each theme weights (number between 0-1) can be set for the sub-themes so that their sum is 1. For example under the theme "People" there are the following sub-themes and their mutual importance can be indicated by weights that sum to 1: Health, Safety, Access to (other) services, Education, Diversity & social cohesion, Quality of housing and the built environment.
- KPIs. Under each sub-theme weights (number between 0-1) can be set for the KPIs so that their sum is 1. For example under the sub-theme "Safety" there are the following KPIs and their mutual importance can be indicated by weights that sum to 1: Traffic accidents, Crime rate, Cybersecurity, Data privacy.

The reason for not supporting the latter method for setting weights in the web input form is that in the CITYkeys framework there are significant differences in the numbers of KPIs under each sub-theme. When setting weights over the whole framework in such a case, there is the risk of a bias in the weights: KPIs that are under a sub-theme that consists of a small number of KPIs get too much importance and KPIs that are under sub-themes that consist of many indicators get too little weights.

In principle, CITYkeys framework is built so that the sub-themes are linked to cities' policy targets. Therefore, setting weights at sub-theme level is expected to be most suitable for policy-makers, while the themes are too general and KPIs too technical for them. T3.3 will further discuss the weighting issue.

4.3.2 Manual input form for city assessment

This section presents through examples the three main parts of the input form for city assessment: 1) general information, 2) indicator value input, 3) target values and weights (potential content to be further discussed in T2.3/T3.3).

4.3.2.1 General information input form

Table 6 presents the structure of the first part of the city assessment web form "General information".

Table 6. General information for city assessment

City	
Country	
City area in km^2	
Number of inhabitants	
Description of the data collection process and methods	
Name of assessor(s) and position	
Date of assessment	
Data sources used	
Time period of the data used in the indicator assessment	

4.3.2.2 Indicator value input form

There are two types of city KPIs in the CITYkeys framework that require slightly different assessment:

1) Quantitative KPIs which require an absolute numerical value as output. An example of such KPIs is "Open data" which is calculated as number of open datasets provided by the city per 100.000 inhabitants.
2) Qualitative KPIs which require an integer between 1 and 5 as output. These indicators are evaluated through a qualitative assessment based on a five level Likert scale described qualitatively.

The indicator value input templates are slightly different for quantitative and qualitative KPIs and are presented though examples in Tables 7-8.

Table 7. Indicator value input form for quantitative KPIs, example of KPI "Open data"

Indicator name	Open data
Relevant sectors	ICT; Governance and community involvement
Status of the indicator assessment (select one option)	Assessed ☐ Will be assessed later ☐ Indicator not applicable ☐ Indicator not relevant ☐ Needed data not available ☐

Possible further explanation why the indicator isn't assessed	[*text to be added by the user*]	
Definition	This indicator indicates the amount of open data provided by the city. A data set is defined as open, when it fulfils three requirements: 1) data is available and is in readable form; 2) data is published with a license which allows re-use and redistribution; 3) data is published with equal terms for every user (The Open Knowledge Foundation, 2015).	
Assessment method	Typically cities have a portal through which they provide all their open data sets available to their inhabitants and companies and the number of open data sets can be counted easily from there.	
Calculation formula	(number of open government datasets/total population) *100.000	
Indicator value	[*input number by the user*]	
Performance level (optional, requires definition of target values)	5. Excellent 4. Good 3. Average 2. Poor 1. Not acceptable	
Link to complete indicator description	**Open data** (full KPI description from CITYkeys D1.4 Appendix 2)	
Comments Please explain the data sources used and add any information that is relevant for the interpretation of the assessment result	[*text to be added by the user*]	

Table 8. Indicator value input form for qualitative KPIs, example of KPI "Data privacy"

Indicator name	Data privacy
Relevant sectors	ICT, Safety
Status of the indicator assessment (select one option)	Assessed ☐ Will be assessed later ☐ Indicator not applicable ☐ Indicator not relevant ☐ Needed data not available ☐
Possible further explanation why the indicator isn't assessed	[*text to be added by the user*]

Definition	The level of data protection by the city
Assessment method	Qualitative evaluation on a five level Likert scale
Performance level (select)	5. Relevant national and European regulations on data protection and privacy are followed and written agreements are made for use of citizens' private/personal data. All the collected personal/private data, especially sensitive personal data, is accessed only by agreed persons and is heavily protected from others (e.g. locked or database on internal server with firewalls and restricted access). 4. City follows all the relevant national and European regulations/laws related to data privacy and protection. If personal/private data is collected from citizens, proper authorisations with written agreements are made. 3. City follows relevant national regulations on protection of personal data and the EU Directive on the Protection of Personal Data (95/46/EG). 2. City follows national regulations/laws on protection of personal data. 1. City doesn't follow national regulations/laws on protection of personal data.
Link to complete indicator description	**Data privacy** (full KPI description from CITYkeys D1.4 Appendix 2)
Comments Please explain the data sources used and add any information that is relevant for the interpretation of the assessment result	[*text to be added by the user*]

4.3.2.3 Target value and weight input form

Similarly to the project level assessment, also at city level it is possible to set target values and weights for indicators. Target values can be set for quantitative KPIs on a five level performance scale:

5. Excellent
4. Good
3. Average
2. Poor
1. Not acceptable

Since all the qualitative KPIs are evaluated on such a case, this uniform assessment scale can enable the comparability of the KPIs and give possibility for visualisations of the overall assessment result that are otherwise not possible. This is especially useful for users that lack the indicator substance related competence to interpret the numerical output values of the quantitative KPIs. It also helps to easily monitor on which themes the city is performing particularly well or poorly.

Weights can be set for the KPIs to indicate their mutual importance. Please see 4.3.1.3 for more details.

5. Methodology and algorithms for calculating CITYkeys KPIs

5.1 Main concept

The main concept for calculating CITYkeys KPIs is shown in figure 4.

