D4.2. Data Layer

Lead: Engineering - Ingegneria Informatica S.p.A.

Contributors: Cetaqua Andalucía

Date: June 2021

Public

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

Project deliverable

Project Number	**Project Acronym**	**Project Title**
1922	GOTHAM	Governance Tool for sustainable water resources allocation in the Mediterranean through stakeholder's collaboration. Towards a paradigm shift in groundwater management by end users.

Instrument:	**Thematic Priority:**
Research and Innovation Action	Mediterranean water co-operation

Title
D4.2 Data Layer

Contractual Delivery Date	**Actual Delivery Date**
30/06/2021	30/06/2021

Organisation name of lead contractor for this deliverable	**Document version**
Engineering – Ingegneria Informatica S.p.A.	V1.0

Dissemination level		**Deliverable Type**	
Public	X	Document, Report	
Confidential		Demonstrator	X

Authors (organisation)
ENG, CET

Reviewers (organisation)
ENG, CET

Abstract

The aim of this report is to document the software prototype (D4.2) developed in the context of Task 4.2 of the Gotham project. The result of the Task represents the Data Layer components of the overall GTool architecture (see D4.1). In such architecture, the Data Layer is responsible for collecting, ingesting and pre-processing the heterogeneous data that will be used by the Analysis Layer and the Service Layer to deliver the final results to the end-users. Ingested data are exposed

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

through well-defined APIs so that they can be accessed easily by the other components.
Keywords *GTool, data import, data pre-processing, API.*

Disclaimer

GOTHAM has been financed by the European Commission.

Project summary

The overarching objective of the GOTHAM project is to develop and validate a user-driven tool that enables effective groundwater governance to ultimately preserve the quantity and quality of this strategic resource in the Mediterranean basin. The GOTHAM Tool (GTool) uses an integrated methodological approach that targets optimal allocation of water resources from an environmental, social and economic perspective, including stakeholder knowledge, priorities and behaviour.

One of the main strengths of the tool is that it provides a common framework for collaboration and engagement of the different water users (mainly, agricultural communities but also municipal and industrial users), as well as other relevant stakeholders such as water producers/operators and regulator(s). The GTool will enable them to exchange information in order to reach the optimal water governance at each point in time as well as in future scenarios.

The concept of the proposed GTool targets effective groundwater governance for the improvement of the management and preservation of this essential and strategic resource. This effective groundwater management remains an important and complex challenge in the Mediterranean and elsewhere, but is essential to ensure long-term sustainable use of the resource.

In this regard, GOTHAM integrates multicriteria decision methods for stakeholder group decision making and social learning, and use socio- hydrological water balance framework as a theoretical foundation for water allocation to evaluate the dynamic balance between the societal and ecological systems in catchments. GOTHAM project presents a bottom-up decision-making approach inspired in this methodological framework.

GOTHAM project presents a scalable and user-specific tool for decentralising water resources management, using big data analysis. The proposed user-based tool leverages six analytical modules:

- The **water balance and water quality dynamics module** uses advanced investigation of the main aquifer formations and real-time monitoring (on site and distant), including preliminary analysis of the background hydrogeological and hydro-meteorological information to create a baseline.
- The **water availability and demand forecasting module** predicts different water scenarios and assess their impact on groundwater quality and quantity status using remote-sensing measurements to model agriculture water demand and assess water availability.
- The **Managed Aquifer Recharge (MAR) and aquifer remediation module** mobilises multicriteria analysis (QGIS environment), including hydrogeological, economic, and chemical (water quality) indicators as well as regulatory restrictions to evaluate the feasibility of MAR schemes.
- The **agro-economic module** simulates the effect of different economic instruments, such as water tariff structures, water markets contexts and incentives for water savings (water demand management) and assessing the economic use values and trade-offs between users in alternative resource allocation scenarios.
- The **user's engagement module** enables to fix water priorities (water boundary conditions) by water users, taking into consideration water resources to meet water demands.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

- The **optimised water allocation module** calculates the optimal mix of water sources satisfying their requirements

Gtool uses data visualisation techniques to deliver the results into customisable dashboards tailored for the needs of each stakeholder.

Broad outreach activities will take place in Europe, Lebanon and Jordan, therefore contributing to GOTHAM impact maximization.

The further development and exploitation (beyond the project) of the GTool will be done by CETaqua, both on B2B and B2C approaches.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

Table of content

Table of figures

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

1. Introduction

The aim of this report is to document the software prototype (D4.2) developed in the context of Task 4.2 of the Gotham project. The result of the Task represents the Data Layer components of the overall GTool architecture, which is depicted below and that has been defined in Task 4.1 (D4.1).

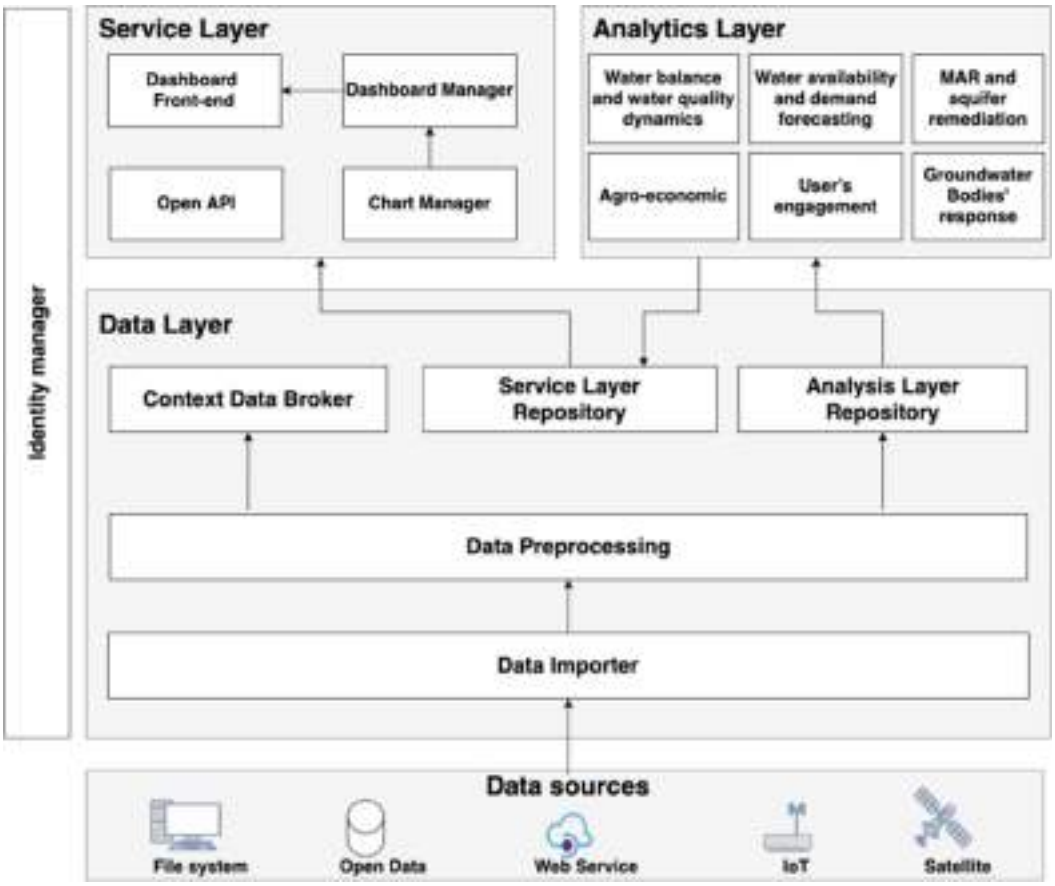

Figure 1 - System Architecture

In general terms:

- the GTool Data Layer will enable a smooth interfacing with the different and heterogeneous data sources of the water domain.
- the GTool Analytics Layer will provide multicriteria analysis (availability, demand, quality, cost) regarding water allocation scenarios and their impact on the environmental, economic and social sustainability of the water resources.
- the GTool Service Layer exposes, in different ways, data and results produced by the Analytics Layer and data imported from other sources.

The rest of the document describes in a technical way each component of the Data Layer, providing a general description, a table with some technical details and some indications for the installation.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

2. Software Modules

2.1. Docker

2.1.1. Description

Docker is an opensource software platform that allows to create, test and deploy applications. It gathers software into standardized units called Containers that offer everything needed for their proper execution, including libraries, system tools, code and runtime. It is also possible to distribute and recalibrate resources for an application in any environment, always keeping the executed code under control.

This is a native Linux technology but is available for other popular platforms. The most important feature is the isolation capacity of the processes. Isolating processes ensures that they can be run independently but simultaneously. This independence is the goal of Containers: the ability to run multiple processes and applications separately to make the most of the existing infrastructure while maintaining the level of security that would be guaranteed by the presence of separate systems (such as with virtual machines).

Using containers, resources can be isolated, services limited and processes started so that it is possible to have a completely private perspective of the operating system, with their own identifier, file system, and network interfaces. Multiple containers share the same kernel, but each of them can be forced to use a certain amount of resources, such as CPU, memory and I/O. Using Docker to create and manage containers can simplify the creation of distributed systems, allowing different applications or processes to work independently on the same physical machine or on different virtual machines. This allows a **Platform as a service (PaaS)** style of development.

The community version of Docker is released under the **Apache 2** license, while the Enterprise version is released with a proprietary license.

2.1.2. Technical Details

Type	Middleware
Programming Language	GoLang
Additional libraries	Docker Compose

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

2.1.3. Installation

The installation procedure depends on the operating system used and also changes depending on the version of the operating system in use. It is well described in the Docker official documentation[1].

2.2. Orion Context Broker

2.2.1. Description

As reported in D4.1, Orion Context Broker is an implementation of FIWARE's Publish/Subscribe Context Broker Generic Enabler and provides NGSI9 and NGSI10 interfaces. The component is responsible for managing and aggregating data from all sensors / devices of interest, using the Publish / Subscribe paradigm. Clients can access this data as entity attributes representing devices distributed in a territory. The broker therefore represents the hub where all IoT devices can be managed using different communication patterns. For example, a consumer (e.g. web application) can obtain data in different ways:

- By sending requests to the broker or directly to the producer;
- Subscribing to data updates that correspond to specific conditions (eg change of an entity);
- Searching for producers through standard API.

Consumers can also send commands to devices by updating the attributes related to the related entity commands, provided they have access rights for that operation.

Orion Context Broker is released under GNU Affero General Public License v3.0

2.2.2. Technical Details

Type	Middleware
Programming Language	C++
Additional libraries	Figway
Database	MongoDB

[1] https://docs.docker.com/

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

2.2.3. Installation

The installation procedure is simplified by the use of Docker technology. In particular, to be able to deploy the entire frontend environment, simply run the docker-compose up command in the folder that contains the docker-compose.yml (0) file.

2.3. Data Import and pre-processing

2.3.1. Description

This set of components includes both adapters that allow to import data from external data sources and dedicated scripts for pre-processing the ingested data in order to make it compatible with the expectations of the Analysis Layer. It is capable to handle time series or GIS files from the file system, from third-party services, from open data portals and from other sources. Depending on the actual necessities that emerged from the Analysis Layer, the data are stored in the Analysis Layer Repository or directly in the Service Layer Repository for its direct usage through visualizations. When data are imported in the Analysis Layer Repository, they are made available through standard APIs.

2.3.1.1. Data import and pre-processing for the Analytics Layer Repository

The data pre-processing component eventually will transform data coming from the acquisition stage into a standard and homogeneous format capable of describing all the entities and interactions of the variables assessed. To do this, the open-source workflow tool Airflow was used. Airflow is a platform to programmatically create, plan and monitor workflows. Pipelines are created as directed acyclic graphs (DAGs) of tasks and Airflow's scheduler executes tasks on an array of workers following specified dependencies. The pipelines are configured as code (Python), which allows them to be generated dynamically. It is also possible to simply define our operator by extending the libraries provided by the tool. Airflow has a rich user interface that makes it easy to view pipelines running in production, troubleshoot when necessary and monitor progress and thanks to its modular architecture it uses a queue of messages to orchestrate an arbitrary number of workers.