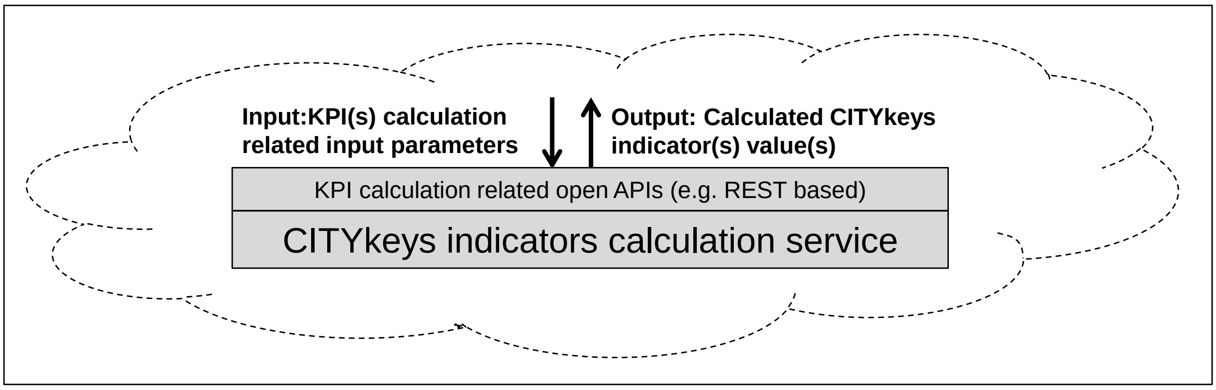


Figure 4. Main concept for calculating CITYkeys KPIs

The CITYkeys indicators calculation service is server side software which calculates CITYkeys indicator(s) value(s) based on the needed input parameters. The needed input parameters and APIs vary by KPIs (see chapter 5.3). The calculation service has open APIs (e.g. REST based) and it can be called by other applications.

5.2 Methodology

CITYkeys indicators calculation service can be used in many different ways. The different functionalities for calculating CITYkeys KPIs are shown in figure 5. It is to be noted however that as described in CITYkeys D2.1 currently especially for project KPIs the required data is usually not yet collected in databases or open data platforms to be read automatically. Even in CITYkeys testing phase it will often need to be input manually. However, the optimal CITYkeys platform components are described here as in future more automatic indicator data reading will be possible at least for a selection of CITYkeys KPIs. These different functionalities will be tested partly in different city pilots based on the priorities and data management realities in each partner city.

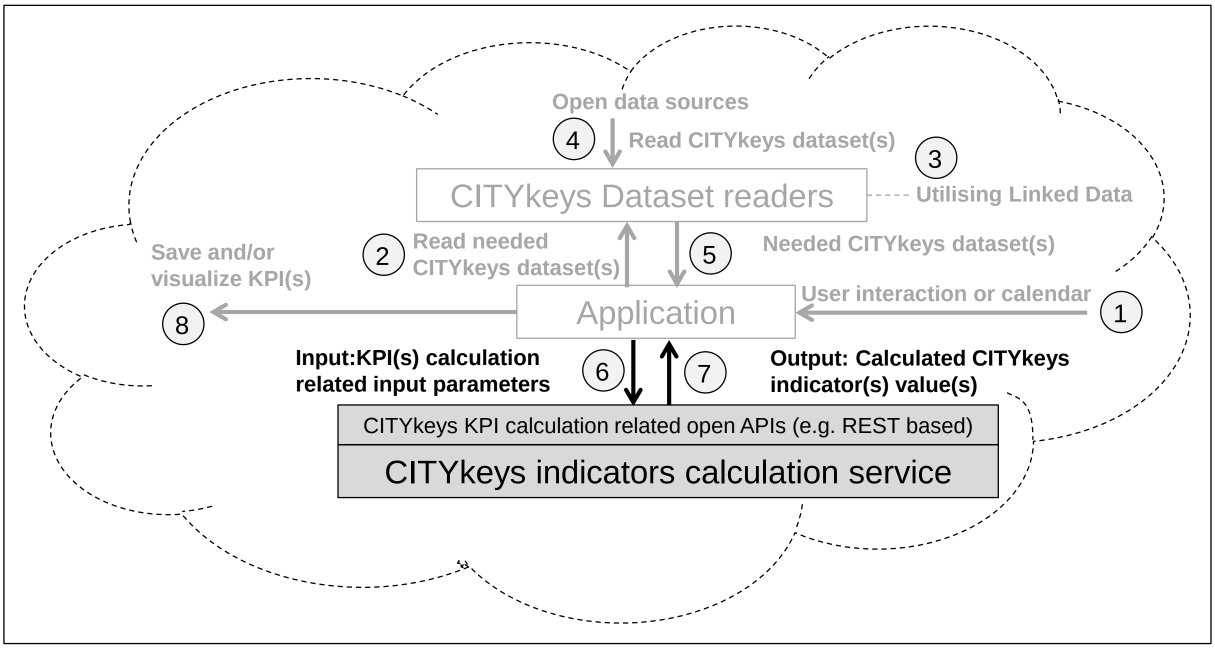


Figure 5. An example how to use the CITYkeys indicators calculating service

The different phases how CITYkeys indicators calculating service could be used are described in the following:

1) External application e.g. by end user interaction or calendar starts the CITYkeys calculation process
    - E.g. calculate KPI "Carbon dioxide emission reduction".
2) Related application calls CITYkeys dataset readers to read needed CITYkeys dataset(s)
    - E.g. read CITYkeys datasets (see chapter 4.2.3 and 4.2.4) which include variables "CO2 emissions after the project" and "CO2 emissions before the project" which are needed to calculate "Carbon dioxide emission reduction" related indicator value.
3) CITYkeys dataset readers checks the needed open data service address e.g. via Linked Data definitions.
4) CITYkeys dataset readers call the open data services and get the needed CITYkeys datasets (see CITYkeys datasets definitions in chapter 4.2.3 and 4.2.4).
5) CITYkeys application gets the needed datasets.
6) CITYkeys application calls the CITYkeys indicator calculation KPI specific service with needed parameters described in chapter 5.3.
    - E.g. read CITYkeys datasets which include variables "CO2 emissions after the project" and "CO2 emissions before the project" which are needed to calculate "Carbon dioxide emission reduction" related indicator value.
7) CITYkeys indicator calculation service calculates the requested CITYkeys indicator(s) values(s) using indicator related algorithms described in chapter 5.3 and returns the value(s) to the calling software.
    - E.g. "Carbon dioxide emission reduction" = [(CO2 emissions after the project - CO2 emissions before the project)/(CO2 emissions before the project)]*100

8) CITYkeys application can now e.g. save, visualize or publish the calculated CITYkeys indicator(s) value(s).

5.3 Algorithms

In this section the CITYkeys project and city indicator calculations and algorithms are described. Based on the data collection, harmonization and selection from WP1 and T2.1, all quantitative KPIs are listed and described by means of input parameters and their respective formulas (algorithms). The complete definitions of the indicators and their assessment methodologies can be found in the public project deliverable D1.4 Smart city KPIs and related methodology – final[11].

In addition, it is to be noted that the following variable notation used in the indicator calculation formulas has a space character between the words but in the implementation phase the CITYkeys dataset specifications (see chapters 4.2.3 and 4.2.4) related notation will be used without spaces and using capitals as the first letter of the words, except the first word (see chapters 4.2.3 and 4.2.4).

5.3.1 Algorithms for CITYkeys project KPIs

The indicators for theme People and related subthemes are described in Table 9. From CITYkeys platform point of view these indicator values are time-stamped which makes it possible to study related values also in different phase of the project.

Table 9. Project indicators of the people theme

Indicator Title	Indicator Unit	Formula	Remark
Reduction in waiting time	% in hours	(waiting time in hours after project/waiting time in hours before project)*100-100	
Reduction of traffic accidents	% of transportation fatalities	(transportation fatalities after project/transportation fatalities before project)*100-100	
Reduction in crime rate	% of crimes	(crimes after project/crimes before project)*100-100	
Extending the bike route network	% in km	(km’s cycling roads after the project/km’s cycling roads before the project)*100-100	
People reached	% of people	(number of people reached/total number of people considered as the total target group of the project)*100	
Increased use of ground floors	% in m2	(extra ground floor space used commercially/publically created by the project (in m2)/current total ground floor space (in m2))*100	
Increased access to urban public outdoor recreation space	m2	(urban public outdoor recreation space in m2 within 500m after the project) - (urban public outdoor recreation space in m2 within 500m before the	

[11] Available at http://citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D1-4-Smart-City-KPIs-WSWE-A7LN3E

		project)	
Increased access to green space	m2	(green space (m2) within 500m after the project) - (green space (m2) within 500m before the project)	

In the following Table 10**Error! Reference source not found.** the indicators for theme planet and related subthemes are described:

Table 10. Project indicators of the planet theme

Indicator Title	Indicator Unit	Formula	Remark
Reduction in annual final energy consumption	% in kWh	(Energy consumption after the project)/(Energy consumption before the project)]*100-100	
Reduction in lifecycle energy use	% in kWh	[(Life cycle energy use - Reference scenario)/(Reference scenario)]*100	
Increase in local renewable energy production	% in kWh	[(Energy production after the project - Energy production before the project)/(Energy production before the project)]*100	Energy production refers to local RES based energy production
Carbon dioxide emission reduction	% in tonnes	[(CO2 emissions after the project - CO2 emissions before the project)/(CO2 emissions before the project)]*100	
Reduction in lifecycle CO2 emissions	% in tonnes	[(CO2 lifecycle emissions after the project - CO2 lifecycle emissions before the project)/(CO2 lifecycle emissions before the project] *100	
Maximum Hourly Deficit	MHDx	$MHDx = Max\left[\int_0^{8760} \frac{[Lx(t) - Gx(t) + Sx(t)]dt}{\int_0^{8760} Lx(t)dt}\right]$	where Sx(t) is the storage discharge rate (negative value) and the functions L(t) and G(t) describe the hourly energy load and generation for the energy type x (electricity, heat or cooling)
Increased efficiency of resource consumption	% in tonnes	[(final material consumption - baseline material consumption)/(baseline material consumption)]*100	
Share of recycled input materials	% in tonnes	(recycled and reused materials used by the project (tons)/total material consumption by the project(tons))*100	
Share of renewable materials	% in tonnes	(renewable materials used by the project (tons)/total material consumption by the project(tons))*100	
Share of materials recyclable	% in tonnes	(materials used by the project that can be recycled after used(tons)/total materials used by the project(tons))*100	
Reduction in water consumption	% in m3	[(decrease in volume (m3) of the water used due the project)/(volume (m3) of total water consumption of the city)]*100	
Increase in water re-used	% in m3	[(volume (m3) of rain and grey water re-used on site after the project)/(total volume (m3) of water used on site)]/[(volume (m3) of rain and grey water re-used on site before the project)/(total volume (m3) of water used on site)]*100-100	