The files to be taken account from the different sources are two different types: text (.csv,.txt) and geographical (.tif). For this reason, there will be two different analyses to make and for each different web data sources there will be a different process. Therefore, for each source, we will have a different autonomous pipeline.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

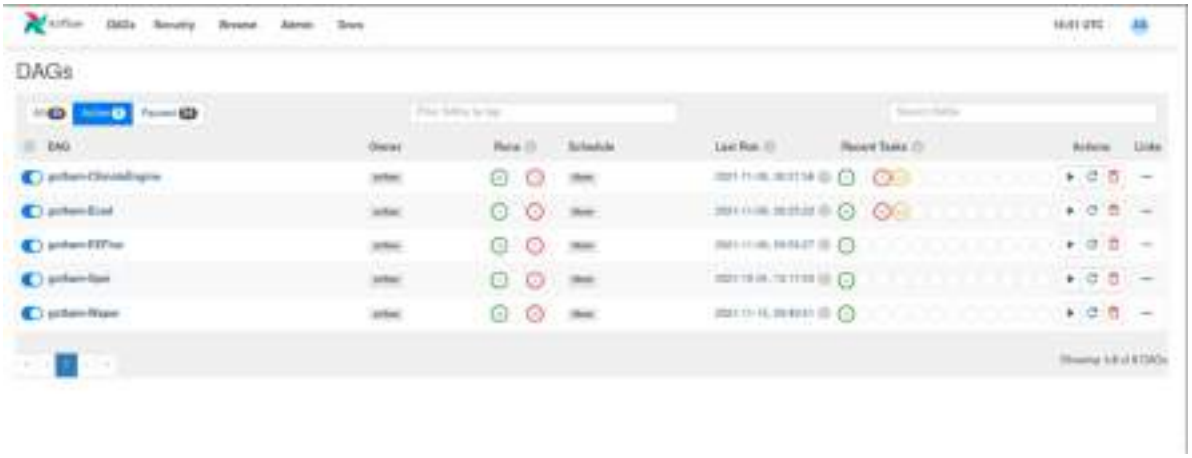

Figure 2 - List of DAGs created for data sources in Airflow

Once the pipelines have extracted the information from the individual files, they can move on the next task. It is necessary to distinguish between two different processes depending on the file under consideration: the timeseries data is inserted into a Postgres database and then each individual measurements are adapted into a FIWARE smart data model WeatherObserved object and sent via a POST request to the ContextBroker. For geospatial files the procedure is different, meta information (srs, bbox, height, width) is extracted and saved in the Postgres database, while the tiff file is inserted and made available via a GeoServer.

The following sections describe how data from different data sourced are ingested and pre-processed for their storage in the Analysis Layer Repository.

European Climate Assessment & Dataset (ECA&D) (https://www.ecad.eu)

The ECA dataset contains daily observation series from meteorological stations across Europe and the Mediterranean. Our purpose is to obtain measurements of parts of Spain and Lebanon. Once downloaded from the web portal provided, the files are presented as data archives. Within them, it is always possible to find files containing metadata (measuring station identifier, country code, lat, log, etc.), in addition to these, there is a different file for each measurement variable. All regions do not have the same variables, in fact Lebanon has only maximum temperature, minimum temperature and pressure amount, unlike Spain which has many more (cloud cover, global radiation, humidity, maximum temperature, mean temperature, minimum temperature, precipitation amount, sea level pressure, sunshine wind direction, wind gust and wind speed). Within each individual measurement file for a variable, in addition to a header with data usage documentation, each value is accompanied by the source identifier, the corresponding date and the precision of the measurement.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

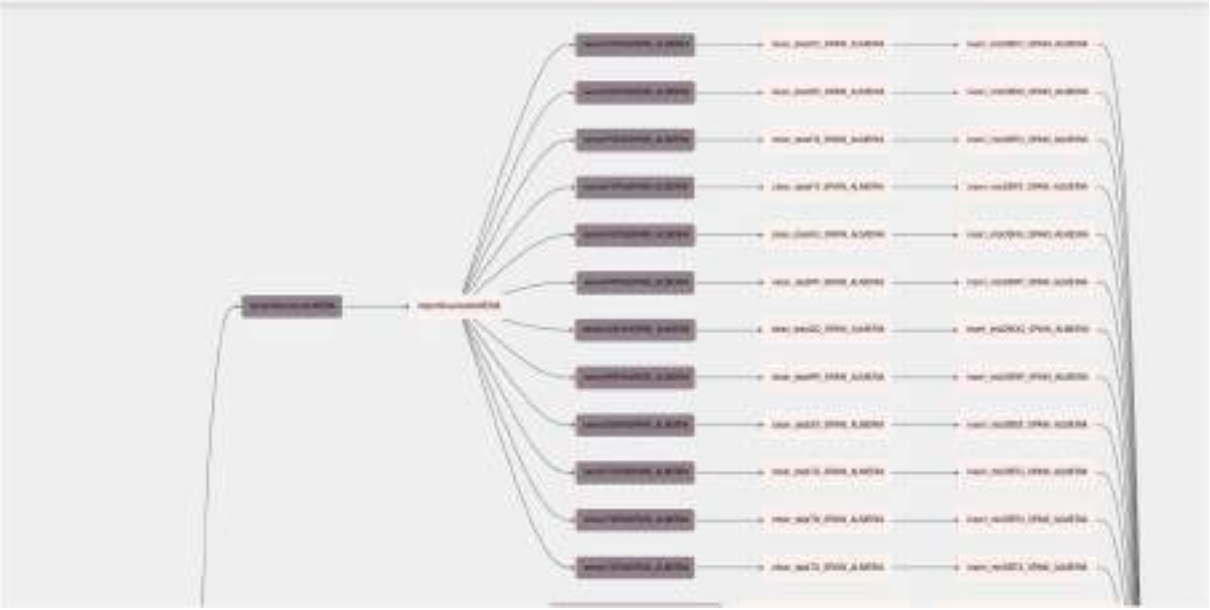

Figure 3 – Section of DAG relating to the ECAD source

The pipeline is dynamically constructed on the basis of the measuring stations to be considered and the variables present. Data must be downloaded manually and placed in the specific folder on the filesystem. So, the first task is to check whether the files exist and are present in the correct location. This check is performed with a FileSensor. Initially, a check is made to see if the metadata file is present, as without it, it would not be possible to extract all the meta-information and the execution of that branch of the pipeline is stopped. The check is repeated several times until the task is defined as failed and terminated if nothing is found. Subsequently, if it is detected, the information is extracted and inserted into the Postgres database, a source table is created for each analysed region. It will contain all the information related to the measuring station (source identifier, station identifier, source name, country code, latitude, longitude, station elevaton, element identifier, ecc...).

Successively, in parallel, a Python operator is created that checks if every file corresponding to a weather variable is present in the file system. If the file is detected, execution proceeds, otherwise the branch of that specific variable is interrupted and the rest continue.

Once the existence of the file is verified, the file is opened by removing all the header, measurements that considered inaccurate are removed (based on a field within the file), the dates present are formatted to be ingested by the db and the Context Broker without error, and finally the names of the columns to be inserted in the database table are created, such as the unit of measurement (measure identifiable in the header). The data pre-processed in this way are ready to be imported. For each file, it is first checked that the values imported into the db do not already exists and only then, if they do not exist or if there is more up-to-date data, can they be processed and imported. The obtained data are individually inserted into a FIWARE WeatherObserved object template in NGSI-LD format. It is necessary to specify that some measurements have been modified,

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

as the units of measurement used were not the same as those use within the FIWARE model (like sunshine variable which in the FIWARE standard is measured in l/m2, while in the files it is measured in 0.1mm). The object is populated with the metadata corresponding to that measurement and as a last step the attribute with the measured value is added. Only when all measurements have been entered into the ContextBroker by a POST request, the collected data are entered into a dedicated Postgres table. A table is created for each measured region and weather variable respectively (almeria_cc, almeria_tx, ecc...), and consist of the fields: the source identifier, the date of the individual measurement, the value with associated unit of measurement and the accuracy.

Finally, the pipeline ends with a DummyOperator. This operator does not perform any precise action, but specifies the policies for closing the workflow. If at least one of its previous tasks has been successfully completed, the workflow can be defined as successfully completed.

Consejo Superior de Investigaciones Científicas, SPEI Global Drought Monitor (https://spei.csic.es/map/)

To calculate the SPEI measure, the SPEI Global Drought Monitor tool was considered.

It provides near real-time information on drought conditions on a global scale, with a spatial resolution of 1 degree and a monthly temporal resolution. SPEI time scales between 1 and 48 months are provided. The dataset is updated during the first days of the following month based on the most reliable and up-to-date climate data sources.[https://spei.csic.es/map/maps.html#months=1#month=9#year=2021]

The main advantage of the SPEI Global Drought Monitor is therefore its near real-time nature, a feature that is best suited for drought monitoring and early warning purposes.

Once the geographical area has been selected on the web portal provided and the files downloaded, only one per region (Spain, Lebanon, Jordan), the workflow is executed. The files in this case are downloaded in .csv format, where each line contains the corresponding date and measurement for SPEI 1 to 48 and must be placed in the correct location on the filesystem.

Figure 4 – DAG related to CSIC data source

Each branch of the pipeline is created dynamically based on the region entered. The first task in this case check whether the downloaded .csv file is present in the corresponding folder on the filesystem. If the file is not found, that branch will terminate execution. Ensuring the existence of the file, the csv is analysed using the Python Pandas library. Null values are removed from the file and all measurements, except SPEI 12, SPEI 24 and the reference date which are saved on a monthly basis, are removed. The dates are formatted so that they can be useful for subsequent tasks, from text format to numeric format.

Once a DataFrame has been created using the Pandas library with the remain data, it is checked to see if any values are already present in the db. If the file has never been analysed or only if there are more recent values, the data processing continues. After this check the date is transformed into the format accepted by the ContextBroker and the database, the meta-information is extracted, which in this case is not defined by the web portal but is defined manually (e.g., the coordinates drawn on the map to extract the information). Following these steps, each monthly measurement is adapted to a template of a FIWARE smart data model WeatherObserved object to which the value for measuring SPEI is added, and sent via POST request to the ContextBroker in NGSI-LD format. The meta-information in this case will be much less (address, dataProvider, dateObserved, location and source), as there is less information to be extracted or extracted related to the data. Only after the information has been entered into the Broker, it can be entered into the Postgres database. A table is created for each analysed region (e.g., spain_spei, ecc...), which contains three columns: date, spei12 and spei24. Finally in order to be able to store metadata and use it when the data are exposed, a different table is created for each geographical region. For each of these the following is stored: nation, country code and coordinates used to draw the area of interest on the web portal.

The workflow ends with a DummyOperator that defines the pipeline as successfully terminated if at least one task preceding it ends successfully.

Earth Engine Evapotranspiration Flux (EEFlux) (https://eeflux-level1.appspot.com/)

Eeflux is a version of METRIC (Mapping Evapotranspiration at high Resolution with Internalized Calibration) that operates on the Google Earth Engine system. EEFlux processes individual Landsat scenes from any period from 1984 through present and for nearly every land area on the Globe. To apply EEFlux, the user can specify start and stop dates to search in the date window and then move the orange pin on the Google map into the area of interest. When the 'search image' button is pressed, EEFlux will search the Google archive for those Landsat images that reside within the date range and that cover the area identified by the pin. For areas that are covered by two paths, images from both paths will be presented. EEFlux will then provide a list of images for the location and permit the user to select one image to process. The list of images available includes an assessment of percent cloud cover for the entire scene. [https://eeflux-level1.appspot.com/]

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

The only variable to take into account from this source is DEM – Digital Elevation Model in meters. The files returned from this source are in .tif format.

As in the previous cases, the activities are divided and carried out in parallel according to the areas under consideration. In contrast to what has been said so far, the procedure will be different. The objective in these cases, when dealing with geospatial elements, is to extract all the information from the file and insert it into GeoServer.

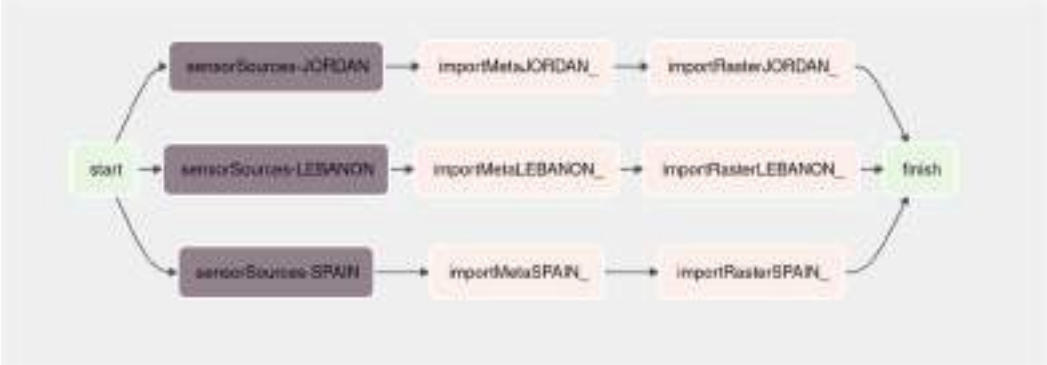

Figure 5 – DAG related to EEFlux data source

Once the tiff files had been downloaded and placed on the filesystem, the first task developed was a FileSensor. This has the simple purpose of accessing the position defined within the filesystem and checking whether the downloaded tiff file is present. Obviously, if it is not present, there is no sense in continuing processing and that branch is aborted. Knowing the precise location of the files, the next task has the role of accessing that folder and for each file within it extracting all available information.