Self-sufficiency - Water	% in m3	[(Increased volume of the water used from local resources)/(volume of total water consumption of the city)]*100	
Increase in compactness	% of people or workplaces	{[(# of inhabitants or work places after project completion) - (# of inhabitants or work places before project completion)]/(# of inhabitants or work places before project completion)}*100	Note! Compactness shall be calculated as the increase in the number of inhabitants (#) or the number of work places (#) divided by the project area [ha]. The evaluator should indicate clearly which measure is used.
Self-suffiency - Food	% in tonnes	[(Extra food produced in 100 km radius because of the project (tons)/(Total food demand within the project boundaries (tons) within 100 km radius)]*100	
Decreased emissions of Nitrogen dioxides (NO2)	% in tonnes	[(NO2 emissions after the project)/(NO2 emissions before the project)]*100-100	
Decreased emissions of Particulate matter (PM2,5)	% in tonnes	[(PM2.5 emissions after the project)/(PM2.5 emissions before the project)]*100-100	
Reduced exposure to noise pollution	% in dB	[(dB level after the project project)/(dB level before the project)]*100-100	
Reduction in the amount of solid waste collected	% in tonnes	[(Solid waste (ton/time period) after the project)/(Solid waste (ton/time period) before the project)]*100-100	
Increase in green and blue space	% in m2	(blue and green space after project (m2)/blue and green space before project(m2))*100-100	

In the following Table 11 the indicators for theme prosperity and related subthemes are described:

Table 11. Project indicators of the prosperity theme

Indicator title	Indicator unit	Formula	Remark
Increased use of local workforce	% in euros	(Use of local workforce (project costs) in project/total use of workforce (project costs) in project)*100	
Local job creation	# (number)	# of jobs created by the project	
Fuel poverty	% point change in euros	((Energy costs before project)/(Gross household income)×100) - ((Energy costs after project)/(Gross household income)×100)	
Costs of housing	% in euros	(Fixed housing costs after the project (€/ year))/(Gross household income (€/year))×100	
Certified companies involved in the project	% of companies	(Number of companies with ISO 140001 certificate/total companies involved)*100	
CO_2 reduction cost efficiency	€/ton CO_2 saved/year	(annual costs (€) of the project spent in CO_2 emission reduction)(Annual CO_2 emission savings achieved)	

Financial benefit for the end-user	€/household/yr	total (direct) annual costs (€) for end-users (households) after the project - total (direct) annual costs (€) for end-users (households) before the project	
Net Present Value (NPV)	€	$$NPV = I_0 + \sum_{t=1}^{T} \frac{E_t - A_t}{(1+i)^t} = 0$$ Input parameters: I0 = Initial investment in t0 [€] Et = Cash inflow in t [€] At = Cash outflow in t [€] i = discount rate T = Reference study period [years] Nb: The number of years evaluated could be the mean life time of the project/measure or the time expected to return the inversion by the politic authorities.	
Internal rate of return (IRR)	%	$$NPV = I_0 + \sum_{t=1}^{T} \frac{E_t - A_t}{(1+i)^t} = 0$$ Calculated by solving for the discount rate for the formula NPV = 0. Input parameters: I0 = Initial investment in t0 [€] Et = Cash inflow in t [€] At = Cash outflow in t [€] i = discount rate T = Reference study period [years] Nb: The number of years evaluated could be the mean life time of the project/measure or the time expected to return the inversion by the politic authorities.	
Total cost vs. Subsidies	% in euros	(subsidies received/total investments or costs)*100	
Payback Period	Yrs	Amount to be Invested/Estimated Annual Net Cash Flow	
New start-ups	# (number)	The number of start-ups resulting from the project	
Decreased travel time	% in hours	[(travel times in peak hours after the project - travel times in peak hours before the project)/(travel times in peak hours before the project)]*100	

In the following Table 12 the indicators for theme governance and related subthemes are described:

Table 12. Project indicators of the governance theme

Indicator title	Indicator unit	Formula	Remark
Participatory governance	% of people	(actively engaged users in the project/total population of the city)*100	

5.3.2 Algorithms for CITYkeys city KPIs

In the following Table 13 indicators for the theme people and related subthemes are described:

Table 13. City indicators for theme people and subthemes

Indicator Title	Indicator unit	Formula	Remark
Access to basic health care services	% of people	[(population with access to basic health care services within 500m)/ (total population)]*100	
Traffic accidents	#/100.000	(number of fatalities related to transportation of any kind)/[(total population)/100000]	Number of accidents per 100.000 people
Crime rate	#/100.000	(total number of all crimes reported)/[(total population)/100000]	Number of crime incidents per 100.000 people
Access to public transport	% of people	(number of inhabitants with a transportation stop within 500m/total population)*100	
Access to vehicle sharing solutions for city travel	#/100.000	(total number of vehicles available for sharing)/ [(total population)/ 100000]	
Length of bike route network	% in km	(total kilometers of bicycle paths and lanes)/ (total length of streets excluding motorways)	
Access to public amenities	% of people	(number of inhabitants with a public amenity within 500m/total population)*100	
Access to commercial amenities	% of people	(number of inhabitants with at least six commercial amenities within 500m/total population)*100	
Access to high speed internet	# (number)	(number of fixed (wired)-broadband subscriptions) / [(total population)/100]	Per 100 inhabitants
Access to public free WiFi	% of m2	(sum of WiFi nodes coverage/total city urban surface)*100	
Environmental education	% of schools	(number of schools with environmental education programs/total number of schools)*100	
Digital literacy	% of people	(number of people reached/number of people in target group)*100	

Diversity of housing types	Simpson Diversity Index	1-SUM(n/N)^2, where n=total number of dwelling units in a single category and N=the total number of dwelling units in all categories. SUM=Sigma symbol.	See D1.4 p. 196 for the dwelling category definitions.
Ground floor usage	% of m2	(ground floor space used commercially or publically (in m2)/total ground floor space (in m2))*100	
Public outdoor recreation space	m2/cap	(outdoor public recreation space in m2) / total population	
Green space	hectares/100.000	(total green area (in hectares) in the city)/ [(total population)/ 100000]	

In the following Table 14 the indicators for the theme planet and related subthemes are described:

Table 14. City indicators for theme planet and subthemes

Indicator title	Indicator unit	Formula	Remark
Annual final energy consumption	MWh/cap/yr	(total annual use of final energy (MWh) within a city)/ (total population)	
Renewable energy generated within the city	% of MWh	(total annual consumption of energy from renewable sources)/(Total annual energy consumption)*100	
CO2 emissions	t CO2/capita/yr	[total amount of direct CO2 emissions in tonnes (equivalent carbon dioxide units) generated over a calendar year by all activities within the city, including indirect emissions outside city boundaries]/(total population)	
Domestic material consumption	t/cap/year	[annual total weight (in tons) of Direct Material Input (DMI) - annual total weight (in tons) of material exports]/(total population)	
Water consumption	litres/cap/year	(city's total water consumption in litres per day)/(total population)	
Grey and rain water use	% of houses	[(houses with grey and rain water reuse capability)/(total number of houses)]*100	
Water Exploitation Index	% of m3	(volume of water abstraction in the geographically relevant area/volume of long term freshwater resources in the geographically relevant area)*100	
Water losses	% of m3	{[(volume of water supplied)-(volume of water billed)]/(volume of water supplied)}*100	
Population	#/km2	(total population)/(overall area of the city (km2))	

density			
Local food production	% of tonnes	(food produced in 100 km radius (tons)/total food demand within city (tons))*100	
Brownfield use	% of km2	[brownfield area redeveloped in the last year (km2)]/[total brownfield area in the city (km2)]	
Urban Heat Island (Maximum difference temperature)	°C	Whether there is one or several measurement stations in the built environment, compare the air temperature measurements of these stations with a station outside the city which functions as a reference station, and look for the largest temperature difference (hourly average) during the summer months.	
Nitrogen dioxide emissions (NO2)	g/cap	[annual NO2 emissions (g)]/(total population)	
Fine particulate matter emissions (PM2.5)	g/cap	[annual PM2.5 emissions (g)]/(total population)	
Air quality index	-	Complicated. See definition in D1.4 page 221	
Noise pollution	% of people	{[(#of inhabitants exposed to noise > 55dB(A))]/(total population)}*100	
Recycling rate	% of tonnes	[(total amount of the city's solid waste that is recycled in tonnes)/(total amount of solid waste produced in the city in tonnes)]*100	
Municipal solid waste	t/cap/yr	(annual amount of genererated municipal solid waste t/yr)/(capita)	
Share of green and water spaces	% in km2	{[(water area) + (green space area)]/(Total land area)}*100	
Native species	% of species	The percentage change in number of native species shall be calculated as the total net change in species (numerator) divided by the total number of species from the 5 taxonomic groups from most recent survey (denominator). The result shall then be multiplied by 100 and expressed as a percentage (ISO/DS 37120, 2013). More information in CITYkeys D1.4.	

In the following Table 15 the city indicators for the theme prosperity and its subthemes are shown:

Table 15. City indicators for theme prosperity and subthemes

Indicator title	Indicator unit	Formula	Remark
Unemployment rate	% of people	(working-age residents not in paid employment or self-employment, but available for work, and seeking work)/(total labour force)*100	
Youth unemployment	% of people	(total number of unemployed youth)/(youth labour force)*100	

rate			
Fuel poverty	% of households	(modelled fuel costs)/income	NOTE! Calculated for each household separately. Modelled fuel costs = modelled consumption * price. Household is considered poor if the value is >0,1
Affordability of housing	% of people	(number of people living in affordable housing)/(total population)*100	
Share of certified companies	% of companies	(number of companies with ISO 140001 certificate/total number of companies in the city)*100	
Share of Green Public Procurement	% in M euros	(million EUR annual procurement using environmental criteria/million EUR total annual procurement of the city administration)*100	
Green jobs	% of jobs	(number of green jobs/total number of jobs)*100	
Gross Domestic Product	€/cap	(gross domestic product (GDP) on the level of the city)/number of inhabitants	
New business registered	#/100.000	(number of new companies registered)/[(total population)/100000]	
Median disposable income	€/household	median disposable annual household income	
Creative industry	% of people	(people working in creative industries/total workforce)*100	
Innovation hubs in the city	#/100.000	(innovation hubs in the city)/[(total population)/100000]	
Research intensity	% in euros	(total expenditure on R&D/city GDP)*100	
Open data	#/100.000	(number of open government datasets)/[(total population)/100000]	(number of open government datasets/total population)x100000
Congestion	% in hours	((travel times in peak hours - travel times during non-congested periods (free flow*))/travel times during non-congested periods)*100	
Public transport use	#/cap/year	#of trips made annually in the city with public transport/total population	
Net migration	#/1000	((move-ins – move-outs)/total population)*1000	
Population Dependency Ratio	#/100	100 x ((population (0-14) + population (65+)) / population (15-64)	
International Events Hold	#/100.000	(# of international events/total population)*100000	
Tourism intensity	nights/100.000	(# of tourist nights/total population)*100000	

The following Table 16 shows the indicators for theme governance and related subthemes:

Table 16. City indicators for theme governance and subthemes

Indicator title	Indicator unit	Formula	Remark
Open public participation	#/100.000	(# of participation processes/total population)*100000	
Voter participation	% of people	(number of people who voted in last municipal elections/total population eligible to vote)*100	
Expenditures by the municipality for a transition towards a Smart City	€/capita	(total annual expenditures by the municipality for a transition towards a Smart city/total population)	

The indicators listed in the previous tables are based on the CITYkeys selection of indicators and their descriptions given in the public project deliverable D1.4 Smart city KPIs and related methodology – final[12]. All quantitative indicators were listed in the tables. Qualitative indicators were excluded due to the fact, that they are not directly useable within formula based calculations, but rather displayed in a multiple choice user interface for user assessment on a uniform five level qualitative Likert scale.

[12] Available at http://citykeys-project.eu/citykeys/resources/general/download/CITYkeys-D1-4-Smart-City-KPIs-WSWE-A7LN3E

6. EXAMPLE OF THE METHODOLOGY APPLICATION IN A LIGHTHOUSE PROJECT

After implementing the first prototype of the CITYkeys platform, the indicator based assessment methodology will be tested in assessing different smart city projects and city performance. CITYkeys partner cities have presented their preliminary testing cases in the project report D2.1. Since each project has its own characteristics which involve different data collection issues, these testing plans are being further specified in parallel with the CITYkeys data collection system's technical specification and implementation work. This section shows one example how one partner city, city of Vienna, is planning its testing phase and which types of issues related to data collection are foreseen. The smart city project to be evaluated later is the EC funded SMARTER TOGETHER lighthouse project in which Vienna is partner and which has just started. [23] This section tries to combine the applicability of monitoring with a concrete example which is intended to illustrate the feasibility of the concept.

Vienna's Smart City Lighthouse area is part of Simmering, the 11th and one of the outer districts in the South-East of Vienna. Simmering is a traditional workers' district. The area selected for SMARTER TOGETHER project is located in its North-West. It is an area "in between" vast redevelopment sites (Vienna main station, Mautner-Markhof Areal), but not directly connected to them, and is as a whole a refurbishment area. Large social housing estates mainly built between 1945 and 1985 and owned by the City of Vienna – Wiener Wohnen or Non-Profit Housing Cooperatives (i.e. BWSG) need to be refurbished in the upcoming years. This refurbishment will have a big impact of local energy consumption unless measures at the supply side, especially at the district heating infrastructure, are taken. [24]

Information plays a fundamental role for increasing the efficiency of urban infrastructure and improves the way urban planning works in a more precise and predictive way. It is becoming increasingly clear, however, not least as a result of intensive data analysis, that significant efficiency gains can be achieved by tapping into the interdependencies between different segments. Within the project, a Smart Data Platform will be deployed based to enable and stimulate a proper understanding of how infrastructure is used in different domains, the interdependencies between different elements of infrastructure and the effects of external drivers as diverse as public policy, major events and weather patterns and precipitation. The measure will follow an open data and open platform approach to support the generation of an ecosystem of development partners. It will also strengthen network effects and encourages the participation and engagement of citizens. [24]

This Vienna lighthouse project aims to establish an integrated Smart Neighbourhood in a dedicated area, addressing energy, mobility, ICT and infrastructure in an integrated manner while emphasising citizen engagement and governance. [24]

The project will:

- Focus on refurbishment of social housing and a public secondary school with a gym (8.800m^2) at large scale (66.000 m2) reducing energy demand by >60 %
- Renovate the district heating system with a unique integration of local renewable energy sources and related new business models
- Develop Simmering NW into a flagship e-mobility area and reduce mobility-related energy consumption by novel business and implementation concepts as well as citizen engagement

- Involve all main actors of the area (housing operators, energy suppliers, educational institutions and businesses) working together towards integrated and sustainable solutions and at the same time foster the engagement of tenants and residents in the transformation and co-design processes
- Work out governance structures suitable for complex integrated renewal projects
- And as a result further reduce energy consumption and CO_2 emissions and improve the ecological/environmental footprint of the area.

The main focus in testing the CITYkeys assessment method will be on energy impact and the quality of life. For the energy impact this would mean on refurbishment of social housing the following measures are evaluated in CITYkeys by the exemplary project indicator titles "Reduction in annual final energy consumption by buildings" or "Reduction in lifecycle CO_2 emissions".

SMARTER TOGETHER focus also on the improvement of quality of life for its inhabitants through innovative smart services, new ways of transportation, user-driven participation strategies, creation of jobs on local level and other soft factors.

The quality of life could be evaluated in CITYkeys by the exemplary project indicator titles "improved access to public transport", "improved quality of public transport" or "Proximity to jobs"

Before and during the project, the corresponding figures (several times) are recorded and mapped to the planned prototype system CITYkeys.

It shall be noted that the above description should give an impression how the particular implementation and its performance with regard to the CITYkeys indicator system can be evaluated. The exemplary mentioned indicators are not exhaustive and the needed figures and indicators will be strongly dependent from the real implementation done within the SMARTER TOGETHER project efforts in the Vienna demo regions. As the project is at its starting phase currently, not all feasible indicators can be figured out this time, due to missing particular demo project implementations.

Nevertheless to make the general concept work, a method for data collection and processing has to be developed. As the SMARTER TOGETHER project has just started not all activities and particular project goals are readily defined, therefore it is not possible to describe the detailed data collection process in detail for the particular case.

Most of the projects deal with refurbishment of buildings, integration of renewable sources into existing energy systems and mobility issues as it is noted in the bullet points above. Therefore it has to be evaluated for what general objective the data to be processed has to be used:

- Operational goal – e.g. data is needed for operating the appliance, building, etc.
- Critical goal – e.g. data is needed for detection and monitoring of incidents, emergency management, etc.
- Analytical goal – e.g. data is needed to determine patterns, find relationships, generate new information, etc.
- Strategic goal – e.g. data is needed for long term decision making, city planning, etc.

According to these objectives it has to be further figured out, what systems and stakeholders as well as what kind of barriers regarding to data ownership and legal constraints and what barriers with regard to existing technologies and standards, do exist.

Nevertheless the planned or possible implementation approach and concept can be described as follows.

The methods for data collection are depending strongly on the existing systems which have to be incorporated according to the project goals. In general it is thinkable to collect the data automatically by the possibly already existing systems, e.g. Building Management Systems (BMS), HVAC systems and other systems consisting of relevant sensors where data can be retrieved from. If there is no (smart) system existing it could be either equipped with additional sensors or the needed data can be retrieved manually (manual reading from e.g. electricity meter).

The data processing is strongly linked to the data gathering. If there are existing systems a part or all of the data will be (locally) processed by these systems. Afterwards the pre-processed data should be transferred to a (city) wide data management platform. In case of Vienna the ongoing and meanwhile settled Open Government Initiative (OGD) platform could be used to collect all the (locally) gathered and pre-processed data in an own section of its platform (dedicated to demo projects). This concept would be comparable to the Community Management System (CMS) platform of Lyon.