The information is extracted from each of the tiff files and is as follows: bbox, srs, height, width, the name of the file, and the variable to which it refers. The name of variable is extracted from the name of the tiff file. The web portal downloads the image by assigning them a name with a fixed syntax in which it includes the variable in question, and therefore knowing the way the files are saved it will be possible to extract the name. While the remaining information is extracted using the Python library OsGeo useful for manipulating raster data. Once this information has been extracted, it must be stored on the database. In this case, only one table is created for meta-information (source_eeflux), in which each row corresponds to a file. In fact, the columns of this table are the defined information that is extracted (value name, bbox, srs, height, width, name of tiff file). Before analysing each file to extract meta-information, it is checked whether the analysed file already exists in the database. In fact, the saved data is only updated if it is the first time that file with the associated variable has been analysed, or if it is a different file for a variable already stored in the database.

Once this is done, it is necessary to load the images as stores and expose the layers in GeoServer. To do this, knowing the position of each file and its name, the geo.Geoserver library is used. Geoserver-

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

rest is the open-source library written in python, with MIT License and constantly evolving. The geoserver-rest package is useful for the management of geospatial data in GeoServer. This package is useful for creating, updating and deleting GeoServer workspaces, stores, layers, and style files. Using this library it is necessary to define, after defining the connection string to GeoServer, the name to be assigned to the layer, the workspace and the position of the raster file to be loaded.

The workflow ends with a DummyOperator and will be successfully terminated if at least one of the branches completes successfully.

Climate Engine (https://app.climateengine.org/climateEngine)

Climate Engine is a mission-driven company that helps organisations reduce their exposure to climate risk and meet their sustainability mandates. It provides a 'no-code' user interface to Google Earth Engine to quickly and easily visualise various EO processes and variables. In the interface provided, it will be possible to select a product dataset to access from the Google Cloud. [https://climateengine.com/about/, https://climateengine.com/research-app/]

Once the product, the reference database and the variable to be analysed have been selected, it is possible to load the layer and download a zone of defined coordinates with the option of deciding on the time period for the search. This data source, returns archives of raster files (.tif) which must be analysed as in the previous case.

Figure 6 – DAG related to ClimateEngine data source

The basic procedure for analysing raster files remains the same. Once the file archives have been downloaded, they are saved in the appropriate folder. To make the files reachable by the task, it is first necessary to extract them from the compressed files. To do this, a special task has been created with the assignment of extracting the tiff files from within the archives. Once the file has been extracted, if an older file exists for that variable within the folder, it is replaced by the most recent one, otherwise it is inserted by adding it to the others.

The workflow always takes into account the various geographical regions, so a branch will be created automatically for each one. After preparing the file within the folder, a FileSensor checks that there are files to be processed, if it does not find any or if there is an error in locating the folder, the flow of that branch stops. Knowing the location of the files, via the OSGeo library all meta information is

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

extracted. The extracted information will then be placed in a single source_climateengine table, where each row corresponds to a file and associated variable. Data is only written to the database if the variable has not yet been analysed or if the saved data comes from a different file than the one just read. After this, each individual raster file is inserted into the GeoServer as a store and exposed as a layer via the Python library geo.Geoserver.

The workflow ends with a DummyOperator, which serves no specific purpose ,but which will make the pipeline appear to have ended successfully if at least one previous task ends successfully.

FAO Water Productivity Open-access portal (WAPOR) (https://wapor.apps.fao.org/home/WAPOR_2/1)

The programme on WaPOR, FAO's portal to monitor Water Productivity through Open-access of Remotely sensed derived data, assists countries in monitoring water productivity, identifying water productivity gaps, proposing solutions to reduce these gaps and contributing to a sustainable increase of agricultural production. At the same time, it takes into account ecosystems and the equitable use of water resources, which should lead eventually to an overall reduction of water stress. By providing near real time pixel information, WaPOR opens the door for service-providers to assist farmers in obtaining more reliable yields and improving their livelihoods. [https://www.fao.org/in-action/remote-sensing-for-water-productivity/overview/about-the-programme/en/]

Unlike the other sources that return one image per variable, for this one it is necessary to download a series of image from 2009 to the present on two variables, precipitation and evapotranspiration. Furthermore, it will not be a possible to choose a geographical region as it only provides information for African and middle east countries. After registered on the portal, it will be possible to generate an API token, useful for using the exposed API to retrieve images. The result of the download will be raster images (.tif).

Figure 7 – DAG related to Wapor data source

As it is not possible to subdivide the workflow by regions to be considered, it is subdivided by variables. The first task aims to download all the files in their database from the exposed API. After authenticating and obtaining the Access Token through the API Token, it will be possible to request the image date by date. For the variable precipitation, the information is saved daily, while for the variable evapotranspiration it is saved monthly. Before sending the request for a specific file, it is

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

necessary to check that it has not already been downloaded, so that the most up-to-date and not overwritten data will always be in the folder.

Immediately after downloading the files via the API, a FileSensor is used to check whether at least one file has been downloaded and is present in the correct folder; if this is not the case, execution stops. Ensure that the files are present, knowing the precise location, it is possible to analyse them with the library osGeo and extract all metadata. The extracted information (bbox, srs, height, width, the name of the file, and the variable to which it refers) is inserted in a row of the table containing the meta-information on the database, only if the corresponding file has not yet been analysed before. A useful information in this case will also be the date to which the image refers. This is why when the variable name is saved in the metadata table, the date corresponding to that image is also attached to it. The date is extracted from the name under which the file is saved by the WAPOR portal.

The files must not be overwritten as they are a time series, which means that there cannot be two identical files with the same date. The Python library gsconfig is used to check whether the layer to be created already exists or not. Obviously, if the same layer already exists on the GeoServer, it is not inserted again. The raster files are loaded onto the GeoServer with the geo.Geoserver library.