On top of the centralized database solution (OGD, CKAN based) applications for analysing and decision management can be developed.

The most popular benefit for these platforms lies in the leverage for realizing high-level services for the purpose of the e.g. integrated assessment of static and dynamic (energetic) issues within a city/regional context. Apart from that, they can be used to integrate and enable the information flow from and to citizens (social participation).

7. CONCLUSIONS

The final goal of this task was the specification of data collection methods and calculation algorithms of CITYkeys KPIs. The related sub goals were the description and analysis of existing city data related ICT systems and open platforms to be potentially used in CITYkeys performance measurement system for testing.

7.1 Summary of achievements

This document presents the state-of-the-art on relevant city ICT systems and available open platforms that are potential candidates to be used in CITYkeys prototype implementation including

- Description of the existing city ICT systems and available open platforms like FIWARE, SCIS and CitySDK development kit
- Description of the existing open city data publishing, sharing, finding, using and visualising platforms like CKAN and Socrata
- Description of the most common interactions (APIs) that citizens have with their municipality like Open311, Linked Data and Tourism API

The main results presented in this report are the methodologies for city data collection and indicators calculation including

- Description of CITYkeys prototype platform based on the state-of-the-art analysis of the existing platforms for smart cities,
- Specification of the CITYkeys city and project datasets based on the data needed in the CITYkeys indicator calculation algorithms,
- Description of the ICT based methodologies to collect the needed city and project level datasets utilising open data APIs,
- Specification of the methodologies to collect the city and project level indicator values manually via web page including information input forms for general information for city and project assessment, assessment of quantitative KPIs and qualitative KPIs and technical solution for implementing target values and indicator weighting coefficients,
- Specification of the calculation algorithms for city and project level KPIs and related ICT methodologies (required main software components and instructions on how to use them) to read CITYkeys datasets (e.g. via CKAN REST API) and other datasets to calculate related KPIs and store related values as open KPI data,
- Plans on the application of the CITYkeys indicator assessment and data collection methodology in testing phase by one of the part cities (city of Vienna to evaluate the lighthouse project SMARTER TOGETHER).

7.2 Relation to continued developments and lessons learnt

This task presented the technical specifications for data collection and indicator calculation so that the KPI calculation methodologies defined in CITYkeys WP1 could be directly implemented for smart city and smart city project assessment in the implementation (T2.3) and testing (T2.4) phases. It is to be noted, however, that certain indicator assessment related content will be further discussed in CITYkeys T2.3 and/or T3.3, e.g. target values for quantitative KPIs for scoring them on a uniform scale as well as weights for indicating the mutual importance of KPIs in the framework. The technical specifications for implementing them directly to the performance measurement system are, however, already in place.

The challenge related with CITYkeys data collection is the fact that, on average, only 15 % of data needed by CITYkeys city datasets are available directly as open data in CITYkeys partner cities (i.e. Rotterdam, Tampere, Vienna, Zaragoza and Zagreb). Also a big share of the remaining available datasets is not yet accessible in easily machine readable formats. Especially the majority of the needed project data is not readily collected and typically requires analyses of project documentation or interviews with the project manager. In addition there are a lot of different dataset definitions done in different projects and a need for comprehensive global standards or at least for common dataset related ontology definitions.

The target of the technical work package in the project (WP2) is to develop a practical approach that allows testing the feasibility of the KPIs framework implementation. Therefore this document presents the description of the optimal CITYkeys platform, possible to be implemented when all pre-conditions are in place, as well as modules for system flexibility to adapt to different European cities' realities, including features for non-automatic dataset and KPI value inputs. In the upcoming CITYkeys deliverable d2.3 a prototype will be presented with the objective of supporting the testing phase. Finally these different functionalities will be tested in different city pilots to the extent possible based on the priorities and data management realities in each partner city.

As cities' have not yet made open and easily accessible much of their smart city data and projects' standard data collection practices are not yet in place, the objective of CITYkeys is to improve these underlying processes to the extent possible. After testing the practical feasibility of the CITYkeys assessment methodology in T2.4 "Testing", recommendations and guidelines for data sources, data collection, KPIs calculation and visual integration of the results will be issued. Those will be reported in later project reports (D2.4 Report on the case studies and D3.1 User handbook).

8. ACRONYMS AND TERMS

.Net	A software framework developed by Microsoft.
2D	Two Dimensional.
3D	Three Dimensional.
95/46/EG	EU Directive on the Protection of Personal Data.
Affero GNU GPL	Free, copyleft license for software.
Analyst	An extension to OpenTripPlanner (applying OpenTripPlanner's routing engine to problems in transportation planning, public policy, and the social sciences).
API	Application Programming Interface.
Backend	Any software performing either the final stage in a process, or a task not apparent to the user.
BSD	Berkeley Software Distribution (a Unix operating system).
CITYkeys	A project funded by the European Union HORIZON 2020 programme.
CitySDK	A service development kit for cities and developers.
CKAN	Comprehensive Knowledge Archive Network (Open source data portal software).
CO_2	Carbon Dioxide.
CoAP	Constrained Application Protocol.
CONCERTO	A European Commission initiative within the European Research Framework Programme (FP6 and FP7).
CouchDB	An open source database designed for modern web and mobile apps.
CSV	Comma-separated values (Text file).
DCAT	Data Catalog Vocabulary.
EEPOS	A project funded by the European Commission under the 7th Framework Programme.
ESU	Energy Supply Unit.
ETL	Extract, Transform and Load.
EU	European Union.
FIWARE	Future Internet Core Platform.
FME	Feature Manipulation Engine.
Frontend	An interface between the user and the backend.
Geo-JSON	An open standard format designed for representing simple geographical features (based on JavaScript Object Notation).
GeoServer	An open-source server for sharing geospatial data.
GET	A HTTP Method for RESTful Services.