Also in this workflow, the flow ends with a DummyOperator that defines the pipeline as successfully terminated if at least one branch is successfully terminated.

2.3.1.2. Data import and pre-processing for the Service Layer Repository

In this section we document the import procedures that are available for making available data directly to the Service Layer Repository, so that they can be used for creating visualizations without needing previous analysis.

Import a time series from the local filesystem.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

Figure 8 – Dashram screen: add a dataset from the filesystem.

Dashram provides the ability to import timeseries from its filesystem. Two types of files are supported for uploading: xlsx and csv.

For both possibilities, it is possible to load the file from the graphical interface and complete the required set of parameters. In addition to loading the file, it is also necessary to enter the name of the table that will be created in the database from the data. Obviously, it will also be necessary to choose which database to add the table to and what behaviour to follow if the name defined indicates a table that is already present in the database. In addition to the mandatory parameters, there are fields to indicate the scheme, an indicator to count the number of rows to be discarded if there is a header, how to format dates, which character to consider as a null element, etc... The difference between the csv upload page and the xlsx upload page is the specification of the delimiter in the csv which is not present in the Excell file. Once the data has been loaded, it can be found in the set of available datasets. Because they were created with data loaded from the filesystem, the dataset is defined as physical.

Import a time series from an open data portal

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

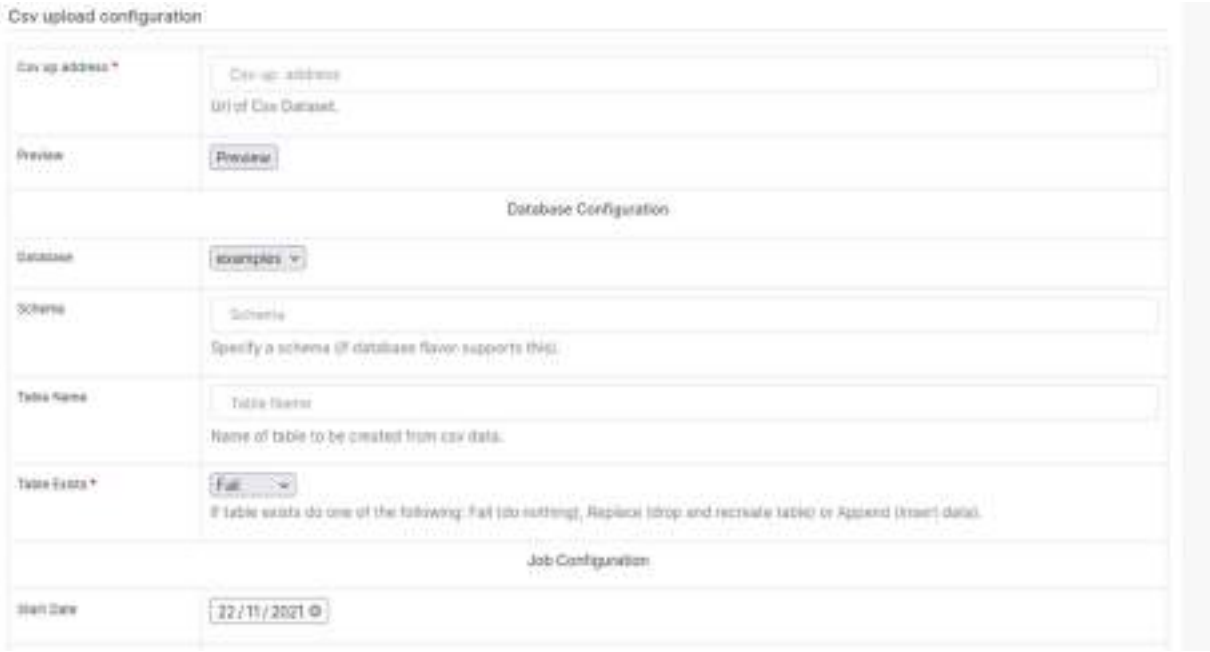

Figure 9 – Dashram screen: add a dataset from the open data portal.

To import data from open data portals, Dashram divides the import process into several blocks: Csv upload configuration, Database Configuration, Job Configuration and Advanced Configuration.

In the first block it is possible to define the URL of the request or the URL of the csv dataset with the ability to preview the data. In the database part it is possible to choose the schema to be assigned, the database in which to insert the table created from the data received and the name to be assigned to it. The job configuration part allows, if required, data collection to be carried out periodically. In fact, it is possible to define the start date on which to extract the data and various information required for this purpose. In the last section, there is information useful for storing and manipulating data. In fact, it is possible to define the delimiter character, whether there are header lines to be skipped, how dates should be formatted and therefore used, how many lines to count, etc...

Import a time series using a REST service.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

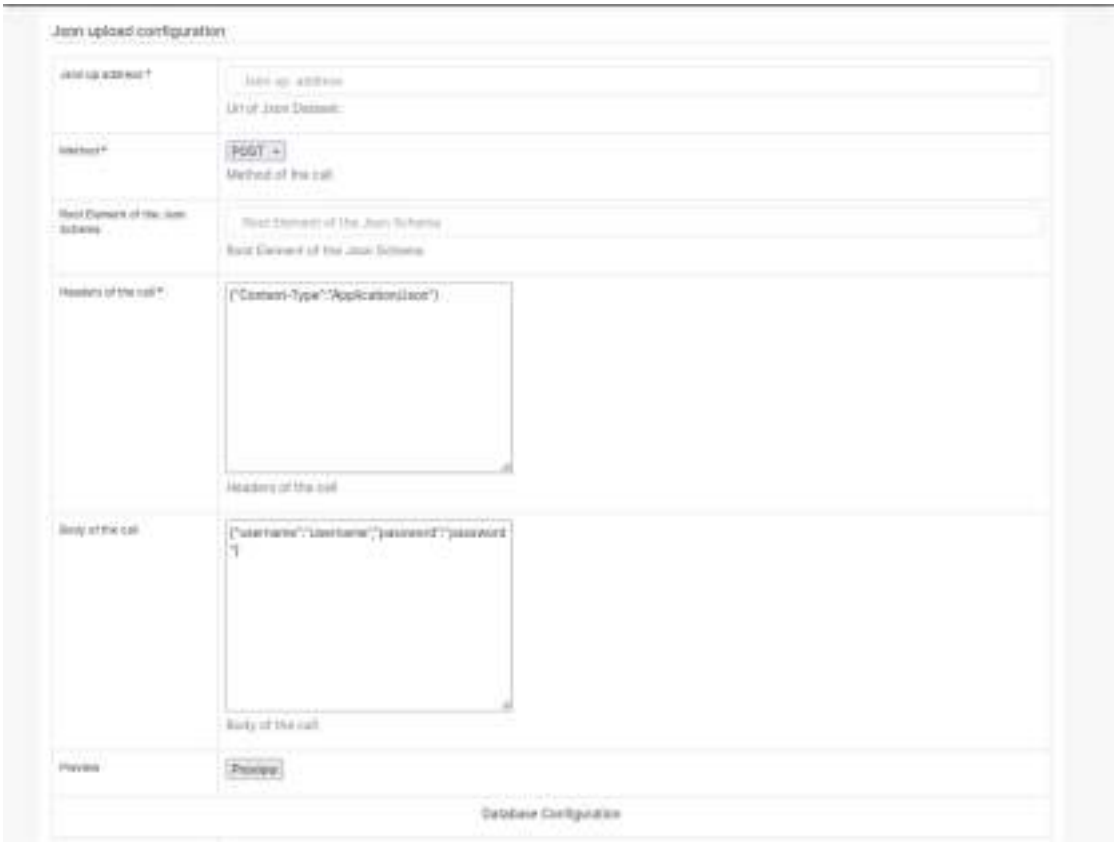

Figure 10 – Dashram screen: add a time-series using REST service.

Dashram also allows information to be entered through the use of REST services.

In this case the import of information is divided into blocks: Json upload configuration, Database Configuration, Job Configuration, Advanced Configuration. The only section that changes from the previous ones is the first one, in which it is necessary to insert not only the URL of the request or the URL of the json dataset, but also the method used to make the request (POST or GET), the headers and the body of the request.

2.3.2. Technical Details

Type	Middleware
Programming Language	Python
Additional libraries	Pandas, arirflow, requests, os, geo, sqlalchemy, osgeo, zipfile, json, gsconfig, datetime
Database	Postgres

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

2.3.3. Installation

The installation procedure is simplified by the use of Docker technology. In particular, in order to deploy the entire frontend environment, simply run the command docker-compose up in the folder that contains the file docker-compose.yml (0).

This will activate all modules related to the operation of Airflow.

2.4. Analysis Layer Repository

2.4.1. DBMS

2.4.1.1. Description

PostgreSQL is an object relational database (ORDBMS) and originated from the POST INGRES project of the University of California at Berkeley. It was started in 1986 under the leadership of Michael Stonebraker and sponsored by the Defense Advanced Research Project Agency (DARPA) and the National Science Foundation (NSF). PostgreSQL is not driven by any vendor and consequently has no customer list to satisfy. This means that a feature will only be implemented in PostgreSQL if it makes technical and scientific sense. PostgreSQL is a non-commercial, all volunteer, free software project, and as such there is no formal list of feature requirements required for development.

Below we list some of the features that characterize it:

- Possibility of complex queries
- Foreign keys for linking data from two tables
- Triggers that are automatically activated upon entry and check, confirm, modify, delete or alternatively insert the reference data
- Updatable views
- Complete transaction concept
- Multiversion Concurrency Control (MVCC) to efficiently perform simultaneous access to the database

Postgres is used in numerous industries and scenarios. It stands out as a first-class foundation for safely managing a wide range of applications. For example, for Online Banking software thanks to the integrated transaction concept and the support of MVCC (Multiversion Concurrency Control) for efficient simultaneous access performance. Analysis programs such as Matlab or R are also often used in conjunction with this database. Thanks to the PostGIS extension, which provides hundreds of functions for working with geospatial data, Postgres also convinces when working with geospatial data and geographic data.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

PosgreSQL is also required as a solution for web projects: the object relational system works with various modern frameworks such as Django, Node.js or Ruby on Rails and supports classic web languages such as PHP. In addition, thanks to the support for synchronous and asynchronous replication, the stored data is easily distributed among multiple servers to ensure high reliability and minimum access times to critical data.

PostgreSQL is released with a proprietary but free license similar to the BSD or MIT license available at this url The PostgreSQL Licence[2].

In this case, the Postgres database is used to hold and store all timeseries measurement information and all useful metainformation from the raster data.

2.4.1.2. Installation

The installation procedure is simplified by the use of Docker technology. In particular, to be able to deploy the entire frontend environment, simply run the docker-compose up command in the folder that contains the docker-compose.yml (0) file.

2.4.2. Fileserver

2.4.2.1. Description

Spatial Fileserver. Used to store raster files and shape files. All software was developed using docker container technology.

For the moment, therefore, the images that are downloaded from the sources are inserted directly into the filesystem of the docker container, in the folder where the code used by Airflow to generate the DAGs is placed. A folder is created for each data source. According to the differentiation that that source makes, it will be possible to find a first level of folders according to the geographical distinction, if the source returns data for different areas. Then find a distinction by variables, according to the variables treated.

2.4.3. Data manager

Datasource manager. At the top of the previous three components there is the datasource manager that is responsible to manage the requests from the Analytics Layer for providing datasources used to make analyses. Datasource manager also enforce the access rights permission to the datasources.

[2] https://www.postgresql.org/about/licence/

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

The data Import and pre-processing module has the task of preparing and saving data to make them available to the data manager.

When all textual information about time series is entered into the Postgres database and sent to a Context Broker and all raster images are exposed via layers in a GeoServer, they can be made available to requests. In order to expose this stored data, APIs were created using the web framework for developing RESTful APIs in Python, FastApi. The APIs created are, like the different types of data that can be obtained from the sources received, two separate ones to make requests for textual data and geographical data.

Figure 11 – APIs created with FastApi to return collected data.

The APIs can be used on command line, but in addiction to this possibility, a web page is provided where they are documented. For each call, the form, syntax and all available values as defined.