GIS .... Geographic Information System.

GPL .... General Public License.

Horizon 2020 .... EU Framework Programme for Research and Innovation (2014-2020).

HTTP .... Hypertext Transfer Protocol.

HW .... Hardware.

I2ND .... Interface to Networks and Devices.

ICT .... Information and Communications Technologies.

IDAS .... An open source component for backend device management (under AGPLv3 license) integrated with the FIWARE ecosystem.

IDL .... Interface Definition Language.

IoT .... Internet of Things.

IRR .... Internal Rate of Return.

Java .... A general-purpose programming language.

JavaScript .... A high-level, untyped, and interpreted programming language of HTML and the Web.

JSON .... JavaScript Object Notation.

JSON-LD .... JavaScript Object Notation for Linked Data (a method of encoding Linked Data using JSON).

Kiara .... An open source Java based communication middleware (under LGPLv3 license) integrated with the FIWARE ecosystem.

KPI .... Key Performance Indicator.

Linked Data .... Method for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF.

Likert scale .... A psychometric scale commonly involved in research employing questionnaires.

Linux .... A free and open source software operating system for computers.

MHDx .... Maximum Hourly Deficit.

NPV .... Net Present Value.

OFNIC .... A distributed Software Defined Network controller for enterprises' OpenFlow-enabled network (open source under GPL v3.0 license) integrated with the FIWARE ecosystem.

Open311 .... An open standard protocol and related APIs providing open communication with public services and local government.

OpenStack .... A free and open-source software platform for cloud computing.

OpenStreetMap .... A project creating and distributing free geographic data of the world.

OpenTripPlanner .... An open source multi-modal trip planner.

ORM .... Object Relational Mapper.

PDF .... Portable Document Format.

PEP Proxy .... Performance-Enhancing Proxy.

PHP............................. A server-side scripting language designed for web development.

POST .... A HTTP Method for RESTful Services.

PPP .... Public-Private Partnership.

PUT .... A HTTP Method for RESTful Services.

Python.......................... A high-level programming language.

R .... A programming language and software environment for statistical computing and graphics.

RDF .... Resource Description Framework.

REST .... Representational State Transfer.

RESTful API .... A web service that was implemented using HTTP protocol and the constraints or principles of REST.

RPC .... Remote Procedure Call.

Ruby .... A dynamic, reflective, object-oriented, general-purpose programming language.

Scala .... A programming language.

SCIS .... Smart Cities Information System.

SMARTER TOGETHER…A lighthouse (SCC1) project funded by the European Union under the HORIZON 2020 programme.

SODA .... Socrata Open Data API.

SOLR........................... An open source enterprise search platform.

SpagoBI....................... An Open Source Business Intelligence suite.

SQLAlchemy............... Python SQL toolkit and Object Relational Mapper.

SW............................... Software.

Swift .... A programming language created for iOS, OS X, watchOS, tvOS and Linux development by Apple Inc.

TCO............................. Total Cost of Ownership.

TSV .... Tab-separated values.

Turtle .... Terse RDF Triple Language (a format for expressing data in the RDF data model).

UI................................. User Interface.

URL .... Uniform Resource Locator.

WFS............................. Web Feature Service (Open Geospatial Consortium Interface Standard).

Wilma .... An open source PEP Proxy (under MIT license) integrated with the FIWARE ecosystem.

WMS .... Web Map Service.

XML .... Extensible Markup Language.

9. REFERENCES

[1] CKAN documentation, https://media.readthedocs.org/pdf/ckan/latest/ckan.pdf
[2] CKAN developers, about, http://ckan.org/developers/about-ckan/
[3] NYC Open Data service based on Socrata, https://data.cityofnewyork.us/
[4] Quora, https://www.quora.com/Whats-the-difference-between-Socrata-and-CKAN-in-open-data-platform
[5] Socrata company, about, https://www.socrata.com/company-info/
[6] Socrata Open Data API (SODA), https://dev.socrata.com/
[7] Socrata web interface, publishing data, https://support.socrata.com/hc/en-us/articles/202949708-Append-and-replace-dataset-rows-wizard
[8] Socrata DataSync, automatic processes, http://socrata.github.io/datasync/
[9] Feature Manipulation Engine (FME) - https://en.wikipedia.org/wiki/FME_%28software%29
[10] Safe SW development company (FME) – http://www.safe.com/how-it-works/
[11] Socrata Open Data publisher API – https://dev.socrata.com/publishers/getting-started.html
[12] SCIS, nutshell, http://smartcities-infosystem.eu/scis-nutshell
[13] Concerto initiative, http://www.smartcities-infosystem.eu/concerto/concerto-initiative
[14] FIWARE, Wikipedia, https://en.wikipedia.org/wiki/FIWARE
[15] FIWARE, about, https://www.fiware.org/about-us/
[16] FIWARE, generic enablers (GE), catalogues, http://catalogue.fiware.org/
[17] FIWARE, license models, http://www.fiwareassociation.com/licenses-on-fiware-terra-incognita/
[18] Open Source Initiative, https://opensource.org/licenses/alphabetical
[19] CitySDK, about, http://www.citysdk.eu/about-the-project/
[20] CitySDK, Open311 API, http://www.citysdk.eu/citysdk-toolkit/using-the-apis/open311-api/
[21] CitySDK, Linked Data API, http://www.citysdk.eu/citysdk-toolkit/using-the-apis/linked-data-platform/
[22] CitySDK, Tourism API, http://www.citysdk.eu/citysdk-toolkit/using-the-apis/tourism-api/
[23] SMARTER TOGETHER VIENNA, https://www.wien.gv.at/bauen-wohnen/smartprojekt-simmering.html
[24] SMARTER TOGETHER Part. B sections 1-3