The first API was developed in order to be able to extract information related to the time series meteorological measurements stored in the Postgres database. However, in order to receive the information, it is necessary to input several parameters. The source, which indicates the source to which the request refers. At the moment it is ecad and csic. The region of reference, as these sources collect information for different geographical territories, so this needs to be defined. In addition to the different areas, the sources return data for different variables, so it is necessary to specify one or more of them related to that source. And finally, it is necessary to define the start and end date of the request, because each data set collects different time periods. As a first step, a check is made for each value entered; if it does not match, an error message is returned and the request does not proceed. Once the checks have been passed, all the information will be available to make a query to the Postgres database. In addition to extracting all the required measurement, the metadata available in the database will also be extracted for each source. Obtained both measurements and metadata, this information will populate a FIWARE smart data model WeatherObserved object. The result will be an array of these object returned to the requester. There is a non-mandatory attribute in the request structure that is used to define how to return the data, as the FIWARE standard defines that the data may be in normalised or keyvalue form. This attribute has the default value

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

False, so the objects are in normalised form and only if it is modified, it will change the format of the output.

The second API delivered, on the other hand, aims to provide the raster images collected previously. The parameters common to all requests are the data source to be queried, the desired image output format and the variable required for that source. Once the data source from which to request images has been defined, checks are carried out on the other parameters entered. If there are any incorrect or missing data, an error message will be returned. In addition to the mandatory parameters for each source, there are parameters that only become mandatory depending on the source chosen. In fact, there are some data sources that return one image per variable (Climate engine, Eeflux), while other sources return a time series of tiff files (Wapor), which is why the parameters start date and end date of the request will only become mandatory for the last type. If values are entered when not required, they will not be considered. The same applies to the places covered by the source, if the given source does not have different zones to choose from, that parameter becomes not required. The API service exposed by GeoServer is used to return the required images. The WMS protocol is used, which provides a standard interface for requesting a geospatial map image. Using the WMS protocol and the API provided by GeoServer, various settings can be defined when requesting each layer, such as the bbox, srs, height and width of the requested image. This information was extracted earlier in the pre-processing phase and will therefore be retrieved from the Postgres database. The WMS protocol provides images in various formats, from tiff to Png, which is why one of the mandatory parameters to specify is the format of the data to be returned. This information must be entered by means of acronyms, which are documented on the web page of the API itself. Once the entered parameters have been checked, the dates entered are formatted in the correct format accepted for the request, the name of the layer to be extracted is defined and the WMS request to be made to the GeoServer is constructed. When the images are obtained, they are stored inside the docker container, after that they are encoded in base64 and then transformed into a string in order to be returned to the requester. In addition to the image strings, the name of the file and the chosen format will also be returned, useful information when decoding. Once the encoded image has been returned, the tiff file obtained from the request and placed into the container will be deleted.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

2.4.3.1. Geoserver

2.4.3.1.1. Description

Geoserver is responsible for sharing geospatial data. It is designed for interoperability; it publishes data from any major spatial data source using open standards.

It implements industry standard OGC protocols such as Web Feature Service (WFS), Web Map Service (WMS), and Web Coverage Service (WCS). Additional formats and publication options are available as extensions including Web Processing Service (WPS), and Web Map Tile Service (WMTS).

The GeoServer uses the WMS protocol to make various requests available: GetCapabilities, GetMap, GetFeatureInfo etc... For our purpose we use only the GetMap operation. The core parameters specify one or more layers and styles to appear on the map, a bounding box for the map extent, a target spatial reference system, and a width, height, and format for the output.

The response is a map image, or other map output artifact, depending on the format requested. GeoServer provides a wide variety of output formats, described in WMS output formats.

GeoServer is distributed under the GNU General Public License Version 2.0

2.4.3.1.2. Installation

The installation procedure is simplified by the use of Docker technology. In particular, to be able to deploy the entire frontend environment, simply run the docker-compose up command in the folder that contains the docker-compose.yml (0) file.

2.5. Service Layer Repository

2.5.1. DBMS

2.5.1.1. Description

This component aims to receive information from the Analytics Layer and make it available in graphical form via charts and dashboards. The Dashram tool is used to do this.

The Dashram tool used as a Service Layer repository allows one or more relational databases to be inserted via its UI. From the dedicated page, it is possible to insert the connection link to the database, and it is also possible to test the connection itself to see if it works properly.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922

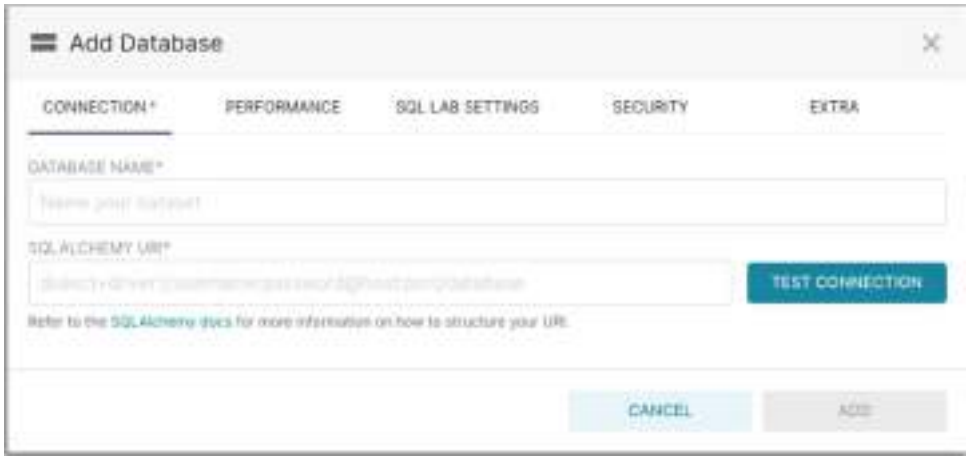

Figure 12 – Dashram screen: add a database.

In addition to this it is possible to define the performance, define the duration (in seconds) of the cache timeout for the graphs in this database. Define JSON strings to insert additional connection configurations, choose whether to expose the database in the sql editor part and finally it is possible to insert extra configurations relating to the schema, metadata, version, etc...

Once the database has been added, it can be seen in the list along with all the others added previously, where some details can be previewed.

A way to retrieve raster images from the Service Layer Repository and display them is under development.

2.5.1.2. Installation

The installation procedure is simplified by the use of Docker technology. In particular, to be able to deploy the entire Service Layer Repository, simply run the docker-compose up command in the folder that contains the docker-compose.yml (0) file.

GOTHAM is part of the PRIMA Programme supported by the European Union. The PRIMA programme is supported under Horizon 2020 the European Union's Framework Programme for Research and Innovation. Grant Agreement number: 1922